Arm® Cortex®-M

32位微控制器

NuMicro® 家族
M480系列
技术参考手册

The information described in this document is the exclusive intellectual property of Nuvoton Technology Corporation and shall not be reproduced without permission from Nuvoton.

Nuvoton is providing this document only for reference purposes of NuMicro® microcontroller based system design. Nuvoton assumes no responsibility for errors or omissions.

All data and specifications are subject to change without notice.

For additional information or questions, please contact:Nuvoton Technology Corporation.

www.nuvoton.com

目录

1 概述 35

2 特性 36

3 料号信息 48

3.1 封装类型 48

3.2 M480 系列选型指南 49

3.2.1 M481 基础系列 (M481xIDAE) 49

3.2.2 M481 基础系列 (M481xGCAE / M481xE8AE) 50

3.2.3 M482 USB FS OTG 系列 (M482xIDAE) 51

3.2.4 M482 USB FS OTG 系列 (M482xGCAE / M482xE8AE) 52

3.2.5 M483 CAN 系列 (M483xIDAE) 53

3.2.6 M483 CAN 系列 (M483xGCAE / M483xE8AE) 54

3.2.7 M484USB HS OTG系列 55

3.2.8 M485 Crypto 系列 56

3.2.9 M487 Ethernet 系列 57

3.3 M480 选型 58

4 引脚配置 59

4.1 引脚配置 59

4.1.1 QFN-33引脚图 59

4.1.2 LQFP-48引脚图 (0/1 USB FS) 60

4.1.3 LQFP-64引脚图 (0/1 USB FS) 61

4.1.4 LQFP-64引脚图 (0/1 USB FS with V_{BAT}) 62

4.1.5 LQFP-64引脚图(1 USB HS) 63

4.1.6 LQFP-64引脚图(USB FS + USB HS) 64

4.1.7 LQFP-128引脚图(1 USB FS) 65

4.1.8 LQFP-128引脚图(1 USB FS with V_{BAT}) 66

4.1.9 LQFP-128引脚图(USB FS + USB HS) 67

4.1.10 LQFP-144引脚图 68

4.2 M48xxGAAE/M48xxIDAE 管脚描述 69

4.2.1 M481 系列管脚描述 69

4.2.2 M482系列管脚描述 88

4.2.3 M483系列管脚描述 106

4.2.4 M484系列管脚描述 137

4.2.5 M485系列管脚描述 166

4.2.6 M487系列管脚描述 197

4.2.7 多功能管脚表 232

4.2.8 多功能管脚表以GPIO排列 260

4.3 M48xxE8AE/M48xxGCAE 管脚描述 294
4.3.1 M481 系列管脚描述 294
4.3.2 M482系列管脚描述 313
4.3.3 M483系列管脚描述 344
4.3.4 多功能管脚表 374
4.3.5 多功能管脚表以GPIO排列 399
5 框图 429
5.1 M480框图 429
6 功能描述 431
6.1 Arm® Cortex®-M4F内核 431
6.2 系统复位 434
6.2.1 概述 434
6.2.2 系统复位 434
6.2.3 系统电源分配 439
6.2.4 电源模式和唤醒源 442
6.2.5 电源模式转换 446
6.2.6 系统内存映射 446
6.2.7 SRAM 内存结构 449
6.2.8 总线矩阵 452
6.2.9 HIRC 自动校准 452
6.2.10寄存器锁定控制 453
6.2.11寄存器映射 456
6.2.12寄存器描述 458
6.2.13系统定时器(SysTick) 519
6.2.14系统定时器控制寄存器映射 520
6.2.15可嵌套中断向量控制器(NVIC) 524
6.2.16系统控制寄存器 559
6.3 时钟控制器 568
6.3.1 概述 568
6.3.2 时钟发生器 572
6.3.3 系统时钟和SysTick 时钟 573
6.3.4 外设时钟 574
6.3.5 掉电模式时钟 575
6.3.6 时钟输出 575
6.3.7 USB时钟源 576
6.3.8 寄存器映射 577
6.3.9 寄存器描述 579
6.4 真随机数发生器(TRNG) 632

6.4.1 概述 632
6.4.2 特征 632
6.4.3 框图 632
6.4.4 基本配置 632
6.4.5 功能描述 632
6.4.6 寄存器映射 634
6.4.7 寄存器 描述 635
6.5 存储控制器 (FMC) 639
6.5.1 概述 639
6.5.2 特征 639
6.5.3 框图 641
6.5.4 功能描述 644
6.5.5 寄存器映射 703
6.5.6 寄存器描述 705
6.6 通用 I/O (GPIO) 739
6.6.1 概述 739
6.6.2 特征 739
6.6.3 框图 740
6.6.4 基本配置 740
6.6.5 功能描述 741
6.6.6 寄存器映射 744
6.6.7 寄存器描述 748
6.7 PDMA 控制器 (PDMA) 766
6.7.1 概述 766
6.7.2 特征 766
6.7.3 框图 766
6.7.4 基本配置 766
6.7.5 功能描述 767
6.7.6 寄存器映射 775
6.7.7 寄存器描述 777
6.8 定时器控制器(TMR) 812
6.8.1 概述 812
6.8.2 特性 812
6.8.3 框图 813
6.8.4 基本配置 818
6.8.5 定时器功能描述 819
6.8.6 PWM功能描述 824
6.8.7 寄存器映射 841
6.8.8 寄存器描述 846

6.9 看门狗定时器(WDT) 890
6.9.1 概述 890
6.9.2 特性 890
6.9.3 框图 890
6.9.4 基本配置 890
6.9.5 功能描述 891
6.9.6 寄存器映射 894
6.9.7 寄存器描述 895
6.10 窗口看门狗定时器(WWDT) 900
6.10.1概述 900
6.10.2特性 900
6.10.3框图 900
6.10.4基本配置 900
6.10.5功能描述 901
6.10.6寄存器映射 905
6.10.7寄存器描述 906
6.11 实时时钟(RTC) 911
6.11.1概述 911
6.11.2特性 911
6.11.3框图 912
6.11.4基本配置 912
6.11.5功能描述 913
6.11.6寄存器映射 922
6.11.7寄存器描述 924
6.12 EPWM发生器和捕获定时器(EPWM) 957
6.12.1概述 957
6.12.2特性 957
6.12.3框图 958
6.12.4基本配置 963
6.12.5功能描述 965
6.12.6寄存器映射 1001
6.12.7寄存器描述 1008
6.13 基本 PWM 发生器和捕获定时器(BPWM) 1087
6.13.1概述 1087
6.13.2特性 1087
6.13.3框图 1088
6.13.4基本配置 1090
6.13.5功能描述 1092

6.13.6寄存器映射 1107
6.13.7寄存器描述 1110
6.14 正交编码接口(QEI) 1143
6.14.1概述 1143
6.14.2特性 1143
6.14.3框图 1143
6.14.4基本配置 1144
6.14.5功能描述 1145
6.14.6寄存器映射 1153
6.14.7寄存器描述 1154
6.15 增强型输入捕捉定时器(ECAP) 1164
6.15.1概述 1164
6.15.2特性 1164
6.15.3框图 1164
6.15.4基本配置 1165
6.15.5功能描述 1166
6.15.6寄存器映射 1171
6.15.7寄存器描述 1172
6.16 UART接口控制器(UART) 1183
6.16.1概述 1183
6.16.2特性 1183
6.16.3框图 1184
6.16.4基本配置 1187
6.16.5功能描述 1193
6.16.6寄存器映射 1217
6.16.7寄存器描述 1219
6.17 以太网控制器(EMAC) 1256
6.17.1概述 1256
6.17.2特性 1256
6.17.3框图 1257
6.17.4基本配置 1257
6.17.5功能描述 1258
6.17.6DMA描述符数据结构 1260
6.17.7寄存器映射 1274
6.17.8寄存器描述 1277
6.18 智能卡主机接口(SC) 1329
6.18.1概述 1329
6.18.2特征 1329

6.18.3框图 1329
6.18.4基本配置 1331
6.18.5功能描述 1334
6.18.6寄存器映射 1346
6.18.7寄存器描述 1347
6.19 I²S控制器(I²S) 1373
6.19.1 概述 1373
6.19.2 特征 1373
6.19.3 框图 1373
6.19.4基本配置 1373
6.19.5功能描述 1374
6.19.6寄存器映射 1385
6.19.7寄存器描述 1386
6.20 SPI接口(SPI) 1404
6.20.1概述 1404
6.20.2特征 1404
6.20.3框图 1404
6.20.4基本配置 1405
6.20.5功能描述 1408
6.20.6时序图 1425
6.20.7编程示例 1426
6.20.8寄存器映射 1429
6.20.9寄存器描述 1430
6.21 QSPI接口 (QSPI) 1453
6.21.1概述 1453
6.21.2特征 1453
6.21.3框图 1453
6.21.4基本配置 1454
6.21.5功能描述 1456
6.21.6时序图 1473
6.21.7编程示例 1475
6.21.8寄存器映射 1477
6.21.9寄存器描述 1478
6.22 SPI同步串行接口控制器(SPI主机模式) 1492
6.22.1概述 1492
6.22.2特性 1492
6.22.3框图 1493
6.22.4基本配置 1493

6.22.5功能描述 1494
6.22.6SPI flash控制器在AMBA 系统中映射地址 1504
6.22.7寄存器映射 1505
6.22.8寄存器描述 1506
6.23 I²C接口 1526
6.23.1概述 1526
6.23.2特性 1526
6.23.3框图 1527
6.23.4基本配置 1527
6.23.5功能描述 1529
6.23.6寄存器映射 1552
6.23.7寄存器描述 1553
6.24 USCI – 通用串行接口 1577
6.24.1概述 1577
6.24.2特性 1577
6.24.3框图 1577
6.24.4功能描述 1577
6.25 USCI – UART模式 1588
6.25.1概述 1588
6.25.2特性 1588
6.25.3框图 1588
6.25.4基本配置 1589
6.25.5功能描述 1590
6.25.6寄存器映射 1599
6.25.7寄存器描述 1600
6.26 USCI – SPI模式 1623
6.26.1概述 1623
6.26.2特性 1623
6.26.3框图 1624
6.26.4基本配置 1624
6.26.5功能描述 1626
6.26.6寄存器映射 1639
6.26.7寄存器描述 1640
6.27 USCI – I2C模式 1663
6.27.1概述 1663
6.27.2特性 1663
6.27.3框图 1664
6.27.4基本配置 1664

6.27.5功能描述 1665
6.27.6寄存器映射 1683
6.27.7寄存器描述 1684
6.28 CAN接口 1702
6.28.1概述 1702
6.28.2特性 1702
6.28.3框图 1702
6.28.4基本配置 1703
6.28.5功能描述 1705
6.28.6测试模式 1706
6.28.7CAN通信 1708
6.28.8寄存器映射 1723
6.28.9寄存器描述 1728
6.29 SD卡主机接口 1763
6.29.1概述 1763
6.29.2框图 1764
6.29.3基本配置 1764
6.29.4功能描述 1765
6.29.5寄存器映射 1768
6.29.6寄存器描述 1769
6.30 EBI外部总线接口 1789
6.30.1概述 1789
6.30.2特性 1789
6.30.3框图 1790
6.30.4基本配置 1790
6.30.5功能描述 1792
6.30.6寄存器映射 1802
6.30.7寄存器描述 1803
6.31 USBD USB 1.1设备控制器 1807
6.31.1概述 1807
6.31.2特性 1807
6.31.3框图 1808
6.31.4M48xID/M48xGA基本配置 1808
6.31.5M48xGC/M48xE8基本配置 1808
6.31.6功能描述 1809
6.31.7寄存器映射 1814
6.31.8寄存器描述 1817
6.32 HSUSBD 高速USB2.0设备接口 1844

6.32.1概述 1844
6.32.2特性 1844
6.32.3框图 1844
6.32.4基本配置 1844
6.32.5功能描述 1844
6.32.6寄存器映射 1847
6.32.7寄存器描述 1852
6.33 USB 1.1 主控制器 (USBH) 1901
6.33.1概述 1901
6.33.2特性 1901
6.33.3框图 1902
6.33.4基本配置 1903
6.33.5功能描述 1903
6.33.6寄存器表 1905
6.33.7寄存器描述 1906
6.34 USB 2.0 主控制器(USBH) 1937
6.34.1概述 1937
6.34.2特性 1937
6.34.3框图 1938
6.34.4基本配置 1939
6.34.5功能描述 1939
6.34.6寄存器映射 1941
6.34.7寄存器描述 1943
6.35 USB On-The-Go (OTG) 1994
6.35.1概述 1994
6.35.2特性 1994
6.35.3框图 1994
6.35.4基本配置 1994
6.35.5功能描述 1995
6.35.6寄存器映射 1998
6.35.7寄存器描述 1999
6.36 HSOTG 高速 USB OTG 2008
6.36.1概述 2008
6.36.2特性 2008
6.36.3框图 2008
6.36.4基本配置 2009
6.36.5功能描述 2009
6.36.6寄存器映射 2013
6.36.7寄存器描述 2014

6.37 CRC 控制器 2023
6.37.1概述 2023
6.37.2特性 2023
6.37.3框图 2023
6.37.4基本配置 2024
6.37.5功能描述 2024
6.37.6寄存器映射 2026
6.37.7寄存器描述 2027
6.38 CRYPTO 加解密算法加速器 2032
6.38.1概述 2032
6.38.2特性 2032
6.38.3框图 2033
6.38.4基本配置 2034
6.38.5功能描述 2034
6.38.6寄存器映射 2058
6.38.7寄存器描述 2068
6.39 摄像头接口控制器(CCAP) 2136
6.39.1概述 2136
6.39.2特性 2136
6.39.3框图 2136
6.39.4基本配置 2136
6.39.5功能描述 2138
6.39.6寄存器表 2142
6.39.7寄存器描述 2143
6.40 EADC 增强型 12位模数转换器 2160
6.40.1概述 2160
6.40.2特性 2160
6.40.3框图 2161
6.40.4基本配置 2162
6.40.5功能描述 2163
6.40.6寄存器映射 2178
6.40.7寄存器描述 2182
6.41 DAC 数模转换器 2216
6.41.1概述 2216
6.41.2特性 2216
6.41.3框图 2217
6.41.4基本配置 2217
6.41.5功能描述 2218

6.41.6寄存器映射 2223
6.41.7寄存器描述 2224
6.42 ACMP 模拟比较器控制器 2238
6.42.1概述 2238
6.42.2特性 2238
6.42.3框图 2239
6.42.4基本配置 2239
6.42.5功能描述 2240
6.42.6寄存器映射 2245
6.42.7寄存器描述 2246
6.43 OPA 运算放大器 2253
6.43.1概述 2253
6.43.2特性 2253
6.43.3框图 2254
6.43.4基本配置 2254
6.43.5功能描述 2255
6.43.6寄存器映射 2257
6.43.7寄存器描述 2258
6.44 外设互动 2265
6.44.1概述 2265
6.44.2外设互动矩阵表 2265
6.44.3功能描述 2265
7 应用电路 2268
7.1 电源供电及外部V_{REF} 2268
7.2 电源供电及内部 V_{REF} 2269
7.3 电源和 V_{REF}以及外部RTC使用电池电源 2270
7.4 电源和 V_{REF}以及内部RTC使用电池电源 2271
7.5 外设应用电路 2272
8 电气特性 2273
9 缩写 2274
9.1 缩写 2274
10 封装尺寸 2276
10.1 QFN 33L (5x5x0.8 mm^3 Pitch 0.5 mm) 2276
10.2 LQFP 48L (7x7x1.4 mm^3 Footprint 2.0mm) 2277
10.3 LQFP 64L (7x7x1.4 mm^3 footprint 2.0 mm) 2278

10.4 LQFP 128L (14x14x1.4 mm^3 footprint 2.0 mm) 2279
10.5 LQFP 144L (20x20x1.4 mm^3 footprint 2.0 mm) 2280
11 修订历史..... 2281

图目录

图4.1-1QFN-33引脚图 (0/1 USB FS) 59
图4.1-2LQFP-48引脚图 (0/1 USB FS) 60
图4.1-3LQFP-64引脚图 (0/1 USB FS) 61
图4.1-4LQFP-64引脚图 (0/1 USB FS with V_{BAT}) 62
图4.1-5LQFP-64引脚图 (1 USB HS) 63
图4.1-6LQFP-64引脚图(USB FS + USB HS) 64
图4.1-7LQFP-128引脚图(1 USB FS) 65
图4.1-8LQFP-128引脚图(1 USB FS) 66
图4.1-9LQFP-128引脚图(USB FS + USB HS) 67
图4.1-10LQFP-144引脚图 68
图5.1-1 NuMicro®M480框图 (M48xID/M48xGA) 429
图5.1-2 NuMicro®M480框图 (M48xGC) 430
图 5.1-3 NuMicro® M480框图(M48xE8) 430
图6.1-1Cortex®-M4F框图 431
图6.2-1系统复位源 435
图6.2-2 nRESET 复位时序 437
图6.2-3(POR) 上电复位波形 437
图6.2-4低电压复位 (LVR) 波形 438
图6.2-5欠压侦测 (BOD) 波形 439
图6.2-6 NuMicro®M480电源分布图 440
图6.2-7电源模式状态机 443
图6.2-8 NuMicro®M480电源模式图 446
图6.2-9 SRAM框图 450
图6.2-10SRAM内存结构 451
图6.2-11 NuMicro®M480总线矩阵框图 452
图6.3-1时钟发生器全局示意图(M48xID/M48xGA) 570
图6.3-2时钟发生器全局示意图(M48xGC/M48xE8) 572
图6.3-3时钟发生器框图 572
图6.3-4系统时钟框图 573
图6.3-5HXT停止保护过程 574
图6.3-6时钟控制框图 574
图6.3-7时钟输出框图 575

图6.3-8USB 时钟源 576
图6.4-1真实随机数发生器框图 632
图6.5-1Flash存储控制器框图 642
图6.5-2数据Flash与APROM共享存储空间(128K字节/256K字节/512K字节) 645
图6.5-3SPROM 安全模式 652
图6.5-4新XOM设置流程 654
图6.5-5 OTP 内存映射 667
图6.5-6 KPROM内存映射 668
图6.5-7Flash内存映射 677
图6.5-8支持IAP模式的系统存储映射(仅M48xID/M48xGA) 678
图6.5-9支持IAP模式的系统存储映射(仅M48xGC/M48xE8) 679
图6.5-10支持IAP的LDROM模式 680
图6.5-11支持IAP的APROM模式 680
图6.5-12支持IAP的Boot Loader模式 681
图6.5-13不支持IAP模式的系统存储映射 682
图6.5-14启动源选择 683
图6.5-15ISP 流程示例 686
图6.5-16ISP 32-位编程流程 688
图6.5-17ISP 64-位编程流程 689
图6.5-18多字编程时间 689
图6.5-19固件(firmware)在SRAM中的多字编程 689
图6.5-20固件(firmware)在Boot Loader中的多字编程 690
图6.5-21多字编程流程 691
图6.5-22快速Flash编程校验流程 692
图6.5-23校验流程 692
图6.5-24Flash CRC32 Checksum 计算 693
图6.5-25自动调节Flash访问周期 696
图6.5-26Flash 安全密匙设置流程 698
图6.5-27KEY 比较流程 701
图6.6-1GPIO 控制器框图 740
图6.6-2推挽输出 741
图6.6-3开漏输出 742
图6.6-4准双向 I/O 模式 742

图6.6-5 GPIO上升沿触发中断 743
图6.6-6 GPIO下降沿触发中断 743
图6.7-1 PDMA 控制器框图 766
图6.7-2描述符表入口结构 767
图6.7-3基本模式的有限状态机 768
图6.7-4描述符表格链接单结构 769
图6.7-5Scatter-Gather模式的有限状态机 769
图6.7-6基本模式的单字传输和批量传输举例 771
图6.7-7PDMA 通道 0 超时计数器操作例子 772
图6.7-8跨步功能块传输 773
图6.7-9两种情况的区块传输 774
图6.8-1定时器控制器框图 814
图6.8-2定时器控制器的时钟源 815
图6.8-3 PWM发生器框图 816
图6.8-4PWM系统时钟源控制 816
图6.8-5 PWM计数器时钟源控制 817
图6.8-6 PWM独立模式结构图 817
图6.8-7 PWM互补模式结构图 818
图6.8-8连续计数模式 821
图6.8-9TMx_EXT引脚外部捕获模式 822
图6.8-10外部复位计数器模式 822
图6.8-11内部定时器触发 823
图6.8-12定时器间互触发捕获时序 824
图6.8-13 PWM上数计数模式的预分频波形 825
图6.8-14 PWM上计数模式 825
图6.8-15下计数模式 826
图6.8-16 PWM上下计数模式 826
图6.8-17上下数模式PWM 的比较器事件 827
图6.8-18上数计数模式周期装载模式 828
图6.8-19上计数模式立即装载模式 829
图6.8-20在上下数模式PWM 脉冲的产生 829
图6.8-21在上数模式PWM 脉冲的产生 830
图6.8-22在下数模式PWM 脉冲的产生 830

图6.8-23上数模式和上下数模式PWM占空比周期0%到100%的例子 831
图6.8-24 PWM独立模式输出波形 832
图6.8-25 PWM互补模式输出波形 832
图6.8-26独立模式下PWMx_CH0 输出控制 832
图6.8-27互补模式下PWMx_CH0 和 PWMx_CH1 输出控制 833
图6.8-28死区插入 833
图6.8-29PWM 输出屏蔽控制波形 833
图6.8-30刹车管脚噪音滤波器框图 834
图6.8-31PWMx_CH0 和 PWMx_CH1的刹车事件框图 835
图6.8-32PWMx_CH0 和 PWMx_CH1的边沿侦测器刹车波形 836
图6.8-33PWMx_CH0 和 PWMx_CH1的电平侦测器刹车波形 837
图6.8-34刹车源框图 838
图6.8-35系统失败刹车框图 839
图6.8-36PWMx_CH0 和 PWMx_CH1 带死区插入的极性控制 839
图6.8-37 PWM中断架构图 840
图6.8-38 PWM触发 ADC 框图 840
图6.9-1看门狗定时器框图 890
图6.9-2看门狗定时器时钟控制 891
图6.9-3看门狗定时器定时溢出间隔和复位周期时序图 892
图6.10-1WWDT框图 900
图6.10-2WWDT时钟控制 901
图6.10-3WWDT复位和重载动作 902
图6.10-4当 CNTDAT > CMPDATWWDT 重载计数器 902
图6.10-5当 WWDT_CNT < WINCMPWWDT 重载计数器 903
图6.10-6WWDT中断和复位信号 903
图6.11-1RTC框图 912
图6.11-2动态率定义 918
图6.11-3备份I/O控制框图 920
图6.12-1EPWM发生器简要方块图 959
图6.12-2 EPWM时钟源控制 960
图6.12-3EPWM时钟源控制 961
图6.12-4EPWM独立模式架构图 962
图6.12-5EPWM互补模式架构图 963

图6.12-6EPWM_CH0上计数模式的预分频器波形 965
图6.12-7当设置清计数器 EPWMx计数器的波形 966
图6.12-8EPWM上计数器模式 966
图6.12-9EPWM下计数器模式 967
图6.12-10EPWM上下计数器模式 968
图6.12-11EPWM在上下计数方式中的比较点事件 969
图6.12-12EPWM双缓存说明 969
图6.12-13向上计数方式的周期装载模式 970
图6.12-14向上计数方式的立即装载模式 971
图6.12-15向上计数模式窗口装载 972
图6.12-16上下计数方式的中心装载模式 973
图6.12-17EPWMOne-shot模式输出波形 974
图6.12-18EPWM脉冲产生 975
图6.12-19EPWM0%到100%占空比脉冲产生 975
图6.12-20EPWM独立模式波形 976
图6.12-21EPWM互补模式波形 977
图6.12-22EPWM组功能波形 978
图6.12-23EPWMSYNC_IN噪音滤波框图 978
图6.12-24 EPWM计数器同步功能框图 979
图6.12-25EPWM同步源来自SYNC_IN 信号的同步功能 980
图6.12-26EPWMx_CH0独立模式下的输出控制 980
图6.12-27EPWMx_CH0 和 EPWMx_CH1 在互补模式下的输出控制 981
图6.12-28死区插入 982
图6.12-29屏蔽控制波形图解 982
图6.12-30刹车噪音滤波器方块图 983
图6.12-31EPWMx_CH0 和 EPWMx_CH1 通道组刹车方块图 984
图6.12-32EPWMx_CH0 和 EPWMx_CH1通道组的边沿检测波形 985
图6.12-33EPWMx_CH0 和 EPWMx_CH1通道组的电平检测波形 985
图6.12-34刹车源框图 986
图6.12-35系统故障刹车框图 987
图6.12-36EPWM LEB功能波形 987
图6.12-37带上升沿死区插入初始状态和极性控制 988
图6.12-38EPWMx_CH0 累加中断波形 989

图6.12-39EPWMx_CH0 和 EPWMx_CH1组中断架构图 990
图6.12-40故障检测功能中断框图 991
图6.12-41EPWMx_CH0和EPWMx_CH1对触发EADC事件。 992
图6.12-42EPWMx_CH0触发EADC框图 992
图6.12-43上下计数方式EPWM触发EADC的时序波形 993
图6.12-44EPWM_CH0 和 EPWM_CH1组触发DAC框图 993
图6.12-45EPWM_CH0捕获框图 994
图6.12-46捕获操作波形图 995
图6.12-47通道0捕获PDMA 操作波形 996
图6.12-48累加器 PDMA 功能架构 997
图6.12-49EPWM累加器停止模式波 998
图6.12-50故障检测功能框图 999
图6.12-51 EPWM屏蔽功能波形 1000
图6.12-52 EPWM抗尖峰脉冲功能波形 1000
图6.13-1BPWM发生器框图 1088
图6.13-2BPWM时钟源控制 1089
图6.13-3BPWM时钟源控制 1089
图6.13-4BPWM独立模式框架图 1090
图6.13-5BPWM_CH0 CLKPSC波形 1092
图6.13-6 BPWM计数器清空波形 1093
图6.13-7BPWM向上计数模式 1093
图6.13-8BPWM向下计数方式 1094
图6.13-9BPWM上下计数方式 1094
图6.13-10上下计数方式时 BPWM CMPDAT 事件 1095
图6.13-11向上计数方式下的周期载入模式 1096
图6.13-12向上计数方式下的立即载入模式 1097
图6.13-13上下计数方式下的中间载入模式 1098
图6.13-14 BPWM脉冲发生 (左：不对称脉冲，右：多样脉冲) 1099
图6.13-15 BPWM0% 到 100% 脉冲产生 (左:上计数器模式, 右：上下计数器模式) 1099
图6.13-16BPWM_CH0输出控制的 3 步 1100
图6.13-17屏蔽功能波形说明 1101
图6.13-18初始状态和极性控制 1102
图6.13-19BPWM_CH0 和 BPWM_CH1对中断架构图 1102

图6.13-20BPWM_CH0 和 BPWM_CH1 触发 EADC0/1源的框架图 1103
图6.13-21BPWM CH0~ CH5 触发 EADC0/1的框架图 1103
图6.13-22上下计数方式下 BPWM 触发 EADC0/1的时序波形图 1104
图6.13-23BPWM_CH0捕捉框图 1105
图6.13-24捕捉操作波形图 1106
图6.14-1 QEI框图 1143
图6.14-2 QEI时钟控制 1144
图6.14-3噪声滤波器 1145
图6.14-4噪声滤波器采样时钟源选择 1146
图6.14-5QEA/QEB/IDX 通过噪声滤波器所需时序 1146
图6.14-6 X4计数模式 1147
图6.14-7 X2计数模式 1148
图6.14-8比较操作 1149
图6.14-9 QEI_CNT 重载/复位控制 1150
图6.14-10捕捉 QEI 计数器的触发控制 1150
图6.14-11捕捉和锁存 QEI 计数器 1151
图6.14-12QEI 中断架构图 1152
图6.15-1输入捕捉定时器/计数器架构 1164
图6.15-2输入捕捉定时器/计数器时钟源控制 1166
图6.15-3噪声滤波器采样时钟选择 1167
图6.15-4输入捕捉定时器/计数器功能框图 1168
图6.15-5输入捕捉定时器/计数器中断架构图 1170
图6.16-1 UART时钟控制框图 1186
图6.16-2 UART结构图 1186
图6.16-3自动波特率检测 1196
图6.16-4发送延时操作 1197
图6.16-5 UART nCTS唤醒示例1 1198
图6.16-6 UART nCTS 唤醒示例2 1198
图6.16-7 UART数据唤醒 1199
图6.16-8UART 接收数据 FIFO 达到阈值唤醒 1199
图6.16-9 UART RS-485 AAD模式地址匹配唤醒 1199
图6.16-10 UART数据接收 FIFO 超时唤醒 1200
图6.16-11自动流控框图 1203

图6.16-12 UART nCTS自动流控使能 1204
图6.16-13 UART nRTS自动流控功能使能 1204
图6.16-14 UART nRTS软件控制的自动流控 1205
图6.16-15 IrDA控制框图 1205
图6.16-16 IrDA TX/RX时序图 1206
图6.16-17LIN 帧结构 1207
图6.16-18LIN 字节结构 1207
图6.16-19LIN模式暂停检测 1209
图6.16-20 LIN帧 ID 和校验格式 1209
图6.16-21 LIN同步域测量 1211
图6.16-22AR模式下，SLVEUEN=1 时，UART_BAUD 更新顺序 1212
图6.16-23AR 模式下，SLVEUEN=0 时，UART_BAUD 更新顺序0 1212
图6.16-24 RS-485 nRTS自动方向控制模式下 nRTS 管脚驱动电平 1214
图6.16-25 RS-485软件控制时的驱动电平 1215
图6.16-26RS-485帧结构 1216
图6.17-1EMAC框图 1257
图6.17-2以太网帧格式 1259
图6.17-3 RXDMA描述符数据结构 1261
图6.17-4 TXDMA描述符数据结构 1267
图6.18-1SC时钟控制框图(时钟控制器含7位的预分频计数器) 1330
图6.18-2SC控制器框图 1330
图6.18-3SC数据格式 1334
图6.18-4SC激活序列 1336
图6.18-5SC热复位序列 1337
图6.18-6SC释放序列 1338
图6.18-7基本操作流程 1339
图6.18-8初始化字符TS 1340
图6.18-9SC错误信号 1341
图6.18-10发送时块保护时间操作 1344
图6.18-11接收时的块保护时间操作 1344
图6.18-12扩展保护时间操作 1344
图6.19-1I^2S控制器框架 1373
图6.19-2I^2S时钟控制框图 1375

图6.19-3主机模式接口框图 1375
图6.19-4从机模式接口框图 1375
图6.19-5I²S通道宽度和数据宽度(CHWIDTH≦DATWIDTH) 1376
图6.19-6 I²S通道宽度和数据宽度(CHWIDTH>DATWIDTH) 1376
图6.19-7 I²S数据格式时序图(FORMAT=0x0;CHWIDTH≦DATWIDTH) 1377
图6.19-8MSB对齐数据格式(FORMAT=0x1;CHWIDTH>DATWIDTH) 1377
图6.19-9LSB对齐数据格式(FORMAT=0x2;CHWIDTH>DATWIDTH) 1377
图6.19-10标准PCM音频时序图(FORMAT=0x4;CHWIDTH≦DATWIDTH) 1378
图6.19-11MSB对齐的PCM数据格式(FORMAT=0x5;CHWIDTH>DATWIDTH) 1378
图6.19-12LSB对齐的PCM数据格式(FORMAT=0x6;CHWIDTH>DATWIDTH) 1378
图6.19-13用32位通道块传输6通道24位音频数据的TDM传输(PCM标准模式;FORMAT=0x4) ... 1379
图6.19-14用32位通道块传输6通道24位音频数据的TDM传输(PCMMSB对齐;FORMAT=0x5) .. 1379
图6.19-15用32位通道块传输6通道24位音频数据的TDM传输(PCMLSB对齐;FORMAT=0x6) ... 1379
图6.19-16 I²S中断 1380
图6.19-17不同设置下2通道音频数据的FIFO示意 1381
图6.19-18不同设置下4通道音频数据的FIFO示意 1382
图6.19-19不同设置下6通道音频数据的FIFO示意(1) 1383
图6.19-20不同设置下6通道音频数据的FIFO示意(2) 1384
图6.20-1框架图 1405
图6.20-2 SPI 外设时钟 1408
图6.20-3 SPI 全双工主机模式 1409
图6.20-4 SPI 全双工从机模式 1409
图6.20-532位的收发 1410
图6.20-6自动从机选择(SSACTPOL = 0, SUSPITV > 0x2) 1411
图6.20-7自动从机选择(SSACTPOL = 0, SUSPITV < 0x3) 1411
图6.20-8字节重排功能 1412
图6.20-9字节挂起波形图 1412
图6.20-10 SPI半双工主机模式 1413
图6.20-11 SPI 半双工从机模式 1413
图6.20-12 FIFO 阈值比较器 1414
图6.20-13 8-16位宽发送 FIFO 缓存示例 1415
图6.20-1417-32位宽发送 FIFO 缓存示例 1416

图6.20-15接收16位宽FIFO 缓存示例 1417
图6.20-16接收32位宽FIFO 缓存示例 1417
图6.20-17 TX下溢事件和从机溢出事件 1418
图6.20-18从机模式位计数错误 1418
图6.20-19I²S数据格式时序 1420
图6.20-20对齐数据格式时序图 1420
图6.20-21PCM 模式 A 时序图 1421
图6.20-22PCM 模式 B 时序图 1421
图6.20-23不同 I2S 模式下的FIFO 内容 1422
图6.20-24 SPI主机模式时序 1425
图6.20-25SPI主机模式时序 (SPIx_CLK交替相位) 1425
图6.20-26 SPI从机模式时序 1426
图6.20-27 SPI从机模式时序 (SPIx_CLK交替相位) 1426
图6.21-1QSPI框架图 1454
图6.21-2QSPI外设时钟 1456
图6.21-3QSPI全双工主机模式 1457
图6.21-4QSPI全双工从机模式 1457
图6.21-532位的收发 1458
图6.21-6自动从机选择(SSACTPOL=0,SUSPITV>0x2) 1459
图6.21-7自动从机选择(SSACTPOL=0,SUSPITV<0x3) 1459
图6.21-8字节重排功能 1460
图6.21-9字节挂起波形图 1460
图6.21-10QSPI半双工主机模式 1461
图6.21-11QSPI半双工从机模式 1461
图6.21-122位传输模式 1462
图6.21-132位传输模式的时序图(主机模式) 1463
图6.21-14双输出模式的位时序 1463
图6.21-15双输入模式的位时序 1464
图6.21-16四I/O输出模式位顺序 1465
图6.21-17四I/O输入模式位顺序 1465
图6.21-18TX DTR模式下QSPI输出数据时序图 1466
图6.21-19 FIFO阈值比较器 1467

图6.21-20发送FIFO缓存示例 1468
图6.21-21接收FIFO缓存示例 1469
图6.21-22 TX下溢事件和从机溢出事件 1469
图6.21-232位传输模式FIFO缓存示例 1470
图6.21-24 TX下溢事件(QSPI0从机三线模式使能) 1470
图6.21-25从机模式位计数错误 1471
图6.21-26从机超时事件 1471
图6.21-27QSPI主机模式时序 1473
图6.21-28QSPI主机模式时序(QSPIx_CLK交替相位) 1474
图6.21-29QSPI从机模式时序 1474
图6.21-30从机模式时序(QSPIx_CLK交替相位) 1475
图6.22-1SPIM方框图 1493
图6.22-2SPIM时序图 1494
图6.22-3类型_1 四线模式写的编程命令模式 1497
图6.22-4类型_2四线模式写的编程命令模式 1497
图6.22-5类型_3四线模式写的编程命令模式 1498
图6.22-6连续读模式被禁用下命令为0xEB的快速读四线I/O 1499
图6.22-7连续读模式使能下命令为0xEB的快速读四线 1499
图6.22-8 DTR/DDR快速读命令 0x0D 1502
图6.22-9 DTR/DDR快速读双线I/O命令0xBD 1502
图6.22-10 DTR/DDR快速读四线I/O命令0xED 1503
图6.22-11SPIM_SS 激活建立时间与SPIM_CLK 的关联和 SPIM_SS 取消选择时间 1523
图6.23-1 I^2C 控制器框图 1527
图6.23-2 I^2C 总线时序 1529
图6.23-3 I^2C 协议 1530
图6.23-4起始和停止状况 1530
图6.23-5 I^2C 总线上的位传输 1531
图6.23-6 I^2C 总线上的应答信号 1531
图6.23-77位地址情况下主机向从机传输数据 1532
图6.23-87位地址情况下主机向从机读取数据 1532
图6.23-910位地址情况下主机向从机传输数据 1532
图6.23-1010位地址情况下主机向从机读取数据 1533
图6.23-11通过当前状态控制I^2C总线 1533

图6.23-12主机发送模式控制流程 1534
图6.23-13主机接收模式控制流程 1535
图6.23-14从机模式控制流程 1536
图6.23-15 GC模式 1537
图6.23-16仲裁丢失 1538
图6.23-17总线管理主要元素包协议图 1540
图6.23-18 7位可寻址的设备与主机通信 1541
图6.23-19 7位可寻址的设备对警报响应地址的响应 1541
图6.23-20总线管理警报功能 1542
图6.23-21系统管理总线超时时序 1543
图6.23-22总线时钟低电平超时时序 1543
图6.23-23建立时间错误调整 1545
图6.23-24保持时间错误调整 1545
图6.23-25 I^2C 数据以为方向 1546
图6.23-26 I^2C 超时计数模块框图 1548
图6.23-27 I^2C 唤醒的相关信号波形 1548
图6.23-28 EEPROM随机读取 1550
图6.23-29 EEPROM随机读取协议 1551
图6.24-1USCI框图 1577
图6.24-2输入调节USCIx_DAT[1:0]和USCIx_CTL[1:0] 1578
图6.24-3输入调节USCIx_CLK 1579
图6.24-4数据缓冲框图 1580
图6.24-5数据访问结构 1581
图6.24-6传送数据路径 1581
图6.24-7接收数据路径 1582
图6.24-8相关协议时钟发生器 1583
图6.24-9基本时钟分频计数器 1584
图6.24-10时序测量计数器框图 1584
图6.24-11采样时间计数器 1585
图6.24-12事件和中断结构 1586
图6.25-1USCI-UART模式框图 1588
图6.25-2全双工通信的UART信号连接 1590
图6.25-3 UART标准帧格式 1591

图6.25-4 UART位时序(数据采样时间) 1593
图6.25-5 UART 自动波特率控制 1595
图6.25-6输入数据唤醒 1596
图6.25-7nCTS唤醒状态1 1596
图6.25-8nCTS唤醒状态2 1596
图6.26-1 SPI主机模式应用框图 1623
图6.26-2 SPI从机模式应用框图 1623
图6.26-3USCI SPI模式框图 1624
图6.26-44-写全双工SPI通信信号（主机模式） 1626
图6.26-54-写全双工SPI通信信号（从机模式） 1627
图6.26-6不同时钟模式下的SPI通信(SCLKMODE=0x0) 1628
图6.26-7不同时钟模式下的SPI通信(SCLKMODE=0x1) 1628
图6.26-8不同时钟模式下的SPI通信(SCLKMODE=0x2) 1629
图6.26-9不同时钟模式下的SPI通信(SCLKMODE=0x3) 1629
图6.26-10 MSB优先16位数据长度传输一个字的格式 1630
图6.26-11两个字传输间的字暂停 1630
图6.26-12自动从机选择(SUSPITV ≧ 0x3) 1631
图6.26-13自动从机选择(SUSPITV < 0x3) 1632
图6.26-14一个输出数据通道半双工通信(SPI主机模式) 1633
图6.26-15一个输入数据通道半双工通信(SPI主机模式) 1633
图6.26-16主机模式下SPI时序 1635
图6.26-17主机模式下SPI时序 (串行总线时钟的反相位) 1635
图6.26-18从机模式下SPI时序 1636
图6.26-19从机模式下SPI时序(串行总线时钟的反相位) 1636
图6.27-1 I^2C总线时序 1663
图6.27-2USCI I^2C模式框图 1664
图6.27-3I^2C协议 1665
图6.27-4START和STOP状况 1666
图6.27-5 I^2C总线上的位传输 1667
图6.27-6 I^2C总线上的应答信号 1667
图6.27-7仲裁丢失 1668
图6.27-8根据当前I^2C状态控制I^2C总线 1670
图6.27-97位地址格式主机向从机传输数据 1671

图6.27-107位地址情况下主机向从机读取数据 1671
图6.27-1110位地址格式主机向从机传输数据 1671
图6.27-12 10位地址格式主机向从机读取数据 1671
图6.27-137位地址主机发送模式控制流程 1672
图6.27-14 7位地址主机接收模式控制流程 1673
图6.27-15 10位地址主机发送模式控制流程 1674
图6.27-16 10位地址接收模式控制流程 1675
图6.27-177位地址从机模式控制流程 1676
图6.27-18 10位地址从机控制流程 1677
图6.27-197位地址的GC模式 1678
图6.27-20建立时间错误调整 1680
图6.27-21保持时间错误调整 1680
图6.27-22 I^2C超时计数模式框图 1681
图6.27-23 EEPROM随机读取 1682
图6.27-24 EEPROM随机读取协议 1682
图6.28-1 CAN外设框图 1703
图6.28-2 CAN内核静默模式 1706
图6.28-3 CAN内核回环模式 1707
图6.28-4 CAN内核回环模式与静默模式组合 1707
图6.28-5 Ifn寄存器和报文RAM间的数据传输 1709
图6.28-6应用软件处理FIFO缓存 1714
图6.28-7位时间 1716
图6.28-8传播时间段 1716
图6.28-9同步在“Late”及“Early”边沿 1718
图6.28-10过滤显性干扰短脉冲 1719
图6.28-11 CAN内核CAN协议控制器架构 1720
图6.29-1 SD主机控制器框图 1764
图6.29-2 PAD (物理地址描述符) 表格式 1767
图6.30-1 EBI框图 1790
图6.30-2 16位EBI数据宽度与16位设备的连接 1793
图6.30-3 8位EBI数据宽度与8位设备的连接 1793
图6.30-4分离模式下16位EBI数据宽度与16位设备的连接 1794
图6.30-5分离模式下8位EBI数据宽度与8位设备的连接 1794

图6.30-6 16位数据宽度的时序控制波形 1796
图6.30-7 8位数据宽度的时序控制波形 1797
图6.30-816位数据宽度字节写的时序控制波形 1798
图6.30-9插入空闲周期的时序控制 1799
图6.30-10位数据宽度在分离模式下的时序控制波形 1800
图6.30-11连续数据访问模式的时序控制波形 1801
图6.31-1 USB框图 1808
图6.31-2唤醒中断操作流程 1810
图6.31-3端点SRAM结构 1811
图6.31-4 Setup事务后接着是Data In事务 1812
图6.31-5数据输出 1812
图6.31-6LPM状态转换图 1813
图 6.32-1 USB设备控制器框图 1844
图6.33-1 USB 1.1 主控制器框图 1902
图6.34-1 USB 2.0 FS主机控制器框图 1938
图6.35-1 USB OTG 框图 1994
图6.35-2 USB 设备模式 1995
图6.35-3 USB 主机模式 1996
图6.36-1 USB OTG 框图 2009
图6.36-2 USB设备模式 2010
图6.36-3 USB主机模式 2011
图6.37-1CRC 发生器框图 2024
图6.37-2校验和位顺序反转功能框图 2025
图6.37-3写数据位顺序反转功能框图 2025
图6.38-1加解算法加速器框图 2034
图6.38-2 PRNG 功能框图 2037
图6.38-3电子密码本模式 2038
图6.38-4密码块链模式 2039
图6.38-5密码反馈模式 2040
图6.38-6输出反馈模式 2041
图6.38-7计数器模式 2042
图6.38-8 CBC-CS1 加密 2042
图6.38-9列出C_{n-1}^{*}是局部块的情况下CBC-CS1-Decrypt 算法 2042

图6.38-10 CBC-CS1 解密 2043
图6.38-11ECC的主要层次结构图 2047
图6.39-1摄像头接口功能框图 2136
图6.40-1 EADC 转换框图 2161
图6.40-2模块0~3框图 2164
图6.40-3模块 4~15 框图 2165
图6.40-4模块 16~18 框图 2165
图6.40-5 EADC0/1时钟控制 2166
图6.40-6 EADC0/1 转换时序图, n=0~18 2167
图6.40-7模块转换优先级仲裁框图 2168
图6.40-8选定模块EADC0/1 EOC 信号 ADINT0~3中断 2170
图6.40-9 EPWM触发 EADC0/1 开始转换 2170
图6.40-10外部触发 EADC0/1 开始转换 2171
图6.40-11转换开始延时时间 2172
图6.40-12 EADC0/1_ST防抖时序 2172
图6.40-13 ADC延长采样时间时序 2173
图6.40-14 ADC转换结果监控逻辑框图 2174
图6.40-15 EADC0/1 同步触发 2175
图6.40-16 ADC控制器中断 2176
图6.40-17 ADC带校准的启动序列 2177
图6.41-1数模转换器框图 2217
图6.41-2数据保持寄存器格式 2219
图6.41-3DAC 转换由软件写触发 2219
图6.41-4 DAC转换由硬件触发事件开始 2220
图6.41-5DAC0 和 DAC1 组模式和非组模式更新示例 2220
图6.41-6DAC PDMA 下溢条件示例 2221
图6.41-7软件PDMA模式 DAC 连续转换 2221
图6.41-8 DAC中断源 2222
图6.42-1模拟比较器框图 2239
图6.42-2 ACMP0比较器迟滞回差功能 2240
图6.42-3窗口锁存模式 2241
图6.42-4滤波功能示例 2241
图6.42-5比较器控制器中断 2242

图6.42-6比较器参考电压框图 2242
图6.42-7窗口比较模式示例 2243
图6.42-8窗口比较模式示例 2243
图6.43-1运算放大器框图 2254

表目录

表6.2-1寄存器复位值 437
表6.2-2电源模式表 442
表6.2-3电源模式区别表 442
表6.2-4电源模式定义表 443
表6.2-5不同电源模式的时钟 444
表6.2-6重新进入掉电状态的条件 446
表6.2-7片上地址空间分布 449
表6.2-8SRAM结构 450
表6.2-9异常类型 525
表6.2-10中断向量表 528
表6.2-11优先级分组 563
表6.3-1输出频率公式的符号定义 607
表6.5-1不同芯片型号FMC特征比较表 640
表6.5-2区块地址范围 644
表6.5-3向量映射支持情况 683
表6.5-4 ISP命令表 685
表6.5-5 FMC用于Flash编程的控制寄存器 687
表6.5-6自动调节功能下最优化的Flash访问周期 695
表6.5-7对CPU的四种保护的锁定效果表 701
表6.5-8对ICE/ ICP/Writer的四种保护的锁定效果表 702
表6.7-1通道优先级表 768
表6.8-1TIMER01管脚配置 819
表6.8-2TIMER23管脚配置 819
表6.8-3上数计数模式中PWM 脉冲产生事件优先级 830
表6.8-4下数计数模式中PWM 脉冲产生事件优先级 831
表6.8-5上下数计数模式中PWM 脉冲产生事件优先级 831
表6.9-1看门狗定时器定时溢出间隔周期选择 892
表6.10-1 WWDT预分频值选择 901
表6.10-2 CMPDAT设置限制 904
表6.11-1RTC 读/写使能 914
表6.11-212/24小时制选择 915
表6.11-3上电后寄存器值 917

表6.11-4寄存器电源域 917
表6.11-5动态样本源选择 919
表6.11-6Tamper控制位对0对的影响 919
表6.11-7控制位对1和2对的影响 919
表6.12-1EPWM时钟源控制寄存器设置表 960
表6.12-2向上计数器EPWM脉冲发生事件优先级 976
表6.12-3向下计数器EPWM脉冲发生优先级 976
表6.12-4上下计数器EPWM脉冲产生事件优先级 976
表6.12-5EPWM中断向量列表 991
表6.13-1BPWM特性比较表 1087
表6.13-2BPWM时钟源控制寄存器设置表 1089
表6.13-3向上计数方式下的 BPWM 脉冲发生事件优先 1099
表6.13-4向下计数方式的 BPWM 脉冲发生事件优先级 1100
表6.13-5上下计数方式的 BPWM 脉冲发生事件优先级 1100
表6.14-1计数方向 1149
表6.15-1一个典型的滤波设置 1166
表6.16-1 M480系列 UART 特性 1184
表6.16-2 UART中断 1187
表6.16-3 UART接口控制器引脚 1193
表6.16-4 UART 控制器波特率公式表 1194
表6.16-5 UART 控制器波特率参数设置示例 1194
表6.16-6 UART 控制器波特率寄存器设置示例 1194
表6.16-7波特率补偿示例1 1195
表6.16-8波特率补偿示例2 1196
表6.16-9 UART控制器中断源和标志位列表 1202
表6.16-10 UART线控的数据位和停止位长度设置 1202
表6.16-11 UART线控校验位设置 1203
表6.16-12 LIN主机模式报头选择 1207
表6.17-1仲裁器仲裁结果 1258
表6.17-264-位参考时间计数器 1260
表6.17-3RXDMA描述符字1(TSEN(EMAC_TSCTL[0])为0) 1264
表6.17-4RXDMA描述符字1(TSEN(EMAC_TSCTL[0])为1) 1264
表6.17-5RXDMA描述符字2 1265

表6.17-6RXDMA描述符字3(TSEN(EMAC_TSCTL[0])为0) 1266
表6.17-7RXDMA描述符字3(TSEN(EMAC_TSCTL[0])为1) 1266
表6.17-8 TXDMA描述符字0 1269
表6.17-9 TXDMA描述符字1(TSEN(EMAC_TSCTL[0])是0) 1270
表6.17-10 TXDMA描述符字1(TSEN(EMAC_TSCTL[0])是1) 1270
表6.17-11TXDMA描述符字2 1272
表6.17-12TXDMA描述符字3(TSEN(EMAC_TSCTL[0])为0) 1273
表6.17-13 TXDMA描述符字3(TSEN(EMAC_TSCTL[0])为1) 1273
表6.17-14不同CAMCMR设置及可接收的包类型 1279
表6.17-15MII管理帧格式 1294
表6.17-16MII管理功能配置时序 1294
表6.18-1SC主控制器管脚描述 1331
表6.18-2UART管脚描述 1331
表6.18-3 Timer0/Timer1/Timer2操作模式 1344
表6.20-1 SPI/I^{2}S 接口管脚描述(SPI0~SPI2) 1408
表6.20-2I^{2}S/PCM主机模式和单声道模式下Dummy数据个数 1423
表6.20-3I^{2}S/PCM主机模式和立体声模式下Dummy数据个数 1423
表6.20-4I^{2}S/PCM从机模式和单声道模式下Dummy数据个数 1423
表6.20-5I^{2}S/PCM从机模式和立体声模式下Dummy数据个数 1424
表6.20-6I^{2}S从机模式和立体声模式下Dummy数据个数 1424
表6.20-7PCM I^{2}S从机模式和立体声模式下Dummy数据个数 1424
表6.22-1SPI flash控制器在AMBA 系统中映射地址 1504
表6.22-2从机片选电平和激活使能的功能描述 1511
表6.23-1 SMBus保留地址 1539
表6.23-2I^{2}C传输速率与PCLK之间的关系 1544
表6.23-3 I^{2}C 状态码描述 1547
表6.24-1不同协议的输入信号 1578
表6.24-2不同协议输出信号 1579
表6.24-3 Data Transfer Events and Interrupt Handling 1586
表6.24-4具体协议事件和中断处理 1586
表6.25-1 UART协议输入信号 1591
表6.25-2UART协议的输出信号 1591
表6.26-1串行总线时钟配置 1627

表6.27-1 I²C传输速率与PCLK之间的关系 1679
表6.28-1初始化传送对象 1711
表6.28-2初始化接收对象 1712
表6.28-3 CAN位时间参数 1716
表6.28-4 CAN寄存器中每个位的功能 1727
表6.28-5错误码 1732
表6.28-6中断源 1735
表6.28-7 IF1和IF2报文接口寄存器 1738
表6.28-8报文对象在内存中的结构 1752
表6.29-1SDH特性比较表 1763
表6.29-2 SD0 引脚配置 1765
表6.29-3 SD1引脚配置 1765
表6.30-1 EBI地址映像 1792
表6.30-2时序控制参数 1795
表6.31-1 USB链路电源管理(Lx)状态 1813
表6.32-1分散收集描述符格式 1846
表6.33-1 USB 1.1 主控制器管脚配置 1903
表6.38-1不同芯片加密特性比较表 2033
表6.38-2每个引擎错误条件和错误标识 2035
表6.38-3 DMA使能位列表 2036
表6.38-4 DMA 串流位表 2036
表6.38-5通道选择位表 2036
表6.38-6 ECC参数和相应寄存器列表 2047
表6.38-7各种操作所需的输入数据 2048
表6.38-8轻量级二进制不可约多项式 2052
表6.40-1分辨率与转换周期的关系 2168
表6.40-2 EADC0/1 差分模式信道选择 2175
表6.40-3 EADC0/1 省电模式 2176
表6.40-4 EADC0/1 校准启动 2177
表6.42-1窗口比较逻辑真值表 2243
表6.44-1外设互动矩阵表 2265
表9.1-1缩写表 2275

1 概述

NuMicro®M480 系列单片机内置 Arm®Cortex®-M4F 内核，安全引导，硬件加密，支持 DSP 指令，集成浮点单元。M480 系列根据其特点和应用分为六个子系列。M480 系列支持最大 512KB 的 Flash 大小和最大 160KB 的 SRAM 大小。操作频率 192 MHz 时的动态功耗 175/130 μA/MHz 和待机电流可以降低至 1μA。M480 系列支持安全启动功能，它提供了一个恒定的数字签名系统软件识别在启动期间，以保护 Flash 内容的完整性免受攻击。嵌入式硬件加密引擎提供了快速、简单的加密、解密、ID 认证、私钥和公钥特性。此外，M480 系列支持 10/100Mbps 以太网 RMII、高速 USB 2.0 OTG、双 12 位 5 MSPS SAR ADC、摄像头接口和多功能外围设备，适用于物联网、工业自动化、传感器网络、汽车设备、RC 飞机、智能家居、网络网关和消费电子产品。

NuMicro® M480全系列包含

- NuMicro® M481 Base系列：高性能，低动态功耗，高速UART/SPI/I2C/PWM外设
- NuMicro® M482 USB FS OTG系列：支持USB 2.0全速OTG PHY，符合游戏或个人电脑配件
- NuMicro® M483 CAN系列：提供2组CAN 2.0B总线及2组USB2.0接口，双ADC和9组UART接口
- NuMicro® M484USB HS OTG系列：同时USB 2.0全速OTG PHY，并提供装置，主机，OTG等传输模式
- NuMicro® M485 加密系列：内建硬体加密引擎及乱数产生器（RNG）
- NuMicro® M487 以太网系列：内建10 / 100M乙太网控制器（MAC），支持RMII，MDC，MDIO，可快速实现网路连接

Series	USB Full Speed	USB High Speed	CAN 2.0B	Cryptography	Ethernet
M481					
M482	√				
M483	√	√	√		
M484	√	√			
M485	√	√		√	
M487	√	√	√	√	√

2 特性

核心和系统	
ARM® Cortex®-M4	• ARM® Cortex®-M4F，主频192MHz • 内存保护单元 (MPU) • 内置嵌套矢量中断控制器(NVIC) • IEEE 754 兼容浮点运算单元(FPU) • DSP 指令扩展，带硬件除法器及单周期 32硬件乘法器 • 24位系统定时器 • 支持可屏蔽中断 • 通过WFI/WFE指令指令，支持低功耗休眠功能
BOD 欠压检测	• 8个电压选择支持中断或复位功能: 3.0V/2.8V/2.6V/2.4V/2.2V/2.0V/1.8V/1.6V
低压复位	• 电压门限: 1.5 V
安全	• 96位唯一序列号 (UID) • 128位用户代码 (UCID) • 内建温度传感器，精度 1℃
记忆装置	
Boot loader **(M48xID/M48xGA)**	• 32 KB保护启动程序 • 支持安全启动功能 SHA-256 和 AES-256 从 APROM, LDROM 和外部 SPI flash • ISP功能支持接口： UART和 USB • 支持 ISP/IAP 库
Boot Loader **(M48xGC/M48xE8)**	• 8KB的mask ROM用于安全启动 • 使用 ECC 验证数据 APROM, LDROM 和外部 SPI Flash
内嵌闪存	• 双 bank512 KB/256 KB应用代码空间(APROM) 用于代码安全升级 (M48xID/M48xGA) • 单bank256KB的APROM。(M48xGC/M48xE8) • 四个eXecute-Only-Memory区域用于数据保护(M48xGC/M48xE8) • 在连续地址读访问中最大可运行到零等待的192 MHz • 4 KB 加载代码空间 (LDROM) • 8 KB密钥保护存储空間 (KPROM) 保护固件编程.

	• 4 K字节安全保护内存 (SPROM)保护知识产权 • 3 K字节单次编程存储器 (OTP)保护数据安全.(M48xID/M48xGA) • 2 K字节单次编程存储器 (OTP)保护数据安全.(M48xGC/M48xE8) • 擦除页大小： 4 KB • 支持快速编程CRC校验. • 支持在系统编程 (ISP), 和在应用编程(IAP) ， 在电路编程（ICP） • 可配置启动方式,包括Boot Loader,加载代码空间(LDROM) 或应用代码空间(APROM) • 数据空间大小可配置 • 2线 SWD 编程接口 • 支持32-bit、64-bit和多字编程功能.
SRAM	• 160 KB的片上存储器包括:(M48xID / M48xGA) – 32KB SRAM位于bank0，支持硬件奇偶校验和保持模式;异常(NMI)在奇偶校验错误时生成 – 96/ 32KB SRAM，位于bank1 – 32KB SRAM位于bank2，可作为外部SPI闪存缓存 • 128 KB的片上存储器包括:(M48xGC / M48xE8) – 32KB SRAM位于bank0，支持硬件奇偶校验和保持模式;异常(NMI)在奇偶校验错误时生成 – 96 KBSRAM位于bank1 • 字节、半字和字访问 • PDMA 操作
CRC 计算单元	• 支持四个常用多项式 CRC-CCITT, CRC-8, CRC-16, 和 CRC-32 • 初值及种子值可配 • 支持反向运算 • 支持补码运算 • 支持 8/16/32位运算 • 支持 8位写模式: 1-AHB 时钟操作 • 支持 16位写模式: 2-AHB 时钟操作 • 支持 32位写模式: 4-AHB 时钟操作 • 支持 DMA 完成运算
PDMA (外设DMA)	• 16 个传输通道 • 支持基本和分散聚集(catter-Gather)传输模式 • 分散聚集(catter-Gather)传输模式支持环形缓冲管理 • 支持固定模式和循环模式优先级 • 支持一次或突发传输 • 支持字节、半字，字传输 • 源地址和目标地址都支持递增或固定.

• 时钟	
外部时钟控制	• 可外接 4~24 MHz 晶体 (HXT) • 可外接 32.768 kHz 晶体 (LXT) 用于 RTC 和低功耗系统时钟 • 有晶体失效检测功能 • 时钟失效可配置为非屏蔽中断
内置时钟控制	• 48MHz内部高速振荡器（HIRC48）无需USB晶振(M48xGC/M48xE8) • 内建 12 MHz RC 时钟 (HIRC) 精度±2% • 内建 10 kHz RC时钟 (LIRC) 用于WDT和唤醒功能 • 内建 PLL 可把HIRC或HXT倍频到 480 MHz
RTC 实时时钟	• 实时时钟是独立的电源并拥有独立的电池引脚(M48xGC/M48xE8) • RTC时钟源包括低速外部晶体振荡器 • 电池电源可保存80字节数据，可以被开路检测脚清除（M48xID/M48xGA) • 电池电源可保存20字节数据，可以被开路检测脚清除（M48xGC/M48xE8) • 支持6个可选的静态和动态数据清除引脚 • 可将MCU从任何省电模式唤醒 • 时钟补偿功能，可在5秒内达到±5ppm 精度 • 有报警功能 (秒分时，日月年) • RTC定时和报警支持中断 • 闰年计算 • 有 1 Hz时钟输出
计时器	
TMR 定时器控制器	**定时器控制器** • 4 组 32-位定时器，带24位向上计数器和一个8位的预分频计数器 • 提供单次, 周期, 触发输出和连续计数四种操作模式 • 支持外部管脚捕获事件计数功能 • 支持外部引脚捕获，可用于复位24位向上定时器 • 如果定时器中断信号产生，支持芯片从空闲/掉电模式唤醒 **PWM** • 8个独立PWM 输出，16位计数器，12位预分频，最大时钟 192MHz • 12位死区时间 • 计数方式有：上计数，下计数，上/下计数 • 支持刹车功能 • 对每一个PWM输出管脚支持屏蔽功能和三态使能

EPWM	• 12个独立PWM 输出，16位计数器，12位预分频，最大时钟 192MHz • 可配成12个输入捕获，16位时间值 • 12位死区时间 • 计数方式有：上计数，下计数，上/下计数 • 可配置成3对互补模式PWM输出 • 有相位同步功能 • 支持刹车功能可自恢复 • 对每一个PWM输出管脚支持屏蔽功能和三态使能 • 可立即触发EADC/DAC启动转换 • 可延时触发EADC/DAC启动转换（M48xGC/M48xE8） • 硬件短路输出检测（M48xGC/M48xE8）
BPWM	• 如果定时器中断信号产生，支持芯片从空闲/掉电模式唤醒 • 8个独立PWM 输出，16位计数器，12位预分频，最大时钟 192MHz • 12位死区时间 • 计数方式有：上计数，下计数，上/下计数 • 支持刹车功能 • 对每一个PWM输出管脚支持屏蔽功能和三态使能
WDT 看门狗	• 18向上计数器 • 时钟源： LIRC (默认选择), HCLK/2048 和 LXT • 8 种时间选择 1.6ms ~ 26.0sec (看所选时钟) • 时钟选LIRC或LXT时，支持唤醒功能 • 时间到后，可复位，即可仅产生中断 • 可选复位延时： 1026、130、18 或 3 WDT_CLK • 支持上电复位后即启动计数功能
WWDT 看门狗	• 时钟源： HCLK/2048 (默认选择) 和 LIRC • 带11位预分频器的6位计数器 • 空闲和掉电模式下停止计数
模拟接口	

EADC（M48xID/M48xGA）	• 一組12位精度，19路ADC输入，采样率5 MSPS • 3个内部通道：VBAT,带隙电压，温度传感器 • 支持外部VREF或片内参考电压: 1.6V, 2.0V, 2.5V, 和 3.0V. • 两种节电模式:掉电模式，待命模式 • 支持校正功能. • 启动转换触发源：软件，引脚， Timer 0~3 溢出, PWM 触发 • 可配置EADC采样时间 • 模块 0~3双缓存 • 支持 PDMA 控制
EADC（M48xGC/M48xE8）	• 一組12位精度，19路ADC输入，最高采样率5 MSPS，最多支持16路单端输入通道或8对差分输入通道，可以保证10位精度 • 一組12位精度，16路ADC输入，最高采样率5 MSPS，最多支持16路单端输入通道或8对差分输入通道，可以保证10位精度 • 3个内部通道：VBAT,带隙电压，温度传感器 • 支持外部VREF或片内参考电压: 1.6V, 2.0V, 2.5V, 和 3.0V. • 两种节电模式:掉电模式，待命模式 • 支持校正功能. • 启动转换触发源：软件，引脚， Timer 0~3 溢出, PWM 触发 • 可配置EADC采样时间 • 模块 0~3双缓存 • 支持同时触发模式 • 支持 PDMA 控制
DAC	• 12位DAC, 1MSPS 转换速率 • 支持8位模式 • 建立时间 8us • 缓存模式最大输出电压： AVDD -0.2V • 启动转换触发源：软件，Timer 0~3, EPWM 和DAC外部触发引脚. • 两路 DAC可同步转换（M48xID/M48xGA） • 支持 PDMA 模式
模拟比较器	• 两个轨对轨比较器 • 正输入端有四个引脚选择. • 负输入端可选：引脚、带隙电压源、16级AVDD分压、 VREF 和 DAC 输出 • 可选低功耗模式，速度可编程 • 输出变化时可产生中断 • 支持唤醒功能 • 支持 PWM 的Brake功能，可逐周期控制 PWM

	• 支持窗比较模式和窗锁存功能 • 斯密特回差电压可编程: 0mV, 10mV, 20mV 和 30mV.
运放（M48xID/M48xGA）	• 3 个运放 • 输出可通过一个斯密特缓冲后触发中断
通信接口	
低功耗UART	• UART波特率最高 17.45MHz • 波特率自动测量功能和补偿功能 • 支持低功耗模式，在系统时钟关闭时，仍可用 LXT(32.768 KHz) 以 9600bps 工作 • 16字节 FIFOs，可配置触发字节数 • 支持硬件流控 CTS，RTS • 支持红外模式 IrDA • UART0，UART1 支持 LIN 功能 • 支持 RS-485 9位模式和方向控制 • 支持在空闲模式唤醒功能： nCTS唤醒, Rx唤醒, 接收FIFO数据个数到门限值唤醒， RS-485 地址匹配唤醒. • 支持硬件或软件把RTS 引脚配置成485的收发自动控制 • 支持唤醒功能 • 可配置8位接收 FIFO超时定时器 • 支持Break错，帧错，奇偶校验错，FiFo溢出检测功能 • 支持 PDMA 收发
智能卡接口	• ISO-7816-3接口，兼容 ISO-7816-3 T=0, T=1 • 支持 UART功能 • 收发各4字节缓存 • 位时长可配置 (11 ETU ~ 266 ETU) • 一个24位定时器两个8位定时器可用于复位应答序列 (ATR) 和等待时间定时 • 支持反向约定 • 出错后重发功能 • 硬件附着/分离序列处理功能 • 硬件热复位序列 • 拔卡后有硬件分离功能
I^2C	• 3路 I^2C，支持主从模式 • 支持标准速率 (100 kbps), 快速 (400 kbps) 和快速加模式 (1 Mbps)和高速模式 (3.4 Mbps) • 支持10位模式 • 支持多种速率

	• 从机支持多个地址 (四个从机地址加掩码) • 支持 SMBus， PMBus • 支持多地址掉电唤醒 • 支持 DMA传输
SPI主机（M48xID/M48xGA）	• 外接SPI 闪存最大32M 字节，支持标准的1位、2位和4位I/O传输模式，最高96 Mbit/s. • 有32KB高速缓存 cache • 支持16位金钥保护代码 • 代码从SPI到RAM内存间互传输，支持DMA传输 • 支持 CPU 直接从 SPI 闪存读. • 支持 SPI 主机协议，位长可配置为8, 16, 24, 32 • 突发模式可完成四次连续收发 • 支持eXcute-In-Place (XIP)
Quad SPI	• QSPI控制器，支持主从模式，在VDD =2.7V~3.6V最高96 Mbit/s. • 支持2线和4线传输 • 支持1或2数据通道半双工传输（M48xID/M48xGA） • 支持1数据通道半双工传输（M48xGC/M48xE8） • 支持双边沿数据发送（M48xGC/M48xE8） • 支持仅接收模式 • 收发位长可配置为 8 ~ 32 • 收发各8级FiFo • 高低位在前可配置 • 支持字节重排序功能 • 支持字节或字挂起功能 • 支持 3线模式，无片选 • 支持 PDMA 模式
SPI/ I^2S	• SPI/ I^2S 控制器，支持主从模式， • SPI/ I^2S收发各4级32位（或8级16位）FiFo • 收发都可用 DMA传输 SPI • 主从模式都支持最高96 Mbit/s@ 2.7V-3.6V • SPI收发位长可配置为 8 ~ 32 • SPI高低位在前可配置 • SPI支持字节重排序功能 • SPI支持字节或字挂起功能 • SPI支持半双工传输 I^2S

	• 支持单声道和立体声 • 支持位长： 8, 16, 24和 32位 • 支持 PCM 模式 A和模式B, I2S和高位调整格式
I²S	• 一个 I²S接口，可与外部音频CODEC相连 • 支持位长： 8, 16, 24和 32位 • 支持单声道和立体声 • 收发各16级FiFo • 支持 I²S 协议: 菲利普标准，高位和低位调整格式 • 支持PCM 协议: PCM 标准, 高位和低位调整格式 • PCM 协议支持 TDM 多通道传输,通道数可配置为 2, 4, 6, 8 • 收发都可用PDMA传输
USCI多用串口 （M48xID/M48xGA）	• 两个多用串口，支持 UART, SPI 和 I²C 模式 • TX，RX 单字节缓存 UART: • 一个发送缓存，两个接收缓存 • 支持硬件流控和可设置的流量控制触发水平 • 支持9位传输 • 通过内建捕获功能可测对方波特率 • 支持唤醒功能 • 支持DMA传输 SPI: • 支持主机模式或从机模式 • 一个发送缓存，两个接收缓存 • 附带16级缓存 • 传输位长 4~16位 (四线模式仅支持 8 ~16) • 高低位在前可配 • 支持字挂起 • 支持 3线模式，无片选 • 片选支持唤醒功能 • 支持 DMA传输 I²C: • 支持主从模式 • 一个发送缓存，两个接收缓存 • 支持标准速率 (100 kbps), 快速 (400 kbps) 和高速模式 (1 Mbps) • 支持10位地址 • 支持10位超时 • 支持总线监听.

	• 支持数据线跳变或地址匹配唤醒 • 支持多地址识别 • 支持设备地址标志 • 建立和保持时间可配置
CAN 2.0	• CAN 控制器 • CAN 协议 2.0A/ B • 速率可达1M bit/s • 32 个消息队列,每个队列都有消息掩码 • 可配置的FiFo模式 • 对于定时触应用可禁止重发 • 支持中断 • 支持掉电唤醒
SD 卡接口	• SD卡接口,兼容SD卡v2.0 • 支持最高50 MHz，在3.3V操作下实现200 Mbps。 • 支持单数据线和4数据线模式 • 3.3V时时钟最高可达48 MHz ，传输速率达 192 Mbps • 专门的 DMA 控制 • 支持 SD, SDHC和 SDIO 卡. • 支持 DMA 的分散聚集模式，以加速系统与SD/SDHC/SDIO卡之间的数据传输.
EBI外总线	• 支持三个内存块Bank，三个支持极性控制的片选引脚 • 每个内存块Bank可达 1 M字节, 实际可访问空间依芯片封装而定 • 数据位宽支持8/16位 • 16条数据线时支持字节写 • 支持地址数据复用模式 • 支持地址和数据分离模式 • 可连接 i80接口的LCD • 支持 PDMA 模式
GPIO	• 四种输入输出模式:准双向，推挽输出，OD输出，高阻输入 • 输入电平可选TTL/斯密特 • 可配置边沿或电平中断 • 独立上/下拉电阻控制 • 大电流输出驱动 • 软件可选速率 • 支持5V容限除了模拟管脚
控制接口	

QEI正交编码计数接口	• 两路正交编码计数器. • 两个计数输入: QEI_A, QEI_B 和一个定位输入 QEI_INDEX • 支持2/4倍自由计数模式和比较计数模式 • 与ECAP配合可测量脉宽
ECAP输入捕获定时器	• 支持最大两个输入捕获定时/计数单元，24位上数定时器/计数器 • 每个单元有3个捕获通道，各自独立时间值锁存器 • 计数器支持捕获复位和捕获重装功能. • 可捕获上沿，下沿，和双沿都捕获 • 支持比较/匹配功能.
高级接口	
全速USB 2.0OTG	全速USB 2.0 OTG • 支持USB OTG规范2.0版本 • 内建全速 USB 2.0 OTG 物理层 • 可配置为仅主机，仅设备，由ID端选择 全速USB 2.0 主机 • 支持USB规范1.1版本 • 支持OHCI规范 1.0版本 • 支持全速(12Mbps) 和低速 (1.5Mbps). • 支持控制传输，批量传输，中断传输，等时传输 • 支持端口路径逻辑将全速，低速设备接入OHCI 控制器. • 一个根集线器 Root Hub. • 支持电源控制和电流检测. • 实时传输支持DMA 传输. 全速USB 2.0设备 • 兼容USB 2.0全速规范 • 总线不活动超3ms后支持总线挂起 • 支持12个端点，1K字节数据缓存 • 一个中断向量，四个中断事件：唤醒中断, 插拔中断,数据中断和总线中断 • 远程唤醒功能 • 支持无晶振模式(M48xGC/M48xE8) • USB2.0连接电源管理（M48xID/M48xGA）
高速USB 2.0 OTG（M48xID/M48xGA）	高速USB 2.0 OTG • 支持USB OTG规范2.0版本 • 内建高速 USB 2.0 OTG 物理层 • 可配置为仅主机，仅设备，由ID端选择

	USB 2.0主机 • 支持USB规范V2.0版本 • 兼容 EHCI V1.0 • 兼容 OHCI V1.0 • 支持高速(480Mbps), 全速 (12Mbps) , 低速 (1.5Mbps) • 支持端口路径逻辑将全速，低速设备接入OHCI 控制器. • 支持控制，批量，中断，等时和分割传输 • 一个根集线器 Root Hub. • 集成了一个FS/LS路由控制器 • 内建 DMA 高速USB 2.0设备 • 支持USB规范V2.0版本 • 支持12个端点，每个都可配成批量，中断，等时模式输入或输出 • 4096 字节缓存 • 端点数据包最大 1024 字节 • 输入端点三种控制模式：自动有效模式，手支有效模式，飞行模式 • 支持挂起，恢复和远程唤醒功能 • 支持 DMA 操作
Ethernet 以太网 （M48xID/M48xGA）	• 支持 IEEE 标准. 802.3 CSMA/CD 协议 • 支持以太网帧时间戳 IEEE Std. 1588 – 2002协议 • 支持半双工或全双工 10 /100 Mbps • 支持 RMII接口 • 支持流控暂停和远程暂停功能 • 支持长帧(超 1518 字节) 和短帧 (低于 64字节) 接收 • MAC 地址识别支持13 CAM • 支持魔法包唤醒功能 • 支持 MII 管理功能控制外部物理层 • 支持 DMA 功能
数字摄像头接口	
数字摄像头接口 (CCAP) (M48xGC/M48xE8)	• 支持CCIR601, CCIR656及4位数据接口的CMOS传感器 • 支持YUV4:2:2 及 RGB565数据格式输入 • 输出颜色数据格式支持YUV4:2:2, RGB565, RGB555 及 Y • 支持输出1位Y(亮度)，可设置8位门槛亮度值 • 支持图像裁剪及缩小
加密加速器	

Elliptic Curve Cryptography (ECC)（M48xID/M48xGA）	• 硬件ECC加速器. • 支持192位和256位密钥长度。 • 支持质数域 GF(p) 和二进制段 GF(2m) • 支持 NIST P-192, P-224, P-256, P-384 和 P-521 • 支持 NIST B-163, B-233, B-283, B-409 和 B-571 • 支持 NIST K-163, K-233, K-283, K-409 和 K-571 • GF(p) 和 GF(2m)支持点乘,加和加倍操作 • GF(p)支持加，减，乘，和模除操作
Advanced Encryption Standard(AES)	• 硬件 AES 加速 • 支持 FIPS NIST 197 • 支持 128, 192和 256位密钥 • 支持 ECB, CBC, CFB, OFB, CTR, CBC-CS1, CBC-CS2, 和 CBC-CS3 模式 • 兼容 NIST 800 38A
Data Encryption Standard (DES)（M48xID/M48xGA）	• 硬件DES加速 • 支持 ECB, CBC, CFB, OFB,和CTR 模式 • 支持 FIPS 46-3
Triple Data Encryption Standard (3DES)（M48xID/M48xGA）	• 硬件Triple DES加速 • 支持2密钥和3密钥模式 • 支持 ECB, CBC, CFB, OFB, 和 CTR 模式 • 支持 FIPS NIST 800-67 • 执行 X9.52 标准
Secure Hash Algorithm (SHA)	• 硬件SHA加速 • 支持SHA-160, SHA-224, SHA-256, SHA-384, 他 SHA-512（M48xID/M48xGA） • 支持SHA256（M48xID/M48xGA） • 兼容FIPS NIST 180, 180-2
keyed-Hash Message Authentication Code (HMAC)	• 硬件HMAC加速 • 支持 HMAC-SHA-160, HMAC-SHA-224, HMAC-SHA-256, HMAC-SHA-384, 和HMAC-SHA-512（M48xID/M48xGA） • 支持HMAC-SHA-256 (M48xID/M48xGA) • 兼容FIPS NIST 180, 180-2
真随机数发生器 (TRNG) (M48xGC/M48xE8)	• 每秒800个随机位

3 料号信息

3.1 封装类型

Part No.	QFN33	LQFP48	LQFP64	LQFP128	LQFP144
M481	M481ZGCAE M481ZIDAE M481ZE8AE	M481LGCAE M481LIDAE M481LE8AE	M481SGCAE M481SIDAE M481SE8AE M481SGCAE2A		
M482	M482ZGCAE M482ZIDAE M482ZE8AE	M482LGCAE M482LIDAE M482LE8AE	M482SGCAE M482SIDAE M482SE8AE	M482KGCAE M482KIDAE	
M483			M483SGCAE M483SIDAE M483SE8AE M483SGCAE2A	M483KGCAE M483KIDAE M483KGCAE2A	
M484			M484SIDAE M484SIDAE2U	M484KIDAE	
M485		M485LIDAE	M485SIDAE	M485KIDAE	
M487			M487SIDAE	M487KIDAE	M487JIDAE

3.2 M480 系列选型指南

3.2.1 M481 基础系列 (M481xIDAE)

PART NUMBER		M481		
		ZIDAE	LIDAE	SIDAE
Flash (KB)		512		
SRAM (KB)		160 (include 32 KB cache for XIP)		
ISP Loader ROM (KB)		4		
I/O		26	41	52
32-bit Timer		4		
Peripheral DMA		16		
Tamper		-	-	1
Connectivity	LPUART	6		
	ISO-7816	3		
	SPI Master	1 (Support XIP)		
	Quad SPI	1		
	SPI/I^{2}S	3	3	4
	I^{2}S	1		
	I^{2}C	3		
	USCI	2		
	CAN	-		
	LIN	2		
	SDHC	1	2	2
16-bit PWM		24		
QEI		1	2	2
ECAP		-	1	1
USB 2.0 FS OTG		-		
USB 2.0 HSOTG		-		
12-bit ADC		10	12	16
12-bit DAC		2		
Analog Comparator		2		
Operational Amplifier		1	2	2
Ethernet		-		
Cryptography		-		
External Bus Interface		-	√	√
Package		QFN 32	LQFP 48	LQFP 64

3.2.2 M481 基础系列 (M481xGCAE / M481xE8AE)

PART NUMBER		M481						
		ZE8AE	ZGCAE	LE8AE	LGCAE	SE8AE	SGCAE	SGCAE2A
Flash (KB) (Support XOM)		128	256	128	256	128	256	256
SRAM (KB)		64	128	64	128	64	128	128
ISP Loader ROM (KB)		4						
I/O		26		41		52		52
32-bit Timer		4						
Peripheral DMA		16						
Tamper		-		-		1		1
Connectivity	LPUART	8						
	ISO-7816	1						
	SPI Master	-						
	Quad SPI	2						
	SPI/I^{2}S	2		3		3		3
	I^{2}S	1						
	I^{2}C	3						
	USCI	-						
	CAN	-						
	LIN	2						
	SDHC	1						
16-bit PWM		24						
QEI		1		2		2		2
ECAP		-		1		1		1
USB 2.0 FS OTG		-						
USB 2.0 HS OTG		-						
12-bit ADC		10		12		16		**8+8**
12-bit DAC		1						
Analog Comparator		2						
Operational Amplifier		-						
Ethernet		-						
Cryptography		AES-256						
TRNG		√						
External Bus Interface		-		√		√		√
Camera Interface		-		-		√		√
Package		QFN33		LQFP 48		LQFP 64		LQFP 64

3.2.3 M482 USB FS OTG 系列 (M482xIDAE)

PART NUMBER		M482			
		ZIDAE	LIDAE	SIDAE	KIDAE
Flash (KB)		512			
SRAM (KB)		160 (include 32 KB cache for XIP)			
ISP Loader ROM(KB)		4			
I/O		26	41	52	100
32-bit Timer		4			
Peripheral DMA		16			
Tamper		-	-	1	6
Connectivity	LPUART	6			
	ISO-7816	3			
	SPI Master	1 (Support XIP)			
	Quad SPI	1			
	SPI/I^{2}S	3	3	4	4
	I^{2}S	1			
	I^{2}C	3			
	USCI	2			
	CAN	-			
	LIN	2			
	SDHC	2			
16-bit PWM		24			
QEI		1	2	2	2
ECAP		-	1	1	2
USB 2.0 FS OTG		√			
USB 2.0 HS OTG		-			
12-bit ADC		10	12	16	16
12-bit DAC		2			
Analog Comparator		2			
Operational Amplifier		1	2	2	3
Ethernet		-			
Cryptography		-			
External Bus Interface		-	√	√	√
Package		QFN33	LQFP 48	LQFP 64	LQFP 128

3.2.4 M482 USB FS OTG 系列 (M482xGCAE / M482xE8AE)

PART NUMBER		M482						
		ZE8AE	ZGCAE	LE8AE	LGCAE	SE8AE	SGCAE	KGCAE
Flash (KB) (Support XOM)		128	256	128	256	128	256	256
SRAM (KB)		64	128	64	128	64	128	128
ISP Loader ROM (KB)		4						
I/O		26		41		52		100
32-bit Timer		4						
Peripheral DMA		16						
Tamper		-		-		1		6
Connectivity	LPUART	8						
	ISO-7816	1						
	SPI Master	-						
	Quad SPI	2						
	SPI/I^{2}S	2		3		3		3
	I^{2}S	1						
	I^{2}C	3						
	USCI	-						
	CAN	-						
	LIN	2						
	SDHC	1						
16-bit PWM		24						
QEI		1		2		2		2
ECAP		-		1		1		2
USB 2.0 FS OTG		√ (Crystal-less)						
USB 2.0 HS OTG		-						
12-bit ADC		10		12		16		16
12-bit DAC		1						
Analog Comparator		2						
Operational Amplifier		-						
Ethernet		-						
Cryptography		AES-256						
TRNG		√						
External Bus Interface		-		√		√		√
Camera Interface		-		-		√		√
Package		QFN33		LQFP 48		LQFP 64		LQFP 128

3.2.5 M483 CAN 系列 (M483xIDAE)

PART NUMBER		M483	
		SIDAE	KIDAE
Flash (KB)		512	
SRAM (KB)		160 (include 32 KB cache for XIP)	
ISP Loader ROM (KB)		4	
I/O		44	100
32-bit Timer		4	
Peripheral DMA		16	
Tamper		1	6
Connectivity	LPUART	6	
	ISO-7816	3	
	SPI Master	1	
	Quad SPI	1	
	SPI/I^2S	4	
	I^2S	1	
	I^2C	3	
	USCI	2	
	CAN	2	
	LIN	2	
	SDHC	2	
16-bit PWM		24	
QEI		2	
ECAP		1	2
USB 2.0 FS OTG		-	√
USB 2.0 HSOTG		√	
12-bit ADC		16	
12-bit DAC		2	
Analog Comparator		2	
Operational Amplifier		2	3
Ethernet		-	
Cryptography		-	
External Bus Interface		√	
Package		LQFP 64	LQFP 128

3.2.6 M483 CAN 系列 (M483xGCAE / M483xE8AE)

PART NUMBER		M483				
		SE8AE	SGCAE	SGCAE2A	KGCAE	KGCAE2A
Flash (KB) (Support XOM)		128	256	256	256	
SRAM (KB)		64	128	128	128	
ISP Loader ROM (KB)		4				
I/O		52		52	100	
32-bit Timer		4				
Peripheral DMA		16				
Tamper		1		1	6	
Connectivity	LPUART	8				
	ISO-7816	1				
	SPI Master	-				
	Quad SPI	2				
	SPI/I^2S	3				
	I^2S	1				
	I^2C	3				
	USCI	-				
	CAN	2		2	3	
	LIN	2				
	SDHC	1				
16-bit PWM		24				
QEI		2				
ECAP		1		1	2	
USB 2.0 FS OTG		√ (Crystal-less)				
USB 2.0 HS OTG		-				
12-bit ADC		16		**8+8**	16	**16+8**
12-bit DAC		1				
Analog Comparator		2				
Operational Amplifier		-				
Ethernet		-				
Cryptography		AES-256				
TRNG		√				
External Bus Interface		√				
Camera Interface		√				
Package		LQFP 64		LQFP 64	LQFP 128	

3.2.7 M484USB HS OTG系列

PART NUMBER		M484		
		SIDAE	SIDAE2U	KIDAE
Flash (KB)		512		
SRAM (KB)		160 (include 32 KB cache for XIP)		
ISP Loader ROM (KB)		4		
I/O		44	44	100
32-bit Timer		4		
Peripheral DMA		16		
Tamper		1	1	6
Connectivity	LPUART	6		
	ISO-7816	3		
	SPI Master	1		
	Quad SPI	1		
	SPI/I^{2}S	4		
	I^{2}S	1		
	I^{2}C	3		
	USCI	2		
	CAN	-		
	LIN	2		
	SDHC	2		
16-bit PWM		24		
QEI		2		
ECAP		1	1	2
USB 2.0 FS OTG		-	√	√
USB 2.0 HSOTG		√		
12-bit ADC		16		
12-bit DAC		2		
Analog Comparator		2		
Operational Amplifier		2	2	3
Ethernet		-		
Cryptography		-		
External Bus Interface		√		
Package		LQFP 64	LQFP 64	LQFP 128

3.2.8 M485 Crypto 系列

PART NUMBER		M485		
		LIDAE	SIDAE	KIDAE
Flash (KB)		512		
SRAM (KB)		160 (include 32 KB cache for XIP)		
ISP Loader ROM (KB)		4		
I/O		41	44	100
32-bit Timer		4		
Peripheral DMA		16		
Tamper		-	1	6
Connectivity	LPUART	6		
	ISO-7816	3		
	SPI Master	1		
	Quad SPI	1		
	SPI/I^{2}S	3	4	4
	I^{2}S	1		
	I^{2}C	3		
	USCI	2		
	CAN	-		
	LIN	2		
	SDHC	2		
16-bit PWM		24		
QEI		2		
ECAP		1	1	2
USB 2.0 FS OTG		√	-	√
USB 2.0 HSOTG		-	√	√
12-bit ADC		12	16	16
12-bit DAC		2		
Analog Comparator		2		
Operational Amplifier		2	2	3
Ethernet		-		
Cryptography		√		
External Bus Interface		√		
Package		LQFP 48	LQFP 64	LQFP 128

3.2.9 M487 Ethernet 系列

PART NUMBER		M487		
		SIDAE	KIDAE	JIDAE
Flash (KB)		512		
SRAM (KB)		160		
ISP Loader ROM (KB)		4		
I/O		44	100	114
32-bit Timer		4		
Peripheral DMA		16		
Tamper		1	6	6
Connectivity	LPUART	6		
	ISO-7816	3		
	SPI Master	1		
	Quad SPI	1		
	SPI/I^{2}S	4		
	I^{2}S	1		
	I^{2}C	3		
	USCI	2		
	CAN	2		
	LIN	2		
	SDHC	2		
16-bit PWM		24		
QEI		2		
ECAP		1	2	2
USB 2.0 FS OTG		-	√	√
USB 2.0 HS OTG		√		
12-bit ADC		16		
12-bit DAC		2		
Analog Comparator		2		
Operational Amplifier		2	3	3
Ethernet		√		
Cryptography		√		
External Bus Interface		√		
Package		LQFP 64	LQFP 128	LQFP 144

3.3 M480 选型

M4	81	Z	G	D	A	E	2A
Core	Series	Package	Flash Size	SRAM Size	Revision	Temperature	Peripheral
Cortex®-M4F	81: Base 82: USB FS 83: CAN 84: USB HS 85: Crypto 87: Ethernet	Z: QFN33 (5x5 mm) L: LQFP48 (7x7 mm) C: WLCSP S: LQFP64 (7x7 mm) O: QFN88 (10x10 mm) V: LQFP100 (14x14 mm) K: LQFP128 (14x14 mm) J: LQFP144 (20x20 mm)	A: 8 Kbytes B: 16 Kbytes C: 32 Kbytes D: 64 Kbytes E: 128 Kbytes F: 192 Kbytes G: 256 Kbytes H: 384 Kbytes I: 512 Kbytes	1: 4 Kbytes 2: 8 Kbytes 3: 16 Kbytes 4: 20 Kbytes 5: 24 Kbytes 6: 32 Kbytes 7: 48 Kbytes 8: 64 Kbytes 9: 80 Kbytes A: 96 Kbytes B: 112 Kbytes C: 128 Kbytes D: 160 Kbytes		E:-40° C ~ 105° C	2A: 2 EADCs 2U: 2 USB ports

4 引脚配置

4.1 引脚配置

用户可以在第4章或使用NuTool - PinConfig找到引脚配置信息。The NuTool —— PinConfigure 包含所有NuMicro家族芯片系列型号，帮助用户正确配置 GPIO 功能.

4.1.1 QFN-33引脚图

对应料号: M481ZE8AE, M481ZGCAE, M481ZIDAE, M482ZE8AE, M482ZGCAE, M482ZIDAE

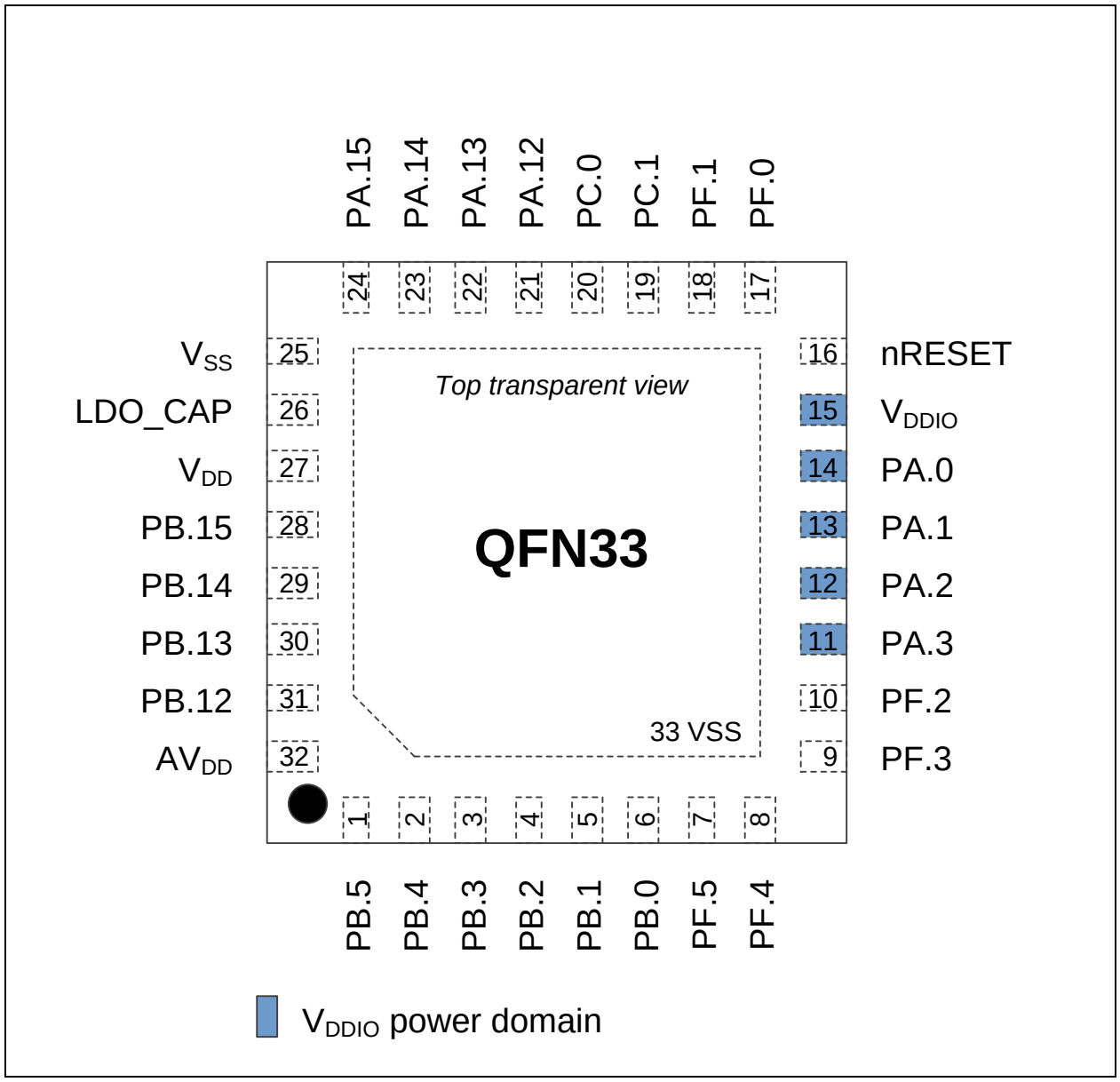


图 4.1-1QFN-33 引脚图 (0/1 USB FS)

4.1.2 LQFP-48引脚图 (0/1 USB FS)

对应料号: M481LE8AE, M481LGCAE, M481LIDAE, M482LE8AE, M482LGCAE, M482LIDAE, M485LIDAE

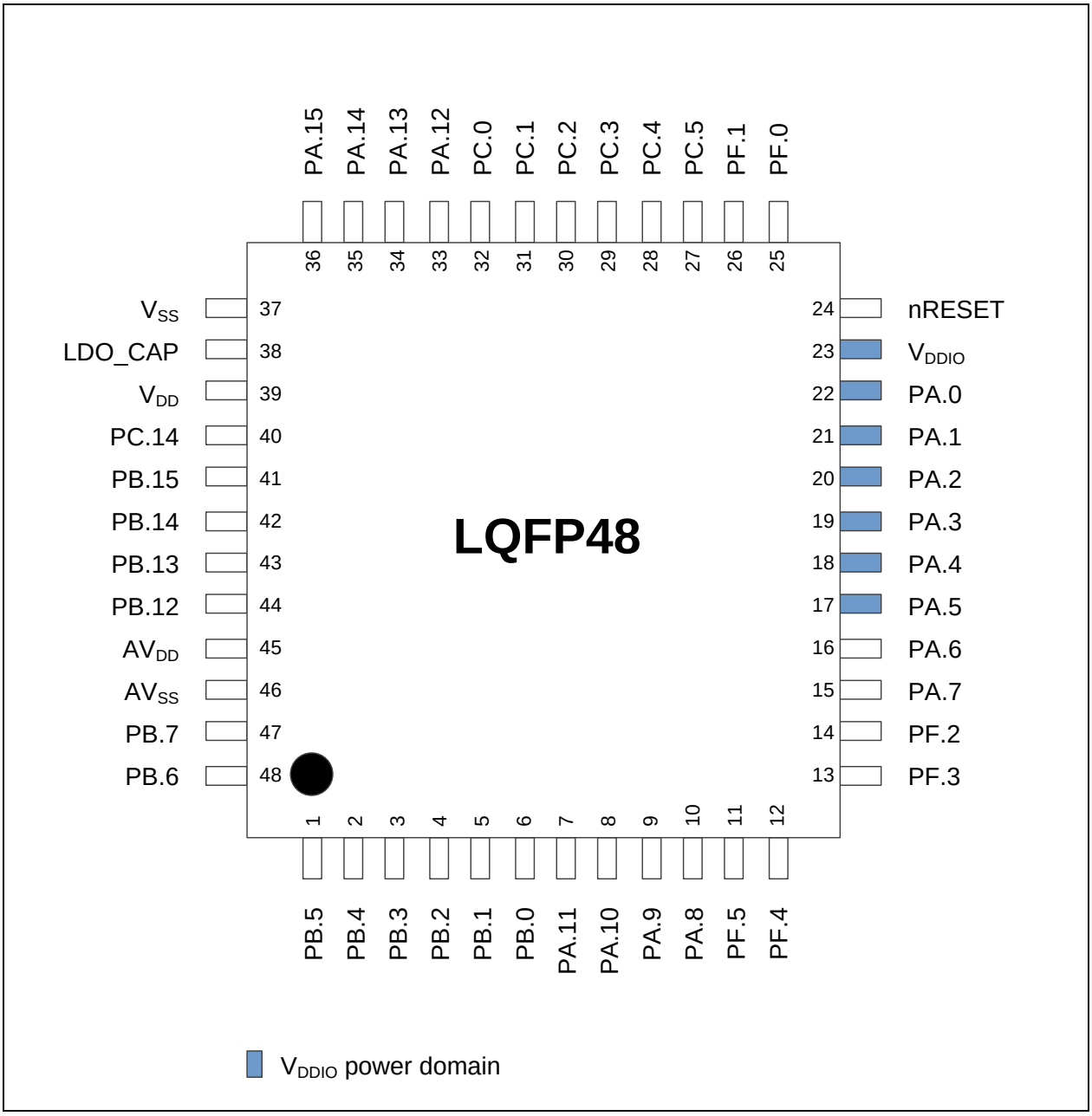


图 4.1-2LQFP-48 引脚图 (0/1 USB FS)

4.1.3 LQFP-64引脚图 (0/1 USB FS)

对应料号: M481SIDAE, M482SIDAE

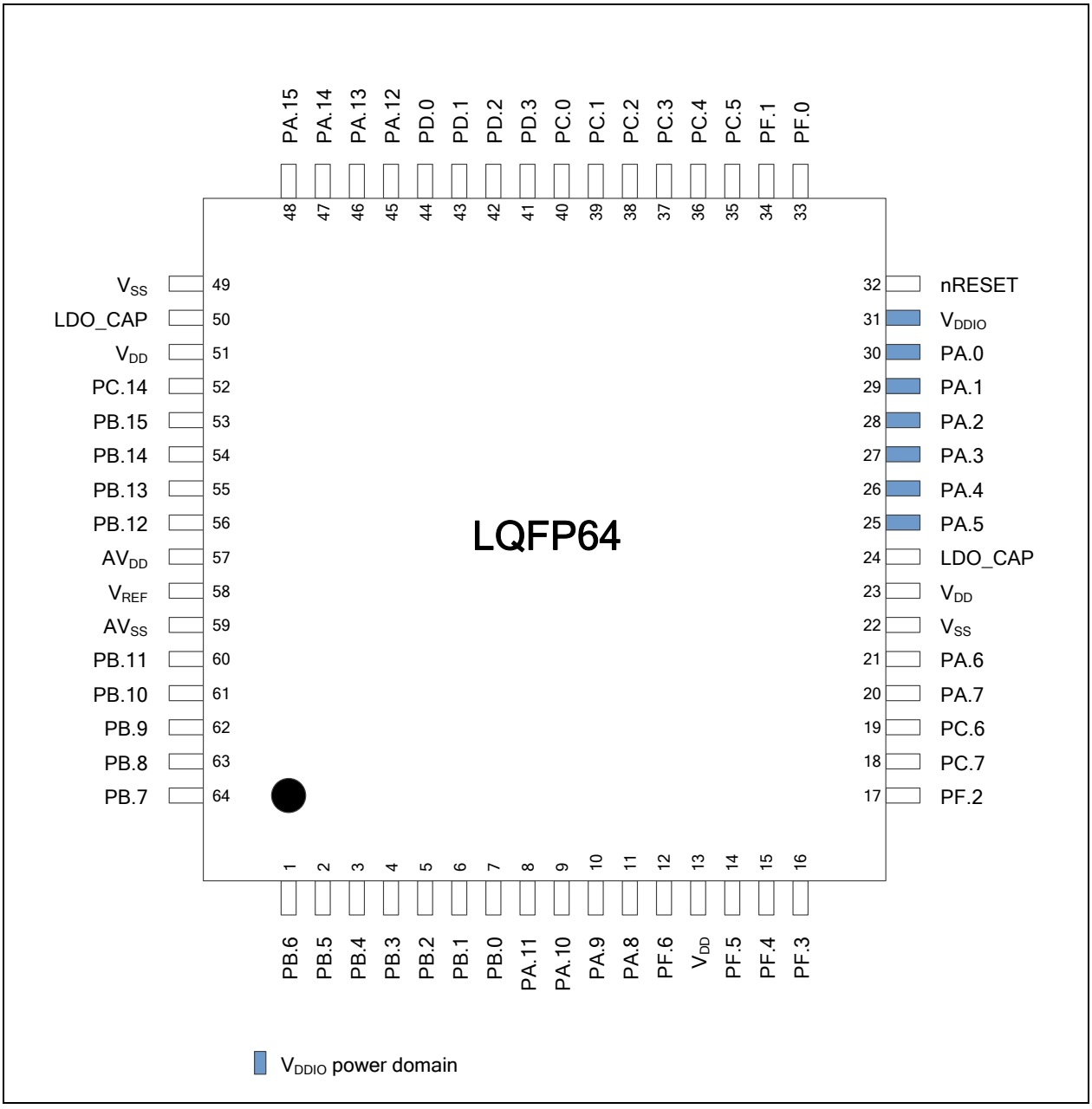


图 4.1-3LQFP-64 引脚图 (0/1 USB FS)

4.1.4 LQFP-64引脚图 (0/1 USB FS with V_{BAT})

对应料号：M481SE8AE, M481SGCAE, M481SGCAE2A, M482SE8AE, M482SGCAE, M483SE8AE, M483SGCAE, M483SGCAE2A

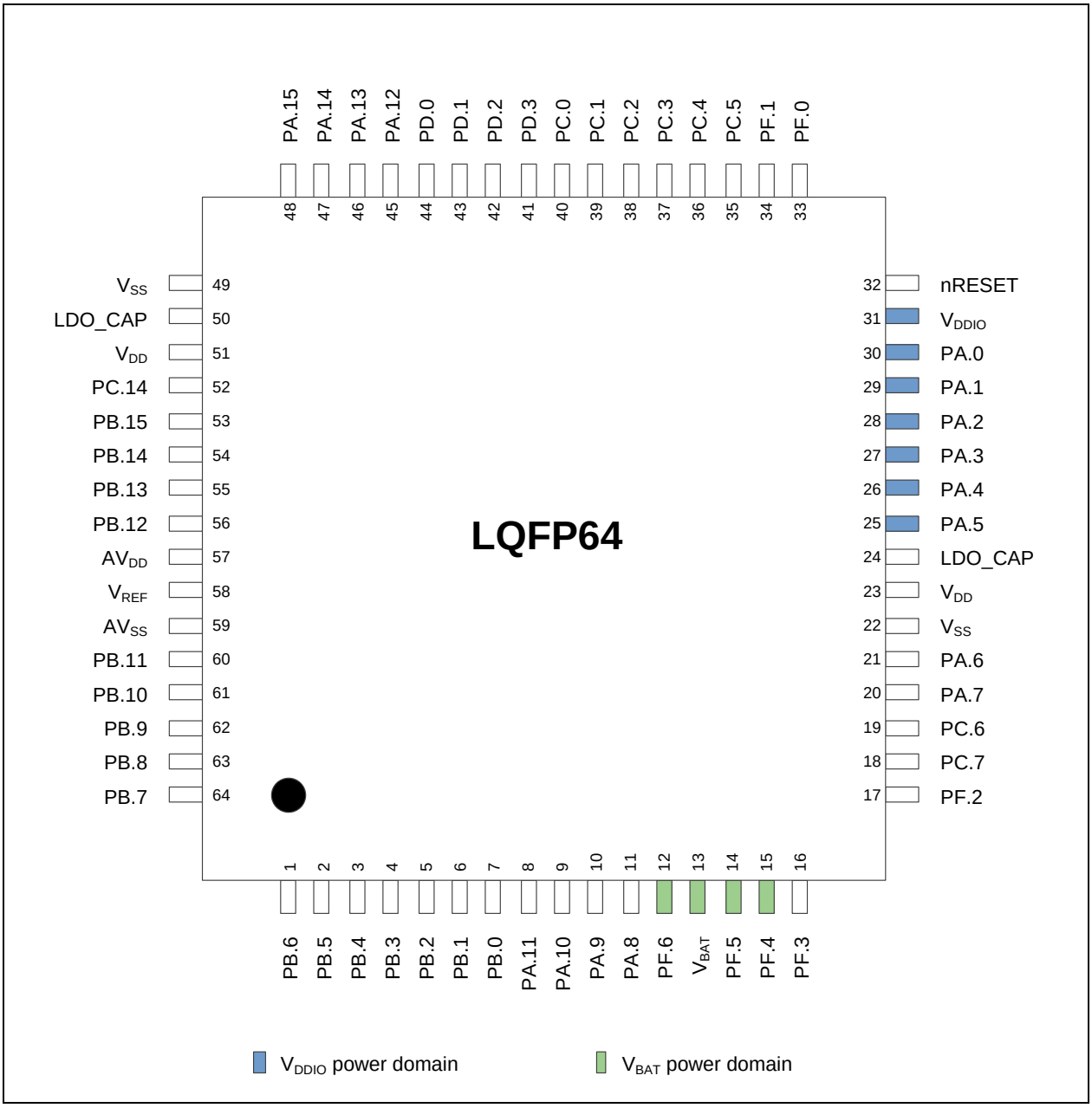


图 4.1-4LQFP-64 引脚图 (0/1 USB FS with V_{BAT})

4.1.5 LQFP-64引脚图(1 USB HS)

对应料号: M483SIDAE, M484SIDAE, M485SIDAE, M487SIDAE

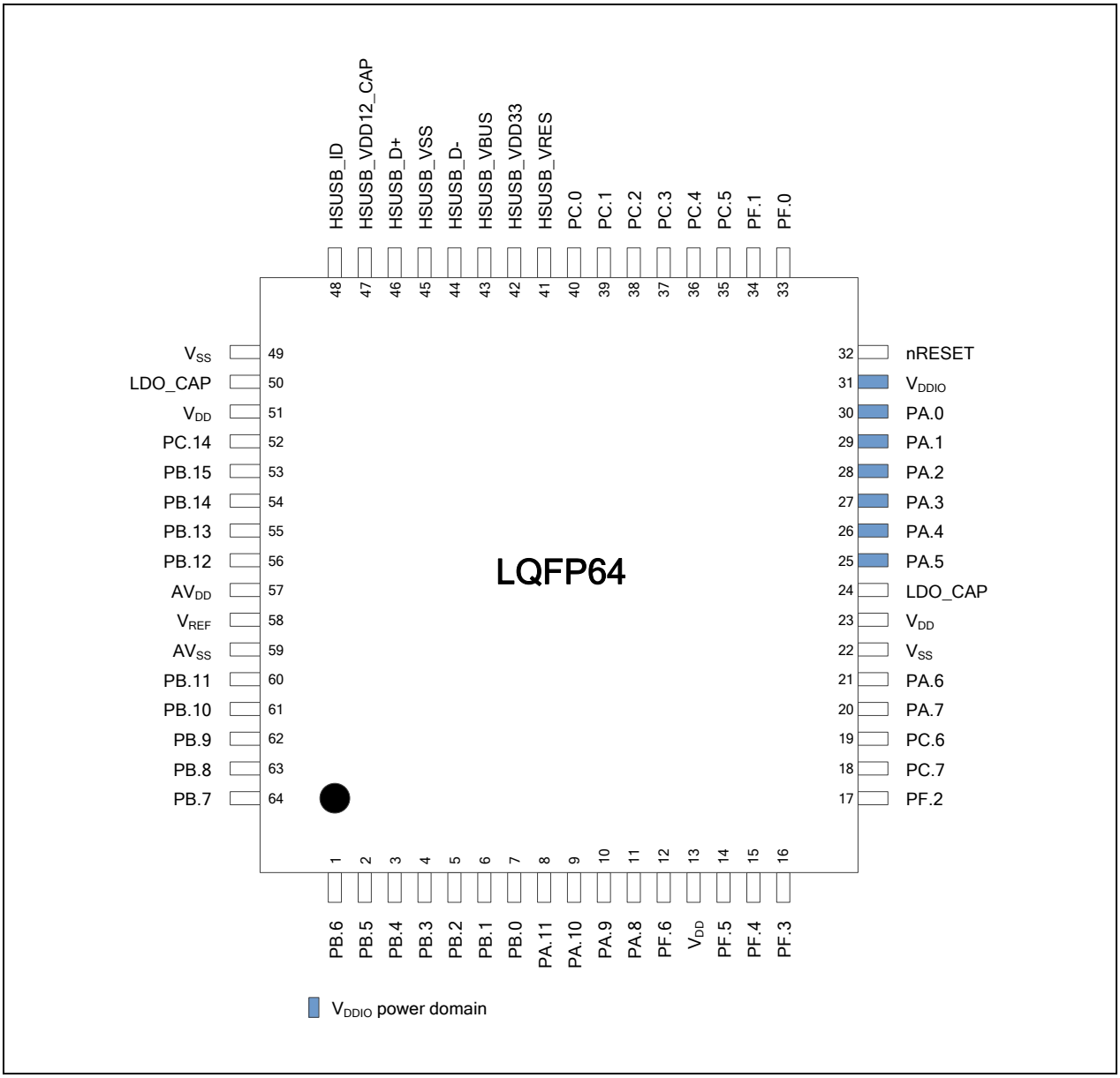


图 4.1-5LQFP-64 引脚图 (1 USB HS)

4.1.6 LQFP-64引脚图(USB FS + USB HS)

对应料号: M484SIDAE2U

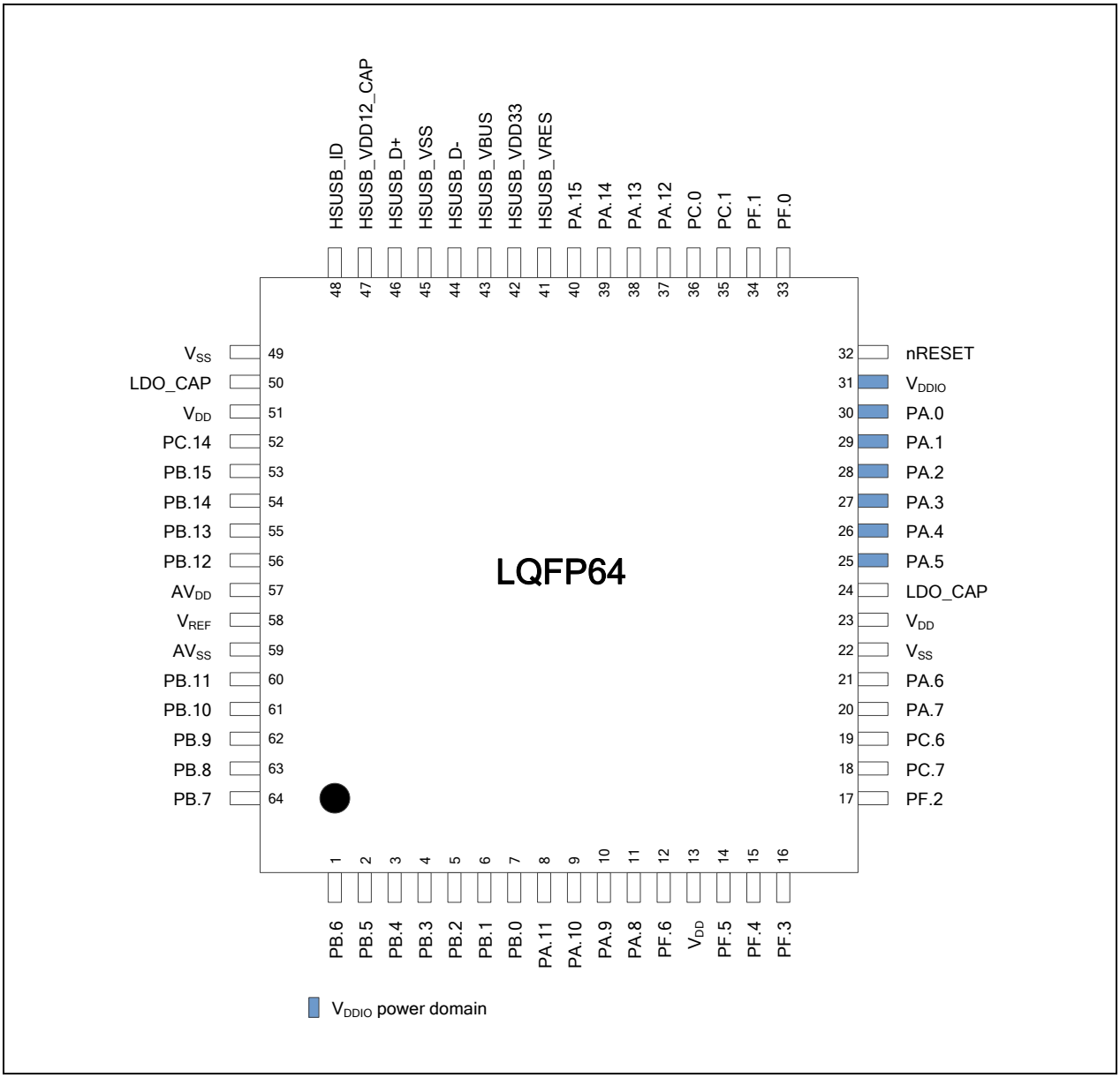


图 4.1-6LQFP-64 引脚图(USB FS + USB HS)

4.1.7 LQFP-128引脚图(1 USB FS)

对应料号: M482KGAAE, M482KIDAE

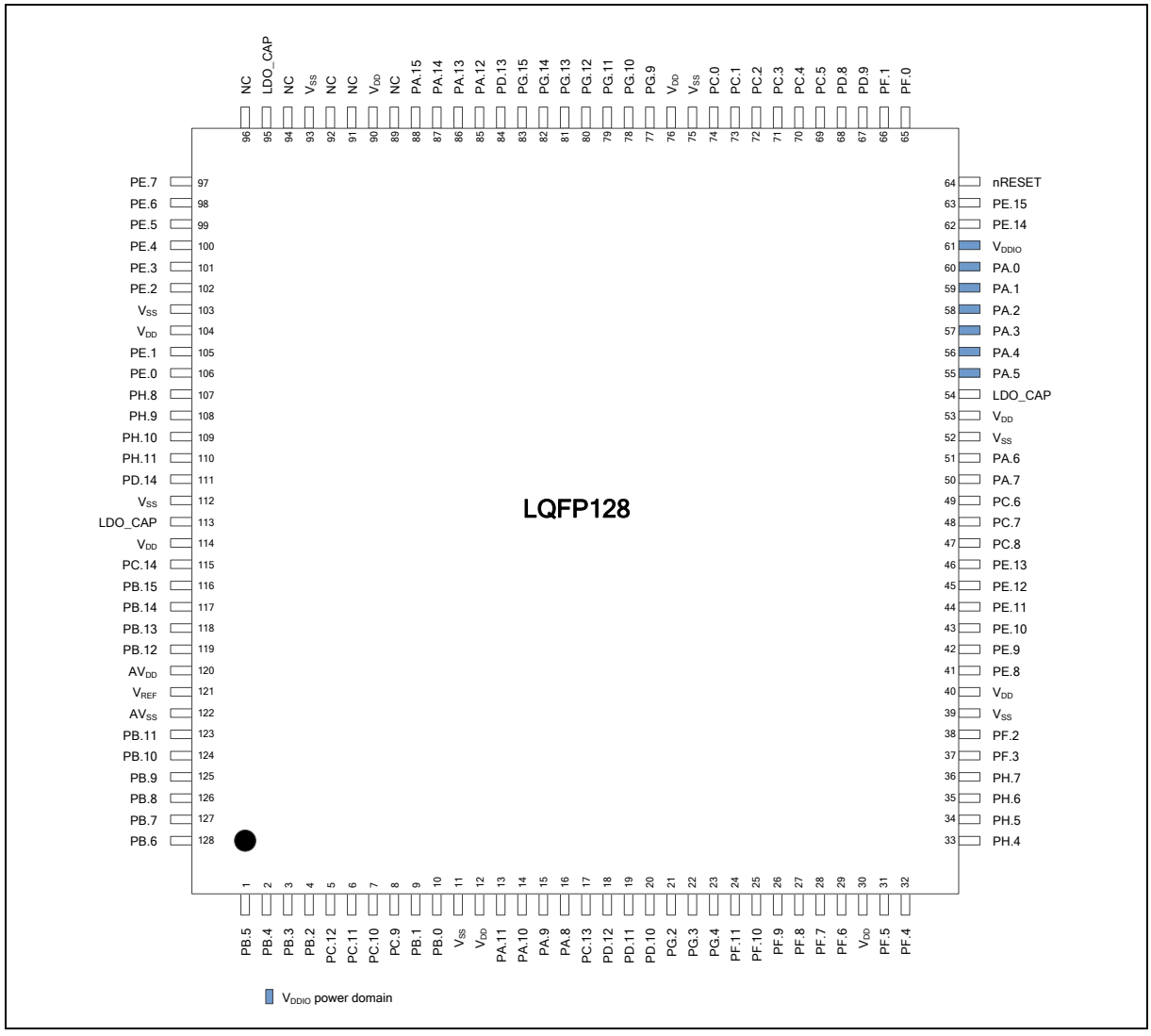


图 4.1-7LQFP-128 引脚图(1 USB FS)

4.1.8 LQFP-128引脚图(1 USB FS with V_{BAT})

对应料号: M483KGCAE, M483KGCAE2A

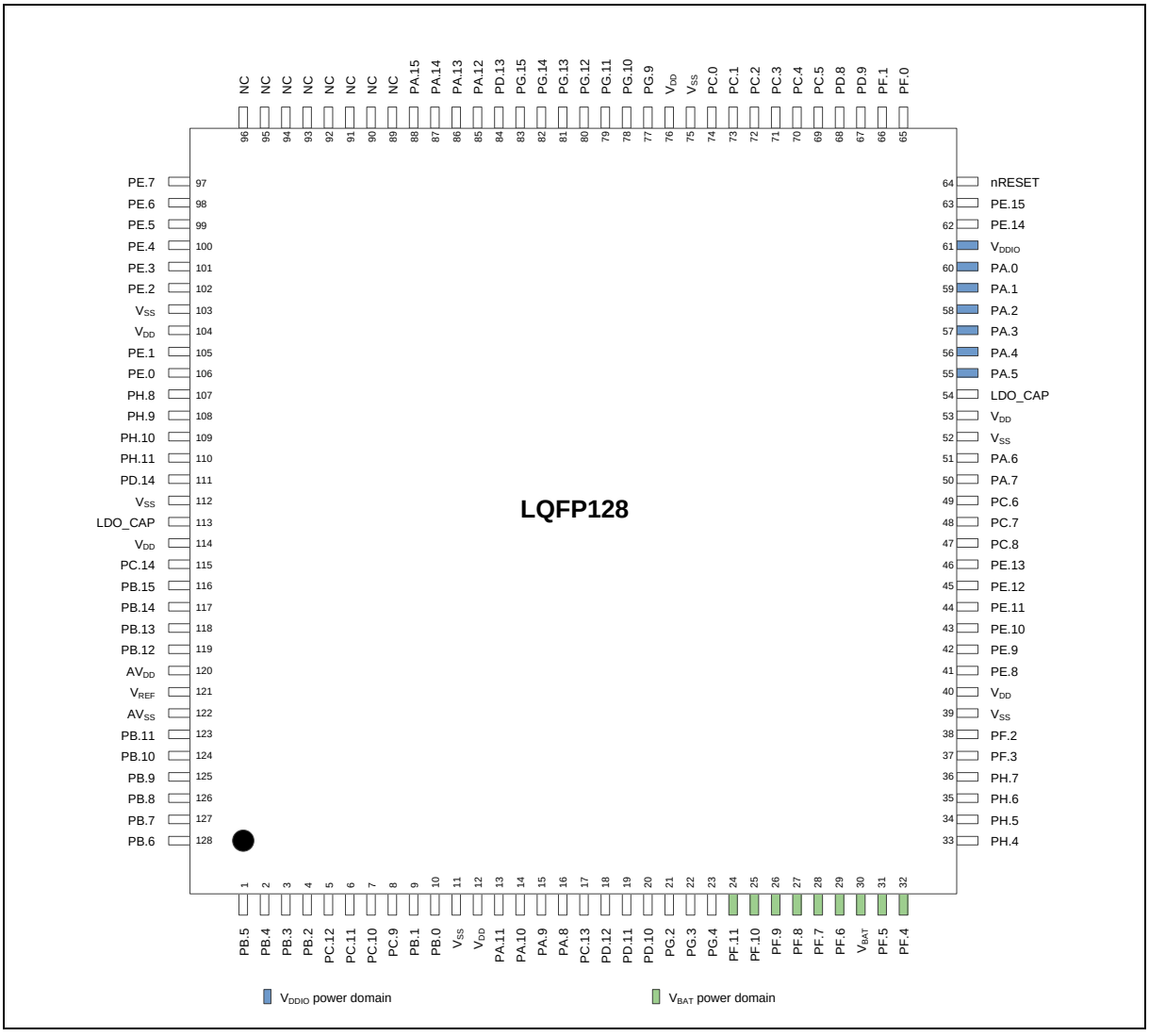


图 4.1-8LQFP-128 引脚图(1 USB FS)

4.1.9 LQFP-128引脚图(USB FS + USB HS)

对应料号: M483KIDAE, M484KIDAE, M485KIDAE, M487KIDAE

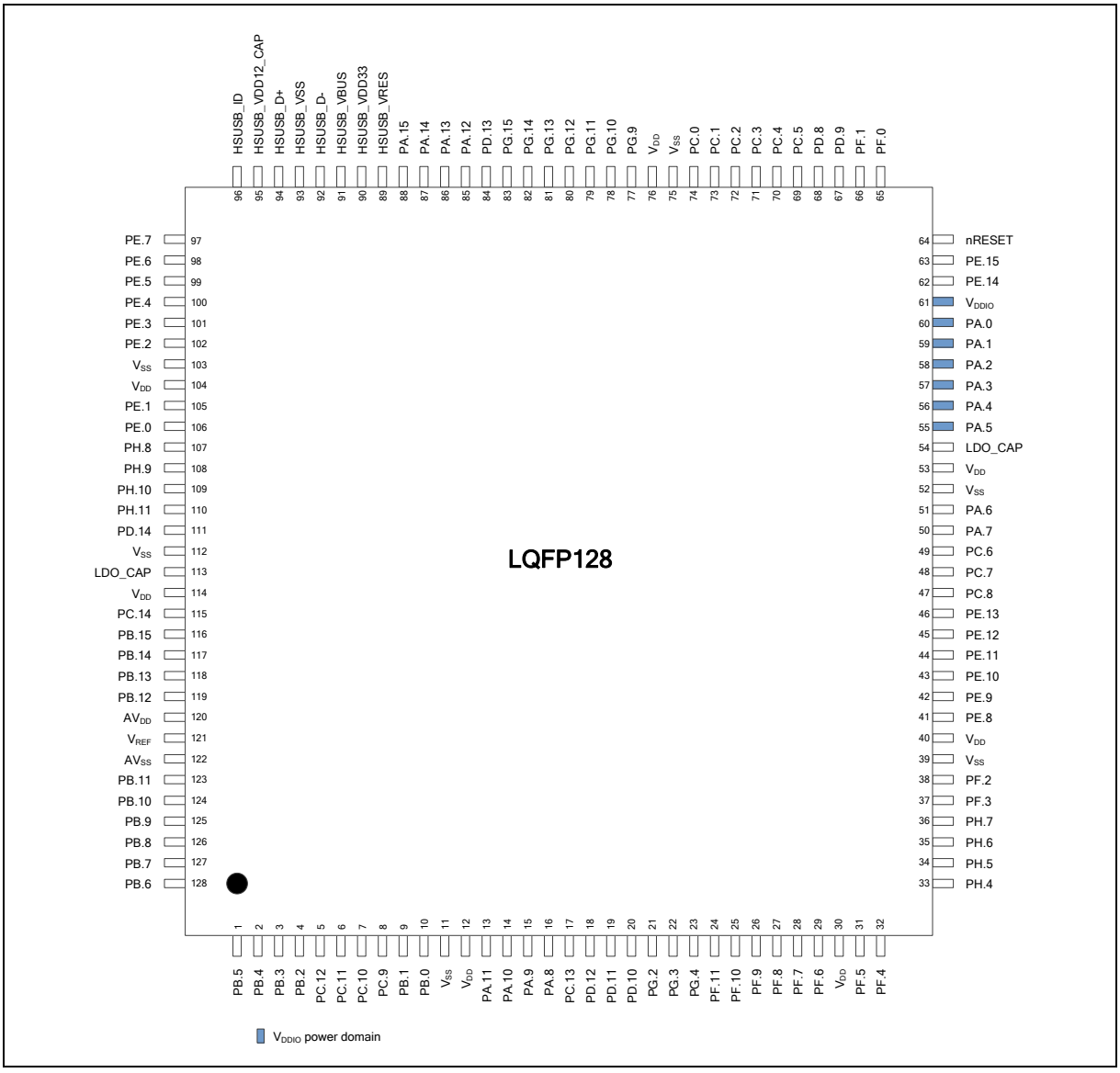


图 4.1-9LQFP-128 引脚图(USB FS + USB HS)

4.1.10 LQFP-144引脚图

对应料号: M487JIDAE

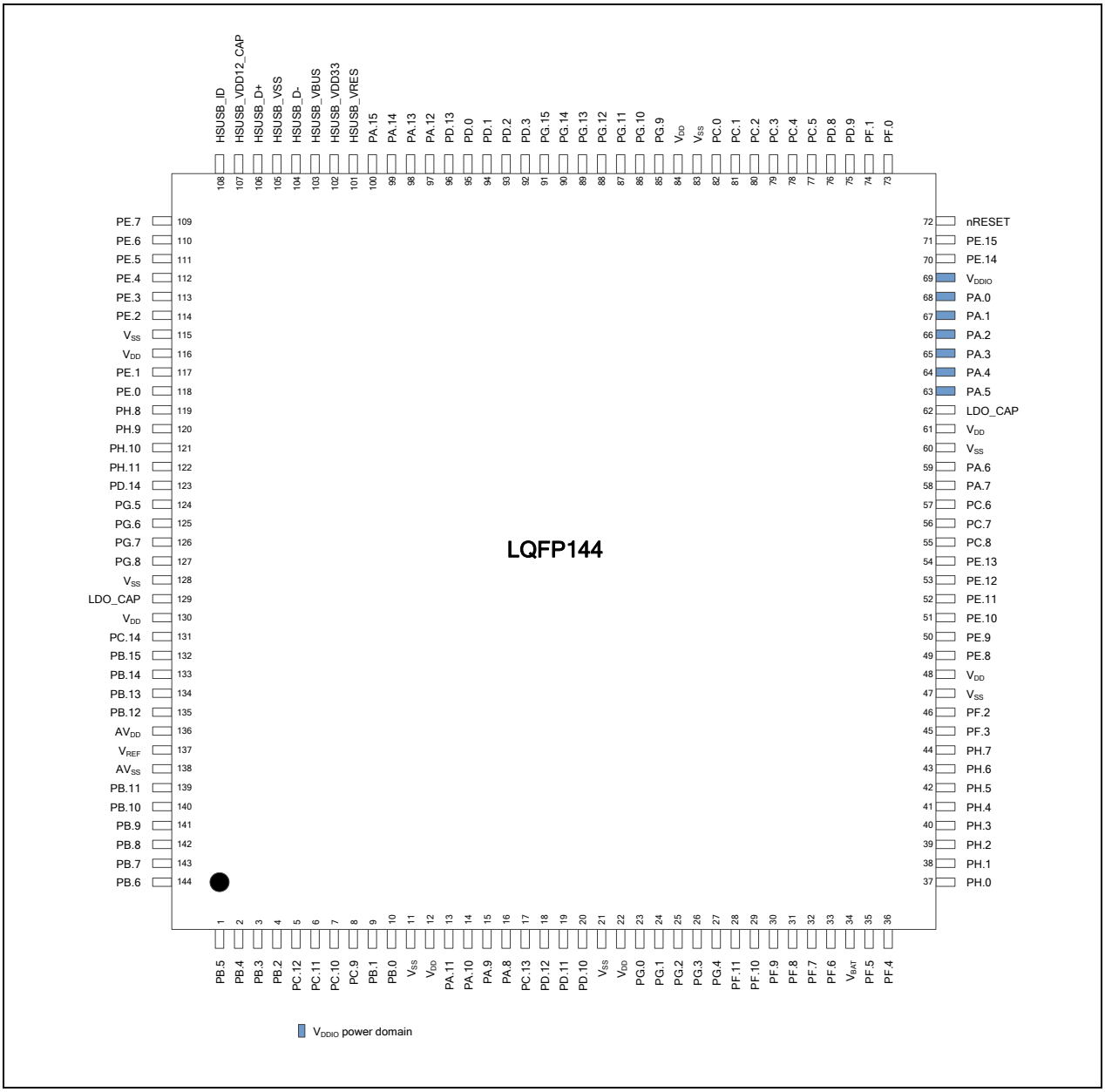


图 4.1-10LQFP-144 引脚图

4.2 M48xxGAAE/M48xxIDAE 管脚描述

4.2.1 M481 系列管脚描述

MFP* = 多功能管脚(请参考SYS_GPx_MFPL 和 SYS_GPx_MFPH)

PA.0 MFP0 意思是 SYS_GPA_MFPL[3:0] = 0x0.

PA.9 MFP5 意思是 SYS_GPA_MFPH[7:4] = 0x5.

32 管脚	48 管脚	64 管脚	管脚名称	类型	MFP	描述
	48	1	PB.6	I/O	MFP0	通用数字I/O脚
			EADC0_CH6	A	MFP1	EADC0通道6模拟输入
			EBI_nWRH	O	MFP2	EBI 高字节写使能输出管脚
			USCI1_DAT1	I/O	MFP4	USCI1数据1管脚.
			UART1_RXD	I	MFP6	UART1数据接收输入管脚
			SD1_CLK	O	MFP7	SD/SDIO1 时钟输出管脚
			EBI_nCS1	O	MFP8	EBI片选1输出管脚
			BPWM1_CH5	I/O	MFP10	EPWM1 通道5输出/捕获输入
			EPWM1_BRAKE1	I	MFP11	EPWM1 刹车1输入管脚
			EPWM1_CH5	I/O	MFP12	EPWM1 通道5输出/捕获输入
			INT4	I	MFP13	外部中断4输入管脚
			ACMP1_O	O	MFP15	模拟比较器1输出脚
1	1	2	PB.5	I/O	MFP0	通用数字I/O脚
			EADC0_CH5	A	MFP1	EADC0 通道5模拟输入
			ACMP1_N	A	MFP1	模拟比较器1负极输入管脚
			EBI_ADR0	O	MFP2	EBI地址总线位0
			SD0_DAT3	I/O	MFP3	SD/SDIO0数据线位3
			SPI1_MISO	I/O	MFP5	SPI1 MISO(主机输入，从机输出)管脚
			I^2C0_SCL	I/O	MFP6	I2C0 时钟管脚
			UART5_TXD	O	MFP7	UART5数据发送输出脚
			USCI1_CTL0	I/O	MFP8	USCI1 控制0管脚
			SC0_CLK	O	MFP9	智能卡0时钟管脚
			I^2S0_BCLK	O	MFP10	I2S0位时钟输出管脚
			EPWM0_CH0	I/O	MFP11	EPWM0 通道0输出/捕获输入

32 管脚	48 管脚	64 管脚	管脚名称	类型	MFP	描述
			TM0	I/O	MFP14	Timer0 事件计数输入/反转输出管脚
			INT0	I	MFP15	外部中断0输入管脚
2	2	3	PB.4	I/O	MFP0	通用数字I/O脚
			EADC0_CH4	A	MFP1	EADC0 通道4模拟输入
			ACMP1_P1	A	MFP1	模拟比较器1正极输入1脚
			EBI_ADR1	O	MFP2	EBI 地址总线位1
			SD0_DAT2	I/O	MFP3	SD/SDIO0 数据线位2
			SPI1_MOSI	I/O	MFP5	SPI1 MOSI (主机输出，从机输入) 管脚
			I^2C0_SDA	I/O	MFP6	I2C0 数据输入/输出管脚
			UART5_RXD	I	MFP7	UART5 数据接收输入管脚
			USCI1_CTL1	I/O	MFP8	USCI1 控制1管脚
			SC0_DAT	I/O	MFP9	智能卡0数据输入脚
			I^2S0_MCLK	O	MFP10	I2S0 主时钟输出脚
			EPWM0_CH1	I/O	MFP11	EPWM0 通道1输出/捕获输入脚
			TM1	I/O	MFP14	Timer1 事件计数输入/反转输出脚
			INT1	I	MFP15	外部中断1输入脚
3	3	4	PB.3	I/O	MFP0	通用数字I/O脚
			EADC0_CH3	A	MFP1	EADC0通道3模拟输入
			ACMP0_N	A	MFP1	模拟比较器0负极输入脚
			EBI_ADR2	O	MFP2	EBI 地址总线位2
			SD0_DAT1	I/O	MFP3	SD/SDIO0 数据线位1
			SPI1_CLK	I/O	MFP5	SPI1 串行时钟管脚
			UART1_TXD	O	MFP6	UART1 数据发送输出脚
			UART5_nRTS	O	MFP7	UART5 请求发送输出脚
			USCI1_DAT1	I/O	MFP8	USCI1数据1管脚.
			SC0_RST	O	MFP9	智能卡0复位脚
			I^2S0_DI	I	MFP10	I2S0 数据输入脚
			EPWM0_CH2	I/O	MFP11	EPWM0 通道2输出/捕获输入脚
			TM2	I/O	MFP14	Timer2 事件计数输入/反转输出脚
			INT2	I	MFP15	外部中断2输入脚

32 管脚	48 管脚	64 管脚	管脚名称	类型	MFP	描述
4	4	5	PB.2	I/O	MFP0	通用数字I/O脚
			EADC0_CH2	A	MFP1	EADC0 通道2 模拟输入
			ACMP0_P1	A	MFP1	模拟比较器0正极输入1脚
			OPA0_O	A	MFP1	运放0输出脚
			EBI_ADR3	O	MFP2	EBI 地址总线位3
			SD0_DAT0	I/O	MFP3	SD/SDIO0 数据线位0
			SPI1_SS	I/O	MFP5	SPI1 从机选择脚
			UART1_RXD	I	MFP6	UART1数据接收输入管脚
			UART5_nCTS	I	MFP7	UART5 禁止发送输入管脚
			USCI1_DAT0	I/O	MFP8	USCI1 数据0脚
			SC0_PWR	O	MFP9	智能卡0电源脚
			I^2S0_DO	O	MFP10	I^2S0 数据输出脚
			EPWM0_CH3	I/O	MFP11	EPWM0 通道3输出/捕获输入脚
			TM3	I/O	MFP14	Timer3 事件计数输入/反转输出脚
			INT3	I	MFP15	外部中断3输入脚
5	5	6	PB.1	I/O	MFP0	通用数字I/O脚
			EADC0_CH1	A	MFP1	EADC0 通道1 模拟输入
			OPA0_N	A	MFP1	运放0负极输入脚
			EBI_ADR8	O	MFP2	EBI 地址总线位 8.
			SD0_CLK	O	MFP3	SD/SDIO0 时钟输出脚
			SPI1_I2SMCLK	I/O	MFP5	SPI1 I^2S 主机时钟输出脚
			SPI3_I2SMCLK	I/O	MFP6	SPI3 I^2S 主机时钟输出脚
			UART2_TXD	O	MFP7	UART2 数据发送输出脚.
			USCI1_CLK	I/O	MFP8	USCI1 时钟脚
			I^2C1_SCL	I/O	MFP9	I^2C1 时钟脚
			I^2S0_LRCK	O	MFP10	I^2S0左右通道时钟输出脚.
			EPWM0_CH4	I/O	MFP11	EPWM0 通道4输出/捕获输入脚
			EPWM1_CH4	I/O	MFP12	EPWM1 通道4输出/捕获输入脚
			EPWM0_BRAKE0	I	MFP13	EPWM0 刹车0 输入脚
6	6	7	PB.0	I/O	MFP0	通用数字I/O脚

32 管脚	48 管脚	64 管脚	管脚名称	类型	MFP	描述
			EADC0_CH0	A	MFP1	EADC0 通道0 模拟输入
			OPA0_P	A	MFP1	运放 0 正极输入脚
			EBI_ADR9	O	MFP2	EBI 地址总线位 9.
			SD0_CMD	I/O	MFP3	SD/SDIO0 命令/响应脚
			UART2_RXD	I	MFP7	UART2 数据接收输入脚
			SPI0_I2SMCLK	I/O	MFP8	SPI0 I^2S 主机时钟输出脚
			I^2C1_SDA	I/O	MFP9	I^2C1 数据输入/输出脚
			EPWM0_CH5	I/O	MFP11	EPWM0 通道5 输出/捕获输入
			EPWM1_CH5	I/O	MFP12	EPWM1 通道5输出/捕获输入
			EPWM0_BRAKE1	I	MFP13	EPWM0 刹车1 输入脚
	7	8	PA.11	I/O	MFP0	通用数字I/O脚
			ACMP0_P0	A	MFP1	模拟比较器 0 正极输入脚0
			EBI_nRD	O	MFP2	EBI 读使能输出脚
			SC2_PWR	O	MFP3	智能卡 2电源脚
			SPI2_SS	I/O	MFP4	SPI2 从机选择脚
			SD1_DAT3	I/O	MFP5	SD/SDIO1 数据线位3.
			USCI0_CLK	I/O	MFP6	USCI0 时钟脚
			I^2C2_SCL	I/O	MFP7	I^2C2 时钟脚
			BPWM0_CH0	I/O	MFP9	BPWM0 通道0 输出/捕获输入
			EPWM0_SYNC_OUT	O	MFP10	EPWM0计数器同步触发输出脚
			TM0_EXT	I/O	MFP13	Timer0 事件计数输入/反转输出管脚
			DAC1_ST	I	MFP14	DAC1 外部触发输入
	8	9	PA.10	I/O	MFP0	通用数字I/O脚
			ACMP1_P0	A	MFP1	模拟比较器 1 正极输入脚0
			OPA1_O	A	MFP1	运放 1输出脚
			EBI_nWR	O	MFP2	EBI 写使能输出脚
			SC2_RST	O	MFP3	智能卡 2 复位脚
			SPI2_CLK	I/O	MFP4	SPI2 串行时钟脚
			SD1_DAT2	I/O	MFP5	SD/SDIO1 数据线位2.
			USCI0_DAT0	I/O	MFP6	USCI0 数据0脚

32 管脚	48 管脚	64 管脚	管脚名称	类型	MFP	描述
			I^2C2_SDA	I/O	MFP7	I^2C2 数据输入/输出脚
			BPWM0_CH1	I/O	MFP9	BPWM0 通道1 输出/捕获输入
			QEI1_INDEX	I	MFP10	正交编码器1 索引输入
			ECAP0_IC0	I	MFP11	增强型捕获计数器 0 输入脚0
			TM1_EXT	I/O	MFP13	Timer1 事件计数输入/反转输出脚
			DAC0_ST	I	MFP14	DAC0 外部触发输入
			SWDH_CLK	O	MFP15	串行调试总线主时钟输出
	9	10	PA.9	I/O	MFP0	通用数字I/O脚
			OPA1_N	A	MFP1	运放 1 负极输入脚
			EBI_MCLK	O	MFP2	EBI 外部时钟输出脚.
			SC2_DAT	I/O	MFP3	智能卡 2 数据管脚.
			SPI2_MISO	I/O	MFP4	SPI2 MISO（主机输入，从机输出）脚.
			SD1_DAT1	I/O	MFP5	SD/SDIO1 数据线位1.
			USCI0_DAT1	I/O	MFP6	USCI0 数据1 管脚.
			UART1_TXD	O	MFP7	UART1 数据发送输出脚
			BPWM0_CH2	I/O	MFP9	BPWM0 通道2 输出/捕获输入
			QEI1_A	I	MFP10	正交编码器1 相位A输入
			ECAP0_IC1	I	MFP11	增强型捕获计数器 0 输入 1 管脚.
			TM2_EXT	I/O	MFP13	Timer2 事件计数输入/反转输出脚
			SWDH_DAT	I/O	MFP15	串行调试总线主机数据输入/输出管脚
	10	11	PA.8	I/O	MFP0	通用数字I/O脚
			OPA1_P	A	MFP1	运放 1 正极输入脚
			EBI_ALE	O	MFP2	EBI地址锁存使能输出脚
			SC2_CLK	O	MFP3	智能卡 2 时钟脚
			SPI2_MOSI	I/O	MFP4	SPI2 MOSI (主机输出，从机输入) 管脚.
			SD1_DAT0	I/O	MFP5	SD/SDIO1 数据线位0.
			USCI0_CTL1	I/O	MFP6	USCI0 控制1 管脚.
			UART1_RXD	I	MFP7	UART1数据接收输入管脚
			BPWM0_CH3	I/O	MFP9	BPWM0 通道3输出/捕获输入脚
			QEI1_B	I	MFP10	正交编码器1 相位B 输入

32 管脚	48 管脚	64 管脚	管脚名称	类型	MFP	描述
			ECAP0_IC2	I	MFP11	增强型捕获计数器 0 输入 2 管脚.
			TM3_EXT	I/O	MFP13	Timer3 事件计数输入/反转输出脚
			INT4	I	MFP15	外部中断4输入管脚
		12	PF.6	I/O	MFP0	通用数字I/O脚
			EBI_ADR19	O	MFP2	EBI 地址总线位 19.
			SC0_CLK	O	MFP3	智能卡0时钟管脚
			I^2S0_LRCK	O	MFP4	I^2S0左右通道时钟输出脚.
			SPI0_MOSI	I/O	MFP5	SPI0 MOSI (主机输出，从机输入) 管脚.
			UART4_RXD	I	MFP6	UART4 数据接收输入脚
			EBI_nCS0	O	MFP7	EBI片选0输出脚
			TAMPER0	I/O	MFP10	TAMPER循环检测管脚 0.
		13	V_{DD}	P	MFP0	I/O和内部PLL及数字电路的LDO电源
7	11	14	PF.5	I/O	MFP0	通用数字I/O脚
			UART2_RXD	I	MFP2	UART2 数据接收输入脚
			UART2_nCTS	I	MFP4	UART2 禁止发送输入管脚
			BPWM0_CH4	I/O	MFP8	BPWM0 通道4输出/捕获输入脚
			EPWM0_SYNC_OUT	O	MFP9	EPWM0计数器同步触发输出脚
			X32_IN	I	MFP10	外部 32.768 kHz 晶振输入脚
			EADC0_ST	I	MFP11	EADC0 外部触发输入
8	12	15	PF.4	I/O	MFP0	通用数字I/O脚
			UART2_TXD	O	MFP2	UART2 数据发送输出脚.
			UART2_nRTS	O	MFP4	UART2 请求发送输出脚
			BPWM0_CH5	I/O	MFP8	BPWM0 通道5 输出/捕获输入
			X32_OUT	O	MFP10	外部 32.768 kHz 晶振输出脚
9	13	16	PF.3	I/O	MFP0	通用数字I/O脚
			EBI_nCS0	O	MFP2	EBI片选0输出脚
			UART0_TXD	O	MFP3	UART0 数据发送输出脚.
			I^2C0_SCL	I/O	MFP4	I2C0 时钟管脚
			XT1_IN	I	MFP10	外部 4~24 MHz (高速) 晶振输入脚
			BPWM1_CH0	I/O	MFP11	BPWM1 通道0 输出/捕获输入

32 管脚	48 管脚	64 管脚	管脚名称	类型	MFP	描述
10	14	17	PF.2	I/O	MFP0	通用数字I/O脚
			EBI_nCS1	O	MFP2	EBI片选1输出管脚
			UART0_RXD	I	MFP3	UART0 数据接收输入脚
			I^{2}C0_SDA	I/O	MFP4	I2C0 数据输入/输出管脚
			QSPI0_CLK	I/O	MFP5	QSPI0 串行时钟脚
			XT1_OUT	O	MFP10	外部 4~24 MHz (高速) 晶振输出脚
			BPWM1_CH1	I/O	MFP11	BPWM1 通道1 输出/捕获输入
		18	PC.7	I/O	MFP0	通用数字I/O脚
			EBI_AD9	I/O	MFP2	EBI地址/数据总线位9.
			SPI1_MISO	I/O	MFP4	SPI1 MISO(主机输入，从机输出)管脚
			UART4_TXD	O	MFP5	UART4 数据发送输出脚.
			SC2_PWR	O	MFP6	智能卡 2电源脚
			UART0_nCTS	I	MFP7	UART0 禁止发送输入管脚
			I^{2}C1_SMBAL	O	MFP8	I^{2}C1 SMBus SMBALTER 管脚
			EPWM1_CH2	I/O	MFP11	EPWM1 通道2 输出/捕获输入
			BPWM1_CH0	I/O	MFP12	BPWM1 通道0 输出/捕获输入
			TM0	I/O	MFP14	Timer0 事件计数输入/反转输出管脚
			INT3	I	MFP15	外部中断3输入脚
		19	PC.6	I/O	MFP0	通用数字I/O脚
			EBI_AD8	I/O	MFP2	EBI地址/数据总线位8.
			SPI1_MOSI	I/O	MFP4	SPI1 MOSI (主机输出，从机输入) 管脚
			UART4_RXD	I	MFP5	UART4 数据接收输入脚
			SC2_RST	O	MFP6	智能卡 2 复位脚
			UART0_nRTS	O	MFP7	UART0 请求发送输出脚
			I^{2}C1_SMBSUS	O	MFP8	I^{2}C1 SMBus SMBSUS 管脚 (PMBus 控制管脚)
			EPWM1_CH3	I/O	MFP11	EPWM1 通道3输出/捕获输入脚
			BPWM1_CH1	I/O	MFP12	BPWM1 通道1 输出/捕获输入
			TM1	I/O	MFP14	Timer1 事件计数输入/反转输出脚
			INT2	I	MFP15	外部中断2输入脚
	15	20	PA.7	I/O	MFP0	通用数字I/O脚

32 管脚	48 管脚	64 管脚	管脚名称	类型	MFP	描述
			EBI_AD7	I/O	MFP2	EBI地址/数据总线位7.
			SPI1_CLK	I/O	MFP4	SPI1 串行时钟管脚
			SC2_DAT	I/O	MFP6	智能卡 2 数据管脚.
			UART0_TXD	O	MFP7	UART0 数据发送输出脚.
			I^2C1_SCL	I/O	MFP8	I^2C1 时钟脚
			EPWM1_CH4	I/O	MFP11	EPWM1 通道4输出/捕获输入脚
			BPWM1_CH2	I/O	MFP12	BPWM1 通道2 输出/捕获输入
			ACMP0_WLAT	I	MFP13	模拟比较器 0窗口锁存输入管脚
			TM2	I/O	MFP14	Timer2 事件计数输入/反转输出脚
			INT1	I	MFP15	外部中断1输入脚
	16	21	PA.6	I/O	MFP0	通用数字I/O脚
			EBI_AD6	I/O	MFP2	EBI地址/数据总线位6.
			SPI1_SS	I/O	MFP4	SPI1 从机选择脚
			SD1_nCD	I	MFP5	SD/SDIO1 卡检测输入管脚
			SC2_CLK	O	MFP6	智能卡 2 时钟脚
			UART0_RXD	I	MFP7	UART0 数据接收输入脚
			I^2C1_SDA	I/O	MFP8	I^2C1 数据输入/输出脚
			EPWM1_CH5	I/O	MFP11	EPWM1 通道5输出/捕获输入
			BPWM1_CH3	I/O	MFP12	BPWM1 通道3输出/捕获输入脚
			ACMP1_WLAT	I	MFP13	模拟比较器 1窗口锁存输入管脚
			TM3	I/O	MFP14	Timer3 事件计数输入/反转输出脚
			INT0	I	MFP15	外部中断0输入管脚
		22	V_{SS}	P	MFP0	数字电路地管脚
		23	V_{DD}	P	MFP0	I/O和内部PLL及数字电路的LDO电源
		24	LDO_CAP	A	MFP0	LDO输出脚
	17	25	PA.5	I/O	MFP0	通用数字I/O脚
			SPIM_D2	I/O	MFP2	SPIM QuadI/O模式下数据2管脚
			QSPI0_MISO1	I/O	MFP3	QSPI0 MISO1（主机输入，从机输出）脚.
			SPI1_I2SMCLK	I/O	MFP4	SPI1 I^2S 主机时钟输出脚
			SD1_CMD	I/O	MFP5	SD/SDIO1 命令/响应脚

32 管脚	48 管脚	64 管脚	管脚名称	类型	MFP	描述
			SC2_nCD	I	MFP6	智能卡 2 卡检测管脚.
			UART0_nCTS	I	MFP7	UART0 禁止发送输入管脚
			UART5_TXD	O	MFP8	UART5数据发送输出脚
			I^{2}C0_SCL	I/O	MFP9	I2C0 时钟管脚
			BPWM0_CH5	I/O	MFP12	BPWM0 通道5 输出/捕获输入
			EPWM0_CH0	I/O	MFP13	EPWM0 通道0输出/捕获输入
			QEI0_INDEX	I	MFP14	正交编码器0 索引输入
	18	26	PA.4	I/O	MFP0	通用数字I/O脚
			SPIM_D3	I/O	MFP2	SPIM QuadI/O模式下数据3管脚
			QSPI0_MOSI1	I/O	MFP3	QSPI0 MOSI1 (主机输出，从机输入) 管脚.
			SPI0_I2SMCLK	I/O	MFP4	SPI0 I^{2}S 主机时钟输出脚
			SD1_CLK	O	MFP5	SD/SDIO1 时钟输出管脚
			SC0_nCD	I	MFP6	智能卡 0 卡检测管脚.
			UART0_nRTS	O	MFP7	UART0 请求发送输出脚
			UART5_RXD	I	MFP8	UART5 数据接收输入管脚
			I^{2}C0_SDA	I/O	MFP9	I2C0 数据输入/输出管脚
			BPWM0_CH4	I/O	MFP12	BPWM0 通道4输出/捕获输入脚
			EPWM0_CH1	I/O	MFP13	EPWM0 通道1输出/捕获输入脚
			QEI0_A	I	MFP14	正交编码器0 相位A输入
11	19	27	PA.3	I/O	MFP0	通用数字I/O脚
			SPIM_SS	I/O	MFP2	SPIM 从机选择脚
			QSPI0_SS	I/O	MFP3	QSPI0 从机选择脚
			SPI0_SS	I/O	MFP4	SPI0 从机选择脚
			SD1_DAT3	I/O	MFP5	SD/SDIO1 数据线位3.
			SC0_PWR	O	MFP6	智能卡0电源脚
			UART4_TXD	O	MFP7	UART4 数据发送输出脚.
			UART1_TXD	O	MFP8	UART1 数据发送输出脚
			I^{2}C1_SCL	I/O	MFP9	I^{2}C1 时钟脚
			BPWM0_CH3	I/O	MFP12	BPWM0 通道3输出/捕获输入脚
			EPWM0_CH2	I/O	MFP13	EPWM0 通道2输出/捕获输入脚

32 管脚	48 管脚	64 管脚	管脚名称	类型	MFP	描述
			QEI0_B	I	MFP14	正交编码器0 相位B 输入
12	20	28	PA.2	I/O	MFP0	通用数字I/O脚
			SPIM_CLK	I/O	MFP2	SPIM 串行时钟脚
			QSPI0_CLK	I/O	MFP3	QSPI0 串行时钟脚
			SPI0_CLK	I/O	MFP4	SPI0 串行时钟脚
			SD1_DAT2	I/O	MFP5	SD/SDIO1 数据线位2.
			SC0_RST	O	MFP6	智能卡0复位脚
			UART4_RXD	I	MFP7	UART4 数据接收输入脚
			UART1_RXD	I	MFP8	UART1数据接收输入管脚
			I^2C1_SDA	I/O	MFP9	I^2C1 数据输入/输出脚
			BPWM0_CH2	I/O	MFP12	BPWM0 通道2 输出/捕获输入
			EPWM0_CH3	I/O	MFP13	EPWM0 通道3输出/捕获输入脚
13	21	29	PA.1	I/O	MFP0	通用数字I/O脚
			SPIM_MISO	I/O	MFP2	SPIM MISO（主机输入，从机输出）脚.
			QSPI0_MISO0	I/O	MFP3	QSPI0 MISO0（主机输入，从机输出）脚.
			SPI0_MISO	I/O	MFP4	SPI0 MISO（主机输入，从机输出）脚.
			SD1_DAT1	I/O	MFP5	SD/SDIO1 数据线位1.
			SC0_DAT	I/O	MFP6	智能卡0数据输入脚
			UART0_TXD	O	MFP7	UART0 数据发送输出脚.
			UART1_nCTS	I	MFP8	UART1 禁止发送输入管脚
			I^2C2_SCL	I/O	MFP9	I^2C2 时钟脚
			BPWM0_CH1	I/O	MFP12	BPWM0 通道1 输出/捕获输入
			EPWM0_CH4	I/O	MFP13	EPWM0 通道4输出/捕获输入脚
			DAC1_ST	I	MFP15	DAC1 外部触发输入
14	22	30	PA.0	I/O	MFP0	通用数字I/O脚
			SPIM_MOSI	I/O	MFP2	SPIM MOSI (主机输出，从机输入) 管脚.
			QSPI0_MOSI0	I/O	MFP3	QSPI0 MOSI0 (主机输出，从机输入) 管脚.
			SPI0_MOSI	I/O	MFP4	SPI0 MOSI (主机输出，从机输入) 管脚.
			SD1_DAT0	I/O	MFP5	SD/SDIO1 数据线位0.
			SC0_CLK	O	MFP6	智能卡0时钟管脚

32 管脚	48 管脚	64 管脚	管脚名称	类型	MFP	描述
			UART0_RXD	I	MFP7	UART0 数据接收输入脚
			UART1_nRTS	O	MFP8	UART1 请求发送输出脚
			I^2C2_SDA	I/O	MFP9	I^2C2 数据输入/输出脚
			BPWM0_CH0	I/O	MFP12	BPWM0 通道0 输出/捕获输入
			EPWM0_CH5	I/O	MFP13	EPWM0 通道5 输出/捕获输入
			DAC0_ST	I	MFP15	DAC0 外部触发输入
15	23	31	V_{DDIO}	P	MFP0	PE.1, PE.8~PE.13 的电源
16	24	32	nRESET	I	MFP0	外部复位输入:低电平有效,有内部上拉，设置此管脚为低将复位初始状态
17	25	33	PF.0	I/O	MFP0	通用数字I/O脚
			UART1_TXD	O	MFP2	UART1 数据发送输出脚
			I^2C1_SCL	I/O	MFP3	I^2C1 时钟脚
			BPWM1_CH0	I/O	MFP12	BPWM1 通道0 输出/捕获输入
			ICE_DAT	O	MFP14	串行调试总线数据管脚.
18	26	34	PF.1	I/O	MFP0	通用数字I/O脚
			UART1_RXD	I	MFP2	UART1数据接收输入管脚
			I^2C1_SDA	I/O	MFP3	I^2C1 数据输入/输出脚
			BPWM1_CH1	I/O	MFP12	BPWM1 通道1 输出/捕获输入
			ICE_CLK	I	MFP14	串行调试总线时钟脚
	27	35	PC.5	I/O	MFP0	通用数字I/O脚
			EBI_AD5	I/O	MFP2	EBI地址/数据总线位5.
			SPIM_D2	I/O	MFP3	SPIM QuadI/O模式下数据2管脚
			QSPI0_MISO1	I/O	MFP4	QSPI0 MISO1（主机输入，从机输出）脚.
			UART2_TXD	O	MFP8	UART2 数据发送输出脚.
			I^2C1_SCL	I/O	MFP9	I^2C1 时钟脚
			UART4_TXD	O	MFP11	UART4 数据发送输出脚.
			EPWM1_CH0	I/O	MFP12	EPWM1 通道0 输出/捕获输入
	28	36	PC.4	I/O	MFP0	通用数字I/O脚
			EBI_AD4	I/O	MFP2	EBI地址/数据总线位4.
			SPIM_D3	I/O	MFP3	SPIM QuadI/O模式下数据3管脚
			QSPI0_MOSI1	I/O	MFP4	QSPI0 MOSI1 (主机输出，从机输入) 管脚.

32 管脚	48 管脚	64 管脚	管脚名称	类型	MFP	描述
			SC1_nCD	I	MFP5	智能卡 1 卡检测管脚.
			I²S0_BCLK	O	MFP6	I2S0位时钟输出管脚
			SPI1_I2SMCLK	I/O	MFP7	SPI1 I²S 主机时钟输出脚
			UART2_RXD	I	MFP8	UART2 数据接收输入脚
			I²C1_SDA	I/O	MFP9	I²C1 数据输入/输出脚
			UART4_RXD	I	MFP11	UART4 数据接收输入脚
			EPWM1_CH1	I/O	MFP12	EPWM1 通道1 输出/捕获输入
	29	37	PC.3	I/O	MFP0	通用数字I/O脚
			EBI_AD3	I/O	MFP2	EBI地址/数据总线位3.
			SPIM_SS	I/O	MFP3	SPIM 从机选择脚
			QSPI0_SS	I/O	MFP4	QSPI0 从机选择脚
			SC1_PWR	O	MFP5	智能卡 1电源脚
			I²S0_MCLK	O	MFP6	I2S0 主时钟输出脚
			SPI1_MISO	I/O	MFP7	SPI1 MISO(主机输入，从机输出)管脚
			UART2_nRTS	O	MFP8	UART2 请求发送输出脚
			I²C0_SMBAL	O	MFP9	I²C0 SMBus SMBALTER 管脚
			UART3_TXD	O	MFP11	UART3 数据发送输出脚.
			EPWM1_CH2	I/O	MFP12	EPWM1 通道2 输出/捕获输入
	30	38	PC.2	I/O	MFP0	通用数字I/O脚
			EBI_AD2	I/O	MFP2	EBI地址/数据总线位2.
			SPIM_CLK	I/O	MFP3	SPIM 串行时钟脚
			QSPI0_CLK	I/O	MFP4	QSPI0 串行时钟脚
			SC1_RST	O	MFP5	智能卡 1 复位脚
			I²S0_DI	I	MFP6	I2S0 数据输入脚
			SPI1_MOSI	I/O	MFP7	SPI1 MOSI (主机输出，从机输入) 管脚
			UART2_nCTS	I	MFP8	UART2 禁止发送输入管脚
			I²C0_SMBSUS	O	MFP9	I²C0 SMBus SMBSUS 管脚 (PMBus 控制管脚)
			UART3_RXD	I	MFP11	UART3 数据接收输入脚
			EPWM1_CH3	I/O	MFP12	EPWM1 通道3输出/捕获输入脚
19	31	39	PC.1	I/O	MFP0	通用数字I/O脚

32 管脚	48 管脚	64 管脚	管脚名称	类型	MFP	描述
			EBI_AD1	I/O	MFP2	EBI地址/数据总线位1.
			SPIM_MISO	I/O	MFP3	SPIM MISO（主机输入，从机输出）脚.
			QSPI0_MISO0	I/O	MFP4	QSPI0 MISO0（主机输入，从机输出）脚.
			SC1_DAT	I/O	MFP5	智能卡 1 数据管脚.
			I^2S0_DO	O	MFP6	I^2S0 数据输出脚
			SPI1_CLK	I/O	MFP7	SPI1 串行时钟管脚
			UART2_TXD	O	MFP8	UART2 数据发送输出脚.
			I^2C0_SCL	I/O	MFP9	I2C0 时钟管脚
			EPWM1_CH4	I/O	MFP12	EPWM1 通道4输出/捕获输入脚
			ACMP0_O	O	MFP14	模拟比较器 0输出脚
20	32	40	PC.0	I/O	MFP0	通用数字I/O脚
			EBI_AD0	I/O	MFP2	EBI地址/数据总线位0.
			SPIM_MOSI	I/O	MFP3	SPIM MOSI (主机输出，从机输入) 管脚.
			QSPI0_MOSI0	I/O	MFP4	QSPI0 MOSI0 (主机输出，从机输入) 管脚.
			SC1_CLK	O	MFP5	智能卡 1 时钟脚
			I^2S0_LRCK	O	MFP6	I^2S0左右通道时钟输出脚.
			SPI1_SS	I/O	MFP7	SPI1 从机选择脚
			UART2_RXD	I	MFP8	UART2 数据接收输入脚
			I^2C0_SDA	I/O	MFP9	I2C0 数据输入/输出管脚
			EPWM1_CH5	I/O	MFP12	EPWM1 通道5输出/捕获输入
			ACMP1_O	O	MFP14	模拟比较器1输出脚
		41	PD.3	I/O	MFP0	通用数字I/O脚
			EBI_AD10	I/O	MFP2	EBI地址/数据总线位10.
			USCI0_CTL1	I/O	MFP3	USCI0 控制1 管脚.
			SPI0_SS	I/O	MFP4	SPI0 从机选择脚
			UART3_nRTS	O	MFP5	UART3 请求发送输出脚
			USCI1_CTL0	I/O	MFP6	USCI1 控制0管脚
			SC2_PWR	O	MFP7	智能卡 2电源脚
			SC1_nCD	I	MFP8	智能卡 1 卡检测管脚.
			UART0_TXD	O	MFP9	UART0 数据发送输出脚.

32 管脚	48 管脚	64 管脚	管脚名称	类型	MFP	描述
		42	PD.2	I/O	MFP0	通用数字I/O脚
			EBI_AD11	I/O	MFP2	EBI地址/数据总线位11.
			USCI0_DAT1	I/O	MFP3	USCI0 数据1 管脚.
			SPI0_CLK	I/O	MFP4	SPI0 串行时钟脚
			UART3_nCTS	I	MFP5	UART3 禁止发送输入管脚
			SC2_RST	O	MFP7	智能卡 2 复位脚
			UART0_RXD	I	MFP9	UART0 数据接收输入脚
		43	PD.1	I/O	MFP0	通用数字I/O脚
			EBI_AD12	I/O	MFP2	EBI地址/数据总线位12.
			USCI0_DAT0	I/O	MFP3	USCI0 数据0脚
			SPI0_MISO	I/O	MFP4	SPI0 MISO（主机输入，从机输出）脚.
			UART3_TXD	O	MFP5	UART3 数据发送输出脚.
			I^2C2_SCL	I/O	MFP6	I^2C2 时钟脚
			SC2_DAT	I/O	MFP7	智能卡 2 数据管脚.
		44	PD.0	I/O	MFP0	通用数字I/O脚
			EBI_AD13	I/O	MFP2	EBI地址/数据总线位13.
			USCI0_CLK	I/O	MFP3	USCI0 时钟脚
			SPI0_MOSI	I/O	MFP4	SPI0 MOSI (主机输出，从机输入) 管脚.
			UART3_RXD	I	MFP5	UART3 数据接收输入脚
			I^2C2_SDA	I/O	MFP6	I^2C2 数据输入/输出脚
			SC2_CLK	O	MFP7	智能卡 2 时钟脚
			TM2	I/O	MFP14	Timer2 事件计数输入/反转输出脚
21	33	45	PA.12	I/O	MFP0	通用数字I/O脚
			I^2S0_BCLK	O	MFP2	I2S0位时钟输出管脚
			UART4_TXD	O	MFP3	UART4 数据发送输出脚.
			I^2C1_SCL	I/O	MFP4	I^2C1 时钟脚
			SPI2_SS	I/O	MFP5	SPI2 从机选择脚
			SC2_PWR	O	MFP7	智能卡 2电源脚
			BPWM1_CH2	I/O	MFP11	BPWM1 通道2 输出/捕获输入
			QEI1_INDEX	I	MFP12	正交编码器1 索引输入

32 管脚	48 管脚	64 管脚	管脚名称	类型	MFP	描述
22	34	46	PA.13	I/O	MFP0	通用数字I/O脚
			I^2S0_MCLK	O	MFP2	I2S0 主时钟输出脚
			UART4_RXD	I	MFP3	UART4 数据接收输入脚
			I^2C1_SDA	I/O	MFP4	I^2C1 数据输入/输出脚
			SPI2_CLK	I/O	MFP5	SPI2 串行时钟脚
			SC2_RST	O	MFP7	智能卡 2 复位脚
			BPWM1_CH3	I/O	MFP11	BPWM1 通道3输出/捕获输入脚
			QEI1_A	I	MFP12	正交编码器1 相位A输入
23	35	47	PA.14	I/O	MFP0	通用数字I/O脚
			I^2S0_DI	I	MFP2	I2S0 数据输入脚
			UART0_TXD	O	MFP3	UART0 数据发送输出脚.
			SPI2_MISO	I/O	MFP5	SPI2 MISO（主机输入，从机输出）脚.
			I^2C2_SCL	I/O	MFP6	I^2C2 时钟脚
			SC2_DAT	I/O	MFP7	智能卡 2 数据管脚.
			BPWM1_CH4	I/O	MFP11	BPWM1 通道4输出/捕获输入脚
			QEI1_B	I	MFP12	正交编码器1 相位B 输入
24	36	48	PA.15	I/O	MFP0	通用数字I/O脚
			I^2S0_DO	O	MFP2	I^2S0 数据输出脚
			UART0_RXD	I	MFP3	UART0 数据接收输入脚
			SPI2_MOSI	I/O	MFP5	SPI2 MOSI (主机输出，从机输入) 管脚.
			I^2C2_SDA	I/O	MFP6	I^2C2 数据输入/输出脚
			SC2_CLK	O	MFP7	智能卡 2 时钟脚
			BPWM1_CH5	I/O	MFP11	EPWM1 通道5输出/捕获输入
			EPWM0_SYNC_IN	I	MFP12	EPWM0 计数器同步触发输入脚
25	37	49	V_{SS}	P	MFP0	数字电路地管脚
26	38	50	LDO_CAP	A	MFP0	LDO输出脚
27	39	51	V_{DD}	P	MFP0	I/O和内部PLL及数字电路的LDO电源
	40	52	PC.14	I/O	MFP0	通用数字I/O脚
			EBI_AD11	I/O	MFP2	EBI地址/数据总线位11.
			SC1_nCD	I	MFP3	智能卡 1 卡检测管脚.

32 管脚	48 管脚	64 管脚	管脚名称	类型	MFP	描述
			SPI0_I2SMCLK	I/O	MFP4	SPI0 I²S 主机时钟输出脚
			USCI0_CTL0	I/O	MFP5	USCI0 控制0 管脚.
			QSPI0_CLK	I/O	MFP6	QSPI0 串行时钟脚
			EPWM0_SYNC_IN	I	MFP11	EPWM0 计数器同步触发输入脚
			TM1	I/O	MFP13	Timer1 事件计数输入/反转输出脚
28	41	53	PB.15	I/O	MFP0	通用数字I/O脚
			EADC0_CH15	A	MFP1	EADC0 通道15 模拟输入
			EBI_AD12	I/O	MFP2	EBI地址/数据总线位12.
			SC1_PWR	O	MFP3	智能卡 1电源脚
			SPI0_SS	I/O	MFP4	SPI0 从机选择脚
			USCI0_CTL1	I/O	MFP5	USCI0 控制1 管脚.
			UART0_nCTS	I	MFP6	UART0 禁止发送输入管脚
			UART3_TXD	O	MFP7	UART3 数据发送输出脚.
			I²C2_SMBAL	O	MFP8	I²C2 SMBus SMBALTER 管脚
			EPWM1_CH0	I/O	MFP11	EPWM1 通道0 输出/捕获输入
			TM0_EXT	I/O	MFP13	Timer0 事件计数输入/反转输出管脚
29	42	54	PB.14	I/O	MFP0	通用数字I/O脚
			EADC0_CH14	A	MFP1	EADC0 通道14 模拟输入
			EBI_AD13	I/O	MFP2	EBI地址/数据总线位13.
			SC1_RST	O	MFP3	智能卡 1 复位脚
			SPI0_CLK	I/O	MFP4	SPI0 串行时钟脚
			USCI0_DAT1	I/O	MFP5	USCI0 数据1 管脚.
			UART0_nRTS	O	MFP6	UART0 请求发送输出脚
			UART3_RXD	I	MFP7	UART3 数据接收输入脚
			I²C2_SMBSUS	O	MFP8	I²C2 SMBus SMBSUS 管脚 (PMBus 控制管脚)
			EPWM1_CH1	I/O	MFP11	EPWM1 通道1 输出/捕获输入
			TM1_EXT	I/O	MFP13	Timer1 事件计数输入/反转输出脚
			CLKO	O	MFP14	时钟输出
30	43	55	PB.13	I/O	MFP0	通用数字I/O脚
			EADC0_CH13	A	MFP1	EADC0 通道13 模拟输入

32 管脚	48 管脚	64 管脚	管脚名称	类型	MFP	描述
			DAC1_OUT	A	MFP1	DAC1 模拟输出通道
			ACMP0_P3	A	MFP1	模拟比较器 0 正极输入 3 管脚.
			ACMP1_P3	A	MFP1	模拟比较器 1 正极输入 3 管脚.
			EBI_AD14	I/O	MFP2	EBI地址/数据总线位14.
			SC1_DAT	I/O	MFP3	智能卡 1 数据管脚.
			SPI0_MISO	I/O	MFP4	SPI0 MISO（主机输入，从机输出）脚.
			USCI0_DAT0	I/O	MFP5	USCI0 数据0脚
			UART0_TXD	O	MFP6	UART0 数据发送输出脚.
			UART3_nRTS	O	MFP7	UART3 请求发送输出脚
			I^2C2_SCL	I/O	MFP8	I^2C2 时钟脚
			EPWM1_CH2	I/O	MFP11	EPWM1 通道2 输出/捕获输入
			TM2_EXT	I/O	MFP13	Timer2 事件计数输入/反转输出脚
31	44	56	PB.12	I/O	MFP0	通用数字I/O脚
			EADC0_CH12	A	MFP1	EADC0 通道12 模拟输入
			DAC0_OUT	A	MFP1	DAC0 模拟输出通道
			ACMP0_P2	A	MFP1	模拟比较器 0 正极输入 2 管脚.
			ACMP1_P2	A	MFP1	模拟比较器 1 正极输入 2 管脚.
			EBI_AD15	I/O	MFP2	EBI地址/数据总线位15.
			SC1_CLK	O	MFP3	智能卡 1 时钟脚
			SPI0_MOSI	I/O	MFP4	SPI0 MOSI (主机输出，从机输入) 管脚.
			USCI0_CLK	I/O	MFP5	USCI0 时钟脚
			UART0_RXD	I	MFP6	UART0 数据接收输入脚
			UART3_nCTS	I	MFP7	UART3 禁止发送输入管脚
			I^2C2_SDA	I/O	MFP8	I^2C2 数据输入/输出脚
			SD0_nCD	I	MFP9	SD/SDIO0 卡检测输入管脚
			EPWM1_CH3	I/O	MFP11	EPWM1 通道3输出/捕获输入脚
			TM3_EXT	I/O	MFP13	Timer3 事件计数输入/反转输出脚
32	45	57	AV_{DD}	P	MFP0	内部模拟电路电源
		58	V_{REF}	A	MFP0	ADC参考电压输入. 注: 此管脚需接1uF电容到地.
	46	59	AV_{SS}	P	MFP0	模拟电路地管脚

32 管脚	48 管脚	64 管脚	管脚名称	类型	MFP	描述
		60	PB.11	I/O	MFP0	通用数字I/O脚
			EADC0_CH11	A	MFP1	EADC0 通道11 模拟输入
			EBI_ADR16	O	MFP2	EBI 地址总线位 16.
			UART0_nCTS	I	MFP5	UART0 禁止发送输入管脚
			UART4_TXD	O	MFP6	UART4 数据发送输出脚.
			I^2C1_SCL	I/O	MFP7	I^2C1 时钟脚
			SPI0_I2SMCLK	I/O	MFP9	SPI0 I^2S 主机时钟输出脚
			BPWM1_CH0	I/O	MFP10	BPWM1 通道0 输出/捕获输入
			SPI3_CLK	I/O	MFP11	SPI3 串行时钟脚
		61	PB.10	I/O	MFP0	通用数字I/O脚
			EADC0_CH10	A	MFP1	EADC0 通道10 模拟输入
			EBI_ADR17	O	MFP2	EBI 地址总线位 17.
			USCI1_CTL0	I/O	MFP4	USCI1 控制0管脚
			UART0_nRTS	O	MFP5	UART0 请求发送输出脚
			UART4_RXD	I	MFP6	UART4 数据接收输入脚
			I^2C1_SDA	I/O	MFP7	I^2C1 数据输入/输出脚
			BPWM1_CH1	I/O	MFP10	BPWM1 通道1 输出/捕获输入
			SPI3_SS	I/O	MFP11	SPI3 从机选择脚
		62	PB.9	I/O	MFP0	通用数字I/O脚
			EADC0_CH9	A	MFP1	EADC0 通道9 模拟输入
			EBI_ADR18	O	MFP2	EBI 地址总线位 18.
			USCI1_CTL1	I/O	MFP4	USCI1 控制1管脚
			UART0_TXD	O	MFP5	UART0 数据发送输出脚.
			UART1_nCTS	I	MFP6	UART1 禁止发送输入管脚
			I^2C1_SMBAL	O	MFP7	I^2C1 SMBus SMBALTER 管脚
			BPWM1_CH2	I/O	MFP10	BPWM1 通道2 输出/捕获输入
			SPI3_MISO	I/O	MFP11	SPI3 MISO（主机输入，从机输出）脚.
			INT7	I	MFP13	外部中断7 输入脚
		63	PB.8	I/O	MFP0	通用数字I/O脚
			EADC0_CH8	A	MFP1	EADC0 通道8 模拟输入

32 管脚	48 管脚	64 管脚	管脚名称	类型	MFP	描述
			EBI_ADR19	O	MFP2	EBI 地址总线位 19.
			USCI1_CLK	I/O	MFP4	USCI1 时钟脚
			UART0_RXD	I	MFP5	UART0 数据接收输入脚
			UART1_nRTS	O	MFP6	UART1 请求发送输出脚
			I^2C1_SMBSUS	O	MFP7	I^2C1 SMBus SMBSUS 管脚 (PMBus 控制管脚)
			BPWM1_CH3	I/O	MFP10	BPWM1 通道3输出/捕获输入脚
			SPI3_MOSI	I/O	MFP11	SPI3 MOSI (主机输出，从机输入) 管脚.
			INT6	I	MFP13	外部中断6 输入脚
	47	64	PB.7	I/O	MFP0	通用数字I/O脚
			EADC0_CH7	A	MFP1	EADC0 通道7 模拟输入
			EBI_nWRL	O	MFP2	EBI低字节写使能输出脚
			USCI1_DAT0	I/O	MFP4	USCI1 数据0脚
			UART1_TXD	O	MFP6	UART1 数据发送输出脚
			SD1_CMD	I/O	MFP7	SD/SDIO1 命令/响应脚
			EBI_nCS0	O	MFP8	EBI片选0输出脚
			BPWM1_CH4	I/O	MFP10	BPWM1 通道4输出/捕获输入脚
			EPWM1_BRAKE0	I	MFP11	EPWM1 刹车0 输入脚
			EPWM1_CH4	I/O	MFP12	EPWM1 通道4输出/捕获输入脚
			INT5	I	MFP13	外部中断5 输入脚
			ACMP0_O	O	MFP15	模拟比较器 0输出脚

4.2.2 M482系列管脚描述

48 管脚	64 管脚	管脚名称	类型	MFP	描述
48	1	PB.6	I/O	MFP0	通用数字I/O脚
		EADC0_CH6	A	MFP1	EADC0通道6模拟输入
		EBI_nWRH	O	MFP2	EBI 高字节写使能输出管脚
		USCI1_DAT1	I/O	MFP4	USCI1数据1管脚.
		UART1_RXD	I	MFP6	UART1数据接收输入管脚
		SD1_CLK	O	MFP7	SD/SDIO1 时钟输出管脚
		EBI_nCS1	O	MFP8	EBI片选1输出管脚
		BPWM1_CH5	I/O	MFP10	EPWM1 通道5输出/捕获输入
		EPWM1_BRAKE1	I	MFP11	EPWM1 刹车1输入管脚
		EPWM1_CH5	I/O	MFP12	EPWM1 通道5输出/捕获输入
		INT4	I	MFP13	外部中断4输入管脚
		USB_VBUS_EN	O	MFP14	USB 外部 VBUS 电源使能管脚.
		ACMP1_O	O	MFP15	模拟比较器1输出脚
1	2	PB.5	I/O	MFP0	通用数字I/O脚
		EADC0_CH5	A	MFP1	EADC0 通道5模拟输入
		ACMP1_N	A	MFP1	模拟比较器1负极输入管脚
		EBI_ADR0	O	MFP2	EBI地址总线位0
		SD0_DAT3	I/O	MFP3	SD/SDIO0数据线位3
		SPI1_MISO	I/O	MFP5	SPI1 MISO(主机输入，从机输出)管脚
		I^2C0_SCL	I/O	MFP6	I2C0 时钟管脚
		UART5_TXD	O	MFP7	UART5数据发送输出脚
		USCI1_CTL0	I/O	MFP8	USCI1 控制0管脚
		SC0_CLK	O	MFP9	智能卡0时钟管脚
		I^2S0_BCLK	O	MFP10	I2S0位时钟输出管脚
		EPWM0_CH0	I/O	MFP11	EPWM0 通道0输出/捕获输入
		TM0	I/O	MFP14	Timer0 事件计数输入/反转输出管脚
		INT0	I	MFP15	外部中断0输入管脚
2	3	PB.4	I/O	MFP0	通用数字I/O脚
		EADC0_CH4	A	MFP1	EADC0 通道4模拟输入

		ACMP1_P1	A	MFP1	模拟比较器1正极输入1脚
		EBI_ADR1	O	MFP2	EBI 地址总线位1
		SD0_DAT2	I/O	MFP3	SD/SDIO0 数据线位2
		SPI1_MOSI	I/O	MFP5	SPI1 MOSI (主机输出，从机输入) 管脚
		I^2C0_SDA	I/O	MFP6	I2C0 数据输入/输出管脚
		UART5_RXD	I	MFP7	UART5 数据接收输入管脚
		USCI1_CTL1	I/O	MFP8	USCI1 控制1管脚
		SC0_DAT	I/O	MFP9	智能卡0数据输入脚
		I^2S0_MCLK	O	MFP10	I2S0 主时钟输出脚
		EPWM0_CH1	I/O	MFP11	EPWM0 通道1输出/捕获输入脚
		TM1	I/O	MFP14	Timer1 事件计数输入/反转输出脚
		INT1	I	MFP15	外部中断1输入脚
3	4	PB.3	I/O	MFP0	通用数字I/O脚
		EADC0_CH3	A	MFP1	EADC0 通道3模拟输入
		ACMP0_N	A	MFP1	模拟比较器0负极输入脚
		EBI_ADR2	O	MFP2	EBI 地址总线位2
		SD0_DAT1	I/O	MFP3	SD/SDIO0 数据线位1
		SPI1_CLK	I/O	MFP5	SPI1 串行时钟管脚
		UART1_TXD	O	MFP6	UART1 数据发送输出脚
		UART5_nRTS	O	MFP7	UART5 请求发送输出脚
		USCI1_DAT1	I/O	MFP8	USCI1数据1管脚.
		SC0_RST	O	MFP9	智能卡0复位脚
		I^2S0_DI	I	MFP10	I2S0 数据输入脚
		EPWM0_CH2	I/O	MFP11	EPWM0 通道2输出/捕获输入脚
		TM2	I/O	MFP14	Timer2 事件计数输入/反转输出脚
		INT2	I	MFP15	外部中断2输入脚
4	5	PB.2	I/O	MFP0	通用数字I/O脚
		EADC0_CH2	A	MFP1	EADC0 通道2 模拟输入
		ACMP0_P1	A	MFP1	模拟比较器0正极输入1脚
		OPA0_O	A	MFP1	运放0输出脚
		EBI_ADR3	O	MFP2	EBI 地址总线位3
		SD0_DAT0	I/O	MFP3	SD/SDIO0 数据线位0

		SPI1_SS	I/O	MFP5	SPI1 从机选择脚
		UART1_RXD	I	MFP6	UART1数据接收输入管脚
		UART5_nCTS	I	MFP7	UART5 禁止发送输入管脚
		USCI1_DAT0	I/O	MFP8	USCI1 数据0脚
		SC0_PWR	O	MFP9	智能卡0电源脚
		I^2S0_DO	O	MFP10	I^2S0 数据输出脚
		EPWM0_CH3	I/O	MFP11	EPWM0 通道3输出/捕获输入脚
		TM3	I/O	MFP14	Timer3 事件计数输入/反转输出脚
		INT3	I	MFP15	外部中断3输入脚
5	6	PB.1	I/O	MFP0	通用数字I/O脚
		EADC0_CH1	A	MFP1	EADC0 通道1 模拟输入
		OPA0_N	A	MFP1	运放0负极输入脚
		EBI_ADR8	O	MFP2	EBI 地址总线位 8.
		SD0_CLK	O	MFP3	SD/SDIO0 时钟输出脚
		SPI1_I2SMCLK	I/O	MFP5	SPI1 I^2S 主机时钟输出脚
		SPI3_I2SMCLK	I/O	MFP6	SPI3 I^2S 主机时钟输出脚
		UART2_TXD	O	MFP7	UART2 数据发送输出脚.
		USCI1_CLK	I/O	MFP8	USCI1 时钟脚
		I^2C1_SCL	I/O	MFP9	I^2C1 时钟脚
		I^2S0_LRCK	O	MFP10	I^2S0左右通道时钟输出脚.
		EPWM0_CH4	I/O	MFP11	EPWM0 通道4输出/捕获输入脚
		EPWM1_CH4	I/O	MFP12	EPWM1 通道4输出/捕获输入脚
		EPWM0_BRAKE0	I	MFP13	EPWM0 刹车0 输入脚
6	7	PB.0	I/O	MFP0	通用数字I/O脚
		EADC0_CH0	A	MFP1	EADC0 通道0 模拟输入
		OPA0_P	A	MFP1	运放 0 正极输入脚
		EBI_ADR9	O	MFP2	EBI 地址总线位 9.
		SD0_CMD	I/O	MFP3	SD/SDIO0 命令/响应脚
		UART2_RXD	I	MFP7	UART2 数据接收输入脚
		SPI0_I2SMCLK	I/O	MFP8	SPI0 I^2S 主机时钟输出脚
		I^2C1_SDA	I/O	MFP9	I^2C1 数据输入/输出脚
		EPWM0_CH5	I/O	MFP11	EPWM0 通道5 输出/捕获输入

		EPWM1_CH5	I/O	MFP12	EPWM1 通道5输出/捕获输入
		EPWM0_BRAKE1	I	MFP13	EPWM0 刹车1 输入脚
7	8	PA.11	I/O	MFP0	通用数字I/O脚
		ACMP0_P0	A	MFP1	模拟比较器 0 正极输入脚0
		EBI_nRD	O	MFP2	EBI 读使能输出脚
		SC2_PWR	O	MFP3	智能卡 2电源脚
		SPI2_SS	I/O	MFP4	SPI2 从机选择脚
		SD1_DAT3	I/O	MFP5	SD/SDIO1 数据线位3.
		USCI0_CLK	I/O	MFP6	USCI0 时钟脚
		I^2C2_SCL	I/O	MFP7	I^2C2 时钟脚
		BPWM0_CH0	I/O	MFP9	BPWM0 通道0 输出/捕获输入
		EPWM0_SYNC_OUT	O	MFP10	EPWM0计数器同步触发输出脚
		TM0_EXT	I/O	MFP13	Timer0 事件计数输入/反转输出管脚
		DAC1_ST	I	MFP14	DAC1 外部触发输入
8	9	PA.10	I/O	MFP0	通用数字I/O脚
		ACMP1_P0	A	MFP1	模拟比较器 1 正极输入脚0
		OPA1_O	A	MFP1	运放 1输出脚
		EBI_nWR	O	MFP2	EBI 写使能输出脚
		SC2_RST	O	MFP3	智能卡 2 复位脚
		SPI2_CLK	I/O	MFP4	SPI2 串行时钟脚
		SD1_DAT2	I/O	MFP5	SD/SDIO1 数据线位2.
		USCI0_DAT0	I/O	MFP6	USCI0 数据0脚
		I^2C2_SDA	I/O	MFP7	I^2C2 数据输入/输出脚
		BPWM0_CH1	I/O	MFP9	BPWM0 通道1 输出/捕获输入
		QEI1_INDEX	I	MFP10	正交编码器1 索引输入
		ECAP0_IC0	I	MFP11	增强型捕获计数器 0 输入脚0
		TM1_EXT	I/O	MFP13	Timer1 事件计数输入/反转输出脚
		DAC0_ST	I	MFP14	DAC0 外部触发输入
		SWDH_CLK	O	MFP15	串行调试总线主时钟输出
9	10	PA.9	I/O	MFP0	通用数字I/O脚
		OPA1_N	A	MFP1	运放 1 负极输入脚
		EBI_MCLK	O	MFP2	EBI 外部时钟输出脚.

		SC2_DAT	I/O	MFP3	智能卡 2 数据管脚.
		SPI2_MISO	I/O	MFP4	SPI2 MISO（主机输入，从机输出）脚.
		SD1_DAT1	I/O	MFP5	SD/SDIO1 数据线位1.
		USCI0_DAT1	I/O	MFP6	USCI0 数据1 管脚.
		UART1_TXD	O	MFP7	UART1 数据发送输出脚
		BPWM0_CH2	I/O	MFP9	BPWM0 通道2 输出/捕获输入
		QEI1_A	I	MFP10	正交编码器1 相位A输入
		ECAP0_IC1	I	MFP11	增强型捕获计数器 0 输入 1 管脚.
		TM2_EXT	I/O	MFP13	Timer2 事件计数输入/反转输出脚
		SWDH_DAT	I/O	MFP15	串行调试总线主机数据输入/输出管脚
10	11	PA.8	I/O	MFP0	通用数字I/O脚
		OPA1_P	A	MFP1	运放 1 正极输入脚
		EBI_ALE	O	MFP2	EBI地址锁存使能输出脚
		SC2_CLK	O	MFP3	智能卡 2 时钟脚
		SPI2_MOSI	I/O	MFP4	SPI2 MOSI (主机输出，从机输入) 管脚.
		SD1_DAT0	I/O	MFP5	SD/SDIO1 数据线位0.
		USCI0_CTL1	I/O	MFP6	USCI0 控制1 管脚.
		UART1_RXD	I	MFP7	UART1数据接收输入管脚
		BPWM0_CH3	I/O	MFP9	BPWM0 通道3输出/捕获输入脚
		QEI1_B	I	MFP10	正交编码器1 相位B 输入
		ECAP0_IC2	I	MFP11	增强型捕获计数器 0 输入 2 管脚.
		TM3_EXT	I/O	MFP13	Timer3 事件计数输入/反转输出脚
		INT4	I	MFP15	外部中断4输入管脚
	12	PF.6	I/O	MFP0	通用数字I/O脚
		EBI_ADR19	O	MFP2	EBI 地址总线位 19.
		SC0_CLK	O	MFP3	智能卡0时钟管脚
		I^2S0_LRCK	O	MFP4	I^2S0左右通道时钟输出脚.
		SPI0_MOSI	I/O	MFP5	SPI0 MOSI (主机输出，从机输入) 管脚.
		UART4_RXD	I	MFP6	UART4 数据接收输入脚
		EBI_nCS0	O	MFP7	EBI片选0输出脚
		TAMPER0	I/O	MFP10	TAMPER循环检测管脚 0.
	13	V_{DD}	P	MFP0	I/O和内部PLL及数字电路的LDO电源

11	14	PF.5	I/O	MFP0	通用数字I/O脚
		UART2_RXD	I	MFP2	UART2 数据接收输入脚
		UART2_nCTS	I	MFP4	UART2 禁止发送输入管脚
		BPWM0_CH4	I/O	MFP8	BPWM0 通道4输出/捕获输入脚
		EPWM0_SYNC_OUT	O	MFP9	EPWM0计数器同步触发输出脚
		X32_IN	I	MFP10	外部 32.768 kHz 晶振输入脚
		EADC0_ST	I	MFP11	EADC0 外部触发输入
12	15	PF.4	I/O	MFP0	通用数字I/O脚
		UART2_TXD	O	MFP2	UART2 数据发送输出脚.
		UART2_nRTS	O	MFP4	UART2 请求发送输出脚
		BPWM0_CH5	I/O	MFP8	BPWM0 通道5 输出/捕获输入
		X32_OUT	O	MFP10	外部 32.768 kHz 晶振输出脚
13	16	PF.3	I/O	MFP0	通用数字I/O脚
		EBI_nCS0	O	MFP2	EBI片选0输出脚
		UART0_TXD	O	MFP3	UART0 数据发送输出脚.
		I^2C0_SCL	I/O	MFP4	I2C0 时钟管脚
		XT1_IN	I	MFP10	外部 4~24 MHz (高速) 晶振输入脚
		BPWM1_CH0	I/O	MFP11	BPWM1 通道0 输出/捕获输入
14	17	PF.2	I/O	MFP0	通用数字I/O脚
		EBI_nCS1	O	MFP2	EBI片选1输出管脚
		UART0_RXD	I	MFP3	UART0 数据接收输入脚
		I^2C0_SDA	I/O	MFP4	I2C0 数据输入/输出管脚
		QSPI0_CLK	I/O	MFP5	QSPI0 串行时钟脚
		XT1_OUT	O	MFP10	外部 4~24 MHz (高速) 晶振输出脚
		BPWM1_CH1	I/O	MFP11	BPWM1 通道1 输出/捕获输入
	18	PC.7	I/O	MFP0	通用数字I/O脚
		EBI_AD9	I/O	MFP2	EBI地址/数据总线位9.
		SPI1_MISO	I/O	MFP4	SPI1 MISO(主机输入，从机输出)管脚
		UART4_TXD	O	MFP5	UART4 数据发送输出脚.
		SC2_PWR	O	MFP6	智能卡 2电源脚
		UART0_nCTS	I	MFP7	UART0 禁止发送输入管脚
		I^2C1_SMBAL	O	MFP8	I^2C1 SMBus SMBALTER 管脚

		EPWM1_CH2	I/O	MFP11	EPWM1 通道2 输出/捕获输入
		BPWM1_CH0	I/O	MFP12	BPWM1 通道0 输出/捕获输入
		TM0	I/O	MFP14	Timer0 事件计数输入/反转输出管脚
		INT3	I	MFP15	外部中断3输入脚
	19	PC.6	I/O	MFP0	通用数字I/O脚
		EBI_AD8	I/O	MFP2	EBI地址/数据总线位8.
		SPI1_MOSI	I/O	MFP4	SPI1 MOSI (主机输出，从机输入) 管脚
		UART4_RXD	I	MFP5	UART4 数据接收输入脚
		SC2_RST	O	MFP6	智能卡 2 复位脚
		UART0_nRTS	O	MFP7	UART0 请求发送输出脚
		I^2C1_SMBSUS	O	MFP8	I^2C1 SMBus SMBSUS 管脚 (PMBus 控制管脚)
		EPWM1_CH3	I/O	MFP11	EPWM1 通道3输出/捕获输入脚
		BPWM1_CH1	I/O	MFP12	BPWM1 通道1 输出/捕获输入
		TM1	I/O	MFP14	Timer1 事件计数输入/反转输出脚
		INT2	I	MFP15	外部中断2输入脚
15	20	PA.7	I/O	MFP0	通用数字I/O脚
		EBI_AD7	I/O	MFP2	EBI地址/数据总线位7.
		SPI1_CLK	I/O	MFP4	SPI1 串行时钟管脚
		SC2_DAT	I/O	MFP6	智能卡 2 数据管脚.
		UART0_TXD	O	MFP7	UART0 数据发送输出脚.
		I^2C1_SCL	I/O	MFP8	I^2C1 时钟脚
		EPWM1_CH4	I/O	MFP11	EPWM1 通道4输出/捕获输入脚
		BPWM1_CH2	I/O	MFP12	BPWM1 通道2 输出/捕获输入
		ACMP0_WLAT	I	MFP13	模拟比较器 0窗口锁存输入管脚
		TM2	I/O	MFP14	Timer2 事件计数输入/反转输出脚
		INT1	I	MFP15	外部中断1输入脚
16	21	PA.6	I/O	MFP0	通用数字I/O脚
		EBI_AD6	I/O	MFP2	EBI地址/数据总线位6.
		SPI1_SS	I/O	MFP4	SPI1 从机选择脚
		SD1_nCD	I	MFP5	SD/SDIO1 卡检测输入管脚
		SC2_CLK	O	MFP6	智能卡 2 时钟脚
		UART0_RXD	I	MFP7	UART0 数据接收输入脚

		I²C1_SDA	I/O	MFP8	I²C1 数据输入/输出脚
		EPWM1_CH5	I/O	MFP11	EPWM1 通道5输出/捕获输入
		BPWM1_CH3	I/O	MFP12	BPWM1 通道3输出/捕获输入脚
		ACMP1_WLAT	I	MFP13	模拟比较器 1窗口锁存输入管脚
		TM3	I/O	MFP14	Timer3 事件计数输入/反转输出脚
		INT0	I	MFP15	外部中断0输入管脚
	22	V_{SS}	P	MFP0	数字电路地管脚
	23	V_{DD}	P	MFP0	I/O和内部PLL及数字电路的LDO电源
	24	LDO_CAP	A	MFP0	LDO输出脚
17	25	PA.5	I/O	MFP0	通用数字I/O脚
		SPIM_D2	I/O	MFP2	SPIM QuadI/O模式下数据2管脚
		QSPI0_MISO1	I/O	MFP3	QSPI0 MISO1（主机输入，从机输出）脚.
		SPI1_I2SMCLK	I/O	MFP4	SPI1 I²S 主机时钟输出脚
		SD1_CMD	I/O	MFP5	SD/SDIO1 命令/响应脚
		SC2_nCD	I	MFP6	智能卡 2 卡检测管脚.
		UART0_nCTS	I	MFP7	UART0 禁止发送输入管脚
		UART5_TXD	O	MFP8	UART5数据发送输出脚
		I²C0_SCL	I/O	MFP9	I2C0 时钟管脚
		BPWM0_CH5	I/O	MFP12	BPWM0 通道5 输出/捕获输入
		EPWM0_CH0	I/O	MFP13	EPWM0 通道0输出/捕获输入
		QEI0_INDEX	I	MFP14	正交编码器0 索引输入
18	26	PA.4	I/O	MFP0	通用数字I/O脚
		SPIM_D3	I/O	MFP2	SPIM QuadI/O模式下数据3管脚
		QSPI0_MOSI1	I/O	MFP3	QSPI0 MOSI1 (主机输出，从机输入) 管脚.
		SPI0_I2SMCLK	I/O	MFP4	SPI0 I²S 主机时钟输出脚
		SD1_CLK	O	MFP5	SD/SDIO1 时钟输出管脚
		SC0_nCD	I	MFP6	智能卡 0 卡检测管脚.
		UART0_nRTS	O	MFP7	UART0 请求发送输出脚
		UART5_RXD	I	MFP8	UART5 数据接收输入管脚
		I²C0_SDA	I/O	MFP9	I2C0 数据输入/输出管脚
		BPWM0_CH4	I/O	MFP12	BPWM0 通道4输出/捕获输入脚
		EPWM0_CH1	I/O	MFP13	EPWM0 通道1输出/捕获输入脚

		QEI0_A	I	MFP14	正交编码器0 相位A输入
19	27	PA.3	I/O	MFP0	通用数字I/O脚
		SPIM_SS	I/O	MFP2	SPIM 从机选择脚
		QSPI0_SS	I/O	MFP3	QSPI0 从机选择脚
		SPI0_SS	I/O	MFP4	SPI0 从机选择脚
		SD1_DAT3	I/O	MFP5	SD/SDIO1 数据线位3.
		SC0_PWR	O	MFP6	智能卡0电源脚
		UART4_TXD	O	MFP7	UART4 数据发送输出脚.
		UART1_TXD	O	MFP8	UART1 数据发送输出脚
		I^2C1_SCL	I/O	MFP9	I^2C1 时钟脚
		BPWM0_CH3	I/O	MFP12	BPWM0 通道3输出/捕获输入脚
		EPWM0_CH2	I/O	MFP13	EPWM0 通道2输出/捕获输入脚
		QEI0_B	I	MFP14	正交编码器0 相位B 输入
20	28	PA.2	I/O	MFP0	通用数字I/O脚
		SPIM_CLK	I/O	MFP2	SPIM 串行时钟脚
		QSPI0_CLK	I/O	MFP3	QSPI0 串行时钟脚
		SPI0_CLK	I/O	MFP4	SPI0 串行时钟脚
		SD1_DAT2	I/O	MFP5	SD/SDIO1 数据线位2.
		SC0_RST	O	MFP6	智能卡0复位脚
		UART4_RXD	I	MFP7	UART4 数据接收输入脚
		UART1_RXD	I	MFP8	UART1数据接收输入管脚
		I^2C1_SDA	I/O	MFP9	I^2C1 数据输入/输出脚
		BPWM0_CH2	I/O	MFP12	BPWM0 通道2 输出/捕获输入
		EPWM0_CH3	I/O	MFP13	EPWM0 通道3输出/捕获输入脚
21	29	PA.1	I/O	MFP0	通用数字I/O脚
		SPIM_MISO	I/O	MFP2	SPIM MISO（主机输入，从机输出）脚.
		QSPI0_MISO0	I/O	MFP3	QSPI0 MISO0（主机输入，从机输出）脚.
		SPI0_MISO	I/O	MFP4	SPI0 MISO（主机输入，从机输出）脚.
		SD1_DAT1	I/O	MFP5	SD/SDIO1 数据线位1.
		SC0_DAT	I/O	MFP6	智能卡0数据输入脚
		UART0_TXD	O	MFP7	UART0 数据发送输出脚.
		UART1_nCTS	I	MFP8	UART1 禁止发送输入管脚

		I²C2_SCL	I/O	MFP9	I²C2 时钟脚
		BPWM0_CH1	I/O	MFP12	BPWM0 通道1 输出/捕获输入
		EPWM0_CH4	I/O	MFP13	EPWM0 通道4输出/捕获输入脚
		DAC1_ST	I	MFP15	DAC1 外部触发输入
22	30	PA.0	I/O	MFP0	通用数字I/O脚
		SPIM_MOSI	I/O	MFP2	SPIM MOSI (主机输出，从机输入) 管脚.
		QSPI0_MOSI0	I/O	MFP3	QSPI0 MOSI0 (主机输出，从机输入) 管脚.
		SPI0_MOSI	I/O	MFP4	SPI0 MOSI (主机输出，从机输入) 管脚.
		SD1_DAT0	I/O	MFP5	SD/SDIO1 数据线位0.
		SC0_CLK	O	MFP6	智能卡0时钟管脚
		UART0_RXD	I	MFP7	UART0 数据接收输入脚
		UART1_nRTS	O	MFP8	UART1 请求发送输出脚
		I²C2_SDA	I/O	MFP9	I²C2 数据输入/输出脚
		BPWM0_CH0	I/O	MFP12	BPWM0 通道0 输出/捕获输入
		EPWM0_CH5	I/O	MFP13	EPWM0 通道5 输出/捕获输入
		DAC0_ST	I	MFP15	DAC0 外部触发输入
23	31	V_{DDIO}	P	MFP0	PE.1, PE.8~PE.13 的电源
24	32	nRESET	I	MFP0	外部复位输入:低电平有效,有内部上拉，设置此管脚为低将复位初始状态
25	33	PF.0	I/O	MFP0	通用数字I/O脚
		UART1_TXD	O	MFP2	UART1 数据发送输出脚
		I²C1_SCL	I/O	MFP3	I²C1 时钟脚
		BPWM1_CH0	I/O	MFP12	BPWM1 通道0 输出/捕获输入
		ICE_DAT	O	MFP14	串行调试总线数据管脚.
26	34	PF.1	I/O	MFP0	通用数字I/O脚
		UART1_RXD	I	MFP2	UART1数据接收输入管脚
		I²C1_SDA	I/O	MFP3	I²C1 数据输入/输出脚
		BPWM1_CH1	I/O	MFP12	BPWM1 通道1 输出/捕获输入
		ICE_CLK	I	MFP14	串行调试总线时钟脚
27	35	PC.5	I/O	MFP0	通用数字I/O脚
		EBI_AD5	I/O	MFP2	EBI地址/数据总线位5.
		SPIM_D2	I/O	MFP3	SPIM QuadI/O模式下数据2管脚
		QSPI0_MISO1	I/O	MFP4	QSPI0 MISO1（主机输入，从机输出）脚.

		UART2_TXD	O	MFP8	UART2 数据发送输出脚.
		I²C1_SCL	I/O	MFP9	I²C1 时钟脚
		UART4_TXD	O	MFP11	UART4 数据发送输出脚.
		EPWM1_CH0	I/O	MFP12	EPWM1 通道0 输出/捕获输入
28	36	PC.4	I/O	MFP0	通用数字I/O脚
		EBI_AD4	I/O	MFP2	EBI地址/数据总线位4.
		SPIM_D3	I/O	MFP3	SPIM QuadI/O模式下数据3管脚
		QSPI0_MOSI1	I/O	MFP4	QSPI0 MOSI1 (主机输出，从机输入) 管脚.
		SC1_nCD	I	MFP5	智能卡 1 卡检测管脚.
		I²S0_BCLK	O	MFP6	I2S0位时钟输出管脚
		SPI1_I2SMCLK	I/O	MFP7	SPI1 I²S 主机时钟输出脚
		UART2_RXD	I	MFP8	UART2 数据接收输入脚
		I²C1_SDA	I/O	MFP9	I²C1 数据输入/输出脚
		UART4_RXD	I	MFP11	UART4 数据接收输入脚
		EPWM1_CH1	I/O	MFP12	EPWM1 通道1 输出/捕获输入
29	37	PC.3	I/O	MFP0	通用数字I/O脚
		EBI_AD3	I/O	MFP2	EBI地址/数据总线位3.
		SPIM_SS	I/O	MFP3	SPIM 从机选择脚
		QSPI0_SS	I/O	MFP4	QSPI0 从机选择脚
		SC1_PWR	O	MFP5	智能卡 1电源脚
		I²S0_MCLK	O	MFP6	I2S0 主时钟输出脚
		SPI1_MISO	I/O	MFP7	SPI1 MISO(主机输入，从机输出)管脚
		UART2_nRTS	O	MFP8	UART2 请求发送输出脚
		I²C0_SMBAL	O	MFP9	I²C0 SMBus SMBALTER 管脚
		UART3_TXD	O	MFP11	UART3 数据发送输出脚.
		EPWM1_CH2	I/O	MFP12	EPWM1 通道2 输出/捕获输入
30	38	PC.2	I/O	MFP0	通用数字I/O脚
		EBI_AD2	I/O	MFP2	EBI地址/数据总线位2.
		SPIM_CLK	I/O	MFP3	SPIM 串行时钟脚
		QSPI0_CLK	I/O	MFP4	QSPI0 串行时钟脚
		SC1_RST	O	MFP5	智能卡 1 复位脚
		I²S0_DI	I	MFP6	I2S0 数据输入脚

		SPI1_MOSI	I/O	MFP7	SPI1 MOSI (主机输出，从机输入) 管脚
		UART2_nCTS	I	MFP8	UART2 禁止发送输入管脚
		I²C0_SMBSUS	O	MFP9	I²C0 SMBus SMBSUS 管脚 (PMBus 控制管脚)
		UART3_RXD	I	MFP11	UART3 数据接收输入脚
		EPWM1_CH3	I/O	MFP12	EPWM1 通道3输出/捕获输入脚
31	39	PC.1	I/O	MFP0	通用数字I/O脚
		EBI_AD1	I/O	MFP2	EBI地址/数据总线位1.
		SPIM_MISO	I/O	MFP3	SPIM MISO（主机输入，从机输出）脚.
		QSPI0_MISO0	I/O	MFP4	QSPI0 MISO0（主机输入，从机输出）脚.
		SC1_DAT	I/O	MFP5	智能卡 1 数据管脚.
		I²S0_DO	O	MFP6	I²S0 数据输出脚
		SPI1_CLK	I/O	MFP7	SPI1 串行时钟管脚
		UART2_TXD	O	MFP8	UART2 数据发送输出脚.
		I²C0_SCL	I/O	MFP9	I2C0 时钟管脚
		EPWM1_CH4	I/O	MFP12	EPWM1 通道4输出/捕获输入脚
		ACMP0_O	O	MFP14	模拟比较器 0输出脚
32	40	PC.0	I/O	MFP0	通用数字I/O脚
		EBI_AD0	I/O	MFP2	EBI地址/数据总线位0.
		SPIM_MOSI	I/O	MFP3	SPIM MOSI (主机输出，从机输入) 管脚.
		QSPI0_MOSI0	I/O	MFP4	QSPI0 MOSI0 (主机输出，从机输入) 管脚.
		SC1_CLK	O	MFP5	智能卡 1 时钟脚
		I²S0_LRCK	O	MFP6	I²S0左右通道时钟输出脚.
		SPI1_SS	I/O	MFP7	SPI1 从机选择脚
		UART2_RXD	I	MFP8	UART2 数据接收输入脚
		I²C0_SDA	I/O	MFP9	I2C0 数据输入/输出管脚
		EPWM1_CH5	I/O	MFP12	EPWM1 通道5输出/捕获输入
		ACMP1_O	O	MFP14	模拟比较器1输出脚
	41	PD.3	I/O	MFP0	通用数字I/O脚
		EBI_AD10	I/O	MFP2	EBI地址/数据总线位10.
		USCI0_CTL1	I/O	MFP3	USCI0 控制1 管脚.
		SPI0_SS	I/O	MFP4	SPI0 从机选择脚
		UART3_nRTS	O	MFP5	UART3 请求发送输出脚

		USCI1_CTL0	I/O	MFP6	USCI1 控制0管脚
		SC2_PWR	O	MFP7	智能卡 2电源脚
		SC1_nCD	I	MFP8	智能卡 1 卡检测管脚.
		UART0_TXD	O	MFP9	UART0 数据发送输出脚.
	42	PD.2	I/O	MFP0	通用数字I/O脚
		EBI_AD11	I/O	MFP2	EBI地址/数据总线位11.
		USCI0_DAT1	I/O	MFP3	USCI0 数据1 管脚.
		SPI0_CLK	I/O	MFP4	SPI0 串行时钟脚
		UART3_nCTS	I	MFP5	UART3 禁止发送输入管脚
		SC2_RST	O	MFP7	智能卡 2 复位脚
		UART0_RXD	I	MFP9	UART0 数据接收输入脚
	43	PD.1	I/O	MFP0	通用数字I/O脚
		EBI_AD12	I/O	MFP2	EBI地址/数据总线位12.
		USCI0_DAT0	I/O	MFP3	USCI0 数据0脚
		SPI0_MISO	I/O	MFP4	SPI0 MISO（主机输入，从机输出）脚.
		UART3_TXD	O	MFP5	UART3 数据发送输出脚.
		I^2C2_SCL	I/O	MFP6	I^2C2 时钟脚
		SC2_DAT	I/O	MFP7	智能卡 2 数据管脚.
	44	PD.0	I/O	MFP0	通用数字I/O脚
		EBI_AD13	I/O	MFP2	EBI地址/数据总线位13.
		USCI0_CLK	I/O	MFP3	USCI0 时钟脚
		SPI0_MOSI	I/O	MFP4	SPI0 MOSI (主机输出，从机输入) 管脚.
		UART3_RXD	I	MFP5	UART3 数据接收输入脚
		I^2C2_SDA	I/O	MFP6	I^2C2 数据输入/输出脚
		SC2_CLK	O	MFP7	智能卡 2 时钟脚
		TM2	I/O	MFP14	Timer2 事件计数输入/反转输出脚
33	45	PA.12	I/O	MFP0	通用数字I/O脚
		I^2S0_BCLK	O	MFP2	I2S0位时钟输出管脚
		UART4_TXD	O	MFP3	UART4 数据发送输出脚.
		I^2C1_SCL	I/O	MFP4	I^2C1 时钟脚
		SPI2_SS	I/O	MFP5	SPI2 从机选择脚
		SC2_PWR	O	MFP7	智能卡 2电源脚

		BPWM1_CH2	I/O	MFP11	BPWM1 通道2 输出/捕获输入
		QEI1_INDEX	I	MFP12	正交编码器1 索引输入
		USB_VBUS	P	MFP14	USB主机或HUB的电源
34	46	PA.13	I/O	MFP0	通用数字I/O脚
		I^2S0_MCLK	O	MFP2	I2S0 主时钟输出脚
		UART4_RXD	I	MFP3	UART4 数据接收输入脚
		I^2C1_SDA	I/O	MFP4	I^2C1 数据输入/输出脚
		SPI2_CLK	I/O	MFP5	SPI2 串行时钟脚
		SC2_RST	O	MFP7	智能卡 2 复位脚
		BPWM1_CH3	I/O	MFP11	BPWM1 通道3输出/捕获输入脚
		QEI1_A	I	MFP12	正交编码器1 相位A输入
		USB_D-	A	MFP14	USB差分信号D+.
35	47	PA.14	I/O	MFP0	通用数字I/O脚
		I^2S0_DI	I	MFP2	I2S0 数据输入脚
		UART0_TXD	O	MFP3	UART0 数据发送输出脚.
		SPI2_MISO	I/O	MFP5	SPI2 MISO（主机输入，从机输出）脚.
		I^2C2_SCL	I/O	MFP6	I^2C2 时钟脚
		SC2_DAT	I/O	MFP7	智能卡 2 数据管脚.
		BPWM1_CH4	I/O	MFP11	BPWM1 通道4输出/捕获输入脚
		QEI1_B	I	MFP12	正交编码器1 相位B 输入
		USB_D+	A	MFP14	USB差分信号D-.
36	48	PA.15	I/O	MFP0	通用数字I/O脚
		I^2S0_DO	O	MFP2	I^2S0 数据输出脚
		UART0_RXD	I	MFP3	UART0 数据接收输入脚
		SPI2_MOSI	I/O	MFP5	SPI2 MOSI (主机输出，从机输入) 管脚.
		I^2C2_SDA	I/O	MFP6	I^2C2 数据输入/输出脚
		SC2_CLK	O	MFP7	智能卡 2 时钟脚
		BPWM1_CH5	I/O	MFP11	EPWM1 通道5输出/捕获输入
		EPWM0_SYNC_IN	I	MFP12	EPWM0 计数器同步触发输入脚
		USB_OTG_ID	I	MFP14	USB_ 标识
37	49	V_{SS}	P	MFP0	数字电路地管脚
38	50	LDO_CAP	A	MFP0	LDO输出脚

39	51	V_{DD}	P	MFP0	I/O和内部PLL及数字电路的LDO电源
40	52	PC.14	I/O	MFP0	通用数字I/O脚
		EBI_AD11	I/O	MFP2	EBI地址/数据总线位11.
		SC1_nCD	I	MFP3	智能卡 1 卡检测管脚.
		SPI0_I2SMCLK	I/O	MFP4	SPI0 I^2S 主机时钟输出脚
		USCI0_CTL0	I/O	MFP5	USCI0 控制0 管脚.
		QSPI0_CLK	I/O	MFP6	QSPI0 串行时钟脚
		EPWM0_SYNC_IN	I	MFP11	EPWM0 计数器同步触发输入脚
		TM1	I/O	MFP13	Timer1 事件计数输入/反转输出脚
		USB_VBUS_ST	I	MFP14	USB 外部VBUS 电源状态管脚.
41	53	PB.15	I/O	MFP0	通用数字I/O脚
		EADC0_CH15	A	MFP1	EADC0 通道15 模拟输入
		EBI_AD12	I/O	MFP2	EBI地址/数据总线位12.
		SC1_PWR	O	MFP3	智能卡 1电源脚
		SPI0_SS	I/O	MFP4	SPI0 从机选择脚
		USCI0_CTL1	I/O	MFP5	USCI0 控制1 管脚.
		UART0_nCTS	I	MFP6	UART0 禁止发送输入管脚
		UART3_TXD	O	MFP7	UART3 数据发送输出脚.
		I^2C2_SMBAL	O	MFP8	I^2C2 SMBus SMBALTER 管脚
		EPWM1_CH0	I/O	MFP11	EPWM1 通道0 输出/捕获输入
		TM0_EXT	I/O	MFP13	Timer0 事件计数输入/反转输出管脚
		USB_VBUS_EN	O	MFP14	USB 外部 VBUS 电源使能管脚.
42	54	PB.14	I/O	MFP0	通用数字I/O脚
		EADC0_CH14	A	MFP1	EADC0 通道14 模拟输入
		EBI_AD13	I/O	MFP2	EBI地址/数据总线位13.
		SC1_RST	O	MFP3	智能卡 1 复位脚
		SPI0_CLK	I/O	MFP4	SPI0 串行时钟脚
		USCI0_DAT1	I/O	MFP5	USCI0 数据1 管脚.
		UART0_nRTS	O	MFP6	UART0 请求发送输出脚
		UART3_RXD	I	MFP7	UART3 数据接收输入脚
		I^2C2_SMBSUS	O	MFP8	I^2C2 SMBus SMBSUS 管脚 (PMBus 控制管脚)
		EPWM1_CH1	I/O	MFP11	EPWM1 通道1 输出/捕获输入

		TM1_EXT	I/O	MFP13	Timer1 事件计数输入/反转输出脚
		CLKO	O	MFP14	时钟输出
43	55	PB.13	I/O	MFP0	通用数字I/O脚
		EADC0_CH13	A	MFP1	EADC0 通道13 模拟输入
		DAC1_OUT	A	MFP1	DAC1 模拟输出通道
		ACMP0_P3	A	MFP1	模拟比较器 0 正极输入 3 管脚.
		ACMP1_P3	A	MFP1	模拟比较器 1 正极输入 3 管脚.
		EBI_AD14	I/O	MFP2	EBI地址/数据总线位14.
		SC1_DAT	I/O	MFP3	智能卡 1 数据管脚.
		SPI0_MISO	I/O	MFP4	SPI0 MISO（主机输入，从机输出）脚.
		USCI0_DAT0	I/O	MFP5	USCI0 数据0脚
		UART0_TXD	O	MFP6	UART0 数据发送输出脚.
		UART3_nRTS	O	MFP7	UART3 请求发送输出脚
		I^2C2_SCL	I/O	MFP8	I^2C2 时钟脚
		EPWM1_CH2	I/O	MFP11	EPWM1 通道2 输出/捕获输入
		TM2_EXT	I/O	MFP13	Timer2 事件计数输入/反转输出脚
44	56	PB.12	I/O	MFP0	通用数字I/O脚
		EADC0_CH12	A	MFP1	EADC0 通道12 模拟输入
		DAC0_OUT	A	MFP1	DAC0 模拟输出通道
		ACMP0_P2	A	MFP1	模拟比较器 0 正极输入 2 管脚.
		ACMP1_P2	A	MFP1	模拟比较器 1 正极输入 2 管脚.
		EBI_AD15	I/O	MFP2	EBI地址/数据总线位15.
		SC1_CLK	O	MFP3	智能卡 1 时钟脚
		SPI0_MOSI	I/O	MFP4	SPI0 MOSI (主机输出，从机输入) 管脚.
		USCI0_CLK	I/O	MFP5	USCI0 时钟脚
		UART0_RXD	I	MFP6	UART0 数据接收输入脚
		UART3_nCTS	I	MFP7	UART3 禁止发送输入管脚
		I^2C2_SDA	I/O	MFP8	I^2C2 数据输入/输出脚
		SD0_nCD	I	MFP9	SD/SDIO0 卡检测输入管脚
		EPWM1_CH3	I/O	MFP11	EPWM1 通道3输出/捕获输入脚
		TM3_EXT	I/O	MFP13	Timer3 事件计数输入/反转输出脚
45	57	AV_{DD}	P	MFP0	内部模拟电路电源

	58	V_{REF}	A	MFP0	ADC参考电压输入. 注: 此管脚需接1uF电容到地.
46	59	AV_{SS}	P	MFP0	模拟电路地管脚
	60	PB.11	I/O	MFP0	通用数字I/O脚
		EADC0_CH11	A	MFP1	EADC0 通道11 模拟输入
		EBI_ADR16	O	MFP2	EBI 地址总线位 16.
		UART0_nCTS	I	MFP5	UART0 禁止发送输入管脚
		UART4_TXD	O	MFP6	UART4 数据发送输出脚.
		I^2C1_SCL	I/O	MFP7	I^2C1 时钟脚
		SPI0_I2SMCLK	I/O	MFP9	SPI0 I^2S 主机时钟输出脚
		BPWM1_CH0	I/O	MFP10	BPWM1 通道0 输出/捕获输入
		SPI3_CLK	I/O	MFP11	SPI3 串行时钟脚
	61	PB.10	I/O	MFP0	通用数字I/O脚
		EADC0_CH10	A	MFP1	EADC0 通道10 模拟输入
		EBI_ADR17	O	MFP2	EBI 地址总线位 17.
		USCI1_CTL0	I/O	MFP4	USCI1 控制0管脚
		UART0_nRTS	O	MFP5	UART0 请求发送输出脚
		UART4_RXD	I	MFP6	UART4 数据接收输入脚
		I^2C1_SDA	I/O	MFP7	I^2C1 数据输入/输出脚
		BPWM1_CH1	I/O	MFP10	BPWM1 通道1 输出/捕获输入
		SPI3_SS	I/O	MFP11	SPI3 从机选择脚
	62	PB.9	I/O	MFP0	通用数字I/O脚
		EADC0_CH9	A	MFP1	EADC0 通道9 模拟输入
		EBI_ADR18	O	MFP2	EBI 地址总线位 18.
		USCI1_CTL1	I/O	MFP4	USCI1 控制1管脚
		UART0_TXD	O	MFP5	UART0 数据发送输出脚.
		UART1_nCTS	I	MFP6	UART1 禁止发送输入管脚
		I^2C1_SMBAL	O	MFP7	I^2C1 SMBus SMBALTER 管脚
		BPWM1_CH2	I/O	MFP10	BPWM1 通道2 输出/捕获输入
		SPI3_MISO	I/O	MFP11	SPI3 MISO（主机输入，从机输出）脚.
		INT7	I	MFP13	外部中断7 输入脚
	63	PB.8	I/O	MFP0	通用数字I/O脚
		EADC0_CH8	A	MFP1	EADC0 通道8 模拟输入

		EBI_ADR19	O	MFP2	EBI 地址总线位 19.
		USCI1_CLK	I/O	MFP4	USCI1 时钟脚
		UART0_RXD	I	MFP5	UART0 数据接收输入脚
		UART1_nRTS	O	MFP6	UART1 请求发送输出脚
		I^2C1_SMBSUS	O	MFP7	I^2C1 SMBus SMBSUS 管脚 (PMBus 控制管脚)
		BPWM1_CH3	I/O	MFP10	BPWM1 通道3输出/捕获输入脚
		SPI3_MOSI	I/O	MFP11	SPI3 MOSI (主机输出，从机输入) 管脚.
		INT6	I	MFP13	外部中断6 输入脚
47	64	PB.7	I/O	MFP0	通用数字I/O脚
		EADC0_CH7	A	MFP1	EADC0 通道7 模拟输入
		EBI_nWRL	O	MFP2	EBI低字节写使能输出脚
		USCI1_DAT0	I/O	MFP4	USCI1 数据0脚
		UART1_TXD	O	MFP6	UART1 数据发送输出脚
		SD1_CMD	I/O	MFP7	SD/SDIO1 命令/响应脚
		EBI_nCS0	O	MFP8	EBI片选0输出脚
		BPWM1_CH4	I/O	MFP10	BPWM1 通道4输出/捕获输入脚
		EPWM1_BRAKE0	I	MFP11	EPWM1 刹车0 输入脚
		EPWM1_CH4	I/O	MFP12	EPWM1 通道4输出/捕获输入脚
		INT5	I	MFP13	外部中断5 输入脚
		USB_VBUS_ST	I	MFP14	USB 外部VBUS 电源状态管脚.
		ACMP0_O	O	MFP15	模拟比较器 0输出脚

4.2.3 M483系列管脚描述

64管脚	128管脚	管脚名称	类型	MFP	描述
2	1	PB.5	I/O	MFP0	通用数字I/O脚
		EADC0_CH5	A	MFP1	EADC0通道5模拟输入
		ACMP1_N	A	MFP1	模拟比较器1负极输入管脚
		EBI_ADR0	O	MFP2	EBI地址总线位0
		SD0_DAT3	I/O	MFP3	SD/SDIO0数据线位3
		SPI1_MISO	I/O	MFP5	SPI1 MISO(主机输入，从机输出)管脚
		I^2C0_SCL	I/O	MFP6	I2C0 时钟管脚
		UART5_TXD	O	MFP7	UART5数据发送输出脚
		USCI1_CTL0	I/O	MFP8	USCI1 控制0管脚
		SC0_CLK	O	MFP9	智能卡0时钟管脚
		I^2S0_BCLK	O	MFP10	I2S0位时钟输出管脚
		EPWM0_CH0	I/O	MFP11	EPWM0 通道0输出/捕获输入
		TM0	I/O	MFP14	Timer0 事件计数输入/反转输出管脚
		INT0	I	MFP15	外部中断0输入管脚
3	2	PB.4	I/O	MFP0	通用数字I/O脚
		EADC0_CH4	A	MFP1	EADC0 通道4模拟输入
		ACMP1_P1	A	MFP1	模拟比较器1正极输入1脚
		EBI_ADR1	O	MFP2	EBI 地址总线位1
		SD0_DAT2	I/O	MFP3	SD/SDIO0 数据线位2
		SPI1_MOSI	I/O	MFP5	SPI1 MOSI (主机输出，从机输入) 管脚
		I^2C0_SDA	I/O	MFP6	I2C0 数据输入/输出管脚
		UART5_RXD	I	MFP7	UART5 数据接收输入管脚
		USCI1_CTL1	I/O	MFP8	USCI1 控制1管脚
		SC0_DAT	I/O	MFP9	智能卡0数据输入脚
		I^2S0_MCLK	O	MFP10	I2S0 主时钟输出脚
		EPWM0_CH1	I/O	MFP11	EPWM0 通道1输出/捕获输入脚
		TM1	I/O	MFP14	Timer1 事件计数输入/反转输出脚
		INT1	I	MFP15	外部中断1输入脚
4	3	PB.3	I/O	MFP0	通用数字I/O脚

64 管脚	128 管脚	管脚名称	类型	MFP	描述
		EADC0_CH3	A	MFP1	EADC0 通道3模拟输入
		ACMP0_N	A	MFP1	模拟比较器0负极输入脚
		EBI_ADR2	O	MFP2	EBI 地址总线位2
		SD0_DAT1	I/O	MFP3	SD/SDIO0 数据线位1
		SPI1_CLK	I/O	MFP5	SPI1 串行时钟管脚
		UART1_TXD	O	MFP6	UART1 数据发送输出脚
		UART5_nRTS	O	MFP7	UART5 请求发送输出脚
		USCI1_DAT1	I/O	MFP8	USCI1数据1管脚.
		SC0_RST	O	MFP9	智能卡0复位脚
		I^2S0_DI	I	MFP10	I2S0 数据输入脚
		EPWM0_CH2	I/O	MFP11	EPWM0 通道2输出/捕获输入脚
		TM2	I/O	MFP14	Timer2 事件计数输入/反转输出脚
		INT2	I	MFP15	外部中断2输入脚
5	4	PB.2	I/O	MFP0	通用数字I/O脚
		EADC0_CH2	A	MFP1	EADC0 通道2 模拟输入
		ACMP0_P1	A	MFP1	模拟比较器0正极输入1脚
		OPA0_O	A	MFP1	运放0输出脚
		EBI_ADR3	O	MFP2	EBI 地址总线位3
		SD0_DAT0	I/O	MFP3	SD/SDIO0 数据线位0
		SPI1_SS	I/O	MFP5	SPI1 从机选择脚
		UART1_RXD	I	MFP6	UART1数据接收输入管脚
		UART5_nCTS	I	MFP7	UART5 禁止发送输入管脚
		USCI1_DAT0	I/O	MFP8	USCI1 数据0脚
		SC0_PWR	O	MFP9	智能卡0电源脚
		I^2S0_DO	O	MFP10	I^2S0 数据输出脚
		EPWM0_CH3	I/O	MFP11	EPWM0 通道3输出/捕获输入脚
		TM3	I/O	MFP14	Timer3 事件计数输入/反转输出脚
		INT3	I	MFP15	外部中断3输入脚
	5	PC.12	I/O	MFP0	通用数字I/O脚
		EBI_ADR4	O	MFP2	EBI 地址总线位 4.
		UART0_TXD	O	MFP3	UART0 数据发送输出脚.

64 管脚	128 管脚	管脚名称	类型	MFP	描述
		I^2C0_SCL	I/O	MFP4	I2C0 时钟管脚
		SPI3_MISO	I/O	MFP6	SPI3 MISO（主机输入，从机输出）脚.
		SC0_nCD	I	MFP9	智能卡 0 卡检测管脚.
		ECAP1_IC2	I	MFP11	增强型捕获计数器 1 输入 2 管脚.
		EPWM1_CH0	I/O	MFP12	EPWM1 通道0 输出/捕获输入
		ACMP0_O	O	MFP14	模拟比较器 0输出脚
	6	PC.11	I/O	MFP0	通用数字I/O脚
		EBI_ADR5	O	MFP2	EBI 地址总线位 5.
		UART0_RXD	I	MFP3	UART0 数据接收输入脚
		I^2C0_SDA	I/O	MFP4	I2C0 数据输入/输出管脚
		SPI3_MOSI	I/O	MFP6	SPI3 MOSI (主机输出，从机输入) 管脚.
		ECAP1_IC1	I	MFP11	增强型捕获计数器 1 输入 1 管脚.
		EPWM1_CH1	I/O	MFP12	EPWM1 通道1 输出/捕获输入
		ACMP1_O	O	MFP14	模拟比较器1输出脚
	7	PC.10	I/O	MFP0	通用数字I/O脚
		EBI_ADR6	O	MFP2	EBI 地址总线位 6.
		SPI3_CLK	I/O	MFP6	SPI3 串行时钟脚
		UART3_TXD	O	MFP7	UART3 数据发送输出脚.
		CAN1_TXD	O	MFP9	CAN1总线发送输出
		ECAP1_IC0	I	MFP11	增强型捕获计数器 1 输入脚0
		EPWM1_CH2	I/O	MFP12	EPWM1 通道2 输出/捕获输入
	8	PC.9	I/O	MFP0	通用数字I/O脚
		EBI_ADR7	O	MFP2	EBI 地址总线位 7.
		SPI3_SS	I/O	MFP6	SPI3 从机选择脚
		UART3_RXD	I	MFP7	UART3 数据接收输入脚
		CAN1_RXD	I	MFP9	CAN1 总线接收输入
		EPWM1_CH3	I/O	MFP12	EPWM1 通道3输出/捕获输入脚
6	9	PB.1	I/O	MFP0	通用数字I/O脚
		EADC0_CH1	A	MFP1	EADC0 通道1 模拟输入
		OPA0_N	A	MFP1	运放0负极输入脚
		EBI_ADR8	O	MFP2	EBI 地址总线位 8.

64 管脚	128 管脚	管脚名称	类型	MFP	描述
		SD0_CLK	O	MFP3	SD/SDIO0 时钟输出脚
		SPI1_I2SMCLK	I/O	MFP5	SPI1 I^2S 主机时钟输出脚
		SPI3_I2SMCLK	I/O	MFP6	SPI3 I^2S 主机时钟输出脚
		UART2_TXD	O	MFP7	UART2 数据发送输出脚.
		USCI1_CLK	I/O	MFP8	USCI1 时钟脚
		I^2C1_SCL	I/O	MFP9	I^2C1 时钟脚
		I^2S0_LRCK	O	MFP10	I^2S0左右通道时钟输出脚.
		EPWM0_CH4	I/O	MFP11	EPWM0 通道4输出/捕获输入脚
		EPWM1_CH4	I/O	MFP12	EPWM1 通道4输出/捕获输入脚
		EPWM0_BRAKE0	I	MFP13	EPWM0 刹车0 输入脚
7	10	PB.0	I/O	MFP0	通用数字I/O脚
		EADC0_CH0	A	MFP1	EADC0 通道0 模拟输入
		OPA0_P	A	MFP1	运放 0 正极输入脚
		EBI_ADR9	O	MFP2	EBI 地址总线位 9.
		SD0_CMD	I/O	MFP3	SD/SDIO0 命令/响应脚
		UART2_RXD	I	MFP7	UART2 数据接收输入脚
		SPI0_I2SMCLK	I/O	MFP8	SPI0 I^2S 主机时钟输出脚
		I^2C1_SDA	I/O	MFP9	I^2C1 数据输入/输出脚
		EPWM0_CH5	I/O	MFP11	EPWM0 通道5 输出/捕获输入
		EPWM1_CH5	I/O	MFP12	EPWM1 通道5输出/捕获输入
		EPWM0_BRAKE1	I	MFP13	EPWM0 刹车1 输入脚
	11	V_{SS}	P	MFP0	数字电路地管脚
	12	V_{DD}	P	MFP0	I/O和内部PLL及数字电路的LDO电源
8	13	PA.11	I/O	MFP0	通用数字I/O脚
		ACMP0_P0	A	MFP1	模拟比较器 0 正极输入脚0
		EBI_nRD	O	MFP2	EBI 读使能输出脚
		SC2_PWR	O	MFP3	智能卡 2电源脚
		SPI2_SS	I/O	MFP4	SPI2 从机选择脚
		SD1_DAT3	I/O	MFP5	SD/SDIO1 数据线位3.
		USCI0_CLK	I/O	MFP6	USCI0 时钟脚
		I^2C2_SCL	I/O	MFP7	I^2C2 时钟脚

64 管脚	128 管脚	管脚名称	类型	MFP	描述
		BPWM0_CH0	I/O	MFP9	BPWM0 通道0 输出/捕获输入
		EPWM0_SYNC_OUT	O	MFP10	EPWM0计数器同步触发输出脚
		TM0_EXT	I/O	MFP13	Timer0 事件计数输入/反转输出管脚
		DAC1_ST	I	MFP14	DAC1 外部触发输入
9	14	PA.10	I/O	MFP0	通用数字I/O脚
		ACMP1_P0	A	MFP1	模拟比较器 1 正极输入脚0
		OPA1_O	A	MFP1	运放 1输出脚
		EBI_nWR	O	MFP2	EBI 写使能输出脚
		SC2_RST	O	MFP3	智能卡 2 复位脚
		SPI2_CLK	I/O	MFP4	SPI2 串行时钟脚
		SD1_DAT2	I/O	MFP5	SD/SDIO1 数据线位2.
		USCI0_DAT0	I/O	MFP6	USCI0 数据0脚
		I^2C2_SDA	I/O	MFP7	I^2C2 数据输入/输出脚
		BPWM0_CH1	I/O	MFP9	BPWM0 通道1 输出/捕获输入
		QEI1_INDEX	I	MFP10	正交编码器1 索引输入
		ECAP0_IC0	I	MFP11	增强型捕获计数器 0 输入脚0
		TM1_EXT	I/O	MFP13	Timer1 事件计数输入/反转输出脚
		DAC0_ST	I	MFP14	DAC0 外部触发输入
		SWDH_CLK	O	MFP15	串行调试总线主时钟输出
10	15	PA.9	I/O	MFP0	通用数字I/O脚
		OPA1_N	A	MFP1	运放 1 负极输入脚
		EBI_MCLK	O	MFP2	EBI 外部时钟输出脚.
		SC2_DAT	I/O	MFP3	智能卡 2 数据管脚.
		SPI2_MISO	I/O	MFP4	SPI2 MISO（主机输入，从机输出）脚.
		SD1_DAT1	I/O	MFP5	SD/SDIO1 数据线位1.
		USCI0_DAT1	I/O	MFP6	USCI0 数据1 管脚.
		UART1_TXD	O	MFP7	UART1 数据发送输出脚
		BPWM0_CH2	I/O	MFP9	BPWM0 通道2 输出/捕获输入
		QEI1_A	I	MFP10	正交编码器1 相位A输入
		ECAP0_IC1	I	MFP11	增强型捕获计数器 0 输入 1 管脚.
		TM2_EXT	I/O	MFP13	Timer2 事件计数输入/反转输出脚

64 管脚	128 管脚	管脚名称	类型	MFP	描述
		SWDH_DAT	I/O	MFP15	串行调试总线主机数据输入/输出管脚
11	16	PA.8	I/O	MFP0	通用数字I/O脚
		OPA1_P	A	MFP1	运放 1 正极输入脚
		EBI_ALE	O	MFP2	EBI地址锁存使能输出脚
		SC2_CLK	O	MFP3	智能卡 2 时钟脚
		SPI2_MOSI	I/O	MFP4	SPI2 MOSI (主机输出，从机输入) 管脚.
		SD1_DAT0	I/O	MFP5	SD/SDIO1 数据线位0.
		USCI0_CTL1	I/O	MFP6	USCI0 控制1 管脚.
		UART1_RXD	I	MFP7	UART1数据接收输入管脚
		BPWM0_CH3	I/O	MFP9	BPWM0 通道3输出/捕获输入脚
		QEI1_B	I	MFP10	正交编码器1 相位B 输入
		ECAP0_IC2	I	MFP11	增强型捕获计数器 0 输入 2 管脚.
		TM3_EXT	I/O	MFP13	Timer3 事件计数输入/反转输出脚
		INT4	I	MFP15	外部中断4输入管脚
	17	PC.13	I/O	MFP0	通用数字I/O脚
		EBI_ADR10	O	MFP2	EBI 地址总线位 10.
		SC2_nCD	I	MFP3	智能卡 2 卡检测管脚.
		SPI2_I2SMCLK	I/O	MFP4	SPI2 I^2S 主机时钟输出脚
		CAN1_TXD	O	MFP5	CAN1总线发送输出
		USCI0_CTL0	I/O	MFP6	USCI0 控制0 管脚.
		UART2_TXD	O	MFP7	UART2 数据发送输出脚.
		BPWM0_CH4	I/O	MFP9	BPWM0 通道4输出/捕获输入脚
		CLKO	O	MFP13	时钟输出
		EADC0_ST	I	MFP14	EADC0 外部触发输入
	18	PD.12	I/O	MFP0	通用数字I/O脚
		OPA2_O	A	MFP1	运放 2输出脚
		EBI_nCS0	O	MFP2	EBI片选0输出脚
		CAN1_RXD	I	MFP5	CAN1 总线接收输入
		UART2_RXD	I	MFP7	UART2 数据接收输入脚
		BPWM0_CH5	I/O	MFP9	BPWM0 通道5 输出/捕获输入
		QEI0_INDEX	I	MFP10	正交编码器0 索引输入

64 管脚	128 管脚	管脚名称	类型	MFP	描述
		CLKO	O	MFP13	时钟输出
		EADC0_ST	I	MFP14	EADC0 外部触发输入
		INT5	I	MFP15	外部中断5 输入脚
	19	PD.11	I/O	MFP0	通用数字I/O脚
		OPA2_N	A	MFP1	运放 2 负极输入脚
		EBI_nCS1	O	MFP2	EBI片选1输出管脚
		UART1_TXD	O	MFP3	UART1 数据发送输出脚
		CAN0_TXD	O	MFP4	CAN0总线发送输出
		QEI0_A	I	MFP10	正交编码器0 相位A输入
		INT6	I	MFP15	外部中断6 输入脚
	20	PD.10	I/O	MFP0	通用数字I/O脚
		OPA2_P	A	MFP1	运放 2 正极输入脚
		EBI_nCS2	O	MFP2	EBI片选2输出脚
		UART1_RXD	I	MFP3	UART1数据接收输入管脚
		CAN0_RXD	I	MFP4	CAN0 总线接收输入
		QEI0_B	I	MFP10	正交编码器0 相位B 输入
		INT7	I	MFP15	外部中断7 输入脚
	21	PG.2	I/O	MFP0	通用数字I/O脚
		EBI_ADR11	O	MFP2	EBI 地址总线位 11.
		SPI2_SS	I/O	MFP3	SPI2 从机选择脚
		I^2C0_SMBAL	O	MFP4	I^2C0 SMBus SMBALTER 管脚
		I^2C1_SCL	I/O	MFP5	I^2C1 时钟脚
		TM0	I/O	MFP13	Timer0 事件计数输入/反转输出管脚
	22	PG.3	I/O	MFP0	通用数字I/O脚
		EBI_ADR12	O	MFP2	EBI 地址总线位 12.
		SPI2_CLK	I/O	MFP3	SPI2 串行时钟脚
		I^2C0_SMBSUS	O	MFP4	I^2C0 SMBus SMBSUS 管脚 (PMBus 控制管脚)
		I^2C1_SDA	I/O	MFP5	I^2C1 数据输入/输出脚
		TM1	I/O	MFP13	Timer1 事件计数输入/反转输出脚
	23	PG.4	I/O	MFP0	通用数字I/O脚
		EBI_ADR13	O	MFP2	EBI 地址总线位 13.

64 管脚	128 管脚	管脚名称	类型	MFP	描述
		SPI2_MISO	I/O	MFP3	SPI2 MISO（主机输入，从机输出）脚.
		TM2	I/O	MFP13	Timer2 事件计数输入/反转输出脚
	24	PF.11	I/O	MFP0	通用数字I/O脚
		EBI_ADR14	O	MFP2	EBI 地址总线位 14.
		SPI2_MOSI	I/O	MFP3	SPI2 MOSI (主机输出，从机输入) 管脚.
		TAMPER5	I/O	MFP10	TAMPER循环检测管脚 5.
		TM3	I/O	MFP13	Timer3 事件计数输入/反转输出脚
	25	PF.10	I/O	MFP0	通用数字I/O脚
		EBI_ADR15	O	MFP2	EBI 地址总线位 15.
		SC0_nCD	I	MFP3	智能卡 0 卡检测管脚.
		I^2S0_BCLK	O	MFP4	I2S0位时钟输出管脚
		SPI0_I2SMCLK	I/O	MFP5	SPI0 I^2S 主机时钟输出脚
		TAMPER4	I/O	MFP10	TAMPER循环检测管脚 4.
	26	PF.9	I/O	MFP0	通用数字I/O脚
		EBI_ADR16	O	MFP2	EBI 地址总线位 16.
		SC0_PWR	O	MFP3	智能卡0电源脚
		I^2S0_MCLK	O	MFP4	I2S0 主时钟输出脚
		SPI0_SS	I/O	MFP5	SPI0 从机选择脚
		TAMPER3	I/O	MFP10	TAMPER循环检测管脚 3.
	27	PF.8	I/O	MFP0	通用数字I/O脚
		EBI_ADR17	O	MFP2	EBI 地址总线位 17.
		SC0_RST	O	MFP3	智能卡0复位脚
		I^2S0_DI	I	MFP4	I2S0 数据输入脚
		SPI0_CLK	I/O	MFP5	SPI0 串行时钟脚
		TAMPER2	I/O	MFP10	TAMPER循环检测管脚 2.
	28	PF.7	I/O	MFP0	通用数字I/O脚
		EBI_ADR18	O	MFP2	EBI 地址总线位 18.
		SC0_DAT	I/O	MFP3	智能卡0数据输入脚
		I^2S0_DO	O	MFP4	I^2S0 数据输出脚
		SPI0_MISO	I/O	MFP5	SPI0 MISO（主机输入，从机输出）脚.
		UART4_TXD	O	MFP6	UART4 数据发送输出脚.

64 管脚	128 管脚	管脚名称	类型	MFP	描述
		TAMPER1	I/O	MFP10	TAMPER循环检测管脚 1.
12	29	PF.6	I/O	MFP0	通用数字I/O脚
		EBI_ADR19	O	MFP2	EBI 地址总线位 19.
		SC0_CLK	O	MFP3	智能卡0时钟管脚
		I^2S0_LRCK	O	MFP4	I^2S0左右通道时钟输出脚.
		SPI0_MOSI	I/O	MFP5	SPI0 MOSI (主机输出，从机输入) 管脚.
		UART4_RXD	I	MFP6	UART4 数据接收输入脚
		EBI_nCS0	O	MFP7	EBI片选0输出脚
		TAMPER0	I/O	MFP10	TAMPER循环检测管脚 0.
13	30	V_{DD}	P	MFP0	I/O和内部PLL及数字电路的LDO电源
14	31	PF.5	I/O	MFP0	通用数字I/O脚
		UART2_RXD	I	MFP2	UART2 数据接收输入脚
		UART2_nCTS	I	MFP4	UART2 禁止发送输入管脚
		BPWM0_CH4	I/O	MFP8	BPWM0 通道4输出/捕获输入脚
		EPWM0_SYNC_OUT	O	MFP9	EPWM0计数器同步触发输出脚
		X32_IN	I	MFP10	外部 32.768 kHz 晶振输入脚
		EADC0_ST	I	MFP11	EADC0 外部触发输入
15	32	PF.4	I/O	MFP0	通用数字I/O脚
		UART2_TXD	O	MFP2	UART2 数据发送输出脚.
		UART2_nRTS	O	MFP4	UART2 请求发送输出脚
		BPWM0_CH5	I/O	MFP8	BPWM0 通道5 输出/捕获输入
		X32_OUT	O	MFP10	外部 32.768 kHz 晶振输出脚
	33	PH.4	I/O	MFP0	通用数字I/O脚
		EBI_ADR3	O	MFP2	EBI 地址总线位3
		SPI1_MISO	I/O	MFP3	SPI1 MISO(主机输入，从机输出)管脚
	34	PH.5	I/O	MFP0	通用数字I/O脚
		EBI_ADR2	O	MFP2	EBI 地址总线位2
		SPI1_MOSI	I/O	MFP3	SPI1 MOSI (主机输出，从机输入) 管脚
	35	PH.6	I/O	MFP0	通用数字I/O脚
		EBI_ADR1	O	MFP2	EBI 地址总线位1
		SPI1_CLK	I/O	MFP3	SPI1 串行时钟管脚

64 管脚	128 管脚	管脚名称	类型	MFP	描述
	36	PH.7	I/O	MFP0	通用数字I/O脚
		EBI_ADR0	O	MFP2	EBI地址总线位0
		SPI1_SS	I/O	MFP3	SPI1 从机选择脚
16	37	PF.3	I/O	MFP0	通用数字I/O脚
		EBI_nCS0	O	MFP2	EBI片选0输出脚
		UART0_TXD	O	MFP3	UART0 数据发送输出脚.
		I^2C0_SCL	I/O	MFP4	I2C0 时钟管脚
		XT1_IN	I	MFP10	外部 4~24 MHz (高速) 晶振输入脚
		BPWM1_CH0	I/O	MFP11	BPWM1 通道0 输出/捕获输入
17	38	PF.2	I/O	MFP0	通用数字I/O脚
		EBI_nCS1	O	MFP2	EBI片选1输出管脚
		UART0_RXD	I	MFP3	UART0 数据接收输入脚
		I^2C0_SDA	I/O	MFP4	I2C0 数据输入/输出管脚
		QSPI0_CLK	I/O	MFP5	QSPI0 串行时钟脚
		XT1_OUT	O	MFP10	外部 4~24 MHz (高速) 晶振输出脚
		BPWM1_CH1	I/O	MFP11	BPWM1 通道1 输出/捕获输入
	39	V_{SS}	P	MFP0	数字电路地管脚
	40	V_{DD}	P	MFP0	I/O和内部PLL及数字电路的LDO电源
	41	PE.8	I/O	MFP0	通用数字I/O脚
		EBI_ADR10	O	MFP2	EBI 地址总线位 10.
		I^2S0_BCLK	O	MFP4	I2S0位时钟输出管脚
		SPI2_CLK	I/O	MFP5	SPI2 串行时钟脚
		USCI1_CTL1	I/O	MFP6	USCI1 控制1管脚
		UART2_TXD	O	MFP7	UART2 数据发送输出脚.
		EPWM0_CH0	I/O	MFP10	EPWM0 通道0输出/捕获输入
		EPWM0_BRAKE0	I	MFP11	EPWM0 刹车0 输入脚
		ECAP0_IC0	I	MFP12	增强型捕获计数器 0 输入脚0
		TRACE_CLK	O	MFP14	ETM跟踪时钟输出脚
	42	PE.9	I/O	MFP0	通用数字I/O脚
		EBI_ADR11	O	MFP2	EBI 地址总线位 11.
		I^2S0_MCLK	O	MFP4	I2S0 主时钟输出脚

64 管脚	128 管脚	管脚名称	类型	MFP	描述
		SPI2_MISO	I/O	MFP5	SPI2 MISO（主机输入，从机输出）脚.
		USCI1_CTL0	I/O	MFP6	USCI1 控制0管脚
		UART2_RXD	I	MFP7	UART2 数据接收输入脚
		EPWM0_CH1	I/O	MFP10	EPWM0 通道1输出/捕获输入脚
		EPWM0_BRAKE1	I	MFP11	EPWM0 刹车1 输入脚
		ECAP0_IC1	I	MFP12	增强型捕获计数器 0 输入 1 管脚.
		TRACE_DATA0	O	MFP14	ETM跟踪数据0 输出管脚
	43	PE.10	I/O	MFP0	通用数字I/O脚
		EBI_ADR12	O	MFP2	EBI 地址总线位 12.
		I^2S0_DI	I	MFP4	I2S0 数据输入脚
		SPI2_MOSI	I/O	MFP5	SPI2 MOSI (主机输出，从机输入) 管脚.
		USCI1_DAT0	I/O	MFP6	USCI1 数据0脚
		UART3_TXD	O	MFP7	UART3 数据发送输出脚.
		EPWM0_CH2	I/O	MFP10	EPWM0 通道2输出/捕获输入脚
		EPWM1_BRAKE0	I	MFP11	EPWM1 刹车0 输入脚
		ECAP0_IC2	I	MFP12	增强型捕获计数器 0 输入 2 管脚.
		TRACE_DATA1	O	MFP14	ETM跟踪数据1 输出管脚
	44	PE.11	I/O	MFP0	通用数字I/O脚
		EBI_ADR13	O	MFP2	EBI 地址总线位 13.
		I^2S0_DO	O	MFP4	I^2S0 数据输出脚
		SPI2_SS	I/O	MFP5	SPI2 从机选择脚
		USCI1_DAT1	I/O	MFP6	USCI1数据1管脚.
		UART3_RXD	I	MFP7	UART3 数据接收输入脚
		UART1_nCTS	I	MFP8	UART1 禁止发送输入管脚
		EPWM0_CH3	I/O	MFP10	EPWM0 通道3输出/捕获输入脚
		EPWM1_BRAKE1	I	MFP11	EPWM1 刹车1输入管脚
		ECAP1_IC2	I	MFP13	增强型捕获计数器 1 输入 2 管脚.
		TRACE_DATA2	O	MFP14	ETM跟踪数据2 输出管脚
	45	PE.12	I/O	MFP0	通用数字I/O脚
		EBI_ADR14	O	MFP2	EBI 地址总线位 14.
		I^2S0_LRCK	O	MFP4	I^2S0左右通道时钟输出脚.

64 管脚	128 管脚	管脚名称	类型	MFP	描述
		SPI2_I2SMCLK	I/O	MFP5	SPI2 I²S 主机时钟输出脚
		USCI1_CLK	I/O	MFP6	USCI1 时钟脚
		UART1_nRTS	O	MFP8	UART1 请求发送输出脚
		EPWM0_CH4	I/O	MFP10	EPWM0 通道4输出/捕获输入脚
		ECAP1_IC1	I	MFP13	增强型捕获计数器 1 输入 1 管脚.
		TRACE_DATA3	O	MFP14	ETM跟踪数据3 输出管脚
	46	PE.13	I/O	MFP0	通用数字I/O脚
		EBI_ADR15	O	MFP2	EBI 地址总线位 15.
		I²C0_SCL	I/O	MFP4	I2C0 时钟管脚
		UART4_nRTS	O	MFP5	UART4 请求发送输出脚
		UART1_TXD	O	MFP8	UART1 数据发送输出脚
		EPWM0_CH5	I/O	MFP10	EPWM0 通道5 输出/捕获输入
		EPWM1_CH0	I/O	MFP11	EPWM1 通道0 输出/捕获输入
		BPWM1_CH5	I/O	MFP12	EPWM1 通道5输出/捕获输入
		ECAP1_IC0	I	MFP13	增强型捕获计数器 1 输入脚0
	47	PC.8	I/O	MFP0	通用数字I/O脚
		EBI_ADR16	O	MFP2	EBI 地址总线位 16.
		I²C0_SDA	I/O	MFP4	I2C0 数据输入/输出管脚
		UART4_nCTS	I	MFP5	UART4 禁止发送输入管脚
		UART1_RXD	I	MFP8	UART1数据接收输入管脚
		EPWM1_CH1	I/O	MFP11	EPWM1 通道1 输出/捕获输入
		BPWM1_CH4	I/O	MFP12	BPWM1 通道4输出/捕获输入脚
18	48	PC.7	I/O	MFP0	通用数字I/O脚
		EBI_AD9	I/O	MFP2	EBI地址/数据总线位9.
		SPI1_MISO	I/O	MFP4	SPI1 MISO(主机输入，从机输出)管脚
		UART4_TXD	O	MFP5	UART4 数据发送输出脚.
		SC2_PWR	O	MFP6	智能卡 2电源脚
		UART0_nCTS	I	MFP7	UART0 禁止发送输入管脚
		I²C1_SMBAL	O	MFP8	I²C1 SMBus SMBALTER 管脚
		EPWM1_CH2	I/O	MFP11	EPWM1 通道2 输出/捕获输入
		BPWM1_CH0	I/O	MFP12	BPWM1 通道0 输出/捕获输入

64 管脚	128 管脚	管脚名称	类型	MFP	描述
		TM0	I/O	MFP14	Timer0 事件计数输入/反转输出管脚
		INT3	I	MFP15	外部中断3输入脚
19	49	PC.6	I/O	MFP0	通用数字I/O脚
		EBI_AD8	I/O	MFP2	EBI地址/数据总线位8.
		SPI1_MOSI	I/O	MFP4	SPI1 MOSI (主机输出，从机输入) 管脚
		UART4_RXD	I	MFP5	UART4 数据接收输入脚
		SC2_RST	O	MFP6	智能卡 2 复位脚
		UART0_nRTS	O	MFP7	UART0 请求发送输出脚
		I^2C1_SMBSUS	O	MFP8	I^2C1 SMBus SMBSUS 管脚 (PMBus 控制管脚)
		EPWM1_CH3	I/O	MFP11	EPWM1 通道3输出/捕获输入脚
		BPWM1_CH1	I/O	MFP12	BPWM1 通道1 输出/捕获输入
		TM1	I/O	MFP14	Timer1 事件计数输入/反转输出脚
		INT2	I	MFP15	外部中断2输入脚
20	50	PA.7	I/O	MFP0	通用数字I/O脚
		EBI_AD7	I/O	MFP2	EBI地址/数据总线位7.
		SPI1_CLK	I/O	MFP4	SPI1 串行时钟管脚
		SC2_DAT	I/O	MFP6	智能卡 2 数据管脚.
		UART0_TXD	O	MFP7	UART0 数据发送输出脚.
		I^2C1_SCL	I/O	MFP8	I^2C1 时钟脚
		EPWM1_CH4	I/O	MFP11	EPWM1 通道4输出/捕获输入脚
		BPWM1_CH2	I/O	MFP12	BPWM1 通道2 输出/捕获输入
		ACMP0_WLAT	I	MFP13	模拟比较器 0窗口锁存输入管脚
		TM2	I/O	MFP14	Timer2 事件计数输入/反转输出脚
		INT1	I	MFP15	外部中断1输入脚
21	51	PA.6	I/O	MFP0	通用数字I/O脚
		EBI_AD6	I/O	MFP2	EBI地址/数据总线位6.
		SPI1_SS	I/O	MFP4	SPI1 从机选择脚
		SD1_nCD	I	MFP5	SD/SDIO1 卡检测输入管脚
		SC2_CLK	O	MFP6	智能卡 2 时钟脚
		UART0_RXD	I	MFP7	UART0 数据接收输入脚
		I^2C1_SDA	I/O	MFP8	I^2C1 数据输入/输出脚

64 管脚	128 管脚	管脚名称	类型	MFP	描述
		EPWM1_CH5	I/O	MFP11	EPWM1 通道5输出/捕获输入
		BPWM1_CH3	I/O	MFP12	BPWM1 通道3输出/捕获输入脚
		ACMP1_WLAT	I	MFP13	模拟比较器 1窗口锁存输入管脚
		TM3	I/O	MFP14	Timer3 事件计数输入/反转输出脚
		INT0	I	MFP15	外部中断0输入管脚
22	52	V_{SS}	P	MFP0	数字电路地管脚
23	53	V_{DD}	P	MFP0	I/O和内部PLL及数字电路的LDO电源
24	54	LDO_CAP	A	MFP0	LDO输出脚
25	55	PA.5	I/O	MFP0	通用数字I/O脚
		SPIM_D2	I/O	MFP2	SPIM QuadI/O模式下数据2管脚
		QSPI0_MISO1	I/O	MFP3	QSPI0 MISO1（主机输入，从机输出）脚.
		SPI1_I2SMCLK	I/O	MFP4	SPI1 I^2S 主机时钟输出脚
		SD1_CMD	I/O	MFP5	SD/SDIO1 命令/响应脚
		SC2_nCD	I	MFP6	智能卡 2 卡检测管脚.
		UART0_nCTS	I	MFP7	UART0 禁止发送输入管脚
		UART5_TXD	O	MFP8	UART5数据发送输出脚
		I^2C0_SCL	I/O	MFP9	I2C0 时钟管脚
		CAN0_TXD	O	MFP10	CAN0总线发送输出
		BPWM0_CH5	I/O	MFP12	BPWM0 通道5 输出/捕获输入
		EPWM0_CH0	I/O	MFP13	EPWM0 通道0输出/捕获输入
		QEI0_INDEX	I	MFP14	正交编码器0 索引输入
26	56	PA.4	I/O	MFP0	通用数字I/O脚
		SPIM_D3	I/O	MFP2	SPIM QuadI/O模式下数据3管脚
		QSPI0_MOSI1	I/O	MFP3	QSPI0 MOSI1 (主机输出，从机输入) 管脚.
		SPI0_I2SMCLK	I/O	MFP4	SPI0 I^2S 主机时钟输出脚
		SD1_CLK	O	MFP5	SD/SDIO1 时钟输出管脚
		SC0_nCD	I	MFP6	智能卡 0 卡检测管脚.
		UART0_nRTS	O	MFP7	UART0 请求发送输出脚
		UART5_RXD	I	MFP8	UART5 数据接收输入管脚
		I^2C0_SDA	I/O	MFP9	I2C0 数据输入/输出管脚
		CAN0_RXD	I	MFP10	CAN0 总线接收输入

64 管脚	128 管脚	管脚名称	类型	MFP	描述
		BPWM0_CH4	I/O	MFP12	BPWM0 通道4输出/捕获输入脚
		EPWM0_CH1	I/O	MFP13	EPWM0 通道1输出/捕获输入脚
		QEI0_A	I	MFP14	正交编码器0 相位A输入
27	57	PA.3	I/O	MFP0	通用数字I/O脚
		SPIM_SS	I/O	MFP2	SPIM 从机选择脚
		QSPI0_SS	I/O	MFP3	QSPI0 从机选择脚
		SPI0_SS	I/O	MFP4	SPI0 从机选择脚
		SD1_DAT3	I/O	MFP5	SD/SDIO1 数据线位3.
		SC0_PWR	O	MFP6	智能卡0电源脚
		UART4_TXD	O	MFP7	UART4 数据发送输出脚.
		UART1_TXD	O	MFP8	UART1 数据发送输出脚
		I^2C1_SCL	I/O	MFP9	I^2C1 时钟脚
		BPWM0_CH3	I/O	MFP12	BPWM0 通道3输出/捕获输入脚
		EPWM0_CH2	I/O	MFP13	EPWM0 通道2输出/捕获输入脚
		QEI0_B	I	MFP14	正交编码器0 相位B 输入
28	58	PA.2	I/O	MFP0	通用数字I/O脚
		SPIM_CLK	I/O	MFP2	SPIM 串行时钟脚
		QSPI0_CLK	I/O	MFP3	QSPI0 串行时钟脚
		SPI0_CLK	I/O	MFP4	SPI0 串行时钟脚
		SD1_DAT2	I/O	MFP5	SD/SDIO1 数据线位2.
		SC0_RST	O	MFP6	智能卡0复位脚
		UART4_RXD	I	MFP7	UART4 数据接收输入脚
		UART1_RXD	I	MFP8	UART1数据接收输入管脚
		I^2C1_SDA	I/O	MFP9	I^2C1 数据输入/输出脚
		BPWM0_CH2	I/O	MFP12	BPWM0 通道2 输出/捕获输入
		EPWM0_CH3	I/O	MFP13	EPWM0 通道3输出/捕获输入脚
29	59	PA.1	I/O	MFP0	通用数字I/O脚
		SPIM_MISO	I/O	MFP2	SPIM MISO（主机输入，从机输出）脚.
		QSPI0_MISO0	I/O	MFP3	QSPI0 MISO0（主机输入，从机输出）脚.
		SPI0_MISO	I/O	MFP4	SPI0 MISO（主机输入，从机输出）脚.
		SD1_DAT1	I/O	MFP5	SD/SDIO1 数据线位1.

64 管脚	128 管脚	管脚名称	类型	MFP	描述
		SC0_DAT	I/O	MFP6	智能卡0数据输入脚
		UART0_TXD	O	MFP7	UART0 数据发送输出脚.
		UART1_nCTS	I	MFP8	UART1 禁止发送输入管脚
		I^2C2_SCL	I/O	MFP9	I^2C2 时钟脚
		BPWM0_CH1	I/O	MFP12	BPWM0 通道1 输出/捕获输入
		EPWM0_CH4	I/O	MFP13	EPWM0 通道4输出/捕获输入脚
		DAC1_ST	I	MFP15	DAC1 外部触发输入
30	60	PA.0	I/O	MFP0	通用数字I/O脚
		SPIM_MOSI	I/O	MFP2	SPIM MOSI (主机输出，从机输入) 管脚.
		QSPI0_MOSI0	I/O	MFP3	QSPI0 MOSI0 (主机输出，从机输入) 管脚.
		SPI0_MOSI	I/O	MFP4	SPI0 MOSI (主机输出，从机输入) 管脚.
		SD1_DAT0	I/O	MFP5	SD/SDIO1 数据线位0.
		SC0_CLK	O	MFP6	智能卡0时钟管脚
		UART0_RXD	I	MFP7	UART0 数据接收输入脚
		UART1_nRTS	O	MFP8	UART1 请求发送输出脚
		I^2C2_SDA	I/O	MFP9	I^2C2 数据输入/输出脚
		BPWM0_CH0	I/O	MFP12	BPWM0 通道0 输出/捕获输入
		EPWM0_CH5	I/O	MFP13	EPWM0 通道5 输出/捕获输入
		DAC0_ST	I	MFP15	DAC0 外部触发输入
31	61	V_{DDIO}	P	MFP0	PE.1, PE.8~PE.13 的电源
	62	PE.14	I/O	MFP0	通用数字I/O脚
		EBI_AD8	I/O	MFP2	EBI地址/数据总线位8.
		UART2_TXD	O	MFP3	UART2 数据发送输出脚.
		CAN0_TXD	O	MFP4	CAN0总线发送输出
		SD1_nCD	I	MFP5	SD/SDIO1 卡检测输入管脚
	63	PE.15	I/O	MFP0	通用数字I/O脚
		EBI_AD9	I/O	MFP2	EBI地址/数据总线位9.
		UART2_RXD	I	MFP3	UART2 数据接收输入脚
		CAN0_RXD	I	MFP4	CAN0 总线接收输入
32	64	nRESET	I	MFP0	外部复位输入:低电平有效,有内部上拉，设置此管脚为低将复位初始状态

64 管脚	128 管脚	管脚名称	类型	MFP	描述
33	65	PF.0	I/O	MFP0	通用数字I/O脚
		UART1_TXD	O	MFP2	UART1 数据发送输出脚
		I²C1_SCL	I/O	MFP3	I²C1 时钟脚
		BPWM1_CH0	I/O	MFP12	BPWM1 通道0 输出/捕获输入
		ICE_DAT	O	MFP14	串行调试总线数据管脚.
34	66	PF.1	I/O	MFP0	通用数字I/O脚
		UART1_RXD	I	MFP2	UART1数据接收输入管脚
		I²C1_SDA	I/O	MFP3	I²C1 数据输入/输出脚
		BPWM1_CH1	I/O	MFP12	BPWM1 通道1 输出/捕获输入
		ICE_CLK	I	MFP14	串行调试总线时钟脚
	67	PD.9	I/O	MFP0	通用数字I/O脚
		EBI_AD7	I/O	MFP2	EBI地址/数据总线位7.
		I²C2_SCL	I/O	MFP3	I²C2 时钟脚
		UART2_nCTS	I	MFP4	UART2 禁止发送输入管脚
	68	PD.8	I/O	MFP0	通用数字I/O脚
		EBI_AD6	I/O	MFP2	EBI地址/数据总线位6.
		I²C2_SDA	I/O	MFP3	I²C2 数据输入/输出脚
		UART2_nRTS	O	MFP4	UART2 请求发送输出脚
35	69	PC.5	I/O	MFP0	通用数字I/O脚
		EBI_AD5	I/O	MFP2	EBI地址/数据总线位5.
		SPIM_D2	I/O	MFP3	SPIM QuadI/O模式下数据2管脚
		QSPI0_MISO1	I/O	MFP4	QSPI0 MISO1（主机输入，从机输出）脚.
		UART2_TXD	O	MFP8	UART2 数据发送输出脚.
		I²C1_SCL	I/O	MFP9	I²C1 时钟脚
		CAN0_TXD	O	MFP10	CAN0总线发送输出
		UART4_TXD	O	MFP11	UART4 数据发送输出脚.
		EPWM1_CH0	I/O	MFP12	EPWM1 通道0 输出/捕获输入
36	70	PC.4	I/O	MFP0	通用数字I/O脚
		EBI_AD4	I/O	MFP2	EBI地址/数据总线位4.
		SPIM_D3	I/O	MFP3	SPIM QuadI/O模式下数据3管脚
		QSPI0_MOSI1	I/O	MFP4	QSPI0 MOSI1 (主机输出，从机输入) 管脚.

64 管脚	128 管脚	管脚名称	类型	MFP	描述
		SC1_nCD	I	MFP5	智能卡 1 卡检测管脚.
		I²S0_BCLK	O	MFP6	I2S0位时钟输出管脚
		SPI1_I2SMCLK	I/O	MFP7	SPI1 I²S 主机时钟输出脚
		UART2_RXD	I	MFP8	UART2 数据接收输入脚
		I²C1_SDA	I/O	MFP9	I²C1 数据输入/输出脚
		CAN0_RXD	I	MFP10	CAN0 总线接收输入
		UART4_RXD	I	MFP11	UART4 数据接收输入脚
		EPWM1_CH1	I/O	MFP12	EPWM1 通道1 输出/捕获输入
37	71	PC.3	I/O	MFP0	通用数字I/O脚
		EBI_AD3	I/O	MFP2	EBI地址/数据总线位3.
		SPIM_SS	I/O	MFP3	SPIM 从机选择脚
		QSPI0_SS	I/O	MFP4	QSPI0 从机选择脚
		SC1_PWR	O	MFP5	智能卡 1电源脚
		I²S0_MCLK	O	MFP6	I2S0 主时钟输出脚
		SPI1_MISO	I/O	MFP7	SPI1 MISO(主机输入，从机输出)管脚
		UART2_nRTS	O	MFP8	UART2 请求发送输出脚
		I²C0_SMBAL	O	MFP9	I²C0 SMBus SMBALTER 管脚
		CAN1_TXD	O	MFP10	CAN1总线发送输出
		UART3_TXD	O	MFP11	UART3 数据发送输出脚.
		EPWM1_CH2	I/O	MFP12	EPWM1 通道2 输出/捕获输入
38	72	PC.2	I/O	MFP0	通用数字I/O脚
		EBI_AD2	I/O	MFP2	EBI地址/数据总线位2.
		SPIM_CLK	I/O	MFP3	SPIM 串行时钟脚
		QSPI0_CLK	I/O	MFP4	QSPI0 串行时钟脚
		SC1_RST	O	MFP5	智能卡 1 复位脚
		I²S0_DI	I	MFP6	I2S0 数据输入脚
		SPI1_MOSI	I/O	MFP7	SPI1 MOSI (主机输出，从机输入) 管脚
		UART2_nCTS	I	MFP8	UART2 禁止发送输入管脚
		I²C0_SMBSUS	O	MFP9	I²C0 SMBus SMBSUS 管脚 (PMBus 控制管脚)
		CAN1_RXD	I	MFP10	CAN1 总线接收输入
		UART3_RXD	I	MFP11	UART3 数据接收输入脚

64 管脚	128 管脚	管脚名称	类型	MFP	描述
		EPWM1_CH3	I/O	MFP12	EPWM1 通道3输出/捕获输入脚
39	73	PC.1	I/O	MFP0	通用数字I/O脚
		EBI_AD1	I/O	MFP2	EBI地址/数据总线位1.
		SPIM_MISO	I/O	MFP3	SPIM MISO（主机输入，从机输出）脚.
		QSPI0_MISO0	I/O	MFP4	QSPI0 MISO0（主机输入，从机输出）脚.
		SC1_DAT	I/O	MFP5	智能卡 1 数据管脚.
		I^2S0_DO	O	MFP6	I^2S0 数据输出脚
		SPI1_CLK	I/O	MFP7	SPI1 串行时钟管脚
		UART2_TXD	O	MFP8	UART2 数据发送输出脚.
		I^2C0_SCL	I/O	MFP9	I2C0 时钟管脚
		EPWM1_CH4	I/O	MFP12	EPWM1 通道4输出/捕获输入脚
		ACMP0_O	O	MFP14	模拟比较器 0输出脚
40	74	PC.0	I/O	MFP0	通用数字I/O脚
		EBI_AD0	I/O	MFP2	EBI地址/数据总线位0.
		SPIM_MOSI	I/O	MFP3	SPIM MOSI (主机输出，从机输入) 管脚.
		QSPI0_MOSI0	I/O	MFP4	QSPI0 MOSI0 (主机输出，从机输入) 管脚.
		SC1_CLK	O	MFP5	智能卡 1 时钟脚
		I^2S0_LRCK	O	MFP6	I^2S0左右通道时钟输出脚.
		SPI1_SS	I/O	MFP7	SPI1 从机选择脚
		UART2_RXD	I	MFP8	UART2 数据接收输入脚
		I^2C0_SDA	I/O	MFP9	I2C0 数据输入/输出管脚
		EPWM1_CH5	I/O	MFP12	EPWM1 通道5输出/捕获输入
		ACMP1_O	O	MFP14	模拟比较器1输出脚
	75	V_{SS}	P	MFP0	数字电路地管脚
	76	V_{DD}	P	MFP0	I/O和内部PLL及数字电路的LDO电源
	77	PG.9	I/O	MFP0	通用数字I/O脚
		EBI_AD0	I/O	MFP2	EBI地址/数据总线位0.
		SD1_DAT3	I/O	MFP3	SD/SDIO1 数据线位3.
		SPIM_D2	I/O	MFP4	SPIM QuadI/O模式下数据2管脚
		BPWM0_CH5	I/O	MFP12	BPWM0 通道5 输出/捕获输入
	78	PG.10	I/O	MFP0	通用数字I/O脚

64 管脚	128 管脚	管脚名称	类型	MFP	描述
		EBI_AD1	I/O	MFP2	EBI地址/数据总线位1.
		SD1_DAT2	I/O	MFP3	SD/SDIO1 数据线位2.
		SPIM_D3	I/O	MFP4	SPIM QuadI/O模式下数据3管脚
		BPWM0_CH4	I/O	MFP12	BPWM0 通道4输出/捕获输入脚
	79	PG.11	I/O	MFP0	通用数字I/O脚
		EBI_AD2	I/O	MFP2	EBI地址/数据总线位2.
		SD1_DAT1	I/O	MFP3	SD/SDIO1 数据线位1.
		SPIM_SS	I/O	MFP4	SPIM 从机选择脚
		BPWM0_CH3	I/O	MFP12	BPWM0 通道3输出/捕获输入脚
	80	PG.12	I/O	MFP0	通用数字I/O脚
		EBI_AD3	I/O	MFP2	EBI地址/数据总线位3.
		SD1_DAT0	I/O	MFP3	SD/SDIO1 数据线位0.
		SPIM_CLK	I/O	MFP4	SPIM 串行时钟脚
		BPWM0_CH2	I/O	MFP12	BPWM0 通道2 输出/捕获输入
	81	PG.13	I/O	MFP0	通用数字I/O脚
		EBI_AD4	I/O	MFP2	EBI地址/数据总线位4.
		SD1_CMD	I/O	MFP3	SD/SDIO1 命令/响应脚
		SPIM_MISO	I/O	MFP4	SPIM MISO（主机输入，从机输出）脚.
		BPWM0_CH1	I/O	MFP12	BPWM0 通道1 输出/捕获输入
	82	PG.14	I/O	MFP0	通用数字I/O脚
		EBI_AD5	I/O	MFP2	EBI地址/数据总线位5.
		SD1_CLK	O	MFP3	SD/SDIO1 时钟输出管脚
		SPIM_MOSI	I/O	MFP4	SPIM MOSI (主机输出，从机输入) 管脚.
		BPWM0_CH0	I/O	MFP12	BPWM0 通道0 输出/捕获输入
	83	PG.15	I/O	MFP0	通用数字I/O脚
		SD1_nCD	I	MFP3	SD/SDIO1 卡检测输入管脚
		CLKO	O	MFP14	时钟输出
		EADC0_ST	I	MFP15	EADC0 外部触发输入
	84	PD.13	I/O	MFP0	通用数字I/O脚
		EBI_AD10	I/O	MFP2	EBI地址/数据总线位10.
		SD0_nCD	I	MFP3	SD/SDIO0 卡检测输入管脚

64 管脚	128 管脚	管脚名称	类型	MFP	描述
		SPI0_I2SMCLK	I/O	MFP4	SPI0 I²S 主机时钟输出脚
		SPI1_I2SMCLK	I/O	MFP5	SPI1 I²S 主机时钟输出脚
		SC2_nCD	I	MFP7	智能卡 2 卡检测管脚.
	85	PA.12	I/O	MFP0	通用数字I/O脚
		I²S0_BCLK	O	MFP2	I2S0位时钟输出管脚
		UART4_TXD	O	MFP3	UART4 数据发送输出脚.
		I²C1_SCL	I/O	MFP4	I²C1 时钟脚
		SPI2_SS	I/O	MFP5	SPI2 从机选择脚
		CAN0_TXD	O	MFP6	CAN0总线发送输出
		SC2_PWR	O	MFP7	智能卡 2电源脚
		BPWM1_CH2	I/O	MFP11	BPWM1 通道2 输出/捕获输入
		QEI1_INDEX	I	MFP12	正交编码器1 索引输入
		USB_VBUS	P	MFP14	USB主机或HUB的电源
	86	PA.13	I/O	MFP0	通用数字I/O脚
		I²S0_MCLK	O	MFP2	I2S0 主时钟输出脚
		UART4_RXD	I	MFP3	UART4 数据接收输入脚
		I²C1_SDA	I/O	MFP4	I²C1 数据输入/输出脚
		SPI2_CLK	I/O	MFP5	SPI2 串行时钟脚
		CAN0_RXD	I	MFP6	CAN0 总线接收输入
		SC2_RST	O	MFP7	智能卡 2 复位脚
		BPWM1_CH3	I/O	MFP11	BPWM1 通道3输出/捕获输入脚
		QEI1_A	I	MFP12	正交编码器1 相位A输入
		USB_D-	A	MFP14	USB差分信号D+.
	87	PA.14	I/O	MFP0	通用数字I/O脚
		I²S0_DI	I	MFP2	I2S0 数据输入脚
		UART0_TXD	O	MFP3	UART0 数据发送输出脚.
		SPI2_MISO	I/O	MFP5	SPI2 MISO（主机输入，从机输出）脚.
		I²C2_SCL	I/O	MFP6	I²C2 时钟脚
		SC2_DAT	I/O	MFP7	智能卡 2 数据管脚.
		BPWM1_CH4	I/O	MFP11	BPWM1 通道4输出/捕获输入脚
		QEI1_B	I	MFP12	正交编码器1 相位B 输入

64 管脚	128 管脚	管脚名称	类型	MFP	描述
		USB_D+	A	MFP14	USB差分信号D-.
	88	PA.15	I/O	MFP0	通用数字I/O脚
		I^2S0_DO	O	MFP2	I^2S0 数据输出脚
		UART0_RXD	I	MFP3	UART0 数据接收输入脚
		SPI2_MOSI	I/O	MFP5	SPI2 MOSI (主机输出，从机输入) 管脚.
		I^2C2_SDA	I/O	MFP6	I^2C2 数据输入/输出脚
		SC2_CLK	O	MFP7	智能卡 2 时钟脚
		BPWM1_CH5	I/O	MFP11	EPWM1 通道5输出/捕获输入
		EPWM0_SYNC_IN	I	MFP12	EPWM0 计数器同步触发输入脚
		USB_OTG_ID	I	MFP14	USB_ 标识
41	89	HSUSB_VRES	A	MFP0	HSUSB模块参考电阻
42	90	$HSUSB_V_{DD33}$	P	MFP0	HSUSB VDD33 电源供电
43	91	HSUSB_VBUS	P	MFP0	HSUSB USB主机或HUB的电源
44	92	HSUSB_D-	A	MFP0	HSUSB差分信号D+.
45	93	HSUSB_VSS	P	MFP0	HSUSB 地管脚
46	94	HSUSB_D+	A	MFP0	HSUSB差分信号D-.
47	95	$HSUSB_V_{DD12}$_CAP	A	MFP0	HSUSB内部1.2v电压输出去耦管脚. 注: 此管脚需接1uF电容到地.
48	96	HSUSB_ID	I	MFP0	HSUSB 标识
	97	PE.7	I/O	MFP0	通用数字I/O脚
		SD0_CMD	I/O	MFP3	SD/SDIO0 命令/响应脚
		SPIM_D2	I/O	MFP4	SPIM QuadI/O模式下数据2管脚
		UART5_TXD	O	MFP8	UART5数据发送输出脚
		CAN1_TXD	O	MFP9	CAN1总线发送输出
		QEI1_INDEX	I	MFP11	正交编码器1 索引输入
		EPWM0_CH0	I/O	MFP12	EPWM0 通道0输出/捕获输入
		BPWM0_CH5	I/O	MFP13	BPWM0 通道5 输出/捕获输入
	98	PE.6	I/O	MFP0	通用数字I/O脚
		SD0_CLK	O	MFP3	SD/SDIO0 时钟输出脚
		SPIM_D3	I/O	MFP4	SPIM QuadI/O模式下数据3管脚
		SPI3_I2SMCLK	I/O	MFP5	SPI3 I^2S 主机时钟输出脚

64 管脚	128 管脚	管脚名称	类型	MFP	描述
		SC0_nCD	I	MFP6	智能卡 0 卡检测管脚.
		USCI0_CTL0	I/O	MFP7	USCI0 控制0 管脚.
		UART5_RXD	I	MFP8	UART5 数据接收输入管脚
		CAN1_RXD	I	MFP9	CAN1 总线接收输入
		QEI1_A	I	MFP11	正交编码器1 相位A输入
		EPWM0_CH1	I/O	MFP12	EPWM0 通道1输出/捕获输入脚
		BPWM0_CH4	I/O	MFP13	BPWM0 通道4输出/捕获输入脚
	99	PE.5	I/O	MFP0	通用数字I/O脚
		EBI_nRD	O	MFP2	EBI 读使能输出脚
		SD0_DAT3	I/O	MFP3	SD/SDIO0数据线位3
		SPIM_SS	I/O	MFP4	SPIM 从机选择脚
		SPI3_SS	I/O	MFP5	SPI3 从机选择脚
		SC0_PWR	O	MFP6	智能卡0电源脚
		USCI0_CTL1	I/O	MFP7	USCI0 控制1 管脚.
		QEI1_B	I	MFP11	正交编码器1 相位B 输入
		EPWM0_CH2	I/O	MFP12	EPWM0 通道2输出/捕获输入脚
		BPWM0_CH3	I/O	MFP13	BPWM0 通道3输出/捕获输入脚
	100	PE.4	I/O	MFP0	通用数字I/O脚
		EBI_nWR	O	MFP2	EBI 写使能输出脚
		SD0_DAT2	I/O	MFP3	SD/SDIO0 数据线位2
		SPIM_CLK	I/O	MFP4	SPIM 串行时钟脚
		SPI3_CLK	I/O	MFP5	SPI3 串行时钟脚
		SC0_RST	O	MFP6	智能卡0复位脚
		USCI0_DAT1	I/O	MFP7	USCI0 数据1 管脚.
		QEI0_INDEX	I	MFP11	正交编码器0 索引输入
		EPWM0_CH3	I/O	MFP12	EPWM0 通道3输出/捕获输入脚
		BPWM0_CH2	I/O	MFP13	BPWM0 通道2 输出/捕获输入
	101	PE.3	I/O	MFP0	通用数字I/O脚
		EBI_MCLK	O	MFP2	EBI 外部时钟输出脚.
		SD0_DAT1	I/O	MFP3	SD/SDIO0 数据线位1
		SPIM_MISO	I/O	MFP4	SPIM MISO（主机输入，从机输出）脚.

64 管脚	128 管脚	管脚名称	类型	MFP	描述
		SPI3_MISO	I/O	MFP5	SPI3 MISO（主机输入，从机输出）脚.
		SC0_DAT	I/O	MFP6	智能卡0数据输入脚
		USCI0_DAT0	I/O	MFP7	USCI0 数据0脚
		QEI0_A	I	MFP11	正交编码器0 相位A输入
		EPWM0_CH4	I/O	MFP12	EPWM0 通道4输出/捕获输入脚
		BPWM0_CH1	I/O	MFP13	BPWM0 通道1 输出/捕获输入
	102	PE.2	I/O	MFP0	通用数字I/O脚
		EBI_ALE	O	MFP2	EBI地址锁存使能输出脚
		SD0_DAT0	I/O	MFP3	SD/SDIO0 数据线位0
		SPIM_MOSI	I/O	MFP4	SPIM MOSI (主机输出，从机输入) 管脚.
		SPI3_MOSI	I/O	MFP5	SPI3 MOSI (主机输出，从机输入) 管脚.
		SC0_CLK	O	MFP6	智能卡0时钟管脚
		USCI0_CLK	I/O	MFP7	USCI0 时钟脚
		QEI0_B	I	MFP11	正交编码器0 相位B 输入
		EPWM0_CH5	I/O	MFP12	EPWM0 通道5 输出/捕获输入
		BPWM0_CH0	I/O	MFP13	BPWM0 通道0 输出/捕获输入
	103	V_{SS}	P	MFP0	数字电路地管脚
	104	V_{DD}	P	MFP0	I/O和内部PLL及数字电路的LDO电源
	105	PE.1	I/O	MFP0	通用数字I/O脚
		EBI_AD10	I/O	MFP2	EBI地址/数据总线位10.
		QSPI0_MISO0	I/O	MFP3	QSPI0 MISO0（主机输入，从机输出）脚.
		SC2_DAT	I/O	MFP4	智能卡 2 数据管脚.
		I^2S0_BCLK	O	MFP5	I2S0位时钟输出管脚
		SPI1_MISO	I/O	MFP6	SPI1 MISO(主机输入，从机输出)管脚
		UART3_TXD	O	MFP7	UART3 数据发送输出脚.
		I^2C1_SCL	I/O	MFP8	I^2C1 时钟脚
		UART4_nCTS	I	MFP9	UART4 禁止发送输入管脚
	106	PE.0	I/O	MFP0	通用数字I/O脚
		EBI_AD11	I/O	MFP2	EBI地址/数据总线位11.
		QSPI0_MOSI0	I/O	MFP3	QSPI0 MOSI0 (主机输出，从机输入) 管脚.
		SC2_CLK	O	MFP4	智能卡 2 时钟脚

64 管脚	128 管脚	管脚名称	类型	MFP	描述
		I²S0_MCLK	O	MFP5	I2S0 主时钟输出脚
		SPI1_MOSI	I/O	MFP6	SPI1 MOSI (主机输出，从机输入) 管脚
		UART3_RXD	I	MFP7	UART3 数据接收输入脚
		I²C1_SDA	I/O	MFP8	I²C1 数据输入/输出脚
		UART4_nRTS	O	MFP9	UART4 请求发送输出脚
	107	PH.8	I/O	MFP0	通用数字I/O脚
		EBI_AD12	I/O	MFP2	EBI地址/数据总线位12.
		QSPI0_CLK	I/O	MFP3	QSPI0 串行时钟脚
		SC2_PWR	O	MFP4	智能卡 2电源脚
		I²S0_DI	I	MFP5	I2S0 数据输入脚
		SPI1_CLK	I/O	MFP6	SPI1 串行时钟管脚
		UART3_nRTS	O	MFP7	UART3 请求发送输出脚
		I²C1_SMBAL	O	MFP8	I²C1 SMBus SMBALTER 管脚
		I²C2_SCL	I/O	MFP9	I²C2 时钟脚
		UART1_TXD	O	MFP10	UART1 数据发送输出脚
	108	PH.9	I/O	MFP0	通用数字I/O脚
		EBI_AD13	I/O	MFP2	EBI地址/数据总线位13.
		QSPI0_SS	I/O	MFP3	QSPI0 从机选择脚
		SC2_RST	O	MFP4	智能卡 2 复位脚
		I²S0_DO	O	MFP5	I²S0 数据输出脚
		SPI1_SS	I/O	MFP6	SPI1 从机选择脚
		UART3_nCTS	I	MFP7	UART3 禁止发送输入管脚
		I²C1_SMBSUS	O	MFP8	I²C1 SMBus SMBSUS 管脚 (PMBus 控制管脚)
		I²C2_SDA	I/O	MFP9	I²C2 数据输入/输出脚
		UART1_RXD	I	MFP10	UART1数据接收输入管脚
	109	PH.10	I/O	MFP0	通用数字I/O脚
		EBI_AD14	I/O	MFP2	EBI地址/数据总线位14.
		QSPI0_MISO1	I/O	MFP3	QSPI0 MISO1（主机输入，从机输出）脚.
		SC2_nCD	I	MFP4	智能卡 2 卡检测管脚.
		I²S0_LRCK	O	MFP5	I²S0左右通道时钟输出脚.
		SPI1_I2SMCLK	I/O	MFP6	SPI1 I²S 主机时钟输出脚

64 管脚	128 管脚	管脚名称	类型	MFP	描述
		UART4_TXD	O	MFP7	UART4 数据发送输出脚.
		UART0_TXD	O	MFP8	UART0 数据发送输出脚.
	110	PH.11	I/O	MFP0	通用数字I/O脚
		EBI_AD15	I/O	MFP2	EBI地址/数据总线位15.
		QSPI0_MOSI1	I/O	MFP3	QSPI0 MOSI1 (主机输出，从机输入) 管脚.
		UART4_RXD	I	MFP7	UART4 数据接收输入脚
		UART0_RXD	I	MFP8	UART0 数据接收输入脚
		EPWM0_CH5	I/O	MFP11	EPWM0 通道5 输出/捕获输入
	111	PD.14	I/O	MFP0	通用数字I/O脚
		EBI_nCS0	O	MFP2	EBI片选0输出脚
		SPI3_I2SMCLK	I/O	MFP3	SPI3 I^2S 主机时钟输出脚
		SC1_nCD	I	MFP4	智能卡 1 卡检测管脚.
		EPWM0_CH4	I/O	MFP11	EPWM0 通道4输出/捕获输入脚
49	112	V_{SS}	P	MFP0	数字电路地管脚
50	113	LDO_CAP	A	MFP0	LDO输出脚
51	114	V_{DD}	P	MFP0	I/O和内部PLL及数字电路的LDO电源
52	115	PC.14	I/O	MFP0	通用数字I/O脚
		EBI_AD11	I/O	MFP2	EBI地址/数据总线位11.
		SC1_nCD	I	MFP3	智能卡 1 卡检测管脚.
		SPI0_I2SMCLK	I/O	MFP4	SPI0 I^2S 主机时钟输出脚
		USCI0_CTL0	I/O	MFP5	USCI0 控制0 管脚.
		QSPI0_CLK	I/O	MFP6	QSPI0 串行时钟脚
		EPWM0_SYNC_IN	I	MFP11	EPWM0 计数器同步触发输入脚
		ETM_TRACE_CLK	I	MFP12	ETM接收追踪时钟输入管脚
		TM1	I/O	MFP13	Timer1 事件计数输入/反转输出脚
		USB_VBUS_ST	I	MFP14	USB 外部VBUS 电源状态管脚.
		HSUSB_VBUS_ST	I	MFP15	HSUSB 外部VBUS 电源状态管脚.
53	116	PB.15	I/O	MFP0	通用数字I/O脚
		EADC0_CH15	A	MFP1	EADC0 通道15 模拟输入
		EBI_AD12	I/O	MFP2	EBI地址/数据总线位12.
		SC1_PWR	O	MFP3	智能卡 1电源脚

64 管脚	128 管脚	管脚名称	类型	MFP	描述
		SPI0_SS	I/O	MFP4	SPI0 从机选择脚
		USCI0_CTL1	I/O	MFP5	USCI0 控制1 管脚.
		UART0_nCTS	I	MFP6	UART0 禁止发送输入管脚
		UART3_TXD	O	MFP7	UART3 数据发送输出脚.
		I^2C2_SMBAL	O	MFP8	I^2C2 SMBus SMBALTER 管脚
		EPWM1_CH0	I/O	MFP11	EPWM1 通道0 输出/捕获输入
		ETM_TRACE_DATA0	I	MFP12	ETM接收追踪数据0 输入管脚
		TM0_EXT	I/O	MFP13	Timer0 事件计数输入/反转输出管脚
		USB_VBUS_EN	O	MFP14	USB 外部 VBUS 电源使能管脚.
		HSUSB_VBUS_EN	O	MFP15	HSUSB 外部 VBUS 电源使能管脚.
54	117	PB.14	I/O	MFP0	通用数字I/O脚
		EADC0_CH14	A	MFP1	EADC0 通道14 模拟输入
		EBI_AD13	I/O	MFP2	EBI地址/数据总线位13.
		SC1_RST	O	MFP3	智能卡 1 复位脚
		SPI0_CLK	I/O	MFP4	SPI0 串行时钟脚
		USCI0_DAT1	I/O	MFP5	USCI0 数据1 管脚.
		UART0_nRTS	O	MFP6	UART0 请求发送输出脚
		UART3_RXD	I	MFP7	UART3 数据接收输入脚
		I^2C2_SMBSUS	O	MFP8	I^2C2 SMBus SMBSUS 管脚 (PMBus 控制管脚)
		EPWM1_CH1	I/O	MFP11	EPWM1 通道1 输出/捕获输入
		ETM_TRACE_DATA1	I	MFP12	ETM接收追踪数据1 输入管脚
		TM1_EXT	I/O	MFP13	Timer1 事件计数输入/反转输出脚
		CLKO	O	MFP14	时钟输出
55	118	PB.13	I/O	MFP0	通用数字I/O脚
		EADC0_CH13	A	MFP1	EADC0 通道13 模拟输入
		DAC1_OUT	A	MFP1	DAC1 模拟输出通道
		ACMP0_P3	A	MFP1	模拟比较器 0 正极输入 3 管脚.
		ACMP1_P3	A	MFP1	模拟比较器 1 正极输入 3 管脚.
		EBI_AD14	I/O	MFP2	EBI地址/数据总线位14.
		SC1_DAT	I/O	MFP3	智能卡 1 数据管脚.
		SPI0_MISO	I/O	MFP4	SPI0 MISO（主机输入，从机输出）脚.

64 管脚	128 管脚	管脚名称	类型	MFP	描述
		USCI0_DAT0	I/O	MFP5	USCI0 数据0脚
		UART0_TXD	O	MFP6	UART0 数据发送输出脚.
		UART3_nRTS	O	MFP7	UART3 请求发送输出脚
		I^2C2_SCL	I/O	MFP8	I^2C2 时钟脚
		EPWM1_CH2	I/O	MFP11	EPWM1 通道2 输出/捕获输入
		ETM_TRACE_DATA2	I	MFP12	ETM接收追踪数据2 输入管脚
		TM2_EXT	I/O	MFP13	Timer2 事件计数输入/反转输出脚
56	119	PB.12	I/O	MFP0	通用数字I/O脚
		EADC0_CH12	A	MFP1	EADC0 通道12 模拟输入
		DAC0_OUT	A	MFP1	DAC0 模拟输出通道
		ACMP0_P2	A	MFP1	模拟比较器 0 正极输入 2 管脚.
		ACMP1_P2	A	MFP1	模拟比较器 1 正极输入 2 管脚.
		EBI_AD15	I/O	MFP2	EBI地址/数据总线位15.
		SC1_CLK	O	MFP3	智能卡 1 时钟脚
		SPI0_MOSI	I/O	MFP4	SPI0 MOSI (主机输出，从机输入) 管脚.
		USCI0_CLK	I/O	MFP5	USCI0 时钟脚
		UART0_RXD	I	MFP6	UART0 数据接收输入脚
		UART3_nCTS	I	MFP7	UART3 禁止发送输入管脚
		I^2C2_SDA	I/O	MFP8	I^2C2 数据输入/输出脚
		SD0_nCD	I	MFP9	SD/SDIO0 卡检测输入管脚
		EPWM1_CH3	I/O	MFP11	EPWM1 通道3输出/捕获输入脚
		ETM_TRACE_DATA3	I	MFP12	ETM接收追踪数据3 输入管脚
		TM3_EXT	I/O	MFP13	Timer3 事件计数输入/反转输出脚
57	120	AV_{DD}	P	MFP0	内部模拟电路电源
58	121	V_{REF}	A	MFP0	ADC参考电压输入. 注: 此管脚需接1uF电容到地.
59	122	AV_{SS}	P	MFP0	模拟电路地管脚
60	123	PB.11	I/O	MFP0	通用数字I/O脚
		EADC0_CH11	A	MFP1	EADC0 通道11 模拟输入
		EBI_ADR16	O	MFP2	EBI 地址总线位 16.
		UART0_nCTS	I	MFP5	UART0 禁止发送输入管脚

64 管脚	128 管脚	管脚名称	类型	MFP	描述
		UART4_TXD	O	MFP6	UART4 数据发送输出脚.
		I²C1_SCL	I/O	MFP7	I²C1 时钟脚
		CAN0_TXD	O	MFP8	CAN0总线发送输出
		SPI0_I2SMCLK	I/O	MFP9	SPI0 I²S 主机时钟输出脚
		BPWM1_CH0	I/O	MFP10	BPWM1 通道0 输出/捕获输入
		SPI3_CLK	I/O	MFP11	SPI3 串行时钟脚
		HSUSB_VBUS_ST	I	MFP14	HSUSB 外部VBUS 电源状态管脚.
61	124	PB.10	I/O	MFP0	通用数字I/O脚
		EADC0_CH10	A	MFP1	EADC0 通道10 模拟输入
		EBI_ADR17	O	MFP2	EBI 地址总线位 17.
		USCI1_CTL0	I/O	MFP4	USCI1 控制0管脚
		UART0_nRTS	O	MFP5	UART0 请求发送输出脚
		UART4_RXD	I	MFP6	UART4 数据接收输入脚
		I²C1_SDA	I/O	MFP7	I²C1 数据输入/输出脚
		CAN0_RXD	I	MFP8	CAN0 总线接收输入
		BPWM1_CH1	I/O	MFP10	BPWM1 通道1 输出/捕获输入
		SPI3_SS	I/O	MFP11	SPI3 从机选择脚
		HSUSB_VBUS_EN	O	MFP14	HSUSB 外部 VBUS 电源使能管脚.
62	125	PB.9	I/O	MFP0	通用数字I/O脚
		EADC0_CH9	A	MFP1	EADC0 通道9 模拟输入
		EBI_ADR18	O	MFP2	EBI 地址总线位 18.
		USCI1_CTL1	I/O	MFP4	USCI1 控制1管脚
		UART0_TXD	O	MFP5	UART0 数据发送输出脚.
		UART1_nCTS	I	MFP6	UART1 禁止发送输入管脚
		I²C1_SMBAL	O	MFP7	I²C1 SMBus SMBALTER 管脚
		BPWM1_CH2	I/O	MFP10	BPWM1 通道2 输出/捕获输入
		SPI3_MISO	I/O	MFP11	SPI3 MISO（主机输入，从机输出）脚.
		INT7	I	MFP13	外部中断7 输入脚
63	126	PB.8	I/O	MFP0	通用数字I/O脚
		EADC0_CH8	A	MFP1	EADC0 通道8 模拟输入
		EBI_ADR19	O	MFP2	EBI 地址总线位 19.

64 管脚	128 管脚	管脚名称	类型	MFP	描述
		USCI1_CLK	I/O	MFP4	USCI1 时钟脚
		UART0_RXD	I	MFP5	UART0 数据接收输入脚
		UART1_nRTS	O	MFP6	UART1 请求发送输出脚
		I^2C1_SMBSUS	O	MFP7	I^2C1 SMBus SMBSUS 管脚 (PMBus 控制管脚)
		BPWM1_CH3	I/O	MFP10	BPWM1 通道3输出/捕获输入脚
		SPI3_MOSI	I/O	MFP11	SPI3 MOSI (主机输出，从机输入) 管脚.
		INT6	I	MFP13	外部中断6 输入脚
64	127	PB.7	I/O	MFP0	通用数字I/O脚
		EADC0_CH7	A	MFP1	EADC0 通道7 模拟输入
		EBI_nWRL	O	MFP2	EBI低字节写使能输出脚
		USCI1_DAT0	I/O	MFP4	USCI1 数据0脚
		CAN1_TXD	O	MFP5	CAN1总线发送输出
		UART1_TXD	O	MFP6	UART1 数据发送输出脚
		SD1_CMD	I/O	MFP7	SD/SDIO1 命令/响应脚
		EBI_nCS0	O	MFP8	EBI片选0输出脚
		BPWM1_CH4	I/O	MFP10	BPWM1 通道4输出/捕获输入脚
		EPWM1_BRAKE0	I	MFP11	EPWM1 刹车0 输入脚
		EPWM1_CH4	I/O	MFP12	EPWM1 通道4输出/捕获输入脚
		INT5	I	MFP13	外部中断5 输入脚
		USB_VBUS_ST	I	MFP14	USB 外部VBUS 电源状态管脚.
		ACMP0_O	O	MFP15	模拟比较器 0输出脚
1	128	PB.6	I/O	MFP0	通用数字I/O脚
		EADC0_CH6	A	MFP1	EADC0通道6模拟输入
		EBI_nWRH	O	MFP2	EBI 高字节写使能输出管脚
		USCI1_DAT1	I/O	MFP4	USCI1数据1管脚.
		CAN1_RXD	I	MFP5	CAN1 总线接收输入
		UART1_RXD	I	MFP6	UART1数据接收输入管脚
		SD1_CLK	O	MFP7	SD/SDIO1 时钟输出管脚
		EBI_nCS1	O	MFP8	EBI片选1输出管脚
		BPWM1_CH5	I/O	MFP10	EPWM1 通道5输出/捕获输入
		EPWM1_BRAKE1	I	MFP11	EPWM1 刹车1输入管脚

64 管脚	128 管脚	管脚名称	类型	MFP	描述
		EPWM1_CH5	I/O	MFP12	EPWM1 通道5输出/捕获输入
		INT4	I	MFP13	外部中断4输入管脚
		USB_VBUS_EN	O	MFP14	USB 外部 VBUS 电源使能管脚.
		ACMP1_O	O	MFP15	模拟比较器1输出脚

4.2.4 M484系列管脚描述

64 管脚	128 管脚	管脚名称	类型	MFP	描述
2	1	PB.5	I/O	MFP0	通用数字I/O脚
		EADC0_CH5	A	MFP1	EADC0 通道5模拟输入
		ACMP1_N	A	MFP1	模拟比较器1负极输入管脚
		EBI_ADR0	O	MFP2	EBI地址总线位0
		SD0_DAT3	I/O	MFP3	SD/SDIO0数据线位3
		SPI1_MISO	I/O	MFP5	SPI1 MISO(主机输入，从机输出)管脚
		I^2C0_SCL	I/O	MFP6	I2C0 时钟管脚
		UART5_TXD	O	MFP7	UART5数据发送输出脚
		USCI1_CTL0	I/O	MFP8	USCI1 控制0管脚
		SC0_CLK	O	MFP9	智能卡0时钟管脚
		I^2S0_BCLK	O	MFP10	I2S0位时钟输出管脚
		EPWM0_CH0	I/O	MFP11	EPWM0 通道0输出/捕获输入
		TM0	I/O	MFP14	Timer0 事件计数输入/反转输出管脚
		INT0	I	MFP15	外部中断0输入管脚
3	2	PB.4	I/O	MFP0	通用数字I/O脚
		EADC0_CH4	A	MFP1	EADC0通道4模拟输入
		ACMP1_P1	A	MFP1	模拟比较器1正极输入1脚
		EBI_ADR1	O	MFP2	EBI 地址总线位1
		SD0_DAT2	I/O	MFP3	SD/SDIO0 数据线位2
		SPI1_MOSI	I/O	MFP5	SPI1 MOSI (主机输出，从机输入) 管脚
		I^2C0_SDA	I/O	MFP6	I2C0 数据输入/输出管脚
		UART5_RXD	I	MFP7	UART5 数据接收输入管脚
		USCI1_CTL1	I/O	MFP8	USCI1 控制1管脚
		SC0_DAT	I/O	MFP9	智能卡0数据输入脚
		I^2S0_MCLK	O	MFP10	I2S0 主时钟输出脚
		EPWM0_CH1	I/O	MFP11	EPWM0 通道1输出/捕获输入脚
		TM1	I/O	MFP14	Timer1 事件计数输入/反转输出脚
		INT1	I	MFP15	外部中断1输入脚
4	3	PB.3	I/O	MFP0	通用数字I/O脚

64 管脚	128 管脚	管脚名称	类型	MFP	描述
		EADC0_CH3	A	MFP1	EADC0通道3模拟输入
		ACMP0_N	A	MFP1	模拟比较器0负极输入脚
		EBI_ADR2	O	MFP2	EBI 地址总线位2
		SD0_DAT1	I/O	MFP3	SD/SDIO0 数据线位1
		SPI1_CLK	I/O	MFP5	SPI1 串行时钟管脚
		UART1_TXD	O	MFP6	UART1 数据发送输出脚
		UART5_nRTS	O	MFP7	UART5 请求发送输出脚
		USCI1_DAT1	I/O	MFP8	USCI1数据1管脚.
		SC0_RST	O	MFP9	智能卡0复位脚
		I^2S0_DI	I	MFP10	I2S0 数据输入脚
		EPWM0_CH2	I/O	MFP11	EPWM0 通道2输出/捕获输入脚
		TM2	I/O	MFP14	Timer2 事件计数输入/反转输出脚
		INT2	I	MFP15	外部中断2输入脚
5	4	PB.2	I/O	MFP0	通用数字I/O脚
		EADC0_CH2	A	MFP1	EADC0 通道2 模拟输入
		ACMP0_P1	A	MFP1	模拟比较器0正极输入1脚
		OPA0_O	A	MFP1	运放0输出脚
		EBI_ADR3	O	MFP2	EBI 地址总线位3
		SD0_DAT0	I/O	MFP3	SD/SDIO0 数据线位0
		SPI1_SS	I/O	MFP5	SPI1 从机选择脚
		UART1_RXD	I	MFP6	UART1数据接收输入管脚
		UART5_nCTS	I	MFP7	UART5 禁止发送输入管脚
		USCI1_DAT0	I/O	MFP8	USCI1 数据0脚
		SC0_PWR	O	MFP9	智能卡0电源脚
		I^2S0_DO	O	MFP10	I^2S0 数据输出脚
		EPWM0_CH3	I/O	MFP11	EPWM0 通道3输出/捕获输入脚
		TM3	I/O	MFP14	Timer3 事件计数输入/反转输出脚
		INT3	I	MFP15	外部中断3输入脚
	5	PC.12	I/O	MFP0	通用数字I/O脚
		EBI_ADR4	O	MFP2	EBI 地址总线位 4.
		UART0_TXD	O	MFP3	UART0 数据发送输出脚.

64 管脚	128 管脚	管脚名称	类型	MFP	描述
		I^2C0_SCL	I/O	MFP4	I2C0 时钟管脚
		SPI3_MISO	I/O	MFP6	SPI3 MISO（主机输入，从机输出）脚.
		SC0_nCD	I	MFP9	智能卡 0 卡检测管脚.
		ECAP1_IC2	I	MFP11	增强型捕获计数器 1 输入 2 管脚.
		EPWM1_CH0	I/O	MFP12	EPWM1 通道0 输出/捕获输入
		ACMP0_O	O	MFP14	模拟比较器 0输出脚
	6	PC.11	I/O	MFP0	通用数字I/O脚
		EBI_ADR5	O	MFP2	EBI 地址总线位 5.
		UART0_RXD	I	MFP3	UART0 数据接收输入脚
		I^2C0_SDA	I/O	MFP4	I2C0 数据输入/输出管脚
		SPI3_MOSI	I/O	MFP6	SPI3 MOSI (主机输出，从机输入) 管脚.
		ECAP1_IC1	I	MFP11	增强型捕获计数器 1 输入 1 管脚.
		EPWM1_CH1	I/O	MFP12	EPWM1 通道1 输出/捕获输入
		ACMP1_O	O	MFP14	模拟比较器1输出脚
	7	PC.10	I/O	MFP0	通用数字I/O脚
		EBI_ADR6	O	MFP2	EBI 地址总线位 6.
		SPI3_CLK	I/O	MFP6	SPI3 串行时钟脚
		UART3_TXD	O	MFP7	UART3 数据发送输出脚.
		ECAP1_IC0	I	MFP11	增强型捕获计数器 1 输入脚0
		EPWM1_CH2	I/O	MFP12	EPWM1 通道2 输出/捕获输入
	8	PC.9	I/O	MFP0	通用数字I/O脚
		EBI_ADR7	O	MFP2	EBI 地址总线位 7.
		SPI3_SS	I/O	MFP6	SPI3 从机选择脚
		UART3_RXD	I	MFP7	UART3 数据接收输入脚
		EPWM1_CH3	I/O	MFP12	EPWM1 通道3输出/捕获输入脚
6	9	PB.1	I/O	MFP0	通用数字I/O脚
		EADC0_CH1	A	MFP1	EADC0 通道1 模拟输入
		OPA0_N	A	MFP1	运放0负极输入脚
		EBI_ADR8	O	MFP2	EBI 地址总线位 8.
		SD0_CLK	O	MFP3	SD/SDIO0 时钟输出脚
		SPI1_I2SMCLK	I/O	MFP5	SPI1 I^2S 主机时钟输出脚

64 管脚	128 管脚	管脚名称	类型	MFP	描述
		SPI3_I2SMCLK	I/O	MFP6	SPI3 I^2S 主机时钟输出脚
		UART2_TXD	O	MFP7	UART2 数据发送输出脚.
		USCI1_CLK	I/O	MFP8	USCI1 时钟脚
		I^2C1_SCL	I/O	MFP9	I^2C1 时钟脚
		I^2S0_LRCK	O	MFP10	I^2S0左右通道时钟输出脚.
		EPWM0_CH4	I/O	MFP11	EPWM0 通道4输出/捕获输入脚
		EPWM1_CH4	I/O	MFP12	EPWM1 通道4输出/捕获输入脚
		EPWM0_BRAKE0	I	MFP13	EPWM0 刹车0 输入脚
7	10	PB.0	I/O	MFP0	通用数字I/O脚
		EADC0_CH0	A	MFP1	EADC0 通道0 模拟输入
		OPA0_P	A	MFP1	运放 0 正极输入脚
		EBI_ADR9	O	MFP2	EBI 地址总线位 9.
		SD0_CMD	I/O	MFP3	SD/SDIO0 命令/响应脚
		UART2_RXD	I	MFP7	UART2 数据接收输入脚
		SPI0_I2SMCLK	I/O	MFP8	SPI0 I^2S 主机时钟输出脚
		I^2C1_SDA	I/O	MFP9	I^2C1 数据输入/输出脚
		EPWM0_CH5	I/O	MFP11	EPWM0 通道5 输出/捕获输入
		EPWM1_CH5	I/O	MFP12	EPWM1 通道5输出/捕获输入
		EPWM0_BRAKE1	I	MFP13	EPWM0 刹车1 输入脚
	11	V_{SS}	P	MFP0	数字电路地管脚
	12	V_{DD}	P	MFP0	I/O和内部PLL及数字电路的LDO电源
8	13	PA.11	I/O	MFP0	通用数字I/O脚
		ACMP0_P0	A	MFP1	模拟比较器 0 正极输入脚0
		EBI_nRD	O	MFP2	EBI 读使能输出脚
		SC2_PWR	O	MFP3	智能卡 2电源脚
		SPI2_SS	I/O	MFP4	SPI2 从机选择脚
		SD1_DAT3	I/O	MFP5	SD/SDIO1 数据线位3.
		USCI0_CLK	I/O	MFP6	USCI0 时钟脚
		I^2C2_SCL	I/O	MFP7	I^2C2 时钟脚
		BPWM0_CH0	I/O	MFP9	BPWM0 通道0 输出/捕获输入
		EPWM0_SYNC_OUT	O	MFP10	EPWM0计数器同步触发输出脚

64 管脚	128 管脚	管脚名称	类型	MFP	描述
		TM0_EXT	I/O	MFP13	Timer0 事件计数输入/反转输出管脚
		DAC1_ST	I	MFP14	DAC1 外部触发输入
9	14	PA.10	I/O	MFP0	通用数字I/O脚
		ACMP1_P0	A	MFP1	模拟比较器 1 正极输入脚0
		OPA1_O	A	MFP1	运放 1输出脚
		EBI_nWR	O	MFP2	EBI 写使能输出脚
		SC2_RST	O	MFP3	智能卡 2 复位脚
		SPI2_CLK	I/O	MFP4	SPI2 串行时钟脚
		SD1_DAT2	I/O	MFP5	SD/SDIO1 数据线位2.
		USCI0_DAT0	I/O	MFP6	USCI0 数据0脚
		I^2C2_SDA	I/O	MFP7	I^2C2 数据输入/输出脚
		BPWM0_CH1	I/O	MFP9	BPWM0 通道1 输出/捕获输入
		QEI1_INDEX	I	MFP10	正交编码器1 索引输入
		ECAP0_IC0	I	MFP11	增强型捕获计数器 0 输入脚0
		TM1_EXT	I/O	MFP13	Timer1 事件计数输入/反转输出脚
		DAC0_ST	I	MFP14	DAC0 外部触发输入
		SWDH_CLK	O	MFP15	串行调试总线主时钟输出
10	15	PA.9	I/O	MFP0	通用数字I/O脚
		OPA1_N	A	MFP1	运放 1 负极输入脚
		EBI_MCLK	O	MFP2	EBI 外部时钟输出脚.
		SC2_DAT	I/O	MFP3	智能卡 2 数据管脚.
		SPI2_MISO	I/O	MFP4	SPI2 MISO（主机输入，从机输出）脚.
		SD1_DAT1	I/O	MFP5	SD/SDIO1 数据线位1.
		USCI0_DAT1	I/O	MFP6	USCI0 数据1 管脚.
		UART1_TXD	O	MFP7	UART1 数据发送输出脚
		BPWM0_CH2	I/O	MFP9	BPWM0 通道2 输出/捕获输入
		QEI1_A	I	MFP10	正交编码器1 相位A输入
		ECAP0_IC1	I	MFP11	增强型捕获计数器 0 输入 1 管脚.
		TM2_EXT	I/O	MFP13	Timer2 事件计数输入/反转输出脚
		SWDH_DAT	I/O	MFP15	串行调试总线主机数据输入/输出管脚
11	16	PA.8	I/O	MFP0	通用数字I/O脚

64 管脚	128 管脚	管脚名称	类型	MFP	描述
		OPA1_P	A	MFP1	运放 1 正极输入脚
		EBI_ALE	O	MFP2	EBI地址锁存使能输出脚
		SC2_CLK	O	MFP3	智能卡 2 时钟脚
		SPI2_MOSI	I/O	MFP4	SPI2 MOSI (主机输出，从机输入) 管脚.
		SD1_DAT0	I/O	MFP5	SD/SDIO1 数据线位0.
		USCI0_CTL1	I/O	MFP6	USCI0 控制1 管脚.
		UART1_RXD	I	MFP7	UART1数据接收输入管脚
		BPWM0_CH3	I/O	MFP9	BPWM0 通道3输出/捕获输入脚
		QEI1_B	I	MFP10	正交编码器1 相位B 输入
		ECAP0_IC2	I	MFP11	增强型捕获计数器 0 输入 2 管脚.
		TM3_EXT	I/O	MFP13	Timer3 事件计数输入/反转输出脚
		INT4	I	MFP15	外部中断4输入管脚
	17	PC.13	I/O	MFP0	通用数字I/O脚
		EBI_ADR10	O	MFP2	EBI 地址总线位 10.
		SC2_nCD	I	MFP3	智能卡 2 卡检测管脚.
		SPI2_I2SMCLK	I/O	MFP4	SPI2 I^2S 主机时钟输出脚
		USCI0_CTL0	I/O	MFP6	USCI0 控制0 管脚.
		UART2_TXD	O	MFP7	UART2 数据发送输出脚.
		BPWM0_CH4	I/O	MFP9	BPWM0 通道4输出/捕获输入脚
		CLKO	O	MFP13	时钟输出
		EADC0_ST	I	MFP14	EADC0 外部触发输入
	18	PD.12	I/O	MFP0	通用数字I/O脚
		OPA2_O	A	MFP1	运放 2输出脚
		EBI_nCS0	O	MFP2	EBI片选0输出脚
		UART2_RXD	I	MFP7	UART2 数据接收输入脚
		BPWM0_CH5	I/O	MFP9	BPWM0 通道5 输出/捕获输入
		QEI0_INDEX	I	MFP10	正交编码器0 索引输入
		CLKO	O	MFP13	时钟输出
		EADC0_ST	I	MFP14	EADC0 外部触发输入
		INT5	I	MFP15	外部中断5 输入脚
	19	PD.11	I/O	MFP0	通用数字I/O脚

64 管脚	128 管脚	管脚名称	类型	MFP	描述
		OPA2_N	A	MFP1	运放 2 负极输入脚
		EBI_nCS1	O	MFP2	EBI片选1输出管脚
		UART1_TXD	O	MFP3	UART1 数据发送输出脚
		QEI0_A	I	MFP10	正交编码器0 相位A输入
		INT6	I	MFP15	外部中断6 输入脚
	20	PD.10	I/O	MFP0	通用数字I/O脚
		OPA2_P	A	MFP1	运放 2 正极输入脚
		EBI_nCS2	O	MFP2	EBI片选2输出脚
		UART1_RXD	I	MFP3	UART1数据接收输入管脚
		QEI0_B	I	MFP10	正交编码器0 相位B 输入
		INT7	I	MFP15	外部中断7 输入脚
	21	PG.2	I/O	MFP0	通用数字I/O脚
		EBI_ADR11	O	MFP2	EBI 地址总线位 11.
		SPI2_SS	I/O	MFP3	SPI2 从机选择脚
		I^2C0_SMBAL	O	MFP4	I^2C0 SMBus SMBALTER 管脚
		I^2C1_SCL	I/O	MFP5	I^2C1 时钟脚
		TM0	I/O	MFP13	Timer0 事件计数输入/反转输出管脚
	22	PG.3	I/O	MFP0	通用数字I/O脚
		EBI_ADR12	O	MFP2	EBI 地址总线位 12.
		SPI2_CLK	I/O	MFP3	SPI2 串行时钟脚
		I^2C0_SMBSUS	O	MFP4	I^2C0 SMBus SMBSUS 管脚 (PMBus 控制管脚)
		I^2C1_SDA	I/O	MFP5	I^2C1 数据输入/输出脚
		TM1	I/O	MFP13	Timer1 事件计数输入/反转输出脚
	23	PG.4	I/O	MFP0	通用数字I/O脚
		EBI_ADR13	O	MFP2	EBI 地址总线位 13.
		SPI2_MISO	I/O	MFP3	SPI2 MISO（主机输入，从机输出）脚.
		TM2	I/O	MFP13	Timer2 事件计数输入/反转输出脚
	24	PF.11	I/O	MFP0	通用数字I/O脚
		EBI_ADR14	O	MFP2	EBI 地址总线位 14.
		SPI2_MOSI	I/O	MFP3	SPI2 MOSI (主机输出，从机输入) 管脚.
		TAMPER5	I/O	MFP10	TAMPER循环检测管脚 5.

64 管脚	128 管脚	管脚名称	类型	MFP	描述
		TM3	I/O	MFP13	Timer3 事件计数输入/反转输出脚
	25	PF.10	I/O	MFP0	通用数字I/O脚
		EBI_ADR15	O	MFP2	EBI 地址总线位 15.
		SC0_nCD	I	MFP3	智能卡 0 卡检测管脚.
		I^{2}S0_BCLK	O	MFP4	I2S0位时钟输出管脚
		SPI0_I2SMCLK	I/O	MFP5	SPI0 I^{2}S 主机时钟输出脚
		TAMPER4	I/O	MFP10	TAMPER循环检测管脚 4.
	26	PF.9	I/O	MFP0	通用数字I/O脚
		EBI_ADR16	O	MFP2	EBI 地址总线位 16.
		SC0_PWR	O	MFP3	智能卡0电源脚
		I^{2}S0_MCLK	O	MFP4	I2S0 主时钟输出脚
		SPI0_SS	I/O	MFP5	SPI0 从机选择脚
		TAMPER3	I/O	MFP10	TAMPER循环检测管脚 3.
	27	PF.8	I/O	MFP0	通用数字I/O脚
		EBI_ADR17	O	MFP2	EBI 地址总线位 17.
		SC0_RST	O	MFP3	智能卡0复位脚
		I^{2}S0_DI	I	MFP4	I2S0 数据输入脚
		SPI0_CLK	I/O	MFP5	SPI0 串行时钟脚
		TAMPER2	I/O	MFP10	TAMPER循环检测管脚 2.
	28	PF.7	I/O	MFP0	通用数字I/O脚
		EBI_ADR18	O	MFP2	EBI 地址总线位 18.
		SC0_DAT	I/O	MFP3	智能卡0数据输入脚
		I^{2}S0_DO	O	MFP4	I^{2}S0 数据输出脚
		SPI0_MISO	I/O	MFP5	SPI0 MISO（主机输入，从机输出）脚.
		UART4_TXD	O	MFP6	UART4 数据发送输出脚.
		TAMPER1	I/O	MFP10	TAMPER循环检测管脚 1.
12	29	PF.6	I/O	MFP0	通用数字I/O脚
		EBI_ADR19	O	MFP2	EBI 地址总线位 19.
		SC0_CLK	O	MFP3	智能卡0时钟管脚
		I^{2}S0_LRCK	O	MFP4	I^{2}S0左右通道时钟输出脚.
		SPI0_MOSI	I/O	MFP5	SPI0 MOSI (主机输出，从机输入) 管脚.

64 管脚	128 管脚	管脚名称	类型	MFP	描述
		UART4_RXD	I	MFP6	UART4 数据接收输入脚
		EBI_nCS0	O	MFP7	EBI片选0输出脚
		TAMPER0	I/O	MFP10	TAMPER循环检测管脚 0.
13	30	V_{DD}	P	MFP0	I/O和内部PLL及数字电路的LDO电源
14	31	PF.5	I/O	MFP0	通用数字I/O脚
		UART2_RXD	I	MFP2	UART2 数据接收输入脚
		UART2_nCTS	I	MFP4	UART2 禁止发送输入管脚
		BPWM0_CH4	I/O	MFP8	BPWM0 通道4输出/捕获输入脚
		EPWM0_SYNC_OUT	O	MFP9	EPWM0计数器同步触发输出脚
		X32_IN	I	MFP10	外部 32.768 kHz 晶振输入脚
		EADC0_ST	I	MFP11	EADC0 外部触发输入
15	32	PF.4	I/O	MFP0	通用数字I/O脚
		UART2_TXD	O	MFP2	UART2 数据发送输出脚.
		UART2_nRTS	O	MFP4	UART2 请求发送输出脚
		BPWM0_CH5	I/O	MFP8	BPWM0 通道5 输出/捕获输入
		X32_OUT	O	MFP10	外部 32.768 kHz 晶振输出脚
	33	PH.4	I/O	MFP0	通用数字I/O脚
		EBI_ADR3	O	MFP2	EBI 地址总线位3
		SPI1_MISO	I/O	MFP3	SPI1 MISO(主机输入，从机输出)管脚
	34	PH.5	I/O	MFP0	通用数字I/O脚
		EBI_ADR2	O	MFP2	EBI 地址总线位2
		SPI1_MOSI	I/O	MFP3	SPI1 MOSI (主机输出，从机输入) 管脚
	35	PH.6	I/O	MFP0	通用数字I/O脚
		EBI_ADR1	O	MFP2	EBI 地址总线位1
		SPI1_CLK	I/O	MFP3	SPI1 串行时钟管脚
	36	PH.7	I/O	MFP0	通用数字I/O脚
		EBI_ADR0	O	MFP2	EBI地址总线位0
		SPI1_SS	I/O	MFP3	SPI1 从机选择脚
16	37	PF.3	I/O	MFP0	通用数字I/O脚
		EBI_nCS0	O	MFP2	EBI片选0输出脚
		UART0_TXD	O	MFP3	UART0 数据发送输出脚.

64 管脚	128 管脚	管脚名称	类型	MFP	描述
		I^2C0_SCL	I/O	MFP4	I2C0 时钟管脚
		XT1_IN	I	MFP10	外部 4~24 MHz (高速) 晶振输入脚
		BPWM1_CH0	I/O	MFP11	BPWM1 通道0 输出/捕获输入
17	38	PF.2	I/O	MFP0	通用数字I/O脚
		EBI_nCS1	O	MFP2	EBI片选1输出管脚
		UART0_RXD	I	MFP3	UART0 数据接收输入脚
		I^2C0_SDA	I/O	MFP4	I2C0 数据输入/输出管脚
		QSPI0_CLK	I/O	MFP5	QSPI0 串行时钟脚
		XT1_OUT	O	MFP10	外部 4~24 MHz (高速) 晶振输出脚
		BPWM1_CH1	I/O	MFP11	BPWM1 通道1 输出/捕获输入
	39	V_{SS}	P	MFP0	数字电路地管脚
	40	V_{DD}	P	MFP0	I/O和内部PLL及数字电路的LDO电源
	41	PE.8	I/O	MFP0	通用数字I/O脚
		EBI_ADR10	O	MFP2	EBI 地址总线位 10.
		I^2S0_BCLK	O	MFP4	I2S0位时钟输出管脚
		SPI2_CLK	I/O	MFP5	SPI2 串行时钟脚
		USCI1_CTL1	I/O	MFP6	USCI1 控制1管脚
		UART2_TXD	O	MFP7	UART2 数据发送输出脚.
		EPWM0_CH0	I/O	MFP10	EPWM0 通道0输出/捕获输入
		EPWM0_BRAKE0	I	MFP11	EPWM0 刹车0 输入脚
		ECAP0_IC0	I	MFP12	增强型捕获计数器 0 输入脚0
		TRACE_CLK	O	MFP14	ETM跟踪时钟输出脚
	42	PE.9	I/O	MFP0	通用数字I/O脚
		EBI_ADR11	O	MFP2	EBI 地址总线位 11.
		I^2S0_MCLK	O	MFP4	I2S0 主时钟输出脚
		SPI2_MISO	I/O	MFP5	SPI2 MISO（主机输入，从机输出）脚.
		USCI1_CTL0	I/O	MFP6	USCI1 控制0管脚
		UART2_RXD	I	MFP7	UART2 数据接收输入脚
		EPWM0_CH1	I/O	MFP10	EPWM0 通道1输出/捕获输入脚
		EPWM0_BRAKE1	I	MFP11	EPWM0 刹车1 输入脚
		ECAP0_IC1	I	MFP12	增强型捕获计数器 0 输入 1 管脚.

64 管脚	128 管脚	管脚名称	类型	MFP	描述
		TRACE_DATA0	O	MFP14	ETM跟踪数据0 输出管脚
	43	PE.10	I/O	MFP0	通用数字I/O脚
		EBI_ADR12	O	MFP2	EBI 地址总线位 12.
		I²S0_DI	I	MFP4	I2S0 数据输入脚
		SPI2_MOSI	I/O	MFP5	SPI2 MOSI (主机输出，从机输入) 管脚.
		USCI1_DAT0	I/O	MFP6	USCI1 数据0脚
		UART3_TXD	O	MFP7	UART3 数据发送输出脚.
		EPWM0_CH2	I/O	MFP10	EPWM0 通道2输出/捕获输入脚
		EPWM1_BRAKE0	I	MFP11	EPWM1 刹车0 输入脚
		ECAP0_IC2	I	MFP12	增强型捕获计数器 0 输入 2 管脚.
		TRACE_DATA1	O	MFP14	ETM跟踪数据1 输出管脚
	44	PE.11	I/O	MFP0	通用数字I/O脚
		EBI_ADR13	O	MFP2	EBI 地址总线位 13.
		I²S0_DO	O	MFP4	I²S0 数据输出脚
		SPI2_SS	I/O	MFP5	SPI2 从机选择脚
		USCI1_DAT1	I/O	MFP6	USCI1数据1管脚.
		UART3_RXD	I	MFP7	UART3 数据接收输入脚
		UART1_nCTS	I	MFP8	UART1 禁止发送输入管脚
		EPWM0_CH3	I/O	MFP10	EPWM0 通道3输出/捕获输入脚
		EPWM1_BRAKE1	I	MFP11	EPWM1 刹车1输入管脚
		ECAP1_IC2	I	MFP13	增强型捕获计数器 1 输入 2 管脚.
		TRACE_DATA2	O	MFP14	ETM跟踪数据2 输出管脚
	45	PE.12	I/O	MFP0	通用数字I/O脚
		EBI_ADR14	O	MFP2	EBI 地址总线位 14.
		I²S0_LRCK	O	MFP4	I²S0左右通道时钟输出脚.
		SPI2_I2SMCLK	I/O	MFP5	SPI2 I²S 主机时钟输出脚
		USCI1_CLK	I/O	MFP6	USCI1 时钟脚
		UART1_nRTS	O	MFP8	UART1 请求发送输出脚
		EPWM0_CH4	I/O	MFP10	EPWM0 通道4输出/捕获输入脚
		ECAP1_IC1	I	MFP13	增强型捕获计数器 1 输入 1 管脚.
		TRACE_DATA3	O	MFP14	ETM跟踪数据3 输出管脚

64 管脚	128 管脚	管脚名称	类型	MFP	描述
	46	PE.13	I/O	MFP0	通用数字I/O脚
		EBI_ADR15	O	MFP2	EBI 地址总线位 15.
		I²C0_SCL	I/O	MFP4	I2C0 时钟管脚
		UART4_nRTS	O	MFP5	UART4 请求发送输出脚
		UART1_TXD	O	MFP8	UART1 数据发送输出脚
		EPWM0_CH5	I/O	MFP10	EPWM0 通道5 输出/捕获输入
		EPWM1_CH0	I/O	MFP11	EPWM1 通道0 输出/捕获输入
		BPWM1_CH5	I/O	MFP12	EPWM1 通道5输出/捕获输入
		ECAP1_IC0	I	MFP13	增强型捕获计数器 1 输入脚0
	47	PC.8	I/O	MFP0	通用数字I/O脚
		EBI_ADR16	O	MFP2	EBI 地址总线位 16.
		I²C0_SDA	I/O	MFP4	I2C0 数据输入/输出管脚
		UART4_nCTS	I	MFP5	UART4 禁止发送输入管脚
		UART1_RXD	I	MFP8	UART1数据接收输入管脚
		EPWM1_CH1	I/O	MFP11	EPWM1 通道1 输出/捕获输入
		BPWM1_CH4	I/O	MFP12	BPWM1 通道4输出/捕获输入脚
18	48	PC.7	I/O	MFP0	通用数字I/O脚
		EBI_AD9	I/O	MFP2	EBI地址/数据总线位9.
		SPI1_MISO	I/O	MFP4	SPI1 MISO(主机输入，从机输出)管脚
		UART4_TXD	O	MFP5	UART4 数据发送输出脚.
		SC2_PWR	O	MFP6	智能卡 2电源脚
		UART0_nCTS	I	MFP7	UART0 禁止发送输入管脚
		I²C1_SMBAL	O	MFP8	I²C1 SMBus SMBALTER 管脚
		EPWM1_CH2	I/O	MFP11	EPWM1 通道2 输出/捕获输入
		BPWM1_CH0	I/O	MFP12	BPWM1 通道0 输出/捕获输入
		TM0	I/O	MFP14	Timer0 事件计数输入/反转输出管脚
		INT3	I	MFP15	外部中断3输入脚
19	49	PC.6	I/O	MFP0	通用数字I/O脚
		EBI_AD8	I/O	MFP2	EBI地址/数据总线位8.
		SPI1_MOSI	I/O	MFP4	SPI1 MOSI (主机输出，从机输入) 管脚
		UART4_RXD	I	MFP5	UART4 数据接收输入脚

64 管脚	128 管脚	管脚名称	类型	MFP	描述
		SC2_RST	O	MFP6	智能卡 2 复位脚
		UART0_nRTS	O	MFP7	UART0 请求发送输出脚
		I^2C1_SMBSUS	O	MFP8	I^2C1 SMBus SMBSUS 管脚 (PMBus 控制管脚)
		EPWM1_CH3	I/O	MFP11	EPWM1 通道3输出/捕获输入脚
		BPWM1_CH1	I/O	MFP12	BPWM1 通道1 输出/捕获输入
		TM1	I/O	MFP14	Timer1 事件计数输入/反转输出脚
		INT2	I	MFP15	外部中断2输入脚
20	50	PA.7	I/O	MFP0	通用数字I/O脚
		EBI_AD7	I/O	MFP2	EBI地址/数据总线位7.
		SPI1_CLK	I/O	MFP4	SPI1 串行时钟管脚
		SC2_DAT	I/O	MFP6	智能卡 2 数据管脚.
		UART0_TXD	O	MFP7	UART0 数据发送输出脚.
		I^2C1_SCL	I/O	MFP8	I^2C1 时钟脚
		EPWM1_CH4	I/O	MFP11	EPWM1 通道4输出/捕获输入脚
		BPWM1_CH2	I/O	MFP12	BPWM1 通道2 输出/捕获输入
		ACMP0_WLAT	I	MFP13	模拟比较器 0窗口锁存输入管脚
		TM2	I/O	MFP14	Timer2 事件计数输入/反转输出脚
		INT1	I	MFP15	外部中断1输入脚
21	51	PA.6	I/O	MFP0	通用数字I/O脚
		EBI_AD6	I/O	MFP2	EBI地址/数据总线位6.
		SPI1_SS	I/O	MFP4	SPI1 从机选择脚
		SD1_nCD	I	MFP5	SD/SDIO1 卡检测输入管脚
		SC2_CLK	O	MFP6	智能卡 2 时钟脚
		UART0_RXD	I	MFP7	UART0 数据接收输入脚
		I^2C1_SDA	I/O	MFP8	I^2C1 数据输入/输出脚
		EPWM1_CH5	I/O	MFP11	EPWM1 通道5输出/捕获输入
		BPWM1_CH3	I/O	MFP12	BPWM1 通道3输出/捕获输入脚
		ACMP1_WLAT	I	MFP13	模拟比较器 1窗口锁存输入管脚
		TM3	I/O	MFP14	Timer3 事件计数输入/反转输出脚
		INT0	I	MFP15	外部中断0输入管脚
22	52	V_{SS}	P	MFP0	数字电路地管脚

64 管脚	128 管脚	管脚名称	类型	MFP	描述
23	53	V_{DD}	P	MFP0	I/O和内部PLL及数字电路的LDO电源
24	54	LDO_CAP	A	MFP0	LDO输出脚
25	55	PA.5	I/O	MFP0	通用数字I/O脚
		SPIM_D2	I/O	MFP2	SPIM QuadI/O模式下数据2管脚
		QSPI0_MISO1	I/O	MFP3	QSPI0 MISO1（主机输入，从机输出）脚.
		SPI1_I2SMCLK	I/O	MFP4	SPI1 I^2S 主机时钟输出脚
		SD1_CMD	I/O	MFP5	SD/SDIO1 命令/响应脚
		SC2_nCD	I	MFP6	智能卡 2 卡检测管脚.
		UART0_nCTS	I	MFP7	UART0 禁止发送输入管脚
		UART5_TXD	O	MFP8	UART5数据发送输出脚
		I^2C0_SCL	I/O	MFP9	I2C0 时钟管脚
		BPWM0_CH5	I/O	MFP12	BPWM0 通道5 输出/捕获输入
		EPWM0_CH0	I/O	MFP13	EPWM0 通道0输出/捕获输入
		QEI0_INDEX	I	MFP14	正交编码器0 索引输入
26	56	PA.4	I/O	MFP0	通用数字I/O脚
		SPIM_D3	I/O	MFP2	SPIM QuadI/O模式下数据3管脚
		QSPI0_MOSI1	I/O	MFP3	QSPI0 MOSI1 (主机输出，从机输入) 管脚.
		SPI0_I2SMCLK	I/O	MFP4	SPI0 I^2S 主机时钟输出脚
		SD1_CLK	O	MFP5	SD/SDIO1 时钟输出管脚
		SC0_nCD	I	MFP6	智能卡 0 卡检测管脚.
		UART0_nRTS	O	MFP7	UART0 请求发送输出脚
		UART5_RXD	I	MFP8	UART5 数据接收输入管脚
		I^2C0_SDA	I/O	MFP9	I2C0 数据输入/输出管脚
		BPWM0_CH4	I/O	MFP12	BPWM0 通道4输出/捕获输入脚
		EPWM0_CH1	I/O	MFP13	EPWM0 通道1输出/捕获输入脚
		QEI0_A	I	MFP14	正交编码器0 相位A输入
27	57	PA.3	I/O	MFP0	通用数字I/O脚
		SPIM_SS	I/O	MFP2	SPIM 从机选择脚
		QSPI0_SS	I/O	MFP3	QSPI0 从机选择脚
		SPI0_SS	I/O	MFP4	SPI0 从机选择脚
		SD1_DAT3	I/O	MFP5	SD/SDIO1 数据线位3.

64 管脚	128 管脚	管脚名称	类型	MFP	描述
		SC0_PWR	O	MFP6	智能卡0电源脚
		UART4_TXD	O	MFP7	UART4 数据发送输出脚.
		UART1_TXD	O	MFP8	UART1 数据发送输出脚
		I²C1_SCL	I/O	MFP9	I²C1 时钟脚
		BPWM0_CH3	I/O	MFP12	BPWM0 通道3输出/捕获输入脚
		EPWM0_CH2	I/O	MFP13	EPWM0 通道2输出/捕获输入脚
		QEI0_B	I	MFP14	正交编码器0 相位B 输入
28	58	PA.2	I/O	MFP0	通用数字I/O脚
		SPIM_CLK	I/O	MFP2	SPIM 串行时钟脚
		QSPI0_CLK	I/O	MFP3	QSPI0 串行时钟脚
		SPI0_CLK	I/O	MFP4	SPI0 串行时钟脚
		SD1_DAT2	I/O	MFP5	SD/SDIO1 数据线位2.
		SC0_RST	O	MFP6	智能卡0复位脚
		UART4_RXD	I	MFP7	UART4 数据接收输入脚
		UART1_RXD	I	MFP8	UART1数据接收输入管脚
		I²C1_SDA	I/O	MFP9	I²C1 数据输入/输出脚
		BPWM0_CH2	I/O	MFP12	BPWM0 通道2 输出/捕获输入
		EPWM0_CH3	I/O	MFP13	EPWM0 通道3输出/捕获输入脚
29	59	PA.1	I/O	MFP0	通用数字I/O脚
		SPIM_MISO	I/O	MFP2	SPIM MISO（主机输入，从机输出）脚.
		QSPI0_MISO0	I/O	MFP3	QSPI0 MISO0（主机输入，从机输出）脚.
		SPI0_MISO	I/O	MFP4	SPI0 MISO（主机输入，从机输出）脚.
		SD1_DAT1	I/O	MFP5	SD/SDIO1 数据线位1.
		SC0_DAT	I/O	MFP6	智能卡0数据输入脚
		UART0_TXD	O	MFP7	UART0 数据发送输出脚.
		UART1_nCTS	I	MFP8	UART1 禁止发送输入管脚
		I²C2_SCL	I/O	MFP9	I²C2 时钟脚
		BPWM0_CH1	I/O	MFP12	BPWM0 通道1 输出/捕获输入
		EPWM0_CH4	I/O	MFP13	EPWM0 通道4输出/捕获输入脚
		DAC1_ST	I	MFP15	DAC1 外部触发输入
30	60	PA.0	I/O	MFP0	通用数字I/O脚

64 管脚	128 管脚	管脚名称	类型	MFP	描述
		SPIM_MOSI	I/O	MFP2	SPIM MOSI (主机输出，从机输入) 管脚.
		QSPI0_MOSI0	I/O	MFP3	QSPI0 MOSI0 (主机输出，从机输入) 管脚.
		SPI0_MOSI	I/O	MFP4	SPI0 MOSI (主机输出，从机输入) 管脚.
		SD1_DAT0	I/O	MFP5	SD/SDIO1 数据线位0.
		SC0_CLK	O	MFP6	智能卡0时钟管脚
		UART0_RXD	I	MFP7	UART0 数据接收输入脚
		UART1_nRTS	O	MFP8	UART1 请求发送输出脚
		I^2C2_SDA	I/O	MFP9	I^2C2 数据输入/输出脚
		BPWM0_CH0	I/O	MFP12	BPWM0 通道0 输出/捕获输入
		EPWM0_CH5	I/O	MFP13	EPWM0 通道5 输出/捕获输入
		DAC0_ST	I	MFP15	DAC0 外部触发输入
31	61	V_{DDIO}	P	MFP0	PE.1, PE.8~PE.13 的电源
	62	PE.14	I/O	MFP0	通用数字I/O脚
		EBI_AD8	I/O	MFP2	EBI地址/数据总线位8.
		UART2_TXD	O	MFP3	UART2 数据发送输出脚.
		SD1_nCD	I	MFP5	SD/SDIO1 卡检测输入管脚
	63	PE.15	I/O	MFP0	通用数字I/O脚
		EBI_AD9	I/O	MFP2	EBI地址/数据总线位9.
		UART2_RXD	I	MFP3	UART2 数据接收输入脚
32	64	nRESET	I	MFP0	外部复位输入:低电平有效,有内部上拉，设置此管脚为低将复位初始状态
33	65	PF.0	I/O	MFP0	通用数字I/O脚
		UART1_TXD	O	MFP2	UART1 数据发送输出脚
		I^2C1_SCL	I/O	MFP3	I^2C1 时钟脚
		BPWM1_CH0	I/O	MFP12	BPWM1 通道0 输出/捕获输入
		ICE_DAT	O	MFP14	串行调试总线数据管脚.
34	66	PF.1	I/O	MFP0	通用数字I/O脚
		UART1_RXD	I	MFP2	UART1数据接收输入管脚
		I^2C1_SDA	I/O	MFP3	I^2C1 数据输入/输出脚
		BPWM1_CH1	I/O	MFP12	BPWM1 通道1 输出/捕获输入
		ICE_CLK	I	MFP14	串行调试总线时钟脚

64 管脚	128 管脚	管脚名称	类型	MFP	描述
	67	PD.9	I/O	MFP0	通用数字I/O脚
		EBI_AD7	I/O	MFP2	EBI地址/数据总线位7.
		I²C2_SCL	I/O	MFP3	I²C2 时钟脚
		UART2_nCTS	I	MFP4	UART2 禁止发送输入管脚
	68	PD.8	I/O	MFP0	通用数字I/O脚
		EBI_AD6	I/O	MFP2	EBI地址/数据总线位6.
		I²C2_SDA	I/O	MFP3	I²C2 数据输入/输出脚
		UART2_nRTS	O	MFP4	UART2 请求发送输出脚
35	69	PC.5	I/O	MFP0	通用数字I/O脚
		EBI_AD5	I/O	MFP2	EBI地址/数据总线位5.
		SPIM_D2	I/O	MFP3	SPIM QuadI/O模式下数据2管脚
		QSPI0_MISO1	I/O	MFP4	QSPI0 MISO1（主机输入，从机输出）脚.
		UART2_TXD	O	MFP8	UART2 数据发送输出脚.
		I²C1_SCL	I/O	MFP9	I²C1 时钟脚
		UART4_TXD	O	MFP11	UART4 数据发送输出脚.
		EPWM1_CH0	I/O	MFP12	EPWM1 通道0 输出/捕获输入
36	70	PC.4	I/O	MFP0	通用数字I/O脚
		EBI_AD4	I/O	MFP2	EBI地址/数据总线位4.
		SPIM_D3	I/O	MFP3	SPIM QuadI/O模式下数据3管脚
		QSPI0_MOSI1	I/O	MFP4	QSPI0 MOSI1 (主机输出，从机输入) 管脚.
		SC1_nCD	I	MFP5	智能卡 1 卡检测管脚.
		I²S0_BCLK	O	MFP6	I2S0位时钟输出管脚
		SPI1_I2SMCLK	I/O	MFP7	SPI1 I²S 主机时钟输出脚
		UART2_RXD	I	MFP8	UART2 数据接收输入脚
		I²C1_SDA	I/O	MFP9	I²C1 数据输入/输出脚
		UART4_RXD	I	MFP11	UART4 数据接收输入脚
		EPWM1_CH1	I/O	MFP12	EPWM1 通道1 输出/捕获输入
37	71	PC.3	I/O	MFP0	通用数字I/O脚
		EBI_AD3	I/O	MFP2	EBI地址/数据总线位3.
		SPIM_SS	I/O	MFP3	SPIM 从机选择脚
		QSPI0_SS	I/O	MFP4	QSPI0 从机选择脚

64 管脚	128 管脚	管脚名称	类型	MFP	描述
		SC1_PWR	O	MFP5	智能卡 1电源脚
		I²S0_MCLK	O	MFP6	I2S0 主时钟输出脚
		SPI1_MISO	I/O	MFP7	SPI1 MISO(主机输入，从机输出)管脚
		UART2_nRTS	O	MFP8	UART2 请求发送输出脚
		I²C0_SMBAL	O	MFP9	I²C0 SMBus SMBALTER 管脚
		UART3_TXD	O	MFP11	UART3 数据发送输出脚.
		EPWM1_CH2	I/O	MFP12	EPWM1 通道2 输出/捕获输入
38	72	PC.2	I/O	MFP0	通用数字I/O脚
		EBI_AD2	I/O	MFP2	EBI地址/数据总线位2.
		SPIM_CLK	I/O	MFP3	SPIM 串行时钟脚
		QSPI0_CLK	I/O	MFP4	QSPI0 串行时钟脚
		SC1_RST	O	MFP5	智能卡 1 复位脚
		I²S0_DI	I	MFP6	I2S0 数据输入脚
		SPI1_MOSI	I/O	MFP7	SPI1 MOSI (主机输出，从机输入) 管脚
		UART2_nCTS	I	MFP8	UART2 禁止发送输入管脚
		I²C0_SMBSUS	O	MFP9	I²C0 SMBus SMBSUS 管脚 (PMBus 控制管脚)
		UART3_RXD	I	MFP11	UART3 数据接收输入脚
		EPWM1_CH3	I/O	MFP12	EPWM1 通道3输出/捕获输入脚
39	73	PC.1	I/O	MFP0	通用数字I/O脚
		EBI_AD1	I/O	MFP2	EBI地址/数据总线位1.
		SPIM_MISO	I/O	MFP3	SPIM MISO（主机输入，从机输出）脚.
		QSPI0_MISO0	I/O	MFP4	QSPI0 MISO0（主机输入，从机输出）脚.
		SC1_DAT	I/O	MFP5	智能卡 1 数据管脚.
		I²S0_DO	O	MFP6	I²S0 数据输出脚
		SPI1_CLK	I/O	MFP7	SPI1 串行时钟管脚
		UART2_TXD	O	MFP8	UART2 数据发送输出脚.
		I²C0_SCL	I/O	MFP9	I2C0 时钟管脚
		EPWM1_CH4	I/O	MFP12	EPWM1 通道4输出/捕获输入脚
		ACMP0_O	O	MFP14	模拟比较器 0输出脚
40	74	PC.0	I/O	MFP0	通用数字I/O脚
		EBI_AD0	I/O	MFP2	EBI地址/数据总线位0.

64 管脚	128 管脚	管脚名称	类型	MFP	描述
		SPIM_MOSI	I/O	MFP3	SPIM MOSI (主机输出，从机输入) 管脚.
		QSPI0_MOSI0	I/O	MFP4	QSPI0 MOSI0 (主机输出，从机输入) 管脚.
		SC1_CLK	O	MFP5	智能卡 1 时钟脚
		I^2S0_LRCK	O	MFP6	I^2S0左右通道时钟输出脚.
		SPI1_SS	I/O	MFP7	SPI1 从机选择脚
		UART2_RXD	I	MFP8	UART2 数据接收输入脚
		I^2C0_SDA	I/O	MFP9	I2C0 数据输入/输出管脚
		EPWM1_CH5	I/O	MFP12	EPWM1 通道5输出/捕获输入
		ACMP1_O	O	MFP14	模拟比较器1输出脚
	75	V_{SS}	P	MFP0	数字电路地管脚
	76	V_{DD}	P	MFP0	I/O和内部PLL及数字电路的LDO电源
	77	PG.9	I/O	MFP0	通用数字I/O脚
		EBI_AD0	I/O	MFP2	EBI地址/数据总线位0.
		SD1_DAT3	I/O	MFP3	SD/SDIO1 数据线位3.
		SPIM_D2	I/O	MFP4	SPIM QuadI/O模式下数据2管脚
		BPWM0_CH5	I/O	MFP12	BPWM0 通道5 输出/捕获输入
	78	PG.10	I/O	MFP0	通用数字I/O脚
		EBI_AD1	I/O	MFP2	EBI地址/数据总线位1.
		SD1_DAT2	I/O	MFP3	SD/SDIO1 数据线位2.
		SPIM_D3	I/O	MFP4	SPIM QuadI/O模式下数据3管脚
		BPWM0_CH4	I/O	MFP12	BPWM0 通道4输出/捕获输入脚
	79	PG.11	I/O	MFP0	通用数字I/O脚
		EBI_AD2	I/O	MFP2	EBI地址/数据总线位2.
		SD1_DAT1	I/O	MFP3	SD/SDIO1 数据线位1.
		SPIM_SS	I/O	MFP4	SPIM 从机选择脚
		BPWM0_CH3	I/O	MFP12	BPWM0 通道3输出/捕获输入脚
	80	PG.12	I/O	MFP0	通用数字I/O脚
		EBI_AD3	I/O	MFP2	EBI地址/数据总线位3.
		SD1_DAT0	I/O	MFP3	SD/SDIO1 数据线位0.
		SPIM_CLK	I/O	MFP4	SPIM 串行时钟脚
		BPWM0_CH2	I/O	MFP12	BPWM0 通道2 输出/捕获输入

64 管脚	128 管脚	管脚名称	类型	MFP	描述
	81	PG.13	I/O	MFP0	通用数字I/O脚
		EBI_AD4	I/O	MFP2	EBI地址/数据总线位4.
		SD1_CMD	I/O	MFP3	SD/SDIO1 命令/响应脚
		SPIM_MISO	I/O	MFP4	SPIM MISO（主机输入，从机输出）脚.
		BPWM0_CH1	I/O	MFP12	BPWM0 通道1 输出/捕获输入
	82	PG.14	I/O	MFP0	通用数字I/O脚
		EBI_AD5	I/O	MFP2	EBI地址/数据总线位5.
		SD1_CLK	O	MFP3	SD/SDIO1 时钟输出管脚
		SPIM_MOSI	I/O	MFP4	SPIM MOSI (主机输出，从机输入) 管脚.
		BPWM0_CH0	I/O	MFP12	BPWM0 通道0 输出/捕获输入
	83	PG.15	I/O	MFP0	通用数字I/O脚
		SD1_nCD	I	MFP3	SD/SDIO1 卡检测输入管脚
		CLKO	O	MFP14	时钟输出
		EADC0_ST	I	MFP15	EADC0 外部触发输入
	84	PD.13	I/O	MFP0	通用数字I/O脚
		EBI_AD10	I/O	MFP2	EBI地址/数据总线位10.
		SD0_nCD	I	MFP3	SD/SDIO0 卡检测输入管脚
		SPI0_I2SMCLK	I/O	MFP4	SPI0 I^2S 主机时钟输出脚
		SPI1_I2SMCLK	I/O	MFP5	SPI1 I^2S 主机时钟输出脚
		SC2_nCD	I	MFP7	智能卡 2 卡检测管脚.
	85	PA.12	I/O	MFP0	通用数字I/O脚
		I^2S0_BCLK	O	MFP2	I2S0位时钟输出管脚
		UART4_TXD	O	MFP3	UART4 数据发送输出脚.
		I^2C1_SCL	I/O	MFP4	I^2C1 时钟脚
		SPI2_SS	I/O	MFP5	SPI2 从机选择脚
		SC2_PWR	O	MFP7	智能卡 2电源脚
		BPWM1_CH2	I/O	MFP11	BPWM1 通道2 输出/捕获输入
		QEI1_INDEX	I	MFP12	正交编码器1 索引输入
		USB_VBUS	P	MFP14	USB主机或HUB的电源
	86	PA.13	I/O	MFP0	通用数字I/O脚
		I^2S0_MCLK	O	MFP2	I2S0 主时钟输出脚

64 管脚	128 管脚	管脚名称	类型	MFP	描述
		UART4_RXD	I	MFP3	UART4 数据接收输入脚
		I²C1_SDA	I/O	MFP4	I²C1 数据输入/输出脚
		SPI2_CLK	I/O	MFP5	SPI2 串行时钟脚
		SC2_RST	O	MFP7	智能卡 2 复位脚
		BPWM1_CH3	I/O	MFP11	BPWM1 通道3输出/捕获输入脚
		QEI1_A	I	MFP12	正交编码器1 相位A输入
		USB_D-	A	MFP14	USB差分信号D+.
	87	PA.14	I/O	MFP0	通用数字I/O脚
		I²S0_DI	I	MFP2	I2S0 数据输入脚
		UART0_TXD	O	MFP3	UART0 数据发送输出脚.
		SPI2_MISO	I/O	MFP5	SPI2 MISO（主机输入，从机输出）脚.
		I²C2_SCL	I/O	MFP6	I²C2 时钟脚
		SC2_DAT	I/O	MFP7	智能卡 2 数据管脚.
		BPWM1_CH4	I/O	MFP11	BPWM1 通道4输出/捕获输入脚
		QEI1_B	I	MFP12	正交编码器1 相位B 输入
		USB_D+	A	MFP14	USB差分信号D-.
	88	PA.15	I/O	MFP0	通用数字I/O脚
		I²S0_DO	O	MFP2	I²S0 数据输出脚
		UART0_RXD	I	MFP3	UART0 数据接收输入脚
		SPI2_MOSI	I/O	MFP5	SPI2 MOSI (主机输出，从机输入) 管脚.
		I²C2_SDA	I/O	MFP6	I²C2 数据输入/输出脚
		SC2_CLK	O	MFP7	智能卡 2 时钟脚
		BPWM1_CH5	I/O	MFP11	EPWM1 通道5输出/捕获输入
		EPWM0_SYNC_IN	I	MFP12	EPWM0 计数器同步触发输入脚
		USB_OTG_ID	I	MFP14	USB_ 标识
41	89	HSUSB_VRES	A	MFP0	HSUSB模块参考电阻
42	90	HSUSB_V_{DD33}	P	MFP0	HSUSB VDD33 电源供电
43	91	HSUSB_VBUS	P	MFP0	HSUSB USB主机或HUB的电源
44	92	HSUSB_D-	A	MFP0	HSUSB差分信号D+.
45	93	HSUSB_VSS	P	MFP0	HSUSB 地管脚
46	94	HSUSB_D+	A	MFP0	HSUSB差分信号D-.

64 管脚	128 管脚	管脚名称	类型	MFP	描述
47	95	HSUSB_V_{DD12}_CAP	A	MFP0	HSUSB内部1.2v电压输出去耦管脚. 注: 此管脚需接1uF电容到地.
48	96	HSUSB_ID	I	MFP0	HSUSB 标识
	97	PE.7	I/O	MFP0	通用数字I/O脚
		SD0_CMD	I/O	MFP3	SD/SDIO0 命令/响应脚
		SPIM_D2	I/O	MFP4	SPIM QuadI/O模式下数据2管脚
		UART5_TXD	O	MFP8	UART5数据发送输出脚
		QEI1_INDEX	I	MFP11	正交编码器1 索引输入
		EPWM0_CH0	I/O	MFP12	EPWM0 通道0输出/捕获输入
		BPWM0_CH5	I/O	MFP13	BPWM0 通道5 输出/捕获输入
	98	PE.6	I/O	MFP0	通用数字I/O脚
		SD0_CLK	O	MFP3	SD/SDIO0 时钟输出脚
		SPIM_D3	I/O	MFP4	SPIM QuadI/O模式下数据3管脚
		SPI3_I2SMCLK	I/O	MFP5	SPI3 I^2S 主机时钟输出脚
		SC0_nCD	I	MFP6	智能卡 0 卡检测管脚.
		USCI0_CTL0	I/O	MFP7	USCI0 控制0 管脚.
		UART5_RXD	I	MFP8	UART5 数据接收输入管脚
		QEI1_A	I	MFP11	正交编码器1 相位A输入
		EPWM0_CH1	I/O	MFP12	EPWM0 通道1输出/捕获输入脚
		BPWM0_CH4	I/O	MFP13	BPWM0 通道4输出/捕获输入脚
	99	PE.5	I/O	MFP0	通用数字I/O脚
		EBI_nRD	O	MFP2	EBI 读使能输出脚
		SD0_DAT3	I/O	MFP3	SD/SDIO0数据线位3
		SPIM_SS	I/O	MFP4	SPIM 从机选择脚
		SPI3_SS	I/O	MFP5	SPI3 从机选择脚
		SC0_PWR	O	MFP6	智能卡0电源脚
		USCI0_CTL1	I/O	MFP7	USCI0 控制1 管脚.
		QEI1_B	I	MFP11	正交编码器1 相位B 输入
		EPWM0_CH2	I/O	MFP12	EPWM0 通道2输出/捕获输入脚
		BPWM0_CH3	I/O	MFP13	BPWM0 通道3输出/捕获输入脚
	100	PE.4	I/O	MFP0	通用数字I/O脚

64 管脚	128 管脚	管脚名称	类型	MFP	描述
		EBI_nWR	O	MFP2	EBI 写使能输出脚
		SD0_DAT2	I/O	MFP3	SD/SDIO0 数据线位2
		SPIM_CLK	I/O	MFP4	SPIM 串行时钟脚
		SPI3_CLK	I/O	MFP5	SPI3 串行时钟脚
		SC0_RST	O	MFP6	智能卡0复位脚
		USCI0_DAT1	I/O	MFP7	USCI0 数据1 管脚.
		QEI0_INDEX	I	MFP11	正交编码器0 索引输入
		EPWM0_CH3	I/O	MFP12	EPWM0 通道3输出/捕获输入脚
		BPWM0_CH2	I/O	MFP13	BPWM0 通道2 输出/捕获输入
	101	PE.3	I/O	MFP0	通用数字I/O脚
		EBI_MCLK	O	MFP2	EBI 外部时钟输出脚.
		SD0_DAT1	I/O	MFP3	SD/SDIO0 数据线位1
		SPIM_MISO	I/O	MFP4	SPIM MISO（主机输入，从机输出）脚.
		SPI3_MISO	I/O	MFP5	SPI3 MISO（主机输入，从机输出）脚.
		SC0_DAT	I/O	MFP6	智能卡0数据输入脚
		USCI0_DAT0	I/O	MFP7	USCI0 数据0脚
		QEI0_A	I	MFP11	正交编码器0 相位A输入
		EPWM0_CH4	I/O	MFP12	EPWM0 通道4输出/捕获输入脚
		BPWM0_CH1	I/O	MFP13	BPWM0 通道1 输出/捕获输入
	102	PE.2	I/O	MFP0	通用数字I/O脚
		EBI_ALE	O	MFP2	EBI地址锁存使能输出脚
		SD0_DAT0	I/O	MFP3	SD/SDIO0 数据线位0
		SPIM_MOSI	I/O	MFP4	SPIM MOSI (主机输出，从机输入) 管脚.
		SPI3_MOSI	I/O	MFP5	SPI3 MOSI (主机输出，从机输入) 管脚.
		SC0_CLK	O	MFP6	智能卡0时钟管脚
		USCI0_CLK	I/O	MFP7	USCI0 时钟脚
		QEI0_B	I	MFP11	正交编码器0 相位B 输入
		EPWM0_CH5	I/O	MFP12	EPWM0 通道5 输出/捕获输入
		BPWM0_CH0	I/O	MFP13	BPWM0 通道0 输出/捕获输入
	103	V_{SS}	P	MFP0	数字电路地管脚
	104	V_{DD}	P	MFP0	I/O和内部PLL及数字电路的LDO电源

64 管脚	128 管脚	管脚名称	类型	MFP	描述
	105	PE.1	I/O	MFP0	通用数字I/O脚
		EBI_AD10	I/O	MFP2	EBI地址/数据总线位10.
		QSPI0_MISO0	I/O	MFP3	QSPI0 MISO0（主机输入，从机输出）脚.
		SC2_DAT	I/O	MFP4	智能卡 2 数据管脚.
		I^{2}S0_BCLK	O	MFP5	I2S0位时钟输出管脚
		SPI1_MISO	I/O	MFP6	SPI1 MISO(主机输入，从机输出)管脚
		UART3_TXD	O	MFP7	UART3 数据发送输出脚.
		I^{2}C1_SCL	I/O	MFP8	I^{2}C1 时钟脚
		UART4_nCTS	I	MFP9	UART4 禁止发送输入管脚
	106	PE.0	I/O	MFP0	通用数字I/O脚
		EBI_AD11	I/O	MFP2	EBI地址/数据总线位11.
		QSPI0_MOSI0	I/O	MFP3	QSPI0 MOSI0 (主机输出，从机输入) 管脚.
		SC2_CLK	O	MFP4	智能卡 2 时钟脚
		I^{2}S0_MCLK	O	MFP5	I2S0 主时钟输出脚
		SPI1_MOSI	I/O	MFP6	SPI1 MOSI (主机输出，从机输入) 管脚
		UART3_RXD	I	MFP7	UART3 数据接收输入脚
		I^{2}C1_SDA	I/O	MFP8	I^{2}C1 数据输入/输出脚
		UART4_nRTS	O	MFP9	UART4 请求发送输出脚
	107	PH.8	I/O	MFP0	通用数字I/O脚
		EBI_AD12	I/O	MFP2	EBI地址/数据总线位12.
		QSPI0_CLK	I/O	MFP3	QSPI0 串行时钟脚
		SC2_PWR	O	MFP4	智能卡 2电源脚
		I^{2}S0_DI	I	MFP5	I2S0 数据输入脚
		SPI1_CLK	I/O	MFP6	SPI1 串行时钟管脚
		UART3_nRTS	O	MFP7	UART3 请求发送输出脚
		I^{2}C1_SMBAL	O	MFP8	I^{2}C1 SMBus SMBALTER 管脚
		I^{2}C2_SCL	I/O	MFP9	I^{2}C2 时钟脚
		UART1_TXD	O	MFP10	UART1 数据发送输出脚
	108	PH.9	I/O	MFP0	通用数字I/O脚
		EBI_AD13	I/O	MFP2	EBI地址/数据总线位13.
		QSPI0_SS	I/O	MFP3	QSPI0 从机选择脚

64 管脚	128 管脚	管脚名称	类型	MFP	描述
		SC2_RST	O	MFP4	智能卡 2 复位脚
		I²S0_DO	O	MFP5	I²S0 数据输出脚
		SPI1_SS	I/O	MFP6	SPI1 从机选择脚
		UART3_nCTS	I	MFP7	UART3 禁止发送输入管脚
		I²C1_SMBSUS	O	MFP8	I²C1 SMBus SMBSUS 管脚 (PMBus 控制管脚)
		I²C2_SDA	I/O	MFP9	I²C2 数据输入/输出脚
		UART1_RXD	I	MFP10	UART1数据接收输入管脚
	109	PH.10	I/O	MFP0	通用数字I/O脚
		EBI_AD14	I/O	MFP2	EBI地址/数据总线位14.
		QSPI0_MISO1	I/O	MFP3	QSPI0 MISO1（主机输入，从机输出）脚.
		SC2_nCD	I	MFP4	智能卡 2 卡检测管脚.
		I²S0_LRCK	O	MFP5	I²S0左右通道时钟输出脚.
		SPI1_I2SMCLK	I/O	MFP6	SPI1 I²S 主机时钟输出脚
		UART4_TXD	O	MFP7	UART4 数据发送输出脚.
		UART0_TXD	O	MFP8	UART0 数据发送输出脚.
	110	PH.11	I/O	MFP0	通用数字I/O脚
		EBI_AD15	I/O	MFP2	EBI地址/数据总线位15.
		QSPI0_MOSI1	I/O	MFP3	QSPI0 MOSI1 (主机输出，从机输入) 管脚.
		UART4_RXD	I	MFP7	UART4 数据接收输入脚
		UART0_RXD	I	MFP8	UART0 数据接收输入脚
		EPWM0_CH5	I/O	MFP11	EPWM0 通道5 输出/捕获输入
	111	PD.14	I/O	MFP0	通用数字I/O脚
		EBI_nCS0	O	MFP2	EBI片选0输出脚
		SPI3_I2SMCLK	I/O	MFP3	SPI3 I²S 主机时钟输出脚
		SC1_nCD	I	MFP4	智能卡 1 卡检测管脚.
		EPWM0_CH4	I/O	MFP11	EPWM0 通道4输出/捕获输入脚
49	112	V_{SS}	P	MFP0	数字电路地管脚
50	113	LDO_CAP	A	MFP0	LDO输出脚
51	114	V_{DD}	P	MFP0	I/O和内部PLL及数字电路的LDO电源
52	115	PC.14	I/O	MFP0	通用数字I/O脚
		EBI_AD11	I/O	MFP2	EBI地址/数据总线位11.

64 管脚	128 管脚	管脚名称	类型	MFP	描述
		SC1_nCD	I	MFP3	智能卡 1 卡检测管脚.
		SPI0_I2SMCLK	I/O	MFP4	SPI0 I^{2}S 主机时钟输出脚
		USCI0_CTL0	I/O	MFP5	USCI0 控制0 管脚.
		QSPI0_CLK	I/O	MFP6	QSPI0 串行时钟脚
		EPWM0_SYNC_IN	I	MFP11	EPWM0 计数器同步触发输入脚
		ETM_TRACE_CLK	I	MFP12	ETM接收追踪时钟输入管脚
		TM1	I/O	MFP13	Timer1 事件计数输入/反转输出脚
		USB_VBUS_ST	I	MFP14	USB 外部VBUS 电源状态管脚.
		HSUSB_VBUS_ST	I	MFP15	HSUSB 外部VBUS 电源状态管脚.
53	116	PB.15	I/O	MFP0	通用数字I/O脚
		EADC0_CH15	A	MFP1	EADC0 通道15 模拟输入
		EBI_AD12	I/O	MFP2	EBI地址/数据总线位12.
		SC1_PWR	O	MFP3	智能卡 1电源脚
		SPI0_SS	I/O	MFP4	SPI0 从机选择脚
		USCI0_CTL1	I/O	MFP5	USCI0 控制1 管脚.
		UART0_nCTS	I	MFP6	UART0 禁止发送输入管脚
		UART3_TXD	O	MFP7	UART3 数据发送输出脚.
		I^{2}C2_SMBAL	O	MFP8	I^{2}C2 SMBus SMBALTER 管脚
		EPWM1_CH0	I/O	MFP11	EPWM1 通道0 输出/捕获输入
		ETM_TRACE_DATA0	I	MFP12	ETM接收追踪数据0 输入管脚
		TM0_EXT	I/O	MFP13	Timer0 事件计数输入/反转输出管脚
		USB_VBUS_EN	O	MFP14	USB 外部 VBUS 电源使能管脚.
		HSUSB_VBUS_EN	O	MFP15	HSUSB 外部 VBUS 电源使能管脚.
54	117	PB.14	I/O	MFP0	通用数字I/O脚
		EADC0_CH14	A	MFP1	EADC0 通道14 模拟输入
		EBI_AD13	I/O	MFP2	EBI地址/数据总线位13.
		SC1_RST	O	MFP3	智能卡 1 复位脚
		SPI0_CLK	I/O	MFP4	SPI0 串行时钟脚
		USCI0_DAT1	I/O	MFP5	USCI0 数据1 管脚.
		UART0_nRTS	O	MFP6	UART0 请求发送输出脚
		UART3_RXD	I	MFP7	UART3 数据接收输入脚

64 管脚	128 管脚	管脚名称	类型	MFP	描述
		I²C2_SMBSUS	O	MFP8	I²C2 SMBus SMBSUS 管脚 (PMBus 控制管脚)
		EPWM1_CH1	I/O	MFP11	EPWM1 通道1 输出/捕获输入
		ETM_TRACE_DATA1	I	MFP12	ETM接收追踪数据1 输入管脚
		TM1_EXT	I/O	MFP13	Timer1 事件计数输入/反转输出脚
		CLKO	O	MFP14	时钟输出
55	118	PB.13	I/O	MFP0	通用数字I/O脚
		EADC0_CH13	A	MFP1	EADC0 通道13 模拟输入
		DAC1_OUT	A	MFP1	DAC1 模拟输出通道
		ACMP0_P3	A	MFP1	模拟比较器 0 正极输入 3 管脚.
		ACMP1_P3	A	MFP1	模拟比较器 1 正极输入 3 管脚.
		EBI_AD14	I/O	MFP2	EBI地址/数据总线位14.
		SC1_DAT	I/O	MFP3	智能卡 1 数据管脚.
		SPI0_MISO	I/O	MFP4	SPI0 MISO（主机输入，从机输出）脚.
		USCI0_DAT0	I/O	MFP5	USCI0 数据0脚
		UART0_TXD	O	MFP6	UART0 数据发送输出脚.
		UART3_nRTS	O	MFP7	UART3 请求发送输出脚
		I²C2_SCL	I/O	MFP8	I²C2 时钟脚
		EPWM1_CH2	I/O	MFP11	EPWM1 通道2 输出/捕获输入
		ETM_TRACE_DATA2	I	MFP12	ETM接收追踪数据2 输入管脚
		TM2_EXT	I/O	MFP13	Timer2 事件计数输入/反转输出脚
56	119	PB.12	I/O	MFP0	通用数字I/O脚
		EADC0_CH12	A	MFP1	EADC0 通道12 模拟输入
		DAC0_OUT	A	MFP1	DAC0 模拟输出通道
		ACMP0_P2	A	MFP1	模拟比较器 0 正极输入 2 管脚.
		ACMP1_P2	A	MFP1	模拟比较器 1 正极输入 2 管脚.
		EBI_AD15	I/O	MFP2	EBI地址/数据总线位15.
		SC1_CLK	O	MFP3	智能卡 1 时钟脚
		SPI0_MOSI	I/O	MFP4	SPI0 MOSI (主机输出，从机输入) 管脚.
		USCI0_CLK	I/O	MFP5	USCI0 时钟脚
		UART0_RXD	I	MFP6	UART0 数据接收输入脚
		UART3_nCTS	I	MFP7	UART3 禁止发送输入管脚

64 管脚	128 管脚	管脚名称	类型	MFP	描述
		I^2C2_SDA	I/O	MFP8	I^2C2 数据输入/输出脚
		SD0_nCD	I	MFP9	SD/SDIO0 卡检测输入管脚
		EPWM1_CH3	I/O	MFP11	EPWM1 通道3输出/捕获输入脚
		ETM_TRACE_DATA3	I	MFP12	ETM接收追踪数据3 输入管脚
		TM3_EXT	I/O	MFP13	Timer3 事件计数输入/反转输出脚
57	120	AV_{DD}	P	MFP0	内部模拟电路电源
58	121	V_{REF}	A	MFP0	ADC参考电压输入. 注: 此管脚需接1uF电容到地.
59	122	AV_{SS}	P	MFP0	模拟电路地管脚
60	123	PB.11	I/O	MFP0	通用数字I/O脚
		EADC0_CH11	A	MFP1	EADC0 通道11 模拟输入
		EBI_ADR16	O	MFP2	EBI 地址总线位 16.
		UART0_nCTS	I	MFP5	UART0 禁止发送输入管脚
		UART4_TXD	O	MFP6	UART4 数据发送输出脚.
		I^2C1_SCL	I/O	MFP7	I^2C1 时钟脚
		SPI0_I2SMCLK	I/O	MFP9	SPI0 I^2S 主机时钟输出脚
		BPWM1_CH0	I/O	MFP10	BPWM1 通道0 输出/捕获输入
		SPI3_CLK	I/O	MFP11	SPI3 串行时钟脚
		HSUSB_VBUS_ST	I	MFP14	HSUSB 外部VBUS 电源状态管脚.
61	124	PB.10	I/O	MFP0	通用数字I/O脚
		EADC0_CH10	A	MFP1	EADC0 通道10 模拟输入
		EBI_ADR17	O	MFP2	EBI 地址总线位 17.
		USCI1_CTL0	I/O	MFP4	USCI1 控制0管脚
		UART0_nRTS	O	MFP5	UART0 请求发送输出脚
		UART4_RXD	I	MFP6	UART4 数据接收输入脚
		I^2C1_SDA	I/O	MFP7	I^2C1 数据输入/输出脚
		BPWM1_CH1	I/O	MFP10	BPWM1 通道1 输出/捕获输入
		SPI3_SS	I/O	MFP11	SPI3 从机选择脚
		HSUSB_VBUS_EN	O	MFP14	HSUSB 外部 VBUS 电源使能管脚.
62	125	PB.9	I/O	MFP0	通用数字I/O脚
		EADC0_CH9	A	MFP1	EADC0 通道9 模拟输入

64 管脚	128 管脚	管脚名称	类型	MFP	描述
		EBI_ADR18	O	MFP2	EBI 地址总线位 18.
		USCI1_CTL1	I/O	MFP4	USCI1 控制1管脚
		UART0_TXD	O	MFP5	UART0 数据发送输出脚.
		UART1_nCTS	I	MFP6	UART1 禁止发送输入管脚
		I^2C1_SMBAL	O	MFP7	I^2C1 SMBus SMBALTER 管脚
		BPWM1_CH2	I/O	MFP10	BPWM1 通道2 输出/捕获输入
		SPI3_MISO	I/O	MFP11	SPI3 MISO（主机输入，从机输出）脚.
		INT7	I	MFP13	外部中断7 输入脚
63	126	PB.8	I/O	MFP0	通用数字I/O脚
		EADC0_CH8	A	MFP1	EADC0 通道8 模拟输入
		EBI_ADR19	O	MFP2	EBI 地址总线位 19.
		USCI1_CLK	I/O	MFP4	USCI1 时钟脚
		UART0_RXD	I	MFP5	UART0 数据接收输入脚
		UART1_nRTS	O	MFP6	UART1 请求发送输出脚
		I^2C1_SMBSUS	O	MFP7	I^2C1 SMBus SMBSUS 管脚 (PMBus 控制管脚)
		BPWM1_CH3	I/O	MFP10	BPWM1 通道3输出/捕获输入脚
		SPI3_MOSI	I/O	MFP11	SPI3 MOSI (主机输出，从机输入) 管脚.
		INT6	I	MFP13	外部中断6 输入脚
64	127	PB.7	I/O	MFP0	通用数字I/O脚
		EADC0_CH7	A	MFP1	EADC0 通道7 模拟输入
		EBI_nWRL	O	MFP2	EBI低字节写使能输出脚
		USCI1_DAT0	I/O	MFP4	USCI1 数据0脚
		UART1_TXD	O	MFP6	UART1 数据发送输出脚
		SD1_CMD	I/O	MFP7	SD/SDIO1 命令/响应脚
		EBI_nCS0	O	MFP8	EBI片选0输出脚
		BPWM1_CH4	I/O	MFP10	BPWM1 通道4输出/捕获输入脚
		EPWM1_BRAKE0	I	MFP11	EPWM1 刹车0 输入脚
		EPWM1_CH4	I/O	MFP12	EPWM1 通道4输出/捕获输入脚
		INT5	I	MFP13	外部中断5 输入脚
		USB_VBUS_ST	I	MFP14	USB 外部VBUS 电源状态管脚.
		ACMP0_O	O	MFP15	模拟比较器 0输出脚

64 管脚	128 管脚	管脚名称	类型	MFP	描述
1	128	PB.6	I/O	MFP0	通用数字I/O脚
		EADC0_CH6	A	MFP1	EADC0通道6模拟输入
		EBI_nWRH	O	MFP2	EBI 高字节写使能输出管脚
		USCI1_DAT1	I/O	MFP4	USCI1数据1管脚.
		UART1_RXD	I	MFP6	UART1数据接收输入管脚
		SD1_CLK	O	MFP7	SD/SDIO1 时钟输出管脚
		EBI_nCS1	O	MFP8	EBI片选1输出管脚
		BPWM1_CH5	I/O	MFP10	EPWM1 通道5输出/捕获输入
		EPWM1_BRAKE1	I	MFP11	EPWM1 刹车1输入管脚
		EPWM1_CH5	I/O	MFP12	EPWM1 通道5输出/捕获输入
		INT4	I	MFP13	外部中断4输入管脚
		USB_VBUS_EN	O	MFP14	USB 外部 VBUS 电源使能管脚.
		ACMP1_O	O	MFP15	模拟比较器1输出脚

4.2.5 M485系列管脚描述

64 管脚	128 管脚	管脚名称	类型	MFP	描述
2	1	PB.5	I/O	MFP0	通用数字I/O脚
		EADC0_CH5	A	MFP1	EADC0 通道5模拟输入
		ACMP1_N	A	MFP1	模拟比较器1负极输入管脚
		EBI_ADR0	O	MFP2	EBI地址总线位0
		SD0_DAT3	I/O	MFP3	SD/SDIO0数据线位3
		SPI1_MISO	I/O	MFP5	SPI1 MISO(主机输入，从机输出)管脚
		I^2C0_SCL	I/O	MFP6	I2C0 时钟管脚
		UART5_TXD	O	MFP7	UART5数据发送输出脚
		USCI1_CTL0	I/O	MFP8	USCI1 控制0管脚
		SC0_CLK	O	MFP9	智能卡0时钟管脚
		I^2S0_BCLK	O	MFP10	I2S0位时钟输出管脚
		EPWM0_CH0	I/O	MFP11	EPWM0 通道0输出/捕获输入
		TM0	I/O	MFP14	Timer0 事件计数输入/反转输出管脚
		INT0	I	MFP15	外部中断0输入管脚

64 管脚	128 管脚	管脚名称	类型	MFP	描述
3	2	PB.4	I/O	MFP0	通用数字I/O脚
		EADC0_CH4	A	MFP1	EADC0 通道4模拟输入
		ACMP1_P1	A	MFP1	模拟比较器1正极输入1脚
		EBI_ADR1	O	MFP2	EBI 地址总线位1
		SD0_DAT2	I/O	MFP3	SD/SDIO0 数据线位2
		SPI1_MOSI	I/O	MFP5	SPI1 MOSI (主机输出，从机输入) 管脚
		I^2C0_SDA	I/O	MFP6	I2C0 数据输入/输出管脚
		UART5_RXD	I	MFP7	UART5 数据接收输入管脚
		USCI1_CTL1	I/O	MFP8	USCI1 控制1管脚
		SC0_DAT	I/O	MFP9	智能卡0数据输入脚
		I^2S0_MCLK	O	MFP10	I2S0 主时钟输出脚
		EPWM0_CH1	I/O	MFP11	EPWM0 通道1输出/捕获输入脚
		TM1	I/O	MFP14	Timer1 事件计数输入/反转输出脚
		INT1	I	MFP15	外部中断1输入脚
4	3	PB.3	I/O	MFP0	通用数字I/O脚
		EADC0_CH3	A	MFP1	EADC0 通道3模拟输入
		ACMP0_N	A	MFP1	模拟比较器0负极输入脚
		EBI_ADR2	O	MFP2	EBI 地址总线位2
		SD0_DAT1	I/O	MFP3	SD/SDIO0 数据线位1
		SPI1_CLK	I/O	MFP5	SPI1 串行时钟管脚
		UART1_TXD	O	MFP6	UART1 数据发送输出脚
		UART5_nRTS	O	MFP7	UART5 请求发送输出脚
		USCI1_DAT1	I/O	MFP8	USCI1数据1管脚.
		SC0_RST	O	MFP9	智能卡0复位脚
		I^2S0_DI	I	MFP10	I2S0 数据输入脚
		EPWM0_CH2	I/O	MFP11	EPWM0 通道2输出/捕获输入脚
		TM2	I/O	MFP14	Timer2 事件计数输入/反转输出脚
		INT2	I	MFP15	外部中断2输入脚
5	4	PB.2	I/O	MFP0	通用数字I/O脚
		EADC0_CH2	A	MFP1	EADC0 通道2 模拟输入
		ACMP0_P1	A	MFP1	模拟比较器0正极输入1脚

64 管脚	128 管脚	管脚名称	类型	MFP	描述
		OPA0_O	A	MFP1	运放0输出脚
		EBI_ADR3	O	MFP2	EBI 地址总线位3
		SD0_DAT0	I/O	MFP3	SD/SDIO0 数据线位0
		SPI1_SS	I/O	MFP5	SPI1 从机选择脚
		UART1_RXD	I	MFP6	UART1数据接收输入管脚
		UART5_nCTS	I	MFP7	UART5 禁止发送输入管脚
		USCI1_DAT0	I/O	MFP8	USCI1 数据0脚
		SC0_PWR	O	MFP9	智能卡0电源脚
		I^2S0_DO	O	MFP10	I^2S0 数据输出脚
		EPWM0_CH3	I/O	MFP11	EPWM0 通道3输出/捕获输入脚
		TM3	I/O	MFP14	Timer3 事件计数输入/反转输出脚
		INT3	I	MFP15	外部中断3输入脚
	5	PC.12	I/O	MFP0	通用数字I/O脚
		EBI_ADR4	O	MFP2	EBI 地址总线位 4.
		UART0_TXD	O	MFP3	UART0 数据发送输出脚.
		I^2C0_SCL	I/O	MFP4	I2C0 时钟管脚
		SPI3_MISO	I/O	MFP6	SPI3 MISO（主机输入，从机输出）脚.
		SC0_nCD	I	MFP9	智能卡 0 卡检测管脚.
		ECAP1_IC2	I	MFP11	增强型捕获计数器 1 输入 2 管脚.
		EPWM1_CH0	I/O	MFP12	EPWM1 通道0 输出/捕获输入
		ACMP0_O	O	MFP14	模拟比较器 0输出脚
	6	PC.11	I/O	MFP0	通用数字I/O脚
		EBI_ADR5	O	MFP2	EBI 地址总线位 5.
		UART0_RXD	I	MFP3	UART0 数据接收输入脚
		I^2C0_SDA	I/O	MFP4	I2C0 数据输入/输出管脚
		SPI3_MOSI	I/O	MFP6	SPI3 MOSI (主机输出，从机输入) 管脚.
		ECAP1_IC1	I	MFP11	增强型捕获计数器 1 输入 1 管脚.
		EPWM1_CH1	I/O	MFP12	EPWM1 通道1 输出/捕获输入
		ACMP1_O	O	MFP14	模拟比较器1输出脚
	7	PC.10	I/O	MFP0	通用数字I/O脚
		EBI_ADR6	O	MFP2	EBI 地址总线位 6.

64 管脚	128 管脚	管脚名称	类型	MFP	描述
		SPI3_CLK	I/O	MFP6	SPI3 串行时钟脚
		UART3_TXD	O	MFP7	UART3 数据发送输出脚.
		CAN1_TXD	O	MFP9	CAN1总线发送输出
		ECAP1_IC0	I	MFP11	增强型捕获计数器 1 输入脚0
		EPWM1_CH2	I/O	MFP12	EPWM1 通道2 输出/捕获输入
	8	PC.9	I/O	MFP0	通用数字I/O脚
		EBI_ADR7	O	MFP2	EBI 地址总线位 7.
		SPI3_SS	I/O	MFP6	SPI3 从机选择脚
		UART3_RXD	I	MFP7	UART3 数据接收输入脚
		CAN1_RXD	I	MFP9	CAN1 总线接收输入
		EPWM1_CH3	I/O	MFP12	EPWM1 通道3输出/捕获输入脚
6	9	PB.1	I/O	MFP0	通用数字I/O脚
		EADC0_CH1	A	MFP1	EADC0 通道1 模拟输入
		OPA0_N	A	MFP1	运放0负极输入脚
		EBI_ADR8	O	MFP2	EBI 地址总线位 8.
		SD0_CLK	O	MFP3	SD/SDIO0 时钟输出脚
		SPI1_I2SMCLK	I/O	MFP5	SPI1 I^2S 主机时钟输出脚
		SPI3_I2SMCLK	I/O	MFP6	SPI3 I^2S 主机时钟输出脚
		UART2_TXD	O	MFP7	UART2 数据发送输出脚.
		USCI1_CLK	I/O	MFP8	USCI1 时钟脚
		I^2C1_SCL	I/O	MFP9	I^2C1 时钟脚
		I^2S0_LRCK	O	MFP10	I^2S0左右通道时钟输出脚.
		EPWM0_CH4	I/O	MFP11	EPWM0 通道4输出/捕获输入脚
		EPWM1_CH4	I/O	MFP12	EPWM1 通道4输出/捕获输入脚
		EPWM0_BRAKE0	I	MFP13	EPWM0 刹车0 输入脚
7	10	PB.0	I/O	MFP0	通用数字I/O脚
		EADC0_CH0	A	MFP1	EADC0 通道0 模拟输入
		OPA0_P	A	MFP1	运放 0 正极输入脚
		EBI_ADR9	O	MFP2	EBI 地址总线位 9.
		SD0_CMD	I/O	MFP3	SD/SDIO0 命令/响应脚
		UART2_RXD	I	MFP7	UART2 数据接收输入脚

64 管脚	128 管脚	管脚名称	类型	MFP	描述
		SPI0_I2SMCLK	I/O	MFP8	SPI0 I²S 主机时钟输出脚
		I²C1_SDA	I/O	MFP9	I²C1 数据输入/输出脚
		EPWM0_CH5	I/O	MFP11	EPWM0 通道5 输出/捕获输入
		EPWM1_CH5	I/O	MFP12	EPWM1 通道5输出/捕获输入
		EPWM0_BRAKE1	I	MFP13	EPWM0 刹车1 输入脚
	11	V_{SS}	P	MFP0	数字电路地管脚
	12	V_{DD}	P	MFP0	I/O和内部PLL及数字电路的LDO电源
8	13	PA.11	I/O	MFP0	通用数字I/O脚
		ACMP0_P0	A	MFP1	模拟比较器 0 正极输入脚0
		EBI_nRD	O	MFP2	EBI 读使能输出脚
		SC2_PWR	O	MFP3	智能卡 2电源脚
		SPI2_SS	I/O	MFP4	SPI2 从机选择脚
		SD1_DAT3	I/O	MFP5	SD/SDIO1 数据线位3.
		USCI0_CLK	I/O	MFP6	USCI0 时钟脚
		I²C2_SCL	I/O	MFP7	I²C2 时钟脚
		BPWM0_CH0	I/O	MFP9	BPWM0 通道0 输出/捕获输入
		EPWM0_SYNC_OUT	O	MFP10	EPWM0计数器同步触发输出脚
		TM0_EXT	I/O	MFP13	Timer0 事件计数输入/反转输出管脚
		DAC1_ST	I	MFP14	DAC1 外部触发输入
9	14	PA.10	I/O	MFP0	通用数字I/O脚
		ACMP1_P0	A	MFP1	模拟比较器 1 正极输入脚0
		OPA1_O	A	MFP1	运放 1输出脚
		EBI_nWR	O	MFP2	EBI 写使能输出脚
		SC2_RST	O	MFP3	智能卡 2 复位脚
		SPI2_CLK	I/O	MFP4	SPI2 串行时钟脚
		SD1_DAT2	I/O	MFP5	SD/SDIO1 数据线位2.
		USCI0_DAT0	I/O	MFP6	USCI0 数据0脚
		I²C2_SDA	I/O	MFP7	I²C2 数据输入/输出脚
		BPWM0_CH1	I/O	MFP9	BPWM0 通道1 输出/捕获输入
		QEI1_INDEX	I	MFP10	正交编码器1 索引输入
		ECAP0_IC0	I	MFP11	增强型捕获计数器 0 输入脚0

64 管脚	128 管脚	管脚名称	类型	MFP	描述
		TM1_EXT	I/O	MFP13	Timer1 事件计数输入/反转输出脚
		DAC0_ST	I	MFP14	DAC0 外部触发输入
		SWDH_CLK	O	MFP15	串行调试总线主时钟输出
10	15	PA.9	I/O	MFP0	通用数字I/O脚
		OPA1_N	A	MFP1	运放 1 负极输入脚
		EBI_MCLK	O	MFP2	EBI 外部时钟输出脚.
		SC2_DAT	I/O	MFP3	智能卡 2 数据管脚.
		SPI2_MISO	I/O	MFP4	SPI2 MISO（主机输入，从机输出）脚.
		SD1_DAT1	I/O	MFP5	SD/SDIO1 数据线位1.
		USCI0_DAT1	I/O	MFP6	USCI0 数据1 管脚.
		UART1_TXD	O	MFP7	UART1 数据发送输出脚
		BPWM0_CH2	I/O	MFP9	BPWM0 通道2 输出/捕获输入
		QEI1_A	I	MFP10	正交编码器1 相位A输入
		ECAP0_IC1	I	MFP11	增强型捕获计数器 0 输入 1 管脚.
		TM2_EXT	I/O	MFP13	Timer2 事件计数输入/反转输出脚
		SWDH_DAT	I/O	MFP15	串行调试总线主机数据输入/输出管脚
11	16	PA.8	I/O	MFP0	通用数字I/O脚
		OPA1_P	A	MFP1	运放 1 正极输入脚
		EBI_ALE	O	MFP2	EBI地址锁存使能输出脚
		SC2_CLK	O	MFP3	智能卡 2 时钟脚
		SPI2_MOSI	I/O	MFP4	SPI2 MOSI (主机输出，从机输入) 管脚.
		SD1_DAT0	I/O	MFP5	SD/SDIO1 数据线位0.
		USCI0_CTL1	I/O	MFP6	USCI0 控制1 管脚.
		UART1_RXD	I	MFP7	UART1数据接收输入管脚
		BPWM0_CH3	I/O	MFP9	BPWM0 通道3输出/捕获输入脚
		QEI1_B	I	MFP10	正交编码器1 相位B 输入
		ECAP0_IC2	I	MFP11	增强型捕获计数器 0 输入 2 管脚.
		TM3_EXT	I/O	MFP13	Timer3 事件计数输入/反转输出脚
		INT4	I	MFP15	外部中断4输入管脚
	17	PC.13	I/O	MFP0	通用数字I/O脚
		EBI_ADR10	O	MFP2	EBI 地址总线位 10.

64 管脚	128 管脚	管脚名称	类型	MFP	描述
		SC2_nCD	I	MFP3	智能卡 2 卡检测管脚.
		SPI2_I2SMCLK	I/O	MFP4	SPI2 I^2S 主机时钟输出脚
		CAN1_TXD	O	MFP5	CAN1总线发送输出
		USCI0_CTL0	I/O	MFP6	USCI0 控制0 管脚.
		UART2_TXD	O	MFP7	UART2 数据发送输出脚.
		BPWM0_CH4	I/O	MFP9	BPWM0 通道4输出/捕获输入脚
		CLKO	O	MFP13	时钟输出
		EADC0_ST	I	MFP14	EADC0 外部触发输入
	18	PD.12	I/O	MFP0	通用数字I/O脚
		OPA2_O	A	MFP1	运放 2输出脚
		EBI_nCS0	O	MFP2	EBI片选0输出脚
		CAN1_RXD	I	MFP5	CAN1 总线接收输入
		UART2_RXD	I	MFP7	UART2 数据接收输入脚
		BPWM0_CH5	I/O	MFP9	BPWM0 通道5 输出/捕获输入
		QEI0_INDEX	I	MFP10	正交编码器0 索引输入
		CLKO	O	MFP13	时钟输出
		EADC0_ST	I	MFP14	EADC0 外部触发输入
		INT5	I	MFP15	外部中断5 输入脚
	19	PD.11	I/O	MFP0	通用数字I/O脚
		OPA2_N	A	MFP1	运放 2 负极输入脚
		EBI_nCS1	O	MFP2	EBI片选1输出管脚
		UART1_TXD	O	MFP3	UART1 数据发送输出脚
		CAN0_TXD	O	MFP4	CAN0总线发送输出
		QEI0_A	I	MFP10	正交编码器0 相位A输入
		INT6	I	MFP15	外部中断6 输入脚
	20	PD.10	I/O	MFP0	通用数字I/O脚
		OPA2_P	A	MFP1	运放 2 正极输入脚
		EBI_nCS2	O	MFP2	EBI片选2输出脚
		UART1_RXD	I	MFP3	UART1数据接收输入管脚
		CAN0_RXD	I	MFP4	CAN0 总线接收输入
		QEI0_B	I	MFP10	正交编码器0 相位B 输入

64 管脚	128 管脚	管脚名称	类型	MFP	描述
		INT7	I	MFP15	外部中断7 输入脚
	21	PG.2	I/O	MFP0	通用数字I/O脚
		EBI_ADR11	O	MFP2	EBI 地址总线位 11.
		SPI2_SS	I/O	MFP3	SPI2 从机选择脚
		I^2C0_SMBAL	O	MFP4	I^2C0 SMBus SMBALTER 管脚
		I^2C1_SCL	I/O	MFP5	I^2C1 时钟脚
		TM0	I/O	MFP13	Timer0 事件计数输入/反转输出管脚
	22	PG.3	I/O	MFP0	通用数字I/O脚
		EBI_ADR12	O	MFP2	EBI 地址总线位 12.
		SPI2_CLK	I/O	MFP3	SPI2 串行时钟脚
		I^2C0_SMBSUS	O	MFP4	I^2C0 SMBus SMBSUS 管脚 (PMBus 控制管脚)
		I^2C1_SDA	I/O	MFP5	I^2C1 数据输入/输出脚
		TM1	I/O	MFP13	Timer1 事件计数输入/反转输出脚
	23	PG.4	I/O	MFP0	通用数字I/O脚
		EBI_ADR13	O	MFP2	EBI 地址总线位 13.
		SPI2_MISO	I/O	MFP3	SPI2 MISO（主机输入，从机输出）脚.
		TM2	I/O	MFP13	Timer2 事件计数输入/反转输出脚
	24	PF.11	I/O	MFP0	通用数字I/O脚
		EBI_ADR14	O	MFP2	EBI 地址总线位 14.
		SPI2_MOSI	I/O	MFP3	SPI2 MOSI (主机输出，从机输入) 管脚.
		TAMPER5	I/O	MFP10	TAMPER循环检测管脚 5.
		TM3	I/O	MFP13	Timer3 事件计数输入/反转输出脚
	25	PF.10	I/O	MFP0	通用数字I/O脚
		EBI_ADR15	O	MFP2	EBI 地址总线位 15.
		SC0_nCD	I	MFP3	智能卡 0 卡检测管脚.
		I^2S0_BCLK	O	MFP4	I2S0位时钟输出管脚
		SPI0_I2SMCLK	I/O	MFP5	SPI0 I^2S 主机时钟输出脚
		TAMPER4	I/O	MFP10	TAMPER循环检测管脚 4.
	26	PF.9	I/O	MFP0	通用数字I/O脚
		EBI_ADR16	O	MFP2	EBI 地址总线位 16.
		SC0_PWR	O	MFP3	智能卡0电源脚

64 管脚	128 管脚	管脚名称	类型	MFP	描述
		I^2S0_MCLK	O	MFP4	I2S0 主时钟输出脚
		SPI0_SS	I/O	MFP5	SPI0 从机选择脚
		TAMPER3	I/O	MFP10	TAMPER循环检测管脚 3.
	27	PF.8	I/O	MFP0	通用数字I/O脚
		EBI_ADR17	O	MFP2	EBI 地址总线位 17.
		SC0_RST	O	MFP3	智能卡0复位脚
		I^2S0_DI	I	MFP4	I2S0 数据输入脚
		SPI0_CLK	I/O	MFP5	SPI0 串行时钟脚
		TAMPER2	I/O	MFP10	TAMPER循环检测管脚 2.
	28	PF.7	I/O	MFP0	通用数字I/O脚
		EBI_ADR18	O	MFP2	EBI 地址总线位 18.
		SC0_DAT	I/O	MFP3	智能卡0数据输入脚
		I^2S0_DO	O	MFP4	I^2S0 数据输出脚
		SPI0_MISO	I/O	MFP5	SPI0 MISO（主机输入，从机输出）脚.
		UART4_TXD	O	MFP6	UART4 数据发送输出脚.
		TAMPER1	I/O	MFP10	TAMPER循环检测管脚 1.
12	29	PF.6	I/O	MFP0	通用数字I/O脚
		EBI_ADR19	O	MFP2	EBI 地址总线位 19.
		SC0_CLK	O	MFP3	智能卡0时钟管脚
		I^2S0_LRCK	O	MFP4	I^2S0左右通道时钟输出脚.
		SPI0_MOSI	I/O	MFP5	SPI0 MOSI (主机输出，从机输入) 管脚.
		UART4_RXD	I	MFP6	UART4 数据接收输入脚
		EBI_nCS0	O	MFP7	EBI片选0输出脚
		TAMPER0	I/O	MFP10	TAMPER循环检测管脚 0.
13	30	V_{DD}	P	MFP0	I/O和内部PLL及数字电路的LDO电源
14	31	PF.5	I/O	MFP0	通用数字I/O脚
		UART2_RXD	I	MFP2	UART2 数据接收输入脚
		UART2_nCTS	I	MFP4	UART2 禁止发送输入管脚
		BPWM0_CH4	I/O	MFP8	BPWM0 通道4输出/捕获输入脚
		EPWM0_SYNC_OUT	O	MFP9	EPWM0计数器同步触发输出脚
		X32_IN	I	MFP10	外部 32.768 kHz 晶振输入脚

64 管脚	128 管脚	管脚名称	类型	MFP	描述
		EADC0_ST	I	MFP11	EADC0 外部触发输入
15	32	PF.4	I/O	MFP0	通用数字I/O脚
		UART2_TXD	O	MFP2	UART2 数据发送输出脚.
		UART2_nRTS	O	MFP4	UART2 请求发送输出脚
		BPWM0_CH5	I/O	MFP8	BPWM0 通道5 输出/捕获输入
		X32_OUT	O	MFP10	外部 32.768 kHz 晶振输出脚
	33	PH.4	I/O	MFP0	通用数字I/O脚
		EBI_ADR3	O	MFP2	EBI 地址总线位3
		SPI1_MISO	I/O	MFP3	SPI1 MISO(主机输入，从机输出)管脚
	34	PH.5	I/O	MFP0	通用数字I/O脚
		EBI_ADR2	O	MFP2	EBI 地址总线位2
		SPI1_MOSI	I/O	MFP3	SPI1 MOSI (主机输出，从机输入) 管脚
	35	PH.6	I/O	MFP0	通用数字I/O脚
		EBI_ADR1	O	MFP2	EBI 地址总线位1
		SPI1_CLK	I/O	MFP3	SPI1 串行时钟管脚
	36	PH.7	I/O	MFP0	通用数字I/O脚
		EBI_ADR0	O	MFP2	EBI地址总线位0
		SPI1_SS	I/O	MFP3	SPI1 从机选择脚
16	37	PF.3	I/O	MFP0	通用数字I/O脚
		EBI_nCS0	O	MFP2	EBI片选0输出脚
		UART0_TXD	O	MFP3	UART0 数据发送输出脚.
		I^2C0_SCL	I/O	MFP4	I2C0 时钟管脚
		XT1_IN	I	MFP10	外部 4~24 MHz (高速) 晶振输入脚
		BPWM1_CH0	I/O	MFP11	BPWM1 通道0 输出/捕获输入
17	38	PF.2	I/O	MFP0	通用数字I/O脚
		EBI_nCS1	O	MFP2	EBI片选1输出管脚
		UART0_RXD	I	MFP3	UART0 数据接收输入脚
		I^2C0_SDA	I/O	MFP4	I2C0 数据输入/输出管脚
		QSPI0_CLK	I/O	MFP5	QSPI0 串行时钟脚
		XT1_OUT	O	MFP10	外部 4~24 MHz (高速) 晶振输出脚
		BPWM1_CH1	I/O	MFP11	BPWM1 通道1 输出/捕获输入

64 管脚	128 管脚	管脚名称	类型	MFP	描述
	39	V_{SS}	P	MFP0	数字电路地管脚
	40	V_{DD}	P	MFP0	I/O和内部PLL及数字电路的LDO电源
	41	PE.8	I/O	MFP0	通用数字I/O脚
		EBI_ADR10	O	MFP2	EBI 地址总线位 10.
		I^2S0_BCLK	O	MFP4	I2S0位时钟输出管脚
		SPI2_CLK	I/O	MFP5	SPI2 串行时钟脚
		USCI1_CTL1	I/O	MFP6	USCI1 控制1管脚
		UART2_TXD	O	MFP7	UART2 数据发送输出脚.
		EPWM0_CH0	I/O	MFP10	EPWM0 通道0输出/捕获输入
		EPWM0_BRAKE0	I	MFP11	EPWM0 刹车0 输入脚
		ECAP0_IC0	I	MFP12	增强型捕获计数器 0 输入脚0
		TRACE_CLK	O	MFP14	ETM跟踪时钟输出脚
	42	PE.9	I/O	MFP0	通用数字I/O脚
		EBI_ADR11	O	MFP2	EBI 地址总线位 11.
		I^2S0_MCLK	O	MFP4	I2S0 主时钟输出脚
		SPI2_MISO	I/O	MFP5	SPI2 MISO（主机输入，从机输出）脚.
		USCI1_CTL0	I/O	MFP6	USCI1 控制0管脚
		UART2_RXD	I	MFP7	UART2 数据接收输入脚
		EPWM0_CH1	I/O	MFP10	EPWM0 通道1输出/捕获输入脚
		EPWM0_BRAKE1	I	MFP11	EPWM0 刹车1 输入脚
		ECAP0_IC1	I	MFP12	增强型捕获计数器 0 输入 1 管脚.
		TRACE_DATA0	O	MFP14	ETM跟踪数据0 输出管脚
	43	PE.10	I/O	MFP0	通用数字I/O脚
		EBI_ADR12	O	MFP2	EBI 地址总线位 12.
		I^2S0_DI	I	MFP4	I2S0 数据输入脚
		SPI2_MOSI	I/O	MFP5	SPI2 MOSI (主机输出，从机输入) 管脚.
		USCI1_DAT0	I/O	MFP6	USCI1 数据0脚
		UART3_TXD	O	MFP7	UART3 数据发送输出脚.
		EPWM0_CH2	I/O	MFP10	EPWM0 通道2输出/捕获输入脚
		EPWM1_BRAKE0	I	MFP11	EPWM1 刹车0 输入脚
		ECAP0_IC2	I	MFP12	增强型捕获计数器 0 输入 2 管脚.

64 管脚	128 管脚	管脚名称	类型	MFP	描述
		TRACE_DATA1	O	MFP14	ETM跟踪数据1 输出管脚
	44	PE.11	I/O	MFP0	通用数字I/O脚
		EBI_ADR13	O	MFP2	EBI 地址总线位 13.
		I²S0_DO	O	MFP4	I²S0 数据输出脚
		SPI2_SS	I/O	MFP5	SPI2 从机选择脚
		USCI1_DAT1	I/O	MFP6	USCI1数据1管脚.
		UART3_RXD	I	MFP7	UART3 数据接收输入脚
		UART1_nCTS	I	MFP8	UART1 禁止发送输入管脚
		EPWM0_CH3	I/O	MFP10	EPWM0 通道3输出/捕获输入脚
		EPWM1_BRAKE1	I	MFP11	EPWM1 刹车1输入管脚
		ECAP1_IC2	I	MFP13	增强型捕获计数器 1 输入 2 管脚.
		TRACE_DATA2	O	MFP14	ETM跟踪数据2 输出管脚
	45	PE.12	I/O	MFP0	通用数字I/O脚
		EBI_ADR14	O	MFP2	EBI 地址总线位 14.
		I²S0_LRCK	O	MFP4	I²S0左右通道时钟输出脚.
		SPI2_I2SMCLK	I/O	MFP5	SPI2 I²S 主机时钟输出脚
		USCI1_CLK	I/O	MFP6	USCI1 时钟脚
		UART1_nRTS	O	MFP8	UART1 请求发送输出脚
		EPWM0_CH4	I/O	MFP10	EPWM0 通道4输出/捕获输入脚
		ECAP1_IC1	I	MFP13	增强型捕获计数器 1 输入 1 管脚.
		TRACE_DATA3	O	MFP14	ETM跟踪数据3 输出管脚
	46	PE.13	I/O	MFP0	通用数字I/O脚
		EBI_ADR15	O	MFP2	EBI 地址总线位 15.
		I²C0_SCL	I/O	MFP4	I2C0 时钟管脚
		UART4_nRTS	O	MFP5	UART4 请求发送输出脚
		UART1_TXD	O	MFP8	UART1 数据发送输出脚
		EPWM0_CH5	I/O	MFP10	EPWM0 通道5 输出/捕获输入
		EPWM1_CH0	I/O	MFP11	EPWM1 通道0 输出/捕获输入
		BPWM1_CH5	I/O	MFP12	EPWM1 通道5输出/捕获输入
		ECAP1_IC0	I	MFP13	增强型捕获计数器 1 输入脚0
	47	PC.8	I/O	MFP0	通用数字I/O脚

64 管脚	128 管脚	管脚名称	类型	MFP	描述
		EBI_ADR16	O	MFP2	EBI 地址总线位 16.
		I²C0_SDA	I/O	MFP4	I2C0 数据输入/输出管脚
		UART4_nCTS	I	MFP5	UART4 禁止发送输入管脚
		UART1_RXD	I	MFP8	UART1数据接收输入管脚
		EPWM1_CH1	I/O	MFP11	EPWM1 通道1 输出/捕获输入
		BPWM1_CH4	I/O	MFP12	BPWM1 通道4输出/捕获输入脚
18	48	PC.7	I/O	MFP0	通用数字I/O脚
		EBI_AD9	I/O	MFP2	EBI地址/数据总线位9.
		SPI1_MISO	I/O	MFP4	SPI1 MISO(主机输入，从机输出)管脚
		UART4_TXD	O	MFP5	UART4 数据发送输出脚.
		SC2_PWR	O	MFP6	智能卡 2电源脚
		UART0_nCTS	I	MFP7	UART0 禁止发送输入管脚
		I²C1_SMBAL	O	MFP8	I²C1 SMBus SMBALTER 管脚
		EPWM1_CH2	I/O	MFP11	EPWM1 通道2 输出/捕获输入
		BPWM1_CH0	I/O	MFP12	BPWM1 通道0 输出/捕获输入
		TM0	I/O	MFP14	Timer0 事件计数输入/反转输出管脚
		INT3	I	MFP15	外部中断3输入脚
19	49	PC.6	I/O	MFP0	通用数字I/O脚
		EBI_AD8	I/O	MFP2	EBI地址/数据总线位8.
		SPI1_MOSI	I/O	MFP4	SPI1 MOSI (主机输出，从机输入) 管脚
		UART4_RXD	I	MFP5	UART4 数据接收输入脚
		SC2_RST	O	MFP6	智能卡 2 复位脚
		UART0_nRTS	O	MFP7	UART0 请求发送输出脚
		I²C1_SMBSUS	O	MFP8	I²C1 SMBus SMBSUS 管脚 (PMBus 控制管脚)
		EPWM1_CH3	I/O	MFP11	EPWM1 通道3输出/捕获输入脚
		BPWM1_CH1	I/O	MFP12	BPWM1 通道1 输出/捕获输入
		TM1	I/O	MFP14	Timer1 事件计数输入/反转输出脚
		INT2	I	MFP15	外部中断2输入脚
20	50	PA.7	I/O	MFP0	通用数字I/O脚
		EBI_AD7	I/O	MFP2	EBI地址/数据总线位7.
		SPI1_CLK	I/O	MFP4	SPI1 串行时钟管脚

64 管脚	128 管脚	管脚名称	类型	MFP	描述
		SC2_DAT	I/O	MFP6	智能卡 2 数据管脚.
		UART0_TXD	O	MFP7	UART0 数据发送输出脚.
		I^2C1_SCL	I/O	MFP8	I^2C1 时钟脚
		EPWM1_CH4	I/O	MFP11	EPWM1 通道4输出/捕获输入脚
		BPWM1_CH2	I/O	MFP12	BPWM1 通道2 输出/捕获输入
		ACMP0_WLAT	I	MFP13	模拟比较器 0窗口锁存输入管脚
		TM2	I/O	MFP14	Timer2 事件计数输入/反转输出脚
		INT1	I	MFP15	外部中断1输入脚
21	51	PA.6	I/O	MFP0	通用数字I/O脚
		EBI_AD6	I/O	MFP2	EBI地址/数据总线位6.
		SPI1_SS	I/O	MFP4	SPI1 从机选择脚
		SD1_nCD	I	MFP5	SD/SDIO1 卡检测输入管脚
		SC2_CLK	O	MFP6	智能卡 2 时钟脚
		UART0_RXD	I	MFP7	UART0 数据接收输入脚
		I^2C1_SDA	I/O	MFP8	I^2C1 数据输入/输出脚
		EPWM1_CH5	I/O	MFP11	EPWM1 通道5输出/捕获输入
		BPWM1_CH3	I/O	MFP12	BPWM1 通道3输出/捕获输入脚
		ACMP1_WLAT	I	MFP13	模拟比较器 1窗口锁存输入管脚
		TM3	I/O	MFP14	Timer3 事件计数输入/反转输出脚
		INT0	I	MFP15	外部中断0输入管脚
22	52	V_{SS}	P	MFP0	数字电路地管脚
23	53	V_{DD}	P	MFP0	I/O和内部PLL及数字电路的LDO电源
24	54	LDO_CAP	A	MFP0	LDO输出脚
25	55	PA.5	I/O	MFP0	通用数字I/O脚
		SPIM_D2	I/O	MFP2	SPIM QuadI/O模式下数据2管脚
		QSPI0_MISO1	I/O	MFP3	QSPI0 MISO1（主机输入，从机输出）脚.
		SPI1_I2SMCLK	I/O	MFP4	SPI1 I^2S 主机时钟输出脚
		SD1_CMD	I/O	MFP5	SD/SDIO1 命令/响应脚
		SC2_nCD	I	MFP6	智能卡 2 卡检测管脚.
		UART0_nCTS	I	MFP7	UART0 禁止发送输入管脚
		UART5_TXD	O	MFP8	UART5数据发送输出脚

64 管脚	128 管脚	管脚名称	类型	MFP	描述
		I²C0_SCL	I/O	MFP9	I2C0 时钟管脚
		CAN0_TXD	O	MFP10	CAN0总线发送输出
		BPWM0_CH5	I/O	MFP12	BPWM0 通道5 输出/捕获输入
		EPWM0_CH0	I/O	MFP13	EPWM0 通道0输出/捕获输入
		QEI0_INDEX	I	MFP14	正交编码器0 索引输入
26	56	PA.4	I/O	MFP0	通用数字I/O脚
		SPIM_D3	I/O	MFP2	SPIM QuadI/O模式下数据3管脚
		QSPI0_MOSI1	I/O	MFP3	QSPI0 MOSI1 (主机输出，从机输入) 管脚.
		SPI0_I2SMCLK	I/O	MFP4	SPI0 I²S 主机时钟输出脚
		SD1_CLK	O	MFP5	SD/SDIO1 时钟输出管脚
		SC0_nCD	I	MFP6	智能卡 0 卡检测管脚.
		UART0_nRTS	O	MFP7	UART0 请求发送输出脚
		UART5_RXD	I	MFP8	UART5 数据接收输入管脚
		I²C0_SDA	I/O	MFP9	I2C0 数据输入/输出管脚
		CAN0_RXD	I	MFP10	CAN0 总线接收输入
		BPWM0_CH4	I/O	MFP12	BPWM0 通道4输出/捕获输入脚
		EPWM0_CH1	I/O	MFP13	EPWM0 通道1输出/捕获输入脚
		QEI0_A	I	MFP14	正交编码器0 相位A输入
27	57	PA.3	I/O	MFP0	通用数字I/O脚
		SPIM_SS	I/O	MFP2	SPIM 从机选择脚
		QSPI0_SS	I/O	MFP3	QSPI0 从机选择脚
		SPI0_SS	I/O	MFP4	SPI0 从机选择脚
		SD1_DAT3	I/O	MFP5	SD/SDIO1 数据线位3.
		SC0_PWR	O	MFP6	智能卡0电源脚
		UART4_TXD	O	MFP7	UART4 数据发送输出脚.
		UART1_TXD	O	MFP8	UART1 数据发送输出脚
		I²C1_SCL	I/O	MFP9	I²C1 时钟脚
		BPWM0_CH3	I/O	MFP12	BPWM0 通道3输出/捕获输入脚
		EPWM0_CH2	I/O	MFP13	EPWM0 通道2输出/捕获输入脚
		QEI0_B	I	MFP14	正交编码器0 相位B 输入
28	58	PA.2	I/O	MFP0	通用数字I/O脚

64 管脚	128 管脚	管脚名称	类型	MFP	描述
		SPIM_CLK	I/O	MFP2	SPIM 串行时钟脚
		QSPI0_CLK	I/O	MFP3	QSPI0 串行时钟脚
		SPI0_CLK	I/O	MFP4	SPI0 串行时钟脚
		SD1_DAT2	I/O	MFP5	SD/SDIO1 数据线位2.
		SC0_RST	O	MFP6	智能卡0复位脚
		UART4_RXD	I	MFP7	UART4 数据接收输入脚
		UART1_RXD	I	MFP8	UART1数据接收输入管脚
		I^2C1_SDA	I/O	MFP9	I^2C1 数据输入/输出脚
		BPWM0_CH2	I/O	MFP12	BPWM0 通道2 输出/捕获输入
		EPWM0_CH3	I/O	MFP13	EPWM0 通道3输出/捕获输入脚
29	59	PA.1	I/O	MFP0	通用数字I/O脚
		SPIM_MISO	I/O	MFP2	SPIM MISO（主机输入，从机输出）脚.
		QSPI0_MISO0	I/O	MFP3	QSPI0 MISO0（主机输入，从机输出）脚.
		SPI0_MISO	I/O	MFP4	SPI0 MISO（主机输入，从机输出）脚.
		SD1_DAT1	I/O	MFP5	SD/SDIO1 数据线位1.
		SC0_DAT	I/O	MFP6	智能卡0数据输入脚
		UART0_TXD	O	MFP7	UART0 数据发送输出脚.
		UART1_nCTS	I	MFP8	UART1 禁止发送输入管脚
		I^2C2_SCL	I/O	MFP9	I^2C2 时钟脚
		BPWM0_CH1	I/O	MFP12	BPWM0 通道1 输出/捕获输入
		EPWM0_CH4	I/O	MFP13	EPWM0 通道4输出/捕获输入脚
		DAC1_ST	I	MFP15	DAC1 外部触发输入
30	60	PA.0	I/O	MFP0	通用数字I/O脚
		SPIM_MOSI	I/O	MFP2	SPIM MOSI (主机输出，从机输入) 管脚.
		QSPI0_MOSI0	I/O	MFP3	QSPI0 MOSI0 (主机输出，从机输入) 管脚.
		SPI0_MOSI	I/O	MFP4	SPI0 MOSI (主机输出，从机输入) 管脚.
		SD1_DAT0	I/O	MFP5	SD/SDIO1 数据线位0.
		SC0_CLK	O	MFP6	智能卡0时钟管脚
		UART0_RXD	I	MFP7	UART0 数据接收输入脚
		UART1_nRTS	O	MFP8	UART1 请求发送输出脚
		I^2C2_SDA	I/O	MFP9	I^2C2 数据输入/输出脚

64 管脚	128 管脚	管脚名称	类型	MFP	描述
		BPWM0_CH0	I/O	MFP12	BPWM0 通道0 输出/捕获输入
		EPWM0_CH5	I/O	MFP13	EPWM0 通道5 输出/捕获输入
		DAC0_ST	I	MFP15	DAC0 外部触发输入
31	61	V_{DDIO}	P	MFP0	PE.1, PE.8~PE.13 的电源
	62	PE.14	I/O	MFP0	通用数字I/O脚
		EBI_AD8	I/O	MFP2	EBI地址/数据总线位8.
		UART2_TXD	O	MFP3	UART2 数据发送输出脚.
		CAN0_TXD	O	MFP4	CAN0总线发送输出
		SD1_nCD	I	MFP5	SD/SDIO1 卡检测输入管脚
	63	PE.15	I/O	MFP0	通用数字I/O脚
		EBI_AD9	I/O	MFP2	EBI地址/数据总线位9.
		UART2_RXD	I	MFP3	UART2 数据接收输入脚
		CAN0_RXD	I	MFP4	CAN0 总线接收输入
32	64	nRESET	I	MFP0	外部复位输入:低电平有效,有内部上拉，设置此管脚为低将复位初始状态
33	65	PF.0	I/O	MFP0	通用数字I/O脚
		UART1_TXD	O	MFP2	UART1 数据发送输出脚
		I^2C1_SCL	I/O	MFP3	I^2C1 时钟脚
		BPWM1_CH0	I/O	MFP12	BPWM1 通道0 输出/捕获输入
		ICE_DAT	O	MFP14	串行调试总线数据管脚.
34	66	PF.1	I/O	MFP0	通用数字I/O脚
		UART1_RXD	I	MFP2	UART1数据接收输入管脚
		I^2C1_SDA	I/O	MFP3	I^2C1 数据输入/输出脚
		BPWM1_CH1	I/O	MFP12	BPWM1 通道1 输出/捕获输入
		ICE_CLK	I	MFP14	串行调试总线时钟脚
	67	PD.9	I/O	MFP0	通用数字I/O脚
		EBI_AD7	I/O	MFP2	EBI地址/数据总线位7.
		I^2C2_SCL	I/O	MFP3	I^2C2 时钟脚
		UART2_nCTS	I	MFP4	UART2 禁止发送输入管脚
	68	PD.8	I/O	MFP0	通用数字I/O脚
		EBI_AD6	I/O	MFP2	EBI地址/数据总线位6.

64 管脚	128 管脚	管脚名称	类型	MFP	描述
		I²C2_SDA	I/O	MFP3	I²C2 数据输入/输出脚
		UART2_nRTS	O	MFP4	UART2 请求发送输出脚
35	69	PC.5	I/O	MFP0	通用数字I/O脚
		EBI_AD5	I/O	MFP2	EBI地址/数据总线位5.
		SPIM_D2	I/O	MFP3	SPIM QuadI/O模式下数据2管脚
		QSPI0_MISO1	I/O	MFP4	QSPI0 MISO1（主机输入，从机输出）脚.
		UART2_TXD	O	MFP8	UART2 数据发送输出脚.
		I²C1_SCL	I/O	MFP9	I²C1 时钟脚
		CAN0_TXD	O	MFP10	CAN0总线发送输出
		UART4_TXD	O	MFP11	UART4 数据发送输出脚.
		EPWM1_CH0	I/O	MFP12	EPWM1 通道0 输出/捕获输入
36	70	PC.4	I/O	MFP0	通用数字I/O脚
		EBI_AD4	I/O	MFP2	EBI地址/数据总线位4.
		SPIM_D3	I/O	MFP3	SPIM QuadI/O模式下数据3管脚
		QSPI0_MOSI1	I/O	MFP4	QSPI0 MOSI1 (主机输出，从机输入) 管脚.
		SC1_nCD	I	MFP5	智能卡 1 卡检测管脚.
		I²S0_BCLK	O	MFP6	I2S0位时钟输出管脚
		SPI1_I2SMCLK	I/O	MFP7	SPI1 I²S 主机时钟输出脚
		UART2_RXD	I	MFP8	UART2 数据接收输入脚
		I²C1_SDA	I/O	MFP9	I²C1 数据输入/输出脚
		CAN0_RXD	I	MFP10	CAN0 总线接收输入
		UART4_RXD	I	MFP11	UART4 数据接收输入脚
		EPWM1_CH1	I/O	MFP12	EPWM1 通道1 输出/捕获输入
37	71	PC.3	I/O	MFP0	通用数字I/O脚
		EBI_AD3	I/O	MFP2	EBI地址/数据总线位3.
		SPIM_SS	I/O	MFP3	SPIM 从机选择脚
		QSPI0_SS	I/O	MFP4	QSPI0 从机选择脚
		SC1_PWR	O	MFP5	智能卡 1电源脚
		I²S0_MCLK	O	MFP6	I2S0 主时钟输出脚
		SPI1_MISO	I/O	MFP7	SPI1 MISO(主机输入，从机输出)管脚
		UART2_nRTS	O	MFP8	UART2 请求发送输出脚

64 管脚	128 管脚	管脚名称	类型	MFP	描述
		I²C0_SMBAL	O	MFP9	I²C0 SMBus SMBALTER 管脚
		CAN1_TXD	O	MFP10	CAN1总线发送输出
		UART3_TXD	O	MFP11	UART3 数据发送输出脚.
		EPWM1_CH2	I/O	MFP12	EPWM1 通道2 输出/捕获输入
38	72	PC.2	I/O	MFP0	通用数字I/O脚
		EBI_AD2	I/O	MFP2	EBI地址/数据总线位2.
		SPIM_CLK	I/O	MFP3	SPIM 串行时钟脚
		QSPI0_CLK	I/O	MFP4	QSPI0 串行时钟脚
		SC1_RST	O	MFP5	智能卡 1 复位脚
		I²S0_DI	I	MFP6	I2S0 数据输入脚
		SPI1_MOSI	I/O	MFP7	SPI1 MOSI (主机输出，从机输入) 管脚
		UART2_nCTS	I	MFP8	UART2 禁止发送输入管脚
		I²C0_SMBSUS	O	MFP9	I²C0 SMBus SMBSUS 管脚 (PMBus 控制管脚)
		CAN1_RXD	I	MFP10	CAN1 总线接收输入
		UART3_RXD	I	MFP11	UART3 数据接收输入脚
		EPWM1_CH3	I/O	MFP12	EPWM1 通道3输出/捕获输入脚
39	73	PC.1	I/O	MFP0	通用数字I/O脚
		EBI_AD1	I/O	MFP2	EBI地址/数据总线位1.
		SPIM_MISO	I/O	MFP3	SPIM MISO（主机输入，从机输出）脚.
		QSPI0_MISO0	I/O	MFP4	QSPI0 MISO0（主机输入，从机输出）脚.
		SC1_DAT	I/O	MFP5	智能卡 1 数据管脚.
		I²S0_DO	O	MFP6	I²S0 数据输出脚
		SPI1_CLK	I/O	MFP7	SPI1 串行时钟管脚
		UART2_TXD	O	MFP8	UART2 数据发送输出脚.
		I²C0_SCL	I/O	MFP9	I2C0 时钟管脚
		EPWM1_CH4	I/O	MFP12	EPWM1 通道4输出/捕获输入脚
		ACMP0_O	O	MFP14	模拟比较器 0输出脚
40	74	PC.0	I/O	MFP0	通用数字I/O脚
		EBI_AD0	I/O	MFP2	EBI地址/数据总线位0.
		SPIM_MOSI	I/O	MFP3	SPIM MOSI (主机输出，从机输入) 管脚.
		QSPI0_MOSI0	I/O	MFP4	QSPI0 MOSI0 (主机输出，从机输入) 管脚.

64 管脚	128 管脚	管脚名称	类型	MFP	描述
		SC1_CLK	O	MFP5	智能卡 1 时钟脚
		I²S0_LRCK	O	MFP6	I²S0左右通道时钟输出脚.
		SPI1_SS	I/O	MFP7	SPI1 从机选择脚
		UART2_RXD	I	MFP8	UART2 数据接收输入脚
		I²C0_SDA	I/O	MFP9	I2C0 数据输入/输出管脚
		EPWM1_CH5	I/O	MFP12	EPWM1 通道5输出/捕获输入
		ACMP1_O	O	MFP14	模拟比较器1输出脚
	75	V_{SS}	P	MFP0	数字电路地管脚
	76	V_{DD}	P	MFP0	I/O和内部PLL及数字电路的LDO电源
	77	PG.9	I/O	MFP0	通用数字I/O脚
		EBI_AD0	I/O	MFP2	EBI地址/数据总线位0.
		SD1_DAT3	I/O	MFP3	SD/SDIO1 数据线位3.
		SPIM_D2	I/O	MFP4	SPIM QuadI/O模式下数据2管脚
		BPWM0_CH5	I/O	MFP12	BPWM0 通道5 输出/捕获输入
	78	PG.10	I/O	MFP0	通用数字I/O脚
		EBI_AD1	I/O	MFP2	EBI地址/数据总线位1.
		SD1_DAT2	I/O	MFP3	SD/SDIO1 数据线位2.
		SPIM_D3	I/O	MFP4	SPIM QuadI/O模式下数据3管脚
		BPWM0_CH4	I/O	MFP12	BPWM0 通道4输出/捕获输入脚
	79	PG.11	I/O	MFP0	通用数字I/O脚
		EBI_AD2	I/O	MFP2	EBI地址/数据总线位2.
		SD1_DAT1	I/O	MFP3	SD/SDIO1 数据线位1.
		SPIM_SS	I/O	MFP4	SPIM 从机选择脚
		BPWM0_CH3	I/O	MFP12	BPWM0 通道3输出/捕获输入脚
	80	PG.12	I/O	MFP0	通用数字I/O脚
		EBI_AD3	I/O	MFP2	EBI地址/数据总线位3.
		SD1_DAT0	I/O	MFP3	SD/SDIO1 数据线位0.
		SPIM_CLK	I/O	MFP4	SPIM 串行时钟脚
		BPWM0_CH2	I/O	MFP12	BPWM0 通道2 输出/捕获输入
	81	PG.13	I/O	MFP0	通用数字I/O脚
		EBI_AD4	I/O	MFP2	EBI地址/数据总线位4.

64 管脚	128 管脚	管脚名称	类型	MFP	描述
		SD1_CMD	I/O	MFP3	SD/SDIO1 命令/响应脚
		SPIM_MISO	I/O	MFP4	SPIM MISO（主机输入，从机输出）脚.
		BPWM0_CH1	I/O	MFP12	BPWM0 通道1 输出/捕获输入
	82	PG.14	I/O	MFP0	通用数字I/O脚
		EBI_AD5	I/O	MFP2	EBI地址/数据总线位5.
		SD1_CLK	O	MFP3	SD/SDIO1 时钟输出管脚
		SPIM_MOSI	I/O	MFP4	SPIM MOSI (主机输出，从机输入) 管脚.
		BPWM0_CH0	I/O	MFP12	BPWM0 通道0 输出/捕获输入
	83	PG.15	I/O	MFP0	通用数字I/O脚
		SD1_nCD	I	MFP3	SD/SDIO1 卡检测输入管脚
		CLKO	O	MFP14	时钟输出
		EADC0_ST	I	MFP15	EADC0 外部触发输入
	84	PD.13	I/O	MFP0	通用数字I/O脚
		EBI_AD10	I/O	MFP2	EBI地址/数据总线位10.
		SD0_nCD	I	MFP3	SD/SDIO0 卡检测输入管脚
		SPI0_I2SMCLK	I/O	MFP4	SPI0 I^2S 主机时钟输出脚
		SPI1_I2SMCLK	I/O	MFP5	SPI1 I^2S 主机时钟输出脚
		SC2_nCD	I	MFP7	智能卡 2 卡检测管脚.
	85	PA.12	I/O	MFP0	通用数字I/O脚
		I^2S0_BCLK	O	MFP2	I2S0位时钟输出管脚
		UART4_TXD	O	MFP3	UART4 数据发送输出脚.
		I^2C1_SCL	I/O	MFP4	I^2C1 时钟脚
		SPI2_SS	I/O	MFP5	SPI2 从机选择脚
		CAN0_TXD	O	MFP6	CAN0总线发送输出
		SC2_PWR	O	MFP7	智能卡 2电源脚
		BPWM1_CH2	I/O	MFP11	BPWM1 通道2 输出/捕获输入
		QEI1_INDEX	I	MFP12	正交编码器1 索引输入
		USB_VBUS	P	MFP14	USB主机或HUB的电源
	86	PA.13	I/O	MFP0	通用数字I/O脚
		I^2S0_MCLK	O	MFP2	I2S0 主时钟输出脚
		UART4_RXD	I	MFP3	UART4 数据接收输入脚

64 管脚	128 管脚	管脚名称	类型	MFP	描述
		I²C1_SDA	I/O	MFP4	I²C1 数据输入/输出脚
		SPI2_CLK	I/O	MFP5	SPI2 串行时钟脚
		CAN0_RXD	I	MFP6	CAN0 总线接收输入
		SC2_RST	O	MFP7	智能卡 2 复位脚
		BPWM1_CH3	I/O	MFP11	BPWM1 通道3输出/捕获输入脚
		QEI1_A	I	MFP12	正交编码器1 相位A输入
		USB_D-	A	MFP14	USB差分信号D+.
	87	PA.14	I/O	MFP0	通用数字I/O脚
		I²S0_DI	I	MFP2	I2S0 数据输入脚
		UART0_TXD	O	MFP3	UART0 数据发送输出脚.
		SPI2_MISO	I/O	MFP5	SPI2 MISO（主机输入，从机输出）脚.
		I²C2_SCL	I/O	MFP6	I²C2 时钟脚
		SC2_DAT	I/O	MFP7	智能卡 2 数据管脚.
		BPWM1_CH4	I/O	MFP11	BPWM1 通道4输出/捕获输入脚
		QEI1_B	I	MFP12	正交编码器1 相位B 输入
		USB_D+	A	MFP14	USB差分信号D-.
	88	PA.15	I/O	MFP0	通用数字I/O脚
		I²S0_DO	O	MFP2	I²S0 数据输出脚
		UART0_RXD	I	MFP3	UART0 数据接收输入脚
		SPI2_MOSI	I/O	MFP5	SPI2 MOSI (主机输出，从机输入) 管脚.
		I²C2_SDA	I/O	MFP6	I²C2 数据输入/输出脚
		SC2_CLK	O	MFP7	智能卡 2 时钟脚
		BPWM1_CH5	I/O	MFP11	EPWM1 通道5输出/捕获输入
		EPWM0_SYNC_IN	I	MFP12	EPWM0 计数器同步触发输入脚
		USB_OTG_ID	I	MFP14	USB_ 标识
41	89	HSUSB_VRES	A	MFP0	HSUSB模块参考电阻
42	90	HSUSB_V_{DD33}	P	MFP0	HSUSB VDD33 电源供电
43	91	HSUSB_VBUS	P	MFP0	HSUSB USB主机或HUB的电源
44	92	HSUSB_D-	A	MFP0	HSUSB差分信号D+.
45	93	HSUSB_VSS	P	MFP0	HSUSB 地管脚
46	94	HSUSB_D+	A	MFP0	HSUSB差分信号D-.

64 管脚	128 管脚	管脚名称	类型	MFP	描述
47	95	HSUSB_V_{DD12}_CAP	A	MFP0	HSUSB内部1.2v电压输出去耦管脚. 注: 此管脚需接1uF电容到地.
48	96	HSUSB_ID	I	MFP0	HSUSB 标识
	97	PE.7	I/O	MFP0	通用数字I/O脚
		SD0_CMD	I/O	MFP3	SD/SDIO0 命令/响应脚
		SPIM_D2	I/O	MFP4	SPIM QuadI/O模式下数据2管脚
		UART5_TXD	O	MFP8	UART5数据发送输出脚
		CAN1_TXD	O	MFP9	CAN1总线发送输出
		QEI1_INDEX	I	MFP11	正交编码器1 索引输入
		EPWM0_CH0	I/O	MFP12	EPWM0 通道0输出/捕获输入
		BPWM0_CH5	I/O	MFP13	BPWM0 通道5 输出/捕获输入
	98	PE.6	I/O	MFP0	通用数字I/O脚
		SD0_CLK	O	MFP3	SD/SDIO0 时钟输出脚
		SPIM_D3	I/O	MFP4	SPIM QuadI/O模式下数据3管脚
		SPI3_I2SMCLK	I/O	MFP5	SPI3 I^2S 主机时钟输出脚
		SC0_nCD	I	MFP6	智能卡 0 卡检测管脚.
		USCI0_CTL0	I/O	MFP7	USCI0 控制0 管脚.
		UART5_RXD	I	MFP8	UART5 数据接收输入管脚
		CAN1_RXD	I	MFP9	CAN1 总线接收输入
		QEI1_A	I	MFP11	正交编码器1 相位A输入
		EPWM0_CH1	I/O	MFP12	EPWM0 通道1输出/捕获输入脚
		BPWM0_CH4	I/O	MFP13	BPWM0 通道4输出/捕获输入脚
	99	PE.5	I/O	MFP0	通用数字I/O脚
		EBI_nRD	O	MFP2	EBI 读使能输出脚
		SD0_DAT3	I/O	MFP3	SD/SDIO0数据线位3
		SPIM_SS	I/O	MFP4	SPIM 从机选择脚
		SPI3_SS	I/O	MFP5	SPI3 从机选择脚
		SC0_PWR	O	MFP6	智能卡0电源脚
		USCI0_CTL1	I/O	MFP7	USCI0 控制1 管脚.
		QEI1_B	I	MFP11	正交编码器1 相位B 输入
		EPWM0_CH2	I/O	MFP12	EPWM0 通道2输出/捕获输入脚

64 管脚	128 管脚	管脚名称	类型	MFP	描述
		BPWM0_CH3	I/O	MFP13	BPWM0 通道3输出/捕获输入脚
	100	PE.4	I/O	MFP0	通用数字I/O脚
		EBI_nWR	O	MFP2	EBI 写使能输出脚
		SD0_DAT2	I/O	MFP3	SD/SDIO0 数据线位2
		SPIM_CLK	I/O	MFP4	SPIM 串行时钟脚
		SPI3_CLK	I/O	MFP5	SPI3 串行时钟脚
		SC0_RST	O	MFP6	智能卡0复位脚
		USCI0_DAT1	I/O	MFP7	USCI0 数据1 管脚.
		QEI0_INDEX	I	MFP11	正交编码器0 索引输入
		EPWM0_CH3	I/O	MFP12	EPWM0 通道3输出/捕获输入脚
		BPWM0_CH2	I/O	MFP13	BPWM0 通道2 输出/捕获输入
	101	PE.3	I/O	MFP0	通用数字I/O脚
		EBI_MCLK	O	MFP2	EBI 外部时钟输出脚.
		SD0_DAT1	I/O	MFP3	SD/SDIO0 数据线位1
		SPIM_MISO	I/O	MFP4	SPIM MISO（主机输入，从机输出）脚.
		SPI3_MISO	I/O	MFP5	SPI3 MISO（主机输入，从机输出）脚.
		SC0_DAT	I/O	MFP6	智能卡0数据输入脚
		USCI0_DAT0	I/O	MFP7	USCI0 数据0脚
		QEI0_A	I	MFP11	正交编码器0 相位A输入
		EPWM0_CH4	I/O	MFP12	EPWM0 通道4输出/捕获输入脚
		BPWM0_CH1	I/O	MFP13	BPWM0 通道1 输出/捕获输入
	102	PE.2	I/O	MFP0	通用数字I/O脚
		EBI_ALE	O	MFP2	EBI地址锁存使能输出脚
		SD0_DAT0	I/O	MFP3	SD/SDIO0 数据线位0
		SPIM_MOSI	I/O	MFP4	SPIM MOSI (主机输出，从机输入) 管脚.
		SPI3_MOSI	I/O	MFP5	SPI3 MOSI (主机输出，从机输入) 管脚.
		SC0_CLK	O	MFP6	智能卡0时钟管脚
		USCI0_CLK	I/O	MFP7	USCI0 时钟脚
		QEI0_B	I	MFP11	正交编码器0 相位B 输入
		EPWM0_CH5	I/O	MFP12	EPWM0 通道5 输出/捕获输入
		BPWM0_CH0	I/O	MFP13	BPWM0 通道0 输出/捕获输入

64 管脚	128 管脚	管脚名称	类型	MFP	描述
	103	V_{SS}	P	MFP0	数字电路地管脚
	104	V_{DD}	P	MFP0	I/O和内部PLL及数字电路的LDO电源
	105	PE.1	I/O	MFP0	通用数字I/O脚
		EBI_AD10	I/O	MFP2	EBI地址/数据总线位10.
		QSPI0_MISO0	I/O	MFP3	QSPI0 MISO0（主机输入，从机输出）脚.
		SC2_DAT	I/O	MFP4	智能卡 2 数据管脚.
		I^2S0_BCLK	O	MFP5	I2S0位时钟输出管脚
		SPI1_MISO	I/O	MFP6	SPI1 MISO(主机输入，从机输出)管脚
		UART3_TXD	O	MFP7	UART3 数据发送输出脚.
		I^2C1_SCL	I/O	MFP8	I^2C1 时钟脚
		UART4_nCTS	I	MFP9	UART4 禁止发送输入管脚
	106	PE.0	I/O	MFP0	通用数字I/O脚
		EBI_AD11	I/O	MFP2	EBI地址/数据总线位11.
		QSPI0_MOSI0	I/O	MFP3	QSPI0 MOSI0 (主机输出，从机输入) 管脚.
		SC2_CLK	O	MFP4	智能卡 2 时钟脚
		I^2S0_MCLK	O	MFP5	I2S0 主时钟输出脚
		SPI1_MOSI	I/O	MFP6	SPI1 MOSI (主机输出，从机输入) 管脚
		UART3_RXD	I	MFP7	UART3 数据接收输入脚
		I^2C1_SDA	I/O	MFP8	I^2C1 数据输入/输出脚
		UART4_nRTS	O	MFP9	UART4 请求发送输出脚
	107	PH.8	I/O	MFP0	通用数字I/O脚
		EBI_AD12	I/O	MFP2	EBI地址/数据总线位12.
		QSPI0_CLK	I/O	MFP3	QSPI0 串行时钟脚
		SC2_PWR	O	MFP4	智能卡 2电源脚
		I^2S0_DI	I	MFP5	I2S0 数据输入脚
		SPI1_CLK	I/O	MFP6	SPI1 串行时钟管脚
		UART3_nRTS	O	MFP7	UART3 请求发送输出脚
		I^2C1_SMBAL	O	MFP8	I^2C1 SMBus SMBALTER 管脚
		I^2C2_SCL	I/O	MFP9	I^2C2 时钟脚
		UART1_TXD	O	MFP10	UART1 数据发送输出脚
	108	PH.9	I/O	MFP0	通用数字I/O脚

64 管脚	128 管脚	管脚名称	类型	MFP	描述
		EBI_AD13	I/O	MFP2	EBI地址/数据总线位13.
		QSPI0_SS	I/O	MFP3	QSPI0 从机选择脚
		SC2_RST	O	MFP4	智能卡 2 复位脚
		I²S0_DO	O	MFP5	I²S0 数据输出脚
		SPI1_SS	I/O	MFP6	SPI1 从机选择脚
		UART3_nCTS	I	MFP7	UART3 禁止发送输入管脚
		I²C1_SMBSUS	O	MFP8	I²C1 SMBus SMBSUS 管脚 (PMBus 控制管脚)
		I²C2_SDA	I/O	MFP9	I²C2 数据输入/输出脚
		UART1_RXD	I	MFP10	UART1数据接收输入管脚
	109	PH.10	I/O	MFP0	通用数字I/O脚
		EBI_AD14	I/O	MFP2	EBI地址/数据总线位14.
		QSPI0_MISO1	I/O	MFP3	QSPI0 MISO1（主机输入，从机输出）脚.
		SC2_nCD	I	MFP4	智能卡 2 卡检测管脚.
		I²S0_LRCK	O	MFP5	I²S0左右通道时钟输出脚.
		SPI1_I2SMCLK	I/O	MFP6	SPI1 I²S 主机时钟输出脚
		UART4_TXD	O	MFP7	UART4 数据发送输出脚.
		UART0_TXD	O	MFP8	UART0 数据发送输出脚.
	110	PH.11	I/O	MFP0	通用数字I/O脚
		EBI_AD15	I/O	MFP2	EBI地址/数据总线位15.
		QSPI0_MOSI1	I/O	MFP3	QSPI0 MOSI1 (主机输出，从机输入) 管脚.
		UART4_RXD	I	MFP7	UART4 数据接收输入脚
		UART0_RXD	I	MFP8	UART0 数据接收输入脚
		EPWM0_CH5	I/O	MFP11	EPWM0 通道5 输出/捕获输入
	111	PD.14	I/O	MFP0	通用数字I/O脚
		EBI_nCS0	O	MFP2	EBI片选0输出脚
		SPI3_I2SMCLK	I/O	MFP3	SPI3 I²S 主机时钟输出脚
		SC1_nCD	I	MFP4	智能卡 1 卡检测管脚.
		EPWM0_CH4	I/O	MFP11	EPWM0 通道4输出/捕获输入脚
49	112	V_{SS}	P	MFP0	数字电路地管脚
50	113	LDO_CAP	A	MFP0	LDO输出脚
51	114	V_{DD}	P	MFP0	I/O和内部PLL及数字电路的LDO电源

64 管脚	128 管脚	管脚名称	类型	MFP	描述
52	115	PC.14	I/O	MFP0	通用数字I/O脚
		EBI_AD11	I/O	MFP2	EBI地址/数据总线位11.
		SC1_nCD	I	MFP3	智能卡 1 卡检测管脚.
		SPI0_I2SMCLK	I/O	MFP4	SPI0 I^2S 主机时钟输出脚
		USCI0_CTL0	I/O	MFP5	USCI0 控制0 管脚.
		QSPI0_CLK	I/O	MFP6	QSPI0 串行时钟脚
		EPWM0_SYNC_IN	I	MFP11	EPWM0 计数器同步触发输入脚
		ETM_TRACE_CLK	I	MFP12	ETM接收追踪时钟输入管脚
		TM1	I/O	MFP13	Timer1 事件计数输入/反转输出脚
		USB_VBUS_ST	I	MFP14	USB 外部VBUS 电源状态管脚.
		HSUSB_VBUS_ST	I	MFP15	HSUSB 外部VBUS 电源状态管脚.
53	116	PB.15	I/O	MFP0	通用数字I/O脚
		EADC0_CH15	A	MFP1	EADC0 通道15 模拟输入
		EBI_AD12	I/O	MFP2	EBI地址/数据总线位12.
		SC1_PWR	O	MFP3	智能卡 1电源脚
		SPI0_SS	I/O	MFP4	SPI0 从机选择脚
		USCI0_CTL1	I/O	MFP5	USCI0 控制1 管脚.
		UART0_nCTS	I	MFP6	UART0 禁止发送输入管脚
		UART3_TXD	O	MFP7	UART3 数据发送输出脚.
		I^2C2_SMBAL	O	MFP8	I^2C2 SMBus SMBALTER 管脚
		EPWM1_CH0	I/O	MFP11	EPWM1 通道0 输出/捕获输入
		ETM_TRACE_DATA0	I	MFP12	ETM接收追踪数据0 输入管脚
		TM0_EXT	I/O	MFP13	Timer0 事件计数输入/反转输出管脚
		USB_VBUS_EN	O	MFP14	USB 外部 VBUS 电源使能管脚.
		HSUSB_VBUS_EN	O	MFP15	HSUSB 外部 VBUS 电源使能管脚.
54	117	PB.14	I/O	MFP0	通用数字I/O脚
		EADC0_CH14	A	MFP1	EADC0 通道14 模拟输入
		EBI_AD13	I/O	MFP2	EBI地址/数据总线位13.
		SC1_RST	O	MFP3	智能卡 1 复位脚
		SPI0_CLK	I/O	MFP4	SPI0 串行时钟脚
		USCI0_DAT1	I/O	MFP5	USCI0 数据1 管脚.

64 管脚	128 管脚	管脚名称	类型	MFP	描述
		UART0_nRTS	O	MFP6	UART0 请求发送输出脚
		UART3_RXD	I	MFP7	UART3 数据接收输入脚
		I²C2_SMBSUS	O	MFP8	I²C2 SMBus SMBSUS 管脚 (PMBus 控制管脚)
		EPWM1_CH1	I/O	MFP11	EPWM1 通道1 输出/捕获输入
		ETM_TRACE_DATA1	I	MFP12	ETM接收追踪数据1 输入管脚
		TM1_EXT	I/O	MFP13	Timer1 事件计数输入/反转输出脚
		CLKO	O	MFP14	时钟输出
55	118	PB.13	I/O	MFP0	通用数字I/O脚
		EADC0_CH13	A	MFP1	EADC0 通道13 模拟输入
		DAC1_OUT	A	MFP1	DAC1 模拟输出通道
		ACMP0_P3	A	MFP1	模拟比较器 0 正极输入 3 管脚.
		ACMP1_P3	A	MFP1	模拟比较器 1 正极输入 3 管脚.
		EBI_AD14	I/O	MFP2	EBI地址/数据总线位14.
		SC1_DAT	I/O	MFP3	智能卡 1 数据管脚.
		SPI0_MISO	I/O	MFP4	SPI0 MISO（主机输入，从机输出）脚.
		USCI0_DAT0	I/O	MFP5	USCI0 数据0脚
		UART0_TXD	O	MFP6	UART0 数据发送输出脚.
		UART3_nRTS	O	MFP7	UART3 请求发送输出脚
		I²C2_SCL	I/O	MFP8	I²C2 时钟脚
		EPWM1_CH2	I/O	MFP11	EPWM1 通道2 输出/捕获输入
		ETM_TRACE_DATA2	I	MFP12	ETM接收追踪数据2 输入管脚
		TM2_EXT	I/O	MFP13	Timer2 事件计数输入/反转输出脚
56	119	PB.12	I/O	MFP0	通用数字I/O脚
		EADC0_CH12	A	MFP1	EADC0 通道12 模拟输入
		DAC0_OUT	A	MFP1	DAC0 模拟输出通道
		ACMP0_P2	A	MFP1	模拟比较器 0 正极输入 2 管脚.
		ACMP1_P2	A	MFP1	模拟比较器 1 正极输入 2 管脚.
		EBI_AD15	I/O	MFP2	EBI地址/数据总线位15.
		SC1_CLK	O	MFP3	智能卡 1 时钟脚
		SPI0_MOSI	I/O	MFP4	SPI0 MOSI (主机输出，从机输入) 管脚.
		USCI0_CLK	I/O	MFP5	USCI0 时钟脚

64 管脚	128 管脚	管脚名称	类型	MFP	描述
		UART0_RXD	I	MFP6	UART0 数据接收输入脚
		UART3_nCTS	I	MFP7	UART3 禁止发送输入管脚
		I^2C2_SDA	I/O	MFP8	I^2C2 数据输入/输出脚
		SD0_nCD	I	MFP9	SD/SDIO0 卡检测输入管脚
		EPWM1_CH3	I/O	MFP11	EPWM1 通道3输出/捕获输入脚
		ETM_TRACE_DATA3	I	MFP12	ETM接收追踪数据3 输入管脚
		TM3_EXT	I/O	MFP13	Timer3 事件计数输入/反转输出脚
57	120	AV_{DD}	P	MFP0	内部模拟电路电源
58	121	V_{REF}	A	MFP0	ADC参考电压输入. 注: 此管脚需接1uF电容到地.
59	122	AV_{SS}	P	MFP0	模拟电路地管脚
60	123	PB.11	I/O	MFP0	通用数字I/O脚
		EADC0_CH11	A	MFP1	EADC0 通道11 模拟输入
		EBI_ADR16	O	MFP2	EBI 地址总线位 16.
		UART0_nCTS	I	MFP5	UART0 禁止发送输入管脚
		UART4_TXD	O	MFP6	UART4 数据发送输出脚.
		I^2C1_SCL	I/O	MFP7	I^2C1 时钟脚
		CAN0_TXD	O	MFP8	CAN0总线发送输出
		SPI0_I2SMCLK	I/O	MFP9	SPI0 I^2S 主机时钟输出脚
		BPWM1_CH0	I/O	MFP10	BPWM1 通道0 输出/捕获输入
		SPI3_CLK	I/O	MFP11	SPI3 串行时钟脚
		HSUSB_VBUS_ST	I	MFP14	HSUSB 外部VBUS 电源状态管脚.
61	124	PB.10	I/O	MFP0	通用数字I/O脚
		EADC0_CH10	A	MFP1	EADC0 通道10 模拟输入
		EBI_ADR17	O	MFP2	EBI 地址总线位 17.
		USCI1_CTL0	I/O	MFP4	USCI1 控制0管脚
		UART0_nRTS	O	MFP5	UART0 请求发送输出脚
		UART4_RXD	I	MFP6	UART4 数据接收输入脚
		I^2C1_SDA	I/O	MFP7	I^2C1 数据输入/输出脚
		CAN0_RXD	I	MFP8	CAN0 总线接收输入
		BPWM1_CH1	I/O	MFP10	BPWM1 通道1 输出/捕获输入

64 管脚	128 管脚	管脚名称	类型	MFP	描述
		SPI3_SS	I/O	MFP11	SPI3 从机选择脚
		HSUSB_VBUS_EN	O	MFP14	HSUSB 外部 VBUS 电源使能管脚.
62	125	PB.9	I/O	MFP0	通用数字I/O脚
		EADC0_CH9	A	MFP1	EADC0 通道9 模拟输入
		EBI_ADR18	O	MFP2	EBI 地址总线位 18.
		USCI1_CTL1	I/O	MFP4	USCI1 控制1管脚
		UART0_TXD	O	MFP5	UART0 数据发送输出脚.
		UART1_nCTS	I	MFP6	UART1 禁止发送输入管脚
		I^2C1_SMBAL	O	MFP7	I^2C1 SMBus SMBALTER 管脚
		BPWM1_CH2	I/O	MFP10	BPWM1 通道2 输出/捕获输入
		SPI3_MISO	I/O	MFP11	SPI3 MISO（主机输入，从机输出）脚.
		INT7	I	MFP13	外部中断7 输入脚
63	126	PB.8	I/O	MFP0	通用数字I/O脚
		EADC0_CH8	A	MFP1	EADC0 通道8 模拟输入
		EBI_ADR19	O	MFP2	EBI 地址总线位 19.
		USCI1_CLK	I/O	MFP4	USCI1 时钟脚
		UART0_RXD	I	MFP5	UART0 数据接收输入脚
		UART1_nRTS	O	MFP6	UART1 请求发送输出脚
		I^2C1_SMBSUS	O	MFP7	I^2C1 SMBus SMBSUS 管脚 (PMBus 控制管脚)
		BPWM1_CH3	I/O	MFP10	BPWM1 通道3输出/捕获输入脚
		SPI3_MOSI	I/O	MFP11	SPI3 MOSI (主机输出，从机输入) 管脚.
		INT6	I	MFP13	外部中断6 输入脚
64	127	PB.7	I/O	MFP0	通用数字I/O脚
		EADC0_CH7	A	MFP1	EADC0 通道7 模拟输入
		EBI_nWRL	O	MFP2	EBI低字节写使能输出脚
		USCI1_DAT0	I/O	MFP4	USCI1 数据0脚
		CAN1_TXD	O	MFP5	CAN1总线发送输出
		UART1_TXD	O	MFP6	UART1 数据发送输出脚
		SD1_CMD	I/O	MFP7	SD/SDIO1 命令/响应脚
		EBI_nCS0	O	MFP8	EBI片选0输出脚
		BPWM1_CH4	I/O	MFP10	BPWM1 通道4输出/捕获输入脚

64 管脚	128 管脚	管脚名称	类型	MFP	描述
		EPWM1_BRAKE0	I	MFP11	EPWM1 刹车0 输入脚
		EPWM1_CH4	I/O	MFP12	EPWM1 通道4输出/捕获输入脚
		INT5	I	MFP13	外部中断5 输入脚
		USB_VBUS_ST	I	MFP14	USB 外部VBUS 电源状态管脚.
		ACMP0_O	O	MFP15	模拟比较器 0输出脚
1	128	PB.6	I/O	MFP0	通用数字I/O脚
		EADC0_CH6	A	MFP1	EADC0通道6模拟输入
		EBI_nWRH	O	MFP2	EBI 高字节写使能输出管脚
		USCI1_DAT1	I/O	MFP4	USCI1数据1管脚.
		CAN1_RXD	I	MFP5	CAN1 总线接收输入
		UART1_RXD	I	MFP6	UART1数据接收输入管脚
		SD1_CLK	O	MFP7	SD/SDIO1 时钟输出管脚
		EBI_nCS1	O	MFP8	EBI片选1输出管脚
		BPWM1_CH5	I/O	MFP10	EPWM1 通道5输出/捕获输入
		EPWM1_BRAKE1	I	MFP11	EPWM1 刹车1输入管脚
		EPWM1_CH5	I/O	MFP12	EPWM1 通道5输出/捕获输入
		INT4	I	MFP13	外部中断4输入管脚
		USB_VBUS_EN	O	MFP14	USB 外部 VBUS 电源使能管脚.
		ACMP1_O	O	MFP15	模拟比较器1输出脚

4.2.6 M487系列管脚描述

64管脚	128管脚	144管脚	管脚名称	类型	MFP	描述
2	1	1	PB.5	I/O	MFP0	通用数字I/O脚
			EADC0_CH5	A	MFP1	EADC0 通道5模拟输入
			ACMP1_N	A	MFP1	模拟比较器1负极输入管脚
			EBI_ADR0	O	MFP2	EBI地址总线位0
			SD0_DAT3	I/O	MFP3	SD/SDIO0数据线位3
			EMAC_RMII_REFCLK	I	MFP4	EMAC RMII 参考时钟输入脚
			SPI1_MISO	I/O	MFP5	SPI1 MISO(主机输入，从机输出)管脚
			I^2C0_SCL	I/O	MFP6	I2C0 时钟管脚
			UART5_TXD	O	MFP7	UART5数据发送输出脚
			USCI1_CTL0	I/O	MFP8	USCI1 控制0管脚
			SC0_CLK	O	MFP9	智能卡0时钟管脚
			I^2S0_BCLK	O	MFP10	I2S0位时钟输出管脚
			EPWM0_CH0	I/O	MFP11	EPWM0 通道0输出/捕获输入
			TM0	I/O	MFP14	Timer0 事件计数输入/反转输出管脚
			INT0	I	MFP15	外部中断0输入管脚
3	2	2	PB.4	I/O	MFP0	通用数字I/O脚
			EADC0_CH4	A	MFP1	EADC0 通道4模拟输入
			ACMP1_P1	A	MFP1	模拟比较器1正极输入1脚
			EBI_ADR1	O	MFP2	EBI 地址总线位1
			SD0_DAT2	I/O	MFP3	SD/SDIO0 数据线位2
			EMAC_RMII_RXD0	I	MFP4	EMAC RMII接收数据总线位 0.
			SPI1_MOSI	I/O	MFP5	SPI1 MOSI (主机输出，从机输入) 管脚
			I^2C0_SDA	I/O	MFP6	I2C0 数据输入/输出管脚
			UART5_RXD	I	MFP7	UART5 数据接收输入管脚
			USCI1_CTL1	I/O	MFP8	USCI1 控制1管脚
			SC0_DAT	I/O	MFP9	智能卡0数据输入脚
			I^2S0_MCLK	O	MFP10	I2S0 主时钟输出脚
			EPWM0_CH1	I/O	MFP11	EPWM0 通道1输出/捕获输入脚
			TM1	I/O	MFP14	Timer1 事件计数输入/反转输出脚

64 管脚	128 管脚	144 管脚	管脚名称	类型	MFP	描述
			INT1	I	MFP15	外部中断1输入脚
4	3	3	PB.3	I/O	MFP0	通用数字I/O脚
			EADC0_CH3	A	MFP1	EADC0 通道3模拟输入
			ACMP0_N	A	MFP1	模拟比较器0负极输入脚
			EBI_ADR2	O	MFP2	EBI 地址总线位2
			SD0_DAT1	I/O	MFP3	SD/SDIO0 数据线位1
			EMAC_RMII_RXD1	I	MFP4	EMAC RMII接收数据总线位 1.
			SPI1_CLK	I/O	MFP5	SPI1 串行时钟管脚
			UART1_TXD	O	MFP6	UART1 数据发送输出脚
			UART5_nRTS	O	MFP7	UART5 请求发送输出脚
			USCI1_DAT1	I/O	MFP8	USCI1数据1管脚.
			SC0_RST	O	MFP9	智能卡0复位脚
			I^2S0_DI	I	MFP10	I2S0 数据输入脚
			EPWM0_CH2	I/O	MFP11	EPWM0 通道2输出/捕获输入脚
			TM2	I/O	MFP14	Timer2 事件计数输入/反转输出脚
			INT2	I	MFP15	外部中断2输入脚
5	4	4	PB.2	I/O	MFP0	通用数字I/O脚
			EADC0_CH2	A	MFP1	EADC0 通道2 模拟输入
			ACMP0_P1	A	MFP1	模拟比较器0正极输入1脚
			OPA0_O	A	MFP1	运放0输出脚
			EBI_ADR3	O	MFP2	EBI 地址总线位3
			SD0_DAT0	I/O	MFP3	SD/SDIO0 数据线位0
			EMAC_RMII_CRSDV	I	MFP4	EMAC RMII 载波侦测/接收数据输入脚
			SPI1_SS	I/O	MFP5	SPI1 从机选择脚
			UART1_RXD	I	MFP6	UART1数据接收输入管脚
			UART5_nCTS	I	MFP7	UART5 禁止发送输入管脚
			USCI1_DAT0	I/O	MFP8	USCI1 数据0脚
			SC0_PWR	O	MFP9	智能卡0电源脚
			I^2S0_DO	O	MFP10	I^2S0 数据输出脚
			EPWM0_CH3	I/O	MFP11	EPWM0 通道3输出/捕获输入脚

64 管脚	128 管脚	144 管脚	管脚名称	类型	MFP	描述
			TM3	I/O	MFP14	Timer3 事件计数输入/反转输出脚
			INT3	I	MFP15	外部中断3输入脚
	5	5	PC.12	I/O	MFP0	通用数字I/O脚
			EBI_ADR4	O	MFP2	EBI 地址总线位 4.
			UART0_TXD	O	MFP3	UART0 数据发送输出脚.
			I^2C0_SCL	I/O	MFP4	I2C0 时钟管脚
			SPI3_MISO	I/O	MFP6	SPI3 MISO（主机输入，从机输出）脚.
			SC0_nCD	I	MFP9	智能卡 0 卡检测管脚.
			ECAP1_IC2	I	MFP11	增强型捕获计数器 1 输入 2 管脚.
			EPWM1_CH0	I/O	MFP12	EPWM1 通道0 输出/捕获输入
			ACMP0_O	O	MFP14	模拟比较器 0输出脚
	6	6	PC.11	I/O	MFP0	通用数字I/O脚
			EBI_ADR5	O	MFP2	EBI 地址总线位 5.
			UART0_RXD	I	MFP3	UART0 数据接收输入脚
			I^2C0_SDA	I/O	MFP4	I2C0 数据输入/输出管脚
			SPI3_MOSI	I/O	MFP6	SPI3 MOSI (主机输出，从机输入) 管脚.
			ECAP1_IC1	I	MFP11	增强型捕获计数器 1 输入 1 管脚.
			EPWM1_CH1	I/O	MFP12	EPWM1 通道1 输出/捕获输入
			ACMP1_O	O	MFP14	模拟比较器1输出脚
	7	7	PC.10	I/O	MFP0	通用数字I/O脚
			EBI_ADR6	O	MFP2	EBI 地址总线位 6.
			SPI3_CLK	I/O	MFP6	SPI3 串行时钟脚
			UART3_TXD	O	MFP7	UART3 数据发送输出脚.
			CAN1_TXD	O	MFP9	CAN1总线发送输出
			ECAP1_IC0	I	MFP11	增强型捕获计数器 1 输入脚0
			EPWM1_CH2	I/O	MFP12	EPWM1 通道2 输出/捕获输入
	8	8	PC.9	I/O	MFP0	通用数字I/O脚
			EBI_ADR7	O	MFP2	EBI 地址总线位 7.
			SPI3_SS	I/O	MFP6	SPI3 从机选择脚
			UART3_RXD	I	MFP7	UART3 数据接收输入脚

64 管脚	128 管脚	144 管脚	管脚名称	类型	MFP	描述
			CAN1_RXD	I	MFP9	CAN1 总线接收输入
			EPWM1_CH3	I/O	MFP12	EPWM1 通道3输出/捕获输入脚
6	9	9	PB.1	I/O	MFP0	通用数字I/O脚
			EADC0_CH1	A	MFP1	EADC0 通道1 模拟输入
			OPA0_N	A	MFP1	运放0负极输入脚
			EBI_ADR8	O	MFP2	EBI 地址总线位 8.
			SD0_CLK	O	MFP3	SD/SDIO0 时钟输出脚
			EMAC_RMII_RXERR	I	MFP4	EMAC RMII 接收数据错误输入脚
			SPI1_I2SMCLK	I/O	MFP5	SPI1 I^2S 主机时钟输出脚
			SPI3_I2SMCLK	I/O	MFP6	SPI3 I^2S 主机时钟输出脚
			UART2_TXD	O	MFP7	UART2 数据发送输出脚.
			USCI1_CLK	I/O	MFP8	USCI1 时钟脚
			I^2C1_SCL	I/O	MFP9	I^2C1 时钟脚
			I^2S0_LRCK	O	MFP10	I^2S0左右通道时钟输出脚.
			EPWM0_CH4	I/O	MFP11	EPWM0 通道4输出/捕获输入脚
			EPWM1_CH4	I/O	MFP12	EPWM1 通道4输出/捕获输入脚
			EPWM0_BRAKE0	I	MFP13	EPWM0 刹车0 输入脚
7	10	10	PB.0	I/O	MFP0	通用数字I/O脚
			EADC0_CH0	A	MFP1	EADC0 通道0 模拟输入
			OPA0_P	A	MFP1	运放 0 正极输入脚
			EBI_ADR9	O	MFP2	EBI 地址总线位 9.
			SD0_CMD	I/O	MFP3	SD/SDIO0 命令/响应脚
			UART2_RXD	I	MFP7	UART2 数据接收输入脚
			SPI0_I2SMCLK	I/O	MFP8	SPI0 I^2S 主机时钟输出脚
			I^2C1_SDA	I/O	MFP9	I^2C1 数据输入/输出脚
			EPWM0_CH5	I/O	MFP11	EPWM0 通道5 输出/捕获输入
			EPWM1_CH5	I/O	MFP12	EPWM1 通道5输出/捕获输入
			EPWM0_BRAKE1	I	MFP13	EPWM0 刹车1 输入脚
	11	11	V_{SS}	P	MFP0	数字电路地管脚
	12	12	V_{DD}	P	MFP0	I/O和内部PLL及数字电路的LDO电源

64 管脚	128 管脚	144 管脚	管脚名称	类型	MFP	描述
8	13	13	PA.11	I/O	MFP0	通用数字I/O脚
			ACMP0_P0	A	MFP1	模拟比较器 0 正极输入脚0
			EBI_nRD	O	MFP2	EBI 读使能输出脚
			SC2_PWR	O	MFP3	智能卡 2电源脚
			SPI2_SS	I/O	MFP4	SPI2 从机选择脚
			SD1_DAT3	I/O	MFP5	SD/SDIO1 数据线位3.
			USCI0_CLK	I/O	MFP6	USCI0 时钟脚
			I^2C2_SCL	I/O	MFP7	I^2C2 时钟脚
			BPWM0_CH0	I/O	MFP9	BPWM0 通道0 输出/捕获输入
			EPWM0_SYNC_OUT	O	MFP10	EPWM0计数器同步触发输出脚
			TM0_EXT	I/O	MFP13	Timer0 事件计数输入/反转输出管脚
			DAC1_ST	I	MFP14	DAC1 外部触发输入
9	14	14	PA.10	I/O	MFP0	通用数字I/O脚
			ACMP1_P0	A	MFP1	模拟比较器 1 正极输入脚0
			OPA1_O	A	MFP1	运放 1输出脚
			EBI_nWR	O	MFP2	EBI 写使能输出脚
			SC2_RST	O	MFP3	智能卡 2 复位脚
			SPI2_CLK	I/O	MFP4	SPI2 串行时钟脚
			SD1_DAT2	I/O	MFP5	SD/SDIO1 数据线位2.
			USCI0_DAT0	I/O	MFP6	USCI0 数据0脚
			I^2C2_SDA	I/O	MFP7	I^2C2 数据输入/输出脚
			BPWM0_CH1	I/O	MFP9	BPWM0 通道1 输出/捕获输入
			QEI1_INDEX	I	MFP10	正交编码器1 索引输入
			ECAP0_IC0	I	MFP11	增强型捕获计数器 0 输入脚0
			TM1_EXT	I/O	MFP13	Timer1 事件计数输入/反转输出脚
			DAC0_ST	I	MFP14	DAC0 外部触发输入
			SWDH_CLK	O	MFP15	串行调试总线主时钟输出
10	15	15	PA.9	I/O	MFP0	通用数字I/O脚
			OPA1_N	A	MFP1	运放 1 负极输入脚
			EBI_MCLK	O	MFP2	EBI 外部时钟输出脚.

64 管脚	128 管脚	144 管脚	管脚名称	类型	MFP	描述
			SC2_DAT	I/O	MFP3	智能卡 2 数据管脚.
			SPI2_MISO	I/O	MFP4	SPI2 MISO（主机输入，从机输出）脚.
			SD1_DAT1	I/O	MFP5	SD/SDIO1 数据线位1.
			USCI0_DAT1	I/O	MFP6	USCI0 数据1 管脚.
			UART1_TXD	O	MFP7	UART1 数据发送输出脚
			BPWM0_CH2	I/O	MFP9	BPWM0 通道2 输出/捕获输入
			QEI1_A	I	MFP10	正交编码器1 相位A输入
			ECAP0_IC1	I	MFP11	增强型捕获计数器 0 输入 1 管脚.
			TM2_EXT	I/O	MFP13	Timer2 事件计数输入/反转输出脚
			SWDH_DAT	I/O	MFP15	串行调试总线主机数据输入/输出管脚
11	16	16	PA.8	I/O	MFP0	通用数字I/O脚
			OPA1_P	A	MFP1	运放 1 正极输入脚
			EBI_ALE	O	MFP2	EBI地址锁存使能输出脚
			SC2_CLK	O	MFP3	智能卡 2 时钟脚
			SPI2_MOSI	I/O	MFP4	SPI2 MOSI (主机输出，从机输入) 管脚.
			SD1_DAT0	I/O	MFP5	SD/SDIO1 数据线位0.
			USCI0_CTL1	I/O	MFP6	USCI0 控制1 管脚.
			UART1_RXD	I	MFP7	UART1数据接收输入管脚
			BPWM0_CH3	I/O	MFP9	BPWM0 通道3输出/捕获输入脚
			QEI1_B	I	MFP10	正交编码器1 相位B 输入
			ECAP0_IC2	I	MFP11	增强型捕获计数器 0 输入 2 管脚.
			TM3_EXT	I/O	MFP13	Timer3 事件计数输入/反转输出脚
			INT4	I	MFP15	外部中断4输入管脚
	17	17	PC.13	I/O	MFP0	通用数字I/O脚
			EBI_ADR10	O	MFP2	EBI 地址总线位 10.
			SC2_nCD	I	MFP3	智能卡 2 卡检测管脚.
			SPI2_I2SMCLK	I/O	MFP4	SPI2 I^2S 主机时钟输出脚
			CAN1_TXD	O	MFP5	CAN1总线发送输出
			USCI0_CTL0	I/O	MFP6	USCI0 控制0 管脚.
			UART2_TXD	O	MFP7	UART2 数据发送输出脚.

64管脚	128管脚	144管脚	管脚名称	类型	MFP	描述
			BPWM0_CH4	I/O	MFP9	BPWM0 通道4输出/捕获输入脚
			CLKO	O	MFP13	时钟输出
			EADC0_ST	I	MFP14	EADC0 外部触发输入
	18	18	PD.12	I/O	MFP0	通用数字I/O脚
			OPA2_O	A	MFP1	运放 2输出脚
			EBI_nCS0	O	MFP2	EBI片选0输出脚
			CAN1_RXD	I	MFP5	CAN1 总线接收输入
			UART2_RXD	I	MFP7	UART2 数据接收输入脚
			BPWM0_CH5	I/O	MFP9	BPWM0 通道5 输出/捕获输入
			QEI0_INDEX	I	MFP10	正交编码器0 索引输入
			CLKO	O	MFP13	时钟输出
			EADC0_ST	I	MFP14	EADC0 外部触发输入
			INT5	I	MFP15	外部中断5 输入脚
	19	19	PD.11	I/O	MFP0	通用数字I/O脚
			OPA2_N	A	MFP1	运放 2 负极输入脚
			EBI_nCS1	O	MFP2	EBI片选1输出管脚
			UART1_TXD	O	MFP3	UART1 数据发送输出脚
			CAN0_TXD	O	MFP4	CAN0总线发送输出
			QEI0_A	I	MFP10	正交编码器0 相位A输入
			INT6	I	MFP15	外部中断6 输入脚
	20	20	PD.10	I/O	MFP0	通用数字I/O脚
			OPA2_P	A	MFP1	运放 2 正极输入脚
			EBI_nCS2	O	MFP2	EBI片选2输出脚
			UART1_RXD	I	MFP3	UART1数据接收输入管脚
			CAN0_RXD	I	MFP4	CAN0 总线接收输入
			QEI0_B	I	MFP10	正交编码器0 相位B 输入
			INT7	I	MFP15	外部中断7 输入脚
		21	V_{SS}	P	MFP0	数字电路地管脚
		22	V_{DD}	P	MFP0	I/O和内部PLL及数字电路的LDO电源
		23	PG.0	I/O	MFP0	通用数字I/O脚

64 管脚	128 管脚	144 管脚	管脚名称	类型	MFP	描述
			EBI_ADR8	O	MFP2	EBI 地址总线位 8.
			I²C0_SCL	I/O	MFP4	I2C0 时钟管脚
			I²C1_SMBAL	O	MFP5	I²C1 SMBus SMBALTER 管脚
			UART2_RXD	I	MFP6	UART2 数据接收输入脚
			CAN1_TXD	O	MFP7	CAN1总线发送输出
			UART1_TXD	O	MFP8	UART1 数据发送输出脚
		24	PG.1	I/O	MFP0	通用数字I/O脚
			EBI_ADR9	O	MFP2	EBI 地址总线位 9.
			SPI2_I2SMCLK	I/O	MFP3	SPI2 I²S 主机时钟输出脚
			I²C0_SDA	I/O	MFP4	I2C0 数据输入/输出管脚
			I²C1_SMBSUS	O	MFP5	I²C1 SMBus SMBSUS 管脚 (PMBus 控制管脚)
			UART2_TXD	O	MFP6	UART2 数据发送输出脚.
			CAN1_RXD	I	MFP7	CAN1 总线接收输入
			UART1_RXD	I	MFP8	UART1数据接收输入管脚
	21	25	PG.2	I/O	MFP0	通用数字I/O脚
			EBI_ADR11	O	MFP2	EBI 地址总线位 11.
			SPI2_SS	I/O	MFP3	SPI2 从机选择脚
			I²C0_SMBAL	O	MFP4	I²C0 SMBus SMBALTER 管脚
			I²C1_SCL	I/O	MFP5	I²C1 时钟脚
			TM0	I/O	MFP13	Timer0 事件计数输入/反转输出管脚
	22	26	PG.3	I/O	MFP0	通用数字I/O脚
			EBI_ADR12	O	MFP2	EBI 地址总线位 12.
			SPI2_CLK	I/O	MFP3	SPI2 串行时钟脚
			I²C0_SMBSUS	O	MFP4	I²C0 SMBus SMBSUS 管脚 (PMBus 控制管脚)
			I²C1_SDA	I/O	MFP5	I²C1 数据输入/输出脚
			TM1	I/O	MFP13	Timer1 事件计数输入/反转输出脚
	23	27	PG.4	I/O	MFP0	通用数字I/O脚
			EBI_ADR13	O	MFP2	EBI 地址总线位 13.
			SPI2_MISO	I/O	MFP3	SPI2 MISO（主机输入，从机输出）脚.
			TM2	I/O	MFP13	Timer2 事件计数输入/反转输出脚

64 管脚	128 管脚	144 管脚	管脚名称	类型	MFP	描述
	24	28	PF.11	I/O	MFP0	通用数字I/O脚
			EBI_ADR14	O	MFP2	EBI 地址总线位 14.
			SPI2_MOSI	I/O	MFP3	SPI2 MOSI (主机输出，从机输入) 管脚.
			TAMPER5	I/O	MFP10	TAMPER循环检测管脚 5.
			TM3	I/O	MFP13	Timer3 事件计数输入/反转输出脚
	25	29	PF.10	I/O	MFP0	通用数字I/O脚
			EBI_ADR15	O	MFP2	EBI 地址总线位 15.
			SC0_nCD	I	MFP3	智能卡 0 卡检测管脚.
			I^{2}S0_BCLK	O	MFP4	I2S0位时钟输出管脚
			SPI0_I2SMCLK	I/O	MFP5	SPI0 I^{2}S 主机时钟输出脚
			TAMPER4	I/O	MFP10	TAMPER循环检测管脚 4.
	26	30	PF.9	I/O	MFP0	通用数字I/O脚
			EBI_ADR16	O	MFP2	EBI 地址总线位 16.
			SC0_PWR	O	MFP3	智能卡0电源脚
			I^{2}S0_MCLK	O	MFP4	I2S0 主时钟输出脚
			SPI0_SS	I/O	MFP5	SPI0 从机选择脚
			TAMPER3	I/O	MFP10	TAMPER循环检测管脚 3.
	27	31	PF.8	I/O	MFP0	通用数字I/O脚
			EBI_ADR17	O	MFP2	EBI 地址总线位 17.
			SC0_RST	O	MFP3	智能卡0复位脚
			I^{2}S0_DI	I	MFP4	I2S0 数据输入脚
			SPI0_CLK	I/O	MFP5	SPI0 串行时钟脚
			TAMPER2	I/O	MFP10	TAMPER循环检测管脚 2.
	28	32	PF.7	I/O	MFP0	通用数字I/O脚
			EBI_ADR18	O	MFP2	EBI 地址总线位 18.
			SC0_DAT	I/O	MFP3	智能卡0数据输入脚
			I^{2}S0_DO	O	MFP4	I^{2}S0 数据输出脚
			SPI0_MISO	I/O	MFP5	SPI0 MISO（主机输入，从机输出）脚.
			UART4_TXD	O	MFP6	UART4 数据发送输出脚.
			TAMPER1	I/O	MFP10	TAMPER循环检测管脚 1.

64 管脚	128 管脚	144 管脚	管脚名称	类型	MFP	描述
12	29	33	PF.6	I/O	MFP0	通用数字I/O脚
			EBI_ADR19	O	MFP2	EBI 地址总线位 19.
			SC0_CLK	O	MFP3	智能卡0时钟管脚
			I^2S0_LRCK	O	MFP4	I^2S0左右通道时钟输出脚.
			SPI0_MOSI	I/O	MFP5	SPI0 MOSI (主机输出，从机输入) 管脚.
			UART4_RXD	I	MFP6	UART4 数据接收输入脚
			EBI_nCS0	O	MFP7	EBI片选0输出脚
			TAMPER0	I/O	MFP10	TAMPER循环检测管脚 0.
13	30	34	V_{DD}	P	MFP0	I/O和内部PLL及数字电路的LDO电源
14	31	35	PF.5	I/O	MFP0	通用数字I/O脚
			UART2_RXD	I	MFP2	UART2 数据接收输入脚
			UART2_nCTS	I	MFP4	UART2 禁止发送输入管脚
			BPWM0_CH4	I/O	MFP8	BPWM0 通道4输出/捕获输入脚
			EPWM0_SYNC_OUT	O	MFP9	EPWM0计数器同步触发输出脚
			X32_IN	I	MFP10	外部 32.768 kHz 晶振输入脚
			EADC0_ST	I	MFP11	EADC0 外部触发输入
15	32	36	PF.4	I/O	MFP0	通用数字I/O脚
			UART2_TXD	O	MFP2	UART2 数据发送输出脚.
			UART2_nRTS	O	MFP4	UART2 请求发送输出脚
			BPWM0_CH5	I/O	MFP8	BPWM0 通道5 输出/捕获输入
			X32_OUT	O	MFP10	外部 32.768 kHz 晶振输出脚
		37	PH.0	I/O	MFP0	通用数字I/O脚
			EBI_ADR7	O	MFP2	EBI 地址总线位 7.
			UART5_TXD	O	MFP4	UART5数据发送输出脚
			TM0_EXT	I/O	MFP13	Timer0 事件计数输入/反转输出管脚
		38	PH.1	I/O	MFP0	通用数字I/O脚
			EBI_ADR6	O	MFP2	EBI 地址总线位 6.
			UART5_RXD	I	MFP4	UART5 数据接收输入管脚
			TM1_EXT	I/O	MFP13	Timer1 事件计数输入/反转输出脚
		39	PH.2	I/O	MFP0	通用数字I/O脚

64 管脚	128 管脚	144 管脚	管脚名称	类型	MFP	描述
			EBI_ADR5	O	MFP2	EBI 地址总线位 5.
			UART5_nRTS	O	MFP4	UART5 请求发送输出脚
			UART4_TXD	O	MFP5	UART4 数据发送输出脚.
			I^{2}C0_SCL	I/O	MFP6	I2C0 时钟管脚
			TM2_EXT	I/O	MFP13	Timer2 事件计数输入/反转输出脚
		40	PH.3	I/O	MFP0	通用数字I/O脚
			EBI_ADR4	O	MFP2	EBI 地址总线位 4.
			SPI1_I2SMCLK	I/O	MFP3	SPI1 I^{2}S 主机时钟输出脚
			UART5_nCTS	I	MFP4	UART5 禁止发送输入管脚
			UART4_RXD	I	MFP5	UART4 数据接收输入脚
			I^{2}C0_SDA	I/O	MFP6	I2C0 数据输入/输出管脚
			TM3_EXT	I/O	MFP13	Timer3 事件计数输入/反转输出脚
	33	41	PH.4	I/O	MFP0	通用数字I/O脚
			EBI_ADR3	O	MFP2	EBI 地址总线位3
			SPI1_MISO	I/O	MFP3	SPI1 MISO(主机输入，从机输出)管脚
	34	42	PH.5	I/O	MFP0	通用数字I/O脚
			EBI_ADR2	O	MFP2	EBI 地址总线位2
			SPI1_MOSI	I/O	MFP3	SPI1 MOSI (主机输出，从机输入) 管脚
	35	43	PH.6	I/O	MFP0	通用数字I/O脚
			EBI_ADR1	O	MFP2	EBI 地址总线位1
			SPI1_CLK	I/O	MFP3	SPI1 串行时钟管脚
	36	44	PH.7	I/O	MFP0	通用数字I/O脚
			EBI_ADR0	O	MFP2	EBI地址总线位0
			SPI1_SS	I/O	MFP3	SPI1 从机选择脚
16	37	45	PF.3	I/O	MFP0	通用数字I/O脚
			EBI_nCS0	O	MFP2	EBI片选0输出脚
			UART0_TXD	O	MFP3	UART0 数据发送输出脚.
			I^{2}C0_SCL	I/O	MFP4	I2C0 时钟管脚
			XT1_IN	I	MFP10	外部 4~24 MHz (高速) 晶振输入脚
			BPWM1_CH0	I/O	MFP11	BPWM1 通道0 输出/捕获输入

64 管脚	128 管脚	144 管脚	管脚名称	类型	MFP	描述
17	38	46	PF.2	I/O	MFP0	通用数字I/O脚
			EBI_nCS1	O	MFP2	EBI片选1输出管脚
			UART0_RXD	I	MFP3	UART0 数据接收输入脚
			I^2C0_SDA	I/O	MFP4	I2C0 数据输入/输出管脚
			QSPI0_CLK	I/O	MFP5	QSPI0 串行时钟脚
			XT1_OUT	O	MFP10	外部 4~24 MHz (高速) 晶振输出脚
			BPWM1_CH1	I/O	MFP11	BPWM1 通道1 输出/捕获输入
	39	47	V_{SS}	P	MFP0	数字电路地管脚
	40	48	V_{DD}	P	MFP0	I/O和内部PLL及数字电路的LDO电源
	41	49	PE.8	I/O	MFP0	通用数字I/O脚
			EBI_ADR10	O	MFP2	EBI 地址总线位 10.
			EMAC_RMII_MDC	O	MFP3	EMAC RMII PHY管理时钟输出脚.
			I^2S0_BCLK	O	MFP4	I2S0位时钟输出管脚
			SPI2_CLK	I/O	MFP5	SPI2 串行时钟脚
			USCI1_CTL1	I/O	MFP6	USCI1 控制1管脚
			UART2_TXD	O	MFP7	UART2 数据发送输出脚.
			EPWM0_CH0	I/O	MFP10	EPWM0 通道0输出/捕获输入
			EPWM0_BRAKE0	I	MFP11	EPWM0 刹车0 输入脚
			ECAP0_IC0	I	MFP12	增强型捕获计数器 0 输入脚0
			TRACE_CLK	O	MFP14	ETM跟踪时钟输出脚
	42	50	PE.9	I/O	MFP0	通用数字I/O脚
			EBI_ADR11	O	MFP2	EBI 地址总线位 11.
			EMAC_RMII_MDIO	I/O	MFP3	EMAC RMII PHY管理数据管脚.
			I^2S0_MCLK	O	MFP4	I2S0 主时钟输出脚
			SPI2_MISO	I/O	MFP5	SPI2 MISO（主机输入，从机输出）脚.
			USCI1_CTL0	I/O	MFP6	USCI1 控制0管脚
			UART2_RXD	I	MFP7	UART2 数据接收输入脚
			EPWM0_CH1	I/O	MFP10	EPWM0 通道1输出/捕获输入脚
			EPWM0_BRAKE1	I	MFP11	EPWM0 刹车1 输入脚
			ECAP0_IC1	I	MFP12	增强型捕获计数器 0 输入 1 管脚.

64 管脚	128 管脚	144 管脚	管脚名称	类型	MFP	描述
			TRACE_DATA0	O	MFP14	ETM跟踪数据0 输出管脚
	43	51	PE.10	I/O	MFP0	通用数字I/O脚
			EBI_ADR12	O	MFP2	EBI 地址总线位 12.
			EMAC_RMII_TXD0	O	MFP3	EMAC RMII 发送数据总线位0.
			I^2S0_DI	I	MFP4	I2S0 数据输入脚
			SPI2_MOSI	I/O	MFP5	SPI2 MOSI (主机输出，从机输入) 管脚.
			USCI1_DAT0	I/O	MFP6	USCI1 数据0脚
			UART3_TXD	O	MFP7	UART3 数据发送输出脚.
			EPWM0_CH2	I/O	MFP10	EPWM0 通道2输出/捕获输入脚
			EPWM1_BRAKE0	I	MFP11	EPWM1 刹车0 输入脚
			ECAP0_IC2	I	MFP12	增强型捕获计数器 0 输入 2 管脚.
			TRACE_DATA1	O	MFP14	ETM跟踪数据1 输出管脚
	44	52	PE.11	I/O	MFP0	通用数字I/O脚
			EBI_ADR13	O	MFP2	EBI 地址总线位 13.
			EMAC_RMII_TXD1	O	MFP3	EMAC RMII 发送数据总线位1.
			I^2S0_DO	O	MFP4	I^2S0 数据输出脚
			SPI2_SS	I/O	MFP5	SPI2 从机选择脚
			USCI1_DAT1	I/O	MFP6	USCI1数据1管脚.
			UART3_RXD	I	MFP7	UART3 数据接收输入脚
			UART1_nCTS	I	MFP8	UART1 禁止发送输入管脚
			EPWM0_CH3	I/O	MFP10	EPWM0 通道3输出/捕获输入脚
			EPWM1_BRAKE1	I	MFP11	EPWM1 刹车1输入管脚
			ECAP1_IC2	I	MFP13	增强型捕获计数器 1 输入 2 管脚.
			TRACE_DATA2	O	MFP14	ETM跟踪数据2 输出管脚
	45	53	PE.12	I/O	MFP0	通用数字I/O脚
			EBI_ADR14	O	MFP2	EBI 地址总线位 14.
			EMAC_RMII_TXEN	O	MFP3	EMAC RMII 发送使能输出脚
			I^2S0_LRCK	O	MFP4	I^2S0左右通道时钟输出脚.
			SPI2_I2SMCLK	I/O	MFP5	SPI2 I^2S 主机时钟输出脚
			USCI1_CLK	I/O	MFP6	USCI1 时钟脚

64 管脚	128 管脚	144 管脚	管脚名称	类型	MFP	描述
			UART1_nRTS	O	MFP8	UART1 请求发送输出脚
			EPWM0_CH4	I/O	MFP10	EPWM0 通道4输出/捕获输入脚
			ECAP1_IC1	I	MFP13	增强型捕获计数器 1 输入 1 管脚.
			TRACE_DATA3	O	MFP14	ETM跟踪数据3 输出管脚
	46	54	PE.13	I/O	MFP0	通用数字I/O脚
			EBI_ADR15	O	MFP2	EBI 地址总线位 15.
			EMAC_PPS	O	MFP3	EMAC 每秒脉冲输出脚
			I^2C0_SCL	I/O	MFP4	I2C0 时钟管脚
			UART4_nRTS	O	MFP5	UART4 请求发送输出脚
			UART1_TXD	O	MFP8	UART1 数据发送输出脚
			EPWM0_CH5	I/O	MFP10	EPWM0 通道5 输出/捕获输入
			EPWM1_CH0	I/O	MFP11	EPWM1 通道0 输出/捕获输入
			BPWM1_CH5	I/O	MFP12	EPWM1 通道5输出/捕获输入
			ECAP1_IC0	I	MFP13	增强型捕获计数器 1 输入脚0
	47	55	PC.8	I/O	MFP0	通用数字I/O脚
			EBI_ADR16	O	MFP2	EBI 地址总线位 16.
			EMAC_RMII_REFCLK	I	MFP3	EMAC RMII 参考时钟输入脚
			I^2C0_SDA	I/O	MFP4	I2C0 数据输入/输出管脚
			UART4_nCTS	I	MFP5	UART4 禁止发送输入管脚
			UART1_RXD	I	MFP8	UART1数据接收输入管脚
			EPWM1_CH1	I/O	MFP11	EPWM1 通道1 输出/捕获输入
			BPWM1_CH4	I/O	MFP12	BPWM1 通道4输出/捕获输入脚
18	48	56	PC.7	I/O	MFP0	通用数字I/O脚
			EBI_AD9	I/O	MFP2	EBI地址/数据总线位9.
			EMAC_RMII_RXD0	I	MFP3	EMAC RMII接收数据总线位 0.
			SPI1_MISO	I/O	MFP4	SPI1 MISO(主机输入，从机输出)管脚
			UART4_TXD	O	MFP5	UART4 数据发送输出脚.
			SC2_PWR	O	MFP6	智能卡 2电源脚
			UART0_nCTS	I	MFP7	UART0 禁止发送输入管脚
			I^2C1_SMBAL	O	MFP8	I^2C1 SMBus SMBALTER 管脚

64 管脚	128 管脚	144 管脚	管脚名称	类型	MFP	描述
			EPWM1_CH2	I/O	MFP11	EPWM1 通道2 输出/捕获输入
			BPWM1_CH0	I/O	MFP12	BPWM1 通道0 输出/捕获输入
			TM0	I/O	MFP14	Timer0 事件计数输入/反转输出管脚
			INT3	I	MFP15	外部中断3输入脚
19	49	57	PC.6	I/O	MFP0	通用数字I/O脚
			EBI_AD8	I/O	MFP2	EBI地址/数据总线位8.
			EMAC_RMII_RXD1	I	MFP3	EMAC RMII接收数据总线位 1.
			SPI1_MOSI	I/O	MFP4	SPI1 MOSI (主机输出，从机输入) 管脚
			UART4_RXD	I	MFP5	UART4 数据接收输入脚
			SC2_RST	O	MFP6	智能卡 2 复位脚
			UART0_nRTS	O	MFP7	UART0 请求发送输出脚
			I^2C1_SMBSUS	O	MFP8	I^2C1 SMBus SMBSUS 管脚 (PMBus 控制管脚)
			EPWM1_CH3	I/O	MFP11	EPWM1 通道3输出/捕获输入脚
			BPWM1_CH1	I/O	MFP12	BPWM1 通道1 输出/捕获输入
			TM1	I/O	MFP14	Timer1 事件计数输入/反转输出脚
			INT2	I	MFP15	外部中断2输入脚
20	50	58	PA.7	I/O	MFP0	通用数字I/O脚
			EBI_AD7	I/O	MFP2	EBI地址/数据总线位7.
			EMAC_RMII_CRSDV	I	MFP3	EMAC RMII 载波侦测/接收数据输入脚
			SPI1_CLK	I/O	MFP4	SPI1 串行时钟管脚
			SC2_DAT	I/O	MFP6	智能卡 2 数据管脚.
			UART0_TXD	O	MFP7	UART0 数据发送输出脚.
			I^2C1_SCL	I/O	MFP8	I^2C1 时钟脚
			EPWM1_CH4	I/O	MFP11	EPWM1 通道4输出/捕获输入脚
			BPWM1_CH2	I/O	MFP12	BPWM1 通道2 输出/捕获输入
			ACMP0_WLAT	I	MFP13	模拟比较器 0窗口锁存输入管脚
			TM2	I/O	MFP14	Timer2 事件计数输入/反转输出脚
			INT1	I	MFP15	外部中断1输入脚
21	51	59	PA.6	I/O	MFP0	通用数字I/O脚
			EBI_AD6	I/O	MFP2	EBI地址/数据总线位6.

64 管脚	128 管脚	144 管脚	管脚名称	类型	MFP	描述
			EMAC_RMII_RXERR	I	MFP3	EMAC RMII 接收数据错误输入脚
			SPI1_SS	I/O	MFP4	SPI1 从机选择脚
			SD1_nCD	I	MFP5	SD/SDIO1 卡检测输入管脚
			SC2_CLK	O	MFP6	智能卡 2 时钟脚
			UART0_RXD	I	MFP7	UART0 数据接收输入脚
			I^2C1_SDA	I/O	MFP8	I^2C1 数据输入/输出脚
			EPWM1_CH5	I/O	MFP11	EPWM1 通道5输出/捕获输入
			BPWM1_CH3	I/O	MFP12	BPWM1 通道3输出/捕获输入脚
			ACMP1_WLAT	I	MFP13	模拟比较器 1窗口锁存输入管脚
			TM3	I/O	MFP14	Timer3 事件计数输入/反转输出脚
			INT0	I	MFP15	外部中断0输入管脚
22	52	60	V_{SS}	P	MFP0	数字电路地管脚
23	53	61	V_{DD}	P	MFP0	I/O和内部PLL及数字电路的LDO电源
24	54	62	LDO_CAP	A	MFP0	LDO输出脚
25	55	63	PA.5	I/O	MFP0	通用数字I/O脚
			SPIM_D2	I/O	MFP2	SPIM QuadI/O模式下数据2管脚
			QSPI0_MISO1	I/O	MFP3	QSPI0 MISO1（主机输入，从机输出）脚.
			SPI1_I2SMCLK	I/O	MFP4	SPI1 I^2S 主机时钟输出脚
			SD1_CMD	I/O	MFP5	SD/SDIO1 命令/响应脚
			SC2_nCD	I	MFP6	智能卡 2 卡检测管脚.
			UART0_nCTS	I	MFP7	UART0 禁止发送输入管脚
			UART5_TXD	O	MFP8	UART5数据发送输出脚
			I^2C0_SCL	I/O	MFP9	I2C0 时钟管脚
			CAN0_TXD	O	MFP10	CAN0总线发送输出
			BPWM0_CH5	I/O	MFP12	BPWM0 通道5 输出/捕获输入
			EPWM0_CH0	I/O	MFP13	EPWM0 通道0输出/捕获输入
			QEI0_INDEX	I	MFP14	正交编码器0 索引输入
26	56	64	PA.4	I/O	MFP0	通用数字I/O脚
			SPIM_D3	I/O	MFP2	SPIM QuadI/O模式下数据3管脚
			QSPI0_MOSI1	I/O	MFP3	QSPI0 MOSI1 (主机输出，从机输入) 管脚.

64 管脚	128 管脚	144 管脚	管脚名称	类型	MFP	描述
			SPI0_I2SMCLK	I/O	MFP4	SPI0 I²S 主机时钟输出脚
			SD1_CLK	O	MFP5	SD/SDIO1 时钟输出管脚
			SC0_nCD	I	MFP6	智能卡 0 卡检测管脚.
			UART0_nRTS	O	MFP7	UART0 请求发送输出脚
			UART5_RXD	I	MFP8	UART5 数据接收输入管脚
			I²C0_SDA	I/O	MFP9	I2C0 数据输入/输出管脚
			CAN0_RXD	I	MFP10	CAN0 总线接收输入
			BPWM0_CH4	I/O	MFP12	BPWM0 通道4输出/捕获输入脚
			EPWM0_CH1	I/O	MFP13	EPWM0 通道1输出/捕获输入脚
			QEI0_A	I	MFP14	正交编码器0 相位A输入
27	57	65	PA.3	I/O	MFP0	通用数字I/O脚
			SPIM_SS	I/O	MFP2	SPIM 从机选择脚
			QSPI0_SS	I/O	MFP3	QSPI0 从机选择脚
			SPI0_SS	I/O	MFP4	SPI0 从机选择脚
			SD1_DAT3	I/O	MFP5	SD/SDIO1 数据线位3.
			SC0_PWR	O	MFP6	智能卡0电源脚
			UART4_TXD	O	MFP7	UART4 数据发送输出脚.
			UART1_TXD	O	MFP8	UART1 数据发送输出脚
			I²C1_SCL	I/O	MFP9	I²C1 时钟脚
			BPWM0_CH3	I/O	MFP12	BPWM0 通道3输出/捕获输入脚
			EPWM0_CH2	I/O	MFP13	EPWM0 通道2输出/捕获输入脚
			QEI0_B	I	MFP14	正交编码器0 相位B 输入
28	58	66	PA.2	I/O	MFP0	通用数字I/O脚
			SPIM_CLK	I/O	MFP2	SPIM 串行时钟脚
			QSPI0_CLK	I/O	MFP3	QSPI0 串行时钟脚
			SPI0_CLK	I/O	MFP4	SPI0 串行时钟脚
			SD1_DAT2	I/O	MFP5	SD/SDIO1 数据线位2.
			SC0_RST	O	MFP6	智能卡0复位脚
			UART4_RXD	I	MFP7	UART4 数据接收输入脚
			UART1_RXD	I	MFP8	UART1数据接收输入管脚

64 管脚	128 管脚	144 管脚	管脚名称	类型	MFP	描述
			I^2C1_SDA	I/O	MFP9	I^2C1 数据输入/输出脚
			BPWM0_CH2	I/O	MFP12	BPWM0 通道2 输出/捕获输入
			EPWM0_CH3	I/O	MFP13	EPWM0 通道3输出/捕获输入脚
29	59	67	PA.1	I/O	MFP0	通用数字I/O脚
			SPIM_MISO	I/O	MFP2	SPIM MISO（主机输入，从机输出）脚.
			QSPI0_MISO0	I/O	MFP3	QSPI0 MISO0（主机输入，从机输出）脚.
			SPI0_MISO	I/O	MFP4	SPI0 MISO（主机输入，从机输出）脚.
			SD1_DAT1	I/O	MFP5	SD/SDIO1 数据线位1.
			SC0_DAT	I/O	MFP6	智能卡0数据输入脚
			UART0_TXD	O	MFP7	UART0 数据发送输出脚.
			UART1_nCTS	I	MFP8	UART1 禁止发送输入管脚
			I^2C2_SCL	I/O	MFP9	I^2C2 时钟脚
			BPWM0_CH1	I/O	MFP12	BPWM0 通道1 输出/捕获输入
			EPWM0_CH4	I/O	MFP13	EPWM0 通道4输出/捕获输入脚
			DAC1_ST	I	MFP15	DAC1 外部触发输入
30	60	68	PA.0	I/O	MFP0	通用数字I/O脚
			SPIM_MOSI	I/O	MFP2	SPIM MOSI (主机输出，从机输入) 管脚.
			QSPI0_MOSI0	I/O	MFP3	QSPI0 MOSI0 (主机输出，从机输入) 管脚.
			SPI0_MOSI	I/O	MFP4	SPI0 MOSI (主机输出，从机输入) 管脚.
			SD1_DAT0	I/O	MFP5	SD/SDIO1 数据线位0.
			SC0_CLK	O	MFP6	智能卡0时钟管脚
			UART0_RXD	I	MFP7	UART0 数据接收输入脚
			UART1_nRTS	O	MFP8	UART1 请求发送输出脚
			I^2C2_SDA	I/O	MFP9	I^2C2 数据输入/输出脚
			BPWM0_CH0	I/O	MFP12	BPWM0 通道0 输出/捕获输入
			EPWM0_CH5	I/O	MFP13	EPWM0 通道5 输出/捕获输入
			DAC0_ST	I	MFP15	DAC0 外部触发输入
31	61	69	V_{DDIO}	P	MFP0	PE.1, PE.8~PE.13 的电源
	62	70	PE.14	I/O	MFP0	通用数字I/O脚
			EBI_AD8	I/O	MFP2	EBI地址/数据总线位8.

64 管脚	128 管脚	144 管脚	管脚名称	类型	MFP	描述
			UART2_TXD	O	MFP3	UART2 数据发送输出脚.
			CAN0_TXD	O	MFP4	CAN0总线发送输出
			SD1_nCD	I	MFP5	SD/SDIO1 卡检测输入管脚
	63	71	PE.15	I/O	MFP0	通用数字I/O脚
			EBI_AD9	I/O	MFP2	EBI地址/数据总线位9.
			UART2_RXD	I	MFP3	UART2 数据接收输入脚
			CAN0_RXD	I	MFP4	CAN0 总线接收输入
32	64	72	nRESET	I	MFP0	外部复位输入:低电平有效,有内部上拉，设置此管脚为低将复位初始状态
33	65	73	PF.0	I/O	MFP0	通用数字I/O脚
			UART1_TXD	O	MFP2	UART1 数据发送输出脚
			I^2C1_SCL	I/O	MFP3	I^2C1 时钟脚
			BPWM1_CH0	I/O	MFP12	BPWM1 通道0 输出/捕获输入
			ICE_DAT	O	MFP14	串行调试总线数据管脚.
34	66	74	PF.1	I/O	MFP0	通用数字I/O脚
			UART1_RXD	I	MFP2	UART1数据接收输入管脚
			I^2C1_SDA	I/O	MFP3	I^2C1 数据输入/输出脚
			BPWM1_CH1	I/O	MFP12	BPWM1 通道1 输出/捕获输入
			ICE_CLK	I	MFP14	串行调试总线时钟脚
	67	75	PD.9	I/O	MFP0	通用数字I/O脚
			EBI_AD7	I/O	MFP2	EBI地址/数据总线位7.
			I^2C2_SCL	I/O	MFP3	I^2C2 时钟脚
			UART2_nCTS	I	MFP4	UART2 禁止发送输入管脚
	68	76	PD.8	I/O	MFP0	通用数字I/O脚
			EBI_AD6	I/O	MFP2	EBI地址/数据总线位6.
			I^2C2_SDA	I/O	MFP3	I^2C2 数据输入/输出脚
			UART2_nRTS	O	MFP4	UART2 请求发送输出脚
35	69	77	PC.5	I/O	MFP0	通用数字I/O脚
			EBI_AD5	I/O	MFP2	EBI地址/数据总线位5.
			SPIM_D2	I/O	MFP3	SPIM QuadI/O模式下数据2管脚
			QSPI0_MISO1	I/O	MFP4	QSPI0 MISO1（主机输入，从机输出）脚.

64 管脚	128 管脚	144 管脚	管脚名称	类型	MFP	描述
			UART2_TXD	O	MFP8	UART2 数据发送输出脚.
			I²C1_SCL	I/O	MFP9	I²C1 时钟脚
			CAN0_TXD	O	MFP10	CAN0总线发送输出
			UART4_TXD	O	MFP11	UART4 数据发送输出脚.
			EPWM1_CH0	I/O	MFP12	EPWM1 通道0 输出/捕获输入
36	70	78	PC.4	I/O	MFP0	通用数字I/O脚
			EBI_AD4	I/O	MFP2	EBI地址/数据总线位4.
			SPIM_D3	I/O	MFP3	SPIM QuadI/O模式下数据3管脚
			QSPI0_MOSI1	I/O	MFP4	QSPI0 MOSI1 (主机输出，从机输入) 管脚.
			SC1_nCD	I	MFP5	智能卡 1 卡检测管脚.
			I²S0_BCLK	O	MFP6	I2S0位时钟输出管脚
			SPI1_I2SMCLK	I/O	MFP7	SPI1 I²S 主机时钟输出脚
			UART2_RXD	I	MFP8	UART2 数据接收输入脚
			I²C1_SDA	I/O	MFP9	I²C1 数据输入/输出脚
			CAN0_RXD	I	MFP10	CAN0 总线接收输入
			UART4_RXD	I	MFP11	UART4 数据接收输入脚
			EPWM1_CH1	I/O	MFP12	EPWM1 通道1 输出/捕获输入
37	71	79	PC.3	I/O	MFP0	通用数字I/O脚
			EBI_AD3	I/O	MFP2	EBI地址/数据总线位3.
			SPIM_SS	I/O	MFP3	SPIM 从机选择脚
			QSPI0_SS	I/O	MFP4	QSPI0 从机选择脚
			SC1_PWR	O	MFP5	智能卡 1电源脚
			I²S0_MCLK	O	MFP6	I2S0 主时钟输出脚
			SPI1_MISO	I/O	MFP7	SPI1 MISO(主机输入，从机输出)管脚
			UART2_nRTS	O	MFP8	UART2 请求发送输出脚
			I²C0_SMBAL	O	MFP9	I²C0 SMBus SMBALTER 管脚
			CAN1_TXD	O	MFP10	CAN1总线发送输出
			UART3_TXD	O	MFP11	UART3 数据发送输出脚.
			EPWM1_CH2	I/O	MFP12	EPWM1 通道2 输出/捕获输入
38	72	80	PC.2	I/O	MFP0	通用数字I/O脚

64 管脚	128 管脚	144 管脚	管脚名称	类型	MFP	描述
			EBI_AD2	I/O	MFP2	EBI地址/数据总线位2.
			SPIM_CLK	I/O	MFP3	SPIM 串行时钟脚
			QSPI0_CLK	I/O	MFP4	QSPI0 串行时钟脚
			SC1_RST	O	MFP5	智能卡 1 复位脚
			I^2S0_DI	I	MFP6	I2S0 数据输入脚
			SPI1_MOSI	I/O	MFP7	SPI1 MOSI (主机输出，从机输入) 管脚
			UART2_nCTS	I	MFP8	UART2 禁止发送输入管脚
			I^2C0_SMBSUS	O	MFP9	I^2C0 SMBus SMBSUS 管脚 (PMBus 控制管脚)
			CAN1_RXD	I	MFP10	CAN1 总线接收输入
			UART3_RXD	I	MFP11	UART3 数据接收输入脚
			EPWM1_CH3	I/O	MFP12	EPWM1 通道3输出/捕获输入脚
39	73	81	PC.1	I/O	MFP0	通用数字I/O脚
			EBI_AD1	I/O	MFP2	EBI地址/数据总线位1.
			SPIM_MISO	I/O	MFP3	SPIM MISO（主机输入，从机输出）脚.
			QSPI0_MISO0	I/O	MFP4	QSPI0 MISO0（主机输入，从机输出）脚.
			SC1_DAT	I/O	MFP5	智能卡 1 数据管脚.
			I^2S0_DO	O	MFP6	I^2S0 数据输出脚
			SPI1_CLK	I/O	MFP7	SPI1 串行时钟管脚
			UART2_TXD	O	MFP8	UART2 数据发送输出脚.
			I^2C0_SCL	I/O	MFP9	I2C0 时钟管脚
			EPWM1_CH4	I/O	MFP12	EPWM1 通道4输出/捕获输入脚
			ACMP0_O	O	MFP14	模拟比较器 0输出脚
40	74	82	PC.0	I/O	MFP0	通用数字I/O脚
			EBI_AD0	I/O	MFP2	EBI地址/数据总线位0.
			SPIM_MOSI	I/O	MFP3	SPIM MOSI (主机输出，从机输入) 管脚.
			QSPI0_MOSI0	I/O	MFP4	QSPI0 MOSI0 (主机输出，从机输入) 管脚.
			SC1_CLK	O	MFP5	智能卡 1 时钟脚
			I^2S0_LRCK	O	MFP6	I^2S0左右通道时钟输出脚.
			SPI1_SS	I/O	MFP7	SPI1 从机选择脚
			UART2_RXD	I	MFP8	UART2 数据接收输入脚

64 管脚	128 管脚	144 管脚	管脚名称	类型	MFP	描述
			I^2C0_SDA	I/O	MFP9	I2C0 数据输入/输出管脚
			EPWM1_CH5	I/O	MFP12	EPWM1 通道5输出/捕获输入
			ACMP1_O	O	MFP14	模拟比较器1输出脚
	75	83	V_{SS}	P	MFP0	数字电路地管脚
	76	84	V_{DD}	P	MFP0	I/O和内部PLL及数字电路的LDO电源
	77	85	PG.9	I/O	MFP0	通用数字I/O脚
			EBI_AD0	I/O	MFP2	EBI地址/数据总线位0.
			SD1_DAT3	I/O	MFP3	SD/SDIO1 数据线位3.
			SPIM_D2	I/O	MFP4	SPIM QuadI/O模式下数据2管脚
			BPWM0_CH5	I/O	MFP12	BPWM0 通道5 输出/捕获输入
	78	86	PG.10	I/O	MFP0	通用数字I/O脚
			EBI_AD1	I/O	MFP2	EBI地址/数据总线位1.
			SD1_DAT2	I/O	MFP3	SD/SDIO1 数据线位2.
			SPIM_D3	I/O	MFP4	SPIM QuadI/O模式下数据3管脚
			BPWM0_CH4	I/O	MFP12	BPWM0 通道4输出/捕获输入脚
	79	87	PG.11	I/O	MFP0	通用数字I/O脚
			EBI_AD2	I/O	MFP2	EBI地址/数据总线位2.
			SD1_DAT1	I/O	MFP3	SD/SDIO1 数据线位1.
			SPIM_SS	I/O	MFP4	SPIM 从机选择脚
			BPWM0_CH3	I/O	MFP12	BPWM0 通道3输出/捕获输入脚
	80	88	PG.12	I/O	MFP0	通用数字I/O脚
			EBI_AD3	I/O	MFP2	EBI地址/数据总线位3.
			SD1_DAT0	I/O	MFP3	SD/SDIO1 数据线位0.
			SPIM_CLK	I/O	MFP4	SPIM 串行时钟脚
			BPWM0_CH2	I/O	MFP12	BPWM0 通道2 输出/捕获输入
	81	89	PG.13	I/O	MFP0	通用数字I/O脚
			EBI_AD4	I/O	MFP2	EBI地址/数据总线位4.
			SD1_CMD	I/O	MFP3	SD/SDIO1 命令/响应脚
			SPIM_MISO	I/O	MFP4	SPIM MISO（主机输入，从机输出）脚.
			BPWM0_CH1	I/O	MFP12	BPWM0 通道1 输出/捕获输入

64 管脚	128 管脚	144 管脚	管脚名称	类型	MFP	描述
	82	90	PG.14	I/O	MFP0	通用数字I/O脚
			EBI_AD5	I/O	MFP2	EBI地址/数据总线位5.
			SD1_CLK	O	MFP3	SD/SDIO1 时钟输出管脚
			SPIM_MOSI	I/O	MFP4	SPIM MOSI (主机输出，从机输入) 管脚.
			BPWM0_CH0	I/O	MFP12	BPWM0 通道0 输出/捕获输入
	83	91	PG.15	I/O	MFP0	通用数字I/O脚
			SD1_nCD	I	MFP3	SD/SDIO1 卡检测输入管脚
			CLKO	O	MFP14	时钟输出
			EADC0_ST	I	MFP15	EADC0 外部触发输入
		92	PD.3	I/O	MFP0	通用数字I/O脚
			EBI_AD10	I/O	MFP2	EBI地址/数据总线位10.
			USCI0_CTL1	I/O	MFP3	USCI0 控制1 管脚.
			SPI0_SS	I/O	MFP4	SPI0 从机选择脚
			UART3_nRTS	O	MFP5	UART3 请求发送输出脚
			USCI1_CTL0	I/O	MFP6	USCI1 控制0管脚
			SC2_PWR	O	MFP7	智能卡 2电源脚
			SC1_nCD	I	MFP8	智能卡 1 卡检测管脚.
			UART0_TXD	O	MFP9	UART0 数据发送输出脚.
		93	PD.2	I/O	MFP0	通用数字I/O脚
			EBI_AD11	I/O	MFP2	EBI地址/数据总线位11.
			USCI0_DAT1	I/O	MFP3	USCI0 数据1 管脚.
			SPI0_CLK	I/O	MFP4	SPI0 串行时钟脚
			UART3_nCTS	I	MFP5	UART3 禁止发送输入管脚
			SC2_RST	O	MFP7	智能卡 2 复位脚
			UART0_RXD	I	MFP9	UART0 数据接收输入脚
		94	PD.1	I/O	MFP0	通用数字I/O脚
			EBI_AD12	I/O	MFP2	EBI地址/数据总线位12.
			USCI0_DAT0	I/O	MFP3	USCI0 数据0脚
			SPI0_MISO	I/O	MFP4	SPI0 MISO（主机输入，从机输出）脚.
			UART3_TXD	O	MFP5	UART3 数据发送输出脚.

64 管脚	128 管脚	144 管脚	管脚名称	类型	MFP	描述
			I²C2_SCL	I/O	MFP6	I²C2 时钟脚
			SC2_DAT	I/O	MFP7	智能卡 2 数据管脚.
		95	PD.0	I/O	MFP0	通用数字I/O脚
			EBI_AD13	I/O	MFP2	EBI地址/数据总线位13.
			USCI0_CLK	I/O	MFP3	USCI0 时钟脚
			SPI0_MOSI	I/O	MFP4	SPI0 MOSI (主机输出，从机输入) 管脚.
			UART3_RXD	I	MFP5	UART3 数据接收输入脚
			I²C2_SDA	I/O	MFP6	I²C2 数据输入/输出脚
			SC2_CLK	O	MFP7	智能卡 2 时钟脚
			TM2	I/O	MFP14	Timer2 事件计数输入/反转输出脚
	84	96	PD.13	I/O	MFP0	通用数字I/O脚
			EBI_AD10	I/O	MFP2	EBI地址/数据总线位10.
			SD0_nCD	I	MFP3	SD/SDIO0 卡检测输入管脚
			SPI0_I2SMCLK	I/O	MFP4	SPI0 I²S 主机时钟输出脚
			SPI1_I2SMCLK	I/O	MFP5	SPI1 I²S 主机时钟输出脚
			SC2_nCD	I	MFP7	智能卡 2 卡检测管脚.
	85	97	PA.12	I/O	MFP0	通用数字I/O脚
			I²S0_BCLK	O	MFP2	I2S0位时钟输出管脚
			UART4_TXD	O	MFP3	UART4 数据发送输出脚.
			I²C1_SCL	I/O	MFP4	I²C1 时钟脚
			SPI2_SS	I/O	MFP5	SPI2 从机选择脚
			CAN0_TXD	O	MFP6	CAN0总线发送输出
			SC2_PWR	O	MFP7	智能卡 2电源脚
			BPWM1_CH2	I/O	MFP11	BPWM1 通道2 输出/捕获输入
			QEI1_INDEX	I	MFP12	正交编码器1 索引输入
			USB_VBUS	P	MFP14	USB主机或HUB的电源
	86	98	PA.13	I/O	MFP0	通用数字I/O脚
			I²S0_MCLK	O	MFP2	I2S0 主时钟输出脚
			UART4_RXD	I	MFP3	UART4 数据接收输入脚
			I²C1_SDA	I/O	MFP4	I²C1 数据输入/输出脚

64 管脚	128 管脚	144 管脚	管脚名称	类型	MFP	描述
			SPI2_CLK	I/O	MFP5	SPI2 串行时钟脚
			CAN0_RXD	I	MFP6	CAN0 总线接收输入
			SC2_RST	O	MFP7	智能卡 2 复位脚
			BPWM1_CH3	I/O	MFP11	BPWM1 通道3输出/捕获输入脚
			QEI1_A	I	MFP12	正交编码器1 相位A输入
			USB_D-	A	MFP14	USB差分信号D+.
	87	99	PA.14	I/O	MFP0	通用数字I/O脚
			I^2S0_DI	I	MFP2	I2S0 数据输入脚
			UART0_TXD	O	MFP3	UART0 数据发送输出脚.
			SPI2_MISO	I/O	MFP5	SPI2 MISO（主机输入，从机输出）脚.
			I^2C2_SCL	I/O	MFP6	I^2C2 时钟脚
			SC2_DAT	I/O	MFP7	智能卡 2 数据管脚.
			BPWM1_CH4	I/O	MFP11	BPWM1 通道4输出/捕获输入脚
			QEI1_B	I	MFP12	正交编码器1 相位B 输入
			USB_D+	A	MFP14	USB差分信号D-.
	88	100	PA.15	I/O	MFP0	通用数字I/O脚
			I^2S0_DO	O	MFP2	I^2S0 数据输出脚
			UART0_RXD	I	MFP3	UART0 数据接收输入脚
			SPI2_MOSI	I/O	MFP5	SPI2 MOSI (主机输出，从机输入) 管脚.
			I^2C2_SDA	I/O	MFP6	I^2C2 数据输入/输出脚
			SC2_CLK	O	MFP7	智能卡 2 时钟脚
			BPWM1_CH5	I/O	MFP11	EPWM1 通道5输出/捕获输入
			EPWM0_SYNC_IN	I	MFP12	EPWM0 计数器同步触发输入脚
			USB_OTG_ID	I	MFP14	USB_ 标识
41	89	101	HSUSB_VRES	A	MFP0	HSUSB模块参考电阻
42	90	102	$HSUSB_V_{DD33}$	P	MFP0	HSUSB VDD33 电源供电
43	91	103	HSUSB_VBUS	P	MFP0	HSUSB USB主机或HUB的电源
44	92	104	HSUSB_D-	A	MFP0	HSUSB差分信号D+.
45	93	105	HSUSB_VSS	P	MFP0	HSUSB 地管脚
46	94	106	HSUSB_D+	A	MFP0	HSUSB差分信号D-.

64 管脚	128 管脚	144 管脚	管脚名称	类型	MFP	描述
47	95	107	HSUSB_V_{DD12}_CAP	A	MFP0	HSUSB内部1.2v电压输出去耦管脚. 注: 此管脚需接1uF电容到地.
48	96	108	HSUSB_ID	I	MFP0	HSUSB 标识
	97	109	PE.7	I/O	MFP0	通用数字I/O脚
			SD0_CMD	I/O	MFP3	SD/SDIO0 命令/响应脚
			SPIM_D2	I/O	MFP4	SPIM QuadI/O模式下数据2管脚
			UART5_TXD	O	MFP8	UART5数据发送输出脚
			CAN1_TXD	O	MFP9	CAN1总线发送输出
			QEI1_INDEX	I	MFP11	正交编码器1 索引输入
			EPWM0_CH0	I/O	MFP12	EPWM0 通道0输出/捕获输入
			BPWM0_CH5	I/O	MFP13	BPWM0 通道5 输出/捕获输入
	98	110	PE.6	I/O	MFP0	通用数字I/O脚
			SD0_CLK	O	MFP3	SD/SDIO0 时钟输出脚
			SPIM_D3	I/O	MFP4	SPIM QuadI/O模式下数据3管脚
			SPI3_I2SMCLK	I/O	MFP5	SPI3 I^2S 主机时钟输出脚
			SC0_nCD	I	MFP6	智能卡 0 卡检测管脚.
			USCI0_CTL0	I/O	MFP7	USCI0 控制0 管脚.
			UART5_RXD	I	MFP8	UART5 数据接收输入管脚
			CAN1_RXD	I	MFP9	CAN1 总线接收输入
			QEI1_A	I	MFP11	正交编码器1 相位A输入
			EPWM0_CH1	I/O	MFP12	EPWM0 通道1输出/捕获输入脚
			BPWM0_CH4	I/O	MFP13	BPWM0 通道4输出/捕获输入脚
	99	111	PE.5	I/O	MFP0	通用数字I/O脚
			EBI_nRD	O	MFP2	EBI 读使能输出脚
			SD0_DAT3	I/O	MFP3	SD/SDIO0数据线位3
			SPIM_SS	I/O	MFP4	SPIM 从机选择脚
			SPI3_SS	I/O	MFP5	SPI3 从机选择脚
			SC0_PWR	O	MFP6	智能卡0电源脚
			USCI0_CTL1	I/O	MFP7	USCI0 控制1 管脚.
			QEI1_B	I	MFP11	正交编码器1 相位B 输入
			EPWM0_CH2	I/O	MFP12	EPWM0 通道2输出/捕获输入脚

64 管脚	128 管脚	144 管脚	管脚名称	类型	MFP	描述
			BPWM0_CH3	I/O	MFP13	BPWM0 通道3输出/捕获输入脚
	100	112	PE.4	I/O	MFP0	通用数字I/O脚
			EBI_nWR	O	MFP2	EBI 写使能输出脚
			SD0_DAT2	I/O	MFP3	SD/SDIO0 数据线位2
			SPIM_CLK	I/O	MFP4	SPIM 串行时钟脚
			SPI3_CLK	I/O	MFP5	SPI3 串行时钟脚
			SC0_RST	O	MFP6	智能卡0复位脚
			USCI0_DAT1	I/O	MFP7	USCI0 数据1 管脚.
			QEI0_INDEX	I	MFP11	正交编码器0 索引输入
			EPWM0_CH3	I/O	MFP12	EPWM0 通道3输出/捕获输入脚
			BPWM0_CH2	I/O	MFP13	BPWM0 通道2 输出/捕获输入
	101	113	PE.3	I/O	MFP0	通用数字I/O脚
			EBI_MCLK	O	MFP2	EBI 外部时钟输出脚.
			SD0_DAT1	I/O	MFP3	SD/SDIO0 数据线位1
			SPIM_MISO	I/O	MFP4	SPIM MISO（主机输入，从机输出）脚.
			SPI3_MISO	I/O	MFP5	SPI3 MISO（主机输入，从机输出）脚.
			SC0_DAT	I/O	MFP6	智能卡0数据输入脚
			USCI0_DAT0	I/O	MFP7	USCI0 数据0脚
			QEI0_A	I	MFP11	正交编码器0 相位A输入
			EPWM0_CH4	I/O	MFP12	EPWM0 通道4输出/捕获输入脚
			BPWM0_CH1	I/O	MFP13	BPWM0 通道1 输出/捕获输入
	102	114	PE.2	I/O	MFP0	通用数字I/O脚
			EBI_ALE	O	MFP2	EBI地址锁存使能输出脚
			SD0_DAT0	I/O	MFP3	SD/SDIO0 数据线位0
			SPIM_MOSI	I/O	MFP4	SPIM MOSI (主机输出，从机输入) 管脚.
			SPI3_MOSI	I/O	MFP5	SPI3 MOSI (主机输出，从机输入) 管脚.
			SC0_CLK	O	MFP6	智能卡0时钟管脚
			USCI0_CLK	I/O	MFP7	USCI0 时钟脚
			QEI0_B	I	MFP11	正交编码器0 相位B 输入
			EPWM0_CH5	I/O	MFP12	EPWM0 通道5 输出/捕获输入

64 管脚	128 管脚	144 管脚	管脚名称	类型	MFP	描述
			BPWM0_CH0	I/O	MFP13	BPWM0 通道0 输出/捕获输入
	103	115	V_{SS}	P	MFP0	数字电路地管脚
	104	116	V_{DD}	P	MFP0	I/O和内部PLL及数字电路的LDO电源
	105	117	PE.1	I/O	MFP0	通用数字I/O脚
			EBI_AD10	I/O	MFP2	EBI地址/数据总线位10.
			QSPI0_MISO0	I/O	MFP3	QSPI0 MISO0（主机输入，从机输出）脚.
			SC2_DAT	I/O	MFP4	智能卡 2 数据管脚.
			I^2S0_BCLK	O	MFP5	I2S0位时钟输出管脚
			SPI1_MISO	I/O	MFP6	SPI1 MISO(主机输入，从机输出)管脚
			UART3_TXD	O	MFP7	UART3 数据发送输出脚.
			I^2C1_SCL	I/O	MFP8	I^2C1 时钟脚
			UART4_nCTS	I	MFP9	UART4 禁止发送输入管脚
	106	118	PE.0	I/O	MFP0	通用数字I/O脚
			EBI_AD11	I/O	MFP2	EBI地址/数据总线位11.
			QSPI0_MOSI0	I/O	MFP3	QSPI0 MOSI0 (主机输出，从机输入) 管脚.
			SC2_CLK	O	MFP4	智能卡 2 时钟脚
			I^2S0_MCLK	O	MFP5	I2S0 主时钟输出脚
			SPI1_MOSI	I/O	MFP6	SPI1 MOSI (主机输出，从机输入) 管脚
			UART3_RXD	I	MFP7	UART3 数据接收输入脚
			I^2C1_SDA	I/O	MFP8	I^2C1 数据输入/输出脚
			UART4_nRTS	O	MFP9	UART4 请求发送输出脚
	107	119	PH.8	I/O	MFP0	通用数字I/O脚
			EBI_AD12	I/O	MFP2	EBI地址/数据总线位12.
			QSPI0_CLK	I/O	MFP3	QSPI0 串行时钟脚
			SC2_PWR	O	MFP4	智能卡 2电源脚
			I^2S0_DI	I	MFP5	I2S0 数据输入脚
			SPI1_CLK	I/O	MFP6	SPI1 串行时钟管脚
			UART3_nRTS	O	MFP7	UART3 请求发送输出脚
			I^2C1_SMBAL	O	MFP8	I^2C1 SMBus SMBALTER 管脚
			I^2C2_SCL	I/O	MFP9	I^2C2 时钟脚

64 管脚	128 管脚	144 管脚	管脚名称	类型	MFP	描述
			UART1_TXD	O	MFP10	UART1 数据发送输出脚
	108	120	PH.9	I/O	MFP0	通用数字I/O脚
			EBI_AD13	I/O	MFP2	EBI地址/数据总线位13.
			QSPI0_SS	I/O	MFP3	QSPI0 从机选择脚
			SC2_RST	O	MFP4	智能卡 2 复位脚
			I^{2}S0_DO	O	MFP5	I^{2}S0 数据输出脚
			SPI1_SS	I/O	MFP6	SPI1 从机选择脚
			UART3_nCTS	I	MFP7	UART3 禁止发送输入管脚
			I^{2}C1_SMBSUS	O	MFP8	I^{2}C1 SMBus SMBSUS 管脚 (PMBus 控制管脚)
			I^{2}C2_SDA	I/O	MFP9	I^{2}C2 数据输入/输出脚
			UART1_RXD	I	MFP10	UART1数据接收输入管脚
	109	121	PH.10	I/O	MFP0	通用数字I/O脚
			EBI_AD14	I/O	MFP2	EBI地址/数据总线位14.
			QSPI0_MISO1	I/O	MFP3	QSPI0 MISO1（主机输入，从机输出）脚.
			SC2_nCD	I	MFP4	智能卡 2 卡检测管脚.
			I^{2}S0_LRCK	O	MFP5	I^{2}S0左右通道时钟输出脚.
			SPI1_I2SMCLK	I/O	MFP6	SPI1 I^{2}S 主机时钟输出脚
			UART4_TXD	O	MFP7	UART4 数据发送输出脚.
			UART0_TXD	O	MFP8	UART0 数据发送输出脚.
	110	122	PH.11	I/O	MFP0	通用数字I/O脚
			EBI_AD15	I/O	MFP2	EBI地址/数据总线位15.
			QSPI0_MOSI1	I/O	MFP3	QSPI0 MOSI1 (主机输出，从机输入) 管脚.
			UART4_RXD	I	MFP7	UART4 数据接收输入脚
			UART0_RXD	I	MFP8	UART0 数据接收输入脚
			EPWM0_CH5	I/O	MFP11	EPWM0 通道5 输出/捕获输入
	111	123	PD.14	I/O	MFP0	通用数字I/O脚
			EBI_nCS0	O	MFP2	EBI片选0输出脚
			SPI3_I2SMCLK	I/O	MFP3	SPI3 I^{2}S 主机时钟输出脚
			SC1_nCD	I	MFP4	智能卡 1 卡检测管脚.
			EPWM0_CH4	I/O	MFP11	EPWM0 通道4输出/捕获输入脚

64 管脚	128 管脚	144 管脚	管脚名称	类型	MFP	描述
		124	PG.5	I/O	MFP0	通用数字I/O脚
			EBI_nCS1	O	MFP2	EBI片选1输出管脚
			SPI3_SS	I/O	MFP3	SPI3 从机选择脚
			SC1_PWR	O	MFP4	智能卡 1电源脚
			EPWM0_CH3	I/O	MFP11	EPWM0 通道3输出/捕获输入脚
		125	PG.6	I/O	MFP0	通用数字I/O脚
			EBI_nCS2	O	MFP2	EBI片选2输出脚
			SPI3_CLK	I/O	MFP3	SPI3 串行时钟脚
			SC1_RST	O	MFP4	智能卡 1 复位脚
			EPWM0_CH2	I/O	MFP11	EPWM0 通道2输出/捕获输入脚
		126	PG.7	I/O	MFP0	通用数字I/O脚
			EBI_nWRL	O	MFP2	EBI低字节写使能输出脚
			SPI3_MISO	I/O	MFP3	SPI3 MISO（主机输入，从机输出）脚.
			SC1_DAT	I/O	MFP4	智能卡 1 数据管脚.
			EPWM0_CH1	I/O	MFP11	EPWM0 通道1输出/捕获输入脚
		127	PG.8	I/O	MFP0	通用数字I/O脚
			EBI_nWRH	O	MFP2	EBI 高字节写使能输出管脚
			SPI3_MOSI	I/O	MFP3	SPI3 MOSI (主机输出，从机输入) 管脚.
			SC1_CLK	O	MFP4	智能卡 1 时钟脚
			EPWM0_CH0	I/O	MFP11	EPWM0 通道0输出/捕获输入
49	112	128	V_{SS}	P	MFP0	数字电路地管脚
50	113	129	LDO_CAP	A	MFP0	LDO输出脚
51	114	130	V_{DD}	P	MFP0	I/O和内部PLL及数字电路的LDO电源
52	115	131	PC.14	I/O	MFP0	通用数字I/O脚
			EBI_AD11	I/O	MFP2	EBI地址/数据总线位11.
			SC1_nCD	I	MFP3	智能卡 1 卡检测管脚.
			SPI0_I2SMCLK	I/O	MFP4	SPI0 I^2S 主机时钟输出脚
			USCI0_CTL0	I/O	MFP5	USCI0 控制0 管脚.
			QSPI0_CLK	I/O	MFP6	QSPI0 串行时钟脚
			EPWM0_SYNC_IN	I	MFP11	EPWM0 计数器同步触发输入脚

64 管脚	128 管脚	144 管脚	管脚名称	类型	MFP	描述
			ETM_TRACE_CLK	I	MFP12	ETM接收追踪时钟输入管脚
			TM1	I/O	MFP13	Timer1 事件计数输入/反转输出脚
			USB_VBUS_ST	I	MFP14	USB 外部VBUS 电源状态管脚.
			HSUSB_VBUS_ST	I	MFP15	HSUSB 外部VBUS 电源状态管脚.
53	116	132	PB.15	I/O	MFP0	通用数字I/O脚
			EADC0_CH15	A	MFP1	EADC0 通道15 模拟输入
			EBI_AD12	I/O	MFP2	EBI地址/数据总线位12.
			SC1_PWR	O	MFP3	智能卡 1电源脚
			SPI0_SS	I/O	MFP4	SPI0 从机选择脚
			USCI0_CTL1	I/O	MFP5	USCI0 控制1 管脚.
			UART0_nCTS	I	MFP6	UART0 禁止发送输入管脚
			UART3_TXD	O	MFP7	UART3 数据发送输出脚.
			I^2C2_SMBAL	O	MFP8	I^2C2 SMBus SMBALTER 管脚
			EPWM1_CH0	I/O	MFP11	EPWM1 通道0 输出/捕获输入
			ETM_TRACE_DATA0	I	MFP12	ETM接收追踪数据0 输入管脚
			TM0_EXT	I/O	MFP13	Timer0 事件计数输入/反转输出管脚
			USB_VBUS_EN	O	MFP14	USB 外部 VBUS 电源使能管脚.
			HSUSB_VBUS_EN	O	MFP15	HSUSB 外部 VBUS 电源使能管脚.
54	117	133	PB.14	I/O	MFP0	通用数字I/O脚
			EADC0_CH14	A	MFP1	EADC0 通道14 模拟输入
			EBI_AD13	I/O	MFP2	EBI地址/数据总线位13.
			SC1_RST	O	MFP3	智能卡 1 复位脚
			SPI0_CLK	I/O	MFP4	SPI0 串行时钟脚
			USCI0_DAT1	I/O	MFP5	USCI0 数据1 管脚.
			UART0_nRTS	O	MFP6	UART0 请求发送输出脚
			UART3_RXD	I	MFP7	UART3 数据接收输入脚
			I^2C2_SMBSUS	O	MFP8	I^2C2 SMBus SMBSUS 管脚 (PMBus 控制管脚)
			EPWM1_CH1	I/O	MFP11	EPWM1 通道1 输出/捕获输入
			ETM_TRACE_DATA1	I	MFP12	ETM接收追踪数据1 输入管脚
			TM1_EXT	I/O	MFP13	Timer1 事件计数输入/反转输出脚

64 管脚	128 管脚	144 管脚	管脚名称	类型	MFP	描述
			CLKO	O	MFP14	时钟输出
55	118	134	PB.13	I/O	MFP0	通用数字I/O脚
			EADC0_CH13	A	MFP1	EADC0 通道13 模拟输入
			DAC1_OUT	A	MFP1	DAC1 模拟输出通道
			ACMP0_P3	A	MFP1	模拟比较器 0 正极输入 3 管脚.
			ACMP1_P3	A	MFP1	模拟比较器 1 正极输入 3 管脚.
			EBI_AD14	I/O	MFP2	EBI地址/数据总线位14.
			SC1_DAT	I/O	MFP3	智能卡 1 数据管脚.
			SPI0_MISO	I/O	MFP4	SPI0 MISO（主机输入，从机输出）脚.
			USCI0_DAT0	I/O	MFP5	USCI0 数据0脚
			UART0_TXD	O	MFP6	UART0 数据发送输出脚.
			UART3_nRTS	O	MFP7	UART3 请求发送输出脚
			I^2C2_SCL	I/O	MFP8	I^2C2 时钟脚
			EPWM1_CH2	I/O	MFP11	EPWM1 通道2 输出/捕获输入
			ETM_TRACE_DATA2	I	MFP12	ETM接收追踪数据2 输入管脚
			TM2_EXT	I/O	MFP13	Timer2 事件计数输入/反转输出脚
56	119	135	PB.12	I/O	MFP0	通用数字I/O脚
			EADC0_CH12	A	MFP1	EADC0 通道12 模拟输入
			DAC0_OUT	A	MFP1	DAC0 模拟输出通道
			ACMP0_P2	A	MFP1	模拟比较器 0 正极输入 2 管脚.
			ACMP1_P2	A	MFP1	模拟比较器 1 正极输入 2 管脚.
			EBI_AD15	I/O	MFP2	EBI地址/数据总线位15.
			SC1_CLK	O	MFP3	智能卡 1 时钟脚
			SPI0_MOSI	I/O	MFP4	SPI0 MOSI (主机输出，从机输入) 管脚.
			USCI0_CLK	I/O	MFP5	USCI0 时钟脚
			UART0_RXD	I	MFP6	UART0 数据接收输入脚
			UART3_nCTS	I	MFP7	UART3 禁止发送输入管脚
			I^2C2_SDA	I/O	MFP8	I^2C2 数据输入/输出脚
			SD0_nCD	I	MFP9	SD/SDIO0 卡检测输入管脚
			EPWM1_CH3	I/O	MFP11	EPWM1 通道3输出/捕获输入脚

64 管脚	128 管脚	144 管脚	管脚名称	类型	MFP	描述
			ETM_TRACE_DATA3	I	MFP12	ETM接收追踪数据3 输入管脚
			TM3_EXT	I/O	MFP13	Timer3 事件计数输入/反转输出脚
57	120	136	AV_{DD}	P	MFP0	内部模拟电路电源
58	121	137	V_{REF}	A	MFP0	ADC参考电压输入. 注: 此管脚需接1uF电容到地.
59	122	138	AV_{SS}	P	MFP0	模拟电路地管脚
60	123	139	PB.11	I/O	MFP0	通用数字I/O脚
			EADC0_CH11	A	MFP1	EADC0 通道11 模拟输入
			EBI_ADR16	O	MFP2	EBI 地址总线位 16.
			EMAC_RMII_MDC	O	MFP3	EMAC RMII PHY管理时钟输出脚.
			UART0_nCTS	I	MFP5	UART0 禁止发送输入管脚
			UART4_TXD	O	MFP6	UART4 数据发送输出脚.
			I^2C1_SCL	I/O	MFP7	I^2C1 时钟脚
			CAN0_TXD	O	MFP8	CAN0总线发送输出
			SPI0_I2SMCLK	I/O	MFP9	SPI0 I^2S 主机时钟输出脚
			BPWM1_CH0	I/O	MFP10	BPWM1 通道0 输出/捕获输入
			SPI3_CLK	I/O	MFP11	SPI3 串行时钟脚
			HSUSB_VBUS_ST	I	MFP14	HSUSB 外部VBUS 电源状态管脚.
61	124	140	PB.10	I/O	MFP0	通用数字I/O脚
			EADC0_CH10	A	MFP1	EADC0 通道10 模拟输入
			EBI_ADR17	O	MFP2	EBI 地址总线位 17.
			EMAC_RMII_MDIO	I/O	MFP3	EMAC RMII PHY管理数据管脚.
			USCI1_CTL0	I/O	MFP4	USCI1 控制0管脚
			UART0_nRTS	O	MFP5	UART0 请求发送输出脚
			UART4_RXD	I	MFP6	UART4 数据接收输入脚
			I^2C1_SDA	I/O	MFP7	I^2C1 数据输入/输出脚
			CAN0_RXD	I	MFP8	CAN0 总线接收输入
			BPWM1_CH1	I/O	MFP10	BPWM1 通道1 输出/捕获输入
			SPI3_SS	I/O	MFP11	SPI3 从机选择脚
			HSUSB_VBUS_EN	O	MFP14	HSUSB 外部 VBUS 电源使能管脚.
62	125	141	PB.9	I/O	MFP0	通用数字I/O脚

64 管脚	128 管脚	144 管脚	管脚名称	类型	MFP	描述
			EADC0_CH9	A	MFP1	EADC0 通道9 模拟输入
			EBI_ADR18	O	MFP2	EBI 地址总线位 18.
			EMAC_RMII_TXD0	O	MFP3	EMAC RMII 发送数据总线位0.
			USCI1_CTL1	I/O	MFP4	USCI1 控制1管脚
			UART0_TXD	O	MFP5	UART0 数据发送输出脚.
			UART1_nCTS	I	MFP6	UART1 禁止发送输入管脚
			I^2C1_SMBAL	O	MFP7	I^2C1 SMBus SMBALTER 管脚
			BPWM1_CH2	I/O	MFP10	BPWM1 通道2 输出/捕获输入
			SPI3_MISO	I/O	MFP11	SPI3 MISO（主机输入，从机输出）脚.
			INT7	I	MFP13	外部中断7 输入脚
63	126	142	PB.8	I/O	MFP0	通用数字I/O脚
			EADC0_CH8	A	MFP1	EADC0 通道8 模拟输入
			EBI_ADR19	O	MFP2	EBI 地址总线位 19.
			EMAC_RMII_TXD1	O	MFP3	EMAC RMII 发送数据总线位1.
			USCI1_CLK	I/O	MFP4	USCI1 时钟脚
			UART0_RXD	I	MFP5	UART0 数据接收输入脚
			UART1_nRTS	O	MFP6	UART1 请求发送输出脚
			I^2C1_SMBSUS	O	MFP7	I^2C1 SMBus SMBSUS 管脚 (PMBus 控制管脚)
			BPWM1_CH3	I/O	MFP10	BPWM1 通道3输出/捕获输入脚
			SPI3_MOSI	I/O	MFP11	SPI3 MOSI (主机输出，从机输入) 管脚.
			INT6	I	MFP13	外部中断6 输入脚
64	127	143	PB.7	I/O	MFP0	通用数字I/O脚
			EADC0_CH7	A	MFP1	EADC0 通道7 模拟输入
			EBI_nWRL	O	MFP2	EBI低字节写使能输出脚
			EMAC_RMII_TXEN	O	MFP3	EMAC RMII 发送使能输出脚
			USCI1_DAT0	I/O	MFP4	USCI1 数据0脚
			CAN1_TXD	O	MFP5	CAN1总线发送输出
			UART1_TXD	O	MFP6	UART1 数据发送输出脚
			SD1_CMD	I/O	MFP7	SD/SDIO1 命令/响应脚
			EBI_nCS0	O	MFP8	EBI片选0输出脚

64 管脚	128 管脚	144 管脚	管脚名称	类型	MFP	描述
			BPWM1_CH4	I/O	MFP10	BPWM1 通道4输出/捕获输入脚
			EPWM1_BRAKE0	I	MFP11	EPWM1 刹车0 输入脚
			EPWM1_CH4	I/O	MFP12	EPWM1 通道4输出/捕获输入脚
			INT5	I	MFP13	外部中断5 输入脚
			USB_VBUS_ST	I	MFP14	USB 外部VBUS 电源状态管脚.
			ACMP0_O	O	MFP15	模拟比较器 0输出脚
1	128	144	PB.6	I/O	MFP0	通用数字I/O脚
			EADC0_CH6	A	MFP1	EADC0通道6模拟输入
			EBI_nWRH	O	MFP2	EBI 高字节写使能输出管脚
			EMAC_PPS	O	MFP3	EMAC 每秒脉冲输出脚
			USCI1_DAT1	I/O	MFP4	USCI1数据1管脚.
			CAN1_RXD	I	MFP5	CAN1 总线接收输入
			UART1_RXD	I	MFP6	UART1数据接收输入管脚
			SD1_CLK	O	MFP7	SD/SDIO1 时钟输出管脚
			EBI_nCS1	O	MFP8	EBI片选1输出管脚
			BPWM1_CH5	I/O	MFP10	EPWM1 通道5输出/捕获输入
			EPWM1_BRAKE1	I	MFP11	EPWM1 刹车1输入管脚
			EPWM1_CH5	I/O	MFP12	EPWM1 通道5输出/捕获输入
			INT4	I	MFP13	外部中断4输入管脚
			USB_VBUS_EN	O	MFP14	USB 外部 VBUS 电源使能管脚.
			ACMP1_O	O	MFP15	模拟比较器1输出脚

4.2.7 多功能管脚表

组	管脚名称	GPIO	MFP	类型	描述
ACMP0	ACMP0_N	PB.3	MFP1	A	模拟比较器0负极输入脚
	ACMP0_O	PC.12	MFP14	O	模拟比较器 0输出脚
		PC.1	MFP14	O	
		PB.7	MFP15	O	
	ACMP0_P0	PA.11	MFP1	A	模拟比较器 0 正极输入脚0
	ACMP0_P1	PB.2	MFP1	A	模拟比较器0正极输入1脚
	ACMP0_P2	PB.12	MFP1	A	模拟比较器 0 正极输入 2 管脚.
	ACMP0_P3	PB.13	MFP1	A	模拟比较器 0 正极输入 3 管脚.
	ACMP0_WLAT	PA.7	MFP13	I	模拟比较器 0窗口锁存输入管脚
ACMP1	ACMP1_N	PB.5	MFP1	A	模拟比较器1负极输入管脚
	ACMP1_O	PC.11	MFP14	O	模拟比较器1输出脚
		PC.0	MFP14	O	
		PB.6	MFP15	O	
	ACMP1_P0	PA.10	MFP1	A	模拟比较器 1 正极输入脚0
	ACMP1_P1	PB.4	MFP1	A	模拟比较器1正极输入1脚
	ACMP1_P2	PB.12	MFP1	A	模拟比较器 1 正极输入 2 管脚.
	ACMP1_P3	PB.13	MFP1	A	模拟比较器 1 正极输入 3 管脚.
	ACMP1_WLAT	PA.6	MFP13	I	模拟比较器 1窗口锁存输入管脚
BPWM0	BPWM0_CH0	PA.11	MFP9	I/O	BPWM0 通道0 输出/捕获输入
		PA.0	MFP12	I/O	
		PG.14	MFP12	I/O	
		PE.2	MFP13	I/O	
	BPWM0_CH1	PA.10	MFP9	I/O	BPWM0 通道1 输出/捕获输入
		PA.1	MFP12	I/O	
		PG.13	MFP12	I/O	
		PE.3	MFP13	I/O	
	BPWM0_CH2	PA.9	MFP9	I/O	BPWM0 通道2 输出/捕获输入
		PA.2	MFP12	I/O	
		PG.12	MFP12	I/O	
		PE.4	MFP13	I/O	
	BPWM0_CH3	PA.8	MFP9	I/O	BPWM0 通道3输出/捕获输入脚

组	管脚名称	GPIO	MFP	类型	描述
		PA.3	MFP12	I/O	
		PG.11	MFP12	I/O	
		PE.5	MFP13	I/O	
	BPWM0_CH4	PC.13	MFP9	I/O	BPWM0 通道4输出/捕获输入脚
		PF.5	MFP8	I/O	
		PA.4	MFP12	I/O	
		PG.10	MFP12	I/O	
		PE.6	MFP13	I/O	
	BPWM0_CH5	PD.12	MFP9	I/O	BPWM0 通道5 输出/捕获输入
		PF.4	MFP8	I/O	
		PA.5	MFP12	I/O	
		PG.9	MFP12	I/O	
		PE.7	MFP13	I/O	
BPWM1	BPWM1_CH0	PF.3	MFP11	I/O	BPWM1 通道0 输出/捕获输入
		PC.7	MFP12	I/O	
		PF.0	MFP12	I/O	
		PB.11	MFP10	I/O	
	BPWM1_CH1	PF.2	MFP11	I/O	BPWM1 通道1 输出/捕获输入
		PC.6	MFP12	I/O	
		PF.1	MFP12	I/O	
		PB.10	MFP10	I/O	
	BPWM1_CH2	PA.7	MFP12	I/O	BPWM1 通道2 输出/捕获输入
		PA.12	MFP11	I/O	
		PB.9	MFP10	I/O	
	BPWM1_CH3	PA.6	MFP12	I/O	BPWM1 通道3输出/捕获输入脚
		PA.13	MFP11	I/O	
		PB.8	MFP10	I/O	
	BPWM1_CH4	PC.8	MFP12	I/O	BPWM1 通道4输出/捕获输入脚
		PA.14	MFP11	I/O	
		PB.7	MFP10	I/O	
	BPWM1_CH5	PE.13	MFP12	I/O	EPWM1 通道5输出/捕获输入
		PA.15	MFP11	I/O	
		PB.6	MFP10	I/O	

组	管脚名称	GPIO	MFP	类型	描述
CAN0	CAN0_RXD	PD.10	MFP4	I	CAN0 总线接收输入
		PA.4	MFP10	I	
		PE.15	MFP4	I	
		PC.4	MFP10	I	
		PA.13	MFP6	I	
		PB.10	MFP8	I	
	CAN0_TXD	PD.11	MFP4	O	CAN0总线发送输出
		PA.5	MFP10	O	
		PE.14	MFP4	O	
		PC.5	MFP10	O	
		PA.12	MFP6	O	
		PB.11	MFP8	O	
CAN1	CAN1_RXD	PC.9	MFP9	I	CAN1 总线接收输入
		PD.12	MFP5	I	
		PG.1	MFP7	I	
		PC.2	MFP10	I	
		PE.6	MFP9	I	
		PB.6	MFP5	I	
	CAN1_TXD	PC.10	MFP9	O	CAN1总线发送输出
		PC.13	MFP5	O	
		PG.0	MFP7	O	
		PC.3	MFP10	O	
		PE.7	MFP9	O	
		PB.7	MFP5	O	
CLKO	CLKO	PC.13	MFP13	O	时钟输出
		PD.12	MFP13	O	
		PG.15	MFP14	O	
		PB.14	MFP14	O	
DAC0	DAC0_OUT	PB.12	MFP1	A	DAC0 模拟输出通道
	DAC0_ST	PA.10	MFP14	I	DAC0 外部触发输入
		PA.0	MFP15	I	
DAC1	DAC1_OUT	PB.13	MFP1	A	DAC1 模拟输出通道
	DAC1_ST	PA.11	MFP14	I	DAC1 外部触发输入

组	管脚名称	GPIO	MFP	类型	描述
		PA.1	MFP15	I	
EADC0	EADC0_CH0	PB.0	MFP1	A	EADC0 通道0 模拟输入
	EADC0_CH1	PB.1	MFP1	A	EADC0 通道1 模拟输入
	EADC0_CH2	PB.2	MFP1	A	EADC0 通道2 模拟输入
	EADC0_CH3	PB.3	MFP1	A	EADC0 通道3模拟输入
	EADC0_CH4	PB.4	MFP1	A	EADC0 通道4模拟输入
	EADC0_CH5	PB.5	MFP1	A	EADC0 通道5模拟输入
	EADC0_CH6	PB.6	MFP1	A	EADC0通道6模拟输入
	EADC0_CH7	PB.7	MFP1	A	EADC0 通道7 模拟输入
	EADC0_CH8	PB.8	MFP1	A	EADC0 通道8 模拟输入
	EADC0_CH9	PB.9	MFP1	A	EADC0 通道9 模拟输入
	EADC0_CH10	PB.10	MFP1	A	EADC0 通道10 模拟输入
	EADC0_CH11	PB.11	MFP1	A	EADC0 通道11 模拟输入
	EADC0_CH12	PB.12	MFP1	A	EADC0 通道12 模拟输入
	EADC0_CH13	PB.13	MFP1	A	EADC0 通道13 模拟输入
	EADC0_CH14	PB.14	MFP1	A	EADC0 通道14 模拟输入
	EADC0_CH15	PB.15	MFP1	A	EADC0 通道15 模拟输入
	EADC0_ST	PC.13	MFP14	I	EADC0 外部触发输入
		PD.12	MFP14	I	
		PF.5	MFP11	I	
		PG.15	MFP15	I	
EBI	EBI_AD0	PC.0	MFP2	I/O	EBI地址/数据总线位0.
		PG.9	MFP2	I/O	
	EBI_AD1	PC.1	MFP2	I/O	EBI地址/数据总线位1.
		PG.10	MFP2	I/O	
	EBI_AD2	PC.2	MFP2	I/O	EBI地址/数据总线位2.
		PG.11	MFP2	I/O	
	EBI_AD3	PC.3	MFP2	I/O	EBI地址/数据总线位3.
		PG.12	MFP2	I/O	
	EBI_AD4	PC.4	MFP2	I/O	EBI地址/数据总线位4.
		PG.13	MFP2	I/O	
	EBI_AD5	PC.5	MFP2	I/O	EBI地址/数据总线位5.

组	管脚名称	GPIO	MFP	类型	描述
		PG.14	MFP2	I/O	
	EBI_AD6	PA.6	MFP2	I/O	EBI地址/数据总线位6.
		PD.8	MFP2	I/O	
	EBI_AD7	PA.7	MFP2	I/O	EBI地址/数据总线位7.
		PD.9	MFP2	I/O	
	EBI_AD8	PC.6	MFP2	I/O	EBI地址/数据总线位8.
		PE.14	MFP2	I/O	
	EBI_AD9	PC.7	MFP2	I/O	EBI地址/数据总线位9.
		PE.15	MFP2	I/O	
	EBI_AD10	PD.3	MFP2	I/O	EBI地址/数据总线位10.
		PD.13	MFP2	I/O	
		PE.1	MFP2	I/O	
	EBI_AD11	PD.2	MFP2	I/O	EBI地址/数据总线位11.
		PE.0	MFP2	I/O	
		PC.14	MFP2	I/O	
	EBI_AD12	PD.1	MFP2	I/O	EBI地址/数据总线位12.
		PH.8	MFP2	I/O	
		PB.15	MFP2	I/O	
	EBI_AD13	PD.0	MFP2	I/O	EBI地址/数据总线位13.
		PH.9	MFP2	I/O	
		PB.14	MFP2	I/O	
	EBI_AD14	PH.10	MFP2	I/O	EBI地址/数据总线位14.
		PB.13	MFP2	I/O	
	EBI_AD15	PH.11	MFP2	I/O	EBI地址/数据总线位15.
		PB.12	MFP2	I/O	
	EBI_ADR0	PB.5	MFP2	O	EBI地址总线位0
		PH.7	MFP2	O	
	EBI_ADR1	PB.4	MFP2	O	EBI 地址总线位1
		PH.6	MFP2	O	
	EBI_ADR2	PB.3	MFP2	O	EBI 地址总线位2
		PH.5	MFP2	O	
	EBI_ADR3	PB.2	MFP2	O	EBI 地址总线位3
		PH.4	MFP2	O	

组	管脚名称	GPIO	MFP	类型	描述
	EBI_ADR4	PC.12	MFP2	O	EBI 地址总线位 4.
		PH.3	MFP2	O	
	EBI_ADR5	PC.11	MFP2	O	EBI 地址总线位 5.
		PH.2	MFP2	O	
	EBI_ADR6	PC.10	MFP2	O	EBI 地址总线位 6.
		PH.1	MFP2	O	
	EBI_ADR7	PC.9	MFP2	O	EBI 地址总线位 7.
		PH.0	MFP2	O	
	EBI_ADR8	PB.1	MFP2	O	EBI 地址总线位 8.
		PG.0	MFP2	O	
	EBI_ADR9	PB.0	MFP2	O	EBI 地址总线位 9.
		PG.1	MFP2	O	
	EBI_ADR10	PC.13	MFP2	O	EBI 地址总线位 10.
		PE.8	MFP2	O	
	EBI_ADR11	PG.2	MFP2	O	EBI 地址总线位 11.
		PE.9	MFP2	O	
	EBI_ADR12	PG.3	MFP2	O	EBI 地址总线位 12.
		PE.10	MFP2	O	
	EBI_ADR13	PG.4	MFP2	O	EBI 地址总线位 13.
		PE.11	MFP2	O	
	EBI_ADR14	PF.11	MFP2	O	EBI 地址总线位 14.
		PE.12	MFP2	O	
	EBI_ADR15	PF.10	MFP2	O	EBI 地址总线位 15.
		PE.13	MFP2	O	
	EBI_ADR16	PF.9	MFP2	O	EBI 地址总线位 16.
		PC.8	MFP2	O	
		PB.11	MFP2	O	
	EBI_ADR17	PF.8	MFP2	O	EBI 地址总线位 17.
		PB.10	MFP2	O	
	EBI_ADR18	PF.7	MFP2	O	EBI 地址总线位 18.
		PB.9	MFP2	O	
	EBI_ADR19	PF.6	MFP2	O	EBI 地址总线位 19.
		PB.8	MFP2	O	

组	管脚名称	GPIO	MFP	类型	描述
	EBI_ALE	PA.8	MFP2	O	EBI地址锁存使能输出脚
		PE.2	MFP2	O	
	EBI_MCLK	PA.9	MFP2	O	EBI 外部时钟输出脚.
		PE.3	MFP2	O	
	EBI_nCS0	PD.12	MFP2	O	EBI片选0输出脚
		PF.6	MFP7	O	
		PF.3	MFP2	O	
		PD.14	MFP2	O	
		PB.7	MFP8	O	
	EBI_nCS1	PD.11	MFP2	O	EBI片选1输出管脚
		PF.2	MFP2	O	
		PG.5	MFP2	O	
		PB.6	MFP8	O	
	EBI_nCS2	PD.10	MFP2	O	EBI片选2输出脚
		PG.6	MFP2	O	
	EBI_nRD	PA.11	MFP2	O	EBI 读使能输出脚
		PE.5	MFP2	O	
	EBI_nWR	PA.10	MFP2	O	EBI 写使能输出脚
		PE.4	MFP2	O	
	EBI_nWRH	PG.8	MFP2	O	EBI 高字节写使能输出管脚
		PB.6	MFP2	O	
	EBI_nWRL	PG.7	MFP2	O	EBI低字节写使能输出脚
		PB.7	MFP2	O	
ECAP0	ECAP0_IC0	PA.10	MFP11	I	增强型捕获计数器 0 输入脚0
		PE.8	MFP12	I	
	ECAP0_IC1	PA.9	MFP11	I	增强型捕获计数器 0 输入 1 管脚.
		PE.9	MFP12	I	
	ECAP0_IC2	PA.8	MFP11	I	增强型捕获计数器 0 输入 2 管脚.
		PE.10	MFP12	I	
ECAP1	ECAP1_IC0	PC.10	MFP11	I	增强型捕获计数器 1 输入脚0
		PE.13	MFP13	I	
	ECAP1_IC1	PC.11	MFP11	I	增强型捕获计数器 1 输入 1 管脚.
		PE.12	MFP13	I	

组	管脚名称	GPIO	MFP	类型	描述
	ECAP1_IC2	PC.12	MFP11	I	增强型捕获计数器 1 输入 2 管脚.
		PE.11	MFP13	I	
EMAC	EMAC_RMII_MDC	PE.8	MFP3	O	EMAC RMII PHY管理时钟输出脚.
		PB.11	MFP3	O	
	EMAC_RMII_MDIO	PE.9	MFP3	I/O	EMAC RMII PHY管理数据管脚.
		PB.10	MFP3	I/O	
	EMAC_RMII_RXD0	PB.4	MFP4	I	EMAC RMII接收数据总线位 0.
		PC.7	MFP3	I	
	EMAC_RMII_RXD1	PB.3	MFP4	I	EMAC RMII接收数据总线位 1.
		PC.6	MFP3	I	
	EMAC_RMII_CRSDV	PB.2	MFP4	I	EMAC RMII 载波侦测/接收数据输入脚
		PA.7	MFP3	I	
	EMAC_RMII_RXERR	PB.1	MFP4	I	EMAC RMII 接收数据错误输入脚
		PA.6	MFP3	I	
	EMAC_RMII_TXD0	PE.10	MFP3	O	EMAC RMII 发送数据总线位0.
		PB.9	MFP3	O	
	EMAC_RMII_TXD1	PE.11	MFP3	O	EMAC RMII 发送数据总线位1.
		PB.8	MFP3	O	
	EMAC_RMII_TXEN	PE.12	MFP3	O	EMAC RMII 发送使能输出脚
		PB.7	MFP3	O	
	EMAC_PPS	PE.13	MFP3	O	EMAC 每秒脉冲输出脚
		PB.6	MFP3	O	
	EMAC_RMII_REFCLK	PB.5	MFP4	I	EMAC RMII 参考时钟输入脚
		PC.8	MFP3	I	
EPWM0	EPWM0_BRAKE0	PB.1	MFP13	I	EPWM0 刹车0 输入脚
		PE.8	MFP11	I	
	EPWM0_BRAKE1	PB.0	MFP13	I	EPWM0 刹车1 输入脚
		PE.9	MFP11	I	
	EPWM0_CH0	PB.5	MFP11	I/O	EPWM0 通道0输出/捕获输入
		PE.8	MFP10	I/O	
		PA.5	MFP13	I/O	
		PE.7	MFP12	I/O	
		PG.8	MFP11	I/O	

组	管脚名称	GPIO	MFP	类型	描述
	EPWM0_CH1	PB.4	MFP11	I/O	EPWM0 通道1输出/捕获输入脚
		PE.9	MFP10	I/O	
		PA.4	MFP13	I/O	
		PE.6	MFP12	I/O	
		PG.7	MFP11	I/O	
	EPWM0_CH2	PB.3	MFP11	I/O	EPWM0 通道2输出/捕获输入脚
		PE.10	MFP10	I/O	
		PA.3	MFP13	I/O	
		PE.5	MFP12	I/O	
		PG.6	MFP11	I/O	
	EPWM0_CH3	PB.2	MFP11	I/O	EPWM0 通道3输出/捕获输入脚
		PE.11	MFP10	I/O	
		PA.2	MFP13	I/O	
		PE.4	MFP12	I/O	
		PG.5	MFP11	I/O	
	EPWM0_CH4	PB.1	MFP11	I/O	EPWM0 通道4输出/捕获输入脚
		PE.12	MFP10	I/O	
		PA.1	MFP13	I/O	
		PE.3	MFP12	I/O	
		PD.14	MFP11	I/O	
	EPWM0_CH5	PB.0	MFP11	I/O	EPWM0 通道5 输出/捕获输入
		PE.13	MFP10	I/O	
		PA.0	MFP13	I/O	
		PE.2	MFP12	I/O	
		PH.11	MFP11	I/O	
	EPWM0_SYNC_IN	PA.15	MFP12	I	EPWM0 计数器同步触发输入脚
		PC.14	MFP11	I	
	EPWM0_SYNC_OUT	PA.11	MFP10	O	EPWM0计数器同步触发输出脚
		PF.5	MFP9	O	
EPWM1	EPWM1_BRAKE0	PE.10	MFP11	I	EPWM1 刹车0 输入脚
		PB.7	MFP11	I	
	EPWM1_BRAKE1	PE.11	MFP11	I	EPWM1 刹车1输入管脚
		PB.6	MFP11	I	

组	管脚名称	GPIO	MFP	类型	描述
	EPWM1_CH0	PC.12	MFP12	I/O	EPWM1 通道0 输出/捕获输入
		PE.13	MFP11	I/O	
		PC.5	MFP12	I/O	
		PB.15	MFP11	I/O	
	EPWM1_CH1	PC.11	MFP12	I/O	EPWM1 通道1 输出/捕获输入
		PC.8	MFP11	I/O	
		PC.4	MFP12	I/O	
		PB.14	MFP11	I/O	
	EPWM1_CH2	PC.10	MFP12	I/O	EPWM1 通道2 输出/捕获输入
		PC.7	MFP11	I/O	
		PC.3	MFP12	I/O	
		PB.13	MFP11	I/O	
	EPWM1_CH3	PC.9	MFP12	I/O	EPWM1 通道3输出/捕获输入脚
		PC.6	MFP11	I/O	
		PC.2	MFP12	I/O	
		PB.12	MFP11	I/O	
	EPWM1_CH4	PB.1	MFP12	I/O	EPWM1 通道4输出/捕获输入脚
		PA.7	MFP11	I/O	
		PC.1	MFP12	I/O	
		PB.7	MFP12	I/O	
	EPWM1_CH5	PB.0	MFP12	I/O	EPWM1 通道5输出/捕获输入
		PA.6	MFP11	I/O	
		PC.0	MFP12	I/O	
		PB.6	MFP12	I/O	
ETM	ETM_TRACE_CLK	PC.14	MFP12	I	ETM接收追踪时钟输入管脚
	ETM_TRACE_DATA0	PB.15	MFP12	I	ETM接收追踪数据0 输入管脚
	ETM_TRACE_DATA1	PB.14	MFP12	I	ETM接收追踪数据1 输入管脚
	ETM_TRACE_DATA2	PB.13	MFP12	I	ETM接收追踪数据2 输入管脚
	ETM_TRACE_DATA3	PB.12	MFP12	I	ETM接收追踪数据3 输入管脚
HSUSB	HSUSB_VBUS_EN	PB.15	MFP15	O	HSUSB 外部 VBUS 电源使能管脚.
		PB.10	MFP14	O	
	HSUSB_VBUS_ST	PC.14	MFP15	I	HSUSB 外部VBUS 电源状态管脚.
		PB.11	MFP14	I	

组	管脚名称	GPIO	MFP	类型	描述
I²C0	I²C0_SCL	PB.5	MFP6	I/O	I2C0 时钟管脚
		PC.12	MFP4	I/O	
		PG.0	MFP4	I/O	
		PH.2	MFP6	I/O	
		PF.3	MFP4	I/O	
		PE.13	MFP4	I/O	
		PA.5	MFP9	I/O	
		PC.1	MFP9	I/O	
		PD.7	MFP4	I/O	
	I²C0_SDA	PB.4	MFP6	I/O	I2C0 数据输入/输出管脚
		PC.11	MFP4	I/O	
		PG.1	MFP4	I/O	
		PH.3	MFP6	I/O	
		PF.2	MFP4	I/O	
		PC.8	MFP4	I/O	
		PA.4	MFP9	I/O	
		PC.0	MFP9	I/O	
		PD.6	MFP4	I/O	
	I²C0_SMBAL	PG.2	MFP4	O	I²C0 SMBus SMBALTER 管脚
		PC.3	MFP9	O	
	I²C0_SMBSUS	PG.3	MFP4	O	I²C0 SMBus SMBSUS 管脚 (PMBus 控制管脚)
		PC.2	MFP9	O	
I²C1	I²C1_SCL	PB.1	MFP9	I/O	I²C1 时钟脚
		PG.2	MFP5	I/O	
		PA.7	MFP8	I/O	
		PA.3	MFP9	I/O	
		PF.0	MFP3	I/O	
		PC.5	MFP9	I/O	
		PD.5	MFP4	I/O	
		PA.12	MFP4	I/O	
		PE.1	MFP8	I/O	
		PB.11	MFP7	I/O	
	I²C1_SDA	PB.0	MFP9	I/O	I²C1 数据输入/输出脚

组	管脚名称	GPIO	MFP	类型	描述
		PG.3	MFP5	I/O	
		PA.6	MFP8	I/O	
		PA.2	MFP9	I/O	
		PF.1	MFP3	I/O	
		PC.4	MFP9	I/O	
		PD.4	MFP4	I/O	
		PA.13	MFP4	I/O	
		PE.0	MFP8	I/O	
		PB.10	MFP7	I/O	
	I²C1_SMBAL	PG.0	MFP5	O	I²C1 SMBus SMBALTER 管脚
		PC.7	MFP8	O	
		PH.8	MFP8	O	
		PB.9	MFP7	O	
	I²C1_SMBSUS	PG.1	MFP5	O	I²C1 SMBus SMBSUS 管脚 (PMBus 控制管脚)
		PC.6	MFP8	O	
		PH.9	MFP8	O	
		PB.8	MFP7	O	
I²C2	I²C2_SCL	PA.11	MFP7	I/O	I²C2 时钟脚
		PA.1	MFP9	I/O	
		PD.9	MFP3	I/O	
		PD.1	MFP6	I/O	
		PA.14	MFP6	I/O	
		PH.8	MFP9	I/O	
		PB.13	MFP8	I/O	
	I²C2_SDA	PA.10	MFP7	I/O	I²C2 数据输入/输出脚
		PA.0	MFP9	I/O	
		PD.8	MFP3	I/O	
		PD.0	MFP6	I/O	
		PA.15	MFP6	I/O	
		PH.9	MFP9	I/O	
		PB.12	MFP8	I/O	
	I²C2_SMBAL	PB.15	MFP8	O	I²C2 SMBus SMBALTER 管脚
	I²C2_SMBSUS	PB.14	MFP8	O	I²C2 SMBus SMBSUS 管脚 (PMBus 控制管脚)

组	管脚名称	GPIO	MFP	类型	描述
I²S0	I²S0_BCLK	PB.5	MFP10	O	I2S0位时钟输出管脚
		PF.10	MFP4	O	
		PE.8	MFP4	O	
		PC.4	MFP6	O	
		PA.12	MFP2	O	
		PE.1	MFP5	O	
	I²S0_DI	PB.3	MFP10	I	I2S0 数据输入脚
		PF.8	MFP4	I	
		PE.10	MFP4	I	
		PC.2	MFP6	I	
		PA.14	MFP2	I	
		PH.8	MFP5	I	
	I²S0_DO	PB.2	MFP10	O	I²S0 数据输出脚
		PF.7	MFP4	O	
		PE.11	MFP4	O	
		PC.1	MFP6	O	
		PA.15	MFP2	O	
		PH.9	MFP5	O	
	I²S0_LRCK	PB.1	MFP10	O	I²S0左右通道时钟输出脚.
		PF.6	MFP4	O	
		PE.12	MFP4	O	
		PC.0	MFP6	O	
		PH.10	MFP5	O	
	I²S0_MCLK	PB.4	MFP10	O	I2S0 主时钟输出脚
		PF.9	MFP4	O	
		PE.9	MFP4	O	
		PC.3	MFP6	O	
		PA.13	MFP2	O	
		PE.0	MFP5	O	
ICE	ICE_CLK	PF.1	MFP14	I	串行调试总线时钟脚
	ICE_DAT	PF.0	MFP14	O	串行调试总线数据管脚.
INT0	INT0	PB.5	MFP15	I	外部中断0输入管脚
		PA.6	MFP15	I	

组	管脚名称	GPIO	MFP	类型	描述
INT1	INT1	PB.4	MFP15	I	外部中断1输入脚
		PA.7	MFP15	I	
INT2	INT2	PB.3	MFP15	I	外部中断2输入脚
		PC.6	MFP15	I	
INT3	INT3	PB.2	MFP15	I	外部中断3输入脚
		PC.7	MFP15	I	
INT4	INT4	PA.8	MFP15	I	外部中断4输入管脚
		PB.6	MFP13	I	
INT5	INT5	PD.12	MFP15	I	外部中断5 输入脚
		PB.7	MFP13	I	
INT6	INT6	PD.11	MFP15	I	外部中断6 输入脚
		PB.8	MFP13	I	
INT7	INT7	PD.10	MFP15	I	外部中断7 输入脚
		PB.9	MFP13	I	
OPA0	OPA0_N	PB.1	MFP1	A	运放0负极输入脚
	OPA0_O	PB.2	MFP1	A	运放0输出脚
	OPA0_P	PB.0	MFP1	A	运放 0 正极输入脚
OPA1	OPA1_N	PA.9	MFP1	A	运放 1 负极输入脚
	OPA1_O	PA.10	MFP1	A	运放 1输出脚
	OPA1_P	PA.8	MFP1	A	运放 1 正极输入脚
OPA2	OPA2_N	PD.11	MFP1	A	运放 2 负极输入脚
	OPA2_O	PD.12	MFP1	A	运放 2输出脚
	OPA2_P	PD.10	MFP1	A	运放 2 正极输入脚
QEI0	QEI0_A	PD.11	MFP10	I	正交编码器0 相位A输入
		PA.4	MFP14	I	
		PE.3	MFP11	I	
	QEI0_B	PD.10	MFP10	I	正交编码器0 相位B 输入
		PA.3	MFP14	I	
		PE.2	MFP11	I	
	QEI0_INDEX	PD.12	MFP10	I	正交编码器0 索引输入
		PA.5	MFP14	I	
		PE.4	MFP11	I	
QEI1	QEI1_A	PA.9	MFP10	I	正交编码器1 相位A输入

组	管脚名称	GPIO	MFP	类型	描述
		PA.13	MFP12	I	
		PE.6	MFP11	I	
	QEI1_B	PA.8	MFP10	I	正交编码器1 相位B 输入
		PA.14	MFP12	I	
		PE.5	MFP11	I	
	QEI1_INDEX	PA.10	MFP10	I	正交编码器1 索引输入
		PA.12	MFP12	I	
		PE.7	MFP11	I	
SC0	SC0_CLK	PB.5	MFP9	O	智能卡0时钟管脚
		PF.6	MFP3	O	
		PA.0	MFP6	O	
		PE.2	MFP6	O	
	SC0_DAT	PB.4	MFP9	I/O	智能卡0数据输入脚
		PF.7	MFP3	I/O	
		PA.1	MFP6	I/O	
		PE.3	MFP6	I/O	
	SC0_PWR	PB.2	MFP9	O	智能卡0电源脚
		PF.9	MFP3	O	
		PA.3	MFP6	O	
		PE.5	MFP6	O	
	SC0_RST	PB.3	MFP9	O	智能卡0复位脚
		PF.8	MFP3	O	
		PA.2	MFP6	O	
		PE.4	MFP6	O	
	SC0_nCD	PC.12	MFP9	I	智能卡 0 卡检测管脚.
		PF.10	MFP3	I	
		PA.4	MFP6	I	
		PE.6	MFP6	I	
SC1	SC1_CLK	PC.0	MFP5	O	智能卡 1 时钟脚
		PD.4	MFP8	O	
		PG.8	MFP4	O	
		PB.12	MFP3	O	
	SC1_DAT	PC.1	MFP5	I/O	智能卡 1 数据管脚.

组	管脚名称	GPIO	MFP	类型	描述
		PD.5	MFP8	I/O	
		PG.7	MFP4	I/O	
		PB.13	MFP3	I/O	
	SC1_PWR	PC.3	MFP5	O	智能卡 1电源脚
		PD.7	MFP8	O	
		PG.5	MFP4	O	
		PB.15	MFP3	O	
	SC1_RST	PC.2	MFP5	O	智能卡 1 复位脚
		PD.6	MFP8	O	
		PG.6	MFP4	O	
		PB.14	MFP3	O	
	SC1_nCD	PC.4	MFP5	I	智能卡 1 卡检测管脚.
		PD.3	MFP8	I	
		PD.14	MFP4	I	
		PC.14	MFP3	I	
SC2	SC2_CLK	PA.8	MFP3	O	智能卡 2 时钟脚
		PA.6	MFP6	O	
		PD.0	MFP7	O	
		PA.15	MFP7	O	
		PE.0	MFP4	O	
	SC2_DAT	PA.9	MFP3	I/O	智能卡 2 数据管脚.
		PA.7	MFP6	I/O	
		PD.1	MFP7	I/O	
		PA.14	MFP7	I/O	
		PE.1	MFP4	I/O	
	SC2_PWR	PA.11	MFP3	O	智能卡 2电源脚
		PC.7	MFP6	O	
		PD.3	MFP7	O	
		PA.12	MFP7	O	
		PH.8	MFP4	O	
	SC2_RST	PA.10	MFP3	O	智能卡 2 复位脚
		PC.6	MFP6	O	
		PD.2	MFP7	O	

组	管脚名称	GPIO	MFP	类型	描述
		PA.13	MFP7	O	
		PH.9	MFP4	O	
	SC2_nCD	PC.13	MFP3	I	智能卡 2 卡检测管脚.
		PA.5	MFP6	I	
		PD.13	MFP7	I	
		PH.10	MFP4	I	
SD0	SD0_CLK	PB.1	MFP3	O	SD/SDIO0 时钟输出脚
		PE.6	MFP3	O	
	SD0_CMD	PB.0	MFP3	I/O	SD/SDIO0 命令/响应脚
		PE.7	MFP3	I/O	
	SD0_DAT0	PB.2	MFP3	I/O	SD/SDIO0 数据线位0
		PE.2	MFP3	I/O	
	SD0_DAT1	PB.3	MFP3	I/O	SD/SDIO0 数据线位1
		PE.3	MFP3	I/O	
	SD0_DAT2	PB.4	MFP3	I/O	SD/SDIO0 数据线位2
		PE.4	MFP3	I/O	
	SD0_DAT3	PB.5	MFP3	I/O	SD/SDIO0数据线位3
		PE.5	MFP3	I/O	
	SD0_nCD	PD.13	MFP3	I	SD/SDIO0 卡检测输入管脚
		PB.12	MFP9	I	
SD1	SD1_CLK	PA.4	MFP5	O	SD/SDIO1 时钟输出管脚
		PG.14	MFP3	O	
		PB.6	MFP7	O	
	SD1_CMD	PA.5	MFP5	I/O	SD/SDIO1 命令/响应脚
		PG.13	MFP3	I/O	
		PB.7	MFP7	I/O	
	SD1_DAT0	PA.8	MFP5	I/O	SD/SDIO1 数据线位0.
		PA.0	MFP5	I/O	
		PG.12	MFP3	I/O	
	SD1_DAT1	PA.9	MFP5	I/O	SD/SDIO1 数据线位1.
		PA.1	MFP5	I/O	
		PG.11	MFP3	I/O	
	SD1_DAT2	PA.10	MFP5	I/O	SD/SDIO1 数据线位2.

组	管脚名称	GPIO	MFP	类型	描述
		PA.2	MFP5	I/O	
		PG.10	MFP3	I/O	
	SD1_DAT3	PA.11	MFP5	I/O	SD/SDIO1 数据线位3.
		PA.3	MFP5	I/O	
		PG.9	MFP3	I/O	
	SD1_nCD	PA.6	MFP5	I	SD/SDIO1 卡检测输入管脚
		PE.14	MFP5	I	
		PG.15	MFP3	I	
QSPI0	QSPI0_CLK	PF.2	MFP5	I/O	QSPI0 串行时钟脚
		PA.2	MFP3	I/O	
		PC.2	MFP4	I/O	
		PH.8	MFP3	I/O	
		PC.14	MFP6	I/O	
	QSPI0_MISO0	PA.1	MFP3	I/O	QSPI0 MISO0（主机输入，从机输出）脚.
		PC.1	MFP4	I/O	
		PE.1	MFP3	I/O	
	QSPI0_MISO1	PA.5	MFP3	I/O	QSPI0 MISO1（主机输入，从机输出）脚.
		PC.5	MFP4	I/O	
		PH.10	MFP3	I/O	
	QSPI0_MOSI0	PA.0	MFP3	I/O	QSPI0 MOSI0 (主机输出，从机输入) 管脚.
		PC.0	MFP4	I/O	
		PE.0	MFP3	I/O	
	QSPI0_MOSI1	PA.4	MFP3	I/O	QSPI0 MOSI1 (主机输出，从机输入) 管脚.
		PC.4	MFP4	I/O	
		PH.11	MFP3	I/O	
	QSPI0_SS	PA.3	MFP3	I/O	QSPI0 从机选择脚
		PC.3	MFP4	I/O	
		PH.9	MFP3	I/O	
SPI0	SPI0_CLK	PF.8	MFP5	I/O	SPI0 串行时钟脚
		PA.2	MFP4	I/O	
		PD.2	MFP4	I/O	
		PB.14	MFP4	I/O	
	SPI0_I2SMCLK	PB.0	MFP8	I/O	SPI0 I^2S 主机时钟输出脚

组	管脚名称	GPIO	MFP	类型	描述
		PF.10	MFP5	I/O	
		PA.4	MFP4	I/O	
		PD.13	MFP4	I/O	
		PC.14	MFP4	I/O	
		PB.11	MFP9	I/O	
	SPI0_MISO	PF.7	MFP5	I/O	SPI0 MISO（主机输入，从机输出）脚.
		PA.1	MFP4	I/O	
		PD.1	MFP4	I/O	
		PB.13	MFP4	I/O	
	SPI0_MOSI	PF.6	MFP5	I/O	SPI0 MOSI (主机输出，从机输入) 管脚.
		PA.0	MFP4	I/O	
		PD.0	MFP4	I/O	
		PB.12	MFP4	I/O	
	SPI0_SS	PF.9	MFP5	I/O	SPI0 从机选择脚
		PA.3	MFP4	I/O	
		PD.3	MFP4	I/O	
		PB.15	MFP4	I/O	
SPI1	SPI1_CLK	PB.3	MFP5	I/O	SPI1 串行时钟管脚
		PH.6	MFP3	I/O	
		PA.7	MFP4	I/O	
		PC.1	MFP7	I/O	
		PD.5	MFP5	I/O	
		PH.8	MFP6	I/O	
	SPI1_I2SMCLK	PB.1	MFP5	I/O	SPI1 I^2S 主机时钟输出脚
		PH.3	MFP3	I/O	
		PA.5	MFP4	I/O	
		PC.4	MFP7	I/O	
		PD.13	MFP5	I/O	
		PH.10	MFP6	I/O	
	SPI1_MISO	PB.5	MFP5	I/O	SPI1 MISO(主机输入，从机输出)管脚
		PH.4	MFP3	I/O	
		PC.7	MFP4	I/O	
		PC.3	MFP7	I/O	

组	管脚名称	GPIO	MFP	类型	描述
		PD.7	MFP5	I/O	
		PE.1	MFP6	I/O	
	SPI1_MOSI	PB.4	MFP5	I/O	SPI1 MOSI (主机输出，从机输入) 管脚
		PH.5	MFP3	I/O	
		PC.6	MFP4	I/O	
		PC.2	MFP7	I/O	
		PD.6	MFP5	I/O	
		PE.0	MFP6	I/O	
	SPI1_SS	PB.2	MFP5	I/O	SPI1 从机选择脚
		PH.7	MFP3	I/O	
		PA.6	MFP4	I/O	
		PC.0	MFP7	I/O	
		PD.4	MFP5	I/O	
		PH.9	MFP6	I/O	
SPI2	SPI2_CLK	PA.10	MFP4	I/O	SPI2 串行时钟脚
		PG.3	MFP3	I/O	
		PE.8	MFP5	I/O	
		PA.13	MFP5	I/O	
	SPI2_I2SMCLK	PC.13	MFP4	I/O	SPI2 I^2S 主机时钟输出脚
		PG.1	MFP3	I/O	
		PE.12	MFP5	I/O	
	SPI2_MISO	PA.9	MFP4	I/O	SPI2 MISO（主机输入，从机输出）脚.
		PG.4	MFP3	I/O	
		PE.9	MFP5	I/O	
		PA.14	MFP5	I/O	
	SPI2_MOSI	PA.8	MFP4	I/O	SPI2 MOSI (主机输出，从机输入) 管脚.
		PF.11	MFP3	I/O	
		PE.10	MFP5	I/O	
		PA.15	MFP5	I/O	
	SPI2_SS	PA.11	MFP4	I/O	SPI2 从机选择脚
		PG.2	MFP3	I/O	
		PE.11	MFP5	I/O	
		PA.12	MFP5	I/O	

组	管脚名称	GPIO	MFP	类型	描述
SPI3	SPI3_CLK	PC.10	MFP6	I/O	SPI3 串行时钟脚
		PE.4	MFP5	I/O	
		PG.6	MFP3	I/O	
		PB.11	MFP11	I/O	
	SPI3_I2SMCLK	PB.1	MFP6	I/O	SPI3 I^2S 主机时钟输出脚
		PE.6	MFP5	I/O	
		PD.14	MFP3	I/O	
	SPI3_MISO	PC.12	MFP6	I/O	SPI3 MISO（主机输入，从机输出）脚.
		PE.3	MFP5	I/O	
		PG.7	MFP3	I/O	
		PB.9	MFP11	I/O	
	SPI3_MOSI	PC.11	MFP6	I/O	SPI3 MOSI (主机输出，从机输入) 管脚.
		PE.2	MFP5	I/O	
		PG.8	MFP3	I/O	
		PB.8	MFP11	I/O	
	SPI3_SS	PC.9	MFP6	I/O	SPI3 从机选择脚
		PE.5	MFP5	I/O	
		PG.5	MFP3	I/O	
		PB.10	MFP11	I/O	
SPIM	SPIM_CLK	PA.2	MFP2	I/O	SPIM 串行时钟脚
		PC.2	MFP3	I/O	
		PG.12	MFP4	I/O	
		PE.4	MFP4	I/O	
	SPIM_D2	PA.5	MFP2	I/O	SPIM QuadI/O模式下数据2管脚
		PC.5	MFP3	I/O	
		PG.9	MFP4	I/O	
		PE.7	MFP4	I/O	
	SPIM_D3	PA.4	MFP2	I/O	SPIM QuadI/O模式下数据3管脚
		PC.4	MFP3	I/O	
		PG.10	MFP4	I/O	
		PE.6	MFP4	I/O	
	SPIM_MISO	PA.1	MFP2	I/O	SPIM MISO（主机输入，从机输出）脚.
		PC.1	MFP3	I/O	

组	管脚名称	GPIO	MFP	类型	描述
		PG.13	MFP4	I/O	
		PE.3	MFP4	I/O	
	SPIM_MOSI	PA.0	MFP2	I/O	SPIM MOSI (主机输出，从机输入) 管脚.
		PC.0	MFP3	I/O	
		PG.14	MFP4	I/O	
		PE.2	MFP4	I/O	
	SPIM_SS	PA.3	MFP2	I/O	SPIM 从机选择脚
		PC.3	MFP3	I/O	
		PG.11	MFP4	I/O	
		PE.5	MFP4	I/O	
SWDH	SWDH_CLK	PA.10	MFP15	O	串行调试总线主时钟输出
	SWDH_DAT	PA.9	MFP15	I/O	串行调试总线主机数据输入/输出管脚
TAMPER0	TAMPER0	PF.6	MFP10	I/O	TAMPER循环检测管脚 0.
TAMPER1	TAMPER1	PF.7	MFP10	I/O	TAMPER循环检测管脚 1.
TAMPER2	TAMPER2	PF.8	MFP10	I/O	TAMPER循环检测管脚 2.
TAMPER3	TAMPER3	PF.9	MFP10	I/O	TAMPER循环检测管脚 3.
TAMPER4	TAMPER4	PF.10	MFP10	I/O	TAMPER循环检测管脚 4.
TAMPER5	TAMPER5	PF.11	MFP10	I/O	TAMPER循环检测管脚 5.
TM0	TM0	PB.5	MFP14	I/O	Timer0 事件计数输入/反转输出管脚
		PG.2	MFP13	I/O	
		PC.7	MFP14	I/O	
	TM0_EXT	PA.11	MFP13	I/O	Timer0 事件计数输入/反转输出管脚
		PH.0	MFP13	I/O	
		PB.15	MFP13	I/O	
TM1	TM1	PB.4	MFP14	I/O	Timer1 事件计数输入/反转输出脚
		PG.3	MFP13	I/O	
		PC.6	MFP14	I/O	
		PC.14	MFP13	I/O	
	TM1_EXT	PA.10	MFP13	I/O	Timer1 事件计数输入/反转输出脚
		PH.1	MFP13	I/O	
		PB.14	MFP13	I/O	
TM2	TM2	PB.3	MFP14	I/O	Timer2 事件计数输入/反转输出脚
		PG.4	MFP13	I/O	

组	管脚名称	GPIO	MFP	类型	描述
		PA.7	MFP14	I/O	
		PD.0	MFP14	I/O	
	TM2_EXT	PA.9	MFP13	I/O	Timer2 事件计数输入/反转输出脚
		PH.2	MFP13	I/O	
		PB.13	MFP13	I/O	
TM3	TM3	PB.2	MFP14	I/O	Timer3 事件计数输入/反转输出脚
		PF.11	MFP13	I/O	
		PA.6	MFP14	I/O	
	TM3_EXT	PA.8	MFP13	I/O	Timer3 事件计数输入/反转输出脚
		PH.3	MFP13	I/O	
		PB.12	MFP13	I/O	
TRACE	TRACE_CLK	PE.8	MFP14	O	ETM跟踪时钟输出脚
	TRACE_DATA0	PE.9	MFP14	O	ETM跟踪数据0 输出管脚
	TRACE_DATA1	PE.10	MFP14	O	ETM跟踪数据1 输出管脚
	TRACE_DATA2	PE.11	MFP14	O	ETM跟踪数据2 输出管脚
	TRACE_DATA3	PE.12	MFP14	O	ETM跟踪数据3 输出管脚
UART0	UART0_RXD	PC.11	MFP3	I	UART0 数据接收输入脚
		PF.2	MFP3	I	
		PA.6	MFP7	I	
		PA.0	MFP7	I	
		PD.2	MFP9	I	
		PA.15	MFP3	I	
		PH.11	MFP8	I	
		PB.12	MFP6	I	
		PB.8	MFP5	I	
	UART0_TXD	PC.12	MFP3	O	UART0 数据发送输出脚.
		PF.3	MFP3	O	
		PA.7	MFP7	O	
		PA.1	MFP7	O	
		PD.3	MFP9	O	
		PA.14	MFP3	O	
		PH.10	MFP8	O	
		PB.13	MFP6	O	

组	管脚名称	GPIO	MFP	类型	描述
		PB.9	MFP5	O	
	UART0_nCTS	PC.7	MFP7	I	UART0 禁止发送输入管脚
		PA.5	MFP7	I	
		PB.15	MFP6	I	
		PB.11	MFP5	I	
	UART0_nRTS	PC.6	MFP7	O	UART0 请求发送输出脚
		PA.4	MFP7	O	
		PB.14	MFP6	O	
		PB.10	MFP5	O	
UART1	UART1_RXD	PB.2	MFP6	I	UART1数据接收输入管脚
		PA.8	MFP7	I	
		PD.10	MFP3	I	
		PG.1	MFP8	I	
		PC.8	MFP8	I	
		PA.2	MFP8	I	
		PF.1	MFP2	I	
		PD.6	MFP3	I	
		PH.9	MFP10	I	
		PB.6	MFP6	I	
	UART1_TXD	PB.3	MFP6	O	UART1 数据发送输出脚
		PA.9	MFP7	O	
		PD.11	MFP3	O	
		PG.0	MFP8	O	
		PE.13	MFP8	O	
		PA.3	MFP8	O	
		PF.0	MFP2	O	
		PD.7	MFP3	O	
		PH.8	MFP10	O	
		PB.7	MFP6	O	
	UART1_nCTS	PE.11	MFP8	I	UART1 禁止发送输入管脚
		PA.1	MFP8	I	
		PB.9	MFP6	I	
	UART1_nRTS	PE.12	MFP8	O	UART1 请求发送输出脚

组	管脚名称	GPIO	MFP	类型	描述
		PA.0	MFP8	O	
		PB.8	MFP6	O	
UART2	UART2_RXD	PB.0	MFP7	I	UART2 数据接收输入脚
		PD.12	MFP7	I	
		PG.0	MFP6	I	
		PF.5	MFP2	I	
		PE.9	MFP7	I	
		PE.15	MFP3	I	
		PC.4	MFP8	I	
		PC.0	MFP8	I	
	UART2_TXD	PB.1	MFP7	O	UART2 数据发送输出脚.
		PC.13	MFP7	O	
		PG.1	MFP6	O	
		PF.4	MFP2	O	
		PE.8	MFP7	O	
		PE.14	MFP3	O	
		PC.5	MFP8	O	
		PC.1	MFP8	O	
	UART2_nCTS	PF.5	MFP4	I	UART2 禁止发送输入管脚
		PD.9	MFP4	I	
		PC.2	MFP8	I	
	UART2_nRTS	PF.4	MFP4	O	UART2 请求发送输出脚
		PD.8	MFP4	O	
		PC.3	MFP8	O	
UART3	UART3_RXD	PC.9	MFP7	I	UART3 数据接收输入脚
		PE.11	MFP7	I	
		PC.2	MFP11	I	
		PD.0	MFP5	I	
		PE.0	MFP7	I	
		PB.14	MFP7	I	
	UART3_TXD	PC.10	MFP7	O	UART3 数据发送输出脚.
		PE.10	MFP7	O	
		PC.3	MFP11	O	

组	管脚名称	GPIO	MFP	类型	描述
		PD.1	MFP5	O	
		PE.1	MFP7	O	
		PB.15	MFP7	O	
	UART3_nCTS	PD.2	MFP5	I	UART3 禁止发送输入管脚
		PH.9	MFP7	I	
		PB.12	MFP7	I	
	UART3_nRTS	PD.3	MFP5	O	UART3 请求发送输出脚
		PH.8	MFP7	O	
		PB.13	MFP7	O	
UART4	UART4_RXD	PF.6	MFP6	I	UART4 数据接收输入脚
		PH.3	MFP5	I	
		PC.6	MFP5	I	
		PA.2	MFP7	I	
		PC.4	MFP11	I	
		PA.13	MFP3	I	
		PH.11	MFP7	I	
		PB.10	MFP6	I	
	UART4_TXD	PF.7	MFP6	O	UART4 数据发送输出脚.
		PH.2	MFP5	O	
		PC.7	MFP5	O	
		PA.3	MFP7	O	
		PC.5	MFP11	O	
		PA.12	MFP3	O	
		PH.10	MFP7	O	
		PB.11	MFP6	O	
	UART4_nCTS	PC.8	MFP5	I	UART4 禁止发送输入管脚
		PE.1	MFP9	I	
	UART4_nRTS	PE.13	MFP5	O	UART4 请求发送输出脚
		PE.0	MFP9	O	
UART5	UART5_RXD	PB.4	MFP7	I	UART5 数据接收输入管脚
		PH.1	MFP4	I	
		PA.4	MFP8	I	
		PE.6	MFP8	I	

组	管脚名称	GPIO	MFP	类型	描述
	UART5_TXD	PB.5	MFP7	O	UART5数据发送输出脚
		PH.0	MFP4	O	
		PA.5	MFP8	O	
		PE.7	MFP8	O	
	UART5_nCTS	PB.2	MFP7	I	UART5 禁止发送输入管脚
		PH.3	MFP4	I	
	UART5_nRTS	PB.3	MFP7	O	UART5 请求发送输出脚
		PH.2	MFP4	O	
USB	USB_D+	PA.14	MFP14	A	USB差分信号D-.
	USB_D-	PA.13	MFP14	A	USB差分信号D+.
	USB_OTG_ID	PA.15	MFP14	I	USB_ 标识
	USB_VBUS	PA.12	MFP14	P	USB主机或HUB的电源
	USB_VBUS_EN	PB.15	MFP14	O	USB 外部 VBUS 电源使能管脚.
		PB.6	MFP14	O	
	USB_VBUS_ST	PD.4	MFP14	I	USB 外部VBUS 电源状态管脚.
		PC.14	MFP14	I	
		PB.7	MFP14	I	
USCI0	USCI0_CLK	PA.11	MFP6	I/O	USCI0 时钟脚
		PD.0	MFP3	I/O	
		PE.2	MFP7	I/O	
		PB.12	MFP5	I/O	
	USCI0_CTL0	PC.13	MFP6	I/O	USCI0 控制0 管脚.
		PD.4	MFP3	I/O	
		PE.6	MFP7	I/O	
		PC.14	MFP5	I/O	
	USCI0_CTL1	PA.8	MFP6	I/O	USCI0 控制1 管脚.
		PD.3	MFP3	I/O	
		PE.5	MFP7	I/O	
		PB.15	MFP5	I/O	
	USCI0_DAT0	PA.10	MFP6	I/O	USCI0 数据0脚
		PD.1	MFP3	I/O	
		PE.3	MFP7	I/O	
		PB.13	MFP5	I/O	

组	管脚名称	GPIO	MFP	类型	描述
	USCI0_DAT1	PA.9	MFP6	I/O	USCI0 数据1 管脚.
		PD.2	MFP3	I/O	
		PE.4	MFP7	I/O	
		PB.14	MFP5	I/O	
USCI1	USCI1_CLK	PB.1	MFP8	I/O	USCI1 时钟脚
		PE.12	MFP6	I/O	
		PD.7	MFP6	I/O	
		PB.8	MFP4	I/O	
	USCI1_CTL0	PB.5	MFP8	I/O	USCI1 控制0管脚
		PE.9	MFP6	I/O	
		PD.3	MFP6	I/O	
		PB.10	MFP4	I/O	
	USCI1_CTL1	PB.4	MFP8	I/O	USCI1 控制1管脚
		PE.8	MFP6	I/O	
		PD.4	MFP6	I/O	
		PB.9	MFP4	I/O	
	USCI1_DAT0	PB.2	MFP8	I/O	USCI1 数据0脚
		PE.10	MFP6	I/O	
		PD.5	MFP6	I/O	
		PB.7	MFP4	I/O	
	USCI1_DAT1	PB.3	MFP8	I/O	USCI1数据1管脚.
		PE.11	MFP6	I/O	
		PD.6	MFP6	I/O	
		PB.6	MFP4	I/O	
X32	X32_IN	PF.5	MFP10	I	外部 32.768 kHz 晶振输入脚
	X32_OUT	PF.4	MFP10	O	外部 32.768 kHz 晶振输出脚
XT1	XT1_IN	PF.3	MFP10	I	外部 4~24 MHz (高速) 晶振输入脚
	XT1_OUT	PF.2	MFP10	O	外部 4~24 MHz (高速) 晶振输出脚

4.2.8 多功能管脚表以GPIO排列

	管脚名称	类型	MFP	描述
PA.0	PA.0	I/O	MFP0	通用数字I/O脚
	SPIM_MOSI	I/O	MFP2	SPIM MOSI (主机输出，从机输入) 管脚.
	QSPI0_MOSI0	I/O	MFP3	QSPI0 MOSI0 (主机输出，从机输入) 管脚.
	SPI0_MOSI	I/O	MFP4	SPI0 MOSI (主机输出，从机输入) 管脚.
	SD1_DAT0	I/O	MFP5	SD/SDIO1 数据线位0.
	SC0_CLK	O	MFP6	智能卡0时钟管脚
	UART0_RXD	I	MFP7	UART0 数据接收输入脚
	UART1_nRTS	O	MFP8	UART1 请求发送输出脚
	I^2C2_SDA	I/O	MFP9	I^2C2 数据输入/输出脚
	BPWM0_CH0	I/O	MFP12	BPWM0 通道0 输出/捕获输入
	EPWM0_CH5	I/O	MFP13	EPWM0 通道5 输出/捕获输入
	DAC0_ST	I	MFP15	DAC0 外部触发输入
PA.1	PA.1	I/O	MFP0	通用数字I/O脚
	SPIM_MISO	I/O	MFP2	SPIM MISO（主机输入，从机输出）脚.
	QSPI0_MISO0	I/O	MFP3	QSPI0 MISO0（主机输入，从机输出）脚.
	SPI0_MISO	I/O	MFP4	SPI0 MISO（主机输入，从机输出）脚.
	SD1_DAT1	I/O	MFP5	SD/SDIO1 数据线位1.
	SC0_DAT	I/O	MFP6	智能卡0数据输入脚
	UART0_TXD	O	MFP7	UART0 数据发送输出脚.
	UART1_nCTS	I	MFP8	UART1 禁止发送输入管脚
	I^2C2_SCL	I/O	MFP9	I^2C2 时钟脚
	BPWM0_CH1	I/O	MFP12	BPWM0 通道1 输出/捕获输入
	EPWM0_CH4	I/O	MFP13	EPWM0 通道4输出/捕获输入脚
	DAC1_ST	I	MFP15	DAC1 外部触发输入
PA.2	PA.2	I/O	MFP0	通用数字I/O脚
	SPIM_CLK	I/O	MFP2	SPIM 串行时钟脚
	QSPI0_CLK	I/O	MFP3	QSPI0 串行时钟脚
	SPI0_CLK	I/O	MFP4	SPI0 串行时钟脚
	SD1_DAT2	I/O	MFP5	SD/SDIO1 数据线位2.
	SC0_RST	O	MFP6	智能卡0复位脚

	管脚名称	类型	MFP	描述
	UART4_RXD	I	MFP7	UART4 数据接收输入脚
	UART1_RXD	I	MFP8	UART1数据接收输入管脚
	I²C1_SDA	I/O	MFP9	I²C1 数据输入/输出脚
	BPWM0_CH2	I/O	MFP12	BPWM0 通道2 输出/捕获输入
	EPWM0_CH3	I/O	MFP13	EPWM0 通道3输出/捕获输入脚
PA.3	PA.3	I/O	MFP0	通用数字I/O脚
	SPIM_SS	I/O	MFP2	SPIM 从机选择脚
	QSPI0_SS	I/O	MFP3	QSPI0 从机选择脚
	SPI0_SS	I/O	MFP4	SPI0 从机选择脚
	SD1_DAT3	I/O	MFP5	SD/SDIO1 数据线位3.
	SC0_PWR	O	MFP6	智能卡0电源脚
	UART4_TXD	O	MFP7	UART4 数据发送输出脚.
	UART1_TXD	O	MFP8	UART1 数据发送输出脚
	I²C1_SCL	I/O	MFP9	I²C1 时钟脚
	BPWM0_CH3	I/O	MFP12	BPWM0 通道3输出/捕获输入脚
	EPWM0_CH2	I/O	MFP13	EPWM0 通道2输出/捕获输入脚
	QEI0_B	I	MFP14	正交编码器0 相位B 输入
PA.4	PA.4	I/O	MFP0	通用数字I/O脚
	SPIM_D3	I/O	MFP2	SPIM QuadI/O模式下数据3管脚
	QSPI0_MOSI1	I/O	MFP3	QSPI0 MOSI1 (主机输出，从机输入) 管脚.
	SPI0_I2SMCLK	I/O	MFP4	SPI0 I²S 主机时钟输出脚
	SD1_CLK	O	MFP5	SD/SDIO1 时钟输出管脚
	SC0_nCD	I	MFP6	智能卡 0 卡检测管脚.
	UART0_nRTS	O	MFP7	UART0 请求发送输出脚
	UART5_RXD	I	MFP8	UART5 数据接收输入管脚
	I²C0_SDA	I/O	MFP9	I2C0 数据输入/输出管脚
	CAN0_RXD	I	MFP10	CAN0 总线接收输入
	BPWM0_CH4	I/O	MFP12	BPWM0 通道4输出/捕获输入脚
	EPWM0_CH1	I/O	MFP13	EPWM0 通道1输出/捕获输入脚
	QEI0_A	I	MFP14	正交编码器0 相位A输入
PA.5	PA.5	I/O	MFP0	通用数字I/O脚

	管脚名称	类型	MFP	描述
	SPIM_D2	I/O	MFP2	SPIM QuadI/O模式下数据2管脚
	QSPI0_MISO1	I/O	MFP3	QSPI0 MISO1（主机输入，从机输出）脚.
	SPI1_I2SMCLK	I/O	MFP4	SPI1 I^2S 主机时钟输出脚
	SD1_CMD	I/O	MFP5	SD/SDIO1 命令/响应脚
	SC2_nCD	I	MFP6	智能卡 2 卡检测管脚.
	UART0_nCTS	I	MFP7	UART0 禁止发送输入管脚
	UART5_TXD	O	MFP8	UART5数据发送输出脚
	I^2C0_SCL	I/O	MFP9	I2C0 时钟管脚
	CAN0_TXD	O	MFP10	CAN0总线发送输出
	BPWM0_CH5	I/O	MFP12	BPWM0 通道5 输出/捕获输入
	EPWM0_CH0	I/O	MFP13	EPWM0 通道0输出/捕获输入
	QEI0_INDEX	I	MFP14	正交编码器0 索引输入
PA.6	PA.6	I/O	MFP0	通用数字I/O脚
	EBI_AD6	I/O	MFP2	EBI地址/数据总线位6.
	EMAC_RMII_RXERR	I	MFP3	EMAC RMII 接收数据错误输入脚
	SPI1_SS	I/O	MFP4	SPI1 从机选择脚
	SD1_nCD	I	MFP5	SD/SDIO1 卡检测输入管脚
	SC2_CLK	O	MFP6	智能卡 2 时钟脚
	UART0_RXD	I	MFP7	UART0 数据接收输入脚
	I^2C1_SDA	I/O	MFP8	I^2C1 数据输入/输出脚
	EPWM1_CH5	I/O	MFP11	EPWM1 通道5输出/捕获输入
	BPWM1_CH3	I/O	MFP12	BPWM1 通道3输出/捕获输入脚
	ACMP1_WLAT	I	MFP13	模拟比较器 1窗口锁存输入管脚
	TM3	I/O	MFP14	Timer3 事件计数输入/反转输出脚
	INT0	I	MFP15	外部中断0输入管脚
PA.7	PA.7	I/O	MFP0	通用数字I/O脚
	EBI_AD7	I/O	MFP2	EBI地址/数据总线位7.
	EMAC_RMII_CRSDV	I	MFP3	EMAC RMII 载波侦测/接收数据输入脚
	SPI1_CLK	I/O	MFP4	SPI1 串行时钟管脚
	SC2_DAT	I/O	MFP6	智能卡 2 数据管脚.
	UART0_TXD	O	MFP7	UART0 数据发送输出脚.

	管脚名称	类型	MFP	描述
	I²C1_SCL	I/O	MFP8	I²C1 时钟脚
	EPWM1_CH4	I/O	MFP11	EPWM1 通道4输出/捕获输入脚
	BPWM1_CH2	I/O	MFP12	BPWM1 通道2 输出/捕获输入
	ACMP0_WLAT	I	MFP13	模拟比较器 0窗口锁存输入管脚
	TM2	I/O	MFP14	Timer2 事件计数输入/反转输出脚
	INT1	I	MFP15	外部中断1输入脚
PA.8	PA.8	I/O	MFP0	通用数字I/O脚
	OPA1_P	A	MFP1	运放 1 正极输入脚
	EBI_ALE	O	MFP2	EBI地址锁存使能输出脚
	SC2_CLK	O	MFP3	智能卡 2 时钟脚
	SPI2_MOSI	I/O	MFP4	SPI2 MOSI (主机输出，从机输入) 管脚.
	SD1_DAT0	I/O	MFP5	SD/SDIO1 数据线位0.
	USCI0_CTL1	I/O	MFP6	USCI0 控制1 管脚.
	UART1_RXD	I	MFP7	UART1数据接收输入管脚
	BPWM0_CH3	I/O	MFP9	BPWM0 通道3输出/捕获输入脚
	QEI1_B	I	MFP10	正交编码器1 相位B 输入
	ECAP0_IC2	I	MFP11	增强型捕获计数器 0 输入 2 管脚.
	TM3_EXT	I/O	MFP13	Timer3 事件计数输入/反转输出脚
	INT4	I	MFP15	外部中断4输入管脚
PA.9	PA.9	I/O	MFP0	通用数字I/O脚
	OPA1_N	A	MFP1	运放 1 负极输入脚
	EBI_MCLK	O	MFP2	EBI 外部时钟输出脚.
	SC2_DAT	I/O	MFP3	智能卡 2 数据管脚.
	SPI2_MISO	I/O	MFP4	SPI2 MISO（主机输入，从机输出）脚.
	SD1_DAT1	I/O	MFP5	SD/SDIO1 数据线位1.
	USCI0_DAT1	I/O	MFP6	USCI0 数据1 管脚.
	UART1_TXD	O	MFP7	UART1 数据发送输出脚
	BPWM0_CH2	I/O	MFP9	BPWM0 通道2 输出/捕获输入
	QEI1_A	I	MFP10	正交编码器1 相位A输入
	ECAP0_IC1	I	MFP11	增强型捕获计数器 0 输入 1 管脚.
	TM2_EXT	I/O	MFP13	Timer2 事件计数输入/反转输出脚

	管脚名称	类型	MFP	描述
	SWDH_DAT	I/O	MFP15	串行调试总线主机数据输入/输出管脚
PA.10	PA.10	I/O	MFP0	通用数字I/O脚
	ACMP1_P0	A	MFP1	模拟比较器 1 正极输入脚0
	OPA1_O	A	MFP1	运放 1输出脚
	EBI_nWR	O	MFP2	EBI 写使能输出脚
	SC2_RST	O	MFP3	智能卡 2 复位脚
	SPI2_CLK	I/O	MFP4	SPI2 串行时钟脚
	SD1_DAT2	I/O	MFP5	SD/SDIO1 数据线位2.
	USCI0_DAT0	I/O	MFP6	USCI0 数据0脚
	I^2C2_SDA	I/O	MFP7	I^2C2 数据输入/输出脚
	BPWM0_CH1	I/O	MFP9	BPWM0 通道1 输出/捕获输入
	QEI1_INDEX	I	MFP10	正交编码器1 索引输入
	ECAP0_IC0	I	MFP11	增强型捕获计数器 0 输入脚0
	TM1_EXT	I/O	MFP13	Timer1 事件计数输入/反转输出脚
	DAC0_ST	I	MFP14	DAC0 外部触发输入
	SWDH_CLK	O	MFP15	串行调试总线主时钟输出
PA.11	PA.11	I/O	MFP0	通用数字I/O脚
	ACMP0_P0	A	MFP1	模拟比较器 0 正极输入脚0
	EBI_nRD	O	MFP2	EBI 读使能输出脚
	SC2_PWR	O	MFP3	智能卡 2电源脚
	SPI2_SS	I/O	MFP4	SPI2 从机选择脚
	SD1_DAT3	I/O	MFP5	SD/SDIO1 数据线位3.
	USCI0_CLK	I/O	MFP6	USCI0 时钟脚
	I^2C2_SCL	I/O	MFP7	I^2C2 时钟脚
	BPWM0_CH0	I/O	MFP9	BPWM0 通道0 输出/捕获输入
	EPWM0_SYNC_OUT	O	MFP10	EPWM0计数器同步触发输出脚
	TM0_EXT	I/O	MFP13	Timer0 事件计数输入/反转输出管脚
	DAC1_ST	I	MFP14	DAC1 外部触发输入
PA.12	PA.12	I/O	MFP0	通用数字I/O脚
	I^2S0_BCLK	O	MFP2	I2S0位时钟输出管脚
	UART4_TXD	O	MFP3	UART4 数据发送输出脚.

	管脚名称	类型	MFP	描述
	I²C1_SCL	I/O	MFP4	I²C1 时钟脚
	SPI2_SS	I/O	MFP5	SPI2 从机选择脚
	CAN0_TXD	O	MFP6	CAN0总线发送输出
	SC2_PWR	O	MFP7	智能卡 2电源脚
	BPWM1_CH2	I/O	MFP11	BPWM1 通道2 输出/捕获输入
	QEI1_INDEX	I	MFP12	正交编码器1 索引输入
	USB_VBUS	P	MFP14	USB主机或HUB的电源
PA.13	PA.13	I/O	MFP0	通用数字I/O脚
	I²S0_MCLK	O	MFP2	I2S0 主时钟输出脚
	UART4_RXD	I	MFP3	UART4 数据接收输入脚
	I²C1_SDA	I/O	MFP4	I²C1 数据输入/输出脚
	SPI2_CLK	I/O	MFP5	SPI2 串行时钟脚
	CAN0_RXD	I	MFP6	CAN0 总线接收输入
	SC2_RST	O	MFP7	智能卡 2 复位脚
	BPWM1_CH3	I/O	MFP11	BPWM1 通道3输出/捕获输入脚
	QEI1_A	I	MFP12	正交编码器1 相位A输入
	USB_D-	A	MFP14	USB差分信号D+.
PA.14	PA.14	I/O	MFP0	通用数字I/O脚
	I²S0_DI	I	MFP2	I2S0 数据输入脚
	UART0_TXD	O	MFP3	UART0 数据发送输出脚.
	SPI2_MISO	I/O	MFP5	SPI2 MISO（主机输入，从机输出）脚.
	I²C2_SCL	I/O	MFP6	I²C2 时钟脚
	SC2_DAT	I/O	MFP7	智能卡 2 数据管脚.
	BPWM1_CH4	I/O	MFP11	BPWM1 通道4输出/捕获输入脚
	QEI1_B	I	MFP12	正交编码器1 相位B 输入
	USB_D+	A	MFP14	USB差分信号D-.
PA.15	PA.15	I/O	MFP0	通用数字I/O脚
	I²S0_DO	O	MFP2	I²S0 数据输出脚
	UART0_RXD	I	MFP3	UART0 数据接收输入脚
	SPI2_MOSI	I/O	MFP5	SPI2 MOSI (主机输出，从机输入) 管脚.
	I²C2_SDA	I/O	MFP6	I²C2 数据输入/输出脚

	管脚名称	类型	MFP	描述
	SC2_CLK	O	MFP7	智能卡 2 时钟脚
	BPWM1_CH5	I/O	MFP11	EPWM1 通道5输出/捕获输入
	EPWM0_SYNC_IN	I	MFP12	EPWM0 计数器同步触发输入脚
	USB_OTG_ID	I	MFP14	USB_ 标识
PB.0	PB.0	I/O	MFP0	通用数字I/O脚
	EADC0_CH0	A	MFP1	EADC0 通道0 模拟输入
	OPA0_P	A	MFP1	运放 0 正极输入脚
	EBI_ADR9	O	MFP2	EBI 地址总线位 9.
	SD0_CMD	I/O	MFP3	SD/SDIO0 命令/响应脚
	UART2_RXD	I	MFP7	UART2 数据接收输入脚
	SPI0_I2SMCLK	I/O	MFP8	SPI0 I^2S 主机时钟输出脚
	I^2C1_SDA	I/O	MFP9	I^2C1 数据输入/输出脚
	EPWM0_CH5	I/O	MFP11	EPWM0 通道5 输出/捕获输入
	EPWM1_CH5	I/O	MFP12	EPWM1 通道5输出/捕获输入
	EPWM0_BRAKE1	I	MFP13	EPWM0 刹车1 输入脚
PB.1	PB.1	I/O	MFP0	通用数字I/O脚
	EADC0_CH1	A	MFP1	EADC0 通道1 模拟输入
	OPA0_N	A	MFP1	运放0负极输入脚
	EBI_ADR8	O	MFP2	EBI 地址总线位 8.
	SD0_CLK	O	MFP3	SD/SDIO0 时钟输出脚
	EMAC_RMII_RXERR	I	MFP4	EMAC RMII 接收数据错误输入脚
	SPI1_I2SMCLK	I/O	MFP5	SPI1 I^2S 主机时钟输出脚
	SPI3_I2SMCLK	I/O	MFP6	SPI3 I^2S 主机时钟输出脚
	UART2_TXD	O	MFP7	UART2 数据发送输出脚.
	USCI1_CLK	I/O	MFP8	USCI1 时钟脚
	I^2C1_SCL	I/O	MFP9	I^2C1 时钟脚
	I^2S0_LRCK	O	MFP10	I^2S0左右通道时钟输出脚.
	EPWM0_CH4	I/O	MFP11	EPWM0 通道4输出/捕获输入脚
	EPWM1_CH4	I/O	MFP12	EPWM1 通道4输出/捕获输入脚
	EPWM0_BRAKE0	I	MFP13	EPWM0 刹车0 输入脚
PB.2	PB.2	I/O	MFP0	通用数字I/O脚

	管脚名称	类型	MFP	描述
	EADC0_CH2	A	MFP1	EADC0 通道2 模拟输入
	ACMP0_P1	A	MFP1	模拟比较器0正极输入1脚
	OPA0_O	A	MFP1	运放0输出脚
	EBI_ADR3	O	MFP2	EBI 地址总线位3
	SD0_DAT0	I/O	MFP3	SD/SDIO0 数据线位0
	EMAC_RMII_CRSDV	I	MFP4	EMAC RMII 载波侦测/接收数据输入脚
	SPI1_SS	I/O	MFP5	SPI1 从机选择脚
	UART1_RXD	I	MFP6	UART1数据接收输入管脚
	UART5_nCTS	I	MFP7	UART5 禁止发送输入管脚
	USCI1_DAT0	I/O	MFP8	USCI1 数据0脚
	SC0_PWR	O	MFP9	智能卡0电源脚
	I^2S0_DO	O	MFP10	I^2S0 数据输出脚
	EPWM0_CH3	I/O	MFP11	EPWM0 通道3输出/捕获输入脚
	TM3	I/O	MFP14	Timer3 事件计数输入/反转输出脚
	INT3	I	MFP15	外部中断3输入脚
PB.3	PB.3	I/O	MFP0	通用数字I/O脚
	EADC0_CH3	A	MFP1	EADC0 通道3模拟输入
	ACMP0_N	A	MFP1	模拟比较器0负极输入脚
	EBI_ADR2	O	MFP2	EBI 地址总线位2
	SD0_DAT1	I/O	MFP3	SD/SDIO0 数据线位1
	EMAC_RMII_RXD1	I	MFP4	EMAC RMII接收数据总线位 1.
	SPI1_CLK	I/O	MFP5	SPI1 串行时钟管脚
	UART1_TXD	O	MFP6	UART1 数据发送输出脚
	UART5_nRTS	O	MFP7	UART5 请求发送输出脚
	USCI1_DAT1	I/O	MFP8	USCI1数据1管脚.
	SC0_RST	O	MFP9	智能卡0复位脚
	I^2S0_DI	I	MFP10	I2S0 数据输入脚
	EPWM0_CH2	I/O	MFP11	EPWM0 通道2输出/捕获输入脚
	TM2	I/O	MFP14	Timer2 事件计数输入/反转输出脚
	INT2	I	MFP15	外部中断2输入脚
PB.4	PB.4	I/O	MFP0	通用数字I/O脚

	管脚名称	类型	MFP	描述
	EADC0_CH4	A	MFP1	EADC0 通道4模拟输入
	ACMP1_P1	A	MFP1	模拟比较器1正极输入1脚
	EBI_ADR1	O	MFP2	EBI 地址总线位1
	SD0_DAT2	I/O	MFP3	SD/SDIO0 数据线位2
	EMAC_RMII_RXD0	I	MFP4	EMAC RMII接收数据总线位 0.
	SPI1_MOSI	I/O	MFP5	SPI1 MOSI (主机输出，从机输入) 管脚
	I^2C0_SDA	I/O	MFP6	I2C0 数据输入/输出管脚
	UART5_RXD	I	MFP7	UART5 数据接收输入管脚
	USCI1_CTL1	I/O	MFP8	USCI1 控制1管脚
	SC0_DAT	I/O	MFP9	智能卡0数据输入脚
	I^2S0_MCLK	O	MFP10	I2S0 主时钟输出脚
	EPWM0_CH1	I/O	MFP11	EPWM0 通道1输出/捕获输入脚
	TM1	I/O	MFP14	Timer1 事件计数输入/反转输出脚
	INT1	I	MFP15	外部中断1输入脚
PB.5	PB.5	I/O	MFP0	通用数字I/O脚
	EADC0_CH5	A	MFP1	EADC0 通道5模拟输入
	ACMP1_N	A	MFP1	模拟比较器1负极输入管脚
	EBI_ADR0	O	MFP2	EBI地址总线位0
	SD0_DAT3	I/O	MFP3	SD/SDIO0数据线位3
	EMAC_RMII_REFCLK	I	MFP4	EMAC RMII 参考时钟输入脚
	SPI1_MISO	I/O	MFP5	SPI1 MISO(主机输入，从机输出)管脚
	I^2C0_SCL	I/O	MFP6	I2C0 时钟管脚
	UART5_TXD	O	MFP7	UART5数据发送输出脚
	USCI1_CTL0	I/O	MFP8	USCI1 控制0管脚
	SC0_CLK	O	MFP9	智能卡0时钟管脚
	I^2S0_BCLK	O	MFP10	I2S0位时钟输出管脚
	EPWM0_CH0	I/O	MFP11	EPWM0 通道0输出/捕获输入
	TM0	I/O	MFP14	Timer0 事件计数输入/反转输出管脚
	INT0	I	MFP15	外部中断0输入管脚
PB.6	PB.6	I/O	MFP0	通用数字I/O脚
	EADC0_CH6	A	MFP1	EADC0通道6模拟输入

	管脚名称	类型	MFP	描述
	EBI_nWRH	O	MFP2	EBI 高字节写使能输出管脚
	EMAC_PPS	O	MFP3	EMAC 每秒脉冲输出脚
	USCI1_DAT1	I/O	MFP4	USCI1数据1管脚.
	CAN1_RXD	I	MFP5	CAN1 总线接收输入
	UART1_RXD	I	MFP6	UART1数据接收输入管脚
	SD1_CLK	O	MFP7	SD/SDIO1 时钟输出管脚
	EBI_nCS1	O	MFP8	EBI片选1输出管脚
	BPWM1_CH5	I/O	MFP10	EPWM1 通道5输出/捕获输入
	EPWM1_BRAKE1	I	MFP11	EPWM1 刹车1输入管脚
	EPWM1_CH5	I/O	MFP12	EPWM1 通道5输出/捕获输入
	INT4	I	MFP13	外部中断4输入管脚
	USB_VBUS_EN	O	MFP14	USB 外部 VBUS 电源使能管脚.
	ACMP1_O	O	MFP15	模拟比较器1输出脚
PB.7	PB.7	I/O	MFP0	通用数字I/O脚
	EADC0_CH7	A	MFP1	EADC0 通道7 模拟输入
	EBI_nWRL	O	MFP2	EBI低字节写使能输出脚
	EMAC_RMII_TXEN	O	MFP3	EMAC RMII 发送使能输出脚
	USCI1_DAT0	I/O	MFP4	USCI1 数据0脚
	CAN1_TXD	O	MFP5	CAN1总线发送输出
	UART1_TXD	O	MFP6	UART1 数据发送输出脚
	SD1_CMD	I/O	MFP7	SD/SDIO1 命令/响应脚
	EBI_nCS0	O	MFP8	EBI片选0输出脚
	BPWM1_CH4	I/O	MFP10	BPWM1 通道4输出/捕获输入脚
	EPWM1_BRAKE0	I	MFP11	EPWM1 刹车0 输入脚
	EPWM1_CH4	I/O	MFP12	EPWM1 通道4输出/捕获输入脚
	INT5	I	MFP13	外部中断5 输入脚
	USB_VBUS_ST	I	MFP14	USB 外部VBUS 电源状态管脚.
	ACMP0_O	O	MFP15	模拟比较器 0输出脚
PB.8	PB.8	I/O	MFP0	通用数字I/O脚
	EADC0_CH8	A	MFP1	EADC0 通道8 模拟输入
	EBI_ADR19	O	MFP2	EBI 地址总线位 19.

	管脚名称	类型	MFP	描述
	EMAC_RMII_TXD1	O	MFP3	EMAC RMII 发送数据总线位1.
	USCI1_CLK	I/O	MFP4	USCI1 时钟脚
	UART0_RXD	I	MFP5	UART0 数据接收输入脚
	UART1_nRTS	O	MFP6	UART1 请求发送输出脚
	I²C1_SMBSUS	O	MFP7	I²C1 SMBus SMBSUS 管脚 (PMBus 控制管脚)
	BPWM1_CH3	I/O	MFP10	BPWM1 通道3输出/捕获输入脚
	SPI3_MOSI	I/O	MFP11	SPI3 MOSI (主机输出，从机输入) 管脚.
	INT6	I	MFP13	外部中断6 输入脚
PB.9	PB.9	I/O	MFP0	通用数字I/O脚
	EADC0_CH9	A	MFP1	EADC0 通道9 模拟输入
	EBI_ADR18	O	MFP2	EBI 地址总线位 18.
	EMAC_RMII_TXD0	O	MFP3	EMAC RMII 发送数据总线位0.
	USCI1_CTL1	I/O	MFP4	USCI1 控制1管脚
	UART0_TXD	O	MFP5	UART0 数据发送输出脚.
	UART1_nCTS	I	MFP6	UART1 禁止发送输入管脚
	I²C1_SMBAL	O	MFP7	I²C1 SMBus SMBALTER 管脚
	BPWM1_CH2	I/O	MFP10	BPWM1 通道2 输出/捕获输入
	SPI3_MISO	I/O	MFP11	SPI3 MISO（主机输入，从机输出）脚.
	INT7	I	MFP13	外部中断7 输入脚
PB.10	PB.10	I/O	MFP0	通用数字I/O脚
	EADC0_CH10	A	MFP1	EADC0 通道10 模拟输入
	EBI_ADR17	O	MFP2	EBI 地址总线位 17.
	EMAC_RMII_MDIO	I/O	MFP3	EMAC RMII PHY管理数据管脚.
	USCI1_CTL0	I/O	MFP4	USCI1 控制0管脚
	UART0_nRTS	O	MFP5	UART0 请求发送输出脚
	UART4_RXD	I	MFP6	UART4 数据接收输入脚
	I²C1_SDA	I/O	MFP7	I²C1 数据输入/输出脚
	CAN0_RXD	I	MFP8	CAN0 总线接收输入
	BPWM1_CH1	I/O	MFP10	BPWM1 通道1 输出/捕获输入
	SPI3_SS	I/O	MFP11	SPI3 从机选择脚
	HSUSB_VBUS_EN	O	MFP14	HSUSB 外部 VBUS 电源使能管脚.

	管脚名称	类型	MFP	描述
PB.11	PB.11	I/O	MFP0	通用数字I/O脚
	EADC0_CH11	A	MFP1	EADC0 通道11 模拟输入
	EBI_ADR16	O	MFP2	EBI 地址总线位 16.
	EMAC_RMII_MDC	O	MFP3	EMAC RMII PHY管理时钟输出脚.
	UART0_nCTS	I	MFP5	UART0 禁止发送输入管脚
	UART4_TXD	O	MFP6	UART4 数据发送输出脚.
	I^2C1_SCL	I/O	MFP7	I^2C1 时钟脚
	CAN0_TXD	O	MFP8	CAN0总线发送输出
	SPI0_I2SMCLK	I/O	MFP9	SPI0 I^2S 主机时钟输出脚
	BPWM1_CH0	I/O	MFP10	BPWM1 通道0 输出/捕获输入
	SPI3_CLK	I/O	MFP11	SPI3 串行时钟脚
	HSUSB_VBUS_ST	I	MFP14	HSUSB 外部VBUS 电源状态管脚.
PB.12	PB.12	I/O	MFP0	通用数字I/O脚
	EADC0_CH12	A	MFP1	EADC0 通道12 模拟输入
	DAC0_OUT	A	MFP1	DAC0 模拟输出通道
	ACMP0_P2	A	MFP1	模拟比较器 0 正极输入 2 管脚.
	ACMP1_P2	A	MFP1	模拟比较器 1 正极输入 2 管脚.
	EBI_AD15	I/O	MFP2	EBI地址/数据总线位15.
	SC1_CLK	O	MFP3	智能卡 1 时钟脚
	SPI0_MOSI	I/O	MFP4	SPI0 MOSI (主机输出，从机输入) 管脚.
	USCI0_CLK	I/O	MFP5	USCI0 时钟脚
	UART0_RXD	I	MFP6	UART0 数据接收输入脚
	UART3_nCTS	I	MFP7	UART3 禁止发送输入管脚
	I^2C2_SDA	I/O	MFP8	I^2C2 数据输入/输出脚
	SD0_nCD	I	MFP9	SD/SDIO0 卡检测输入管脚
	EPWM1_CH3	I/O	MFP11	EPWM1 通道3输出/捕获输入脚
	ETM_TRACE_DATA3	I	MFP12	ETM接收追踪数据3 输入管脚
	TM3_EXT	I/O	MFP13	Timer3 事件计数输入/反转输出脚
PB.13	PB.13	I/O	MFP0	通用数字I/O脚
	EADC0_CH13	A	MFP1	EADC0 通道13 模拟输入
	DAC1_OUT	A	MFP1	DAC1 模拟输出通道

	管脚名称	类型	MFP	描述
	ACMP0_P3	A	MFP1	模拟比较器 0 正极输入 3 管脚.
	ACMP1_P3	A	MFP1	模拟比较器 1 正极输入 3 管脚.
	EBI_AD14	I/O	MFP2	EBI地址/数据总线位14.
	SC1_DAT	I/O	MFP3	智能卡 1 数据管脚.
	SPI0_MISO	I/O	MFP4	SPI0 MISO（主机输入，从机输出）脚.
	USCI0_DAT0	I/O	MFP5	USCI0 数据0脚
	UART0_TXD	O	MFP6	UART0 数据发送输出脚.
	UART3_nRTS	O	MFP7	UART3 请求发送输出脚
	I^2C2_SCL	I/O	MFP8	I^2C2 时钟脚
	EPWM1_CH2	I/O	MFP11	EPWM1 通道2 输出/捕获输入
	ETM_TRACE_DATA2	I	MFP12	ETM接收追踪数据2 输入管脚
	TM2_EXT	I/O	MFP13	Timer2 事件计数输入/反转输出脚
PB.14	PB.14	I/O	MFP0	通用数字I/O脚
	EADC0_CH14	A	MFP1	EADC0 通道14 模拟输入
	EBI_AD13	I/O	MFP2	EBI地址/数据总线位13.
	SC1_RST	O	MFP3	智能卡 1 复位脚
	SPI0_CLK	I/O	MFP4	SPI0 串行时钟脚
	USCI0_DAT1	I/O	MFP5	USCI0 数据1 管脚.
	UART0_nRTS	O	MFP6	UART0 请求发送输出脚
	UART3_RXD	I	MFP7	UART3 数据接收输入脚
	I^2C2_SMBSUS	O	MFP8	I^2C2 SMBus SMBSUS 管脚 (PMBus 控制管脚)
	EPWM1_CH1	I/O	MFP11	EPWM1 通道1 输出/捕获输入
	ETM_TRACE_DATA1	I	MFP12	ETM接收追踪数据1 输入管脚
	TM1_EXT	I/O	MFP13	Timer1 事件计数输入/反转输出脚
	CLKO	O	MFP14	时钟输出
PB.15	PB.15	I/O	MFP0	通用数字I/O脚
	EADC0_CH15	A	MFP1	EADC0 通道15 模拟输入
	EBI_AD12	I/O	MFP2	EBI地址/数据总线位12.
	SC1_PWR	O	MFP3	智能卡 1电源脚
	SPI0_SS	I/O	MFP4	SPI0 从机选择脚
	USCI0_CTL1	I/O	MFP5	USCI0 控制1 管脚.

	管脚名称	类型	MFP	描述
	UART0_nCTS	I	MFP6	UART0 禁止发送输入管脚
	UART3_TXD	O	MFP7	UART3 数据发送输出脚.
	I²C2_SMBAL	O	MFP8	I²C2 SMBus SMBALTER 管脚
	EPWM1_CH0	I/O	MFP11	EPWM1 通道0 输出/捕获输入
	ETM_TRACE_DATA0	I	MFP12	ETM接收追踪数据0 输入管脚
	TM0_EXT	I/O	MFP13	Timer0 事件计数输入/反转输出管脚
	USB_VBUS_EN	O	MFP14	USB 外部 VBUS 电源使能管脚.
	HSUSB_VBUS_EN	O	MFP15	HSUSB 外部 VBUS 电源使能管脚.
PC.0	PC.0	I/O	MFP0	通用数字I/O脚
	EBI_AD0	I/O	MFP2	EBI地址/数据总线位0.
	SPIM_MOSI	I/O	MFP3	SPIM MOSI (主机输出，从机输入) 管脚.
	QSPI0_MOSI0	I/O	MFP4	QSPI0 MOSI0 (主机输出，从机输入) 管脚.
	SC1_CLK	O	MFP5	智能卡 1 时钟脚
	I²S0_LRCK	O	MFP6	I²S0左右通道时钟输出脚.
	SPI1_SS	I/O	MFP7	SPI1 从机选择脚
	UART2_RXD	I	MFP8	UART2 数据接收输入脚
	I²C0_SDA	I/O	MFP9	I2C0 数据输入/输出管脚
	EPWM1_CH5	I/O	MFP12	EPWM1 通道5输出/捕获输入
	ACMP1_O	O	MFP14	模拟比较器1输出脚
PC.1	PC.1	I/O	MFP0	通用数字I/O脚
	EBI_AD1	I/O	MFP2	EBI地址/数据总线位1.
	SPIM_MISO	I/O	MFP3	SPIM MISO（主机输入，从机输出）脚.
	QSPI0_MISO0	I/O	MFP4	QSPI0 MISO0（主机输入，从机输出）脚.
	SC1_DAT	I/O	MFP5	智能卡 1 数据管脚.
	I²S0_DO	O	MFP6	I²S0 数据输出脚
	SPI1_CLK	I/O	MFP7	SPI1 串行时钟管脚
	UART2_TXD	O	MFP8	UART2 数据发送输出脚.
	I²C0_SCL	I/O	MFP9	I2C0 时钟管脚
	EPWM1_CH4	I/O	MFP12	EPWM1 通道4输出/捕获输入脚
	ACMP0_O	O	MFP14	模拟比较器 0输出脚
PC.2	PC.2	I/O	MFP0	通用数字I/O脚

	管脚名称	类型	MFP	描述
	EBI_AD2	I/O	MFP2	EBI地址/数据总线位2.
	SPIM_CLK	I/O	MFP3	SPIM 串行时钟脚
	QSPI0_CLK	I/O	MFP4	QSPI0 串行时钟脚
	SC1_RST	O	MFP5	智能卡 1 复位脚
	I^{2}S0_DI	I	MFP6	I2S0 数据输入脚
	SPI1_MOSI	I/O	MFP7	SPI1 MOSI (主机输出，从机输入) 管脚
	UART2_nCTS	I	MFP8	UART2 禁止发送输入管脚
	I^{2}C0_SMBSUS	O	MFP9	I^{2}C0 SMBus SMBSUS 管脚 (PMBus 控制管脚)
	CAN1_RXD	I	MFP10	CAN1 总线接收输入
	UART3_RXD	I	MFP11	UART3 数据接收输入脚
	EPWM1_CH3	I/O	MFP12	EPWM1 通道3输出/捕获输入脚
PC.3	PC.3	I/O	MFP0	通用数字I/O脚
	EBI_AD3	I/O	MFP2	EBI地址/数据总线位3.
	SPIM_SS	I/O	MFP3	SPIM 从机选择脚
	QSPI0_SS	I/O	MFP4	QSPI0 从机选择脚
	SC1_PWR	O	MFP5	智能卡 1电源脚
	I^{2}S0_MCLK	O	MFP6	I2S0 主时钟输出脚
	SPI1_MISO	I/O	MFP7	SPI1 MISO(主机输入，从机输出)管脚
	UART2_nRTS	O	MFP8	UART2 请求发送输出脚
	I^{2}C0_SMBAL	O	MFP9	I^{2}C0 SMBus SMBALTER 管脚
	CAN1_TXD	O	MFP10	CAN1总线发送输出
	UART3_TXD	O	MFP11	UART3 数据发送输出脚.
	EPWM1_CH2	I/O	MFP12	EPWM1 通道2 输出/捕获输入
PC.4	PC.4	I/O	MFP0	通用数字I/O脚
	EBI_AD4	I/O	MFP2	EBI地址/数据总线位4.
	SPIM_D3	I/O	MFP3	SPIM QuadI/O模式下数据3管脚
	QSPI0_MOSI1	I/O	MFP4	QSPI0 MOSI1 (主机输出，从机输入) 管脚.
	SC1_nCD	I	MFP5	智能卡 1 卡检测管脚.
	I^{2}S0_BCLK	O	MFP6	I2S0位时钟输出管脚
	SPI1_I2SMCLK	I/O	MFP7	SPI1 I^{2}S 主机时钟输出脚
	UART2_RXD	I	MFP8	UART2 数据接收输入脚

	管脚名称	类型	MFP	描述
	I²C1_SDA	I/O	MFP9	I²C1 数据输入/输出脚
	CAN0_RXD	I	MFP10	CAN0 总线接收输入
	UART4_RXD	I	MFP11	UART4 数据接收输入脚
	EPWM1_CH1	I/O	MFP12	EPWM1 通道1 输出/捕获输入
PC.5	PC.5	I/O	MFP0	通用数字I/O脚
	EBI_AD5	I/O	MFP2	EBI地址/数据总线位5.
	SPIM_D2	I/O	MFP3	SPIM QuadI/O模式下数据2管脚
	QSPI0_MISO1	I/O	MFP4	QSPI0 MISO1（主机输入，从机输出）脚.
	UART2_TXD	O	MFP8	UART2 数据发送输出脚.
	I²C1_SCL	I/O	MFP9	I²C1 时钟脚
	CAN0_TXD	O	MFP10	CAN0总线发送输出
	UART4_TXD	O	MFP11	UART4 数据发送输出脚.
	EPWM1_CH0	I/O	MFP12	EPWM1 通道0 输出/捕获输入
PC.6	PC.6	I/O	MFP0	通用数字I/O脚
	EBI_AD8	I/O	MFP2	EBI地址/数据总线位8.
	EMAC_RMII_RXD1	I	MFP3	EMAC RMII接收数据总线位 1.
	SPI1_MOSI	I/O	MFP4	SPI1 MOSI (主机输出，从机输入) 管脚
	UART4_RXD	I	MFP5	UART4 数据接收输入脚
	SC2_RST	O	MFP6	智能卡 2 复位脚
	UART0_nRTS	O	MFP7	UART0 请求发送输出脚
	I²C1_SMBSUS	O	MFP8	I²C1 SMBus SMBSUS 管脚 (PMBus 控制管脚)
	EPWM1_CH3	I/O	MFP11	EPWM1 通道3输出/捕获输入脚
	BPWM1_CH1	I/O	MFP12	BPWM1 通道1 输出/捕获输入
	TM1	I/O	MFP14	Timer1 事件计数输入/反转输出脚
	INT2	I	MFP15	外部中断2输入脚
PC.7	PC.7	I/O	MFP0	通用数字I/O脚
	EBI_AD9	I/O	MFP2	EBI地址/数据总线位9.
	EMAC_RMII_RXD0	I	MFP3	EMAC RMII接收数据总线位 0.
	SPI1_MISO	I/O	MFP4	SPI1 MISO(主机输入，从机输出)管脚
	UART4_TXD	O	MFP5	UART4 数据发送输出脚.
	SC2_PWR	O	MFP6	智能卡 2电源脚

	管脚名称	类型	MFP	描述
	UART0_nCTS	I	MFP7	UART0 禁止发送输入管脚
	I²C1_SMBAL	O	MFP8	I²C1 SMBus SMBALTER 管脚
	EPWM1_CH2	I/O	MFP11	EPWM1 通道2 输出/捕获输入
	BPWM1_CH0	I/O	MFP12	BPWM1 通道0 输出/捕获输入
	TM0	I/O	MFP14	Timer0 事件计数输入/反转输出管脚
	INT3	I	MFP15	外部中断3输入脚
PC.8	PC.8	I/O	MFP0	通用数字I/O脚
	EBI_ADR16	O	MFP2	EBI 地址总线位 16.
	EMAC_RMII_REFCLK	I	MFP3	EMAC RMII 参考时钟输入脚
	I²C0_SDA	I/O	MFP4	I2C0 数据输入/输出管脚
	UART4_nCTS	I	MFP5	UART4 禁止发送输入管脚
	UART1_RXD	I	MFP8	UART1数据接收输入管脚
	EPWM1_CH1	I/O	MFP11	EPWM1 通道1 输出/捕获输入
	BPWM1_CH4	I/O	MFP12	BPWM1 通道4输出/捕获输入脚
PC.9	PC.9	I/O	MFP0	通用数字I/O脚
	EBI_ADR7	O	MFP2	EBI 地址总线位 7.
	SPI3_SS	I/O	MFP6	SPI3 从机选择脚
	UART3_RXD	I	MFP7	UART3 数据接收输入脚
	CAN1_RXD	I	MFP9	CAN1 总线接收输入
	EPWM1_CH3	I/O	MFP12	EPWM1 通道3输出/捕获输入脚
PC.10	PC.10	I/O	MFP0	通用数字I/O脚
	EBI_ADR6	O	MFP2	EBI 地址总线位 6.
	SPI3_CLK	I/O	MFP6	SPI3 串行时钟脚
	UART3_TXD	O	MFP7	UART3 数据发送输出脚.
	CAN1_TXD	O	MFP9	CAN1总线发送输出
	ECAP1_IC0	I	MFP11	增强型捕获计数器 1 输入脚0
	EPWM1_CH2	I/O	MFP12	EPWM1 通道2 输出/捕获输入
PC.11	PC.11	I/O	MFP0	通用数字I/O脚
	EBI_ADR5	O	MFP2	EBI 地址总线位 5.
	UART0_RXD	I	MFP3	UART0 数据接收输入脚
	I²C0_SDA	I/O	MFP4	I2C0 数据输入/输出管脚

	管脚名称	类型	MFP	描述
	SPI3_MOSI	I/O	MFP6	SPI3 MOSI (主机输出，从机输入) 管脚.
	ECAP1_IC1	I	MFP11	增强型捕获计数器 1 输入 1 管脚.
	EPWM1_CH1	I/O	MFP12	EPWM1 通道1 输出/捕获输入
	ACMP1_O	O	MFP14	模拟比较器1输出脚
PC.12	PC.12	I/O	MFP0	通用数字I/O脚
	EBI_ADR4	O	MFP2	EBI 地址总线位 4.
	UART0_TXD	O	MFP3	UART0 数据发送输出脚.
	I^2C0_SCL	I/O	MFP4	I2C0 时钟管脚
	SPI3_MISO	I/O	MFP6	SPI3 MISO（主机输入，从机输出）脚.
	SC0_nCD	I	MFP9	智能卡 0 卡检测管脚.
	ECAP1_IC2	I	MFP11	增强型捕获计数器 1 输入 2 管脚.
	EPWM1_CH0	I/O	MFP12	EPWM1 通道0 输出/捕获输入
	ACMP0_O	O	MFP14	模拟比较器 0输出脚
PC.13	PC.13	I/O	MFP0	通用数字I/O脚
	EBI_ADR10	O	MFP2	EBI 地址总线位 10.
	SC2_nCD	I	MFP3	智能卡 2 卡检测管脚.
	SPI2_I2SMCLK	I/O	MFP4	SPI2 I^2S 主机时钟输出脚
	CAN1_TXD	O	MFP5	CAN1总线发送输出
	USCI0_CTL0	I/O	MFP6	USCI0 控制0 管脚.
	UART2_TXD	O	MFP7	UART2 数据发送输出脚.
	BPWM0_CH4	I/O	MFP9	BPWM0 通道4输出/捕获输入脚
	CLKO	O	MFP13	时钟输出
	EADC0_ST	I	MFP14	EADC0 外部触发输入
PC.14	PC.14	I/O	MFP0	通用数字I/O脚
	EBI_AD11	I/O	MFP2	EBI地址/数据总线位11.
	SC1_nCD	I	MFP3	智能卡 1 卡检测管脚.
	SPI0_I2SMCLK	I/O	MFP4	SPI0 I^2S 主机时钟输出脚
	USCI0_CTL0	I/O	MFP5	USCI0 控制0 管脚.
	QSPI0_CLK	I/O	MFP6	QSPI0 串行时钟脚
	EPWM0_SYNC_IN	I	MFP11	EPWM0 计数器同步触发输入脚
	ETM_TRACE_CLK	I	MFP12	ETM接收追踪时钟输入管脚

	管脚名称	类型	MFP	描述
	TM1	I/O	MFP13	Timer1 事件计数输入/反转输出脚
	USB_VBUS_ST	I	MFP14	USB 外部VBUS 电源状态管脚.
	HSUSB_VBUS_ST	I	MFP15	HSUSB 外部VBUS 电源状态管脚.
PD.0	PD.0	I/O	MFP0	通用数字I/O脚
	EBI_AD13	I/O	MFP2	EBI地址/数据总线位13.
	USCI0_CLK	I/O	MFP3	USCI0 时钟脚
	SPI0_MOSI	I/O	MFP4	SPI0 MOSI (主机输出，从机输入) 管脚.
	UART3_RXD	I	MFP5	UART3 数据接收输入脚
	I^2C2_SDA	I/O	MFP6	I^2C2 数据输入/输出脚
	SC2_CLK	O	MFP7	智能卡 2 时钟脚
	TM2	I/O	MFP14	Timer2 事件计数输入/反转输出脚
PD.1	PD.1	I/O	MFP0	通用数字I/O脚
	EBI_AD12	I/O	MFP2	EBI地址/数据总线位12.
	USCI0_DAT0	I/O	MFP3	USCI0 数据0脚
	SPI0_MISO	I/O	MFP4	SPI0 MISO（主机输入，从机输出）脚.
	UART3_TXD	O	MFP5	UART3 数据发送输出脚.
	I^2C2_SCL	I/O	MFP6	I^2C2 时钟脚
	SC2_DAT	I/O	MFP7	智能卡 2 数据管脚.
PD.2	PD.2	I/O	MFP0	通用数字I/O脚
	EBI_AD11	I/O	MFP2	EBI地址/数据总线位11.
	USCI0_DAT1	I/O	MFP3	USCI0 数据1 管脚.
	SPI0_CLK	I/O	MFP4	SPI0 串行时钟脚
	UART3_nCTS	I	MFP5	UART3 禁止发送输入管脚
	SC2_RST	O	MFP7	智能卡 2 复位脚
	UART0_RXD	I	MFP9	UART0 数据接收输入脚
PD.3	PD.3	I/O	MFP0	通用数字I/O脚
	EBI_AD10	I/O	MFP2	EBI地址/数据总线位10.
	USCI0_CTL1	I/O	MFP3	USCI0 控制1 管脚.
	SPI0_SS	I/O	MFP4	SPI0 从机选择脚
	UART3_nRTS	O	MFP5	UART3 请求发送输出脚
	USCI1_CTL0	I/O	MFP6	USCI1 控制0管脚

	管脚名称	类型	MFP	描述
	SC2_PWR	O	MFP7	智能卡 2电源脚
	SC1_nCD	I	MFP8	智能卡 1 卡检测管脚.
	UART0_TXD	O	MFP9	UART0 数据发送输出脚.
PD.4	PD.4	I/O	MFP0	通用数字I/O脚
	USCI0_CTL0	I/O	MFP3	USCI0 控制0 管脚.
	I²C1_SDA	I/O	MFP4	I²C1 数据输入/输出脚
	SPI1_SS	I/O	MFP5	SPI1 从机选择脚
	USCI1_CTL1	I/O	MFP6	USCI1 控制1管脚
	SC1_CLK	O	MFP8	智能卡 1 时钟脚
	USB_VBUS_ST	I	MFP14	USB 外部VBUS 电源状态管脚.
PD.5	PD.5	I/O	MFP0	通用数字I/O脚
	I²C1_SCL	I/O	MFP4	I²C1 时钟脚
	SPI1_CLK	I/O	MFP5	SPI1 串行时钟管脚
	USCI1_DAT0	I/O	MFP6	USCI1 数据0脚
	SC1_DAT	I/O	MFP8	智能卡 1 数据管脚.
PD.6	PD.6	I/O	MFP0	通用数字I/O脚
	UART1_RXD	I	MFP3	UART1数据接收输入管脚
	I²C0_SDA	I/O	MFP4	I2C0 数据输入/输出管脚
	SPI1_MOSI	I/O	MFP5	SPI1 MOSI (主机输出，从机输入) 管脚
	USCI1_DAT1	I/O	MFP6	USCI1数据1管脚.
	SC1_RST	O	MFP8	智能卡 1 复位脚
PD.7	PD.7	I/O	MFP0	通用数字I/O脚
	UART1_TXD	O	MFP3	UART1 数据发送输出脚
	I²C0_SCL	I/O	MFP4	I2C0 时钟管脚
	SPI1_MISO	I/O	MFP5	SPI1 MISO(主机输入，从机输出)管脚
	USCI1_CLK	I/O	MFP6	USCI1 时钟脚
	SC1_PWR	O	MFP8	智能卡 1电源脚
PD.8	PD.8	I/O	MFP0	通用数字I/O脚
	EBI_AD6	I/O	MFP2	EBI地址/数据总线位6.
	I²C2_SDA	I/O	MFP3	I²C2 数据输入/输出脚
	UART2_nRTS	O	MFP4	UART2 请求发送输出脚

	管脚名称	类型	MFP	描述
PD.9	PD.9	I/O	MFP0	通用数字I/O脚
	EBI_AD7	I/O	MFP2	EBI地址/数据总线位7.
	I²C2_SCL	I/O	MFP3	I²C2 时钟脚
	UART2_nCTS	I	MFP4	UART2 禁止发送输入管脚
PD.10	PD.10	I/O	MFP0	通用数字I/O脚
	OPA2_P	A	MFP1	运放 2 正极输入脚
	EBI_nCS2	O	MFP2	EBI片选2输出脚
	UART1_RXD	I	MFP3	UART1数据接收输入管脚
	CAN0_RXD	I	MFP4	CAN0 总线接收输入
	QEI0_B	I	MFP10	正交编码器0 相位B 输入
	INT7	I	MFP15	外部中断7 输入脚
PD.11	PD.11	I/O	MFP0	通用数字I/O脚
	OPA2_N	A	MFP1	运放 2 负极输入脚
	EBI_nCS1	O	MFP2	EBI片选1输出管脚
	UART1_TXD	O	MFP3	UART1 数据发送输出脚
	CAN0_TXD	O	MFP4	CAN0总线发送输出
	QEI0_A	I	MFP10	正交编码器0 相位A输入
	INT6	I	MFP15	外部中断6 输入脚
PD.12	PD.12	I/O	MFP0	通用数字I/O脚
	OPA2_O	A	MFP1	运放 2输出脚
	EBI_nCS0	O	MFP2	EBI片选0输出脚
	CAN1_RXD	I	MFP5	CAN1 总线接收输入
	UART2_RXD	I	MFP7	UART2 数据接收输入脚
	BPWM0_CH5	I/O	MFP9	BPWM0 通道5 输出/捕获输入
	QEI0_INDEX	I	MFP10	正交编码器0 索引输入
	CLKO	O	MFP13	时钟输出
	EADC0_ST	I	MFP14	EADC0 外部触发输入
	INT5	I	MFP15	外部中断5 输入脚
PD.13	PD.13	I/O	MFP0	通用数字I/O脚
	EBI_AD10	I/O	MFP2	EBI地址/数据总线位10.
	SD0_nCD	I	MFP3	SD/SDIO0 卡检测输入管脚

	管脚名称	类型	MFP	描述
	SPI0_I2SMCLK	I/O	MFP4	SPI0 I²S 主机时钟输出脚
	SPI1_I2SMCLK	I/O	MFP5	SPI1 I²S 主机时钟输出脚
	SC2_nCD	I	MFP7	智能卡 2 卡检测管脚.
PD.14	PD.14	I/O	MFP0	通用数字I/O脚
	EBI_nCS0	O	MFP2	EBI片选0输出脚
	SPI3_I2SMCLK	I/O	MFP3	SPI3 I²S 主机时钟输出脚
	SC1_nCD	I	MFP4	智能卡 1 卡检测管脚.
	EPWM0_CH4	I/O	MFP11	EPWM0 通道4输出/捕获输入脚
PE.0	PE.0	I/O	MFP0	通用数字I/O脚
	EBI_AD11	I/O	MFP2	EBI地址/数据总线位11.
	QSPI0_MOSI0	I/O	MFP3	QSPI0 MOSI0 (主机输出，从机输入) 管脚.
	SC2_CLK	O	MFP4	智能卡 2 时钟脚
	I²S0_MCLK	O	MFP5	I2S0 主时钟输出脚
	SPI1_MOSI	I/O	MFP6	SPI1 MOSI (主机输出，从机输入) 管脚
	UART3_RXD	I	MFP7	UART3 数据接收输入脚
	I²C1_SDA	I/O	MFP8	I²C1 数据输入/输出脚
	UART4_nRTS	O	MFP9	UART4 请求发送输出脚
PE.1	PE.1	I/O	MFP0	通用数字I/O脚
	EBI_AD10	I/O	MFP2	EBI地址/数据总线位10.
	QSPI0_MISO0	I/O	MFP3	QSPI0 MISO0（主机输入，从机输出）脚.
	SC2_DAT	I/O	MFP4	智能卡 2 数据管脚.
	I²S0_BCLK	O	MFP5	I2S0位时钟输出管脚
	SPI1_MISO	I/O	MFP6	SPI1 MISO(主机输入，从机输出)管脚
	UART3_TXD	O	MFP7	UART3 数据发送输出脚.
	I²C1_SCL	I/O	MFP8	I²C1 时钟脚
	UART4_nCTS	I	MFP9	UART4 禁止发送输入管脚
PE.2	PE.2	I/O	MFP0	通用数字I/O脚
	EBI_ALE	O	MFP2	EBI地址锁存使能输出脚
	SD0_DAT0	I/O	MFP3	SD/SDIO0 数据线位0
	SPIM_MOSI	I/O	MFP4	SPIM MOSI (主机输出，从机输入) 管脚.
	SPI3_MOSI	I/O	MFP5	SPI3 MOSI (主机输出，从机输入) 管脚.

	管脚名称	类型	MFP	描述
	SC0_CLK	O	MFP6	智能卡0时钟管脚
	USCI0_CLK	I/O	MFP7	USCI0 时钟脚
	QEI0_B	I	MFP11	正交编码器0 相位B 输入
	EPWM0_CH5	I/O	MFP12	EPWM0 通道5 输出/捕获输入
	BPWM0_CH0	I/O	MFP13	BPWM0 通道0 输出/捕获输入
PE.3	PE.3	I/O	MFP0	通用数字I/O脚
	EBI_MCLK	O	MFP2	EBI 外部时钟输出脚.
	SD0_DAT1	I/O	MFP3	SD/SDIO0 数据线位1
	SPIM_MISO	I/O	MFP4	SPIM MISO（主机输入，从机输出）脚.
	SPI3_MISO	I/O	MFP5	SPI3 MISO（主机输入，从机输出）脚.
	SC0_DAT	I/O	MFP6	智能卡0数据输入脚
	USCI0_DAT0	I/O	MFP7	USCI0 数据0脚
	QEI0_A	I	MFP11	正交编码器0 相位A输入
	EPWM0_CH4	I/O	MFP12	EPWM0 通道4输出/捕获输入脚
	BPWM0_CH1	I/O	MFP13	BPWM0 通道1 输出/捕获输入
PE.4	PE.4	I/O	MFP0	通用数字I/O脚
	EBI_nWR	O	MFP2	EBI 写使能输出脚
	SD0_DAT2	I/O	MFP3	SD/SDIO0 数据线位2
	SPIM_CLK	I/O	MFP4	SPIM 串行时钟脚
	SPI3_CLK	I/O	MFP5	SPI3 串行时钟脚
	SC0_RST	O	MFP6	智能卡0复位脚
	USCI0_DAT1	I/O	MFP7	USCI0 数据1 管脚.
	QEI0_INDEX	I	MFP11	正交编码器0 索引输入
	EPWM0_CH3	I/O	MFP12	EPWM0 通道3输出/捕获输入脚
	BPWM0_CH2	I/O	MFP13	BPWM0 通道2 输出/捕获输入
PE.5	PE.5	I/O	MFP0	通用数字I/O脚
	EBI_nRD	O	MFP2	EBI 读使能输出脚
	SD0_DAT3	I/O	MFP3	SD/SDIO0数据线位3
	SPIM_SS	I/O	MFP4	SPIM 从机选择脚
	SPI3_SS	I/O	MFP5	SPI3 从机选择脚
	SC0_PWR	O	MFP6	智能卡0电源脚

	管脚名称	类型	MFP	描述
	USCI0_CTL1	I/O	MFP7	USCI0 控制1 管脚.
	QEI1_B	I	MFP11	正交编码器1 相位B 输入
	EPWM0_CH2	I/O	MFP12	EPWM0 通道2输出/捕获输入脚
	BPWM0_CH3	I/O	MFP13	BPWM0 通道3输出/捕获输入脚
PE.6	PE.6	I/O	MFP0	通用数字I/O脚
	SD0_CLK	O	MFP3	SD/SDIO0 时钟输出脚
	SPIM_D3	I/O	MFP4	SPIM QuadI/O模式下数据3管脚
	SPI3_I2SMCLK	I/O	MFP5	SPI3 I^2S 主机时钟输出脚
	SC0_nCD	I	MFP6	智能卡 0 卡检测管脚.
	USCI0_CTL0	I/O	MFP7	USCI0 控制0 管脚.
	UART5_RXD	I	MFP8	UART5 数据接收输入管脚
	CAN1_RXD	I	MFP9	CAN1 总线接收输入
	QEI1_A	I	MFP11	正交编码器1 相位A输入
	EPWM0_CH1	I/O	MFP12	EPWM0 通道1输出/捕获输入脚
	BPWM0_CH4	I/O	MFP13	BPWM0 通道4输出/捕获输入脚
PE.7	PE.7	I/O	MFP0	通用数字I/O脚
	SD0_CMD	I/O	MFP3	SD/SDIO0 命令/响应脚
	SPIM_D2	I/O	MFP4	SPIM QuadI/O模式下数据2管脚
	UART5_TXD	O	MFP8	UART5数据发送输出脚
	CAN1_TXD	O	MFP9	CAN1总线发送输出
	QEI1_INDEX	I	MFP11	正交编码器1 索引输入
	EPWM0_CH0	I/O	MFP12	EPWM0 通道0输出/捕获输入
	BPWM0_CH5	I/O	MFP13	BPWM0 通道5 输出/捕获输入
PE.8	PE.8	I/O	MFP0	通用数字I/O脚
	EBI_ADR10	O	MFP2	EBI 地址总线位 10.
	EMAC_RMII_MDC	O	MFP3	EMAC RMII PHY管理时钟输出脚.
	I^2S0_BCLK	O	MFP4	I2S0位时钟输出管脚
	SPI2_CLK	I/O	MFP5	SPI2 串行时钟脚
	USCI1_CTL1	I/O	MFP6	USCI1 控制1管脚
	UART2_TXD	O	MFP7	UART2 数据发送输出脚.
	EPWM0_CH0	I/O	MFP10	EPWM0 通道0输出/捕获输入

	管脚名称	类型	MFP	描述
	EPWM0_BRAKE0	I	MFP11	EPWM0 刹车0 输入脚
	ECAP0_IC0	I	MFP12	增强型捕获计数器 0 输入脚0
	TRACE_CLK	O	MFP14	ETM跟踪时钟输出脚
PE.9	PE.9	I/O	MFP0	通用数字I/O脚
	EBI_ADR11	O	MFP2	EBI 地址总线位 11.
	EMAC_RMII_MDIO	I/O	MFP3	EMAC RMII PHY管理数据管脚.
	I^2S0_MCLK	O	MFP4	I2S0 主时钟输出脚
	SPI2_MISO	I/O	MFP5	SPI2 MISO（主机输入，从机输出）脚.
	USCI1_CTL0	I/O	MFP6	USCI1 控制0管脚
	UART2_RXD	I	MFP7	UART2 数据接收输入脚
	EPWM0_CH1	I/O	MFP10	EPWM0 通道1输出/捕获输入脚
	EPWM0_BRAKE1	I	MFP11	EPWM0 刹车1 输入脚
	ECAP0_IC1	I	MFP12	增强型捕获计数器 0 输入 1 管脚.
	TRACE_DATA0	O	MFP14	ETM跟踪数据0 输出管脚
PE.10	PE.10	I/O	MFP0	通用数字I/O脚
	EBI_ADR12	O	MFP2	EBI 地址总线位 12.
	EMAC_RMII_TXD0	O	MFP3	EMAC RMII 发送数据总线位0.
	I^2S0_DI	I	MFP4	I2S0 数据输入脚
	SPI2_MOSI	I/O	MFP5	SPI2 MOSI (主机输出，从机输入) 管脚.
	USCI1_DAT0	I/O	MFP6	USCI1 数据0脚
	UART3_TXD	O	MFP7	UART3 数据发送输出脚.
	EPWM0_CH2	I/O	MFP10	EPWM0 通道2输出/捕获输入脚
	EPWM1_BRAKE0	I	MFP11	EPWM1 刹车0 输入脚
	ECAP0_IC2	I	MFP12	增强型捕获计数器 0 输入 2 管脚.
	TRACE_DATA1	O	MFP14	ETM跟踪数据1 输出管脚
PE.11	PE.11	I/O	MFP0	通用数字I/O脚
	EBI_ADR13	O	MFP2	EBI 地址总线位 13.
	EMAC_RMII_TXD1	O	MFP3	EMAC RMII 发送数据总线位1.
	I^2S0_DO	O	MFP4	I^2S0 数据输出脚
	SPI2_SS	I/O	MFP5	SPI2 从机选择脚
	USCI1_DAT1	I/O	MFP6	USCI1数据1管脚.

	管脚名称	类型	MFP	描述
	UART3_RXD	I	MFP7	UART3 数据接收输入脚
	UART1_nCTS	I	MFP8	UART1 禁止发送输入管脚
	EPWM0_CH3	I/O	MFP10	EPWM0 通道3输出/捕获输入脚
	EPWM1_BRAKE1	I	MFP11	EPWM1 刹车1输入管脚
	ECAP1_IC2	I	MFP13	增强型捕获计数器 1 输入 2 管脚.
	TRACE_DATA2	O	MFP14	ETM跟踪数据2 输出管脚
PE.12	PE.12	I/O	MFP0	通用数字I/O脚
	EBI_ADR14	O	MFP2	EBI 地址总线位 14.
	EMAC_RMII_TXEN	O	MFP3	EMAC RMII 发送使能输出脚
	I^{2}S0_LRCK	O	MFP4	I^{2}S0左右通道时钟输出脚.
	SPI2_I2SMCLK	I/O	MFP5	SPI2 I^{2}S 主机时钟输出脚
	USCI1_CLK	I/O	MFP6	USCI1 时钟脚
	UART1_nRTS	O	MFP8	UART1 请求发送输出脚
	EPWM0_CH4	I/O	MFP10	EPWM0 通道4输出/捕获输入脚
	ECAP1_IC1	I	MFP13	增强型捕获计数器 1 输入 1 管脚.
	TRACE_DATA3	O	MFP14	ETM跟踪数据3 输出管脚
PE.13	PE.13	I/O	MFP0	通用数字I/O脚
	EBI_ADR15	O	MFP2	EBI 地址总线位 15.
	EMAC_PPS	O	MFP3	EMAC 每秒脉冲输出脚
	I^{2}C0_SCL	I/O	MFP4	I2C0 时钟管脚
	UART4_nRTS	O	MFP5	UART4 请求发送输出脚
	UART1_TXD	O	MFP8	UART1 数据发送输出脚
	EPWM0_CH5	I/O	MFP10	EPWM0 通道5 输出/捕获输入
	EPWM1_CH0	I/O	MFP11	EPWM1 通道0 输出/捕获输入
	BPWM1_CH5	I/O	MFP12	EPWM1 通道5输出/捕获输入
	ECAP1_IC0	I	MFP13	增强型捕获计数器 1 输入脚0
PE.14	PE.14	I/O	MFP0	通用数字I/O脚
	EBI_AD8	I/O	MFP2	EBI地址/数据总线位8.
	UART2_TXD	O	MFP3	UART2 数据发送输出脚.
	CAN0_TXD	O	MFP4	CAN0总线发送输出
	SD1_nCD	I	MFP5	SD/SDIO1 卡检测输入管脚

	管脚名称	类型	MFP	描述
PE.15	PE.15	I/O	MFP0	通用数字I/O脚
	EBI_AD9	I/O	MFP2	EBI地址/数据总线位9.
	UART2_RXD	I	MFP3	UART2 数据接收输入脚
	CAN0_RXD	I	MFP4	CAN0 总线接收输入
PF.0	PF.0	I/O	MFP0	通用数字I/O脚
	UART1_TXD	O	MFP2	UART1 数据发送输出脚
	I^2C1_SCL	I/O	MFP3	I^2C1 时钟脚
	BPWM1_CH0	I/O	MFP12	BPWM1 通道0 输出/捕获输入
	ICE_DAT	O	MFP14	串行调试总线数据管脚.
PF.1	PF.1	I/O	MFP0	通用数字I/O脚
	UART1_RXD	I	MFP2	UART1数据接收输入管脚
	I^2C1_SDA	I/O	MFP3	I^2C1 数据输入/输出脚
	BPWM1_CH1	I/O	MFP12	BPWM1 通道1 输出/捕获输入
	ICE_CLK	I	MFP14	串行调试总线时钟脚
PF.2	PF.2	I/O	MFP0	通用数字I/O脚
	EBI_nCS1	O	MFP2	EBI片选1输出管脚
	UART0_RXD	I	MFP3	UART0 数据接收输入脚
	I^2C0_SDA	I/O	MFP4	I2C0 数据输入/输出管脚
	QSPI0_CLK	I/O	MFP5	QSPI0 串行时钟脚
	XT1_OUT	O	MFP10	外部 4~24 MHz (高速) 晶振输出脚
	BPWM1_CH1	I/O	MFP11	BPWM1 通道1 输出/捕获输入
PF.3	PF.3	I/O	MFP0	通用数字I/O脚
	EBI_nCS0	O	MFP2	EBI片选0输出脚
	UART0_TXD	O	MFP3	UART0 数据发送输出脚.
	I^2C0_SCL	I/O	MFP4	I2C0 时钟管脚
	XT1_IN	I	MFP10	外部 4~24 MHz (高速) 晶振输入脚
	BPWM1_CH0	I/O	MFP11	BPWM1 通道0 输出/捕获输入
PF.4	PF.4	I/O	MFP0	通用数字I/O脚
	UART2_TXD	O	MFP2	UART2 数据发送输出脚.
	UART2_nRTS	O	MFP4	UART2 请求发送输出脚
	BPWM0_CH5	I/O	MFP8	BPWM0 通道5 输出/捕获输入

	管脚名称	类型	MFP	描述
	X32_OUT	O	MFP10	外部 32.768 kHz 晶振输出脚
PF.5	PF.5	I/O	MFP0	通用数字I/O脚
	UART2_RXD	I	MFP2	UART2 数据接收输入脚
	UART2_nCTS	I	MFP4	UART2 禁止发送输入管脚
	BPWM0_CH4	I/O	MFP8	BPWM0 通道4输出/捕获输入脚
	EPWM0_SYNC_OUT	O	MFP9	EPWM0计数器同步触发输出脚
	X32_IN	I	MFP10	外部 32.768 kHz 晶振输入脚
	EADC0_ST	I	MFP11	EADC0 外部触发输入
PF.6	PF.6	I/O	MFP0	通用数字I/O脚
	EBI_ADR19	O	MFP2	EBI 地址总线位 19.
	SC0_CLK	O	MFP3	智能卡0时钟管脚
	I^2S0_LRCK	O	MFP4	I^2S0左右通道时钟输出脚.
	SPI0_MOSI	I/O	MFP5	SPI0 MOSI (主机输出，从机输入) 管脚.
	UART4_RXD	I	MFP6	UART4 数据接收输入脚
	EBI_nCS0	O	MFP7	EBI片选0输出脚
	TAMPER0	I/O	MFP10	TAMPER循环检测管脚 0.
PF.7	PF.7	I/O	MFP0	通用数字I/O脚
	EBI_ADR18	O	MFP2	EBI 地址总线位 18.
	SC0_DAT	I/O	MFP3	智能卡0数据输入脚
	I^2S0_DO	O	MFP4	I^2S0 数据输出脚
	SPI0_MISO	I/O	MFP5	SPI0 MISO（主机输入，从机输出）脚.
	UART4_TXD	O	MFP6	UART4 数据发送输出脚.
	TAMPER1	I/O	MFP10	TAMPER循环检测管脚 1.
PF.8	PF.8	I/O	MFP0	通用数字I/O脚
	EBI_ADR17	O	MFP2	EBI 地址总线位 17.
	SC0_RST	O	MFP3	智能卡0复位脚
	I^2S0_DI	I	MFP4	I2S0 数据输入脚
	SPI0_CLK	I/O	MFP5	SPI0 串行时钟脚
	TAMPER2	I/O	MFP10	TAMPER循环检测管脚 2.
PF.9	PF.9	I/O	MFP0	通用数字I/O脚
	EBI_ADR16	O	MFP2	EBI 地址总线位 16.

	管脚名称	类型	MFP	描述
	SC0_PWR	O	MFP3	智能卡0电源脚
	I²S0_MCLK	O	MFP4	I2S0 主时钟输出脚
	SPI0_SS	I/O	MFP5	SPI0 从机选择脚
	TAMPER3	I/O	MFP10	TAMPER循环检测管脚 3.
PF.10	PF.10	I/O	MFP0	通用数字I/O脚
	EBI_ADR15	O	MFP2	EBI 地址总线位 15.
	SC0_nCD	I	MFP3	智能卡 0 卡检测管脚.
	I²S0_BCLK	O	MFP4	I2S0位时钟输出管脚
	SPI0_I2SMCLK	I/O	MFP5	SPI0 I²S 主机时钟输出脚
	TAMPER4	I/O	MFP10	TAMPER循环检测管脚 4.
PF.11	PF.11	I/O	MFP0	通用数字I/O脚
	EBI_ADR14	O	MFP2	EBI 地址总线位 14.
	SPI2_MOSI	I/O	MFP3	SPI2 MOSI (主机输出，从机输入) 管脚.
	TAMPER5	I/O	MFP10	TAMPER循环检测管脚 5.
	TM3	I/O	MFP13	Timer3 事件计数输入/反转输出脚
PG.0	PG.0	I/O	MFP0	通用数字I/O脚
	EBI_ADR8	O	MFP2	EBI 地址总线位 8.
	I²C0_SCL	I/O	MFP4	I2C0 时钟管脚
	I²C1_SMBAL	O	MFP5	I²C1 SMBus SMBALTER 管脚
	UART2_RXD	I	MFP6	UART2 数据接收输入脚
	CAN1_TXD	O	MFP7	CAN1总线发送输出
	UART1_TXD	O	MFP8	UART1 数据发送输出脚
PG.1	PG.1	I/O	MFP0	通用数字I/O脚
	EBI_ADR9	O	MFP2	EBI 地址总线位 9.
	SPI2_I2SMCLK	I/O	MFP3	SPI2 I²S 主机时钟输出脚
	I²C0_SDA	I/O	MFP4	I2C0 数据输入/输出管脚
	I²C1_SMBSUS	O	MFP5	I²C1 SMBus SMBSUS 管脚 (PMBus 控制管脚)
	UART2_TXD	O	MFP6	UART2 数据发送输出脚.
	CAN1_RXD	I	MFP7	CAN1 总线接收输入
	UART1_RXD	I	MFP8	UART1数据接收输入管脚
PG.2	PG.2	I/O	MFP0	通用数字I/O脚

	管脚名称	类型	MFP	描述
	EBI_ADR11	O	MFP2	EBI 地址总线位 11.
	SPI2_SS	I/O	MFP3	SPI2 从机选择脚
	I²C0_SMBAL	O	MFP4	I²C0 SMBus SMBALTER 管脚
	I²C1_SCL	I/O	MFP5	I²C1 时钟脚
	TM0	I/O	MFP13	Timer0 事件计数输入/反转输出管脚
PG.3	PG.3	I/O	MFP0	通用数字I/O脚
	EBI_ADR12	O	MFP2	EBI 地址总线位 12.
	SPI2_CLK	I/O	MFP3	SPI2 串行时钟脚
	I²C0_SMBSUS	O	MFP4	I²C0 SMBus SMBSUS 管脚 (PMBus 控制管脚)
	I²C1_SDA	I/O	MFP5	I²C1 数据输入/输出脚
	TM1	I/O	MFP13	Timer1 事件计数输入/反转输出脚
PG.4	PG.4	I/O	MFP0	通用数字I/O脚
	EBI_ADR13	O	MFP2	EBI 地址总线位 13.
	SPI2_MISO	I/O	MFP3	SPI2 MISO（主机输入，从机输出）脚.
	TM2	I/O	MFP13	Timer2 事件计数输入/反转输出脚
PG.5	PG.5	I/O	MFP0	通用数字I/O脚
	EBI_nCS1	O	MFP2	EBI片选1输出管脚
	SPI3_SS	I/O	MFP3	SPI3 从机选择脚
	SC1_PWR	O	MFP4	智能卡 1电源脚
	EPWM0_CH3	I/O	MFP11	EPWM0 通道3输出/捕获输入脚
PG.6	PG.6	I/O	MFP0	通用数字I/O脚
	EBI_nCS2	O	MFP2	EBI片选2输出脚
	SPI3_CLK	I/O	MFP3	SPI3 串行时钟脚
	SC1_RST	O	MFP4	智能卡 1 复位脚
	EPWM0_CH2	I/O	MFP11	EPWM0 通道2输出/捕获输入脚
PG.7	PG.7	I/O	MFP0	通用数字I/O脚
	EBI_nWRL	O	MFP2	EBI低字节写使能输出脚
	SPI3_MISO	I/O	MFP3	SPI3 MISO（主机输入，从机输出）脚.
	SC1_DAT	I/O	MFP4	智能卡 1 数据管脚.
	EPWM0_CH1	I/O	MFP11	EPWM0 通道1输出/捕获输入脚
PG.8	PG.8	I/O	MFP0	通用数字I/O脚

	管脚名称	类型	MFP	描述
	EBI_nWRH	O	MFP2	EBI 高字节写使能输出管脚
	SPI3_MOSI	I/O	MFP3	SPI3 MOSI (主机输出，从机输入) 管脚.
	SC1_CLK	O	MFP4	智能卡 1 时钟脚
	EPWM0_CH0	I/O	MFP11	EPWM0 通道0输出/捕获输入
PG.9	PG.9	I/O	MFP0	通用数字I/O脚
	EBI_AD0	I/O	MFP2	EBI地址/数据总线位0.
	SD1_DAT3	I/O	MFP3	SD/SDIO1 数据线位3.
	SPIM_D2	I/O	MFP4	SPIM QuadI/O模式下数据2管脚
	BPWM0_CH5	I/O	MFP12	BPWM0 通道5 输出/捕获输入
PG.10	PG.10	I/O	MFP0	通用数字I/O脚
	EBI_AD1	I/O	MFP2	EBI地址/数据总线位1.
	SD1_DAT2	I/O	MFP3	SD/SDIO1 数据线位2.
	SPIM_D3	I/O	MFP4	SPIM QuadI/O模式下数据3管脚
	BPWM0_CH4	I/O	MFP12	BPWM0 通道4输出/捕获输入脚
PG.11	PG.11	I/O	MFP0	通用数字I/O脚
	EBI_AD2	I/O	MFP2	EBI地址/数据总线位2.
	SD1_DAT1	I/O	MFP3	SD/SDIO1 数据线位1.
	SPIM_SS	I/O	MFP4	SPIM 从机选择脚
	BPWM0_CH3	I/O	MFP12	BPWM0 通道3输出/捕获输入脚
PG.12	PG.12	I/O	MFP0	通用数字I/O脚
	EBI_AD3	I/O	MFP2	EBI地址/数据总线位3.
	SD1_DAT0	I/O	MFP3	SD/SDIO1 数据线位0.
	SPIM_CLK	I/O	MFP4	SPIM 串行时钟脚
	BPWM0_CH2	I/O	MFP12	BPWM0 通道2 输出/捕获输入
PG.13	PG.13	I/O	MFP0	通用数字I/O脚
	EBI_AD4	I/O	MFP2	EBI地址/数据总线位4.
	SD1_CMD	I/O	MFP3	SD/SDIO1 命令/响应脚
	SPIM_MISO	I/O	MFP4	SPIM MISO（主机输入，从机输出）脚.
	BPWM0_CH1	I/O	MFP12	BPWM0 通道1 输出/捕获输入
PG.14	PG.14	I/O	MFP0	通用数字I/O脚
	EBI_AD5	I/O	MFP2	EBI地址/数据总线位5.

	管脚名称	类型	MFP	描述
	SD1_CLK	O	MFP3	SD/SDIO1 时钟输出管脚
	SPIM_MOSI	I/O	MFP4	SPIM MOSI (主机输出，从机输入) 管脚.
	BPWM0_CH0	I/O	MFP12	BPWM0 通道0 输出/捕获输入
PG.15	PG.15	I/O	MFP0	通用数字I/O脚
	SD1_nCD	I	MFP3	SD/SDIO1 卡检测输入管脚
	CLKO	O	MFP14	时钟输出
	EADC0_ST	I	MFP15	EADC0 外部触发输入
PH.0	PH.0	I/O	MFP0	通用数字I/O脚
	EBI_ADR7	O	MFP2	EBI 地址总线位 7.
	UART5_TXD	O	MFP4	UART5数据发送输出脚
	TM0_EXT	I/O	MFP13	Timer0 事件计数输入/反转输出管脚
PH.1	PH.1	I/O	MFP0	通用数字I/O脚
	EBI_ADR6	O	MFP2	EBI 地址总线位 6.
	UART5_RXD	I	MFP4	UART5 数据接收输入管脚
	TM1_EXT	I/O	MFP13	Timer1 事件计数输入/反转输出脚
PH.2	PH.2	I/O	MFP0	通用数字I/O脚
	EBI_ADR5	O	MFP2	EBI 地址总线位 5.
	UART5_nRTS	O	MFP4	UART5 请求发送输出脚
	UART4_TXD	O	MFP5	UART4 数据发送输出脚.
	I^2C0_SCL	I/O	MFP6	I2C0 时钟管脚
	TM2_EXT	I/O	MFP13	Timer2 事件计数输入/反转输出脚
PH.3	PH.3	I/O	MFP0	通用数字I/O脚
	EBI_ADR4	O	MFP2	EBI 地址总线位 4.
	SPI1_I2SMCLK	I/O	MFP3	SPI1 I^2S 主机时钟输出脚
	UART5_nCTS	I	MFP4	UART5 禁止发送输入管脚
	UART4_RXD	I	MFP5	UART4 数据接收输入脚
	I^2C0_SDA	I/O	MFP6	I2C0 数据输入/输出管脚
	TM3_EXT	I/O	MFP13	Timer3 事件计数输入/反转输出脚
PH.4	PH.4	I/O	MFP0	通用数字I/O脚
	EBI_ADR3	O	MFP2	EBI 地址总线位3
	SPI1_MISO	I/O	MFP3	SPI1 MISO(主机输入，从机输出)管脚

	管脚名称	类型	MFP	描述
PH.5	PH.5	I/O	MFP0	通用数字I/O脚
	EBI_ADR2	O	MFP2	EBI 地址总线位2
	SPI1_MOSI	I/O	MFP3	SPI1 MOSI (主机输出，从机输入) 管脚
PH.6	PH.6	I/O	MFP0	通用数字I/O脚
	EBI_ADR1	O	MFP2	EBI 地址总线位1
	SPI1_CLK	I/O	MFP3	SPI1 串行时钟管脚
PH.7	PH.7	I/O	MFP0	通用数字I/O脚
	EBI_ADR0	O	MFP2	EBI地址总线位0
	SPI1_SS	I/O	MFP3	SPI1 从机选择脚
PH.8	PH.8	I/O	MFP0	通用数字I/O脚
	EBI_AD12	I/O	MFP2	EBI地址/数据总线位12.
	QSPI0_CLK	I/O	MFP3	QSPI0 串行时钟脚
	SC2_PWR	O	MFP4	智能卡 2电源脚
	I^2S0_DI	I	MFP5	I2S0 数据输入脚
	SPI1_CLK	I/O	MFP6	SPI1 串行时钟管脚
	UART3_nRTS	O	MFP7	UART3 请求发送输出脚
	I^2C1_SMBAL	O	MFP8	I^2C1 SMBus SMBALTER 管脚
	I^2C2_SCL	I/O	MFP9	I^2C2 时钟脚
	UART1_TXD	O	MFP10	UART1 数据发送输出脚
PH.9	PH.9	I/O	MFP0	通用数字I/O脚
	EBI_AD13	I/O	MFP2	EBI地址/数据总线位13.
	QSPI0_SS	I/O	MFP3	QSPI0 从机选择脚
	SC2_RST	O	MFP4	智能卡 2 复位脚
	I^2S0_DO	O	MFP5	I^2S0 数据输出脚
	SPI1_SS	I/O	MFP6	SPI1 从机选择脚
	UART3_nCTS	I	MFP7	UART3 禁止发送输入管脚
	I^2C1_SMBSUS	O	MFP8	I^2C1 SMBus SMBSUS 管脚 (PMBus 控制管脚)
	I^2C2_SDA	I/O	MFP9	I^2C2 数据输入/输出脚
	UART1_RXD	I	MFP10	UART1数据接收输入管脚
PH.10	PH.10	I/O	MFP0	通用数字I/O脚
	EBI_AD14	I/O	MFP2	EBI地址/数据总线位14.

	管脚名称	类型	MFP	描述
	QSPI0_MISO1	I/O	MFP3	QSPI0 MISO1（主机输入，从机输出）脚.
	SC2_nCD	I	MFP4	智能卡 2 卡检测管脚.
	I²S0_LRCK	O	MFP5	I²S0左右通道时钟输出脚.
	SPI1_I2SMCLK	I/O	MFP6	SPI1 I²S 主机时钟输出脚
	UART4_TXD	O	MFP7	UART4 数据发送输出脚.
	UART0_TXD	O	MFP8	UART0 数据发送输出脚.
PH.11	PH.11	I/O	MFP0	通用数字I/O脚
	EBI_AD15	I/O	MFP2	EBI地址/数据总线位15.
	QSPI0_MOSI1	I/O	MFP3	QSPI0 MOSI1 (主机输出，从机输入) 管脚.
	UART4_RXD	I	MFP7	UART4 数据接收输入脚
	UART0_RXD	I	MFP8	UART0 数据接收输入脚
	EPWM0_CH5	I/O	MFP11	EPWM0 通道5 输出/捕获输入

4.3 M48xxE8AE/M48xxGCAE 管脚描述

4.3.1 M481 系列管脚描述

MFP* = 多功能管脚(参考 SYS_GPx_MFPL 和 SYS_GPx_MFPH)

PA.0 MFP0 意思是 SYS_GPA_MFPL[3:0] = 0x0.

PA.9 MFP5 意思是 SYS_GPA_MFPH[7:4] = 0x5.

32管脚	48管脚	64管脚	管脚名称	类型	MFP	描述
	48	1	PB.6	I/O	MFP0	通用数字I/O脚
			EADC0_CH6	A	MFP1	EADC0通道6模拟输入
			EBI_nWRH	O	MFP2	EBI 高字节写使能输出管脚
			CAN1_RXD	I	MFP5	CAN1 总线接收输入
			UART1_RXD	I	MFP6	UART1数据接收输入管脚
			EBI_nCS1	O	MFP8	EBI片选1输出管脚
			BPWM1_CH5	I/O	MFP10	EPWM1 通道5输出/捕获输入
			EPWM1_BRAKE1	I	MFP11	EPWM1 刹车1输入管脚
			EPWM1_CH5	I/O	MFP12	EPWM1 通道5输出/捕获输入
			INT4	I	MFP13	外部中断4输入管脚
			USB_VBUS_EN	O	MFP14	USB 外部 VBUS 电源使能管脚.
			ACMP1_O	O	MFP15	模拟比较器1输出脚
1	1	2	PB.5	I/O	MFP0	通用数字I/O脚
			EADC0_CH5	A	MFP1	EADC0 通道5模拟输入
			ACMP1_N	A	MFP1	模拟比较器 1 负极输入管脚.
			EBI_ADR0	O	MFP2	EBI地址总线位0
			SD0_DAT3	I/O	MFP3	SD/SDIO0数据线位3
			SPI1_MISO	I/O	MFP5	SPI1 MISO（主机输入，从机输出）脚.
			I^2C0_SCL	I/O	MFP6	I2C0 时钟管脚
			UART5_TXD	O	MFP7	UART5数据发送输出脚
			SC0_CLK	O	MFP9	智能卡0时钟管脚
			I^2S0_BCLK	O	MFP10	I2S0位时钟输出管脚
			EPWM0_CH0	I/O	MFP11	EPWM0 通道0输出/捕获输入
			UART2_TXD	O	MFP12	UART2 数据发送输出脚.
			TM0	I/O	MFP14	Timer0 事件计数输入/反转输出管脚

32 管脚	48 管脚	64 管脚	管脚名称	类型	MFP	描述
			INT0	I	MFP15	外部中断0输入管脚
2	2	3	PB.4	I/O	MFP0	通用数字I/O脚
			EADC0_CH4	A	MFP1	EADC0 通道4模拟输入
			ACMP1_P1	A	MFP1	模拟比较器1正极输入1脚
			EBI_ADR1	O	MFP2	EBI 地址总线位1
			SD0_DAT2	I/O	MFP3	SD/SDIO0 数据线位2
			SPI1_MOSI	I/O	MFP5	SPI1 MOSI (主机输出，从机输入) 管脚.
			I^2C0_SDA	I/O	MFP6	I2C0 数据输入/输出管脚
			UART5_RXD	I	MFP7	UART5 数据接收输入管脚
			SC0_DAT	I/O	MFP9	智能卡0数据输入脚
			I^2S0_MCLK	O	MFP10	I2S0 主时钟输出脚
			EPWM0_CH1	I/O	MFP11	EPWM0 通道1输出/捕获输入脚
			UART2_RXD	I	MFP12	UART2 数据接收输入脚
			TM1	I/O	MFP14	Timer1 事件计数输入/反转输出脚
			INT1	I	MFP15	外部中断1输入脚
3	3	4	PB.3	I/O	MFP0	通用数字I/O脚
			EADC0_CH3	A	MFP1	EADC0 通道3模拟输入
			EADC1_CH11	A	MFP1	EADC1 通道11 模拟输入
			ACMP0_N	A	MFP1	模拟比较器 0 负极输入管脚.
			EBI_ADR2	O	MFP2	EBI 地址总线位2
			SD0_DAT1	I/O	MFP3	SD/SDIO0 数据线位1
			SPI1_CLK	I/O	MFP5	SPI1 串行时钟脚
			UART1_TXD	O	MFP6	UART1 数据发送输出脚
			UART5_nRTS	O	MFP7	UART5 请求发送输出脚
			SC0_RST	O	MFP9	智能卡0复位脚
			I^2S0_DI	I	MFP10	I2S0 数据输入脚
			EPWM0_CH2	I/O	MFP11	EPWM0 通道2输出/捕获输入脚
			I^2C1_SCL	I/O	MFP12	I^2C1 时钟脚
			TM2	I/O	MFP14	Timer2 事件计数输入/反转输出脚
			INT2	I	MFP15	外部中断2输入脚

32 管脚	48 管脚	64 管脚	管脚名称	类型	MFP	描述
4	4	5	PB.2	I/O	MFP0	通用数字I/O脚
			EADC0_CH2	A	MFP1	EADC0 通道2 模拟输入
			EADC1_CH10	A	MFP1	EADC1 通道10 模拟输入
			ACMP0_P1	A	MFP1	模拟比较器0正极输入1脚
			EBI_ADR3	O	MFP2	EBI 地址总线位3
			SD0_DAT0	I/O	MFP3	SD/SDIO0 数据线位0
			SPI1_SS	I/O	MFP5	SPI1从机选择管脚.
			UART1_RXD	I	MFP6	UART1数据接收输入管脚
			UART5_nCTS	I	MFP7	UART5 禁止发送输入管脚
			SC0_PWR	O	MFP9	智能卡0电源脚
			I^2S0_DO	O	MFP10	I^2S0 数据输出脚
			EPWM0_CH3	I/O	MFP11	EPWM0 通道3输出/捕获输入脚
			I^2C1_SDA	I/O	MFP12	I^2C1 数据输入/输出脚
			TM3	I/O	MFP14	Timer3 事件计数输入/反转输出脚
			INT3	I	MFP15	外部中断3输入脚
5	5	6	PB.1	I/O	MFP0	通用数字I/O脚
			EADC0_CH1	A	MFP1	EADC0 通道1 模拟输入
			EADC1_CH9	A	MFP1	EADC1 通道9 模拟输入
			EBI_ADR8	O	MFP2	EBI 地址总线位 8.
			SD0_CLK	O	MFP3	SD/SDIO0 时钟输出脚
			SPI1_I2SMCLK	I/O	MFP5	SPI1 I^2S 主机时钟输出脚
			UART2_TXD	O	MFP7	UART2 数据发送输出脚.
			I^2C1_SCL	I/O	MFP9	I^2C1 时钟脚
			I^2S0_LRCK	O	MFP10	I^2S0左右通道时钟输出脚.
			EPWM0_CH4	I/O	MFP11	EPWM0 通道4输出/捕获输入脚
			EPWM1_CH4	I/O	MFP12	EPWM1 通道4输出/捕获输入脚
			EPWM0_BRAKE0	I	MFP13	EPWM0 刹车0 输入脚
6	6	7	PB.0	I/O	MFP0	通用数字I/O脚
			EADC0_CH0	A	MFP1	EADC0 通道0 模拟输入
			EADC1_CH8	A	MFP1	EADC1 通道8 模拟输入

32 管脚	48 管脚	64 管脚	管脚名称	类型	MFP	描述
			EBI_ADR9	O	MFP2	EBI 地址总线位 9.
			SD0_CMD	I/O	MFP3	SD/SDIO0 命令/响应脚
			SPI2_I2SMCLK	I/O	MFP4	SPI2 I^2S 主机时钟输出脚
			UART2_RXD	I	MFP7	UART2 数据接收输入脚
			SPI0_I2SMCLK	I/O	MFP8	SPI0 I^2S 主机时钟输出脚
			I^2C1_SDA	I/O	MFP9	I^2C1 数据输入/输出脚
			EPWM0_CH5	I/O	MFP11	EPWM0 通道5 输出/捕获输入
			EPWM1_CH5	I/O	MFP12	EPWM1 通道5输出/捕获输入
			EPWM0_BRAKE1	I	MFP13	EPWM0 刹车1 输入脚
	7	8	PA.11	I/O	MFP0	通用数字I/O脚
			EADC1_CH7	A	MFP1	EADC1 通道7 模拟输入
			ACMP0_P0	A	MFP1	模拟比较器 0 正极输入脚0
			EBI_nRD	O	MFP2	EBI 读使能输出脚
			SPI2_SS	I/O	MFP4	SPI2从机选择管脚.
			I^2C2_SCL	I/O	MFP7	I^2C2 时钟脚
			UART6_TXD	O	MFP8	UART6 数据发送输出脚.
			BPWM0_CH0	I/O	MFP9	BPWM0 通道0 输出/捕获输入
			EPWM0_SYNC_OUT	O	MFP10	EPWM0计数器同步触发输出脚
			TM0_EXT	I/O	MFP13	Timer0 外部捕获输入/反转输出脚
	8	9	PA.10	I/O	MFP0	通用数字I/O脚
			EADC1_CH6	A	MFP1	EADC1 通道6 模拟输入
			ACMP1_P0	A	MFP1	模拟比较器 1 正极输入脚0
			EBI_nWR	O	MFP2	EBI 写使能输出脚
			SPI2_CLK	I/O	MFP4	SPI2 串行时钟管脚
			I^2C2_SDA	I/O	MFP7	I^2C2 数据输入/输出脚
			UART6_RXD	I	MFP8	UART6 数据接收输入脚
			BPWM0_CH1	I/O	MFP9	BPWM0 通道1 输出/捕获输入
			QEI1_INDEX	I	MFP10	正交编码器1 索引输入
			ECAP0_IC0	I	MFP11	增强型捕获计数器 0 输入脚0
			TM1_EXT	I/O	MFP13	Timer1 外部捕获输入/反转输出脚

32 管脚	48 管脚	64 管脚	管脚名称	类型	MFP	描述
			DAC0_ST	I	MFP14	DAC0 外部触发输入
			SWDH_CLK	O	MFP15	串行调试总线主时钟输出
	9	10	PA.9	I/O	MFP0	通用数字I/O脚
			EADC1_CH5	A	MFP1	EADC1 通道5 模拟输入
			EBI_MCLK	O	MFP2	EBI 外部时钟输出脚.
			SPI2_MISO	I/O	MFP4	SPI2 MISO(主机输入，从机输出)管脚
			UART1_TXD	O	MFP7	UART1 数据发送输出脚
			UART7_TXD	O	MFP8	UART7 数据发送输出脚.
			BPWM0_CH2	I/O	MFP9	BPWM0 通道2 输出/捕获输入
			QEI1_A	I	MFP10	正交编码器1 相位A输入
			ECAP0_IC1	I	MFP11	增强型捕获计数器 0 输入 1 管脚.
			TM2_EXT	I/O	MFP13	Timer2 外部捕获输入/反转输出脚
			SWDH_DAT	I/O	MFP15	串行调试总线主机数据输入/输出管脚
	10	11	PA.8	I/O	MFP0	通用数字I/O脚
			EADC1_CH4	A	MFP1	EADC1 通道4 模拟输入
			EBI_ALE	O	MFP2	EBI地址锁存使能输出脚
			SPI2_MOSI	I/O	MFP4	SPI2 MOSI (主机输出，从机输入) 管脚
			UART1_RXD	I	MFP7	UART1数据接收输入管脚
			UART7_RXD	I	MFP8	UART7 数据接收输入脚
			BPWM0_CH3	I/O	MFP9	BPWM0 通道3输出/捕获输入脚
			QEI1_B	I	MFP10	正交编码器1 相位B 输入
			ECAP0_IC2	I	MFP11	增强型捕获计数器 0 输入 2 管脚.
			TM3_EXT	I/O	MFP13	Timer3 外部捕获输入/反转输出脚
			INT4	I	MFP15	外部中断4输入管脚
		12	PF.6	I/O	MFP0	通用数字I/O脚
			EBI_ADR19	O	MFP2	EBI 地址总线位 19.
			SC0_CLK	O	MFP3	智能卡0时钟管脚
			I^2S0_LRCK	O	MFP4	I^2S0左右通道时钟输出脚.
			SPI0_MOSI	I/O	MFP5	SPI0 MOSI (主机输出，从机输入) 管脚.
			UART4_RXD	I	MFP6	UART4 数据接收输入脚

32管脚	48管脚	64管脚	管脚名称	类型	MFP	描述
			EBI_nCS0	O	MFP7	EBI片选0输出脚
			CAN2_RXD	I	MFP8	CAN2 总线接收输入
			TAMPER0	I/O	MFP10	TAMPER循环检测管脚 0.
		13	V_{BAT}	P	MFP0	RTC电池供电
7	11	14	PF.5	I/O	MFP0	通用数字I/O脚
			UART2_RXD	I	MFP2	UART2 数据接收输入脚
			UART2_nCTS	I	MFP4	UART2 禁止发送输入管脚
			EPWM0_CH0	I/O	MFP7	EPWM0 通道0输出/捕获输入
			BPWM0_CH4	I/O	MFP8	BPWM0 通道4输出/捕获输入脚
			EPWM0_SYNC_OUT	O	MFP9	EPWM0计数器同步触发输出脚
			X32_IN	I	MFP10	外部 32.768 kHz 晶振输入脚
			EADC0_ST	I	MFP11	EADC0 外部触发输入
8	12	15	PF.4	I/O	MFP0	通用数字I/O脚
			UART2_TXD	O	MFP2	UART2 数据发送输出脚.
			UART2_nRTS	O	MFP4	UART2 请求发送输出脚
			EPWM0_CH1	I/O	MFP7	EPWM0 通道1输出/捕获输入脚
			BPWM0_CH5	I/O	MFP8	BPWM0 通道5 输出/捕获输入
			X32_OUT	O	MFP10	外部 32.768 kHz 晶振输出脚
			EADC1_ST	I	MFP11	EADC1 外部触发输入
9	13	16	PF.3	I/O	MFP0	通用数字I/O脚
			EBI_nCS0	O	MFP2	EBI片选0输出脚
			UART0_TXD	O	MFP3	UART0 数据发送输出脚.
			I^2C0_SCL	I/O	MFP4	I2C0 时钟管脚
			XT1_IN	I	MFP10	外部 4~24 MHz (高速) 晶振输入脚
			BPWM1_CH0	I/O	MFP11	BPWM1 通道0 输出/捕获输入
10	14	17	PF.2	I/O	MFP0	通用数字I/O脚
			EBI_nCS1	O	MFP2	EBI片选1输出管脚
			UART0_RXD	I	MFP3	UART0 数据接收输入脚
			I^2C0_SDA	I/O	MFP4	I2C0 数据输入/输出管脚
			QSPI0_CLK	I/O	MFP5	Quad SPI0 串行时钟脚

32 管脚	48 管脚	64 管脚	管脚名称	类型	MFP	描述
			XT1_OUT	O	MFP10	外部 4~24 MHz (高速) 晶振输出脚
			BPWM1_CH1	I/O	MFP11	BPWM1 通道1 输出/捕获输入
		18	PC.7	I/O	MFP0	通用数字I/O脚
			EBI_AD9	I/O	MFP2	EBI地址/数据总线位9.
			SPI1_MISO	I/O	MFP4	SPI1 MISO（主机输入，从机输出）脚.
			UART4_TXD	O	MFP5	UART4 数据发送输出脚.
			UART0_nCTS	I	MFP7	UART0 禁止发送输入管脚
			I^{2}C1_SMBAL	O	MFP8	I^{2}C1 SMBus SMBALTER 管脚
			UART6_TXD	O	MFP9	UART6 数据发送输出脚.
			EPWM1_CH2	I/O	MFP11	EPWM1 通道2 输出/捕获输入
			BPWM1_CH0	I/O	MFP12	BPWM1 通道0 输出/捕获输入
			TM0	I/O	MFP14	Timer0 事件计数输入/反转输出管脚
			INT3	I	MFP15	外部中断3输入脚
		19	PC.6	I/O	MFP0	通用数字I/O脚
			EBI_AD8	I/O	MFP2	EBI地址/数据总线位8.
			SPI1_MOSI	I/O	MFP4	SPI1 MOSI (主机输出，从机输入) 管脚.
			UART4_RXD	I	MFP5	UART4 数据接收输入脚
			UART0_nRTS	O	MFP7	UART0 请求发送输出脚
			I^{2}C1_SMBSUS	O	MFP8	I^{2}C1 SMBus SMBSUS 管脚 (PMBus 控制管脚)
			UART6_RXD	I	MFP9	UART6 数据接收输入脚
			EPWM1_CH3	I/O	MFP11	EPWM1 通道3输出/捕获输入脚
			BPWM1_CH1	I/O	MFP12	BPWM1 通道1 输出/捕获输入
			TM1	I/O	MFP14	Timer1 事件计数输入/反转输出脚
			INT2	I	MFP15	外部中断2输入脚
	15	20	PA.7	I/O	MFP0	通用数字I/O脚
			EBI_AD7	I/O	MFP2	EBI地址/数据总线位7.
			SPI1_CLK	I/O	MFP4	SPI1 串行时钟脚
			UART0_TXD	O	MFP7	UART0 数据发送输出脚.
			I^{2}C1_SCL	I/O	MFP8	I^{2}C1 时钟脚
			QSPI1_MISO1	I/O	MFP9	Quad SPI1 MISO1（主机输入，从机输出）脚.

32管脚	48管脚	64管脚	管脚名称	类型	MFP	描述
			EPWM1_CH4	I/O	MFP11	EPWM1 通道4输出/捕获输入脚
			BPWM1_CH2	I/O	MFP12	BPWM1 通道2 输出/捕获输入
			ACMP0_WLAT	I	MFP13	模拟比较器 0窗口锁存输入管脚
			TM2	I/O	MFP14	Timer2 事件计数输入/反转输出脚
			INT1	I	MFP15	外部中断1输入脚
	16	21	PA.6	I/O	MFP0	通用数字I/O脚
			EBI_AD6	I/O	MFP2	EBI地址/数据总线位6.
			SPI1_SS	I/O	MFP4	SPI1从机选择管脚.
			UART0_RXD	I	MFP7	UART0 数据接收输入脚
			I^2C1_SDA	I/O	MFP8	I^2C1 数据输入/输出脚
			QSPI1_MOSI1	I/O	MFP9	Quad SPI1 MOSI1 (主机输出，从机输入) 管脚.
			EPWM1_CH5	I/O	MFP11	EPWM1 通道5输出/捕获输入
			BPWM1_CH3	I/O	MFP12	BPWM1 通道3输出/捕获输入脚
			ACMP1_WLAT	I	MFP13	模拟比较器 1窗口锁存输入管脚
			TM3	I/O	MFP14	Timer3 事件计数输入/反转输出脚
			INT0	I	MFP15	外部中断0输入管脚
		22	V_{SS}	P	MFP0	数字电路地管脚
		23	V_{DD}	P	MFP0	I/O和内部PLL及数字电路的LDO电源
		24	LDO_CAP	A	MFP0	LDO输出脚
	17	25	PA.5	I/O	MFP0	通用数字I/O脚
			QSPI0_MISO1	I/O	MFP3	Quad SPI0 MISO1（主机输入，从机输出）脚.
			SPI1_I2SMCLK	I/O	MFP4	SPI1 I^2S 主机时钟输出脚
			UART0_nCTS	I	MFP7	UART0 禁止发送输入管脚
			UART5_TXD	O	MFP8	UART5数据发送输出脚
			I^2C0_SCL	I/O	MFP9	I2C0 时钟管脚
			CAN0_TXD	O	MFP10	CAN0总线发送输出
			UART0_TXD	O	MFP11	UART0 数据发送输出脚.
			BPWM0_CH5	I/O	MFP12	BPWM0 通道5 输出/捕获输入
			EPWM0_CH0	I/O	MFP13	EPWM0 通道0输出/捕获输入
			QEI0_INDEX	I	MFP14	正交编码器0 索引输入

32 管脚	48 管脚	64 管脚	管脚名称	类型	MFP	描述
	18	26	PA.4	I/O	MFP0	通用数字I/O脚
			QSPI0_MOSI1	I/O	MFP3	Quad SPI0 MOSI1 (主机输出，从机输入) 管脚.
			SPI0_I2SMCLK	I/O	MFP4	SPI0 I^2S 主机时钟输出脚
			SC0_nCD	I	MFP6	智能卡 0 卡检测管脚.
			UART0_nRTS	O	MFP7	UART0 请求发送输出脚
			UART5_RXD	I	MFP8	UART5 数据接收输入管脚
			I^2C0_SDA	I/O	MFP9	I2C0 数据输入/输出管脚
			CAN0_RXD	I	MFP10	CAN0 总线接收输入
			UART0_RXD	I	MFP11	UART0 数据接收输入脚
			BPWM0_CH4	I/O	MFP12	BPWM0 通道4输出/捕获输入脚
			EPWM0_CH1	I/O	MFP13	EPWM0 通道1输出/捕获输入脚
			QEI0_A	I	MFP14	正交编码器0 相位A输入
11	19	27	PA.3	I/O	MFP0	通用数字I/O脚
			QSPI0_SS	I/O	MFP3	Quad SPI0从机选择管脚.
			SPI0_SS	I/O	MFP4	SPI0从机选择管脚.
			SC0_PWR	O	MFP6	智能卡0电源脚
			UART4_TXD	O	MFP7	UART4 数据发送输出脚.
			UART1_TXD	O	MFP8	UART1 数据发送输出脚
			I^2C1_SCL	I/O	MFP9	I^2C1 时钟脚
			I^2C0_SMBAL	O	MFP10	I^2C0 SMBus SMBALTER 管脚
			BPWM0_CH3	I/O	MFP12	BPWM0 通道3输出/捕获输入脚
			EPWM0_CH2	I/O	MFP13	EPWM0 通道2输出/捕获输入脚
			QEI0_B	I	MFP14	正交编码器0 相位B 输入
			EPWM1_BRAKE1	I	MFP15	EPWM1 刹车1输入管脚
12	20	28	PA.2	I/O	MFP0	通用数字I/O脚
			QSPI0_CLK	I/O	MFP3	Quad SPI0 串行时钟脚
			SPI0_CLK	I/O	MFP4	SPI0 串行时钟脚
			SC0_RST	O	MFP6	智能卡0复位脚
			UART4_RXD	I	MFP7	UART4 数据接收输入脚
			UART1_RXD	I	MFP8	UART1数据接收输入管脚

32 管脚	48 管脚	64 管脚	管脚名称	类型	MFP	描述
			I^2C1_SDA	I/O	MFP9	I^2C1 数据输入/输出脚
			I^2C0_SMBSUS	O	MFP10	I^2C0 SMBus SMBSUS 管脚 (PMBus 控制管脚)
			BPWM0_CH2	I/O	MFP12	BPWM0 通道2 输出/捕获输入
			EPWM0_CH3	I/O	MFP13	EPWM0 通道3输出/捕获输入脚
13	21	29	PA.1	I/O	MFP0	通用数字I/O脚
			QSPI0_MISO0	I/O	MFP3	Quad SPI0 MISO0（主机输入，从机输出）脚.
			SPI0_MISO	I/O	MFP4	SPI0 MISO（主机输入，从机输出）脚.
			SC0_DAT	I/O	MFP6	智能卡0数据输入脚
			UART0_TXD	O	MFP7	UART0 数据发送输出脚.
			UART1_nCTS	I	MFP8	UART1 禁止发送输入管脚
			I^2C2_SCL	I/O	MFP9	I^2C2 时钟脚
			CAP_DATA7	I	MFP10	图像数据输入总线位7.
			BPWM0_CH1	I/O	MFP12	BPWM0 通道1 输出/捕获输入
			EPWM0_CH4	I/O	MFP13	EPWM0 通道4输出/捕获输入脚
14	22	30	PA.0	I/O	MFP0	通用数字I/O脚
			QSPI0_MOSI0	I/O	MFP3	Quad SPI0 MOSI0 (主机输出，从机输入) 管脚.
			SPI0_MOSI	I/O	MFP4	SPI0 MOSI (主机输出，从机输入) 管脚.
			SC0_CLK	O	MFP6	智能卡0时钟管脚
			UART0_RXD	I	MFP7	UART0 数据接收输入脚
			UART1_nRTS	O	MFP8	UART1 请求发送输出脚
			I^2C2_SDA	I/O	MFP9	I^2C2 数据输入/输出脚
			CAP_DATA6	I	MFP10	图像数据输入总线位6.
			BPWM0_CH0	I/O	MFP12	BPWM0 通道0 输出/捕获输入
			EPWM0_CH5	I/O	MFP13	EPWM0 通道5 输出/捕获输入
			DAC0_ST	I	MFP15	DAC0 外部触发输入
15	23	31	V_{DDIO}	P	MFP0	PA.0~PA.5的电源
16	24	32	nRESET	I	MFP0	外部复位输入:低电平有效,有内部上拉，设置此管脚为低将复位初始状态
17	25	33	PF.0	I/O	MFP0	通用数字I/O脚
			UART1_TXD	O	MFP2	UART1 数据发送输出脚
			I^2C1_SCL	I/O	MFP3	I^2C1 时钟脚

32 管脚	48 管脚	64 管脚	管脚名称	类型	MFP	描述
			UART0_TXD	O	MFP4	UART0 数据发送输出脚.
			BPWM1_CH0	I/O	MFP12	BPWM1 通道0 输出/捕获输入
			ICE_DAT	I/O	MFP14	ADC 外部触发输入
18	26	34	PF.1	I/O	MFP0	通用数字I/O脚
			UART1_RXD	I	MFP2	UART1数据接收输入管脚
			I^{2}C1_SDA	I/O	MFP3	I^{2}C1 数据输入/输出脚
			UART0_RXD	I	MFP4	UART0 数据接收输入脚
			BPWM1_CH1	I/O	MFP12	BPWM1 通道1 输出/捕获输入
			ICE_CLK	I/O	MFP14	ADC 外部触发输入
	27	35	PC.5	I/O	MFP0	通用数字I/O脚
			EBI_AD5	I/O	MFP2	EBI地址/数据总线位5.
			QSPI0_MISO1	I/O	MFP4	Quad SPI0 MISO1（主机输入，从机输出）脚.
			UART2_TXD	O	MFP8	UART2 数据发送输出脚.
			I^{2}C1_SCL	I/O	MFP9	I^{2}C1 时钟脚
			CAN0_TXD	O	MFP10	CAN0总线发送输出
			UART4_TXD	O	MFP11	UART4 数据发送输出脚.
			EPWM1_CH0	I/O	MFP12	EPWM1 通道0 输出/捕获输入
			CAP_DATA5	I	MFP13	图像数据输入总线位5.
			QSPI1_SS	I/O	MFP14	Quad SPI1从机选择管脚.
	28	36	PC.4	I/O	MFP0	通用数字I/O脚
			EBI_AD4	I/O	MFP2	EBI地址/数据总线位4.
			QSPI0_MOSI1	I/O	MFP4	Quad SPI0 MOSI1 (主机输出，从机输入) 管脚.
			I^{2}S0_BCLK	O	MFP6	I2S0位时钟输出管脚
			SPI1_I2SMCLK	I/O	MFP7	SPI1 I^{2}S 主机时钟输出脚
			UART2_RXD	I	MFP8	UART2 数据接收输入脚
			I^{2}C1_SDA	I/O	MFP9	I^{2}C1 数据输入/输出脚
			CAN0_RXD	I	MFP10	CAN0 总线接收输入
			UART4_RXD	I	MFP11	UART4 数据接收输入脚
			EPWM1_CH1	I/O	MFP12	EPWM1 通道1 输出/捕获输入
			CAP_DATA4	I	MFP13	图像数据输入总线位4.

32 管脚	48 管脚	64 管脚	管脚名称	类型	MFP	描述
			QSPI1_CLK	I/O	MFP14	Quad SPI1 串行时钟脚
	29	37	PC.3	I/O	MFP0	通用数字I/O脚
			EBI_AD3	I/O	MFP2	EBI地址/数据总线位3.
			QSPI0_SS	I/O	MFP4	Quad SPI0从机选择管脚.
			I^2S0_MCLK	O	MFP6	I2S0 主时钟输出脚
			SPI1_MISO	I/O	MFP7	SPI1 MISO（主机输入，从机输出）脚.
			UART2_nRTS	O	MFP8	UART2 请求发送输出脚
			I^2C0_SMBAL	O	MFP9	I^2C0 SMBus SMBALTER 管脚
			CAN1_TXD	O	MFP10	CAN1总线发送输出
			UART3_TXD	O	MFP11	UART3 数据发送输出脚.
			EPWM1_CH2	I/O	MFP12	EPWM1 通道2 输出/捕获输入
			CAP_DATA3	I	MFP13	图像数据输入总线位3.
			QSPI1_MISO0	I/O	MFP14	Quad SPI1 MISO0（主机输入，从机输出）脚.
	30	38	PC.2	I/O	MFP0	通用数字I/O脚
			EBI_AD2	I/O	MFP2	EBI地址/数据总线位2.
			QSPI0_CLK	I/O	MFP4	Quad SPI0 串行时钟脚
			I^2S0_DI	I	MFP6	I2S0 数据输入脚
			SPI1_MOSI	I/O	MFP7	SPI1 MOSI (主机输出，从机输入) 管脚.
			UART2_nCTS	I	MFP8	UART2 禁止发送输入管脚
			I^2C0_SMBSUS	O	MFP9	I^2C0 SMBus SMBSUS 管脚 (PMBus 控制管脚)
			CAN1_RXD	I	MFP10	CAN1 总线接收输入
			UART3_RXD	I	MFP11	UART3 数据接收输入脚
			EPWM1_CH3	I/O	MFP12	EPWM1 通道3输出/捕获输入脚
			CAP_DATA2	I	MFP13	图像数据输入总线位2.
			QSPI1_MOSI0	I/O	MFP14	Quad SPI1 MOSI0 (主机输出，从机输入) 管脚.
19	31	39	PC.1	I/O	MFP0	通用数字I/O脚
			EBI_AD1	I/O	MFP2	EBI地址/数据总线位1.
			QSPI0_MISO0	I/O	MFP4	Quad SPI0 MISO0（主机输入，从机输出）脚.
			I^2S0_DO	O	MFP6	I^2S0 数据输出脚
			SPI1_CLK	I/O	MFP7	SPI1 串行时钟脚

32 管脚	48 管脚	64 管脚	管脚名称	类型	MFP	描述
			UART2_TXD	O	MFP8	UART2 数据发送输出脚.
			I²C0_SCL	I/O	MFP9	I2C0 时钟管脚
			CAN2_TXD	O	MFP10	CAN2总线发送输出
			EPWM1_CH4	I/O	MFP12	EPWM1 通道4输出/捕获输入脚
			CAP_DATA1	I	MFP13	图像数据输入总线位1.
			ACMP0_O	O	MFP14	模拟比较器 0输出脚
			EADC0_ST	I	MFP15	EADC0 外部触发输入
20	32	40	PC.0	I/O	MFP0	通用数字I/O脚
			EBI_AD0	I/O	MFP2	EBI地址/数据总线位0.
			QSPI0_MOSI0	I/O	MFP4	Quad SPI0 MOSI0 (主机输出，从机输入) 管脚.
			I²S0_LRCK	O	MFP6	I²S0左右通道时钟输出脚.
			SPI1_SS	I/O	MFP7	SPI1从机选择管脚.
			UART2_RXD	I	MFP8	UART2 数据接收输入脚
			I²C0_SDA	I/O	MFP9	I2C0 数据输入/输出管脚
			CAN2_RXD	I	MFP10	CAN2 总线接收输入
			EPWM1_CH5	I/O	MFP12	EPWM1 通道5输出/捕获输入
			CAP_DATA0	I	MFP13	图像数据输入总线位0.
			ACMP1_O	O	MFP14	模拟比较器1输出脚
			EADC1_ST	I	MFP15	EADC1 外部触发输入
		41	PD.3	I/O	MFP0	通用数字I/O脚
			EBI_AD10	I/O	MFP2	EBI地址/数据总线位10.
			SPI0_SS	I/O	MFP4	SPI0从机选择管脚.
			UART3_nRTS	O	MFP5	UART3 请求发送输出脚
			UART0_TXD	O	MFP9	UART0 数据发送输出脚.
		42	PD.2	I/O	MFP0	通用数字I/O脚
			EBI_AD11	I/O	MFP2	EBI地址/数据总线位11.
			SPI0_CLK	I/O	MFP4	SPI0 串行时钟脚
			UART3_nCTS	I	MFP5	UART3 禁止发送输入管脚
			UART0_RXD	I	MFP9	UART0 数据接收输入脚
		43	PD.1	I/O	MFP0	通用数字I/O脚

32 管脚	48 管脚	64 管脚	管脚名称	类型	MFP	描述
			EBI_AD12	I/O	MFP2	EBI地址/数据总线位12.
			SPI0_MISO	I/O	MFP4	SPI0 MISO（主机输入，从机输出）脚.
			UART3_TXD	O	MFP5	UART3 数据发送输出脚.
			I^2C2_SCL	I/O	MFP6	I^2C2 时钟脚
		44	PD.0	I/O	MFP0	通用数字I/O脚
			EBI_AD13	I/O	MFP2	EBI地址/数据总线位13.
			SPI0_MOSI	I/O	MFP4	SPI0 MOSI (主机输出，从机输入) 管脚.
			UART3_RXD	I	MFP5	UART3 数据接收输入脚
			I^2C2_SDA	I/O	MFP6	I^2C2 数据输入/输出脚
			TM2	I/O	MFP14	Timer2 事件计数输入/反转输出脚
21	33	45	PA.12	I/O	MFP0	通用数字I/O脚
			I^2S0_BCLK	O	MFP2	I2S0位时钟输出管脚
			UART4_TXD	O	MFP3	UART4 数据发送输出脚.
			I^2C1_SCL	I/O	MFP4	I^2C1 时钟脚
			SPI2_SS	I/O	MFP5	SPI2从机选择管脚.
			CAN0_TXD	O	MFP6	CAN0总线发送输出
			BPWM1_CH2	I/O	MFP11	BPWM1 通道2 输出/捕获输入
			QEI1_INDEX	I	MFP12	正交编码器1 索引输入
			USB_VBUS	P	MFP14	USB主机或HUB的电源
22	34	46	PA.13	I/O	MFP0	通用数字I/O脚
			I^2S0_MCLK	O	MFP2	I2S0 主时钟输出脚
			UART4_RXD	I	MFP3	UART4 数据接收输入脚
			I^2C1_SDA	I/O	MFP4	I^2C1 数据输入/输出脚
			SPI2_CLK	I/O	MFP5	SPI2 串行时钟管脚
			CAN0_RXD	I	MFP6	CAN0 总线接收输入
			BPWM1_CH3	I/O	MFP11	BPWM1 通道3输出/捕获输入脚
			QEI1_A	I	MFP12	正交编码器1 相位A输入
			USB_D-	A	MFP14	USB差分信号D-.
23	35	47	PA.14	I/O	MFP0	通用数字I/O脚
			I^2S0_DI	I	MFP2	I2S0 数据输入脚

32 管脚	48 管脚	64 管脚	管脚名称	类型	MFP	描述
			UART0_TXD	O	MFP3	UART0 数据发送输出脚.
			SPI2_MISO	I/O	MFP5	SPI2 MISO(主机输入，从机输出)管脚
			I^{2}C2_SCL	I/O	MFP6	I^{2}C2 时钟脚
			BPWM1_CH4	I/O	MFP11	BPWM1 通道4输出/捕获输入脚
			QEI1_B	I	MFP12	正交编码器1 相位B 输入
			USB_D+	A	MFP14	USB差分信号D+.
24	36	48	PA.15	I/O	MFP0	通用数字I/O脚
			I^{2}S0_DO	O	MFP2	I^{2}S0 数据输出脚
			UART0_RXD	I	MFP3	UART0 数据接收输入脚
			SPI2_MOSI	I/O	MFP5	SPI2 MOSI (主机输出，从机输入) 管脚
			I^{2}C2_SDA	I/O	MFP6	I^{2}C2 数据输入/输出脚
			BPWM1_CH5	I/O	MFP11	EPWM1 通道5输出/捕获输入
			EPWM0_SYNC_IN	I	MFP12	EPWM0 计数器同步触发输入脚
			USB_OTG_ID	I	MFP14	USB_ 标识
25	37	49	V_{SS}	P	MFP0	数字电路地管脚
26	38	50	LDO_CAP	A	MFP0	LDO输出脚
27	39	51	V_{DD}	P	MFP0	I/O和内部PLL及数字电路的LDO电源
	40	52	PC.14	I/O	MFP0	通用数字I/O脚
			EBI_AD11	I/O	MFP2	EBI地址/数据总线位11.
			SPI0_I2SMCLK	I/O	MFP4	SPI0 I^{2}S 主机时钟输出脚
			QSPI0_CLK	I/O	MFP6	Quad SPI0 串行时钟脚
			EPWM0_SYNC_IN	I	MFP11	EPWM0 计数器同步触发输入脚
			ETM_TRACE_CLK	I	MFP12	ETM接收追踪时钟输入管脚
			TM1	I/O	MFP13	Timer1 事件计数输入/反转输出脚
			USB_VBUS_ST	I	MFP14	USB 外部VBUS 电源状态管脚.
28	41	53	PB.15	I/O	MFP0	通用数字I/O脚
			EADC0_CH15	A	MFP1	EADC0 通道15 模拟输入
			EADC1_CH15	A	MFP1	EADC1 通道15 模拟输入
			EBI_AD12	I/O	MFP2	EBI地址/数据总线位12.
			SPI0_SS	I/O	MFP4	SPI0从机选择管脚.

32 管脚	48 管脚	64 管脚	管脚名称	类型	MFP	描述
			UART0_nCTS	I	MFP6	UART0 禁止发送输入管脚
			UART3_TXD	O	MFP7	UART3 数据发送输出脚.
			I^{2}C2_SMBAL	O	MFP8	I^{2}C2 SMBus SMBALTER 管脚
			EPWM0_BRAKE1	I	MFP10	EPWM0 刹车1 输入脚
			EPWM1_CH0	I/O	MFP11	EPWM1 通道0 输出/捕获输入
			ETM_TRACE_DATA0	I	MFP12	ETM接收追踪数据0 输入管脚
			TM0_EXT	I/O	MFP13	Timer0 外部捕获输入/反转输出脚
			USB_VBUS_EN	O	MFP14	USB 外部 VBUS 电源使能管脚.
29	42	54	PB.14	I/O	MFP0	通用数字I/O脚
			EADC0_CH14	A	MFP1	EADC0 通道14 模拟输入
			EADC1_CH14	A	MFP1	EADC1 通道14 模拟输入
			EBI_AD13	I/O	MFP2	EBI地址/数据总线位13.
			SPI0_CLK	I/O	MFP4	SPI0 串行时钟脚
			UART0_nRTS	O	MFP6	UART0 请求发送输出脚
			UART3_RXD	I	MFP7	UART3 数据接收输入脚
			I^{2}C2_SMBSUS	O	MFP8	I^{2}C2 SMBus SMBSUS 管脚 (PMBus 控制管脚)
			EPWM1_CH1	I/O	MFP11	EPWM1 通道1 输出/捕获输入
			ETM_TRACE_DATA1	I	MFP12	ETM接收追踪数据1 输入管脚
			TM1_EXT	I/O	MFP13	Timer1 外部捕获输入/反转输出脚
			CLKO	O	MFP14	时钟输出
			USB_VBUS_ST	I	MFP15	USB 外部VBUS 电源状态管脚.
30	43	55	PB.13	I/O	MFP0	通用数字I/O脚
			EADC0_CH13	A	MFP1	EADC0 通道13 模拟输入
			EADC1_CH13	A	MFP1	EADC1 通道13 模拟输入
			ACMP0_P3	A	MFP1	模拟比较器 0 正极输入 3 管脚.
			ACMP1_P3	A	MFP1	模拟比较器 1 正极输入 3 管脚.
			EBI_AD14	I/O	MFP2	EBI地址/数据总线位14.
			SPI0_MISO	I/O	MFP4	SPI0 MISO（主机输入，从机输出）脚.
			UART0_TXD	O	MFP6	UART0 数据发送输出脚.
			UART3_nRTS	O	MFP7	UART3 请求发送输出脚

32 管脚	48 管脚	64 管脚	管脚名称	类型	MFP	描述
			I^2C2_SCL	I/O	MFP8	I^2C2 时钟脚
			CAP_PIXCLK	I	MFP10	图像采集接口像素时钟输入脚
			EPWM1_CH2	I/O	MFP11	EPWM1 通道2 输出/捕获输入
			ETM_TRACE_DATA2	I	MFP12	ETM接收追踪数据2 输入管脚
			TM2_EXT	I/O	MFP13	Timer2 外部捕获输入/反转输出脚
31	44	56	PB.12	I/O	MFP0	通用数字I/O脚
			EADC0_CH12	A	MFP1	EADC0 通道12 模拟输入
			EADC1_CH12	A	MFP1	EADC1 通道12 模拟输入
			DAC0_OUT	A	MFP1	DAC0 模拟输出通道
			ACMP0_P2	A	MFP1	模拟比较器 0 正极输入 2 管脚.
			ACMP1_P2	A	MFP1	模拟比较器 1 正极输入 2 管脚.
			EBI_AD15	I/O	MFP2	EBI地址/数据总线位15.
			SPI0_MOSI	I/O	MFP4	SPI0 MOSI (主机输出，从机输入) 管脚.
			UART0_RXD	I	MFP6	UART0 数据接收输入脚
			UART3_nCTS	I	MFP7	UART3 禁止发送输入管脚
			I^2C2_SDA	I/O	MFP8	I^2C2 数据输入/输出脚
			SD0_nCD	I	MFP9	SD/SDIO0 卡检测输入管脚
			CAP_SCLK	O	MFP10	图像采集接口传感器时钟管脚.
			EPWM1_CH3	I/O	MFP11	EPWM1 通道3输出/捕获输入脚
			ETM_TRACE_DATA3	I	MFP12	ETM接收追踪数据3 输入管脚
			TM3_EXT	I/O	MFP13	Timer3 外部捕获输入/反转输出脚
32	45	57	AV_{DD}	P	MFP0	内部模拟电路电源
		58	V_{REF}	A	MFP0	ADC参考电压输入. 注: 此管脚需接1uF电容到地.
	46	59	AV_{SS}	P	MFP0	模拟电路地管脚
		60	PB.11	I/O	MFP0	通用数字I/O脚
			EADC0_CH11	A	MFP1	EADC0 通道11 模拟输入
			EBI_ADR16	O	MFP2	EBI 地址总线位 16.
			UART0_nCTS	I	MFP5	UART0 禁止发送输入管脚
			UART4_TXD	O	MFP6	UART4 数据发送输出脚.
			I^2C1_SCL	I/O	MFP7	I^2C1 时钟脚

32 管脚	48 管脚	64 管脚	管脚名称	类型	MFP	描述
			CAN0_TXD	O	MFP8	CAN0总线发送输出
			SPI0_I2SMCLK	I/O	MFP9	SPI0 I^{2}S 主机时钟输出脚
			BPWM1_CH0	I/O	MFP10	BPWM1 通道0 输出/捕获输入
			CAP_SFIELD	I	MFP12	图像输入接口SFIELD 输入脚
		61	PB.10	I/O	MFP0	通用数字I/O脚
			EADC0_CH10	A	MFP1	EADC0 通道10 模拟输入
			EBI_ADR17	O	MFP2	EBI 地址总线位 17.
			UART0_nRTS	O	MFP5	UART0 请求发送输出脚
			UART4_RXD	I	MFP6	UART4 数据接收输入脚
			I^{2}C1_SDA	I/O	MFP7	I^{2}C1 数据输入/输出脚
			CAN0_RXD	I	MFP8	CAN0 总线接收输入
			BPWM1_CH1	I/O	MFP10	BPWM1 通道1 输出/捕获输入
			CAP_VSYNC	I	MFP12	图像采集接口vsync 输入脚
		62	PB.9	I/O	MFP0	通用数字I/O脚
			EADC0_CH9	A	MFP1	EADC0 通道9 模拟输入
			EBI_ADR18	O	MFP2	EBI 地址总线位 18.
			UART0_TXD	O	MFP5	UART0 数据发送输出脚.
			UART1_nCTS	I	MFP6	UART1 禁止发送输入管脚
			I^{2}C1_SMBAL	O	MFP7	I^{2}C1 SMBus SMBALTER 管脚
			UART7_TXD	O	MFP8	UART7 数据发送输出脚.
			I^{2}C0_SCL	I/O	MFP9	I2C0 时钟管脚
			BPWM1_CH2	I/O	MFP10	BPWM1 通道2 输出/捕获输入
			CAN2_TXD	O	MFP12	CAN2总线发送输出
			INT7	I	MFP13	外部中断7 输入脚
			CAP_HSYNC	I	MFP14	图像采集接口hsync 输入脚
		63	PB.8	I/O	MFP0	通用数字I/O脚
			EADC0_CH8	A	MFP1	EADC0 通道8 模拟输入
			EBI_ADR19	O	MFP2	EBI 地址总线位 19.
			UART0_RXD	I	MFP5	UART0 数据接收输入脚
			UART1_nRTS	O	MFP6	UART1 请求发送输出脚

32 管脚	48 管脚	64 管脚	管脚名称	类型	MFP	描述
			I^2C1_SMBSUS	O	MFP7	I^2C1 SMBus SMBSUS 管脚 (PMBus 控制管脚)
			UART7_RXD	I	MFP8	UART7 数据接收输入脚
			I^2C0_SDA	I/O	MFP9	I2C0 数据输入/输出管脚
			BPWM1_CH3	I/O	MFP10	BPWM1 通道3输出/捕获输入脚
			CAN2_RXD	I	MFP12	CAN2 总线接收输入
			INT6	I	MFP13	外部中断6 输入脚
	47	64	PB.7	I/O	MFP0	通用数字I/O脚
			EADC0_CH7	A	MFP1	EADC0 通道7 模拟输入
			EBI_nWRL	O	MFP2	EBI低字节写使能输出脚
			CAN1_TXD	O	MFP5	CAN1总线发送输出
			UART1_TXD	O	MFP6	UART1 数据发送输出脚
			EBI_nCS0	O	MFP8	EBI片选0输出脚
			BPWM1_CH4	I/O	MFP10	BPWM1 通道4输出/捕获输入脚
			EPWM1_BRAKE0	I	MFP11	EPWM1 刹车0 输入脚
			EPWM1_CH4	I/O	MFP12	EPWM1 通道4输出/捕获输入脚
			INT5	I	MFP13	外部中断5 输入脚
			USB_VBUS_ST	I	MFP14	USB 外部VBUS 电源状态管脚.
			ACMP0_O	O	MFP15	模拟比较器 0输出脚

4.3.2 M482系列管脚描述

32管脚	48管脚	64管脚	128管脚	管脚名称	类型	MFP	描述
1	1	2	1	PB.5	I/O	MFP0	通用数字I/O脚
				EADC0_CH5	A	MFP1	EADC0 通道5模拟输入
				ACMP1_N	A	MFP1	模拟比较器 1 负极输入管脚.
				EBI_ADR0	O	MFP2	EBI地址总线位0
				SD0_DAT3	I/O	MFP3	SD/SDIO0数据线位3
				SPI1_MISO	I/O	MFP5	SPI1 MISO（主机输入，从机输出）脚.
				I^2C0_SCL	I/O	MFP6	I2C0 时钟管脚
				UART5_TXD	O	MFP7	UART5数据发送输出脚
				SC0_CLK	O	MFP9	智能卡0时钟管脚
				I^2S0_BCLK	O	MFP10	I2S0位时钟输出管脚
				EPWM0_CH0	I/O	MFP11	EPWM0 通道0输出/捕获输入
				UART2_TXD	O	MFP12	UART2 数据发送输出脚.
				TM0	I/O	MFP14	Timer0 事件计数输入/反转输出管脚
				INT0	I	MFP15	外部中断0输入管脚
2	2	3	2	PB.4	I/O	MFP0	通用数字I/O脚
				EADC0_CH4	A	MFP1	EADC0 通道4模拟输入
				ACMP1_P1	A	MFP1	模拟比较器1正极输入1脚
				EBI_ADR1	O	MFP2	EBI 地址总线位1
				SD0_DAT2	I/O	MFP3	SD/SDIO0 数据线位2
				SPI1_MOSI	I/O	MFP5	SPI1 MOSI (主机输出，从机输入) 管脚.
				I^2C0_SDA	I/O	MFP6	I2C0 数据输入/输出管脚
				UART5_RXD	I	MFP7	UART5 数据接收输入管脚
				SC0_DAT	I/O	MFP9	智能卡0数据输入脚
				I^2S0_MCLK	O	MFP10	I2S0 主时钟输出脚
				EPWM0_CH1	I/O	MFP11	EPWM0 通道1输出/捕获输入脚
				UART2_RXD	I	MFP12	UART2 数据接收输入脚
				TM1	I/O	MFP14	Timer1 事件计数输入/反转输出脚
				INT1	I	MFP15	外部中断1输入脚
3	3	4	3	PB.3	I/O	MFP0	通用数字I/O脚

32 管脚	48 管脚	64 管脚	128 管脚	管脚名称	类型	MFP	描述
				EADC0_CH3	A	MFP1	EADC0 通道3模拟输入
				EADC1_CH11	A	MFP1	EADC1 通道11 模拟输入
				ACMP0_N	A	MFP1	模拟比较器 0 负极输入管脚.
				EBI_ADR2	O	MFP2	EBI 地址总线位2
				SD0_DAT1	I/O	MFP3	SD/SDIO0 数据线位1
				SPI1_CLK	I/O	MFP5	SPI1 串行时钟脚
				UART1_TXD	O	MFP6	UART1 数据发送输出脚
				UART5_nRTS	O	MFP7	UART5 请求发送输出脚
				SC0_RST	O	MFP9	智能卡0复位脚
				I^2S0_DI	I	MFP10	I2S0 数据输入脚
				EPWM0_CH2	I/O	MFP11	EPWM0 通道2输出/捕获输入脚
				I^2C1_SCL	I/O	MFP12	I^2C1 时钟脚
				TM2	I/O	MFP14	Timer2 事件计数输入/反转输出脚
				INT2	I	MFP15	外部中断2输入脚
4	4	5	4	PB.2	I/O	MFP0	通用数字I/O脚
				EADC0_CH2	A	MFP1	EADC0 通道2 模拟输入
				EADC1_CH10	A	MFP1	EADC1 通道10 模拟输入
				ACMP0_P1	A	MFP1	模拟比较器0正极输入1脚
				EBI_ADR3	O	MFP2	EBI 地址总线位3
				SD0_DAT0	I/O	MFP3	SD/SDIO0 数据线位0
				SPI1_SS	I/O	MFP5	SPI1从机选择管脚.
				UART1_RXD	I	MFP6	UART1数据接收输入管脚
				UART5_nCTS	I	MFP7	UART5 禁止发送输入管脚
				SC0_PWR	O	MFP9	智能卡0电源脚
				I^2S0_DO	O	MFP10	I^2S0 数据输出脚
				EPWM0_CH3	I/O	MFP11	EPWM0 通道3输出/捕获输入脚
				I^2C1_SDA	I/O	MFP12	I^2C1 数据输入/输出脚
				TM3	I/O	MFP14	Timer3 事件计数输入/反转输出脚
				INT3	I	MFP15	外部中断3输入脚
			5	PC.12	I/O	MFP0	通用数字I/O脚

32 管脚	48 管脚	64 管脚	128 管脚	管脚名称	类型	MFP	描述
				EBI_ADR4	O	MFP2	EBI 地址总线位 4.
				UART0_TXD	O	MFP3	UART0 数据发送输出脚.
				I^{2}C0_SCL	I/O	MFP4	I2C0 时钟管脚
				UART6_TXD	O	MFP5	UART6 数据发送输出脚.
				SC0_nCD	I	MFP9	智能卡 0 卡检测管脚.
				ECAP1_IC2	I	MFP11	增强型捕获计数器 1 输入 2 管脚.
				EPWM1_CH0	I/O	MFP12	EPWM1 通道0 输出/捕获输入
				ACMP0_O	O	MFP14	模拟比较器 0输出脚
			6	PC.11	I/O	MFP0	通用数字I/O脚
				EBI_ADR5	O	MFP2	EBI 地址总线位 5.
				UART0_RXD	I	MFP3	UART0 数据接收输入脚
				I^{2}C0_SDA	I/O	MFP4	I2C0 数据输入/输出管脚
				UART6_RXD	I	MFP5	UART6 数据接收输入脚
				ECAP1_IC1	I	MFP11	增强型捕获计数器 1 输入 1 管脚.
				EPWM1_CH1	I/O	MFP12	EPWM1 通道1 输出/捕获输入
				ACMP1_O	O	MFP14	模拟比较器1输出脚
			7	PC.10	I/O	MFP0	通用数字I/O脚
				EBI_ADR6	O	MFP2	EBI 地址总线位 6.
				UART6_nRTS	O	MFP5	UART6 请求发送输出脚
				UART3_TXD	O	MFP7	UART3 数据发送输出脚.
				CAN1_TXD	O	MFP9	CAN1总线发送输出
				ECAP1_IC0	I	MFP11	增强型捕获计数器 1 输入脚0
				EPWM1_CH2	I/O	MFP12	EPWM1 通道2 输出/捕获输入
				EADC1_ST	I	MFP14	EADC1 外部触发输入
			8	PC.9	I/O	MFP0	通用数字I/O脚
				EBI_ADR7	O	MFP2	EBI 地址总线位 7.
				UART6_nCTS	I	MFP5	UART6 禁止发送输入管脚
				UART3_RXD	I	MFP7	UART3 数据接收输入脚
				CAN1_RXD	I	MFP9	CAN1 总线接收输入
				EPWM1_CH3	I/O	MFP12	EPWM1 通道3输出/捕获输入脚

32管脚	48管脚	64管脚	128管脚	管脚名称	类型	MFP	描述
				EADC1_ST	I	MFP14	EADC1 外部触发输入
5	5	6	9	PB.1	I/O	MFP0	通用数字I/O脚
				EADC0_CH1	A	MFP1	EADC0 通道1 模拟输入
				EADC1_CH9	A	MFP1	EADC1 通道9 模拟输入
				EBI_ADR8	O	MFP2	EBI 地址总线位 8.
				SD0_CLK	O	MFP3	SD/SDIO0 时钟输出脚
				SPI1_I2SMCLK	I/O	MFP5	SPI1 I^2S 主机时钟输出脚
				UART2_TXD	O	MFP7	UART2 数据发送输出脚.
				I^2C1_SCL	I/O	MFP9	I^2C1 时钟脚
				I^2S0_LRCK	O	MFP10	I^2S0左右通道时钟输出脚.
				EPWM0_CH4	I/O	MFP11	EPWM0 通道4输出/捕获输入脚
				EPWM1_CH4	I/O	MFP12	EPWM1 通道4输出/捕获输入脚
				EPWM0_BRAKE0	I	MFP13	EPWM0 刹车0 输入脚
6	6	7	10	PB.0	I/O	MFP0	通用数字I/O脚
				EADC0_CH0	A	MFP1	EADC0 通道0 模拟输入
				EADC1_CH8	A	MFP1	EADC1 通道8 模拟输入
				EBI_ADR9	O	MFP2	EBI 地址总线位 9.
				SD0_CMD	I/O	MFP3	SD/SDIO0 命令/响应脚
				SPI2_I2SMCLK	I/O	MFP4	SPI2 I^2S 主机时钟输出脚
				UART2_RXD	I	MFP7	UART2 数据接收输入脚
				SPI0_I2SMCLK	I/O	MFP8	SPI0 I^2S 主机时钟输出脚
				I^2C1_SDA	I/O	MFP9	I^2C1 数据输入/输出脚
				EPWM0_CH5	I/O	MFP11	EPWM0 通道5 输出/捕获输入
				EPWM1_CH5	I/O	MFP12	EPWM1 通道5输出/捕获输入
				EPWM0_BRAKE1	I	MFP13	EPWM0 刹车1 输入脚
			11	V_{SS}	P	MFP0	数字电路地管脚
			12	V_{DD}	P	MFP0	I/O和内部PLL及数字电路的LDO电源
	7	8	13	PA.11	I/O	MFP0	通用数字I/O脚
				EADC1_CH7	A	MFP1	EADC1 通道7 模拟输入
				ACMP0_P0	A	MFP1	模拟比较器 0 正极输入脚0

32 管脚	48 管脚	64 管脚	128 管脚	管脚名称	类型	MFP	描述
				EBI_nRD	O	MFP2	EBI 读使能输出脚
				SPI2_SS	I/O	MFP4	SPI2从机选择管脚.
				I^{2}C2_SCL	I/O	MFP7	I^{2}C2 时钟脚
				UART6_TXD	O	MFP8	UART6 数据发送输出脚.
				BPWM0_CH0	I/O	MFP9	BPWM0 通道0 输出/捕获输入
				EPWM0_SYNC_OUT	O	MFP10	EPWM0计数器同步触发输出脚
				TM0_EXT	I/O	MFP13	Timer0 外部捕获输入/反转输出脚
	8	9	14	PA.10	I/O	MFP0	通用数字I/O脚
				EADC1_CH6	A	MFP1	EADC1 通道6 模拟输入
				ACMP1_P0	A	MFP1	模拟比较器 1 正极输入脚0
				EBI_nWR	O	MFP2	EBI 写使能输出脚
				SPI2_CLK	I/O	MFP4	SPI2 串行时钟管脚
				I^{2}C2_SDA	I/O	MFP7	I^{2}C2 数据输入/输出脚
				UART6_RXD	I	MFP8	UART6 数据接收输入脚
				BPWM0_CH1	I/O	MFP9	BPWM0 通道1 输出/捕获输入
				QEI1_INDEX	I	MFP10	正交编码器1 索引输入
				ECAP0_IC0	I	MFP11	增强型捕获计数器 0 输入脚0
				TM1_EXT	I/O	MFP13	Timer1 外部捕获输入/反转输出脚
				DAC0_ST	I	MFP14	DAC0 外部触发输入
				SWDH_CLK	O	MFP15	串行调试总线主时钟输出
	9	10	15	PA.9	I/O	MFP0	通用数字I/O脚
				EADC1_CH5	A	MFP1	EADC1 通道5 模拟输入
				EBI_MCLK	O	MFP2	EBI 外部时钟输出脚.
				SPI2_MISO	I/O	MFP4	SPI2 MISO(主机输入，从机输出)管脚
				UART1_TXD	O	MFP7	UART1 数据发送输出脚
				UART7_TXD	O	MFP8	UART7 数据发送输出脚.
				BPWM0_CH2	I/O	MFP9	BPWM0 通道2 输出/捕获输入
				QEI1_A	I	MFP10	正交编码器1 相位A输入
				ECAP0_IC1	I	MFP11	增强型捕获计数器 0 输入 1 管脚.
				TM2_EXT	I/O	MFP13	Timer2 外部捕获输入/反转输出脚

32管脚	48管脚	64管脚	128管脚	管脚名称	类型	MFP	描述
				SWDH_DAT	I/O	MFP15	串行调试总线主机数据输入/输出管脚
	10	11	16	PA.8	I/O	MFP0	通用数字I/O脚
				EADC1_CH4	A	MFP1	EADC1 通道4 模拟输入
				EBI_ALE	O	MFP2	EBI地址锁存使能输出脚
				SPI2_MOSI	I/O	MFP4	SPI2 MOSI (主机输出，从机输入) 管脚
				UART1_RXD	I	MFP7	UART1数据接收输入管脚
				UART7_RXD	I	MFP8	UART7 数据接收输入脚
				BPWM0_CH3	I/O	MFP9	BPWM0 通道3输出/捕获输入脚
				QEI1_B	I	MFP10	正交编码器1 相位B 输入
				ECAP0_IC2	I	MFP11	增强型捕获计数器 0 输入 2 管脚.
				TM3_EXT	I/O	MFP13	Timer3 外部捕获输入/反转输出脚
				INT4	I	MFP15	外部中断4输入管脚
			17	PC.13	I/O	MFP0	通用数字I/O脚
				EADC1_CH3	A	MFP1	EADC1 通道3 模拟输入
				EBI_ADR10	O	MFP2	EBI 地址总线位 10.
				SPI2_I2SMCLK	I/O	MFP4	SPI2 I^2S 主机时钟输出脚
				CAN1_TXD	O	MFP5	CAN1总线发送输出
				UART2_TXD	O	MFP7	UART2 数据发送输出脚.
				BPWM0_CH4	I/O	MFP9	BPWM0 通道4输出/捕获输入脚
				CLKO	O	MFP13	时钟输出
				EADC0_ST	I	MFP14	EADC0 外部触发输入
			18	PD.12	I/O	MFP0	通用数字I/O脚
				EADC1_CH2	A	MFP1	EADC1 通道2 模拟输入
				EBI_nCS0	O	MFP2	EBI片选0输出脚
				CAN1_RXD	I	MFP5	CAN1 总线接收输入
				UART2_RXD	I	MFP7	UART2 数据接收输入脚
				BPWM0_CH5	I/O	MFP9	BPWM0 通道5 输出/捕获输入
				QEI0_INDEX	I	MFP10	正交编码器0 索引输入
				CLKO	O	MFP13	时钟输出
				EADC0_ST	I	MFP14	EADC0 外部触发输入

32 管脚	48 管脚	64 管脚	128 管脚	管脚名称	类型	MFP	描述
				INT5	I	MFP15	外部中断5 输入脚
			19	PD.11	I/O	MFP0	通用数字I/O脚
				EADC1_CH1	A	MFP1	EADC1 通道1 模拟输入
				EBI_nCS1	O	MFP2	EBI片选1输出管脚
				UART1_TXD	O	MFP3	UART1 数据发送输出脚
				CAN0_TXD	O	MFP4	CAN0总线发送输出
				QEI0_A	I	MFP10	正交编码器0 相位A输入
				INT6	I	MFP15	外部中断6 输入脚
			20	PD.10	I/O	MFP0	通用数字I/O脚
				EADC1_CH0	A	MFP1	EADC1 通道0 模拟输入
				EBI_nCS2	O	MFP2	EBI片选2输出脚
				UART1_RXD	I	MFP3	UART1数据接收输入管脚
				CAN0_RXD	I	MFP4	CAN0 总线接收输入
				QEI0_B	I	MFP10	正交编码器0 相位B 输入
				INT7	I	MFP15	外部中断7 输入脚
			21	PG.2	I/O	MFP0	通用数字I/O脚
				EBI_ADR11	O	MFP2	EBI 地址总线位 11.
				SPI2_SS	I/O	MFP3	SPI2从机选择管脚.
				I^2C0_SMBAL	O	MFP4	I^2C0 SMBus SMBALTER 管脚
				I^2C1_SCL	I/O	MFP5	I^2C1 时钟脚
				CAP_DATA7	I	MFP7	图像数据输入总线位7.
				TM0	I/O	MFP13	Timer0 事件计数输入/反转输出管脚
			22	PG.3	I/O	MFP0	通用数字I/O脚
				EBI_ADR12	O	MFP2	EBI 地址总线位 12.
				SPI2_CLK	I/O	MFP3	SPI2 串行时钟管脚
				I^2C0_SMBSUS	O	MFP4	I^2C0 SMBus SMBSUS 管脚 (PMBus 控制管脚)
				I^2C1_SDA	I/O	MFP5	I^2C1 数据输入/输出脚
				CAP_DATA6	I	MFP7	图像数据输入总线位6.
				TM1	I/O	MFP13	Timer1 事件计数输入/反转输出脚
			23	PG.4	I/O	MFP0	通用数字I/O脚

32 管脚	48 管脚	64 管脚	128 管脚	管脚名称	类型	MFP	描述
				EBI_ADR13	O	MFP2	EBI 地址总线位 13.
				SPI2_MISO	I/O	MFP3	SPI2 MISO(主机输入，从机输出)管脚
				CAP_DATA5	I	MFP7	图像数据输入总线位5.
				TM2	I/O	MFP13	Timer2 事件计数输入/反转输出脚
			24	PF.11	I/O	MFP0	通用数字I/O脚
				EBI_ADR14	O	MFP2	EBI 地址总线位 14.
				SPI2_MOSI	I/O	MFP3	SPI2 MOSI (主机输出，从机输入) 管脚
				UART5_TXD	O	MFP6	UART5数据发送输出脚
				CAP_DATA4	I	MFP7	图像数据输入总线位4.
				TM3	I/O	MFP13	Timer3 事件计数输入/反转输出脚
			25	PF.10	I/O	MFP0	通用数字I/O脚
				EBI_ADR15	O	MFP2	EBI 地址总线位 15.
				SC0_nCD	I	MFP3	智能卡 0 卡检测管脚.
				I^2S0_BCLK	O	MFP4	I2S0位时钟输出管脚
				SPI0_I2SMCLK	I/O	MFP5	SPI0 I^2S 主机时钟输出脚
				UART5_RXD	I	MFP6	UART5 数据接收输入管脚
				CAP_DATA3	I	MFP7	图像数据输入总线位3.
			26	PF.9	I/O	MFP0	通用数字I/O脚
				EBI_ADR16	O	MFP2	EBI 地址总线位 16.
				SC0_PWR	O	MFP3	智能卡0电源脚
				I^2S0_MCLK	O	MFP4	I2S0 主时钟输出脚
				SPI0_SS	I/O	MFP5	SPI0从机选择管脚.
				UART5_nRTS	O	MFP6	UART5 请求发送输出脚
				CAP_DATA2	I	MFP7	图像数据输入总线位2.
				CAN1_TXD	O	MFP8	CAN1总线发送输出
			27	PF.8	I/O	MFP0	通用数字I/O脚
				EBI_ADR17	O	MFP2	EBI 地址总线位 17.
				SC0_RST	O	MFP3	智能卡0复位脚
				I^2S0_DI	I	MFP4	I2S0 数据输入脚
				SPI0_CLK	I/O	MFP5	SPI0 串行时钟脚

32 管脚	48 管脚	64 管脚	128 管脚	管脚名称	类型	MFP	描述
				UART5_nCTS	I	MFP6	UART5 禁止发送输入管脚
				CAP_DATA1	I	MFP7	图像数据输入总线位1.
				CAN1_RXD	I	MFP8	CAN1 总线接收输入
				TAMPER2	I/O	MFP10	TAMPER循环检测管脚 2.
			28	PF.7	I/O	MFP0	通用数字I/O脚
				EBI_ADR18	O	MFP2	EBI 地址总线位 18.
				SC0_DAT	I/O	MFP3	智能卡0数据输入脚
				I^2S0_DO	O	MFP4	I^2S0 数据输出脚
				SPI0_MISO	I/O	MFP5	SPI0 MISO（主机输入，从机输出）脚.
				UART4_TXD	O	MFP6	UART4 数据发送输出脚.
				CAP_DATA0	I	MFP7	图像数据输入总线位0.
				CAN2_TXD	O	MFP8	CAN2总线发送输出
				TAMPER1	I/O	MFP10	TAMPER循环检测管脚 1.
		12	29	PF.6	I/O	MFP0	通用数字I/O脚
				EBI_ADR19	O	MFP2	EBI 地址总线位 19.
				SC0_CLK	O	MFP3	智能卡0时钟管脚
				I^2S0_LRCK	O	MFP4	I^2S0左右通道时钟输出脚.
				SPI0_MOSI	I/O	MFP5	SPI0 MOSI (主机输出，从机输入) 管脚.
				UART4_RXD	I	MFP6	UART4 数据接收输入脚
				EBI_nCS0	O	MFP7	EBI片选0输出脚
				CAN2_RXD	I	MFP8	CAN2 总线接收输入
				TAMPER0	I/O	MFP10	TAMPER循环检测管脚 0.
		13	30	V_{BAT}	P	MFP0	RTC电池供电
7	11	14	31	PF.5	I/O	MFP0	通用数字I/O脚
				UART2_RXD	I	MFP2	UART2 数据接收输入脚
				UART2_nCTS	I	MFP4	UART2 禁止发送输入管脚
				EPWM0_CH0	I/O	MFP7	EPWM0 通道0输出/捕获输入
				BPWM0_CH4	I/O	MFP8	BPWM0 通道4输出/捕获输入脚
				EPWM0_SYNC_OUT	O	MFP9	EPWM0计数器同步触发输出脚
				X32_IN	I	MFP10	外部 32.768 kHz 晶振输入脚

32 管脚	48 管脚	64 管脚	128 管脚	管脚名称	类型	MFP	描述
				EADC0_ST	I	MFP11	EADC0 外部触发输入
8	12	15	32	PF.4	I/O	MFP0	通用数字I/O脚
				UART2_TXD	O	MFP2	UART2 数据发送输出脚.
				UART2_nRTS	O	MFP4	UART2 请求发送输出脚
				EPWM0_CH1	I/O	MFP7	EPWM0 通道1输出/捕获输入脚
				BPWM0_CH5	I/O	MFP8	BPWM0 通道5 输出/捕获输入
				X32_OUT	O	MFP10	外部 32.768 kHz 晶振输出脚
				EADC1_ST	I	MFP11	EADC1 外部触发输入
			33	PH.4	I/O	MFP0	通用数字I/O脚
				EBI_ADR3	O	MFP2	EBI 地址总线位3
				SPI1_MISO	I/O	MFP3	SPI1 MISO（主机输入，从机输出）脚.
				UART7_nRTS	O	MFP4	UART7 请求发送输出脚
				UART6_TXD	O	MFP5	UART6 数据发送输出脚.
			34	PH.5	I/O	MFP0	通用数字I/O脚
				EBI_ADR2	O	MFP2	EBI 地址总线位2
				SPI1_MOSI	I/O	MFP3	SPI1 MOSI (主机输出，从机输入) 管脚.
				UART7_nCTS	I	MFP4	UART7 禁止发送输入管脚
				UART6_RXD	I	MFP5	UART6 数据接收输入脚
			35	PH.6	I/O	MFP0	通用数字I/O脚
				EBI_ADR1	O	MFP2	EBI 地址总线位1
				SPI1_CLK	I/O	MFP3	SPI1 串行时钟脚
				UART7_TXD	O	MFP4	UART7 数据发送输出脚.
			36	PH.7	I/O	MFP0	通用数字I/O脚
				EBI_ADR0	O	MFP2	EBI地址总线位0
				SPI1_SS	I/O	MFP3	SPI1从机选择管脚.
				UART7_RXD	I	MFP4	UART7 数据接收输入脚
9	13	16	37	PF.3	I/O	MFP0	通用数字I/O脚
				EBI_nCS0	O	MFP2	EBI片选0输出脚
				UART0_TXD	O	MFP3	UART0 数据发送输出脚.
				I^2C0_SCL	I/O	MFP4	I2C0 时钟管脚

32管脚	48管脚	64管脚	128管脚	管脚名称	类型	MFP	描述
				XT1_IN	I	MFP10	外部 4~24 MHz (高速) 晶振输入脚
				BPWM1_CH0	I/O	MFP11	BPWM1 通道0 输出/捕获输入
10	14	17	38	PF.2	I/O	MFP0	通用数字I/O脚
				EBI_nCS1	O	MFP2	EBI片选1输出管脚
				UART0_RXD	I	MFP3	UART0 数据接收输入脚
				I^2C0_SDA	I/O	MFP4	I2C0 数据输入/输出管脚
				QSPI0_CLK	I/O	MFP5	Quad SPI0 串行时钟脚
				XT1_OUT	O	MFP10	外部 4~24 MHz (高速) 晶振输出脚
				BPWM1_CH1	I/O	MFP11	BPWM1 通道1 输出/捕获输入
			39	V_{SS}	P	MFP0	数字电路地管脚
			40	V_{DD}	P	MFP0	I/O和内部PLL及数字电路的LDO电源
			41	PE.8	I/O	MFP0	通用数字I/O脚
				EBI_ADR10	O	MFP2	EBI 地址总线位 10.
				I^2S0_BCLK	O	MFP4	I2S0位时钟输出管脚
				SPI2_CLK	I/O	MFP5	SPI2 串行时钟管脚
				UART2_TXD	O	MFP7	UART2 数据发送输出脚.
				EPWM0_CH0	I/O	MFP10	EPWM0 通道0输出/捕获输入
				EPWM0_BRAKE0	I	MFP11	EPWM0 刹车0 输入脚
				ECAP0_IC0	I	MFP12	增强型捕获计数器 0 输入脚0
				TRACE_DATA3	O	MFP14	ETM跟踪数据3 输出管脚
			42	PE.9	I/O	MFP0	通用数字I/O脚
				EBI_ADR11	O	MFP2	EBI 地址总线位 11.
				I^2S0_MCLK	O	MFP4	I2S0 主时钟输出脚
				SPI2_MISO	I/O	MFP5	SPI2 MISO(主机输入，从机输出)管脚
				UART2_RXD	I	MFP7	UART2 数据接收输入脚
				EPWM0_CH1	I/O	MFP10	EPWM0 通道1输出/捕获输入脚
				EPWM0_BRAKE1	I	MFP11	EPWM0 刹车1 输入脚
				ECAP0_IC1	I	MFP12	增强型捕获计数器 0 输入 1 管脚.
				TRACE_DATA2	O	MFP14	ETM跟踪数据2 输出管脚
			43	PE.10	I/O	MFP0	通用数字I/O脚

32 管脚	48 管脚	64 管脚	128 管脚	管脚名称	类型	MFP	描述
				EBI_ADR12	O	MFP2	EBI 地址总线位 12.
				I²S0_DI	I	MFP4	I2S0 数据输入脚
				SPI2_MOSI	I/O	MFP5	SPI2 MOSI (主机输出，从机输入) 管脚
				UART3_TXD	O	MFP7	UART3 数据发送输出脚.
				EPWM0_CH2	I/O	MFP10	EPWM0 通道2输出/捕获输入脚
				EPWM1_BRAKE0	I	MFP11	EPWM1 刹车0 输入脚
				ECAP0_IC2	I	MFP12	增强型捕获计数器 0 输入 2 管脚.
				TRACE_DATA1	O	MFP14	ETM跟踪数据1 输出管脚
			44	PE.11	I/O	MFP0	通用数字I/O脚
				EBI_ADR13	O	MFP2	EBI 地址总线位 13.
				I²S0_DO	O	MFP4	I²S0 数据输出脚
				SPI2_SS	I/O	MFP5	SPI2从机选择管脚.
				UART3_RXD	I	MFP7	UART3 数据接收输入脚
				UART1_nCTS	I	MFP8	UART1 禁止发送输入管脚
				EPWM0_CH3	I/O	MFP10	EPWM0 通道3输出/捕获输入脚
				EPWM1_BRAKE1	I	MFP11	EPWM1 刹车1输入管脚
				ECAP1_IC2	I	MFP13	增强型捕获计数器 1 输入 2 管脚.
				TRACE_DATA0	O	MFP14	ETM跟踪数据0 输出管脚
			45	PE.12	I/O	MFP0	通用数字I/O脚
				EBI_ADR14	O	MFP2	EBI 地址总线位 14.
				I²S0_LRCK	O	MFP4	I²S0左右通道时钟输出脚.
				SPI2_I2SMCLK	I/O	MFP5	SPI2 I²S 主机时钟输出脚
				UART1_nRTS	O	MFP8	UART1 请求发送输出脚
				EPWM0_CH4	I/O	MFP10	EPWM0 通道4输出/捕获输入脚
				ECAP1_IC1	I	MFP13	增强型捕获计数器 1 输入 1 管脚.
				TRACE_CLK	O	MFP14	ETM跟踪时钟输出脚
			46	PE.13	I/O	MFP0	通用数字I/O脚
				EBI_ADR15	O	MFP2	EBI 地址总线位 15.
				I²C0_SCL	I/O	MFP4	I2C0 时钟管脚
				UART4_nRTS	O	MFP5	UART4 请求发送输出脚

32 管脚	48 管脚	64 管脚	128 管脚	管脚名称	类型	MFP	描述
				UART1_TXD	O	MFP8	UART1 数据发送输出脚
				EPWM0_CH5	I/O	MFP10	EPWM0 通道5 输出/捕获输入
				EPWM1_CH0	I/O	MFP11	EPWM1 通道0 输出/捕获输入
				BPWM1_CH5	I/O	MFP12	EPWM1 通道5输出/捕获输入
				ECAP1_IC0	I	MFP13	增强型捕获计数器 1 输入脚0
			47	PC.8	I/O	MFP0	通用数字I/O脚
				EBI_ADR16	O	MFP2	EBI 地址总线位 16.
				I^2C0_SDA	I/O	MFP4	I2C0 数据输入/输出管脚
				UART4_nCTS	I	MFP5	UART4 禁止发送输入管脚
				UART1_RXD	I	MFP8	UART1数据接收输入管脚
				EPWM1_CH1	I/O	MFP11	EPWM1 通道1 输出/捕获输入
				BPWM1_CH4	I/O	MFP12	BPWM1 通道4输出/捕获输入脚
		18	48	PC.7	I/O	MFP0	通用数字I/O脚
				EBI_AD9	I/O	MFP2	EBI地址/数据总线位9.
				SPI1_MISO	I/O	MFP4	SPI1 MISO（主机输入，从机输出）脚.
				UART4_TXD	O	MFP5	UART4 数据发送输出脚.
				UART0_nCTS	I	MFP7	UART0 禁止发送输入管脚
				I^2C1_SMBAL	O	MFP8	I^2C1 SMBus SMBALTER 管脚
				UART6_TXD	O	MFP9	UART6 数据发送输出脚.
				EPWM1_CH2	I/O	MFP11	EPWM1 通道2 输出/捕获输入
				BPWM1_CH0	I/O	MFP12	BPWM1 通道0 输出/捕获输入
				TM0	I/O	MFP14	Timer0 事件计数输入/反转输出管脚
				INT3	I	MFP15	外部中断3输入脚
		19	49	PC.6	I/O	MFP0	通用数字I/O脚
				EBI_AD8	I/O	MFP2	EBI地址/数据总线位8.
				SPI1_MOSI	I/O	MFP4	SPI1 MOSI (主机输出，从机输入) 管脚.
				UART4_RXD	I	MFP5	UART4 数据接收输入脚
				UART0_nRTS	O	MFP7	UART0 请求发送输出脚
				I^2C1_SMBSUS	O	MFP8	I^2C1 SMBus SMBSUS 管脚 (PMBus 控制管脚)
				UART6_RXD	I	MFP9	UART6 数据接收输入脚

32管脚	48管脚	64管脚	128管脚	管脚名称	类型	MFP	描述
				EPWM1_CH3	I/O	MFP11	EPWM1 通道3输出/捕获输入脚
				BPWM1_CH1	I/O	MFP12	BPWM1 通道1 输出/捕获输入
				TM1	I/O	MFP14	Timer1 事件计数输入/反转输出脚
				INT2	I	MFP15	外部中断2输入脚
	15	20	50	PA.7	I/O	MFP0	通用数字I/O脚
				EBI_AD7	I/O	MFP2	EBI地址/数据总线位7.
				SPI1_CLK	I/O	MFP4	SPI1 串行时钟脚
				UART0_TXD	O	MFP7	UART0 数据发送输出脚.
				I^2C1_SCL	I/O	MFP8	I^2C1 时钟脚
				QSPI1_MISO1	I/O	MFP9	Quad SPI1 MISO1（主机输入，从机输出）脚.
				EPWM1_CH4	I/O	MFP11	EPWM1 通道4输出/捕获输入脚
				BPWM1_CH2	I/O	MFP12	BPWM1 通道2 输出/捕获输入
				ACMP0_WLAT	I	MFP13	模拟比较器 0窗口锁存输入管脚
				TM2	I/O	MFP14	Timer2 事件计数输入/反转输出脚
				INT1	I	MFP15	外部中断1输入脚
	16	21	51	PA.6	I/O	MFP0	通用数字I/O脚
				EBI_AD6	I/O	MFP2	EBI地址/数据总线位6.
				SPI1_SS	I/O	MFP4	SPI1从机选择管脚.
				UART0_RXD	I	MFP7	UART0 数据接收输入脚
				I^2C1_SDA	I/O	MFP8	I^2C1 数据输入/输出脚
				QSPI1_MOSI1	I/O	MFP9	Quad SPI1 MOSI1 (主机输出，从机输入) 管脚.
				EPWM1_CH5	I/O	MFP11	EPWM1 通道5输出/捕获输入
				BPWM1_CH3	I/O	MFP12	BPWM1 通道3输出/捕获输入脚
				ACMP1_WLAT	I	MFP13	模拟比较器 1窗口锁存输入管脚
				TM3	I/O	MFP14	Timer3 事件计数输入/反转输出脚
				INT0	I	MFP15	外部中断0输入管脚
		22	52	V_{SS}	P	MFP0	数字电路地管脚
		23	53	V_{DD}	P	MFP0	I/O和内部PLL及数字电路的LDO电源
		24	54	LDO_CAP	A	MFP0	LDO输出脚
	17	25	55	PA.5	I/O	MFP0	通用数字I/O脚

32 管脚	48 管脚	64 管脚	128 管脚	管脚名称	类型	MFP	描述
				QSPI0_MISO1	I/O	MFP3	Quad SPI0 MISO1（主机输入，从机输出）脚.
				SPI1_I2SMCLK	I/O	MFP4	SPI1 I^2S 主机时钟输出脚
				UART0_nCTS	I	MFP7	UART0 禁止发送输入管脚
				UART5_TXD	O	MFP8	UART5数据发送输出脚
				I^2C0_SCL	I/O	MFP9	I2C0 时钟管脚
				CAN0_TXD	O	MFP10	CAN0总线发送输出
				UART0_TXD	O	MFP11	UART0 数据发送输出脚.
				BPWM0_CH5	I/O	MFP12	BPWM0 通道5 输出/捕获输入
				EPWM0_CH0	I/O	MFP13	EPWM0 通道0输出/捕获输入
				QEI0_INDEX	I	MFP14	正交编码器0 索引输入
	18	26	56	PA.4	I/O	MFP0	通用数字I/O脚
				QSPI0_MOSI1	I/O	MFP3	Quad SPI0 MOSI1 (主机输出，从机输入) 管脚.
				SPI0_I2SMCLK	I/O	MFP4	SPI0 I^2S 主机时钟输出脚
				SC0_nCD	I	MFP6	智能卡 0 卡检测管脚.
				UART0_nRTS	O	MFP7	UART0 请求发送输出脚
				UART5_RXD	I	MFP8	UART5 数据接收输入管脚
				I^2C0_SDA	I/O	MFP9	I2C0 数据输入/输出管脚
				CAN0_RXD	I	MFP10	CAN0 总线接收输入
				UART0_RXD	I	MFP11	UART0 数据接收输入脚
				BPWM0_CH4	I/O	MFP12	BPWM0 通道4输出/捕获输入脚
				EPWM0_CH1	I/O	MFP13	EPWM0 通道1输出/捕获输入脚
				QEI0_A	I	MFP14	正交编码器0 相位A输入
11	19	27	57	PA.3	I/O	MFP0	通用数字I/O脚
				QSPI0_SS	I/O	MFP3	Quad SPI0从机选择管脚.
				SPI0_SS	I/O	MFP4	SPI0从机选择管脚.
				SC0_PWR	O	MFP6	智能卡0电源脚
				UART4_TXD	O	MFP7	UART4 数据发送输出脚.
				UART1_TXD	O	MFP8	UART1 数据发送输出脚
				I^2C1_SCL	I/O	MFP9	I^2C1 时钟脚
				I^2C0_SMBAL	O	MFP10	I^2C0 SMBus SMBALTER 管脚

32 管脚	48 管脚	64 管脚	128 管脚	管脚名称	类型	MFP	描述
				BPWM0_CH3	I/O	MFP12	BPWM0 通道3输出/捕获输入脚
				EPWM0_CH2	I/O	MFP13	EPWM0 通道2输出/捕获输入脚
				QEI0_B	I	MFP14	正交编码器0 相位B 输入
				EPWM1_BRAKE1	I	MFP15	EPWM1 刹车1输入管脚
12	20	28	58	PA.2	I/O	MFP0	通用数字I/O脚
				QSPI0_CLK	I/O	MFP3	Quad SPI0 串行时钟脚
				SPI0_CLK	I/O	MFP4	SPI0 串行时钟脚
				SC0_RST	O	MFP6	智能卡0复位脚
				UART4_RXD	I	MFP7	UART4 数据接收输入脚
				UART1_RXD	I	MFP8	UART1数据接收输入管脚
				I^2C1_SDA	I/O	MFP9	I^2C1 数据输入/输出脚
				I^2C0_SMBSUS	O	MFP10	I^2C0 SMBus SMBSUS 管脚 (PMBus 控制管脚)
				BPWM0_CH2	I/O	MFP12	BPWM0 通道2 输出/捕获输入
				EPWM0_CH3	I/O	MFP13	EPWM0 通道3输出/捕获输入脚
13	21	29	59	PA.1	I/O	MFP0	通用数字I/O脚
				QSPI0_MISO0	I/O	MFP3	Quad SPI0 MISO0（主机输入，从机输出）脚.
				SPI0_MISO	I/O	MFP4	SPI0 MISO（主机输入，从机输出）脚.
				SC0_DAT	I/O	MFP6	智能卡0数据输入脚
				UART0_TXD	O	MFP7	UART0 数据发送输出脚.
				UART1_nCTS	I	MFP8	UART1 禁止发送输入管脚
				I^2C2_SCL	I/O	MFP9	I^2C2 时钟脚
				CAP_DATA7	I	MFP10	图像数据输入总线位7.
				BPWM0_CH1	I/O	MFP12	BPWM0 通道1 输出/捕获输入
				EPWM0_CH4	I/O	MFP13	EPWM0 通道4输出/捕获输入脚
14	22	30	60	PA.0	I/O	MFP0	通用数字I/O脚
				QSPI0_MOSI0	I/O	MFP3	Quad SPI0 MOSI0 (主机输出，从机输入) 管脚.
				SPI0_MOSI	I/O	MFP4	SPI0 MOSI (主机输出，从机输入) 管脚.
				SC0_CLK	O	MFP6	智能卡0时钟管脚
				UART0_RXD	I	MFP7	UART0 数据接收输入脚
				UART1_nRTS	O	MFP8	UART1 请求发送输出脚

32 管脚	48 管脚	64 管脚	128 管脚	管脚名称	类型	MFP	描述
				I^2C2_SDA	I/O	MFP9	I^2C2 数据输入/输出脚
				CAP_DATA6	I	MFP10	图像数据输入总线位6.
				BPWM0_CH0	I/O	MFP12	BPWM0 通道0 输出/捕获输入
				EPWM0_CH5	I/O	MFP13	EPWM0 通道5 输出/捕获输入
				DAC0_ST	I	MFP15	DAC0 外部触发输入
15	23	31	61	V_{DDIO}	P	MFP0	PA.0~PA.5的电源
			62	PE.14	I/O	MFP0	通用数字I/O脚
				EBI_AD8	I/O	MFP2	EBI地址/数据总线位8.
				UART2_TXD	O	MFP3	UART2 数据发送输出脚.
				CAN0_TXD	O	MFP4	CAN0总线发送输出
				UART6_TXD	O	MFP6	UART6 数据发送输出脚.
			63	PE.15	I/O	MFP0	通用数字I/O脚
				EBI_AD9	I/O	MFP2	EBI地址/数据总线位9.
				UART2_RXD	I	MFP3	UART2 数据接收输入脚
				CAN0_RXD	I	MFP4	CAN0 总线接收输入
				UART6_RXD	I	MFP6	UART6 数据接收输入脚
16	24	32	64	nRESET	I	MFP0	外部复位输入:低电平有效,有内部上拉，设置此管脚为低将复位初始状态
17	25	33	65	PF.0	I/O	MFP0	通用数字I/O脚
				UART1_TXD	O	MFP2	UART1 数据发送输出脚
				I^2C1_SCL	I/O	MFP3	I^2C1 时钟脚
				UART0_TXD	O	MFP4	UART0 数据发送输出脚.
				BPWM1_CH0	I/O	MFP12	BPWM1 通道0 输出/捕获输入
				ICE_DAT	I/O	MFP14	ADC 外部触发输入
18	26	34	66	PF.1	I/O	MFP0	通用数字I/O脚
				UART1_RXD	I	MFP2	UART1数据接收输入管脚
				I^2C1_SDA	I/O	MFP3	I^2C1 数据输入/输出脚
				UART0_RXD	I	MFP4	UART0 数据接收输入脚
				BPWM1_CH1	I/O	MFP12	BPWM1 通道1 输出/捕获输入
				ICE_CLK	I/O	MFP14	ADC 外部触发输入
			67	PD.9	I/O	MFP0	通用数字I/O脚

32 管脚	48 管脚	64 管脚	128 管脚	管脚名称	类型	MFP	描述
				EBI_AD7	I/O	MFP2	EBI地址/数据总线位7.
				I²C2_SCL	I/O	MFP3	I²C2 时钟脚
				UART2_nCTS	I	MFP4	UART2 禁止发送输入管脚
				UART7_TXD	O	MFP5	UART7 数据发送输出脚.
				CAN2_TXD	O	MFP6	CAN2总线发送输出
			68	PD.8	I/O	MFP0	通用数字I/O脚
				EBI_AD6	I/O	MFP2	EBI地址/数据总线位6.
				I²C2_SDA	I/O	MFP3	I²C2 数据输入/输出脚
				UART2_nRTS	O	MFP4	UART2 请求发送输出脚
				UART7_RXD	I	MFP5	UART7 数据接收输入脚
				CAN2_RXD	I	MFP6	CAN2 总线接收输入
	27	35	69	PC.5	I/O	MFP0	通用数字I/O脚
				EBI_AD5	I/O	MFP2	EBI地址/数据总线位5.
				QSPI0_MISO1	I/O	MFP4	Quad SPI0 MISO1（主机输入，从机输出）脚.
				UART2_TXD	O	MFP8	UART2 数据发送输出脚.
				I²C1_SCL	I/O	MFP9	I²C1 时钟脚
				CAN0_TXD	O	MFP10	CAN0总线发送输出
				UART4_TXD	O	MFP11	UART4 数据发送输出脚.
				EPWM1_CH0	I/O	MFP12	EPWM1 通道0 输出/捕获输入
				CAP_DATA5	I	MFP13	图像数据输入总线位5.
				QSPI1_SS	I/O	MFP14	Quad SPI1从机选择管脚.
	28	36	70	PC.4	I/O	MFP0	通用数字I/O脚
				EBI_AD4	I/O	MFP2	EBI地址/数据总线位4.
				QSPI0_MOSI1	I/O	MFP4	Quad SPI0 MOSI1 (主机输出，从机输入) 管脚.
				I²S0_BCLK	O	MFP6	I2S0位时钟输出管脚
				SPI1_I2SMCLK	I/O	MFP7	SPI1 I²S 主机时钟输出脚
				UART2_RXD	I	MFP8	UART2 数据接收输入脚
				I²C1_SDA	I/O	MFP9	I²C1 数据输入/输出脚
				CAN0_RXD	I	MFP10	CAN0 总线接收输入
				UART4_RXD	I	MFP11	UART4 数据接收输入脚

32 管脚	48 管脚	64 管脚	128 管脚	管脚名称	类型	MFP	描述
				EPWM1_CH1	I/O	MFP12	EPWM1 通道1 输出/捕获输入
				CAP_DATA4	I	MFP13	图像数据输入总线位4.
				QSPI1_CLK	I/O	MFP14	Quad SPI1 串行时钟脚
	29	37	71	PC.3	I/O	MFP0	通用数字I/O脚
				EBI_AD3	I/O	MFP2	EBI地址/数据总线位3.
				QSPI0_SS	I/O	MFP4	Quad SPI0从机选择管脚.
				I^2S0_MCLK	O	MFP6	I2S0 主时钟输出脚
				SPI1_MISO	I/O	MFP7	SPI1 MISO（主机输入，从机输出）脚.
				UART2_nRTS	O	MFP8	UART2 请求发送输出脚
				I^2C0_SMBAL	O	MFP9	I^2C0 SMBus SMBALTER 管脚
				CAN1_TXD	O	MFP10	CAN1总线发送输出
				UART3_TXD	O	MFP11	UART3 数据发送输出脚.
				EPWM1_CH2	I/O	MFP12	EPWM1 通道2 输出/捕获输入
				CAP_DATA3	I	MFP13	图像数据输入总线位3.
				QSPI1_MISO0	I/O	MFP14	Quad SPI1 MISO0（主机输入，从机输出）脚.
	30	38	72	PC.2	I/O	MFP0	通用数字I/O脚
				EBI_AD2	I/O	MFP2	EBI地址/数据总线位2.
				QSPI0_CLK	I/O	MFP4	Quad SPI0 串行时钟脚
				I^2S0_DI	I	MFP6	I2S0 数据输入脚
				SPI1_MOSI	I/O	MFP7	SPI1 MOSI (主机输出，从机输入) 管脚.
				UART2_nCTS	I	MFP8	UART2 禁止发送输入管脚
				I^2C0_SMBSUS	O	MFP9	I^2C0 SMBus SMBSUS 管脚 (PMBus 控制管脚)
				CAN1_RXD	I	MFP10	CAN1 总线接收输入
				UART3_RXD	I	MFP11	UART3 数据接收输入脚
				EPWM1_CH3	I/O	MFP12	EPWM1 通道3输出/捕获输入脚
				CAP_DATA2	I	MFP13	图像数据输入总线位2.
				QSPI1_MOSI0	I/O	MFP14	Quad SPI1 MOSI0 (主机输出，从机输入) 管脚.
19	31	39	73	PC.1	I/O	MFP0	通用数字I/O脚
				EBI_AD1	I/O	MFP2	EBI地址/数据总线位1.
				QSPI0_MISO0	I/O	MFP4	Quad SPI0 MISO0（主机输入，从机输出）脚.

32 管脚	48 管脚	64 管脚	128 管脚	管脚名称	类型	MFP	描述
				I^2S0_DO	O	MFP6	I^2S0 数据输出脚
				SPI1_CLK	I/O	MFP7	SPI1 串行时钟脚
				UART2_TXD	O	MFP8	UART2 数据发送输出脚.
				I^2C0_SCL	I/O	MFP9	I2C0 时钟管脚
				CAN2_TXD	O	MFP10	CAN2总线发送输出
				EPWM1_CH4	I/O	MFP12	EPWM1 通道4输出/捕获输入脚
				CAP_DATA1	I	MFP13	图像数据输入总线位1.
				ACMP0_O	O	MFP14	模拟比较器 0输出脚
				EADC0_ST	I	MFP15	EADC0 外部触发输入
20	32	40	74	PC.0	I/O	MFP0	通用数字I/O脚
				EBI_AD0	I/O	MFP2	EBI地址/数据总线位0.
				QSPI0_MOSI0	I/O	MFP4	Quad SPI0 MOSI0 (主机输出，从机输入) 管脚.
				I^2S0_LRCK	O	MFP6	I^2S0左右通道时钟输出脚.
				SPI1_SS	I/O	MFP7	SPI1从机选择管脚.
				UART2_RXD	I	MFP8	UART2 数据接收输入脚
				I^2C0_SDA	I/O	MFP9	I2C0 数据输入/输出管脚
				CAN2_RXD	I	MFP10	CAN2 总线接收输入
				EPWM1_CH5	I/O	MFP12	EPWM1 通道5输出/捕获输入
				CAP_DATA0	I	MFP13	图像数据输入总线位0.
				ACMP1_O	O	MFP14	模拟比较器1输出脚
				EADC1_ST	I	MFP15	EADC1 外部触发输入
			75	V_{SS}	P	MFP0	数字电路地管脚
			76	V_{DD}	P	MFP0	I/O和内部PLL及数字电路的LDO电源
			77	PG.9	I/O	MFP0	通用数字I/O脚
				EBI_AD0	I/O	MFP2	EBI地址/数据总线位0.
				QSPI1_MISO1	I/O	MFP5	Quad SPI1 MISO1（主机输入，从机输出）脚.
				CAP_PIXCLK	I	MFP7	图像采集接口像素时钟输入脚
				BPWM0_CH5	I/O	MFP12	BPWM0 通道5 输出/捕获输入
			78	PG.10	I/O	MFP0	通用数字I/O脚
				EBI_AD1	I/O	MFP2	EBI地址/数据总线位1.

32 管脚	48 管脚	64 管脚	128 管脚	管脚名称	类型	MFP	描述
				QSPI1_MOSI1	I/O	MFP5	Quad SPI1 MOSI1 (主机输出，从机输入) 管脚.
				CAP_SCLK	O	MFP7	图像采集接口传感器时钟管脚.
				BPWM0_CH4	I/O	MFP12	BPWM0 通道4输出/捕获输入脚
			79	PG.11	I/O	MFP0	通用数字I/O脚
				EBI_AD2	I/O	MFP2	EBI地址/数据总线位2.
				QSPI1_SS	I/O	MFP5	Quad SPI1从机选择管脚.
				UART7_TXD	O	MFP6	UART7 数据发送输出脚.
				CAP_SFIELD	I	MFP7	图像输入接口SFIELD 输入脚
				BPWM0_CH3	I/O	MFP12	BPWM0 通道3输出/捕获输入脚
			80	PG.12	I/O	MFP0	通用数字I/O脚
				EBI_AD3	I/O	MFP2	EBI地址/数据总线位3.
				QSPI1_CLK	I/O	MFP5	Quad SPI1 串行时钟脚
				UART7_RXD	I	MFP6	UART7 数据接收输入脚
				CAP_VSYNC	I	MFP7	图像采集接口vsync 输入脚
				BPWM0_CH2	I/O	MFP12	BPWM0 通道2 输出/捕获输入
			81	PG.13	I/O	MFP0	通用数字I/O脚
				EBI_AD4	I/O	MFP2	EBI地址/数据总线位4.
				QSPI1_MISO0	I/O	MFP5	Quad SPI1 MISO0（主机输入，从机输出）脚.
				UART6_TXD	O	MFP6	UART6 数据发送输出脚.
				CAP_HSYNC	I	MFP7	图像采集接口hsync 输入脚
				BPWM0_CH1	I/O	MFP12	BPWM0 通道1 输出/捕获输入
			82	PG.14	I/O	MFP0	通用数字I/O脚
				EBI_AD5	I/O	MFP2	EBI地址/数据总线位5.
				QSPI1_MOSI0	I/O	MFP5	Quad SPI1 MOSI0 (主机输出，从机输入) 管脚.
				UART6_RXD	I	MFP6	UART6 数据接收输入脚
				BPWM0_CH0	I/O	MFP12	BPWM0 通道0 输出/捕获输入
			83	PG.15	I/O	MFP0	通用数字I/O脚
				CLKO	O	MFP14	时钟输出
				EADC0_ST	I	MFP15	EADC0 外部触发输入
			84	PD.13	I/O	MFP0	通用数字I/O脚

32 管脚	48 管脚	64 管脚	128 管脚	管脚名称	类型	MFP	描述
				EBI_AD10	I/O	MFP2	EBI地址/数据总线位10.
				SD0_nCD	I	MFP3	SD/SDIO0 卡检测输入管脚
				SPI0_I2SMCLK	I/O	MFP4	SPI0 I^2S 主机时钟输出脚
				SPI1_I2SMCLK	I/O	MFP5	SPI1 I^2S 主机时钟输出脚
21	33	45	85	PA.12	I/O	MFP0	通用数字I/O脚
				I^2S0_BCLK	O	MFP2	I2S0位时钟输出管脚
				UART4_TXD	O	MFP3	UART4 数据发送输出脚.
				I^2C1_SCL	I/O	MFP4	I^2C1 时钟脚
				SPI2_SS	I/O	MFP5	SPI2从机选择管脚.
				CAN0_TXD	O	MFP6	CAN0总线发送输出
				BPWM1_CH2	I/O	MFP11	BPWM1 通道2 输出/捕获输入
				QEI1_INDEX	I	MFP12	正交编码器1 索引输入
				USB_VBUS	P	MFP14	USB主机或HUB的电源
22	34	46	86	PA.13	I/O	MFP0	通用数字I/O脚
				I^2S0_MCLK	O	MFP2	I2S0 主时钟输出脚
				UART4_RXD	I	MFP3	UART4 数据接收输入脚
				I^2C1_SDA	I/O	MFP4	I^2C1 数据输入/输出脚
				SPI2_CLK	I/O	MFP5	SPI2 串行时钟管脚
				CAN0_RXD	I	MFP6	CAN0 总线接收输入
				BPWM1_CH3	I/O	MFP11	BPWM1 通道3输出/捕获输入脚
				QEI1_A	I	MFP12	正交编码器1 相位A输入
				USB_D-	A	MFP14	USB差分信号D-.
23	35	47	87	PA.14	I/O	MFP0	通用数字I/O脚
				I^2S0_DI	I	MFP2	I2S0 数据输入脚
				UART0_TXD	O	MFP3	UART0 数据发送输出脚.
				SPI2_MISO	I/O	MFP5	SPI2 MISO(主机输入，从机输出)管脚
				I^2C2_SCL	I/O	MFP6	I^2C2 时钟脚
				BPWM1_CH4	I/O	MFP11	BPWM1 通道4输出/捕获输入脚
				QEI1_B	I	MFP12	正交编码器1 相位B 输入
				USB_D+	A	MFP14	USB差分信号D+.

32管脚	48管脚	64管脚	128管脚	管脚名称	类型	MFP	描述
24	36	48	88	PA.15	I/O	MFP0	通用数字I/O脚
				I^2S0_DO	O	MFP2	I^2S0 数据输出脚
				UART0_RXD	I	MFP3	UART0 数据接收输入脚
				SPI2_MOSI	I/O	MFP5	SPI2 MOSI (主机输出，从机输入) 管脚
				I^2C2_SDA	I/O	MFP6	I^2C2 数据输入/输出脚
				BPWM1_CH5	I/O	MFP11	EPWM1 通道5输出/捕获输入
				EPWM0_SYNC_IN	I	MFP12	EPWM0 计数器同步触发输入脚
				USB_OTG_ID	I	MFP14	USB_ 标识
			89	NC	-	MFP0	没用管脚，浮空
			90	NC	-	MFP0	没用管脚，浮空
			91	NC	-	MFP0	没用管脚，浮空
			92	NC	-	MFP0	没用管脚，浮空
			93	NC	-	MFP0	没用管脚，浮空
			94	NC	-	MFP0	没用管脚，浮空
			95	NC	-	MFP0	没用管脚，浮空
			96	NC	-	MFP0	没用管脚，浮空
			97	PE.7	I/O	MFP0	通用数字I/O脚
				SD0_CMD	I/O	MFP3	SD/SDIO0 命令/响应脚
				UART5_TXD	O	MFP8	UART5数据发送输出脚
				CAN1_TXD	O	MFP9	CAN1总线发送输出
				QEI1_INDEX	I	MFP11	正交编码器1 索引输入
				EPWM0_CH0	I/O	MFP12	EPWM0 通道0输出/捕获输入
				BPWM0_CH5	I/O	MFP13	BPWM0 通道5 输出/捕获输入
			98	PE.6	I/O	MFP0	通用数字I/O脚
				SD0_CLK	O	MFP3	SD/SDIO0 时钟输出脚
				SC0_nCD	I	MFP6	智能卡 0 卡检测管脚.
				UART5_RXD	I	MFP8	UART5 数据接收输入管脚
				CAN1_RXD	I	MFP9	CAN1 总线接收输入
				QEI1_A	I	MFP11	正交编码器1 相位A输入
				EPWM0_CH1	I/O	MFP12	EPWM0 通道1输出/捕获输入脚

32管脚	48管脚	64管脚	128管脚	管脚名称	类型	MFP	描述
				BPWM0_CH4	I/O	MFP13	BPWM0 通道4输出/捕获输入脚
			99	PE.5	I/O	MFP0	通用数字I/O脚
				EBI_nRD	O	MFP2	EBI 读使能输出脚
				SD0_DAT3	I/O	MFP3	SD/SDIO0数据线位3
				SC0_PWR	O	MFP6	智能卡0电源脚
				UART6_TXD	O	MFP8	UART6 数据发送输出脚.
				UART7_nRTS	O	MFP9	UART7 请求发送输出脚
				QEI1_B	I	MFP11	正交编码器1 相位B 输入
				EPWM0_CH2	I/O	MFP12	EPWM0 通道2输出/捕获输入脚
				BPWM0_CH3	I/O	MFP13	BPWM0 通道3输出/捕获输入脚
			100	PE.4	I/O	MFP0	通用数字I/O脚
				EBI_nWR	O	MFP2	EBI 写使能输出脚
				SD0_DAT2	I/O	MFP3	SD/SDIO0 数据线位2
				SC0_RST	O	MFP6	智能卡0复位脚
				UART6_RXD	I	MFP8	UART6 数据接收输入脚
				UART7_nCTS	I	MFP9	UART7 禁止发送输入管脚
				QEI0_INDEX	I	MFP11	正交编码器0 索引输入
				EPWM0_CH3	I/O	MFP12	EPWM0 通道3输出/捕获输入脚
				BPWM0_CH2	I/O	MFP13	BPWM0 通道2 输出/捕获输入
			101	PE.3	I/O	MFP0	通用数字I/O脚
				EBI_MCLK	O	MFP2	EBI 外部时钟输出脚.
				SD0_DAT1	I/O	MFP3	SD/SDIO0 数据线位1
				SC0_DAT	I/O	MFP6	智能卡0数据输入脚
				UART6_nRTS	O	MFP8	UART6 请求发送输出脚
				UART7_TXD	O	MFP9	UART7 数据发送输出脚.
				QEI0_A	I	MFP11	正交编码器0 相位A输入
				EPWM0_CH4	I/O	MFP12	EPWM0 通道4输出/捕获输入脚
				BPWM0_CH1	I/O	MFP13	BPWM0 通道1 输出/捕获输入
			102	PE.2	I/O	MFP0	通用数字I/O脚
				EBI_ALE	O	MFP2	EBI地址锁存使能输出脚

32管脚	48管脚	64管脚	128管脚	管脚名称	类型	MFP	描述
				SD0_DAT0	I/O	MFP3	SD/SDIO0 数据线位0
				SC0_CLK	O	MFP6	智能卡0时钟管脚
				UART6_nCTS	I	MFP8	UART6 禁止发送输入管脚
				UART7_RXD	I	MFP9	UART7 数据接收输入脚
				QEI0_B	I	MFP11	正交编码器0 相位B 输入
				EPWM0_CH5	I/O	MFP12	EPWM0 通道5 输出/捕获输入
				BPWM0_CH0	I/O	MFP13	BPWM0 通道0 输出/捕获输入
			103	V_{SS}	P	MFP0	数字电路地管脚
			104	V_{DD}	P	MFP0	I/O和内部PLL及数字电路的LDO电源
			105	PE.1	I/O	MFP0	通用数字I/O脚
				EBI_AD10	I/O	MFP2	EBI地址/数据总线位10.
				QSPI0_MISO0	I/O	MFP3	Quad SPI0 MISO0（主机输入，从机输出）脚.
				I^2S0_BCLK	O	MFP5	I2S0位时钟输出管脚
				SPI1_MISO	I/O	MFP6	SPI1 MISO（主机输入，从机输出）脚.
				UART3_TXD	O	MFP7	UART3 数据发送输出脚.
				I^2C1_SCL	I/O	MFP8	I^2C1 时钟脚
				UART4_nCTS	I	MFP9	UART4 禁止发送输入管脚
			106	PE.0	I/O	MFP0	通用数字I/O脚
				EBI_AD11	I/O	MFP2	EBI地址/数据总线位11.
				QSPI0_MOSI0	I/O	MFP3	Quad SPI0 MOSI0 (主机输出，从机输入) 管脚.
				I^2S0_MCLK	O	MFP5	I2S0 主时钟输出脚
				SPI1_MOSI	I/O	MFP6	SPI1 MOSI (主机输出，从机输入) 管脚.
				UART3_RXD	I	MFP7	UART3 数据接收输入脚
				I^2C1_SDA	I/O	MFP8	I^2C1 数据输入/输出脚
				UART4_nRTS	O	MFP9	UART4 请求发送输出脚
			107	PH.8	I/O	MFP0	通用数字I/O脚
				EBI_AD12	I/O	MFP2	EBI地址/数据总线位12.
				QSPI0_CLK	I/O	MFP3	Quad SPI0 串行时钟脚
				I^2S0_DI	I	MFP5	I2S0 数据输入脚
				SPI1_CLK	I/O	MFP6	SPI1 串行时钟脚

32 管脚	48 管脚	64 管脚	128 管脚	管脚名称	类型	MFP	描述
				UART3_nRTS	O	MFP7	UART3 请求发送输出脚
				I²C1_SMBAL	O	MFP8	I²C1 SMBus SMBALTER 管脚
				I²C2_SCL	I/O	MFP9	I²C2 时钟脚
				UART1_TXD	O	MFP10	UART1 数据发送输出脚
			108	PH.9	I/O	MFP0	通用数字I/O脚
				EBI_AD13	I/O	MFP2	EBI地址/数据总线位13.
				QSPI0_SS	I/O	MFP3	Quad SPI0从机选择管脚.
				I²S0_DO	O	MFP5	I²S0 数据输出脚
				SPI1_SS	I/O	MFP6	SPI1从机选择管脚.
				UART3_nCTS	I	MFP7	UART3 禁止发送输入管脚
				I²C1_SMBSUS	O	MFP8	I²C1 SMBus SMBSUS 管脚 (PMBus 控制管脚)
				I²C2_SDA	I/O	MFP9	I²C2 数据输入/输出脚
				UART1_RXD	I	MFP10	UART1数据接收输入管脚
			109	PH.10	I/O	MFP0	通用数字I/O脚
				EBI_AD14	I/O	MFP2	EBI地址/数据总线位14.
				QSPI0_MISO1	I/O	MFP3	Quad SPI0 MISO1（主机输入，从机输出）脚.
				I²S0_LRCK	O	MFP5	I²S0左右通道时钟输出脚.
				SPI1_I2SMCLK	I/O	MFP6	SPI1 I²S 主机时钟输出脚
				UART4_TXD	O	MFP7	UART4 数据发送输出脚.
				UART0_TXD	O	MFP8	UART0 数据发送输出脚.
			110	PH.11	I/O	MFP0	通用数字I/O脚
				EBI_AD15	I/O	MFP2	EBI地址/数据总线位15.
				QSPI0_MOSI1	I/O	MFP3	Quad SPI0 MOSI1 (主机输出，从机输入) 管脚.
				UART4_RXD	I	MFP7	UART4 数据接收输入脚
				UART0_RXD	I	MFP8	UART0 数据接收输入脚
				EPWM0_CH5	I/O	MFP11	EPWM0 通道5 输出/捕获输入
			111	PD.14	I/O	MFP0	通用数字I/O脚
				EBI_nCS0	O	MFP2	EBI片选0输出脚
				SPI0_I2SMCLK	I/O	MFP5	SPI0 I²S 主机时钟输出脚
				EPWM0_CH4	I/O	MFP11	EPWM0 通道4输出/捕获输入脚

32管脚	48管脚	64管脚	128管脚	管脚名称	类型	MFP	描述
25	37	49	112	V_{SS}	P	MFP0	数字电路地管脚
26	38	50	113	LDO_CAP	A	MFP0	LDO输出脚
27	39	51	114	V_{DD}	P	MFP0	I/O和内部PLL及数字电路的LDO电源
	40	52	115	PC.14	I/O	MFP0	通用数字I/O脚
				EBI_AD11	I/O	MFP2	EBI地址/数据总线位11.
				SPI0_I2SMCLK	I/O	MFP4	SPI0 I^2S 主机时钟输出脚
				QSPI0_CLK	I/O	MFP6	Quad SPI0 串行时钟脚
				EPWM0_SYNC_IN	I	MFP11	EPWM0 计数器同步触发输入脚
				ETM_TRACE_CLK	I	MFP12	ETM接收追踪时钟输入管脚
				TM1	I/O	MFP13	Timer1 事件计数输入/反转输出脚
				USB_VBUS_ST	I	MFP14	USB 外部VBUS 电源状态管脚.
28	41	53	116	PB.15	I/O	MFP0	通用数字I/O脚
				EADC0_CH15	A	MFP1	EADC0 通道15 模拟输入
				EADC1_CH15	A	MFP1	EADC1 通道15 模拟输入
				EBI_AD12	I/O	MFP2	EBI地址/数据总线位12.
				SPI0_SS	I/O	MFP4	SPI0从机选择管脚.
				UART0_nCTS	I	MFP6	UART0 禁止发送输入管脚
				UART3_TXD	O	MFP7	UART3 数据发送输出脚.
				I^2C2_SMBAL	O	MFP8	I^2C2 SMBus SMBALTER 管脚
				EPWM0_BRAKE1	I	MFP10	EPWM0 刹车1 输入脚
				EPWM1_CH0	I/O	MFP11	EPWM1 通道0 输出/捕获输入
				ETM_TRACE_DATA0	I	MFP12	ETM接收追踪数据0 输入管脚
				TM0_EXT	I/O	MFP13	Timer0 外部捕获输入/反转输出脚
				USB_VBUS_EN	O	MFP14	USB 外部 VBUS 电源使能管脚.
29	42	54	117	PB.14	I/O	MFP0	通用数字I/O脚
				EADC0_CH14	A	MFP1	EADC0 通道14 模拟输入
				EADC1_CH14	A	MFP1	EADC1 通道14 模拟输入
				EBI_AD13	I/O	MFP2	EBI地址/数据总线位13.
				SPI0_CLK	I/O	MFP4	SPI0 串行时钟脚
				UART0_nRTS	O	MFP6	UART0 请求发送输出脚

32 管脚	48 管脚	64 管脚	128 管脚	管脚名称	类型	MFP	描述
				UART3_RXD	I	MFP7	UART3 数据接收输入脚
				I²C2_SMBSUS	O	MFP8	I²C2 SMBus SMBSUS 管脚 (PMBus 控制管脚)
				EPWM1_CH1	I/O	MFP11	EPWM1 通道1 输出/捕获输入
				ETM_TRACE_DATA1	I	MFP12	ETM接收追踪数据1 输入管脚
				TM1_EXT	I/O	MFP13	Timer1 外部捕获输入/反转输出脚
				CLKO	O	MFP14	时钟输出
				USB_VBUS_ST	I	MFP15	USB 外部VBUS 电源状态管脚.
30	43	55	118	PB.13	I/O	MFP0	通用数字I/O脚
				EADC0_CH13	A	MFP1	EADC0 通道13 模拟输入
				EADC1_CH13	A	MFP1	EADC1 通道13 模拟输入
				ACMP0_P3	A	MFP1	模拟比较器 0 正极输入 3 管脚.
				ACMP1_P3	A	MFP1	模拟比较器 1 正极输入 3 管脚.
				EBI_AD14	I/O	MFP2	EBI地址/数据总线位14.
				SPI0_MISO	I/O	MFP4	SPI0 MISO（主机输入，从机输出）脚.
				UART0_TXD	O	MFP6	UART0 数据发送输出脚.
				UART3_nRTS	O	MFP7	UART3 请求发送输出脚
				I²C2_SCL	I/O	MFP8	I²C2 时钟脚
				CAP_PIXCLK	I	MFP10	图像采集接口像素时钟输入脚
				EPWM1_CH2	I/O	MFP11	EPWM1 通道2 输出/捕获输入
				ETM_TRACE_DATA2	I	MFP12	ETM接收追踪数据2 输入管脚
				TM2_EXT	I/O	MFP13	Timer2 外部捕获输入/反转输出脚
31	44	56	119	PB.12	I/O	MFP0	通用数字I/O脚
				EADC0_CH12	A	MFP1	EADC0 通道12 模拟输入
				EADC1_CH12	A	MFP1	EADC1 通道12 模拟输入
				DAC0_OUT	A	MFP1	DAC0 模拟输出通道
				ACMP0_P2	A	MFP1	模拟比较器 0 正极输入 2 管脚.
				ACMP1_P2	A	MFP1	模拟比较器 1 正极输入 2 管脚.
				EBI_AD15	I/O	MFP2	EBI地址/数据总线位15.
				SPI0_MOSI	I/O	MFP4	SPI0 MOSI (主机输出，从机输入) 管脚.
				UART0_RXD	I	MFP6	UART0 数据接收输入脚

32 管脚	48 管脚	64 管脚	128 管脚	管脚名称	类型	MFP	描述
				UART3_nCTS	I	MFP7	UART3 禁止发送输入管脚
				I²C2_SDA	I/O	MFP8	I²C2 数据输入/输出脚
				SD0_nCD	I	MFP9	SD/SDIO0 卡检测输入管脚
				CAP_SCLK	O	MFP10	图像采集接口传感器时钟管脚.
				EPWM1_CH3	I/O	MFP11	EPWM1 通道3输出/捕获输入脚
				ETM_TRACE_DATA3	I	MFP12	ETM接收追踪数据3 输入管脚
				TM3_EXT	I/O	MFP13	Timer3 外部捕获输入/反转输出脚
32	45	57	120	AV_{DD}	P	MFP0	内部模拟电路电源
		58	121	V_{REF}	A	MFP0	ADC参考电压输入. 注: 此管脚需接1uF电容到地.
	46	59	122	AV_{SS}	P	MFP0	模拟电路地管脚
		60	123	PB.11	I/O	MFP0	通用数字I/O脚
				EADC0_CH11	A	MFP1	EADC0 通道11 模拟输入
				EBI_ADR16	O	MFP2	EBI 地址总线位 16.
				UART0_nCTS	I	MFP5	UART0 禁止发送输入管脚
				UART4_TXD	O	MFP6	UART4 数据发送输出脚.
				I²C1_SCL	I/O	MFP7	I²C1 时钟脚
				CAN0_TXD	O	MFP8	CAN0总线发送输出
				SPI0_I2SMCLK	I/O	MFP9	SPI0 I²S 主机时钟输出脚
				BPWM1_CH0	I/O	MFP10	BPWM1 通道0 输出/捕获输入
				CAP_SFIELD	I	MFP12	图像输入接口SFIELD 输入脚
		61	124	PB.10	I/O	MFP0	通用数字I/O脚
				EADC0_CH10	A	MFP1	EADC0 通道10 模拟输入
				EBI_ADR17	O	MFP2	EBI 地址总线位 17.
				UART0_nRTS	O	MFP5	UART0 请求发送输出脚
				UART4_RXD	I	MFP6	UART4 数据接收输入脚
				I²C1_SDA	I/O	MFP7	I²C1 数据输入/输出脚
				CAN0_RXD	I	MFP8	CAN0 总线接收输入
				BPWM1_CH1	I/O	MFP10	BPWM1 通道1 输出/捕获输入
				CAP_VSYNC	I	MFP12	图像采集接口vsync 输入脚
		62	125	PB.9	I/O	MFP0	通用数字I/O脚

32 管脚	48 管脚	64 管脚	128 管脚	管脚名称	类型	MFP	描述
				EADC0_CH9	A	MFP1	EADC0 通道9 模拟输入
				EBI_ADR18	O	MFP2	EBI 地址总线位 18.
				UART0_TXD	O	MFP5	UART0 数据发送输出脚.
				UART1_nCTS	I	MFP6	UART1 禁止发送输入管脚
				I^2C1_SMBAL	O	MFP7	I^2C1 SMBus SMBALTER 管脚
				UART7_TXD	O	MFP8	UART7 数据发送输出脚.
				I^2C0_SCL	I/O	MFP9	I2C0 时钟管脚
				BPWM1_CH2	I/O	MFP10	BPWM1 通道2 输出/捕获输入
				CAN2_TXD	O	MFP12	CAN2总线发送输出
				INT7	I	MFP13	外部中断7 输入脚
				CAP_HSYNC	I	MFP14	图像采集接口hsync 输入脚
		63	126	PB.8	I/O	MFP0	通用数字I/O脚
				EADC0_CH8	A	MFP1	EADC0 通道8 模拟输入
				EBI_ADR19	O	MFP2	EBI 地址总线位 19.
				UART0_RXD	I	MFP5	UART0 数据接收输入脚
				UART1_nRTS	O	MFP6	UART1 请求发送输出脚
				I^2C1_SMBSUS	O	MFP7	I^2C1 SMBus SMBSUS 管脚 (PMBus 控制管脚)
				UART7_RXD	I	MFP8	UART7 数据接收输入脚
				I^2C0_SDA	I/O	MFP9	I2C0 数据输入/输出管脚
				BPWM1_CH3	I/O	MFP10	BPWM1 通道3输出/捕获输入脚
				CAN2_RXD	I	MFP12	CAN2 总线接收输入
				INT6	I	MFP13	外部中断6 输入脚
	47	64	127	PB.7	I/O	MFP0	通用数字I/O脚
				EADC0_CH7	A	MFP1	EADC0 通道7 模拟输入
				EBI_nWRL	O	MFP2	EBI低字节写使能输出脚
				CAN1_TXD	O	MFP5	CAN1总线发送输出
				UART1_TXD	O	MFP6	UART1 数据发送输出脚
				EBI_nCS0	O	MFP8	EBI片选0输出脚
				BPWM1_CH4	I/O	MFP10	BPWM1 通道4输出/捕获输入脚
				EPWM1_BRAKE0	I	MFP11	EPWM1 刹车0 输入脚

32 管脚	48 管脚	64 管脚	128 管脚	管脚名称	类型	MFP	描述
				EPWM1_CH4	I/O	MFP12	EPWM1 通道4输出/捕获输入脚
				INT5	I	MFP13	外部中断5 输入脚
				USB_VBUS_ST	I	MFP14	USB 外部VBUS 电源状态管脚.
				ACMP0_O	O	MFP15	模拟比较器 0输出脚
	48	1	128	PB.6	I/O	MFP0	通用数字I/O脚
				EADC0_CH6	A	MFP1	EADC0通道6模拟输入
				EBI_nWRH	O	MFP2	EBI 高字节写使能输出管脚
				CAN1_RXD	I	MFP5	CAN1 总线接收输入
				UART1_RXD	I	MFP6	UART1数据接收输入管脚
				EBI_nCS1	O	MFP8	EBI片选1输出管脚
				BPWM1_CH5	I/O	MFP10	EPWM1 通道5输出/捕获输入
				EPWM1_BRAKE1	I	MFP11	EPWM1 刹车1输入管脚
				EPWM1_CH5	I/O	MFP12	EPWM1 通道5输出/捕获输入
				INT4	I	MFP13	外部中断4输入管脚
				USB_VBUS_EN	O	MFP14	USB 外部 VBUS 电源使能管脚.
				ACMP1_O	O	MFP15	模拟比较器1输出脚

4.3.3 M483系列管脚描述

64 管脚	128 管脚	管脚名称	类型	MFP	描述
2	1	PB.5	I/O	MFP0	通用数字I/O脚
		EADC0_CH5	A	MFP1	EADC0 通道5模拟输入
		ACMP1_N	A	MFP1	模拟比较器 1 负极输入管脚.
		EBI_ADR0	O	MFP2	EBI地址总线位0
		SD0_DAT3	I/O	MFP3	SD/SDIO0数据线位3
		SPI1_MISO	I/O	MFP5	SPI1 MISO（主机输入，从机输出）脚.
		I^2C0_SCL	I/O	MFP6	I2C0 时钟管脚
		UART5_TXD	O	MFP7	UART5数据发送输出脚
		SC0_CLK	O	MFP9	智能卡0时钟管脚
		I^2S0_BCLK	O	MFP10	I2S0位时钟输出管脚
		EPWM0_CH0	I/O	MFP11	EPWM0 通道0输出/捕获输入
		UART2_TXD	O	MFP12	UART2 数据发送输出脚.
		TM0	I/O	MFP14	Timer0 事件计数输入/反转输出管脚
		INT0	I	MFP15	外部中断0输入管脚
3	2	PB.4	I/O	MFP0	通用数字I/O脚
		EADC0_CH4	A	MFP1	EADC0 通道4模拟输入
		ACMP1_P1	A	MFP1	模拟比较器1正极输入1脚
		EBI_ADR1	O	MFP2	EBI 地址总线位1
		SD0_DAT2	I/O	MFP3	SD/SDIO0 数据线位2
		SPI1_MOSI	I/O	MFP5	SPI1 MOSI (主机输出，从机输入) 管脚.
		I^2C0_SDA	I/O	MFP6	I2C0 数据输入/输出管脚
		UART5_RXD	I	MFP7	UART5 数据接收输入管脚
		SC0_DAT	I/O	MFP9	智能卡0数据输入脚
		I^2S0_MCLK	O	MFP10	I2S0 主时钟输出脚
		EPWM0_CH1	I/O	MFP11	EPWM0 通道1输出/捕获输入脚
		UART2_RXD	I	MFP12	UART2 数据接收输入脚
		TM1	I/O	MFP14	Timer1 事件计数输入/反转输出脚
		INT1	I	MFP15	外部中断1输入脚
4	3	PB.3	I/O	MFP0	通用数字I/O脚

64 管脚	128 管脚	管脚名称	类型	MFP	描述
		EADC0_CH3	A	MFP1	EADC0 通道3模拟输入
		EADC1_CH11	A	MFP1	EADC1 通道11 模拟输入
		ACMP0_N	A	MFP1	模拟比较器 0 负极输入管脚.
		EBI_ADR2	O	MFP2	EBI 地址总线位2
		SD0_DAT1	I/O	MFP3	SD/SDIO0 数据线位1
		SPI1_CLK	I/O	MFP5	SPI1 串行时钟脚
		UART1_TXD	O	MFP6	UART1 数据发送输出脚
		UART5_nRTS	O	MFP7	UART5 请求发送输出脚
		SC0_RST	O	MFP9	智能卡0复位脚
		I^2S0_DI	I	MFP10	I2S0 数据输入脚
		EPWM0_CH2	I/O	MFP11	EPWM0 通道2输出/捕获输入脚
		I^2C1_SCL	I/O	MFP12	I^2C1 时钟脚
		TM2	I/O	MFP14	Timer2 事件计数输入/反转输出脚
		INT2	I	MFP15	外部中断2输入脚
5	4	PB.2	I/O	MFP0	通用数字I/O脚
		EADC0_CH2	A	MFP1	EADC0 通道2 模拟输入
		EADC1_CH10	A	MFP1	EADC1 通道10 模拟输入
		ACMP0_P1	A	MFP1	模拟比较器0正极输入1脚
		EBI_ADR3	O	MFP2	EBI 地址总线位3
		SD0_DAT0	I/O	MFP3	SD/SDIO0 数据线位0
		SPI1_SS	I/O	MFP5	SPI1从机选择管脚.
		UART1_RXD	I	MFP6	UART1数据接收输入管脚
		UART5_nCTS	I	MFP7	UART5 禁止发送输入管脚
		SC0_PWR	O	MFP9	智能卡0电源脚
		I^2S0_DO	O	MFP10	I^2S0 数据输出脚
		EPWM0_CH3	I/O	MFP11	EPWM0 通道3输出/捕获输入脚
		I^2C1_SDA	I/O	MFP12	I^2C1 数据输入/输出脚
		TM3	I/O	MFP14	Timer3 事件计数输入/反转输出脚
		INT3	I	MFP15	外部中断3输入脚
	5	PC.12	I/O	MFP0	通用数字I/O脚
		EBI_ADR4	O	MFP2	EBI 地址总线位 4.

64 管脚	128 管脚	管脚名称	类型	MFP	描述
		UART0_TXD	O	MFP3	UART0 数据发送输出脚.
		I²C0_SCL	I/O	MFP4	I2C0 时钟管脚
		UART6_TXD	O	MFP5	UART6 数据发送输出脚.
		SC0_nCD	I	MFP9	智能卡 0 卡检测管脚.
		ECAP1_IC2	I	MFP11	增强型捕获计数器 1 输入 2 管脚.
		EPWM1_CH0	I/O	MFP12	EPWM1 通道0 输出/捕获输入
		ACMP0_O	O	MFP14	模拟比较器 0输出脚
	6	PC.11	I/O	MFP0	通用数字I/O脚
		EBI_ADR5	O	MFP2	EBI 地址总线位 5.
		UART0_RXD	I	MFP3	UART0 数据接收输入脚
		I²C0_SDA	I/O	MFP4	I2C0 数据输入/输出管脚
		UART6_RXD	I	MFP5	UART6 数据接收输入脚
		ECAP1_IC1	I	MFP11	增强型捕获计数器 1 输入 1 管脚.
		EPWM1_CH1	I/O	MFP12	EPWM1 通道1 输出/捕获输入
		ACMP1_O	O	MFP14	模拟比较器1输出脚
	7	PC.10	I/O	MFP0	通用数字I/O脚
		EBI_ADR6	O	MFP2	EBI 地址总线位 6.
		UART6_nRTS	O	MFP5	UART6 请求发送输出脚
		UART3_TXD	O	MFP7	UART3 数据发送输出脚.
		CAN1_TXD	O	MFP9	CAN1总线发送输出
		ECAP1_IC0	I	MFP11	增强型捕获计数器 1 输入脚0
		EPWM1_CH2	I/O	MFP12	EPWM1 通道2 输出/捕获输入
		EADC1_ST	I	MFP14	EADC1 外部触发输入
	8	PC.9	I/O	MFP0	通用数字I/O脚
		EBI_ADR7	O	MFP2	EBI 地址总线位 7.
		UART6_nCTS	I	MFP5	UART6 禁止发送输入管脚
		UART3_RXD	I	MFP7	UART3 数据接收输入脚
		CAN1_RXD	I	MFP9	CAN1 总线接收输入
		EPWM1_CH3	I/O	MFP12	EPWM1 通道3输出/捕获输入脚
		EADC1_ST	I	MFP14	EADC1 外部触发输入
6	9	PB.1	I/O	MFP0	通用数字I/O脚

64 管脚	128 管脚	管脚名称	类型	MFP	描述
		EADC0_CH1	A	MFP1	EADC0 通道1 模拟输入
		EADC1_CH9	A	MFP1	EADC1 通道9 模拟输入
		EBI_ADR8	O	MFP2	EBI 地址总线位 8.
		SD0_CLK	O	MFP3	SD/SDIO0 时钟输出脚
		SPI1_I2SMCLK	I/O	MFP5	SPI1 I^2S 主机时钟输出脚
		UART2_TXD	O	MFP7	UART2 数据发送输出脚.
		I^2C1_SCL	I/O	MFP9	I^2C1 时钟脚
		I^2S0_LRCK	O	MFP10	I^2S0左右通道时钟输出脚.
		EPWM0_CH4	I/O	MFP11	EPWM0 通道4输出/捕获输入脚
		EPWM1_CH4	I/O	MFP12	EPWM1 通道4输出/捕获输入脚
		EPWM0_BRAKE0	I	MFP13	EPWM0 刹车0 输入脚
7	10	PB.0	I/O	MFP0	通用数字I/O脚
		EADC0_CH0	A	MFP1	EADC0 通道0 模拟输入
		EADC1_CH8	A	MFP1	EADC1 通道8 模拟输入
		EBI_ADR9	O	MFP2	EBI 地址总线位 9.
		SD0_CMD	I/O	MFP3	SD/SDIO0 命令/响应脚
		SPI2_I2SMCLK	I/O	MFP4	SPI2 I^2S 主机时钟输出脚
		UART2_RXD	I	MFP7	UART2 数据接收输入脚
		SPI0_I2SMCLK	I/O	MFP8	SPI0 I^2S 主机时钟输出脚
		I^2C1_SDA	I/O	MFP9	I^2C1 数据输入/输出脚
		EPWM0_CH5	I/O	MFP11	EPWM0 通道5 输出/捕获输入
		EPWM1_CH5	I/O	MFP12	EPWM1 通道5输出/捕获输入
		EPWM0_BRAKE1	I	MFP13	EPWM0 刹车1 输入脚
	11	V_{SS}	P	MFP0	数字电路地管脚
	12	V_{DD}	P	MFP0	I/O和内部PLL及数字电路的LDO电源
8	13	PA.11	I/O	MFP0	通用数字I/O脚
		EADC1_CH7	A	MFP1	EADC1 通道7 模拟输入
		ACMP0_P0	A	MFP1	模拟比较器 0 正极输入脚0
		EBI_nRD	O	MFP2	EBI 读使能输出脚
		SPI2_SS	I/O	MFP4	SPI2从机选择管脚.
		I^2C2_SCL	I/O	MFP7	I^2C2 时钟脚

64 管脚	128 管脚	管脚名称	类型	MFP	描述
		UART6_TXD	O	MFP8	UART6 数据发送输出脚.
		BPWM0_CH0	I/O	MFP9	BPWM0 通道0 输出/捕获输入
		EPWM0_SYNC_OUT	O	MFP10	EPWM0计数器同步触发输出脚
		TM0_EXT	I/O	MFP13	Timer0 外部捕获输入/反转输出脚
9	14	PA.10	I/O	MFP0	通用数字I/O脚
		EADC1_CH6	A	MFP1	EADC1 通道6 模拟输入
		ACMP1_P0	A	MFP1	模拟比较器 1 正极输入脚0
		EBI_nWR	O	MFP2	EBI 写使能输出脚
		SPI2_CLK	I/O	MFP4	SPI2 串行时钟管脚
		I^2C2_SDA	I/O	MFP7	I^2C2 数据输入/输出脚
		UART6_RXD	I	MFP8	UART6 数据接收输入脚
		BPWM0_CH1	I/O	MFP9	BPWM0 通道1 输出/捕获输入
		QEI1_INDEX	I	MFP10	正交编码器1 索引输入
		ECAP0_IC0	I	MFP11	增强型捕获计数器 0 输入脚0
		TM1_EXT	I/O	MFP13	Timer1 外部捕获输入/反转输出脚
		DAC0_ST	I	MFP14	DAC0 外部触发输入
		SWDH_CLK	O	MFP15	串行调试总线主时钟输出
10	15	PA.9	I/O	MFP0	通用数字I/O脚
		EADC1_CH5	A	MFP1	EADC1 通道5 模拟输入
		EBI_MCLK	O	MFP2	EBI 外部时钟输出脚.
		SPI2_MISO	I/O	MFP4	SPI2 MISO(主机输入，从机输出)管脚
		UART1_TXD	O	MFP7	UART1 数据发送输出脚
		UART7_TXD	O	MFP8	UART7 数据发送输出脚.
		BPWM0_CH2	I/O	MFP9	BPWM0 通道2 输出/捕获输入
		QEI1_A	I	MFP10	正交编码器1 相位A输入
		ECAP0_IC1	I	MFP11	增强型捕获计数器 0 输入 1 管脚.
		TM2_EXT	I/O	MFP13	Timer2 外部捕获输入/反转输出脚
		SWDH_DAT	I/O	MFP15	串行调试总线主机数据输入/输出管脚
11	16	PA.8	I/O	MFP0	通用数字I/O脚
		EADC1_CH4	A	MFP1	EADC1 通道4 模拟输入
		EBI_ALE	O	MFP2	EBI地址锁存使能输出脚

64 管脚	128 管脚	管脚名称	类型	MFP	描述
		SPI2_MOSI	I/O	MFP4	SPI2 MOSI (主机输出，从机输入) 管脚
		UART1_RXD	I	MFP7	UART1数据接收输入管脚
		UART7_RXD	I	MFP8	UART7 数据接收输入脚
		BPWM0_CH3	I/O	MFP9	BPWM0 通道3输出/捕获输入脚
		QEI1_B	I	MFP10	正交编码器1 相位B 输入
		ECAP0_IC2	I	MFP11	增强型捕获计数器 0 输入 2 管脚.
		TM3_EXT	I/O	MFP13	Timer3 外部捕获输入/反转输出脚
		INT4	I	MFP15	外部中断4输入管脚
	17	PC.13	I/O	MFP0	通用数字I/O脚
		EADC1_CH3	A	MFP1	EADC1 通道3 模拟输入
		EBI_ADR10	O	MFP2	EBI 地址总线位 10.
		SPI2_I2SMCLK	I/O	MFP4	SPI2 I^2S 主机时钟输出脚
		CAN1_TXD	O	MFP5	CAN1总线发送输出
		UART2_TXD	O	MFP7	UART2 数据发送输出脚.
		BPWM0_CH4	I/O	MFP9	BPWM0 通道4输出/捕获输入脚
		CLKO	O	MFP13	时钟输出
		EADC0_ST	I	MFP14	EADC0 外部触发输入
	18	PD.12	I/O	MFP0	通用数字I/O脚
		EADC1_CH2	A	MFP1	EADC1 通道2 模拟输入
		EBI_nCS0	O	MFP2	EBI片选0输出脚
		CAN1_RXD	I	MFP5	CAN1 总线接收输入
		UART2_RXD	I	MFP7	UART2 数据接收输入脚
		BPWM0_CH5	I/O	MFP9	BPWM0 通道5 输出/捕获输入
		QEI0_INDEX	I	MFP10	正交编码器0 索引输入
		CLKO	O	MFP13	时钟输出
		EADC0_ST	I	MFP14	EADC0 外部触发输入
		INT5	I	MFP15	外部中断5 输入脚
	19	PD.11	I/O	MFP0	通用数字I/O脚
		EADC1_CH1	A	MFP1	EADC1 通道1 模拟输入
		EBI_nCS1	O	MFP2	EBI片选1输出管脚
		UART1_TXD	O	MFP3	UART1 数据发送输出脚

64 管脚	128 管脚	管脚名称	类型	MFP	描述
		CAN0_TXD	O	MFP4	CAN0总线发送输出
		QEI0_A	I	MFP10	正交编码器0 相位A输入
		INT6	I	MFP15	外部中断6 输入脚
	20	PD.10	I/O	MFP0	通用数字I/O脚
		EADC1_CH0	A	MFP1	EADC1 通道0 模拟输入
		EBI_nCS2	O	MFP2	EBI片选2输出脚
		UART1_RXD	I	MFP3	UART1数据接收输入管脚
		CAN0_RXD	I	MFP4	CAN0 总线接收输入
		QEI0_B	I	MFP10	正交编码器0 相位B 输入
		INT7	I	MFP15	外部中断7 输入脚
	21	PG.2	I/O	MFP0	通用数字I/O脚
		EBI_ADR11	O	MFP2	EBI 地址总线位 11.
		SPI2_SS	I/O	MFP3	SPI2从机选择管脚.
		I^2C0_SMBAL	O	MFP4	I^2C0 SMBus SMBALTER 管脚
		I^2C1_SCL	I/O	MFP5	I^2C1 时钟脚
		CAP_DATA7	I	MFP7	图像数据输入总线位7.
		TM0	I/O	MFP13	Timer0 事件计数输入/反转输出管脚
	22	PG.3	I/O	MFP0	通用数字I/O脚
		EBI_ADR12	O	MFP2	EBI 地址总线位 12.
		SPI2_CLK	I/O	MFP3	SPI2 串行时钟管脚
		I^2C0_SMBSUS	O	MFP4	I^2C0 SMBus SMBSUS 管脚 (PMBus 控制管脚)
		I^2C1_SDA	I/O	MFP5	I^2C1 数据输入/输出脚
		CAP_DATA6	I	MFP7	图像数据输入总线位6.
		TM1	I/O	MFP13	Timer1 事件计数输入/反转输出脚
	23	PG.4	I/O	MFP0	通用数字I/O脚
		EBI_ADR13	O	MFP2	EBI 地址总线位 13.
		SPI2_MISO	I/O	MFP3	SPI2 MISO(主机输入，从机输出)管脚
		CAP_DATA5	I	MFP7	图像数据输入总线位5.
		TM2	I/O	MFP13	Timer2 事件计数输入/反转输出脚
	24	PF.11	I/O	MFP0	通用数字I/O脚
		EBI_ADR14	O	MFP2	EBI 地址总线位 14.

64 管脚	128 管脚	管脚名称	类型	MFP	描述
		SPI2_MOSI	I/O	MFP3	SPI2 MOSI (主机输出，从机输入) 管脚
		UART5_TXD	O	MFP6	UART5数据发送输出脚
		CAP_DATA4	I	MFP7	图像数据输入总线位4.
		TM3	I/O	MFP13	Timer3 事件计数输入/反转输出脚
	25	PF.10	I/O	MFP0	通用数字I/O脚
		EBI_ADR15	O	MFP2	EBI 地址总线位 15.
		SC0_nCD	I	MFP3	智能卡 0 卡检测管脚.
		I^2S0_BCLK	O	MFP4	I2S0位时钟输出管脚
		SPI0_I2SMCLK	I/O	MFP5	SPI0 I^2S 主机时钟输出脚
		UART5_RXD	I	MFP6	UART5 数据接收输入管脚
		CAP_DATA3	I	MFP7	图像数据输入总线位3.
	26	PF.9	I/O	MFP0	通用数字I/O脚
		EBI_ADR16	O	MFP2	EBI 地址总线位 16.
		SC0_PWR	O	MFP3	智能卡0电源脚
		I^2S0_MCLK	O	MFP4	I2S0 主时钟输出脚
		SPI0_SS	I/O	MFP5	SPI0从机选择管脚.
		UART5_nRTS	O	MFP6	UART5 请求发送输出脚
		CAP_DATA2	I	MFP7	图像数据输入总线位2.
		CAN1_TXD	O	MFP8	CAN1总线发送输出
	27	PF.8	I/O	MFP0	通用数字I/O脚
		EBI_ADR17	O	MFP2	EBI 地址总线位 17.
		SC0_RST	O	MFP3	智能卡0复位脚
		I^2S0_DI	I	MFP4	I2S0 数据输入脚
		SPI0_CLK	I/O	MFP5	SPI0 串行时钟脚
		UART5_nCTS	I	MFP6	UART5 禁止发送输入管脚
		CAP_DATA1	I	MFP7	图像数据输入总线位1.
		CAN1_RXD	I	MFP8	CAN1 总线接收输入
		TAMPER2	I/O	MFP10	TAMPER循环检测管脚 2.
	28	PF.7	I/O	MFP0	通用数字I/O脚
		EBI_ADR18	O	MFP2	EBI 地址总线位 18.
		SC0_DAT	I/O	MFP3	智能卡0数据输入脚

64 管脚	128 管脚	管脚名称	类型	MFP	描述
		I^2S0_DO	O	MFP4	I^2S0 数据输出脚
		SPI0_MISO	I/O	MFP5	SPI0 MISO（主机输入，从机输出）脚.
		UART4_TXD	O	MFP6	UART4 数据发送输出脚.
		CAP_DATA0	I	MFP7	图像数据输入总线位0.
		CAN2_TXD	O	MFP8	CAN2总线发送输出
		TAMPER1	I/O	MFP10	TAMPER循环检测管脚 1.
12	29	PF.6	I/O	MFP0	通用数字I/O脚
		EBI_ADR19	O	MFP2	EBI 地址总线位 19.
		SC0_CLK	O	MFP3	智能卡0时钟管脚
		I^2S0_LRCK	O	MFP4	I^2S0左右通道时钟输出脚.
		SPI0_MOSI	I/O	MFP5	SPI0 MOSI (主机输出，从机输入) 管脚.
		UART4_RXD	I	MFP6	UART4 数据接收输入脚
		EBI_nCS0	O	MFP7	EBI片选0输出脚
		CAN2_RXD	I	MFP8	CAN2 总线接收输入
		TAMPER0	I/O	MFP10	TAMPER循环检测管脚 0.
13	30	V_{BAT}	P	MFP0	RTC电池供电
14	31	PF.5	I/O	MFP0	通用数字I/O脚
		UART2_RXD	I	MFP2	UART2 数据接收输入脚
		UART2_nCTS	I	MFP4	UART2 禁止发送输入管脚
		EPWM0_CH0	I/O	MFP7	EPWM0 通道0输出/捕获输入
		BPWM0_CH4	I/O	MFP8	BPWM0 通道4输出/捕获输入脚
		EPWM0_SYNC_OUT	O	MFP9	EPWM0计数器同步触发输出脚
		X32_IN	I	MFP10	外部 32.768 kHz 晶振输入脚
		EADC0_ST	I	MFP11	EADC0 外部触发输入
15	32	PF.4	I/O	MFP0	通用数字I/O脚
		UART2_TXD	O	MFP2	UART2 数据发送输出脚.
		UART2_nRTS	O	MFP4	UART2 请求发送输出脚
		EPWM0_CH1	I/O	MFP7	EPWM0 通道1输出/捕获输入脚
		BPWM0_CH5	I/O	MFP8	BPWM0 通道5 输出/捕获输入
		X32_OUT	O	MFP10	外部 32.768 kHz 晶振输出脚
		EADC1_ST	I	MFP11	EADC1 外部触发输入

64 管脚	128 管脚	管脚名称	类型	MFP	描述
	33	PH.4	I/O	MFP0	通用数字I/O脚
		EBI_ADR3	O	MFP2	EBI 地址总线位3
		SPI1_MISO	I/O	MFP3	SPI1 MISO（主机输入，从机输出）脚.
		UART7_nRTS	O	MFP4	UART7 请求发送输出脚
		UART6_TXD	O	MFP5	UART6 数据发送输出脚.
	34	PH.5	I/O	MFP0	通用数字I/O脚
		EBI_ADR2	O	MFP2	EBI 地址总线位2
		SPI1_MOSI	I/O	MFP3	SPI1 MOSI (主机输出，从机输入) 管脚.
		UART7_nCTS	I	MFP4	UART7 禁止发送输入管脚
		UART6_RXD	I	MFP5	UART6 数据接收输入脚
	35	PH.6	I/O	MFP0	通用数字I/O脚
		EBI_ADR1	O	MFP2	EBI 地址总线位1
		SPI1_CLK	I/O	MFP3	SPI1 串行时钟脚
		UART7_TXD	O	MFP4	UART7 数据发送输出脚.
	36	PH.7	I/O	MFP0	通用数字I/O脚
		EBI_ADR0	O	MFP2	EBI地址总线位0
		SPI1_SS	I/O	MFP3	SPI1从机选择管脚.
		UART7_RXD	I	MFP4	UART7 数据接收输入脚
16	37	PF.3	I/O	MFP0	通用数字I/O脚
		EBI_nCS0	O	MFP2	EBI片选0输出脚
		UART0_TXD	O	MFP3	UART0 数据发送输出脚.
		I^2C0_SCL	I/O	MFP4	I2C0 时钟管脚
		XT1_IN	I	MFP10	外部 4~24 MHz (高速) 晶振输入脚
		BPWM1_CH0	I/O	MFP11	BPWM1 通道0 输出/捕获输入
17	38	PF.2	I/O	MFP0	通用数字I/O脚
		EBI_nCS1	O	MFP2	EBI片选1输出管脚
		UART0_RXD	I	MFP3	UART0 数据接收输入脚
		I^2C0_SDA	I/O	MFP4	I2C0 数据输入/输出管脚
		QSPI0_CLK	I/O	MFP5	Quad SPI0 串行时钟脚
		XT1_OUT	O	MFP10	外部 4~24 MHz (高速) 晶振输出脚
		BPWM1_CH1	I/O	MFP11	BPWM1 通道1 输出/捕获输入

64 管脚	128 管脚	管脚名称	类型	MFP	描述
	39	V_{SS}	P	MFP0	数字电路地管脚
	40	V_{DD}	P	MFP0	I/O和内部PLL及数字电路的LDO电源
	41	PE.8	I/O	MFP0	通用数字I/O脚
		EBI_ADR10	O	MFP2	EBI 地址总线位 10.
		I^2S0_BCLK	O	MFP4	I2S0位时钟输出管脚
		SPI2_CLK	I/O	MFP5	SPI2 串行时钟管脚
		UART2_TXD	O	MFP7	UART2 数据发送输出脚.
		EPWM0_CH0	I/O	MFP10	EPWM0 通道0输出/捕获输入
		EPWM0_BRAKE0	I	MFP11	EPWM0 刹车0 输入脚
		ECAP0_IC0	I	MFP12	增强型捕获计数器 0 输入脚0
		TRACE_DATA3	O	MFP14	ETM跟踪数据3 输出管脚
	42	PE.9	I/O	MFP0	通用数字I/O脚
		EBI_ADR11	O	MFP2	EBI 地址总线位 11.
		I^2S0_MCLK	O	MFP4	I2S0 主时钟输出脚
		SPI2_MISO	I/O	MFP5	SPI2 MISO(主机输入，从机输出)管脚
		UART2_RXD	I	MFP7	UART2 数据接收输入脚
		EPWM0_CH1	I/O	MFP10	EPWM0 通道1输出/捕获输入脚
		EPWM0_BRAKE1	I	MFP11	EPWM0 刹车1 输入脚
		ECAP0_IC1	I	MFP12	增强型捕获计数器 0 输入 1 管脚.
		TRACE_DATA2	O	MFP14	ETM跟踪数据2 输出管脚
	43	PE.10	I/O	MFP0	通用数字I/O脚
		EBI_ADR12	O	MFP2	EBI 地址总线位 12.
		I^2S0_DI	I	MFP4	I2S0 数据输入脚
		SPI2_MOSI	I/O	MFP5	SPI2 MOSI (主机输出，从机输入) 管脚
		UART3_TXD	O	MFP7	UART3 数据发送输出脚.
		EPWM0_CH2	I/O	MFP10	EPWM0 通道2输出/捕获输入脚
		EPWM1_BRAKE0	I	MFP11	EPWM1 刹车0 输入脚
		ECAP0_IC2	I	MFP12	增强型捕获计数器 0 输入 2 管脚.
		TRACE_DATA1	O	MFP14	ETM跟踪数据1 输出管脚
	44	PE.11	I/O	MFP0	通用数字I/O脚
		EBI_ADR13	O	MFP2	EBI 地址总线位 13.

64 管脚	128 管脚	管脚名称	类型	MFP	描述
		I²S0_DO	O	MFP4	I²S0 数据输出脚
		SPI2_SS	I/O	MFP5	SPI2从机选择管脚.
		UART3_RXD	I	MFP7	UART3 数据接收输入脚
		UART1_nCTS	I	MFP8	UART1 禁止发送输入管脚
		EPWM0_CH3	I/O	MFP10	EPWM0 通道3输出/捕获输入脚
		EPWM1_BRAKE1	I	MFP11	EPWM1 刹车1输入管脚
		ECAP1_IC2	I	MFP13	增强型捕获计数器 1 输入 2 管脚.
		TRACE_DATA0	O	MFP14	ETM跟踪数据0 输出管脚
	45	PE.12	I/O	MFP0	通用数字I/O脚
		EBI_ADR14	O	MFP2	EBI 地址总线位 14.
		I²S0_LRCK	O	MFP4	I²S0左右通道时钟输出脚.
		SPI2_I2SMCLK	I/O	MFP5	SPI2 I²S 主机时钟输出脚
		UART1_nRTS	O	MFP8	UART1 请求发送输出脚
		EPWM0_CH4	I/O	MFP10	EPWM0 通道4输出/捕获输入脚
		ECAP1_IC1	I	MFP13	增强型捕获计数器 1 输入 1 管脚.
		TRACE_CLK	O	MFP14	ETM跟踪时钟输出脚
	46	PE.13	I/O	MFP0	通用数字I/O脚
		EBI_ADR15	O	MFP2	EBI 地址总线位 15.
		I²C0_SCL	I/O	MFP4	I2C0 时钟管脚
		UART4_nRTS	O	MFP5	UART4 请求发送输出脚
		UART1_TXD	O	MFP8	UART1 数据发送输出脚
		EPWM0_CH5	I/O	MFP10	EPWM0 通道5 输出/捕获输入
		EPWM1_CH0	I/O	MFP11	EPWM1 通道0 输出/捕获输入
		BPWM1_CH5	I/O	MFP12	EPWM1 通道5输出/捕获输入
		ECAP1_IC0	I	MFP13	增强型捕获计数器 1 输入脚0
	47	PC.8	I/O	MFP0	通用数字I/O脚
		EBI_ADR16	O	MFP2	EBI 地址总线位 16.
		I²C0_SDA	I/O	MFP4	I2C0 数据输入/输出管脚
		UART4_nCTS	I	MFP5	UART4 禁止发送输入管脚
		UART1_RXD	I	MFP8	UART1数据接收输入管脚
		EPWM1_CH1	I/O	MFP11	EPWM1 通道1 输出/捕获输入

64 管脚	128 管脚	管脚名称	类型	MFP	描述
		BPWM1_CH4	I/O	MFP12	BPWM1 通道4输出/捕获输入脚
18	48	PC.7	I/O	MFP0	通用数字I/O脚
		EBI_AD9	I/O	MFP2	EBI地址/数据总线位9.
		SPI1_MISO	I/O	MFP4	SPI1 MISO（主机输入，从机输出）脚.
		UART4_TXD	O	MFP5	UART4 数据发送输出脚.
		UART0_nCTS	I	MFP7	UART0 禁止发送输入管脚
		I^2C1_SMBAL	O	MFP8	I^2C1 SMBus SMBALTER 管脚
		UART6_TXD	O	MFP9	UART6 数据发送输出脚.
		EPWM1_CH2	I/O	MFP11	EPWM1 通道2 输出/捕获输入
		BPWM1_CH0	I/O	MFP12	BPWM1 通道0 输出/捕获输入
		TM0	I/O	MFP14	Timer0 事件计数输入/反转输出管脚
		INT3	I	MFP15	外部中断3输入脚
19	49	PC.6	I/O	MFP0	通用数字I/O脚
		EBI_AD8	I/O	MFP2	EBI地址/数据总线位8.
		SPI1_MOSI	I/O	MFP4	SPI1 MOSI (主机输出，从机输入) 管脚.
		UART4_RXD	I	MFP5	UART4 数据接收输入脚
		UART0_nRTS	O	MFP7	UART0 请求发送输出脚
		I^2C1_SMBSUS	O	MFP8	I^2C1 SMBus SMBSUS 管脚 (PMBus 控制管脚)
		UART6_RXD	I	MFP9	UART6 数据接收输入脚
		EPWM1_CH3	I/O	MFP11	EPWM1 通道3输出/捕获输入脚
		BPWM1_CH1	I/O	MFP12	BPWM1 通道1 输出/捕获输入
		TM1	I/O	MFP14	Timer1 事件计数输入/反转输出脚
		INT2	I	MFP15	外部中断2输入脚
20	50	PA.7	I/O	MFP0	通用数字I/O脚
		EBI_AD7	I/O	MFP2	EBI地址/数据总线位7.
		SPI1_CLK	I/O	MFP4	SPI1 串行时钟脚
		UART0_TXD	O	MFP7	UART0 数据发送输出脚.
		I^2C1_SCL	I/O	MFP8	I^2C1 时钟脚
		QSPI1_MISO1	I/O	MFP9	Quad SPI1 MISO1（主机输入，从机输出）脚.
		EPWM1_CH4	I/O	MFP11	EPWM1 通道4输出/捕获输入脚
		BPWM1_CH2	I/O	MFP12	BPWM1 通道2 输出/捕获输入

64 管脚	128 管脚	管脚名称	类型	MFP	描述
		ACMP0_WLAT	I	MFP13	模拟比较器 0窗口锁存输入管脚
		TM2	I/O	MFP14	Timer2 事件计数输入/反转输出脚
		INT1	I	MFP15	外部中断1输入脚
21	51	PA.6	I/O	MFP0	通用数字I/O脚
		EBI_AD6	I/O	MFP2	EBI地址/数据总线位6.
		SPI1_SS	I/O	MFP4	SPI1从机选择管脚.
		UART0_RXD	I	MFP7	UART0 数据接收输入脚
		I^2C1_SDA	I/O	MFP8	I^2C1 数据输入/输出脚
		QSPI1_MOSI1	I/O	MFP9	Quad SPI1 MOSI1 (主机输出，从机输入) 管脚.
		EPWM1_CH5	I/O	MFP11	EPWM1 通道5输出/捕获输入
		BPWM1_CH3	I/O	MFP12	BPWM1 通道3输出/捕获输入脚
		ACMP1_WLAT	I	MFP13	模拟比较器 1窗口锁存输入管脚
		TM3	I/O	MFP14	Timer3 事件计数输入/反转输出脚
		INT0	I	MFP15	外部中断0输入管脚
22	52	V_{SS}	P	MFP0	数字电路地管脚
23	53	V_{DD}	P	MFP0	I/O和内部PLL及数字电路的LDO电源
24	54	LDO_CAP	A	MFP0	LDO输出脚
25	55	PA.5	I/O	MFP0	通用数字I/O脚
		QSPI0_MISO1	I/O	MFP3	Quad SPI0 MISO1（主机输入，从机输出）脚.
		SPI1_I2SMCLK	I/O	MFP4	SPI1 I^2S 主机时钟输出脚
		UART0_nCTS	I	MFP7	UART0 禁止发送输入管脚
		UART5_TXD	O	MFP8	UART5数据发送输出脚
		I^2C0_SCL	I/O	MFP9	I2C0 时钟管脚
		CAN0_TXD	O	MFP10	CAN0总线发送输出
		UART0_TXD	O	MFP11	UART0 数据发送输出脚.
		BPWM0_CH5	I/O	MFP12	BPWM0 通道5 输出/捕获输入
		EPWM0_CH0	I/O	MFP13	EPWM0 通道0输出/捕获输入
		QEI0_INDEX	I	MFP14	正交编码器0 索引输入
26	56	PA.4	I/O	MFP0	通用数字I/O脚
		QSPI0_MOSI1	I/O	MFP3	Quad SPI0 MOSI1 (主机输出，从机输入) 管脚.
		SPI0_I2SMCLK	I/O	MFP4	SPI0 I^2S 主机时钟输出脚

64 管脚	128 管脚	管脚名称	类型	MFP	描述
		SC0_nCD	I	MFP6	智能卡 0 卡检测管脚.
		UART0_nRTS	O	MFP7	UART0 请求发送输出脚
		UART5_RXD	I	MFP8	UART5 数据接收输入管脚
		I²C0_SDA	I/O	MFP9	I2C0 数据输入/输出管脚
		CAN0_RXD	I	MFP10	CAN0 总线接收输入
		UART0_RXD	I	MFP11	UART0 数据接收输入脚
		BPWM0_CH4	I/O	MFP12	BPWM0 通道4输出/捕获输入脚
		EPWM0_CH1	I/O	MFP13	EPWM0 通道1输出/捕获输入脚
		QEI0_A	I	MFP14	正交编码器0 相位A输入
27	57	PA.3	I/O	MFP0	通用数字I/O脚
		QSPI0_SS	I/O	MFP3	Quad SPI0从机选择管脚.
		SPI0_SS	I/O	MFP4	SPI0从机选择管脚.
		SC0_PWR	O	MFP6	智能卡0电源脚
		UART4_TXD	O	MFP7	UART4 数据发送输出脚.
		UART1_TXD	O	MFP8	UART1 数据发送输出脚
		I²C1_SCL	I/O	MFP9	I²C1 时钟脚
		I²C0_SMBAL	O	MFP10	I²C0 SMBus SMBALTER 管脚
		BPWM0_CH3	I/O	MFP12	BPWM0 通道3输出/捕获输入脚
		EPWM0_CH2	I/O	MFP13	EPWM0 通道2输出/捕获输入脚
		QEI0_B	I	MFP14	正交编码器0 相位B 输入
		EPWM1_BRAKE1	I	MFP15	EPWM1 刹车1输入管脚
28	58	PA.2	I/O	MFP0	通用数字I/O脚
		QSPI0_CLK	I/O	MFP3	Quad SPI0 串行时钟脚
		SPI0_CLK	I/O	MFP4	SPI0 串行时钟脚
		SC0_RST	O	MFP6	智能卡0复位脚
		UART4_RXD	I	MFP7	UART4 数据接收输入脚
		UART1_RXD	I	MFP8	UART1数据接收输入管脚
		I²C1_SDA	I/O	MFP9	I²C1 数据输入/输出脚
		I²C0_SMBSUS	O	MFP10	I²C0 SMBus SMBSUS 管脚 (PMBus 控制管脚)
		BPWM0_CH2	I/O	MFP12	BPWM0 通道2 输出/捕获输入
		EPWM0_CH3	I/O	MFP13	EPWM0 通道3输出/捕获输入脚

64 管脚	128 管脚	管脚名称	类型	MFP	描述
29	59	PA.1	I/O	MFP0	通用数字I/O脚
		QSPI0_MISO0	I/O	MFP3	Quad SPI0 MISO0（主机输入，从机输出）脚.
		SPI0_MISO	I/O	MFP4	SPI0 MISO（主机输入，从机输出）脚.
		SC0_DAT	I/O	MFP6	智能卡0数据输入脚
		UART0_TXD	O	MFP7	UART0 数据发送输出脚.
		UART1_nCTS	I	MFP8	UART1 禁止发送输入管脚
		I^2C2_SCL	I/O	MFP9	I^2C2 时钟脚
		CAP_DATA7	I	MFP10	图像数据输入总线位7.
		BPWM0_CH1	I/O	MFP12	BPWM0 通道1 输出/捕获输入
		EPWM0_CH4	I/O	MFP13	EPWM0 通道4输出/捕获输入脚
30	60	PA.0	I/O	MFP0	通用数字I/O脚
		QSPI0_MOSI0	I/O	MFP3	Quad SPI0 MOSI0 (主机输出，从机输入) 管脚.
		SPI0_MOSI	I/O	MFP4	SPI0 MOSI (主机输出，从机输入) 管脚.
		SC0_CLK	O	MFP6	智能卡0时钟管脚
		UART0_RXD	I	MFP7	UART0 数据接收输入脚
		UART1_nRTS	O	MFP8	UART1 请求发送输出脚
		I^2C2_SDA	I/O	MFP9	I^2C2 数据输入/输出脚
		CAP_DATA6	I	MFP10	图像数据输入总线位6.
		BPWM0_CH0	I/O	MFP12	BPWM0 通道0 输出/捕获输入
		EPWM0_CH5	I/O	MFP13	EPWM0 通道5 输出/捕获输入
		DAC0_ST	I	MFP15	DAC0 外部触发输入
31	61	V_{DDIO}	P	MFP0	PA.0~PA.5的电源
	62	PE.14	I/O	MFP0	通用数字I/O脚
		EBI_AD8	I/O	MFP2	EBI地址/数据总线位8.
		UART2_TXD	O	MFP3	UART2 数据发送输出脚.
		CAN0_TXD	O	MFP4	CAN0总线发送输出
		UART6_TXD	O	MFP6	UART6 数据发送输出脚.
	63	PE.15	I/O	MFP0	通用数字I/O脚
		EBI_AD9	I/O	MFP2	EBI地址/数据总线位9.
		UART2_RXD	I	MFP3	UART2 数据接收输入脚
		CAN0_RXD	I	MFP4	CAN0 总线接收输入

64 管脚	128 管脚	管脚名称	类型	MFP	描述
		UART6_RXD	I	MFP6	UART6 数据接收输入脚
32	64	nRESET	I	MFP0	外部复位输入:低电平有效,有内部上拉，设置此管脚为低将复位初始状态
33	65	PF.0	I/O	MFP0	通用数字I/O脚
		UART1_TXD	O	MFP2	UART1 数据发送输出脚
		I^2C1_SCL	I/O	MFP3	I^2C1 时钟脚
		UART0_TXD	O	MFP4	UART0 数据发送输出脚.
		BPWM1_CH0	I/O	MFP12	BPWM1 通道0 输出/捕获输入
		ICE_DAT	I/O	MFP14	ADC 外部触发输入
34	66	PF.1	I/O	MFP0	通用数字I/O脚
		UART1_RXD	I	MFP2	UART1数据接收输入管脚
		I^2C1_SDA	I/O	MFP3	I^2C1 数据输入/输出脚
		UART0_RXD	I	MFP4	UART0 数据接收输入脚
		BPWM1_CH1	I/O	MFP12	BPWM1 通道1 输出/捕获输入
		ICE_CLK	I/O	MFP14	ADC 外部触发输入
	67	PD.9	I/O	MFP0	通用数字I/O脚
		EBI_AD7	I/O	MFP2	EBI地址/数据总线位7.
		I^2C2_SCL	I/O	MFP3	I^2C2 时钟脚
		UART2_nCTS	I	MFP4	UART2 禁止发送输入管脚
		UART7_TXD	O	MFP5	UART7 数据发送输出脚.
		CAN2_TXD	O	MFP6	CAN2总线发送输出
	68	PD.8	I/O	MFP0	通用数字I/O脚
		EBI_AD6	I/O	MFP2	EBI地址/数据总线位6.
		I^2C2_SDA	I/O	MFP3	I^2C2 数据输入/输出脚
		UART2_nRTS	O	MFP4	UART2 请求发送输出脚
		UART7_RXD	I	MFP5	UART7 数据接收输入脚
		CAN2_RXD	I	MFP6	CAN2 总线接收输入
35	69	PC.5	I/O	MFP0	通用数字I/O脚
		EBI_AD5	I/O	MFP2	EBI地址/数据总线位5.
		QSPI0_MISO1	I/O	MFP4	Quad SPI0 MISO1（主机输入，从机输出）脚.
		UART2_TXD	O	MFP8	UART2 数据发送输出脚.

64 管脚	128 管脚	管脚名称	类型	MFP	描述
		I²C1_SCL	I/O	MFP9	I²C1 时钟脚
		CAN0_TXD	O	MFP10	CAN0总线发送输出
		UART4_TXD	O	MFP11	UART4 数据发送输出脚.
		EPWM1_CH0	I/O	MFP12	EPWM1 通道0 输出/捕获输入
		CAP_DATA5	I	MFP13	图像数据输入总线位5.
		QSPI1_SS	I/O	MFP14	Quad SPI1从机选择管脚.
36	70	PC.4	I/O	MFP0	通用数字I/O脚
		EBI_AD4	I/O	MFP2	EBI地址/数据总线位4.
		QSPI0_MOSI1	I/O	MFP4	Quad SPI0 MOSI1 (主机输出，从机输入) 管脚.
		I²S0_BCLK	O	MFP6	I2S0位时钟输出管脚
		SPI1_I2SMCLK	I/O	MFP7	SPI1 I²S 主机时钟输出脚
		UART2_RXD	I	MFP8	UART2 数据接收输入脚
		I²C1_SDA	I/O	MFP9	I²C1 数据输入/输出脚
		CAN0_RXD	I	MFP10	CAN0 总线接收输入
		UART4_RXD	I	MFP11	UART4 数据接收输入脚
		EPWM1_CH1	I/O	MFP12	EPWM1 通道1 输出/捕获输入
		CAP_DATA4	I	MFP13	图像数据输入总线位4.
		QSPI1_CLK	I/O	MFP14	Quad SPI1 串行时钟脚
37	71	PC.3	I/O	MFP0	通用数字I/O脚
		EBI_AD3	I/O	MFP2	EBI地址/数据总线位3.
		QSPI0_SS	I/O	MFP4	Quad SPI0从机选择管脚.
		I²S0_MCLK	O	MFP6	I2S0 主时钟输出脚
		SPI1_MISO	I/O	MFP7	SPI1 MISO（主机输入，从机输出）脚.
		UART2_nRTS	O	MFP8	UART2 请求发送输出脚
		I²C0_SMBAL	O	MFP9	I²C0 SMBus SMBALTER 管脚
		CAN1_TXD	O	MFP10	CAN1总线发送输出
		UART3_TXD	O	MFP11	UART3 数据发送输出脚.
		EPWM1_CH2	I/O	MFP12	EPWM1 通道2 输出/捕获输入
		CAP_DATA3	I	MFP13	图像数据输入总线位3.
		QSPI1_MISO0	I/O	MFP14	Quad SPI1 MISO0（主机输入，从机输出）脚.
38	72	PC.2	I/O	MFP0	通用数字I/O脚

64 管脚	128 管脚	管脚名称	类型	MFP	描述
		EBI_AD2	I/O	MFP2	EBI地址/数据总线位2.
		QSPI0_CLK	I/O	MFP4	Quad SPI0 串行时钟脚
		I²S0_DI	I	MFP6	I2S0 数据输入脚
		SPI1_MOSI	I/O	MFP7	SPI1 MOSI (主机输出，从机输入) 管脚.
		UART2_nCTS	I	MFP8	UART2 禁止发送输入管脚
		I²C0_SMBSUS	O	MFP9	I²C0 SMBus SMBSUS 管脚 (PMBus 控制管脚)
		CAN1_RXD	I	MFP10	CAN1 总线接收输入
		UART3_RXD	I	MFP11	UART3 数据接收输入脚
		EPWM1_CH3	I/O	MFP12	EPWM1 通道3输出/捕获输入脚
		CAP_DATA2	I	MFP13	图像数据输入总线位2.
		QSPI1_MOSI0	I/O	MFP14	Quad SPI1 MOSI0 (主机输出，从机输入) 管脚.
39	73	PC.1	I/O	MFP0	通用数字I/O脚
		EBI_AD1	I/O	MFP2	EBI地址/数据总线位1.
		QSPI0_MISO0	I/O	MFP4	Quad SPI0 MISO0（主机输入，从机输出）脚.
		I²S0_DO	O	MFP6	I²S0 数据输出脚
		SPI1_CLK	I/O	MFP7	SPI1 串行时钟脚
		UART2_TXD	O	MFP8	UART2 数据发送输出脚.
		I²C0_SCL	I/O	MFP9	I2C0 时钟管脚
		CAN2_TXD	O	MFP10	CAN2总线发送输出
		EPWM1_CH4	I/O	MFP12	EPWM1 通道4输出/捕获输入脚
		CAP_DATA1	I	MFP13	图像数据输入总线位1.
		ACMP0_O	O	MFP14	模拟比较器 0输出脚
		EADC0_ST	I	MFP15	EADC0 外部触发输入
40	74	PC.0	I/O	MFP0	通用数字I/O脚
		EBI_AD0	I/O	MFP2	EBI地址/数据总线位0.
		QSPI0_MOSI0	I/O	MFP4	Quad SPI0 MOSI0 (主机输出，从机输入) 管脚.
		I²S0_LRCK	O	MFP6	I²S0左右通道时钟输出脚.
		SPI1_SS	I/O	MFP7	SPI1从机选择管脚.
		UART2_RXD	I	MFP8	UART2 数据接收输入脚
		I²C0_SDA	I/O	MFP9	I2C0 数据输入/输出管脚
		CAN2_RXD	I	MFP10	CAN2 总线接收输入

64 管脚	128 管脚	管脚名称	类型	MFP	描述
		EPWM1_CH5	I/O	MFP12	EPWM1 通道5输出/捕获输入
		CAP_DATA0	I	MFP13	图像数据输入总线位0.
		ACMP1_O	O	MFP14	模拟比较器1输出脚
		EADC1_ST	I	MFP15	EADC1 外部触发输入
	75	V_{SS}	P	MFP0	数字电路地管脚
	76	V_{DD}	P	MFP0	I/O和内部PLL及数字电路的LDO电源
	77	PG.9	I/O	MFP0	通用数字I/O脚
		EBI_AD0	I/O	MFP2	EBI地址/数据总线位0.
		QSPI1_MISO1	I/O	MFP5	Quad SPI1 MISO1（主机输入，从机输出）脚.
		CAP_PIXCLK	I	MFP7	图像采集接口像素时钟输入脚
		BPWM0_CH5	I/O	MFP12	BPWM0 通道5 输出/捕获输入
	78	PG.10	I/O	MFP0	通用数字I/O脚
		EBI_AD1	I/O	MFP2	EBI地址/数据总线位1.
		QSPI1_MOSI1	I/O	MFP5	Quad SPI1 MOSI1 (主机输出，从机输入) 管脚.
		CAP_SCLK	O	MFP7	图像采集接口传感器时钟管脚.
		BPWM0_CH4	I/O	MFP12	BPWM0 通道4输出/捕获输入脚
	79	PG.11	I/O	MFP0	通用数字I/O脚
		EBI_AD2	I/O	MFP2	EBI地址/数据总线位2.
		QSPI1_SS	I/O	MFP5	Quad SPI1从机选择管脚.
		UART7_TXD	O	MFP6	UART7 数据发送输出脚.
		CAP_SFIELD	I	MFP7	图像输入接口SFIELD 输入脚
		BPWM0_CH3	I/O	MFP12	BPWM0 通道3输出/捕获输入脚
	80	PG.12	I/O	MFP0	通用数字I/O脚
		EBI_AD3	I/O	MFP2	EBI地址/数据总线位3.
		QSPI1_CLK	I/O	MFP5	Quad SPI1 串行时钟脚
		UART7_RXD	I	MFP6	UART7 数据接收输入脚
		CAP_VSYNC	I	MFP7	图像采集接口vsync 输入脚
		BPWM0_CH2	I/O	MFP12	BPWM0 通道2 输出/捕获输入
	81	PG.13	I/O	MFP0	通用数字I/O脚
		EBI_AD4	I/O	MFP2	EBI地址/数据总线位4.
		QSPI1_MISO0	I/O	MFP5	Quad SPI1 MISO0（主机输入，从机输出）脚.

64 管脚	128 管脚	管脚名称	类型	MFP	描述
		UART6_TXD	O	MFP6	UART6 数据发送输出脚.
		CAP_HSYNC	I	MFP7	图像采集接口hsync 输入脚
		BPWM0_CH1	I/O	MFP12	BPWM0 通道1 输出/捕获输入
	82	PG.14	I/O	MFP0	通用数字I/O脚
		EBI_AD5	I/O	MFP2	EBI地址/数据总线位5.
		QSPI1_MOSI0	I/O	MFP5	Quad SPI1 MOSI0 (主机输出，从机输入) 管脚.
		UART6_RXD	I	MFP6	UART6 数据接收输入脚
		BPWM0_CH0	I/O	MFP12	BPWM0 通道0 输出/捕获输入
	83	PG.15	I/O	MFP0	通用数字I/O脚
		CLKO	O	MFP14	时钟输出
		EADC0_ST	I	MFP15	EADC0 外部触发输入
	84	PD.13	I/O	MFP0	通用数字I/O脚
		EBI_AD10	I/O	MFP2	EBI地址/数据总线位10.
		SD0_nCD	I	MFP3	SD/SDIO0 卡检测输入管脚
		SPI0_I2SMCLK	I/O	MFP4	SPI0 I^2S 主机时钟输出脚
		SPI1_I2SMCLK	I/O	MFP5	SPI1 I^2S 主机时钟输出脚
45	85	PA.12	I/O	MFP0	通用数字I/O脚
		I^2S0_BCLK	O	MFP2	I2S0位时钟输出管脚
		UART4_TXD	O	MFP3	UART4 数据发送输出脚.
		I^2C1_SCL	I/O	MFP4	I^2C1 时钟脚
		SPI2_SS	I/O	MFP5	SPI2从机选择管脚.
		CAN0_TXD	O	MFP6	CAN0总线发送输出
		BPWM1_CH2	I/O	MFP11	BPWM1 通道2 输出/捕获输入
		QEI1_INDEX	I	MFP12	正交编码器1 索引输入
		USB_VBUS	P	MFP14	USB主机或HUB的电源
46	86	PA.13	I/O	MFP0	通用数字I/O脚
		I^2S0_MCLK	O	MFP2	I2S0 主时钟输出脚
		UART4_RXD	I	MFP3	UART4 数据接收输入脚
		I^2C1_SDA	I/O	MFP4	I^2C1 数据输入/输出脚
		SPI2_CLK	I/O	MFP5	SPI2 串行时钟管脚
		CAN0_RXD	I	MFP6	CAN0 总线接收输入

64 管脚	128 管脚	管脚名称	类型	MFP	描述
		BPWM1_CH3	I/O	MFP11	BPWM1 通道3输出/捕获输入脚
		QEI1_A	I	MFP12	正交编码器1 相位A输入
		USB_D-	A	MFP14	USB差分信号D-.
47	87	PA.14	I/O	MFP0	通用数字I/O脚
		I^2S0_DI	I	MFP2	I2S0 数据输入脚
		UART0_TXD	O	MFP3	UART0 数据发送输出脚.
		SPI2_MISO	I/O	MFP5	SPI2 MISO(主机输入，从机输出)管脚
		I^2C2_SCL	I/O	MFP6	I^2C2 时钟脚
		BPWM1_CH4	I/O	MFP11	BPWM1 通道4输出/捕获输入脚
		QEI1_B	I	MFP12	正交编码器1 相位B 输入
		USB_D+	A	MFP14	USB差分信号D+.
48	88	PA.15	I/O	MFP0	通用数字I/O脚
		I^2S0_DO	O	MFP2	I^2S0 数据输出脚
		UART0_RXD	I	MFP3	UART0 数据接收输入脚
		SPI2_MOSI	I/O	MFP5	SPI2 MOSI (主机输出，从机输入) 管脚
		I^2C2_SDA	I/O	MFP6	I^2C2 数据输入/输出脚
		BPWM1_CH5	I/O	MFP11	EPWM1 通道5输出/捕获输入
		EPWM0_SYNC_IN	I	MFP12	EPWM0 计数器同步触发输入脚
		USB_OTG_ID	I	MFP14	USB_ 标识
	89	NC	-	MFP0	没用管脚，浮空
	90	NC	-	MFP0	没用管脚，浮空
	91	NC	-	MFP0	没用管脚，浮空
	92	NC	-	MFP0	没用管脚，浮空
	93	NC	-	MFP0	没用管脚，浮空
	94	NC	-	MFP0	没用管脚，浮空
	95	NC	-	MFP0	没用管脚，浮空
	96	NC	-	MFP0	没用管脚，浮空
	97	PE.7	I/O	MFP0	通用数字I/O脚
		SD0_CMD	I/O	MFP3	SD/SDIO0 命令/响应脚
		UART5_TXD	O	MFP8	UART5数据发送输出脚
		CAN1_TXD	O	MFP9	CAN1总线发送输出

64 管脚	128 管脚	管脚名称	类型	MFP	描述
		QEI1_INDEX	I	MFP11	正交编码器1 索引输入
		EPWM0_CH0	I/O	MFP12	EPWM0 通道0输出/捕获输入
		BPWM0_CH5	I/O	MFP13	BPWM0 通道5 输出/捕获输入
	98	PE.6	I/O	MFP0	通用数字I/O脚
		SD0_CLK	O	MFP3	SD/SDIO0 时钟输出脚
		SC0_nCD	I	MFP6	智能卡 0 卡检测管脚.
		UART5_RXD	I	MFP8	UART5 数据接收输入管脚
		CAN1_RXD	I	MFP9	CAN1 总线接收输入
		QEI1_A	I	MFP11	正交编码器1 相位A输入
		EPWM0_CH1	I/O	MFP12	EPWM0 通道1输出/捕获输入脚
		BPWM0_CH4	I/O	MFP13	BPWM0 通道4输出/捕获输入脚
	99	PE.5	I/O	MFP0	通用数字I/O脚
		EBI_nRD	O	MFP2	EBI 读使能输出脚
		SD0_DAT3	I/O	MFP3	SD/SDIO0数据线位3
		SC0_PWR	O	MFP6	智能卡0电源脚
		UART6_TXD	O	MFP8	UART6 数据发送输出脚.
		UART7_nRTS	O	MFP9	UART7 请求发送输出脚
		QEI1_B	I	MFP11	正交编码器1 相位B 输入
		EPWM0_CH2	I/O	MFP12	EPWM0 通道2输出/捕获输入脚
		BPWM0_CH3	I/O	MFP13	BPWM0 通道3输出/捕获输入脚
	100	PE.4	I/O	MFP0	通用数字I/O脚
		EBI_nWR	O	MFP2	EBI 写使能输出脚
		SD0_DAT2	I/O	MFP3	SD/SDIO0 数据线位2
		SC0_RST	O	MFP6	智能卡0复位脚
		UART6_RXD	I	MFP8	UART6 数据接收输入脚
		UART7_nCTS	I	MFP9	UART7 禁止发送输入管脚
		QEI0_INDEX	I	MFP11	正交编码器0 索引输入
		EPWM0_CH3	I/O	MFP12	EPWM0 通道3输出/捕获输入脚
		BPWM0_CH2	I/O	MFP13	BPWM0 通道2 输出/捕获输入
	101	PE.3	I/O	MFP0	通用数字I/O脚
		EBI_MCLK	O	MFP2	EBI 外部时钟输出脚.

64 管脚	128 管脚	管脚名称	类型	MFP	描述
		SD0_DAT1	I/O	MFP3	SD/SDIO0 数据线位1
		SC0_DAT	I/O	MFP6	智能卡0数据输入脚
		UART6_nRTS	O	MFP8	UART6 请求发送输出脚
		UART7_TXD	O	MFP9	UART7 数据发送输出脚.
		QEI0_A	I	MFP11	正交编码器0 相位A输入
		EPWM0_CH4	I/O	MFP12	EPWM0 通道4输出/捕获输入脚
		BPWM0_CH1	I/O	MFP13	BPWM0 通道1 输出/捕获输入
	102	PE.2	I/O	MFP0	通用数字I/O脚
		EBI_ALE	O	MFP2	EBI地址锁存使能输出脚
		SD0_DAT0	I/O	MFP3	SD/SDIO0 数据线位0
		SC0_CLK	O	MFP6	智能卡0时钟管脚
		UART6_nCTS	I	MFP8	UART6 禁止发送输入管脚
		UART7_RXD	I	MFP9	UART7 数据接收输入脚
		QEI0_B	I	MFP11	正交编码器0 相位B 输入
		EPWM0_CH5	I/O	MFP12	EPWM0 通道5 输出/捕获输入
		BPWM0_CH0	I/O	MFP13	BPWM0 通道0 输出/捕获输入
	103	V_{SS}	P	MFP0	数字电路地管脚
	104	V_{DD}	P	MFP0	I/O和内部PLL及数字电路的LDO电源
	105	PE.1	I/O	MFP0	通用数字I/O脚
		EBI_AD10	I/O	MFP2	EBI地址/数据总线位10.
		QSPI0_MISO0	I/O	MFP3	Quad SPI0 MISO0（主机输入，从机输出）脚.
		I^2S0_BCLK	O	MFP5	I2S0位时钟输出管脚
		SPI1_MISO	I/O	MFP6	SPI1 MISO（主机输入，从机输出）脚.
		UART3_TXD	O	MFP7	UART3 数据发送输出脚.
		I^2C1_SCL	I/O	MFP8	I^2C1 时钟脚
		UART4_nCTS	I	MFP9	UART4 禁止发送输入管脚
	106	PE.0	I/O	MFP0	通用数字I/O脚
		EBI_AD11	I/O	MFP2	EBI地址/数据总线位11.
		QSPI0_MOSI0	I/O	MFP3	Quad SPI0 MOSI0 (主机输出，从机输入) 管脚.
		I^2S0_MCLK	O	MFP5	I2S0 主时钟输出脚
		SPI1_MOSI	I/O	MFP6	SPI1 MOSI (主机输出，从机输入) 管脚.

64 管脚	128 管脚	管脚名称	类型	MFP	描述
		UART3_RXD	I	MFP7	UART3 数据接收输入脚
		I²C1_SDA	I/O	MFP8	I²C1 数据输入/输出脚
		UART4_nRTS	O	MFP9	UART4 请求发送输出脚
	107	PH.8	I/O	MFP0	通用数字I/O脚
		EBI_AD12	I/O	MFP2	EBI地址/数据总线位12.
		QSPI0_CLK	I/O	MFP3	Quad SPI0 串行时钟脚
		I²S0_DI	I	MFP5	I2S0 数据输入脚
		SPI1_CLK	I/O	MFP6	SPI1 串行时钟脚
		UART3_nRTS	O	MFP7	UART3 请求发送输出脚
		I²C1_SMBAL	O	MFP8	I²C1 SMBus SMBALTER 管脚
		I²C2_SCL	I/O	MFP9	I²C2 时钟脚
		UART1_TXD	O	MFP10	UART1 数据发送输出脚
	108	PH.9	I/O	MFP0	通用数字I/O脚
		EBI_AD13	I/O	MFP2	EBI地址/数据总线位13.
		QSPI0_SS	I/O	MFP3	Quad SPI0从机选择管脚.
		I²S0_DO	O	MFP5	I²S0 数据输出脚
		SPI1_SS	I/O	MFP6	SPI1从机选择管脚.
		UART3_nCTS	I	MFP7	UART3 禁止发送输入管脚
		I²C1_SMBSUS	O	MFP8	I²C1 SMBus SMBSUS 管脚 (PMBus 控制管脚)
		I²C2_SDA	I/O	MFP9	I²C2 数据输入/输出脚
		UART1_RXD	I	MFP10	UART1数据接收输入管脚
	109	PH.10	I/O	MFP0	通用数字I/O脚
		EBI_AD14	I/O	MFP2	EBI地址/数据总线位14.
		QSPI0_MISO1	I/O	MFP3	Quad SPI0 MISO1（主机输入，从机输出）脚.
		I²S0_LRCK	O	MFP5	I²S0左右通道时钟输出脚.
		SPI1_I2SMCLK	I/O	MFP6	SPI1 I²S 主机时钟输出脚
		UART4_TXD	O	MFP7	UART4 数据发送输出脚.
		UART0_TXD	O	MFP8	UART0 数据发送输出脚.
	110	PH.11	I/O	MFP0	通用数字I/O脚
		EBI_AD15	I/O	MFP2	EBI地址/数据总线位15.
		QSPI0_MOSI1	I/O	MFP3	Quad SPI0 MOSI1 (主机输出，从机输入) 管脚.

64 管脚	128 管脚	管脚名称	类型	MFP	描述
		UART4_RXD	I	MFP7	UART4 数据接收输入脚
		UART0_RXD	I	MFP8	UART0 数据接收输入脚
		EPWM0_CH5	I/O	MFP11	EPWM0 通道5 输出/捕获输入
	111	PD.14	I/O	MFP0	通用数字I/O脚
		EBI_nCS0	O	MFP2	EBI片选0输出脚
		SPI0_I2SMCLK	I/O	MFP5	SPI0 I^2S 主机时钟输出脚
		EPWM0_CH4	I/O	MFP11	EPWM0 通道4输出/捕获输入脚
49	112	V_{SS}	P	MFP0	数字电路地管脚
50	113	LDO_CAP	A	MFP0	LDO输出脚
51	114	V_{DD}	P	MFP0	I/O和内部PLL及数字电路的LDO电源
52	115	PC.14	I/O	MFP0	通用数字I/O脚
		EBI_AD11	I/O	MFP2	EBI地址/数据总线位11.
		SPI0_I2SMCLK	I/O	MFP4	SPI0 I^2S 主机时钟输出脚
		QSPI0_CLK	I/O	MFP6	Quad SPI0 串行时钟脚
		EPWM0_SYNC_IN	I	MFP11	EPWM0 计数器同步触发输入脚
		ETM_TRACE_CLK	I	MFP12	ETM接收追踪时钟输入管脚
		TM1	I/O	MFP13	Timer1 事件计数输入/反转输出脚
		USB_VBUS_ST	I	MFP14	USB 外部VBUS 电源状态管脚.
53	116	PB.15	I/O	MFP0	通用数字I/O脚
		EADC0_CH15	A	MFP1	EADC0 通道15 模拟输入
		EADC1_CH15	A	MFP1	EADC1 通道15 模拟输入
		EBI_AD12	I/O	MFP2	EBI地址/数据总线位12.
		SPI0_SS	I/O	MFP4	SPI0从机选择管脚.
		UART0_nCTS	I	MFP6	UART0 禁止发送输入管脚
		UART3_TXD	O	MFP7	UART3 数据发送输出脚.
		I^2C2_SMBAL	O	MFP8	I^2C2 SMBus SMBALTER 管脚
		EPWM0_BRAKE1	I	MFP10	EPWM0 刹车1 输入脚
		EPWM1_CH0	I/O	MFP11	EPWM1 通道0 输出/捕获输入
		ETM_TRACE_DATA0	I	MFP12	ETM接收追踪数据0 输入管脚
		TM0_EXT	I/O	MFP13	Timer0 外部捕获输入/反转输出脚
		USB_VBUS_EN	O	MFP14	USB 外部 VBUS 电源使能管脚.

64 管脚	128 管脚	管脚名称	类型	MFP	描述
54	117	PB.14	I/O	MFP0	通用数字I/O脚
		EADC0_CH14	A	MFP1	EADC0 通道14 模拟输入
		EADC1_CH14	A	MFP1	EADC1 通道14 模拟输入
		EBI_AD13	I/O	MFP2	EBI地址/数据总线位13.
		SPI0_CLK	I/O	MFP4	SPI0 串行时钟脚
		UART0_nRTS	O	MFP6	UART0 请求发送输出脚
		UART3_RXD	I	MFP7	UART3 数据接收输入脚
		I^2C2_SMBSUS	O	MFP8	I^2C2 SMBus SMBSUS 管脚 (PMBus 控制管脚)
		EPWM1_CH1	I/O	MFP11	EPWM1 通道1 输出/捕获输入
		ETM_TRACE_DATA1	I	MFP12	ETM接收追踪数据1 输入管脚
		TM1_EXT	I/O	MFP13	Timer1 外部捕获输入/反转输出脚
		CLKO	O	MFP14	时钟输出
		USB_VBUS_ST	I	MFP15	USB 外部VBUS 电源状态管脚.
55	118	PB.13	I/O	MFP0	通用数字I/O脚
		EADC0_CH13	A	MFP1	EADC0 通道13 模拟输入
		EADC1_CH13	A	MFP1	EADC1 通道13 模拟输入
		ACMP0_P3	A	MFP1	模拟比较器 0 正极输入 3 管脚.
		ACMP1_P3	A	MFP1	模拟比较器 1 正极输入 3 管脚.
		EBI_AD14	I/O	MFP2	EBI地址/数据总线位14.
		SPI0_MISO	I/O	MFP4	SPI0 MISO（主机输入，从机输出）脚.
		UART0_TXD	O	MFP6	UART0 数据发送输出脚.
		UART3_nRTS	O	MFP7	UART3 请求发送输出脚
		I^2C2_SCL	I/O	MFP8	I^2C2 时钟脚
		CAP_PIXCLK	I	MFP10	图像采集接口像素时钟输入脚
		EPWM1_CH2	I/O	MFP11	EPWM1 通道2 输出/捕获输入
		ETM_TRACE_DATA2	I	MFP12	ETM接收追踪数据2 输入管脚
		TM2_EXT	I/O	MFP13	Timer2 外部捕获输入/反转输出脚
56	119	PB.12	I/O	MFP0	通用数字I/O脚
		EADC0_CH12	A	MFP1	EADC0 通道12 模拟输入
		EADC1_CH12	A	MFP1	EADC1 通道12 模拟输入
		DAC0_OUT	A	MFP1	DAC0 模拟输出通道

64 管脚	128 管脚	管脚名称	类型	MFP	描述
		ACMP0_P2	A	MFP1	模拟比较器 0 正极输入 2 管脚.
		ACMP1_P2	A	MFP1	模拟比较器 1 正极输入 2 管脚.
		EBI_AD15	I/O	MFP2	EBI地址/数据总线位15.
		SPI0_MOSI	I/O	MFP4	SPI0 MOSI (主机输出，从机输入) 管脚.
		UART0_RXD	I	MFP6	UART0 数据接收输入脚
		UART3_nCTS	I	MFP7	UART3 禁止发送输入管脚
		I^2C2_SDA	I/O	MFP8	I^2C2 数据输入/输出脚
		SD0_nCD	I	MFP9	SD/SDIO0 卡检测输入管脚
		CAP_SCLK	O	MFP10	图像采集接口传感器时钟管脚.
		EPWM1_CH3	I/O	MFP11	EPWM1 通道3输出/捕获输入脚
		ETM_TRACE_DATA3	I	MFP12	ETM接收追踪数据3 输入管脚
		TM3_EXT	I/O	MFP13	Timer3 外部捕获输入/反转输出脚
57	120	AV_{DD}	P	MFP0	内部模拟电路电源
58	121	V_{REF}	A	MFP0	ADC参考电压输入. 注: 此管脚需接1uF电容到地.
59	122	AV_{SS}	P	MFP0	模拟电路地管脚
60	123	PB.11	I/O	MFP0	通用数字I/O脚
		EADC0_CH11	A	MFP1	EADC0 通道11 模拟输入
		EBI_ADR16	O	MFP2	EBI 地址总线位 16.
		UART0_nCTS	I	MFP5	UART0 禁止发送输入管脚
		UART4_TXD	O	MFP6	UART4 数据发送输出脚.
		I^2C1_SCL	I/O	MFP7	I^2C1 时钟脚
		CAN0_TXD	O	MFP8	CAN0总线发送输出
		SPI0_I2SMCLK	I/O	MFP9	SPI0 I^2S 主机时钟输出脚
		BPWM1_CH0	I/O	MFP10	BPWM1 通道0 输出/捕获输入
		CAP_SFIELD	I	MFP12	图像输入接口SFIELD 输入脚
61	124	PB.10	I/O	MFP0	通用数字I/O脚
		EADC0_CH10	A	MFP1	EADC0 通道10 模拟输入
		EBI_ADR17	O	MFP2	EBI 地址总线位 17.
		UART0_nRTS	O	MFP5	UART0 请求发送输出脚
		UART4_RXD	I	MFP6	UART4 数据接收输入脚

64 管脚	128 管脚	管脚名称	类型	MFP	描述
		I²C1_SDA	I/O	MFP7	I²C1 数据输入/输出脚
		CAN0_RXD	I	MFP8	CAN0 总线接收输入
		BPWM1_CH1	I/O	MFP10	BPWM1 通道1 输出/捕获输入
		CAP_VSYNC	I	MFP12	图像采集接口vsync 输入脚
62	125	PB.9	I/O	MFP0	通用数字I/O脚
		EADC0_CH9	A	MFP1	EADC0 通道9 模拟输入
		EBI_ADR18	O	MFP2	EBI 地址总线位 18.
		UART0_TXD	O	MFP5	UART0 数据发送输出脚.
		UART1_nCTS	I	MFP6	UART1 禁止发送输入管脚
		I²C1_SMBAL	O	MFP7	I²C1 SMBus SMBALTER 管脚
		UART7_TXD	O	MFP8	UART7 数据发送输出脚.
		I²C0_SCL	I/O	MFP9	I2C0 时钟管脚
		BPWM1_CH2	I/O	MFP10	BPWM1 通道2 输出/捕获输入
		CAN2_TXD	O	MFP12	CAN2总线发送输出
		INT7	I	MFP13	外部中断7 输入脚
		CAP_HSYNC	I	MFP14	图像采集接口hsync 输入脚
63	126	PB.8	I/O	MFP0	通用数字I/O脚
		EADC0_CH8	A	MFP1	EADC0 通道8 模拟输入
		EBI_ADR19	O	MFP2	EBI 地址总线位 19.
		UART0_RXD	I	MFP5	UART0 数据接收输入脚
		UART1_nRTS	O	MFP6	UART1 请求发送输出脚
		I²C1_SMBSUS	O	MFP7	I²C1 SMBus SMBSUS 管脚 (PMBus 控制管脚)
		UART7_RXD	I	MFP8	UART7 数据接收输入脚
		I²C0_SDA	I/O	MFP9	I2C0 数据输入/输出管脚
		BPWM1_CH3	I/O	MFP10	BPWM1 通道3输出/捕获输入脚
		CAN2_RXD	I	MFP12	CAN2 总线接收输入
		INT6	I	MFP13	外部中断6 输入脚
64	127	PB.7	I/O	MFP0	通用数字I/O脚
		EADC0_CH7	A	MFP1	EADC0 通道7 模拟输入
		EBI_nWRL	O	MFP2	EBI低字节写使能输出脚
		CAN1_TXD	O	MFP5	CAN1总线发送输出

64 管脚	128 管脚	管脚名称	类型	MFP	描述
		UART1_TXD	O	MFP6	UART1 数据发送输出脚
		EBI_nCS0	O	MFP8	EBI片选0输出脚
		BPWM1_CH4	I/O	MFP10	BPWM1 通道4输出/捕获输入脚
		EPWM1_BRAKE0	I	MFP11	EPWM1 刹车0 输入脚
		EPWM1_CH4	I/O	MFP12	EPWM1 通道4输出/捕获输入脚
		INT5	I	MFP13	外部中断5 输入脚
		USB_VBUS_ST	I	MFP14	USB 外部VBUS 电源状态管脚.
		ACMP0_O	O	MFP15	模拟比较器 0输出脚
1	128	PB.6	I/O	MFP0	通用数字I/O脚
		EADC0_CH6	A	MFP1	EADC0通道6模拟输入
		EBI_nWRH	O	MFP2	EBI 高字节写使能输出管脚
		CAN1_RXD	I	MFP5	CAN1 总线接收输入
		UART1_RXD	I	MFP6	UART1数据接收输入管脚
		EBI_nCS1	O	MFP8	EBI片选1输出管脚
		BPWM1_CH5	I/O	MFP10	EPWM1 通道5输出/捕获输入
		EPWM1_BRAKE1	I	MFP11	EPWM1 刹车1输入管脚
		EPWM1_CH5	I/O	MFP12	EPWM1 通道5输出/捕获输入
		INT4	I	MFP13	外部中断4输入管脚
		USB_VBUS_EN	O	MFP14	USB 外部 VBUS 电源使能管脚.
		ACMP1_O	O	MFP15	模拟比较器1输出脚

4.3.4 多功能管脚表

组	管脚名称	GPIO	MFP	类型	描述
ACMP0	ACMP0_N	PB.3	MFP1	A	模拟比较器 0 负极输入管脚.
	ACMP0_O	PC.12	MFP14	O	模拟比较器 0输出脚
		PC.1	MFP14	O	
		PB.7	MFP15	O	
	ACMP0_P0	PA.11	MFP1	A	模拟比较器 0 正极输入脚0
	ACMP0_P1	PB.2	MFP1	A	模拟比较器0正极输入1脚
	ACMP0_P2	PB.12	MFP1	A	模拟比较器 0 正极输入 2 管脚.
	ACMP0_P3	PB.13	MFP1	A	模拟比较器 0 正极输入 3 管脚.
	ACMP0_WLAT	PA.7	MFP13	I	模拟比较器 0窗口锁存输入管脚
ACMP1	ACMP1_N	PB.5	MFP1	A	模拟比较器 1 负极输入管脚.
	ACMP1_O	PC.11	MFP14	O	模拟比较器1输出脚
		PC.0	MFP14	O	
		PB.6	MFP15	O	
	ACMP1_P0	PA.10	MFP1	A	模拟比较器 1 正极输入脚0
	ACMP1_P1	PB.4	MFP1	A	模拟比较器1正极输入1脚
	ACMP1_P2	PB.12	MFP1	A	模拟比较器 1 正极输入 2 管脚.
	ACMP1_P3	PB.13	MFP1	A	模拟比较器 1 正极输入 3 管脚.
	ACMP1_WLAT	PA.6	MFP13	I	模拟比较器 1窗口锁存输入管脚
BPWM0	BPWM0_CH0	PA.11	MFP9	I/O	BPWM0 通道0 输出/捕获输入
		PA.0	MFP12	I/O	
		PG.14	MFP12	I/O	
		PE.2	MFP13	I/O	
	BPWM0_CH1	PA.10	MFP9	I/O	BPWM0 通道1 输出/捕获输入
		PA.1	MFP12	I/O	
		PG.13	MFP12	I/O	
		PE.3	MFP13	I/O	
	BPWM0_CH2	PA.9	MFP9	I/O	BPWM0 通道2 输出/捕获输入
		PA.2	MFP12	I/O	
		PG.12	MFP12	I/O	
		PE.4	MFP13	I/O	
	BPWM0_CH3	PA.8	MFP9	I/O	BPWM0 通道3输出/捕获输入脚

组	管脚名称	GPIO	MFP	类型	描述
		PA.3	MFP12	I/O	
		PG.11	MFP12	I/O	
		PE.5	MFP13	I/O	
	BPWM0_CH4	PC.13	MFP9	I/O	BPWM0 通道4输出/捕获输入脚
		PF.5	MFP8	I/O	
		PA.4	MFP12	I/O	
		PG.10	MFP12	I/O	
		PE.6	MFP13	I/O	
	BPWM0_CH5	PD.12	MFP9	I/O	BPWM0 通道5 输出/捕获输入
		PF.4	MFP8	I/O	
		PA.5	MFP12	I/O	
		PG.9	MFP12	I/O	
		PE.7	MFP13	I/O	
BPWM1	BPWM1_CH0	PF.3	MFP11	I/O	BPWM1 通道0 输出/捕获输入
		PC.7	MFP12	I/O	
		PF.0	MFP12	I/O	
		PB.11	MFP10	I/O	
	BPWM1_CH1	PF.2	MFP11	I/O	BPWM1 通道1 输出/捕获输入
		PC.6	MFP12	I/O	
		PF.1	MFP12	I/O	
		PB.10	MFP10	I/O	
	BPWM1_CH2	PA.7	MFP12	I/O	BPWM1 通道2 输出/捕获输入
		PA.12	MFP11	I/O	
		PB.9	MFP10	I/O	
	BPWM1_CH3	PA.6	MFP12	I/O	BPWM1 通道3输出/捕获输入脚
		PA.13	MFP11	I/O	
		PB.8	MFP10	I/O	
	BPWM1_CH4	PC.8	MFP12	I/O	BPWM1 通道4输出/捕获输入脚
		PA.14	MFP11	I/O	
		PB.7	MFP10	I/O	
	BPWM1_CH5	PE.13	MFP12	I/O	EPWM1 通道5输出/捕获输入
		PA.15	MFP11	I/O	
		PB.6	MFP10	I/O	

组	管脚名称	GPIO	MFP	类型	描述
CAN0	CAN0_RXD	PD.10	MFP4	I	CAN0 总线接收输入
		PA.4	MFP10	I	
		PE.15	MFP4	I	
		PC.4	MFP10	I	
		PA.13	MFP6	I	
		PB.10	MFP8	I	
	CAN0_TXD	PD.11	MFP4	O	CAN0总线发送输出
		PA.5	MFP10	O	
		PE.14	MFP4	O	
		PC.5	MFP10	O	
		PA.12	MFP6	O	
		PB.11	MFP8	O	
CAN1	CAN1_RXD	PC.9	MFP9	I	CAN1 总线接收输入
		PD.12	MFP5	I	
		PF.8	MFP8	I	
		PC.2	MFP10	I	
		PE.6	MFP9	I	
		PB.6	MFP5	I	
	CAN1_TXD	PC.10	MFP9	O	CAN1总线发送输出
		PC.13	MFP5	O	
		PF.9	MFP8	O	
		PC.3	MFP10	O	
		PE.7	MFP9	O	
		PB.7	MFP5	O	
CAN2	CAN2_RXD	PF.6	MFP8	I	CAN2 总线接收输入
		PD.8	MFP6	I	
		PC.0	MFP10	I	
		PB.8	MFP12	I	
	CAN2_TXD	PF.7	MFP8	O	CAN2总线发送输出
		PD.9	MFP6	O	
		PC.1	MFP10	O	
		PB.9	MFP12	O	
CAP	CAP_DATA0	PF.7	MFP7	I	图像数据输入总线位0.

组	管脚名称	GPIO	MFP	类型	描述
		PC.0	MFP13	I	
	CAP_DATA1	PF.8	MFP7	I	图像数据输入总线位1.
		PC.1	MFP13	I	
	CAP_DATA2	PF.9	MFP7	I	图像数据输入总线位2.
		PC.2	MFP13	I	
	CAP_DATA3	PF.10	MFP7	I	图像数据输入总线位3.
		PC.3	MFP13	I	
	CAP_DATA4	PF.11	MFP7	I	图像数据输入总线位4.
		PC.4	MFP13	I	
	CAP_DATA5	PG.4	MFP7	I	图像数据输入总线位5.
		PC.5	MFP13	I	
	CAP_DATA6	PG.3	MFP7	I	图像数据输入总线位6.
		PA.0	MFP10	I	
	CAP_DATA7	PG.2	MFP7	I	图像数据输入总线位7.
		PA.1	MFP10	I	
	CAP_HSYNC	PG.13	MFP7	I	图像采集接口hsync 输入脚
		PB.9	MFP14	I	
	CAP_PIXCLK	PG.9	MFP7	I	图像采集接口像素时钟输入脚
		PB.13	MFP10	I	
	CAP_SCLK	PG.10	MFP7	O	图像采集接口传感器时钟管脚.
		PB.12	MFP10	O	
	CAP_SFIELD	PG.11	MFP7	I	图像输入接口SFIELD 输入脚
		PB.11	MFP12	I	
	CAP_VSYNC	PG.12	MFP7	I	图像采集接口vsync 输入脚
		PB.10	MFP12	I	
CLKO	CLKO	PC.13	MFP13	O	时钟输出
		PD.12	MFP13	O	
		PG.15	MFP14	O	
		PB.14	MFP14	O	
DAC0	DAC0_OUT	PB.12	MFP1	A	DAC0 模拟输出通道
	DAC0_ST	PA.10	MFP14	I	DAC0 外部触发输入
		PA.0	MFP15	I	
EADC0	EADC0_CH0	PB.0	MFP1	A	EADC0 通道0 模拟输入

组	管脚名称	GPIO	MFP	类型	描述
	EADC0_CH1	PB.1	MFP1	A	EADC0 通道1 模拟输入
	EADC0_CH2	PB.2	MFP1	A	EADC0 通道2 模拟输入
	EADC0_CH3	PB.3	MFP1	A	EADC0通道3模拟输入
	EADC0_CH4	PB.4	MFP1	A	EADC0通道4模拟输入
	EADC0_CH5	PB.5	MFP1	A	EADC0通道5模拟输入
	EADC0_CH6	PB.6	MFP1	A	EADC0通道6模拟输入
	EADC0_CH7	PB.7	MFP1	A	EADC0 通道7 模拟输入
	EADC0_CH8	PB.8	MFP1	A	EADC0 通道8 模拟输入
	EADC0_CH9	PB.9	MFP1	A	EADC0 通道9 模拟输入
	EADC0_CH10	PB.10	MFP1	A	EADC0 通道10 模拟输入
	EADC0_CH11	PB.11	MFP1	A	EADC0 通道11 模拟输入
	EADC0_CH12	PB.12	MFP1	A	EADC0 通道12 模拟输入
	EADC0_CH13	PB.13	MFP1	A	EADC0 通道13 模拟输入
	EADC0_CH14	PB.14	MFP1	A	EADC0 通道14 模拟输入
	EADC0_CH15	PB.15	MFP1	A	EADC0 通道15 模拟输入
	EADC0_ST	PC.13	MFP14	I	EADC0 外部触发输入
		PD.12	MFP14	I	
		PF.5	MFP11	I	
		PC.1	MFP15	I	
		PG.15	MFP15	I	
EADC1	EADC1_CH0	PD.10	MFP1	A	EADC1 通道0 模拟输入
	EADC1_CH1	PD.11	MFP1	A	EADC1 通道1 模拟输入
	EADC1_CH2	PD.12	MFP1	A	EADC1 通道2 模拟输入
	EADC1_CH3	PC.13	MFP1	A	EADC1 通道3 模拟输入
	EADC1_CH4	PA.8	MFP1	A	EADC1 通道4 模拟输入
	EADC1_CH5	PA.9	MFP1	A	EADC1 通道5 模拟输入
	EADC1_CH6	PA.10	MFP1	A	EADC1 通道6 模拟输入
	EADC1_CH7	PA.11	MFP1	A	EADC1 通道7 模拟输入
	EADC1_CH8	PB.0	MFP1	A	EADC1 通道8 模拟输入
	EADC1_CH9	PB.1	MFP1	A	EADC1 通道9 模拟输入
	EADC1_CH10	PB.2	MFP1	A	EADC1 通道10 模拟输入
	EADC1_CH11	PB.3	MFP1	A	EADC1 通道11 模拟输入

组	管脚名称	GPIO	MFP	类型	描述
	EADC1_CH12	PB.12	MFP1	A	EADC1 通道12 模拟输入
	EADC1_CH13	PB.13	MFP1	A	EADC1 通道13 模拟输入
	EADC1_CH14	PB.14	MFP1	A	EADC1 通道14 模拟输入
	EADC1_CH15	PB.15	MFP1	A	EADC1 通道15 模拟输入
	EADC1_ST	PC.10	MFP14	I	EADC1 外部触发输入
		PC.9	MFP14	I	
		PF.4	MFP11	I	
		PC.0	MFP15	I	
EBI	EBI_AD0	PC.0	MFP2	I/O	EBI地址/数据总线位0.
		PG.9	MFP2	I/O	
	EBI_AD1	PC.1	MFP2	I/O	EBI地址/数据总线位1.
		PG.10	MFP2	I/O	
	EBI_AD2	PC.2	MFP2	I/O	EBI地址/数据总线位2.
		PG.11	MFP2	I/O	
	EBI_AD3	PC.3	MFP2	I/O	EBI地址/数据总线位3.
		PG.12	MFP2	I/O	
	EBI_AD4	PC.4	MFP2	I/O	EBI地址/数据总线位4.
		PG.13	MFP2	I/O	
	EBI_AD5	PC.5	MFP2	I/O	EBI地址/数据总线位5.
		PG.14	MFP2	I/O	
	EBI_AD6	PA.6	MFP2	I/O	EBI地址/数据总线位6.
		PD.8	MFP2	I/O	
	EBI_AD7	PA.7	MFP2	I/O	EBI地址/数据总线位7.
		PD.9	MFP2	I/O	
	EBI_AD8	PC.6	MFP2	I/O	EBI地址/数据总线位8.
		PE.14	MFP2	I/O	
	EBI_AD9	PC.7	MFP2	I/O	EBI地址/数据总线位9.
		PE.15	MFP2	I/O	
	EBI_AD10	PD.3	MFP2	I/O	EBI地址/数据总线位10.
		PD.13	MFP2	I/O	
		PE.1	MFP2	I/O	
	EBI_AD11	PD.2	MFP2	I/O	EBI地址/数据总线位11.
		PE.0	MFP2	I/O	

组	管脚名称	GPIO	MFP	类型	描述
		PC.14	MFP2	I/O	
	EBI_AD12	PD.1	MFP2	I/O	EBI地址/数据总线位12.
		PH.8	MFP2	I/O	
		PB.15	MFP2	I/O	
	EBI_AD13	PD.0	MFP2	I/O	EBI地址/数据总线位13.
		PH.9	MFP2	I/O	
		PB.14	MFP2	I/O	
	EBI_AD14	PH.10	MFP2	I/O	EBI地址/数据总线位14.
		PB.13	MFP2	I/O	
	EBI_AD15	PH.11	MFP2	I/O	EBI地址/数据总线位15.
		PB.12	MFP2	I/O	
	EBI_ADR0	PB.5	MFP2	O	EBI地址总线位0
		PH.7	MFP2	O	
	EBI_ADR1	PB.4	MFP2	O	EBI 地址总线位1
		PH.6	MFP2	O	
	EBI_ADR2	PB.3	MFP2	O	EBI 地址总线位2
		PH.5	MFP2	O	
	EBI_ADR3	PB.2	MFP2	O	EBI 地址总线位3
		PH.4	MFP2	O	
	EBI_ADR4	PC.12	MFP2	O	EBI 地址总线位 4.
	EBI_ADR5	PC.11	MFP2	O	EBI 地址总线位 5.
	EBI_ADR6	PC.10	MFP2	O	EBI 地址总线位 6.
	EBI_ADR7	PC.9	MFP2	O	EBI 地址总线位 7.
	EBI_ADR8	PB.1	MFP2	O	EBI 地址总线位 8.
	EBI_ADR9	PB.0	MFP2	O	EBI 地址总线位 9.
	EBI_ADR10	PC.13	MFP2	O	EBI 地址总线位 10.
		PE.8	MFP2	O	
	EBI_ADR11	PG.2	MFP2	O	EBI 地址总线位 11.
		PE.9	MFP2	O	
	EBI_ADR12	PG.3	MFP2	O	EBI 地址总线位 12.
		PE.10	MFP2	O	
	EBI_ADR13	PG.4	MFP2	O	EBI 地址总线位 13.
		PE.11	MFP2	O	

组	管脚名称	GPIO	MFP	类型	描述
	EBI_ADR14	PF.11	MFP2	O	EBI 地址总线位 14.
		PE.12	MFP2	O	
	EBI_ADR15	PF.10	MFP2	O	EBI 地址总线位 15.
		PE.13	MFP2	O	
	EBI_ADR16	PF.9	MFP2	O	EBI 地址总线位 16.
		PC.8	MFP2	O	
		PB.11	MFP2	O	
	EBI_ADR17	PF.8	MFP2	O	EBI 地址总线位 17.
		PB.10	MFP2	O	
	EBI_ADR18	PF.7	MFP2	O	EBI 地址总线位 18.
		PB.9	MFP2	O	
	EBI_ADR19	PF.6	MFP2	O	EBI 地址总线位 19.
		PB.8	MFP2	O	
	EBI_ALE	PA.8	MFP2	O	EBI地址锁存使能输出脚
		PE.2	MFP2	O	
	EBI_MCLK	PA.9	MFP2	O	EBI 外部时钟输出脚.
		PE.3	MFP2	O	
	EBI_nCS0	PD.12	MFP2	O	EBI片选0输出脚
		PF.6	MFP7	O	
		PF.3	MFP2	O	
		PD.14	MFP2	O	
		PB.7	MFP8	O	
	EBI_nCS1	PD.11	MFP2	O	EBI片选1输出管脚
		PF.2	MFP2	O	
		PB.6	MFP8	O	
	EBI_nCS2	PD.10	MFP2	O	EBI片选2输出脚
	EBI_nRD	PA.11	MFP2	O	EBI 读使能输出脚
		PE.5	MFP2	O	
	EBI_nWR	PA.10	MFP2	O	EBI 写使能输出脚
		PE.4	MFP2	O	
	EBI_nWRH	PB.6	MFP2	O	EBI 高字节写使能输出管脚
	EBI_nWRL	PB.7	MFP2	O	EBI低字节写使能输出脚
ECAP0	ECAP0_IC0	PA.10	MFP11	I	增强型捕获计数器 0 输入脚0

组	管脚名称	GPIO	MFP	类型	描述
		PE.8	MFP12	I	
	ECAP0_IC1	PA.9	MFP11	I	增强型捕获计数器 0 输入 1 管脚.
		PE.9	MFP12	I	
	ECAP0_IC2	PA.8	MFP11	I	增强型捕获计数器 0 输入 2 管脚.
		PE.10	MFP12	I	
ECAP1	ECAP1_IC0	PC.10	MFP11	I	增强型捕获计数器 1 输入脚0
		PE.13	MFP13	I	
	ECAP1_IC1	PC.11	MFP11	I	增强型捕获计数器 1 输入 1 管脚.
		PE.12	MFP13	I	
	ECAP1_IC2	PC.12	MFP11	I	增强型捕获计数器 1 输入 2 管脚.
		PE.11	MFP13	I	
EPWM0	EPWM0_BRAKE0	PB.1	MFP13	I	EPWM0 剎车0 输入脚
		PE.8	MFP11	I	
	EPWM0_BRAKE1	PB.0	MFP13	I	EPWM0 剎车1 输入脚
		PE.9	MFP11	I	
		PB.15	MFP10	I	
	EPWM0_CH0	PB.5	MFP11	I/O	EPWM0 通道0输出/捕获输入
		PF.5	MFP7	I/O	
		PE.8	MFP10	I/O	
		PA.5	MFP13	I/O	
		PE.7	MFP12	I/O	
	EPWM0_CH1	PB.4	MFP11	I/O	EPWM0 通道1输出/捕获输入脚
		PF.4	MFP7	I/O	
		PE.9	MFP10	I/O	
		PA.4	MFP13	I/O	
		PE.6	MFP12	I/O	
	EPWM0_CH2	PB.3	MFP11	I/O	EPWM0 通道2输出/捕获输入脚
		PE.10	MFP10	I/O	
		PA.3	MFP13	I/O	
		PE.5	MFP12	I/O	
	EPWM0_CH3	PB.2	MFP11	I/O	EPWM0 通道3输出/捕获输入脚
		PE.11	MFP10	I/O	
		PA.2	MFP13	I/O	

组	管脚名称	GPIO	MFP	类型	描述
		PE.4	MFP12	I/O	
	EPWM0_CH4	PB.1	MFP11	I/O	EPWM0 通道4输出/捕获输入脚
		PE.12	MFP10	I/O	
		PA.1	MFP13	I/O	
		PE.3	MFP12	I/O	
		PD.14	MFP11	I/O	
	EPWM0_CH5	PB.0	MFP11	I/O	EPWM0 通道5 输出/捕获输入
		PE.13	MFP10	I/O	
		PA.0	MFP13	I/O	
		PE.2	MFP12	I/O	
		PH.11	MFP11	I/O	
	EPWM0_SYNC_IN	PA.15	MFP12	I	EPWM0 计数器同步触发输入脚
		PC.14	MFP11	I	
	EPWM0_SYNC_OUT	PA.11	MFP10	O	EPWM0计数器同步触发输出脚
		PF.5	MFP9	O	
EPWM1	EPWM1_BRAKE0	PE.10	MFP11	I	EPWM1 刹车0 输入脚
		PB.7	MFP11	I	
	EPWM1_BRAKE1	PE.11	MFP11	I	EPWM1 刹车1输入管脚
		PA.3	MFP15	I	
		PB.6	MFP11	I	
	EPWM1_CH0	PC.12	MFP12	I/O	EPWM1 通道0 输出/捕获输入
		PE.13	MFP11	I/O	
		PC.5	MFP12	I/O	
		PB.15	MFP11	I/O	
	EPWM1_CH1	PC.11	MFP12	I/O	EPWM1 通道1 输出/捕获输入
		PC.8	MFP11	I/O	
		PC.4	MFP12	I/O	
		PB.14	MFP11	I/O	
	EPWM1_CH2	PC.10	MFP12	I/O	EPWM1 通道2 输出/捕获输入
		PC.7	MFP11	I/O	
		PC.3	MFP12	I/O	
		PB.13	MFP11	I/O	
	EPWM1_CH3	PC.9	MFP12	I/O	EPWM1 通道3输出/捕获输入脚

组	管脚名称	GPIO	MFP	类型	描述
		PC.6	MFP11	I/O	
		PC.2	MFP12	I/O	
		PB.12	MFP11	I/O	
	EPWM1_CH4	PB.1	MFP12	I/O	EPWM1 通道4输出/捕获输入脚
		PA.7	MFP11	I/O	
		PC.1	MFP12	I/O	
		PB.7	MFP12	I/O	
	EPWM1_CH5	PB.0	MFP12	I/O	EPWM1 通道5输出/捕获输入
		PA.6	MFP11	I/O	
		PC.0	MFP12	I/O	
		PB.6	MFP12	I/O	
ETM	ETM_TRACE_CLK	PC.14	MFP12	I	ETM接收追踪时钟输入管脚
	ETM_TRACE_DATA0	PB.15	MFP12	I	ETM接收追踪数据0 输入管脚
	ETM_TRACE_DATA1	PB.14	MFP12	I	ETM接收追踪数据1 输入管脚
	ETM_TRACE_DATA2	PB.13	MFP12	I	ETM接收追踪数据2 输入管脚
	ETM_TRACE_DATA3	PB.12	MFP12	I	ETM接收追踪数据3 输入管脚
I^2C0	I^2C0_SCL	PB.5	MFP6	I/O	I2C0 时钟管脚
		PC.12	MFP4	I/O	
		PF.3	MFP4	I/O	
		PE.13	MFP4	I/O	
		PA.5	MFP9	I/O	
		PC.1	MFP9	I/O	
		PB.9	MFP9	I/O	
	I^2C0_SDA	PB.4	MFP6	I/O	I2C0 数据输入/输出管脚
		PC.11	MFP4	I/O	
		PF.2	MFP4	I/O	
		PC.8	MFP4	I/O	
		PA.4	MFP9	I/O	
		PC.0	MFP9	I/O	
		PB.8	MFP9	I/O	
	I^2C0_SMBAL	PG.2	MFP4	O	I^2C0 SMBus SMBALTER 管脚
		PA.3	MFP10	O	
		PC.3	MFP9	O	

组	管脚名称	GPIO	MFP	类型	描述
	I²C0_SMBSUS	PG.3	MFP4	O	I²C0 SMBus SMBSUS 管脚 (PMBus 控制管脚)
		PA.2	MFP10	O	
		PC.2	MFP9	O	
I²C1	I²C1_SCL	PB.3	MFP12	I/O	I²C1 时钟脚
		PB.1	MFP9	I/O	
		PG.2	MFP5	I/O	
		PA.7	MFP8	I/O	
		PA.3	MFP9	I/O	
		PF.0	MFP3	I/O	
		PC.5	MFP9	I/O	
		PA.12	MFP4	I/O	
		PE.1	MFP8	I/O	
		PB.11	MFP7	I/O	
	I²C1_SDA	PB.2	MFP12	I/O	I²C1 数据输入/输出脚
		PB.0	MFP9	I/O	
		PG.3	MFP5	I/O	
		PA.6	MFP8	I/O	
		PA.2	MFP9	I/O	
		PF.1	MFP3	I/O	
		PC.4	MFP9	I/O	
		PA.13	MFP4	I/O	
		PE.0	MFP8	I/O	
		PB.10	MFP7	I/O	
	I²C1_SMBAL	PC.7	MFP8	O	I²C1 SMBus SMBALTER 管脚
		PH.8	MFP8	O	
		PB.9	MFP7	O	
	I²C1_SMBSUS	PC.6	MFP8	O	I²C1 SMBus SMBSUS 管脚 (PMBus 控制管脚)
		PH.9	MFP8	O	
		PB.8	MFP7	O	
I²C2	I²C2_SCL	PA.11	MFP7	I/O	I²C2 时钟脚
		PA.1	MFP9	I/O	
		PD.9	MFP3	I/O	
		PD.1	MFP6	I/O	

组	管脚名称	GPIO	MFP	类型	描述
		PA.14	MFP6	I/O	
		PH.8	MFP9	I/O	
		PB.13	MFP8	I/O	
	I²C2_SDA	PA.10	MFP7	I/O	I²C2 数据输入/输出脚
		PA.0	MFP9	I/O	
		PD.8	MFP3	I/O	
		PD.0	MFP6	I/O	
		PA.15	MFP6	I/O	
		PH.9	MFP9	I/O	
		PB.12	MFP8	I/O	
	I²C2_SMBAL	PB.15	MFP8	O	I²C2 SMBus SMBALTER 管脚
	I²C2_SMBSUS	PB.14	MFP8	O	I²C2 SMBus SMBSUS 管脚 (PMBus 控制管脚)
I²S0	I²S0_BCLK	PB.5	MFP10	O	I2S0位时钟输出管脚
		PF.10	MFP4	O	
		PE.8	MFP4	O	
		PC.4	MFP6	O	
		PA.12	MFP2	O	
		PE.1	MFP5	O	
	I²S0_DI	PB.3	MFP10	I	I2S0 数据输入脚
		PF.8	MFP4	I	
		PE.10	MFP4	I	
		PC.2	MFP6	I	
		PA.14	MFP2	I	
		PH.8	MFP5	I	
	I²S0_DO	PB.2	MFP10	O	I²S0 数据输出脚
		PF.7	MFP4	O	
		PE.11	MFP4	O	
		PC.1	MFP6	O	
		PA.15	MFP2	O	
		PH.9	MFP5	O	
	I²S0_LRCK	PB.1	MFP10	O	I²S0左右通道时钟输出脚.
		PF.6	MFP4	O	
		PE.12	MFP4	O	

组	管脚名称	GPIO	MFP	类型	描述
		PC.0	MFP6	O	
		PH.10	MFP5	O	
	I^2S0_MCLK	PB.4	MFP10	O	I2S0 主时钟输出脚
		PF.9	MFP4	O	
		PE.9	MFP4	O	
		PC.3	MFP6	O	
		PA.13	MFP2	O	
		PE.0	MFP5	O	
ICE	ICE_CLK	PF.1	MFP14	I/O	ADC 外部触发输入
	ICE_DAT	PF.0	MFP14	I/O	ADC 外部触发输入
INT0	INT0	PB.5	MFP15	I	外部中断0输入管脚
		PA.6	MFP15	I	
INT1	INT1	PB.4	MFP15	I	外部中断1输入脚
		PA.7	MFP15	I	
INT2	INT2	PB.3	MFP15	I	外部中断2输入脚
		PC.6	MFP15	I	
INT3	INT3	PB.2	MFP15	I	外部中断3输入脚
		PC.7	MFP15	I	
INT4	INT4	PA.8	MFP15	I	外部中断4输入管脚
		PB.6	MFP13	I	
INT5	INT5	PD.12	MFP15	I	外部中断5 输入脚
		PB.7	MFP13	I	
INT6	INT6	PD.11	MFP15	I	外部中断6 输入脚
		PB.8	MFP13	I	
INT7	INT7	PD.10	MFP15	I	外部中断7 输入脚
		PB.9	MFP13	I	
QEI0	QEI0_A	PD.11	MFP10	I	正交编码器0 相位A输入
		PA.4	MFP14	I	
		PE.3	MFP11	I	
	QEI0_B	PD.10	MFP10	I	正交编码器0 相位B 输入
		PA.3	MFP14	I	
		PE.2	MFP11	I	
	QEI0_INDEX	PD.12	MFP10	I	正交编码器0 索引输入

组	管脚名称	GPIO	MFP	类型	描述
		PA.5	MFP14	I	
		PE.4	MFP11	I	
QEI1	QEI1_A	PA.9	MFP10	I	正交编码器1 相位A输入
		PA.13	MFP12	I	
		PE.6	MFP11	I	
	QEI1_B	PA.8	MFP10	I	正交编码器1 相位B 输入
		PA.14	MFP12	I	
		PE.5	MFP11	I	
	QEI1_INDEX	PA.10	MFP10	I	正交编码器1 索引输入
		PA.12	MFP12	I	
		PE.7	MFP11	I	
QSPI0	QSPI0_CLK	PF.2	MFP5	I/O	Quad SPI0 串行时钟脚
		PA.2	MFP3	I/O	
		PC.2	MFP4	I/O	
		PH.8	MFP3	I/O	
		PC.14	MFP6	I/O	
	QSPI0_MISO0	PA.1	MFP3	I/O	Quad SPI0 MISO0（主机输入，从机输出）脚.
		PC.1	MFP4	I/O	
		PE.1	MFP3	I/O	
	QSPI0_MISO1	PA.5	MFP3	I/O	Quad SPI0 MISO1（主机输入，从机输出）脚.
		PC.5	MFP4	I/O	
		PH.10	MFP3	I/O	
	QSPI0_MOSI0	PA.0	MFP3	I/O	Quad SPI0 MOSI0 (主机输出，从机输入) 管脚.
		PC.0	MFP4	I/O	
		PE.0	MFP3	I/O	
	QSPI0_MOSI1	PA.4	MFP3	I/O	Quad SPI0 MOSI1 (主机输出，从机输入) 管脚.
		PC.4	MFP4	I/O	
		PH.11	MFP3	I/O	
	QSPI0_SS	PA.3	MFP3	I/O	Quad SPI0从机选择管脚.
		PC.3	MFP4	I/O	
		PH.9	MFP3	I/O	
QSPI1	QSPI1_CLK	PC.4	MFP14	I/O	Quad SPI1 串行时钟脚
		PG.12	MFP5	I/O	

组	管脚名称	GPIO	MFP	类型	描述
	QSPI1_MISO0	PC.3	MFP14	I/O	Quad SPI1 MISO0（主机输入，从机输出）脚.
		PG.13	MFP5	I/O	
	QSPI1_MISO1	PA.7	MFP9	I/O	Quad SPI1 MISO1（主机输入，从机输出）脚.
		PG.9	MFP5	I/O	
	QSPI1_MOSI0	PC.2	MFP14	I/O	Quad SPI1 MOSI0 (主机输出，从机输入) 管脚.
		PG.14	MFP5	I/O	
	QSPI1_MOSI1	PA.6	MFP9	I/O	Quad SPI1 MOSI1 (主机输出，从机输入) 管脚.
		PG.10	MFP5	I/O	
	QSPI1_SS	PC.5	MFP14	I/O	Quad SPI1从机选择管脚.
		PG.11	MFP5	I/O	
SC0	SC0_CLK	PB.5	MFP9	O	智能卡0时钟管脚
		PF.6	MFP3	O	
		PA.0	MFP6	O	
		PE.2	MFP6	O	
	SC0_DAT	PB.4	MFP9	I/O	智能卡0数据输入脚
		PF.7	MFP3	I/O	
		PA.1	MFP6	I/O	
		PE.3	MFP6	I/O	
	SC0_PWR	PB.2	MFP9	O	智能卡0电源脚
		PF.9	MFP3	O	
		PA.3	MFP6	O	
		PE.5	MFP6	O	
	SC0_RST	PB.3	MFP9	O	智能卡0复位脚
		PF.8	MFP3	O	
		PA.2	MFP6	O	
		PE.4	MFP6	O	
	SC0_nCD	PC.12	MFP9	I	智能卡 0 卡检测管脚.
		PF.10	MFP3	I	
		PA.4	MFP6	I	
		PE.6	MFP6	I	
SD0	SD0_CLK	PB.1	MFP3	O	SD/SDIO0 时钟输出脚
		PE.6	MFP3	O	
	SD0_CMD	PB.0	MFP3	I/O	SD/SDIO0 命令/响应脚

组	管脚名称	GPIO	MFP	类型	描述
		PE.7	MFP3	I/O	
	SD0_DAT0	PB.2	MFP3	I/O	SD/SDIO0 数据线位0
		PE.2	MFP3	I/O	
	SD0_DAT1	PB.3	MFP3	I/O	SD/SDIO0 数据线位1
		PE.3	MFP3	I/O	
	SD0_DAT2	PB.4	MFP3	I/O	SD/SDIO0 数据线位2
		PE.4	MFP3	I/O	
	SD0_DAT3	PB.5	MFP3	I/O	SD/SDIO0数据线位3
		PE.5	MFP3	I/O	
	SD0_nCD	PD.13	MFP3	I	SD/SDIO0 卡检测输入管脚
		PB.12	MFP9	I	
SPI0	SPI0_CLK	PF.8	MFP5	I/O	SPI0 串行时钟脚
		PA.2	MFP4	I/O	
		PD.2	MFP4	I/O	
		PB.14	MFP4	I/O	
	SPI0_I2SMCLK	PB.0	MFP8	I/O	SPI0 I^2S 主机时钟输出脚
		PF.10	MFP5	I/O	
		PA.4	MFP4	I/O	
		PD.13	MFP4	I/O	
		PD.14	MFP5	I/O	
		PC.14	MFP4	I/O	
		PB.11	MFP9	I/O	
	SPI0_MISO	PF.7	MFP5	I/O	SPI0 MISO（主机输入，从机输出）脚.
		PA.1	MFP4	I/O	
		PD.1	MFP4	I/O	
		PB.13	MFP4	I/O	
	SPI0_MOSI	PF.6	MFP5	I/O	SPI0 MOSI (主机输出，从机输入) 管脚.
		PA.0	MFP4	I/O	
		PD.0	MFP4	I/O	
		PB.12	MFP4	I/O	
	SPI0_SS	PF.9	MFP5	I/O	SPI0从机选择管脚.
		PA.3	MFP4	I/O	
		PD.3	MFP4	I/O	

组	管脚名称	GPIO	MFP	类型	描述
		PB.15	MFP4	I/O	
SPI1	SPI1_CLK	PB.3	MFP5	I/O	SPI1 串行时钟脚
		PH.6	MFP3	I/O	
		PA.7	MFP4	I/O	
		PC.1	MFP7	I/O	
		PH.8	MFP6	I/O	
	SPI1_I2SMCLK	PB.1	MFP5	I/O	SPI1 I^2S 主机时钟输出脚
		PA.5	MFP4	I/O	
		PC.4	MFP7	I/O	
		PD.13	MFP5	I/O	
		PH.10	MFP6	I/O	
	SPI1_MISO	PB.5	MFP5	I/O	SPI1 MISO（主机输入，从机输出）脚.
		PH.4	MFP3	I/O	
		PC.7	MFP4	I/O	
		PC.3	MFP7	I/O	
		PE.1	MFP6	I/O	
	SPI1_MOSI	PB.4	MFP5	I/O	SPI1 MOSI (主机输出，从机输入) 管脚.
		PH.5	MFP3	I/O	
		PC.6	MFP4	I/O	
		PC.2	MFP7	I/O	
		PE.0	MFP6	I/O	
	SPI1_SS	PB.2	MFP5	I/O	SPI1从机选择管脚.
		PH.7	MFP3	I/O	
		PA.6	MFP4	I/O	
		PC.0	MFP7	I/O	
		PH.9	MFP6	I/O	
SPI2	SPI2_CLK	PA.10	MFP4	I/O	SPI2 串行时钟管脚
		PG.3	MFP3	I/O	
		PE.8	MFP5	I/O	
		PA.13	MFP5	I/O	
	SPI2_I2SMCLK	PB.0	MFP4	I/O	SPI2 I^2S 主机时钟输出脚
		PC.13	MFP4	I/O	
		PE.12	MFP5	I/O	

组	管脚名称	GPIO	MFP	类型	描述
	SPI2_MISO	PA.9	MFP4	I/O	SPI2 MISO(主机输入，从机输出)管脚
		PG.4	MFP3	I/O	
		PE.9	MFP5	I/O	
		PA.14	MFP5	I/O	
	SPI2_MOSI	PA.8	MFP4	I/O	SPI2 MOSI (主机输出，从机输入) 管脚
		PF.11	MFP3	I/O	
		PE.10	MFP5	I/O	
		PA.15	MFP5	I/O	
	SPI2_SS	PA.11	MFP4	I/O	SPI2从机选择管脚.
		PG.2	MFP3	I/O	
		PE.11	MFP5	I/O	
		PA.12	MFP5	I/O	
SWDH	SWDH_CLK	PA.10	MFP15	O	串行调试总线主时钟输出
	SWDH_DAT	PA.9	MFP15	I/O	串行调试总线主机数据输入/输出管脚
TAMPER0	TAMPER0	PF.6	MFP10	I/O	TAMPER循环检测管脚 0.
TAMPER1	TAMPER1	PF.7	MFP10	I/O	TAMPER循环检测管脚 1.
TAMPER2	TAMPER2	PF.8	MFP10	I/O	TAMPER循环检测管脚 2.
TM0	TM0	PB.5	MFP14	I/O	Timer0 事件计数输入/反转输出管脚
		PG.2	MFP13	I/O	
		PC.7	MFP14	I/O	
	TM0_EXT	PA.11	MFP13	I/O	Timer0 外部捕获输入/反转输出脚
		PB.15	MFP13	I/O	
TM1	TM1	PB.4	MFP14	I/O	Timer1 事件计数输入/反转输出脚
		PG.3	MFP13	I/O	
		PC.6	MFP14	I/O	
		PC.14	MFP13	I/O	
	TM1_EXT	PA.10	MFP13	I/O	Timer1 外部捕获输入/反转输出脚
		PB.14	MFP13	I/O	
TM2	TM2	PB.3	MFP14	I/O	Timer2 事件计数输入/反转输出脚
		PG.4	MFP13	I/O	
		PA.7	MFP14	I/O	
		PD.0	MFP14	I/O	
	TM2_EXT	PA.9	MFP13	I/O	Timer2 外部捕获输入/反转输出脚

组	管脚名称	GPIO	MFP	类型	描述
		PB.13	MFP13	I/O	
TM3	TM3	PB.2	MFP14	I/O	Timer3 事件计数输入/反转输出脚
		PF.11	MFP13	I/O	
		PA.6	MFP14	I/O	
	TM3_EXT	PA.8	MFP13	I/O	Timer3 外部捕获输入/反转输出脚
		PB.12	MFP13	I/O	
TRACE	TRACE_CLK	PE.12	MFP14	O	ETM跟踪时钟输出脚
	TRACE_DATA0	PE.11	MFP14	O	ETM跟踪数据0 输出管脚
	TRACE_DATA1	PE.10	MFP14	O	ETM跟踪数据1 输出管脚
	TRACE_DATA2	PE.9	MFP14	O	ETM跟踪数据2 输出管脚
	TRACE_DATA3	PE.8	MFP14	O	ETM跟踪数据3 输出管脚
UART0	UART0_RXD	PC.11	MFP3	I	UART0 数据接收输入脚
		PF.2	MFP3	I	
		PA.6	MFP7	I	
		PA.4	MFP11	I	
		PA.0	MFP7	I	
		PF.1	MFP4	I	
		PD.2	MFP9	I	
		PA.15	MFP3	I	
		PH.11	MFP8	I	
		PB.12	MFP6	I	
		PB.8	MFP5	I	
	UART0_TXD	PC.12	MFP3	O	UART0 数据发送输出脚.
		PF.3	MFP3	O	
		PA.7	MFP7	O	
		PA.5	MFP11	O	
		PA.1	MFP7	O	
		PF.0	MFP4	O	
		PD.3	MFP9	O	
		PA.14	MFP3	O	
		PH.10	MFP8	O	
		PB.13	MFP6	O	
		PB.9	MFP5	O	

组	管脚名称	GPIO	MFP	类型	描述
	UART0_nCTS	PC.7	MFP7	I	UART0 禁止发送输入管脚
		PA.5	MFP7	I	
		PB.15	MFP6	I	
		PB.11	MFP5	I	
	UART0_nRTS	PC.6	MFP7	O	UART0 请求发送输出脚
		PA.4	MFP7	O	
		PB.14	MFP6	O	
		PB.10	MFP5	O	
UART1	UART1_RXD	PB.2	MFP6	I	UART1数据接收输入管脚
		PA.8	MFP7	I	
		PD.10	MFP3	I	
		PC.8	MFP8	I	
		PA.2	MFP8	I	
		PF.1	MFP2	I	
		PH.9	MFP10	I	
		PB.6	MFP6	I	
	UART1_TXD	PB.3	MFP6	O	UART1 数据发送输出脚
		PA.9	MFP7	O	
		PD.11	MFP3	O	
		PE.13	MFP8	O	
		PA.3	MFP8	O	
		PF.0	MFP2	O	
		PH.8	MFP10	O	
		PB.7	MFP6	O	
	UART1_nCTS	PE.11	MFP8	I	UART1 禁止发送输入管脚
		PA.1	MFP8	I	
		PB.9	MFP6	I	
	UART1_nRTS	PE.12	MFP8	O	UART1 请求发送输出脚
		PA.0	MFP8	O	
		PB.8	MFP6	O	
UART2	UART2_RXD	PB.4	MFP12	I	UART2 数据接收输入脚
		PB.0	MFP7	I	
		PD.12	MFP7	I	

组	管脚名称	GPIO	MFP	类型	描述
		PF.5	MFP2	I	
		PE.9	MFP7	I	
		PE.15	MFP3	I	
		PC.4	MFP8	I	
		PC.0	MFP8	I	
	UART2_TXD	PB.5	MFP12	O	UART2 数据发送输出脚.
		PB.1	MFP7	O	
		PC.13	MFP7	O	
		PF.4	MFP2	O	
		PE.8	MFP7	O	
		PE.14	MFP3	O	
		PC.5	MFP8	O	
		PC.1	MFP8	O	
	UART2_nCTS	PF.5	MFP4	I	UART2 禁止发送输入管脚
		PD.9	MFP4	I	
		PC.2	MFP8	I	
	UART2_nRTS	PF.4	MFP4	O	UART2 请求发送输出脚
		PD.8	MFP4	O	
		PC.3	MFP8	O	
UART3	UART3_RXD	PC.9	MFP7	I	UART3 数据接收输入脚
		PE.11	MFP7	I	
		PC.2	MFP11	I	
		PD.0	MFP5	I	
		PE.0	MFP7	I	
		PB.14	MFP7	I	
	UART3_TXD	PC.10	MFP7	O	UART3 数据发送输出脚.
		PE.10	MFP7	O	
		PC.3	MFP11	O	
		PD.1	MFP5	O	
		PE.1	MFP7	O	
		PB.15	MFP7	O	
	UART3_nCTS	PD.2	MFP5	I	UART3 禁止发送输入管脚
		PH.9	MFP7	I	

组	管脚名称	GPIO	MFP	类型	描述
		PB.12	MFP7	I	
	UART3_nRTS	PD.3	MFP5	O	UART3 请求发送输出脚
		PH.8	MFP7	O	
		PB.13	MFP7	O	
UART4	UART4_RXD	PF.6	MFP6	I	UART4 数据接收输入脚
		PC.6	MFP5	I	
		PA.2	MFP7	I	
		PC.4	MFP11	I	
		PA.13	MFP3	I	
		PH.11	MFP7	I	
		PB.10	MFP6	I	
	UART4_TXD	PF.7	MFP6	O	UART4 数据发送输出脚.
		PC.7	MFP5	O	
		PA.3	MFP7	O	
		PC.5	MFP11	O	
		PA.12	MFP3	O	
		PH.10	MFP7	O	
		PB.11	MFP6	O	
	UART4_nCTS	PC.8	MFP5	I	UART4 禁止发送输入管脚
		PE.1	MFP9	I	
	UART4_nRTS	PE.13	MFP5	O	UART4 请求发送输出脚
		PE.0	MFP9	O	
UART5	UART5_RXD	PB.4	MFP7	I	UART5 数据接收输入管脚
		PF.10	MFP6	I	
		PA.4	MFP8	I	
		PE.6	MFP8	I	
	UART5_TXD	PB.5	MFP7	O	UART5数据发送输出脚
		PF.11	MFP6	O	
		PA.5	MFP8	O	
		PE.7	MFP8	O	
	UART5_nCTS	PB.2	MFP7	I	UART5 禁止发送输入管脚
		PF.8	MFP6	I	
	UART5_nRTS	PB.3	MFP7	O	UART5 请求发送输出脚

组	管脚名称	GPIO	MFP	类型	描述
		PF.9	MFP6	O	
UART6	UART6_RXD	PC.11	MFP5	I	UART6 数据接收输入脚
		PA.10	MFP8	I	
		PH.5	MFP5	I	
		PC.6	MFP9	I	
		PE.15	MFP6	I	
		PG.14	MFP6	I	
		PE.4	MFP8	I	
	UART6_TXD	PC.12	MFP5	O	UART6 数据发送输出脚.
		PA.11	MFP8	O	
		PH.4	MFP5	O	
		PC.7	MFP9	O	
		PE.14	MFP6	O	
		PG.13	MFP6	O	
		PE.5	MFP8	O	
	UART6_nCTS	PC.9	MFP5	I	UART6 禁止发送输入管脚
		PE.2	MFP8	I	
	UART6_nRTS	PC.10	MFP5	O	UART6 请求发送输出脚
		PE.3	MFP8	O	
UART7	UART7_RXD	PA.8	MFP8	I	UART7 数据接收输入脚
		PH.7	MFP4	I	
		PD.8	MFP5	I	
		PG.12	MFP6	I	
		PE.2	MFP9	I	
		PB.8	MFP8	I	
	UART7_TXD	PA.9	MFP8	O	UART7 数据发送输出脚.
		PH.6	MFP4	O	
		PD.9	MFP5	O	
		PG.11	MFP6	O	
		PE.3	MFP9	O	
		PB.9	MFP8	O	
	UART7_nCTS	PH.5	MFP4	I	UART7 禁止发送输入管脚
		PE.4	MFP9	I	

组	管脚名称	GPIO	MFP	类型	描述
	UART7_nRTS	PH.4	MFP4	O	UART7 请求发送输出脚
		PE.5	MFP9	O	
USB	USB_D+	PA.14	MFP14	A	USB差分信号D+.
	USB_D-	PA.13	MFP14	A	USB差分信号D-.
	USB_OTG_ID	PA.15	MFP14	I	USB_ 标识
	USB_VBUS	PA.12	MFP14	P	USB主机或HUB的电源
	USB_VBUS_EN	PB.15	MFP14	O	USB 外部 VBUS 电源使能管脚.
		PB.6	MFP14	O	
	USB_VBUS_ST	PC.14	MFP14	I	USB 外部VBUS 电源状态管脚.
		PB.14	MFP15	I	
		PB.7	MFP14	I	
X32	X32_IN	PF.5	MFP10	I	外部 32.768 kHz 晶振输入脚
	X32_OUT	PF.4	MFP10	O	外部 32.768 kHz 晶振输出脚
XT1	XT1_IN	PF.3	MFP10	I	外部 4~24 MHz (高速) 晶振输入脚
	XT1_OUT	PF.2	MFP10	O	外部 4~24 MHz (高速) 晶振输出脚

4.3.5 多功能管脚表以GPIO排列

	管脚名称	类型	MFP	描述
PA.0	PA.0	I/O	MFP0	通用数字I/O脚
	QSPI0_MOSI0	I/O	MFP3	Quad SPI0 MOSI0 (主机输出，从机输入) 管脚.
	SPI0_MOSI	I/O	MFP4	SPI0 MOSI (主机输出，从机输入) 管脚.
	SC0_CLK	O	MFP6	智能卡0时钟管脚
	UART0_RXD	I	MFP7	UART0 数据接收输入脚
	UART1_nRTS	O	MFP8	UART1 请求发送输出脚
	I^2C2_SDA	I/O	MFP9	I^2C2 数据输入/输出脚
	CAP_DATA6	I	MFP10	图像数据输入总线位6.
	BPWM0_CH0	I/O	MFP12	BPWM0 通道0 输出/捕获输入
	EPWM0_CH5	I/O	MFP13	EPWM0 通道5 输出/捕获输入
	DAC0_ST	I	MFP15	DAC0 外部触发输入
PA.1	PA.1	I/O	MFP0	通用数字I/O脚
	QSPI0_MISO0	I/O	MFP3	Quad SPI0 MISO0（主机输入，从机输出）脚.
	SPI0_MISO	I/O	MFP4	SPI0 MISO（主机输入，从机输出）脚.
	SC0_DAT	I/O	MFP6	智能卡0数据输入脚
	UART0_TXD	O	MFP7	UART0 数据发送输出脚.
	UART1_nCTS	I	MFP8	UART1 禁止发送输入管脚
	I^2C2_SCL	I/O	MFP9	I^2C2 时钟脚
	CAP_DATA7	I	MFP10	图像数据输入总线位7.
	BPWM0_CH1	I/O	MFP12	BPWM0 通道1 输出/捕获输入
	EPWM0_CH4	I/O	MFP13	EPWM0 通道4输出/捕获输入脚
PA.2	PA.2	I/O	MFP0	通用数字I/O脚
	QSPI0_CLK	I/O	MFP3	Quad SPI0 串行时钟脚
	SPI0_CLK	I/O	MFP4	SPI0 串行时钟脚
	SC0_RST	O	MFP6	智能卡0复位脚
	UART4_RXD	I	MFP7	UART4 数据接收输入脚
	UART1_RXD	I	MFP8	UART1数据接收输入管脚
	I^2C1_SDA	I/O	MFP9	I^2C1 数据输入/输出脚
	I^2C0_SMBSUS	O	MFP10	I^2C0 SMBus SMBSUS 管脚 (PMBus 控制管脚)
	BPWM0_CH2	I/O	MFP12	BPWM0 通道2 输出/捕获输入

	管脚名称	类型	MFP	描述
	EPWM0_CH3	I/O	MFP13	EPWM0 通道3输出/捕获输入脚
PA.3	PA.3	I/O	MFP0	通用数字I/O脚
	QSPI0_SS	I/O	MFP3	Quad SPI0从机选择管脚.
	SPI0_SS	I/O	MFP4	SPI0从机选择管脚.
	SC0_PWR	O	MFP6	智能卡0电源脚
	UART4_TXD	O	MFP7	UART4 数据发送输出脚.
	UART1_TXD	O	MFP8	UART1 数据发送输出脚
	I^2C1_SCL	I/O	MFP9	I^2C1 时钟脚
	I^2C0_SMBAL	O	MFP10	I^2C0 SMBus SMBALTER 管脚
	BPWM0_CH3	I/O	MFP12	BPWM0 通道3输出/捕获输入脚
	EPWM0_CH2	I/O	MFP13	EPWM0 通道2输出/捕获输入脚
	QEI0_B	I	MFP14	正交编码器0 相位B 输入
	EPWM1_BRAKE1	I	MFP15	EPWM1 刹车1输入管脚
PA.4	PA.4	I/O	MFP0	通用数字I/O脚
	QSPI0_MOSI1	I/O	MFP3	Quad SPI0 MOSI1 (主机输出，从机输入) 管脚.
	SPI0_I2SMCLK	I/O	MFP4	SPI0 I^2S 主机时钟输出脚
	SC0_nCD	I	MFP6	智能卡 0 卡检测管脚.
	UART0_nRTS	O	MFP7	UART0 请求发送输出脚
	UART5_RXD	I	MFP8	UART5 数据接收输入管脚
	I^2C0_SDA	I/O	MFP9	I2C0 数据输入/输出管脚
	CAN0_RXD	I	MFP10	CAN0 总线接收输入
	UART0_RXD	I	MFP11	UART0 数据接收输入脚
	BPWM0_CH4	I/O	MFP12	BPWM0 通道4输出/捕获输入脚
	EPWM0_CH1	I/O	MFP13	EPWM0 通道1输出/捕获输入脚
	QEI0_A	I	MFP14	正交编码器0 相位A输入
PA.5	PA.5	I/O	MFP0	通用数字I/O脚
	QSPI0_MISO1	I/O	MFP3	Quad SPI0 MISO1（主机输入，从机输出）脚.
	SPI1_I2SMCLK	I/O	MFP4	SPI1 I^2S 主机时钟输出脚
	UART0_nCTS	I	MFP7	UART0 禁止发送输入管脚
	UART5_TXD	O	MFP8	UART5数据发送输出脚
	I^2C0_SCL	I/O	MFP9	I2C0 时钟管脚

	管脚名称	类型	MFP	描述
	CAN0_TXD	O	MFP10	CAN0总线发送输出
	UART0_TXD	O	MFP11	UART0 数据发送输出脚.
	BPWM0_CH5	I/O	MFP12	BPWM0 通道5 输出/捕获输入
	EPWM0_CH0	I/O	MFP13	EPWM0 通道0输出/捕获输入
	QEI0_INDEX	I	MFP14	正交编码器0 索引输入
PA.6	PA.6	I/O	MFP0	通用数字I/O脚
	EBI_AD6	I/O	MFP2	EBI地址/数据总线位6.
	SPI1_SS	I/O	MFP4	SPI1从机选择管脚.
	UART0_RXD	I	MFP7	UART0 数据接收输入脚
	I^2C1_SDA	I/O	MFP8	I^2C1 数据输入/输出脚
	QSPI1_MOSI1	I/O	MFP9	Quad SPI1 MOSI1 (主机输出，从机输入) 管脚.
	EPWM1_CH5	I/O	MFP11	EPWM1 通道5输出/捕获输入
	BPWM1_CH3	I/O	MFP12	BPWM1 通道3输出/捕获输入脚
	ACMP1_WLAT	I	MFP13	模拟比较器 1窗口锁存输入管脚
	TM3	I/O	MFP14	Timer3 事件计数输入/反转输出脚
	INT0	I	MFP15	外部中断0输入管脚
PA.7	PA.7	I/O	MFP0	通用数字I/O脚
	EBI_AD7	I/O	MFP2	EBI地址/数据总线位7.
	SPI1_CLK	I/O	MFP4	SPI1 串行时钟脚
	UART0_TXD	O	MFP7	UART0 数据发送输出脚.
	I^2C1_SCL	I/O	MFP8	I^2C1 时钟脚
	QSPI1_MISO1	I/O	MFP9	Quad SPI1 MISO1（主机输入，从机输出）脚.
	EPWM1_CH4	I/O	MFP11	EPWM1 通道4输出/捕获输入脚
	BPWM1_CH2	I/O	MFP12	BPWM1 通道2 输出/捕获输入
	ACMP0_WLAT	I	MFP13	模拟比较器 0窗口锁存输入管脚
	TM2	I/O	MFP14	Timer2 事件计数输入/反转输出脚
	INT1	I	MFP15	外部中断1输入脚
PA.8	PA.8	I/O	MFP0	通用数字I/O脚
	EADC1_CH4	A	MFP1	EADC1 通道4 模拟输入
	EBI_ALE	O	MFP2	EBI地址锁存使能输出脚
	SPI2_MOSI	I/O	MFP4	SPI2 MOSI (主机输出，从机输入) 管脚

	管脚名称	类型	MFP	描述
	UART1_RXD	I	MFP7	UART1数据接收输入管脚
	UART7_RXD	I	MFP8	UART7 数据接收输入脚
	BPWM0_CH3	I/O	MFP9	BPWM0 通道3输出/捕获输入脚
	QEI1_B	I	MFP10	正交编码器1 相位B 输入
	ECAP0_IC2	I	MFP11	增强型捕获计数器 0 输入 2 管脚.
	TM3_EXT	I/O	MFP13	Timer3 外部捕获输入/反转输出脚
	INT4	I	MFP15	外部中断4输入管脚
PA.9	PA.9	I/O	MFP0	通用数字I/O脚
	EADC1_CH5	A	MFP1	EADC1 通道5 模拟输入
	EBI_MCLK	O	MFP2	EBI 外部时钟输出脚.
	SPI2_MISO	I/O	MFP4	SPI2 MISO(主机输入，从机输出)管脚
	UART1_TXD	O	MFP7	UART1 数据发送输出脚
	UART7_TXD	O	MFP8	UART7 数据发送输出脚.
	BPWM0_CH2	I/O	MFP9	BPWM0 通道2 输出/捕获输入
	QEI1_A	I	MFP10	正交编码器1 相位A输入
	ECAP0_IC1	I	MFP11	增强型捕获计数器 0 输入 1 管脚.
	TM2_EXT	I/O	MFP13	Timer2 外部捕获输入/反转输出脚
	SWDH_DAT	I/O	MFP15	串行调试总线主机数据输入/输出管脚
PA.10	PA.10	I/O	MFP0	通用数字I/O脚
	EADC1_CH6	A	MFP1	EADC1 通道6 模拟输入
	ACMP1_P0	A	MFP1	模拟比较器 1 正极输入脚0
	EBI_nWR	O	MFP2	EBI 写使能输出脚
	SPI2_CLK	I/O	MFP4	SPI2 串行时钟管脚
	I^2C2_SDA	I/O	MFP7	I^2C2 数据输入/输出脚
	UART6_RXD	I	MFP8	UART6 数据接收输入脚
	BPWM0_CH1	I/O	MFP9	BPWM0 通道1 输出/捕获输入
	QEI1_INDEX	I	MFP10	正交编码器1 索引输入
	ECAP0_IC0	I	MFP11	增强型捕获计数器 0 输入脚0
	TM1_EXT	I/O	MFP13	Timer1 外部捕获输入/反转输出脚
	DAC0_ST	I	MFP14	DAC0 外部触发输入
	SWDH_CLK	O	MFP15	串行调试总线主时钟输出

	管脚名称	类型	MFP	描述
PA.11	PA.11	I/O	MFP0	通用数字I/O脚
	EADC1_CH7	A	MFP1	EADC1 通道7 模拟输入
	ACMP0_P0	A	MFP1	模拟比较器 0 正极输入脚0
	EBI_nRD	O	MFP2	EBI 读使能输出脚
	SPI2_SS	I/O	MFP4	SPI2从机选择管脚.
	I^2C2_SCL	I/O	MFP7	I^2C2 时钟脚
	UART6_TXD	O	MFP8	UART6 数据发送输出脚.
	BPWM0_CH0	I/O	MFP9	BPWM0 通道0 输出/捕获输入
	EPWM0_SYNC_OUT	O	MFP10	EPWM0计数器同步触发输出脚
	TM0_EXT	I/O	MFP13	Timer0 外部捕获输入/反转输出脚
PA.12	PA.12	I/O	MFP0	通用数字I/O脚
	I^2S0_BCLK	O	MFP2	I2S0位时钟输出管脚
	UART4_TXD	O	MFP3	UART4 数据发送输出脚.
	I^2C1_SCL	I/O	MFP4	I^2C1 时钟脚
	SPI2_SS	I/O	MFP5	SPI2从机选择管脚.
	CAN0_TXD	O	MFP6	CAN0总线发送输出
	BPWM1_CH2	I/O	MFP11	BPWM1 通道2 输出/捕获输入
	QEI1_INDEX	I	MFP12	正交编码器1 索引输入
	USB_VBUS	P	MFP14	USB主机或HUB的电源
PA.13	PA.13	I/O	MFP0	通用数字I/O脚
	I^2S0_MCLK	O	MFP2	I2S0 主时钟输出脚
	UART4_RXD	I	MFP3	UART4 数据接收输入脚
	I^2C1_SDA	I/O	MFP4	I^2C1 数据输入/输出脚
	SPI2_CLK	I/O	MFP5	SPI2 串行时钟管脚
	CAN0_RXD	I	MFP6	CAN0 总线接收输入
	BPWM1_CH3	I/O	MFP11	BPWM1 通道3输出/捕获输入脚
	QEI1_A	I	MFP12	正交编码器1 相位A输入
	USB_D-	A	MFP14	USB差分信号D-.
PA.14	PA.14	I/O	MFP0	通用数字I/O脚
	I^2S0_DI	I	MFP2	I2S0 数据输入脚
	UART0_TXD	O	MFP3	UART0 数据发送输出脚.

	管脚名称	类型	MFP	描述
	SPI2_MISO	I/O	MFP5	SPI2 MISO(主机输入，从机输出)管脚
	I²C2_SCL	I/O	MFP6	I²C2 时钟脚
	BPWM1_CH4	I/O	MFP11	BPWM1 通道4输出/捕获输入脚
	QEI1_B	I	MFP12	正交编码器1 相位B 输入
	USB_D+	A	MFP14	USB差分信号D+.
PA.15	PA.15	I/O	MFP0	通用数字I/O脚
	I²S0_DO	O	MFP2	I²S0 数据输出脚
	UART0_RXD	I	MFP3	UART0 数据接收输入脚
	SPI2_MOSI	I/O	MFP5	SPI2 MOSI (主机输出，从机输入) 管脚
	I²C2_SDA	I/O	MFP6	I²C2 数据输入/输出脚
	BPWM1_CH5	I/O	MFP11	EPWM1 通道5输出/捕获输入
	EPWM0_SYNC_IN	I	MFP12	EPWM0 计数器同步触发输入脚
	USB_OTG_ID	I	MFP14	USB_ 标识
PB.0	PB.0	I/O	MFP0	通用数字I/O脚
	EADC0_CH0	A	MFP1	EADC0 通道0 模拟输入
	EADC1_CH8	A	MFP1	EADC1 通道8 模拟输入
	EBI_ADR9	O	MFP2	EBI 地址总线位 9.
	SD0_CMD	I/O	MFP3	SD/SDIO0 命令/响应脚
	SPI2_I2SMCLK	I/O	MFP4	SPI2 I²S 主机时钟输出脚
	UART2_RXD	I	MFP7	UART2 数据接收输入脚
	SPI0_I2SMCLK	I/O	MFP8	SPI0 I²S 主机时钟输出脚
	I²C1_SDA	I/O	MFP9	I²C1 数据输入/输出脚
	EPWM0_CH5	I/O	MFP11	EPWM0 通道5 输出/捕获输入
	EPWM1_CH5	I/O	MFP12	EPWM1 通道5输出/捕获输入
	EPWM0_BRAKE1	I	MFP13	EPWM0 刹车1 输入脚
PB.1	PB.1	I/O	MFP0	通用数字I/O脚
	EADC0_CH1	A	MFP1	EADC0 通道1 模拟输入
	EADC1_CH9	A	MFP1	EADC1 通道9 模拟输入
	EBI_ADR8	O	MFP2	EBI 地址总线位 8.
	SD0_CLK	O	MFP3	SD/SDIO0 时钟输出脚
	SPI1_I2SMCLK	I/O	MFP5	SPI1 I²S 主机时钟输出脚

	管脚名称	类型	MFP	描述
	UART2_TXD	O	MFP7	UART2 数据发送输出脚.
	I²C1_SCL	I/O	MFP9	I²C1 时钟脚
	I²S0_LRCK	O	MFP10	I²S0左右通道时钟输出脚.
	EPWM0_CH4	I/O	MFP11	EPWM0 通道4输出/捕获输入脚
	EPWM1_CH4	I/O	MFP12	EPWM1 通道4输出/捕获输入脚
	EPWM0_BRAKE0	I	MFP13	EPWM0 刹车0 输入脚
PB.2	PB.2	I/O	MFP0	通用数字I/O脚
	EADC0_CH2	A	MFP1	EADC0 通道2 模拟输入
	EADC1_CH10	A	MFP1	EADC1 通道10 模拟输入
	ACMP0_P1	A	MFP1	模拟比较器0正极输入1脚
	EBI_ADR3	O	MFP2	EBI 地址总线位3
	SD0_DAT0	I/O	MFP3	SD/SDIO0 数据线位0
	SPI1_SS	I/O	MFP5	SPI1从机选择管脚.
	UART1_RXD	I	MFP6	UART1数据接收输入管脚
	UART5_nCTS	I	MFP7	UART5 禁止发送输入管脚
	SC0_PWR	O	MFP9	智能卡0电源脚
	I²S0_DO	O	MFP10	I²S0 数据输出脚
	EPWM0_CH3	I/O	MFP11	EPWM0 通道3输出/捕获输入脚
	I²C1_SDA	I/O	MFP12	I²C1 数据输入/输出脚
	TM3	I/O	MFP14	Timer3 事件计数输入/反转输出脚
	INT3	I	MFP15	外部中断3输入脚
PB.3	PB.3	I/O	MFP0	通用数字I/O脚
	EADC0_CH3	A	MFP1	EADC0 通道3模拟输入
	EADC1_CH11	A	MFP1	EADC1 通道11 模拟输入
	ACMP0_N	A	MFP1	模拟比较器 0 负极输入管脚.
	EBI_ADR2	O	MFP2	EBI 地址总线位2
	SD0_DAT1	I/O	MFP3	SD/SDIO0 数据线位1
	SPI1_CLK	I/O	MFP5	SPI1 串行时钟脚
	UART1_TXD	O	MFP6	UART1 数据发送输出脚
	UART5_nRTS	O	MFP7	UART5 请求发送输出脚
	SC0_RST	O	MFP9	智能卡0复位脚

	管脚名称	类型	MFP	描述
	I²S0_DI	I	MFP10	I2S0 数据输入脚
	EPWM0_CH2	I/O	MFP11	EPWM0 通道2输出/捕获输入脚
	I²C1_SCL	I/O	MFP12	I²C1 时钟脚
	TM2	I/O	MFP14	Timer2 事件计数输入/反转输出脚
	INT2	I	MFP15	外部中断2输入脚
PB.4	PB.4	I/O	MFP0	通用数字I/O脚
	EADC0_CH4	A	MFP1	EADC0通道4模拟输入
	ACMP1_P1	A	MFP1	模拟比较器1正极输入1脚
	EBI_ADR1	O	MFP2	EBI 地址总线位1
	SD0_DAT2	I/O	MFP3	SD/SDIO0 数据线位2
	SPI1_MOSI	I/O	MFP5	SPI1 MOSI (主机输出，从机输入) 管脚.
	I²C0_SDA	I/O	MFP6	I2C0 数据输入/输出管脚
	UART5_RXD	I	MFP7	UART5 数据接收输入管脚
	SC0_DAT	I/O	MFP9	智能卡0数据输入脚
	I²S0_MCLK	O	MFP10	I2S0 主时钟输出脚
	EPWM0_CH1	I/O	MFP11	EPWM0 通道1输出/捕获输入脚
	UART2_RXD	I	MFP12	UART2 数据接收输入脚
	TM1	I/O	MFP14	Timer1 事件计数输入/反转输出脚
	INT1	I	MFP15	外部中断1输入脚
PB.5	PB.5	I/O	MFP0	通用数字I/O脚
	EADC0_CH5	A	MFP1	EADC0通道5模拟输入
	ACMP1_N	A	MFP1	模拟比较器 1 负极输入管脚.
	EBI_ADR0	O	MFP2	EBI地址总线位0
	SD0_DAT3	I/O	MFP3	SD/SDIO0数据线位3
	SPI1_MISO	I/O	MFP5	SPI1 MISO（主机输入，从机输出）脚.
	I²C0_SCL	I/O	MFP6	I2C0 时钟管脚
	UART5_TXD	O	MFP7	UART5数据发送输出脚
	SC0_CLK	O	MFP9	智能卡0时钟管脚
	I²S0_BCLK	O	MFP10	I2S0位时钟输出管脚
	EPWM0_CH0	I/O	MFP11	EPWM0 通道0输出/捕获输入
	UART2_TXD	O	MFP12	UART2 数据发送输出脚.

	管脚名称	类型	MFP	描述
	TM0	I/O	MFP14	Timer0 事件计数输入/反转输出管脚
	INT0	I	MFP15	外部中断0输入管脚
PB.6	PB.6	I/O	MFP0	通用数字I/O脚
	EADC0_CH6	A	MFP1	EADC0通道6模拟输入
	EBI_nWRH	O	MFP2	EBI 高字节写使能输出管脚
	CAN1_RXD	I	MFP5	CAN1 总线接收输入
	UART1_RXD	I	MFP6	UART1数据接收输入管脚
	EBI_nCS1	O	MFP8	EBI片选1输出管脚
	BPWM1_CH5	I/O	MFP10	EPWM1 通道5输出/捕获输入
	EPWM1_BRAKE1	I	MFP11	EPWM1 刹车1输入管脚
	EPWM1_CH5	I/O	MFP12	EPWM1 通道5输出/捕获输入
	INT4	I	MFP13	外部中断4输入管脚
	USB_VBUS_EN	O	MFP14	USB 外部 VBUS 电源使能管脚.
	ACMP1_O	O	MFP15	模拟比较器1输出脚
PB.7	PB.7	I/O	MFP0	通用数字I/O脚
	EADC0_CH7	A	MFP1	EADC0 通道7 模拟输入
	EBI_nWRL	O	MFP2	EBI低字节写使能输出脚
	CAN1_TXD	O	MFP5	CAN1总线发送输出
	UART1_TXD	O	MFP6	UART1 数据发送输出脚
	EBI_nCS0	O	MFP8	EBI片选0输出脚
	BPWM1_CH4	I/O	MFP10	BPWM1 通道4输出/捕获输入脚
	EPWM1_BRAKE0	I	MFP11	EPWM1 刹车0 输入脚
	EPWM1_CH4	I/O	MFP12	EPWM1 通道4输出/捕获输入脚
	INT5	I	MFP13	外部中断5 输入脚
	USB_VBUS_ST	I	MFP14	USB 外部VBUS 电源状态管脚.
	ACMP0_O	O	MFP15	模拟比较器 0输出脚
PB.8	PB.8	I/O	MFP0	通用数字I/O脚
	EADC0_CH8	A	MFP1	EADC0 通道8 模拟输入
	EBI_ADR19	O	MFP2	EBI 地址总线位 19.
	UART0_RXD	I	MFP5	UART0 数据接收输入脚
	UART1_nRTS	O	MFP6	UART1 请求发送输出脚

	管脚名称	类型	MFP	描述
	I²C1_SMBSUS	O	MFP7	I²C1 SMBus SMBSUS 管脚 (PMBus 控制管脚)
	UART7_RXD	I	MFP8	UART7 数据接收输入脚
	I²C0_SDA	I/O	MFP9	I2C0 数据输入/输出管脚
	BPWM1_CH3	I/O	MFP10	BPWM1 通道3输出/捕获输入脚
	CAN2_RXD	I	MFP12	CAN2 总线接收输入
	INT6	I	MFP13	外部中断6 输入脚
PB.9	PB.9	I/O	MFP0	通用数字I/O脚
	EADC0_CH9	A	MFP1	EADC0 通道9 模拟输入
	EBI_ADR18	O	MFP2	EBI 地址总线位 18.
	UART0_TXD	O	MFP5	UART0 数据发送输出脚.
	UART1_nCTS	I	MFP6	UART1 禁止发送输入管脚
	I²C1_SMBAL	O	MFP7	I²C1 SMBus SMBALTER 管脚
	UART7_TXD	O	MFP8	UART7 数据发送输出脚.
	I²C0_SCL	I/O	MFP9	I2C0 时钟管脚
	BPWM1_CH2	I/O	MFP10	BPWM1 通道2 输出/捕获输入
	CAN2_TXD	O	MFP12	CAN2总线发送输出
	INT7	I	MFP13	外部中断7 输入脚
	CAP_HSYNC	I	MFP14	图像采集接口hsync 输入脚
PB.10	PB.10	I/O	MFP0	通用数字I/O脚
	EADC0_CH10	A	MFP1	EADC0 通道10 模拟输入
	EBI_ADR17	O	MFP2	EBI 地址总线位 17.
	UART0_nRTS	O	MFP5	UART0 请求发送输出脚
	UART4_RXD	I	MFP6	UART4 数据接收输入脚
	I²C1_SDA	I/O	MFP7	I²C1 数据输入/输出脚
	CAN0_RXD	I	MFP8	CAN0 总线接收输入
	BPWM1_CH1	I/O	MFP10	BPWM1 通道1 输出/捕获输入
	CAP_VSYNC	I	MFP12	图像采集接口vsync 输入脚
PB.11	PB.11	I/O	MFP0	通用数字I/O脚
	EADC0_CH11	A	MFP1	EADC0 通道11 模拟输入
	EBI_ADR16	O	MFP2	EBI 地址总线位 16.
	UART0_nCTS	I	MFP5	UART0 禁止发送输入管脚

	管脚名称	类型	MFP	描述
	UART4_TXD	O	MFP6	UART4 数据发送输出脚.
	I²C1_SCL	I/O	MFP7	I²C1 时钟脚
	CAN0_TXD	O	MFP8	CAN0总线发送输出
	SPI0_I2SMCLK	I/O	MFP9	SPI0 I²S 主机时钟输出脚
	BPWM1_CH0	I/O	MFP10	BPWM1 通道0 输出/捕获输入
	CAP_SFIELD	I	MFP12	图像输入接口SFIELD 输入脚
PB.12	PB.12	I/O	MFP0	通用数字I/O脚
	EADC0_CH12	A	MFP1	EADC0 通道12 模拟输入
	EADC1_CH12	A	MFP1	EADC1 通道12 模拟输入
	DAC0_OUT	A	MFP1	DAC0 模拟输出通道
	ACMP0_P2	A	MFP1	模拟比较器 0 正极输入 2 管脚.
	ACMP1_P2	A	MFP1	模拟比较器 1 正极输入 2 管脚.
	EBI_AD15	I/O	MFP2	EBI地址/数据总线位15.
	SPI0_MOSI	I/O	MFP4	SPI0 MOSI (主机输出，从机输入) 管脚.
	UART0_RXD	I	MFP6	UART0 数据接收输入脚
	UART3_nCTS	I	MFP7	UART3 禁止发送输入管脚
	I²C2_SDA	I/O	MFP8	I²C2 数据输入/输出脚
	SD0_nCD	I	MFP9	SD/SDIO0 卡检测输入管脚
	CAP_SCLK	O	MFP10	图像采集接口传感器时钟管脚.
	EPWM1_CH3	I/O	MFP11	EPWM1 通道3输出/捕获输入脚
	ETM_TRACE_DATA3	I	MFP12	ETM接收追踪数据3 输入管脚
	TM3_EXT	I/O	MFP13	Timer3 外部捕获输入/反转输出脚
PB.13	PB.13	I/O	MFP0	通用数字I/O脚
	EADC0_CH13	A	MFP1	EADC0 通道13 模拟输入
	EADC1_CH13	A	MFP1	EADC1 通道13 模拟输入
	ACMP0_P3	A	MFP1	模拟比较器 0 正极输入 3 管脚.
	ACMP1_P3	A	MFP1	模拟比较器 1 正极输入 3 管脚.
	EBI_AD14	I/O	MFP2	EBI地址/数据总线位14.
	SPI0_MISO	I/O	MFP4	SPI0 MISO（主机输入，从机输出）脚.
	UART0_TXD	O	MFP6	UART0 数据发送输出脚.
	UART3_nRTS	O	MFP7	UART3 请求发送输出脚

	管脚名称	类型	MFP	描述
	I²C2_SCL	I/O	MFP8	I²C2 时钟脚
	CAP_PIXCLK	I	MFP10	图像采集接口像素时钟输入脚
	EPWM1_CH2	I/O	MFP11	EPWM1 通道2 输出/捕获输入
	ETM_TRACE_DATA2	I	MFP12	ETM接收追踪数据2 输入管脚
	TM2_EXT	I/O	MFP13	Timer2 外部捕获输入/反转输出脚
PB.14	PB.14	I/O	MFP0	通用数字I/O脚
	EADC0_CH14	A	MFP1	EADC0 通道14 模拟输入
	EADC1_CH14	A	MFP1	EADC1 通道14 模拟输入
	EBI_AD13	I/O	MFP2	EBI地址/数据总线位13.
	SPI0_CLK	I/O	MFP4	SPI0 串行时钟脚
	UART0_nRTS	O	MFP6	UART0 请求发送输出脚
	UART3_RXD	I	MFP7	UART3 数据接收输入脚
	I²C2_SMBSUS	O	MFP8	I²C2 SMBus SMBSUS 管脚 (PMBus 控制管脚)
	EPWM1_CH1	I/O	MFP11	EPWM1 通道1 输出/捕获输入
	ETM_TRACE_DATA1	I	MFP12	ETM接收追踪数据1 输入管脚
	TM1_EXT	I/O	MFP13	Timer1 外部捕获输入/反转输出脚
	CLKO	O	MFP14	时钟输出
	USB_VBUS_ST	I	MFP15	USB 外部VBUS 电源状态管脚.
PB.15	PB.15	I/O	MFP0	通用数字I/O脚
	EADC0_CH15	A	MFP1	EADC0 通道15 模拟输入
	EADC1_CH15	A	MFP1	EADC1 通道15 模拟输入
	EBI_AD12	I/O	MFP2	EBI地址/数据总线位12.
	SPI0_SS	I/O	MFP4	SPI0从机选择管脚.
	UART0_nCTS	I	MFP6	UART0 禁止发送输入管脚
	UART3_TXD	O	MFP7	UART3 数据发送输出脚.
	I²C2_SMBAL	O	MFP8	I²C2 SMBus SMBALTER 管脚
	EPWM0_BRAKE1	I	MFP10	EPWM0 刹车1 输入脚
	EPWM1_CH0	I/O	MFP11	EPWM1 通道0 输出/捕获输入
	ETM_TRACE_DATA0	I	MFP12	ETM接收追踪数据0 输入管脚
	TM0_EXT	I/O	MFP13	Timer0 外部捕获输入/反转输出脚
	USB_VBUS_EN	O	MFP14	USB 外部 VBUS 电源使能管脚.

	管脚名称	类型	MFP	描述
PC.0	PC.0	I/O	MFP0	通用数字I/O脚
	EBI_AD0	I/O	MFP2	EBI地址/数据总线位0.
	QSPI0_MOSI0	I/O	MFP4	Quad SPI0 MOSI0 (主机输出，从机输入) 管脚.
	I²S0_LRCK	O	MFP6	I²S0左右通道时钟输出脚.
	SPI1_SS	I/O	MFP7	SPI1从机选择管脚.
	UART2_RXD	I	MFP8	UART2 数据接收输入脚
	I²C0_SDA	I/O	MFP9	I2C0 数据输入/输出管脚
	CAN2_RXD	I	MFP10	CAN2 总线接收输入
	EPWM1_CH5	I/O	MFP12	EPWM1 通道5输出/捕获输入
	CAP_DATA0	I	MFP13	图像数据输入总线位0.
	ACMP1_O	O	MFP14	模拟比较器1输出脚
	EADC1_ST	I	MFP15	EADC1 外部触发输入
PC.1	PC.1	I/O	MFP0	通用数字I/O脚
	EBI_AD1	I/O	MFP2	EBI地址/数据总线位1.
	QSPI0_MISO0	I/O	MFP4	Quad SPI0 MISO0（主机输入，从机输出）脚.
	I²S0_DO	O	MFP6	I²S0 数据输出脚
	SPI1_CLK	I/O	MFP7	SPI1 串行时钟脚
	UART2_TXD	O	MFP8	UART2 数据发送输出脚.
	I²C0_SCL	I/O	MFP9	I2C0 时钟管脚
	CAN2_TXD	O	MFP10	CAN2总线发送输出
	EPWM1_CH4	I/O	MFP12	EPWM1 通道4输出/捕获输入脚
	CAP_DATA1	I	MFP13	图像数据输入总线位1.
	ACMP0_O	O	MFP14	模拟比较器 0输出脚
	EADC0_ST	I	MFP15	EADC0 外部触发输入
PC.2	PC.2	I/O	MFP0	通用数字I/O脚
	EBI_AD2	I/O	MFP2	EBI地址/数据总线位2.
	QSPI0_CLK	I/O	MFP4	Quad SPI0 串行时钟脚
	I²S0_DI	I	MFP6	I2S0 数据输入脚
	SPI1_MOSI	I/O	MFP7	SPI1 MOSI (主机输出，从机输入) 管脚.
	UART2_nCTS	I	MFP8	UART2 禁止发送输入管脚
	I²C0_SMBSUS	O	MFP9	I²C0 SMBus SMBSUS 管脚 (PMBus 控制管脚)

	管脚名称	类型	MFP	描述
	CAN1_RXD	I	MFP10	CAN1 总线接收输入
	UART3_RXD	I	MFP11	UART3 数据接收输入脚
	EPWM1_CH3	I/O	MFP12	EPWM1 通道3输出/捕获输入脚
	CAP_DATA2	I	MFP13	图像数据输入总线位2.
	QSPI1_MOSI0	I/O	MFP14	Quad SPI1 MOSI0 (主机输出，从机输入) 管脚.
PC.3	PC.3	I/O	MFP0	通用数字I/O脚
	EBI_AD3	I/O	MFP2	EBI地址/数据总线位3.
	QSPI0_SS	I/O	MFP4	Quad SPI0从机选择管脚.
	I^2S0_MCLK	O	MFP6	I2S0 主时钟输出脚
	SPI1_MISO	I/O	MFP7	SPI1 MISO（主机输入，从机输出）脚.
	UART2_nRTS	O	MFP8	UART2 请求发送输出脚
	I^2C0_SMBAL	O	MFP9	I^2C0 SMBus SMBALTER 管脚
	CAN1_TXD	O	MFP10	CAN1总线发送输出
	UART3_TXD	O	MFP11	UART3 数据发送输出脚.
	EPWM1_CH2	I/O	MFP12	EPWM1 通道2 输出/捕获输入
	CAP_DATA3	I	MFP13	图像数据输入总线位3.
	QSPI1_MISO0	I/O	MFP14	Quad SPI1 MISO0（主机输入，从机输出）脚.
PC.4	PC.4	I/O	MFP0	通用数字I/O脚
	EBI_AD4	I/O	MFP2	EBI地址/数据总线位4.
	QSPI0_MOSI1	I/O	MFP4	Quad SPI0 MOSI1 (主机输出，从机输入) 管脚.
	I^2S0_BCLK	O	MFP6	I2S0位时钟输出管脚
	SPI1_I2SMCLK	I/O	MFP7	SPI1 I^2S 主机时钟输出脚
	UART2_RXD	I	MFP8	UART2 数据接收输入脚
	I^2C1_SDA	I/O	MFP9	I^2C1 数据输入/输出脚
	CAN0_RXD	I	MFP10	CAN0 总线接收输入
	UART4_RXD	I	MFP11	UART4 数据接收输入脚
	EPWM1_CH1	I/O	MFP12	EPWM1 通道1 输出/捕获输入
	CAP_DATA4	I	MFP13	图像数据输入总线位4.
	QSPI1_CLK	I/O	MFP14	Quad SPI1 串行时钟脚
PC.5	PC.5	I/O	MFP0	通用数字I/O脚
	EBI_AD5	I/O	MFP2	EBI地址/数据总线位5.

	管脚名称	类型	MFP	描述
	QSPI0_MISO1	I/O	MFP4	Quad SPI0 MISO1（主机输入，从机输出）脚.
	UART2_TXD	O	MFP8	UART2 数据发送输出脚.
	I²C1_SCL	I/O	MFP9	I²C1 时钟脚
	CAN0_TXD	O	MFP10	CAN0总线发送输出
	UART4_TXD	O	MFP11	UART4 数据发送输出脚.
	EPWM1_CH0	I/O	MFP12	EPWM1 通道0 输出/捕获输入
	CAP_DATA5	I	MFP13	图像数据输入总线位5.
	QSPI1_SS	I/O	MFP14	Quad SPI1从机选择管脚.
PC.6	PC.6	I/O	MFP0	通用数字I/O脚
	EBI_AD8	I/O	MFP2	EBI地址/数据总线位8.
	SPI1_MOSI	I/O	MFP4	SPI1 MOSI (主机输出，从机输入) 管脚.
	UART4_RXD	I	MFP5	UART4 数据接收输入脚
	UART0_nRTS	O	MFP7	UART0 请求发送输出脚
	I²C1_SMBSUS	O	MFP8	I²C1 SMBus SMBSUS 管脚 (PMBus 控制管脚)
	UART6_RXD	I	MFP9	UART6 数据接收输入脚
	EPWM1_CH3	I/O	MFP11	EPWM1 通道3输出/捕获输入脚
	BPWM1_CH1	I/O	MFP12	BPWM1 通道1 输出/捕获输入
	TM1	I/O	MFP14	Timer1 事件计数输入/反转输出脚
	INT2	I	MFP15	外部中断2输入脚
PC.7	PC.7	I/O	MFP0	通用数字I/O脚
	EBI_AD9	I/O	MFP2	EBI地址/数据总线位9.
	SPI1_MISO	I/O	MFP4	SPI1 MISO（主机输入，从机输出）脚.
	UART4_TXD	O	MFP5	UART4 数据发送输出脚.
	UART0_nCTS	I	MFP7	UART0 禁止发送输入管脚
	I²C1_SMBAL	O	MFP8	I²C1 SMBus SMBALTER 管脚
	UART6_TXD	O	MFP9	UART6 数据发送输出脚.
	EPWM1_CH2	I/O	MFP11	EPWM1 通道2 输出/捕获输入
	BPWM1_CH0	I/O	MFP12	BPWM1 通道0 输出/捕获输入
	TM0	I/O	MFP14	Timer0 事件计数输入/反转输出管脚
	INT3	I	MFP15	外部中断3输入脚
PC.8	PC.8	I/O	MFP0	通用数字I/O脚

	管脚名称	类型	MFP	描述
	EBI_ADR16	O	MFP2	EBI 地址总线位 16.
	I²C0_SDA	I/O	MFP4	I2C0 数据输入/输出管脚
	UART4_nCTS	I	MFP5	UART4 禁止发送输入管脚
	UART1_RXD	I	MFP8	UART1数据接收输入管脚
	EPWM1_CH1	I/O	MFP11	EPWM1 通道1 输出/捕获输入
	BPWM1_CH4	I/O	MFP12	BPWM1 通道4输出/捕获输入脚
PC.9	PC.9	I/O	MFP0	通用数字I/O脚
	EBI_ADR7	O	MFP2	EBI 地址总线位 7.
	UART6_nCTS	I	MFP5	UART6 禁止发送输入管脚
	UART3_RXD	I	MFP7	UART3 数据接收输入脚
	CAN1_RXD	I	MFP9	CAN1 总线接收输入
	EPWM1_CH3	I/O	MFP12	EPWM1 通道3输出/捕获输入脚
	EADC1_ST	I	MFP14	EADC1 外部触发输入
PC.10	PC.10	I/O	MFP0	通用数字I/O脚
	EBI_ADR6	O	MFP2	EBI 地址总线位 6.
	UART6_nRTS	O	MFP5	UART6 请求发送输出脚
	UART3_TXD	O	MFP7	UART3 数据发送输出脚.
	CAN1_TXD	O	MFP9	CAN1总线发送输出
	ECAP1_IC0	I	MFP11	增强型捕获计数器 1 输入脚0
	EPWM1_CH2	I/O	MFP12	EPWM1 通道2 输出/捕获输入
	EADC1_ST	I	MFP14	EADC1 外部触发输入
PC.11	PC.11	I/O	MFP0	通用数字I/O脚
	EBI_ADR5	O	MFP2	EBI 地址总线位 5.
	UART0_RXD	I	MFP3	UART0 数据接收输入脚
	I²C0_SDA	I/O	MFP4	I2C0 数据输入/输出管脚
	UART6_RXD	I	MFP5	UART6 数据接收输入脚
	ECAP1_IC1	I	MFP11	增强型捕获计数器 1 输入 1 管脚.
	EPWM1_CH1	I/O	MFP12	EPWM1 通道1 输出/捕获输入
	ACMP1_O	O	MFP14	模拟比较器1输出脚
PC.12	PC.12	I/O	MFP0	通用数字I/O脚
	EBI_ADR4	O	MFP2	EBI 地址总线位 4.

	管脚名称	类型	MFP	描述
	UART0_TXD	O	MFP3	UART0 数据发送输出脚.
	I²C0_SCL	I/O	MFP4	I2C0 时钟管脚
	UART6_TXD	O	MFP5	UART6 数据发送输出脚.
	SC0_nCD	I	MFP9	智能卡 0 卡检测管脚.
	ECAP1_IC2	I	MFP11	增强型捕获计数器 1 输入 2 管脚.
	EPWM1_CH0	I/O	MFP12	EPWM1 通道0 输出/捕获输入
	ACMP0_O	O	MFP14	模拟比较器 0输出脚
PC.13	PC.13	I/O	MFP0	通用数字I/O脚
	EADC1_CH3	A	MFP1	EADC1 通道3 模拟输入
	EBI_ADR10	O	MFP2	EBI 地址总线位 10.
	SPI2_I2SMCLK	I/O	MFP4	SPI2 I²S 主机时钟输出脚
	CAN1_TXD	O	MFP5	CAN1总线发送输出
	UART2_TXD	O	MFP7	UART2 数据发送输出脚.
	BPWM0_CH4	I/O	MFP9	BPWM0 通道4输出/捕获输入脚
	CLKO	O	MFP13	时钟输出
	EADC0_ST	I	MFP14	EADC0 外部触发输入
PC.14	PC.14	I/O	MFP0	通用数字I/O脚
	EBI_AD11	I/O	MFP2	EBI地址/数据总线位11.
	SPI0_I2SMCLK	I/O	MFP4	SPI0 I²S 主机时钟输出脚
	QSPI0_CLK	I/O	MFP6	Quad SPI0 串行时钟脚
	EPWM0_SYNC_IN	I	MFP11	EPWM0 计数器同步触发输入脚
	ETM_TRACE_CLK	I	MFP12	ETM接收追踪时钟输入管脚
	TM1	I/O	MFP13	Timer1 事件计数输入/反转输出脚
	USB_VBUS_ST	I	MFP14	USB 外部VBUS 电源状态管脚.
PD.0	PD.0	I/O	MFP0	通用数字I/O脚
	EBI_AD13	I/O	MFP2	EBI地址/数据总线位13.
	SPI0_MOSI	I/O	MFP4	SPI0 MOSI (主机输出，从机输入) 管脚.
	UART3_RXD	I	MFP5	UART3 数据接收输入脚
	I²C2_SDA	I/O	MFP6	I²C2 数据输入/输出脚
	TM2	I/O	MFP14	Timer2 事件计数输入/反转输出脚
PD.1	PD.1	I/O	MFP0	通用数字I/O脚

	管脚名称	类型	MFP	描述
	EBI_AD12	I/O	MFP2	EBI地址/数据总线位12.
	SPI0_MISO	I/O	MFP4	SPI0 MISO（主机输入，从机输出）脚.
	UART3_TXD	O	MFP5	UART3 数据发送输出脚.
	I²C2_SCL	I/O	MFP6	I²C2 时钟脚
PD.2	PD.2	I/O	MFP0	通用数字I/O脚
	EBI_AD11	I/O	MFP2	EBI地址/数据总线位11.
	SPI0_CLK	I/O	MFP4	SPI0 串行时钟脚
	UART3_nCTS	I	MFP5	UART3 禁止发送输入管脚
	UART0_RXD	I	MFP9	UART0 数据接收输入脚
PD.3	PD.3	I/O	MFP0	通用数字I/O脚
	EBI_AD10	I/O	MFP2	EBI地址/数据总线位10.
	SPI0_SS	I/O	MFP4	SPI0从机选择管脚.
	UART3_nRTS	O	MFP5	UART3 请求发送输出脚
	UART0_TXD	O	MFP9	UART0 数据发送输出脚.
PD.8	PD.8	I/O	MFP0	通用数字I/O脚
	EBI_AD6	I/O	MFP2	EBI地址/数据总线位6.
	I²C2_SDA	I/O	MFP3	I²C2 数据输入/输出脚
	UART2_nRTS	O	MFP4	UART2 请求发送输出脚
	UART7_RXD	I	MFP5	UART7 数据接收输入脚
	CAN2_RXD	I	MFP6	CAN2 总线接收输入
PD.9	PD.9	I/O	MFP0	通用数字I/O脚
	EBI_AD7	I/O	MFP2	EBI地址/数据总线位7.
	I²C2_SCL	I/O	MFP3	I²C2 时钟脚
	UART2_nCTS	I	MFP4	UART2 禁止发送输入管脚
	UART7_TXD	O	MFP5	UART7 数据发送输出脚.
	CAN2_TXD	O	MFP6	CAN2总线发送输出
PD.10	PD.10	I/O	MFP0	通用数字I/O脚
	EADC1_CH0	A	MFP1	EADC1 通道0 模拟输入
	EBI_nCS2	O	MFP2	EBI片选2输出脚
	UART1_RXD	I	MFP3	UART1数据接收输入管脚
	CAN0_RXD	I	MFP4	CAN0 总线接收输入

	管脚名称	类型	MFP	描述
	QEI0_B	I	MFP10	正交编码器0 相位B 输入
	INT7	I	MFP15	外部中断7 输入脚
PD.11	PD.11	I/O	MFP0	通用数字I/O脚
	EADC1_CH1	A	MFP1	EADC1 通道1 模拟输入
	EBI_nCS1	O	MFP2	EBI片选1输出管脚
	UART1_TXD	O	MFP3	UART1 数据发送输出脚
	CAN0_TXD	O	MFP4	CAN0总线发送输出
	QEI0_A	I	MFP10	正交编码器0 相位A输入
	INT6	I	MFP15	外部中断6 输入脚
PD.12	PD.12	I/O	MFP0	通用数字I/O脚
	EADC1_CH2	A	MFP1	EADC1 通道2 模拟输入
	EBI_nCS0	O	MFP2	EBI片选0输出脚
	CAN1_RXD	I	MFP5	CAN1 总线接收输入
	UART2_RXD	I	MFP7	UART2 数据接收输入脚
	BPWM0_CH5	I/O	MFP9	BPWM0 通道5 输出/捕获输入
	QEI0_INDEX	I	MFP10	正交编码器0 索引输入
	CLKO	O	MFP13	时钟输出
	EADC0_ST	I	MFP14	EADC0 外部触发输入
	INT5	I	MFP15	外部中断5 输入脚
PD.13	PD.13	I/O	MFP0	通用数字I/O脚
	EBI_AD10	I/O	MFP2	EBI地址/数据总线位10.
	SD0_nCD	I	MFP3	SD/SDIO0 卡检测输入管脚
	SPI0_I2SMCLK	I/O	MFP4	SPI0 I^2S 主机时钟输出脚
	SPI1_I2SMCLK	I/O	MFP5	SPI1 I^2S 主机时钟输出脚
PD.14	PD.14	I/O	MFP0	通用数字I/O脚
	EBI_nCS0	O	MFP2	EBI片选0输出脚
	SPI0_I2SMCLK	I/O	MFP5	SPI0 I^2S 主机时钟输出脚
	EPWM0_CH4	I/O	MFP11	EPWM0 通道4输出/捕获输入脚
PE.0	PE.0	I/O	MFP0	通用数字I/O脚
	EBI_AD11	I/O	MFP2	EBI地址/数据总线位11.
	QSPI0_MOSI0	I/O	MFP3	Quad SPI0 MOSI0 (主机输出，从机输入) 管脚.

	管脚名称	类型	MFP	描述
	I²S0_MCLK	O	MFP5	I2S0 主时钟输出脚
	SPI1_MOSI	I/O	MFP6	SPI1 MOSI (主机输出，从机输入) 管脚.
	UART3_RXD	I	MFP7	UART3 数据接收输入脚
	I²C1_SDA	I/O	MFP8	I²C1 数据输入/输出脚
	UART4_nRTS	O	MFP9	UART4 请求发送输出脚
PE.1	PE.1	I/O	MFP0	通用数字I/O脚
	EBI_AD10	I/O	MFP2	EBI地址/数据总线位10.
	QSPI0_MISO0	I/O	MFP3	Quad SPI0 MISO0（主机输入，从机输出）脚.
	I²S0_BCLK	O	MFP5	I2S0位时钟输出管脚
	SPI1_MISO	I/O	MFP6	SPI1 MISO（主机输入，从机输出）脚.
	UART3_TXD	O	MFP7	UART3 数据发送输出脚.
	I²C1_SCL	I/O	MFP8	I²C1 时钟脚
	UART4_nCTS	I	MFP9	UART4 禁止发送输入管脚
PE.2	PE.2	I/O	MFP0	通用数字I/O脚
	EBI_ALE	O	MFP2	EBI地址锁存使能输出脚
	SD0_DAT0	I/O	MFP3	SD/SDIO0 数据线位0
	SC0_CLK	O	MFP6	智能卡0时钟管脚
	UART6_nCTS	I	MFP8	UART6 禁止发送输入管脚
	UART7_RXD	I	MFP9	UART7 数据接收输入脚
	QEI0_B	I	MFP11	正交编码器0 相位B 输入
	EPWM0_CH5	I/O	MFP12	EPWM0 通道5 输出/捕获输入
	BPWM0_CH0	I/O	MFP13	BPWM0 通道0 输出/捕获输入
PE.3	PE.3	I/O	MFP0	通用数字I/O脚
	EBI_MCLK	O	MFP2	EBI 外部时钟输出脚.
	SD0_DAT1	I/O	MFP3	SD/SDIO0 数据线位1
	SC0_DAT	I/O	MFP6	智能卡0数据输入脚
	UART6_nRTS	O	MFP8	UART6 请求发送输出脚
	UART7_TXD	O	MFP9	UART7 数据发送输出脚.
	QEI0_A	I	MFP11	正交编码器0 相位A输入
	EPWM0_CH4	I/O	MFP12	EPWM0 通道4输出/捕获输入脚
	BPWM0_CH1	I/O	MFP13	BPWM0 通道1 输出/捕获输入

	管脚名称	类型	MFP	描述
PE.4	PE.4	I/O	MFP0	通用数字I/O脚
	EBI_nWR	O	MFP2	EBI 写使能输出脚
	SD0_DAT2	I/O	MFP3	SD/SDIO0 数据线位2
	SC0_RST	O	MFP6	智能卡0复位脚
	UART6_RXD	I	MFP8	UART6 数据接收输入脚
	UART7_nCTS	I	MFP9	UART7 禁止发送输入管脚
	QEI0_INDEX	I	MFP11	正交编码器0 索引输入
	EPWM0_CH3	I/O	MFP12	EPWM0 通道3输出/捕获输入脚
	BPWM0_CH2	I/O	MFP13	BPWM0 通道2 输出/捕获输入
PE.5	PE.5	I/O	MFP0	通用数字I/O脚
	EBI_nRD	O	MFP2	EBI 读使能输出脚
	SD0_DAT3	I/O	MFP3	SD/SDIO0数据线位3
	SC0_PWR	O	MFP6	智能卡0电源脚
	UART6_TXD	O	MFP8	UART6 数据发送输出脚.
	UART7_nRTS	O	MFP9	UART7 请求发送输出脚
	QEI1_B	I	MFP11	正交编码器1 相位B 输入
	EPWM0_CH2	I/O	MFP12	EPWM0 通道2输出/捕获输入脚
	BPWM0_CH3	I/O	MFP13	BPWM0 通道3输出/捕获输入脚
PE.6	PE.6	I/O	MFP0	通用数字I/O脚
	SD0_CLK	O	MFP3	SD/SDIO0 时钟输出脚
	SC0_nCD	I	MFP6	智能卡 0 卡检测管脚.
	UART5_RXD	I	MFP8	UART5 数据接收输入管脚
	CAN1_RXD	I	MFP9	CAN1 总线接收输入
	QEI1_A	I	MFP11	正交编码器1 相位A输入
	EPWM0_CH1	I/O	MFP12	EPWM0 通道1输出/捕获输入脚
	BPWM0_CH4	I/O	MFP13	BPWM0 通道4输出/捕获输入脚
PE.7	PE.7	I/O	MFP0	通用数字I/O脚
	SD0_CMD	I/O	MFP3	SD/SDIO0 命令/响应脚
	UART5_TXD	O	MFP8	UART5数据发送输出脚
	CAN1_TXD	O	MFP9	CAN1总线发送输出
	QEI1_INDEX	I	MFP11	正交编码器1 索引输入

	管脚名称	类型	MFP	描述
	EPWM0_CH0	I/O	MFP12	EPWM0 通道0输出/捕获输入
	BPWM0_CH5	I/O	MFP13	BPWM0 通道5 输出/捕获输入
PE.8	PE.8	I/O	MFP0	通用数字I/O脚
	EBI_ADR10	O	MFP2	EBI 地址总线位 10.
	I²S0_BCLK	O	MFP4	I2S0位时钟输出管脚
	SPI2_CLK	I/O	MFP5	SPI2 串行时钟管脚
	UART2_TXD	O	MFP7	UART2 数据发送输出脚.
	EPWM0_CH0	I/O	MFP10	EPWM0 通道0输出/捕获输入
	EPWM0_BRAKE0	I	MFP11	EPWM0 刹车0 输入脚
	ECAP0_IC0	I	MFP12	增强型捕获计数器 0 输入脚0
	TRACE_DATA3	O	MFP14	ETM跟踪数据3 输出管脚
PE.9	PE.9	I/O	MFP0	通用数字I/O脚
	EBI_ADR11	O	MFP2	EBI 地址总线位 11.
	I²S0_MCLK	O	MFP4	I2S0 主时钟输出脚
	SPI2_MISO	I/O	MFP5	SPI2 MISO(主机输入，从机输出)管脚
	UART2_RXD	I	MFP7	UART2 数据接收输入脚
	EPWM0_CH1	I/O	MFP10	EPWM0 通道1输出/捕获输入脚
	EPWM0_BRAKE1	I	MFP11	EPWM0 刹车1 输入脚
	ECAP0_IC1	I	MFP12	增强型捕获计数器 0 输入 1 管脚.
	TRACE_DATA2	O	MFP14	ETM跟踪数据2 输出管脚
PE.10	PE.10	I/O	MFP0	通用数字I/O脚
	EBI_ADR12	O	MFP2	EBI 地址总线位 12.
	I²S0_DI	I	MFP4	I2S0 数据输入脚
	SPI2_MOSI	I/O	MFP5	SPI2 MOSI (主机输出，从机输入) 管脚
	UART3_TXD	O	MFP7	UART3 数据发送输出脚.
	EPWM0_CH2	I/O	MFP10	EPWM0 通道2输出/捕获输入脚
	EPWM1_BRAKE0	I	MFP11	EPWM1 刹车0 输入脚
	ECAP0_IC2	I	MFP12	增强型捕获计数器 0 输入 2 管脚.
	TRACE_DATA1	O	MFP14	ETM跟踪数据1 输出管脚
PE.11	PE.11	I/O	MFP0	通用数字I/O脚
	EBI_ADR13	O	MFP2	EBI 地址总线位 13.

	管脚名称	类型	MFP	描述
	I²S0_DO	O	MFP4	I²S0 数据输出脚
	SPI2_SS	I/O	MFP5	SPI2从机选择管脚.
	UART3_RXD	I	MFP7	UART3 数据接收输入脚
	UART1_nCTS	I	MFP8	UART1 禁止发送输入管脚
	EPWM0_CH3	I/O	MFP10	EPWM0 通道3输出/捕获输入脚
	EPWM1_BRAKE1	I	MFP11	EPWM1 刹车1输入管脚
	ECAP1_IC2	I	MFP13	增强型捕获计数器 1 输入 2 管脚.
	TRACE_DATA0	O	MFP14	ETM跟踪数据0 输出管脚
PE.12	PE.12	I/O	MFP0	通用数字I/O脚
	EBI_ADR14	O	MFP2	EBI 地址总线位 14.
	I²S0_LRCK	O	MFP4	I²S0左右通道时钟输出脚.
	SPI2_I2SMCLK	I/O	MFP5	SPI2 I²S 主机时钟输出脚
	UART1_nRTS	O	MFP8	UART1 请求发送输出脚
	EPWM0_CH4	I/O	MFP10	EPWM0 通道4输出/捕获输入脚
	ECAP1_IC1	I	MFP13	增强型捕获计数器 1 输入 1 管脚.
	TRACE_CLK	O	MFP14	ETM跟踪时钟输出脚
PE.13	PE.13	I/O	MFP0	通用数字I/O脚
	EBI_ADR15	O	MFP2	EBI 地址总线位 15.
	I²C0_SCL	I/O	MFP4	I2C0 时钟管脚
	UART4_nRTS	O	MFP5	UART4 请求发送输出脚
	UART1_TXD	O	MFP8	UART1 数据发送输出脚
	EPWM0_CH5	I/O	MFP10	EPWM0 通道5 输出/捕获输入
	EPWM1_CH0	I/O	MFP11	EPWM1 通道0 输出/捕获输入
	BPWM1_CH5	I/O	MFP12	EPWM1 通道5输出/捕获输入
	ECAP1_IC0	I	MFP13	增强型捕获计数器 1 输入脚0
PE.14	PE.14	I/O	MFP0	通用数字I/O脚
	EBI_AD8	I/O	MFP2	EBI地址/数据总线位8.
	UART2_TXD	O	MFP3	UART2 数据发送输出脚.
	CAN0_TXD	O	MFP4	CAN0总线发送输出
	UART6_TXD	O	MFP6	UART6 数据发送输出脚.
PE.15	PE.15	I/O	MFP0	通用数字I/O脚

	管脚名称	类型	MFP	描述
	EBI_AD9	I/O	MFP2	EBI地址/数据总线位9.
	UART2_RXD	I	MFP3	UART2 数据接收输入脚
	CAN0_RXD	I	MFP4	CAN0 总线接收输入
	UART6_RXD	I	MFP6	UART6 数据接收输入脚
PF.0	PF.0	I/O	MFP0	通用数字I/O脚
	UART1_TXD	O	MFP2	UART1 数据发送输出脚
	I^2C1_SCL	I/O	MFP3	I^2C1 时钟脚
	UART0_TXD	O	MFP4	UART0 数据发送输出脚.
	BPWM1_CH0	I/O	MFP12	BPWM1 通道0 输出/捕获输入
	ICE_DAT	I/O	MFP14	ADC 外部触发输入
PF.1	PF.1	I/O	MFP0	通用数字I/O脚
	UART1_RXD	I	MFP2	UART1数据接收输入管脚
	I^2C1_SDA	I/O	MFP3	I^2C1 数据输入/输出脚
	UART0_RXD	I	MFP4	UART0 数据接收输入脚
	BPWM1_CH1	I/O	MFP12	BPWM1 通道1 输出/捕获输入
	ICE_CLK	I/O	MFP14	ADC 外部触发输入
PF.2	PF.2	I/O	MFP0	通用数字I/O脚
	EBI_nCS1	O	MFP2	EBI片选1输出管脚
	UART0_RXD	I	MFP3	UART0 数据接收输入脚
	I^2C0_SDA	I/O	MFP4	I2C0 数据输入/输出管脚
	QSPI0_CLK	I/O	MFP5	Quad SPI0 串行时钟脚
	XT1_OUT	O	MFP10	外部 4~24 MHz (高速) 晶振输出脚
	BPWM1_CH1	I/O	MFP11	BPWM1 通道1 输出/捕获输入
PF.3	PF.3	I/O	MFP0	通用数字I/O脚
	EBI_nCS0	O	MFP2	EBI片选0输出脚
	UART0_TXD	O	MFP3	UART0 数据发送输出脚.
	I^2C0_SCL	I/O	MFP4	I2C0 时钟管脚
	XT1_IN	I	MFP10	外部 4~24 MHz (高速) 晶振输入脚
	BPWM1_CH0	I/O	MFP11	BPWM1 通道0 输出/捕获输入
PF.4	PF.4	I/O	MFP0	通用数字I/O脚
	UART2_TXD	O	MFP2	UART2 数据发送输出脚.

	管脚名称	类型	MFP	描述
	UART2_nRTS	O	MFP4	UART2 请求发送输出脚
	EPWM0_CH1	I/O	MFP7	EPWM0 通道1输出/捕获输入脚
	BPWM0_CH5	I/O	MFP8	BPWM0 通道5 输出/捕获输入
	X32_OUT	O	MFP10	外部 32.768 kHz 晶振输出脚
	EADC1_ST	I	MFP11	EADC1 外部触发输入
PF.5	PF.5	I/O	MFP0	通用数字I/O脚
	UART2_RXD	I	MFP2	UART2 数据接收输入脚
	UART2_nCTS	I	MFP4	UART2 禁止发送输入管脚
	EPWM0_CH0	I/O	MFP7	EPWM0 通道0输出/捕获输入
	BPWM0_CH4	I/O	MFP8	BPWM0 通道4输出/捕获输入脚
	EPWM0_SYNC_OUT	O	MFP9	EPWM0计数器同步触发输出脚
	X32_IN	I	MFP10	外部 32.768 kHz 晶振输入脚
	EADC0_ST	I	MFP11	EADC0 外部触发输入
PF.6	PF.6	I/O	MFP0	通用数字I/O脚
	EBI_ADR19	O	MFP2	EBI 地址总线位 19.
	SC0_CLK	O	MFP3	智能卡0时钟管脚
	I^2S0_LRCK	O	MFP4	I^2S0左右通道时钟输出脚.
	SPI0_MOSI	I/O	MFP5	SPI0 MOSI (主机输出，从机输入) 管脚.
	UART4_RXD	I	MFP6	UART4 数据接收输入脚
	EBI_nCS0	O	MFP7	EBI片选0输出脚
	CAN2_RXD	I	MFP8	CAN2 总线接收输入
	TAMPER0	I/O	MFP10	TAMPER循环检测管脚 0.
PF.7	PF.7	I/O	MFP0	通用数字I/O脚
	EBI_ADR18	O	MFP2	EBI 地址总线位 18.
	SC0_DAT	I/O	MFP3	智能卡0数据输入脚
	I^2S0_DO	O	MFP4	I^2S0 数据输出脚
	SPI0_MISO	I/O	MFP5	SPI0 MISO（主机输入，从机输出）脚.
	UART4_TXD	O	MFP6	UART4 数据发送输出脚.
	CAP_DATA0	I	MFP7	图像数据输入总线位0.
	CAN2_TXD	O	MFP8	CAN2总线发送输出
	TAMPER1	I/O	MFP10	TAMPER循环检测管脚 1.

	管脚名称	类型	MFP	描述
PF.8	PF.8	I/O	MFP0	通用数字I/O脚
	EBI_ADR17	O	MFP2	EBI 地址总线位 17.
	SC0_RST	O	MFP3	智能卡0复位脚
	I^{2}S0_DI	I	MFP4	I2S0 数据输入脚
	SPI0_CLK	I/O	MFP5	SPI0 串行时钟脚
	UART5_nCTS	I	MFP6	UART5 禁止发送输入管脚
	CAP_DATA1	I	MFP7	图像数据输入总线位1.
	CAN1_RXD	I	MFP8	CAN1 总线接收输入
	TAMPER2	I/O	MFP10	TAMPER循环检测管脚 2.
PF.9	PF.9	I/O	MFP0	通用数字I/O脚
	EBI_ADR16	O	MFP2	EBI 地址总线位 16.
	SC0_PWR	O	MFP3	智能卡0电源脚
	I^{2}S0_MCLK	O	MFP4	I2S0 主时钟输出脚
	SPI0_SS	I/O	MFP5	SPI0从机选择管脚.
	UART5_nRTS	O	MFP6	UART5 请求发送输出脚
	CAP_DATA2	I	MFP7	图像数据输入总线位2.
	CAN1_TXD	O	MFP8	CAN1总线发送输出
PF.10	PF.10	I/O	MFP0	通用数字I/O脚
	EBI_ADR15	O	MFP2	EBI 地址总线位 15.
	SC0_nCD	I	MFP3	智能卡 0 卡检测管脚.
	I^{2}S0_BCLK	O	MFP4	I2S0位时钟输出管脚
	SPI0_I2SMCLK	I/O	MFP5	SPI0 I^{2}S 主机时钟输出脚
	UART5_RXD	I	MFP6	UART5 数据接收输入管脚
	CAP_DATA3	I	MFP7	图像数据输入总线位3.
PF.11	PF.11	I/O	MFP0	通用数字I/O脚
	EBI_ADR14	O	MFP2	EBI 地址总线位 14.
	SPI2_MOSI	I/O	MFP3	SPI2 MOSI (主机输出，从机输入) 管脚
	UART5_TXD	O	MFP6	UART5数据发送输出脚
	CAP_DATA4	I	MFP7	图像数据输入总线位4.
	TM3	I/O	MFP13	Timer3 事件计数输入/反转输出脚
PG.2	PG.2	I/O	MFP0	通用数字I/O脚

	管脚名称	类型	MFP	描述
	EBI_ADR11	O	MFP2	EBI 地址总线位 11.
	SPI2_SS	I/O	MFP3	SPI2从机选择管脚.
	I²C0_SMBAL	O	MFP4	I²C0 SMBus SMBALTER 管脚
	I²C1_SCL	I/O	MFP5	I²C1 时钟脚
	CAP_DATA7	I	MFP7	图像数据输入总线位7.
	TM0	I/O	MFP13	Timer0 事件计数输入/反转输出管脚
PG.3	PG.3	I/O	MFP0	通用数字I/O脚
	EBI_ADR12	O	MFP2	EBI 地址总线位 12.
	SPI2_CLK	I/O	MFP3	SPI2 串行时钟管脚
	I²C0_SMBSUS	O	MFP4	I²C0 SMBus SMBSUS 管脚 (PMBus 控制管脚)
	I²C1_SDA	I/O	MFP5	I²C1 数据输入/输出脚
	CAP_DATA6	I	MFP7	图像数据输入总线位6.
	TM1	I/O	MFP13	Timer1 事件计数输入/反转输出脚
PG.4	PG.4	I/O	MFP0	通用数字I/O脚
	EBI_ADR13	O	MFP2	EBI 地址总线位 13.
	SPI2_MISO	I/O	MFP3	SPI2 MISO(主机输入，从机输出)管脚
	CAP_DATA5	I	MFP7	图像数据输入总线位5.
	TM2	I/O	MFP13	Timer2 事件计数输入/反转输出脚
PG.9	PG.9	I/O	MFP0	通用数字I/O脚
	EBI_AD0	I/O	MFP2	EBI地址/数据总线位0.
	QSPI1_MISO1	I/O	MFP5	Quad SPI1 MISO1（主机输入，从机输出）脚.
	CAP_PIXCLK	I	MFP7	图像采集接口像素时钟输入脚
	BPWM0_CH5	I/O	MFP12	BPWM0 通道5 输出/捕获输入
PG.10	PG.10	I/O	MFP0	通用数字I/O脚
	EBI_AD1	I/O	MFP2	EBI地址/数据总线位1.
	QSPI1_MOSI1	I/O	MFP5	Quad SPI1 MOSI1 (主机输出，从机输入) 管脚.
	CAP_SCLK	O	MFP7	图像采集接口传感器时钟管脚.
	BPWM0_CH4	I/O	MFP12	BPWM0 通道4输出/捕获输入脚
PG.11	PG.11	I/O	MFP0	通用数字I/O脚
	EBI_AD2	I/O	MFP2	EBI地址/数据总线位2.
	QSPI1_SS	I/O	MFP5	Quad SPI1从机选择管脚.

	管脚名称	类型	MFP	描述
	UART7_TXD	O	MFP6	UART7 数据发送输出脚.
	CAP_SFIELD	I	MFP7	图像输入接口SFIELD 输入脚
	BPWM0_CH3	I/O	MFP12	BPWM0 通道3输出/捕获输入脚
PG.12	PG.12	I/O	MFP0	通用数字I/O脚
	EBI_AD3	I/O	MFP2	EBI地址/数据总线位3.
	QSPI1_CLK	I/O	MFP5	Quad SPI1 串行时钟脚
	UART7_RXD	I	MFP6	UART7 数据接收输入脚
	CAP_VSYNC	I	MFP7	图像采集接口vsync 输入脚
	BPWM0_CH2	I/O	MFP12	BPWM0 通道2 输出/捕获输入
PG.13	PG.13	I/O	MFP0	通用数字I/O脚
	EBI_AD4	I/O	MFP2	EBI地址/数据总线位4.
	QSPI1_MISO0	I/O	MFP5	Quad SPI1 MISO0（主机输入，从机输出）脚.
	UART6_TXD	O	MFP6	UART6 数据发送输出脚.
	CAP_HSYNC	I	MFP7	图像采集接口hsync 输入脚
	BPWM0_CH1	I/O	MFP12	BPWM0 通道1 输出/捕获输入
PG.14	PG.14	I/O	MFP0	通用数字I/O脚
	EBI_AD5	I/O	MFP2	EBI地址/数据总线位5.
	QSPI1_MOSI0	I/O	MFP5	Quad SPI1 MOSI0 (主机输出，从机输入) 管脚.
	UART6_RXD	I	MFP6	UART6 数据接收输入脚
	BPWM0_CH0	I/O	MFP12	BPWM0 通道0 输出/捕获输入
PG.15	PG.15	I/O	MFP0	通用数字I/O脚
	CLKO	O	MFP14	时钟输出
	EADC0_ST	I	MFP15	EADC0 外部触发输入
PH.4	PH.4	I/O	MFP0	通用数字I/O脚
	EBI_ADR3	O	MFP2	EBI 地址总线位3
	SPI1_MISO	I/O	MFP3	SPI1 MISO（主机输入，从机输出）脚.
	UART7_nRTS	O	MFP4	UART7 请求发送输出脚
	UART6_TXD	O	MFP5	UART6 数据发送输出脚.
PH.5	PH.5	I/O	MFP0	通用数字I/O脚
	EBI_ADR2	O	MFP2	EBI 地址总线位2
	SPI1_MOSI	I/O	MFP3	SPI1 MOSI (主机输出，从机输入) 管脚.

	管脚名称	类型	MFP	描述
	UART7_nCTS	I	MFP4	UART7 禁止发送输入管脚
	UART6_RXD	I	MFP5	UART6 数据接收输入脚
PH.6	PH.6	I/O	MFP0	通用数字I/O脚
	EBI_ADR1	O	MFP2	EBI 地址总线位1
	SPI1_CLK	I/O	MFP3	SPI1 串行时钟脚
	UART7_TXD	O	MFP4	UART7 数据发送输出脚.
PH.7	PH.7	I/O	MFP0	通用数字I/O脚
	EBI_ADR0	O	MFP2	EBI地址总线位0
	SPI1_SS	I/O	MFP3	SPI1从机选择管脚.
	UART7_RXD	I	MFP4	UART7 数据接收输入脚
PH.8	PH.8	I/O	MFP0	通用数字I/O脚
	EBI_AD12	I/O	MFP2	EBI地址/数据总线位12.
	QSPI0_CLK	I/O	MFP3	Quad SPI0 串行时钟脚
	I^2S0_DI	I	MFP5	I2S0 数据输入脚
	SPI1_CLK	I/O	MFP6	SPI1 串行时钟脚
	UART3_nRTS	O	MFP7	UART3 请求发送输出脚
	I^2C1_SMBAL	O	MFP8	I^2C1 SMBus SMBALTER 管脚
	I^2C2_SCL	I/O	MFP9	I^2C2 时钟脚
	UART1_TXD	O	MFP10	UART1 数据发送输出脚
PH.9	PH.9	I/O	MFP0	通用数字I/O脚
	EBI_AD13	I/O	MFP2	EBI地址/数据总线位13.
	QSPI0_SS	I/O	MFP3	Quad SPI0从机选择管脚.
	I^2S0_DO	O	MFP5	I^2S0 数据输出脚
	SPI1_SS	I/O	MFP6	SPI1从机选择管脚.
	UART3_nCTS	I	MFP7	UART3 禁止发送输入管脚
	I^2C1_SMBSUS	O	MFP8	I^2C1 SMBus SMBSUS 管脚 (PMBus 控制管脚)
	I^2C2_SDA	I/O	MFP9	I^2C2 数据输入/输出脚
	UART1_RXD	I	MFP10	UART1数据接收输入管脚
PH.10	PH.10	I/O	MFP0	通用数字I/O脚
	EBI_AD14	I/O	MFP2	EBI地址/数据总线位14.
	QSPI0_MISO1	I/O	MFP3	Quad SPI0 MISO1（主机输入，从机输出）脚.

	管脚名称	类型	MFP	描述
	I²S0_LRCK	O	MFP5	I²S0左右通道时钟输出脚.
	SPI1_I2SMCLK	I/O	MFP6	SPI1 I²S 主机时钟输出脚
	UART4_TXD	O	MFP7	UART4 数据发送输出脚.
	UART0_TXD	O	MFP8	UART0 数据发送输出脚.
H.11	PH.11	I/O	MFP0	通用数字I/O脚
	EBI_AD15	I/O	MFP2	EBI地址/数据总线位15.
	QSPI0_MOSI1	I/O	MFP3	Quad SPI0 MOSI1 (主机输出，从机输入) 管脚.
	UART4_RXD	I	MFP7	UART4 数据接收输入脚
	UART0_RXD	I	MFP8	UART0 数据接收输入脚
	EPWM0_CH5	I/O	MFP11	EPWM0 通道5 输出/捕获输入

5 框图

5.1 M480框图

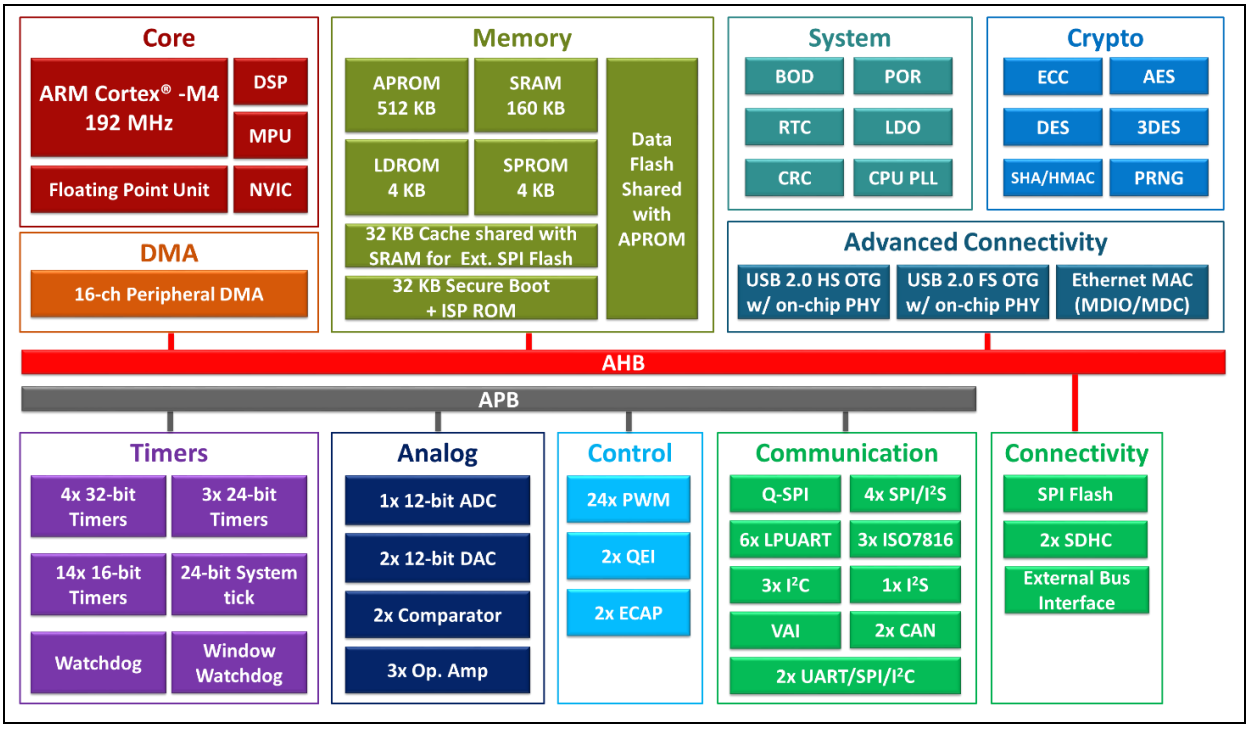


图 5.1-1 NuMicro®M480 框图 (M48xID/M48xGA)

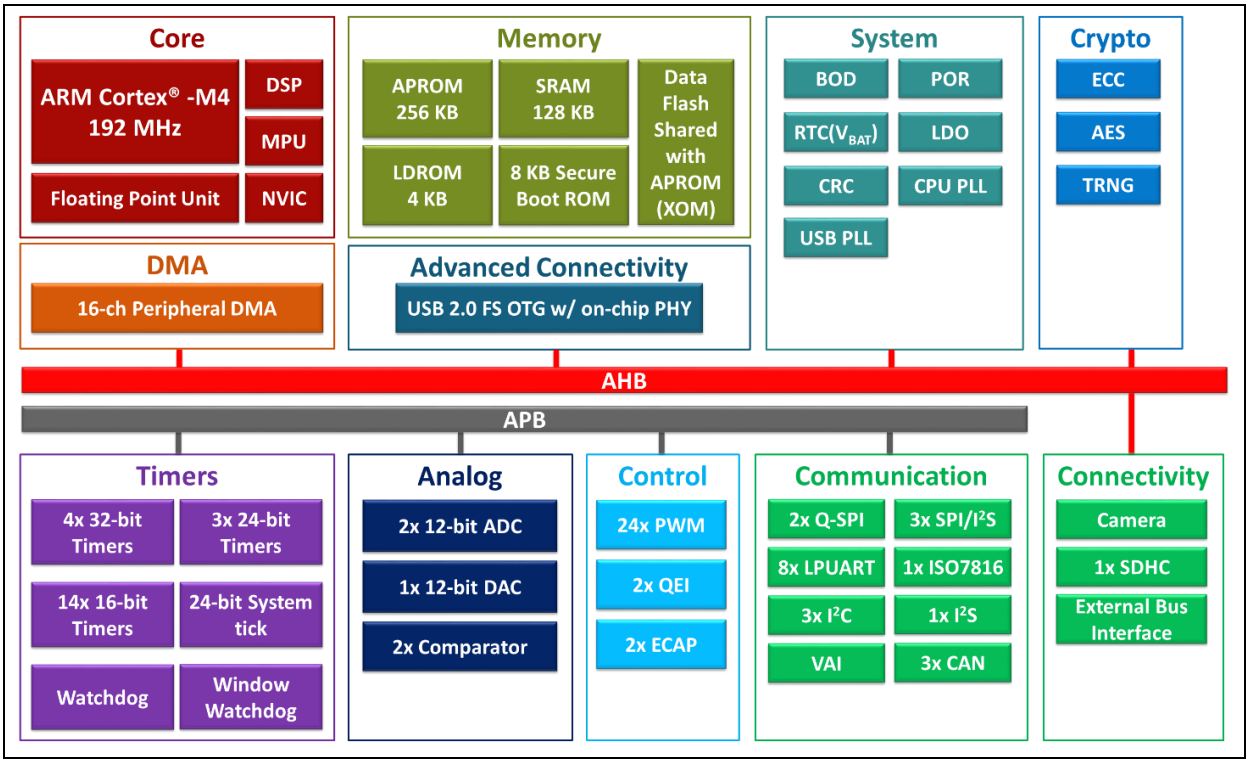


图 5.1-2 NuMicro®M480 框图 (M48xGC)

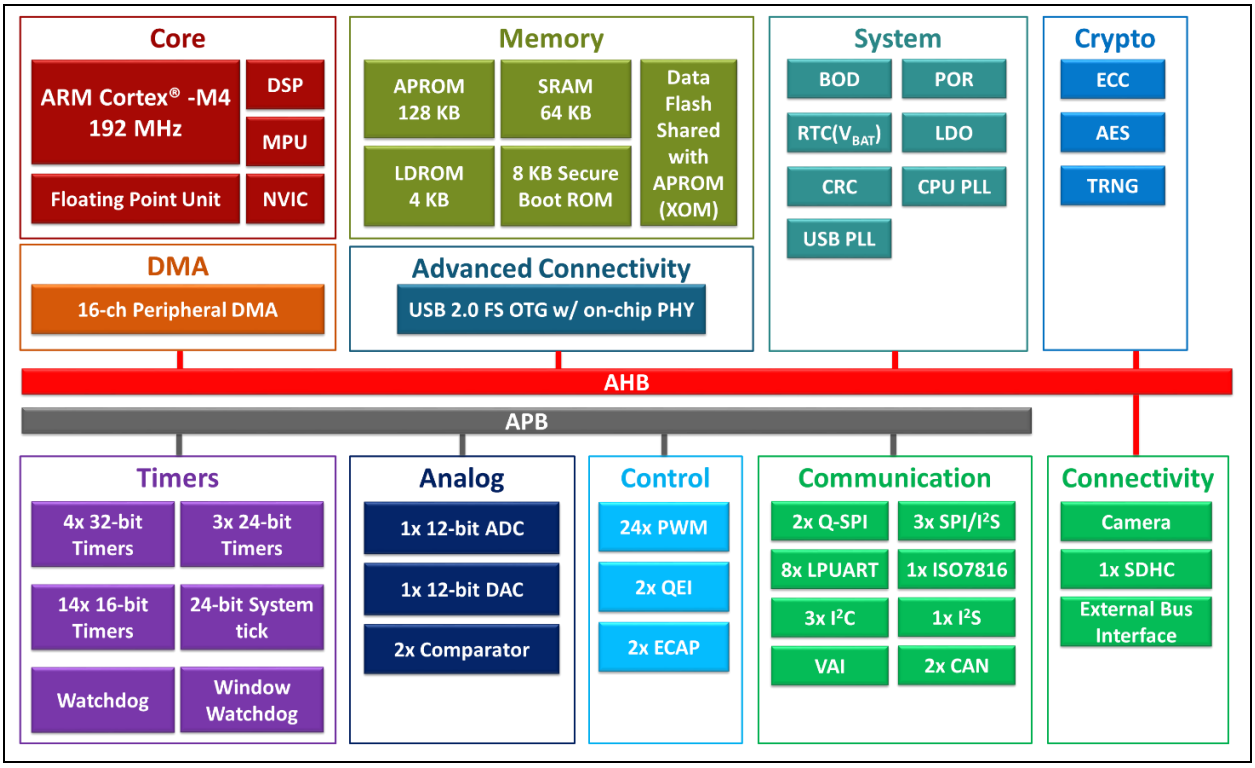


图 5.1-3 NuMicro® M480框图(M48xE8)

6 功能描述

6.1 Arm® Cortex®-M4F内核

Cortex®-M4处理器是多级流水线、32位精简指令处理器 ,三个 AMBA AHB-Lite接口达到最佳的并行处理，包含一个NVIC中断向量控制器。处理器有一个可选的硬件调试单元，可执行与其它Cortex-M处理器兼容的Rhumb指令。两种处理模式：线程模式Thread和句柄模式 Handler。进入异常时就进入句柄模式——异常只能在这种模式下返回。复位后是线程模式，异常返回后也可进入线程模式。M480的内核是Cortex®-M4F，又多了一个浮点运算功能。请参考 Cortex®-M4 和 Cortex®-M4F 处理器. 图6.1-1是功能框图.

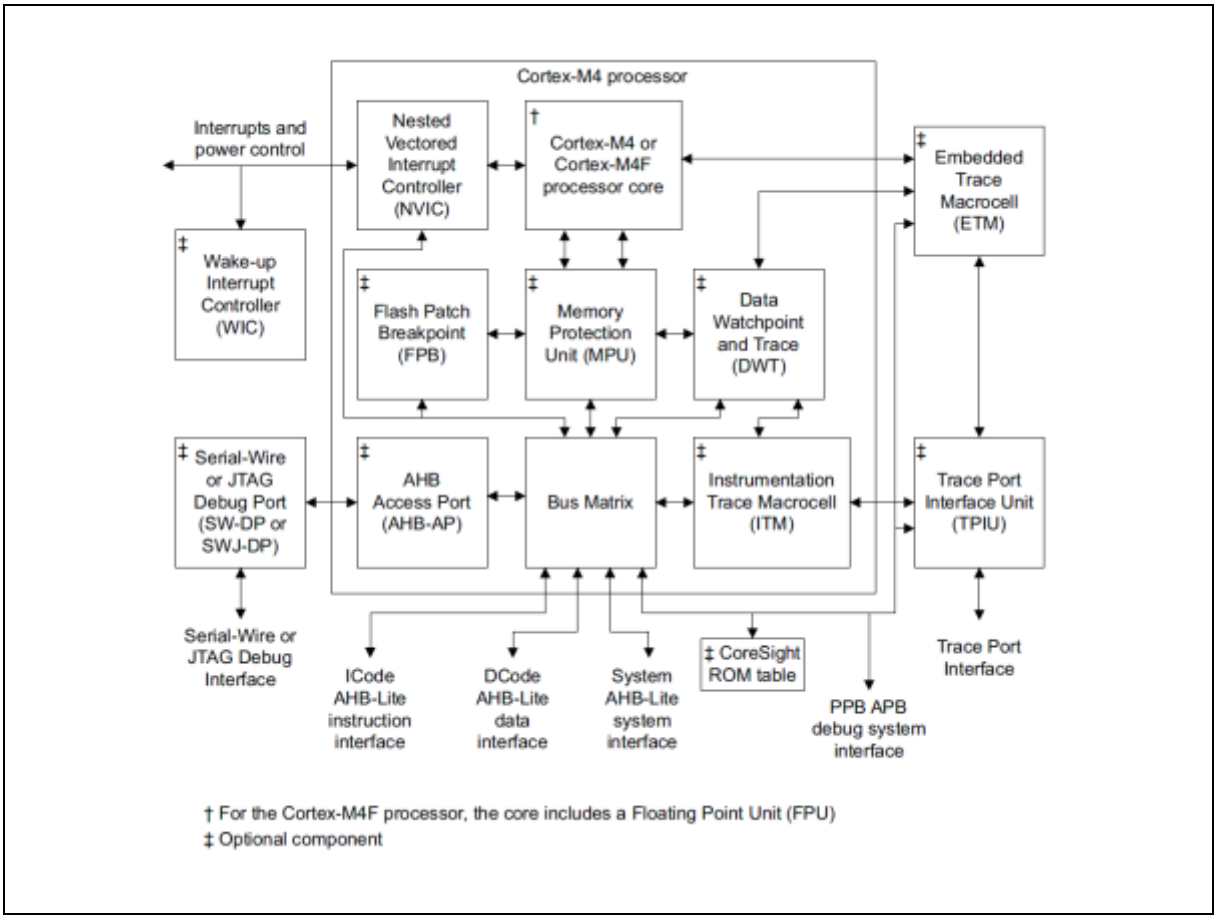


图 6.1-1Cortex®-M4F 框图

Cortex®-M4F 处理器特性:

- 低逻辑站数量，加速中断处理功能:
  - Thumb指令集，请参考*ARMv7-M Architecture Reference Manual*
  - Banked 栈指针
  - 有除法指令, SDIV ， UDIV
  - 句柄和线程模式
  - Thumb 指令和调试功能

- 支持中断/继续指令 LDM, STM, PUSH, 和 POP
- 中断上下文保存/退出恢复，参考*Interrupt Service Routine (ISR)* entry and exit
- 支持ARMv6 大端和小端数据访问
- 支持ARMv6非对齐访问

- Cortex®-M4F的浮点单元(FPU) 功能:
  - 单精度浮点运算，32位指令
  - 组合了乘/累加用于增加精度(用 MAC)
  - 硬件支持转换，加，减，带累加的乘，除，和平方根运算
  - 支持IEEE环绕模式
  - 32个32位寄存器，可组成16个双字寄存器-
  - 三级流水线
- NVIC可嵌套中断向量控制器，与处理器紧密接合，加速中断响应时间，功能包括：
  - 外中断可配置为 1 ~ 240
  - 优先先级可以有 3 ~ 8位
  - 优先级可动态改变
  - 可配置优先级组，组内相互之间不抢先
  - 支持尾链和高优先级先处理功能.
  - 硬件入栈上下文程状态寄存器
  - 支持唤醒功能
- MPU内存保护单元:
  - 可管理8个内存区域
  - 分区可使能禁止读写功能
  - 其它区域有缺省属性
- 调试单元:
  - 可调试访问所有内存空间和寄存器.
  - SWD接口或 JTAG 接口
  - 可选的FLASH断点功能FPB
  - 可选的数据观察和跟踪单元(DWT)
  - 可选的指令跟踪单元ITM支持printf() 风格的调试
  - 可选的跟踪单元TPIU可用于跟踪分析, 包含单线输出接口模式
  - 可选的嵌入跟踪单元ETM可跟踪指令
- 总线接口:
  - 三条AHB-Lite总线：指令总线，数据总线，系统总线

- 基于 APB 总线的各外设单独总线PPB
- 支持位段读写.
- 内存访问对齐
- 写缓存
- 多处理器系统独立传输

6.2 系统复位

6.2.1 概述

系统管理包括以下部分:

- 系统复位
- 系统电源分布
- SRAM内存结构
- 系统定时器(SysTick)
- 可嵌套中断向量控制(NVIC)
- 系统控制寄存器

6.2.2 系统复位

下面事件可以触发系统复位，可以通过读取SYS_RSTSTS 寄存器以知道复位源是哪个。硬件复位源来自外设信号，软件复位通过设置控制寄存器触发。

- 硬件复位源
  - 上电复位POR
  - 引脚nRESET 复位
  - WDT/WWDT 复位
  - 低压复位(LVR)
  - 欠压复位BOD
  - CPU锁死复位（Lockup）
- 软件复位源
  - 芯片复位：CHIPRST (SYS_IPRST0[0])写1，复位整个芯片。
  - MCU 复位，SYSRESETREQ (AIRCR[2])写1，复位CPU和外设，从APROM 或 LDROM 启动不变。
  - CPU 复位，CPURST (SYS_IPRST0[1])写1，仅复位 Cortex®-M4内核，外设不复位。

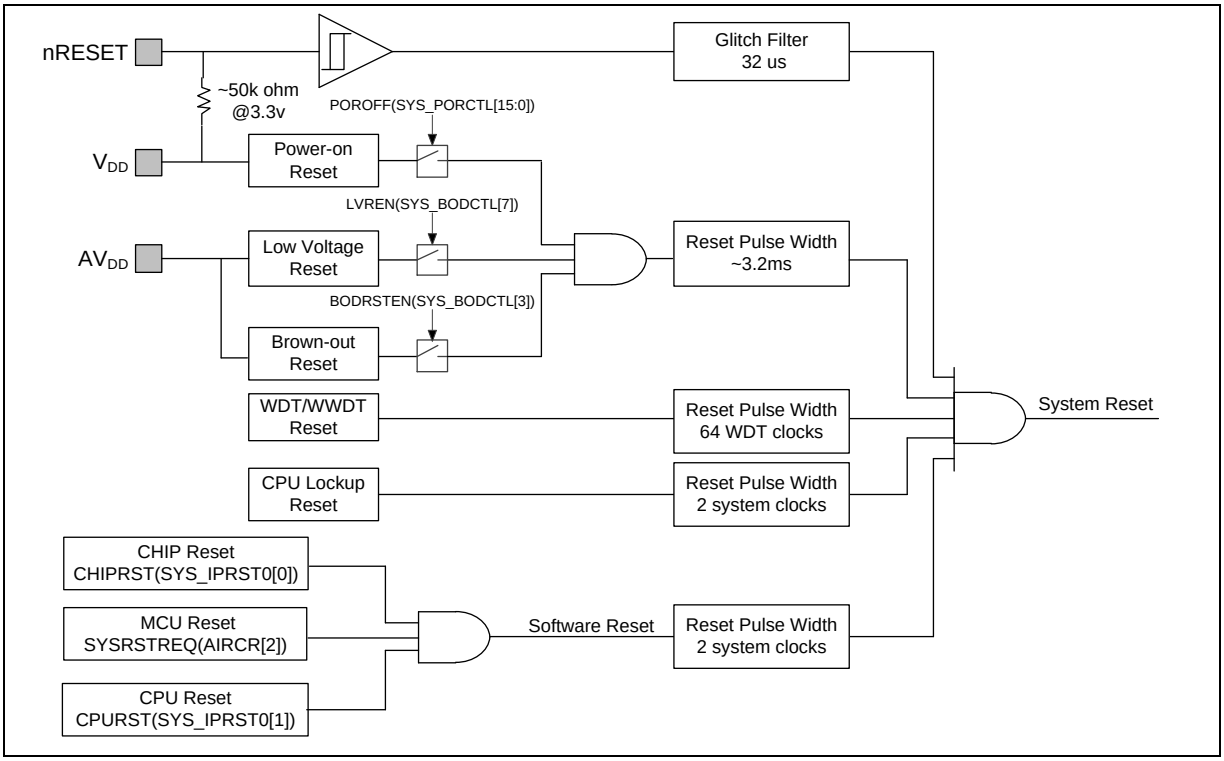


图 6.2-1 系统复位源

共9个复位源。见表6.2-1.

复位源寄存器	POR	NRESET	WDT	LVR	BOD	Lockup	CHIP	MCU	CPU
SYS_RSTSTS	Bit 0 = 1	Bit 1 = 1	Bit 2 = 1	Bit 3 = 1	Bit 4 = 1	Bit 8 = 1	Bit 0 = 1	Bit 5 = 1	Bit 7 = 1
CHIPRST (SYS_IPRST0[0])	0x0	-	-	-	-	-	-	-	-
BODEN (SYS_BODCTL[0])	从 CONFIG0 加载到此寄存器位	从 CONFIG0 加载到此寄存器位	从 CONFIG0 加载到此寄存器位	从 CONFIG0 加载到此寄存器位	-	从 CONFIG0 加载到此寄存器位	从 CONFIG0 加载到此寄存器位	从 CONFIG0 加载到此寄存器位	-
BODVL (SYS_BODCTL[2:1])									
BODRSTEN (SYS_BODCTL[3])									
HXTEN (CLK_PWRCTL[0])	从 CONFIG0 加载到此寄存器位	从 CONFIG0 加载到此寄存器位	从 CONFIG0 加载到此寄存器位	从 CONFIG0 加载到此寄存器位	从 CONFIG0 加载到此寄存器位	从 CONFIG0 加载到此寄存器位	从 CONFIG0 加载到此寄存器位	从 CONFIG0 加载到此寄存器位	
LXTEN (CLK_PWRCTL[1])	0x0	-	-	-	-	-	-	-	-
WDTCKEN (CLK_APBCLK0[0])	0x1	-	0x1	-	-	-	0x1	-	-
HCLKSEL	从 CONFIG0	从 CONFIG0	从 CONFIG0	从 CONFIG0	从 CONFIG0	从 CONFIG0	从 CONFIG0	从 CONFIG0	-

(CLK_CLKSEL0[2:0])	加载到此寄存器位	加载到此寄存器位	加载到此寄存器位	加载到此寄存器位	加载到此寄存器位	加载到此寄存器位	加载到此寄存器位	加载到此寄存器位	
WDTSEL (CLK_CLKSEL1[1:0])	0x3	0x3	-	-	-	-	-	-	-
HXTSTB (CLK_STATUS[0])	0x0	-	-	-	-	-	-	-	-
LXTSTB (CLK_STATUS[1])	0x0	-	-	-	-	-	-	-	-
PLLSTB (CLK_STATUS[2])	0x0	-	-	-	-	-	-	-	-
HIRCSTB (CLK_STATUS[4])	0x0	-	-	-	-	-	-	-	-
CLKSFAIL (CLK_STATUS[7])	0x0	0x0	-	-	-	-	-	-	-
RSTEN (WDT_CTL[1]) WDTEN (WDT_CTL[7])	从CONFIG0加载到此寄存器位	从CONFIG0加载到此寄存器位	从CONFIG0加载到此寄存器位	从CONFIG0加载到此寄存器位	从CONFIG0加载到此寄存器位	-	从CONFIG0加载到此寄存器位	-	-
WDT_CTL except bit 1 and bit 7.	0x0700	0x0700	0x0700	0x0700	0x0700	-	0x0700	-	-
WDT_ALTCTL	0x0000	0x0000	0x0000	0x0000	0x0000	-	0x0000	-	-
WWDT_RLDCNT	0x0000	0x0000	0x0000	0x0000	0x0000	-	0x0000	-	-
WWDT_CTL	0x3F0800	0x3F0800	0x3F0800	0x3F0800	0x3F0800	-	0x3F0800	-	-
WWDT_STATUS	0x0000	0x0000	0x0000	0x0000	0x0000	-	0x0000	-	-
WWDT_CNT	0x3F	0x3F	0x3F	0x3F	0x3F	-	0x3F	-	-
BS (FMC_ISPCTL[1]) BL (FMC_ISPCTL[16])	从CONFIG0加载到此寄存器位	从CONFIG0加载到此寄存器位	从CONFIG0加载到此寄存器位	从CONFIG0加载到此寄存器位	从CONFIG0加载到此寄存器位	-	从CONFIG0加载到此寄存器位	-	-
FMC_DFBA	从CONFIG1加载到此寄存器位	从CONFIG1加载到此寄存器位	从CONFIG1加载到此寄存器位	从CONFIG1加载到此寄存器位	从CONFIG1加载到此寄存器位	-	从CONFIG1加载到此寄存器位	-	-
CBS (FMC_ISPSTS[2:1))	从CONFIG0加载到此寄存器位	从CONFIG0加载到此寄存器位	从CONFIG0加载到此寄存器位	从CONFIG0加载到此寄存器位	从CONFIG0加载到此寄存器位	-	从CONFIG0加载到此寄存器位	-	-
VECMAP (FMC_ISPSTS[23:9])	从CONFIG0加载到此寄存器位	从CONFIG0加载到此寄存器位	从CONFIG0加载到此寄存器位	从CONFIG0加载到此寄存器位	从CONFIG0加载到此寄存器位	-	从CONFIG0加载到此寄存器位	-	-
Other Peripheral	复位值								-

Registers		
FMC Registers	复位值	
注意: '-' 表示保持原值		

表 6.2-1 寄存器复位值

6.2.2.1 *nRESET 复位*

任何时候都可以将引脚nRESET 持续拉低至0.2V_{DD}以下66us，异步复位芯片。引脚电压升至0.7 V_{DD}以上后，再保持66us后，PINRF(SYS_RSTSTS[1])置1。复位时序见图6.2-2

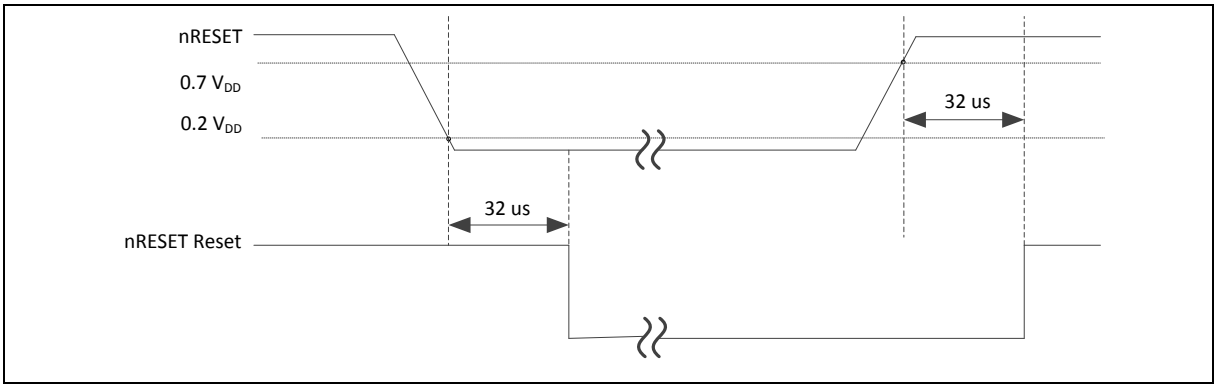


图 6.2-2 nRESET 复位时序

6.2.2.2 *上电复位 (POR)*

上电后POR电路让芯片保持复位，直到电压升至可工作的幅度，位PORF(SYS_RSTSTS[0])置1表明发生了上电，此位写1清0。复位时序见.图6.2-3

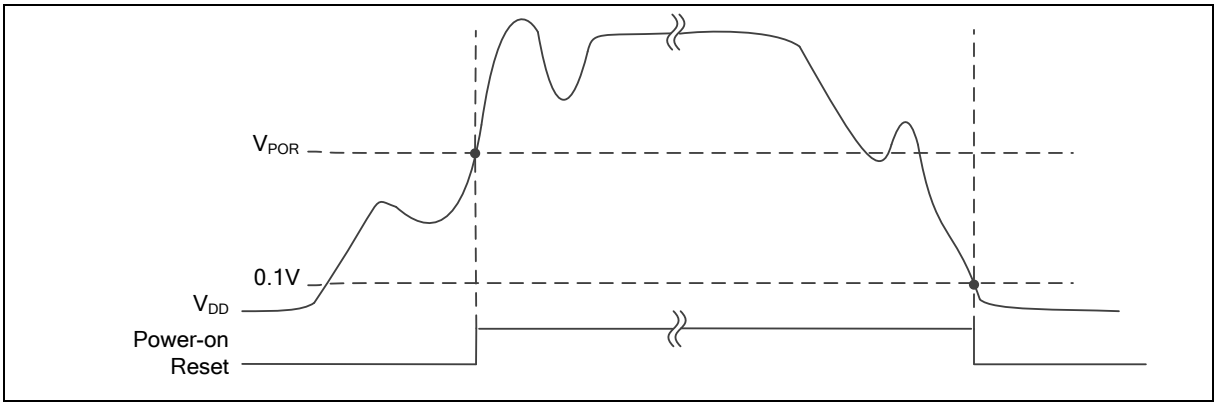


图 6.2-3(POR) 上电复位波形

6.2.2.3 *低电压复位 (LVR)*

如果LVREN (SYS_BODCTL[7]) =1使能了低压复位，200us后低压复位将开始工作。如果检测到AV_{DD}电压低于V_{LVR}超过LVRDGSEL(SYS_BODCTL[14:12])配置的防抖时间，将产生低压复位，直到电压高于

V_{LVR}的时间超过LVRDGSEL(SYS_BODCTL[14:12])配置的防抖时间后，CPU退出复位开始工作。缺省值低压复位使能但防抖时间为0。复位波形见图6.2-4

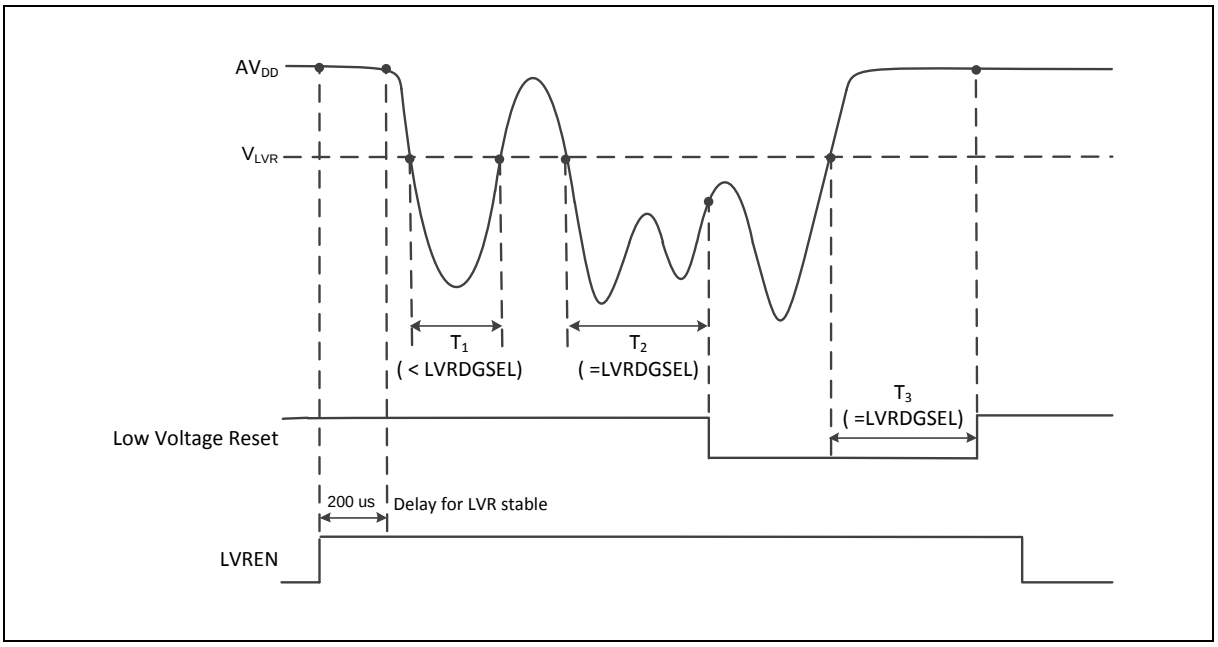


图 6.2-4 低电压复位 (LVR) 波形

6.2.2.4 欠压复位 (BOD 复位)

配置BODEN (SYS_BODCTL[0])可使能欠压复位功能，若 AV_{DD}低于BODVL (SYS_BODCTL[18:16])配置的电压V_{BOD}的时间，超过BODDGSEL (SYS_BODCTL[10:8])的值,将产生复位，直到电压高于V_{BOD}的时间长于BODDGSEL的值后，退出复位开始工作. 上电后或芯片复位后BODEN, BODVL 和BODRSTEN (SYS_BODCTL[3])的缺省值由CONFIG0的对应位加载.图6.2-5是复位时序波形.

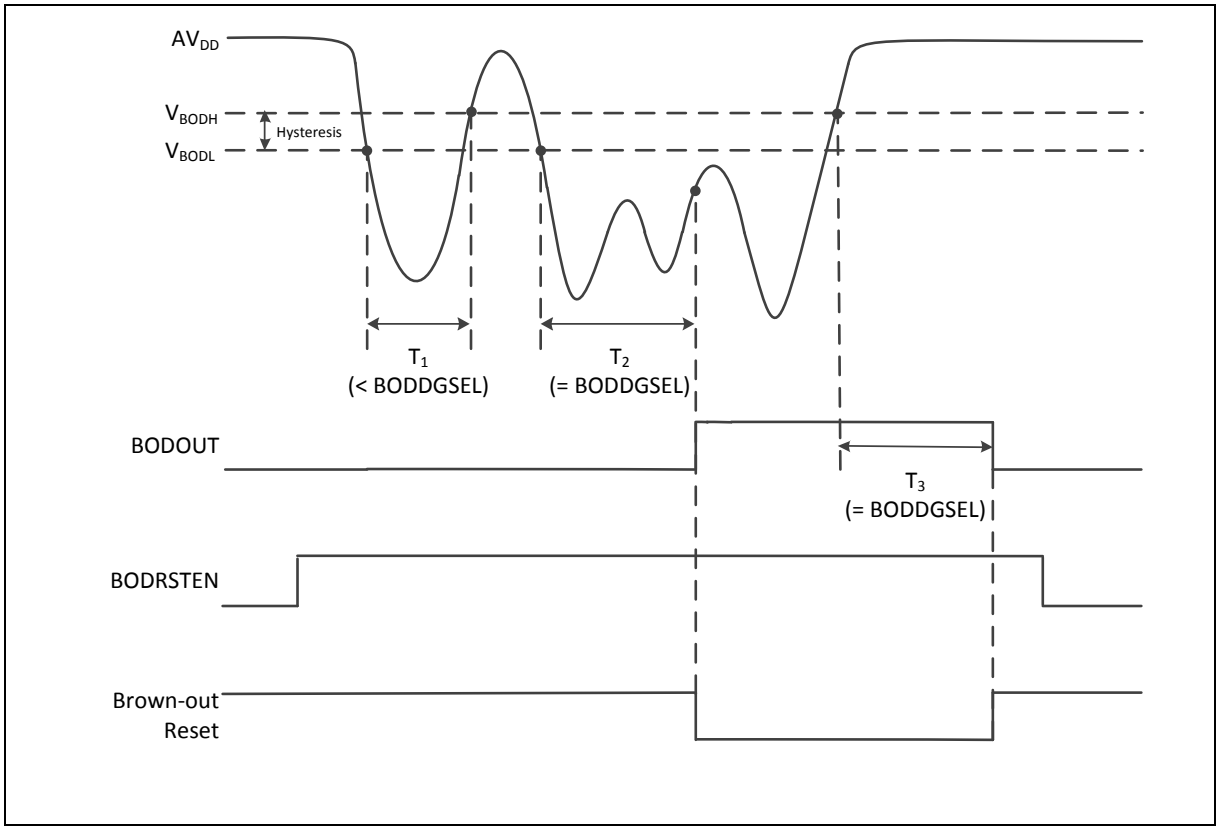


图 6.2-5 欠压侦测 (BOD) 波形

6.2.2.5 WDT复位

如果系统失控，WDT定时器溢出会产生复位，复位源寄存器WDTRF(SYS_RSTSTS[2])置1指示产生了WDT复位。

6.2.2.6 CPU 锁死复位

非调试状态出现硬件错误异常hardfault将产生CPU 锁死复位。Debug调试状态这个复位源将被忽略

6.2.2.7 CPU 复位, 芯片复位和MCU复位

CPURST(SYS_IPRST0[1])写1产生Cortex®-M4内核复位，所有外设都不复位，保持原工作状态不变。

CHIPRST(SYS_IPRST0[1])写1产生芯片复位，与重新上电一样。启动选项 BS(FMC_ISPCTL[1]) 将从 CONFIG0重新加载

SYSRESETREQ(AIRCR[2]) 写1产生 MCU 复位，与芯片复位不同的是，不会由CONFIG0加载 BS(FMC_ISPCTL[1]) 所以从 APROM 还是从 LDROM启动将不变。

6.2.3 系统电源分配

电源的四个部分:

- 模拟部分的电源由 AV_{DD}和 AV_{SS}提供
- 数字部分包括GPIO口和内核1.8 V由 V_{DD}和 V_{SS}供电

- USB 部分由VBUS 供电
- RTC部分和它的80个字节内存由V_{BAT}供电

靠近片内 LDO 和 VDD33的输出引脚，要有个电容。AV_{DD}的数值要与V_{DD}一样. 图6.2-6是M480电源分布框图.

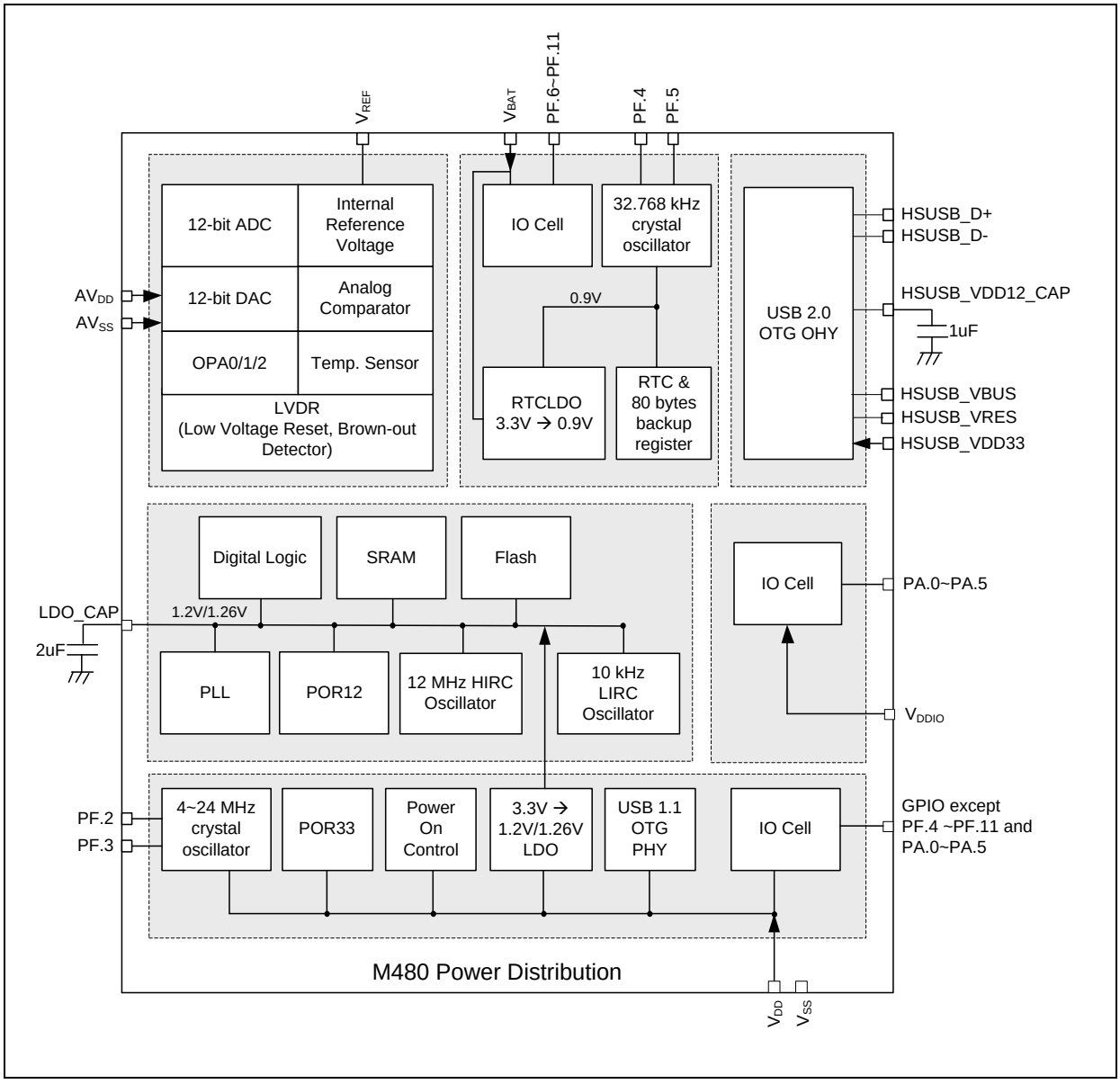


图 6.2-6 NuMicro®M480 电源分布图

注意:

1. 当 VBAT 电源第一次通电时，将发生开机复位，并复位所有 VBAT 域电路。VBAT 域中的 I/O (PF.4 ~ PF.11)将变为浮态，并在 VBAT 域中产生额外的泄漏。用户应先打开 VDD 电源，复位芯片并设置 I/O 控制，使这些 I/Os 成为静态状态，防止额外的泄漏。

2. 如果 VBAT 和 VDD 连接在一起，VBAT 域 I/O (PF.4 ~ PF.11)在通电过程中会出现无法预测的 1.5V 故障。为了防止这种不可预测的故障发生，用户应避免使用这些引脚作为其他 IC 的活动或非活动控制引脚。

6.2.4 电源模式和唤醒源

NuMicro® M480系列支持几种操作模式来降低功耗，如表6.2-2所列

模式	CPU 操作最大速度 (MHz)	LDO_CAP (V)	时钟禁止
正常模式	160	1.20	所有时钟通过控制寄存器禁止
增强模式	192	1.26	所有时钟通过控制寄存器禁止
空闲模式	CPU 进入睡眠模式	1.20/1.26	仅 CPU 时钟禁止
快速唤醒掉电模式 (FWPD)	CPU 进入深度睡眠模式	1.20/1.26	除了LIRC/LXT外，大部分时钟禁止。在时钟源选择LIRC/LXT的前提下.仅 RTC/WDT/Timer/UART外设时钟仍然使能。
正常掉电模式 (NPD)	CPU进入深度睡眠模式	1.20/1.26	除了LIRC/LXT外，大部分时钟禁止。在时钟源选择LIRC/LXT的前提下.仅 RTC/WDT/Timer/UART外设时钟仍然使能。
低耗电掉电模式 (LLPD)	CPU进入深度睡眠模式	0.9	除了LIRC/LXT外，大部分时钟禁止。在时钟源选择LIRC/LXT的前提下.仅 RTC/WDT/Timer/UART外设时钟仍然使能。
待机掉电模式 (SPD)	关机	悬浮不定	仅LIRC/LXT仍然使能，用于RTC 功能和唤醒定时器。
深度掉电模式 (DPD)	关机	悬浮不定	仅LIRC/LXT仍然使能，用于RTC 功能和唤醒定时器。

表 6.2-2 电源模式表

表6.2-3列举了进入不同工作模式的方法。上电后是正常工作模式。通过设置SLEEPDEEP (SCR[2]), PDEN (CLK_PWRCT:[7]) 和 PDMSEL (CLK_PMUCTL[2:0]) 和执行 WFI 指令用户可以进入各模式。

寄存器/指令模式	SLEEPDEEP (SCR[2])	PDEN (CLK_PWRCTL[7])	PDMSEL (CLK_PMUCTL[2:0])	CPU 运行 WFI 指令
正常模式	0	0	0	NO
空闲模式	0	0	0	YES
快速唤醒掉电模式	1	1	2	YES
正常掉电模式	1	1	0	YES
低耗电掉电模式	1	1	1	YES
待机掉电模式	1	1	4	YES
深度掉电模式	1	1	6	YES

表 6.2-3 电源模式区别表

在空闲模式和掉电模式有几个唤醒源。表6.2-4列举了各个模式下可用的时钟.

电源模式	正常模式	空闲模式	掉电模式

定义	CPU 在工作状态	CPU 在睡眠状态	CPU 在睡眠状态且除LXT 和 LIRC 外所有时钟停止。SRAM 数据保持。
进入条件	系统复位后芯片工作在正常模式	CPU 执行 WFI 指令.	使能CPU休眠和掉电功能后，执行指令 WFI。
唤醒源	N/A	All 中断	RTC, WDT, I²C, Timer, UART, BOD, GPIO, EINT, USCI, USBD,ACMP和BOD.
可用时钟	All	除 CPU时钟外的所有时钟	LXT 和 LIRC
唤醒后	N/A	CPU 返回正常模式	CPU返回正常模式

表 6.2-4 电源模式定义表

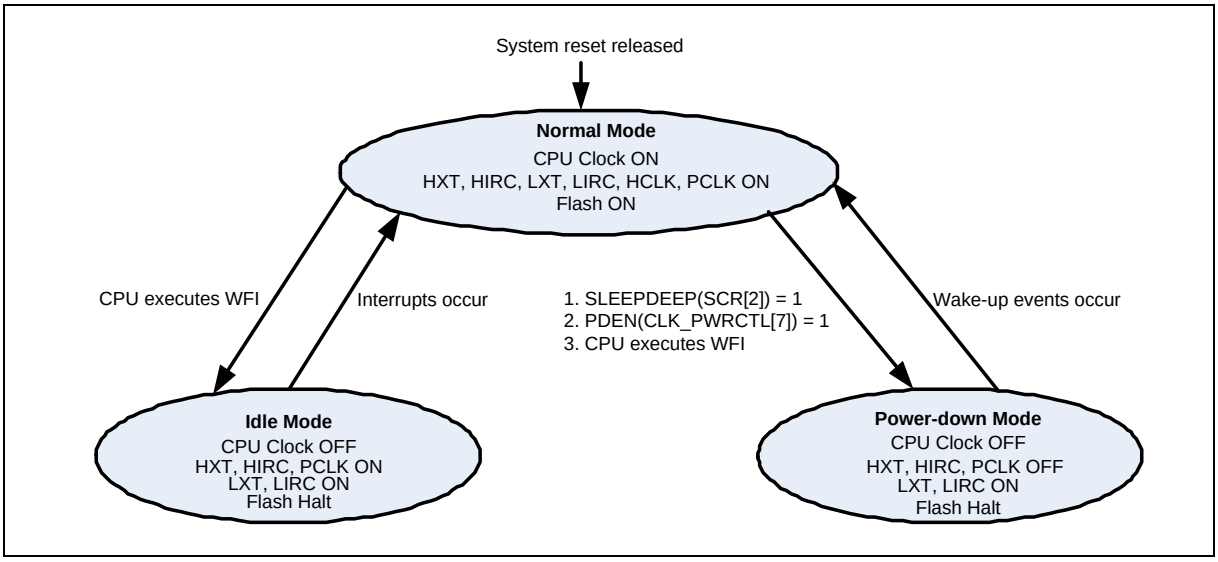


图 6.2-7 电源模式状态机

	空闲模式	NPD, LLPD, FWPD	SPD	DPD
HXT	ON	停止	停止	停止
HIRC	ON	停止	停止	停止
LXT	ON	ON/OFF[1]	ON/OFF[1]	ON/OFF[1]
LIRC	ON	ON/OFF[2]	ON/OFF[2]	ON/OFF[2,8]
PLL	ON	停止	停止	停止
HCLK/PCLK	ON	停止	停止	停止
CPU	停止	停止	停止	停止
SRAM 保持	ON	ON	ON/OFF[7]	停止
FLASH	ON	停止	停止	停止
TIMER	ON	ON/OFF[3]	ON/OFF[3]	停止

WDT	ON	ON/OFF[4]	ON/OFF[4]	停止
RTC	ON	ON/OFF[5]	ON/OFF[5]	ON/OFF[5]
UART	ON	ON/OFF[6]	ON/OFF[6]	停止
其他	ON	停止	停止	停止

表 6.2-5 不同电源模式的时钟

注意:

1. LXT ON 或 OFF 取决于在正常模式时软件的设置
2. LIRC ON 或 OFF 取决于在正常模式时软件的设置.
3. 是否 TIMER 时钟源选择为LIRC/LXT 且 LIRC/LXT 使能.
4. 是否WDT时钟源选择为LIRC且LIRC使能
5. 是否RTC时钟源选择为LXT且LXT使能.
6. 是否UART时钟源选择为LXT且LXT使能.
7. SRAM保留大小取决于正常模式下的SW设置.
8. 当芯片进入DPD模式进行省电时，如果关闭了定时唤醒功能，则会自动关闭LIRC.

正常掉电模式(NPD)下的唤醒源:

RTC, WDT, I²C, Timer, UART, USCI, BOD, EBOD, GPIO, USBD, 和ACMP都可以让掉电状态的 CPU 唤醒到正常工作模式

表6.2-6列举了如何再次进入掉电状态

在设置 PDEN(CLK_PWRCTL[7]) 和执行 WFI 进入掉电之前，需等这些条件满足

唤醒源	唤醒条件	省电模式			再次进入省电模式的条件
		NPD/ FWPD/ LLPD	SPD	DPD	
BOD	BOD复位 / 中断	V	-	-	软件对 BODIF (SYS_BODCTL[4])写1清0后.
	BOD复位	-	V	-	软件写入1 (CLK_PMUSTS[31])后，进入SPD模式时清除BODWK (CLK_PMUSTS[13])
LVR	LVR复位	V	-	-	软件对LVRF (SYS_RSTSTS[3])写1清0后.
		-	V	-	软件写入1 (CLK_PMUSTS[31])后，在进入SPD模式时清除LVRWK (CLK_PMUSTS[12])。
POR	POR复位	V	V	-	软件对PORF (SYS_RSTSTS[0])写1清0后.
INT	外中断	V	-	-	软件写1清零Px_INTSRC[n]之后
GPIO	GPIO中断	V	-	-	软件写1清零Px_INTSRC[n]之后

GPIO(PA~PD) Wake-up pin	上或下跳沿, 64个引脚	-	V	-	软件写入1 (CLK_PMUSTS[31])后，当进入SPD模式时清除GPxWK (CLK_PMUSTS[11:8])。
GPIO(PC.0/PB.0/PB.2/PB.12/PF.6) Wake-up pin	上或下跳沿, 5个引脚	-	-	V	当进入DPD模式时，软件写入1 (CLK_PMUSTS[31])清除PINWKx (CLK_PMUSTS[6:3]和CLK_PMUSTS[0])。
TIMER	定时器中断	V	-	-	软件写1清零TWKF (TIMERx_INTSTS[1]) 和TIF (TIMERx_INTSTS[0])后
Wakeup timer	唤醒定时器溢出后	-	V	V	当进入SPD或DPD模式，软件对TMRWK (CLK_PMUSTS[1])写1清0后.
WDT	WDT 中断	V	-	-	软件写1清零WKF (WDT_CTL[5]) (写保护)后.
RTC	闹钟中断	V	-	-	软件写1清零ALMIF (RTC_INTSTS[0])后
	周期节拍中断	V	-	-	软件写1清零TICKIF (RTC_INTSTS[1])后
	闹钟唤醒	-	V	V	当进入SPD模式，软件对RTCWK(CLK_PMUSTS[2])写1清0后.
	周期节拍唤醒	-	V	V	当进入SPD模式，软件对RTCWK(CLK_PMUSTS[2])写1清0后
	Tamper事件唤醒	-	V	V	当进入SPD模式，软件对RTCWK(CLK_PMUSTS[2])写1清0后
UART	nCTS 唤醒	V	-	-	软件写1清零CTSWKF (UARTx_WKSTS[0])后.
	RX 数据唤醒	V	-	-	软件写1清零DATWKF (UARTx_WKSTS[1])后.
	接收 FIFO 数据个数到门限唤醒	V	-	-	软件写1清零RFRTWKF (UARTx_WKSTS[2])后.
	RS-485自动地址识别匹配唤醒	V	-	-	软件写1清零RS485WKF (UARTx_WKSTS[3])后.
	接收 FIFO 门限，超时唤醒	V	-	-	软件写1清零TOUTWKF (UARTx_WKSTS[4]).后
USCI UART	CTS 跳变	V	-	-	软件写1清零WKF (UUART_WKSTS[0])后.
	数据跳变	V	-	-	软件写1清零WKF (UUART_WKSTS[0])后.
USCI I^2C	数据跳变	V	-	-	软件写1清零WKF (UI2C_WKSTS[0])后.
	地址匹配	V	-	-	软件写1清零WKAKDONE (UI2C_PROTSTS[16]后, 然后写1 清0 WKF (UI2C_WKSTS[0]).
USCI SPI	SS 跳变	V	-	-	软件写1清零WKF (USPI_WKSTS[0])后.
I2C	地址匹配唤醒	V	-	-	软件写1清零WKAKDONE (I2C_WKSTS[1])后，然后写1清0 WKIF(I2C_WKSTS[0]).
USBD	远程唤醒	V	-	-	软件写1清零BUSIF (USBD_INTSTS[0])后.
ACMP	比较器掉电唤醒中断	V	-	-	软件写1清零WKIF0 (ACMP_STATUS[8]) 和WKIF1 (ACMP_STATUS[9])后.
ACMP	比较器输出改变	-	V	-	当进入SPD模式，软件写1清零 ACMPWK(CLK_PMUSTS[14])后

表 6.2-6 重新进入掉电状态的条件

6.2.5 电源模式转换

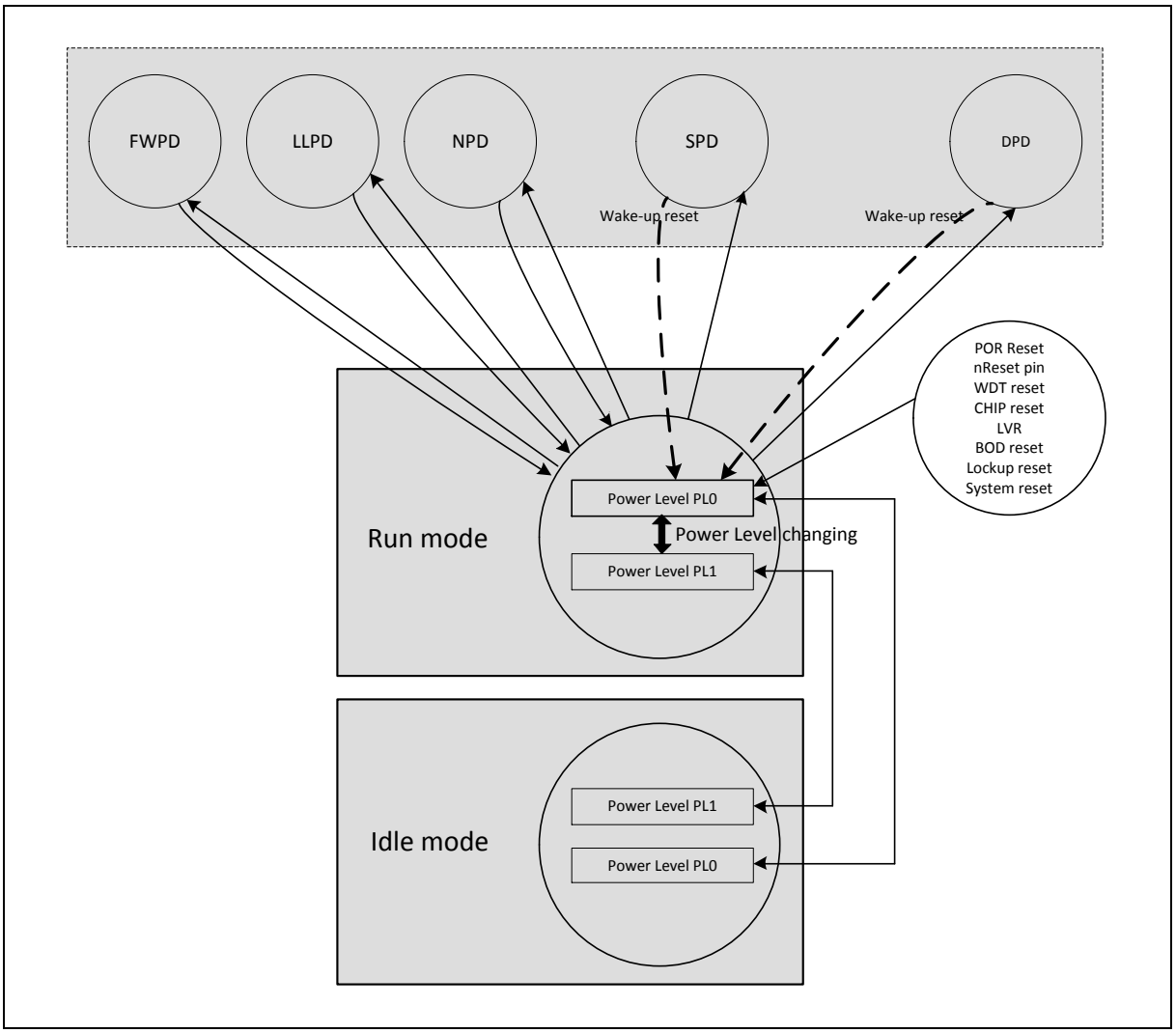


图 6.2-8 NuMicro®M480 电源模式图

6.2.6 系统内存映射

M480可访问 4G字节空间，表6.2-7是空间定义，M480仅支持小端数据格式。

地址空间	符号	描述
Flash和 SRAM		
0x0000_0000 – 0x0003_FFFF	FLASH_BA	FLASH 区 (256KB)
0x0000_0000 – 0x0007_FFFF	FLASH_BA	FLASH区(512KB)
0x0800_0000 – 0x09FF_FFFF	SPIM_BA	SPIM区(32MB)
0x2000_0000 – 0x2000_7FFF	SRAM0_BA	SRAM区(32KB)

0x2000_8000 – 0x2001_FFFF	SRAM1_BA	SRAM区 (96KB)
0x2002_0000 – 0x2002_7FFF	SRAM2_BA	仅用于 CPU 和SPIM 高速缓存SRAM 区 (32KB)
0x6000_0000 – 0x6FFF_FFFF	EXTMEM_BA	外接内存区 (256MB)
外设区 (0x4000_0000 – 0x400F_FFFF)		
0x4000_0000 – 0x4000_01FF	SYS_BA	系统控制寄存器
0x4000_0200 – 0x4000_02FF	CLK_BA	时钟控制寄存器
0x4000_0300 – 0x4000_03FF	NMI_BA	非屏蔽中断控制寄存器
0x4000_4000 – 0x4000_4FFF	GPIO_BA	引脚输入输出控制寄存器
0x4000_7000 – 0x4000_7FFF	SPIM_BA	SPIM 控制寄存器
0x4000_8000 – 0x4000_8FFF	PDMA_BA	外设 DMA 控制寄存器
0x4000_9000 – 0x4000_9FFF	USBH_BA	USB主机端控制寄存器
0x4000_B000 – 0x4000_BFFF	EMAC_BA	以太网 MAC控制寄存器
0x4000_C000 – 0x4000_CFFF	FMC_BA	闪存控制寄存器
0x4000_D000 – 0x4000_DFFF	SDH0_BA	SD卡接口0控制寄存器
0x4000_E000 – 0x4000_EFFF	SDH1_BA	SD卡接口1 控制寄存器
0x4001_0000 – 0x4001_0FFF	EBI_BA	外部总线接口控制寄存器
0x4001_9000 – 0x4001_9FFF	HSUSBD_BA	高速USB设备端控制寄存器
0x4001_A000 – 0x4001_AFFF	HSUSBH _BA	高速USB主机端控制寄存器
0x4003_0000 – 0x4003_0FFF	CCAP_BA	CCAP 控制寄存器
0x4003_1000 – 0x4003_1FFF	CRC_BA	CRC 运算控制寄存器
0x4003_E000 – 0x4003_EFFF	SWDC_BA	烧录仿真接口SWD 控制寄存器
0x4003_F000 – 0x4003_FFFF	ETMC_BA	ETM 控制寄存器
0x5008_0000 – 0x5008_0FFF	CRYP_BA	加解码算法控制寄存器
APB 控制区 (0x4000_0000 ~ 0x400F_FFFF)		
0x4004_0000 – 0x4004_0FFF	WDT_BA	看门狗控制寄存器
0x4004_1000 – 0x4004_1FFF	RTC_BA	实时时钟 (RTC) 控制寄存器
0x4004_3000 – 0x4004_3FFF	EADC0_BA	增强的ADC0 (EADC0) 控制寄存器
0x4004_5000 – 0x4004_5FFF	ACMP01_BA	模拟比较器 0/1控制寄存器
0x4004_6000 – 0x4004_6FFF	OPA_BA	运放控制寄存器
0x4004_7000 – 0x4004_7FFF	DAC_BA	DAC控制寄存器
0x4004_8000 – 0x4004_8FFF	I2S0_BA	I^2S0控制寄存器
0x4004_B000 – 0x4004_BFFF	EADC1_BA	增强的ADC1 (EADC1) 控制寄存器

0x4004_D000 – 0x4004_DFFF	OTG_BA	OTG控制寄存器
0x4004_F000 – 0x4004_FFFF	HSOTG_BA	高速OTG控制寄存器
0x4005_0000 – 0x4005_0FFF	TMR01_BA	Timer0/Timer1控制寄存器
0x4005_1000 – 0x4005_1FFF	TMR23_BA	Timer2/Timer3控制寄存器
0x4005_8000 – 0x4005_8FFF	EPWM0_BA	EPWM0控制寄存器
0x4005_9000 – 0x4005_9FFF	EPWM1_BA	EPWM1控制寄存器
0x4005_A000 – 0x4005_AFFF	BPWM0_BA	BPWM0控制寄存器
0x4005_B000 – 0x4005_BFFF	BPWM1_BA	BPWM1控制寄存器
0x4006_0000 – 0x4006_0FFF	QSPI0_BA	QSPI0控制寄存器
0x4006_1000 – 0x4006_1FFF	SPI0_BA	SPI0控制寄存器
0x4006_2000 – 0x4006_2FFF	SPI1_BA	SPI1控制寄存器
0x4006_3000 – 0x4006_3FFF	SPI2_BA	SPI2控制寄存器
0x4006_4000 – 0x4006_4FFF	SPI3_BA	SPI3控制寄存器
0x4006_9000 – 0x4006_9FFF	QSPI1_BA	QSPI1 控制寄存器
0x4007_0000 – 0x4007_0FFF	UART0_BA	UART0控制寄存器
0x4007_1000 – 0x4007_1FFF	UART1_BA	UART1控制寄存器
0x4007_2000 – 0x4007_2FFF	UART2_BA	UART2控制寄存器
0x4007_3000 – 0x4007_3FFF	UART3_BA	UART3控制寄存器
0x4007_4000 – 0x4007_4FFF	UART4_BA	UART4控制寄存器
0x4007_5000 – 0x4007_5FFF	UART5_BA	UART5控制寄存器
0x4007_6000 – 0x4007_6FFF	UART6_BA	UART6 控制寄存器
0x4007_7000 – 0x4007_7FFF	UART7_BA	UART7 控制寄存器
0x4008_0000 – 0x4008_0FFF	I^2C0_BA	I^2C0控制寄存器
0x4008_1000 – 0x4008_1FFF	I^2C1_BA	I^2C1控制寄存器
0x4008_2000 – 0x4008_2FFF	I^2C2_BA	I^2C2控制寄存器
0x4009_0000 – 0x4009_0FFF	SC0_BA	智能卡接口0 控制寄存器
0x4009_1000 – 0x4009_1FFF	SC1_BA	智能卡接口1控制寄存器
0x4009_2000 – 0x4009_2FFF	SC2_BA	智能卡接口2控制寄存器
0x4009_3000 – 0x4009_3FFF	SC3_BA	智能卡接口3控制寄存器
0x400A_0000 – 0x400A_0FFF	CAN0_BA	CAN0控制寄存器
0x400A_1000 – 0x400A_1FFF	CAN1_BA	CAN1 控制寄存器
0x400A_2000 – 0x400A_2FFF	CAN2_BA	CAN2控制寄存器

0x400B_0000 – 0x400B_0FFF	QEI0_BA	QEI0 控制寄存器
0x400B_1000 – 0x400B_1FFF	QEI1_BA	QEI1 控制寄存器
0x400B_4000 – 0x400B_4FFF	ECAP0_BA	ECAP0控制寄存器
0x400B_5000 – 0x400B_5FFF	ECAP1_BA	ECAP1 控制寄存器
0x400B_9000 – 0x400B_9FFF	TRNG_BA	TRNG 控制寄存器
0x400C_0000 – 0x400C_0FFF	USBD_BA	USB设备端控制寄存器
0x400D_0000 – 0x400D_0FFF	USCI0_BA	USCI0 控制寄存器
0x400D_1000 – 0x400D_1FFF	USCI1_BA	USCI1控制寄存器
系统控制区 (0xE000_E000 ~ 0xE000_EFFF)		
0xE000_E010 – 0xE000_E0FF	SCS_BA	系统定时器控制寄存器
0xE000_E100 – 0xE000_ECFF	SCS_BA	外中断控制寄存器
0xE000_ED00 – 0xE000_ED8F	SCS_BA	系统控制控制寄存器

表 6.2-7 片上地址空间分布

6.2.7 SRAM 内存结构

160K字节的SRAM分成三个区块: bank032K字节，bank1 96K字节, bank2 32K字节。三个区块地址连续， bank0有奇偶校验， bank2与SPIM高速缓存共享,可切换到外SPI Flash 缓存空间。注：当bank2读数据时额外增加两个等待周期。

- 160K字节
- 支持字节 / 半字 / 字的写入
- bank0可独立访问
- bank0支持奇偶校验
- 支持地址溢出出错响应
- 首地址可重映射到 0x1000_0000

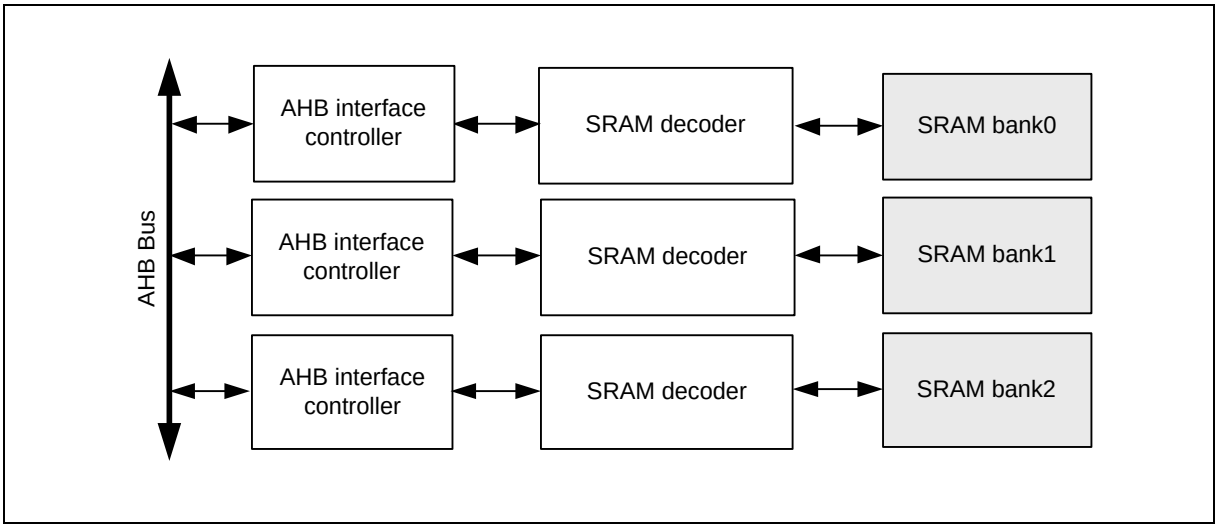


图 6.2-9 SRAM 框图

图6.2-9是SRAM框图，bank0地址范围0x2000_0000 ~ 0x2000_7FFF。bank1地址范围0x2000_8000 ~ 0x2001_FFFF。bank2地址范围0x2002_0000 ~ 0x2002_7FFF。CPU若读写地址范围0x2002_8000 ~ 0x3FFF_FFFF这些非法空间，会产生hardfault。

	160KB设备	128KB设备	96KB设备	读取访问
SRAM bank0	0x2000_0000 ~ 0x2000_7FFFor 0x1000_0000 ~0x1000_7FFF	0x2000_0000 ~ 0x2000_7FFFor 0x1000_0000 ~0x1000_7FFF	0x2000_0000 ~ 0x2000_7FFFor 0x1000_0000 ~0x1000_7FFF	持续访问零等待周期
SRAM bank1	0x2000_8000 ~ 0x2001_FFFFor 0x1000_8000 ~0x1001_FFFF	0x2000_8000 ~ 0x2001_FFFFor 0x1000_8000 ~0x1001_FFFF	0x2000_8000 ~ 0x2000_FFFFor 0x1000_8000 ~0x1000_FFFF	持续访问零等待周期
SRAM bank2	0x2002_0000 ~ 0x2002_7FFFor 0x1002_0000 ~0x1002_7FFF		0x2001_0000 ~ 0x2001_7FFFor 0x1001_0000 ~0x1001_7FFF	两个等待周期用于连续访问

表6.2-8SRAM结构

每个空间都可以从地址 0x2000_0000重映射到0x1000_0000。CPU可以从地址0x1000_0000 ~ 0x1000_7FFF读写 bank0，也可以从地址0x1000_8000~0x1001_FFFF读写bank1，也可以从地址0x1002_0000 ~ 0x1002_7FFF读写bank2。

CCMEN(SPIM_CTL1[2])=0时, SRAM bank2切换成外部 SPI Flash 高速缓存，此时SRAM bank2不能再当做通用SRAM，读写这个区域会返回HRESP AHB错误。

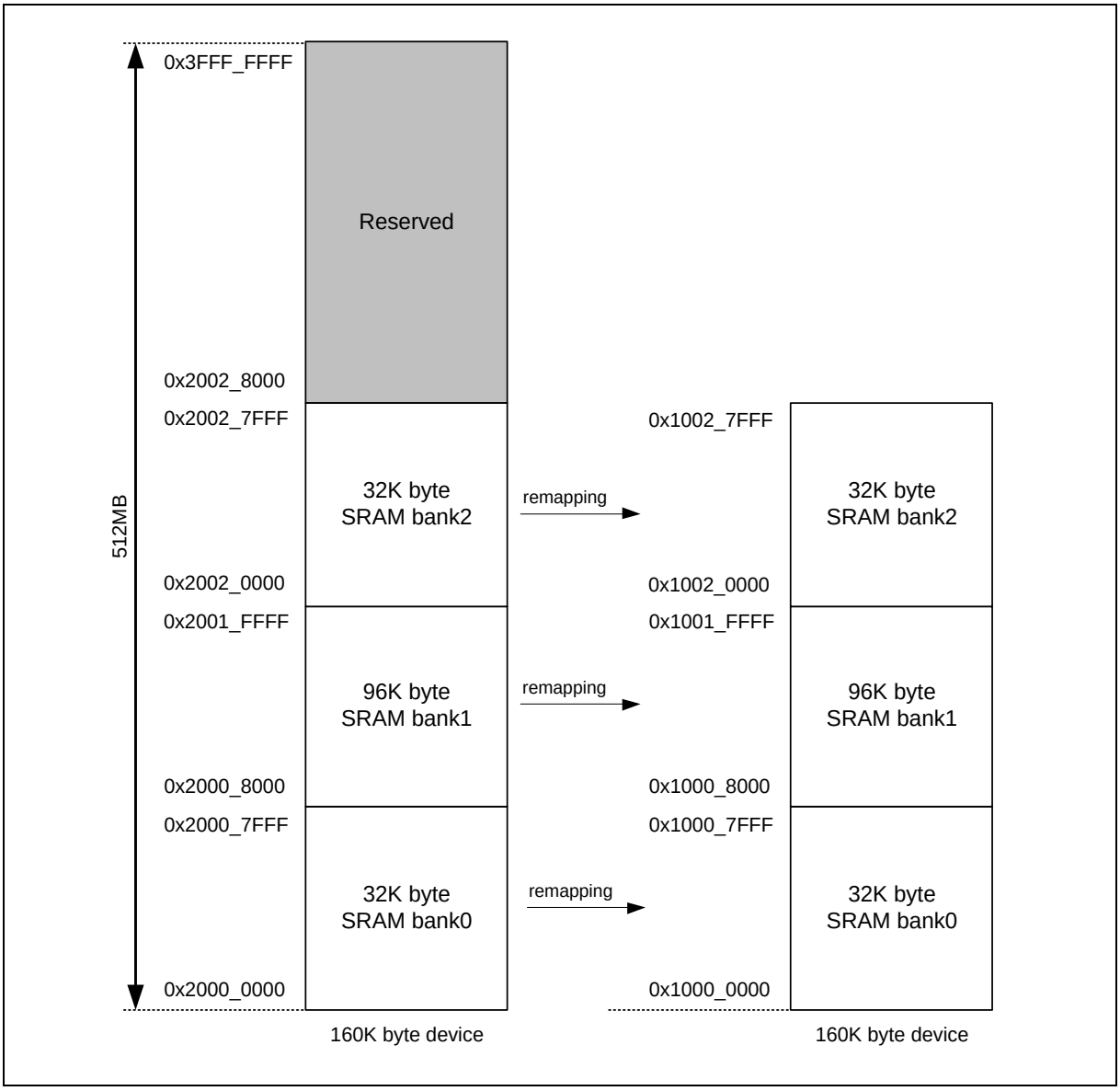


图 6.2-10SRAM 内存结构

SRAM bank0 有奇偶校验功能，读出时的奇偶校验值若与写入时不一样，PERRIF (SYS_SRAM_STATUS[0]) 将置1，SYS_SRAM_ERRADDR记录产生奇偶错的地址。若PERRIEN (SYS_SRAM_INTCTL[0])=1将产生中断。在PERRIF(SYS_SRAM_STATUS[0]) 写1清0之前，芯片将不再检测奇偶校验错。

6.2.8 总线矩阵

M480支持总线仲裁。总线仲裁法则由INTACTEN (SYS_AHBMCTL[0])决定，CPU有最高总线优先级。

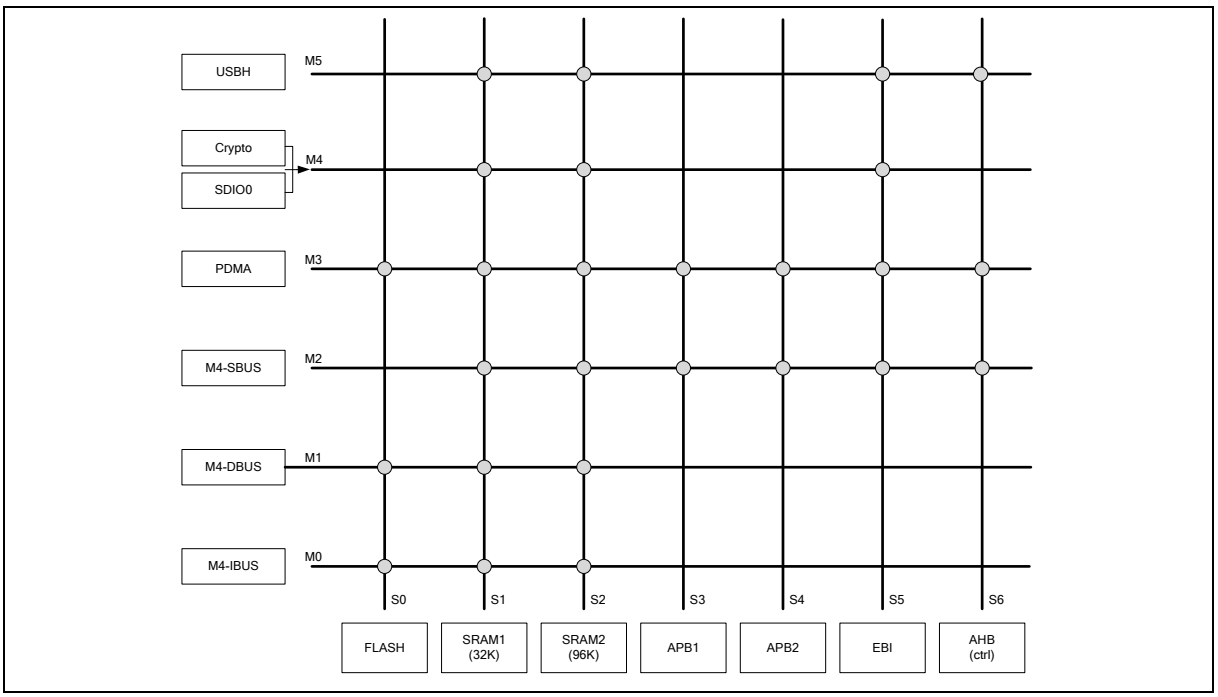


图 6.2-11 NuMicro®M480 总线矩阵框图

6.2.9 HIRC 自动校准

芯片支持自动校准功能：HIRC 校准（12MHz RC振荡器和48 MHz RC振荡器），根据外部低速时钟（32.768kHz晶振）或者内部USB同步模式，自动的获取精确的HIRC输出频率，全温度范围内精度在0.25%。

例如，系统需要一个精确的12MHz时钟。在这种情况下，如果既没有使用PLL作为系统时钟源，也没有在系统中焊接32.768 kHz晶体，用户必须将REFCKSEL (SYS_IRCTCTL[10]参考时钟选择）设置为“1”，将FREQSEL (SYS_IRCTCTL[1:0]频率选择）设为“01”，自动校准功能将被启用。中断状态位FREQLOCK (SYS_IRCTISTS[8] HIRC频率锁定状态）“1”表示HIRC输出频率在0.25％偏差内精确。

在另一种情况下，系统需要一个精确的48MHz时钟。在这种情况下，如果既没有使用PLL作为系统时钟源，也没有在系统中焊接32.768 kHz晶体，用户必须将REFCKSEL (SYS_HIRCTCTL[10]参考时钟选择）设置为“1”，将FREQSEL (SYS_HIRCTCTL[1:0]频率选择）设为“10”，自动微调功能将被启用。中断状态位FREQLOCK (SYS_HIRCTSTS[8] HIRC频率锁定状态）“1”表示HIRC输出频率在0.25％偏差内精确。

6.2.10 寄存器锁定控制

部分系统控制寄存器有做保护机制，以避免误写和妨碍芯片的运作。在上电复位后直到用户解除了这些寄存器的保护之前，这些系统控制寄存器都是受保护的。如果用户要操作这些受保护的寄存器，需要向一个特定的寄存器SYS_REGLCTL地址为0x4000_0100，连续的写入三个数据“59h”, “16h” “88h”，才能解除这些系统寄存器的保护，在写入这个三个数据的过程中，不能有任何的干扰，如写入别的值，顺序错误，写入别的地址，这些都将终止这个解除系统寄存器保护的动作。

SYS_IPRST0	**地址 0x4000_0008**
SYS_BODCTL	地址 0x4000_0018
SYS_VREFCTL	地址0x4000_0028
SYS_USBPHY	地址0x4000_002C
SYS_SRAM_BISTCTL	地址0x4000_00D0
SYS_PORDISAN	地址0x4000_01EC
SYS_PLCTL	地址0x4000_01F8
CLK_PWRCTL	地址0x4000_0200
CLK_APBCLK0	地址0x4000_0208
CLK_CLKSEL0	地址0x4000_0210
CLK_CLKSEL1	地址0x4000_0214
CLK_PLLCTL	地址0x4000_0240
CLK_PMUCTL	地址0x4000_0290
NMIEN	地址0x4000_0300
AHBMCTL	地址0x4000_0400
FMC_FTCTL	地址0x4000_5018
FMC_ICPCMD	地址0x4000_501C
FMC_ISPCTL	地址0x4000_C000
FMC_ISPTRG	地址0x4000_C010
FMC_ISPSTS	地址0x4000_C040
FMC_CYCCTL	地址0x4000_C04C
FMC_KPKEYTRG	地址0x4000_C05C
FMC_KPKEYSTS	地址0x4000_C060
WDT_CTL	地址0x4004_0000
WDT_ALTCTL	地址0x4004_0004
TIMER0_CTL	地址0x4005_0000
TIMER1_CTL	地址0x4005_0100

TIMER2_CTL	地址0x4005_1000
TIMER3_CTL	地址0x4005_1100
TIMER0_PWMCTL	地址0x4005_0040
TIMER1_PWMCTL	地址0x4005_0140
TIMER2_PWMCTL	地址0x4005_1040
TIMER3_PWMCTL	地址0x4005_1140
TIMER0_PWMDTCTL	地址0x4005_0058
TIMER1_PWMDTCTL	地址0x4005_0158
TIMER2_PWMDTCTL	地址0x4005_1058
TIMER3_PWMDTCTL	地址0x4005_1158
TIMER0_PWMBRKCTL	地址0x4005_0070
TIMER1_PWMBRKCTL	地址0x4005_0170
TIMER2_PWMBRKCTL	地址0x4005_1070
TIMER3_PWMBRKCTL	地址0x4005_1170
TIMER0_PWMSWBRK	地址0x4005_007C
TIMER1_PWMSWBRK	地址0x4005_017C
TIMER2_PWMSWBRK	地址0x4005_107C
TIMER3_PWMSWBRK	地址0x4005_117C
TIMER0_PWMINTEN1	地址0x4005_0084
TIMER1_PWMINTEN1	地址0x4005_0184
TIMER2_PWMINTEN1	地址0x4005_1084
TIMER3_PWMINTEN1	地址0x4005_1184
TIMER0_PWMINTSTS1	地址0x4005_008C
TIMER1_PWMINTSTS1	地址0x4005_018C
TIMER2_PWMINTSTS1	地址0x4005_108C
TIMER3_PWMINTSTS1	地址0x4005_118C
EPWM_CTL0	地址0x4005_8000/0x4005_9000
EPWM_CTL1	地址0x4005_8000/0x4005_9000
EPWM_DTCTL0_1	地址0x4005_8070/0x4005_9070
EPWM_DTCTL2_3	地址0x4005_8074/0x4005_9074
EPWM_DTCTL4_5	地址0x4005_8078/0x4005_9078
EPWM_BRKCTL0_1	地址0x4005_80C8/0x4005_90C8

EPWM_BRKCTL2_3	地址0x4005_80CC/0x4005_90CC
EPWM_BRKCTL4_5	地址0x4005_80D0/0x4005_90D0
EPWM_SWBRK	地址0x4005_80DC/0x4005_90DC
EPWM_INTEN1	地址0x4005_80E4/0x4005_90E4
EPWM_INTSTS1	地址0x4005_80EC/0x4005_90EC
BPWM_CTL0	地址0x4005_A000/0x4005_B000
SYST_VAL	地址0xE000_E018

6.2.11 寄存器映射

R: 只读, **W**: 只写, **R/W**: 读写

寄存器	偏移	R/W	描述	复位值
SYS基地址: **SYS_BA = 0x4000_0000**				
SYS_PDID	SYS_BA+0x00	R	芯片型号	0x0045_1A93[1]
SYS_RSTSTS	SYS_BA+0x04	R/W	复位源寄存器	0x0000_0043
SYS_IPRST0	SYS_BA+0x08	R/W	外设复位控制寄存器0	0x0000_0000
SYS_IPRST1	SYS_BA+0x0C	R/W	外设复位控制寄存器1	0x0000_0000
SYS_IPRST2	SYS_BA+0x10	R/W	外设复位控制寄存器2	0x0000_0000
SYS_BODCTL	SYS_BA+0x18	R/W	BOD 欠压控制寄存器	0x000X_038X
SYS_IVSCTL	SYS_BA+0x1C	R/W	片内电压控制寄存器	0x0000_0000
SYS_VREFCTL	SYS_BA+0x28	R/W	V_{REF}控制寄存器	0x0000_0200
SYS_USBPHY	SYS_BA+0x2C	R/W	USB PHY控制寄存器	0x0003_0007
SYS_GPA_MFPL	SYS_BA+0x30	R/W	GPIOA 低字节多功能控制寄存器	0x0000_0000
SYS_GPA_MFPH	SYS_BA+0x34	R/W	GPIOA 高字节多功能控制寄存器	0x0000_0000
SYS_GPB_MFPL	SYS_BA+0x38	R/W	GPIOB低字节多功能控制寄存器	0x0000_0000
SYS_GPB_MFPH	SYS_BA+0x3C	R/W	GPIOB高字节多功能控制寄存器	0x0000_0000
SYS_GPC_MFPL	SYS_BA+0x40	R/W	GPIOC低字节多功能控制寄存器	0x0000_0000
SYS_GPC_MFPH	SYS_BA+0x44	R/W	GPIOC高字节多功能控制寄存器	0x0000_0000
SYS_GPD_MFPL	SYS_BA+0x48	R/W	GPIOD低字节多功能控制寄存器	0x0000_0000
SYS_GPD_MFPH	SYS_BA+0x4C	R/W	GPIOD高字节多功能控制寄存器	0x0000_0000
SYS_GPE_MFPL	SYS_BA+0x50	R/W	GPIOE低字节多功能控制寄存器	0x0000_0000
SYS_GPE_MFPH	SYS_BA+0x54	R/W	GPIOE高字节多功能控制寄存器	0x0000_0000
SYS_GPF_MFPL	SYS_BA+0x58	R/W	GPIOF低字节多功能控制寄存器	0x0000_00EE
SYS_GPF_MFPH	SYS_BA+0x5C	R/W	GPIOF高字节多功能控制寄存器	0x0000_0000
SYS_GPG_MFPL	SYS_BA+0x60	R/W	GPIOG低字节多功能控制寄存器	0x0000_0000
SYS_GPG_MFPH	SYS_BA+0x64	R/W	GPIOG高字节多功能控制寄存器	0x0000_0000
SYS_GPH_MFPL	SYS_BA+0x68	R/W	GPIOH低字节多功能控制寄存器	0x0000_0000
SYS_GPH_MFPH	SYS_BA+0x6C	R/W	GPIOH高字节多功能控制寄存器	0x0000_0000

SYS_GPA_MFOS	SYS_BA+0x80	R/W	GPIOA 多功能输出选择寄存器	0x0000_0000
SYS_GPB_MFOS	SYS_BA+0x84	R/W	GPIOB多功能输出选择寄存器	0x0000_0000
SYS_GPC_MFOS	SYS_BA+0x88	R/W	GPIOC多功能输出选择寄存器	0x0000_0000
SYS_GPD_MFOS	SYS_BA+0x8C	R/W	GPIOD多功能输出选择寄存器	0x0000_0000
SYS_GPE_MFOS	SYS_BA+0x90	R/W	GPIOE多功能输出选择寄存器	0x0000_0000
SYS_GPF_MFOS	SYS_BA+0x94	R/W	GPIOF多功能输出选择寄存器	0x0000_0000
SYS_GPG_MFOS	SYS_BA+0x98	R/W	GPIOG多功能输出选择寄存器	0x0000_0000
SYS_GPH_MFOS	SYS_BA+0x9C	R/W	GPIOH多功能输出选择寄存器	0x0000_0000
SYS_SRAM_INTCTL	SYS_BA+0xC0	R/W	系统SRAM中断使能控制寄存器	0x0000_0000
SYS_SRAM_STATUS	SYS_BA+0xC4	R/W	系统SRAM奇偶错误状态寄存器	0x0000_0000
SYS_SRAM_ERRADDR	SYS_BA+0xC8	R	系统SRAM奇偶检查错误地址寄存器	0x0000_0000
SYS_SRAM_BISTCTL	SYS_BA+0xD0	R/W	系统SRAM BIST测试控制寄存器	0x0000_0000
SYS_SRAM_BISTSTS	SYS_BA+0xD4	R	系统 SRAM BIST测试状态寄存器	0x00xx_00xx
SYS_HIRCTCTL	SYS_BA+0xE4	R/W	HIRC48M校准控制寄存器	0x0008_0000
SYS_HIRCTIEN	SYS_BA+0xE8	R/W	HIRC48M校准中断使能寄存器	0x0000_0000
SYS_HIRCTISTS	SYS_BA+0xEC	R/W	HIRC48M校准中断状态寄存器	0x0000_0000
SYS_IRCTCTL	SYS_BA+0xF0	R/W	HIRC 校准控制寄存器	0x0008_0000
SYS_IRCTIEN	SYS_BA+0xF4	R/W	HIRC 校准中断使能寄存器	0x0000_0000
SYS_IRCTISTS	SYS_BA+0xF8	R/W	HIRC 校准中断状态寄存器	0x0000_0000
SYS_REGLCTL	SYS_BA+0x100	R/W	寄存器锁控制寄存器	0x0000_0000
SYS_PORDISAN	SYS_BA+0x1EC	R/W	模拟POR 禁用控制寄存器	0x0000_0000
SYS_CSERVER	SYS_BA+0x1F4	R	芯片系列版本寄存器	0x0000_00XX
SYS_PLCTL	SYS_BA+0x1F8	R/W	内核电压控制寄存器	0x0000_0000
SYS_PLSTS	SYS_BA+0x1FC	R	内核电压状态寄存器	0x0000_010X
SYS_AHBMCTL	SYS_BA+0x400	R/W	AHB 总线矩阵优先级控制寄存器	0x0000_0001

6.2.12 寄存器描述

芯片型号寄存器 (SYS_PDID)

寄存器	偏移	R/W	描述	复位值
SYS_PDID	SYS_BA+0x00	R	芯片型号寄存器	0x0045_1A93[1]

[1] 默认的每个型号都是唯一的。

31	30	29	28	27	26	25	24
PDID							
23	22	21	20	19	18	17	16
PDID							
15	14	13	12	11	10	9	8
PDID							
7	6	5	4	3	2	1	0
PDID							

位	描述	
[31:0]	PDID	**芯片型号(只读)** 该寄存器为设备的器件号码。软件可以读取该寄存器来识别所使用的器件。

系统复位状态寄存器(SYS_RSTSTS)

该寄存器为软件提供特定的信息来识别该芯片上次操作中的复位源

寄存器	偏移	R/W	描述	复位值
SYS_RSTSTS	SYS_BA+0x04	R/W	系统复位状态寄存器	0x0000_0043

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							CPULKRF
7	6	5	4	3	2	1	0
CPURF	HRESETRF	SYSRF	BODRF	LVRF	WDTRF	PINRF	PORF

位	描述	
[31:9]	Reserved	保留.
[8]	CPULKRF	**CPU锁死复位** 0 = 未发生CPU锁死复位。 1 = 发生Cortex®-M4锁死复位。 **注1**：写1清0。 **注2**：当CPU锁死发生在已连接ICE时，该标志位将置1但芯片不复位。
[7]	CPURF	**CPU复位** 如果软件对CPURST（SYS_IPRST0[1]）写1来复位Cortex®-M4内核和FMC，CPU复位标志将被硬件置位。 0 = 未产生下述复位。 1 = 复位原因是对CPURST写1，使Cortex®-M4内核和FMC复位。 **注**：写1清0。
[6]	HRESETRF	**HRESET复位** 通过HRESET的“复位信号”HRESET复位标志置1。 0 = 没有HRESET复位。 1 = HRESET复位。 **注:**写1清0。

位	描述	
[5]	**SYSRF**	**系统复位** 系统复位标志被来自Cortex®-M4内核的“复位信号”置位，来指示前一个复位。 0= 没有来自Cortex®-M4的复位。 1= Cortex®-M4发出复位信号，通过将1写入该位来复位系统。SYSRESETREQ（AIRCR[2]，应用中断和复位控制寄存器，地址=0xE000ED0C）在Cortex®-M4内核的系统控制寄存器中。 **注**：写1清0。
[4]	**BODRF**	**BOD复位** BOD复位标志被来自欠压侦测的“复位信号”置位，来指示前一个复位源。 0 = 没有来自BOD的复位。 1 = BOD已经产生复位信号来复位系统。 **注**：写1清0。
[3]	**LVRF**	**LVR复位** LVR复位标志被来自低电压复位控制器的“复位信号”置位，来指示前一个复位源。 0 = 没有来自LVR的复位。 1 = LVR控制器已经产生复位信号来复位系统。 **注**：写1清0。
[2]	**WDTRF**	**WDT复位** WDT复位标志被看门狗定时器或窗口看门狗定时器的“复位信号”置位，来指示前一个复位源。 0 = 没有来自看门狗定时器或者窗口看门狗定时器的复位。 1 = 看门狗定时器或者窗口看门狗定时器已经产生复位信号来复位系统。 **注1**：写1清0。 **注2**：系统被WDT超时复位复位后看门狗定时器寄存器RSTF（WDT_CTL[2]）置位。系统被WWDT超时复位复位后看门狗定时器寄存器WWDTRF（WWDT_STATUS[1]）置位。
[1]	**PINRF**	**NRESET 管脚复位** nRESET脚复位标志被nRESET脚的“复位信号”置位，来指示前一个复位源。 0 = 没有来自nRESET脚的复位。 1 = nRESET脚已经产生复位信号来复位系统。 **注**：写1清0。

位	描述	
[0]	**PORF**	**POR管脚复位** POR复位标志被上电复位（POR）控制器或CHIPRST（SYS_IPRST0[0]）位的“复位信号”置位，来指示前一个复位源。 0 =没有来自POR或CHIPRST的复位。 1 = 上电复位（POR）或CHIPRST已经产生复位信号来复位系统。 **注：**写1清0。

外设复位控制寄存器0 (SYS_IPRST0)

寄存器	偏移	R/W	描述	复位值
SYS_IPRST0	SYS_BA+0x08	R/W	外设复位控制寄存器0	0x0000_0000

31	30	29	28	27	26	25	24
ETMCRST	SWDCRST	Reserved					
23	22	21	20	19	18	17	16
Reserved						SDH1RST	HSUSBHRST
15	14	13	12	11	10	9	8
Reserved	SPIMRST	Reserved	CRPTRST	Reserved	HSUSBDRST	Reserved	CCAPRST
7	6	5	4	3	2	1	0
CRCRST	SDH0RST	EMACRST	Reserved	EBIRST	PDMARST	CPURST	CHIPRST

位	描述	
[31]	**ETMCRST**	**ETM 控制器复位** 该位置1将产生一个复位信号给ETM控制器。用户需要将该位置0才能释放复位状态。 0 = ETM 控制器正常操作。 1 = ETM 控制器复位。
[30]	**SWDCRST**	**SWD控制器复位** 该位置1将产生一个复位信号给SWD控制器。用户需要将该位置0才能释放复位状态。 0 = SWD控制器正常操作。 1 = SWD控制器复位。
[29:18]	**Reserved**	保留。
[17]	**SDH1RST**	**SDHOST1控制器复位(写保护)** 该位置1将产生一个复位信号给SDHOST1控制器。用户需要将该位置0才能释放复位状态。 0 = SDHOST1控制器正常操作。 1 = SDHOST1控制器复位。 **注:** 该位为写保护，具体参考SYS_REGLCTL 寄存器。
[16]	**HSUSBHRST**	**HSUSBH控制器复位(写保护)** 该位置1将产生一个复位信号给HSUSBH控制器。用户需要将该位置0才能释放复位状态。 0 = HSUSBH控制器正常操作。 1 = HSUSBH控制器复位。 **注:** 该位为写保护，具体参考SYS_REGLCTL 寄存器。
[15]	**Reserved**	保留。
[14]	**SPIMRST**	**SPIM控制器复位** 该位置1将产生一个复位信号给SPIM控制器。用户需要将该位置0才能释放复位状态。

		0 = SPIM控制器正常操作。 1 = SPIM控制器复位。
[13]	**Reserved**	保留。
[12]	**CRPTRST**	**CRYPTO控制器复位(写保护)** 该位置1将产生一个复位信号给CRYPTO控制器。用户需要将该位置0才能释放复位状态。 0 = CRYPTO控制器正常操作。 1 = CRYPTO控制器复位。 **注:** 该位为写保护，具体参考SYS_REGLCTL 寄存器。
[11]	**Reserved**	保留。
[10]	**HSUSBDRST**	**HSUSBD控制器复位(写保护)** 该位置1将产生一个复位信号给HSUSBD控制器。用户需要将该位置0才能释放复位状态。 0 = HSUSBD控制器正常操作。 1 = HSUSBD控制器复位。 **注:** 该位为写保护，具体参考SYS_REGLCTL 寄存器。
[9]	**Reserved**	保留。
[8]	**CCAPRST**	**CCAP控制器复位(写保护)** 该位置1将产生一个复位信号给CCAP控制器。用户需要将该位置0才能释放复位状态。 0 = CCAP控制器正常操作。 1 = CCAP控制器复位。 **注:** 该位为写保护，具体参考SYS_REGLCTL 寄存器。
[7]	**CRCRST**	**CRC控制器复位(写保护)** 该位置1将产生一个复位信号给CRC控制器。用户需要将该位置0才能释放复位状态。 0 = CRC控制器正常操作。 1 = CRC控制器复位。 **注:** 该位为写保护，具体参考SYS_REGLCTL 寄存器。
[6]	**SDH0RST**	**SDHOST0控制器复位(写保护)** 该位置1将产生一个复位信号给SDHOST0控制器。用户需要将该位置0才能释放复位状态。 0 = SDHOST0控制器正常操作。 1 = SDHOST0控制器复位。 **注:** 该位为写保护，具体参考SYS_REGLCTL 寄存器。
[5]	**EMACRST**	**EMAC控制器复位(写保护)** 该位置1将产生一个复位信号给EMAC控制器。用户需要将该位置0才能释放复位状态。 0 = EMAC控制器正常操作。 1 = EMAC控制器复位。 **注:** 该位为写保护，具体参考SYS_REGLCTL 寄存器。
[4]	**Reserved**	保留。
[3]	**EBIRST**	**EBI控制器复位(写保护)** 该位置1将产生一个复位信号给EBI控制器。用户需要将该位置0才能释放复位状态。 0 = EBI控制器正常操作。 1 = EBI控制器复位。

		注: 该位为写保护，具体参考SYS_REGLCTL 寄存器。
[2]	**PDMARST**	**PDMA控制器复位(写保护)** 该位置1将产生一个复位信号给PDMA控制器。用户需要将该位置0才能释放复位状态。 0 = PDMA控制器正常操作。 1 = PDMA控制器复位。 **注:** 该位为写保护，具体参考SYS_REGLCTL 寄存器。
[1]	**CPURST**	**CPU复位(写保护)** 设置此位将只复位CPU和FMC，且该位会在2个时钟周期后自动归零。 0 = CPU正常运行。 1 = CPU复位一次。 **注:** 该位为写保护，具体参考SYS_REGLCTL 寄存器。
[0]	**CHIPRST**	**芯片复位（写保护）** 设置此位将复位整个芯片，包扣CPU和全部外设，该位会在2个时钟周期后自动归零。 CHIPRST与POR复位一样，所有芯片控制器将复位且芯片设定将会从Fash重新加载。 CHIPRST和SYSRESETREQ（AIRCR[2]）不同请参考第7.2.2节。 0 = 芯片正常工作。 1 = 芯片一次复位。 **注:** 该位是写保护的。请参考SYS_REGLCTL寄存器。

外设复位控制寄存器1 (SYS_IPRST1)

对这些位写1对应外设将产生异步复位。用户需要对这些位写0后对应外设才正常工作。

寄存器	偏移	R/W	描述	复位值
SYS_IPRST1	SYS_BA+0x0C	R/W	外设复位控制寄存器1	0x0000_0000

31	30	29	28	27	26	25	24
TRNGRST	HSOTGRST	I2S0RST	EADCRST	USBDRST	OTGRST	CAN1RST	CAN0RST
23	22	21	20	19	18	17	16
UART7RST	UART6RST	UART5RST	UART4RST	UART3RST	UART2RST	UART1RST	UART0RST
15	14	13	12	11	10	9	8
SPI2RST	SPI1RST	SPI0RST	QSPI0RST	Reserved	I^2C2RST	I^2C1RST	I^2C0RST
7	6	5	4	3	2	1	0
ACMP01RST	Reserved	TMR3RST	TMR2RST	TMR1RST	TMR0RST	GPIORST	Reserved

位	描述	
[31]	**TRNGRST**	**TRNG控制器复位** 0 = TRNG控制器正常运行。 1 = TRNG控制器复位。
[30]	**HSOTGRST**	**HSOTG控制器复位** 0 = HSOTG控制器正常运行。 1 = HSOTG控制器复位。
[29]	**I2S0RST**	**I2S0控制器复位** 0 = I2S0控制器正常运行。 1 = I2S0控制器复位。
[28]	**EADCRST**	**EADC控制器复位** 0 = EADC控制器正常运行。 1 = EADC控制器复位。
[27]	**USBDRST**	**USBD控制器复位** 0 = USBD控制器正常运行。 1 = USBD控制器复位。
[26]	**OTGRST**	**OTG控制器复位** 0 = OTG控制器正常运行。 1 = OTG控制器复位。
[25]	**CAN1RST**	**CAN1控制器复位** 0 = CAN1控制器正常运行。 1 = CAN1控制器复位。

[24]	**CAN0RST**	**CAN0控制器复位** 0 = CAN0控制器正常运行。 1 = CAN0控制器复位。
[23]	**UART7RST**	**UART7控制器复位** 0 = UART7控制器正常运行。 1 = UART7控制器复位。
[22]	**UART6RST**	**UART6控制器复位** 0 = UART6控制器正常运行。 1 = UART6控制器复位。
[21]	**UART5RST**	**UART5控制器复位** 0 = UART5控制器正常运行。 1 = UART5控制器复位。
[20]	**UART4RST**	**UART4控制器复位** 0 = UART4控制器正常运行。 1 = UART4控制器复位。
[19]	**UART3RST**	**UART3控制器复位** 0 = UART3控制器正常运行。 1 = UART3控制器复位。
[18]	**UART2RST**	**UART2控制器复位** 0 = UART2控制器正常运行。 1 = UART2控制器复位。
[17]	**UART1RST**	**UART1控制器复位** 0 = UART1控制器正常运行。 1 = UART1控制器复位。
[16]	**UART0RST**	**UART0控制器复位** 0 = UART0控制器正常运行。 1 = UART0控制器复位。
[15]	**SPI2RST**	**SPI2控制器复位** 0 = SPI2控制器正常运行。 1 = SPI2控制器复位。
[14]	**SPI1RST**	**SPI1控制器复位** 0 = SPI1控制器正常运行。 1 = SPI1控制器复位。
[13]	**SPI0RST**	**SPI0控制器复位** 0 = SPI0控制器正常运行。 1 = SPI0控制器复位。
[12]	**QSPI0RST**	**Qual SPI0控制器复位** 0 = Qual SPI0控制器正常运行。 1 = Qual SPI0控制器复位。

[11]	**Reserved**	保留。
[10]	**I²C2RST**	**I²C2控制器复位** 0 = I²C2控制器正常运行。 1 = I²C2控制器复位。
[9]	**I²C1RST**	**I²C1控制器复位** 0 = I²C1控制器正常运行。 1 = I²C1控制器复位。
[8]	**I²C0RST**	**I²C0控制器复位** 0 = I²C0控制器正常运行。 1 = I²C0控制器复位。
[7]	**ACMP01RST**	**模拟比较器0/1控制器复位** 0 = 模拟比较器0/1控制器正常运行。 1 =模拟比较器0/1控制器复位。
[6]	**Reserved**	保留。
[5]	**TMR3RST**	**Timer3控制器复位** 0 = Timer3控制器正常运行。 1 = Timer3控制器复位。
[4]	**TMR2RST**	**Timer2控制器复位** 0 = Timer2控制器正常运行。 1 = Timer2控制器复位。
[3]	**TMR1RST**	**Timer1控制器复位** 0 = Timer1控制器正常运行。 1 = Timer1控制器复位。
[2]	**TMR0RST**	**Timer0控制器复位** 0 = Timer0控制器正常运行。 1 = Timer0控制器复位。
[1]	**GPIORST**	**GPIO控制器复位** 0 = GPIO控制器正常运行。 1 = GPIO控制器复位。
[0]	**Reserved**	保留。

外设复位控制寄存器2 (SYS_IPRST2)

对这些位写1对应外设将产生异步复位。用户需要对这些位写0后对应外设才正常工作。

寄存器	偏移	R/W	描述	复位值
SYS_IPRST2	SYS_BA+0x10	R/W	外设复位控制寄存器2	0x0000_0000

31	30	29	28	27	26	25	24
EADC1RST	OPARST	Reserved	CAN2RST	ECAP1RST	ECAP0RST	Reserved	
23	22	21	20	19	18	17	16
QEI1RST	QEI0RST	Reserved		BPWM1RST	BPWM0RST	EPWM1RST	EPWM0RST
15	14	13	12	11	10	9	8
Reserved			DACRST	Reserved		USCI1RST	USCI0RST
7	6	5	4	3	2	1	0
Reserved	SPI3RST	Reserved	QSPI1RST	Reserved	SC2RST	SC1RST	SC0RST

位	描述	
[31]	EADC1RST	**EADC1控制器复位** 0 = EADC1控制器正常运行。 1 = EADC1控制器复位。
[30]	OPARST	**OP放大器 (OPA)控制器复位** 0 = OPA控制器正常运行。 1 = OPA控制器复位。
[29]	Reserved	保留。
[28]	CAN2RST	**CAN2控制器复位** 0 = CAN2控制器正常运行。 1 = CAN2控制器复位。
[27]	ECAP1RST	**ECAP1控制器复位** 0 = ECAP1控制器正常运行。 1 = ECAP1控制器复位。
[26]	ECAP0RST	**ECAP0控制器复位** 0 = ECAP0控制器正常运行。 1 = ECAP0控制器复位。
[25:24]	Reserved	保留。
[23]	QEI1RST	**QEI1控制器复位** 0 = QEI1控制器正常运行。 1 = QEI1控制器复位。

[22]	**QEI0RST**	**QEI0控制器复位** 0 = QEI0控制器正常运行。 1 = QEI0控制器复位。
[21:20]	**Reserved**	保留。
[19]	**BPWM1RST**	**BPWM1控制器复位** 0 = BPWM1控制器正常运行。 1 = BPWM1控制器复位。
[18]	**BPWM0RST**	**BPWM0控制器复位** 0 = BPWM0控制器正常运行。 1 = BPWM0控制器复位。
[17]	**EPWM1RST**	**EPWM1控制器复位** 0 = EPWM1控制器正常运行。 1 = EPWM1控制器复位。
[16]	**EPWM0RST**	**EPWM0控制器复位** 0 = EPWM0控制器正常运行。 1 = EPWM0控制器复位。
[15:13]	**Reserved**	保留。
[12]	**DACRST**	**DAC控制器复位** 0 = DAC控制器正常运行。 1 = DAC控制器复位。
[11:10]	**Reserved**	保留。
[9]	**USCI1RST**	**USCI1控制器复位** 0 = USCI1控制器正常运行。 1 = USCI1控制器复位。
[8]	**USCI0RST**	**USCI0控制器复位** 0 = USCI0控制器正常运行。 1 = USCI0控制器复位。
[7]	**Reserved**	保留。
[6]	**SPI3RST**	**SPI3控制器复位** 0 = SPI3控制器正常运行。 1 = SPI3控制器复位。
[5]	**Reserved**	保留。
[4]	**QSPI1RST**	**QSPI1控制器复位** 0 = QSPI1控制器正常运行。 1 = QSPI1控制器复位。
[3]	**Reserved**	保留
[2]	**SC2RST**	**SC2控制器复位** 0 = SC2控制器正常运行。

		1 = SC2控制器复位。
[1]	**SC1RST**	**SC1控制器复位** 0 = SC1控制器正常运行。 1 = SC1控制器复位。
[0]	**SC0RST**	**SC0控制器复位** 0 = SC0控制器正常运行。 1 = SC0控制器复位。

BOD控制寄存器 (SYS_BODCTL)

SYS_BODCTL控制寄存器的部分位由Flash配置初始值，部分位是写保护位。

寄存器	偏移	R/W	描述	复位值
SYS_BODCTL	SYS_BA+0x18	R/W	BOD控制寄存器	0x000X_038X

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved					BODVL		
15	14	13	12	11	10	9	8
Reserved	LVRDGSEL			Reserved	BODDGSEL		
7	6	5	4	3	2	1	0
LVREN	BODOUT	BODLPM	BODIF	BODRSTEN	Reserved		BODEN

位	描述	
[31:19]	**Reserved**	保留。
[18:16]	**BODVL**	**BOD门限电压（写保护）** 上电缺省值由CBOV（CONFIG0 [23：21]）加载进来。 000 = BOD门限电压为1.6V 。 001 = BOD门限电压为1.8V 。 010 = BOD门限电压为2.0V 。 011 = BOD门限电压为2.2V 。 100 = BOD门限电压为2.4V 。 101 = BOD门限电压为2.6V 。 110 = BOD门限电压为2.8V 。 111 = BOD门限电压为3.0V 。 **注：** 该位是写保护的。请参考SYS_REGLCTL寄存器。
[15]	**Reserved**	保留。

位	描述	
[14:12]	**LVRDGSEL**	**LVR 防抖时间（写保护）** 000 =无防抖。 001 = 4个系统时钟（HCLK）。 010 = 8个系统时钟（HCLK）。 011 = 16个系统时钟（HCLK）。 100 = 32个系统时钟（HCLK）。 101 = 64个系统时钟（HCLK）。 110 = 128个系统时钟（HCLK）。 111 = 256个系统时钟（HCLK）。 **注：**该位写保护。请参考SYS_REGLCTL寄存器。
[11]	**Reserved**	保留。
[10:8]	**BODDGSEL**	**BOD防抖时间（写保护）** 000 = BOD防抖时间一个RC10K时钟 。 001 = 防抖时间4个系统时钟（HCLK）。 010 = 防抖时间8个系统时钟（HCLK）。 011 = 防抖时间16个系统时钟（HCLK）。 100 = 防抖时间32个系统时钟（HCLK）。 101 = 防抖时间64个系统时钟（HCLK）。 110 = 防抖时间128个系统时钟（HCLK）。 111 = 防抖时间256个系统时钟（HCLK）。 **注：**该位写保护。请参考SYS_REGLCTL寄存器。
[7]	**LVREN**	**低压复位使能位（写保护）** 当输入电源电压小于LVR电路设定时，LVR功能将复位芯片。LVR功能默认开启。 0 = 低压复位禁止。 1 = 低压复位使能。 **注1：**LVR使能后，将在LVR输出稳定200us后有效（缺省状态）。 **注2**：该位是写保护的。请参考SYS_REGLCTL寄存器。
[6]	**BODOUT**	**BOD检测电路的输出状态** 0 = BOD输出为0 这表明VDD高于BODVL的设定值或者BODEN被禁止。 1 = BOD输出为1 表明VDD低于BODVL的设定值。如果BODEN为0，BOD功能禁用，该位总是0。

位	描述	
[5]	**BODLPM**	**BOD低耗电模式（写保护位）** 0 = BOD 正常模式（默认）。 1 = BOD低功耗模式使能。 **注1：** BOD在正常模式下消耗约100uA，低功率模式可以将电流降低到1/10左右，但会降低BOD响应速度。 **注2：** 该位是写保护的。请参考SYS_REGLCTL寄存器。
[4]	**BODIF**	**BOD中断标志** 0 = BOD没有检测到任何电压从V_{DD}落下或上升过设定的电压的BODVL 。 1 = 当BOD电路检测到V_{DD}由高到低跨过BODVL设定电压时或者由低到高跨过时，此位置1，若使能了中断，会触发中断。 **注：** 写1清0。
[3]	**BODRSTEN**	**BOD复位使能（写保护）** 缺省值由CBORST（CONFIG0[20]）加载进来。 0 = BOD中断功能使能 1 = BOD复位功能使能 **注1：** 当BOD检测功能使能（BODEN 为高），并且使能了BOD复位功能（BODRSTEN高），BOD电路检测到欠压时（BODOUT输高）会复位芯片。 当BOD功能使能（BODEN为高）且BOD中断功能使能（BODRSTEN为低），BOD电路检测到欠压时（BODOUT输出高）将申请中断直到BODEN 清零。BOD中断可以通过禁用NVIC BOD中断或禁用BOD功能（将BODEN设置为低）来阻止。 **注2：** 该位是写保护的。请参考SYS_REGLCTL寄存器。
[2:1]	**Reserved**	保留。
[0]	**BODEN**	**BOD使能（写保护）** 缺省值由 CBODEN（CONFIG0 [19]）。加载进来。 0 = BOD被禁用。 1 = BOD使能。 **注：** 该位是写保护的。请参考SYS_REGLCTL寄存器。

内部电压控制寄存器(SYS_IVSCTL)

寄存器	偏移	R/W	描述	复位值
SYS_IVSCTL	SYS_BA+0x1C	R/W	内部电压控制寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved						VBATUGEN	VTEMPEN

位	描述	
[31:8]	**Reserved**	保留。
[7:2]	**Reserved**	保留。
[1]	**VBATUGEN**	**V_{BAT}增益缓冲使能** 该位用于启用/禁用V_{BAT}增益缓冲器功能。 0 = 关闭V_{BAT}缓冲（默认）。 1 = 使能V_{BAT}缓冲。 **注：** 该位置1后，V_{BAT}单位增益缓冲器输出电压的值可以从ADC转换结果中获得。
[0]	**VTEMPEN**	**温度传感器使能** 该位被用来使能/禁止温度传感器功能。 0 = 温度传感器关闭（默认）。 1 = 使能温度传感器。

V_{REF}控制寄存器(SYS_VREFCTL)

寄存器	偏移	R/W	描述	复位值
SYS_VREFCTL	SYS_BA+0x28	R/W	V_{REF}控制寄存器	0x0000_0200

31	30	29	28	27	26	25	24
Reserved					VBGISEL		VBFGEN
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
PRELOAD_SEL		Reserved	VREFCTL				

位	描述	
[31:27]	**Reserved**	保留。
[26:25]	**VBGISEL**	**芯片内部 Bandgap 电流选择位（只写）** 00 = Bandgap 电压缓存电流为 4.2uA。 01 = Bandgap 电压缓存电流为 7.3uA。 10 = Bandgap 电压缓存电流为 10.4uA。 11 = Bandgap 电压缓存电流为 13.5uA。 **注:**当ADC转源选择bandgap电压时，建议置VBGISEL为10。
[24]	**VBGFEN**	**芯片内部 Bandgap 电压使能位（只写）** 0 = 内部 bandgap 电压由 ADC/ACMP 控制。 1 = 内部 bandgap 电压使能。
[23:8]	**Reserved**	保留。
[7:6]	**PRELOAD_SEL**	**预加载时间选择（写保护）** 00 = 预加载时间为60us， 0.1uF 电容。 01 =预加载时间为310us， 1uF电容。 10 =预加载时间为1270us， 4.7uF电容。 11 =预加载时间为2650us， 10uF电容。
[5]	**Reserved**	保留。

[4:0]	**VREFCTL**	**V_{REF}控制位 (写保护)** 00000=V_{REF}来自外部管脚。 00011=V_{REF}是内部 1.6V。 00111=V_{REF}是内部2.0V。 01011=V_{REF}是内部2.5V。 01111=V_{REF}是内部3.0V。 其它=保留。 **注:** 该位是写保护的。请参考SYS_REGLCTL寄存器。

USB PHY控制寄存器(SYS_USBPHY)

寄存器	偏移	R/W	描述	复位值
SYS_USBPHY	SYS_BA+0x2C	R/W	USB PHY控制寄存器	0x0003_0007

31	30	29	28	27	26	25	24
Reserved						HSUSBACT	HSUSBEN
23	22	21	20	19	18	17	16
Reserved						HSUSBROLE	
15	14	13	12	11	10	9	8
Reserved							USBEN
7	6	5	4	3	2	1	0
Reserved					SBO	USBROLE	

位	描述	
[31:26]	**Reserved**	保留。
[25]	**HSUSBACT**	**HSUSB PHY 激活控制** 该位用于控制HSUSB PHY 在复位状态还是激活状态。 0 = HSUSB PHY 在复位状态。 1 = HSUSB PHY 在激活状态。 **注:**在设置HSUSBEN (SYS_USBPHY[24]) 使能HSUSB PHY后，用户应该在改成激活模式前，保持HSUSB PHY 在复位模式至少10uS 。
[24]	**HSUSBEN**	**HSUSB PHY 使能** 该位用于使能/禁止 HSUSB PHY。 0 = HSUSB PHY 禁止。 1 = HSUSB PHY 使能。
[23:18]	**Reserved**	保留。
[17:16]	**HSUSBROLE**	**HSUSB 模式选择 (写保护)** 这两位用于选择HSUSB模式。 00 = 标准 HS USB 设备模式。 01 = 标准 HS USB 主机模式。 10 = 由 ID 引脚决定。 11 = On-The-Go设备模式。 **注:** 此位的写是受保护的，请参考寄存器 SYS_REGLCTL。
[15:9]	**Reserved**	保留。

[8]	**USBEN**	**USB PHY 使能** 0 = USB PHY 禁止。 1 = USB PHY 使能。
[7:3]	**Reserved**	保留。
[2]	**SBO**	**注:**该位必须写1，否则结果未知。
[1:0]	**USBROLE**	**USB 模式选择 (写保护)** 这两位用于选择USB模式。 00 = 标准USB 设备模式。 01 = 标准USB 主机模式。 10 = 由 ID 引脚决定。 11 = On-The-Go设备模式。 **注:** 此位的写是受保护的，请参考寄存器 SYS_REGLCTL。

GPIOA低字节多功能控制寄存器 (SYS_GPA_MFPL)

寄存器	偏移	R/W	描述	复位值
SYS_GPA_MFPL	SYS_BA+0x30	R/W	GPIOA低字节多功能控制寄存器	0x0000_0000

31	30	29	28	27	26	25	24
PA7MFP				PA6MFP			
23	22	21	20	19	18	17	16
PA5MFP				PA4MFP			
15	14	13	12	11	10	9	8
PA3MFP				PA2MFP			
7	6	5	4	3	2	1	0
PA1MFP				PA0MFP			

位	描述	
[31:28]	PA7MFP	PA.7多功能管脚选择
[27:24]	PA6MFP	PA.6多功能管脚选择
[23:20]	PA5MFP	PA.5多功能管脚选择
[19:16]	PA4MFP	PA.4多功能管脚选择
[15:12]	PA3MFP	PA.3多功能管脚选择
[11:8]	PA2MFP	PA.2多功能管脚选择
[7:4]	PA1MFP	PA.1多功能管脚选择
[3:0]	PA0MFP	PA.0多功能管脚选择

GPIOA高字节多功能控制寄存器 (SYS_GPA_MFPH)

寄存器	偏移	R/W	描述	复位值
SYS_GPA_MFPH	SYS_BA+0x34	R/W	GPIOA高字节多功能控制寄存器	0x0000_0000

31	30	29	28	27	26	25	24
PA15MFP				PA14MFP			
23	22	21	20	19	18	17	16
PA13MFP				PA12MFP			
15	14	13	12	11	10	9	8
PA11MFP				PA10MFP			
7	6	5	4	3	2	1	0
PA9MFP				PA8MFP			

位	描述	
[31:28]	PA15MFP	PA.15多功能管脚选择
[27:24]	PA14MFP	PA.14多功能管脚选择
[23:20]	PA13MFP	PA.13多功能管脚选择
[19:16]	PA12MFP	PA.12多功能管脚选择
[15:12]	PA11MFP	PA.11多功能管脚选择
[11:8]	PA10MFP	PA.10多功能管脚选择
[7:4]	PA9MFP	PA.9多功能管脚选择
[3:0]	PA8MFP	PA.8多功能管脚选择

GPIOB低字节多功能控制寄存器(SYS_GPB_MFPL)

寄存器	偏移	R/W	描述	复位值
SYS_GPB_MFPL	SYS_BA+0x38	R/W	GPIOB低字节多功能控制寄存器	0x0000_0000

31	30	29	28	27	26	25	24
PB7MFP				PB6MFP			
23	22	21	20	19	18	17	16
PB5MFP				PB4MFP			
15	14	13	12	11	10	9	8
PB3MFP				PB2MFP			
7	6	5	4	3	2	1	0
PB1MFP				PB0MFP			

位	描述	
[31:28]	PB7MFP	PB.7多功能管脚选择
[27:24]	PB6MFP	PB.6多功能管脚选择
[23:20]	PB5MFP	PB.5多功能管脚选择
[19:16]	PB4MFP	PB.4多功能管脚选择
[15:12]	PB3MFP	PB.3多功能管脚选择
[11:8]	PB2MFP	PB.2多功能管脚选择
[7:4]	PB1MFP	PB.1多功能管脚选择
[3:0]	PB0MFP	PB.0多功能管脚选择

GPIOB高字节多功能控制寄存器(SYS_GPB_MFPH)

寄存器	偏移	R/W	描述	复位值
SYS_GPB_MFPH	SYS_BA+0x3C	R/W	GPIOB高字节多功能控制寄存器	0x0000_0000

31	30	29	28	27	26	25	24
PB15MFP				PB14MFP			
23	22	21	20	19	18	17	16
PB13MFP				PB12MFP			
15	14	13	12	11	10	9	8
PB11MFP				PB10MFP			
7	6	5	4	3	2	1	0
PB9MFP				PB8MFP			

Bits	Description	
[31:28]	PB15MFP	PB.15多功能管脚选择
[27:24]	PB14MFP	PB.14多功能管脚选择
[23:20]	PB13MFP	PB.13多功能管脚选择
[19:16]	PB12MFP	PB.12多功能管脚选择
[15:12]	PB11MFP	PB.11多功能管脚选择
[11:8]	PB10MFP	PB.10多功能管脚选择
[7:4]	PB9MFP	PB.9多功能管脚选择
[3:0]	PB8MFP	PB.8多功能管脚选择

GPIOC低字节多功能控制寄存器(SYS_GPC_MFPL)

寄存器	偏移	R/W	描述	复位值
SYS_GPC_MFPL	SYS_BA+0x40	R/W	GPIOC低字节多功能控制寄存器	0x0000_0000

31	30	29	28	27	26	25	24
PC7MFP				PC6MFP			
23	22	21	20	19	18	17	16
PC5MFP				PC4MFP			
15	14	13	12	11	10	9	8
PC3MFP				PC2MFP			
7	6	5	4	3	2	1	0
PC1MFP				PC0MFP			

位	描述	
[31:28]	PC7MFP	PC.7多功能管脚选择
[27:24]	PC6MFP	PC.6多功能管脚选择
[23:20]	PC5MFP	PC.5多功能管脚选择
[19:16]	PC4MFP	PC.4多功能管脚选择
[15:12]	PC3MFP	PC.3多功能管脚选择
[11:8]	PC2MFP	PC.2多功能管脚选择
[7:4]	PC1MFP	PC.1多功能管脚选择
[3:0]	PC0MFP	PC.0多功能管脚选择

GPIOC高字节多功能控制寄存器(SYS_GPC_MFPH)

寄存器	偏移	R/W	描述	复位值
SYS_GPC_MFPH	SYS_BA+0x44	R/W	GPIOC高字节多功能控制寄存器	0x0000_0000

31	30	29	28	27	26	25	24
PC15MFP				PC14MFP			
23	22	21	20	19	18	17	16
PC13MFP				PC12MFP			
15	14	13	12	11	10	9	8
PC11MFP				PC10MFP			
7	6	5	4	3	2	1	0
PC9MFP				PC8MFP			

位	描述	
[31:28]	PC15MFP	PC.15多功能管脚选择
[27:24]	PC14MFP	PC.14多功能管脚选择
[23:20]	PC13MFP	PC.13多功能管脚选择
[19:16]	PC12MFP	PC.12多功能管脚选择
[15:12]	PC11MFP	PC.11多功能管脚选择
[11:8]	PC10MFP	PC.10多功能管脚选择
[7:4]	PC9MFP	PC.9多功能管脚选择
[3:0]	PC8MFP	PC.8多功能管脚选择

GPIOD低字节多功能控制寄存器(SYS_GPD_MFPL)

寄存器	偏移	R/W	描述	复位值
SYS_GPD_MFPL	SYS_BA+0x48	R/W	GPIOD低字节多功能控制寄存器	0x0000_0000

31	30	29	28	27	26	25	24
PD7MFP				PD6MFP			
23	22	21	20	19	18	17	16
PD5MFP				PD4MFP			
15	14	13	12	11	10	9	8
PD3MFP				PD2MFP			
7	6	5	4	3	2	1	0
PD1MFP				PD0MFP			

位	描述	
[31:28]	PD7MFP	PD.7多功能管脚选择
[27:24]	PD6MFP	PD.6多功能管脚选择
[23:20]	PD5MFP	PD.5多功能管脚选择
[19:16]	PD4MFP	PD.4多功能管脚选择
[15:12]	PD3MFP	PD.3多功能管脚选择
[11:8]	PD2MFP	PD.2多功能管脚选择
[7:4]	PD1MFP	PD.1多功能管脚选择
[3:0]	PD0MFP	PD.0多功能管脚选择

GPIOD高字节多功能控制寄存器(SYS_GPD_MFPH)

寄存器	偏移	R/W	描述	复位值
SYS_GPD_MFPH	SYS_BA+0x4C	R/W	GPIOD高字节多功能控制寄存器	0x0000_0000

31	30	29	28	27	26	25	24
PD15MFP				PD14MFP			
23	22	21	20	19	18	17	16
PD13MFP				PD12MFP			
15	14	13	12	11	10	9	8
PD11MFP				PD10MFP			
7	6	5	4	3	2	1	0
PD9MFP				PD8MFP			

位	描述	
[31:28]	PD15MFP	PD.15多功能管脚选择
[27:24]	PD14MFP	PD.14多功能管脚选择
[23:20]	PD13MFP	PD.13多功能管脚选择
[19:16]	PD12MFP	PD.12多功能管脚选择
[15:12]	PD11MFP	PD.11多功能管脚选择
[11:8]	PD10MFP	PD.10多功能管脚选择
[7:4]	PD9MFP	PD.9多功能管脚选择
[3:0]	PD8MFP	PD.8多功能管脚选择

GPIOE低字节多功能控制寄存器(SYS_GPE_MFPL)

寄存器	偏移	R/W	描述	复位值
SYS_GPE_MFPL	SYS_BA+0x50	R/W	GPIOE低字节多功能控制寄存器	0x0000_0000

31	30	29	28	27	26	25	24
PE7MFP				PE6MFP			
23	22	21	20	19	18	17	16
PE5MFP				PE4MFP			
15	14	13	12	11	10	9	8
PE3MFP				PE2MFP			
7	6	5	4	3	2	1	0
PE1MFP				PE0MFP			

位	描述	
[31:28]	PE7MFP	PE.7多功能管脚选择
[27:24]	PE6MFP	PE.6多功能管脚选择
[23:20]	PE5MFP	PE.5多功能管脚选择
[19:16]	PE4MFP	PE.4多功能管脚选择
[15:12]	PE3MFP	PE.3多功能管脚选择
[11:8]	PE2MFP	PE.2多功能管脚选择
[7:4]	PE1MFP	PE.1多功能管脚选择
[3:0]	PE0MFP	PE.0多功能管脚选择

GPIOE高字节多功能控制寄存器(SYS_GPE_MFPH)

寄存器	偏移	R/W	描述	复位值
SYS_GPE_MFPH	SYS_BA+0x54	R/W	GPIOE高字节多功能控制寄存器	0x0000_0000

31	30	29	28	27	26	25	24
PE15MFP				PE14MFP			
23	22	21	20	19	18	17	16
PE13MFP				PE12MFP			
15	14	13	12	11	10	9	8
PE11MFP				PE10MFP			
7	6	5	4	3	2	1	0
PE9MFP				PE8MFP			

位	描述	
[31:28]	PE15MFP	PE.15多功能管脚选择
[27:24]	PE14MFP	PE.14多功能管脚选择
[23:20]	PE13MFP	PE.13多功能管脚选择
[19:16]	PE12MFP	PE.12多功能管脚选择
[15:12]	PE11MFP	PE.11多功能管脚选择
[11:8]	PE10MFP	PE.10多功能管脚选择
[7:4]	PE9MFP	PE.9多功能管脚选择
[3:0]	PE8MFP	PE.8多功能管脚选择

GPIOF低字节多功能控制寄存器(SYS_GPF_MFPL)

寄存器	偏移	R/W	描述	复位值
SYS_GPF_MFPL	SYS_BA+0x58	R/W	GPIOF低字节多功能控制寄存器	0x0000_00EE

31	30	29	28	27	26	25	24
PF7MFP				PF6MFP			
23	22	21	20	19	18	17	16
PF5MFP				PF4MFP			
15	14	13	12	11	10	9	8
PF3MFP				PF2MFP			
7	6	5	4	3	2	1	0
PF1MFP				PF0MFP			

位	描述	
[31:28]	PF7MFP	PF.7多功能管脚选择
[27:24]	PF6MFP	PF.6多功能管脚选择
[23:20]	PF5MFP	PF.5多功能管脚选择
[19:16]	PF4MFP	PF.4多功能管脚选择
[15:12]	PF3MFP	PF.3多功能管脚选择
[11:8]	PF2MFP	PF.2多功能管脚选择
[7:4]	PF1MFP	PF.1多功能管脚选择
[3:0]	PF0MFP	PF.0多功能管脚选择

GPIOF高字节多功能控制寄存器(SYS_GPF_MFPH)

寄存器	偏移	R/W	描述	复位值
SYS_GPF_MFPH	SYS_BA+0x5C	R/W	GPIOF高字节多功能控制寄存器	0x0000_0000

31	30	29	28	27	26	25	24
PF15MFP				PF14MFP			
23	22	21	20	19	18	17	16
PF13MFP				PF12MFP			
15	14	13	12	11	10	9	8
PF11MFP				PF10MFP			
7	6	5	4	3	2	1	0
PF9MFP				PF8MFP			

位	描述	
[31:28]	PF15MFP	PF.15多功能管脚选择
[27:24]	PF14MFP	PF.14多功能管脚选择
[23:20]	PF13MFP	PF.13多功能管脚选择
[19:16]	PF12MFP	PF.12多功能管脚选择
[15:12]	PF11MFP	PF.11多功能管脚选择
[11:8]	PF10MFP	PF.10多功能管脚选择
[7:4]	PF9MFP	PF.9多功能管脚选择
[3:0]	PF8MFP	PF.8多功能管脚选择

GPIOG低字节多功能控制寄存器(SYS_GPG_MFPL)

寄存器	偏移	R/W	描述	复位值
SYS_GPG_MFPL	SYS_BA+0x60	R/W	GPIOG低字节多功能控制寄存器	0x0000_0000

31	30	29	28	27	26	25	24
PG7MFP				PG6MFP			
23	22	21	20	19	18	17	16
PG5MFP				PG4MFP			
15	14	13	12	11	10	9	8
PG3MFP				PG2MFP			
7	6	5	4	3	2	1	0
PG1MFP				PG0MFP			

位	描述	
[31:28]	PG7MFP	PG.7多功能管脚选择
[27:24]	PG6MFP	PG.6多功能管脚选择
[23:20]	PG5MFP	PG.5多功能管脚选择
[19:16]	PG4MFP	PG.4多功能管脚选择
[15:12]	PG3MFP	PG.3多功能管脚选择
[11:8]	PG2MFP	PG.2多功能管脚选择
[7:4]	PG1MFP	PG.1多功能管脚选择
[3:0]	PG0MFP	PG.0多功能管脚选择

GPIOG高字节多功能控制寄存器(SYS_GPG_MFPH)

寄存器	偏移	R/W	描述	复位值
SYS_GPG_MFPH	SYS_BA+0x64	R/W	GPIOG高字节多功能控制寄存器	0x0000_0000

31	30	29	28	27	26	25	24
PG15MFP				PG14MFP			
23	22	21	20	19	18	17	16
PG13MFP				PG12MFP			
15	14	13	12	11	10	9	8
PG11MFP				PG10MFP			
7	6	5	4	3	2	1	0
PG9MFP				PG8MFP			

位	描述	
[31:28]	PG15MFP	PG.15多功能管脚选择
[27:24]	PG14MFP	PG.14多功能管脚选择
[23:20]	PG13MFP	PG.13多功能管脚选择
[19:16]	PG12MFP	PG.12多功能管脚选择
[15:12]	PG11MFP	PG.11多功能管脚选择
[11:8]	PG10MFP	PG.10多功能管脚选择
[7:4]	PG9MFP	PG.9多功能管脚选择
[3:0]	PG8MFP	PG.8多功能管脚选择

GPIOH低字节多功能控制寄存器(SYS_GPH_MFPL)

寄存器	偏移	R/W	描述	复位值
SYS_GPH_MFPL	SYS_BA+0x68	R/W	GPIOH低字节多功能控制寄存器	0x0000_0000

31	30	29	28	27	26	25	24
PH7MFP				PH6MFP			
23	22	21	20	19	18	17	16
PH5MFP				PH4MFP			
15	14	13	12	11	10	9	8
PH3MFP				PH2MFP			
7	6	5	4	3	2	1	0
PH1MFP				PH0MFP			

位	描述	
[31:28]	PH7MFP	PH.7多功能管脚选择
[27:24]	PH6MFP	PH.6多功能管脚选择
[23:20]	PH5MFP	PH.5多功能管脚选择
[19:16]	PH4MFP	PH.4多功能管脚选择
[15:12]	PH3MFP	PH.3多功能管脚选择
[11:8]	PH2MFP	PH.2多功能管脚选择
[7:4]	PH1MFP	PH.1多功能管脚选择
[3:0]	PH0MFP	PH.0多功能管脚选择

GPIOH高字节多功能控制寄存器(SYS_GPH_MFPH)

寄存器	偏移	R/W	描述	复位值
SYS_GPH_MFPH	SYS_BA+0x6C	R/W	GPIOH高字节多功能控制寄存器	0x0000_0000

31	30	29	28	27	26	25	24
PH15MFP				PH14MFP			
23	22	21	20	19	18	17	16
PH13MFP				PH12MFP			
15	14	13	12	11	10	9	8
PH11MFP				PH10MFP			
7	6	5	4	3	2	1	0
PH9MFP				PH8MFP			

位	描述	
[31:28]	PH15MFP	PH.15多功能管脚选择
[27:24]	PH14MFP	PH.14多功能管脚选择
[23:20]	PH13MFP	PH.13多功能管脚选择
[19:16]	PH12MFP	PH.12多功能管脚选择
[15:12]	PH11MFP	PH.11多功能管脚选择
[11:8]	PH10MFP	PH.10多功能管脚选择
[7:4]	PH9MFP	PH.9多功能管脚选择
[3:0]	PH8MFP	PH.8多功能管脚选择

GPIO A-H多功能输出选择寄存器 (SYS_GPx_MFOS)

寄存器	偏移	R/W	描述	复位值
SYS_GPA_MFOS	SYS_BA+0x80	R/W	GPIOA多功能输出选择寄存器	0x0000_0000
SYS_GPB_MFOS	SYS_BA+0x84	R/W	GPIOB多功能输出选择寄存器	0x0000_0000
SYS_GPC_MFOS	SYS_BA+0x88	R/W	GPIOC多功能输出选择寄存器	0x0000_0000
SYS_GPD_MFOS	SYS_BA+0x8C	R/W	GPIOD多功能输出选择寄存器	0x0000_0000
SYS_GPE_MFOS	SYS_BA+0x90	R/W	GPIOE多功能输出选择寄存器	0x0000_0000
SYS_GPF_MFOS	SYS_BA+0x94	R/W	GPIOF多功能输出选择寄存器	0x0000_0000
SYS_GPG_MFOS	SYS_BA+0x98	R/W	GPIOG多功能输出选择寄存器	0x0000_0000
SYS_GPH_MFOS	SYS_BA+0x9C	R/W	GPIOH多功能输出选择寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
MFOS							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
MFOS							

位	描述	
[31:16]	**Reserved**	保留。
[n] n=0,1..15	**MFOS**	**GPIOA-h Pin[n] 多功能管脚输出模式选择** 该位用于Px.n 管脚的输出模式类型。 0 =引脚为推挽输出模式。 1 =引脚为开漏输出模式。 **注:** 对于A/B/E/G，n最大=15。 对于C/D，n最大=14。 对于F/H，n最大=11。

SRAM奇偶校验错误中断使能控制寄存器 (SYS_SRAM_INTCTL)

寄存器	偏移	R/W	描述	复位值
SYS_SRAM_INTCTL	SYS_BA+0xC0	R/W	SRAM中断使能控制寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved							PERRIEN

位	描述	
[31:1]	Reserved	保留。
[0]	PERRIEN	SRAM奇偶校验错误中断使能位 0 = SRAM奇偶校验错误中断禁用。 1 = SRAM奇偶校验错误中断使能。

SRAM奇偶校验状态寄存器(SYS_SRAM_STATUS)

寄存器	偏移	R/W	描述	复位值
SYS_SRAM_STATUS	SYS_BA+0xC4	R/W	SRAM奇偶校验错误状态寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved							PERRIF

位	描述	
[31:1]	Reserved	保留。
[0]	PERRIF	SRAM奇偶校验错误标志 该位表示SRAM奇偶校验错误发生，写1清0。 0 = SRAM 无奇偶校验错。 1 = SRAM奇偶校验出错。

SRAM奇偶校验错误地址寄存器(SYS_SRAM_ERRADDR)

寄存器	偏移	R/W	描述	复位值
SYS_SRAM_ERRADDR	SYS_BA+0xC8	R	SRAM奇偶校验错误地址寄存器	0x0000_0000

31	30	29	28	27	26	25	24
ERRADDR							
23	22	21	20	19	18	17	16
ERRADDR							
15	14	13	12	11	10	9	8
ERRADDR							
7	6	5	4	3	2	1	0
ERRADDR							

位	描述	
[31:0]	ERRADDR	**SRAM奇偶校验错误地址** 该寄存器表示SRAM奇偶校验出错的地址。

SRAM BIST测试控制寄存器 (SYS_SRAM_BISTCTL)

寄存器	偏移	R/W	描述	复位值
SYS_SRAM_BISTCTL	SYS_BA+0xD0	R/W	SRAM BIST测试控制寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved						HSUSBHBIST	HSUSBDBIST
7	6	5	4	3	2	1	0
PDMABIST	EMCBIST	SPIMBIST	USBBIST	CANBIST	CRBIST	SRBIST1	SRBIST0

位	描述	
[31:10]	**Reserved**	保留。
[9]	**HSUSBHBIST**	**HSUSBH BIST使能位（写保护）** 该位使能HSUSBH RAM 的BIST测试。 0 = HSUSBH BIST测试禁止。 1 = HSUSBH BIST测试使能。 **注:** 该位为写保护，具体参考SYS_REGLCTL 寄存器。
[8]	**HSUSBDBIST**	**HSUSBD BIST使能位（写保护）** 该位使能HSUSBD RAM 的BIST测试。 0 = HSUSBD BIST测试禁止。 1 = HSUSBD BIST测试使能。 **注:** 该位为写保护，具体参考SYS_REGLCTL 寄存器。
[7]	**PDMABIST**	**PDMA BIST使能位（写保护）** 该位使能PDMA RAM 的BIST测试。 0 = PDMA BIST测试禁止。 1 = PDMA BIST测试使能。 **注:** 该位为写保护，具体参考SYS_REGLCTL 寄存器。
[6]	**EMCBIST**	**EMC BIST使能位（写保护）** 该位使能EMCRAM 的BIST测试。 0 = EMCBIST测试禁止。 1 = EMCBIST测试使能。 **注:** 该位为写保护，具体参考SYS_REGLCTL 寄存器。
[5]	**SPIMBIST**	**SPIM BIST使能位（写保护）**

		该位使能SPIM RAM 的BIST测试。 0 = SPIM BIST测试禁止。 1 = SPIM BIST测试使能。 **注:** 该位为写保护，具体参考SYS_REGLCTL 寄存器。
[4]	**USBBIST**	**USB BIST使能位（写保护）** 该位使能USB RAM 的BIST测试。 0 = USB BIST测试禁止。 1 = USB BIST测试使能。 **注:** 该位为写保护，具体参考SYS_REGLCTL 寄存器。
[3]	**CANBIST**	**CAN BIST使能位（写保护）** 该位使能CAN RAM 的BIST测试。 0 = CAN BIST测试禁止。 1 = CAN BIST测试使能。 **注:** 该位为写保护，具体参考SYS_REGLCTL 寄存器。
[2]	**CRBIST**	**CACHE BIST使能位（写保护）** 该位使能CACHE RAM 的BIST测试。 0 = CACHE BIST测试禁止。 1 = CACHE BIST测试使能。 **注:** 该位为写保护，具体参考SYS_REGLCTL 寄存器。
[1]	**SRBIST1**	**SRAM Bank1BIST使能位（写保护）** 该位使能SRAM bank1的BIST测试。 0 = SRAM bank1BIST测试禁止。 1 = SRAM bank1BIST测试使能。 **注:** 该位为写保护，具体参考SYS_REGLCTL 寄存器。
[0]	**SRBIST0**	**SRAM Bank0 BIST使能位（写保护）** 该位使能SRAM bank0的BIST测试。 0 = SRAM bank0 BIST测试禁止。 1 = SRAM bank0BIST测试使能。 **注:** 该位为写保护，具体参考SYS_REGLCTL 寄存器。

SRAM BIST测试状态寄存器(SYS_SRAM_BISTSTS)

寄存器	偏移	R/W	描述	复位值
SYS_SRAM_BISTSTS	SYS_BA+0xD4	R	SRAM BIST测试状态寄存器	0x00xx_00xx

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
PDMABEND	Reserved		USBBEND	CANBEND	CRBEND	SRBEND1	SRBEND0
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
PDMABEF	Reserved		USBBEF	CANBEF	CRBISTEF	SRBISTEF1	SRBISTEF0

位	描述	
[31:24]	**Reserved**	保留。
[23]	**PDMABEND**	**PDMA SRAM BIST 测试完成** 0 = PDMA SRAM BIST正在测试。 1 = PDMA SRAM BIST 测试完成。
[22:21]	**Reserved**	保留。
[20]	**USBBEND**	**USB SRAM BIST 测试完成** 0 = USB SRAM BIST正在测试。 1 = USB SRAM BIST测试完成。
[19]	**CANBEND**	**CAN SRAM BIST 测试完成** 0 = CAN SRAM BIST正在测试。 1 = CAN SRAM BIST测试完成。
[18]	**CRBEND**	**CACHE SRAM BIST 测试完成** 0 = CACHE SRAM BIST正在测试。 1 = CACHE SRAM BIST测试完成。
[17]	**SRBEND1**	**2nd SRAM BIST 测试完成** 0 = 2nd SRAM BIST正在测试。 1 = 2ndSRAM BIST测试完成。
[16]	**SRBEND0**	**1st SRAM BIST 测试完成** 0 = 1st SRAM BIST正在测试。 1 = 1stSRAM BIST测试完成。
[15:8]	**Reserved**	保留。

[7]	**PDMABEF**	**PDMA SRAM BIST 识别标志** 0 = PDMA SRAM BIST测试通过。 1 = PDMA SRAM BIST测试失败。
[6:5]	**Reserved**	保留。
[4]	**USBBEF**	**USB SRAM BIST 识别标志** 0 = USB SRAM BIST测试通过。 1 = USB SRAM BIST测试失败。
[3]	**CANBEF**	**CAN SRAM BIST 识别标志** 0 = CAN SRAM BIST测试通过。 1 = CAN SRAM BIST测试失败。
[2]	**CRBISTEF**	**CACHE SRAM BIST 识别标志** 0 = CACHE SRAM BIST测试通过。 1 = CACHE SRAM BIST测试失败。
[1]	**SRBISTEF1**	**2nd SRAM BIST 识别标志** 0 = 2nd SRAM BIST测试通过。 1 = 2nd SRAM BIST测试失败。
[0]	**SRBISTEF0**	**1st SRAM BIST 识别标志** 0 = 1nd SRAM BIST测试通过。 1 = 1nd SRAM BIST测试失败。

HIRC48M校准控制寄存器(SYS_HIRCTCTL)

寄存器	偏移	R/W	描述	复位值
SYS_HIRCTCTL	SYS_BA+0xE4	R/W	HIRC48M校准控制寄存器	0x0008_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved			BOUNDARY				
15	14	13	12	11	10	9	8
Reserved					REFCKSEL	BOUNDEN	CESTOPEN
7	6	5	4	3	2	1	0
RETRYCNT		LOOPSEL		Reserved		FREQSEL	

位	描述	
[31:21]	**Reserved**	保留。
[20:16]	**BOUNDARY**	**边界选择** 边界范围在0x1到0x31，0x0保留。 **注：**该字段当BOUNDEN（SYS_HIRCTRIMCTL[9]）使能时有效。
[15:11]	**Reserved**	保留。
[10]	**REFCKSEL**	**参考时钟选择** 0 = HIRC校准参考时钟来自LXT（32.768 kHz）。 1 = HIRC校准参考时钟来自内部USB同步模式。 **注：**在测试模式HIRC校准参考时钟为20kHz。
[9]	**BOUNDEN**	**边界使能位** 0 = 边界功能禁止。 1 = 边界功能使能。
[8]	**CESTOPEN**	**时钟错误停止使能位** 0 =如果时钟不准确，微调操作将继续进行。 1 =如果时钟不准确，微调操作将停止。
[7:6]	**RETRYCNT**	**修正值更新限制计数** 该字段定义自动微调电路在HIRC锁定频率之前尝试更新HIRC微调值的次数。 一旦HIRC锁定，内部校准值更新计数器将被重置。 如果微调值更新计数器达到此限制值，且HIRC的频率仍不锁定，则自动微调操作将被禁用，并将FREQSEL清除为00。 00 =校准重试计数限制在64次。

		01 =校准重试计数限制在128次。 10 =校准重试计数限制在256次。 11 =校准重试计数限制在512次。
[5:4]	**LOOPSEL**	**校准计算循环选择** 该字段定义校准值计算基于多少个参考时钟。 00 =校准值计算基于参考时钟4个时钟的平均差。 01 =校准值计算基于参考时钟8个时钟的平均差。 10 =校准值计算基于参考时钟16个时钟的平均差。 11 =校准值计算基于参考时钟32个时钟的平均差。 **注：**例如，如果LOOPSEL设置位00，自动校准电流将会计算校准值基于参考时钟4个时钟的平均值。
[3:2]	**Reserved**	保留。
[1:0]	**FREQSEL**	**校准频率选择** 该字段表示内部高速RC振荡器（HIRC）自动校准到48MHz目标频率。 在校准操作时，如果检测到时钟错误，CESTOPEN被置1或校准限制次数到达，该字段将被自动请00。 00 =禁止HIRC自动校准功能。 01 =使能HIRC自动校准功能且校准HIRC到48MHz。 10 = 保留。 11 = 保留。

HIRC48M校准中断使能寄存器 (SYS_HIRCTIEN)

寄存器	偏移	R/W	描述	复位值
SYS_HIRCTIEN	SYS_BA+0xE8	R/W	HIRC48M校准中断使能寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved					CLKEIEN	TFAILIEN	Reserved

位	描述	
[31:3]	Reserved	保留。
[2]	CLKEIEN	**时钟错误中断使能位** 该位控制如果在自动微调操作期间，当时钟不准确时，CPU是否会收到中断。 如果该位设置为1，并且在自动微调操作期间设置了CLKERRIF（SYS_HIRCTRIMSTS[2]），则会触发一个中断，通知时钟频率不准确。 0 = 禁止 CLKERRIF(SYS_IRCTISTS[2]) 状态去触发CPU的中断。 1 = 使能 CLKERRIF(SYS_IRCTISTS[2]) 状态去触发CPU的中断。
[1]	TFAILIEN	**校准失败中断使能位** 该位控制当达到HIRC校准值更新限制计数且HIRC频率仍不锁定在由FREQSEL（SYS_IRCTCTL [1：0]）设置的目标频率时是否触发中断。 如果该位为高，并且在自动微调操作期间设置了TFAILIF（SYS_IRCTISTS [1]），则会触发一个中断，通知已达到HIRC校准值更新限制计数。 0 =禁止TFAILIF（SYS_IRCTISTS [1]）状态去触发CPU的中断。 1 =使能TFAILIF（SYS_ IRCTISTS [1]）状态去触发CPU的中断。
[0]	Reserved	保留。

HIRC48M校准中断状态寄存器(SYS_HIRCTISTS)

寄存器	偏移	R/W	描述	复位值
SYS_HIRCTISTS	SYS_BA+0xEC	R/W	HIRC48M校准中断状态寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved				OVBDIF	CLKERRIF	TFAILIF	FREQLOCK

位	描述	
[31:4]	Reserved	保留。
[3]	OVBDIF	**越界状态** 当越界功能被设置，如果发生了越界，此标志将被置位。 0 = 没有发生越界现象 1 = 发生越界现象 **注：**写清除标志。
[2]	CLKERRIF	**时钟错误中断状态** 当32.768kHz外部低速晶振（LXT）或48MHz内部高速RC振荡器（HIRC）的频率被移到不合理的值时，该位将被置位，表示时钟频率不准确。 一旦该位设置为1，自动校准操作停止，并且如果CESTOPEN（SYS HIRCTRIMCTL[8]）设置为1，硬件会自动将FREQSEL（SYS_IRCTCL[1:0]）清除为00。 如果该位置1且CLKEIEN（SYS_IRCTIEN[2]）为高，将会产生中断通知时钟频率不准确。 该位写1清0。 0 = 时钟频率准确。 1 = 时钟频率不准确。
[1]	TFAILIF	**校准失败中断状态** 该位表示达到了HIRC校准值更新限制计数，HIRC时钟频率仍没锁定。一旦设置了该位，自动校准操作停止，硬件自动将FREQSEL（SYS_ IRCTCTL [1：0]）清除为00。 如果设置了该位且TFAILIEN（SYS_ IRCTIEN [1]）为高，则会触发一个中断，通知已达到HIRC校准值更新限制计数。写入1清除为0。 0 =校准值更新限制计数未到达。 1 =校准值限制测试达到且HIRC频率仍没有锁定。
[0]	FREQLOCK	**HIRC频率锁定状态** 该位只是HIRC频率被锁定。

		该位是状态位，不触发任何中断。 写1清除为0。如果频率锁定并且RC_TRIM使能，此位将自动设置。 0 = 内部高速振荡器频率没锁定在48MHz 。 1 = 内部高速振荡器频率锁定在48MHz。

HIRC校准控制寄存器 (SYS_IRCTCTL)

寄存器	偏移	R/W	模式	复位值
SYS_IRCTCTL	SYS_BA+0xF0	R/W	HIRC校准控制寄存器	0x0008_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved			BOUNDARY				
15	14	13	12	11	10	9	8
Reserved					REFCKSEL	BOUNDEN	CESTOPEN
7	6	5	4	3	2	1	0
RETRYCNT		LOOPSEL		Reserved		FREQSEL	

位	描述	
[31:21]	**Reserved**	保留。
[20:16]	**BOUNDARY**	**边界选择** 边界范围在0x1到0x1F，0x0保留。 **注1：**该字段BOUNDEN（SYS_HIRCTRIMCTL[9]）使能时有效。
[15:11]	**Reserved**	保留。.
[10]	**REFCKSEL**	**参考时钟选择** 0 = HIRC校准参考时钟来自LXT（32.768 kHz）。 1 = HIRC校准参考时钟来自内部USB同步模式。 **注:**在测试模式HIRC校准参考时钟为20kHz。
[9]	**BOUNDEN**	**边界使能位** 0 = 边界功能禁止。 1 = 边界功能使能。
[8]	**CESTOPEN**	**时钟错误停止使能位** 0 = 如果时钟不精准，校准操作将继续进行。 1 = 如果时钟不精准，校准操作将停止。
[7:6]	**RETRYCNT**	**校准值更新限制计数** 该字段定义自动校准电路在HIRC锁定频率之前尝试更新HIRC校准值的次数。 一旦HIRC锁定，内部校准值更新计数器将被重置。 如果校准值更新计数器达到此限制值，且HIRC的频率仍不锁定，则自动校准操作将被禁用，并将FREQSEL清除为00。 00 = 校准重试计数限制在64次。 01 = 校准重试计数限制在128次。 10 = 校准重试计数限制在256次。 11 = 校准重试计数限制在512次。

[5:4]	**LOOPSEL**	**校准计数循环选择** 该字段定义校准值计算基于多少个参考时钟。 00 =.校准值计算基于参考时钟4个时钟的平均差。 01 = 校准值计算基于参考时钟8个时钟的平均差。 10 = 校准值计算基于参考时钟16个时钟的平均差。 11 = 校准值计算基于参考时钟32个时钟的平均差。 **注：**例如，如果LOOPSEL设置位00，自动校准电流将会计算校准值基于参考时钟4个时钟的平均值。
[3:2]	**Reserved**	保留。
[1:0]	**FREQSEL**	**校准频率选择** 该字段表示内部高速RC振荡器（HIRC）自动校准到12MHz目标频率。 在校准操作时，如果检测到时钟错误，CESTOPEN被置1或校准限制次数到达，该字段将被自动请00。 00 = 禁止HIRC自动校准功能。 01 = 使能HIRC自动校准功能且校准HIRC到12MHz。 10 = 保留。 11 = 保留。

HIRC校准中断使能寄存器(SYS_IRCTIEN)

寄存器	偏移	R/W	描述	复位值
SYS_IRCTIEN	SYS_BA+0xF4	R/W	HIRC校准中断使能寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved					CLKEIEN	TFAILIEN	Reserved

位	描述	
[31:3]	Reserved	保留。
[2]	CLKEIEN	**时钟错误中断使能位** 该位控制如果在自动校准操作期间，当时钟不准确时，CPU是否会收到中断。 如果该位设置为1，并且在自动校准操作期间设置了CLKERRIF（SYS_ IRCTISTS [2]），则会触发一个中断，通知时钟频率不准确。 0 = 禁止CLKERRIF（SYS_ IRCTISTS [2]）状态去触发CPU的中断。 1 = 使能CLKERRIF（SYS_ IRCTISTS [2]）状态去触发CPU的中断。
[1]	TFAILIEN	**校准失败中断使能位** 该位控制当达到HIRC校准值更新限制计数且HIRC频率仍不锁定在由FREQSEL（SYS_ IRCTCTL [1：0]）.设置的目标频率时是否触发中断。 如果该位为高，并且在自动微调操作期间设置了TFAILIF（SYS_ IRCTISTS [1]），则会触发一个中断，通知已达到HIRC微调值更新限制计数。 0 = 禁止TFAILIF（SYS_ IRCTISTS [1]）状态去触发CPU的中断。 1 = 使能TFAILIF（SYS_ IRCTISTS [1]）状态去触发CPU的中断。
[0]	Reserved	保留。

HIRC校准中断状态寄存器 (SYS_IRCTISTS)

寄存器	偏移	R/W	描述	复位值
SYS_IRCTISTS	SYS_BA+0xF8	R/W	HIRC校准中断状态寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved				OVBDIF	CLKERRIF	TFAILIF	FREQLOCK

位	描述	
[31:4]	**Reserved**	保留。
[3]	**OVBDIF**	**越界状态** 当越界功能被设置，如果发生越界，该标志将置位。 0 = 没有越界现象发生。 1 = 发生越界现象。 **注 1:**写1清0。
[2]	**CLKERRIF**	**时钟错误中断状态** 当32.768kHz外部低速晶振（LXT）或12MHz内部高速RC振荡器（HIRC）的频率被移到不合理的值时，该位将被置位，表示时钟频率不准确。 一旦该位设置为1，自动校准操作停止，并且如果CESTOPEN（SYS IRCTCTL [8]）设置为1，硬件会自动将FREKSEL（SYS IRCTCL [1：0]）清除为00。 0 = 时钟频率准确 1 = 时钟频率不准确
[1]	**TFAILIF**	**校准识别中断状态** 该位表示达到了HIRC校准值更新限制计数，HIRC时钟频率仍没锁定。一旦设置了该位，自动校准操作停止，硬件自动将FREQSEL（SYS_ IRCTCTL [1：0]）清除为00。 如果设置了该位且TFAILIEN（SYS_ IRCTIEN [1]）为高，则会触发一个中断，通知已达到HIRC校准值更新限制计数。写入1清除为0。 0 = 校准值更新限制计数未到达。 1 = 校准值限制测试达到且HIRC频率仍没有锁定。
[0]	**FREQLOCK**	**HIRC频率锁定状态** 该位表示HIRC频率被锁定。 该状态位不触发任何中断。 写1清0。如果频率被锁定且RC_TRIM使能，该位将自动置位。 0 = 内部高速振荡器频率没有锁定在12 MHz。

		1 =内部高速振荡器频率锁定在12 MHz。

寄存器锁定控制寄存器 (SYS_REGLCTL)

一些系统控制寄存器需要被保护防止代码误写或干扰芯片操作，这些系统控制寄存器在上电复位后被保护直到用户禁止寄存器保护。为了让用户对这些受保护的寄存器进行编程，需要在寄存器保护禁用时序之后再进行一个特殊的编程。寄存器保护禁用时序将数据“59h”、“16h”、“88h”连续写入寄存器SYS_REGLCTL地址0x4000_0100。在这三个数据写入过程中，任何不同的数据值、不同的时序或任何其他对其他地址的写入都将中止整个时序。

禁用保护后，用户可以检查地址0x4000 _0100的保护禁用位0，1为保护禁用，0为保护启用。然后，用户可以更新目标保护寄存器值，然后将任何数据写入地址“0x4000_0100”以启用寄存器保护。

寄存器被写入以禁止/启用寄存器保护并读取REGLCTL状态。

寄存器	偏移	R/W	描述	复位值
SYS_REGLCTL	SYS_BA+0x100	R/W	寄存器锁定控制寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
REGLCTL							

位	描述	
[31:8]	Reserved	保留。
[7:0]	REGLCTL	**解锁代码(只写)** 一些寄存器有写保护功能。写这些寄存器需要禁止保护功能，通过依次写“59h”，“16h”，“88h”到这些寄存器禁止保护。当这个时序完成，REGLCTL位将被置1，写保护寄存器可正常写。 **REGLCTL[0]** **寄存器锁定控制禁用索引（只读）** 0 = 写保护寄存器启用写保护。忽略对受保护寄存器的任何写入 1 = 写保护寄存器禁止写保护

模拟POR禁用控制寄存器(SYS_PORDISAN)

寄存器	偏移	R/W	描述	复位值
SYS_PORDISAN	SYS_BA+0x1EC	R/W	模拟POR禁用控制寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
POROFFAN							
7	6	5	4	3	2	1	0
POROFFAN							

位	描述	
[31:16]	**Reserved**	保留。
[15:0]	**POROFFAN**	**上电复位使能位（写保护）** 上电后，用户通过写0x5AA5到该寄存器关闭内部模拟POR电路以降低功耗。 当该寄存器写入别的值或者芯片通过其他复位源复位，模拟POR电路将再次有效，其他复位源包括： nRESET，看门狗复位，LVR复位，BOD复位，ICE复位命令和软件复位芯片。 **注:** 该位写保护，参考SYS_REGLCTL 寄存器。 **注:**当系统进入休眠模式M48xID/M48xGA 系列通过硬件禁用POR 。

芯片系列版本寄存器 (SYS_CSERVER)

寄存器	偏移	R/W	描述	复位值
SYS_CSERVER	SYS_BA+0x1F4	R	芯片系列版本寄存器	0x0000_00XX

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
VERSION							

位	描述	
[31:8]	Reserved	保留。
[7:0]	VERSION	**芯片系列版本** 这些位表明芯片的系列版本。 00 = M48xID/M48xGA。 01 = M48xGC/M48xE8。 其它 = 保留。

电源模式控制寄存器(SYS_PLCTL)

寄存器	偏移	R/W	描述	复位值
SYS_PLCTL	SYS_BA+0x1F8	R/W	电源模式控制寄存器	0x0000_0000

31	30	29	28	27	26	25	24
LVSPRD							
23	22	21	20	19	18	17	16
Reserved		LVSSTEP					
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved						PLSEL	

位	描述	
[31:24]	**LVSPRD**	**LDO 电压调整周期（写保护）** LVSPRD 的值为LDO电压每改变一步的周期时间。 LDO 电压调整周期= (LVSPRD + 1) * 1us。
[23:22]	**Reserved**	保留。
[21:16]	**LVSSTEP**	**LDO 电压增加步数（写保护）** LVSSTEP 的值为LDO电压调整的步进值。 LDO 电压增量= (LVSSTEP + 1) * 10mV。
[15:2]	**Reserved**	保留。
[1:0]	**PLSEL**	**内核电压选择（写保护）** 该位表示内核电压的状态。 00 = 电源模式为PL0。 01 = 电源模式为 PL1。 其它 = 保留。

电源模式状态寄存器(SYS_PLSTS)

寄存器	偏移	R/W	描述	复位值
SYS_PLSTS	SYS_BA+0x1FC	R	电源模式状态寄存器	0x0000_010X

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved						PLSTATUS	
7	6	5	4	3	2	1	0
Reserved							PLCBUSY

位	描述	
[31:10]	**Reserved**	保留。
[9:8]	**PLSTATUS**	**电压状态（只读）** 该域反映当前的电压。 00 = 电源模式设定PL0。 01 = 电源模式设定PL1。 其他 = 保留
[7:1]	**Reserved**	保留。
[0]	**PLCBUSY**	**电压改变指示 (只读)** 当电压改变时，该位通过硬件设置。电压改变完成该位硬件自动清零。 0 = 电压改变完成。 1 = 电压正在改变。

AHB总线矩阵优先级控制寄存器(SYS_AHBMCTL)

寄存器	偏移	R/W	描述	复位值
SYS_AHBMCTL	SYS_BA+0x400	R/W	AHB总线矩阵优先级控制寄存器	0x0000_0001

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved							INTACTEN

位	描述	
[31:1]	**Reserved**	保留。
[0]	**INTACTEN**	**Cortex M4 内核最高AHB总线优先级使能位 (写保护位)** 使能Cortex M4 内核最高AHB总线优先级。 0 = 轮换模式。 1 = 中断发生时，Cortex®-M4 CPU 有最高优先级。 **注:** 此位的写是受保护的，请参考寄存器 SYS_REGLCTL。

6.2.13 系统定时器(SysTick)

Cortex®-M4包含一个集成的系统计时器Systick，它提供了一个简单的24位读写、减量、零计数器，并具有灵活的控制机制。该计数器可以用作实时操作系统（RTOS）时钟或简单计数器。

当系统计时器启动时，它将从SysTick当前值寄存器（SYST_VAL）中的值倒计时到零，并在下一个时钟周期中重新加载（wrap）到SysTick重新加载值寄存器（SYST_LOAD）中的值，然后在随后的时钟上递减。当计数器转换为零时，将设置COUNTFLAG状态位。COUNTFLAG位在读取时清除。

在复位时SYST_VAL值是未知的。软件应该在使能前写寄存器清零。这样确保使能定时器时从SYST_LOAD的值计数而不是任意的值。

如果SYST_LOAD为0，在用该值重新加载计时器后，该计时器的当前值将保持为零。该机制可用于独立于定时器使能位禁用该功能。

详细请参考 “*ARM® Cortex™-M4 Technical Reference Manual*” 和 “*ARM® v6-M Architecture Reference Manual*”。

6.2.14 系统定时器控制寄存器映射

R: 只读, **W**: 只写, **R/W**: 读写

寄存器	偏移	R/W	描述	复位值
SYST基地址: **SCS_BA = 0xE000_E000**				
SYST_CTRL	SCS_BA+0x10	R/W	SysTick控制和状态寄存器	0x0000_0000
SYST_LOAD	SCS_BA+0x14	R/W	SysTick重载值寄存器	0xXXXX_XXXX
SYST_VAL	SCS_BA+0x18	R/W	SysTick当前值寄存器	0xXXXX_XXXX

6.2.14.1 系统定时器控制寄存器描述

SysTick控制和状态寄存器 (SYST_CTRL)

寄存器	偏移	R/W	描述	复位值
SYST_CTRL	SCS_BA+0x10	R/W	SysTick控制和状态寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							COUNTFLAG
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved					CLKSRC	TICKINT	ENABLE

位	描述	
[31:17]	Reserved	保留。
[16]	COUNTFLAG	System Tick计数器标志 当定时器自从上次寄存器被读计数到0时，返回1。 当计数由1到0时，COUNTFLAG置位。 读或者写当前值寄存器时，COUNTFLAG会清0。
[15:3]	Reserved	保留。
[2]	CLKSRC	System Tick时钟源选择 0 = 时钟源位是（可选）外部参考时钟。 1 = 内核时钟座位SysTick。
[1]	TICKINT	System Tick中断使能 0 = 计数到0不申请中断，软件可查寻 COUNTFLAG得知是否计数跨0。 1 = 计数到0申请中断，软件写0到当前值寄存器不会申请中断。
[0]	ENABLE	System Tick计数使能 0 = 计数禁止。 1 = 计数器将运行在多次触发的方式。

SysTick重载值寄存器 (SYST_LOAD)

寄存器	偏移	R/W	描述	复位值
SYST_LOAD	SCS_BA+0x14	R/W	SysTick重载值寄存器	0xXXXX_XXXX

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
RELOAD							
15	14	13	12	11	10	9	8
RELOAD							
7	6	5	4	3	2	1	0
RELOAD							

位	描述	
[31:24]	Reserved	保留。
[23:0]	RELOAD	System Tick重载值 当计数值到0时，该值载入到当前值寄存器。

SysTick当前值寄存器 (SYST_VAL)

寄存器	偏移	R/W	描述	复位值
SYST_VAL	SCS_BA+0x18	R/W	SysTick当前值寄存器	0xXXXX_XXXX

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
CURRENT							
15	14	13	12	11	10	9	8
CURRENT							
7	6	5	4	3	2	1	0
CURRENT							

位	描述	
[31:24]	Reserved	保留。
[23:0]	CURRENT	System Tick当前值 当前计数器值。这是计数器在采样时的值。计数器不提供读取-修改-写入保护。寄存器写入清除。任何值的软件写入都会将寄存器清零。

6.2.15 可嵌套中断向量控制器(NVIC)

NVIC与处理器内核接口的紧密结合，能够使中断低延时处理以及后到的中断有效处理。NVIC保持了堆栈，嵌套，中断来使能链尾中断。虽然只能在优先模式下完全访问NVIC，但在用户模式下，如果使能了配置和控制的相关寄存器，也可以产生中断进入挂起状态。任何其它用户模式下访问会导致总线故障。除有特别说明之外，用户可以使用字节，半字，字的方式访问所有的NVIC寄存器。NVIC寄存器都放在SCS（系统控制位置）的里面。NVIC寄存器与调试寄存器都按小端格式读写。

NVIC 支持:

- 可定义中断数量，范围在1-240个中断。
- 可编程优先级 0-16 ，数值越小优先级越高，所以0级是最高中断优先级。
- 支持电平和边沿触发中断。
- 优先级可动态调整。
- 将优先级值分组为组优先级和子优先级字段。
- 末尾连锁中断。
- 支持非屏蔽中断（NMI）。
- 中断唤醒控制器（WIC）支持超低功耗模式。

处理器自动入栈中断如果的状态，中断返回时处理器自动出栈该状态，无指令开销。这提供了低延迟的异常处理。

6.2.15.1 异常模式和中断向量表

表6.2-9异常类型列举了M480系列支持的异常模型。软件可以对这些异常以及所有中断设置16级优先级。用户可配置的最高优先级为“0x00”，最低优先级为“0xF0”（4-LSB始终为0），所有用户可配置中断的默认优先级为“0x00”。注意，优先级“0”被视为系统上的第四个优先级，在三个系统异常“复位”、“NMI”和“硬件故障”之后。

当发生任何中断时，处理器将自动从内存中的向量表中获取中断服务例程（ISR）的起始地址。在系统复位时，向量表固定在地址0x00000000。特权软件可以写入VTOR，将vector表的起始地址重新定位到不同的内存位置，范围从0x00000080到0x3FFFFF80。

向量表包含复位时堆栈指针的初始化值，以及所有异常处理程序的入口点地址。前一节上的向量号定义与异常处理程序项关联的向量表中条目的顺序，如前一节所示。

异常类型	向量号	向量地址	优先级
Reset	1	0x00000004	-3
NMI	2	0x00000008	-2
Hard Fault	3	0x0000000C	-1
Memory Manager Fault	4	0x00000010	可配置
Bus Fault	5	0x00000014	可配置
Usage Fault	6	0x00000018	可配置
Reserved	7 ~ 10		保留

SVCall	11	0x0000002C	可配置
Debug Monitor	12	0x00000030	可配置
Reserved	13		保留
PendSV	14	0x00000038	可配置
SysTick	15	0x0000003C	可配置
Interrupt (IRQ0 ~ IRQ)	16 ~ 111	0x00000000 + (Vector Number)*4	可配置

表 6.2-9 异常类型

向量序号	中断号 (中断寄存器中的位)	中断名	中断描述
0 ~ 15	-	-	系统异常
16	0	BODOUT	BOD 欠压中断
17	1	IRC_INT	IRC 校准中断
18	2	PWRWU_INT	时钟控制和唤醒状态中断
19	3	SRAM_PERR	SRAM 校验错中断
20	4	CLKFAIL	时钟失效中断
21	5	Reserved	保留
22	6	RTC_INT	实时时钟中断
23	7	TAMPER_INT	备份寄存器防篡改中断
24	8	WDT_INT	看门狗定时器中断
25	9	WWDT_INT	窗口看门狗定时器中断
26	10	EINT0	外部中断PA.0, PD.2 或PE.4
27	11	EINT1	外部中断PB.0, PD.3 或 PE.5
28	12	EINT2	外部中断 PC.0
29	13	EINT3	外部中断 PD.0
30	14	EINT4	外部中断PE.0
31	15	EINT5	外部中断PF.0
32	16	GPA_INT	外部中断PA[15:0]
33	17	GPB_INT	外部中断PB[15:0]
34	18	GPC_INT	外部中断PC[15:0]
35	19	GPD_INT	外部中断PD[15:0]
36	20	GPE_INT	外部中断PE[15:0]

37	21	GPF_INT	外部中断PF[15:0]
38	22	QSPI0_INT	QSPI0 中断
39	23	SPI0_INT	SPI0 中断
40	24	BRAKE0_INT	EPWM0刹车中断
41	25	EPWM0_P0_INT	EPWM0_P0中断
42	26	EPWM0_P1_INT	EPWM0_P1中断
43	27	EPWM0_P2_INT	EPWM0_P2中断
44	28	BRAKE1_INT	EPWM1刹车中断
45	29	EPWM1_P0_INT	EPWM1_P0中断
46	30	EPWM1_P1_INT	EPWM1_P1中断
47	31	EPWM1_P2_INT	EPWM1_P2中断
48	32	TMR0_INT	Timer 0中断
49	33	TMR1_INT	Timer 1中断
50	34	TMR2_INT	Timer 2中断
51	35	TMR3_INT	Timer 3中断
52	36	UART0_INT	UART0中断
53	37	UART1_INT	UART1中断
54	38	I^2C0_INT	I^2C0中断
55	39	I^2C1_INT	I^2C1中断
56	40	PDMA_INT	PDMA中断
57	41	DAC_INT	DAC中断
58	42	EADC0_INT0	EADC0中断源0
59	43	EADC0_INT1	EADC0中断源1
60	44	ACMP01_INT	ACMP0和ACMP1中断
61	45	Reserved	保留
62	46	EADC0_INT2	EADC0中断源2
63	47	EADC0_INT3	EADC0中断源3
64	48	UART2_INT	UART2中断
65	49	UART3_INT	UART3中断
66	50	QSPI1_INT	QSPI1中断
67	51	SPI1_INT	SPI1中断
68	52	SPI2_INT	SPI2中断

69	53	USBD_INT	USB设备中断
70	54	USBH_INT	USB主机中断
71	55	USBOTG_INT	USB OTG 中断
72	56	CAN0_INT	CAN0 中断
73	57	CAN1_INT	CAN1 中断
74	58	SC0_INT	智能卡接口0中断
75	59	SC1_INT	智能卡接口1中断
76	60	SC2_INT	智能卡接口2中断
77	61	SC3_INT	智能卡接口3中断
78	62	SPI3_INT	SPI3中断
80	64	SDHOST0_INT	SD接口0中断
81	65	HSUSBD_INT	HSUSBD中断
82	66	EMAC_TX	以太网 MAC 发送中断
83	67	EMAC_RX	以太网 MAC 接收中断
84	68	I2S0_INT	I2S0中断
85	69	Reserved	保留
86	70	OPA0_INT	模拟 OP0 中断
87	71	CRYPTO	Crypto 中断
88	72	GPG_INT	外部中断PG[15:0]
89	73	EINT6	外部中断PG.0
90	74	UART4_INT	UART4中断
91	75	UART5_INT	UART5中断
92	76	USCI0_INT	USCI0中断
93	77	USCI1_INT	USCI1中断
94	78	BPWM0_INT	BPWM0中断
95	79	BPWM1_INT	BPWM1中断
96	80	SPIM_INT	SPI 主机中断
97	81	CCAP_INT	CCAP中断
98	82	I^2C2_INT	I^2C2中断
99	83	Reserved	保留
100	84	QEI0_INT	QEI0中断
101	85	QEI1_INT	QEI1中断

102	86	ECAP0_INT	ECAP0中断
103	87	ECAP1_INT	ECAP1中断
105	88	GPH_INT	外部中断PH[?:0]
105	89	EINT7	外部中断PH.?
106	90	SDHOST1_INT	SD接口1中断
107	91	Reserved	保留
108	92	HSUSBH_INT	HSUSBH中断
109	93	HSOTG_INT	HSOTG中断
			ETMC中断
112	96	Reserved	保留
113	97	Reserved	保留
114	98	Reserved	保留
115	99	Reserved	保留
116	100	Reserved	保留
117	101	TRNG_INT	TRNG中断
118	102	UART6_INT	UART6中断
119	103	UART7_INT	UART7中断
120	104	EADC1_INT0	EADC1中断源0
121	105	EADC1_INT1	EADC1中断源1
122	106	EADC1_INT2	EADC1中断源2
123	107	EADC1_INT3	EADC1中断源3
124	108	CAN2_INT	CAN2中断

表 6.2-10 中断向量表

6.2.15.2 操作描述

通过写相应中断的置位使能或清使能寄存器的相关位，可以使能 NVIC 中断或禁用NVIC中断，这些寄存器通过写1使能和写1清零，读取两者寄存器都返回当前相应中断的使能状态。当中断禁用时，中断声明将使中断挂起，因此中断不被激活，如果在禁用时中断被激活，该中断就保持在激活状态，直到通过复位或异常返回来时清除。清使能位可以阻止新的相应中断被激活。

NVIC中断可以使用互补的寄存器对来挂起/取消挂起以使能/禁用这些中断，这些寄存器分别为设置挂起寄存器与清除挂起寄存器，可以写1使能和写1禁用，读取两者寄存器都返回当前相应中断的状态。寄存器清除挂起在中断响应时的不影响执行状态。

可以通过更新32位寄存器中的8位字段来设置NVIC 中断的优先级（每个寄存器支持4个中断）。

与 NVIC 相关的的通用寄存器都可以在系统控制空间的一块内存区域中访问，下一节将作出描述。

6.2.15.3 NVIC控制寄存器

R:只读, **W:** 只写, **R/W:** 读写

寄存器	偏移	R/W	描述	复位值
NVIC 基地址: **NVIC_BA = 0xE000_E100**				
NVIC_ISER0	NVIC_BA+0x00	R/W	IRQ0 ~ IRQ108设置使能控制寄存器	0x0000_0000
NVIC_ISER1	NVIC_BA+0x04	R/W	IRQ0 ~ IRQ108设置使能控制寄存器	0x0000_0000
NVIC_ISER2	NVIC_BA+0x08	R/W	IRQ0 ~ IRQ108设置使能控制寄存器	0x0000_0000
NVIC_ISER3	NVIC_BA+0x0C	R/W	IRQ0 ~ IRQ108设置使能控制寄存器	0x0000_0000
NVIC_ICER0	NVIC_BA+0x80	R/W	IRQ0 ~ IRQ108清除使能控制寄存器	0x0000_0000
NVIC_ICER1	NVIC_BA+0x84	R/W	IRQ0 ~ IRQ108清除使能控制寄存器r	0x0000_0000
NVIC_ICER2	NVIC_BA+0x88	R/W	IRQ0 ~ IRQ108清除使能控制寄存器	0x0000_0000
NVIC_ICER3	NVIC_BA+0x8C	R/W	IRQ0 ~ IRQ108清除使能控制寄存器	0x0000_0000
NVIC_ISPR0	NVIC_BA+0x100	R/W	IRQ0 ~ IRQ108设置挂起控制寄存器	0x0000_0000
NVIC_ISPR1	NVIC_BA+0x104	R/W	IRQ0 ~ IRQ108设置挂起控制寄存器	0x0000_0000
NVIC_ISPR2	NVIC_BA+0x108	R/W	IRQ0 ~ IRQ108设置挂起控制寄存器	0x0000_0000
NVIC_ISPR3	NVIC_BA+0x10C	R/W	IRQ0 ~ IRQ108设置挂起控制寄存器	0x0000_0000
NVIC_ICPR0	NVIC_BA+0x180	R/W	IRQ0 ~ IRQ108清除挂起控制寄存器	0x0000_0000
NVIC_ICPR1	NVIC_BA+0x184	R/W	IRQ0 ~ IRQ108清除挂起控制寄存器	0x0000_0000
NVIC_ICPR2	NVIC_BA+0x188	R/W	IRQ0 ~ IRQ108清除挂起控制寄存器	0x0000_0000
NVIC_ICPR3	NVIC_BA+0x18C	R/W	IRQ0 ~ IRQ108清除挂起控制寄存器	0x0000_0000
NVIC_IABR0	NVIC_BA+0x200	R/W	IRQ0 ~ IRQ108有效位寄存器	0x0000_0000
NVIC_IABR1	NVIC_BA+0x204	R/W	IRQ0 ~ IRQ108有效位寄存器	0x0000_0000
NVIC_IABR2	NVIC_BA+0x208	R/W	IRQ0 ~ IRQ108有效位寄存器	0x0000_0000
NVIC_IABR3	NVIC_BA+0x20C	R/W	IRQ0 ~ IRQ108有效位寄存器	0x0000_0000
NVIC_IPRn **n=0,1..28**	NVIC_BA+0x300 +0x4*n	R/W	IRQ0 ~ IRQ71优先级控制寄存器	0x0000_0000
STIR	NVIC_BA+0xE00	R/W	软件触发中断寄存器	0x0000_0000

IRQ0 ~ IRQ108设置使能控制寄存器 (NVIC_ISER0)

寄存器	偏移	R/W	描述	复位值
NVIC_ISER0	NVIC_BA+0x00	R/W	IRQ0 ~ IRQ108设置使能控制寄存器	0x0000_0000

31	30	29	28	27	26	25	24
SETENA							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
SETENA							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
SETENA							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
SETENA							

位	描述	
[31:0]	**SETENA**	**中断设置使能位** NVIC_ISER0-NVIC_ISER3寄存器使能中断，并显示那些中断使能。 写操作： 0 = 无效。 1 = 中断使能。 读操作： 0 = 中断禁止。 1 = 中断使能。

IRQ0 ~ IRQ108设置使能控制寄存器 (NVIC_ISER1)

寄存器	偏移	R/W	描述	复位值
NVIC_ISER1	NVIC_BA+0x04	R/W	IRQ0 ~ IRQ108设置使能控制寄存器	0x0000_0000

31	30	29	28	27	26	25	24
SETENA							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
SETENA							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
SETENA							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
SETENA							

位	描述	
[31:0]	**SETENA**	**中断设置使能位** NVIC_ISER0-NVIC_ISER3寄存器使能中断，并显示那些中断使能。 写操作： 0 = 无效。 1 = 中断使能。 读操作： 0 = 中断禁止。 1 = 中断使能。

IRQ0 ~ IRQ108设置使能控制寄存器(NVIC_ISER2)

寄存器	偏移	R/W	描述	复位值
NVIC_ISER2	NVIC_BA+0x08	R/W	IRQ0 ~ IRQ108设置使能控制寄存器	0x0000_0000

31	30	29	28	27	26	25	24
SETENA							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
SETENA							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
SETENA							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
SETENA							

位	描述	
[31:0]	**SETENA**	**中断设置使能位** NVIC_ISER0-NVIC_ISER3寄存器使能中断，并显示那些中断使能。 写操作： 0 = 无效。 1 = 中断使能。 读操作： 0 = 中断禁止。 1 = 中断使能。

IRQ0 ~ IRQ108设置使能控制寄存器(NVIC_ISER3)

寄存器	偏移	R/W	描述	复位值
NVIC_ISER3	NVIC_BA+0x0C	R/W	IRQ0 ~ IRQ108设置使能控制寄存器	0x0000_0000

31	30	29	28	27	26	25	24
SETENA							
23	22	21	20	19	18	17	16
SETENA							
15	14	13	12	11	10	9	8
SETENA							
7	6	5	4	3	2	1	0
SETENA							

位	描述	
[31:0]	SETENA	**中断设置使能位** NVIC_ISER0-NVIC_ISER3寄存器使能中断，并显示那些中断使能。 写操作： 0 = 无效。 1 = 中断使能。 读操作： 0 = 中断禁止。 1 = 中断使能。

IRQ0 ~ IRQ108清除使能控制寄存器 (NVIC_ICER0)

寄存器	偏移	R/W	描述	复位值
NVIC_ICER0	NVIC_BA+0x80	R/W	IRQ0 ~ IRQ108清除使能控制寄存器	0x0000_0000

31	30	29	28	27	26	25	24
CALENA							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
CALENA							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
CALENA							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
CALENA							

位	描述	
[31:0]	**CALENA**	**中断清除使能位** NVIC_ICER0-NVIC_ICER3寄存器使能中断，并显示哪些中断使能。 写操作： 0 = 无效。 1 = 中断禁止。 读操作： 0 = 中断禁止。 1 =中断使能。

IRQ0 ~ IRQ108清除使能控制寄存器(NVIC_ICER1)

寄存器	偏移	R/W	描述	复位值
NVIC_ICER1	NVIC_BA+0x84	R/W	IRQ0 ~ IRQ108清除使能控制寄存器	0x0000_0000

31	30	29	28	27	26	25	24
CALENA							
23	22	21	20	19	18	17	16
CALENA							
15	14	13	12	11	10	9	8
CALENA							
7	6	5	4	3	2	1	0
CALENA							

位	描述	
[31:0]	CALENA	**中断清除使能位** NVIC_ICER0-NVIC_ICER3寄存器使能中断，并显示哪些中断使能。 写操作： 0 = 无效。 1 = 中断禁止。 读操作： 0 = 中断禁止。 1 =中断使能。

IRQ0 ~ IRQ108清除使能控制寄存器(NVIC_ICER2)

寄存器	偏移	R/W	描述	复位值
NVIC_ICER2	NVIC_BA+0x88	R/W	IRQ0 ~ IRQ108清除使能控制寄存器	0x0000_0000

31	30	29	28	27	26	25	24
CALENA							
23	22	21	20	19	18	17	16
CALENA							
15	14	13	12	11	10	9	8
CALENA							
7	6	5	4	3	2	1	0
CALENA							

位	描述	
[31:0]	CALENA	**中断清除使能位** NVIC_ICER0-NVIC_ICER3寄存器使能中断，并显示哪些中断使能。 写操作： 0 = 无效。 1 = 中断禁止。 读操作： 0 = 中断禁止。 1 =中断使能。

IRQ0 ~ IRQ108清除使能控制寄存器(NVIC_ICER3)

寄存器	偏移	R/W	描述	复位值
NVIC_ICER3	NVIC_BA+0x8C	R/W	IRQ0 ~ IRQ108清除使能控制寄存器	0x0000_0000

31	30	29	28	27	26	25	24
CALENA							
23	22	21	20	19	18	17	16
CALENA							
15	14	13	12	11	10	9	8
CALENA							
7	6	5	4	3	2	1	0
CALENA							

位	描述	
[31:0]	CALENA	**中断清除使能位** NVIC_ICER0-NVIC_ICER3寄存器使能中断，并显示哪些中断使能。 写操作： 0 = 无效。 1 = 中断禁止。 读操作： 0 = 中断禁止。 1 =中断使能。

IRQ0 ~ IRQ108设置挂起控制寄存器 (NVIC_ISPR0)

寄存器	偏移	R/W	描述	复位值
NVIC_ISPR0	NVIC_BA+0x100	R/W	IRQ0 ~ IRQ108设置挂起控制寄存器	0x0000_0000

31	30	29	28	27	26	25	24
SETPEND							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
SETPEND							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
SETPEND							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
SETPEND							

位	描述	
[31:0]	**SETPEND**	**中断设置挂起** NVIC_ISPR0-NVIC_ISPR3寄存器强制中断进入挂起状态，并显示那些中断挂起。 写操作： 0 = 无效。 1 = 将中断状态更改为挂起。 读操作： 0 = 中断不处于挂起。 1 = 中断处于挂起。

IRQ0 ~ IRQ108设置挂起控制寄存器(NVIC_ISPR1)

寄存器	偏移	R/W	描述	复位值
NVIC_ISPR1	NVIC_BA+0x104	R/W	IRQ0 ~ IRQ108设置挂起控制寄存器	0x0000_0000

31	30	29	28	27	26	25	24
SETPEND							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
SETPEND							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
SETPEND							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
SETPEND							

位	描述	
[31:0]	**SETPEND**	**中断设置挂起** NVIC_ISPR0-NVIC_ISPR3寄存器强制中断进入挂起状态，并显示那些中断挂起。 写操作： 0 = 无效。 1 = 将中断状态更改为挂起。 读操作： 0 = 中断不处于挂起。 1 = 中断处于挂起。

IRQ0 ~ IRQ108设置挂起控制寄存器(NVIC_ISPR2)

寄存器	偏移	R/W	描述	复位值
NVIC_ISPR2	NVIC_BA+0x108	R/W	IRQ0 ~ IRQ108设置挂起控制寄存器	0x0000_0000

31	30	29	28	27	26	25	24
SETPEND							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
SETPEND							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
SETPEND							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
SETPEND							

位	描述	
[31:0]	**SETPEND**	**中断设置挂起** NVIC_ISPR0-NVIC_ISPR3寄存器强制中断进入挂起状态，并显示那些中断挂起。 写操作： 0 = 无效。 1 = 将中断状态更改为挂起。 读操作： 0 = 中断不处于挂起。 1 = 中断处于挂起。

IRQ0 ~ IRQ108设置挂起控制寄存器(NVIC_ISPR3)

寄存器	偏移	R/W	描述	复位值
NVIC_ISPR3	NVIC_BA+0x10C	R/W	IRQ0 ~ IRQ108设置挂起控制寄存器	0x0000_0000

31	30	29	28	27	26	25	24
SETPEND							
23	22	21	20	19	18	17	16
SETPEND							
15	14	13	12	11	10	9	8
SETPEND							
7	6	5	4	3	2	1	0
SETPEND							

位	描述	
[31:0]	SETPEND	**中断设置挂起** NVIC_ISPR0-NVIC_ISPR3寄存器强制中断进入挂起状态，并显示那些中断挂起。 写操作： 0 = 无效。 1 = 将中断状态更改为挂起。 读操作： 0 = 中断不处于挂起。 1 = 中断处于挂起。

IRQ0 ~ IRQ108清除挂起控制寄存器(NVIC_ICPR0)

寄存器	偏移	R/W	描述	复位值
NVIC_ICPR0	NVIC_BA+0x180	R/W	IRQ0 ~ IRQ108清除挂起控制寄存器	0x0000_0000

31	30	29	28	27	26	25	24
CALPEND							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
CALPEND							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
CALPEND							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
CALPEND							

位	描述	
[31:0]	**CALPEND**	**中断清除挂起位** NVIC_ICPR0-NVIC_ICPR3寄存器从中断中清除挂起状态，并且显示那些中断处于挂起状态。 写操作： 0 = 无效。 1 = 清除中断挂起状态。 读操作： 0 = 中断不处于挂起。 1 = 中断处于挂起。

IRQ0 ~ IRQ108清除挂起控制寄存器(NVIC_ICPR1)

寄存器	偏移	R/W	描述	复位值
NVIC_ICPR1	NVIC_BA+0x184	R/W	IRQ0 ~ IRQ108清除挂起控制寄存器	0x0000_0000

31	30	29	28	27	26	25	24
CALPEND							
23	22	21	20	19	18	17	16
CALPEND							
15	14	13	12	11	10	9	8
CALPEND							
7	6	5	4	3	2	1	0
CALPEND							

位	描述	
[31:0]	CALPEND	**中断清除挂起位** NVIC_ICPR0-NVIC_ICPR3寄存器从中断中清除挂起状态，并且显示那些中断处于挂起状态。 写操作： 0 = 无效。 1 = 清除中断挂起状态。 读操作： 0 = 中断不处于挂起。 1 = 中断处于挂起。

IRQ0 ~ IRQ108清除挂起控制寄存器(NVIC_ICPR2)

寄存器	偏移	R/W	描述	复位值
NVIC_ICPR2	NVIC_BA+0x188	R/W	IRQ0 ~ IRQ108清除挂起控制寄存器	0x0000_0000

31	30	29	28	27	26	25	24
CALPEND							
23	22	21	20	19	18	17	16
CALPEND							
15	14	13	12	11	10	9	8
CALPEND							
7	6	5	4	3	2	1	0
CALPEND							

位	描述	
[31:0]	CALPEND	**中断清除挂起位** NVIC_ICPR0-NVIC_ICPR3寄存器从中断中清除挂起状态，并且显示那些中断处于挂起状态。 写操作： 0 = 无效。 1 = 清除中断挂起状态。 读操作： 0 = 中断不处于挂起。 1 = 中断处于挂起。

IRQ0 ~ IRQ108清除挂起控制寄存器(NVIC_ICPR3)

寄存器	偏移	R/W	描述	复位值
NVIC_ICPR3	NVIC_BA+0x18C	R/W	IRQ0 ~ IRQ108清除挂起控制寄存器	0x0000_0000

31	30	29	28	27	26	25	24
CALPEND							
23	22	21	20	19	18	17	16
CALPEND							
15	14	13	12	11	10	9	8
CALPEND							
7	6	5	4	3	2	1	0
CALPEND							

位	描述	
[31:0]	CALPEND	**中断清除挂起位** NVIC_ICPR0-NVIC_ICPR3寄存器从中断中清除挂起状态，并且显示那些中断处于挂起状态。 写操作： 0 = 无效。 1 = 清除中断挂起状态。 读操作： 0 = 中断不处于挂起。 1 = 中断处于挂起。

IRQ0 ~ IRQ108有效位寄存器 (NVIC_IABR0)

寄存器	偏移	R/W	描述	复位值
NVIC_IABR0	NVIC_BA+0x200	R/W	IRQ0 ~ IRQ108有效位寄存器	0x0000_0000

31	30	29	28	27	26	25	24
ACTIVE							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
ACTIVE							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
ACTIVE							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
ACTIVE							

位	描述	
[31:0]	**ACTIVE**	**中断活动标志位** NVIC_IABR0-NVIC_IABR3寄存器显示那些中断处于活动状态。 0 = 中断未激活。 1 = 中断已激活。

IRQ0 ~ IRQ108有效位寄存器(NVIC_IABR1)

寄存器	偏移	R/W	描述	复位值
NVIC_IABR1	NVIC_BA+0x204	R/W	IRQ0 ~ IRQ108有效位寄存器	0x0000_0000

31	30	29	28	27	26	25	24
ACTIVE							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
ACTIVE							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
ACTIVE							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
ACTIVE							

位	描述	
[31:0]	**ACTIVE**	**中断活动标志位** NVIC_IABR0-NVIC_IABR3寄存器显示那些中断处于活动状态。 0 = 中断未激活。 1 = 中断已激活。

IRQ0 ~ IRQ108有效位寄存器(NVIC_IABR2)

寄存器	偏移	R/W	描述	复位值
NVIC_IABR2	NVIC_BA+0x208	R/W	IRQ0 ~ IRQ108有效位寄存器	0x0000_0000

31	30	29	28	27	26	25	24
ACTIVE							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
ACTIVE							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
ACTIVE							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
ACTIVE							

位	描述	
[31:0]	**ACTIVE**	**中断活动标志位** NVIC_IABR0-NVIC_IABR3寄存器显示那些中断处于活动状态。 0 = 中断未激活。 1 = 中断已激活。

IRQ0 ~ IRQ108有效位寄存器(NVIC_IABR3)

寄存器	偏移	R/W	描述	复位值
NVIC_IABR3	NVIC_BA+0x20C	R/W	IRQ0 ~ IRQ108有效位寄存器	0x0000_0000

31	30	29	28	27	26	25	24
ACTIVE							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
ACTIVE							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
ACTIVE							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
ACTIVE							

位	描述	
[31:0]	**ACTIVE**	**中断活动标志位** NVIC_IABR0-NVIC_IABR3寄存器显示那些中断处于活动状态。 0 = 中断未激活。 1 = 中断已激活。

IRQ0 ~ IRQ71中断优先级寄存器 (NVIC_IPRn)

寄存器	偏移	R/W	描述	复位值
NVIC_IPRn **n=0,1..28**	NVIC_BA+0x300 +0x4*n	R/W	IRQ0 ~ IRQ71优先级控制寄存器	0x0000_0000

31	30	29	28	27	26	25	24
PRI_4n_3				**Reserved**			
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
PRI_4n_2				**Reserved**			
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
PRI_4n_1				**Reserved**			
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
PRI_4n_0				**Reserved**			

位	描述	
[31:28]	**PRI_4n_3**	**IRQ_4n+3的优先级** “0”最高优先级，“15”最低优先级。
[27:24]	**Reserved**	保留。
[23:20]	**PRI_4n_2**	**IRQ_4n+2的优先级** “0”最高优先级，“15”最低优先级。
[19:16]	**Reserved**	Reserved.
[15:12]	**PRI_4n_1**	**IRQ_4n+1的优先级** “0”最高优先级，“15”最低优先级。
[11:8]	**Reserved**	保留。
[7:4]	**PRI_4n_0**	**IRQ_4n+0的优先级** “0”最高优先级，“15”最低优先级。
[3:0]	**Reserved**	保留。

软件触发中断寄存器(STIR)

寄存器	偏移	R/W	描述	复位值
STIR	NVIC_BA+0xE00	R/W	软件触发中断寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							INTID
7	6	5	4	3	2	1	0
INTID							

位	描述	
[31:9]	Reserved	保留。
[8:0]	INTID	中断 ID 写 STIR 软件触发一个中断。 触发的中断ID,范围0~0x63，如写0x03，就是触发IRQ3。

6.2.15.4 NMI控制寄存器

R:只读, **W:** 只写, **R/W:** 读写

寄存器	偏移	R/W	描述	复位值
NMI 基地址: **NMI_BA = 0x4000_0300**				
NMIEN	NMI_BA+0x00	R/W	NMI 源中断使能寄存器	0x0000_0000
NMISTS	NMI_BA+0x04	R	NMI 源中断状态寄存器	0x0000_0000

NMI中断源使能寄存器(NMIEN)

寄存器	偏移	R/W	描述	复位值
NMIEN	NMI_BA+0x00	R/W	NMI中断源使能寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
UART1_INT	UART0_INT	EINT5	EINT4	EINT3	EINT2	EINT1	EINT0
7	6	5	4	3	2	1	0
TAMPER_INT	RTC_INT	Reserved	CLKFAIL	SRAM_PERR	PWRWU_INT	IRC_INT	BODOUT

位	描述	
[31:16]	Reserved	保留。
[15]	UART1_INT	**UART1非屏蔽使能位(写保护位)** 0 =禁止UART1 做为非屏蔽中断源。 1 =使能 UART1做为非屏蔽中断源。 **注:** 此位的写是受保护的，请参考寄存器 SYS_REGLCTL。
[14]	UART0_INT	**UART0 非屏蔽使能位(写保护位)** 0 =禁止 UART0做为非屏蔽中断源。 1 =使能UART0做为非屏蔽中断源。 **注:** 此位的写是受保护的，请参考寄存器 SYS_REGLCTL。
[13]	EINT5	**外中断 PF.0 非屏蔽使能位(写保护位)** 0 =禁止外中断 PF.0 做为非屏蔽中断源. 1 = 使能外中断PF.0 做为非屏蔽中断源 **注:** 此位的写是受保护的，请参考寄存器 SYS_REGLCTL.
[12]	EINT4	**外中断 PE.0 非屏蔽使能位(写保护位)** 0 =禁止外中断PE.0做为非屏蔽中断源。 1 = 使能外中断PE.0做为非屏蔽中断源。 **注:** 此位的写是受保护的，请参考寄存器 SYS_REGLCTL。
[11]	EINT3	**外中断 PD.0 非屏蔽使能位(写保护位)** 0 =禁止外中断 PD.0做为非屏蔽中断源。 1 = 使能外中断 PD.0做为非屏蔽中断源。 **注:** 此位的写是受保护的，请参考寄存器 SYS_REGLCTL。
[10]	EINT2	**外中断 PC.0 非屏蔽使能位(写保护位)** 0 =禁止外中断 PC.0做为非屏蔽中断源。

		1 = 使能外中断 PC.0做为非屏蔽中断源。 **注:** 此位的写是受保护的，请参考寄存器 SYS_REGLCTL。
[9]	**EINT1**	**外中断 PB.0, PD.3 或 PE.5 非屏蔽使能位(写保护位)** 0 =禁止外中断 PB.0, PD.3或 PE.5做为非屏蔽中断源。 1 = 使能外中断PB.0, PD.3 或 PE.5做为非屏蔽中断源。 **注:** 此位的写是受保护的，请参考寄存器 SYS_REGLCTL。
[8]	**EINT0**	**外中断PA.0, PD.2 或 PE.4非屏蔽使能位(写保护位)** 0 =禁止外中断 PA.0, PD.2 或 PE.4做为非屏蔽中断源。 1 = 使能外中断 PA.0, PD.2 或 PE.4做为非屏蔽中断源。 **注:** 此位的写是受保护的，请参考寄存器 SYS_REGLCTL。
[7]	**TAMPER_INT**	**TAMPER_INT非屏蔽中断使能位（写保护）** 0 = 备份寄存器防篡改检测中断非屏蔽中断源禁止。 1 = 备份寄存器防篡改检测中断非屏蔽中断源使能。 **注：** 该位是写保护的。请参考SYS_REGLCTL寄存器。
[6]	**RTC_INT**	**RTC非屏蔽中断使能位（写保护）** 0 = RTC非屏蔽中断禁止。 1 = RTC非屏蔽中断使能。 **注：** 该位是写保护的。请参考SYS_REGLCTL寄存器。
[5]	**Reserved**	保留。
[4]	**CLKFAIL**	**时钟失效非屏蔽中断使能位（写保护）** 0 = 时钟失效检测中断非屏蔽中断禁止。 1 = 时钟失效检测中断非屏蔽中断使能。 **注：** 该位是写保护的。请参考SYS_REGLCTL寄存器。
[3]	**SRAM_PERR**	**SRAM奇偶校验错非屏蔽使能位(写保护位)** 0 =禁止SRAM奇偶校验错做为非屏蔽中断源。 1 = 使能SRAM 奇偶校验错做为非屏蔽中断源。 **注:** 此位的写是受保护的，请参考寄存器 SYS_REGLCTL。
[2]	**PWRWU_INT**	**掉电模式唤醒非屏蔽中断使能位（写保护）** 0 = 掉电模式唤醒非屏蔽中断禁止。 1 = 掉电模式唤醒非屏蔽中断使能。 **注：** 该位是写保护的。请参考SYS_REGLCTL寄存器。
[1]	**IRC_INT**	**IRC校准非屏蔽中断使能位（写保护）** 0 = IRC校准非屏蔽中断禁止。 1 = IRC校准非屏蔽中断使能。 **注：** 该位是写保护的。请参考SYS_REGLCTL寄存器。
[0]	**BODOUT**	**BOD非屏蔽中断使能位（写保护）** 0 = BOD非屏蔽中断禁止。 1 = BOD非屏蔽中断使能。 **注：** 该位是写保护的。请参考SYS_REGLCTL寄存器。

NMI 中断源状态寄存器(NMISTS)

寄存器	偏移	R/W	描述	复位值
NMISTS	NMI_BA+0x04	R	NMI中断源状态寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
UART1_INT	UART0_INT	EINT5	EINT4	EINT3	EINT2	EINT1	EINT0
7	6	5	4	3	2	1	0
TAMPER_INT	RTC_INT	Reserved	CLKFAIL	SRAM_PERR	PWRWU_INT	IRC_INT	BODOUT

位	描述	
[31:16]	**Reserved**	保留。
[15]	**UART1_INT**	**UART1 中断标志（只读）** 0 = UART1中断未发生。 1 = UART1发生中断。
[14]	**UART0_INT**	**UART0 中断标志（只读）** 0 = UART1中断未发生。 1 = UART1发生中断。
[13]	**EINT5**	**外部中断PF.0脚中断标志（只读）** 0 = 外部中断PF.0中断未发生。 1 = 外部中断PF.0发生中断。
[12]	**EINT4**	**外部中断PE.0脚中断标志（只读）** 0 = 外部中断PE.0中断未发生。 1 = 外部中断PE.0发生中断。
[11]	**EINT3**	**外部中断PD.0脚中断标志（只读）** 0 = 外部中断PD.0中断未发生。 1 = 外部中断PD.0发生中断。
[10]	**EINT2**	**外部中断PC.0脚中断标志（只读）** 0 = 外部中断PC.0中断未发生。 1 = 外部中断PC.0发生中断。
[9]	**EINT1**	**外部中断PB.0，PD.3或PE.5脚中断标志（只读）** 0 = 外部中断PB.0，PD.3或PE.5中断未发生。 1 = 外部中断PB.0，PD.3或PE.5发生中断。

[8]	**EINT0**	**外部中断PA.0，PD.2或PE.4脚中断标志（只读）** 0 = 外部中断PA.0，PD.2或PE.4中断未发生。 1 = 外部中断PA.0，PD.2或PE.4发生中断。
[7]	**TAMPER_INT**	**TAMPER_INT 中断标志（只读）** 0 = 备份寄存器防篡改中断未发生。 1 = 备份寄存器防篡改发生中断。
[6]	**RTC_INT**	**RTC 中断标志（只读）** 0 = RTC中断未发生。 1 = RTC发生中断。
[5]	**Reserved**	保留。
[4]	**CLKFAIL**	**时钟失效检测中断标志（只读）** 0 = 时钟失效检测中断未发生。 1 = 时钟失效检测发生中断。
[3]	**SRAM_PERR**	**SRAM 奇偶核验中断标志位(只读)** 0 =未发生SRAM奇偶核验中断。 1 =发生了SRAM 奇偶核验中断。
[2]	**PWRWU_INT**	**掉电模式唤醒中断标志（只读）** 0 = 掉电模式唤醒中断未发生。 1 = 掉电模式唤醒发生中断。
[1]	**IRC_INT**	**IRC TRIM 中断标志（只读）** 0 = HIRC TRIM中断未发生。 1 = HIRC TRIM发生中断。
[0]	**BODOUT**	**BOD 中断标志（只读）** 0 = BOD中断未发生。 1 = BOD发生中断。

6.2.15.5 AHB总线矩阵优先级控制寄存器

R:只读, **W:** 只写, **R/W:** 读写

寄存器	偏移	R/W	描述	复位值
AHB 基地址: **AHB_BA = 0x4000_0400**				
AHBMCTL	0x40000400	R/W	AHB总线矩阵优先级控制寄存器	0x0000_0001

AHB总线矩阵优先级控制寄存器(AHBMCTL)

寄存器	偏移	R/W	描述	复位值
AHBMCTL	0x40000400	R/W	AHB总线矩阵优先级控制寄存器	0x0000_0001

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved							INTACTEN

位	描述	
[31:1]	Reserved	保留。
[0]	INTACTEN	**Cortex®-M4 M4 内核最高优先级使能位(写保护位)** 使能Cortex®-M4内核最高优先级。 0 = 最高优先级轮换方式。 1 = 中断发生时M4 内核有最高优先级使用总线。 **注:**此位的写是受保护的，请参考寄存器 SYS_REGLCTL。

6.2.16 系统控制寄存器

Cortex®-M4状态和运行模式控制由系统控制寄存器管理。包括CPUID，Cortex®-M4中断优先级和Cortex®-M4电源管理可以通过这些系统控制寄存器控制。

详细请参考 “*ARM® Cortex™-M4 Technical Reference Manual*” 和 “*ARM® v6-M Architecture Reference Manual*”。

R:只读, **W:** 只写, **R/W:** 读写

寄存器	偏移	R/W	描述	复位值
SCR 基地址: **SCS_BA = 0xE000_E000**				
ICSR	SCS_BA+0xD04	R/W	中断控制和状态寄存器	0x0000_0000
AIRCR	SCS_BA+0xD0C	R/W	应用中断和复位控制寄存器	0xFA05_0000
SCR	SCS_BA+0xD10	R/W	系统控制寄存器	0x0000_0000
SHPR1	SCS_BA+0xD18	R/W	系统中断优先级寄存器1	0x0000_0000
SHPR2	SCS_BA+0xD1C	R/W	系统中断优先级寄存器2	0x0000_0000
SHPR3	SCS_BA+0xD20	R/W	系统中断优先级寄存器3	0x0000_0000

中断控制状态寄存器 (ICSR)

寄存器	偏移	R/W	描述	复位值
ICSR	SCS_BA+0xD04	R/W	中断控制和状态寄存器	0x0000_0000

31	30	29	28	27	26	25	24
NMIPENDSET	Reserved		PENDSVSET	PENDSVRTC_CAL	PENDSTSET	PENDSTRTC_CAL	Reserved
23	22	21	20	19	18	17	16
ISRPREEMPT	ISRPENDING	Reserved				VECTPENDING	
15	14	13	12	11	10	9	8
VECTPENDING				RETTOBASE	Reserved		
7	6	5	4	3	2	1	0
Reserved	VECTACTIVE						

位	描述	
[31]	**NMIPENDSET**	**NMI中断挂起位** 写操作： 0 = 无效。 1 = 将NMI异常状态更改为挂起。 读操作： 0 = NMI异常状态无挂起。 1 = NMI异常状态挂起。 **注：** NMI中断优先级最高，写1后立即去执行NMI中断代码清0此位。此意味着只有在执行NMI中断代码时，此位写1再读，才能读到1。
[30:29]	**Reserved**	保留。
[28]	**PENDSVSET**	**PendSV中断挂起位** 写： 0 =无效。 1 = 挂起 PendSV 中断。 读： 0 = PendSV 没挂起。 1 = PendSV 已挂起。 **注:** 此位写1是挂起 PendSV的唯一方式。
[27]	**PENDSVRTC_CAL**	**PendSV中断清除位（只写）** 写： 0 =无效。 1 = 清除 PendSV 的挂起。 **注:** 清除PENDSV 位，必须 “写 0 到PENDSVSET 且写 1 到PENDSVRTC_CAL” 在同一时间。

[26]	**PENDSTSET**	**SysTick中断挂起位** 写： 0 =无效。 1 =挂起SysTick中断。 读： 0 = SysTick 没挂起。 1 = SysTick已挂起。
[25]	**PENDSTRTC_CAL**	**SysTick中断挂起清除位** 写： 0 =无效。 1 = 清除SysTick 的挂起状态。 **注:** 该位只写。清除PENDST位，必须 “写 0 到 PENDSTSET 且写 1 到 PENDSTRTC_CAL” 在同一时间。
[24]	**Reserved**	保留。
[23]	**ISRPREEMPT**	**中断抢先位（只读）** 若为1，调试停止又开始执行后，会去执行中断代码。
[22]	**ISRPENDING**	**中断挂起标志，NMI 和 Faults 除外 (只读)** 0 = 无中断挂起。 1 =有中断挂起。
[21:18]	**Reserved**	保留。
[17:12]	**VECTPENDING**	**挂起中断的优先级最高的中断号** 使能挂起中断的优先级最高的中断号。 0 = 无挂起的中断。 非0 = 使能且挂起的最高优先级的中断号。 BASEPRI 和 FAULTMASK 会影响这个结果，但 PRIMASK 不会。
[11]	**RETTOBASE**	**抢先的异常标志** 指示是否有抢先的异常。 0 = 在执行抢先的中断。 1 = 无抢先的中断，或当前中断是唯一活动的中断。
[10:7]	**Reserved**	保留。
[6:0]	**VECTACTIVE**	**当前活动的异常或中断** 0 = 线程模式。 非0 = 目前活动的异常或中断。

应用中断和复位控制寄存器(AIRCR)

寄存器	偏移	R/W	描述	复位值
AIRCR	SCS_BA+0xD0C	R/W	应用中断和复位控制寄存器	0xFA05_0000

31	30	29	28	27	26	25	24
VECTORKEY							
23	22	21	20	19	18	17	16
VECTORKEY							
15	14	13	12	11	10	9	8
ENDIANNESS	Reserved				PRIGROUP		
7	6	5	4	3	2	1	0
Reserved					SYSRESETREQ	VECTCLRACTIVE	VECTRESET

位	描述	
[31:16]	**VECTORKEY**	**访问钥匙** 写此寄存器时这部分必须为 0x05FA, 否则结果不可预知。 如此可防止此寄存器数据被意外改写。
[15]	**ENDIANNESS**	**数据顺序** 0 =小端格式。 1 = 大端格式。
[14:11]	**Reserved**	保留。
[10:8]	**PRIGROUP**	**中断优先级组** 这个值决定了组优先级和子优先级的划分。
[7:3]	**Reserved**	保留。
[2]	**SYSRESETREQ**	**系统复位请求位** 将这个位写入1将产生一个复位信号到芯片，并表明需要复位。 该位只能写且随复位时序自清除。
[1]	**VECTCLRACTIVE**	**异常活动状态清除位** 写1会清除中断的所有活动状态。 此位只写且只能在内核停止时写。 **注:**这是仿真器重新初始化栈区的方法。
[0]	**VECTRESET**	保留。

PRIGROUP	二进制小数点	组优先级位	子优先级位	组优先级数	子优先级数
0b000	bxxxxxxx.y	[7:1]	[0]	128	2
0b001	bxxxxxx.yy	[7:2]	[1:0]	64	4
0b010	bxxxxx.yyy	[7:3]	[2:0]	32	8
0b011	bxxxx.yyyy	[7:4]	[3;0]	16	16
0b100	bxxx.yyyyy	[7:5]	[4:0]	8	32
0b101	bxx.yyyyyy	[7:6]	[5:0]	4	64
0b110	bx.yyyyyyy	[7]	[6:0]	2	128
0b111	b.yyyyyyyy	None	[7:0]	1	256

表 6.2-11 优先级分组

系统控制寄存器(SCR)

寄存器	偏移	R/W	描述	复位值
SCR	SCS_BA+0xD10	R/W	系统控制寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved			SEVONPEND	Reserved	SLEEPDEEP	SLEEPONEXIT	Reserved

位	描述	
[31:5]	Reserved	保留。
[4]	SEVONPEND	**挂起事件的唤醒功能选择** 0 = 仅使能的中断或事件唤醒休眠中的CPU。 1 =所有的中断或事件都唤醒CPU，包括未使能的。 如果一个中断或事件被挂起，可以唤醒 WFE状态的处理器。若处理器未在等事件，事件会被注册并影响下次的WFE。 处理器可以被外部事件或执行指令SEV 唤醒。
[3]	Reserved	保留。
[2]	SLEEPDEEP	**休眠模式选择**. 0 = 休眠。 1 =深度休眠。
[1]	SLEEPONEXIT	**返回立即休眠使能位** 该位显示从处理程序模式返回到线程模式时是否进入睡眠。 0 = 返回线程模式时不睡眠。 1 = 从中断返回到线程模式时进入睡眠或深度睡眠。 设置此位使能一个中断处理程序来防止返回到一个空的主程序。
[0]	Reserved	保留。

系统中断优先级寄存器1 (SHPR1)

寄存器	偏移	R/W	描述	复位值
SHPR1	SCS_BA+0xD18	R/W	系统中断优先级寄存器1	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
PRI_6				Reserved			
15	14	13	12	11	10	9	8
PRI_5				Reserved			
7	6	5	4	3	2	1	0
PRI_4				Reserved			

位	描述	
[31:24]	**Reserved**	保留。
[23:20]	**PRI_6**	系统中断6的优先级, 用法错中断。
[20:16]	**Reserved**	保留。
[15:12]	**PRI_5**	系统中断5的优先级, 总线错中断。
[11:8]	**Reserved**	保留。
[7:4]	**PRI_4**	系统中断4的优先级, 内存错中断。
[3:0]	**Reserved**	保留。

系统中断优先级寄存器2 (SHPR2)

寄存器	偏移	R/W	描述	复位值
SHPR2	SCS_BA+0xD1C	R/W	系统中断优先级寄存器2	0x0000_0000

31	30	29	28	27	26	25	24
PRI_11				Reserved			
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved							

位	描述	
[31:28]	PRI_11	**系统中断11 – SVCall的优先级** “0” 最高， “3” 最低。
[27:0]	Reserved	保留。

系统中断优先级寄存器3 (SHPR3)

寄存器	偏移	R/W	描述	复位值
SHPR3	SCS_BA+0xD20	R/W	系统中断优先级寄存器3	0x0000_0000

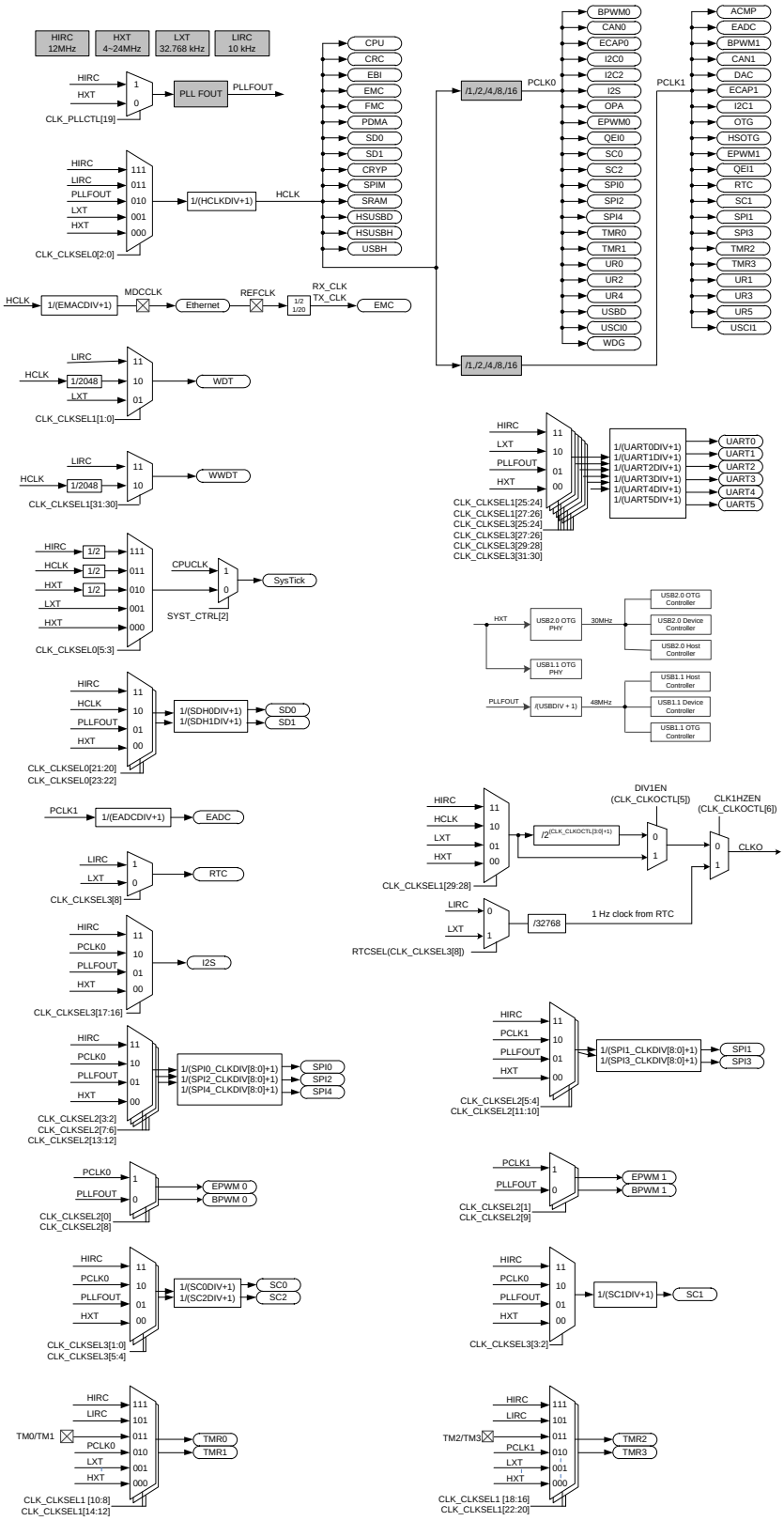
31	30	29	28	27	26	25	24
PRI_15				Reserved			
23	22	21	20	19	18	17	16
PRI_14				Reserved			
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved							

位	描述	
[31:28]	**PRI_15**	**系统中断 15 – SysTick系统定时器的优先级** “0” 最高， “3” 最低。
[27:24]	**Reserved**	保留。
[23:20]	**PRI_14**	**系统中断14 – PendSV的中断优先级** “0” 最高， “3” 最低。
[19:0]	**Reserved**	保留。

6.3 时钟控制器

6.3.1 概述

时钟控制器为整个芯片提供时钟源，包括系统时钟和所有外围设备时钟。该控制器还通过单独时钟的开或关，时钟源选择和分频器来进行功耗控制。在CPU使能低功耗PDEN(CLK_PWRCTL[7])位和Cortex®-M4内核执行WFI指令后，芯片进入低功耗模式。直到唤醒中断发生，芯片才会退出低功耗模式。在低功耗模式下，时钟控制器会关闭外部4~24MHz高速晶振和内部12MHz高速RC振荡器，以降低整个系统功耗。图6.3-1时钟发生器全局示意图(M48xID/M48xGA)所示各模块时钟发生器和时钟源的简图。



注意：对于高速USB 2.0 应用，HXT必须是12 MHz。

图 6.3-1 时钟发生器全局示意图(M48xID/M48xGA)

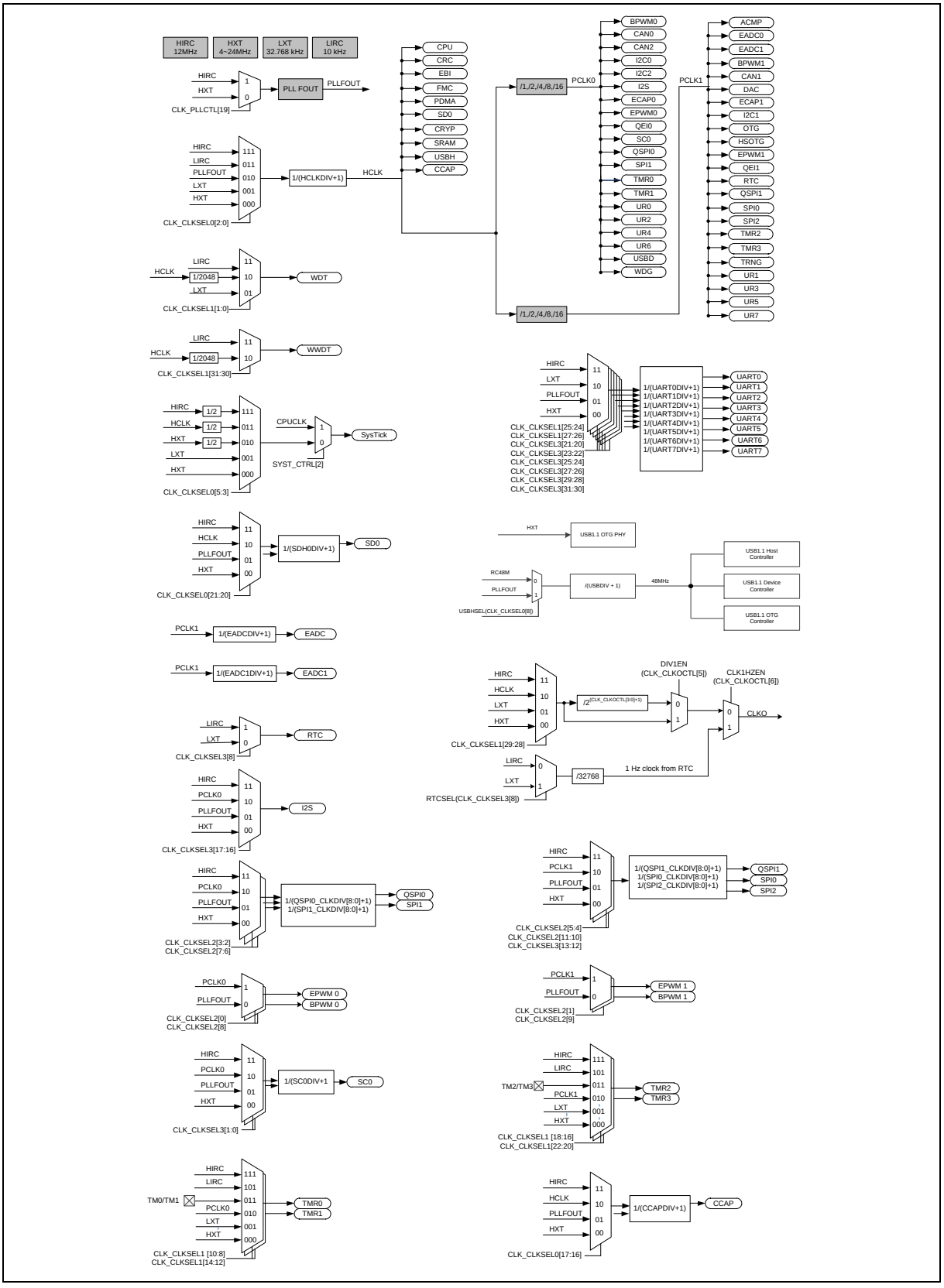

HIRC 12MHz
HXT 4~24MHz
LXT 32.768 kHz
LIRC 10 kHz
HIRC
HXT
CLK_PLLCTL[19]
PLL FOUT
PLLFOUT
CLK_CLKSEL0[2:0]
1/(HCLKDIV+1)
HCLK
CPU
CRC
EBI
FMC
PDMA
SD0
CRYP
SRAM
USBH
CCAP
/1,/2,/4,/8,/16
PCLK0
PCLK1
BPWM0
CAN0
CAN2
I2C0
I2C2
I2S
ECAP0
EPWM0
QEI0
SC0
QSPI0
SPI1
TMR0
TMR1
UR0
UR2
UR4
UR6
USBD
WDG
ACMP
EADC0
EADC1
BPWM1
CAN1
DAC
ECAP1
I2C1
OTG
HSOTG
EPWM1
QEI1
RTC
QSPI1
SPI0
SPI2
TMR2
TMR3
TRNG
UR1
UR3
UR5
UR7
1/2048
WDT
CLK_CLKSEL1[1:0]
WWDT
CLK_CLKSEL1[31:30]
CPUCLK
SysTick
SYST_CTRL[2]
CLK_CLKSEL0[5:3]
1/(SDH0DIV+1)
SD0
CLK_CLKSEL0[21:20]
1/(UART0DIV+1)
1/(UART1DIV+1)
1/(UART2DIV+1)
1/(UART3DIV+1)
1/(UART4DIV+1)
1/(UART5DIV+1)
1/(UART6DIV+1)
1/(UART7DIV+1)
UART0
UART1
UART2
UART3
UART4
UART5
UART6
UART7
CLK_CLKSEL1[25:24]
CLK_CLKSEL1[27:26]
CLK_CLKSEL3[21:20]
CLK_CLKSEL3[23:22]
CLK_CLKSEL3[25:24]
CLK_CLKSEL3[27:26]
CLK_CLKSEL3[29:28]
CLK_CLKSEL3[31:30]
USB1.1 OTG PHY
RC48M
PLLFOUT
/(USBDIV + 1)
48MHz
USBHSEL(CLK_CLKSEL0[8])
USB1.1 Host Controller
USB1.1 Device Controller
USB1.1 OTG Controller
1/(EADCDIV+1)
EADC
1/(EADC1DIV+1)
EADC1
DIV1EN (CLK_CLKOCTL[5])
CLK1HZEN (CLK_CLKOCTL[6])
CLKO
CLK_CLKSEL1[29:28]
/32768
1 Hz clock from RTC
RTCSEL(CLK_CLKSEL3[8])
RTC
CLK_CLKSEL3[8]
I2S
CLK_CLKSEL3[17:16]
1/(QSPI0_CLKDIV[8:0]+1)
1/(SPI1_CLKDIV[8:0]+1)
QSPI0
SPI1
CLK_CLKSEL2[3:2]
CLK_CLKSEL2[7:6]
1/(QSPI1_CLKDIV[8:0]+1)
1/(SPI0_CLKDIV[8:0]+1)
1/(SPI2_CLKDIV[8:0]+1)
QSPI1
SPI0
SPI2
CLK_CLKSEL2[5:4]
CLK_CLKSEL2[11:10]
CLK_CLKSEL3[13:12]
EPWM 0
BPWM 0
CLK_CLKSEL2[0]
CLK_CLKSEL2[8]
EPWM 1
BPWM 1
CLK_CLKSEL2[1]
CLK_CLKSEL2[9]
1/(SC0DIV+1
SC0
CLK_CLKSEL3[1:0]
TM2/TM3
TMR2
TMR3
CLK_CLKSEL1 [18:16]
CLK_CLKSEL1[22:20]
TM0/TM1
TMR0
TMR1
CLK_CLKSEL1 [10:8]
CLK_CLKSEL1[14:12]
1/(CCAPDIV+1)
CCAP
CLK_CLKSEL0[17:16]

图 6.3-2 时钟发生器全局示意图(M48xGC/M48xE8)

6.3.2 时钟发生器

时钟发生器由如下5个时钟源组成：

- 外部32.768 kHz低速晶振（LXT）
- 外部4~24 MHz高速晶振(HXT)
- 可编程的PLL输出时钟频率(PLLFOUT)，PLL 由外部 4~24 MHz 晶振或内部 12 MHz 振荡器提供时钟源
- 内部12 MHz高速振荡器(HIRC)
- 内部10 kHz低速RC振荡器(LIRC)

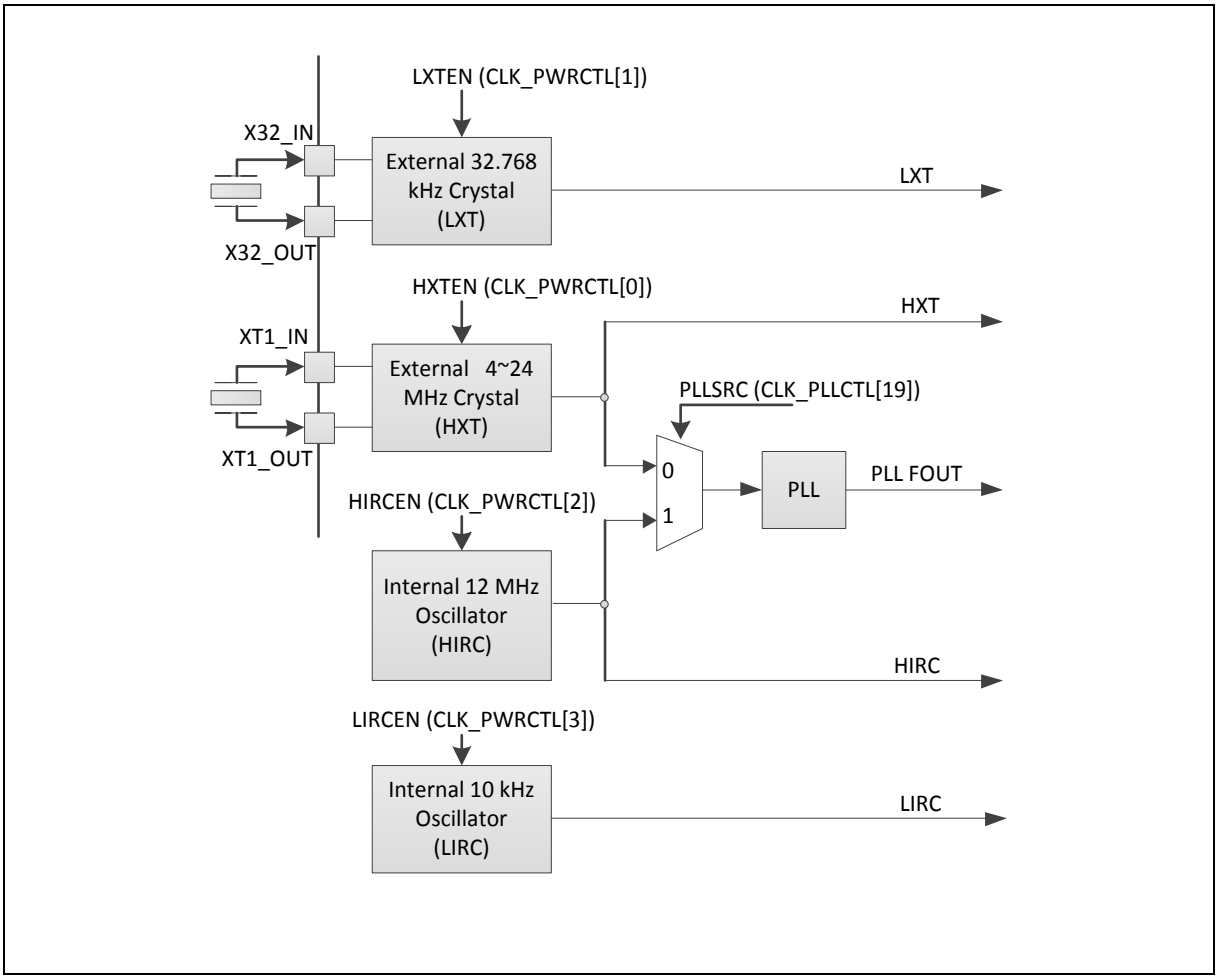


图 6.3-3 时钟发生器框图

6.3.3 系统时钟和SysTick 时钟

系统时钟有 5 个可选时钟源，由时钟发生器产生。时钟源切换取决于寄存器HCLKSEL (CLK_CLKSEL0[2:0])。其框图如下图6.3-4所示。

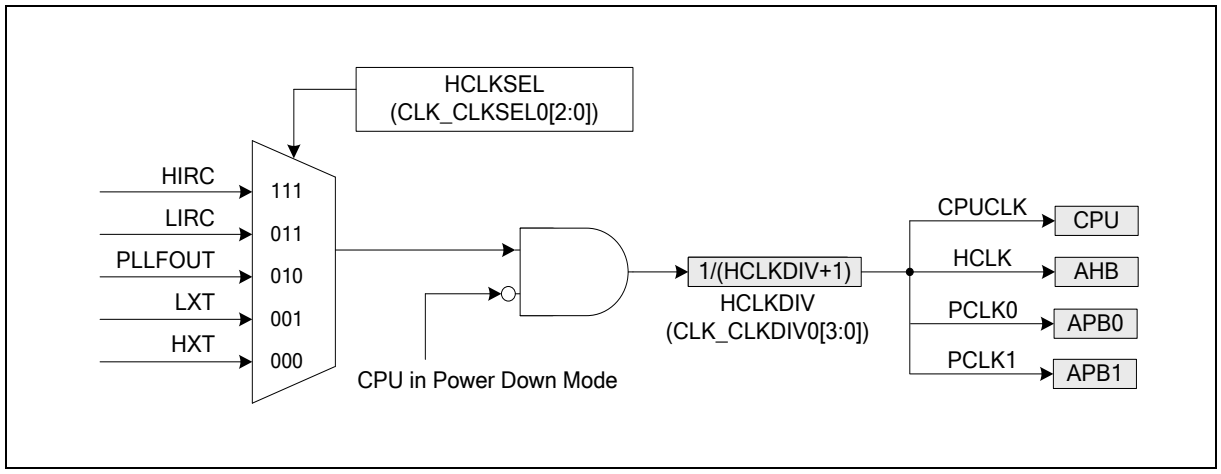


图 6.3-4 系统时钟框图

有两个时钟失败检测器来观察HXT和LXT时钟源的状况，而且都有独立的使能和中断控制设置。当HXT检测器使能时，HIRC时钟也自动使能。当LXT检测器使能时，LIRC时钟也自动使能。

当HXT时钟检测器使能，如果检测到HXT时钟停止且满足下面条件：系统的时钟源来自HXT或者系统时钟源来自PLL(PLL的输入时钟为HXT)，系统时钟将自动切换到HIRC。如果HXT时钟停止条件检测到，HXTFIF (CLK_CLKDSTS[0])将被设置为1，此时如果HXTFIE (CLK_CLKDCTL[5])有置位，芯片将进入中断。用户可以试着去恢复HXT，通过禁止HXT和重新使能HXT来确认时钟稳定标志位是否已设置为1。如果HXT时钟稳定标志位有设置为1，这就意味着HXT在重新使能后已恢复震荡，此时用户可以重新把系统时钟切换到HXT了。

图6.3-5展示了HXT时钟停止检测和系统时钟切换到HIRC的过程。

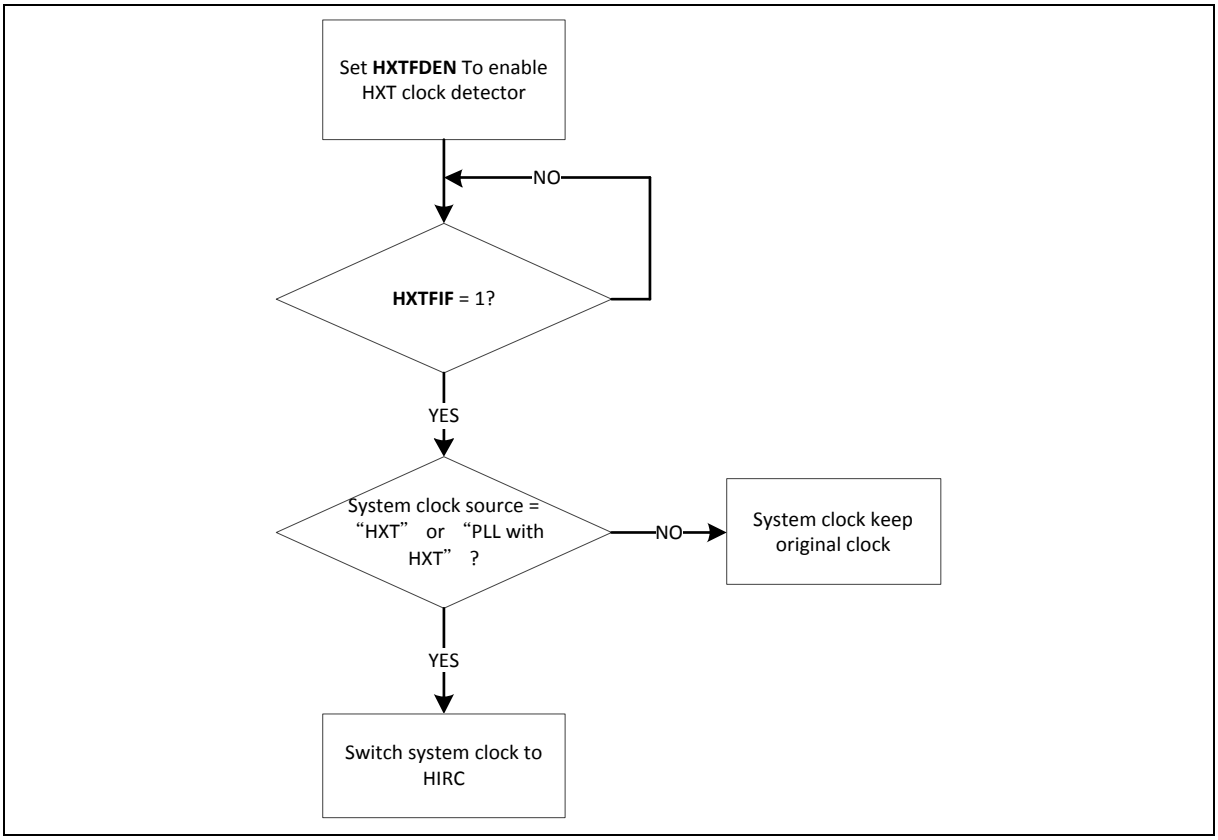


图 6.3-5HXT 停止保护过程

Cortex®-M4内核的SysTick 时钟源可以选择CPU时钟或外部时钟(SYST_CTRL[2]) 。如果使用外部时钟，SysTick时钟(STCLK)有5 个可选时钟源。时钟源切换取决于寄存器STCLKSEL (CLK_CLKSEL0[5:3])。其框图如图6.3-6所示。

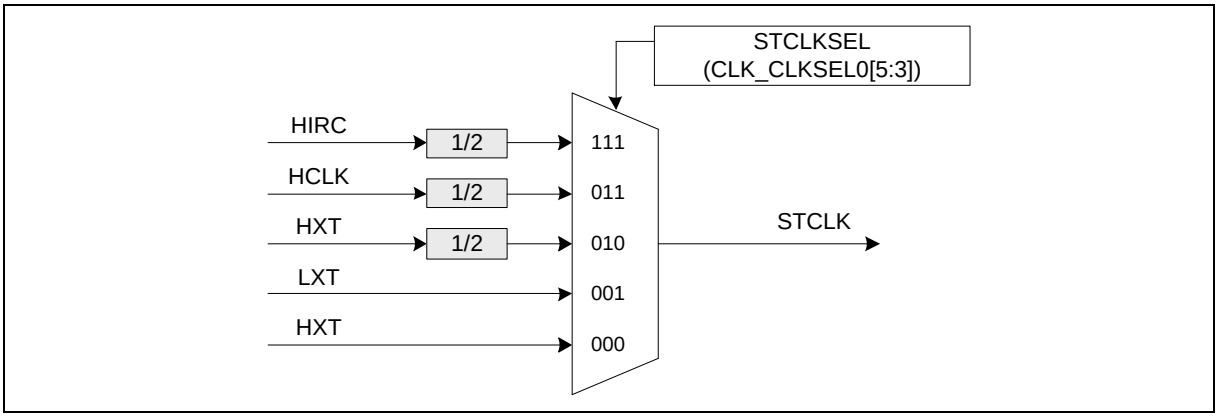


图 6.3-6 时钟控制框图

6.3.4 外设时钟

外设时钟可以有不同的时钟源做切换设置。主要取决于不同的外设。请参看寄存器CLK_CLKSEL1 、

CLK_CLKSEL2 和CLK_CLKSEL3的描述。

6.3.5 掉电模式时钟

当芯片进入掉电模式，系统时钟和一些时钟源以及一些外设时钟将被关闭。也有一些时钟源与外设时钟仍在工作。

如下时钟仍在工作：

- 时钟发生器
  - 10 kHz内部低速RC振荡器
  - 32.768 kHz外部低速晶振时钟
- 外设时钟 (当模块的时钟源来自32.768 kHz外部低速晶振时钟或10 kHz内部低速RC振荡器)

6.3.6 时钟输出

该设备带有一个2的若干次幂的频率分频器，该分频器由16个链式的二分频器组成的移位寄存器构成。其中哪一级的值被输出，由一个16选1的多路转换器选择，该多路转换器接到CLKO管脚上。因此共有16种的时钟分频选择，分频范围从$F_{in}/2^1$到 $F_{in}/2^{16}$，此处F_{in}是到时钟分频器的时钟输入频率。

输出公式：$\mathbf{F_{out} = F_{in}/2^{(N+1)}}$，其中$F_{in}$为输入时钟频率，$F_{out}$为时钟分频器输出频率，N 为FREQSEL (CLK_CLKOCTL[3:0])中的4位值。

往CLKOEN (CLK_CLKOCTL[4])写1，分级计数器开始计数。往CLKOEN (CLK_CLKOCTL[4])写0，分级计数器持续计数，直到分频时钟达到低电平并会保持在低电平状态。

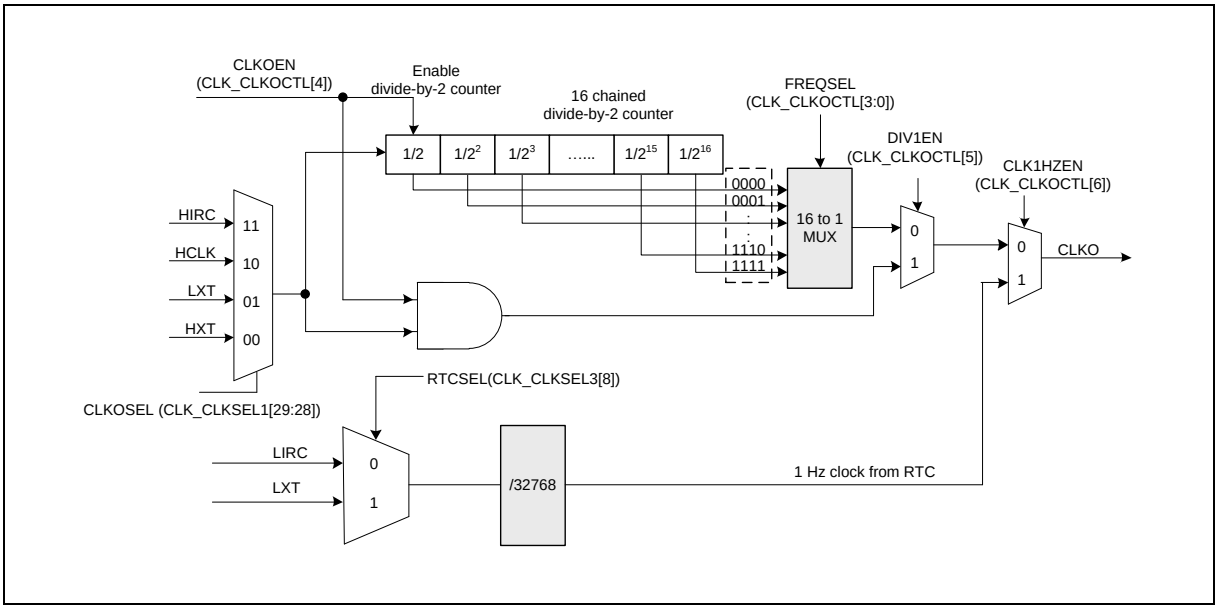


图 6.3-7 时钟输出框图

6.3.7 USB时钟源

USB 1.0 和 2.0系统的时钟源来自USB2.0 PHY时钟或可编程PLL的输出。产生的时钟如图6.3-8所示。

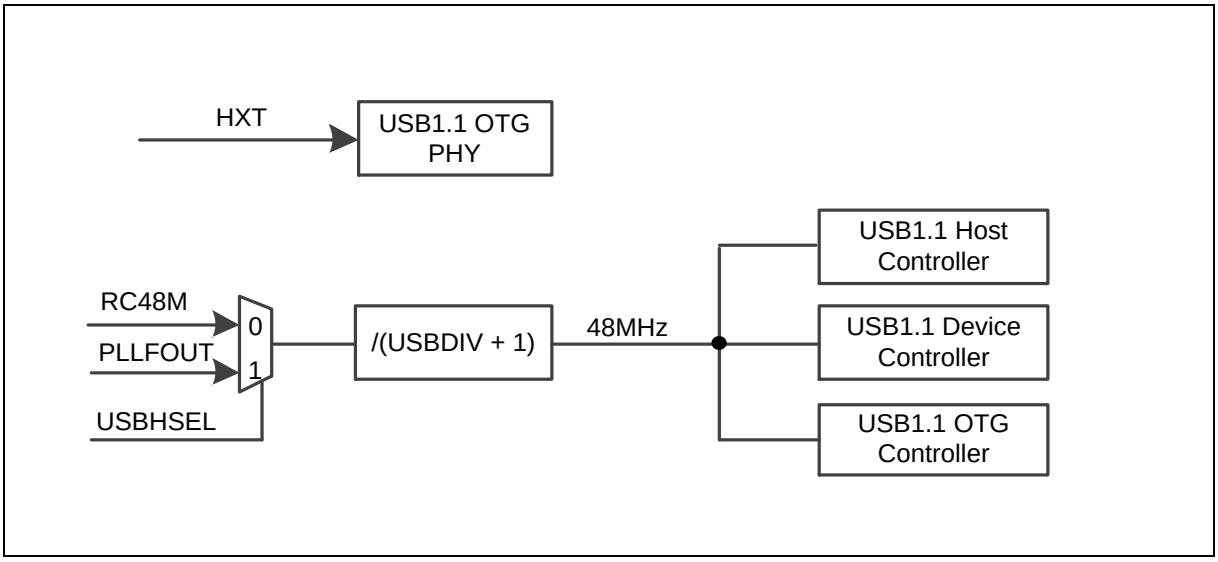


图 6.3-8USB 时钟源

6.3.8 寄存器映射

R:只读, **W:**只写, **R/W:**读写

寄存器	偏移	R/W	描述	复位值
CLK基地址: **CLK_BA = 0x4000_0200**				
CLK_PWRCTL	CLK_BA+0x00	R/W	系统掉电控制寄存器	0x0000_XX1X
CLK_AHBCLK	CLK_BA+0x04	R/W	AHB 设备时钟使能控制寄存器	0x0000_8004
CLK_APBCLK0	CLK_BA+0x08	R/W	APB 设备时钟使能控制寄存器0	0x0000_0001
CLK_APBCLK1	CLK_BA+0x0C	R/W	APB 设备时钟使能控制寄存器1	0x0000_0000
CLK_CLKSEL0	CLK_BA+0x10	R/W	时钟源选择控制寄存器0	0x00F3_013X
CLK_CLKSEL1	CLK_BA+0x14	R/W	时钟源选择控制寄存器1	0xBF77_7703
CLK_CLKSEL2	CLK_BA+0x18	R/W	时钟源选择控制寄存器2	0x0000_03AB
CLK_CLKSEL3	CLK_BA+0x1C	R/W	时钟源选择控制寄存器3	0xFF00_003F
CLK_CLKDIV0	CLK_BA+0x20	R/W	时钟分频数目寄存器0	0x0000_0000
CLK_CLKDIV1	CLK_BA+0x24	R/W	时钟分频数目寄存器1	0x0000_0000
CLK_CLKDIV2	CLK_BA+0x28	R/W	时钟分频数目寄存器2	0x0000_0000
CLK_CLKDIV3	CLK_BA+0x2C	R/W	时钟分频数目寄存器3	0x0000_0000
CLK_CLKDIV4	CLK_BA+0x30	R/W	时钟分频数目寄存器4	0x0000_0000
CLK_PCLKDIV	CLK_BA+0x34	R/W	APB时钟分频器寄存器	0x0000_0000
CLK_PLLCTL	CLK_BA+0x40	R/W	PLL控制寄存器	0x0005_C02E
CLK_STATUS	CLK_BA+0x50	R	时钟状态监测寄存器	0x0000_00XX
CLK_CLKOCTL	CLK_BA+0x60	R/W	时钟输出控制寄存器	0x0000_0000
CLK_CLKDCTL	CLK_BA+0x70	R/W	时钟失败检测器控制寄存器	0x0000_0000
CLK_CLKDSTS	CLK_BA+0x74	R/W	时钟失败检测器状态寄存器	0x0000_0000
CLK_CDUPB	CLK_BA+0x78	R/W	时钟频率检测器上边界寄存器	0x0000_0000
CLK_CDLOWB	CLK_BA+0x7c	R/W	时钟频率检测器下边界寄存器	0x0000_0000
CLK_PMUCTL	CLK_BA+0x90	R/W	电源管理控制寄存器	0x0000_0080
CLK_PMUSTS	CLK_BA+0x94	R/W	电源管理状态寄存器	0x0000_0000
CLK_LDOCTL	CLK_BA+0x98	R/W	LDO控制寄存器	0x0000_0000
CLK_SWKDBCTL	CLK_BA+0x9C	R/W	待机掉电唤醒消抖控制寄存器	0x0000_0000

CLK_PASWKCTL	CLK_BA+0xA0	R/W	GPA待机掉电唤醒控制寄存器	0x0000_0000
CLK_PBSWKCTL	CLK_BA+0xA4	R/W	GPB待机掉电唤醒控制寄存器	0x0000_0000
CLK_PCSWKCTL	CLK_BA+0xA8	R/W	GPC待机掉电唤醒控制寄存器	0x0000_0000
CLK_PDSWKCTL	CLK_BA+0xAC	R/W	GPD待机掉电唤醒控制寄存器	0x0000_0000
CLK_IOPDCTL	CLK_BA+0xB0	R/W	GPIO待机掉电控制寄存器	0x0000_0000

6.3.9 寄存器描述

系统掉电控制寄存器(CLK_PWRCTL)

寄存器	偏移	R/W	描述	复位值
CLK_PWRCTL	CLK_BA+0x00	R/W	系统掉电控制寄存器	0x0000_XX1X

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved					HIRC48MEN	HIRCSTBS	
15	14	13	12	11	10	9	8
Reserved		HXTTBEN	HXTSELTYP	HXTGAIN		Reserved	
7	6	5	4	3	2	1	0
PDEN	PDWKIF	PDWKIEN	PDWKDLY	LIRCEN	HIRCEN	LXTEN	HXTEN

位	描述	
[31:19]	**Reserved**	保留。
[18]	**HIRC48MEN**	**HIRC48M使能位（保护）** 0 = 48 MHz内部高速RC振荡器（HIRC）禁用。 1 = 48 MHz内部高速RC振荡器（HIRC）使能。 **注:** 该位写保护，具体参看SYS_REGLCTL 寄存器。
[17:16]	**HIRCSTBS**	**HIRC 稳定判断周期选择(写保护)** 该位写保护，操作前需要进行解锁操作。 00 = HIRC 稳定周期= 64个时钟。 01 = HIRC 稳定周期= 24个时钟。 其它：保留。 **注:** 该位写保护，具体参看SYS_REGLCTL 寄存器。
[15:14]	**Reserved**	保留。
[13]	**HXTTBEN**	**HXT 晶振 TURBO 模式 (写保护)** 该位写保护，操作前需要进行解锁操作。 0 = HXT 晶振 TURBO 模式禁止。 1 = HXT 晶振 TURBO 模式使能。
[12]	**HXTSELTYP**	**HXT 晶振类别选择位(写保护)** 该位写保护，操作前需要进行解锁操作。 0 =选择 INV类型。 1 =选择GM.类型。 **注:** 该位写保护，参看SYS_REGLCTL寄存器。

[11:10]	**HXTGAIN**	**HXT增益控制位（写保护）** 该位写保护，操作前需要进行解锁操作。 增益控制用来放大晶振的增益，来确保晶振工作正常。增益控制使能时比增益关闭时，晶振需消耗更多的能量。 00 = HXT频率低于8 MHz。 01 = HXT频率在8 MHz 到 12 MHz之间。 10 = HXT频率在12 MHz 到 16 MHz之间。 11 = HXT频率高于16 MHz。 **注:** 该位写保护，参看SYS_REGLCTL寄存器。
[9:8]	**Reserved**	保留。
[7]	**PDEN**	**系统掉电模式使能位(写保护)** 当该位被置1，掉电模式使能，芯片保持运行状态直到CPU睡眠模式也激活，然后芯片进入掉电模式。 当芯片从掉电模式中被唤醒，该位自动清零。用户需要重新设置该位才能进入下一次的掉电模式。 在掉电模式下，HXT 和 HIRC将被禁用，但是LXT 和 LIRC不受掉电模式的控制。 在掉电模式下，PLL 与系统时钟将被禁用，忽略时钟源选择。如果外设时钟源为LXT 或LIRC，则外设的时钟不受掉电模式的控制。 0 = 在CPU 休眠命令 WFI后，芯片不会进入掉电模式。 1 = 在CPU 休眠命令 WFI后，芯片进入掉电模式。 **注:** 该位写保护，参看SYS_REGLCTL寄存器。
[6]	**PDWKIF**	**芯片掉电模式唤醒中断状态** 该位由掉电唤醒事件置位，表示从掉电模式恢复过来。 任何唤醒源发生，该标志置位。请参考电源模式和唤醒源章节。 **注1：** 该位写 1清零。 **注2：** 只有在PDWKIEN (CLK_PWRCTL[5])置1后，该位才工作。
[5]	**PDWKIEN**	**掉电模式唤醒中断使能位（写保护）** 0 = 禁用掉电模式唤醒中断。 1 = 使能掉电模式唤醒中断。 **注1:**当PDWKIF 和 PDWKIEN都为1时，该中断将产生。 **注2:** 该位写保护，参看SYS_REGLCTL寄存器。
[4]	**PDWKDLY**	**使能唤醒延时计数器（写保护）** 当芯片从掉电模式中被唤醒时，时钟控制将会延迟一定的时钟周期以等待系统时钟稳定。 当芯片工作在外部 4~24 MHz高速晶振时，延迟时钟周期为 4096 个时钟周期；当芯片工作在内部 12 MHz 高速振荡器时，延迟时钟周期为 256 个时钟周期。 0 = 禁用时钟周期的延迟。 1 = 使能时钟周期的延迟。 **注:** 该位写保护，参看SYS_REGLCTL寄存器。
[3]	**LIRCEN**	**LIRC使能位（写保护）** 0 = 禁用10 kHz 内部低速 RC 振荡器(LIRC)。 1 = 使能10 kHz 内部低速 RC 振荡器(LIRC)。 **注:** 该位写保护，参看SYS_REGLCTL寄存器。

[2]	**HIRCEN**	**HIRC使能位 (写保护)** 0 = 禁用12 MHz 内部高速RC振荡器(HIRC)。 1 = 使能12 MHz 内部高速RC振荡器(HIRC)。 **注:** 该位写保护，参看SYS_REGLCTL寄存器。
[1]	**LXTEN**	**LXT 使能位（写保护）** 0 = 禁用32.768 kHz 外部低速晶振(LXT)。 1 = 使能32.768 kHz 外部低速晶振(LXT)。 **注:** 1.该位写保护，参看SYS_REGLCTL寄存器。 2.当LXT使能，GPIO必须设为输入模式。
[0]	**HXTEN**	**HXT 使能位（写保护）** 该位的默认值由 flash 控制器的用户配置寄存器CONFIG0 [26:24] 设置。当默认的时钟源为HXT时，该位自动置1。 0 = 禁用4 ~ 24 MHz外部高速晶振(HXT)。 1 = 使能4~24 MHz外部高速晶振(HXT)。 **注:** 1.该位写保护，参看SYS_REGLCTL寄存器。 2.当HXT使能，GPIO必须设为输入模式。

AHB设备时钟使能控制寄存器 (CLK_AHBCLK)

该寄存器的各位用于使能/禁用系统时钟，AHB总线设备时钟。

寄存器	偏移	R/W	描述	复位值
CLK_AHBCLK	CLK_BA+0x04	R/W	AHB设备时钟使能控制寄存器	0x0000_8004

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved						SDH1CKEN	USBHCKEN
15	14	13	12	11	10	9	8
FMCIDLE	SPIMCKEN	Reserved	CRPTCKEN	Reserved	HSUSBDCKEN	SENCKEN	CCAPCKEN
7	6	5	4	3	2	1	0
CRCCKEN	SDH0CKEN	EMACCKEN	Reserved	EBICKEN	ISPCKEN	PDMACKEN	Reserved

位	描述	
[31:18]	**Reserved**	保留。
[17]	**SDH1CKEN**	**SD1 控制器时钟使能位** 0 = SD1 引擎时钟禁止。 1 = SD1 引擎时钟使能。
[16]	**USBHCKEN**	**USB HOST 控制器时钟使能位** 0 = USB HOST 外设时钟禁止。 1 = USB HOST外设时钟使能。
[15]	**FMCIDLE**	**空闲模式时，Flash 内存控制器时钟使能位** 0 = 芯片工作于空闲模式时，禁止FMC 外设时钟。 1 = 芯片工作于空闲模式时，使能FMC 外设时钟。
[14]	**SPIMCKEN**	**SPIM 控制器时钟使能位** 0 = SPIM 控制器时钟禁止。 1 = SPIM 控制器时钟使能。
[13]	**Reserved**	保留。
[12]	**CRPTCKEN**	**加解密算法加速器时钟使能位** 0 = 加解密加速器时钟禁止。 1 = 加解密加速器时钟使能。
[11]	**Reserved**	保留。
[10]	**HSUSBDCKEN**	**HSUSB Device 时钟使能位** 0 = HSUSB device 控制器时钟禁止。

		1 = HSUSB device 控制器时钟使能。
[9]	**SENCKEN**	**传感器时钟使能控制** 0 = CCAP 传感器时钟禁止。 1 = CCAP 传感器时钟使能。
[8]	**CCAPCKEN**	**相机捕获接口控制器时钟使能控制** 0 = CCAP 控制器的时钟禁止。 1 = CCAP 控制器的时钟使能。
[7]	**CRCCKEN**	**CRC产生控制器时钟使能位** 0 = CRC 外设时钟禁止。 1 = CRC外设时钟使能。
[6]	**SDH0CKEN**	**SD0 控制器时钟使能位** 0 = SD0 引擎时钟禁止。 1 = SD0 引擎时钟使能。
[5]	**EMACCKEN**	**以太网控制器时钟使能位** 0 = 以太网控制器引擎时钟禁止。 1 = 以太网控制器引擎时钟使能。
[4]	**Reserved**	保留。
[3]	**EBICKEN**	**EBI 控制器时钟使能位** 0 = EBI 外设时钟禁止。 1 = EBI 外设时钟使能。
[2]	**ISPCKEN**	**Flash ISP 控制器时钟使能位** 0 = Flash ISP 外设时钟禁止。 1 = Flash ISP 外设时钟使能。
[1]	**PDMACKEN**	**PDMA 控制器时钟使能位** 0 = PDMA 外设时钟禁止。 1 = PDMA 外设时钟使能。
[0]	**Reserved**	保留。

APB设备时钟使能控制寄存器0 (CLK_APBCLK0)

该寄存器各位用于使能/禁用外设控制器时钟。

寄存器	偏移	R/W	描述	复位值
CLK_APBCLK0	CLK_BA+0x08	R/W	APB设备时钟使能控制寄存器0	0x0000_0001

31	30	29	28	27	26	25	24
Reserved	HSOTGCKEN	$I^2S0CKEN$	EADCCKEN	USBDCKEN	OTGCKEN	CAN1CKEN	CAN0CKEN
23	22	21	20	19	18	17	16
UART7CKEN	UART6CKEN	UART5CKEN	UART4CKEN	UART3CKEN	UART2CKEN	UART1CKEN	UART0CKEN
15	14	13	12	11	10	9	8
SPI^2CKEN	SPI1CKEN	SPI0CKEN	QSPI0CKEN	Reserved	$I^2C2CKEN$	$I^2C1CKEN$	$I^2C0CKEN$
7	6	5	4	3	2	1	0
ACMP01CKEN	CLKOCKEN	TMR3CKEN	TMR2CKEN	TMR1CKEN	TMR0CKEN	RTCCKEN	WDTCKEN

位	描述	
[31]	Reserved	保留。
[30]	HSOTGCKEN	**HSUSB OTG 时钟使能位** 0 = HSUSB OTG 时钟禁止。 1 = HSUSB OTG 时钟使能。
[29]	$I^2S0CKEN$	**I^2S0 时钟使能位** 0 = I^2S0 时钟禁止。 1 = I^2S0 时钟使能。
[28]	EADCCKEN	**增强型模数转换器 (EADC) 时钟使能位** 0 = EADC 时钟禁止。 1 = EADC 时钟使能。
[27]	USBDCKEN	**USB 设备时钟使能位** 0 = USB 设备时钟禁止。 1 = USB 设备时钟使能。
[26]	OTGCKEN	**USB OTG 时钟使能位** 0 = USB OTG 时钟禁止。 1 = USB OTG 时钟使能。
[25]	CAN1CKEN	**CAN1 时钟使能位** 0 = CAN1 时钟禁止。 1 = CAN1 时钟使能。
[24]	CAN0CKEN	**CAN0 时钟使能位**

		0 = CAN0 时钟禁止。 1 = CAN0 时钟使能。
[23]	**UART7CKEN**	**UART7时钟使能位** 0 = UART7时钟禁止。 1 = UART7时钟使能。
[22]	**UART6CKEN**	**UART6时钟使能位** 0 = UART6时钟禁止。 1 = UART6时钟使能。
[21]	**UART5CKEN**	**UART5时钟使能位** 0 = UART5 时钟禁止。 1 = UART5 时钟使能。
[20]	**UART4CKEN**	**UART4时钟使能位** 0 = UART4时钟禁止。 1 = UART4时钟使能。
[19]	**UART3CKEN**	**UART3时钟使能位** 0 = UART3时钟禁止。 1 = UART3时钟使能。
[18]	**UART2CKEN**	**UART2时钟使能位** 0 = UART2时钟禁止。 1 = UART2时钟使能。
[17]	**UART1CKEN**	**UART1时钟使能位** 0 = UART1时钟禁止。 1 = UART1时钟使能。
[16]	**UART0CKEN**	**UART0时钟使能位** 0 = UART0时钟禁止。 1 = UART0时钟使能。
[15]	**SPI^{2}CKEN**	**SPI2时钟使能位** 0 = SPI2时钟禁止。 1 = SPI2时钟使能。
[14]	**SPI1CKEN**	**SPI1时钟使能位** 0 = SPI1时钟禁止。 1 = SPI1时钟使能。
[13]	**SPI0CKEN**	**SPI0时钟使能位** 0 = SPI0时钟禁止。 1 = SPI0时钟使能。
[12]	**QSPI0CKEN**	**QSPI0时钟使能位** 0 = QSPI0时钟禁止。 1 = QSPI0时钟使能。
[11]	**Reserved**	保留。

[10]	**I²C2CKEN**	**I2C2 时钟使能位** 0 = I2C2 时钟禁止。 1 = I2C2 时钟使能。
[9]	**I²C1CKEN**	**I2C1 时钟使能位** 0 = I2C1 时钟禁止。 1 = I2C1 时钟使能。
[8]	**I²C0CKEN**	**I2C0 时钟使能位** 0 = I2C0 时钟禁止。 1 = I2C0时钟使能。
[7]	**ACMP01CKEN**	**模拟比较器 0/1时钟使能位** 0 =模拟比较器 0/1 时钟禁止。 1 =模拟比较器 0/1 时钟使能。
[6]	**CLKOCKEN**	**CLKO 时钟使能位** 0 = CLKO 时钟禁止。 1 = CLKO 时钟使能。
[5]	**TMR3CKEN**	**Timer3 时钟使能位** 0 = Timer3 时钟禁止。 1 = Timer3 时钟使能。
[4]	**TMR2CKEN**	**Timer2 时钟使能位** 0 = Timer2 时钟禁止。 1 = Timer2 时钟使能。
[3]	**TMR1CKEN**	**Timer1 时钟使能位** 0 = Timer1 时钟禁止。 1 = Timer1 时钟使能。
[2]	**TMR0CKEN**	**Timer0 时钟使能位** 0 = Timer0 时钟禁止。 1 = Timer0 时钟使能。
[1]	**RTCCKEN**	**实时时钟（Real-Time-Clock） APB 接口时钟使能位** 该位仅用于控制 RTC APB 时钟，RTC 外设的时钟源由RTCSEL(CLK_CLKSEL3[8])寄存器设置，可以选择32.768 kHz外部低速晶振或内部10kHz低速RC振荡器。 0 = RTC 时钟禁止。 1 = RTC 时钟使能。
[0]	**WDTCKEN**	**看门狗定时器时钟使能位 (写保护)** 0 = 看门狗定时器时钟禁止。 1 = 看门狗定时器时钟使能。 **注:** 该位写保护，参看SYS_REGLCTL寄存器。

APB设备时钟使能控制寄存器1 (CLK_APBCLK1)

该寄存器各位用于使能/禁用外设控制器时钟

寄存器	偏移	R/W	描述	复位值
CLK_APBCLK1	CLK_BA+0x0C	R/W	APB设备时钟使能控制寄存器1	0x0000_0000

31	30	29	28	27	26	25	24
EADC1CKEN	OPACKEN	Reserved	CAN2CKEN	ECAP1CKEN	ECAP0CKEN	TRNGCKEN	Reserved
23	22	21	20	19	18	17	16
QEI1CKEN	QEI0CKEN	Reserved		BPWM1CKEN	BPWM0CKEN	EPWM1CKEN	EPWM0CKEN
15	14	13	12	11	10	9	8
Reserved			DACCKEN	Reserved		USCI1CKEN	USCI0CKEN
7	6	5	4	3	2	1	0
Reserved			QSPI1CKEN	Reserved	SC2CKEN	SC1CKEN	SC0CKEN

位	描述	
[31]	EADC1CKEN	**增强型模-数-转换器1（EADC1）时钟使能位** 0 = EADC1时钟禁止。 1 = EADC1时钟使能。
[30]	OPACKEN	**OP 放大器 (OPA) 时钟使能位** 0 = OPA 时钟禁止。 1 = OPA 时钟使能。
[29]	Reserved	保留。
[28]	CAN2CKEN	**CAN2时钟使能位** 0 = CAN2时钟禁止。 1 = CAN2时钟使能。
[27]	ECAP1CKEN	**ECAP1时钟使能位** 0 = ECAP1时钟禁止。 1 = ECAP1时钟使能。
[26]	ECAP0CKEN	**ECAP0时钟使能位** 0 = ECAP0时钟禁止。 1 = ECAP0时钟使能。
[25]	TRNGCKEN	**TRNG时钟使能位** 0 = TRNG时钟禁止。 1 = TRNG时钟使能。
[24]	Reserved	保留。

[23]	**QEI1CKEN**	**QEI1时钟使能位** 0 = QEI1时钟禁止。 1 = QEI1时钟使能。
[22]	**QEI0CKEN**	**QEI0时钟使能位** 0 = QEI0时钟禁止。 1 = QEI0时钟使能。
[21:20]	**Reserved**	保留。
[19]	**BPWM1CKEN**	**BPWM1时钟使能位** 0 = BPWM1时钟禁止。 1 = BPWM1时钟使能。
[18]	**BPWM0CKEN**	**BPWM0时钟使能位** 0 = BPWM0时钟禁止。 1 = BPWM0时钟使能。
[17]	**EPWM1CKEN**	**EPWM1时钟使能位** 0 = EPWM1时钟禁止。 1 = EPWM1时钟使能。
[16]	**EPWM0CKEN**	**EPWM0时钟使能位** 0 = EPWM0时钟禁止。 1 = EPWM0时钟使能。
[15:13]	**Reserved**	保留。
[12]	**DACCKEN**	**DAC时钟使能位** 0 = DAC时钟禁止。 1 = DAC时钟使能。
[11:10]	**Reserved**	保留。
[9]	**USCI1CKEN**	**USCI1时钟使能位** 0 = USCI1时钟禁止。 1 = USCI1时钟使能。
[8]	**USCI0CKEN**	**USCI0时钟使能位** 0 = USCI0时钟禁止。 1 = USCI0时钟使能。
[7:5]	**Reserved**	保留。
[4]	**QSPI1CKEN**	**QSPI1时钟使能位** 0 = QSPI1时钟禁止。 1 = QSPI1时钟使能。
[3]	**Reserved**	保留。
[2]	**SC2CKEN**	**SC2时钟使能位** 0 = SC2时钟禁止。 1 = SC2时钟使能。

[1]	**SC1CKEN**	**SC1时钟使能位** 0 = SC1时钟禁止。 1 = SC1时钟使能。
[0]	**SC0CKEN**	**SC0时钟使能位** 0 = SC0时钟禁止。 1 = SC0时钟使能。

时钟源选择控制寄存器0 (CLK_CLKSEL0)

寄存器	偏移	R/W	描述	复位值
CLK_CLKSEL0	CLK_BA+0x10	R/W	时钟源选择控制寄存器0	0x00F3_013X

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
SDH1SEL		SDH0SEL		Reserved		CCAPSEL	
15	14	13	12	11	10	9	8
Reserved							USBSEL
7	6	5	4	3	2	1	0
Reserved		STCLKSEL			HCLKSEL		

位	描述	
[31:22]	**Reserved**	保留。
[23:22]	**SDH1SEL**	**SD1 引擎时钟源选择 (写保护)** 00 = 时钟源来自 HXT 时钟。 01 = 时钟源来自 PLL 时钟。 10 = 时钟源来自 HCLK.。 11 = 时钟源来自 HIRC 时钟。
[21:20]	**SDH0SEL**	**SD0 引擎时钟源选择 (写保护)** 00 = 时钟源来自 HXT 时钟。 01 = 时钟源来自 PLL 时钟。 10 = 时钟源来自 HCLK.。 11 = 时钟源来自 HIRC 时钟。
[19:18]	**Reserved**	保留。
[17:16]	**CCAPSEL**	**CCAP引擎时钟源选择(写保护)** 00 = 时钟源来自 HXT 时钟。 01 = 时钟源来自 PLL 时钟。 10 = 时钟源来自 HCLK.。 11 = 时钟源来自 HIRC 时钟。
[15:9]	**Reserved**	保留。
[8]	**USBSEL**	**USB 时钟源选择(写保护)** 0 =时钟源来自RC48M。 1 =时钟源来自PLL。

[7:6]	**Reserved**	保留。
[5:3]	**STCLKSEL**	**Cortex®-M4 SysTick时钟源选择(写保护)** 如果 SYST_CTRL[2]=0, SysTick 使用下列时钟源。 000 = 时钟源来自HXT。 001 = 时钟源来自LXT。 010 = 时钟源来自HXT/2。 011 = 时钟源来自HCLK/2。 111 = 时钟源来自HIRC/2。 **注1:**如果 SysTick 时钟源不是来自 HCLK (例如 SYST_ CTRL [2] = 0), SysTick 时钟源必须小于或等于HCLK/2。 **注2:** 该位写保护，参看SYS_REGLCTL寄存器。
[2:0]	**HCLKSEL**	**HCLK 时钟源选择(写保护)** 在时钟切换前，相关时钟源（预选和新选）必须打开。 000 =时钟来自 HXT。 001 =时钟来自LXT。 010 =时钟来自PLL。 011 =时钟来自LIRC。 111=时钟来自HIRC. 其它 = 保留。 **注:**该位写保护，参看SYS_REGLCTL寄存器。

时钟源选择控制寄存器1(CLK_CLKSEL1)

在切换时钟前，两个时钟源（预选与新选）都必须打开。

寄存器	偏移	R/W	描述	复位值
CLK_CLKSEL1	CLK_BA+0x14	R/W	时钟源选择控制寄存器1	0xBF77_7703

31	30	29	28	27	26	25	24
WWDTSEL		CLKOSEL		UART1SEL		UART0SEL	
23	22	21	20	19	18	17	16
Reserved	TMR3SEL			Reserved	TMR2SEL		
15	14	13	12	11	10	9	8
Reserved	TMR1SEL			Reserved	TMR0SEL		
7	6	5	4	3	2	1	0
Reserved						WDTSEL	

位	描述	
[31:30]	**WWDTSEL**	**窗口看门狗定时器时钟源选择** 10 = 时钟源为HCLK/2048。 11 = 时钟源为10 kHz 内部低速振荡器时钟(LIRC)。 其它=保留。
[29:28]	**CLKOSEL**	**时钟输出分频器时钟源选择** 00 = 时钟源为外部 4~24 MHz 高速晶振时钟(HXT)。 01 = 时钟源为外部 32.768 kHz 低速晶振时钟(LXT)。 10 = 时钟源为 HCLK。 11 = 时钟源为内部 12 MHz 高速振荡器时钟(HIRC)。
[27:26]	**UART1SEL**	**UART1 时钟源选择** 00 = 时钟源为外部 4~24 MHz 高速晶振时钟(HXT)。 01 = 时钟源为PLL。 10 = 时钟源为外部32.768 kHz 低速晶振时钟(LXT)。 11 = 时钟源为内部 12 MHz 高速振荡器时钟(HIRC)。
[25:24]	**UART0SEL**	**UART0 时钟源选择** 00 = 时钟源为外部 4~24 MHz 高速晶振时钟(HXT)。 01 = 时钟源为PLL。 10 = 时钟源为外部32.768 kHz 低速晶振时钟(LXT)。 11 = 时钟源为内部 12 MHz 高速振荡器时钟(HIRC)。
[23]	**Reserved**	保留。
[22:20]	**TMR3SEL**	**TIMER3 时钟源选择**

		000 = 时钟源为外部 4~24 MHz高速晶振时钟(HXT)。 001 = 时钟源为外部 32.768 kHz 低速晶振时钟(LXT)。 010 = 时钟源为PCLK1。 011 = 时钟源为外部时钟TM3 管脚。 101 = 时钟源为10 kHz 内部低速振荡器时钟(LIRC)。 111 = 时钟源为内部 12 MHz 高速振荡器时钟(HIRC)。 其它 = 保留。
[19]	**Reserved**	保留。
[18:16]	**TMR2SEL**	**TIMER2 时钟源选择** 000 = 时钟源为外部 4~24 MHz高速晶振时钟(HXT)。 001 = 时钟源为外部 32.768 kHz 低速晶振时钟(LXT)。 010 = 时钟源为PCLK1。 011 = 时钟源为外部时钟TM2 管脚。 101 = 时钟源为10 kHz 内部低速振荡器时钟(LIRC)。 111 = 时钟源为内部 12 MHz 高速振荡器时钟(HIRC)。 其它 = 保留。
[15]	**Reserved**	保留。
[14:12]	**TMR1SEL**	**TIMER1 时钟源选择** 000 = 时钟源为外部 4~24 MHz高速晶振时钟(HXT)。 001 = 时钟源为外部 32.768 kHz 低速晶振时钟(LXT)。 010 = 时钟源为PCLK0。 011 = 时钟源为外部时钟TM1 管脚。 101 = 时钟源为10 kHz 内部低速振荡器时钟(LIRC)。 111 = 时钟源为内部 12 MHz 高速振荡器时钟(HIRC)。 其它 = 保留。
[11]	**Reserved**	保留。
[10:8]	**TMR0SEL**	**TIMER0 时钟源选择** 000 = 时钟源为外部 4~24 MHz高速晶振时钟(HXT)。 001 = 时钟源为外部 32.768 kHz 低速晶振时钟(LXT)。 010 = 时钟源为PCLK0。 011 = 时钟源为外部时钟TM0 管脚。 101 = 时钟源为10 kHz 内部低速振荡器时钟(LIRC)。 111 = 时钟源为内部 12 MHz 高速振荡器时钟(HIRC)。 其它 = 保留。
[7:2]	**Reserved**	保留。
[1:0]	**WDTSEL**	**看门狗定时器时钟源选择(写保护)** 00 = 保留。 01 = 时钟源为外部 32.768 kHz 低速晶振时钟(LXT)。 10 = 时钟源为HCLK/2048。 11 = 时钟源为10 kHz 内部低速振荡器时钟(LIRC)。 **注:** 该位写保护，参看SYS_REGLCTL寄存器。

时钟源选择控制寄存器2 (CLK_CLKSEL2)

在切换时钟前，两个时钟源（预选与新选）都必须打开。

寄存器	偏移	R/W	描述	复位值
CLK_CLKSEL2	CLK_BA+0x18	R/W	时钟源选择控制寄存器2	0x0000_03AB

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved				SPI^{2}SEL		BPWM1SEL	BPWM0SEL
7	6	5	4	3	2	1	0
SPI1SEL		SPI0SEL		QSPI0SEL		EPWM1SEL	EPWM0SEL

位	描述	
[31:14]	**Reserved**	保留。
[13:12]	**Reserved**	保留。
[11:10]	**SPI^{2}SEL**	**SPI2时钟源选择** 00 = 时钟源为外部 4~24 MHz 高速晶振时钟(HXT)。 01 = 时钟源为PLL。 10 = 时钟源为PCLK1。 11 = 时钟源为内部 12 MHz 高速振荡器时钟(HIRC)。
[9]	**BPWM1SEL**	**BPWM1 时钟源选择** BPWM1 的外设时钟源由BPWM1SEL.来定义。 0 = 时钟源为PLL。 1 = 时钟源为PCLK1。
[8]	**BPWM0SEL**	**BPWM0 时钟源选择** BPWM0 的外设时钟源由BPWM0SEL.来定义。 0 = 时钟源为PLL。 1 = 时钟源为PCLK0。
[7:6]	**SPI1SEL**	**SPI1时钟源选择** 00 = 时钟源为外部 4~24 MHz 高速晶振时钟(HXT)。 01 = 时钟源为PLL。 10 = 时钟源为PCLK0。 11 = 时钟源为内部 12 MHz 高速振荡器时钟(HIRC)。
[5:4]	**SPI0SEL**	**SPI0时钟源选择**

		00 = 时钟源为外部 4~24 MHz 高速晶振时钟(HXT)。 01 = 时钟源为PLL。 10 = 时钟源为PCLK1。 11 = 时钟源为内部 12 MHz 高速振荡器时钟(HIRC)。
[3:2]	**QSPI0SEL**	**QSPI0时钟源选择** 00 = 时钟源为外部 4~24 MHz 高速晶振时钟(HXT)。 01 = 时钟源为PLL。 10 = 时钟源为PCLK0。 11 = 时钟源为内部 12 MHz 高速振荡器时钟(HIRC)。
[1]	**EPWM1SEL**	**EPWM1 时钟源选择** EPWM1 的外设时钟源由EPWM1SEL.来定义。 0 = 时钟源为PLL。 1 = 时钟源为PCLK1。
[0]	**EPWM0SEL**	**EPWM0 时钟源选择** EPWM0 的外设时钟源由EPWM0SEL.来定义。 0 = 时钟源为PLL。 1 = 时钟源为PCLK0。

时钟源选择控制寄存器3 (CLK_CLKSEL3)

在时钟切换之前，必须打开相关的时钟源（预选和新选)。

寄存器	偏移	R/W	描述	复位值
CLK_CLKSEL3	CLK_BA+0x1C	R/W	时钟源选择控制寄存器3	0xFF00_003F

31	30	29	28	27	26	25	24
UART5SEL		UART4SEL		UART3SEL		UART2SEL	
23	22	21	20	19	18	17	16
UART7SEL		UART6SEL		Reserved		I^2S0SEL	
15	14	13	12	11	10	9	8
Reserved		QSPI1SEL		Reserved			RTCSEL
7	6	5	4	3	2	1	0
Reserved		SC2SEL		SC1SEL		SC0SEL	

位	描述	
[31:30]	**UART5SEL**	**UART5时钟源选择** 00 = 时钟源为外部 4~24 MHz 高速晶振时钟(HXT)。 01 = 时钟源为PLL。 10 = 时钟源为外部32.768 kHz 低速晶振时钟(LXT)。 11 = 时钟源为内部 12 MHz 高速振荡器时钟(HIRC)。
[29:28]	**UART4SEL**	**UART4时钟源选择** 00 = 时钟源为外部 4~24 MHz 高速晶振时钟(HXT)。 01 = 时钟源为PLL。 10 = 时钟源为外部32.768 kHz 低速晶振时钟(LXT)。 11 = 时钟源为内部 12 MHz 高速振荡器时钟(HIRC)。
[27:26]	**UART3SEL**	**UART3时钟源选择** 00 = 时钟源为外部 4~24 MHz 高速晶振时钟(HXT)。 01 = 时钟源为PLL。 10 = 时钟源为外部32.768 kHz 低速晶振时钟(LXT)。 11 = 时钟源为内部 12 MHz 高速振荡器时钟(HIRC)。
[25:24]	**UART2SEL**	**UART2时钟源选择** 00 = 时钟源为外部 4~24 MHz 高速晶振时钟(HXT)。 01 = 时钟源为PLL。 10 = 时钟源为外部32.768 kHz 低速晶振时钟(LXT)。 11 = 时钟源为内部 12 MHz 高速振荡器时钟(HIRC)。
[23:22]	**UART7SEL**	**UART7时钟源选择**

		00 = 时钟源为外部 4~24 MHz 高速晶振时钟(HXT)。 01 = 时钟源为PLL。 10 = 时钟源为外部32.768 kHz 低速晶振时钟(LXT)。 11 = 时钟源为内部 12 MHz 高速振荡器时钟(HIRC)。
[21:20]	**UART6SEL**	**UART6时钟源选择** 00 = 时钟源为外部 4~24 MHz 高速晶振时钟(HXT)。 01 = 时钟源为PLL。 10 = 时钟源为外部32.768 kHz 低速晶振时钟(LXT)。 11 = 时钟源为内部 12 MHz 高速振荡器时钟(HIRC)。
[19:18]	**Reserved**	保留。
[17:16]	**I^{2}S0SEL**	**I^{2}S0 时钟源选择** 00 = 时钟源为外部 4~24 MHz 高速晶振时钟(HXT)。 01 = 时钟源为PLL。 10 = 时钟源为PCLK0。 11 = 时钟源为内部 12 MHz 高速振荡器时钟(HIRC)。
[15:14]	**Reserved**	保留。
[13:12]	**QSPI1SEL**	**QSPI1时钟源选择** 00 = 时钟源为外部 4~24 MHz 高速晶振时钟(HXT)。 01 = 时钟源为PLL。 10 = 时钟源为PCLK1。 11 = 时钟源为内部 12 MHz 高速振荡器时钟(HIRC)。
[11:9]	**Reserved**	保留。
[8]	**RTCSEL**	**RTC 时钟源选择** 0 = 时钟源为外部32.768 kHz 低速晶振时钟(LXT)。 1 = 时钟源为内部 10 kHz 低速振荡器时钟 (LIRC)。
[7:6]	**Reserved**	保留。
[5:4]	**SC2SEL**	**SC2 时钟源选择** 00 = 时钟源为外部 4~24 MHz 高速晶振时钟(HXT)。 01 = 时钟源为PLL。 10 = 时钟源为PCLK0。 11 = 时钟源为内部 12 MHz 高速振荡器时钟(HIRC)。
[3:2]	**SC1SEL**	**SC1时钟源选择** 00 = 时钟源为外部 4~24 MHz 高速晶振时钟(HXT)。 01 = 时钟源为PLL。 10 = 时钟源为PCLK1。 11 = 时钟源为内部 12 MHz 高速振荡器时钟(HIRC)。
[1:0]	**SC0SEL**	**SC0 时钟源选择** 00 = 时钟源为外部 4~24 MHz 高速晶振时钟(HXT)。 01 = 时钟源为PLL。 10 = 时钟源为PCLK0。

		11 = 时钟源为内部 12 MHz 高速振荡器时钟(HIRC)。

时钟分频数寄存器0 (CLK_CLKDIV0)

寄存器	偏移	R/W	描述	复位值
CLK_CLKDIV0	CLK_BA+0x20	R/W	时钟分频数寄存器0	0x0000_0000

31	30	29	28	27	26	25	24
SDH0DIV							
23	22	21	20	19	18	17	16
EADCDIV							
15	14	13	12	11	10	9	8
UART1DIV				UART0DIV			
7	6	5	4	3	2	1	0
USBDIV				HCLKDIV			

位	描述	
[31:24]	SDH0DIV	**SD0 时钟除频数，时钟源来自SD0时钟源** SD0 时钟频率= (SD0 时钟源频率) / (SDH0DIV + 1)。
[23:16]	EADCDIV	**EADC时钟除频数，时钟源来自EADC时钟源** EADC 时钟频率= (EADC时钟源频率) / (EADCDIV + 1)。
[15:12]	UART1DIV	**UART1 时钟除频数，时钟源来自UART1时钟源** UART1时钟频率= (UART1时钟源频率) / (UART1DIV + 1)。
[11:8]	UART0DIV	**UART0 时钟除频数，时钟源来自UART0时钟源** UART0时钟频率= (UART0时钟源频率) / (UART0DIV + 1)。
[7:4]	USBDIV	**USB时钟除频数，时钟源来自PLL时钟** USB时钟频率= (PLL频率) / (USBDIV + 1)。
[3:0]	HCLKDIV	**HCLK时钟除频数，时钟源来自HCLK时钟源** HCLK 时钟频率= (HCLK时钟源频率) / (HCLKDIV + 1)。

时钟分频数寄存器1 (CLK_CLKDIV1)

寄存器	偏移	R/W	描述	复位值
CLK_CLKDIV1	CLK_BA+0x24	R/W	时钟分频数寄存器1	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
SC2DIV							
15	14	13	12	11	10	9	8
SC1DIV							
7	6	5	4	3	2	1	0
SC0DIV							

位	描述	
[31:24]	**Reserved**	保留。
[23:16]	**SC2DIV**	**SC2时钟的除频数，时钟源来自SC2时钟源** SC2 时钟频率= (SC2时钟源频率) / (SC2DIV + 1)。
[15:8]	**SC1DIV**	**SC1时钟的除频数，时钟源来自SC1时钟源** SC1时钟频率= (SC1时钟源频率) / (SC1DIV + 1)。
[7:0]	**SC0DIV**	**SC0时钟的除频数，时钟源来自SC0时钟源** SC0时钟频率= (SC0时钟源频率) / (SC0DIV + 1)。

时钟分频数寄存器2 (CLK_CLKDIV2)

寄存器	偏移	R/W	描述	复位值
CLK_CLKDIV2	CLK_BA+0x28	R/W	时钟分频数寄存器2	0x0000_0000

31	30	29	28	27	26	25	24
EADC1DIV							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved				**I²SDIV**			

位	描述	
[31:24]	**EADC1DIV**	**EADC1时钟的除频数，时钟源来自EADC时钟源** EADC1时钟频率= (EADC1 时钟频率) / (EADC1DIV + 1)。
[23:4]	**Reserved**	保留。
[3:0]	**I²SDIV**	**I²S时钟的除频数，时钟源来自I²S时钟源** I²S时钟频率= (I²S时钟频率) / (I²SDIV + 1)。

时钟分频数寄存器3 (CLK_CLKDIV3)

寄存器	偏移	R/W	描述	复位值
CLK_CLKDIV3	CLK_BA+0x2C	R/W	时钟分频数寄存器3	0x0000_0000

31	30	29	28	27	26	25	24
SDH1DIV							
23	22	21	20	19	18	17	16
EMACDIV							
15	14	13	12	11	10	9	8
VSENSEDIV							
7	6	5	4	3	2	1	0
CCAPDIV							

位	描述	
[31:24]	SDH1DIV	**SD1时钟的除频数，时钟源来自SD1时钟源** SD1 时钟频率= (SD1时钟源频率) / (SDH1DIV + 1)。
[23:16]	EMACDIV	**以太网时钟的除频数，时钟源来自HCLK** EMAC MDCLK时钟频率= (HCLK) / (EMACDIV + 1)。
[15:8]	VSENSEDIV	**视频像素时钟的除频数，时钟源来自CCAP时钟源** 视频像素时钟频率= (CCAP时钟频率) / (VSENSEDIV + 1)。
[7:0]	CCAPDIV	**相机捕获传感器的时钟除频数，时钟源来自CCAP时钟源** 图像传感器时钟频率= (CCAP 时钟频率) / (CCAPDIV + 1)。

时钟分频数寄存器4 (CLK_CLKDIV4)

寄存器	偏移	R/W	描述	复位值
CLK_CLKDIV4	CLK_BA+0x30	R/W	时钟分频数寄存器4	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
UART7DIV				UART6DIV			
15	14	13	12	11	10	9	8
UART5DIV				UART4DIV			
7	6	5	4	3	2	1	0
UART3DIV				UART2DIV			

位	描述	
[31:24]	Reserved	保留。
[23:20]	UART7DIV	**UART7时钟除频数，时钟源来自UART7时钟源** UART7时钟频率= (UART7时钟源频率) / (UART7DIV + 1)。
[19:16]	UART6DIV	**UART6时钟除频数，时钟源来自UART6时钟源** UART6时钟频率= (UART6时钟源频率) / (UART6DIV + 1)。
[15:12]	UART5DIV	**UART5 时钟除频数，时钟源来自UART5时钟源** UART5时钟频率= (UART5时钟源频率) / (UART5DIV + 1)。
[11:8]	UART4DIV	**UART4 时钟除频数，时钟源来自UART4时钟源** UART4时钟频率= (UART4时钟源频率) / (UART4DIV + 1)。
[7:4]	UART3DIV	**UART3 时钟除频数，时钟源来自UART3时钟源** UART3时钟频率= (UART3时钟源频率) / (UART3DIV + 1)。
[3:0]	UART2DIV	**UART2 时钟除频数，时钟源来自UART2时钟源** UART2时钟频率= (UART2时钟源频率) / (UART2DIV + 1)。

APB时钟分频器寄存器(CLK_PCLKDIV)

寄存器	偏移	R/W	描述	复位值
CLK_PCLKDIV	CLK_BA+0x34	R/W	APB时钟分频器寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved	APB1DIV			Reserved	APB0DIV		

位	描述	
[31:7]	Reserved	保留。
[6:4]	APB1DIV	**APB1 时钟分频器** APB1 时钟可以被分频，时钟源来自 HCLK。 000: PCLK1 = HCLK。 001: PCLK1 = 1/2 HCLK。 010: PCLK1 = 1/4 HCLK。 011: PCLK1 = 1/8 HCLK。 100: PCLK1 = 1/16 HCLK。 其他:保留。 **注:** PCLK1 必须低于96MHz。
[3]	Reserved	保留。
[2:0]	APB0DIV	**APB0 时钟分频器** APB0 时钟可以被分频，时钟源来自 HCLK。 000: PCLK0 = HCLK。 001: PCLK0 = 1/2 HCLK。 010: PCLK0 = 1/4 HCLK。 011: PCLK0 = 1/8 HCLK。 100: PCLK0 = 1/16 HCLK。 其他:保留。 **注:** PCLK0 必须低于96MHz。

PLL 控制寄存器 (CLK_PLLCTL)

PLL参考时钟输入来自4~32MHz外部高速晶体振荡器（HXT）时钟输入或来自12MHz内部高速RC振荡器（HIRC）。该寄存器用于控制PLL输出频率和PLL操作模式。

对这些位进行编程需要在地址0x4000_0100处写入“59h”，“16h”和”88h”以禁用寄存器保护。请参阅SYS_BA+0x100地址处的寄存器SYS_REGLCTL。

只有在PD(CLK_PLLCTL[16]) 置位时，CLK_PLLCTL才能修改。

寄存器	偏移	R/W	描述	复位值
CLK_PLLCTL	CLK_BA+0x40	R/W	PLL 控制寄存器	0x0005_C02E

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
STBSEL	Reserved			PLLSRC	OE	BP	PD
15	14	13	12	11	10	9	8
OUTDIV		INDIV					FBDIV
7	6	5	4	3	2	1	0
FBDIV							

位	描述	
[31:24]	**Reserved**	保留。
[23]	**STBSEL**	**PLL稳定计数个数选择 (写保护)** 0 = PLL 稳定时间为1200个 PLL 时钟源周期(适合时钟源频率等于或小于12MHz)。 1 = PLL 稳定时间为2400个PLL 时钟源周期 (适合时钟源频率大于12 MHz)。 **注:** 该位写保护，参看SYS_REGLCTL寄存器。
[22:20]	**Reserved**	保留。
[19]	**PLLSRC**	**PLL 时钟源选择(写保护)** 0 = PLL 时钟源为外部 4~24 MHz 高速晶振(HXT)。 1 = PLL 时钟源为内部12 MHz 高速振荡器(HIRC) 。 **注意:** 该位写保护，参看SYS_REGLCTL寄存器。
[18]	**OE**	**PLL OE (FOUT 使能) 管脚控制 (写保护)** 0 = PLL FOUT 使能。 1 = PLL FOUT 固定为低。 **注1:** 该位写保护，参看SYS_REGLCTL寄存器。 **注2:** 如果HCLK切换到PLL,OE(CLK_PLLCTL[18])不能被设为1，以避免时钟停止。
[17]	**BP**	**PLL 旁路控制 (写保护)**

		0 = PLL 正常模式（默认）。 1 = PLL 时钟输出与PLL时钟源(FIN)输入相同。 **注:** 该位写保护，参看SYS_REGLCTL寄存器。.
[16]	**PD**	**掉电模式(写保护)** 如果设置寄存器CLK_PWRCTL的PDEN位为 1，PLL 将进入掉电模式。 0 = PLL 正常模式。 1 = PLL 进入掉电模式（默认）。 **注:** 该位写保护，参看SYS_REGLCTL寄存器。
[15:14]	**OUTDIV**	**PLL 输出分频控制(写保护)** 参考下表公式。 **注:** 该位写保护，参看SYS_REGLCTL寄存器。
[13:9]	**INDIV**	**PLL 输入分频控制(写保护)** 参考下表公式。 **注:** 该位写保护，参看SYS_REGLCTL寄存器。
[8:0]	**FBDIV**	**PLL 反馈分频控制(写保护)** 参考下表公式。 **注:** 该位写保护，参看SYS_REGLCTL寄存器。

输出时钟频率公式:

$$FREF = FIN \times \frac{1}{NR}$$

$$FVCO = FIN \times \frac{2 * NF}{NR}$$

$$FOUT = FIN \times \frac{2 * NF}{NR} \times \frac{1}{NO}$$

这里的 FREF是用于PFD（相位频率侦测器）的比较频率。

为了PLL在正常模式更好的工作，下面约束条件必须满足：

4 MHz ≤FREF ≤8 MHz

200 MHz≤ FVCO ≤500 MHz

50MHz≤ FOUT≤500 MHz

符号	描述
FOUT	输出时钟频率
FIN	输入（参考）时钟频率
NR	输入分频 (INDIV + 1)
NF	反馈分频(FBDIV + 2)

NO	OUTDIV	=	“00”	:	NO	=	1
	OUTDIV	=	“01”	:	NO	=	2
	OUTDIV	=	“10”	:	NO	=	2
	OUTDIV = “11” : NO = 4						

表6.3-1输出频率公式的符号定义

时钟状态监测寄存器(CLK_STATUS)

该寄存器各位用于监控芯片时钟源是否稳定，时钟切换是否失败。

寄存器	偏移	R/W	描述	复位值
CLK_STATUS	CLK_BA+0x50	R	时钟状态监测寄存器	0x0000_00XX

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
CLKSFAIL	Reserved		HIRCSTB	LIRCSTB	PLLSTB	LXTSTB	HXTSTB

位	描述	
[31:8]	Reserved	保留。
[7]	CLKSFAIL	**时钟切换失败标志（只读）** 当用户切换系统时钟源时，该位被更新。如果切换目标时钟稳定，该位将被设置为0。如果切换目标时钟不稳定，该位将被设置为1。 0 = 时钟切换成功。 1 = 时钟切换失败。 **注:** 该位写1清0。
[7:5]	Reserved	保留。
[4]	HIRCSTB	**HIRC时钟源稳定标志（只读）** 0 = 12 MHz 内部高速RC 振荡器(HIRC) 时钟不稳定或者禁用。 1 = 12 MHz 内部高速RC 振荡器(HIRC) 时钟使能并稳定。
[3]	LIRCSTB	**LIRC 时钟源稳定标志（只读）** 0 = 内部10 kHz低速RC振荡器 (LIRC) 时钟不稳定或者禁用。 1 = 内部10 kHz低速RC振荡器(LIRC) 时钟使能并稳定。
[2]	PLLSTB	**内部 PLL 时钟源稳定标志(只读)** 0 = 内部 PLL时钟不稳定或者禁用。 1 = 内部PLL 时钟使能并稳定。
[1]	LXTSTB	**LXT 时钟源稳定标志（只读）** 0 = 外部32.768 kHz低速晶振(LXT) 时钟不稳定或者禁用。 1 = 外部32.768 kHz低速晶振(LXT) 时钟使能并稳定。
[0]	HXTSTB	**HXT 时钟源稳定标志（只读）**

		0 = 外部4~24 MHz高速晶振(HXT) 时钟不稳定或者禁用。 1 = 外部4~24 MHz高速晶振(HXT) 时钟使能并稳定。

时钟输出控制寄存器(CLK_CLKOCTL)

寄存器	偏移	R/W	描述	复位值
CLK_CLKOCTL	CLK_BA+0x60	R/W	时钟输出控制寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved	CLK1HZEN	DIV1EN	CLKOEN	FREQSEL			

位	描述	
[31:7]	**Reserved**	保留。
[6]	**CLK1HZEN**	**时钟输出 1Hz 使能位** 0 = 禁止32.768 kHz频率补偿输出 1Hz 时钟。 1 = 使能32.768 kHz频率补偿输出 1Hz 时钟。
[5]	**DIV1EN**	**时钟输出分频 1使能位** 0 = 时钟输出的时钟来自被FREQSEL分频的时钟源。 1 = 时钟输出的时钟来自时钟源。
[4]	**CLKOEN**	**时钟输出使能位** 0 = 禁用时钟输出功能。 1 = 使能时钟输出功能。
[3:0]	**FREQSEL**	**时钟输出频率选择位** 输出频率的公式： $F_{out} = F_{in}/2^{(N+1)}$. F_{in}为输入时钟频率。 F_{out}为分频器输出时钟频率。 N为FREQSEL[3:0]的4位值。

时钟失败侦测器控制寄存器(CLK_CLKDCTL)

寄存器	偏移	R/W	描述	复位值
CLK_CLKDCTL	CLK_BA+0x70	R/W	时钟失败检测器控制寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved						HXTFQIEN	HXTFQDEN
15	14	13	12	11	10	9	8
Reserved		LXTFIEN	LXTFDEN	Reserved			
7	6	5	4	3	2	1	0
Reserved		HXTFIEN	HXTFDEN	Reserved			

位	描述	
[31:18]	**Reserved**	保留。
[17]	**HXTFQIEN**	**HXT 时钟频率范围监测中断使能位** 0 = 禁止外部4~24 MHz高速晶振(HXT) 时钟频率范围监测失败中断。 1 = 使能外部4~24 MHz高速晶振(HXT) 时钟频率范围监测失败中断。
[16]	**HXTFQDEN**	**HXT 时钟频率监测使能位** 0 = 禁止外部4~24 MHz高速晶振(HXT)时钟频率范围监测。 1 = 使能外部4~24 MHz高速晶振(HXT)时钟频率范围监测。
[15:14]	**Reserved**	保留。
[13]	**LXTFIEN**	**LXT 时钟失败中断使能位** 0 = 禁止外部32.768 kHz低速晶振(LXT)时钟失败中断。 1 = 使能外部32.768 kHz低速晶振(LXT)时钟失败中断。
[12]	**LXTFDEN**	**LXT 时钟失败检测器使能位** 0 = 禁止外部32.768 kHz低速晶振(LXT)时钟失败检测器。 1 = 使能外部32.768 kHz低速晶振(LXT)时钟失败检测器。
[11:6]	**Reserved**	保留。
[5]	**HXTFIEN**	**HXT 时钟失败中断使能位** 0 = 禁止外部4~24 MHz高速晶振(HXT)时钟失败中断。 1 = 使能外部4~24 MHz高速晶振(HXT)时钟失败中断。
[4]	**HXTFDEN**	**HXT 时钟失败检测器使能位** 0 = 禁止外部4~24 MHz高速晶振(HXT)时钟失败检测器。 1 = 使能外部4~24 MHz高速晶振(HXT)时钟失败检测器。

[3:0]	**Reserved**	保留。

时钟失败侦测器状态寄存器(CLK_CLKDSTS)

寄存器	偏移	R/W	描述	复位值
CLK_CLKDSTS	CLK_BA+0x74	R/W	时钟失败侦测器状态寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							HXTFQIF
7	6	5	4	3	2	1	0
Reserved						LXTFIF	HXTFIF

位	描述	
[31:9]	**Reserved**	保留。
[8]	**HXTFQIF**	**HXT 时钟频率范围监测中断标志位** 0 = 外部4~24 MHz高速晶振(HXT)时钟正常。 1 = 外部4~24 MHz高速晶振(HXT)时钟频率异常。 **注：** 该位写1清除。
[7:2]	**Reserved**	保留。
[1]	**LXTFIF**	**LXT 时钟失败中断标志位** 0 =外部32.768 kHz低速晶振(LXT)时钟正常。 1 =外部32.768 kHz低速晶振(LXT)时钟停止。 **注：** 该位写1清除。
[0]	**HXTFIF**	**HXT 时钟失败中断标志位** 0 =外部4~24 MHz高速晶振(HXT)时钟正常。 1 =外部4~24 MHz高速晶振(HXT)时钟停止。 **注：** 该位写1清除。

时钟频率范围检测器上边界寄存器(CLK_CDUPB)

寄存器	偏移	R/W	描述	复位值
CLK_CDUPB	CLK_BA+0x78	R/W	时钟频率检测器上边界寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved						UPERBD	
7	6	5	4	3	2	1	0
UPERBD							

位	描述	
[31:10]	Reserved	保留。
[9:0]	UPERBD	HXT 时钟频率范围检测器上边界值 该位定义频率范围监测窗口的最大值。 当HXT频率监测值高于该寄存器的值时，HXT频率范围检测器中断标志将置1。

时钟频率范围检测器下边界寄存器(CLK_CDLOWB)

寄存器	偏移	R/W	描述	复位值
CLK_CDLOWB	CLK_BA+0x7c	R/W	时钟频率范围检测器下边界寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved						LOWERBD	
7	6	5	4	3	2	1	0
LOWERBD							

位	描述	
[31:10]	Reserved	保留。
[9:0]	LOWERBD	HXT 时钟频率范围检测器下边界 该位定义频率范围监测窗口的最小值。 当HXT频率监测值低于该寄存器的值时，HXT频率范围检测器中断标志将置1。

电源管理控制寄存器(CLK_PMUCTL)

寄存器	偏移	R/W	描述	复位值
CLK_PMUCTL	CLK_BA+0x90	R/W	电源管理控制寄存器	0x0000_0080

31	30	29	28	27	26	25	24
WKPINEN4		WKPINEN3		WKPINEN2		WKPINEN1	
23	22	21	20	19	18	17	16
RTCWKEN	Reserved				ACMPSPWK	WKPINEN0	
15	14	13	12	11	10	9	8
Reserved			WKTMRIS				WKTMREN
7	6	5	4	3	2	1	0
Reserved	SRETSEL			DPDHOLDEN	PDMSEL		

位	描述	
[31:30]	**WKPINEN4**	**唤醒管脚4使能位（写保护）** 该寄存器写保护。请参照打开锁定顺序进行编程。 此位是GPF.6做为唤醒脚的控制寄存器。 00 = 禁用深度掉电模式下唤醒脚。 01 = 使能深度掉电模式下唤醒脚上升沿唤醒。 10 = 使能深度掉电模式下唤醒脚下降沿唤醒。 11 = 使能深度掉电模式下唤醒脚双边沿唤醒。 **注：**该位是写保护的。请参考SYS_REGLCTL寄存器。
[29:28]	**WKPINEN3**	**唤醒管脚3使能位（写保护）** 该寄存器写保护。请参照打开锁定顺序进行编程。 此位是GPB.12做为唤醒脚的控制寄存器。 00 = 禁用深度掉电模式下唤醒脚。 01 = 使能深度掉电模式下唤醒脚上升沿唤醒。 10 = 使能深度掉电模式下唤醒脚下降沿唤醒。 11 = 使能深度掉电模式下唤醒脚双边沿唤醒。 **注：**该位是写保护的。请参考SYS_REGLCTL寄存器。
[27:26]	**WKPINEN2**	**唤醒管脚2使能位（写保护）** 该寄存器写保护。请参照打开锁定顺序进行编程。 此位是GPB.2做为唤醒脚的控制寄存器。 00 = 禁用深度掉电模式下唤醒脚。 01 = 使能深度掉电模式下唤醒脚上升沿唤醒。 10 = 使能深度掉电模式下唤醒脚下降沿唤醒。 11 = 使能深度掉电模式下唤醒脚双边沿唤醒。 **注：**该位是写保护的。请参考SYS_REGLCTL寄存器。

[25:24]	**WKPINEN1**	**唤醒管脚1使能位（写保护）** 该寄存器写保护。请参照打开锁定顺序进行编程。 此位是GPB.0做为唤醒脚的控制寄存器。 00 = 禁用深度掉电模式下唤醒脚。 01 = 使能深度掉电模式下唤醒脚上升沿唤醒。 10 = 使能深度掉电模式下唤醒脚下降沿唤醒。 11 = 使能深度掉电模式下唤醒脚双边沿唤醒。 **注：**该位是写保护的。请参考SYS_REGLCTL寄存器。
[23]	**RTCWKEN**	**RTC 唤醒使能 (写保护)** 该寄存器写保护。请参照打开锁定顺序进行编程。 0 = 禁止RTC在深度睡眠模式或空闲模式唤醒系统。 1 = 使能RTC在深度睡眠模式或空闲模式唤醒系统。 **注：**该位是写保护位。参考SYS_REGLCTL寄存器。
[18]	**ACMPSPWK**	**ACMP 待机掉电模式唤醒使能 (写保护)** 该寄存器写保护。请参照打开锁定顺序进行编程。 0 = 禁止ACMP在待机掉电模式唤醒系统。 1 = 使能ACMP在待机掉电模式唤醒系统。 **注：**该位是写保护位。参考SYS_REGLCTL寄存器。
[17:16]	**WKPINEN0**	**唤醒管脚0使能位（写保护）** 该寄存器写保护。请参照打开锁定顺序进行编程。 此位是GPC.0做为唤醒脚的控制寄存器。 00 = 禁用深度掉电模式下唤醒脚。 01 = 使能深度掉电模式下唤醒脚上升沿唤醒。 10 = 使能深度掉电模式下唤醒脚下降沿唤醒。 11 = 使能深度掉电模式下唤醒脚双边沿唤醒。 **注：**该位是写保护的。请参考SYS_REGLCTL寄存器。
[15:13]	**Reserved**	保留。
[12:9]	**WKTMRIS**	**唤醒定时器超时间隔选择 (写保护)** 该寄存器写保护。请参照打开锁定顺序进行编程。 当芯片在DPD/SPD模式，这些位控制唤醒定时器超时间隔。 0000 = 超时间隔是128个 OSC10K 时钟 (12.8 ms)。 0001 =超时间隔是256个 OSC10K 时钟(25.6 ms)。 0010 =超时间隔是512个 OSC10K 时钟 (51.2 ms)。 0011 =超时间隔是1024个 OSC10K 时钟 (102.4ms)。 0100 =超时间隔是4096个 OSC10K 时钟 (409.6ms)。 0101 =超时间隔是8192个 OSC10K 时钟(819.2ms)。 0110 =超时间隔是16384个 OSC10K 时钟 (1638.4ms)。 0111 =超时间隔是65536个 OSC10K 时钟 (6553.6ms)。 1000 =超时间隔是131072个 OSC10K 时钟(13107.2ms)。 1001 =超时间隔是262144个 OSC10K 时钟(26214.4ms)。 1010 =超时间隔是524288个 OSC10K 时钟(52428.8ms)。 1011 =超时间隔是1048576个 OSC10K 时钟(104857.6ms)。

		其它 =超时间隔是128个 OSC10K 时钟(12.8ms)。 **注：**该位是写保护位。参考SYS_REGLCTL寄存器。
[8]	**WKTMREN**	**唤醒定时器使能 (写保护)** 该寄存器写保护。请参照打开锁定顺序进行编程。 0 = 在 DPD/SPD 模式，唤醒定时器禁止。 1 = 在 DPD/SPD 模式，唤醒定时器使能。 **注：**该位是写保护位。参考SYS_REGLCTL寄存器。
[7]	**Reserved**	保留。
[6:4]	**SRETSEL**	**SRAM保持范围选择位(写保护)** 当芯片进入SPD模式，选择SRAM保持范围。 000 = SRAM不保持。 001 = 当芯片进入SPD模式，SRAM保持16K 。 010 =当芯片进入SPD模式，SRAM保持32K 。 011 =当芯片进入SPD模式，SRAM保持64K 。 100 =当芯片进入SPD模式，SRAM保持128K 。 其它 = 保留。 **注：**该位是写保护位。参考SYS_REGLCTL寄存器。
[3]	**DPDHOLDEN**	**深度掉电模式GPIO保持使能位** 0=当进入深度掉电模式，所有I/O状态为三态。 1= 当进入深度掉电模式，所有I/O保持在正常操作时的状态。在芯片唤醒后,I/O仍保持这个状态，直到用户置CLK_IOPDCTL[0]为1，才释放I/O保持状态。
[2:0]	**PDMSEL**	**掉电模式选择 (写保护)** 该寄存器写保护。请参照打开锁定顺序进行编程。 这些位控制芯片当CPU执行WFI/WFE指令时的掉电模式等级选择。 000 = 选择掉电模式 (NPD)。 001 = 选择低漏电掉电模式 (LLPD)。 010 =选择快速唤醒掉电模式 (FWPD)。 011 = 保留。 100 = 选择待机掉电模式0 (SPD)。 101 =保留。 110 = 选择深度掉电模式 (DPD)。 111 = 保留。 **注：**该位是写保护位。参考SYS_REGLCTL寄存器。

电源管理状态寄存器(CLK_PMUSTS)

寄存器	偏移	R/W	描述	复位值
CLK_PMUSTS	CLK_BA+0x94	R/W	电源管理状态寄存器	0x0000_0000

31	30	29	28	27	26	25	24
CLRWK	Reserved						
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved	ACMPWK	BODWK	LVRWK	GPDWK	GPCWK	GPBWK	GPAWK
7	6	5	4	3	2	1	0
Reserved	PINWK4	PINWK3	PINWK2	PINWK1	RTCWK	TMRWK	PINWK0

为	描述	
[31]	CLRWK	**清唤醒标志** 0 = 不清标志。 1= 清除所有唤醒标志。
[30:15]	Reserved	保留。
[14]	ACMPWK	**ACMP 唤醒标志 (只读)** 该标志表示设备被一个ACMP转变请求从待机掉电模式（SPD）唤醒。当进入SPD模式，该标志被清除。
[13]	BODWK	**BOD唤醒标志 (只读)** 该标志表示设备被一个BOD发生请求从待机掉电模式（SPD）唤醒。当进入SPD模式，该标志被清除。
[12]	LVRWK	**LVR唤醒标志 (只读)** 该标志表示设备被一个LVR发生请求从待机掉电模式（SPD）唤醒。当进入SPD模式，该标志被清除。
[11]	GPDWK	**GPD唤醒标志 (只读)** 该标志表示设备被GPD组管脚的一个跳变请求从待机掉电模式（SPD）唤醒。当进入SPD模式，该标志被清除。
[10]	GPCWK	**GPC唤醒标志 (只读)** 该标志表示设备被GPC组管脚的一个跳变请求从待机掉电模式（SPD）唤醒。当进入SPD模式，该标志被清除。
[9]	GPBWK	**GPB唤醒标志 (只读)** 该标志表示设备被GPB组管脚的一个跳变请求从待机掉电模式（SPD）唤醒。当进入SPD模式，该标志被清除。
[8]	GPAWK	**GPA唤醒标志 (只读)** 该标志表示设备被GPA组管脚的一个跳变请求从待机掉电模式（SPD）唤醒。当进入SPD

		模式，该标志被清除。
[7]	**Reserved**	保留。
[6]	**PINWK4**	**引脚唤醒4标志（只读）** 此标志表明芯片从深度掉电模式唤醒是由唤醒脚（PF.6）的转换请求的。当进入DPD模式时，将清除此标志。
[5]	**PINWK3**	**引脚唤醒3标志（只读）** 此标志表明芯片从深度掉电模式唤醒是由唤醒脚（PB.12）的转换请求的。当进入DPD模式时，将清除此标志。
[4]	**PINWK2**	**引脚唤醒2标志（只读）** 此标志表明芯片从深度掉电模式唤醒是由唤醒脚（PB.2）的转换请求的。当进入DPD模式时，将清除此标志。
[3]	**PINWK1**	**引脚唤醒1标志（只读）** 此标志表明芯片从深度掉电模式唤醒是由唤醒脚（PB.0）的转换请求的。当进入DPD模式时，将清除此标志。
[2]	**RTCWK**	**RTC唤醒标志 (只读)** 该标志表示设备被RTC闹钟请求从深度掉电模式(DPD)或者待机掉电模式（SPD）唤醒。当进入DPD模式或者SPD模式，该标志被清除。
[1]	**TMRWK**	**Timer唤醒标志 (只读)** 该标志表示设备被唤醒定时器超时请求从深度掉电模式(DPD)或待机掉电模式（SPD）唤醒。当进入DPD或SPD模式，该标志被清除。
[0]	**PINWK0**	**引脚唤醒0标志（只读）** 此标志表明芯片从深度掉电模式唤醒是由唤醒脚（GPC.0）的转换请求的。当进入DPD模式时，将清除此标志。

LDO 控制寄存器 (CLK_LDOCTL)

寄存器	偏移	R/W	描述	复位值
CLK_LDOCTL	CLK_BA+0x98	R/W	LDO 控制寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved							DPDWKSEL

位	描述	
[31:1]	Reserved	保留。
[0]	DPDWKSEL	DPD 唤醒时间选择位 0 = DPD唤醒时间在3个LIRC周期。 1 = DPD唤醒时间在2个LIRC周期。

待机掉电唤醒消抖控制寄存器(CLK_SWKDBCTL)

寄存器	偏移	R/W	描述	复位值
CLK_SWKDBCTL	CLK_BA+0x9C	R/W	待机掉电唤醒消抖控制寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved				SWKDBCLKSEL			

位	描述	
[31:4]	**Reserved**	保留。
[3:0]	**SWKDBCLKSEL**	**待机掉电唤醒消抖采样周期选择** 0000 = 每 1个时钟采样一次唤醒输入。 0001 = 每 2个时钟采样一次唤醒输入。 0010 = 每 4个时钟采样一次唤醒输入。 0011 = 每 8个时钟采样一次唤醒输入。 0100 = 每 16个时钟采样一次唤醒输入。 0101 = 每 32个时钟采样一次唤醒输入。 0110 = 每 64个时钟采样一次唤醒输入。 0111 = 每 128个时钟采样一次唤醒输入。 1000 = 每 256个时钟采样一次唤醒输入。 1001 = 每2*256个时钟采样一次唤醒输入。 1010 = 每4*256个时钟采样一次唤醒输入。 1011 = 每8*256个时钟采样一次唤醒输入。 1100 = 每16*256个时钟采样一次唤醒输入。 1101 = 每32*256个时钟采样一次唤醒输入。 1110 = 每64*256个时钟采样一次唤醒输入。 1111 = 每128*256个时钟采样一次唤醒输入。 **注:**消抖计数器时钟源是内部 10 kHz 低速RC 振荡器 (LIRC)。

GPA待机掉电唤醒控制寄存器(CLK_PASWKCTL)

寄存器	偏移	R/W	描述	复位值
CLK_PASWKCTL	CLK_BA+0xA0	R/W	GPA待机掉电唤醒控制寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							DBEN
7	6	5	4	3	2	1	0
WKPSEL				Reserved	PFWKEN	PRWKEN	WKEN

位	描述	
[31:9]	**Reserved**	保留。
[8]	**DBEN**	**GPA输入信号消抖使能位** DBEN位用来使能对应IO口消抖功能，如果输入信号脉冲宽度不能被连续的两个消抖采样周期采样到，输入信号的变化被认为是信号抖动，不会触发唤醒。消抖时钟源是内部 10 kHz低速RC 振荡器 (LIRC)。 0 = 待机掉电唤醒引脚消抖功能禁止。 1 = 待机掉电唤醒引脚消抖功能使能。 消抖功能仅对边沿触发有效。
[7:4]	**WKPSEL**	**GPA 待机掉电唤醒管脚选择** 0000 = GPA.0 唤醒功能使能。 0001 = GPA.1唤醒功能使能。 0010 = GPA.2唤醒功能使能。 0011 = GPA.3唤醒功能使能。 0100 = GPA.4唤醒功能使能。 0101 = GPA.5唤醒功能使能.。 0110 = GPA.6唤醒功能使能。 0111 = GPA.7唤醒功能使能。 1000 = GPA.8唤醒功能使能。 1001 = GPA.9唤醒功能使能。 1010 = GPA.10唤醒功能使能。 1011 = GPA.11唤醒功能使能。 1100 = GPA.12唤醒功能使能。 1101 = GPA.13唤醒功能使能。 1110 = GPA.14唤醒功能使能。

		1111 = GPA.15唤醒功能使能。
[3]	**Reserved**	保留。
[2]	**PFWKEN**	**管脚下降沿唤醒使能位** 0 = GPA 组管脚下降沿唤醒功能禁止。 1 = GPA组管脚下降沿唤醒功能使能。
[1]	**PRWKEN**	**管脚上升沿唤醒使能位** 0 = GPA组管脚上升沿唤醒功能禁止。 1 = GPA 组管脚上升沿唤醒功能使能。
[0]	**WKEN**	**待机掉电管脚唤醒使能位** 0 = GPA组管脚唤醒功能禁止。 1 = GPA 组管脚唤醒功能使能。

GPB待机掉电唤醒控制寄存器(CLK_PBSWKCTL)

寄存器	偏移	R/W	描述	复位值
CLK_PBSWKCTL	CLK_BA+0xA4	R/W	GPB 待机掉电唤醒控制寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							DBEN
7	6	5	4	3	2	1	0
WKPSEL				Reserved	PFWKEN	PRWKEN	WKEN

位	描述	
[31:9]	**Reserved**	保留。
[8]	**DBEN**	**GPB输入信号消抖使能位** DBEN位用来使能对应IO口消抖功能，如果输入信号脉冲宽度不能被连续的两个消抖采样周期采样到，输入信号的变化被认为是信号抖动，不会触发唤醒。消抖时钟源是内部 10 kHz低速RC 振荡器 (LIRC)。 0 = 待机掉电唤醒引脚消抖功能禁止。 1 = 待机掉电唤醒引脚消抖功能使能。 消抖功能仅对边沿触发有效。
[7:4]	**WKPSEL**	**GPB 待机掉电唤醒管脚选择** 0000 = GPB.0 唤醒功能使能。 0001 = GPB.1唤醒功能使能。 0010 = GPB.2唤醒功能使能。 0011 = GPB.3唤醒功能使能。 0100 = GPB.4唤醒功能使能。 0101 = GPB.5唤醒功能使能。 0110 = GPB.6唤醒功能使能。 0111 = GPB.7唤醒功能使能。 1000 = GPB.8唤醒功能使能。 1001 = GPB.9唤醒功能使能。 1010 = GPB.10唤醒功能使能。 1011 = GPB.11唤醒功能使能。 1100 = GPB.12唤醒功能使能。 1101 = GPB.13唤醒功能使能。 1110 = GPB.14唤醒功能使能。

		1111 = GPB.15唤醒功能使能。
[3]	**Reserved**	保留。
[2]	**PFWKEN**	**管脚下降沿唤醒使能位** 0 = GPB 组管脚下降沿唤醒功能禁止。 1 = GPB组管脚下降沿唤醒功能使能。
[1]	**PRWKEN**	**管脚上升沿唤醒使能位** 0 = GPB组管脚上升沿唤醒功能禁止。 1 = GPB 组管脚上升沿唤醒功能使能。
[0]	**WKEN**	**待机掉电管脚唤醒使能位** 0 = GPB组管脚唤醒功能禁止。 1 = GPB 组管脚唤醒功能使能。

GPC待机掉电唤醒控制寄存器(CLK_PCSWKCTL)

寄存器	偏移	R/W	描述	复位值
CLK_PCSWKCTL	CLK_BA+0xA8	R/W	GPC待机掉电唤醒控制寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							DBEN
7	6	5	4	3	2	1	0
WKPSEL				Reserved	PFWKEN	PRWKEN	WKEN

位	描述	
[31:9]	**Reserved**	保留。
[8]	**DBEN**	**GPC输入信号消抖使能位** DBEN位用来使能对应IO口消抖功能，如果输入信号脉冲宽度不能被连续的两个消抖采样周期采样到，输入信号的变化被认为是信号抖动，不会触发唤醒。消抖时钟源是内部 10 kHz 低速RC 振荡器 (LIRC)。 0 = 待机掉电唤醒引脚消抖功能禁止。 1 = 待机掉电唤醒引脚消抖功能使能。 消抖功能仅对边沿触发有效。
[7:4]	**WKPSEL**	**GPB 待机掉电唤醒管脚选择** 0000 = GPC.0 唤醒功能使能。 0001 = GPC.1唤醒功能使能。 0010 = GPC.2唤醒功能使能。 0011 = GPC.3唤醒功能使能。 0100 = GPC.4唤醒功能使能。 0101 = GPC.5唤醒功能使能。 0110 = GPC.6唤醒功能使能。 0111 = GPC.7唤醒功能使能。 1000 = GPC.8唤醒功能使能。 1001 = GPC.9唤醒功能使能。 1010 = GPC.10唤醒功能使能。 1011 = GPC.11唤醒功能使能。 1100 = GPC.12唤醒功能使能。 1101 = GPC.13唤醒功能使能。 1110 = GPC.14唤醒功能使能。

		1111 = GPC.15唤醒功能使能。
[3]	**Reserved**	保留。
[2]	**PFWKEN**	**管脚下降沿唤醒使能位** 0 = GPC 组管脚下降沿唤醒功能禁止。 1 = GPC组管脚下降沿唤醒功能使能。
[1]	**PRWKEN**	**管脚上升沿唤醒使能位** 0 = GPC组管脚上升沿唤醒功能禁止。 1 = GPC 组管脚上升沿唤醒功能使能。
[0]	**WKEN**	**待机掉电管脚唤醒使能位** 0 = GPC组管脚唤醒功能禁止。 1 = GPC 组管脚唤醒功能使能。

GPD待机掉电唤醒控制寄存器(CLK_PDSWKCTL)

寄存器	偏移	R/W	描述	复位值
CLK_PDSWKCTL	CLK_BA+0xAC	R/W	GPD待机掉电唤醒控制寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							DBEN
7	6	5	4	3	2	1	0
WKPSEL				Reserved	PFWKEN	PRWKEN	WKEN

位	描述	
[31:9]	**Reserved**	保留。
[8]	**DBEN**	**GPD输入信号消抖使能位** DBEN位用来使能对应IO口消抖功能，如果输入信号脉冲宽度不能被连续的两个消抖采样周期采样到，输入信号的变化被认为是信号抖动，不会触发唤醒。消抖时钟源是内部 10 kHz 低速RC 振荡器 (LIRC)。 0 = 待机掉电唤醒引脚消抖功能禁止。 1 = 待机掉电唤醒引脚消抖功能使能。 消抖功能仅对边沿触发有效。
[7:4]	**WKPSEL**	**GPD 待机掉电唤醒管脚选择** 0000 = GPD.0 唤醒功能使能。 0001 = GPD.1唤醒功能使能。 0010 = GPD.2唤醒功能使能。 0011 = GPD.3唤醒功能使能。 0100 = GPD.4唤醒功能使能。 0101 = GPD.5唤醒功能使能。 0110 = GPD.6唤醒功能使能。 0111 = GPD.7唤醒功能使能。 1000 = GPD.8唤醒功能使能。 1001 = GPD.9唤醒功能使能。 1010 = GPD.10唤醒功能使能。 1011 = GPD.11唤醒功能使能。 1100 = GPD.12唤醒功能使能。 1101 = GPD.13唤醒功能使能。 1110 = GPD.14唤醒功能使能。

		1111 = GPD.15唤醒功能使能。
[3]	**Reserved**	保留。
[2]	**PFWKEN**	**管脚下降沿唤醒使能位** 0 = GPD 组管脚下降沿唤醒功能禁止。 1 = GPD组管脚下降沿唤醒功能使能。
[1]	**PRWKEN**	**管脚上升沿唤醒使能位** 0 = GPD组管脚上升沿唤醒功能禁止。 1 = GPD 组管脚上升沿唤醒功能使能。
[0]	**WKEN**	**待机掉电管脚唤醒使能位** 0 = GPD组管脚唤醒功能禁止。 1 = GPD 组管脚唤醒功能使能。

GPIO待机掉电控制寄存器(CLK_IOPDCTL)

寄存器	偏移	R/W	描述	复位值
CLK_IOPDCTL	CLK_BA+0xB0	R/W	GPIO待机掉电控制寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved							IOHR

位	描述	
[31:1]	**Reserved**	保留。
[0]	**IOHR**	**GPIO 保持释放** 当GPIO进入深度掉电模式或者待机掉电模式，所有I/O状态保持正常操作时的状态。在芯片从深度掉电模式或者待机掉电模式唤醒后，I/O仍然保持状态直到用户设置该位释放I/O保持的状态。 **注：** 该位由硬件自动清0。

6.4 真随机数发生器(TRNG)

6.4.1 概述

真实随机数发生器（TRNG）用于从物理现象中提取随机性。

6.4.2 特征

- 每秒产生800个随机位

6.4.3 框图

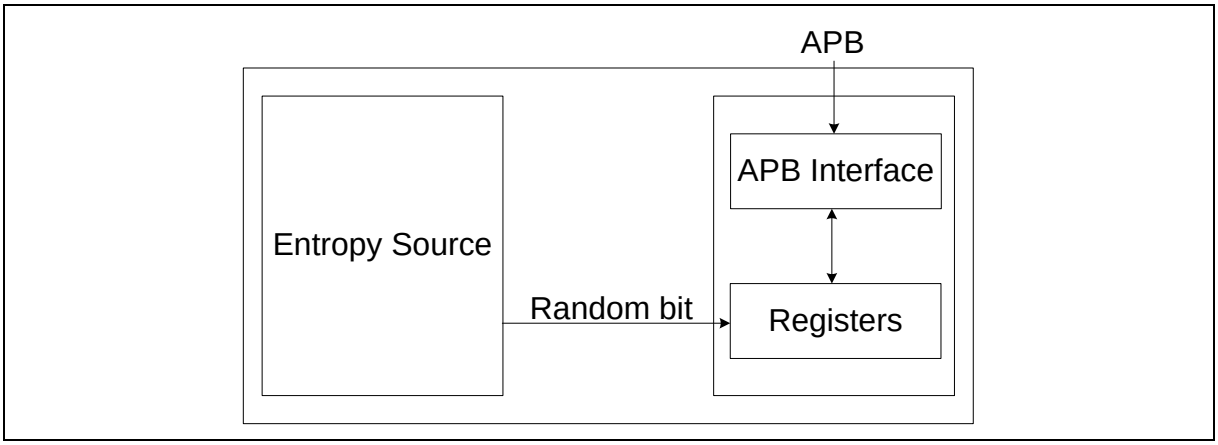


图 6.4-1 真实随机数发生器框图

6.4.4 基本配置

- 时钟源配置
  - 时钟源来自LXT:
    - 在LXTEN (CLK_PWRCTL[1])中使能32kHz 时钟
  - 时钟源来自LIRC:
    - 在LIRC32KEN (RTC_LXTCTL[0])和C32KS (RTC_LXTCTL[7])中使能32 kHz 时钟
  - 在TRNGCKEN (CLK_APBCLK1[25])中使能TRNG外设时钟。

6.4.5 功能描述

当TRNG由ACT（TRNG_ACT [7]）激活，并由TRNGEN (TRNG_CTL[0])使能时，TRNG可提供随机数。如果DVIF为0，TRNG_DATA寄存器返回值0x0。当DVIF(TRNG_CTL[1])置1时，随机位只能从DATA (TRNG_DATA[7:0])中读取一次。读取TRNG_DATA寄存器后，DVIF和TRNG_DATA将设置为0。如果使能DVIEN (CTRN_CTL[6])，当DVIF置位时，TRNG将生成中断。

获取真实随机数的编程步骤如下：

1. 在CLKPSC (TRNG_CTL[5:2])中选择TRNG外设的时钟频率
2. 使能ACT (TRNG_ACT[7]) 位

3. 等待直到READY (TRNG_CTL[7])位被置1
4. 使能TRNGEN (TRNG_CTL[0])位
5. 等待直到DVIF (TRNG_CTL[1])位被置1
6. 读TRNG_DATA寄存器获取随机数
7. 重复步骤5和6获取更多的随机数

6.4.6 寄存器映射

R: 只读, **W**: 只写, **R/W**: 读写

寄存器	偏移	R/W	描述	复位值
TRNG 基地址: **TRNG_BA = 0x400B_9000**				
TRNG_CTL	TRNG_BA+0x00	R/W	TRNG控制寄存器和状态	0x0000_0000
TRNG_DATA	TRNG_BA+0x04	R	TRNG数据寄存器	0x0000_0000
TRNG_ACT	TRNG_BA+0x0C	R/W	TRNG激活寄存器	0x0000_0002

6.4.7 寄存器描述

TRNG控制寄存器和状态 (TRNG_CTL)

寄存器	偏移	R/W	描述	复位值
TRNG_CTL	TRNG_BA+0x00	R/W	TRNG控制寄存器和状态	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
READY	DVIEN	CLKPSC				DVIF	TRNGEN

位	描述	
[31:8]	Reversed	保留。
[7]	READY	**随机数发生器准备（只读）** 在ACT (TRNG_ACT[7]) 位被置1，在延迟90us~120us 后READY 将置1。 0 = TRNG没有准备好或者没激活。 1 = TRNG已准备好可以使用。
[6]	DVIEN	**数据有效中断使能位** 0 = 中断禁止。 1 = 中断使能。
[5:2]	CLKPSC	**时钟预分频** CLKPSC的选定值是外设时钟频率范围，CLKPSC设置必须高于或等于实际外设时钟频率（用于正确的随机位生成）。要更改CLKPSC设置，需将TRNGEN位设置为0，更改CLKPSC，再将TRNGEN位设置为1以重新启用TRNG。 0000 = 80 ~ 100 MHz。 0001 = 60 ~ 80 MHz。 0010 = 50 ~60 MHz。 0011 = 40 ~50 MHz。 0100 = 30 ~40 MHz。 0101 = 25 ~30 MHz。 0110 = 20 ~25 MHz。 0111 = 15 ~20 MHz。

		1000 = 12 ~15 MHz。 1001 = 9 ~12 MHz。 1010 = 7 ~9 MHz。 1011 = 6 ~7 MHz。 1100 = 5 ~6 MHz。 1101 = 4 ~5 MHz。 1111 = 保留。
[1]	**DVIF**	**数据有效 (只读)** 0 = 数据无效。读TRNG_ DATA返归为0x00000000。 1= 数据有效。可以从DATA(TRNG_DATA[7:0])中读取一个有效的随机数。 **注:** 该位通过读TRNG_DATA 清0。
[0]	**TRNGEN**	**随机数发生器使能位** 该位只能在ACT (TRNG_ACT[7]) 置1，且READY (TRNG_CTL[7]) 为1时才能置1。 0 = TRNG禁止。 1 = TRNG使能。 **注:** TRNGEN 是数字部分使能位，当TRNG 不需要产生随机数时，TRNGEN 位和ACT (TRNG_ACT[7]) 位应该置0以降低功耗。

TRNG数据寄存器 (TRNG_DATA)

寄存器	偏移	R/W	描述	复位值
TRNG_DATA	TRNG_BA+0x04	R	TRNG数据寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
DATA							

位	描述	
[31:8]	Reserved	保留。
[7:0]	DATA	**生成的随机数数据(只读)** DATA 存放了通过TRNG生成的随机数数据，该域只能读一次。

TRNG激活寄存器(TRNG_ACT)

寄存器	偏移	R/W	描述	复位值
TRNG_ACT	TRNG_BA+0x0C	R/W	TRNG激活寄存器	0x0000_0002

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
ACT	Reserved						

位	描述	
[31:8]	**Reserved**	保留。
[7]	**ACT**	**随机数生成器激活** 在使能ACT位，它将激活TRNG模块并且等待READY（TRNG_CTL[7]）位变成1。 0 = TRNG没激活。 1 = TRNG激活。 **注:** ACT是使能模拟部分，当TRNG不要求产生随机数时，TRNGEN (TRNG_CTL[0]) 位和ACT位应该置0以降低功耗。
[6:0]	**Reserved**	保留。

6.5 存储控制器 (FMC)

6.5.1 概述

FMC配备了128K/256K/512K字节片上内存，用于存储应用程序和存储基于应用的一些数据，数据Flash可配置大小。因此，总的应用存储器 (APROM) 大小是128K/256K/512K字节。一个用户配置区，用于系统初始化，一个4K字节的引导存储器(LDROM)，用于在系统编程(ISP)功能。一个4K字节的安全保护存储器(SPROM)可以隐藏用户的程序。一个3K字节的单次编程存储器(OTP)用于存储需要单次编程的数据。一个32K的Boot Loader, 内建ISP功能和安全启动功能。一个8K字节Boot Loader内建安全引导功能。一个零等待周期的4K高速缓存cache，用于提高对flash的访问性能。支持在应用编程(IAP)，更新flash程序后，执行用户程序之间切换时，无需复位。

6.5.2 特征

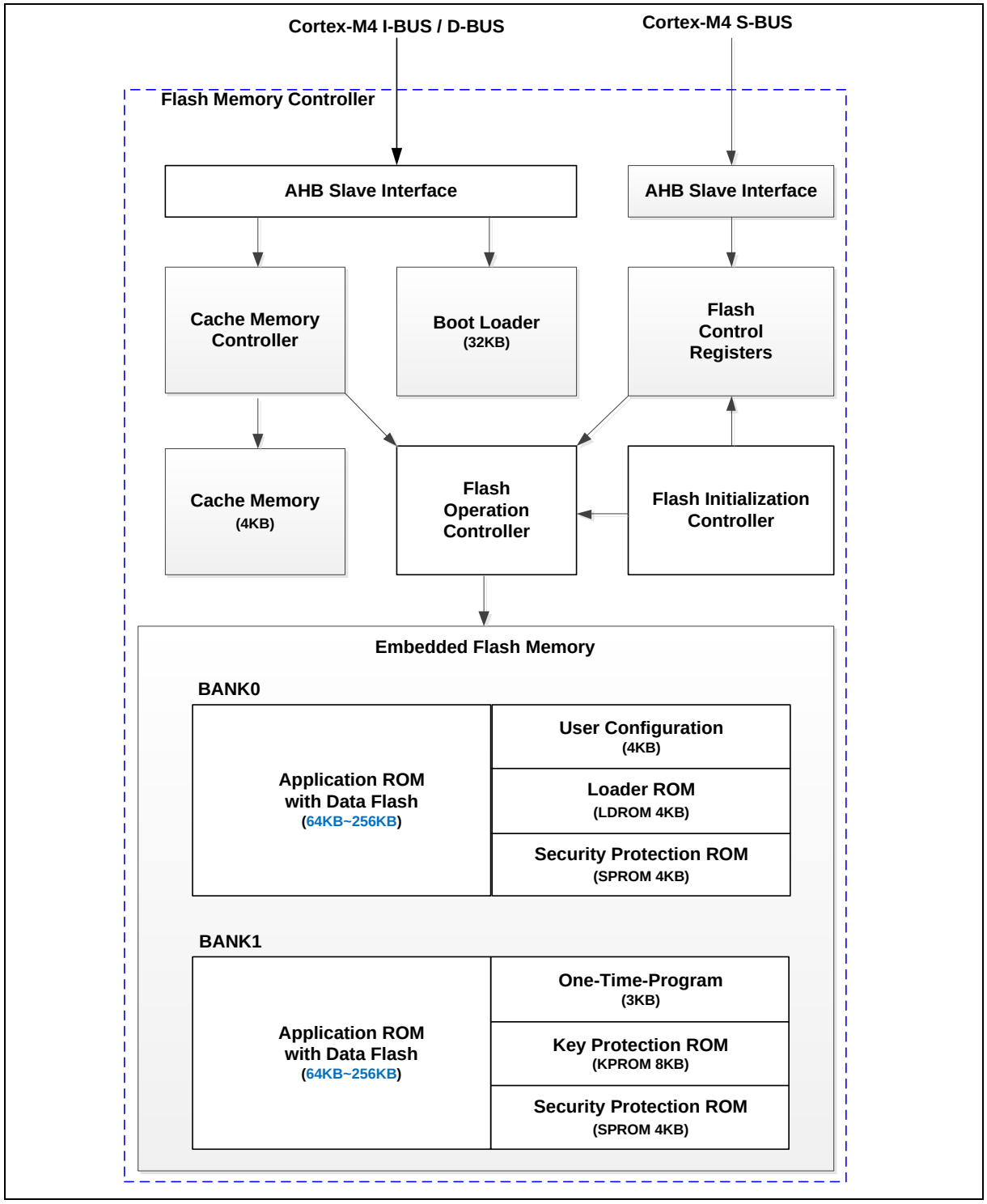
- 支持双bank Flash用于安全固件升级
- 支持128K/256K字节应用程序存储空间(APROM)
- 支持512K字节应用程序存储空间
- 支持4K字节引导存储器(LDROM)
- 支持4K字节安全保护存储器(SPROM) 用于隐藏用户的程序
- 支持双bank Flash 中镜像SPROM，在写入其他ROM时读取其他SPROM代码
- 支持4个XOM（执行存储器）区域来隐藏APROM中的用户程序
- 支持可配置大小的数据Flash
- 支持16字节用户配置区，用于控制系统初始化
- 支持3K字节单次编程存储器 (OTP)
- 对所有内嵌的Flash，支持4K字节页擦除
- APROM支持区块擦除，除了XOM区域
- Boot Loader内建了在系统编程(ISP)功能
- 支持安全启动功能用于代码存储和可靠性需求
- 支持安全密匙保护功能，用于保护APROM, LDROM, SPROM, 用户配置区和KPROM
- 支持 32-位/64-位和多字Flash编程功能
- 支持快速Flash编程校验功能
- 支持 CRC32 校验和计算功能
- 支持Flash位全为1校验功能
- 支持在系统编程(ISP) /在应用编程(IAP)来更新片上Flash
- 支持cache存储器来提高Flash访问效率和减少功耗
- 支持自动调节Flash访问周期功能以优化Flash的访问执行效率

FMC 特征	M48xID/M48xGA	M48xGC/M48xE8
双-bank Flash	●	
128/256K字节 APROM	●	●
512K字节 APROM	●	
4K字节 LDROM	●	●
4K字节 SPROM	●	
4个XOM 区域		●
可配置数据Flash大小	●	●
16字节用户配置区(UCFG)	●	●
3K字节 OTP	●	●
8K字节 KPROM	●	●
4K字节页擦除	●	●
APROM 可以块擦除，除了XOM	●	●
32K字节Boot Loader内建ISP功能	●	
8K字节 Boot Loader内建ISP功能		●
AES 安全启动功能	●	
ECC 安全启动功能		●
安全密钥保护功能用于保护APROM和LDROM	●	●
安全密钥保护功能用于保护UCFG和KPROM	●	●
安全密钥保护功能用于保护SPROM	●	
32-位/64-位和多字节Flash编程功能	●	●
快速Flash 编程校验功能	●	●
CRC32校验计算功能	●	●
4K字节cache内存	●	●
自动调整Flash访问周期功能		●
应用编程功能(IAP)	●	●
在复位上升沿通过PF.0从boot loader启动	●	

表 6.5-1 不同芯片型号 FMC 特征比较表

6.5.3 框图

存储控制器(FMC)包括AHB从接口，cache内存控制器，boot loader，Flash 控制寄存器，Flash初始化控制器，Flash操作控制和片上Flash。 Flash存储控制器框图如下所示。



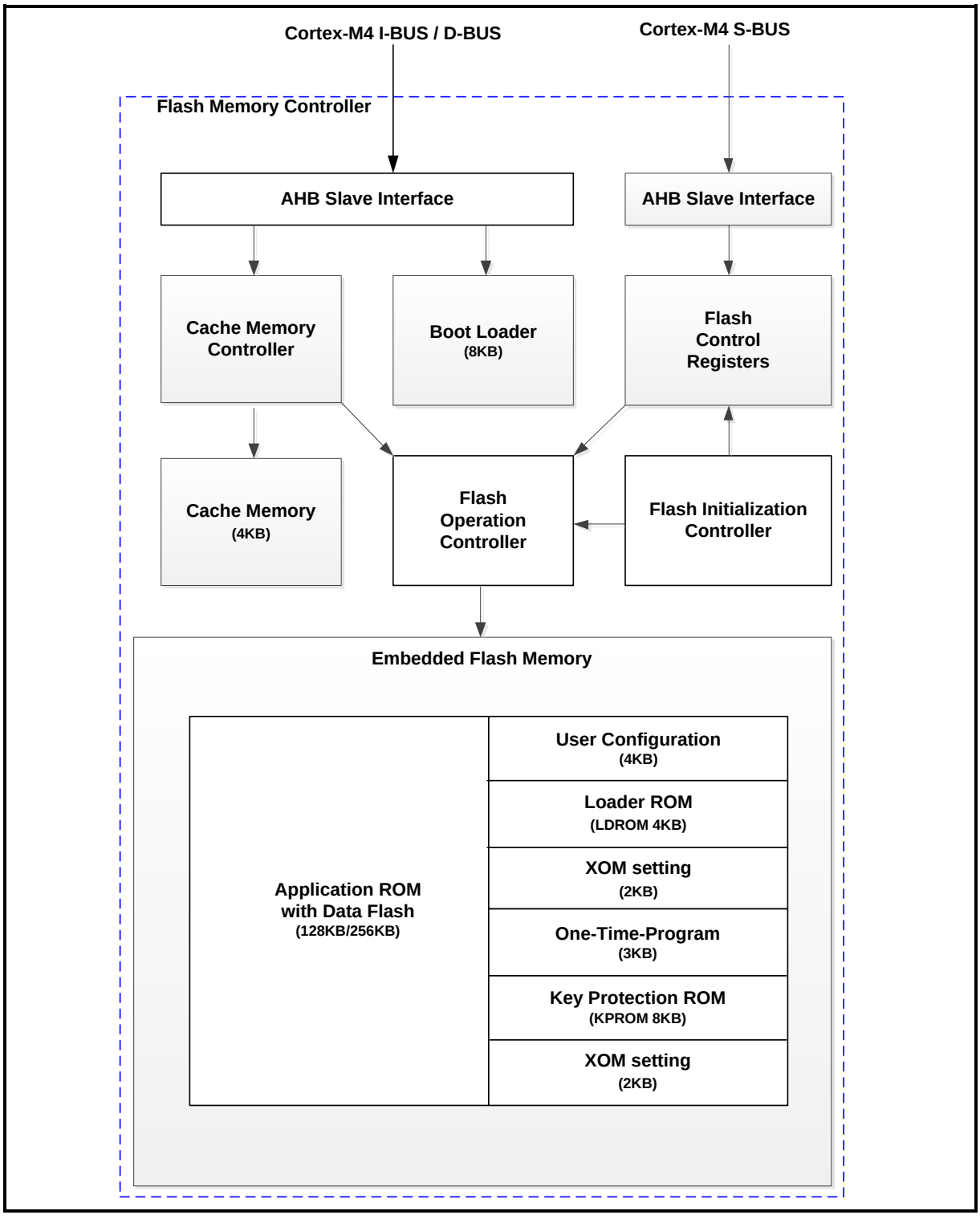


图 6.5-1Flash 存储控制器框图

AHB 从接口

在Flash存储控制器中有两个AHB从接口，一个是来自Cortex®-M4 I-总线与D-总线，用于指令和数据读取；另一个是来自Cortex®-M4 S-总线，用于Flash控制寄存器的访问，也用于ISP寄存器的访问。

Cache 存储控制器

一个零等待周期的4K字节cache，位于Cortex®-M4 CPU和片上Flash之间。Cache存储控制器提高了Flash的访问效率并降低了功耗。

Boot Loader

Boot Loader 大小是8K字节/32K字节，包括了内建 ISP功能来更新片上Flash。 Boot Loader的内容是只读的，不可编程。

Flash 控制寄存器

所有的ISP控制和状态寄存器都在Flash控制寄存器中。详细的寄存器描述会在寄存器介绍章节介绍。

Flash 初始化控制器

当芯片上电或复位，Flash初始化控制器将开始自动访问Flash，并且检测Flash的稳定, 重载用户配置内容到Flash控制寄存器用于系统初始化。

Flash 操作控制器

对Flash操作，例如Flash 擦除，Flash 编程和读Flash，有明确的控制时序。Flash操作控制器在收到Cache存储控制器, Flash 控制寄存器和Flash初始化控制器的请求后, 产生了这些控制时序。

片上 Flash 存储

片上Flash存储器是用于存储用户应用程序和参数的。它包括了用户配置区，4K字节的LDROM ,两个 4K字节 SPROM, 2两个4K字节XOM设置页，3K字节OTP，8K字节KPROM和包含数据Flash的128K字节/256K字节/512K字节APROM。页擦除的Flash大小是4K字节，最小可编程位大小是32位。

6.5.4 功能描述

FMC功能包括存储器组织，启动选择，安全启动，IAP，ISP，片上Flash编程，和checksum 计算。在存储器组织中也介绍了Flash存储器映射和系统存储器映射。

6.5.4.1 存储器组织

FMC存储包含了片上Flash和boot loader。片上存储器是可编程的，包括APROM，LDROM，SPROM，XOM设置页，数据Flash和用户配置区，OTP 和 KPROM。Boot Loader是一个掩膜的存储器带ISP启动代码，支持固件更新，启动控制，安全控制和固件执行。地址映射包括Flash存储映射和5个系统地址映射：支持IAP功能的LDROM，不支持IAP功能的LDROM，支持IAP功能的APROM，不支持IAP功能的APROM，以及支持IAP功能的Boot Loader。

BANK	Flash 存储区块	地址范围
0	APROM为512 Kbytes	0x00_0000 ~ 0x03_ffff
	APROM为256 Kbytes	0x00_0000 ~ 0x01_ffff
	APROM为128 Kbytes	0x00_0000 ~ 0x00_ffff
	用户配置区	0x30_0000 ~ 0x30_000f
	LDROM	0x10_0000 ~ 0x10_0fff
	SPROM	0x20_0000 ~ 0x20_0fff
1	APROM为512 Kbytes	0x04_0000 ~ 0x07_ffff
	APROM为256 Kbytes	0x02_0000 ~ 0x03_ffff
	APROM为128 Kbytes	0x01_0000 ~ 0x01_ffff
	OTP	0x31_0000 ~ 0x31_0bff
	KPROM-KEY	0x30_1000 ~ 0x30_11ff
	KPROM - KPCNTROM	0x30_1200 ~ 0x30_1fff

Flash存储区块	地址范围
APROM为256 Kbytes	0x00_0000 ~ 0x01_ffff
APROM为128 Kbytes	0x00_0000 ~ 0x00_ffff
用户配置	0x30_0000 ~ 0x30_000f
LDROM	0x10_0000 ~ 0x10_0fff
XOM设置	0x20_0000 ~ 0x20_0fff
OTP	0x31_0000 ~ 0x31_0bff
KPROM-KEY	0x30_1000 ~ 0x30_11ff
KPROM - KPCNTROM	0x30_1200 ~ 0x30_1fff

表 6.5-2 区块地址范围

6.5.4.2 LDROM APROM和Data Flash

LDROM 是用于通过引导程序，来执行在系统编程(ISP)的功能。LDROM是4KB的片上Flash存储器，Flash地址是从0x0010_0000到0x0010_0FFF。APROM是用户应用程序的主要存储器。APROM 大小是128K字节/256K字节/512K字节。数据Flash是用于存储应用参数的（不是指令）。数据FLASH与APROM共享存储空间，大小可配置，数据Flash的基地址由寄存器DFBA (CONFIG1[19:0])设定。所有片上Flash存储器, 页擦除的大小是4KB。

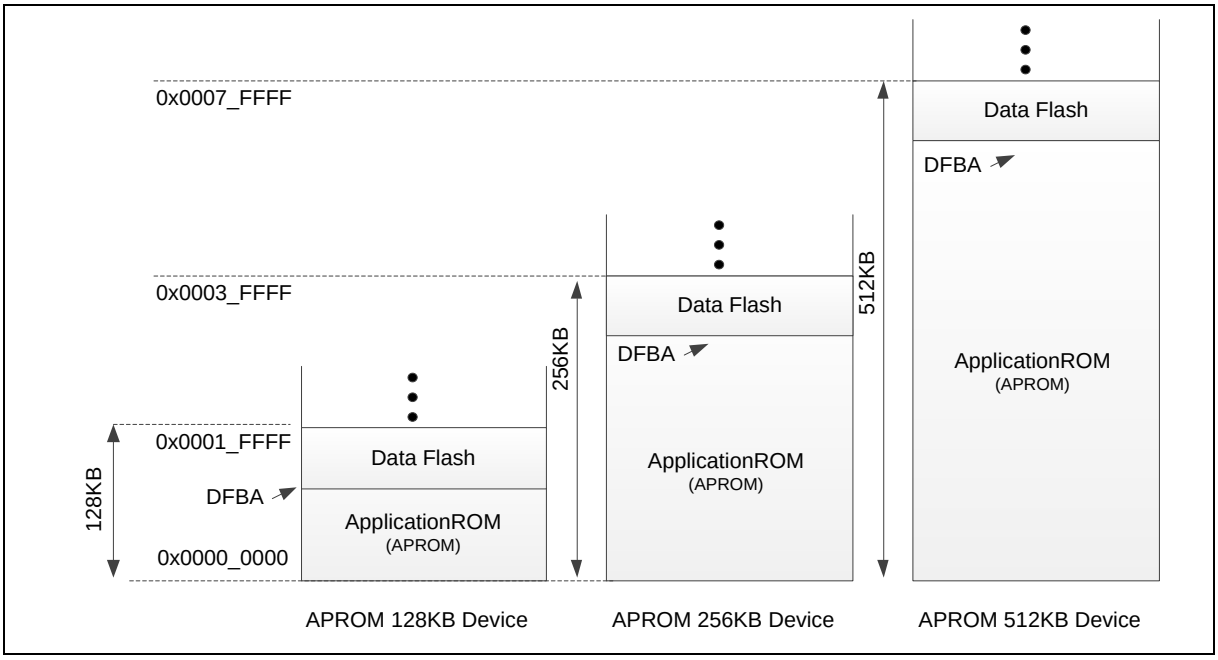


图 6.5-2 数据 Flash 与 APROM 共享存储空间(128K 字节/256K 字节/512K 字节)

6.5.4.3 用户配置区

用户配置区是内部可编程的配置区域，用于启动选项，比如Flash安全锁，启动选择，欠电压电平设置和数据Flash基地址。用户配置区的作用类似保险丝用于上电时的缺省设置。在上电的时候，用户配置区设定会加载到相应的控制寄存器。用户可以通过不同的应用需求来设定寄存器。用户配置区可以通过ISP方式更新，位于地址0x0030_0000，有四个32位字寄存器 (CONFIG0, CONFIG1, CONFIG2 和 CONFIG3)。用户配置所有的更改将在系统重启后生效。用户配置区也是flash结构有擦写寿命限制。

注:

1. 推荐用ICP工具来编辑用户配置区（CONFIG0, CONFIG1, CONFIG2 和 CONFIG3），如果用固件更新，设定的值在系统重启之前需要检查。

CONFIG0 (地址 = 0x0030_0000)

31	30	29	28	27	26	25	24
CWDTEN[2]	CWDTPDEN	Reserved		CFGXT1	Reserved		
23	22	21	20	19	18	17	16
CBOV			CBORST	CBODEN	Reserved		
15	14	13	12	11	10	9	8
Reserved				ICELOCK	CIOINI	Reserved	
7	6	5	4	3	2	1	0
CBS		MBS	CWDTEN[1:0]		Reserved	LOCK	DFEN

位	描述	
[31]	**CWDTEN[2]**	**看门狗定时器硬件使能位** 当看门狗定时器硬件使能功能被使能，上电后，看门狗使能位WDTEN (WDT_CTL[7])和看门狗复位使能位RSTEN (WDT_CTL[1])会自动为1。看门狗的时钟源为LIRC , LIRC不能被禁用。 **CWDTEN[2:0]**在寄存器 CONFIG0的相应位[31][4][3]， 011 = WDT 看门狗硬件使能功能有效。除芯片进入掉电模式以外，WDT时钟一直开启。当芯片进入掉电模式，如果寄存器CWDTPDEN=0， WDT 时钟是一直开启的，如果寄存器CWDTPDEN =1，WDT时钟由寄存器LIRCEN (CLK_PWRCTL[3])控制。请参考寄存器CWDTPDEN的位描述。 111 = WDT 硬件使能功能无效。 其他 = WDT 硬件使能功能有效。WDT时钟一直开启。
[30]	**CWDTPDEN**	**看门狗时钟掉电使能** 0 = 当芯片进入掉电模式时，看门狗时钟保持使能。 1 = 当芯片进入掉电模式时，看门狗时钟由LIRCEN (CLK_PWRCTL[3])控制。 **注:**该位只有在CWDTEN[2:0]=011时才起作用。
[29:28]	**Reserved**	保留。
[27]	**CFGXT1**	**HXT 模式选择** 0 =HXT工作用外部时钟模式。PF.3配置为外部时钟输入脚。 1 = HXT工作为晶振模式。PF.2和PF.3配置为外部高速晶振(HXT)管脚。 **注：**当CFGXT1 = 0, PF.3多功能管脚作为GPIO模式，XT1_IN的DC特征和GPIO一样。
[26:24]	**Reserved**	保留。

[23:21]	**CBOV**	**欠压电压选择** 000 =欠压电压为1.6V。 001 =欠压电压为1.8V。 010 =欠压电压为2.0V。 011 =欠压电压为2.2V。 100 =欠压电压为2.4V。 101 =欠压电压为2.6V。 110 =欠压电压为2.8V。 111 =欠压电压为3.0V。
[20]	**CBORST**	**欠压复位使能位** 0 =上电后，使能欠压复位。 1 =上电后，禁用欠压复位。
[19]	**CBODEN**	**欠压侦测器使能位** 0= 上电后，欠压侦测器使能。 1=上电后，欠压侦测器禁止。
[18:12]	**Reserved**	保留。
[11]	**ICELOCK**	**ICE 锁死位** 该位仅用于禁止 ICE 功能。用户可以使用它配合LOCK (CONFIG0[1]) 位或 ALOCK (CONFIG2[7:0]) 位来增加系统的安全等级。 0 = ICE 功能禁止。 1 = ICE 功能使能。
[10]	**CIOINI**	**I/O 初始状态选择** 0 = 上电后所有GPIO默认为准双向模式。 1 = 上电后所有GPIO默认为三态模式。
[9:8]	**Reserved**	保留。
[7:6]	**CBS**	**芯片启动选择** 当CBS[0] = 0，LDROM的基地址映射到0x100000, APROM的基地址映射到0x0. 用户既可以访问APROM也可以访问LDROM，无需启动切换。换句话说，如果是IAP模式，存放在LDROM与APROM的代码可以互相调用。 当MBS =0，CBS[0]值会被强制为0。 00=由LDROM启动支持IAP功能。 01=由LDROM启动不支持IAP功能。 10=由APROM启动支持IAP功能。 11=由APROM启动不支持IAP功能。 **注：** 当CBS[0] = 1并且MBS = 1，BS (FMC_ISPCTL[1])只用于启动切换的控制。 当CBS[0] = 0或MBS = 0，VECMAP (FMC_ISPSTS[23:9])只用于地址0x0~0x1ff重新映射。

[5]	**MBS**	**Boot Loader 启动选择** 0 = 从Boot loader启动，CBS设定无效。 1 = 依据CBS设定数值，从APROM或LDROM启动。 **注:** 当CBS[0] = 1并且MBS = 1，BS (FMC_ISPCTL[1])只用于启动切换的控制。 当CBS[0] = 0或MBS = 0，VECMAP (FMC_ISPSTS[23:9])只用于地址0x0~0x1ff重新映射。
[4:3]	**CWDTEN**	**看门狗定时器硬件使能位** 当看门狗定时器硬件使能功能有效，上电后，看门狗使能位WDTEN (WDT_CTL[7])和看门狗复位使能位RSTEN (WDT_CTL[1])会自动为1。看门狗的时钟源为LIRC，LIRC不能被禁用。 **CWDTEN[2:0]**在寄存器 CONFIG0的相应位[31][4][3]， 011 = WDT 看门狗硬件使能功能有效。除芯片进入掉电模式以外，WDT时钟一直开启。当芯片进入掉电模式，如果寄存器CWDTPDEN=0， WDT 时钟是一直开启的，如果寄存器CWDTPDEN =1，WDT时钟由寄存器LIRCEN (CLK_PWRCTL[3])控制。请参考寄存器CWDTPDEN的位描述。 111 = WDT 硬件使能功能无效。 其他 = WDT 硬件使能功能有效。WDT时钟一直开启。
[2]	**Reserved**	保留。
[1]	**LOCK**	**加密锁控制** 0 = 加密FLASH数据。 1 = 如果ALOCK(CONFIG2[7:0]) 和 SBLOCK (CONFIG2[15:8]) 不是 0x5A.，解除Flash 数据加密。 当Flash数据被LOCK加密，用户可以在Lock效果表中找到FMC的Lock效果。
[0]	**DFEN**	**数据FLASH使能位** 当DFEN=0时 APROM与数据Flash共享，数据Flash的基地址由DFBA (CONFIG1[19:0])设定。 0 = 使能数据FLASH。 1 = 禁用数据FLASH。

注:如果是保留位，必须是1

CONFIG1 (地址 = 0x0030_0004)

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved				DFBA			
15	14	13	12	11	10	9	8
DFBA							
7	6	5	4	3	2	1	0
DFBA							

位	描述	
[31:20]	**Reserved**	保留。
[19:0]	**DFBA**	**数据Flash基地址** 只有当DFEN (CONFIG0[0])=0时，该寄存器才工作。如果DFEN (CONFIG0[0])=0，数据Flash基地址由用户定义。因为片上FLASH擦除单位为4K字节，所以强制bit11-0位为0。

注:.如果是保留位，必须是1。

CONFIG2 (地址 = 0x0030_0008)

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
SBLOCK							
7	6	5	4	3	2	1	0
ALOCK							

位	描述	
[31:16]	**Reserved**	保留。
[15:8]	**SBLOCK**	**安全启动锁控制** 0x5A = 如果 LOCK (CONFIG0[1]) 是 1 且 ALOCK (CONFIG2[7:0]) 是 0x5A，安全启动功能禁止，存储器内容是未锁状态， 其他 = 安全启动功能使能且 LOCK/ALOCK 也是锁死的. 当安全启动功能使能，FMC会从Boot Loader启动，带IAP模式 (也就是CONFIG0中CBS[0]的值被指定为0) 。然后执行安全启动。 当SBLOCK ≠0x5A，连ICE，仅芯片擦除的ISP命令可以通过ICE执行，当flash数据被SBLOCK位锁定，用户必须在上锁效果表中查找FMC上锁的效果。
[7:0]	**ALOCK**	**高级安全锁控制** 0x5A = 如果LOCK (CONFIG0[1]) 是 1，Flash存储器内容是未锁的。 其他 = Flash 存储器内容是锁定的。 当Flash数据被ALOCK位锁定，用户必须在上锁效果表中查找FMC上锁的效果。

注:.如果是保留位，必须是1。

CONFIG3 (地址 = 0x0030_000C)

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved		SPIMPSL		Reserved	UART1PSL		

位	描述	
[31:6]	**Reserved**	保留。
[5:4]	**SPIMPSL**	**Boot Loader SPIM 多功能管脚选择** 00: SPIM_CLK (PA.2), SPIM_SS (PA.3), SPIM_MISO (PA.1), SPIM_MOSI (PA.0)。 01: SPIM_CLK (PC.2), SPIM_SS (PC.3), SPIM_MISO (PC.0), SPIM_MOSI (PC.1)。 10: SPIM_CLK (PG.12), SPIM_SS (PG.11), SPIM_MISO (PG.13), SPIM_MOSI (PG.14)。 11: SPIM_CLK (PE.4), SPIM_SS (PE.5), SPIM_MISO (PE.3), SPIM_MOSI (PE.2)。 **注：** 当芯片复位，默认值由CONFIG3[5:4]设定。
[3:2]	**Reserved**	保留。
[1:0]	**UART1PSL**	**Boot Loader UART1多功能管脚选择** 00: UART1_TXD (PB.3), UART1_RXD (PB.2)。 01: UART1_TXD (PA.9), UART1_RXD (PA.8)。 10: UART1_TXD (PA.3), UART1_RXD (PA.2)。 11: UART1_TXD (PB.7), UART1_RXD (PB.6)。 **注：** 当芯片复位，默认值由CONFIG3[1:0]设定。

注:.如果是保留位，必须是1。

6.5.4.4 安全程序空间 (SPROM)

该芯片提供安全程序空间用于用户存储安全应用方面的指令。这块安全空间是4K字节，地址是0x20_0000 ~ 0x20_0FFF。这块空间只能被“页擦除”命令擦除。SPROM 最后一个字节(地址: 0x0020_0FFF) 是用来标识该空间是否加密的。如图6.5-3. SPROM (1) SPROM 非安全模式 (最后一个字节是 0xFF), (2)SPROM 调试模式 (最后一个字节是0xAA), (3) SPROM 安全模式 (最后一个字节不是0xFF 或 0xAA)。

为了在安全模式隐藏SPROM代码，CPU仅可以执行从SPROM获取指令，然后获取SPROM数据，任何CPU直接数据访问SPROM，没有执行从SPROM获取指令，CPU都会得到全部是相同(0xffff_ffff).的数据。为了安全保护，当Cortex®-M4 ICE (In-Circuit-Emulator) 正在SPROM安全代码调试，任何ICE在SPROM程序设置的断点和通过ICE或PDMA取出SPROM的安全代码都是非法的。

SPROM安全标志显示在SCODE (FMC_ISPSTS[31])，该标志在SPROM页擦除操作完成后被清0。如果SPROM的最后一个字节不是0xFF，在flash初始化的时候该标志被置位。此外用户也可以通过写1到寄存器FMC_ISPSTS[31]来置位SCODE 。在执行SPROM页擦除之前，SPUEN (FMC_ISPCTL[2])必须置1，且在SPROM安全模式ISPDAT[31:0]寄存器必须设置为0x0055AA03。

为了实现安全固件更新，我们分配一个SPROM页带安全应用的相同内容在每个bank中，以确保安全应用可以在任何bank中执行。当同时更新固件代码和访问SPROM，FMC会在任何空闲的bank中返回SPROM的指令。请注意当在任何SPROM执行ISP flash页擦除命令功能或flash数据编程功能，所有的SPROM会被自动地擦除或编程。

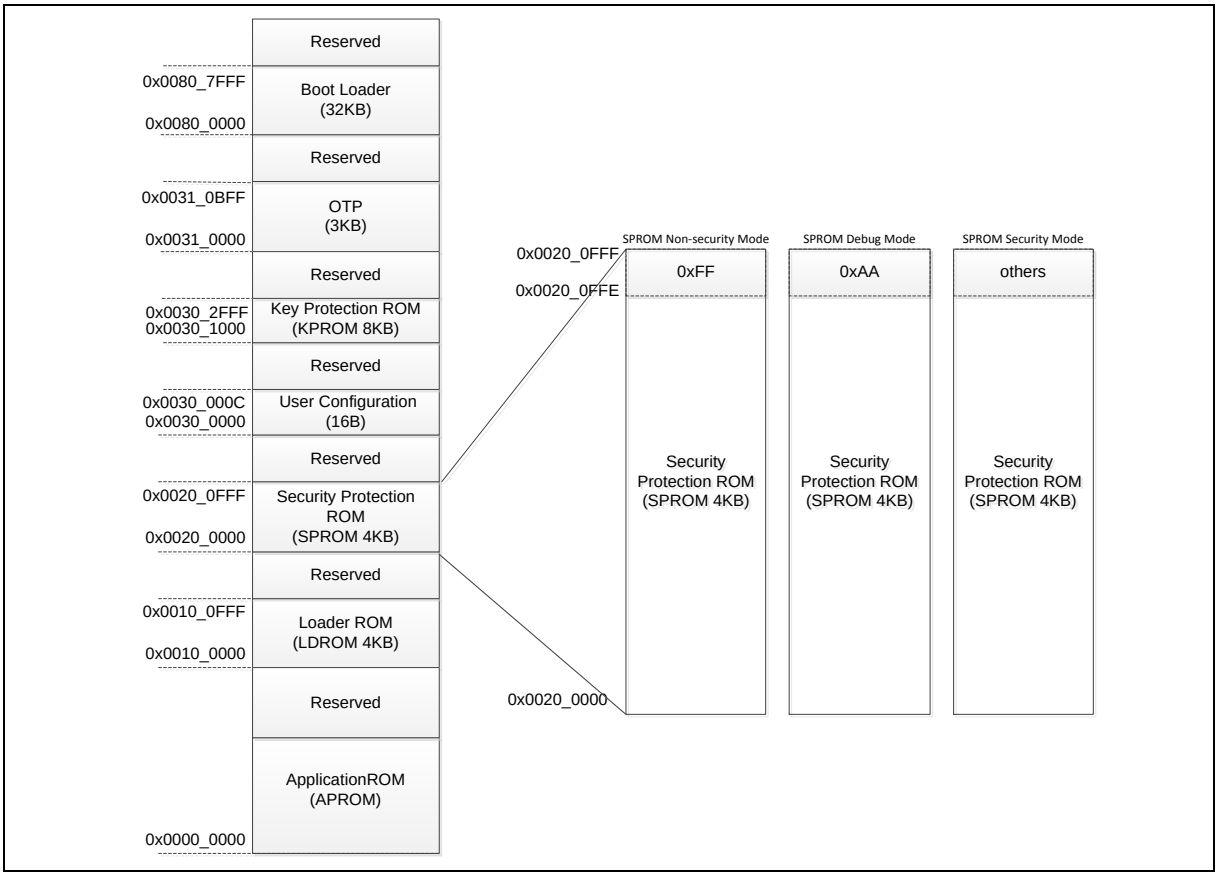


图 6.5-3SPROM 安全模式

6.5.4.5 执行存储器(XOM)

执行存储器（XOM）用于存储安全应用程序的指令，这些指令不允许通过AHB总线进行数据存取。APROM最多有四块XOM区域。要定义新的XOM区域，用户必须通过XOM设置页面完成XOM设置（0x20_0000 ~ 0x20_0008 为XOM区域0; 0x20_0010 ~ 0x20_0018为XOM区域1 ; 0x20_0020 ~ 0x20_0028为XOM区域2 ; 0x20_0030 ~ 0x20_0038为XOM区域3）通过系统编程（ISP）功能。用户设置XOM区域时，首先设置XOM的基地址然后设置XOM的页大小，如图6.5-4所示。用户不能将XOM的区域设置在芯片启动的地方。用户也不能将XOMs设置成彼此空间有重叠，因为这会导致XOM不能正常工作。设置并恢复Flash初始化后，用户可以从FMC_XOMSTS中检查XOM状态，并检查FMC_XOMR0STS～FMC_XOMR3STS的每个XOM范围（即基地址和大小）。用户可以执行命令0去读XOM设定页和检查XOM状态。用户需要在设置或擦除XOM后进行芯片复位。

使用XOM设置页面和XOM区域时，用户必须注意以下限制：

- 所有XOM区域都必须在APROM的范围内，且必须是页对齐。
- 完成XOM设置后，任何XOM区域都不能编程。
- 运行期间不能更改任何XOM设置。
- 只能使用Flash 32位程序命令对XOM设置页进行编程。
- 通过批量擦除命令或XOM页面擦除功能清除XOM设置和XOM区域（针对XOM的特殊擦除命令）。
- 用户必须重新启动Flash初始化以启用任何新的XOM设置。
- 打开调试模式，ICE才能进入XOM。

一旦XOM区域处于启用状态，用户只能采用整体擦除命令或XOM页面擦除功能清除XOM区域及其对应的XOM设置。要执行XOM页面擦除功能，用户必须通过写入FMC_ISPADDR中的目标XOM区域的基址和FMC_ISPDAT中的常量值0x0055aa03来执行FLASH页面擦除（0x22）。注意，在每一个起效的XOM区域，有效的指令只有整体擦除，页擦除，校验和读CID/DID。XOM设定页不能访问，除了整体擦除，字编程和32-位读。

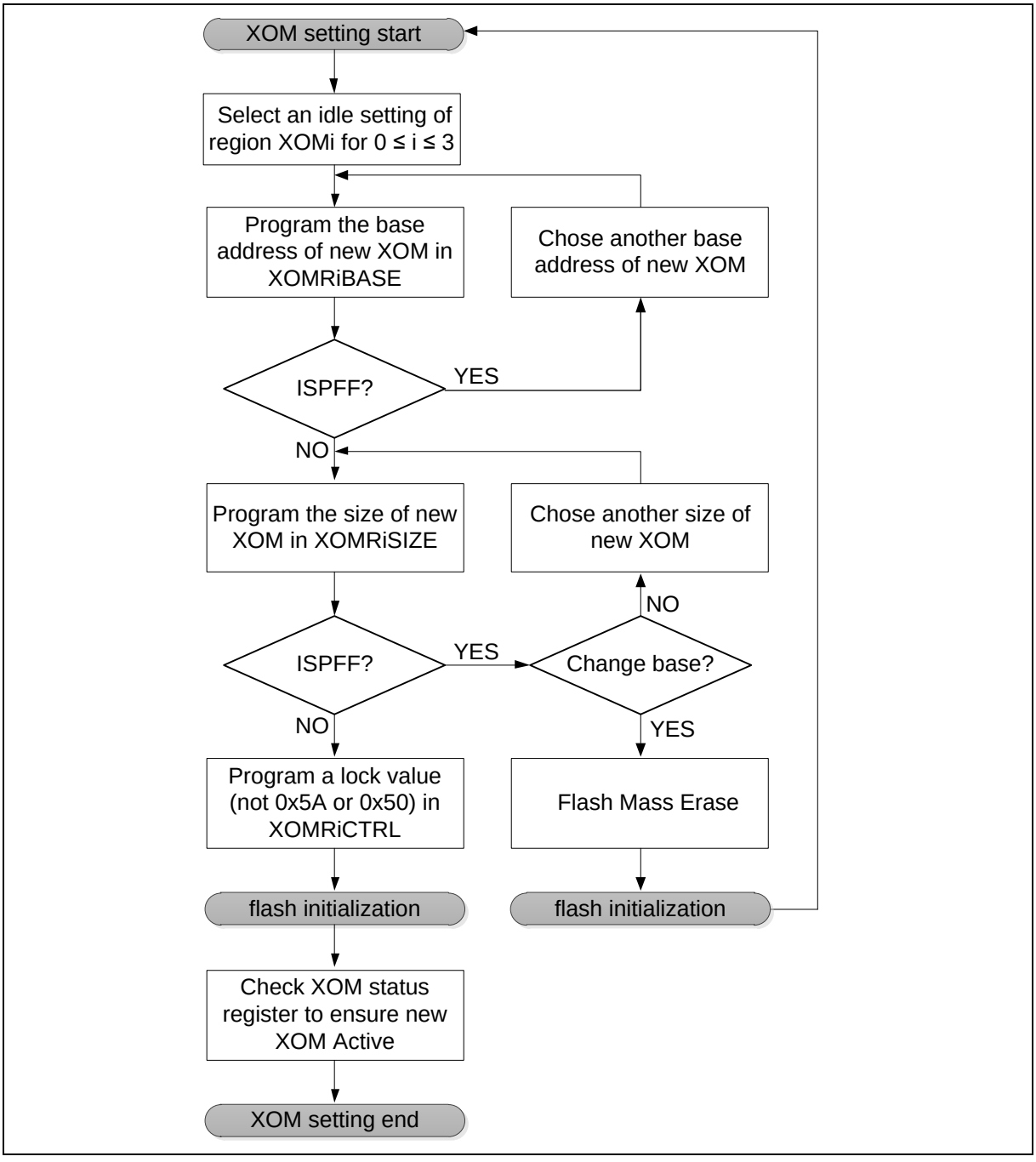


图 6.5-4 新 XOM 设置流程

XOMR0BASE (地址 = 0x0020_0000)

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
XOMR0BASE							
15	14	13	12	11	10	9	8
XOMR0BASE							
7	6	5	4	3	2	1	0
XOMR0BASE							

位	描述	
[31:24]	**Reserved**	保留。
[23:0]	**XOMR0BASE**	**XOM区域0的基地址** XOMR0BASE必须页对齐。 **注:**页大小为4K字节。

XOMR0SIZE (地址 = 0x0020_0004)

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
XOMR0SIZE							

位	描述	
[31:8]	**Reserved**	保留。
[7:0]	**XOMR0SIZE**	**XOM区域0页数** XOMR0SIZE必须页对齐且少于63页。 XOMR0SIZE最小值为1页。 如果XOMR0SIZE被写入0，寄存器值将被设置为1。 **注：**页面大小为4K字节。

XOMR0CTRL (地址 = 0x0020_0008)

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
XOMR0CTRL							

位	描述	
[31:8]	**Reserved**	保留。
[7:0]	**XOMR0CTRL**	**控制XOM区域0** 0x5a = XOM区域0关闭。 0x50 = XOM处于调试模式。 其他情况 = XOM 区域0启用状态。 **注1：** 但对于Flash行为，用户不能将0写入1.用户需要检查锁定值是否有效更改0x5A（0101_1010）。 **注2：** 上电或复位周期后，启用状态生效。

XOMR1BASE (地址 = 0x0020_0010)

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
XOMR1BASE							
15	14	13	12	11	10	9	8
XOMR1BASE							
7	6	5	4	3	2	1	0
XOMR1BASE							

位	描述	
[31:24]	**Reserved**	保留。
[23:0]	**XOMR1BASE**	**XOM区域1的基地址** XOMR0BASE必须页对齐。 **注:**页大小为4K字节。

XOMR1SIZE (地址 = 0x0020_0014)

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
XOMR1SIZE							

位	描述	
[31:8]	**Reserved**	保留。
[7:0]	**XOMR1SIZE**	**XOM区域1页数** XOMR0SIZE必须页对齐且少于63页。 XOMR0SIZE最小值为1页。 如果XOMR0SIZE被写入0，寄存器值将被设置为1。 **注：**页面大小为4K字节。

XOMR1CTRL (地址 = 0x0020_0018)

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
XOMR1CTRL							

位	描述	
[31:8]	**Reserved**	保留。
[7:0]	**XOMR1CTRL**	**控制XOM区域1** 0x5a = XOM区域0关闭。 0x50 = XOM处于调试模式。 其他情况 = XOM 区域0启用状态。 **注1：** 但对于Flash行为，用户不能将0写入1.用户需要检查锁定值是否有效更改0x5A（0101_1010）。 **注2：** 上电或复位周期后，启用状态生效。

XOMR2BASE (地址 = 0x0020_0020)

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
XOMR2BASE							
15	14	13	12	11	10	9	8
XOMR2BASE							
7	6	5	4	3	2	1	0
XOMR2BASE							

位	描述	
[31:24]	**Reserved**	保留。
[23:0]	**XOMR2BASE**	**XOM区域0的基地址** XOMR0BASE必须页对齐。 **注:**页大小为4K字节。

XOMR2SIZE (地址 = 0x0020_0024)

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
XOMR2SIZE							

位	描述	
[31:8]	**Reserved**	保留。
[7:0]	**XOMR2SIZE**	**XOM区域2页数** XOMR0SIZE必须页对齐且少于63页。 XOMR0SIZE最小值为1页。 如果XOMR0SIZE被写入0，寄存器值将被设置为1。 **注：**页面大小为4K字节。

XOMR2CTRL (地址 = 0x0020_0028)

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
XOMR2CTRL							

位	描述	
[31:8]	**Reserved**	保留。
[7:0]	**XOMR2CTRL**	**控制XOM区域2** 0x5a = XOM区域0关闭。 0x50 = XOM处于调试模式。 其他情况 = XOM 区域0启用状态。 **注1：** 但对于Flash行为，用户不能将0写入1.用户需要检查锁定值是否有效更改0x5A（0101_1010）。 **注2：** 上电或复位周期后，启用状态生效。

XOMR3BASE (地址 = 0x0020_0030)

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
XOMR3BASE							
15	14	13	12	11	10	9	8
XOMR3BASE							
7	6	5	4	3	2	1	0
XOMR3BASE							

位	描述	
[31:24]	**Reserved**	保留。
[23:0]	**XOMR3BASE**	**XOM区域3的基地址** XOMR0BASE必须页对齐。 **注:**页大小为4K字节。

XOMR3SIZE (地址 = 0x0020_0034)

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
XOMR3SIZE							

位	描述	
[31:8]	**Reserved**	保留。
[7:0]	**XOMR3SIZE**	**XOM区域3页数** XOMR0SIZE必须页对齐且少于63页。 XOMR0SIZE最小值为1页。 如果XOMR0SIZE被写入0，寄存器值将被设置为1。 **注：**页面大小为4K字节。

XOMR3CTRL (地址 = 0x0020_0038)

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
XOMR3CTRL							

位	描述	
[31:8]	**Reserved**	保留。
[7:0]	**XOMR3CTRL**	**控制XOM区域3** 0x5a = XOM区域0关闭。 0x50 = XOM处于调试模式。 其他情况 = XOM 区域0启用状态。 **注1：** 但对于Flash行为，用户不能将0写入1.用户需要检查锁定值是否有效更改0x5A（0101_1010）。 **注2：** 上电或复位周期后，启用状态生效。

6.5.4.6 *单次编程存储器 (OTP)*

有一个单次编程存储器(OTP)用来保存重要信息，该区域不允许用户再次修改，OTP是3K字节大小，地址空间是0x31_0000 ~ 0x31_0BFF。OTP数据最大空间是2K字节，地址是0x31_0000 ~ 0x31_07FF，每64位OTP数据有一个32位的“LOCK”位，地址从0x31_0800 到 0x31_0BFF，如图6.5-5所示。

“LOCK BIT”的目的是记录在OTP被编程的地址是否锁定，锁定 LOCK BIT≠0xFFFF_FFFF，未锁定 LOCK BIT= 0xFFFF_FFFF。例如，当LOCK BIT0不是0xFFFFFFFF，不管其内容是否全为1，OTP0不能再次被编程。 “Flash页擦除 /批量擦除命令/ Flash64-位编程/ Flash多字编程” 永远不允许在 OTP执行。在从 0x31_0000 到 0x31_07FF更新OTP的内容之前，用户必须先检查“LOCK BIT”。完成OTP数据编程之后，用户必须写一个非0xFFFF_FFFF的数据到之前编程数据的“LOCK BIT”，来确保不能再次修改。OTP通过ISP命令是只读的，CPU直接读是不可读的。

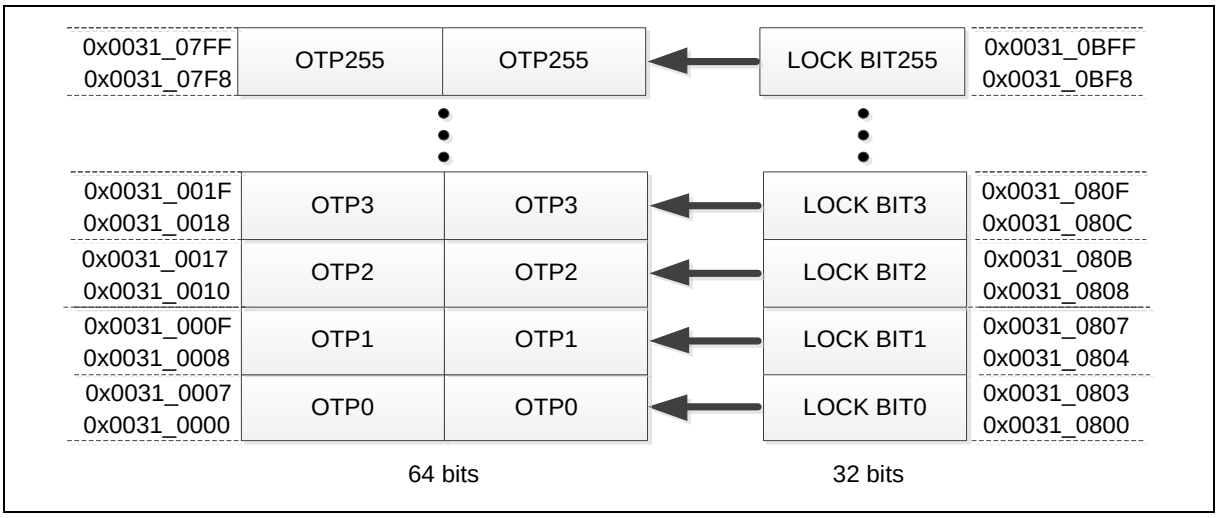


图 6.5-5 OTP 内存映射

6.5.4.7 *密钥保护存储(KPROM)*

密钥保护存储(KPROM)是用来存储96位的安全密钥及限制不匹配的密钥访问。KPROM包含28字节，第一页地址在0x30_1000 ~ 0x30_1017及第二页地址在0x30_1200 ~ 0x30_1203，在KEYLOCK (FMC_KPKEYSTS [1])没有被锁定时以上两页可以编程及擦除该区域。KPROM不能通过ISP/ICE/ICP /烧录器命令和接口直接读取。如果需要校验KPROM的数据，只能通过匹配密钥操作来校验安全密匙(KPKEY0ROM, KPKEY1ROM, 和 KPKEY2ROM)的数据。KPROM内存映射如图6.5-6所示。

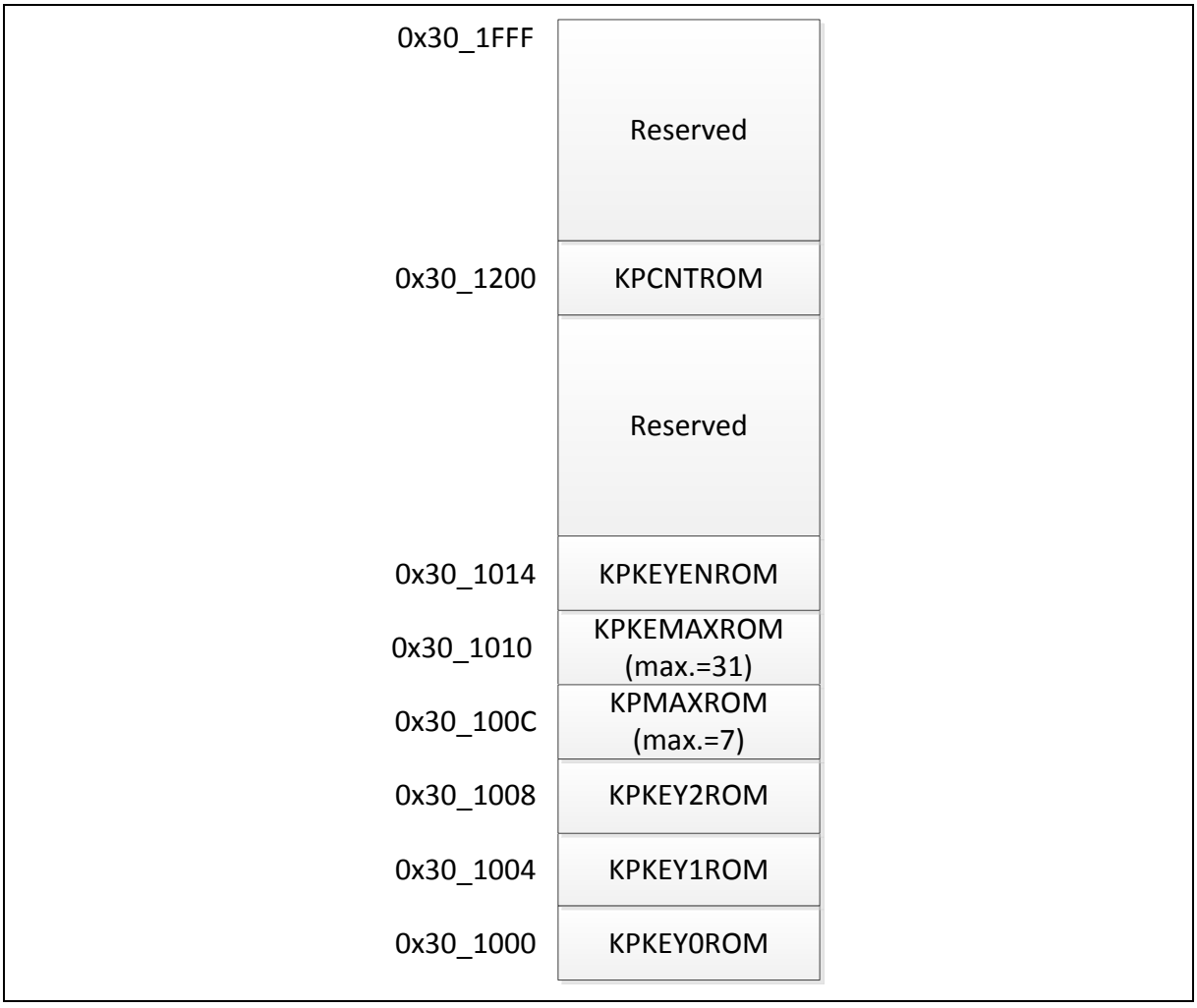


图 6.5-6 KPROM 内存映射

KPKEY0ROM (地址 = 0x0030_1000)

31	30	29	28	27	26	25	24
KPKEY0ROM							
23	22	21	20	19	18	17	16
KPKEY0ROM							
15	14	13	12	11	10	9	8
KPKEY0ROM							
7	6	5	4	3	2	1	0
KPKEY0ROM							

位	描述	
[31:0]	**KPKEY0ROM**	**KEY #0(只写)** KPKEY0ROM是96位安全密钥的前32位。KEYLOCK (FMC_KPKEYSTS [1])没有被锁定时，该区域可以被擦除和编程。KPKEY0ROM不能直接读取。

注意:这些位默认为1。

KPKEY1ROM (地址 = 0x0030_1004)

31	30	29	28	27	26	25	24
KPKEY1ROM							
23	22	21	20	19	18	17	16
KPKEY1ROM							
15	14	13	12	11	10	9	8
KPKEY1ROM							
7	6	5	4	3	2	1	0
KPKEY1ROM							

位	描述	
[31:0]	**KPKEY1ROM**	**KEY #1(只写)** KPKEY1ROM是96位安全密钥的第二个32位。KEYLOCK (FMC_KPKEYSTS [1])没有被锁定时，该区域可以被擦除和编程。KPKEY1ROM不能直接读取。

KPKEY2ROM (地址 = 0x0030_1008)

31	30	29	28	27	26	25	24
KPKEY2ROM							
23	22	21	20	19	18	17	16
KPKEY2ROM							
15	14	13	12	11	10	9	8
KPKEY2ROM							
7	6	5	4	3	2	1	0
KPKEY2ROM							

位	描述	
[31:0]	**KPKEY2ROM**	**KEY #2(只写)** KPKEY2ROM是96位安全密钥的第二个32位。KEYLOCK (FMC_KPKEYSTS [1])没有被锁定时，该区域可以被擦除和编程。KPKEY2ROM不能直接读取。

KPMAXROM (地址 = 0x0030_100C)

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved					KPMAXROM		

位	描述	
[31:3]	**Reserved**	保留。
[2:0]	**KPMAXROM**	**密钥尝试的最大错误上电次数(只写)** KPMAXROM在安全密钥不匹配的情况下，KPMAXROM限制了尝试的上电次数。上电时或复位后，KPMAXROM会被拷贝到寄存器KPMAX (FMC_KPCNT [10:8])。KEYLOCK (FMC_ KPKEYSTS [1])没有被锁定时，KPMAXROM可以被擦除和编程。KPMAXROM不能直接读取。

KPKEMAXROM (地址 = 0x0030_1010)

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved			KPKEMAXROM				

位	描述	
[31:5]	**Reserved**	保留。
[4:0]	**KPKEMAXROM**	**每次上电密钥尝试最大错误次数(只写)** KPKEMAXROM在安全密钥不匹配的情况下，KPKEMAXROM限制了每次上电时的尝试次数。上电时或复位后，KPKEMAXROM会被拷贝到寄存器KPKEMAX(FMC_KPKECNT[12:8])。KEYLOCK (FMC_ KPKEYSTS [1])没有被锁定时，KPKEMAXROM可以被擦除和编程。KPKEMAXROM不能直接读取。

KPKEYENROM (地址 = 0x0030_1014)

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
KPKEYENROM							

位	描述	
[31:8]	**Reserved**	保留。
[7:0]	**KPKEYENROM**	**KEYLOCK使能控制位(只写)** 在上电时或复位后，KEYLOCK (FMC_ KPKEYSTS [1]) 可以由KPKEYENROM使能。KEYLOCK (FMC_KEYSTS [1])没有被锁定时，KPKEYENROM可以被擦除和编程但不能直接读取。 0xFF=上电时或复位后，寄存器KEYLOCK(FMC_KPKEYSTS [1])清0。 其他=上电时或复位后，寄存器KEYLOCK(FMC_KPKEYSTS [1])置1。KPROM, LDROM 和APROM写保护。 **注：**当KPKEYENROM被通过ISP功能编辑为非0xFF值时，KEYLOCK (FMC_KPKEYSTS [1])将立即被使能。

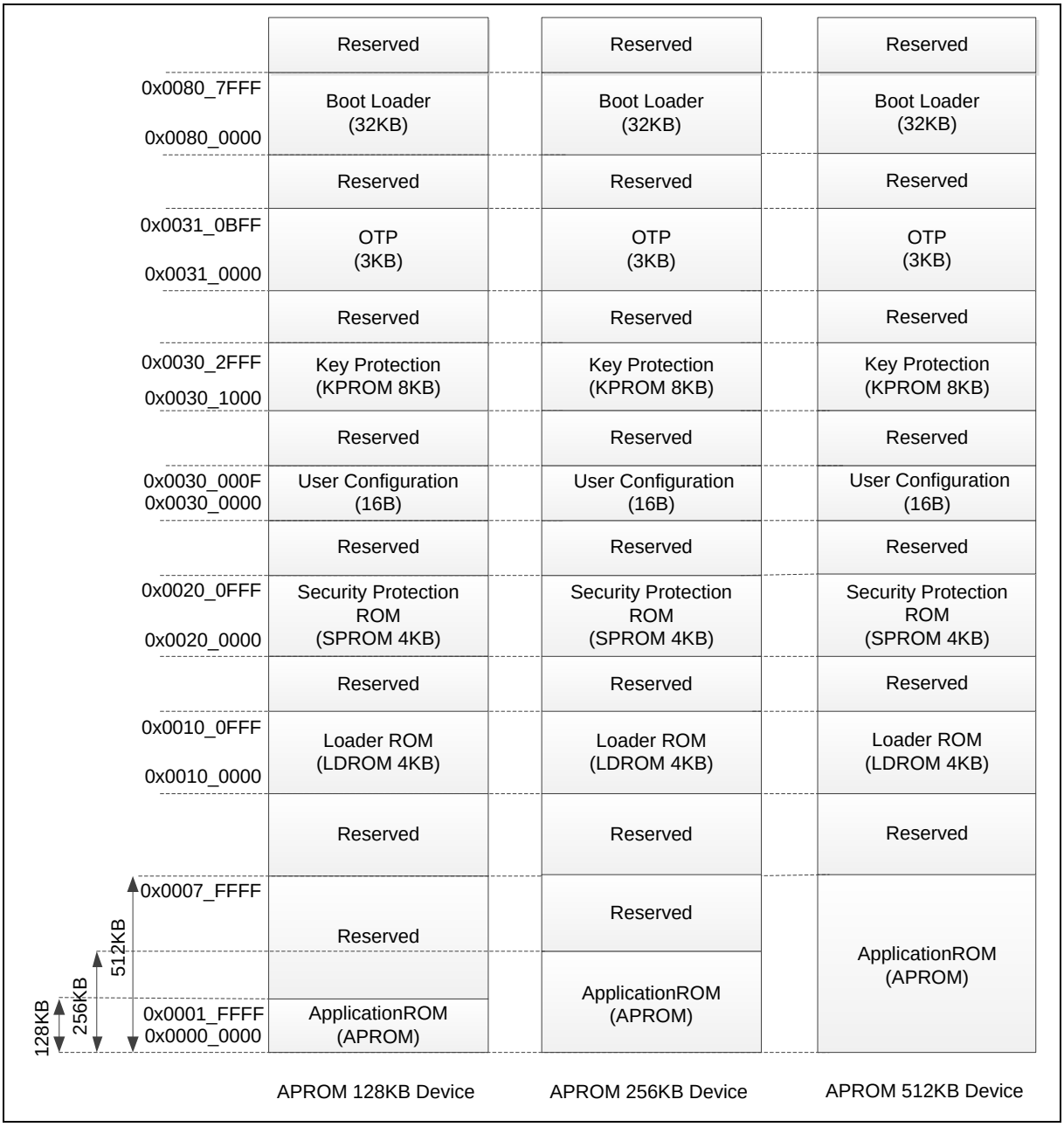
KPCNTROM (地址 = 0x0030_1200)

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
KPCNTROM							

位	描述	
[31:8]	**Reserved**	保留
[7:0]	**KPCNTROM**	**密钥错误尝试上电计数(只写)** KPCNTROM会记录上电/断电时的密钥不匹配次数。第一次密钥不匹配时，FMC会清除一位(由KPCNT解码得到)。当密钥匹配时，KPCNTROM会被硬件擦除且不能直接读出。KPCNTROM(8位)在上电或复位后编码成3位之后存储在KPCNT(FMC_KPCNT[2:0])，如下所示： KPCNTROM → KPCNT 1111_1111 → 000 1111_1110 → 001 1111_1100 → 010 1111_1000 → 011 1111_0000 → 100 1110_0000 → 101 1100_0000 → 110 1000_0000 → 111 其他→ 111

6.5.4.8 Flash内存映射

Flash存储映射和系统存储映射不同。系统存储映射是CPU用来从FMC存储取代码和数据。而Flash存储映射是ISP用于读取、编程及擦写FMC存储。Flash内存映射如图6.5-7所示。



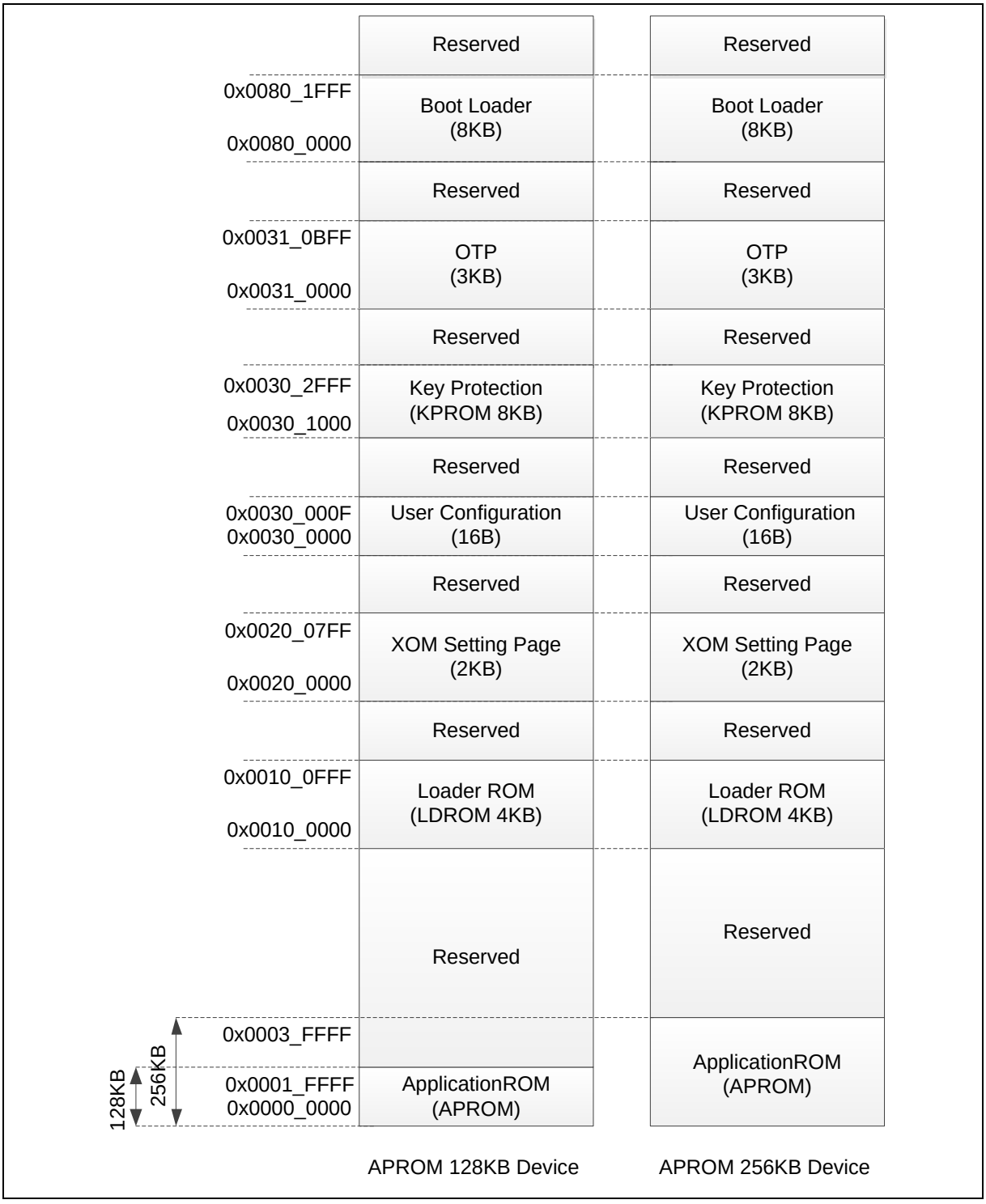


图 6.5-7Flash 内存映射

6.5.4.9 支持IAP模式的系统存储映射

在 CPU 访问 FMC 存储器获取代码或数据的时候，用到系统存储器映射。Boot Loader

(0x0080_0000~0x0080_7FFF),SPROM(0x0020_0000~0x0020_0FFF) 和 LDROM(0x0010_0000~0x0010_0FFF)在Flash中的地址映射相同。数据Flash与APROM共享，数据Flash的基地址由CONFIG1设定。在Flash初始化时，CONFIG1的内容被加载到DFBA(数据Flash基地址寄存器)。

当APROM 大小分别是 64KB/128KB/256KB/384KB/512KB时，DFBA~(0x0001_FFFF/0x0003_FFFF) 是Cortex®-M4数据存取的数据Flash区，0x0000_0200~(DFBA-1)是Cortex®-M4指令存取的APROM区。

系统存储向量的地址在0x0000_0000 到 0x0000_01FF。

图6.5-8表示支持IAP模式的系统存储映射。

在CPU启动期间，APROM, LDROM和Boot loader可以映射到系统存储向量区。当芯片启动的时候，有三种支持IAP的系统存储器映射模式：(1)支持IAP的LDROM (2)支持IAP的APROM (3)支持IAP的Boot loader。

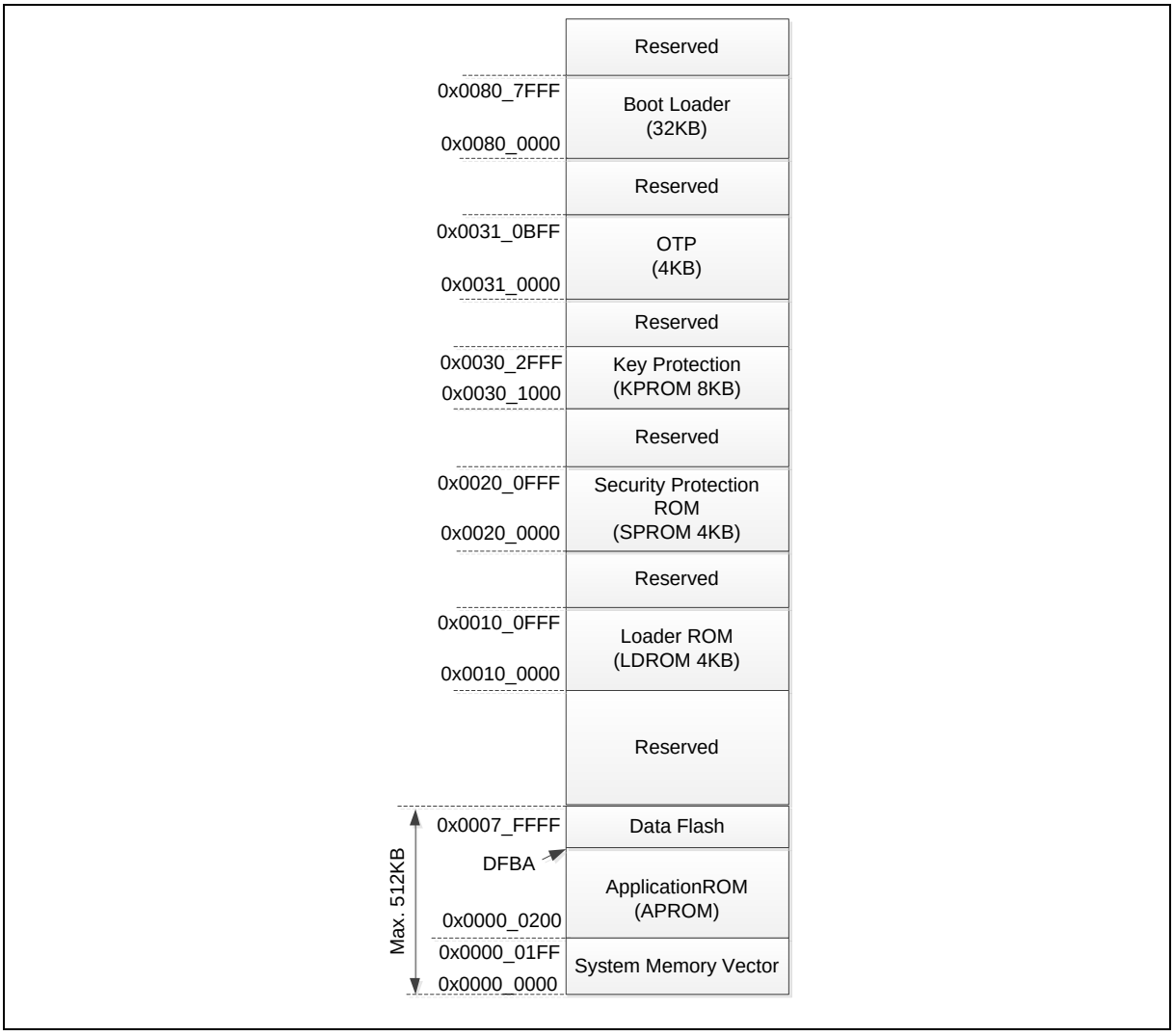


图 6.5-8 支持 IAP 模式的系统存储映射(仅 M48xID/M48xGA)

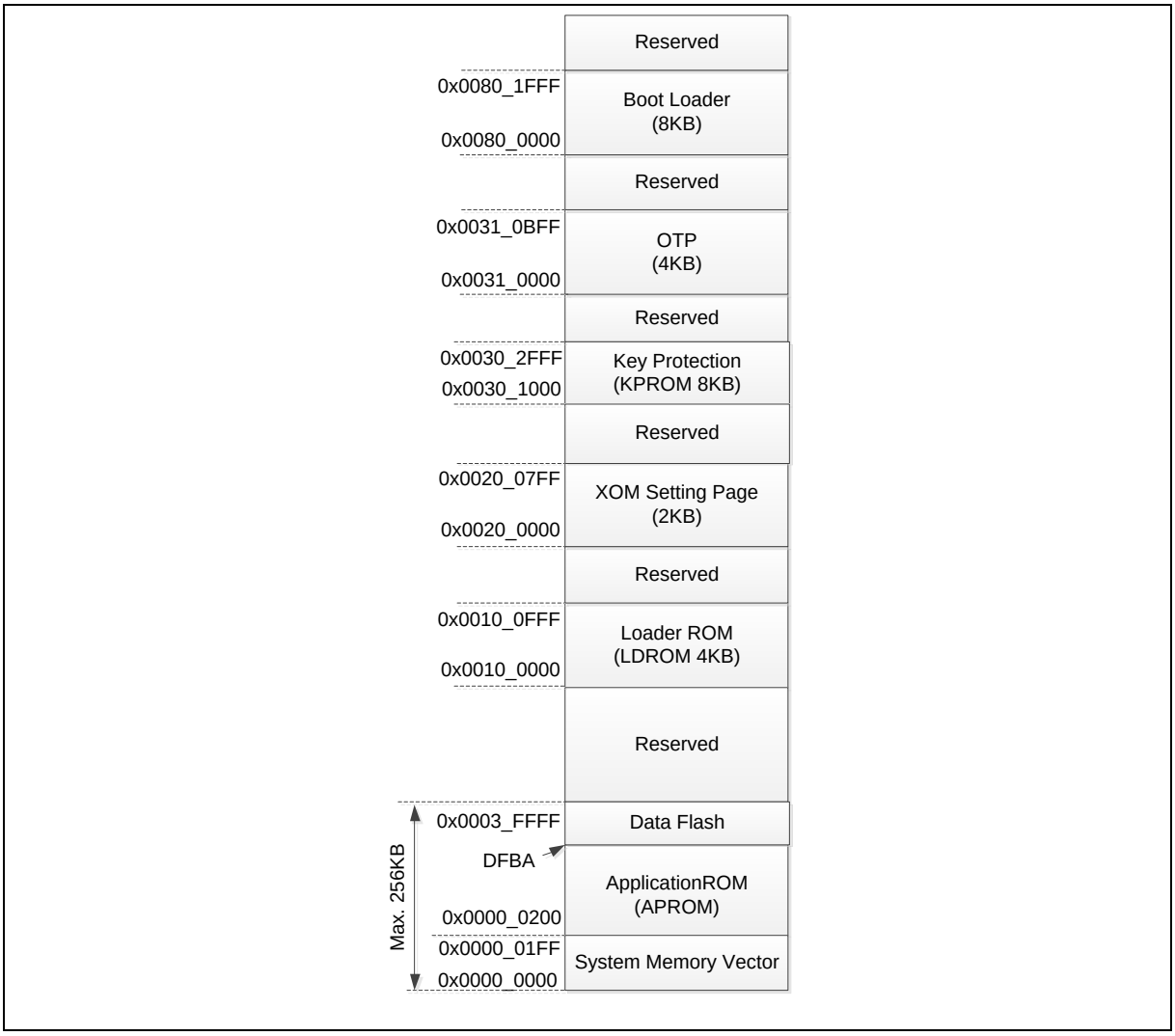


图 6.5-9 支持 IAP 模式的系统存储映射(仅 M48xGC/M48xE8)

支持IAP的LDROM模式，LDROM (0x0010_0000~0x0010_01FF)映射到系统存储向量区，用于Cortex®-M4F指令或数据存取。图6.5-10表示支持IAP的LDROM模式的存储映射。

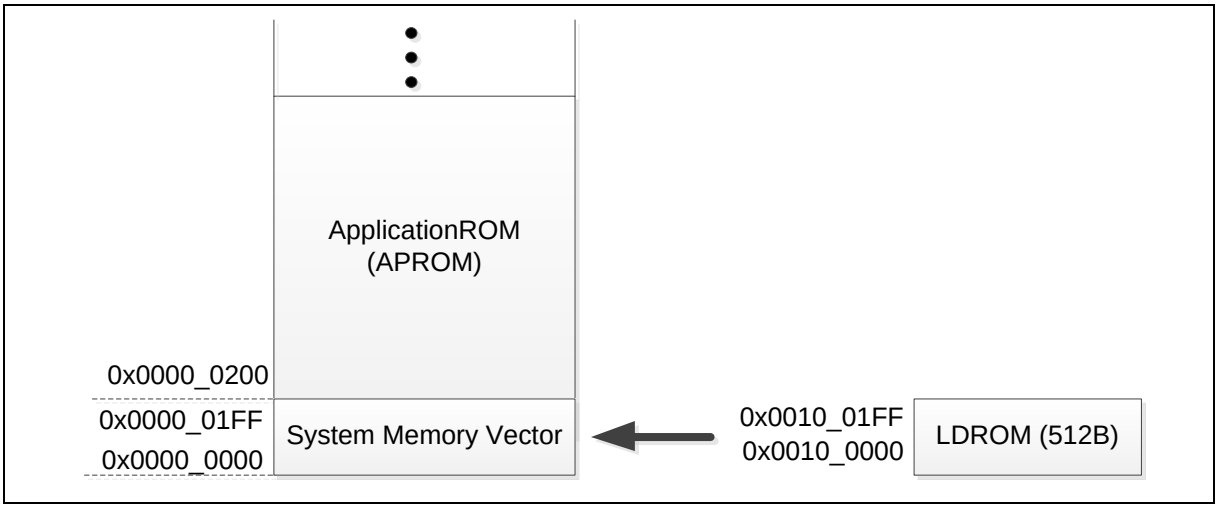


图 6.5-10 支持 IAP 的 LDROM 模式

支持IAP的APROM模式，APROM (0x0000_0000~0x0000_01FF) 映射到系统存储向量区，用于Cortex®-M4指令或数据存取。图6.5-11表示支持IAP的APROM模式的存储映射。

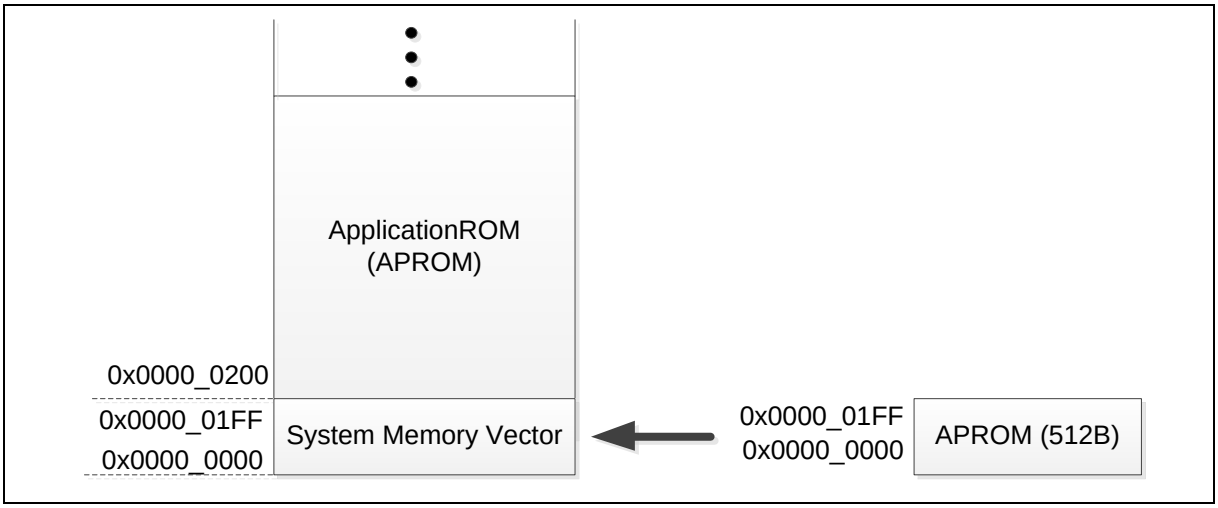


图 6.5-11 支持 IAP 的 APROM 模式

支持IAP的Boot Loader模式，Boot Loader (0x0080_0000~0x0080_01FF) 映射到系统存储向量区，用于Cortex®-M4指令或数据存取。图6.5-12表示支持IAP的Boot Loader模式的存储映射。

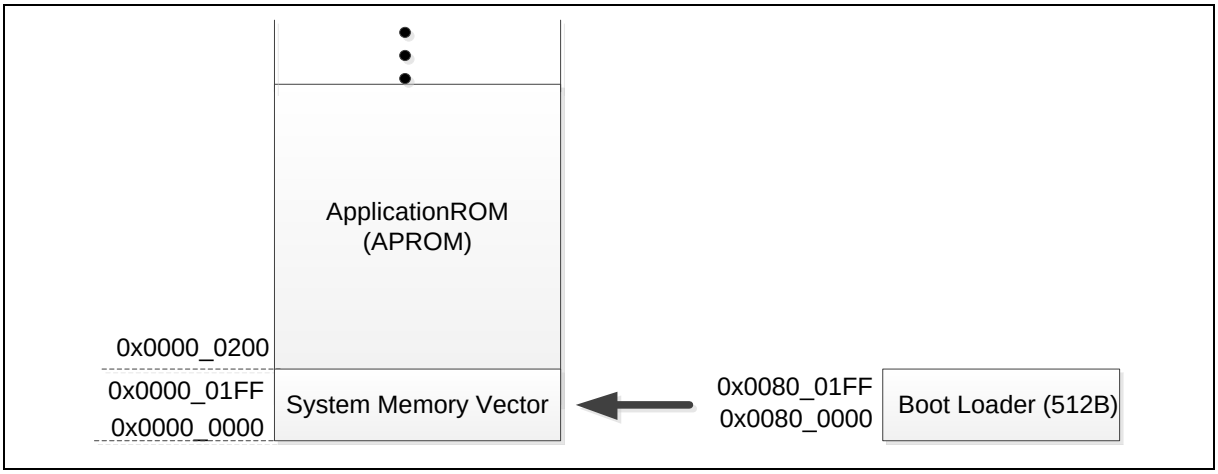


图 6.5-12 支持 IAP 的 Boot Loader 模式

在支持IAP的系统存储映射模式中，当CPU运行时，APROM, LDROM,和Boot Loader可以重新映射到系统存储向量区。用户可以写重映射的目标地址到寄存器FMC_ISPADDR，然后通过“向量重映射”指令(0x2E)触发ISP流程。在寄存器VECMAP (FMC_ISPSTS[23:9])，显示最终的系统存储向量映射地址。

6.5.4.10 不支持IAP功能的系统内存映射

不支持IAP功能的系统内存映射，CPU仍然可以访问Boot Loader(0x0080_0000~0x0080_7FFF)和SPROM(0x0020_0000~0x0020_0FFF)，但不支持系统内存向量映射。在芯片启动时，有两种不支持IAP的系统内存映射：(1) 不支持IAP功能的LDROM模式 (2) 不支持IAP功能的APROM模式。在不支持IAP功能的LDROM模式中，LDROM 基地址映射到0x0000_0000，CPU 编程不能访问APROM。在不支持IAP功能的APROM模式中，APROM的基地址映射到0x0000_0000，CPU编程不能访问LDROM。数据Flash与APROM共享，数据Flash的基地址由CONFIG1设定。在Flash初始化期间，CONFIG1内容被加载到DFBA(数据Flash基地址)。当APROM大小分别是 64KB/128KB/256KB/384KB/512KB 时，DFBA~(0x0001_FFFF/0x0003_FFFF) 是 Cortex®-M4 的数据 Flash 区， 0x0000_0000~(DFBA-1) 是 Cortex®-M4的APROM区。图6.5-13表示不支持IAP模式的系统存储映射。

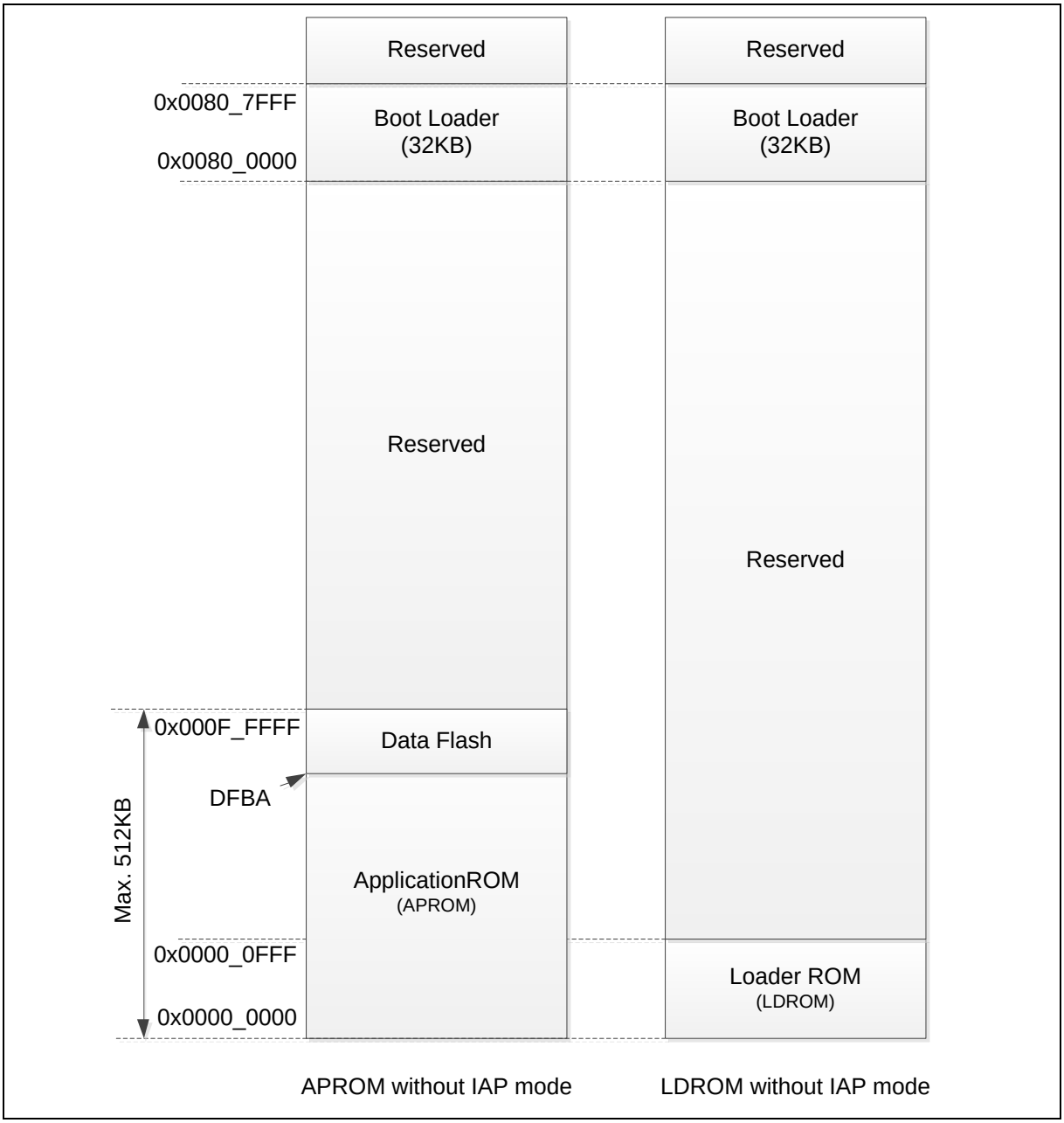
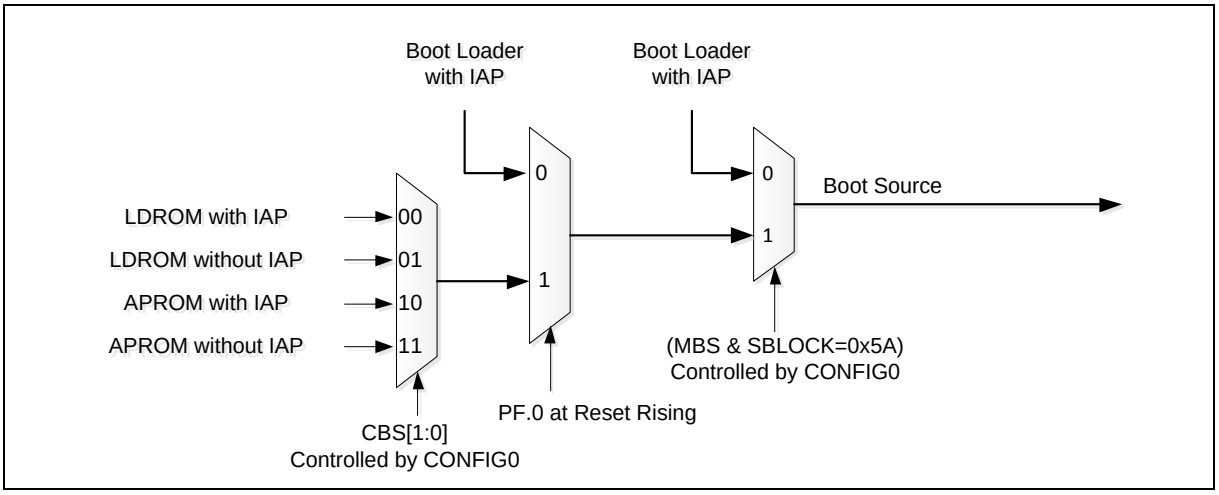


图 6.5-13 不支持 IAP 模式的系统存储映射

6.5.4.11 启动选择

M480系列提供了五种启动方式供用户选择，包括支持IAP功能的LDROM, 不支持IAP功能的LDROM, 支持IAP功能的APROM, 不支持IAP功能的APROM,支持IAP功能的Boot Loader。用户可以用下面三种方法来让CPU从Boot loader 中取指令。在复位管脚上升沿时拉低PF.6管脚，MBS (CONFIG0[5])为低或SBLOCK (CONFIG2[15:8]) 不是 0x5A。其他情况启动来源和系统内存映射由CBS (CONFIG0[7:6]) 和MBS (CONFIG0[5]) 设置。

启动源选择如图6.5-14，每一个启动选项是否支持向量映射如表6.5-3.



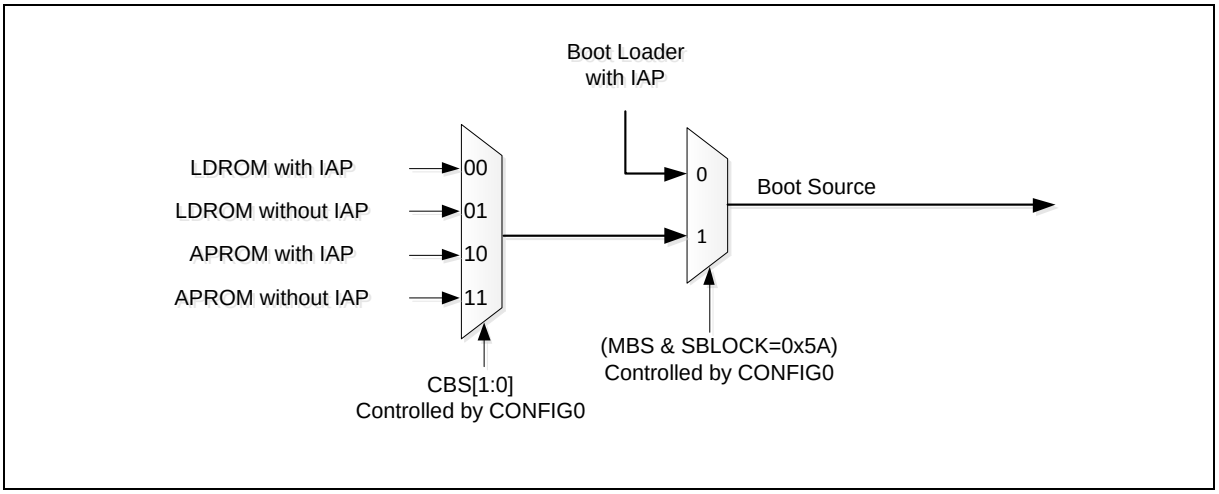


图 6.5-14 启动源选择

MBS	CBS[1:0]	启动选择/系统存储映射	向量映射是否支持
1	00	LDROM with IAP	√
1	01	LDROM without IAP	-
1	10	APROM with IAP	√
1	11	APROM without IAP	-
0	XX	Boot Loader with IAP	√

表 6.5-3 向量映射支持情况

6.5.4.12 在应用编程(IAP)

M480 系列提供了在应用编程(IAP)功能，用户可以切换APROM, LDROM 和 Boot Loader之间的代码。用户可以使能IAP功能通过设定芯片的启动位寄存器CBS (CONFIG0[7:6]) 等于10 或00，或者MBS (CONFIG0[5]) 等于 0，或者SBLOCK (CONFIG2[15:8]) 是一个非0x5A 的值。(如果MSB(CONFIG0[5])=0或执行安全启动，CBS[0] (FMC_ISPSTS[1])将显示为0，并且忽略CONFIG0[6]的设定)。

当支持IAP功能的芯片启动模式使能，任何可以执行的代码(512字节对齐)允许随时映射到系统内存向量。用户可以改变重映射地址到FMC_ISPADDR，然后用”向量重映射”命令触发ISP流程。

6.5.4.13 在系统编程(ISP)

M480系列支持在系统编程(ISP)模式，可以通过软件控制重新烧写片上Flash。使用ISP可以在目标板上直接编程，固件与多种接口相搭配，如UART, USB, I2C, SPI, 和 CAN (依据各芯片特性)。使得用户可以使用多种方法来更新APROM的代码。ISP功能操作的目标Flash存储空间不能跨bank，下表列出了所有ISP命令。表6.5-4列出了所有ISP的指令。

ISP 对片上Flash操作提供如下功能

- 支持Flash页擦除功能
- 支持Flash数据编程
- 支持读Flash数据功能
- 支持读公司ID功能
- 支持读设备ID功能
- 支持读UID功能
- 支持内存CRC32 checksum计算功能
- 支持Falsh位全为1校验功能
- 支持系统内存向量重映射功能

ISP 命令

ISP 命令	FMC_ISPCMD	FMC_ISPADDR	FMC_ISPDAT FMC_MPDAT0~FMC_MPDAT3
Flash Page Erase	0x22	FLASH存储器的有效地址 它必须是页（4K字节）地址对齐。注意 FMC_ISPADDR[11:0] 会被忽略。	N/A
FlashBank Erase	0x23	目标bank的APROM的有效地址 注意 FMC_ISPADDR[15:0] 会被忽略。	N/A
Flash Block Erase	0x25	APROM的有效地址 它必须是4页（16K字节）地址对齐 注意 FMC_ISPADDR[13:0] 会被忽略。.	N/A
Flash批量擦除 (该命令仅当 MERASE(CONFIG0[13]) 位 = 0.时有效)	0x26	0x0000_0000	N/A

Flash32-位编程	0x21	Flash存储器的有效地址	FMC_ISPDAT :编程数据
Flash64-位编程	0x61	Flash存储器的有效地址	FMC_MPDAT0: 编程数据的低位（LSB） FMC_MPDAT1: 编程数据的高位（MSB）
Flash多字编程	0x27	Flash存储器APROM 和 LDROM的有效地址	FMC_MPDAT0: 第一个编程数据 FMC_MPDAT1: 第二个编程数据 FMC_MPDAT2: 第三个编程数据 FMC_MPDAT3: 第四个编程数据
Flash读	0x00	Flash存储器的有效地址	FMC_ISPDAT: 返回数据
Flash64-位读	0x40	Flash存储器的有效地址	FMC_ISPDAT: FMC_ISPADDR的返回数据 FMC_MPDAT0: FMC_ISPADDR的返回数据 FMC_MPDAT1: FMC_ISPADDR+4的返回数据
读公司 ID	0x0B	0x0000_0000	FMC_ISPDAT: 0x0000_00DA
读设备 ID	0x0C	0x0000_0000	FMC_ISPDAT: 返回设备 ID
读 CRC32 Checksum	0x0D	0x0000_0000	FMC_ISPDAT: 返回 Checksum
执行 CRC32 Checksum 计算	0x2D	内存组织有效起始地址 它必须由4K字节/页排列	FMC_ISPDAT: 要计算的内存大小，必须 4 K字节对齐
读Flash位全为1校验结果	0x08	保存"Flash位全为1校验"的地址	FMC_ISPDAT: 返回的结果 0xA110_0000 : Flash位不全为1 0xA11F_FFFF: Flash位全为1
执行Flash位全为1校验	0x28	内存组织的有效起始地址 必须 4 K字节页对齐	FMC_ISPDAT: Flash大小 必须 4 Kbytes 对齐
读UID	0x04	0x0000_0000	FMC_ISPDAT: U ID 字0
		0x0000_0004	FMC_ISPDAT: U ID 字1
		0x0000_0008	FMC_ISPDAT: U ID 字2
		0x0000_0070	FMC_ISPDAT[11:0]: 内建V_{BG}的ADC转换结果
向量重映射	0x2E	APROM,LDROM 或 boot loader 有效地址，它必须512字节地址对齐	N/A

表 6.5-4 ISP 命令表

ISP 流程

FMC控制器提供了片上Flash内存的读，擦除和编程操作。一些FMC控制器的寄存器是写保护的，所以在设定之前要解锁。

在解锁了保护寄存器之后，用户需要设定FMC_ISPCTL控制寄存器来决定更新LDROM、 APROM、SPROM或配置区，然后设定ISPEN (FMC_ISPCTL[0]) 来使能ISP功能。

一旦FMC_ISPCTL寄存器被设置成功，用户可以设定FMC_ISPCMD(参考上述ISP命令列表)来完成相应的操作。设置FMC_ISPADDR作为flash内存的目标地址。FMC_ISPDAT根据命令FMC_ISPCMD，可以设定为要编程的数据或作为读数据的返回数据。

一旦FMC_ISPCTL寄存器被设置成功，用户可以设定FMC_ISPCMD(参考表6.5-4 ISP 命令列表) 来完成相应的操作。设置FMC_ISPADDR作为flash内存的目标地址。FMC_ISPDAT根据命令FMC_ISPCMD，可以设定为要编程的数据或作为读数据的返回数据。ISP流程如图6.5-15。

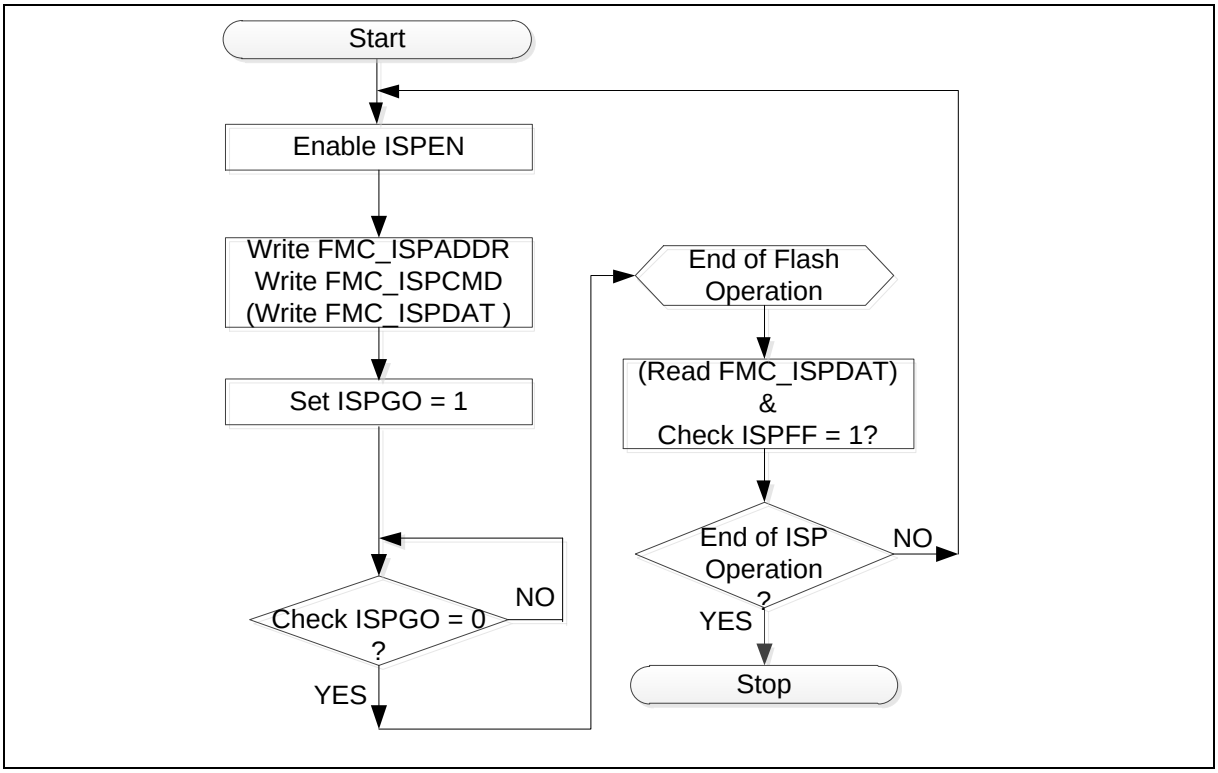


图 6.5-15ISP 流程示例

最终，设定ISPGO (FMC_ISPTRG[0]) 寄存器来执行ISP功能。当ISP功能启动，ISPBUSY(FMC_ISPSTS[0]) 和 MPBUSY(FMC_MPSTS[0]) 会被置1当ISP功能完成以后，ISPGO(FMC_ISPTRG[0]), ISPBUSY(FMC_ISPSTS[0]) 和 MPBUSY(FMC_MPSTS[0])自动清0。

ISP完成后，几个错误条件需要检查。如果出现错误，ISP操作就不会生效并且ISP失败标志被置位。ISPFF(FMC_ISPSTS[6])标志只由软件清除。即使ISPFF(FMC_ISPSTS[6])保持1，也可以开启下一个ISP流程。因此，建议在ISP操作完成后检查ISPFF(FMC_ISPSTS[6])位，如果被置1要清0。

当ISPGO(FMC_ISPTRG[0])置位，然后CPU访问相同的bank，CPU将一直等到ISP操作完成。这个时期内外围设备跟通常一样保持运行。任何中断请求都不会响应直到CPU完成ISP操作。当ISP操作完成，ISPGO位将被硬件自动清0。用户可以通过ISPGO(FMC_ISPTRG[0])位来检查ISP操作是否完成。

当CPU访问操作和ISP命令执行在不同的bank，CPU和 ISP命令可以并行工作。

6.5.4.14 片上Flash 编程

M480系列提供了32-位，64-位和多-字Flash存储器编程功能来加速Flash更新流程，表6.5-5列出了FMC控制寄存器对应的编程功能。

寄存器	描述	32-位编程	64-位编程	多字节编程
FMC_ISPCTL	ISP控制寄存器	✓	✓	✓
FMC_ISPADDR	ISP地址寄存器	✓	✓	✓
FMC_ISPDAT	ISP 数据寄存器	✓	N/A	N/A
FMC_ISPCMD	ISP 命令寄存器	0x21	0x61	0x27
FMC_ISPTRG	ISP 触发寄存器	✓	✓	✓
FMC_ISPSTS	ISP 状态寄存器	✓	✓	N/A
FMC_MPDAT0	ISP 数据0寄存器	N/A	✓	✓
FMC_MPDAT1	ISP数据1寄存器	N/A	✓	✓
FMC_MPDAT2	ISP数据2寄存器	N/A	N/A	✓
FMC_MPDAT3	ISP数据3寄存器	N/A	N/A	✓
FMC_MPSTA	ISP 多字编程状态	N/A	N/A	✓
FMC_MPADDR	ISP 多字编程地址	N/A	N/A	✓

表 6.5-5 FMC 用于 Flash 编程的控制寄存器

64-位编程

M480系列64-位编程功能比32-位编程功能快。FMC_ISPDAT是32-位编程数据寄存器。在64-位编程中，有两个编程数据寄存器，一个是低位FMC_MPDAT0，另一个是高位FMC_MPDAT1，ISP命令是0x61,其他寄存器与32-位编程寄存器相同。图6.5-16和图6.5-17为32-位 / 64-位编程流程。

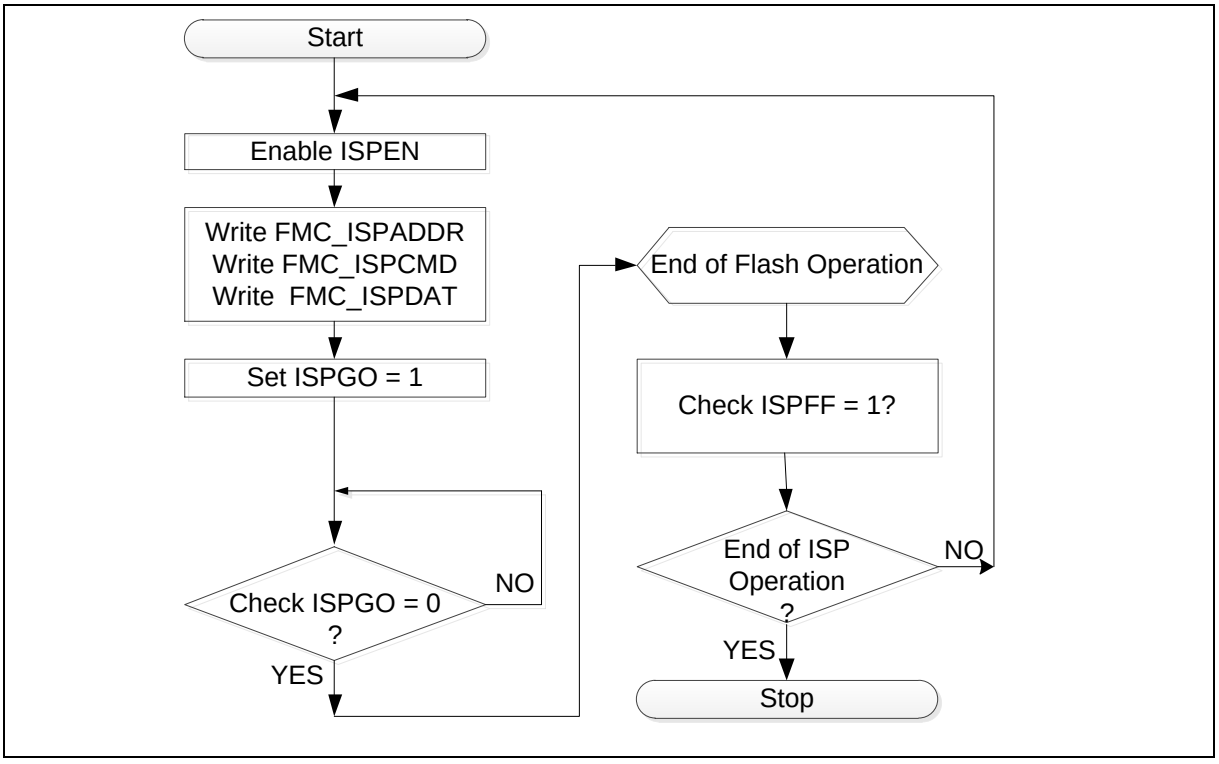


图 6.5-16ISP 32-位编程流程

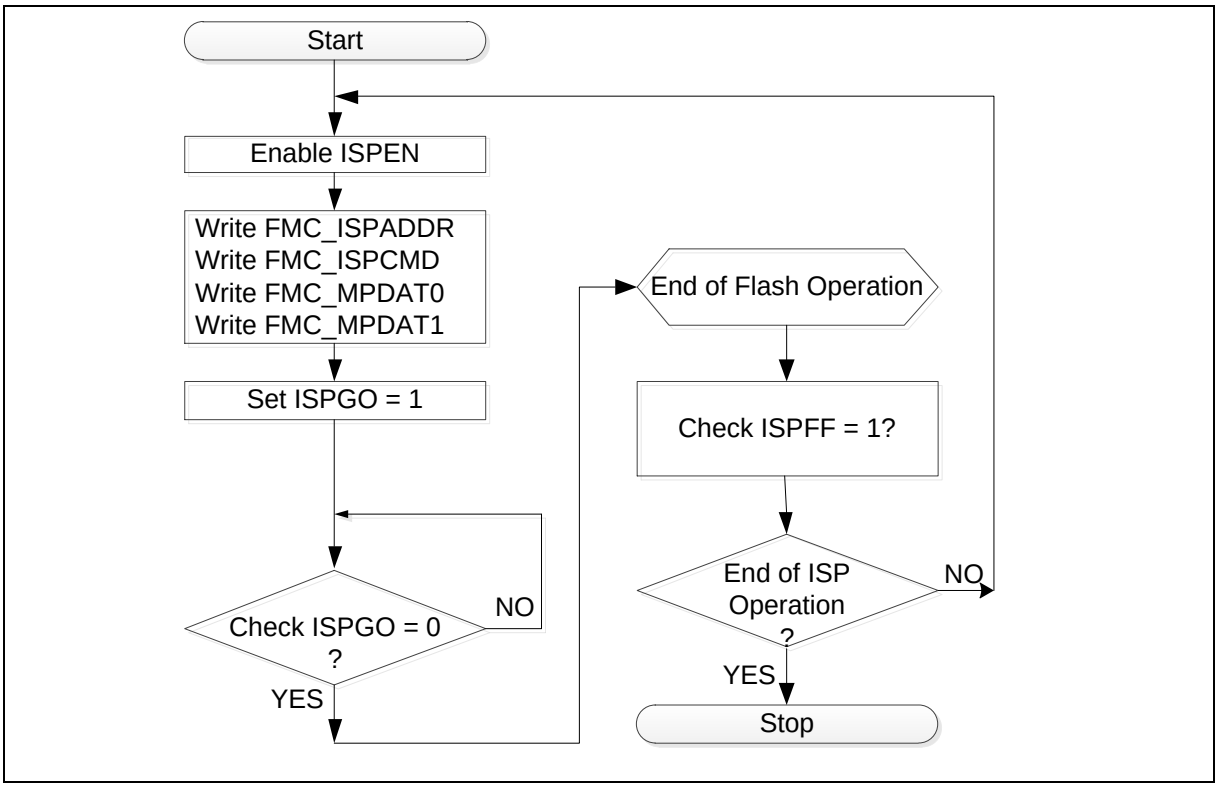


图 6.5-17ISP 64-位编程流程

多字编程

M480支持多字编程功能来加速Flash的更新速度。最大的编程长度是512字节，最小的编程长度是8字节。如果程序大于8字节，多字编程是最快的编程方式，因为每次操作只需一次Flash建立时间与保持时间。图6.5-18比较了每种编程的时间。

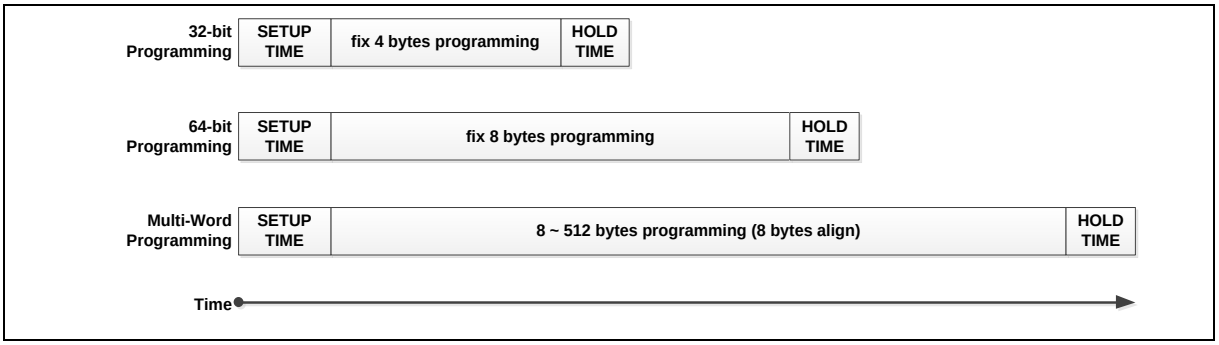


图 6.5-18 多字编程时间

在多字编程操作中，Cortex®-M4F需要监测编程缓冲器的空闲状态。CPU需要持续准备下一次的编程数据。多字编程固件(firmware)不可以放在APROM或者LDROM中，因为CPU取指令不能停止。固件(firmware)应存放在Boot loader或片上SRAM中，用于避免CPU停止。

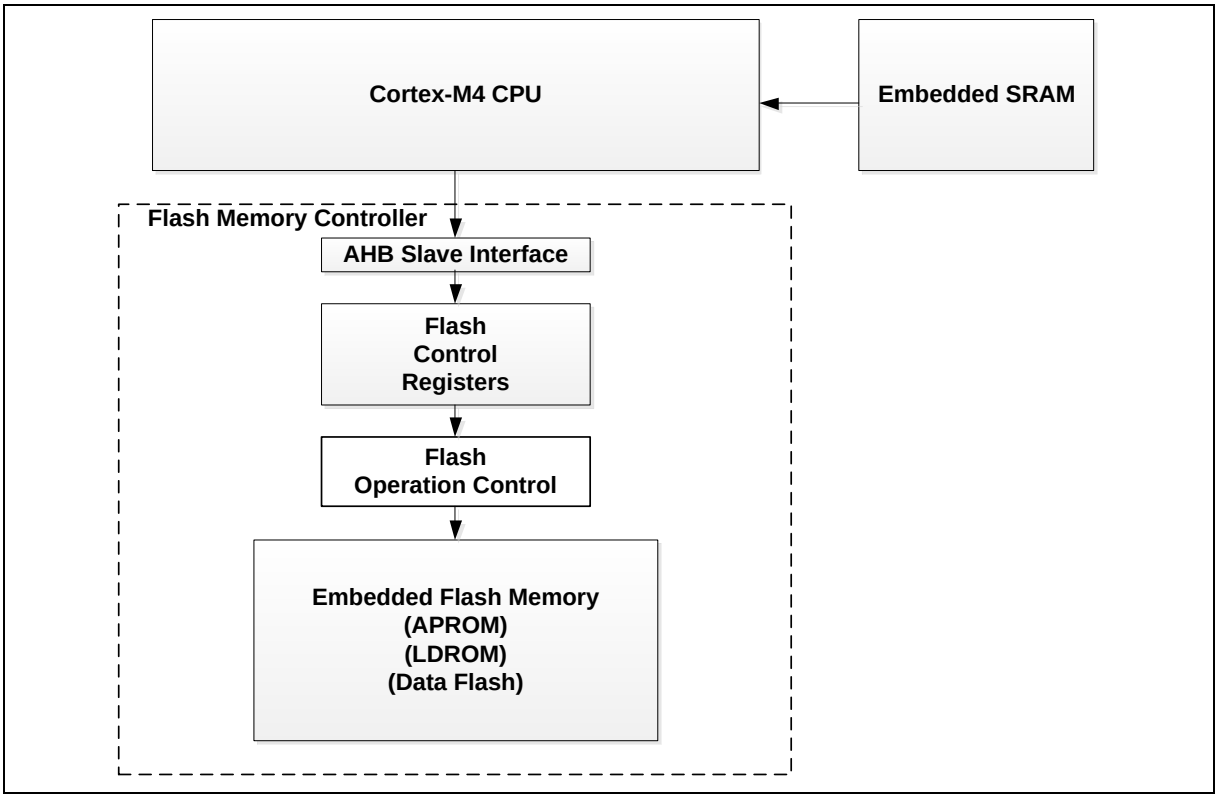


图 6.5-19 固件(firmware)在 SRAM 中的多字编程

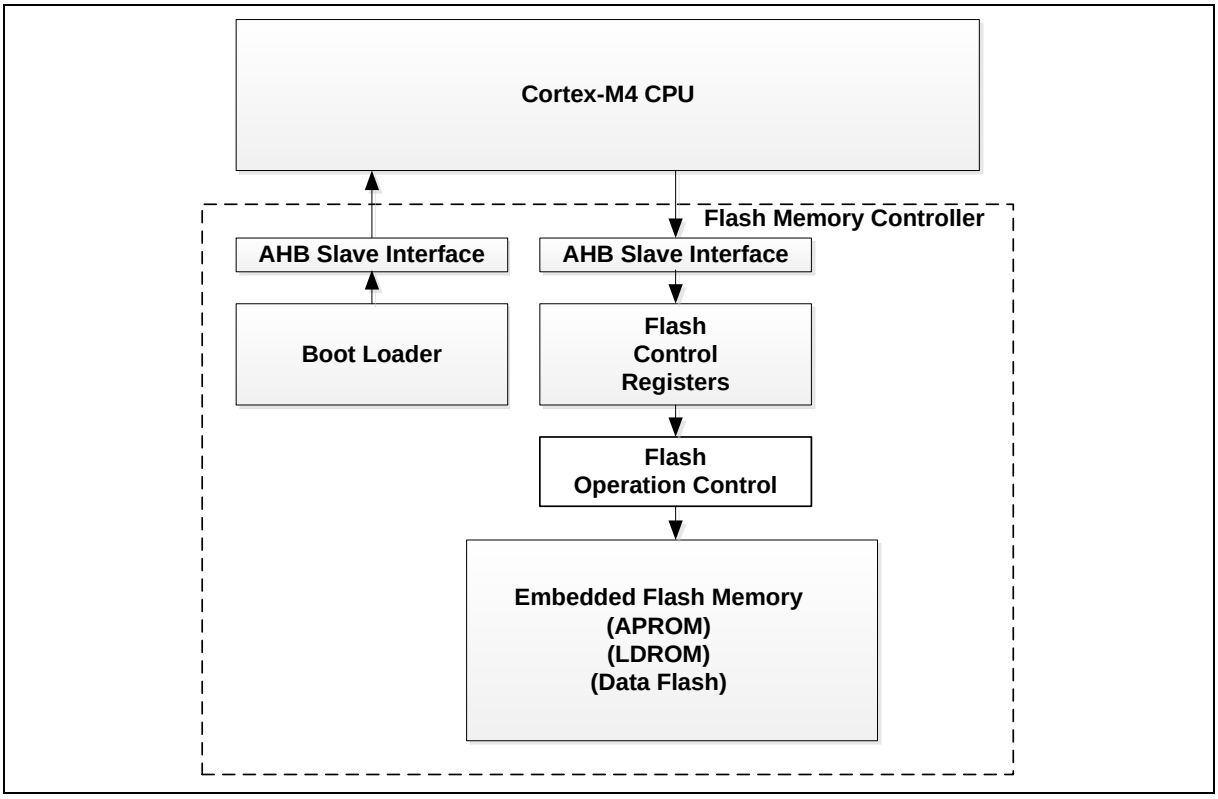


图 6.5-20 固件(firmware)在 Boot Loader 中的多字编程

多字编程的流程如图6.5-21. ISP的起始地址(FMC_ISPADDR)是8字节对齐的，FMC_ISPADDR[2:0]应为0. ISPDAT0(FMC_MPDAT0)是第一个数据字(偏移量 0x0), ISPDAT1(FMC_MPDAT1)是第二个数据字(偏移量 0x4), ISPDAT2(FMC_MPDAT2)是第三个数据字(偏移量 0x8), ISPDAT3(FMC_MPDAT3)是第四个数据字(偏移量 0xC)。如果ISP起始地址FMC_ISPADDR [3] =0，第一个字数据应放在FMC_MPDAT0，第二个字数据应放在FMC_MPDAT1，第三个字数据应放在FMC_MPDAT2，第四个字数据应放在FMC_MPDAT3。如果ISP起始地址FMC_ISPADDR [3] =1，第一个字数据应放在FMC_MPDAT2，第二个字数据应放在FMC_MPDAT3，第三个字数据应放在FMC_MPDAT0，第四个字数据应放在FMC_MPDAT1.最大的可编程大小是512字节，并以512字节地址对齐。当FMC控制器在多字编程操作，CPU需要监测缓冲器的状态D3~D0(FMC_MPSTS[7:4])以及MPBUSY (FMC_MPSTS[0])来等待缓冲器清空 ((D1,D0)=00 或 (D3,D2)=00) 。然后CPU需要及时更新下一次的编程数据(FMC_MPDAT0, FMC_MPDAT1, FMC_MPDAT2 和 FMC_MPDAT3)。否则，FMC控制器将退出多字编程操作(MPBUSY (FMC_MPSTS[0]) = 0). 如果CPU无法及时更新数据(MPBUSY (FMC_MPSTS[0]) =0), CPU需要重新开始一个新的多字编程来延续流程，FMC_MPADDR提供了上一次编程地址的信息。结尾操作，CPU必须检查ISPFF (FMC_MPSTS[2])来确认多字操作是否已经成功的完成了。

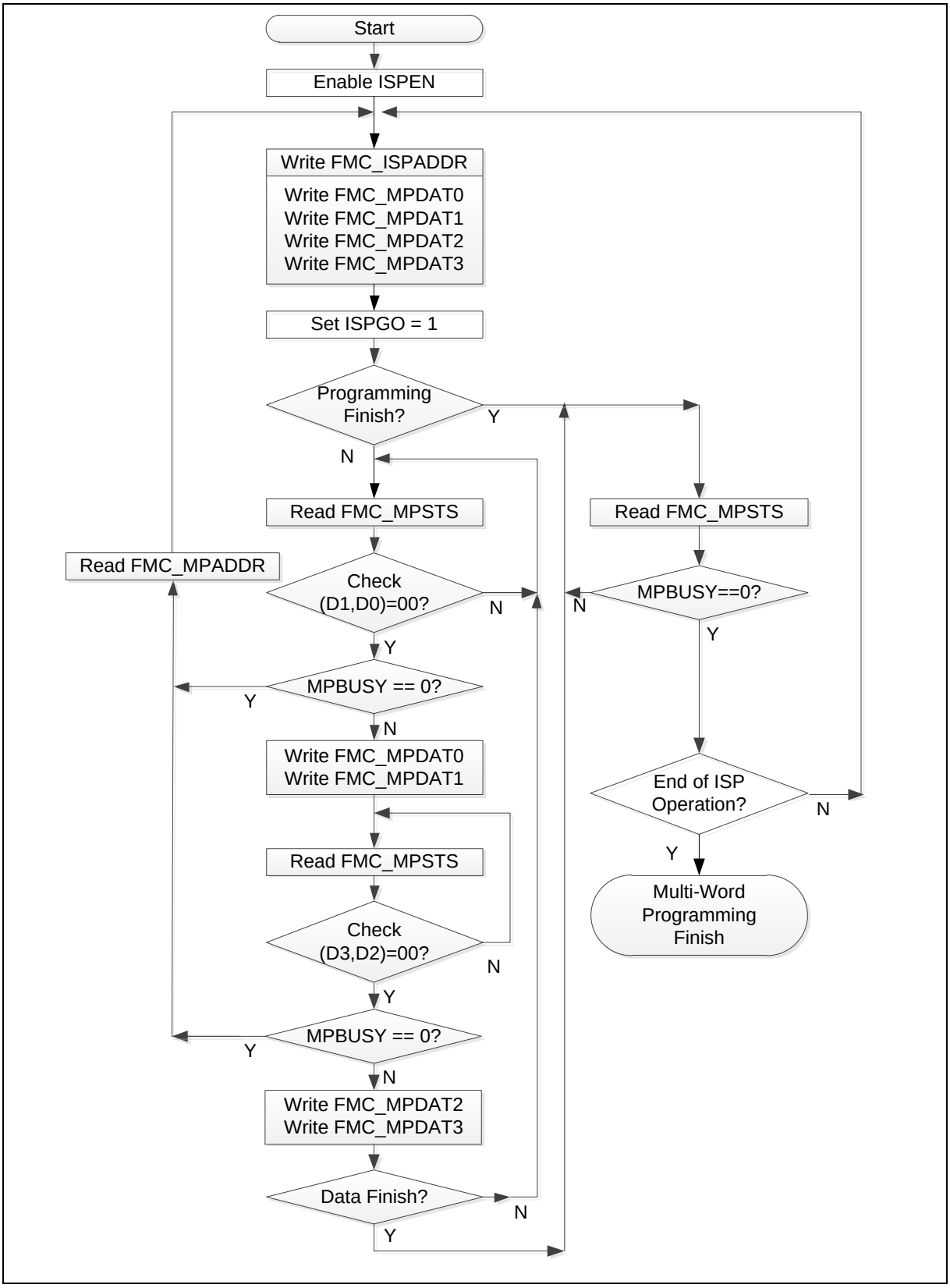


图 6.5-21 多字编程流程

6.5.4.15 快速Flash编程校验

在传统的Flash编程操作，控制器接收到编程触发事件后控制片上Flash编程时序如图6.5-22所示。

M480系列提供快速Flash编程验证功能，由硬件实现，可以节省CPU读回和比较的时间。当片上Flash已经编程结束，硬件控制器访问Flash做数据的读取和比较。最终，比较的结果会存在寄存器PGFF (FMC_ISPSTS[5])。如果输入的编程数据与输出的数据不同，PGFF置1. 这个标志将保持，直到软件清0或有一个新的擦除操作。快速Flash编程验证流程如图6.5-22。

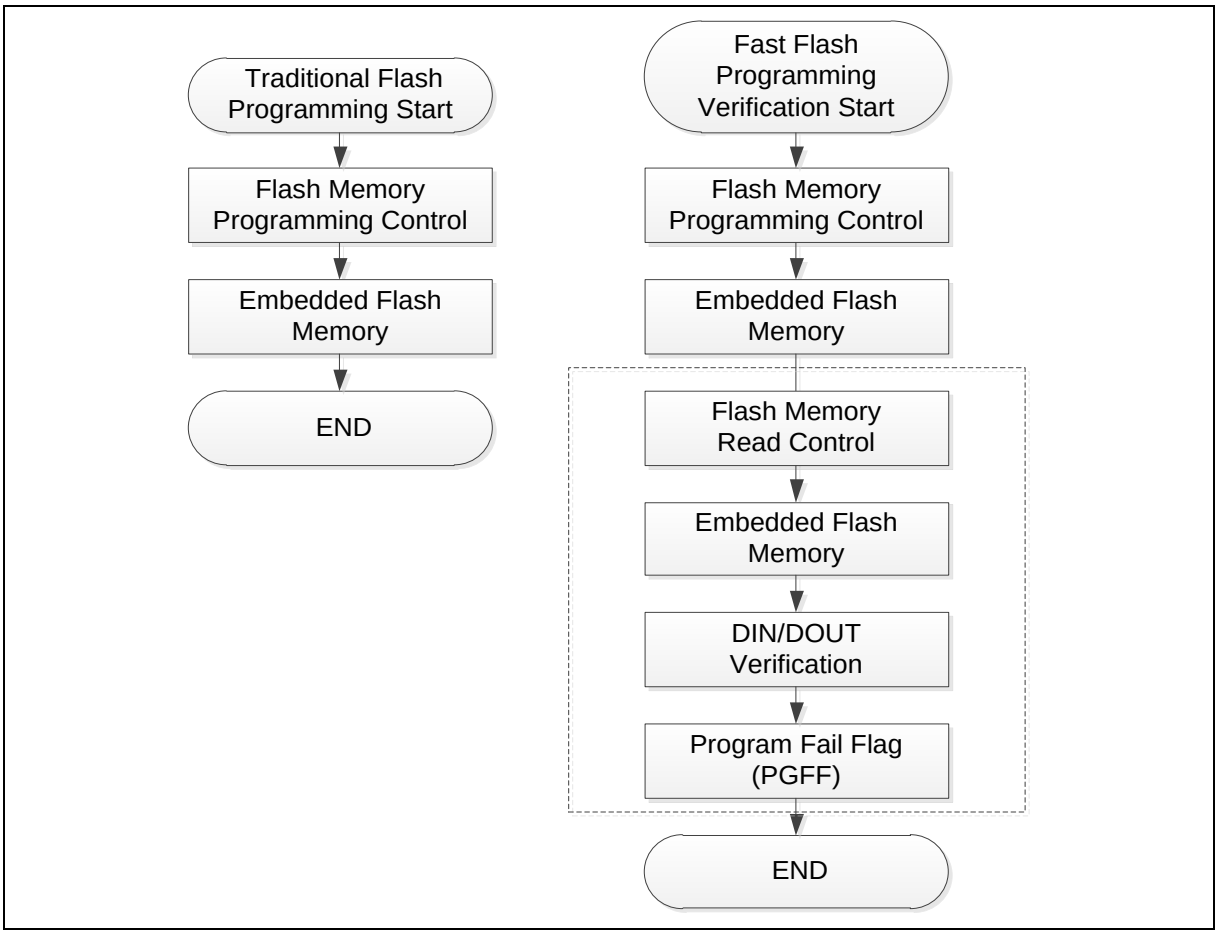


图 6.5-22 快速 Flash 编程校验流程

在传统的Flash更新操作，Flash存储器有三个步骤来完成Flash存储器更新流程，(1) Flash 擦除 (2) Flash 编程 (3) Flash 读回所有数据校验。在M480系列，步骤(3)只读 FMC_ISPSTS 来检查的PGFF 标志，不用读回数据来确认。图6.5-23对比了传统编程校验流程和快速编程校验流程。

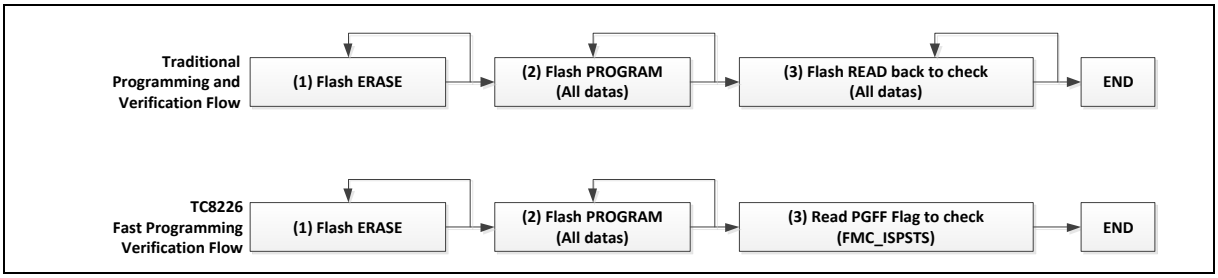


图 6.5-23 校验流程

快速的Flash校验功能，用于32-位编程和64-位编程操作，但是不适合多字编程操作，原因是多字编程操作是片上Flash HV (高电压)持续编程。

6.5.4.16 CRC32 Checksum计算

M480系列支持CRC32 checksum计算功能来帮助用户迅速检查APROM, LDROM, SPROM和Boot Loader内容。CRC32多项式是：

CRC-32: $X^{32} + X^{26} + X^{23} + X^{22} + X^{16} + X^{12} + X^{11} + X^{10} + X^8 + X^7 + X^5 + X^4 + X^2 + X + 1$

种子 = 0xFFFF_FFFF

CRC32 checksum计算流程如图6.5-24。

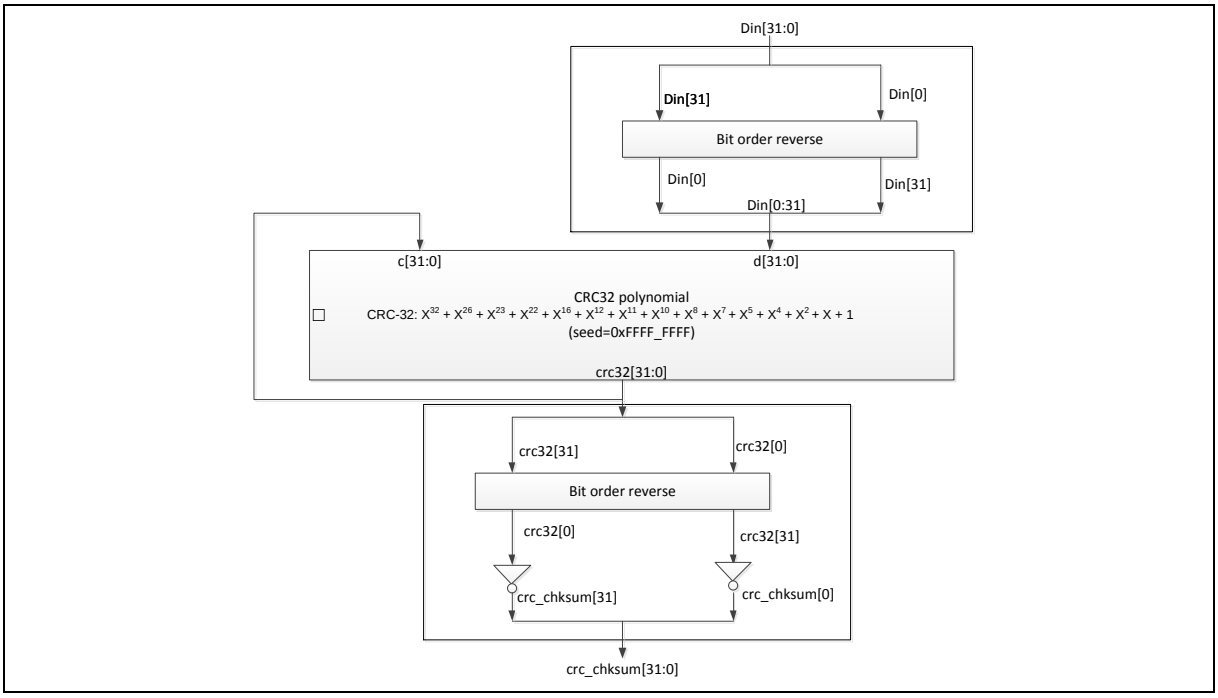


图 6.5-24Flash CRC32 Checksum 计算

完成CRC32 checksum计算有三个步骤：

1. “执行内存CRC32Checksum”ISP操作。
2. “读内存CRC32Checksum”ISP操作。
3. 读取FMC_ISPDAT寄存器得到checksum数据。

在步骤1，用户需设置内存的起始地址(FMC_ISPADDR)和大小(FMC_ISPDAT)，两者都要4K字节对齐，大小需≥ 4 K字节，起始地址适用于APROM, LDROM, SPROM和 Boot Loader。

在步骤2，FMC_ISPADDR需与步骤1保持一致。

在步骤3，checksum是从寄存器FMC_ISPDAT读出。如果checksum是0x0000_0000，有两种可能性：(1) 还在Checksum计算过程中，(2) 地址和大小超出了设备限制。

注意： CRC32 checksum的范围在一次操作中不能跨bank。

6.5.4.17 Flash 全为1校验

M480 系列支持flash全为1校验功能来帮助用户快速检查APROM, LDROM在Flash擦除操作后一个内存块内容是否为空白。

完成Flash全为1的校验有两或三个步骤。

两步骤流程：

1. “运行 Flash 全为1校验” ISP操作。
2. 读 ALLONE(FMC_ISPSTA[7])位获取校验结果。

ALLONE : 1, 校验存储区块中所有flash位都是1。

ALLONE : 0, 校验存储区块中所有flash位不全都是1。

三步骤流程：

1. “运行 Flash 全为1校验” ISP操作。
2. 执行ISP “读 Flash 全为1结果” 操作。
3. 读 FMC_ISPDAT 获取校验结果。

   FMC_ISPDATA : 0xA11F_FFFF, 校验存储区块中所有Flash位都是1。

   FMC_ISPDATA : 0xA110_0000, 校验存储区块中所有Flash位不全都是1。

在步骤 1中, 用户需设置内存的起始地址(FMC_ISPADDR)和大小(FMC_ISPDAT)用于校验，两者都要4K字节对齐，大小需≥ 4 K字节，起始地址适用于APROM, LDROM和SPROM 。

在步骤 2中, FMC_ISPADDR 需与步骤1保持一致。

注意：执行全为1校验的范围在一次操作中不能跨bank。

6.5.4.18 自动调节Flash访问周期

M480系列支持自动调节Flash访问周期功能。用户不需要手动设置Flash访问周期当系统时钟（HCLK）改变时，硬件将监控HCLK频率并产生一个最优化的周期数使Flash控制器获得最佳的执行效率。

任何更新HCLK时钟源和分频器寄存器的操作都会触发自动调节事件，包括HCLKSEL(CLK_CLKSEL0)，CLK_PLLCTL和HCLKDIV(CLK_CLKDIV0)。当FADIS(FMC_CYCCTL[8])置1并检测到一个事件，FMC将设置最大数（例如 8）临时的存入CYCLE (FMC_CYCCTL[3:0])寄存器，使得时钟的改变不会影响到Flash的访问。HIRC时钟是必须的用于自动调节能产生一个精准的周期给HCLK记录。如果要禁用该功能，用户置FADIS(FMC_CYCCTL[8])为0并且确认FCYCDIS(FMC_ISPSTS[4])被置1。HCLK检测频率和优化CYCLE数如表6.5-6所示。

由于HIRC有误差(12 MHz ± 2%),相应的HCLK和HIRC就不同步，FMC可能会产生两个最优化的周期数在同一个HCLK时钟频率。

HCLK 时钟频率	最优化的 CYCLE 数
0 MHz ~33 MHz	1
29 MHz~66 MHz	2
59 MHz~98 MHz	3
91 MHz~131 MHz	4
120MHz~163MHz	5
151 MHz~194 MHz	6
>195 MHz	7

表 6.5-6 自动调节功能下最优化的 Flash 访问周期

自动调节Flash访问周期流程如图6.5-25。

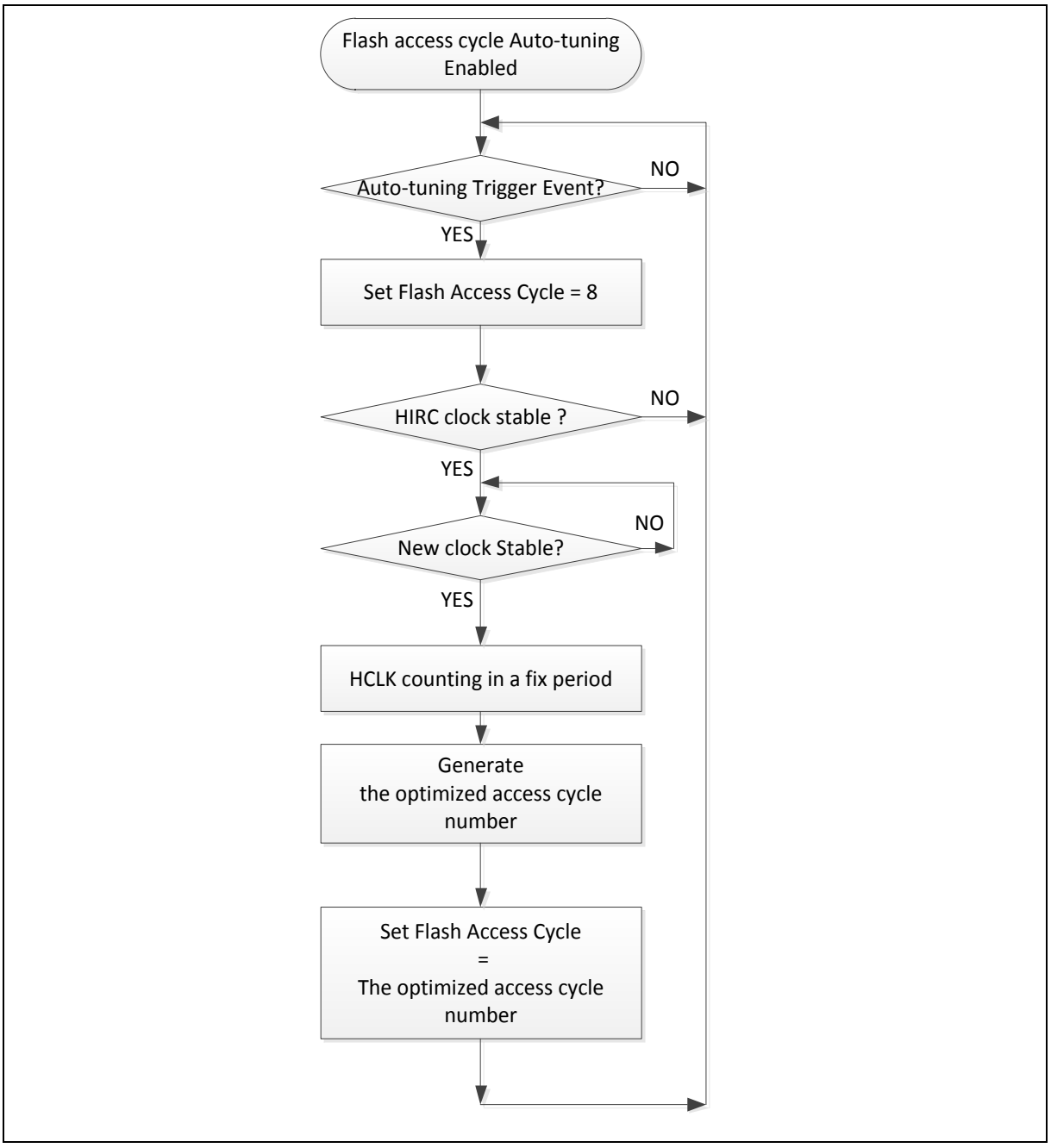


图 6.5-25 自动调节 Flash 访问周期

6.5.4.19 安全密匙保护

M480系列支持安全密匙保护功能来保护APROM, LDROM, SPROM, CONFIG 和 KPROM的数据用于编程和擦除的命令。当KEYLOCK (FMC_KPKEYSTS [1]) 激活，KPROM, LDROM 和 APROM (Data Flash 除外) 都在写保护状态。基于KPKEYENROM[1:0]的值，SPROM 和 CONFIG的写保护是可选的。任何编程和页擦除操作都是非法的，整个芯片批量擦除除外。如果APROM, LDROM, SPROM, CONFIG 和 KPROM需要修改，必须先执行正确的密匙比较操作来清除KEYLOCK (FMC_KPKEYSTS [1])的状态。当

Flash数据被KEYLOCK锁死，用户可以在锁定效果表中查找FMC的锁定效果。

安全密匙设置流程

安全密匙设置流程如图6.5-26所示。所有安全密匙信息存储在KPROM。首先KPROM (8KB)必须执行擦除操作来禁止安全密匙，然后写三个安全密匙到KPKEY0ROM, KPKEY1ROM 和 KPKEY2ROM。继续写KPMAXROM 和 KPKEMAXROM来定义安全密匙比较重试的限制次数。如果安全密匙比较不匹配，KPMAXROM用来限制上电的次数。KPKEMAXROM用来限制每次上电密匙不匹配操作的次数。最终，写一个非0xFF的值到KPKEYENROM来使能安全密匙保护功能。KPROM数据是不可读的，数据不能被读回检查。检查安全密匙的唯一方法是执行密匙比较操作。

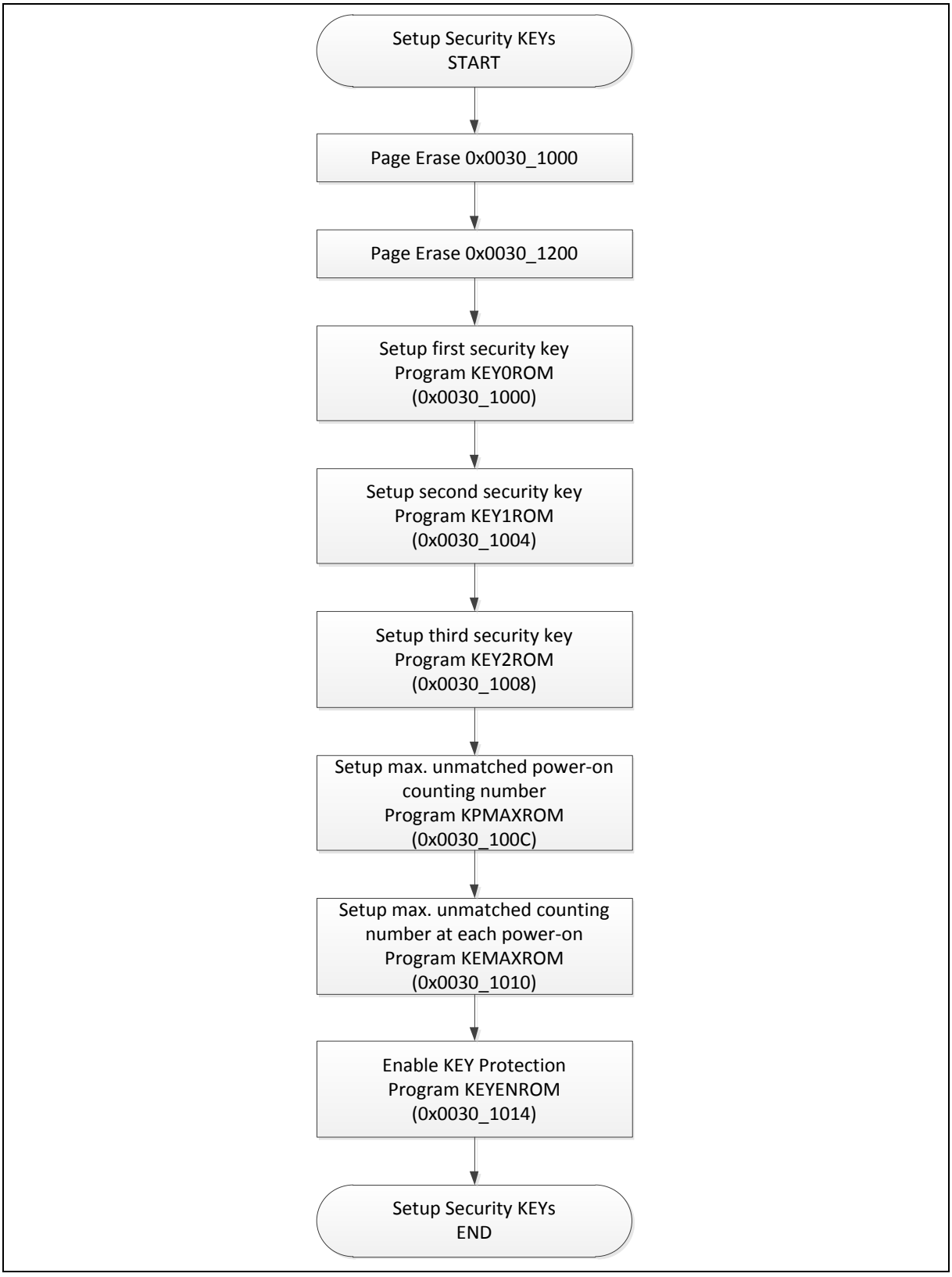


图 6.5-26Flash 安全密匙设置流程

密匙比较操作流程

密钥比较功能用来验证KPROM里的用户密钥和安全密钥。具体操作见图6.5-27。

1. 向寄存器FMC_ KPKEY0/FMC_ KPKEY1/FMC_ KPKEY2写入用户密钥。
2. 置KPKEYGO (FMC_KPKEYTRG [0])位为1开始密钥比较操作。
3. FMC控制器读取KPROM里的安全密钥并与用户密钥作对比。

若密钥匹配，不匹配计数寄存器(FMC_ KPKECNT)会清零，KPROM的KPCNTROM页会被擦除。

若密钥不匹配，不匹配计数寄存器(FMC_KPKECNT和FMC_KPCNT)加1，KPROM的KPCNTROM页会被写0。

4. 硬件更新状态标识和比较结果到FMC_KPKEYSTS寄存器供用户读取。

如果密钥匹配，KEYMATCH(FMC_ KPKEYSTS [2])位置1，KEYLOCK(FMC_ KPKEYSTS [1])清零。KETLOCK会保持0大概10~30分钟。当KEYFLAG(FMC_ KPKEYSTS [4])激活时，KEYLOCK会自动返回1。

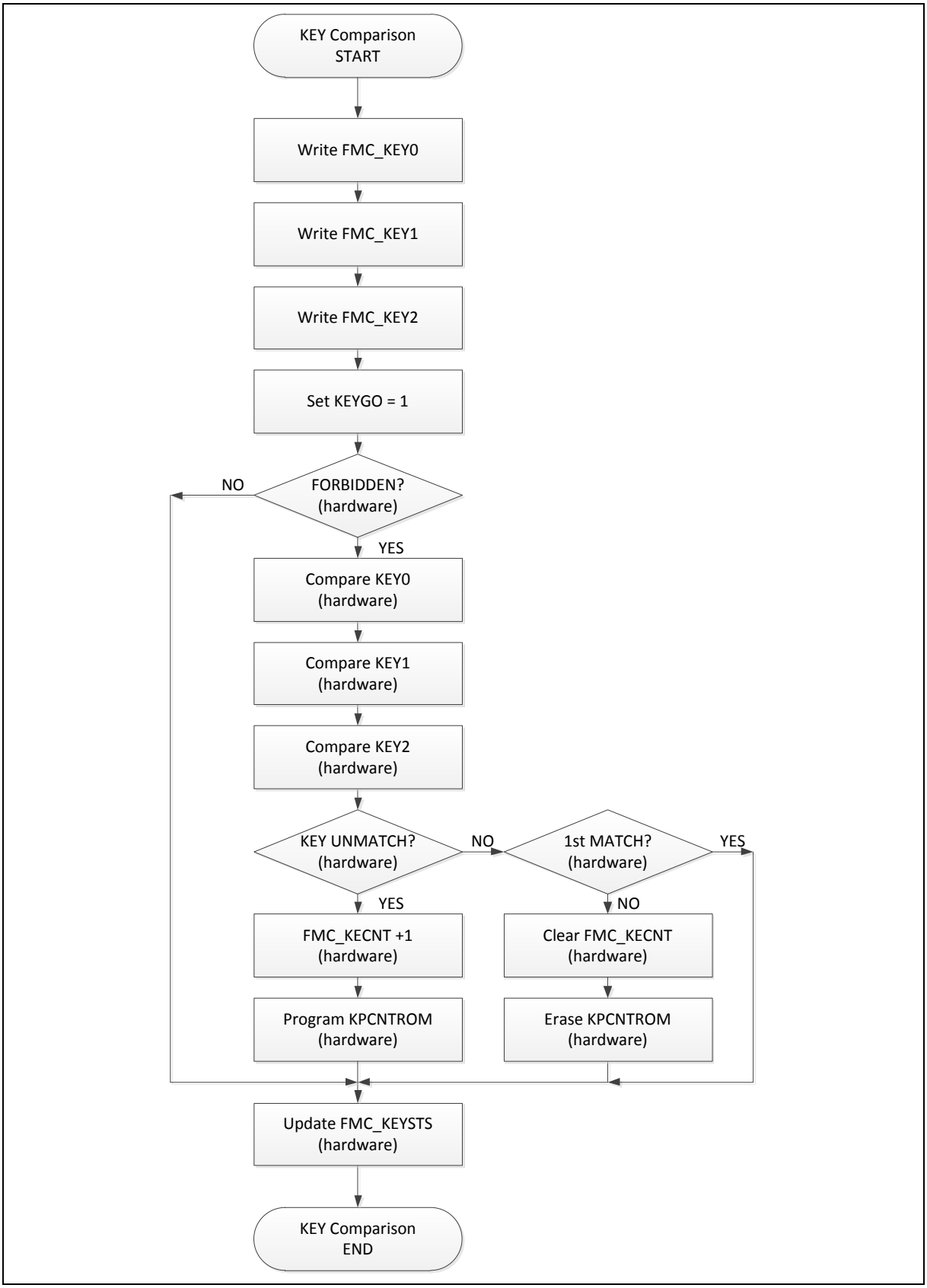

KEY Comparison START
Write FMC_KEY0
Write FMC_KEY1
Write FMC_KEY2
Set KEYGO = 1
FORBIDDEN? (hardware)
NO
YES
Compare KEY0 (hardware)
Compare KEY1 (hardware)
Compare KEY2 (hardware)
KEY UNMATCH? (hardware)
NO
1st MATCH? (hardware)
YES
YES
NO
FMC_KECNT +1 (hardware)
Clear FMC_KECNT (hardware)
Program KPCNTROM (hardware)
Erase KPCNTROM (hardware)
Update FMC_KEYSTS (hardware)
KEY Comparison END

图 6.5-27KEY 比较流程

6.5.4.20 锁定效果表

M480系列支持四种保护，包括安全锁控制（也就是 CONFIG0[1]中的LOCK 和CONFIG2[7:0]中的ALOCK ）、安全密匙保护功能（也就是KPROM 和KEYLOCK）、安全启动（也就是CONFIG2[15:8])中的SBLOCK ）和安全保护存储（也就是 SPROM 和 SPLOCK)。在这个章节M480准备了一些锁定效果表，帮助用户理解在APROM, LDROM, SPROM CONFIG 和其他以上四种对CPU, ICE 和 ICP/Writer的保护的锁定效果。

LOCK/ALOCK	**OFF**		**ON**		**OFF**		**OFF**		**OFF**					
KEYLOCK	OFF		OFF		OFF		OFF		ON					
KPKEYENROM[1:0]	X		X		X		X		00		01		10	
SBLOCK	OFF		OFF		ON		OFF		OFF					
SPLOCK	OFF		OFF		OFF		ON		OFF					
APROM	R	W	R	W	R	W	R	W	R	-	R	-	R	-
Data Flash	R	W	R	W	R	W	R	W	R	W	R	W	R	W
LDROM	R	W	R	W	R	W	R	W	R	-	R	-	R	-
SPROM	R	W	R	W	R	W	R_1	W_1	R	-	R	-	R	W
CFG	R	W	R	W	R	W	R	W	R	-	R	W	R	-
KPROM	-	W	-	W	-	W	-	W	-	-	-	-	-	-
OTP	R	W_2	R	W_2	R	W_2	R	W_2	R	W_2	R	W_2	R	W_2
Secure Boot Key	-	W	-	W	-	W	-	W	-	W	-	W	-	W
Boot Loader	R	-	R	-	R	-	R	-	R	-	R	-	R	-
DID/UID/Checksum	R	-	R	-	R	-	R	-	R	-	R	-	R	-

表 6.5-7 对 CPU 的四种保护的锁定效果表

表6.5-7和表6.5-8列出对ICE/ ICP/Writer的四种保护的锁定效果表

LOCK/ALOCK	**OFF**		**ON**		**OFF**		**OFF**		**OFF**					
KEYLOCK	OFF		OFF		OFF		OFF		ON					
KPKEYENROM[1:0]	X		X		X		X		00		01		10	
SBLOCK	OFF		OFF		ON		OFF		OFF					
SPLOCK	OFF		OFF		OFF		ON		OFF					
APROM	R	W	-	-	-	-	R	W	R	-	R	-	R	-
Data Flash	R	W	-	-	-	-	R	W	R	W	R	W	R	W
LDROM	R	W	-	-	-	-	R	W	R	-	R	-	R	-
SPROM	R	W	-	-	-	-	D	-	R	-	R	-	R	W

CFG	R	W	R	-	R	-	R	W	R	-	R	W	R	-
OTP	R	W_2	-	-	-	-	R	W_2	R	W_2	R	W_2	R	W_2
KPROM	-	W	-	-	-	-	-	W	-	-	-	-	-	-
Secure Boot Key	-	W	-	-	-	-	-	W	-	W	-	W	-	W
Boot Loader	-	-	-	-	-	-	-	-	-	-	-	-	-	-
DID/UID/Checksum	R	-	R	-	R	-	R	-	R	-	R	-	R	-

表 6.5-8 对 ICE/ ICP/Writer 的四种保护的锁定效果表

注:

1. 符号 “R”意思是可读，符号 “W” 意思是可写。
2. 符号 “X” 意思是不管值是多少。
3. 符号 “R_1” 意思是 SPROM 对于 CPU 指令可读，但是对于CPU数据或ISP命令不可读。符号 “W_1” 意思是 SPROM 不能编程，但是可以也擦除。符号“D” 意思是 SPROM 可以在调试模式用于 ICE (不可读, 返回 0x0000_0000)。
4. 符号 “W_2” 意思是 OTP 仅当对应的 “LOCK BIT” 是 0xFFFF_FFFF时可写。
5. SBLOCK = LOCK/ALOCK 对于 CPU 和 ICP/Writer。
6. 当SBLOCK不是0x5A，对于ICE所有操作 (除了批量擦除) 是无效的。
7. 批量擦除(也就是整个芯片擦除) 可以在任何时间任何地址执行，SPROM 除外。
8. 当任何保护与另外一种保护冲突，FMC会采用最严格的一个保护。例如，当KEYLOCK 打开，对于ICE APROM 是可读的，一旦KEYLOCK 和 LOCK都打开对于ICE APROM 是不可读的。
9. 对于CPU, ICE 和 ICP/Writer，KPROM 和安全启动密匙是不可读的。
10. CE 和 ICP/Writer， Boot loader 是不可读的。

6.5.5 寄存器映射

R: 只读, **W**: 只写, **R/W**: 读写

寄存器	偏移	R/W	描述	复位值
FMC 基地址 **FMC_BA = 0x4000_C000**				
FMC_ISPCTL	FMC_BA+0x00	R/W	ISP控制寄存器	0x0000_0000
FMC_ISPADDR	FMC_BA+0x04	R/W	ISP地址寄存器	0x0000_0000
FMC_ISPDAT	FMC_BA+0x08	R/W	ISP数据寄存器	0x0000_0000
FMC_ISPCMD	FMC_BA+0x0C	R/W	ISP命令寄存器	0x0000_0000
FMC_ISPTRG	FMC_BA+0x10	R/W	ISP触发控制寄存器	0x0000_0000
FMC_DFBA	FMC_BA+0x14	R	Data Flash基地址	0xXXXX_XXXX
FMC_ISPSTS	FMC_BA+0x40	R/W	ISP状态寄存器	0x0000_0000
FMC_CYCCTL	FMC_BA+0x4C	R/W	Flash访问周期控制寄存器	0x0000_0000
FMC_KPKEY0	FMC_BA+0x50	W	KPROM KEY0数据寄存器	0x0000_0000
FMC_KPKEY1	FMC_BA+0x54	W	KPROM KEY1数据寄存器	0x0000_0000
FMC_KPKEY2	FMC_BA+0x58	W	KPROM KEY2数据寄存器	0x0000_0000
FMC_KPKEYTRG	FMC_BA+0x5C	R/W	KPROM密钥比较触发控制寄存器	0x0000_0000
FMC_KPKEYSTS	FMC_BA+0x60	R/W	KPROM KEY 密钥比较状态寄存器	0x0000_0000
FMC_KPKEYCNT	FMC_BA+0x64	R	KPROM 密钥不匹配计数寄存器	0x0000_XX00
FMC_KPCNT	FMC_BA+0x68	R	KPROM 密钥不匹配上电计数寄存器	0x0000_0X00
FMC_MPDAT0	FMC_BA+0x80	R/W	ISP Data0寄存器	0x0000_0000
FMC_MPDAT1	FMC_BA+0x84	R/W	ISP Data1寄存器	0x0000_0000
FMC_MPDAT2	FMC_BA+0x88	R/W	ISP Data2寄存器	0x0000_0000
FMC_MPDAT3	FMC_BA+0x8C	R/W	ISP Data3寄存器	0x0000_0000
FMC_MPSTS	FMC_BA+0xC0	R	ISP多字编程状态寄存器	0x0000_0000
FMC_MPADDR	FMC_BA+0xC4	R	ISP 多字编程地址寄存器	0x0000_0000
FMC_XOMR0STS	FMC_BA+0xD0	R	XOM区域0状态寄存器	0x0FF0_00FF
FMC_XOMR1STS	FMC_BA+0xD4	R	XOM区域1状态寄存器	0x0FF0_00FF
FMC_XOMR2STS	FMC_BA+0xD8	R	XOM区域2状态寄存器	0x0FF0_00FF
FMC_XOMR3STS	FMC_BA+0xDC	R	XOM区域3状态寄存器	0x0FF0_00FF

FMC_XOMSTS	FMC_BA+0xE0	R	XOM状态寄存器	0x0000_0000

6.5.6 寄存器描述

ISP控制寄存器 (FMC_ISPCTL)

寄存器	偏移	R/W	描述	复位值
FMC_ISPCTL	FMC_BA+0x00	R/W	ISP控制寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							BL
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved	ISPFF	LDUEN	CFGUEN	APUEN	Reserved	BS	ISPEN

位	描述	
[31:17]	**Reserved**	保留。
[16]	**BL**	**Boot Loader 启动 (写保护)** 该位被初始化为MBS (CONFIG0[5])的取反值。除了CPU 复位或系统复位，任何复位BL都将重载初值。该位用于检查芯片是否从 Boot Loader 启动。当更新FMC_ISPCTL寄存器的时候，用户应该保持该位的初始值。 0 = 从 APROM 或 LDROM启动。 1 = 从 Boot Loader启动。 **注:**该位写保护，参见SYS_REGLCTL寄存器。
[15:8]	**Reserved**	保留。
[7]	**Reserved**	保留。

[6]	**ISPFF**	**ISP失败标识(写保护)** 当已触发的ISP符合以下条件之一，该由硬件位置1： 该位写1清0。 ● APUEN为0时，APROM写APROM。 ● KEYLOCK为1时，APROM(除了 Data Flash) 进行擦除/编程。 ● LDUEN为0时，LDROM写LDROM。 ● KEYLOCK为1时，LDROM 被擦写或编程。 ● CFGUEN为0时，CONFIG被擦写或编程。 ● KEYLOCK 和KEYENROM[1:0] 为1时，CONFIG被擦写或编程。 ● KEYLOCK 为1时，KPROM 被擦写或编程。 ● 欠压检测到低电压时，擦除或编程。 ● 目标地址非法，例如超出有效范围。 ● 无效的ISP命令。 ● 在LOCK模式并连接ICE,收到页擦除命令。 ● SBLOCK 不为0x5A,连接ICE进行非法操作（除了芯片擦除）。 ● ICE连接状态下，读boot loader的任何内容。 ● 新XOM区域的基地址和大小错误，区域重叠或者写了两次。 ● XOM的页擦除功能设定错误。 ● 起效的XOM区域被访问（除了芯片擦除，页擦除，校验和读CID/DID）。 ● XOM设置页被访问（除了芯片擦除，字编程和读）。 ● 侵犯载入代码的读保护。 ● OTP 擦除，多字编程或64-位读。 ● 校验和或者Flash全为1校验没有被执行在他们有效的范围内。 ● 在APROM中区域擦除没有被执行。 **注:** 该位写保护，参见SYS_REGLCTL寄存器。
[5]	**LDUEN**	**LDROM 更新使能位 (写保护)** 0 = LDROM 不能被更新。 1 = LDROM 可以被更新。 **注:**该位写保护，参见SYS_REGLCTL寄存器。
[4]	**CFGUEN**	**CONFIG更新使能位(写保护)** 0 = CONFIG 不能被更新。 1 = CONFIG可以被更新。 **注:**该位写保护，参见寄存器SYS_REGLCTL。
[3]	**APUEN**	**APROM更新使能位(写保护)** 0 = 当芯片运行在APROM时，APROM不能被更新。 1 = 当芯片运行在APROM时，APROM可以被更新。 **注:**该位写保护，参见寄存器SYS_REGLCTL。
[2]	**Reserved**	保留。

[1]	**BS**	**启动选择（写保护）** 当CONFIG0中的MBS为1时，置位/清零该位选择下次由LDROM/APROM启动。该位也可以作为MCU启动状态标识位，用来检查芯片启动源。除了CPU复位和系统复位，任何其他复位后，该位都会被初始化为寄存器CBS[1]（CONFIG0[7]）的反转值。 0 = 当 MBS（CONFIG0[5]）是1，从APROM启动。 1 = 当 MBS（CONFIG0[5]）是1，从LDROM启动。 **注：**该位是写保护的。请参考SYS_REGLCTL寄存器。
[0]	**ISPEN**	**ISP 使能位（写保护）** ISP使能位。设置该位使能ISP功能。 0 = ISP功能禁止。 1 = ISP功能使能。 **注：**该位是写保护的。请参考SYS_REGLCTL寄存器。

ISP 地址 (FMC_ISPADDR)

寄存器	偏移	R/W	描述	复位值
FMC_ISPADDR	FMC_BA+0x04	R/W	ISP地址寄存器	0x0000_0000

31	30	29	28	27	26	25	24
ISPADDR							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
ISPADDR							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
ISPADDR							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
ISPADDR							

位	描述	
[31:0]	**ISPADDR**	**ISP 地址** M480系列内嵌FLASH, ISP 32-位操作时，ISPADDR[1:0]必须为00；ISP 64-位操作时，ISPADDR[2:0]必须为000。 对于CRC32 校验和计算命令，该地址是Flash 校验和计算的起始地址，需要4K字节对齐。 对于Flash的32-位编程，ISP地址需要字对齐（4字节）。对于Flash的64-位编程，ISP地址需要双字对齐（8字节）。

ISP 数据寄存器 (FMC_ISPDAT)

寄存器	偏移	R/W	描述	复位值
FMC_ISPDAT	FMC_BA+0x08	R/W	ISP数据寄存器	0x0000_0000

31	30	29	28	27	26	25	24
ISPDAT							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
ISPDAT							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
ISPDAT							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
ISPDAT							

位	描述	
[31:0]	**ISPDAT**	**ISP 数据** ISP写操作之前，写数据到该寄存器。 ISP读操作后，可从该寄存器读数据。 当 ISPFF (FMC_ISPCTL[6]) 是 1, ISPDAT = 0xffff_ffff 执行CRC32 Checksum计算时，ISPDAT代表内存大小(字节)，4 K字节对齐。 读ISP Checksum命令时，ISPDAT代表Checksum的计算结果。 ISPDAT = 0x0000_0000，代表(1)还在Checksum计算中，或(2) Checksum计算的内存范围出错。

ISP 命令寄存器(FMC_ISPCMD)

寄存器	偏移	R/W	描述	复位值
FMC_ISPCMD	FMC_BA+0x0C	R/W	ISP命令寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved	CMD						

位	描述	
[31:7]	Reserved	保留。
[6:0]	CMD	ISP 命令 命令表如下： 0x00=读FLASH。 0x04= 读 UID。 0x08= 读 Flash 位全为1校验结果。 0x0B=读公司 ID。 0x0C=读设备ID。 0x0D=读Checksum。 0x21= FLASH 32-位编程。 0x22= FLASH 页擦除.，擦除两个bank中的任意页， OTP除外。 0x23= FLASH Bank 擦除，擦除BANK0 或 BANK1中APROM的所有页。 0x25= FLASH 块擦除，擦除BANK0 或 BANK1中APROM的四页（四页对齐）。 0x27= FLASH 多-字编程。 0x28= 执行 Flash 位全为1校验。 0x2D= 执行 Checksum 计算。 0x2E=向量重映射。 0x40= FLASH 64-位读。 0x61= FLASH 64-位编程。 其他是无效指令。

ISP 触发控制寄存器 (FMC_ISPTRG)

寄存器	偏移	R/W	描述	复位值
FMC_ISPTRG	FMC_BA+0x10	R/W	ISP触发控制寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved							ISPGO

位	描述	
[31:1]	Reserved	保留。
[0]	ISPGO	**启动ISP触发寄存器（写保护）** 写 1 开始ISP操作，当ISP操作结束后，该位由硬件自动清零。 0 = ISP 操作结束。 1 = ISP 正在执行。 **注：** 该位是写保护位。参考SYS_REGLCTL寄存器。

Data Flash基地址寄存器 (FMC_DFBA)

寄存器	偏移	R/W	描述	复位值
FMC_DFBA	FMC_BA+0x14	R	Data Flash基地址	0xXXXX_XXXX

31	30	29	28	27	26	25	24
DFBA							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
DFBA							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
DFBA							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
DFBA							

位	描述	
[31:0]	**DFBA**	**Data Flash基地址** 该寄存器为数据FLASH开始地址寄存器，只读。 数据Flash与APROM共用存储空间。从CONFIG1加载寄存器内容。 当DFEN (CONFIG0[0]) =0该寄存器有效。

Flash存取时间控制寄存器 (FMC_FTCTL)

寄存器	偏移	R/W	描述	复位值
FMC_FTCTL	FMC_BA+0x18	R/W	Flash存取时间控制寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved						**CACHEINV**	**Reserved**
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved							

位	描述	
[31:10]	**Reserved**	保留。
[9]	**CACHEINV**	**FLASH缓存失效（写保护）** 0 = Flash缓存失效结束（默认）。 1 = Flash缓存失效。 **注1：** 写1启动缓存失效。当运行完成后该位清零。 **注2：** 该位写保护。请参考SYS_REGLCTL寄存器。
[8:0]	**Reserved**	保留。

ISP 状态寄存器 (FMC_ISPSTS)

寄存器	偏移	R/W	描述	复位值
FMC_ISPSTS	FMC_BA+0x40	R/W	ISP状态寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
VECMAP							
15	14	13	12	11	10	9	8
VECMAP							Reserved
7	6	5	4	3	2	1	0
ALLONE	ISPFF	PGFF	FCYCDIS	MBS	CBS		ISPBUSY

位	描述	
[31:24]	Reserved	保留。
[23:9]	VECMAP	向量页映射地址（只读） 当前 flash 地址空间0x0000_0000~0x0000_01FF映射到地址{VECMAP[14:0], 9'h000} ~ {VECMAP[14:0], 9'h1FF}。
[8]	Reserved	保留。
[7]	ALLONE	Flash 全为1校验标志 在执行Flash位全为1校验后，如果Flash位全为1，该位由硬件置1，如果不是全为1，该位由硬件清0。该位也可以通过写1清0。 0 =在执行Flash位全为1校验后，所有Flash位不全为1 。 1 =在执行Flash位全为1校验后，所有Flash位全为1。

[6]	**ISPFF**	**ISP失败标识(写保护)** 该位是 ISPFF (FMC_ISPCTL[6])的镜像，需要写1到FMC_ISPCTL[6] 或 FMC_ISPSTS[6]来清0 。当已触发的ISP符合以下条件之一，该由硬件位置1： ● APUEN为0时，APROM写APROM。 ● KEYLOCK为1时，APROM(除了 Data Flash) 进行擦除/编程。 ● LDUEN为0时，LDROM写LDROM。 ● KEYLOCK为1时，LDROM 被擦写或编程。 ● CFGUEN为0时，CONFIG被擦写或编程。 ● KEYLOCK 和KEYENROM[1:0] 为1时，CONFIG被擦写或编程。 ● KEYLOCK 为1时，KPROM 被擦写或编程。 ● 欠压检测到低电压时，擦除或编程。 ● 目标地址非法，例如超出有效范围。 ● 无效的ISP命令。 ● 在LOCK模式并连接ICE,收到页擦除命令。 ● SBLOCK 不为0x5A,连接ICE进行非法操作（除了芯片擦除）。 ● ICE连接状态下，读boot loader的任何内容。 ● 新XOM区域的基地址和大小错误，区域重叠或者写了两次。 ● XOM的页擦除功能设定错误。 ● 起效的XOM区域被访问（除了芯片擦除，页擦除，校验和读CID/DID）。 ● XOM设置页被访问（除了芯片擦除，字编程和读）。 ● 侵犯载入代码的读保护。 ● OTP 擦除，多字编程或64-位读。 ● 校验和或者Flash全为1校验没有被执行在他们有效的范围内。 ● 在APROM中区域擦除没有被执行。 **注:** 该位写保护，参见SYS_REGLCTL寄存器。
[5]	**PGFF**	**快速校验标志（只读）** 如果ISP校验数据不匹配时，该位置1。当执行ISP Flash擦除或读CID操作时，该位被清零。 0 = Flash 编程成功。 1 = Flash编程失败。编程数据与Flash内存的数据不同。
[4]	**FCYCDIS**	**Flash 访问周期自动调整禁止标志 (只读)** 如果flash访问周期自动调整功能禁止，该位置1。 自动调整功能通过FADIS(FMC_CYCCTL[8])位禁止或HIRC时钟没有稳定禁止。 0 = Flash访问周期自动调整使能。 1 = Flash访问周期自动调整禁止。
[3]	**MBS**	**从Boot Loader启动的标志（只读）** 除了CPU 复位(CPU= 1)及system 复位 (SYS)外的任何复位发生时，该位由MBS (CONFIG0[5])初始化。 0 =从Boot Loader启动。 1 = 从LDROM/APROM启动（参考CBS位设定）。

[2:1]	**CBS**	**启动选择配置（只读）** 除了CPU 复位(CPU= 1)及系统复位 (SYS)外的任何复位发生时，该位由CBS (CONFIG0[7:6])初始化。 当MBS (FMC_ISPSTS[3])= 1时，如下功能生效。 00 = LDROM支持 IAP模式。 01 = LDROM不支持 IAP模式。 10 = APROM 支持IAP模式。 11 = APROM 不支持IAP模式。
[0]	**ISPBUSY**	**ISP 忙标志位（只读）** 写1开始ISP流程，当ISP操作结束后，该位由硬件自动清零。 该位是ISPGO(FMC_ISPTRG[0])的镜像。 0 =ISP操作结束。 1 = ISP 正在进行。

Flash访问周期控制寄存器 (FMC_CYCCTL)

寄存器	偏移	R/W	描述	复位值
FMC_CYCCTL	FMC_BA+0x4C	R/W	Flash访问周期控制寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							FADIS
7	6	5	4	3	2	1	0
Reserved				CYCLE			

位	描述	
[31:9]	Reserved	保留。
[8]	FADIS	**Flash访问周期自动调整禁用位（写保护）** 设置该位禁用Flash访问周期自动调整功能。 0 = Flash访问周期自动调整使能。 1 = Flash访问周期自动调整禁用。 **注:** 该位写保护，参见SYS_REGLCTL寄存器。当FMC在自动调整，在准备用ISP操作时，需要监控是否完成调整的标志。
[7:3]	Reserved	保留。

[3:0]	**CYCLE**	**Flash 访问周期控制(写保护)** 当FCYCDIS (FMC_ISPSTS[4]) 是 0，该寄存器由硬件自动更新，当自动调整功能禁止 (FADIS (FMC_CYCTL[8]) 是1），该寄存器由软件更新。 0000 = CPU 访问零等待，Flash 访问周期是 1； HCLK 工作频率范围是<27MHz，cache由硬件禁止 0001 =如果没用cache， CPU 访问等待1个周期；Flash访问周期是1； HCLK 工作频率范围是<27MHz 0010 = 如果没用cache， CPU 访问等待2个周期；Flash访问周期是2； 最优的 HCLK 工作频率范围是27~54 MHz 0011 =如果没用cache， CPU 访问等待3个周期；Flash访问周期是3； 最优的 HCLK 工作频率范围是 54~81MHz 0100 =如果没用cache， CPU 访问等待4个周期；Flash访问周期是4； 最优的 HCLK 工作频率范围是81~108MHz 0101 =如果没用cache， CPU 访问等待5个周期；Flash访问周期是5； 最优的 HCLK 工作频率范围是108~135MHz 0110 =如果没用cache， CPU 访问等待6个周期；Flash访问周期是6； 最优的 HCLK 工作频率范围是135~162MHz 0111 =如果没用cache， CPU 访问等待7个周期；Flash访问周期是7； 最优的 HCLK 工作频率范围是162~192MHz 1000 =如果没用cache， CPU 访问等待8个周期；Flash访问周期是8;. 最优的 HCLK 工作频率范围是>192MHz **注:**该位写保护，参见SYS_REGLCTL寄存器。

KPROM KEY0 数据寄存器 (FMC_KPKEY0)

寄存器	偏移	R/W	描述	复位值
FMC_KPKEY0	FMC_BA+0x50	W	KPROM KEY0数据寄存器	0x0000_0000

31	30	29	28	27	26	25	24
KPKEY0							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
KPKEY0							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
KPKEY0							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
KPKEY0							

位	描述	
[31:0]	**KPKEY0**	**KPROM KEY0 数据 (只写)** 在密匙比较操作之前，写 KPKEY0 数据到该寄存器。

KPROM KEY1 数据寄存器 (FMC_KPKEY1)

寄存器	偏移	R/W	描述	复位值
FMC_KPKEY1	FMC_BA+0x54	W	KPROM KEY1 数据寄存器	0x0000_0000

31	30	29	28	27	26	25	24
KPKEY1							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
KPKEY1							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
KPKEY1							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
KPKEY1							

位	描述	
[31:0]	**KPKEY1**	**KPROM KEY1数据 (只写)** 在密匙比较操作之前，写 KPKEY1 数据到该寄存器。

KPROM KEY2 数据寄存器 (FMC_KPKEY2)

寄存器	偏移	R/W	描述	复位值
FMC_KPKEY2	FMC_BA+0x58	W	KPROM KEY2 数据寄存器	0x0000_0000

31	30	29	28	27	26	25	24
KKPEY2							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
KPKEY2							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
KPKEY2							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
KPKEY2							

位	描述	
[31:0]	**KPKEY2**	**KPROM KEY2数据 (只写)** 在密匙比较操作之前，写 KPKEY2 数据到该寄存器。

KPROM KEY 比较触发控制寄存器(FMC_KPKEYTRG)

寄存器	偏移	R/W	描述	复位值
FMC_KPKEYTRG	FMC_BA+0x5C	R/W	KPROM KEY比较触发控制寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved						TCEN	KPKEYGO

位	描述	
[31:2]	Reserved	保留。
[1]	TCEN	**超时计数使能(写保护)** 0 = 超时计数禁止。 1 = 在密匙比较完成后，如果输入密匙匹配，超时计数使能。 超时时间至少10 分钟，平均20分钟左右。 **注:**该位写保护，参见SYS_REGLCTL寄存器。
[0]	KPKEYGO	**KPROM 密匙比较开始触发器(写保护)** 写1开始密匙比较操作，当密匙比较操作完成，该位会由硬件自动清0。当FORBID (FMC_KPKEYSTS [3]) 为 0，该触发器操作有效。 0 = 密匙比较操作完成。 1 = 密匙比较操作正在进行。 **注:**该位写保护，参见SYS_REGLCTL寄存器。

KPROM KEY 状态寄存器 (FMC_KPKEYSTS)

寄存器	偏移	R/W	描述	复位值
FMC_KPKEYSTS	FMC_BA+0x60	R/W	KPROM KEY密匙比较状态寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved		CFGFLAG	KEYFLAG	FORBID	KEYMATCH	KEYLOCK	KEYBUSY

位	描述	
[31:6]	Reserved	保留。
[5]	CFGFLAG	**CONFIG写保护使能位(只读)** KEYENROM[0]为0时，上电及复位会置位该位。擦写KPROM时，硬件会清0该位。 KEYENROM[0]写0时，该位由硬件置1。 0 = CONFIG写保护关闭。 1 = CONFIG写保护使能。
[4]	KEYFLAG	**密钥保护使能标志(只读)** 上电时或复位后，若KEYENROM[7:0]不为0xFF，该位置1位。KPROM被擦写时，该位由硬件自动清零。KEYENROM被写入非0xFF时，硬件自动置该位为1。 0 = 安全密匙保护关闭。 1 = 安全密匙保护使能。
[3]	FORBID	**密钥比较禁止标识(只读)** 当KPKECNT (FMC_ KPKECNT [4:0])大于KPKEMAX (FMC_KPKEY0[12:8]) 或者 KPCNT (FMC_KPCNT [2:0])大于KPMAX(FMC_KPCNT[10:8])时，该位置1。 0 = 密钥比较未禁止。 1 = 密钥比较禁止，KEYGO[FMC_KEYTRG[0]不能触发。

[2]	**KEYMATCH**	**密钥匹配标识(只读)** 密钥比较操作后，如果KEY0、KEY1和KEY2同KPROM里的96位密钥相匹配，该位置1。密钥不匹配时，该位清零。 如下情况时，该位也会清零： ● CPU写1到KEYLOCK(FMC_KPKEYSTS[1]) ● 超时事件 ● KPROM被擦除 ● KEYENROM写入非0xFF的值 ● 掉电模式下 0 = KEY0、KEY1和KEY2与KPROM的设置不匹配。 1 = KEY0、KEY1和KEY2与KPROM的设置相匹配。
[1]	**KEYLOCK**	**密钥锁定标识** KEYMATCH(FMC_KEYSTS[2])为0时，该位置1。安全密钥保护时KEYMATCH为1，则清零该位。批量擦除后，用户必须复位或上电/断电来清0该位。如下状况时，该位置1： ● CPU写1到KEYLOCK(FMC_KPKEYSTS[1]) ● 上电或复位时，KEYFLAG(FMC_KPKEYSTS[4])为1 ● KEYENROM写入非0xFF的值 ● 超时事件 ● FORBID(FMC_KPKEYSTS[3])为1 0 = KPROM, LDROM和APROM(不包含Data Flash)不处于写保护状态。 1 = KPROM, LDROM和APROM(不包含Data Flash)处于写保护状态。 SPROM写保护取决于SPFLAG。 CONFIG写保护取决于CFGFLAG。
[0]	**KEYBUSY**	**密钥比较忙标识(只读)** 0 = 密钥比较结束。 1 = 密钥比较进行中。

KPROM 密匙不匹配计数寄存器 (FMC_KPKEYCNT)

寄存器	偏移	R/W	描述	复位值
FMC_KPKEYCNT	FMC_BA+0x64	R	KPROM 密钥不匹配计数寄存器	0x0000_XX00

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved		KPKEMAX					
7	6	5	4	3	2	1	0
Reserved		KPKECNT					

位	描述	
[31:14]	Reserved	保留。
[13:8]	KPKEMAX	**每次上电错误密钥尝试次数最大值(只读)** KPKEMAX是每次上电错误密钥尝试次数的最大值。当擦除或编程KPROM的KPKEMAXROM，KPKEMAX的值会被更新。KPKEMAX用来限制KPKECNT(FMC_KPKEY0[5:0])的最大计数值。KPKECNT大于KPKEMAX时，FORBID(FMC_ KPKEYSTS [3])会被置1。
[7:6]	Reserved	保留。
[5:0]	KPKECNT	**每次上电密钥错误尝试次数(只读)** 安全密钥保护下，密钥错误则KPKECNT加一。系统上电或密钥匹配成功，KECNT清零。

KPROM 密钥不匹配上电次数寄存器(FMC_KPCNT)

寄存器	偏移	R/W	描述	复位值
FMC_KPCNT	FMC_BA+0x68	R	KPROM 密钥不匹配上电次数寄存器	0x0000_0X00

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved				KPMAX			
7	6	5	4	3	2	1	0
Reserved				KPCNT			

位	描述	
[31:12]	Reserved	保留。
[11:8]	KPMAX	**密钥尝试上电次数最大值(只读)** KPMAX是密钥尝试错误的上电次数最大值。当擦除或写KPROM的KPMAXROM， KPMAX的值也会被更新。KPMAX用来限制KPCNT(FMC_ KPCNT [3:0])的最大计数值。KPCNT大于KPMAX时，FORBID(FMC_ KPKEYSTS [3])会被置1。
[7:4]	Reserved	保留。
[3:0]	KPCNT	**密钥错误上电次数(只读)** KPCNT是安全密钥保护下密钥错误的上电次数。密钥匹配成功，KPCNT清零。

ISP 数据 0 寄存器 (FMC_MPDAT0)

寄存器	偏移	R/W	描述	复位值
FMC_MPDAT0	FMC_BA+0x80	R/W	ISP数据0 寄存器	0x0000_0000

31	30	29	28	27	26	25	24
ISPDAT0							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
ISPDAT0							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
ISPDAT0							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
ISPDAT0							

位	描述	
[31:0]	**ISPDAT0**	**ISP 数据 0** 该寄存器是对于32位/64位/多字编程的第一个32位。也是FMC_ISPDAT的镜像寄存器，这两个寄存器保持一致。

ISP数据 1 寄存器(FMC_MPDAT1)

寄存器	偏移	R/W	描述	复位值
FMC_MPDAT1	FMC_BA+0x84	R/W	ISP数据1 寄存器	0x0000_0000

31	30	29	28	27	26	25	24
ISPDAT1							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
ISPDAT1							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
ISPDAT1							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
ISPDAT1							

位	描述	
[31:0]	**ISPDAT1**	**ISP 数据 1** 该寄存器是对于64位/多字编程的第二个32位。

ISP数据 2 寄存器(FMC_MPDAT2)

寄存器	偏移	R/W	描述	复位值
FMC_MPDAT2	FMC_BA+0x88	R/W	ISP数据2 寄存器	0x0000_0000

31	30	29	28	27	26	25	24
ISPDAT2							
23	22	21	20	19	18	17	16
ISPDAT2							
15	14	13	12	11	10	9	8
ISPDAT2							
7	6	5	4	3	2	1	0
ISPDAT2							

位	描述	
[31:0]	ISPDAT2	**ISP 数据 2** 该寄存器是对于多字编程的第三个32位。

ISP数据 3 寄存器(FMC_MPDAT3)

寄存器	偏移	R/W	描述	复位值
FMC_MPDAT3	FMC_BA+0x8C	R/W	ISP数据3 寄存器	0x0000_0000

31	30	29	28	27	26	25	24
ISPDAT3							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
ISPDAT3							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
ISPDAT3							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
ISPDAT3							

位	描述	
[31:0]	**ISPDAT3**	**ISP 数据 3** 该寄存器是对于多字编程的第四个32位。

ISP 多字编程状态寄存器(FMC_MPSTS)

寄存器	偏移	R/W	描述	复位值
FMC_MPSTS	FMC_BA+0xC0	R	ISP多字编程状态寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
D3	D2	D1	D0	KEYBUSY	ISPFF	PPGO	MPBUSY

位	描述	
[31:8]	Reserved	保留。
[7]	D3	**ISP数据3 标志 (只读)** 当FMC_MPDAT3被写入数据时为1。当FMC_MPDAT3内的资料写入到Flash的动作完成时，这个位会自动清零。 0 = FMC_MPDAT3寄存器为空，编程到Flash已经结束。 1 = 数据已写入FMC_MPDAT3，但还未编程到Flash中。
[6]	D2	**ISP数据2 标志 (只读)** 当FMC_MPDAT2被写入数据时为1。当FMC_MPDAT2内的资料写入到flash的动作完成时，这个位会自动清零。 0 = FMC_MPDAT2寄存器为空，编程到flash已经结束。 1 =数据已写入FMC_MPDAT2，但还未编程到flash中。
[5]	D1	**ISP数据1 标志 (只读)** 当FMC_MPDAT1被写入数据时为1。当FMC_MPDAT1内的资料写入到flash的动作完成时，这个位会自动清零。 0 = FMC_MPDAT1寄存器为空，编程到flash已经结束。 1 =数据已写入FMC_MPDAT1，但还未编程到flash中。
[4]	D0	**ISP数据0 标志 (只读)** 当FMC_MPDAT0被写入数据时为1。当FMC_MPDAT0内的资料写入到flash的动作完成时，这个位会自动清零。 0 = FMC_MPDAT0寄存器为空，编程到flash已经结束。 1 =数据已写入FMC_MPDAT0，但还未编程到flash中。
[3]	KEYBUSY	KEY比较进行中 (只读) 0 = KEY 比较完成。 1 = KEY 比较进行中。

[2]	**ISPFF**	**ISP 失败标志(只读)** 该位是寄存器ISPFF (FMC_ISPCTL[6])的镜像，需向FMC_ISPCTL[6] 或FMC_ISPSTS[6]写1清零。当已启动的ISP符合下面的任何一个条件时，该位由硬件置1： ● APUEN为0时，APROM写APROM。 ● KEYLOCK为1时，APROM(除了 Data Flash) 进行擦除/编程。 ● LDUEN为0时，LDROM写LDROM。 ● KEYLOCK为1时，LDROM 被擦写或编程。 ● CFGUEN为0时，CONFIG被擦写或编程。 ● KEYLOCK 和KEYENROM[1:0] 为1时，CONFIG被擦写或编程。 ● KEYLOCK 为1时，KPROM 被擦写或编程。 ● 欠压检测到低电压时，擦除或编程。 ● 目标地址非法，例如超出有效范围。 ● 无效的ISP命令。 ● 在LOCK模式并连接ICE,收到页擦除命令。 ● SBLOCK 不为0x5A,连接ICE进行非法操作（除了芯片擦除）。 ● ICE连接状态下，读boot loader的任何内容。 ● 新XOM区域的基地址和大小错误，区域重叠或者写了两次。 ● XOM的页擦除功能设定错误。 ● 起效的XOM区域被访问（除了芯片擦除，页擦除，校验和读CID/DID）。 ● XOM设置页被访问（除了芯片擦除，字编程和读）。 ● 侵犯载入代码的读保护。 ● OTP 擦除，多字编程或64-位读。 ● 校验和或者Flash全为1校验没有被执行在他们有效的范围内。 ● 在APROM中区域擦除没有被执行。
[1]	**PPGO**	**ISP 多字编程状态（只读）** 0 = ISP 多字编程操作未进行。 1 = ISP 多字编程操作正在执行。
[0]	**MPBUSY**	**ISP多字编程忙标志(只读)** 写1到ISPGO开始ISP多字编程操作，当ISP多字操作完成后，该位由硬件自动清零。 该位是ISPGO(FMC_ISPTRG[0]寄存器的镜像。 0 = ISP 多字编程操作完成。 1 = 正在执行ISP 多字编程操作。

ISP多字编程地址寄存器(FMC_MPADDR)

寄存器	偏移	R/W	描述	复位值
FMC_MPADDR	FMC_BA+0xC4	R	ISP多字编程地址寄存器	0x0000_0000

31	30	29	28	27	26	25	24
MPADDR							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
MPADDR							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
MPADDR							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
MPADDR							

位	描述	
[31:0]	**MPADDR**	**ISP多字编程地址** 当ISPGO标志为1时，MPADDR是ISP 多字编程操作的地址。当ISP多字编程完成后，MPADDR会保留ISP的最终地址。

XOM区域0状态寄存器(FMC_XOMR0STS)

寄存器	偏移	R/W	描述	复位值
FMC_XOMR0STS	FMC_BA+0xD0	R	XOM 区域0状态寄存器	0x0FF0_00FF

31	30	29	28	27	26	25	24
BASE							
23	22	21	20	19	18	17	16
BASE							
15	14	13	12	11	10	9	8
BASE							
7	6	5	4	3	2	1	0
SIZE							

位	描述	
[31:8]	BASE	**XOM区域0基地址（页对齐）** BASE是XOM区域0的基地址。
[7:0]	SIZE	**XOM区域0大小（页对齐）** SIZE是XOM区域0的大小占多少页。

XOM区域1状态寄存器(FMC_XOMR1STS)

寄存器	偏移	R/W	描述	复位值
FMC_XOMR1STS	FMC_BA+0xD4	R	XOM区域1状态寄存器	0x0FF0_00FF

31	30	29	28	27	26	25	24
BASE							
23	22	21	20	19	18	17	16
BASE							
15	14	13	12	11	10	9	8
BASE							
7	6	5	4	3	2	1	0
SIZE							

位	描述	
[31:8]	BASE	**XOM区域1基地址（页对齐）** BASE是XOM区域1的基地址。
[7:0]	SIZE	**XOM区域1大小（页对齐）** SIZE是XOM区域1的大小占多少页。

XOM区域2状态寄存器(FMC_XOMR2STS)

寄存器	偏移	R/W	描述	复位值
FMC_XOMR2STS	FMC_BA+0xD8	R	XOM区域2状态寄存器	0x0FF0_00FF

31	30	29	28	27	26	25	24
BASE							
23	22	21	20	19	18	17	16
BASE							
15	14	13	12	11	10	9	8
BASE							
7	6	5	4	3	2	1	0
SIZE							

位	描述	
[31:8]	BASE	**XOM区域2基地址（页对齐）** BASE是XOM区域2的基地址。
[7:0]	SIZE	**XOM区域2大小（页对齐）** SIZE是XOM区域2的大小占多少页。

XOM区域3状态寄存器(FMC_XOMR3STS)

寄存器	偏移	R/W	描述	复位值
FMC_XOMR3STS	FMC_BA+0xDC	R	XOM区域3状态寄存器	0x0FF0_00FF

31	30	29	28	27	26	25	24
BASE							
23	22	21	20	19	18	17	16
BASE							
15	14	13	12	11	10	9	8
BASE							
7	6	5	4	3	2	1	0
SIZE							

位	描述	
[31:8]	BASE	**XOM区域3基地址（页对齐）** BASE是XOM区域3的基地址。
[7:0]	SIZE	**XOM区域3大小（页对齐）** SIZE是XOM区域3的大小占多少页。

XOM状态寄存器(FMC_XOMSTS)

寄存器	偏移	R/W	描述	复位值
FMC_XOMSTS	FMC_BA+0xE0	R	XOM状态寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved			XOMPEF	XOMR3ON	XOMR2ON	XOMR1ON	XOMR0ON

位	描述	
[31:5]	Reserved	保留。
[4]	XOMPEF	**XOM页擦除失败** XOM页面擦除功能状态。如果XOMPEF设置为1，则用户需要再次擦除XOM区域。 0 = 成功。 1 = 失败。
[3]	XOMR3ON	**XOM区域3打开** XOM区域3启用状态。 0 = 未启用。 1 = XOM区域3启用。
[2]	XOMR2ON	**XOM区域2打开** XOM区域2启用状态。 0 = 未启用。 1 = XOM区域2启用。
[1]	XOMR1ON	**XOM区域1打开** XOM区域1启用状态。 0 = 未启用。 1 = XOM区域1启用。
[0]	XOMR0ON	**XOM区域0打开** XOM区域0启用状态。 0 = 未启用。 1 = XOM区域0启用。

6.6 通用 I/O (GPIO)

6.6.1 概述

NUC480 系列多达118 个通用I/O管脚和其他功能管脚共享，这取决于芯片的配置。118个管脚分配在PA, PB, PC, PD, PE, PF, PG 和 PH这8个端口上。PA, PB, PE和PG有16个管脚，PC，PD有15个管脚，PF，PH有12个管脚。每个管脚都是独立的，都有相应的寄存器位来控制管脚功能模式与数据。

I/O管脚的I/O类型可由软件独立地配置为输入，推挽式的输出，开漏输出或准双向模式。复位之后，所有管脚的 I/O 管脚类型取决于CIOINI (CONFIG0[10])的设置。

6.6.2 特征

- 四种 I/O 模式:
  - 准双向模式
  - 推挽输出
  - 开漏输出
  - 高阻态输入
- 可选TTL/施密特触发输入
- I/O可以配置为边沿/电平触发的中断源
- 支持高驱动力及高翻转速率的I/O模式
- 通过CIOINI (CONFIG0[10])可配置所有I/O复位之后的默认模式
  - 如果CIOIN = 0, 芯片复位后所有的GPIO管脚是准双向模式
  - 如果CIOIN = 1, 芯片复位后所有的GPIO管脚是高阻输入模式
- I/O管脚仅在准双向模式，内部上拉电阻才使能
- 使能管脚中断功能将也使能了唤醒功能

6.6.3 框图

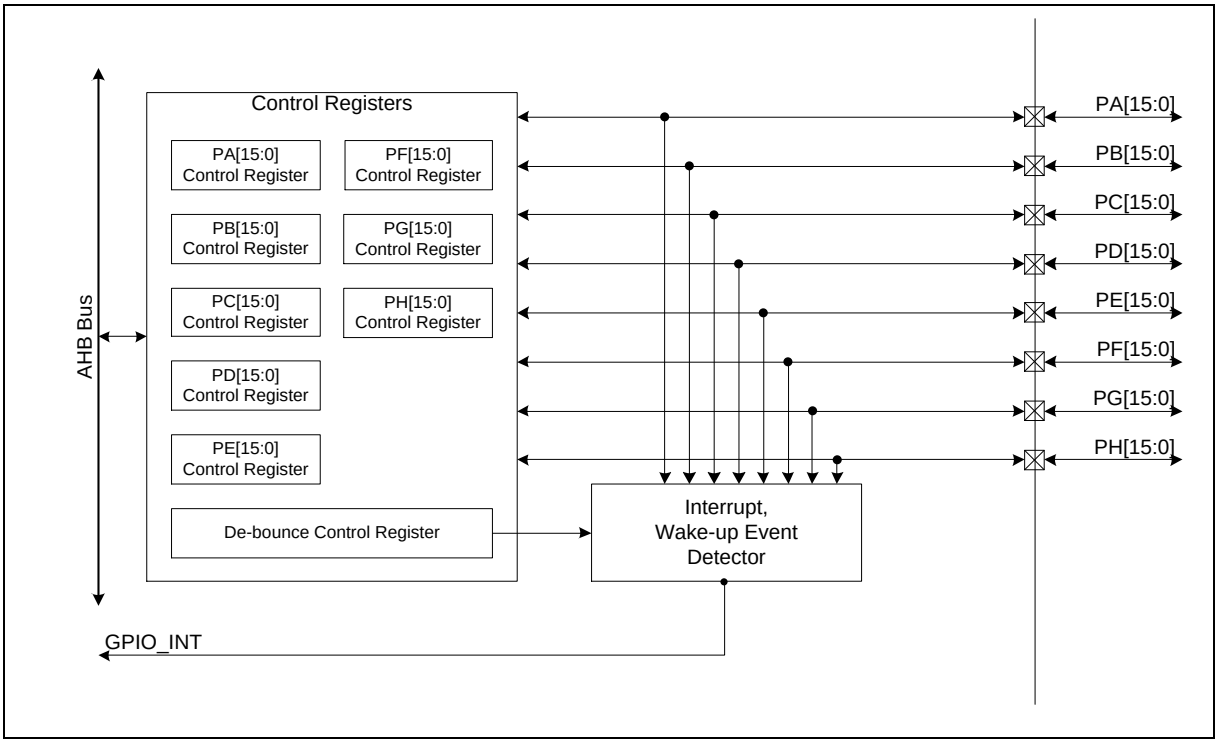


图 6.6-1GPIO 控制器框图

注:The PC.15/PD.15/PF.12/PF.13/PF.14/PF.15/PH.12/PH.13/PH.14/PH.15 管脚不可用。

6.6.4 基本配置

- 复位配置
  - 在GPIORST （SYS_IPRST1[1]）中复位GPIO
- 管脚配置

组	管脚名称	GPIO	MFP
INT0	INT0	PA.6, PB.5	MFP15
INT1	INT1	PA.7, PB.4	MFP15
INT2	INT2	PB.3, PC.6	MFP15
INT3	INT3	PB.2, PC.7	MFP15
INT4	INT4	PB.6	MFP13
		PA.8	MFP15
INT5	INT5	PB.7	MFP13
		PD.12	MFP15
INT6	INT6	PB.8	MFP13
		PD.11	MFP15

INT7	INT7	PB.9	MFP13
		PD.10	MFP15

6.6.5 功能描述

6.6.5.1 输入模式

设置 MODEn (Px_MODE[2n+1:2n]) 为00， Px.n 管脚为输入模式，I/O管脚为三态（高阻），没有输出驱动能力。管脚 (Px_PIN[n]) 的值反映相应端口的状态。

6.6.5.2 推挽输出模式

图6.6-2为推挽输出模式的框图。设置 MODEn (Px_MODE[2n+1:2n]) 为 01， Px.n 管脚为推挽输出模式，I/O支持数字输出功能，有拉/灌电流能力。DOUT (Px_DOUT[n]) 相应位[n]的值被驱动到相应管脚上。

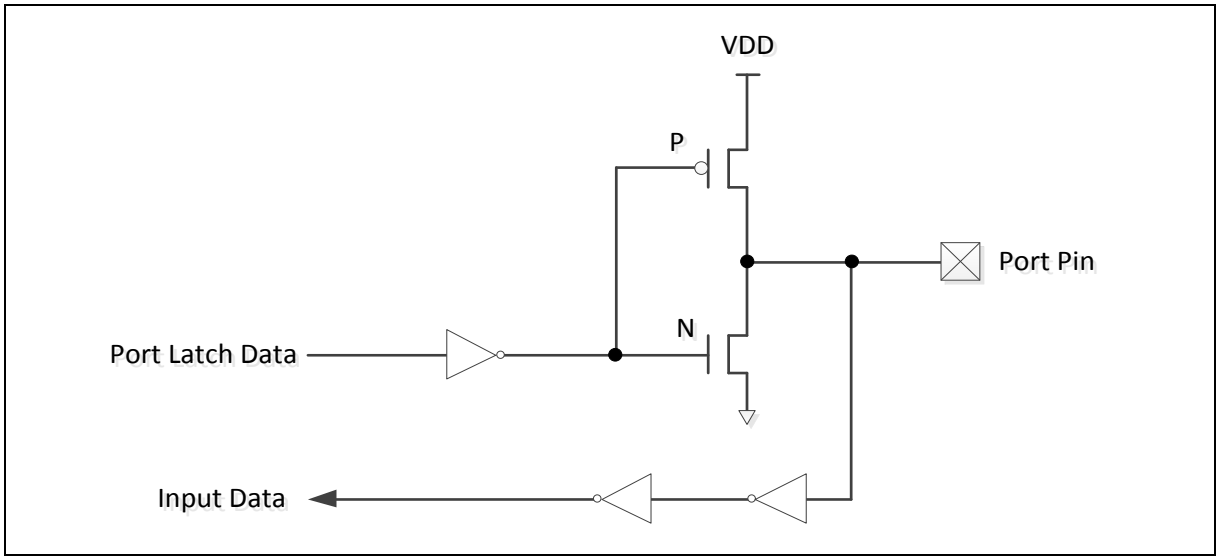


图 6.6-2 推挽输出

6.6.5.3 开漏输出模式

图6.6-3为开漏输出模式的框图。设置 MODEn (Px_MODE[2n+1:2n]) 为 10, Px.n 管脚为开漏模式，且I/O管脚数字输出功能仅支持灌电流，要驱动到高电平需要一个外加上拉电阻。如果DOUT (Px_DOUT[n]) 相应位为‘0’, 管脚上输出低；如果DOUT (Px_DOUT[n]) 相应位为‘1’, 该管脚输出高由外部上拉电阻控制。

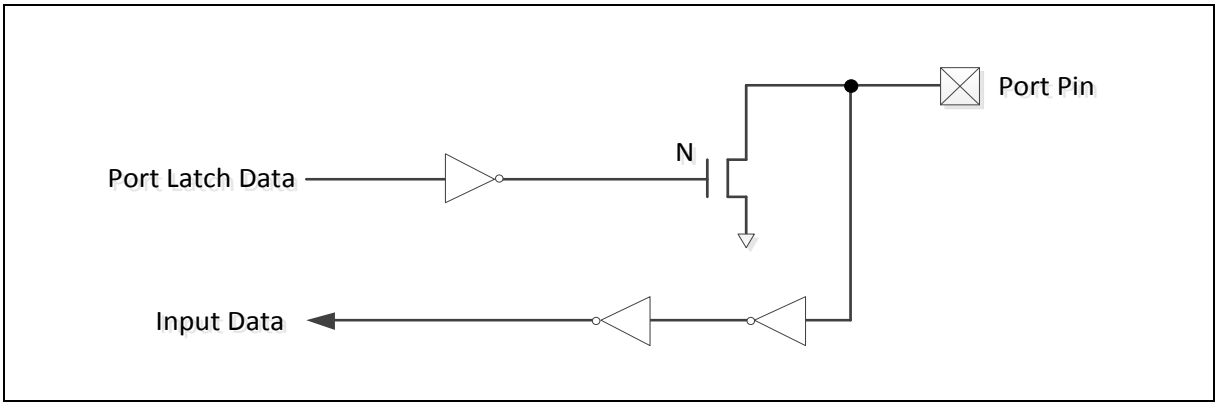


图 6.6-3 开漏输出

6.6.5.4 *准双向模式*

图6.6-4为准双向模式的框图。设置MODEn (Px_MODE[2n+1:2n])为11，Px.n 管脚为准双向模式，I/O同时支持数字输出和输入功能，但拉电流能力仅达数百uA。要实现数字输入，需要先将DOUT (Px_DOUT[n])相应位置1。准双向输出是80C51 及其派生产品常见的模式。若DOUT (Px_DOUT[n])相应位 [n]为‘0’，管脚上输出为“低”。若DOUT (Px_DOUT[n])相应位 [n]为‘1’，该管脚将检测管脚值。若管脚值为高，没有任何动作，若管脚值为低，在该管脚上将驱动2个时钟周期的高电平，然后禁止强输出驱动，其后管脚状态由内部上拉电阻控制。注意：准双向模式输出电流的大小仅有200 uA到30 uA(相应VDD的电压从5.0 V到2.5 V)。

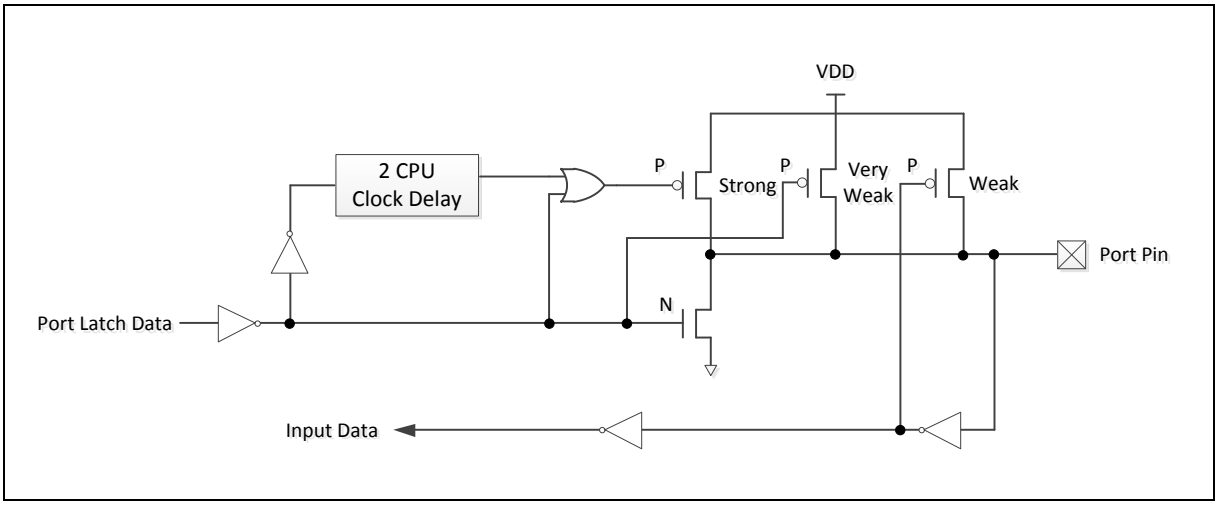


图 6.6-4 准双向 I/O 模式

6.6.5.5 *GPIO 中断和唤醒功能*

每个GPIO管脚都可以通过RHIEN (Px_INTEN[n+16])/FLIEN (Px_INTEN[n])位和TYPE (Px_INTTYPE[n])设置成芯片的中断源。有五种中断条件可以设置：低电平触发、高电平触发、下降沿触发和上升沿触发以及上升与下降沿同时触发。当芯片进入空闲/掉电模式时，GPIO也可以唤醒系统。设置GPIO为唤醒触发的条件与GPIO中断触发的条件相同。

6.6.5.6 *GPIO消抖功能*

GPIO消抖功能可以用来采样每一个GPIO管脚的中断输入，防止因干扰产生不想要的中断。GPIO消抖功能仅支持边沿触发。支持三种边沿触发条件，下降沿触发FLIEN (Px_INTEN[n])和上升沿触发RHIEN

(Px_INTEN[n+16])以及上升与下降沿同时触发TYPE (Px_INTTYPE[n])。如果用户想使用消抖功能，消抖使能控制寄存器Px_DBEN必须设置。消抖时钟源可以是HCLK 或 LIRC (10kHz)，有DBCLKSRC (Px_DBCTL[4])的设置决定。DBCLKSEL (Px_DBCTL[3:0])寄存器可以控制采样周期。

图6.6-5为GPIO上升沿触发中断。两个有效采样信号的时间间隔由DBCLKSRC (Px_DBCTL[4]) 和DBCLKSEL (Px_DBCTL[3:0])决定。IO口的有效数据需要采样两次。对于上升沿设置，如果引脚的状态在设置DBEN (Px_DBEN)之前是低。当产生一个管脚高有效数据，中断会产生。但是如果在设置DBEN (Px_DBEN)之前管脚状态是高，当先产生一个低有效数据，然后产生一个高有效数据时，中断会产生。下降沿触发中断和上升沿触发相反，图6.6-6所示。

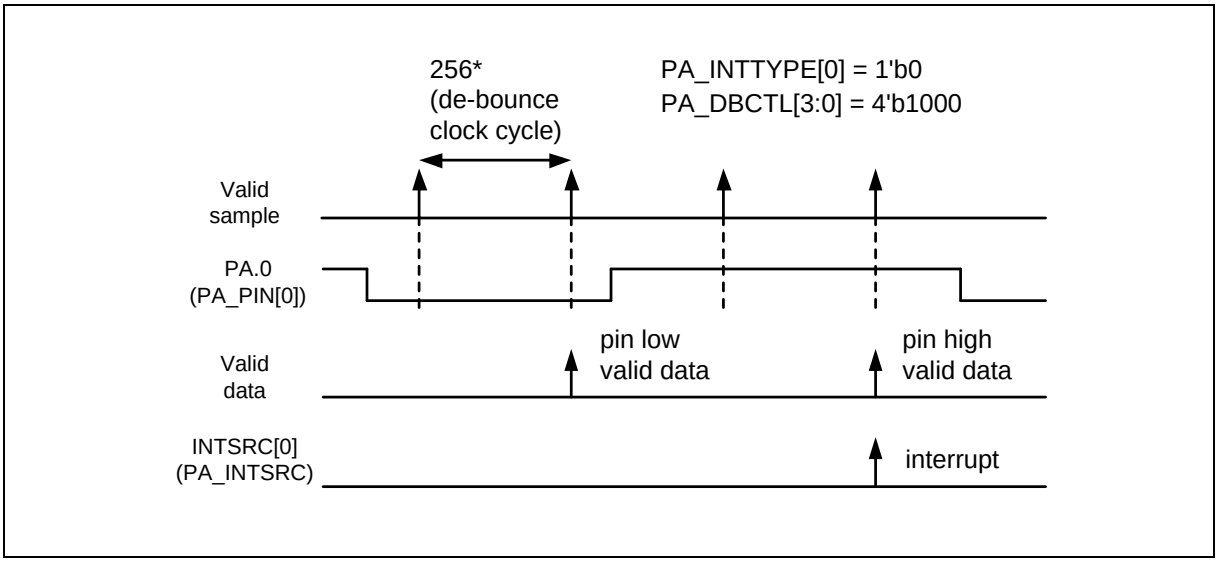


图 6.6-5 GPIO 上升沿触发中断

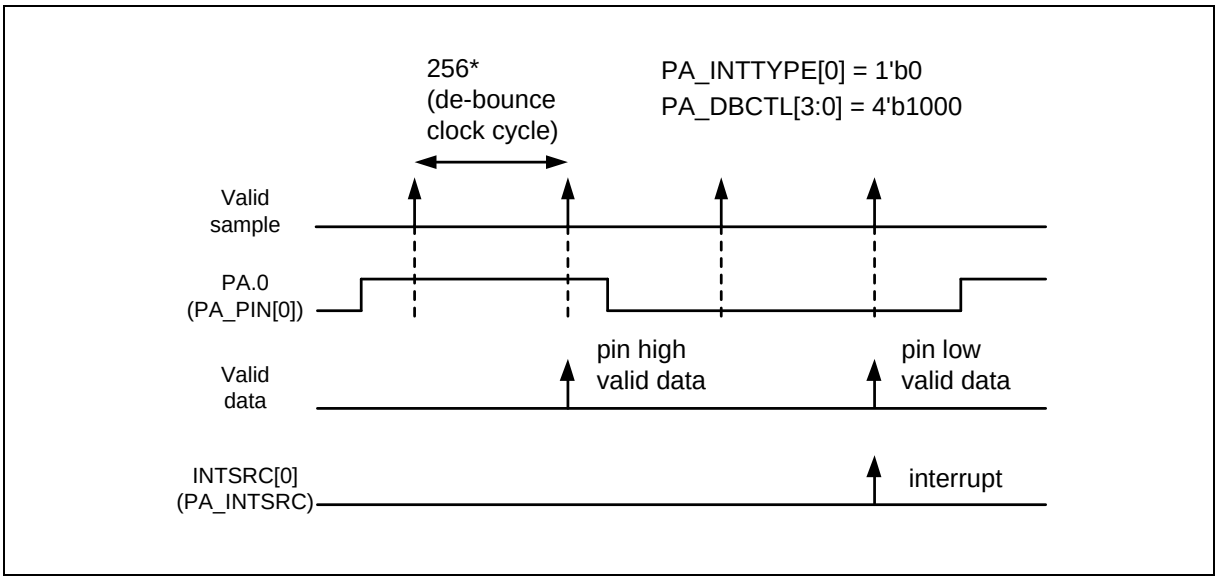


图 6.6-6 GPIO 下降沿触发中断

6.6.6 寄存器映射

R: 只读, **W**: 只写, **R/W**: 读写

寄存器	偏移	R/W	描述	复位值
GPIO 基地址: **GPIO_BA = 0x4000_4000**				
PA_MODE	GPIO_BA+0x000	R/W	PA I/O模式控制	0xXXXX_XXXX
PA_DINOFF	GPIO_BA+0x004	R/W	PA数字输入通道禁止控制	0x0000_0000
PA_DOUT	GPIO_BA+0x008	R/W	PA数据输出值	0x0000_FFFF
PA_DATMSK	GPIO_BA+0x00C	R/W	PA 数据输出写屏蔽	0x0000_0000
PA_PIN	GPIO_BA+0x010	R	PA 管脚状态值	0x0000_XXXX
PA_DBEN	GPIO_BA+0x014	R/W	PA消抖使能寄存器	0x0000_0000
PA_INTTYPE	GPIO_BA+0x018	R/W	PA中断模式控制	0x0000_0000
PA_INTEN	GPIO_BA+0x01C	R/W	PA中断使能控制	0x0000_0000
PA_INTSRC	GPIO_BA+0x020	R/W	PA中断源标志	0x0000_XXXX
PA_SMTEN	GPIO_BA+0x024	R/W	PA施密特触发输入使能寄存器	0x0000_0000
PA_SLEWCTL	GPIO_BA+0x028	R/W	PA高翻转速率控制寄存器	0x0000_0000
PA_PUSEL	GPIO_BA+0x030	R/W	PA上拉和下拉选择寄存器	0x0000_0000
PB_MODE	GPIO_BA+0x040	R/W	PB I/O模式控制	0xXXXX_XXXX
PB_DINOFF	GPIO_BA+0x044	R/W	PB数字输入通道禁止控制	0x0000_0000
PB_DOUT	GPIO_BA+0x048	R/W	PB 数据输出值	0x0000_FFFF
PB_DATMSK	GPIO_BA+0x04C	R/W	PB数据输出写屏蔽	0x0000_0000
PB_PIN	GPIO_BA+0x050	R	PB管脚状态值	0x0000_XXXX
PB_DBEN	GPIO_BA+0x054	R/W	PB消抖使能寄存器	0x0000_0000
PB_INTTYPE	GPIO_BA+0x058	R/W	PB中断模式控制	0x0000_0000
PB_INTEN	GPIO_BA+0x05C	R/W	PB中断使能控制	0x0000_0000
PB_INTSRC	GPIO_BA+0x060	R/W	PB中断源标志	0x0000_XXXX
PB_SMTEN	GPIO_BA+0x064	R/W	PB施密特触发输入使能寄存器	0x0000_0000
PB_SLEWCTL	GPIO_BA+0x068	R/W	PB高翻转速率控制寄存器	0x0000_0000
PB_PUSEL	GPIO_BA+0x070	R/W	PB上拉和下拉选择寄存器	0x0000_0000
PC_MODE	GPIO_BA+0x080	R/W	PC I/O模式控制	0xXXXX_XXXX

PC_DINOFF	GPIO_BA+0x084	R/W	PC数字输入通道禁止控制	0x0000_0000
PC_DOUT	GPIO_BA+0x088	R/W	PC数据输出值	0x0000_FFFF
PC_DATMSK	GPIO_BA+0x08C	R/W	PC数据输出写屏蔽	0x0000_0000
PC_PIN	GPIO_BA+0x090	R	PC管脚状态值	0x0000_XXXX
PC_DBEN	GPIO_BA+0x094	R/W	PC消抖使能寄存器	0x0000_0000
PC_INTTYPE	GPIO_BA+0x098	R/W	PC中断模式控制	0x0000_0000
PC_INTEN	GPIO_BA+0x09C	R/W	PC中断使能控制	0x0000_0000
PC_INTSRC	GPIO_BA+0x0A0	R/W	PC中断源标志	0x0000_XXXX
PC_SMTEN	GPIO_BA+0x0A4	R/W	PC施密特触发输入使能寄存器	0x0000_0000
PC_SLEWCTL	GPIO_BA+0x0A8	R/W	PC高翻转速率控制寄存器	0x0000_0000
PC_PUSEL	GPIO_BA+0x0B0	R/W	PC上拉和下拉选择寄存器	0x0000_0000
PD_MODE	GPIO_BA+0x0C0	R/W	PD I/O模式控制	0xXXXX_XXXX
PD_DINOFF	GPIO_BA+0x0C4	R/W	PD数字输入通道禁止控制	0x0000_0000
PD_DOUT	GPIO_BA+0x0C8	R/W	PD数据输出值	0x0000_FFFF
PD_DATMSK	GPIO_BA+0x0CC	R/W	PD数据输出写屏蔽	0x0000_0000
PD_PIN	GPIO_BA+0x0D0	R	PD管脚状态值	0x0000_XXXX
PD_DBEN	GPIO_BA+0x0D4	R/W	PD消抖使能寄存器	0x0000_0000
PD_INTTYPE	GPIO_BA+0x0D8	R/W	PD中断模式控制	0x0000_0000
PD_INTEN	GPIO_BA+0x0DC	R/W	PD中断使能控制	0x0000_0000
PD_INTSRC	GPIO_BA+0x0E0	R/W	PD中断源标志	0x0000_XXXX
PD_SMTEN	GPIO_BA+0x0E4	R/W	PD施密特触发输入使能寄存器	0x0000_0000
PD_SLEWCTL	GPIO_BA+0x0E8	R/W	PD高翻转速率控制寄存器	0x0000_0000
PD_PUSEL	GPIO_BA+0x0F0	R/W	PD上拉和下拉选择寄存器	0x0000_0000
PE_MODE	GPIO_BA+0x100	R/W	PE I/O模式控制	0xXXXX_XXXX
PE_DINOFF	GPIO_BA+0x104	R/W	PE数字输入通道禁止控制	0x0000_0000
PE_DOUT	GPIO_BA+0x108	R/W	PE数据输出值	0x0000_FFFF
PE_DATMSK	GPIO_BA+0x10C	R/W	PE数据输出写屏蔽	0x0000_0000
PE_PIN	GPIO_BA+0x110	R	PE管脚状态值	0x0000_XXXX
PE_DBEN	GPIO_BA+0x114	R/W	PE消抖使能寄存器	0x0000_0000
PE_INTTYPE	GPIO_BA+0x118	R/W	PE中断模式控制	0x0000_0000

PE_INTEN	GPIO_BA+0x11C	R/W	PE中断使能控制	0x0000_0000
PE_INTSRC	GPIO_BA+0x120	R/W	PE中断源标志	0x0000_XXXX
PE_SMTEN	GPIO_BA+0x124	R/W	PE施密特触发输入使能寄存器	0x0000_0000
PE_SLEWCTL	GPIO_BA+0x128	R/W	PE高翻转速率控制寄存器	0x0000_0000
PE_PUSEL	GPIO_BA+0x130	R/W	PE上拉和下拉选择寄存器	0x0000_0000
PF_MODE	GPIO_BA+0x140	R/W	PF I/O模式控制	0xXXXX_XXXX
PF_DINOFF	GPIO_BA+0x144	R/W	PF数字输入通道禁止控制	0x0000_0000
PF_DOUT	GPIO_BA+0x148	R/W	PF数据输出值	0x0000_FFFF
PF_DATMSK	GPIO_BA+0x14C	R/W	PF数据输出写屏蔽	0x0000_0000
PF_PIN	GPIO_BA+0x150	R	PF管脚状态值	0x0000_XXXX
PF_DBEN	GPIO_BA+0x154	R/W	PF消抖使能寄存器	0x0000_0000
PF_INTTYPE	GPIO_BA+0x158	R/W	PF中断模式控制	0x0000_0000
PF_INTEN	GPIO_BA+0x15C	R/W	PF中断使能控制	0x0000_0000
PF_INTSRC	GPIO_BA+0x160	R/W	PF中断源标志	0x0000_XXXX
PF_SMTEN	GPIO_BA+0x164	R/W	PF施密特触发输入使能寄存器	0x0000_0000
PF_SLEWCTL	GPIO_BA+0x168	R/W	PF高翻转速率控制寄存器	0x0000_0000
PF_PUSEL	GPIO_BA+0x170	R/W	PF上拉和下拉选择寄存器	0x0000_0000
PG_MODE	GPIO_BA+0x180	R/W	PG I/O模式控制	0xXXXX_XXXX
PG_DINOFF	GPIO_BA+0x184	R/W	PG数字输入通道禁止控制	0x0000_0000
PG_DOUT	GPIO_BA+0x188	R/W	PG数据输出值	0x0000_1FFF
PG_DATMSK	GPIO_BA+0x18C	R/W	PG数据输出写屏蔽	0x0000_0000
PG_PIN	GPIO_BA+0x190	R	PG管脚状态值	0x0000_XXXX
PG_DBEN	GPIO_BA+0x194	R/W	PG消抖使能寄存器	0x0000_0000
PG_INTTYPE	GPIO_BA+0x198	R/W	PG中断模式控制	0x0000_0000
PG_INTEN	GPIO_BA+0x19C	R/W	PG中断使能控制	0x0000_0000
PG_INTSRC	GPIO_BA+0x1A0	R/W	PG中断源标志	0x0000_XXXX
PG_SMTEN	GPIO_BA+0x1A4	R/W	PG施密特触发输入使能寄存器	0x0000_0000
PG_SLEWCTL	GPIO_BA+0x1A8	R/W	PG高翻转速率控制寄存器	0x0000_0000
PG_PUSEL	GPIO_BA+0x1B0	R/W	PG上拉和下拉选择寄存器	0x0000_0000
PH_MODE	GPIO_BA+0x1C0	R/W	PH I/O模式控制	0xXXXX_XXXX

PH_DINOFF	GPIO_BA+0x1C4	R/W	PH数字输入通道禁止控制	0x0000_0000
PH_DOUT	GPIO_BA+0x1C8	R/W	PH数据输出值	0x0000_1FFF
PH_DATMSK	GPIO_BA+0x1CC	R/W	PH数据输出写屏蔽	0x0000_0000
PH_PIN	GPIO_BA+0x1D0	R	PH管脚状态值	0x0000_XXXX
PH_DBEN	GPIO_BA+0x1D4	R/W	PH消抖使能寄存器	0x0000_0000
PH_INTTYPE	GPIO_BA+0x1D8	R/W	PH中断模式控制	0x0000_0000
PH_INTEN	GPIO_BA+0x1DC	R/W	PH中断使能控制	0x0000_0000
PH_INTSRC	GPIO_BA+0x1E0	R/W	PH中断源标志	0x0000_XXXX
PH_SMTEN	GPIO_BA+0x1E4	R/W	PH施密特触发输入使能寄存器	0x0000_0000
PH_SLEWCTL	GPIO_BA+0x1E8	R/W	PH高翻转速率控制寄存器	0x0000_0000
PH_PUSEL	GPIO_BA+0x1F0	R/W	PH上拉和下拉选择寄存器	0x0000_0000
GPIO_DBCTL	GPIO_BA+0x440	R/W	中断消抖控制寄存器	0x0000_0020
PAn_PDIO n=0,1..15	GPIO_BA+0x800+(0x04 * n)	R/W	GPIO PA.n管脚数据输入/输出寄存器	0x0000_000X
PBn_PDIO n=0,1..15	GPIO_BA+0x840+(0x04 * n)	R/W	GPIO PB.n管脚数据输入/输出寄存器	0x0000_000X
PCn_PDIO n=0,1..15	GPIO_BA+0x880+(0x04 * n)	R/W	GPIO PC.n管脚数据输入/输出寄存器	0x0000_000X
PDn_PDIO n=0,1..15	GPIO_BA+0x8C0+(0x04 * n)	R/W	GPIO PD.n管脚数据输入/输出寄存器	0x0000_000X
PEn_PDIO n=0,1..15	GPIO_BA+0x900+(0x04 * n)	R/W	GPIO PE.n管脚数据输入/输出寄存器	0x0000_000X
PFn_PDIO n=0,1..15	GPIO_BA+0x940+(0x04 * n)	R/W	GPIO PF.n管脚数据输入/输出寄存器	0x0000_000X
PGn_PDIO n=0,1..12	GPIO_BA+0x980+(0x04 * n)	R/W	GPIO PG.n管脚数据输入/输出寄存器	0x0000_000X
PHn_PDIO n=0,1..12	GPIO_BA+0x9C0+(0x04 * n)	R/W	GPIO PH.n管脚数据输入/输出寄存器	0x0000_000X

6.6.7 寄存器描述

端口 A-HI/O 模式控制 (Px_MODE)

寄存器	偏移	R/W	描述	复位值
PA_MODE	GPIO_BA+0x000	R/W	PA I/O模式控制	0xXXXX_XXXX
PB_MODE	GPIO_BA+0x040	R/W	PB I/O模式控制	0xXXXX_XXXX
PC_MODE	GPIO_BA+0x080	R/W	PC I/O模式控制	0xXXXX_XXXX
PD_MODE	GPIO_BA+0x0C0	R/W	PD I/O模式控制	0xXXXX_XXXX
PE_MODE	GPIO_BA+0x100	R/W	PE I/O模式控制	0xXXXX_XXXX
PF_MODE	GPIO_BA+0x140	R/W	PF I/O模式控制	0xXXXX_XXXX
PG_MODE	GPIO_BA+0x180	R/W	PG I/O模式控制	0xXXXX_XXXX
PH_MODE	GPIO_BA+0x1C0	R/W	PH I/O模式控制	0xXXXX_XXXX

31	30	29	28	27	26	25	24
MODE15		MODE14		MODE13		MODE12	
23	22	21	20	19	18	17	16
MODE11		MODE10		MODE9		MODE8	
15	14	13	12	11	10	9	8
MODE7		MODE6		MODE5		MODE4	
7	6	5	4	3	2	1	0
MODE3		MODE2		MODE1		MODE0	

位	描述	
[2n+1:2n] n=0,1..15	MODEn	**端口 A-H I/O 管脚[n]模式控制** 决定GPIO Px.n的I/O 类型。 00 = Px.n 管脚为高阻输入模式。 01 = Px.n 管脚为推挽输出模式。 10 = Px.n 管脚为开漏输出模式。 11 = Px.n 管脚为准双向模式。 **注1**:由CIOINI (CONFIG0 [10]).决定初始值，如果CIOINI置0，默认值是0xFFFF_FFFF上电后所有端口是准双向模式。如果CIOINI置1，默认值是0x0000_0000，上电后所有端口是输入模式。 **注2:** 端口A/B/E/G：最大. n=15。 端口C/D：最大. n=14。 端口F/H：最大. n=11。

端口 A-H数字输入通道关闭控制(Px_DINOFF)

寄存器	偏移	R/W	描述	复位值
PA_DINOFF	GPIO_BA+0x004	R/W	PA数字输入通道关闭控制	0x0000_0000
PB_DINOFF	GPIO_BA+0x044	R/W	PB数字输入通道关闭控制	0x0000_0000
PC_DINOFF	GPIO_BA+0x084	R/W	PC数字输入通道关闭控制	0x0000_0000
PD_DINOFF	GPIO_BA+0x0C4	R/W	PD数字输入通道关闭控制	0x0000_0000
PE_DINOFF	GPIO_BA+0x104	R/W	PE数字输入通道关闭控制	0x0000_0000
PF_DINOFF	GPIO_BA+0x144	R/W	PF数字输入通道关闭控制	0x0000_0000
PG_DINOFF	GPIO_BA+0x184	R/W	PG数字输入通道关闭控制	0x0000_0000
PH_DINOFF	GPIO_BA+0x1C4	R/W	PH数字输入通道关闭控制	0x0000_0000

31	30	29	28	27	26	25	24
DINOFF							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
DINOFF							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved							

位	描述	
[n+16] n=0,1..15	**DINOFF[n]**	**端口 A-H管脚[n]关闭数字输入通道使能** 用于控制GPIO的数字输入通路是否使能。如果输入为模拟信号，用户可以关闭输入通道防止漏电。 0 = 使能IO数据输入通道。 1 = 关闭IO的数字输入通道(数字输入拉低)。 **注意**: 端口A/B/E/G：最大n=15。 端口C/D：最大n=14。 端口F/H：最大n=11。
[15:0]	**Reserved**	保留。

端口 A-H数据输出值(Px_DOUT)

寄存器	偏移	R/W	描述	复位值
PA_DOUT	GPIO_BA+0x008	R/W	PA数据输出值	0x0000_FFFF
PB_DOUT	GPIO_BA+0x048	R/W	PB数据输出值	0x0000_FFFF
PC_DOUT	GPIO_BA+0x088	R/W	PC数据输出值	0x0000_FFFF
PD_DOUT	GPIO_BA+0x0C8	R/W	PD数据输出值	0x0000_FFFF
PE_DOUT	GPIO_BA+0x108	R/W	PE数据输出值	0x0000_FFFF
PF_DOUT	GPIO_BA+0x148	R/W	PF数据输出值	0x0000_FFFF
PG_DOUT	GPIO_BA+0x188	R/W	PG数据输出值	0x0000_1FFF
PH_DOUT	GPIO_BA+0x1C8	R/W	PH数据输出值	0x0000_1FFF

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
DOUT							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
DOUT							

位	描述	
[31:16]	**Reserved**	保留。
[n] n=0,1..15	**DOUT[n]**	**端口 A-H管脚[n]输出值** 在GPIO配置成推挽输出，开漏和准双向模式时，控制GPIO相应管脚的状态. 0 = GPIO配置成推挽输出，开漏和准双向模式时，Px.n为低。 1 = GPIO配置成推挽输出，准双向模式时，Px.n为高。 **注意：** 端口A/B/E/G：最大n=15。 端口C/D：最大n=14。 端口F/H：最大n=11。

端口 A-H数据输出写屏蔽(Px_DATMSK)

寄存器	偏移	R/W	描述	复位值
PA_DATMSK	GPIO_BA+0x00C	R/W	PA数据输出写屏蔽	0x0000_0000
PB_DATMSK	GPIO_BA+0x04C	R/W	PB数据输出写屏蔽	0x0000_0000
PC_DATMSK	GPIO_BA+0x08C	R/W	PC数据输出写屏蔽	0x0000_0000
PD_DATMSK	GPIO_BA+0x0CC	R/W	PD数据输出写屏蔽	0x0000_0000
PE_DATMSK	GPIO_BA+0x10C	R/W	PE数据输出写屏蔽	0x0000_0000
PF_DATMSK	GPIO_BA+0x14C	R/W	PF数据输出写屏蔽	0x0000_0000
PG_DATMSK	GPIO_BA+0x18C	R/W	PG数据输出写屏蔽	0x0000_0000
PH_DATMSK	GPIO_BA+0x1CC	R/W	PH数据输出写屏蔽	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
DATMSK							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
DATMSK							

位	描述	
[31:8]	**Reserved**	保留。
[n] n=0,1..15	**DATMSK[n]**	**端口 A-H管脚[n]数据输出写屏蔽** 用于保护对应DOUT (Px_DOUT[n])位，. 当设置DATMSK (Px_DATMSK[n])为 ‘1’时，相应DOUT (Px_DOUT[n])位被保护，写信号被屏蔽时，写数据到收保护的位会被忽略。 0 =相应的DOUT (Px_DOUT[n])位可以被更新。 1 =保护相应的DOUT (Px_DOUT[n])位。 **注1**：该功能只保护相应的DOUT (Px_DOUT[n])位，不保护(PDIO (Pxn_PDIO[0]))相应位。 **注2:** 端口A/B/E/G：最大 n=15。 端口C/D：最大 n=14。 端口F/H：最大 n=11。

端口 A-H管脚状态值(Px_PIN)

寄存器	偏移	R/W	描述	复位值
PA_PIN	GPIO_BA+0x010	R	PA管脚状态值	0x0000_XXXX
PB_PIN	GPIO_BA+0x050	R	PB管脚状态值	0x0000_XXXX
PC_PIN	GPIO_BA+0x090	R	PC管脚状态值	0x0000_XXXX
PD_PIN	GPIO_BA+0x0D0	R	PD管脚状态值	0x0000_XXXX
PE_PIN	GPIO_BA+0x110	R	PE管脚状态值	0x0000_XXXX
PF_PIN	GPIO_BA+0x150	R	PF管脚状态值	0x0000_XXXX
PG_PIN	GPIO_BA+0x190	R	PG管脚状态值	0x0000_XXXX
PH_PIN	GPIO_BA+0x1D0	R	PH管脚状态值	0x0000_XXXX

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
PIN							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
PIN							

位	描述	
[31:16]	**Reserved**	保留。
[n] n=0,1..15	**PIN[n]**	**端口 A-H管脚[n]状态值** 这些位的值为各个GPIO Px.n管脚真实状态的反映。如果值为‘1’，表示相应管脚状态为高，否则为低。 **注：** 端口A/B/E/G：最大n=15。 端口C/D：最大n=14。 端口F/H：最大n=11。

端口 A-H消抖使能控制寄存器(Px_DBEN)

寄存器	偏移	R/W	描述	复位值
PA_DBEN	GPIO_BA+0x014	R/W	PA消抖使能控制寄存器	0x0000_0000
PB_DBEN	GPIO_BA+0x054	R/W	PB消抖使能控制寄存器	0x0000_0000
PC_DBEN	GPIO_BA+0x094	R/W	PC消抖使能控制寄存器	0x0000_0000
PD_DBEN	GPIO_BA+0x0D4	R/W	PD消抖使能控制寄存器	0x0000_0000
PE_DBEN	GPIO_BA+0x114	R/W	PE消抖使能控制寄存器	0x0000_0000
PF_DBEN	GPIO_BA+0x154	R/W	PF消抖使能控制寄存器	0x0000_0000
PG_DBEN	GPIO_BA+0x194	R/W	PG消抖使能控制寄存器	0x0000_0000
PH_DBEN	GPIO_BA+0x1D4	R/W	PH消抖使能控制寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
DBEN							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
DBEN							

位	描述	
[31:16]	**Reserved**	保留。
[n] n=0,1..15	**DBEN[n]**	**端口A-H Pin[n] 输入信号去抖动使能位** DBEN[n]用于使能相应位的消抖功能. 如果输入信号脉冲宽度不能被两个连续的消抖采样周期所采样，则被视为信号抖动，从而不触发中断. 消抖时钟源由DBCLKSRC (GPIO_DBCTL [4])控制, 一个消抖周期长度由DBCLKSEL (GPIO_DBCTL [3:0])控制。 0 = 禁用Px.n消抖功能。 1 = 使能Px.n消抖功能。 消抖功能对于边沿触发中断有效, 对于电平触发中断模式，消抖功能使能位不起作用。 **注:** 端口A/B/E/G：最大 n=15。 端口C/D：最大 n=14。 端口F/H：最大 n=11。

端口A-H中断模式控制(Px_INTTYPE)

寄存器	偏移	R/W	描述	复位值
PA_INTTYPE	GPIO_BA+0x018	R/W	PA中断触发模式控制	0x0000_0000
PB_INTTYPE	GPIO_BA+0x058	R/W	PB中断触发模式控制	0x0000_0000
PC_INTTYPE	GPIO_BA+0x098	R/W	PC中断触发模式控制	0x0000_0000
PD_INTTYPE	GPIO_BA+0x0D8	R/W	PD中断触发模式控制	0x0000_0000
PE_INTTYPE	GPIO_BA+0x118	R/W	PE中断触发模式控制	0x0000_0000
PF_INTTYPE	GPIO_BA+0x158	R/W	PF中断触发模式控制	0x0000_0000
PG_INTTYPE	GPIO_BA+0x198	R/W	PG中断触发模式控制	0x0000_0000
PH_INTTYPE	GPIO_BA+0x1D8	R/W	PH中断触发模式控制	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
TYPE							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
TYPE							

位	描述	
[31:16]	**Reserved**	保留。
[n] n=0,1..15	**TYPE[n]**	**端口A-H Pin[n] 边沿或电平检测中断触发模式控制** TYPE (Px_INTTYPE[n]) 位用于控制电平触发或边沿触发中断. 若为边沿触发中断，触发源可由去抖动控制，如果是电平触发中断，输入源由一个HCLK时钟采样并产生中断。 0 = 边沿触发中断。 1 = 电平触发中断。 如果设置管脚为电平触发模式，则在寄存器RHIEN (Px_INTEN[n+16])/FLIEN (Px_INTEN[n])中，只能设置一个电平(高电平或者低电平)；若设置了两个电平都触发中断，则设置被忽略，不会产生中断。 去抖动功能对于边沿触发中断有效，对于电平触发中断无效。 **注:** 端口A/B/E/G：最大n=15。 端口C/D：最大n=14。 端口F/H：最大n=11。

端口 A-H中断使能控制寄存器(Px_INTEN)

寄存器	偏移	R/W	描述	复位值
PA_INTEN	GPIO_BA+0x01C	R/W	PA中断使能控制寄存器	0x0000_0000
PB_INTEN	GPIO_BA+0x05C	R/W	PB中断使能控制寄存器	0x0000_0000
PC_INTEN	GPIO_BA+0x09C	R/W	PC中断使能控制寄存器	0x0000_0000
PD_INTEN	GPIO_BA+0x0DC	R/W	PD中断使能控制寄存器	0x0000_0000
PE_INTEN	GPIO_BA+0x11C	R/W	PE中断使能控制寄存器	0x0000_0000
PF_INTEN	GPIO_BA+0x15C	R/W	PF中断使能控制寄存器	0x0000_0000
PG_INTEN	GPIO_BA+0x19C	R/W	PG中断使能控制寄存器	0x0000_0000
PH_INTEN	GPIO_BA+0x1DC	R/W	PH中断使能控制寄存器	0x0000_0000

31	30	29	28	27	26	25	24
RHIEN							
23	22	21	20	19	18	17	16
RHIEN							
15	14	13	12	11	10	9	8
FLIEN							
7	6	5	4	3	2	1	0
FLIEN							

位	描述	
[n+16] n=0,1..15	**RHIEN[n]**	**端口A-H Pin[n] 输入上升沿或输入高电平中断使能位** RHIEN (Px_INTEN[n+16])用于使能相应GPIO Px.n输入的中断. 置‘1’也使能了管脚唤醒功能 当设置RHIEN (Px_INTEN[n+16])位为‘1’: 如果中断是电平触发模式(TYPE (Px_INTTYPE[n]) = 1)，输入Px.n的状态为高电平时，产生中断。 如果中断是边沿触发模式(TYPE (Px_INTTYPE[n]) =0)，输入Px.n的状态由低电平到高电平变化时，产生中断。 1 = 使能Px.n高电平或由低电平到高电平变化的中断。 0 = 禁用Px.n高电平或由低电平到高电平变化的中断。 **注:** 端口A/B/E/G：最大 n=15。 端口C/D：最大 n=14。 端口F/H：最大 n=11。
[n]	**FLIEN[n]**	**使能端口A-H Pin[n] 输入下降沿或输入低电平的中断**

n=0,1..15		FLIEN (Px_INTEN[n])用于使能相应GPIO Px.n]输入的中断。置 '1' 也使能了管脚唤醒功能 当设置FLIEN (Px_INTEN[n])位为 '1': 如果中断是电平触发模式(TYPE (Px_INTTYPE[n]) = 1)，输入Px.n的状态为低电平时，产生中断。 如果中断是边沿触发模式(TYPE (Px_INTTYPE[n]) = 0)，输入Px.n的状态由高电平到低电平变化时，产生中断。 0 = 禁用Px.n低电平或由高电平到低电平变化的中断。 1 = 使能Px.n低电平或由高电平到低电平变化的中断。 **注:** 端口A/B/E/G：最大n=15。 端口C/D：最大n=14。 端口F/H：最大n=11。

端口 A-H中断源标志(Px_INTSRC)

Register	Offset	R/W	Description	Reset Value
PA_INTSRC	GPIO_BA+0x020	R/W	PA中断源标志	0x0000_XXXX
PB_INTSRC	GPIO_BA+0x060	R/W	PB中断源标志	0x0000_XXXX
PC_INTSRC	GPIO_BA+0x0A0	R/W	PC中断源标志	0x0000_XXXX
PD_INTSRC	GPIO_BA+0x0E0	R/W	PD中断源标志	0x0000_XXXX
PE_INTSRC	GPIO_BA+0x120	R/W	PE中断源标志	0x0000_XXXX
PF_INTSRC	GPIO_BA+0x160	R/W	PF中断源标志	0x0000_XXXX
PG_INTSRC	GPIO_BA+0x1A0	R/W	PG中断源标志	0x0000_XXXX
PH_INTSRC	GPIO_BA+0x1E0	R/W	PH中断源标志	0x0000_XXXX

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
INTSRC							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
INTSRC							

位	描述	
[31:16]	**Reserved**	保留。
[n] n=0,1..15	**INTSRC[n]**	**端口A-H Pin[n]中断触发源标志** 写： 0= 无动作。 1= 清相应的挂起中断标志。 读： 0 = Px.n没有中断。 1 = Px.n产生中断。 **注:** 端口A/B/E/G：最大 n=15。 端口C/D：最大 n=14。 端口F/H：最大n=11。

端口 A-H输入施密特触发使能寄存器(Px_SMTEN)

寄存器	偏移	R/W	描述	复位值
PA_SMTEN	GPIO_BA+0x024	R/W	PA输入施密特触发使能寄存器	0x0000_0000
PB_SMTEN	GPIO_BA+0x064	R/W	PB输入施密特触发使能寄存器	0x0000_0000
PC_SMTEN	GPIO_BA+0x0A4	R/W	PC输入施密特触发使能寄存器	0x0000_0000
PD_SMTEN	GPIO_BA+0x0E4	R/W	PD输入施密特触发使能寄存器	0x0000_0000
PE_SMTEN	GPIO_BA+0x124	R/W	PE输入施密特触发使能寄存器	0x0000_0000
PF_SMTEN	GPIO_BA+0x164	R/W	P输入施密特触发使能寄存器	0x0000_0000
PG_SMTEN	GPIO_BA+0x1A4	R/W	PG输入施密特触发使能寄存器	0x0000_0000
PH_SMTEN	GPIO_BA+0x1E4	R/W	PH输入施密特触发使能寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
SMTEN							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
SMTEN							

位	描述	
[31:16]	**Reserved**	保留。
[n] n=0,1..15	**SMTEN[n]**	**端口 A-H Pin[n] 施密特触发输入使能位** 0 = 禁用 Px.n 施密特触发输入功能。 1 = 使能Px.n 施密特触发输入功能。 **注:** 端口A/B/E/G：最大n=15。 端口C/D：最大n=14。 端口F/H：最大n=11。

端口 A-H高翻转速率控制寄存器(Px_SLEWCTL)

寄存器	偏移	R/W	描述	复位值
PA_SLEWCTL	GPIO_BA+0x028	R/W	PA高翻转速率控制寄存器	0x0000_0000
PB_SLEWCTL	GPIO_BA+0x068	R/W	PB高翻转速率控制寄存器	0x0000_0000
PC_SLEWCTL	GPIO_BA+0x0A8	R/W	PC高翻转速率控制寄存器	0x0000_0000
PD_SLEWCTL	GPIO_BA+0x0E8	R/W	PD高翻转速率控制寄存器	0x0000_0000
PE_SLEWCTL	GPIO_BA+0x128	R/W	PE高翻转速率控制寄存器	0x0000_0000
PF_SLEWCTL	GPIO_BA+0x168	R/W	PF高翻转速率控制寄存器	0x0000_0000
PG_SLEWCTL	GPIO_BA+0x1A8	R/W	PG高翻转速率控制寄存器	0x0000_0000
PH_SLEWCTL	GPIO_BA+0x1E8	R/W	PH高翻转速率控制寄存器	0x0000_0000

31	30	29	28	27	26	25	24
HSREN15		HSREN14		HSREN13		HSREN12	
23	22	21	20	19	18	17	16
HSREN11		HSREN10		HSREN9		HSREN8	
15	14	13	12	11	10	9	8
HSREN07		HSREN6		HSREN5		HSREN4	
7	6	5	4	3	2	1	0
HSREN03		HSREN2		HSREN1		HSREN0	

位	描述	
[2n+1:2n] n=0,1..15	HSRENn	**端口A-H Pin[n] 高翻转速率控制** 00 = Px.n 输出是正常翻转速度模式 (2.7V 时最大 40 MHz)。 01 = Px.n 输出是高翻转速度模式 (2.7V 时最大 80 MHz)。 10 = Px.n 输出是快速翻转速度模式 (2.7V 时最大 100 MHz)。 11 = 保留。 **注:** 端口A/B/E/G：最大n=15。 端口C/D：最大n=14。 端口F/H：最大n=11。

端口 A-H上拉和下拉选择寄存器(Px_PUSEL)

寄存器	偏移	R/W	描述	复位值
PA_PUSEL	GPIO_BA+0x030	R/W	PA上拉和下拉选择寄存器	0x0000_0000
PB_PUSEL	GPIO_BA+0x070	R/W	PB上拉和下拉选择寄存器	0x0000_0000
PC_PUSEL	GPIO_BA+0x0B0	R/W	PC上拉和下拉选择寄存器	0x0000_0000
PD_PUSEL	GPIO_BA+0x0F0	R/W	PD上拉和下拉选择寄存器	0x0000_0000
PE_PUSEL	GPIO_BA+0x130	R/W	PE上拉和下拉选择寄存器	0x0000_0000
PF_PUSEL	GPIO_BA+0x170	R/W	PF上拉和下拉选择寄存器	0x0000_0000
PG_PUSEL	GPIO_BA+0x1B0	R/W	PG上拉和下拉选择寄存器	0x0000_0000
PH_PUSEL	GPIO_BA+0x1F0	R/W	PH上拉和下拉选择寄存器	0x0000_0000

31	30	29	28	27	26	25	24
PUSEL15		PUSEL14		PUSEL13		PUSEL12	
23	22	21	20	19	18	17	16
PUSEL11		PUSEL10		PUSEL9		PUSEL8	
15	14	13	12	11	10	9	8
PUSEL7		PUSEL6		PUSEL5		PUSEL4	
7	6	5	4	3	2	1	0
PUSEL3		PUSEL2		PUSEL1		PUSEL0	

位	描述	
[2n+1:2n] n=0,1..15	PUSELn	**端口 A-H Pin[n] 上拉和下拉使能寄存器** 决定Px.n 管脚每一个I/O上拉/下拉是否使能。 00 = Px.n 上拉和下拉禁止。 01 = Px.n 上拉使能。 10 = Px.n 下拉使能。 **注1:** 基本上，上拉控制和下拉控制有如下限制。 仅当MODEn设置成三态或开漏模式时，单独上拉控制才有效。 仅当MODEn设置成三态模式。单独下拉控制才有效。 在三态模式，上拉和下拉都设置成1，保持I/O在三态模式。 **注2:** 端口A/B/E/G：最大n=15。 端口C/D：最大n=14。 端口F/H：最大n=11。

断消抖控制寄存器(GPIO_DBCTL)

寄存器	偏移	R/W	描述	复位值
GPIO_DBCTL	GPIO_BA+0x440	R/W	中断消抖控制寄存器	0x0000_0020

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved		ICLKON	DBCLKSRC	DBCLKSEL			

位	描述	
[31:6]	**Reserved**	保留。
[5]	**ICLKON**	**中断时钟开模式** 0 = 边沿侦测电路仅在 I/O管脚对应的RHIEN (Px_INTEN[n+16])/FLIEN (Px_INTEN[n])位置1有效。 1 = 复位之后所有 I/O边沿侦测电路使能。 **注1：**如果没有相关的特定应用，建议关掉时钟以减少耗电。 **注2:** 端口A/B/E/G：最大n=15。 端口C/D：最大n=14。 端口F/H：最大n=11。
[4]	**DBCLKSRC**	**消抖计数器时钟源选择** 0 = 去抖动计数器时钟源为 HCLK。 1 = 去抖动计数器时钟源为内部 10 KHz (LIRC)时钟。

位	描述	
[3:0]	**DBCLKSEL**	**消抖采样周期选择** 0000 = 采样中断信号，每1 个时钟一次。 0001 = 采样中断信号，每2 个时钟一次。 0010 = 采样中断信号，每4 个时钟一次。 0011 = 采样中断信号，每8 个时钟一次。 0100 = 采样中断信号，每16 个时钟一次。 0101 = 采样中断信号，每32 个时钟一次。 0110 = 采样中断信号，每64 个时钟一次。 0111 = 采样中断信号，每128 个时钟一次。 1000 = 采样中断信号，每256 个时钟一次。 1001 = 采样中断信号，每2*256 个时钟一次。 1010 = 采样中断信号，每4*256 个时钟一次。 1011 = 采样中断信号，每8*256 个时钟一次。 1100 = 采样中断信号，每16*256 个时钟一次。 1101 = 采样中断信号，每32*256 个时钟一次。 1110 = 采样中断信号，每64*256 个时钟一次。 1111 = 采样中断信号，每128*256 个时钟一次。

GPIO Px.n管脚数据/输入输出寄存器(Pxn_PDIO)

寄存器	偏移	R/W	描述	复位值
PAn_PDIO n=0,1..15	GPIO_BA+0x800+(0x04 * n)	R/W	GPIO PA.n管脚数据输入/输出寄存器	0x0000_000X
PBn_PDIO n=0,1..15	GPIO_BA+0x840+(0x04 * n)	R/W	GPIO PB.n管脚数据输入/输出寄存器	0x0000_000X
PCn_PDIO n=0,1..15	GPIO_BA+0x880+(0x04 * n)	R/W	GPIO PC.n管脚数据输入/输出寄存器	0x0000_000X
PDn_PDIO n=0,1..15	GPIO_BA+0x8C0+(0x04 * n)	R/W	GPIO PD.n管脚数据输入/输出寄存器	0x0000_000X
PEn_PDIO n=0,1..15	GPIO_BA+0x900+(0x04 * n)	R/W	GPIO PE.n管脚数据输入/输出寄存器	0x0000_000X
PFn_PDIO n=0,1..15	GPIO_BA+0x940+(0x04 * n)	R/W	GPIO PF.n管脚数据输入/输出寄存器	0x0000_000X
PGn_PDIO n=0,1..12	GPIO_BA+0x980+(0x04 * n)	R/W	GPIO PG.n管脚数据输入/输出寄存器	0x0000_000X
PHn_PDIO n=0,1..12	GPIO_BA+0x9C0+(0x04 * n)	R/W	GPIO PH.n管脚数据输入/输出寄存器	0x0000_000X

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved							PDIO

位	描述	
[31:1]	Reserved	保留。
[0]	PDIO	**GPIO Px.n管脚数据输入/输出控制** 写该位可以控制一个GPIO管脚的输出值 0 = 设置相应GPIO管脚为低。 1 = 设置相应GPIO管脚为高。 读该寄存器得到GPIO管脚状态。 例如：写PA0_PDIO即把值写到DOUT (Px_DOUT[0])位上，读PA0_PDIO即读取PIN (PA_PIN[0])的值。

		注1：写操作不受DATMSK (Px_DATMSK[n])影响。 **注2:** 端口A/B/E/G：最大n=15。 端口C/D：最大n=14。 端口F/H：最大n=11。

6.7 PDMA 控制器 (PDMA)

6.7.1 概述

直接存储器存取(PDMA)用于高速数据传输。PDMA控制器可以从一个地址到另一个地址传输数据，无需CPU介入。这样做的好处是减少CPU的工作量，把节省下的CPU资源做其他应用。PDMA控制器包含16个通道，每个通道支持内存和外设之间的数据传输和内存与内存之间的数据传输。

6.7.2 特征

- 支持16个可独立配置的通道
- 支持2种优先级选择：固定优先级(fixed priority)或轮循调度优先级(round-robin priority))
- 传输数据宽度支持8位，16位，32位
- 支持源地址与目的地址方向递增或固定，支持字节，半字，字传输。
- 支持软件，SPI, UART, DAC, ADC 和 PWM请求
- 支持Scatter-Gather模式，通过描述符链表执行灵活的数据传输。
- 支持单一和批量传输类型
- 通道0和通道1支持超时功能
- 通道0到通道5支持跨步功能
- 增强跨步功能以用于图像处理(M48xGC/M48xE8)

6.7.3 框图

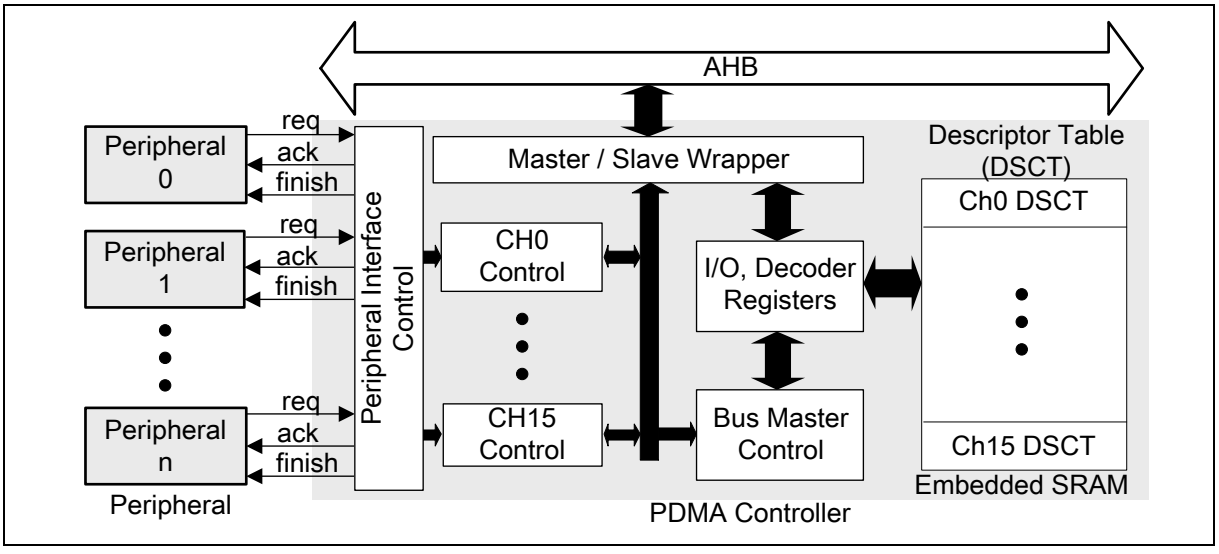


图 6.7-1 PDMA 控制器框图

6.7.4 基本配置

- 时钟源配置

- 在PDMACKEN (CLK_AHBCLK [1])中，使能 PDMA 控制器时钟。

- 复位配置
  - 在PDMARST (SYS_IPRST0[2])中，复位PDMA控制器。

6.7.5 功能描述

直接存储器存取(PDMA)控制器可以从一个地址到另一个地址传输数据，无需CPU介入。PDMA有16个独立通道，因为每次只有一个通道工作，因此，PDMA 控制器支持两种通道优先级：固定优先级和调度优先级，PDMA控制器通道执行的优先级是从高到低的。PDMA 控制器支持两种运行模式：基本模式和Scatter-gather模式。基本模式用于按照一个传输描述符表格传输数据。Scatter-gather模式对于每个PDMA都有多个传输描述符表格，所以PDMA控制器通过这些表格，实现灵活的数据传输，传输描述符表数据结构包含了传送信息，例如：传输源地址，传输目的地址，传输计数，批量传输数据大小，传输类型和操作模式。图6.7-2是描述符表(DSCT)数据结构图。

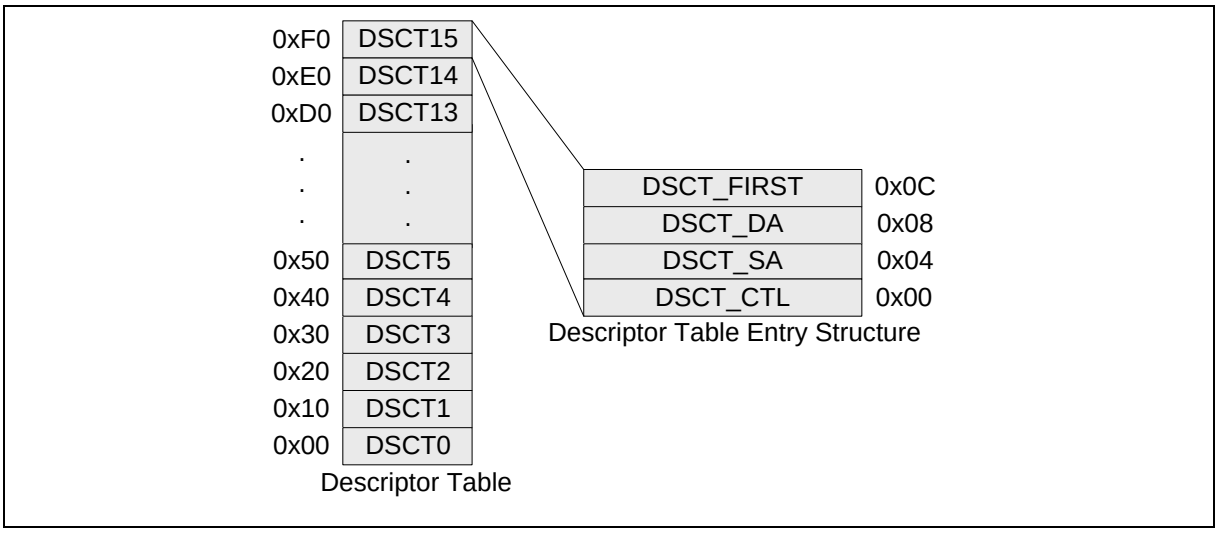


图 6.7-2 描述符表入口结构

PDMA 控制器也支持单一和批量数据的传输类型，请求源可以是软件请求或外设请求。内存之间的数据传输是使用软件请求。单一传输的意思是软件或外设准备好传输一个数据(每个数据需要一次请求)，批量传输的意思是软件或外设将传输多个数据(多个数据仅需一次请求)。

6.7.5.1 通道优先级

PDMA控制器支持两级通道优先等级，包括固定优先级和轮循调度优先级。固定优先级比轮循调度优先级的优先级别更高。如果多路通道设定为固定优先级或调度优先级，高通道的优先级别也高。优先级原则如表6.7-1。

PDMA_PRISET	通道号	优先级设定	优先级递减顺序
1	15	通道15, 固定优先级	最高
1	14	通道14, 固定优先级	---
---	---	---	---
1	0	通道0, 固定优先级	---

0	15	通道15, 轮循调度优先级	---
0	14	通道14, 轮循调度优先级	---
---	---	---	---
0	0	通道0, 轮循调度优先级	最低

表 6.7-1 通道优先级表

6.7.5.2 PDMA操作模式

PDMA支持两种操作模式，包括: 基本模式和Scatter-Gather模式。

基本模式

基本模式用于执行一个描述符表格的传输模式。该模式用于内存与内存之间的数据传输或接口与内存之间的数据传输。PDMA 控制器操作模式可以通过寄存器OPMODE (PDMA_DSCTn_CTL[1:0], n 代表 PDMA 通道数) 设定, 默认设置是在空闲状态(OPMODE (PDMA_DSCTn_CTL[1:0]) = 0x0)，推荐用户设定描述符表为空闲状态。如果操作模式不是空闲状态，用户重新配置通道设定可能会引起操作错误。

用户必须填写传输计数寄存器TXCNT (PDMA_DSCTn_CTL31:16])和传输宽度选择寄存器TXWIDTH (PDMA_DSCTn_CTL[13:12]), 目的地址递增大小寄存器DAINC (PDMA_DSCTn_CTL[11:10])，源地址递增大小寄存器SAINC (PDMA_DSCTn_CTL[9:8]), 批量大小寄存器BURSIZE (PDMA_DSCTn_CTL[6:4])和传输类型寄存器TXTYPE(PDMA_DSCTn_CTL[2])，然后PDMA控制器将在接收到请求信号后执行传输操作。如果相应的PDMA 中断位INTENn (PDMA_INTEN[15:0])使能，传输完成后将给CPU产生一个中断，操作模式将被更新为空闲模式，如图6.7-3。如果软件配置操作模式为空闲状态，PDMA控制器不会执行任何传输，并清除这个操作请求。如果相应的PDMA中断位被使能，完成这个任务后也会给CPU产生中断。

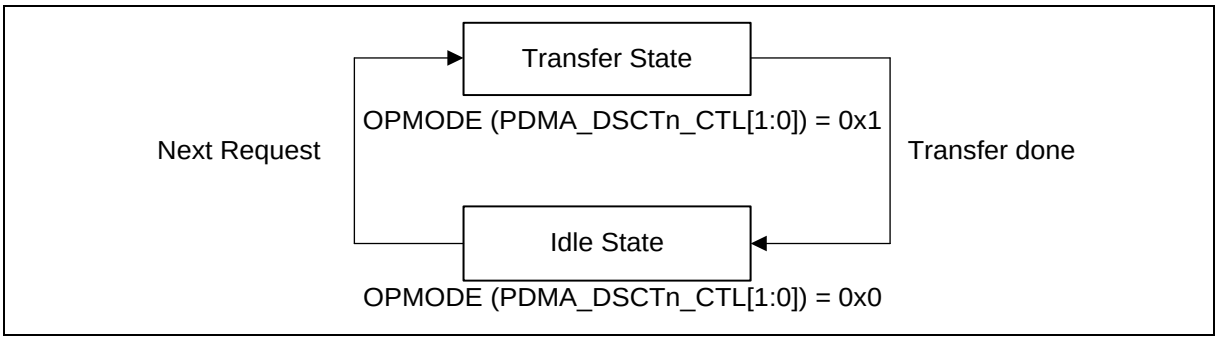


图 6.7-3 基本模式的有限状态机

Scatter-Gather 模式

Scatter-Gather模式是一个复杂模式，通过如图6.7-4描述符表格链接单结构描述符链表设定，可以实现灵活的数据传输。通过该模式，用户可以实现外设的回环访问。用于多路PDMA任务或用于系统内存的不同位置(非相邻位置)的数据搬移。Scatter-gather模式仅需要一个请求来完成所有表格入口任务，直到最后一个任务。这也意味着Scatter-Gather模式仅用于在内存与内存之间传输数据，没有握手信号。

在Scatter-Gather模式，这个表用于跳转到下一个表的入口。第一个任务不会做数据搬移操作。如果相应的PDMA 中断位使能，且TBINTDIS (PDMA_DSCTn_CTL[7])=0，完成每个任务后，将给CPU产生一个中断，(任务完成，TBINTDIS位为“0”，会置位相应的寄存器TDIFn (PDMA_TDSTS[15:0])标志，如果TBINTDIS位为“1”，禁用TDIFn)。

如果触发了通道15，在Scatter-Gather(OPMODE (OPMODE (PDMA_DSCTn_CTL[1:0]) = 0x2)模式，

硬件将把寄存器PDMA_DSCTn_FIRST (链接地址)和PDMA_SCATBA (基地址)相加来获取真正的PDMA任务地址信息。例如，基地址是0x2000_0000(PDMA_SCATBA寄存器中仅MSB 16位有效)，目前链接地址是0x0000_0100 (在寄存器PDMA_DSCTn_ FIRST中，仅LSB 16位(不包括[1:0])有效)，那么，下一个DSCT的起始地址是0x2000_0100。

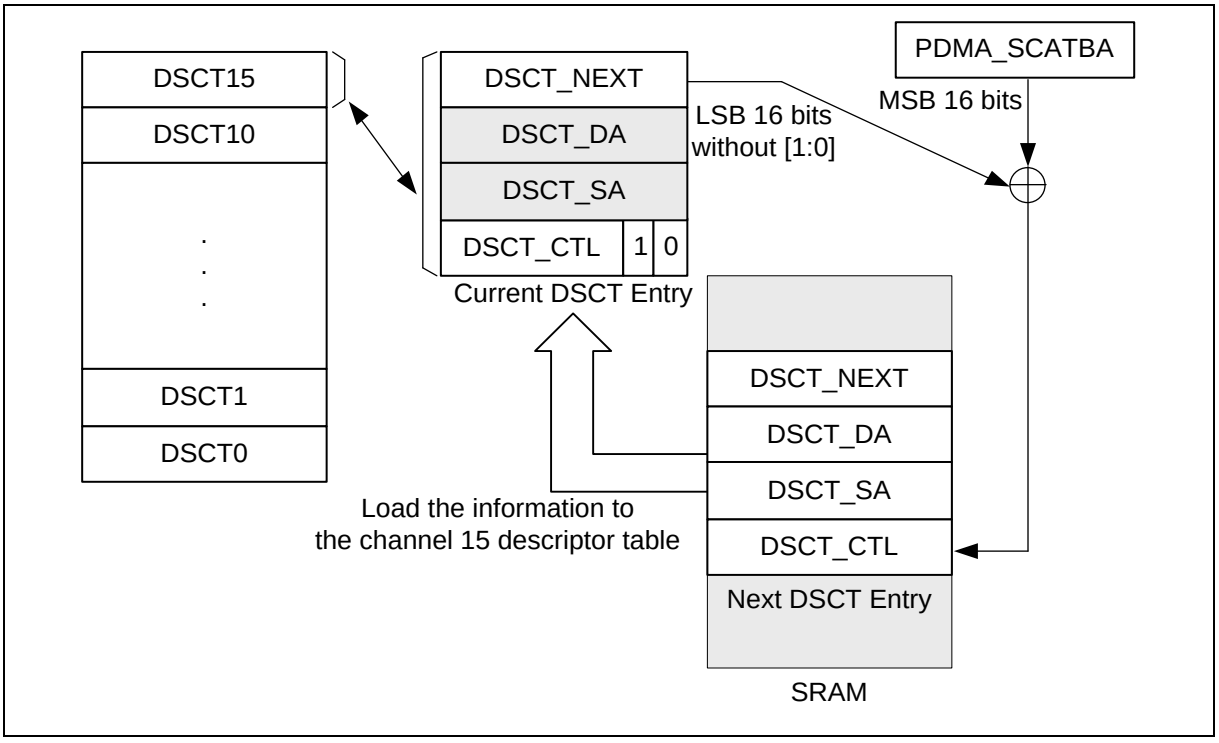


图 6.7-4 描述符表格链接单结构

上述链接列表操作是Scatter-Gather 模式的DSCT状态。如图6.7-5所示，当加载信息结束后，通过该信息将自动进入搬移数据状态并开始传输。然而，如果下一个PDMA信息也是Scatter-Gather模式，当前任务结束后，硬件将抓下一个PDMA块信息。Scatter-Gather模式只有在PDMA控制器操作模式切换到基本模式并完成最后一次传输或者直接切换到空闲状态后才会结束。

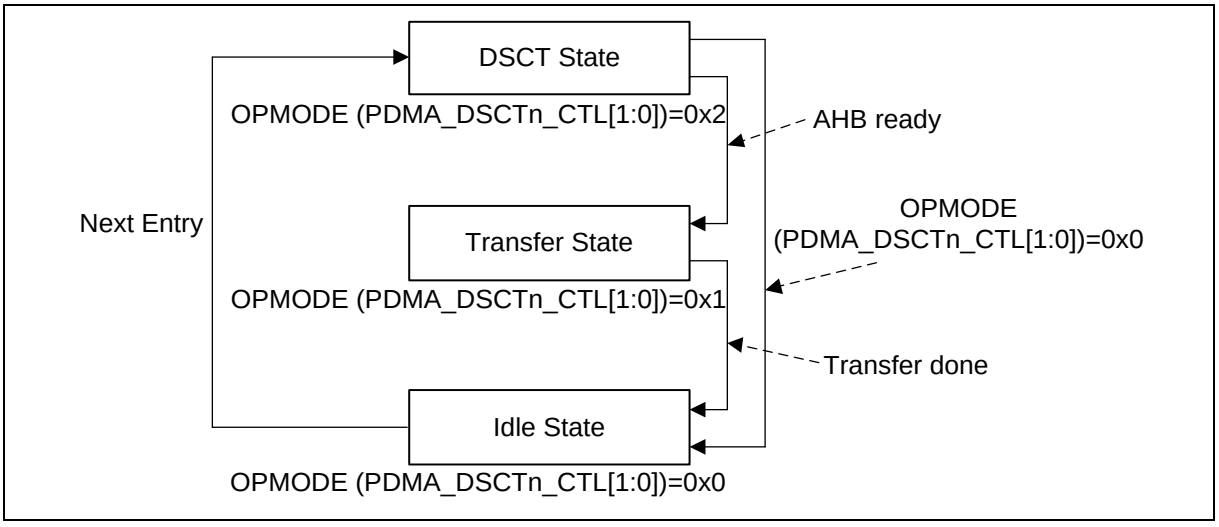


图 6.7-5Scatter-Gather 模式的有限状态机

6.7.5.3 传输类型

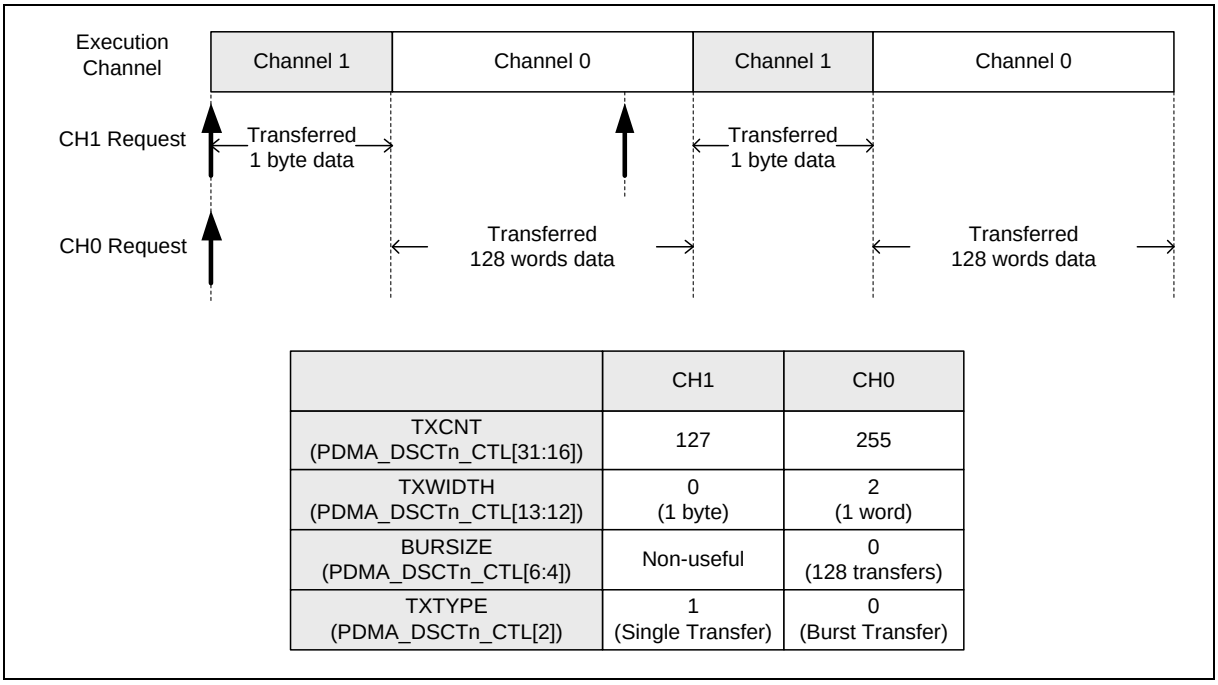
PDMA 控制器支持两种传输模式：单一传输和批量传输模式，通过寄存器TXTYPE (PDMA_DSCTn_CTL[2])设定。

当PDMA控制器运行在单一传输模式，每搬移一个数据需要一次请求，当搬移一次数据，寄存器TXCNT (PDMA_DSCTn_CTL[31:16])会减1，直到寄存器TXCNT (PDMA_DSCTn_CTL[31:16])为0，搬移才会完成。在该模式，寄存器BURSIZE (PDMA_DSCTn_CTL[6:4])不用于控制搬移数据大小。BURSIZE (PDMA_DSCTn_CTL[6:4])数值是固定的。

在批量搬移模式，PDMA控制器搬移TXCNT (PDMA_DSCTn_CTL[31:16])个数据，仅需一次请求。当搬移BURSIZE (PDMA_DSCTn_CTL[6:4]) 个数据后，寄存器TXCNT (PDMA_DSCTn_CTL[31:16])中的BURSIZE数目会减去BURSIZE。直到TXCNT (PDMA_DSCTn_CTL[31:16])递减为0，搬移数据才完成。注意批量传输模式仅用于PDMA控制器在内存与内存之间做批量数据的搬移，对内存到外设或外设到内存传输，用户需要用单次请求模式请注意，PDMA在闪存和存储器之间传输数据应该在MCU进入空闲或掉电模式之前完成，以防止访问错误的数据

图6.7-6是基本传输模式的单一传输和批量传输模式举例。在这个例子，通道1使用单一传输模式，TXCNT (PDMA_DSCTn_CTL[31:16]) = 127 。通道0使用批量传输模式，BURSIZE (PDMA_DSCTn_CTL[6:4]) = 128 并且TXCNT (PDMA_DSCTn_CTL[31:16]) = 255。操作顺序如下:

1. 通道0与通道1同时接收到触发信号。
2. 默认通道1的优先级高于通道0；PDMA控制器将先加载通道1的描述符表并执行。但通道1是单一传输模式，所以PDMA控制器仅传输一个数据。
3. 然后，PDMA控制器切换到通道0并加载通道0的描述符表。通道0是批量传输模式，大小是128个。所以PDMA控制器将传输128个数据。
4. 当通道0搬运128个数据，通道1得到另一个请求信号，所以当通道0搬运完128个数据，PDMA控制器将切回通道1，来搬运通道1的下一个数据。
5. 当通道1搬运完数据， PDMA控制器切换到低优先级的通道0来继续搬运下一组128个数据，如果没有收到通道1的请求，PDMA控制器将继续搬运下一组128个数据。
6. 当通道0完成256次数据搬移，通道1完成128次数据搬移，PDMA将完成数据搬移。



	CH1	CH0
TXCNT (PDMA_DSCTn_CTL[31:16])	127	255
TXWIDTH (PDMA_DSCTn_CTL[13:12])	0 (1 byte)	2 (1 word)
BURSIZE (PDMA_DSCTn_CTL[6:4])	Non-useful	0 (128 transfers)
TXTYPE (PDMA_DSCTn_CTL[2])	1 (Single Transfer)	0 (Burst Transfer)

图 6.7-6 基本模式的单字传输和批量传输举例

6.7.5.4 通道超时

仅PDMA通道0和通道1支持超时功能。当传输通道使能，且选择到外设，对应通道超时TOUTENn (PDMA_TOUTEN [n], n=0,1)功能使能，然后当通道接收到来自外设的触发信号，通道对应的超时计数器开始从0向上计数。

超时计数器基于HCLK预分频的输出，这由对应通道的TOUTPSCn (PDMA_TOUTPSC [2+4n:4n], n=0,1)来设置。如果超时计数器从0开始向上计数到对应通道的TOCn (PDMA_TOC0_1 [16(n+1)-1):16n], n=0,1)的值，当对应的TOUTIENn (PDMA_TOUTIEN [n], n=0,1) 使能，PDMA控制器会产生中断信号。当超时发送，对应通道的REQTOFn (PDMA_INTSTS [n+8], n=0,1)会被置位，表示通道超时发生。

当计数器计数到TOCn (PDMA_TOC0_1 [16(n+1)-1:16n], n=0,1)，接收到触发信号超时功能禁止或芯片进入掉电模式。超时计数器复位到0。

图6.7-7是超时计数器工作的例子，操作顺序如下描述:

1. 当通过设置TOUTEN0(PDMA_TOUTEN[0])为1使能超时功能，通道0超时计数器还没有开始计数。
2. 当接收到第一个外设请求。超时计数器从0开始计数到TOC0(PDMA_TOC0_1[15:0])的值。
3. 超时计数器由接收到第二个外设请求复位。
4. 当超时计数器数到5，通道0请求超时标志(REQTOF0(PDMA_INTSTS[8]))被置高。超时计数器保持从0到5计数，用户可以清REQTOF0标志，然后通过查询REQTOF0标志检查下次超时的发生。
5. 当超时功能禁止超时计数器复位为0。

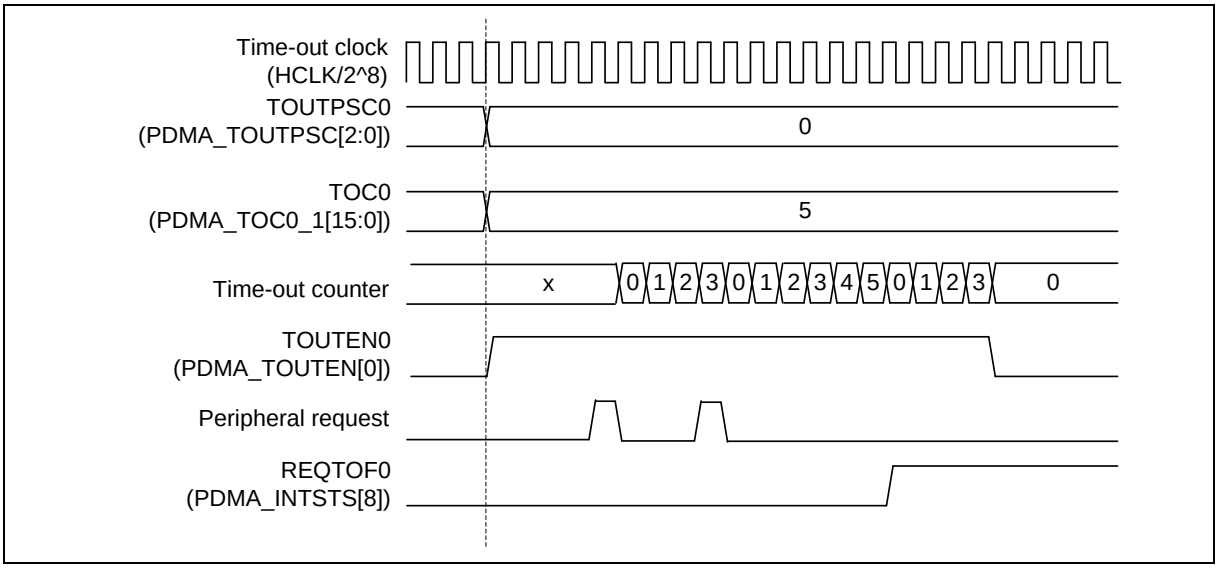


图 6.7-7PDMA 通道 0 超时计数器操作例子

6.7.5.5 跨步功能

PDMA支持通道0到通道5这6个通道的跨步功能。跨步功能可以从一个地址传输数据到另一个地址，且支持带跨步的块传输。当操作在跨步功能，传输地址可以固定或连续递增

设置STRIDE_EN (PDMA_DSCTn_CTL[15])位使能跨步功能，然后写一个有效的源地址到PDMA_DSCTn_SA寄存器，一个源地址偏移计数到SASOL (PDMA_ASOCRn[15:0])寄存器，一个目的地址到PDMA_DSCTn_DA寄存器，一个目的地址偏移计数到DASOL (PDMA_ASOCRn[31:16])，写一个传输计数到TXCNT (PDMA_DSCTn_CTL)寄存器，一个跨步传输计数到STC (PDMA_STCn[15:0])。接下来触发SWREQn (PDMA_SWREQ[5:0])。DMA会继续传输直到TXCNT (PDMA_DSCTn_CTL)下数到0。图6.7-8表示在源内存和目的内存之间块传输的关系。跨步功能也支持外设到内存或内存到外设传输。

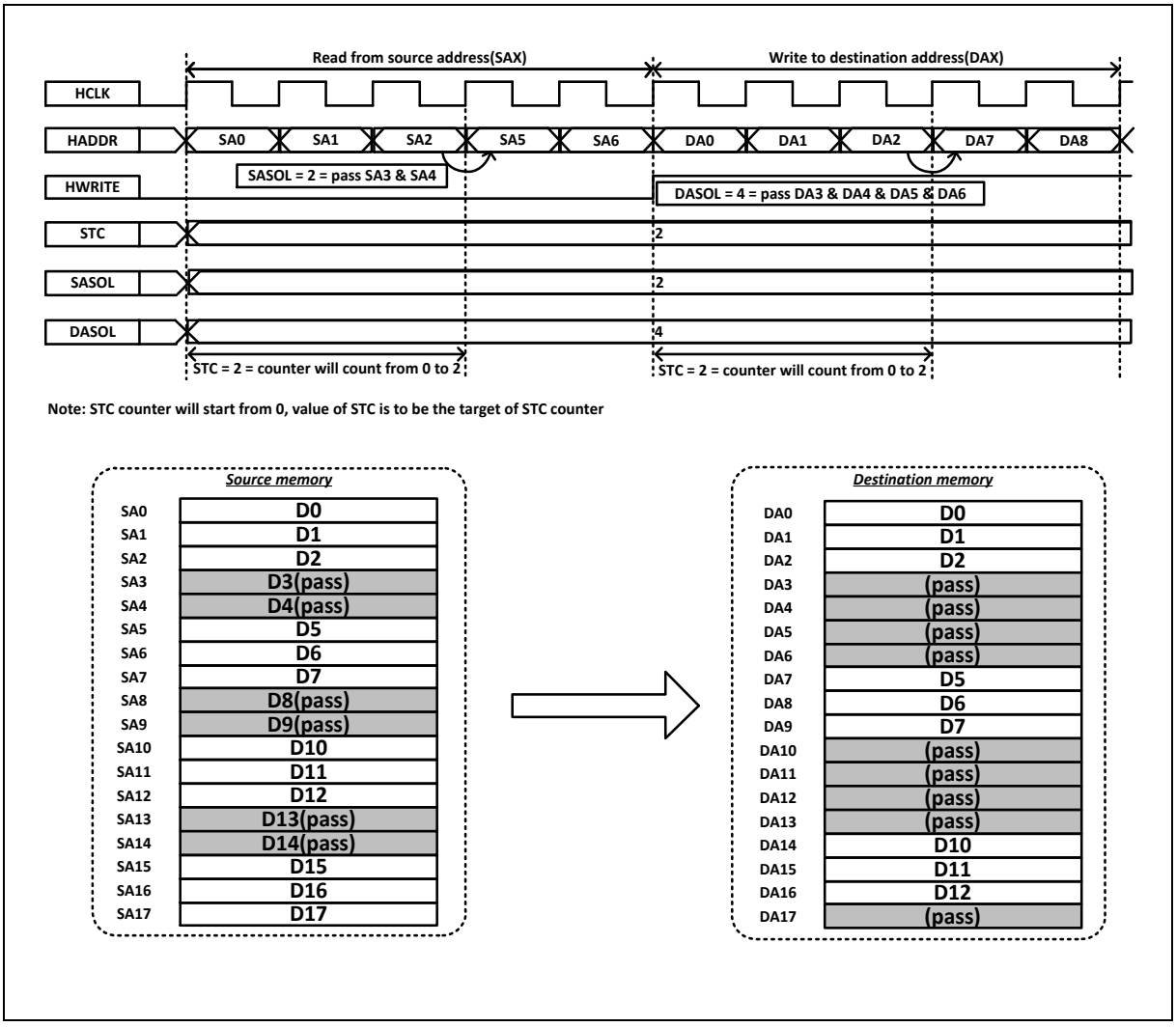


图 6.7-8 跨步功能块传输

6.7.5.6 *增加跨步功能*

为使跨步功能应用于DSP，图像处理和其它应用，通道0和通道1支持间隔计数和重复计数。间隔计数用于决定每个区块传输的间隔。SAICNT (PDMA_AICTLn[15:0])是间隔计数的源地址，DAICNT (PDMA_AICTLn[31:16])是间隔计数的目的地址。重复计数用于决定只有一个触发的区块传输的重复时间。如果RCNT (PDMA_RCNT[15:0])区块重复传输次数为0，传输就停止。下面展示两种传输的情况。第一种情况间隔计数SAICNT=3，DAICNT=9和重复计数count = 2。另一种情况是间隔计数SAICNT=3，DAICNT=6和重复计数count = 2。

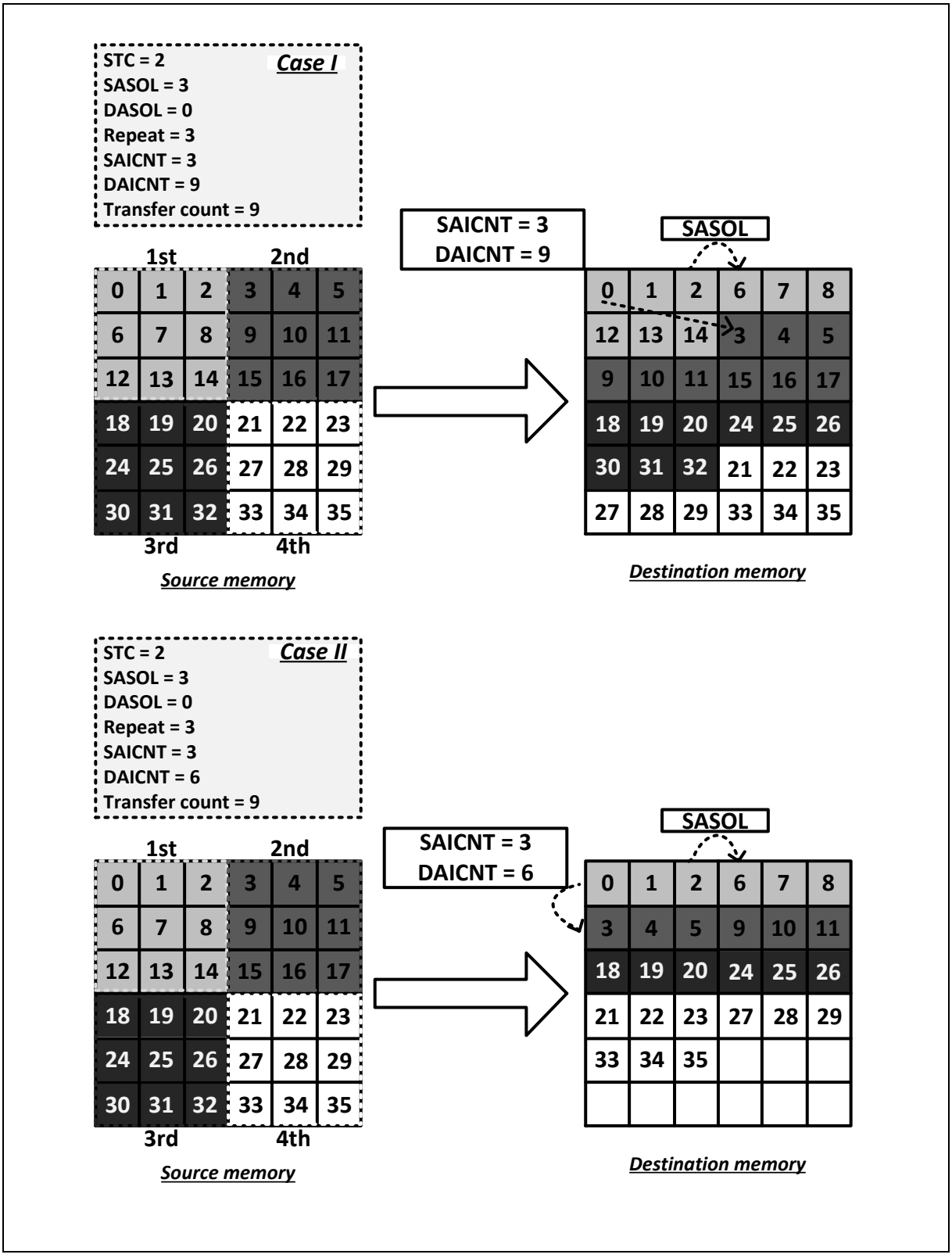


图 6.7-9 两种情况的区块传输

6.7.6 寄存器映射

R:只读, **W:**只写, **R/W:**读写

寄存器	偏移	R/W	描述	复位值
PDMA 基地址: **PDMA_BA = 0x4000_8000** **DSCT_CTL_BA = 0x4000_8000** **DSCT_SA_BA = 0x4000_8004** **DSCT_DA_BA = 0x4000_8008** **DSCT_NEXT_BA = 0x4000_800c** **CURSCAT_BA = 0x4000_8100**				
PDMA_DSCTn_CTL **n = 0,1..15**	DSCT_CTL_BA + 0x10 * n	R/W	PDMA通道n描述符表控制寄存器	0xXXXX_XXXX
PDMA_DSCTn_SA **n = 0,1..15**	DSCT_SA_BA + 0x10 * n	R/W	PDMA通道n源地址寄存器	0xXXXX_XXXX
PDMA_DSCTn_DA **n = 0,1..15**	DSCT_DA_BA + 0x10 * n	R/W	PDMA通道n目的地址寄存器	0xXXXX_XXXX
PDMA_DSCTn_NEXT **n = 0,1..15**	DSCT_NEXT_BA + 0x10 * n	R/W	PDMA通道n的第一个Scatter-Gather描述符表偏移地址	0xXXXX_XXXX
PDMA_CURSCATn **n = 0,1..15**	CURSCAT_BA + 0x004 * n	R	PDMA通道n的目前Scatter-Gather描述符表地址	0xXXXX_XXXX
PDMA_CHCTL	PDMA_BA + 0x400	R/W	PDMA通道控制寄存器	0x0000_0000
PDMA_PAUSE	PDMA_BA + 0x404	W	PDMA 传输停止控制寄存器	0x0000_0000
PDMA_SWREQ	PDMA_BA + 0x408	W	PDMA软件请求寄存器	0x0000_0000
PDMA_TRGSTS	PDMA_BA + 0x40C	R	PDMA 通道请求状态寄存器	0x0000_0000
PDMA_PRISET	PDMA_BA + 0x410	R/W	PDMA 固定优先级设定寄存器	0x0000_0000
PDMA_PRICLR	PDMA_BA + 0x414	W	PDMA固定优先级清除寄存器	0x0000_0000
PDMA_INTEN	PDMA_BA + 0x418	R/W	PDMA中断使能寄存器	0x0000_0000
PDMA_INTSTS	PDMA_BA + 0x41C	R/W	PDMA 中断状态寄存器	0x0000_0000
PDMA_ABTSTS	PDMA_BA + 0x420	R/W	PDMA 通道读/写目标终止标志寄存器	0x0000_0000
PDMA_TDSTS	PDMA_BA + 0x424	R/W	PDMA 通道传输完成标志寄存器	0x0000_0000
PDMA_ALIGN	PDMA_BA + 0x428	R/W	PDMA 传输对齐状态寄存器	0x0000_0000
PDMA_TACTSTS	PDMA_BA + 0x42C	R	PDMA 传输激活标志寄存器	0x0000_0000
PDMA_TOUTPSC	PDMA_BA + 0x430	R/W	PDMA 超时预分频器寄存器	0x0000_0000
PDMA_TOUTEN	PDMA_BA + 0x434	R/W	PDMA 超时使能寄存器	0x0000_0000
PDMA_TOUTIEN	PDMA_BA + 0x438	R/W	PDMA 超时中断使能寄存器	0x0000_0000

PDMA_SCATBA	PDMA_BA + 0x43C	R/W	PDMA Scatter-Gather描述符表基地址寄存器	0x2000_0000
PDMA_TOC0_1	PDMA_BA + 0x440	R/W	PDMA 超时计数器通道1和通道0寄存器	0xFFFF_FFFF
PDMA_CHRST	PDMA_BA + 0x460	R/W	PDMA 通道复位寄存器	0x0000_0000
PDMA_REQSEL0_3	PDMA_BA + 0x480	R/W	PDMA 请求源选择寄存器0	0x0000_0000
PDMA_REQSEL4_7	PDMA_BA + 0x484	R/W	PDMA请求源选择寄存器1	0x0000_0000
PDMA_REQSEL8_11	PDMA_BA + 0x488	R/W	PDMA请求源选择寄存器2	0x0000_0000
PDMA_REQSEL12_15	PDMA_BA + 0x48C	R/W	PDMA请求源选择寄存器3	0x0000_0000
PDMA_STC0	PDMA_BA + 0x500	R/W	PDMA 通道 0的跨步传输计数寄存器	0x0000_0000
PDMA_ASOCTL0	PDMA_BA + 0x504	R/W	PDMA 通道 0的地址跨步偏移量寄存器	0x0000_0000
PDMA_STC1	PDMA_BA + 0x508	R/W	PDMA 通道 1的跨步传输计数寄存器	0x0000_0000
PDMA_ASOCTL1	PDMA_BA + 0x50C	R/W	PDMA 通道 1的地址跨步偏移量寄存器	0x0000_0000
PDMA_STC2	PDMA_BA + 0x510	R/W	PDMA 通道 2的跨步传输计数寄存器	0x0000_0000
PDMA_ASOCTL2	PDMA_BA + 0x514	R/W	PDMA 通道 2的地址跨步偏移量寄存器	0x0000_0000
PDMA_STC3	PDMA_BA + 0x518	R/W	PDMA 通道 3的跨步传输计数寄存器	0x0000_0000
PDMA_ASOCTL3	PDMA_BA + 0x51C	R/W	PDMA 通道 3的地址跨步偏移量寄存器	0x0000_0000
PDMA_STC4	PDMA_BA + 0x520	R/W	PDMA 通道 4的跨步传输计数寄存器	0x0000_0000
PDMA_ASOCTL4	PDMA_BA + 0x524	R/W	PDMA 通道 4的地址跨步偏移量寄存器	0x0000_0000
PDMA_STC5	PDMA_BA + 0x528	R/W	PDMA 通道 5的跨步传输计数寄存器	0x0000_0000
PDMA_ASOCTL5	PDMA_BA + 0x52C	R/W	PDMA 通道 5的地址跨步偏移量寄存器	0x0000_0000
PDMA_AICTL0	PDMA_BA + 0x600	R/W	PDMA 通道0地址间隔控制寄存器	0x0000_0000
PDMA_RCNT0	PDMA_BA + 0x604	R/W	PDMA 通道0重复计数寄存器	0x0000_0000
PDMA_AICTL1	PDMA_BA + 0x608	R/W	PDMA 通道1地址间隔控制寄存器	0x0000_0000
PDMA_RCNT1	PDMA_BA + 0x60C	R/W	PDMA 通道1重复计数寄存器	0x0000_0000

6.7.7 寄存器描述

描述符表控制寄存器(PDMA_DSCTn_CTL)

寄存器	偏移	R/W	描述	复位值
PDMA_DSCTn_CTL	DSCT_CTL_BA + 0x10 * n	R/W	PDMA 通道 n的描述符表控制寄存器	0xXXXX_XXXX

31	30	29	28	27	26	25	24
TXCNT							
23	22	21	20	19	18	17	16
TXCNT							
15	14	13	12	11	10	9	8
STRIDEEN	Reserved	TXWIDTH		DAINC		SAINC	
7	6	5	4	3	2	1	0
TBINTDIS	BURSIZE			Reserved	TXTYPE	OPMODE	

位	描述	
[31:16]	**TXCNT**	**传输计数** TXCNT代表PDMA传输的请求数，实际的传输数是(TXCNT + 1);最大的传输计数是65536，依据TXWIDTH寄存器设定，每次传输数据单位可以是字节，半-字，或字。 **注：**当PDMA完成每次数据传输，该寄存器计数会立刻递减。
[15]	**STRIDEEN**	**跨步模式使能位** 0 = 跨步传输模式禁止。 1 = 跨步传输模式使能。
[14]	**Reserved**	保留。
[13:12]	**TXWIDTH**	**传输宽度选择** 该寄存器用于数据传输宽度设定。 00 = 每次操作数据传输宽度是一个字节(8位)。 01= 每次操作数据传输宽度是半-字(16位)。 10 = 每次操作数据传输宽度是字(32位)。 11 = 保留。 **注：**PDMA传输源地址(PDMA_DSCT_SA)和PDMA传输目的地址(PDMA_DSCT_DA)应根据TXWIDTH的设定对齐。
[11:10]	**DAINC**	**目的地址增加** 该寄存器用于设定目的地址的递增量大小。 11 = 无增量(固定地址)。 其他= 增量的大小依据寄存器TXWIDTH设定。
[9:8]	**SAINC**	**源地址增加** 该寄存器用于设定源地址递增量的大小。

位	描述	
		11 = 无增量(固定地址)。 其他 = 增量的大小依据寄存器TXWIDTH设定。
[7]	**TBINTDIS**	**禁用表中断** 该寄存器是用于是否使能表中断。当PDMA完成传输任务，如果TBINTDIS位使能，将不会产生中断。 0 = 表中断使能 1 = 禁用表中断 **注：** 该功能仅用于scatter-gather模式。
[6:4]	**BURSIZE**	**批量大小** 000 = 128 个数据传输。 001 = 64 个数据传输。 010 = 32 个数据传输。 011 = 16 个数据传输。 100 = 8 个数据传输。 101 = 4 个数据传输。 110 = 2 个数据传输。 111 = 1 个数据传输。 **注：** 这个寄存器只用于批量传输模式。
[3]	**Reserved**	保留。
[2]	**TXTYPE**	**传输类型** 0 = 批量传输类型。 1 = 单一传输类型。
[1:0]	**OPMODE**	**PDMA 工作模式选择** 00 = 空闲状态：通道停止或该表格完成，当PDMA完成通道表格任务，OPMODE会自动清零到空闲状态。 01 = 基本模式：描述符表格只有一个任务。当这个任务完成， PDMA_INTSTS[n]将被置位。 10 = Scatter-Gather模式：在该模式，用户必须把下一个描述符表格地址输入寄存器PDMA_DSCT_ FIRST，PDMA控制机将忽略这次任务，然后加载执行下一个任务。 11 = 保留。 **注:** 在填写传输任务到描述符表之前，用户必须要检查描述符表是否完成。

起始源地址寄存器(PDMA_DSCTn_SA)

寄存器	偏移	R/W	描述	复位值
PDMA_DSCTn_SA	DSCT_SA_BA + 0x10 * n	R/W	PDMA 通道n 源地址寄存器	0xXXXX_XXXX

31	30	29	28	27	26	25	24
SA							
23	22	21	20	19	18	17	16
SA							
15	14	13	12	11	10	9	8
SA							
7	6	5	4	3	2	1	0
SA							

位	描述	
[31:0]	SA	**PDMA 传输源地址寄存器** 该寄存器是32-位PDMA的源地址。

目的地址寄存器(PDMA_DSCTn_DA)

寄存器	偏移	R/W	描述	复位值
PDMA_DSCTn_DA	DSCT_DA_BA + 0x10 * n	R/W	PDMA 通道n 目的地址	0xXXXX_XXXX

31	30	29	28	27	26	25	24
DA							
23	22	21	20	19	18	17	16
DA							
15	14	13	12	11	10	9	8
DA							
7	6	5	4	3	2	1	0
DA							

位	描述	
[31:0]	DA	**PDMA传输目的地址寄存器** 该寄存器是32-位PDMA控制器的目的地址寄存器。

下一个Scatter-Gather描述表偏移地址(PDMA_DSCTn_NEXT)

寄存器	偏移	R/W	描述	复位值
PDMA_DSCTn_NEXT	DSCT_NEXT_BA + 0x10 * n	R/W	PDMA 通道n 下一个Scatter-Gather描述符表偏移地址	0xXXXX_XXXX

31	30	29	28	27	26	25	24
EXENEXT							
23	22	21	20	19	18	17	16
EXENEXT							
15	14	13	12	11	10	9	8
NEXT							
7	6	5	4	3	2	1	0
NEXT							

位	描述	
[31:16]	EXENEXT	**PDMA再下一个描述符表偏移地址** 该寄存器是系统内存的再下一个描述符表的偏移地址。 **注意:**该域写操作无效。
[15:0]	NEXT	**PDMA下一个描述符表偏移地址** 该寄存器是系统内存的下一个描述符表的偏移地址。 **写操作：** 如果系统内存的基地址是0x2000_0000 (PDMA_SCATBA)，且下一个描述符表地址是从0x2000_0100开始，该域必须填0x0100。 **读操作：** 当工作在scatter-gather模式，最后两位FIRST[1:0]保留。 **注1:**下一个描述符表地址必须是字对齐。 **注2:** 在填写传输任务到描述符表格之前，用户必须检查描述符表是否完成。

当前 Scatter-gather 描述符表格地址(PDMA_CURSCATn)

寄存器	偏移	R/W	描述	复位值
PDMA_CURSCATn	CURSCAT_BA + 0x004 * n	R	PDMA通道n的当前Scatter-Gather描述符表地址	0xXXXX_XXXX

31	30	29	28	27	26	25	24
CURADDR							
23	22	21	20	19	18	17	16
CURADDR							
15	14	13	12	11	10	9	8
CURADDR							
7	6	5	4	3	2	1	0
CURADDR							

位	描述	
[31:0]	CURADDR	**PDMA 当前描述地址寄存器(只读)** 该寄存器是PDMA控制器的当前32-位描述符地址。 **注:** 该寄存器只读，仅用于Scatter-Gather模式来表示当前描述符地址。

通道控制寄存器(PDMA_CHCTL)

寄存器	偏移	R/W	描述	复位值
PDMA_CHCTL	PDMA_BA + 0x400	R/W	PDMA 通道控制寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
CHEN15	CHEN14	CHEN13	CHEN12	CHEN11	CHEN10	CHEN9	CHEN8
7	6	5	4	3	2	1	0
CHEN7	CHEN6	CHEN5	CHEN4	CHEN3	CHEN2	CHEN1	CHEN0

位	描述	
[31:16]	**Reserved**	保留。
[n] n=0,1..15	**CHENn**	**PDMA 通道使能位** 该位为"1"是使能PDMA通道n操作，如果不使能该位，该通道无法工作。 0 = 禁用 PDMA通道 [n]。 1 = 使能 PDMA通道 [n]。 **注意:** PDMA_PAUSE 或 PDMA_CHRST寄存器对应的位被置位，会清0该位。

PDMA传输暂停控制寄存器(PDMA_PAUSE)

寄存器	偏移	R/W	描述	复位值
PDMA_PAUSE	PDMA_BA + 0x404	W	PDMA 传输暂停控制寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
PAUSE15	PAUSE14	PAUSE13	PAUSE12	PAUSE11	PAUSE10	PAUSE9	PAUSE8
7	6	5	4	3	2	1	0
PAUSE7	PAUSE6	PAUSE5	PAUSE4	PAUSE3	PAUSE2	PAUSE1	PAUSE0

位	描述	
[31:16]	Reserved	保留。
[n] n=0,1..15	PAUSEn	**PDMA 传输暂停控制寄存器(只写)** 用户可以暂停PDMA传输，通过PAUSEn位设定。当软件设定PAUSEn位，将停止正在传输数据的通道，然后清除通道使能位CHEN(PDMA_CHCTL [n], n=0,1..15)及传输请求标志(PDMA_TRGSTS[n:0], n=0,1..15)。如果重新使能暂停的通道，剩余的传输会继续。 0 = 无效。 1 = 停止PDMA的通道[n]传输。

PDMA 软件请求寄存器(PDMA_SWREQ)

寄存器	偏移	R/W	描述	复位值
PDMA_SWREQ	PDMA_BA + 0x408	W	PDMA 软件请求寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
SWREQ15	SWREQ14	SWREQ13	SWREQ12	SWREQ11	SWREQ10	SWREQ9	SWREQ8
7	6	5	4	3	2	1	0
SWREQ7	SWREQ6	SWREQ5	SWREQ4	SWREQ3	SWREQ2	SWREQ1	SWREQ0

位	描述	
[31:16]	**Reserved**	保留。
[n] n=0,1..15	**SWREQn**	**PDMA 软件请求寄存器 (只写)** 设置该位为"1"会在PDMA通道n产生一个软件请求。 0 = 无效。 1 = 产生一个软件请求。 **注1：**用户可以通过读寄存器PDMA_TRGSTS来知道哪个通道被触发，触发标志可通过软件请求或外设请求触发。 **注2：**如果用户没有使能相关的PDMA通道，软件请求会被忽略。

PDMA通道请求状态控制器(PDMA_TRGSTS)

寄存器	偏移	R/W	描述	复位值
PDMA_TRGSTS	PDMA_BA + 0x40C	R	PDMA通道请求状态寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
REQSTS15	REQSTS14	REQSTS13	REQSTS12	REQSTS11	REQSTS10	REQSTS9	REQSTS8
7	6	5	4	3	2	1	0
REQSTS7	REQSTS6	REQSTS5	REQSTS4	REQSTS3	REQSTS2	REQSTS1	REQSTS0

位	描述	
[31:16]	**Reserved**	保留。
[n] n=0,1..15	**REQSTSn**	**PDMA 通道请求状态 (只读)** 这个标志用来表示通道n是否有软件或外设请求。当PDMA控制器完成通道传输，该位会被自动清除。 0 = PDMA 通道n无请求。 1 = PDMA 通道n有请求。 **注:** 如果软件通过设定PDMA_ PAUSE或PDMA_CHRST 寄存器暂停或复位每个PDMA传输，该位会在完成当前传输后自动清零。

PDMA固定优先级设定寄存器(PDMA_PRISET)

寄存器	偏移	R/W	描述	复位值
PDMA_PRISET	PDMA_BA + 0x410	R/W	PDMA固定优先级设定寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
FPRISET15	FPRISET14	FPRISET13	FPRISET12	FPRISET11	FPRISET10	FPRISET9	FPRISET8
7	6	5	4	3	2	1	0
FPRISET7	FPRISET6	FPRISET5	FPRISET4	FPRISET3	FPRISET2	FPRISET1	FPRISET0

位	描述	
[31:16]	**Reserved**	保留。
[n] n=0,1..15	**FPRISETn**	**PDMA 固定优先级设定寄存器** 设置该位为”1” 来使能固定优先级的等级。 **写操作：** 0 = 无效。 1 = 设定PDMA通道n为固定优先级通道。 **读操作：** 0 = PDMA是轮循调度(round-robin)优先级。 1 = PDMA是固定优先级。 **注:**该位只设定为固定优先级，通过设定寄存器PDMA_PRICLR来清除固定优先级。

PDMA固定优先级清除寄存器(PDMA_PRICLR)

寄存器	偏移	R/W	描述	复位值
PDMA_PRICLR	PDMA_BA + 0x414	W	PDMA固定优先级清除寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
FPRICLR15	FPRICLR14	FPRICLR13	FPRICLR12	FPRICLR11	FPRICLR10	FPRICLR9	FPRICLR8
7	6	5	4	3	2	1	0
FPRICLR7	FPRICLR6	FPRICLR5	FPRICLR4	FPRICLR3	FPRICLR2	FPRICLR1	FPRICLR0

位	描述	
[31:16]	**Reserved**	保留。
[n] n=0,1..15	**FPRICLRn**	**PDMA 固定优先级清除寄存器 (只写)** 设定该位为1来清除固定优先级等级。 0 = 无效。 1 = 清除PDMA通道n的固定优先等级设定。 **注：** 用户可以通过读寄存器PDMA_PRISET来知道当前通道的优先等级。

PDMA中断使能寄存器(PDMA_INTEN)

寄存器	偏移	R/W	描述	复位值
PDMA_INTEN	PDMA_BA + 0x418	R/W	PDMA中断使能寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
INTEN15	INTEN14	INTEN13	INTEN12	INTEN11	INTEN10	INTEN9	INTEN8
7	6	5	4	3	2	1	0
INTEN7	INTEN6	INTEN5	INTEN4	INTEN3	INTEN2	INTEN1	INTEN0

位	描述	
[31:16]	Reserved	保留。
[n] n=0,1..15	INTENn	PDMA 中断使能位 该寄存器用于使能PDMA的通道n中断。 0 = 禁用PDMA通道n的中断。 1 = 使能PDMA通道n的中断。

PDMA中断状态寄存器(PDMA_INTSTS)

寄存器	偏移	R/W	描述	复位值
PDMA_INTSTS	PDMA_BA + 0x41C	R/W	PDMA中断状态寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved						REQTOF1	REQTOF0
7	6	5	4	3	2	1	0
Reserved					ALIGNF	TDIF	ABTIF

位	描述	
[31:10]	**Reserved**	保留。
[9]	**REQTOF1**	**通道1请求超时标志** 该标志表示PDMA控制器已经等待了外设请求由PDMA_TOC1定义的时间周期，用户可以写1清除这些位。 0 = 没有请求超时。 1 = 外设请求超时。 **注**：清0该位之前请禁用超时功能。
[8]	**REQTOF0**	**通道0请求超时标志** 该标志表示PDMA控制器已经等待了外设请求由PDMA_TOC0定义的时间周期，用户可以写1清除这些位。 0 = 没有请求超时。 1 = 外设请求超时。 **注**：清0该位之前请禁用超时功能。
[7:3]	**Reserved**	保留。
[2]	**ALIGNF**	**传输对齐中断标志 (只读)** 0 = PDMA 通道源地址和目的地址都遵循传输宽度的设定对齐。 1 = PDMA通道源地址和目的地址没有遵循传输宽度的设定对齐。
[1]	**TDIF**	**传输完成中断标志(只读)** 该位指示PDMA控制器已经完成了数据传输；用户可以读取寄存器PDMA_TDSTS来获知哪个通道完成了数据传输。 0 = PDMA通道未完成了数据传输。 1 = PDMA通道已经完成了数据传输。
[0]	**ABTIF**	**PDMA 读/写目标终止中断标志(只读)**

位	描述	
		该位指示PDMA的传输数据目标存在终止错误；软件可以读寄存器PDMA_ABTSTS来找出哪个通道存在目标终止错误。 0 = 未收到AHB总线错误响应。 1 = 收到AHB总线错误响应。

PDMA通道读/写目标终止标志寄存器(PDMA_ABTSTS)

寄存器	偏移	R/W	描述	复位值
PDMA_ABTSTS	PDMA_BA + 0x420	R/W	PDMA通道读/写目标终止标志寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
ABTIF15	ABTIF14	ABTIF13	ABTIF12	ABTIF11	ABTIF10	ABTIF9	ABTIF8
7	6	5	4	3	2	1	0
ABTIF7	ABTIF6	ABTIF5	ABTIF4	ABTIF3	ABTIF2	ABTIF1	ABTIF0

位	描述	
[31:16]	**Reserved**	保留。
[n] n=0,1..15	**ABTIFn**	**PDMA 读/写目标终止中断状态标志** 该位指示哪个PDMA控制器存在目标终止错误；用户可以写1清除这些位。 0 = 当通道n传输数据时，未收到AHB总线错误响应。 1 = 当通道n传输数据时，收到AHB总线错误响应。 **注:** 如果通道n目标终止，REQSRC应设置为0以禁用外设请求。

PDMA通道传输数据完成标志寄存器(PDMA_TDSTS)

寄存器	偏移	R/W	描述	复位值
PDMA_TDSTS	PDMA_BA + 0x424	R/W	PDMA通道传输完成标志寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
TDIF15	TDIF14	TDIF13	TDIF12	TDIF11	TDIF10	TDIF9	TDIF8
7	6	5	4	3	2	1	0
TDIF7	TDIF6	TDIF5	TDIF4	TDIF3	TDIF2	TDIF1	TDIF0

位	描述	
[31:16]	**Reserved**	保留。
[n] n=0,1..15	**TDIFn**	**传输完成标志寄存器** 该位指示PDMA控制器通道传输数据是否完成，用户可以写1清除这些位。 0 = PDMA 通道传输数据未完成。 1 = PDMA 通道传输数据完成。

PDMA 传输对齐状态寄存器(PDMA_ALIGN)

寄存器	偏移	R/W	描述	复位值
PDMA_ALIGN	PDMA_BA + 0x428	R/W	PDMA 传输对齐状态寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
ALIGN15	ALIGN14	ALIGN13	ALIGN12	ALIGN11	ALIGN10	ALIGN9	ALIGN8
7	6	5	4	3	2	1	0
ALIGN7	ALIGN6	ALIGN5	ALIGN4	ALIGN3	ALIGN2	ALIGN1	ALIGN0

位	描述	
[31:16]	**Reserved**	保留。
[n] n=0,1..15	**ALIGNn**	**传输对齐标志寄存器** 0 = PDMA 通道源地址和目的地址都遵循传输宽度的设定对齐。 1 = PDMA通道源地址和目的地址没有遵循传输宽度的设定对齐。 **注:**源地址和目标地址必须对齐。

PDMA传输激活标志寄存器(PDMA_TACTSTS)

寄存器	偏移	R/W	描述	复位值
PDMA_TACTSTS	PDMA_BA + 0x42C	R	PDMA 传输激活标志寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
TXACTF15	TXACTF14	TXACTF13	TXACTF12	TXACTF11	TXACTF10	TXACTF9	TXACTF8
7	6	5	4	3	2	1	0
TXACTF7	TXACTF6	TXACTF5	TXACTF4	TXACTF3	TXACTF2	TXACTF1	TXACTF0

位	描述	
[31:16]	**Reserved**	保留。
[n] n=0,1..15	**TXACTFn**	**传输激活标志寄存器(只读)** 该位用于指示哪个PDMA通道正在传输。 0 = PDMA通道没有在传输数据。 1 = PDMA正在传输数据。

PDMA 超时预分频器寄存器(PDMA_TOUTPSC)

寄存器	偏移	R/W	描述	复位值
PDMA_TOUTPSC	PDMA_BA + 0x430	R/W	PDMA 超时预分频器寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved	TOUTPSC1			Reserved	TOUTPSC0		

位	描述	
[31:7]	**Reserved**	保留
[6:4]	**TOUTPSC1**	**PDMA 通道 1 超时时钟源预分频位** 000 = PDMA 通道 1 超时时钟源是 HCLK/2^8。 001 = PDMA通道 1 超时时钟源是HCLK/2^9。 010 = PDMA通道 1 超时时钟源是HCLK/2^{10}。 011 = PDMA通道 1 超时时钟源是HCLK/2^{11}。 100 = PDMA通道 1 超时时钟源是HCLK/2^{12}。 101 = PDMA通道 1 超时时钟源是HCLK/2^{13}。 110 = PDMA通道 1 超时时钟源是HCLK/2^{14}。 111 = PDMA通道 1 超时时钟源是HCLK/2^{15}。
[3]	**Reserved**	保留。
[2:0]	**TOUTPSC0**	**PDMA 通道 0超时时钟源预分频位** 000 = PDMA 通道 0 超时时钟源是HCLK/2^8。 001 = PDMA 通道 0 超时时钟源是HCLK/2^9。 010 = PDMA 通道 0 超时时钟源是 HCLK/2^{10}。 011 = PDMA 通道 0 超时时钟源是 HCLK/2^{11}。 100 = PDMA 通道 0 超时时钟源是 HCLK/2^{12}。 101 = PDMA 通道 0 超时时钟源是HCLK/2^{13}。 110 = PDMA 通道 0 超时时钟源是HCLK/2^{14}。 111 = PDMA 通道 0 超时时钟源是HCLK/2^{15}。

PDMA 超时使能寄存器(PDMA_TOUTEN)

寄存器	偏移	R/W	描述	复位值
PDMA_TOUTEN	PDMA_BA + 0x434	R/W	PDMA 超时使能寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved						TOUTEN1	TOUTEN0

位	描述	
[31:2]	Reserved	保留。
[n] n=0,1	TOUTENn	PDMA 超时使能位 0 = PDMA 通道 n 超时功能禁止。 1 = PDMA 通道 n 超时功能使能。

PDMA 超时中断使能寄存器(PDMA_TOUTIEN)

寄存器	偏移	R/W	描述	复位值
PDMA_TOUTIEN	PDMA_BA + 0x438	R/W	PDMA 超时中断使能寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved						TOUTIEN1	TOUTIEN0

位	描述	
[31:2]	Reserved	保留。
[n] n=0,1	TOUTIENn	PDMA 超时中断使能位 0 = PDMA 通道 n 超时中断禁止。 1 = PDMA 通道 n 超时中断使能。

PDMA Scatter-gather描述符表基地址寄存器(PDMA_SCATBA)

寄存器	偏移	R/W	描述	复位值
PDMA_SCATBA	PDMA_BA + 0x43C	R/W	PDMA Scatter-Gather描述符表基地址寄存器	0x2000_0000

31	30	29	28	27	26	25	24
SCATBA							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
SCATBA							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved							

位	描述	
[31:16]	**SCATBA**	**PDMA Scatter-Gather 描述符表地址寄存器** 在Scatter-Gather模式，该寄存器是基地址用于计算下一个链接地址。 下一个链接地址是： 下一个链接地址= PDMA_SCATBA + PDMA_DSCT_ FIRST。 **注：**仅在Scatter-Gather模式有效。
[15:0]	**Reserved**	保留。

PDMA 超时周期计数器寄存器0 (PDMA_TOC0_1)

寄存器	偏移	R/W	描述	复位值
PDMA_TOC0_1	PDMA_BA + 0x440	R/W	PDMA 超时计数器通道1和通道0寄存器	0xFFFF_FFFF

31	30	29	28	27	26	25	24
TOC1							
23	22	21	20	19	18	17	16
TOC1							
15	14	13	12	11	10	9	8
TOC0							
7	6	5	4	3	2	1	0
TOC0							

位	描述	
[31:16]	TOC1	**通道 1超时计数器** 该域控制通道1超时周期时间，计数单位是基于10 kHz时钟。超时周期示例参考TOC0位描述。
[15:0]	TOC0	**通道 0超时计数器** 该域控制通道0超时周期时间，计数单位是基于TOUTPSC0 (PDMA_TOUTPSC[2:0])定义的时钟数。 超时周期 = （超时周期时钟）*（16-位TOCn）,n=0,1。

PDMA 通道复位寄存器(PDMA_CHRST)

寄存器	偏移	R/W	描述	复位值
PDMA_CHRST	PDMA_BA + 0x460	R/W	PDMA 通道复位寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
CH15RST	CH14RST	CH13RST	CH12RST	CH11RST	CH10RST	CH9RST	CH8RST
7	6	5	4	3	2	1	0
CH7RST	CH6RST	CH5RST	CH4RST	CH3RST	CH2RST	CH1RST	CH0RST

位	描述	
[31:16]	**Reserved**	保留。
[15:0]	**CHnRST**	**通道 N 复位** 0 = 对应通道n不复位。 1 = 对应通道n复位。 **注:**该功能不支持跨步模式。

PDMA请求源选择寄存器0 (PDMA_REQSEL0_3)

寄存器	偏移	R/W	描述	复位值
PDMA_REQSEL0_3	PDMA_BA + 0x480	R/W	PDMA请求源选择寄存器0	0x0000_0000

31	30	29	28	27	26	25	24
Reserved	REQSRC3						
23	22	21	20	19	18	17	16
Reserved	REQSRC2						
15	14	13	12	11	10	9	8
Reserved	REQSRC1						
7	6	5	4	3	2	1	0
Reserved	REQSRC0						

位	描述	
[31]	**Reserved**	保留。
[30:24]	**REQSRC3**	**通道3请求源选择** 该域定义哪个外设连接到PDMA通道3.。用户可以通过设定寄存器REQSRC3配置外设。 **注:**通道配置与寄存器REQSRC0相同。请参考REQSRC0说明。
[23]	**Reserved**	保留。
[22:16]	**REQSRC2**	**通道2请求源选择** 该域定义哪个外设连接到PDMA通道2.。用户可以通过设定寄存器REQSRC2配置外设。 **注:**通道配置与寄存器REQSRC0相同。请参考REQSRC0说明。
[15]	**Reserved**	保留。
[14:8]	**REQSRC1**	**通道1请求源选择** 该域定义哪个外设连接到PDMA通道1。用户可以通过设定寄存器REQSRC1配置外设。 **注:**通道配置与寄存器REQSRC0相同。请参考REQSRC0说明。
[7]	**Reserved**	保留。
[6:0]	**REQSRC0**	**通道0请求源选择** 该域定义哪个外设连接到PDMA通道0。用户可以通过设定寄存器REQSRC0配置外设。 0 =禁止 PDMA 外设请求。 1 =保留。 2 =通道连接到USB_TX。 3 =通道连接到USB_RX。 4 =通道连接到UART0_TX。 5 =通道连接到UART0_RX。

位	描述	
		6 =通道连接到UART1_TX。 7 =通道连接到UART1_RX。 8 =通道连接到UART2_TX。 9 =通道连接到UART2_RX。 10=通道连接到UART3_TX。 11 =通道连接到UART3_RX。 12 =通道连接到UART4_TX。 13 =通道连接到UART4_RX。 14 =通道连接到UART5_TX。 15 =通道连接到UART5_RX。 16 =通道连接到 USCI0_TX。 17 =通道连接到USCI0_RX。 18 =通道连接到USCI1_TX。 19 =通道连接到USCI1_RX。 20 =通道连接到QSPI0_TX。 21 =通道连接到QSPI0_RX。 22 =通道连接到 SPI0_TX。 23 =通道连接到SPI0_RX。 24 =通道连接到SPI1_TX。 25 =通道连接到SPI1_RX。 26 =通道连接到SPI2_TX。 27 =通道连接到SPI2_RX。 28 =通道连接到SPI3_TX。 29 =通道连接到SPI3_RX。 30 =通道连接到QSPI1_TX。 31 =通道连接到QSPI1_RX。 32 =通道连接到EPWM0_P1_RX。 33 =通道连接到EPWM0_P2_RX。 34 =通道连接到 EPWM0_P3_RX。 35 =通道连接到EPWM1_P1_RX。 36 =通道连接到EPWM1_P2_RX。 37 =通道连接到EPWM1_P3_RX。 38 =通道连接到I^2C0_TX。 39 =通道连接到I^2C0_RX。 40 =通道连接到I^2C1_TX。 41 =通道连接到I^2C1_RX。 42 =通道连接到I^2C2_TX。 43 =通道连接到I^2C2_RX。 44 =通道连接到I^2S0_TX。 45 =通道连接到I^2S0_RX。 46 =通道连接到TMR0。 47 =通道连接到TMR1。

位	描述	
		48 =通道连接到TMR2。 49 =通道连接到TMR3。 50 =通道连接到EADC0_RX。 51 =通道连接到DAC0_TX。 52 =通道连接到 DAC1_TX。 53 =通道连接到EPWM0_CH0_TX。 54 =通道连接到EPWM0_CH1_TX。 55 =通道连接到EPWM0_CH2_TX。 56 =通道连接到EPWM0_CH3_TX。 57 =通道连接到EPWM0_CH4_TX。 58 =通道连接到EPWM0_CH5_TX。 59 =通道连接到EPWM1_CH0_TX。 60 =通道连接到EPWM1_CH1_TX。 61 =通道连接到EPWM1_CH2_TX。 62 =通道连接到EPWM1_CH3_TX。 63 =通道连接到EPWM1_CH4_TX。 64 =通道连接到EPWM1_CH5_TX。 65 =保留。 66 =通道连接到UART6_TX。 67 =通道连接到UART6_RX。 68 =通道连接到 UART7_TX。 69 =通道连接到 UART7_RX。 70 =通道连接到EADC1_RX。 其它 = 保留。 **注1：**一个外设不可以同时连接到两个通道。 **注2：**当内存与内存之间数据传输时，该寄存器无效。

PDMA请求源选择寄存器1 (PDMA_REQSEL4_7)

寄存器	偏移	R/W	描述	复位值
PDMA_REQSEL4_7	PDMA_BA + 0x484	R/W	PDMA请求源选择寄存器1	0x0000_0000

31	30	29	28	27	26	25	24
Reserved	REQSRC7						
23	22	21	20	19	18	17	16
Reserved	REQSRC6						
15	14	13	12	11	10	9	8
Reserved	REQSRC5						
7	6	5	4	3	2	1	0
Reserved	REQSRC4						

位	描述	
[31]	**Reserved**	保留。
[30:24]	**REQSRC7**	**通道7请求源选择** 该域定义哪个外设连接到PDMA通道7.。用户可以通过设定寄存器REQSRC7配置外设。 **注:**通道配置与寄存器REQSRC0相同。请参考REQSRC0说明。
[23]	**Reserved**	保留。
[22:16]	**REQSRC6**	**通道6请求源选择** 该域定义哪个外设连接到PDMA通道6。用户可以通过设定寄存器REQSRC6配置外设。 **注:**通道配置与寄存器REQSRC0相同。请参考REQSRC0说明。
[15]	**Reserved**	保留。
[14:8]	**REQSRC5**	**通道5请求源选择** 该域定义哪个外设连接到PDMA通道5.。用户可以通过设定寄存器REQSRC5配置外设。 **注:**通道配置与寄存器REQSRC0相同。请参考REQSRC0说明。
[7]	**Reserved**	保留。
[6:0]	**REQSRC4**	**通道4请求源选择** 该域定义哪个外设连接到PDMA通道4.。用户可以通过设定寄存器REQSRC4配置外设。 **注:**通道配置与寄存器REQSRC0相同。请参考REQSRC0说明。

PDMA 请求源选择寄存器2 (PDMA_REQSEL8_11)

寄存器	偏移	R/W	描述	复位值
PDMA_REQSEL8_11	PDMA_BA + 0x488	R/W	PDMA 请求源选择寄存器2	0x0000_0000

31	30	29	28	27	26	25	24
Reserved	REQSRC11						
23	22	21	20	19	18	17	16
Reserved	REQSRC10						
15	14	13	12	11	10	9	8
Reserved	REQSRC9						
7	6	5	4	3	2	1	0
Reserved	REQSRC8						

位	描述	
[31]	**Reserved**	保留。
[30:24]	**REQSRC11**	**通道11请求源选择** 该域定义哪个外设连接到PDMA通道11.。用户可以通过设定寄存器REQSRC11配置外设。 **注:**通道配置与寄存器REQSRC0相同。请参考REQSRC0说明。
[23]	**Reserved**	保留。
[22:16]	**REQSRC10**	**通道10请求源选择** 该域定义哪个外设连接到PDMA通道10.。用户可以通过设定寄存器REQSRC10配置外设。 **注:**通道配置与寄存器REQSRC0相同。请参考REQSRC0说明。
[15]	**Reserved**	保留。
[14:8]	**REQSRC9**	**通道9请求源选择** 该域定义哪个外设连接到PDMA通道9.。用户可以通过设定寄存器REQSRC9配置外设。 **注:**通道配置与寄存器REQSRC0相同。请参考REQSRC0说明。
[7]	**Reserved**	保留。
[6:0]	**REQSRC8**	**通道8请求源选择** 该域定义哪个外设连接到PDMA通道8.。用户可以通过设定寄存器REQSRC8配置外设。 **注:**通道配置与寄存器REQSRC0相同。请参考REQSRC0说明。

PDMA 请求源选择寄存器3 (PDMA_REQSEL12_15)

寄存器	偏移	R/W	描述	复位值
PDMA_REQSEL12_15	PDMA_BA + 0x48C	R/W	PDMA 请求源选择寄存器3	0x0000_0000

31	30	29	28	27	26	25	24
Reserved	REQSRC15						
23	22	21	20	19	18	17	16
Reserved	REQSRC14						
15	14	13	12	11	10	9	8
Reserved	REQSRC13						
7	6	5	4	3	2	1	0
Reserved	REQSRC12						

位	描述	
[31]	Reserved	保留。
[30:24]	REQSRC15	**通道15请求源选择** 该域定义哪个外设连接到PDMA通道15.。用户可以通过设定寄存器REQSRC15配置外设。 **注:**通道配置与寄存器REQSRC0相同。请参考REQSRC0说明。
[23]	Reserved	保留。
[22:16]	REQSRC14	**通道14请求源选择** 该域定义哪个外设连接到PDMA通道14.。用户可以通过设定寄存器REQSRC14配置外设。 **注:**通道配置与寄存器REQSRC0相同。请参考REQSRC0说明。
[15]	Reserved	保留。
[14:8]	REQSRC13	**通道13请求源选择** 该域定义哪个外设连接到PDMA通道13.。用户可以通过设定寄存器REQSRC13配置外设。 **注:**通道配置与寄存器REQSRC0相同。请参考REQSRC0说明。
[7]	Reserved	保留。
[6:0]	REQSRC12	**通道12请求源选择** 该域定义哪个外设连接到PDMA通道12.。用户可以通过设定寄存器REQSRC12配置外设。 **注:**通道配置与寄存器REQSRC0相同。请参考REQSRC0说明。

PDMA 跨步传输计数寄存器n (PDMA_STCn)

寄存器	偏移	R/W	描述	复位值
PDMA_STC0	PDMA_BA+0x500	R/W	PDMA 通道 0的跨步传输计数寄存器	0x0000_0000
PDMA_STC1	PDMA_BA+0x508	R/W	PDMA 通道1的跨步传输计数寄存器	0x0000_0000
PDMA_STC2	PDMA_BA+0x510	R/W	PDMA 通道2的跨步传输计数寄存器	0x0000_0000
PDMA_STC3	PDMA_BA+0x518	R/W	PDMA 通道3的跨步传输计数寄存器	0x0000_0000
PDMA_STC4	PDMA_BA+0x520	R/W	PDMA 通道4的跨步传输计数寄存器	0x0000_0000
PDMA_STC5	PDMA_BA+0x528	R/W	PDMA 通道5的跨步传输计数寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
11	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
STC							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
STC							

位	描述	
[31:16]	**Reserved**	保留。
[15:0]	**STC**	**PDMA 跨步传输计数** 这16-位寄存器定义每一行跨步传输计数值，实际传输数为(STC + 1)。

PDMA 地址跨步偏移量控制寄存器n(PDMA_ASOCTLn)

寄存器	偏移	R/W	描述	复位值
PDMA_ASOCTL0	PDMA_BA+0x504	R/W	PDMA 通道 0的地址跨步偏移量寄存器	0x0000_0000
PDMA_ASOCTL1	PDMA_BA+0x50C	R/W	PDMA 通道1的地址跨步偏移量寄存器	0x0000_0000
PDMA_ASOCTL2	PDMA_BA+0x514	R/W	PDMA 通道2的地址跨步偏移量寄存器	0x0000_0000
PDMA_ASOCTL3	PDMA_BA+0x51C	R/W	PDMA 通道3的地址跨步偏移量寄存器	0x0000_0000
PDMA_ASOCTL4	PDMA_BA+0x524	R/W	PDMA 通道4的地址跨步偏移量寄存器	0x0000_0000
PDMA_ASOCTL5	PDMA_BA+0x52C	R/W	PDMA 通道5的地址跨步偏移量寄存器	0x0000_0000

31	30	29	28	27	26	25	24
DASOL							
11	**22**	**21**	**20**	**19**	**18**	**17**	**16**
DASOL							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
SASOL							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
SASOL							

位	描述	
[31:16]	**DASOL**	**PDMA 目的地址跨步偏移长度** 该16-bit 寄存器定义目的地址每一行跨步传输偏移计数值。
[15:0]	**SASOL**	**PDMA 源地址跨步偏移长度** 该16-bit 寄存器定义源地址每一行跨步传输偏移计数值。

PDMA地址间隔控制寄存器n(PDMA_AICTLn)

寄存器	偏移	R/W	描述	复位值
PDMA_AICTL0	PDMA_BA+0x600	R/W	PDMA通道0地址间隔控制寄存器	0x0000_0000
PDMA_AICTL1	PDMA_BA+0x608	R/W	PDMA通道1地址间隔控制寄存器	0x0000_0000

31	30	29	28	27	26	25	24
DAICNT							
11	22	21	20	19	18	17	16
DAICNT							
15	14	13	12	11	10	9	8
SAICNT							
7	6	5	4	3	2	1	0
SAICNT							

位	描述	
[31:16]	**DAICNT**	**PDMA目标地址间隔计数** 该16-位寄存器定义了每一行目标地址间隔数。 **注:**当重复计数(PDMA_RCNTn) 为0时，该寄存器应置为0。
[15:0]	**SAICNT**	**PDMA源地址间隔计数** 该16-位寄存器定义了每一行源地址间隔数。 **注:**当重复计数(PDMA_RCNTn) 为0时，该寄存器应置为0。

PDMA重复计数寄存器n(PDMA_RCNTn)

寄存器	偏移	R/W	描述	复位值
PDMA_RCNT0	PDMA_BA+0x604	R/W	PDMA通道0重复计数寄存器	0x0000_0000
PDMA_RCNT1	PDMA_BA+0x60C	R/W	PDMA通道1重复计数寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
11	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
RCNT							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
RCNT							

位	描述	
[31:16]	**Reserved**	保留。
[15:0]	**RCNT**	**PDMA重复计数** 该16-位寄存器定义了区块传输的重复次数。 **注:**当跨步功能禁用时，该寄存器应置0。

6.8 定时器控制器(TMR)

6.8.1 概述

定时器控制器包含4组32位定时器，TIMER0~TIMER3，提供用户便捷的计数定时功能。定时器可执行很多功能，如频率测量，时间延迟，时钟发生，外部输入管脚事件计数和外部捕获管脚脉宽测量等。

定时器计数器也提供四个PWM产生器，每个PWM产生器支持两个PWM输出通道，可以是独立模式或互补模式。PWM的输出状态可以通过管脚屏蔽控制、极性控制和刹车控制，支持死区发生器。

6.8.2 特性

6.8.2.1 定时器功能特性

- 4 组32位定时器，带24位向上计数器和一个8位的预分频计数器
- 每个定时器都有独立的时钟源
- 提供单次、周期、触发输出和连续计数四种操作模式
- 通过CNT (TIMERx_CNT[23:0])可读取内部24位向上计数器的值
- 支持事件计数功能
- 通过CAPDAT (TIMERx_CAP[23:0])可读取24位捕获值
- 支持外部管脚捕获，可用于脉宽测量
- 支持外部引脚捕获，可用于复位24位向上定时器
- 如果定时器中断信号产生，支持芯片从空闲/掉电模式唤醒
- 支持Timer0 ~ Timer3超时中断信号或捕获中断信号触发EPWM、BPWM、ADC、DAC和PDMA功能
- 支持当ACMP输出信号跳变触发内部捕获
- 支持内部时钟（HIRC、LIRC）和外部时钟（HXT、LXT）用于捕获事件
- 支持定时器间互触发模式
- 支持事件计数源来自内部USB SOF信号

6.8.2.2 PWM 功能特性

- 支持最大时钟频率到PCLK
- 支持PWM独立模式带两个输出通道
- 支持PWM互补模式带成对的输出通道
  - 12位死区事件插入，带 12位预分频
- 支持12位预分频，从1到4096
- 支持16位PWM计数器
  - 上数、下数和上下数计数操作模式
  - 单次或自动重载计数器操作模式

- 对每一个PWM输出管脚支持屏蔽功能和三态使能
- 支持刹车功能
  - 刹车源来自管脚、模拟比较器和系统安全事件（时钟失败、欠压侦测、SRAM校验错误和CPU锁死）
  - 对刹车源，支持刹车管脚噪音滤波控制
  - 边沿检测刹车源控制刹车状态直到刹车状态清除
  - 电平侦测刹车源用于在刹车条件移除后自动恢复功能
- 支持在下面事件中断：
  - PWM零点、周期点、上数比较点或下数比较点事件
  - 刹车条件发生
- 支持在下面事件触发ADC：
  - PWM零点、周期点、上数比较点或下数比较点事件

6.8.3 框图

定时器控制器和时钟控制框图如下图所示：

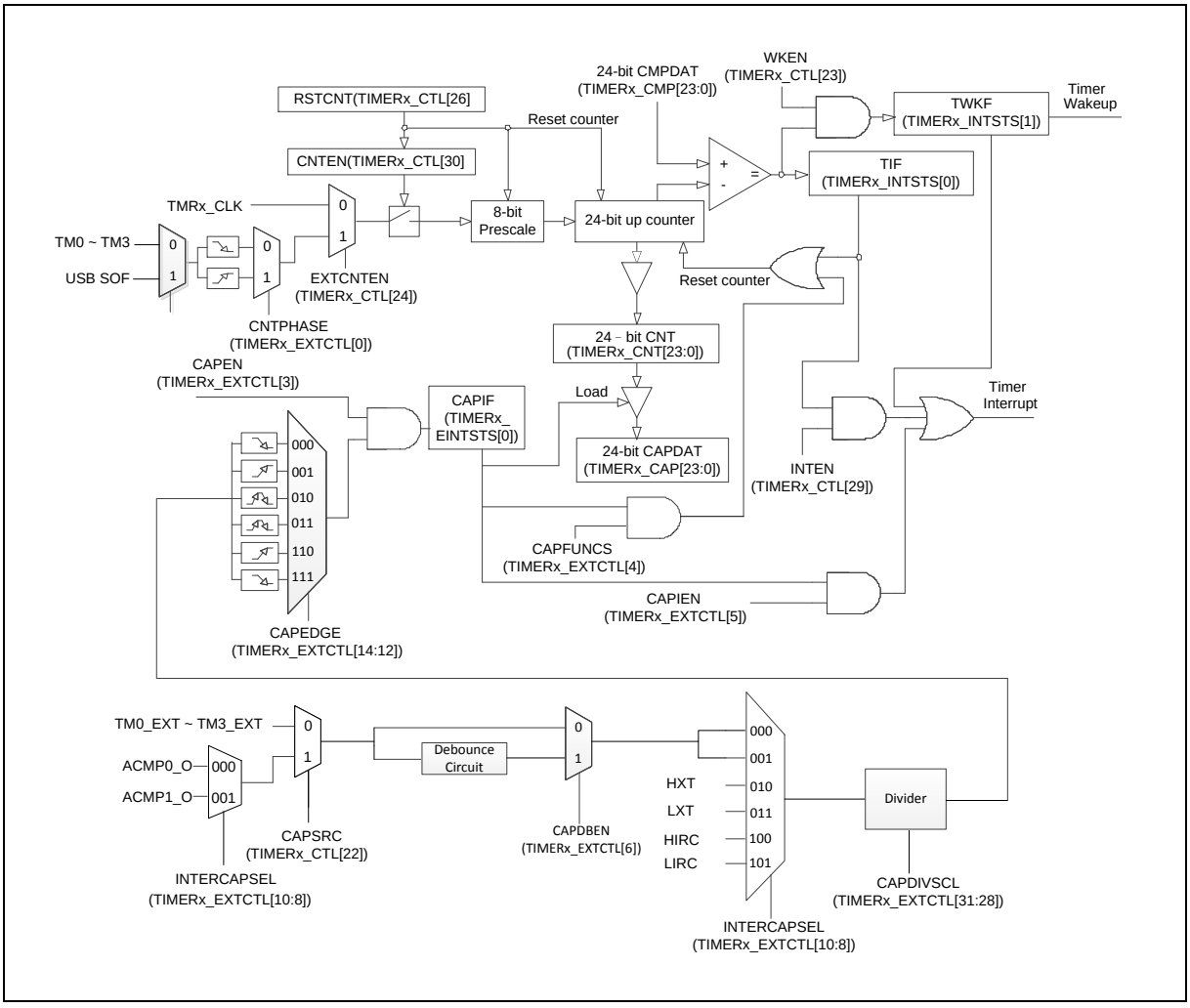


图 6.8-1 定时器控制器框图

设置FUNMODE (TIMERx_ALTCTL[0])为0使能定时器模式。在定时器模式，定时器0~定时器3的时钟源可以在寄存器TMRxCKEN (CLK_APBCLK0[5:2])使能，在寄存器TMR0SEL (CLK_CLKSEL1[10:8]) (定时器0)，TMR1SEL (CLK_CLKSEL1[14:12]) (定时器1)，TMR2SEL (CLK_CLKSEL1[18:16]) (定时器2)，TMR3SEL (CLK_CLKSEL1[22:20]) (定时器3)选择不同时钟频率，如下图：

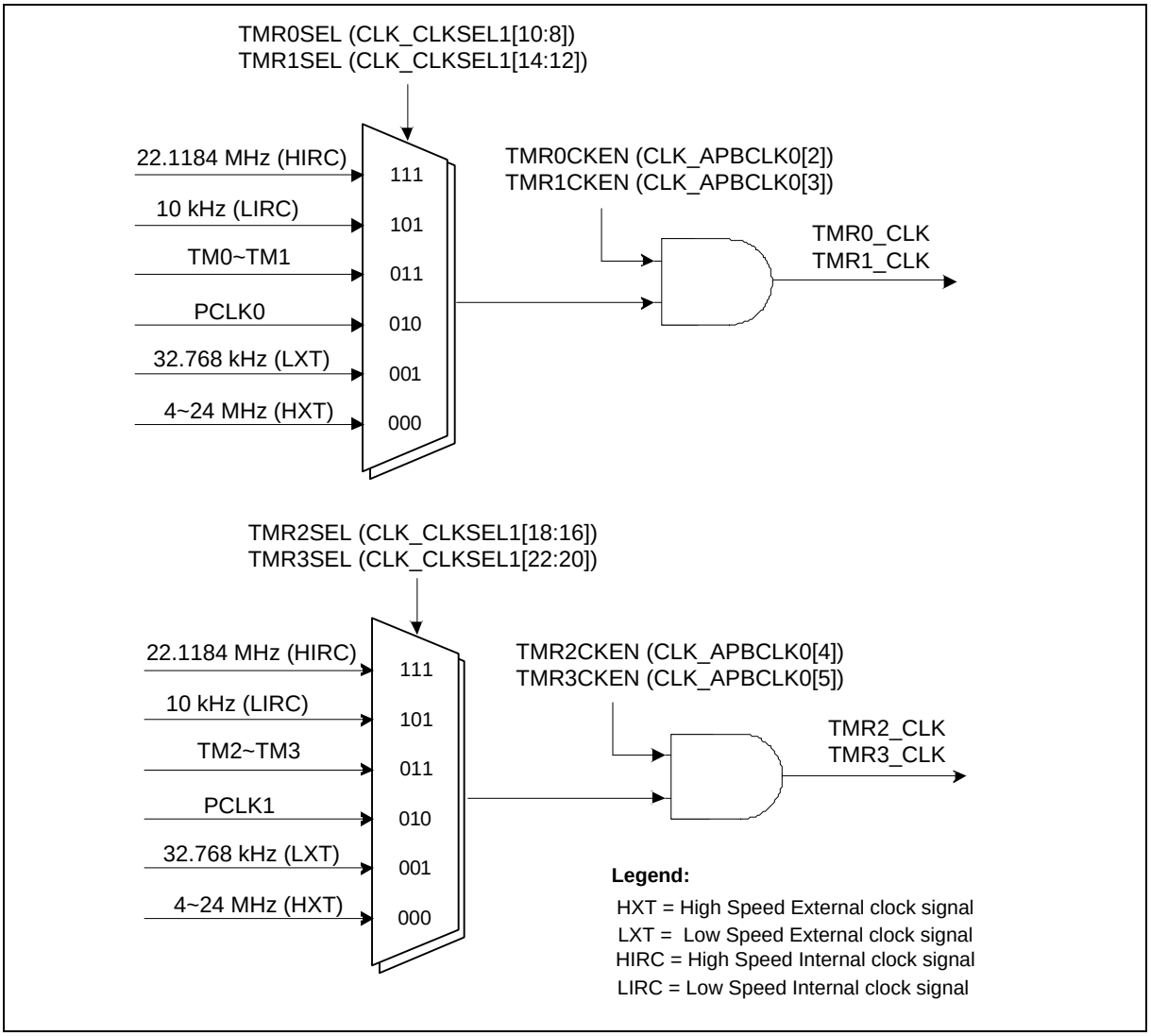


图 6.8-2 定时器控制器的时钟源

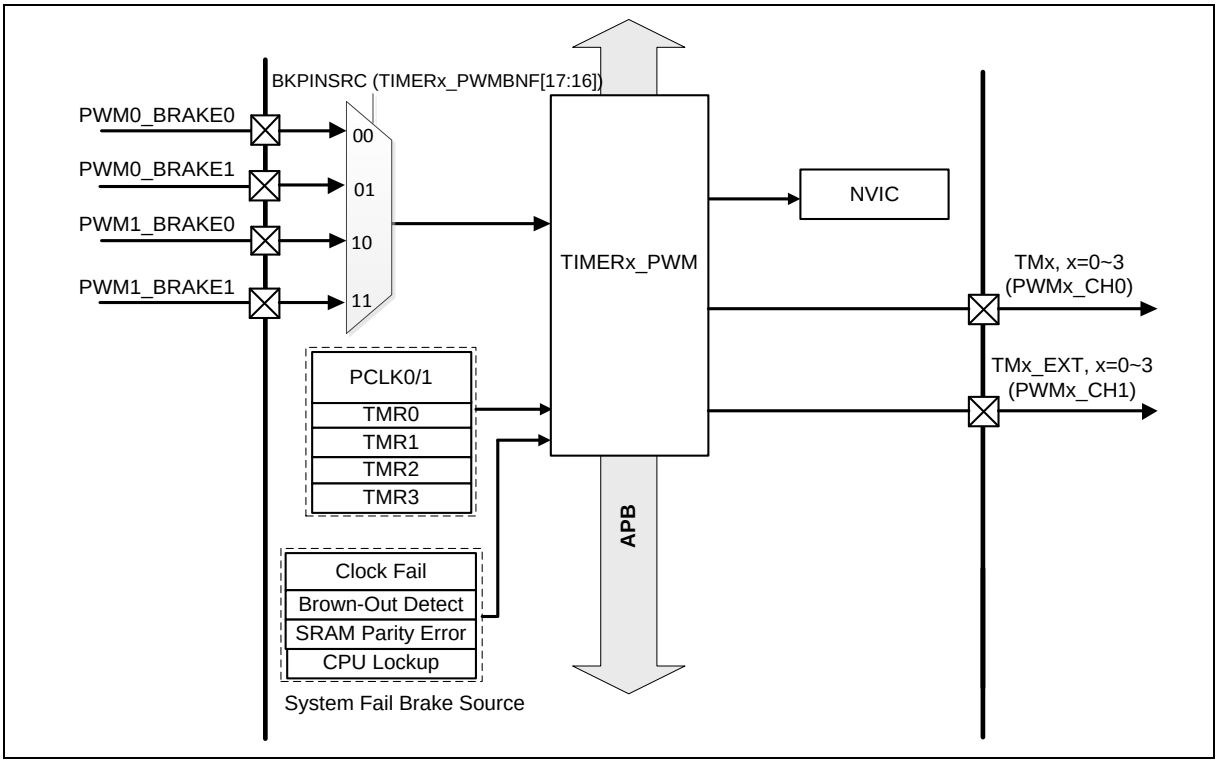


图 6.8-3 PWM 发生器框图

设置 FUNMODE (TIMERx_ALTCTL[0])为1使能PWM模式。在PWM模式，定时器0~定时器3的时钟源可以在寄存器TMRxCKEN (CLK_APBCLK0[5:2])使能，TMR0_CLK和TMR1_CLK时钟源固定为PCLK0，TMR2_CLK 和 TMR3_CLK时钟源固定为PCLK1。PWM系统时钟频率是PCLKx的频率，如图6. 8-4.

PWM计数器的时钟源 (TIMERx_PWMCLK) 可以从PWM 系统时钟 (TMRx_CLK) 或定时器中断事件(TMRx_INT)中选，如图6. 8-5.

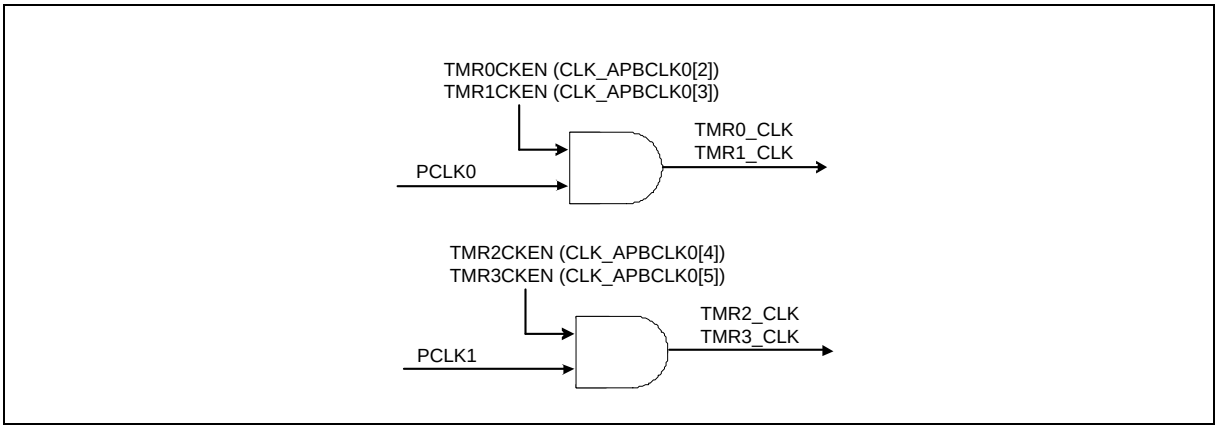


图 6.8-4PWM 系统时钟源控制

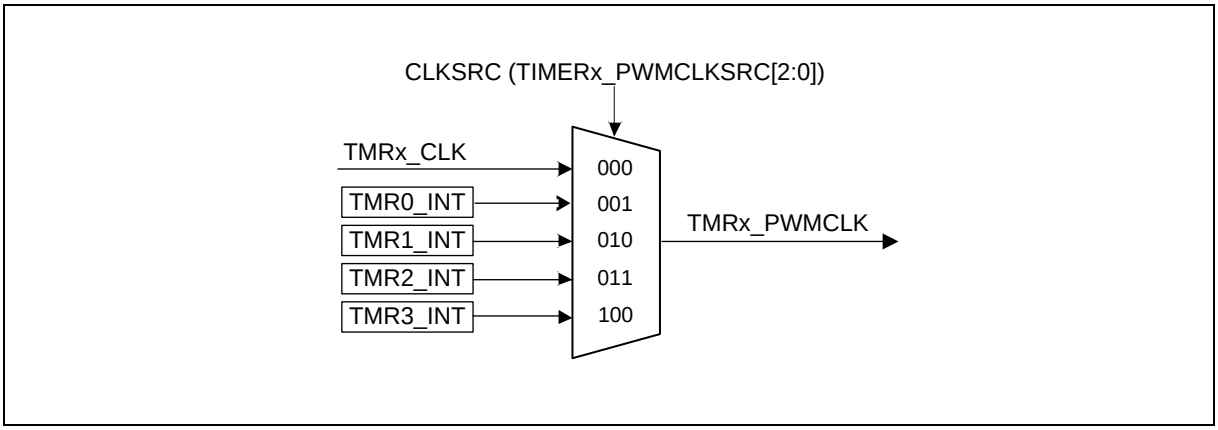


图 6.8-5 PWM 计数器时钟源控制

图6. 8-6和图6. 8-7表示PWM独立模式和互补模式的结构。每个PWM发生器，独立模式和互补模式都支持PWMx_CH0 和 PWMx_CH1 输出通道。

当计数器数到0，PERIOD (TIMERx_PWMPERIOD[15:0]) 或等于 CMP (TIMERx_PWMCMPDAT[15:0]), 相关事件产生。这些事件传到对应的PWM发生器产生PWM脉冲、中断信号和启动ADC转换的触发信号。输出控制模块用于决定PWM脉冲的输出，输出控制模块的刹车功能也产生中断事件，死区控制仅在PWM互补模式有效。

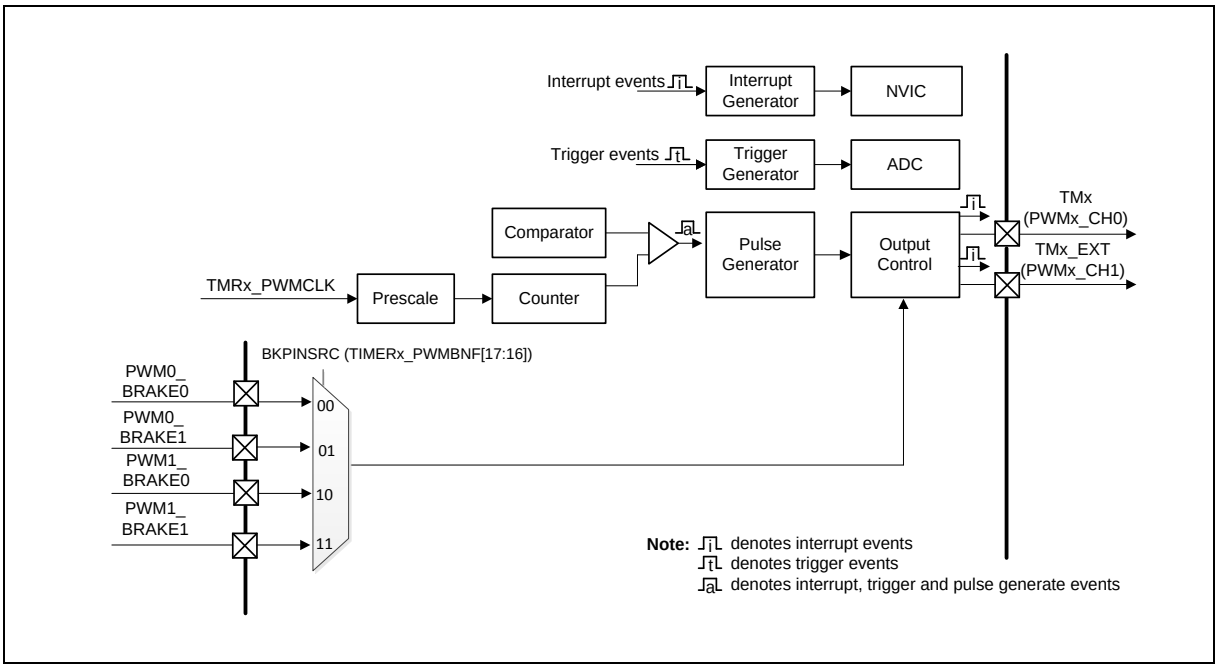


图 6.8-6 PWM 独立模式结构图

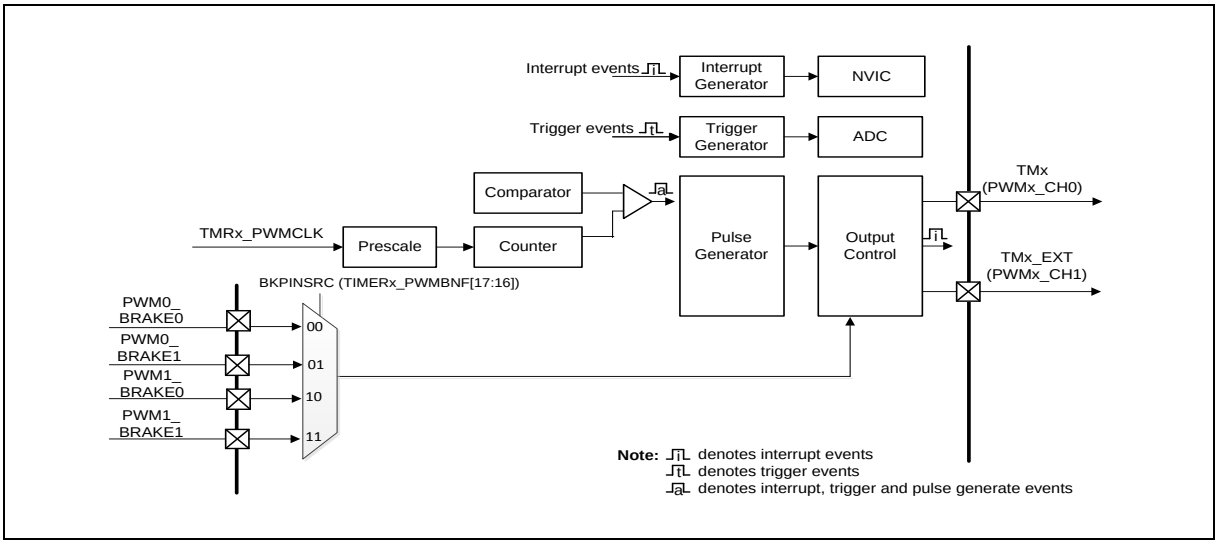


图 6.8-7 PWM 互补模式结构图

6.8.4 基本配置

设置FUNMODE (TIMERx_ALTCTL[0])为0使能定时器模式。在定时器模式，定时器0~定时器3的时钟源可以在寄存器TMRxCKEN (CLK_APBCLK0[5:2])使能，在寄存器TMR0SEL (CLK_CLKSEL1[10:8]) (定时器0)，TMR1SEL (CLK_CLKSEL1[14:12]) (定时器1)，TMR2SEL (CLK_CLKSEL1[18:16]) (定时器2)，TMR3SEL (CLK_CLKSEL1[22:20]) (定时器3)选择不同时钟频率。

设置 FUNMODE (TIMERx_ALTCTL[0])为1使能PWM 模式。在PWM模式，定时器0~定时器3的时钟源可以在寄存器TMRxCKEN (CLK_APBCLK0[5:2])使能，TMR0_CLK和TMR1_CLK时钟源固定为PCLK0，TMR2_CLK 和 TMR3_CLK时钟源固定为PCLK1。

6.8.4.1 TIMER01基本配置

- 时钟源配置
  - 在TMR0CKEN (CLK_APBCLK0[2])中使能TIMER0外设时钟
  - 在TMR1CKEN (CLK_APBCLK0[3])中使能TIMER1外设时钟
- 复位配置
  - 在TMR0RST (SYS_IPRST1[2])中复位TIMER0控制器
  - 在TMR1RST (SYS_IPRST1[3])中复位TIMER1控制器
- 管脚配置

组	管脚名称	GPIO	MFP
TM0	TM0	PG.2	MFP13
		PB.5, PC.7	MFP14
	TM0_EXT	PA.11, PB.15, PH.0	MFP13
TM1	TM1	PC.14, PG.3	MFP13
		PB.4, PC.6	MFP14

	TM1_EXT	PA.10, PB.14, PH.1	MFP13

表 6.8-1TIMER01 管脚配置

6.8.4.2 TIMER23基本配置

- 时钟源配置
  - 在TMR2CKEN (CLK_APBCLK0[4])中使能TIMER2外设时钟
  - 在TMR2CKEN (CLK_APBCLK0[5])中使能TIMER3外设时钟
- 复位配置
  - 在TMR2RST (SYS_IPRST1[4])中复位TIMER2控制器
  - 在TMR3RST (SYS_IPRST1[5])中复位TIMER3控制器
- 管脚配置

组	管脚名称	GPIO	MFP
TM2	TM2	PG.4	MFP13
		PA.7, PB.3, PD.0	MFP14
	TM2_EXT	PA.9, PB.13, PH.2	MFP13
TM3	TM3	PF.11	MFP13
		PA.6, PB.2	MFP14
	TM3_EXT	PA.8, PB.12, PH.3	MFP13

表 6.8-2TIMER23 管脚配置

6.8.5 定时器功能描述

6.8.5.1 定时器中断标志

定时器控制器支持下列中断标志：TIF(TIMERx_INTSTS[0])标志，该标志在当定时器计数器值CNT (TIMERx_CNT[23:0]) 与定时器比较值CMPDAT (TIMERx_CMP[23:0]) 相匹配时置位； CAPIF (TIMERx_EINTSTS[0])标志，该标志在当TMx _EXT管脚、ACMP0_O、ACMP1_O或内部时钟源(HXT、LXT、HIRC、LIRC)的变化与CAPEDGE (TIMERx_EXTCTL[14:12])的设置一致时置位。TWKF (TIMERx_INTSTS[1])位表示中断唤醒标志。

6.8.5.2 定时器计数模式

定时器控制器提供四种定时器计数模式：单次、周期、触发输出和连续计数模式

6.8.5.3 单次模式

如果定时器工作在单周期 (one-shot) 模式(TIMERx_CTL[28:27]为00)，且CNTEN (TIMERx_CTL[30])置1)，则定时器的计数器开始计数。一旦CNT (TIMERx_CNT[23:0])计数器的值达到CMPDAT (TIMERx_CMP[23:0])的值时，TIF (TIMERx_INTSTS[0])标志将变为1，CNT的值和 CNTEN位将由定时器控制器自动清零，然后定时器计数操作停止。与此同时，如果INTEN (TIMERx_CTL[29])位使能，则定时器中断信号产生并送到 NVIC通知CPU。

通过RSTACT (TIMERx_CNT[31])，用户可以监视计数器复位操作是否生效。设置ICEDEBUG

(TIMERx_CTL[31]) 为 1 禁止 ICE 调试模式对定时器计数的影响。

6.8.5.4 周期模式

如果定时器工作在周期 (periodic) 模式(TIMERx_CTL[28:27]为01)且CNTEN (TIMERx_CTL[30])置1，则定时器的计数器开始向上计数。一旦CNT (TIMERx_CNT[23:0]) 计数器的值达到CMPDAT (TIMERx_CMP[23:0])的值时，TIF (TIMERx_INTSTS[0])标志将变为1，CNT的值将由定时器控制器自动清零，然后定时器重新计数。与此同时，如果INTEN (TIMERx_CTL[29])使能，则定时器中断信号产生并送到 NVIC 通知 CPU 。在该模式，定时器控制器周期性地操作计数和与CMPDAT的值比较，直到CNTEN位由软件清0。

可以通过设置PERIOSEL (TIMERx_CTL[20])位选择在周期模式下的定时器操作。

6.8.5.5 触发输出模式

如果定时器工作在触发输出模式(TIMERx_CTL[28:27]为10)且CNTEN (TIMERx_CTL[30])位置1，则定时器的计数器开始计数。触发输出模式的计数操做大部分与周期模式是一样的，除了该模式当TIF (TIMERx_INTSTS[0])位设置时，有相关的TM0 ~ TM3或TM0_EXT ~ TM3_EXT管脚会输出信号。用户可以设置TGLPINSEL (TIMERx_CTL[21])选择Tx或Tx_EXT作为触发输出管脚，触发输出信号的周期是50%。

6.8.5.6 连续计数模式

如果定时器工作在连续计数 (continuous counting) 模式(TIMERx_CTL[28:27] 为 11) 且 CNTEN (TIMERx_CTL[30])位置1，则定时器的计数器开始计数。一旦CNT (TIMERx_CNT[23:0])的值达到CMPDAT (TIMERx_CMP[23:0])的值时，TIF (TIMERx_INTSTS[0])标志将变为1，但CNT的值继续保持向上计数。与此同时，如果INTEN (TIMERx_CTL[29])使能，则定时器中断信号产生并送到 NVIC 通知 CPU 。在该模式，用户可以立刻改变不同的CMPDAT值，而不需要停止定时器计数和重新开始定时器计数。

例如，CMPDAT的值设置为 80。当CNT 达到 80时，TIF标志将被置1，定时器计数器继续计数，而且CNT的值将不回到0，而是继续计数81、82、83…到 (2^{24} -1)，然后再一次 0、1、2、3…到 2^{24} -1，如此往复。接下来，如果软件改变CMPDAT的值为200并且清除TIF标志位，当CNT的值达到200时，TIF标志将再次变为1,。最后，软件改变CMPDAT的值为500并且清除TIF标志，当CNT的值达到500时，TIF标志将再次变为1。

在该模式，计数器计数是连续的。所以该操作模式叫做连续计数模式。

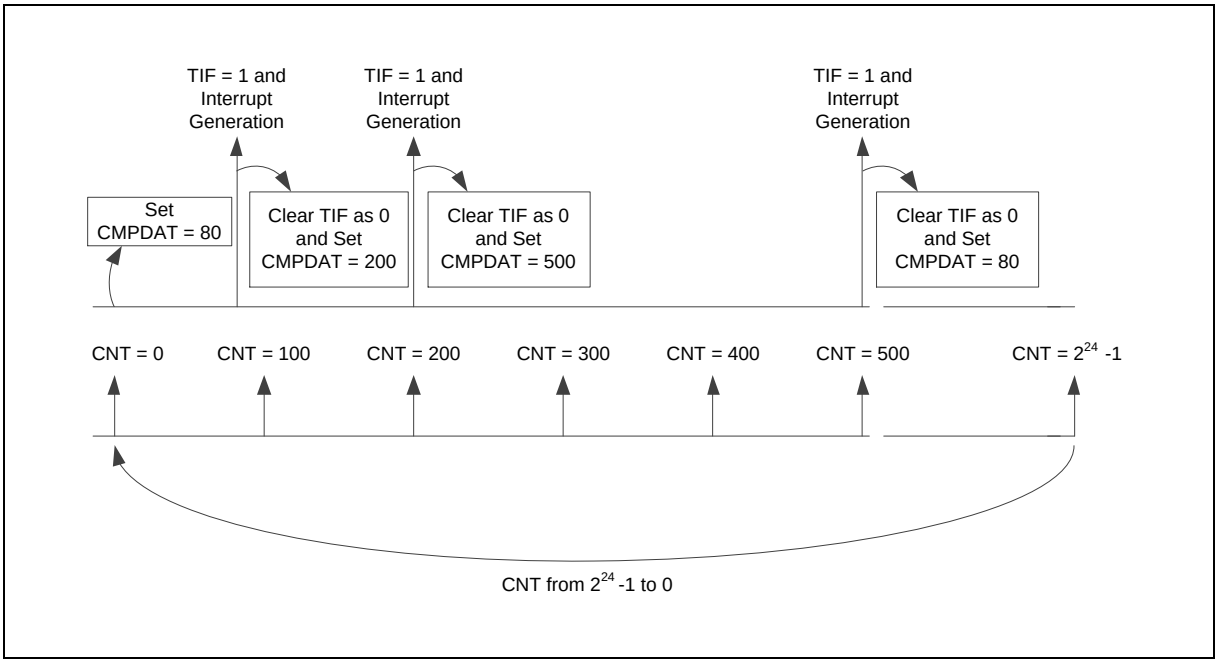


图 6.8-8 连续计数模式

6.8.5.7 事件计数模式

定时器控制器也提供这样的应用，能对输入事件(来自管脚TMx (x= 0~3))计数并将事件的次数反映到CNT (TIMERx_CNT[23:0]) 的值。也可以称为事件计数功能。该功能下，EXTCNTEN (TIMERx_CTL[24])位需置位并且定时器外设时钟源必须设为PCLK。

如果ECNTSSEL (TIMERx_EXTCTL[16]) 是 0，事件计数源来自TMx管脚。软件可以通过CNTDBEN (TIMERx_EXTCTL[7])位来使能或关闭TMx管脚消抖电路。如果TMx管脚的消抖电路关闭，输入事件频率必须少于1/3 PCLKxHCLK，如果消抖电路打开，输入事件的频率须小于1/8 PCLKxHCLK，以保证CNT的值是正确的。软件也可以通过设置CNTPHASE (TIMERx_EXTCTL[0])来选择TMx管脚边沿检测的相位。

事件计数模式下，定时器计数操作模式可以设置为单次，周期，和连续计数模式来计算来自TMx管脚的输入事件CNT (TIMERx_CNT[23:0])的值。

如果ECNTSSEL (TIMERx_EXTCTL[16]) 是 1，事件计数源由USB设备侦测帧起始包(SOF)产生。请参考USB设备规格。

6.8.5.8 外部捕获模式

事件捕获功能是当检测到捕获源边沿电平有变化时，CNT (TIMERx_CNT[23:0])值会加载到CAPDAT (TIMERx_CAP[23:0])。在该模式下，需把CAPEN (TIMERx_EXTCTL[3])设置为1，CAPFUNCS (TIMERx_EXTCTL[4])位设置为0，用来选择TMx_EXT变化时用作事件捕获功能，而且定时器外设时钟源必须设为PCLK。

如果CAPSRC (TIMERx_CTL[22]) 是 0，捕获事件由TMx_EXT管脚跳变触发。软件可以通过CAPDBEN (TIMERx_EXTCTL[6])位来使能或关闭TMx_EXT管脚消抖电路。在TMx_EXT的消抖电路关闭时，TMx_EXT管脚的转变频率必须少于1/3 PCLKxHCLK，在TMx_EXT的消抖电路打开时，TMx_EXT管脚的转变频率必须少于1/8 PCLKxHCLK，以保证捕获功能能够正常工作。软件也可以通过设置CAPEDGE (TIMERx_EXTCTL[14:12])位来选择TMx_EXT管脚的边沿转变检测方式。

如果CAPSRC (TIMERx_CTL[22])是1，捕获事件可以被在ACMP0（如果INTERCAPSEL(TIMERx_EXTCTL[10:8])为 000）或ACMP1（如果INTERCAPSEL为001）上的内部输出跳转信号触发。如果INTERCAPSEL为010, 011, 100, 101，则捕获源分别是HXT、LXT、 HIRC、LIRC。用户只能当原始捕获源跟目标捕获源都关掉的时候才能切换捕获源。例如：如果用户想要切换LXT或LIRC，两个捕获源都要先关闭。

在事件捕获模式，用户不用考虑选择什么定时器计数模式，捕获事件只有在捕获源上检测到边沿跳转发生。

用户可以使能CAPIEN (TIMERx_EXTCTL[5])来使用捕获中断功能，当捕获源边沿跳转与设定的一样，CAPIF为高。

用户必须考虑到，如果CPU不去清CAPIF状态，定时器将保持寄存器TIMERx_CAP不改变且丢失新的捕获值。

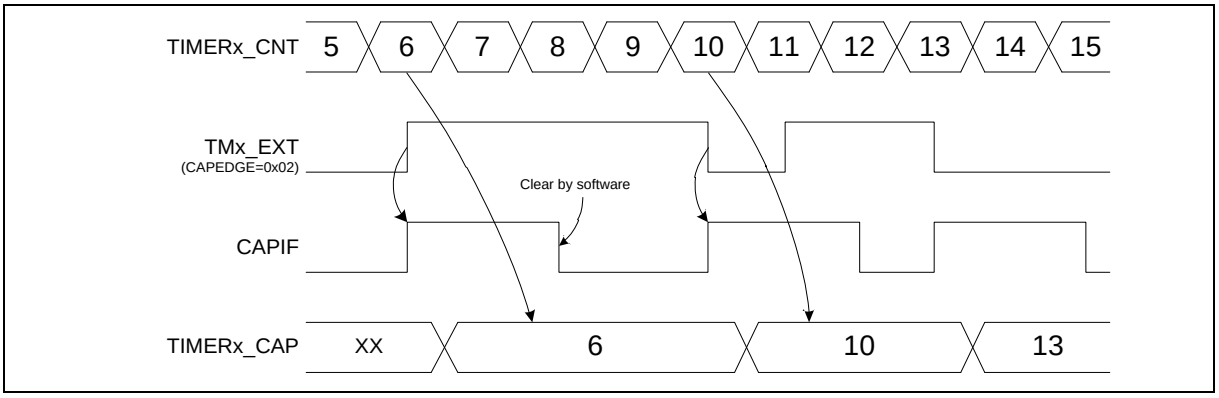


图 6.8-9TMx_EXT 引脚外部捕获模式

6.8.5.9 外部复位计数器模式

当捕获事件产生，定时器同样提供捕获事件复位计数器功能来复位CNT (TIMERx_CNT[23:0])的值。在该模式， CAPFUNCS (TIMERx_EXTCTL[4])位必须设置为1来选择TMx_EXT转变、内部ACMPx 输出信号或内部时钟源 (HXT, LXT, HIRC, LIRC)触发复位计数器值。

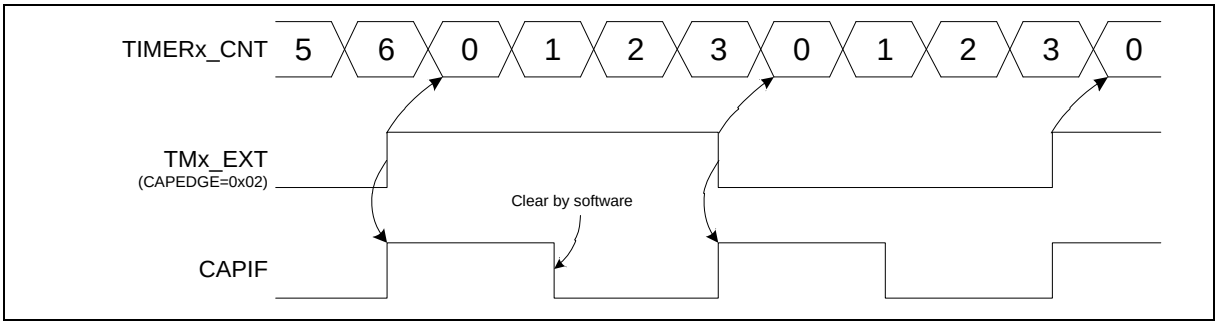


图 6.8-10 外部复位计数器模式

6.8.5.10 定时器触发功能

定时器控制器提供定时器超时溢出中断或捕获中断来触发EPWM、ADC、DAC和PDMA。如果TRGSSEL (TIMERx_TRGCTL[0])为0, 超时溢出中断信号用于触发BPWM、EPWM、EADC、DAC和PDMA。如果TRGSSEL (TIMERx_TRGCTL[0])为1, 捕获中断信号用于触发PWM、ADC、DAC和PDMA。

当TRGPWM (TIMERx_TRGCTL[1])被置1，如果定时中断信号产生，定时器控制器将产生一个触发脉冲作为EPWM和BPWM外部时钟源。

当TRGADC (TIMERx_TRGCTL[2])被置1，如果定时中断信号产生，定时器控制器将触发ADC开始转换。

当TRGDAC (TIMERx_TRGCTL[3])被置1，如果定时中断信号产生，定时器控制器将触发DAC开始转换。

当 TRGPDMA (TIMERx_TRGCTL[4]) 被置1，如果定时中断信号产生，定时器控制器将触发PDMA。

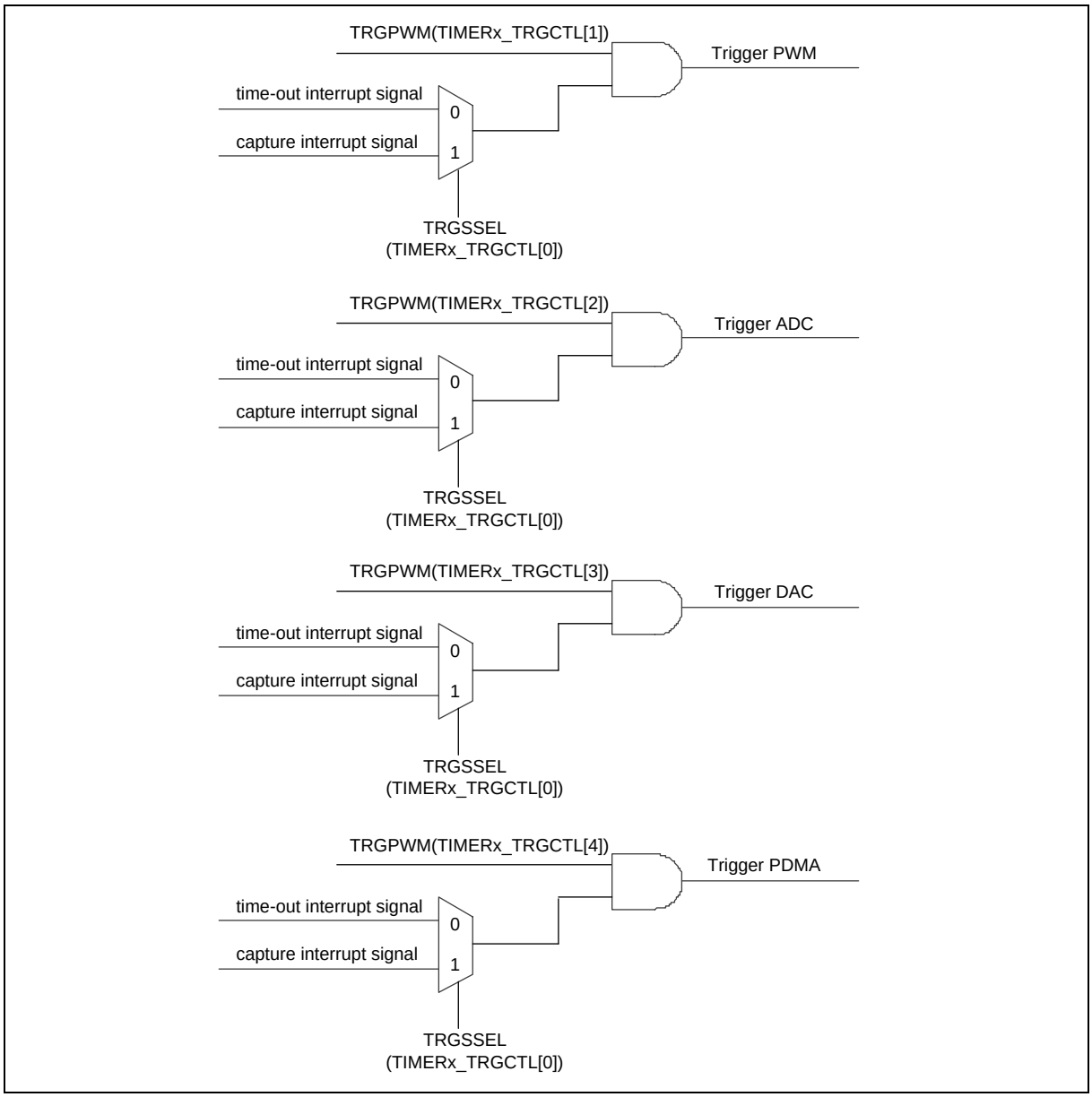


图 6.8-11 内部定时器触发

6.8.5.11 定时器间互触发捕获模式

在该模式下，Timer0/2将会被强制工作在计数器模式，每有一个外部事件计数一次，并会在特定的时刻

产生相应的信号 (INTR_TMR_TRG) 触发Timer1/3开始或停止计数。而Timer1/3被强制工作在捕获模式，由Timer0/2计数器状态触发开始或停止计数。

设置 Timer0 定时器间互触发捕获使能，触发计数捕获功能强制在Timer1。设置 Timer2 定时器间互触发捕获使能，触发计数捕获功能强制在Timer3。

开始触发

Timer0/2的INTRGEN (TIMERx_CTL[19])被置位，当Timer0/2计数器的值从 0x0 变为 0x1 时，Timer0/2产生一个上升沿触发信号。Timer1/3计数器则自动立即开始计数。

停止触发

当 Timer0/2 CNT 达到 Timer0/2 CMPDAT 值， Timer0/2 会产生一个下降跳变信号。Timer0/2 计数器模式功能被禁止，INTRGEN (TIMERx_CTL[19])被硬件清0。然后Timer1/3 停止计数，同时Timer1/3 CNT的值保存到Timer1/3 CAPDAT (TIMERx_CAP[23:0])。

用户使用定时器间互触发模式测量外部事件(TMx)的周期会更精确图6.8-12是定时器间互触发模式，定时器0用作事件计数，定时器1用作触发计数捕获模式的例子。

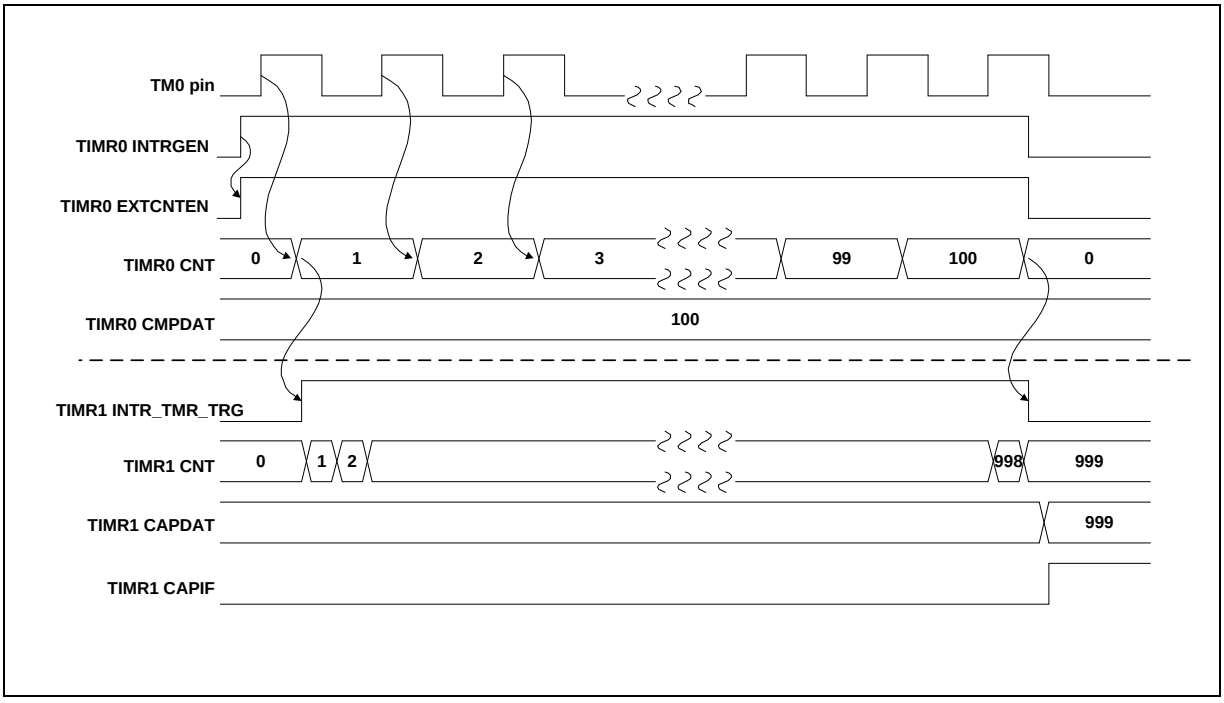


图 6.8-12 定时器间互触发捕获时序

6.8.6 PWM功能描述

6.8.6.1 PWM预分频

PWM 预分频用于分频时钟源， PWM 时钟被 (CLKPSC+ 1) 分频。通过 CLKPSC (TIMERx_PWMCLKPSC[11:0])设置分频值。图6.8-13是在上数计数模式PWM预分频的波形。

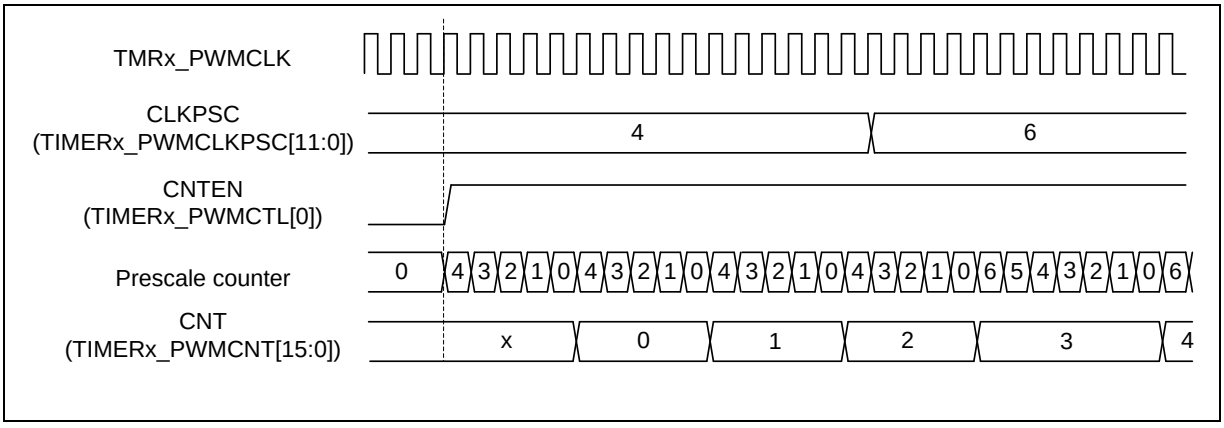


图 6.8-13 PWM 上数计数模式的预分频波形

6.8.6.2 PWM计数器

PWM 支持三种计数器操作模式，上数，下数和上下数计数模式。

6.8.6.3 上计数模式

当PWM设置为上计数模式，CNTTYPE (TIMERx_PWMCTL[2:1])为0x0，计数器开始从0到PERIOD (TIMERx_PWMPERIOD[15:0])向上计数。当前计数值可以从CNT (TIMERx_PWMCNT[15:0])读出。当计数器和预分频器计数到0将产生零点事件，当计数到PERIOD且预分频器计数到0将产生周期点事件。下图是一个向上计数器例子，PWM周期时间= (PERIOD+1) * (CLKPSC+1) * TMRx_PWMCLK。

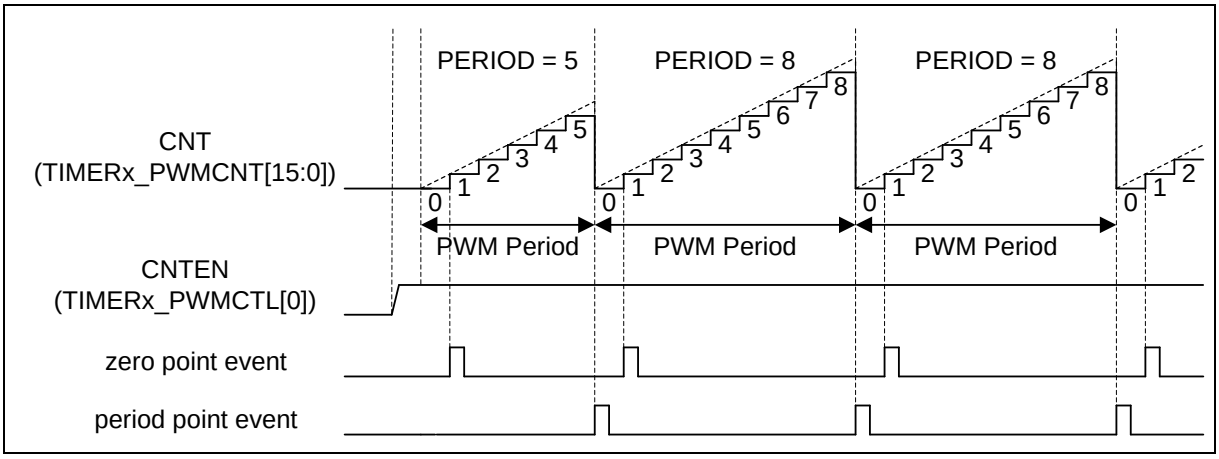


图 6.8-14 PWM 上计数模式

6.8.6.4 下计数模式

当PWM设置为下计数模式，CNTTYPE (TIMERx_PWMCTL[2:1])为0x1，计数器开始从PERIOD到0向下计数。当前计数值可以从CNT (TIMERx_PWMCNT[15:0])读出。当计数器和预分频器计数到0将产生零点事件，当计数到PERIOD且预分频器计数到0将产生周期点事件。下图是一个向下计数器例子，PWM周期时间= (PERIOD+1) * (CLKPSC+1) * TMRx_PWMCLK。

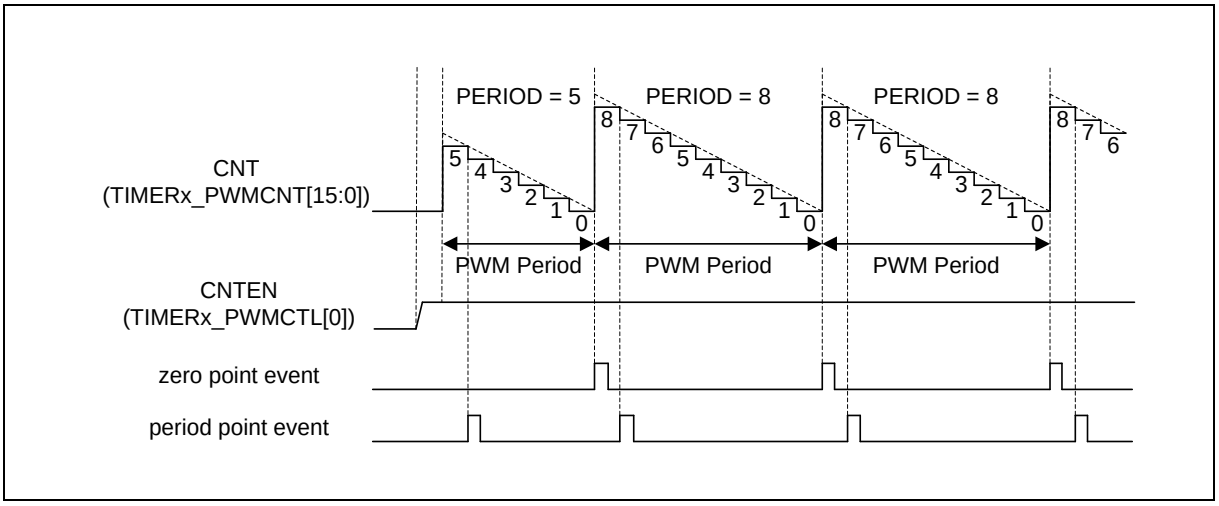
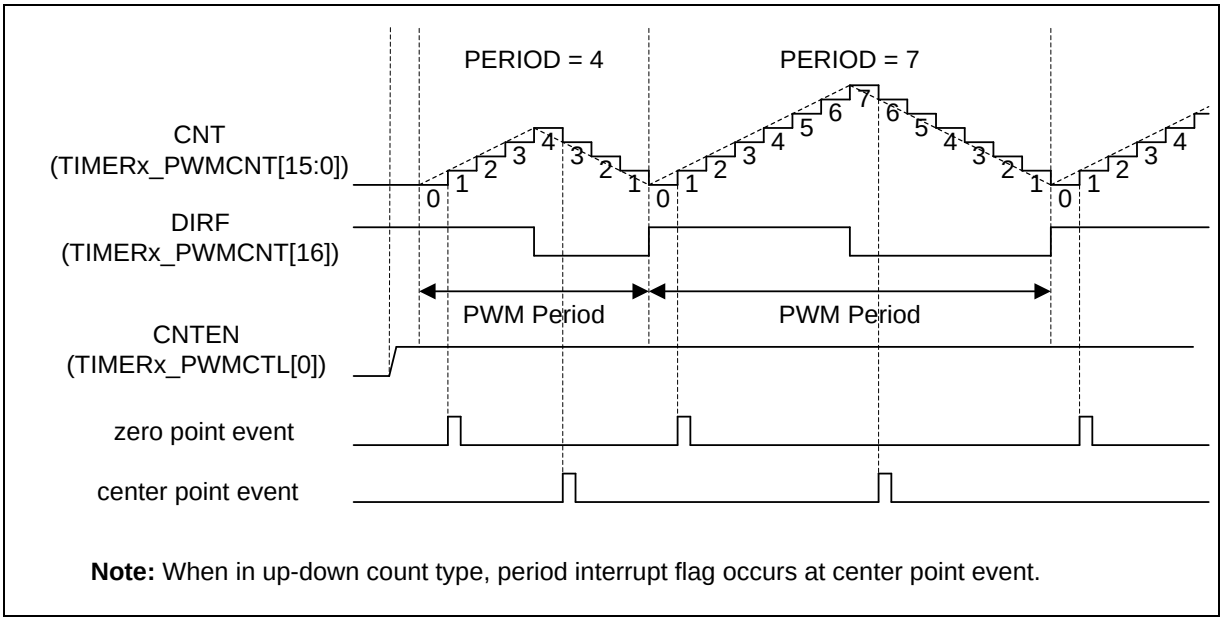


图 6.8-15 下计数模式

6.8.6.5 *上下计数模式*

当PWM设置为上下计数模式，CNTTYPE (TIMERx_PWMCTL[2:1])为0x2，计数器开始从0到PERIOD向上计数然后再向下计数到0。当前计数值可以从CNT (TIMERx_PWMCNT[15:0])读出。当计数器和预分频器计数到0将产生零点事件，当计数到PERIOD且预分频器计数到0将产生中心点事件。下图是一个上下计数器例子，PWM周期时间= (2 * PERIOD) * (CLKPSC+1) * TMRx_PWMCLK。DIRF (TIMERx_PWMCNT[16])是计数器的方向指示标志，该标志为1表示向上计数，为0表示向下计数。



Note: When in up-down count type, period interrupt flag occurs at center point event.

图 6.8-16 PWM 上下计数模式

6.8.6.6 *PWM计数器操作模式*

PWM计数器支持两种操作模式：单次模式和自动重载模式。如果CNTMODE (TIMERx_PWMCTL[3])=1，计数器工作在单次模式，如果CNTMODE (TIMERx_PWMCTL[3])=0计数器工作在自动重载模式。

在这两种模式中，CMP (TIMERx_PWMCMPDAT[15:0]) 和 PERIOD (TIMERx_PWMPERIOD[15:0]) 应先写入，然后设置CNTEN (TIMERx_PWMCTL[0]) 为1开始计数器工作。

在单次模式，PWM计数器值会在PWM周期完成后，根据计数模式重载为默认值。用户可以写CMP来继续单次操作产生下一个单次脉冲，无论当前单次计数器是正在运行还是已完成。

在自动重载模式，PWM计数器是连续运行，如果在自动重载模式用户设置PERIOD为0在一个PWM周期完成后，PWM计数器值会根据计数模式重载为默认值。

6.8.6.7 PWM比较器

CMP (TIMERx_PWMCMPDAT[15:0])是PWM的比较寄存器，CMP的值会连续不断地和对应的计数器值比较。当计数器等于CMP，PWM产生一个比较点事件。该事件产生PWM输出脉冲，中断信号或触发ADC开始转换。在上下数模式，一个PWM周期会产生两个事件。如下图所示，CMPU是上数比较点事件，CMPD是下数比较点事件。

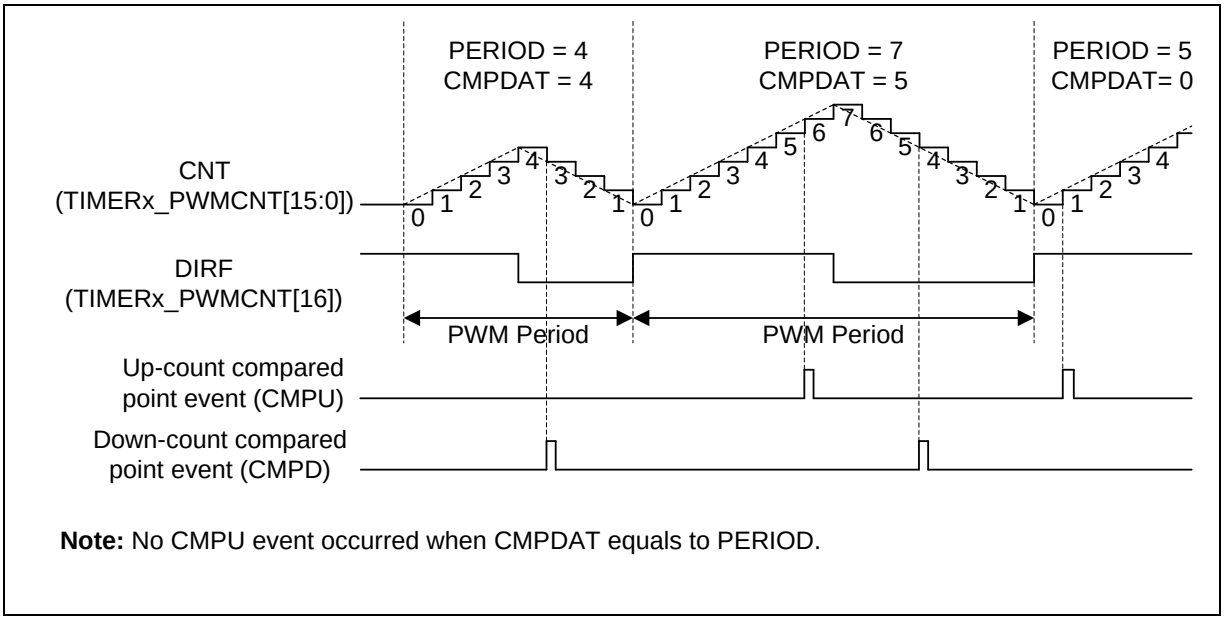


图 6.8-17 上下数模式 PWM 的比较器事件

6.8.6.8 周期装载模式

当 IMMLDEN (TIMERx_PWMCTL[9]) 位置 0，PWM 工作在周期装载模式。PBUF (TIMERx_PWMPBUF[15:0]) 是PERIOD 的缓存寄存器。CMPBUF (TIMERx_PWMCMPBUF[15:0])是CMP 的缓存寄存器。在周期装载模式，当每一个 PWM 周期完成。PERIOD (TIMERx_PWMPERIOD[15:0]) 和 CMP (TIMERx_PWMCMPDAT[15:0])。下图为上数计数模式周期装载时序。这里的PERIOD DATA0表示PERIOD的初始值，PERIOD DATA1表示PERIOD数据第一次更新的值，依次类推，CMP也遵循这个规则。以下步骤是图6. 8-18的时序。

1. 在点1，用户写CMP DATA1 到 CMP；
2. 在点2，PWM周期结束的时候，周期装载CMP DATA1 到 CMPBUF；
3. 在点3，写PERIOD DATA1 到 PERIOD；
4. 在点4，PWM周期结束的时候，周期装载PERIOD DATA1 到PBUF；
5. 在点5，写 PERIOD DATA2 到 PERIOD；

6. 在点6，PWM周期结束的时候，周期装载PERIOD DATA2 到PBUF。

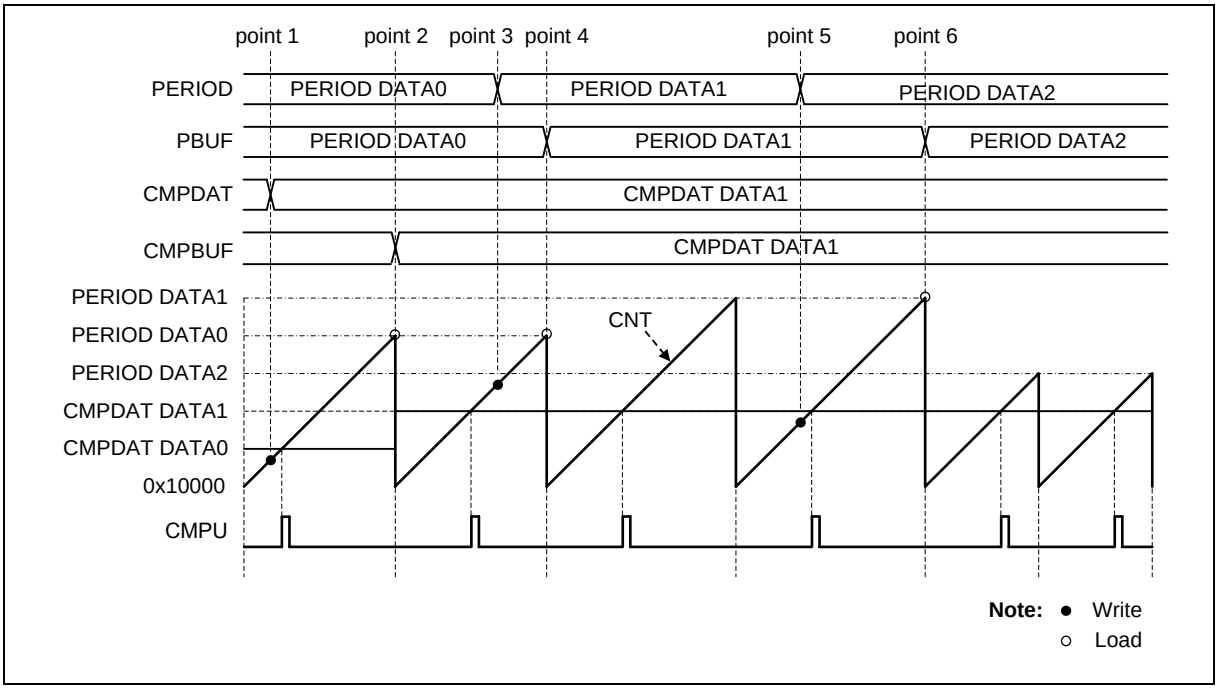


图 6.8-18 上数计数模式周期装载模式

6.8.6.9 立即加载模式

当 IMMLDEN (TIMERx_PWMCTL[9]) 位置 1，PWM工作在立即装载模式。当用户更新PERIOD (TIMERx_PWMPERIOD[15:0]) 或 CMP (TIMERx_PWMCMPDAT[15:0])，PERIOD 或 CMP 会在当前计数器计数完成后被装载到 PBUF (TIMERx_PWMPBUF[15:0]) 或 CMPBUF (TIMERx_PWMCMPBUF[15:0])。如果更新PERIOD值小于当前计数器的值，计数器会循环计数。以下步骤是图6. 8-19的时序。

1. 用户在点1写 CMP DATA1 且硬件会在当前计数器计数完成后装载CMP DATA1 到 CMPBUF；
2. 用户在点2写PERIOD DATA1 且 PERIOD DATA1 大于当前计数器的值，PWM计数器会继续计数直到等于PERIOD DATA1 以完成一个PWM周期；
3. 用户在点3写PERIOD DATA2 且 PERIOD DATA2小于当前计数器值，PWM计数器继续计数到最大计数器值0x1FFFF，且从0x10000 到 PERIOD DATA2 循环完成一个PWM周期。

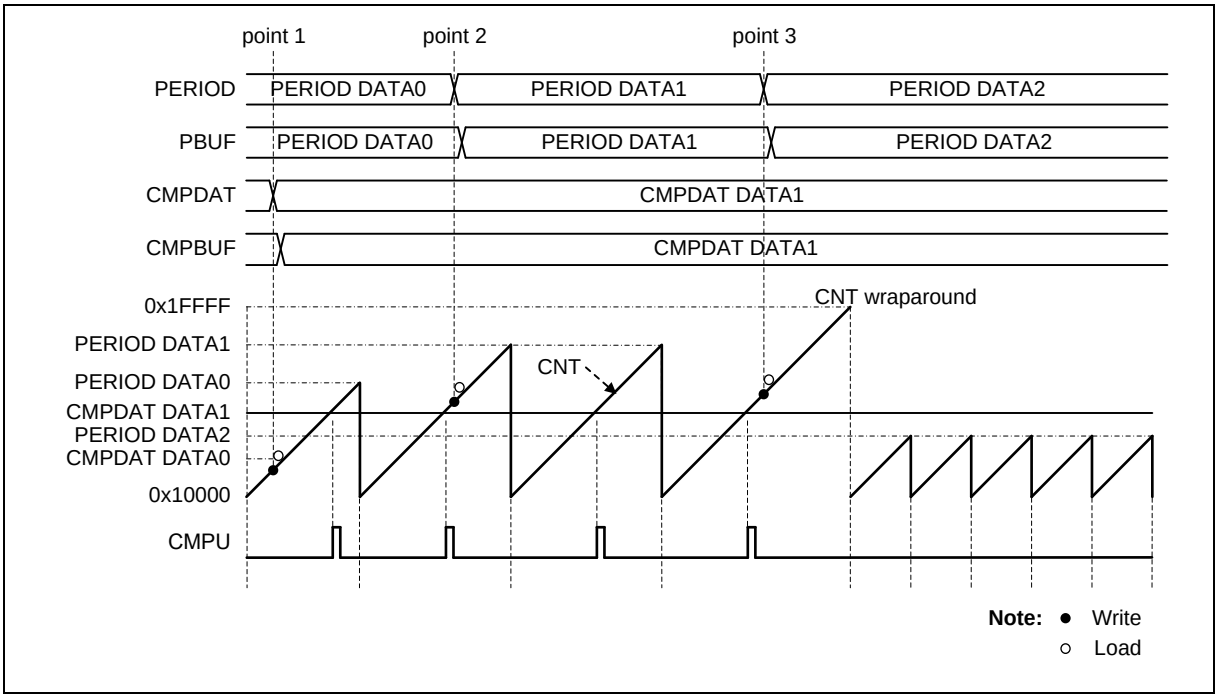


图 6.8-19 上计数模式立即装载模式

6.8.6.10 PWM脉冲产生器

PWM 脉冲产生器用计数器和比较器事件产生PWM输出脉冲。在上数或下数计数模式，事件有零点事件、周期点事件。在上下数计数模式有中心点事件，在三种计数模式下，都有计数器等于比较器点事件。

每一个事件点可以产生PWM输出波形，图6.8-20、图6.8-21和图6.8-22表示不同的计数模式。

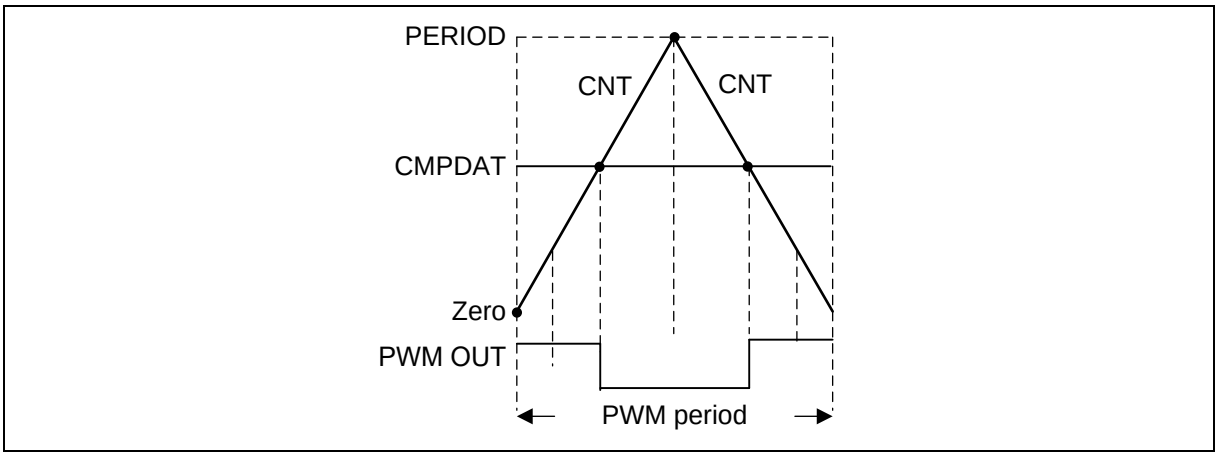


图 6.8-20 在上下数模式 PWM 脉冲的产生

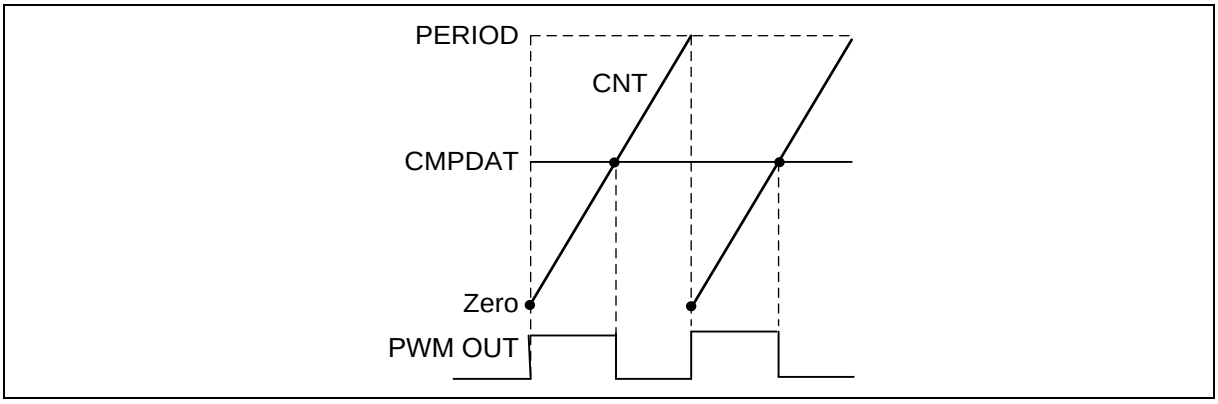


图 6.8-21 在上数模式 PWM 脉冲的产生

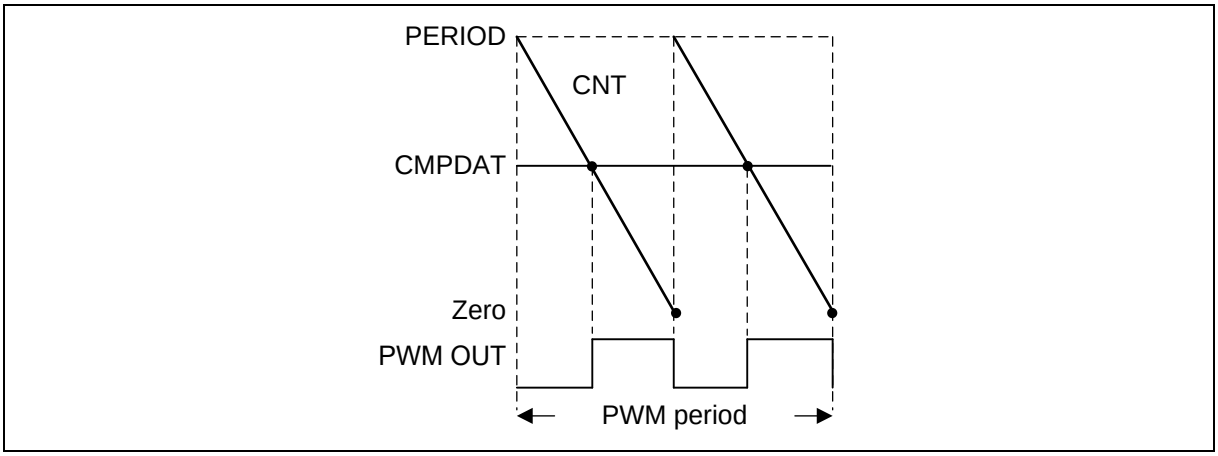


图 6.8-22 在下数模式 PWM 脉冲的产生

PWM产生的事件有时候会在同一时间产生，因此不同计数器类型的事件优先级必须注意到，如表6.8-4、表6.8-4和表6.8-5所示，包含向上计数类型事件优先级、向下计数类型事件优先级和上下计数类型事件优先级。

优先级	0点和 CMPU 点事件 (CMP = 0)	PWM 输出
1 (高)	向上比较事件	低
2 (低)	0点事件	高

表 6.8-3 上数计数模式中 PWM 脉冲产生事件优先级

优先级	0点和 CMPD点事件 (CMP = 0)	PWM 输出
1 (高)	0点事件	低
2 (低)	向下比较事件	高

优先级	周期点和 CMPD 点事件 (CMP = PERIOD)	PWM 输出
1 (高)	向下比较事件	高
2 (低)	周期事件	低

表 6.8-4 下数计数模式中 PWM 脉冲产生事件优先级

优先级	CMPU 和 CMPU 点事件 (CMP = PERIOD)	PWM 输出
1 (高)	向下比较事件	高
2 (低)	向上比较事件	低

表 6.8-5 上下数计数模式中 PWM 脉冲产生事件优先级

根据事件优先级限制，仅在上数和上下数计数模式才支持0%和100%占空比周期的PWM输出。下图是在上数模式和上下数模式PWM占空比周期从0% 到 100%的例子。这里的PERIOD是4，CMP的值不同。

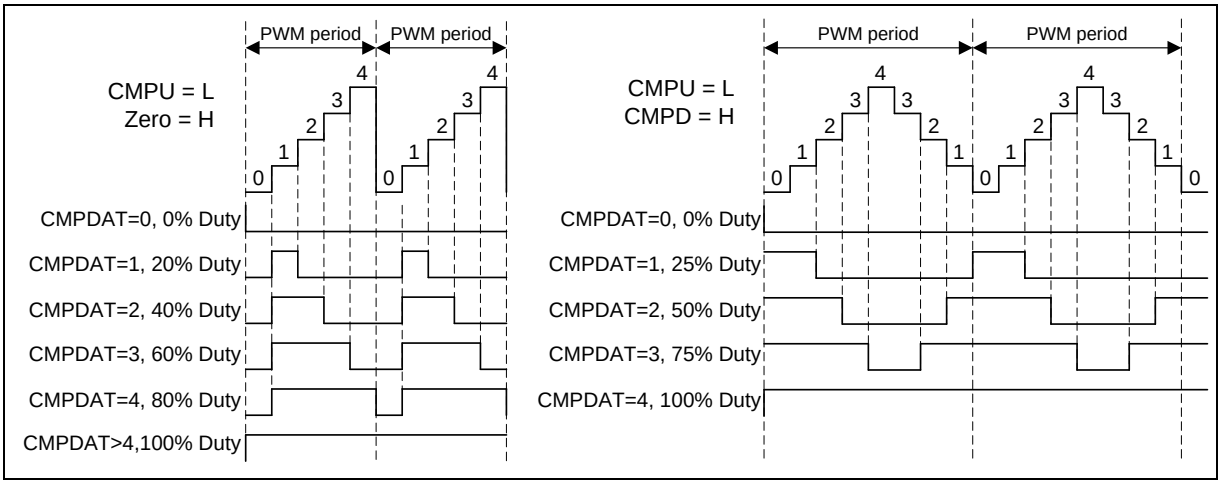


图 6.8-23 上数模式和上下数模式 PWM 占空比周期 0%到 100%的例子

6.8.6.11 PWM输出模式

PWM支持两种输出模式，独立模式可以用于直流电机系统；互补模式带死区插入的功能，可以用于交流感应电机和永磁同步电机。

6.8.6.12 独立模式

当 OUTMODE (TIMERx_PWMCTL[16]) 位置0 ，PWM输出工作在独立模式。在该模式下，PWMx_CH0 和 PWMx_CH1可以输出相同的波形，如下图：

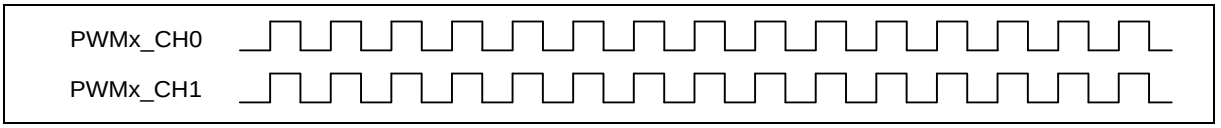


图 6.8-24 PWM 独立模式输出波形

6.8.6.13 互补模式

当 OUTMODE (TIMERx_PWMCTL[16]) 位置1，PWM输出工作在互补模式。在该模式下，PWMx_CH0 和 PWMx_CH1必须总是输出互补的波形，如下图：

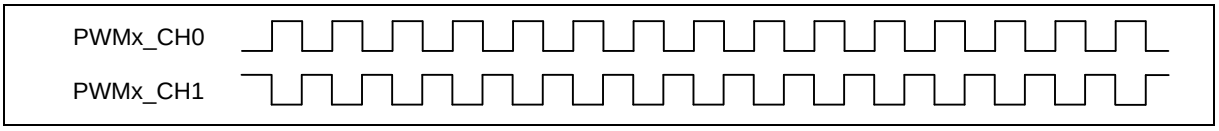


图 6.8-25 PWM 互补模式输出波形

6.8.6.14 PWM输出控制

PWM脉冲产生后，在独立模式有四个步骤控制输出波形，在互补模式有五步控制输出波形，用户可以设置POEN0 (TIMERx_PWMPOEN[0]) 和 POEN1 (TIMERx_PWMPOEN[1]) 为1使能PWMx_CH0 和 PWMx_CH1输出波形。

独立模式中，有屏蔽、刹车、管脚极性和输出使能控制输出波形，如下图：

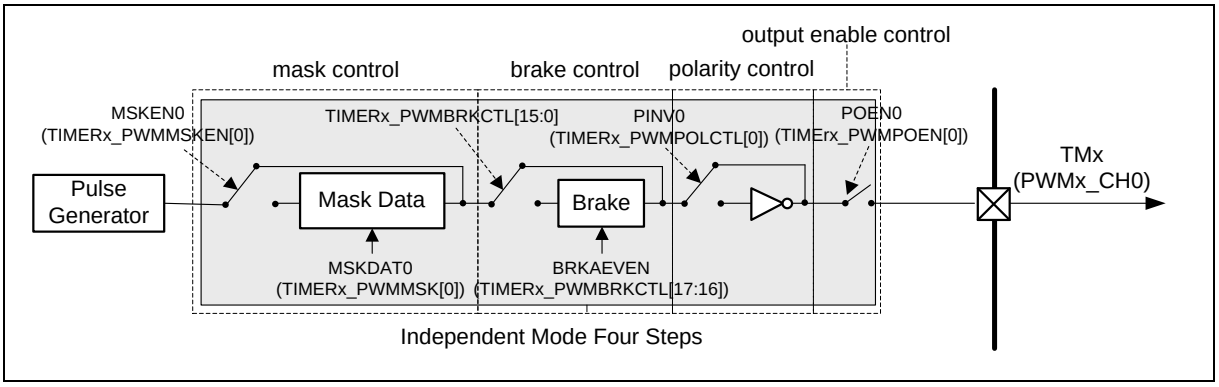


图 6.8-26 独立模式下 PWMx_CH0 输出控制

互补模式中，多一个死区控制，其他四个步骤和独立模式一样。如下图控制PWMx_CH0和PWMx_CH1输出。

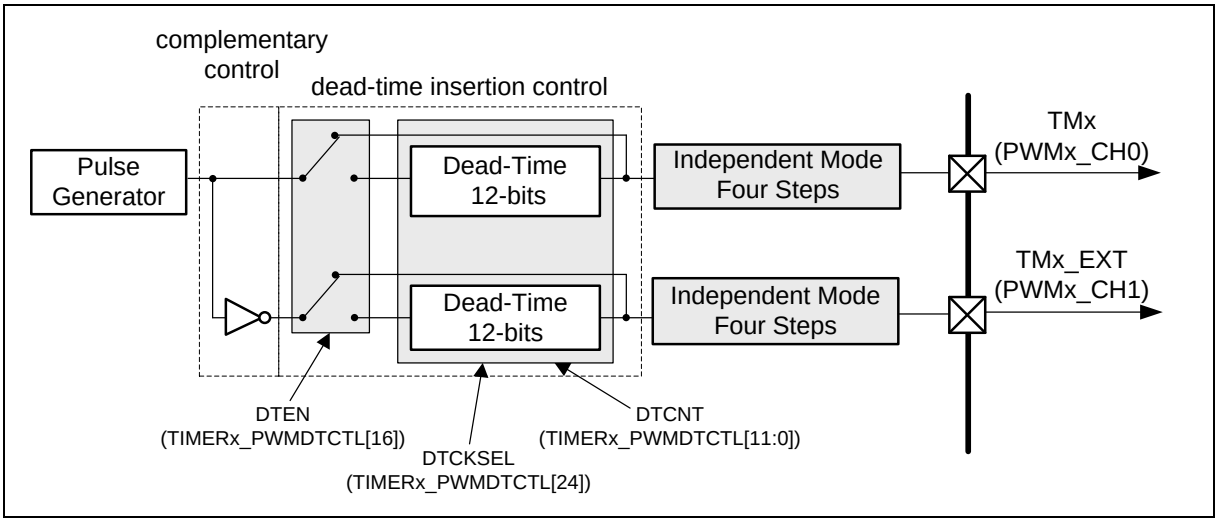


图 6.8-27 互补模式下 PWMx_CH0 和 PWMx_CH1 输出控制

6.8.6.15 死区插入控制

互补应用中，互补通道也许用来驱动外部设备例如开关电源，在互补输出通道间死区发生器插入一个低电平间隔。通过设置 DTEN (TIMERx_PWMDTCTL[16]) 位来使能死区功能， DTCNT (TIMERx_PWMDTCTL[11:0]) 和 DTCKSEL (TIMERx_PWMDTCTL[24])用于控制死区间隔，死区时间可以通过以下公式计算：

死区时间间隔 = (DTCNT + 1) * TMRx_PWMCLK 周期（如果 DTCKSEL 是 0）

死区时间间隔= (DTCNT + 1) * TMRx_PWMCLK * (CLKPSC + 1) 周期（如果 DTCKSEL 是 1）

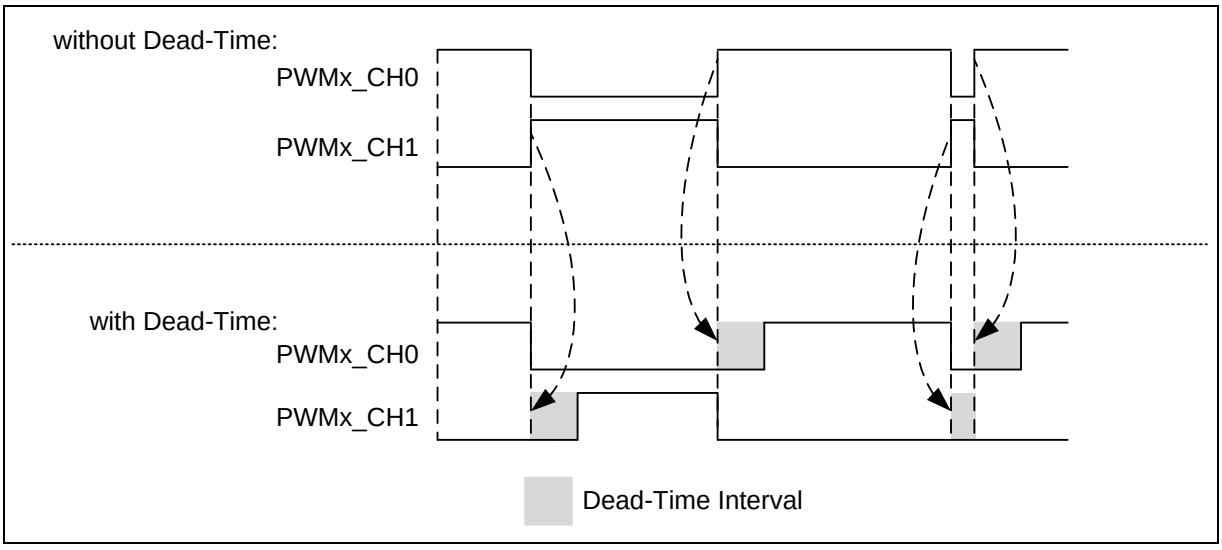


图 6.8-28 死区插入

6.8.6.16 PWM屏蔽输出控制

通过设置 MSKEN0/1 (TIMERx_PWMMSKEN[1:0]) 和 MSKDAT0/1 (TIMERx_PWMMSK[1:0]). PWMx_CH0/CH1 输出值可以被屏蔽到指定的逻辑状态。当控制各种电子换向电机(ECM) 如BLDC，PWM输出屏蔽功能是很有用的。下图是PWMx_CH0 和 PWMx_CH1中PWM屏蔽控制的例子。

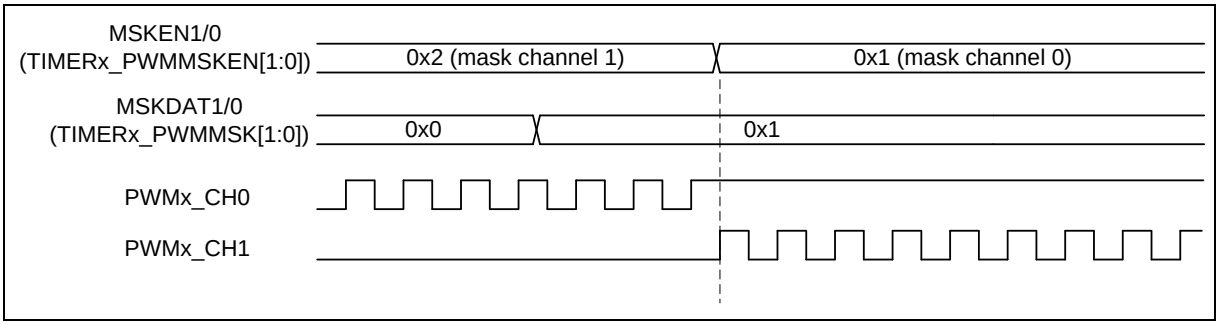


图 6.8-29PWM 输出屏蔽控制波形

6.8.6.17 PWM刹车控制

每一个PWM产生器支持一个外部输入刹车管脚作为PWM刹车事件源。用户可以通过BKPINSRC (TIMERx_PWMBNF[17:16])，PWMx_BRAKEy (x=0,1 ， y=0,1)选择刹车管脚源。有一个3位噪音滤波

计数器来滤波外部刹车管脚的信号。用户可以设置BRKNFEN (TIMERx_PWMBNF[0])位来使能刹车管脚噪音滤波功能且噪音滤波采样时钟可以通过设置BRKNFSEL (TIMERx_PWMBNF[3:1])来选择以适应不同的噪音特性。通过设置BRKFCNT (TIMERx_PWMBNF[6:4])，用户可以定义多少个采样时钟周期来识别有效的刹车信号边沿。此外刹车管脚极性可以通过设置BRKPINV (TIMERx_PWMBNF[7])来反转。设置BRKPINV为0，当PWMx_BRAKEy管脚状态从低到高，刹车事件发生。设置BRKPINV为1，当PWMx_BRAKEy管脚状态从高到低，刹车事件发生。

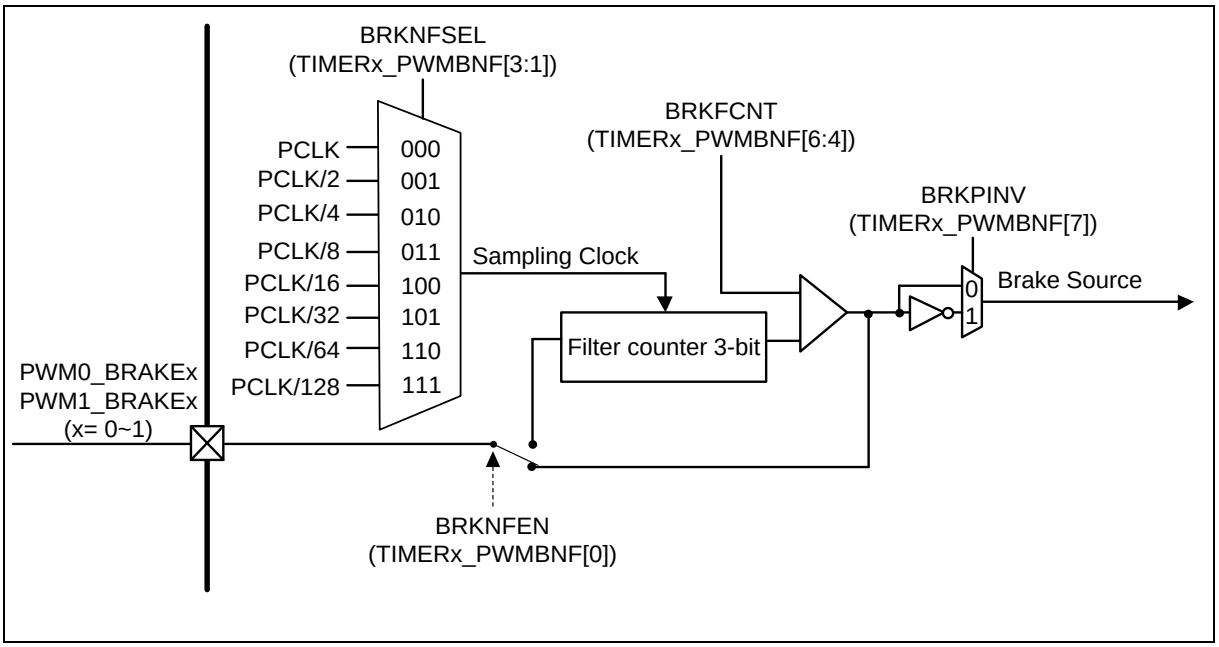


图 6.8-30 刹车管脚噪音滤波器框图

用户可以设置BRKAEVEN (TIMERx_PWMBRKCTL[17:16]) 控制PWMx_CH0刹车事件发生时的输出状态，设置BRKAODD (TIMERx_PWMBRKCTL[19:18])控制PWMx_CH1刹车事件发生时的输出状态。当刹车事件发生，有两个刹车侦测器源，边沿侦测刹车源和电平侦测刹车源。下图为PWMx_CH0 和PWMx_CH1的刹车事件框图。

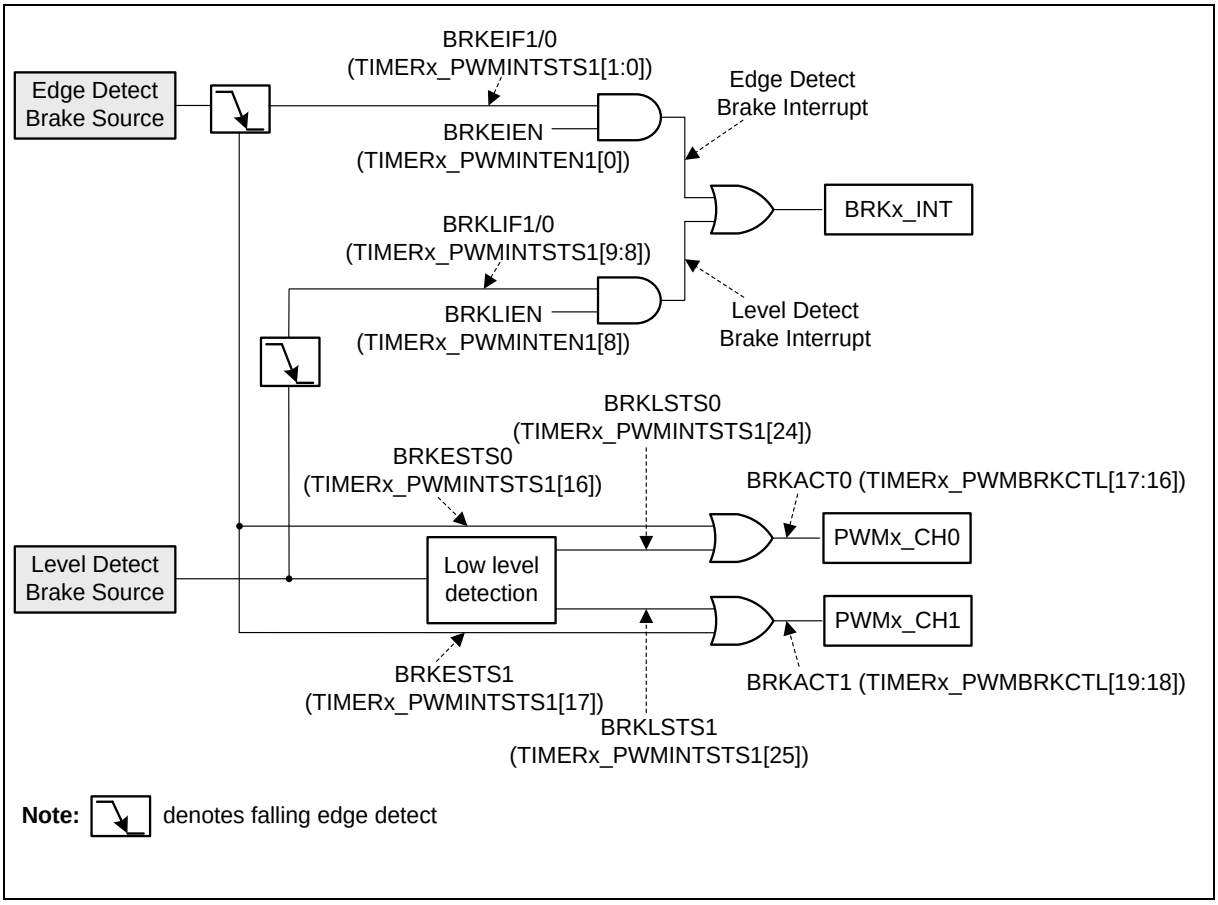


图 6.8-31PWMx_CH0 和 PWMx_CH1 的刹车事件框图

当边沿侦测器侦测到刹车信号，刹车功能产生中断状态（PWMx_CH1/0 是 BRKEIF1/0 (TIMERx_PWMINTSTS1[1:0]) ）和刹车事件状态（PWMx_CH1/0 是 BRKESTS1/0 (TIMERx_PWMINTSTS1[17:16])）。中断状态BRKEIF1/0通过写1清0，刹车事件状态BRKESTS1/0会保持直到当对应的BRKEIF1/0标志已经清除且PWM产生器重新开始正常输出，下一个PWM周期开始。

下图是PWMx_CH0 和 PWMx_CH1边沿侦测器刹车波形的例子。在这个情况中，边沿侦测刹车源发生两次刹车事件。第一次刹车事件发生BRKEIF0 和 BRKEIF1标志都被置位，且BRKESTS0 和 BRKESTS1状态也被置位表示PWMx_CH0 和 PWMx_CH1的刹车状态。对于第一个发生的事件，用户写1清除BRKEIF0，之后，当下一个PWM周期开始BRKESTS0由硬件清0，PWMx_CH0输出正常的波形，即使边沿刹车事件仍然在发生。同时BRKESTS1保持1且PWMx_CH1在刹车状态保持输出低。第二次事件也触发相同的标志，但是在这时候用户写1清除BRKEIF1，之后在下一个PWM周期开始后PWMx_CH1正常输出。

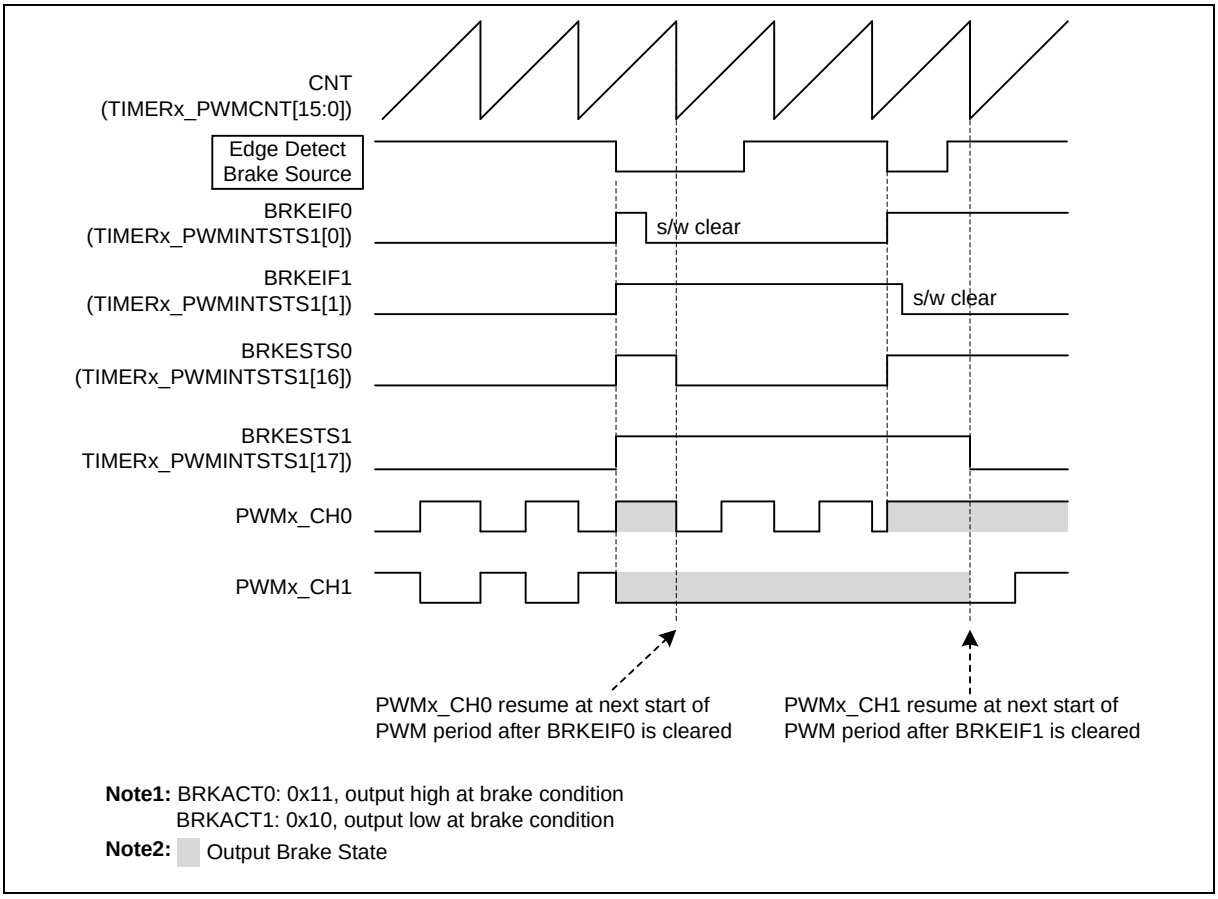


图 6.8-32PWMx_CH0 和 PWMx_CH1 的边沿侦测器刹车波形

当电平侦测器侦测到刹车信号，刹车功能产生中断状态（PWMx_CH1/0 是 BRKLIF1/0 (TIMERx_PWMINTSTS1[9:8]) ）和刹车事件状态（PWMx_CH1/0 是 BRKLSTS1/0 (TIMERx_PWMINTSTS1[25:24]).）。中断状态BRKLIF1/0通过写1清0，刹车事件状态BRKLSTS1/0仅在当前周期完成后且刹车条件移除清0，然后当下一个PWM周期开始PWM产生器重新开始正常输出。

下图是PWMx_CH0 和 PWMx_CH1电平侦测器刹车波形的例子。在这个情况中，BRKLIF0 和 BRKLIF1 仅表示刹车事件已发生。写1清除该标志，不影响BRKLSTS0 和 BRKLSTS1的刹车事件状态。BRKLSTS0 和 BRKLSTS1刹车状态都会在当刹车条件移除（无论BRKLIF0 和 BRKLIF1状态怎样），下一个PWM周期开始后自动清除。

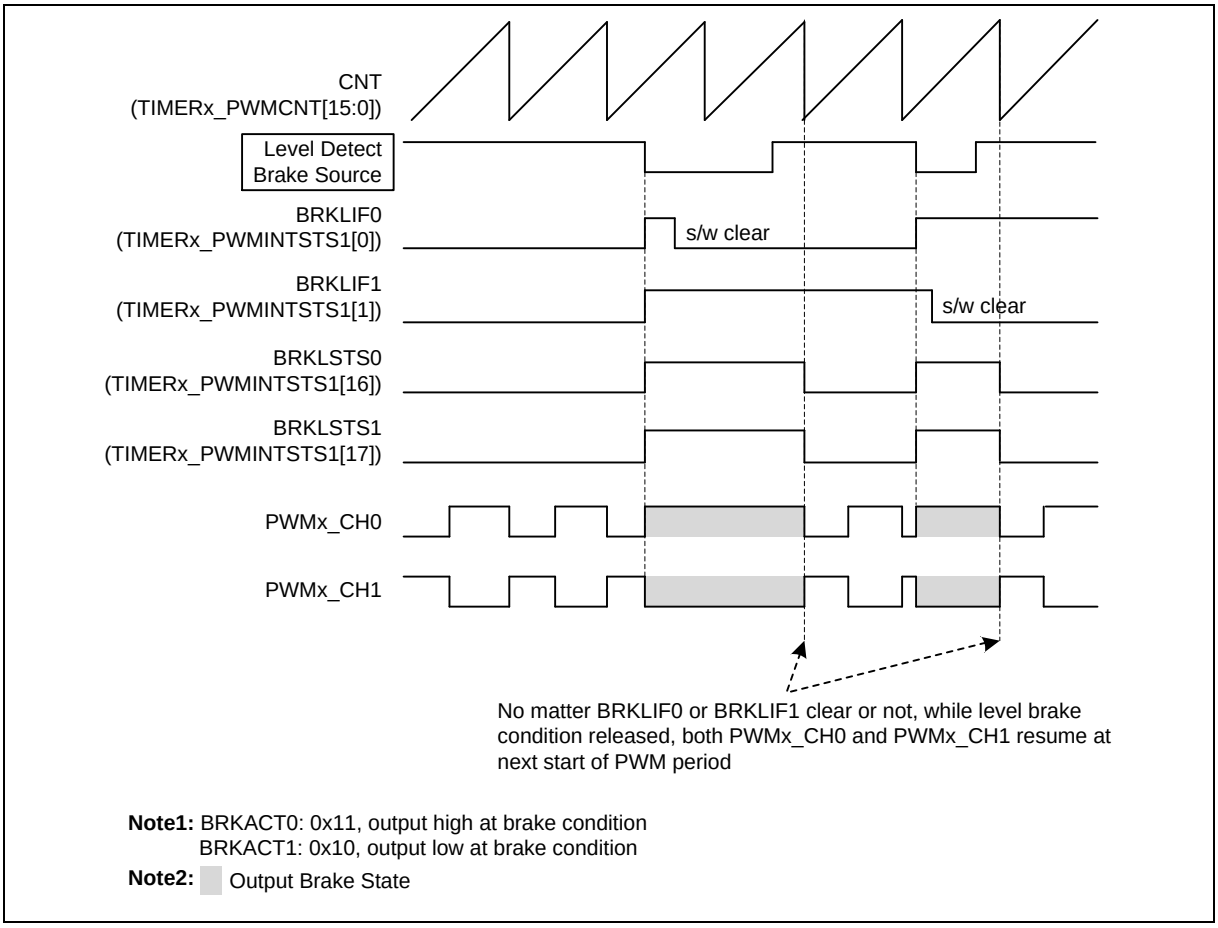


图 6.8-33PWMx_CH0 和 PWMx_CH1 的电平侦测器刹车波形

两种侦测器侦测相同的五个刹车源，如下图。一个来自PWMx_BRAKEy (x=0、1和 y=0、1)外部输入信号，两个来自内部ACMP比较器信号，一个来自系统失败事件和一个来自软件触发刹车事件。仅当内部ACMP0_O 或 ACMP1_O 信号从低到高的时候ACMP刹车源才会被侦测到。

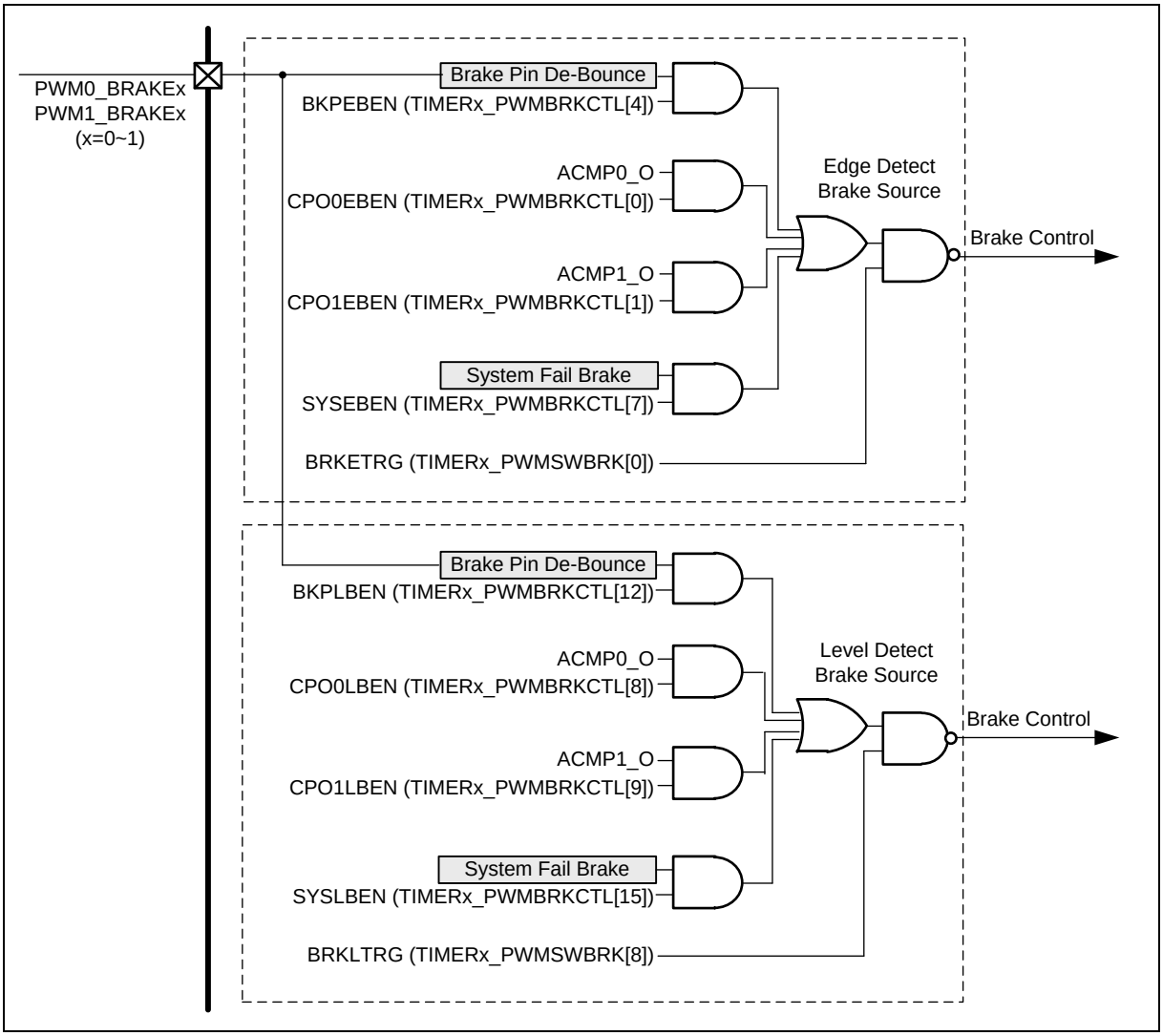


图 6.8-34 刹车源框图

在上面表述的刹车源中，来自系统失败事件的刹车源可以指定为不同的系统失败条件中的某一种。这些条件包括时钟失败、BOD侦测、SRAM校验错误和CPU锁死，如下图：

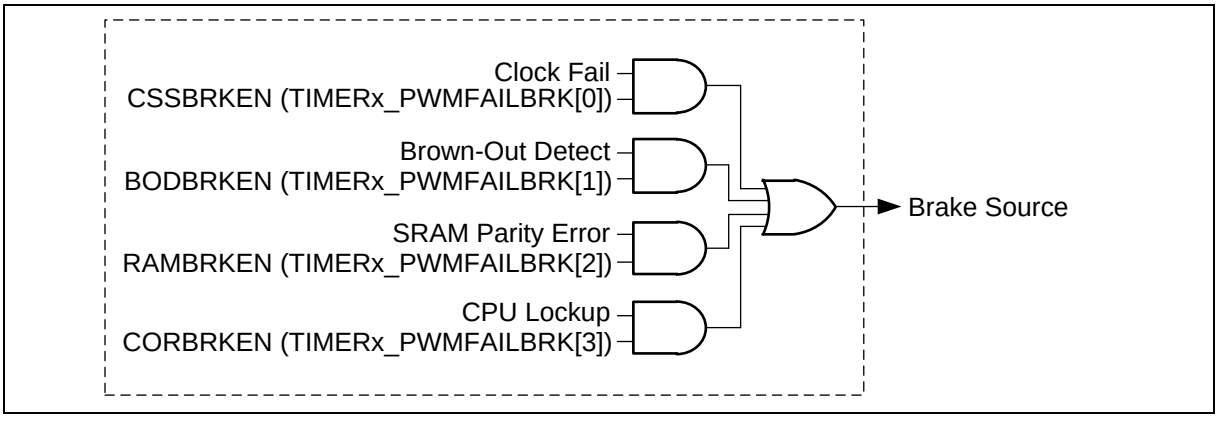


图 6.8-35 系统失败刹车框图

6.8.6.18 极性控制

PWMx_CH0 和 PWMx_CH1各自都有一个独立的极性控制器来控制PWM输出极性。用户可以控制PWMx_CH0 （在 PINV0 (TIMERx_PWMPOLCTL[0]) ） 和 PWMx_CH1 （在 PINV1 (TIMERx_PWMPOLCTL[1])）的极性状态。下图为PWMx_CH0 和 PWMx_CH1带极性控制和死区插入的输出。

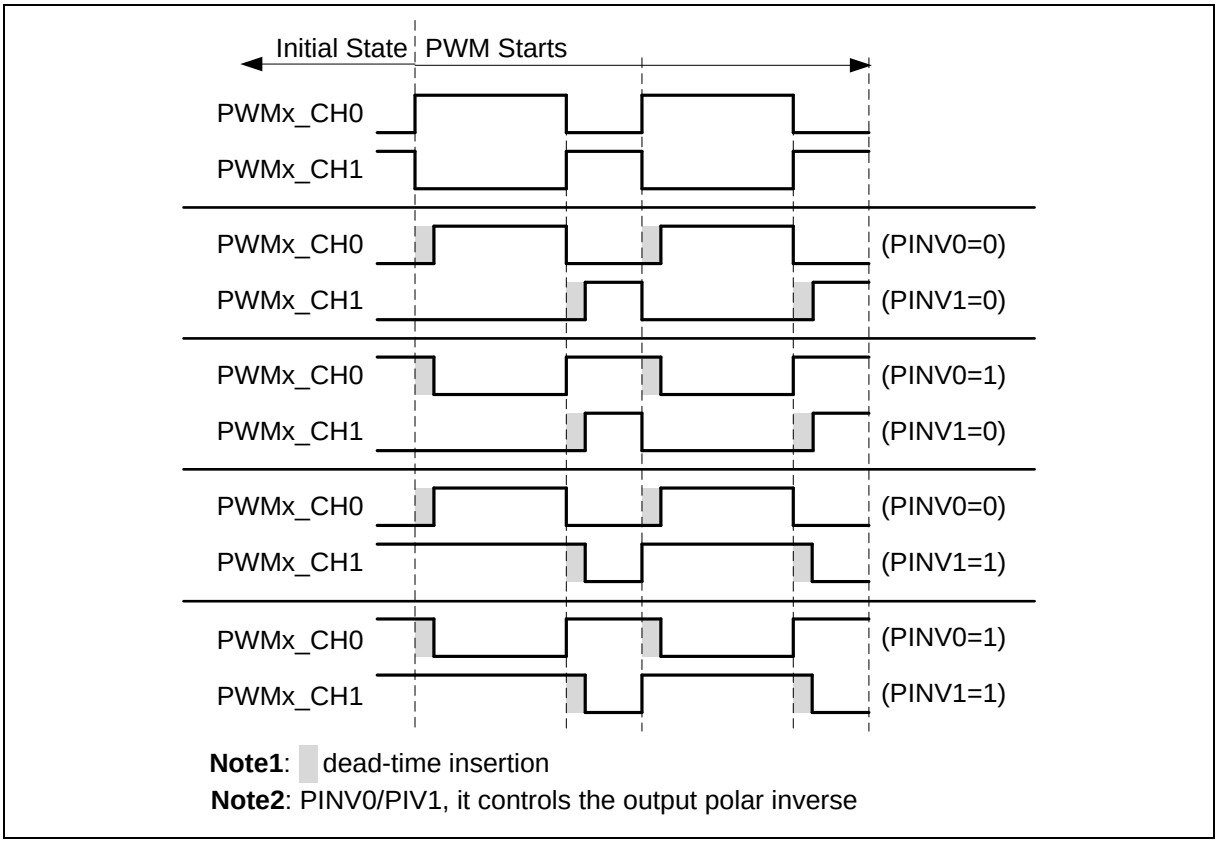


图 6.8-36PWMx_CH0 和 PWMx_CH1 带死区插入的极性控制

6.8.6.19 PWM中断产生器

每一个PWM都有独立的中断，如下图。

PWM 中断 (PWMx_INT) 来自 PWM 互补对事件。计数器可以产生零点中断标志 ZIF (TIMERx_PWMINTSTS0[0]) 和周期点中断标志 PIF (TIMERx_PWMINTSTS0[1])。当计数器等于比较值CMP (TIMERx_PWMCMPDAT[15:0])，不同的中断标志会被触发，这取决于计数方向。如果计数器和CMP匹配发生在上数方向，比较器上中断标志CMPUIF (TIMERx_PWMINTSTS0[2])被置位，如果匹配发生在下数方向，比较器下中断标志CMPDIF (TIMERx_PWMINTSTS0[3])被置位。如果对应中断使能位置位，中断触发事件也会产生中断信号。当PWM刹车事件发生，根据PWM刹车设置，相关的中断事件会触发。

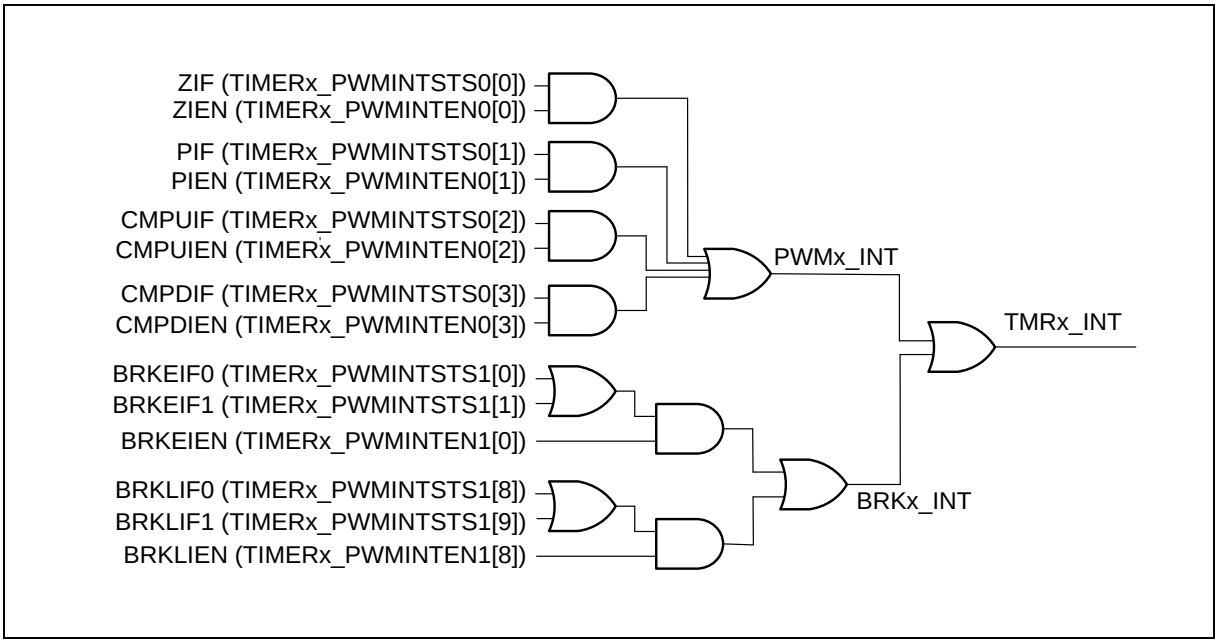


图 6.8-37 PWM 中断架构图

6.8.6.20 PWM触发ADC产生器

PWM计数事件是ADC转换触发源之一。当TRGEN (TIMERx_PWMADCTS[7])被使能后，通过设置TRGSEL (TIMERx_PWMADCTS[3:0])选择PWM计数事件触发ADC转换。

有五个PWM计数器事件可以选择作为触发源启动ADC转换，如下图：

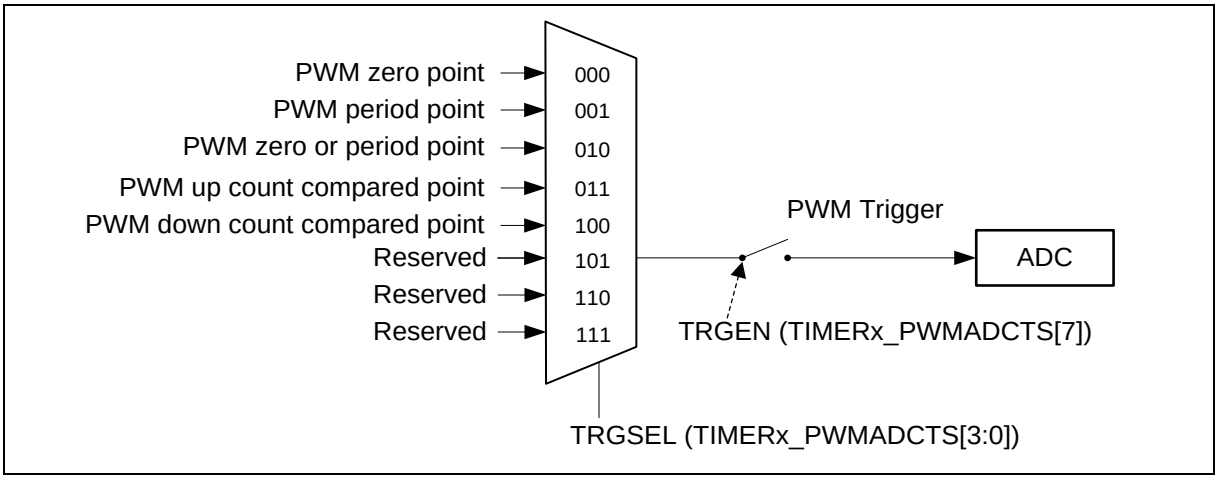


图 6.8-38 PWM 触发 ADC 框图

6.8.7 寄存器映射

R: 只读, **W**: 只写, **R/W**: 读/写

寄存器	偏移地址	R/W	描述	复位值
TIMER 基地址: **TMR01_BA = 0x4005_0000** **TMR23_BA = 0x4005_1000**				
TIMER0_CTL	TMR01_BA+0x00	R/W	Timer0控制寄存器	0x0000_0005
TIMER0_CMP	TMR01_BA+0x04	R/W	Timer0比较寄存器	0x0000_0000
TIMER0_INTSTS	TMR01_BA+0x08	R/W	Timer0中断状态寄存器	0x0000_0000
TIMER0_CNT	TMR01_BA+0x0C	R/W	Timer0数据寄存器	0x0000_0000
TIMER0_CAP	TMR01_BA+0x10	R	Timer0捕获数据寄存器	0x0000_0000
TIMER0_EXTCTL	TMR01_BA+0x14	R/W	Timer0外部控制寄存器	0x0000_0000
TIMER0_EINTSTS	TMR01_BA+0x18	R/W	Timer0外部中断状态寄存器	0x0000_0000
TIMER0_TRGCTL	TMR01_BA+0x1C	R/W	Timer0触发控制寄存器	0x0000_0000
TIMER0_ALTCTL	TMR01_BA+0x20	R/W	Timer0选择控制寄存器	0x0000_0000
TIMER0_PWMCTL	TMR01_BA+0x40	R/W	Timer0 PWM控制寄存器	0x0000_0000
TIMER0_PWMCLKSRC	TMR01_BA+0x44	R/W	Timer0 PWM计数器时钟源寄存器	0x0000_0000
TIMER0_PWMCLKPSC	TMR01_BA+0x48	R/W	Timer0 PWM计数器时钟预分频寄存器	0x0000_0000
TIMER0_PWMCNTCLR	TMR01_BA+0x4C	R/W	Timer0 PWM清计数器寄存器	0x0000_0000
TIMER0_PWMPERIOD	TMR01_BA+0x50	R/W	Timer0 PWM周期寄存器	0x0000_0000
TIMER0_PWMCMPDAT	TMR01_BA+0x54	R/W	Timer0 PWM比较寄存器	0x0000_0000
TIMER0_PWMDTCTL	TMR01_BA+0x58	R/W	Timer0 PWM死区控制寄存器	0x0000_0000
TIMER0_PWMCNT	TMR01_BA+0x5C	R	Timer0 PWM计数器寄存器	0x0000_0000
TIMER0_PWMMSKEN	TMR01_BA+0x60	R/W	Timer0 PWM输出屏蔽使能寄存器	0x0000_0000
TIMER0_PWMMSK	TMR01_BA+0x64	R/W	Timer0 PWM输出屏蔽数据控制寄存器	0x0000_0000
TIMER0_PWMBNF	TMR01_BA+0x68	R/W	Timer0 PWM刹车管脚噪音滤波寄存器	0x0000_0000
TIMER0_PWMFAILBRK	TMR01_BA+0x6C	R/W	Timer0 PWM系统失败刹车控制寄存器	0x0000_0000
TIMER0_PWMBRKCTL	TMR01_BA+0x70	R/W	Timer0 PWM刹车控制寄存器	0x0000_0000
TIMER0_PWMPOLCTL	TMR01_BA+0x74	R/W	Timer0 PWM管脚输出极性控制寄存器	0x0000_0000
TIMER0_PWMPOEN	TMR01_BA+0x78	R/W	Timer0 PWM管脚输出使能寄存器	0x0000_0000
TIMER0_PWMSWBRK	TMR01_BA+0x7C	W	Timer0 PWM软件触发刹车控制寄存器	0x0000_0000

TIMER0_PWMINTEN0	TMR01_BA+0x80	R/W	Timer0 PWM中断使能寄存器0	0x0000_0000
TIMER0_PWMINTEN1	TMR01_BA+0x84	R/W	Timer0 PWM中断使能寄存器1	0x0000_0000
TIMER0_PWMINTSTS0	TMR01_BA+0x88	R/W	Timer0 PWM中断状态寄存器0	0x0000_0000
TIMER0_PWMINTSTS1	TMR01_BA+0x8C	R/W	Timer0 PWM中断状态寄存器1	0x0000_0000
TIMER0_PWMADCTS	TMR01_BA+0x90	R/W	Timer0 PWM ADC触发源选择控制寄存器	0x0000_0000
TIMER0_PWMSCTL	TMR01_BA+0x94	R/W	Timer0 PWM同步控制寄存器	0x0000_0000
TIMER0_PWMSTRG	TMR01_BA+0x98	W	Timer0 PWM同步触发寄存器	0x0000_0000
TIMER0_PWMSTATUS	TMR01_BA+0x9C	R/W	Timer0 PWM状态寄存器	0x0000_0000
TIMER0_PWMPBUF	TMR01_BA+0xA0	R	Timer0 PWM周期缓存寄存器	0x0000_0000
TIMER0_PWMCMPBUF	TMR01_BA+0xA4	R	Timer0 PWM比较器缓存寄存器	0x0000_0000
TIMER1_CTL	TMR01_BA+0x100	R/W	Timer1控制寄存器	0x0000_0005
TIMER1_CMP	TMR01_BA+0x104	R/W	Timer1比较寄存器	0x0000_0000
TIMER1_INTSTS	TMR01_BA+0x108	R/W	Timer1中断状态寄存器	0x0000_0000
TIMER1_CNT	TMR01_BA+0x10C	R/W	Timer1数据寄存器	0x0000_0000
TIMER1_CAP	TMR01_BA+0x110	R	Timer1捕获数据寄存器	0x0000_0000
TIMER1_EXTCTL	TMR01_BA+0x114	R/W	Timer1外部控制寄存器	0x0000_0000
TIMER1_EINTSTS	TMR01_BA+0x118	R/W	Timer1外部中断状态寄存器	0x0000_0000
TIMER1_TRGCTL	TMR01_BA+0x11C	R/W	Timer1触发控制寄存器	0x0000_0000
TIMER1_ALTCTL	TMR01_BA+0x120	R/W	Timer1选择控制寄存器	0x0000_0000
TIMER1_PWMCTL	TMR01_BA+0x140	R/W	Timer1 PWM控制寄存器	0x0000_0000
TIMER1_PWMCLKSRC	TMR01_BA+0x144	R/W	Timer1 PWM计数器时钟源寄存器	0x0000_0000
TIMER1_PWMCLKPSC	TMR01_BA+0x148	R/W	Timer1 PWM计数器时钟预分频寄存器	0x0000_0000
TIMER1_PWMCNTCLR	TMR01_BA+0x14C	R/W	Timer1 PWM清计数器寄存器	0x0000_0000
TIMER1_PWMPERIOD	TMR01_BA+0x150	R/W	Timer1 PWM周期寄存器	0x0000_0000
TIMER1_PWMCMPDAT	TMR01_BA+0x154	R/W	Timer1 PWM比较寄存器	0x0000_0000
TIMER1_PWMDTCTL	TMR01_BA+0x158	R/W	Timer1 PWM死区控制寄存器	0x0000_0000
TIMER1_PWMCNT	TMR01_BA+0x15C	R	Timer1 PWM计数器寄存器	0x0000_0000
TIMER1_PWMMSKEN	TMR01_BA+0x160	R/W	Timer1 PWM输出屏蔽使能寄存器	0x0000_0000
TIMER1_PWMMSK	TMR01_BA+0x164	R/W	Timer1 PWM输出屏蔽数据控制寄存器	0x0000_0000
TIMER1_PWMBNF	TMR01_BA+0x168	R/W	Timer1 PWM刹车管脚噪音滤波寄存器	0x0000_0000

TIMER1_PWMFAILBRK	TMR01_BA+0x16C	R/W	Timer1 PWM系统失败刹车控制寄存器	0x0000_0000
TIMER1_PWMBRKCTL	TMR01_BA+0x170	R/W	Timer1 PWM刹车控制寄存器	0x0000_0000
TIMER1_PWMPOLCTL	TMR01_BA+0x174	R/W	Timer1 PWM管脚输出极性控制寄存器	0x0000_0000
TIMER1_PWMPOEN	TMR01_BA+0x178	R/W	Timer1 PWM管脚输出使能寄存器	0x0000_0000
TIMER1_PWMSWBRK	TMR01_BA+0x17C	W	Timer1 PWM软件触发刹车控制寄存器	0x0000_0000
TIMER1_PWMINTEN0	TMR01_BA+0x180	R/W	Timer1 PWM中断使能寄存器0	0x0000_0000
TIMER1_PWMINTEN1	TMR01_BA+0x184	R/W	Timer1 PWM中断使能寄存器1	0x0000_0000
TIMER1_PWMINTSTS0	TMR01_BA+0x188	R/W	Timer1 PWM中断状态寄存器0	0x0000_0000
TIMER1_PWMINTSTS1	TMR01_BA+0x18C	R/W	Timer1 PWM中断状态寄存器1	0x0000_0000
TIMER1_PWMADCTS	TMR01_BA+0x190	R/W	Timer1 PWMADC触发源选择控制寄存器	0x0000_0000
TIMER1_PWMSCTL	TMR01_BA+0x194	R/W	Timer1 PWM同步控制寄存器	0x0000_0000
TIMER1_PWMSTATUS	TMR01_BA+0x19C	R/W	Timer1 PWM状态寄存器	0x0000_0000
TIMER1_PWMPBUF	TMR01_BA+0x1A0	R	Timer1 PWM周期缓存寄存器	0x0000_0000
TIMER1_PWMCMPBUF	TMR01_BA+0x1A4	R	Timer1 PWM比较器缓存寄存器	0x0000_0000
TIMER2_CTL	TMR23_BA+0x00	R/W	Timer2控制寄存器	0x0000_0005
TIMER2_CMP	TMR23_BA+0x04	R/W	Timer2比较寄存器	0x0000_0000
TIMER2_INTSTS	TMR23_BA+0x08	R/W	Timer2中断状态寄存器	0x0000_0000
TIMER2_CNT	TMR23_BA+0x0C	R/W	Timer2数据寄存器	0x0000_0000
TIMER2_CAP	TMR23_BA+0x10	R	Timer2捕获数据寄存器	0x0000_0000
TIMER2_EXTCTL	TMR23_BA+0x14	R/W	Timer2外部控制寄存器	0x0000_0000
TIMER2_EINTSTS	TMR23_BA+0x18	R/W	Timer2外部中断状态寄存器	0x0000_0000
TIMER2_TRGCTL	TMR23_BA+0x1C	R/W	Timer2触发控制寄存器	0x0000_0000
TIMER2_ALTCTL	TMR23_BA+0x20	R/W	Timer2选择控制寄存器	0x0000_0000
TIMER2_PWMCTL	TMR23_BA+0x40	R/W	Timer2 PWM控制寄存器	0x0000_0000
TIMER2_PWMCLKSRC	TMR23_BA+0x44	R/W	Timer2 PWM计数器时钟源寄存器	0x0000_0000
TIMER2_PWMCLKPSC	TMR23_BA+0x48	R/W	Timer2 PWM计数器时钟预分频寄存器	0x0000_0000
TIMER2_PWMCNTCLR	TMR23_BA+0x4C	R/W	Timer2 PWM清计数器寄存器	0x0000_0000
TIMER2_PWMPERIOD	TMR23_BA+0x50	R/W	Timer2 PWM周期寄存器	0x0000_0000
TIMER2_PWMCMPDAT	TMR23_BA+0x54	R/W	Timer2 PWM比较寄存器	0x0000_0000
TIMER2_PWMDTCTL	TMR23_BA+0x58	R/W	Timer2 PWM死区控制寄存器	0x0000_0000

TIMER2_PWMCNT	TMR23_BA+0x5C	R	Timer2 PWM计数器寄存器	0x0000_0000
TIMER2_PWMMSKEN	TMR23_BA+0x60	R/W	Timer2 PWM输出屏蔽使能寄存器	0x0000_0000
TIMER2_PWMMSK	TMR23_BA+0x64	R/W	Timer2 PWM输出屏蔽数据控制寄存器	0x0000_0000
TIMER2_PWMBNF	TMR23_BA+0x68	R/W	Timer2 PWM刹车管脚噪音滤波寄存器	0x0000_0000
TIMER2_PWMFAILBRK	TMR23_BA+0x6C	R/W	Timer2 PWM系统失败刹车控制寄存器	0x0000_0000
TIMER2_PWMBRKCTL	TMR23_BA+0x70	R/W	Timer2 PWM刹车控制寄存器	0x0000_0000
TIMER2_PWMPOLCTL	TMR23_BA+0x74	R/W	Timer2 PWM管脚输出极性控制寄存器	0x0000_0000
TIMER2_PWMPOEN	TMR23_BA+0x78	R/W	Timer2 PWM管脚输出使能寄存器	0x0000_0000
TIMER2_PWMSWBRK	TMR23_BA+0x7C	W	Timer2 PWM软件触发刹车控制寄存器	0x0000_0000
TIMER2_PWMINTEN0	TMR23_BA+0x80	R/W	Timer2 PWM中断使能寄存器0	0x0000_0000
TIMER2_PWMINTEN1	TMR23_BA+0x84	R/W	Timer2 PWM中断使能寄存器1	0x0000_0000
TIMER2_PWMINTSTS0	TMR23_BA+0x88	R/W	Timer2 PWM中断状态寄存器0	0x0000_0000
TIMER2_PWMINTSTS1	TMR23_BA+0x8C	R/W	Timer2 PWM中断状态寄存器1	0x0000_0000
TIMER2_PWMADCTS	TMR23_BA+0x90	R/W	Timer2 PWM ADC触发源选择控制寄存器	0x0000_0000
TIMER2_PWMSCTL	TMR23_BA+0x94	R/W	Timer2 PWM同步控制寄存器	0x0000_0000
TIMER2_PWMSTRG	TMR23_BA+0x98	W	Timer2 PWM同步触发寄存器	0x0000_0000
TIMER2_PWMSTATUS	TMR23_BA+0x9C	R/W	Timer2 PWM状态寄存器	0x0000_0000
TIMER2_PWMPBUF	TMR23_BA+0xA0	R	Timer2 PWM周期缓存寄存器	0x0000_0000
TIMER2_PWMCMPBUF	TMR23_BA+0xA4	R	Timer2 PWM比较器缓存寄存器	0x0000_0000
TIMER3_CTL	TMR23_BA+0x100	R/W	Timer3控制寄存器	0x0000_0005
TIMER3_CMP	TMR23_BA+0x104	R/W	Timer3比较寄存器	0x0000_0000
TIMER3_INTSTS	TMR23_BA+0x108	R/W	Timer3中断状态寄存器	0x0000_0000
TIMER3_CNT	TMR23_BA+0x10C	R/W	Timer3数据寄存器	0x0000_0000
TIMER3_CAP	TMR23_BA+0x110	R	Timer3捕获数据寄存器	0x0000_0000
TIMER3_EXTCTL	TMR23_BA+0x114	R/W	Timer3外部控制寄存器	0x0000_0000
TIMER3_EINTSTS	TMR23_BA+0x118	R/W	Timer3外部中断状态寄存器	0x0000_0000
TIMER3_TRGCTL	TMR23_BA+0x11C	R/W	Timer3触发控制寄存器	0x0000_0000
TIMER3_ALTCTL	TMR23_BA+0x120	R/W	Timer3选择控制寄存器	0x0000_0000
TIMER3_PWMCTL	TMR23_BA+0x140	R/W	Timer3 PWM控制寄存器	0x0000_0000
TIMER3_PWMCLKSRC	TMR23_BA+0x144	R/W	Timer3 PWM计数器时钟源寄存器	0x0000_0000

TIMER3_PWMCLKPSC	TMR23_BA+0x148	R/W	Timer3 PWM计数器时钟预分频寄存器	0x0000_0000
TIMER3_PWMCNTCLR	TMR23_BA+0x14C	R/W	Timer3 PWM清计数器寄存器	0x0000_0000
TIMER3_PWMPERIOD	TMR23_BA+0x150	R/W	Timer3 PWM周期寄存器	0x0000_0000
TIMER3_PWMCMPDAT	TMR23_BA+0x154	R/W	Timer3 PWM比较寄存器	0x0000_0000
TIMER3_PWMDTCTL	TMR23_BA+0x158	R/W	Timer3 PWM死区控制寄存器	0x0000_0000
TIMER3_PWMCNT	TMR23_BA+0x15C	R	Timer3 PWM计数器寄存器	0x0000_0000
TIMER3_PWMMSKEN	TMR23_BA+0x160	R/W	Timer3 PWM输出屏蔽使能寄存器	0x0000_0000
TIMER3_PWMMSK	TMR23_BA+0x164	R/W	Timer3 PWM输出屏蔽数据控制寄存器	0x0000_0000
TIMER3_PWMBNF	TMR23_BA+0x168	R/W	Timer3 PWM刹车管脚噪音滤波寄存器	0x0000_0000
TIMER3_PWMFAILBRK	TMR23_BA+0x16C	R/W	Timer3 PWM系统失败刹车控制寄存器	0x0000_0000
TIMER3_PWMBRKCTL	TMR23_BA+0x170	R/W	Timer3 PWM刹车控制寄存器	0x0000_0000
TIMER3_PWMPOLCTL	TMR23_BA+0x174	R/W	Timer3 PWM管脚输出极性控制寄存器	0x0000_0000
TIMER3_PWMPOEN	TMR23_BA+0x178	R/W	Timer3 PWM管脚输出使能寄存器	0x0000_0000
TIMER3_PWMSWBRK	TMR23_BA+0x17C	W	Timer3 PWM软件触发刹车控制寄存器	0x0000_0000
TIMER3_PWMINTEN0	TMR23_BA+0x180	R/W	Timer3 PWM中断使能寄存器0	0x0000_0000
TIMER3_PWMINTEN1	TMR23_BA+0x184	R/W	Timer3 PWM中断使能寄存器1	0x0000_0000
TIMER3_PWMINTSTS0	TMR23_BA+0x188	R/W	Timer3 PWM 中断状态寄存器0	0x0000_0000
TIMER3_PWMINTSTS1	TMR23_BA+0x18C	R/W	Timer3 PWM中断状态寄存器1	0x0000_0000
TIMER3_PWMADCTS	TMR23_BA+0x190	R/W	Timer3 PWM ADC触发源选择控制寄存器	0x0000_0000
TIMER3_PWMSCTL	TMR23_BA+0x194	R/W	Timer3 PWM同步控制寄存器	0x0000_0000
TIMER3_PWMSTATUS	TMR23_BA+0x19C	R/W	Timer3 PWM状态寄存器	0x0000_0000
TIMER3_PWMPBUF	TMR23_BA+0x1A0	R	Timer3 PWM周期缓存寄存器	0x0000_0000
TIMER3_PWMCMPBUF	TMR23_BA+0x1A4	R	Timer3 PWM比较器缓存寄存器	0x0000_0000

6.8.8 寄存器描述

定时器控制寄存器(TIMERx_CTL)

寄存器	偏移地址	R/W	描述	复位值
TIMER0_CTL	TMR01_BA+0x00	R/W	Timer0控制寄存器	0x0000_0005
TIMER1_CTL	TMR01_BA+0x100	R/W	Timer1控制寄存器	0x0000_0005
TIMER2_CTL	TMR23_BA+0x00	R/W	Timer2控制寄存器	0x0000_0005
TIMER3_CTL	TMR23_BA+0x100	R/W	Timer3控制寄存器	0x0000_0005

31	30	29	28	27	26	25	24
ICEDEBUG	CNTEN	INTEN	OPMODE		Reserved	ACTSTS	EXTCNTEN
23	22	21	20	19	18	17	16
WKEN	CAPSRC	TGLPINSEL	PERIOSEL	INTRGEN	Reserved		
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
PSC							

位	描述	
[31]	**ICEDEBUG**	**仿真器 (ICE) 调试模式响应禁止位（写保护位）** 0 = 仿真器调试模式响应影响定时器计数 当CPU被ICE停住时，定时器计数器将被固定住。 1 = 仿真器调试模式响应禁用 无论CPU是否被ICE停住，定时器计数器将持续计数下去。 **注：**该位为写保护位，详情请参考SYS_REGLCTL 寄存器。
[30]	**CNTEN**	**定时器计数使能位** 0 = 停止/暂停计数 1 = 开始计数 **注1：**在停止状态，设置CNTEN为 1 将使能 24位向上计数器从上次停止的计数值继续计数； **注2：**在 one-shot 模式下(TIMER_CTL[28:27] = 00)，当相应的定时中断标志TIF (TIMERx_INTSTS[0])产生时，该位由硬件自动清零； **注3：**设置使能/禁止该位需要2 * TMR_CLK个时钟周期生效。用户可以读ACTSTS (TIMERx_CTL[25])来检查使能/禁止命令是否完成。
[29]	**INTEN**	**定时器中断使能位** 0 = 禁用定时器超时中断 1 = 使能定时器超时中断

		注：如果该位使能，当定时器中断标志TIF置 1，定时器中断信号产生并通知CPU。
[28:27]	**OPMODE**	**定时器计数模式选择** 00 = 定时器工作在单次触发模式 (one-shot) 01 =定时器工作在周期模式 (Periodic) 10 =定时器工作在触发模式（Toggle） 11 = 定时器工作在连续计数模式(Continuous Counting）
[26]	**Reserved**	保留
[25]	**ACTSTS**	**定时器激活状态位（只读）** 该位表示24位向上计数器的状态。 0 = 定时器计数器未激活 1 = 定时器计数器激活
[24]	**EXTCNTEN**	**事件计数模式使能位** 该位用来使能外部计数管脚功能。 0 = 禁用事件计数器模式 1 = 使能事件计数器模式 **注**：当定时器用作事件计数器，该位需要设置成1，并选择PCLK作为定时器时钟源。
[23]	**WKEN**	**唤醒功能使能位** 如果该位置1，当定时器中断标志TIF (TIMERx_INTSTS[0]) 被置1，且INTEN (TIMERx_CTL[29])被使能，定时器中断信号将产生一个唤醒触发事件给CPU。 0 = 如果定时器中断信号产生，唤醒功能禁止 1 =如果定时器中断信号产生，唤醒功能使能
[22]	**CAPSRC**	**捕获引脚源选择** 0 = 捕获源来自TMx_EXT (x= 0~3) 引脚 1 = 捕获源来自内部ACMP输出信号、内部时钟源(HIRC 12M、LIRC) 或外部时钟 (HXT、LXT) **注**：当CAPSRC = 1，用户可以设置INTERCAPSEL (TIMERx_EXTCTL[10:8])决定哪一个内部ACMP输出信号或哪一个时钟作为定时器捕获源。
[21]	**TGLPINSEL**	**触发输出管脚选择** 0 = 触发模式输出到TMx脚(定时器事件计数管脚) 1 = 触发模式输出到TMx _EXT (定时器外部捕获管脚)
[20]	**PERIOSEL**	**周期模式行为选择使能位** 0 =在周期模式行为选择禁止 当定时器正在周期模式运行的时候，用户更新CMPDAT，CNT会复位到默认值。 1 = 在周期模式行为选择使能 当定时器正在周期模式运行的时候，用户更新CMPDAT有如下限制： 如果CMPDAT值大于CNT，CMPDAT会被更新且CNT保持继续计数。 如果CMPDAT值等于CNT，定时器超时中断会立即产生。 如果CMPDAT值小于CNT，CNT会复位到默认值。
[19]	**INTRGEN**	**定时器间互触发模式使能控制** 设置该位会使能定时器间互触发捕获功能。 定时器0/2会工作在事件计数模式，计数外部时钟源或事件。

		定时器1/3会工作在捕获功能的触发计数模式。 0 = 定时器间互触发捕获模式禁止 1 =定时器间互触发捕获模式使能 **注：** 对于Timer1/3, 该位被忽略，读回总是0。
[18:8]	**Reserved**	保留
[7:0]	**PSC**	**预分频计数器** 定时器的输入时钟源或事件计数源被PSC +1预分频，然后再输入到定时器。如果该位为0，表示不分频。 **注：** 更新预分频计数器值会复位内部8位预分频计数器和24位上数计数器值。

定时器比较寄存器(TIMERx_CMP)

寄存器	偏移地址	R/W	描述	复位值
TIMER0_CMP	TMR01_BA+0x04	R/W	Timer0比较寄存器	0x0000_0000
TIMER1_CMP	TMR01_BA+0x104	R/W	Timer1比较寄存器	0x0000_0000
TIMER2_CMP	TMR23_BA+0x04	R/W	Timer2比较寄存器	0x0000_0000
TIMER3_CMP	TMR23_BA+0x104	R/W	Timer3比较寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
CMPDAT							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
CMPDAT							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
CMPDAT							

位	描述	
[31:24]	**Reserved**	保留
[23:0]	**CMPDAT**	**定时器比较值** CMPDAT是24位比较寄存器。当内部24位向上计数器的值等于CMPDAT的值时，TIF (TIMERx_INTSTS[0] 定时器中断标志)将被置1。 超时溢出周期 = (定时器输入时钟周期) * (8位 PSC + 1) * (24位 CMPDAT) **注**1：不能向CMPDAT里写 0x0 或 0x1，否则内核将运行到未知状态。 **注**2：当定时器工作在连续计数模式，即使软件写一个新的值到CMPDAT，24位向上计数定时器将保持继续计数。如果定时器工作在其他模式，如果软件写一个新的值到CMPDAT，定时器将使用新比较值并退出当前计数，重新从0开始计数。

定时器中断状态寄存器(TIMERx_INTSTS)

寄存器	偏移地址	R/W	描述	复位值
TIMER0_INTSTS	TMR01_BA+0x08	R/W	Timer0中断状态寄存器	0x0000_0000
TIMER1_INTSTS	TMR01_BA+0x108	R/W	Timer1中断状态寄存器	0x0000_0000
TIMER2_INTSTS	TMR23_BA+0x08	R/W	Timer2中断状态寄存器	0x0000_0000
TIMER3_INTSTS	TMR23_BA+0x108	R/W	Timer3中断状态寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved						**TWKF**	**TIF**

位	描述	
[31:2]	**Reserved**	保留
[1]	**TWKF**	**定时器唤醒标志** 该位表示定时器的中断唤醒标志状态。 0 = 定时器不会引起CPU 唤醒 1 =如果定时器超时中断信号产生， CPU 从空闲或掉电模式唤醒 **注：**该位必须通过软件写1清0
[0]	**TIF**	**定时器中断标志** 当内部 24位向上计数定时器CNT (TIMERx_CNT[23:0])的值与定时器比较值CMPDAT (TIMERx_CMP[23:0])匹配时，该位会显示其中断状态。 0 = 无影响 1 = 计数定时器与CMPDAT的值相匹配 注：该位写 1 清零。

定时器数据寄存器(TIMERx_CNT)

寄存器	偏移地址	R/W	描述	复位值
TIMER0_CNT	TMR01_BA+0x0C	R/W	Timer0数据寄存器	0x0000_0000
TIMER1_CNT	TMR01_BA+0x10C	R/W	Timer1数据寄存器	0x0000_0000
TIMER2_CNT	TMR23_BA+0x0C	R/W	Timer2数据寄存器	0x0000_0000
TIMER3_CNT	TMR23_BA+0x10C	R/W	Timer3数据寄存器	0x0000_0000

31	30	29	28	27	26	25	24
RSTACT	**Reserved**						
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
CNT							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
CNT							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
CNT							

位	描述	
[31]	**RSTACT**	**定时器数据寄存器复位生效 (只读)** 该位表示计数器复位操作是否生效 当用户写CNT寄存器，定时器开始复位内部24位定时器上数计数器到0，且重载8位预分频计数器，同时定时器置位该标志为1表示计数器复位操作正在进行。一旦计数器复位操作完成，该位自动清0。 0 = 复位操作已完成 1 =通过写TIMERx_CNT触发的复位操作正在进行
[30:24]	**Reserved**	保留
[23:0]	**CNT**	**定时器数据寄存器** 读操作 读该寄存器，获取 CNT值，例如： 如果 EXTCNTEN (TIMERx_CTL[24]) 是 0，用户可以读CNT的值以获取当前24位计数器值。 如果 EXTCNTEN (TIMERx_CTL[24]) 是 1，用户可以读CNT的值以获取当前24位事件输入计数器值。 写操作 写任何值到该寄存器会复位当前CNT值为0 ，且重载内部8位预分频计数器。

定时器捕获数据寄存器(TIMERx_CAP)

寄存器	偏移地址	R/W	描述	复位值
TIMER0_CAP	TMR01_BA+0x10	R	Timer0捕获数据寄存器	0x0000_0000
TIMER1_CAP	TMR01_BA+0x110	R	Timer1捕获数据寄存器	0x0000_0000
TIMER2_CAP	TMR23_BA+0x10	R	Timer2捕获数据寄存器	0x0000_0000
TIMER3_CAP	TMR23_BA+0x110	R	Timer3捕获数据寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
CAPDAT							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
CAPDAT							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
CAPDAT							

位	描述	
[31:24]	**Reserved**	保留
[23:0]	**CAPDAT**	**定时器捕获数据寄存器** 当 CAPEN (TIMERx_EXTCTL[3]) 位被置1, CAPFUNCS (TIMERx_EXTCTL[4])位为0, TMx _EXT 管脚的变化与CAPEDGE (TIMERx_EXTCTL[14:12])设定相匹配时, CAPIF (TIMERx_EINTSTS[0]) 将被置1且当前定时器计数器CNT (TIMERx_CNT[23:0]) 的值将被自动载入到CAPDAT。

定时器外部控制寄存器(TIMERx_EXTCTL)

寄存器	偏移地址	R/W	描述	复位值
TIMER0_EXTCTL	TMR01_BA+0x14	R/W	Timer0外部控制寄存器	0x0000_0000
TIMER1_EXTCTL	TMR01_BA+0x114	R/W	Timer1外部控制寄存器	0x0000_0000
TIMER2_EXTCTL	TMR23_BA+0x14	R/W	Timer2外部控制寄存器	0x0000_0000
TIMER3_EXTCTL	TMR23_BA+0x114	R/W	Timer3外部控制寄存器	0x0000_0000

31	30	29	28	27	26	25	24
CAPDIVSCL				Reserved			
23	22	21	20	19	18	17	16
Reserved							ECNTSSEL
15	14	13	12	11	10	9	8
Reserved	CAPEDGE			Reserved	ICAPSEL		
7	6	5	4	3	2	1	0
CNTDBEN	CAPDBEN	CAPIEN	CAPFUNCS	CAPEN	Reserved		CNTPHASE

位	描述	
[31:28]	**CAPDIVSCL**	**定时器捕获源除法器** **该位决定捕获源除法器除数大小** 0000 = 捕获源/1 0001 =捕获源/2 0010 =捕获源/4 0011 =捕获源/8 0100 =捕获源/16 0101 =捕获源/32 0110 =捕获源/64 0111 =捕获源/128 1000 =捕获源/256 1001~1111 = 保留 **注：** 设置 INTERCAPSEL (TIMERx_EXTCTL[10:8]) 和 CAPSRC (TIMERx_CTL[22])选择捕获源。
[27:17]	**Reserved**	保留
[16]	**ECNTSSEL**	**事件计数器源选择，用于触发事件计数器功能** 0 = 事件计数器输入源来自TMx (x= 0~3) 管脚

		1 = 事件计数器输入源来自 USB 内部 SOF 输出信号
[15]	**Reserved**	保留
[14:12]	**CAPEDGE**	**定时器外部捕获管脚边沿侦测** 当第一个捕获事件产生，CNT (TIMERx_CNT[23:0])会被复位为0，且第一个CAPDAT (TIMERx_CAP[23:0])应该是0 。 000 = 当在TMx_EXT (x= 0~3)管脚侦测到下降沿，捕获事件发生。 001 = 当在TMx_EXT (x= 0~3)管脚侦测到上升沿，捕获事件发生。 010 =当在TMx_EXT (x= 0~3)管脚侦测到下降沿和上升沿，捕获事件都发生。第一个捕获事件发生在下降沿。 011 = 当在TMx_EXT (x= 0~3)管脚侦测到下降沿和上升沿，捕获事件都发生。第一个捕获事件发生在上升沿。 110 = 第一个捕获事件发生在下降沿，接下来的捕获事件发生上升沿。 111 =第一个捕获事件发生在上升沿，接下来的捕获事件发生下降沿。 100, 101 = 保留
[11]	**Reserved**	保留
[10:8]	**ICAPSEL**	**内部捕获源选择** 000 = 捕获源来自内部ACMP0输出信号 001 = 捕获源来自内部ACMP1输出信号 010 = 捕获源来自HXT 011 = 捕获源来自LXT 100 = 捕获源来自HIRC 101 = 捕获源来自LIRC 110 = 保留 111 = 保留 **注：**这些位只有当CAPSRC (TIMERx_CTL[22])为1时有效。
[7]	**CNTDBEN**	**定时器外部计数器输入管脚消抖使能位** 0 = TMx (x= 0~3)管脚消抖禁用 1 = TMx (x= 0~3)管脚消抖使能 **注：**如果该位使能，TMx管脚边沿检测带消抖电路。
[6]	**CAPDBEN**	**定时器外部捕获输入管脚消抖使能位** 0 = TMx _EXT (x= 0~3)管脚或ACMP输出消抖禁用 1 = TMx _EXT (x= 0~3)管脚或ACMP输出消抖使能 注：如果该位使能，TMx _EXT管脚或ACMP输出边沿检测带消抖电路。
[5]	**CAPIEN**	**定时器外部捕获中断使能位** 0 = TMx _EXT (x= 0~3) 管脚检测中断禁止 1 = TMx _EXT (x= 0~3)管脚检测中断使能 **注：** CAPIEN 用于使能定时器外部中断。如果 CAPIEN 使能，当 CAPIF (TIMERx_EINTSTS[0])标志设为1时，定时器会产生一个外部捕获中断信号。 例如：当 CAPIEN = 1、CAPEN = 1且CAPEDGE = 00，TMx_EXT 管脚上的由1变0变化，将导致CAPIF 被置位，然后中断信号产生并发送到NVIC并告知CPU。
[4]	**CAPFUNCS**	**捕获功能选择** 0 =外部捕获模式使能

		1 =外部复位模式使能 **注：** 当CAPFUNCS为0，TMx _EXT (x= 0~3) 管脚上的变化用于保存24位定时器计数值(CNT) 到CAPDAT。 **注2:**当 CAPFUNCS为 1, TMx _EXT (x= 0~3)管脚上的变化用于复位24位定时器计数器值(CNT) 到 CAPDAT，然后CNT值立即复位。
[3]	**CAPEN**	**定时器外部捕获管脚使能** 该位使能TMx _EXT管脚捕获输入功能 0 = TMx _EXT (x= 0~3)管脚禁止 1 = TMx _EXT (x= 0~3)管脚使能
[2:1]	**Reserved**	保留
[0]	**CNTPHASE**	**定时器外部计数管脚相位检测选择** 该位表示TMx (x= 0~3)管脚相位检测。 0 =管脚的下降沿将被统计 1 =管脚的上升沿将被统计

定时器外部中断状态寄存器(TIMERx_EINTSTS)

寄存器	偏移地址	R/W	描述	复位值
TIMER0_EINTSTS	TMR01_BA+0x18	R/W	Timer0外部中断状态寄存器	0x0000_0000
TIMER1_EINTSTS	TMR01_BA+0x118	R/W	Timer1外部中断状态寄存器	0x0000_0000
TIMER2_EINTSTS	TMR23_BA+0x18	R/W	Timer2外部中断状态寄存器	0x0000_0000
TIMER3_EINTSTS	TMR23_BA+0x118	R/W	Timer3外部中断状态寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved							**CAPIF**

位	描述	
[31:1]	**Reserved**	保留
[0]	**CAPIF**	**定时器外部捕获中断标志位** 该位表示定时器外部捕获中断标志状态。 0 = TMx _EXT (x= 0~3)管脚中断没有发生 1 = TMx _EXT (x= 0~3)管脚中断发生 **注1**：该位写1清零； **注2**：当 CAPEN (TIMERx_EXTCTL[3]) 位被置1, CAPFUNCS (TIMERx_EXTCTL[4]) 位为0, TMx _EXT (x= 0~3) 管脚的变化与CAPEDGE (TIMERx_EXTCTL[2:1]) 设定匹配, 该位将被硬件置1； **注3:**CPU清CAPIF状态前会检测到一个新的捕获事件.如果以上条件发生，定时器将保持TIMERx_CAP的值不变并丢掉新的捕获值。

定时器触发控制寄存器(TIMERx_TRGCTL)

寄存器	偏移地址	R/W	描述	复位值
TIMER0_TRGCTL	TMR01_BA+0x1C	R/W	Timer0 触发控制寄存器	0x0000_0000
TIMER1_TRGCTL	TMR01_BA+0x11C	R/W	Timer1触发控制寄存器	0x0000_0000
TIMER2_TRGCTL	TMR23_BA+0x1C	R/W	Timer2触发控制寄存器	0x0000_0000
TIMER3_TRGCTL	TMR23_BA+0x11C	R/W	Timer3触发控制寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	3	4	3	2	1	0
Reserved			TRGPDMA	TRGDAC	TRGEADC	TRGPWM	TRGSSEL

位	描述	
[31:5]	**Reserved**	保留
[4]	**TRGPDMA**	**触发 PDMA 使能位** 如果该位置1，每一个定时器超时事件或捕获事件都可以触发PDMA传输 0 = 定时器中断触发 PDMA 禁止 1 = 定时器中断触发 PDMA 使能 **注：** 如果 TRGSSEL (TIMERx_TRGCTL[0]) = 0, 超时中断信号会触发PDMA传输 如果 TRGSSEL (TIMERx_TRGCTL[0]) = 1，捕获中断信号 PDMA 传输
[3]	**TRGDAC**	**触发 DAC 使能位** 如果该位置1，每一个定时器超时事件或捕获事件都可以触发DAC 0 = 定时器中断触发DAC禁止. 1 = 定时器中断触发DAC使能. **注：** 如果 TRGSSEL (TIMERx_TRGCTL[0]) = 0, 超时中断信号会触发DAC 如果 TRGSSEL (TIMERx_TRGCTL[0]) = 1，捕获中断信号DAC
[2]	**TRGEADC**	**触发 ADC 使能位** 如果该位置1，每一个定时器超时事件或捕获事件都可以触发ADC转换 0 = 定时器中断触发EADC禁止. 1 = 定时器中断触发EADC使能. **注：** 如果 TRGSSEL (TIMERx_TRGCTL[0]) = 0, 超时中断信号会触发EADC转换 如果 TRGSSEL (TIMERx_TRGCTL[0]) = 1，捕获中断信号ADC转换

[1]	**TRGPWM**	**触发EPWM和BPWM使能位** 如果该位置1，每一个定时器超时事件或捕获事件都可以作为EPWM计数器时钟源 0 = 定时器中断触发EPWM和BPWM禁止. 1 = 定时器中断触发EPWM和BPWM使能. **注：**如果 TRGSSEL (TIMERx_TRGCTL[0]) = 0，超时中断信号作为EPWM计数器时钟源 如果 TRGSSEL (TIMERx_TRGCTL[0]) = 1，捕获中断信号作为EPWM和BPWM计数器时钟源
[0]	**TRGSSEL**	**触发源选择位** 该位用于选择内部触发源来自定时器超时中断信号或捕获中断信号 0 = 超时中断信号用于内部触发EPWM、BPWM、PDMA和ADC 1 = 捕获中断信号用于内部触发EPWM、BPWM、PDMA和ADC

定时器选择控制寄存器(TIMERx_ALTCTL)

寄存器	偏移地址	R/W	描述	复位值
TIMER0_ALTCTL	TMR01_BA+0x20	R/W	Timer0 选择控制寄存器	0x0000_0000
TIMER1_ALTCTL	TMR01_BA+0x120	R/W	Timer1选择控制寄存器	0x0000_0000
TIMER2_ALTCTL	TMR23_BA+0x20	R/W	Timer2选择控制寄存器	0x0000_0000
TIMER3_ALTCTL	TMR23_BA+0x120	R/W	Timer3选择控制寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved							**FUNCSEL**

位	描述	
[31:1]	**Reserved**	保留
[0]	**FUNCSEL**	**功能选择** 0 = 定时器控制器用于定时器功能 1 =定时器控制器用于PWM功能. **注：** 当定时器用作 PWM，定时器控制器时钟源会被强制自动选为PCLKx。

定时器 PWM 控制寄存器(TIMERx_PWMCTL)

寄存器	偏移地址	R/W	描述	复位值
TIMER0_PWMCTL	TMR01_BA+0x40	R/W	Timer0 PWM 控制寄存器	0x0000_0000
TIMER1_PWMCTL	TMR01_BA+0x140	R/W	Timer1 PWM控制寄存器	0x0000_0000
TIMER2_PWMCTL	TMR23_BA+0x40	R/W	Timer2 PWM控制寄存器	0x0000_0000
TIMER3_PWMCTL	TMR23_BA+0x140	R/W	Timer3 PWM控制寄存器	0x0000_0000

31	30	29	28	27	26	25	24
DBGTRIOFF	DBGHALT	Reserved					
23	22	21	20	19	18	17	16
Reserved							OUTMODE
15	14	13	12	11	10	9	8
Reserved						IMMLDEN	CTRLD
7	6	5	4	3	2	1	0
Reserved				CNTMODE	CNTTYPE		CNTEN

位	描述	
[31]	DBGTRIOFF	**仿真器 (ICE) 调试模式响应禁止位（写保护位）** 0 = 仿真器调试模式响应影响PWM输出 当ICE调试模式响应时，PWM输出管脚被强制为三态模式。 1 = 仿真器调试模式响应禁用 无论ICE调试模式响应与否，PWM输出管脚保持输出。 **注：**该位为写保护位，详情请参考SYS_REGLCTL寄存器。
[30]	DBGHALT	**ICE调试模式计数器暂停 (写保护)** 如果调试模式计数器暂停使能，PWM计数器会保持当前值直到退出ICE调试模式。 0 = ICE调试模式计数器暂停禁止 1 = ICE 调试模式计数器暂停使能 **注：**该位为写保护位，详情请参考SYS_REGLCTL寄存器。
[29:17]	Reserved	保留
[16]	OUTMODE	**PWM 输出模式** 该位控制对应PWM通道输出模式 0 = PWM 独立模式 1 = PWM 互补模式.
[15:10]	Reserved	保留
[9]	IMMLDEN	**立即加载使能位** 0 = 当当前PWM周期完成，无论CTRLD是否使能，PERIOD加载到PBUF。如果CTRLD禁

		止，当当前PWM周期完成，CMP会加载到CMPBUF。如果CTRLD使能，在上下数计数模式，CMP会在当前周期的中心点加载到CMPBUF。 1 = 当用户更新PERIOD/CMP，PERIOD/CMP 会立即加载到 PBUF/CMPBUF **注:**如果IMMLDEN使能，CTRLD无效。
[8]	**CTRLD**	**中心重载** 在上下数计数模式，当当前PWM周期完成，PERIOD会加载到PBUF ，CMP会在当前周期的中心点加载到CMPBUF。
[7:4]	**Reserved**	保留
[3]	**CNTMODE**	**PWM 计数器模式** 0 = 自动重载模式. 1 = 单周期模式
[2:1]	**CNTTYPE**	**PWM 计数模式** 00 = 上计数模式. 01 = 下计数模式 10 = 上下计数模式 11 = 保留
[0]	**CNTEN**	**PWM 计数器使能位** 0 = PWM 计数器和时钟预分频停止运行 1 = PWM 计数器和时钟预分频开始运行

定时器PWM计数器时钟源寄存器(TIMERx PWMCLKSRC)

寄存器	偏移地址	R/W	描述	复位值
TIMER0_PWMCLKSRC	TMR01_BA+0x44	R/W	Timer0 PWM计数器时钟源寄存器	0x0000_0000
TIMER1_PWMCLKSRC	TMR01_BA+0x144	R/W	Timer1 PWM计数器时钟源寄存器	0x0000_0000
TIMER2_PWMCLKSRC	TMR23_BA+0x44	R/W	Timer2 PWM计数器时钟源寄存器	0x0000_0000
TIMER3_PWMCLKSRC	TMR23_BA+0x144	R/W	Timer3 PWM计数器时钟源寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved					**CLKSRC**		

位	描述	
[31:3]	**Reserved**	保留
[2:0]	**CLKSRC**	**PWM 计数器时钟源选择** PWM计数器时钟源可以选择来自TMRx_CLK或内部定时器超时或捕获事件 000 = TMRx_CLK 001 = 内部TIMER0 超时或捕获事件 010 =内部TIMER1超时或捕获事件 011 = 内部TIMER2超时或捕获事件 100 = 内部TIMER3超时或捕获事件 其他 = 保留

定时器 PWM 计数器时钟预分频寄存器(TIMERx PWMCLKPSC)

寄存器	偏移地址	R/W	描述	复位值
TIMER0_PWMCLKPSC	TMR01_BA+0x48	R/W	Timer0 PWM 计数器时钟预分频寄存器	0x0000_0000
TIMER1_PWMCLKPSC	TMR01_BA+0x148	R/W	Timer1 PWM计数器时钟预分频寄存器	0x0000_0000
TIMER2_PWMCLKPSC	TMR23_BA+0x48	R/W	Timer2 PWM计数器时钟预分频寄存器	0x0000_0000
TIMER3_PWMCLKPSC	TMR23_BA+0x148	R/W	Timer3 PWM计数器时钟预分频寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved				CLKPSC			
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
CLKPSC							

位	描述	
[31:12]	**Reserved**	保留
[11:0]	**CLKPSC**	**PWM 计数器时钟预分频** PWM计数器的实际时钟由计数器时钟预分频决定，时钟被 (CLKPSC + 1)分频。如果CLKPSC是0，PWM计数器时钟不分频

定时器 PWM 清计数器寄存器(TIMERx PWMCNTCLR)

寄存器	偏移地址	R/W	描述	复位值
TIMER0_PWMCNTCLR	TMR01_BA+0x4C	R/W	Timer0 PWM 清计数器寄存器	0x0000_0000
TIMER1_PWMCNTCLR	TMR01_BA+0x14C	R/W	Timer1 PWM清计数器寄存器	0x0000_0000
TIMER2_PWMCNTCLR	TMR23_BA+0x4C	R/W	Timer2 PWM清计数器寄存器	0x0000_0000
TIMER3_PWMCNTCLR	TMR23_BA+0x14C	R/W	Timer3 PWM清计数器寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved							CNTCLR

位	描述	
[31:1]	**Reserved**	保留
[0]	**CNTCLR**	**清 PWM 计数器控制位** 该位由硬件自动清除 0 = 无效 1 = 在上数和上下数计数模式，清 16位PWM计数器为0x10000；在下数计数模式复位计数器值为PERIOD

定时器 PWM 周期寄存器(TIMERx PWMPERIOD)

寄存器	偏移地址	R/W	描述	复位值
TIMER0_PWMPERIOD	TMR01_BA+0x50	R/W	Timer0 PWM周期寄存器	0x0000_0000
TIMER1_PWMPERIOD	TMR01_BA+0x150	R/W	Timer1 PWM周期寄存器	0x0000_0000
TIMER2_PWMPERIOD	TMR23_BA+0x50	R/W	Timer2 PWM周期寄存器	0x0000_0000
TIMER3_PWMPERIOD	TMR23_BA+0x150	R/W	Timer3 PWM周期寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
PERIOD							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
PERIOD							

位	描述	
[31:16]	**Reserved**	保留
[15:0]	**PERIOD**	**PWM 周期寄存器** 在上计数模式，PWM计数器从0计数到PERIOD，又从0重新开始。 在下计数模式，PWM计数器从PERIOD计数到0，又从PERIOD重新开始。 在上下计数模式，PWM计数器从0计数到PERIOD，然后再递减到0，如此反复。 在上计数和下计数模式 PWM 周期时间 = (PERIOD + 1) * (CLKPSC + 1) * TMRx_PWMCLK 在上下计数模式： PWM 周期时间 = 2 * PERIOD * (CLKPSC+ 1) * TMRx_PWMCLK **注：**在各种计数模式中，用户应该关注DIRF (TIMERx_PWMCNT[16])位来监视当前计数器的方向。

定时器 PWM 比较寄存器(TIMERx PWMCMPDAT)

寄存器	偏移地址	R/W	描述	复位值
TIMER0_PWMCMPDAT	TMR01_BA+0x54	R/W	Timer0 PWM比较寄存器	0x0000_0000
TIMER1_PWMCMPDAT	TMR01_BA+0x154	R/W	Timer1 PWM比较寄存器	0x0000_0000
TIMER2_PWMCMPDAT	TMR23_BA+0x54	R/W	Timer2 PWM比较寄存器	0x0000_0000
TIMER3_PWMCMPDAT	TMR23_BA+0x154	R/W	Timer3 PWM比较寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
CMP							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
CMP							

位	描述	
[31:16]	**Reserved**	保留
[15:0]	**CMP**	**PWM 比较寄存器** PWM CMP用于与PWM CNT比较产生PWM 输出波形、中断事件和触发ADC开始转换。

定时器 PWM 死区控制寄存器(TIMERx PWMDTCTL)

寄存器	偏移地址	R/W	描述	复位值
TIMER0_PWMDTCTL	TMR01_BA+0x58	R/W	Timer0 PWM死区控制寄存器	0x0000_0000
TIMER1_PWMDTCTL	TMR01_BA+0x158	R/W	Timer1 PWM死区控制寄存器	0x0000_0000
TIMER2_PWMDTCTL	TMR23_BA+0x58	R/W	Timer2 PWM死区控制寄存器	0x0000_0000
TIMER3_PWMDTCTL	TMR23_BA+0x158	R/W	Timer3 PWM死区控制寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							DTCKSEL
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							DTEN
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved				DTCNT			
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
DTCNT							

位	描述	
[31:25]	**Reserved**	保留
[24]	**DTCKSEL**	**死区时钟选择 (写保护)** 0 = 死区时钟源来自TMRx_PWMCLK，不带计数器时钟分频 1 = 死区时钟源来自TMRx_PWMCLK，带计数器时钟分频 **注：**该位为写保护位，详情请参考SYS_REGLCTL寄存器。
[23:17]	**Reserved**	保留
[16]	**DTEN**	**对 PWMx_CH0 和 PWMx_CH1使能死区插入 (写保护)** 仅当PWM互补模式使能，死区插入功能才有效。如果死区插入无效，PWMx_CH0 和PWMx_CH1是不带任何延时的互补输出。 0 = 死区插入禁止. 1 = 死区插入使能. **注：**该位为写保护位，详情请参考SYS_REGLCTL寄存器。
[15:12]	**Reserved**	保留
[11:0]	**DTCNT**	**死区计数器 (写保护)** 死区时间可以通过下面两个公式计算 死区时间= (DTCNT[11:0] + 1) * TMRx_PWMCLK（如果 DTCKSEL 是 0） 死区时间= (DTCNT[11:0] + 1) * TMRx_PWMCLK * (CLKPSC + 1)（如果 DTCKSEL 是 1） **注：**该位为写保护位，详情请参考SYS_REGLCTL寄存器。

定时器 PWM 计数器寄存器(TIMERx PWMCNT)

寄存器	偏移地址	R/W	描述	复位值
TIMER0_PWMCNT	TMR01_BA+0x5C	R	Timer0 PWM计数器寄存器	0x0000_0000
TIMER1_PWMCNT	TMR01_BA+0x15C	R	Timer1 PWM计数器寄存器	0x0000_0000
TIMER2_PWMCNT	TMR23_BA+0x5C	R	Timer2 PWM计数器寄存器	0x0000_0000
TIMER3_PWMCNT	TMR23_BA+0x15C	R	Timer3 PWM计数器寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							DIRF
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
CNT							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
CNT							

位	描述	
[31:17]	**Reserved**	保留
[16]	**DIRF**	**PWM 计数器方向指示标志(只读)** 0 = 计数器在下计数方向 1 = 计数器在上计数方向
[15:0]	**CNT**	**PWM 计数器值寄存器 (只读)** 用户可以读 CNT以知道当前计数器值是多少

定时器 PWM 输出屏蔽使能寄存器(TIMERx PWMMSKEN)

寄存器	偏移地址	R/W	描述	复位值
TIMER0_PWMMSKEN	TMR01_BA+0x60	R/W	Timer0 PWM输出屏蔽使能寄存器	0x0000_0000
TIMER1_PWMMSKEN	TMR01_BA+0x160	R/W	Timer1 PWM输出屏蔽使能寄存器	0x0000_0000
TIMER2_PWMMSKEN	TMR23_BA+0x60	R/W	Timer2 PWM输出屏蔽使能寄存器	0x0000_0000
TIMER3_PWMMSKEN	TMR23_BA+0x160	R/W	Timer3 PWM输出屏蔽使能寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved						**MSKEN1**	**MSKEN0**

位	描述	
[31:2]	**Reserved**	保留
[1]	**MSKEN1**	**PWMx_CH1 输出屏蔽使能位** 当该位使能，PWMx_CH1 输出信号被屏蔽，PWMx_CH1 输出 MSKDAT1 (TIMER_PWMMSK[1]) 数据。 0 = PWMx_CH1输出信号不屏蔽 1 = PWMx_CH1 输出信号屏蔽且输出MSKDAT1 数据
[0]	**MSKEN0**	**PWMx_CH0输出屏蔽使能位** 当该位使能，PWMx_CH0 输出信号被屏蔽，PWMx_CH0 输出 MSKDAT0 (TIMER_PWMMSK[0]) 数据。 0 = PWMx_CH0输出信号不屏蔽 1 = PWMx_CH0 输出信号屏蔽且输出MSKDAT0 数据

定时器 PWM 输出屏蔽数据控制寄存器(TIMERx PWMMSK)

寄存器	偏移地址	R/W	描述	复位值
TIMER0_PWMMSK	TMR01_BA+0x64	R/W	Timer0 PWM 输出屏蔽数据控制寄存器	0x0000_0000
TIMER1_PWMMSK	TMR01_BA+0x164	R/W	Timer1 PWM 输出屏蔽数据控制寄存器	0x0000_0000
TIMER2_PWMMSK	TMR23_BA+0x64	R/W	Timer2 PWM 输出屏蔽数据控制寄存器	0x0000_0000
TIMER3_PWMMSK	TMR23_BA+0x164	R/W	Timer3 PWM 输出屏蔽数据控制寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved						**MSKDAT1**	**MSKDAT0**

位	描述	
[31:2]	**Reserved**	保留
[1]	**MSKDAT1**	**PWMx_CH1 输出屏蔽数据控制位** 当PWMx_CH1输出屏蔽功能使能(MSKEN1 = 1)，该位用于控制PWMx_CH1管脚的输出状态。 0 = PWMx_CH1输出逻辑低 1 = PWMx_CH1输出逻辑高
[0]	**MSKDAT0**	**PWMx_CH0输出屏蔽数据控制位** 当PWMx_CH0输出屏蔽功能使能(MSKEN0 = 1)，该位用于控制PWMx_CH0管脚的输出状态。 0 = PWMx_CH0输出逻辑低 1 = PWMx_CH0输出逻辑高

定时器 PWM 刹车管脚噪音滤波寄存器(TIMERx PWMBNF)

寄存器	偏移地址	R/W	描述	复位值
TIMER0_PWMBNF	TMR01_BA+0x68	R/W	Timer0 PWM刹车管脚噪音滤波寄存器	0x0000_0000
TIMER1_PWMBNF	TMR01_BA+0x168	R/W	Timer1 PWM刹车管脚噪音滤波寄存器	0x0000_0000
TIMER2_PWMBNF	TMR23_BA+0x68	R/W	Timer2 PWM刹车管脚噪音滤波寄存器	0x0000_0000
TIMER3_PWMBNF	TMR23_BA+0x168	R/W	Timer3 PWM刹车管脚噪音滤波寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved						BKPINSRC	
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
BRKPINV	BRKFCNT			BRKNFSEL			BRKNFEN

位	描述	
[31:18]	**Reserved**	保留
[17:16]	**BKPINSRC**	**刹车管脚源选择** 00 =刹车管脚源来自 PWM0_BRAKE0 管脚 01 =刹车管脚源来自 PWM0_BRAKE1 管脚 10 =刹车管脚源来自 PWM1_BRAKE0 管脚 11 =刹车管脚源来自 PWM1_BRAKE1 管脚
[15:8]	**Reserved**	保留
[7]	**BRKPINV**	**刹车管脚侦测控制位** 0 =在边沿侦测模式， PWMx_BRAKEy管脚状态从低到高跳变，或在电平侦测模式，管脚状态是高，则刹车管脚事件会被侦测到。 1 =在边沿侦测模式， PWMx_BRAKEy管脚状态从高到低跳变，或在电平侦测模式，管脚状态是低，则刹车管脚事件会被侦测到。
[6:4]	**BRKFCNT**	**刹车管脚噪音滤波计数** 该域用于控制噪音滤波采样时间 一次噪音滤波采样时间 = (BRKDBCS的周期时间) * BRKFCNT
[3:1]	**BRKNFSEL**	**刹车管脚噪音滤波时钟选择** 000 = 噪音滤波时钟是 PCLKx 001 =噪音滤波时钟是PCLKx/2 010 =噪音滤波时钟是PCLKx/4

		011 =噪音滤波时钟是PCLKx/8 100 =噪音滤波时钟是PCLKx/16 101 =噪音滤波时钟是PCLKx/32 110 =噪音滤波时钟是PCLKx/64 111 =噪音滤波时钟是PCLKx/128
[0]	**BRKNFEN**	**刹车管脚噪音滤波使能位** 0 = PWMx_BRAKEy 管脚噪音滤波侦测禁止 1 = PWMx_BRAKEy 管脚噪音滤波侦测使能

定时器PWM 系统失败刹车控制寄存器(TIMERx PWMFAILBRK)

寄存器	偏移地址	R/W	描述	复位值
TIMER0_PWMFAILBRK	TMR01_BA+0x6C	R/W	Timer0 PWM系统失败刹车控制寄存器	0x0000_0000
TIMER1_PWMFAILBRK	TMR01_BA+0x16C	R/W	Timer1 PWM系统失败刹车控制寄存器	0x0000_0000
TIMER2_PWMFAILBRK	TMR23_BA+0x6C	R/W	Timer2 PWM系统失败刹车控制寄存器	0x0000_0000
TIMER3_PWMFAILBRK	TMR23_BA+0x16C	R/W	Timer3 PWM系统失败刹车控制寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
7	**6**	**2**	**4**	**3**	**2**	**1**	**0**
Reserved				**CORBRKEN**	**RAMBRKEN**	**BODBRKEN**	**CSSBRKEN**

位	描述	
[31:4]	**Reserved**	保留
[3]	**CORBRKEN**	**内核锁死侦测触发PWM刹车功能使能位** 0 = 禁止刹车功能由内核锁死事件触发 1 = 使能刹车功能由内核锁死事件触发
[2]	**RAMBRKEN**	**SRAM 校验错误侦测触发PWM刹车功能使能位** 0 = 禁止刹车功能由SRAM校验错误侦测触发 1 = 使能刹车功能由SRAM校验错误侦测触发
[1]	**BODBRKEN**	**欠压侦测触发PWM刹车功能使能位** 0 = 禁止刹车功能由BOD事件触发 1 =使能刹车功能由BOD事件触发
[0]	**CSSBRKEN**	**时钟安全系统侦测侦测触发PWM刹车功能使能位** 0 = 禁止刹车功能由时钟失败侦测触发 1 = 使能刹车功能由时钟失败侦测触发

定时器 PWM 刹车控制寄存器(TIMERx PWMBRKCTL)

寄存器	偏移地址	R/W	描述	复位值
TIMER0_PWMBRKCTL	TMR01_BA+0x70	R/W	Timer0 PWM刹车控制寄存器	0x0000_0000
TIMER1_PWMBRKCTL	TMR01_BA+0x170	R/W	Timer1 PWM刹车控制寄存器	0x0000_0000
TIMER2_PWMBRKCTL	TMR23_BA+0x70	R/W	Timer2 PWM刹车控制寄存器	0x0000_0000
TIMER3_PWMBRKCTL	TMR23_BA+0x170	R/W	Timer3 PWM刹车控制寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved				BRKAODD		BRKAEVEN	
15	14	13	12	11	10	9	8
SYSLBEN	Reserved		BRKPLEN	Reserved		CPO1LBEN	CPO0LBEN
7	6	5	4	3	2	1	0
SYSEBEN	Reserved		BRKPEEN	Reserved		CPO1EBEN	CPO0EBEN

位	描述	
[31:20]	**Reserved**	保留
[19:18]	**BRKAODD**	**PWMx_CH1的PWM刹车动作选择 (写保护)** 00 = PWMx_BRAKEy刹车事件不会影响PWMx_CH1 输出 01 = 当PWMx_BRAKEy 刹车事件发生，PWMx_CH1 输出三态 10 =当PWMx_BRAKEy 刹车事件发生，PWMx_CH1 输出低 11 =当PWMx_BRAKEy 刹车事件发生，PWMx_CH1 输出高 **注：**该位为写保护位，详情请参考SYS_REGLCTL寄存器
[17:16]	**BRKAEVEN**	**PWMx_CH0的PWM刹车动作选择 (写保护)** 00 = PWMx_BRAKEy刹车事件不会影响PWMx_CH0 输出 01 = 当PWMx_BRAKEy 刹车事件发生，PWMx_CH0 输出三态 10 =当PWMx_BRAKEy 刹车事件发生，PWMx_CH0 输出低 11 =当PWMx_BRAKEy 刹车事件发生，PWMx_CH0 输出高 **注：**该位为写保护位，详情请参考SYS_REGLCTL寄存器
[15]	**SYSLBEN**	**使能系统失败作为电平侦测刹车源 (写保护)** 0 = 系统失败条件作为电平侦测刹车源禁止 1 =系统失败条件作为电平侦测刹车源使能 **注：**该位为写保护位，详情请参考SYS_REGLCTL寄存器。
[14:13]	**Reserved**	保留
[12]	**BRKPLEN**	**使能 TM_BRAKEx 管脚作为电平侦测刹车源 (写保护)**

		0 = PWMx_BRAKEy 管脚事件作为电平侦测刹车源禁止 1 = PWMx_BRAKEy管脚事件作为电平侦测刹车源使能. **注：** 该位为写保护位，详情请参考SYS_REGLCTL寄存器。
[11:10]	**Reserved**	保留
[9]	**CPO1LBEN**	**使能内部 ACMP1_O数字输出作为电平侦测刹车源(写保护)** 0 = 内部 ACMP1_O 信号作为电平侦测刹车源禁止 1 = 内部 ACMP1_O信号作为电平侦测刹车源使能 **注1：** 作为刹车事件仅内部 ACMP1_O信号从低到高会被侦测到； **注2：** 该位为写保护位，详情请参考SYS_REGLCTL寄存器。
[8]	**CPO0LBEN**	**使能内部 ACMP0_O数字输出作为电平侦测刹车源(写保护)** 0 = 内部 ACMP0_O 信号作为电平侦测刹车源禁止 1 = 内部 ACMP0_O信号作为电平侦测刹车源使能 **注1：** 作为刹车事件仅内部 ACMP0_O信号从低到高会被侦测到； **注2：** 该位为写保护位，详情请参考SYS_REGLCTL寄存器。
[7]	**SYSEBEN**	**使能系统失败作为边沿侦测刹车源(写保护)** 0 = 系统失败条件作为边沿侦测刹车源禁止 1 =系统失败条件作为边沿侦测刹车源使能 **注：** 该位为写保护位，详情请参考SYS_REGLCTL寄存器。
[6:5]	**Reserved**	保留
[4]	**BRKPEEN**	**使能 TM_BRAKEx 管脚作为边沿侦测刹车源 (写保护)** 0 = PWMx_BRAKEy 管脚事件作为边沿侦测刹车源禁止 1 = PWMx_BRAKEy管脚事件作为边沿侦测刹车源使能 **注：** 该位为写保护位，详情请参考SYS_REGLCTL寄存器。
[3:2]	**Reserved**	保留
[1]	**CPO1EBEN**	**使能内部 ACMP1_O数字输出作为边沿侦测刹车源(写保护)** 0 = 内部 ACMP1_O 信号作为边沿侦测刹车源禁止 1 = 内部 ACMP1_O信号作为边沿侦测刹车源使能 **注1：** 作为刹车事件仅内部 ACMP1_O信号从低到高会被侦测到； **注2：** 该位为写保护位。详情请参考SYS_REGLCTL 寄存器。
[0]	**CPO0EBEN**	**使能内部 ACMP0_O数字输出作为边沿侦测刹车源(写保护)** 0 = 内部 ACMP0_O 信号作为边沿侦测刹车源禁止 1 = 内部 ACMP0_O信号作为边沿侦测刹车源使能 **注1：** 作为刹车事件仅内部 ACMP0_O信号从低到高会被侦测到； **注2：** 该位为写保护位。详情请参考SYS_REGLCTL 寄存器。

定时器 PWM 管脚输出极性控制寄存器(TIMERx PWMPOLCTL)

寄存器	偏移地址	R/W	描述	复位值
TIMER0_PWMPOLCTL	TMR01_BA+0x74	R/W	Timer0 PWM管脚输出极性控制寄存器	0x0000_0000
TIMER1_PWMPOLCTL	TMR01_BA+0x174	R/W	Timer1 PWM管脚输出极性控制寄存器	0x0000_0000
TIMER2_PWMPOLCTL	TMR23_BA+0x74	R/W	Timer2 PWM管脚输出极性控制寄存器	0x0000_0000
TIMER3_PWMPOLCTL	TMR23_BA+0x174	R/W	Timer3 PWM管脚输出极性控制寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved						PINV1	PINV0

位	描述	
[31:2]	**Reserved**	保留
[1]	**PINV1**	**PWMx_CH1 输出管脚极性控制位** 该位用于控制PWMx_CH1输出的极性状态 0 = PWMx_CH1输出管脚极性反转禁止 1 = PWMx_CH1输出管脚极性反转使能.
[0]	**PINV0**	**PWMx_CH0 输出管脚极性控制位** 该位用于控制PWMx_CH0输出的极性状态 0 = PWMx_CH0输出管脚极性反转禁止 1 = PWMx_CH0输出管脚极性反转使能.

定时器 PWM 管脚输出使能寄存器(TIMERx PWMPOEN)

寄存器	偏移地址	R/W	描述	复位值
TIMER0_PWMPOEN	TMR01_BA+0x78	R/W	Timer0 PWM管脚输出使能寄存器	0x0000_0000
TIMER1_PWMPOEN	TMR01_BA+0x178	R/W	Timer1 PWM管脚输出使能寄存器	0x0000_0000
TIMER2_PWMPOEN	TMR23_BA+0x78	R/W	Timer2 PWM管脚输出使能寄存器	0x0000_0000
TIMER3_PWMPOEN	TMR23_BA+0x178	R/W	Timer3 PWM管脚输出使能寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved						**POEN1**	**POEN0**

位	描述	
[31:2]	**Reserved**	保留
[1]	**POEN1**	**PWMx_CH1 输出管脚使能位** 0 = PWMx_CH1 管脚在三态模式 1 = PWMx_CH1 管脚在输出模式
[0]	**POEN0**	**PWMx_CH0 输出管脚使能位** 0 = PWMx_CH0 管脚在三态模式 1 = PWMx_CH0 管脚在输出模式

定时器 PWM 软件触发刹车控制寄存器(TIMERx PWMSWBRK)

寄存器	偏移地址	R/W	描述	复位值
TIMER0_PWMSWBRK	TMR01_BA+0x7C	W	Timer0 PWM软件触发刹车控制寄存器	0x0000_0000
TIMER1_PWMSWBRK	TMR01_BA+0x17C	W	Timer1 PWM软件触发刹车控制寄存器	0x0000_0000
TIMER2_PWMSWBRK	TMR23_BA+0x7C	W	Timer2 PWM软件触发刹车控制寄存器	0x0000_0000
TIMER3_PWMSWBRK	TMR23_BA+0x17C	W	Timer3 PWM软件触发刹车控制寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							**BRKLTRG**
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved							**BRKETRG**

位	描述	
[31:9]	**Reserved**	保留
[8]	**BRKLTRG**	**软件触发电平侦测刹车源(只写) (写保护)** 写 1 到该位触发PWM电平侦测刹车源，然后TIMERx_PWMINTSTS1 寄存器中BRKLIF0 和 BRKLIF1 会自动置1 。 **注：**该位为写保护位，详情请参考SYS_REGLCTL寄存器。
[7:1]	**Reserved**	保留
[0]	**BRKETRG**	**软件触发边沿侦测刹车源(只写) (写保护)** 写 1 到该位触发PWM边沿侦测刹车源，然后TIMERx_PWMINTSTS1 寄存器中BRKEIF0和BRKEIF1会自动置1 。 **注：**该位为写保护位，详情请参考SYS_REGLCTL寄存器。

定时器 PWM 中断使能寄存器 0(TIMERx PWMINTEN0)

寄存器	偏移地址	R/W	描述	复位值
TIMER0_PWMINTEN0	TMR01_BA+0x80	R/W	Timer0 PWM中断使能寄存器 0	0x0000_0000
TIMER1_PWMINTEN0	TMR01_BA+0x180	R/W	Timer1 PWM中断使能寄存器 0	0x0000_0000
TIMER2_PWMINTEN0	TMR23_BA+0x80	R/W	Timer2 PWM中断使能寄存器 0	0x0000_0000
TIMER3_PWMINTEN0	TMR23_BA+0x180	R/W	Timer3 PWM中断使能寄存器 0	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved				CMPDIEN	CMPUIEN	PIEN	ZIEN

位	描述	
[31:4]	**Reserved**	保留
[3]	**CMPDIEN**	**PWM下数比较中断使能位** 0 = 下数比较中断禁止. 1 = 下数比较中断使能
[2]	**CMPUIEN**	**PWM上数比较中断使能位** 0 = 上数比较中断禁止. 1 = 上数比较中断使能
[1]	**PIEN**	**PWM 周期点中断使能位** 0 = 周期点中断禁止 1 = 周期点中断使能 **注:**在上下计数模式中，周期点就是当前PWM周期的中心点的意思。
[0]	**ZIEN**	**PWM 零点中断使能** 0 = 零点中断禁止. 1 =零点中断使能

定时器 PWM 中断使能寄存器 1 (TIMERx PWMINTEN1)

寄存器	偏移地址	R/W	描述	复位值
TIMER0_PWMINTEN1	TMR01_BA+0x84	R/W	Timer0 PWM中断使能寄存器1	0x0000_0000
TIMER1_PWMINTEN1	TMR01_BA+0x184	R/W	Timer1 PWM中断使能寄存器1	0x0000_0000
TIMER2_PWMINTEN1	TMR23_BA+0x84	R/W	Timer2 PWM中断使能寄存器1	0x0000_0000
TIMER3_PWMINTEN1	TMR23_BA+0x184	R/W	Timer3 PWM中断使能寄存器1	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							BRKLIEN
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved							BRKEIEN

位	描述	
[31:9]	**Reserved**	保留
[8]	**BRKLIEN**	**PWM 电平侦测刹车中断使能 (写保护)** 0 = PWM 电平侦测刹车中断禁止. 1 = PWM电平侦测刹车中断使能. **注：**该位为写保护位，详情请参考SYS_REGLCTL寄存器。
[7:1]	**Reserved**	保留
[0]	**BRKEIEN**	**PWM 边沿侦测刹车中断使能 (写保护)** 0 = PWM 边沿侦测刹车中断禁止. 1 = PWM边沿侦测刹车中断使能. **注：**该位为写保护位，详情请参考SYS_REGLCTL寄存器。

Timer PWM中断状态寄存器0 (TIMERx PWMINTSTS0)

寄存器	偏移地址	R/W	描述	复位值
TIMER0_PWMINTSTS0	TMR01_BA+0x88	R/W	Timer0 PWM 中断状态寄存器0	0x0000_0000
TIMER1_PWMINTSTS0	TMR01_BA+0x188	R/W	Timer1 PWM中断状态寄存器0	0x0000_0000
TIMER2_PWMINTSTS0	TMR23_BA+0x88	R/W	Timer2 PWM中断状态寄存器0	0x0000_0000
TIMER3_PWMINTSTS0	TMR23_BA+0x188	R/W	Timer3 PWM中断状态寄存器0	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved				CMPDIF	CMPUIF	PIF	ZIF

位	描述	
[31:4]	Reserved	保留
[3]	CMPDIF	**PWM下数比较中断标志** 当TIMERx_PWM计数器在下计数方向且计数到CMP比较值，该位由硬件置位。 **注1：** 如果 CMP 等于 PERIOD, 在下计数模式没有 CMPDIF 标志； **注2：** 该位写1清0。
[2]	CMPUIF	**PWM上数比较中断标志** 当TIMERx_PWM计数器在上计数方向且计数到CMP比较值，该位由硬件置位。 **注1：** 如果 CMP 等于 PERIOD, 在上计数模式和上下计数模式没有 CMPDIF 标志； **注2：** 该位写1清0。
[1]	PIF	**PWM 周期点中断标志** 当TIMERx_PWM计数器计数到PERIOD，该位由硬件置位。 **注1：** 在上下计数模式，PIF标志是当前PWM周期中心点标志的意思； **注2：** 该位写1清0。
[0]	ZIF	**PWM 零点中断标志** 当TIMERx_PWM计数器计数到0，该位由硬件置位。 **注：** 该位写1清0。

定时器 PWM 中断状态寄存器 1(TIMERx PWMINTSTS1)

寄存器	偏移地址	R/W	描述	复位值
TIMER0_PWMINTSTS1	TMR01_BA+0x8C	R/W	Timer0 PWM中断状态寄存器1	0x0000_0000
TIMER1_PWMINTSTS1	TMR01_BA+0x18C	R/W	Timer1 PWM中断状态寄存器1	0x0000_0000
TIMER2_PWMINTSTS1	TMR23_BA+0x8C	R/W	Timer2 PWM中断状态寄存器1	0x0000_0000
TIMER3_PWMINTSTS1	TMR23_BA+0x18C	R/W	Timer3 PWM中断状态寄存器1	0x0000_0000

31	30	29	28	27	26	25	24
Reserved						BRKLSTS1	BRKLSTS0
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved						BRKESTS1	BRKESTS0
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved						BRKLIF1	BRKLIF0
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved						BRKEIF1	BRKEIF0

位	描述	
[31:26]	**Reserved**	保留
[25]	**BRKLSTS1**	**PWMx_CH1的电平侦测刹车状态 (只读)** 0 = PWMx_CH1电平侦测刹车状态释放 1 = PWMx_CH1在电平侦测刹车状态 **注：** 如果TIMERx_PWM电平侦测刹车源释放，当当前PWM周期完成，PWMx_CH0 和 PWMx_CH1 都会释放刹车状态，且PWMx_CH0 和 PWMx_CH1 从下一个完整PWM周期开始重新开始输出波形。
[24]	**BRKLSTS0**	**PWMx_CH0的电平侦测刹车状态 (只读)** 0 = PWMx_CH0电平侦测刹车状态释放 1 = PWMx_CH0 在电平侦测刹车状态 **注：** 如果TIMERx_PWM电平侦测刹车源释放，当当前PWM周期完成，PWMx_CH0 和 PWMx_CH0 都会释放刹车状态，且PWMx_CH0 和 PWMx_CH1 从下一个完整PWM周期开始重新开始输出波形。
[23:18]	**Reserved**	保留
[17]	**BRKESTS1**	**PWMx_CH1的边沿侦测刹车状态 (只读)** 0 = PWMx_CH1 边沿侦测刹车状态释放. 1 = PWMx_CH1 在边沿侦测刹车状态 **注：** 用户可以设置 BRKEIF1 为1来清除BRKEIF1 标志，当当前PWM周期完成，PWMx_CH1 会释放刹车状态，且PWMx_CH1 从下一个完整PWM周期开始重新开始输出波形。
[16]	**BRKESTS0**	**PWMx_CH0的边沿侦测刹车状态 (只读)**

		0 = PWMx_CH0 边沿侦测刹车状态释放. 1 = PWMx_CH0 在边沿侦测刹车状态 **注：** 用户可以设置 BRKEIF0 为1来清除BRKEIF0标志，当当前PWM周期完成，PWMx_CH0 会释放刹车状态，且PWMx_CH0 从下一个完整PWM周期开始重新开始输出波形。
[15:10]	**Reserved**	保留
[9]	**BRKLIF1**	**PWMx_CH1 的电平侦测刹车中断标志(写保护)** 0 = PWMx_CH1 电平侦测刹车事件没有发生 1 = PWMx_CH1 发生了电平侦测刹车事件 **注1：** 该位写1清0。 **注2：** 该位为写保护位，详情请参考SYS_REGLCTL寄存器。
[8]	**BRKLIF0**	**PWMx_CH0 的电平侦测刹车中断标志(写保护)** 0 = PWMx_CH0 电平侦测刹车事件没有发生 1 = PWMx_CH0 发生了电平侦测刹车事件 **注1：** 该位写1清0; **注2：** 该位为写保护位，详情请参考SYS_REGLCTL寄存器。
[7:2]	**Reserved**	保留
[1]	**BRKEIF1**	**PWMx_CH1 的边沿侦测刹车中断标志(写保护)** 0 = PWMx_CH1 边沿侦测刹车事件没有发生 1 = PWMx_CH1 发生了边沿侦测刹车事件 **注1：** 该位写1清0。 **注2:**该位为写保护位，详情请参考SYS_REGLCTL寄存器。
[0]	**BRKEIF0**	**PWMx_CH0 的边沿侦测刹车中断标志(写保护)** 0 = PWMx_CH0 边沿侦测刹车事件没有发生 1 = PWMx_CH0 发生了边沿侦测刹车事件 **注1：** 该位写1清0。 **注2：** 该位为写保护位，详情请参考SYS_REGLCTL寄存器。

定时器 PWM ADC 触发控制寄存器(TIMERx PWMADCTS)

寄存器	偏移地址	R/W	描述	复位值
TIMER0_PWMADCTS	TMR01_BA+0x90	R/W	Timer0 PWM ADC触发源选择寄存器	0x0000_0000
TIMER1_PWMADCTS	TMR01_BA+0x190	R/W	Timer1 PWM ADC触发源选择寄存器	0x0000_0000
TIMER2_PWMADCTS	TMR23_BA+0x90	R/W	Timer2 PWM ADC触发源选择寄存器	0x0000_0000
TIMER3_PWMADCTS	TMR23_BA+0x190	R/W	Timer3 PWM ADC触发源选择寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
TRGEN	**Reserved**			**TRGSEL**			

位	描述	
[31:8]	**Reserved**	保留
[7]	**TRGEN**	**PWM 计数器事件触发ADC转换使能位** 0 = PWM 计数器事件触发ADC 转换禁止 1 = PWM计数器事件触发ADC 转换使能
[6:3]	**Reserved**	保留
[2:0]	**TRGSEL**	**PWM 计数器事件源选择来触发ADC转换** 000 = 触发EADC 转换在零点 (ZIF) 001 =触发EADC 转换在周期点 (PIF) 010 =触发EADC 转换在零点或周期点 (ZIF或PIF) 011 =触发EADC 转换在上数比较点 (CMPUIF) 100 =触发EADC 转换在下数比较点 (CMPDIF) 其他 = 保留

定时器 PWM 同步控制寄存器(TIMERx PWMSCTL)

寄存器	偏移地址	R/W	描述	复位值
TIMER0_PWMSCTL	TMR01_BA+0x94	R/W	Timer0 PWM同步控制寄存器	0x0000_0000
TIMER1_PWMSCTL	TMR01_BA+0x194	R/W	Timer1 PWM同步控制寄存器	0x0000_0000
TIMER2_PWMSCTL	TMR23_BA+0x94	R/W	Timer2 PWM同步控制寄存器	0x0000_0000
TIMER3_PWMSCTL	TMR23_BA+0x194	R/W	Timer3 PWM同步控制寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							**SYNCSRC**
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved						**SYNCMODE**	

位	描述	
[31:9]	**Reserved**	保留
[8]	**SYNCSRC**	**PWM 同步计数器启动/清除动作触发源选择** 0 = 计数器由TIMER0_PWMSTRG STRGEN触发同步启动/清除 1 =计数器由TIMER2_PWMSTRG STRGEN触发同步启动/清除 **注1：** 如果 TIMER0/1/2/3 PWM 计数器同步源来自TIMER0，那么TIMER0_PWMSCTL[8]、TIMER1_PWMSCTL[8]、TIMER2_PWMSCTL[8] 和 TIMER3_PWMSCTL[8] 应该要为 0； **注2：** 如果 TIMER0/1/ PWM 计数器同步源来自TIMER0，那么 TIMER0_PWMSCTL[8]和TIMER1_PWMSCTL[8]应该置 0。如果TIMER2/3/ PWM 计数器同步源来自 TIMER2, TIME2_PWMSCTL[8] 和 TIMER3_PWMSCTL[8] 应该置1。
[7:2]	**Reserved**	保留
[1:0]	**SYNCMODE**	**PWM 同步模式使能选择** 00 = PWM 同步功能禁止. 01 = PWM计数器同步启动功能使能 10 = 保留 11 = PWM 计数器同步清除功能使能

定时器 PWM 同步触发寄存器(TIMERx PWMSTRG)

寄存器	偏移地址	R/W	描述	复位值
TIMER0_PWMSTRG	TMR01_BA+0x98	W	Timer0 PWM 同步触发寄存器	0x0000_0000
TIMER2_PWMSTRG	TMR23_BA+0x98	W	Timer2 PWM同步触发寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved							**STRGEN**

位	描述	
[31:1]	**Reserved**	保留
[0]	**STRGEN**	**PWM 计数器同步触发使能位(只写)** PMW 计数器同步功能用于对选择的PWM通道（包括TIMER0/1/2/3 PWM, TIMER0/1 PWM 和 TIMER2/3 PWM）计数器根据TIMERx_PWMSCTL 的设定在相同的时刻启动或清除计数器。 **注：**该位仅对 TIMER0 和 TIMER2有效。

定时器 PWM 状态寄存器(TIMERx PWMSTATUS)

寄存器	偏移地址	R/W	描述	复位值
TIMER0_PWMSTATUS	TMR01_BA+0x9C	R/W	Timer0 PWM状态寄存器	0x0000_0000
TIMER1_PWMSTATUS	TMR01_BA+0x19C	R/W	Timer1 PWM状态寄存器	0x0000_0000
TIMER2_PWMSTATUS	TMR23_BA+0x9C	R/W	Timer2 PWM状态寄存器	0x0000_0000
TIMER3_PWMSTATUS	TMR23_BA+0x19C	R/W	Timer3 PWM状态寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							**EADCTRGF**
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved							**CNTMAXF**

位	描述	
[31:17]	**Reserved**	保留
[16]	**EADCTRGF**	**触发 ADC 开始转换标志** 0 =未发生EADC计数器事件触发ADC开始转换 1 = 发生了EADC计数器事件触发ADC开始转换. **注：** 该位写1清0。
[15:1]	**Reserved**	保留
[0]	**CNTMAXF**	**PWM 计数器等于 0xFFFF 标志** 0 = 表示 PWM 计数器值没有到达它的最大值0xFFFF 1 = 表示 PWM 计数器值到达它的最大值 **注：** 该位写1清0。

定时器 PWM 周期值缓存寄存器(TIMERx PWMPBUF)

寄存器	偏移地址	R/W	描述	复位值
TIMER0_PWMPBUF	TMR01_BA+0xA0	R	Timer0 PWM周期值缓存寄存器	0x0000_0000
TIMER1_PWMPBUF	TMR01_BA+0x1A0	R	Timer1 PWM周期值缓存寄存器	0x0000_0000
TIMER2_PWMPBUF	TMR23_BA+0xA0	R	Timer2 PWM周期值缓存寄存器	0x0000_0000
TIMER3_PWMPBUF	TMR23_BA+0x1A0	R	Timer3 PWM周期值缓存寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
PBUF							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
PBUF							

位	描述	
[31:16]	**Reserved**	保留
[15:0]	**PBUF**	**PWM 周期值缓存寄存器(只读)** 用于使PERIOD生效

定时器 PWM 比较值缓存寄存器(TIMERx PWMCMPBUF)

寄存器	偏移地址	R/W	描述	复位值
TIMER0_PWMCMPBUF	TMR01_BA+0xA4	R	Timer0 PWM比较值缓存寄存器	0x0000_0000
TIMER1_PWMCMPBUF	TMR01_BA+0x1A4	R	Timer1 PWM比较值缓存寄存器	0x0000_0000
TIMER2_PWMCMPBUF	TMR23_BA+0xA4	R	Timer2 PWM比较值缓存寄存器	0x0000_0000
TIMER3_PWMCMPBUF	TMR23_BA+0x1A4	R	Timer3 PWM比较值缓存寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
CMPBUF							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
CMPBUF							

位	描述	
[31:16]	**Reserved**	保留
[15:0]	**CMPBUF**	**PWM 周期值缓存寄存器(只读)** 用于使CMP生效

6.9 看门狗定时器(WDT)

6.9.1 概述

设计看门狗定时器的目的是，当系统运行到一个未知状态时，通过它来使系统复位。这种做法可以预防系统进入到无限期的死循环。此外，该看门狗定时器支持系统从空闲/掉电模式唤醒。

6.9.2 特性

- 18位的向上看门狗定时器可满足用户溢出时间间隔要求
- 溢出时间间隔(2^4 ~ 2^{18})个WDT_CLK时钟周期可选，如WDT_CLK = 10 kHz，那么溢出时间间隔是1.6 ms ~ 26.214 s
- 系统复位保持时间(1 / WDT_CLK) * 63
- 支持看门狗定时器复位延时周期, 包括 1026、130、18 或 3 个WDT_CLK的复位延时时间
- 通过设置CONFIG0中CWDTEN[2:0]位，支持芯片上电或复位条件下看门狗强制打开

- 如果时钟源选择内部低速10k时钟或LXT时钟，支持看门狗定时器溢出唤醒

6.9.3 框图

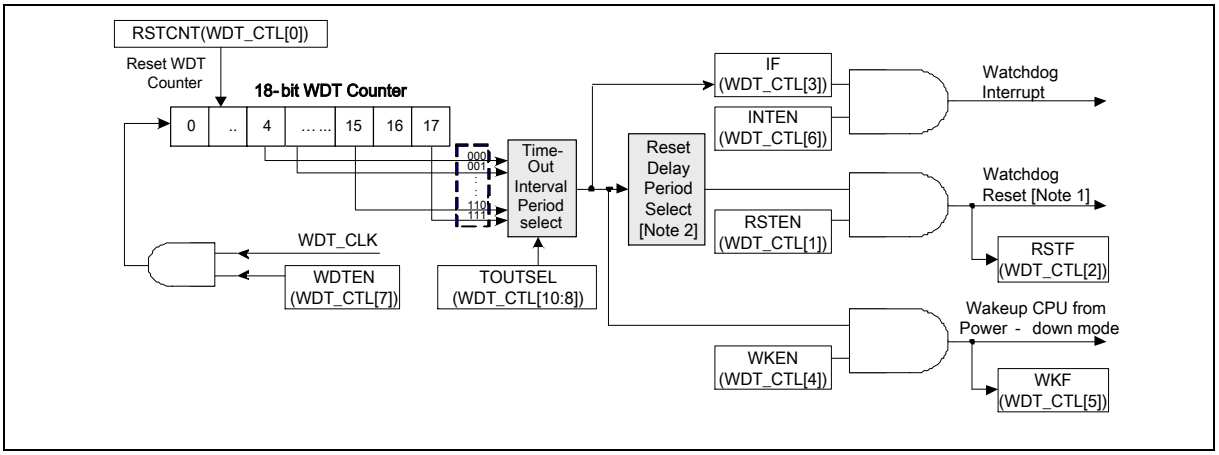


图 6.9-1 看门狗定时器框图

注1：看门狗定时器复位CPU将持续63 个WDT_CLK；

注2：看门狗定时器复位延时周期可以选择3/18/130/1026 个WDT_CLK。

6.9.4 基本配置

- 时钟源配置
  - 通过WDTSEL (CLK_CLKSEL1[1:0])选择 WDT 外设时钟源
  - 通过 WDTCKEN (CLK_APBCLK0[0])使能WDT外设时钟
  - 通过CWDTEN[2:0] (CWDTEN[2] 是 Config0[31], CWDTEN[1:0] 是 Config0[4:3])配置在芯片上电或复位后强制使能 WDT 控制器

WDT 时钟控制如图6.9-2。

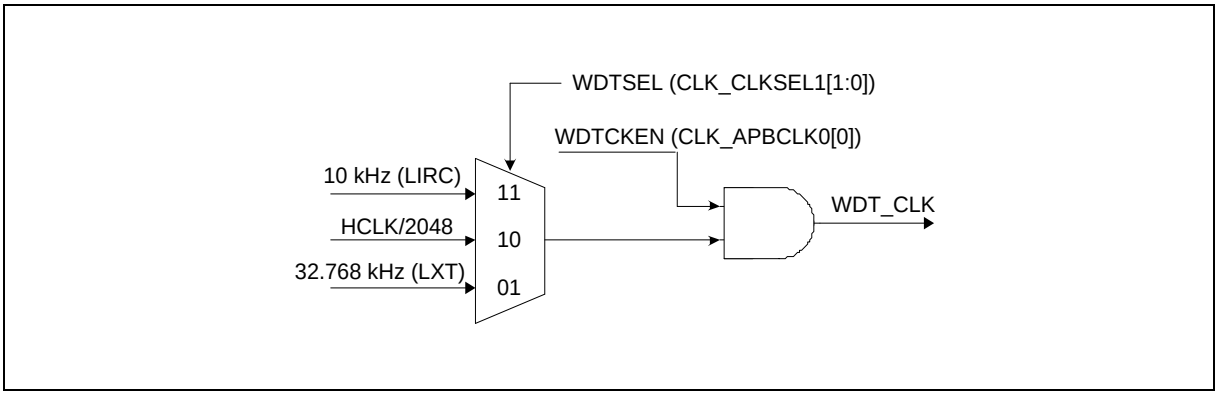


图 6.9-2 看门狗定时器时钟控制

6.9.5 功能描述

看门狗定时器(WDT)包含一个18位的可设置溢出时间间隔的向上计数器。表6.9-1显示了WDT溢出时间间隔选择，图6.9-3展示看门狗定时器溢出时间间隔和复位周期。

6.9.5.1 WDT定时溢出中断

对WDTEN (WDTCR[7])置位可以使能WDT功能，然后WDT计数器向上计数。SYNC (WDT_CTL[30])表示使能/禁止WDTEN功能是否完成。通过设置TOUTSEL (WDTCR[10:8])，可以选择8个不同的溢出时间间隔。当WDT计数器计到TOUTSEL (WDTCR[10:8])设定值，WDT产生溢出中断，然后IF (WDT_CTL[3])标志立即被置1。如果INTEN (WDT_CTL[6])使能，产生WDT超时中断。

6.9.5.2 WDT 复位延时周期和复位系统

如果(WDT_CTL[3])标志被置位后，还会有一个固定的延时周期T_{RSTD}，用户应在T_{RSTD}延时计完之前对RSTCNT (WDT_CTL[0])或WDT_RSTCNT置1来复位18位的WDT向上计数器的值，以防止产生WDT时间溢出复位信号。此外，用户还应该设置RSTDSEL (WDT_ALTCTL [1:0])选择复位延时周期用于清看门狗定时器，如T_{RSTD}时间计满以后，WDT向上计数器的值仍没有被清除，若RSTEN (WDT_CTL[1])位为1，则RSTF (WDT_CTL[2])标志会被置1，然后芯片立即复位。参考图6.9-3，T_{RST}复位周期将保持至少63个WDT时钟周期，然后芯片从复位向量地址(0x0000_0000)重新开始执行程序。芯片由WDT定时溢出复位后，RSTF (WDT_CTL[2])标志将会保持1。用户可以通过检查该标志来确认系统是否因WDT定时溢出复位。

TOUTSEL	超时间隔周期 T_{TIS}	复位延时周期 T_{RSTD}
000	$2^4 * T_{WDT}$	(3/18/130/1026) * T_{WDT}
001	$2^6 * T_{WDT}$	(3/18/130/1026) * T_{WDT}
010	$2^8 * T_{WDT}$	(3/18/130/1026) * T_{WDT}
011	$2^{10} * T_{WDT}$	(3/18/130/1026) * T_{WDT}
100	$2^{12} * T_{WDT}$	(3/18/130/1026) * T_{WDT}
101	$2^{14} * T_{WDT}$	(3/18/130/1026) * T_{WDT}
110	$2^{16} * T_{WDT}$	(3/18/130/1026) * T_{WDT}
111	$2^{18} * T_{WDT}$	(3/18/130/1026) * T_{WDT}

表 6.9-1 看门狗定时器定时溢出间隔周期选择

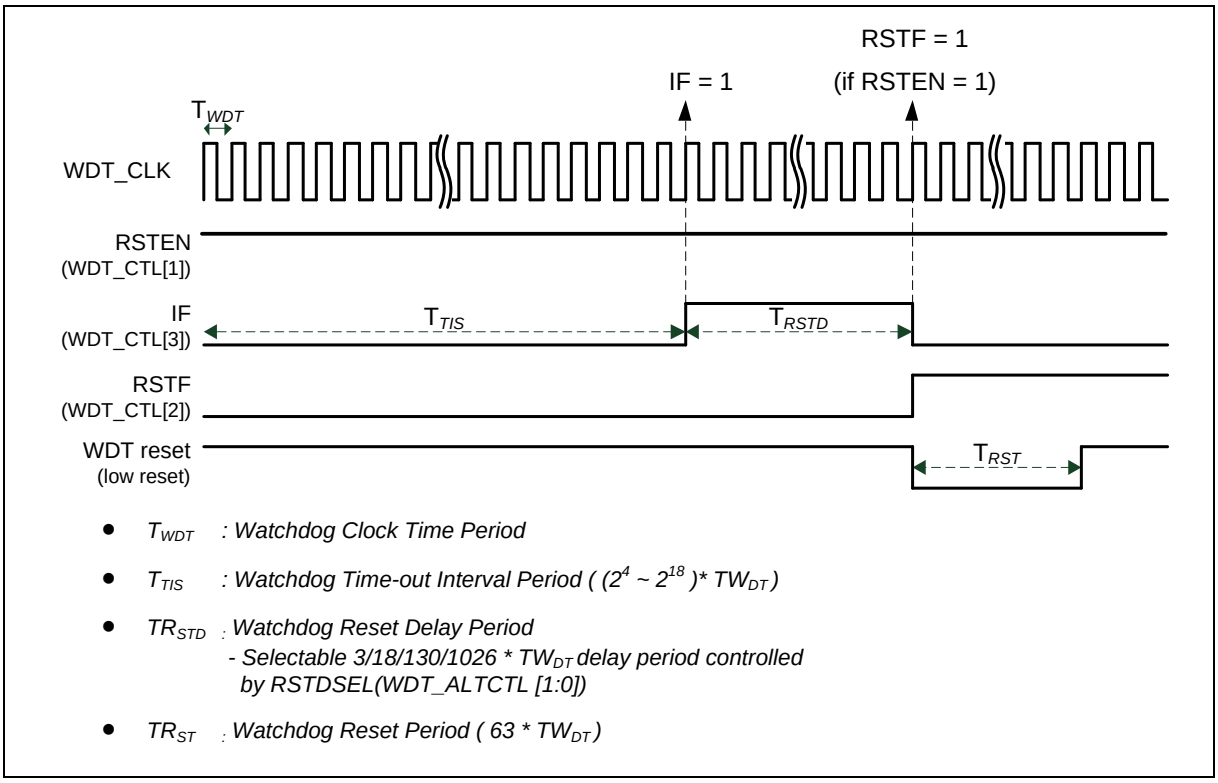


图 6.9-3 看门狗定时器定时溢出间隔和复位周期时序图

6.9.5.3 WDT唤醒

如果看门狗时钟源选择为内部10k或外部低速时钟，如果WKEN (WDT_CTL[4])位被使能，看门狗定时溢出中断产生后，系统能从Power-down模式下被唤醒。注意，用户需在系统进入power down模式前，设置LXTEN (CLK_PWRCTL [1]) 或LIRCEN (CLK_PWRCTL [3])作为时钟源，因为当系统进入power down模式后外设时钟将被禁止。同时，WKF (WDT_CTL[5])标志会被自动置1。用户可以通过查询WKF (WDT_CTL[5])标志知道系统是否是被看门狗定时溢出中断唤醒。

6.9.5.4 WDT ICE调试

当 ICE 连接到 MCU，WDT 计数器是否还在计数由ICEDEBUG (WDT_CTL[31])决定。ICEDEBUG默认值是0，当CPU被ICE停住，WDT停止计数。如果ICEDEBUG置1，无论CPU是否被ICE停住，WDT将保持计数。

6.9.6 寄存器映射

R: 只读, **W**: 只写, **R/W**: 读/写

寄存器	偏移地址	R/W	描述	复位值
WDT基地址: **WDT_BA = 0x4004_0000**				
WDT_CTL	WDT_BA+0x00	R/W	WDT 控制寄存器	0x0000_0700
WDT_ALTCTL	WDT_BA+0x04	R/W	WDT 选择控制寄存器	0x0000_0000
WDT_RSTCNT	WDT_BA+0x08	W	WDT 复位计数器寄存器	0x0000_0000

6.9.7 寄存器描述

WDT控制寄存器(WDT_CTL)

寄存器	偏移地址	R/W	描述	复位值
WDT_CTL	WDT_BA+0x00	R/W	WDT 控制寄存器	0x0000_0700

31	30	29	28	27	26	25	24
ICEDEBUG	SYNC	Reserved					
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved				TOUTSEL			
7	6	5	4	3	2	1	0
WDTEN	INTEN	WKF	WKEN	IF	RSTF	RSTEN	RSTCNT

位	描述	
[31]	ICEDEBUG	**ICE 调试模式下看门狗计数控制位(写保护)** 0 = ICE 调试模式影响看门狗计数，当ICE让CPU运行停止，看门狗计数器也停止. 1 = ICE 调试模式不影响看门狗计数。不论CPU受ICE停止还是运行，看门狗继续计数 **注：**该位为写保护位，详情请参考SYS_REGLCTL 寄存器。
[30]	SYNC	**WDT 使能控制 SYNC 标志指示 (只读)** 如果用户使能/禁止WDTEN (WDT_CTL[7])，该标志可以指示使能/禁止WDTEN操作是否完成。 0 = 设置 WDTEN 位完成. 1 = 设置 WDTEN位正在同步，还未生效。 **注：**执行使能/禁止 WDTEN 位需要2 * WDT_CLK 个时钟周期来生效。
[29:11]	Reserved	保留
[10:8]	TOUTSEL	**看门狗定时器溢出时间间隔选择 (写保护)** 以下三个位用于选择看门狗定时溢出周期 000 = 2^4 * WDT_CLK 001 = 2^6 * WDT_CLK 010 = 2^8 * WDT_CLK 011 = 2^{10} * WDT_CLK 100 = 2^{12} * WDT_CLK 101 = 2^{14} * WDT_CLK 110 = 2^{16} * WDT_CLK 111 = 2^{18} * WDT_CLK **注：**该位为写保护位，详情请参考SYS_REGLCTL寄存器。

[7]	**WDTEN**	**看门狗定时器使能控制位 (写保护)** 0 =看门狗定时器禁止(该操作会复位看门狗计数器的值) 1 =看门狗定时器使能 **注1：** 该位为写保护位，详情请参考SYS_REGLCTL寄存器。 **注2：** 如果CWDTEN[2:0] (Config0[31] 和 Config0[4:3]的组合) 位的值非111，该位会被强制置1，且用户不能把它改为0。
[6]	**INTEN**	**看门狗定时器定时溢出中断使能控制(写保护)** 如果该位使能，看门狗定时溢出后会产生中断信号，并告知CPU 0 = 看门狗定时溢出中断禁止 1 = 看门狗定时溢出中断使能 **注：** 该位为写保护位，详情请参考SYS_REGLCTL寄存器。
[5]	**WKF**	**看门狗定时器溢出唤醒标志(写保护)** 该位表明看门狗定时器中断唤醒标志的状态 0 = 看门狗定时器没有导致芯片唤醒 1 = 芯片从Idle或Power-down 模式被唤醒 **注1：** 该位为写保护位，详情请参考SYS_REGLCTL寄存器。 **注2：** 该位写1清0。
[4]	**WKEN**	**看门狗定时器定时溢出唤醒功能控制位 (写保护)** 如果此位被置1, 当IF (WDT_CTL[3])被置1且INTEN (WDT_CTL[6])被使能，看门狗定时溢出中断信号将会触发唤醒MCU. 0 = 唤醒触发事件禁止，即便看门狗定时器定时溢出中断发生 1 = 唤醒触发事件使能，如果看门狗定时溢出中断产生，将唤醒MCU **注1：** 该位为写保护位，详情请参考SYS_REGLCTL寄存器。 **注2：** MCU能被看门狗定时器定时溢出中断信号唤醒的条件是看门狗定时器时钟源必须为内部10k时钟或外部低速时钟。
[3]	**IF**	**看门狗定时器定时中断标志** 当看门狗定时器计数器的值达到溢出时间间隔，该位将被置1 0 = 没有发生看门狗定时溢出中断 1 = 有看门狗定时溢出中断发生. **注：** 该位写1清零。
[2]	**RSTF**	**看门狗定时器复位标志** 该位用来指示系统是否因看门狗定时器定时溢出发生复位 0 = 没有发生看门狗定时溢出复位 1 = 发生了看门狗定时溢出复位. **注：** 该位写1清零。
[1]	**RSTEN**	**看门狗定时器溢出复位使能位 (写保护)** 如果看门狗计数器的值达到设定值，且指定的WDT复位延时时间已经计完，计数器值还没有被清零，如果该位被设置，则会使能看门狗定时溢出复位 0 = 看门狗定时溢出复位功能禁止 1 =看门狗定时溢出复位功能使能 **注：** 该位为写保护位，详情请参考SYS_REGLCTL寄存器。
[0]	**RSTCNT**	**清看门狗计数器 (写保护)**

		0 = 不影响. 1 =18位看门狗定时器计数值复位清零。 **注1：**该位为写保护位，详情请参考SYS_REGLCTL寄存器。 **注2：**该位硬件自动清零。

WDT选择控制寄存器(WDT_ALTCTL)

寄存器	偏移地址	R/W	描述	复位值
T_ALTCTL	WDT_BA+0x04	R/W	WDT 选择控制寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved						RSTDSEL	

位	描述	
[31:2]	Reserved	保留
[1:0]	RSTDSEL	**看门狗定时器复位延时选择(写保护)** 当看门狗定时器发生溢出，用户有一个复位延时时间来将看门狗计数器清零，以防止看门狗溢出复位。通过置位RSTCNT (WDT_CTL[0])清看门狗计数器。对不同的看门狗溢出时间，用户也可以设置RSTDSEL值选择适当的看门狗复位延时时间。 00 = 看门狗定时器复位延时时间是1026 * WDT_CLK. 01 =看门狗定时器复位延时时间是130 * WDT_CLK. 10 =看门狗定时器复位延时时间是18 * WDT_CLK. 11 =看门狗定时器复位延时时间是3 * WDT_CLK. **注1：**该位为写保护位，详情请参考SYS_REGLCTL寄存器。 **注：**如果看门狗定时器超时复位发生，该寄存器的值将被清零。

WDT复位计数器寄存器 (WDT_RSTCNT)

寄存器	偏移	R/W	描述	Reset Value
WDT_RSTCNT	WDT_BA+0x08	W	WDT复位计数器寄存器	0x0000_0000

31	30	29	28	27	26	25	24
RSTCNT							
23	22	21	20	19	18	17	16
RSTCNT							
15	14	13	12	11	10	9	8
RSTCNT							
7	6	5	4	3	2	1	0
RSTCNT							

位	描述	
[31:0]	RSTCNT	**WDT 复位计数器寄存器** 写 0x00005AA5 到该域会复位内部WDT18位上数计数器值为0。 **注1：**执行 RSTCNT 复位计数器需要2 * WDT_CLK 周期生效。 **注2：**RSTCNT (WDT_CTL[0]) 位是写保护位， RSTCNT (WDT_RSTCNT[31:0]) 不是写保护位.

6.10 窗口看门狗定时器(WWDT)

6.10.1 概述

窗口看门狗定时器(WWDT)用于在一个窗口时间内执行系统复位，以防止程序在不可预知条件下跑到一个不可控的状态。

6.10.2 特性

- 6位向下计数值(CNTDAT, WWDT_CNT[5:0])和6位比较值(CMPDAT, WWDT_CTL[21:16])，使得窗口周期更加灵活
- 支持4位值(PSCSEL, WWDT_CTL[11:8])选择看门狗预分频值，预分频计数器最大可达11位
- WWDT计数器在空闲或掉电模式下暂停

6.10.3 框图

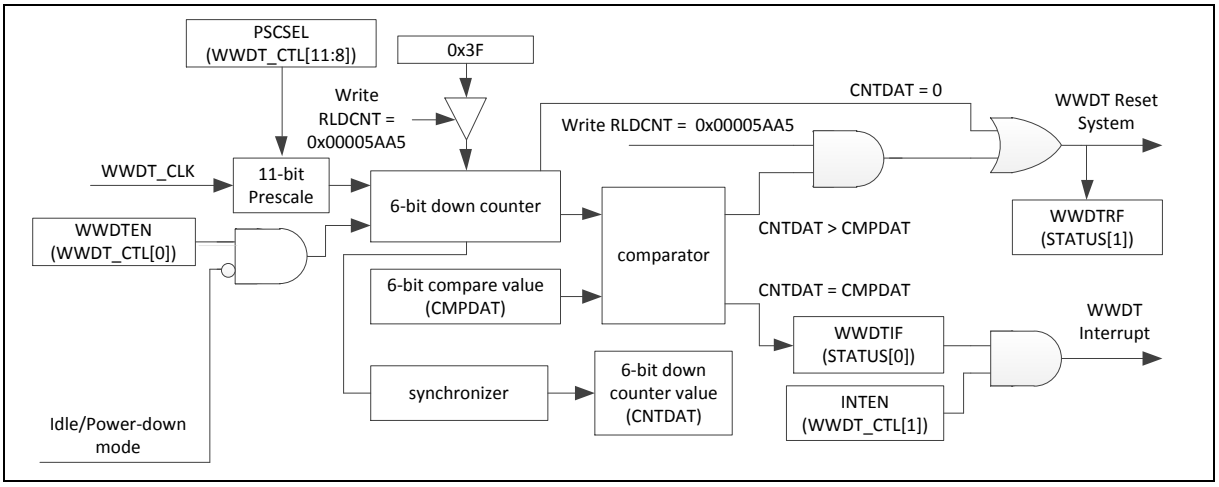


图 6.10-1WWDT 框图

6.10.4 基本配置

- 时钟源配置
  - 通过WWDTSEL (CLK_CLKSEL1[31:30])选择WWDT外设时钟源
  - 通过WDTCKEN (CLK_APBCLK0[0])使能WWDT外设时钟

WWDT时钟控制如下图所示：

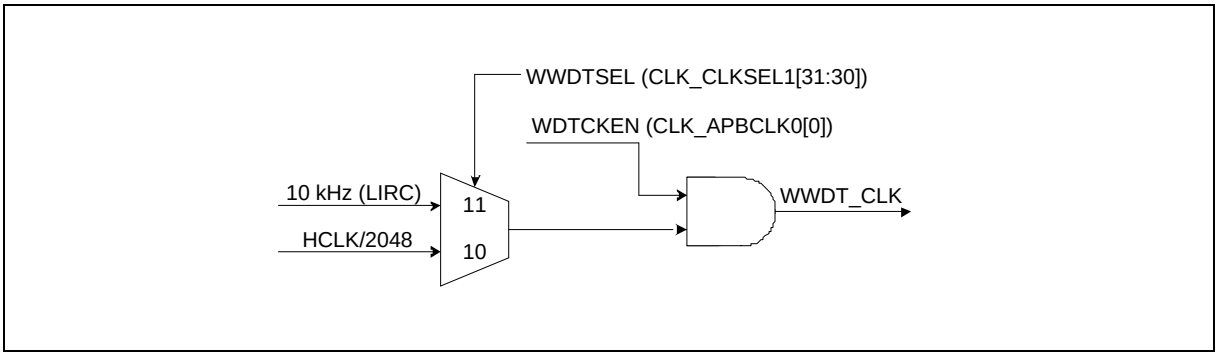


图 6.10-2WWDT 时钟控制

6.10.5 功能描述

窗口看门狗定时器(WWDT)是一个 6位向下计数器，该计数器带一个可选择预分频值，不同的预分频值对应不同的看门狗定时溢出时间。6位窗口看门狗定时器的时钟源可以是系统时钟经2048 (HCLK/2048)分频的时钟，也可以是内部10k低速RC振荡时钟源，看门狗的时钟源带一个可选择的11位预分频值，该值可通过PSCSEL (WWDT_CTL[11:8])位来设置选择，对应预分频值如下表：

PSCSEL	预分频值	最大定时溢出间隔	最大定时溢出间隔 (WWDT_CLK=10 KHz)
0000	1	1 * 64 * T_{WWDT}	6.4 ms
0001	2	2 * 64 * T_{WWDT}	12.8 ms
0010	4	4 * 64 * T_{WWDT}	25.6 ms
0011	8	8 * 64 * T_{WWDT}	51.2 ms
0100	16	16 * 64 * T_{WWDT}	102.4 ms
0101	32	32 * 64 * T_{WWDT}	204.8 ms
0110	64	64 * 64 * T_{WWDT}	409.6 ms
0111	128	128 * 64 * T_{WWDT}	819.2 ms
1000	192	192 * 64 * T_{WWDT}	1.2288 s
1001	256	256 * 64 * T_{WWDT}	1.6384 s
1010	384	384 * 64 * T_{WWDT}	2.4576 s
1011	512	512 * 64 * T_{WWDT}	3.2768 s
1100	768	768 * 64 * T_{WWDT}	4.9152 s
1101	1024	1024 * 64 * T_{WWDT}	6.5536 s
1110	1536	1536 * 64 * T_{WWDT}	9.8304 s
1111	2048	2048 * 64 * T_{WWDT}	13.1072 s

表 6.10-1 WWDT 预分频值选择

6.10.5.1 WWDT 计数

当WWDTEN (WWDT_CTL[0])位被使能，窗口看门狗向下计数器将会从0x3F向下递减计数到0。为了防止程序在非用户指定位置关闭窗口看门狗定时器，窗口看门狗定时器控制寄存器WWDT_CTL在芯片上电

或复位后仅可写一次。当WWDTEN (WWDT_CTL[0])位被软件使能以后，用户不能禁止窗口看门狗定时器 (WWDTEN)、修改计数器预分频周期(PSCSEL)或修改窗口比较值 (CMPDAT)，除非芯片复位。

为了避免在CPU时钟禁止时系统复位，当CPU进入空闲或掉电模式，WWDT计数器会停止计数。在CPU进入正常模式后，WWDT计数器会开始下计数。

6.10.5.2 WWDT 比较匹配中断

窗口看门狗定时器向下计数过程中，当窗口看门狗定时器计数值(CNTDAT) 等于比较值CMPDAT，WWDTIF (WWDT_STATUS[0])会被置1， WWDTIF 可以被软件清零; 如果INTEN (WWDT_CTL[1])位被使能，当WWDTIF 位被硬件置1，就会产生窗口看门狗比较匹配中断。

6.10.5.3 WWDT 复位系统

图6. 10-3表示WWDT三种复位和重载动作情况

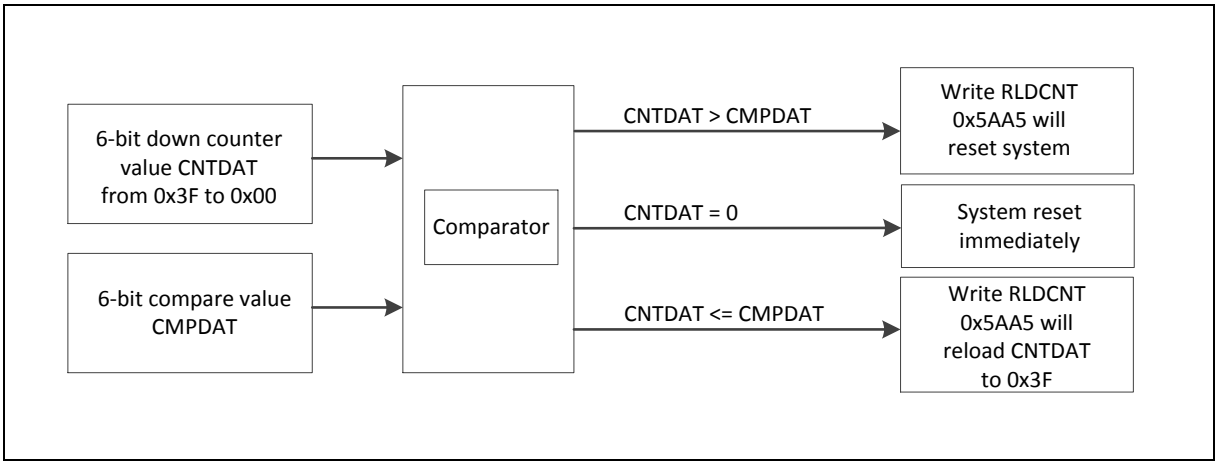


图 6.10-3WWDT 复位和重载动作

如果当前 CNTDAT (WWDT_CNT[5:0]) 大于 CMPDAT (WWDT_CTL[21:16]) 且用户写0x00005AA5 到 WWDT_RLDCNT 寄存器， WWDT 立即产生复位系统信号，芯片复位。当CNTDAT > CMPDAT ，WWDT重载计数器波形如下图：

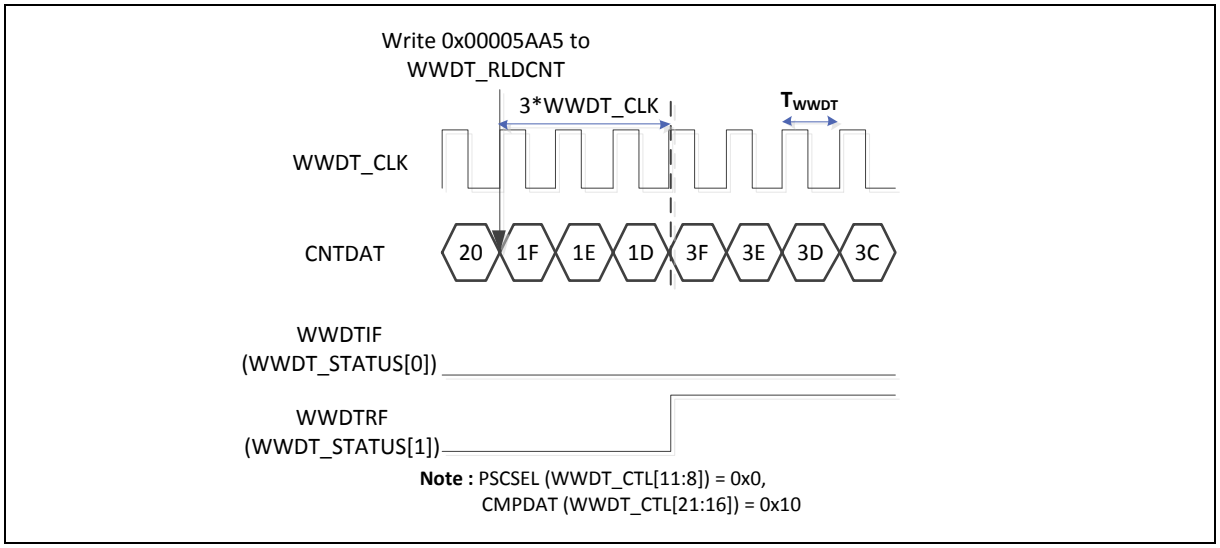


图 6.10-4 当 CNTDAT > CMPDATWWDT 重载计数器

当 WWDTIF (WWDT_STATUS[0]) 置位，用户必须通过写0x00005AA5 到 WWDT_RLDCNT重载WWDT计数器值为0x3F，以防止WWDT计数器值到0产生系统复位信号使系统复位。图6. 10-5表示当CNTDAT < CMPDAT WWDT重载计数器波形。图6. 10-6表示如果用户在计数器数到0之前没有写0x00005AA5 到 WWDT_RLD WWDT产生复位系统信号的波形。

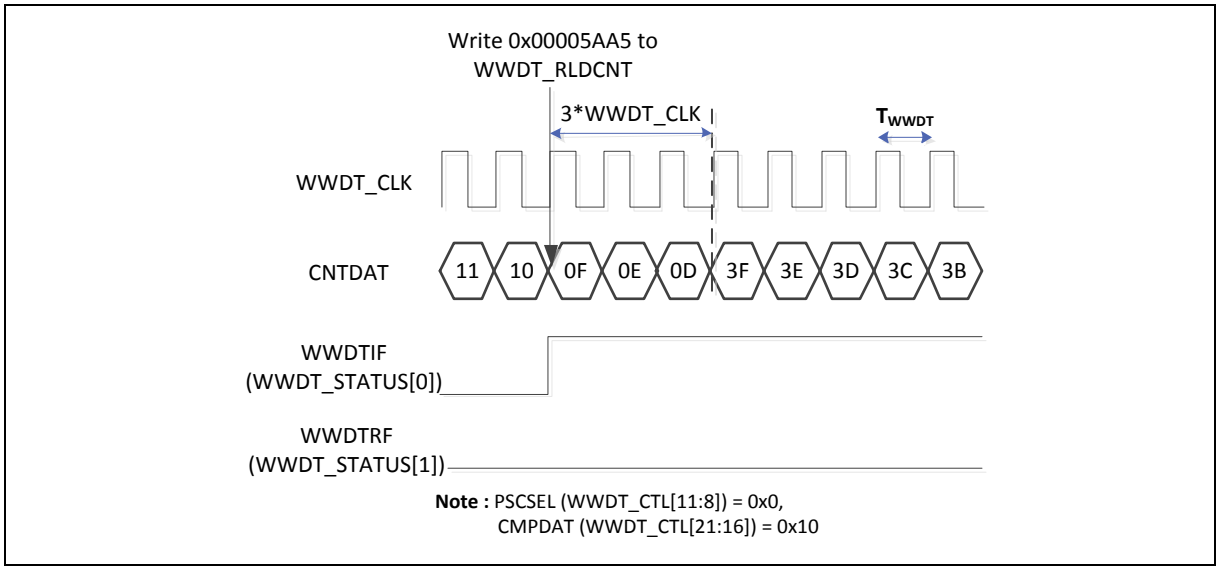


图 6.10-5 当 WWDT_CNT < WINCMPWWDT 重载计数器

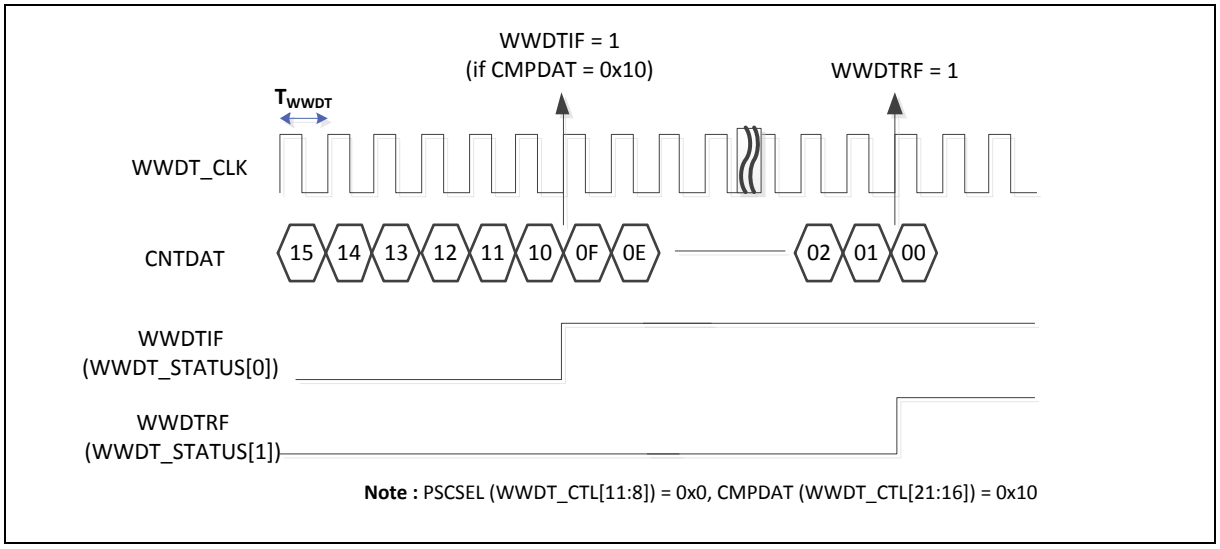


图 6.10-6WWDT 中断和复位信号

6.10.5.4 WWDT 窗口设置限制

当用户对WWDT_RLDCNT寄存器写0x00005AA5来重载WWDT的值到0x3F的时候，大概需要3个窗口看门狗定时器时钟来同步重载命令到实际执行重载操作。这意味着如果用户设置PSCSEL (WWDT_CTL[11:8])的值为0000，计数器预分频值应该设置为1，CMPDAT (WWDT_CTL[21:16])寄存器的值必须大于2；否则，当WWDTIF标志产生后，写WWDT_RLDCNT来重载窗口看门狗定时器的值为0x3F操作无效，WWDTIF (WWDT_STATUS[0])将会产生，窗口看门狗定时器系统复位将发生。WWDT CMPDAT设置限制如下表。

如果CMPDATA 设置为0x3F或0x0, 则中断不会发生。因为WWDT计数到0x0发生复位，所以当CMPDATA 为 0x0时中断不会发生。

PSCSEL	预分频值	有效 CMPDAT 值
0000	1	0x3 ~ 0x3E
0001	2	0x2 ~ 0x3E
其它	**其它**	0x1 ~ 0x3E

表 6.10-2 CMPDAT 设置限制

6.10.5.5 WWDT ICE 调试

当 ICE 连接到 MCU，WWDT 计数器是否计数由ICEDEBUG (WWDT_CTL[31])决定。ICEDEBUG默认值是0，当 CPU 被 ICE停住，WWDT 计数器停止计数。如果ICEDEBUG 置 1， WWDT 计数器会保持计数，无论CPU 被 ICE停住与否。

6.10.6 寄存器映射

R: 只读, **W**: 只写, **R/W**: 读/写

寄存器	偏移地址	R/W	描述	复位值
WWDT基地址: **WWDT_BA = 0x4004_0100**				
WWDT_RLDCNT	WWDT_BA+0x00	W	窗口看门狗定时器重载计数器寄存器	0x0000_0000
WWDT_CTL	WWDT_BA+0x04	R/W	窗口看门狗定时器控制寄存器	0x003F_0800
WWDT_STATUS	WWDT_BA+0x08	R/W	窗口看门狗定时器状态寄存器	0x0000_0000
WWDT_CNT	WWDT_BA+0x0C	R	窗口看门狗定时器计数器值寄存器	0x0000_003F

6.10.7 寄存器描述

WWDT重载计数器寄存器(WWDT_RLDCNT)

寄存器	偏移地址	R/W	描述	复位值
WWDT_RLDCNT	WWDT_BA+0x00	W	WWDT定时器重载计数器寄存器	0x0000_0000

31	30	29	28	27	26	25	24
RLDCNT							
23	22	21	20	19	18	17	16
RLDCNT							
15	14	13	12	11	10	9	8
RLDCNT							
7	6	5	4	3	2	1	0
RLDCNT							

位	描述	
[31:0]	RLDCNT	**窗口看门狗定时器重载计数器寄存器** 对此寄存器写0x00005AA5 将会重载窗口看门狗定时器计数器的值到0x3F。 **注:**用户只能当窗口看门狗计数器的值在0到CMPDAT (WWDT_CTL[21:16])之间才可以写WWDT_RLDCNT寄存器来重载窗口看门狗计数器。如果窗口看门狗计数器的值大于CMPDAT 时写WWDT_RLDCNT寄存器，窗口看门狗定时器复位信号会立即产生。

WWDT控制寄存器(WWDT_CTL)

寄存器	偏移地址	R/W	描述	复位值
WWDT_CTL	WWDT_BA+0x04	R/W	WWDT控制寄存器	0x003F_0800

注：此寄存器仅可以当芯片上电或复位后写一次。

31	30	29	28	27	26	25	24
ICEDEBUG	**Reserved**						
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved		**CMPDAT**					
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved				**PSCSEL**			
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved						**INTEN**	**WWDTEN**

位	描述	
[31]	**ICEDEBUG**	**ICE 调试模式下窗口看门狗计数控制位** 0 =ICE 调试模式下影响窗口看门狗定时器计数，当CPU被ICE暂停后，窗口看门狗向下计数器计数值将保持不变 1 = ICE 调试模式下不影响窗口看门狗定时器计数，不管CPU是否被ICE暂停，窗口看门狗定时器向下计数器将会保持继续计数
[30:22]	**Reserved**	保留
[21:16]	**CMPDAT**	**窗口看门狗定时器窗口比较寄存器** 设置此寄存器来调整有效重载窗口 **注：** 当前窗口看门狗定时器计数器的值在0到CMPDAT之间时，用户才可以写WWDT_RLDCNT来重载窗口看门狗定时器计数器的值。如果当前窗口看门狗定时器计数器的值大于CMPDAT时用户写WWDT_RLDCNT寄存器，窗口看门狗定时器复位信号会立即产生。
[15:12]	**Reserved**	保留
[11:8]	**PSCSEL**	**窗口看门狗定时器计数器预分频周期选择** 0000 =预分频为1; 最大定时溢出周期是1 * 64 * WWDT_CLK 0001 =预分频为2; 最大定时溢出周期是2 * 64 * WWDT_CLK 0010 =预分频为4; 最大定时溢出周期是4 * 64 * WWDT_CLK 0011 =预分频为8; 最大定时溢出周期是8 * 64 * WWDT_CLK 0100 =预分频为16; 最大定时溢出周期是16 * 64 * WWDT_CLK 0101 =预分频为32; 最大定时溢出周期是32 * 64 * WWDT_CLK 0110 =预分频为64; 最大定时溢出周期是64 * 64 * WWDT_CLK 0111 =预分频为128; 最大定时溢出周期是128 * 64 * WWDT_CLK 1000 =预分频为192; 最大定时溢出周期是192 * 64 * WWDT_CLK 1001 =预分频为256; 最大定时溢出周期是256 * 64 * WWDT_CLK

		1010 =预分频为384; 最大定时溢出周期是384 * 64 * WWDT_CLK 1011 =预分频为512; 最大定时溢出周期是512 * 64 * WWDT_CLK 1100 =预分频为768; 最大定时溢出周期是768 * 64 * WWDT_CLK 1101 =预分频为1024; 最大定时溢出周期是1024 * 64 * WWDT_CLK 1110 =预分频为1536; 最大定时溢出周期是1536 * 64 * WWDT_CLK 1111 =预分频为2048; 最大定时溢出周期是2048 * 64 * WWDT_CLK
[7:2]	**Reserved**	保留
[1]	**INTEN**	**窗口看门狗定时器中断使能位** 如果该位被使能，如果有窗口看门狗定时器计数器比较匹配中断信号产生，就会通知CPU 0 =窗口看门狗定时器计数器比较匹配中断禁止 1 =窗口看门狗定时器计数器比较匹配中断使能
[0]	**WWDTEN**	**窗口看门狗定时器使能控制位** 设置该位来使能窗口看门狗定时器计数器计数 0 =窗口看门狗定时器计数器停止 1 =窗口看门狗定时器计数器计数开始

WWDT状态寄存器(WWDT_STATUS)

寄存器	偏移地址	R/W	描述	复位值
WWDT_STATUS	WWDT_BA+0x08	R/W	WWDT状态寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved						WWDTRF	WWDTIF

位	描述	
[31:2]	**Reserved**	保留
[1]	**WWDTRF**	**窗口看门狗定时器溢出复位标志** 该位指示系统是否已被窗口看门狗定时器溢出复位 0 =窗口看门狗定时器没有发生复位 1 =窗口看门狗定时器发生了复位 **注**：该位是写1清零。
[0]	**WWDTIF**	**窗口看门狗定时器比较匹配中断标志** 该位指示当WWDT 的计数值与CMPDAT (WWDT_CTL[21:16])匹配时，窗口看门狗定时器的比较匹配中断状态标志 0 =无影响 1 =窗口看门狗定时器计数器的值与CMPDAT寄存器的值匹配 注：该位是写1清零。

WWDT计数器值寄存器 (WWDT_CNT)

寄存器	偏移地址	R/W	描述	复位值
WWDT_CNT	WWDT_BA+0x0C	R	WWDT计数器值寄存器	0x0000_003F

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved		**CNTDAT**					

位	描述	
[31:6]	**Reserved**	保留
[5:0]	**CNTDAT**	**窗口看门狗定时器计数器值** CNTDAT将会被持续更新，以监视6位向下计数器的值。

6.11 实时时钟(RTC)

6.11.1 概述

实时时钟 (RTC) 控制器用于记录实时时间及日历等信息。RTC控制器支持可配置的时间节拍和闹钟定时中断。时间及日历等信息的表示格式为BCD 码。可对外接晶振的频率精度进行数字频率补偿。

6.11.2 特性

- 支持独立RTC电源域V_{BAT}(M48xGC/M48xE8)
- 支持时间计数（秒、分、时）和日历计数（日、月、年），用户可以通过访问寄存器RTC_TIME和RTC_CAL查看时间及日历。
- 可设定闹钟时间（秒、分、时）和日历（日、月、年），参看寄存器RTC_TALM 和RTC_CALM。
- 可设定闹钟时间（秒、分、时）和日历（日、月、年）的掩码使能功能，参看RTC_TAMSK和 RTC_CAMSK寄存器。
- 可选择12小时或 24小时制式，参看RTC_CLKFMT寄存器。
- 可选RTC_TIME和RTC_TALM支持1/128秒HZCNT
- 支持闰年自动识别，参看RTC_LEAPYEAR寄存器
- 支持周内日期计数，参看RTC_WEEKDAY寄存器
- 支持RTC时钟源频率补偿功能，参看寄存器RTC_FREQADJ
- 所有时间、日期的数据格式为 BCD 码
- 支持周期RTC时间节拍中断，提供 8个周期选项供选择，分别为：1/128、1/64、1/32、1/16、1/8、1/4、1/2 及1秒
- 支持 RTC 定时节拍和闹钟定时中断
- 支持RTC中断从空闲模式或掉电模式下唤醒芯片
- 通过RTC_DSTCTL寄存器，支持夏令时软件控制
- 支持6 个独立 tamper 管脚
- 提供20/80字节的备用寄存器用于存储用户信息，并提供tamper检测脚用于清除备用寄存器中内容
- 支持Flash整体擦除同样清除20个字节备份寄存器内容(M48xGC/M48xE8)

6.11.3 框图

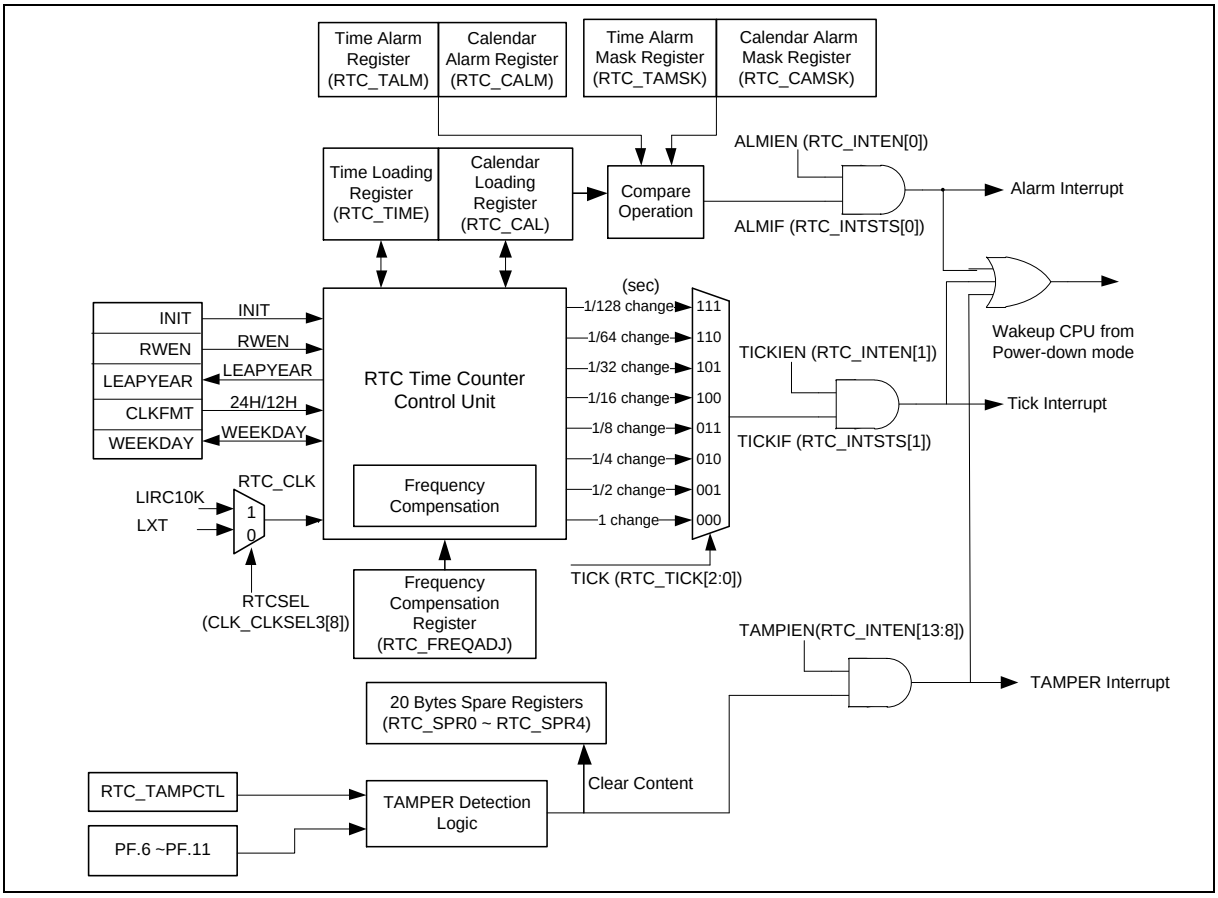


图 6.11-1RTC 框图

6.11.4 基本配置

- 时钟源配置
  - RTC控制器时钟源通过RTCCKEN (APBCLK0[1])使能， RTC 时间计数器时钟源通过CLK_CLKSEL3[8] 选择LXT 或 LIRC
  - LXT时钟源可以通过设置RTC_LXTCTL[7] = 1选择为32K晶振；通过设置RTC_LXTCTL[7] = 0选择为LIRC32K
- 管脚配置

组	管脚名称	GPIO	MFP
X32	X32_OUT	PF.4	MFP10
	X32_IN	PF.5	MFP10
TAMPER0	TAMPER0	PF.6	MFP10
TAMPER1	TAMPER1	PF.7	MFP10

TAMPER2	TAMPER2	PF.8	MFP10
TAMPER3	TAMPER3	PF.9	MFP10
TAMPER4	TAMPER4	PF.10	MFP10
TAMPER5	TAMPER5	PF.11	MFP10

6.11.5 功能描述

6.11.5.1 RTC初始化

当 RTC 模块上电后，RTC 处于 reset 状态。用户需要写入数据 (0x a5eb1357) 到RTC 初始化寄存器 RTC_INIT(INIT[31:0])中，让 RTC 离开 reset 状态。一旦RTC_INIT寄存器被写入 0xa5eb1357，RTC 将会永远处于Active状态。用户可以读Active 位(INIT[0])来判断RTC是处于Active状态或是reset状态。

6.11.5.2 RTC读/写使能

如果 RWENF (RTC_RWEN[16]) 位置 1，意味着RTC寄存器是可读写的。当在1120个PCLK周期内执行写RTC寄存器命令超过6次，RTCBUSY (RTC_RWEN[24]) 标志会被置1，RWENF (RTC_RWEN[16]) 会被清0。当RWENF 是 1 和 0时， RTC寄存器访问属性如下表：

寄存器	INIR =0	RWENF =1	RWENF =0 Or RTCBUSY=1
RTC_INIT	可用	R/W	R/W
RTC_RWEN	可用	R/W	R
RTC_FREQADJ	可用	R/W	R
RTC_TIME	不可用	R/W	R
RTC_CAL	不可用	R/W	R
RTC_CLKFMT	不可用	R/W	R
RTC_WEEKDAY	不可用	R/W	R
RTC_TALM	不可用	R/W	R
RTC_CALM	不可用	R/W	R
RTC_LEAPYEAR	不可用	R	R
RTC_INTEN	可用	R/W	R/W
RTC_INTSTS	可用	R/W	R/W
RTC_TICK	不可用	R/W	R
RTC_TAMSK	不可用	R/W	R
RTC_CAMSK	不可用	R/W	R
RTC_SPRCTL	可用	R/W	不可用
RTC_SPRx	可用	R/W	不可用

RTC_LXTCTL	可用	R/W	R/W
RTC_GPIOCTL0	可用	R/W	R/W
RTC_GPIOCTL1	可用	R/W	R/W
RTC_DSTCTL	不可用	R/W	R
RTC_TAMPCTL	可用	R/W	R
RTC_TAMPSEED	可用	R/W	R

表 6.11-1RTC 读/写使能

6.11.5.3 频率补偿(M48xGC/M48xE8)

频率补偿电路支持动态补偿调整补偿值，不用复位补偿电路。动态补偿使能位为DYN_COMP_EN(RTC_CLKFMT[16])。

如果动态补偿使能，连续写入RTC_FREQADJ的最小间隔是0.24秒，FCR_BUSY(RTC_FREQADJ[31])用于指示新寄存器写操作禁止。补偿值将在写入RTC_FREQADJ后0.24~1.16秒加载进补偿电路。

RTC_FREQADJ寄存器允许用户对时钟输入做数字补偿。请遵循如下例子和公式写32K晶振的实际频率到RTC_FREQADJ寄存器，如下是高于或低于32768 Hz的补偿例子。

例1:

频率计数器测量：32773.65 Hz

整数部分：32773 => RTC_FREQADJ[12:8] = 0x15,参考INTEGER(RTC_FREQADJ[12:8])获取详细设置值。

小数部分：0.65 X 64 = 41.6(0x2A) => RTC_FREQADJ[5:0]=0x2A

例2:

频率计数器测量：32763.25 Hz

整数部分：32763=> RTC_FREQADJ[12:8] = 0x0B, 参考INTEGER(RTC_FREQADJ[12:8]) 获取详细设置值。

小数部分：0.25 X 64 = 16(0x10) => RTC_FREQADJ[5:0] = 0x10

注：如果补偿没有执行，RTC_FREQADJ 寄存器为默认值(0x0000_1000)。用户可以在生产时通过时钟输出功能，使用频率计数器测量RTC时钟源。同时，用户可以使用时钟输出功能检查RTC频率补偿的结果。

6.11.5.4 频率补偿(M48xID/M48xGA)

RTC_FREQADJ 寄存器允许用户对输入时钟做数字补偿，请参考下面例子和公式写32k晶振的实际频率到RTC_FREQADJ寄存器。下面是频率高于或低于32768 Hz补偿的例子。

例**1:**

频率计数器测量：32773.65 Hz (> 32768 Hz)

FREQADJ = (32768 * 0x200000) / 32773.65 = 0x1FFE96

例**2:**

频率计数器测量：32763.25 Hz (< 32768 Hz)

FREQADJ = (32768 * 0x200000) / 32763.25 = 0x200130

注:当补偿没有执行， RTC_FREQADJ 寄存器是值默认值(0x0020_0000)。用户可以通过时钟输出功能使用频率计数器来测量RTC时钟源，同时用户可以使用时钟输出功能检查RTC的时钟补偿结果。

6.11.5.5 时间和日历计数器

RTC_TIME 和 RTC_CAL 用于加载实时时间和日期。 RTC_TALM 和 RTC_CALM 用于设置闹钟时间和日期。

6.11.5.6 12/24 小时制选择

设置 24HEN (RTC_CLKFMT[0]) 选择 12/24 小时时标制式。当 RTC 运行在 12 小时制模式时，RTC_TIME[21]（TENHR[1:0])中的高位代表AM/PM提示符）。（如果RTC_TIME[21]=1，表示目前为PM时间，RTC_TIME[21]=0，表示目前为AM时间）。

注：小时信息写入到RTC_TIME[21:16]，信息格式为BCD格式			
24小时制 (24HEN = 1)		12小时制 (PM 时间 + 0x20) (24HEN = 0) (PM 时间 + 0x20)	
0x00 (AM12)	0x12 (PM12)	0x12 (AM12)	0x32 (PM12)
0x01 (AM01)	0x13 (PM01)	0x01 (AM01)	0x21 (PM01)
0x02 (AM02)	0x14 (PM02)	0x02 (AM02)	0x22 (PM02)
0x03 (AM03)	0x15 (PM03)	0x03 (AM03)	0x23 (PM03)
0x04 (AM04)	0x16 (PM04)	0x04 (AM04)	0x24 (PM04)
0x05 (AM05)	0x17 (PM05)	0x05 (AM05)	0x25 (PM05)
0x06 (AM06)	0x18 (PM06)	0x06 (AM06)	0x26 (PM06)
0x07 (AM07)	0x19 (PM07)	0x07 (AM07)	0x27 (PM07)
0x08 (AM08)	0x20 (PM08)	0x08 (AM08)	0x28 (PM08)
0x09 (AM09)	0x21 (PM09)	0x09 (AM09)	0x29 (PM09)
0x10 (AM10)	0x22 (PM10)	0x10 (AM10)	0x30 (PM10)
0x11 (AM11)	0x23 (PM11)	0x11 (AM11)	0x31 (PM11)

表 6.11-212/24 小时制选择

6.11.5.7 星期计数器

RTC控制器提供一周日期寄存器WEEKDAY (RTC_WEEKDAY [2：0])，其值由0至6，用于表示周日至周六。

6.11.5.8 时间周期节拍中断

通过设置TICK（RTC_TICK [2:0]）来选择周期中断，周期中断有8个选项：1/128、1/64、1/32、1/16、1/8 、 1/4 、 1/2 以 及 1 秒 。 当 TICKIEN (RTC_INTEN[1]) 被 置 1 ， 周 期 中 断 使 能 后 ， MCU 根 据 RTC_TICK[2:0]寄存器内设定的值周期性的发生中断。

6.11.5.9 闹钟中断

当RTC_TIME和RTC_CAL中的值等于RTC_TALM和RTC_CALM中的设定值，如果此时闹钟中断已设为使能 (ALMIEN (RTC_INTEN[0])=1)，则闹钟中断标志ALMIF (RTC_INTSTS[0])被置位同时发出闹钟中断请求。

RTC控制器提供时间屏蔽寄存器（RTC_TAMSK）和日历闹钟屏蔽寄存器（RTC_CAMSK），用于屏蔽指定数字并产生周期中断而无需改动每个闹钟中断服务程序中的闹钟匹配条件（匹配条件根据 RTC_TALM 和 RTC_CALM中配置决定）。

6.11.5.10 夏令时

RTC控制器也提供RTC_DSTCTL 寄存器来存储夏令时应用的控制设置。用户可以读RTC_DSTCTL值来检查当前RTC时间日期计数器是否运行在夏令时模式还是正常模式。

6.11.5.11 1 Hz 时钟输出

RTC控制器提供1Hz 时钟输出到 CLKO 功能管脚。用户可以设置 CLK1HZEN (CLK_CLKOCTL[6]) 为 1 且使能 RTC，1Hz 时钟会输出到 CLKO 功能管脚。

6.11.5.12 应用指南

1. RTC_TALM、RTC_CALM、RTC_TIME 和 RTC_CAL寄存器中数据格式为BCD 格式。
2. 用户必须保证载入值为有效合理值。例如，RTC_CAL = 201a (年)、13 (月)、00 (日)或RTC_CAL与RTC_WEEKDAY不匹配，等等。
3. 在RTC_CAL 和 RTC_CALM 中，仅 2 位 BCD 码用于指示“年”。目前仅支持20xy年份，不支持19xy 或 21xy年。
4. 12小时制时间设定范例
   如果RTC时间为PM12:59:30，RTC_TIME设定为:
   1) HOUR:
      RTC_TIME[21:16]: 0x32 (0x12+0x20)，可表示为：TENHR (RTC_TIME[21:20]) is 0x3, HR (RTC_TIME[19:16]) is 0x2。
   2) MIN:
      RTC_TIME[14:8]: 0x59 可表示为：TENMIN (RTC_TIME[14:12]) is 0x5, MIN (RTC_TIME[11:8]) is 0x9。
   3) SEC:
      RTC_TIME[6:0]: 0x30 ， 可表示为：TENSEC (RTC_TIME[6:4]) is 0x3, SEC (RTC_TIME[3:0]) is 0x0。

表6. 11-3展示在内核和电池初次上电后的寄存器值。

寄存器	复位状态
RTC_INIT	0
RTC_RWEN	0
RTC_CAL	15/8/8 (年/月/日)
RTC_TIME	00:00:00 (时 : 分 : 秒)
RTC_CALM	00/00/00 (年/月/日)
RTC_TALM	00:00:00 (时 : 分 : 秒)
RTC_CLKFMT	1 (24小时模式)
RTC_WEEKDAY	6 (星期六)
RTC_INTEN	0
RTC_INTSTS	0
RTC_LEAPYEAR	0
RTC_TICK	0
RTC_DSTCTL	0

表 6.11-3 上电后寄存器值

5. 下表列出寄存器属于内核电源域，其他属于电池电源域.

寄存器	电源域
RTC_RWEN	内核电源域
RTC_INTEN	内核电源域
RTC_INTSTS	内核电源域
Others	电池电源域

表 6.11-4 寄存器电源域

6.11.5.13 备用寄存器和篡改(Tamper)事件检测(M48xGC/M48xE8)

RTC模块提供了20字节的备用寄存器用于存储用户的重要信息，这些寄存器位于RTC时钟域，在向5个中任意一个备用寄存器(RTC_SPR0 ~ RTC_SPR4)写值前，用户必须使能寄存器SPRRWEN (RTC_SPRCTL[2])。

当侦测到信号变化满足RTC_TAMPCTL中定义的条件，tamper中断标志会TAMPxIF (RTC_INTSTS[13:8])置位，同时备用寄存器(RTC_SPR0 ~ RTC_SPR4)中的20个字节数据将被硬件自动清除，以防保密数据被盗取。当前RTC时间和日期会加载到RTC_TAMPTIME 和 RTC_TAMPCAL寄存器，这些值仅可以被自动清除或当所有的TAMPxIF都被清0时再次更新。如果TAMPxIF置1，且tamper侦测中断使能，会产生中断到NVIC。

在Flash整体擦除后，这20字节备用寄存器内容将自动清除。

RTC支持6个独立tamper引脚。

镜态Tamper编程时序例子

使能Tamper 编程时序:

1) 清除TAMPxIF (RTC_INTSTS[x+8], x=0, 1, …, 5)

2) 设置 TAMPxLV (RTC_TAMPCTL[4x+9], x=0, 1, ...,5) 和 TAMPxDBEN (RTC_TAMPCTL[4x+10], x=0, 1, ...,5).

3) 使能 TAMPxEN (RTC_TAMPCTL[4x+8], x=0, 1, ...,5).

禁用或重配置Tamper编程时序:

1) 禁用 TAMPxEN (RTC_TAMPCTL[4x+8], x=0, 1, ...,5).

2) 等待3个 RTC CLK.

3) 重配置TAMPxLV (RTC_TAMPCTL[4x+9], x=0, 1, ...,5) 和 TAMPxDBEN (RTC_TAMPCTL[4x+10], x=0, 1, ...,5)..

4) 使能 TAMPxEN (RTC_TAMPCTL[4x+8], x=0, 1, ...,5)

6.11.5.14 备用寄存器和盗取(Tamper)事件检测(M48xID/M48xGA)

RTC模块提供了80字节的备用寄存器用于存储用户的重要信息，这些寄存器位于RTC时钟域，在向20个中任意一个备用寄存器(RTC_SPR0 ~ RTC_SPR19)写值前，用户必须使能寄存器SPRRWEN (RTC_SPRCTL[2])。

当侦测到信号变化满足RTC_TAMPCTL中定义的条件，tamper中断标志会TAMPxIF (RTC_INTSTS[13:8])置位，同时备用寄存器(RTC_SPR0 ~ RTC_SPR19)中的80个字节数据将被硬件自动清除，以防保密数据被盗取。当前RTC时间和日期会加载到RTC_TAMPTIME 和 RTC_TAMPCAL寄存器，这些值仅可以被自动清除或当所有的TAMPxIF都被清0时再次更新。如果TAMPxIF置1，且tamper侦测中断使能，会产生中断到NVIC。

RTC支持3对动态循环tamper管脚或6个独立的tamper管脚。DYNRATE (RTC_TAMPCTL[7:5])决定一个参考位的持续时间，如图6. 11-2.所示。在表6. 11-5中，设置DYNSRC (RTC_TAMPCTL[3:2])可以选择侦测信号是硬件随机值产生器产生的新值或之前的随机值或用户定义的种子值(RTC_TAMPSEED[31:0])，用于动态循环tamper管脚。设置DYN1ISS (RTC_TAMPCTL[0])为1可以选择对1侦测源来自tamper 0，设置DYN12SS (RTC_TAMPCTL[1]) 为1可以选择对2侦测源也来自tamper 0。TAMPxLV (RTC_TAMPCTL[4x+9], x=0, 1, ..., 5)取决于用于独立tamper侦测的TAMPERx管脚的电平属性。Tamper控制效果如表6. 11-6和表6. 11-7所示

DYNRATE	Duration
000	1/32 sec
001	1/16 sec
010	1/8 sec
011	1/4 sec
100	1/2 sec
101	1 sec
110	2 sec
111	4 sec

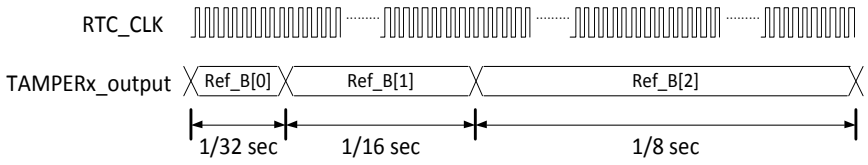


图 6.11-2 动态率定义

DYNPRxEN	DYNSRC[1:0]	侦测信号描述
0	x	TAMPERn 是静态 tamper 侦测 (n=0 ~ 5)

1	00	当目前动态样本使用完后，产生新的随机值作为参考样本
1	01	当目前的动态样本使用完后,不会产生另一组新的随机数值,供动态样本使用,而是重复使用之前的动态样本
1	10	当目前动态样本使用完后，产生新的随机值作为参考样本
1	11	当动态样本使用完后,重复使用用户输入到种子值作为参考样本

表 6.11-5 动态样本源选择

Tamper 配置			Tamper控制位影响					
DYNPR0EN	TAMPER0EN	TAMPER1EN	TAMP0LV	TAMP1LV	TAMP0DBEN	TAMP1DBEN	TAMPER 0	TAMPER 1
0	0	0	-	-	-	-	-	-
0	0	1	-	V	-	V	-	Static
0	1	0	V	-	V	-	Static	X
0	1	1	V	V	V	V	Static	Static
1	x	0	-	-	-	-	Dynamic Out	-
1	x	1	-	-	-	-	Dynamic Out	Dynamic In

表 6.11-6Tamper 控制位对 0 对的影响

Tamper 配置				Tamper控制位影响					
DYNPRxEN	DYNxISS	TAMPERnEN	TAMPEmREN	TAMPnLV	TAMPmLV	TAMPnDBEN	TAMPmDBEN	TAMPER n	TAMPER m
0	x	0	0	-	-	-	-	-	-
0	x	0	1	-	V	-	V	-	Static
0	x	1	0	V	-	V	-	Static	-
0	x	1	1	V	V	V	V	Static	Static
1	0	x	0	-	-	-	-	Dynamic Out	-
1	0	x	1	-	-	-	-	Dynamic Out	Dynamic In
1	1	0	1	-	-	-	-	-	Dynamic Input form TAMPER 0
1	1	1	1	V	V	-	-	Static	Dynamic Input form TAMPER 0
x= 1 and 2		n= 2 and 4, m= 3 and 5							

表 6.11-7 控制位对 1 和 2 对的影响

清TAMPxIF (RTC_INTSTS[x+8], x=0, 1, …, 5)

1. 设置 TAMPxLV (RTC_TAMPCTL[4x+9], x=0, 1, ..., 5) 和 TAMPxDBEN (RTC_TAMPCTL[4x+10], x=0, 1, ..., 5)；
2. 使能 TAMPxEN (RTC_TAMPCTL[4x+8], x=0, 1, ..., 5)。

动态 Tamper 编程顺序范例.

1. 清TAMPxIF (RTC_INTSTS[x+8], x=0, 1, …, 5)；
2. 填SEED (RTC_TAMPSEED[31:0])；
3. 设置 DYNSRC (RTC_TAMPCTL[3:2]), DYNxISS (RTC_TAMPCTL[x], x=0, 1) 和 DYNRATE (RTC_TAMPCTL[7:5])；
4. 使能 DYNPRxEN (RTC_TAMPCTL[8x+16], x=0, 1, 2)；
5. 使能 TAMPxEN (RTC_TAMPCTL[4x+8], x=0, 1, ..., 5)；
6. 设置 SEEDRLD (RTC_TAMPCTL[4])。

6.11.5.15 备份域GPIO功能

当PF.4/X32O和PF.5/X32I引脚不用于低速32K晶振功能，可以被用于GPIO引脚功能。CTLSEL0 (RTC_GPIOCTL0[3])用于选择PF.4/X32O引脚通过RTC或GPIO模块控制。PF.5/X32I引脚通过CTLSEL1（RTC_GPIOCTL0[11]）控制。PF.6~PF.11引脚通过RTC_GPIOCTL0 和 RTC_GPIOCTL1寄存器的CTLSEL2~CTLSEL7控制。图6. 11-3展示备份I/O控制框图。

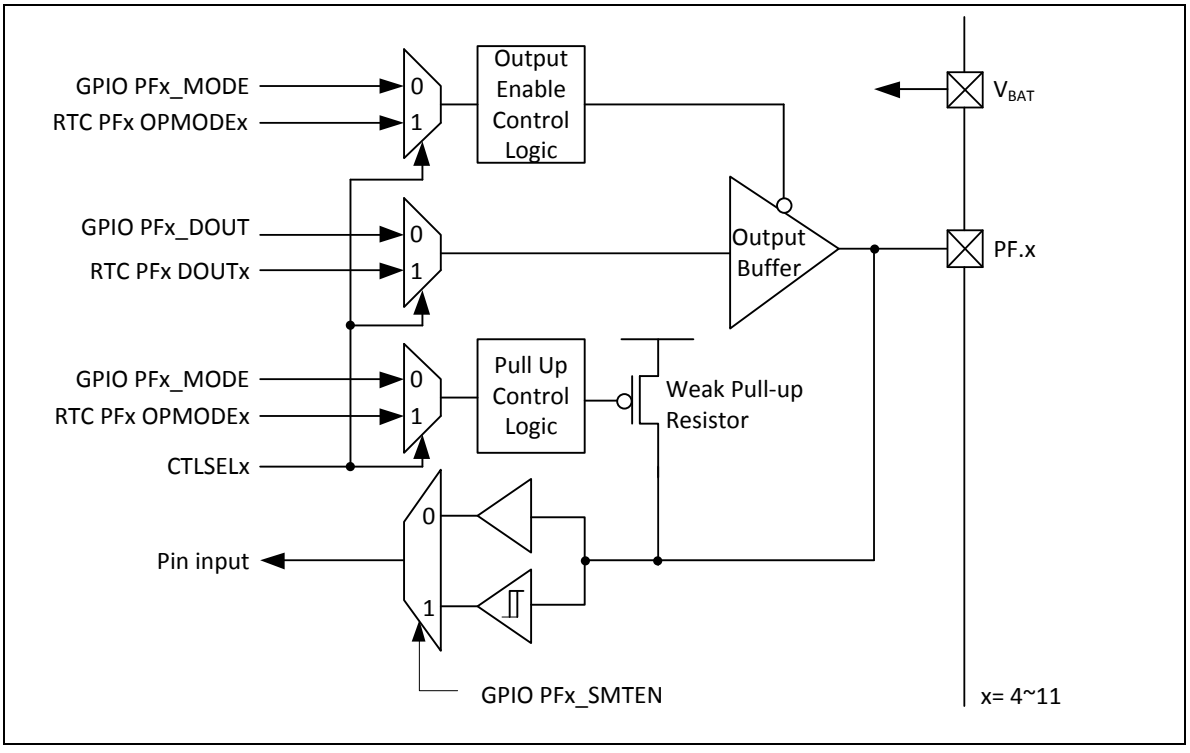


图6.11-3备份I/O控制框图

注：只要任何RTC寄存器有写操作，将成功切换GPIO引脚到RTC域，通过RTC备份寄存器控制

(M48xGC/M48xE8)。

6.11.6 寄存器映射

R: 只读, **W**: 只写, **R/W**: 读/写

寄存器	偏移	R/W	描述	复位值
RTC基地址: **RTC_BA = 0x4004_1000**				
RTC_INIT	RTC_BA+0x00	R/W	RTC初始化寄存器	0x0000_0000
RTC_RWEN	RTC_BA+0x04	R	RTC访问使能寄存器	0x0001_0000
RTC_FREQADJ	RTC_BA+0x08	R/W	RTC频率补偿寄存器	0x0001_0000
RTC_TIME	RTC_BA+0x0C	R/W	时间载入寄存器	0x0000_0000
RTC_CAL	RTC_BA+0x10	R/W	日历载入寄存器	0x0015_0808
RTC_CLKFMT	RTC_BA+0x14	R/W	时间制式（时标）选择寄存器	0x0000_0001
RTC_WEEKDAY	RTC_BA+0x18	R/W	星期寄存器	0x0000_0006
RTC_TALM	RTC_BA+0x1C	R/W	时间闹钟寄存器	0x0000_0000
RTC_CALM	RTC_BA+0x20	R/W	日历闹钟寄存器	0x0000_0000
RTC_LEAPYEAR	RTC_BA+0x24	R	闰年指示寄存器	0x0000_0000
RTC_INTEN	RTC_BA+0x28	R/W	RTC中断使能寄存器	0x0000_0000
RTC_INTSTS	RTC_BA+0x2C	R/W	RTC中断指示寄存器	0x0000_0000
RTC_TICK	RTC_BA+0x30	R/W	RTC时钟节拍寄存器	0x0000_0000
RTC_TAMSK	RTC_BA+0x34	R/W	RTC时间闹钟屏蔽寄存器	0x0000_0000
RTC_CAMSK	RTC_BA+0x38	R/W	RTC日历闹钟屏蔽寄存器	0x0000_0000
RTC_SPRCTL	RTC_BA+0x3C	R/W	RTC备用功能控制寄存器	0x0000_0000
RTC_SPR0	RTC_BA+0x40	R/W	RTC备用寄存器0	0x0000_0000
RTC_SPR1	RTC_BA+0x44	R/W	RTC备用寄存器1	0x0000_0000
RTC_SPR2	RTC_BA+0x48	R/W	RTC 备用寄存器2	0x0000_0000
RTC_SPR3	RTC_BA+0x4C	R/W	RTC 备用寄存器3	0x0000_0000
RTC_SPR4	RTC_BA+0x50	R/W	RTC 备用寄存器4	0x0000_0000
RTC_LXTCTL	RTC_BA+0x100	R/W	RTC 32.768 kHz晶振控制寄存器	0xXXXX_XX0E
RTC_GPIOCTL0	RTC_BA+0x104	R/W	RTC GPIO 控制寄存器 0	0x0000_0000
RTC_GPIOCTL1	RTC_BA+0x108	R/W	RTC GPIO控制寄存器1	0x0000_0000
RTC_DSTCTL	RTC_BA+0x110	R/W	RTC 夏令时控制寄存器	0x0000_0000

RTC_TAMPCTL	RTC_BA+0x120	R/W	RTC Tamper 管脚控制寄存器	0x0000_0000
RTC_TAMPTIME	RTC_BA+0x130	R	RTC Tamper 时间寄存器	0x0000_0000
RTC_TAMPCAL	RTC_BA+0x134	R	RTC Tamper 日历寄存器	0x0000_0000

6.11.7 寄存器描述

RTC初始化寄存器(RTC_INIT)

寄存器	偏移	R/W	描述	复位值
RTC_INIT	RTC_BA+0x00	R/W	RTC初始化寄存器	0x0000_0000

31	30	29	28	27	26	25	24
INIT							
23	22	21	20	19	18	17	16
INIT							
15	14	13	12	11	10	9	8
INIT							
7	6	5	4	3	2	1	0
INIT							INIT/ACTIVE

位	描述	
[31:1]	INIT[31:1]	**RTC 初始化** 当 RTC 模块上电后，RTC 处于复位状态。用户需要写入数据 (0xa5eb1357) 到 INIT 使得 RTC 离开复位状态。一旦 INIT 被写入 0xa5eb1357，RTC 将会一直处于非复位状态。 INIT寄存器的这些位只能被写，如果读的话，总是为0。
[0]	INIT[0]/ACTIVE	**RTC 激活状态（只读）** 0 = RTC 在复位状态 1 = RTC 在正常激活状态

RTC 访问使能寄存器 (RTC_RWEN)

寄存器	偏移	R/W	描述	复位值
RTC_RWEN	RTC_BA+0x04	R	RTC访问使能寄存器	0x0001_0000

31	30	29	28	27	26	25	24
Reserved							**RTCBUSY**
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							**RWENF**
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved							

位	描述	
[31:25]	**Reserved**	保留
[24]	**RTCBUSY**	**RTC写忙标志** 该位表示 RTC寄存器是否可写.在RTCBUSY期间，RTC寄存器R/W无效。 0 =RTC 寄存器可写 1 =RTC 寄存器不可写, RTC 在忙状态 **注1：** 当在1120个 PCLK周期内执行写RTC寄存器命令超过6次，RTCBUSY 标志会被置位； **注 2：** 该位参考RWENF (当 RTCBUSY时，RWENF = 0)。
[23:17]	**Reserved**	保留
[16]	**RWENF**	**RTC 寄存器访问使能标志 (只读)** 0 = 禁止RTC寄存器读/写访问 1 = 使能RTC寄存器读/写访问 **注：** 在RTCBUSY为1期间，RWENF将被屏蔽为0。
[15:0]	**Reserved**	保留

RTC频率补偿寄存器(RTC_FREQADJ)

寄存器	偏移	R/W	描述	复位值
RTC_FREQADJ	RTC_BA+0x08	R/W	RTC频率补偿寄存器	0x0001_0000

31	30	29	28	27	26	25	24
FCR_BUSY	Reserved						
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved			INTEGER				
7	6	5	4	3	2	1	0
Reserved		FRACTION					

位	描述	
[31]	FCR_BUSY	**频率补偿寄存器写操作忙(只读)** 0 = 新寄存器写操作允许 1 = 最后一次写操作还在进行，新的寄存器写操作禁止 **注：** 该位只在DYN_COMP_EN(RTC_CLKFMT[16])使能时使用。
[30:13]	Reserved	保留
[12:8]	INTEGER	**整数部分** 00000 = 检测值的整数部分为32752 00001 =检测值的整数部分为32753 00010 =检测值的整数部分为32754 00011 =检测值的整数部分为32755 00100 =检测值的整数部分为32756 00101 =检测值的整数部分为32757 00110 = 检测值的整数部分为32758 00111 = 检测值的整数部分为32759 01000 = 检测值的整数部分为32760 01001 = 检测值的整数部分为32761 01010 = 检测值的整数部分为32762 01011 = 检测值的整数部分为32763 01100 = 检测值的整数部分为32764 01101 = 检测值的整数部分为32765 01110 = 检测值的整数部分为32766 01111 = 检测值的整数部分为32767 10000 = 检测值的整数部分为32768 10001 = 检测值的整数部分为32769

		10010 = 检测值的整数部分为32770 10011 = 检测值的整数部分为32771 10100 = 检测值的整数部分为32772 10101 = 检测值的整数部分为32773 10110 = 检测值的整数部分为32774 10111 = 检测值的整数部分为32775 11000 = 检测值的整数部分为32776 11001 = 检测值的整数部分为32777 11010 = 检测值的整数部分为32778 11011 = 检测值的整数部分为32779 11100 = 检测值的整数部分为32780 11101 = 检测值的整数部分为32781 11110 = 检测值的整数部分为32782 11111 = 检测值的整数部分为32783
[7:6]	**Reserved**	保留
[5:0]	**FRACTION**	**小数部分** 公式：FRACTION = (检测值小数部分) X 64 **注：**在 FCR的数字必须是16进制格式。

注：如果动态补偿没有使能，i.e.DYN_COMP_EN(RTC_CLKFMT[16]) = 0，当写RTC_FREQADJ，FREQADJ的计数器将被复位用于开始补偿，RTC时间将重新开始。

RTC 时间载入寄存器 (RTC_TIME)

寄存器	偏移	R/W	描述	复位值
RTC_TIME	RTC_BA+0x0C	R/W	RTC时间载入寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved	HZCNT						
23	22	21	20	19	18	17	16
Reserved		TENHR		HR			
15	14	13	12	11	10	9	8
Reserved	TENMIN			MIN			
7	6	5	4	3	2	1	0
Reserved	TENSEC			SEC			

位	描述	
[31]	**Reserved**	保留
[30:24]	**HZCNT**	亚秒计数器(0x00 ~0x7F)
[23:22]	**Reserved**	保留
[21:20]	**TENHR**	小时的十位数(0~2) 当RTC采用12小时制时，RTC_TIME[21](TENHR[1:0]的高位)表示AM/PM提示符(如果IRTC_TIME[21]=1，表示为PM时间)
[19:16]	**HR**	小时的个位数(0~9)
[15]	**Reserved**	保留
[14:12]	**TENMIN**	分钟的十位数(0~5)
[11:8]	**MIN**	分钟的个位数(0~9)
[7]	**Reserved**	保留
[6:4]	**TENSEC**	秒钟的十位数(0~5)
[3:0]	**SEC**	秒钟的个位数(0~9)

注:

1. RTC_TIME 为 BCD 计数方式(除了 HZCNT)，RTC 不会对载入值的合理性进行检测；
2. 括号内列出载入值的合理范围；
3. 当写RTC_TIME, RTC_CAL, RTC_WEEKDAY时，FREQADJ'的计数器会复位以开始补偿。RTC时间会重新开始；
4. 当RTC_CLKFMT[8] = 0写RTC_TIME，HZCNT将设置为0，当RTC_CLKFMT[8] = 1，用户可以写值0x00 ~0x7F到HZCNT。

RTC日历载入寄存器(RTC_CAL)

寄存器	偏移	R/W	描述	复位值
RTC_CAL	RTC_BA+0x10	R/W	RTC日历载入寄存器	0x0015_0808

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
TENYEAR				YEAR			
15	14	13	12	11	10	9	8
Reserved			TENMON	MON			
7	6	5	4	3	2	1	0
Reserved		TENDAY		DAY			

位	描述	
[31:24]	Reserved	保留
[23:20]	TENYEAR	年日历的十位数(0~9)
[19:16]	YEAR	年日历的个位数(0~9)
[15:13]	Reserved	保留
[12]	TENMON	月日历的十位数(0~1)
[11:8]	MON	月日历的个位数(0~9)
[7:6]	Reserved	保留
[5:4]	TENDAY	天日历的十位数(0~3)
[3:0]	DAY	天日历的个位数(0~9)

注：

1. RTC_CAL为 BCD 计数格式，RTC 不会检测载入值的合理性；
2. 括号内列出载入值的合理范围；
3. 当写RTC_TIME、RTC_CAL、RTC_WEEKDAY，FREQADJ的计数器将被复位用于开始补偿，RTC时间将重新开始。

RTC时间制式选择寄存器(RTC_CLKFMT)

寄存器	偏移	R/W	描述	复位值
RTC_CLKFMT	RTC_BA+0x14	R/W	RTC时间制式选择寄存器	0x0000_0001

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							DYN_COMP_EN
15	14	13	12	11	10	9	8
Reserved							HZCNTEN
7	6	5	4	3	2	1	0
Reserved							24HEN

位	描述	
[31:17]	Reserved	保留
[16]	DYN_COMP_EN	**动态补偿使能位** 0 = 动态补偿禁用 1 = 动态补偿使能
[15:9]	Reserved	保留
[8]	HZCNTEN	**亚秒计数器使能位** 0 = RTC_TIME和RTC_TALM的HZCNT禁用 1 =RTC_TIME和RTC_TALM的HZCNT使能
[7:1]	Reserved	保留
[0]	24HEN	**24小时/ 12小时时间制式选择** 用于表示RTC_TIME 和 RTC_TALM的时间为24小时或12小时制式 0 =选择12小时制带AM和PM指示 1 =选择24小时制

RTC星期寄存器(RTC_WEEKDAY)

寄存器	偏移	R/W	描述	复位值
RTC_WEEKDAY	RTC_BA+0x18	R/W	RTC星期寄存器	0x0000_0006

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved					WEEKDAY		

位	描述	
[31:3]	Reserved	保留
[2:0]	WEEKDAY	星期寄存器 000 = 星期日 001 = 星期一 010 = 星期二 011 = 星期三 100 = 星期四 101 =星期五 110 =星期六 111 = 保留

注：当写RTC_TIME、RTC_CAL、RTC_WEEKDAY，FREQADJ的计数器将被复位用于开始补偿，RTC时间将重新开始。

RTC时间闹钟寄存器(RTC_TALM)

寄存器	偏移	R/W	描述	复位值
RTC_TALM	RTC_BA+0x1C	R/W	RTC时间闹钟寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved	HZCNT						
23	22	21	20	19	18	17	16
Reserved		TENHR		HR			
15	14	13	12	11	10	9	8
Reserved	TENMIN			MIN			
7	6	5	4	3	2	1	0
Reserved	TENSEC			SEC			

位	描述	
[31]	Reserved	保留
[30:24]	HZCNT	亚秒计数器(0x00 ~0x7F)
[21:20]	TENHR	闹钟中小时时间的十位数(0~2) 当 RTC采用12-小时制时，RTC_TIME[21](TENHR[1:0]的高位)表示 AM/PM 提示符.(如果RTC_TIME[21]=1,表示为PM时间.)
[19:16]	HR	闹钟中小时时间的个位数(0~9)
[15]	Reserved	保留
[14:12]	TENMIN	闹钟中分钟时间的十位数(0~5)
[11:8]	MIN	闹钟中分钟时间的个位数(0~9)
[7]	Reserved	保留
[6:4]	TENSEC	闹钟中秒钟时间的十位数(0~5)
[3:0]	SEC	闹钟中秒钟时间的个位数(0~9)

注:

1. RTC_TALM 为 BCD 计数方式，RTC 不会对载入值的合理性进行检测；
2. 括号内列出载入值的合理范围；
3. 在 RTC 访问使能位RWENF (RTC_RWEN[16])激活后，该寄存器的值可以被读回。

RTC日历闹钟寄存器(RTC_CALM)

寄存器	偏移	R/W	描述	复位值
RTC_CALM	RTC_BA+0x20	R/W	RTC日历闹钟寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
TENYEAR				YEAR			
15	14	13	12	11	10	9	8
Reserved			TENMON	MON			
7	6	5	4	3	2	1	0
Reserved		TENDAY		DAY			

位	描述	
[31:24]	Reserved	保留
[23:20]	TENYEAR	闹钟中年日历的十位数(0~9)
[19:16]	YEAR	闹钟中年日历的个位数(0~9)
[15:13]	Reserved	保留
[12]	TENMON	闹钟中月日历的十位数 (0~1)
[11:8]	MON	闹钟中月日历的个位数(0~9)
[7:6]	Reserved	保留
[5:4]	TENDAY	闹钟中天日历的十位数(0~3)
[3:0]	DAY	闹钟中天日历的个位数(0~9)

注：

1. RTC_CALM 为 BCD 计数方式，RTC 不会对载入值的合理性进行检测；
2. 括号内列出载入值的合理范围；
3. 在 RTC 访问使能位RWENF (RTC_RWEN[16])激活后，该寄存器的值可以被读回。

RTC闰年指示寄存器(RTC_LEAPYEAR)

寄存器	偏移	R/W	描述	复位值
RTC_LEAPYEAR	RTC_BA+0x24	R	RTC闰年指示寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved							LEAPYEAR

位	描述	
[31:1]	Reserved	保留
[0]	LEAPYEAR	闰年指示寄存器（只读） 0 = 表示该年非闰年 1 = 表示该年为闰年

RTC中断使能寄存器(RTC_INTEN)

寄存器	偏移	R/W	描述	复位值
RTC_INTEN	RTC_BA+0x28	R/W	RTC中断使能寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved						CLKSTOPIEN	CLKFIEN
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved		TAMP5IEN	TAMP4IEN	TAMP3IEN	TAMP2IEN	TAMP1IEN	TAMP0IEN
7	6	5	4	3	2	1	0
Reserved						TICKIEN	ALMIEN

位	描述	
[31:26]	**Reserved**	保留
[25]	**CLKSTOPIEN**	**LXT时钟频率检测停止中断使能位** 0 = LXT频率停止中断禁用 1 = LXT频率停止中断使能
[24]	**CLKFIEN**	**LXT时钟频率检测失败中断使能位** 0 = LXT频率失败中断禁用 1 = LXT频率失败中断使能
[23:14]	**Reserved**	保留
[13]	**TAMP5IEN**	**Tamper 5 或对 2 中断使能位** 设置 TAMP5IEN 为 1 也可以使能tamper 5 中断事件唤醒芯片功能 0 = Tamper 5 或对 2 中断禁止 1 = Tamper 5或对 2 中断使能
[12]	**TAMP4IEN**	**Tamper 4中断使能位** 设置 TAMP4IEN 为 1 也可以使能tamper 4中断事件唤醒芯片功能 0 = Tamper 4中断禁止 1 = Tamper 4中断使能
[11]	**TAMP3IEN**	**Tamper 3或对1中断使能位** 设置 TAMP3IEN 为 1 也可以使能tamper 3中断事件唤醒芯片功能 0 = Tamper 3或对1中断禁止 1 = Tamper 3或对1中断使能
[10]	**TAMP2IEN**	**Tamper 2中断使能位** 设置 TAMP2IEN 为 1 也可以使能tamper 2中断事件唤醒芯片功能

		0 = Tamper 2中断禁止 1 = Tamper 2中断使能
[9]	**TAMP1IEN**	**Tamper 1或对0中断使能位** 设置 TAMP1IEN 为1也可以使能tamper 1中断事件唤醒芯片功能 0 = Tamper 1或对0中断禁止 1 = Tamper 3或对1中断使能
[8]	**TAMP0IEN**	**Tamper 0中断使能位** 设置 TAMP0IEN 为 1 也可以使能tamper 0中断事件唤醒芯片功能 0 = Tamper 0中断禁止 1 = Tamper 0中断使能
[7:2]	**Reserved**	保留
[1]	**TICKIEN**	**时钟节拍中断使能位** 设置 TICKIEN 为 1也可以使能芯片节拍中断唤醒功能 0 = 禁止RTC 时钟节拍中断 1 = 使能RTC 时钟节拍中断
[0]	**ALMIEN**	**闹钟中断使能位** 设置ALMIEN为 1也可以使能芯片闹钟中断唤醒功能 0 = 禁止RTC 闹钟中断 1 = 使能RTC 闹钟中断

RTC中断状态寄存器 (RTC_INTSTS)

寄存器	偏移	R/W	描述	复位值
RTC_INTSTS	RTC_BA+0x2C	R/W	RTC中断状态寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved		TAMP5IF	TAMP4IF	TAMP3IF	TAMP2IF	TAMP1IF	TAMP0IF
7	6	5	4	3	2	1	0
Reserved						TICKIF	ALMIF

位	描述	
[31:14]	**Reserved**	保留
[13]	**TAMP5IF**	**Tamper 5中断标志** 当TAMP5_PIN 侦测到电平不等于TAMP5LV (RTC_TAMPCTL[29])，或在DYNPR2EN (RTC_TAMPCTL[31])激活期间TAMP4_PIN 和 TAMP5_PIN未连接，或在DYNPR2EN (RTC_TAMPCTL[31]) 和 DYN2ISS (RTC_TAMPCTL[1])激活期间TAMP0_PIN 和 TAMP5_PIN未连接，该位置1。 0 = 没有Tamper 5中断标志产生 1 = Tamper 5中断标志产生 **注1：** 该位写1清零； **注2：** 如果Tamper设置条件没有恢复，该标志位将会再次产生； **注3：** 清除所有的TAPMxIF将会自动清除RTC_TAMPTIME和RTC_TAMPCAL。
[12]	**TAMP4IF**	**Tamper 4中断标志** 当TAMP4_PIN 侦测到电平不等于TAMP4LV (RTC_TAMPCTL[25])，该位置1。 0 = 没有Tamper 4中断标志产生 1 = Tamper 4中断标志产生 **注1：** 该位写1清零； **注2：** 如果Tamper设置条件没有恢复，该标志位将会再次产生； **注3：** 清除所有的TAPMxIF将会自动清除RTC_TAMPTIME和RTC_TAMPCAL。
[11]	**TAMP3IF**	**Tamper 3中断标志** 当TAMP3_PIN 侦测到电平不等于TAMP3LV (RTC_TAMPCTL[21])，或在DYNPR1EN (RTC_TAMPCTL[23])激活期间TAMP2_PIN 和 TAMP3_PIN未连接，或在DYNPR1EN (RTC_TAMPCTL[23]) 和 DYN1ISS (RTC_TAMPCTL[0])激活期间TAMP0_PIN 和 TAMP3_PIN未连接，该位置1。 0 = 没有Tamper 3中断标志产生 1 = Tamper 3中断标志产生

		注1： 该位写1清零； **注2：** 如果Tamper设置条件没有恢复，该标志位将会再次产生； **注3：** 清除所有的TAPMxIF将会自动清除RTC_TAMPTIME和RTC_TAMPCAL。
[10]	**TAMP2IF**	**Tamper 2中断标志** 当TAMP2_PIN 侦测到电平不等于TAMP2LV (RTC_TAMPCTL[17])，该位置1。 0 = 没有Tamper 2中断标志产生 1 = Tamper 2中断标志产生 **注1：** 该位写1清零； **注2：** 如果Tamper设置条件没有恢复，该标志位将会再次产生； **注3：** 清除所有的TAPMxIF将会自动清除RTC_TAMPTIME和RTC_TAMPCAL。
[9]	**TAMP1IF**	**Tamper 1或对0中断标志** 当TAMP1_PIN 侦测到电平不等于TAMP1LV (RTC_TAMPCTL[13])，或在DYNPR0EN (RTC_TAMPCTL[15])激活期间TAMP0_PIN 和 TAMP1_PIN未连接，该位置1。 0 = 没有Tamper 1中断标志产生 1 = Tamper 1中断标志产生 **注1：** 该位写1清零； **注2：** 如果Tamper设置条件没有恢复，该标志位将会再次产生； **注3：** 清除所有的TAPMxIF将会自动清除RTC_TAMPTIME和RTC_TAMPCAL。
[8]	**TAMP0IF**	**Tamper 0中断标志** 当TAMP0_PIN 侦测到电平不等于TAMP0LV (RTC_TAMPCTL[9])，该位置1。 0 = 没有Tamper 0中断标志产生 1 = Tamper 0中断标志产生 **注1：** 该位写1清零； **注2：** 如果Tamper设置条件没有恢复，该标志位将会再次产生； **注3：** 清除所有的TAPMxIF将会自动清除RTC_TAMPTIME和RTC_TAMPCAL。
[7:2]	**Reserved**	保留
[1]	**TICKIF**	**RTC时钟节拍中断标志** 0 = 未发生节拍情况 1 = 发生节拍情况 **注：** 该位写1清0。
[0]	**ALMIF**	**RTC 闹钟中断标志** 0 = 不匹配闹钟条件 1 = 匹配闹钟条件 **注：** 该位写1清0。

RTC时钟节拍寄存器(RTC_TICK)

寄存器	偏移	R/W	描述	复位值
RTC_TICK	RTC_BA+0x30	R/W	RTC时钟节拍寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved					TICK		

位	描述	
[31:3]	Reserved	保留
[2:0]	TICK	**时钟节拍寄存器** 该位用来设定产生时钟节拍中断的周期. 000 = 时钟节拍为1秒 001 = 时钟节拍为1/2 秒 010 = 时钟节拍为1/4 秒 011 = 时钟节拍为1/8 秒 100 = 时钟节拍为1/16 秒 101 = 时钟节拍为1/32 秒 110 = 时钟节拍为1/64 秒 111 = 时钟节拍为1/128 秒 **注**：在 RTC 访问使能位RWENF (RTC_RWEN[16])激活后，该寄存器的值可以被读回。

注：在 RTC 访问使能位RWENF (RTC_RWEN[16])激活后，该寄存器的值可以被读回。

RTC时间闹钟屏蔽寄存器(RTC_TAMSK)

寄存器	偏移	R/W	描述	复位值
RTC_TAMSK	RTC_BA+0x34	R/W	RTC时间闹钟屏蔽寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved		MTENHR	MHR	MTENMIN	MMIN	MTENSEC	MSEC

位	描述	
[31:6]	Reserved	保留
[5]	MTENHR	屏蔽闹钟小时时间的十位数(0~2)
[4]	MHR	屏蔽闹钟小时时间的个位数(0~9)
[3]	MTENMIN	屏蔽闹钟分钟的十位数(0~5)
[2]	MMIN	屏蔽闹钟分钟的个位数(0~9)
[1]	MTENSEC	屏蔽闹钟秒钟的十位数(0~5)
[0]	MSEC	屏蔽闹钟秒钟的个位数(0~9)

注:

1. RTC_TAMSK 为 BCD 计数方式，RTC 不会对载入值的合理性进行检测；
2. 括号内列出载入值的合理范围；
3. MTENHR/MHR 基于24小时制。

RTC日历闹钟屏蔽寄存器(RTC_CAMSK)

寄存器	偏移	R/W	描述	复位值
RTC_CAMSK	RTC_BA+0x38	R/W	RTC日历闹钟屏蔽寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved		MTENYEAR	MYEAR	MTENMON	MMON	MTENDAY	MDAY

位	描述	
[31:6]	Reserved	保留
[5]	MTENYEAR	屏蔽闹钟年日历的十位数(0~9)
[4]	MYEAR	屏蔽闹钟年日历的个位数(0~9)
[3]	MTENMON	屏蔽闹钟月日历的十位数(0~1)
[2]	MMON	屏蔽闹钟月日历的个位数(0~9)
[1]	MTENDAY	屏蔽闹钟天日历的个位数(0~3)
[0]	MDAY	屏蔽闹钟天日历的十位数(0~9)

注:

1. RTC_CALM为 BCD 计数方式，RTC不会对载入值的合理性进行检测；
2. 括号内列出载入值的合理范围。

RTC备用功能控制寄存器(RTC_SPRCTL)

寄存器	偏移	R/W	描述	复位值
RTC_SPRCTL	RTC_BA+0x3C	R/W	RTC备用功能控制寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							LXTFCLR
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved		SPRCSTS	Reserved		SPRRWEN	Reserved	

位	描述	
[31:17]	**Reserved**	保留
[16]	**LXTFCLR**	**LXT 时钟失败/停止清除备份使能位** 0 = LXT失败/停止清除备份寄存器内容禁用 1 = LXT失败/停止清除备用寄存器内容使能
[15:6]	**Reserved**	保留
[5]	**SPRCSTS**	**SPR 清除标志** 该位表明如果侦测到指定盗取事件后, RTC_SPR0 ~ RTC_SPR19中内容是否已被清除. 0 = 特定寄存器内容没有被清除 1 = 特定寄存器内容已被清除 写1可清除该位 **注：** RTC_INTSTS[13:8] 不等于0，该位保持1。
[4:3]	**Reserved**	保留
[2]	**SPRRWEN**	**SPR 寄存器使能位** 0 = 备用寄存器禁用 1 = 备用寄存器使能 **注：** 当备用寄存器禁用后，RTC_SPR0 ~ RTC_SPR19将不可访问。
[1:0]	**Reserved**	保留

RTC备用寄存器(RTC_SPRx)

寄存器	偏移	R/W	描述	复位值
RTC_SPR0	RTC_BA+0x40	R/W	RTC备用寄存器0	0x0000_0000
RTC_SPR1	RTC_BA+0x44	R/W	RTC备用寄存器1	0x0000_0000
RTC_SPR2	RTC_BA+0x48	R/W	RTC备用寄存器2	0x0000_0000
RTC_SPR3	RTC_BA+0x4C	R/W	RTC备用寄存器3	0x0000_0000
RTC_SPR4	RTC_BA+0x50	R/W	RTC备用寄存器4	0x0000_0000

31	30	29	28	27	26	25	24
SPARE							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
SPARE							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
SPARE							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
SPARE							

位	描述	
[31:0]	**SPARE**	**备用寄存器** 这个区域用于存储和备份用户的信息。 一旦tamper引脚事件被侦测到，这部分区域的内容将会被硬件自动清除。在向备用寄存器存储备份信息之前，用户应该确认使能位REWNF (RTC_RWEN[16])处于使能状态。

RTC 32K晶振控制寄存器(RTC_LXTCTL)

寄存器	偏移	R/W	描述	复位值
RTC_LXTCTL	RTC_BA+0x100	R/W	RTC 32.768 kHz晶振控制寄存器	0xXXXX_XX06

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
C32KS	Reserved			GAIN			LIRC32KEN

位	描述	
[31:4]	Reserved	保留
[3:1]	GAIN	**晶振增益选择** 根据晶振外部负载和工作温度范围,用户可以选择晶振增益。增益值越大对应越强的驱动能力和更高的功耗. 000 = L0模式 001 = L1模式 010 = L2模式 011 = L3模式(默认) 100 = L4模式 101 = L5模式 110 = L6模式 111 = L7模式
[0]	LIRC32KEN	**LIRC 32K源使能位** 0 = LIRC32K禁用 1 = LIRC32K使能 **注：**LIRC32K在打开之后不可以再关闭，除非VBAT有POR复位。

RTC GPIO控制寄存器0 (RTC_GPIOCTL0)

寄存器	偏移	R/W	描述	复位值
RTC_GPIOCTL0	RTC_BA+0x104	R/W	RTC GPIO控制寄存器0	0x0000_0000

31	30	29	28	27	26	25	24
Reserved		PUSEL3		CTLSEL3	DOUT3	OPMODE3	
23	22	21	20	19	18	17	16
Reserved		PUSEL2		CTLSEL2	DOUT2	OPMODE2	
15	14	13	12	11	10	9	8
Reserved		PUSEL1		CTLSEL1	DOUT1	OPMODE1	
7	6	5	4	3	2	1	0
Reserved		PUSEL0		CTLSEL0	DOUT0	OPMODE0	

位	描述	
[31:30]	**Reserved**	保留
[29:28]	**PUSEL3**	**IO 上拉和下拉使能** 决定 PF.7 I/O 上拉或下拉 00 = PF.7 上拉和下拉禁止 01 = PF.7 上拉使能 10 = PF.7 下拉使能 11 = PF.7上拉和下拉禁止 **注：**基本上，上拉控制和下拉控制有如下限制 仅当OPMODE3设置成三态或开漏模式时，单独上拉/下拉控制才有效。
[27]	**CTLSEL3**	**IO 管脚状态备用选择** 当 TAMP1EN 禁止， PF.7 管脚 (TAMPER1 管脚) 可以用作 GPIO 功能。用户可以编程 CTLSEL3 来决定 PF.7 I/O 功能是由系统电源域GPIO模块控制还是由V_{BAT}电源域 RTC_GPIOCTL0 控制寄存器控制。 0 = PF.7 管脚 I/O 功能由 GPIO 模块控制. 当系统电源关闭，硬件自动变CTLSEL3 = 1 1 = PF.7 管脚 I/O 功能由V_{BAT}电源域控制 CTLSEL3置1后，PF.7 管脚功能和I/O 状态由 OPMODE3[1:0] 和DOUT3控制。 **注：**当系统电源关闭且RTC_INIT[0] (RTC激活状态)是1，CTLSEL3会被硬件自动置1。
[26]	**DOUT3**	**IO 输出数据** 0 = PF.7 输出低 1 = PF.7 输出高
[25:24]	**OPMODE3**	**IO 操作模式** 00 = PF.7是仅输入模式

		01 = PF.7是推挽输出模式 10 = PF.7 是开漏模式. 11 = PF.7 是准双向模式
[23:22]	**Reserved**	保留
[21:20]	**PUSEL2**	**IO 上拉和下拉使能** 决定 PF.6 I/O 上拉或下拉 00 = PF.6上拉和下拉禁止. 01 = PF.6上拉使能. 10 = PF.6下拉使能. 11 = PF.6上拉和下拉禁止 **注：**基本上，上拉控制和下拉控制有如下限制 仅当OPMODE2设置成三态或开漏模式时，单独上拉/下拉控制才有效。
[19]	**CTLSEL2**	**IO 管脚状态备用选择** 当TAMP0EN禁止， PF.6管脚 (TAMPER0管脚) 可以用作 GPIO 功能。用户可以编程CTLSEL2来决定 PF.6 I/O 功能是由系统电源域GPIO模块控制还是由V_{BAT}电源域RTC_GPIOCTL0 控制寄存器控制。 0 = PF.6 管脚 I/O 功能由 GPIO 模块控制. 当系统电源关闭，硬件自动变CTLSEL2 = 1. 1 = PF.6管脚 I/O 功能由V_{BAT}电源域控制 CTLSEL2置1后，PF.6管脚功能和I/O 状态由OPMODE2 [1:0] 和DOUT2控制。 **注：**当系统电源关闭且RTC_INIT[0] (RTC激活状态)是1，CTLSEL2会被硬件自动置1。
[18]	**DOUT2**	**IO 输出数据** 0 = PF.6 输出低. 1 = PF.6 输出高.
[17:16]	**OPMODE2**	**IO 操作模式** 00 = PF.6是仅输入模式 01 = PF.6是推挽输出模式 10 = PF.6 是开漏模式 11 = PF.6 是准双向模式
[15:14]	**Reserved**	保留
[13:12]	**PUSEL1**	**IO上拉和下拉使能位** 决定PF.5 I/O上拉或下拉 00 = PF.5上拉和下拉禁止 01 = PF.5上拉使能 10 = PF.5下拉使能 11 = PF.5上拉和下拉禁止 **注：**基本上，上拉控制和下拉控制有如下限制 仅当OPMODE1设置成三态或开漏模式时，单独上拉/下拉控制才有效。
[11]	**CTLSEL1**	**IO 管脚状态备用选择** 当32 kHz晶振禁止， PF.5 管脚 (X32KI管脚) 可以用作 GPIO 功能。用户可以编程CTLSEL1来决定 PF.5 I/O功能是由系统电源域GPIO模块控制还是由V_{BAT}电源域RTC_GPIOCTL0 控制寄存器控制。

		0 = PF.5 管脚 I/O 功能由 GPIO 模块控制 当系统电源关闭，硬件自动变CTLSEL1 = 1。 1 = PF.5管脚 I/O 功能由V_{BAT}电源域控制 CTLSEL1置1后，PF.5管脚功能和I/O 状态由OPMODE1 [1:0] 和DOUT1控制。 **注：**当系统电源关闭且RTC_INIT[0] (RTC激活状态)是1，CTLSEL1会被硬件自动置1。
[10]	**DOUT1**	**IO 输出数据** 0 = PF.5 输出低 1 = PF.5 输出高
[9:8]	**OPMODE1**	**IO 操作模式** 00 = PF.5是仅输入模式 01 = PF.5是推挽输出模式 10 = PF.5 是开漏模式 11 = PF.5 是准双向模式
[7:6]	**Reserved**	保留
[5:4]	**PUSEL0**	**IO 上拉和下拉使能** 决定 PF.4 I/O 上拉或下拉 00 = PF.4上拉和下拉禁止 01 = PF.4上拉使能 11 = PF.4上拉和下拉禁止 **注：**基本上，上拉控制和下拉控制有如下限制 仅当OPMODE0设置成三态或开漏模式时，单独上拉/下拉控制才有效。
[3]	**CTLSEL0**	**IO 管脚状态备用选择** 当32 kHz晶振禁止， PF.4 管脚 (X32KO管脚) 可以用作 GPIO 功能。用户可以编程CTLSEL0来决定 PF.4 I/O 功能是由系统电源域GPIO模块控制还是由V_{BAT}电源域RTC_GPIOCTL0 控制寄存器控制。 0 = PF.4 管脚 I/O 功能由 GPIO 模块控制 当系统电源关闭，硬件自动变CTLSEL0 = 1 1 = PF.4管脚 I/O 功能由V_{BAT}电源域控制 CTLSEL0置1后，PF.4管脚功能和I/O 状态由OPMODE0 [1:0] 和DOUT0控制。 **注：**当系统电源关闭且RTC_INIT[0] (RTC激活状态)是1，CTLSEL0会被硬件自动置1。
[2]	**DOUT0**	**IO 输出数据** 0 = PF.4 输出低 1 = PF.4 输出高
[1:0]	**OPMODE0**	**IO 操作模式** 00 = PF.4是仅输入模式，不带上拉电阻 01 = PF.4是推挽输出模式 10 = PF.4是开漏模式. 11 = PF.4 是准双向模式带内部上拉

注：如果没有编程，RTC GPIO将为输入模式，用户必须避免引脚浮空问题。

RTC GPIO控制寄存器1 (RTC_GPIOCTL1)

寄存器	偏移	R/W	描述	复位值
RTC_GPIOCTL1	RTC_BA+0x108	R/W	RTC GPIO控制寄存器1	0x0000_0000

31	30	29	28	27	26	25	24
Reserved		PUSEL7		CTLSEL7	DOUT7	OPMODE7	
23	22	21	20	19	18	17	16
Reserved		PUSEL6		CTLSEL6	DOUT6	OPMODE6	
15	14	13	12	11	10	9	8
Reserved		PUSEL5		CTLSEL5	DOUT5	OPMODE5	
7	6	5	4	3	2	1	0
Reserved		PUSEL4		CTLSEL4	DOUT4	OPMODE4	

位	描述	
[31:30]	**Reserved**	保留
[29:28]	**PUSEL7**	**IO 上拉和下拉使能** 决定 PF.11 I/O 上拉或下拉 00 = PF.11上拉和下拉禁止 01 = PF.11上拉使能 11 = PF.11上拉和下拉禁止 **注：**基本上，上拉控制和下拉控制有如下限制 仅当OPMODE7设置成三态或开漏模式时，单独上拉/下拉控制才有效。
[27]	**CTLSEL7**	**I/O管脚状态备用选择** 当TAMP5EN禁止， PF.11管脚 (TAMPER5管脚) 可以用作 GPIO 功能。用户可以编程CTLSEL7来决定 PF.11 I/O 功能是由系统电源域GPIO模块控制还是由V_{BAT}电源域RTC_GPIOCTL1 控制寄存器控制。 0 = PF.11 管脚 I/O 功能由 GPIO 模块控制 当系统电源关闭，硬件自动变CTLSEL7 = 1 1 = PF.11管脚 I/O 功能由V_{BAT}电源域控制 CTLSEL7置1后，PF.11管脚功能和I/O 状态由OPMODE7 [1:0] 和DOUT7控制。 **注：**当系统电源关闭且RTC_INIT[0] (RTC激活状态)是1，CTLSEL7会被硬件自动置1。
[26]	**DOUT7**	**I/O输出数据** 0 = PF.11 输出低 1 = PF.11 输出高
[25:24]	**OPMODE7**	**I/O操作模式** 00 = PF.11是仅输入模式 01 = PF.11是推挽输出模式

		10 = PF.11 是开漏模式 11 = PF.11 是准双向模式
[23:22]	**Reserved**	保留
[21:20]	**PUSEL6**	**I/O上拉和下拉使能** 决定 PF.10 I/O 上拉或下拉 00 = PF.10上拉和下拉禁止 01 = PF.10上拉使能 10 = PF.10下拉使能 11 = PF.10上拉和下拉禁止 **注：**基本上，上拉控制和下拉控制有如下限制 仅当OPMODE6设置成三态或开漏模式时，单独上拉/下拉控制才有效。
[19]	**CTLSEL6**	**I/O管脚状态备用选择** 当TAMP4EN禁止， PF.10管脚 (TAMPER4管脚) 可以用作 GPIO 功能。用户可以编程CTLSEL6来决定 PF.10 I/O 功能是由系统电源域GPIO模块控制还是由V_{BAT}电源域RTC_GPIOCTL1 控制寄存器控制。 0 = PF.10 管脚 I/O 功能由 GPIO 模块控制 当系统电源关闭，硬件自动变CTLSEL6 = 1. 1 = PF.10管脚 I/O 功能由V_{BAT}电源域控制 CTLSEL6置1后，PF.10管脚功能和I/O 状态由OPMODE6 [1:0] 和DOUT6控制。 **注：**当系统电源关闭且RTC_INIT[0] (RTC Active Status)是1，CTLSEL6会被硬件自动置1。
[18]	**DOUT6**	**I/O输出数据** 0 = PF.10 输出低 1 = PF.10 输出高
[17:16]	**OPMODE6**	**I/O操作模式** 00 = PF.10是仅输入模式 01 = PF.10是推挽输出模式 10 = PF.10 是开漏模式 11 = PF.10 是准双向模式
[15:14]	**Reserved**	保留
[13:12]	**PUSEL5**	**I/O上拉和下拉使能** 决定 PF.9 I/O 上拉或下拉 00 = PF.9上拉和下拉禁止 01 = PF.9 上拉使能 10 = PF.9下拉使能 11 = PF.9上拉和下拉禁止 **注：**基本上，上拉控制和下拉控制有如下限制 仅当OPMODE5设置成三态或开漏模式时，单独上拉/下拉控制才有效。
[11]	**CTLSEL5**	**I/O管脚状态备用选择** 当TAMP3EN禁止， PF.9管脚 (TAMPER3管脚) 可以用作 GPIO 功能。用户可以编程CTLSEL5来决定 PF.9 I/O 功能是由系统电源域GPIO模块控制还是由V_{BAT}电源域RTC_GPIOCTL1 控制寄存器控制。

		0 = PF.9 管脚 I/O 功能由 GPIO 模块控制 当系统电源关闭，硬件自动变CTLSEL5 = 1 1 = PF.9管脚 I/O 功能由V_{BAT}电源域控制 CTLSEL5置1后，PF.11管脚功能和I/O 状态由OPMODE5 [1:0] 和DOUT5控制。 **注：**当系统电源关闭且RTC_INIT[0] (RTC Active Status)是1，CTLSEL5会被硬件自动置1。
[10]	**DOUT5**	**I/O输出数据** 0 = PF.9 输出低 1 = PF.9 输出高
[9:8]	**OPMODE5**	**I/O操作模式** 00 = PF.9是仅输入模式 01 = PF.9是推挽输出模式 10 = PF.9 是开漏模式 11 = PF.9 是准双向模式
[7:6]	**Reserved**	保留
[5:4]	**PUSEL4**	**I/O上拉和下拉使能** 决定 PF.8 I/O 上拉或下拉 00 = PF.8上拉和下拉禁止 01 = PF.8 上拉下拉使能 10 = PF.8下拉使能 11 = PF.8上拉和下拉禁止 **注:**基本上，上拉控制和下拉控制有如下限制 仅当OPMODE4设置成三态或开漏模式时，单独上拉/下拉控制才有效。
[3]	**CTLSEL4**	**I/O管脚状态备用选择** 当TAMP2EN禁止， PF.8管脚 (TAMPER2管脚) 可以用作 GPIO 功能。用户可以编程CTLSEL4来决定 PF.8 I/O 功能是由系统电源域GPIO模块控制还是由V_{BAT}电源域RTC_GPIOCTL1 控制寄存器控制。 0 = PF.8 管脚 I/O 功能由 GPIO 模块控制 当系统电源关闭，硬件自动变CTLSEL4 = 1. 1 = PF.8管脚 I/O 功能由V_{BAT}电源域控制 CTLSEL4置1后，PF.8管脚功能和I/O 状态由OPMODE4 [1:0] 和DOUT4控制。 **注：**当系统电源关闭且RTC_INIT[0] (RTC Active Status)是1，CTLSEL4会被硬件自动置1。
[2]	**DOUT4**	**I/O输出数据** 0 = PF.8 输出低 1 = PF.8 输出高
[1:0]	**OPMODE4**	**I/O操作模式** 00 = PF.8是仅输入模式 01 = PF.8是推挽输出模式 10 = PF.8 是开漏模式 11 = PF.8 是准双向模式

注：如果没有编程，RTC GPIO将为输入模式，用户必须避免引脚浮空问题。

RTC夏令时控制寄存器(RTC_DSTCTL)

寄存器	偏移	R/W	描述	复位值
RTC_DSTCTL	RTC_BA+0x110	R/W	RTC夏令时控制寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved					DSBAK	SUBHR	ADDHR

位	描述	
[31:3]	**Reserved**	保留
[2]	**DSBAK**	**夏令时更改** 0= 不执行夏令时更改 1= 执行夏令时更改
[1]	**SUBHR**	**减 1 小时** 0 = 无影响 1 = 表示RTC小时数字已经被减掉一个小时用于冬季时间的变化
[0]	**ADDHR**	**加 1 小时** 0 = 无影响 1 = 表示RTC小时数字已经被加上一个小时用于夏季时间的变化

RTC Tamper管脚控制寄存器 (RTC_TAMPCTL)

寄存器	偏移	R/W	描述	复位值
RTC_TAMPCTL	RTC_BA+0x120	R/W	RTC Tamper管脚控制寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved	TAMP5DBEN	TAMP5LV	TAMP5EN	Reserved	TAMP4DBEN	TAMP4LV	TAMP4EN
23	22	21	20	19	18	17	16
Reserved	TAMP3DBEN	TAMP3LV	TAMP3EN	Reserved	TAMP2DBEN	TAMP2LV	TAMP2EN
15	14	13	12	11	10	9	8
Reserved	TAMP1DBEN	TAMP1LV	TAMP1EN	Reserved	TAMP0DBEN	TAMP0LV	TAMP0EN
7	6	5	4	3	2	1	0
Reserved							

位	描述	
6.11.7.1 [31]	**Reserved**	保留
[30]	**TAMP5DBEN**	**Tamper 5 消抖使能位** 0 = Tamper 5 消抖禁止 1 = Tamper 5 消抖使能，Tamper检测引脚将同步1个RTC时钟
[29]	**TAMP5LV**	**Tamper 5 电平** 该位取决于用于静态tampter 侦测的tampter 管脚的电平属性 0 = 侦测电平是低 1 = 侦测电平是高
[28]	**TAMP5EN**	**Tamper 5 侦测使能位** 0 = Tamper 5 侦测禁止 1 = Tamper 5 侦测使能 **注：** 参考是 RTC时钟，Tamper 侦测需要同步2 ~ 3 个RTC时钟。
[27]	**Reserved**	保留
[26]	**TAMP4DBEN**	**Tamper 4消抖使能位** 0 = Tamper 4消抖禁止 1 = Tamper 4消抖使能，Tamper检测引脚将同步1个RTC时钟
[25]	**TAMP4LV**	**Tamper 4 电平** 该位取决于用于静态tampter 侦测的tampter 管脚的电平属性 0 = 侦测电平是低 1 = 侦测电平是高
[24]	**TAMP4EN**	**Tamper 4侦测使能位**

		0 = Tamper 4侦测禁止 1 = Tamper 4侦测使能 **注：** 参考是 RTC时钟， Tamper 侦测需要同步2 ~ 3 个RTC时钟.
[23]	**Reserved**	保留
[22]	**TAMP3DBEN**	**Tamper 3消抖使能位** 0 = Tamper 3消抖禁止. 1 = Tamper 3消抖使能，Tamper检测引脚将同步1个RTC时钟
[21]	**TAMP3LV**	**Tamper 3 电平** 该位取决于用于静态tampter 侦测的tampter 管脚的电平属性 0 = 侦测电平是低 1 = 侦测电平是高
[20]	**TAMP3EN**	**Tamper 3侦测使能位** 0 = Tamper 3侦测禁止 1 = Tamper 3侦测使能 **注：** 参考是 RTC时钟， Tamper 侦测需要同步2 ~ 3 个RTC时钟
[19]	**Reserved**	保留
[18]	**TAMP2DBEN**	**Tamper 2消抖使能位** 0 = Tamper 2消抖禁止 1 = Tamper 2消抖使能，Tamper检测引脚将同步1个RTC时钟
[17]	**TAMP2LV**	**Tamper 2 电平** 该位取决于用于静态tampter 侦测的tampter 管脚的电平属性 0 = 侦测电平是低 1 = 侦测电平是高
[16]	**TAMP2EN**	**Tamper 2侦测使能位** 0 = Tamper 2侦测禁止 1 = Tamper 2侦测使能 **注：** 参考是 RTC时钟，Tamper 侦测需要同步2 ~ 3 个RTC时钟
[15]	**Reserved**	保留
[14]	**TAMP1DBEN**	**Tamper 1消抖使能位** 0 = Tamper 1消抖禁止 1 = Tamper 1消抖使能，Tamper检测引脚将同步1个RTC时钟
[13]	**TAMP1LV**	**Tamper 1 电平** 该位取决于用于静态tampter 侦测的tampter 管脚的电平属性 0 = 侦测电平是低 1 = 侦测电平是高
[12]	**TAMP1EN**	**Tamper 1侦测使能位** 0 = Tamper 1侦测禁止 1 = Tamper 1侦测使能 **注：** 参考是 RTC时钟，Tamper 侦测需要同步2 ~ 3 个RTC时钟

[11]	**Reserved**	保留
[10]	**TAMP0DBEN**	**Tamper 0消抖使能位** 0 = Tamper 0消抖禁止 1 = Tamper 0消抖使能，Tamper检测引脚将同步1个RTC时钟
[9]	**TAMP0LV**	**Tamper 0 电平** 该位取决于用于静态tampter 侦测的tampter 管脚的电平属性 0 = 侦测电平是低 1 = 侦测电平是高
[8]	**TAMP0EN**	**Tamper 0侦测使能位** 0 = Tamper 0侦测禁止 1 = Tamper 0侦测使能 **注：**参考是 RTC时钟，Tamper 侦测需要同步2 ~ 3 个RTC时钟
[7:0]	**Reserved**	保留

RTC Tamper时间寄存器(RTC_TAMPTIME)

寄存器	偏移	R/W	描述	复位值
RTC_TAMPTIME	RTC_BA+0x130	R	RTC Tamper时间寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved	HZCNT						
23	22	21	20	19	18	17	16
Reserved		TENHR		HR			
15	14	13	12	11	10	9	8
Reserved	TENMIN			MIN			
7	6	5	4	3	2	1	0
Reserved	TENSEC			SEC			

位	描述	
[31]	**Reserved**	保留
[30:24]	**HZCNT**	亚秒计数器(0x00 ~0x7F)
[21:20]	**TENHR**	TAMPER 时间小时的十位 (0~2) 注：仅支持24小时制
[19:16]	**HR**	TAMPER 时间小时的个位(0~9)
[15]	**Reserved**	保留
[14:12]	**TENMIN**	TAMPER 时间分钟的十位(0~5)
[11:8]	**MIN**	TAMPER 时间分钟的个位(0~9)
[7]	**Reserved**	保留
[6:4]	**TENSEC**	TAMPER 时间秒的十位(0~5)
[3:0]	**SEC**	TAMPER 时间秒的个位(0~9)

注：

1. RTC_TAMPTIME是BCD格式数字计数器.
2. 数值的合理范围在括号中列出了
3. 该域不能更新直到所有TAMPxIF 被清除

RTC Tamper日期寄存器(RTC_TAMPCAL)

寄存器	偏移	R/W	描述	复位值
RTC_TAMPCAL	RTC_BA+0x134	R	RTC Tamper日期寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
TENYEAR				YEAR			
15	14	13	12	11	10	9	8
Reserved			TENMON	MON			
7	6	5	4	3	2	1	0
Reserved		TENDAY		DAY			

位	描述	
[31:24]	**Reserved**	保留
[23:20]	**TENYEAR**	TAMPER 日期年的十位 (0~9)
[19:16]	**YEAR**	TAMPER日期年的个位(0~9)
[15:13]	**Reserved**	保留
[12]	**TENMON**	TAMPER 日期月的十位 (0~1)
[11:8]	**MON**	TAMPER 日期月的个位 (0~9)
[7:6]	**Reserved**	保留
[5:4]	**TENDAY**	TAMPER 日期日的十位 (0~3)
[3:0]	**DAY**	TAMPER日期日的个位(0~9)

注：

1. RTC_TAMPCAL是BCD格式数字计数器
2. 数值的合理范围在括号中列出了
3. 该域不能更新直到所有TAMPxIF 被清除

6.12 EPWM发生器和捕获定时器(EPWM)

6.12.1 概述

该芯片提供了两路EPWM发生器EPWM0 和 EPWM1。每路EPWM支持6通道EPWM输出或输入捕获。有一个12位的预分频器把时钟源分频后输入给16位的计数器，另外还有一个16位的比较器。EPWM计数器支持向上、向下、上下计数方式。EPWM用比较器和计数器的比较来产生事件，这些事件用来产生EPWM脉冲、中断、EADC/DAC转换触发信号。

EPWM发生器支持两种标准EPWM输出模式：独立模式和互补模式，它们的架构不同。标准输出模式又有两种输出功能：组功能和同步功能。组功能可以在独立模式和互补模式下使能。同步功能只有在互补模式下才可以被使能。互补模式，有两个比较器产生各种带12位死区时间的EPWM脉宽，另外还有一个自由触发比较器来产生给EADC的触发信号。EPWM输出控制单元，它支持极性输出，独立管脚屏蔽和刹车功能。

EPWM也支持输入捕获功能，当输入通道有向上跳变、向下跳变、或者两者都有的跳变时，锁存EPWM计数器的值到相应的寄存器中。捕获功能也支持通过PDMA把捕获到的数据搬移到内存。

6.12.2 特性

6.12.2.1 EPWM功能特性

- 支持最大时钟频率到PLL 的最大频率
- 支持两个EPWM模块，每个模块提供6个输出通道
- 支持独立模式的EPWM输出/输入捕获
- 支持3组互补通道的互补模式
  - 12位解析度的死区插入
  - 同步功能用于相位控制
  - 每个周期两个比较值
- 支持12位从1到4096的预分频
- 支持16位解析度的EPWM计数器
  - 向上，向下和上下计数操作类型
- 支持one-shot或自动装载计数器工作模式
- 支持组功能
- 支持同步功能
- 每个EPWM管脚支持屏蔽功能和三态使能
- 支持刹车功能
  - 刹车源来自管脚、模拟比较器和系统安全事件（时钟故障、SRAM奇偶校验错误、欠压监测和CPU锁死）
  - 刹车源管脚噪声滤波器
  - Leading edge blanking (LEB) 功能用于刹车源来自模拟比较器情况
  - 通过边沿检测刹车源来控制刹车状态直到刹车中断清除

- 刹车条件解除后通过电平检测刹车源来控制自动恢复功能

- 支持下列事件中断:
  - EPWM计数器值为 0、周期值或比较值
  - 发生刹车条件
- 支持下列事件触发EADC/DAC：
  - EPWM计数器值为0、周期值或比较值
  - EPWM 计数器匹配自由触发比较器比较值(仅EADC)
  - 支持EPWM触发EADC事件预分频特性
- 支持EPWM输出累加器停止计数器模式
- 支持错误检测功能

6.12.2.2 捕获功能特性

- 支持12个16位解析度的输入捕获通道
- 支持上升/下降沿捕获条件
- 支持输入上升/下降沿捕获中断
- 支持计数器重载选项的上升/下降沿捕获
- 支持EPWM 的所有通道PDMA数据搬移功能

6.12.3 框图

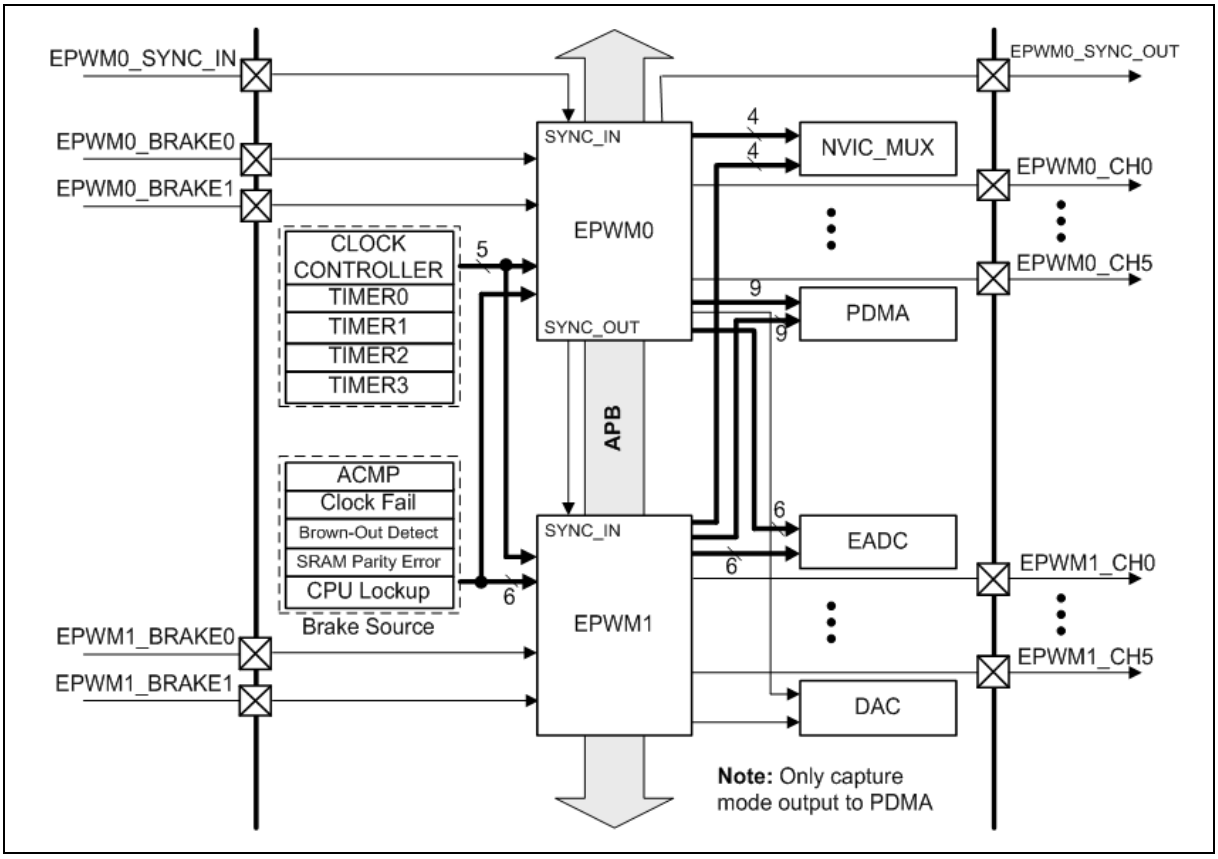


图 6.12-1EPWM 发生器简要方块图

EPWM系统时钟可以设为等于或两倍的PCLK频率，如图6.12-2，寄存器设置细节请参考表6.12-1。每个EPWM发生器有三个输入时钟源，每个时钟源可以选择来自EPWM时钟或者4组定时器触发EPWM输出，如图6.12-3，通过ECLKSRC0 (EPWM_CLKSRC[2:0]) 设置 EPWM_CLK0， ECLKSRC2 (EPWM_CLKSRC[10:8]) 设置 EPWM_CLK2 和 ECLKSRC4 (EPWM_CLKSRC[18:16]) 设置 EPWM_CLK4。

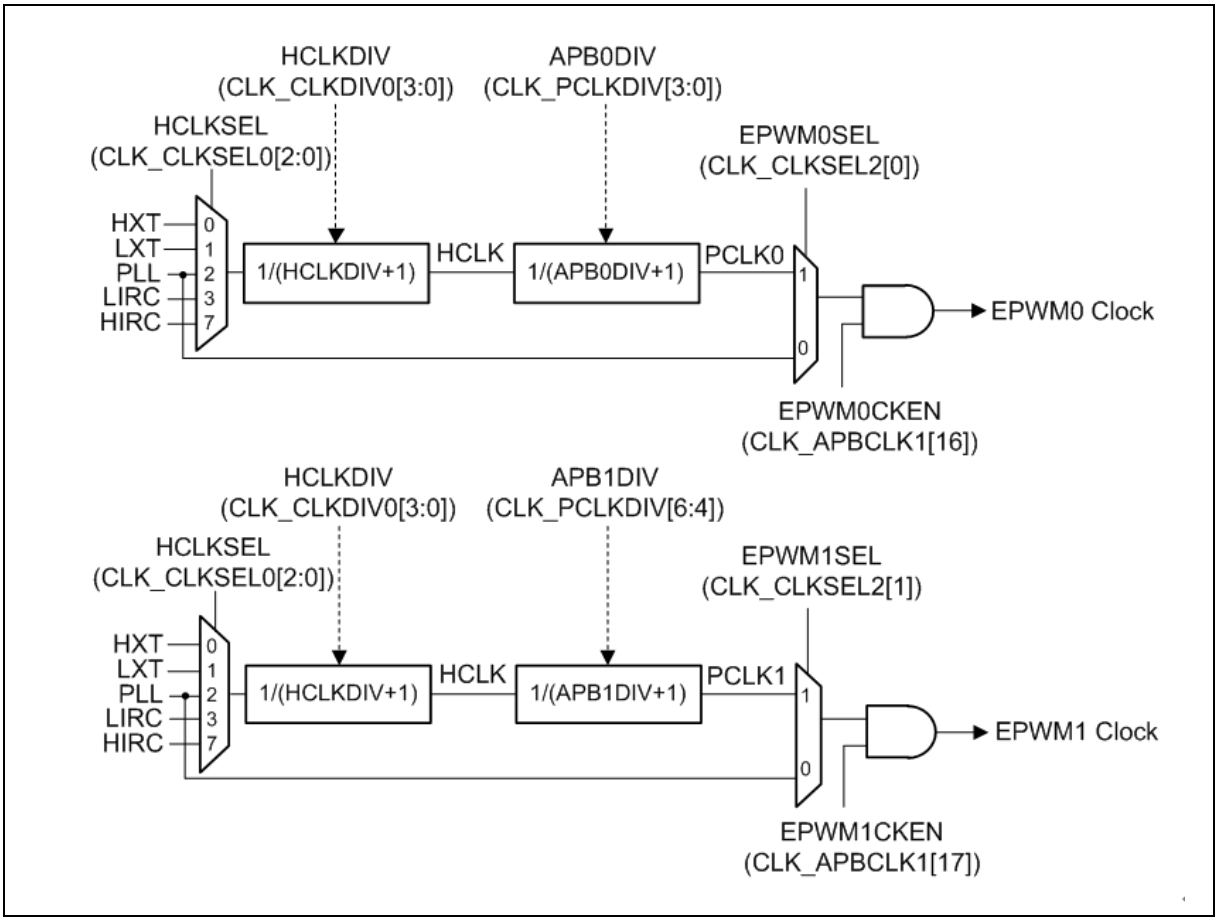


图 6.12-2 EPWM 时钟源控制

频率比 PCLK:EPWM时钟	HCLK	PCLK	EPWM 时钟	HCLKSEL CLK_CLKSEL0[2:0]	HCLKDIV CLK_CLKDIV0[3:0]	APBnDIV (CLK_PCLKDIVn [2+4n:4n]), N为0或1	EPWMnSEL (CLK_CLKSEL2[N]), N为0或1
1:1	HCLK	PCLK	PCLK	不用管	不用管	不用管	1
1:2	PLL	PLL/ 2	PLL	2	0	1	0
1:2	PLL/ 2	PLL/ 2	PLL	2	1	0	0

表 6.12-1EPWM 时钟源控制寄存器设置表

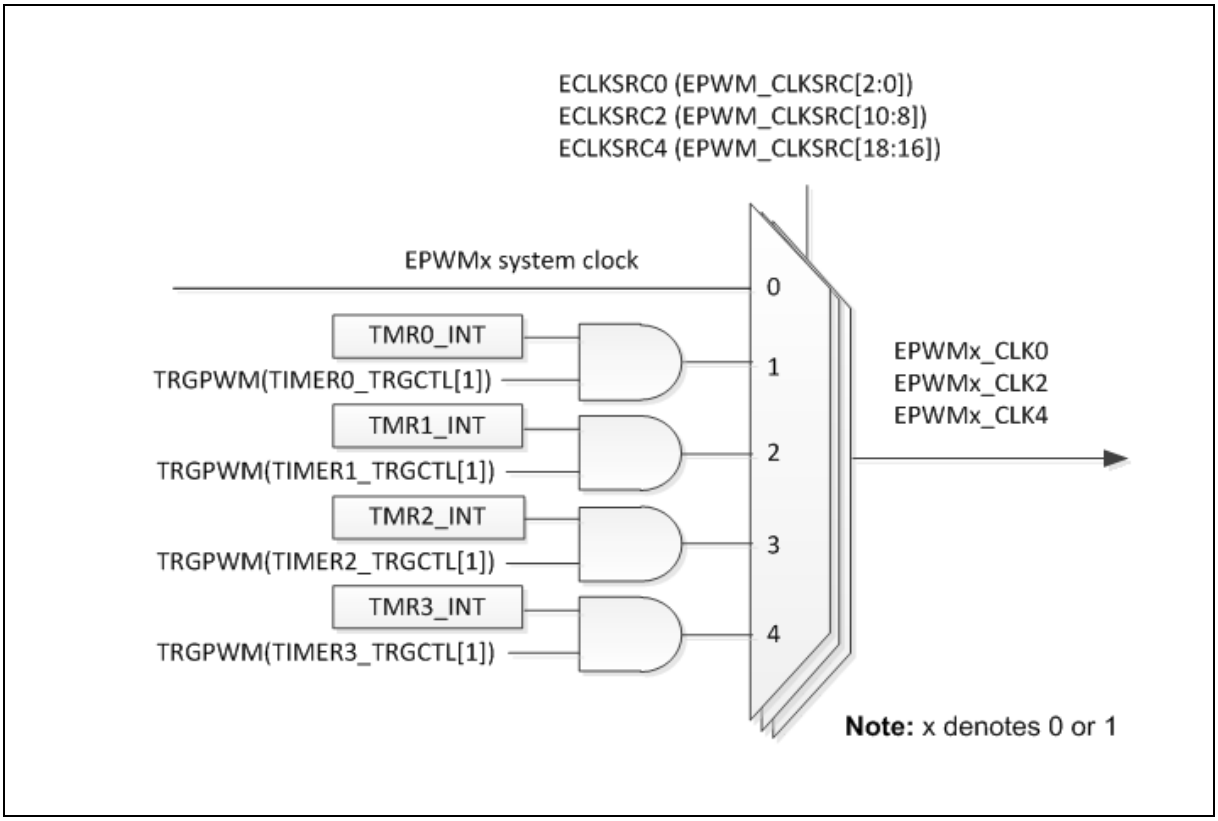


图 6.12-3EPWM 时钟源控制

图6.12-4和图6.12-5表示EPWM独立模式和互补模式的架构。不管独立模式还是互补模式，一对通道组（EPWM_CH0和EPWM_CH1、EPWM_CH2和EPWM_CH3、EPWM_CH4和EPWM_CH5）计数器来自相同的时钟源和预分频。当计数器的值等于0、PERIOD(EPWM_PERIODn[15:0])或比较器值，将产生事件。这些事件通过相应的发生器来产生EPWM脉冲、中断信号、EADC/DAC的转换触发信号。输出控制是用来改变EPWM脉冲输出状态的。输出控制的刹车功能也能产生中断事件。在互补模式，同步功能有效，偶数通道用奇数通道比较器产生事件，自由触发比较器事件只用于产生触发EADC信号。

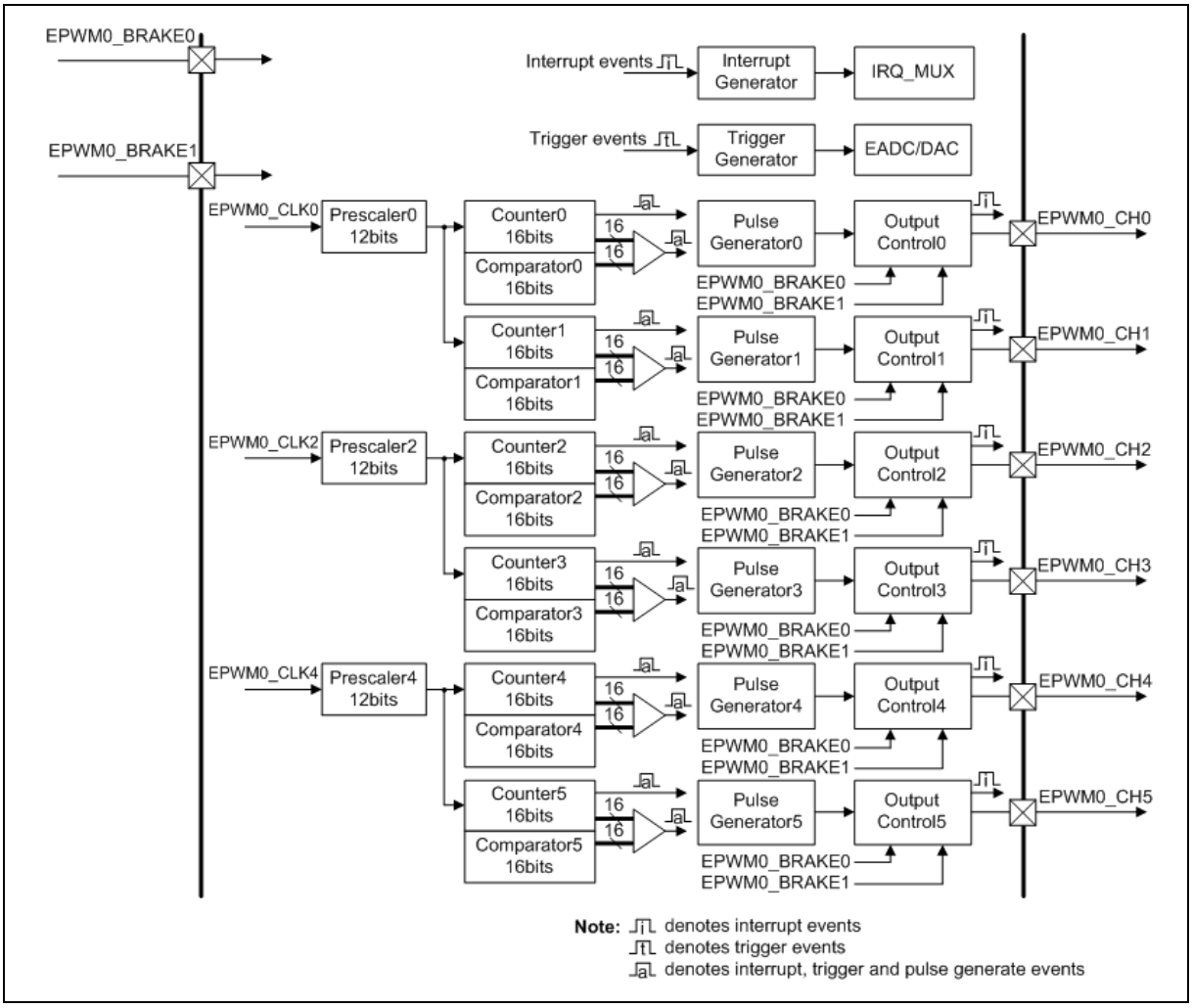


图 6.12-4EPWM 独立模式架构图

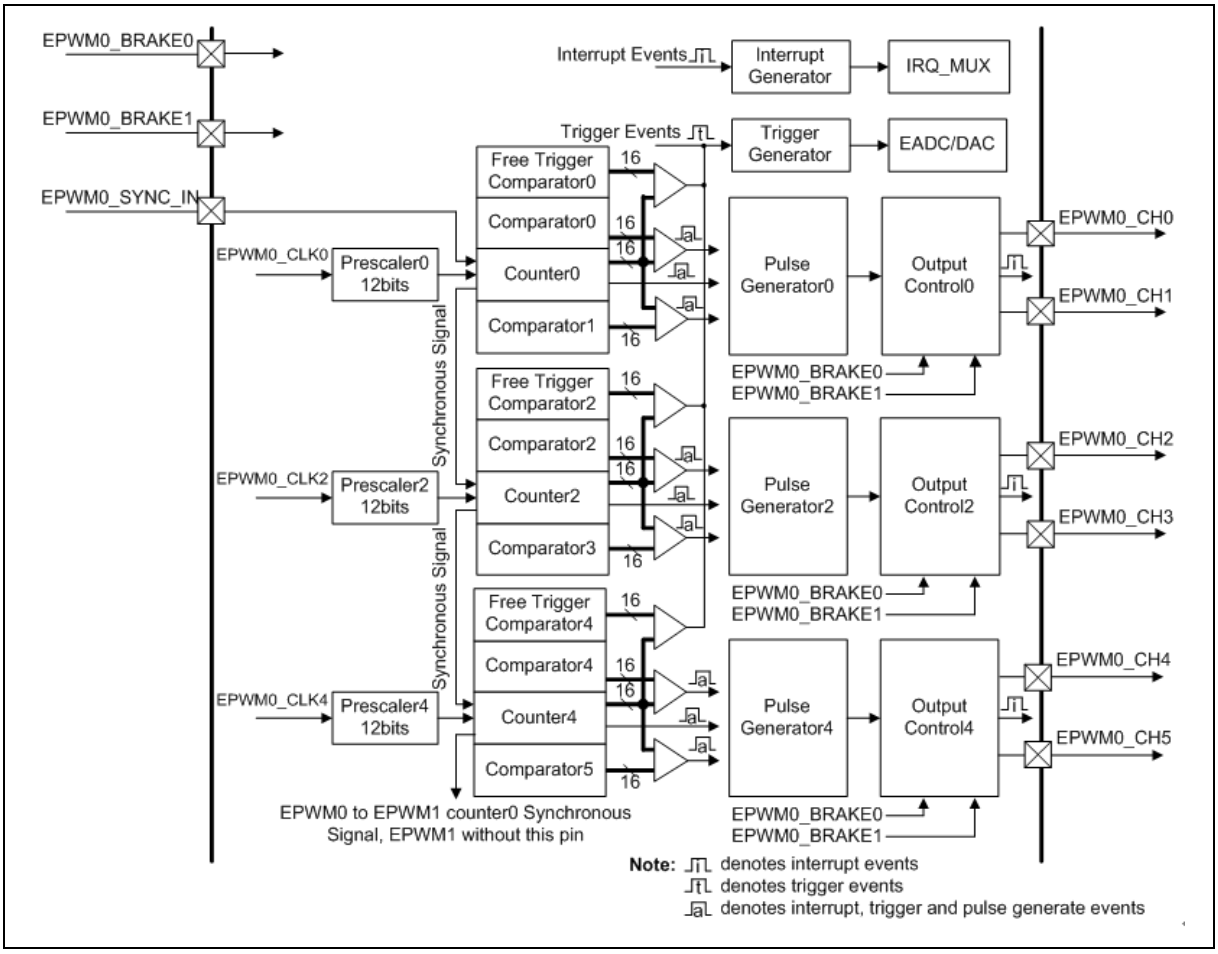


图 6.12-5EPWM 互补模式架构图

6.12.4 基本配置

6.12.4.1 EPWM0基本配置

- 时钟源配置
  - 通过EPWM0SEL (CLK_CLKSEL2[0])选择EPWM0外设时钟源
  - 通过EPWM0CKEN(CLK_APBCLK1[16])使能EPWM0外设时钟
- 复位配置
  - 通过EPWM0RST SYS_IPRST2[16]复位EPWM0
- 管脚配置

组	管脚名称	GPIO	MFP
EPWM0	EPWM0_BRAKE0	PE.8	MFP11
		PB.1	MFP13
	EPWM0_BRAKE1	PE.9	MFP11
		PB.0	MFP13

	EPWM0_CH0	PE.8	MFP10
		PB.5, PG.8	MFP11
		PE.7	MFP12
		PA.5	MFP13
	EPWM0_CH1	PE.9	MFP10
		PB.4, PG.7	MFP11
		PE.6	MFP12
		PA.4	MFP13
	EPWM0_CH2	PE.10	MFP10
		PB.3, PG.6	MFP11
		PE.5	MFP12
		PA.3	MFP13
	EPWM0_CH3	PE.11	MFP10
		PB.2, PG.5	MFP11
		PE.4	MFP12
		PA.2	MFP13
	EPWM0_CH4	PE.12	MFP10
		PB.1, PD.14	MFP11
		PE.3	MFP12
		PA.1	MFP13
	EPWM0_CH5	PE.13	MFP10
		PB.0, PH.11	MFP11
		PE.2	MFP12
		PA.0	MFP13
	EPWM0_SYNC_IN	PC.14	MFP11
		PA.15	MFP12
	EPWM0_SYNC_OUT	PF.5	MFP9
		PA.11	MFP10

6.12.4.2 EPWM1基本配置

- 时钟源配置
  - 通过EPWM1SEL (CLK_CLKSEL2[1])选择EPWM1外设时钟源
  - 通过EPWM1CKEN (CLK_APBCLK1[17])使能EPWM1外设时钟
- 复位配置
  - 通过EPWM1RST SYS_IPRST2[17]复位EPWM1

- 管脚配置

组	管脚名称	GPIO	MFP
EPWM1	EPWM1_BRAKE0	PB.7, PE.10	MFP11
	EPWM1_BRAKE1	PB.6, PE.11	MFP11
	EPWM1_CH0	PB.15, PE.13	MFP11
		PC.5, PC.12	MFP12
	EPWM1_CH1	PB.14, PC.8	MFP11
		PC.4, PC.11	MFP12
	EPWM1_CH2	PB.13, PC.7	MFP11
		PC.3, PC.10	MFP12
	EPWM1_CH3	PB.12, PC.6	MFP11
		PC.2, PC.9	MFP12
	EPWM1_CH4	PA.7	MFP11
		PB.1, PB.7, PC.1	MFP12
	EPWM1_CH5	PA.6	MFP11
		PB.0, PB.6, PC.0	MFP12

6.12.5 功能描述

6.12.5.1 EPWM预分频器

EPWM预分频器是用于时钟源除频，预分频器计数CLKPSC +1次，EPWM计数器只加一次。通过CLKPSC (EPWM_CLKPSCn[11:0]，n = 0、2、4)来设置预分频双缓存。图6.12-6是EPWM通道0预分频波形的例子。预分频计数器会在下一个预分频计数器下数开始时重载CLKPSC。

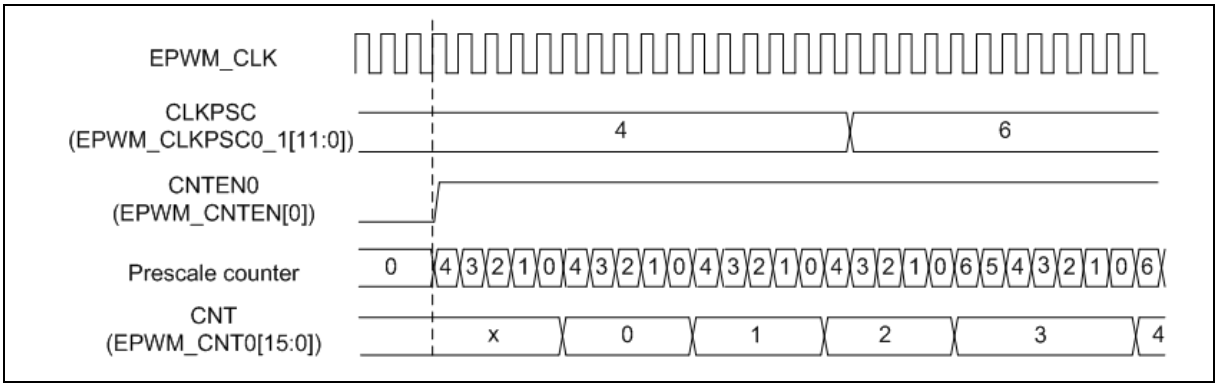


图 6.12-6EPWM_CH0 上计数模式的预分频器波形

6.12.5.2 EPWM计数器

EPWM支持3种计数方式操作：向上计数、向下计数、上下计数方式。

对于EPWM通道0，CNT(EPWM_CNT0[15:0])可以通过置位CNTCLR0 (EPWM_CNTCLR[0])来清0，当

预分频计数器数到0，CNT会被清除，且CNTCLR0被硬件自动置0。

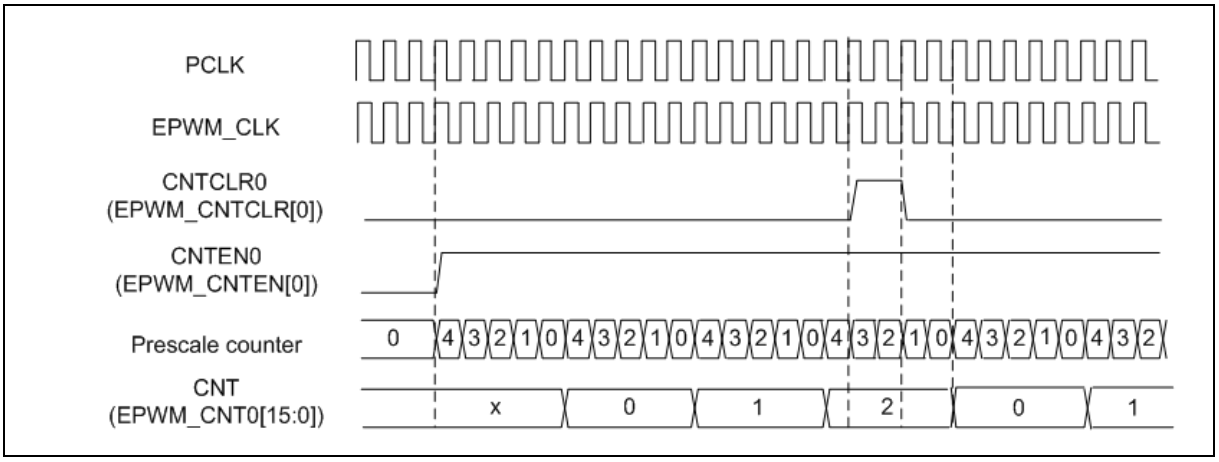


图 6.12-7 当设置清计数器 EPWMx 计数器的波形

6.12.5.3 上计数模式

在向上计数操作中，CNTTYPEn (EPWM_CTL1[2n+1:2n], n = 0,1..5) 是 0x0。16位EPWM计数器是一个向上计数器并从0开始向上计数到PERIOD (EPWM_PERIODn[15:0]，其中n表示通道数)来完成一个EPWM周期。当前计数值可以从CNT (EPWM_CNTn[15:0])读出。当计数器计数到0且预分频计数到0将产生零点事件，当计数到PERIOD且预分频计数到0将产生周期点事件。如下图显示一个向上计数器例子：

EPWM周期 = (PERIOD+1) *(CLKPSC+1)*EPWMx_CLK

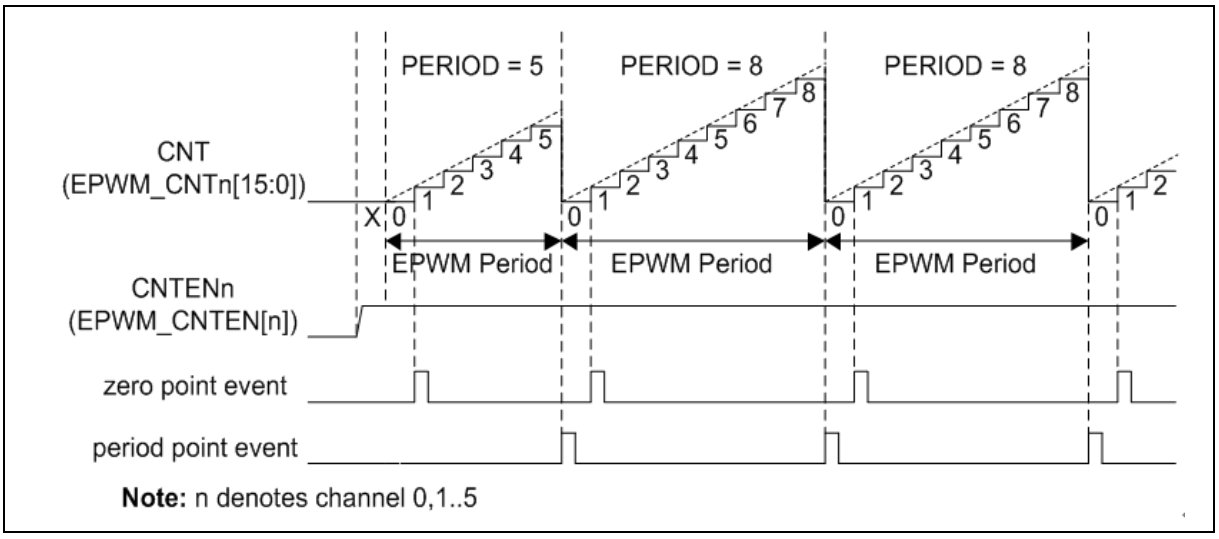


图 6.12-8EPWM 上计数器模式

6.12.5.4 下计数模式

在向下计数方式中，CNTTYPEn (EPWM_CTL1[2n+1:2n], n = 0,1..5) 是 0x1，16位EPWM计数器是一个向下计数器并从PERIOD开始向下计数到0来完成一个EPWM周期。当前计数值可以从CNT (EPWM_CNTn[15:0])读出。当计数器计数到0且预分频计数到0将产生零点事件，当计数到PERIOD且预分频计数到0将产生周期事件，如下图显示一个向下计数器例子：

EPWM周期 = (PERIOD+1) *(CLKPSC+1) * EPWMx_CLK

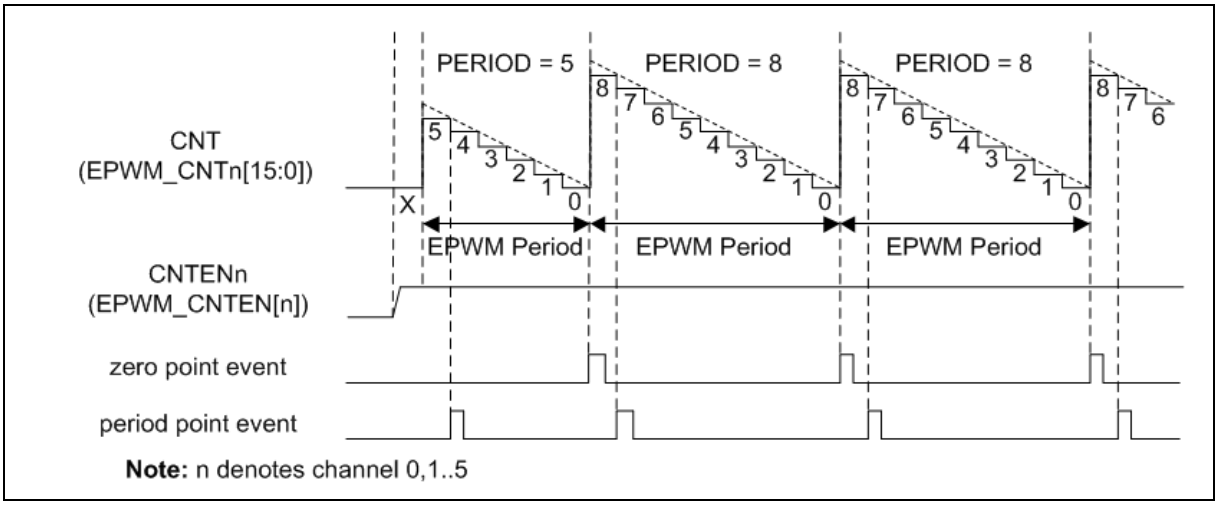


图 6.12-9EPWM 下计数器模式

6.12.5.5 上下计数模式

在上下计数操作中，CNTTYPEn (EPWM_CTL1[2n+1:2n], n = 0,1..5) 是 0x2，16位EPWM计数器是一个上下计数器，并开始向上计数从0到PERIOD然后又向下计数到0完成一个EPWM周期。当前计数值可以从CNT读出。当计数器计数到0且预分频计数到0将产生零点事件，当计数到PERIOD将产生中心点事件，如下图显示一个上下计数器例子：

EPWM周期 =(2*PERIOD) *(CLKPSC+1) * EPWMx_CLK

DIRF (EPWM_CNTn[16])是计数器方向标志，高是向上计数，低是向下计数。

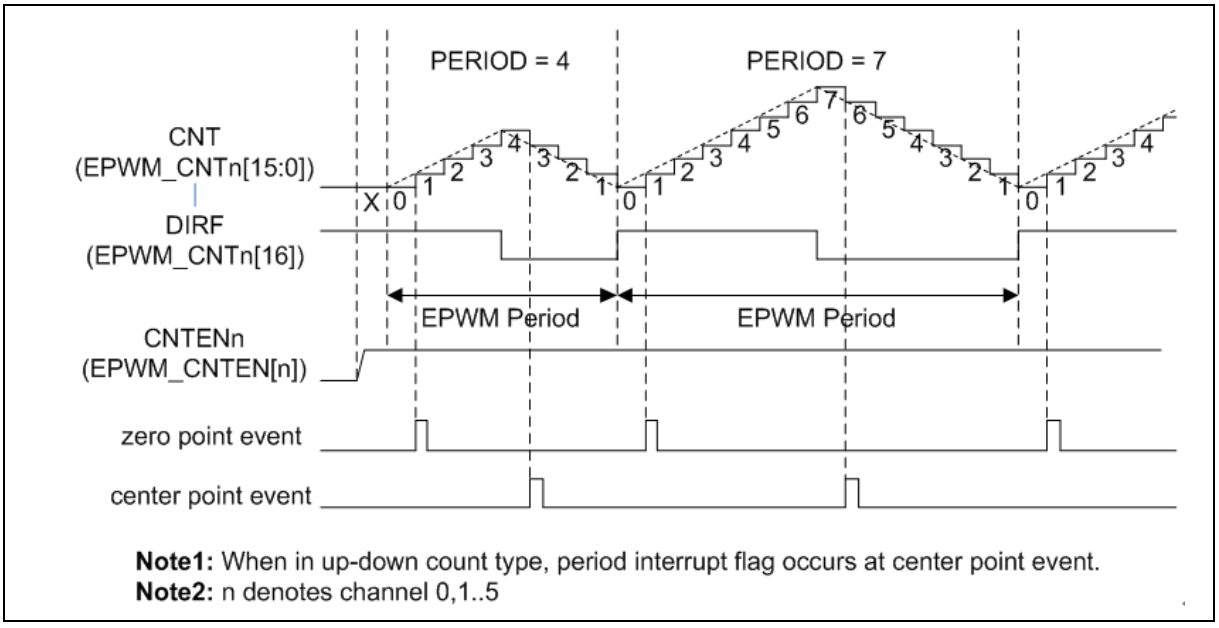


图 6.12-10EPWM 上下计数器模式

6.12.5.6 EPWM比较器

EPWM模块有两个比较器寄存器，一个是CMPDATn(n = 0,1..5)，另一个是FTCMPDATn_m(n = 0,2,4, m = 1,3,5)。CMPDATn是EPWM通道n的基本比较器寄存器。独立模式下每个通道只有一个CMPDATn。CMPDATn值一直与相应通道的计数器值作比较。互补模式下，奇数通道的计数器值无效，相应通道的比较器值一直与互补偶数通道的计数器值作比较。例如，通道0和通道1为互补通道，在互补模式，通道1的比较器持续与通道0计数器相比较，而不与通道1比较。当计数器值等于比较寄存器值，EPWM产生一个事件，并用事件产生一个EPWM脉冲，中断，或者触发EADC/DAC。在上下计数方式中，一个EPWM 周期将产生两个事件，如下图所示。CMPU是上数比较点事件，CMPD是下数比较点事件。

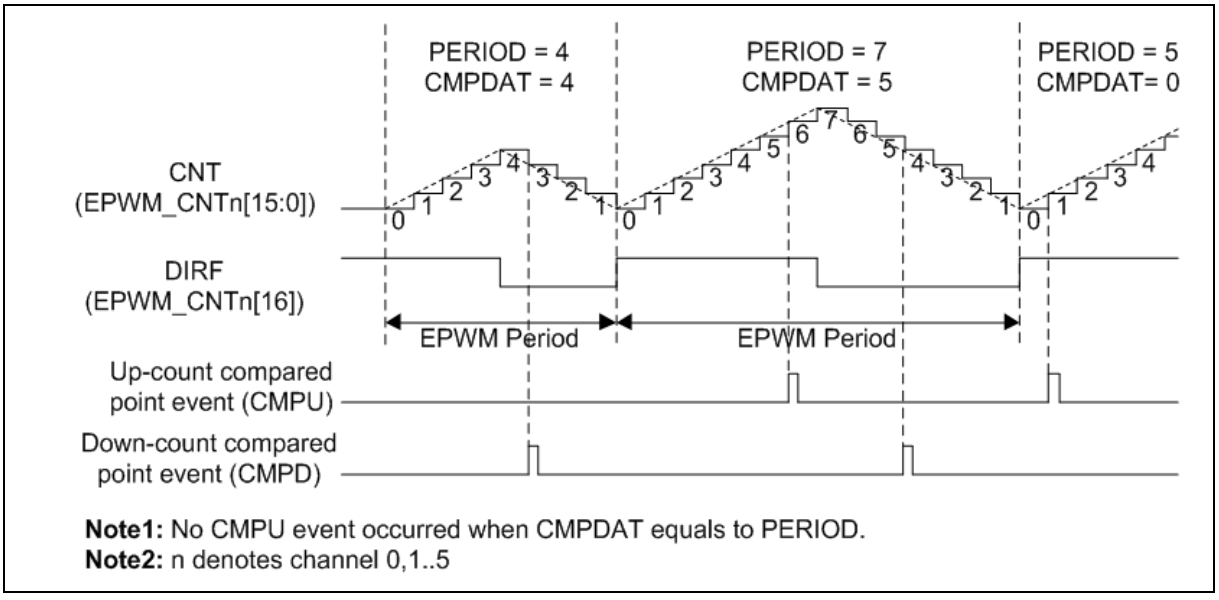


图 6.12-11EPWM 在上下计数方式中的比较点事件

FTCMPDAT 是一个自由触发比较器寄存器。每组互补通道仅支持一个 FTCMPDAT 寄存器。FTCMPDATn_m(n = 0,2,4, m = 1,3,5)的值持续与互补通道的偶通道相比较。当CNT等于FTCMPDAT, 下计数时 FTCMDn (EPWM_FTCI[10:8], n=0,2,4) 被置 1 和上计数时 FTCMUn (EPWM_FTCI[2:0], n=0,2,4)被置1。另外 EPWM产生一个事件，该事件仅可以触发EADC。

6.12.5.7 EPWM双缓存

双缓存是用两个缓存器来把软件写和硬件操作时序分开。载入值到缓存有四种装载模式：周期装载模式，立即装载模式，窗口装载模式和中心装载模式。软件设置好寄存器后，硬件将按照装载模式时序将寄存器值装载到缓存寄存器中。硬件的操作是基于缓存寄存器的值。这样可以避免软硬件不同步时的操作问题。

EPWM 提供 PBUF (EPWM_PBUFn[15:0]) 作为使 PERIOD 生效的缓存寄存器， CMPBUF (EPWM_CMPBUFn[15:0]) 作为使 CMPDAT 生效的缓存寄存器， FTCMPBUF (EPWM_FTCMPBUFn_m[15:0]) 作为使 FTCMPDAT 生效的缓存寄存器， CPSCBUF (EPWM_CPSCBUFn_m[15:0]) 作为是 CLKPSC 生效的缓存寄存器。双缓存的概念是用在装载模式，如下面描述。例如图6.12-12所示，在周期装载模式，通过软件写PERIOD 、 CMPDAT和FTCMPDAT，在下个周期时 EPWM 将载入新值到他们的缓存 PBUF(EPWM_PBUFn[15:0]), CMPBUF (EPWM_CMPBUFn[15:0]) 和 FTCMPBUF (EPWM_FTCBUF[15:0])而不影响当前计数操作。FTCMPU代表向上计数自由触发比较点事件，FTCMPD代表向下计数自由触发比较事件。

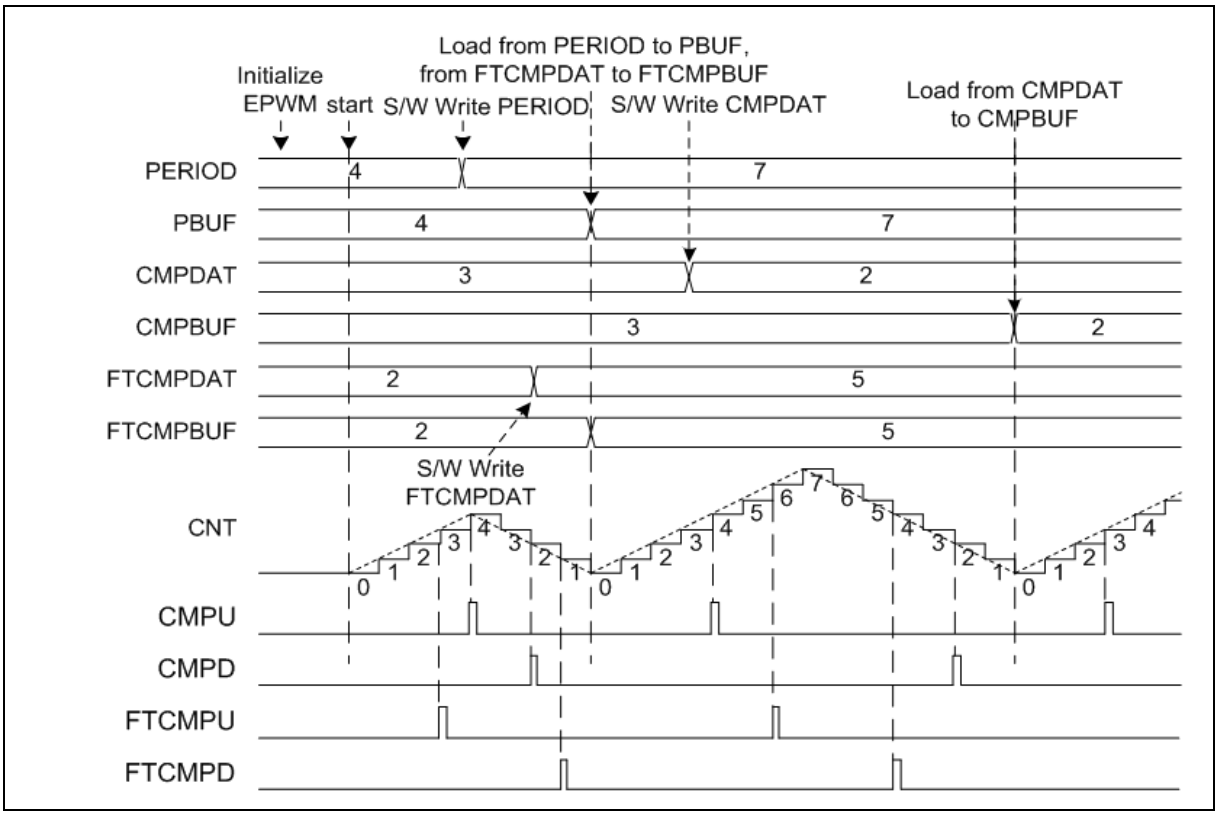


图 6.12-12EPWM 双缓存说明

6.12.5.8 周期装载模式

当立即装载模式，窗口装载模式和中心装载模式禁止，EPWM_CTL0的IMMLDENn位, WINLDENn 位和CTRLDn 位为0 。EPWM工作在周期装载模式。在周期装载模式，当每一个周期完成，CLKPSC(EPWM_CLKPSCn_m[11:0]),PERIOD(EPWM_PERIODn[15:0]) 和CMP(EPWM_CMPDATn[15:0]) 会全部加载到他们的激活寄存器CPSCBUF, PBUF 和 CMPBUF 。例如：在向上计数操作EPWM计数器从零计数到PERIOD后或在向下计数操作从PERIOD计数到零，或在上下计数操作向上计数从零到PERIOD然后向下计数到零。

图6.12-13表示向上计数操作时的周期装载时序，PERIOD DATA0表示PERIOD初始数据，PERIOD DATA1表示通过软件等所更新的第一个PERIOD 数据。CMPDAT同样遵照这规则。以下是图6.12-13.的步骤描述。用户可以通过观察EPWM周期和CMPU事件知道PERIOD 和 CMPDAT的更新条件。

1. point1处，软件写CMPDAT DATA1到CMPDAT
2. point 2处，EPWM周期结束，硬件装载CMPDAT DATA1 到 CMPBUF
3. point 3处，软件写PERIOD DATA1到 PERIOD
4. point 4处，EPWM周期结束，硬件装载CMPDAT DATA1 到 PBUF
5. point 5处，软件写PERIOD DATA2 到 PERIOD
6. point 6处，EPWM周期结束，硬件装载PERIOD DATA2到 PBUF

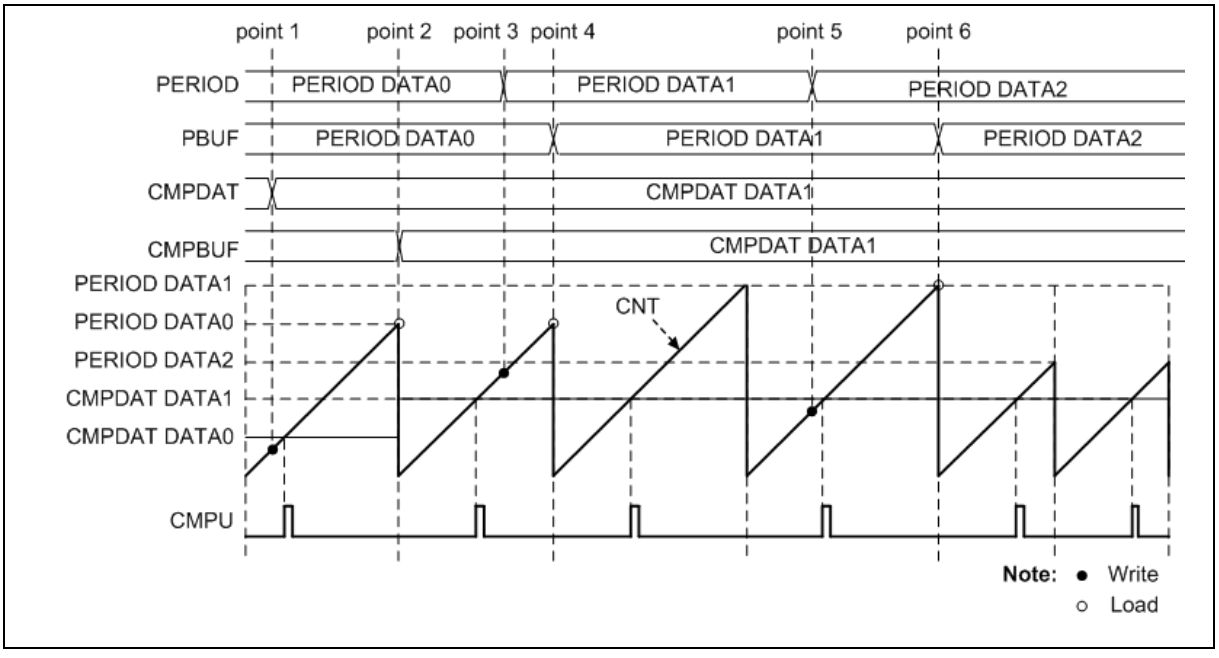


图 6.12-13 向上计数方式的周期装载模式

6.12.5.9 立即装载模式

如果IMMLDENn (EPWM_CTL0[21:16])位被置1时，EPWM工作在立即装载模式。在立即装载模式下，当软件更新 CLKPSC(EPWM_CLKPSCn_m[11:0]), PERIOD(EPWM_PERIODn[15:0]) 或CMP(EPWM_CMPDATn[15:0]) 后，软件立即将CLKPSC、PERIOD 和 CMPDAT 值装载到缓存CPSCBUF (EPWM_CPSCBUFn_m[15:0]), PBUF (EPWM_PBUFn[15:0]) 和 CMPBUF (EPWM_CMPBUFn[15:0])。如果PERIOD更新值小于当前计数值，计数器会计数到0xFFFF，当计数器

计数到0xFFFF且预分频计数到0，标志 CNTMAXF(EPWMx_STATUS[5:0])会置位，然后计数器循环计数。立即装载模式有最高优先级，如果IMMLDENn被设置，通道n的其他装载模式将失效。图6.12-14所示例子的步骤顺序如下描述

1. 软件写CMPDAT DATA1，硬件在point1处将立即CMPDAT DATA1装载到CMPBUF；
2. point 2处，软件写入PERIOD DATA1值大于当前值，计数器将继续计数到PERIOD DATA1来完成装载；
3. point 3处，软件写入PERIOD DATA2值小于当前值，计数器将继续计数到最大值0XFFFF并循环从0开始到PERIOD DATA2来完成这个周期装载。

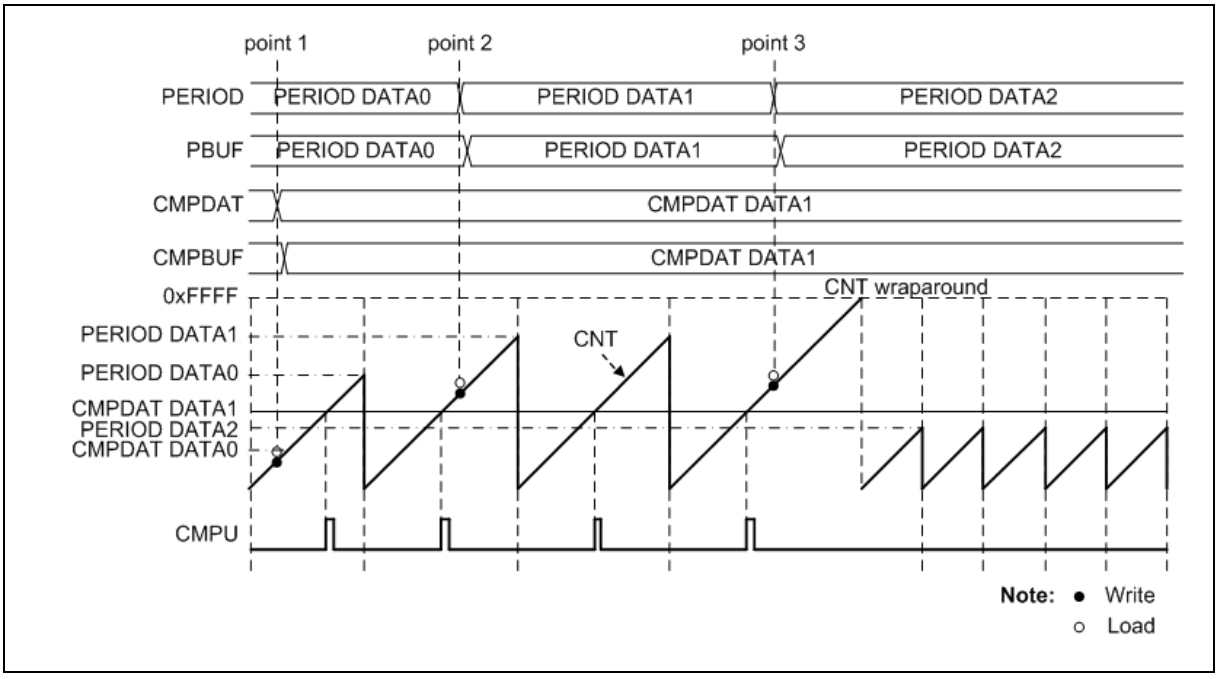


图 6.12-14 向上计数方式的立即装载模式

6.12.5.10 窗口装载模式

如果WINLDENn (EPWM_CTL0[13:8])位被置为1，EPWM工作在窗口装载模式。在窗口装载模式CLKPSC(EPWM_CLKPSCn_m[11:0])、PERIOD(EPWM_PERIODn[15:0])、CMP(EPWM_CMPDATn[15:0])和FTCMPDAT (EPWM_FTCMPDATn_m[15:0])会在每个周期完成装载到他们的激活寄存器 CPSCBUF, PBUF、CMPBUF和FTCMPBUF，但是所有装载仅当窗口打开时有效。每个通道n的装载窗口打开方式都是通过设置相应的LOADn (EPWM_LOAD[5:0])位为1，然后硬件会在EPWM周期终止时关闭该窗口。图6.12-15所示例子的步骤顺序如下描述：

1. point 1处软件写CMPDAT DATA1，在这个周期中装载窗口不会被打开，所以CMPDAT的值不会被装载到CMPBUF
2. point2处软件写装载来打开装载窗口
3. point 3处软件写CLKPSC DATA和PERIOD DATA1
4. point 4处，装载窗口被打开，硬件将装载CLKPSC DATA1 及PERIOD DATA1 和 CMPDAT DATA1 到它们的缓冲区然后再EPWM周期终止时关闭装载窗口。

5. point 5处软件写CLKPSC DATA2和PERIOD DATA2。
6. point 6处硬件在EPWM周期终止时装载CLKPSC DATA2 及PERIOD DATA2到PBUF
7. point 7处软件写CLKPSC DATA3 和PERIOD DATA3
8. point8处软件写装载来打开装载窗口
9. 硬件装载CLKPSC DATA3 及PERIOD DATA3 到PBUF ，然后在point 9处关闭装载窗口。

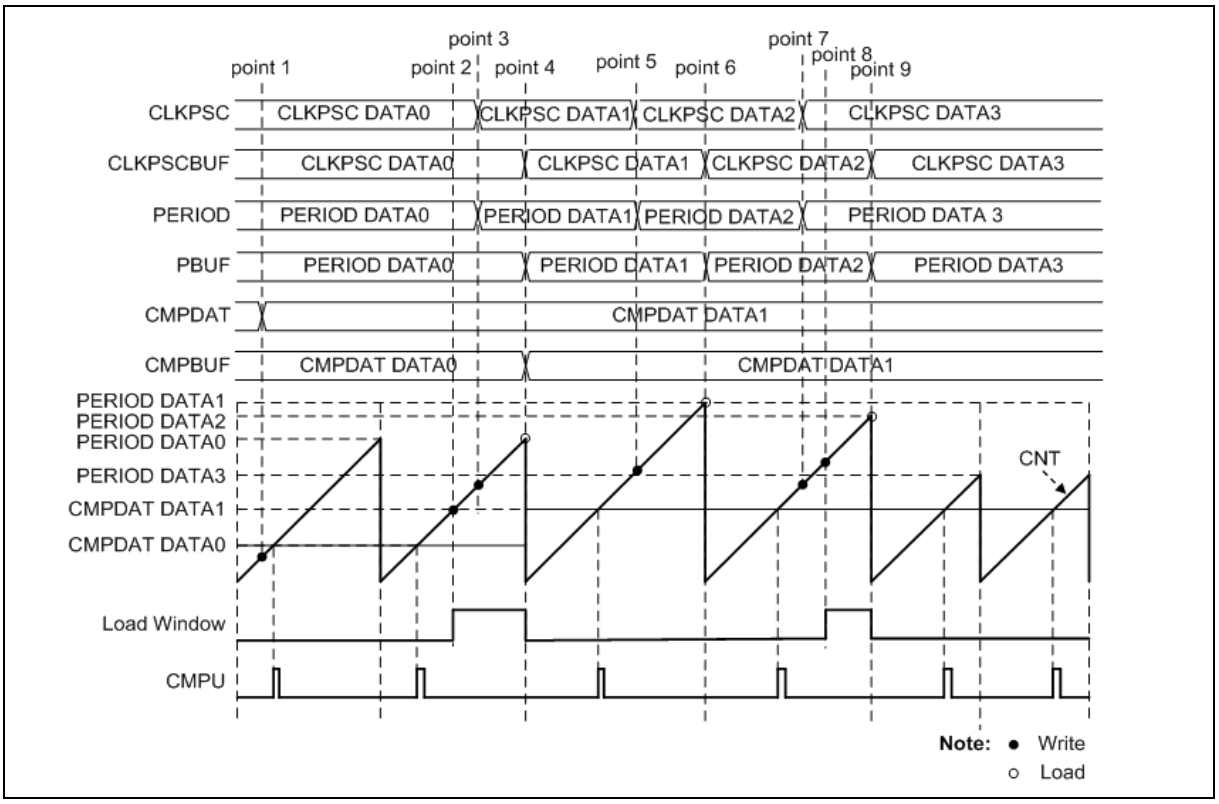


图 6.12-15 向上计数模式窗口装载

6.12.5.11 中心装载模式

当CTRLDn (EPWM_CTL0[5:0])的EPWM通道n相应位被置1，且EPWM计数器设置为上下计数模式，CNTTYPEn (EPWM_CTL1[2n+1:2n], n = 0,1..5) 是 0x2，EPWM工作在中心装载模式。在中心装载模式，在每个周期的中点CMP(EPWM_CMPDATn[15:0])和FTCMPDAT (EPWM_FTCMPDATn_m[15:0])值将装载到CMPBUF和FTCMPBUF寄存器，也就是说当当前周期完成，计数器计数到PERIOD点，CLKPSC(EPWM_CLKPSCn_m[11:0]) 和 PERIOD(EPWM_PERIODn[15:0]) 会装载到激活寄存器CPSCBUF 和 PBUF。周期装载模式可以配合窗口装载模式一起工作，CMP(EPWM_CMPDATn[15:0])会在每一个周期的中心装载到激活寄存器CMPBUF，但是它仅在装载窗口期间有效。图6.12-16所示例子的步骤顺序如下描述：

1. point 1处，软件写CMPDAT DATA1；
2. point 2处，EPWM周期中心点硬件装载CMPDAT DATA1到CMPBUF；
3. point 3处，软件写PERIOD DATA1；

4. point 4处，EPWM周期终点硬件装载PERIOD DATA1到PBUF；
5. point 5处，软件写CMPDAT DATA2；
6. point 6处，在EPWM周期中心点硬件装载CMPDAT DATA2到CMPBUF；
7. point 7处，软件写PERIOD DATA2；
8. point 8处，在EPWM周期终点硬件装载PERIOD DATA2到PBUF。

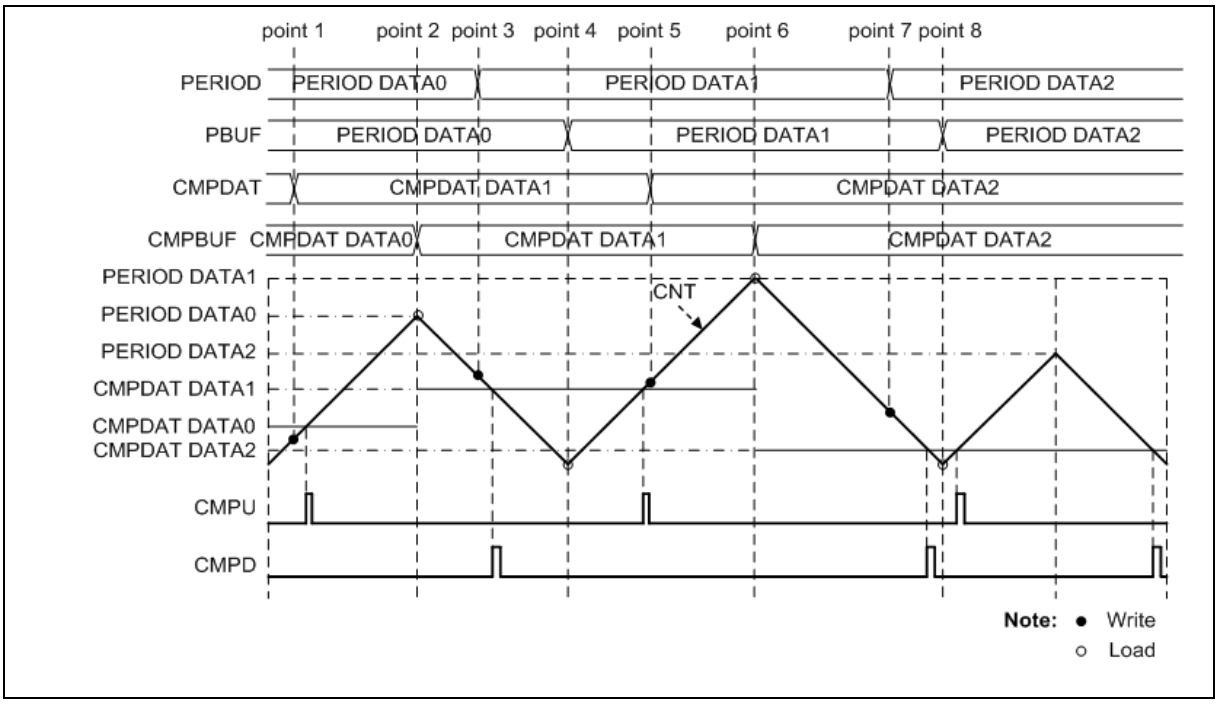


图 6.12-16 上下计数方式的中心装载模式

6.12.5.12 EPWM计数器工作模式

EPWM 计数器支持两种工作模式：单次（One-shot）模式和自动装载模式。如果CNTMODEn (EPWM_CTL1[21:16])位被置1，EPWM计数器将工作在One-shot 模式，如果被设为0则工作在自动装载模式。

One-shot 模式， CMPDAT和PERIOD应该先被写，然后设置CNTENn (EPWM_CNTEN[5:0])通道 n相应位置1来使能EPWM预分频和计数开始运行。EPWM 计数器计完一个周期后，计数器的值将保持为0。

在下一个单次周期，软件需要重新写一个新的CMPDAT比较值来重新启动。如果one-shot计数器保持计数，写CMPDAT将使得下一个单次接着下一个单次。此外，如果继续one-shot操作，对CMPDAT写两次，单次周期值将使用最后一次写入值作为CMPDAT比较值，也仅产生一个one-shot 脉冲周期。图6.12-17是一个例子，操作步骤如下：

1. point 1处软件写PERIOD DATA1，硬件会立即装载PERIOD DATA1到 PBUF；
2. point 2处软件写CMPDAT DATA1（等于CMPDAT DATA0），硬件会立即装载CMPDAT DATA1 到 CMPBUF，该事件也会触发one-shot；
3. point 3处软件写CMPDAT DATA2 会重新触发下一个one-shot (持续one-shot)；

4. point 4处软件写CMPDAT DATA3 来覆盖CMPDAT DATA2 ， 然后再触发下一个one-shot周期 ；
5. point 5处周期装载CMPDAT DATA3 到CMPBUF；
6. point6 处，如果在上一个周期中，没有新的CMPDAT比较值被写入，那么计数器的值将保持为0 。

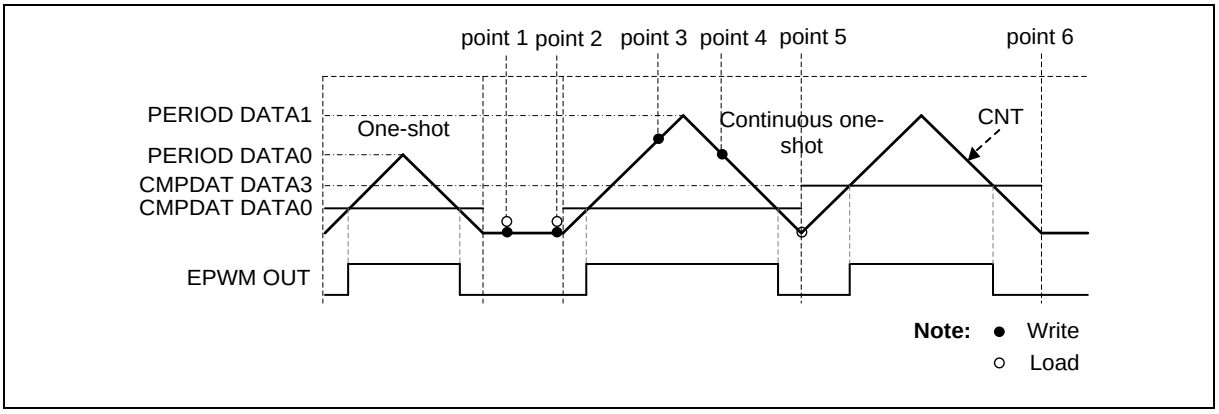


图 6.12-17EPWMOne-shot 模式输出波形

在自动装载模式，应该先写CMPDAT 和 PERIOD ，然后 CNTENn通道n位被置1来使能EPWM预分频器并开始计数。根据不同的装载模式， CLKPSC(EPWM_CLKPSCn_m[11:0]), PERIOD(EPWM_PERIODn[15:0]) 和 CMP(EPWM_CMPDATn[15:0])的值将被自动装载到它们的缓冲区。如果周期PERIOD被设为0， EPWM 计数器将被设为0。

6.12.5.13 EPWM脉冲发生器

EPWM脉冲发生器是用计数器和比较器事件来产生EPWM脉冲。这些事件包括：向上和向下计数方式的零点或周期点，上下计数方式的中心点，以及三种方式中的计数器等于比较值。作为上下计数方式，计数器有两个等于比较器的点，一个在向上计数，一个在向下计数。此外，互补模式有两个比较器值与计数器比较，因此在上下计数方式中有四个相等值，向上或向下计数方式各有两个。

通过设定EPWM_WGCTL0 和 EPWM_WGCTL1，每个事件点可以决定EPWM波形：不变（X），为低（L），为高（H）或切换（T）。用这些点可以很容易产生一个不规则的EPWM脉冲或可变波形，如图6.12-18。图中，EPWM是互补模式，两个比较器n和m来产生EPWM脉冲。n表示偶数通道0、2、4；m表示奇数通道1、3、5，n和m通道是互补组。互补模式用两个通道（CH0和CH1、CH2和CH3、CH4和CH5）作为一组EPWM输出一个互补波形。CMPU表示当向上计数时CNT等于CMPDAT，CMPD表示当向下计数时CNT等于CMPDAT。

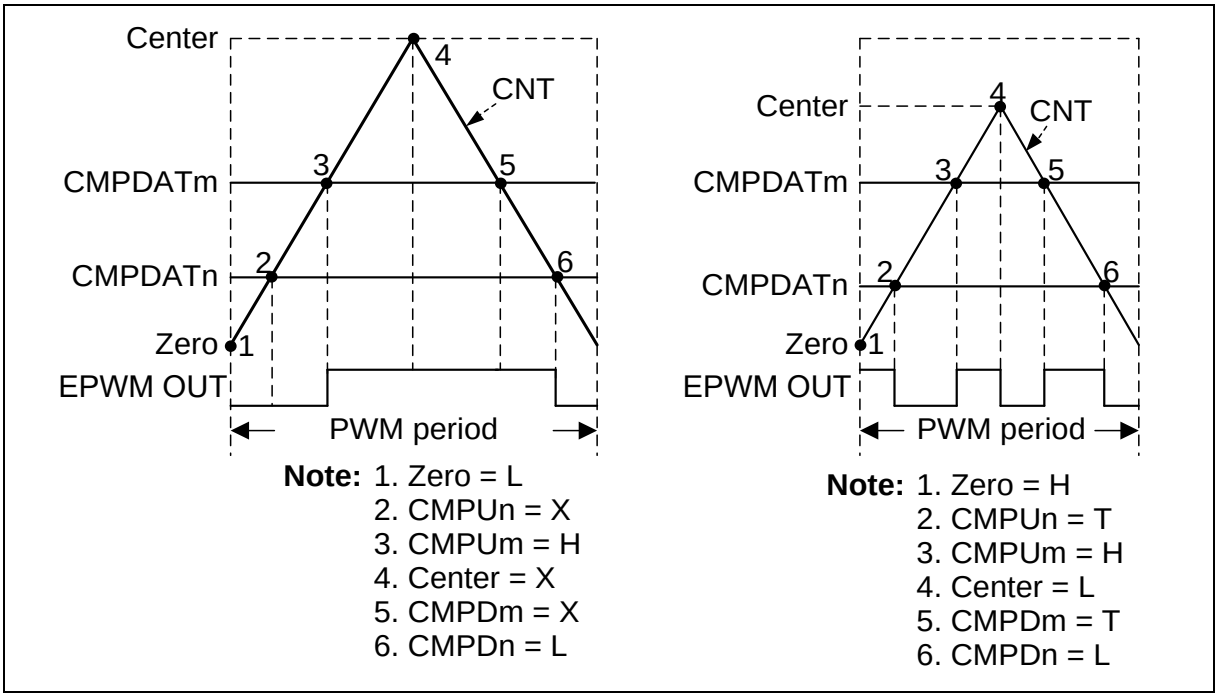


图 6.12-18EPWM 脉冲产生

有时事件会被同时产生，因此，不同计数方式的事件优先级列表如下：向上计数方式（表6.12-2）、向下计数方式（表6.12-3）、上下计数方式（表6.12-4）。通过事件优先级，使用者可以很容易产生0%到 100%占空比的脉冲，图6.12-19所示。

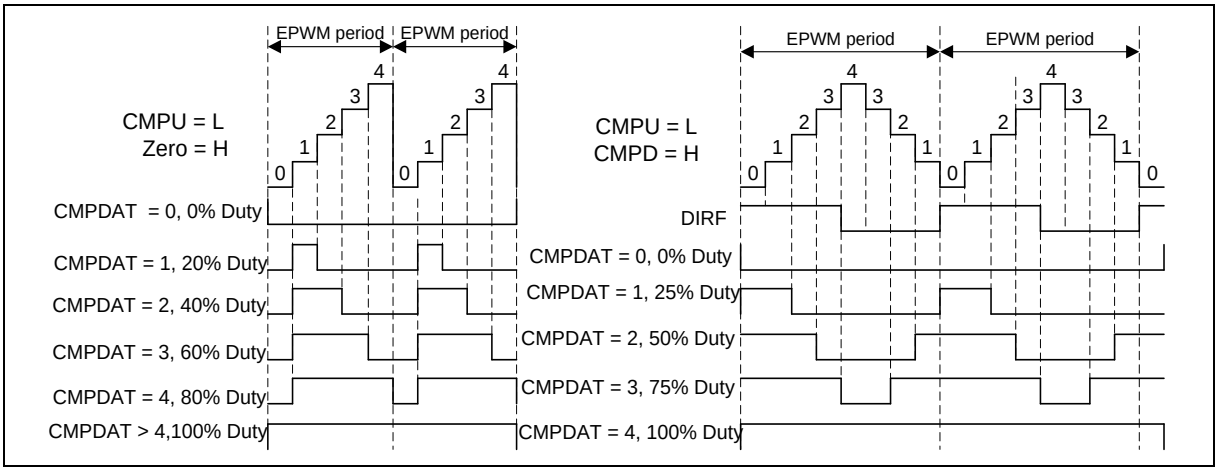


图 6.12-19EPWM0%到 100%占空比脉冲产生

优先级	向上事件
1 (最高)	周期点事件 (CNT = PERIOD)
2	奇数通道的向上比较事件 (CNT = CMPUm)
3	偶数通道的向上比较事件(CNT = CMPUn)

4 (最低)	零点事件 (CNT = 0)

表 6.12-2 向上计数器 EPWM 脉冲发生事件优先级

优先级	向下事件
1 (最高)	零点事件 (CNT = 0)
2	奇数通道的向下比较事件(CNT = CMPDm)
3	偶数通道的向下比较事件(CNT = CMPDn)
4 (最低)	周期点事件 (CNT = PERIOD)

表 6.12-3 向下计数器 EPWM 脉冲发生优先级

优先级	向上事件	向下事件
1 (最高)	奇数通道的向上比较事件(CNT = CMPUm	奇数通道的向下比较事件(CNT = CMPDm)
2	偶数通道的向上比较事件(CNT = CMPUn)	偶数通道的向下比较事件(CNT = CMPDn)
3	零点事件(CNT = 0)	周期点 (中点) 事件 (CNT = PERIOD)

表 6.12-4 上下计数器 EPWM 脉冲产生事件优先级

6.12.5.14 EPWM输出模式

EPWM支持两个输出模式：独立模式，可以应用于直流电机系统；带死区插入互补模式，可以由于交流感应电机和永磁同步电机。

6.12.5.15 独立模式

默认EPWM在独立模式，当通道n相应位OUTMODEn(EPWM_CTL1[26:24])置0 独立模式被使能。独立模式中6个EPWM通道：EPWM_CH0、EPWM_CH1、EPWM_CH2、EPWM_CH3、EPWM_CH4和EPWM_CH5都运行在各自的周期占空比，如下图所示：

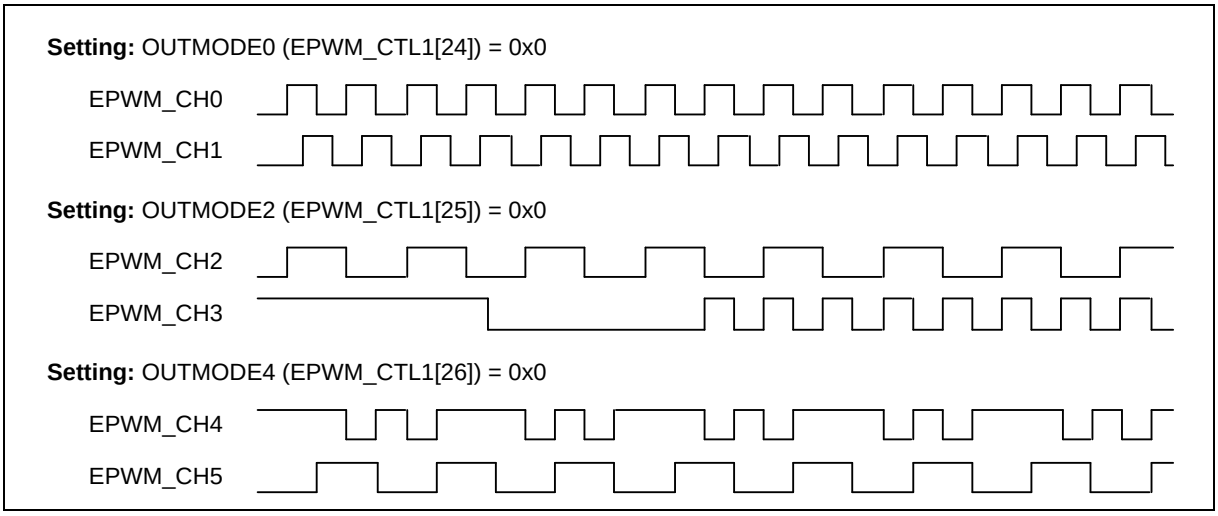


图 6.12-20EPWM 独立模式波形

6.12.5.16 互补模式

当互补通道OUTMODEn (EPWM_CTL1[26:24])相应位被置1，互补模式被使能。互补模式有3个EPWM发生器，模块中总共3组EPWM输出管脚。互补模式中内部奇数EPWM信号必须与相应偶数通道的EPWM信号互补。EPWM_CH1将互补于EPWM_CH0，EPWM_CH3将互补于EPWM_CH2，EPWM_CH5将互补于EPWM_CH4，如下图所示：

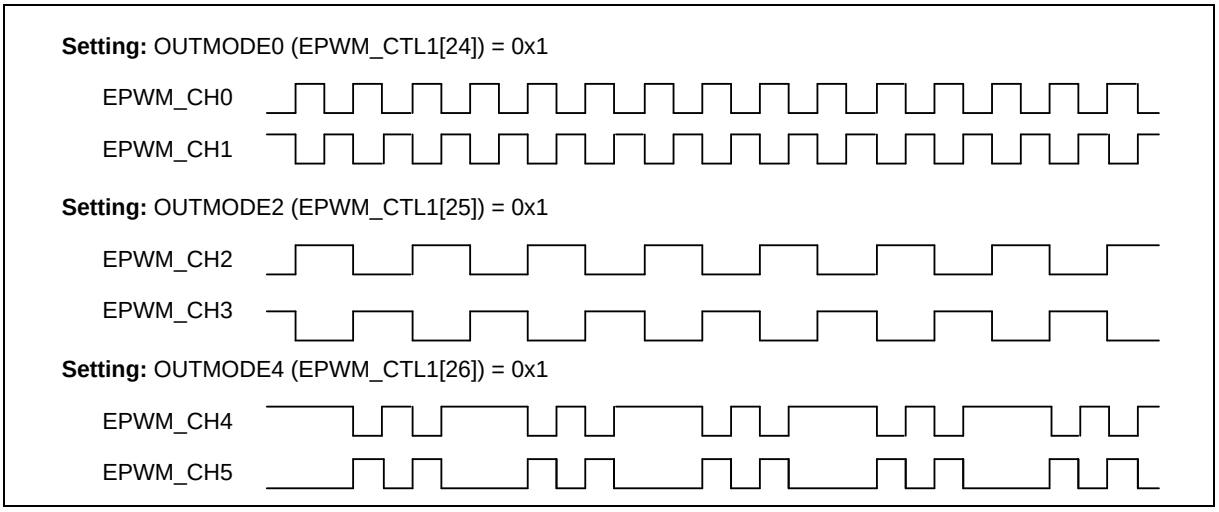


图 6.12-21EPWM 互补模式波形

6.12.5.17 EPWM输出功能

根据输出模式，有两种输出功能：组功能和高级输出控制同步功能。组功能，会强制EPWM_CH2、EPWM_CH4 与 EPWM_CH0 发生器同步， EPWM_CH3 、 EPWM_CH5 与EPWM_CH1同步，主要用于在DC和BLDC马达应用的更新占空比控制。此外，同步功能使得EPWM0和EPWM1任何一个通道在相位控制上，用户可以控制相值和方向。

6.12.5.18 组功能

不管是在独立模式还是互补模式，当GROUPEN (EPWM_CTL0[24])位被置1，组功能会被使能。该控制允许所有EPWM偶数通道通过EPWM_PERIOD0和EPWM_CMPDAT0寄存器输出可控，所有奇数EPWM通道通过EPWM_PERIOD1和EPWM_CMPDAT1 寄存器输出可控。也就是，用户仅只需设置EPWM_CH0 使得 EPWM_CH0, EPWM_CH2 和 EPWM_CH4输出相同的脉冲，设置EPWM_CH1 使得EPWM_CH1, EPWM_CH3 和 EPWM_CH5 输出相同的脉冲。如下图所示。当运行在组功能，独立模式下OUTMODE0, OUTMODE2 和OUTMODE4必须被设置为0，或在互补模式下置为1。

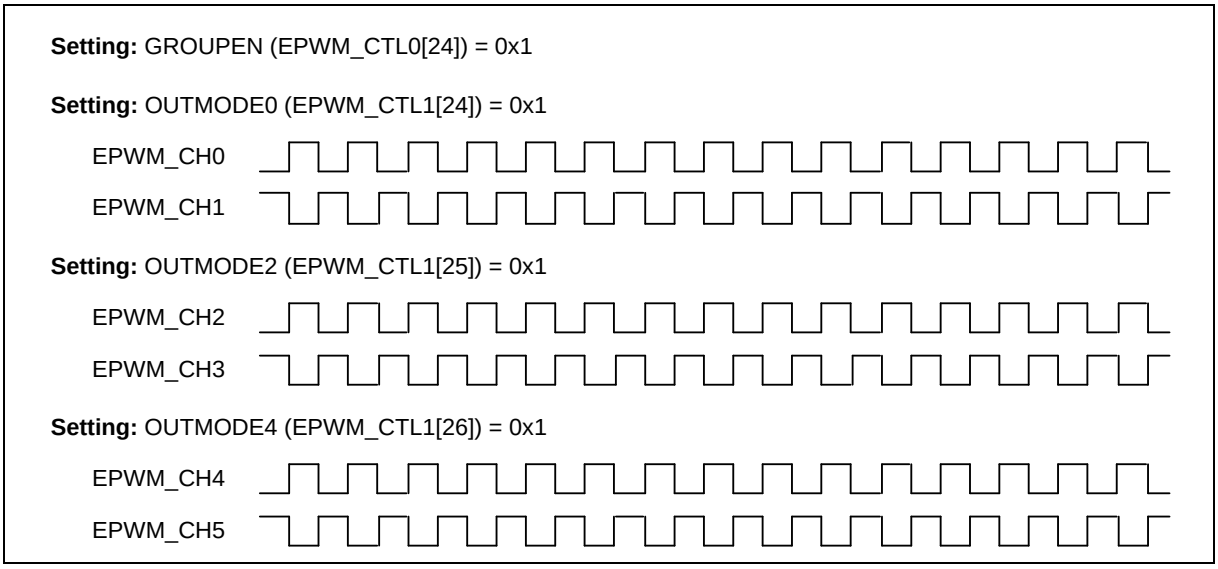


图 6.12-22EPWM 组功能波形

6.12.5.19 同步功能

同步功能仅当在互补模式下才可被使能。图6.12-24是计数器同步功能框图。每个EPWM组计数器有一个SYNC_IN 和一个SYNC_OUT信号。第一个EPWM0组计数器SYNC_IN来自于EPWM0_SYNC_IN管脚，其它的来自于上一个EPWM组计数器的SYNC_OUT信号。来自EPWM0_SYNC_IN管脚的输入信号会被3位的滤波器滤波，如图6.12-23。此外通过设置SINPINV (EPWM_SYNC[23])可以实现对输入信号的极性控制。噪音滤波采样时钟可以通过设置SFLTCNT (EPWM_SYNC[22:20])来选择以适应不同的噪音特性。用户可以定义多少个采样时钟周期来识别一个有效的SYNC_IN信号边沿。配置SNFLTEN (EPWM_SYNC[16])会使能噪音滤波功能，默认是禁止的。

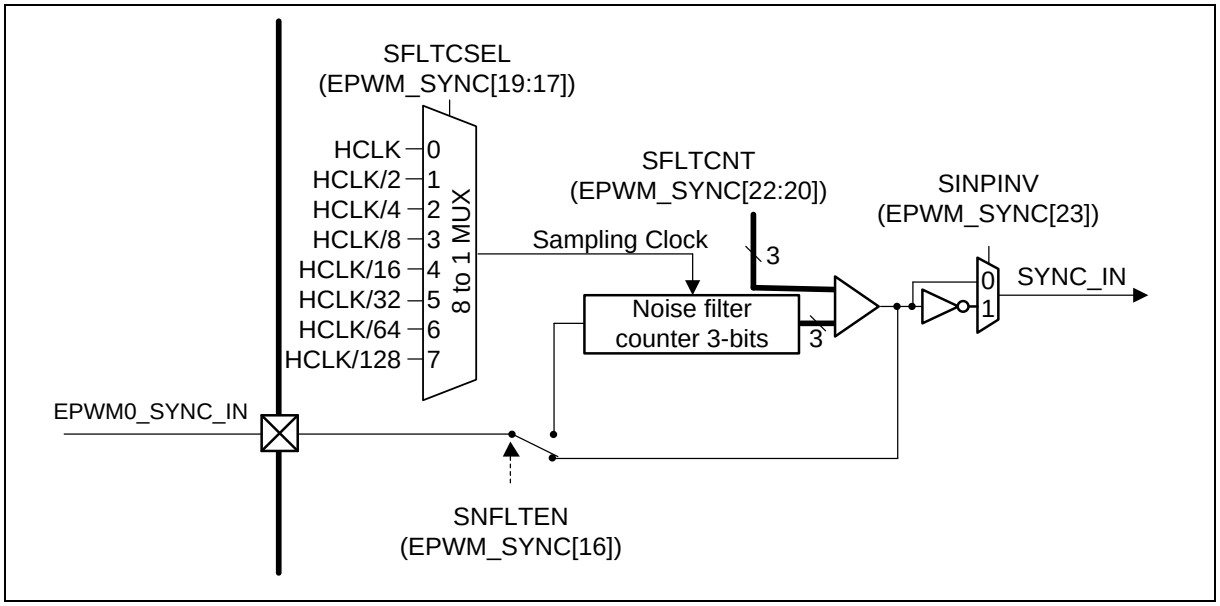


图 6.12-23EPWMSYNC_IN 噪音滤波框图

用户可以使用SINSRCn (EPWM_SYNC[13:8]) 来选择同步源。当SINSRCn位置0，当EPWM0_SYNC_IN管脚是高或设置SWSYNCn (EPWM_SWSYNC[2:0])为1，可以产生SYNC_IN信号

给下一个计数器同步。同步源也可以被选择为CNT = 0 或 CNT = EPWM_CMPDATm (如果是上下计数器类型，在一个EPWM周期中将同步两次)来触发一个同步事件或禁止SYNC_OUT信号。

当PHSENn (EPWM_SYNC[2:0]) 被使能，且同步源有事件发生，计数器将从PHS (EPWM_PHS[15:0]) 寄存器装载数据。该方法将在同时同步计数器到不同相位。在上-下计数模式，同步后用户可以设置PHSDIRn (EPWM_SYNC[26:24])来控制计数方向。尽管同步功能可以在相位中同步通道，但是EPWM刚使能时不能使用。要在同一时刻启动这些计数器，用户则必须设置EPWM同步启动控制寄存器(EPWM_SSCTL[5:0])来使能这些一起同步的通道计数器，然后设置EPWM 同步启动触发寄存器CNTSEN (EPWM_SSTRG[0])。

在应用当中，请不要同时使用组和同步功能，因为此时同步功能将无效。

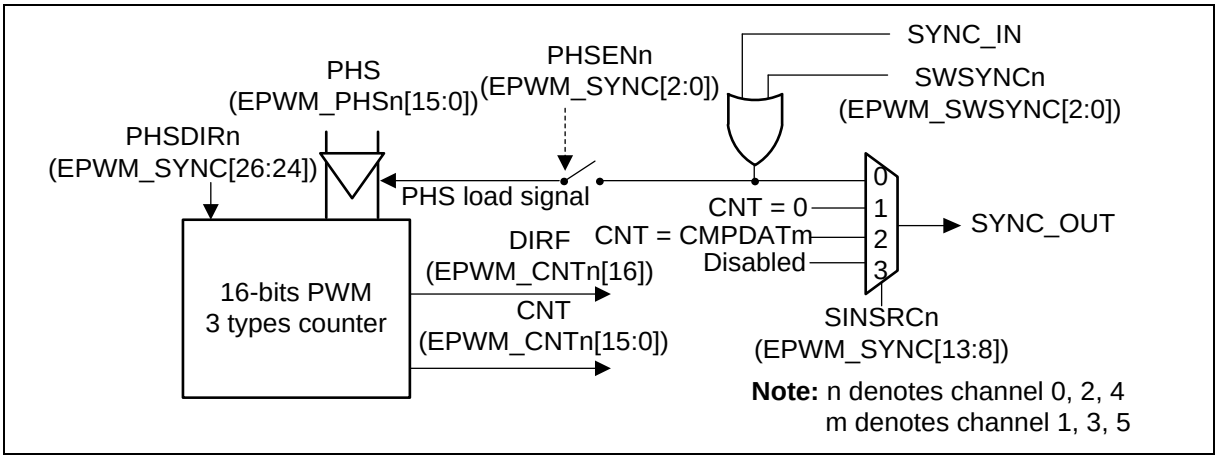


图 6.12-24 EPWM 计数器同步功能框图

图6.12-25是上-下计数模式中同步功能的例子。例中，同步源来自外部EPWM SYNC_IN。刚开始，EPWM_CH0，EPWM_CH2 和 EPWM_CH4处于相同的相位。然后在A点， EPWM SYNC 输入信号成为一个同步事件，导致所有计数器相位切换并且计数方向改变。如果在同步事件到来前修改计数器，用户需要在PHS (EPWM_PHSn[15:0]) 中设置相应的相位值以及在PHSDIRn (EPWM_SYNC[26:24])中设置计数方向。在这种情况下，可以在同步以后通过设置相应通道n计数器的计数方向，实现移动1/3的相位，如图6.12-25左边所示。

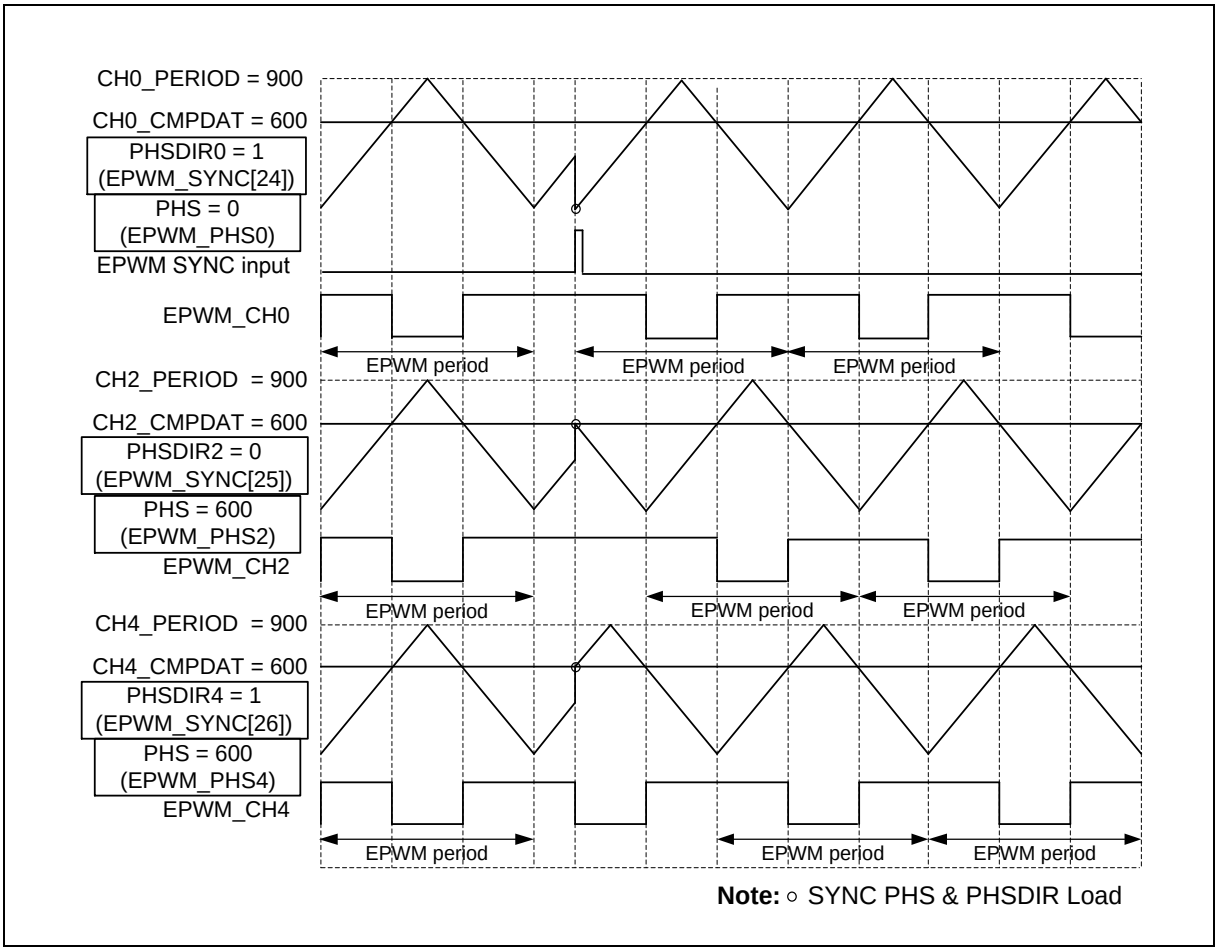


图 6.12-25EPWM 同步源来自 SYNC_IN 信号的同步功能

6.12.5.20 EPWM输出控制

EPWM脉冲产生后，有4到6步来控制EPWM通道输出。独立模式中，屏蔽，刹车，管脚极性和输出使能4步如图6.12-26所示。互补模式中，在以上4步前增加2步：互补通道和死区插入，如图6.12-27所示。

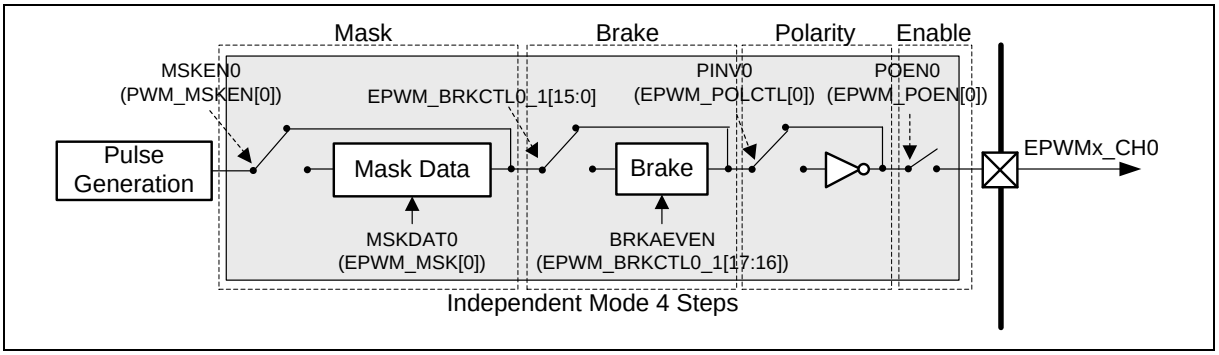


图 6.12-26EPWMx_CH0 独立模式下的输出控制

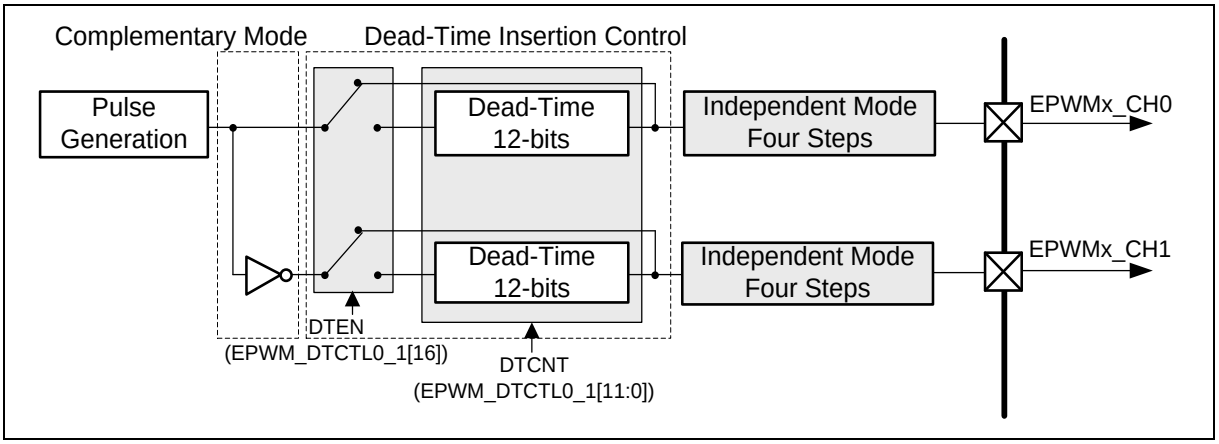


图 6.12-27EPWMx_CH0 和 EPWMx_CH1 在互补模式下的输出控制

6.12.5.21 死区插入

互补应用中，互补通道也许用来驱动外部设备例如开关电源，在互补输出通道间死区发生器插入一个低电平时段叫“死区时间”来安全驱动设备，防止系统或设备烧坏。因此死区控制是正确操作互补系统的重要机制。通过设置通道n DTEN (EPWM_DTCTLn[16])的相应位来使能死区功能，设置DTCNT (EPWM_DTCTLn[11:0])控制死区时间周期，死区时间可以通过以下公式计算：

死区时间 = (DTCNT (EPWM_DTCTLn[11:0])+1) * EPWMx_CLK 周期

死区插入时钟源可以通过设置DTCKSEL (EPWM_DTCTLn_m[24])为1来选择来自预分频器输出。默认时钟源来自EPWM_CLK，就是预分频器的输入，死区时间可以通过下面公式计算：

死区时间= (DTCNT (EPWM_DTCTLn[11:0])+1) *

(CLKPSC (EPWM_CLKPSCn [11:0])+1)*EPWMx_CLK周期

请注意EPWM_DTCTLn_m 是写保护寄存器。

图6.12-28表示一对EPWM信号的死区插入。

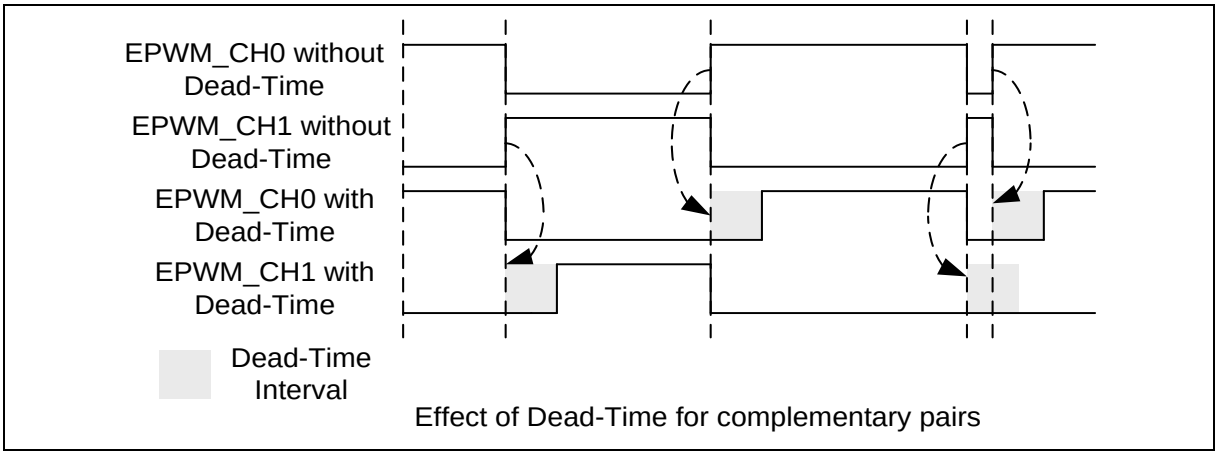


图 6.12-28 死区插入

6.12.5.22 EPWM屏蔽输出功能

通过设置EPWM屏蔽使能控制寄存器(EPWM_MSKEN)和EPWM屏蔽数据寄存器(EPWM_MSK)可以对每个EPWM的输出手动覆盖。通过这些设置，EPWM通道的输出可以被强制设定为特定的逻辑状态，而与比较单元无关。EPWM屏蔽位用于控制各类电子整流电机(ECM)，如BLDC电机，非常有用。EPWM_MSKEN寄存器有六个位，MSKENn (EPWM_MSKEN[5:0])。如果MSKENn置高，通道n输出将被覆盖。EPWM_MSK寄存器有六个位，MSKDATn (EPWM_MSK[5:0])。当通道被覆盖，MSKDATn位值决定EPWM通道n的输出状态值。图6.12-29显示如何将EPWM屏蔽控制器用于覆盖特性的例子。

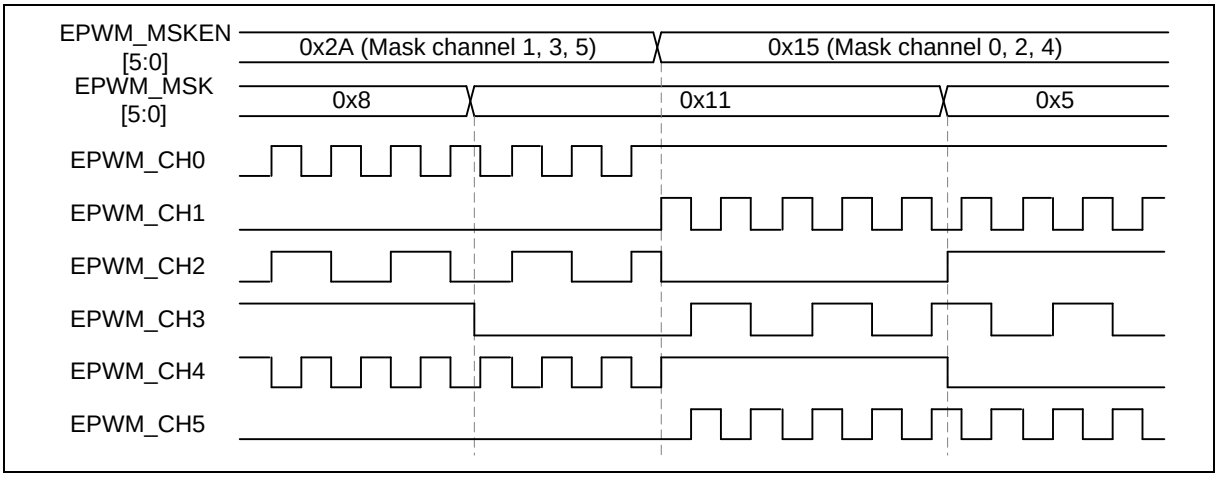


图 6.12-29 屏蔽控制波形图解

6.12.5.23 EPWM刹车

每个EPWM模块有两个外部输入刹车控制信号。用户可以通过EPWM_BNF寄存器的BKxSRC位(x=0,1, y=0,1)选择激活刹车的管脚源来自EPWMx_BRAKEy管脚。外部信号会被一个3位的滤波器滤波。用户可以通过EPWM_BNF寄存器的BRKxNFEN位使能噪音滤波器功能。噪音滤波器的采样时钟可以通过EPWM_BNF寄存器的BRKxNFSEL位选择以适应不同的噪音特性。通过设置BRKxFCNT，用户可以定义滤波器需要多少次采样时钟周期来确认有效刹车信号。

此外，外部信号可以通过设置BRKxPINV(x表示输入管脚0或1)被翻转，来实现刹车控制型号的极性设置。设置BRKxPINV位为0，当EPWMx_BRAKEy(x=0,1, y=0,1)管脚状态从低到高时，刹车事件会发生。设置BRKxPINV位为1，当EPWMx_BRAKEy管脚状态从高到低时，刹车事件会发生。

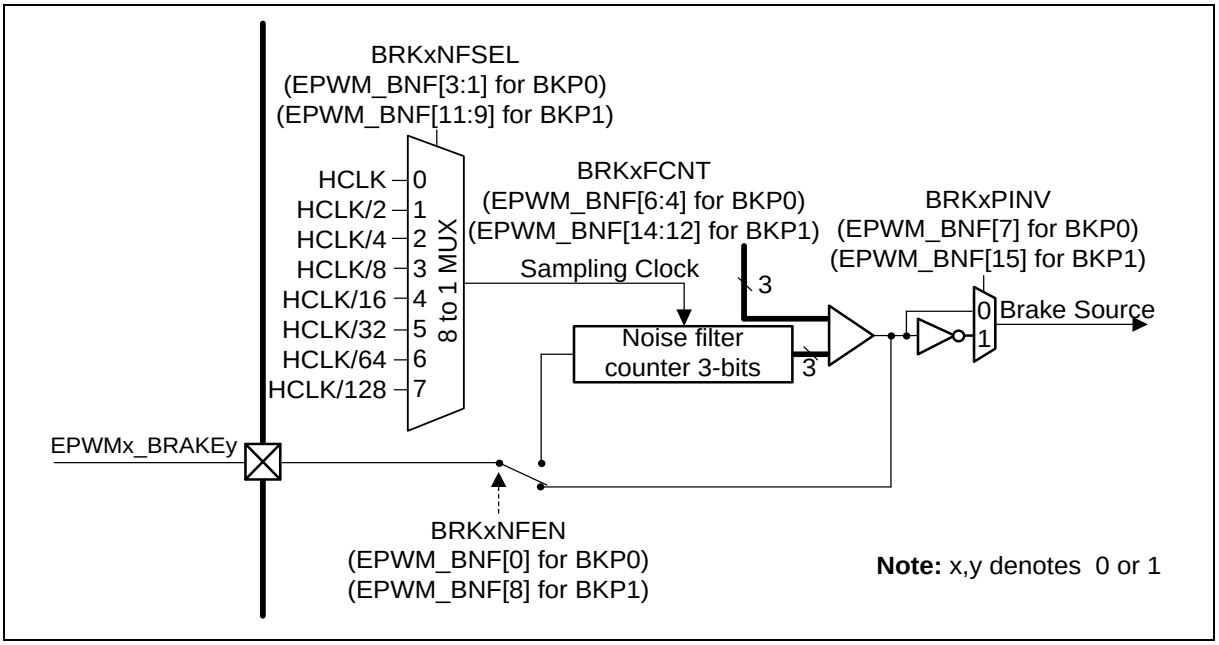


图 6.12-30 刹车噪音滤波器方块图

对于互补模式，一旦刹车事件发生，通常需要设置一个安全输出状态到互补输出组。

每组互补通道共享一个刹车功能，如图6.12-31。当刹车事件发生时，用户可以设置偶数通道的BRKAEVEN (EPWM_BRKCTL0_1[17:16])和奇数通道的BRKAODD (EPWM_BRKCTL0_1[19:18])，确保通道输出安全状态。有两个刹车检测：边沿检测和电平检测。当边沿检测发现刹车信号并且BRKEIENn_m (EPWM_INTEN1[2:0])被使能，刹车功能产生BRK_INT中断。这个中断需要用软件清除，且在中断清除后BRKESTSn (EPWM_INTSTS1[21:16])刹车状态将保持直到下一个EPWM周期开始。通过电平检测，刹车功能也可以用另一种方式操作。一旦电平检测发现刹车信号并且BRKLIENn_m (EPWM_INTEN1[10:8])被使能，刹车功能将产生BRK_INT中断，但是当刹车信号回到高电平，当前EPWM周期结束后，EPWM输出将变为正常，BRKLSTSn (EPWM_INTSTS1[29:24])位被自动清除。

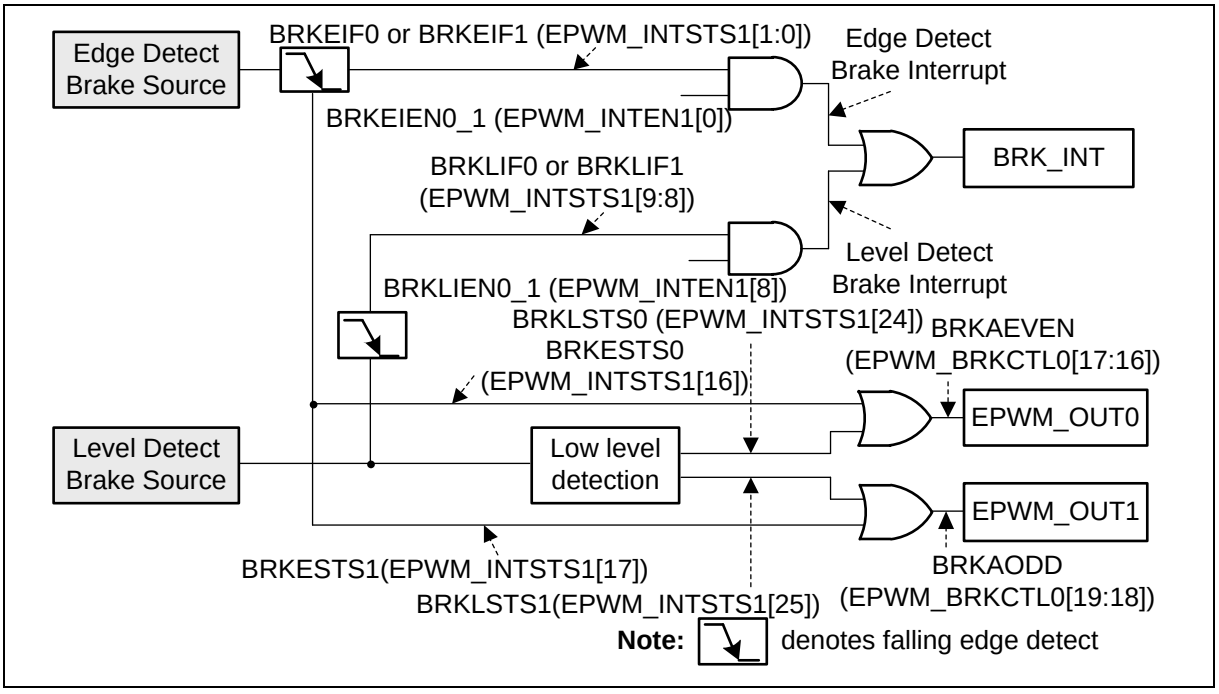


图 6.12-31EPWMx_CH0 和 EPWMx_CH1 通道组刹车方块图

图6.12-32表示EPWM_CH0和EPWM_CH1通道组的边沿检测波形。这种情况下，边沿检测到两次刹车事件。当刹车事件发生BRKEIF0和BRKEIF1都被置1并且BRKESTS0和BRKESTS1也被置1表明EPWM_CH0和EPWM_CH1处于刹车状态。当第一个事件发生，软件需写1清BRKEIF0。然后在下一个EPWM周期开始时硬件清除BRKESTS0。此时即使刹车事件依旧存在，EPWM_CH0仍然输出正常波形。第二个事件也触发相同标志，但此时软件需写1去清BRKEIF1，此后在下个EPWM周期开始EPWM_CH1正常输出。

相对于边沿检测例子，图6.12-33表示EPWM_CH0 和 EPWM_CH1通道组的电平检测波形。这种情况，BRKLIF0和BRKLIF1只可以表示刹车事件已发生。在下一个EPWM周期开始时不管BRKLIF0 和BRKLIF是什么状态，BRKLSTS0 和 BRKLSTS1的刹车状态都会被自动恢复。

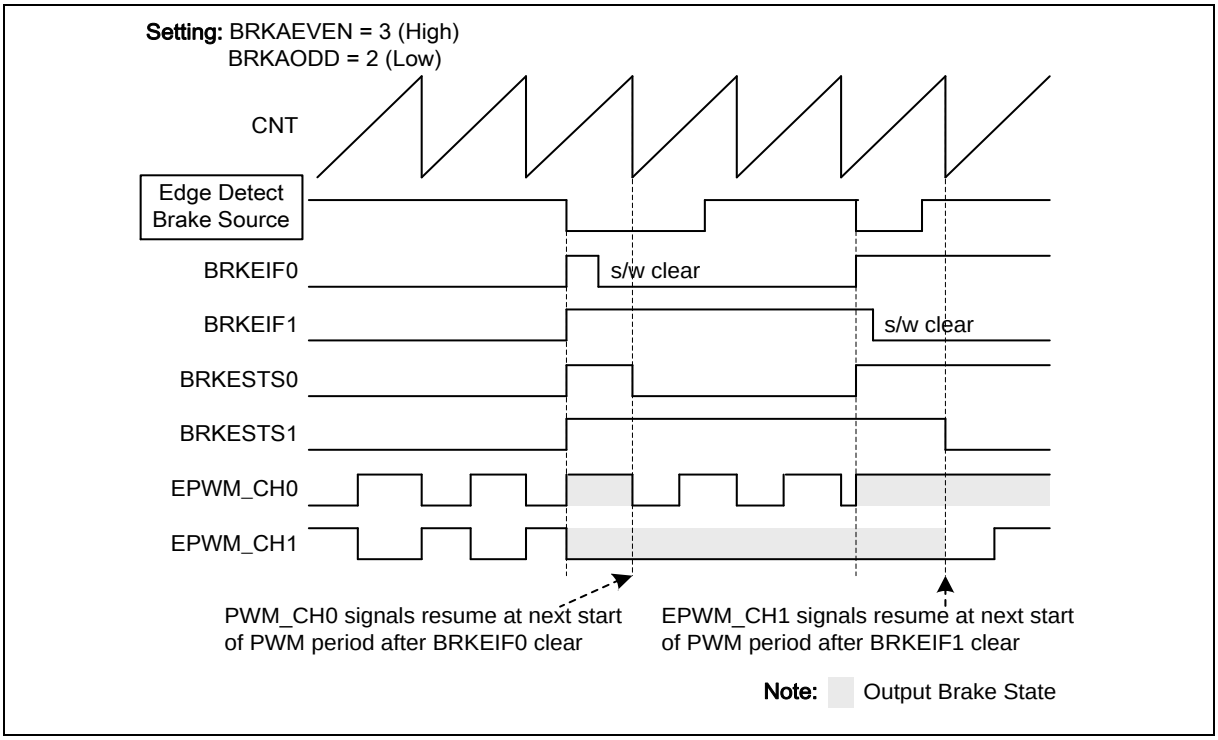


图 6.12-32EPWMx_CH0 和 EPWMx_CH1 通道组的边沿检测波形

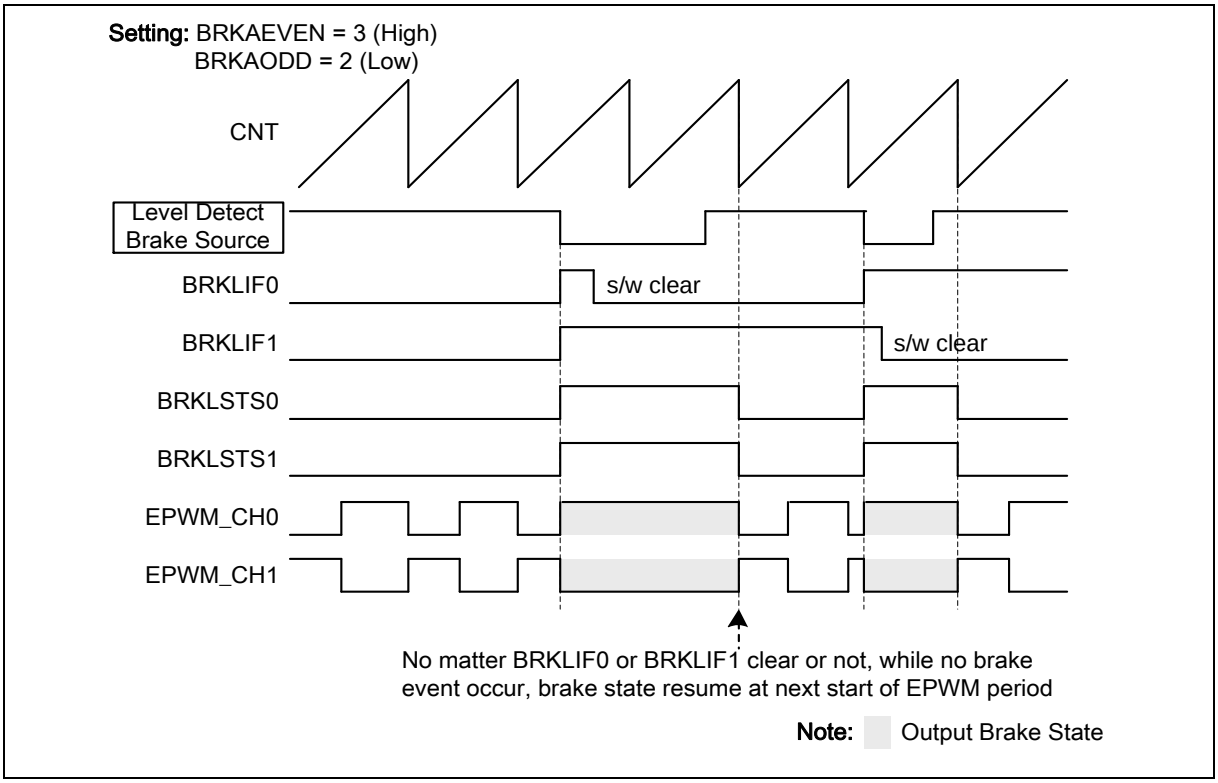


图 6.12-33EPWMx_CH0 和 EPWMx_CH1 通道组的电平检测波形

两种检测器检测七个同样的刹车源：两个从外部输入的信号，两个来自模拟比较器(ACMP)，一个来自EADC 结果监视器(EADCRM)，一个来自系统故障和一个来自软件触发，如图6.12-34。仅当内部ACMP0_O 或 ACMP1_O信号从低到高，ACMP刹车源才会被侦测到。

在以上描述的几个刹车源中，来自系统故障刹车源可以被指定为几种不同系统故障条件，这些故障条件，包括时钟故障，欠压侦测，SRAM奇偶校验错误和内核锁死。图6.12-35显示通过设置相应使能位，系统故障条件可以作为一个系统故障刹车源发送给EPWM刹车。

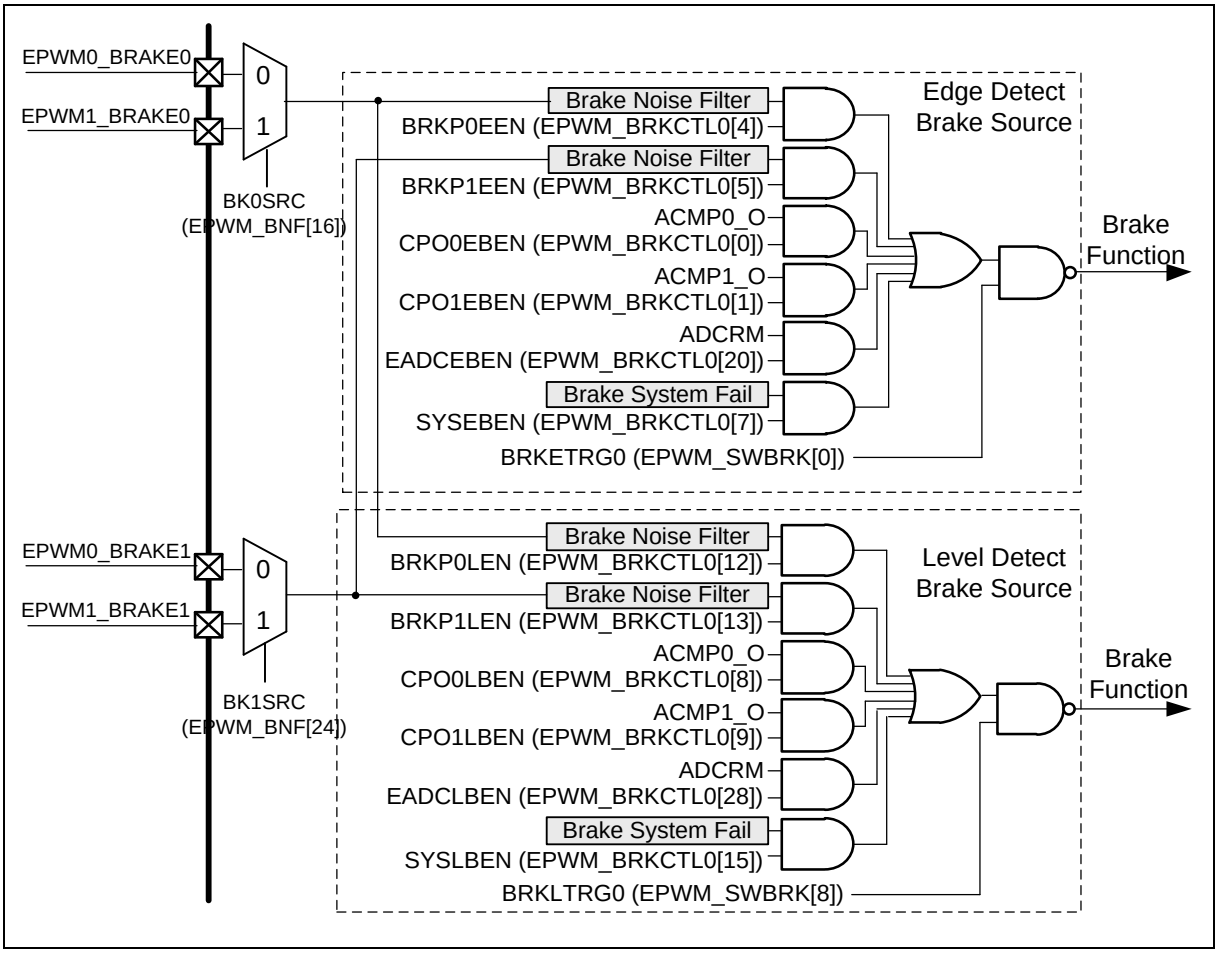


图 6.12-34 刹车源框图

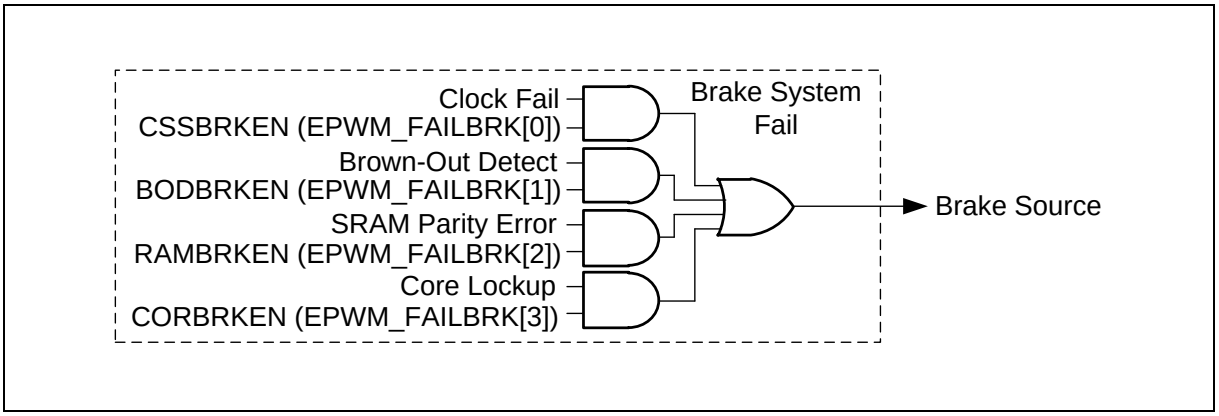


图 6.12-35 系统故障刹车框图

6.12.5.24 LEB功能

EPWM前沿消稳(LEB)的功能是用在 EPWM 的刹车源是 ACMP_O时，当 EPWM 通道输出管脚与 ACMP 输入管脚相邻时，EPWM 输出的跳变可能会产生干扰造成 ACMP 误判而触发 EPWM 刹车。因此，EPWM LEB 提供使用者可以设定一段时间（blanking Time），在这段时间内忽略 ACMP 送过来的触发信号。设置 LEBEN (EPWM_LEBCTL[0]) 使能该功能，LEB 源来自EPWM_CH0, EPWM_CH2 和 EPWM_CH4， SRCENn (EPWM_LEBCTL[10:8]) 作为输入源使能。LEB功能blanking Time由LEBCNT (EPWM_LEBCNT[8:0])决定。当LEB侦测到触发边沿，blanking Time会从LEBCNT+1 计数到 0，计数器时钟基于ECLK。如果新的触发事件发生，blanking计数器会复位到 LEBCNT且再次向下计数。 LEB 触发边沿可以是上升沿，下降沿或上升沿和下降沿都可以，通过设置 TRGTYPE (EPWM_LEBCTL[17:16])来决定。图6.12-36所示由EPWM_CH0 和 EPWM_CH4引起的LEB。

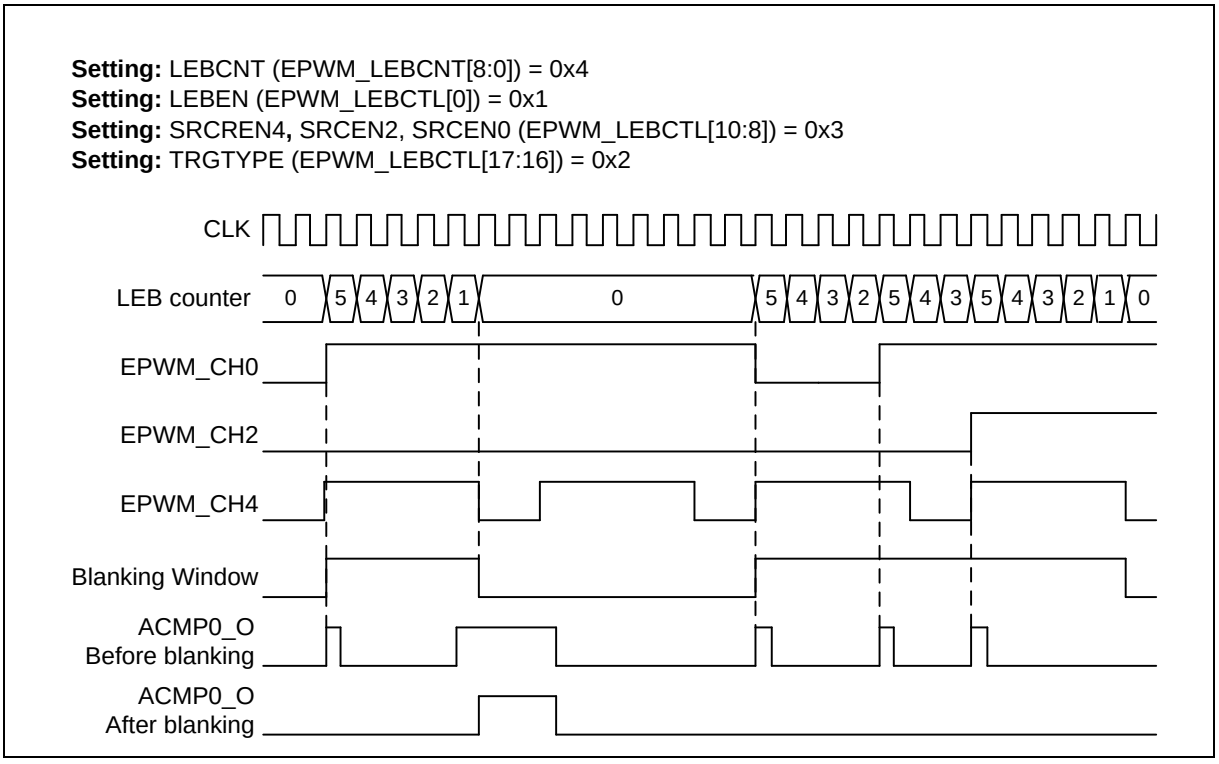
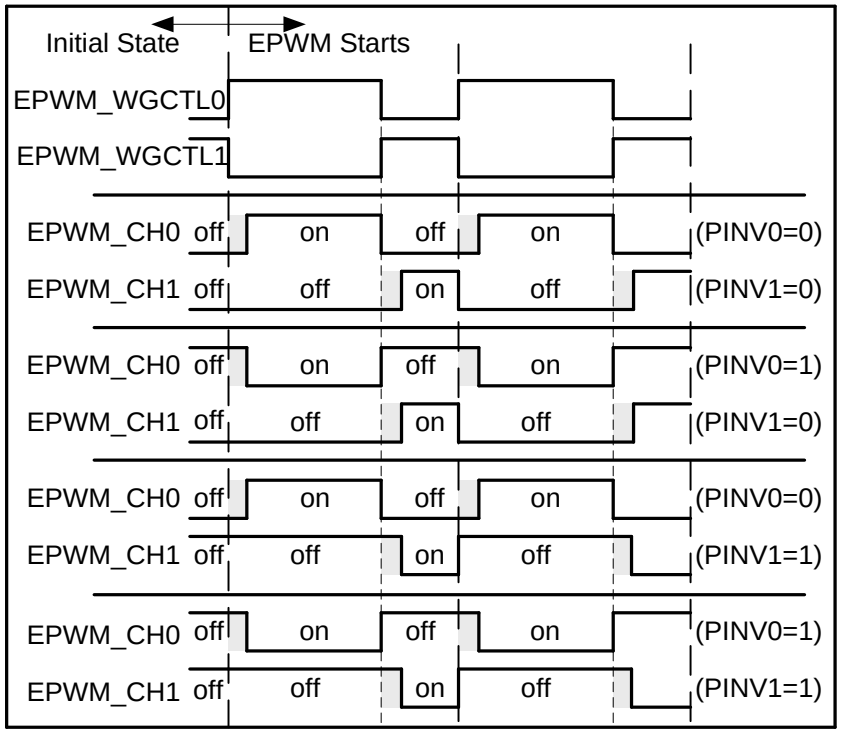


图 6.12-36EPWM LEB 功能波形

6.12.5.25 极性控制

每个EPWM端口从EPWM_CH0 到 EPWM_CH5都有一个独立极性控制模块来配置EPWM输出的极性状态。默认EPWM输出是高，意味着EPWM 关闭状态时为低，打开时为高。对每个独立的EPWM通道，EPWM的极性可以通过EPWM负极控制寄存器(EPWM_POLCTL)来设置。图6.12-37所示，EPWM不同极性设置开始前的初始状态。



Dead-time insertion; It is only effective in complementary mode
Note: PINVx: Negative Polarity control bits; It controls the EPWM output initial state and polarity, x denotes 0 or 1.

图 6.12-37 带上升沿死区插入初始状态和极性控制

6.12.5.26 EPWM中断发生器

EPWM的中断如图6.12-39和图6.12-40所示。

第一个EPWM 中断(EPWM_INT)来自EPWM互补组事件。在NVIC，EPWM共享EPWM_INT向量。计数器可以产生零点中断标志ZIFn (EPWM_INTSTS0[5:0])和周期点中断标志PIFn (EPWM_INTSTS0[13:8])。当EPWM通道n 计数器值等于存在EPWM_CMPDATn比较器值的时候，根据计数方向将触发不同的中断标志。如发生在向上计数则向上中断标志CMPUIFn (EPWM_INTSTS0[21:16])被置1，如果相反方向则向下计数中断标志CMPDIFn (EPWM_INTSTS0[29:24])被置1。如果相应的中断使能位置1，事件触发将产生中断信号。

EPWM_INT可以使用EPWM_IFAn (n=0~5)寄存器来累加已经触发的中断标志次数。通过设置IFAEN (EPWM_IFAn[31], n=0~5)中的一个为 1 来使能累加器。EPWM_INT会切换中断源，从每个事件触发中断切换为每次到达累加次数触发中断。

通过设置IFASEL (EPWM_IFAn[29:28], n=0~5)，用户可以选择4个中断源之一来累加每个通道的中断标志次数，且中断标志的次数会和IFACNT (EPWM_IFAn[15:0], n=0~5)比较。如果用户使能IFAIENn (EPWM_AINTEN[n], n=0~5)位，当中断累加器等于IFACNT会置位IFAIFn (EPWM_AINTSTS[n], n=0~5)作为EPWM_INT信号。每个通道的累加器中断也可以作为PDMA的请求源。图6.12-38是通道0使用EPWM_IFA0寄存器来每发生IFCNT0+1次中断事件输出一次EPWM_INT的例子。

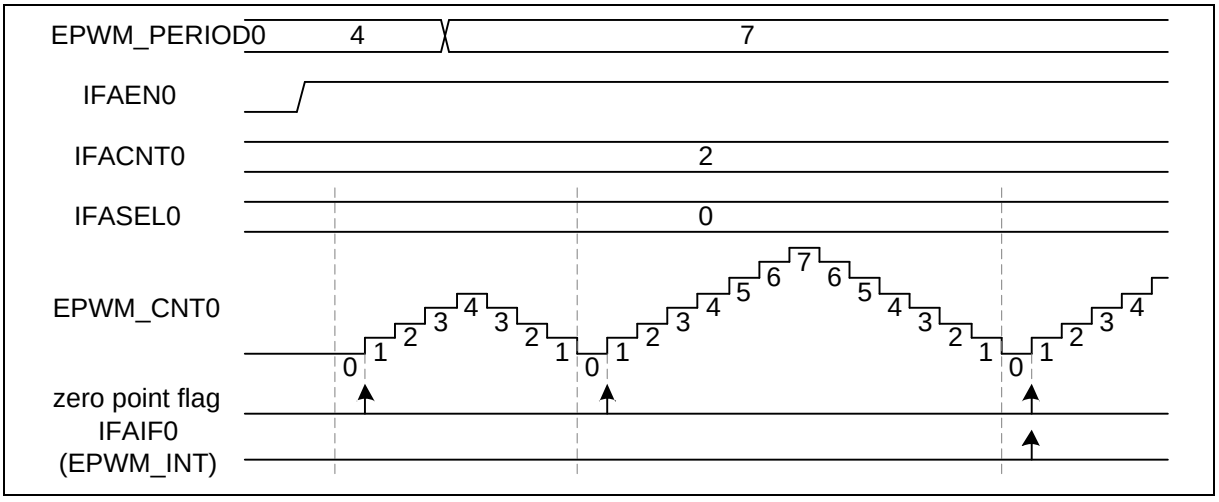


图 6.12-38EPWMx_CH0 累加中断波形

第二个中断是捕获中断(CAP_INT)，在NVIC中与EPWM_INT共用中断入口。当CAPRLIFn (EPWM_CAPIF[5:0])被触发，并且捕获上升沿中断使能位CAPRIENn (EPWM_CAPIEN[5:0])置1，可以产生CAP_INT中断。或者在下降沿条件，当捕获下降沿中断使能位CAPFIENn (EPWM_CAPIEN[13:8])置1， CAPFLIFn (EPWM_CAPIF[13:8])标志可以被触发。

第三个中断是刹车中断(BRK_INT)，在EPWM刹车章节有BRK_INT详细描述。图6.12-39表示EPWM中断架构。

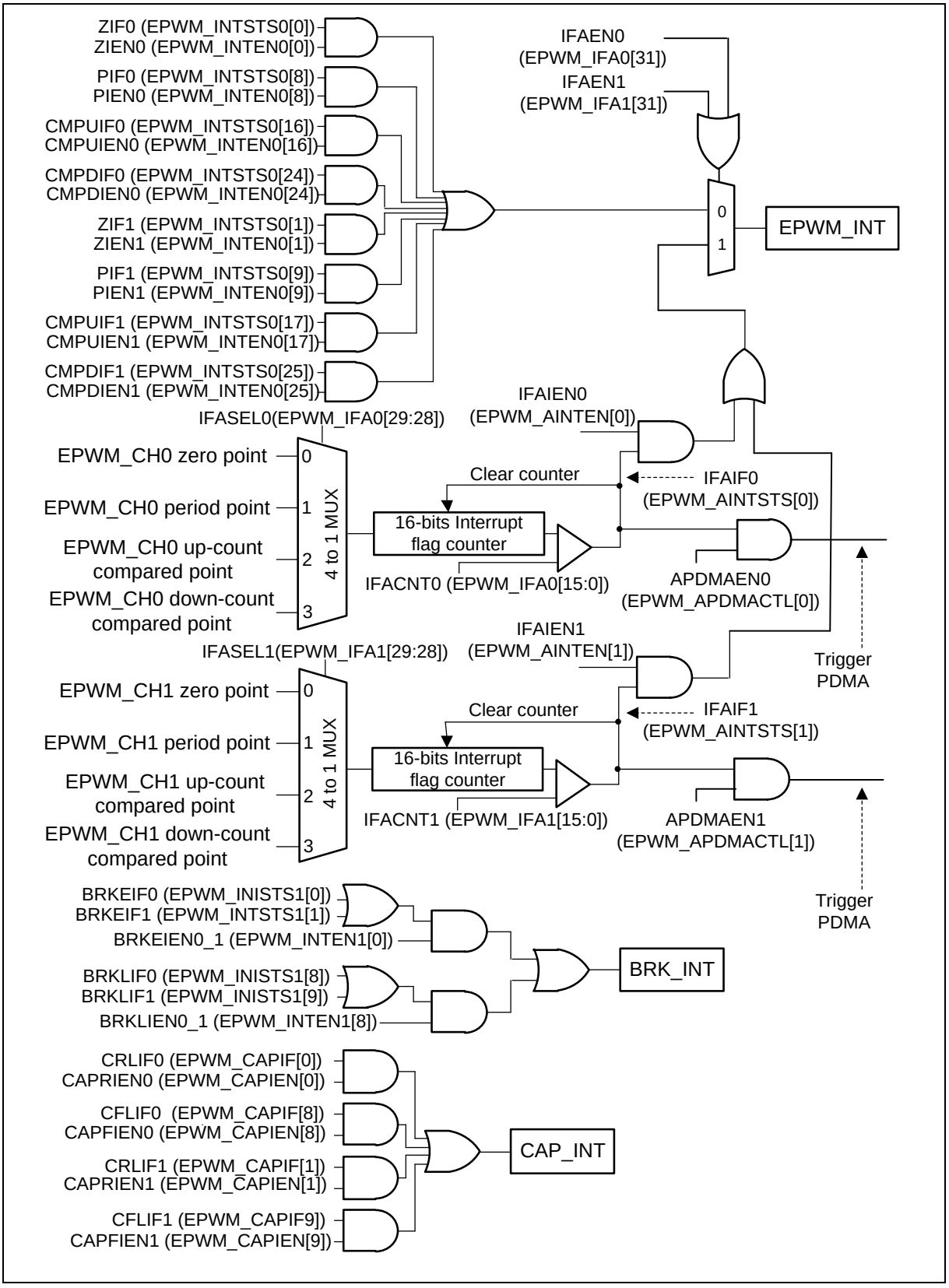


图 6.12-39EPWMx_CH0 和 EPWMx_CH1 组中断架构图

第四个中断是故障检测功能中断 (FLT_INT)，在NVIC中与EPWM_INT共用中断入口。如图6.12-40所示，当FDIENn (EPWM_FDIEN[n], n=0~5)置1，且FDIFn (EPWM_FDSTS[n], n=0~5)置1（通过检测输出短路），FLT_INT产生。

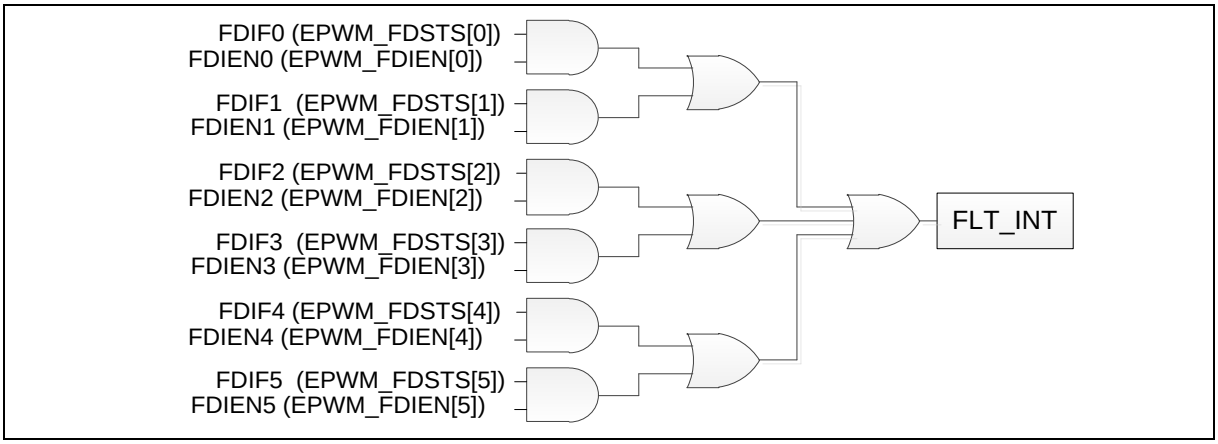


图6.12-40故障检测功能中断框图

EPWMx 通道	EPWM_INT、CAP_INT 和 FLT_INT 向量	BRK_INT 向量
EPWMx_CH0	EPWMxP0_IRQHandler	BRAKEx_IRQHandler
EPWMx_CH1	EPWMxP0_IRQHandler	BRAKEx_IRQHandler
EPWMx_CH2	EPWMxP1_IRQHandler	BRAKEx_IRQHandler
EPWMx_CH3	EPWMxP1_IRQHandler	BRAKEx_IRQHandler
EPWMx_CH4	EPWMxP2_IRQHandler	BRAKEx_IRQHandler
EPWMx_CH5	EPWMxP2_IRQHandler	BRAKEx_IRQHandler

表6.12-5EPWM中断向量列表

6.12.5.27 EPWM触发EADC/DAC 产生器

EPWM可以作为一个EADC转换触发源。每一个EPWM通道组都共享同样的触发源。设置EPWM_EADCTS0 和 EPWM_EADCTS1寄存器的TRGSELn来选择触发源，TRGSELn有TRGSEL0, TRGSEL1…TRGSEL5 ， 它 们 分 别 位 于 EPWM_EADCTS0[3:0], EPWM_EADCTS0[11:8], EPWM_EADCTS0[19:16], EPWM_EADCTS0[27:24], EPWM_EADCTS1[3:0] 和 EPWM_EADTS1[11:8]。设置相应的EPWM_EADCTS0 和 EPWM_EADCTS1 寄存器的TRGENn来使能触发输出到EADC, TRGENn 是 TRGEN0, TRGEN1, …, TRGEN5 ， 他 们 分 别 位 于 EPWM_EADCTS0[7], EPWM_EADCTS0[15], EPWM_EADCTS0[23], EPWM_EADCTS0[31], EPWM_EADCTS1[7] 和EPWM_EADCTS1[15]。数字n(n = 0,1, ..,5)表示EPWM通道数。

如图6.12-41所示，一对通道，有16个EPWM事件可以选择作为触发源。图6.12-42展示带预分频触发EADC框图。通过设置PSCENn (EPWM_EADCPSCCTR[n], n=0~5)位为1，当触发事件次数等于EADCPSCn (n=0~5)，EPWM将触发EADC。读PSCNTn (n=0~5)可以知道多少事件发生。如果PSCENn (n=0~5)为0，用户同样可以写初始化数据到PSCNTn (n=0~5)，预分频计数器将从这个值开始。如果PSCENn (n=0~5)从1设置到0，PSCNTn (n=0~5)将复位到0。图6.12-43为在上下计数模式，触

发EADC时序波形。

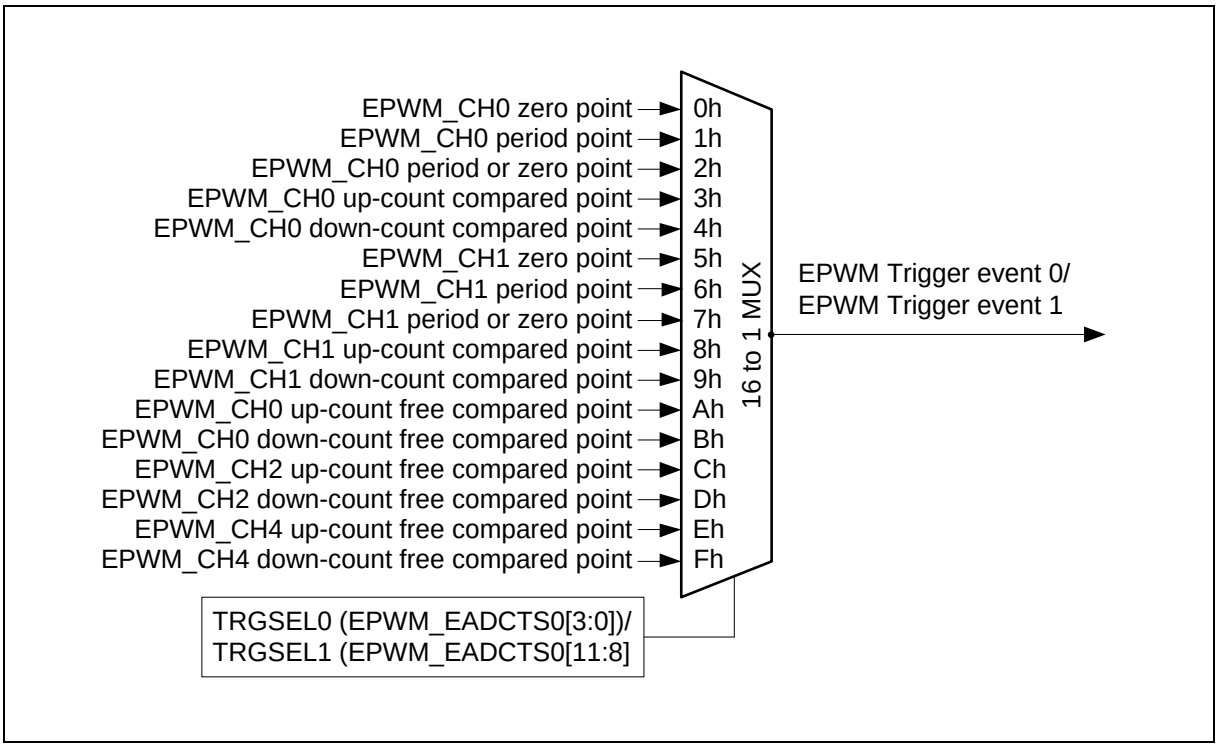


图 6.12-41EPWMx_CH0 和 EPWMx_CH1 对触发 EADC 事件。

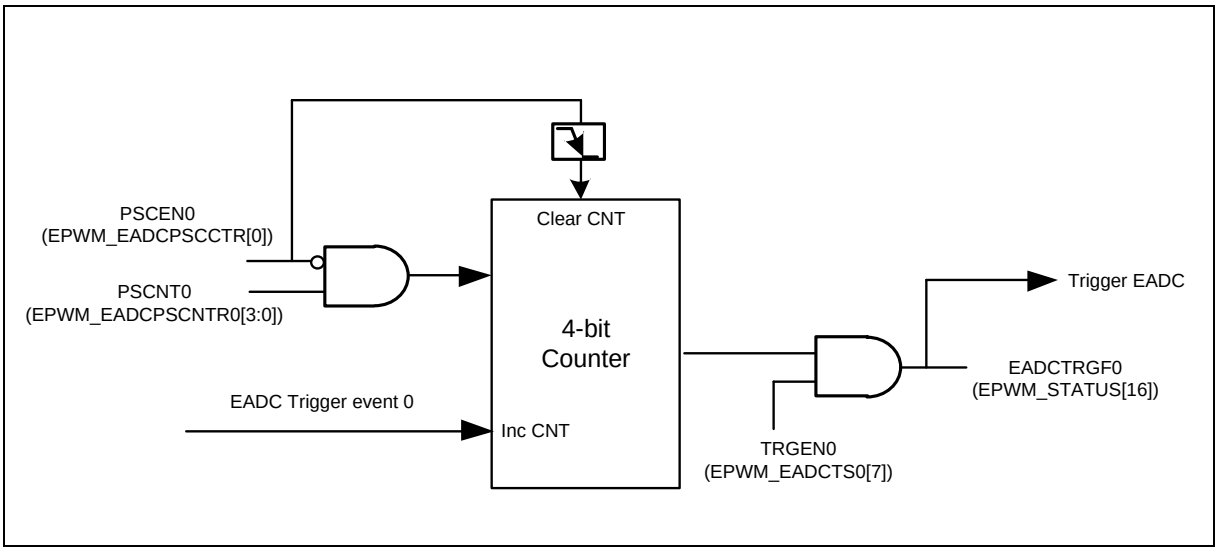


图 6.12-42EPWMx_CH0 触发 EADC 框图

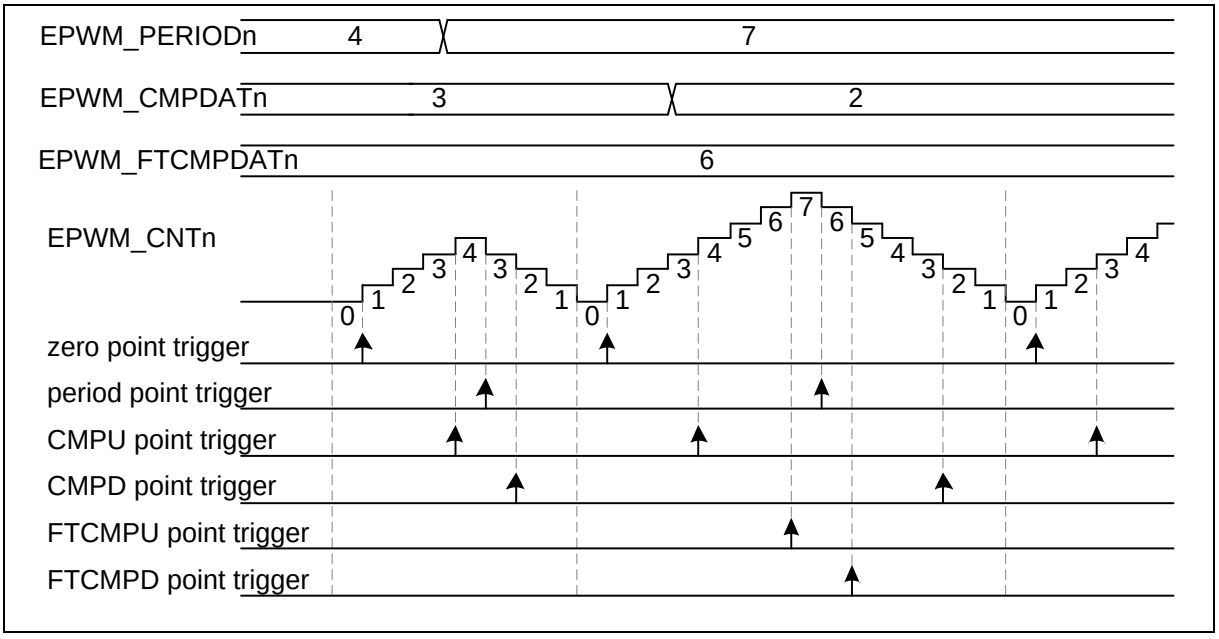


图 6.12-43 上下计数方式 EPWM 触发 EADC 的时序波形

EPWM也可以用来触发DAC转换。每个EPWM对的通道(CH0和CH1, CH2和CH3, CH4和CH5) 产生一个触发信号。使用EPWM触发DAC使能寄存器(EPWM_DACTRGEN)可以决定哪一个点来触发DAC。EPWM触发DAC的时序类似于触发EADC。然而， DAC 触发功能不包括来自于与FTCMPDAT比较的触发事件。也就是， FTCMPDATU和FTCMPDATD没有触发点。如EADC 触发所示。

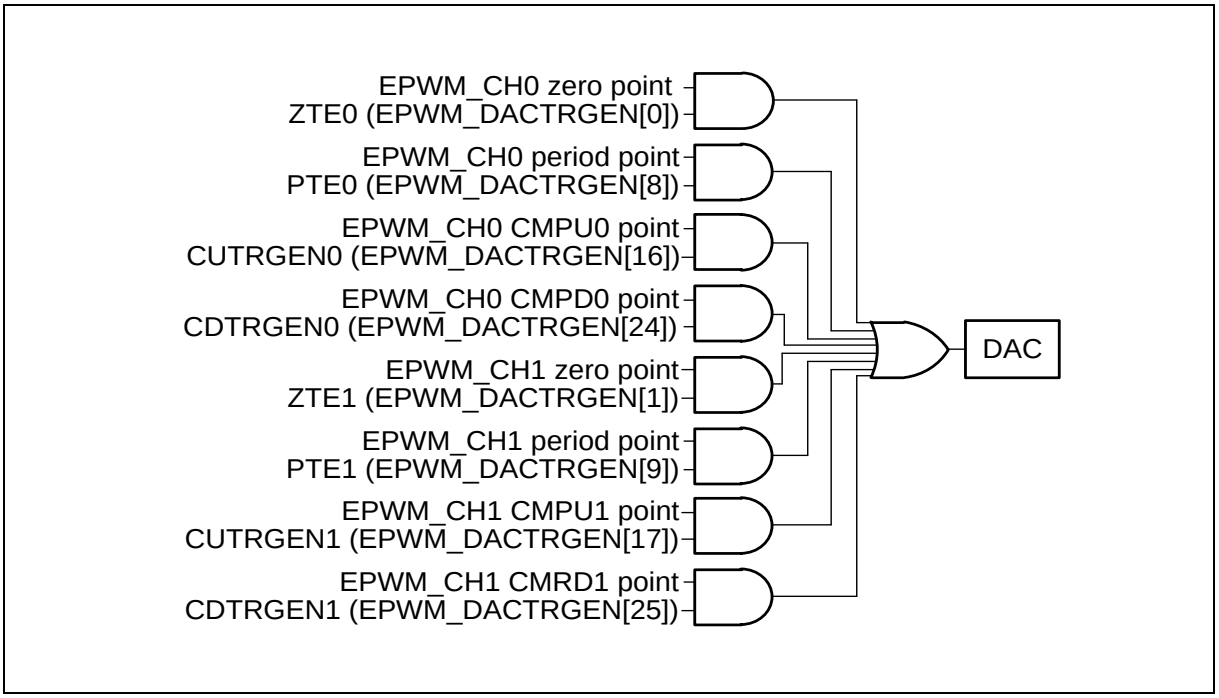


图 6.12-44EPWM_CH0 和 EPWM_CH1 组触发 DAC 框图

6.12.5.28 捕获操作

捕获输入通道和EPWM输出通道共用管脚和计数器。计数器可以是上或下计数方式。如果输入通道有上升沿或下降沿跳变时，捕获功能将EPWM计数器值分别锁存到RCAPDATn (EPWM_RCAPDATn[15:0])或FCAPDATn (EPWM_FCAPDATn[15:0]) 寄存器。如果上升沿或下降沿锁存发生并且相应通道n的上升沿或下降沿中断使能位被设置，捕获功能也将产生一个中断CAP_INT (使用 EPWM_INT向量)，在这里CAPRIENn (EPWM_CAPIEN[5:0])设置上升沿中断使能，CAPFIENn (EPWM_CAPIEN[13:8])设置下降沿中断使能。当上升沿或下降沿锁存发生，相应的EPWM计数器是否将重载EPWM_PERIODn值，这取决于设置RCRLDENn 或 FCRLDENn (RCRLDENn 和 FCRLDENn分别位于 EPWM_CAPCTL[21:16] 和 EPWM_CAPCTL[29:24])。注：相应的GPIO管脚必须通过使能CAPINENn (EPWM_CAPINEN[5:0])相应捕获通道n来配置为捕获功能。图6.12-45是通道0的捕获方块图。

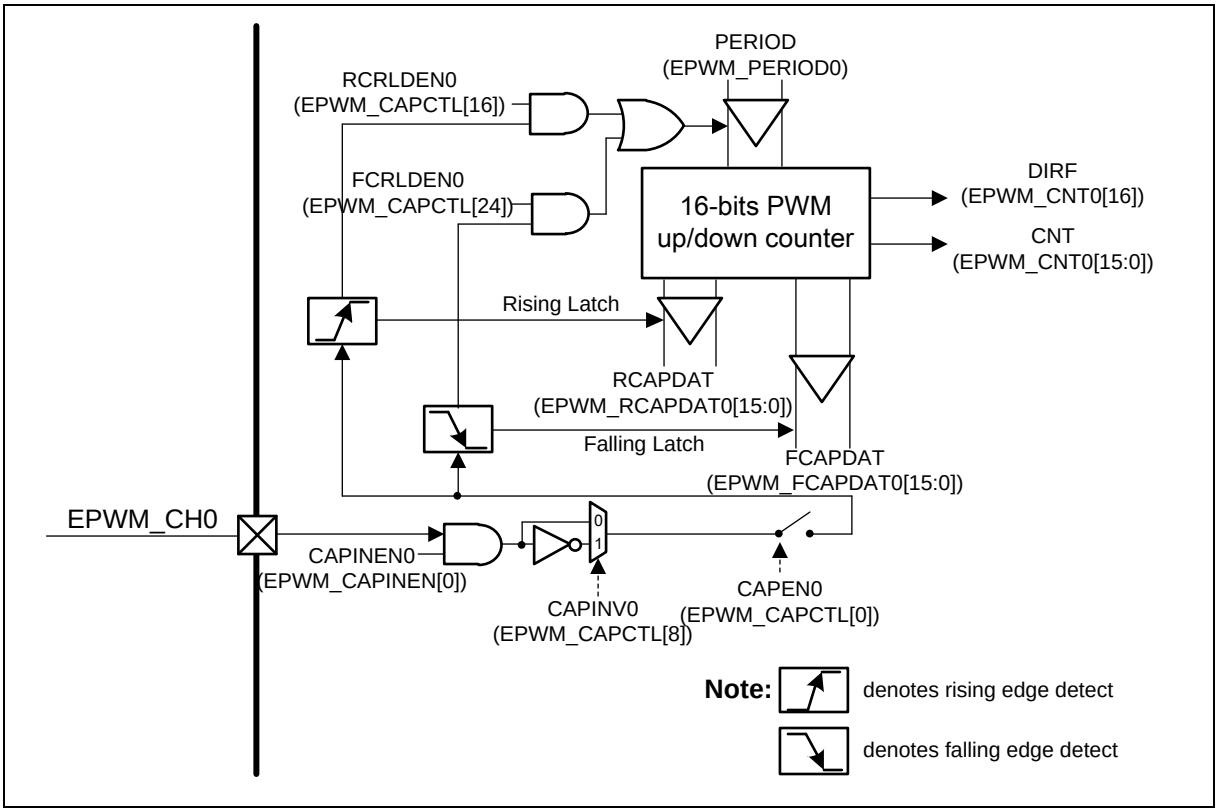


图 6.12-45EPWM_CH0 捕获框图

图6.12-46为捕获功能时序。在这个例子中，捕获计数器被设置成EPWM下计数器类型，而且周期PERIOD被设置成8，所以这个计数器计数的方向是从上到下的，从8到0。当输入捕获管脚检测到一个下降沿时，捕获功能把计数器数值锁存到EPWM_FCAPDATn中。当输入捕获管脚检测到一个上升沿，它锁存计数器数值到EPWM_RCAPDATn中。在这个时序框图中，当一开始检测到下降沿时，捕获器重新加载计数器的数值，数值为周期值（PERIOD），原因是使能了FCRLDENn。但是在第二次下降沿时，计数器不会重新加载，这是因为关闭了FCRLDENn。在这个例子中，计数器在捕获到上升沿时也被重新加载了，原因是RCRLDENn也被使能了。

另外，这种情况如果是设为向上计数方式，计数器将重载零并向上计数到PERIOD值。

图6.12-46也表示中断和中断标志产生的时序例子。当上升沿在通道n被检测到，相应位CRLIFn (EPWM_CAPIF[5:0])将被硬件设置。同样，通道n检测到下降沿相应位CFLIFn (EPWM_CAPIF[13:8])被硬件设置。CRLIFn和CFLIFn可以通过软件写1清除。如果CRLIFn被设置并且CAPRIENn被使能，捕获

功能将产生一个中断。如果CFLIFn被设置并且CAPFIENn被使能，中断也会产生。

在本图没有描述的一个情况是：当CRLIF已经设定了，如果上升锁存再次发生了，运行状态寄存器CRLIFOVn (EPWM_CAPSTS[5:0])将通过硬件被置起，来指示CAPRIF超载。同理，当下降锁存再次发生，对于中断标志CFLIF和超载状态CFLIFOVn (BEPWM_CAPSTS[13:8])，相同的硬件操作也会发生。

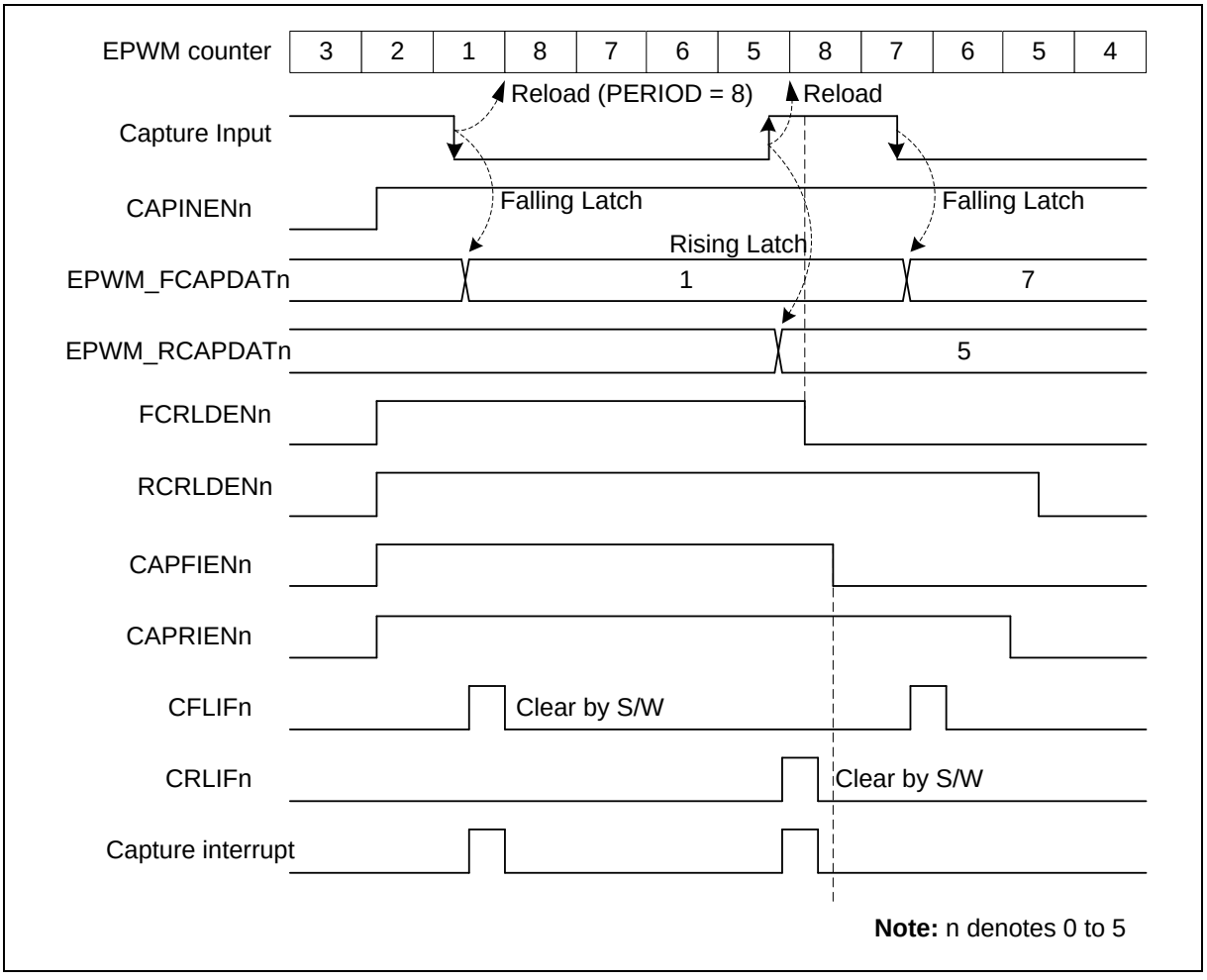


图 6.12-46 捕获操作波形图

满足以下条件捕获脉冲宽度计算，如下公式：

1. 捕获的正或负脉冲宽度短于计数器周期；
2. 计数器以减计数器类型运行；
3. 通过将 PWM_CAPCTL 寄存器的 FCRLDENn 和 RCRLDENn 位设置为 1，可以在下降和上升捕获事件重新加载计数器。

对于负脉冲的情况，通道低电平脉冲宽度为(EPWM_PERIODn + 1 - EPWM_RCAPDATn)个EPWM计数器时间。这里一个WPM计数器时间是(CLKPSC+1) * EPWMx_CLK。在图6.12-46的情况，低脉冲的宽度是8+1-5 = 4个EPWM计数器时间。

对于正脉冲的情况，通道高电平脉冲的宽度为(EPWM_PERIODn + 1 - EPWM_FCAPDATn) 个EPWM计数器时间。这里一个WPM计数器时间是(CLKPSC+1) * EPWMx_CLK。在图6.12-46的情况，高电平

的宽度是8+1-7 = 2个EPWM计数器时间。

6.12.5.29 捕获 PDMA 功能

当EPWM工作在捕获模式， EPWM 模块支持PDMA 数据搬移功能。当相应的PDMA 使能位 CHENn_m (CHEN0_1 在 EPWM_PDMACTL[0], CHEN2_3 在 EPWM_PDMACTL[8] ， CHEN4_5 在 EPWM_PDMACTL[16], n和m 代表互补通道)被使能，当捕获事件发生后，捕获模块将会产生一个到PDMA 控制器的请求。PDMA模块读取了捕获模块CAPBUF (EPWM_PDMACAPn_m[15:0], n, m代表互补通道) 寄存器的值，并搬移寄存器数据到内存后，PDMA 控制器将会给捕获模块产生一个应答。通过设置 CAPMODn_m (CAPMOD0_1 在 EPWM_PDMACTL[2:1], CAPMOD2_3 在 EPWM_PDMACTL[10:9] ， CAPMOD4_5 在 EPWM_PDMACTL[18:17]), PDMA可以搬移上升沿或下降沿或两个边沿捕获到的数据到内存。当使用PDMA 来搬移上升沿、下降沿的数据时，不要忘记设置 CAPORDn_m (CAPORD0_1 在 EPWM_PDMACTL[3], CAPORD2_3 在 EPWM_PDMACTL[11] ， CAPORD4_5 在 EPWM_PDMACTL[19])寄存器来设定搬移数据的顺序(下降沿捕获数据在先还是上升沿捕获数据在先). 互补通道共用一个PDMA 通道。因此，需要设置CHSELn_m (CHSEL0_1 (EPWM_PDMACTL[4]), CHSEL2_3 (EPWM_PDMACTL[12]) ， CHSEL4_5 (EPWM_PDMACTL[20])) 来确定是选择通道n还是通道m作为PDMA 通道。

图6.12-47是捕获PDMA的波形。在该例子当中， CHSEL0_1 (EPWM_PDMACTL[4]) 被设置为0。因此PDMA将会选择通道0 作为捕获数据搬移通道。 CAPMOD0_1 (EPWM_PDMACTL[2:1]) 被设为3，表明上升沿和下降沿捕获到的数据都会被搬移到内存。CAPORD0_1 (EPWM_PDMACTL[1]) 被置1，先搬移上升沿捕获到的数据，然后再搬移下降沿数据。如图6.12-47所示， CRLIF0 和 CFLIF0 信号的最后一部分有些重叠。EPWM_RCAPDAT0 的值11 将被装载到EPWM_PDMACAP0_1来等待搬移，但是EPWM_FCAPDAT0 的值不会被搬移。EPWM_PDMACAP0_1 保存了将通过PDMA搬移到内存的数据。图中HWDATA表示通过PDMA搬移的数据。

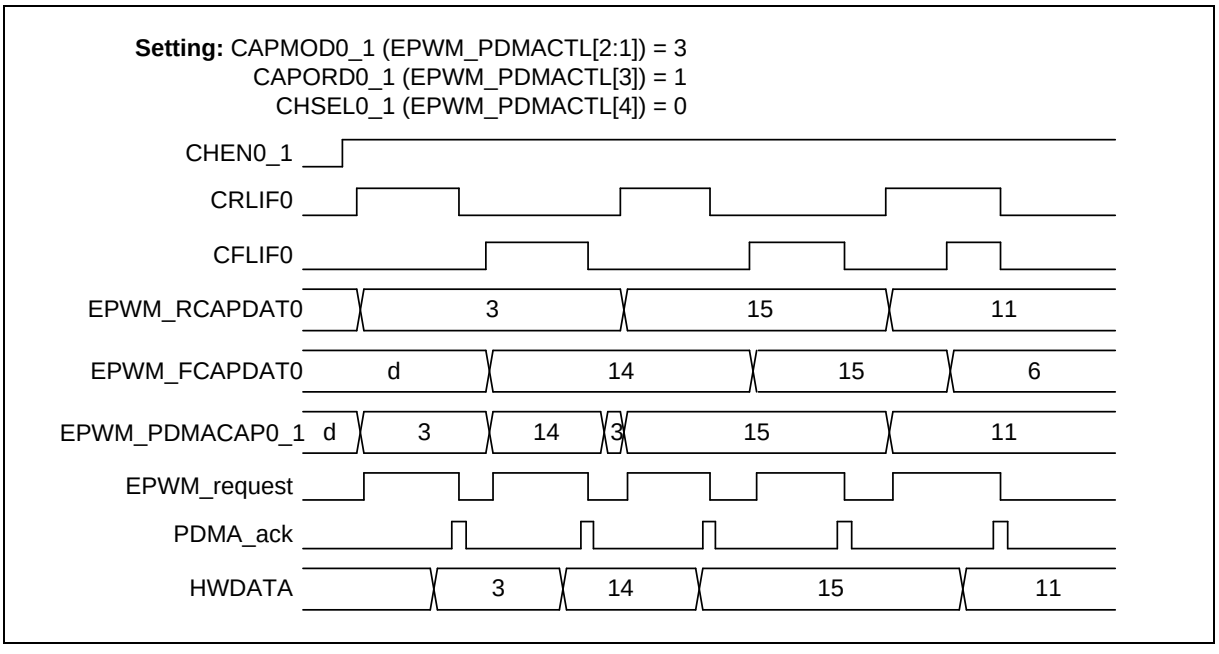


图 6.12-47 通道 0 捕获 PDMA 操作波形

6.12.5.30 累加器 PDMA 功能

EPWM 模块支持当累加器中断发生用PDMA传输功能。图6.12-48表示累加器 PDMA 功能架构。当对应的 PDMA 使能位 APDMAENn (EPWM_APDMACTL[n], n=0~5) 置位，累加器中断发生的时候也就是IFAIFn (EPWM_AINTSTS[n], n=0~5) 置 1，累加器模块会发送一个请求到PDMA控制器。PDMA控制器

在读内存数据且发送数据到指定寄存器(EPWM_PERIOD等)后会产生一个应答给累加器。所以用户可以使用这个功能来改变累加器中断频率。

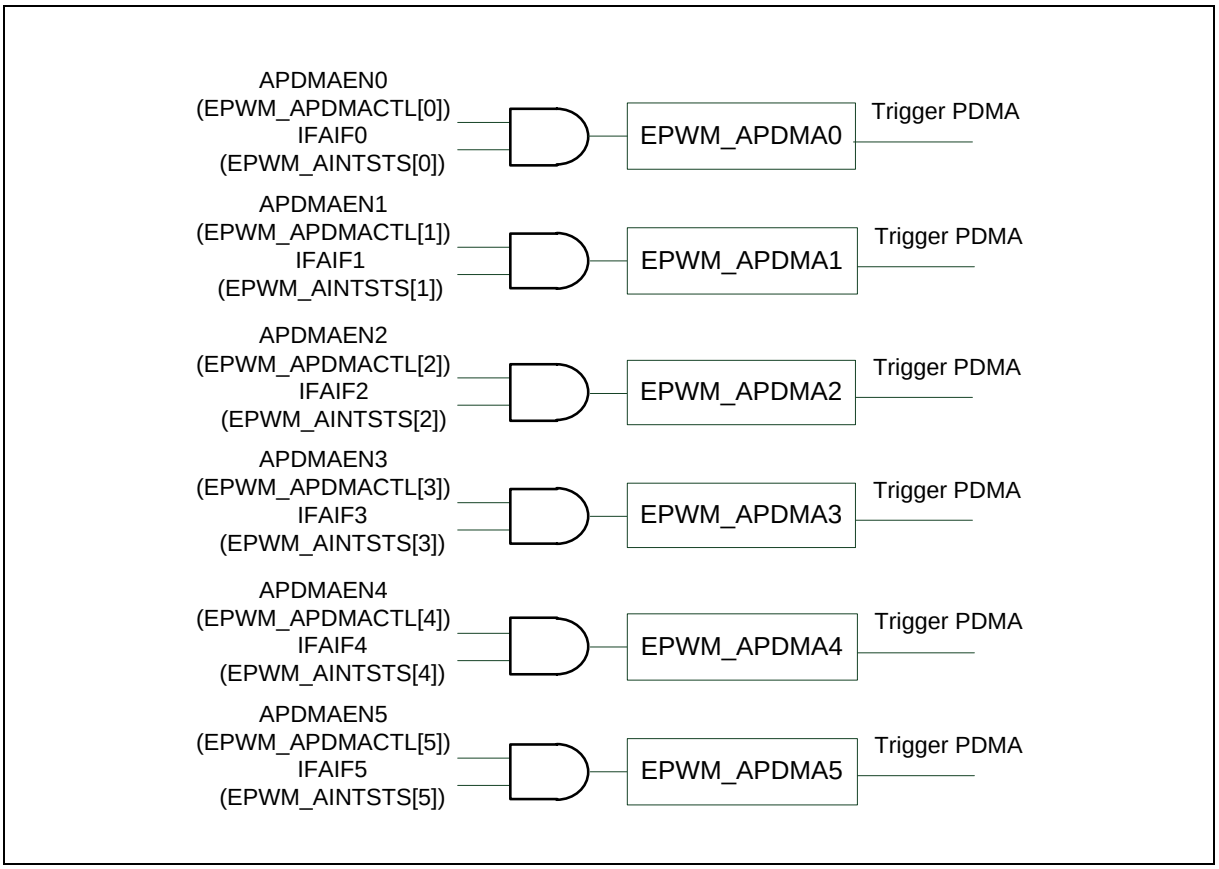


图 6.12-48 累加器 PDMA 功能架构

6.12.5.31 累加器停止模式

EPWM模块支持累加器停止模式，当累加器中断发生，停止计数。

图6.12-49展示累加器停止模式波形。当相应的STPMOD (EPWM_IFAn[24], n=0~5)设置，在计数器完成整个周期，累加器模块将禁用CNT_ENn (EPWM_CNTEN[n], n=0~5)。

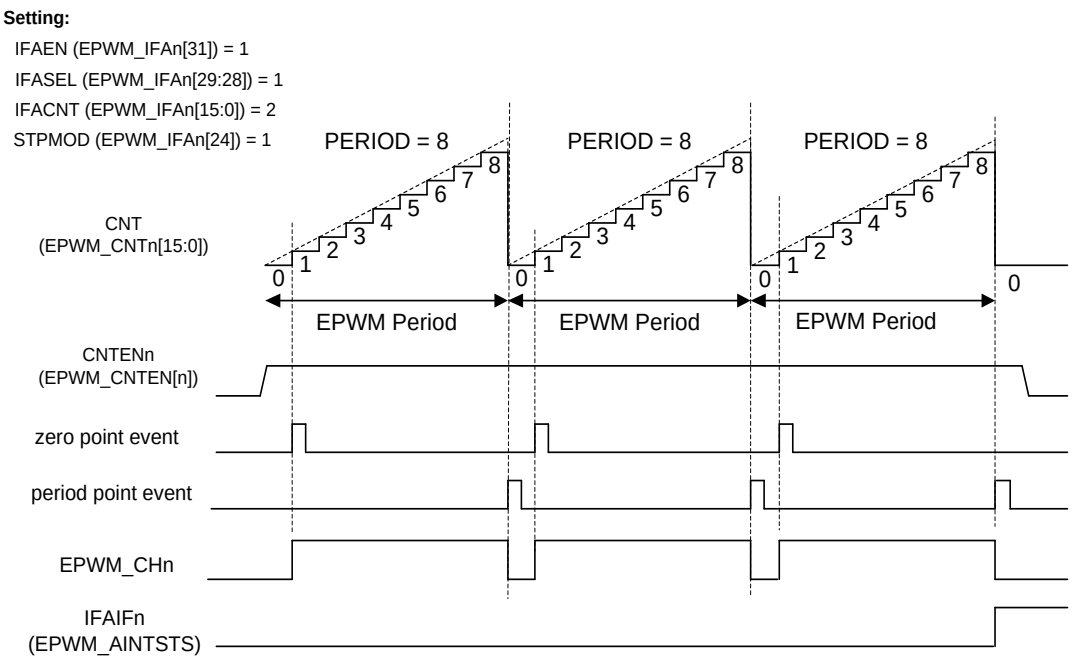


图 6.12-49EPWM 累加器停止模式波

6.12.5.32 故障检测功能

EPWM模块支持故障检测功能检测输出短路，图6.12-50为故障检测功能框图。当相应通道故障检测使能位FDENn(EPWM_FDEN[n])置1，两种状态下将开始检测，一种是EPWM输出从低变高或从高变低，另一种是POENn (EPWM_POEN[n], n=0~5)从0到1或从1到0。

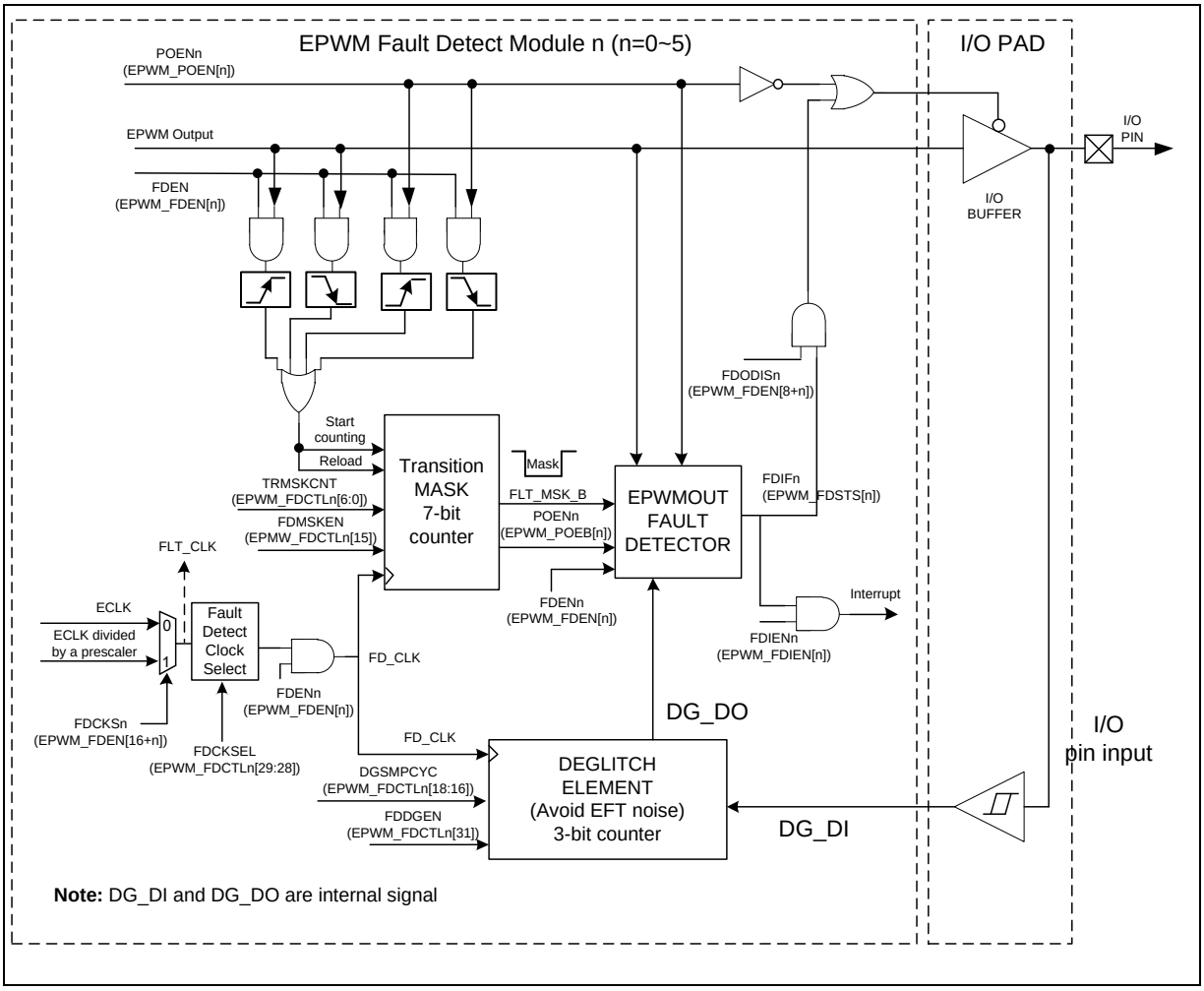


图 6.12-50 故障检测功能框图

图6.12-51展示屏蔽功能波形例子。设置FDMSKEN (EPWM_FDCTLn[15], n=0~5)使能屏蔽功能。刚开始，检测将禁止直到计数器从0计数到TRMSKCNT (EPWM_FDCTLn[6:0], n=0~5)。

图6.12-52展示抗尖峰脉冲功能波形例子。设置FDDGEN (EPWM_FDCTLn[31], n=0~5)使能抗尖峰脉冲功能。反馈信号将被采样DGSMPCYC (EPWM_FDCTLn[18:16], n=0~5) +1次，然后成为有效输入用于检测。FDCKSn (EPWM_FDEN[n], n=0~5)和FDCKSEL (EPWM_FDCTLn[29:28])是用于屏蔽和抗尖峰脉冲功能的时钟设置。详情，请参考寄存器描述。

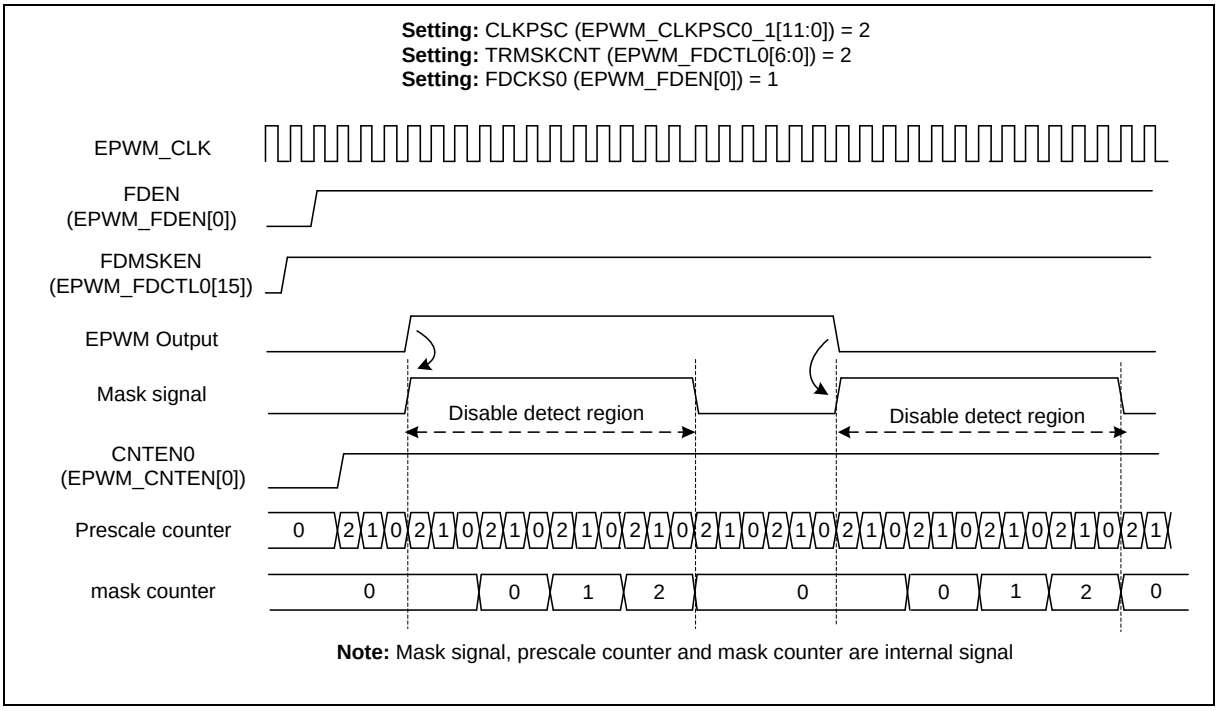


图 6.12-51 EPWM 屏蔽功能波形

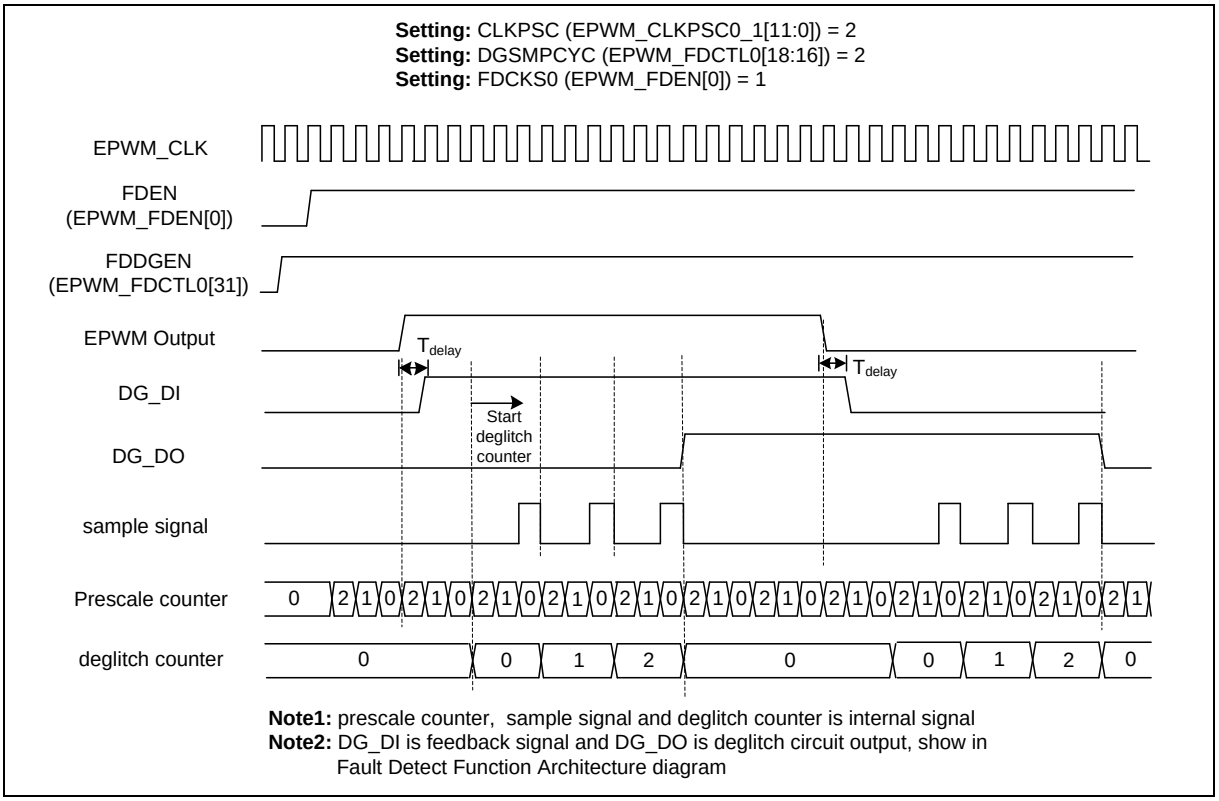


图 6.12-52 EPWM 抗尖峰脉冲功能波形

6.12.6 寄存器映射

R: 只读, **W**: 只写, **R/W**: 读/写

寄存器	偏移	R/W	描述	复位值
EPWM基地址: **EPWM0_BA = 0x4005_8000** **EPWM1_BA = 0x4005_9000** **x=0,1**				
EPWM_CTL0 **x=0,1**	EPWMx_BA+0x00	R/W	EPWM 控制寄存器0	0x0000_0000
EPWM_CTL1 **x=0,1**	EPWMx_BA+0x04	R/W	EPWM 控制寄存器1	0x0000_0000
EPWM_SYNC **x=0,1**	EPWMx_BA+0x08	R/W	EPWM同步寄存器	0x0000_0000
EPWM_SWSYNC **x=0,1**	EPWMx_BA+0x0C	R/W	EPWM 软件控制同步寄存器	0x0000_0000
EPWM_CLKSRC **x=0,1**	EPWMx_BA+0x10	R/W	EPWM 时钟源寄存器	0x0000_0000
EPWM_CLKPSC0_1 **x=0,1**	EPWMx_BA+0x14	R/W	EPWM时钟预分频寄存器0/1	0x0000_0000
EPWM_CLKPSC2_3 **x=0,1**	EPWMx_BA+0x18	R/W	EPWM时钟预分频寄存器2/3	0x0000_0000
EPWM_CLKPSC4_5 **x=0,1**	EPWMx_BA+0x1C	R/W	EPWM时钟预分频寄存器4/5	0x0000_0000
EPWM_CNTEN **x=0,1**	EPWMx_BA+0x20	R/W	EPWM 计数使能寄存器	0x0000_0000
EPWM_CNTCLR **x=0,1**	EPWMx_BA+0x24	R/W	EPWM清计数器寄存器	0x0000_0000
EPWM_LOAD **x=0,1**	EPWMx_BA+0x28	R/W	EPWM装载寄存器	0x0000_0000
EPWM_PERIOD0 **x=0,1**	EPWMx_BA+0x30	R/W	EPWM周期寄存器0	0x0000_0000
EPWM_PERIOD1 **x=0,1**	EPWMx_BA+0x34	R/W	EPWM周期寄存器1	0x0000_0000
EPWM_PERIOD2 **x=0,1**	EPWMx_BA+0x38	R/W	EPWM周期寄存器2	0x0000_0000
EPWM_PERIOD3 **x=0,1**	EPWMx_BA+0x3C	R/W	EPWM周期寄存器3	0x0000_0000
EPWM_PERIOD4	EPWMx_BA+0x40	R/W	EPWM周期寄存器4	0x0000_0000

x=0,1				
EPWM_PERIOD5 x=0,1	EPWMx_BA+0x44	R/W	EPWM周期寄存器5	0x0000_0000
EPWM_CMPDAT0 x=0,1	EPWMx_BA+0x50	R/W	EPWM比较寄存器0	0x0000_0000
EPWM_CMPDAT1 x=0,1	EPWMx_BA+0x54	R/W	EPWM比较寄存器1	0x0000_0000
EPWM_CMPDAT2 x=0,1	EPWMx_BA+0x58	R/W	EPWM比较寄存器2	0x0000_0000
EPWM_CMPDAT3 x=0,1	EPWMx_BA+0x5C	R/W	EPWM比较寄存器3	0x0000_0000
EPWM_CMPDAT4 x=0,1	EPWMx_BA+0x60	R/W	EPWM比较寄存器4	0x0000_0000
EPWM_CMPDAT5 x=0,1	EPWMx_BA+0x64	R/W	EPWM比较寄存器5	0x0000_0000
EPWM_DTCTL0_1 x=0,1	EPWMx_BA+0x70	R/W	EPWM死区控制寄存器0/1	0x0000_0000
EPWM_DTCTL2_3 x=0,1	EPWMx_BA+0x74	R/W	EPWM 死区控制寄存器2/3	0x0000_0000
EPWM_DTCTL4_5 x=0,1	EPWMx_BA+0x78	R/W	EPWM 死区控制寄存器4/5	0x0000_0000
EPWM_PHS0_1 x=0,1	EPWMx_BA+0x80	R/W	EPWM计数器相位寄存器0/1	0x0000_0000
EPWM_PHS2_3 x=0,1	EPWMx_BA+0x84	R/W	EPWM计数器相位寄存器2/3	0x0000_0000
EPWM_PHS4_5 x=0,1	EPWMx_BA+0x88	R/W	EPWM计数器相位寄存器4/5	0x0000_0000
EPWM_CNT0 x=0,1	EPWMx_BA+0x90	R	EPWM计数寄存器0	0x0000_0000
EPWM_CNT1 x=0,1	EPWMx_BA+0x94	R	EPWM 计数寄存器1	0x0000_0000
EPWM_CNT2 x=0,1	EPWMx_BA+0x98	R	EPWM 计数寄存器2	0x0000_0000
EPWM_CNT3 x=0,1	EPWMx_BA+0x9C	R	EPWM计数寄存器3	0x0000_0000
EPWM_CNT4 x=0,1	EPWMx_BA+0xA0	R	EPWM计数寄存器4	0x0000_0000
EPWM_CNT5 x=0,1	EPWMx_BA+0xA4	R	EPWM 计数寄存器5	0x0000_0000

EPWM_WGCTL0 x=0,1	EPWMx_BA+0xB0	R/W	EPWM 发生寄存器0	0x0000_0000
EPWM_WGCTL1 x=0,1	EPWMx_BA+0xB4	R/W	EPWM发生寄存器1	0x0000_0000
EPWM_MSKEN x=0,1	EPWMx_BA+0xB8	R/W	EPWM 屏蔽使能寄存器	0x0000_0000
EPWM_MSK x=0,1	EPWMx_BA+0xBC	R/W	EPWM屏蔽数据寄存器	0x0000_0000
EPWM_BNF x=0,1	EPWMx_BA+0xC0	R/W	EPWM 刹车噪音滤波寄存器	0x0000_0000
EPWM_FAILBRK x=0,1	EPWMx_BA+0xC4	R/W	EPWM 系统故障刹车控制寄存器	0x0000_0000
EPWM_BRKCTL0_1 x=0,1	EPWMx_BA+0xC8	R/W	EPWM 刹车边沿检测控制寄存器0/1	0x0000_0000
EPWM_BRKCTL2_3 x=0,1	EPWMx_BA+0xCC	R/W	EPWM 刹车边沿检测控制寄存器2/3	0x0000_0000
EPWM_BRKCTL4_5 x=0,1	EPWMx_BA+0xD0	R/W	EPWM 刹车边沿检测控制寄存器4/5	0x0000_0000
EPWM_POLCTL x=0,1	EPWMx_BA+0xD4	R/W	EPWM管脚极性反转寄存器	0x0000_0000
EPWM_POEN x=0,1	EPWMx_BA+0xD8	R/W	EPWM输出使能寄存器	0x0000_0000
EPWM_SWBRK x=0,1	EPWMx_BA+0xDC	W	EPWM软件刹车控制寄存器	0x0000_0000
EPWM_INTEN0 x=0,1	EPWMx_BA+0xE0	R/W	EPWM 中断使能寄存器0	0x0000_0000
EPWM_INTEN1 x=0,1	EPWMx_BA+0xE4	R/W	EPWM中断使能寄存器1	0x0000_0000
EPWM_INTSTS0 x=0,1	EPWMx_BA+0xE8	R/W	EPWM中断标志寄存器0	0x0000_0000
EPWM_INTSTS1 x=0,1	EPWMx_BA+0xEC	R/W	EPWM 中断标志寄存器1	0x0000_0000
EPWM_DACTRGEN x=0,1	EPWMx_BA+0xF4	R/W	EPWM 触发 DAC使能寄存器	0x0000_0000
EPWM_EADCTS0 x=0,1	EPWMx_BA+0xF8	R/W	EPWM触发EADC源选择寄存器0	0x0000_0000
EPWM_EADCTS1 x=0,1	EPWMx_BA+0xFC	R/W	EPWM触发EADC源选择寄存器1	0x0000_0000
EPWM_FTCMPDAT0_1	EPWMx_BA+0x100	R/W	EPWM 自由触发比较寄存器0/1	0x0000_0000

x=0,1				
EPWM_FTCMPDAT2_3 x=0,1	EPWMx_BA+0x104	R/W	EPWM 自由触发比较寄存器2/3	0x0000_0000
EPWM_FTCMPDAT4_5 x=0,1	EPWMx_BA+0x108	R/W	EPWM 自由触发比较寄存器4/5	0x0000_0000
EPWM_SSCTL x=0,1	EPWMx_BA+0x110	R/W	EPWM 同步开始控制寄存器	0x0000_0000
EPWM_SSTRG x=0,1	EPWMx_BA+0x114	W	EPWM同步开始触发寄存器	0x0000_0000
EPWM_LEBCTL x=0,1	EPWMx_BA+0x118	R/W	EPWM前沿消稳控制寄存器	0x0000_0000
EPWM_LEBCNT x=0,1	EPWMx_BA+0x11C	R/W	EPWM前沿消稳计数器寄存器	0x0000_0000
EPWM_STATUS x=0,1	EPWMx_BA+0x120	R/W	EPWM 状态寄存器	0x0000_0000
EPWM_IFA0 x=0,1	EPWMx_BA+0x130	R/W	EPWM 中断标志累加器寄存器0	0x0000_0000
EPWM_IFA1 x=0,1	EPWMx_BA+0x134	R/W	EPWM中断标志累加器寄存器1	0x0000_0000
EPWM_IFA2 x=0,1	EPWMx_BA+0x138	R/W	EPWM中断标志累加器寄存器2	0x0000_0000
EPWM_IFA3 x=0,1	EPWMx_BA+0x13C	R/W	EPWM中断标志累加器寄存器3	0x0000_0000
EPWM_IFA4 x=0,1	EPWMx_BA+0x140	R/W	EPWM中断标志累加器寄存器4	0x0000_0000
EPWM_IFA5 x=0,1	EPWMx_BA+0x144	R/W	EPWM中断标志累加器寄存器5	0x0000_0000
EPWM_AINTSTS x=0,1	EPWMx_BA+0x150	R/W	EPWM累加器中断标志寄存器	0x0000_0000
EPWM_AINTEN x=0,1	EPWMx_BA+0x154	R/W	EPWM累加器中断使能寄存器	0x0000_0000
EPWM_APDMACTL x=0,1	EPWMx_BA+0x158	R/W	EPWM 累加器 PDMA 控制寄存器	0x0000_0000
EPWM_FDEN x=0,1	EPWMx_BA+0x160	R/W	EPWM故障检测使能寄存器	0x0000_0000
EPWM_FDCTL0 x=0,1	EPWMx_BA+0x164	R/W	EPWM故障检测控制寄存器0	0x0000_0000
EPWM_FDCTL1 x=0,1	EPWMx_BA+0x168	R/W	EPWM故障检测控制寄存器1	0x0000_0000

EPWM_FDCTL2 x=0,1	EPWMx_BA+0x16C	R/W	EPWM故障检测控制寄存器2	0x0000_0000
EPWM_FDCTL3 x=0,1	EPWMx_BA+0x170	R/W	EPWM故障检测控制寄存器3	0x0000_0000
EPWM_FDCTL4 x=0,1	EPWMx_BA+0x174	R/W	EPWM故障检测控制寄存器4	0x0000_0000
EPWM_FDCTL5 x=0,1	EPWMx_BA+0x178	R/W	EPWM故障检测控制寄存器5	0x0000_0000
EPWM_FDIEN x=0,1	EPWMx_BA+0x17C	R/W	EPWM故障检测中断使能寄存器	0x0000_0000
EPWM_FDSTS x=0,1	EPWMx_BA+0x180	R/W	EPWM故障检测中断标志寄存器	0x0000_0000
EPWM_EADCPSCCTL x=0,1	EPWMx_BA+0x184	R/W	EPWM触发EADC预分频控制寄存器	0x0000_0000
EPWM_EADCPSC0 x=0,1	EPWMx_BA+0x188	R/W	EPWM触发EADC预分频寄存器0	0x0000_0000
EPWM_EADCPSC1 x=0,1	EPWMx_BA+0x18C	R/W	EPWM触发EADC预分频寄存器1	0x0000_0000
EPWM_EADCPSCNT0 x=0,1	EPWMx_BA+0x190	R/W	EPWM触发EADC预分频计数器寄存器0	0x0000_0000
EPWM_EADCPSCNT1 x=0,1	EPWMx_BA+0x194	R/W	EPWM触发EADC预分频计数器寄存器1	0x0000_0000
EPWM_CAPINEN x=0,1	EPWMx_BA+0x200	R/W	EPWM 捕获输入使能寄存器	0x0000_0000
EPWM_CAPCTL x=0,1	EPWMx_BA+0x204	R/W	EPWM捕获控制寄存器	0x0000_0000
EPWM_CAPSTS x=0,1	EPWMx_BA+0x208	R	EPWM 捕获状态寄存器	0x0000_0000
EPWM_RCAPDAT0 x=0,1	EPWMx_BA+0x20C	R	EPWM 上升沿捕获数据寄存器0	0x0000_0000
EPWM_FCAPDAT0 x=0,1	EPWMx_BA+0x210	R	EPWM 下降沿捕获数据寄存器0	0x0000_0000
EPWM_RCAPDAT1 x=0,1	EPWMx_BA+0x214	R	EPWM上升沿捕获数据寄存器1	0x0000_0000
EPWM_FCAPDAT1 x=0,1	EPWMx_BA+0x218	R	EPWM下降沿捕获数据寄存器1	0x0000_0000
EPWM_RCAPDAT2 x=0,1	EPWMx_BA+0x21C	R	EPWM 上升沿捕获数据寄存器2	0x0000_0000
EPWM_FCAPDAT2 x=0,1	EPWMx_BA+0x220	R	EPWM下降沿捕获数据寄存器2	0x0000_0000

EPWM_RCAPDAT3 x=0,1	EPWMx_BA+0x224	R	EPWM 上升沿捕获数据寄存器3	0x0000_0000
EPWM_FCAPDAT3 x=0,1	EPWMx_BA+0x228	R	EPWM 下降沿捕获数据寄存器3	0x0000_0000
EPWM_RCAPDAT4 x=0,1	EPWMx_BA+0x22C	R	EPWM 上升沿捕获数据寄存器4	0x0000_0000
EPWM_FCAPDAT4 x=0,1	EPWMx_BA+0x230	R	EPWM 下降沿捕获数据寄存器4	0x0000_0000
EPWM_RCAPDAT5 x=0,1	EPWMx_BA+0x234	R	EPWM上升沿捕获数据寄存器5	0x0000_0000
EPWM_FCAPDAT5 x=0,1	EPWMx_BA+0x238	R	EPWM 下降沿捕获数据寄存器5	0x0000_0000
EPWM_PDMACTL x=0,1	EPWMx_BA+0x23C	R/W	EPWM PDMA 控制寄存器	0x0000_0000
EPWM_PDMACAP0_1 x=0,1	EPWMx_BA+0x240	R	EPWM捕获通道01 PDMA 寄存器	0x0000_0000
EPWM_PDMACAP2_3 x=0,1	EPWMx_BA+0x244	R	EPWM捕获通道23 PDMA 寄存器	0x0000_0000
EPWM_PDMACAP4_5 x=0,1	EPWMx_BA+0x248	R	EPWM捕获通道45 PDMA 寄存器	0x0000_0000
EPWM_CAPIEN x=0,1	EPWMx_BA+0x250	R/W	EPWM捕获中断使能寄存器	0x0000_0000
EPWM_CAPIF x=0,1	EPWMx_BA+0x254	R/W	EPWM捕获中断标志寄存器	0x0000_0000
EPWM_PBUF0 x=0,1	EPWMx_BA+0x304	R	EPWM PERIOD0缓存	0x0000_0000
EPWM_PBUF1 x=0,1	EPWMx_BA+0x308	R	EPWM PERIOD1缓存	0x0000_0000
EPWM_PBUF2 x=0,1	EPWMx_BA+0x30C	R	EPWM PERIOD2缓存	0x0000_0000
EPWM_PBUF3 x=0,1	EPWMx_BA+0x310	R	EPWM PERIOD3缓存	0x0000_0000
EPWM_PBUF4 x=0,1	EPWMx_BA+0x314	R	EPWM PERIOD4缓存	0x0000_0000
EPWM_PBUF5 x=0,1	EPWMx_BA+0x318	R	EPWM PERIOD5缓存	0x0000_0000
EPWM_CMPBUF0 x=0,1	EPWMx_BA+0x31C	R	EPWM CMPDAT0缓存	0x0000_0000
EPWM_CMPBUF1 x=0,1	EPWMx_BA+0x320	R	EPWM CMPDAT1缓存	0x0000_0000

EPWM_CMPBUF2 x=0,1	EPWMx_BA+0x324	R	EPWM CMPDAT2缓存	0x0000_0000
EPWM_CMPBUF3 x=0,1	EPWMx_BA+0x328	R	EPWM CMPDAT3缓存	0x0000_0000
EPWM_CMPBUF4 x=0,1	EPWMx_BA+0x32C	R	EPWM CMPDAT4缓存	0x0000_0000
EPWM_CMPBUF5 x=0,1	EPWMx_BA+0x330	R	EPWM CMPDAT5缓存	0x0000_0000
EPWM_CPSCBUF0_1 x=0,1	EPWMx_BA+0x334	R	EPWM CLKPSC0_1缓存	0x0000_0000
EPWM_CPSCBUF2_3 x=0,1	EPWMx_BA+0x338	R	EPWM CLKPSC2_3缓存	0x0000_0000
EPWM_CPSCBUF4_5 x=0,1	EPWMx_BA+0x33C	R	EPWM CLKPSC4_5缓存	0x0000_0000
EPWM_FTCBUF0_1 x=0,1	EPWMx_BA+0x340	R	EPWM FTCMPDAT0_1缓存	0x0000_0000
EPWM_FTCBUF2_3 x=0,1	EPWMx_BA+0x344	R	EPWM FTCMPDAT2_3缓存	0x0000_0000
EPWM_FTCBUF4_5 x=0,1	EPWMx_BA+0x348	R	EPWM FTCMPDAT4_5缓存	0x0000_0000
EPWM_FTCI x=0,1	EPWMx_BA+0x34C	R/W	EPWM FTCMPDAT 指示寄存器	0x0000_0000

6.12.7 寄存器描述

EPWM 控制寄存器 0 (EPWM_CTL0)

寄存器	偏移量	R/W	描述	复位值
EPWM_CTL0	EPWMx_BA+0x00	R/W	EPWM 控制寄存器 0	0x0000_0000

31	30	29	28	27	26	25	24
DBGTRIOFF	DBGHALT	Reserved					GROUPEN
23	22	21	20	19	18	17	16
Reserved		IMMLDEN5	IMMLDEN4	IMMLDEN3	IMMLDEN2	IMMLDEN1	IMMLDEN0
15	14	13	12	11	10	9	8
Reserved		WINLDEN5	WINLDEN4	WINLDEN3	WINLDEN2	WINLDEN1	WINLDEN0
7	6	5	4	3	2	1	0
Reserved		CTRLD5	CTRLD4	CTRLD3	CTRLD2	CTRLD1	CTRLD0

位	描述	
[31]	**DBGTRIOFF**	**ICE 调试模式禁止（写保护）** 0=ICE 调试模式影响EPWM输出 ICE 调试模式期间EPWM管脚将强制作为三态模式。 1=禁止ICE 调试模式影响EPWM输出 EPWM管脚将保持输出不管是否在ICE 调试模式 注：该寄存器写保护，详情请参考SYS_REGLCTL 寄存器
[30]	**DBGHALT**	**ICE 调试模式计数暂停（写保护）** 如果计数暂停被使能，EPWM所有计数器将保持当前值直到退出ICE调试模式。 0 =禁止ICE 调试模式计数暂停 1 = 使能ICE 调试模式计数暂停 注：该寄存器写保护，详情请参考SYS_REGLCTL 寄存器
[29:26]	**Reserved**	保留
[24]	**GROUPEN**	**组功能使能位** 0 = 每个EPWM通道输出波形相互独立 1 = EPWM_CH2 、 EPWM_CH4 与EPWM_CH0 输出相同波形， EPWM_CH3 、EPWM_CH5和EPWM_CH1输出相同波形
[23:22]	**Reserved**	保留
[16+n] n=0,1..5	**IMMLDENn**	**立即装载使能位** 每位n控制相应EPWM通道n 0 = 每个周期结束时PERIOD将载入到PBUF。通过设定CTRLD位， CMPDAT将在每个周期的终点或者中心点载入到CMPBUF。 1= 当软件更新PERIOD/CMPDAT， PERIOD/CMPDAT 将立即分别载入到PBUF 和

		CMPBUF。 注：如果IMMLDENn使能，WINLDENn 和 CTRLDn无效
[15:14]	**Reserved**	保留
[8+n] n=0,1..5	**WINLDENn**	**窗口载入使能** 每位n控制相应EPWM通道n 0 = 在每个周期终点，PERIOD 将重载到 PBUF。通过设置CTRLD位，每个周期终点或中心点CMPDAT将重载到CMPBUF 。 1 =在每个周期终点，PERIOD将重载到PBUF 。当有效重载窗口被设置以后，在每个周期终点，CMPDAT将载入到 CMPBUF。有效重载窗口可通过软件写1到EPWM_LOAD 寄存器来设置，装载成功后硬件自动清零。
[7:6]	**Reserved**	保留
[n] n=0,1..5	**CTRLDn**	**中心重载** 每位n控制相应EPWM通道n 在上-下计数方式，在每个周期终点PERIOD将载入到PBUF。每个周期中心点CMPDAT将载入到CMPBUF。

EPWM 控制寄存器 1(EPWM_CTL1)

寄存器	偏移量	R/W	描述	复位值
EPWM_CTL1	EPWMx_BA+0x04	R/W	EPWM 控制寄存器 1	0x0000_0000

31	30	29	28	27	26	25	24
Reserved					OUTMODE4	OUTMODE2	OUTMODE0
23	22	21	20	19	18	17	16
Reserved		CNTMODE5	CNTMODE4	CNTMODE3	CNTMODE2	CNTMODE1	CNTMODE0
15	14	13	12	11	10	9	8
Reserved				CNTTYPE5		CNTTYPE4	
7	6	5	4	3	2	1	0
CNTTYPE3		CNTTYPE2		CNTTYPE1		CNTTYPE0	

位	描述	
[31:27]	**Reserved**	保留
[24+n/2] n=0,2,4	**OUTMODEn**	**EPWM 输出模式** 每位n控制相应EPWM通道n 0=EPWM独立模式 1=EPWM互补模式 注：当操作在成组功能，这些位必须都设置为相同模式。
[23:22]	**Reserved**	保留
[16+n] n=0,1..5	**CNTMODEn**	**EPWM 计数器模式** 每位n控制相应EPWM通道n 0 = 自动重载模式 1 = 单周期（One-shot）模式
[15:12]	**Reserved**	保留
[2n+1:2n] n=0,1..5	**CNTTYPEn**	**EPWM计数器方式类型** 每位n控制相应EPWM通道n 00 = 向上计数类型(支持捕获模式) 01 = 向下计数模式 (支持捕获模式) 10 = 上下计数模式 11 = 保留

EPWM同步寄存器(EPWM_SYNC)

寄存器	偏移量	R/W	描述	复位值
EPWM_SYNC	EPWMx_BA+0x08	R/W	EPWM 同步寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved					PHSDIR4	PHSDIR2	PHSDIR0
23	22	21	20	19	18	17	16
SINPINV	SFLTCNT			SFLTCSEL			SNFLTEN
15	14	13	12	11	10	9	8
Reserved		SINSRC4		SINSRC2		SINSRC0	
7	6	5	4	3	2	1	0
Reserved					PHSEN4	PHSEN2	PHSEN0

位	描述	
[31:27]	**Reserved**	保留
[24+n/2] n=0,2,4	**PHSDIRn**	**EPWM 相位方向控制** 每位n控制相应EPWM通道n 0 = 控制 EPWM 计数器同步后递减计数 1 =控制 EPWM 计数器同步后递增计数
[23]	**SINPINV**	**同步输入管脚反向** 0 = 同步管脚的状态被传到负边沿侦测器 1 =同步管脚的反向状态被传到负边沿侦测器
[22:20]	**SFLTCNT**	**同步边沿侦测滤波计数** 该位控制边沿侦测的计数数目
[19:17]	**SFLTCSEL**	**同步边沿侦测滤波器时钟选择** 000 =滤波器时钟= HCLK 001 =滤波器时钟= HCLK/2 010 =滤波器时钟= HCLK/4 011 =滤波器时钟= HCLK/8 100 =滤波器时钟= HCLK/16 101 =滤波器时钟= HCLK/32 110 =滤波器时钟= HCLK/64 111 =滤波器时钟= HCLK/128
[16]	**SNFLTEN**	**EPWM0同步输入噪声滤波使能** 0 = EPWM0同步输入口噪声滤波禁止 1 = EPWM0同步输入口噪声滤波使能
[15:14]	**Reserved**	保留

[9+n:8+n] n=0,2,4	**SINSRCn**	**EPWM0同步输入源选择** 00 = 同步源来自同步输入或软件同步 01 =计数器等于0 10 =计数器等于 EPWM_CMPDATm，m 为 1、3、5 11 = 无同步输出
[7:3]	**Reserved**	保留
[n/2] n=0,2,4	**PHSENn**	**同步相位使能** 0 = EPWM计数值禁止装载 PHS 值 1 = EPWM计数值使能装载 PHS 值

EPWM软件控制同步寄存器(EPWM_SWSYNC)

寄存器	偏移量	R/W	描述	复位值
EPWM_SWSYNC	EPWMx_BA+0x0C	R/W	EPWM 软件控制同步寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved					SWSYNC4	SWSYNC2	SWSYNC0

位	描述	
[31:3]	**Reserved**	保留
[n/2] n=0,2,4	**SWSYNCn**	**软件控制同步功能** 每位n控制相应EPWM通道n 当 SINSRCn (EPWM_SYNC[13:8]) 设为 0，同步输出源来自同步输入或此位

EPWM 时钟源寄存器(EPWM_CLKSRC)

寄存器	偏移量	R/W	描述	复位值
EPWM_CLKSRC	EPWMx_BA+0x10	R/W	EPWM 时钟源寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved					ECLKSRC4		
15	14	13	12	11	10	9	8
Reserved					ECLKSRC2		
7	6	5	4	3	2	1	0
Reserved					ECLKSRC0		

位	描述	
[31:19]	**Reserved**	保留
[18:16]	**ECLKSRC4**	**EPWM_CH45外部时钟源选择** 000 = EPWMx_CLK，x 表示0 或 1 001 = TIMER0溢出 010 = TIMER1溢出 011 = TIMER2溢出 100 = TIMER3溢出 其他 = 保留
[15:11]	**Reserved**	保留
[10:8]	**ECLKSRC2**	**EPWM_CH23外部时钟源选择** 000 = EPWMx_CLK，x表示0 或 1 001 = TIMER0溢出 010 = TIMER1溢出 011 = TIMER2溢出 100 = TIMER3溢出 其他 =保留
[7:3]	**Reserved**	保留
[2:0]	**ECLKSRC0**	**EPWM_CH01外部时钟源选择** 000 = EPWMx_CLK，x表示0 或 1 001 = TIMER0溢出 010 = TIMER1溢出 011 = TIMER2溢出 100 = TIMER3溢出

		其他 =保留

EPWM 时钟预分频寄存器0_1, 2_3, 4_5(EPWM_CLKPSC0_1, 2_3, 4_5)

寄存器	偏移量	R/W	描述	复位值
EPWM_CLKPSC0_1	EPWMx_BA+0x14	R/W	EPWM 时钟预分频寄存器0/1	0x0000_0000
EPWM_CLKPSC2_3	EPWMx_BA+0x18	R/W	EPWM 时钟预分频寄存器2/3	0x0000_0000
EPWM_CLKPSC4_5	EPWMx_BA+0x1C	R/W	EPWM 时钟预分频寄存器4/5	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved				CLKPSC			
7	6	5	4	3	2	1	0
CLKPSC							

位	描述	
[31:12]	Reserved	保留
[11:0]	CLKPSC	EPWM计数器时钟预分频 EPWM计数器时钟由时钟预分频器决定，每个EPWM组共享一个EPWM寄存器时钟预分频器，EPWM计数器时钟源被(CLKPSC+ 1)除频。

EPWM计数使能寄存器(EPWM_CNTEN)

寄存器	偏移量	R/W	描述	复位值
EPWM_CNTEN	EPWMx_BA+0x20	R/W	EPWM 计数使能寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved		CNTEN5	CNTEN4	CNTEN3	CNTEN2	CNTEN1	CNTEN0

位	描述	
[31:6]	Reserved	保留
[n] n=0,1..5	CNTENn	EPWM计数使能位 每位n控制相应EPWM通道n 0 = EPWM计数器和时钟分频器停止工作 1 = EPWM计数器和时钟分频器开始工作

EPWM 清计数器寄存器(EPWM_CNTCLR)

寄存器	偏移量	R/W	描述	复位值
EPWM_CNTCLR	EPWMx_BA+0x24	R/W	EPWM 清计数器寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved		CNTCLR5	CNTCLR4	CNTCLR3	CNTCLR2	CNTCLR1	CNTCLR0

位	描述	
[31:6]	Reserved	保留
[n] n=0,1..5	CNTCLRn	清 EPWM计数器控制位 此位由硬件自动清零，每位n控制相应EPWM通道n。 0 = 无影响 1 =清16位EPWM计数器到0000H

EPWM装载寄存器(EPWM_LOAD)

寄存器	偏移量	R/W	描述	复位值
EPWM_LOAD	EPWMx_BA+0x28	R/W	EPWM 装载寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved		**LOAD5**	**LOAD4**	**LOAD3**	**LOAD2**	**LOAD1**	**LOAD0**

位	描述	
[31:6]	**Reserved**	保留
[n] n=0,1..5	**LOADn**	**重载EPWM 比较值寄存器 (CMPDAT) 控制位** 该位为软件置位，当当前EPWM周期结束时硬件清零，每位n控制相应EPWM通道n。 写操作： 0 = 无影响 1 = 设置窗口装载模式的装载窗口. 读操作： 0 = 没有装载窗口设置 1 = 装载窗口被设置 **注：** 该位只在窗口装载模式 WINLDENn(EPWM_CTL0[13:8]) = 1下使用。

EPWM周期寄存器0~5 (EPWM_PERIOD0~5)

寄存器	偏移	R/W	描述	复位值
EPWM_PERIOD0	EPWMx_BA+0x30	R/W	EPWM 周期寄存器0	0x0000_0000
EPWM_PERIOD1	EPWMx_BA+0x34	R/W	EPWM 周期寄存器1	0x0000_0000
EPWM_PERIOD2	EPWMx_BA+0x38	R/W	EPWM 周期寄存器2	0x0000_0000
EPWM_PERIOD3	EPWMx_BA+0x3C	R/W	EPWM 周期寄存器3	0x0000_0000
EPWM_PERIOD4	EPWMx_BA+0x40	R/W	EPWM 周期寄存器4	0x0000_0000
EPWM_PERIOD5	EPWMx_BA+0x44	R/W	EPWM 周期寄存器5	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
PERIOD							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
PERIOD							

位	描述	
[31:16]	**Reserved**	保留
[15:0]	**PERIOD**	**EPWM 周期寄存器** 向上计数模式： 此模式下，EPWM计数器从0计数到PERIOD，并从0重新开始计数。 EPWM周期 = (PERIOD+1) * (CLKPSC+1) * EPWM_CLK 向下计数模式： 此模式下，EPWM计数器从PERIOD计数到0，并从PERIOD重新开始计数。 EPWM周期 = (PERIOD+1) * (CLKPSC+1) * EPWM_CLK 上下计数模式： 这个模式EPWM计数器从0计数到PERIOD然后递减到0，并重复。 EPWM周期 = 2 * PERIOD * (CLKPSC+1) * EPWM_CLK

EPWM比较寄存器0~5 (EPWM_CMPDAT0~5)

寄存器	偏移	R/W	描述	复位值
EPWM_CMPDAT0	EPWMx_BA+0x50	R/W	EPWM比较寄存器0	0x0000_0000
EPWM_CMPDAT1	EPWMx_BA+0x54	R/W	EPWM比较寄存器1	0x0000_0000
EPWM_CMPDAT2	EPWMx_BA+0x58	R/W	EPWM比较寄存器2	0x0000_0000
EPWM_CMPDAT3	EPWMx_BA+0x5C	R/W	EPWM比较寄存器3	0x0000_0000
EPWM_CMPDAT4	EPWMx_BA+0x60	R/W	EPWM比较寄存器4	0x0000_0000
EPWM_CMPDAT5	EPWMx_BA+0x64	R/W	EPWM比较寄存器5	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
CMP							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
CMP							

位	描述	
[31:16]	**Reserved**	保留
[15:0]	**CMP**	**EPWM 比较寄存器** CMP用于与CNT (EPWM_CNTn[15:0])比较来产生EPWM波形，中断和触发EADC/DAC. 独立模式，CMPDAT0~5作为6个独立EPWM_CH0~5比较点。 互补模式， CMPDAT0, 2, 4作为第一比较点而CMPDAT1, 3, 5作为第二比较点对应于相应的三个互补组EPWM_CH0 和 EPWM_CH1, EPWM_CH2 和 EPWM_CH3, EPWM_CH4 和 EPWM_CH5。

EPWM死区控制寄存器0_1, 2_3, 4_5(EPWM_DTCTL0_1, 2_3, 4_5)

寄存器	偏移量	R/W	描述	复位值
EPWM_DTCTL0_1	EPWMx_BA+0x70	R/W	EPWM死区控制寄存器0/1	0x0000_0000
EPWM_DTCTL2_3	EPWMx_BA+0x74	R/W	EPWM死区控制寄存器2/3	0x0000_0000
EPWM_DTCTL4_5	EPWMx_BA+0x78	R/W	EPWM死区控制寄存器4/5	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							DTCKSEL
23	22	21	20	19	18	17	16
Reserved							DTEN
15	14	13	12	11	10	9	8
Reserved				DTCNT			
7	6	5	4	3	2	1	0
DTCNT							

位	描述	
[31:25]	**Reserved**	保留
[24]	**DTCKSEL**	**死区时钟源选择 (写保护)** 0 = 死区时钟源来自 EPWM_CLK 1 =死区时钟源来自预分频器输出 **注：**该位为写保护位，详情请参考SYS_REGLCTL寄存器。
[23:17]	**Reserved**	保留
[16]	**DTEN**	**使能EPWM组(EPWM_CH0, EPWM_CH1) (EPWM_CH2, EPWM_CH3) (EPWM_CH4, EPWM_CH5) 死区插入(写保护)** 死区插入只有当该组互补EPWM使能才激活。如果死区插入未激活，该互补组输出管脚没有任何延时。 0 = 禁止管脚组死区插入。 1 = 使能管脚组死区插入。 **注：**该位为写保护位，详情请参考SYS_REGLCTL寄存器。
[15:12]	**Reserved**	保留
[11:0]	**DTCNT**	**死区计数器（写保护）** 死区时间可以根据以下公式计算： 死区时间=(DTCNT[11:0]+1) * EPWM_CLK 周期 注：该寄存器写保护，参考REGWRPROT寄存器 **注：**该位为写保护位，详情请参考SYS_REGLCTL寄存器。

EPWM计数器相位寄存器0_1, 2_3, 4_5 (EPWM_PHS0_1, 2_3, 4_5)

寄存器	偏移量	R/W	描述	复位值
EPWM_PHS0_1	EPWMx_BA+0x80	R/W	EPWM 计数器相位寄存器0/1	0x0000_0000
EPWM_PHS2_3	EPWMx_BA+0x84	R/W	EPWM 计数器相位寄存器2/3	0x0000_0000
EPWM_PHS4_5	EPWMx_BA+0x88	R/W	EPWM 计数器相位寄存器4/5	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
PHS							
7	6	5	4	3	2	1	0
PHS							

位	描述	
[31:16]	**Reserved**	保留
[15:0]	**PHS**	**EPWM 同步开始相位** PHS决定EPWM 同步开始的相位值，该位用于同步功能。

EPWM 计数寄存器0~5(EPWM_CNT0~5)

寄存器	偏移量	R/W	描述	复位值
EPWM_CNT0	EPWMx_BA+0x90	R	EPWM计数寄存器0	0x0000_0000
EPWM_CNT1	EPWMx_BA+0x94	R	EPWM计数寄存器1	0x0000_0000
EPWM_CNT2	EPWMx_BA+0x98	R	EPWM计数寄存器2	0x0000_0000
EPWM_CNT3	EPWMx_BA+0x9C	R	EPWM计数寄存器3	0x0000_0000
EPWM_CNT4	EPWMx_BA+0xA0	R	EPWM计数寄存器4	0x0000_0000
EPWM_CNT5	EPWMx_BA+0xA4	R	EPWM计数寄存器5	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							DIRF
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
CNT							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
CNT							

位	描述	
[31:17]	**Reserved**	保留
[16]	**DIRF**	**EPWM方向标志(只读)** 0 =向下计数 1 =向上计数
[15:0]	**CNT**	**EPWM数据寄存器 (只读)** 用户可以读CNT以知道16位周期计数器当前值。

EPWM产生寄存器0(EPWM_WGCTL0)

寄存器	偏移	R/W	描述	复位值
EPWM_WGCTL0	EPWMx_BA+0xB0	R/W	EPWM 产生寄存器0	0x0000_0000

31	30	29	28	27	26	25	24
Reserved				PRDPCTL5		PRDPCTL4	
23	22	21	20	19	18	17	16
PRDPCTL3		PRDPCTL2		PRDPCTL1		PRDPCTL0	
15	14	13	12	11	10	9	8
Reserved				ZPCTL5		ZPCTL4	
7	6	5	4	3	2	1	0
ZPCTL3		ZPCTL2		ZPCTL1		ZPCTL0	

位	描述	
[31:28]	**Reserved**	保留
[17+2n:16+2n] n=0,1..5	**PRDPCTLn**	**EPWM 周期（中心）点控制** 当EPWM计数器计数到(PERIODn+1)，可以控制EPWM输出电平 00 =不变 01 = EPWM周期（中心）点输出低 10 = EPWM周期（中心）点输出高 11 = EPWM周期（中心）点输出翻转 **注：**当EPWM寄存器工作在上下计数方式，该位是中心点控制。
[15:12]	**Reserved**	保留
[1+2n:2n] n=0,1..5	**ZPCTLn**	**EPWM零点控制** 当EPWM计数器计数到零点可以控制EPWM输出电平 00 =不变 01 = EPWM零点输出低 10 = EPWM零点输出高 11 = EPWM零点输出翻转

EPWM产生寄存器1 (EPWM_WGCTL1)

寄存器	偏移	R/W	描述	复位值
EPWM_WGCTL1	EPWMx_BA+0xB4	R/W	EPWM 产生寄存器1	0x0000_0000

31	30	29	28	27	26	25	24
Reserved				CMPDCTL5		CMPDCTL4	
23	22	21	20	19	18	17	16
CMPDCTL3		CMPDCTL2		CMPDCTL1		CMPDCTL0	
15	14	13	12	11	10	9	8
Reserved				CMPUCTL5		CMPUCTL4	
7	6	5	4	3	2	1	0
CMPUCTL3		CMPUCTL2		CMPUCTL1		CMPUCTL0	

位	描述	
[31:28]	**Reserved**	保留
[17+2n:16+2n] n=0,1..5	**CMPDCTLn**	**EPWM向下比较点控制** 00 =不变 01 = EPWM向下比较点输出低 10 = EPWM向下比较点输出高 11 =向下比较点输出翻转 当寄存器向下计数到CMPDAT,可控制EPWM输出电平 **注**：在互补模式，CMPDCTL 1,3,5作为通道0, 2, 4.的另一个CMPDCTL。
[15:12]	**Reserved**	保留
[1+2n:2n] n=0,1..5	**CMPUCTLn**	**EPWM向上比较点控制** 每位n控制相应EPWM通道n 00 =不变 01 = EPWM 向上比较点输出低 10 = EPWM 向上比较点输出高 11 = EPWM向上比较点输出翻转 当寄存器向上计数到CMPDAT,EPWM可控制输出电平 **注**：在互补模式，CMPUCTL11,3,5作为通道0, 2, 4的另一个CMPUCTL。

EPWM屏蔽使能寄存器(EPWM_MSKEN)

寄存器	偏移量	R/W	描述	复位值
EPWM_MSKEN	EPWMx_BA+0xB8	R/W	EPWM 屏蔽使能寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved		MSKEN5	MSKEN4	MSKEN3	MSKEN2	MSKEN1	MSKEN0

位	描述	
[31:6]	Reserved	保留
[n] n=0,1..5	MSKENn	**EPWM 屏蔽使能寄存器** 每位n控制相应EPWM通道n 当该位使能，EPWM输出信号将被屏蔽，相应的EPWM通道n将输出MSKDATn(EPWM_MSK[5:0])数据 0 =输出信号不屏蔽 1 = EPWM输出信号被屏蔽并输出MSKDATn数据

EPWM 屏蔽数据寄存器(EPWM_MSK)

寄存器	偏移量	R/W	描述	复位值
EPWM_MSK	EPWMx_BA+0xBC	R/W	EPWM 屏蔽数据寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved		**MSKDAT5**	**MSKDAT4**	**MSKDAT3**	**MSKDAT2**	**MSKDAT1**	**MSKDAT0**

位	描述	
[31:6]	**Reserved**	保留
[n] n=0,1..5	**MSKDATn**	**EPWM 屏蔽数据位** 如果相应屏蔽功能使能，该数据位控制EPWMn输出管脚状态。每位n控制相应EPWM通道n 0 = 输出逻辑低到EPWM通道n 1 = 输出逻辑高到EPWM通道n

EPWM 刹车噪音滤波寄存器(EPWM_BNF)

寄存器	偏移	R/W	描述	复位值
EPWM_BNF	EPWMx_BA+0xC0	R/W	EPWM 刹车噪音滤波寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							BK1SRC
23	22	21	20	19	18	17	16
Reserved							BK0SRC
15	14	13	12	11	10	9	8
BRK1PINV	BRK1FCNT			BRK1NFSEL			BRK1NFEN
7	6	5	4	3	2	1	0
BRK0PINV	BRK0FCNT			BRK0NFSEL			BRK0NFEN

位	描述	
[31:25]	**Reserved**	保留
[24]	**BK1SRC**	**刹车 1 管脚源选择** 对于 EPWM0 的设置： 0 = 刹车 1 管脚源来自EPWM0_BRAKE1 1 = 刹车 1 管脚源来自EPWM1_BRAKE1 对于 EPWM1 的设置： 0 =刹车 1 管脚源来自EPWM1_BRAKE1 1 =刹车 1 管脚源来自EPWM0_BRAKE1
[23:17]	**Reserved**	保留
[16]	**BK0SRC**	**刹车 0 管脚源选择** 对于 EPWM0 的设置： 0 = 刹车 0 管脚源来自EPWM0_BRAKE0 1 = 刹车 0 管脚源来自EPWM1_BRAKE0 对于 EPWM1 的设置： 0 =刹车 0 管脚源来自EPWM1_BRAKE0 1 =刹车 0 管脚源来自EPWM0_BRAKE0
[15]	**BRK1PINV**	**刹车1引脚反向** 0 = 如果EPWMx BRAKE1引脚状态在边沿检测模式从低到高转变或在电平检测模式为高电平，刹车事件将被检测到 1 =如果EPWMx BRAKE1引脚状态在边沿检测模式从高到低转变或在电平检测模式为低电平，刹车事件将被检测到
[14:12]	**BRK1FCNT**	**刹车1边沿检测滤波器计数** 该位控制刹车1滤波计数器从0到BRK1FCNT计数

[11:9]	**BRK1NFSEL**	**刹车1边沿检测滤波器时钟选择** 000 =滤波器时钟= HCLK 001 =滤波器时钟 = HCLK/2 010 =滤波器时钟= HCLK/4 011 =滤波器时钟= HCLK/8 100 =滤波器时钟= HCLK/16 101 =滤波器时钟= HCLK/32 110 =滤波器时钟= HCLK/64 111 =滤波器时钟= HCLK/128
[8]	**BRK1NFEN**	**刹车1噪音滤波器使能** 0 = 禁止EPWM刹车1噪音滤波器 1 = 使能EPWM刹车1噪音滤波器
[7]	**BRK0PINV**	**刹车0引脚反向** 0 = 如果EPWMx BRAKE0引脚状态在边沿检测模式从低到高转变或在电平检测模式为高电平，刹车事件将被检测到 1 =如果EPWMx BRAKE0引脚状态在边沿检测模式从高到低转变或在电平检测模式为低电平，刹车事件将被检测到
[6:4]	**BRK0FCNT**	**刹车0边沿检测滤波器计数** 寄存器位控制刹车0滤波计数器从0到BRK0FCNT计数
[3:1]	**BRK0NFSEL**	**刹车0边沿检测滤波器时钟选择** 000 =滤波器时钟= HCLK 001 =滤波器时钟= HCLK/2 010 =滤波器时钟= HCLK/4 011 =滤波器时钟= HCLK/8 100 =滤波器时钟= HCLK/16 101 =滤波器时钟= HCLK/32 110 =滤波器时钟= HCLK/64 111 =滤波器时钟= HCLK/128
[0]	**BRK0NFEN**	**EPWM刹车0噪音滤波器使能** 0 = 禁止EPWM刹车0噪音滤波器 1 = 使能EPWM刹车0噪音滤波器

EPWM系统故障刹车控制寄存器(EPWM_FAILBRK)

寄存器	偏移量	R/W	描述	复位值
EPWM_FAILBRK	EPWMx_BA+0xC4	R/W	EPWM 系统故障刹车控制寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved				CORBRKEN	RAMBRKEN	BODBRKEN	CSSBRKEN

位	描述	
[31:4]	Reserved	保留
[3]	CORBRKEN	**内核锁死检测触发EPWM刹车功能0使能** 0 =禁止通过内核锁死检测触发刹车功能 1 =使能通过内核锁死检测触发刹车功能
[2]	RAMBRKEN	**SRAM 奇偶校验错误检查触发EPWM 刹车功能0 使能** 0 = SRAM奇偶校验错误检查触发刹车功能禁止 1 = SRAM奇偶校验错误检查触发刹车功能使能
[1]	BODBRKEN	**欠压检测触发EPWM刹车功能0使能** 0 =禁止BOD触发刹车功能 1 =使能BOD触发刹车功能
[0]	CSSBRKEN	**时钟安全系统检测触发EPWM刹车功能0使能** 0 =禁止通过CSS检测触发刹车功能 1 =使能通过CSS检测触发刹车功能

EPWM刹车边沿检测控制寄存器0_1, 2_3, 4_5(EPWM_BRKCTL0_1, 2_3, 4_5)

寄存器	偏移量	R/W	描述	复位值
EPWM_BRKCTL0_1	EPWMx_BA+0xC8	R/W	EPWM刹车边沿检测控制寄存器0/1	0x0000_0000
EPWM_BRKCTL2_3	EPWMx_BA+0xCC	R/W	EPWM刹车边沿检测控制寄存器2/3	0x0000_0000
EPWM_BRKCTL4_5	EPWMx_BA+0xD0	R/W	EPWM刹车边沿检测控制寄存器4/5	0x0000_0000

31	30	29	28	27	26	25	24
Reserved		EADC1LBEN	EADC0LBEN	Reserved			
23	22	21	20	19	18	17	16
Reserved		EADC1EBEN	EADC0EBEN	BRKAODD		BRKAEVEN	
15	14	13	12	11	10	9	8
SYSLBEN	Reserved	BRKP1LEN	BRKP0LEN	Reserved		CPO1LBEN	CPO0LBEN
7	6	5	4	3	2	1	0
SYSEBEN	Reserved	BRKP1EEN	BRKP0EEN	Reserved		CPO1EBEN	CPO0EBEN

位	描述	
[31:30]	**Reserved**	保留
[29]	**EADC1LBEN**	**使能EADC1结果检测(EADC1RM)作为电平检测刹车源（写保护）** 0 = EADC1RM作为电平检测刹车源禁用 1 = EADC1RM作为电平检测刹车源使能 **注：**该位写保护，参考SYS_REGLCTL寄存器。
[28]	**EADC0LBEN**	**使能EADC0结果检测 (EADC0RM)作为电平检测刹车源（写保护）** 0 = EADC0RM作为电平检测刹车源禁用 1 = EADC0RM作为电平检测刹车源使能 **注：**该位写保护，参考SYS_REGLCTL寄存器。
[27:22]	**Reserved**	保留
[21]	**EADC1EBEN**	**使能EADC1结果检测 (EADC1RM)作为边沿检测刹车源（写保护）** 0 = EADC1RM作为边沿检测刹车源禁用 1 = EADC1RM作为边沿检测刹车源使能 **注：**该位写保护，参考SYS_REGLCTL寄存器。
[20]	**EADC0EBEN**	**使能EADC0结果检测 (EADC0RM)作为边沿检测刹车源（写保护）** 0 = EADC0RM作为边沿检测刹车源禁用 1 = EADC0RM作为边沿检测刹车源使能 **注：**该位写保护，参考SYS_REGLCTL寄存器。

[19:18]	**BRKAODD**	**EPWM奇数通道刹车行为选择（写保护）** 00 = EPWMx的刹车事件不影响奇数通道输出 01 = 当EPWMx的刹车事件发生，EPWM奇数通道输出三态 10 =当EPWMx的刹车事件发生，EPWM奇数通道输出低电平 11 =当EPWMx的刹车事件发生，EPWM奇数通道输出高电平 **注**：该寄存器写保护，参考SYS_REGLCTL寄存器。
[17:16]	**BRKAEVEN**	**EPWM偶数通道刹车行为选择（写保护）** 00 = EPWMx的刹车事件不影响偶数通道输出 01 = 当EPWMx的刹车事件发生，EPWM偶数通道输出三态 10 =当EPWMx的刹车事件发生，EPWM偶数通道输出低电平 11 =当EPWMx的刹车事件发生，EPWM偶数通道输出高电平 **注**：该寄存器写保护，参考SYS_REGLCTL寄存器。
[15]	**SYSLBEN**	**使能系统故障作为电平检测刹车源（写保护）** 0 =禁止系统故障条件作为电平检测刹车源 1 =使能系统故障条件作为电平检测刹车源 **注**：该寄存器写保护，参考SYS_REGLCTL寄存器。
[14]	**Reserved**	保留
[13]	**BRKP1LEN**	**使能管脚BKP1作为电平检测刹车源（写保护）** 0 =.禁止EPWMx_BRAKE1管脚作为电平检测刹车源 1 =使能EPWMx_BRAKE1管脚作为电平检测刹车源 注：该寄存器写保护，参考SYS_REGLCTL寄存器。
[12]	**BRKP0LEN**	**使能管脚BKP0作为电平检测刹车源（写保护）** 0 = 禁止EPWMx_BRAKE0管脚作为电平检测刹车源 1 = 使能EPWMx_BRAKE0管脚作为电平检测刹车源 注：该寄存器写保护，参考SYS_REGLCTL寄存器。
[11:10]	**Reserved**	保留
[9]	**CPO1LBEN**	**使能 ACMP1_O数字输出作为电平检测刹车源（写保护）** 0 = ACMP1_O 作为电平检测刹车源禁止 1 = ACMP1_O作为电平检测刹车源使能 注：该寄存器写保护，参考SYS_REGLCTL寄存器。
[8]	**CPO0LBEN**	**使能ACMP0_O数字输出作为电平检测刹车源（写保护）** 0 = ACMP0_O作为电平检测刹车源禁止 1 = ACMP0_O作为电平检测刹车源使能 注：该寄存器写保护，参考SYS_REGLCTL寄存器。
[7]	**SYSEBEN**	**使能系统故障作为边沿检测刹车源（写保护）** 0 =系统故障条件作为沿检测刹车源禁止 1 = 系统故障条件作为沿检测刹车源使能 注：该寄存器写保护，参考SYS_REGLCTL寄存器。
[6]	**Reserved**	保留
[5]	**BRKP1EEN**	**使能EPWMx_BRAKE1 引脚作为边沿检测刹车源（写保护）**

		0 = EPWMx_BRAKE1管脚作为边沿检测刹车源禁止 1 = EPWMx_BRAKE1管脚作为边沿检测刹车源使能 注：该寄存器写保护，参考SYS_REGLCTL寄存器。
[4]	**BRKP0EEN**	**使能EPWMx_BRAKE0引脚作为边沿检测刹车源（写保护）** 0 = EPWMx_BRAKE0管脚作为边沿检测刹车源禁止 1 = EPWMx_BRAKE0管脚作为边沿检测刹车源使能 注：该寄存器写保护，参考SYS_REGLCTL寄存器。
[3:2]	**Reserved**	保留
[1]	**CPO1EBEN**	**使能ACMP1_O 数字输出作为边沿检测刹车源（写保护）** 0 = ACMP1_O 作为边沿检测刹车源禁止 1 = ACMP1_O 作为边沿检测刹车源使能 注：该寄存器写保护，参考SYS_REGLCTL寄存器。
[0]	**CPO0EBEN**	**使能ACMP0_O数字输出作为边沿检测刹车源（写保护** 0 = ACMP0_O 作为边沿检测刹车源禁止 1 = ACMP0_O 作为边沿检测刹车源使能 注：该寄存器写保护，参考SYS_REGLCTL寄存器。

EPWM 管脚极性反转控制(EPWM_POLCTL)

寄存器	偏移量	R/W	描述	复位值
EPWM_POLCTL	EPWMx_BA+0xD4	R/W	EPWM 管脚极性反转寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved		PINV5	PINV4	PINV3	PINV2	PINV1	PINV0

位	描述	
[31:6]	**Reserved**	保留
[n] n=0,1..5	**PINVn**	**EPWM引脚极性反相控制** 该寄存器控制EPWMx_CHn输出引脚极性状态 0 = EPWMx_CHn输出引脚极性反相禁用 1 = EPWMx_CHn输出引脚极性反相使能

EPWM输出使能寄存器(EPWM_POEN)

寄存器	偏移量	R/W	描述	复位值
EPWM_POEN	EPWMx_BA+0xD8	R/W	EPWM 输出使能寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved		**POEN5**	**POEN4**	**POEN3**	**POEN2**	**POEN1**	**POEN0**

位	描述	
[31:6]	**Reserved**	保留
[n] n=0,1..5	**POENn**	**EPWM引脚输出使能位** 0 = EPWMx_CHn引脚为三态模式 1 = EPWMx_CHn引脚为输出模式

EPWM软件刹车控制寄存器(EPWM_SWBRK)

寄存器	偏移量	R/W	描述	复位值
EPWM_SWBRK	EPWMx_BA+0xDC	W	EPWM 软件刹车控制寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved					BRKLTRG4	BRKLTRG2	BRKLTRG0
7	6	5	4	3	2	1	0
Reserved					BRKETRG4	BRKETRG2	BRKETRG0

位	描述	
[31:11]	**Reserved**	保留
[8+n/2] n=0,2,4	**BRKLTRGn**	**EPWM电平刹车软件触发（只写）（写保护）** 写1到该位将触发电平刹车，并将EPWM_INTSTS1寄存器的BRKLIFn置1 注：该寄存器写保护，参考SYS_REGLCTL寄存器。
[7:3]	**Reserved**	保留
[n/2] n=0,2,4	**BRKETRGn**	**EPWM边沿刹车软件触发（只写）（写保护）** 写1到该位将触发边沿刹车，并将EPWM_INTSTS1寄存器的BRKEIFn置1 注：该寄存器写保护，参考SYS_REGLCTL寄存器。

EPWM 中断使能寄存器 0(EPWM_INTEN0)

寄存器	偏移量	R/W	描述	复位值
EPWM_INTEN0	EPWMx_BA+0xE0	R/W	EPWM 中断使能寄存器 0	0x0000_0000

31	30	29	28	27	26	25	24
Reserved		CMPDIEN5	CMPDIEN4	CMPDIEN3	CMPDIEN2	CMPDIEN1	CMPDIEN0
23	22	21	20	19	18	17	16
Reserved		CMPUIEN5	CMPUIEN4	CMPUIEN3	CMPUIEN2	CMPUIEN1	CMPUIEN0
15	14	13	12	11	10	9	8
Reserved		PIEN5	PIEN4	PIEN3	PIEN2	PIEN1	PIEN0
7	6	5	4	3	2	1	0
Reserved		ZIEN5	ZIEN4	ZIEN3	ZIEN2	ZIEN1	ZIEN0

位	描述	
[31:30]	**Reserved**	保留
[24+n] n=0,1..5	**CMPDIENn**	**EPWM比较向下计数中断使能** 0 =禁止比较向下计数中断 1 =使能比较向下计数中断 注：在互补模式，CMPDIEN 1,3,5没有用，作为通道 0, 2, 4.的另一个CMPDIEN。
[23:22]	**Reserved**	保留
[16+n] n=0,1..5	**CMPUIENn**	**EPWM比较向上计数中断使能** 0 =禁止比较向上计数中断 1 =使能比较向上计数中断 注：在互补模式，CMPUIEN 1,3,5没有用，作为通道 0, 2, 4.的另一个CMPUIEN。
[15:14]	**Reserved**	保留
[8+n] n=0,1..5	**PIENn**	**EPWM 周期点中断使能** 每位n控制相应EPWM通道n 0 =禁止周期点中断 1 =使能周期点中断 注1：上下计数方式周期点指的是中心点； 注2：在互补模式，奇数通道读数总是0。
[7:6]	**Reserved**	保留
[n] n=0,1..5	**ZIENn**	**EPWM 0点中断使能** 0 = 0点中断使能 1 = 0点中断使能 **注:**在互补模式奇数通道读数总是0。

EPWM中断使能寄存器1(EPWM_INTEN1)

寄存器	偏移量	R/W	描述	复位值
EPWM_INTEN1	EPWMx_BA+0xE4	R/W	EPWM 中断使能寄存器 1	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved					BRKLIEN4_5	BRKLIEN2_3	BRKLIEN0_1
7	6	5	4	3	2	1	0
Reserved					BRKEIEN4_5	BRKEIEN2_3	BRKEIEN0_1

位	描述	
[31:11]	**Reserved**	保留
[10]	**BRKLIEN4_5**	**EPWM通道4/5电平检测刹车中断使能（写保护）** 0 =禁止通道4/5电平检测刹车中断 1 =使能通道4/5电平检测刹车中断 注：该寄存器写保护，参考SYS_REGLCTL寄存器。
[9]	**BRKLIEN2_3**	**EPWM通道2/3电平检测刹车中断使能（写保护）** 0 =禁止通道2/3电平检测刹车中断 1 =使能通道2/3电平检测刹车中断 注：该寄存器写保护，参考 SYS_REGLCTL寄存器。
[8]	**BRKLIEN0_1**	**EPWM通道0/1电平检测刹车中断使能（写保护）** 0 =禁止通道0/1电平检测刹车中断 1 =使能通道0/1电平检测刹车中断 注：该寄存器写保护，参考SYS_REGLCTL寄存器。
[7:3]	**Reserved**	保留
[2]	**BRKEIEN4_5**	**EPWM通道4/5边沿检测刹车中断使能（写保护）** 0 =禁止通道4/5边沿检测刹车中断 1 =使能通道4/5边沿检测刹车中断 注：该寄存器写保护，参考SYS_REGLCTL寄存器。
[1]	**BRKEIEN2_3**	**EPWM通道2/3边沿检测刹车中断使能（写保护）** 0 =禁止通道2/3边沿检测刹车中断 1 =使能通道2/3边沿检测刹车中断 注：该寄存器写保护，参考SYS_REGLCTL寄存器

[0]	**BRKEIEN0_1**	**EPWM通道0/1边沿检测刹车中断使能（写保护）** 0 = 禁止通道0/1边沿检测刹车中断 1 = 使能通道0/1边沿检测刹车中断 注：该寄存器写保护，参考SYS_REGLCTL寄存器。

EPWM中断标志寄存器0 (EPWM_INTSTS0)

寄存器	偏移量	R/W	描述	复位值
EPWM_INTSTS0	EPWMx_BA+0xE8	R/W	EPWM 中断标志寄存器 0	0x0000_0000

31	30	29	28	27	26	25	24
Reserved		CMPDIF5	CMPDIF4	CMPDIF3	CMPDIF2	CMPDIF1	CMPDIF0
23	22	21	20	19	18	17	16
Reserved		CMPUIF5	CMPUIF4	CMPUIF3	CMPUIF2	CMPUIF1	CMPUIF0
15	14	13	12	11	10	9	8
Reserved		PIF5	PIF4	PIF3	PIF2	PIF1	PIF0
7	6	5	4	3	2	1	0
Reserved		ZIF5	ZIF4	ZIF3	ZIF2	ZIF1	ZIF0

位	描述	
[31:30]	**Reserved**	保留
[24+n] n=0,1..5	**CMPDIFn**	**EPWM比较向下计数中断标志** 当EPWM计数器向下计数到EPWM_CMPDATn，标志被硬件置1。软件写1到该位将清标志。 注：在互补模式，CMPDIF 1,3,5作为通道 0, 2, 4.的另一个CMPDIF。
[23:22]	**Reserved**	保留
[16+n] n=0,1..5	**CMPUIFn**	**EPWM比较向上计数中断标志** 当EPWM计数器向上计数到EPWM_CMPDATn，标志被硬件置1。软件写1到该位将清标志。 注：在互补模式，CMPDIF 1,3,5作为通道 0, 2, 4.的另一个CMPUIF。
[15:14]	**Reserved**	保留
[8+n] n=0,1..5	**PIFn**	**EPWM 周期点中断标志** 当EPWM计数值达到EPWM_PERIODn，该位被硬件置1，软件写1到该位将清标志。
[7:6]	**Reserved**	保留
[n] n=0,1..5	**ZIFn**	**EPWM 零点中断标志** 当EPWM计数器减到0该位被硬件置1，硬件对该位置1，软件写1到该位将清标志

EPWM中断标志寄存器1 (EPWM_INTSTS1)

寄存器	偏移	R/W	描述	复位值
EPWM_INTSTS1	EPWMx_BA+0xEC	R/W	EPWM中断标志寄存器1	0x0000_0000

31	30	29	28	27	26	25	24
Reserved		BRKLSTS5	BRKLSTS4	BRKLSTS3	BRKLSTS2	BRKLSTS1	BRKLSTS0
23	22	21	20	19	18	17	16
Reserved		BRKESTS5	BRKESTS4	BRKESTS3	BRKESTS2	BRKESTS1	BRKESTS0
15	14	13	12	11	10	9	8
Reserved		BRKLIF5	BRKLIF4	BRKLIF3	BRKLIF2	BRKLIF1	BRKLIF0
7	6	5	4	3	2	1	0
Reserved		BRKEIF5	BRKEIF4	BRKEIF3	BRKEIF2	BRKEIF1	BRKEIF0

位	描述	
[31:30]	**Reserved**	保留
[24+n] n=0,1..5	**BRKLSTSn**	**EPWM通道n电平检测刹车状态(只读)** 0 = EPWM通道n电平检测刹车状态释放 1 = 当EPWM通道n电平检测刹车检测到任何使能刹车源的下降沿，该标志将置位，表示EPWM通道n在刹车状态 **注：**该位只读且由硬件自动清零。当使能的刹车源返回到高电平，EPWM将释放刹车状态直到当前EPWM周期完成，EPWM波形将从下一个完整EPWM周期开始输出。
[23:22]	**Reserved**	保留
[16+n] n=0,1..5	**BRKESTSn**	**EPWM通道n边沿检测刹车状态(只读)** 0 = EPWM通道n边沿检测刹车状态释放 1 = 当EPWM通道n边沿检测刹车检测到任何使能刹车源的下降沿，该标志将置位，表示EPWM通道n在刹车状态 **注：**该位只读且由硬件自动清零。当边沿检测刹车中断标志清除，EPWM将释放刹车状态直到当前EPWM周期完成，EPWM波形将从下一个完整EPWM周期开始输出。
[15:14]	**Reserved**	保留
[8+n] n=0,1..5	**BRKLIFn**	**EPWM通道n电平检测刹车中断标志（写保护）** 0 = EPWM通道n电平检测刹车事件没发生 1 = 当EPWM通道n电平检测刹车事件发生，该位置1，写1清0。 **注：**该位写保护，参考SYS_REGLCTL寄存器。
[7:6]	**Reserved**	保留
[n] n=0,1..5	**BRKEIFn**	**EPWM通道n边沿检测刹车中断标志（写保护）** 0 = EPWM通道n边沿检测刹车事件没发生 1 = 当EPWM通道n边沿检测刹车事件发生，该位置1，写1清0。

		注：该位写保护，参考SYS_REGLCTL寄存器。

EPWM触发 DAC使能寄存器(EPWM_DACTRGEN)

寄存器	偏移量	R/W	描述	复位值
EPWM_DACTRGEN	EPWMx_BA+0xF4	R/W	EPWM 触发 DAC 使能寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved		CDTRGEN5	CDTRGEN4	CDTRGEN3	CDTRGEN2	CDTRGEN1	CDTRGEN0
23	22	21	20	19	18	17	16
Reserved		CUTRGEN5	CUTRGEN4	CUTRGEN3	CUTRGEN2	CUTRGEN1	CUTRGEN0
15	14	13	12	11	10	9	8
Reserved		PTE5	PTE4	PTE3	PTE2	PTE1	PTE0
7	6	5	4	3	2	1	0
Reserved		ZTE5	ZTE4	ZTE3	ZTE2	ZTE1	ZTE0

位	描述	
[31:30]	**Reserved**	保留
[24+n] n=0,1..5	**CDTRGEn**	**EPWM 比较器向下计数点触发DAC 使能** 如果该位置1，当EPWM计数器向下计数到CMPDAT ， EPWM 可以触发DAC启动。 0 = EPWM比较器向下计数点触发DAC功能禁止 1 = EPWM 比较器向下计数点触发 DAC 功能使能 **注1：**当EPWM 计数器工作在向上计数模式.，该位应该保持 0 **注2：**在互补模式下， CDTRGE1, 3, 5作为通道 0, 2, 4.的另一个CDTRGE
[23:22]	**Reserved**	保留
[16+n] n=0,1..5	**CUTRGEn**	**EPWM 比较器向上计数点触发 DAC使能** 如果该位置1，当EPWM计数器向上计数到CMPDAT ， EPWM 可以触发DAC启动。 0 = EPWM比较器向上计数点触发DAC功能禁止. 1 = EPWM比较器向上计数点触发DAC功能使能 **注1：**当EPWM 计数器工作在向下计数模式.，该位应该保持0； **注2：**在互补模式下， CDTRGE1, 3, 5作为通道 0, 2, 4.的另一个CUTRGE。
[15:14]	**Reserved**	保留
[8+n] n=0,1..5	**PTEn**	**EPWM周期点触发DAC使能** 如果该位设置为1，当EPWM向上计数器计到(PERIODn+1)EPWM可以触发DAC开始转换。 0 = EPWM 周期点触发 DAC 功能禁止. 1 = EPWM周期点触发 DAC 功能使能
[7:6]	**Reserved**	保留
[n] n=0,1..5	**ZTEn**	**EPWM 0点触发DAC 使能** 如果该位置1，当EPWM向下计数器计到0，EPWM 可以触发EADC/DAC/DMA 开始动作。

		0 = EPWM 0点触发DAC功能禁止 1 = EPWM 0点触发DAC功能使能

EPWM触发EADC源选择寄存器0(EPWM_EADCTS0)

寄存器	偏移量	R/W	描述	复位值
EPWM_EADCTS0	EPWMx_BA+0xF8	R/W	EPWM 触发 EADC 源选择寄存器 0	0x0000_0000

31	30	29	28	27	26	25	24
TRGEN3	Reserved			TRGSEL3			
23	22	21	20	19	18	17	16
TRGEN2	Reserved			TRGSEL2			
15	14	13	12	11	10	9	8
TRGEN1	Reserved			TRGSEL1			
7	6	5	4	3	2	1	0
TRGEN0	Reserved			TRGSEL0			

位	描述	
[31]	**TRGEN3**	EPWM_CH3 触发 EADC 使能位
[30:28]	**Reserved**	保留
[27:24]	**TRGSEL3**	**EPWM_CH3 触发 EADC源选择** 0000 = EPWM_CH2 0点 0001 = EPWM_CH2 周期点 0010 = EPWM_CH2 0点或周期点 0011 = EPWM_CH2 向上计数到比较点 0100 = EPWM_CH2向下计数到比较点 0101 = EPWM_CH3 0点 0110 = EPWM_CH3周期点 0111 = EPWM_CH3 0点或周期点 1000 = EPWM_CH3向上计数到比较点 1001 = EPWM_CH3向下计数到比较点 1010 = EPWM_CH0向上计数到自由触发比较点 1011 = EPWM_CH0向下计数到自由触发比较点 1100 = EPWM_CH2向上计数到自由触发比较点 1101 = EPWM_CH2向下计数到自由触发比较点 1110 = EPWM_CH4向上计数到自由触发比较点 1111 = EPWM_CH4向下计数到自由触发比较点
[23]	**TRGEN2**	EPWM_CH2 触发 EADC 使能位
[22:20]	**Reserved**	保留
[19:16]	**TRGSEL2**	**EPWM_CH2触发 EADC源选择** 0000 = EPWM_CH2 0点

		0001 = EPWM_CH2周期点 0010 = EPWM_CH2 0点或周期点 0011 = EPWM_CH2向上计数到比较点 0100 = EPWM_CH2向下计数到比较点 0101 = EPWM_CH3 0点 0110 = EPWM_CH3周期点 0111 = EPWM_CH3 0点或周期点 1000 = EPWM_CH3向上计数到比较点 1001 = EPWM_CH3向下计数到比较点 1010 = EPWM_CH0向上计数到自由触发比较点 1011 = EPWM_CH0向下计数到自由触发比较点 1100 = EPWM_CH2向上计数到自由触发比较点 1101 = EPWM_CH2向下计数到自由触发比较点 1110 = EPWM_CH4向上计数到自由触发比较点 1111 = EPWM_CH4向下计数到自由触发比较点
[15]	**TRGEN1**	EPWM_CH1 触发 EADC 使能位
[14:12]	**Reserved**	保留
[11:8]	**TRGSEL1**	**EPWM_CH1 触发 EADC 源选择** 0000 = EPWM_CH0 0点 0001 = EPWM_CH0周期点 0010 = EPWM_CH0 0点或周期点 0011 = EPWM_CH0向上计数到CMPDAT 值点 0100 = EPWM_CH0向下计数到CMPDAT 值点 0101 = EPWM_CH1 0点 0110 = EPWM_CH1周期点 0111 = EPWM_CH1 0点或周期点 1000 = EPWM_CH1向上计数到比较点 1001 = EPWM_CH1向下计数到比较点 1010 = EPWM_CH0向上计数到自由触发比较点 1011 = EPWM_CH0向下计数到自由触发比较点 1100 = EPWM_CH2向上计数到自由触发比较点 1101 = EPWM_CH2向下计数到自由触发比较点 1110 = EPWM_CH4向上计数到自由触发比较点 1111 = EPWM_CH4向下计数到自由触发比较点
[7]	**TRGEN0**	EPWM_CH0 触发 EADC 使能位
[6:4]	**Reserved**	保留
[3:0]	**TRGSEL0**	**EPWM_CH0 触发 EADC 源选择** 0000 = EPWM_CH0 0点 0001 = EPWM_CH0周期点 0010 = EPWM_CH0 0点或周期点 0011 = EPWM_CH0向上计数到比较点 0100 = EPWM_CH0向下计数到比较点

		0101 = EPWM_CH1 0点 0110 = EPWM_CH1周期点 0111 = EPWM_CH1 0点或周期点 1000 = EPWM_CH1向上计数到比较点 1001 = EPWM_CH1向下计数到比较点 1010 = EPWM_CH0向上计数到自由触发比较点 1011 = EPWM_CH0向下计数到自由触发比较点 1100 = EPWM_CH2向上计数到自由触发比较点 1101 = EPWM_CH2向下计数到自由触发比较点 1110 = EPWM_CH4向上计数到自由触发比较点 1111 = EPWM_CH4向下计数到自由触发比较点

EPWM 触发EADC源选择寄存器1(EPWM_EADCTS1)

寄存器	偏移量	R/W	描述	复位值
EPWM_EADCTS1	EPWMx_BA+0xFC	R/W	EPWM 触发 EADC 源选择寄存器 1	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
TRGEN5	Reserved			TRGSEL5			
7	6	5	4	3	2	1	0
TRGEN4	Reserved			TRGSEL4			

位	描述	
[31:16]	**Reserved**	保留
[15]	**TRGEN5**	EPWM_CH5 触发EADC使能位
[14:12]	**Reserved**	保留
[11:8]	**TRGSEL5**	**EPWM_CH5 触发EADC源选择** 0000 = EPWM_CH4 0点 0001 = EPWM_CH4周期点 0010 = EPWM_CH4 0点或周期点 0011 = EPWM_CH4向上计数到比较点 0100 = EPWM_CH4向下计数到比较点 0101 = EPWM_CH5 0点 0110 = EPWM_CH5周期点 0111 = EPWM_CH5 0点或周期点 1000 = EPWM_CH5向上计数到比较点 1001 = EPWM_CH5向下计数到比较点 1010 = EPWM_CH0向上计数到自由触发比较点 1011 = EPWM_CH0向下计数到自由触发比较点 1100 = EPWM_CH2向上计数到自由触发比较点 1101 = EPWM_CH2向下计数到自由触发比较点 1110 = EPWM_CH4向上计数到自由触发比较点 1111 = EPWM_CH4向下计数到自由触发比较点
[7]	**TRGEN4**	EPWM_CH4 触发 EADC 使能位
[6:4]	**Reserved**	保留
[3:0]	**TRGSEL4**	**EPWM_CH4 触发 EADC 源选择**

		0000 = EPWM_CH4 0点 0001 = EPWM_CH4周期点 0010 = EPWM_CH4 0点或周期点 0011 = EPWM_CH4向上计数到比较点 0100 = EPWM_CH4向下计数到比较点 0101 = EPWM_CH5 0点 0110 = EPWM_CH5周期点 0111 = EPWM_CH5 0点或周期点 1000 = EPWM_CH5向上计数到比较点 1001 = EPWM_CH5向下计数到比较点 1010 = EPWM_CH0向上计数到自由触发比较点 1011 = EPWM_CH0向下计数到自由触发比较点 1100 = EPWM_CH2向上计数到自由触发比较点 1101 = EPWM_CH2向下计数到自由触发比较点 1110 = EPWM_CH4向上计数到自由触发比较点 1111 = EPWM_CH4向下计数到自由触发比较点

EPWM自由触发比较寄存器0_1, 2_3, 4_5(EPWM_FTCMPDAT0_1, 2_3, 4_5)

寄存器	偏移量	R/W	描述	复位值
EPWM_FTCMPDAT0_1	EPWMx_BA+0x100	R/W	EPWM自由触发比较寄存器0/1	0x0000_0000
EPWM_FTCMPDAT2_3	EPWMx_BA+0x104	R/W	EPWM自由触发比较寄存器2/3	0x0000_0000
EPWM_FTCMPDAT4_5	EPWMx_BA+0x108	R/W	EPWM自由触发比较寄存器4/5	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
FTCMP							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
FTCMP							

位	描述	
[31:16]	**Reserved**	保留
[15:0]	**FTCMP**	**EPWM自由触发比较寄存器** FTCMP 用于与偶数通道的CNT (EPWM_CNTm[15:0], m=0,2,4)比较来触发EADC. FTCMPDAT0, 2, 4 对应互补组是 EPWM_CH0和EPWM_CH1,、EPWM_CH2 和 EPWM_CH3、EPWM_CH4 和 EPWM_CH5。

EPWM同步开始控制寄存器(EPWM_SSCTL)

寄存器	偏移量	R/W	描述	复位值
EPWM_SSCTL	EPWMx_BA+0x110	R/W	EPWM 同步开始控制寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved						SSRC	
7	6	5	4	3	2	1	0
Reserved		SSEN5	SSEN4	SSEN3	SSEN2	SSEN1	SSEN0

位	描述	
[31:10]	Reserved	保留
[9:8]	SSRC	EPWM 同步启动源选择 00 = 同步启动源来自EPWM0 01 =同步启动源来自EPWM1 10 =同步启动源来自BEPWM0 11 =同步启动源来自BEPWM1
[7:6]	Reserved	保留
[n] n=0,1..5	SSENn	EPWM 同步开始功能使能 当同步开始功能使能， EPWM计数器使能寄存器(EPWM_CNTEN)可以通过对写EPWM同步开始触发位(CNTSEN)来使能。 0 = EPWM 同步开始功能禁止 1 = EPWM 同步开始功能使能

EPWM同步开始触发寄存器(EPWM_SSTRG)

寄存器	偏移量	R/W	描述	复位值
EPWM_SSTRG	EPWMx_BA+0x114	W	EPWM 同步开始触发寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved							CNTSEN

位	描述	
[31:1]	Reserved	保留
[0]	CNTSEN	**EPWM 计数器同步开始使能(只写)** PMW 计数器同步使能功能用于选择EPWM通道 (包括 EPWM0_CHx 和 EPWM1_CHx) 开始同时计数。 如果相关EPWM通道计数器同步开始功能被使能，该位置1也会置位计数器使能位(CNTENn, n 代表通道0 到 5)。

EPWM前沿消稳控制寄存器(EPWM_LEBCTL)

寄存器	偏移量	R/W	描述	复位值
EPWM_LEBCTL	EPWMx_BA+0x118	R/W	EPWM 前沿消稳控制寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved						TRGTYPE	
15	14	13	12	11	10	9	8
Reserved					SRCEN4	SRCEN2	SRCEN0
7	6	5	4	3	2	1	0
Reserved							LEBEN

位	描述	
[31:18]	**Reserved**	保留
[17:16]	**TRGTYPE**	**EPWM前沿消稳触发模式** 0 = 当侦测到前沿消稳源有上升沿，消稳计数器开始计数 1 =当侦测到前沿消稳源有下降沿，消稳计数器开始计数 2 =当侦测到前沿消稳源有下降沿或下降沿，消稳计数器开始计数 3 = 保留
[15:11]	**Reserved**	保留
[10]	**SRCEN4**	**EPWM前沿消稳源来自 EPWM_CH4 使能位** 0 = EPWM前沿消稳源来自EPWM_CH4 禁止 1 = EPWM前沿消稳源来自EPWM_CH4 使能
[9]	**SRCEN2**	**EPWM前沿消稳源来自 EPWM_CH2使能位** 0 = EPWM前沿消稳源来自EPWM_CH2禁止 1 = EPWM前沿消稳源来自EPWM_CH2使能
[8]	**SRCEN0**	**EPWM前沿消稳源来自 EPWM_CH0使能位** 0 = EPWM前沿消稳源来自EPWM_CH0禁止 1 = EPWM前沿消稳源来自EPWM_CH0使能
[7:1]	**Reserved**	保留
[0]	**LEBEN**	**EPWM前沿消稳使能位** 0 = EPWM前沿消稳禁止 1 = EPWM前沿消稳使能

EPWM前沿消稳计数器寄存器(EPWM_LEBCNT)

寄存器	偏移量	R/W	描述	复位值
EPWM_LEBCNT	EPWMx_BA+0x11C	R/W	EPWM 前沿消稳计数器寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							**LEBCNT**
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
LEBCNT							

位	描述	
[31:9]	**Reserved**	保留
[8:0]	**LEBCNT**	**EPWM前沿消稳计数器** 该计数器值决定了前沿消稳窗口大小：消稳窗口大小 = LEBCNT+1，LEB 计数器时钟基于ECLK。

EPWM状态寄存器(EPWM_STATUS)

寄存器	偏移量	R/W	描述	复位值
EPWM_STATUS	EPWMx_BA+0x120	R/W	EPWM 状态寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							DACTRGF
23	22	21	20	19	18	17	16
Reserved		EADCTRGF5	EADCTRGF4	EADCTRGF3	EADCTRGF2	EADCTRGF1	EADCTRGF0
15	14	13	12	11	10	9	8
Reserved					SYNCINF4	SYNCINF2	SYNCINF0
7	6	5	4	3	2	1	0
Reserved		CNTMAXF5	CNTMAXF4	CNTMAXF3	CNTMAXF2	CNTMAXF1	CNTMAXF0

位	描述	
[31:25]	**Reserved**	保留
[24]	**DACTRGF**	**DAC开始转换标志** 0 = 表示没有 DAC开始转换触发事件发生 1 =表示有 DAC开始转换触发事件发生，该位可以软件写1清零
[23:22]	**Reserved**	保留
[16+n] n=0,1..5	**EADCTRGFn**	**EADC 开始转换标志** 0 =表示没有EADC开始转换触发事件发生 1 =表示有EADC开始转换触发事件发生，该位可以软件写1清零
[15:11]	**Reserved**	保留
[8+n/2] n=0,2,4	**SYNCINFn**	**输入同步锁存标志** 0 =表示没有 SYNC_IN事件发生 1 =表示有SYNC_IN 事件发生，该位可以软件写1清零
[7:6]	**Reserved**	保留
[n] n=0,1..5	**CNTMAXFn**	**时基计数器等于0xFFFF 锁存标志** 0 = 表示时基计数器从未达到最大值 0xFFFF 1 = 表示时基计数器达到其最大值，该位可以软件写1清零

EPWM 中断标志累加器寄存器(EPWM_IFAn)

寄存器	偏移量	R/W	描述	复位值
EPWM_IFA0	EPWMx_BA+0x130	R/W	EPWM 中断标志累加器寄存器0	0x0000_0000
EPWM_IFA1	EPWMx_BA+0x134	R/W	EPWM中断标志累加器寄存器1	0x0000_0000
EPWM_IFA2	EPWMx_BA+0x138	R/W	EPWM中断标志累加器寄存器2	0x0000_0000
EPWM_IFA3	EPWMx_BA+0x13C	R/W	EPWM中断标志累加器寄存器3	0x0000_0000
EPWM_IFA4	EPWMx_BA+0x140	R/W	EPWM中断标志累加器寄存器4	0x0000_0000
EPWM_IFA5	EPWMx_BA+0x144	R/W	EPWM中断标志累加器寄存器5	0x0000_0000

31	30	29	28	27	26	25	24
IFAEN	**Reserved**	**IFASEL**		**Reserved**			**STPMOD**
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
IFACNT							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
IFACNT							

位	描述	
[31]	**IFAEN**	**EPWM_CHn 中断标志累加器使能位** 0 = EPWM_CHn 中断标志累加器禁止 1 = EPWM_CHn 中断标志累加器使能
[30]	**Reserved**	保留
[29:28]	**IFASEL**	**EPWM_CHn中断标志累加器源选择** 00 = EPWM_CHn零点 01 = EPWM_CHn通道n周期 10 = EPWM_CHn向上计数比较点 11 = EPWM_CHn向下计数比较点
[27:25]	**Reserved**	保留
[24]	**STPMOD**	**EPWM_CHn累加器停止模式使能** 0 = EPWM_CHn停止模式禁用 1 = EPWM_CHn停止模式禁用
[23:16]	**Reserved**	保留
[15:0]	**IFACNT**	**EPWM_CHn中断标志计数器**

		该寄存器设置计数值，定义（IFACNT+1）次EPWM_CHn周期发生，置位IFAIFn，请求EPWM周期中断。 EPWM标志在每IFACNT[15:0]次EPWM周期将被设置。

EPWM 累加器中断标志寄存器(EPWM_AINTSTS)

寄存器	偏移量	R/W	描述	复位值
EPWM_AINTSTS	EPWMx_BA+0x150	R/W	EPWM 累加器中断标志寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved		IFAIF5	IFAIF4	IFAIF3	IFAIF2	IFAIF1	IFAIF0

位	描述	
[31:6]	Reserved	保留
[n] n=0,1..5	IFAIFn	EPWM_CHn 中断标志累加器中断标志 当条件和EPWM_IFAn寄存器中IFASEL匹配，该标志被硬件置位。

EPWM 累加器中断使能寄存器(EPWM_AINTEN)

寄存器	偏移量	R/W	描述	复位值
EPWM_AINTEN	EPWMx_BA+0x154	R/W	EPWM 累加器中断使能寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved		IFAIEN5	IFAIEN4	IFAIEN3	IFAIEN2	IFAIEN1	IFAIEN0

位	描述	
[31:6]	**Reserved**	保留
[n] n=0,1..5	**IFAIENn**	**EPWM_CHn 中断标志累加器中断使能位** 0 = 中断标志累加器中断禁止 1 = 中断标志累加器中断使能

EPWM 累加器 PDMA 控制寄存器(EPWM_APDMACTL)

寄存器	偏移量	R/W	描述	复位值
EPWM_APDMACTL	EPWMx_BA+0x158	R/W	EPWM 累加器 PDMA 控制寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved		APDMAEN5	APDMAEN4	APDMAEN3	APDMAEN2	APDMAEN1	APDMAEN0

位	描述	
[31:6]	**Reserved**	保留
[n] n=0,1..5	**APDMAENn**	**通道 N 累加器 PDMA 使能位** 0 = 通道 n PDMA 功能禁止 1 = 通道 n PDMA 功能使能，用于通道n触发PDMA传输内存数据到寄存器

EPWM故障检测使能寄存器(EPWM_FDEN)

寄存器	偏移	R/W	描述	复位值
EPWM_FDEN	EPWMx_BA+0x160	R/W	EPWM故障检测使能寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved		FDCKS5	FDCKS4	FDCKS3	FDCKS2	FDCKS1	FDCKS0
15	14	13	12	11	10	9	8
Reserved		FDODIS5	FDODIS4	FDODIS3	FDODIS2	FDODIS1	FDODIS0
7	6	5	4	3	2	1	0
Reserved		FDEN5	FDEN4	FDEN3	FDEN2	FDEN1	FDEN0

位	描述	
[31:22]	Reserved	保留
[16+n] n=0,1..5	FDCKSn	**EPWM通道n故障检测时钟源选择位** 0 = EPWMx_CLK，x为 0或1 1 = EPWMx_CLK由分频器分频，x为0或1
[8+n] n=0,1..5	FDODISn	**EPWM通道n输出故障检测禁用位** 0 = EPWM检测故障和输出使能 1 = EPWM检测故障和输出禁用
[n] n=0,1..5	FDENn	**EPWM故障检测功能使能位** 0 = 故障检测功能禁用 1 = 故障检测功能使能

EPWM故障检测控制寄存器0~5 (EPWM_FDCTL0~5)

寄存器	偏移	R/W	描述	复位值
EPWM_FDCTL0	EPWMx_BA+0x164	R/W	EPWM故障检测控制寄存器0	0x0000_0000
EPWM_FDCTL1	EPWMx_BA+0x168	R/W	EPWM故障检测控制寄存器1	0x0000_0000
EPWM_FDCTL2	EPWMx_BA+0x16C	R/W	EPWM故障检测控制寄存器2	0x0000_0000
EPWM_FDCTL3	EPWMx_BA+0x170	R/W	EPWM故障检测控制寄存器3	0x0000_0000
EPWM_FDCTL4	EPWMx_BA+0x174	R/W	EPWM故障检测控制寄存器4	0x0000_0000
EPWM_FDCTL5	EPWMx_BA+0x178	R/W	EPWM故障检测控制寄存器5	0x0000_0000

31	30	29	28	27	26	25	24
FDDGEN	Reserved	FDCKSEL		Reserved			
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved					DGSMPCYC		
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
FDMSKEN	Reserved						
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved	TRMSKCNT						

位	描述	
[31]	**FDDGEN**	**故障检测抗尖峰脉冲使能位** 0 = 故障检测抗尖峰脉冲功能禁用 1 =故障检测抗尖峰脉冲功能使能
[30]	**Reserved**	保留
[29:28]	**FDCKSEL**	**EPWM通道故障检测时钟选择** 00 = FLT_CLK/1 01 = FLT_CLK/2 10 = FLT_CLK/4 11 = FLT_CLK/8 **注：**FLT_CLK 是FDCKSn (EPWM_FDENn[16+n], n=0,1..5)选择的时钟。
[27:19]	**Reserved**	保留
[18:16]	**DGSMPCYC**	**抗尖峰脉冲采样周期** FDCKS设置为0： 采样检测信号每EPWMx_CLK * (2^FDCKSEL)周期，检测DGSMPCYC+1次； FDCKS设置为1： 采样检测信号每EPWMx_CLK * CLKPSC * (2^FDCKSEL)周期，检测DGSMPCYC+1

		次。 **注:** CLKPSC (EPWM_CLKPSC)为0： TRMSKCNT>= DGSMPCYC + 2 FDCKS is 1和CLKPSC (EPWM_CLKPSC)为1： TRMSKCNT >= DGSMPCYC + 1 FDCKS is 1和CLKPSC (EPWM_CLKPSC)为2： TRMSKCNT >= DGSMPCYC
[15]	**FDMSKEN**	**故障检测屏蔽使能位** 0 = 故障检测屏蔽功能禁用 1 = 故障检测屏蔽功能使能
[14:7]	**Reserved**	保留
[6:0]	**TRMSKCNT**	**转换屏蔽计数器** 故障检测结果在计数器从0计数到TRMSKCNT之前将被屏蔽。 FDCKS设置为0： 屏蔽时间为EPWMx_CLK * (2^FDCKSEL) * (TRMSKCNT+2) FDCKS设置为1： 屏蔽时间为EPWMx_CLK * CLKPSC * (2^FDCKSEL) * (TRMSKCNT+2) **注：** CLKPSC (EPWM_CLKPSC)为0： TRMSKCNT >= DGSMPCYC + 2 FDCKS为1且CLKPSC (EPWM_CLKPSC)为1： TRMSKCNT >= DGSMPCYC + 1 FDCKS为1且CLKPSC (EPWM_CLKPSC)为2： TRMSKCNT >= DGSMPCYC

EPWM故障检测中断使能寄存器 (EPWM_FDIEN)

寄存器	偏移	R/W	描述	复位值
EPWM_FDIEN	EPWMx_BA+0x17C	R/W	EPWM故障检测中断使能寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved		FDIEN5	FDIEN4	FDIEN3	FDIEN2	FDIEN1	FDIEN0

位	描述	
[31:6]	Reserved	保留
[n]	FDIENn	EPWM通道n故障检测中断使能位 0 = EPWM通道n故障检测中断禁用 1 = EPWM通道n故障检测中断使能

EPWM故障检测中断标志寄存器(EPWM_FDSTS)

寄存器	偏移	R/W	描述	复位值
EPWM_FDSTS	EPWMx_BA+0x180	R/W	EPWM故障检测中断标志寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved		FDIF5	FDIF4	FDIF3	FDIF2	FDIF1	FDIF0

位	描述	
[31:6]	Reserved	保留
[5:0]	FDIFn	EPWM通道n故障检测中断标志位 当EPWM输出短路，故障检测中断标志将置位，该位软件写1清0。

EPWM触发EADC预分频控制寄存器(EPWM_EADCPSCCTL)

寄存器	偏移	R/W	描述	复位值
EPWM_EADCPSCCTL	EPWMx_BA+0x184	R/W	EPWM触发EADC预分频控制寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
		PSCEN5	PSCEN4	PSCEN3	PSCEN2	PSCEN1	PSCEN0

位	描述	
[31:6]	Reserved	保留
[n] n=0,1..5	PSCENn	EPWM触发EADC预分频功能使能位 0 = EPWM触发EADC预分频功能禁用 1 = EPWM触发EADC预分频功能使能

EPWM触发EADC预分频寄存器0 (EPWM_EADCPSC0)

寄存器	偏移	R/W	描述	复位值
EPWM_EADCPSC0	EPWMx_BA+0x188	R/W	EPWM触发EADC预分频寄存器0	0x0000_0000

31	30	29	28	27	26	25	24
Reserved				EADCPSC3			
23	22	21	20	19	18	17	16
Reserved				EADCPSC2			
15	14	13	12	11	10	9	8
Reserved				EADCPSC1			
7	6	5	4	3	2	1	0
Reserved				EADCPSC0			

位	描述	
[31:28]	Reserved	保留
[27:24]	EADCPSC3	**EPWM通道3触发EADC预分频** 该寄存器设置EPWM_CH3触发EADC事件发生次数(EADCPSC3+1)，用于触发EADC和设置触发EADC标志位EADCTRGF3。
[23:20]	Reserved	保留
[19:16]	EADCPSC2	**EPWM 通道2触发EADC预分频** 该寄存器设置EPWM_CH2触发EADC事件发生次数(EADCPSC2+1)，用于触发EADC和设置触发EADC标志位EADCTRGF2。
[15:12]	Reserved	保留
[11:8]	EADCPSC1	**EPWM 通道1触发EADC预分频** 该寄存器设置EPWM_CH1触发EADC事件发生次数(EADCPSC1+1)，用于触发EADC和设置触发EADC标志位EADCTRG1。
[7:4]	Reserved	保留
[3:0]	EADCPSC0	**EPWM 通道0触发EADC预分频** 该寄存器设置EPWM_CH0触发EADC事件发生次数(EADCPSC0+1)，用于触发EADC和设置触发EADC标志位EADCTRGF0。

EPWM触发EADC预分频寄存器1 (EPWM_EADCPSC1)

寄存器	偏移	R/W	描述	复位值
EPWM_EADCPSC1	EPWMx_BA+0x18C	R/W	EPWM触发EADC预分频寄存器1	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved				EADCPSC5			
7	6	5	4	3	2	1	0
Reserved				EADCPSC4			

位	描述	
[31:12]	Reserved	保留
[11:8]	EADCPSC5	EPWM 通道5触发EADC预分频 该寄存器设置EPWM_CH5触发EADC事件发生次数(EADCPSC5+1)，用于触发EADC和设置触发EADC标志位EADCTRGF5。
[7:4]	Reserved	保留
[3:0]	EADCPSC4	EPWM 通道4触发EADC预分频 该寄存器设置EPWM_CH4触发EADC事件发生次数(EADCPSC4+1)，用于触发EADC和设置触发EADC标志位EADCTRGF4。

EPWM触发EADC预分频计数器寄存器0 (EPWM_EADCPSCNT0)

寄存器	偏移	R/W	描述	复位值
EPWM_EADCPSCNT0	EPWMx_BA+0x190	R/W	EPWM 触发EADC预分频计数器寄存器0	0x0000_0000

31	30	29	28	27	26	25	24
Reserved				PSCNT3			
23	22	21	20	19	18	17	16
Reserved				PSCNT2			
15	14	13	12	11	10	9	8
Reserved				PSCNT1			
7	6	5	4	3	2	1	0
Reserved				PSCNT0			

位	描述	
[31:28]	Reserved	保留
[27:24]	PSCNT3	**EPWM触发EADC预分频计数器3** 用户可以监测PSCNT3知道4位EADC预分频计数器当前值。 **注1：** 用户只有在PSCEN3为0时能写； **注2：** 写数据限制： PSCNT3 < EADCPSC3。
[23:20]	Reserved	保留
[19:16]	PSCNT2	**EPWM触发EADC预分频计数器2** 用户可以监测PSCNT2知道4位EADC预分频计数器当前值。 **注1：** 用户只有在PSCEN2为0时能写； **注2：** 写数据限制： PSCNT2< EADCPSC2。
[15:12]	Reserved	保留
[11:8]	PSCNT1	**EPWM触发EADC预分频计数器1** 用户可以监测PSCNT1知道4位EADC预分频计数器当前值。 **注1：** 用户只有在PSCEN1为0时能写； **注2：** 写数据限制： PSCNT1< EADCPSC1。
[7:4]	Reserved	保留
[3:0]	PSCNT0	**EPWM触发EADC预分频计数器0** 用户可以监测PSCNT0知道4位EADC预分频计数器当前值。 **注1：** 用户只有在PSCEN0为0时能写； **注2：** 写数据限制： PSCNT0< EADCPSC0。

EPWM触发EADC预分频计数器寄存器1 (EPWM_EADCPSCNT1)

寄存器	偏移	R/W	描述	复位值
EPWM_EADCPSCNT1	EPWMx_BA+0x194	R/W	EPWM触发EADC预分频计数器寄存器1	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved				PSCNT5			
7	6	5	4	3	2	1	0
Reserved				PSCNT4			

位	描述	
[31:12]	Reserved	保留
[11:8]	PSCNT5	**EPWM触发EADC预分频计数器5** 用户可以监测PSCNT5知道4位EADC预分频计数器当前值。 **注1：** 用户只有在PSCEN5为0时能写； **注2：** 写数据限制： PSCNT5< EADCPSC5。
[7:4]	Reserved	保留
[3:0]	PSCNT4	**EPWM触发EADC预分频计数器4** 用户可以监测PSCNT4知道4位EADC预分频计数器当前值。 **注1：** 用户只有在PSCEN4为0时能写； **注2：** 写数据限制： PSCNT4< EADCPSC4。

EPWM捕获输入使能寄存器(EPWM_CAPINEN)

寄存器	偏移量	R/W	描述	复位值
EPWM_CAPINEN	EPWMx_BA+0x200	R/W	EPWM 捕获输入使能寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved		CAPINEN5	CAPINEN4	CAPINEN3	CAPINEN2	CAPINEN1	CAPINEN0

位	描述	
[31:6]	**Reserved**	保留
[n] n=0,1..5	**CAPINENn**	**捕获输入使能** 0 =禁止EPWM输入捕获，EPWM通道捕获功能的输入总是被认为是0 1 =使能EPWM输入捕获，EPWM通道捕获功能的输入来自相关的复用管脚

EPWM 捕获控制寄存器(EPWM_CAPCTL)

寄存器	偏移量	R/W	描述	复位值
EPWM_CAPCTL	EPWMx_BA+0x204	R/W	EPWM 捕获控制寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved		FCRLDEN5	FCRLDEN4	FCRLDEN3	FCRLDEN2	FCRLDEN1	FCRLDEN0
23	22	21	20	19	18	17	16
Reserved		RCRLDEN5	RCRLDEN4	RCRLDEN3	RCRLDEN2	RCRLDEN1	RCRLDEN0
15	14	13	12	11	10	9	8
Reserved		CAPINV5	CAPINV4	CAPINV3	CAPINV2	CAPINV1	CAPINV0
7	6	5	4	3	2	1	0
Reserved		CAPEN5	CAPEN4	CAPEN3	CAPEN2	CAPEN1	CAPEN0

位	描述	
[31:30]	**Reserved**	保留
[24+n] n=0,1..5	**FCRLDENn**	**下降沿捕获重载使能** 0 =禁止下降沿捕获重载计数器 1 =使能下降沿捕获重载计数器
[23:22]	**Reserved**	保留
[16+n] n=0,1..5	**RCRLDENn**	**上升沿捕获重载使能** 0 =禁止上升沿捕获重载计数器 1 =使能上升沿捕获重载计数器
[15:14]	**Reserved**	保留
[8+n] n=0,1..5	**CAPINVn**	**捕获反向使能** 0 =.禁止捕获源反向 1 =使能捕获源反向，将来自GPIO的信号反向
[7:6]	**Reserved**	保留
[n] n=0,1..5	**CAPENn**	**捕获功能使能** 0 =禁止捕获功能，RCAPDAT/FCAPDAT不会更新 1 =使能捕获功能，当检测到输入信号的上升/下降沿时锁存EPWM计数器值并保存到RCAPDAT（上升沿锁存）和FCAPDAT（下降沿锁存）

EPWM 捕获状态寄存器(EPWM_CAPSTS)

寄存器	偏移量	R/W	描述	复位值
EPWM_CAPSTS	EPWMx_BA+0x208	R	EPWM 捕获状态寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved		CFLIFOV5	CFLIFOV4	CFLIFOV3	CFLIFOV2	CFLIFOV1	CFLIFOV0
7	6	5	4	3	2	1	0
Reserved		CRLIFOV5	CRLIFOV4	CRLIFOV3	CRLIFOV2	CRLIFOV1	CRLIFOV0

位	描述	
[31:14]	**Reserved**	保留
[8+n] n=0,1..5	**CFLIFOVn**	**下降沿捕获锁存中断标志溢出状态（只读）** 如果相应的CFLIF是1，且下降沿锁存事件发生，此标志为1。 注：当用户清相应的CFLIF，该位将自动清零。
[7:6]	**Reserved**	保留
[n] n=0,1..5	**CRLIFOVn**	**上升沿捕获锁存中断标志溢出状态（只读）** 如果相应的CRLIF是1，且上升沿捕获事件发生，此标志为1。 注：当用户清相应的CRLIF，该位将自动清零。

EPWM上升沿捕获数据寄存器0~5(EPWM_RCAPDAT 0~5)

寄存器	偏移量	R/W	描述	复位值
EPWM_RCAPDAT0	EPWMx_BA+0x20C	R	EPWM 上升沿捕获数据寄存器0	0x0000_0000
EPWM_RCAPDAT1	EPWMx_BA+0x214	R	EPWM 上升沿捕获数据寄存器1	0x0000_0000
EPWM_RCAPDAT2	EPWMx_BA+0x21C	R	EPWM 上升沿捕获数据寄存器2	0x0000_0000
EPWM_RCAPDAT3	EPWMx_BA+0x224	R	EPWM 上升沿捕获数据寄存器3	0x0000_0000
EPWM_RCAPDAT4	EPWMx_BA+0x22C	R	EPWM 上升沿捕获数据寄存器4	0x0000_0000
EPWM_RCAPDAT5	EPWMx_BA+0x234	R	EPWM 上升沿捕获数据寄存器5	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
RCAPDAT							
7	6	5	4	3	2	1	0
RCAPDAT							

位	描述	
[31:16]	Reserved	保留
[15:0]	RCAPDAT	EPWM上升沿捕获数据寄存器（只读） 当上升沿捕获条件发生，EPWM计数器值将被保存到该寄存器。

EPWM 下降沿捕获数据寄存器0~5 (EPWM_FCAPDAT 0~5)

寄存器	偏移量	R/W	描述	复位值
EPWM_FCAPDAT0	EPWMx_BA+0x210	R	EPWM 下降沿捕获寄存器0	0x0000_0000
EPWM_FCAPDAT1	EPWMx_BA+0x218	R	EPWM 下降沿捕获寄存器1	0x0000_0000
EPWM_FCAPDAT2	EPWMx_BA+0x220	R	EPWM 下降沿捕获寄存器2	0x0000_0000
EPWM_FCAPDAT3	EPWMx_BA+0x228	R	EPWM 下降沿捕获寄存器3	0x0000_0000
EPWM_FCAPDAT4	EPWMx_BA+0x230	R	EPWM 下降沿捕获寄存器4	0x0000_0000
EPWM_FCAPDAT5	EPWMx_BA+0x238	R	EPWM 下降沿捕获寄存器5	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
FCAPDAT							
7	6	5	4	3	2	1	0
FCAPDAT							

位	描述	
[31:16]	Reserved	保留
[15:0]	FCAPDAT	**EPWM下降沿捕获寄存器(只读)** 当下降沿捕获条件发生，EPWM计数器值将被保存到该寄存器。

EPWM PDMA控制寄存器(EPWM_PDMACTL)

寄存器	偏移量	R/W	描述	复位值
EPWM_PDMACTL	EPWMx_BA+0x23C	R/W	EPWM PDMA 控制寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved			CHSEL4_5	CAPORD4_5	CAPMOD4_5		CHEN4_5
15	14	13	12	11	10	9	8
Reserved			CHSEL2_3	CAPORD2_3	CAPMOD2_3		CHEN2_3
7	6	5	4	3	2	1	0
Reserved			CHSEL0_1	CAPORD0_1	CAPMOD0_1		CHEN0_1

位	描述	
[31:21]	**Reserved**	保留
[20]	**CHSEL4_5**	**选择通道 4/5 用于PDMA传输** 0 = 通道4. 1 = 通道5
[19]	**CAPORD4_5**	**捕获通道 4/5 上升沿/下降沿顺序** 设置该位用于确定，当CAPMOD4_5 =11时，是EPWM_RCAPDAT4/5 还是EPWM_FCAPDAT4/5作为第一次捕获的数据通过PDMA开始搬移到内存。 0 = EPWM_FCAPDAT4/5是先捕获并搬移到内存的数据 1 = EPWM_RCAPDAT4/5是先捕获并搬移到内存的数据
[18:17]	**CAPMOD4_5**	**选择 EPWM_RCAPDAT4/5 或 EPWM_FCAPDAT4/5 来做PDMA 数据搬移** 00 = 保留 01 = EPWM_RCAPDAT4/5. 10 = EPWM_FCAPDAT4/5. 11 = EPWM_RCAPDAT4/5 和 EPWM_FCAPDAT4/5
[16]	**CHEN4_5**	**通道4/5 PDMA 使能** 0 = 通道 4/5 PDMA功能禁止 1 =通道4/5 PDMA 功能使能，用于通道4/5 捕获数据并搬移到内存
[15:13]	**Reserved**	保留
[12]	**CHSEL2_3**	**选择通道 2/3用于PDMA传输** 0 =通道2. 1 =通道3
[11]	**CAPORD2_3**	**捕获通道 2/3 上升沿/下降沿顺序** 设置该位用于确定，当CAPMOD2_3 =11时，是EPWM_RCAPDAT2/3 还是

		EPWM_FCAPDAT2/3作为第一次捕获的数据通过PDMA开始搬移到内存。 0 = EPWM_FCAPDAT2/3是先捕获并搬移到内存的数据 1 = EPWM_RCAPDAT2/3是先捕获并搬移到内存的数据
[10:9]	**CAPMOD2_3**	**选择 EPWM_RCAPDAT2/3 或 EPWM_FCAODAT2/3来做PDMA 数据搬移** 00 = 保留 01 = EPWM_RCAPDAT2/3. 10 = EPWM_FCAPDAT2/3 11 = EPWM_RCAPDAT2/3 和 EPWM_FCAPDAT2/3.
[8]	**CHEN2_3**	**通道2/3 PDMA 使能** 0 = 通道 2/3 PDMA功能禁止 1 = 通道2/3功能使能，用于通道2/3捕获数据并搬移到内存
[7:5]	**Reserved**	保留
[4]	**CHSEL0_1**	**选择通道0/1来做PDMA 数据搬移** 0 =通道0 1 =通道1
[3]	**CAPORD0_1**	**捕获通道0/1上升沿/下降沿顺序** 设置该位用于确定，当CAPMOD0_1 =11时，是EPWM_RCAPDAT0/1还是EPWM_FCAPDAT0/1作为第一次捕获的数据通过PDMA开始搬移到内存。 0 = EPWM_FCAPDAT0/1是先捕获并搬移到内存的数据 1 = EPWM_RCAPDAT0/1是先捕获并搬移到内存的数据
[2:1]	**CAPMOD0_1**	**选择 EPWM_RCAPDAT0/1 或 EPWM_FCAPDAT0/1来做PDMA 数据搬移** 00 = 保留 01 = EPWM_RCAPDAT0/1. 10 = EPWM_FCAPDAT0/1 11 = EPWM_RCAPDAT0/1 和 EPWM_FCAPDAT0/1.
[0]	**CHEN0_1**	**通道0/1 PDMA 使能** 0 = 通道0/1 PDMA功能禁止 1 = 通道0/1 PDMA功能使能，用于通道0/1捕获数据并搬移到内存

EPWM 捕获通道0_1, 2_3, 4_5 PDMA 寄存器(EPWM_PDMACAP 0_1, 2_3, 4_5)

寄存器	偏移量	R/W	描述	复位值
EPWM_PDMACAP0_1	EPWMx_BA+0x240	R	EPWM 捕获通道01 PDMA寄存器	0x0000_0000
EPWM_PDMACAP2_3	EPWMx_BA+0x244	R	EPWM捕获通道23 PDMA寄存器	0x0000_0000
EPWM_PDMACAP4_5	EPWMx_BA+0x248	R	EPWM捕获通道45 PDMA寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
CAPBUF							
7	6	5	4	3	2	1	0
CAPBUF							

位	描述	
[31:16]	Reserved	保留
[15:0]	CAPBUF	EPWM 捕获 PDMA 寄存器（只读） 该寄存器用于作为一个缓存，通过PDMA来传输EPWM 捕获到的上升沿或下降沿数据到内存

EPWM捕获中断使能寄存器(EPWM_CAPIEN)

寄存器	偏移量	R/W	描述	复位值
EPWM_CAPIEN	EPWMx_BA+0x250	R/W	EPWM 捕获中断使能寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved		CAPFIEN5	CAPFIEN4	CAPFIEN3	CAPFIEN2	CAPFIEN1	CAPFIEN0
7	6	5	4	3	2	1	0
Reserved		CAPRIEN5	CAPRIEN4	CAPRIEN3	CAPRIEN2	CAPRIEN1	CAPRIEN0

位	描述	
[31:14]	**Reserved**	保留
[8+n] n=0,1..5	**CAPFIENn**	**EPWM 下降沿捕获锁存中断使能** 0 =.禁止下降沿锁存中断 1 = 使能下降沿锁存中断
[7:6]	**Reserved**	保留
[n] n=0,1..5	**CAPRIENn**	**EPWM上升沿捕获锁存中断使能** 0 =禁止上升沿锁存中断 1 =使能上升沿锁存中断

EPWM捕获中断标志寄存器(EPWM_CAPIF)

寄存器	偏移量	R/W	描述	复位值
EPWM_CAPIF	EPWMx_BA+0x254	R/W	EPWM 捕获中断标志寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved		CFLIF5	CFLIF4	CFLIF3	CFLIF2	CFLIF1	CFLIF0
7	6	5	4	3	2	1	0
Reserved		CRLIF5	CRLIF4	CRLIF3	CRLIF2	CRLIF1	CRLIF0

位	描述	
[31:14]	Reserved	保留
[8+n] n=0,1..5	CFLIFn	**EPWM下降沿捕获锁存中断标志** 该位写1清0。 0 =没有下降沿捕获锁存条件发生 1 =下降沿捕获锁存条件发生，该标志被置高 **注：** 当带PDMA运行捕获功能， PDMA 搬移完数据后，CAPIF对应通道的CFLIF位将被硬件清除。
[7:6]	Reserved	保留
[n] n=0,1..5	CRLIFn	**EPWM上升沿捕获锁存中断标志** 该位写1清0。 0 =没有上升沿捕获锁存条件发生 1 =上升沿捕获锁存条件发生，该标志被置高 **注：** 当带PDMA运行捕获功能，PDMA 搬移完数据后，CAPIF对应通道的CFLIF位将被硬件清除。

EPWM 周期寄存器缓存 0~5(EPWM_PBUF0~5)

寄存器	偏移量	R/W	描述	复位值
EPWM_PBUF0	EPWMx_BA+0x304	R	EPWM PERIOD0缓存	0x0000_0000
EPWM_PBUF1	EPWMx_BA+0x308	R	EPWM PERIOD1缓存	0x0000_0000
EPWM_PBUF2	EPWMx_BA+0x30C	R	EPWM PERIOD2缓存	0x0000_0000
EPWM_PBUF3	EPWMx_BA+0x310	R	EPWM PERIOD3缓存	0x0000_0000
EPWM_PBUF4	EPWMx_BA+0x314	R	EPWM PERIOD4缓存	0x0000_0000
EPWM_PBUF5	EPWMx_BA+0x318	R	EPWM PERIOD5缓存	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
PBUF							
7	6	5	4	3	2	1	0
PBUF							

位	描述	
[31:16]	**Reserved**	保留
[15:0]	**PBUF**	**EPWM周期寄存器缓存（只读）** 用作有效周期寄存器

EPWM 比较寄存器缓存0~5(EPWM_CMPBUF0~5)

寄存器	偏移量	R/W	描述	复位值
EPWM_CMPBUF0	EPWMx_BA+0x31C	R	EPWM CMPDAT0 缓存	0x0000_0000
EPWM_CMPBUF1	EPWMx_BA+0x320	R	EPWM CMPDAT1 缓存	0x0000_0000
EPWM_CMPBUF2	EPWMx_BA+0x324	R	EPWM CMPDAT2 缓存	0x0000_0000
EPWM_CMPBUF3	EPWMx_BA+0x328	R	EPWM CMPDAT3 缓存	0x0000_0000
EPWM_CMPBUF4	EPWMx_BA+0x32C	R	EPWM CMPDAT4 缓存	0x0000_0000
EPWM_CMPBUF5	EPWMx_BA+0x330	R	EPWM CMPDAT5 缓存	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
CMPBUF							
7	6	5	4	3	2	1	0
CMPBUF							

位	描述	
[31:16]	**Reserved**	保留
[15:0]	**CMPBUF**	**EPWM比较寄存器缓存（只读）** 用作有效CMP寄存器

EPWM CLKPSC 缓存 0_1, 2_3, 4_5(EPWM_CPSCBUF0_1, 2_3, 4_5)

寄存器	偏移量	R/W	描述	复位值
EPWM_CPSCBUF0_1	EPWMx_BA+0x334	R	EPWM CLKPSC0_1 缓存	0x0000_0000
EPWM_CPSCBUF2_3	EPWMx_BA+0x338	R	EPWM CLKPSC2_3 缓存	0x0000_0000
EPWM_CPSCBUF4_5	EPWMx_BA+0x33C	R	EPWM CLKPSC4_5 缓存	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved				CPSCBUF			
7	6	5	4	3	2	1	0
CPSCBUF							

位	描述	
[31:12]	**Reserved**	保留
[11:0]	**CPSCBUF**	**EPWM 计数器时钟预分频缓存** 用作 EPWM 计数器时钟预分频有效寄存器

EPWM_FTCMPDAT 缓存(EPWM_FTCBUF0_1,2_3,4_5)

寄存器	偏移量	R/W	描述	复位值
EPWM_FTCBUF0_1	EPWMx_BA+0x340	R	EPWM FTCMPDAT0_1 缓存	0x0000_0000
EPWM_FTCBUF2_3	EPWMx_BA+0x344	R	EPWM FTCMPDAT2_3 缓存	0x0000_0000
EPWM_FTCBUF4_5	EPWMx_BA+0x348	R	EPWM FTCMPDAT4_5 缓存	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
FTCMPBUF							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
FTCMPBUF							

位	描述	
[31:16]	**Reserved**	保留
[15:0]	**FTCMPBUF**	**EPWM FTCMPDAT 缓存 (只读)** 用作有效FTCMPDAT 寄存器

EPWM FTCMPDAT指示寄存器(EPWM_FTCI)

寄存器	偏移量	R/W	描述	复位值
EPWM_FTCI	EPWMx_BA+0x34C	R/W	EPWM FTCMPDAT 指示寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved					**FTCMD4**	**FTCMD2**	**FTCMD0**
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved					**FTCMU4**	**FTCMU2**	**FTCMU0**

位	描述	
[31:11]	**Reserved**	保留
[8+n/2] n=0,2,4	**FTCMDn**	**EPWM FTCMPDAT 向下指示** 当 EPWM 计数器下数到 EPWM_FTCMPDATn 时该指示位被硬件置位，软件可以写 1 来清零。
[7:3]	**Reserved**	保留
[n/2] n=0,2,4	**FTCMUn**	**EPWM FTCMPDAT 向上指示** 当 EPWM 计数器上数到 EPWM_FTCMPDATn 时该指示位被硬件置位，软件可以写 1 来清零。

6.13 基本 PWM 发生器和捕获定时器(BPWM)

6.13.1 概述

如图6. 13-1所示，M480 有 2 组 BPWM 发生器BPWM0 和 BPWM1。每组 BPWM 都支持 6 通道的 BPWM 输出和捕捉输入、12 位的预分频器、16 位比较器，6 路通道共享一个16 位的 BPWM 计数器。BPWM 计数器支持向上、向下和上下的计数方式。BPWM 通过比较比较器和计数器的值来产生事件。这些事件可以用来产生 BPWM 脉冲、中断和触发EADC0/1开始转换的信号。BPWM 的输出控制单元支持极性输出、独立的管脚屏蔽和三态输出使能。

BPWM 发生器还支持输入捕捉功能，在输入通道有上升沿、下降沿或双边沿信号发生时锁存 BPWM 计数器的值到对应的寄存器。

6.13.2 特性

6.13.2.1 BPWM 功能特性

- 时钟频率最高可达到最大的 PLL 频率
- 2 组 BPWM 模块，每组都提供 6 路输出通道
- BPWM 输出/捕捉通道支持独立模式
- 从 1 到 4096 的 12 位预分频器
- 16 位精度的 BPWM 计数器，每个模块都有 1 个 BPWM 计数器
  - 向上、向下及上下/的计数器操作模式
- 每个 BPWM 管脚都支持屏蔽功能和三态使能
- 以下事件可以触发中断：
  - BPWM 计数器到 0 ，或周期值，或比较器的值
- 以下事件可以触发EADC0/1：
  - BPWM 计数器到 0或周期值或比较器的值

6.13.2.2 捕捉功能特性

- 多达 12 路 16 位精度的输入捕捉通道
- 支持向上或向下的捕捉条件
- 支持向上或向下的捕捉中断
- 支持向上或向下的捕捉触发计数器重载

	M48xID/M48xGA	M48xGC/M48xE8
触发EADC数目	1	2

表 6.13-1BPWM 特性比较表

6.13.3 框图

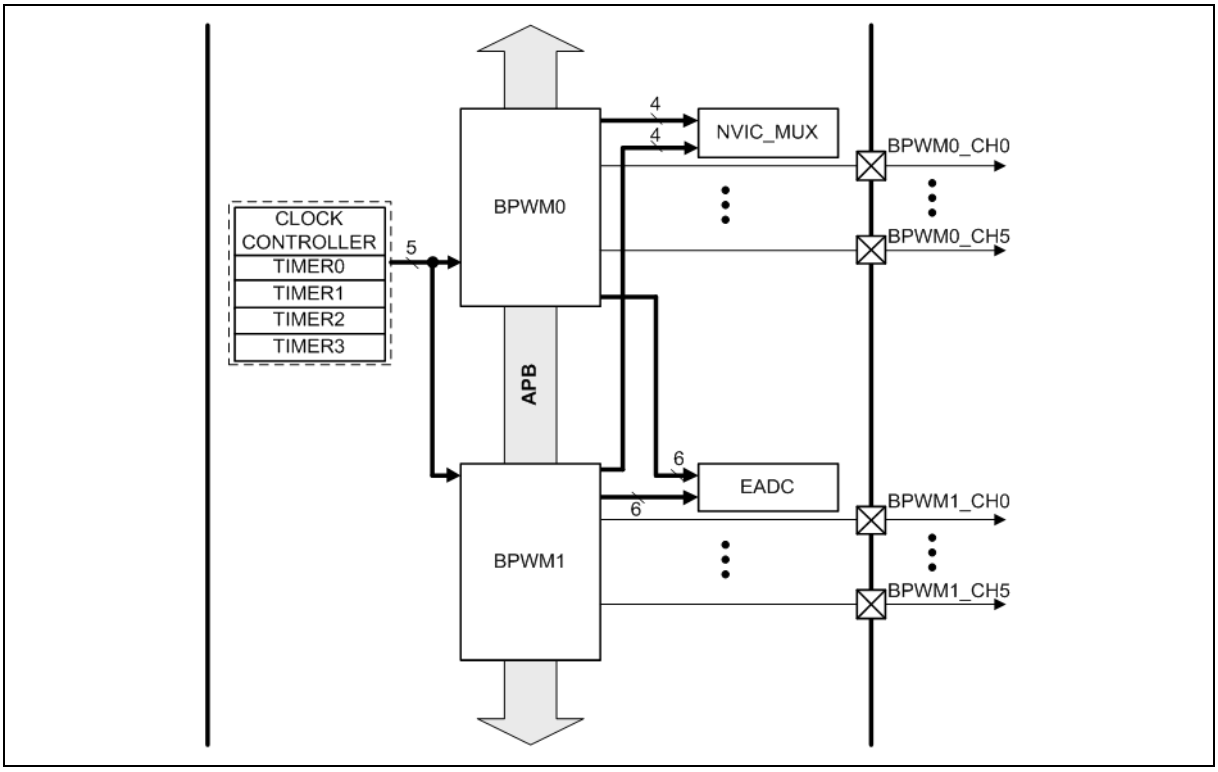


图 6.13-1BPWM 发生器框图

如图6.13-2所示，每个 BPWM 发生器只有一个时钟源输入，通过 BPWM_CLK0 的 ECLKSRC0 (BPWM_CLKSRC[2:0]) 寄存器决定由 BPWM 时钟或者定时器触发 BPWM 输出。通常，BPWM0 一定要设置成 BPWM0SEL (CLK_CLKSEL2[8]) 为 1 选择 PCLK0，BPWM1 一定要设置成 BPWM1SEL (CLK_CLKSEL2[9]) 为 1 选择 PCLK1。

如图6.13-3和表6.13-2所示，当运行在最大PLL时钟频率时，BPWM0和BPWM1时钟必须通过设置 BPWM0SEL (CLK_CLKSEL2[8]) 和 BPWM1SEL (CLK_CLKSEL2[9])为0选择为PLL时钟。

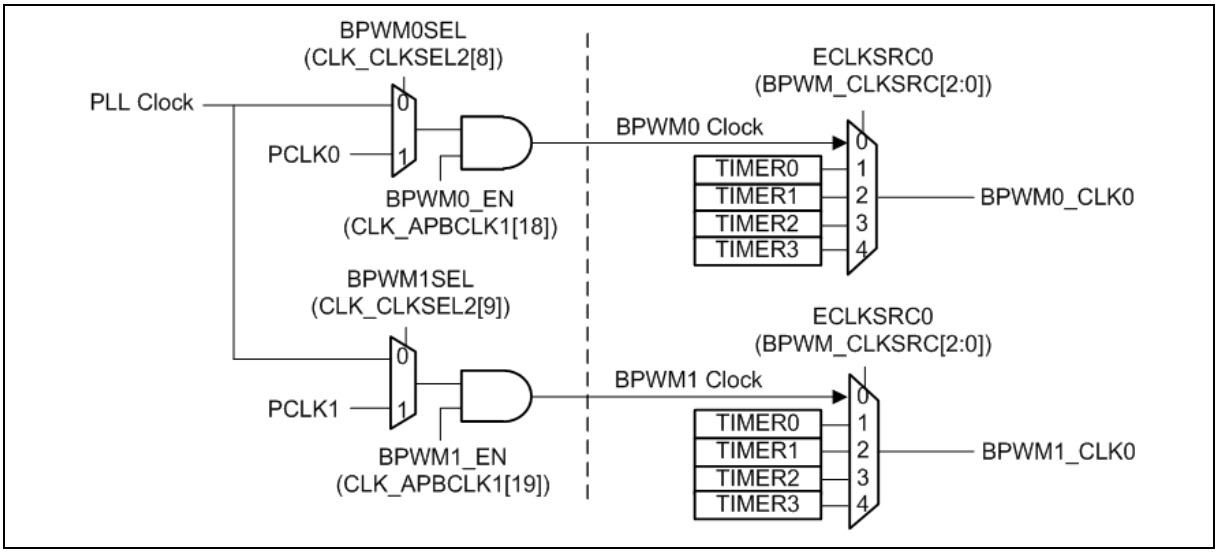


图 6.13-2BPWM 时钟源控制

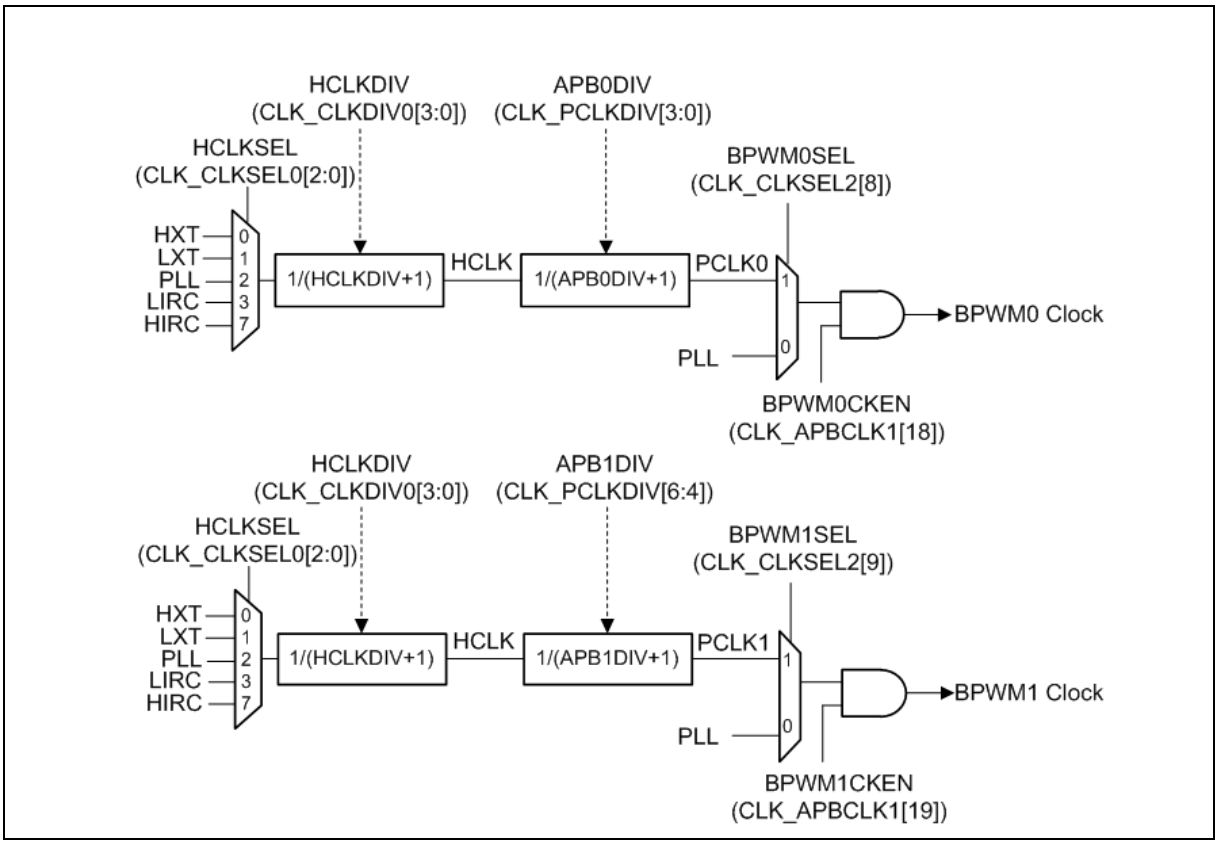


图 6.13-3BPWM 时钟源控制

频率比 PCLK:BPWM时钟	HCLK	PCLK	BPWM Clock	HCLKSEL CLK_CLKSEL0[2:0]	HCLKDIV CLK_CLKDIV0[3:0]	APBnDIV (CLK_PCLKDIVn[2+4n:4n]), N表示0或1	BPWMnSEL (CLK_CLKSEL2[N+8]), N表示0或1
1:1	HCLK	PCLK	PCLK	不关心	不关心	不关心	1
1:2	PLL	PLL/ 2	PLL	2	0	1	0
1:2	PLL/ 2	PLL/ 2	PLL	2	1	0	0

表 6.13-2BPWM 时钟源控制寄存器设置表

图6.13-4所示是BPWM独立模式的架构，所有的6路通道共享一个计数器。当计数器计数到0，PERIOD (BPWM_PERIOD[15:0])或者到比较器的值，将会产生事件。事件被传到对应的发生器来产生 BPWM 脉冲，中断信号或者触发EADC0/1开始转换的信号。输出控制则用来改变 BPWM 脉冲的输出状态。

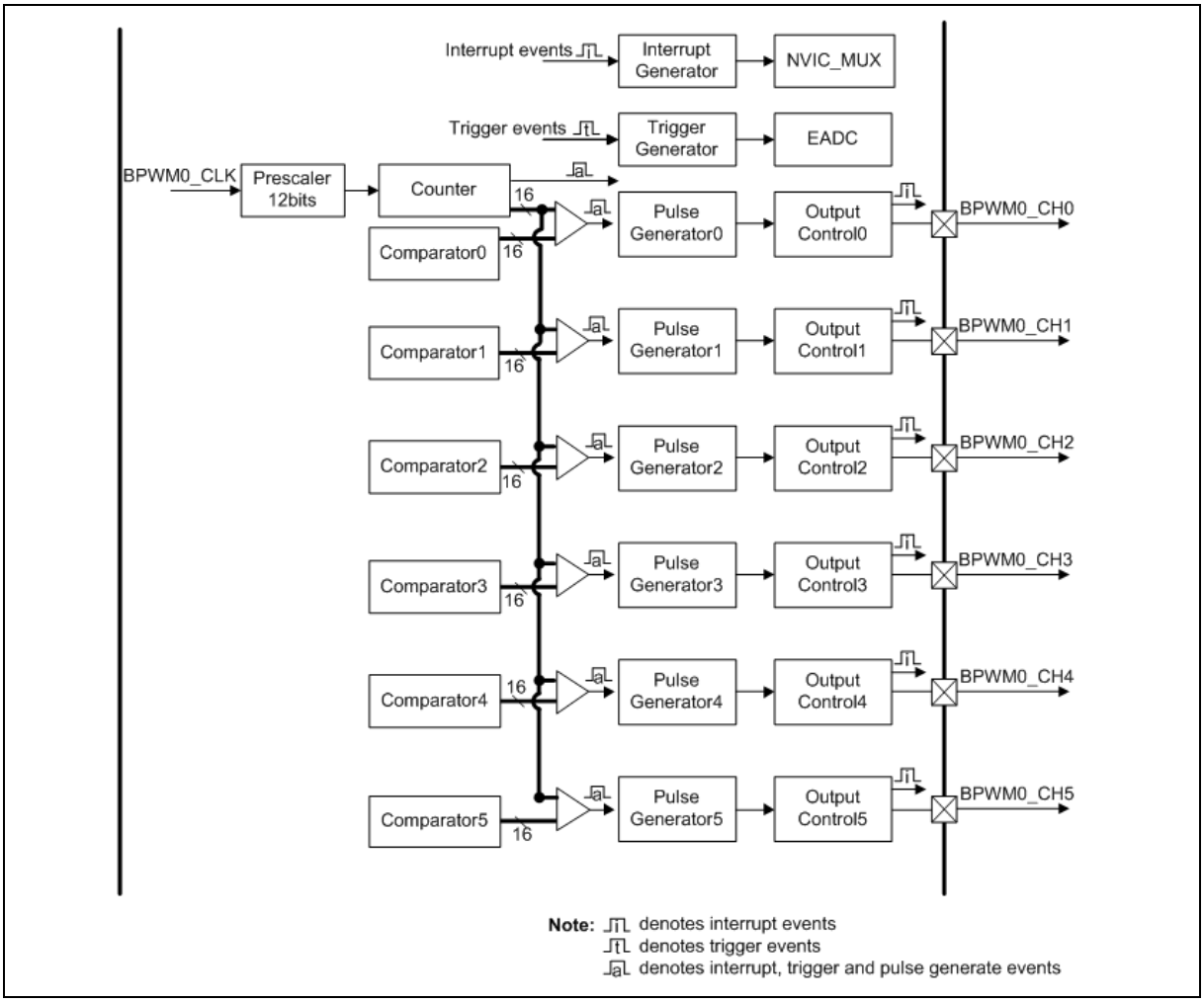


图 6.13-4BPWM 独立模式框架图

6.13.4 基本配置

6.13.4.1 BPWM0基本配置

- 时钟源配置
  - 在寄存器BPWM0SEL (CLK_CLKSEL2[8])选择 BPWM0 的时钟源
  - 在寄存器BPWM0CKEN (CLK_APBCLK1[18])使能 BPWM0 外设时钟
- 复位配置
  - 在寄存器BPWM0RST (SYS_IPRST2[18])复位 BPWM0 控制器
- 管脚配置

组	管脚名	GPIO	MFP
BPWM0	BPWM0_CH0	PA.11	MFP9
		PA.0, PG.14	MFP12
		PE.2	MFP13

	BPWM0_CH1	PA.10	MFP9
		PA.1, PG.13	MFP12
		PE.3	MFP13
	BPWM0_CH2	PA.9	MFP9
		PA.2, PG.12	MFP12
		PE.4	MFP13
	BPWM0_CH3	PA.8	MFP9
		PA.3, PG.11	MFP12
		PE.5	MFP13
	BPWM0_CH4	PF.5	MFP8
		PC.13	MFP9
		PA.4, PG.10	MFP12
		PE.6	MFP13
	BPWM0_CH5	PF.4	MFP8
		PD.12	MFP9
		PA.5, PG.9	MFP12
		PE.7	MFP13

6.13.4.2 BPWM1基本配置

- 时钟源配置
  - 在寄存器BPWM1SEL (CLK_CLKSEL2[9])选择 BPWM1 的时钟源
  - 在寄存器BPWM1CKEN (CLK_APBCLK1[19])使能 BPWM1 的外设时钟
- 复位配置
  - 在寄存器BPWM1RST (SYS_IPRST2[19])复位BPWM1控制器
- 管脚配置

组	管脚名	GPIO	MFP
BPWM1	BPWM1_CH0	PB.11	MFP10
		PF.3	MFP11
		PC.7, PF.0	MFP12
	BPWM1_CH1	PB.10	MFP10
		PF.2	MFP11
		PC.6, PF.1	MFP12
	BPWM1_CH2	PB.9	MFP10
		PA.12	MFP11
		PA.7	MFP12

	BPWM1_CH3	PB.8	MFP10
		PA.13	MFP11
		PA.6	MFP12
	BPWM1_CH4	PB.7	MFP10
		PA.14	MFP11
		PC.8	MFP12
	BPWM1_CH5	PB.6	MFP10
		PA.15	MFP11
		PE.13	MFP12

6.13.5 功能描述

6.13.5.1 BPWM预分频器

BPWM 预分频器用来分频时钟源，预分频器计数 CLKPSC+1 次，而 BPWM 计数器则只计数 1 次。预分频的值是由 CLKPSC (BPWM_CLKPSC[11:0])设置。如图6.13-5是 BPWM 通道 0 CLKPSC 波形示例图。预分频计数器会在下一个预分频向下计数的开始时重载 CLKPSC 的值。

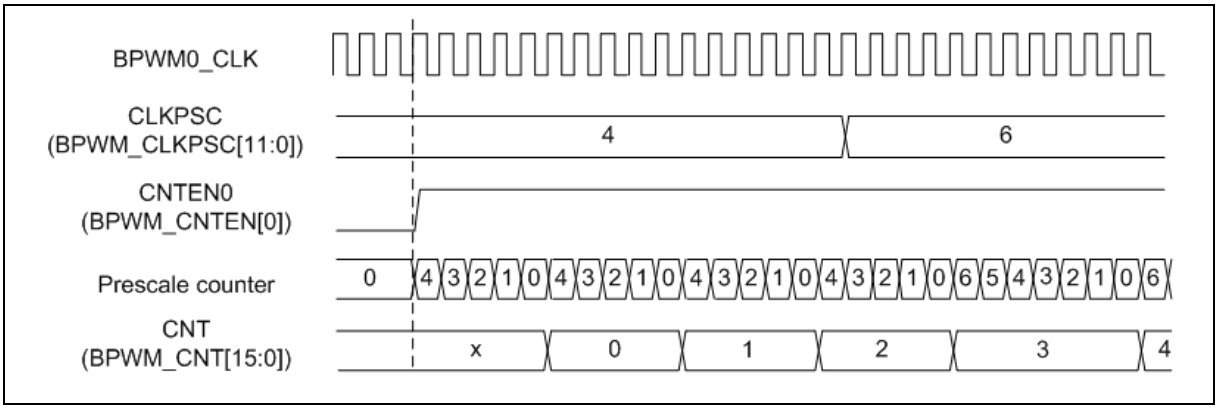


图 6.13-5BPWM_CH0 CLKPSC 波形

6.13.5.2 BPWM计数器

BPWM 的计数器支持 3 种计数操作方式：向上、向下和上下计数。

对于 BPWM 通道 0 ，可以设置当前分频计数器向下计数到0时，CNTCLR0（BPWM_CNTCLR [0]）会清空CNT（BPWM_CNT [15：0]）到0x00。

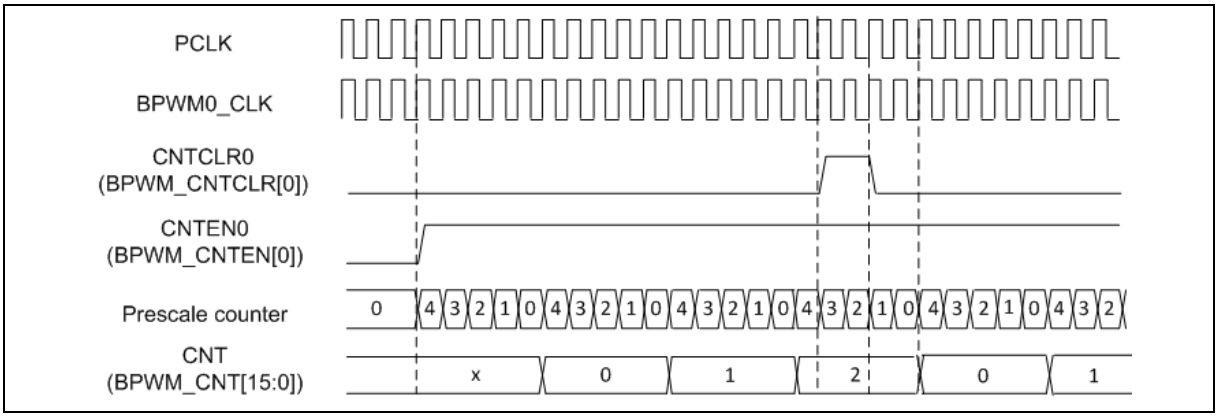


图 6.13-6 BPWM 计数器清空波形

6.13.5.3 向上计数方式

在向上计数操作中，16 位的 BPWM 计数器从 0 开始向上计数到 PERIOD (BPWM_PERIOD) 完成一个 BPWM 周期。从 CNT (BPWM_CNT[15:0]) 寄存器可以读到当前的计数值。当计数器计数到 0 时，BPWM 发生器会产生零位事件；计数器计数到 PERIOD 时则会产生周期位置事件。如图6. 13-7所示是向上计数方式下的周期时间示例，BPWM 的周期时间 = (PERIOD+1) * (CLKPSC+1) * BPWMx_CLK。

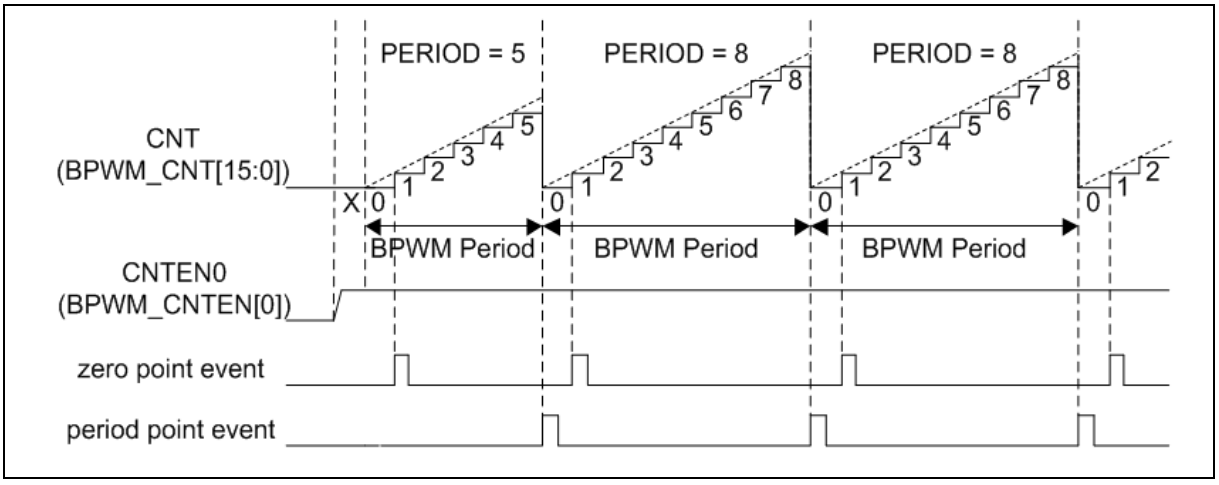


图 6.13-7BPWM 向上计数模式

6.13.5.4 向下计数方式

在向下计数操作中，16 位的 BPWM 计数器从 PERIOD 开始向下计数到 0 完成一个 BPWM 周期。从 CNT (BPWM_CNT[15:0]) 寄存器可以读到当前的计数值。当计数器计数到 0 时，BPWM 发生器会产生零位事件；计数器计数到 PERIOD 时则会产生周期位置事件。如图6.13-8所示是向下计数方式下的周期时间示例，BPWM 的周期时间 = (PERIOD+1) * (CLKPSC+1) * BPWMx_CLK。

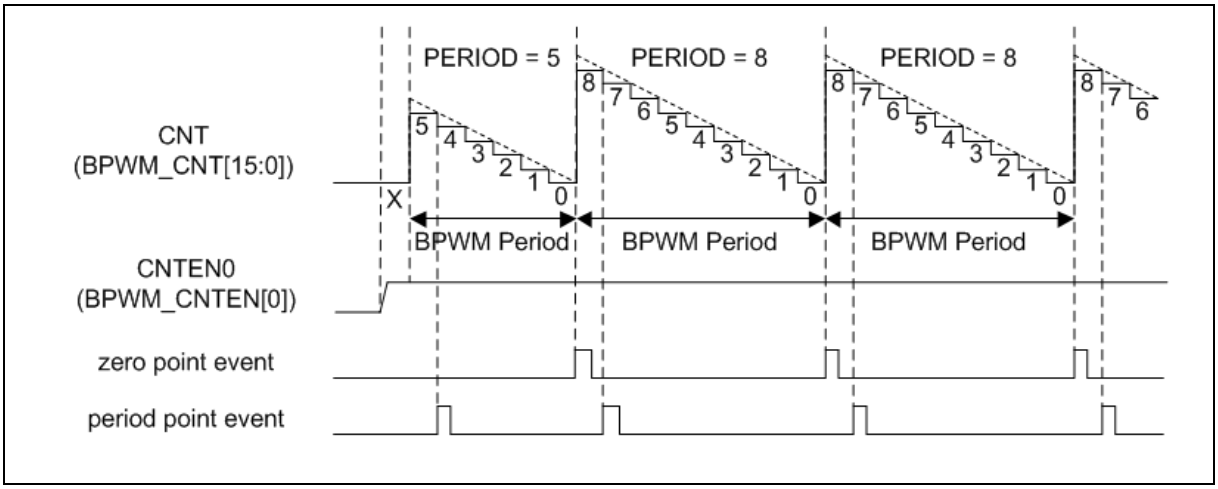


图 6.13-8BPWM 向下计数方式

6.13.5.5 上下计数方式

在上下计数操作中，16 位的 BPWM 计数器先从 0 开始向上计数到 PERIOD，再从 PERIOD 开始向下计数到 0 完成一个 BPWM 周期。从 CNT (BPWM_CNT[15:0]) 寄存器可以读到当前的计数值。当计数器计数到 0 时，BPWM 发生器会产生零位事件；计数器计数到 PERIOD 时则会产生中位事件。如图 6.13-9所示是上下计数方式下的周期时间示例，BPWM 的周期时间 = (2*PERIOD) * (CLKPSC+1) * BPWMx_CLK，寄存器 DIRF (BPWM_CNT[16]) 是计数方向的标志位，向上计数时为 1，向下计数时为 0。

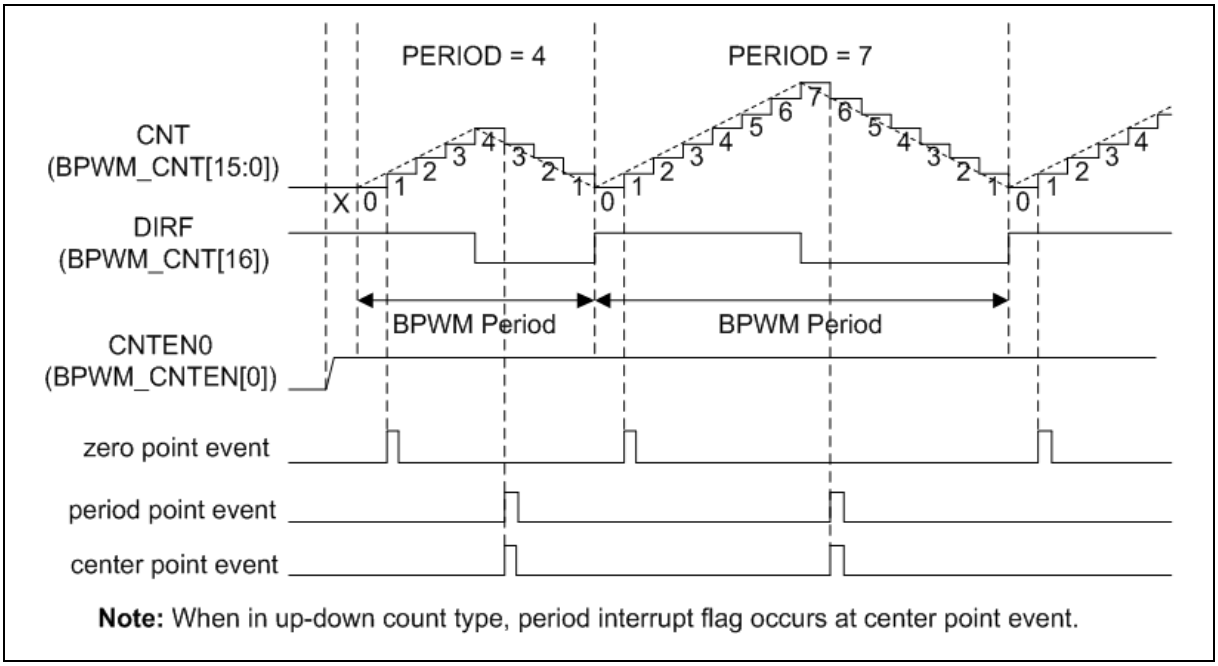


图 6.13-9BPWM 上下计数方式

6.13.5.6 BPWM比较器

CMPDAT (BPWM_CMPDATn[15:0]) 是 BPWM 通道 n 的基本比较器寄存器，每路通道都有且仅有一个 CMPDAT。CMPDAT 的值不断与计数器的值做比较。当计数器计数到比较器寄存器的值时， BPWM

会产生一个事件并用以之产生 BPWM 脉冲，中断或者EADC0/1触发信号。如图6.13-10所示，在上下计数方式里会有 2 个事件产生。

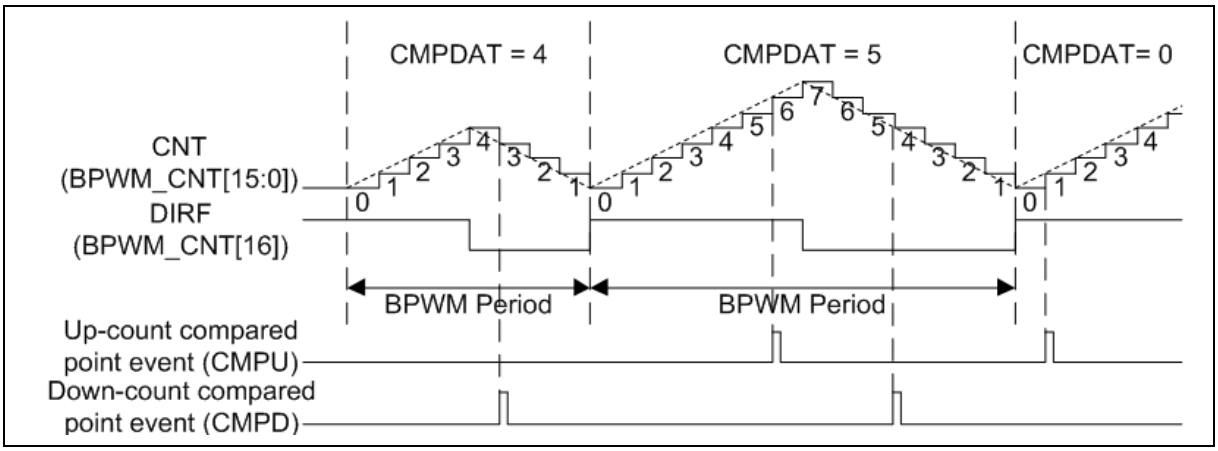


图 6.13-10 上下计数方式时 BPWM CMPDAT 事件

6.13.5.7 周期载入模式

周期载入模式是默认的载入模式。在所有载入模式里它是优先级最低的。当周期结束时，PERIOD 和 CMPDAT 都会被载入到对应的缓存。例如，在向上计数操作里 BPWM 从 0 向上计数到 PERIOD 后，或者向下计数操作里从 PERIOD 向下计数到 0 后，亦或上下计数操作里从 0 计数到 PERIOD 再向下计数到 0 后，都会进行重载。

图6.13-11所示是向上计数方式的重载时机，图示里 PERIOD DATA0 为 PERIOD 的初始值，PERIOD DATA1 为 PERIOD 第一次由软件更新的值，以此类推，CMPDAT 也按这种方式标注。图6.13-11的具体步骤顺序参见下面的描述。用户可以查看 BPWM 的周期和 CMPU 事件来了解 PERIOD 和 CMPDAT 的更新条件。

1. 在位置 1，软件写 CMPDAT DATA1 到 CMPDAT
2. 在位置 2，PWM 周期结束时，硬件载入 CMPDAT DATA1 到 CMPBUF
3. 在位置 3，软件写 PERIOD DATA1 到 PERIOD
4. 在位置 4，PWM 周期结束时，硬件载入 PERIOD DATA1 到 PBUF
5. 在位置 5，软件写 PERIOD DATA2 到 PERIOD
6. 在位置 5，PWM 周期结束时，硬件载入 PERIOD DATA2 到 PBUF.

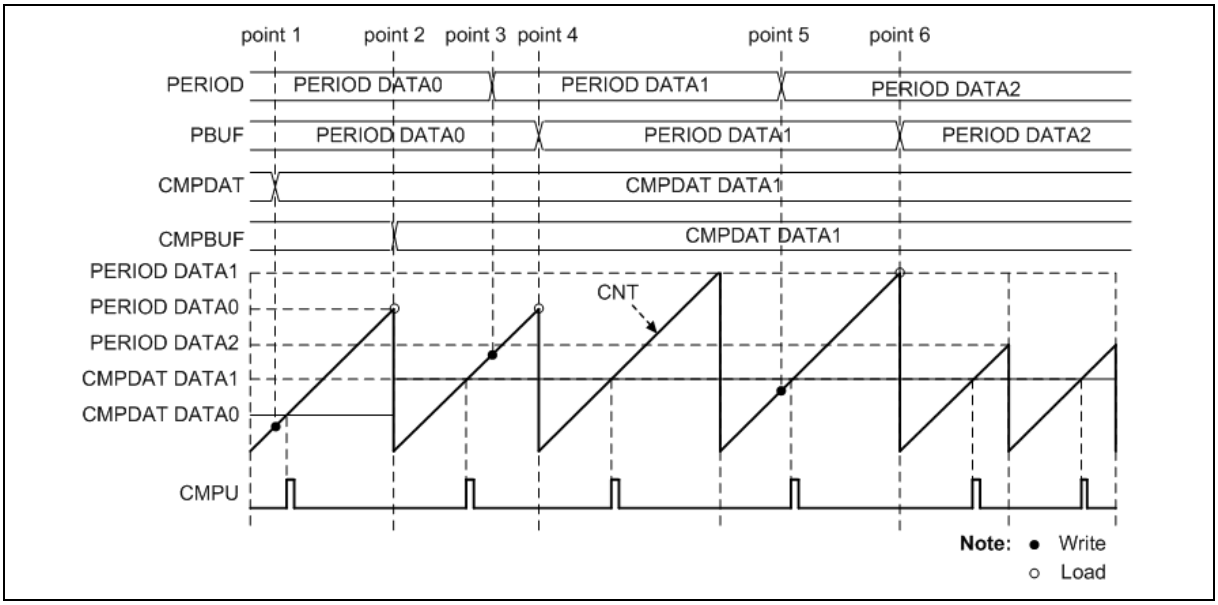


图 6.13-11 向上计数方式下的周期载入模式

6.13.5.8 立即载入模式

如果 BPWM 通道 n 在寄存器IMMLDENn (BPWM_CTL0[21:16])里对应的位置 1 时，当软件更新 PERIOD 或者 CMPDAT 时，硬件会立刻载入 PERIOD 和 CMPDAT 到缓存。如果更新的 PERIOD 值小于当前的计数器值，计数器会一直计数到 0xFFFF，当计数器计数到 0xFFFF 且预分频值计数到 0，寄存器CNTMAXF0(BPWM_STATUS[0])置位，接着计数器回环计数。立即载入模式拥有最高的优先级。如果 IMMLDENn 被置位，其他针对通道 n 的模式设置都会无效。下面是对图6.13-12示例的文字说明：

1. 在位置 1 软件写 CMPDAT DATA1，硬件立即重载 CMPDAT DATA1 到 CMPBUF
2. 在位置 2 软件写入一个比当前计数器大的值 PERIOD DATA1，计数器会一直计数到 PERIOD DATA1 的值以完成一个载入周期.
3. 在位置 3 软件写入一个比当前计数器小的值 PERIOD DATA2，计数器会一直计数到最大值 0xFFFF，再从 0 开始回环计数到 PERIOD DATA2 以完成周期载入

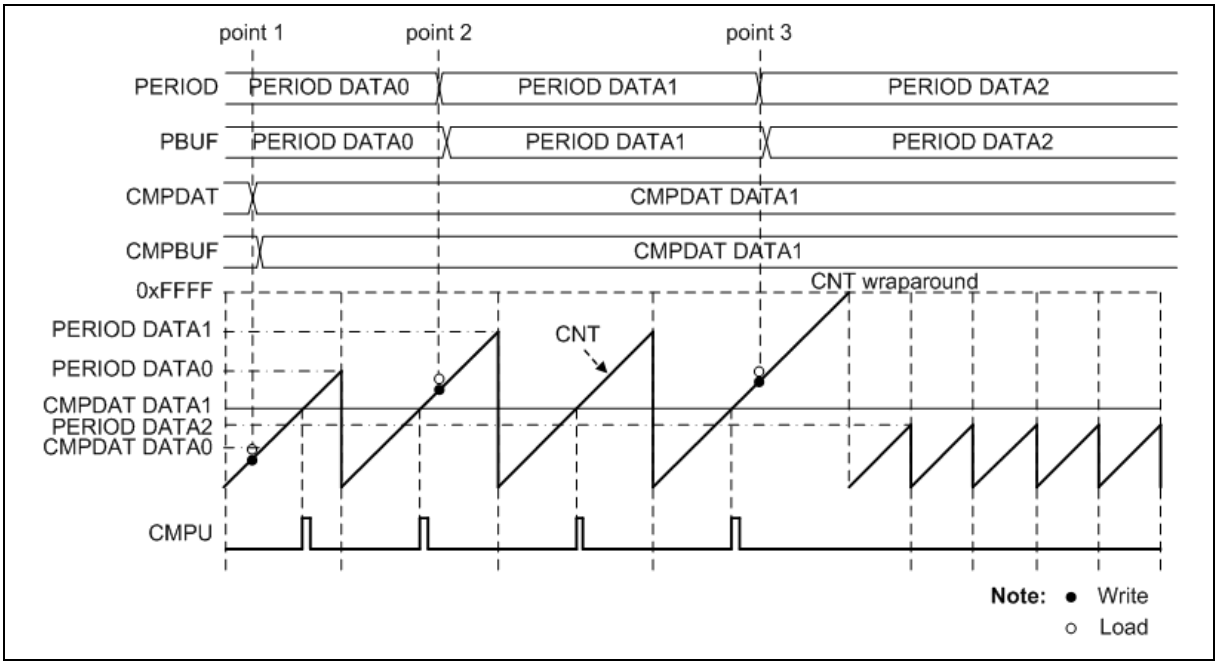


图 6.13-12 向上计数方式下的立即载入模式

6.13.5.9 中心载入模式

如果 BPWM 通道 n 在寄存器 CTRLDn (BPWM_CTL0[5:0]) 里对应的位置 1 且处于上下计数方式时，CMPDAT 的值会在周期中间被载入到 CMPBUFn，即在计数器计数到 PERIOD 时载入。PERIOD 的载入时间同周期载入模式相同。下面是对图6. 13-13示例的文字说明：

1. 在位置 1 软件写 CMPDAT DATA1
2. 在位置 2 PWM 周期中间位置，硬件载入 CMPDAT DATA1 到 CMPBUF
3. 在位置 3 软件写 PERIOD DATA1
4. 在位置 4 PWM 周期末位置，硬件载入 PERIOD DATA1 到 PBUF
5. 在位置 5 软件写 CMPDAT DATA2
6. 在位置 6 PWM 周期中间位置，硬件载入 CMPDAT DATA2 到 CMPBUF
7. 在位置 7 软件写 PERIOD DATA2
8. 在位置 8 PWM 周期末位置，硬件写 PERIOD DATA2 到 PBUF

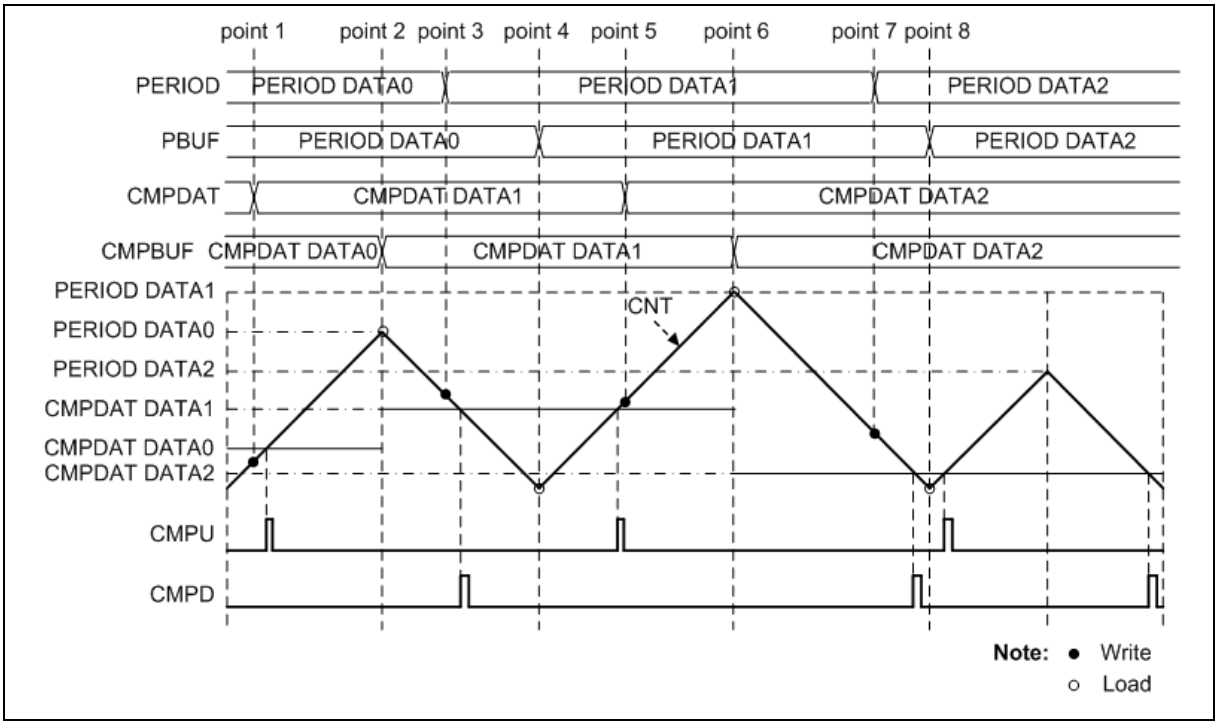


图 6.13-13 上下计数方式下的中间载入模式

6.13.5.10 BPWM脉冲发生器

BPWM 脉冲发生器使用计数器和比较器事件来产生 BPWM 脉冲。对应的事件有：零位、向上计数器方式和向下计数器方式的周期位置、上下计数方式的中心位置以及三种模式下计数器计数到比较器值的位置。对于上下计数方式，有两个计数器计数到比较器值的位置，一是向上计数的过程，二是向下计数的过程。

通过 BPWM_WGCTL0 和 BPWM_WGCTL1 寄存器可以设定每个位置点 BPWM 波形：无动作 (X)、拉低 (L)、拉高 (H) 或者触发 (T)。如图6.13-14，通过这些位置点可以轻易的产生不对称的 BPWM 脉冲及多样的波形。图中，比较器 n 用于产生 BPWM 脉冲，n 指通道 0 ~ 5；CMPU 指向上计数时 CNT 计数到 CMPDAT，CMPD 指向下计数时 CNT 计数到 CMPDAT。

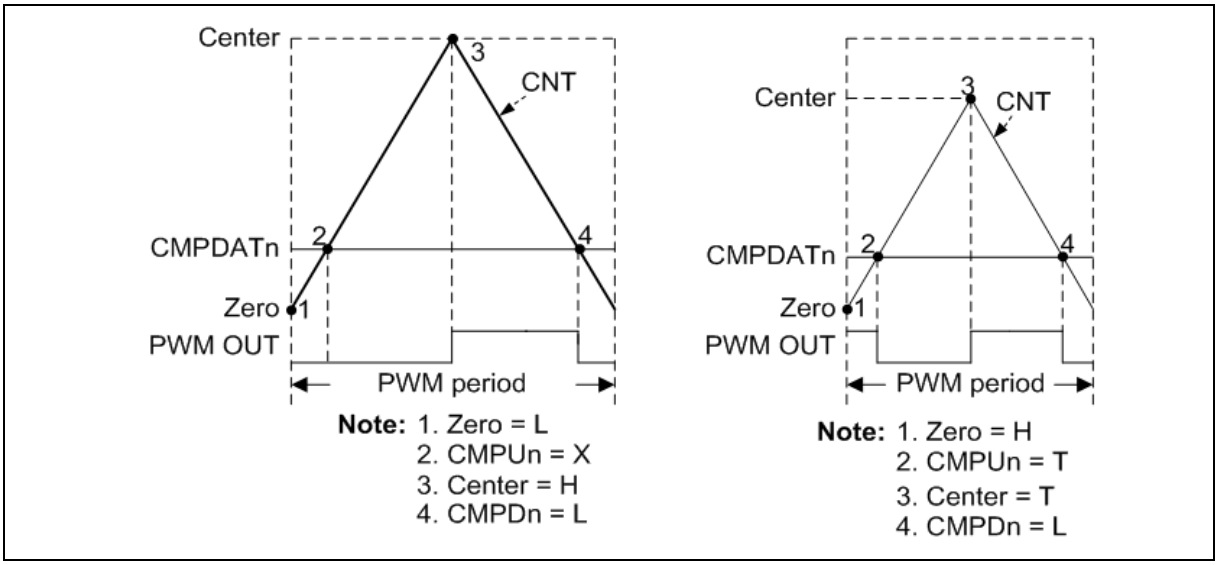


图 6.13-14 BPWM 脉冲发生 (左：不对称脉冲，右：多样脉冲)

多个事件有时候会发生在同一时刻。这时候，不同的计数方式的事件的优先级如下所示：(表6.13-3)是向上计数方式的优先级、(表6.13-4)是向下计数方式的优先级而(表6.13-5)是上下计数方式的优先级。如图6. 13-15所示，通过事件优先级，用户可以轻易产生从 0% 到 100% 占空比的脉冲。

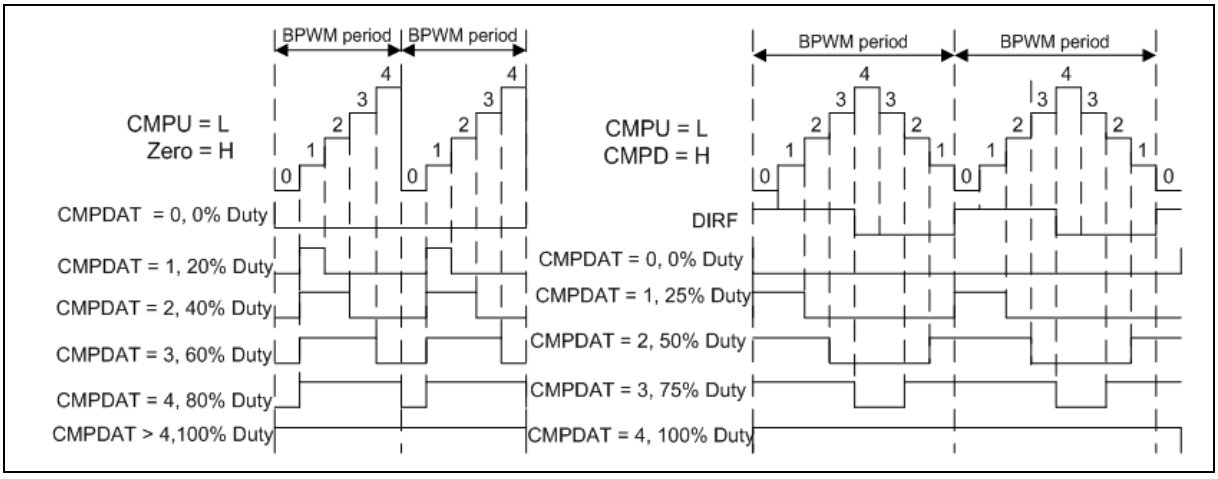


图 6.13-15 BPWM0% 到 100% 脉冲产生 (左:上计数器模式, 右：上下计数器模式)

优先级	向上计数事件
1 (最高)	周期事件(CNT = PERIOD)
2	比较事件(CNT = CMPUn)
3 (最低)	零位事件 (CNT = 0)

表 6.13-3 向上计数方式下的 BPWM 脉冲发生事件优先

优先级	向下计数方式

1 (最高)	零位事件 (CNT = 0)
2	比较事件 (CNT = CMPDn)
3 (最低)	周期事件 (CNT = PERIOD)

表 6.13-4 向下计数方式的 BPWM 脉冲发生事件优先级

优先级	向上事件	向下事件
1 (最高)	向上比较事件(CNT = CMPUn)	向下比较事件(CNT = CMPDn)
2(最低)	零点事件(CNT = 0)	周期 (中心) 事件 (CNT =PERIOD)

表 6.13-5 上下计数方式的 BPWM 脉冲发生事件优先级

6.13.5.11 同步功能

同时使能 BPWM 和 PWM 计数器需要：设定 BPWM 同步开始控制寄存器 (BPWM_SSCTL[0]) 以使能希望开始同步计数的计数器通道，设定寄存器 SSRC (BPWM_SSCTL[9:8]) 来选择同步开始源，然后再设定 BPWM 同步开始触发寄存器 CNTSEN (BPWM_SSTRG[0])。

6.13.5.12 BPWM输出控制

BPWM 脉冲发生后，可以分三步来控制 BPWM 通道输出。这三个步骤如图6.13-16分别是：屏蔽、管脚极性和输出使能。

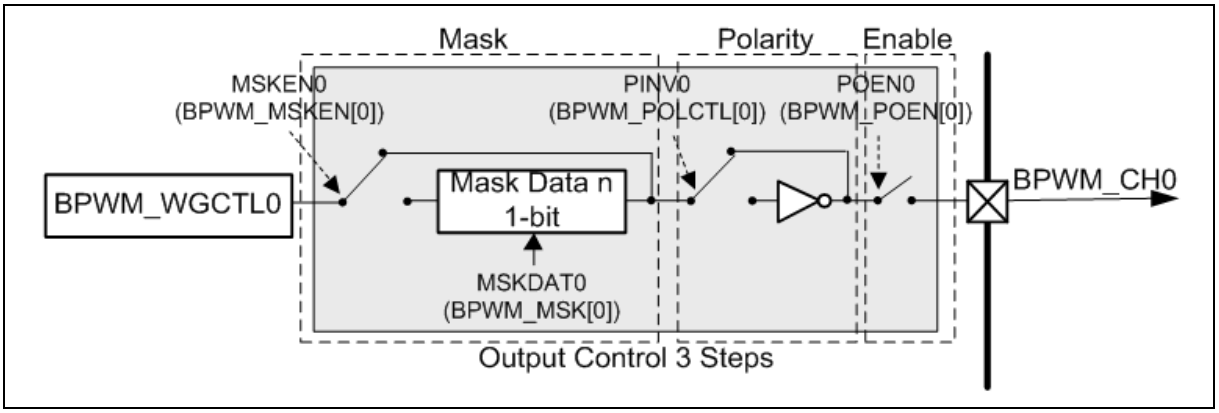


图 6.13-16BPWM_CH0 输出控制的 3 步

6.13.5.13 BPWM屏蔽输出功能

每个 BPWM 的输出通道都可以通过人为地修改 BPWM 屏蔽使能控制寄存器 (BPWM_MSKEN) 和 BPWM 屏蔽数据寄存器 (BPWM_MSK) 对应的位，来输出独立于占空比周期比较单元的特定逻辑状态。在驱动类似 BLDC 电机的电力整流电机时，BPWM 的屏蔽位会非常有用。BPWM_MSKEN 寄存器包含 6 位：MSKENN (BPWM_MSKEN[5:0]) 控制哪路 BPWM 通道输出会被修改，这 6 位是高有效。BPWM_MSK 寄存器包含 6 位，MSKDATn (BPWM_MSK[5:0]) 控制被 MSKDAT 对应位屏蔽的 BPWM 通道输出状态。如图6.13-17所示是通过 BPWM 屏蔽功能修改输出的例子。

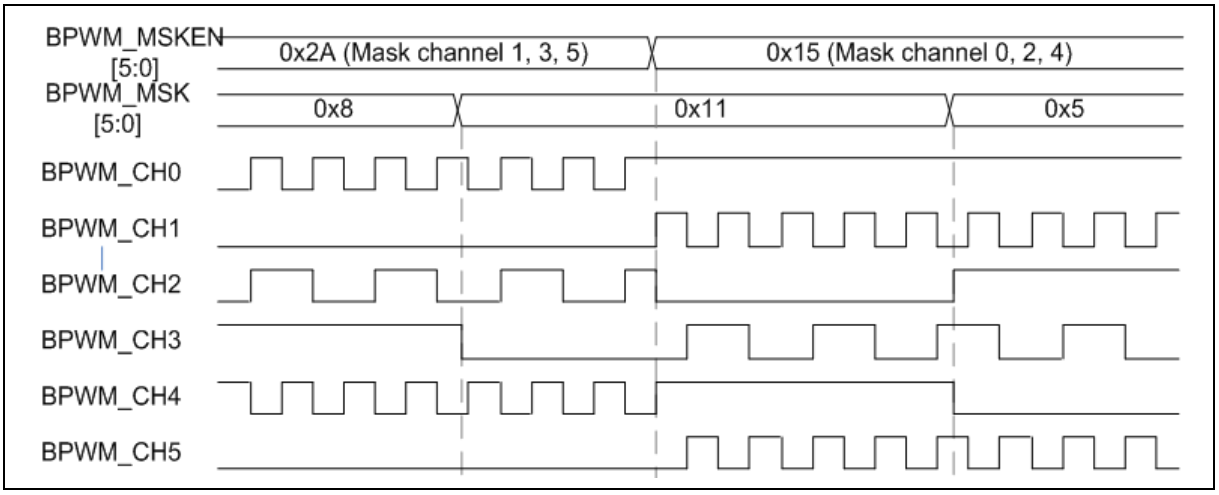
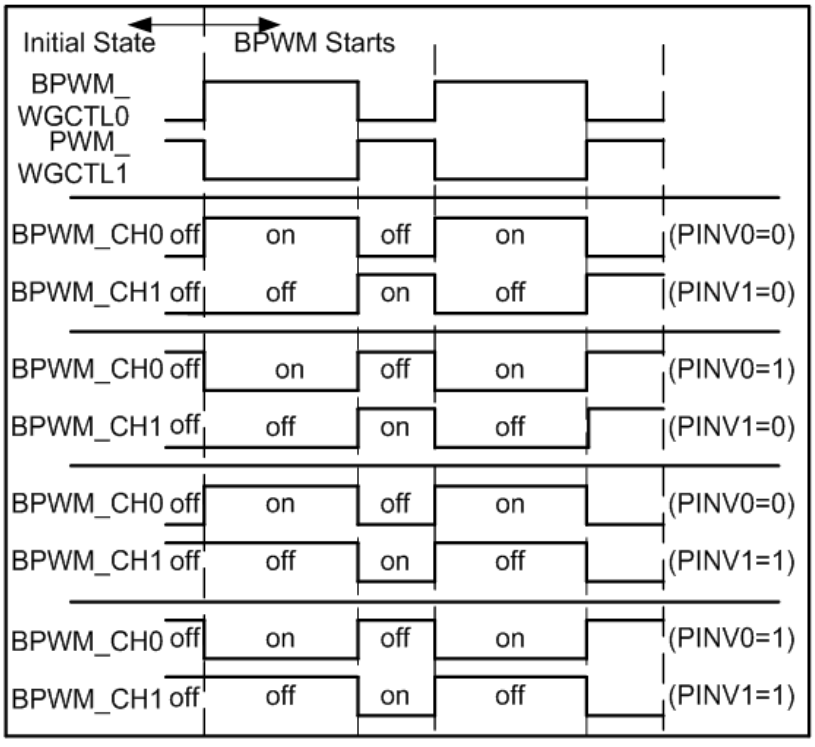


图 6.13-17 屏蔽功能波形说明

6.13.5.14 极性控制

从 BPWM_CH0 到 BPWM_CH5 每个 BPWM 端口都有独立的极性控制模块来配置 BPWM 输出的有效状态的极性。BPWM 默认输出高态有效。也就是说，BPWM 关闭状态为低，打开状态为高。每个独立的 BPWM 通道都可以通过配置 BPWM 阴极极性控制寄存器 (BPWM_POLCTL) 来自由配置有效状态。如图6.13-18是应用不同极性配置前 BPWM 的初始状态。



Note: PINVx: Negative Polarity control bits; It controls the PWM output initial state and polarity, x denotes 0 or 1.

图 6.13-18 初始状态和极性控制

6.13.5.15 BPWM中断发生器

如图6.13-19，每个 BPWM 都有 2 个独立的中断。

BPWM 中断 (BPWM_INT) 来自于对应的 BPWM 事件。计数器可以产生零位中断标志 ZID0 (BPWM_INTSTS0[0]) 和周期位置中断标志 PIF0 (BPWM_INTSTS0[8])。当 BPWM 通道 n 的计数值和 BPWM_CMPDATn 里的比较值相同时，会根据计数的方向触发不同的中断标志。如果发生在向上计数时，向上中断标志 CMPUIFn (BPWM_INSTS0[21:16]) 会被置位；如果发生在向下计数时，向下中断标志 CMPDIFn (BPWM_INSTS0[29:24]) 会被置位。此时如果对应的中断使能位置位，触发事件会产生相应的中断信号。

另一个中断是捕捉中断 (CAP_INT)。在 NVIC 中和 BPWM 中断共用 BPWM_INT 向量。CAPRIFn (BPWM_CAPIF[5:0]) 触发及上升沿捕捉中断使能位 CAPRIENn (BPWM_CAPIEN[5:0]) 置 1 都会触发 CAP_INT；或者，下降沿条件下，下降沿中断使能位 CAPFIENn (BPWM_CAPIEN[13:8]) 置 1 时会触发 CAPFIFn (BPWM_CAPIF[13:8]。

图6.13-19所示是 BPWM 中断的架构。

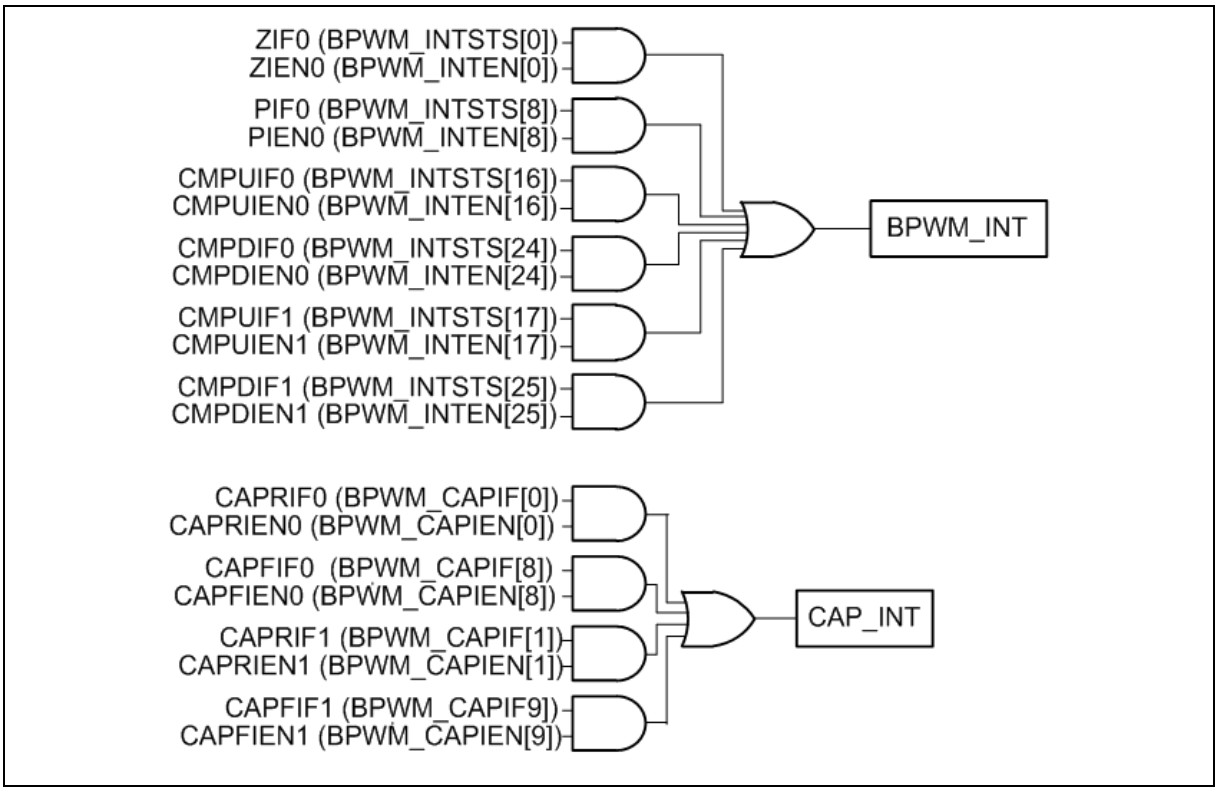


图 6.13-19BPWM_CH0 和 BPWM_CH1 对中断架构图

6.13.5.16 BPWM触发EADC0/1发生器

BPWM 可以作为 EADC0/1转换的触发源。每个 BPWM 通道对共享一个触发源。设置 TRGSELn 选择触发源，TRGSELn 指 TRGSEL0、TRGSEL1 …和 TRGESL5，分别对应到 BPWM_EADCTS0[3:0]、BPWM_EADCTS0[11:8] 、 BPWM_EADCTS0[19:16] 、 BPWM_EADCTS0[27:24] 、

BPWM_EADCTS1[3:0] 和 BPWM_EADCTS1[11:8]。设置 TRGENn 使能触发输出到 EADC0/1，TRGENn 指 TRGEN0、TRGEN1 …和 TRGEN5，分别对应到 BPWM_EADCTS0[7]、BPWM_EADCTS0[15]、BPWM_EADCTS0[23]、BPWM_EADCTS0[31]、BPWM_EADCTS1[7] 和 BPWM_EADTS1[15]。其中，n (n = 0,1,…5) 对应 BPWM 通道数。

一对 BPWM 通道有 7 个事件可以选作触发源。如图6.13-20以 BPWM_CH0 和 BPWM_CH1 为例。设定 PERIOD 和 CMPDAT 可以配置在不同时间点 BPWM 触发EADC0/1开始转换。图6.13-22上下计数方式下触发EADC0/1的时序波形。

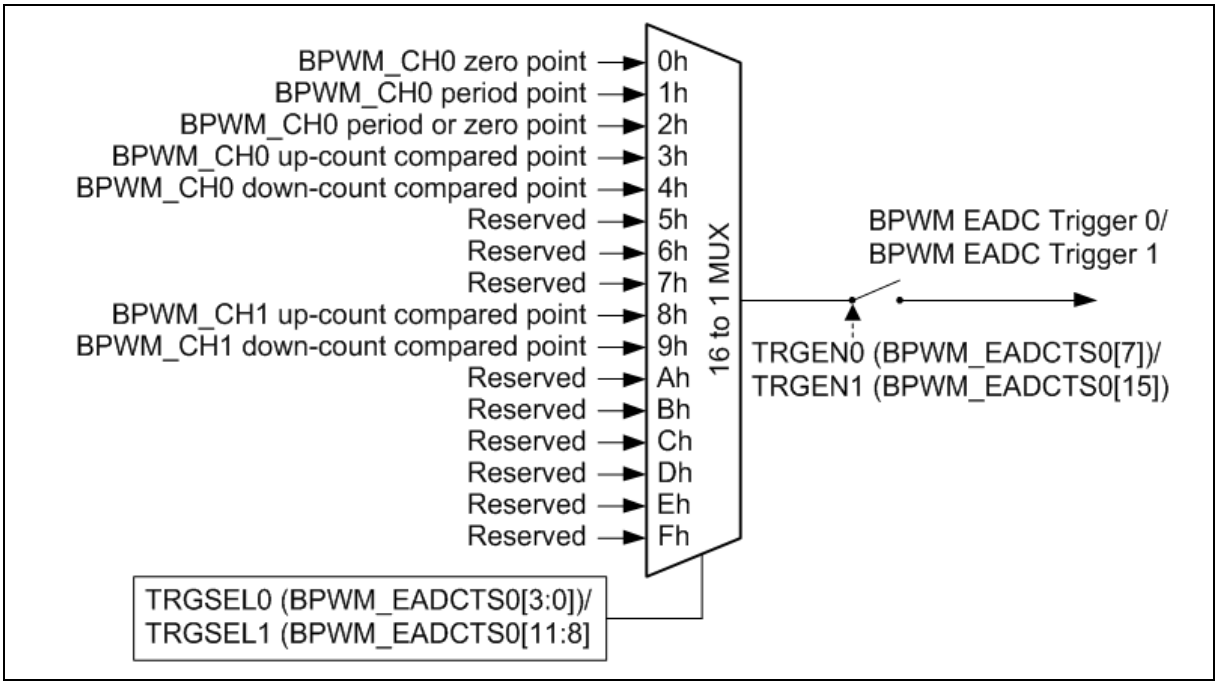


图 6.13-20BPWM_CH0 和 BPWM_CH1 触发 EADC0/1 源的框架图

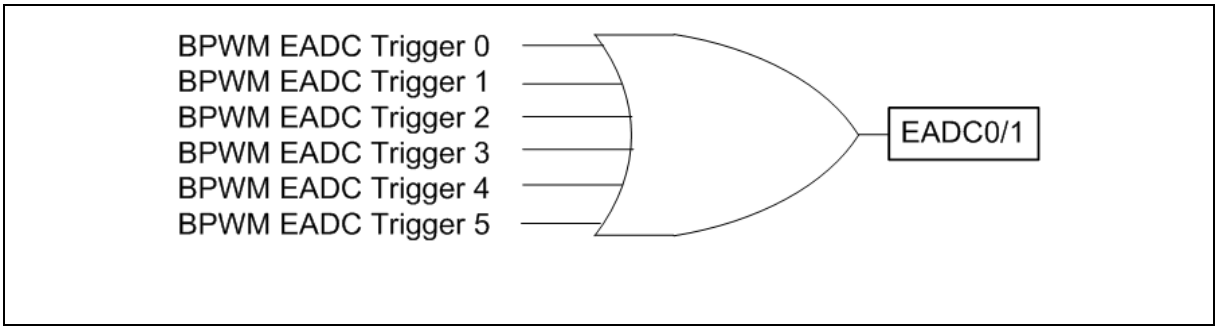


图 6.13-21BPWM CH0~ CH5 触发 EADC0/1 的框架图

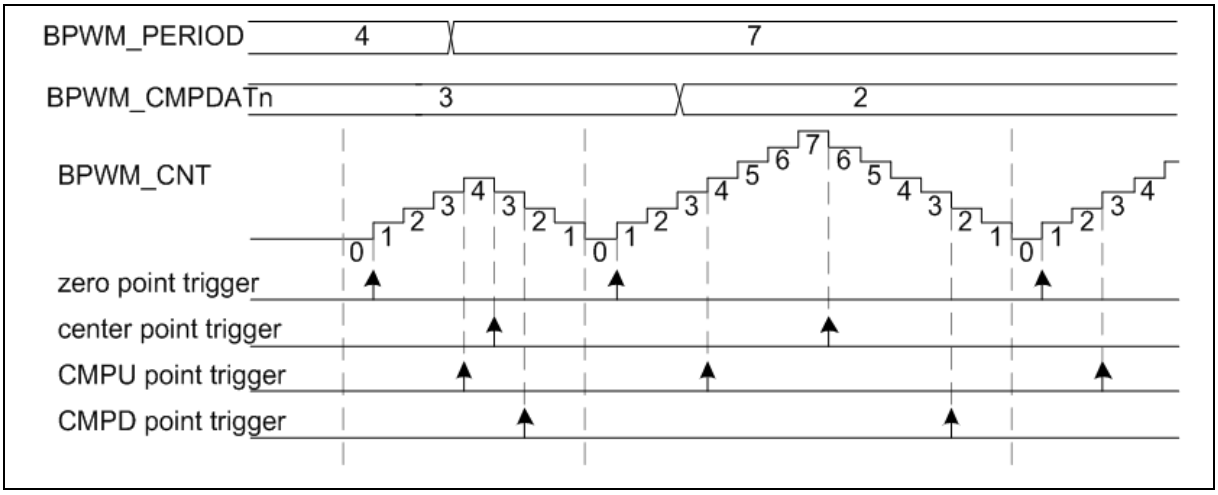


图 6.13-22 上下计数方式下 BPWM 触发 EADC0/1 的时序波形图

6.13.5.17 捕捉操作

输入捕捉通道和 BPWM 输出通道共享相同的管脚和计数器。这里，计数器可以工作在向上或者向下计数方式。捕捉功能会占有 BPWM 计数器，在输入通道有上升沿时锁存 BPWM 计数器到寄存器 RCAPDATn (BPWM_RCAPDATn[15:0])，在输入通道有下降沿时锁存 BPWM 计数器到寄存器 FCAPDATn (BPWM_FCAPDATn[15:0])。捕捉功能在对应的上升沿或者下降沿使能位使能时，上升沿或者下降沿发生时会产生 CAP_INT (使用 BPWM_INT 向量) 中断，其中 CAPRIENn (BPWM_CAPIEN[5:0]) 是上升沿的使能位，CAPFIENn (BPWM_CAPIEN[13:8]) 是下降沿的使能位。当上升沿锁存发生时，对应的 BPWM 计数器会重载 BPWM_PERIODD 的值，这部分由 RCRLDENn 或者 FCRLDENn 决定 (RCRLDENn 和 FCRLDENn 分别在寄存器 BPWM_CAPCTL[21:16] 和 BPWM_CAPCTL[29:24] 内。值得注意的是，需要使能捕捉通道 n 对应的 CAPINENn (BPWM_CAPINEN[5:0]) 寄存器位来配置管脚的捕捉功能。图6. 13-23是通道 0 的捕捉框图。

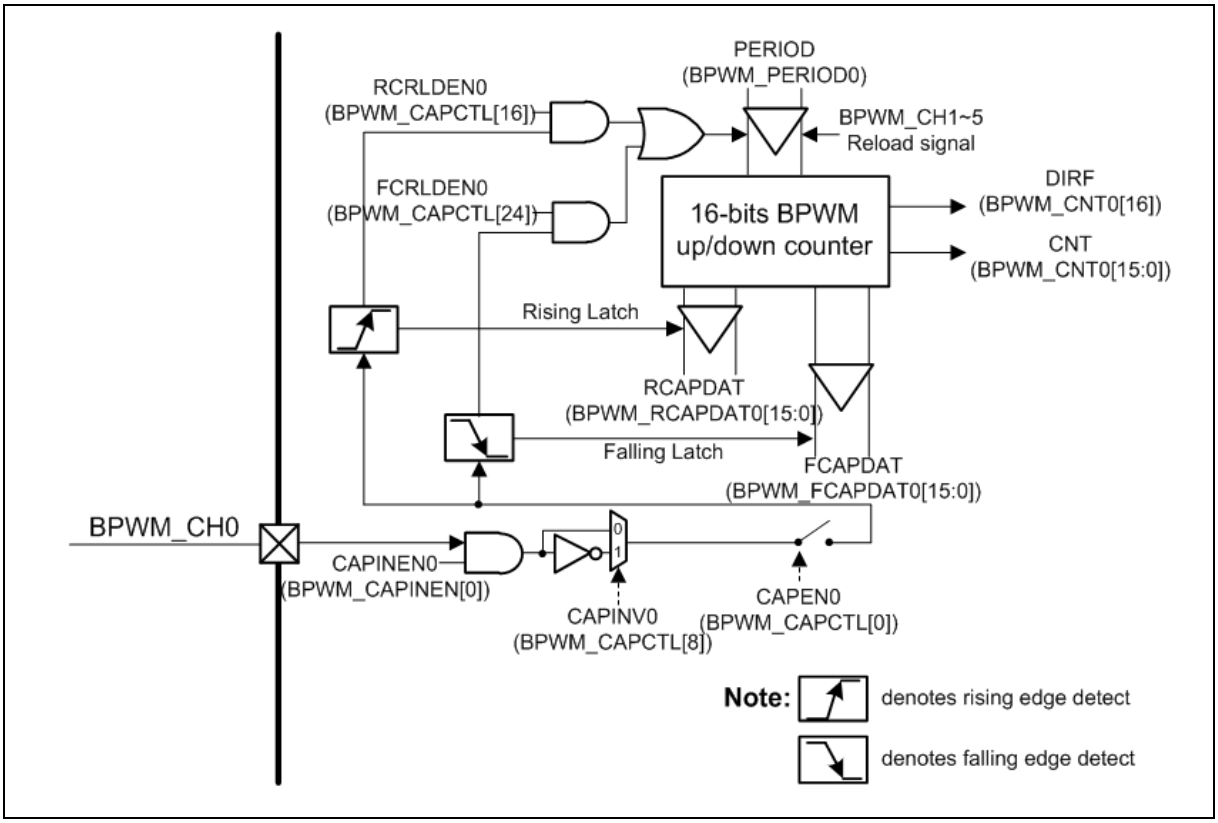


图 6.13-23BPWM_CH0 捕捉框图

图6. 13-24是捕捉功能的时序示例。此时，捕捉计数器处于 BPWM 向下计数方式，PERIOS 为 8，计数器会从 8 到 0 向下计数。当检测到捕捉管脚的下降沿时，捕捉功能会锁存计数器的值到 BPWM_RCAPDATn。在时序图中，第一次检测到下降沿时，由于 FCRLDENn 被使能，捕捉功能会从 PERIOD 的配置值重载计数器的值。但在第二次时，由于禁止了 FCRLDENn，下降沿就不会引起计数器重载了。在这个例子里，由于使能了 RCRLDENn，计数器的值在上升沿时也会被重载。

另外，对于向上计数方式，计数器的值会被重载为 0 再向上计数到 PERIOD 的值。需要特别注意的是，所有通道共享一个计数器，所以计数器重载的时间点也是有所有通道的重载信号一起控制的。

图6.13-24还展示了中断和中断标志发生的时序。当通道 n 检测到了上升沿时，对应的 CAPRIFn (BPWM_CAPIF[5:0]) 位由硬件置位。类似的，当通道 n 检测到了下降沿时，对应的 CAPFIFn (BPWM_CAPIF[13:8]) 位由硬件置位。CAPRIFn (BPWM_CAPIF[5:0]) 和 CAPFIFn (BPWM_CAPIF[13:8]) 由软件写 1 清 0。如果 CAPRIFn (BPWM_CAPIF[5:0]) 且 CAPRIENn 使能，捕捉功能会产生一个中断。如果 CAPFIFn (BPWM_CAPIF[13:8]) 置位且 CAPFIENn 使能，同样产生一个中断。

图中没有列出的一种情况是：如果在 CAPRIFn (BPWM_CAPIF[5:0]) 刚置位时，上升沿锁存又发生了，过载状态寄存器 CRIFOVn (BPWM_CAPSTS[5:0]) 会由硬件置 1 来表示 CAPRIFn (BPWM_CAPIF[5:0]) 过载了。类似的，如果下降沿锁存再次发生了，同样的情况会发生在过载状态寄存器 CRIFOVn (BPWM_CAPSTS[13:8]) 和 CAPFIFn (BPWM_CAPIF[13:8]) 寄存器。

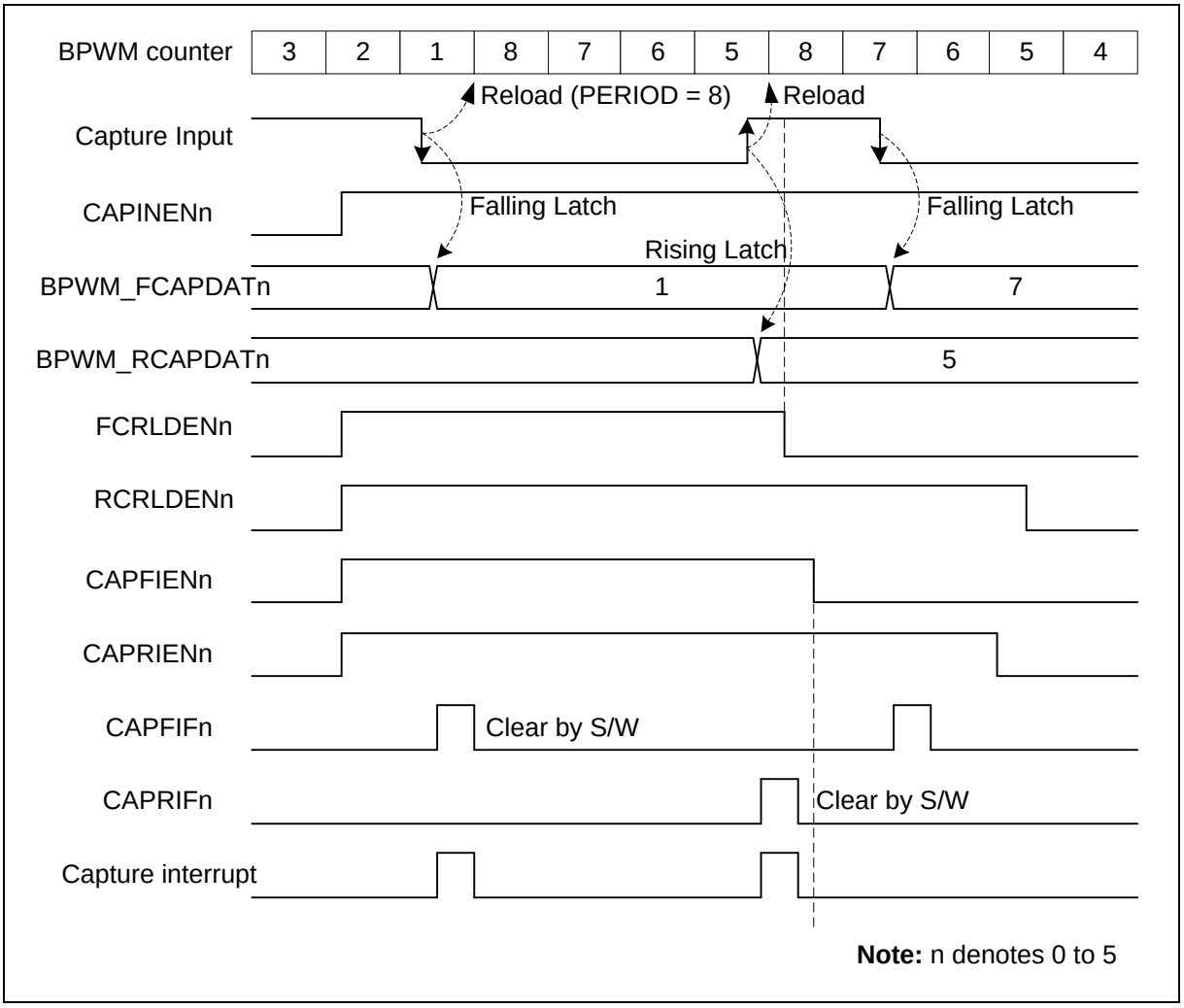


图 6.13-24 捕捉操作波形图

满足以下条件捕获脉冲宽度计算，如下公式：

1. 捕获的正或负脉冲宽度短于计数器周期；
2.计数器以减计数器类型运行；
3. 通过将 PWM_CAPCTL 寄存器的 FCRLDENn 和 RCRLDENn 位设置为 1，可以在下降和上升捕获事件重新加载计数器。

对于负脉冲，通道上的低脉冲宽度为：(BPWM_PERIOD + 1 - BPWM_RCAPDATn) 个 BPWM 计数器时间，其中 BPWM 计数器时间为(CLKPSC+1) * BPWMx_CLK 时钟时间。如图6. 13-24，低脉冲宽度为 8+1-5 = 4 个 BPWM 计数器时间。

对于正脉冲，通道上的高脉冲宽度为：(BPWM_PERIOD + 1 - BPWM_ FCAPDATn) 个 BPWM 计数器时间，其中 BPWM 计数器时间为(CLKPSC+1) * BPWMx_CLK 时钟时间。如图6. 13-24，低脉冲宽度为 8+1-7 = 2 个 BPWM 计数器时间。

6.13.6 寄存器映射

R: 只读, **W**: 只写, **R/W**: 读/写

寄存器	偏移量	R/W	描述	复位值
BPWM基地址: **BPWM0_BA = 0x4005_A000** **BPWM1_BA = 0x4005_B000**				
BPWM_CTL0 **x=0, 1**	BPWMx_BA+0x00	R/W	BPWM 控制寄存器0	0x0000_0000
BPWM_CTL1 **x=0, 1**	BPWMx_BA+0x04	R/W	BPWM 控制寄存器1	0x0000_0000
BPWM_CLKSRC **x=0, 1**	BPWMx_BA+0x10	R/W	BPWM 时钟源寄存器	0x0000_0000
BPWM_CLKPSC **x=0, 1**	BPWMx_BA+0x14	R/W	BPWM 时钟预分频寄存器	0x0000_0000
BPWM_CNTEN **x=0, 1**	BPWMx_BA+0x20	R/W	BPWM 计数器使能寄存器	0x0000_0000
BPWM_CNTCLR **x=0, 1**	BPWMx_BA+0x24	R/W	BPWM 清除计数器寄存器	0x0000_0000
BPWM_PERIOD **x=0, 1**	BPWMx_BA+0x30	R/W	BPWM 周期寄存器	0x0000_0000
BPWM_CMPDAT0 **x=0, 1**	BPWMx_BA+0x50	R/W	BPWM 比较值寄存器0	0x0000_0000
BPWM_CMPDAT1 **x=0, 1**	BPWMx_BA+0x54	R/W	BPWM 比较值寄存器1	0x0000_0000
BPWM_CMPDAT2 **x=0, 1**	BPWMx_BA+0x58	R/W	BPWM 比较值寄存器2	0x0000_0000
BPWM_CMPDAT3 **x=0, 1**	BPWMx_BA+0x5C	R/W	BPWM 比较值寄存器3	0x0000_0000
BPWM_CMPDAT4 **x=0, 1**	BPWMx_BA+0x60	R/W	BPWM 比较值寄存器4	0x0000_0000
BPWM_CMPDAT5 **x=0, 1**	BPWMx_BA+0x64	R/W	BPWM 比较值寄存器5	0x0000_0000
BPWM_CNT **x=0, 1**	BPWMx_BA+0x90	R	BPWM 计数器寄存器	0x0000_0000
BPWM_WGCTL0 **x=0, 1**	BPWMx_BA+0xB0	R/W	BPWM 发生器寄存器0	0x0000_0000
BPWM_WGCTL1 **x=0, 1**	BPWMx_BA+0xB4	R/W	BPWM 发生器寄存器1	0x0000_0000
BPWM_MSKEN	BPWMx_BA+0xB8	R/W	BPWM 屏蔽使能寄存器	0x0000_0000

x=0, 1				
BPWM_MSK **x=0, 1**	BPWMx_BA+0xBC	R/W	BPWM 屏蔽数据寄存器	0x0000_0000
BPWM_POLCTL **x=0, 1**	BPWMx_BA+0xD4	R/W	BPWM 管脚极性反转寄存器	0x0000_0000
BPWM_POEN **x=0, 1**	BPWMx_BA+0xD8	R/W	BPWM 输出使能寄存器	0x0000_0000
BPWM_INTEN **x=0, 1**	BPWMx_BA+0xE0	R/W	BPWM 中断使能寄存器	0x0000_0000
BPWM_INTSTS **x=0, 1**	BPWMx_BA+0xE8	R/W	BPWM 中断标志寄存器	0x0000_0000
BPWM_EADCTS0 **x=0, 1**	BPWMx_BA+0xF8	R/W	BPWM 触发 EADC0/1源选择寄存器0	0x0000_0000
BPWM_EADCTS1 **x=0, 1**	BPWMx_BA+0xFC	R/W	BPWM 触发 EADC0/1源选择寄存器1	0x0000_0000
BPWM_SSCTL **x=0, 1**	BPWMx_BA+0x110	R/W	BPWM 同步开始控制寄存器	0x0000_0000
BPWM_SSTRG **x=0, 1**	BPWMx_BA+0x114	W	BPWM 同步开始触发寄存器	0x0000_0000
BPWM_STATUS **x=0, 1**	BPWMx_BA+0x120	R/W	BPWM 状态寄存器	0x0000_0000
BPWM_CAPINEN **x=0, 1**	BPWMx_BA+0x200	R/W	BPWM 输入捕捉使能寄存器	0x0000_0000
BPWM_CAPCTL **x=0, 1**	BPWMx_BA+0x204	R/W	BPWM 捕捉控制寄存器	0x0000_0000
BPWM_CAPSTS **x=0, 1**	BPWMx_BA+0x208	R	BPWM 捕捉状态寄存器	0x0000_0000
BPWM_RCAPDAT0 **x=0, 1**	BPWMx_BA+0x20C	R	BPWM 上升沿捕捉数据寄存器0	0x0000_0000
BPWM_FCAPDAT0 **x=0, 1**	BPWMx_BA+0x210	R	BPWM 下降沿捕捉数据寄存器0	0x0000_0000
BPWM_RCAPDAT1 **x=0, 1**	BPWMx_BA+0x214	R	BPWM 上升沿捕捉数据寄存器1	0x0000_0000
BPWM_FCAPDAT1 **x=0, 1**	BPWMx_BA+0x218	R	BPWM 下降沿捕捉数据寄存器1	0x0000_0000
BPWM_RCAPDAT2 **x=0, 1**	BPWMx_BA+0x21C	R	BPWM 上升沿捕捉数据寄存器2	0x0000_0000
BPWM_FCAPDAT2 **x=0, 1**	BPWMx_BA+0x220	R	BPWM 下降沿捕捉数据寄存器2	0x0000_0000

BPWM_RCAPDAT3 x=0, 1	BPWMx_BA+0x224	R	BPWM 上升沿捕捉数据寄存器3	0x0000_0000
BPWM_FCAPDAT3 x=0, 1	BPWMx_BA+0x228	R	BPWM 下降沿捕捉数据寄存器3	0x0000_0000
BPWM_RCAPDAT4 x=0, 1	BPWMx_BA+0x22C	R	BPWM 上升沿捕捉数据寄存器4	0x0000_0000
BPWM_FCAPDAT4 x=0, 1	BPWMx_BA+0x230	R	BPWM 下降沿捕捉数据寄存器4	0x0000_0000
BPWM_RCAPDAT5 x=0, 1	BPWMx_BA+0x234	R	BPWM 上升沿捕捉数据寄存器5	0x0000_0000
BPWM_FCAPDAT5 x=0, 1	BPWMx_BA+0x238	R	BPWM 下降沿捕捉数据寄存器5	0x0000_0000
BPWM_CAPIEN x=0, 1	BPWMx_BA+0x250	R/W	BPWM 捕捉中断使能寄存器	0x0000_0000
BPWM_CAPIF x=0, 1	BPWMx_BA+0x254	R/W	BPWM 捕捉中断标志寄存器	0x0000_0000
BPWM_PBUF x=0, 1	BPWMx_BA+0x304	R	BPWM 周期缓存	0x0000_0000
BPWM_CMPBUF0 x=0, 1	BPWMx_BA+0x31C	R	BPWM CMPDAT 0 缓存	0x0000_0000
BPWM_CMPBUF1 x=0, 1	BPWMx_BA+0x320	R	BPWM CMPDAT 1 缓存	0x0000_0000
BPWM_CMPBUF2 x=0, 1	BPWMx_BA+0x324	R	BPWM CMPDAT 2 缓存	0x0000_0000
BPWM_CMPBUF3 x=0, 1	BPWMx_BA+0x328	R	BPWM CMPDAT 3 缓存	0x0000_0000
BPWM_CMPBUF4 x=0, 1	BPWMx_BA+0x32C	R	BPWM CMPDAT 4 缓存	0x0000_0000
BPWM_CMPBUF5 x=0, 1	BPWMx_BA+0x330	R	BPWM CMPDAT 5 缓存	0x0000_0000

6.13.7 寄存器描述

BPWM控制寄存器 0 (BPWM_CTL0)

寄存器	偏移	R/W	描述	复位值
BPWM_CTL0	BPWMx_BA+0x00	R/W	BPWM 控制寄存器0	0x0000_0000

31	30	29	28	27	26	25	24
DBGTRIOFF	DBGHALT	Reserved					
23	22	21	20	19	18	17	16
Reserved		IMMLDEN5	IMMLDEN4	IMMLDEN3	IMMLDEN2	IMMLDEN1	IMMLDEN0
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved		CTRLD5	CTRLD4	CTRLD3	CTRLD2	CTRLD1	CTRLD0

位	描述	
[31]	DBGTRIOFF	**ICE 调试模式确认禁止 (写保护)** 0 = ICE 调试模式确认会影响 BPWM 输出。此时，BPWM 管脚会强制设为三态模式 1 = ICE 调试模式确认禁止。无论 ICE 调试模式是否确认，BPWM 管脚都会保持输出 **注：** 该位写保护，具体请参考 SYS_REGLCTL 寄存器。
[30]	DBGHALT	**ICE 调试模式计数器停止(写保护)** 如果计数器停止使能，BPWM 所有计数器都会保持当前值直到退出 ICE 调试模式。 0 =禁止 ICE 调试模式计数器关闭 1 =使能 ICE 调试模式计数器关闭 **注：** 该位写保护，具体请参考 SYS_REGLCTL 寄存器。
[29:22]	Reserved	保留
[16+n] n=0,1..5	IMMLDENn	**立即重载使能位** 每一位 n 控制着对应的 BPWM 通道n 0 = 每个周期结束位置 PERIOD 重载到 PBUF。设定 CTRLD 位，CMPDAT 会在中间位置或者结束位置重载到 CMPBUF。 1 = 软件更新 PERIOD/CMPDAT 时会立即重载到 PBUF 和 CMPBUF。 **注：** 如果 IMMLDENn 使能， CTRLDn 无效。
[15:6]	Reserved	保留
[n] n=0,1..5	CTRLDn	**中心重载** 每一位 n 控制着对应的 BPWM 通道n 上下计数方式时， PERIOD 会在每个周期结束位置重载到 PBUF。CMPDAT 则会在周期中间位置重载到 CMPBUF。

BPWM控制寄存器 1 (BPWM_CTL1)

寄存器	偏移	R/W	描述	复位值
BPWM_CTL1	BPWMx_BA+0x04	R/W	BPWM控制寄存器1	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved						**CNTTYPE0**	

位	描述	
[31:2]	**Reserved**	保留
[1:0]	**CNTTYPE0**	**BPWM 计数器规则类型 0** 每一位 n 控制着对应的 BPWM 通道n 00 = 向上计数方式 (支持捕捉模式) 01 = 向下计数方式 (支持捕捉模式) 10 = 上下计数方式 11 = 保留

BPWM时钟源寄存器(BPWM_CLKSRC)

寄存器	偏移	R/W	描述	复位值
BPWM_CLKSRC	BPWMx_BA+0x10	R/W	BPWM时钟源寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved					ECLKSRC0		

位	描述	
[31:3]	Reserved	保留
[2:0]	ECLKSRC0	BPWM_CH01 外部时钟源选择 000 = BPWMx_CLK， x 指 0 或 1 001 = TIMER0 溢出 010 = TIMER1 溢出 011 = TIMER2 溢出 100 = TIMER3 溢出 其他 = 保留

BPWM时钟分频寄存器(BPWM_CLKPSC)

寄存器	偏移	R/W	描述	复位值
BPWM_CLKPSC	BPWMx_BA+0x14	R/W	BPWM时钟分频寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved				CLKPSC			
7	6	5	4	3	2	1	0
CLKPSC							

位	描述	
[31:12]	Reserved	保留
[11:0]	CLKPSC	**BPWM 计数器时钟分频** BPWM计数器的时钟会被分频器除频，每个 BPWM 对共享一个 BPWM 计数器时钟分频器，BPWM 计数器会被除 (CLKPSC+ 1)。

BPWM计数器使能寄存器(BPWM_CNTEN)

寄存器	偏移	R/W	描述	复位值
BPWM_CNTEN	BPWMx_BA+0x20	R/W	BPWM 计数器使能寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved							**CNTEN0**

位	描述	
[31:1]	**Reserved**	保留
[0]	**CNTEN0**	**BPWM 计数器 0 使能位** 0 = BPWM 计数器和时钟分频器停止运行 1 = BPWM 计数器和时钟分频器开始运行

BPWM清除计数器寄存器(BPWM_CNTCLR)

寄存器	偏移	R/W	描述	复位值
BPWM_CNTCLR	BPWMx_BA+0x24	R/W	BPWM清除计数器寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved							CNTCLR0

位	描述	
[31:1]	Reserved	保留
[0]	CNTCLR0	**清除 BPWM 计数器控制位 0** 硬件自动清除 0 = 无效 1 = 清除 16 位的 BPWM 计数器到 0000H

BPWM周期寄存器(BPWM_PERIOD)

寄存器	偏移	R/W	描述	复位值
BPWM_PERIOD	BPWMx_BA+0x30	R/W	BPWM周期寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
PERIOD							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
PERIOD							

位	描述	
[31:16]	**Reserved**	保留
[15:0]	**PERIOD**	**BPWM 周期寄存器** 向上计数模式：该模式下，BPWM 计数器从 0 计数到 PERIOD，再重新从 0 计数 向下计数模式：该模式下，BPWM 计数器从 PERIOD 计数到 0，再重新从 PERIOND 计数 以上，BPWM 周期时间 = (PERIOD+1) * BPWM_CLK 周期 上下计数模式：该模式下，BPWM 计数器从 0 计数到 PERIOD，再向下递减到 0 ，依次循环 BPWM 周期时间= 2 * PERIOD * BPWM_CLK周期

BPWM比较器寄存器0~5 (BPWM_CMPDAT0~5)

寄存器	偏移	R/W	描述	复位值
BPWM_CMPDAT0	BPWMx_BA+0x50	R/W	BPWM 比较值寄存器0	0x0000_0000
BPWM_CMPDAT1	BPWMx_BA+0x54	R/W	BPWM 比较值寄存器1	0x0000_0000
BPWM_CMPDAT2	BPWMx_BA+0x58	R/W	BPWM 比较值寄存器2	0x0000_0000
BPWM_CMPDAT3	BPWMx_BA+0x5C	R/W	BPWM 比较值寄存器3	0x0000_0000
BPWM_CMPDAT4	BPWMx_BA+0x60	R/W	BPWM 比较值寄存器4	0x0000_0000
BPWM_CMPDAT5	BPWMx_BA+0x64	R/W	BPWM 比较值寄存器5	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
CMPDAT							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
CMPDAT							

位	描述	
[31:16]	**Reserved**	保留
[15:0]	**CMPDAT**	**BPWM 比较值寄存器** CMPDAT 用来和 CNTR 比较从而产生 BPWM 波形、中断和触发 EADC 信号 在独立模式下，CMPDAT0~5 代表 6 路 BPWM_CH0~5 的比较点

BPWM计数器寄存器(BPWM_CNT)

寄存器	偏移	R/W	描述	复位值
BPWM_CNT	BPWMx_BA+0x90	R	BPWM计数器寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							DIRF
15	14	13	12	11	10	9	8
CNT							
7	6	5	4	3	2	1	0
CNT							

位	描述	
[31:17]	Reserved	保留
[16]	DIRF	BPWM 方向指示标志 (只读) 0 = 计数器向下计数 1 = 计数器向上计数
[15:0]	CNT	BPWM 数据寄存器 (只读) 用户可以从 CNT寄存器读到当前16位周期计数器的值

BPWM发生器寄存器 0 (BPWM_WGCTL0)

寄存器	偏移	R/W	描述	复位值
BPWM_WGCTL0	BPWMx_BA+0xB0	R/W	BPWM发生器寄存器0	0x0000_0000

31	30	29	28	27	26	25	24
Reserved				PRDPCTL5		PRDPCTL4	
23	22	21	20	19	18	17	16
PRDPCTL3		PRDPCTL2		PRDPCTL1		PRDPCTL0	
15	14	13	12	11	10	9	8
Reserved				ZPCTL5		ZPCTL4	
7	6	5	4	3	2	1	0
ZPCTL3		ZPCTL2		ZPCTL1		ZPCTL0	

位	描述	
[31:28]	**Reserved**	保留
[16+2n+1:16+2n] n=0,1..5	**PRDPCTLn**	**BPWM 周期 (中间) 位置控制** 每一位 n 控制着对应的 BPWM 通道n 00 = 无动作 01 = BPWM 周期 (中间) 位置输出低 10 = BPWM 周期 (中间) 位置输出高 11 = BPWM 周期 (中间) 位置输出触发 BPWM 计数器计数到 (PERIOD+1) 时，BPWM 可以控制输出电平 **注：** 该位是 BPWM 计数器上下计数方式时的中间位置控制
[15:12]	**Reserved**	保留
[2n+1:2n] n=0,1..5	**ZPCTLn**	**BPWM 零位控制** 每一位 n 控制着对应的 BPWM 通道 n 00 = 无动作 01 = BPWM 零位输出低 10 = BPWM 零位输出高 11 = BPWM 零位输出触发 BPWM 可以在 BPWM 计数器计数到 0 时控制输出电平

BPWM发生器寄存器 1(BPWM_WGCTL1)

寄存器	偏移	R/W	描述	复位值
BPWM_WGCTL1	BPWMx_BA+0xB4	R/W	BPWM发生器寄存器1	0x0000_0000

31	30	29	28	27	26	25	24
Reserved				CMPDCTL5		CMPDCTL4	
23	22	21	20	19	18	17	16
CMPDCTL3		CMPDCTL2		CMPDCTL1		CMPDCTL0	
15	14	13	12	11	10	9	8
Reserved				CMPUCTL5		CMPUCTL4	
7	6	5	4	3	2	1	0
CMPUCTL3		CMPUCTL2		CMPUCTL1		CMPUCTL0	

位	描述	
[31:28]	**Reserved**	保留
[16+2n+1:16+2n] n=0,1..5	**CMPDCTLn**	**BPWM 向下比较位置控制** 每一位 n 控制着对应的 BPWM 通道n 00 = 无动作 01 = BPWM 向下比较位置输出低 10 = BPWM 向下比较位置输出高 11 = BPWM 向下比较位置输出触发 BPWM 在向下计数到 CMPDAT 时控制输出电平。
[15:12]	**Reserved**	保留
[2n+1:2n] n=0,1..5	**CMPUCTLn**	**BPWM 向上比较位置控制** 每一位 n 控制着对应的 BPWM 通道n 00 = 无动作 01 = BPWM 向上比较位置输出低 10 = BPWM 向上比较位置输出高 11 = BPWM 向上比较位置输出触发 BPWM 在向上计数到 CMPDAT 时控制输出电平。

BPWM屏蔽使能寄存器(BPWM_MSKEN)

寄存器	偏移	R/W	描述	复位值
BPWM_MSKEN	BPWMx_BA+0xB8	R/W	BPWM屏蔽使能寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved		MSKEN5	MSKEN4	MSKEN3	MSKEN2	MSKEN1	MSKEN0

位	描述	
[31:6]	**Reserved**	保留
[n] n=0,1..5	**MSKENn**	**BPWM 屏蔽使能位** 每一位 n 控制着对应的 BPWM 通道 n 该位使能时 BPWM 输出信号被屏蔽。对应的 BPWM 通道 n 会输出 MSKDATn (BPWM_MSK[5:0]) 的值 0 = BPWM 输出信号未被屏蔽 1 = BPWM 输出信号被屏蔽，输出 MSKDATn 的值

BPWM屏蔽数据寄存器(BPWM_MSK)

寄存器	偏移	R/W	描述	复位值
BPWM_MSK	BPWMx_BA+0xBC	R/W	BPWM屏蔽数据寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved		**MSKDAT5**	**MSKDAT4**	**MSKDAT3**	**MSKDAT2**	**MSKDAT1**	**MSKDAT0**

位	描述	
[31:6]	**Reserved**	保留
[n] n=0,1..5	**MSKDATn**	**BPWM 屏蔽数据位** 如果对应的管脚屏蔽功能使能，该数据位控制 BPWM 管脚输出状态 每一位 n 控制着对应的 BPWM 通道 n 0 = BPWMn 输出逻辑电平低 1 = BPWMn 输出逻辑电平高

BPWM管脚极性反转寄存器(BPWM_POLCTL)

寄存器	偏移	R/W	描述	复位值
BPWM_POLCTL	BPWMx_BA+0xD4	R/W	BPWM管脚极性反转寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved		PINV5	PINV4	PINV3	PINV2	PINV1	PINV0

位	描述	
[31:6]	Reserved	保留
[n] n=0,1..5	PINVn	**BPWM 管脚极性反转控制** 该寄存器控制 BPWM 输出极性的状态 每一位 n 控制着对应的 BPWM 通道 n 0 = 禁止BPWM 输出极性反转 1 = 使能BPWM 输出极性反转

BPWM输出使能寄存器(BPWM_POEN)

寄存器	偏移	R/W	描述	复位值
BPWM_POEN	BPWMx_BA+0xD8	R/W	BPWM输出使能寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved		**POEN5**	**POEN4**	**POEN3**	**POEN2**	**POEN1**	**POEN0**

位	描述	
[31:6]	**Reserved**	保留
[n] n=0,1..5	**POENn**	**BPWM 管脚输出使能位** 每一位 n 控制着对应的 BPWM 通道 n 0 = BPWM 管脚处于三态模式 1 = BPWM 管脚处于输出模式

BPWM中断使能寄存器(BPWM_INTEN)

寄存器	偏移	R/W	描述	复位值
BPWM_INTEN	BPWMx_BA+0xE0	R/W	BPWM中断使能寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved		**CMPDIEN5**	**CMPDIEN4**	**CMPDIEN3**	**CMPDIEN2**	**CMPDIEN1**	**CMPDIEN0**
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved		**CMPUIEN5**	**CMPUIEN4**	**CMPUIEN3**	**CMPUIEN2**	**CMPUIEN1**	**CMPUIEN0**
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							**PIEN0**
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved							**ZIEN0**

位	描述	
[31:30]	**Reserved**	保留
[24+n] n=0,1..5	**CMPDIENn**	**BPWM 向下计数中断使能位** 每一位 n 控制着对应的 BPWM 通道 n 0 = 禁止向下计数中断 1 =使能向下计数中断
[23:22]	**Reserved**	保留
[16+n] n=0,1..5	**CMPUIENn**	**BPWM 向上计数中断使能位** 每一位 n 控制着对应的 BPWM 通道 n 0 =禁止向上计数中断 1 =使能向上计数中断
[15:9]	**Reserved**	保留
[8]	**PIEN0**	**BPWM 周期位置中断 0 使能位** 0 = 禁止周期位置中断 1 =使能周期位置中断 **注:**上下计数方式里周期位置指中间位置
[7:1]	**Reserved**	保留
[0]	**ZIEN0**	**BPWM 零位中断 0 使能位** 0 = 禁止零位中断 1 = 使能零位中断

BPWM中断标志寄存器(BPWM_INTSTS)

寄存器	偏移	R/W	描述	复位值
BPWM_INTSTS	BPWMx_BA+0xE8	R/W	BPWM中断标志寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved		CMPDIF5	CMPDIF4	CMPDIF3	CMPDIF2	CMPDIF1	CMPDIF0
23	22	21	20	19	18	17	16
Reserved		CMPUIF5	CMPUIF4	CMPUIF3	CMPUIF2	CMPUIF1	CMPUIF0
15	14	13	12	11	10	9	8
Reserved							PIF0
7	6	5	4	3	2	1	0
Reserved							ZIF0

位	描述	
[31:30]	**Reserved**	保留
[24+n] n=0,1..5	**CMPDIFn**	**BPWM 向下比较计数中断标志位** 每一位 n 控制着对应的 BPWM 通道 n BPWM 向下计数到 BPWM_CMPDATn 时，硬件置位该标志。软件可写 1 清零该位。 **注：**如果 CMPDAT 等于 PERIOD，该标志在向下计数方式时无效
[23:22]	**Reserved**	保留
[16+n] n=0,1..5	**CMPUIFn**	**BPWM 向上比较计数中断标志** 每一位 n 控制着对应的 BPWM 通道 n BPWM 向上计数到 BPWM_CMPDATn 时，硬件置位该标志。软件可写 1 清 0 该位。 **注：**如果 CMPDAT 等于 PERIOD，该标志在向上计数方式时无效.
[15:9]	**Reserved**	保留
[8]	**PIF0**	**BPWM 周期位置中断标志 0** BPWM_CH0 向上计数到 BPWM_PERIOD0 时，硬件置位该位。软件可写 1 清 0 该位。
[7:1]	**Reserved**	保留
[0]	**ZIF0**	**BPWM 零位中断标志 0** BPWM_CH0 向上计数到 0 时，硬件置位该位。软件可写 1 清 0 该位。

BPWM触发 EADC0/1源选择寄存器 0 (BPWM_EADCTS0)

寄存器	偏移	R/W	描述	复位值
BPWM_EADCTS0	BPWMx_BA+0xF8	R/W	BPWM触发 EADC0/1源选择寄存器 0	0x0000_0000

31	30	29	28	27	26	25	24
TRGEN3	Reserved			TRGSEL3			
23	22	21	20	19	18	17	16
TRGEN2	Reserved			TRGSEL2			
15	14	13	12	11	10	9	8
TRGEN1	Reserved			TRGSEL1			
7	6	5	4	3	2	1	0
TRGEN0	Reserved			TRGSEL0			

位	描述	
[31]	**TRGEN3**	BPWM_CH3 触发 EADC0/1使能位
[30:28]	**Reserved**	保留
[27:24]	**TRGSEL3**	**BPWM_CH3 触发 EADC0/1源选择** 0000 = BPWM_CH2 零位 0001 = BPWM_CH2 周期位置 0010 = BPWM_CH2 零或周期位置 0011 = BPWM_CH2 向上计数的 CMPDAT 位置 0100 = BPWM_CH2 向下计数的 CMPDAT 位置 0101 = 保留 0110 = 保留 0111 = 保留 1000 = BPWM_CH3 向上计数的 CMPDAT 位置 1001 = BPWM_CH3 向下计数的 CMPDAT 位置 其他保留
[23]	**TRGEN2**	BPWM_CH2 触发 EADC0/1使能位
[22:20]	**Reserved**	保留
[19:16]	**TRGSEL2**	**BPWM_CH2 触发 EAD0/1C 源选择** 0000 = BPWM_CH2零位 0001 = BPWM_CH2周期位置 0010 = BPWM_CH2 零位或周期位置 0011 = BPWM_CH2 向上计数的 CMPDAT 位置 0100 = BPWM_CH2 向下计数的 CMPDAT 位置

		0101 = 保留 0110 = 保留 0111 = 保留 1000 = BPWM_CH3 向上计数的 CMPDAT 位置. 1001 = BPWM_CH3 向下计数的 CMPDAT 位置. 其他保留
[15]	**TRGEN1**	BPWM_CH1 触发 EADC0/1使能位
[14:12]	**Reserved**	保留
[11:8]	**TRGSEL1**	**BPWM_CH1 触发 EADC0/1源选择** 0000 = BPWM_CH0 零位 0001 = BPWM_CH0 周期位置 0010 = BPWM_CH0零位或周期位置 0011 = BPWM_CH0 向上计数的 CMPDAT 位置 0100 = BPWM_CH0 向下计数的 CMPDAT 位置 0101 = 保留 0110 = 保留 0111 = 保留 1000 = BPWM_CH1 向上计数的 CMPDAT 位置 1001 = BPWM_CH1 向下计数的 CMPDAT 位置 其他保留
[7]	**TRGEN0**	BPWM_CH0 触发 EADC0/1使能位
[6:4]	**Reserved**	保留
[3:0]	**TRGSEL0**	**BPWM_CH0 触发 EADC0/1源选择** 0011 = BPWM_CH0 向上计数的 CMPDAT 位置 0100 = BPWM_CH0 向下计数的 CMPDAT 位置 0101 = 保留 0110 = 保留 0111 = 保留 1000 = BPWM_CH1 向上计数的 CMPDAT 位置 1001 = BPWM_CH1 向下计数的 CMPDAT 位置 其他保留

BPWM触发 EADC0/1源选择寄存器 1(BPWM_EADCTS1)

寄存器	偏移	R/W	描述	复位值
BPWM_EADCTS1	BPWMx_BA+0xFC	R/W	BPWM触发 EADC0/1源选择寄存器 1	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
TRGEN5	Reserved			TRGSEL5			
7	6	5	4	3	2	1	0
TRGEN4	Reserved			TRGSEL4			

位	描述	
[31:16]	**Reserved**	保留
[15]	**TRGEN5**	BPWM_CH5 触发 EADC0/1使能位
[14:12]	**Reserved**	保留
[11:8]	**TRGSEL5**	**BPWM_CH5 触发 EADC0/1源选择** 0000 = BPWM_CH4 零位 0001 = BPWM_CH4 周期位置 0010 = BPWM_CH4 零位或周期位置 0011 = BPWM_CH4 向上计数的 CMPDAT 位置 0100 = BPWM_CH4 向下计数的 CMPDAT 位置 0101 = 保留 0110 = 保留 0111 = 保留 1000 = BPWM_CH5 向上计数的 CMPDAT 位置 1001 = BPWM_CH5 向下计数的 CMPDAT 位置 其他保留
[7]	**TRGEN4**	BPWM_CH4 触发 EADC0/1使能位
[6:4]	**Reserved**	保留
[3:0]	**TRGSEL4**	**BPWM_CH4 触发 EADC0/1源选择** 0000 = BPWM_CH4 零位 0001 = BPWM_CH4 周期位置 0010 = BPWM_CH4零位或周期位置 0011 = BPWM_CH4 向上计数的 CMPDAT 位置

		0100 = BPWM_CH4 向下计数的 CMPDAT 位置 0101 = 保留 0110 = 保留 0111 = 保留 1000 = BPWM_CH5 向上计数的 CMPDAT 位置 1001 = BPWM_CH5 向下计数的 CMPDAT 位置 其他保留

BPWM同步开始寄存器(BPWM_SSCTL)

寄存器	偏移	R/W	描述	复位值
BPWM_SSCTL	BPWMx_BA+0x110	R/W	BPWM同步开始寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved						SSRC	
7	6	5	4	3	2	1	0
Reserved							SSEN0

位	描述	
[31:10]	Reserved	保留
[9:8]	SSRC	**BPWM 同步开始源选择** 00 = 同步开始源来自 PWM0 01 = 同步开始源来自 PWM1 10 = 同步开始源来自 BPWM0 11 = 同步开始源来自 BPWM1
[7:1]	Reserved	保留
[0]	SSEN0	**BPWM 同步开始功能 0 使能位** 同步开始功能使能后，写 BPWM 同步开始使能位 (CNTSEN) 可以使能 BPWM_CH0 计数器使能位 (CNTEN0) 0 = 禁止BPWM 同步开始功能 1 = 使能BPWM 同步开始功能

BPWM同步开始触发寄存器(BPWM_SSTRG)

寄存器	偏移	R/W	描述	复位值
BPWM_SSTRG	BPWMx_BA+0x114	W	BPWM同步开始触发寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved							CNTSEN

位	描述	
[31:1]	Reserved	保留
[0]	CNTSEN	BPWM 计数器同步开始使能位 (只写) BPWM 计数器同步使能功能可以使 PWM 和 BPWM 通道同时开始计数 如果相关的 BPWM 通道计数器同步开始功能也使能了，向该位写 1 还会使能计数器使能位

BPWM状态寄存器(BPWM_STATUS)

寄存器	偏移	R/W	描述	复位值
BPWM_STATUS	BPWMx_BA+0x120	R/W	BPWM状态寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved		EADCTRG5	EADCTRG4	EADCTRG3	EADCTRG2	EADCTRG1	EADCTRG0
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved							CNTMAXF0

位	描述	
[31:22]	Reserved	保留
[16+n] n=0,1..5	EADCTRGn	**EADC0&EADC1开始转换状态** 每一位 n 控制着对应的 BPWM 通道 n 0 = 没有 EADC0/1开始转换触发事件发生 1 = 有 EADC0/1开始转换触发事件发生，软件写 1 清 0 该位
[15:1]	Reserved	保留
[0]	CNTMAXF0	**时间基准计数器 0 为 0xFFFF 锁存标志** 0 = 时间基准计数器从未达到最大值 0xFFFF 1 = 时间基准计数器达到最大值，软件写 1 清 0 该位

BPWM输入捕捉使能寄存器(BPWM_CAPINEN)

寄存器	偏移	R/W	描述	复位值
BPWM_CAPINEN	BPWMx_BA+0x200	R/W	BPWM 输入捕捉使能寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved	Reserved	CAPINEN5	CAPINEN4	CAPINEN3	CAPINEN2	CAPINEN1	CAPINEN0

位	描述	
[31:6]	**Reserved**	保留
[n] n=0,1..5	**CAPINENn**	**输入捕捉使能位** 每一位 n 控制着对应的 BPWM 通道 n 0 = 禁止BPWM 通道输入捕捉通路，BPWM 通道捕捉功能的输入将被视作0 1 = 使能BPWM 通道输入捕捉通路，BPWM 通道捕捉的输入由多功能引脚的状态决定

BPWM捕捉控制寄存器(BPWM_CAPCTL)

寄存器	偏移	R/W	描述	复位值
BPWM_CAPCTL	BPWMx_BA+0x204	R/W	BPWM捕捉控制寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved		FCRLDEN5	FCRLDEN4	FCRLDEN3	FCRLDEN2	FCRLDEN1	FCRLDEN0
23	22	21	20	19	18	17	16
Reserved		RCRLDEN5	RCRLDEN4	RCRLDEN3	RCRLDEN2	RCRLDEN1	RCRLDEN0
15	14	13	12	11	10	9	8
Reserved		CAPINV5	CAPINV4	CAPINV3	CAPINV2	CAPINV1	CAPINV0
7	6	5	4	3	2	1	0
Reserved		CAPEN5	CAPEN4	CAPEN3	CAPEN2	CAPEN1	CAPEN0

位	描述	
[31:30]	**Reserved**	保留
[24+n] n=0,1..5	**FCRLDENn**	**下降沿捕捉重载使能位** 每一位 n 控制着对应的 BPWM 通道 n 0 = 禁止下降沿捕捉重载计数器 1 = 使能下降沿捕捉重载计数器
[23:22]	**Reserved**	保留
[16+n] n=0,1..5	**RCRLDENn**	**上升沿捕捉重载使能位** 每一位 n 控制着对应的 BPWM 通道 n 0 = 禁止上升沿捕捉重载计数器 1 = 使能上升沿捕捉重载计数器
[15:14]	**Reserved**	保留
[8+n] n=0,1..5	**CAPINVn**	**捕捉反相使能位** 每一位 n 控制着对应的 BPWM 通道 n 0 = 禁止捕捉源反相 1 = 使能捕捉源反相。从 GPIO 反转输入信号
[7:6]	**Reserved**	保留
[n] n=0,1..5	**CAPENn**	**捕捉功能使能位** 每一位 n 控制着对应的 BPWM 通道 n 0 = 禁止捕捉功能，RCAPDAT/FCAPDAT不会被更新 1 = 使能捕捉功能。

BPWM捕捉状态寄存器(BPWM_CAPSTS)

寄存器	偏移	R/W	描述	复位值
BPWM_CAPSTS	BPWMx_BA+0x208	R	BPWM捕捉状态寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved		**CFIFOV5**	**CFIFOV4**	**CFIFOV3**	**CFIFOV2**	**CFIFOV1**	**CFIFOV0**
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved		**CRIFOV5**	**CRIFOV4**	**CRIFOV3**	**CRIFOV2**	**CRIFOV1**	**CRIFOV0**

位	描述	
[31:14]	**Reserved**	保留
[8+n] n=0,1..5	**CFIFOVn**	**下降沿捕捉中断标志溢出状态 (只读)** 每一位 n 控制着对应的 BPWM 通道 n 对应的 CAPFIF 为 1 时，如果发生了下降沿锁存，该位会置位。 **注：**用户清除对应的 CAPFIF 时，该位会自动清零。
[7:6]	**Reserved**	保留
[n] n=0,1..5	**CRIFOVn**	**上降沿捕捉中断标志溢出状态 (只读)** 每一位 n 控制着对应的 BPWM 通道 n 对应的 CAPRIF 为 1 时，如果发生了上升沿锁存，该位会置位。 **注：**用户清除对应的 CAPRIF 时，该位会自动清零。

BPWM上升沿捕捉数据寄存器 0~5(BPWM_RCAPDAT 0~5)

寄存器	偏移	R/W	描述	复位值
BPWM_RCAPDAT0	BPWMx_BA+0x20C	R	BPWM 上升沿捕捉数据寄存器0	0x0000_0000
BPWM_RCAPDAT1	BPWMx_BA+0x214	R	BPWM 上升沿捕捉数据寄存器1	0x0000_0000
BPWM_RCAPDAT2	BPWMx_BA+0x21C	R	BPWM 上升沿捕捉数据寄存器2	0x0000_0000
BPWM_RCAPDAT3	BPWMx_BA+0x224	R	BPWM 上升沿捕捉数据寄存器3	0x0000_0000
BPWM_RCAPDAT4	BPWMx_BA+0x22C	R	BPWM 上升沿捕捉数据寄存器4	0x0000_0000
BPWM_RCAPDAT5	BPWMx_BA+0x234	R	BPWM 上升沿捕捉数据寄存器5	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
RCAPDAT							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
RCAPDAT							

位	描述	
[31:16]	**Reserved**	保留
[15:0]	**RCAPDAT**	**BPWM 上升沿捕捉数据 (只读)** 上升沿捕捉发生时，BPWM 计数器值会被保存到该寄存器

BPWM下降沿捕捉数据寄存器 0~5(BPWM_FCAPDAT 0~5)

寄存器	偏移	R/W	描述	复位值
BPWM_FCAPDAT0	BPWMx_BA+0x210	R	BPWM 下降沿捕捉数据寄存器0	0x0000_0000
BPWM_FCAPDAT1	BPWMx_BA+0x218	R	BPWM 下降沿捕捉数据寄存器1	0x0000_0000
BPWM_FCAPDAT2	BPWMx_BA+0x220	R	BPWM 下降沿捕捉数据寄存器2	0x0000_0000
BPWM_FCAPDAT3	BPWMx_BA+0x228	R	BPWM 下降沿捕捉数据寄存器3	0x0000_0000
BPWM_FCAPDAT4	BPWMx_BA+0x230	R	BPWM 下降沿捕捉数据寄存器4	0x0000_0000
BPWM_FCAPDAT5	BPWMx_BA+0x238	R	BPWM 下降沿捕捉数据寄存器5	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
FCAPDAT							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
FCAPDAT							

位	描述	
[31:16]	**Reserved**	保留
[15:0]	**FCAPDAT**	**BPWM 下降沿捕捉数据 (只读)** 下降沿捕捉发生时，BPWM 计数器值会被保存到该寄存器

BPWM捕捉中断使能寄存器(BPWM_CAPIEN)

寄存器	偏移	R/W	描述	复位值
BPWM_CAPIEN	BPWMx_BA+0x250	R/W	BPWM捕捉中断使能寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved		CAPFIEN5	CAPFIEN4	CAPFIEN3	CAPFIEN2	CAPFIEN1	CAPFIEN0
7	6	5	4	3	2	1	0
Reserved		CAPRIEN5	CAPRIEN4	CAPRIEN3	CAPRIEN2	CAPRIEN1	CAPRIEN0

位	描述	
[31:14]	**Reserved**	保留
[13:8]	**CAPFIENn**	**BPWM 捕捉下降沿锁存中断使能位** 每一位 n 控制着对应的 BPWM 通道 n 0 =禁止捕捉下降沿锁存中断 1 = 使能捕捉下降沿锁存中断
[7:6]	**Reserved**	保留
[5:0]	**CAPRIENn**	**BPWM 捕捉上升沿锁存中断使能位** 每一位 n 控制着对应的 BPWM 通道 n 0 =禁止捕捉上升沿锁存中断 1 =使能捕捉上升沿锁存中断

BPWM捕捉中断标志寄存器(BPWM_CAPIF)

寄存器	偏移	R/W	描述	复位值
BPWM_CAPIF	BPWMx_BA+0x254	R/W	BPWM 捕捉中断标志寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved		**CAPFIF5**	**CAPFIF4**	**CAPFIF3**	**CAPFIF2**	**CAPFIF1**	**CAPFIF0**
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved		**CAPRIF5**	**CAPRIF4**	**CAPRIF3**	**CAPRIF2**	**CAPRIF1**	**CAPRIF0**

位	描述	
[31:14]	**Reserved**	保留
[8+n] n=0,1..5	**CAPFIFn**	**BPWM 捕捉下降沿锁存中断标志** 该位写 1 清 0，每一位 n 控制着对应的 BPWM 通道 n 0 = 无捕捉下降沿锁存情况发生 1 = 捕捉下降沿锁存发生，该位置 1
[7:6]	**Reserved**	保留
[n] n=0,1..5	**CAPRIFn**	**BPWM 捕捉上升沿锁存中断标志** 该位写 1 清 0，每一位 n 控制着对应的 BPWM 通道 n 0 = 无捕捉上升沿锁存情况发生 1 = 捕捉上升沿锁存发生，该位置 1

BPWM周期缓存(BPWM_PBUF)

寄存器	偏移	R/W	描述	复位值
BPWM_PBUF	BPWMx_BA+0x304	R	BPWM 周期缓存	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
PBUF							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
PBUF							

位	描述	
[31:16]	**Reserved**	保留
[15:0]	**PBUF**	**BPWM 周期缓存 (只读)** 同 PERIOD 有效寄存器一样使用

BPWM比较值寄存器缓存 0~5(BPWM_CMPBUF0~5)

寄存器	偏移	R/W	描述	复位值
BPWM_CMPBUF0	BPWMx_BA+0x31C	R	BPWM CMPDAT 0缓存	0x0000_0000
BPWM_CMPBUF1	BPWMx_BA+0x320	R	BPWM CMPDAT 1缓存	0x0000_0000
BPWM_CMPBUF2	BPWMx_BA+0x324	R	BPWM CMPDAT 2缓存	0x0000_0000
BPWM_CMPBUF3	BPWMx_BA+0x328	R	BPWM CMPDAT 3缓存	0x0000_0000
BPWM_CMPBUF4	BPWMx_BA+0x32C	R	BPWM CMPDAT 4缓存	0x0000_0000
BPWM_CMPBUF5	BPWMx_BA+0x330	R	BPWM CMPDAT 5缓存	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
CMPBUF							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
CMPBUF							

位	描述	
[31:16]	**Reserved**	保留
[15:0]	**CMPBUF**	**BPWM 比较器缓存 (只读)** 同 CMP 有效寄存器一样使用

6.14 正交编码接口(QEI)

6.14.1 概述

M480 有两组 QEI 控制器。正交编码接口 (QEI) 解读转速和动作传感器的信息。它可以用在任何使用正交编码器作为反馈的应用里。

6.14.2 特性

6.14.2.1 正交编码接口 (QEI) 特性

- 多达 2 个 QEI 控制器，QEI0 及 QEI1
- 2 个 QEI 相位输入，QEA 及 QEB；1 个索引输入
- 1 个 32 位的上/下正交编码脉冲计数器 (QEI_CNT)
- 1 个 32 位的软件锁存正交编码脉冲计数器保存寄存器 (QEI_CNTHOLD)
- 1 个 32 位的正交编码脉冲计数器索引锁存寄存器 (QEI_CNTLATCH)
- 1 个 32 位的正交编码脉冲计数器比较寄存器 (QEI_CNTCMP)，并提供最大计数预设值寄存器 (QEI_CNTMAX)
- 1 个 QEI 控制寄存器 (QEI_CTL) 和 1 个 QEI 状态寄存器 (QEI_STATUS)
- 4 种正交编码脉冲计数器操作模式
  - 支持 x4 自由计数模式
  - 支持 x2 自由计数模式
  - 支持 x4 比较计数模式
  - 支持 x2 比较计数模式
- 编码脉冲宽度测量模式
- 不做消噪时，QEA/QEB/IDX 的输入频率必须低于 PCLK/4
- 有做消噪时，QEA/QEB/IDX 的输入频率必须低于噪声滤波器的 Clk/8

6.14.3 框图

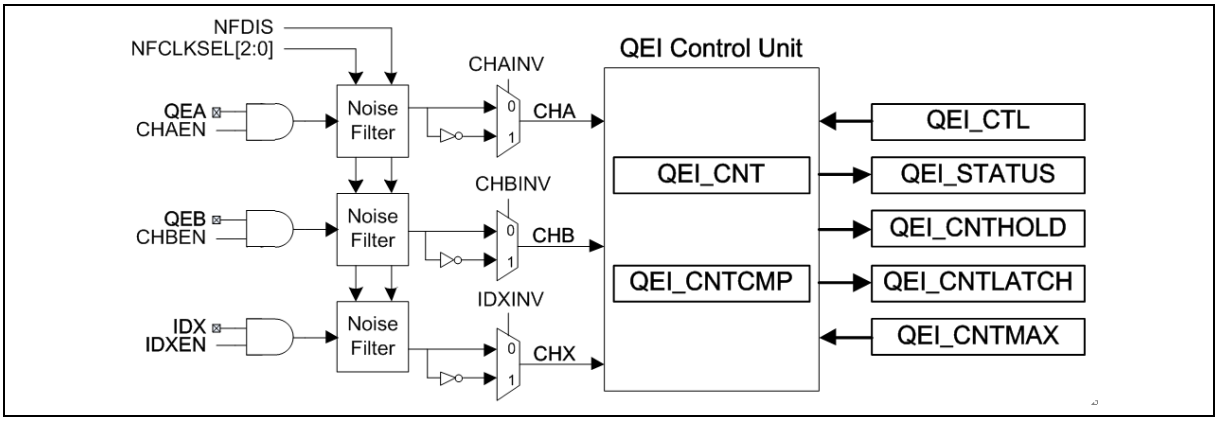


图 6.14-1 QEI 框图

QEI 控制器 QEA 和 QEB 的输入来自于正交编码源的输出，如增量式光电轴角编码器。需要两个相位相差 90 度的通道 A 和 B。正交编码器一般会提供一个索引信号 (管脚 IDX)，可以指示一个绝对位置。每个 QEI 控制单元前都有一个噪声滤波器和极性控制。

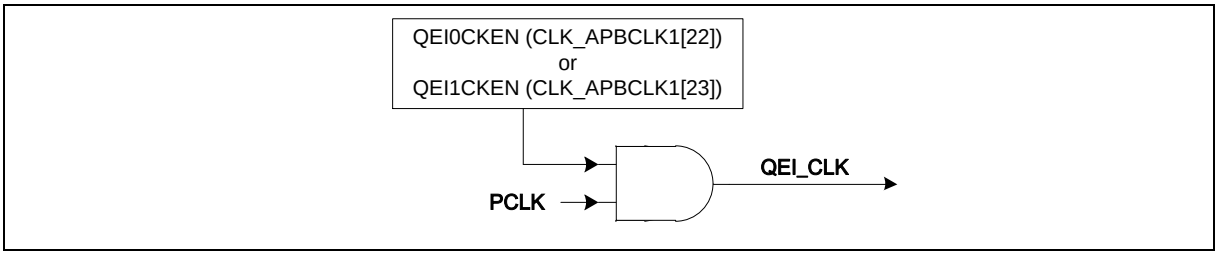


图 6.14-2 QEI 时钟控制

6.14.4 基本配置

6.14.4.1 QEI0 基本配置

- 时钟源配置
  - 在QEI0CKEN (CLK_APBCLK1[22])里使能 QEI0 外设时钟
- 复位配置
  - 在QEI0RST (SYS_IPRST2[22])复位 QEI0 控制器
- 管脚配置

组	管脚名	GPIO	MFP
QEI0	QEI0_A	PD.11	MFP10
		PE.3	MFP11
		PA.4	MFP14
	QEI0_B	PD.10	MFP10
		PE.2	MFP11
		PA.3	MFP14
	QEI0_INDEX	PD.12	MFP10
		PE.4	MFP11
		PA.5	MFP14

6.14.4.2 QEI1基本配置

- 时钟源选择
  - 在QEI1CKEN (CLK_APBCLK1[23])使能 QEI1 外设时钟
- 复位配置
  - 在QEI1RST (SYS_IPRST2[23])复位 QEI1 控制器
- 管脚配置

组	管脚名	GPIO	MFP

QEI1	QEI1_A	PA.9	MFP10
		PE.6	MFP11
		PA.13	MFP12
	QEI1_B	PA.8	MFP10
		PE.5	MFP11
		PA.14	MFP12
	QEI1_INDEX	PA.10	MFP10
		PE.7	MFP11
		PA.12	MFP12

6.14.5 功能描述

QEI 控制器通过检测过滤后的信号 CHA、CHB 以及 CHX 间的相位提前/落后情况来确定方向标志位 DIRF (QEI_STATUS[8]) 的值和脉冲计数器的值。比较计数模式下，通过判断脉冲计数器和最大计数值来判断是否重载脉冲计数器。自由计数模式下，脉冲计数值 CNT (QEI_CNT[31:0]) 会一直计数到 0xFFFF_FFFF。而比较计数模式下，脉冲计数器会计数到 CNTMAX (QEI_CNTMAX[31:0])，接着脉冲计数器会复位成 0，开始下一个计数循环。

6.14.5.1 噪声滤波器

每个 QEI 输入管脚都配备了一个噪声滤波器，可以过滤掉多余的噪声。通过 NFDIS (QEI_CTL[3]) 寄存器位可以关闭 QEA、QEB 和 IDX 的噪声滤波器。如果使能了噪声滤波器，捕捉逻辑会在连续 4 个采样值相同时才传递信号。图6.14-5是一个数字滤波器的实现，输入捕捉要求的脉冲间隔为 4 个 QEI_CLK 。任何小于 4 个 QEI_CLK 的脉冲信号都会被忽略，噪声滤波器所需的时序参见图6.14-5。如图6.14-3，CHA，CHB 和 CHX 分别是 QEA，QEB 和 IDX 经过噪声滤波器和极性控制后的输出。如果关闭了噪声滤波器，输入信号 QEA，QEB 和 IDX 会直接传到内部的 CHA，CHB 和 CHX，没有任何延时。

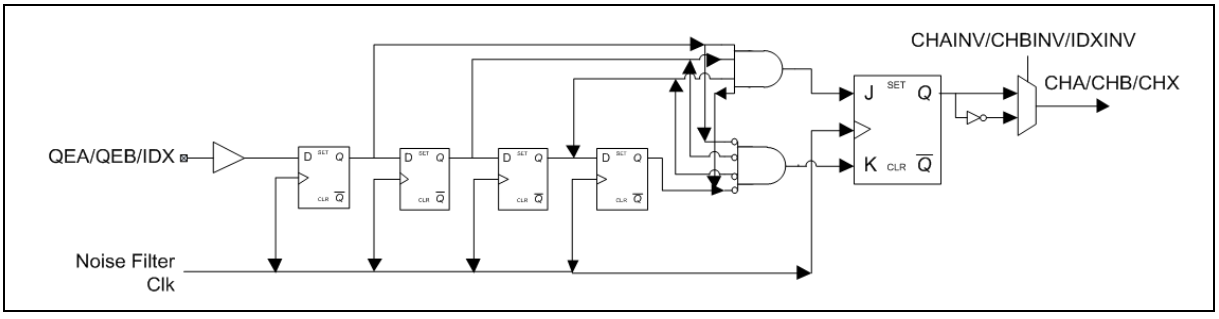


图 6.14-3 噪声滤波器

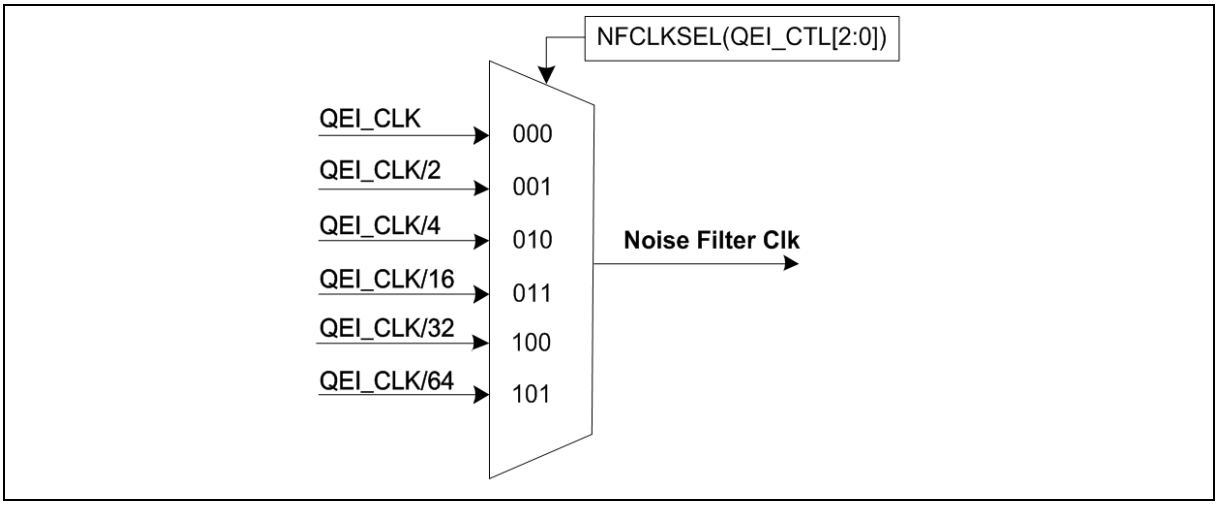


图 6.14-4 噪声滤波器采样时钟源选择

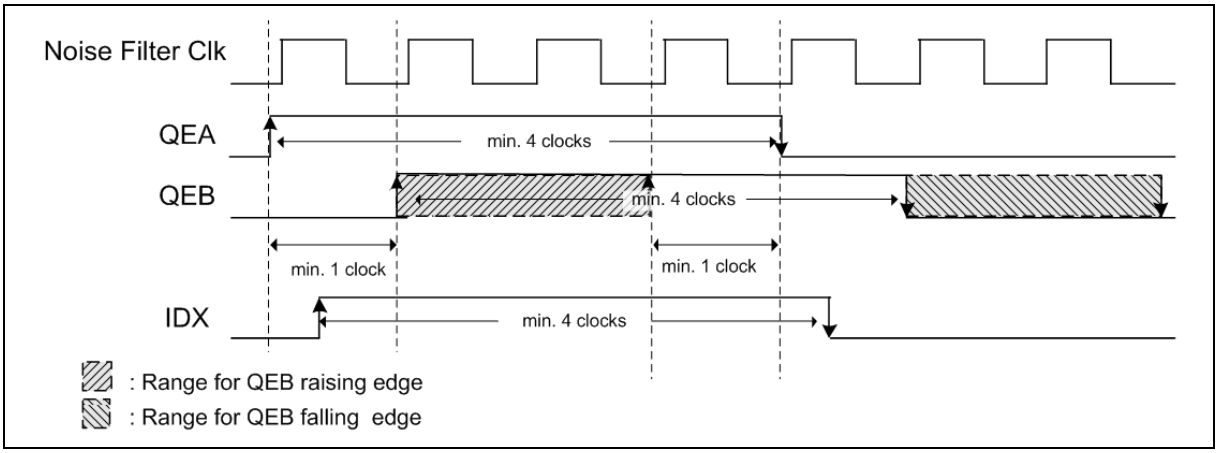


图 6.14-5QEA/QEB/IDX 通过噪声滤波器所需时序

6.14.5.2 正交编码接口操作模式

共有 4 种正交编码脉冲计数操作模式

- 模式 0：x4 自由计数模式
- 模式 1：x2 自由计数模式
- 模式 2：x4 比较计数模式
- 模式 3：x2 比较计数模式

6.14.5.3 自由计数模式

正交编码器脉冲计数器 CNT (QEI_CNT[31:0]) 按方向标志位 DIRF (QEI_STATUS[8]) 向上或向下计数。参考图6. 14-6和图6. 14-7，当计数器溢出或下溢时，会置位 OVUNF (QEI_STATUS[2])。

6.14.5.4 比较计数模式

向上计数还是向下计数取决于方向标志位 DIRF (QEI_STATUS[8])。向上计数时，OVUNF (QEI_STATUS[2]) 会在 CNT (QEI_CNT[31:0]) 溢出从 CNTMAX (QEI_CNTMAX[31:0]) 到 0 时置位，对于 x2 计数模式，这发生在下一个 CHA 边沿，而对于 x4 计数模式，这发生 CHA/CHB 边沿。向下计

数时，OVUNF (QEI_STATUS[2]) 会在 CNT (QEI_CNT[31:0]) 下溢从 0 返回 CNTMAX (QEI_CNTMAX[31:0]) 时置位，对于 x2 计数模式，这发生在下一个 CHA 边沿，而对于 x4 计数模式，这发生 CHA/CHB 边沿。这些模式可以提供给客户转子的位置。如果正交编码器每秒输出 1024 个脉冲到 CHA，用户在 x4 模式可以配置 QEI_MAXCNT 和 CNTCMP (QEI_CNTCMP[31:0]) 为 4095，在 x2 模式则配置为 2047，并在比较计数开始前复位 CNT (QEI_CNT[31:0])。每当 CNT (QEI_CNT[31:0]) 从 CNTCMP (QEI_CNTCMP[31:0]) 溢出，CNTCMP (QEI_CNTCMP[31:0]) 会被设成和 CNTMAX (QEI_CNTMAX[31:0]) 一样的值，也就是说转子会在下一个 CHA/CHB 边沿前转完一圈。具体参考图 6.14-6和图6.14-7。

6.14.5.5 X4/X2 计数模式

在**X4计数模式**, 依据 CHA 和 CHB 的相位关系在每个 CHA 和 CHB 边沿都计数。

QEI x4 计数模式相对于 QEI x2 模式在每个 QEA/QEB 的边缘都计数，频繁的脉冲计数意味着可以提供更高的转子定位精度。设置 QEI 计数模式选择位 MODE (QEI_CTL[9:8]) 为 00b 或 01b 即为 x4 模式。该模式下，QEI 逻辑检测每个 QEA 和 QEB 的输入边沿。

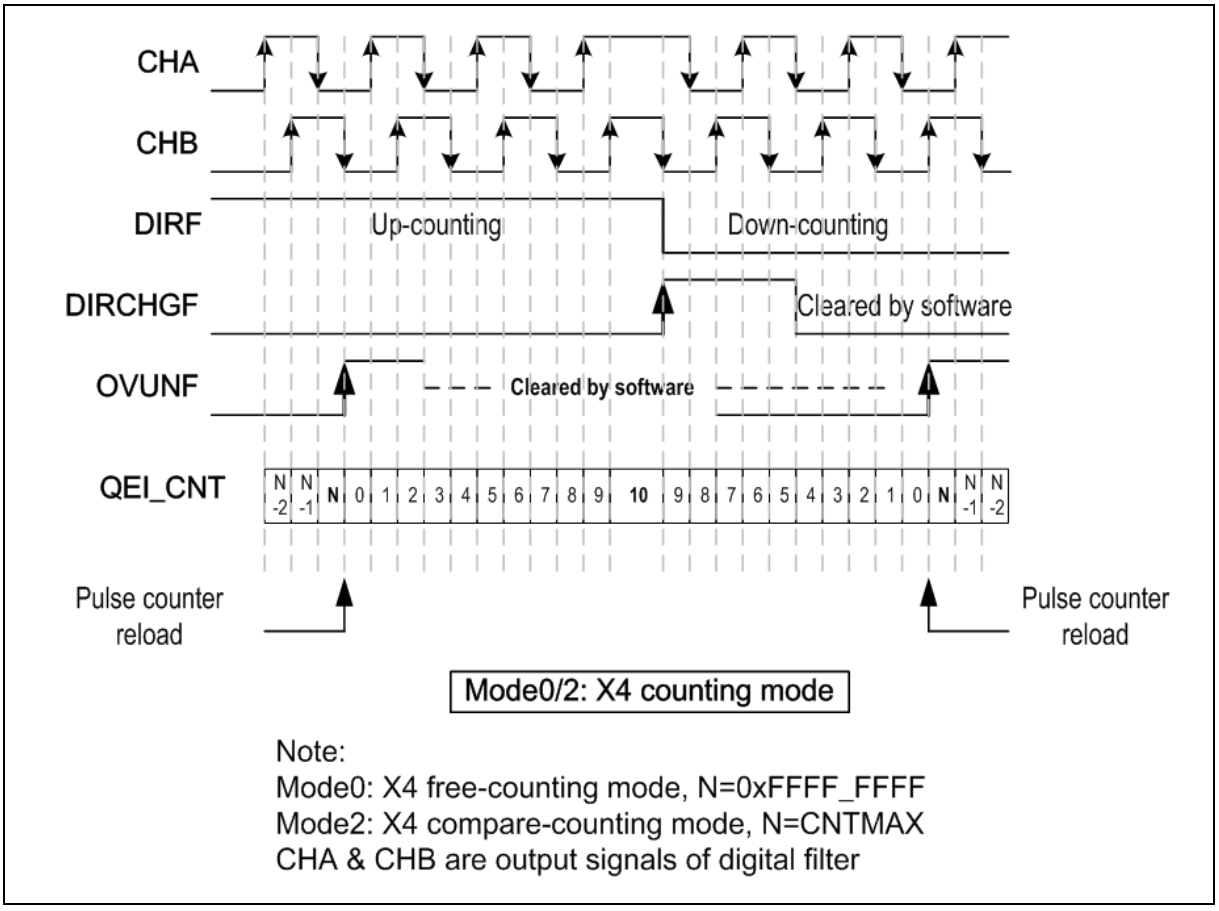


图 6.14-6 X4 计数模式

在**X2计数模式**，根据CHA和CHB的相位，在CHA的边沿增或减计数一次。

设置 QEI 计数模式选择位 MODE (QEI_CTL[9:8]) 为 01b 或 11b 即为 x2 模式。该模式下，QEI 逻辑只检测每个 QEA 的输入边沿。QEA 上的每个上升沿和下降沿信号都会触发脉冲计数。

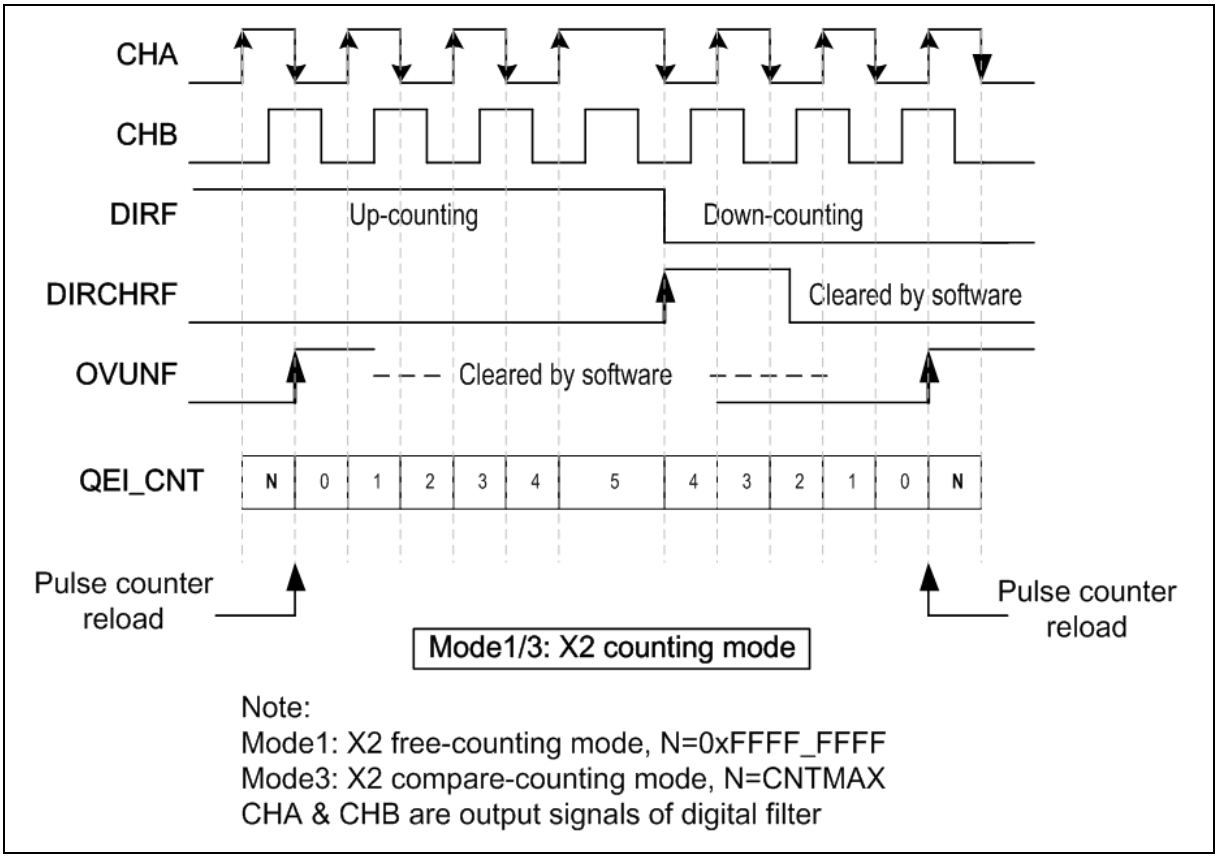


图 6.14-7 X2 计数模式

6.14.5.6 计数方向

CHA 超前 CHB，脉冲信号计数加 1。CHA 落后 CHB，脉冲信号计数减 1。QEI 控制器会产生一个信号来置位 DIRF (QEI_STATUS[8])，并确定当前计数方向。CHA 超前 CHB 时，DIRF (QEI_STATUS[8]) 置 1，位置计数器在每个有效的边沿递增计数。CHA 落后 CHB 时，DIRF (QEI_STATUS[8]) 清 0，位置计数器在每个有效的边沿递减计数。具体参考表6. 14-1。

当前检测到的信号	上一次检测到的信号				DIR (计数方向)
	上升沿		下降沿		
	CHA	CHB	CHA	CHB	
CHA 上升				✓	1 (增加)
		✓			0 (减小)
			✓		翻转 (方向改变)
CHA 下降				✓	0 (减小)
		✓			1 (增加)
	✓				翻转 (方向改变)
CHB 上升	✓				1 (增加)

			✓		0 (减小)
				✓	翻转 (方向改变)
CHB 下降			✓		1 (增加)
	✓				0 (减小)
		✓			翻转 (方向改变)

表 6.14-1 计数方向

6.14.5.7 向上计数

向上计数时DIRF (QEI_STATUS[8])置 1。软件需要先清空OVUNF (QEI_STATUS[2])标志。对于自由计数模式，CNT (QEI_CNT[31:0]) 计数到 0xFFFF_FFFF，在下一个计数方向的边沿置 OVUNF (QEI_STATUS[2]) 高，再复位 CNT (QEI_CNT[31:0]) 到 0。对于比较计数模式，CNT (QEI_CNT[31:0]) 计数到 CNTMAX (QEI_CNTMAX[31:0])，在下一个计数方向的边沿置 OVUNF (QEI_STATUS[2]) 高，再复位 CNT (QEI_CNT[31:0]) 到 0。改变计数方向会触发向下计数，CNT (QEI_CNT[31:0]) 的计数值开始递减。X2 模式下，只在 CHA 边沿置位 OVUNF (QEI_STATUS[2])。而 X4 模式下，CHA 和 CHB 边沿都可以置位 OVUNF (QEI_STATUS[2])。

6.14.5.8 向下计数

改变计数方向会触发向下计数，此时 DIRF (QEI_STATUS[8]) 会清 0 且 DIRCHGF (QEI_STATUS[3]) 会置为 1，CNT (QEI_CNT[31:0]) 则会开始向下计数。对于自由计数模式，向下计数到 0 时，脉冲计数器会重载为 0xFFFF_FFFF，OVUNF (QEI_STATUS[2]) 会在下一个边沿置 1。对于比较计数模式，向下计数到 0 时，脉冲计数器会重载为 CNTMAX (QEI_CNTMAX[31:0])，OVUNF (QEI_STATUS[2]) 会在下一个边沿置 1。X2 模式下，只在 CHA 边沿置位 OVUNF (QEI_STATUS[2])。而 X4 模式下，CHA 和 CHB 边沿都可以置位 OVUNF (QEI_STATUS[2])。

6.14.5.9 比较功能

QEI 控制器通过比较 CNT (QEI_CNT[31:0]) 和比较寄存器 CNTCMP (QEI_CNTCMP[31:0]) 实现动态计数。CNT (QEI_CNT[31:0]) 向上/下计数到 CNTCMP (QEI_CNTCMP[31:0])，CMPF (QEI_STATUS[1]) 置位。写 1 到 CMP_EN (QEI_CTL[28]) 使能比较功能，写0则禁止该功能。

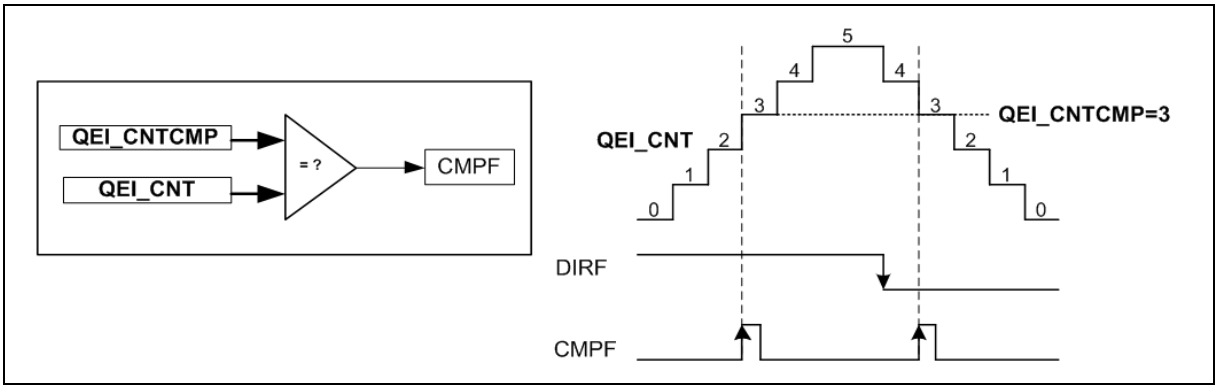


图 6.14-8 比较操作

6.14.5.10 通过 IDX 管脚重载计数器

CHX 信号 (来自滤波器和极性控制输出管脚 IDX) 可以触发 CNT (QEI_CNT[31:0]) 复位为零或是重载为 CNTMAX (QEI_CNTMAX[31:0])。计数器向上计数时，置位 IDX 重载位 IDXRLD_EN (QEI_CTL[27])，

CHX 上的上升沿会导致 QEI 控制器复位 CNT (QEI_CNT[31:0]) 到 0。计数器向下计数时， CHX 上的上升沿会导致 QEI 控制器重载 CNT (QEI_CNT[31:0]) 为 CNTMAX (QEI_CNTMAX[31:0])。具体参考图 6. 14-9。

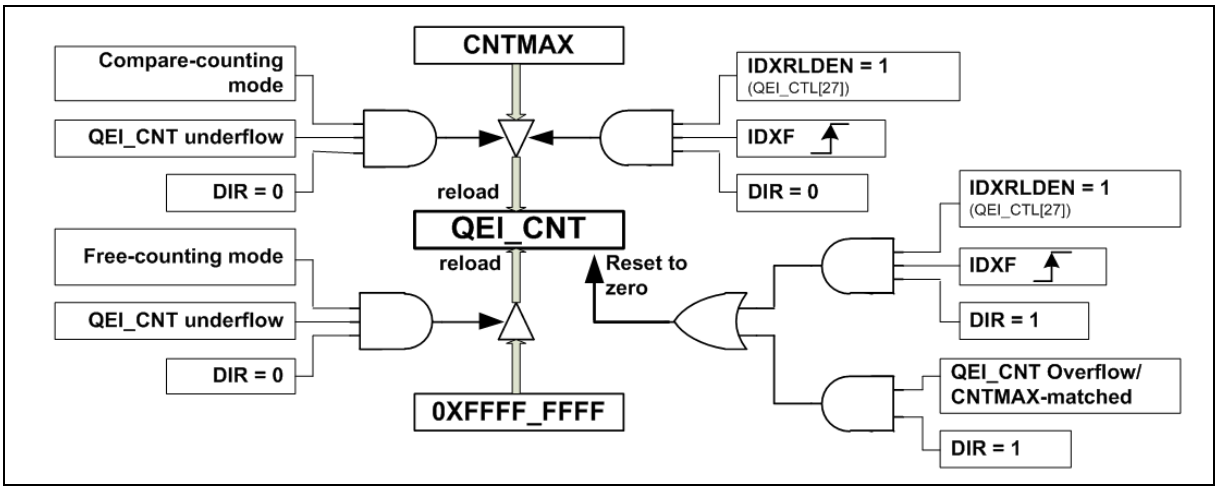


图 6.14-9 QEI_CNT 重载/复位控制

6.14.5.11 捕捉 QEI 计数器

如果HOLDCNT (QEI_CTL[24]) 置位，CNT (QEI_CNT[31:0]) 的值将会被捕捉到 QEI 计数器保持寄存器 CNTHOLD (QEI_CNTHOLD[31:0])，直到下一个 HOLDCNT (QEI_CTL[24]) 触发信号。HOLDCNT 位可以由软件写 1 或者用定时器上升沿中断标志位 TIF (TIMERx_INTSTS[0]) 置位。

注：CNTHOLD (QEI_CNTHOLD[31:0]) 捕捉到 QEI 计数器的值后，HOLDCNT 由硬件自动清 0 。

如果 IDXLATEN (QEI_CTL[25]) 置位，CNT (QEI_CNT[31:0]) 的内容会在每个 CHX 上升沿信号被锁存到 QEI 计数器索引锁存寄存器 CNTLATCH(QEI_CNTLATCH[31:0])。

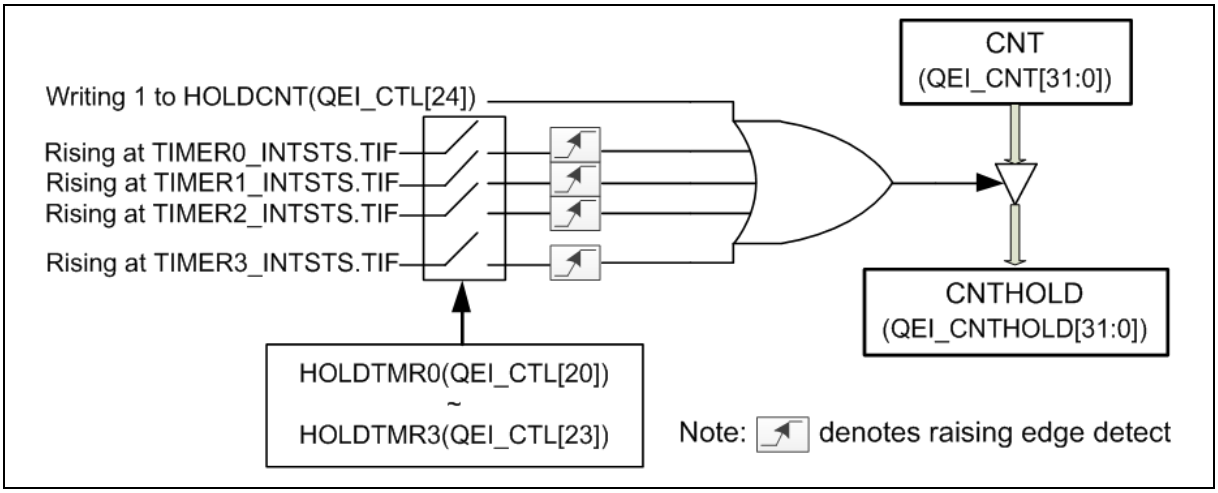


图 6.14-10 捕捉 QEI 计数器的触发控制

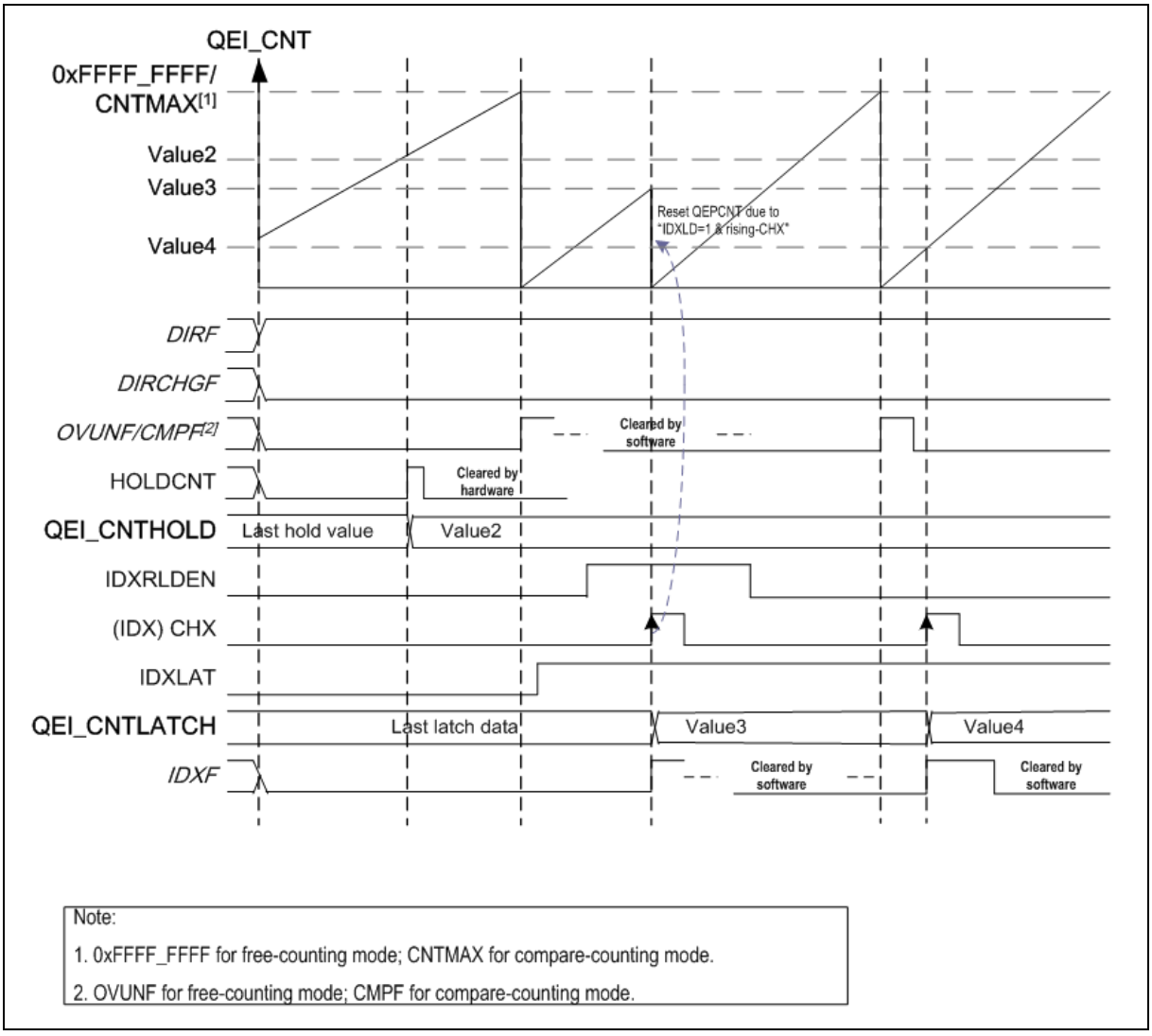


图 6.14-11 捕捉和锁存 QEI 计数器

*6.14.5.12 QEI*中断结构

QEI 控制器有 4 个中断触发源，每个都有对应的中断标志和使能控制位。当 QEI 计数器向上计数 CNT (QEI_CNT[31:0]) 溢出，或者向下计数 CNT (QEI_CNT[31:0]) 下溢时，溢出/下溢标志 OVUNF (QEI_STATUS[2]) 会由硬件置位，如果此时 OVUNIEN (QEI_CTL[16]) 为高，就会触发 QEI 中断请求。当 QEI 控制器检测到了编码器换向时，会触发方向标志 DIRF (QEI_STATUS[8])，硬件会自动置位换向标志 DIRCHGF (QEI_STATUS[3])，如果此时 DIRIEN (QEI_CTL[17]) 位高，就会触发 QEI 中断请求。当 QEI 计数器计数到 QEI 计数器比较寄存器 CNTCMP (QEI_CNTCMP[31:0]，硬件会自动置位 CMPF (QEI_STATUS[1])，如果此时 CMPIEN (QEI_CTL[18]) 位高，就会触发 QEI 中断请求。当 QEI 控制器检测到了 CHX (管脚 IDX 经过滤波器和极性控制的输出) 上的上升沿信号，硬件会自动置位 IDXF (QEI_STATUS[1])，如果此时 IDXIEN (QEI_CTL[19]) 位高，就会触发 QEI 中断请求。需要注意，OVUNF (QEI_STATUS[2]), DIRCHGF (QEI_STATUS[3]), CMPF (QEI_STATUS[1]) 和 IDXF (QEI_STATUS[0]) 这四个标志由硬件自动置位，必须由软件清 0。如图6.14-12是 QEI 控制器的中断架构。

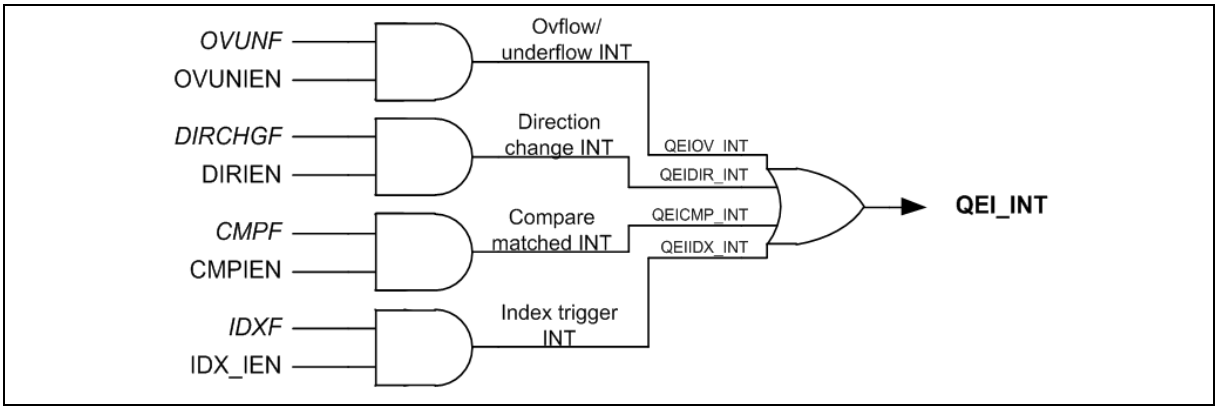


图 6.14-12QEI 中断架构图

6.14.6 寄存器映射

R: 只读, **W**: 只写, **R/W**: 读/写

寄存器	偏移	R/W	描述	复位值
QEI基地址: **QEI0_BA = 0x400B_0000** **QEI1_BA = 0x400B_1000**				
QEI_CNT **x=0, 1**	QEIx_BA+0x00	R/W	QEI 计数值寄存器	0x0000_0000
QEI_CNTHOLD **x=0, 1**	QEIx_BA+0x04	R/W	QEI 计数器保持寄存器	0x0000_0000
QEI_CNTLATCH **x=0,1**	QEIx_BA+0x08	R/W	QEI 计数器索引锁存寄存器	0x0000_0000
QEI_CNTCMP **x=0, 1**	QEIx_BA+0x0C	R/W	QEI 计数器比较寄存器	0x0000_0000
QEI_CNTMAX **x=0, 1**	QEIx_BA+0x14	R/W	QEI 预设最大计数值寄存器	0x0000_0000
QEI_CTL **x=0, 1**	QEIx_BA+0x18	R/W	QEI 控制器控制寄存器	0x0000_0000
QEI_STATUS **x=0, 1**	QEIx_BA+0x2C	R/W	QEI 控制器状态寄存器	0x0000_0000

6.14.7 寄存器描述

QEI计数值寄存器 (QEI_CNT)

寄存器	偏移	R/W	描述	复位值
QEI_CNT	QEIx_BA+0x00	R/W	QEI 计数值寄存器	0x0000_0000

31	30	29	28	27	26	25	24
CNT							
23	22	21	20	19	18	17	16
CNT							
15	14	13	12	11	10	9	8
CNT							
7	6	5	4	3	2	1	0
CNT							

位	描述	
[31:0]	CNT	**QEI 计数值** 32 位的向上/下计数器。当检测到有效的脉冲相位，DIRF (QEI_STATUS[8]) 为 1，则计数值递增；DIRF (QEI_STATUS[8]) 为 0，则计数值递减。该寄存器通过积分的方式来计算当前的编码器位置。下列事件会导致脉冲计数器恢复初始化： 1. QEIEN (QEI_CTL[29]) = 0 时，软件可写； 2. QEIEN (QEI_CTL[29]) = 1 且比较计数模式下，计数值与比较值相等时 (上计数时，值回 0)； 3. QEIEN=1 且 IDXRLDEN (QEI_CTL[27])=1 时 IDX 信号发生变化。

QEI计数器保持寄存器(QEI_CNTHOLD)

寄存器	偏移	R/W	描述	复位值
QEI_CNTHOLD	QEIx_BA+0x04	R/W	QEI 计数器保持寄存器	0x0000_0000

31	30	29	28	27	26	25	24
CNTHOLD							
23	22	21	20	19	18	17	16
CNTHOLD							
15	14	13	12	11	10	9	8
CNTHOLD							
7	6	5	4	3	2	1	0
CNTHOLD							

位	描述	
[31:0]	CNTHOLD	**QEI 计数器保持值** HOLDCNT (QEI_CTL[24])由低到高时，CNT (QEI_CNT[31:0])的值会被复制到CNTHOLD (QEI_CNTHOLD[31:0])寄存器。

QEI计数器索引锁存寄存器(QEI_CNTLATCH)

寄存器	偏移	R/W	描述	复位值
QEI_CNTLATCH	QEIx_BA+0x08	R/W	QEI计数器索引锁存寄存器	0x0000_0000

31	30	29	28	27	26	25	24
CNTLATCH							
23	22	21	20	19	18	17	16
CNTLATCH							
15	14	13	12	11	10	9	8
CNTLATCH							
7	6	5	4	3	2	1	0
CNTLATCH							

位	描述	
[31:0]	CNTLATCH	**QEI 计数器索引锁存** 当IDXF (QEI_STATUS[0])置位，CNT (QEI_CNT[31:0]) 的值会被复制到CNTLATCH (QEI_CNTLATCH[31:0])寄存器。

QEI计数器比较寄存器(QEI_CNTCMP)

寄存器	偏移	R/W	描述	复位值
QEI_CNTCMP	QEIx_BA+0x0C	R/W	QEI计数器比较寄存器	0x0000_0000

31	30	29	28	27	26	25	24
CNTCMP							
23	22	21	20	19	18	17	16
CNTCMP							
15	14	13	12	11	10	9	8
CNTCMP							
7	6	5	4	3	2	1	0
CNTCMP							

位	描述	
[31:0]	CNTCMP	**QEI 计数器比较值** 如果 QEI 控制器处于比较计数模式CMPEN (QEI_CTL[28]) =1，当CNT (QEI_CNT[31:0])的值计数到CNTCMP (QEI_CNTCMP[31:0])，CMPF会被置位，该寄存器软件可写。

QEI预设最大计数值寄存器(QEI_CNTMAX)

寄存器	偏移	R/W	描述	复位值
QEI_CNTMAX	QEIx_BA+0x14	R/W	QEI 预设最大计数值寄存器	0x0000_0000

31	30	29	28	27	26	25	24
CNTMAX							
23	22	21	20	19	18	17	16
CNTMAX							
15	14	13	12	11	10	9	8
CNTMAX							
7	6	5	4	3	2	1	0
CNTMAX							

位	描述	
[31:0]	CNTMAX	**QEI 预设最大计数值** 该寄存器由用户设定，是比较计数模式下的最大值。

QEI控制器控制寄存器(QEI_CTL)

寄存器	偏移	R/W	描述	复位值
QEI_CTL	QEIx_BA+0x18	R/W	QEI控制器控制寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved		QEIEN	CMPEN	IDXRLDEN	Reserved	IDXLATEN	HOLDCNT
23	22	21	20	19	18	17	16
HOLDTMR3	HOLDTMR2	HOLDTMR1	HOLDTMR0	IDXIEN	CMPIEN	DIRIEN	OVUNIEN
15	14	13	12	11	10	9	8
Reserved	IDXINV	CHBINV	CHAINV	Reserved		MODE	
7	6	5	4	3	2	1	0
Reserved	IDXEN	CHBEN	CHAEN	NFDIS	NFCLKSEL		

位	描述	
[31:30]	**Reserved**	保留
[29]	**QEIEN**	**QEI 控制器使能位** 0 = 禁止QEI 控制器功能 1 = 使能QEI 控制器功能
[28]	**CMPEN**	**比较功能使能位** QEI 比较功能用于动态比较 QEI_CNT 和CNTCMP (QEI_CNTCMP[31:0])，如果CNT (QEI_CNT[31:0])计数到CNTCMP (QEI_CNTCMP[31:0])，该位 CMPF 置 1 0 =禁止比较功能 1 =使能比较功能
[27]	**IDXRLDEN**	**索引触发 QEI_CNT 重载使能位** 向上计数时 (DIRF (QEI_STATUS[8]) = 1)，若该位为高，检测到CHX的上升沿时会复位CNT (QEI_CNT[31:0])为 0。向下计数时 (DIRF (QEI_STATUS[8]) = 0)，若该位为高，检测到 CHX 的上升沿时会重载 CNT (QEI_CNT[31:0]) 为 CNTMAX (QEI_CNTMAX[31:0])。 0 = 禁止重载功能 1 = 使能索引信号初始化 QEI_CNT 功能
[26]	**Reserved**	保留
[25]	**IDXLATEN**	**索引锁存 QEI_CNT 使能位** 该位置位时，CHX 信号的每个上升沿都会锁存 CNT (QEI_CNT[31:0]) 的值到CNTLATCH (QEI_CNTLATCH[31:0])。 0 = 禁止索引信号锁存 QEI 计数器值功能 1 = 使能索引信号锁存 QEI 计数器值功能

[24]	**HOLDCNT**	**保持 QEI_CNT 控制** 当该位由低到高时，CNT (QEI_CNT[31:0]) 会被复制到CNTHOLD (QEI_CNTHOLD[31:0])。写 1 到该位或是 Timer0~Timer3 中断标志 TIF (TIMERx_INTSTS[0]) 会置位该位。 0 = 无操作. 1 = QEI_CNT 值会被捕捉和存储到 CNTHOLD (QEI_CNTHOLD[31:0]). **注：**QEI_CNTHOLD 保持 QEI_CNT 后，该位自动清 0 。
[23]	**HOLDTMR3**	**Timer3 保持功能使能位** 0 = TIF (TIMER3_INTSTS[0]) 不影响HOLDCNT 1 = TIF (TIMER3_INTSTS[0]) 置 1 时，会把HOLDCNT 置为1
[22]	**HOLDTMR2**	**Time 2保持功能使能位** 0 = TIF (TIMER2_INTSTS[0]) 不影响HOLDCNT 1 = TIF (TIMER2_INTSTS[0])置 1 时，会把HOLDCNT 置为1
[21]	**HOLDTMR1**	**Timer1保持功能使能位** 0 = TIF (TIMER1_INTSTS[0]) 不影响HOLDCNT 1 = TIF (TIMER1_INTSTS[0])置 1 时，会把HOLDCNT 置为1
[20]	**HOLDTMR0**	**Timer0保持功能使能位** 0 = TIF (TIMER0_INTSTS[0]) 不影响HOLDCNT 1 = TIF (TIMER0_INTSTS[0])置 1 时，会把HOLDCNT 置为1
[19]	**IDXIEN**	**IDXF 触发QEI中断使能位** 0 = 关 IDXF中断 1 = 开 IDXF中断
[18]	**CMPIEN**	**CMPF 触发 QEI 中断使能** 0 =关 CMPF 中断 1 =开 CMPF 中断
[17]	**DIRIEN**	**DIRCHGF 触发 QEI 中断使能** 0 =关DIRCHGF中断 1 =开DIRCHGF中断
[16]	**OVUNIEN**	**OVUNF 触发 QEI 中断使能** 0 =关 OVUNF 中断 1 = 开 OVUNF中断
[15]	**Reserved**	保留
[14]	**IDXINV**	**IDX 反相** 0 = IDX 不反相 1 = IDX反相
[13]	**CHBINV**	**QEB反相** 0 = 不反相 1 = QEB 反相
[12]	**CHAINV**	**QEA 反相** 0 =不反相 1 = QEA 反相

[11:10]	**Reserved**	保留
[9:8]	**MODE**	**QEI 计数模式选择** 00 = X4 自由计数模式 01 = X2 自由计数模式 10 = X4 比较计数模式 11 = X2 比较计数模式
[7]	**Reserved**	保留
[6]	**IDXEN**	**IDX 输入使能** 0 = 关 IDX 输入 1 =开通 IDX 输入
[5]	**CHBEN**	**QEB 输入使能** 0 = 关 QEB 输入 1 = 开通 QEB 输入
[4]	**CHAEN**	**QEA 输入使能** 0 = 关 QEA 输入 1 = 开通 QEA 输入
[3]	**NFDIS**	**QEI 噪声滤波器禁止位** 0 = 使能噪声滤波功能 1 = 禁噪声滤波
[2:0]	**NFCLKSEL**	**噪声滤波器时钟分频选择** 000 = QEI_CLK 001 = QEI_CLK/2 010 = QEI_CLK/4 011 = QEI_CLK/16 100 = QEI_CLK/32 101 = QEI_CLK/64

QEI控制器状态寄存器(QEI_STATUS)

寄存器	偏移	R/W	描述	复位值
QEI_STATUS	QEIx_BA+0x2C	R/W	QEI 控制器状态寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							DIRF
7	6	5	4	3	2	1	0
Reserved				DIRCHGF	OVUNF	CMPF	IDXF

位	描述	
[31:9]	**Reserved**	保留
[8]	**DIRF**	**QEI 计数方向指示位** 0 = 向下计数 1 = 向上计数 **注：**硬件按 CHA 和 CHB 的超前或滞后，决定此位的值。
[7:4]	**Reserved**	保留
[3]	**DIRCHGF**	**方向改变标志** 计数方向改变时此位由硬件自动置 1，软件写 1 可清 0。 0 = 计数方向未改变. 1 = 计数方向已改变. **注:**该位写 1 清 0.
[2]	**OVUNF**	**QEI 溢出/下溢标志** 自由计数模式计数值 QEI_CNT 从 0xFFFF_FFFF 变 0 时，比较计数模式从 QEI_CNTMAX 变 0 时；或反之，从 0 变为 0xFFFF_FFFF 或 QEI_CNTMAX 时，此位置 1。 0 = QEI 计数器未溢出/下溢 1 = QEI 计数器已溢出/下溢 **注：**该位写 1 清 0。
[1]	**CMPF**	**比较匹配标志** 如果 QEI 比较功能使能，QEI 计数器向上/下计数到 CNTCMP (QEI_CNTCMP[31:0])，该位由硬件自动置位。 0 = QEI 未计数到 CNTCMP (QEI_CNTCMP[31:0]) 1 = QEI 计数到CNTCMP (QEI_CNTCMP[31:0]) **注：**该位写 1 清 0。

[0]	**IDXF**	**IDX 检测标志** CHX 信号上检测到上升沿，该标志置位 0 = CHX 信号上检未测到上升沿 1 = CHX 信号上检测到上升沿 **注：**该位写 1 清 0。

6.15 增强型输入捕捉定时器(ECAP)

6.15.1 概述

M480 系列提供了多达两组的增强型输入捕捉定时器/计数器，以用于检测输入通道边沿信号的改变。每个单元含有三路输入通道。定时器/计数器有向上计数、重载及比较匹配的功能。

6.15.2 特性

- 多达两组输入捕捉定时器/计数器单元，CAP0和CAP1
- 每个单元都有3个输入通道
- 每个单元都有独立的中断向量
- 每个输入通道都有对应的捕捉计数器保持寄存器
- 24位的输入捕捉向上计数定时器/计数器
- 在输入端末前附有噪声过滤
- 三种边沿检测：
  - 上升沿检测
  - 下降沿检测
  - 双边沿检测
- 支持用被捕捉的事情复位/重载捕捉计数器
- 支持比较匹配功能

6.15.3 框图

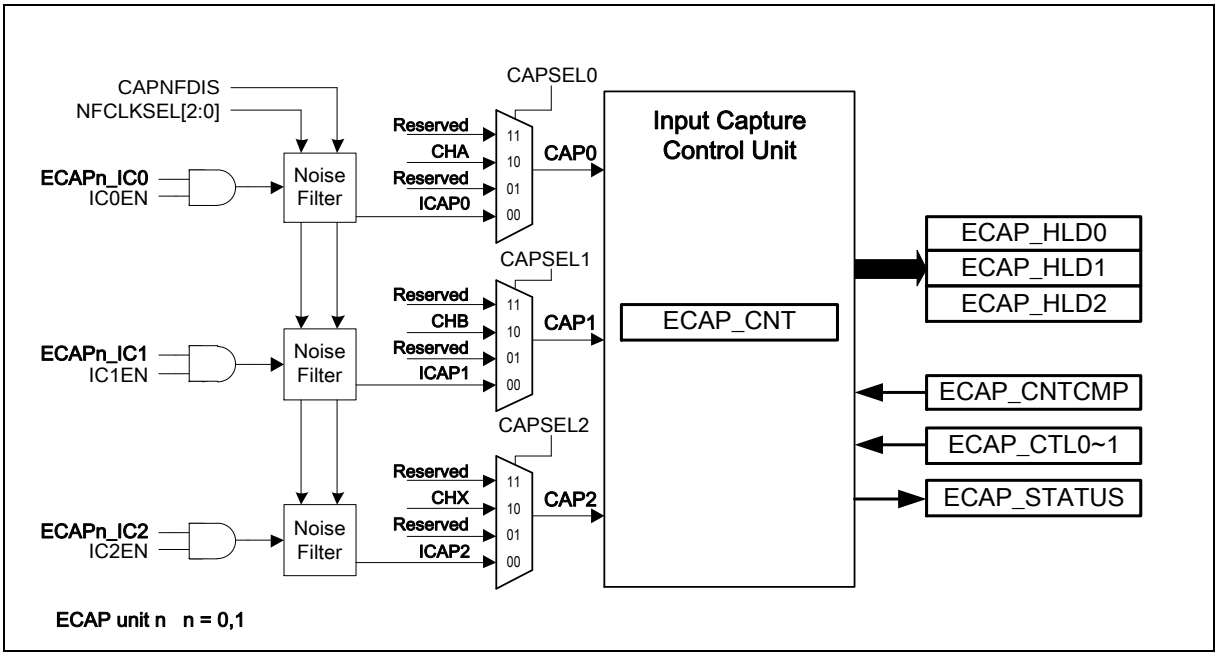


图 6.15-1 输入捕捉定时器/计数器架构

6.15.4 基本配置

6.15.4.1 ECAP0 基本配置

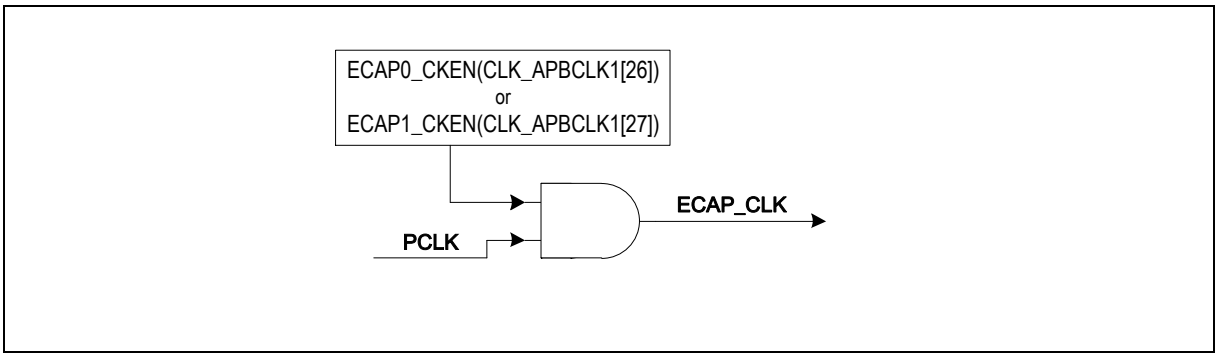
- 时钟源配置
  - 在寄存器 ECAP0CKEN (CLK_APBCLK1[26]) 使能 ECAP0 外设时钟
- 复位配置
  - 在寄存器ECAP0RST (SYS_IPRST2[26])复位ECAP0
- 管脚配置

组	引脚名	GPIO	MFP
ECAP0	ECAP0_IC0	PA.10	MFP11
		PE.8	MFP12
	ECAP0_IC1	PA.9	MFP11
		PE.9	MFP12
	ECAP0_IC2	PA.8	MFP11
		PE.10	MFP12

6.15.4.2 ECAP1基本配置

- 时钟源配置
  - 在寄存器ECAP1CKEN (CLK_APBCLK1[27])使能 ECAP1 外设时钟
- 复位配置
  - 在寄存器ECAP0RST (SYS_IPRST2[27])复位ECAP1
- 管脚配置

组	引脚名	GPIO	MFP
ECAP1	ECAP1_IC0	PC.10	MFP11
		PE.13	MFP13
	ECAP1_IC1	PC.11	MFP11
		PE.12	MFP13
	ECAP1_IC2	PC.12	MFP11
		PE.11	MFP13



6.15.5 功能描述

图 6.15-2 输入捕捉定时器/计数器时钟源控制

图6.15-1是输入捕捉的架构图。每个输入捕捉定时器/计数器单元支持三个输入信号源可编程的输入通道。管脚ECAP_IC0 到ECAP_IC2可以通过噪声滤波器或者直接 (CAPNFDIS = 1)作为输入捕捉单元的信号源，此外，QEI 控制器的输入信号 (CHA，CHB，CHX) 也可以根据软件设置ECAP_CTL0 (CAPSEL0~ CAPSEL 2) 从内部通路连接到捕捉输入。

6.15.5.1 输入噪声滤波器

输入噪声滤波器的结构类似 QEI 的输入滤波器(参考图6.14-3)。ECAP 有 6 个采样率选择，PCLK 运行在 90 MHz 时，ECAP 支持从 33 ns (NFCLKSEL = 000) 到 2.1 us (NFCLKSEL = 101) 的持续时间选择。表6.15-1列出了 NFCLKSEL 、最大噪声持续时间和最小能被采样的信号持续时间的相关设置。

最大可以被滤掉的噪声持续时间不超过 3 个采样时钟周期，最小可以被采样的信号持续时间需超过 4 个采样时钟周期。

NFCLKSEL	最大噪音持续时间	最小能被采样信号持续时间
000	33 ns	44 ns
001	67 ns	89 ns
010	133 ns	178 ns
011	533 ns	711 ns
100	1067 ns	1422 ns
101	2133 ns	2844 ns
		PCLK= 90MHz

表 6.15-1 一个典型的滤波设置

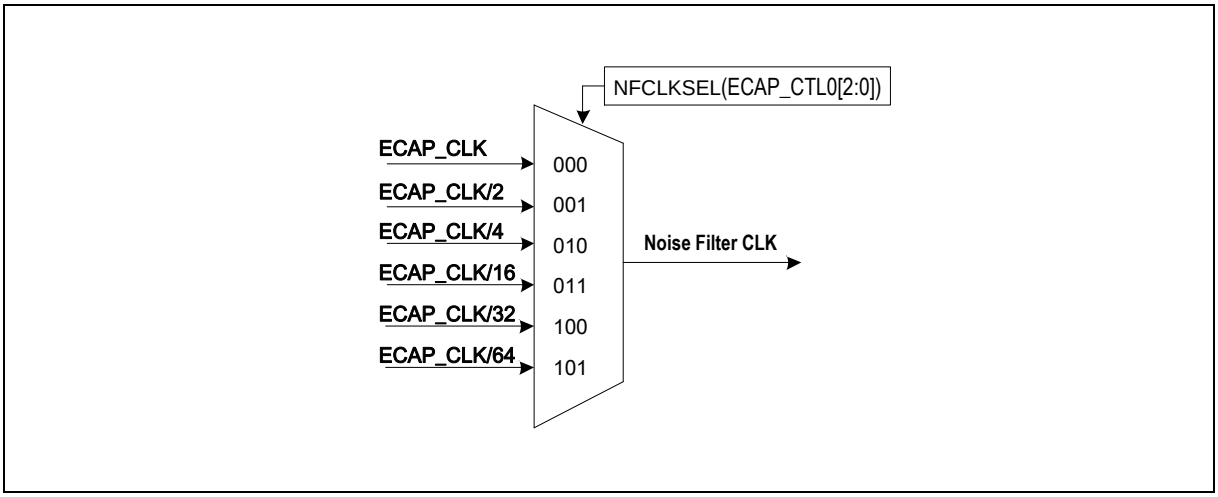


图 6.15-3 噪声滤波器采样时钟选择

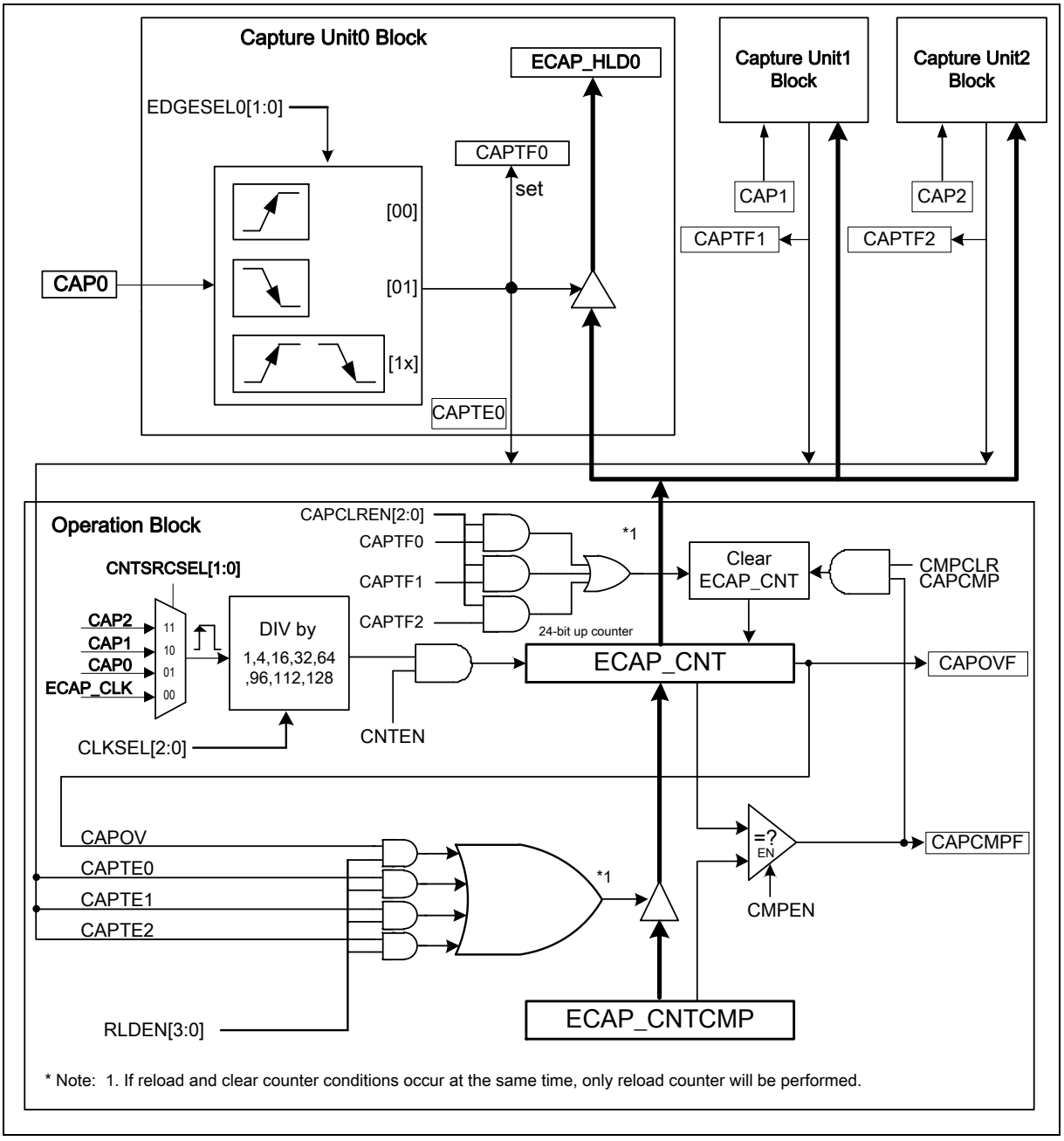


图 6.15-4 输入捕捉定时器/计数器功能框图

6.15.5.2 输入捕捉定时器/计数器操作

输入捕捉定时器/计数器单元包含 2 个主要的功能模块：捕捉模块和操作模块。3 路捕捉通道共有 3 个捕捉单元。

捕捉单元功能检测脉冲宽度和方波周期。输入通道 0~2 在捕捉模块有他们自己的检测器，但是在操作模块里共享一个捕捉定时器/计数器，ECAP_CNT。边沿触发选项由 EDGESEL (ECAP_CTL1[5,4], [3,2], [1,0]) 寄存器配置，可以设成上升沿、下降沿和双边沿触发。每个捕捉单元包含一个使能控制位，IC0EN ~ IC2EN (ECAP_CTL0[6:4])，以使能/禁止每个输入通道和状态位 CAP0 ~ CAP2 (ECAP_STATUS[10:8])，从而让软件监控当前各通道的状态。

输入捕捉支持重载模式和比较模式。两个模式里，捕捉计数器 (ECAP_CNT) 都可以作为 24 位的向上计数器，其时钟分频可通过 CLKSEL[2:0] 设为除 1,4,16,32,64,96,112 和 128，时钟源则可以通过 CNTSRCSEL[1:0]) 设置为系统时钟源、CAP_CLK 或者输入通道 CAP0 ~ CAP2。而在重载模式里，ECAP_CNTCMP 作为重载寄存器，比较模式里 ECAP_CNTCMP 则作为比较寄存器。输入捕捉定时器/计数器使能位 CAPEN 在输入捕捉定时器/计数器功能使用时一定要使能。更多操作细节参见后文。

捕获功能

每当检测到有效的边沿改变信号时，就会触发捕捉事件 (CAPTE0~2)，工作中的 24 位捕捉计数器的值 ECAP_CNT 会被捕捉并传送到被触发通道的捕捉暂存寄存器 ECAP_HOLD0~2 中。触发动作还会使 CAPTFx (ECAP_STATUS[2:0]) 标志位置位，并产生一个中断信号 (当 CAPIENx (ECAP_CTL0[18:16]) 使能时)。触发标志由硬件置位，软件清 0。软件可以去读寄存器 ECAP_STATUS 来读取标志的状态，再写 1 到 ECAP_STATUS 对应位去清 0 标志。

设置 CAPxCLREN (ECAP_CTL1[22:20]) 位后，在发生捕捉事件后会硬件会自动复位捕捉计数器 ECAP_CNT。

比较模式

置 CMPEN (ECAP_CTL0[28]) 位为 1 将会使能比较模式。此时，ECAP_CNTCMP 将作为比较寄存器。随着 ECAP_CNT 递增，一旦达到了 ECAP_CNTCMP 的值，CMPF (ECAP_STATUS[4]) 将被置位，同时若使能了捕捉比较中断位 CMPIEN (ECAP_CTL0[21]) 将会会产生一个中断请求。

设置 CMPCLR (ECAP_CTL0[25]) 位会让硬件在比较-匹配成功后自动复位捕捉计数器。

重载模式

输入捕捉定时器/计数器还可以配置成重载模式。置 RLDEN[3:0] (ECAP_CTL1[11:8]) 位为 1，也就是 OVRLDEN, CAPxRLDEN 为 1 将会使能重载源。

该模式下，ECAP_CNTCMP 作为重载寄存器。ECAP_CNT 溢出后，如果已置位 OVRLDEN，将产生重载信号使得 ECAP_CNTCMP 寄存器的值重载到 ECAP_CNT 寄存器里；另外 CAPxRLDEN, x=0~2 可以配置 CAPTE2 ~ CAPTE0 为重载源。

还需要注意的是如果 CAPxCLREN 和 CAPxRLDEN 都置位了，当触发事件 CAPTEx 到达，只有 RELAOD 功能会被执行。

6.15.5.3 输入捕捉定时器/计数器中断架构

图6.15-5是输入捕捉定时器/计数器中断架构示例。总共有 5 个中断源 (OVF_INT, CMP_INT, CAPTF0_INT~CAPTF2_INT)，共同处于一个输入捕捉单元，每个都有一个中断标志 (CAPOVF, CAPCMPF, CAPTF0~CAPTF2) 来触发中断 (ECAP_INT)。可以用使能控制位 (OVIEN, CMPIEN, CAPIEN0~CAPIEN2) 来使能/禁止这些标志。

需要注意的是所有的中断标志都由硬件自动置位，必须由软件写 1 到 (ECAP_STATUS[5:4],[2:0]) 清 0 对应标志。

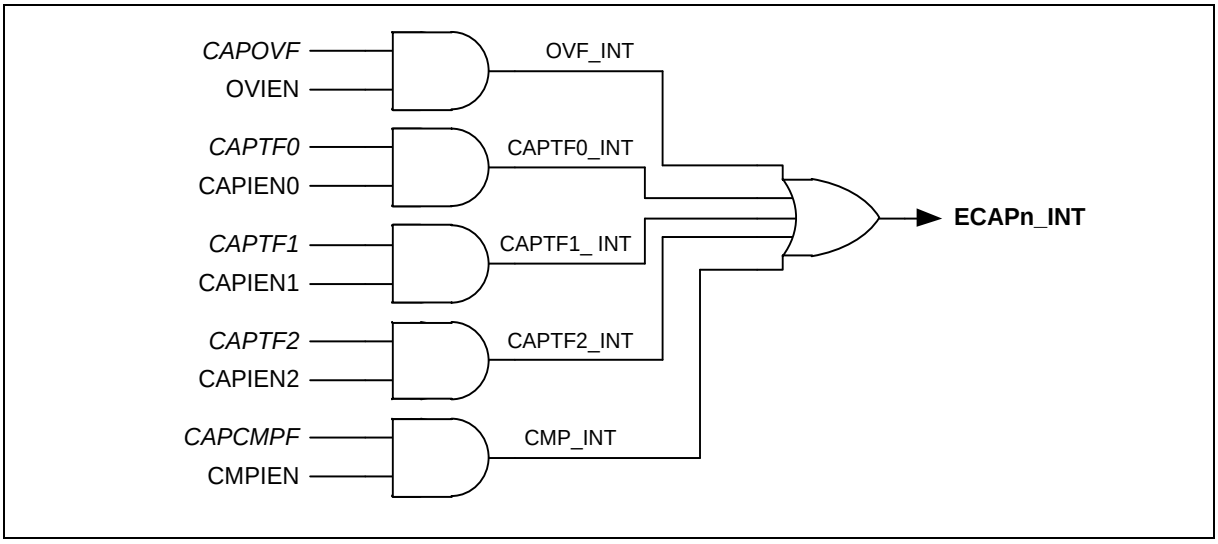


图 6.15-5 输入捕捉定时器/计数器中断架构图

6.15.6 寄存器映射

R: 只读，W: 只写，R/W: 读/写

寄存器	偏移量	R/W	描述	复位值
ECAP 基地址: **ECAPn_BA = 0x400B_4000 + (0x0000_1000 * n)** **n=0, 1**				
ECAP_CNT	ECAPn_BA+0x00	R/W	输入捕捉计数器 (24 位向上计数)	0x0000_0000
ECAP_HLD0	ECAPn_BA+0x04	R/W	输入捕捉保持寄存器 0	0x0000_0000
ECAP_HLD1	ECAPn_BA+0x08	R/W	输入捕捉保持寄存器 1	0x0000_0000
ECAP_HLD2	ECAPn_BA+0x0C	R/W	输入捕捉保持寄存器 2	0x0000_0000
ECAP_CNTCMP	ECAPn_BA+0x10	R/W	输入捕捉比较寄存器	0x0000_0000
ECAP_CTL0	ECAPn_BA+0x14	R/W	输入捕捉控制寄存器 0	0x0000_0000
ECAP_CTL1	ECAPn_BA+0x18	R/W	输入捕捉控制寄存器 1	0x0000_0000
ECAP_STATUS	ECAPn_BA+0x1C	R/W	输入捕捉状态寄存器	0x0000_0000

6.15.7 寄存器描述

输入捕获计数器(ECAP_CNT)

寄存器	偏移量	R/W	描述	复位值
ECAP_CNT	ECAPn_BA+0x00	R/W	输入捕捉计数器 (24 位向上计数)	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
CNT							
15	14	13	12	11	10	9	8
CNT							
7	6	5	4	3	2	1	0
CNT							

位	描述	
[31:24]	Reserved	保留
[23:0]	CNT	**输入捕捉定时器/计数器** 输入捕捉定时器/计数器是一个 24 位的向上计数器。其时钟源来自时钟分频器的输出。

输入捕获计数器保持寄存器(ECAP_HLD0~2)

寄存器	偏移量	R/W	描述	复位值
ECAP_HLD0	ECAPn_BA+0x04	R/W	输入捕捉保持寄存器 0	0x0000_0000
ECAP_HLD1	ECAPn_BA+0x08	R/W	输入捕捉保持寄存器 1	0x0000_0000
ECAP_HLD2	ECAPn_BA+0x0C	R/W	输入捕捉保持寄存器 2	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
HOLD							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
HOLD							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
HOLD							

位	描述	
[31:24]	**Reserved**	保留
[23:0]	**HOLD**	**输入捕捉计数器 Hold 寄存器** 当一个使能的输入捕捉通道检测到了以有效的边沿信号改变，ECAP_CNT 的值将会被锁定在对应的保持寄存器。每个输入通道都有对应的的保持寄存器，按序号 0~2 命名为 ECAP_HOLDx 来表示输入来源于 IC0~IC2 的哪一个。

输入捕获计数器比较寄存器(ECAP_CNTCMP)

寄存器	偏移量	R/W	描述	复位值
ECAP_CNTCMP	ECAPn_BA+0x10	R/W	输入捕捉比较寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
CNTCMP							
15	14	13	12	11	10	9	8
CNTCMP							
7	6	5	4	3	2	1	0
CNTCMP							

位	描述	
[31:24]	**Reserved**	保留
[23:0]	**CNTCMP**	**输入捕捉计数器比较寄存器** 如果使能了比较功能 (CMPEN= 1)，ECAP_CNTCMP 被用作与捕捉计数器 (ECAP_CNT) 相比较。 如果使能了重载功能 (RLDEN[n] = 1, n=0~3)，溢出事件和捕捉事件会触发硬件自动重载 ECAP_CNTCMP 到 ECAP_CNT。

输入捕获控制寄存器寄存器0 (ECAP_CTL0)

寄存器	偏移量	R/W	描述	复位值
ECAP_CTL0	ECAPn_BA+0x14	R/W	输入捕捉控制寄存器0	0x0000_0000

31	30	29	28	27	26	25	24
Reserved		CAPEN	CMPEN	Reserved		CMPCLREN	CNTEN
23	22	21	20	19	18	17	16
Reserved		CMPIEN	OVIEN	Reserved	CAPIEN2	CAPIEN1	CAPIEN0
15	14	13	12	11	10	9	8
Reserved		CAPSEL2		CAPSEL1		CAPSEL0	
7	6	5	4	3	2	1	0
Reserved	IC2EN	IC1EN	IC0EN	CAPNFDIS	NFCLKSEL		

位	描述	
[31:30]	**Reserved**	保留
[29]	**CAPEN**	**输入捕捉定时器/计数器使能位** 0 = 关闭输入捕捉功能 1 = 使能输入捕捉功能
[28]	**CMPEN**	**比较功能使能位** 输入捕捉定时器/计数器的比较功能会动态比较 ECAP_CNT 的值和比较寄存器 ECAP_CNTCMP 的值，如果 ECAP_CNT 的值达到了 ECAP_CNTCMP，CAPCMPF 标志位会被置位 0 = 关闭比较功能 1 = 使能比较功能
[25]	**CMPCLREN**	**比较-匹配控制清除输入捕捉计数器** 如果该位为 1，比较-匹配事件 (CAPCMPF = 1) 发生，捕捉计数器 (ECAP_CNT) 会被清 0 0 = 比较-匹配事件 (CAPCMPF) 清除捕捉计数器 (ECAP_CNT) 禁止 1 =比较-匹配事件 (CAPCMPF) 清除捕捉计数器 (ECAP_CNT) 使能
[24]	**CNTEN**	**输入捕捉计数器开始** 配置该位为 1，捕捉计数器 (ECAP_CNT) 与输入捕捉时钟 (CAP_CLK) 相同步地开始向上计数 0 = ECAP_CNT 停止计数 1 = ECAP_CNT 开始计数
[23:22]	**Reserved**	保留
[21]	**CMPIEN**	**CAPCMPF 触发输入捕捉中断使能位** 0 = 禁止 CAPCMPF 触发输入捕捉中断 1 = 使能 CAPCMPF 触发输入捕捉中断

位	描述	
[20]	**OVIEN**	**CAPOVF 触发输入捕捉中断使能位** 0 = 禁止 CAPOVF 触发输入捕捉中断 1 = 使能 CAPOVF 触发输入捕捉中断
[19]	**Reserved**	保留
[18]	**CAPIEN2**	**输入捕捉通道 2 中断使能位** 0 = 禁止 CAPTF2 标志触发输入捕捉中断 1 = 使能 CAPTF2 标志触发输入捕捉中断
[17]	**CAPIEN1**	**输入捕捉通道 1 中断使能位** 0 = 禁止 CAPTF1 标志触发输入捕捉中断 1 = 使能 CAPTF1 标志触发输入捕捉中断
[16]	**CAPIEN0**	**输入捕捉通道 0 中断使能位** 0 = 禁止 CAPTF0 标志触发输入捕捉中断 1 = 使能 CAPTF0 标志触发输入捕捉中断
[15:14]	**Reserved**	保留
[13:12]	**CAPSEL2**	**CAP2 输入源选择** 00 = CAP2 输入来自于端口管脚 ICAP2 01 = 保留 10 = CAP2 输入来自于 QEI 控制单元 n 的信号 CHx 11 =保留 **注:**输入比较单元 n 与 QEIn 相对应，这里 n=0~1。
[11:10]	**CAPSEL1**	**CAP1 输入源选择** 00 = CAP1 输入来自于端口管脚 ICAP1 01 = 保留 10 = CAP1 输入来自于 QEI 控制单元 n 的信号 CHB 11 =保留 **注:**输入比较单元 n 与 QEIn 相对应，这里 n=0~1。
[9:8]	**CAPSEL0**	**CAP0 输入源选择** 00 = CAP0 输入来自于端口管脚 ICAP0.】 01 = 保留.】 10 = CAP0 输入来自于 QEI 控制单元 n 的信号 CHA.】 11 =保留.】 **注:**输入比较单元 n 与 QEIn 相对应，这里 n=0~1。
[7]	**Reserved**	保留
[6]	**IC2EN**	**端口管脚 IC2 输入到输入捕捉单元使能位** 0 = 禁止 IC2 输入到输入捕捉单元 1 = 使能 IC2 输入到输入捕捉单元

位	描述	
[5]	**IC1EN**	**端口管脚 IC1 输入到输入捕捉单元使能位** 0 = 禁止 IC1 输入到输入捕捉单元 1 = 使能 IC1 输入到输入捕捉单元
[4]	**IC0EN**	**端口管脚 IC0 输入到输入捕捉单元使能位** 0 = 禁止 IC0 输入到输入捕捉单元 1 = 使能 IC0 输入到输入捕捉单元
[3]	**CAPNFDIS**	**输入捕捉噪声过滤禁止位** 0 = 使能输入捕捉噪声过滤 1 = 禁止输入捕捉噪声过滤
[2:0]	**NFCLKSEL**	**噪声过滤时钟预分频选择** 选择噪声过滤时钟采样频率。 000 = CAP_CLK 001 = CAP_CLK/2 010 = CAP_CLK/4 011 = CAP_CLK/16 100 = CAP_CLK/32 101 = CAP_CLK/64

输入捕获定时器/计数器控制寄存器1 (ECAP_CTL1)

寄存器	偏移量	R/W	描述	复位值
ECAP_CTL1	ECAPn_BA+0x18	R/W	输入捕捉控制寄存器1	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved	CAP2CLREN	CAP1CLREN	CAP0CLREN	Reserved		CNTSRCSEL	
15	14	13	12	11	10	9	8
Reserved	CLKSEL			OVRLDEN	CAP2RLDEN	CAP1RLDEN	CAP0RLDEN
7	6	5	4	3	2	1	0
Reserved		EDGESEL2		EDGESEL1		EDGESEL0	

位	描述	
[31:18]	**Reserved**	保留
[22]	**CAP2CLREN**	**捕捉事件 2 清除捕捉计数器** 0 = 禁止事件 CAPTE2 清除捕捉计数器 (ECAP_CNT) 0 = 使能事件 CAPTE2 清除捕捉计数器 (ECAP_CNT)
[21]	**CAP1CLREN**	**捕捉事件 1 清除捕捉计数器** 0 = 禁止事件 CAPTE1 清除捕捉计数器 (ECAP_CNT) 0 = 使能事件 CAPTE1 清除捕捉计数器 (ECAP_CNT)
[20]	**CAP0CLREN**	**捕捉事件 0 清除捕捉计数器** 0 = 禁止事件 CAPTE0 清除捕捉计数器 (ECAP_CNT) 0 = 使能事件 CAPTE0 清除捕捉计数器 (ECAP_CNT)
[17:16]	**CNTSRCSEL**	**捕捉定时器/计数器时钟源选择** 选择捕捉定时器/计数器时钟源 00 = CAP_CLK (默认) 01 = CAP0 10 = CAP1 11 = CAP2
[15]	**Reserved**	保留

位	描述	
[14:12]	**CLKSEL**	**捕捉定时器时钟分频选择** 捕捉定时器时钟的前置分频器可以由 CLKSEL[2:0] 配置 8 种不同的选项： 000 = CAP_CLK/1 001 = CAP_CLK/4 010 = CAP_CLK/16 011 = CAP_CLK/32 100 = CAP_CLK/64 101 = CAP_CLK/96 110 = CAP_CLK/112 111 = CAP_CLK/128
[11]	**Reserved**	保留
[11]	**OVRLDEN**	**使能溢出事件触发捕捉计数器重载功能** 0 = 禁止 CAPOV 触发重载 1 = 使能 CAPOV 触发重载
[10]	**CAP2RLDEN**	**使能 CAPTE2 事件触发捕捉计数器重载功能** 0 = 禁止 CAPTE2 事件触发重载 1 = 使能 CAPTE2 事件触发重载
[9]	**CAP1RLDEN**	**使能 CAPTE1 事件触发捕捉计数器重载功能** 0 = 禁止 CAPTE1 事件触发重载 1 = 使能 CAPTE1 事件触发重载
[8]	**CAP0RLDEN**	**使能 CAPTE0 事件触发捕捉计数器重载功能** 0 = 禁止 CAPTE0 事件触发重载 1 = 使能 CAPTE0 事件触发重载
[7:6]	**Reserved**	保留
[5:4]	**EDGESEL2**	**通道 2 捕捉边沿选择** 输入捕捉可以检测下降沿变化、上升沿变化及双边沿变化 00 = 检测下降沿 01 = 检测上升沿 1x = 检测双边沿
[3:2]	**EDGESEL1**	**通道 1 捕捉边沿选择** 输入捕捉可以检测下降沿变化、上升沿变化及双边沿变化 00 = 检测下降沿 01 = 检测上升沿 1x = 检测双边沿

位	描述	
[1:0]	**EDGESEL0**	**通道 0 捕捉边沿选择** 输入捕捉可以检测下降沿变化、上升沿变化及双边沿变化 00 = 检测下降沿 01 = 检测上升沿 1x = 检测双边沿

输入捕获定时器/计数器状态寄存器(ECAP_STATUS)

寄存器	偏移量	R/W	描述	复位值
ECAP_STATUS	ECAPn_BA+0x1C	R/W	输入捕捉状态寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved					CAP2	CAP1	CAP0
7	6	5	4	3	2	1	0
Reserved		CAPOVF	CAPCMPF	Reserved	CAPTF2	CAPTF1	CAPTF0

位	描述	
[31:6]	**Reserved**	保留
[8]	**CAP2**	**输入通道 2 的值 CAP2 (只读)** 输入通道 2 CAP2 的值 注：该位只读，写无效
[7]	**CAP1**	**输入通道 1 的值 CAP1 (只读)** 输入通道 1 CAP1 的值 注：该位只读，写无效
[6]	**CAP0**	**输入通道 0 的值 CAP0 (只读)** 输入通道 0 CAP0 的值 注：该位只读，写无效
[5]	**CAPOVF**	**输入捕捉计数器溢出标志** 计数器 (ECAP_CNT) 从 0x00FF_FFFF 溢出到 0，该标志由硬件置位 0 = 上次清 0 后没有溢出事件发生 1 = 上次清 0 后有溢出事件发生 **注:**该位写 1 清 0。
[4]	**CAPCMPF**	**输入捕捉比较-匹配标志** 如果使能了输入捕捉比较功能，当捕捉计数器 (ECAP_CNT) 向上计数到 ECAP_CNTCMP 的值，硬件置位该位。 0 = ECAP_CNT 不等于 ECAP_CNTCMP 的值 1 = ECAP_CNT 等于 ECAP_CNTCMP 的值 **注意:**该位写 1 清 0。
[3]	**Reserved**	保留

位	描述	
[2]	**CAPTF2**	**输入捕捉通道 2 触发标志** 当输入捕捉通道 2 检测到 CAP2 输入上的有效边沿时，CAPTF2 将会被置。 0 = 上次清 0 后 CAP2 输入上未检测到有效边沿 1 = 上次清 0 后 CAP2 输入上检测到了有效边沿 **注意:**该位写 1 清 0。
[1]	**CAPTF1**	**输入捕捉通道 1 触发标志** 当输入捕捉通道 1 检测到 CAP1 输入上的有效边沿时，CAPTF1 将会被置。 0 = 上次清 0 后 CAP1 输入上未检测到有效边沿 1 = 上次清 0 后 CAP1 输入上检测到了有效边沿 **注意:**该位写 1 清 0。
[0]	**CAPTF0**	**输入捕捉通道 0 触发标志** 当输入捕捉通道 0 检测到 CAP0 输入上的有效边沿时，CAPTF0 将会被置。 0 = 上次清 0 后 CAP0 输入上未检测到有效边沿 1 = 上次清 0 后 CAP0 输入上检测到了有效边沿 **注意:**该位写 1 清 0。

6.16 UART接口控制器(UART)

6.16.1 概述

芯片提供8路通用异步收发器(UART)。UART控制器作为标准速度的UART还支持流控功能。UART控制器的接收过程是把外设的串行数据转为并行数据，发送过程是把CPU的并行数据转成串行数据发送出去。每个UART通道支持10种类型的中断。UART控制器还支持IrDA SIR，RS-485和波特率自动测量功能。

6.16.2 特性

- 全双工，异步通讯口
- 独立的接收/发送16/16字节FIFO
- 支持硬件自动流控制
- 接收缓存触发等级的数据长度可设
- 每个通道波特率可单独设置
- 支持nCTS，输入数据，接受FIFO数据达到阈值及RS-485地址匹配 (AAD 模式)唤醒功能
- 支持8位接收缓存定时溢出检测功能
- 通过设置寄存器 DLY (UA_TOR [15:8])，可配置两个数据之间 (从上一个 stop 位到下一个 start 位) 的传送时间间隔
- 支持自动波特率检测和波特率补偿功能
  - UART_CLK 选择 LXT 时支持 9600
- 支持间隔错误，帧错误，校验错误和收/发缓冲区溢出检测等功能
- 可编程串行接口特性
  - 数据位长度可设为 5~8 位
  - 可编程校验，包括奇、偶、无校验，或固定校验位生成和检测
  - 可设置停止位长度为 1 位， 1.5 位或 2 位
- 支持 IrDA SIR 功能模式
  - 标准模式下支持 3/16 位宽功能
- 支持LIN 功能模式 (仅UART0/UART1 支持)
  - 支持 LIN 主/从模式
  - 传输中支持可编程分隔域生成功能
  - 支持接收器分隔域检测功能
- 支持RS-485模式
  - 支持 RS-485 9-位模式
  - 支持软/硬件控制nRTS管脚，用于控制RS-485传送方向
- 支持PDMA传输功能

- 支持单线功能模式

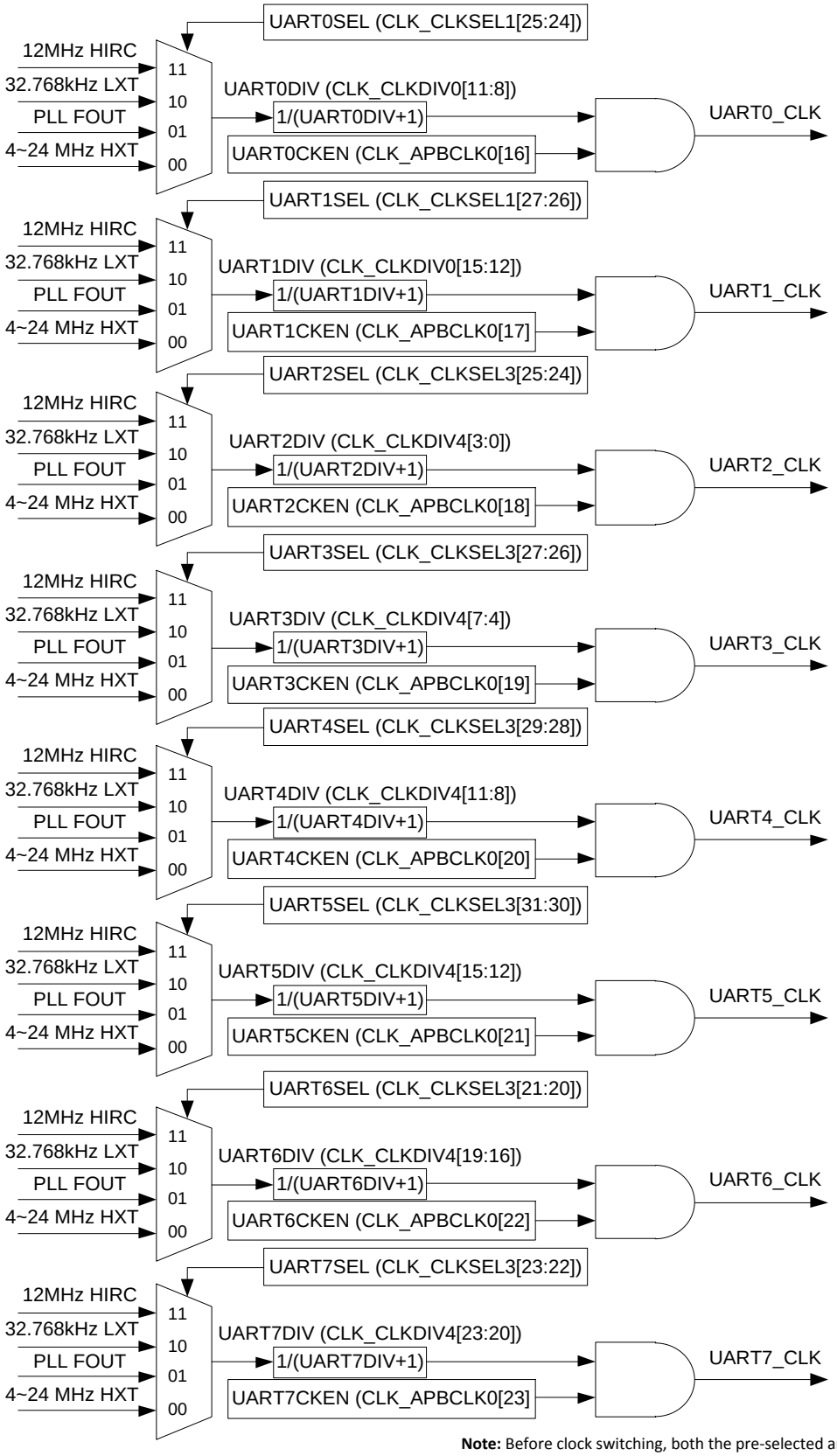
UART 特性	UART0/UART1	UART2 ~ UART7	SC_UART	USCI-UART
FIFO	16字节	16字节	4字节	TX: 1字节 RX: 2字节
自动流控制 (CTS/RTS)	√	√	-	√
IrDA	√	√	-	-
LIN	√	-	-	-
RS-485 功能模式	√	√	-	√
nCTS 唤醒	√	√	-	√
输入数据唤醒	√	√	-	√
接收数据达到FIFO阈值唤醒	√	√	-	-
RS-485 地址匹配 (AAD模式)唤醒	√	√	-	-
自动波特率测量	√	√	-	√
STOP 位长	1, 1.5, 2 位	1, 1.5, 2位	1, 2位	1, 2位
字长	5, 6, 7, 8位	5, 6, 7, 8位	5, 6, 7, 8位	6~13位
奇偶校验	√	√	√	√
Stick 位	√	√	-	-
注: √= 支持				

表 6.16-1 M480 系列 UART 特性

6.16.3 框图

UART 时钟控制和模块框图分别如图6.16-1和图6.16-2所示。

注：UARTx_CLK 的时钟频率不能超过HCLK的30倍。



Note: Before clock switching, both the pre-selected and newly selected clock sources must be turned on and stable.

图 6.16-1 UART 时钟控制框图

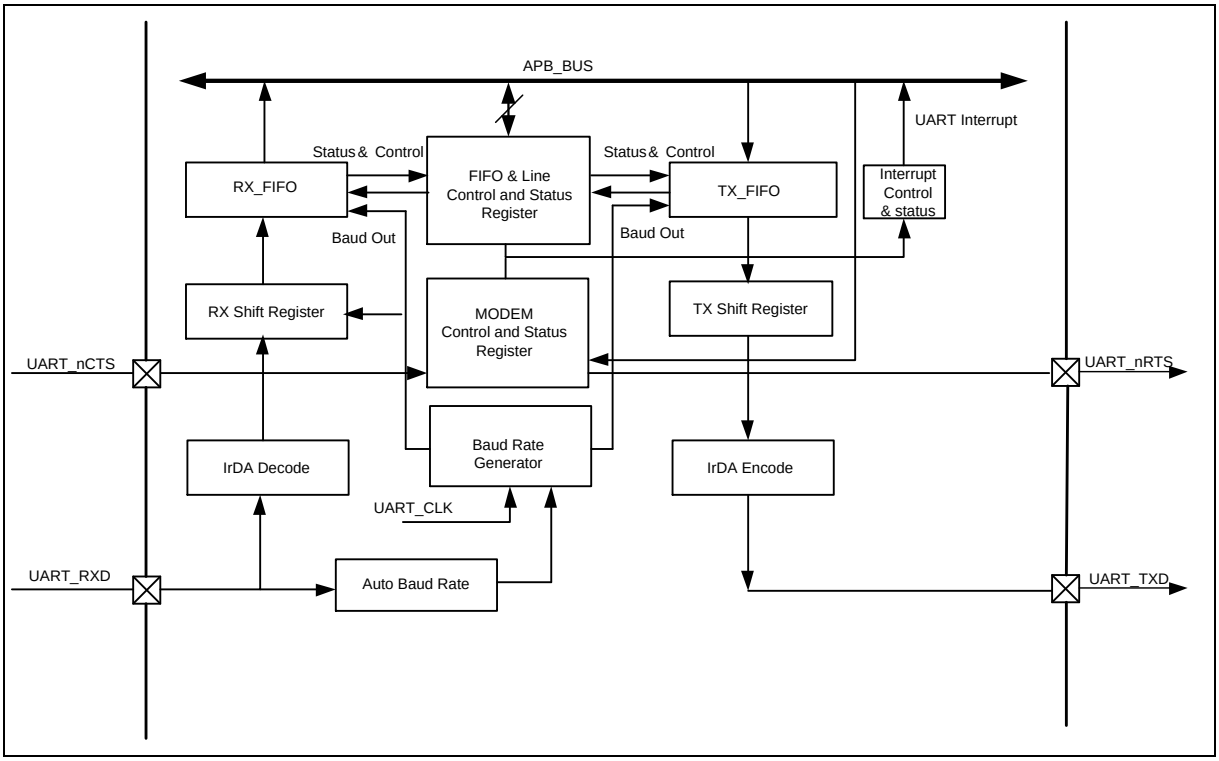


图 6.16-2 UART 结构图

每个模块功能详述如下：

TX_FIFO

发送口带有一个16字节的FIFO缓冲区以减少CPU中断的频率。

RX_FIFO

接收口带有一个16字节的FIFO缓冲区(另加3个出错位，BIF(UART_FIFOSTS[6])，FEF(UART_FIFOSTS[5])，PEF(UART_FIFOSTS[4]))以减少CPU中断的频率。

TX 移位寄存器

该模块用于控制把并行数据串行输出。

RX 移位寄存器

该模块用于控制把串行数据并行输入。

Modem 控制和状态寄存器

该寄存器用于控制到Modem或数传机(或类似于Modem的外设)的接口。

波特率发生器

通过把输入的外部时钟除频得到期望的波特率。详情请参考波特率公式。

IrDA 编码

该模块为IrDA编码控制模块。

IrDA 解码

该模块为IrDA解码控制模块。

FIFO&线控控制和状态寄存器

该寄存器组包括FIFO控制寄存器(UART_FIFO)，FIFO状态寄存器(UART_FIFOSTS)和线控寄存器(UART_LINE)。定时溢出寄存器(UART_TOUT)配置时间溢出中断条件。

波特率自动检测

该模块用于自动检测波特率。

中断控制和状态寄存器

一共有11种类型的中断，包括接收数据可用中断(RDAINT)，发送保持寄存器空中断(THERINT)，发送完成中断(TXENDINT)，接收线状态中断(校验错误或帧错误或break错误)(RLSINT)，MODEM状态中断(MODEMINT)，接收超时中断(RXTOINT)，缓存错误中断(BUFERRINT)，和LIN总线中断(LININT)，唤醒中断(WKINT)，自动波特率中断(ABRINT)和单线位错误检测中断（SWBEINT）。中断使能寄存器(UART_INTEN)用于使能或禁止相应中断，中断状态寄存器(UART_INTSTS)显示哪些中断正在发生。

中断	描述
RDAINT	接收数据可用中断
THERINT	发送保持寄存器空中断
TXENDINT	发送完成中断
RLSINT	接收线状态中断(校验错误或帧错误或break错误)
MODEMINT	MODEM状态中断
RXTOINT	接收缓存超时中断
BUFERRINT	缓存错误中断.
LININT	LIN总线中断
WKINT	唤醒中断
ABRINT	自动波特率中断
SWBEINT	单线位错误检测中断

表 6.16-2 UART 中断

6.16.4 基本配置

*6.16.4.1 UART0*基本配置

- 时钟源配置
  - 在寄存器UART0SEL(CLK_CLKSEL1[26:24])选择UART0外设时钟源
  - 在寄存器UART0DIV(CLK_CLKDIV0[11:8])选择UART0外设时钟分频数

- 在寄存器UART0CKEN(CLK_APBCLK1[16])使能UART0外设时钟

- 用寄存器UART0RST(SYS_IPRST1[16])复位UART0控制器
- 管脚配置

组	管脚名	GPIO	MFP
UART0	UART0_RXD	PA.15, PC.11, PF.2	MFP3
		PF1	MFP4
		PB.8	MFP5
		PB.12	MFP6
		PA.0, PA.6	MFP7
		PH.11	MFP8
		PD.2	MFP9
		PA.4	MFP11
	UART0_TXD	PA.14, PC.12, PF.3	MFP3
		PB.9	MFP5
		PB.13	MFP6
		PA.1, PA.7	MFP7
		PH.10	MFP8
		PD.3	MFP9
		PA.5	MFP11
	UART0_nCTS	PB.11	MFP5
		PB.15	MFP6
		PA.5, PC.7	MFP7
	UART0_nRTS	PB.10	MFP5
		PB.14	MFP6
		PA.4, PC.6	MFP7

*6.16.4.2 UART1*基本配置

- Clock时钟源选择
  - 在寄存器UART1SEL(CLK_CLKSEL1[30:28])配置UART1外设时钟源
  - 在寄存器UART1DIV(CLK_CLKDIV0[15:12])配置UART1外设时钟分频数
  - 在寄存器UART1CKEN(CLK_APBCLK1[17])使能UART1外设时钟
- 用寄存器UART1RST(SYS_IPRST1[17])复位UART1控制器
- 管脚配置

组	管脚名	GPIO	MFP

UART1	UART1_RXD	PF.1	MFP2
		PD.6, PD.10	MFP3
		PB.2, PB.6	MFP6
		PA.8	MFP7
		PA.2, PC.8, PG.1	MFP8
		PH.9	MFP10
	UART1_TXD	PF.0	MFP2
		PD.7, PD.11	MFP3
		PB.3, PB.7	MFP6
		PA.9	MFP7
		PA.3, PE.13, PG.0	MFP8
		PH.8	MFP10
	UART1_nCTS	PB.9	MFP6
		PA.1, PE.11	MFP8
	UART1_nRTS	PB.8	MFP6
		PA.0, PE.12	MFP8

*6.16.4.3 UART2*管脚配置

- 时钟源选择
  - 在寄存器UART2SEL(CLK_CLKSEL3[26:24])配置UART2外设时钟源
  - 在寄存器UART2DIV(CLK_CLKDIV4[3:0])选择UART2外设时钟分频数
  - 在寄存器UART2CKEN(CLK_APBCLK1[18])使能外设时钟
- 寄存器UART2RST(SYS_IPRST1[18])复位UART2控制器
- 管脚配置

组	管脚名	GPIO	MFP
UART2	UART2_RXD	PF.5	MFP2
		PE.15	MFP3
		PG.0	MFP6
		PB.0, PD.12, PE.9	MFP7
		PC.0, PC.4	MFP8
		PB.4	MFP12
	UART2_TXD	PF.4	MFP2
		PE.14	MFP3
		PG.1	MFP6

		PB.1, PC.13, PE.8	MFP7
		PC.1, PC.5	MFP8
		PB.5	MFP12
	UART2_nCTS	PD.9, PF.5	MFP4
		PC.2	MFP8
	UART2_nRTS	PD.8, PF.4	MFP4
		PC.3	MFP8

*6.16.4.4 UART3*管脚配置

- 时钟源选择
  - 在寄存器UART3SEL(CLK_CLKSEL3[27:26])配置UART3外设时钟源
  - 在寄存器UART3DIV(CLK_CLKDIV4[7:4])选择UART3外设时钟分频数
  - 在寄存器UART3CKEN(CLK_APBCLK1[19])使能外设时钟
- 寄存器UART3RST(SYS_IPRST1[19])复位UART3控制器
- 管脚配置

组	管脚名	GPIO	MFP
UART3	UART3_RXD	PD.0	MFP5
		PB.14, PC.9, PE.0, PE.11	MFP7
		PC.2	MFP11
	UART3_TXD	PD.1	MFP5
		PB.15, PC.10, PE.1, PE.10	MFP7
		PC.3	MFP11
	UART3_nCTS	PD.2	MFP5
		PB.12, PH.9	MFP7
	UART3_nRTS	PD.3	MFP5
		PB.13, PH.8	MFP7

*6.16.4.5 UART4*管脚配置

- 时钟源选择
  - 在寄存器UART4SEL(CLK_CLKSEL3[29:28])配置UART4外设时钟源
  - 在寄存器UART4DIV(CLK_CLKDIV4[11:8])选择UART4外设时钟分频数
  - 在寄存器UART4CKEN(CLK_APBCLK1[20])使能外设时钟
- 寄存器UART4RST(SYS_IPRST1[20])复位UART4控制器
- 管脚配置

组	管脚名	GPIO	MFP

UART4	UART4_RXD	PA.13	MFP3
		PC.6, PH.3	MFP5
		PB.10, PF.6	MFP6
		PA.2, PH.11	MFP7
		PC.4	MFP11
	UART4_TXD	PA.12	MFP3
		PC.7, PH.2	MFP5
		PB.11, PF.7	MFP6
		PA.3, PH.10	MFP7
		PC.5	MFP11
	UART4_nCTS	PC.8	MFP5
		PE.1	MFP9
	UART4_nRTS	PE.13	MFP5
		PE.0	MFP9

*6.16.4.6 UART5*管脚配置

- 时钟源选择
  - 在寄存器UART5SEL(CLK_CLKSEL3[31:30])配置UART5外设时钟源
  - 在寄存器UART5DIV(CLK_CLKDIV4[15:12])选择UART5外设时钟分频数
  - 在寄存器UART5CKEN(CLK_APBCLK1[21])使能外设时钟
- 寄存器UART5RST(SYS_IPRST1[21])复位UART5控制器
- 管脚配置

组	管脚名	GPIO	MFP
UART5	UART5_RXD	PH.1	MFP4
		PF.10	MFP6
		PB.4	MFP7
		PA.4, PE.6	MFP8
	UART5_TXD	PH.0	MFP4
		PF.11	MFP6
		PB.5	MFP7
		PA.5, PE.7	MFP8
	UART5_nCTS	PH.3	MFP4
		PF.8	MFP6
		PB.2	MFP7

	UART5_nRTS	PH.2	MFP4
		PF.9	MFP6
		PB.3	MFP7

*6.16.4.7 UART6*管脚配置

- 时钟源选择
  - 在寄存器UART6SEL(CLK_CLKSEL3[21:20])配置UART6外设时钟源
  - 在寄存器UART6DIV(CLK_CLKDIV4[19:16])选择UART6外设时钟分频数
  - 在寄存器UART6CKEN(CLK_APBCLK1[22])使能外设时钟
- 寄存器UART6RST(SYS_IPRST1[22])复位UART6控制器
- 管脚配置

组	管脚名	GPIO	MFP
UART6	UART6_RXD	PC.11, PH.5	MFP5
		PE.15, PG.14	MFP6
		PA.10, PE.4	MFP8
		PC.6	MFP9
	UART6_TXD	PC.12, PH.4	MFP5
		PE.14, PG.13	MFP6
		PA.11, PE.5	MFP8
		PC.7	MFP9
	UART6_nCTS	PC.9	MFP5
		PE.2	MFP8
	UART6_nRTS	PC.10	MFP5
		PE.3	MFP8

*6.16.4.8 UART7*管脚配置

- 时钟源选择
  - 在寄存器UART7SEL(CLK_CLKSEL3[23:22])配置UART7外设时钟源
  - 在寄存器UART7DIV(CLK_CLKDIV4[23:20])选择UART7外设时钟分频数
  - 在寄存器UART7CKEN(CLK_APBCLK1[23])使能外设时钟
- 寄存器UART7RST(SYS_IPRST1[23])复位UART7控制器
- 管脚配置

组	管脚名	GPIO	MFP
UART7	UART7_RXD	PH.7	MFP4
		PD.8	MFP5

		PG.12	MFP6
		PA.8, PB.8	MFP8
		PE.2	MFP9
	UART7_TXD	PH.6	MFP4
		PD9	MFP5
		PG.11	MFP6
		PA.9, PB.9	MFP8
		PE.3	MFP9
	UART7_nCTS	PH.4	MFP4
		PE.4	MFP9
	UART7_nRTS	PH.5	MFP4
		PE.5	MFP9

表6.16-3是UART接口控制器管脚描述:

管脚	类型	描述
UARTx_TXD	Output	UARTx 发送
UARTx_RXD	Input	UARTx 接收
UARTx_nCTS	Input	UARTx modem 清零发送
UARTx_nRTS	Output	UARTx modem 请求发送

表 6.16-3 UART 接口控制器引脚

6.16.5 功能描述

UART 控制器支持四种功能模式，包括 UART,IrDA,LIN 和 RS-485 模式。用户可以通过对 UART_FUNCSEL设置选择功能。四种功能模式将在下面的章节中描述。

*6.16.5.1 UART*控制器波特率发生器

UART控制器包含一个可编程波特率发生器，其通过分频器对输入时钟源分频而得到收发数据所需的串行时钟。表6.16-4列出了各种条件下的UART波特率计算公式和参数的设置。表6.16-5和表6.16-6列出UART波特率参数和寄存器设置。IrDA功能模式，波特率发生器必须是模式0。寄存器表UART_BAUD一栏中将更详细地介绍寄存器内容。共有三种设定模式：设定UART_BAUD[29:28]=00为模式0，设定UART_BAUD[29:28]=10为模式1，设定UART_BAUD[29:28]=11为模式2。

模式	BAUDM1	BAUDM0	波特率公式
Mode 0	0	0	UART_CLK / [16 * (BRD+2)].
Mode 1	1	0	UART_CLK / [(EDIVM1+1) * (BRD+2)], EDIVM1 要>= 8.
Mode 2	1	1	UART_CLK / (BRD+2) 如果 UART_CLK <= 3*HCLK, BRD 要>= 9.

			如果UART_CLK > 3*HCLK, BRD 要>= 3*N – 1. N 大于等于 UART_CLK /HCLK 的最小整数 例如, 如果3*HCLK < UART_CLK =< 4*HCLK, BRD 要>=11. 如果4*HCLK < UART_CLK =< 5*HCLK, BRD 要>=14. (UART_CLK 选择为LXT, BRD 大于等于 1)

表 6.16-4 UART 控制器波特率公式表

UART 外设时钟 = 12 MHz			
波特率	**模式 0**	**模式 1**	**模式 2**
921600	不支持	不推荐	BRD=11
460800	不推荐	BRD=0, EDIVM1 =12	BRD=24
230400	不推荐	BRD =2, EDIVM1 =12	BRD =50
115200	不推荐	BRD =6, EDIVM1 =12	BRD =102
57600	BRD =11	BRD =14, EDIVM1 =12	BRD =206
38400	BRD =18	BRD =22, EDIVM1 =12	BRD =311
19200	BRD =37	BRD =123, EDIVM1 =4	BRD =623
9600	BRD =76	BRD =123, EDIVM1 =9	BRD =1248
4800	BRD =154	BRD =248, EDIVM1 =9	BRD =2498

表 6.16-5 UART 控制器波特率参数设置示例

UART 外设时钟= 12 MHz			
波特率	UART_BAUD 值		
	模式 0	**模式 0**	**Mode 2**
921600	不支持	不推荐	0x3000_000B
460800	不推荐	0x2C00_0000	0x3000_0018
230400	不推荐	0x2C00_0002	0x3000_0032
115200	不推荐	0x2C00_0006	0x3000_0066
57600	0x0000_000B	0x2C00_000E	0x3000_00CE
38400	0x0000_0012	0x2C00_0016	0x3000_0137
19200	0x0000_0025	0x2400_007B	0x3000_026F
9600	0x0000_004C	0x2900_007B	0x3000_04E0
4800	0x0000_009A	0x2900_00F8	0x3000_09C2

表 6.16-6 UART 控制器波特率寄存器设置示例

6.16.5.2 UART 控制器波特率补偿

UART 控制器支持波特率补偿功能。这提高了每一位的精度。由于每一位都由 BRCOMDEC 位 (UART_BRCOMP[31]) 来定义正负补偿，所以补偿的精度为 UART 模块时钟的一半。如果 BRCOMDEC bit (UART_BRCOMP[31]) 为 0，每一位都正向补偿，被补偿的位会超过一个模块时钟长度。如果 BRCOMDEC bit (UART_BRCOMP[31]) 为 1，每一位都负向补偿，被补偿的位会少于一个模块时钟长度。

BRCOMP[8:0] (UART_BRCOMP[8:0]) 总共有 9 位，可以由用户自由定义对应的位是否被补偿。BRCOMP[7:0] 用来配置 UART_DAT[7:0] 的补偿，而 BRCOMP[8] 用来定义校验位。

示例:

UART 外设时钟为 32.768K，波特率为 9600

UART 外设时钟为 32.768K，波特率为 9600 →即每位 3.413 个外设时钟

如果分频值设为 1 (每位 3 个外设时钟)，则每位的误差为 -0.413 个外设时钟，BRCOMPDEC = 0，所以 BRCOMP (UART_BRCOMP[8:0]) 可以设为 9'b010100101 = 0xa5。

Bit	名称	总误差	BRCOMP 补偿	最终误差
0	START	-0.413	x	-0.413
1	UART_DAT[0]	-0.826(-0.413-0.413)	1	0.174
2	UART_DAT[1]	-0.239(0.174-0.413)	0	-0.239
3	UART_DAT[2]	-0.652(-0.239-0.413)	1	0.348
4	UART_DAT[3]	-0.065(0.348-0.413)	0	-0.065
5	UART_DAT[4]	-0.478(-0.065-0.413)	0	-0.478
6	UART_DAT[5]	-0.891(-0.478-0.413)	1	0.109
7	UART_DAT[6]	-0.304(0.109-0.413)	0	-0.304
8	UART_DAT[7]	-0.717(-0.304-0.413)	1	0.283
9	PARITY	-0.130(0.283-0.413)	0	-0.13

表 6.16-7 波特率补偿示例 1

1. UART 外设时钟为 32.768K，波特率为 4800

UART 外设时钟为 32.768K，波特率为 4800 →每位 6.827 个外设时钟

如果分频值设为 5 (每位 7 个外设时钟)，则每位的误差为 0.173 个外设时钟，BRCOMPDEC = 1

,所以 BRCOMP (UART_BRCOMP[8:0]) 可以设为 9'b010000010 = 0x82。

Bit	名称	总误差	BRCOMP 补偿	最终误差
0	START	0.173	x	0.173
1	UART_DAT[0]	0.346(0.173+0.173)	0	0.346
2	UART_DAT[1]	0.519(0.346+0.173)	1	-0.481
3	UART_DAT[2]	-0.308(-0.481+0.173)	0	-0.308
4	UART_DAT[3]	-0.135(-0.308+0.173)	0	-0.135

5	UART_DAT[4]	-0.038(-0.135+0.173)	0	0.038
6	UART_DAT[5]	0.211(0.038+0.173)	0	0.211
7	UART_DAT[6]	0.384(0.211+0.173)	0	0.384
8	UART_DAT[7]	0.557(0.384+0.173)	1	-0.443
9	PARITY	-0.270(-0.443+0.173)	0	-0.270

表 6.16-8 波特率补偿示例 2

UART控制器自动波特率功能:

自动波特率功能可以自动测量从 UART RX 管脚输入数据的波特率。当自动波特率测量结束，测量结果将会放置在 BRD (UA_BAUD[15:0]) 中。 AUDM1 (UART_BAUD[29]) 和 BAUDM0 (UART_BAUD[28]) 都被自动设置为 1。在自动波特率检测帧中， UART RX 数据从起始位到第一个上升沿的时间由 2 ABRDBITS 位时间设定。

从起始位到第一个上升沿之间的 2 ABRDBITS 位时间是通过设定 ABRDBITS (UART_ALTCTL[20:19]) 计算得出。通过设置 ABRDEN (UART_ALTCTL[18]) 可以使能自动波特率检测功能。在初始阶段，UART RX 保持为 1 。一旦检测到下降沿，即为接收到起始位。自动波特率计数器被重启并开始计数。当检测到第一个上升沿时，自动波特率计数器将停止计数。然后，自动波特率计数器数值除以 ABRDBITS (UART_ALTCTL[20:19]) 的结果将会自动载入到 BRD (UART_BAUD[15:0]) 中，而且 ABRDEN (UART_ALTCTL[18]) 会被清零。如图 7.15 3 所示，一旦自动波特率测量结束，ABRDIF (UART_FIFOSTS[1]) 会被置位。当自动波特率计数器溢出，ABRTOIF (UART_FIFOSTS[2]) 将会被置位。ABRDIF (UART_FIFOSTS[1]) 及 ABRDTOIF (UART_FIFOSTS[2]) 会产生自动波特率标志位 ABRIF (UART_ALTCTL[17]) 。 如果使能了 ABRIEN (UART_INTEN[18]) ， ABRIF (UART_ALTCTL[17]) 将会引发自动波特率中断 ABRINT (UART_INTSTS[31]) 产生。

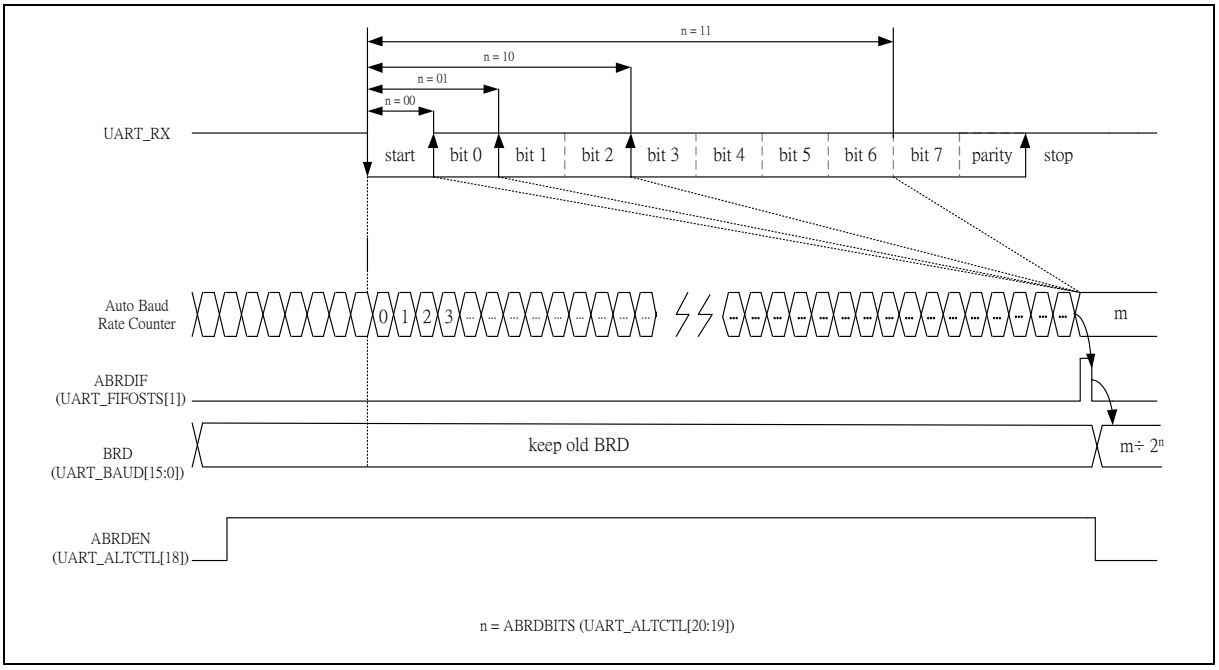


图 6.16-3 自动波特率检测

6.16.5.3 编程顺序示例:

1. 设置 ABRDBITS (UART_ALTCTL[20:19]) 用于决定 UART RX 数据从起始到第一个上升沿时间用的 2 ABRDBITS 位时间
2. 置位 ABRIEN (UART_INTEN[18]) 用于使能自动波特率检测功能中断
3. 置位 ABRDEN (UART_ALTCTL[18]) 用于使能自动波特率检测功能
4. 当 ABRDIF (UART_FIFOSTS[1]) 为 1，表示自动波特率测量结束
5. 执行 UART 发送和接收的动作
6. 当 ABRDTOIF (UART_FIFOSTS[2]) 为 1，表示自动波特率计数器溢出
7. 返回步骤 3。

6.16.5.4 UART 控制器发送延时时间

UART 控制器可以通过设置 DLY (UART_TOUT [15:8]) 来控制传输过程两个数据帧之间即上一个数据帧的停止位和下一个数据帧的起始位之间的间隔。单位是波特。延时操作如图6.16-4。

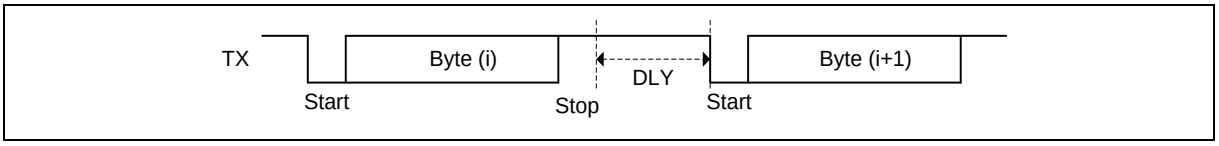


图 6.16-4 发送延时操作

6.16.5.5 UART 控制器 FIFO 控制和状态

UART 控制器内置一个 16 字节的发送 FIFO (TX_FIFO) 和一个 16 字节的接收 FIFO (RX_FIFO), 通讯中,使用这些收发 FIFO 可以减少对 CPU 的中断次数。CPU 在操作过程中任何时间都可以读取 UART 的状态，包括 3 个错误状态 (校验错误，帧错误，间隔错误) 如果接收的数据出现校验错误、帧或间隔错误。FIFO 控制和状态也支持所有的功能模式，包括 UART, IrDA, LIN 和 RS-485 模式。

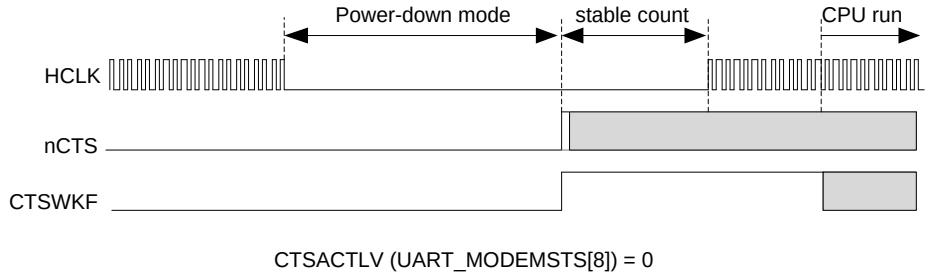
6.16.5.6 UART控制器唤醒功能

UART 控制器支持系统唤醒功能。唤醒功能包括 nCTS唤醒和接收数据唤醒，接收 FIFO 数据达到阈值唤醒，RS-485 地址匹配 (AAD 模式) 唤醒和接收数据 FIFO 阈值超时唤醒。CTSWKF(UART_WKSTS[0]), DATWKF (UART_WKSTS[1]), RFRTWKF (UART_WKSTS[2]), RS485WKF(UART_WKSTS[3]) 或 TOUTWKF (UART_WKSTS[4]) 可以引发产生唤醒中断标志 WKIF(UART_INTSTS[6]) 。如果 WKIEN (UART_INTEN[6]) 使能，唤醒中断标志 WKIF(UART_INTSTS[6]) 引发产生唤醒中断 WKINT (UART_INTSTS[14])。

nCTS管脚唤醒:

当系统处于掉电模式且 WKCTSEN (UART_WKCTL[0]) 置位，nCTS 管脚可以唤醒系统。如果 WKCTSEN (UART_WKCTL[0]) 使能， nCTS 管脚上的跳变会产生 nCTS 唤醒标志 CTSWKF (UART_WKSTS[0]) 。图6.16-5和图6.16-6是唤醒示意图：

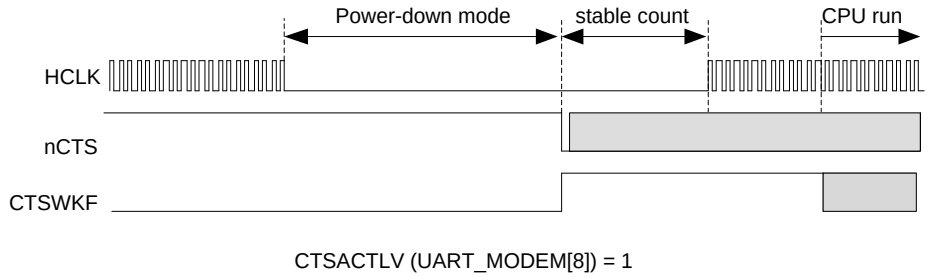
nCTS 唤醒示例 1 (nCTS 发送由低到高)



注: ‘Stable count’ 指 HCLK 恢复稳定的时钟数

图 6.16-5 UART nCTS 唤醒示例 1

nCTS 唤醒示例 2 (nCTS 发送由高到低)



Note: Stable count指 HCLK 恢复稳定的时钟数

图 6.16-6 UART nCTS 唤醒示例 2

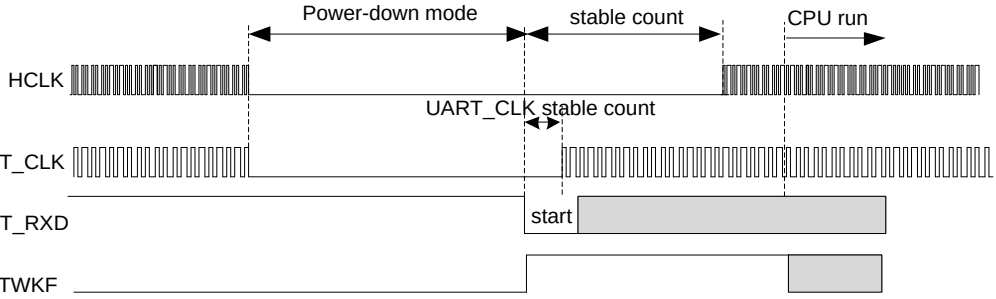
输入数据唤醒

当系统处于掉电模式且 WKDATEN (UART_WKCTL [1]) 置位，输入数据 (UART_RXD) 管脚上的跳变可以唤醒系统。为了能在系统唤醒后正确接收数据，需要设置 STCOMP (UART_DWKCOMP[15:0]) 位域。STCOMP 里的各位代表系统从掉电模式唤醒后，可以接收第 1 个位 (开始位) 需要的时钟数。

输入数据唤醒系统后，输入的数据会被存储到 FIFO 内。如果使能了 WKDATEN (UART_WKCTL[1])，输入数据 (UART_RXD) 管脚上的跳变会触发输入数据唤醒标志 DATWKF (UART_WKSTS[1])。图 6.16-7.是输入数据唤醒的示例。

Note1:UART 控制器的时钟源需要选为 HIRC，以及参考数据表的起始位的补偿时间，以获取关于唤醒时间电特性的详细信息。STCOMP (UART_DWKCOMP[15:0]) = (唤醒稳定时钟数) * (HIRC 频率).

注 2: BRD (UART_BAUD[15:0]) 的值需要大于 STCOMP (UART_DWKCOMP[15:0])。



注 1: ‘Stable count’ 指 HCLK 恢复稳定的时钟数.

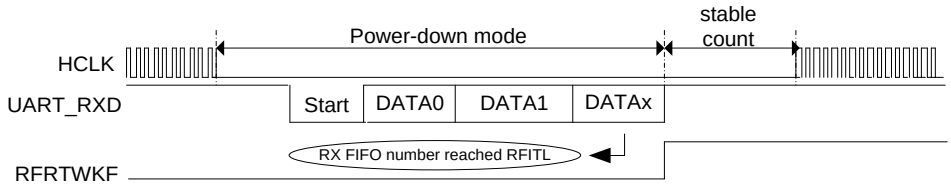
注 2: ‘UART_CLK stable count’ 指 UART 恢复稳定的时钟数

图 6.16-7 UART 数据唤醒

接收数据 FIFO 达到阈值唤醒

设定 WKRFRTEN (UART_WKCTL[2]) 可以使能接收数据 FIFO 达到阈值唤醒功能。在掉电模式，当 RX FIFO 里接收到的数据达到了阈值设定 RFITL (UART_FIFO[7:4])，就会唤醒系统。如果 WKRFRTEN (UART_WKCTL[2]) 使能，RXFIFO 接收到了 RFITL (UART_FIFO[7:4]) 设定个数的值后会触发接收数据 FiFO 达到阈值标志 RFRTWKF (UART_WKSTS[2])。图6.16-8是唤醒示例：

注: 为了接收数据，掉电模式下 UART 时钟源需要设为 LXT。



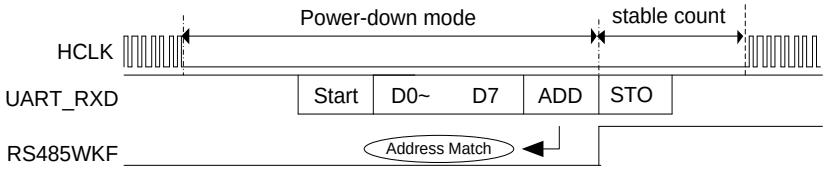
注: ‘Stable count’ 指 HCLK 恢复稳定的时钟数

图 6.16-8UART 接收数据 FIFO 达到阈值唤醒

RS-485 地址匹配 (AAD 模式) 唤醒

设定 WKRFRTEN (UART_WKCTL[2]) 和 WKRS485EN (UART_WKCTL[3]) 使能 RS-485 地址匹配唤醒功能。该功能需要在 ADDRDEN (UART_ALTCTL[15]) 置 1，处于 RS-485 自动地址检测 (AAD) 模式下使用。在掉电模式下，检测到的地址位与 ADDRMV (UART_ALTCTL[31:24]) 相匹配或者 RXFIFO 里接收到的数据达到了阈值设定 RFITL (UART_FIFO[7:4]) 都会唤醒系统。如果使能 WKRS485EN (UART_WKCTL[3])，检测到的地址位与 ADDRMV (UART_ALTCTL[31:24]) 相匹配时会触发 RS485 地址匹配 (AAD 模式) 标志位 RS485WKF (UART_WKSTS[3])。如图6.16-9是唤醒示例。

注:为了接收数据，掉电模式下 UART 时钟源需要设为 LXT。



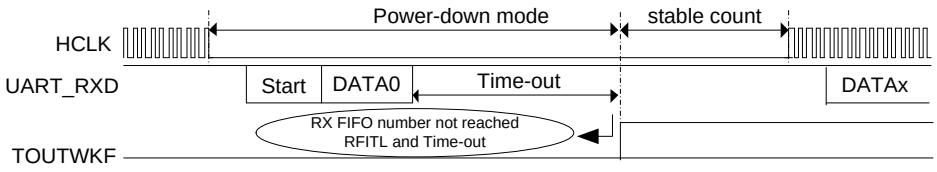
注: ‘Stable count’ 指 HCLK 恢复稳定的时钟数。

图 6.16-9 UART RS-485 AAD 模式地址匹配唤醒

数据接收 FIFO 超时唤醒

设置 WKRFRTEN (UART_WKCTL[2]) 和 WKTOUTEN (UART_WKCTL[4]) 使能数据接收 FIFO 超时唤醒功能。置位 TOCNTEN (UART_INTEN[11]) 使能接收缓存超时计数器。掉电模式下，RXFIFO 里接收到的数据阈值设定 RFITL (UART_FIFO[7:4]) 时，超时计数器计数到 TOIC (UART_TOUT[7:0]) 会唤醒系统。如果 WKTOUTEN (UART_WKCTL[4]) 使能，超时计数器计数到 TOIC (UART_TOUT[7:0]) 的值会触发数据接收 FIFO 超时唤醒标志位 TOUTWKF (UART_WKSTS[4])。如图6.16-10是唤醒例。

注:为了接收数据，掉电模式下 UART 时钟源需要设为 LXT。



注: 'Stable count' 指 HCLK 恢复稳定的时钟数

图 6.16-10 UART 数据接收 FIFO 超时唤醒

6.16.5.7 UART控制器中断和状态

每个 UART 控制器都有 11种类型中断:

- 接收数据可用中断 (RDAINT)
- 发送保持寄存器空中断 (THERINT)
- 发送完成中断 (TXENDIF)
- Line 接收状态中断 (RLSINT)
  - 间隔中断标志 (BIF)
  - 帧错误标志 (FEF)
  - 校验错误标志 (PEF)
  - RS-485 地址位检测标志 (ADDRDETF)
- MODEM 状态中断 (MODEMINT)
  - 检测 nCTS 状态改变标志 (CTSDETF)
- 接收缓存超时中断 (RXTOINT)
- 缓存错误中断 (BUFERRINT)
  - TX 溢出错误中断标志 (TXOVIF)
  - RX 溢出错误中断标志 (RXOVIF)
- LIN 总线中断 (LININT)
  - LIN 间隔检测 (BRKDETF)
  - 位错误检测状态标志 (BITEF)
  - LIN 从机 ID 校验错误标志 (SLVIDPEF)
  - LIN 从机报头错误标志 (SLVHEF)
  - LIN 从机报头检测标志 (SLVHDETF)
- 唤醒中断 (WKINT)
  - nCTS 唤醒标志 (CTSWKF)
  - 输入数据唤醒标志 (DATWKF)
  - 接收数据 FIFO 达到阈值唤醒标志 (RFRTWKF)
  - RS-485 地址匹配 (AAD 模式) 唤醒标志 (RS485WKF)

- 数据接收 FIFO 超时唤醒标志 (TOUTWKF)

- 自动波特率中断 (ABRINT)
  - 自动波特率检测中断标志 (ABRDIF)
    - 自动波特率检测超时中断标志 (ABRDTOIF)
- 单线位错误检测中断(SWBEINT)

表6.16-9所示是各中断源和标志位。当中断使能位使能，中断标志位触发，就会发生中断。用户需要在中断发生后清除中断标志。

中断源	中断指示	中断使能位	中断标志位	标志触发条件	清除标志方法
接收数据可用中断	RDAINT	RDAIEN	RDAIF	N/A	读 UART_DAT
发送保持寄存器空中断	THERINT	THREIEN	THREIF	N/A	写 UART_DAT
发送完成中断	TXENDINT	TXENDIEN	TXENDIF	N/A	写 UART_DAT
Line 接收状态中断	RLSINT	RLSIEN	RLSIF	RLSIF = BIF	**写 1 到**BIF
				RLSIF = FEF	**写 1 到**FEF
				RLSIF = PEF	**写 1 到**PEF
				RLSIF = ADDRDETF	**写 1 到**ADDRDETF
MODEM 状态中断	MODEMINT	MODEMIEN	MODEMIF	MODEMIF= CTSDETF	**写 1 到** CTSDETF
接收缓存超时中断	RXTOINT	RXTOIEN	RXTOIF	N/A	读 UART_DAT
缓存错误中断	BUFERRINT	BUFERRIEN	BUFERRIF	BUFERRIF = TXOVIF	**写 1 到**TXOVIF
				BUFERRIF = RXOVIF	**写 1 到**RXOVIF
LIN 总线中断	LININT	LINIEN	LINIF	LINIF = BRKDETF	**写 1 到**LINIF **和** **写 1 到**BRKDETF
				LINIF = BITEF	**写 1 到**LINIF **和** **写 1 到** BITEF
				LINIF = SLVIDPEF	**写 1 到**LINIF **和** **写 1 到**SLVIDPEF
				LINIF = SLVHEF	**写 1 到**LINIF **和** **写 1 到**SLVHEF
				LINIF = SLVHDETF	**写 1 到**LINIF **和** **写 1 到**SLVHDETF

唤醒中断	WKINT	WKIEN	WKIF	WKIF = CTSWKF	**写 1 到** CTSWKF
				WKIF = DATWKF	**写 1 到**DATWKF
				WKIF = RFRTWKF	**写 1 到** RFRTWKF
				WKIF = RS485WKF	**写 1 到**RS485WKF
				WKIF = TOUTWKF	**写 1 到**TOUTWKF
自动波特率中断	ABRINT	ABRIEN	ABRIF	ABRIF = ABRDIF	**写 1 到**ABRDIF
				ABRIF = ABRDTOIF	**写 1 到**ABRDTOIF
单线位检测错误中断	SWBEINT	SWBEIEN	SWBEIF	N/A	**写1到**SWBEIF

表 6.16-9 UART 控制器中断源和标志位列表

6.16.5.8 UART功能模式

UART 控制器提供了 UART 功能 (用户须设置 FUNCSEL (UART_FUNCSEL [1:0]) 设置为 ’00’ 来使能 UART 功能模式) 。 UART 波特率最高速度是 10 Mbps。

UART 为全双工异步通讯接口。收发各包含一个 16 字节的 FIFO 缓冲区。用户可以设置接收时的 FIFO 触发阈值以及定时溢出检测时间。发送数据帧间 (即从上一帧停止位到下一帧起始位) 时间间隔通过 DLY (UA_TOR [15:8]) 位可设。UART 支持硬件自动流控功能，且 nRTS 流控触发电平可设。如果 RX FIFO 中数据字节数大于或等于 RTSTRGLV (UART_FIFO[19:16])， nRTS 将被释放。

UART 线控功能

UART 控制器通过设置 UART_LINE 寄存器支持串行接口全部特性。可以通过对 UART_LINE 寄存器设置数据位和停止位长度以及校验位。表6.16-10和表6.16-11列出了 UART 数据位和停止位长度的设置以及 UART 校验位的设置。

NSB (UART_LINE[2])	WLS (UART_LINE[1:0])	数据长度 (Bit)	停止位长度 (Bit)
0	00	5	1
0	01	6	1
0	10	7	1
0	11	8	1
1	00	5	1.5
1	01	6	2
1	10	7	2
1	11	8	2

表 6.16-10 UART 线控的数据位和停止位长度设置

校验类型	SPE (UART_LINE[	EPE (UART_LINE[	PSS (UART_LINE[	PBE (UART_LINE[	描述

	5])	4])	7])	3])	
No Parity	x	x	x	0	无奇偶校验位输出
Parity source from UART_DAT	x	x	1	1	软件产生并验证校验位.
Odd Parity	0	0	0	1	奇校验位的计算方法是把数据流中的所有的 1 相加，使得包括校验位在内,1 的总数为奇数个
Even Parity	0	1	0	1	偶校验位的计算方法是把数据流中的所有的 1 相加，使得包括校验位在内,1 的总数为偶数个
Forced Mask Parity	1	0	0	1	奇偶校验位总是逻辑 1 不管数据位中 1 的个数是奇数还是偶数，奇偶校验位永远都是逻辑 1
Forced Space Parity	1	1	0	1	奇偶校验位总是逻辑 0 不管数据位中 1 的个数是奇数还是偶数，奇偶校验位永远都是逻辑 0

表 6.16-11 UART 线控校验位设置

UART 自动流控功能

UART 支持自动流控功能，该功能用到两根信号线 nCTS (清零发送) 和 nRTS (请求发送) 来控制 UART 与外部设备 (如 Modem) 间的数据传输。当自动流控使能后，只有等到 UART 对外部设备发出有效的 nRTS 信号后才允许接收数据，否则不接收。当 RX FIFO 接收到数据的数量等于 RTSTRGLV (UART_FIFO [19:16]) 位的值后， nRTS 信号会被取消。当 UART 检测到外部设备给 nCTS 信号脚有效信号后， UART 开始发送数据，否则 UART 不会发送数据。流控控制的框图参见图6.16-11。

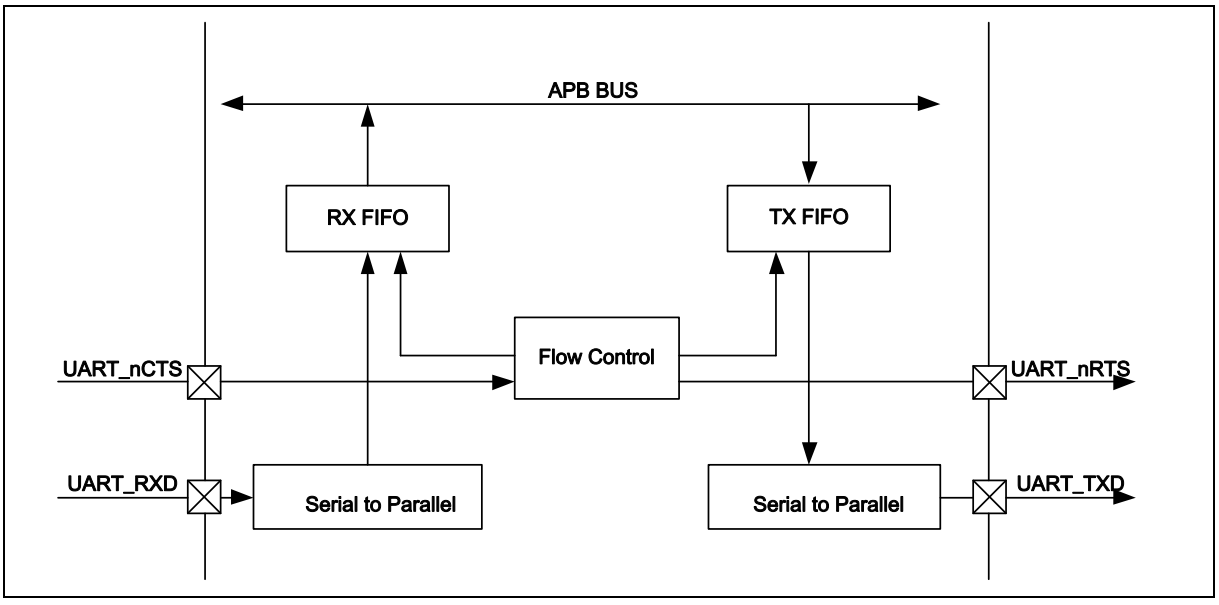


图 6.16-11 自动流控框图

图6.16-12展示了 UART nCTS 自动流控功能。用户必须要先设置 ATOCTSEN (UART_INTEN[13]) 以使能 nCTS 自动流控功能。 CTSACTLV (UART_MODEMSTS [8]) 位可以设置 CTS 脚输入的有效状态

。当 nCTS 脚上任何电平变化将导致 CTSDETF (UART_MODEMSTS[0]) 位被置 1，然后 TX FIFO 将自动将数据发送到 TX 脚,并传送出去。

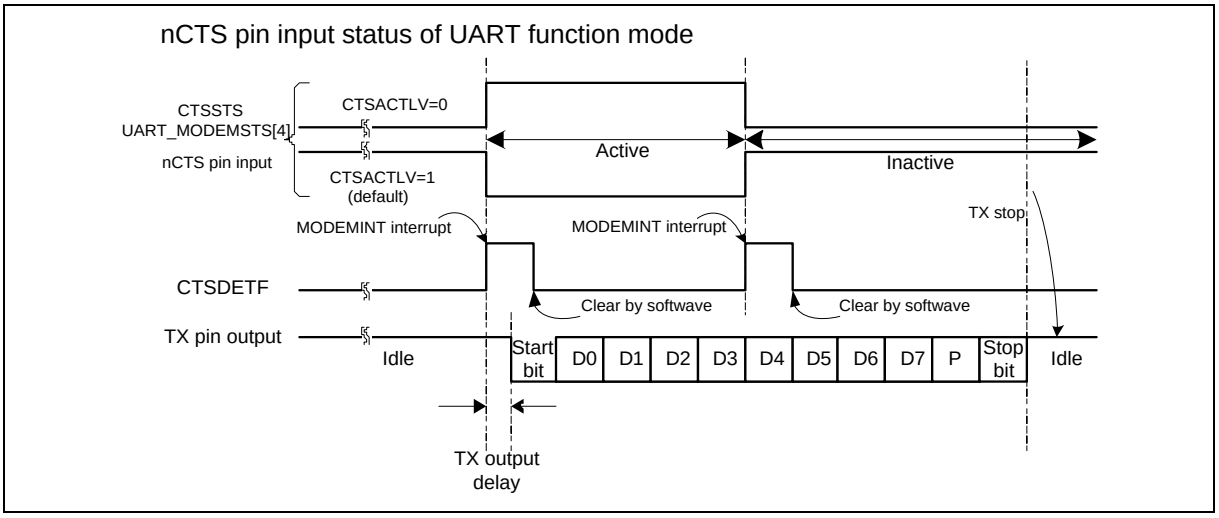


图 6.16-12 UART nCTS 自动流控使能

如图6.16-13所示，UART nRTS 自动流控模式 (ATORTSEN (UART_INTEN[12])=1) 中， nRTS 的触发阈值由 UART FIFO 控制寄存器的 RTSTRGLV (UART_FIFO[19:16]) 控制。

设置 RTSACTLV (UART_MODEM[9]) 可以控制 nRTS 管脚正常输出或反向，信号来自内部的nRTS。用户可以读 RTSSTS (UART_MODEM[13]) 来获取 nRTS 脚输出电压的真实逻辑状态。

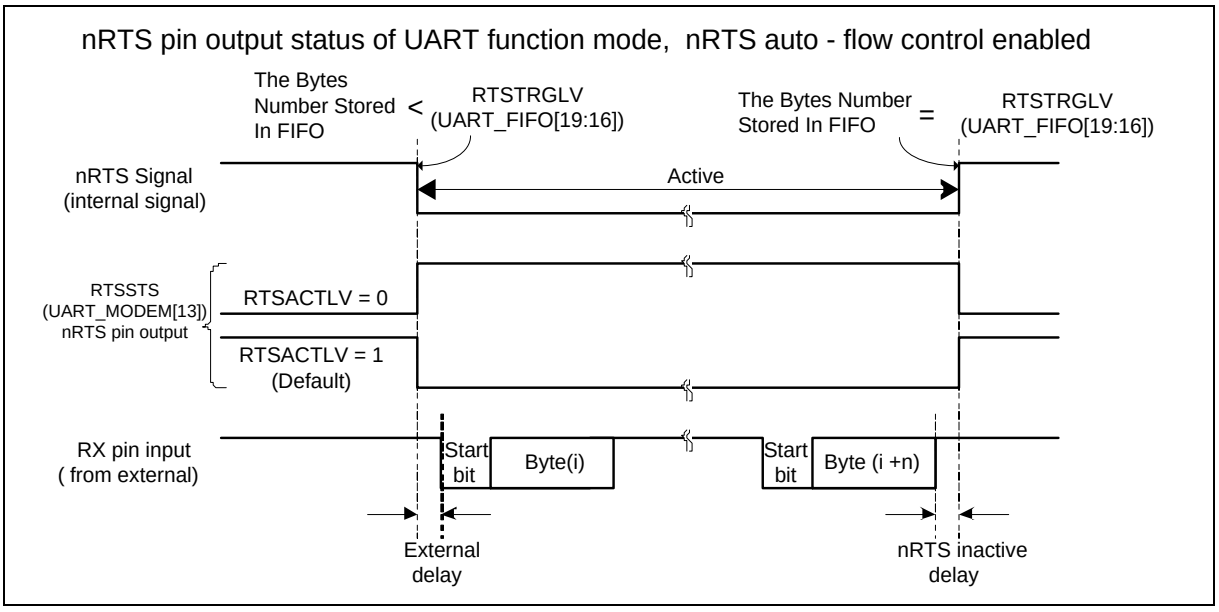


图 6.16-13 UART nRTS 自动流控功能使能

如图 6.16-14 所示，在软件模式下 (ATORTSEN (UART_INTEN[12])=0)，软件改动 RTS (UART_MODEM[1]) 控制位来实现 nRTS 流控。

设置 RTSACTLV (UART_MODEM[9]) 可以控制 nRTS 管脚正常输出或反向，信号来自 RTS(UART_MODEM[1])。用户可以读 RTSSTS (UART_MODEM[13]) 位来获取 nRTS 管脚真实输出电

平状态。

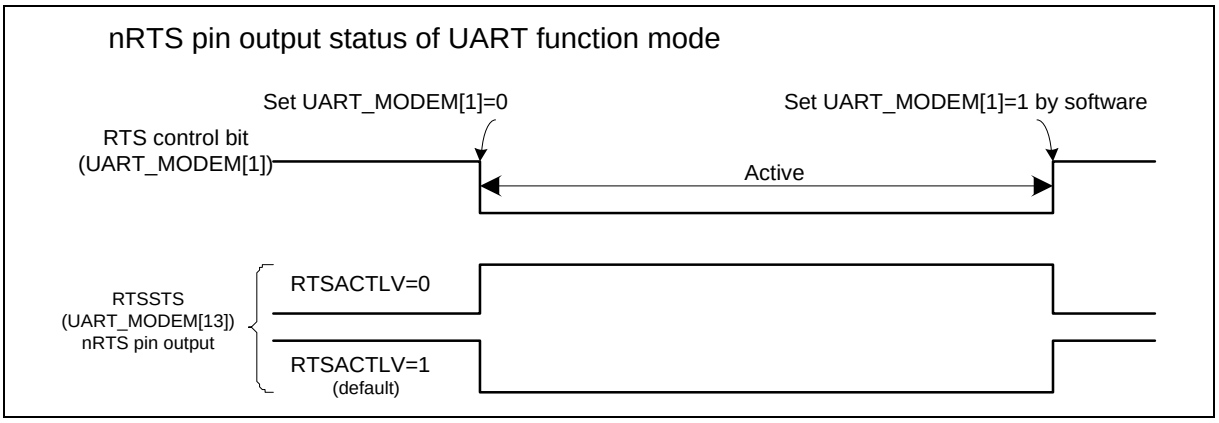


图 6.16-14 UART nRTS 软件控制的自动流控

6.16.5.9 IrDA功能模式

UART 控制器也提供串行 IrDA (SIR, 串行红外) 功能 (用户必须设置 UART_FUNCSEL [1:0] 为'10'来使能 IrDA 功能)。SIR 规范定义了一个短距离红外异步串行传输模式，它包括一个起始位， 8 个数据位，一个停止位。最大速率 115.2kbps。 IrDA SIR 模块包含一个 IrDA SIR 协议编/解码器。 IrDA SIR 协议是半双工工作模式，所以它不能同时收发数据。 IrDA SIR 物理层规定了发送与接收数据的时间上至少 10ms 的时间间隔，该延迟特性需通过软件来完成。

IrDA 模式下，BAUDM1 (UART_BAUD [29]) 需清楚为 0。

波特率 = Clock / (16 * (BRD +2)), 这里 BRD (UART_BAUD[15:0]) 是 UA_BAUD 寄存器中定义的波特率分频器。

: IrDA 主机和从机间波特率不能相差超过±5%。

图6.16-15展示了 IrDA 控制模块框图。

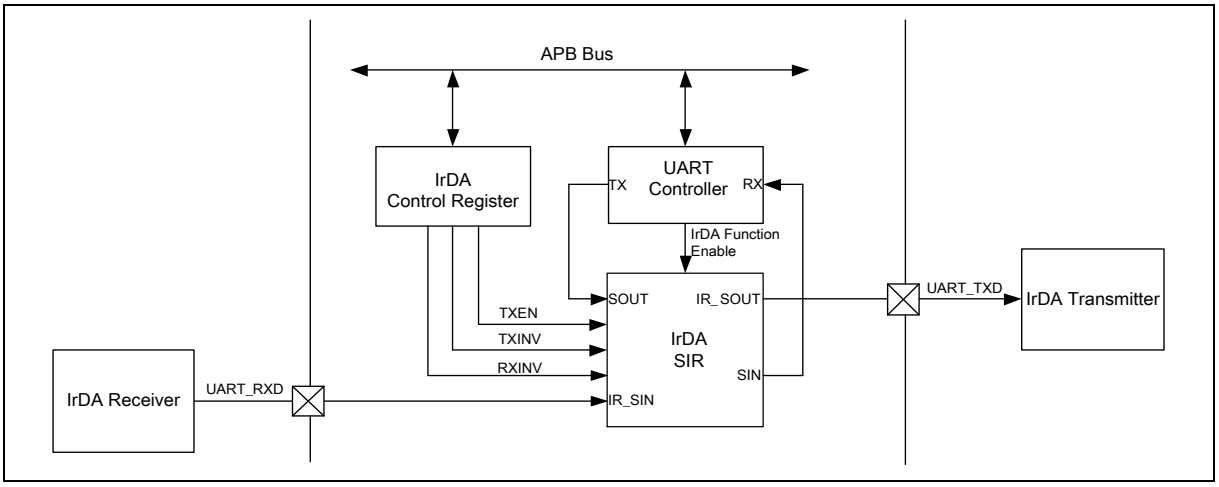


图 6.16-15 IrDA 控制框图

IrDA SIR 发送编码

IrDA SIR 传送编码调制采用 Non-Return-to Zero (NRZ) 编码，数据流编码后从 UART 接口输出。IrDA SIR 物理层指定使用归零反向调制编码 (Return-to-Zero, 反转 (RZI))，用一个红外光脉冲代表逻辑 0，

被调整的脉冲输出到外部输出驱动器和红外线发光二极管。

在正常模式下，传输脉冲的宽度为 3/16 波特率周期。

IrDA SIR 接收解码

IrDA SIR 接收解码 IrDA SIR 接收解码器对输入管脚的 (Return-to-Zero, Inverted (RZI)) 串行位流进行解调，并输出 NRZ 串行位流到 UART 接收数据输入端。

当解码器输入端为低时，表明接收到一个起始位。在空闲状态里，解码器输入端通常为高。正常操作时， RXINV (UART_IRDA[6]) 设置为 1， TXINV (UART_IRDA[5]) 设置为 0。

IrDA SIR 操作

IrDA SIR 编码/解码提供 UART 数据流和半双工串行 SIR 之间的转换。图6.16-16是 IrDA 编码/解码波形图：

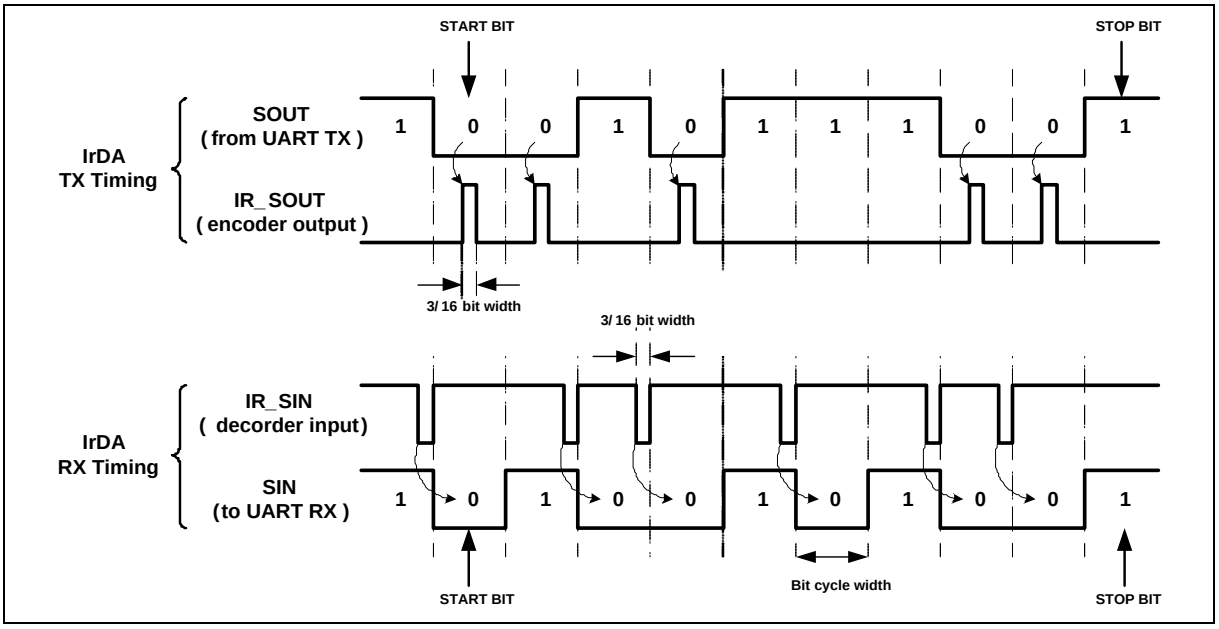


图 6.16-16 IrDA TX/RX 时序图

6.16.5.10 LIN (本地互联网络) 模式

UART 控制器支持 LIN 功能，通过设置 FUNCSEL (UART_FUNCSEL[1:0]) 为 ’01’可以将 UART 设定为 LIN 模式。在主机模式，UART 控制器支持 LIN 间隔/分隔的产生以及间隔/分隔的侦测，在 LIN 从机模式下，支持报头侦测和自动重新同步。

LIN 帧结构

根据 LIN 协议，所有的传输信息被打包为帧。一个帧由一个报头 (主机任务提供) 和一个紧跟其后的应答 (从机任务提供) 组成。报头 (主机任务提供) 由一个间隔域和一个同步域再跟一个帧识别码 (帧 ID) 组成。帧 ID 仅作为定义帧的用途。从机任务负责回应相关的帧 ID。响应由一个数据域和一个校验域组成。图6.16-17是 LIN 帧的结构。

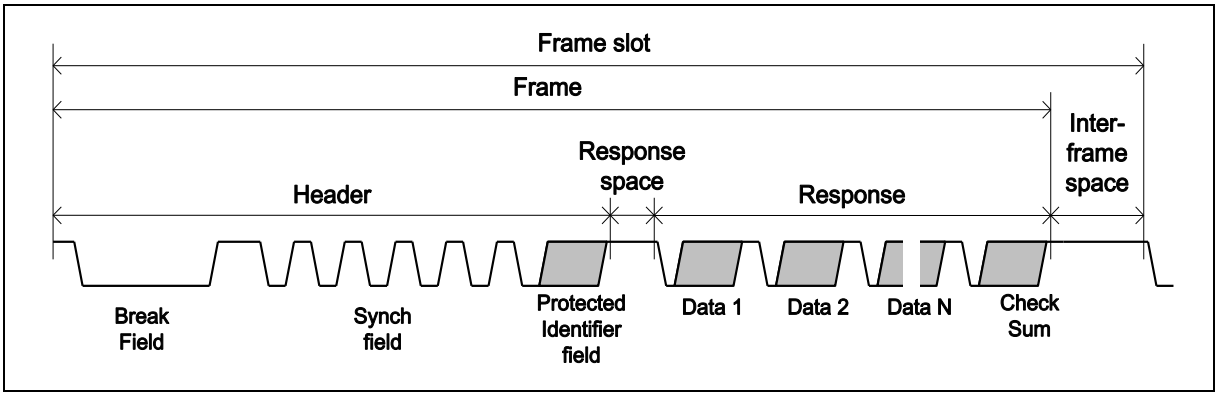


图 6.16-17LIN 帧结构

LIN 字节结构

在 LIN 模式，根据 LIN 的标准，每个字节由值为 0 (显性) 的 START 位开始，接着是 8 位数据位，没有校验位， LSB 优先，由一个值为 1 (隐性) 的 STOP 位结束。字节的结构如图6.16-18.

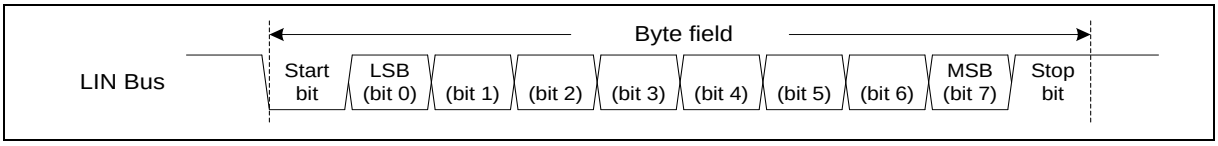


图 6.16-18LIN 字节结构

LIN 主机模式

UART 控制器支持 LIN 主机模式，使能并初始化 LIN 主机模式需要如下步骤：

1. 设置 UART_BAUD 寄存器设定波特率
2. 设置 WLS (UART_LINE[1:0])=‘11’配置数据长度为 8 位，PBE (UART_LINE[3])=0 禁止奇偶校验， NSB (UART_LINE[2])=0 配置 1 位 stop 位
3. 设置 FUNCSEL (UART_FUNCSEL[1:0]) 位为“01”选择 LIN 功能模式。

一个完整的报头由一个间隔域和同步域再跟一个帧标识符 (帧 ID) 组成。 UART控制器可以选择三种报头发送模式，通过设置 HSEL (UART_LINCTL[23:22]) 可以选择“间隔域” 或 “间隔域和同步域” 或 “间隔域, 同步域和帧 ID 域”作为发送的报头。如果选择的报头是“间隔域”，发送一个完整的报头到总线，软件必须依次填同步数据 (0x55) 和帧 ID 数据到 UART_DAT 寄存器。如果选择的报头是“间隔域和同步域”，软件必须填帧 ID 数据到 UART_DAT 寄存器，按顺列来发送一个完整的报头到总线。如果选择的报头是“间隔域, 同步域和帧 ID 域”，硬件会自动控制报头发送顺序，但是软件必须填帧 ID 数据到 PID (UART_LINCTL [31:24])。当选择的报头模式是“间隔域, 同步域和帧 ID 域”时，帧 ID 校验位可以由软件或硬件来计算，这取决于 IDPEN (UART_LINCTL[9]) 位是否置位。

HSEL	间隔域	同步域	ID 域
0	硬件产生	软件产生	软件产生
1	硬件产生	硬件产生	软件产生
2	硬件产生	硬件产生	硬件产生 (先由软件填 ID 到 PID (UART_LINCTL[31:24]))

表 6.16-12 LIN 主机模式报头选择

当 UART 工作于 LIN 数据传输模式时， LIN 总线传输状态可以由硬件或软件监控。通过设置 BITERREN (UART_LINCTL [12]) 为 “1”使能硬件监控。如果在 LIN 发送状态输入管脚 (UART_RX) 状态不同于输出管脚 (UART_TX) 状态，硬件会产生一个中断到 CPU。软件也能通过读回 UART_DAT 寄存器数据监视 LIN 总线传输状态。以下为编程顺序示例。

在主机模式下，没有软件错误监控的步骤:

1. 填保密标识符 ID 到 PID (UART_LINCTL[31:24])
2. 通过设置 HSEL (UART_LINCTL [23:22])=10，选择硬件传输的报头域包括“间隔域 + 同步域+保密标识符 ID 域”
3. 通过设置 SENDH (UART_LINCTL[8]) 为 1 选择硬件传输报头
4. 等待 SENDH (UART_LINCTL[8]) 被硬件清零
5. 等待 TXEMPTYF (UART_FIFOSTS[28]) 被硬件置位

注 1:间隔域的缺省值是 12 个显性位 (间隔域) 和 1 个隐性位 (break/sync 分隔符). 软件可以通过设定 BRKFL (UART_LINCTL [19:16]) 和 BSL (UART_LINCTL[21:20]) 来改变间隔域的长度和 break/sync 分隔符的长度。

注 2:间隔域/同步域分隔符缺省长度是 1 比特时间，字节间隔缺省也是 1 个比特时间。软件可以通过设定 BSL (UART_LINCTL[21:20]) 和 DLY (UART_TOUT[15:8]) 来改变它们的间隔。

注 3: 如果报头包含“间隔域，同步域和帧 ID 域”, 在触发硬件发送报头之前 (设定 SENDH (UART_LINCTL[8])， 软件必须填帧 ID 到 PID (UART_LINCTL[31:24])。根据 IDPEN (UART_LINCTL[9]) 的设定，帧 ID 校验可以由硬件或者软件产生。如果校验由软件产生 IDPEN (UART_LINCTL[9]) 为 0, 软件必须填 8 位数据 (包括 2 个校验位) 到帧 ID 域；如果校验位由硬件产生 (IDPEN (UART_LINCTL[9])= 1), 软件填 ID0~ID5 就好，硬件负责计算 P0 和 P1。

在主机模式下，有软件错误监控的步骤：

1. 设置 UART_LINCTL [23:22]= 00，选择硬件传输的报头域，其内容只包括 “间隔域”
2. 设置 BRKDETEN (UART_LINCTL[10]) 使能间隔域侦测功能
3. 设置 SENDH (UART_LINCTL[8]) 请求硬件传输间隔+ 间隔/同步分隔符
4. 等待 BRKDETF (UART_LINSTS[8]) 标志被硬件置为 “1”
5. 写 0x55 到 UART_DAT 寄存器发送同步域
6. 等待RDAIF (UART_INTSTS[0]) 标志被硬件置为 “1”，然后读回 UART_DAT 寄存器的值
7. 写保护 ID 到 UART_DAT 寄存器，请求帧 ID 域
8. 等待RDAIF (UART_INTSTS[0]) 标志被硬件置为 “1”，然后读回 UART_DAT 寄存器的值。

LIN 暂停和分隔检测

当软件通过设定 BRKDETEN (UART_LINCTL[10]) 使能间隔域侦测功能时，间隔域侦测功能被激活。间隔域侦测电路和 UART 接收电路是完全独立的。

当间隔域侦测功能使能时,电路侦测 UART_RX 引脚的起始信号。如果 UART LIN 控制器侦测到显性连续大于 11 个比特并跟随 1 个隐性比特 (分隔符)，在间隔域结束时 BRKDETF (UART_LINSTS[8]) 标志将被置位，如果 LINIEN (UART_INTEN[8])=1, LININT (UART_INTSTS[15]) 中断将发生，break 侦测和 break 标志如图6.16-19所示。

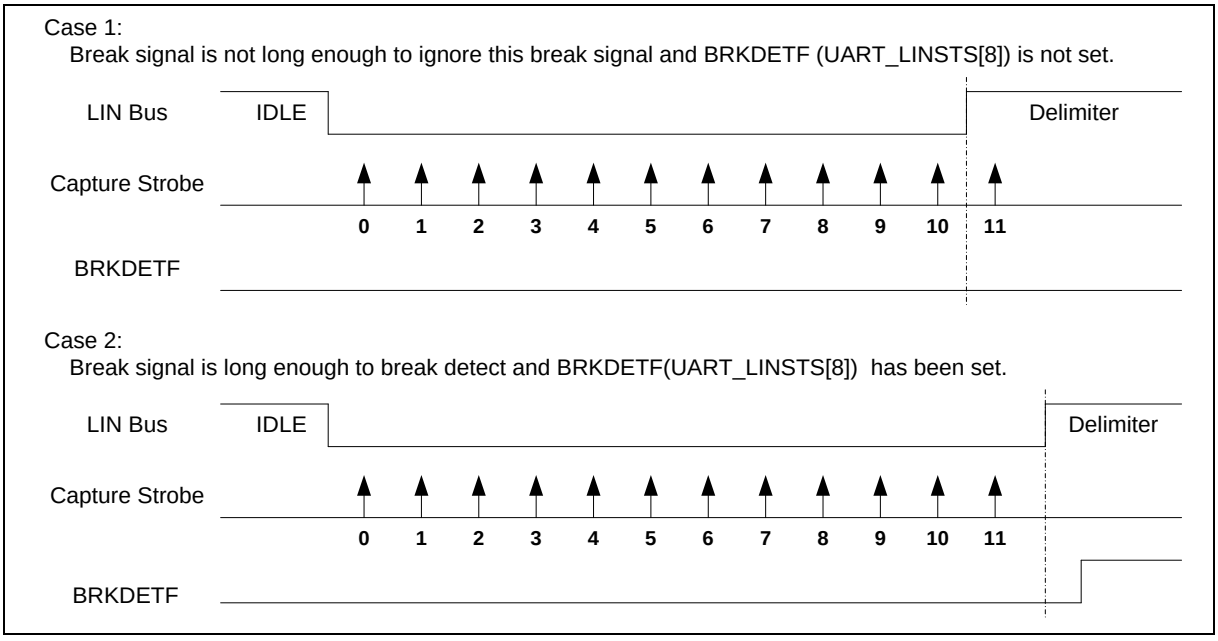


图 6.16-19LIN 模式暂停检测

LIN 帧 ID 和校验格式

LIN 模式下 LIN 的帧 ID 值如下，帧 ID 校验位可以通过软件或硬件产生，产生方式取决于 IDPEN (UART_LINCTL[9]) 位的设置。

如果校验位通过硬件产生，用户需要填写 ID0~ID5 (UART_LINCTL [29:24])，硬件将会计算 P0 (UART_LINCTL[30]) 和 P1 (UART_LINCTL[31])，否则用户必须填写帧 ID 和校验位。

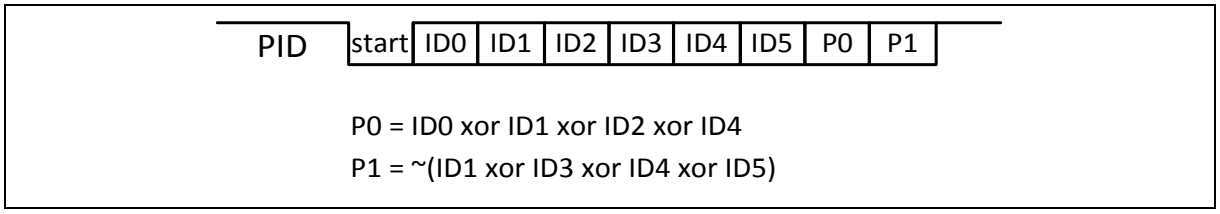


图 6.16-20 LIN 帧 ID 和校验格式

LIN 从机模式

UART 控制器支持 LIN 从模式。使能和初始化 LIN 从模式的必要步骤如下：

1. 设置 UART_BAUD 寄存器设定波特率
2. 设定 WLS (UART_LINE[1:0])= 11 设定数据长度为 8 比特，清 PBE (UART_LINE[3]) 禁止校验，清 NSB (UART_LINE[2]) 设定停止位为 1 比特
3. 设定 FUNCSEL (UART_FUNCSEL[1:0]) 为“01”，选择 LIN 功能
4. 设定 SLVEN (UART_LINCTL[0]) 为 1，使能 LIN 从机模式。

LIN 报头接收

根据 LIN 协议, 从节点必须等待从主节点接收一个有效的报头，然后应用程序可以采取以下动作 (根据主报头帧 ID 的值)。

- 接收响应
- 发送响应
- 忽略响应，等待下一个报头

LIN 从模式下，通过设定 SLVHDEN (UART_LINCTL[1])，使能从机报头检测功能来侦测完整的报头 (接收 “间隔域”, “sync 域” 和 “frame ID 域”)。接收到一个 LIN 报头后, SLVHDETF (UART_LINSTS[0]) 标志将被置位。如果 LINIEN (UART_INTEN[8]) =1, 中断将发生。通过设定 IDPEN (UART_LINCTL[9])，使能帧 ID 校验检查功能。如果收到的帧 ID 校验位不正确时 (break 和 sync 域正确), SLVIDPEF (UART_LINSTS[2]) 标志将被置 1。如果 LINIEN (UART_INTEN[8]) =1, 中断将发生， SLVHDETF (UART_LINSTS[0]) 会被置 1。通过设定 MUTE (UART_LINCTL[4])=1，用户也可以将 LIN 设为 mute 模式。该模式只允许检测报头 (break + sync + frame ID) ，不允许接收任何其它字符。为了避免比特率偏差, 控制器支持自动重同步功能，避免时钟误差错误，通过设定 SLVAREN (UART_LINCTL[2]) 使能该特性。

LIN 发送响应

LIN 从机模式可以发送响应和接收响应。当从机节点是响应的发送者时，通过将数据填到 UART_DAT 寄存器发送响应；如果从机节点是响应的接收者时，从接节点从 LIN 总线接收数据。

LIN 报头超时错误

LIN 从机控制器包含一个报头超时计数器。如果整个报头没有在 57 比特的最大时间限制内收到，报头错误标志 SLVHEF (UART_LINSTS [1]) 将被置位。超时计数器在每个 break 侦测沿使能，在下列条件下将停止：

- LIN 帧 ID 域已经收到了
- 报头错误标志被置起
- 写 1 到 SLVSYNCF (UART_LINSTS[3])，重新侦测新的报头

Mute 模式及 LIN 从 Mute 模式退出的条件

Mute 模式下LIN 从机节点需满足一定的条件时才能接收其它数据。此模式只允许侦测报头，禁止接收任何其它数据。通过设定 MUTE (UART_LINCTL[4] 使能 Mute 模式，设置 HSEL (UART_LINCTL[23:22]) 来退出 Mute 模式。

注:建议校验字节发送之后，将 LIN 从机节点设定为 Mute 模式

LIN 从机控制器从 Mute 模式退出描述如下：如果 HSEL (UART_LINCTL[23:22]) 设为 “间隔域”，当 LIN 从机控制器检测到一个有效的 LIN break + 分隔符时, 控制器将使能接收器 (退出 Mute 模式) ，随后的数据 (sync 数据, 帧 ID , 响应数据) 将被收到 RX-FIFO 中。

如果 HSEL (UART_LINCTL[23:22]) 设为 “间隔域和 sync 域”, 当 LIN 从机控制器检测到一个有效的 LIN break + 分隔符，后面跟一个有效的同步域并且没有帧错误时，控制器将使能接收器 (退出 Mute 模式) ，随后的数据 (帧 ID, 响应数据) 将被收到 RX-FIFO 中。如果 HSEL (UART_LINCTL[23:22]) 设为“间隔域, sync 域和 ID 域”, 当 LIN 从机控制器检测到一个有效的 LIN break + 分隔符和有效的同步域，并且没有帧错误后面跟一个有效的帧 ID 域并且没有帧错误，并且收到的帧 ID 和 PID (UART_LINCTL[31:24]) 中的值匹配时，控制器将使能接收器 (退出 Mute 模式) 随后的数据 (响应数据) 将被收到 RX-FIFO 中。

非自动重新同步从机模式(NAR)

用户可以关闭自动重新同步功能来固定通信波特率。当工作在非自动重新同步模式时，软件需要一些初始化过程，初始化过程如下所示：

1. 设定 UART_BAUD 寄存器设定波特率

2. 设定 FUNCSEL (UART_FUNCSEL[1:0]) 为“ 01”选择 LIN 功能
3. 设定 SLVAREN (UART_LINCTL[2])= 0 关闭自动重新同步功能
4. 设定 SLVEN (UART_LINCTL[0]) 为 1 使能 LIN 从机模式。

自动重新同步从机模式(AR)

自动重新同步模式下，每次收到同步域时，控制器将调整比特率发生器。软件需要一些初始化过程，初始化过程如下所示：

1. 设定 UART_BAUD 寄存器设定波特率
2. 设定 FUNCSEL (UART_FUNCSEL[1:0]) 为“ 01”选择 LIN 功能
3. 设定 SLVAREN (UART_LINCTL[2]) 为 ‘1’使能自动重新同步功能
4. 设定 SLVEN (UART_LINCTL[0]) 为‘1’，使能 LIN 从机模式。

当自动重新同步功能使能后, 每个 LIN 间隔域后面, 用 LIN 的工作时钟持续采样 5 个下降沿的时间，测量的结果储存在内部一个 13-位寄存器中，在第 5 个下降沿结束， UART_BAUD 寄存器的值将被自动更新。如果在 5 个下降沿之前，测量计数器 (13-位) 溢出, 报头错误标志 SLVHEF (UART_LINSTS [1]) 将被置位。

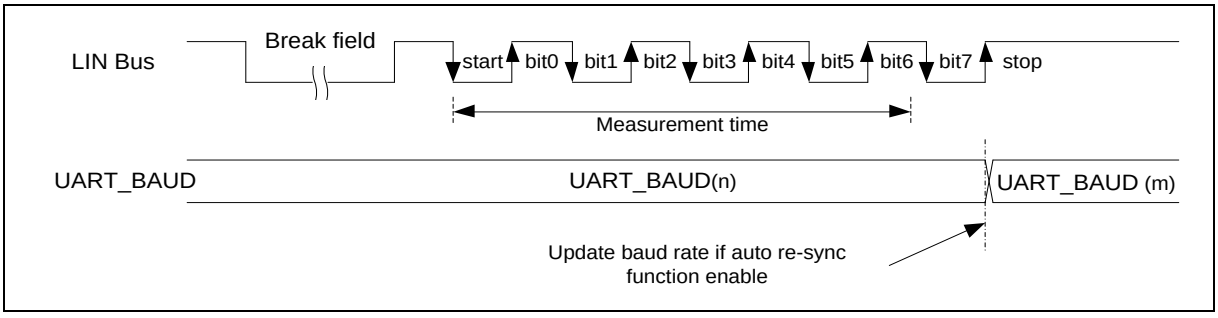


图 6.16-21 LIN 同步域测量

自动重新同步模式 (AR) 下，软件必须通过设定 UART_BAUD 寄存器设定希望的波特率，硬件将保存到内部的 TEMP_REG 寄存器, 每次 LIN 间隔域之后, 5 个下降沿被采样，测量结果保存到内部 13-位寄存器 (BAUD_LIN) ，这个结果将自动更新到 UART_BAUD 寄存器。

为了保证传输波特率,每个新的间隔域收到之前，波特率发生器必须重新加载初始值。初始值在初始化的时候由应用程序设定 (TEMP_REG)。用户可以设定 SLVDUEN (UART_LINCTL [3]) 来使能自动加载初始波特率功能。如果 SLVDUEN (UART_LINCTL [3])= 1, 当前帧结束，收到下一个字符之前，硬件将自动加载初始值到 UART_BAUD 寄存器， UA_BAUD 被更新之后, SLVDUEN (UART_LINCTL [3]) 将被自动清 0。 LIN 波特率的更新方法如图6.16-22所示。

注 1:收到检验字符之前，推荐设定 SLVDUEN 位为 1。

注 2:侦测到报头错误时, 用户需要对 SLVSYNCF (UART_LINSTS[3]) 写 1 来重新检测报头. 当写 1 到该位时, 硬件将重载初始波特率 (TEMP_REG) 并重新搜索新的报头。

注 3:自动重新同步模式下，波特率必须设定为模式 2 (BAUDM1 UART_BAUD [29]) 和 BAUDM0 (UART_BAUD[28]) 必须为 1)。

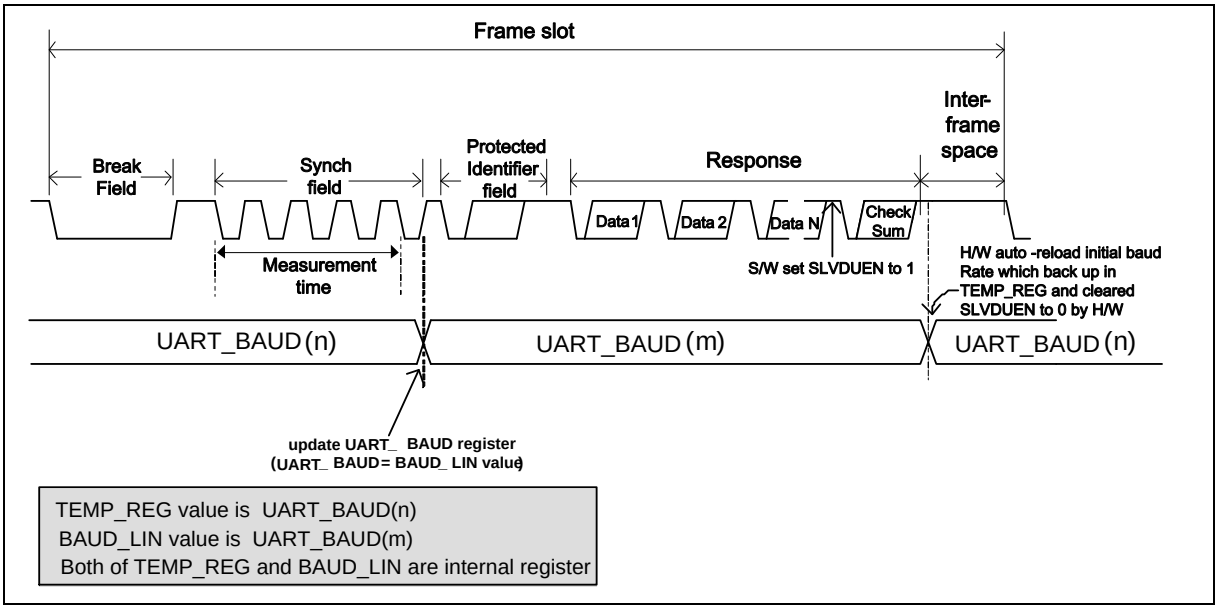


图 6.16-22AR 模式下，SLVEUEN=1 时，UART_BAUD 更新顺序

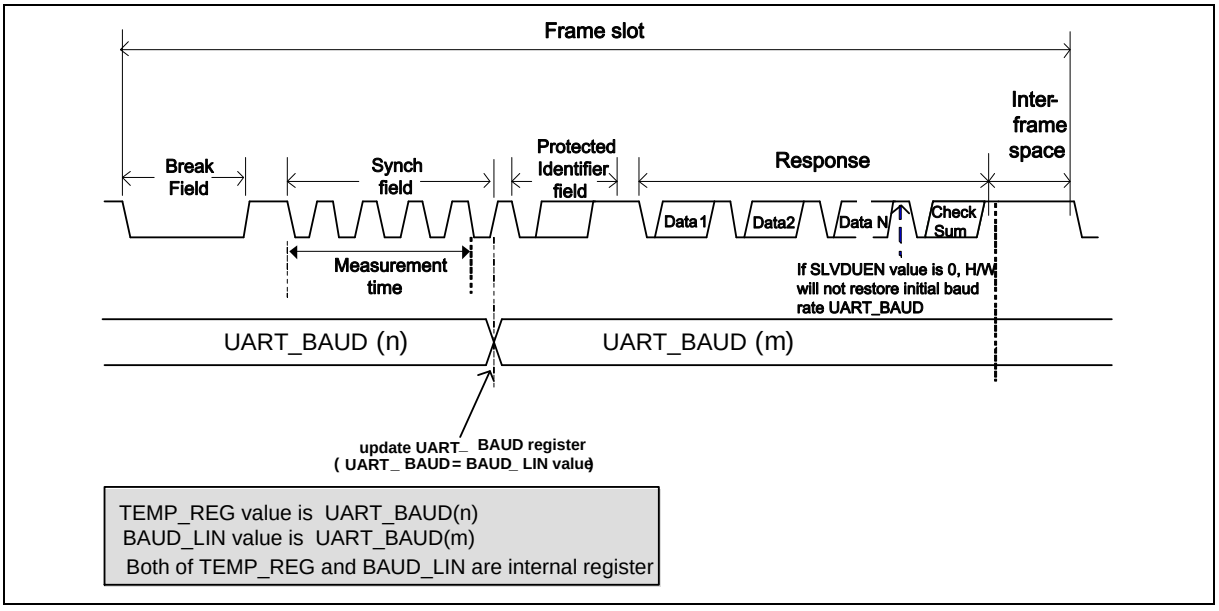


图 6.16-23AR 模式下，SLVEUEN=0 时，UART_BAUD 更新顺序 0

同步域误差错误

自动重新同步模式下，控制器将检测同步域的误差错误。误差错误检测比较当前波特率和接收到的同步域的波特率。两个检测被同步执行。

检查 1:根据同步域的第一个下降沿和最后一个下降沿的测量值：

- 如果误差大于 14.84%, 报头错误标志 SLVHEF (UART_LINSTS[1]) 将被置位
- 如果误差小于 14.06%, 报头错误标志 SLVHEF (UART_LINSTS[1]) 不会被置位
- 如果误差在 14.84% 和 14.06%之间, 报头错误标志 SLVHEF (UART_LINSTS[1]) 可能被置

位也可能没有被置位 (取决于数据失相)。

检查 2: 根据同步域的每一个下降沿的测量值：

- 如果误差大于 18.75%,报头错误标志 SLVHEF (UART_LINSTS[1]) 将被置位
- 如果误差小于 15.62%,报头错误标志 SLVHEF (UART_LINSTS[1]) 不会被置位
- 如果误差在 18.75% 和 15.62%之间, 报头错误标志 SLVHEF (UART_LINSTS[1])) 可能被置位也可能没有被置位。

注:误差检测基于当前波特率时钟。因而，为了保证误差检测的正确性，通过设定 SLVDUEN (UART_LINCTL[3])，新的间隔域收到之前，波特率必须重新加载为初始值 (每个校验和收到之前，推荐设定 SLVDUEN (UART_LINCTL[3]) 为 1)。

LIN 报头错误检测

LIN 从机模式下, 当用户通过设定 SLVHDEN (UART_LINCTL[1]) 使能报头检测功能时, 硬件将处理报头检测流程。如果报头有错误， LIN 报头错误标志 SLVHEF (UART_LINSTS[1]) 将被置位，如果 LINIEN (UART_INTEN[8])=1，中断将发生。报头错误被检测到时，用户必须写 1 到 SLVSYNCF (UART_LINSTS[3]) 来复位检测电路以重新检测新的报头。

如果下列一个条件发生， LIN 报头错误标志 SLVHEF (UART_LINSTS[1]) 将被置位：

- 间隔域分隔符太短 (小于 0.5 比特时间)
- 同步域或者帧 ID 域帧错误
- 同步域数据不是 0x55 (非自动重新同步模式).
- 同步域误差错误 (自动重新同步模式)
- 同步域测量超时 (自动重新同步模式)。

6.16.5.11 RS-485功能模式

UART 控制器另一个可选择的功能是 RS-485 功能 (用户必须设置 UART_FUNCSEL [1:0] = '11'来使能 RS-485 功能)，方向控制则由异步串口的 nRTS 脚来控制。RS-485 收发器的驱动控制是通过 nRTS 控制信号来驱动的。 RS-485 模式下的 RX 和 TX 大多数特性与 UART 相同。

RS-485 模式，控制器可以配置成 RS-485 可寻址的从机模式， RS-485 主机发送可通过设置检验位 (第 9 位) 为 1 标识地址特性。对于数据字符，检验位设置为 0。设置寄存器 UART_LINE 控制第 9 位 (PBE, EPE 和 SPE 置位时，第 9 位发送 0； PBE 和 SPE 置位， EPE 清零时，第 9 位发送 1)。

该控制器支持三种操作模式： RS-485 普通多点操作模式 (NMM)， RS-485 自动地址识别模式 (AAD) 和 RS-485 自动方向控制模式 (AUD)，可通过 UART_ALTCTL 寄存器的设置选择其中一种工作模式，通过设置 DLY (UART_TOUT [15:8]) 可以设置上一个停止位与下一个开始位之间的延迟时间。

RS-485 普通多点操作模式(NMM)

RS-485普通多点操作模式(RS485NMM(UART_ALTCTL[8])=1)，首先，在检测到地址字节之前，软件决定数据是否保存到RX FIFO。如果想忽略接收任何数据在地址字节检测到之前，打开RXOFF(UART_FIFO[8])再使能RS485NMM(UART_ALTCTL[8])，这样将忽略任何数据接收除非地址字节被检测到(bit9=1)并且地址字节被保存到RX FIFO。如果软件想接收任何数据在地址字节检测到之前，关闭RXOFF(UART_FIFO[8])再使能RS485NMM(UART_ALTCTL[8])，将接收任何数据。

如果检测到地址字节(第9位为1)，会产生一个中断到CPU，软件可以通过设置RXOFF(UART_FIFO[8])，决定是否使能或禁用接收器接收数据。如果设置RXOFF(UART_FIFO[8])禁用接收器，会忽略所有接

收到的字节数据，直到检测到下一个地址字节。当检测到地址字节后，控制器将清除RXOFF(UART_FIFO[8])且地址字节数据将存储到RX-FIFO。

RS-485 自动地址识别工作模式(AAD)

RS-485 自动地址识别模式 (RS485AAD (UART_ALTCTL[9]) = 1)，接收器在检测到地址字节 (第 9 位为 1) 并且地址字节数据与 ADDRMV (UART_ALTCTL[31:24]) 的值相匹配之前将忽略所有数据。地址字节数据将存储在 RX-FIFO。所有接收字节数据将被接收并存储于 RX-FIFO 直到一个地址字节与 ADDRMV (UART_ALTCTL[31:24]) 的值不匹配。

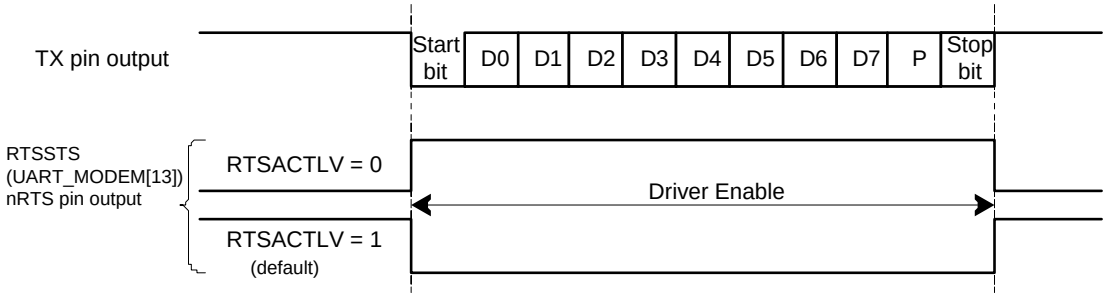
RS-485 自动方向控制功能(AUD)

RS-485 控制器的另一个功能是 RS-485 自动方向控制功能 (RS485AUD (UART_ALTCTL[10] = 1)。RS-485 通过 nRTS 驱动控制异步串口的控制信号，使能 RS-485 驱动器。 nRTS 连接到 RS-485 驱动器，设置 nRTS 线为高 (逻辑 1) 使能 RS-485 驱动器。设置 nRTS 为低 (逻辑 0)，使驱动器进入 tri-state 状态。用户通过设置寄存器 UART_MODEM 中的 RTSACTLV 改变 nRTS 驱动电平。

图6.16-24展示了 AUD 模式下 RS-485 nRTS 驱动电平。nRTS 脚在 TX 数据发送阶段自动驱动收发器。

设置 RTSACTLV (UART_MODEM[9]) 可以控制 nRTS 脚的输出电平。用户可以通过读 RTSSTS (UART_MODEM[13]) 来获取 nRTS 脚上实际的输出逻辑电平。

nRTS pin output status of RS-485 function mode (RS-485 AUD mode enabled)

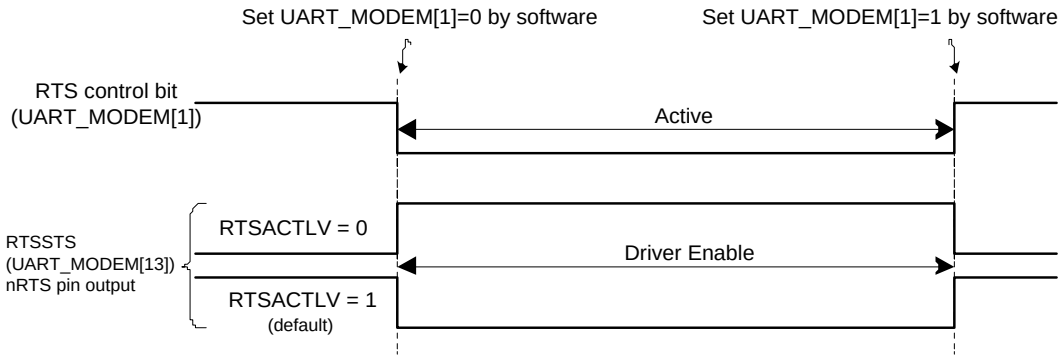


Note: RS485AUD(UART_ALTCTL[10]) = 1, the nRTS pin output by hardware control only.

图 6.16-24 RS-485 nRTS 自动方向控制模式下 nRTS 管脚驱动电平

图6.16-25展示了通过软件控制 (RS485AUD (UART_ALTCTL[10])=0) 模式下， RS-485 nRTS 脚的驱动电平。nRTS 驱动电平通过设置 RTS (UART_MODEM[1]) 来控制。

设置 RTSACTLV (UART_MODEM[9]) 可以控制 nRTS 脚的输出与 RTS (UART_MODEM[1]) 设置值是否相反。用户可以读 RTSSTS (UART_MODEM[13]) 来获取 nRTS 脚上实际的逻辑电平。RS-485 的帧结构参见图6.16-26。

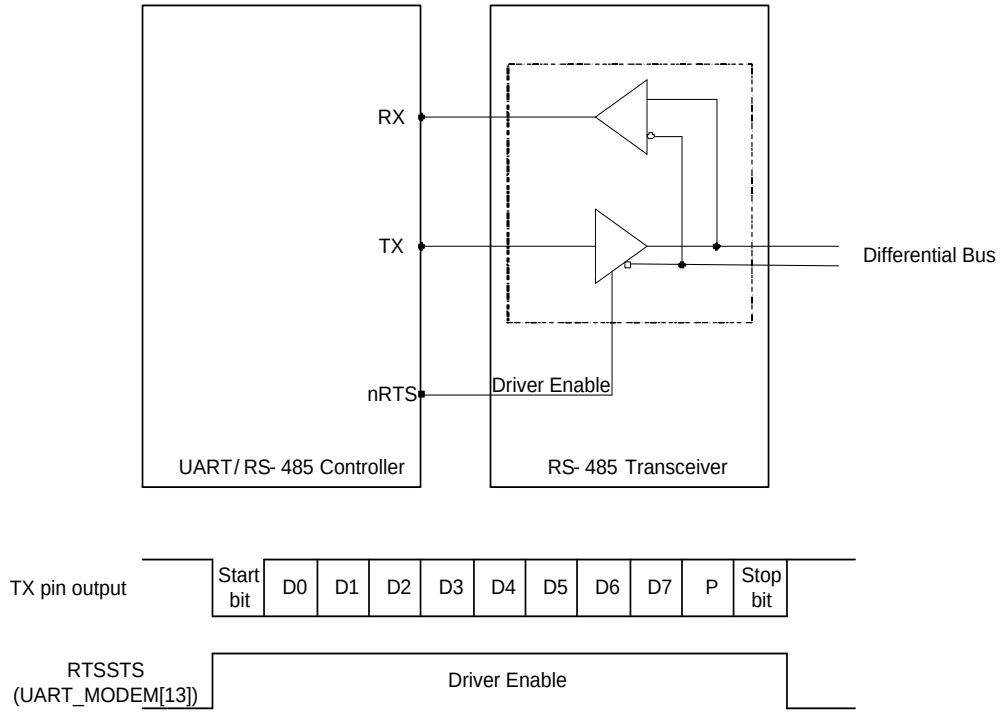


Note: RS485AUD (UART_ALTCTL[10]) = 0, support nRTS control bit by software control only.

图 6.16-25 RS-485 软件控制时的驱动电平

编程流程示例：

1. 设置UART_FUNCSEL中的FUN_SEL选择RS-485功能
2. 设置RXOFF (UART_FIFO[8])，使能或禁用RS-485接收器
3. 设置RS485NMM (UART_ALTCTL[8])或RS485AAD(UART_ALTCTL[9]) 模式
4. 如果选择 RS485AAD(UART_ALTCTL[9])模式，ADDRMV(UART_ALTCTL[31:24])需设置成自动地址匹配值
5. 设置 RS485_AUD(UA_ALT_CSR[10])来决定是否为自动方向控制)。



Note:RS485AUD(UART_ALTCTL[10]) must be set to 1, and RTSACTLV(UART_MODEM[9]) must be set to 0.

图 6.16-26RS-485 帧结构

6.16.5.12 UART 单线半双工

UART控制器有单线半双工功能在UART模式下。设置UART_FUNCSEL[2:0]为'3'b100'来使能UART单线功能。在RX状态，单线总线处于空闲(RXIDLE(UART_FIFOSTS[29])=1)。在写数据到发送缓存(UART_DAT[7:0])前，总线将检测是否空闲(RXIDLE(UART_FIFOSTS[29])。通过写数据到发送缓存，总线立即转换为TX状态。在传输后，总线由TX状态转换为RX状态。

UART单线TX模式下不允许接收数据。如果在TX模式下nRTS有效，nRTS将使对接的UART设备发送数据并引起总线冲突。为减少总线冲突，UART控制器支持流控制功能和位错误检测但不支持自动流控制功能。在单线TX状态，nRTS自动未激活。在TX状态，UART控制器将监测总线状态。如果总线状态不是TX状态，SWBEIF(UART_INTSTS[16])将被设置。

6.16.5.13 PDMA传输功能

UART 控制器支持 PDMA 传输功能.

设定 PDMA相关参数并设定UART_DAT为PDMA 目标地址后，当TXPDMAEN (UART_INTEN[14]) 为1，UART 控制器会请求 PDMA 控制器自动开始PDMA传输。

设定PDMA相关参数并设定UART_DAT为PDMA为源地址后，当RXPDMAEN (UART_INTEN[15]) 为1，UART控制器会请求PDMA控制器在RX FIFO缓存有数据时，自动开始PDMA传输。

注:如果配置了STOPn(PDMA_STOP[n])寄存器停止UART RXPDMA任务，且UART还未完成接收，UART控制器会完成传输并存储当前的接收数据到接收缓存。读寄存器RXEMPTY(UART_FIFOSTS[14]) 可以知道当前接收缓存有无有效数据。

6.16.6 寄存器映射

R: 只读，W: 只写，R/W: 读/写

寄存器	偏移量	R/W	描述	复位值
UART基地址: **UARTx_BA = 0x4007_0000 + (0x1000 * x)** **x=0,1,2,3,4,5,6,7**				
UART_DAT **x=0,1,2,3,4,5,6,7**	UARTx_BA+0x00	R/W	UART 接收/发送缓存寄存器	未定义
UART_INTEN **x=0,1,2,3,4,5,6,7**	UARTx_BA+0x04	R/W	UART 中断使能寄存器	0x0000_0000
UART_FIFO **x=0,1,2,3,4,5,6,7**	UARTx_BA+0x08	R/W	UART FIFO 控制寄存器	0x0000_0101
UART_LINE **x=0,1,2,3,4,5,6,7**	UARTx_BA+0x0C	R/W	UART 线控寄存器	0x0000_0000
UART_MODEM **x=0,1,2,3,4,5,6,7**	UARTx_BA+0x10	R/W	UART Modem 控制寄存器	0x0000_0200
UART_MODEMSTS **x=0,1,2,3,4,5,6,7**	UARTx_BA+0x14	R/W	UART Modem 状态寄存器	0x0000_0110
UART_FIFOSTS **x=0,1,2,3,4,5,6,7**	UARTx_BA+0x18	R/W	UART FIFO 状态寄存器	0xB040_4000
UART_INTSTS **x=0,1,2,3,4,5,6,7**	UARTx_BA+0x1C	R/W	UART 中断状态寄存器	0x0040_0002
UART_TOUT **x=0,1,2,3,4,5,6,7**	UARTx_BA+0x20	R/W	UART 超时寄存器	0x0000_0000
UART_BAUD **x=0,1,2,3,4,5,6,7**	UARTx_BA+0x24	R/W	UART 波特率分频器	0x0F00_0000
UART_IRDA **x=0,1,2,3,4,5,6,7**	UARTx_BA+0x28	R/W	UART IrDA 控制寄存器	0x0000_0040
UART_ALTCTL **x=0,1,2,3,4,5,6,7**	UARTx_BA+0x2C	R/W	UART 选择控制/状态寄存器	0x0000_000C

UART_FUNCSEL x=0,1,2,3,4,5,6,7	UARTx_BA+0x30	R/W	UART 功能选择寄存器	0x0000_0000
UART_LINCTL x=0,1	UARTx_BA+0x34	R/W	UART LIN 控制寄存器	0x000C_0000
UART_LINSTS x=0,1	UARTx_BA+0x38	R/W	UART LIN 状态寄存器	0x0000_0000
UART_BRCOMP x=0,1,2,3,4,5,6,7	UARTx_BA+0x3C	R/W	UART 波特率补偿寄存器	0x0000_0000
UART_WKCTL x=0,1,2,3,4,5,6,7	UARTx_BA+0x40	R/W	UART 唤醒控制寄存器	0x0000_0000
UART_WKSTS x=0,1,2,3,4,5,6,7	UARTx_BA+0x44	R/W	UART 唤醒状态寄存器	0x0000_0000
UART_DWKCOMP x=0,1,2,3,4,5,6,7	UARTx_BA+0x48	R/W	UART 输入数据唤醒补偿寄存器	0x0000_0000

6.16.7 寄存器描述

UART接收/发送缓存寄存器(UART_DAT)

寄存器	偏移量	R/W	描述	复位值
UART_DAT x=0,1,2,3,4,5,6,7	UARTx_BA+0x00	R/W	UART接收/发送缓存寄存器	Undefined

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							PARITY
7	6	5	4	3	2	1	0
DAT							

位	描述	
[31:9]	**Reserved**	保留
[8]	**PARITY**	**校验位接收/发送缓存** 写操作： 写该位，校验位会被存入发送 FIFO。 如果 PBE (UART_LINE[3]) 和 PSS (UART_LINE[7]) 置位，UART 控制器会通过 UART_TXD 发送 DAT (UART_DAT[7:0]) 及该位 如果 PBE (UART_LINE[3]) 和 PSS (UART_LINE[7]) 使能，可以读这位获得校验位 **注:** 仅 PBE (UART_LINE[3]) 和 PSS (UART_LINE[7]) 置位时，该位有效.
[7:0]	**DAT**	**数据接收/发送缓存** 写操作： 写数据到该寄存器, 数据将会保存到发送 FIFO. UART 控制器将会通过 UART_TXD 把存放在 FIFO 中最前面的数据发送出去 读操作： 读该寄存器, UART 将返回从接收 FIFO 中接收到的 8 位数据 (LSB 优先)

UART中断使能寄存器(UART_INTEN)

寄存器	偏移量	R/W	描述	复位值
UART_INTEN x=0,1,2,3,4,5,6,7	UARTx_BA+0x04	R/W	UART中断使能寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved	TXENDIEN	Reserved			ABRIEN	Reserved	SWBEIEN
15	14	13	12	11	10	9	8
RXPDMAEN	TXPDMAEN	ATOCTSEN	ATORTSEN	TOCNTEN	Reserved		LINIEN
7	6	5	4	3	2	1	0
Reserved	WKIEN	BUFERRIEN	RXTOIEN	MODEMIEN	RLSIEN	THREIEN	RDAIEN

位	描述	
[31:23]	**Reserved**	保留
[22]	**TXENDIEN**	**发送完成中断使能位** 使能 TXENDIEN (UART_INTEN[22]), 若 TXENDIF (UART_INTSTS[22]) 置位， (TX FIFO (UART_DAT) 为空且最后一字节的停止位已发送，则会产生发送完成中断 0 = 禁止发送完成中断 1 =使能发送完成中断
[21:19]	**Reserved**	保留
[18]	**ABRIEN**	**自动波特率中断使能位** 0 = 禁止自动波特率中断 1 = 使能自动波特率中断
[17]	**Reserved**	保留.
[16]	**SWBEIEN**	**单线位错误检测中断使能位** 设置该位，使能单线半双工位错误检测中断SWBEINT(UART_INTSTS[24]) 产生，当单线位错误检测SWBEIF(UART_INTSTS[16]) 被值位。 0 = 单线位错误检测中断禁止 1 =单线位错误检测中断使能 **注：**该位在FUNCSEL (UART_FUNCSEL[2:0])选择为单线模式下有效。
[15]	**RXPDMAEN**	**RX PDMA使能位** 该位可以使能或禁止 RXPDMA 功能 0 =禁止RX PDMA. 1 =使能RX PDMA **注:**如果 RLSIEN (UART_INTEN[2]) 为使能并且HWRLSINT (UART_INTSTS[26])=1， RLS（接收线状态）中断将产生。如果 RLS 中断是因为分隔错误标志 BIF

		(UART_FIFOSTS[6])，帧错误标志 FEF (UART_FIFO[5]) 或是奇偶校验错误标志 PEF (UART_FIFOSTS[4]) 产生， UART PDMA 接收请求将被停止。写“1”到错误标志 BIF、 FEF 和 PEF，分别清除分隔错误标志 BIF、帧错误标志 FEF 和奇偶校验错误标志 PEF 后，UART PDMA 接收请求将继续进行。
[14]	**TXPDMAEN**	**TX PDMA 使能位** 该位可以使能或禁止 TX PDMA 功能 0 =禁止TX PDMA 1 =使能TX PDMA **注:**如果 RLSIEN (UART_INTEN[2]) 为使能并且 HWRLSINT (UART_INTSTS[26])=1，RLS (接收线状态) 中断将产生。如果 RLS 中断是因为 break 错误标志 BIF (UART_FIFOSTS[6])，帧错误标志 FEF (UART_FIFO[5]) 或是奇偶校验错误标志 PEF (UART_FIFOSTS[4]) 产生， UART PDMA 接收请求将被停止。写“1”到错误标志 BIF、FEF 和 PEF，分别清除 break 错误标志 BIF、帧错误标志 FEF 和奇偶校验错误标志 PEF 后，UART PDMA 接收请求将继续进行。
[13]	**ATOCTSEN**	**nCTS 自动流控使能位** 0 = 禁止nCTS 自动流控 1 = 使能nCTS 自动流控 **注:**当 nCTS 自动流控使能后，当 nCTS 输入有效， UART 会发送数据到外部设备 (UART 将不会发送数据到外部设备直到 nCTS 有效)。
[12]	**ATORTSEN**	**nRTS 自动流控使能位** 0 = 禁止nRTS 自动流控 1 =使能 nRTS 自动流控 **注:**当 nRTS 自动流控使能后，如果 RX FIFO 中字节的数量等于 RTSTRGLV (UART_FIFO[19:16])， UART 会自动禁止 nRTS 信号。
[11]	**TOCNTEN**	**接收缓存超时计数器使能位** 0 = 禁止接收缓存超时计数器 1 = 使能接收缓存超时计数器
[10:9]	**Reserved**	保留
[8]	**LINIEN**	**LIN 总线中断使能位** 0 = 禁止LIN 总线中断 1 = 使能LIN 总线中断 **注:** 该位 LIN 模式下有效
[7]	**Reserved**	保留
[6]	**WKIEN**	**唤醒中断使能位** 0 =禁止唤醒中断 1 = 使能唤醒中断
[5]	**BUFERRIEN**	**缓存错误中断使能位** 0 = 禁止缓存错误中断 1 = 使能缓存错误中断
[4]	**RXTOIEN**	**RX 超时中断使能位** 0 = 禁止RX 超时中断 1 = 使能RX 超时中断.

[3]	**MODEMIEN**	**Modem 状态中断使能位** 0 = 禁止Modem 状态中断 1 = 使能Modem 状态中断
[2]	**RLSIEN**	**接收 Line 状态中断使能位** 0 = 禁止接收 Line 状态中断 1 = 使能接收 Line 状态中断
[1]	**THREIEN**	**发送保持寄存器空中断使能位** 0 = 禁止发送保持寄存器空中断 1 = 使能发送保持寄存器空中断
[0]	**RDAIEN**	**接收数据可用中断使能位** 0 = 禁止接收数据可用中断 1 = 使能接收数据可用中断

UART FIFO控制寄存器(UART_FIFO)

寄存器	偏移量	R/W	描述	复位值
UART_FIFO x=0,1,2,3,4,5,6,7	UARTx_BA+0x08	R/W	UART FIFO控制寄存器	0x0000_0101

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved				RTSTRGLV			
15	14	13	12	11	10	9	8
Reserved							RXOFF
7	6	5	4	3	2	1	0
RFITL				Reserved	TXRST	RXRST	Reserved

位	描述	
[31:20]	**Reserved**	保留
[19:16]	**RTSTRGLV**	**nRTS 自动流控触发阈值** 0000 = nRTS 触发阈值为 1 字节. 0001 = nRTS 触发阈值为 4 字节. 0010 = nRTS 触发阈值为 8 字节. 0011 = nRTS 触发阈值为 14 字节. Others = 保留. **注:**该区域用于自动 nRTS 流控制
[15:9]	**Reserved**	保留
[8]	**RXOFF**	**接收禁止位** 是否禁用接收 (置 1 禁用接收) 0 =使能接收 1 =禁用接收 **注：**该位用于 RS-485 普通多点模式,需要在 RS485NMM (UART_ALTCTL [8]) 被设置之前设置
[7:4]	**RFITL**	**RX FIFO 中断触发阈值** 当 FIFO 接收字节数等于 RFITL 后， RDAIF (UART_INTSTS[0]) 将被置位 (如果 RDAIEN (UART_INTEN [0]) 使能，将产生中断)。 0000 = RX FIFO 触发中断阈值为 1 字节 0001 = RX FIFO 触发中断阈值为 4 字节 0010 = RX FIFO 触发中断阈值为 8 字节 0011 = RX FIFO 触发中断阈值为 14 字节 其它 = 保留

[3]	**Reserved**	保留
[2]	**TXRST**	**TX 域软件复位** 当 TXRST (UART_FIFO[2]) 置位，发送 FIFO 和 TX 内部状态机中的所有数据将被清除. 0 = 无效 1 = 复位 TX 内部状态机和指针. **注 1:**最少 3 个 UART 外设时钟后，该位自动清 0 . **注 2:**设置该位前，需要先等待TXEMPTYF (UART_FIFOSTS[28]) 被置位
[1]	**RXRST**	**RX 域软件复位** 当 RXRST (UART_FIFO[1]) 置位，发送 FIFO 和 RX 内部状态机中的所有数据将被清除. 0 = 无效 1 = 复位 RX 内部状态机和指针. **注 1**: 最少 3 个 UART 外设时钟后，该位自动清 0 **注 2**: 设置该位前，需要先等待RXEMPTYF (UART_FIFOSTS[29])被置位
[0]	**Reserved**	保留

UART线控寄存器(UART_LINE)

寄存器	偏移量	R/W	描述	复位值
UART_LINE x=0,1,2,3,4,5,6,7	UARTx_BA+0x0C	R/W	UART线控寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved						RXDINV	TXDINV
7	6	5	4	3	2	1	0
PSS	BCB	SPE	EPE	PBE	NSB	WLS	

位	描述	
[31:10]	**Reserved**	保留
[9]	**RXDINV**	**RX 数据反转** 0 = 禁止接收数据信号反转. 1 = 使能接收数据信号反转. **注 1:** 设置该位前，需要先置位 TXRXDIS (UART_FUNCSEL[3])，再等 TXRXACT (UART_FIFOSTS[31]) 被清除。完成配置后，清除 TXRXDIS (UART_FUNCSEL[3]) 以触发 UART 控制器 **注 2:**该位仅在 FUNCSEL (UART_FUNCSEL[1:0]) 配置为 UART, LIN 或 RS485 模式下有效
[8]	**TXDINV**	**TX 数据反转** 0 = 禁止发送数据信号反转. 1 = 使能发送数据信号反转. **注 1:** 设置该位前，需要先置位 TXRXDIS (UART_FUNCSEL[3])，再等 TXRXACT (UART_FIFOSTS[31]) 被清除。完成配置后，清除 TXRXDIS (UART_FUNCSEL[3]) 以触发 UART 控制器 **注 2:**该位仅在 FUNCSEL (UART_FUNCSEL[1:0]) 配置为 UART, LIN 或 RS485 模式下有效
[7]	**PSS**	**校验位源选择** 校验位可以由软件或是自动产生并校验 0 = 校验位由设置EPE (UART_LINE[4]) 和 SPE (UART_LINE[5])并自动检查. 1 = 校验位由软件产生并校验. **注 1:** 仅 PBE (UART_LINE[3]) 置位时，该位有效. **注 2:** 如果 PSS 为 0，校验位自动发送并检查。如果 PSS 为 1，发送出去的校验位可由 PARITY (UART_DAT[8]) 设置，读 PARITY (UART_DAT[8]) 可得校验位

[6]	**BCB**	**Break 控制位** 0 = 禁止Break 控制 1 = 使能Break 控制 **注:**当该位被置逻辑 1，串行数据输出 (TX) 将强制到 Spacing 状态 (logic 0)。该位仅作用于 TX，对传输逻辑不起作用
[5]	**SPE**	**Stick 校验使能位** 0 = 禁止Stick 校验. 1 = 使能Stick 校验. **注:** 如果 PBE (UART_LINE[3]) 和 EPE (UART_LINE[4]) 为逻辑 1，校验位发送和检验值为逻辑 0。如果 PBE (UART_LINE[3]) 是1，EPE (UART_LINE[4]) 是 0，则校验位发送和检验值为 1。
[4]	**EPE**	**偶校验使能位** 0 = 逻辑 1 的奇数数目在每个字节中被发送和检验 1 = 逻辑 1 的偶数数目在每个字中被发送和检验 **注:**该位只在 PBE (UART_LINE[3]) 置位时有效。
[3]	**PBE**	**校验使能位** 0 = 禁止生成校验位 1 = 使能生成校验位 **注:**每一个发送字符中都产生校验位，对每一个传进来的数据进行校验位检测
[2]	**NSB**	**停止位数目** 0= 当发送数据时，产生 1 个“ STOP bit” 1= 当发送数据，选择 5-位字长度时，产生 1.5 “ STOP bit” 当选择 6-, 7- 和 8-位字长度时，产生 2 个“STOP bit”.
[1:0]	**WLS**	**字长选择** 该域选择 UART 的字长. 00 = 5 位. 01 = 6 位. 10 = 7 位. 11 = 8 位.

UART Modem控制寄存器(UART_MODEM)

寄存器	偏移量	R/W	描述	复位值
UART_MODEM x=0,1,2,3,4,5,6,7	UARTx_BA+0x10	R/W	UART Modem控制寄存器	0x0000_0200

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved		RTSSTS	Reserved			RTSACTLV	Reserved
7	6	5	4	3	2	1	0
Reserved						RTS	Reserved

位	描述	
[31:14]	**Reserved**	保留
[13]	**RTSSTS**	**nRTS 管脚状态 (只读)** 该位的值对应于 nRTS 管脚的输出电平 0 = nRTS 管脚为低电平逻辑状态 1 = nRTS 管脚为高电平逻辑状态
[12:10]	**Reserved**	保留
[9]	**RTSACTLV**	**nRTS 脚的有效电平** 该位定义了 nRTS 输出管脚的有效电平 0 = nRTS 管脚输出为高电平有效. 1 = nRTS 管脚输出为低电平有效.(默认) **注 1:** UART 的功能模式请参考图6.16-13和图6.16-14。 **注 2:** RS-485 功能模式请参考图6.16-24和图6.16-25。 **注 3:** 设置该位前，需要先置位 TXRXDIS (UART_FUNCSEL[3])，再等 TXRXACT (UART_FIFOSTS[31]) 被清除。完成配置后，清除 TXRXDIS (UART_FUNCSEL[3]) 以触发 UART 控制器。
[8:2]	**Reserved**	保留

[1]	**RTS**	**nRTS(请求发送) 信号控制** 该位直接控制内部 nRTS 脚信号是否有效，然后使用 RTSACTLV 位的配置驱动 nRTS 脚输出 0 = nRTS 信号有效 1 = nRTS 信号无效 **注 1:** UART 功能模式下，当 nRTS 自动流控被使能后，nRTS 信号控制位无效。 **注 2:** RS-485 模式下，当 RS-485 自动方向模式（AUD) 被使能后，nRTS 信号控制位无效。
[0]	**Reserved**	保留

UART Modem状态寄存器(UART_MODEMSTS)

寄存器	偏移量	R/W	描述	复位值
UART_MODEMSTS x=0,1,2,3,4,5,6,7	UARTx_BA+0x14	R/W	UART Modem状态寄存器	0x0000_0110

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							CTSACTLV
7	6	5	4	3	2	1	0
Reserved			CTSSTS	Reserved			CTSDETF

位	描述	
[31:9]	**Reserved**	保留
[8]	**CTSACTLV**	**nCTS 管脚有效电平** 该位定义了 nCTS 输入管脚脚的有效电平 0 = nCTS 输入脚高电平有效 1 = nCTS 输入脚低电平有效.(默认) **注：** 设置该位前，需要先置位 TXRXDIS (UART_FUNCSEL[3])，再等 TXRXACT (UART_FIFOSTS[31]) 被清除。完成配置后，清除 TXRXDIS (UART_FUNCSEL[3]) 以触发 UART 控制器
[7:5]	**Reserved**	保留
[4]	**CTSSTS**	**nCTS 管脚状态 (只读)** 该位对应于 nCTS 管脚输入逻辑状态 0 = nCTS 管脚输入状态为低电平 1 = nCTS 管脚输入状态为高电平 **注:**当 UART 控制器外设时钟被使能，且 nCTS 多功能管脚被使能，该位才有效。
[3:1]	**Reserved**	保留
[0]	**CTSDETF**	**检测到 nCTS 状态改变标志** 如果 nCTS 输入脚上有电平变化，该位将被置位，如果 MODEMIEN (UART_INTEN [3]) 位被 置位，将会产生 Modem 中断。 0 = nCTS 输入管脚状态无变化.

		1 = nCTS 输入管脚状态已变化. **注:**该位写 1 清 0 .

UART FIFO状态寄存器(UART_FIFOSTS)

寄存器	偏移量	R/W	描述	复位值
UART_FIFOSTS x=0,1,2,3,4,5,6,7	UARTx_BA+0x18	R/W	UART FIFO状态寄存器	0xB040_4000

31	30	29	28	27	26	25	24
TXRXACT	Reserved	RXIDLE	TXEMPTYF	Reserved			TXOVIF
23	22	21	20	19	18	17	16
TXFULL	TXEMPTY	TXPTR					
15	14	13	12	11	10	9	8
RXFULL	RXEMPTY	RXPTR					
7	6	5	4	3	2	1	0
Reserved	BIF	FEF	PEF	ADDRDETF	ABRDTOIF	ABRDIF	RXOVIF

位	描述	
[31]	**TXRXACT**	**TX 和 RX 有效状态 (只读)** 该位反映 TX 和 RX 是否有效 0 = TX 和 RX 无效. 1 = TX 和 RX 有效.(默认) **注:**当 TXRXDIS (UART_FUNCSEL[3]) 置位，TX 和 RX 都空闲时，该位清 0。UART 控制器此时无法发送或接收数据。其他情况，该位均为 1.
[30]	**Reserved**	保留
[29]	**RXIDLE**	**RX 空闲状态 (只读)** RX 空闲时，硬件自动置位该位. 0 = RX 忙. 1 = RX 空闲 (默认)
[28]	**TXEMPTYF**	**发送空标志 (只读)** 当 TX FIFO (UART_DAT) 为空,并且最后一个字节的 STOP 位也已经被发送完毕，该位被硬件置 1. 0 = TX FIFO 不为空或最后一个字节的 STOP 位还没有发送. 1 = TX FIFO 为空而且最后一个字节的 STOP 位已经发送 **注:**当 TX FIFO 不为空或最后一个字节的 STOP 位还没有被发送完毕，那么该位将被自动清零
[27:25]	**Reserved**	保留.
[24]	**TXOVIF**	**TX 溢出错误中断标志 (只读)** 如果 TX FIFO (UART_DAT) 满，如果再向 UART_DAT 写入数据，将会导致此位被置 1.. 0 = TX FIFO 未溢出

		1 = TX FIFO 已溢出.. **注:**该位写 1 清 0 .
[23]	**TXFULL**	**发送 FIFO 满 (只读)** 此位用于指示 TX FIFO 是否已满.. 0 = TX FIFO 未满 1 = TX FIFO 已满. **注:**当 TX FIFO Buffer 中的字节数量等于 16,此位将被置 1.否则被硬件清零.
[22]	**TXEMPTY**	**发送 FIFO 空 (只读)** 此位指示 TX FIFO 是否为空。 0 = TX FIFO 不为空. 1 = TX FIFO 为空. **注:**当 TX FIFO 中的最后一个字节被发送到发送移位寄存器，硬件将把此位置 1.当写入数据到 UART_DAT 中, (TX FIFO 不为空)，此位被清零
[21:16]	**TXPTR**	**TX FIFO 指针 (只读)** 此位指示 TX FIFO 缓冲区指针位置。当 CPU 写一个字节到UART_DAT 寄存器，TXPTR 将累加 1.当 TX FIFO 发送一个字节到发送移位寄存器中，TXPTR 指针将减 1. TXPTR 显示的最大值是 15。当 TX FIFO 缓冲区所填充数据数量达到 16，TXFULL 将被置 1，TXPTR 显示为0。如果 TX FIFO 中发送一个字节到发送移位寄存器，TXFULL 位将被清零，TXPTR 显示 15。
[15]	**RXFULL**	**接收 FIFO 满 (只读)** 该位指示 RX FIFO 是否已满。 0 = RX FIFO 未满 1 = RX FIFO 已满 **注:**当 RX FIFO 缓冲区中的数据数量等于 16 后，此位将被置 1，否则被硬件清零。
[14]	**RXEMPTY**	**接收 FIFO 空 (只读)** 此位指示 RX FIFO 是否为空。 0 = RX FIFO 不为空. 1 = RX FIFO 为空. **注:**当 RX FIFO 中最后一个字节被 CPU 读取后，硬件将对此位置 1，UART 接收到新数据后此位将被清零。
[13:8]	**RXPTR**	**RX FIFO 指针 (只读)** 此位指示 RX FIFO 缓冲区指针。当 UART 从外部设备接收到一个字节，RXPTR 将累加 1.当 RX FIFO 的数据被 CPU 读取一个字节，RXPTR 将递减 1. RXPTR 显示的最大值是 15。当 RX FIFO 的数据达到 16，RXFULL 位将被置 1，RXPTR 显示 0，RX FIFO 当中的数据被 CPU 读取一个后，RXFULL 将被清零，RXPTR 显示 15。
[7]	**Reserved**	保留
[6]	**BIF**	**Break 中断标志位 (只读)** 每当接收到数据输入 (RX) 维持在 “空状态” (电平 0) 的时间长于一个全字的传输时间 (即开始位 + 数据位 + 校验位 + 停止位的总时间)，该位置 1。当 CPU 向该位写 1，该位重置。 0 = 没有 Break 中断产生. 1 = 有 Break 中断产生.

		注:该位写 1 清 0 .
[5]	**FEF**	**帧错误标志 (只读)** 每当接收到的字符没有有效的 “停止位”(及跟在最后的数据位后或奇偶校验位后的数据为 0) 该位置 1。 0 = 没有帧错误产生 1 = 有帧错误产生. **注:**该位写 1 清 0 .
[4]	**PEF**	**奇偶校验错误标志 (只读)** 每当接收到的字符没有有效的奇偶校验位，该位置 1。当 CPU 向该位写 1，该位重置。 0 = 没有奇偶校验错误产生 1 = 有奇偶校验错误产生 **注:**该位写 1 清 0 .
[3]	**ADDRDETF**	**RS-485 地址位检测标志 (只读)** 0 = 接收到的数据没有地址位标记 (bit 9 =’0’). 1 = 接收到的数据有地址位标记 (bit 9 =’1’). **注 1:**此位用于 RS-485 功能模式，且ADDRDEN (UART_ALTCTL[15]))位被置 1 使能地址检测模式 **注 2:**该位写 1 清 0 .
[2]	**ABRDTOIF**	**自动波特率检测超时中断 (只读)** 0 = 自动波特率计数器没有溢出. 1 = 自动波特率计数器溢出. **注:**该位写 1 清 0 .
[1]	**ABRDIF**	**自动波特率检测中断** 当自动波特率检测完成时该位将被置为 “1”. 0 = 自动波特率检测还没有完成. 1 = 自动波特率检测已经完成 **注:**该位写 1 清 0 .
[0]	**RXOVIF**	**RX 溢出错误标志 (只读)** 当 RX FIFO 溢出时被置 1 如果接收到的数据数量大于RX_FIFO (UART_DAT) 16字节，该位将被置位. 0 = RX FIFO 未溢出. 1 = RX FIFO 溢出 **注:**该位写 1 清 0 .

UART中断状态寄存器(UART_INTSTS)

寄存器	偏移量	R/W	描述	复位值
UART_INTSTS x=0,1,2,3,4,5,6,7	UARTx_BA+0x1C	R/W	UART中断状态寄存器	0x0040_0002

31	30	29	28	27	26	25	24
ABRINT	TXENDINT	HWBUFEINT	HWTOINT	HWMODINT	HWRLSINT	Reserved	SWBEINT
23	22	21	20	19	18	17	16
Reserved	TXENDIF	HWBUFEIF	HWTOIF	HWMODIF	HWRLSIF	Reserved	SWBEIF
15	14	13	12	11	10	9	8
LININT	WKINT	BUFERRINT	RXTOINT	MODEMINT	RLSINT	THREINT	RDAINT
7	6	5	4	3	2	1	0
LINIF	WKIF	BUFERRIF	RXTOIF	MODEMIF	RLSIF	THREIF	RDAIF

位	描述	
[31]	**ABRINT**	**自动波特率中断指示 (只读)** ABRIEN (UART_INTEN[18]) 和 ABRIF (UART_ALTCTL[17]) 都为 1，该位置 1 . 0 = 无自动波特率中断产生. 1 = 有自动波特率中断产生.
[30]	**TXENDINT**	**发送空中断指示器(只读)** TXENDIEN (UART_INTEN[22])和 TXENDIF (UART_INTSTS[22]) 都为 1，该位置 1. 0 = 无发送空中断发生. 1 = 有发送空中断发生.
[29]	**HWBUFEINT**	**PDMA 模式缓存错误中断 (只读)** BUFERRIEN (UART_INTEN[5]) 和 HWBUFEIF (UART_INTSTS[21]) 都为 1，该位置 1. 0 = PDMA 模式下，无缓存错误中断发生. 1 = PDMA 模式下，有缓存错误中断发生.
[28]	**HWTOINT**	**PDMA 模式 RX 超时中断指示 (只读)** RXTOIEN (UART_INTEN[4]) 和 HWTOIF (UART_INTSTS[20]) 都为 1,该位被置 1. 0 = PDMA 模式时,无 RX 超时中断产生 1 = PDMA 模式时,有 RX 超时中断产生
[27]	**HWMODINT**	**PDMA 模式 MODEM 状态中断指示 (只读)** MODEMIEN (UART_INTEN[3]) 和 HWMODIF (UART_INTSTS[19]) 都为 1，该位置 1. 0 = PDMA 模式下，无 MODEM 状态中断发生. 1 = PDMA 模式下，有 MODEM 状态中断发生.
[26]	**HWRLSINT**	**PDMA 模式 RLS 中断指示 (只读)** RLSIEN (UART_INTEN[2]) 和 HWRLSIF (UART_INTSTS[18]) 都为 1，该位置 1.

		0 = PDMA 模式下，无 RLS 中断发生. 1 = PDMA 模式下，有 RLS 中断发生.
[25]	**Reserved**	保留.
[24]	**SWBEINT**	**单线位错误检测中断指示(只读)** SWBEIEN (UART_INTEN[16]) 和 SWBEIF (UART_INTSTS[16])都为1，该位置1 0 = 无单线位错误检测中断发生 1 = 有单线位错误检测中断发生
[23]	**Reserved**	保留.
[22]	**TXENDIF**	**发送空中断标志** 该位在 TX FIFO (UART_DAT) 空且最后一字节的停止位已发送出去 (TXEMPTYF (UART_FIFOSTS[28]) 置位) 时置位。如果使能了 TXENDIEN (UART_INTEN[22])，发送空中断触发 0 = 未触发发送空中断标志. 1 = 已触发发送空中断标志 **注:**该位在 TX FIFO 为非空或最后一字节的发送还未完成时由硬件自动清空
[21]	**HWBUFEIF**	**PDMA 模式缓存错误中断标志 (只读)** 当 TX 或 RX FIFO 溢出 (即 (TXOVIF (UART_FIFOSTS [24]) 或 RXOVIF (UART_FIFOSTS[0]) 被置 1),该位被置 1.当 BERRIF (UART_INTSTS[5]) 被置位,传送有可能出错. 如果 BFERRIEN (UART_INTEN [5]) 为使能,缓存错误中断将产生 0= PDMA 模式下,无缓存错误中断标志产生 1= PDMA 模式下,有缓存错误中断标志产生 **注**:当 TXOVIF (UART_FIFOSTS[24]]) 和 RXOVIF (UART_FIFOSTS[0]) 都被清零时,该位也会被清零.
[20]	**HWTOIF**	**PDMA 模式 RX 超时中断标志 (只读)** 当 RX FIFO 不为空而且持续没有变化,当超时计数器达到 TOIC (UART_TOUT[7:0]) 时,该位被置 1.如果 RXTOIEN (UART_INTEN [4]) 为使能,超时溢出中断将产生 0= PDMA 模式下,无超时溢出中断标志产生 1= PDMA 模式下,有超时溢出中断标志产生 **注**:该位只读,用户可以通过读 UART_DAT (RX 为活动状态) 将该位清零.
[19]	**HWMODIF**	**PDMA 模式 MODEM 中断标志 (只读)** 当 nCTS 管脚状态发生变化, (CTSDETF (UART_CTSDETF[0] =1)) 时,该位被置 1.如果 MODEMIEN (UART_INTEN [3]) 位使能, Modem 中断将产生 0= PDMA 模式下,无 Modem 中断标志产生 1= PDMA 模式下,有 Modem 中断标志产生 **注**:该位只读,当写 1 到 CTSDETF (UART_CTSDETF [0]) 时, UART_CTSDETF (US_MSR[0]) 将被清零,该位也将被清零.
[18]	**HWRLSIF**	**PDMA 模式 RLS 中断标志 (只读)** 当 RX 接收数据出现校验错误,帧错误或 Break 错误,即至少 BIF (UART_FIFOSTS[6]), FEF (UART_FIFOSTS[5]) 和 PEF (UART_FIFOSTS[4]) 中的一位为 1,该位被置 1. 如果 RLSIEN (UART_INTEN [2])) 为使能, RLS 中断将产生 0= PDMA 模式时,无 RLS 中断标志产生 1= PDMA 模式时,有 RLS 中断标志产生 **注 1**: 在 RS-485 模式,该域包括"地址检测字节 (bit9 = '1')" **注 2**: 在 UART 模式,该位只读,当 BIF (UART_FIFOSTS[6]) , FEF (UART_FIFOSTS[5]) 和

		PEF (UART_FIFOSTS[4]) 都被清零时,该位也将被清零. **注 3**: 在 RS-485 模式,该位只读,当 BIF (UART_FIFOSTS[6]) , FEF (UART_FIFOSTS[5]) , PEF (UART_FIFOSTS[4]) 和 ADDRDETF (UART_FIFOSTS[3]) 都被清零时,该位也将被清零.
[17]	**Reserved**	保留.
[16]	**SBEIF**	**单线位错误检测中断标志** 在单线模式下，当单线转态不是UART控制器TX转态时，该位置位。 0 = 无单线位错误检测中断标志产生 1 =有单线位错误检测中断标志产生 **注1:** 当 FUNCSEL (UART_FUNCSEL[2:0])选择为UART单线模式下，该位有效。 **注2:**写1清除该位。
[15]	**LININT**	**LIN 总线中断指示 (只读)** LINIEN (UART_INTEN[8]) 和 LINIF (UART_INTSTS[7]) 都为 1，该位置 1. 0 = 无 LIN 总线中断发生. 1 = 有 MODEM 状态中断发生.
[14]	**WKINT**	**UART 唤醒中断指示 (只读)** WKIEN (UART_INTEN[6]) 和 WKIF (UART_INTSTS[6])]) 都为 1，该位置 1. 0 = 无 UART 唤醒中断发生. 1 = 有 UART 唤醒中断发生.
[13]	**BUFERRINT**	**缓存错误中断指示 (只读)** BUFERRIEN (UART_INTEN[5]) 和 BUFERRIF (UART_ INTSTS[5]) 都为 1，该位置 1. 0 = 无缓存错误中断发生. 1 = 有缓存错误中断发生.
[12]	**RXTOINT**	**RX 超时中断指示 (只读)** RXTOIEN (UART_INTEN[4]) 和 RXTOIF (UART_INTSTS[4]) 都为 1，该位置 1. 0 = 无 RX 超时中断发生. 1 = 有 RX 超时中断发生.
[11]	**MODEMINT**	**MODEM 状态中断 (只读)** MODEMIEN (UART_INTEN[3]) 和 MODEMIF (UART_INTSTS[3]) 都为 1，该位置 1. 0 = 无 MODEM 状态中断发生. 1 = 有 MODEM 状态中断发生.
[10]	**RLSINT**	**RLS 中断指示 (只读)** RLSIEN (UART_INTEN[2]) 和 RLSIF (UART_INTSTS[2])都为 1，该位置 1. 0 = 无 RLS 中断发生. 1 = 有 RLS 中断发生.
[9]	**THREINT**	**发送保持寄存器空中断指示 (只读)** THREIEN (UART_INTEN[1]) 和 THREIF (UART_INTSTS[1]) 都为 1，该位置 1. 0 = 无发送保持寄存器空中断发生. 1 = 有发送保持寄存器空中断发生.
[8]	**RDAINT**	**RDA 中断指示 (只读)** RDAIEN (UART_INTEN[0]) 和 RDAIF (UART_INTSTS[0])都为 1，该位置 1.

		0 = 无 RDA 中断发生. 1 = 有 RDA 中断发生.
[7]	**LINIF**	**LIN 总线中断标志(只读)** 当 LIN 从机报头侦测(SLVHDETF (UART_LINSTS[0] =1))，LIN break 侦测(BRKDETF (UART_LINSTS[8]=1))，位错误侦测(BITEF (UART_LINSTS[9]=1，LIN 从机 ID 校验错误(SLVIDPEF (UART_LINSTS[2] = 1) 或 LIN 从机报头错误侦测(SLVHEF (UART_LINSTS[1]) = 1)，该位置 1。如果 LIN_ IEN (UA_IER [8]) 被使能，LIN 中断产生。 0 = SLVHDETF, BRKDETF, BITEF, SLVIDPEF 和 SLVHEF 中没有任何一个标志产生 1 = SLVHDETF, BRKDETF, BITEF, SLVIDPEF 和 SLVHEF 中至少有一个标志产生 **注:**该位只读。当SLVHDETF (UART_LINSTS[0]), BRKDETF (UART_LINSTS[8]), BITEF (UART_LINSTS[9]), SLVIDPEF (UART_LINSTS[2]) 和 SLVHEF (UART_LINSTS[1]) 都被清 0 时，该位清 0。软件写 1 也可以清 0 该位。
[6]	**WKIF**	**UART 唤醒中断标志 (只读)** TOUTWKF (UART_WKSTS[4]), RS485WKF (UART_WKSTS[3]), RFRTWKF (UART_WKSTS[2]), DATWKF (UART_WKSTS[1]) 或 CTSWKF (UART_WKSTS[0])]) 为 1，该位置 1. 0 = 无 UART 唤醒中断标志. 1 = 有 UART 唤醒中断标志. **注:**写 1 清 0 TOUTWKF, RS485WKF, RFRTWKF, DATWKF 和 CTSWKF 后，该位清 0.
[5]	**BUFERRIF**	**缓存错误中断标志(只读)** TX FIFO 或 RX FIFO 溢出 (TXOVIF (UART_FIFOSTS[24]) 或 RXOVIF (UART_FIFOSTS[0]) 置位)，该位置位. 传输错误时，BUFERRIF (UART_INTSTS[5]) 置位. 如果使能了 BUFERRIEN (UART_INTEN [5]), 会产生缓存错误中断 0 = 无缓存错误中断产生. 1 = 有缓存错误中断产生. **注:**写 1 到 RXOVIF (UART_FIFOSTS[0]) 和 TXOVIF (UART_FIFOSTS[24]) 清空 RXOVIF (UART_FIFOSTS[0]) 和 TXOVIF (UART_FIFOSTS[24]) 后，该位清 0
[4]	**RXTOIF**	**RX 超时中断标志 (只读)** 如果 RX FIFO 为非空且无其他事件时，当超时计数器计数到 TOIC (UART_TOUT[7:0]) 时，该位置位。此时如果使能了 RXTOIEN (UART_INTEN [4])，会产生 RX 超时中断。 0 = 无 RX 超时中断标志产生. 1 = 有 RX 超时中断标志产生.. **注:**该位只读，读 UART_DAT (RX 有效时) 清 0 该位.
[3]	**MODEMIF**	**MODEM 中断标志 (只读)** 如果 nCTS 管脚电平发生改变 (CTSDETF (UART_MODEMSTS[0]) = 1)，该位置位。此时如果使能了 MODEMIEN (UART_INTEN [3])，会产生 Modem 中断 0 = 无 Modem 中断标注产生. 1 = 有 Modem 中断标注产生. **注:**该位只读，写 1 到 CTSDETF (UART_MODEMSTS[0]) 清 0 CTSDETF 后，该位清 0
[2]	**RLSIF**	**RLS 中断标志 (只读)** 当 RX 接收数据出现校验错误,帧错误或 Break 错误,即至少 BIF (UART_FIFOSTS[6]), FEF (UART_FIFOSTS[5]) 和 PEF (UART_FIFOSTS[4]) 中的一位为 1,该位被置 1. 如果 RLSIEN (UART_INTEN [2])) 为使能, RLS 中断将产生 0= 无 RLS 中断标志产生 1= 有 RLS 中断标志产生

		注 1: 在 RS-485 模式,当“接收检测任一接收到的地址字节字符 (bit9 = ‘1’) 位”时， 该位也会置1，同时， ADDRDETF (UART_FIFOSTS[3])位也被置1 **注 2:** 该位只读,当 BIF (UART_FIFOSTS[6]) , FEF (UART_FIFOSTS[5]) 和 PEF (UART_FIFOSTS[4]) 都被清零时,该位也将被清零. **注 3:**在 RS-485 模式,该位只读,当 BIF (UART_FIFOSTS[6]) , FEF (UART_FIFOSTS[5]) , PEF (UART_FIFOSTS[4]) 和 ADDRDETF (UART_FIFOSTS[3]) 都被清零时,该位也将被清零.
[1]	**THREIF**	**发送保持寄存器空中断标志 (只读)** 当 TX FIFO 中最后数据传输到发送移位寄存器时，该位置位。如果 THREIEN (UART_INTEN[1]) 使能，THRE 中断将产生。 0 =没有 THRE 中断标志产生 1 =有 THRE 中断标志产生. **注:**该位只读，当写数据到UART_DAT (TX FIFO 非空) 时，该位将被清除
[0]	**RDAIF**	**接收数据可用中断标志 (只读)** 当 RX FIFO中的数据数量等于 RFITL 时，RDAIF (UART_INTSTS[0]) 将被置位。如果 RDAIEN (UART_INTEN [0]) 使能，RDA 中断将产生。 0 =没有 RDA 中断标志产生. 1 =有 RDA 中断标志产生. **注:** 该位只读，当 RX FIFO 的未读数据低于阈值(RFITL (UART_FIFO[7:4]) 时，该位将被清除

UART超时寄存器(UART_TOUT)

寄存器	偏移量	R/W	描述	复位值
UART_TOUT x=0,1,2,3,4,5,6,7	UARTx_BA+0x20	R/W	UART超时寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
DLY							
7	6	5	4	3	2	1	0
TOIC							

位	描述	
[31:16]	**Reserved**	保留
[15:8]	**DLY**	**TX 延迟时间值** 该域用于设置上一次停止位和下一次开始位之间的传输延迟时间
[7:0]	**TOIC**	**超时中断比较器** 使能 TOCNTEN (UART_INTEN[11]) 超时计数功能后，每当 RX FIFO 接收到一个新的数据字，定时溢出计数器都会重置并开始计数 (计数时钟 = 波特率)。一旦超时计数器的内容等于超时中断比较器(TOIC (UART_TOUT[7:0]))，如果此时RXTOIEN (UART_INTEN [4]) 使能，则接收超时中断(RXTOINT (UART_INTSTS[12])) 产生。接收到新的数据或 RX FIFO 为空将把RXTOIF (UART_INTSTS[4])清零。为了避免接收超时中断在接收到一个字符就立即产生，TOIC 的值必须设置在 40 到 255 之间。例如，如果 TOIC 为 40，当 UART 传输设置为 1 位停止位且无奇偶校验位时，在 4 个字符时间长度后还没收到数据，超时中断将产生。

UART波特率分频寄存器(UART_BAUD)

寄存器	偏移量	R/W	描述	复位值
UART_BAUD x=0,1,2,3,4,5,6,7	UARTx_BA+0x24	R/W	UART波特率分频寄存器	0x0F00_0000

31	30	29	28	27	26	25	24
Reserved		BAUDM1	BAUDM0	EDIVM1			
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
BRD							
7	6	5	4	3	2	1	0
BRD							

位	描述	
[31:30]	**Reserved**	保留
[29]	**BAUDM1**	**波特率模式选择位 1** 该位为波特率模式选择位 1。UART 提供三种波特率计算方式。该位和 BAUDM0 (UART_BAUD[28]) 组合选择波特率计算方式。详细描叙在表6.16-4中有介绍。 **注**：IrDA 模式时必须选择模式 0
[28]	**BAUDM0**	**波特率模式选择位 0** 该位为波特率模式选择位 0。UART 提供三种波特率计算方式。该位和BAUDM1 (UART_BAUD[29]) 组合选择波特率计算方式。详细描叙在表6.16-4中有介绍。
[27:24]	**EDIVM1**	**波特率模式 1 的扩展分频** 该域用于波特率计算模式 1，对于波特率计算模式 0 和模式 2 无效。详细描叙在表6.16-4中有介绍
[23:16]	**Reserved**	保留
[15:0]	**BRD**	**波特率分频** 该域用于波特率分频。详细描叙在表6.16-4中有介绍

UART IrDA控制寄存器(UART_IRDA)

寄存器	偏移量	R/W	描述	复位值
UART_IRDA x=0,1,2,3,4,5,6,7	UARTx_BA+0x28	R/W	UART IrDA控制寄存器	0x0000_0040

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved	RXINV	TXINV	Reserved			TXEN	Reserved

位	描述	
[31:7]	**Reserved**	保留
[6]	**RXINV**	**IrDA 接收信号反转** 0 = 不反转接收信号. 1 = 反转接收信号 (默认) **注 1:** 设置该位前，需要置位 TXRXDIS (UART_FUNCSEL[3]) 再等 TXRXACT (UART_FIFOSTS[31]) 清除。配置完成后，清除 TXRXDIS (UART_FUNCSEL[3]) 以激活 UART 控制器 **注 2:**该位在 FUNCSEL (UART_FUNCSEL[1:0]) 选择为 IrDATA 功能时有效
[5]	**TXINV**	**IrDA 发送信号反转** 0 = 不反转发送信号. 1 = 反转发送信号 (默认) **注 1:** 设置该位前，需要置位 TXRXDIS (UART_FUNCSEL[3]) 再等 TXRXACT (UART_FIFOSTS[31]) 清除。配置完成后，清除 TXRXDIS (UART_FUNCSEL[3]) 以激活 UART 控制器 **注 2:**该位在 FUNCSEL (UART_FUNCSEL[1:0]) 选择为 IrDATA 功能时有效
[4:2]	**Reserved**	保留
[1]	**TXEN**	**IrDA 接收/发送选择使能位** 0 = IrDA 禁止发送和使能接收(默认) 1 = IrDA 使能发送和禁止接收.
[0]	**Reserved**	保留

UART选择控制/状态寄存器(UART_ALTCTL)

寄存器	偏移量	R/W	描述	复位值
UART_ALTCTL x=0,1,2,3,4,5,6,7	UARTx_BA+0x2C	R/W	UART选择控制/状态寄存器	0x0000_000C

31	30	29	28	27	26	25	24
ADDRMV							
23	22	21	20	19	18	17	16
Reserved			ABRDBITS		ABRDEN	ABRIF	Reserved
15	14	13	12	11	10	9	8
ADDRDEN	Reserved				RS485AUD	RS485AAD	RS485NMM
7	6	5	4	3	2	1	0
LINTXEN	LINRXEN	Reserved		BRKFL			

位	描述	
[31:24]	**ADDRMV**	**地址匹配值** 该位包括 RS-485 地址匹配值. **注：** 该域用于 RS-485 自动地址检测模式
[23:21]	**Reserved**	保留
[20:19]	**ABRDBITS**	**自动波特率检测位长** 00 = 1-位时长，从起始位到第一个上升沿。输入数据格式应该为 0x01。 01 = 2-位时长，从起始位到第一个上升沿。输入数据格式应该为 0x02。 10 = 4-位时长，从起始位到第一个上升沿。输入数据格式应该为 0x08。 11 = 8-位时长，从起始位到第一个上升沿。输入数据格式应该为 0x80。 **注意：** 计算的位数包括起始位。
[18]	**ABRDEN**	**自动波特率检测使能位** 0 = 禁止自动波特率检测功能. 1 = 使能自动波特率检测功能 自动检测结束后，该位将被自动清零。
[17]	**ABRIF**	**自动波特率中断标志 (只读)** 当自动波特率检测结束，或者自动波特率计数器发生溢出，如果 ABRIEN (UART_INTEN[18]) 为 1，那么自动波特率中断将会产生. **注：**该位为只读位，但可以通过写“1”到 ABRDTOIF (UART_FIFOSTS[2]) 和 ABRDIF (UART_FIFOSTS[1]) 将该位清零。
[16]	**Reserved**	保留
[15]	**ADDRDEN**	**RS-485 地址检测使能位** 该位用于使能 RS-485 地址检测模式.

		0 =禁止地址检测模式. 1 =使能地址检测模式. **注**：该位适用于各种 RS-485 操作模式
[14:11]	**Reserved**	保留
[10]	**RS485AUD**	**RS-485 自动方向模式 (AUD)** 0 =RS-485 自动方向操作模式 (AUD) 禁止 1 =RS-485 自动方向操作模式 (AUD) 使能 **注**：仅在RS-485_AAD 或 RS-485_NMM 操作模式有效
[9]	**RS485AAD**	**RS-485 自动地址检测操作模式 (AAD)** 0 =禁止RS-485 自动地址检测模式 (AAD) 1 =使能RS-485 自动地址检测模式 (AAD) **注**：在 RS-485_NMM 操作模式下无效
[8]	**RS485NMM**	**RS-485 普通多点操作模式(NMM)** 0 =禁止RS-485普通多点操作模式(NMM) 1 =使能RS-485普通多点操作模式(NMM) **注**：在 RS-485_AAD 操作模式下无效.
[7]	**LINTXEN**	**LIN TX Break 模式使能** 0 = 禁止LIN TX Break 模式 1 =使能LIN TX Break 模式 **注**：当 TX 间隔域传输操作完成后，该位将被自动清除.
[6]	**LINRXEN**	**LIN RX 使能** 0 =禁止LIN RX 模式 1 =使能LIN RX 模式
[5:4]	**Reserved**	保留
[3:0]	**BRKFL**	**UART LIN 间隔域长度** 该域表示一个4-位 LIN TX 间隔域数量 **注 1:**该间隔域长度为BRKFL + 1 **注 2:**根据 LIN 规范, 复位值是 0xC (间隔域长度= 13).

UART功能选择寄存器(UART_FUNCSEL)

寄存器	偏移量	R/W	描述	复位值
UART_FUNCSEL x=0,1,2,3,4,5,6,7	UARTx_BA+0x30	R/W	UART功能选择寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved				TXRXDIS	FUNCSEL		

位	描述	
[31:4]	**Reserved**	保留
[3]	**TXRXDIS**	**TX 和 RX 禁止位** 置位该位以禁止 TX 和 RX. 0 = TX 和 RX 使能 1 = TX 和 RX 禁止 **注:** 该位置位后，TX 和 RX 不会立刻关闭。TX 和 RX 在完成当前任务后再关闭。关闭 TX 和 RX 后，TXRXACT (UART_FIFOSTS[31]) 清 0 。
[2:0]	**FUNCSEL**	**功能选择** 000 = UART 功能 001 = LIN 功能 010 = IrDA 功能 011 = RS-485 功能 100 = UART 单线功能 其它 = 保留

UART LIN控制寄存器(UART_LINCTL)

寄存器	偏移量	R/W	描述	复位值
UART_LINCTL x=0,1	UARTx_BA+0x34	R/W	UART LIN控制寄存器	0x000C_0000

31	30	29	28	27	26	25	24
PID							
23	22	21	20	19	18	17	16
HSEL		BSL		BRKFL			
15	14	13	12	11	10	9	8
Reserved			BITERREN	LINRXOFF	BRKDETEN	IDPEN	SENDH
7	6	5	4	3	2	1	0
Reserved			MUTE	SLVDUEN	SLVAREN	SLVHDEN	SLVEN

位	描述	
[31:24]	**PID**	**LIN PID 位** 在 LIN 功能模式下，该域包含 LIN 帧 ID 值。帧 ID 的校验可以由软件或硬件产生，这取决于IDPEN (UART_LINCTL[9]) 的设置。 如果校验由硬件产生，用户填 ID0~ID5, (LIN_PID[24:29]，硬件会计算 P0 (PID[30]) 和 P1 (PID[31])，否则用户必须填帧 ID 和校验位 **注1:**用户可以填任何 8-bit 值到该域，位 24 表示 ID0 (LSB 优先) **注2:**该域可以用在 LIN 主机模式或从机模式
[23:22]	**HSEL**	**LIN 帧报头选择** 00 = LIN 帧报头包括 “间隔域” 01 = LIN 帧报头包括 “间隔域” 和 “同步域””. 10 = LIN 帧报头包括 “间隔域”, “同步域” 域“帧 ID 域”. 11 = 保留 **注:** 该位可以用于 LIN 主机发送帧头域 (SENDH (UART_LINCTL[8])) = 1) 或用于提示 LIN 从机从 mute 模式退出条件 (LIN_MUTE_EN (UA_LIN_CTL[4] = 1)。
[21:20]	**BSL**	**LIN 间隔域/同步域分隔符长度** 00 = LIN 间隔域/同步域分隔符长度为 1 bit 时间. 10 = LIN 间隔域/同步域分隔符长度为 2 bit 时间. 10 = LIN 间隔域/同步域分隔符长度为 3 bit 时间. 11 = LIN 间隔域/同步域分隔符长度为 4 bit 时间. **注:**该位用于 LIN 主机发送帧报头域
[19:16]	**BRKFL**	**LIN 间隔域长度** 该域表示一个4-bit LIN TX 间隔域数量 **注1:**这个寄存器是BRKFL (UART_ALTCTL[3:0]) 的映射寄存器，用户可以通过设置BRKFL (UART_ALTCTL[3:0]) 或BRKFL (UART_LINCTL[19:16])对间隔域的长度进行读/写。

		注 2: 该间隔域长度为BRKFL + 1 **注 3:**根据 LIN 规范，复位值是 12 (间隔域长度= 13).
[15:13]	**Reserved**	保留
[12]	**BITERREN**	**位错误侦测使能** 0 =位错误侦测功能禁止. 1 =位错误侦测使能. **注:**在 LIN 功能模式下，当位错误发生时，BITEF (UART_LINSTS[9]) 标志会被置位。如果LINIEN (UART_INTEN[8]) = 1，会产生一个中断
[11]	**LINRXOFF**	**LIN 接收禁止位** 如果接收使能(RXOFF (UART_LINCTL[11]) = 0)，所有接收到的数据字节会被接收并存储在RX-FIFO，如果接收禁止(RXOFF (UART_LINCTL[11] = 1)，接收到的所有数据会被忽略 0 = LIN 接收使能. 1 = LIN 接收禁止. **注:** 该位仅工作在 LIN 功能模式下有效(FUNCSEL (UART_FUNCSEL[1:0]) = 01)
[10]	**BRKDETEN**	**LIN 间隔域检测使能位** 当检测到显性位长度连续超过 11 位，后跟一个分隔符，在间隔域结束时BRKDETF (UART_LINSTS[8]) 标志被置位。如果LINIEN (UART_INTEN [8])=1，将会产生中断。 0 = LIN 间隔域检测禁止 1 = LIN 间隔域检测使能.
[9]	**IDPEN**	**LIN ID 校验使能位** 0 = LIN 帧 ID 校验禁止. 1 = LIN 帧 ID 校验使能. **注 1:**该位可以用于 LIN 主机发送帧头域(SENDH (UART_LINCTL[8])) = 1 且 HSEL (UART_LINCTL[23:22]) = 10) 或用来使能 LIN 从机对接收到的帧 ID 进行奇偶校验检查。 **注 2:**该位仅在发送器报头是在HSEL (UART_LINCTL[23:22]) = 10 时有用。
[8]	**SENDH**	**LIN TX 发送帧报头使能位** LIN TX 帧报头可以是“间隔域” 或 “间隔和同步域” 或 “间隔, 同步和帧 ID 域”，这取决于HSEL (UART_LINCTL[23:22]) 的设置 0 = LIN TX 帧报头发送禁止. 1 = LIN TX 帧报头发送使能. **注 1:**这些寄存器是SENDH (UART_ALTCTL [7]) 的影子寄存器;用户可以通过SENDH (UART_ALTCTL [7]) 或 SENDH (UART_LINCTL [8]) 进行读写设置 **注 2:**当发送器帧头域 (可能是 “间隔” 或 “间隔 + 同步” 或 “间隔 + 同步 + 帧 ID”通过HSEL (UART_LINCTL[23:22]) 选择) 传输操作完成, 该位会自动清 0.
[7:5]	**Reserved**	保留
[4]	**MUTE**	**LIN Mute 模式使能位** 0 = LIN Mute 模式禁止 1 = LIN Mute 模式使能. **注意:** 退出 Mute 模式条件，该域的控制和相互影响在章节6.15.5.9 (LIN 从机模式) 中有详细解释
[3]	**SLVDUEN**	**LIN 从机分频器更新方式使能位** 0 = UA_BAUD 被软件更新 (如果同时没有自动重同步发生) 1 = UA_BAUD 在下次接收到字符的时候更新。用户必须在接收校验之前设置该位.

		注1:该位仅在 LIN 从模式有效(SLVEN (UART_LINCTL[0]) = 1) **注 2**:该位用于 LIN 从机自动重同步模式. (对于非自动重同步模式，该位需维持清零) **注 3:**该域的控制和相互作用在章节6.15.5.9 中解释 (带自动重同步的从机模式)
[2]	**SLVAREN**	**LIN 从机自动重同步模式使能位** 0 = LIN 从机自动重同步模式禁止 1 = LIN 从机自动重同步模式使能. **注意 1:** 该位仅在 LIN 从机模式(SLVEN (UART_LINCTL[0]) = 1)有效. **注意2:**当工作在自动重同步模式，波特率的设置必须是模式2 (BAUD_M1 (UA_BAUD [29]) 和 BAUD_M0 (UA_BAUD [28]) 必须是 1). **注意 3:**该域的控制和相互作用在章节6.15.5.9 (带自动重同步的从机模式) 中解释
[1]	**SLVHDEN**	**LIN 从机报头侦测使能位** 0 = LIN 从机报头检测禁止. 1 = LIN 从机报头检测使能. **注1:**该位仅在 LIN 从机模式 (SLVEN (UART_LINCTL[0]) = 1)有效. **注 2:** 在 LIN 功能模式，当侦测报头域(间隔域+ 同步域+ 帧 ID)，SLVHDETF (UART_LINSTS [0]) 标志会被置位，如果LINIEN (UART_INTEN[8]) = 1，会产生一个中断
[0]	**SLVEN**	**LIN 从模式使能位** 0 = LIN 从模式禁止 1 = LIN 从模式使能

UART LIN状态寄存器(UART_LINSTS)

寄存器	偏移量	R/W	描述	复位值
UART_LINSTS x=0,1	UARTx_BA+0x38	R/W	UART LIN状态寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved						BITEF	BRKDETF
7	6	5	4	3	2	1	0
Reserved				SLVSYNCF	SLVIDPEF	SLVHEF	SLVHDETF

位	描述	
[31:10]	**Reserved**	保留
[9]	**BITEF**	**位错误侦测状态标志 (只读)**. TX 传输状态，硬件会监视总线状态，如果输入管脚 (UART_RXD) 状态不等于输出管脚 ((UART_TXD) 状态, BITEF (UART_LINSTS[9]) 位会被置位。 当发生位错误，如果LINIEN (UART_INTEN[8]) = 1，会产生一个中断 0 = 未检测到位错误 1 = 检测到位错误 **注 1:**向该位写 1 清 0 **注2:** 该位仅当位错误侦测功能使能时有效(BITERREN (UART_LINCTL [12]) = 1)
[8]	**BRKDETF**	**LIN 间隔域检测标志 (只读)** 当侦测到一个间隔域，该位由硬件置位。可通过软件写 1 将该位清零。 0 = LIN 间隔域没有被检测到 1 = LIN 间隔域被检测到. **注 1:**向该位写 1 清 0. **注 2:**该位仅当 LIN 间隔域侦测功能使能时有效(BRKDETEN (UART_LINCTL[10]) =1)
[7:4]	**Reserved**	保留
[3]	**SLVSYNCF**	**LIN 从机同步域. (只读)** 该位表示在自动重同步模式， LIN 同步域正在被分析。当接收器报头侦测到一些错误，用户必须通过写 1 到该位复位内部电路来重新搜索新的帧报头 0 =当前字符不在 LIN 同步状态. 1 =当前字符在 LIN 同步状态. **注 1：**该位仅在 LIN 从机模式有效 (SLVEN(UART_LINCTL[0]) = 1) **注 2：**向该位写 1 清 0

		注 3:当向该位写 1，硬件会重载初始波特率并重新搜索新的帧报头
[2]	**SLVIDPEF**	**LIN 从机 ID 校验错误标志 (只读)** 当接收帧 ID 校验错误，该位由硬件置 1 0 = 无效. 1 = 收到的帧 ID 校验错误. **注 1:**向该位写 1 清 0. **注 2:**该位仅在 LIN 从机模式 (SLVEN (UART_LINCTL [0])= 1)，并使能LIN 帧ID 校验功能 IDPEN (UART_LINCTL [9]) 时有效
[1]	**SLVHEF**	**LIN 从机报头错误标志 (只读)** 在 LIN 从机模式，当侦测到一个 LIN 报头错误时，该位由硬件置 1，向该位写 1 清 0。报头错误包括“break 间隔符太短 (小于 0.5 位的时间)”， “在同步域或在识别域中帧错误，” “在非自动重同步模式同步域数据不是 0x55”， “自动重同步模式同步域偏离错误”， “自动重同步模式同步域测量超时”和“LIN 报头接收超时’ 0 = LIN 未检测到报头错误 1 = LIN 检测到报头错误. **注 1:**向该位写 1 清 0 **注2:**该位仅在 LIN 从机模式(SLVEN (UART_LINCTL [0]) = 1)，并且使能LIN 从机报头侦测功能(SLVHDEN (UART_LINCTL [1])) 时有效。
[0]	**SLVHDETF**	**LIN 从机报头侦测标志 (只读)** 在 LIN 从机模式，当侦测到一个 LIN 报头，该位由硬件置位，通过软件写 1 清 0 0 = LIN 报头没有被侦测到. 1 = LIN 报头被侦测到 (分隔+ 同步+ 帧 ID) **注 1:**向该位写 1 清 0 **注 2:**该位仅在 LIN 从机模式(SLVEN (UART_LINCTL [0]) = 1)，并且使能LIN 从机报头侦测功能(SLVHDEN (UART_LINCTL [1])) 时有效。 **注3:**当使能ID 校验IDPEN (UART_LINCTL [9])，如果硬件侦测到完整的报头(“分隔+ 同步+ 帧ID”)，SLVHDETF 会被置位，无论帧 ID 是否正确

UART波特率补偿寄存器(UART_BRCOMP)

寄存器	偏移量	R/W	描述	复位值
UART_BRCOMP x=0,1,2,3,4,5,6,7	UARTx_BA+0x3C	R/W	UART波特率补偿寄存器	0x0000_0000

31	30	29	28	27	26	25	24
BRCOMPDEC	Reserved						
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							BRCOMP
7	6	5	4	3	2	1	0
BRCOMP							

位	描述	
[31]	**BRCOMPDEC**	**波特率补偿方向** 0 = 对应位正向 (加 1 个模块时钟) 补偿 1 = 对应位负向 (减 1 个模块时钟) 补偿.
[30:9]	**Reserved**	保留
[8:0]	**BRCOMP**	**波特率补偿模式** 这 9 位用定义相关的位是否被补偿 BRCOMP[7:0] 用来定义 UART_DAT[7:0] 的补偿，BRCOM[8] 则是校验位

UART唤醒控制寄存器(UART_WKCTL)

寄存器	偏移量	R/W	描述	复位值
UART_WKCTL **x=0,1,2,3,4,5,6,7**	UARTx_BA+0x40	R/W	UART唤醒控制寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved			**WKTOUTEN**	**WKRS485EN**	**WKRFRTEN**	**WKDATEN**	**WKCTSEN**

位	描述	
[31:5]	**Reserved**	保留
[4]	**WKTOUTEN**	**数据接收 FIFO 达到阈值超时唤醒系统功能使能位** 0 =数据接收 FIFO 达到阈值超时唤醒系统功能禁止 1 = 数据接收 FIFO 达到阈值超时唤醒系统功能使能, 数据接收 FIFO 达到阈值超时唤醒系统功能可以将处于掉电模式的系统唤醒 **注:** WKRFRTEN (UART_WKCTL[2]) 置 1 时，建议使能该位.
[3]	**WKRS485EN**	**RS-485 地址匹配 (AAD 模式) 唤醒使能位** 0 = RS-485 地址匹配 (AAD 模式) 唤醒系统功能禁止. 1 = RS-485 地址匹配 (AAD 模式) 唤醒系统功能使能, RS-485 地址匹配 (AAD 模式) 唤醒系统功能可以将处于掉电模式的系统唤醒 **注:** 该位在 RS-485 自动地址检测 (AAD 模式)，ADDRDEN (UART_ALTCTL[15]) 置 1 时使用
[2]	**WKRFRTEN**	**数据接收 FIFO 达到阈值唤醒使能位** 0 = 数据接收 FIFO 达到阈值唤醒系统功能禁止. 1 = 数据接收 FIFO 达到阈值唤醒系统功能使能，数据接收 FIFO 达到阈值唤醒系统功能可以将处于掉电模式的系统唤醒
[1]	**WKDATEN**	**输入数据唤醒使能位** 0 = 输入数据唤醒系统功能禁止. 1 = 输入数据唤醒系统功能使能，输入数据唤醒系统功能可以将处于掉电模式的系统唤醒
[0]	**WKCTSEN**	**nCTS 唤醒使能位** 0 = nCTS 唤醒系统功能禁止 1 = nCTS 唤醒系统功能使能，唤醒系统功能可以将处于掉电模式的系统唤醒

UART唤醒状态寄存器(UART_WKSTS)

寄存器	偏移量	R/W	描述	复位值
UART_WKSTS **x=0,1,2,3,4,5,6,7**	UARTx_BA+0x44	R/W	UART唤醒状态寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved			**TOUTWKF**	**RS485WKF**	**RFRTWKF**	**DATWKF**	**CTSWKF**

位	描述	
[31:5]	**Reserved**	保留
[4]	**TOUTWKF**	**接收数据 FIFO 阈值超时唤醒标志** 掉电的芯片被接收数据 FIFO 阈值超时唤醒时，该位置位 0 = 芯片处于掉电模式. 1 = 掉电的芯片被接收数据 FIFO 阈值超时唤醒. **注 1:** WKTOUTEN (UART_WKCTL[4]) 使能, 接收数据 FIFO 阈值超时唤醒导致该位置 1 . **注 2:** 该位写 1 清 0.
[3]	**RS485WKF**	**RS-485 地址匹配 (AAD 模式) 唤醒标志** 掉电的芯片被 RS-485 地址匹配 (AAD 模式) 唤醒时，该位置位 0 = 芯片处于掉电模式. 1 = 掉电的芯片被 RS-485 地址匹配 (AAD 模式) 唤醒 **注 1:** WKRS485EN (UART_WKCTL[3]) 使能, RS-485 地址匹配 (AAD 模式) 唤醒导致该位置 1 **注 2:** 该位写 1 清 0.
[2]	**RFRTWKF**	**接收数据 FIFO 达到阈值唤醒标志** 掉电的芯片被接收数据 FIFO 达到阈值唤醒时，该位置位 0 = 芯片处于掉电模式. 1 = 掉电的芯片被接收数据 FIFO 达到阈值唤醒 **注 1:** WKRFRTEN (UART_WKCTL[2]) 使能, 接收数据 FIFO 达到阈值唤醒导致该位置 1 **注 2:** 该位写 1 清 0.
[1]	**DATWKF**	**输入数据唤醒标志** 掉电的芯片被输入数据唤醒时，该位置位.

		0 = 芯片处于掉电模式. 1 = 掉电的芯片被输入数据唤醒. **注 1:** WKDATEN (UART_WKCTL[1]) 使能，输入数据唤醒导致该位置 1 **注 2:**该位写 1 清 0.
[0]	**CTSWKF**	**nCTS 唤醒标志** 掉电的芯片被nCTS唤醒时，该位置 1 . 0 = 芯片处于掉电模式. 1 = 掉电的芯片被 nCTS 唤醒. **注 1:** WKCTSEN (UART_WKCTL[0]) 使能，nCTS 唤醒导致该位置 1 **注 2:**该位写 1 清 0.

UART输入数据唤醒补偿寄存器(UART_DWKCOMP)

寄存器	偏移量	R/W	描述	复位值
UART_DWKCOMP x=0,1,2,3,4,5,6,7	UARTx_BA+0x48	R/W	UART输入数据唤醒补偿寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
STCOMP							
7	6	5	4	3	2	1	0
STCOMP							

位	描述	
[31:16]	**Reserved**	保留
[15:0]	**STCOMP**	**开始位补偿值** 这些位代表在芯片从掉电模式唤醒到 UART 控制器可以接收第一个开始位需要的 UART_CLK 数目 **注:** 仅在 WKDATEN (UART_WKCTL[1]) 置位时有效.

6.17 以太网控制器(EMAC)

6.17.1 概述

M480为网络应用提供了一组以太网MAC控制器(EMAC)。以太网控制器包括：用于识别以太网MAC地址的内置CAM功能的IEEE 802.3/以太网协议引擎、发送缓存、接收缓存、收发状态机控制器、符合IEEE 1588的时间戳、魔术包解析引擎和状态控制器。支持MII和RMII(简化的MII)接口连接外部PHY。

6.17.2 特性

- 符合IEEE Std. 802.3 CSMA/CD 协议
- 时间戳符合IEEE Std. 1588 – 2002 协议
- 支持10Mbps或100Mbps全双工或半双工
- 支持RMII接口
- MII管理功能可控制外接以太网物理层芯片
- 流控支持暂停和远程暂停功能
- 支持长帧(大于1518字节)和短帧(小于64字节)接收
- 支持16个CAM条目功能识别MAC地址
- 支持魔术包唤醒处于掉电状态的系统
- 收和发各256字节缓存
- 支持DMA功能

6.17.3 框图

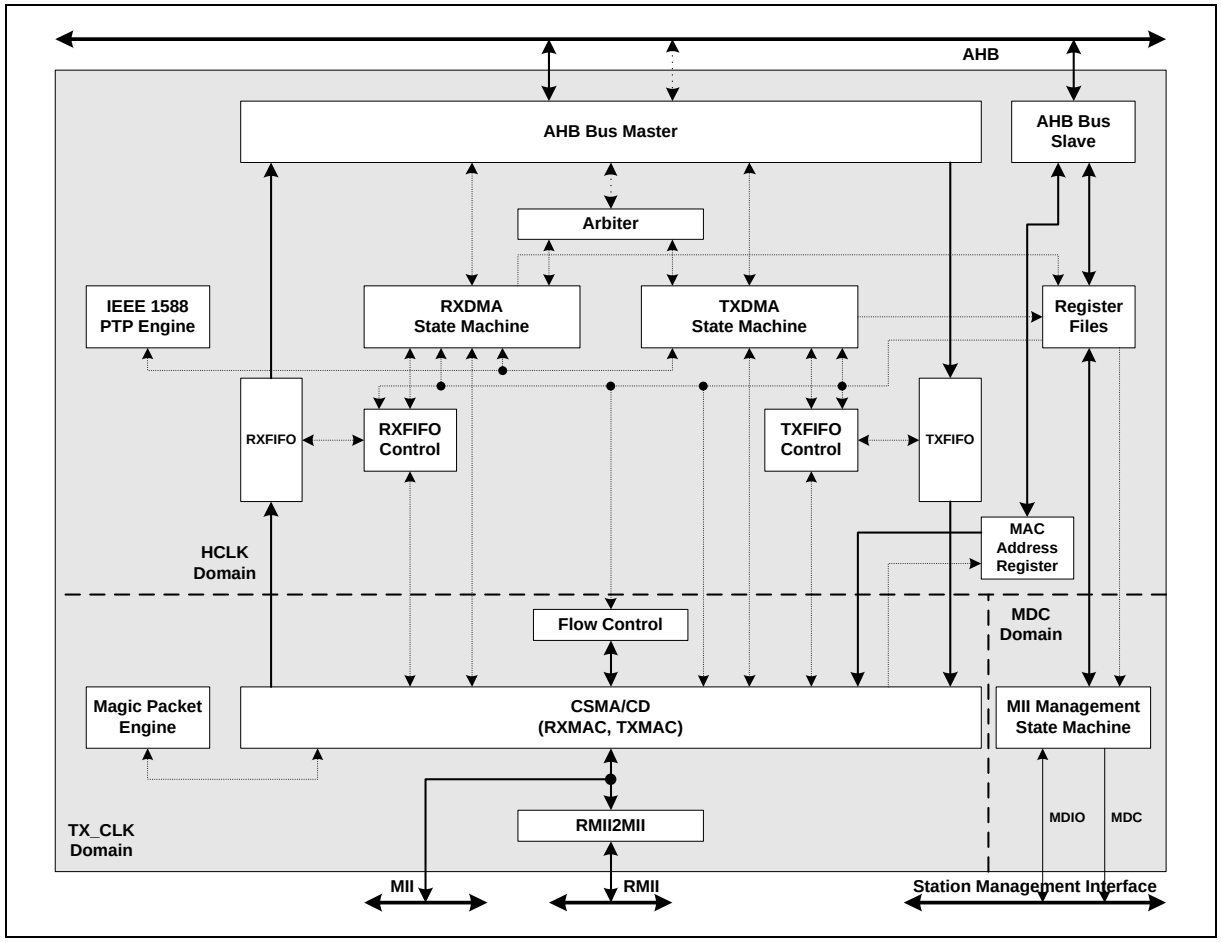


图 6.17-1EMAC 框图

6.17.4 基本配置

6.17.4.1 EMAC基本配置

- 时钟源配置
  - 在寄存器EMACDIV(CLK_CLKDIV3[21:16])选择MDC的时钟分频数
  - 在寄存器EMACCKEN(CLK_AHBCLK[5])使能EMAC时钟
- 复位配置
  - 在寄存器EMACRST(SYS_IPRST0[5])复位EMAC控制器
- 管脚配置

组	管脚名	GPIO	MFP
EMAC	EMAC_RMII_MDC	PB.11, PE.8	MFP3
	EMAC_RMII_MDIO	PB.10, PE.9	MFP3
	EMAC_RMII_RXD0	PC.7	MFP3

		PB.4	MFP4
	EMAC_RMII_RXD1	PC.6	MFP3
		PB.3	MFP4
	EMAC_RMII_CRSDV	PA.7	MFP3
		PB.2	MFP4
	EMAC_RMII_RXERR	PA.6	MFP3
		PB.1	MFP4
	EMAC_RMII_TXD0	PB.9, PE.10	MFP3
	EMAC_RMII_TXD1	PB.8, PE.11	MFP3
	EMAC_RMII_TXEN	PB.7, PE.12	MFP3
	EMAC_PPS	PB.6, PE.13	MFP3
	EMAC_RMII_REFCLK	PC.8	MFP3
		PB.5	MFP4

6.17.5 功能描述

6.17.5.1 仲裁器

在EMAC上，有RXREQ和TXREQ两种总线请求，由仲裁器在这两种请求间进行仲裁决定哪一个请求AHB总线，表6.17-1是仲裁结果示意：

RXREQ	TXREQ	授权
0	0	2个都不授权
0	1	授权TXDMA.
1	0	授权RXDMA
1	1	若TXFIFO有效字节少于RXFIFO空余字节数,授权TXDMA.
1	1	若RXFIFO空余字节数少于等于TXFIFO有效空间，授权RXDAM

表 6.17-1 仲裁器仲裁结果

6.17.5.2 TXDMA状态机

TXDMA状态机从内存读取数据通过AHB传输到256字节的缓存。首先，TXDMA会请求发送MAC把数据发送出去。传输时，TXDMA先获取发送描述符。按发送描述符里定义的缓存地址，TXDMA从系统获取帧数据并存储在内部256字节的发送FIFO里。之后，发送MAC会从发送FIFO读取帧数据并发送出去。完成传输后，TXDMA更新当前帧的发送状态，再把描述符回写到系统内存，以示发送结束。

6.17.5.3 RXDMA状态机

RXDMA状态机从内部的256字节接收FIFO读取数据通过AHB总线传输到系统内存。接收过程中，RXDMA先获取接收描述符。RXDMA从接收描述符里定义的缓存地址读出输入帧要在系统中存储的地址。当MAC指示收到数据后，RXDMA从256字节的接收缓存读取数据并存入系统内存。传输结束后，RXDMA更新当前帧的接收状态，再把描述符回写到系统内存，以示数据已存入指定内存。

6.17.5.4 流控

这里指的是EMAC工作在全双工模式下的流控功能。流控功能在IEEE 802.3 第 31章中有定义。在IEEE 802.3 Std里定义的流控帧仅指同一时间下的暂停帧。控制帧的收发可以通过控制寄存器配置。

控制寄存器EMAC_CTL的位ACP置1，就开始接收控制帧。若收到暂停帧，当前发送帧发送结束后，就暂停发送。

发送控制帧的流程是：先把控制帧的MAC目的地址写入寄存器{EMAC_CAM13M,EMAC_CAM13L},源MAC地址写入寄存器{EMAC_CAM14M,EMAC_CAM14L}，控制帧的长度，操作码和操作写入寄存器{EMAC_CAM15MSB,EMAC_CAM15LSB},然后置1位SDPZ(EMAC_CTL[16])就会发送。发送结束后，位SDPZ(EMAC_CTL[16])自动清0。

6.17.5.5 MII管理状态机

MII管理功能符合IEEE 802.3标准。通过此接口，软件可以读取外部PHY芯片的控制和状态寄存器。MII管理功能提供2个可编程的寄存器EMAC_MIIMDAT(MAC MII管理数据寄存器)和EMAC_MIIMCTL(MAC MII管理数据控制和地址寄存器)。置1位BUSY(EMAC_MIIMCTL[17])将触发MII管理状态机。管理周期结束后，BUSY位自动清0。

6.17.5.6 MAC

EMAC功能完全符合IEEE802.3 规范。如图6.17-2是帧结构和传输操作的示例。

数据帧从发送DMA到MII总线之前，先经MAC包装：加上前导码，分界符SFD，帧校验序列，数据若不够64字节后面填充0，最后再加上CRC码。包装后的帧格式如下所示：

110101010 --- 10101010	10101011	d0	d1	d2	--	dn	Padding	CRC31	CRC30	---	CRC0

图 6.17-2 以太网帧格式

如前面所提到的格式，前导码是一组连续的7个“10101010”，SFD是1个字节的”10101011”。如果数据帧不足64位会在填充数据里补0。发送描述符里的P位用来控制数据帧不足64位时是否需要填充0。如果禁止了填充功能，则不会把填充数据附在数据帧之后。CRC0…CRC32是32位的CRC校验序列。CRC码符合IEEE802.3的标准。如果发送描述符里的CRC禁止功能置高，则不会在数据位后附32位的CRC序列。

MAC 还实现了其他IEEE802.3定义的功能如，帧间距功能，冲突检测，冲突执行，冲突回退和重发送。冲突回退时间与时间片有关，512位时间。从当前发送尝试到下一次发送尝试之间的时间片延时符合IEEE802.3规定的正态分布随机算法。MAC按IEEE802.3标准实现了接收相关的功能：地址识别，帧完整性检验，帧解析和防冲突检测。

6.17.5.7 IEEE 1588标准的时间戳引擎

EMAC时间戳符合IEEE标准1588。时间戳引擎里，64位计数器产生参考时间，详见寄存器EMAC_TSSEC和ETSLSR。

发送帧，若TSEN(EMAC_TSCTL[0])和寄存器TXDES0的位TTSEN都为1，帧发送结束后，64位的时间值将写入TXDES1和TXDES2。

接收帧，若TSEN(EMAC_TSCTL[0])为1，接收结束后，64位的时间值将写入RXDES1和RXDES3。

图6.17-2描述了64位计数器如何产生参考时间。

2个32位计数器EMAC_TSSEC和EMAC_TSSUBSEC组成一个64位计数器，对HCLK进行计数产生时间戳。两种计数方式由TSMODE(EMAC_TSCTL[3])控制，EMAC_TSSUBSEC每次增加EMAC_TSINC值。TSMODE(EMAC_TSCTL[3])=0时，TSLSR每个时钟都增加，TSMODE(EMAC_TSCTL[3])=1时，

TSLSR仅在累加器溢出时增加。

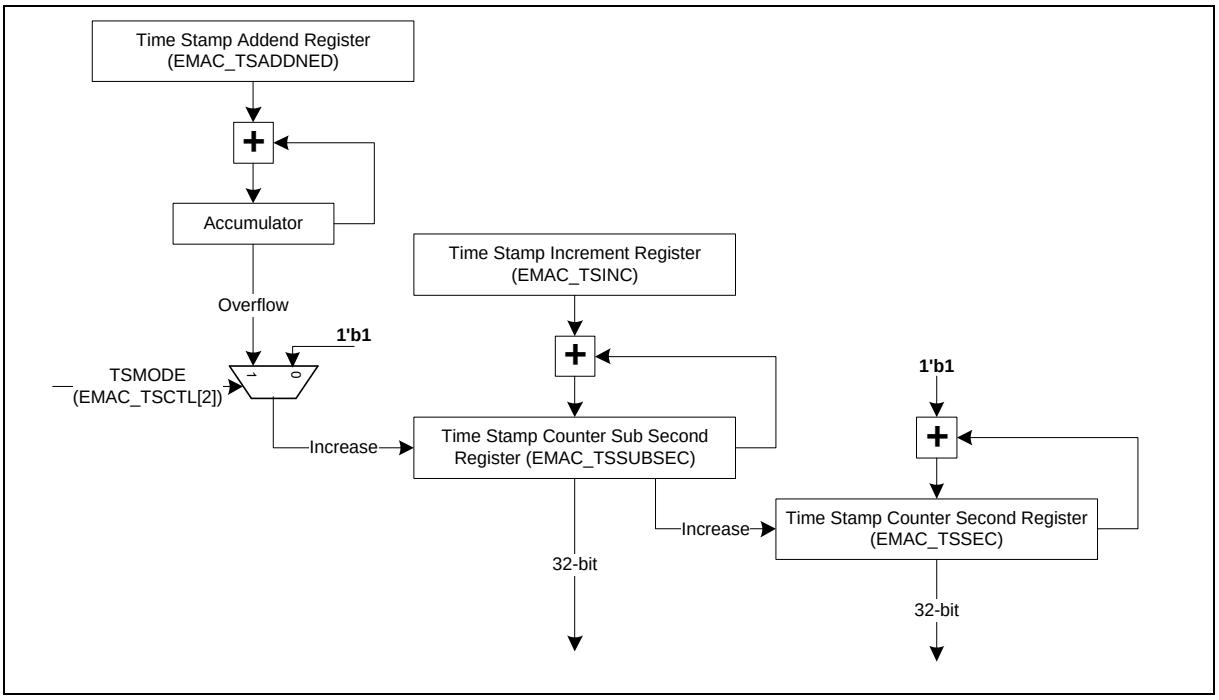


表 6.17-264-位参考时间计数器

6.17.5.8 魔术包解析引擎

EMAC支持魔术包解析。魔术包是一种广播包。它是紧跟在重复16次的48位MAC地址(EMAC_CAM0M和EMAC_CAM0L)之后的6个字节的0xFF。

WOLEN(EMAC_CTL[6])控制着魔术包引擎的开关。WOLEN(EMAC_CTL[6])=1时，收到魔术包，位WOLIF(EMAC_INTSTS[15])置1，同时唤醒芯片。若寄存器MIEN的位WOLIEN=1，则会申请中断。

6.17.6 DMA描述符数据结构

描述符是用来保存每帧的控制、状态和数据信息的链表结构。CPU和EMAC通过描述符交换帧传输的信息。描述符有两种：RXDMA描述符用与帧接收和TXDMA描述符用与帧发送。每个描述符有4个字，包含每帧的控制，状态和数据信息，他们的数据结构参见6.17.6.1 RXDMA数据结构描述和6.17.6.2 TXDMA数据结构描述。

6.17.6.1 RXDMA描述符数据结构

RXDMA描述符包含4个32位字。如图6.17-3是RXDMA描述符的数据结构。

	31		15	0
RXDES 0	OWN	Receive Frame Status		Receive Frame Byte Count
RXDES 1	Receive Frame Buffer Starting Address / Time Stamp Sub Second			
RXDES 2	Reserved			
RXDES 3	Next RxDMA Descriptor Starting Address / Time Stamp Second			

图 6.17-3 RXDMA 描述符数据结构

RXDES 0: RXDMA描述符字0

RXDMA描述符字0包含一个描述符所有者标识，接收帧状态和接收帧字节数。详细描述如下：

31	30	29	28	27	26	25	24
Owner	Reserved						
23	22	21	20	19	18	17	16
RTSAS	RPIF	ALIEIF	RXGDIF	LPIF	Reserved	CRCEIF	RXIF
15	14	13	12	11	10	9	8
RBC							
7	6	5	4	3	2	1	0
RBC							

位	字段	描述
[31]	Owner	**所有者** 该寄存器位表示RX描述符的归属，是CPU还是EMAC。只有描述符的所有者才能修改RX描述符，其他只能读描述符。 O = 1'b1表示EMAC RXDMA是RX描述符的所有者，帧接收的RX描述符可用。帧接收完成后，EMAC RXDMA修改该位的归属为1'b0 O = 1'b0表示CPU是RX描述符的所有者。CPU帧处理完成后，修改该位的归属为1'b1,，EMAC RXDMA获得RX描述符的所有权 0=所有者为CPU. 1=所有者为EMAC.
[30:24]	Reserved	保留
[23]	RTSAS	**RX时间戳状态** 该位表示输入帧成功获得时间戳。当该位置位时，输入帧接收完成后，RX描述符字1和RX描述符字3保存时间戳的值。 0=RX描述符字1和RX描述符字3未保存时间戳 1=RX描述符字1和RX描述符字3已保存时间戳
[22]	RPIF	**短包** RPIF表示RX描述符指向的数据缓存地址里的帧是个短帧(帧长度小于64字节) 0=非短帧. 1=是短帧.
[21]	ALIEIF	**对齐错误** ALIEIF表示RX描述符指向的数据缓存地址里的帧长度不是整字节的 0=帧长度是整字节 1=帧长度非整字节
[20]	RXGDIF	**帧接收完成** RXGDIF表示帧接收已完成且存入了RX描述符指向的数据缓存地址里 0=帧接收未完成 1=帧接收已完成.

[19]	LPIF	**长包中断标识** LPIF表示RX描述符指向的数据缓存地址里的帧是长帧(帧长度大于1518) 0=非长帧. 1=是长帧.
[18]	Reserved	保留
[17]	CRCEIF	**CRC错误** CRCEIF表示RX描述符指向的数据缓存地址里的帧CRC错误 0=CRC正确. 1=CRC错误.
[16]	RXIF	**接收中断** RXIF表示RX描述符指向的数据缓存地址里的帧产生了一个中断 0=无中断. 1=有中断.
[15:0]	RBC	**接收字节计数** RBC表示RX描述符指向的数据缓存地址里的帧的数目。包含4字节的CRC。若使能了STRIPCRC(EMAC_CTL[5])，计算时CRC字节将不包含在内。

RXDES 1: RXDMA描述符字1

RXDMA 1包含了帧接收缓存的开地址或者时间戳的低32位。详情如下：

31	30	29	28	27	26	25	24
RXBSA/TSSUBSEC							
23	22	21	20	19	18	17	16
RXBSA/TSSUBSEC							
15	14	13	12	11	10	9	8
RXBSA/TSSUBSEC							
7	6	5	4	3	2	1	0
RXBSA/TSSUBSEC							

位	字段	描述
[31:0]	RXBSA	**接收缓存的起始地址** RXBSA接收缓存的起始地址.

表 6.17-3RXDMA 描述符字 1(TSEN(EMAC_TSCTL[0])为 0)

位	字段	描述
[31:0]	TSSUBSEC	**时间戳亚秒** 使能TSEN(EMAC_TSCTL[0])，以太网MAC控制器写RX描述符到系统内存时，会存储时间戳低32位(ETSLSR)到该字段

表 6.17-4RXDMA 描述符字 1(TSEN(EMAC_TSCTL[0])为 1)

RXDES 2: RXDMA描述符字 2

RXDMA描述符字2暂时保留

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved							

位	字段	描述
[31:0]	Reserved	保留

表 6.17-5RXDMA 描述符字 2

RXDES 3: RXDMA描述符字3

RXDMA描述符字3包含下一个RXDMA描述符的起始地址或时间戳的高32位。详情如下：

31	30	29	28	27	26	25	24
NRXDSA/TSSEC							
23	22	21	20	19	18	17	16
NRXDSA/TSSEC							
15	14	13	12	11	10	9	8
NRXDSA/TSSEC							
7	6	5	4	3	2	1	0
NRXDSA/TSSEC							

位	字段	描述
[31:0]	NRXDSA	**下一个RX描述符的起始地址** NRXDSA为下一个RX描述符的起始地址当以太网MAC控制器获取下一个RX描述符时，它会忽略NRXDSA[1:0]

表 6.17-6RXDMA 描述符字 3(TSEN(EMAC_TSCTL[0])为 0)

位	字段	描述
[31:0]	TSSEC	**时间戳秒** 使能TSEN(EMAC_TSCTL[0])以太网MAC控制器写RX描述符到系统内存时，会存储时间戳高32位(ETSMSR)到该字段

表 6.17-7RXDMA 描述符字 3(TSEN(EMAC_TSCTL[0])为 1)

6.17.6.2 TXDMA描述符数据结构

TXDMA描述符包含4个32位字。TXDMA描述符数据结构如图6.17-4所示：

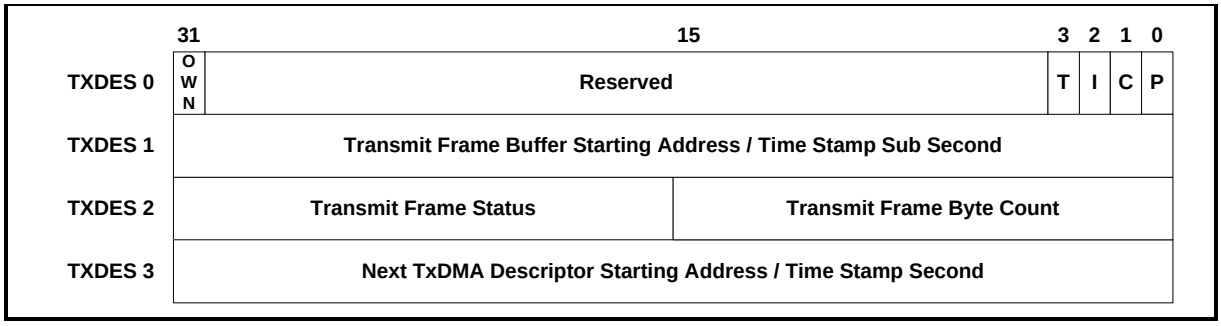


图 6.17-4 TXDMA 描述符数据结构

TXDES 0: TXDMA描述符字0

TXDMA描述符字0包含一个描述符所有者标识。另外，它还包含了发送帧填充控制位，CRC扩充，中断使能和时间戳控制。详情如下：

31	30	29	28	27	26	25	24
Owner	Reserved						
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved				TTSEN	INTEN	CRCAPP	PADEN

位	字段	描述
[31]	Owner	**所有者** 该寄存器位表示TX描述符的归属，是CPU还是EMAC。只有描述符的所有者才能修改TX描述符，其他只能读描述符。 O = 1’b1 表示EMAC TXDMA是TX描述符的所有者，帧发送的TX描述符可用。帧发送完成后，EMACRXDMA修改该位的归属为1’b0，CPU获得TX描述符所有权 O = 1’b0表示CPU是TX描述符的所有者。CPU准备好新帧等待发送，修改该位的归属为1’b1,，EMAC RXDMA获得TX描述符的所有权 0=所有者为CPU. 1=所有者为EMAC.
[30:4]	Reserved	保留
[3]	TTSEN	**TX时间戳使能位** 该位置1且IEEE1588 PTP功能使能，内置时间戳电路会在发送完成帧的SFD后给该帧打上时间戳 0=IEEE 1588时间戳功能禁止. 1=IEEE 1588时间戳功能使能.
[2]	INTEN	**发送中断使能位** 帧传送完成后，INTEN控制着是否触发中断。如果使能了INTEN，帧传送完成后EMAC会触发中断，未使能则不会触发。 0=帧发送中断屏蔽. 1=帧发送中断使能
[1]	CRCAPP	**CRC扩充** CRCAPP控制着是否在帧发送时附加CRC，如果CRCAPP使能，4字节的CRC校验和会被附加到发送帧的结尾。 0=扩充4-字节CRC禁止. 1=扩充4-字节CRC使能.

[0]	PADEN	**填充使能位** PADEN控制是否在帧长度小于60字节时附加填充位。如果PADEN使能，EMAC会自动进行填充。 0=禁止附加填充位. 1=使能附加填充位.

表 6.17-8 TXDMA 描述符字 0

TXDES 1: TXDMA描述符字1

TXDMA描述符字1包括，发送帧缓存起始地址或时间戳低32位。详情如下：

31	30	29	28	27	26	25	24
TXBSA/TSSUBSEC							
23	22	21	20	19	18	17	16
TXBSA/TSSUBSEC							
15	14	13	12	11	10	9	8
TXBSA/TSSUBSEC							
7	6	5	4	3	2	1	0
TXBSA/TSSUBSEC							

位	字段	描述
[31:2]	TXBSA	传输缓存开始地址 TXBSA是传输包保存数据的开始地址

表 6.17-9 TXDMA 描述符字 1(TSEN(EMAC_TSCTL[0])是 0)

位	字段	描述
[31:0]	TSSUBSEC	**时间戳亚秒** 使能TSEN(EMAC_TSCTL[0])，且TX描述符字0的TTSEN使能，以太网MAC控制器写TX描述符到系统内存时，会存储时间戳低32位(EMAC_TSSUBSEC)到该字段。

表 6.17-10 TXDMA 描述符字 1(TSEN(EMAC_TSCTL[0])是 1)

TXDES 2: TXDMA描述符字2

TXDMA描述符字2包含发送帧状态和发送帧字节数，详情如下：

31	30	29	28	27	26	25	24
COLCNT				TTSAS	SQE	TXPAUSED	TXHALT
23	22	21	20	19	18	17	16
LCIF	TXABTIF	NCSIF	EXDEFIF	TXCPIF	Reserved	DEF	TXIF
15	14	13	12	11	10	9	8
TBC							
7	6	5	4	3	2	1	0
TBC							

位	字段	描述
[31:28]	COLCNT	**冲突数** COLCNT表示在包传递过程中发生的连续冲突数。如果传输时连续冲突数达16个，COLCNT归0x0，TXABTIF位置位
[27]	TTSAS	**TX时间戳状态** 该位表示帧成功获得时间戳。该位置高时，TX描述符字1和TX描述符字3在帧的SFD发送后保存时间戳的值 0=TX描述符字1和TX描述符字3未保存时间戳的值 1=TX描述符字1和TX描述符字3已保存时间戳的值
[26]	SQE	**SQE错误** SQE表示在10Mbps半双工模式下包传送结尾发现SQE错误。EMAC工作在10Mbps半双工模式下时，SQECHKEN(EMAC_CTL[17])使能，则会进行SQE错误检测。 0=包发送结尾未发现SQE错误. 1=包发送结尾发现SQE错误.
[25]	TXPAUSED	**发送暂停** 如果EMAC接收到暂停帧，或软件置位SDPZ(EMAC_MCM[16])且EMAC发送了一个暂停帧出去，TXPAUSED表示下个正常的包发送过程会暂停。 0=下一个正常的包发送过程正常 1=下一个正常的包发送过程暂停
[24]	TXHALT	**发送停止** TXON(EMAC_CTL[8])位由软件关闭，下一个正常的包发送过程停止 0=下一个正常的包发送过程正常 1=下一个正常的包发送过程停止
[23]	LCIF	**迟到的冲突** LCIF表示在64字节的冲突窗口外发生了冲突。即在64字节长度的帧发送出去后仍然有冲突存在。迟到的冲突检测仅在EMAC半双工模式下有效。 0=在64字节的冲突窗口外未发生冲突 1=在64字节的冲突窗口外发生了冲突

[22]	TXABTIF	**发送取消** TXABTIF表示在发送过程中由于连续出现了16次冲突而取消了包发送。发送取消功能仅在EMAC半双工模式下有效。 0=在发送过程中没有连续出现16次冲突 1=在发送过程中出现了连续16次冲突
[21]	NCSIF	**无载波监听** NCSIF表示MIII/F信号CRS在包传送开始无效。NCSIF仅在EMAC半双工模式下有效 0=MIII/F信号CRS在包传送开始无效 1=CRS信号正常.
[20]	EXDEFIF	**延期超时** EXDEFIF表示在100Mbps时，发送时帧等待发送超过了0.32768ms.或者在10Mbps时，等待超过3.2768ms.延期超时检测仅在NODEF(EMAC_CTL[9])位关闭，EMAC工作在半双工模式下有效。 0=未发生延期超时 1=在100Mbps时，发送时帧等待发送超过了0.32768ms.或者在10Mbps时，等待超过3.2768ms.
[19]	TXCPIF	**发送完成** TXCPIF表示发送正确完成 0=发送未完成. 1=发送已完成
[18]	Reserved	保留
[17]	DEF	**发送延期** DEF表示发送延期一次。DEF检测仅在半双工模式下有效 0=包发送未延期 1=包发送延期
[16]	TXIF	**发送中断** TXIF表示包发送会触发中断 0=包发送未触发中断 1=包发送会触发中断
[15:0]	TBC	**发送字节数** TBC表示TX描述符指向的数据缓存地址里的帧的数目

表 6.17-11TXDMA 描述符字 2

TXDES3: TXDMA 描述符字3

TXDMA描述符字3包含下一个TXDMA描述符的起始地址或时间戳的高32位。详情如下：

31	30	29	28	27	26	25	24
NTXDSA/TSSEC							
23	22	21	20	19	18	17	16
NTXDSA/TSSEC							
15	14	13	12	11	10	9	8
NTXDSA/TSSEC							
7	6	5	4	3	2	1	0
NTXDSA/TSSEC							

位	字段	描述
[31:0]	NTXDSA	**下一个TX描述符的起始地址** NTXDSA为下一个TX描述符的起始地址当以太网MAC控制器获取下一个TX描述符时，它会忽略NTXDSA[1:0]

表 6.17-12TXDMA 描述符字 3(TSEN(EMAC_TSCTL[0])为 0)

位	字段	描述
[31:0]	TSSEC	**时间戳秒** 使能TSEN(EMAC_TSCTL[0])，且TX描述符字0的TTSEN使能，以太网MAC控制器写TX描述符到系统内存时，会存储时间戳高32位(EMAC_TSSEC)到该字段

表 6.17-13 TXDMA 描述符字 3(TSEN(EMAC_TSCTL[0])为 1)

6.17.7 寄存器映射

R: 只读，W: 只写，R/W: 读/写

寄存器	偏移量	R/W	描述	复位值
EMAC基地址: **EMAC_BA = 0x4000_B000**				
EMAC_CAMCTL	EMAC_BA+0x000	R/W	CAM 比较控制寄存器	0x0000_0000
ECAM_CAMEN	EMAC_BA+0x004	R/W	CAM 使能寄存器	0x0000_0000
EMAC_CAM0M	EMAC_BA+0x008	R/W	CAM0 高位字寄存器	0x0000_0000
EMAC_CAM0L	EMAC_BA+0x00C	R/W	CAM0 低位字寄存器	0x0000_0000
EMAC_CAM1M	EMAC_BA+0x010	R/W	CAM1 高位字寄存器	0x0000_0000
EMAC_CAM1L	EMAC_BA+0x014	R/W	CAM1 低位字寄存器	0x0000_0000
EMAC_CAM2M	EMAC_BA+0x018	R/W	CAM2 高位字寄存器	0x0000_0000
EMAC_CAM2L	EMAC_BA+0x01C	R/W	CAM2 低位字寄存器	0x0000_0000
EMAC_CAM3M	EMAC_BA+0x020	R/W	CAM3 高位字寄存器	0x0000_0000
EMAC_CAM3L	EMAC_BA+0x024	R/W	CAM3 低位字寄存器	0x0000_0000
EMAC_CAM4M	EMAC_BA+0x028	R/W	CAM4 高位字寄存器	0x0000_0000
EMAC_CAM4L	EMAC_BA+0x02C	R/W	CAM4 低位字寄存器	0x0000_0000
EMAC_CAM5M	EMAC_BA+0x030	R/W	CAM5 高位字寄存器	0x0000_0000
EMAC_CAM5L	EMAC_BA+0x034	R/W	CAM5 低位字寄存器	0x0000_0000
EMAC_CAM6M	EMAC_BA+0x038	R/W	CAM6 高位字寄存器	0x0000_0000
EMAC_CAM6L	EMAC_BA+0x03C	R/W	CAM6 低位字寄存器	0x0000_0000
EMAC_CAM7M	EMAC_BA+0x040	R/W	CAM7 高位字寄存器	0x0000_0000
EMAC_CAM7L	EMAC_BA+0x044	R/W	CAM7 低位字寄存器	0x0000_0000
EMAC_CAM8M	EMAC_BA+0x048	R/W	CAM8 高位字寄存器	0x0000_0000
EMAC_CAM8L	EMAC_BA+0x04C	R/W	CAM8 低位字寄存器	0x0000_0000
EMAC_CAM9M	EMAC_BA+0x050	R/W	CAM9 高位字寄存器	0x0000_0000
EMAC_CAM9L	EMAC_BA+0x054	R/W	CAM9 低位字寄存器	0x0000_0000
EMAC_CAM10M	EMAC_BA+0x058	R/W	CAM10 高位字寄存器	0x0000_0000
EMAC_CAM10L	EMAC_BA+0x05C	R/W	CAM10 低位字寄存器	0x0000_0000
EMAC_CAM11M	EMAC_BA+0x060	R/W	CAM11 高位字寄存器	0x0000_0000

EMAC_CAM11L	EMAC_BA+0x064	R/W	CAM11 低位字寄存器	0x0000_0000
EMAC_CAM12M	EMAC_BA+0x068	R/W	CAM12 高位字寄存器	0x0000_0000
EMAC_CAM12L	EMAC_BA+0x06C	R/W	CAM12 低位字寄存器	0x0000_0000
EMAC_CAM13M	EMAC_BA+0x070	R/W	CAM13 高位字寄存器	0x0000_0000
EMAC_CAM13L	EMAC_BA+0x074	R/W	CAM13 低位字寄存器	0x0000_0000
EMAC_CAM14M	EMAC_BA+0x078	R/W	CAM14 高位字寄存器	0x0000_0000
EMAC_CAM14L	EMAC_BA+0x07C	R/W	CAM14 低位字寄存器	0x0000_0000
EMAC_CAM15MSB	EMAC_BA+0x080	R/W	CAM15 高位字寄存器	0x0000_0000
EMAC_CAM15LSB	EMAC_BA+0x084	R/W	CAM15 低位字寄存器	0x0000_0000
EMAC_TXDSA	EMAC_BA+0x088	R/W	发送描述符链表起始寄存器	0xFFFF_FFFC
EMAC_RXDSA	EMAC_BA+0x08C	R/W	接收描述符链表起始寄存器	0xFFFF_FFFC
EMAC_CTL	EMAC_BA+0x090	R/W	MAC 控制寄存器	0x0040_0000
EMAC_MIIMDAT	EMAC_BA+0x094	R/W	MII 管理数据寄存器	0x0000_0000
EMAC_MIIMCTL	EMAC_BA+0x098	R/W	MII 管理控制和地址寄存器	0x0090_0000
EMAC_FIFOCTL	EMAC_BA+0x09C	R/W	FIFO 阈值控制寄存器	0x0000_0101
EMAC_TXST	EMAC_BA+0x0A0	W	请求开始发送寄存器	Undefined
EMAC_RXST	EMAC_BA+0x0A4	W	请求开始接收寄存器	Undefined
EMAC_MRFL	EMAC_BA+0x0A8	R/W	最大接收帧控制寄存器	0x0000_0800
EMAC_INTEN	EMAC_BA+0x0AC	R/W	MAC 中断使能寄存器	0x0000_0000
EMAC_INTSTS	EMAC_BA+0x0B0	R/W	MAC 中断状态寄存器	0x0000_0000
EMAC_GENSTS	EMAC_BA+0x0B4	R/W	MAC 通用状态寄存器	0x0000_0000
EMAC_MPCNT	EMAC_BA+0x0B8	R/W	丢包计数寄存器	0x0000_7FFF
EMAC_RPCNT	EMAC_BA+0x0BC	R	MAC 接收到的暂停帧计数寄存器	0x0000_0000
EMAC_FRSTS	EMAC_BA+0x0C8	R/W	DMA 接收帧状态寄存器	0x0000_0000
EMAC_CTXDSA	EMAC_BA+0x0CC	R	当前发送描述符起始地址寄存器	0x0000_0000
EMAC_CTXBSA	EMAC_BA+0x0D0	R	当前发送缓存起始地址寄存器	0x0000_0000
EMAC_CRXDSA	EMAC_BA+0x0D4	R	当前接收描述符起始地址寄存器	0x0000_0000
EMAC_CRXBSA	EMAC_BA+0x0D8	R	当前接收缓存起始地址寄存器	0x0000_0000
EMAC_TSCTL	EMAC_BA+0x100	R/W	时间戳控制寄存器	0x0000_0000

EMAC_TSSEC	EMAC_BA+0x110	R	时间戳秒计数器寄存器	0x0000_0000
EMAC_TSSUBSEC	EMAC_BA+0x114	R	时间戳亚秒计数器寄存器	0x0000_0000
EMAC_TSINC	EMAC_BA+0x118	R/W	时间戳增量寄存器	0x0000_0000
EMAC_TSADDEND	EMAC_BA+0x11C	R/W	时间戳加数寄存器	0x0000_0000
EMAC_UPDSEC	EMAC_BA+0x120	R/W	时间戳更新秒寄存器	0x0000_0000
EMAC_UPDSUBSEC	EMAC_BA+0x124	R/W	时间戳更新亚秒寄存器	0x0000_0000
EMAC_ALMSEC	EMAC_BA+0x128	R/W	时间戳报警秒寄存器	0x0000_0000
EMAC_ALMSUBSEC	EMAC_BA+0x12C	R/W	时间戳报警亚秒寄存器	0x0000_0000

6.17.8 寄存器描述

CAM命令寄存器(EMAC_CAMCTL)

EMAC支持CAM功能以识别目标MAC地址。EMAC_CAMCTL控制CAM比较功能，单播，组播和广播包接收。

寄存器	偏移量	R/W	描述	复位值
EMAC_CAMCTL	EMAC_BA+0x000	R/W	CAM比较控制寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved			CMPEN	COMPEN	ABP	AMP	AUP

位	描述	
[31:5]	Reserved	保留
[4]	CMPEN	**CAM比较使能位** CMPEN控制着是否使能CAM比较功能(识别目标MAC地址)。如果软件想要接收特定目标MAC地址的封包，向CAM12~0填入相应MAC地址，接着使能CAM条目，置CMPEN为1 0=CAM比较功能(识别目标MAC地址)禁止 1=CAM比较功能(识别目标MAC地址)使能
[3]	COMPEN	**CAM余集比较使能位** COMPEN是CAM比较结果的余集。如果同时使能了CMPEN和COMPEN，定义在CAM条目的特定目标MAC地址的封包将会被丢弃。其余来自不符合定义的MAC地址的封包将会被接收。 0=CAM余集比较禁止 1=CAM余集比较使能
[2]	ABP	**接收广播地址** ABP控制着是否接收广播包。如果使能了ABP，EMAC会接收所有目标MAC地址为广播地址的输入包。 0=EMAC接收CAM比较结果指定地址的封包. 1=EMAC接收所有广播包.

[1]	**AMP**	**接收组播包** AMP控制着是否接收组播包。如果使能了AMP，EMAC会接收所有目标MAC地址为组播地址的输入包。 0=EMAC接收CAM比较结果指定地址的封包. 1=EMAC接收所有组播包.
[0]	**AUP**	**接收单包** AUP控制着是否接收单播包。如果使能了AUP，EMAC会接收所有目标MAC地址为单播地址的输入包。 0=EMAC接收CAM比较结果指定地址的封包. 1=EMAC接收所有单播包.

CAMCMR 设置和比较结果

表6.17-14为不同CAMCMR配置下的地址识别结果。结果列显示了哪些输入封包类型可以通过CAM配置的可识别地址。C,U,M和B分别代表：

C:CAM条目定义了输入封包的目标地址.

U:输入封包为单播包

M:输入封包为组播包

B:输入封包为广播包

CMPEN	CCAM	AUP	AMP	ABP	结果			
0	0	0	0	0	无封包			
0	0	0	0	1	B			
0	0	0	1	0	M			
0	0	0	1	1	M	B		
0	0	1	0	0	C	U		
0	0	1	0	1	C	U	B	
0	0	1	1	0	C	U	M	
0	0	1	1	1	C	U	M	B
0	1	0	0	0	C	U	M	B
0	1	0	0	1	C	U	M	B
0	1	0	1	0	C	U	M	B
0	1	0	1	1	C	U	M	B
0	1	1	0	0	C	U	M	B
0	1	1	0	1	C	U	M	B
0	1	1	1	0	C	U	M	B
0	1	1	1	1	C	U	M	B

1	0	0	0	0	C			
1	0	0	0	1	C	B		
1	0	0	1	0	C	M		
1	0	0	1	1	C	M	B	
1	0	1	0	0	C	U		
1	0	1	0	1	C	U	B	
1	0	1	1	0	C	U	M	
1	0	1	1	1	C	U	M	B
1	1	0	0	0	U	M	B	
1	1	0	0	1	U	M	B	
1	1	0	1	0	U	M	B	
1	1	0	1	1	U	M	B	
1	1	1	0	0	C	U	M	B
1	1	1	0	1	C	U	M	B
1	1	1	1	0	C	U	M	B
1	1	1	1	1	C	U	M	B

表 6.17-14 不同 CAMCMR 设置及可接收的包类型

CAM使能寄存器(ECAM_CAMEN)

ECAM_CAMEN控制着各CAM条目的有效性。每个CAM条目在参与目标MAC地址识别前需要提前使能。

寄存器	偏移量	R/W	描述	复位值
ECAM_CAMEN	EMAC_BA+0x004	R/W	CAM使能寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
CAM15EN	CAM14EN	CAM13EN	CAM12EN	CAM11EN	CAM10EN	CAM9EN	CAM8EN
7	6	5	4	3	2	1	0
CAM7EN	CAM6EN	CAM5EN	CAM4EN	CAM3EN	CAM2EN	CAM1EN	CAM0EN

位	描述	
[31:16]	Reserved	保留
[x]	CAMxEN	**CAM条目X使能位** ECAM_CAMEN控制着各CAM条目X的有效性. CAM条目13，14和15用于暂停帧的发送。软件发送暂停帧之前需要先使能这些CAM条目。 0=CAM条目x禁止. 1=CAM条目x使能.

CAM条目寄存器(EMAC_CAMxMSB,x=0,1,2..14)

EMAC总共有16个CAM条目。在这16个CAM条目中,有13个(0~12条目)用于存储包识别的目标MAC地址。其他3个(13~15条目)用于暂停帧的传输。每个条目包含6个字节，所以需要两个寄存器。

寄存器对{EMAC_CAMxMSB,ECAMxL}是用于存储包识别的目标MAC地址的CAM条目。使用时必须使能CAMxEN(ECAM_CAMEN[x])对应位，x=0~12.

寄存器对{EMAC_CAM13M,EMAC_CAM13L},{EMAC_CAM14M,EMAC_CAM14L}和{EMAC_CAM15MSB,EMAC_CAM15LSB}用于流控。

寄存器	偏移量	R/W	描述	复位值
EMAC_CAM0M	EMAC_BA+0x008	R/W	CAM0高位字寄存器	0x0000_0000
EMAC_CAM1M	EMAC_BA+0x010	R/W	CAM1高位字寄存器	0x0000_0000
EMAC_CAM2M	EMAC_BA+0x018	R/W	CAM2高位字寄存器	0x0000_0000
EMAC_CAM3M	EMAC_BA+0x020	R/W	CAM3高位字寄存器	0x0000_0000
EMAC_CAM4M	EMAC_BA+0x028	R/W	CAM4高位字寄存器	0x0000_0000
EMAC_CAM5M	EMAC_BA+0x030	R/W	CAM5高位字寄存器	0x0000_0000
EMAC_CAM6M	EMAC_BA+0x038	R/W	CAM6高位字寄存器	0x0000_0000
EMAC_CAM7M	EMAC_BA+0x040	R/W	CAM7高位字寄存器	0x0000_0000
EMAC_CAM8M	EMAC_BA+0x048	R/W	CAM8高位字寄存器	0x0000_0000
EMAC_CAM9M	EMAC_BA+0x050	R/W	CAM9高位字寄存器	0x0000_0000
EMAC_CAM10M	EMAC_BA+0x058	R/W	CAM10高位字寄存器	0x0000_0000
EMAC_CAM11M	EMAC_BA+0x060	R/W	CAM11高位字寄存器	0x0000_0000
EMAC_CAM12M	EMAC_BA+0x068	R/W	CAM12高位字寄存器	0x0000_0000
EMAC_CAM13M	EMAC_BA+0x070	R/W	CAM13高位字寄存器	0x0000_0000
EMAC_CAM14M	EMAC_BA+0x078	R/W	CAM14高位字寄存器	0x0000_0000

31	30	29	28	27	26	25	24
MACADDR5							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
MACADDR4							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
MACADDR3							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
MACADDR2							

位	描述	
[31:24]	**MACADDR5**	**MAC地址字节5** CAMxM保存MAC地址的47~16位。x为0~14。寄存器对{EMAC_CAMxM,EMAC_CAMxL}代表一个CAM条目，保存整个MAC地址。 例如CAM条目1保存的MAC地址是00-50-BA-33-BA-44，寄存器EMAC_CAM1M是0x0050_BA33，寄存器EMAC_CAM1L是0xBA44_0000.
[23:16]	**MACADDR4**	**MAC地址字节4**
[15:8]	**MACADDR3**	**MAC地址字节3**
[7:0]	**MACADDR2**	**MAC地址字节2**

CAM条目寄存器(EMAC_CAMxLSB;x=0,1,2..14)

寄存器	偏移量	R/W	描述	复位值
EMAC_CAM0L	EMAC_BA+0x00C	R/W	CAM0低位字寄存器	0x0000_0000
EMAC_CAM1L	EMAC_BA+0x014	R/W	CAM1低位字寄存器	0x0000_0000
EMAC_CAM2L	EMAC_BA+0x01C	R/W	CAM2低位字寄存器	0x0000_0000
EMAC_CAM3L	EMAC_BA+0x024	R/W	CAM3低位字寄存器	0x0000_0000
EMAC_CAM4L	EMAC_BA+0x02C	R/W	CAM4低位字寄存器	0x0000_0000
EMAC_CAM5L	EMAC_BA+0x034	R/W	CAM5低位字寄存器	0x0000_0000
EMAC_CAM6L	EMAC_BA+0x03C	R/W	CAM6低位字寄存器	0x0000_0000
EMAC_CAM7L	EMAC_BA+0x044	R/W	CAM7低位字寄存器	0x0000_0000
EMAC_CAM8L	EMAC_BA+0x04C	R/W	CAM8低位字寄存器	0x0000_0000
EMAC_CAM9L	EMAC_BA+0x054	R/W	CAM9低位字寄存器	0x0000_0000
EMAC_CAM10L	EMAC_BA+0x05C	R/W	CAM10低位字寄存器	0x0000_0000
EMAC_CAM11L	EMAC_BA+0x064	R/W	CAM11低位字寄存器	0x0000_0000
EMAC_CAM12L	EMAC_BA+0x06C	R/W	CAM12低位字寄存器	0x0000_0000
EMAC_CAM13L	EMAC_BA+0x074	R/W	CAM13低位字寄存器	0x0000_0000
EMAC_CAM14L	EMAC_BA+0x07C	R/W	CAM14低位字寄存器	0x0000_0000

31	30	29	28	27	26	25	24
MACADDR1							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
MACADDR0							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved							

位	描述	
[31:24]	**MACADDR1**	**MAC地址字节1** CAMxM保存MAC地址的15~0位。x为0~14。寄存器对{EMAC_CAMxM,EMAC_CAMxL}代表一个CAM条目，保存整个MAC地址。 例如CAM条目1保存的MAC地址是00-50-BA-33-BA-44，寄存器EMAC_CAM1M是0x0050_BA33，寄存器EMAC_CAM1L是0xBA44_0000.

[23:16]	**MACADDR0**	**MAC地址字节0**
[15:0]	**Reserved**	保留

CAM条目寄存器(EMAC_CAM15MSB)

寄存器	偏移量	R/W	描述	复位值
EMAC_CAM15MSB	EMAC_BA+0x080	R/W	CAM15高位字寄存器	0x0000_0000

31	30	29	28	27	26	25	24
LENGTH							
23	22	21	20	19	18	17	16
LENGTH							
15	14	13	12	11	10	9	8
OPCODE							
7	6	5	4	3	2	1	0
OPCODE							

位	描述	
[31:16]	LENGTH	**暂停帧的长度字段** 在暂停帧里，定义为0x8808.
[15:0]	OPCODE	**暂停帧的操作码字段** 在暂停帧里，定义为0x0001.

CAM条目寄存器(EMAC_CAM15LSB)

寄存器	偏移量	R/W	描述	复位值
EMAC_CAM15LSB	EMAC_BA+0x084	R/W	CAM15低位字寄存器	0x0000_0000

31	30	29	28	27	26	25	24
OPERAND							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved							

位	描述	
[31:24]	OPERAND	**暂停参数** 在暂停帧里，OPERAND字段控制着目标地址以太网MAC控制器的暂停时间。OPERAND的单位是时间片，512位时间
[23:0]	Reserved	保留

发送描述符链表起始地址寄存器(EMAC_TXDSA)

TX描述符是EMAC中的一个链表数据结构。EMAC_TXDSA保存着该链表的起始地址，即EMAC_TXDSA保存着第一个TX描述符的起始地址。软件必须在使能TXON(EMAC_CTL[8])前配置完EMAC_TXDSA。

寄存器	偏移量	R/W	描述	复位值
EMAC_TXDSA	EMAC_BA+0x088	R/W	发送描述符链表起始寄存器	0xFFFF_FFFC

31	30	29	28	27	26	25	24
TXDSA							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
TXDSA							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
TXDSA							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
TXDSA							

位	描述	
[31:0]	**TXDSA**	**发送描述符链表起始地址** TXDSA存储着发送描述符链表起始地址。软件使能TXON(EMAC_CTL[8])会将TXDSA的内容载入当前发送描述符起始地址寄存器(EMAC_CTXDSA)。EMAC不会再更新TXDSA的值。操作中，EMAC会忽略TXDSA[1:0]的值。也就是说，TX描述符一定要存储在内存地址的字边界。

接收描述符链表起始地址寄存器(EMAC_RXDSA)

RX描述符是EMAC中的一个链表数据结构。EMAC_RXDSA保存着该链表的起始地址，即EMAC_RXDSA保存着第一个RX描述符的起始地址。软件必须在使能RXON(EMAC_CTL[0])前配置完EMAC_RXDSA。

寄存器	偏移量	R/W	描述	复位值
EMAC_RXDSA	EMAC_BA+0x08C	R/W	接收描述符链表起始寄存器	0xFFFF_FFFC

31	30	29	28	27	26	25	24
RXDSA							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
RXDSA							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
RXDSA							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
RXDSA							

位	描述	
[31:0]	**RXDSA**	**接收描述符链表起始地址** RXDSA存储着发送描述符链表起始地址。软件使能RXON(EMAC_CTL[0])会将RXDSA的内容载入当前接收描述符起始地址寄存器(EMAC_CRXDSA)。EMAC不会再更新RXDSA的值。操作中，EMAC会忽略RXDSA[1:0]的值。也就是说，RX描述符一定要存储在内存地址的字边界。

MAC控制寄存器(EMAC_CTL)

EMAC_CTL提供了EMAC的控制信息。其中部分指令在帧的发送和接收中都有作用，如半/全双工选择位FUDUP(EMAC_CTL[18])，100/10Mbps速度选择位OPMODE(EMAC_CTL[20])。有些指令则分别控制帧的发送和接收如TXON(EMAC_CTL[8])和RXON(EMAC_CTL[0])。

寄存器	偏移量	R/W	描述	复位值
EMAC_CTL	EMAC_BA+0x090	R/W	MAC控制寄存器	0x0040_0000

31	30	29	28	27	26	25	24
Reserved							RST
23	22	21	20	19	18	17	16
Reserved	RMIIEN	Reserved	OPMODE	RMIIRXCTL	FUDUP	SQECHKEN	SDPZ
15	14	13	12	11	10	9	8
Reserved						NODEF	TXON
7	6	5	4	3	2	1	0
Reserved	WOLEN	STRIPCRC	AEP	ACP	ARP	ALP	RXON

位	描述	
[31:25]	**Reserved**	保留
[24]	**RST**	**软件复位** RST实现了软件复位的功能，可以使EMAC恢复默认状态。RST是一个自动清0的位。也就是说，在软件复位完成后，RST会自动清0。使能RST可以复位所有的控制和状态寄存器，除了RMIIEN(EMAC_CTL[22])和OPMODE(EMAC_CTL[20])。 软件复位完成后，EMAC需要重新初始化 0=软件复位完成 1=软件复位使能
[23]	**Reserved**	保留
[22]	**RMIIEN**	**RMII模式使能位** 此位控制通过MII还是RMII连接外部物理层芯片，复位(EMAC_CTL[24])不影响RMIIEN的值 0=RMII模式禁止 1=RMII模式使能. **注:**该字段必须为1.
[21]	**Reserved**	保留
[20]	**OPMODE**	**工作模式选择** OPMODE定义EMAC是处于10Mbps还是100Mbps模式。RST(EMAC_CTL[24])复位不影响OPMODE值 0=EMAC工作做在10Mbps模式. 1=EMAC工作做在100Mbps模式.

[19]	**RMIIRXCTL**	**RMIIRX控制** RMIIRXCTL控制这RMII模式下的数据接收。RMIIEN(EMAC_CTL[[22])置位时，该位必须使能 0=RMIIRX控制禁止. 1=RMIIRX控制使能.
[18]	**FUDUP**	**全双工模式选择** FUDUP控制EMAC工作在半双工还是全双工模式下 0=EMAC工作在半双工模式下 1=EMAC工作在全双工模式下
[17]	**SQECHKEN**	**SQE检验使能位** SQECHKEN控制着SQE检验的使能。SQE检测仅在10Mbps的半双工模式下有效。也就是说SQECKEN并不影响EMAC的工作模式。 0=SQE检验在10Mbps的双工模式下禁止 1=SQE检验在10Mbps的双工模式下使能
[16]	**SDPZ**	**发送暂停帧** SDPZ控制这暂停帧的发送 软件发送暂停帧前，需要确保CAM条目13,14,15已配置完成且对应的CAMEN寄存器使能位已置位。接着，置SDPZ为1使能暂停帧发送。 SDPZ是自清零的位。也就是说，在暂停帧发送完成后，SDPZ会自动清0。 建议仅在全双工模式下使能SNDPAUSE 0=暂停帧发送完成 1=暂停帧发送使能
[15:10]	**Reserved**	保留
[9]	**NODEF**	**关闭延期计数** NODEF控制着是否关闭延期计数功能。如果NODEF置位，延期计数功能关闭。延期计数功能仅在EMAC半双工模式下有效。 0=使能延期计数 1=关闭延期计数
[8]	**TXON**	**帧发送启动** TXON控制着EMAC正常包的发送。TXON置位时，EMAC开始发送封包的进程，包括获取TX描述符，包发送和TX描述符的修改。 使能TXON前需要完成EMAC初始化流程。否则，EMAC将会不工作。 如果EMAC发送过程中关闭了TXON，EMAC会在当前帧发送完成后停止封包发送进程 0=包发送停止 1=包发送启动
[7]	**Reserved**	保留
[6]	**WOLEN**	**LAN口唤醒使能位** WOKEN置1使能，以太网控MAC制器会在检测到魔术包输入时从掉电模式唤醒系统 如果掉电模式下，以太网控制器检测到魔术包输入，以太网MAC控制器会产生一个唤醒事件来唤醒系统 0=魔术包唤醒功能禁止 1=魔术包唤醒功能使能

[5]	**STRIPCRC**	**去除CRC** STRIPCRC标示计算输入封包长度是否包含4字节的CRC校验和。如果STRIPCRC置1，计算输入包长度时将不会计算4字节的CRC校验和。 0=计算输入包时包含4字节的CRC校验和 1=计算输入包时不包含4字节的CRC校验和
[4]	**AEP**	**接收CRC出错包** AEP控制EMAC接收或丢弃CRC出错的包。如果AEP为1，CRC出错的包将被EMAC视为正常的封包进行接收 0=EMAC控制器丢弃CRC出错的包 1=EMAC控制器接收CRC出错的包
[3]	**ACP**	**接收控制包** ACP控制着控制帧的接收。如果ACP为1，EMAC会接收控制帧。否则，控制帧会被丢弃。建议仅在EMAC全双工模式下使能ACP 0=EMAC控制器丢弃控制帧 1=EMAC控制器接收控制帧
[2]	**ARP**	**接收短包** ARP控制是否接收长度小于64字节的短包。如果ARP为1，EMAC接收短包，否则会丢弃短包。 0=EMAC控制器丢弃短包 1=EMAC控制器接收短包
[1]	**ALP**	**接收长包** ALP控制是否接收长度大于1518字节的长包。如果ALP为1，EMAC接收长包，否则会丢弃长包。 0=EMAC控制器丢弃长包 1=EMAC控制器接收长包
[0]	**RXON**	**帧接收启动** RXON控制着EMAC正常包的接收。RXON置位时，EMAC开始接收封包的进程，包括获取RX描述符，包接收和RX描述符的修改。 使能RXON前需要完成EMAC初始化流程。否则，EMAC将会不工作。 如果EMAC接收过程中关闭了RXON，EMAC会在当前帧接收完成后停止封包接收进程 0=包接收停止 1=包接收启动

MII管理数据寄存器(EMAC_MIIMDAT)

EMAC通过MII管理功能控制和读取外部PHY的寄存器状态。EMAC_MIIMDAT寄存器用来存储写入命令将要写入外部PHY寄存器的数据及读取命令从外部PHY寄存器读得的数据。

寄存器	偏移量	R/W	描述	复位值
EMAC_MIIMDAT	EMAC_BA+0x094	R/W	MII管理数据寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
DATA							
7	6	5	4	3	2	1	0
DATA							

位	描述	
[31:16]	Reserved	保留
[15:0]	DATA	**MII管理数据** 16位的DATA为，写入命令将要写入外部PHY寄存器的数据或读取命令从外部PHY寄存器读得的数据

MII管理控制和地址寄存器(EMAC_MIIMCTL)

EMAC通过MII管理功能控制和读取外部PHY的寄存器状态。EMAC_MIIMCTL寄存器用来存储MII管理的命令信息，如寄存器地址，外部PHY地址，MDC时钟率，读/写命令等。

寄存器	偏移量	R/W	描述	复位值
EMAC_MIIMCTL	EMAC_BA+0x098	R/W	MII管理控制和地址寄存器	0x0090_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved				MDCON	PREAMSP	BUSY	WRITE
15	14	13	12	11	10	9	8
Reserved			PHYADDR				
7	6	5	4	3	2	1	0
Reserved			PHYREG				

位	描述	
[31:20]	Reserved	保留
[19]	MDCON	**MDC时钟使能** MDC控制着MDC时钟的发生。MDC置1，MDC时钟使能 0=MDC时钟关闭 1=MDC时钟使能
[18]	PREAMSP	**前导码禁止** PREAMSP控制着MII管理帧的前导码产生。如果PREAMSP为高，MII管理帧产生将会跳过前导码 0=MII管理帧产生将会有正常的前导码 1=MII管理帧产生将会跳过前导码
[17]	BUSY	**忙** 该位控制着MII管理帧发生的使能。软件操作外部PHY寄存器前，需要先置该位为1，再由EMAC产生MII管理帧到外部PHY。该位自动清0。也就是说，该位在MII管理命令完成后会自动清0。 0=MII管理命令完成 1=MII管理命令使能
[16]	WRITE	**写命令** 该命令指示MII管理命令是读还是写 0=MII管理命令是读命令 1=MII管理命令是写命令
[15:13]	Reserved	保留

[12:8]	**PHYADDR**	**PHY地址** PHYADDR保存了MII管理命令的目标PHY地址
[7:5]	**Reserved**	保留
[4:0]	**PHYREG**	**PHY寄存器地址** PHYREG保存了MII管理命令的目标PHY寄存器地址

MII 管理功能帧格式

IEEEStd.802.3的22.2.4章节定义了MII管理功能的相关规范。MII管理功能是指对PHY的控制和状态读取。如下是MII管理帧的格式：

	管理帧字段							
	PRE	ST	OP	PHYAD	REGAD	TA	DATA	IDLE
READ	1…1	01	10	AAAAA	RRRRR	Z0	DDDDDDDDDDDDDDDD	Z
WRITE	1…1	01	01	AAAAA	RRRRR	10	DDDDDDDDDDDDDDDD	Z

表 6.17-15MII 管理帧格式

MII 管理功能配置时序

读	**写**
1.配置合适的EMAC_MDCCR. 2.配置PHYADDR和PHYREG. 3.写0到WRITE寄存器 4.写1到BUSY(EMAC_MIIMCTL[17])，发送一个MII管理帧 5.等BUSY(EMAC_MIIMCTL[17])位变为0. 6.从EMAC_MIIMDAT寄存器读数据 7.完成	1.写数据到EMAC_MIIMDAT寄存器 2.配置合适的EMAC_MDCCR. 3.配置PHYADDR和PHYREG. 4.写1到WRITE寄存器 5.写1到BUSY(EMAC_MIIMCTL[17])，发送一个MII管理帧 6.等BUSY(EMAC_MIIMCTL[17])位变为0. 7.完成

表 6.17-16MII 管理功能配置时序

FIFO阈值控制寄存器(EMAC_FIFOCTL)

EMAC_FIFOCTL定义了内部FIFO的高域和低域，包括TXFIFO和RXFIFO。内部的FIFO域和EMAC请求发生及帧发送开始有关。EMAC_FIFOCTL还定义了访问系统内存时的最大AHB总线时钟数。

寄存器	偏移量	R/W	描述	复位值
EMAC_FIFOCTL	EMAC_BA+0x09C	R/W	FIFO阈值控制寄存器	0x0000_0101

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved		BURSTLEN		Reserved			
15	14	13	12	11	10	9	8
Reserved						TXFIFOTH	
7	6	5	4	3	2	1	0
Reserved						RXFIFOTH	

位	描述	
[31:22]	Reserved	保留
[21:20]	BURSTLEN	**DMA最大长度** 定义了EMAC访问系统内存时需要的最大AHB总线时钟数 00=4字. 01=8字. 10=16字. 11=16字.
[19:10]	Reserved	保留
[9:8]	TXFIFOTH	**TXFIFO门限** TXFIFOTH控制着系统内存到TXFIFO的TXDMA请求。TXFIFOTH不仅定义了TXFIFO的下限值还定义了其上限值。一般来说，上限值是下限值的2倍。在包传送过程中，如果TXFIFO达到了上限值，TXDMA会停止请求把数据从系统内存传到TXFIFO。如果，TXFIFO里的帧数据小于其下限值，TXDMA会把系统内存的帧数据传入TXFIFO。 TXFIFOTH还定义了何时TXMAC开始把帧发送到外部网络。TXFIFO中的数据第一次达到上限时，TXMAC开始发送，若一帧数据个数少于上限，此帧所有数据全写入TXFIFO后也会启动发送。 00=未定义. 01=TXFIFO下限64B上限128B. 10=TXFIFO下限80B上限160B. 11=TXFIFO下限96B上限192B.
[7:2]	Reserved	保留

[1:0]	**RXFIFOTH**	**RXFIFO门限** RXFIFOTH控制着RXFIFO到系统内存的RXDMA请求。RXFIFOTH不仅定义了RXFIFO的下限值还定义了其上限值。一般来说，上限值是下限值的2倍。在包接收过程中，如果RXFIFO达到了上限值，RXDMA会把数据从RXFIFO传到系统内存。如果，RXFIFO里的帧数据小于其下限值，RXDMA会停止把帧数据传入系统内存。 00=按突发传输的配置，.若突发传输长8个字，此处上限亦8个字 01=RXFIFO上限64B下限32B. 10=RXFIFO上限128B下限64B. 11=RXFIFO上限192B下限96B.

请求开始发送寄存器(EMAC_TXST)

软件产生一个写命令到EMAC_TXST寄存器使TXDMA脱离停止状态开始帧传送。

寄存器	偏移量	R/W	描述	复位值
EMAC_TXST	EMAC_BA+0x0A0	W	请求开始发送寄存器	Undefined

31	30	29	28	27	26	25	24
TXST							
23	22	21	20	19	18	17	16
TXST							
15	14	13	12	11	10	9	8
TXST							
7	6	5	4	3	2	1	0
TXST							

位	描述	
[31:0]	TXST	**发送开始请求** TXON(EMAC_CTL[8])使能后，若TX描述符仍不可用，TXDMA的FSM机制进入中止状态，TX描述符准备好以后，必须写个命令到存器EMAC_TXST，让TXDMA继续发送。 此寄存器只写。读，未定义。 仅当TXDMA处于关闭状态时，写该寄存器才有效

请求开始接收寄存器(EMAC_RXST)

软件产生一个写命令到EMAC_RXST寄存器使RXDMA脱离停止状态开始帧接收。

寄存器	偏移量	R/W	描述	复位值
EMAC_RXST	EMAC_BA+0x0A4	W	请求开始接收寄存器	Undefined

31	30	29	28	27	26	25	24
RXST							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
RXST							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
RXST							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
RXST							

位	描述	
[31:0]	**RXST**	**接收开始请求** 使能RXON(EMAC_CTL[0])以后，若RX描述符仍不可用，RXDMA机制将进入中止状态。RX描述符准备好以后，写一个命令到此寄存器，RXDMA将继续接收。 此寄存器只写。读，未定义。 仅当RXDMA处于关闭状态时，写该寄存器才有效。

最大接收帧控制寄存器(EMAC_MRFL)

EMAC_MRFL定义了内存可以存储的最大接收帧长度。建议仅在软件希望接收超过1518字节的帧时才使用该寄存器。

寄存器	偏移量	R/W	描述	复位值
EMAC_MRFL	EMAC_BA+0x0A8	R/W	最大接收帧控制寄存器	0x0000_0800

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
MRFL							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
MRFL							

位	描述	
[31:16]	**Reserved**	保留
[15:0]	**MRFL**	**最大接收帧长度** MRFL定义了最大接收帧长度.接收帧长度超过此定义时，位MFLEIF(EMAC_INTSTS[8])置1，若MFLEIEN(EMAC_INTEN[8])=1，将触发RX中断. 建议仅在软件希望接收超过1518字节的帧时才使用该寄存器

MAC中断使能及存储区(EMAC_INTEN)

EMAC_INTEN控制着EMAC的中断使能。从EMAC到CPU总共有两个中断，帧接收中断RXIF和帧发送中断TXIF。

寄存器	偏移量	R/W	描述	复位值
EMAC_INTEN	EMAC_BA+0x0AC	R/W	MAC中断使能寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved			TSALMIEN	Reserved			TXBEIEN
23	22	21	20	19	18	17	16
TDUIEN	LCIEN	TXABTIEN	NCSIEN	EXDEFIEN	TXCPIEN	TXUDIEN	TXIEN
15	14	13	12	11	10	9	8
WOLIEN	CFRIEN	Reserved		RXBEIEN	RDUIEN	DENIEN	MFLEIEN
7	6	5	4	3	2	1	0
MPCOVIEN	RPIEN	ALIEIEN	RXGDIEN	LPIEN	RXOVIEN	CRCEIEN	RXIEN

位	描述	
[31:29]	**Reserved**	保留
[28]	**TSALMIEN**	**时间戳报警中断使能位** TSALMIEN控制着TSALMIF(EMAC_INTSTS[28])中断的发生。如果TSALMIF(EMAC_INTSTS[28])置位，且TSALMIEN和TXIEN(EMAC_INTEN[16])都使能了，EMAC将触发TX中断到CPU。如果TSALMIEN或TXIEN(EMAC_INTEN[16])关闭了，即使TXTSALMIF(EMAC_INTSTS[28])置位也不会触发TX中断。 0=禁止TXTSALMIF(EMAC_INTSTS[28])触发TX中断. 1=使能TXTSALMIF(EMAC_INTSTS[28])触发TX中断.
[27:25]	**Reserved**	保留
[24]	**TXBEIEN**	**发送总线错误中断使能位** TXBEIEN控制着TXBEIF(EMAC_INTSTS[24])中断的发生。如果TXBEIF(EMAC_INTSTS[24])置位，且TXBEIEN和TXIEN(EMAC_INTEN[16])都使能了，EMAC将触发TX中断到CPU。如果TXBEIEN或TXIEN(EMAC_INTEN[16])关闭了，即使TXBEIF(EMAC_INTSTS[24])置位也不会触发TX中断。 0=禁止TXBEIF(EMAC_INTSTS[24])触发TX中断. 1=使能TXBEIF(EMAC_INTSTS[24])触发TX中断.
[23]	**TDUIEN**	**发送描述符不可用中断使能位** TDUIEN控制着TDUIF(EMAC_INTSTS[23])中断的发生。如果TDUIF置位，且TDUIEN和TXIEN(EMAC_INTEN[16])都使能了，EMAC将触发TX中断到CPU。如果TDUIEN或TXIEN(EMAC_INTEN[16])关闭了，即使TDUIF(EMAC_INTSTS[23])置位也不会触发TX中断。 0=禁止TDUIF(EMAC_INTSTS[23])触发TX中断. 1=使能TDUIF(EMAC_INTSTS[23])触发TX中断.

[22]	**LCIEN**	**迟到冲突中断使能位** LCIEN控制着LCIF(EMAC_INTSTS[22])中断的发生。如果LCIF(EMAC_INTSTS[22])置位，且LCIEN和TXIEN(EMAC_INTEN[16])都使能了，EMAC将触发TX中断到CPU。如果TSALMIEN或TXIEN(EMAC_INTEN[16])关闭了，即使LCIF(EMAC_INTSTS[22])置位也不会触发TX中断。 0=禁止LCIF(EMAC_INTSTS[22])触发TX中断. 1=使能LCIF(EMAC_INTSTS[22])触发TX中断.
[21]	**TXABTIEN**	**发送停止中断使能位** TXABTIEN控制着TXABTIF(EMAC_INTSTS[21])中断的发生。如果TXABTIF(EMAC_INTSTS[21])置位，且TXABTIEN和TXIEN(EMAC_INTEN[16])都使能了，EMAC将触发TX中断到CPU。如果TXABTIEN或TXIEN(EMAC_INTEN[16])关闭了，即使TXABTIF(EMAC_INTSTS[21])置位也不会触发TX中断。 0=禁止TXABTIF(EMAC_INTSTS[21])触发TX中断. 1=使能TXABTIF(EMAC_INTSTS[21])触发TX中断.
[20]	**NCSIEN**	**无载波中断使能位** NCSIEN控制着NCSIF(EMAC_INTSTS[20])中断的发生。如果NCSIF(EMAC_INTSTS[20])置位，且NCSIEN和TXIEN(EMAC_INTEN[16])都使能了，EMAC将触发TX中断到CPU。如果NCSIEN或TXIEN(EMAC_INTEN[16])关闭了，即使NCSIF(EMAC_INTSTS[20])置位也不会触发TX中断。 0=禁止NCSIF(EMAC_INTSTS[20])触发TX中断. 1=使能NCSIF(EMAC_INTSTS[20])触发TX中断.
[19]	**EXDEFIEN**	**延期超时中断使能位** EXDEFIEN控制着EXDEFIF(EMAC_INTSTS[19])中断的发生。如果EXDEFIF(EMAC_INTSTS[19])置位，且EXDEFIEN和TXIEN(EMAC_INTEN[16])都使能了，EMAC将触发TX中断到CPU。如果EXDEFIEN或TXIEN(EMAC_INTEN[16])关闭了，即使EXDEFIF(EMAC_INTSTS[19])置位也不会触发TX中断。 0=禁止EXDEFIF(EMAC_INTSTS[19])触发TX中断. 1=使能EXDEFIF(EMAC_INTSTS[19])触发TX中断.
[18]	**TXCPIEN**	**发送完成中断使能位** TXCPIEN控制着TXCPIF(EMAC_INTSTS[18])中断的发生。如果TXCPIF(EMAC_INTSTS[18])置位，且TXCPIEN和TXIEN(EMAC_INTEN[16])都使能了，EMAC将触发TX中断到CPU。如果TXCPIEN或TXIEN(EMAC_INTEN[16])关闭了，即使TXCPIF(EMAC_INTSTS[18])置位也不会触发TX中断。 0=禁止TXCPIF(EMAC_INTSTS[18])触发TX中断. 1=使能TXCPIF(EMAC_INTSTS[18])触发TX中断.
[17]	**TXUDIEN**	**发送FIFO下溢使能位** TXUDIEN控制着TXUDIF(EMAC_INTSTS[17])中断的发生。如果TXUDIF(EMAC_INTSTS[17])置位，且TXUDIEN和TXIEN(EMAC_INTEN[16])都使能了，EMAC将触发TX中断到CPU。如果TXUDIEN或TXIEN(EMAC_INTEN[16])关闭了，即使TXUDIF(EMAC_INTSTS[17])置位也不会触发TX中断。 0=禁止TXUDIF(EMAC_INTSTS[17])触发TX中断. 1=使能TXUDIF(EMAC_INTSTS[17])触发TX中断.

[16]	**TXIEN**	**发送中断使能位** TXIEN控制着TX中断的发生 如果使能了TXIEN且TXIF(EMAC_INTSTS[16])置位，EMAC将产生TX中断到CPU。如果TXIEN未使能，即使EMAC_INTSTS[24:17]中某一位置且对应的EMAC_INTEN已使能，都不会产生TX中断。也就是说，软件如果想要使能TX中断，必须要先使能该位。想要关闭TX中断时，禁止该位即可。 0=TXIF(EMAC_INTSTS[16])被屏蔽，禁止TX中断的发生 1=TXIF(EMAC_INTSTS[16])未屏蔽，允许TX中断的发生
[15]	**WOLIEN**	**LAN口唤醒中断使能位** WOLIEN控制着WOLIF(EMAC_INTSTS[15])中断的发生。如果WOLIF(EMAC_INTSTS[15])置位，且WOLIEN和RXIEN(EMAC_INTEN[0])都使能了，EMAC将触发RX中断到CPU。如果WOLIEN或RXIEN(EMAC_INTEN[0])关闭了，即使WOLIF(EMAC_INTSTS[15])置位也不会触发RX中断。 0=禁止WOLIF(EMAC_INTSTS[15])触发RX中断. 1=使能WOLIF(EMAC_INTSTS[15])触发RX中断.
[14]	**CFRIEN**	**控制帧接收中断使能位** CFRIEN控制着CFRIF(EMAC_INTSTS[14])中断的发生。如果CFRIF(EMAC_INTSTS[14])置位，且CFRIEN和RXIEN(EMAC_INTEN[0])都使能了，EMAC将触发RX中断到CPU。如果CFRIEN或RXIEN(EMAC_INTEN[0])关闭了，即使CFRIF(EMAC_INTSTS[14])置位也不会触发RX中断。 0=禁止CFRIF(EMAC_INTSTS[14])触发RX中断. 1=使能CFRIF(EMAC_INTSTS[14])触发RX中断.
[13:12]	**Reserved**	保留
[11]	**RXBEIEN**	**接收总线错误中断使能位** RXBEIEN控制着RXBEIF(EMAC_INTSTS[11])中断的发生。如果RXBEIF(EMAC_INTSTS[11])置位，且RXBEIEN和RXIEN(EMAC_INTEN[0])都使能了，EMAC将触发RX中断到CPU。如果RXBEIEN或RXIEN(EMAC_INTEN[0])关闭了，即使RXBEIF(EMAC_INTSTS[11])置位也不会触发RX中断。 0=禁止RXBEIF(EMAC_INTSTS[11])触发RX中断. 1=使能RXBEIF(EMAC_INTSTS[11])触发RX中断.
[10]	**RDUIEN**	**接收描述符不可用中断使能位** RDUIEN控制着RDUIF(EMAC_INTSTS[10])中断的发生。如果RDUIF(EMAC_INTSTS[10])置位，且RDUIEN和RXIEN(EMAC_INTEN[0])都使能了，EMAC将触发RX中断到CPU。如果RDUIEN或RXIEN(EMAC_INTEN[0])关闭了，即使RDUIF(EMAC_MIOSTA[10])置位也不会触发RX中断。 0=禁止RDUIF(EMAC_MIOSTA[10])触发RX中断. 1=使能RDUIF(EMAC_MIOSTA[10])触发RX中断.
[9]	**DENIEN**	**DMA早期通知中断使能位** DENIEN控制着DENIF(EMAC_INTSTS[9])中断的发生。如果DENIF(EMAC_INTSTS[9])置位，且DENIEN和RXIEN(EMAC_INTEN[0])都使能了，EMAC将触发RX中断到CPU。如果DENIEN或RXIEN(EMAC_INTEN[0])关闭了，即使DENIF(EMAC_INTSTS[9])置位也不会触发RX中断。 0=禁止DENIF(EMAC_INTSTS[9])触发RX中断. 1=使能DENIF(EMAC_INTSTS[9])触发RX中断.

[8]	**MFLEIEN**	**超出最大帧长度中断使能位** MFLEIEN 控制着 MFLEIF(EMAC_INTSTS[8]) 中断的发生。如果MFLEIF(EMAC_INTSTS[8])置位，且MFLEIEN和RXIEN(EMAC_INTEN[0])都使能了，EMAC将触发RX中断到CPU。如果MFLEIEN或RXIEN(EMAC_INTEN[0])关闭了，即使MFLEIF(EMAC_INTSTS[8])置位也不会触发RX中断。 0=禁止MFLEIF(EMAC_INTSTS[8])触发RX中断. 1=使能MFLEIF(EMAC_INTSTS[8])触发RX中断.
[7]	**MPCOVIEN**	**丢包计数器溢出中断使能位** MPCOVIEN 控制着 MPCOVIF(EMAC_INTSTS[7]) 中断的发生。如果MPCOVIF(EMAC_INTSTS[7])置位，且MPCOVIEN和RXIEN(EMAC_INTEN[0])都使能了，EMAC将触发RX中断到CPU。如果MPCOVIEN或RXIEN(EMAC_INTEN[0])关闭了，即使MPCOVIF(EMAC_INTSTS[7])置位也不会触发RX中断。 0=禁止MPCOVIF(EMAC_INTSTS[7])触发RX中断. 1=使能MPCOVIF(EMAC_INTSTS[7])触发RX中断.
[6]	**RPIEN**	**短包中断使能位** RPIEN控制着RPIF(EMAC_INTSTS[6])中断的发生。RPIF(EMAC_INTSTS[6])置位，且RPIEN和RXIEN(EMAC_INTEN[0])都使能了，EMAC将触发RX中断到CPU。如果RPIEN或RXIEN(EMAC_INTEN[0])关闭了，即使RPIF(EMAC_INTSTS[6])置位也不会触发RX中断。 0=禁止RPIF(EMAC_INTSTS[6])触发RX中断. 1=使能RPIF(EMAC_INTSTS[6])触发RX中断.
[5]	**ALIEIEN**	**对齐错误中断使能位** ALIEIEN控制着ALIEIF(EMAC_INTSTS[5]) 中断的发生。如果 ALIEIF(EMAC_INTSTS[5])置位，且ALIEIEN和RXIEN(EMAC_INTEN[0])都使能了，EMAC将触发RX中断到CPU。如果ALIEIEN或RXIEN(EMAC_INTEN[0])关闭了，即使ALIEIF(EMAC_INTSTS[5])置位也不会触发RX中断。 0=禁止ALIEIF(EMAC_INTSTS[5])触发RX中断. 1=使能ALIEIF(EMAC_INTSTS[5])触发RX中断.
[4]	**RXGDIEN**	**接收正常中断使能位** RXGDIEN 控制着 RXGDIF(EMAC_INTSTS[4]) 中断的发生。如果RXGDIF(EMAC_INTSTS[4])置位，且RXGDIEN和RXIEN(EMAC_INTEN[0])都使能了，EMAC将触发RX中断到CPU。如果RXGDIEN或RXIEN(EMAC_INTEN[0])关闭了，即使RXGDIF(EMAC_INTSTS[4])置位也不会触发RX中断。 0=禁止RXGDIF(EMAC_INTSTS[4])触发RX中断. 1=使能RXGDIF(EMAC_INTSTS[4])触发RX中断.
[3]	**LPIEN**	**长包中断使能位** LPIEN控制着LPIF(EMAC_INTSTS[3])中断的发生。如果LPIF(EMAC_INTSTS[3])置位，且LPIEN和RXIEN(EMAC_INTEN[0])都使能了，EMAC将触发RX中断到CPU。如果LPIEN或RXIEN(EMAC_INTEN[0])关闭了，即使LPIF(EMAC_INTSTS[3])置位也不会触发RX中断。 0=禁止LPIF(EMAC_INTSTS[3])触发RX中断. 1=使能LPIF(EMAC_INTSTS[3])触发RX中断.

[2]	**RXOVIEN**	**接收FIFO溢出中断使能位** RXOVIEN 控制着 RXOVIF(EMAC_INTSTS[2]) 中断的发生。如果RXOVIF(EMAC_INTSTS[2])置位，且RXOVIEN和RXIEN(EMAC_INTEN[0])都使能了，EMAC将触发RX中断到CPU。如果RXOVIEN或RXIEN(EMAC_INTEN[0])关闭了，即使RXOVIF(EMAC_INTSTS[2])置位也不会触发RX中断。 0=禁止RXOVIF(EMAC_INTSTS[2])触发RX中断. 1=使能RXOVIF(EMAC_INTSTS[2])触发RX中断.
[1]	**CRCEIEN**	**CRC错误中断使能位** CRCEIEN 控制着 CRCEIF(EMAC_INTSTS[1]) 中断的发生。如果CRCEIF(EMAC_INTSTS[1])置位，且CRCEIEN和RXIEN(EMAC_INTEN[0])都使能了，EMAC将触发RX中断到CPU。如果CRCEIEN或RXIEN(EMAC_INTEN[0])关闭了，即使CRCEIF(EMAC_INTSTS[1])置位也不会触发RX中断。 0=禁止CRCEIF(EMAC_INTSTS[1])触发RX中断. 1=使能CRCEIF(EMAC_INTSTS[1])触发RX中断.
[0]	**RXIEN**	**接收中断使能位** RXIEN控制着RX中断的发生 如果使能了RXIEN且RXIF(EMAC_INTSTS[0])置位，EMAC将产生RX中断到CPU。如果RXIEN未使能，即使EMAC_INTSTS[15:1]中某一位置且对应的EMAC_INTEN已使能，都不会产生RX中断。也就是说，软件如果想要使能RX中断，必须要先使能该位。想要关闭TRX中断时，禁止该位即可。 0=RXIF(EMAC_INTSTS[0])被屏蔽，禁止RX中断的发生 1=RXIF(EMAC_INTSTS[0])未屏蔽，允许RX中断的发生

MAC中断状态寄存器(EMAC_INTSTS)

EMAC_INTSTS保存EMAC的状态，如帧发送/接收状态和内部FIFO状态。EMAC_INTSTS的状态将会触发TX或RX中断。EMAC_INTSTS是写1清0的寄存器，写1到对应位即可清0对应状态和中断。

寄存器	偏移量	R/W	描述	复位值
EMAC_INTSTS	EMAC_BA+0x0B0	R/W	MAC中断状态寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved			TSALMIF	Reserved			TXBEIF
23	22	21	20	19	18	17	16
TDUIF	LCIF	TXABTIF	NCSIF	EXDEFIF	TXCPIF	TXUDIF	TXIF
15	14	13	12	11	10	9	8
WOLIF	CFRIF	Reserved		RXBEIF	RDUIF	DENIF	MFLEIF
7	6	5	4	3	2	1	0
MPCOVIF	RPIF	ALIEIF	RXGDIF	LPIF	RXOVIF	CRCEIF	RXIF

位	描述	
[31:29]	**Reserved**	保留
[28]	**TSALMIF**	**时间戳报警中断** TSALMIF置位表示EMAC_TSSEC寄存器的值等与EMAC_ALMSEC寄存器的值，EMAC_TSSUBSEC寄存器的值等与EMAC_TSMLSR寄存器的值 如果TSALMIF置位，且TSALMIEN(EMAC_INTEN[28])使能，TXIF将为高 写1到该位，清除TSALMIF的状态. 0=EMAC_TSSEC寄存器的值不等与EMAC_ALMSEC寄存器的值，或EMAC_TSSUBSEC寄存器的值不等与EMAC_TSMLSR寄存器的值 1=EMAC_TSSEC寄存器的值等与EMAC_ALMSEC寄存器的值，且EMAC_TSSUBSEC寄存器的值等与EMAC_TSMLSR寄存器的值
[27:25]	**Reserved**	保留
[24]	**TXBEIF**	**发送总线错误中断** TXBEIF置位表示在包发送的过程中，EMAC通过TXDMA读取系统内存的时候，内存控制器返回了错误的响应。TXBEIF状态为高时，建议复位EMAC。 如果TXBEIF置位，且TXBEIEN(EMAC_INTEN[24])使能，TXIF将为高 写1到该位，清除TXBEIF的状态. 0=无错误响应 1=接收到错误的响应

[23]	**TDUIF**	**发送描述符不可用中断** TDUIF置位表示包传输过程需要的TX描述符不可用，TXDMA处于关闭状态。一旦TXDMA进入了关闭状态，软件必须在获得了可用TX描述符后写一条命令到TSDR寄存器使TXDMA离开关闭状态。 如果TDUIF置位，且TDUIEN(EMAC_INTEN[23])使能，TXIF将为高 写1到该位，清除TDUIF的状态. 0=TX描述符可用 1=TX描述符不可用
[22]	**LCIF**	**迟到冲突中断** LCIF置位表示在64字节的冲突窗口外发生了新的冲突。即，在64字节的帧发送出去后产生了冲突。迟到冲突检测仅在EMAC半双工模式下有效。 如果LCIF置位，且LCIEN(EMAC_INTEN[22])使能，TXIF将为高 写1到该位，清除LCIF的状态. 0=在64字节的冲突窗口外无新的冲突 1=在64字节的冲突窗口外发生了新的冲突
[21]	**TXABTIF**	**发送停止中断** TXABTIF置位表示在传输过程中发生了连续的16个冲突，接着传输过程会因此停止。该功能仅在EMAC半双工模式下有效 如果TXABTIF置位，且TXABTIEN(EMAC_INTEN[21])使能，TXIF将为高 写1到该位，清除TXABTIF的状态. 0=在传输过程中没有发生连续的16个冲突 1=在传输过程中发生了连续的16个冲突
[20]	**NCSIF**	**无载波中断** NCSIF置高表示MIII/F信号CRS在包发送的开始不正常。该功能仅在EMAC半双工模式下有效 如果NCSIF置位，且NCSIEN(EMAC_INTEN[20])使能，TXIF将为高 写1到该位，清除NCSIF的状态. 0=CRS信号正常 1=CRS信号在包发送的开始不正常
[19]	**EXDEFIF**	**延期超时中断** EXDEFIF置位表示在100Mbps时，发送时帧等待发送超过了0.32768ms.或者在10Mbps时，等待超过3.2768ms.延期超时检测仅在MCMDR寄存器位NODEF(EMAC_CTL[9])关闭，EMAC工作在半双工模式下有效。 如果EXDEFIF置位，且EXDEFIEN(EMAC_INTEN[19])使能，TXIF将为高 写1到该位，清除EXDEFIF的状态. 0=未发生延期超时 1=在100Mbps时，发送时帧等待发送超过了0.32768ms.或者在10Mbps时，等待超过3.2768ms.
[18]	**TXCPIF**	**发送完成中断** TXCPIF置位表示发送正确完成 如果TXCPIF置位，且TXCPIEN(EMAC_INTEN[18])使能，TXIF将为高 写1到该位，清除TXCPIF的状态. 0=发送未完成. 1=发送已完成

[17]	**TXUDIF**	**发送FIFO下溢中断** TXUDIF置位表示发送过程中发生了发送FIFO下溢.当TXFIFO下溢发生，EMAC会自动重新发送包而不需要软件干预。如果TXFIFO频繁发生，建议修改TXFIFO的阈值控制寄存器FFTCR的TXFIFOTH位到更高的级别 如果TXUDIF置位，且TXUDIEN(EMAC_INTEN[17])使能，TXIF将为高 写1到该位，清除TXUDIF的状态. 0=传输过程中未发生TXFIFO下溢事件 1=传输过程中发生了TXFIFO下溢事件
[16]	**TXIF**	**发送中断** TXIF代表TX中断的状态 如果TXIF置位且对应的使能位TXIEN(EMAC_INTEN[16])也同时置位，表示EMAC产生了一个TX中断到CPU。如果TXIF置位但TXIEN(EMAC_INTEN[16])没有，则不会有中断发生 TXIF的值是EMAC_INTSTS[28:17]和EMAC_INTEN[28:17]的值逻辑“与”后的各个位做逻辑“或”的结果。即，如果EMAC_INTSTS[28:17]某一位为高且EMAC_INTEN[28:17]对应的使能位也使能了，TXIF就会置位。由于TXIF是逻辑“或”的结果，清EMAC_INTSTS[28:17]会清0TXIF。 0=EMAC_INTSTS[28:17]所有位都未置位，EMAC_INTEN[28:17]所有位都未使能 1=EMAC_INTSTS[28:17]至少有一位置位且对应的EMAC_INTEN[28:17]位使能了
[15]	**WOLIF**	**LAN口唤醒中断** WOLIF置位表示EMAC收到了魔术封包。该位仅在系统处于掉电模式下且WOLEN置位时有效 如果WOLIF置位，且WOLIEN(EMAC_INTEN[15])使能，RXIF将为高 写1到该位，清除WOLIF的状态. 0=EMAC未收到魔术封包 1=EMAC收到了魔术封包
[14]	**CFRIF**	**控制帧接收中断** CFRIF置位表示EMAC收到了一个流控帧。该位仅在全双工模式下有效 如果CFRIF置位，且CFRIEN(EMAC_INTEN[14])使能，RXIF将为高 写1到该位，清除CFRIF的状态. 0=EMAC未收到流控帧 1=EMAC收到了流控帧
[13:12]	**Reserved**	保留
[11]	**RXBEIF**	**接收总线错误中断** RXBEIF置位表示在包发送的过程中，EMAC通过RXDMA读取系统内存的时候，内存控制器返回了错误的响应。RXBEIF状态为高时，建议复位EMAC。 如果RXBEIF置位，且RXBEIEN(EMAC_INTEN[11])使能，RXIF将为高 写1到该位，清除RXBEIF的状态. 0=无错误响应 1=接收到错误的响应

[10]	**RDUIF**	**接收描述符不可用中断** RDUIF置位表示包接收过程需要的RX描述符不可用，RXDMA处于关闭状态。一旦RXDMA进入了关闭状态，软件必须在获得了可用RX描述符后写一条命令到RSDR寄存器使RXDMA离开关闭状态。 如果RDUIF置位，且RDUIEN(EMAC_INTEN[10])使能，RXIF将为高 写1到该位，清除RDUIF的状态. 0=RX描述符可用 1=RX描述符不可用
[9]	**DENIF**	**DMA早期通知中断** DENIF置位表示EMAC收到了输入包的长度字段 如果DENIF置位，且DENIENI(EMAC_INTEN[9])使能，RXIF将为高 写1到该位，清除DENIF的状态. 0=EMAC未收到输入包的长度字段 1=EMAC收到了输入包的长度字段
[8]	**MFLEIF**	**超出最大帧长度中断** MFLEIF置位表示输入包的长度超过了DMARFC寄存器里配置的长度，该包会被丢弃 如果MFLEIF置位，且MFLEIEN(EMAC_INTEN[8])使能，RXIF将为高 写1到该位，清除MFLEIF的状态. 0=输入包的长度未超过DMARFC寄存器里配置的长度 1=输入包的长度超过了DMARFC寄存器里配置的长度
[7]	**MPCOVIF**	**丢包计数器溢出中断** MPCOVIF置位表示丢包计数器MPCNT溢出. 如果MPCOVIF置位，且MPCOVIEN(EMAC_INTEN[7])使能，RXIF将为高 写1到该位，清除MPCOVIF的状态. 0=丢包计数器MPCNT未溢出. 1=丢包计数器MPCNT溢出
[6]	**RPIF**	**短包中断** RPIF置位表示会丢弃长度少于64字节的输入包。如果ARP(EMAC_CTL[2])置位，短包会被视作正常包，RPIF亦不会置位 如果RPIF置位，且RPIEN(EMAC_INTEN[6])使能，RXIF将为高 写1到该位，清除RPIF的状态. 0=输入包为短包，或软件希望接收短包. 1=丢弃为短包的输入包
[5]	**ALIEIF**	**对齐错误中断** ALIEIF置位表示输入帧的长度不是整字节的 如果ALIEIF置位，且ALIEIEN(EMAC_INTEN[5])使能，RXIF将为高 写1到该位，清除ALIEIF的状态. 0=输入帧的长度是整字节的 1=输入帧的长度不是整字节的

[4]	**RXGDIF**	**接收正常中断** RXGDIF置位表示帧正常接收 如果RXGDIF置位，且RXGDIEN(EAMC_MIEN[4])使能，RXIF将为高 写1到该位，清除RXGDIF的状态. 0=帧接收不正常 1=帧正常接收
[3]	**LPIF**	**长包中断** LPIF置位表示丢弃输入包的长度超过1518个字节的长包。如果ALP(EMAC_CTL[1])置位，长包将被视作正常包，LPIF亦不会置位。 如果LPIF置位，且LPIEN(EMAC_INTEN[3])使能，RXIF将为高 写1到该位，清除LPIF的状态. 0=输入包为长包，或软件希望接收长包. 1=丢弃为长包的输入包
[2]	**RXOVIF**	**接收FIFO溢出中断** RXOVIF置位表示RXFIFO在包的接收过程中发生了溢出。当RXFIFO溢出发生，EMAC会丢弃当前接收到的包。如果RXFIFO频繁溢出，建议修改RXFIFO的阈值控制寄存器FFTCR的RXFIFOTH位到更高的级别 如果RXOVIF置位，且RXOVIEN(EMAC_INTEN[2])使能，RXIF将为高 写1到该位，清除RXOVIF的状态. 0=RXFIFO在包的接收过程中未发生溢出 1=RXFIFO在包的接收过程中发生了溢出
[1]	**CRCEIF**	**CRC错误中断** CRCEIF置位表示输入包有CRC错误，封包将会被丢弃。如果AEP(EMAC_CTL[4])置位，CRC错误的包将被视作正常包，CRCEIF亦不会置位。 如果CRCEIF置位，且CRCEIEN(EMAC_INTEN[1])使能，RXIF将为高 写1到该位，清除CRCEIF的状态. 0=输入包有CRC正常 1=输入包有CRC错误
[0]	**RXIF**	**接收中断** RXIF代表RX中断的状态 如果RXIF置位且对应的使能位RXIEN(EMAC_INTEN[0])也同时置位，表示EMAC产生了一个RX中断到CPU。如果RXIF置位但RXIEN(EMAC_INTEN[0])没有，则不会有中断发生 RXIF的值是EMAC_INTSTS[15:1]和EMAC_INTEN[15:1]的值逻辑“与”后的各个位做逻辑“或”的结果。即，如果EMAC_INTSTS[15:1]某一位为高且EMAC_INTEN[15:1]对应的使能位也使能了，RXIF就会置位。由于RXIF是逻辑“或”的结果，清EMAC_INTSTS[15:1]会清0RXIF。 0=EMAC_INTSTS[15:1]所有位都未置位，EMAC_INTEN[15:1所有位都未使能 1=EMAC_INTSTS[15:1]至少有一位置位且对应的EMAC_INTEN[15:1]位使能了

MAC通用状态寄存器(EMAC_GENSTS)

EMAC_GENSTS存储着EMAC的状态。但是存储在EMAC_GENSTS的状态并不会触发中断。EMAC_GENSTS寄存器写清0，写1到对应位会清除其状态。

寄存器	偏移量	R/W	描述	复位值
EMAC_GENSTS	EMAC_BA+0x0B4	R/W	MACGe通用状态寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved			RPSTS	TXHALT	SQE	TXPAUSED	DEF
7	6	5	4	3	2	1	0
COLCNT				Reserved	RXFFULL	RXHALT	CFR

位	描述	
[31:13]	Reserved	保留
[12]	RPSTS	**远程暂停状态** RPSTS表示远程暂停计数器开始向下计数。 在EMAC控制器发送暂停帧成功后，远程暂停计数器开始向下计数。当该位置位，预示着EMAC控制器在向下计数完成前都不会在进行包传输。 0=远程暂停计数器计数完成 1=远程暂停计数器向下计数中
[11]	TXHALT	**发送停止** TXHALT置位表示由于软件禁止了TXON(EMAC_CTL[8])位，下一个正常包的发送过程将被停止 0=下一个正常包的发送过程正常进行 1=下一个正常包的发送过程将被停止
[10]	SQE	**信号质量错误(SQE)** SQE置位表示在10Mbps半双工模式下包传送结尾发现SQE错误。EMAC工作在10Mbps半双工模式下时，SQECHKEN(EMAC_CTL[17])使能，则会进行SQE错误检测。 0=包发送结尾未发现SQE错误. 1=包发送结尾发现SQE错误.
[9]	TXPAUSED	**发送暂停** TXPAUSED置位表示EMAC接收到暂停帧下个正常的包发送过程会被暂停。 0=下一个正常的包发送正常. 1=下一个正常的包发送被暂停.

[8]	**DEF**	**发送延期** DEF置位表示发送延期一次。DEF检测仅在半双工模式下有效 0=包发送未延期 1=包发送延期
[7:4]	**COLCNT**	**冲突数** COLCNT表示在包传递过程中发生的连续冲突数。如果传输时连续冲突数达16个，COLCNT归0x0，TXABTIF位置位。
[3]	**Reserved**	保留
[2]	**RXFFULL**	**RXFIFO满** RXFFULL表示4个64字节的包存储在RXFIFO中导致其满了，之后的输入包将会被丢弃 0=RXFIFO未满 1=RXFIFO满了，之后的输入包将会被丢弃
[1]	**RXHALT**	**接收停止** RXHALT置位表示由于软件禁止了寄存器MCMDR的RXON位，下一个正常的包接收过程会被停止 0=下一个正常的包接收过程正常进行 1=下一个正常的包接收过程会被停止
[0]	**CFR**	**控制帧接收** CFRIF置位表示EMAC接收到了一个流控帧。该位仅在全双工模式下有效 0=EMAC未接收到流控帧 1=EMAC接收到了流控帧

丢包计数寄存器(EMAC_MPCNT)

EMAC_MPCNT保存由于各种接收错误而产生的丢包的个数。EMAC_MPCNT是一个读清除的寄存器。另外，软件可以写初始值到EMAC_MPCNT，丢包计数器会从设置的值开始计数。如果丢包计数器溢出，MPCOVIF(EMAC_INTSTS[7])会被置位。

寄存器	偏移量	R/W	描述	复位值
EMAC_MPCNT	EMAC_BA+0x0B8	R/W	丢包计数寄存器	0x0000_7FFF

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
MPCNT							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
MPCNT							

位	描述	
[31:16]	**Reserved**	保留
[15:0]	**MPCNT**	**丢包计数值** MPCNT保存由于各种接收错误而产生的丢包的个数。以下错误都会使计数值增加： 1.输入包使RXFIFO溢出 2.由于RXON关闭而使输入包被丢弃 3.输入包发生CRC错误

MAC接收到的暂停帧计数寄存器(EMAC_RPCNT)

EMAC支持暂停帧的接收与识别。如果EMAC接收到了一个暂停帧，暂停帧的OPERAND字段被解析存储到EMAC_RPCNT寄存器。EMAC_RPCNT寄存器在EMAC暂停过程中一直保持不变。EMAC_RPCNT只读，写操作对该寄存器无效。

寄存器	偏移量	R/W	描述	复位值
EMAC_RPCNT	EMAC_BA+0x0BC	R	MAC接收到的暂停帧计数寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
RPCNT							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
RPCNT							

位	描述	
[31:16]	**Reserved**	保留
[15:0]	**RPCNT**	**MAC暂停计数** RPCNT保存暂停帧的OPERAND字段。即，EMAC需要暂停多少个时间片(512位时间)

DMA接收帧状态寄存器(EMAC_FRSTS)

EMAC_FRSTS用来存储每个输入帧的长度字段。

寄存器	偏移量	R/W	描述	复位值
EMAC_FRSTS	EMAC_BA+0x0C8	R/W	DMA接收帧状态寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
RXFLT							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
RXFLT							

位	描述	
[31:16]	**Reserved**	保留
[15:0]	**RXFLT**	**接收帧长度** RXFLT用来存储每个输入帧的长度字段。如果使能了DENIEN(EMAC_INTEN[9])且已收到输入帧的长度字段，DENIF(EMAC_INTSTS[9])会被置位，并触发中断。RXFLT会存储该长度字段。

当前发送描述符起始地址寄存器(EMAC_CTXDSA)

寄存器	偏移量	R/W	描述	复位值
EMAC_CTXDSA	EMAC_BA+0x0CC	R	当前发送描述符起始地址寄存器	0x0000_0000

31	30	29	28	27	26	25	24
CTXDSA							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
CTXDSA							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
CTXDSA							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
CTXDSA							

位	描述	
[31:0]	**CTXDSA**	**当前发送描述符起始地址** CTXDSA保存TXDMA当前处理的TX描述符的起始地址。CTXDSQA只读，写该寄存器无效。

当前发送缓存起始地址寄存器(EMAC_CTXBSA)

寄存器	偏移量	R/W	描述	复位值
EMAC_CTXBSA	EMAC_BA+0x0D0	R	当前发送缓存起始地址寄存器	0x0000_0000

31	30	29	28	27	26	25	24
CTXBSA							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
CTXBSA							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
CTXBSA							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
CTXBSA							

位	描述	
[31:0]	**CTXBSA**	**当前发送缓存起始地址** CTXDSA保存当前TXDMA发送缓存起始地址。CTXBSA只读，写该寄存器无效。

当前接收描述符起始地址寄存器(EMAC_CRXDSA)

寄存器	偏移量	R/W	描述	复位值
EMAC_CRXDSA	EMAC_BA+0x0D4	R	当前接收描述符起始地址寄存器	0x0000_0000

31	30	29	28	27	26	25	24
CRXDSA							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
CRXDSA							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
CRXDSA							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
CRXDSA							

位	描述	
[31:0]	**CRXDSA**	**当前接收描述符起始地址** CRXDSA存储当前RXDMA接收描述符起始地址。CRXDSA只读，写该寄存器无效。

当前接收缓存起始地址寄存器(EMAC_CRXBSA)

寄存器	偏移量	R/W	描述	复位值
EMAC_CRXBSA	EMAC_BA+0x0D8	R	当前接收缓存起始地址寄存器	0x0000_0000

31	30	29	28	27	26	25	24
CRXBSA							
23	22	21	20	19	18	17	16
CRXBSA							
15	14	13	12	11	10	9	8
CRXBSA							
7	6	5	4	3	2	1	0
CRXBSA							

位	描述	
[31:0]	CRXBSA	**当前接收缓存起始地址** CRXBSA保存RXDMA当前接收缓存起始地址。CRXBSA只读，写该寄存器无效。

时间戳控制寄存器(EMAC_TSCTL)

寄存器	偏移量	R/W	描述	复位值
EMAC_TSCTL	EMAC_BA+0x100	R/W	时间戳控制寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved		TSALMEN	Reserved	TSUPDATE	TSMODE	TSIEN	TSEN

位	描述	
[31:6]	**Reserved**	保留
[5]	**TSALMEN**	**时间戳报警使能位** 置位该位使能EMAC控制器在EMAC_TSSEC等于EMAC_ALMSEC和EMAC_TSSUBSEC等于EMAC_ALMSUBSEC时置位TSALMIF(EMAC_INTSTS[28])。 0=EMAC_TSSEC等于EMAC_ALMSEC和EMAC_TSSUBSEC等于EMAC_ALMSUBSEC时禁止报警. 1=EMAC_TSSEC等于EMAC_ALMSEC和EMAC_TSSUBSEC等于EMAC_ALMSUBSEC时使能报警.
[4]	**Reserved**	保留
[3]	**TSUPDATE**	**时间戳计数器时间更新使能位** 置位该位使能EMAC控制器增加寄存器EMAC_UPDSEC和EMAC_UPDSUBSEC的值到PTP时间戳计数器 0=无效 1=EMAC_UPDSEC 更新到 EMAC_TSSEC， EMAC_UPDSUBSEC 更新到 EMAC_TSSUBSEC.
[2]	**TSMODE**	**时间戳计数器精确更新使能位** 该位决定时间戳更新的模式 0=时间戳计数器粗略更新模式 1=时间戳计数器精确更新模式
[1]	**TSIEN**	**时间戳计数器初始化使能位** 置位该位使能EMAC控制器加载寄存器EMAC_UPDSEC和EMAC_UPDSUBSEC的值到PTP时间戳计数器 0=时间戳计数器初始化完成 1=使能时间戳计数器初始化

[0]	**TSEN**	**时间戳功能使能位** 该位控制是否使能符合IEEE1588PTP规范的时间戳功能 0=关闭符合IEEE1588PTP规范的时间戳功能 1=使能符合IEEE1588PTP规范的时间戳功能

时间戳秒计数器寄存器(EMAC_TSSEC)

寄存器	偏移量	R/W	描述	复位值
EMAC_TSSEC	EMAC_BA+0x110	R	时间戳秒计数器寄存器	0x0000_0000

31	30	29	28	27	26	25	24
SEC							
23	22	21	20	19	18	17	16
SEC							
15	14	13	12	11	10	9	8
SEC							
7	6	5	4	3	2	1	0
SEC							

位	描述	
[31:0]	SEC	**时间戳计数器秒寄存器** 该寄存器为64位的参考时间计数器的[63:32]位。当TSEN(EMAC_TSCTL[0])置位时，这32位是时间戳的秒部分。

时间戳亚秒计数器寄存器(EMAC_TSSUBSEC)

寄存器	偏移量	R/W	描述	复位值
EMAC_TSSUBSEC	EMAC_BA+0x114	R	时间戳亚秒计数器寄存器	0x0000_0000

31	30	29	28	27	26	25	24
SUBSEC							
23	22	21	20	19	18	17	16
SUBSEC							
15	14	13	12	11	10	9	8
SUBSEC							
7	6	5	4	3	2	1	0
SUBSEC							

位	描述	
[31:0]	SUBSEC	**时间戳计数器亚秒寄存器** 该寄存器为64位的参考时间计数器的[31:0]位。当TSEN(EMAC_TSCTL[0])置位时，这32位是时间戳的亚秒部分。

时间戳增量寄存器(EMAC_TSINC)

寄存器	偏移量	R/W	描述	复位值
EMAC_TSINC	EMAC_BA+0x118	R/W	时间戳增量寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
CNTINC							

位	描述	
[31:8]	**Reserved**	保留
[7:0]	**CNTINC**	**时间戳计数器增量** 时间戳计数器的增量 如果TSEN(EMAC_TSCTL[0])置位，EMAC增加EMAC_TSSUBSEC的值时，每次增加这里的8位值

时间戳加数寄存器(EMAC_TSADDEND)

寄存器	偏移量	R/W	描述	复位值
EMAC_TSADDEND	EMAC_BA+0x11C	R/W	时间戳加数寄存器	0x0000_0000

31	30	29	28	27	26	25	24
ADDEND							
23	22	21	20	19	18	17	16
ADDEND							
15	14	13	12	11	10	9	8
ADDEND							
7	6	5	4	3	2	1	0
ADDEND							

位	描述	
[31:0]	ADDEND	**时间戳加数寄存器** 该寄存器32位的累加器的值，该累加器用来触发EMAC_TSSUBSEC的增加。 如果TSEN(EMAC_TSCTL[0])和TSMODE(EMAC_TSCTL[2])都为1，EMAC累加器在每个HCLK都增加这个值。一旦累加器溢出，将会触发EMAC_TSSUBSEC增加，增加的值为保存在EMAC_TSINC的8位值。

时间戳更新秒寄存器(EMAC_UPDSEC)

寄存器	偏移量	R/W	描述	复位值
EMAC_UPDSEC	EMAC_BA+0x120	R/W	时间戳更新秒寄存器	0x0000_0000

31	30	29	28	27	26	25	24
SEC							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
SEC							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
SEC							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
SEC							

位	描述	
[31:0]	**SEC**	**时间戳更新秒寄存器** 当TSIEN(EMAC_TSCTL[1])为高，EMAC直接加载该寄存器32位值到EMAC_TSSEC。如果TSUPDATE(EMAC_TSCTL[3])为高，EMAC则把该寄存器的32位值累加到EMAC_TSSEC上。

时间戳更新亚秒寄存器(EMAC_UPDSUBSEC)

寄存器	偏移量	R/W	描述	复位值
EMAC_UPDSUBSEC	EMAC_BA+0x124	R/W	时间戳更新亚秒寄存器	0x0000_0000

31	30	29	28	27	26	25	24
SUBSEC							
23	22	21	20	19	18	17	16
SUBSEC							
15	14	13	12	11	10	9	8
SUBSEC							
7	6	5	4	3	2	1	0
SUBSEC							

位	描述	
[31:0]	SUBSEC	**时间戳更新亚秒** 当TSIEN(EMAC_TSCTL[1])为高，EMAC直接加载该寄存器32位值到EMAC_TSSUBSEC。如果TSUPDATE(EMAC_TSCTL[3])为高，EMAC则把该寄存器的32位值累加到EMAC_TSSUBSEC上。

时间戳报警秒寄存器(EMAC_ALMSEC)

寄存器	偏移量	R/W	描述	复位值
EMAC_ALMSEC	EMAC_BA+0x128	R/W	时间戳报警秒寄存器	0x0000_0000

31	30	29	28	27	26	25	24
SEC							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
SEC							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
SEC							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
SEC							

位	描述	
[31:0]	**SEC**	**时间戳报警秒寄存器** 时间戳计数器报警值的秒部分 该值仅在ALMEN(EMAC_TSCTL[5])为高时有效。如果ALMEN(EMAC_TSCTL[5])为高，EMAC_TSSEC等于EMAC_ALMSEC和EMAC_TSSUBSEC等于EMAC_ALMSUBSEC时，EMAC控制器会置位TSALMIF(EMAC_INTSTS[28])。

时间戳报警亚秒寄存器(EMAC_ALMSUBSEC)

寄存器	偏移量	R/W	描述	复位值
EMAC_ALMSUBSEC	EMAC_BA+0x12C	R/W	时间戳报警亚秒寄存器	0x0000_0000

31	30	29	28	27	26	25	24
SUBSEC							
23	22	21	20	19	18	17	16
SUBSEC							
15	14	13	12	11	10	9	8
SUBSEC							
7	6	5	4	3	2	1	0
SUBSEC							

位	描述	
[31:0]	SUBSEC	**时间戳报警亚秒寄存器** 时间戳计数器报警值的亚秒部分 该值仅在ALMEN(EMAC_TSCTL[5])为高时有效。如果ALMEN(EMAC_TSCTL[5])为高，EMAC_TSSEC等于EMAC_ALMSEC和EMAC_TSSUBSEC等于EMAC_ALMSUBSEC时，EMAC控制器会置位TSALMIF(EMAC_INTSTS[28])

6.18 智能卡主机接口(SC)

6.18.1 概述

智能卡接口控制器(SC控制器)基于ISO/IEC7816-3标准并完全兼容PC/SC规格。它还提供卡插入/移除的状态检测的功能。

6.18.2 特征

- 兼容ISO-7816-3T=0,T=1
- 兼容EMV2000
- 一个ISO-7816-3端口
- 接收和发送各4字节FIFO
- 可编程的发送时钟频率
- 可编程的接收器缓存触发水平
- 可编程的块保护时间选择(11ETU~267ETU)
- 一个24-位和两个8-位计数器用于请求应答(AnswertoRequest(ATR))和等待时间处理
- 支持自动反向功能
- 支持传送器和接收器错误重试和错误数目限制功能
- 支持硬件激活序列处理，可配置上电到时钟产生的时间间隔
- 支持硬件热复位序列处理
- 支持硬件释放序列处理
- 支持当检测到卡移除时，硬件自动释放序列
- 支持UART模式
  - 全双工异步通信
  - 独立的接收和发送各4字节FIFO
  - 支持可编程的波特率发生器
  - 支持可编程的接收器缓存触发水平
  - 通过设置EGT(SC_EGT[7:0])，可编程的发送数据延时时间(最后一个停止位从TX-FIFO离开到释放的时间)。
  - 可编程的偶，奇或者无校验位的生成和检测
  - 可编程的停止位，1位或2位的停止位生成

6.18.3 框图

SC的时钟控制和框图如下图所示。SC控制器包含2个完全异步的时钟区域，PCLK和引擎时钟。需要注意的是，PCLK需要大于等于引擎时钟频率。

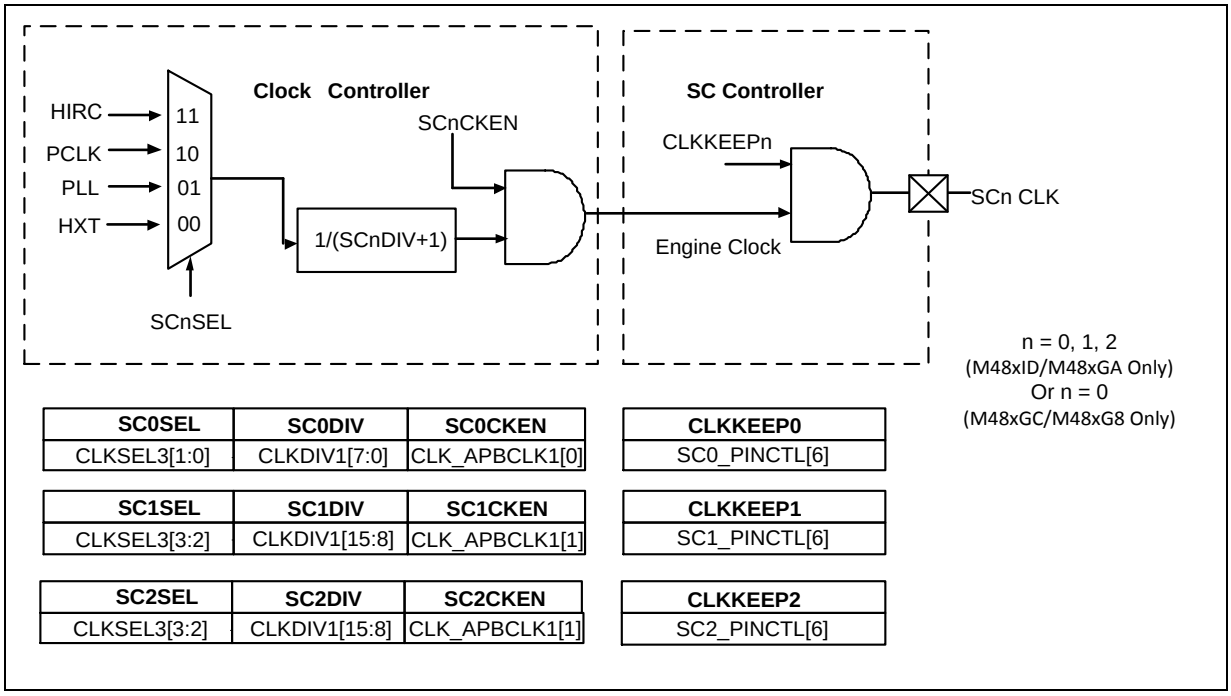


图 6.18-1SC 时钟控制框图(时钟控制器含 7 位的预分频计数器)

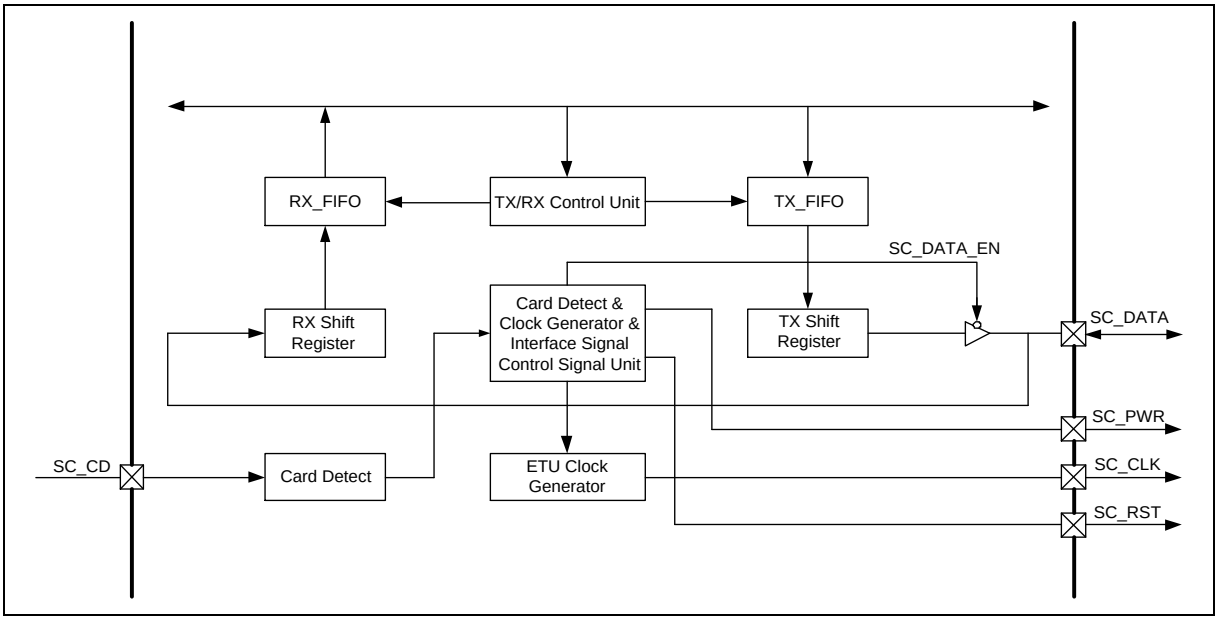


图 6.18-2SC 控制器框图

6.18.4 基本配置

SC主控制器管脚描述如下所示：

管脚	类型	描述
SCn_DATA	双向的	SC主控制器数据
SCn_CD	输入	SC主控制器卡检测
SCn_PWR	输出	SC主控制器卡电源开关
SCn_CLK	输出	SC主控制器时钟
SCn_RST	输出	SC主控制器复位

表 6.18-1SC 主控制器管脚描述

UART模式管脚描述如下所示：

管脚	类型	描述
SCn_DATA	输入	UART接收数据
SCn_CLK	输出	UART发送数据

表 6.18-2UART 管脚描述

6.18.4.1 SC0基本配置

- 时钟源配置
  - 在寄存器SC0SEL(CLK_CLKSEL3[1:0])配置SC0的外设时钟源
  - 在寄存器SC0DIV(CLK_CLKDIV1[7:0])选择SC0外设时钟分频数
  - 在寄存器SC0CKEN(CLK_APBCLK1[0])使能SC0的外设时钟
- 复位配置
  - 在寄存器SC0RST(SYS_IPRST2[0])复位SC0控制器
- 管脚配置

组	管脚名	GPIO	MFP
SC0	SC0_CLK	PB.5	MFP9
		PF.6	MFP3
		PA.0	MFP6
		PE.2	MFP6
	SC0_DAT	PB.4	MFP9
		PF.7	MFP3
		PA.1	MFP6
		PE.3	MFP6

	SC0_PWR	PB.2	MFP9
		PF.9	MFP3
		PA.3	MFP6
		PE.5	MFP6
	SC0_RST	PB.3	MFP9
		PF.8	MFP3
		PA.2	MFP6
		PE.4	MFP6
	SC0_nCD	PC.12	MFP9
		PF.10	MFP3
		PA.4	MFP6
		PE.6	MFP6

6.18.4.2 SC1基本配置

- 时钟源配置
  - 在寄存器SC1SEL(CLK_CLKSEL3[3:2])配置SC1的外设时钟源
  - 在寄存器SC1DIV(CLK_CLKDIV1[23:16])选择SC1外设时钟分频数
  - 在寄存器SC1CKEN(CLK_APBCLK1[2])使能SC1的外设时钟
- 复位配置
  - 在寄存器SC1RST(SYS_IPRST2[1])复位SC1控制器
- 管脚配置

组	管脚名	GPIO	MFP
SC1	SC1_CLK	PC.0	MFP5
		PD.4	MFP8
		PG.8	MFP4
		PB.12	MFP3
	SC1_DAT	PC.1	MFP5
		PD.5	MFP8
		PG.7	MFP4
		PB.13	MFP3
	SC1_PWR	PC.3	MFP5
		PD.7	MFP8
		PG.5	MFP4
		PB.15	MFP3

	SC1_RST	PC.2	MFP5
		PD.6	MFP8
		PG.6	MFP4
		PB.14	MFP3
	SC1_nCD	PC.4	MFP5
		PD.3	MFP8
		PD.14	MFP4
		PC.14	MFP3

6.18.4.3 SC2基本配置

- 时钟源配置
  - 在寄存器SC2SEL(CLK_CLKSEL3[5:4])配置SC2的外设时钟源
  - 在寄存器SC2DIV(CLK_CLKDIV1[23:16])选择SC2外设时钟分频数
  - 在寄存器SC2CKEN(CLK_APBCLK1[2])使能SC2的外设时钟
- 复位配置
  - 在寄存器SC2RST(SYS_IPRST2[2])复位SC2控制器
- 管脚配置

组	管脚名	GPIO	MFP
SC2	SC2_CLK	PA.8	MFP3
		PA.6	MFP6
		PD.0	MFP7
		PA.15	MFP7
		PE.0	MFP4
	SC2_DAT	PA.9	MFP3
		PA.7	MFP6
		PD.1	MFP7
		PA.14	MFP7
		PE.1	MFP4
	SC2_PWR	PA.11	MFP3
		PC.7	MFP6
		PD.3	MFP7
		PA.12	MFP7
		PH.8	MFP4
	SC2_RST	PA.10	MFP3

		PC.6	MFP6
		PD.2	MFP7
		PA.13	MFP7
		PH.9	MFP4
	SC2_nCD	PC.13	MFP3
		PA.5	MFP6
		PD.13	MFP7
		PH.10	MFP4

6.18.5 功能描述

智能卡接口扮演的就是一个半双工异步通信端口的角色，它的数据格式如下图所示的10个连续位组成

。

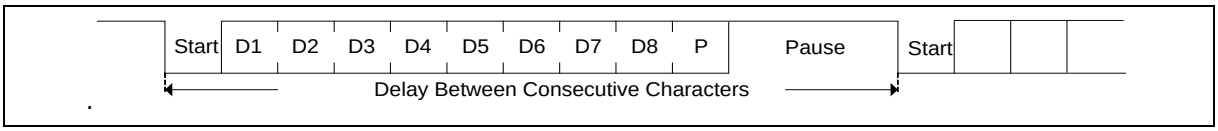


图 6.18-3SC 数据格式

智能卡接口控制器支持硬件激活，热复位和释放序列。后面会对硬件激活，热复位和释放序列作详细说明。

6.18.5.1 智能卡管脚配置

智能卡接口的寄存器状态可以通过读取以下寄存器获得：

1. SCn_RST是一个输出管脚。读RSTSTS(SCn_PINCTL[18])获得对应状态。写0或1到RSTEN(SCn_PINCTL[1])可以拉低或拉高该管脚。

2. SCn_PWR是一个输出管脚。读PWRSTS(SCn_PINCTL[17])获得对应状态。写0或1到PWREN(SCn_PINCTL[0])可以拉低或拉高该管脚。PWRINV(SCn_PINCTL[11])寄存器控制这该管脚的反转功能。用户在智能卡使能前需要先设置PWRINV(SCn_PINCTL[11])寄存器)。

3. SCn_DATA是一个双向管脚。智能卡接收数据时，DATASTS(SCn_PINCTL[16])存储对应的管脚状态。写0或1到SCDATA(SCn_PINCTL[9])可以拉低或拉高该管脚。

4. SCn_CLK是一个输出管脚。它输出智能卡时钟SCnCLK。写0或1到CLKKEEP(SCn_PINCTL[6])寄存器可以关闭或使能该管脚。CSTOPLV(SCn_PINCTL[5])则决定了该管脚在时钟关闭输出时处于低电平还是高电平。

5. SCn_CD(智能卡检测管脚)表示当前卡片的插入状态。读CDPINSTS(SC_CDPINSTS[13]).寄存器获得当前SCn_CD管脚的状态。SCn_CD相关的功能设置寄存器有CDLV(SC_CTL[26]),CDDBSEL(SC_CTL[25:24]),CDIF(SC_INTSTS[7]),CDIEN(SC_INTEN[7])。

6. CDLV(SC_CTL[26])决定了什么样的电平变化会被视作卡片插入。CDDBSEL(SC_CTL[25:24])为消抖周期数。当卡片状态寄存器CINSERT(SCn_STATUS[12])或CREMOVE(SCn_STATUS[11])发生变化时，CDIF会被置位。如果使能了CDIEN，CDIF置1时，SC会产生中断到CPU。建议在使能SC之前先配置该管脚。

6.18.5.2 激活，热复位和释放序列

激活

如图6.18-4是激活序列：

1. 写0到RSTEN(SCn_PINCTL[1])，设置SCn_RST为低电平。然后，等待SYNC(SCn_PINCTL[31])清0。.
2. 写1到PWREN(SCn_PINCTL[0])设置SCn_PWR为高电平。写1到SCDATA(SCn_PINCTL[9])设置SCn_DATA为高电平接收模式。等待SYNC(SCn_PINCTL[31])清0.
3. 写1到CLKKEEP(SCn_PINCTL[6])使能SCn_CLK。等待SYNC(SCn_PINCTL[31])
4. 写1到RSTEN(SCn_PINCTL[1])释放SCn_RST为高电平。等待SYNC(SCn_PINCTL[31])清0

激活序列可以有两种方法控制，其过程如下所示：

- 软件控制时序

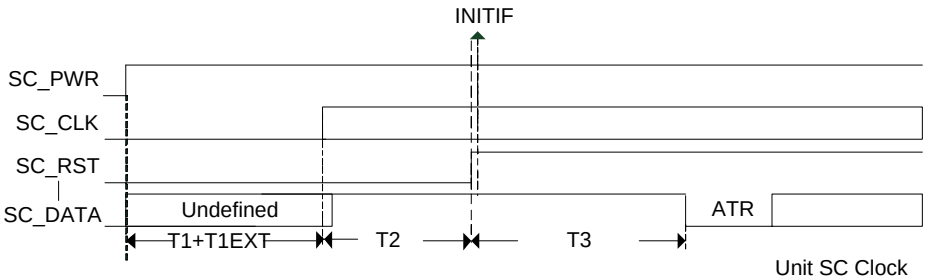
  软件可以通过设置SCn_PINCTL和SCn_TMRCTLx(x=0,1,2)来处理激活序列。通过设置SCn_PINCTL，可以分别设置SCn_PWR,SCn_CLK,SCn_RST和SCn_DATA的管脚状态。编程方法在激活序列中有描述。激活序列时序可以通过设置SCn_TMRCTLx(x=0,1,2)来控制。软件控制时序为用户在激活序列过程中提供了灵活的时序设置。

- 硬件时序控制：

  软件设置ACTEN(SCn_ALTCTL[3])为1，接口将通过硬件执行激活序列。SCn_PWR上电到SCn_CLK开始时间(T1)和SCn_CLK开始到SCn_RST复位的时间(T2)可以通过设置寄存器INITSEL(SC_ALTCTL[9:8])来选择。SCn_PWR上电到开始时间SCn_CLK的长度可以通过寄存器T1EXT(SCn_ACTCTL[4:0])设置。硬件控制时序为用户激活序列提供了一种简单的设置。在硬件激活的过程中，RX接收是关闭的无法接收任何数据。.

如下是由硬件产生的激活控制序列：

1. 通过设置INITSEL(SCn_ALTCTL[9:8])来设置激活时序
2. 通过设置TMRSEL(SCn_CTL[14:13])为'11'，来选择Timer0
3. 设置操作模式OPMODE(SCn_TMRCTL0[27:24])为'0011'，通过设置CNT(SCn_TMRCTL0[23:0])来提供应答请求(ATR)的值
4. 当硬件释放SCn_RST到高时，硬件将产生一个中断INITIF(SCn_INTSTS[8])到CPU，同时INITIEN(SCn_INTEN[8])置为1
5. 如果Timer0递减计数到0(从SCn_RST释放开始)，而且在此之前卡没有回应ATR，硬件将产生TMR0IF(SCn_INTSTS[3])中断到CPU。



Time	Comment
T1	SC_PWR to SC_CLK Start
T1EXT	Configurable cycles for T1
T2	SC_CLK Start to SC_RST Assert
T3	SC_CLK Start to ART Appear

INITSEL (SC_ALTCTL[9:8])	T1	T2
00	83.5	491
01	133	537
10	165	569
11	165	42060

Note : The values are measured by chip I / O pin and the real value will depend on system design

图 6.18-4SC 激活序列

热复位

如图6.18-5是热复位示意：

1. 通过设置RSTEN(SCn_PINCTL[1])为‘0’，设置SCn_RST为低。等待SYNC(SCn_PINCTL[31])清0
2. 通过设置SCDATA(SCn_PINCTL[9])为‘1’，设置SCn_DAT为高。等待SYNC(SCn_PINCTL[31])清0
3. 通过设置RSTEN(SCn_PINCTL[1])为‘1’，设置SCn_RST为高。等待SYNC(SCn_PINCTL[31])清0

热复位序列可以有两种方法来控制，其过程如下所示。

- 软件时序控制：

  软件通过设置SCn_PINCTL和SCn_TMRCTLx(x=0,1,2)来处理热复位序列。可以通过编程SCn_PINCTL来设置SCn_RST和SCn_DATA的引脚状态。也可以通过设置SCn_TMRCTLx(x=0,1,2)来控制热复位序列的时序。软件时序控制为用户热复位序列提供了灵活的时序设置。

- 硬件时序控制：

  软件通过设置WARSTEN(SCn_ALTCTL[4])为1，接口将通过硬件执行热复位序列。可以通过编程INITSEL(SCn_ALTCTL[9:8])来选择SCn_RST到SCn_DATA接收模式的时间(T4)和SCn_DATA接收模式到SCn_RST复位的时间(T5)。硬件时序控制为用户热复位序列提供了一种简单的设置。

如下是由硬件产生的热复位控制序列：

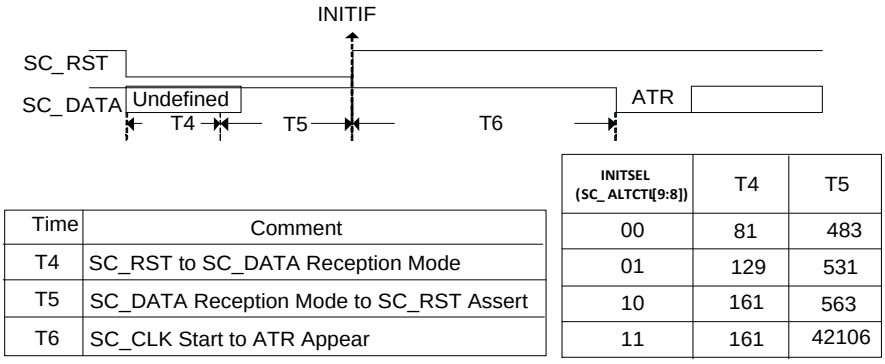
1. 通过设置INITSEL(SCn_ALTCTL[9:8])来设定热复位的时序。
2. 通过设置TMRSEL(SCn_CTL[14:13])为’11’，来选择Timer0

3. 设置操作模式OPMODE(SCn_TMRCTL0[27:24])为'0011'，和通过设置CNT(SCn_TMRCTL0[23:0])值来提供请求应答到请求的值

4. 通过设置CNTEN0(SCn_ALTCTL[5])和WARSTEN(SCn_ALTCTL[4])来启动计数

5. 当硬件释放SCn_RST为高时，硬件将同时产生INITIF(SCn_INTSTS[8])中断给CPU(寄存器INITIEN(SCn_INTEN[8]='1'))

6. 如果Timer0递减计数到0(从SCn_RST开始)，而且在那之前卡没有回应ATR，硬件将产生TMR0IF(SCn_INTSTS[3])中断给CPU。



Time	Comment
T4	SC_RST to SC_DATA Reception Mode
T5	SC_DATA Reception Mode to SC_RST Assert
T6	SC_CLK Start to ATR Appear

INITSEL (SC_ALTCTI[9:8])	T4	T5
00	81	483
01	129	531
10	161	563
11	161	42106

Unit: SC Clock

Note: This value is measured by chip IO pin and the real value will depend on system design

图 6.18-5SC 热复位序列

释放

图6.18-6是释放序列的示意：.

1. 通过设置RSTEN(SCn_PINCTL[1])为'0'，设置SCn_RST为低。等待SYNC(SCn_PINCTL[31])清0
2. 通过设置SCn_CLK(SCn_PINCTL[6])为'0'，关闭SCn_CLK。等待SYNC(SCn_PINCTL[31])清0
3. 通过设置SCn_DATA(SCn_PINCTL[9])为'0'，设置SCn_DATA为低。等待SYNC(SCn_PINCTL[31])清0
4. 通过设置PWREN(SCn_PINCTL[0])为'0'，释放SCn_PWR。等待SYNC(SCn_PINCTL[31])清0。

释放序列可以有两种方法来控制，其过程如下所示：

- 软件时序控制：

  软件通过设置SCn_PINCTL和SCn_TMRCTL0来处理释放序列。可以通过编程SCn_PINCTL来设置SCn_PWR,SCn_CLK,SCn_RST和SCn_DATA的引脚状态。还可以通过设置SCn_TMRCTL0来控制释放序列的时序。软件时序控制为用户释放序列提供了灵活的时序设置。

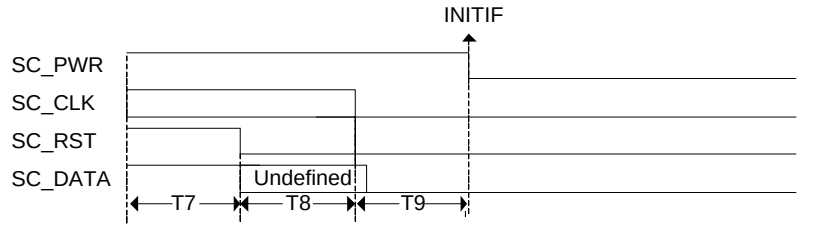
- 硬件时序控制：

  软件通过设置DACTEN(SCn_ALTCTL[2])为1，接口将通过硬件执行释放序列。可以通过配置INITSEL(SCn_ALTCTL[9:8])来选择释放触发到SCn_RST变低的时间(T7)，SCn_RST

变低到SCn_CLK停止的时间(T8)，和SCn_CLK停止到SCn_PWR停止的时间(T9)。硬件时序控制为用户释放序列提供了一种简单的设置。

如果INITIF(SCn_INTSTS[8])为1，当硬件释放SCn_PWR为低时，SC控制器会产生一个中断INITIF(SCn_INTSTS[8])到CPU。

当设置了卡移除检测(ADAC_CDEN(SCn_ALTCTL[11])=1)时，SC控制器也支持自动释放序列。



Time	Comment
T7	Deactivation Trigger to SC_RST Low
T8	SMC_RST Low to Stop SC_CLK
T9	Stop SC_CLK to Stop SC_PWR

INITSEL (SC_ALTCTL[9:8])	T7	T8	T9
00	97	83	87
01	145	131	135
10	177	163	167
11	177	163	167

Unit: SC Clock

图 6.18-6SC 释放序列

6.18.5.3 基本操作流程

智能卡的基本操作流程可以参考ISO7816-3&EMV协议里的规定。

如下是编程流程示意图：

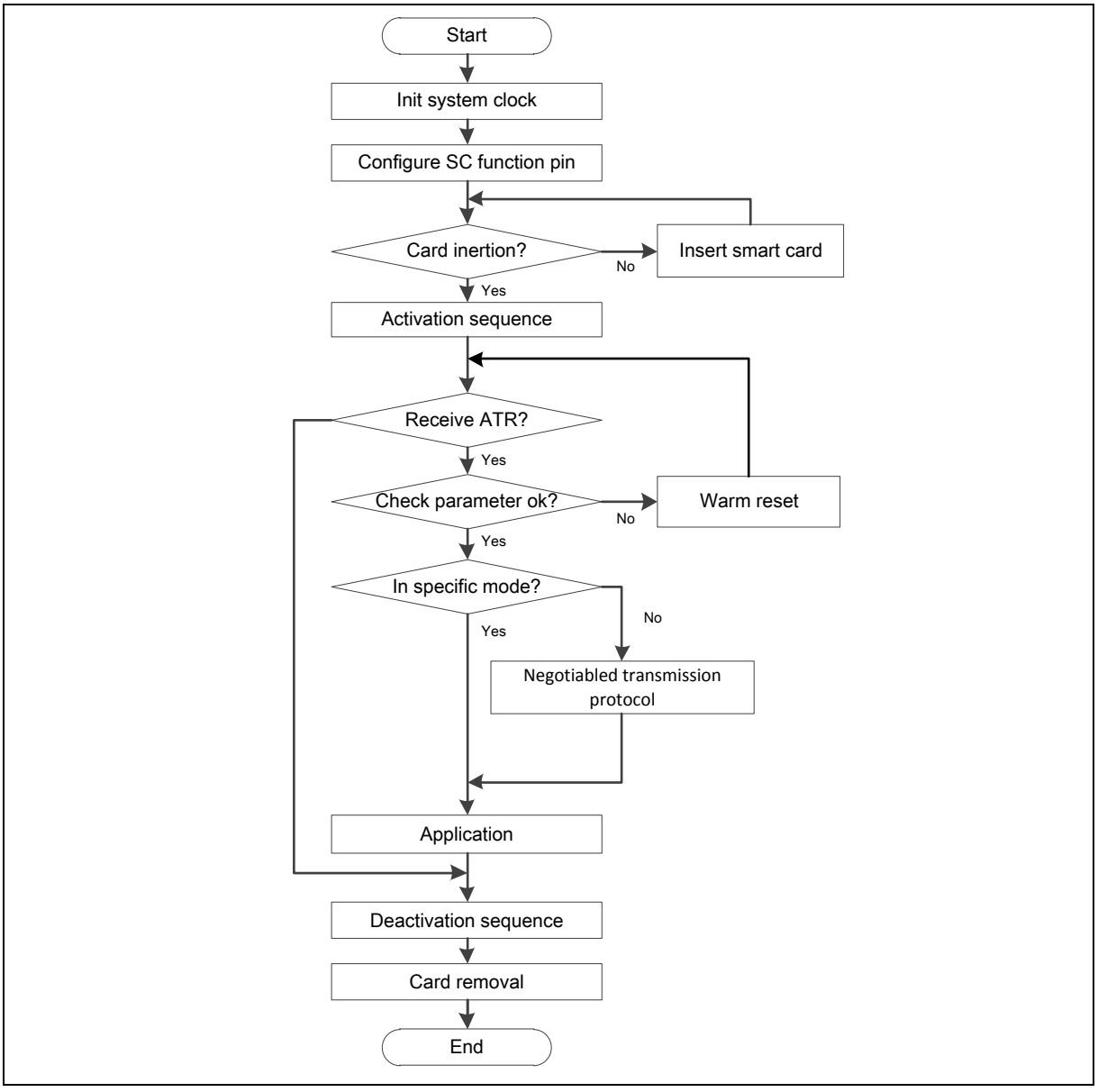
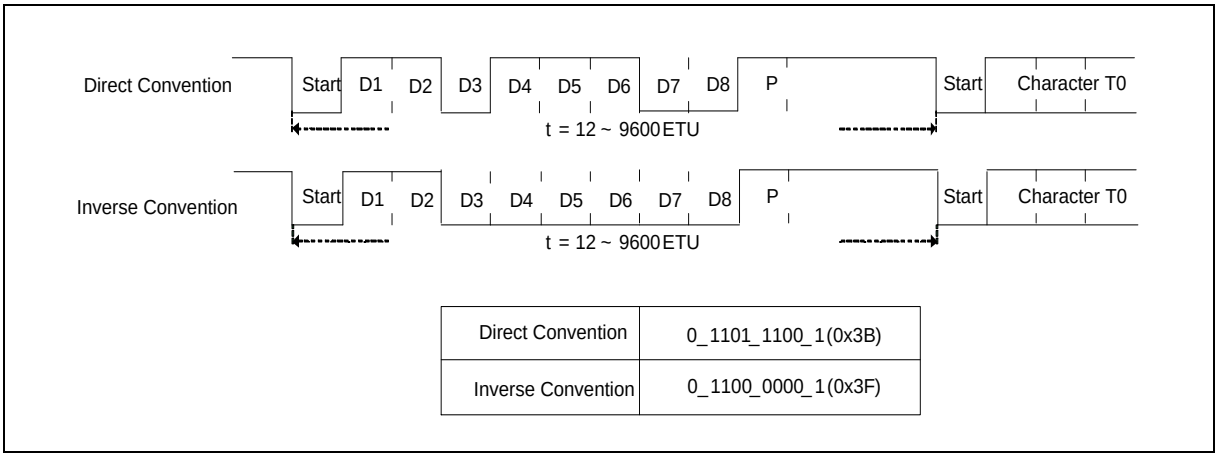


图 6.18-7 基本操作流程

6.18.5.4 初始化TS字符

依据ISO7816-3协议，初始化字符TS有两种可能的模式，如图 7.17 8 所示。如果TS模式是1100_0000，则是反向约定。当按反向约定进行解码后，则传送的字节等于0x3F。如果TS模式是1101_1100，则是直接约定。当按直接约定解码后，则传送的字节等于0x3B。用户可以设置AUTOCEN(SCn_CTL[3])让硬件来决定操作的约定。用户也可以设置CONSEL(SCn_CTL[5:4])为'00'或者'11'，在收到ATR的TS后才去改变操作的约定。

如果软件通过设置AUTOCEN(SCn_CTL[3])使能自动约定功能，则设置步骤必须在ATR状态之前完成，而且第一个数据必须是0x3B或0x3F。在硬件收到第一个数据并放到SCn_DAT中之后，硬件将决定约定模式并自动改变CONSEL(SCn_CTL[5:4])。如果第一个数据既不是0x3B也不是0x3F,ACERRIF(SCn_INTSTS[10]自动约定错误中断状态标志)置位，硬件将产生一个INT_ACON_ERR中断给CPU(如果ACERRIEN(SCn_INTEN[10])='1')。



Direct Convention	0_1101_1100_1(0x3B)
Inverse Convention	0_1100_0000_1(0x3F)

图 6.18-8 初始化字符 TS

6.18.5.5 发送数据流程和数据缓存状态

设置初始化序列后，SC开始按ISO 7816-3规定的格式进行数据传输。配置ETURDIV(SC_ETUCTL[11:0])为273使ETU符合ISO 7816-3的规定。写数据到SC_DAT，SC会发送8位的数据到智能卡。读SC_DAT，SC会返回从智能卡接收到的8位数据。

SC_STATUS是数据缓存状态寄存器。TXPOINT(SC_STATUS[26:24])和RXPOINT(SC_STATUS[18:16])表示发送缓存和接收缓存里的数据量。TXEMPTY(SC_STATUS[9]),TXFULL(SC_STATUS[10]),TXOV(SC_STATUS[8]),RXEMPTY(SC_STATUS[1]),RXFULL(SC_STATUS[2]),RXOV(SC_STATUS[0])则表示发送/接收缓存的状态是满，空还是溢出。置位TBEIEN(SCn_INTEN[1])标志，SC会在发送缓存空时产生中断到CPU。中断产生后，写1到TBEIEN(SCn_INTEN[1])可以清除中断。SC可以通过配置这些寄存器来决定是否传输数据。

置位TXOFF(SC_CTL[2])和RXOFF(SC_CTL[1])分别可以关闭TX和RX。TXACT(SC_STATUS[31])和RXACT(SC_STATUS[23])代表TX发送/RX接收的状态是激活还是关闭。

6.18.5.6 接收缓存超时

RX缓存接收到一个新的值后超时计数器复位，开始向下计数。如果使能了RXTOIEN(SCn_INTEN[9])，计数器计数到1时还没有接收到新的值，或者CPU并没有通过SCn_DAT读取数据，接收超时计数标志RXTOIF(SCn_INTSTS[9])会被置位，硬件会产生一个中断到CPU。

6.18.5.7 错误信号和字符重复

依据ISO7816-3协议T=0模式的描述，如图 7.17 9 所示，如果接收器接收到一个错误的校验位，接收器将会拉低SCn_DATA1.5个位周期去通知发送器校验错误。然后发送器将重传该字符。SC接口控制器支持接收器硬件错误检测功能和发送器硬件重传功能。

软件可以通过设置TXRTYEN(SCn_CTL[23])来使能重传功能。软件也可以在TXRTY(SCn_CTL[22:20])中定义重传的次数限制。如果重传次数在1到TXRTY之间，硬件自动置位TXRERR(SCn_STATUS[29])。重传次数可以达TXRTY+1,当重传次数等于TXRTY+1时，硬件将会置TXOVERR(SCn_STATUS[30])标志，此时如果使能了TERRIEN(SCn_INTEN[2])，SC控制器将会产生一个传输错误中断给CPU，TERRIF(SC_INTSTS[2])会被置位。

用户可以配置RXRTYEN(SCn_CTL[19])使能重新接收功能。软件也可以在RXRTY(SCn_CTL[18:16])定义接收重传次数的限制。如果重收次数在1到RXRTY之间，硬件自动置位RXRERR(SCn_STATUS[21])

。接收重传次数可达RXTRY+1，如果接收错误的次数等于RXTRY+1,接收器将接收该错误的数据到缓存并且硬件将会置位RXOVERR(SCn_STATUS[22])，如果TERRIEN(SCn_INTEN[2])使能，SC控制将产生一个传输错误中断给CPU，TERRIF(SC_INTSTS[2])标志会被置位。

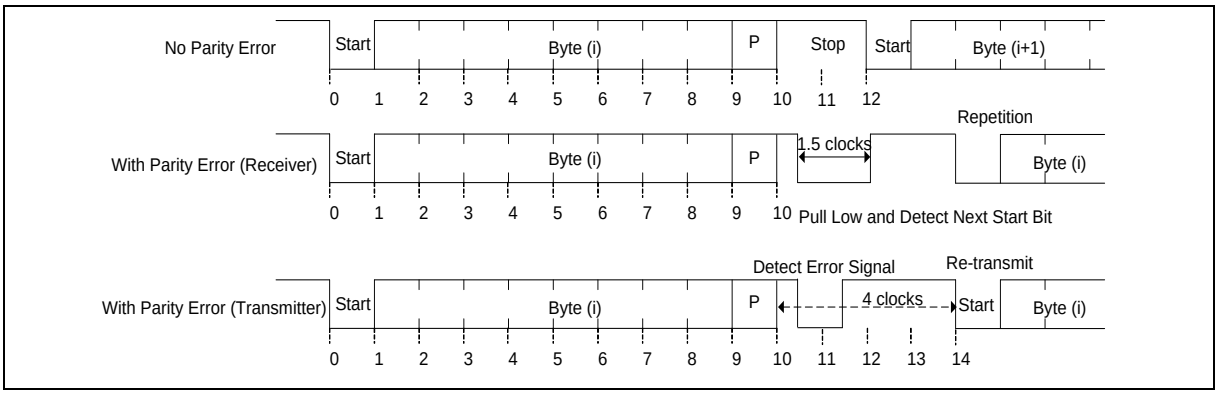


图 6.18-9SC 错误信号

6.18.5.8 内部定时器操作模式

智能卡接口包括一个24位超时计数器和两个8位超时计数器。这些计数器帮助控制器处理不同的实时间隔。每个计数器在使能位(CNTENx在SCn_ALTCTL[7:5], x = 0,1,2中)被置位或者检测到一个START位就开始进行计数。

下面是编程流程：

1. 配置TMRSEL(SCn_CTL[14:13])为11使能计数器
2. 选择操作模式OPMODE(SCn_TMRCTLx[27:24], x = 0, 1, 2)
3. 配置Timer0,Timer1和Timer2的计数器值寄存器 CNT0(SCn_TMRCTL0[23:0]),CNT1(SCn_TMRCTL1[7:0]) 和 CNT2 (SCn_TMRCTL2[7:0] 寄存器)
4. 置位CNTEN0(SCn_ALTCTL[5]),CNTEN1(SCn_ALTCTL[6])或CNTEN2(SCn_ALTCTL[7])使能定时器。ACTSTS0(SCn_ALTCTL[13]),ACTSTS1(SCn_ALTCTL[14])和 ACTSTS2(SCn_ALTCTL[15])指示对应计数器的使能状态
5. 满足计数条件后，定时器开始计数
6. TMR0IEN(SCn_INTEN[3]),TMR1IEN(SCn_INTEN[4]),TMR2IEN(SCn_INTEN[5])使能后，当内部定时器计数器达到中断条件时，SC会产生中断到CPU。

SCn_TMRCTL0,SCn_TMRCTL1和SCn_TMRCTL2定时器操作模式参见表6.18-3。

注1:只有Timer0(SCn_TMRCTL0寄存器)支持模式0011。

注2:START位只能在Tx或Rx空闲时或完成最后一次传输后才能被检测到。

OPMODE(SCn_TMRCTLx[27:24]),X = 0, 1, 2)	操作模式描述

0000	当CNTENx(SCn_ALTCTL[7:5])使能时，向下计数器开始计数，当计数器超时时，停止计数。超时值是CNT(SCn_TMRCTL0[23:0],SCn_TMRCTL1[7:0],SCn_TMRCTL2[7:0])+1的值	
	开始	当CNTENx(SCn_ALTCTL[7:5])使能时开始计数。
	结束	当向下计数器的计数到 0 时，硬件将 TMR0IF, TMR1IF, TMR2IF (SCn_INTSTS[5:3]) 置位并且自动将 CNTENx (SCn_ALTCTL[7:5]) 清零
0001	当第一个 START 位 (接收或者发送) 检测到时向下计时器开始计数，当计数器超时时停止计数。从检测到 START 位到发送数据到 TX 或者从 RX 接收数据会花费 2 个 ETU 的时间。超时值是 CNT (SCn_TMRCTL0[23:0], SCn_TMRCTL1[7:0], SCn_TMRCTL2[7:0])+1 的值。	
	开始	在 CNTENx (SCn_ALTCTL[7:5]) 设置为 1 后，当检测到第一个 START 位 (接收或者传送) 时，计时器开始计数
	结束	当向下计数器的值计数到 0 时，硬件将 TMR0IF, TMR1IF, TMR2IF (SCn_INTSTS[5:3]) 置位并且自动将 CNTENx (SCn_ALTCTL[7:5]) 清零。
0010	当第一个 START 位 (接收) 检测到时，向下计数器开始计数，当计数器超时时，停止计数。从检测到 START 位到 RX 接收数据会花费 2 个 ETU 的时间。超时值是 CNT (SCn_TMRCTL0[23:0], SCn_TMRCTL1[7:0], SCn_TMRCTL2[7:0])+1 的值。.	
	开始	在CNTENx (SCn_ALTCTL[7:5]) 设置为 1 后，当检测到第一个 START 位 (接收) 时，计时器开始计数
	结束	当向下计数器的值等于 0 时，硬件将TMR0IF, TMR1IF, TMR2IF (SCn_INTSTS[5:3]) 置位并且自动将CNTENx (SCn_ALTCTL[7:5])清零。
0011	向下计数器只用于硬件激活，热复位序列下 ATR 时序的测量。 时序开始于 SCn_RST 释放，结束于 ATR 应答收到或超时。 如果在 ATR 应答收到之前，计数器计数到 0，此时如果 TMR0IEN (SCn_INTEN[3]) 已使能，则硬件会置位 TMR0IF (SCn_INTSTS[3])，并产生一个中断给 CPU。超时值为 (SCn_TMRCTL0[23:0])+1。	
	开始	在CNTEN0 (SCn_ALTCTL[5]) 设为 1 后，当 SCn_RST 释放时，计数器开始计数。仅用于硬件激活和热复位模式。
	结束	在 ATR 应答收到之前，如果向下计数器的值计数到 0，硬件将自动将TMR0IF 置位并清 CNTEN0 (SCn_ALTCTL[5])。 当收到 ATR 时，向下计数器计数还未计数到 0，硬件将自动清除 CNTEN0 (SCn_ALTCTL[5])。
0100	开始	在 CNTENx (SCn_ALTCTL[7:5]) 设置为 1 后，计时器开始计数
	重计数&重载	ACTSTSx (SCn_ALTCTL[15:13]) 为 1 时，用户可以在任何时间更改寄存器 CNT (SCn_TMRCTL0[23:0], SCn_TMRCTL0[7:0], SCn_TMRCTL0[7:0]) 的值。在计数器计数到 0 之前，上次填入 CNT (SCn_TMRCTL0[23:0], SCn_TMRCTL1[7:0], SCn_TMRCTL2[7:0]) 的值会被加载到计数器。 但是仅当计数器计数到 0 时，计数器才会重载 CNT (SCn_TMRCTL0[23:0], SCn_TMRCTL1[7:0], SCn_TMRCTL2[7:0]) 并开始计数。
	中断	计数器计数到 0 时，硬件会置位 TMR0IF, TMR1IF, TMR2IF (SCn_INTSTS[5:3])，此时如果使能了 TMRxIEN (SCn_INTEN[5:3]) ，SC 会产生一个中断到 CPU。超时计数值为 CNT (SCn_TMRCTL0[23:0], SCn_TMRCTL1[7:0], SCn_TMRCTL2[7:0]) +1.
	结束	用户清除 CNTENx (SCn_ALTCTL[7:5]) 寄存器，向下计数器停止计数

0101	开始	CNTENx (SCn_ALTCTL[7:5]) 置 1 后，当第一个 START 位 (接收或者发送) 检测到时向下计时器开始计数。从检测到 START 位到发送数据到 TX 或者从 RX 接收数据会花费 2 个 ETU 的时间。
	重载	ACTSTSx (SCn_ALTCTL[15:13]) 为 1 时，用户可以在任何时间更改寄存器 CNT (SCn_TMRCTL0[23:0], SCn_TMRCTL0[7:0], SCn_TMRCTL0[7:0]) 的值。在计数器计数到 0 之前，上次填入 CNT (SCn_TMRCTL0[23:0], SCn_TMRCTL1[7:0], SCn_TMRCTL2[7:0]) 的值会被加载到计数器。 但是仅当计数器计数到 0 时，计数器才会重载 CNT (SCn_TMRCTL0[23:0], SCn_TMRCTL1[7:0], SCn_TMRCTL2[7:0])。
	重计数	向下计数器重载了 CNT 的值后，在检测到下一个 START 位时定时器重新开始计数
	中断	计数器计数到 0 时，硬件会置位 TMR0IF, TMR1IF, TMR2IF (SCn_INTSTS[5:3])，此时如果使能了 TMRxIEN (SCn_INTEN[5:3]) ，SC 会产生一个中断到 CPU。超时计数值为 CNT (SCn_TMRCTL0[23:0], SCn_TMRCTL1[7:0], SCn_TMRCTL2[7:0]) +1
	结束	用户清除 CNTENx (SCn_ALTCTL[7:5]) 寄存器，向下计数器停止计数
0110	开始	CNTENx (SCn_ALTCTL[7:5]) 置 1 后，当第一个 START 位 (接收或者发送) 检测到时向下计时器开始计数。从检测到 START 位到发送数据到 TX 或者从 RX 接收数据会花费 2 个 ETU 的时间。
	重载	ACTSTSx (SCn_ALTCTL[15:13]) 为 1 时，用户可以在任何时间更改寄存器 CNT (SCn_TMRCTL0[23:0], SCn_TMRCTL0[7:0], SCn_TMRCTL0[7:0]) 的值。在计数器计数到 0 之前，上次填入 CNT (SCn_TMRCTL0[23:0], SCn_TMRCTL1[7:0], SCn_TMRCTL2[7:0]) 的值会被加载到计数器。 但是仅当计数器计数到 0 时，计数器才会重载 CNT (SCn_TMRCTL0[23:0], SCn_TMRCTL1[7:0], SCn_TMRCTL2[7:0])。
	重计数	向下计数器重载了 CNT 的值后，在检测到下一个 START 位时定时器重新开始计数
	中断	计数器计数到 0 时，硬件会置位 TMR0IF, TMR1IF, TMR2IF (SCn_INTSTS[5:3])，此时如果使能了 TMRxIEN (SCn_INTEN[5:3]) ，SC 会产生一个中断到 CPU。超时计数值为 CNT (SCn_TMRCTL0[23:0], SCn_TMRCTL1[7:0], SCn_TMRCTL2[7:0]) +1
	结束	用户清除 CNTENx (SCn_ALTCTL[7:5]) 寄存器，向下计数器停止计数
0111	开始	CNTENx (SCn_ALTCTL[7:5]) 置 1 后，当第一个 START 位 (接收或者发送) 检测到时向下计时器开始计数。从检测到 START 位到发送数据到 TX 或者从 RX 接收数据会花费 2 个 ETU 的时间。
	重载&重计数	仅当检测到下一个 START 位，计数器会载入新的计数值 CNT (SCn_TMRCTL0[23:0], SCn_TMRCTL1[7:0], SCn_TMRCTL2[7:0]) 并重新开始计数
	中断	计数器计数到 0 时，硬件会置位 TMR0IF, TMR1IF, TMR2IF (SCn_INTSTS[5:3])，此时如果使能了 TMRxIEN (SCn_INTEN[5:3]) ，SC 会产生一个中断到 CPU。超时计数值为 CNT (SCn_TMRCTL0[23:0], SCn_TMRCTL1[7:0], SCn_TMRCTL2[7:0]) +1.
	结束	用户清除 CNTENx (SCn_ALTCTL[7:5]) 寄存器，向下计数器停止计数
1111	开始	在 CNTENx (SCn_ALTCTL[7:5]) 设置为 1 后，计时器开始计数。超时后停止计数。
	重载&重计数	仅当检测到下一个 START 位，计数器会载入新的计数值 CNT (SCn_TMRCTL0[23:0], SCn_TMRCTL1[7:0], SCn_TMRCTL2[7:0]) 并重新开始计数
	中断	计数器计数到 0 时，硬件会置位 TMR0IF, TMR1IF, TMR2IF (SCn_INTSTS[5:3])，此时如果使能了 TMRxIEN (SCn_INTEN[5:3]) ，SC 会产生一个中断到 CPU。超时计数值为

		CNT (SCn_TMRCTL0[23:0], SCn_TMRCTL1[7:0], SCn_TMRCTL2[7:0]) +1.
	结束	用户清除 CNTENx (SCn_ALTCTL[7:5]) 寄存器，向下计数器停止计数

表 6.18-3 Timer0/Timer1/Timer2 操作模式

6.18.5.9 块保护时间和扩展保护时间

块保护时间是在不同传输方向之间的两个连续字符的第一位边缘之间的最短位长度。该域表示用于块保护时间的计数器。依据ISO7816-3，在T=0的模式下，软件必须填充该域为15(实际的块保护时间=16.5)，在T=1的模式下，软件必须填充该域位21(实际的块保护时间为22.5)。

发送时，智能卡先发送数据到智能卡主控制器。在一段时间(长度超过BGT(SCn_CTL[12:8]))后，智能卡主控制器开始发送数据.

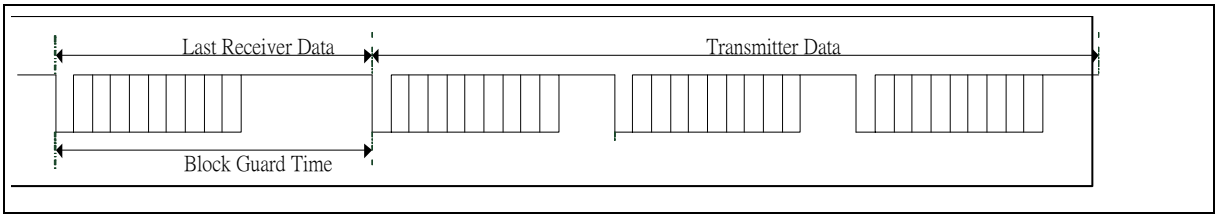


图 6.18-10 发送时块保护时间操作

接收时，智能卡主控制器先发送数据给智能卡。如果智能卡发送数据给智能卡主控制器的时间少于BGT(SCn_CTL[12:8])，当RXBGTEN(SCn_ALTCTL[12])和BGTIEN(SCn_INTEN[6])都使能时，块保护时间中断BGTIF(SCn_INTSTS[6])发生。

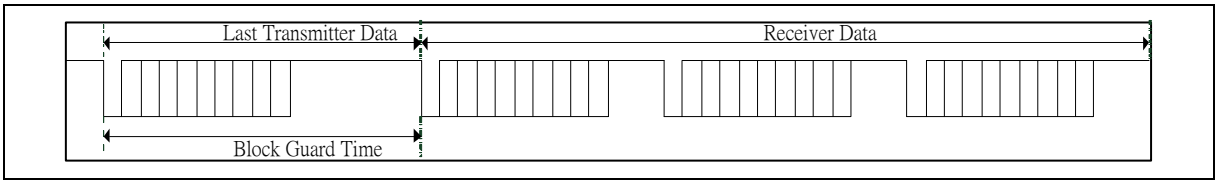


图 6.18-11 接收时的块保护时间操作

扩展保护时间为 EGT(SCn_EGT[7:0])，它只影响由智能卡接口发送的数据，具体格式如图6.18-12和图6.18-11所示：

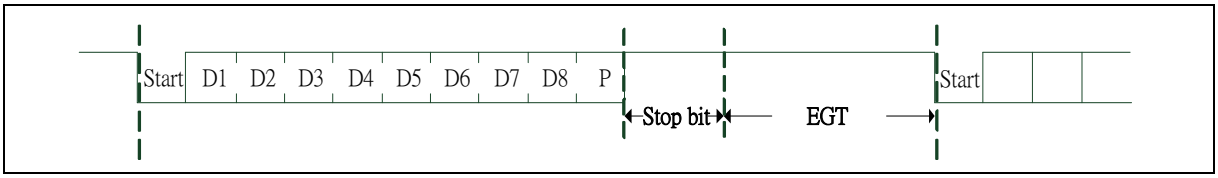


图 6.18-12 扩展保护时间操作

6.18.5.10 UART模式

当UARTEN(SCn_UARTCTL[0])被置位，智能卡控制器也可以作为一个基本的UART功能来使用。下面是一个UART模式的编程例子。

- 编程示例：

1. 置位UARTEN(SCn_UARTCTL[0])进入UART模式。.
2. 配置RXRST(SCn_ALTCTL[1])和RXRST(SCn_ALTCTL[1])进行软件复位，以确保所有的状态机返回空闲状态。
3. 填充0到CONSEL(SCn_CTL[5:4])和AUTOCEN(SCn_CTL[3])。UART模式下，这些位必须为0
4. 配置ETURDIV(SCn_ETUCTL[11:0])选择串口波特率。例如，如果智能卡模块的时钟为12MHZ，而且目标波特率为115200，则ETURDIV应设填为(12000000/115200-1)
5. 选择数据格式，包括数据长度(配置寄存器WLS(SCn_UARTCTL[5:4]))，校验位格式(配置寄存器OPE(SCn_UARTCTL[7])和PBOFF(SCn_UARTCTL[6]))和停止位长度(配置寄存器NSB(SCn_CTL[15]或EGT(SCn_EGT[7:0]))。
6. 通过设定RXTRGLV(SCn_CTL[7:6])域来选择接收缓存触发级别，并通过设定RFTM(SCn_RXTOUT[8:0])域来选择接收缓存超时间隔。
7. 写SCn_DAT(SCn_DAT[7:0])(TX)或者读SCn_DAT(SCn_DAT[7:0])(RX)执行UART功能。

6.18.6 寄存器映射

R: 只读，W: 只写，R/W: 读/写

寄存器	偏移量	R/W	描述	复位值
SC 基地址: **SCn_BA = 0x4009_0000 + (0x1000 * n)** **n=0,1,2**				
SC_DAT	SCn_BA+0x00	R/W	SC 接收/发送缓存保持寄存器	0xXXXX_XXXX
SC_CTL	SCn_BA+0x04	R/W	SC 控制寄存器	0x0000_0000
SC_ALTCTL	SCn_BA+0x08	R/W	SC 交替控制寄存器	0x0000_0000
SC_EGT	SCn_BA+0x0C	R/W	SC 扩展保护寄存器	0x0000_0000
SC_RXTOUT	SCn_BA+0x10	R/W	SC 接收缓存超时寄存器	0x0000_0000
SC_ETUCTL	SCn_BA+0x14	R/W	SC ETU 控制寄存器	0x0000_0173
SC_INTEN	SCn_BA+0x18	R/W	SC 中断使能控制寄存器	0x0000_0000
SC_INTSTS	SCn_BA+0x1C	R/W	SC 中断状态寄存器	0x0000_0002
SC_STATUS	SCn_BA+0x20	R/W	SC 传输状态寄存器	0x0000_X202
SC_PINCTL	SCn_BA+0x24	R/W	SC 管脚控制状态寄存器	0x0000_0000
SC_TMRCTL0	SCn_BA+0x28	R/W	SC 内部定时器 0 寄存器	0x0000_0000
SC_TMRCTL1	SCn_BA+0x2C	R/W	SC 内部定时器 1 寄存器	0x0000_0000
SC_TMRCTL2	SCn_BA+0x30	R/W	SC 内部定时器 2 寄存器	0x0000_0000
SC_UARTCTL	SCn_BA+0x34	R/W	SC UART 模式控制寄存器	0x0000_0000
SC_ACTCTL	SCn_BA+0x4C	R/W	SC 激活控制寄存器	0x0000_0000

6.18.7 寄存器描述

SC接收/发送缓存保持寄存器(SC_DAT)

寄存器	偏移量	R/W	描述	复位值
SC_DAT	SCn_BA+0x00	R/W	SC 接收/发送缓存保持寄存器	0xXXXX_XXXX

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
DAT							

位	描述	
[31:8]	Reserved	保留
[7:0]	DAT	**接收/发送保持缓存器** 写操作： 通过写数据到该寄存器，SC 将发送出一个 8 位数据 **注:**如果SCEN (SCn_CTL[0])没有使能，不能写该寄存器. 读操作： 通过读该寄存器，SC 将返回接收到的一个 8-位数据

SC 控制寄存器(SC_CTL)

寄存器	偏移量	R/W	描述	复位值
SC_CTL	SCn_BA+0x04	R/W	SC 控制寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved	SYNC	Reserved			CDLV	CDDBSEL	
23	22	21	20	19	18	17	16
TXRTYEN	TXRTY			RXRTYEN	RXRTY		
15	14	13	12	11	10	9	8
NSB	TMRSEL		BGT				
7	6	5	4	3	2	1	0
RXTRGLV		CONSEL		AUTOCEN	TXOFF	RXOFF	SCEN

位	描述	
[31]	Reserved	保留
[30]	SYNC	**SYNC标志指示(只读)** 用于同步，软件在写新的值到RXTRY和TXTRY之前必须检查该位 0=同步已经完成，用户可以写入新数据给RXTRY和TXTRY 1=最后值在同步中.
[29:27]	Reserved	保留
[26]	CDLV	**智能卡检测电平选择** 0=当硬件检测到卡检测脚(SCn_CD)电平从高变为低时，为检测到卡 1=当硬件检测到卡检测脚(SCn_CD)电平从低变为高时，为检测到卡。 **注：**软件必须在智能卡引擎使能前选定卡检测电平
[25:24]	CDDBSEL	**智能卡检测消抖选择** 该域为智能卡检测消抖选择 00=每384(128*3)个SC外部时钟去抖采样卡插入一次，每128个SC外部时钟去抖采样卡移除一次 其他配置保留
[23]	TXRTYEN	**TX错误重传使能位** 当校验错误发生时，该位使能发送器重传 0=禁用TX错误重传功能. 1=使能TX错误重传功能
[22:20]	TXRTY	**TX错误重传计数** 该域表明当校验错误发送后，发送器允许尝试的最多重传次数。 **注1：**实际的重传次数是TXTRY+1，所以8次是最多的重传次数。 **注2：**当TXRTYEN使能时，该域不能被修改。修改该域的流程是先禁用TXRTYEN，然后填

		充新的重传值。
[19]	**RXRTYEN**	**RX错误重收使能位** 当校验位错误发生时，该位使能接收器重收功能 0=禁止RX错误重收功能 1=使能RX错误重收功能 **注**：在使能该位之前，软件必须先填充RXRTY
[18:16]	**RXRTY**	**RX错误重收次数** 该域表示当校验错误发生时，接收器允许重收的最多次数。. **注1**：实际的重接收次数是RXRTY+1，所以8次是最多的重收次数。 **注2**：当RXRTYEN使能时，该域不能被修改。修改该域的流程是先禁用RXRTYEN，然后填充新的重收值。
[15]	**NSB**	**停止位长度** 该域表示停止位的长度. 0=停止位的长度是2ETU. 1=停止位的长度是1ETU. **注1:**默认停止位的长度是2。SMC和UART采用NSB来设置停止位的长度。 **注2:**UART模式下，配置NSB为0时，RX可以接收1位或2位停止位的数据。
[14:13]	**TMRSEL**	**定时器通道选择** 00=禁用所有内部定时器 11=使能内部24位定时器和两个8位定时器。软件可以通过设置寄存器SCn_TMRCTL0[23:0]，寄存器SCn_TMRCTL1[7:0]和寄存器SCn_TMRCTL2[7:0来配置定时器 其他配置保留
[12:8]	**BGT**	**块保护时间(BGT)** 块保护时间是在不同传输方向之间的两个连续字符的第一位边缘之间的最短位长度。该域为用于块保护时间的计数器。依据ISO7816-3，在T=0的模式下，软件必须填充该域为15(实际的块保护时间=16.5)，在T=1的模式下，软件必须填充该域为21(实际的块保护时间=22.5) **注:**实际块保护时间是BGT+1.
[7:6]	**RXTRGLV**	**Rx缓存触发水平** 当接收到的字节数等于RXTRGLV时，RDAIF会被置位。此时如果使能了RDAIEN(SCn_INTEN[0])，将会产生中断到CPU 00=Rx缓存触发水平为01字节. 01=Rx缓存触发水平为02字节. 10=Rx缓存触发水平为03字节. 11=保留
[5:4]	**CONSEL**	**约定选择** 00=直接约定. 01=保留 10=保留 11=反向约定 **注:**如果寄存器AUTOCEN(SCn_CTL[3])使能，该域将被忽略
[3]	**AUTOCEN**	**自动约定使能位**

		该位用来使能自动约定功能 0=禁止自动约定功能 1=使能自动约定功能. 如果软件使能自动约定功能，设置步骤必须在应答转复位状态之前完成，而且第一个数据必须是0x3B或者0x3F。在硬件收到第一个数据并保存到缓存后，硬件将决定何种约定并自动改变寄存器CONSEL(SCn_CTL[5:4])的值。如果收到的第一个字节是0x3B，TS为直接约定， CONSEL(SCn_CTL[5:4]) 会被自动设为00。其他情况， TS是反向约定CONSEL(SCn_CTL[5:4])会被设为11 如果第一个数据既不是0x3B也不是0x3F，且使能了ACERRIEN(SCn_INTEN[10])，则硬件将产生中断到CPU
[2]	**TXOFF**	**禁止TX发送控制位** 该位用来禁止TX发送功能 0=使能发送 1=禁止发送
[1]	**RXOFF**	**禁止RX接收控制位** 该位用来禁止RX接收功能 0=使能接收 1=禁止发送 **注1:**如果AUTOCEN(SCn_CTL[3])使能,该域被忽略.
[0]	**SCEN**	**SC控制器使能位** 该位为0时，写1到该位使能SC控制器 0=强制SC进入空闲状态 1=SC控制器使能，所有功能正常 **注1:**在写其他SC寄存器前需要确保SCEN为1，否则智能卡将不能正常工作

SC交替控制寄存器(SC_ALTCTL)

寄存器	偏移量	R/W	描述	复位值
SC_ALTCTL	SCn_BA+0x08	R/W	SC交替控制寄存器	0x0000_0000

31	30	29	28	27	26	25	24
SYNC	Reserved						
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
ACTSTS2	ACTSTS1	ACTSTS0	RXBGTEN	ADACEN	Reserved	INITSEL	
7	6	5	4	3	2	1	0
CNTEN2	CNTEN1	CNTEN0	WARSTEN	ACTEN	DACTEN	RXRST	TXRST

位	描述	
[31]	SYNC	**SYNC标志指示(只读)** 用于同步，软件在写新的值到SCn_ALTCTL之前必须检查该位 0=同步已经完成，用户可以写入新数据给SCn_ALTCTL 1=最后值在同步中.
[30:16]	Reserved	保留
[15]	ACTSTS2	**内部Timer2激活状态(只读)** 该位表示内部定时器Timer2计数器的状态 0=Timer2未激活. 1=Timer2激活. **注:**Timer2激活并不意味着CNT(SCn_TMRCTL2[7:0])处于计数状态
[14]	ACTSTS1	**内部Timer1激活状态(只读)** 该位表示内部定时器Timer1计数器的状态 0=Timer1未激活 1=Timer1激活 **注:**Timer1激活并不意味着CNT(SCn_TMRCTL1[7:0])处于计数状态
[13]	ACTSTS0	**内部Timer0激活状态(只读)** 该位表示内部定时器Timer0计数器的状态 0=Timer0未激活 1=Timer0激活 **注:**Timer0激活并不意味着CNT(SCn_TMRCTL0[23:0])处于计数状态
[12]	RXBGTEN	**接收器块保护时间功能使能位** 0=禁止接收器块保护时间功能 1=使能接收器块保护时间功能.

[11]	**ADACEN**	**智能卡移除时自动释放** 该位用来使能智能卡移除时自动释放功能 0=禁止自动释放 1=使能自动释放 **注:**当智能卡移除时，如果该位置位，硬件会停止所有操作并释放序列。自动序列释放后，硬件会置位INITIF(SCn_INTSTS[8])
[10]	**Reserved**	保留
[9:8]	**INITSEL**	**初始化时间选择** 该域表示硬件激活、热复位或者释放的初始化时间。初始化时间的单位是SC模块时钟 激活:参考图6.18-54SC 激活序列. 热复位: 参考图6.18-5热复位序列. 释放: 参考图6.18-56释放序列. **注:**当释放和热复位处于Timer0操作模式0011时，释放最多要128个SC模块时钟周期
[7]	**CNTEN2**	**内部定时器2开始使能位** 该位使能定时器2开始计数。软件可以写0来停止计数器，置1来重载和计数。 0=停止计数 1=开始计数. **注1:**当寄存器TMRSEL(SCn_CTL[14:13])=11时，该域用于内部8位定时器。当寄存器TMRSEL(SCn_CTL[14:13])不为11时，不要填充CNTEN2。 **注2:**如果操作模式不是自动重载模式(SCn_TMRCTL2[26]=0)，则该位会由硬件自动清零。 **注3:**如果寄存器SCEN(SCn_CTL[0])未使能，不能对该位进行编程。
[6]	**CNTEN1**	**内部定时器1开始使能位** 该位使能定时器1开始计数。软件可以写0来停止计数器，置1来重载和计数。 0=停止计数 1=开始计数 **注1:**当寄存器TMRSEL(SCn_CTL[14:13])=10或11时，该域用于内部8位定时器。当寄存器TMRSEL(SCn_CTL[14:13])不为11时，不要填充CNTEN1。 **注2:**如果操作模式不是自动重载模式(SCn_TMRCTL1[26]=0),则该位将被硬件自动清除。 **注3:**如果寄存器SCEN(SCn_CTL[0]) 未使能，不能对该位进行编程。
[5]	**CNTEN0**	**内部定时器0开始使能位** 该位使能定时器0开始计数。软件可以写0来停止计数器，置1来重载和计数。 0=停止计数 1=开始计数. **注1:**当寄存器TMRSEL(SCn_CTL[14:13])=11时，该域用于内部24位定时器。 **注2:**如果操作模式不是自动重载模式(SCn_TMRCTL0[26]=0)，则该位将被硬件自动清除。 **注3:**如果寄存器SCEN(SCn_CTL[0])未使能，不能对该位进行编程。
[4]	**WARSTEN**	**热复位序列发生器使能位** 该位使能SC控制器通过热复位序列初始化卡 0=无效. 1=使能热复位序列发生器. **注1:**当热复位序列完成后，该位将被自动清除，寄存器INITIF(SCn_INTSTS[8])将被设置为1。 **注2:**该域将被TXRST(SCn_ALTCTL[0])和RXRST(SCn_ALTCTL[1])清除，所以不要同时填

		充WARSTEN、TXRST和RXRST。 **注3:**如果寄存器SCEN(SCn_CTL[0])未使能，该域不能被编程。 **注 4:** 热复位过程中，RX会被自动关闭，无法接收数据。热复位序列完成后，RXOFF(SCn_CTL[1])在热复位完成之前会一直保持不变
[3]	**ACTEN**	**激活序列发生器使能位** 该位使能SC控制器通过激活序列初始化智能卡 0=无效 1=使能激活序列发生器 **注1:**当激活序列完成后，该位将被自动清除，寄存器INITIF(SCn_INTSTS[8])将被设置为1。 **注2:**该域将被寄存器TXRST(SCn_ALTCTL[0])和RXRST(SCn_ALTCTL[1])清除，所以不要同时填充ACTEN、TXRST和RXRST **注3:**如果寄存器SCEN(SCn_CTL[0])未使能，该域不能被编程。 **注 4**: 激活序列过程中，RX会被自动关闭，无法接收数据。硬件激活完成后，RXOFF(SCn_CTL[1])在热复位完成之前会一直保持不变
[2]	**DACTEN**	**释放序列发生器使能位** 该位使能SC控制器通过释放序列初始化智能卡 0=无效 1=使能释放序列发生器 **注1:**当释放序列完成后，该位将被自动清除，寄存器INITIF(SCn_INTSTS[8])将被设置为1。 **注2:**该域将被寄存器TXRST(SCn_ALTCTL[0])和RXRST(SCn_ALTCTL[1])清除，所以不要同时填充DACTEN、TXRST和RXRST **注3:**如果寄存器SCEN(SCn_CTL[0])未使能，该域不能被编程。
[1]	**RXRST**	**RX软件复位** 当RXRST被置位，接收缓存中的所有数据和RX内部状态机将被清除。 0=无效 1=复位RX内部状态机和指针. **注:**在复位完成后，该位将被自动清除
[0]	**TXRST**	**TX软件复位** 当TXRST被置位，发送缓存中的所有数据和TX内部状态机将被清除。 0=无效 1=复位TX内部状态机和指针. **注:**在复位完成后，该位将被自动清除

SC扩展保护寄存器(SC_EGT)

寄存器	偏移量	R/W	描述	复位值
SC_EGT	SCn_BA+0x0C	R/W	SC扩展保护寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
EGT							

位	描述	
[31:8]	Reserved	保留
[7:0]	EGT	**扩展保护时间** 该域表示扩展保护时间的值 **注:**该计数器的单位为ETU

SC接收缓存超时寄存器(SC_RXTOUT)

寄存器	偏移量	R/W	描述	复位值
SC_RXTOUT	SCn_BA+0x10	R/W	SC接收缓存超时寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							RFTM
7	6	5	4	3	2	1	0
RFTM							

位	描述	
[31:9]	Reserved	保留
[8:0]	RFTM	**SC接收FIFO超时寄存器** 当RX接收到一个新的值时，向下的超时计数器复位并开始计数。如果计数器计数到1还没有新的接收值或CPU没去SCn_DAT读取数据，超时标志RXTOIF(SCn_INTSTS[9])会被置位，如果RXTOIEN(SCn_INTEN[9])使能，硬件会产生中断到CPU **注1:**计数单位基于ETU，超时间隔为RFTM+0.5 **注2:**填充该寄存器所有位为0关闭该功能

SCETU控制寄存器(SC_ETUCTL)

寄存器	偏移量	R/W	描述	复位值
SC_ETUCTL	SCn_BA+0x14	R/W	SC ETU控制寄存器	0x0000_0173

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved				ETURDIV			
7	6	5	4	3	2	1	0
ETURDIV							

位	描述	
[31:12]	Reserved	保留
[11:0]	ETURDIV	**ETU除频** 该域用于ETU时钟频率除频 真实的ETU值为ETURDIV+1. **注:**用户配置该寄存器时需要确保大于0x04

SC中断使能控制寄存器(SC_INTEN)

寄存器	偏移量	R/W	描述	复位值
SC_INTEN	SCn_BA+0x18	R/W	SC中断使能控制寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved					ACERRIEN	RXTOIEN	INITIEN
7	6	5	4	3	2	1	0
CDIEN	BGTIEN	TMR2IEN	TMR1IEN	TMR0IEN	TERRIEN	TBEIEN	RDAIEN

位	描述	
[31:11]	Reserved	保留
[10]	ACERRIEN	**自动约定错误中断使能位** 该域用于自动约定错误中断使能. 0=禁止自动约定错误中断. 1=使能自动约定错误中断.
[9]	RXTOIEN	**接收器缓存超时中断使能位** 该域用于接收器缓存超时中断使能 0=禁止接收缓存超时中断. 1=使能接收缓存超时中断.
[8]	INITIEN	**初始化结束中断使能位** 该域用于激活(ACTEN(SCn_ALTCTL[3]=1)，释放((DACTENSCn_ALTCTL[2])=1)和热复位(WARSTEN(SCn_ALTCTL[4]))序列中断使能 0=禁止初始化结束中断. 1=使能初始化结束中断.
[7]	CDIEN	**卡检测中断使能位** 该域用于智能卡检测中断使能，智能卡检测状态的寄存器是CINSERT(SCn_STATUS[13])。 0=禁止卡检测中断. 1=使能卡检测中断.
[6]	BGTIEN	**块保护时间中断使能位** 该域用于块保护时间中断使能 0=禁止块保护时间中断 1=使能块保护时间中断.

		注:该位仅在块保护时间接收方向有效
[5]	**TMR2IEN**	**Timer2中断使能位** 该域用于Timer2中断使能 0=禁止Timer2中断. 1=使能Timer2中断
[4]	**TMR1IEN**	**Timer1中断使能位** 该域用于Timer1中断使能 0=禁止Timer1中断. 1=使能Timer1中断
[3]	**TMR0IEN**	**Timer0中断使能位** 该域用于Timer0中断使能 0=禁止Timer0中断. 1=使能Timer0中断
[2]	**TERRIEN**	**传输错误中断使能位** 该域用于传输错误中断使能。传输错误状态保存在SCn_STATUS寄存器，包括接收器断开错误BEF(SCn_STATUS[6])，帧错误FEF(SCn_STATUS[5])，校验错误PEF(SCn_STATUS[4])，接收缓存溢出错误RXOV(SCn_STATUS[0])，传输缓存溢出错误TXOV(SCn_STATUS[8])，接收器重收超过限制错误RXOVERR(SCn_STATUS[22])和传送器重传超过限制错误TXOVERR(SCn_STATUS[30])。 0=禁止传输错误中断. 1=使能传输错误中断
[1]	**TBEIEN**	**发送缓存空中断使能位** 该域用于发送缓存空中断使能 0=禁止发送缓存空中断. 1=使能发送缓存空中断
[0]	**RDAIEN**	**接收数据到达中断使能位** 该域用于接收数据达到触发水平寄存器RXTRGLV(SCn_CTL[7:6])的中断使能 0=禁止接收数据达到触发水平中断. 1=使能接收数据达到触发水平中断.

SC中断状态寄存器(SC_INTSTS)

寄存器	偏移量	R/W	描述	复位值
SC_INTSTS	SCn_BA+0x1C	R/W	SC中断状态寄存器	0x0000_0002

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved					ACERRIF	RXTOIF	INITIF
7	6	5	4	3	2	1	0
CDIF	BGTIF	TMR2IF	TMR1IF	TMR0IF	TERRIF	TBEIF	RDAIF

位	描述	
[31:11]	Reserved	保留
[10]	ACERRIF	**自动约定错误中断状态标志** 该位指示自动约定序列错误 0=ATR状态收到的TS为0x3B或0x3F. 1=ATR状态收到的TS既不是0x3B也不是0x3F. **注:**该位写1清0
[9]	RXTOIF	**接收缓存超时中断状态标志(只读)** 该域用来指示接收缓存超时中断状态标志 0=接收缓存超时中断未发生 1=接收缓存超时中断已发生 **注:**该位只读，用户需要通过读SCn_DAT寄存器清空所有接收缓存来清除它
[8]	INITIF	**初始化结束中断状态标志** 该位为激活(ACTEN(SCn_ALTCTL[3])), 释放(DACTEN(SCn_ALTCTL[2]))和热复位(WARSTEN(SCn_ALTCTL[4]))序列中断状态标志 0=初始化序列未完成 1=初始化序列已完成 **注:**该位写1清0
[7]	CDIF	**卡检测中断状态标志(只读)** 该位为智能卡检测中断状态标志。智能卡检测状态相关寄存器有CINSERT(SCn_STATUS[12])和CREMOVE(SCn_STATUS[11]) 0=智能卡检测事件未发生 1=智能卡检测事件发生 **注:**该位只读,用户需要清除CINSERT或CREMOVE来清除该位

[6]	**BGTIF**	**块保护时间中断状态标志** 该域用于接收方向上的块保护时间中断状态标志 0=块保护时间中断未发生 1=块保护时间中断发生 **注1:**该位仅在RXBGTEN(SCn_ALTCTL[12])使能时有效 **注2:**该位写1清0
[5]	**TMR2IF**	**Timer2中断状态标志** 该域为Timer2中断状态标志 0=Timer2中断未发生 1=Timer2中断已发生 **注:**该位写1清0
[4]	**TMR1IF**	**Timer1中断状态标志** 该域为Timer1中断状态标志 0=Timer1中断未发生 1=Timer1中断已发生 **注:**该位写1清0
[3]	**TMR0IF**	**Timer0中断状态标志** 该域为Timer0中断状态标志 0=Timer0中断未发生 1=Timer0中断已发生 **注:**该位写1清0
[2]	**TERRIF**	**传输错误中断状态标志** 该域为传输错误中断状态标志。传输错误状态保存在SCn_STATUS寄存器，包括接收器断开错误BEF(SCn_STATUS[6])，帧错误FEF(SCn_STATUS[5])，校验错误PEF(SCn_STATUS[4])，接收缓存溢出错误RXOV(SCn_STATUS[0])，传输缓存溢出错误TXOV(SCn_STATUS[8])，接收器重收超过限制错误RXOVERR(SCn_STATUS[22])和传送器重传超过限制错误TXOVERR(SCn_STATUS[30])。 0=传输错误中断未发生 1=传输错误中断已发生 **注1:**该位为BEF,FEF,PEF,RXOV,TXOV,RXOVERR或TXOVERR的标志 **注2:**该位写1清0
[1]	**TBEIF**	**发送缓存空中断状态标志(只读)** 该域为发送缓存空中断状态标志 0=发送缓存非空 1=发送缓存空 **注:**该位只读.如果用户想要清除该未，用户写值到DAT(SCn_DAT[7:0])，该位会自动清除
[0]	**RDAIF**	**接收数据到达中断状态标志(只读)** 该域为接收数据到达触发水平RXTRGLV(SCn_CTL[7:6])中断状态标志 0=接收缓存的数量小于RXTRGLV的设置. 1=接收缓存的数量达到RXTRGLV的设置. **注:**该位只读.如果用户从SCn_DAT读取数据且接收缓存数据量小于RXTRGLV，该位自动清零

SC传输状态寄存器(SC_STATUS)

寄存器	偏移量	R/W	描述	复位值
SC_STATUS	SCn_BA+0x20	R/W	SC传输状态寄存器	0x0000_X202

31	30	29	28	27	26	25	24
TXACT	TXOVERR	TXRERR	Reserved		TXPOINT		
23	22	21	20	19	18	17	16
RXACT	RXOVERR	RXRERR	Reserved		RXPOINT		
15	14	13	12	11	10	9	8
Reserved		CDPINSTS	CINSERT	CREMOVE	TXFULL	TXEMPTY	TXOV
7	6	5	4	3	2	1	0
Reserved	BEF	FEF	PEF	Reserved	RXFULL	RXEMPTY	RXOV

位	描述	
[31]	TXACT	**发送激活状态标志(只读)** 该位表示Tx发送的状态 0=当TX传输完成或者最后一个字节传输已经完成，则该位自动清除 1=当TX传输处于激活状态而且最后字节的STOP位已经被发送，硬件将该位置位。 **注:**该位只读.
[30]	TXOVERR	**发送重传错误超标** 该位用来表示发送重传次数超过限定的重传次数 0=发送重传次数少于TXRTY(SCn_CTL[22:20])+1. 1=发送重传次数等于或超过TXRTY(SCn_CTL[22:20])+1. **注:**该位写1清0
[29]	TXRERR	**发送重传错误** 该位表示出现错误需要重传，由硬件自动置位 0=无Tx重传 1=Tx发生错误需要重传 **注1:**该位写1清0 **注2:**该位仅仅是个标志，不能触发中断
[28:27]	Reserved	保留
[26:24]	TXPOINT	**发送缓存指针状态(只读)** 该域指示TX缓存指针状态标志。当CPU写数据到SCn_DAT，TXPOINT增加1。当TX缓存的一个字节被传输到发送器移位寄存器时，TXPOINT减少1。
[23]	RXACT	**接收激活状态标志(只读)** 该位表示Rx接收的状态 0=当RX完成接收后，该位自动清除

		1=当RX传输处于激活状态，硬件将该位置位。 **注:**该位只读.
[22]	**RXOVERR**	**接收重收错误超标** 该位用来表示接收重收次数超过限定的重收次数 0=接收重收次数少于RXRTY(SCn_CTL[18:16])+1. 1=接收重收次数等于或超过RXRTY(SCn_CTL[18:16])+1. **注1:**该位写1清0 **注2:**如果通过RXRTYEN(SCn_CTL[19])使能了接收器重收功能，硬件将不会置位该位
[21]	**RXRERR**	**接收器重收错误** 该位表示出现错误需要重收，由硬件自动置位 0=无Rx重收 1=Rx发生错误需要重收 **注1:**该位写1清0 **注2:**该位仅仅是个标志，不能触发中断 **注3:**如果通过RXRTYEN(SCn_CTL[19])使能了接收器重收功能，硬件将不会置位该位
[20:19]	**Reserved**	保留
[18:16]	**RXPOINT**	**接收器缓存指针状态标志(只读)** 该域指示RX缓存指针状态标志。当SC从外部设备收到一个字节，寄存器RXPOINT加1。当RX缓存被CPU读取了一个字节，寄存器RXPOINT减1。
[15:14]	**Reserved**	保留
[13]	**CDPINSTS**	**SCn_CD管脚上的卡检测状态(只读)** 该位为SCn_CD管脚状态的标志 0=SCn_CD管脚状态为低 1=SCn_CD管脚状态为高
[12]	**CINSERT**	**SCn_CD管脚上的卡片插入状态** 该位表示SCn_CD管脚上是否有智能卡插入 0=无效 1=智能卡插入 **注1:**该位写1清0. **注2:**SCEN(SCn_CTL[0])置位后，智能卡插入检测功能开始工作
[11]	**CREMOVE**	**SCn_CD管脚上的卡片移除状态** 该位表示SCn_CD管脚上是否有智能卡移除 0=无效 1=智能卡移除 **注1:**该位写1清0. **注2:**SCEN(SCn_CTL[0])置位后，智能卡移除检测功能开始工作
[10]	**TXFULL**	**发送缓存满状态标志(只读)** 该位表示Tx缓存是否满 0=Tx缓存数小于4 1=Tx缓存数等于4
[9]	**TXEMPTY**	**发送缓存空状态标志(只读)**

		该位表示发送缓存是否为空 0=Tx缓存不为空 1=Tx发送缓存为空，即Tx缓存里的最后一个字节都被送到了发送移位寄存器里了 **注:**写数据到DAT(SCn_DAT[7:0])清0该位
[8]	**TXOV**	**发送溢出错误中断状态标志** Tx缓存溢出时该位置位 0=Tx缓存未溢出 1=Tx缓存已满且DAT(SCn_DAT[7:0])上有新的值写入，Tx缓存溢出 **注:**该位写1清0
[7]	**Reserved**	保留
[6]	**BEF**	**接收器断开错误状态标志** 每当接收到的输入数据(RX)保持在“空状态”(逻辑0)状态的时间长过一个完整的字节传输时间(也就是“开始位”+数据位+校验位+停止位的总时间)，该位被设置为逻辑1。 0=接收器断开错误未发生. 1=接收器断开错误已发生. **注1:**该位写1清0 **注2:**如果通过RXRTYEN(SCn_CTL[19])使能了接收器重收功能，硬件将不会置位该位
[5]	**FEF**	**接收器帧错误状态标志** 每当接收的字符没有一个有效的”“停止位”(也就是紧跟在最后一位数据位或校验位后的停止位被检测为逻辑0)，该位被置为逻辑1。 0=接收器帧错误未发生. 1=接收器帧错误已发生. **注1:**该位写1清0 **注2:**如果通过RXRTYEN(SCn_CTL[19])使能了接收器重收功能，硬件将不会置位该位
[4]	**PEF**	**接收器校验错误状态标志** 接收到的数据位校验位错误时该位置1 0=接收器校验错误未发生 1=接收器校验错误已发生 **注1:**该位写1清0 **注2:**如果通过RXRTYEN(SCn_CTL[19])使能了接收器重收功能，硬件将不会置位该位
[3]	**Reserved**	保留
[2]	**RXFULL**	**接收器缓存满状态标志(只读)** 该位表示Rx缓存是否满 0=Rx缓存数小于4 1=Rx缓存数等于4
[1]	**RXEMPTY**	**接收器缓存空状态标志(只读)** 该位表示Rx缓存是否为空 0=Rx缓存不为空 1=Rx缓存为空，即CPU已经读取了DAT(SCn_DAT[7:0])里的最后一个字节
[0]	**RXOV**	**接收器缓存溢出错误状态标志** 接收器缓存溢出时该位置位

		0=Rx缓存未溢出. 1=接收到的字节数超过了Rx的缓存(4字节)，Rx缓存溢出 **注:**该位写1清0

SC管脚控制状态寄存器(SC_PINCTL)

寄存器	偏移量	R/W	描述	复位值
SC_PINCTL	SCn_BA+0x24	R/W	SC管脚控制状态寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved	SYNC	Reserved					
23	22	21	20	19	18	17	16
Reserved					RSTSTS	PWRSTS	DATASTS
15	14	13	12	11	10	9	8
Reserved				PWRINV	Reserved	SCDATA	Reserved
7	6	5	4	3	2	1	0
Reserved	CLKKEEP	Reserved				RSTEN	PWREN

位	描述	
[31]	**Reserved**	保留
[30]	**SYNC**	**SYNC标志指示(只读)** 用于同步，软件在写新的值到SCn_PINCTL之前必须检查该位 0=同步已经完成，用户可以写入新数据给SCn_PINCTL 1=最后值在同步中.
[29:19]	**Reserved**	保留
[18]	**RSTSTS**	**SCn_RST管脚状态(只读)** 该位为SCn_RST的管脚状态 0=SCn_RST管脚为低电平. 1=SCn_RST管脚为高电平.
[17]	**PWRSTS**	**SCn_PWR管脚状态(只读)** 该位为SCn_PWR的管脚状态 0=SCn_PWR管脚为低电平. 1=SCn_PWR管脚为高电平.
[16]	**DATASTS**	**SCn_DATA管脚状态(只读)** 该位为SCn_DATA的管脚状态 0=TheSCn_DATA管脚为低电平. 1=TheSCn_DATA管脚为高电平.
[15:12]	**Reserved**	保留
[11]	**PWRINV**	**SC_POW管脚反向**

		该位用于SC_PWR管脚反向 通过设置寄存器PWRINV(SC_PINCTL[11])和PWREN(SC_PINCTL[0])，SC_POW管脚的设置有4种组合。PWRINV(SC_PINCTL[11])和PWREN(SC_PINCTL[0])的组合为SC_PWR管脚提供高或低电压选择。 00=SC_PWR为0. 01=SC_PWR为1. 10=SC_PWR为1. 11=SC_PWR为0. **注：**在通过寄存器SSCEN(SC_CTL[0])使能智能卡之前，软件必须设置好寄存器PWRINV(SC_PINCTL[11])。
[10]	**Reserved**	保留
[9]	**SCDATA**	**SCn_DATA管脚信号** 该位为SCn_DATA管脚的信号状态，用户也可以修改该位来驱动SCn_DATA上的电平 0=驱动SCn_DATA管脚为低 1=驱动SCn_DATA管脚为高 读该寄存器获得SCn_DATA的信号状态. 0=SCn_DATA为低 1=SCn_DATA为高 **注:**当SC处于激活，暖复位或者释放模式时，该位将被自动改变，所以SC在这些模式下，不要设置该位。
[8:7]	**Reserved**	保留
[6]	**CLKKEEP**	**SC使能使能位** 0=禁止SC时钟发生器 1=SC时钟发生器正常运行 **注:**当SC处于激活，暖复位或者释放模式时，该位将会自动改变。所以工作在这些模式时，不要填充该位。
[5:2]	**Reserved**	保留
[1]	**RSTEN**	**SC_RST管脚信号** 该位是SC_RST的管脚状态，但是用户可以通过设置RSTEN(SCn_PINCTL[1])驱动SC_RST管脚为高或低 写该域来驱动SC_RST管脚 0=驱动SC_RST管脚为低. 1=驱动SC_RST管脚为高. 读该域来获取SC_RST管脚状态. 0=SC_RST管脚状态为低. 1=SC_RST管脚状态为高. **注:**当SC处于激活，暖复位或释放模式时，该位会自动改变。所以当工作在这些模式时，不要填充该位。
[0]	**PWREN**	**SC_PWREN管脚信号** 软件可以通过设置寄存器PWRINV(SCn_PINCTL[11])和PWREN(SCn_PINCTL[0])来决定SC_PWR管脚为高电平或者低电平 写该域以驱动SC_PWR管脚 编程SC_PWR管脚电平可参考寄存器PWRINV(SC_PINCTL[11])的描述 读该域来获取SC_PWR管脚状态

		0=SC_PWR管脚状态为低 1=SC_PWR管脚状态为高. **注:**当SC处于激活，暖复位或释放模式时，该位会自动改变。所以当操作在这些模式时，不要填充该位。

SC定时器0控制寄存器(SC_TMRCTL0)

寄存器	偏移量	R/W	描述	复位值
SC_TMRCTL0	SCn_BA+0x28	R/W	SC内部定时器0寄存器	0x0000_0000

31	30	29	28	27	26	25	24
SYNC	Reserved			OPMODE			
23	22	21	20	19	18	17	16
CNT							
15	14	13	12	11	10	9	8
CNT							
7	6	5	4	3	2	1	0
CNT							

位	描述	
[31]	SYNC	**SYNC标志指示(只读)** 用于同步，软件在写新的值到SCn_TMRCTL0之前必须检查该位 0=同步已经完成，用户可以写入新数据给SCn_TMRCTL0 1=最后值在同步中.
[30:28]	Reserved	保留
[27:24]	OPMODE	**Timer0工作模式选择** 该域用于选择内部24位定时器Timer0的工作模式 Timer0 的配置方法请参考表6.18-3
[23:0]	CNT	**Timer0计数值** 该域为Timer0计数值 **注:**Timer0的计数单位基于ETU.

SC定时器1控制寄存器(SC_TMRCTL1)

寄存器	偏移量	R/W	描述	复位值
SC_TMRCTL1	SCn_BA+0x2C	R/W	SC内部定时器1寄存器	0x0000_0000

31	30	29	28	27	26	25	24
SYNC	Reserved			OPMODE			
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
CNT							

位	描述	
[31]	SYNC	**SYNC标志指示(只读)** 用于同步，软件在写新的值到SCn_TMRCTL1之前必须检查该位 0=同步已经完成，用户可以写入新数据给SCn_TMRCTL1 1=最后值在同步中.
[30:28]	Reserved	保留
[27:24]	OPMODE	**Timer1工作模式选择** 该域用于选择内部8位定时器Timer1的工作模式 Timer1的配置方法请参考表6. 18-3
[23:8]	Reserved	保留
[7:0]	CNT	**Timer1计数值** 该域为Timer1计数值 **注:**Timer1的计数单位基于ETU.

SC定时器2控制寄存器(SC_TMRCTL2)

寄存器	偏移量	R/W	描述	复位值
SC_TMRCTL2	SCn_BA+0x30	R/W	SC内部定时器2寄存器	0x0000_0000

31	30	29	28	27	26	25	24
SYNC	Reserved			OPMODE			
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
CNT							

位	描述	
[31]	SYNC	**SYNC标志指示(只读)** 用于同步，软件在写新的值到SCn_TMRCTL2之前必须检查该位 0=同步已经完成，用户可以写入新数据给SCn_TMRCTL2 1=最后值在同步中.
[30:28]	Reserved	保留
[27:24]	OPMODE	**Timer2工作模式选择** 该域用于选择内部8位定时器Timer2的工作模式 Timer2的配置方法请参考表6. 18-3
[23:8]	Reserved	保留
[7:0]	CNT	**Timer2计数值** 该域为Timer2计数值 **注:**Timer2的计数单位基于ETU.

SCUART模式控制寄存器(SC_UARTCTL)

寄存器	偏移量	R/W	描述	复位值
SC_UARTCTL	SCn_BA+0x34	R/W	SCUART模式控制寄存器	0x0000_0000

位	描述	
[31:8]	**Reserved**	保留
[7]	**OPE**	**奇检验使能位** 该位用于奇/偶校验的选择 0=奇数个逻辑1被发送，或者在接收模式下检查数据字和校验位 1=偶数个逻辑1被发送，或者在接收模式下检查数据字或校验位 **注意:**.该位仅在PBOFF位为0时起作用
[6]	**PBOFF**	**校验位禁止位** 该位用于关闭校验功能 0=校验位产生或者在串行数据的“最后一个数据位”和“停止位”之间被检查. 1=校验位不产生(发送数据)或者在传输过程中不被检查(接收数据) **注意:**当工作在智能卡模式时，该域必须设置为0(默认设置带校验位)
[5:4]	**WLS**	**数据长度** 该位用来选择UART数据长度 00=字符数据长度为8位. 01=字符数据长度为7位. 10=字符数据长度为6位. 11=字符数据长度为5位. **注意:**当工作在智能卡模式时，WLS必须设置为00。
[3:1]	**Reserved**	保留
[0]	**UARTEN**	**UART模式使能位** 该位用来使能UART模式 0=智能卡模式 1=串口模式 **注 1:** 当工作于串口模式，用户必须设置CONSEL(SC_CTL[5:4])=00 和AUTOCEN(SC_CTL[3])=0。 **注2:**当工作在智能卡模式时，用户必须设置寄存器UARTEN(SC_UARTCTL[0])=00。 **注3:**当串口功能被使能时，硬件将产生一个复位信号来复位内部缓存和内部状态机。

SC激活控制寄存器(SC_ACTCTL)

寄存器	偏移量	R/W	描述	复位值
SC_ACTCTL	SCn_BA+0x4C	R/W	SC激活控制寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved			T1EXT				

位	描述	
[31:5]	Reserved	保留
[4:0]	T1EXT	**T1硬件激活的扩展时间** 该域用来配置硬件激活的扩展时间周期 该周期的放大因数是2048 扩展周期=(寄存器值*2048)个周期 参考SC激活序列举例图6.18-4. SCLK=4MHz,每个周期为0.25us,. 设该寄存器为20 扩展周期=20*2048*0.25us=10.24ms. **注:**设该寄存器为0也符合ISO/IEC7816-3协议

6.19 I²S控制器(I²S)

6.19.1 概述

I²S控制器提供符合IIS协议可外接语音CODEC接口。输入输出有独立的16字节深度的FIFO，可以支持8/16/24/32位的数据宽度。PDMA控制器可以在FIFO和内存间搬运数据。

6.19.2 特征

- 支持主机模式和从机模式
- 每路音频通道都支持8,16,24和32位的数据宽度
- 支持单声道和双声道的音频数据
- 支持I²S协议：飞利浦标准,MSB对齐和LSB对齐的数据格式
- 支持PCM协议：PCM标准,MSB对齐和LSB对齐的数据格式
- PCM协议支持在一个声音采样里TDM多通道传输，通道数可设为2,4,6或8
- 提供2个16字节深度的数据缓存，分别为发送缓存和接收缓存
- 当缓存达到设定的阈值时可以产生中断请求到CPU
- 支持2路PDMA请求，一路用于发送，另一路用于接收

6.19.3 框图

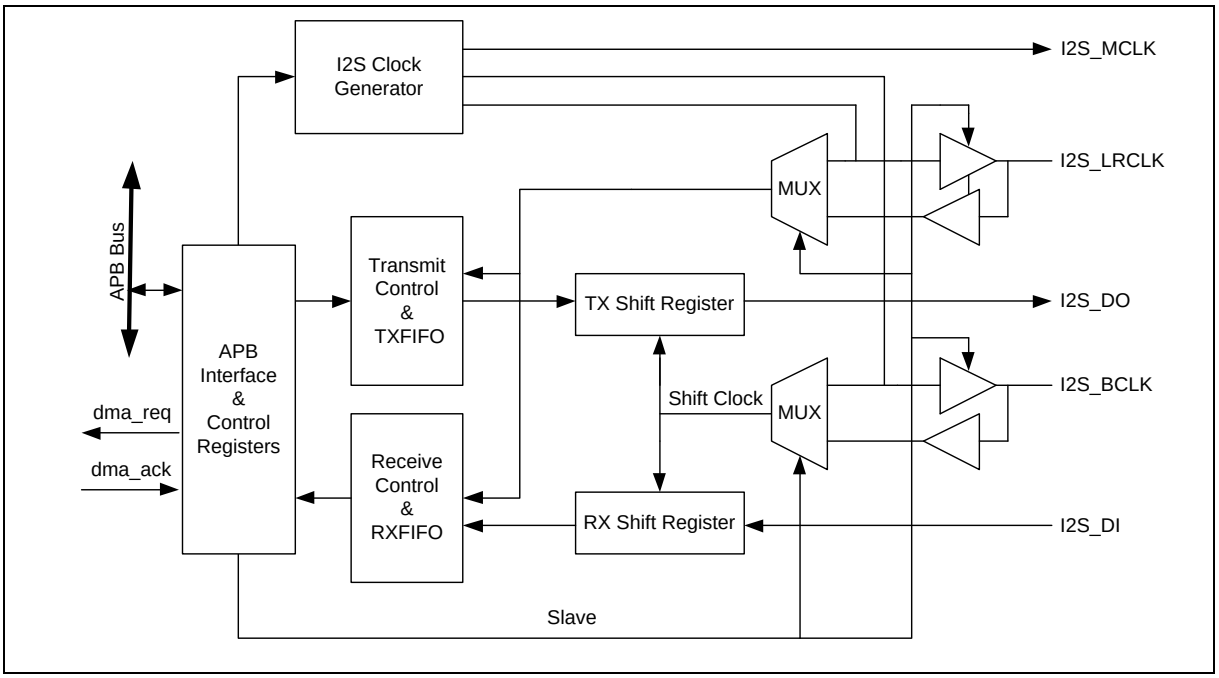


图 6.19-1I²S 控制器框架

6.19.4 基本配置

6.19.4.1 I²S基本配置

- 时钟源配置

- 在寄存器I2SSEL(CLK_CLKSEL3[17:16])配置I²S的外设时钟源
- 在寄存器I2SCKEN(CLK_APBCLK0[29])使能I²S的外设时钟

- 复位配置
  - 在寄存器I2SRST(SYS_IPRST1[29])复位I²S控制器
- 管脚配置

组	管脚名	GPIO	MFP
I²S0	I²S0_BCLK	PA.12	MFP2
		PE.8, PF.10	MFP4
		PE.1	MFP5
		PC.4	MFP6
		PB.5	MFP10
	I²S0_DI	PA.14	MFP2
		PE.10, PF.8	MFP4
		PH.8	MFP5
		PC.2	MFP6
		PB.3	MFP10
	I²S0_DO	PA.15	MFP2
		PE.11, PF.7	MFP4
		PH.9	MFP5
		PC.1	MFP6
		PB.2	MFP10
	I²S0_LRCK	PE.12, PF.6	MFP4
		PH.10	MFP5
		PC.0	MFP6
		PB.1	MFP10
	I²S0_MCLK	PA.13	MFP2
		PE.9, PF.9	MFP4
		PE.0	MFP5
		PC.3	MFP6
		PB.4	MFP10

6.19.5 功能描述

6.19.5.1 I²S时钟

在寄存器I2SSEL(CLK_CLKSEL3[17:16])里选择I²S的4个时钟源。I²S的时钟频率不能大于系统时钟的频

率。

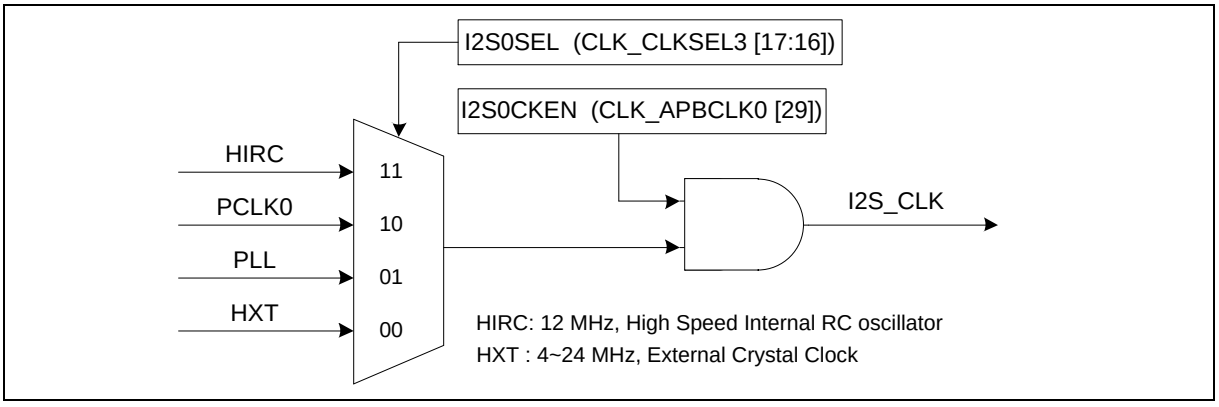


图 6.19-2I^{2}S 时钟控制框图

6.19.5.2 主机/从机接口

通过配置SLAVE(I2S_CTL0[8])寄存器可以切换I²S的主机或从机模式以连接其他的I²S从机或主机。串行总线的时钟I2S_BCLK由主机产生且不管有无数据传输都会一直存在。字选择信号I2S_LRCLK由主机产生，代表着新的数据字的开始和目标音频的通道。I2S_LRCLK和数据都在I2S_BCLK下降沿同步改变。

在某些应用里，尤其是音频-ADC和音频-DAC，主时钟信号I2S_MCLK和I2S_BCLK有一个固定的相位差。I2S_MCLK在寄存器MCLKEN(I2S_CTL0[15])里配置。主机模式下，主机输出I2S_MCLK,I2S_BCLK,I2S_LRCLK到从机。从机模式下，I2S_MCLK输出到主机，I2S_BCLK或I2S_LRCLK则由主机输入。

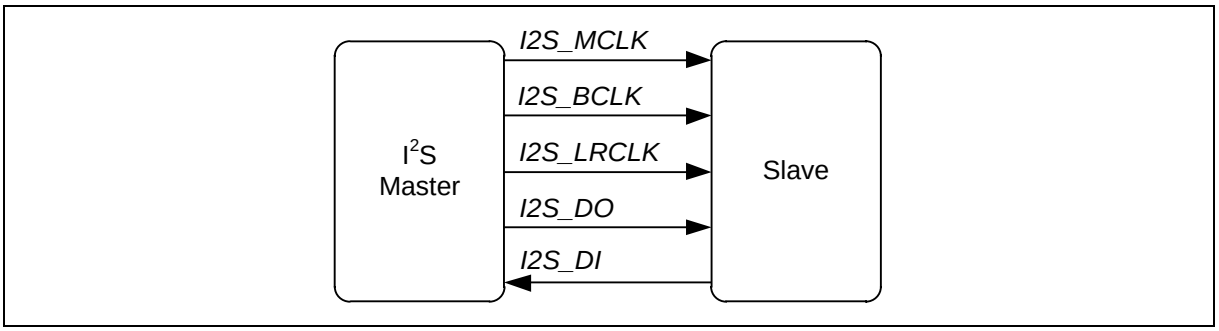


图 6.19-3 主机模式接口框图

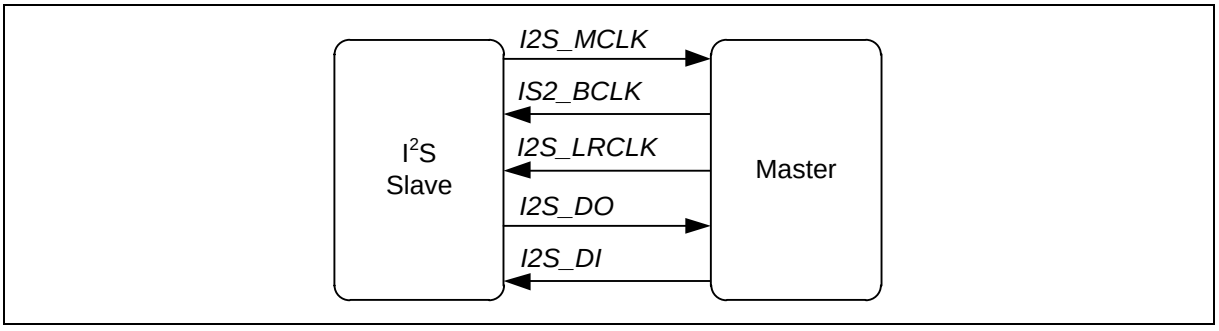


图 6.19-4 从机模式接口框图

6.19.5.3 I²S操作

I²S控制器支持飞利浦标准,MSB对齐和LSB对齐的数据格式。I2S_LRCLK信号指示当前传输的音频通道。音频通道的位宽度寄存器为CHWIDTH(I2S_CTL0[29:28])，数据字的位宽度寄存器为DATWIDTH(I2S_CTL0[5:4])。如果CHWIDTH(I2S_CTL0[29:28])小于DATWIDTH(I2S_CTL0[5:4])，硬件会将通道的位宽度更成和数据位宽度一致。如果，CHWIDTH(I2S_CTL0[29:28])大于DATWIDTH(I2S_CTL0[5:4])，硬件则会填充零位到每个音频通道。

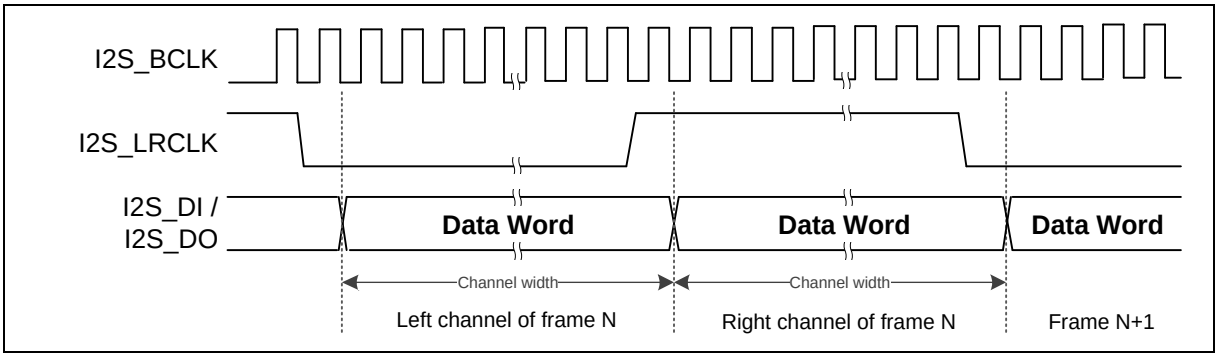


图 6.19-5I²S 通道宽度和数据宽度(CHWIDTH≦DATWIDTH)

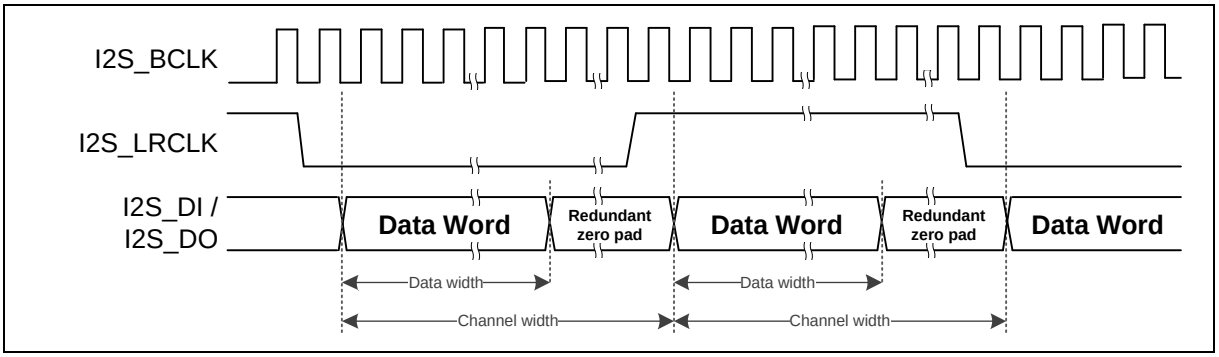


图 6.19-6 I²S 通道宽度和数据宽度(CHWIDTH>DATWIDTH)

数据传输的序列一直是从MSB到LSB。如图，按I²S协议，发送的数据在I2S_BCLK的上升沿被读取，在下降沿发送。在I²S的数据格式里，MSB在I2S_LRCLK变化后的下一个I2S_BCLK周期的下降沿发送并锁存。在MSB对齐的数据格式里，I2S_LRCLK在每个音频通道的第一个数据位(MSB)发送时发生变化。在LSB对齐的数据格式里，I2S_LRCLK在每个音频通道的最后一个I2S_BCLK周期发送并锁存。通过寄存器FORMAT(I2S_CTL0[26:24])可以切换I²S协议的MSB对齐和LSB对齐的数据格式。

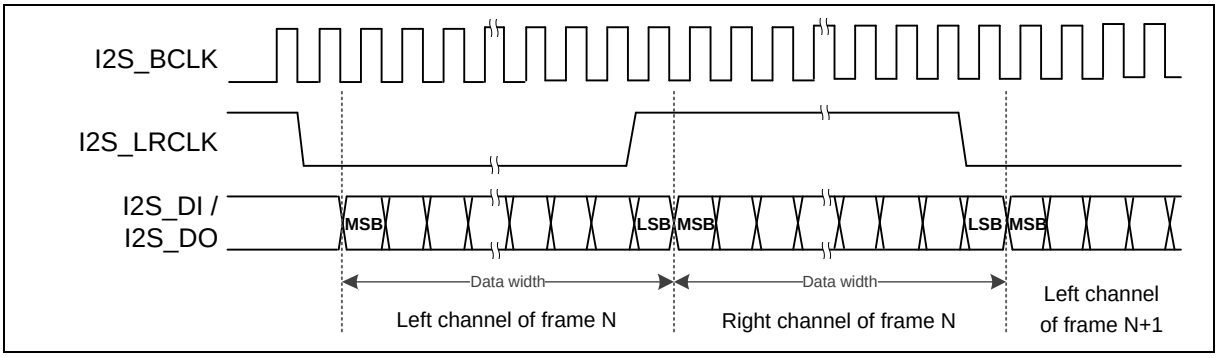


图 6.19-7 I²S 数据格式时序图(FORMAT=0x0;CHWIDTH≦DATWIDTH)

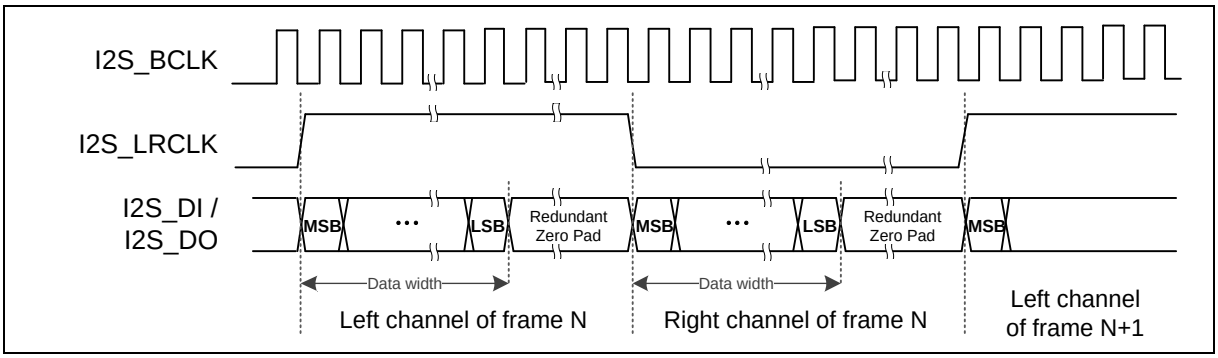


图 6.19-8MSB 对齐数据格式(FORMAT=0x1;CHWIDTH>DATWIDTH)

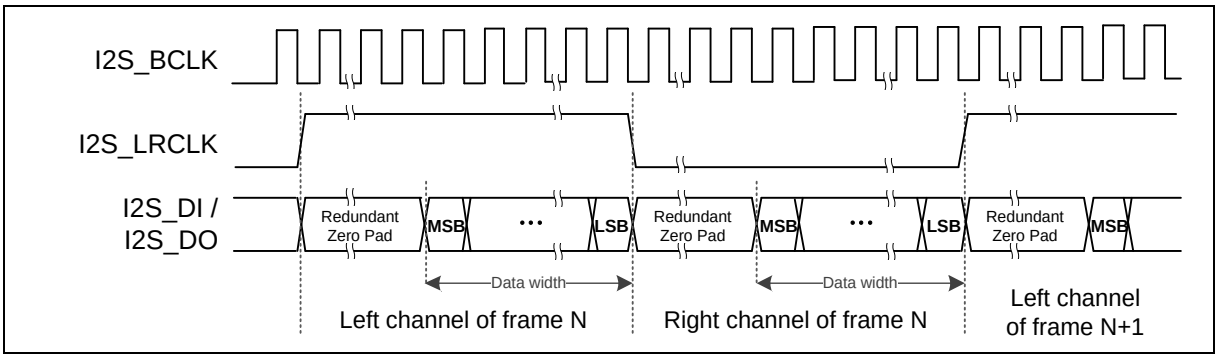


图 6.19-9LSB 对齐数据格式(FORMAT=0x2;CHWIDTH>DATWIDTH)

I²S控制器还支持PCM格式的音频传输，对应的寄存器使能位为FORMAT(I2S_CTL0[26:24])。在PCM协议里，I2S_LRCLK只是以一个上升沿的脉冲来作为音频段(或音频帧)的开始。因此，I2S_LRCLK在PCM协议里也被称为“帧开始”或“帧同步”信号。在主机模式下，用户可以用PCMSYNC(I2S_CTL0[27])来选择2种常用的“帧开始”脉冲宽度，一个是采用和通道宽度一致的脉冲宽度，另一个是采用一个I2S_BCLK周期的脉冲宽度。

和I²S模式一样，PCM模式下，用户可以用DATWIDTH(I2S_CTL0[5:4])和CHWIDTH(I2S_CTL0[29:28])选择不同的数据宽度和通道宽度。另外，通过FORMAT(I2S_CTL0[26:24])用户可以自由切换，PCM标准,MSB对齐和LSB对齐的数据格式三种数据格式。

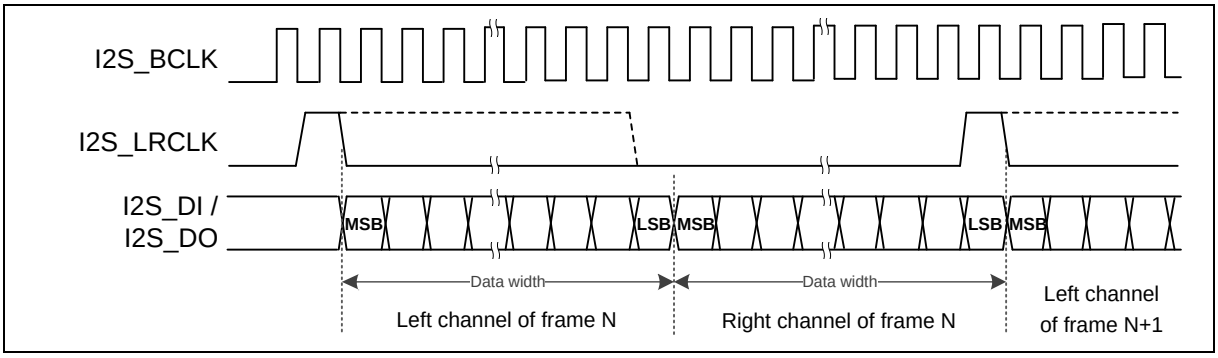


图 6.19-10 标准 PCM 音频时序图(FORMAT=0x4;CHWIDTH≦DATWIDTH)

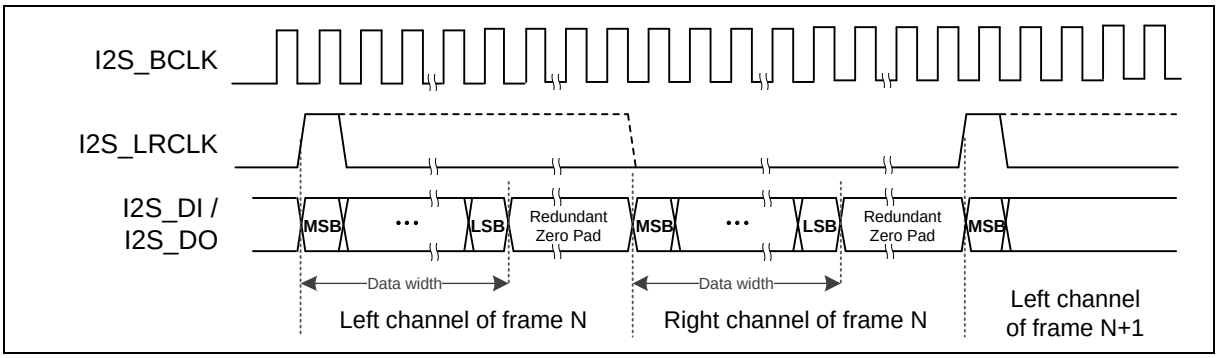


图 6.19-11MSB 对齐的 PCM 数据格式(FORMAT=0x5;CHWIDTH>DATWIDTH)

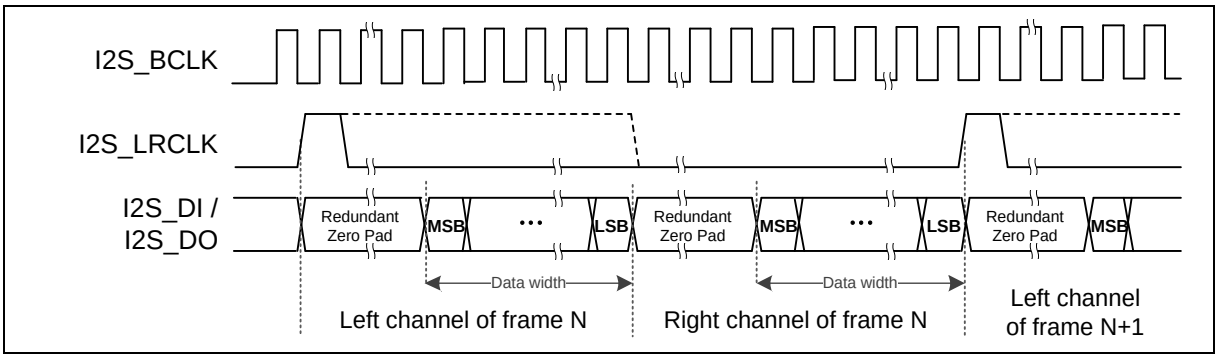


图 6.19-12LSB 对齐的 PCM 数据格式(FORMAT=0x6;CHWIDTH>DATWIDTH)

6.19.5.4 TDM多通道传输

I²S控制器处于PCM模式下时还支持TDM传输。分时多路复用(TDM)允许用户在一个数据线上传输多个通道的音频数据。TDM接口和2通道的PCM接口基本类似，不同的是它可以通过定义不同的I2S_LRCLK周期实现在一个采样帧里传输多个音频通道。TDM接口可以通过TDMCHNUM(I2S_CTL0[31:30])寄存器选择不同的通道数，典型值4个，6个或8个。

类似之前的I²S和PCM模式，每个通道块包含通过零位形成一个完整通道块的音频数据。用户可以用DATWIDTH(I2S_CTL0[5:4])和CHWIDTH(I2S_CTL0[29:28])选择不同的数据宽度和通道宽度。需要注意的是，使能了TDM的PCM模式支持16位，24位和32位宽度的音频数据，不支持8位宽度。所以，如果DATWIDTH(I2S_CTL0[5:4])被设成了0x0，硬件将会按16位宽度进行数据传输。帧开始信号的脉冲宽度则由PCMSYNC(I2S_CTL0[27])选择。

如图6.19-13是用32位通道块传输6通道24位音频数据的TDM传输示例。一般来说，对于2个通道的音频接口，我们称第一个为左声道，另一个为右声道(或者称为，通道0和通道1)。在TDM多路传输里，我们称之为通道0和通道1。

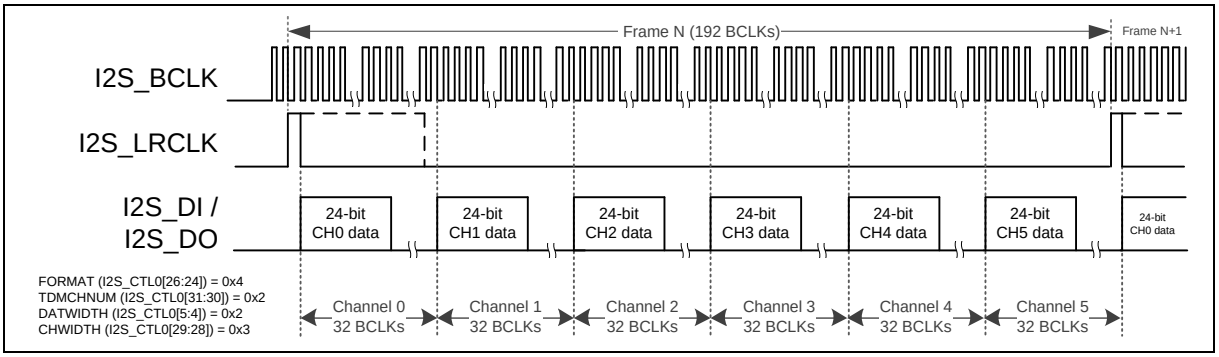


图 6.19-13 用 32 位通道块传输 6 通道 24 位音频数据的 TDM 传输(PCM 标准模式;FORMAT=0x4)

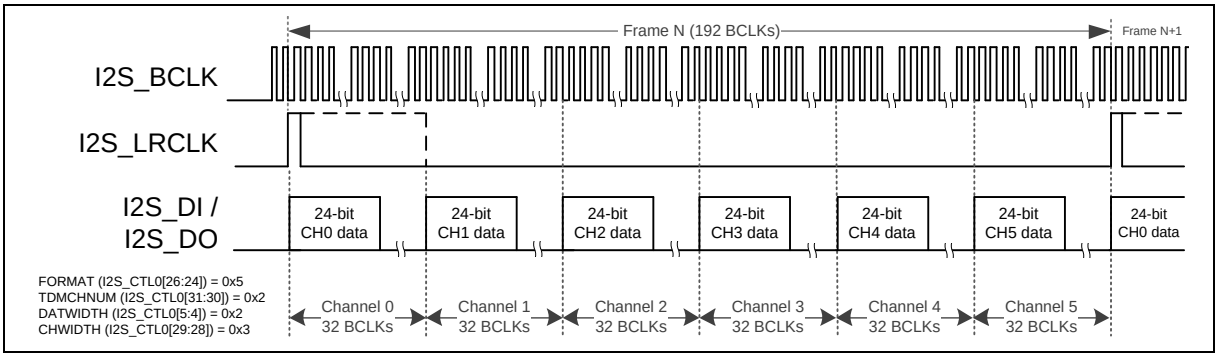


图 6.19-14 用 32 位通道块传输 6 通道 24 位音频数据的 TDM 传输(PCMMSB 对齐;FORMAT=0x5)

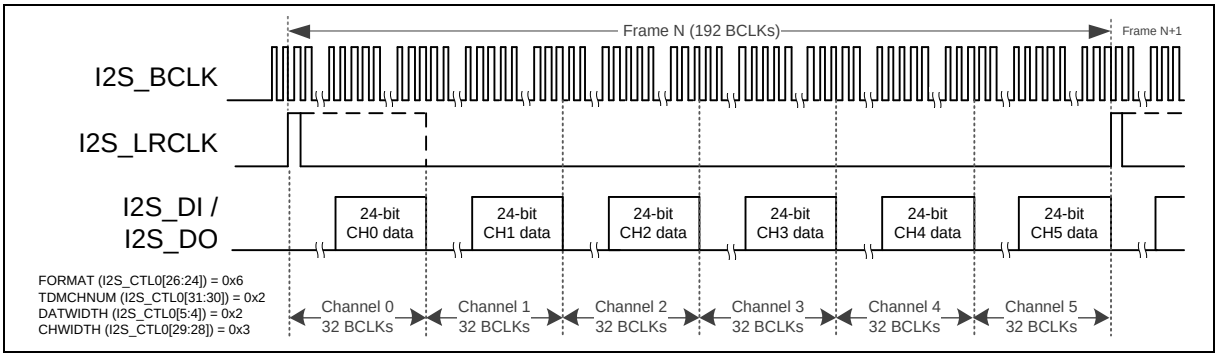


图 6.19-15 用 32 位通道块传输 6 通道 24 位音频数据的 TDM 传输(PCMLSB 对齐;FORMAT=0x6)

6.19.5.5 过零检测

I²S控制器播放音频时，通过CPU或PDMA把内存的数据发送出去。在播放过程中如果用户去切换增益时，很容易产生难听的爆破音。过零检测就是指在这个时候几乎关闭播放而避免产生噪音。过零检测中断可以用来指示增益的变化，从而避免声音产生很大的变化。

如果在寄存器CH0ZCEN到CH7ZCEN(I2S_CTL1[0]到I2S_CTL1[7])使能了对应通道的过零检测功能，硬件去检测对应通道的下一个传输的数据是否是零或者数据MSB是否发生了变化。使能零检测功能后，硬件在检测到对应通道上的零值或MSB变化时，对应通道的状态位CH0ZCIF到

CH7ZCIF(I2S_STATUS1[0]到I2S_STATUS1[7]))会被置位，且该通道会被自动静音(所有数据位都设为0)因此，如果用户想要避免爆破音，可以在修改音频增益前使能过零检测中断CH0ZCIEN到CH7ZCIEN(I2S_IEN[16:23])，然后在过零点去改变声音增益。

6.19.5.6 PDMA模式

I²S可以使能PDMA功能进行数据传输。如果使能了TXPDMAEN(I2S_CTL0[20])进行PDMA发送，I²S控制器会产生一个请求信号，如果TXFIFO未满，就会通过PDMA接口自动获取内存里要发送的音频数据。如果使能了RXPDMAEN(I2S_CTL0[19])进行PDMA接收，I²S控制器会产生一个请求信号，如果RXFIFO非空，就会通过PDMA接口自动把收到的音频数据搬到内存。因此，使用PDMA功能会减省CPU的时间用于其他功能。

6.19.5.7 I^2S中断源

发送时，I²S控制器支持独立音频通道的过零检测中断，发送FIFO阈值中断，发送FIFO溢出中断和发送FIFO下溢中断。接收时，则支持接收FIFO阈值中断，接收FIFO溢出中断和接收FIFO下溢中断。I²S中断发生后，用户可以检查I2STXINT(I2S_STATUS0[2])和I2SRXINT(I2S_STATUS0[1])标志来识别中断源。

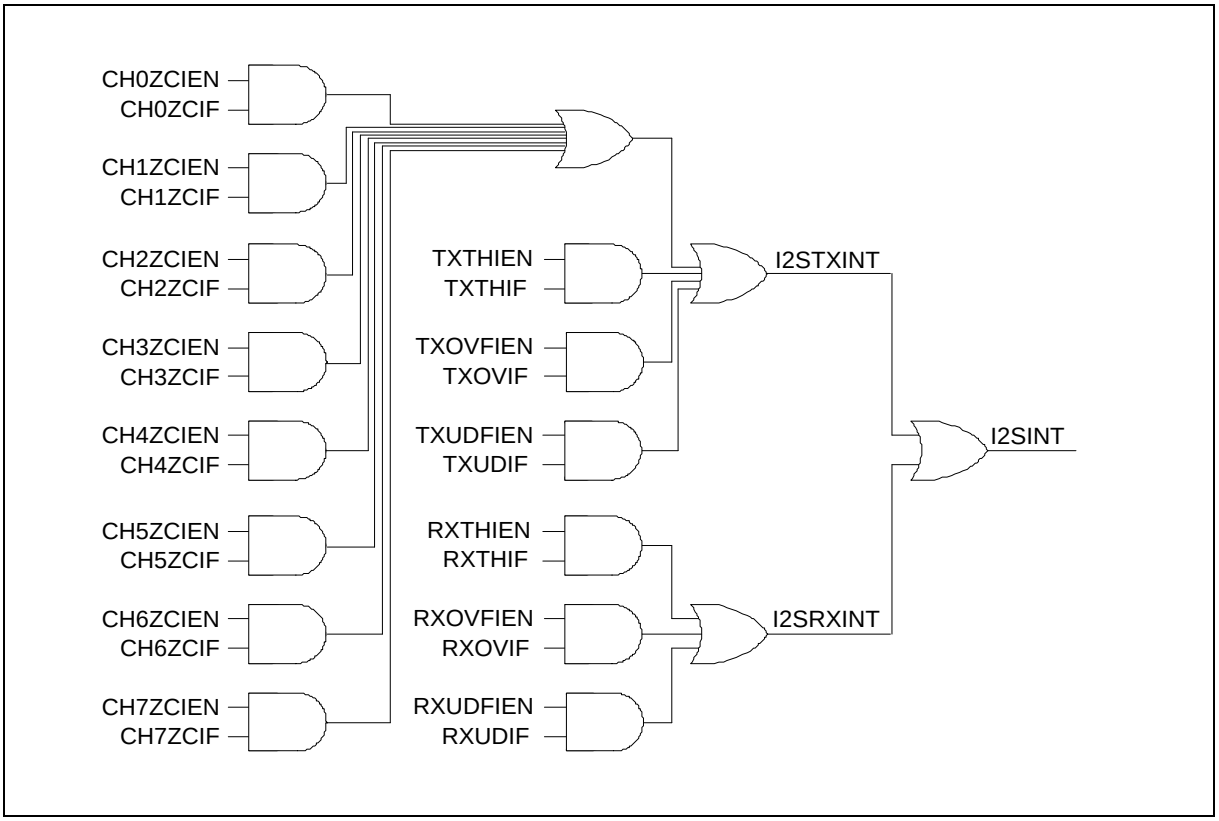


图 6.19-16 I^2S 中断

6.19.5.8 FIFO操作

在2路通道的I²S或PCM模式里，一个通道块的音频数据的位宽可以是8，16，24，或者32位。如图6.19-17是不同设置下音频数据的内存管理。

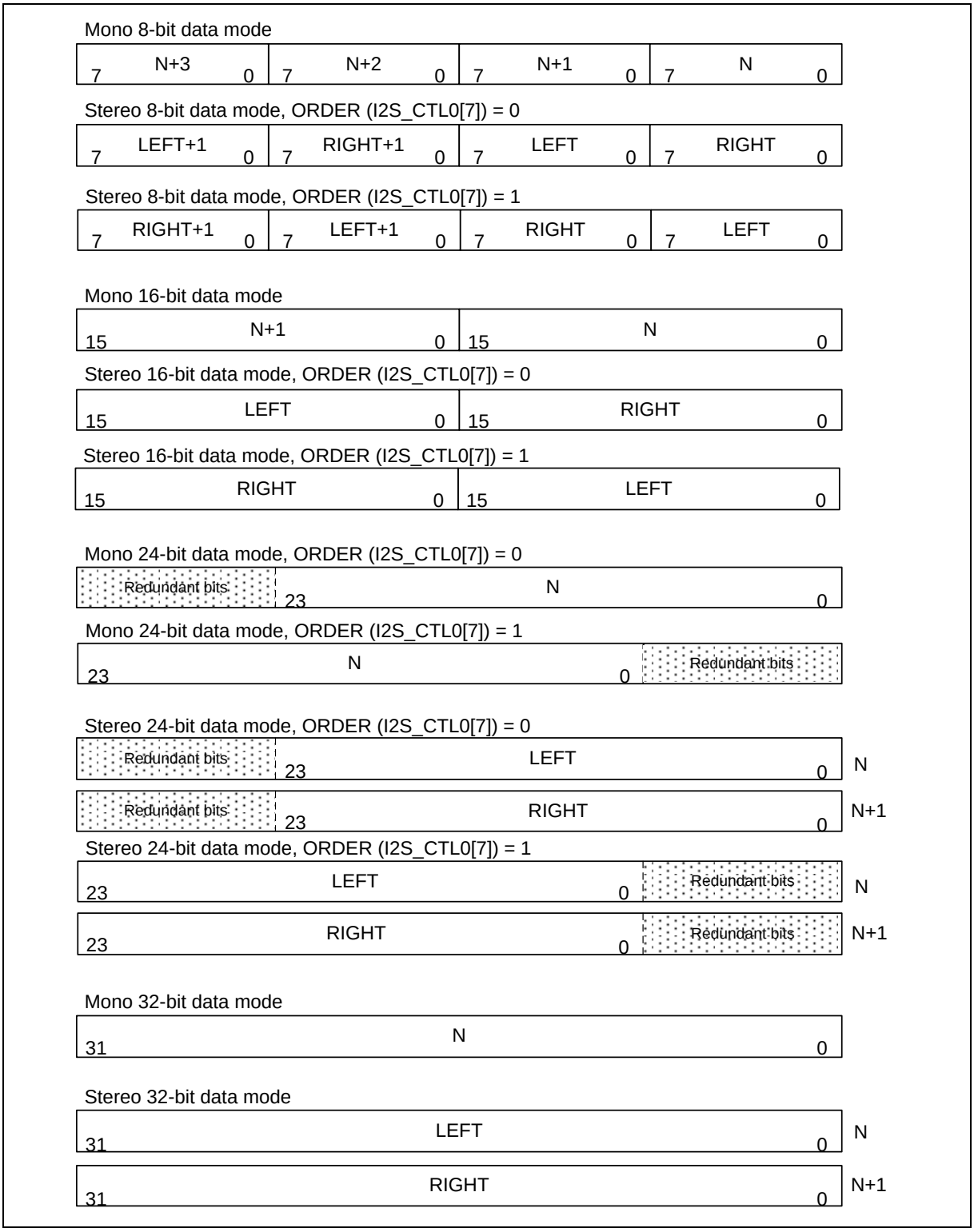


图 6.19-17 不同设置下 2 通道音频数据的 FIFO 示意

在4通道使能TDM的PCM模式里，一个通道块的音频数据的位宽可以是16，24，或者32位。如图6.19-18是不同设置下音频数据的内存管理。

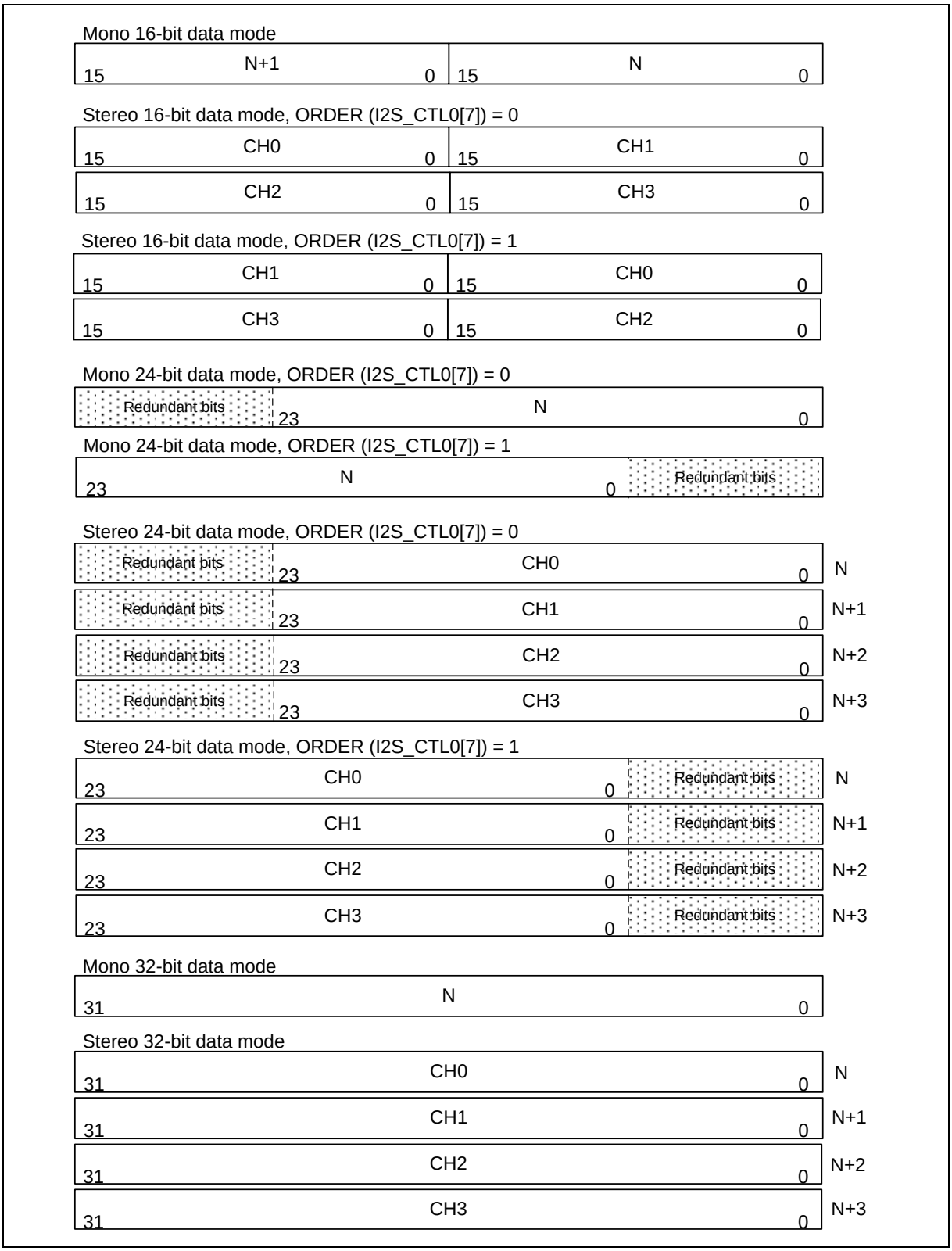


图 6.19-18 不同设置下 4 通道音频数据的 FIFO 示意

6通道TDMPCM数据格式，音频数据宽度可以是16,24,或32位，音频数据的内存分配各种设置如图6.19-19。16位音频数据传输时，可以用ORDER(I2S_CTL0[7])交换存储在发送和接收FIFO里的奇数位通道和偶数位通道。在24位音频数据传输时，ORDER(I2S_CTL0[7])可以用来选择存储在32位FIFO里的音频数据是左对齐还是右对齐。

8通道使能TDM的PCM模式和6通道的情况很类似，这里就不再赘述。

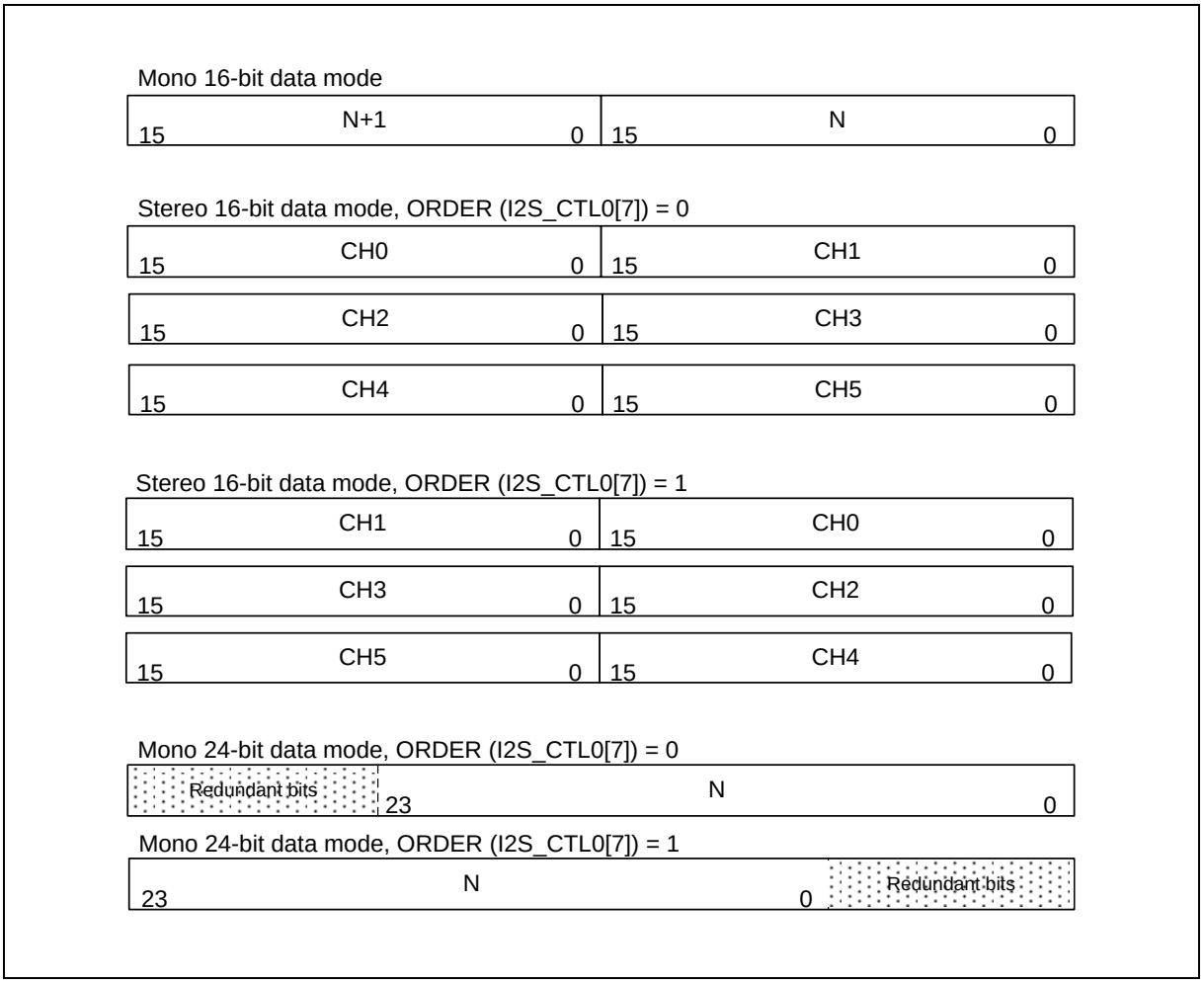


图 6.19-19 不同设置下 6 通道音频数据的 FIFO 示意(1)

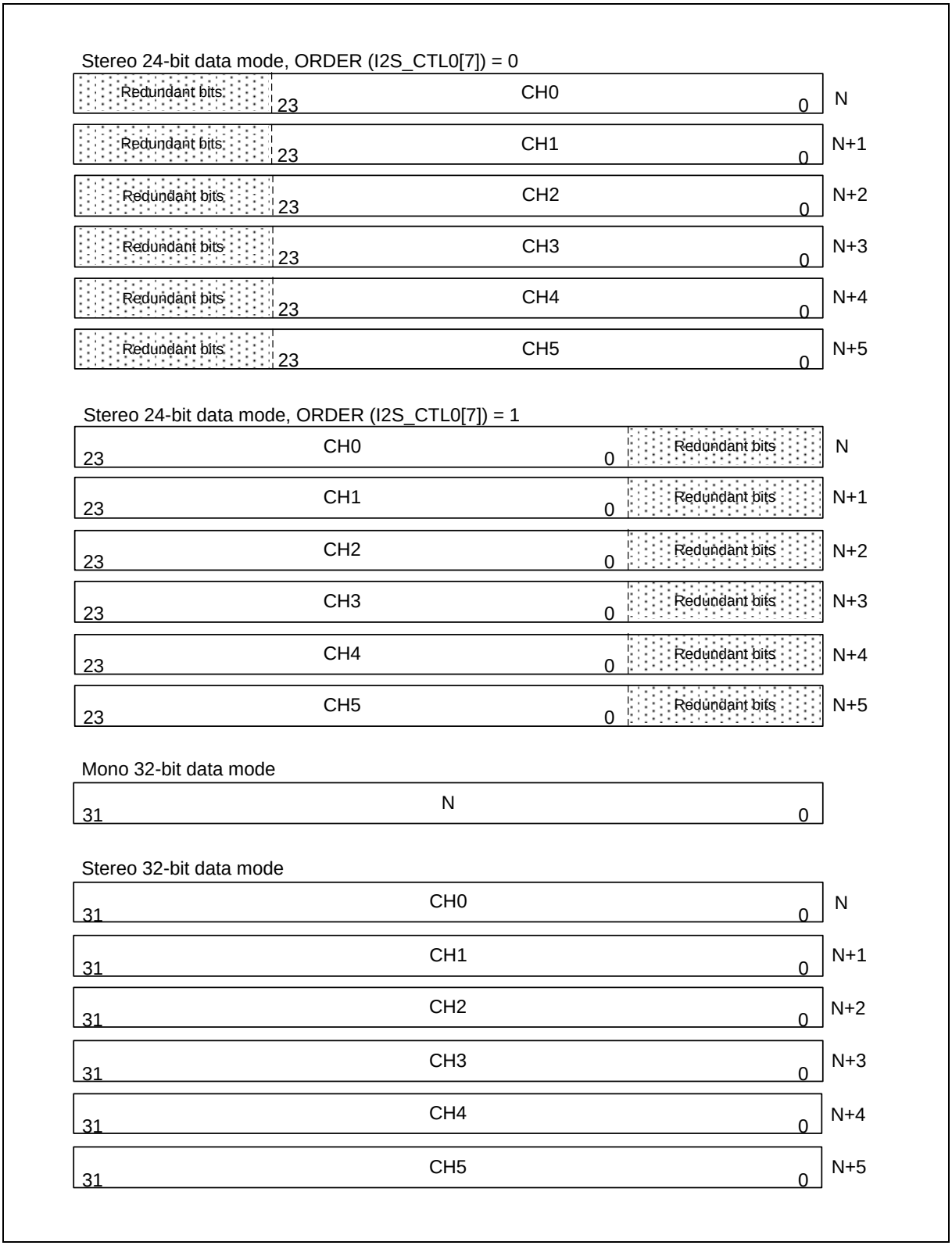


图 6.19-20 不同设置下 6 通道音频数据的 FIFO 示意(2)

6.19.6 寄存器映射

R: 只读，W: 只写，R/W: 读/写

寄存器	偏移量	R/W	描述	复位值
I²S 基地址 **I²S_BA = 0x4004_8000**				
I²S_CTL0	I²S_BA+0x00	R/W	I²S 控制寄存器 0	0x0000_0000
I²S_CTL1	I²S_BA+0x20	R/W	I²S 控制寄存器 1	0x0000_0000
I²S_CLKDIV	I²S_BA+0x04	R/W	I²S 时钟分频器寄存器	0x0000_0000
I²S_IEN	I²S_BA+0x08	R/W	I²S 中断使能寄存器	0x0000_0000
I²S_STATUS0	I²S_BA+0x0C	R/W	I²S 状态寄存器 0	0x0014_1038
I²S_STATUS1	I²S_BA+0x24	R/W	I²S 状态寄存器 1	0x0000_0000
I²S_TXFIFO	I²S_BA+0x10	W	I²S 发送 FIFO 寄存器	0x0000_0000
I²S_RXFIFO	I²S_BA+0x14	R	I²S 接收 FIFO 寄存器	0x0000_0000

6.19.7 寄存器描述

I²S 控制寄存器 0 (I²S_CTL0)

寄存器	偏移量	R/W	描述	复位值
I2S_CTL0	I2S_BA+0x00	R/W	I²S 控制寄存器 0	0x0000_0100

31	30	29	28	27	26	25	24
TDMCHNUM		CHWIDTH		PCMSYNC	FORMAT		
23	22	21	20	19	18	17	16
RXLCH	Reserved	RXPDMAEN	TXPDMAEN	RXFBCLR	TXFBCLR	Reserved	
15	14	13	12	11	10	9	8
MCLKEN	Reserved						SLAVE
7	6	5	4	3	2	1	0
ORDER	MONO	DATWIDTH		MUTE	RXEN	TXEN	I²SEN

Bits	Description	
[31:30]	**TDMCHNUM**	**TDM通道数** 该位用来定义PCM模式(FORMAT[2]=1)下，一个音频帧里的TDM通道数目 00=音频帧里有2个通道. 01=音频帧里有4个通道. 10=音频帧里有6个通道. 11=音频帧里有8个通道.
[29:28]	**CHWIDTH**	**通道宽度** 该位用来定义音频通道的宽度。如果CHWIDTH(I2S_CTL0[29:28])小于DATWIDTH(I2S_CTL0[5:4])，实际传输时，硬件会将通道的位宽度更成和数据位宽度一致 00=音频通道的位宽度为8位 01=音频通道的位宽度为16位 10=音频通道的位宽度为24位 11=音频通道的位宽度为32位
[27]	**PCMSYNC**	**PCM同步脉冲长度选择** 该位表示PCM协议里同步帧信号的高脉冲长度。 0=一个BCLK周期 1=一个通道周期 **注:**该位仅在主机模式下有效

[26:24]	**FORMAT**	**数据格式选择** 000=I²S标准格式 001=I²SMSB对齐 010=I²SLSB对齐 011=保留 100=PCM标准格式 101=PCMMSB对齐 110=PCMLSB对齐 111=保留
[23]	**RXLCH**	**左声道接收使能位** 选择了单声道格式(MONO=1)的情况下，如果RXLCH为0，I²S会接收通道1的数据，如果RXLCH为1，则接收通道0的数据。 0=单声道模式下接收通道1的数据 1=单声道模式下接收通道0的数据
[22]	**Reserved**	保留
[21]	**RXPDMAEN**	**PDMA接收使能** 0=禁止PDMA接收功能 1=使能PDMA接收功能
[20]	**TXPDMAEN**	**PDMA发送使能** 0=禁止PDMA发送功能 1=使能PDMA发送功能
[19]	**RXFBCLR**	**清接收FIFO** 0=无效 1=清RxFIFO **注1:**写1到该位清接收FIFO，内部指针会复位到FIFO起点，RXCNT(I2S_STATUS1[20:16])复位为0且接收FIFO会被清空。 **注2:**该位由硬件自动清0，读该位返回0.
[18]	**TXFBCLR**	**清发送FIFO** 0=无效 1=清TxFIFO. **注1:**写1到该位清发送FIFO，内部指针会复位到FIFO起点，TXCNT(I2S_STATUS1[12:8])复位为0且发送FIFO会被清空，但发送FIFO里的数据不会改变 **注2:**该位由硬件自动清0，读该位返回0.
[17:16]	**Reserved**	保留
[15]	**MCLKEN**	**主机时钟使能控制** 使能MCLKEN，I²S控制器会在I2S_MCLK管脚上为外部音频设备产生主机时钟 0=关闭主机时钟 1=使能主机时钟
[14:9]	**Reserved**	保留

[8]	**SLAVE**	**从机模式使能控制位** 0=主机模式 1=从机模式 **注:**I²S可以工作在主机模式或从机模式。主机模式下，I2S_BCLK和I2S_LRCLK管脚处于输出模式，主机输出I2S_BCLK和I2S_LRCLK到音频编解码芯片。从机模式下，I2S_BCLK和I2S_LRCLK管脚处于输入模式，I2S_BCLK和I2S_LRCLK则由音频编解码芯片输入。
[7]	**ORDER**	**FIFO内双声道数据顺序** 对于8/16位数据宽度，该位用来决定是奇数通道还是偶数通道存储在高字节。对于24位的数据宽度，该位则用于选择音频数据存储在32位数据FIFO时是左对齐还是右对齐。 0=8/16位数据宽度下，偶数通道存储在高字节 24位的LSB音频数据存储在32位数据FIFO时是右对齐 1=8/16位数据宽度下，偶数通道存储在低字节. 24位的MSB音频数据存储在32位数据FIFO时是左对齐.
[6]	**MONO**	**单声道数据控制** 0=双声道格式的数据 1=单声道格式的数据 **注:**录音时，若使能了单声道格式，RXLCH(I2S_CTL0[23])代表哪路声音数据会被录下来
[5:4]	**DATWIDTH**	**数据宽度** 该域表示每个音频通道的数据宽度 00=数据宽度为8位 01=数据宽度为16位 10=数据宽度为24位 11=数据宽度为32位
[3]	**MUTE**	**发送静音功能控制** 0=正常发送 1=发送0值到音频通道
[2]	**RXEN**	**接收使能控制** 0=禁止接收 1=使能接收
[1]	**TXEN**	**发送使能控制** 0=禁止发送 1=使能发送
[0]	**I^2SEN**	**I^2S控制器使能控制** 0=I^2S控制器禁止 1=I^2S控制器使能

I²S控制寄存器1(I²S_CTL1)

寄存器	偏移量	R/W	描述	复位值
I2S_CTL1	I2S_BA+0x20	R/W	I²S控制寄存器1	0x0000_0000

31	30	29	28	27	26	25	24
Reserved						PB16ORD	PBWIDTH
23	22	21	20	19	18	17	16
Reserved				RXTH			
15	14	13	12	11	10	9	8
Reserved				TXTH			
7	6	5	4	3	2	1	0
CH7ZCEN	CH6ZCEN	CH5ZCEN	CH4ZCEN	CH3ZCEN	CH2ZCEN	CH1ZCEN	CH0ZCEN

位	描述	
[31:26]	**Reserved**	保留.
[25]	**PB16ORD**	**16位外设总线R/WFIFO顺序** PBWIDTH为1时，数据FIFO由2条外设总线进行存取。该位用来选择外设总线对32位FIFO进行操作时的顺序。 0=先存取低16位 1=先存取高16位 **注:**仅当PBWIDTH=1时可用
[24]	**PBWIDTH**	**外设总线数据宽度选择** 该位用来选择外设总线APB的数据宽度。使能PDMA功能时，必须置该位为1进入16位的传输模式。 0=32位数据宽度 1=16位数据宽度 **注1:**如果PBWIDTH=1，32位数据总线的低16位可用。 **注2:**如果PBWIDTH=1，2条FIFO写操作完成后，发送FIFO的深度加一。 **注3:**如果PBWIDTH=1，2条FIFO读操作完成后，接收FIFO的深度减一。
[23:20]	**Reserved**	保留

[19:16]	**RXTH**	**接收FIFO阈值水平** 0000=接收FIFO阈值为1数据字 0001=接收FIFO阈值为2数据字 0010=接收FIFO阈值为3数据字 …. 1110=接收FIFO阈值为15数据字 1111=接收FIFO阈值为16数据字 **注:**当接收到的数据数超过阈值水平时，RXTHIF(I2S_STATUS0[10])会被置位
[15:12]	**Reserved**	保留
[11:8]	**TXTH**	**发送FIFO阈值水平** 0000=发送FIFO阈值为0数据字 0001=发送FIFO阈值为1数据字 0010=发送FIFO阈值为2数据字 .... 1110=发送FIFO阈值为14数据字 1111=发送FIFO阈值为15数据字 **注:**当发送FIFO里剩余的数据数小于等于发送阈值水平时，TXTHIF(I2S_STATUS0[18])会被置位
[7]	**CH7ZCEN**	**通道7过零检测使能控制** 0=通道7过零检测关闭 1=通道7过零检测使能 **注1:**该位仅在多路通道PCM模式，TDMCHNUM(I2S_CTL0[31:30])=0x1,0x2,0x3时可用 **注 2:** 如果置位该位，当通道7数据信号位改变或下一个数据所有位都为0，CH7ZCIF(I2S_STATUS1[7])会被置位 **注3:**CH7ZCIF标志被置位时，通道7会被静音
[6]	**CH6ZCEN**	**通道6过零检测使能控制** 0=通道6过零检测关闭 1=通道6过零检测使能 **注1:**该位仅在多路通道PCM模式，TDMCHNUM(I2S_CTL0[31:30])=0x1,0x2,0x3时可用 **注 2:** 如果置位该位，当通道6数据信号位改变或下一个数据所有位都为0，CH6ZCIF(I2S_STATUS1[6])会被置位 **注3:**CH6ZCIF标志被置位时，通道6会被静音
[5]	**CH5ZCEN**	**通道5过零检测使能控制** 0=通道5过零检测关闭 1=通道5过零检测使能 **注1:**该位仅在多路通道PCM模式，TDMCHNUM(I2S_CTL0[31:30])=0x1,0x2,0x3时可用 **注 2:** 如果置位该位，当通道5数据信号位改变或下一个数据所有位都为0，CH5ZCIF(I2S_STATUS1[5])会被置位 **注3:**CH5ZCIF标志被置位时，通道5会被静音

[4]	**CH4ZCEN**	**通道4过零检测使能控制** 0=通道4过零检测关闭 1=通道4过零检测使能 **注1:**该位仅在多路通道PCM模式，TDMCHNUM(I2S_CTL0[31:30])=0x1,0x2,0x3时可用 **注 2:** 如果置位该位，当通道4数据信号位改变或下一个数据所有位都为0，CH4ZCIF(I2S_STATUS1[4])会被置位 **注3:**CH4ZCIF标志被置位时，通道4会被静音
[3]	**CH3ZCEN**	**通道3过零检测使能控制** 0=通道3过零检测关闭 1=通道3过零检测使能 **注1:**该位仅在多路通道PCM模式，TDMCHNUM(I2S_CTL0[31:30])=0x1,0x2,0x3时可用 **注 2:** 如果置位该位，当通道3数据信号位改变或下一个数据所有位都为0，CH3ZCIF(I2S_STATUS1[3])会被置位 **注3:**CH3ZCIF标志被置位时，通道3会被静音
[2]	**CH2ZCEN**	**通道2过零检测使能控制** 0=通道2过零检测关闭 1=通道2过零检测使能 **注1:**该位仅在多路通道PCM模式，TDMCHNUM(I2S_CTL0[31:30])=0x1,0x2,0x3时可用 **注 2:** 如果置位该位，当通道2数据信号位改变或下一个数据所有位都为0，CH2ZCIF(I2S_STATUS1[2])会被置位 **注3:**CH2ZCIF标志被置位时，通道2会被静音
[1]	**CH1ZCEN**	**通道1过零检测使能控制** 0=通道1过零检测关闭 1=通道1过零检测使能 **注1:**通道1在I2S(FORMAT[2]=0)模式下或2通道的PCM模式下也指右声道 **注 2:** 如果置位该位，当通道1数据信号位改变或下一个数据所有位都为0，CH1ZCIF(I2S_STATUS1[1])会被置位 **注3:**CH1ZCIF标志被置位时，通道1会被静音
[0]	**CH0ZCEN**	**过零检测使能控制** 0=通道0过零检测关闭 1=通道0过零检测使能 **注1:**通道0在I2S(FORMAT[2]=0)模式下或2通道的PCM模式下也指右声道 **注 2:** 如果置位该位，当通道0数据信号位改变或下一个数据所有位都为0，CH2ZCIF(I2S_STATUS1[2])会被置位 **注3:**CH0ZCIF标志被置位时，通道0会被静音

I²S时钟分频器寄存器(I²S_CLKDIV)

寄存器	偏移量	R/W	描述	复位值
I²S_CLKDIV	I²S_BA+0x04	R/W	I²S时钟分频器寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved						BCLKDIV	
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
BCLKDIV							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved	MCLKDIV						

位	描述	
[31:18]	**Reserved**	保留
[17:8]	**BCLKDIV**	**位时钟除频** 主机模式下I²S控制器产生位时钟。软件可以配置该寄存器产生特定频率的采样时钟。 F_BCLK=F_I2SCLK/(2*(BCLKDIV+1)). **注:**F_BCLK为BCLK的频率，F_I2SCLK为I2S_CLK的频率
[7]	**Reserved**	保留
[6:0]	**MCLKDIV**	**主时钟除频** 如果外部晶振频率为(2xMCLKDIV)*256fs，则软件可以配置该寄存器产生256fs的时钟频率到音频编解码器芯片。如果MCLKDIV为0，MCLK和外部输入时钟一致 例如，采样率为24KHz，芯片外部时钟为12.288MHz，置MCLKDIV为1 F_MCLK=F_I2SCLK/(2x(MCLKDIV))(MCLKDIV>=1). F_MCLK=F_I2SCLK(MCLKDIV=0). **注:**F_MCLK为MCLK的频率，F_I2SCLK为I2S_CLK的频率

I²S中断使能寄存器(I²S_IEN)

寄存器	偏移量	R/W	描述	复位值
I2S_IEN	I2S_BA+0x08	R/W	I²S中断使能寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
CH7ZCIEN	CH6ZCIEN	CH5ZCIEN	CH4ZCIEN	CH3ZCIEN	CH2ZCIEN	CH1ZCIEN	CH0ZCIEN
15	14	13	12	11	10	9	8
Reserved					TXTHIEN	TXOVFIEN	TXUDFIEN
7	6	5	4	3	2	1	0
Reserved					RXTHIEN	RXOVFIEN	RXUDFIEN

位	描述	
[31:24]	**Reserved**	保留
[23]	**CH7ZCIEN**	**通道7过零检测中断使能控制** 0=关闭中断 1=使能中断 **注1:**该位置位时，如果通道7检测到过零，则产生中断 **注2:**该位仅在多路通道PCM模式和TDMCHNUM(I2S_CTL0[31:30])=0x3时可效
[22]	**CH6ZCIEN**	**通道6过零检测中断使能控制** 0=关闭中断 1=使能中断 **注1:**该位置位时，如果通道6检测到过零，则产生中断 **注2:**该位仅在多路通道PCM模式和TDMCHNUM(I2S_CTL0[31:30])=0x3时可效
[21]	**CH5ZCIEN**	**通道5过零检测中断使能控制** 0=关闭中断 1=使能中断 **注1:**该位置位时，如果通道5检测到过零，则产生中断 **注2:**该位仅在多路通道PCM模式和TDMCHNUM(I2S_CTL0[31:30])=0x3时可效
[20]	**CH4ZCIEN**	**通道4过零检测中断使能控制** 0=关闭中断 1=使能中断 **注1:**该位置位时，如果通道4检测到过零，则产生中断 **注2:**该位仅在多路通道PCM模式和TDMCHNUM(I2S_CTL0[31:30])=0x3时可效

[19]	**CH3ZCIEN**	**通道3过零检测中断使能控制** 0=关闭中断 1=使能中断 **注1:**该位置位时，如果通道3检测到过零，则产生中断 **注2:**该位仅在多路通道PCM模式和TDMCHNUM(I2S_CTL0[31:30])=0x3时可效
[18]	**CH2ZCIEN**	**通道 2 过零检测中断使能控制** 0 = 关闭中断 1 = 使能中断 **注 1:** 该位置位时，如果通道 2 检测到过零，则产生中断 **注 2:** 该位仅在多路通道 PCM 模式和TDMCHNUM (I2S_CTL0[31:30])= 0x3 时可效
[17]	**CH1ZCIEN**	**通道 1 过零检测中断使能控制** 0 = 关闭中断 1 = 使能中断 **注 1:** 该位置位时，如果通道 1 检测到过零，则产生中断 **注 2:** 通道 1 在I2S (FORMAT[2]=0) 模式下或 2 通道的 PCM 模式下也指右声道
[16]	**CH0ZCIEN**	**通道 0 过零检测中断使能控制** 0 = 关闭中断 1 = 使能中断 **注 1:** 该位置位时，如果通道 0 检测到过零，则产生中断 **注 2:** 通道 0 在I2S (FORMAT[2]=0) 模式下或 2 通道的 PCM 模式下也指左声道
[15:11]	**Reserved**	保留
[10]	**TXTHIEN**	**发送 FIFO 阈值水平中断使能控制** 0 = 关闭中断 1 = 使能中断 **注:** 该位置位时，如果发送 FIFO 里的数据少于TXTH (I2S_CTL1[11:8]) 则发生中断
[9]	**TXOVFIEN**	**发送 FIFO 溢出中断使能控制** 0 = 关闭中断 1 = 使能中断 **注:** 该位置位时，如果TXOVIF (I2S_STATUS0[17]) 也置位，则发生中断中断
[8]	**TXUDFIEN**	**发送 FIFO 下溢中断使能控制** 0 = 关闭中断 1 = 使能中断 **注:** 该位置位时，如果TXUDIF (I2S_STATUS0[16]) 也置位，则发生中断中断
[7:3]	**Reserved**	保留
[2]	**RXTHIEN**	**接收 FIFO 阈值水平中断使能控制** 0 = 关闭中断 1 = 使能中断 **注:** 当接收 FIFO 的数据大于等于 RXTH (I2S_CTL1[19:16]) 且RXTHIF (I2S_STATUS0[10]) 为 1，如果使能了RXTHIEN，则会发生中断

[1]	**RXOVFIEN**	**接收 FIFO 溢出中断使能控制** 0 = 关闭中断 1 = 使能中断 **注:** 该位置位时，如果RXOVIF (I2S_STATUS0[9]) 也置位，则发生中断中断
[0]	**RXUDFIEN**	**接收 FIFO 下溢中断使能控制** 0 = 关闭中断 1 = 使能中断 **注:** 接收 FIFO 空时，软件读接收 FIFO 就会导致RXUDIF (I2S_STATUS0[8]) 置位

I²S状态寄存器0 (I²S_STATUS0)

寄存器	偏移量	R/W	描述	复位值
I2S_STATUS0	I2S_BA+0x0C	R/W	I²S 状态寄存器 0	0x0014_1000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved		TXBUSY	TXEMPTY	TXFULL	TXTHIF	TXOVIF	TXUDIF
15	14	13	12	11	10	9	8
Reserved			RXEMPTY	RXFULL	RXTHIF	RXOVIF	RXUDIF
7	6	5	4	3	2	1	0
Reserved		DATACH			I²STXINT	I²SRXINT	I²SINT

位	描述	
[31:22]	**Reserved**	保留
[21]	**TXBUSY**	**发送忙 (只读)** 0 = 发送移位缓存空 1 = 发送移位缓存忙 **注:** 当发送 FIFO 和移位缓存里的数据都被移出时，该位清 0。当有数据载入移位缓存时，该位置 1。
[20]	**TXEMPTY**	**发送 FIFO 空 (只读)** 该为指示发送 FIFO 是否为空 0 = 非空 1 = 空
[19]	**TXFULL**	**发送 FIFO 满 (只读)** 该位指示发送 FIFO 是否满 0 = 未满 1 = 满.
[18]	**TXTHIF**	**发送 FIFO 域值中断标志 (只读)** 0 = FIFO 里的数据超过了阈值水平 1 = FIFO 里的数据低于阈值水平 **注:** 当 FIFO 里的的数据小于等于TXTH (I2S_CTL1[11:8]) 的阈值设定，TXTHIF 该置 1。软件写 TXFIFO 寄存器后，当 TXCNT (I2S_STATUS1[12:8]) 超过 TXTH (I2S_CTL1[11:8]) 时，该位才会改变

[17]	**TXOVIF**	**发送 FIFO 溢出中断标志** 0 = 未溢出 1 = 溢出. **注 1:** 发送 FIFO 满时还向 FIFO 写数据会导致溢出 **注 2:** 写 1 清 0 该位.
[16]	**TXUDIF**	**发送 FIFO 下溢中断标志** 0 = 未下溢 1 = 下溢. **注 1:** 当移位逻辑电路从发送 FIFO 读数据，发送 FIFO 里的数据量小于一帧音频的数据量时，该位置位 **注 2:** 写 1 清 0 该位.
[15:13]	**Reserved**	保留
[12]	**RXEMPTY**	**接收 FIFO 空 (只读)** 0 = 非空. 1 = 空. **注:**该位用来指示接收 FIFO 是否为空
[11]	**RXFULL**	**接收 FIFO 满 (只读)** 0 = 未满 1 = 满. **注:** 该位用来指示接收 FIFO 是否已满
[10]	**RXTHIF**	**接收 FIFO 阈值中断标志 (只读)** 0 = 接收 FIFO 里的数据未超过阈值水平 1 = 接收 FIFO 里的数据超过了阈值水平 **注:** 当接收 FIFO 里的数据未超过了RXTH (I2S_CTL1[19:16]) 设定的阈值水平，该位置 1。软件读 RXFIFO 寄存器后，当 RXCNT (I2S_STATUS1[20:16]) 小于 RXTH (I2S_CTL1[19:16]) 时，该位才会改变
[9]	**RXOVIF**	**接收 FIFO 溢出中断标志** 0 = 未溢出 1 = 溢出 **注 1:** 接收 FIFO 已满，硬件仍向接收 FIFO 收数据会导致该位置 1，此时第一个缓存里的数据会被覆盖 **注 2:** 写 1 清 0 该位.
[8]	**RXUDIF**	**接收 FIFO 下溢中断标志** 0 = 未下溢. 1 = 下溢 **注 1:**当接收 FIFO 为空时，软件还去读接收 FIFO 会导致该位置 1，这种情况就是下溢 **注 2:** 写 1 清 0 该位.
[7:6]	**Reserved**	保留

[5:3]	**DATACH**	**传输数据通道 (只读)** 该位表示当前是哪一个通道在传输数据 000 = 通道0 (即 2 通道的 I²S/PCM 模式下的左声道). 001 = 通道1 (即 2 通道的 I²S/PCM 模式下的右声道). 010 = 通道 2 (用于使能了TDM 的 PCM 模式下的 4 通道传输) 011 = 通道 3 (用于使能了TDM 的 PCM 模式下的 4 通道传输) 100 = 通道 4 (用于使能了TDM 的 PCM 模式下的 6 通道传输) 101 = 通道 5 (用于使能了TDM 的 PCM 模式下的 6 通道传输) 110 = 通道 6 (用于使能了TDM 的 PCM 模式下的 8 通道传输) 111 = 通道 7 (用于使能了 TDM 的 PCM 模式下的 8 通道传输)
[2]	**I²STXINT**	**I²S 发送中断(只读)** 0 = 无发送中断 1 = 有发送中断
[1]	**I²SRXINT**	**I²S 接收中断(只读)** 0 = 无接收中断 1 = 有接收中断
[0]	**I²SINT**	**I²S 中断标志(只读)** 0 = 无 I²S 中断 1 = 有 I²S 中断 **注:** 该位为I2STXINT “或” I2SRXINT 的结构.

I²S状态寄存器1 (I²S_STATUS1)

寄存器	偏移量	R/W	描述	复位值
I2S_STATUS1	I2S_BA+0x24	R/W	I²S 状态寄存器 1	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved			RXCNT				
15	14	13	12	11	10	9	8
Reserved			TXCNT				
7	6	5	4	3	2	1	0
CH7ZCIF	CH6ZCIF	CH5ZCIF	CH4ZCIF	CH3ZCIF	CH2ZCIF	CH1ZCIF	CH0ZCIF

位	描述	
[31:21]	**Reserved**	保留
[20:16]	**RXCNT**	**接收 FIFO 水平 (只读)** 该位表示接收 FIFO 里的可用数据个数 00000 = 无数据. 00001 = 接受 FIFO 里有 1 个字数据. 00010 = 接受 FIFO 里有 2个字数据. …. 01110 = 接受 FIFO 里有 14个字数据. 01111 = 接受 FIFO 里有 15个字数据. 10000 = 接受 FIFO 里有 16个字数据. 其他保留
[15:13]	**Reserved**	保留
[12:8]	**TXCNT**	**发送 FIFO 水平 (只读)** 该位表示发送 FIFO 里的可用数据个数 00000 = 无数据. 00001 = 发送 FIFO 里有 1个字数据 00010 = 发送 FIFO 里有 2个字数据. …. 01110 = 发送 FIFO 里有 14个字数据. 01111 = 发送 FIFO 里有 15个字数据. 10000 = 发送 FIFO 里有 16个字数据. 其他保留

[7]	**CH7ZCIF**	**通道 7 过零中断标志** 该位表示通道 7 下一笔数据符号位改变或所有位均为 0 0 = 通道 7 未检测到过零 1 = 通道 7 检测到过零. **注 1:** 写 1 清 0 该位. **注 2:** 该位仅在多路通道 PCM 模式和TDMCHNUM (I2S_CTL0[31:30]) = 0x3 时可用
[6]	**CH6ZCIF**	**通道 6 过零中断标志** 该位表示通道 6 下一笔数据符号位改变或所有位均为 0 0 = 通道 6 未检测到过零 1 = 通道 6 检测到过零. **注 1:** 写 1 清 0 该位. **注 2:** 该位仅在多路通道 PCM 模式和TDMCHNUM (I2S_CTL0[31:30]) = 0x3 时可用
[5]	**CH5ZCIF**	**通道 5 过零中断标志** 该位表示通道 5 下一笔数据符号位改变或所有位均为 0 0 = 通道 5 未检测到过零 1 = 通道 5 检测到过零. **注 1:** 写 1 清 0 该位. **注 2:** 该位仅在多路通道 PCM 模式和TDMCHNUM (I2S_CTL0[31:30]) = 0x3 时可用
[4]	**CH4ZCIF**	**通道 4 过零中断标志** 该位表示通道 4 下一笔数据符号位改变或所有位均为 0 0 = 通道 4 未检测到过零 1 = 通道 4 检测到过零. **注 1:** 写 1 清 0 该位. **注 2:** 该位仅在多路通道 PCM 模式和TDMCHNUM (I2S_CTL0[31:30]) = 0x3 时可用
[3]	**CH3ZCIF**	**通道 3 过零中断标志** 该位表示通道 3 下一笔数据符号位改变或所有位均为 0 0 = 通道 3 未检测到过零 1 = 通道 3 检测到过零. **注 1:** 写 1 清 0 该位. **注 2:** 该位仅在多路通道 PCM 模式和TDMCHNUM (I2S_CTL0[31:30]) = 0x3 时可用
[2]	**CH2ZCIF**	**通道 2 过零中断标志** 该位表示通道 2 下一笔数据符号位改变或所有位均为 0 0 = 通道 2 未检测到过零 1 = 通道 2 检测到过零. **注 1:** 写 1 清 0 该位. **注 2:** 该位仅在多路通道 PCM 模式和TDMCHNUM (I2S_CTL0[31:30]) = 0x3 时可用

[1]	**CH1ZCIF**	**通道 1 过零中断标志** 该位表示通道 1 下一笔数符号号位改变或所有位均为 0 0 = 通道 1 未检测到过零 1 = 通道 1 检测到过零. **注 1:** 写 1 清 0 该位. **注 2:** 通道 1 在I2S (FORMAT[2]=0) 模式下或 2 通道的 PCM 模式下也指右声道
[0]	**CH0ZCIF**	**通道 0 过零中断标志** 该位表示通道 0 下一笔数据符号位改变或所有位均为 0 0 = 通道 0 未检测到过零 1 = 通道 0 检测到过零. **注 1:** 写 1 清 0 该位. **注 2:** 通道 0 在I2S (FORMAT[2]=0) 模式下或 2 通道的 PCM 模式下也指左声道

I²S发送FIFO寄存器(I²S_TXFIFO)

寄存器	偏移量	R/W	描述	复位值
I2S_TXFIFO	I2S_BA+0x10	W	I²S发送FIFO寄存器	0x0000_0000

31	30	29	28	27	26	25	24
TXFIFO							
23	22	21	20	19	18	17	16
TXFIFO							
15	14	13	12	11	10	9	8
TXFIFO							
7	6	5	4	3	2	1	0
TXFIFO							

位	描述	
[31:0]	TXFIFO	**发送FIFO位** I²S有16个字(16*32位)数据缓存用于数据发送。写数据到该寄存器以准备发送。其剩余的数据字数可以从TXCNT(I2S_STATUS1[12:8])里读取

I²S接收FIFO寄存器(I²S_RXFIFO)

寄存器	偏移量	R/W	描述	复位值
I2S_RXFIFO	I2S_BA+0x14	R	I²S接收FIFO寄存器	0x0000_0000

31	30	29	28	27	26	25	24
RXFIFO							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
RXFIFO							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
RXFIFO							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
RXFIFO							

位	描述	
[31:0]	**RXFIFO**	**接收FIFO位** I²S有16个字(16*32位)数据缓存用于数据接收。从该寄存器读取FIFO里的数据。其剩余的数据字数可以从RXCNT(I2S_STATUS1[20:16])里读取

6.20 SPI接口(SPI)

6.20.1 概述

SPI接口是全双工同步串行数据通讯接口，可做为主机或从机，用4线双向通讯。M480包含3组SPI控制器。SPI口都支持I2S模式。每个都支持PDMA传输功能。

6.20.2 特征

- SPI模式
  - 3组SPI控制器
  - 支持主机模式和从机模式
  - 主/机模式时钟最高96MHz(VDD=2.7~3.6V)
  - 当主机器件支持RX数据采样时钟对齐功能，从机模式时钟最高可达到96MHz(VDD=2.7~3.6V)
  - 当主机器件不支持RX数据采样时钟对齐功能，从机模式时钟最高可达到96MHz(VDD=2.7~3.6V)
  - 传输位长可为8~32
  - 收发独立的4级FIFO缓存
  - 高低位在前可配置
  - 支持字节重排功能
  - 支持字节或字暂停模式
  - 支持 PDMA传输
  - 支持一数据通道半双工传输
  - 支持只接收模式
- I^2S模式(SPI0~SPI3)
  - 支持主机/从模式
  - 可处理8，16，24和32位数据宽度
  - 每组控制器提供2个4级FIFO缓存，一个用于发送，另一个用于接收
  - 支持单声道和双声道格式
  - 支持PCMA，PCMB，I²S和MSB对齐的数据格式
  - 支持2个PDMA请求，一个用于发送，另一个用于接收

6.20.3 框图

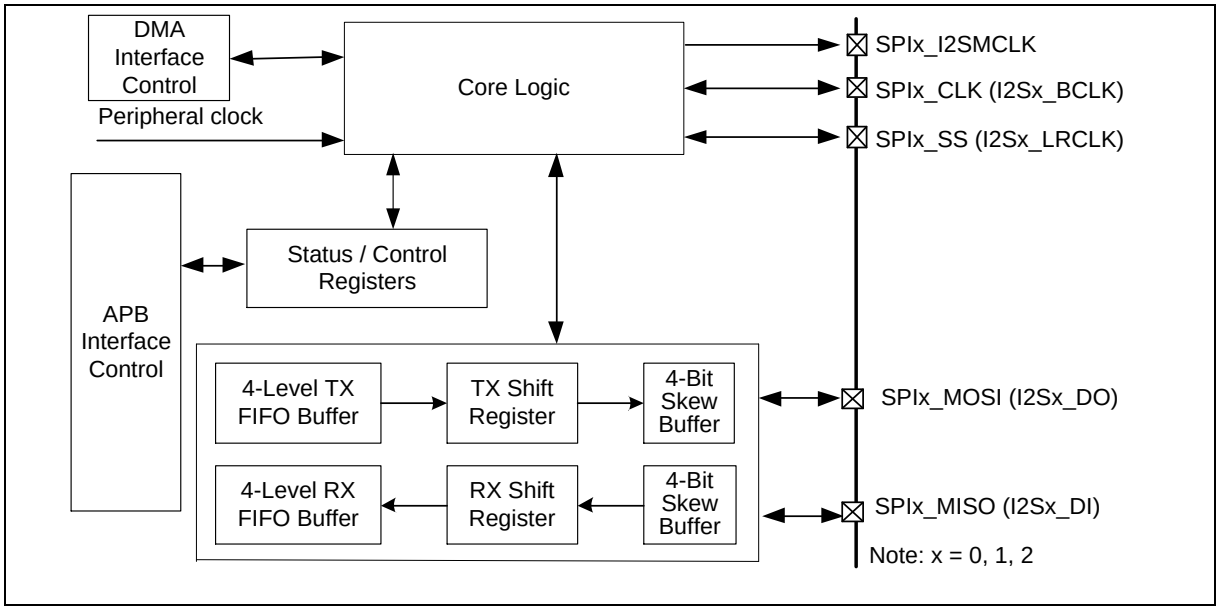


图 6.20-1 框架图

TXFIFO 缓存:

发送FIFO4级深度，32位宽。数据写入寄存器SPIx_TX就写入发送FIFO。SPI0~SPI3的SPI模式，如果数据长度为8~16位，则发送FIFO可以设为8级。

RX FIFO 缓存:

接收FIFO 4级深度，32位宽。接收控制会把数据存到该缓存。软件可通过读SPIx_RX得到该FIFO数据。SPI0~SPI3处于SPI模式时，如果数据长度为8~16位，发送FIFO可以设为8级。

TX 移位寄存器:

发送移位寄存器是一个32位的寄存器。发送数据从TX FIFO缓存加载到这里，并一位一位的移到斜移缓存。

RX 移位寄存器:

接收移位寄存器是一个32位的寄存器。接收数据从斜移缓存一位一位的移到这里，当一个事务完成时再载入到RX FIFO。

斜移缓存:

斜移缓存是一个4级1位的缓存。总共有2个斜移缓存分别在发送和接收端。在接收端，斜移缓存用于把数据从SPI总线移位到Rx移位寄存器。在发送端，斜移缓存用于把数据从Tx移位寄存器移位到SPI总线。

6.20.4 基本配置

6.20.4.1 SPI0基本配置

- 时钟配置
  - 寄存器SPI0SEL(CLK_CLKSEL2[5:4])配置SPI0的时钟
  - 寄存器SPI0CKEN(CLK_APBCLK0[13])使能SPI0的时钟

- 复位配置
  - 寄存器SPI0RST(SYS_IPRST1[13])复位SPI0控制器
- 管脚配置

组	管脚名	GPIO	MFP
SPI0	SPI0_CLK	PA.2, PB.14, PD.2	MFP4
		PF.8	MFP5
	SPI0_I2SMCLK	PA.4, PC.14, PD.13	MFP4
		PD.14, PF.10	MFP5
		PB.0	MFP8
		PB.11	MFP9
	SPI0_MISO	PA.1, PB.13, PD.1	MFP4
		PF.7	MFP5
	SPI0_MOSI	PA.0, PB.12, PD.0	MFP4
		PF.6	MFP5
	SPI0_SS	PA.3, PB.15, PD.3	MFP4
		PF.9	MFP5

6.20.4.2 SPI1基本配置

- 时钟配置
  - 寄存器SPI1SEL (CLK_CLKSEL2[7:6])配置 SPI1的时钟
  - 寄存器SPI1CKEN (CLK_APBCLK0[14])使能 SPI1的时钟
- 复位配置
  - 寄存器SPI1RST (SYS_IPRST1[14])复位 SPI1控制器
- 管脚配置

组	管脚名	GPIO	MFP
SPI1	SPI1_CLK	PH.6	MFP3
		PA.7	MFP4
		PB.3	MFP5
		PH.8	MFP6
		PC.1	MFP7
	SPI1_I2SMCLK	PA.5	MFP4
		PB.1, PD.13	MFP5
		PH.10	MFP6
		PC.4	MFP7

	SPI1_MISO	PH.4	MFP3
		PC.7	MFP4
		PB.5	MFP5
		PE.1	MFP6
		PC.3	MFP7
	SPI1_MOSI	PH.5	MFP3
		PC.6	MFP4
		PB.4	MFP5
		PE.0	MFP6
		PC.2	MFP7
	SPI1_SS	PH.7	MFP3
		PA.6	MFP4
		PB.2	MFP5
		PH.9	MFP6
		PC.0	MFP7

6.20.4.3 SPI2基本配置

- 时钟配置
  - 寄存器SPI2SEL (CLK_CLKSEL2[11:10])配置 SPI2的时钟
  - 寄存器SPI2CKEN (CLK_APBCLK0[15])使能 SPI2的时钟
- 复位配置
  - 寄存器SPI2RST (SYS_IPRST1[15])复位 SPI2控制器
- 管脚配置

组	管脚名	GPIO	MFP
SPI2	SPI2_CLK	PG.3	MFP3
		PA.10	MFP4
		PA.13, PE.8	MFP5
	SPI2_I2SMCLK	PB.0, PC.13	MFP4
		PE.12	MFP5
	SPI2_MISO	PG.4	MFP3
		PA.9	MFP4
		PA.14, PE.9	MFP5
	SPI2_MOSI	PF.11	MFP3
		PA.8	MFP4

		PA.15, PE.10	MFP5
	SPI2_SS	PG.2	MFP3
		PA.11	MFP4
		PA.12, PE.11	MFP5

SPI/I^{2}S (SPI0~SPI2) 接口管脚描述如下:

管脚	SPI 模式	I^{2}S 模式
SPIx_SS	SPI 从机片选	I^{2}S 左右通道时钟(I2Sx_LRCLK)
SPIx_CLK	SPI 时钟	I^{2}S 位时钟 (I2Sx_BCLK)
SPIx_MISO	SPI 主机输入或从机输出	I^{2}S 数据输入 (I2Sx_DI)
SPIx_MOSI	SPI 主机输出或从机输入	I^{2}S 数据输出 (I2Sx_DO)
SPIx_I2SMCLK	不可用	I^{2}S 主时钟输出

表 6.20-1 SPI/I^{2}S 接口管脚描述(SPI0~SPI2)

6.20.5 功能描述

6.20.5.1 术语

SPI 外设时钟和 SPI 总线时钟

SPI 时钟决定于时钟分频器 (SPIx_CLKDIV) 和时钟源的设置，时钟源可以设置为 HXT, HIRC, PLL 时钟或 PCLK.。寄存器 CLK_CLKSEL2 的 SPIxSEL 位选择 SPI 的时钟源。寄存器 DIVIDER (SPI_CLKDIV[8:0]) 的值决定时钟的分频值。

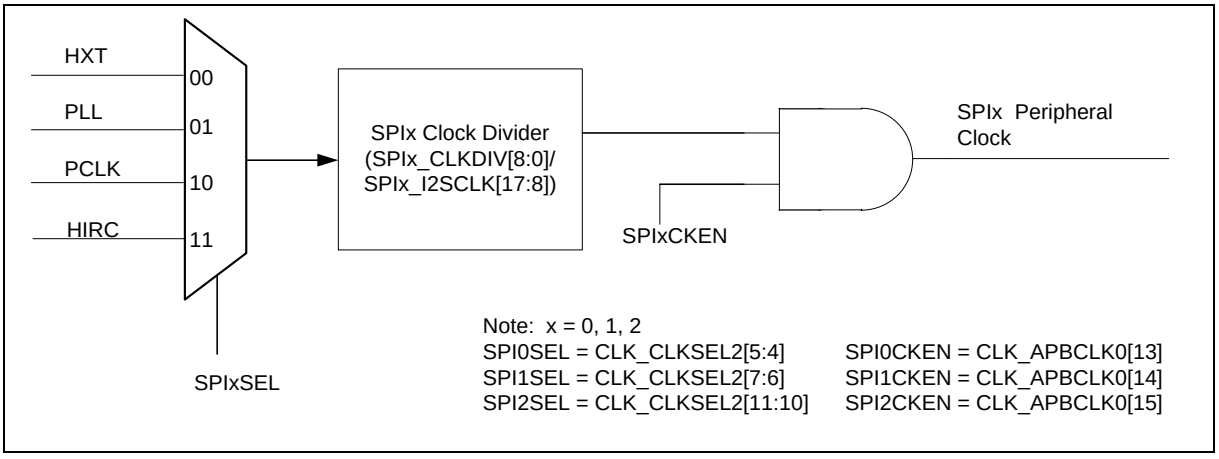


图 6.20-2 SPI 外设时钟

SPI主机模式下，SPI 总线的时钟频率等于外设的时钟速率。从机模式下，SPI 总线时钟由主机设备提供。无论主机模式还是从机模式，SPI 时钟不能快于系统时钟。如果时钟源不是系统时钟，SPI时钟不管是主机模式还是从机模式都必须慢于系统时钟频率。

I2S 模式下，外设时钟速率等于 I2S 的位时钟速率，I2S 的位时钟速率由寄存器 SPIx_I2SCLK配置。

主从模式

可以通过寄存器 SLAVE (SPIx_CTL[18]) 设成主机模式或从机模式。SPI 传送过程中，通过 HALFDPX (SPIx_CTL[14]) 可以切换全双工模式或半双工模式。示意图如下：

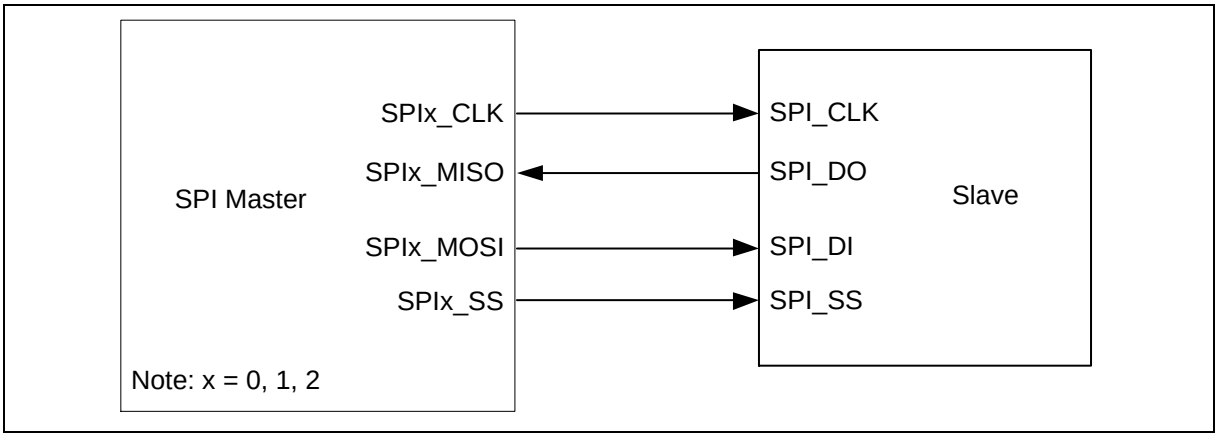


图 6.20-3 SPI 全双工主机模式

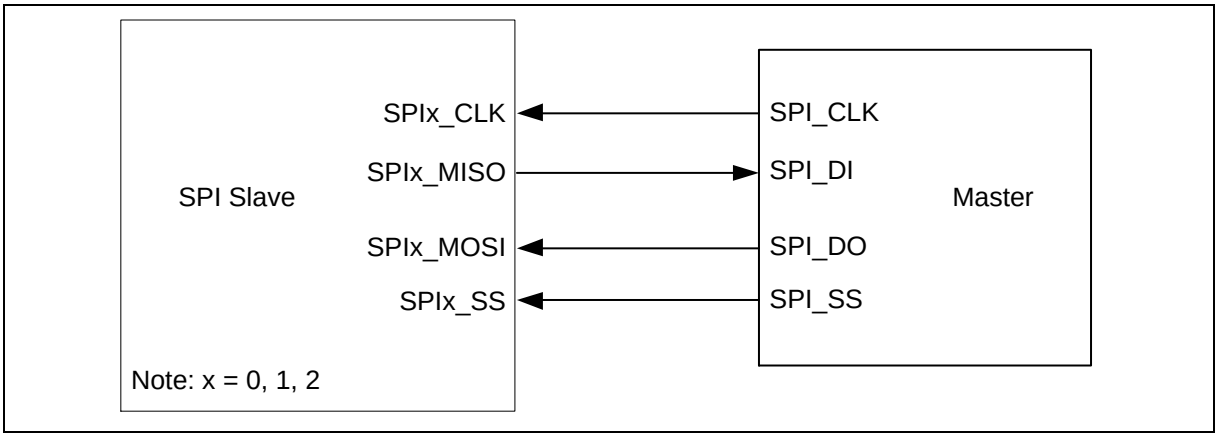


图 6.20-4 SPI 全双工从机模式

从机选择

主机模式下，SPI 通过从机片选SPIx_SS（输出）使能从机设备。从机模式，其它主机通过 SPIx_SS（输入）使能 SPI 控制器。片选有效边沿到第一个 SPI 时钟输入应多于 3 个 SPI 外设时钟周期。

从机信号可通过寄存器 SSACTPOL (SPIx_SSCTL[2])设定为低有效或高有效。从机模式下，从机片选信号的无效时间必须大于等于 3 个外设时钟周期。

时钟极性

CLKPOL (SPIx_CTL[3]) 定义SPI 时钟空闲时的电平。如果 CLKPOL = 1，SPI 时钟空闲时输出高电平，如果 CLKPOL = 0，则 SPI 时钟空闲时输出低电平。

TXNEG (SPIx_CTL[2]) 定义数据是在 SPI 时钟的下沿发送还是上沿发送。RXNEG (SPIx_CTL[1]) 定义数据在 SPI 时钟的下沿接收还是上沿接收。

注意: TXNEG 和 RXNEG 的设置是互斥的，收发不能都配置为上沿或都是下沿。

收发位长

位长由 DWIDTH (SPIx_CTL[12:8]) 来配置，最多可为 32 位。

当 SPI 完成一次 DWIDTH (SPI_CTL[12:8])定义的位长的收/发时，传输中断标志(SPIx_STATUS[1])将被置1。

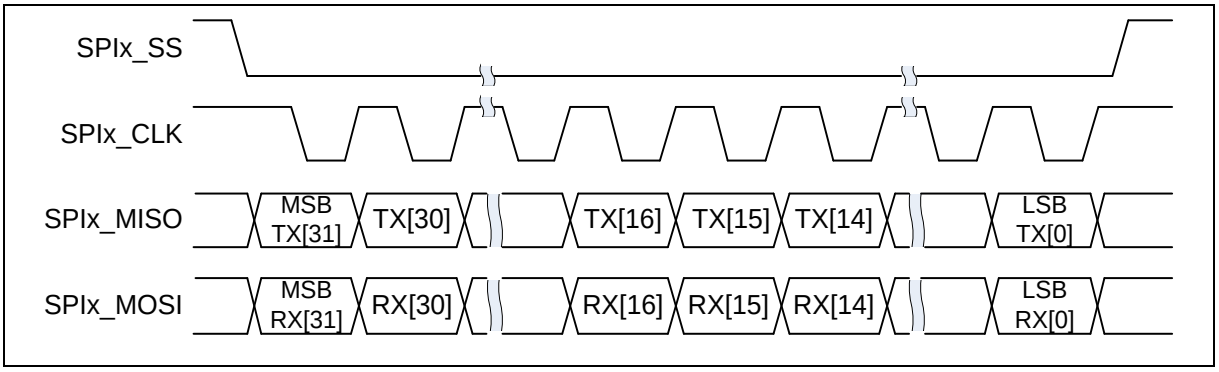


图 6.20-532 位的收发

高低位在前

LSB (SPIx_CTL[13]) 定义传输顺序。LSB (SPIx_CTL[13])=1 时，第 0 位会先传输。若 (SPIx_CTL[13]) = 0，则先传高位。

休眠间隔

主机模式下，SUSPITV (SPIx_CTL[7:4]) 可配置在两次连续传输之间，插入 0.5~15.5 个 SPI 时钟周期的空闲：空闲时间的计算是从前一次传输的最后一个时钟沿，到下一次传输的第一个时钟沿。SUSPITV 的默认值是 0x3 (3.5 个 SPI 时钟周期)。

6.20.5.2 自动从机选择

主机模式下，如果 AUTOSS (SPIx_SSCTL[3]) 置位，从机选择信号根据 SS (SPIx_SSCTL[0]) 自动输出到 SPIx_SS 管脚上。当数据写入 FIFO数据传输启动，从机选择信号自动设置为有效状态。当 SPI 总线空闲时，从机选择信号将自动变为无效状态。SPI 总线不空闲时——比如TX FIFO，TX 移位寄存器或 TX 斜移缓存不空——在 SUSPITV (SPIx_CTL[7:4]) 的值大于或等于 3 的条件下，从机选择信号在两次传输之间将被置为无效状态。

主机模式下，如果 SUSPITV 的值小于 3 且 AUTOSS 置 1时，两次连续传输之间，从机选择信号将保持有效。

如果 AUTOSS 被清零，从机选择输出信号的有效电平由 SSACTPOL (SPIx_SSCTL[2]) 来定义。

从机片选信号有效边沿，到第一个SPI时钟边沿的间隔为 1 个 SPI 总线时钟周期。SPI 总线的最后一个时钟，到从机选择信号无效边沿的间隔为 1.5 个 SPI 总线时钟周期。

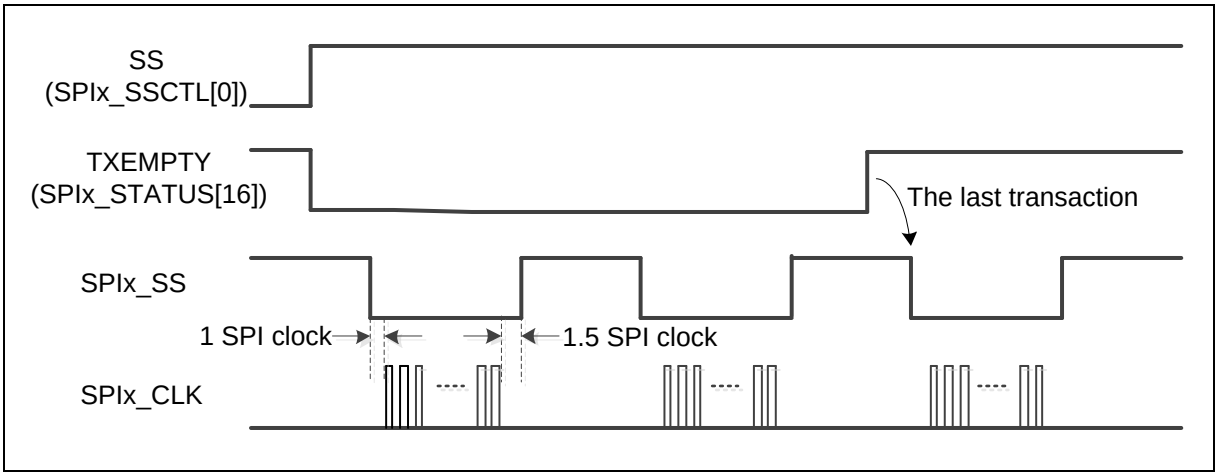


图 6.20-6 自动从机选择(SSACTPOL = 0, SUSPITV > 0x2)

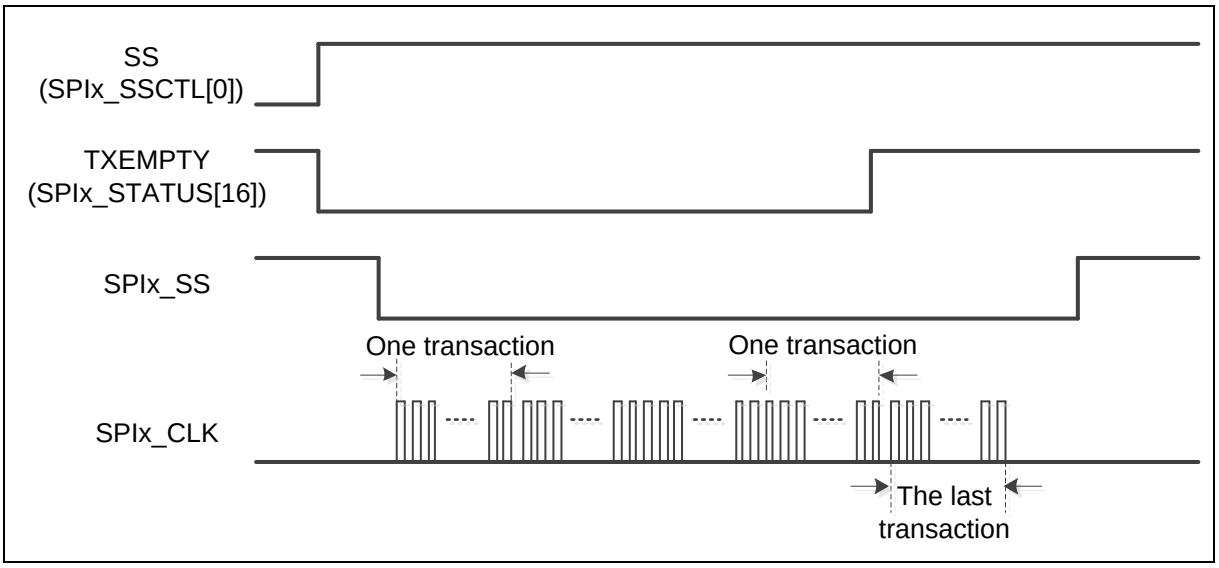


图 6.20-7 自动从机选择(SSACTPOL = 0, SUSPITV < 0x3)

6.20.5.3 字节重排和挂起功能

当传输设置为 MSB 优先 (LSB = 0) 且使能了 REORDER (SPIx_CTL[19])，在 32 位传输模式 (DWIDTH = 0) 下，存储在 TX 缓存与 RX 缓存的数据将按 [BYTE0, BYTE1, BYTE2, BYTE3] 的顺序重新排列。数据发送/接收的顺序为 BYTE0, BYTE1, BYTE2, 和 BYTE3。如果 DWIDTH 设为 24 位传输模式，存储在 TX 缓存与 RX 缓存的数据将按 [未知字节, BYTE0, BYTE1, BYTE2] 的顺序重新排列。SPI 控制器将按照 BYTE0, BYTE1, BYTE2 的顺序发送/接收数据，每个字节 MSB 优先发送/接收。16 位传输模式的规则与上面相同。字节重排序功能只适用于 DWIDTH 为 16, 24, 和 32 位时。

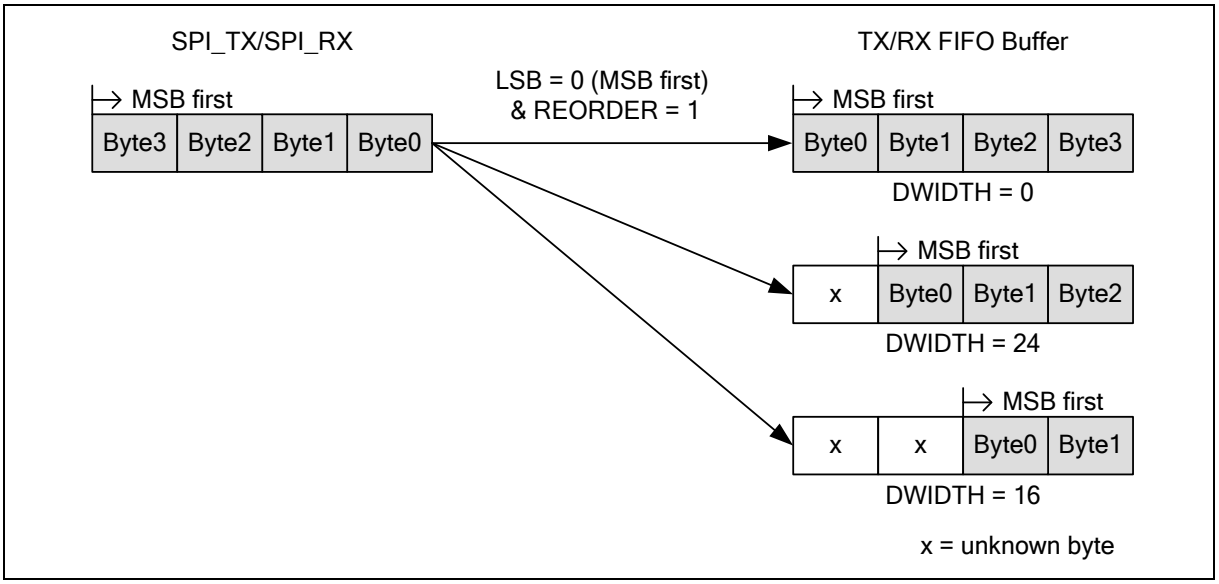


图 6.20-8 字节重排功能

主机模式下，如果 REORDER (SPIx_CTL[19]) 为 1，两次连续传输字节之间将插入 0.5 ~ 15.5 个 SPI 时钟周期的挂起间隔。挂起间隔时间由 SUSPITV (SPIx_CTL[7:4]) 设置。

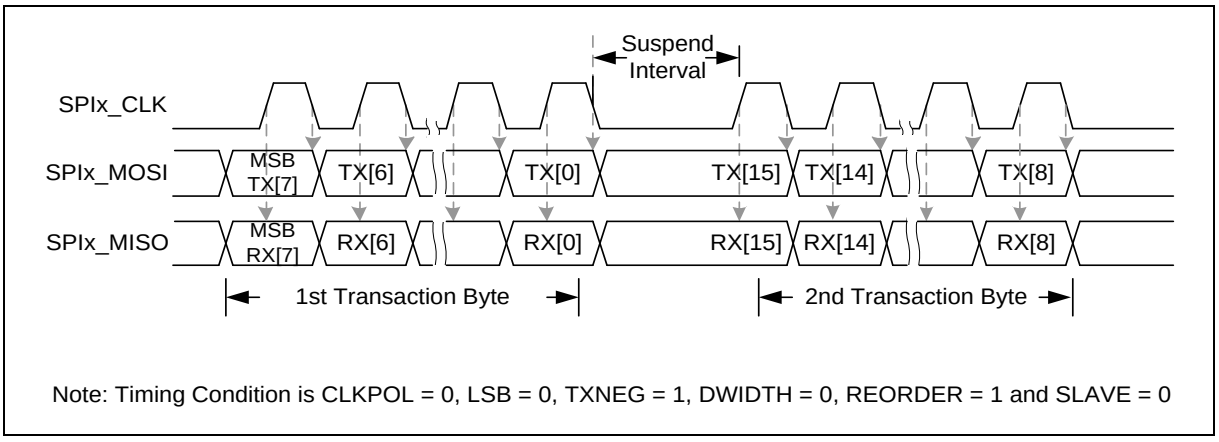


图 6.20-9 字节挂起波形图

6.20.5.4 半双工通讯

设置 HALFDPX (SPIx_CTL[14]) 位可以使能 SPI 工作在半双工模式。半双工模式下，仅有一条数据线进行发送或接收，方向由DATDIR (SPIx_CTL[20]) 配置。半双工模式下，没有用到的 SPIx_MISO 管脚可以配置成 GPIO 供其他功能使用。使能或关闭 HALFDPX (SPIx_CTL[14]) 控制位将会同时产生 TXFBCLR (SPIx_FIFOCTL[9]) 和 RXFBCLR (SPIx_FIFOCTL[8])。

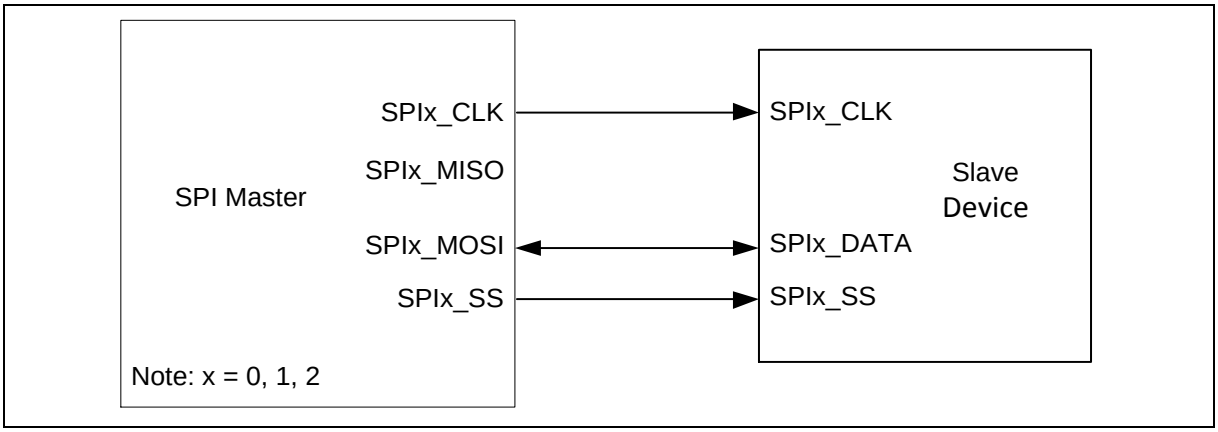


图 6.20-10 SPI 半双工主机模式

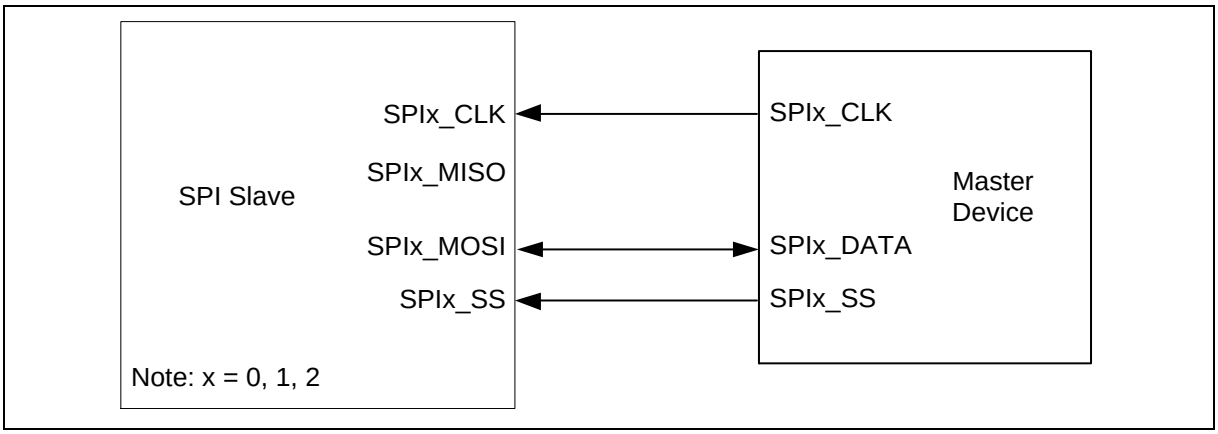


图 6.20-11 SPI 半双工从机模式

6.20.5.5 只接收模式

对于 SPI 主机，可配置 RXONLY (SPIx_CTL[15]) 让其工作在只接收模式，SPI 主机一直不断地产生 SPI 总线时钟。该模式下，如果使能了 AUTOSS (SPIx_SSCTL[3]) ，SPI 主机将一直保持从机片选信号有效。

在只接收模式下，没有用到的 SPIx_MOSI 管脚可以配置成 GPIO 供其他功能使用。由于 SPI 总线时钟一直存在，BUSY (SPIx_STATUS[0]) 将会被置位。进入该模式会同时产生 TXFBCLR (SPIx_FIFOCTL[9]) 和 RXFBCLR (SPIx_FIFOCTL[8])。使能该模式后，SPI 总线时钟会在 6 个外设时钟周期内发送出去。该模式下，写入发送 FIFO 的数据也会被载入发送移位寄存器发送出去。

当设置RXONLY (SPIx_CTL[15]) 使能，，按照数据位宽度DWIDTH (SPIx_CTL[12:8])设置的SPI RX数据将被接收到RX FIFO中，SPI时钟将被发送到SPI从设备中，直到RX FIFO满为止。

当数据位宽度设置为 8~16位，SPI主机将发送SPI 输出时钟到SPI从机。SPI从机接收数据，当RX FIFO计数器(SPIx_STATUS[27:24]) 小于或等于6。

当数据位宽度设置为17~32位，SPI主机将发送SPI 输出时钟到SPI从机。SPI从机接收数据，当RX FIFO计数器(SPIx_STATUS[27:24]) 小于或等于6。

6.20.5.6 PDMA 传输功能

SPI 控制器支持 PDMA 传输功能.

当 TXPDMAEN (SPIx_PDMACTL[0]) 置 1 时，控制器会请求 PDMA控制发送过程。

当 RXPDMAEN (SPIx_PDMACTL[1]) 置 1 时，将启动 PDMA 接收过程。当接收 FIFO 有数据时，控制器会请求 PDMA 接收数据。

注: SPI 只支持单一 PDMA 请求 (读/写), 不支持 PDMA 突发请求.

6.20.5.7 FIFO缓存操作

SPI 控制器配备了 4 个 32 位宽的发送和接收 FIFO 缓存。存放在发送 FIFO 缓存的数据会通过发送控制逻辑进行读取和发送。如果发送 FIFO 缓存满了，TXFULL (SPIx_STATUS[17]) 会被置 1。当 SPI 传输逻辑单元抽出发送 FIFO 缓存的最后一个数据，发送 FIFO 缓存就为空了，TXEMPTY (SPIx_STATUS[16]) 位被置 1。注意 TXEMPTY (SPIx_STATUS[16]) 标志在最后一笔数据传输还在进行时就被置 1 了。在主机模式，当 FIFO 缓存写入数据或者 SPI 总线上有任何事务， BUSY (SPIx_STATUS[0]) 位被设置为 1。(如：从机片选信号激活和 SPI 控制器在从机模式正在接收数据)。当传送缓存为空且当前事务已经完成后，该位将设置为 0。因此，软件可以检查 BUSY (SPIx_STATUS[0]) 位的状态以确认 SPI 是否已经空闲。

接收控制逻辑存储 SPI 接收到的数据到接收 FIFO 缓存。有 FIFO 相关的状态位，像 RXEMPTY (SPIx_STATUS[8]) 和 RXFULL (SPIx_STATUS[9])，来表明当前 FIFO 缓存的状态。

发送和接收的阀值可以通过设置 TXTH (SPIx_FIFOCTL[30:28]) 和 RXTH (SPIx_FIFOCTL[26:24]) 来设定。当存储在发送 FIFO 缓存的有效数据数小于或等于 TXTH (SPIx_FIFOCTL[30:28]) 的设定时，TXTHIF (SPIx_STATUS[18]) 位会被置 1。当存储在接收 FIFO 缓存的有效数据数大于 RXTH (SPIx_FIFOCTL[26:24]) 的设定，RXTHIF (SPIx_STATUS[10]) 位会被置 1。

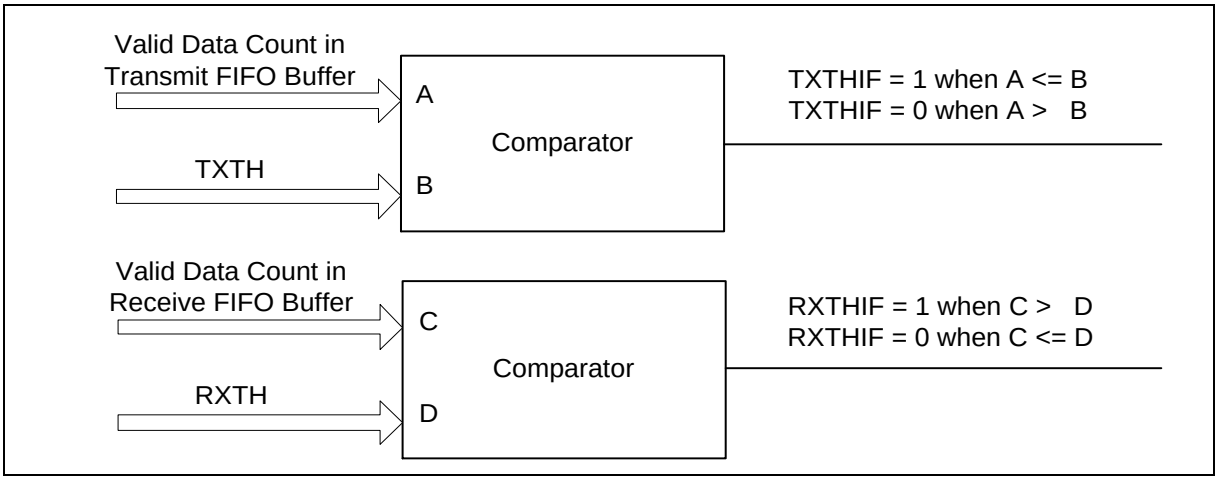


图 6.20-12 FIFO 阈值比较器

在主机模式，当第一个数据写入 SPIx_TX 寄存器时，TXEMPTY (SPIx_STATUS[16]) 标志将会被清除为 0。在 1 个 APB 时钟周期和 6 个外设时钟周期后，发送将开始。用户可以立即写下一个数据到 SPIx_TX 寄存器。SPI 控制器将会在两个连续的事务之间插入一个休眠间隔，休眠间隔的长度由 SUSPITV (SPIx_CTL[7:4]) 的设定值决定。如果 SUSPITV (SPIx_CTL[7:4]) 等于 0，SPI 控制器可以执行连续发送。只要 TXFULL (SPIx_STATUS[17]) 为 0，用户就可以向 SPIx_TX 寄存器写入新的数据。

如图 6.20-13的例 1 所示，该图指明了 TXEMPTY (SPIx_STATUS[16]) 的更新条件以及 FIFO 缓存，8~16 位移位寄存器和斜移缓存之间的关系。当 Data 0 写入 FIFO 缓存时，TXEMPTY (SPIx_STATUS[16]) 位被设置为 0。Data 0 将由内核逻辑加载到移位寄存器，此时，TXEMPTY (SPIx_STATUS[16]) 将设置为 1。移位寄存器中 Data 0 将逐位移入斜移缓存传输直到传输完成。

如图 6.20-13的例2所示，该图为当 FIFO 缓存中有 8 个数据时，更新 TXFULL (SPIx_STATUS[17]) 的

条件。当 TXFULL = 1 时，下一数据 Data 9 不能被写入 FIFO 缓存。

如图 6.20-14的例 1 所示，该图指明了 TXEMPTY (SPIx_STATUS[16]) 的更新条件以及 FIFO 缓存，17~32位移位寄存器和斜移缓存之间的关系。

如图 6.20-14的例2所示，该图为当 FIFO 缓存中有4个数据时，更新 TXFULL (SPIx_STATUS[17]) 的条件。当 TXFULL = 1 时，下一数据 Data 9 不能被写入 FIFO 缓存。

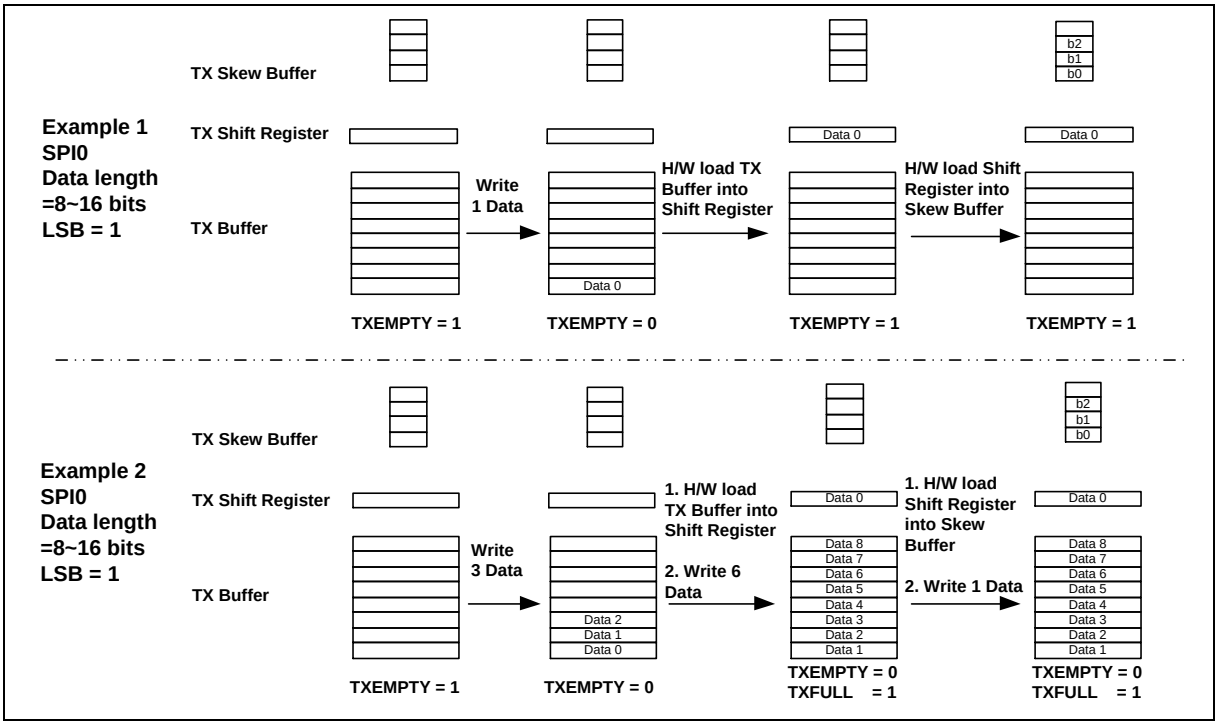


图 6.20-13 8-16 位宽发送 FIFO 缓存示例

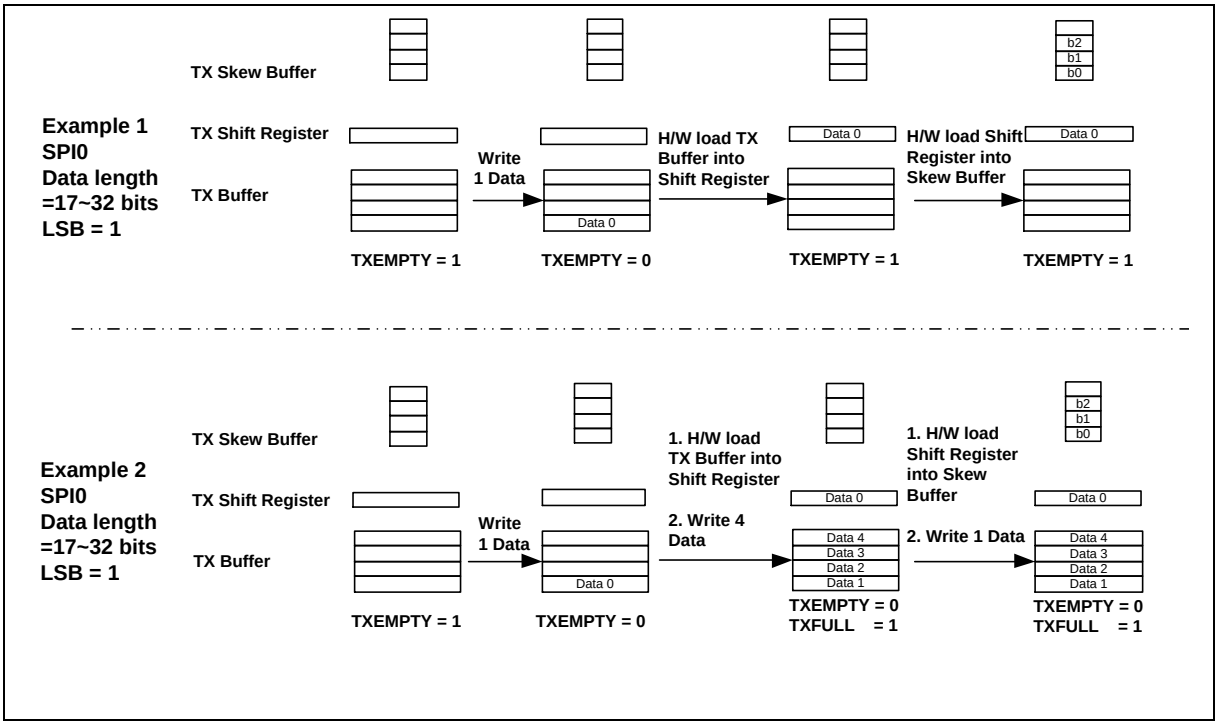


图 6.20-1417-32 位宽发送 FIFO 缓存示例

如果要发送的数据更新及时，接下来的事务将会被自动触发。如果在所有数据传输完成之后， SPIx_TX 寄存器没有被更新，则传输停止。

在主机模式接收操作中，串行数据从 SPIx_MISO 管脚接收并被存储在接收 FIFO 缓存。

接收数据 (Data 0's b0, b1, …b31) 通过串行时钟 (SPIx_CLK) 先存到斜移缓存，然后再逐位移到移位寄存器。当接收数据位达到 DWIDTH (SPIx_CTL[12:8]) 的值时，内部逻辑将移位寄存器中的数据加载到 FIFO 缓存。

当接收 FIFO 缓存有未读数据时 (参见图6.20-15示例 1)，RXEMPTY (SPIx_STATUS[8]) 将清 0。只要 RXEMPTY (SPIx_STATUS[8]) 为 0，用户就可以通过 SPIx_RX 寄存器来读取接收的数据。如果接收 FIFO 缓存包含 8 个未读数据，RXFULL (SPIx_STATUS[9]) 将设置为 1 (参见图6.20-15示例 2)。

当接收 FIFO 缓存有未读数据时 (参见图6.20-16示例 1)，RXEMPTY (SPIx_STATUS[8]) 将清 0。只要 RXEMPTY (SPIx_STATUS[8]) 为 0，用户就可以通过 SPIx_RX 寄存器来读取接收的数据。如果接收 FIFO 缓存包含4个未读数据，RXFULL (SPIx_STATUS[9]) 将设置为 1 (参见图6.20-16示例 2)。

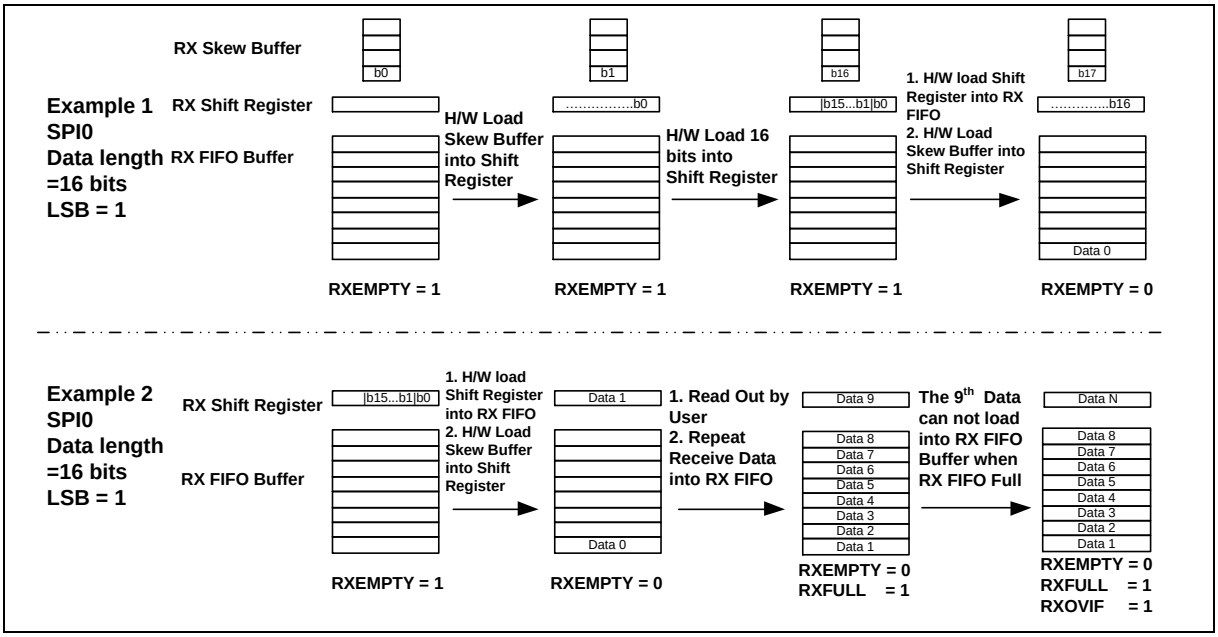


图 6.20-15 接收 16 位宽 FIFO 缓存示例

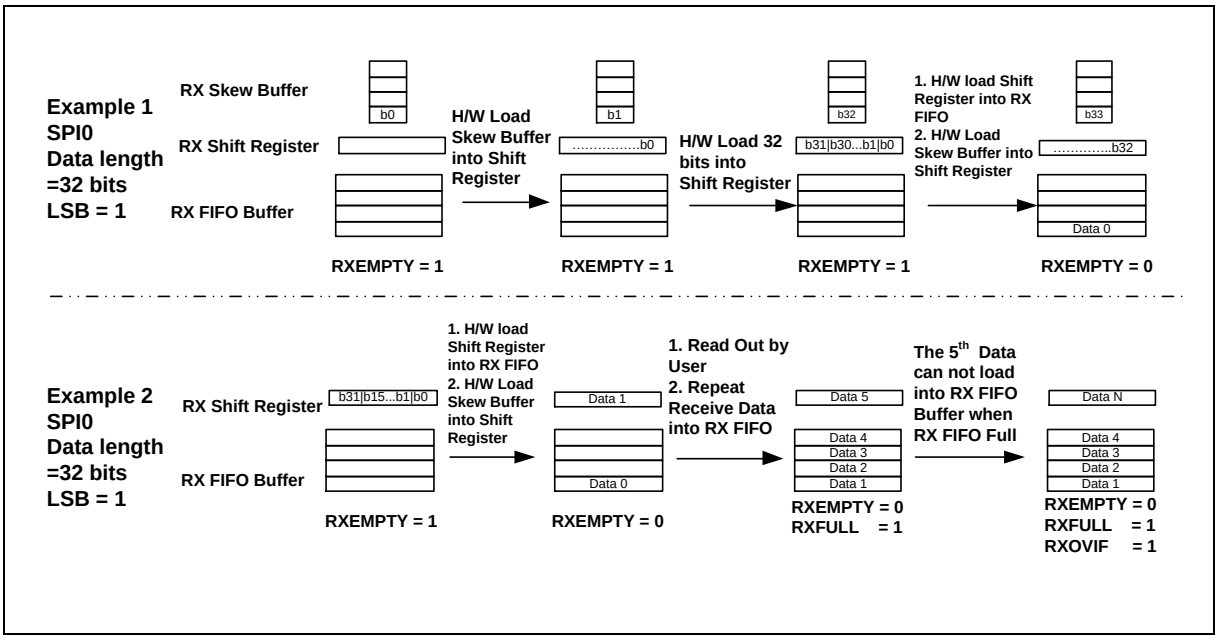


图 6.20-16 接收 32 位宽 FIFO 缓存示例

在从机模式中，当软件写数据到 SPIx_TX 寄存器时，数据将会被加载到发送 FIFO 缓存，同时 TXEMPTY (SPIx_STATUS[16]) 标志也将被设置为 0。当从设备从主机接收到时钟信号时，发送操作将会开始。只要 TXFULL (SPIx_STATUS[17]) 标志为 0，用户就可以写数据到 SPIx_TX 寄存器。所有数据都被 SPI 发送逻辑单元发送出去后，而且软件没有再更新 SPIx_TX 寄存器，TXEMPTY (SPIx_STATUS[16]) 标志将会被设置为 1。

当从机片选信号有效时，如果没有任何数据写入 SPIx_TX 寄存器，发送下溢标志 TXUFIF (SPIx_STATUS[19]) 将被设置为 1。输出的数据在本次传输中将会通过设置 TXUFPOL

(SPIx_FIFOCTL[6]) 保留，直到从机片选信号为无效状态。当传送下溢事件发生时，从机溢出运行标志 SLVURIF (SPIx_STATUS[7]) 将置为 1。同时 SPIx_SS 进入无效状态。

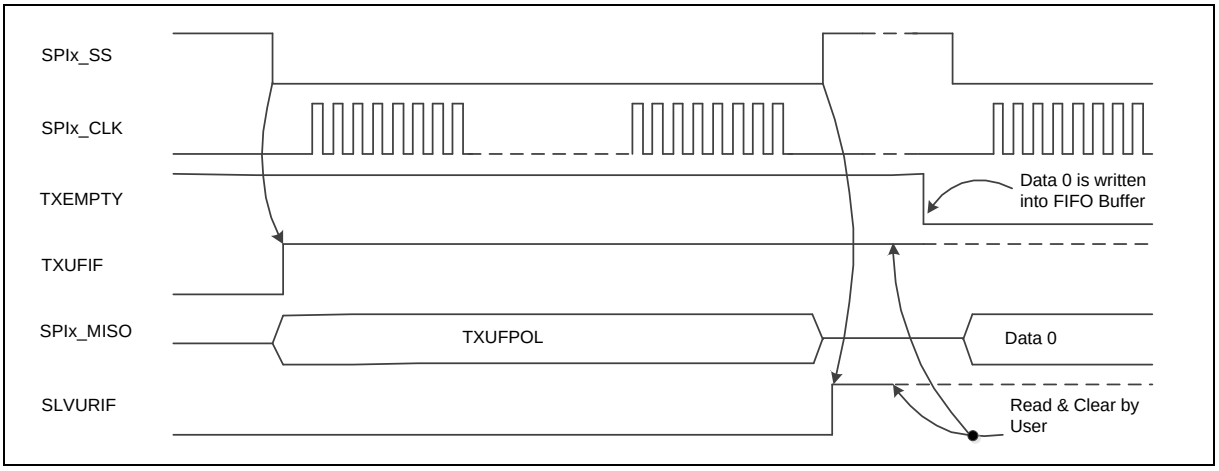


图 6.20-17 TX 下溢事件和从机溢出事件

在从机模式接收操作中，串行数据从 SPIx_MOSI 管脚接收并存储到 SPIx_RX 寄存器。接收机制类似于主机模式的接收操作。如果接收 FIFO 缓存包含 4 个未读数据，RXFULL (SPIx_STATUS[9]) 将被置为 1，RXOVIF (SPIx_STATUS[11]) 也会被置为 1。如果在 SPIx_MOSI 管脚上有更多串行数据要接收，接下来的数据将被丢掉 (参看接收 FIFO 缓存的示例图)。当从机片选线进入无效状态时，如果接收到位数数据与 DWIDTH (SPIx_CTL[12:8]) 设定的不一致，SLVBEIF (SPIx_STATUS[6]) 将设置为 1。

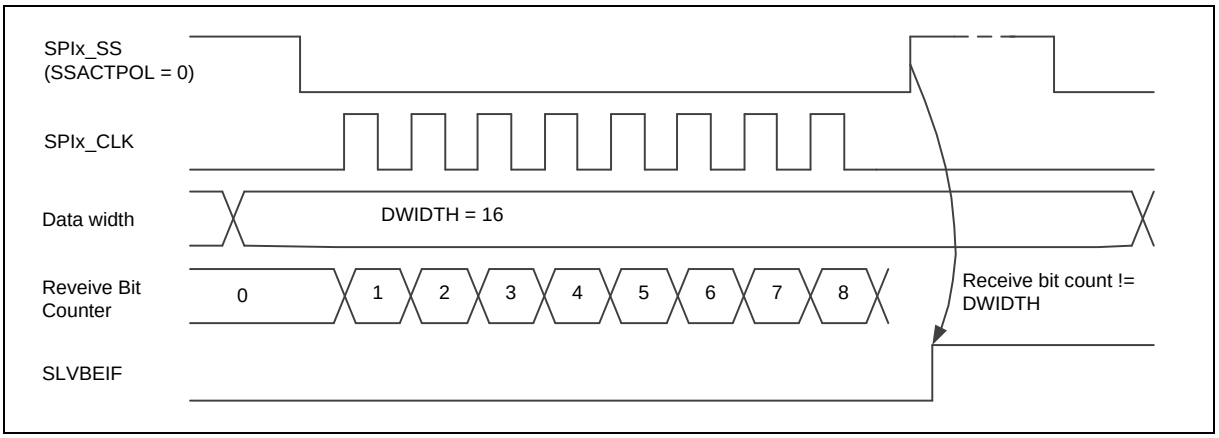


图 6.20-18 从机模式位计数错误

控制器内置了一个接收超时功能模块。当接收 FIFO 非空，而且在主机模式时接收 FIFO 超过 64 个 SPI 外设时钟周期或者从机模式时超过 576 个 SPI 外设时钟周期没有读操作，则接收超时发生，RXTOIF (SPI_STATUS[12]) 置 1。当接收 FIFO 有用户读取时，超时状态将自动清除。

6.20.5.8 中断

- SPI单元传输中断

  当 SPI 控制器完成一个单元传输，单元传输中断标志 UNITIF (SPIx_STATUS[1]) 将会被设为 1。如果单元传输中断使能位 UNITIEN (SPI_CTL[17]) 被置位，则单元传输中断事件将给 CPU 产生中断请求。单位传输中断标志位只能写 1 清零。.

- SPI从机片选有效/无效中断

  从机模式下，如果SPIEN (SPIx_CTL[0]) 和 SLAVE (SPIx_CTL[18]) 置 1，从机片选信号进入有效/无效状态，从机片选有效/无效中断标志 SSACTIF (SPIx_STATUS[2]) 和 SSINAIF (SPIx_STATUS[3]) 会被置位。此时如果 SSINAIEN (SPIx_SSCTL[13]) 或 SSACTIEN (SPIx_SSCTL[12]) 为 1，SPI 控制器会产生中断。

- 从机位计数错误中断

  在从机模式，当从机片选信号线进入无效状态时，如果发送或接收到位数据个数与 DWIDTH (SPIx_CTL[12:8]) 设置的不一致，SLVBEIF (SPIx_STATUS[6]) 将被设置为 1。未传输完成的数据将在 TX 和 RX 移位寄存器中弃除。如果 SLVBEIEN (SPIx_SSCTL[8]) 位为 1，SPI 控制器将发生一个中断。

  注:如果从机片选信号激活，但是没有任何串行时钟输入，当从机片选信号进入无效状态时，SLVBEIF (SPIx_STATUS[6]) 也将设置为 1。

- TX 下溢中断

  在 SPI 从机模式，如果没有任何数据写入 SPIx_TX 寄存器，当从机选择信号激活时，TXUFIF (SPIx_STATUS[19]) 将设置为 1。如果 TXUFIEN (SPIx_FIFOCTL[7]) 为 1，SPI 控制器将发生发送下溢中断。

  注: SPI 从机模式发生下溢事件后，有两种方法可以使其恢复空闲状态并进入下个传输：(1) 置 TXRST 为 1 (2) 从机片选信号切为无效状态。

- 从机 TX 下溢运行中断

  当 SPIx_SS 进入无效状态时，如果有 TX 下溢事件发生，SLVURIF (SPIx_STATUS[7]) 将被设置为 1。如果 SLVURIEN (SPIx_SSCTL[9]) 为 1，SPI 控制器将发生发送溢出运行中断。

  注意: 在从机三线模式，从机片选信号被认为是一直有效的，用户需要去查看 TXUFIF (SPI_STATUS[19]) 位来确定是否有发生 TX 下溢事件。

- 接收溢出中断

  在从机模式，如果接收 FIFO 缓存已有 4 个未读数据， RXFULL (SPIx_STATUS[9]) 和 RXOVIF (SPIx_STATUS[11]) 标志将会被设置为 1。如果从 SPI 总线上接收到更多串行数据，多余的数据将会丢失。如果 RXOVIEN (SPIx_FIFOCTL[5]) 为 1，SPI 控制器将发生接收溢出中断。

- 接收 FIFO 超时中断

  如果在 FIFO 里有一个接收到的数据，在主机模式下用户超过 64 个 SPI 外设时钟周期没有去读取，或者从机模式下超过 576 个 SPI 外设时钟周期没有去读取，如果接收超时中断使能位 RXTOIEN (SPIx_FIFOCTL[4]) 有设置为 1，则会向系统发出一个 RX 超时中断。

- 发送 FIFO 中断

  在 FIFO 模式，如果发送 FIFO 缓存的有效数据少于或等于 TXTH (SPIx_FIFOCTL[30:28]) 的设定值，发送 FIFO 中断标志 TXTHIF (SPIx_STATUS[18]) 会被置 1。如果发送 FIFO 中断位 TXTHIEN (SPIx_FIFOCTL[3]) 为 1，则 SPI 控制器会向系统产生一个发送 FIFO 中断。

  .

- 接收 FIFO 中断

  在 FIFO 模式，如果接收 FIFO 缓存的有效数据大于 RXTH (SPIx_FIFOCTL[26:24]) 的设定值，接收 FIFO 中断标志 RXTHIF (SPIx_STATUS[10]) 会被设置为 1。如果接收 FIFO 中断位

RXTHIEN (SPIx_FIFOCTL[2]) 有设置为 1，SPI 控制器将会向系统产生一个接收 FIFO 中断。.

6.20.5.9 I²S模式

SPI0~SPI2控制器支持 I²S 模式，支持 PCM 模式 A，PCM 模式 B 和 MSB 对齐以及 I²S 数据格式。音频通道的位宽由 WDWIDTH (SPIx_I2SCTL[5:4]) 设定。传输顺序一般为 MSB (最高位优先)。数据在时钟线的上升沿时读取下降沿时传送。

在 I²S 数据格式中，最高位优先传送，在音频通道的第二个时钟发送。I2S_LRCLK 信号指示正在传输的音频通道。.

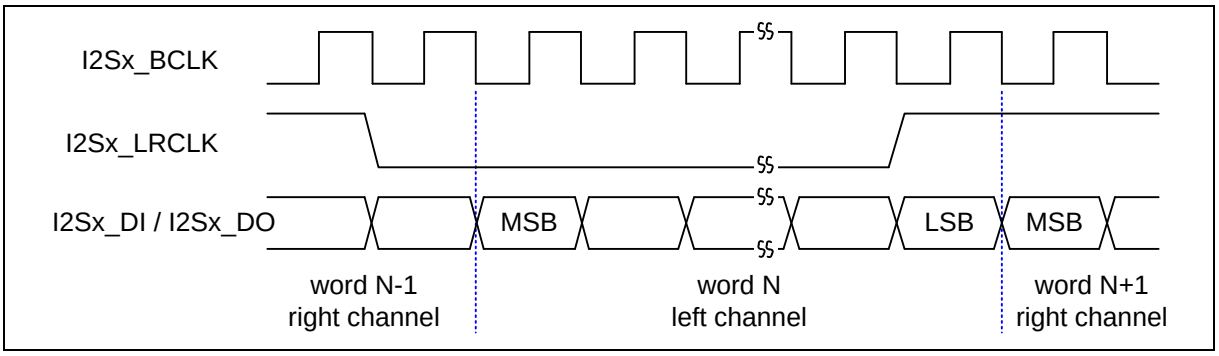


图 6.20-19I²S 数据格式时序

在 MSB 对齐数据格式，最高位优先传送，在音频通道的第一个时钟发送。

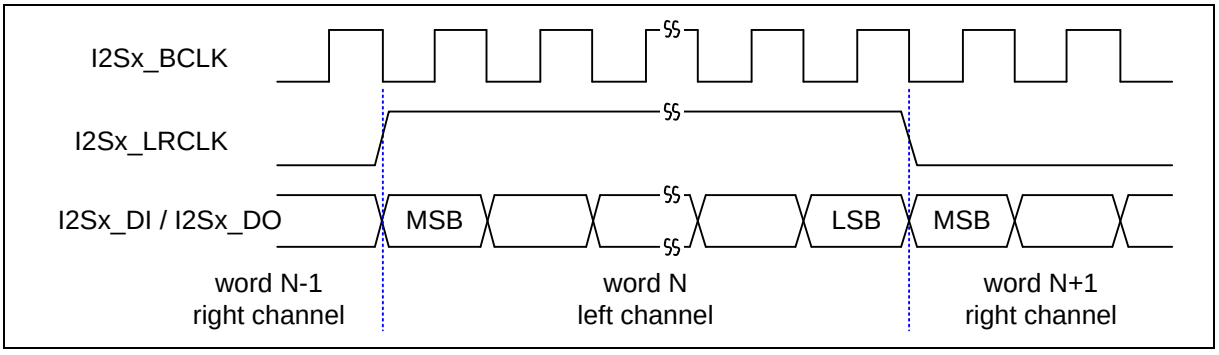


图 6.20-20 对齐数据格式时序图

I2Sx_LRCLK 也支持 PCM 模式 A 和 PCM 模式 B。PCM 模式下 I2Sx_LRCLK 用来指示音频帧的开始。

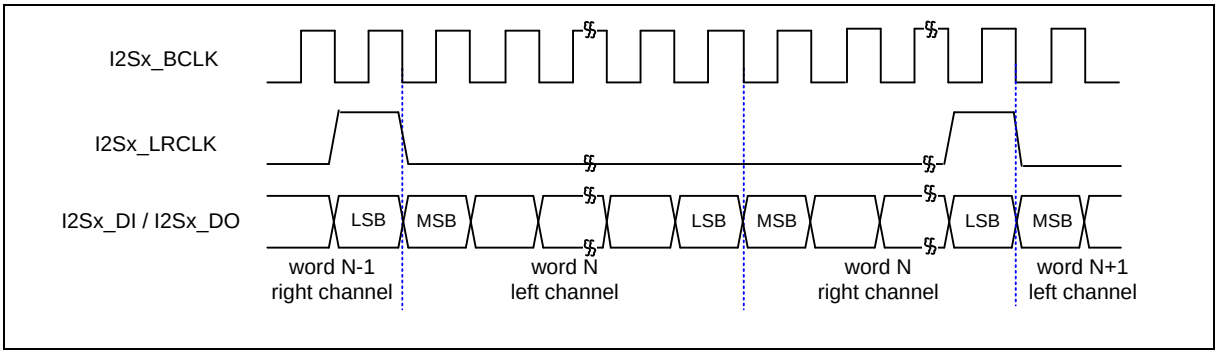


图 6.20-21PCM 模式 A 时序图

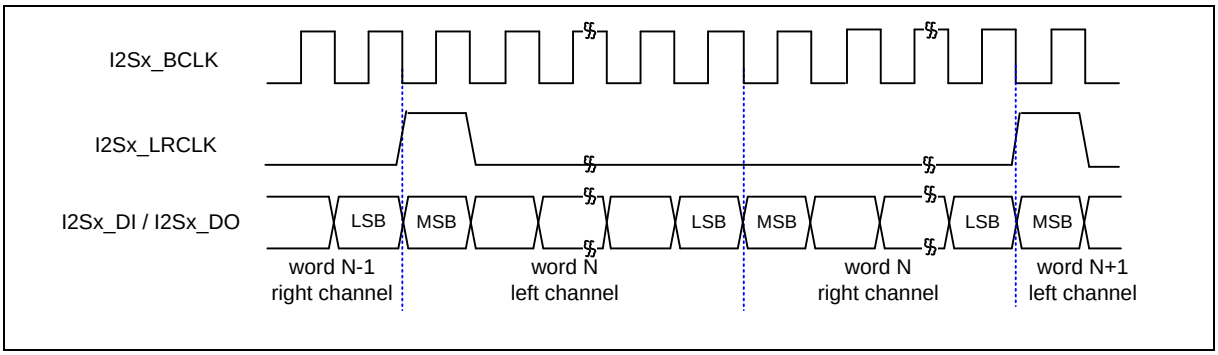


图 6.20-22PCM 模式 B 时序图

6.20.5.10 I²S 模式 FIFO 操作

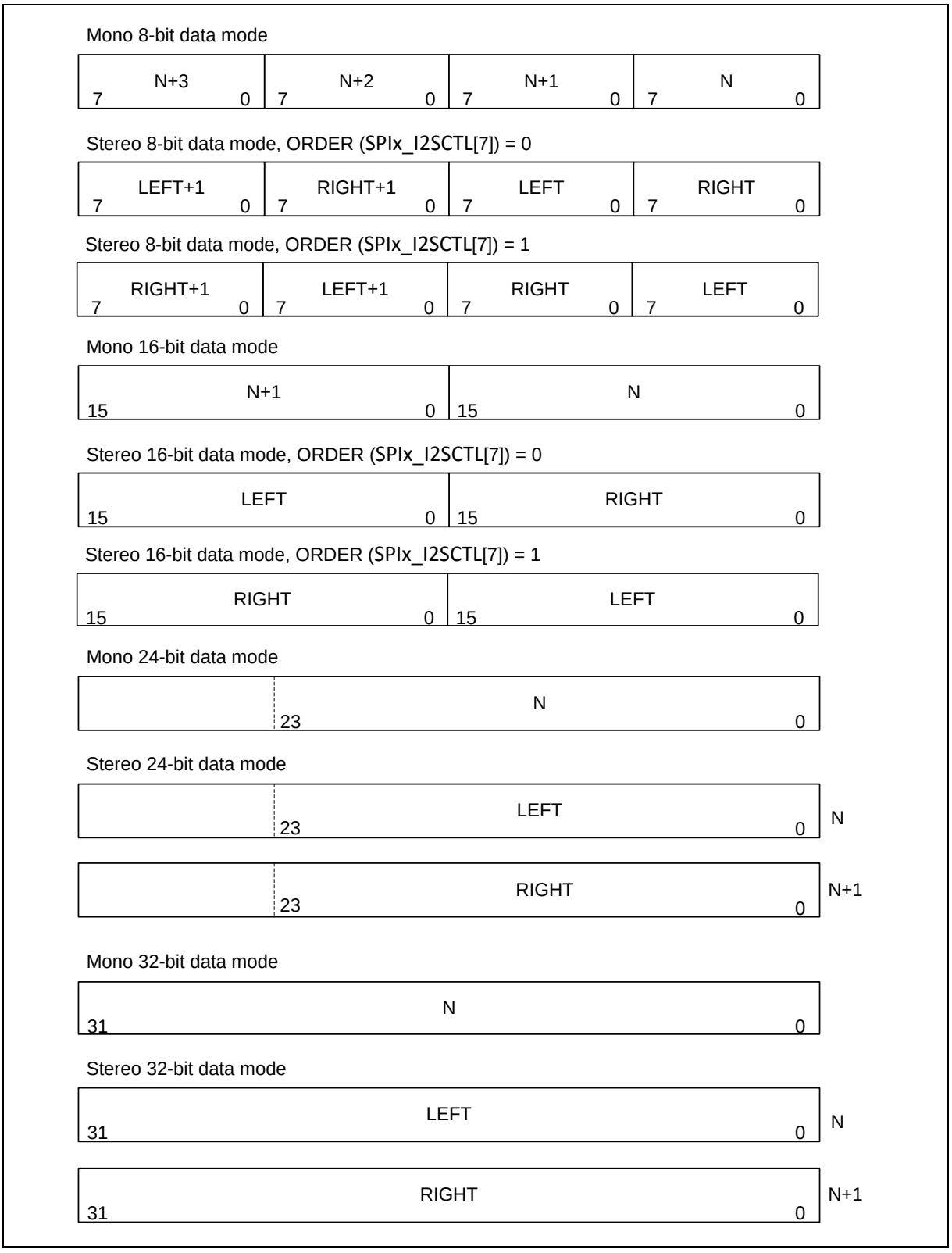


图 6.20-23 不同 I2S 模式下的 FIFO 内容

6.20.5.11 I²S/PCM主机模式和单声道模式的dummy数据个数

在I²S/PCM主机开始发送TX数据到外部从设备前，应该设定控制寄存器I2SEN(SPIx_I2SCTL[0])使能，TXEN(SPIx_I2SCTL[1])使能，和写TX数据到TX FIFO。之后主机发送dummy数据(数据为0值)到外部从设备后，主机在发送TX FIFO数据到外部从设备。表6.20-2为单声道模式的dummy数据个数，dummy数据个数的单位是L通道+R通道。

数据宽度 (SPIx_I2SCTL[5:4])	虚拟数据数目 (单元 = L 通道 + R 通道)
8位	0
16位	0
24位	1
32位	1

表 6.20-2I²S/PCM 主机模式和单声道模式下 Dummy 数据个数

6.20.5.12 I²S/PCM主机模式和立体声模式的dummy数据个数

在I²S/PCM主机开始发送TX数据到外部从设备前，应该设定控制寄存器I2SEN(SPIx_I2SCTL[0])使能，TXEN(SPIx_I2SCTL[1])使能，和写TX数据到TX FIFO。之后主机发送dummy数据(数据为0值)到外部从设备后，主机在发送TX FIFO数据到外部从设备。表6.20-3为立体声模式的dummy数据个数，dummy数据个数的单位是L通道+R通道。

数据宽度(SPIx_I2SCTL[5:4])	Dummy数据个数(单位 = L 通道 + R 通道)
8位	0
16位	0
24位	1
32位	1

表 6.20-3I²S/PCM 主机模式和立体声模式下 Dummy 数据个数

6.20.5.13 I²S/PCM从机模式和单声道模式的dummy数据个数

在I²S/PCM从机开始发送TX数据到外部主设备前，应该设定控制寄存器I2SEN(SPIx_I2SCTL[0])使能，TXEN(SPIx_I2SCTL[1])使能，SLAVE模式(SPIx_I2SCTL[8])使能和写TX数据到TX FIFO。之后从机发送dummy数据(数据为0值)到外部主设备后，从机在发送TX FIFO数据到外部主设备。表6.20-4为单声道模式的dummy数据个数，dummy数据个数的单位是L通道+R通道。

数据宽度(SPIx_I2SCTL[5:4])	Dummy数据个数(单位 = L 通道 + R 通道)
8位	3
16位	2
24位	2
32位	2

表 6.20-4I²S/PCM 从机模式和单声道模式下 Dummy 数据个数

6.20.5.14 I²S/PCM从机模式和立体声模式的dummy数据个数

在I²S/PCM从机开始发送TX数据到外部主设备前，应该设定控制寄存器I2SEN(SPIx_I2SCTL[0])使能，TXEN(SPIx_I2SCTL[1])使能，SLAVE模式(SPIx_I2SCTL[8])使能和写TX数据到TX FIFO。之后从机发送dummy数据(数据为0值)到外部主设备后，从机在发送TX FIFO数据到外部主设备。表6.20-5为立体声模式的dummy数据个数，dummy数据个数的单位是L通道+R通道。

数据宽度(SPIx_I2SCTL[5:4])	Dummy数据个数(单位 = L 通道 + R 通道)
8位	2
16位	1
24位	1
32位	1

表 6.20-5I²S/PCM 从机模式和立体声模式下 Dummy 数据个数

6.20.5.15 I²S从机模式和立体声模式的dummy数据个数

在I²S/PCM从机开始发送TX数据到外部主设备前，应该设定控制寄存器I2SEN(SPIx_I2SCTL[0])使能，TXEN(SPIx_I2SCTL[1])使能，SLAVE模式(SPIx_I2SCTL[8])使能和写TX数据到TX FIFO。之后从机发送dummy数据(数据为0值)到外部主设备后，从机在发送TX FIFO数据到外部主设备。表6.20-6为立体声模式的dummy数据个数，dummy数据个数的单位是L通道+R通道。

数据宽度(SPIx_I2SCTL[5:4])	Dummy数据个数(单位 = L 通道 + R 通道)
8位	3
16位	2
24位	2
32位	2

表 6.20-6I²S 从机模式和立体声模式下 Dummy 数据个数

6.20.5.16 PCM从机模式和立体声模式的dummy数据个数

在I²S/PCM从机开始发送TX数据到外部主设备前，应该设定控制寄存器I2SEN(SPIx_I2SCTL[0])使能，TXEN(SPIx_I2SCTL[1])使能，SLAVE模式(SPIx_I2SCTL[8])使能和写TX数据到TX FIFO。之后从机发送dummy数据(数据为0值)到外部主设备后，从机在发送TX FIFO数据到外部主设备。表6.20-7为立体声模式的dummy数据个数，dummy数据个数的单位是L通道+R通道。

数据宽度(SPIx_I2SCTL[5:4])	Dummy数据个数(单位 = L 通道 + R 通道)
8位	2
16位	1
24位	1
32位	1

表 6.20-7PCM I²S 从机模式和立体声模式下 Dummy 数据个数

6.20.6 时序图

从机选择信号的有效状态可以由 SSACTPOL (SPIx_SSCTL[2]) 配置。串行时钟的空闲状态可以通过 CLKPOL (SPIx_CTL[3]) 配置为高电平或低电平。传输字段长度在 DWIDTH (SPIx_CTL[12:8]) 中定义，发送/接收数据是以 MSB 还是 LSB 优先由 LSB 位 (SPIx_CTL[13]) 定义。用户也可以通过 TXNEG/RXNEG (SPIx_CTL[2:1]) 来选择发送/接收数据时串行时钟的边沿。四种 SPI 主机/从机操作时序图以及相关设置如下图：

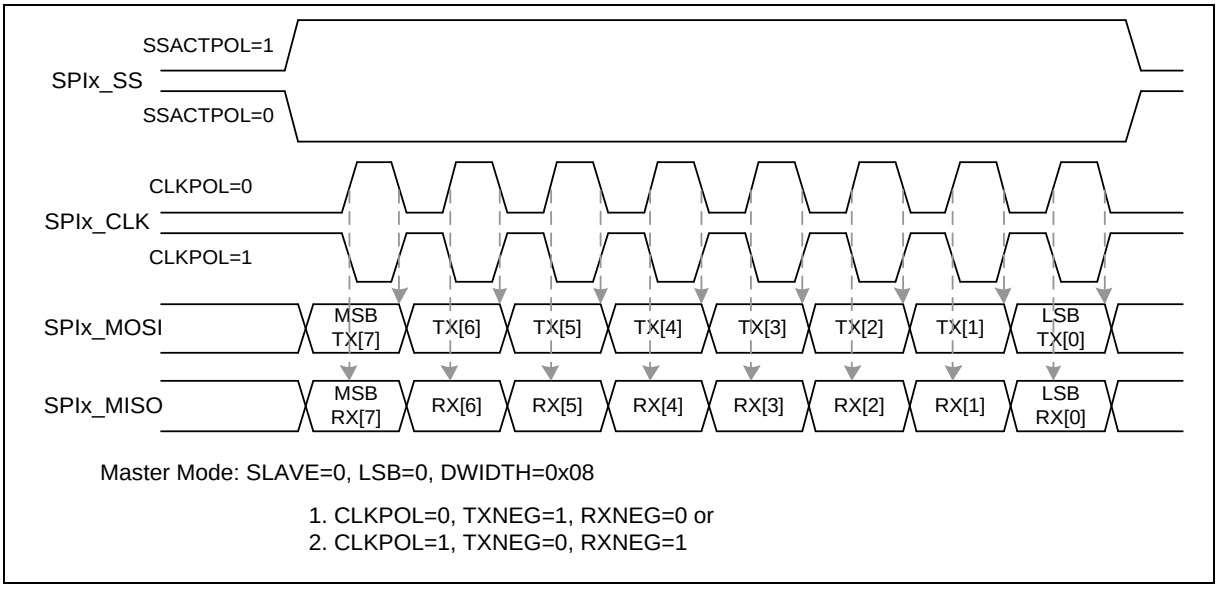


图 6.20-24 SPI 主机模式时序

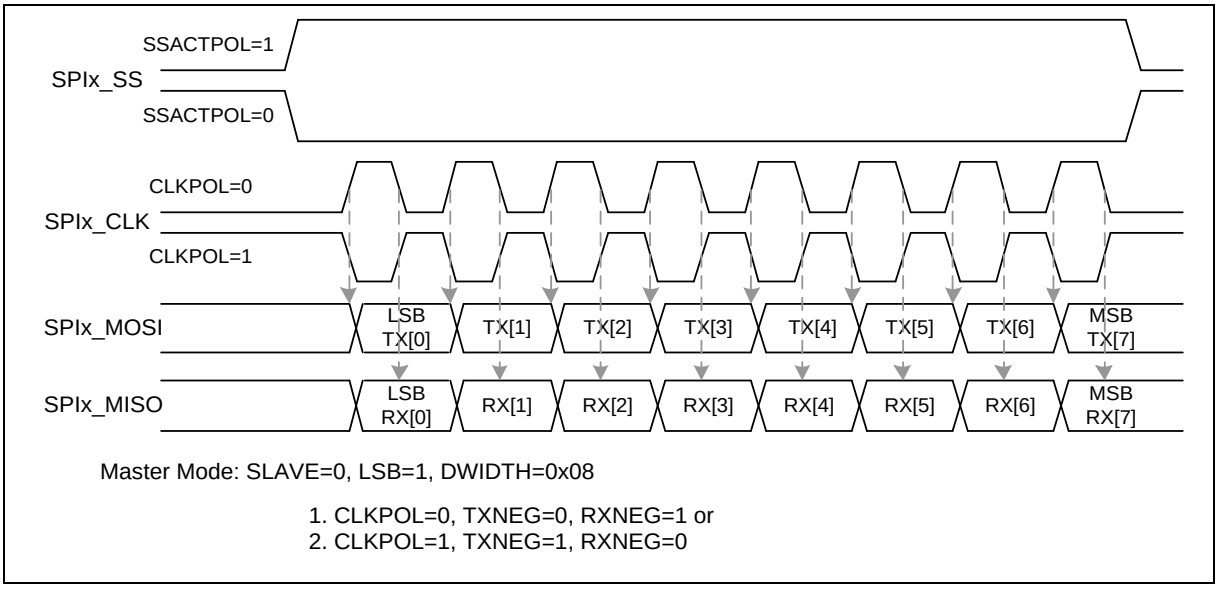


图 6.20-25SPI 主机模式时序 (SPIx_CLK 交替相位)

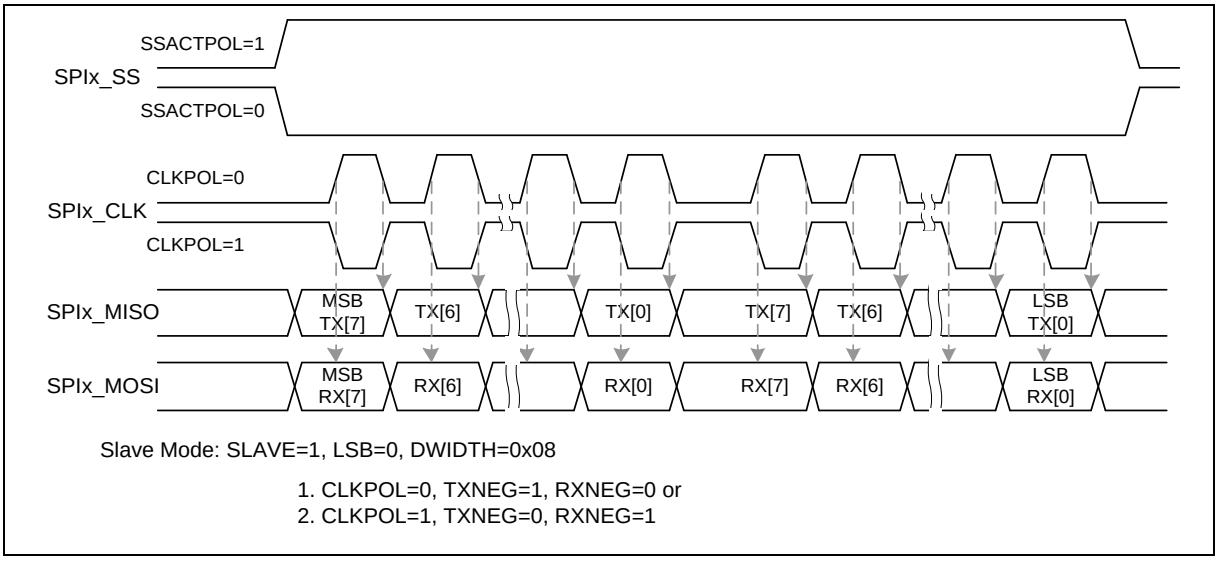


图 6.20-26 SPI 从机模式时序

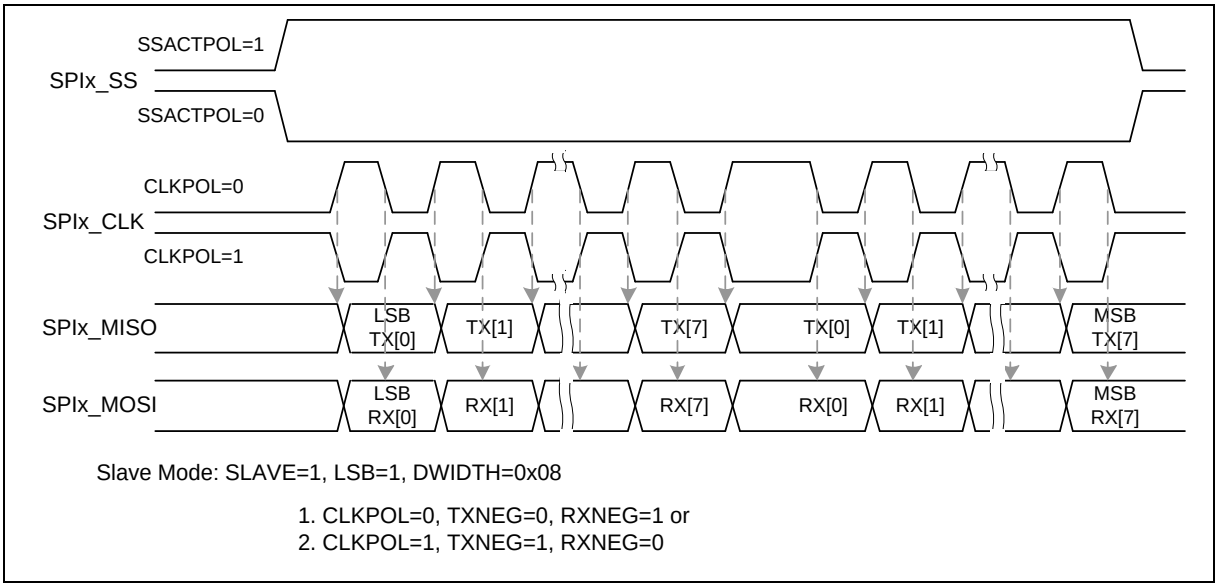


图 6.20-27 SPI 从机模式时序 (SPIx_CLK 交替相位)

6.20.7 编程示例

例 1:

SPI 配置为全双工主机时序如下：

- SPI 时钟上沿锁存数据
- SPI 时钟下沿发送数据
- MSB 优先
- SPI 时钟空闲时处于低电平
- 每次只收发一个字节

- 片选信号低电平有效。

操作流程如下:

1. 在寄存器 DIVIDER (SPIx_CLKDIV [8:0]) 配置 SPI 输出时钟频率
2. 写寄存器 SPIx_SSCTL 配置 SPI 主机相关模式:
   1) 清 0 AUTOSS (SPIx_SSCTL[3]) 关闭自动从机选择功能。
   2) 清 0 SSACTPOL (SPIx_SSCTL[2]) 设置片选信号低电平有效。
   3) 置SS (SPIx_SSCTL[0]) 为 1 使能从机片选信号，激活片外从机设备。
3. 写寄存器 SPIx_CTL 控制 SPI 主机行为：
   1) 设 SLAVE (SPIx_CTL[18]) 为 0 配置 SPI 为主机设备。
   2) 清 0 CLKPOL (SPIx_CTL[3]) 强制 SPI 空闲状态为低电平。
   3) 置 TXNEG (SPIx_CTL[2]) 为 1 配置 SPI 总线时钟下沿传输数据。
   4) 清 0 RXNEG (SPIx_CTL[1]) 配置 SPI 总线时钟上沿锁存数据。
   5) 设置传输长度 DWIDTH 为 8 位 (SPIx_CTL[12:8] = 0x08)。
   6) 清 0 LSB (SPIx_CTL[13]) 配置 MSB 传输优先。
4. 置 SPIEN (SPIx_CTL[0]) 位 1 使能数据传输。
5. SPI 主机想要写一个字节的数据到片外设备时，将数据写到 SPIx_TX 寄存器即可。
6. 等待 SPI 中断发生 (如果 UNITIEN (SPIx_CTL[17]) 为 1)，或者轮询单元传输中断标志 UNITIF (SPIx_STATUS[1])。
7. 从 SPIx_RX 读接收到的数据。
8. 重复步骤 5 继续其他数据的传输，或者配置 SS (SPIx_SSCTL[0]) 为 0 让片选信号无效以关闭片外从机设备。

例 2:

SPI为全双工从机。片外主机按下面的规则通讯：

- SPI 时钟上沿锁存数据
- SPI 时钟下沿发送数据
- LSB 优先
- SPI 时钟空闲时处于高电平
- 每次只收发一个字节
- 从机片选信号高电平有效.

操作流程如下:

1. 写寄存器 SPIx_SSCTL 配置为从机模式.
2. 置 SSACTPOL (SPIx_SSCTL[2]) 为 1 配置片选信号高电平有效。.
3. 写SPIx_CTL配置从机行为
   1) 设 SLAVE (SPIx_CTL[18]) 为 1 配置 SPI 控制球为从机设备。

2) 置 CLKPOL (SPIx_CTL[3]) 为 1 选择 SPI 空闲状态为高电平。

3) 置 TXNEG (SPIx_CTL[2]) 为 1 配置 SPI 总线时钟下降沿传输数据。

4) 清 0 RXNEG (SPIx_CTL[1]) 配置 SPI 总线时钟上升沿锁存数据。

5) 设置传输长度 DWIDTH 为 8 位 (SPIx_CTL[12:8] = 0x08)。

4. 置 LSB (SPIx_CTL[13]) 为 1 配置 LSB 传输优先

5. 置 SPIEN (SPIx_CTL[0]) 为 1。等待片外主机设备的片选信号和 SPI 时钟输入来启动数据传输

6. 如果 SPI 从机想要发送一个字节的数据到片外主机，写数据到 SPIx_TX 寄存器即可。

7. 如果 SPI 从机想要从片外主机接收一个字的数据，且用户不关心什么数据被传输，则软件软件不需要去更新 SPIx_TX 寄存器

8. 等待 SPI 中等待 SPI 中断发生 (如果 UNITIEN (SPIx_CTL[17]) 为 1)，或者轮询单元传输中断标志 UNITIF (SPIx_STATUS[1]) 。.

9. 从 SPIx_RX 寄存器读接收到的数据。.

10. 回到7继续下个字节发送或停止发送。

6.20.8 寄存器映射

R: 只读，W: 只写，R/W: 读/写

寄存器	偏移量	R/W	描述	复位值
SPI基地址: **SPIx_BA = 0x4006_1000 + (0x0000_1000 * x)** **x=0, 1, 2**				
SPIx_CTL	SPIx_BA+0x00	R/W	SPI 控制寄存器	0x0000_0034
SPIx_CLKDIV	SPIx_BA+0x04	R/W	SPI 时钟分频寄存器	0x0000_0000
SPIx_SSCTL	SPIx_BA+0x08	R/W	SPI 从机选择控制寄存器	0x0000_0000
SPIx_PDMACTL	SPIx_BA+0x0C	R/W	SPI PDMA 控制寄存器	0x0000_0000
SPIx_FIFOCTL	SPIx_BA+0x10	R/W	SPI FIFO 控制寄存器	0x2200_0000
SPIx_STATUS	SPIx_BA+0x14	R/W	SPI 状态寄存器	0x0005_0110
SPIx_TX	SPIx_BA+0x20	W	SPI 数据发送寄存器	0x0000_0000
SPIx_RX	SPIx_BA+0x30	R	SPI 数据接收寄存器	0x0000_0000
SPIx_I2SCTL	SPIx_BA+0x60	R/W	I2S 控制寄存器	0x0000_0000
SPIx_I2SCLK	SPIx_BA+0x64	R/W	I2S 时钟分频控制寄存器	0x0000_0000
SPIx_I2SSTS	SPIx_BA+0x68	R/W	I2S 状态寄存器	0x0005_0100

6.20.9 寄存器描述

SPI控制寄存器(SPIx_CTL)

寄存器	偏移量	R/W	描述	复位值
SPIx_CTL	SPIx_BA+0x00	R/W	SPI 控制寄存器	0x0000_0034

注: I^2S 模式不支持

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved			DATDIR	REORDER	SLAVE	UNITIEN	Reserved
15	14	13	12	11	10	9	8
RXONLY	HALFDPX	LSB	DWIDTH				
7	6	5	4	3	2	1	0
SUSPITV				CLKPOL	TXNEG	RXNEG	SPIEN

位	描述	
[31:21]	**Reserved**	保留
[20]	**DATDIR**	**数据方向控制** 该位用来选则半双工模式和双/四 I/O 模式下数据输入/输出方向 0 = SPI 数据输入 1 = SPI 数据输出
[19]	**REORDER**	**字节重排功能使能位** 0 = 字节重排功能禁止 1 = 字节重排功能使能. 每两个字节之间会被插入一个休眠间隔，休眠间隔的时间取决于 SUSPITV 的设置 **注:**仅在 DWIDTH 为 16，24 和 32 位时，该功能有效。
[18]	**SLAVE**	**从机模式控制** 0 = 主机模式. 1 = 从机模式
[17]	**UNITIEN**	**单元传输中断使能位** 0 = SPI 单元传输中断禁止 1 = SPI 单元传输中断使能
[16]	**Reserved**	保留
[15]	**RXONLY**	**只接收模式使能位 (仅主机)** 该只用于主机模式。在只接收模式，SPI 主机总线时钟会一直保持以接收数据，忙标志会一直置位。 0 = 只接收模式关闭

		1 = 只接收模式使能
[14]	**HALFDPX**	**SPI 半双工传输使能位** 该位用来切换 SPI 传输全双工模式和半双工模式。半双工模式下 DATDIR (SPIx_CTL[20]) 用来选择数据传输方向。 0 = SPI 工作在全双工模式 1 = SPI 工作在半双工模式
[13]	**LSB**	**LSB 优先发送** 0 = MSB, 具体发送/接收寄存器的哪一位首先被发送/接收，取决于 DWIDTH 的设定值。 1 = LSB, SPI TX 寄存器的位 0，首先被发送到 SPI 数据输出管脚，从 SPI 数据输入管脚上接收到的第一个数据位将会被放置到 RX 寄存器 (SPI_RX 的位 0) LSB 的位置.
[12:8]	**DWIDTH**	**数据宽度** 该位用来标示每次传输时会有多少位被发送/接收。最低为 8 位，最高为 32 位。 DWIDTH = 0x08 .... 8 位. DWIDTH = 0x09 .... 9 位. ..... DWIDTH = 0x1F .... 31 位. DWIDTH = 0x00 .... 32 位. **注:**对于 SPI0~SPI3 SPI 模式，该位决定了 TX/RX FIFO 的深度。因此，修改该位时，硬件会自动清空 SPI0~SPI3的 TX/RX FIFO。
[7:4]	**SUSPITV**	**休眠间隔 (仅主机)** 该四位用来配置在一次数据传输过程中连续两个发送/接收事务之间的休眠间隔。休眠间隔是从当前事务字的最后一个时钟边沿到接下来的事务的第一个边沿时钟。默认值是 0x3. 休眠间隔的周期可以根据下面公式获得： (SUSPITV[3:0] + 0.5) * SPICLK 时钟周期 例: SUSPITV = 0x0 .... 0.5 SPICLK 时钟周期 SUSPITV = 0x1 .... 1.5 SPICLK 时钟周期 ..... SUSPITV = 0xE .... 14.5 SPICLK 时钟周期 SUSPITV = 0xF .... 15.5 SPICLK 时钟周期
[3]	**CLKPOL**	**时钟极性** 0 = SPI 总线空闲时默认低 1 = SPI 总线空闲时默认高
[2]	**TXNEG**	**下降沿发送** 0 = 在 SPI 总线时钟的上升沿发送数据 1 = 在 SPI 总线时钟的下降沿发送数据
[1]	**RXNEG**	**下降沿接收** 0 = 在 SPI 总线时钟的上升沿锁存数据 1 = 在 SPI 总线时钟的下降沿锁存数据

[0]	**SPIEN**	**SPI 传输控制使能位** 在主机模式下， FIFO 缓存有数据时，该位设置为 1 后，开始传输。在从机模式下，该位置为 1 时，该设备已准备好接收数据。 0 = 禁止控制传输 1 = 使能控制传输

SPI时钟分频寄存器(SPIx_CLKDIV)

寄存器	偏移量	R/W	描述	复位值
SPIx_CLKDIV	SPIx_BA+0x04	R/W	SPI 时钟分频寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							DIVIDER
7	6	5	4	3	2	1	0
DIVIDER							

位	描述	
[31:9]	Reserved	保留
[8:0]	DIVIDER	**时钟除频** 该域的值是产生外设时钟，f_{spi_eclk}和 SPI 主机的总线时钟的除频器的值。频率计算公式如下： $f_{spi_eclk} = \frac{f_{spi_clock_src}}{(DIVIDER+1)}$ 其中， $f_{spi_clock_src}$ 为外设时钟源，定义于时钟控制寄存器 CLK_CLKSEL2 中 **注:** I^2S 模式不支持

注意:谨慎配置**DIVIDER**，SPI 时钟频率不能大于系统时钟频率

SPI从机选择控制寄存器(SPIx_SSCTL)

寄存器	偏移量	R/W	描述	复位值
SPIx_SSCTL	SPIx_BA+0x08	R/W	SPI 从机选择控制寄存器	0x0000_0000

注: I^2S 模式不支持

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved		SSINAIEN	SSACTIEN	Reserved		SLVURIEN	SLVBEIEN
7	6	5	4	3	2	1	0
Reserved				AUTOSS	SSACTPOL	Reserved	SS

位	描述	
[31:14]	**Reserved**	保留
[13]	**SSINAIEN**	**从机片选无效中断使能位** 0 = 从机片选无效中断禁止 1 = 从机片选无效中断使能
[12]	**SSACTIEN**	**从机片选有效中断使能位** 0 = 从机片选有效中断禁止 1 = 从机片选有效中断使能
[11:10]	**Reserved**	保留
[9]	**SLVURIEN**	**从机模式 TX 下溢中断使能位** 0 =从机模式 TX 下溢中断禁止 1 =从机模式 TX 下溢中断使能
[8]	**SLVBEIEN**	**从机模式位计数错误中断使能位** 0 =从机模式位计数错误中断禁止 1 = 从机模式位计数错误中断使能
[7:4]	**Reserved**	保留
[3]	**AUTOSS**	**自动从机片选功能使能位 (仅主机)** 0 = 自动从机片选功能禁止，从机片选信号由 SS (SPIx_SSCTL[0]) 控制 1 = 自动从机片选功能使能
[2]	**SSACTPOL**	**从机片选有效极性** 该位定义从机片选信号 (SPIx_SS) 有效的极性 0 = 片选信号 SPIx_SS 低电平有效 1 =片选信号 SPIx_SS 高电平有效

[1]	**Reserved**	保留
[0]	**SS**	**从机片选控制位 (仅主机)** 如果 AUTOSS 为 0, 0 = 设置 SPIx_SS 为无效状态 1 = 设置 SPIx_SS 为有效状态

SPI PDMA 控制寄存器 (SPIx_PDMACTL)

寄存器	偏移量	R/W	描述	复位值
SPIx_PDMACTL	SPIx_BA+0x0C	R/W	SPI PDMA 控制寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved					PDMARST	RXPDMAEN	TXPDMAEN

位	描述	
[31:3]	Reserved	保留
[2]	PDMARST	**PDMA 复位** 0 = 无效 1 = 复位 SPI 控制器的 PDMA 控制逻辑。该位自动清 0。
[1]	RXPDMAEN	**PDMA 接收使能位** 0 = PDMA 接收禁止 1 = PDMA 接收使能
[0]	TXPDMAEN	**PDMA 发送使能位** 0 = PDMA 发送禁止 1 = PDMA 发送使能 **注:**在 SPI 主机模式支持全双工传输，如果发送和接收 PDMA 功能都使能，接收 PDMA 功能不能在发送 PDMA 功能之前使能。用户可以先使能发送 PDMA 功能或者同时使能两个功能。

SPI FIFO控制寄存器(SPIx_FIFOCTL)

寄存器	偏移量	R/W	描述	复位值
SPIx_FIFOCTL	SPIx_BA+0x10	R/W	SPI FIFO 控制寄存器	0x2200_0000

31	30	29	28	27	26	25	24
Reserved	TXTH			Reserved	RXTH		
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved						TXFBCLR	RXFBCLR
7	6	5	4	3	2	1	0
TXUFIEN	TXUFPOL	RXOVIEN	RXTOIEN	TXTHIEN	RXTHIEN	TXRST	RXRST

位	描述	
[31]	**Reserved**	保留
[30:28]	**TXTH**	**发送 FIFO 阈值** 如果发送 FIFO 缓存的有效数据个数小于或者等于 TXTH 的设置，TXTHIF 将会被设置为 1，否则 TXTHIF 将会被设置为 0。对于 SPI0~SPI3，该域的 MSB 只在 SPI 数据长度为 8~16 时有效。
[27]	**Reserved**	保留
[26:24]	**RXTH**	**接收 FIFO 域值** 如果接收 FIFO 缓存的有效数据个数大于 RXTH 的设置，RXTHIF 将会被设置为 1，否则 RXTHIF 将会被设置为 0。对于SPI0~SPI3，该域的 MSB 只在 SPI 数据长度为 8~16 时有效。
[23:10]	**Reserved**	保留
[9]	**TXFBCLR**	**清发送 FIFO 缓存** 0 = 无效 1 = 清除发送 FIFO 指针，TXFULL 位将清除为 0 而且 TXEMPTY 位也将设置为 1。该位置 1 后，在一个系统时钟后，由硬件清 0。 **注:** TX 移位寄存器不会清空
[8]	**RXFBCLR**	**清接收 FIFO 缓存** 0 = 无效 1 =清除接收 FIFO 指针，RXFULL 位将清除为 0 而且 RXEMPTY 位将设置为 1。该位设置为 1 后，在一个系统时钟后，由硬件清 0。 **注:** RX 移位寄存器不会清空
[7]	**TXUFIEN**	**TX 下溢中断使能位** 从机模式下 TX 下溢中断事件发生时，TXUFIF (SPIx_STATUS[19]) 会被置 1。该位用来使能 TX 下溢中断。

		0 = 从机 TX 下溢中断禁止 1 = 从机 TX 下溢中断使能
[6]	**TXUFPOL**	**TX 下溢数据极性** 0 = 在从机模式，如果有发送下溢事件发生，SPI 数据输出保持为 0。 1 = 在从机模式，如果有发送下溢事件发生，SPI 数据输出保持为 1。 **注:** 1. 从机片选信号有效时如果没有任何数据发送则会发生下溢事件。 2. I^2S 模式下该位需要设为 0. 3. TX 下溢事件发生时，SPIx_MISO 管脚状态将由该位决定而不管 TX FIFO 是否为空。存储在 TX FIFO 的数据会在下一个传输帧通过 SPIx_MISO 发送出去。
[5]	**RXOVIEN**	**接收 FIFO 溢出中断使能位** 0 = 接收 FIFO 溢出中断禁止 1 = 接收 FIFO 溢出中断使能
[4]	**RXTOIEN**	**从机接收超时中断使能位** 0 = 从机接收超时中断禁止 1 = 从机接收超时中断使能
[3]	**TXTHIEN**	**发送 FIFO 阈值中断使能位** 0 = TX FIFO 阈值中断禁止 1 = TX FIFO 阈值中断使能
[2]	**RXTHIEN**	**接收 FIFO 阈值中断使能位** 0 = RX FIFO 阈值中断禁止 1 = RX FIFO 阈值中断使能
[1]	**TXRST**	**发送复位** 0 = 无效 1 = 复位发送 FIFO 指针和发送电路。TXFULL 位将被清 0，TXEMPTY 位将被置 1。该位置 1 后，大约经过 3 个系统时钟周期和 2 个外设时钟周期，硬件会将该位清除为 0。如果用户想确认复位是否已经完成，可以读取 TXRXRST (SPI_STATUS[23]) 来确认。 **注:** 如果从机模式下发生 TX 下溢事件发生，该位也可以用来使 SPI 返回空闲状态。

[0]	**RXRST**	**接收复位** 0 = 无效 1 = 复位接收 FIFO 指针和接收电路。RXFULL 位将被清除为 0，RXEMPTY 位将被设置为 1。该位置 1 后，大约经过 3 个系统时钟周期和 2 个外设时钟周期，硬件将该位清除为 0。如果用户想确认复位是否已经完成，可以读 TXRXRST (SPI_STATUS[23]) 来确认。

SPI状态寄存器(SPIx_STATUS)

寄存器	偏移量	R/W	描述	复位值
SPIx_STATUS	SPIx_BA+0x14	R/W	SPI状态寄存器	0x0005_0110

注:不支持 I²S模式

31	30	29	28	27	26	25	24
TXCNT				RXCNT			
23	22	21	20	19	18	17	16
TXRXRST	Reserved			TXUFIF	TXTHIF	TXFULL	TXEMPTY
15	14	13	12	11	10	9	8
SPIENSTS	Reserved		RXTOIF	RXOVIF	RXTHIF	RXFULL	RXEMPTY
7	6	5	4	3	2	1	0
SLVURIF	SLVBEIF	Reserved	SSLINE	SSINAIF	SSACTIF	UNITIF	BUSY

位	描述	
[31:28]	**TXCNT**	**发送 FIFO 数据个数(只读)** 该位域表明发送 FIFO 缓存的有效数据个数。
[27:24]	**RXCNT**	**接收 FIFO数据个数(只读)** 该域表明接收 FIFO 缓存有效数据个数..
[23]	**TXRXRST**	**TX 或 RX 复位状态 (只读)** 0 = TXRST 或 RXRST 的复位功能已经完成 1 = TXRST 或 RXRST 正在复位 **注意:** TXRST 和 RXRST 的复位操作都需要 3 个系统时钟周期 + 2 个外设时钟周期。用户可以检查该状态位来检测复位功能是否完成。
[22:20]	**Reserved**	保留
[19]	**TXUFIF**	**TX 下溢中断标志** 当发送下溢事件发生时，该位被设置为 1，数据输出管脚的状态取决于 TXUFPOL 的设置。 0 = 无影响 1 = 当从机片选信号有效时，发送 FIFO 和发送移位寄存器中没有数据。 **注 1**: 该位写 1 清 0。 **注 2**: 当从机片选信号有效时，如果复位从机的发送电路，在 3 个系统时钟周期和 2 个外设时钟周期后，此时复位操作已经完成，该位将被设置为 1。
[18]	**TXTHIF**	**发送 FIFO 阈值中断标志 (只读)** 0 = 发送 FIFO 缓存中的有效数据个数大于 TXTH 设置的值 1 = 发送 FIFO 缓存中的有效数据个数少于或等于 TXTH 设置的值
[17]	**TXFULL**	**发送 FIFO 缓存满标志 (只读)** 0 = 发送 FIFO 缓存未满 1 = 发送 FIFO 缓存已满

[16]	**TXEMPTY**	**发送 FIFO 缓存空标志 (只读)** 0 = 发送 FIFO 缓存不空 1 = 发送 FIFO 缓存已空
[15]	**SPIENSTS**	**SPI 使能状态位 (只读)** 0 = 禁止 SPI 控制器. 1 = 使能 SPI 控制器. **注意:** SPI 外设时钟与系统时钟不同步。为了确保 SPI 控制器已经被禁止，该位指示了 SPI 控制器的真实状态。
[14:13]	**Reserved**	保留
[12]	**RXTOIF**	**接收超时中断标志** 0 = 没有接收 FIFO 超时事件 1 = 接收 FIFO 缓存非空且主机模式下，超过 64 个 SPI 时钟周期或从机模式下超过 576 个 SPI 引擎时钟周期，接收 FIFO 缓存上没有读操作。当接收 FIFO 缓存被软件读取，超时状态会自动清 0。 **注:**该位写 1 清 0
[11]	**RXOVIF**	**接收 FIFO 溢出中断标志** 当接收 FIFO 缓存已经满了，接下来的数据将被丢弃，同时该位被设置为 1. 0 = 接收 FIFO 没有溢出 1 = 接收 FIFO 溢出 **注:**该位写 1 清 0
[10]	**RXTHIF**	**接收 FIFO 阀值中断标志(只读)** 0 = 接收 FIFO 缓存的有效数据个数少于或等于 RXTH 设置的值 1 = 接收 FIFO 缓存的有效数据个数大于 RXTH 设置的值
[9]	**RXFULL**	**接收 FIFO 缓存满标志 (只读)** 0 = 接收 FIFO 缓存未满 1 = 接收 FIFO 缓存已满
[8]	**RXEMPTY**	**接收 FIFO 缓存空标志 (只读)** 0 = 接收 FIFO 缓存不空 1 = 接收 FIFO 缓存为空
[7]	**SLVURIF**	**从机模式发送下溢中断标志** 在从机模式，如果发送下溢事件发生，而且从机片选线进入无效状态，该中断标志将被设置为 1。 0 = 未发生从机发送下溢事件 1 = 发生从机发送下溢事件 **注:**该位写 1 清 0
[6]	**SLVBEIF**	**从机模式位计数错误中断标志** 在从机模式，当从机片选信号线进入无效状态时，如果位计数与 DWIDTH 不一致，该中断标志将被设置为 1。 0 = 未发生从机模式位计数错误事件 1 = 发生从机模式位计数错误事件 **注:** 如果从机片选激活，但是没有总线时钟输入，当从机片选进入无效状态时，SLVBEIF 也将被设置为 1.该位写 1 清 0。

[5]	**Reserved**	保留.
[4]	**SSLINE**	**从机片选线总线状态 (只读)** 0 = 从机片选线状态为 0 1 = 从机片选线状态为 1 **注:** 该位只在从机模式有效。如果 SSACTPOL (SPIx_SSCTL[2]) 设置为 0，而且 SSLINE 的值为 1，SPI 从机片选处于无效状态。
[3]	**SSINAIF**	**从机片选无效中断标志** 0 = 从机片选无效中断被清除或没有发生 1 = 发生了从机片选无效中断 **注:** 该位只在从机模式有效，该位写 1 清 0。
[2]	**SSACTIF**	**从机片选激活中断标志** 0 = 从机片选激活中断被清除或未发生 1 = 发生了从机片选激活中断 **注:** 该位只在从机模式有效，该位写 1 清 0。
[1]	**UNITIF**	**单元传输中断标志** 0 = 该位清除后，没有事务完成 1 = SPI 控制器已完成一个单元传输 **注:** 该位写 1 清 0。
[0]	**BUSY**	**忙状态 (只读)** 0 = SPI 控制器处于空闲状态 1 = SPI 控制器处于忙状态. 如下是忙条件: a. SPIx_CTL[0] = 1 且 TXEMPTY = 0。 b. 对于 SPI 主机模式，SPIx_CTL[0] = 1 且 TXEMPTY = 1 但是当前传输尚未完成。 c. 对于 SPI 主机模式，SPIx_CTL[0] = 1 和 RXONLY = 1。 d. 对于 SPI 从机模式，SPIx_CTL[0] = 1，从机选择有效时仍有串口时钟输入。 e. 对于 SPI 从机模式，SPIx_CTL[0] = 1 ，从机选择无效时发送缓存或发送移位寄存器仍非空。

SPI数据发送寄存器(SPIx_TX)

寄存器	偏移量	R/W	描述	复位值
SPIx_TX	SPIx_BA+0x20	W	SPI 数据发送寄存器	0x0000_0000

31	30	29	28	27	26	25	24
TX							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
TX							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
TX							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
TX							

位	描述	
[31:0]	**TX**	**数据发送寄存器** 数据发送寄存器会把要发送的数据传入 4 级的发送 FIFO 缓存。该寄存器的有效位数在 SPI 模式下取决于 DWIDTH (SPIx_CTL[12:8])，在 I²S 模式下取决于 WDWIDTH (SPIx_I2SCTL[5:4])。 SPI 模式下，如果 DWIDTH 为 0x08，TX[7:0] 位会被发送。如果 DWIDTH 为 0x00，SPI 控制器将发送 32 位。 I²S 模式下，如果 WDWIDTH (SPIx_I2SCTL[5:4]) 为 0x2，数据宽度为 24 位，对应到该寄存器是 TX[23:0]。如果 WDWIDTH 为 0x0, 0x1, 或 0x3，该寄存器所有位均有效，具体数据排列由 I²S 模式 FIFO 操作决定。 **注:** 主机模式下，SPI 控制器写该寄存器后，1 个 APB 时钟加 6 个外设时钟后才会开始 SPI 总线时钟的传输。.

SPI数据接收寄存器(SPIx_RX)

寄存器	偏移量	R/W	描述	复位值
SPIx_RX	SPIx_BA+0x30	R	SPI 数据接收寄存器	0x0000_0000

31	30	29	28	27	26	25	24
RX							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
RX							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
RX							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
RX							

位	描述	
[31:0]	**RX**	**数据接收寄存器** SPI 控制器有 4 级 FIFO 缓存。从 SPI 数据输入管脚接收到的数据会被保存到该寄存器。如果 RXEMPTY (SPIx_STATUS[8] 或 SPIx_I2SSTS[8]) 不为 1，软件可以通过读该寄存器获取 FIFO 缓存的数据。 该寄存器只读。

I2S 控制寄存器(SPIx_I2SCTL)

寄存器	偏移量	R/W	描述	复位值
SPIx_I2SCTL	SPIx_BA+0x60	R/W	I2S 控制寄存器	0x0000_0000

注意:不支持SPI模式

31	30	29	28	27	26	25	24
SLVERRIEN	Reserved	FORMAT		Reserved		LZCIEN	RZCIEN
23	22	21	20	19	18	17	16
RXLCH	Reserved					LZCEN	RZCEN
15	14	13	12	11	10	9	8
MCLKEN	Reserved						SLAVE
7	6	5	4	3	2	1	0
ORDER	MONO	WDWIDTH		MUTE	RXEN	TXEN	I^2SEN

位	描述	
[31]	**SLVERRIEN**	**从机模式位时钟丢失中断使能位** 如该位置1且位时钟丢失事件发生，中断将产生。 0 = 禁止中断 1 = 使能中断
[30]	**Reserved**	保留
[29:28]	**FORMAT**	**数据格式选择** 00 = I^2S 数据格式 01 = MSB 对齐数据格式 10 = PCM 模式 A. 11 = PCM 模式 B.
[27:26]	**Reserved**	保留
[25]	**LZCIEN**	**左声道过零检测中断使能位** 如果该位置 1，左声道检测到过零事件时发生中断 0 = 关闭中断 1 = 使能中断
[24]	**RZCIEN**	**右声道过零检测中断使能位** 如果该位置 1，右声道检测到过零事件时发生中断 0 = 关闭中断 1 = 使能中断

[23]	**RXLCH**	**左声道接收使能位** I²S 控制器处于单声道模式 (MONO = 1) 时，如果 RXLCH 为 0，I²S 控制器接收右声道的数据；如果 RXLCH 为 1，I²S 控制器接收左声道的数据。 0 = 单声道模式下，接收右声道的数据 1 = 单声道模式下，接收左声道的数据
[22:18]	**Reserved**	保留
[17]	**LZCEN**	**左声道过零检测使能位** 如果该位为 1，当左声道数据信号位改变或下一笔数据位全为 0 ，SPIx_I2SSTS 寄存器的 LZCIF 会被置 1。该功能仅在发送操作下有效。 0 = 左声道过零检测关闭 1 = 左声道过零检测使能
[16]	**RZCEN**	**右声道过零检测使能位** 如果该位为 1，当右声道数据信号位改变或下一笔数据位全为 0 ，SPIx_I2SSTS 寄存器的 RZCIF 会被置 1。该功能仅在发送操作下有效。 0 = 右声道过零检测关闭 1 = 右声道过零检测使能
[15]	**MCLKEN**	**主时钟使能位** 如果 MCLKEN 置 1，I²S 控制球会在 SPIx_I2SMCLK 管脚产生主时钟以供外部音频设备使用 0 = 主时钟关闭 1 = 主时钟使能
[14:9]	**Reserved**	保留
[8]	**SLAVE**	**从机模式** I²S 可以工作在主机模式或从机模式。主机模式时，I2Sx_BCLK 和 I2Sx_LRCLK 管脚都为输出模式，而且可以通过该芯片为音频解码芯片发送位时钟。从机模式时，I2Sx_BCLK 和 I2Sx_LRCLK 管脚都为输入模式，而且 I2Sx_BCLK 和 I2Sx_LRCLK 的信号都来自外部的音频解码芯片。 0 = 主机模式 1 = 从机模式
[7]	**ORDER**	**FIFO 内双通道数据顺序** 0 = 左声道位于高字节 1 = 左声道位于低字节
[6]	**MONO**	**单通道数据** 0 = 双通道的数据格式 1 = 单通道的数据格式

[5:4]	**WDWIDTH**	**字宽度** 00 = 数据宽度为 8 位 01 = 数据宽度为 16 位 10 = 数据宽度为 24 位 11 = 数据宽度为 32 位
[3]	**MUTE**	**发送静音使能位** 0 = 发送数据来自缓存 1 = 发送数据为 0
[2]	**RXEN**	**接收使能位** 0 = 数据接收关闭 1 = 数据接收使能
[1]	**TXEN**	**发送使能位** 0 = 数据发送关闭 1 = 数据发送使能
[0]	**I²SEN**	**I²S 控制器使能位** 0 = I²S 模式禁止 1 = I2S 模式使能 **注:** 1. 主机模式下使能该位后，I2Sx_BCLK 将开始输出 2. 用户在更改寄存器 SPIx_I2SCTL, SPIx_I2SCLK, 和 SPIx_FIFOCTL 前，需要清 0 I2SEN (SPIx_I2SCTL[0]) 并确认 I2SENSTS (SPIx_I2SSTS[15]) 为 0。

I²S时钟分频控制寄存器(SPIx_I2SCLK)

寄存器	偏移量	R/W	描述	复位值
SPIx_I2SCLK	SPIx_BA+0x64	R/W	I2S 时钟分频控制寄存器	0x0000_0000

注意:不支持SPI模式

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved						BCLKDIV	
15	14	13	12	11	10	9	8
BCLKDIV							
7	6	5	4	3	2	1	0
Reserved	MCLKDIV						

位	描述	
[31:18]	Reserved	保留.
[17:8]	BCLKDIV	**位时钟除频** 主机模式下位时钟由 I²S 控制器产生。位时钟和 f_{BCLK}有以下关系： $f_{BCLK} = \frac{f_{i2s_clock_src}}{2 \times (BCLKDIV + 1)}$ 其中 $f_{i2s_clock_src}$ 为 I²S 外设时钟源，由寄存器 CLK_CLKSEL2 定义。 从机模式下，该域用来定义外设时钟的频率，公式为： $f_{i2s_clock_src} \div \left(\frac{BCLKDIV}{2} + 1 \right)$. I²S 从机模式外设时钟频率必须等于或大于 6 倍输入位时钟的频率。 **注:**在软件复位释放SPI IP和设置时钟分频寄存器之间的时间间隔必须大于或等于8个外设时钟。
[7]	Reserved	保留.

[6:0]	**MCLKDIV**	**主时钟除频** MCLKEN 为 1 时， I²S 控制器会产生时钟到外部音频设备。主时钟，f_{MCLK}由下列公式决定： MCLKDIV >= 1,. $f_{MCLK} = \frac{f_{i2s_clock_src}}{2 \times MCLKDIV}$ MCLKDIV = 0,. $f_{MCLK} = f_{i2s_clock_src}$ 其中， $f_{i2s_clock_src}$ 为 I²S 外设时钟频率，由 CLK_CLKSEL2 决定。通常来说，主时钟的频率为采样时钟频率的 256 倍。

注意:谨慎配置**BCLKDIV**，频率不能超过系统时钟

I²S状态寄存器(SPIx_I2SSTS)

寄存器	偏移量	R/W	描述	复位值
SPIx_I2SSTS	SPIx_BA+0x68	R/W	I2S 状态寄存器	0x0005_0100

注:不支持 SPI 模式

31	30	29	28	27	26	25	24
Reserved	TXCNT			Reserved	RXCNT		
23	22	21	20	19	18	17	16
TXRXRST	SLVERRIF	LZCIF	RZCIF	TXUFIF	TXTHIF	TXFULL	TXEMPTY
15	14	13	12	11	10	9	8
I²SENSTS	Reserved		RXTOIF	RXOVIF	RXTHIF	RXFULL	RXEMPTY
7	6	5	4	3	2	1	0
Reserved			RIGHT	Reserved			

位	描述	
[31]	**Reserved**	保留
[30:28]	**TXCNT**	**发送 FIFO 数据计数 (只读)** 该位表示发送 FIFO 缓存的有效数据个数
[27]	**Reserved**	保留
[26:24]	**RXCNT**	**接收 FIFO 数据计数 (只读)** 该位表示接收 FIFO 缓存的有效数据个数
[23]	**TXRXRST**	**TX 或 RX 复位状态 (只读)** 0 = TXRST 或 RXRST 复位完成 1 = TXRST 或 RXRST 复位中 **注:** TXRST 或 RXRST 复位需要花费 3 个系统时钟周期 + 2 个外设时钟周期。用户可以检查该位来判断复位是否完成。
[22]	**SLVERRIF**	**从机模式位时钟丢失中断标志** 0 = 没有位时钟丢失事件发生 1 = 有位时钟丢失事件发生 **注:** 该位写1清除
[21]	**LZCIF**	**左通道过零中断标志位** 0 = 左通道未发生过零事件 1 = 左通道已发生过零事件
[20]	**RZCIF**	**右通道过零中断标志位** 0 = 右通道未发生过零事件 1 = 右通道已发生过零事件
[19]	**TXUFIF**	**发送 FIFO 下溢中断标志** 如果在发送 FIFO 缓存为空， FIFO 缓存里也没有数据写入，此时如果仍有总线时钟输入，

		该位置 1 **注:**该位写 1 清 0
[18]	**TXTHIF**	**发送 FIFO 阈值中断标志 (只读)** 0 = 传送 FIFO 缓存中的有效数据个数大于 TXTH 设置的值 1 = 传送 FIFO 缓存中的有效数据个数小于或等于 TXTH 设置的值 **注:**如果 TXTHIEN = 1 且 TXTHIF = 1, SPI/I^{2}S 控制器会产生一个 SPI 中断请求
[17]	**TXFULL**	**发送 FIFO 缓存满标志位 (只读)** 0 = 发送 FIFO 缓存未满 1 = 发送 FIFO 缓存已满
[16]	**TXEMPTY**	**发送 FIFO 缓存空标志位 (只读)** 0 = 发送 FIFO 缓存非空 1 = 发送 FIFO 缓存已空
[15]	**I^{2}SENSTS**	**I^{2}S 使能状态 (只读)** 0 = SPI/I^{2}S 控制逻辑关闭 1 = SPI/I^{2}S 控制逻辑使能 **注:** SPI/I2S 外设时钟与系统时钟不同步。为了确保 SPI/I2S 控制器已经被禁止，该位指示了 SPI/I2S 控制器的真实状态。
[14:13]	**Reserved**	保留
[12]	**RXTOIF**	**接收超时中断标志** 0 = 无接收 FIFO 超时中断事件 1 = 接收 FIFO 缓存非空且主机模式下超过 64 个 SPI 时钟周期或从机模式下超过 576 个 SPI 引擎时钟周期，接收 FIFO 缓存上没有读操作。当接收 FIFO 缓存被软件读取，超时状态会自动清 0。 **注:**该位写 1 清 0
[11]	**RXOVIF**	**接收 FIFO 下溢中断标志** 如果接收 FIFO 缓存已满，之后接收的数据会被丢弃且该位置 1。 **注:**该位写 1 清 0
[10]	**RXTHIF**	**接收 FIFO 阈值中断标志 (只读)** 0 = 接收 FIFO 缓存的有效数据个数少于或等于 RXTH 设置的值 1 = 接收 FIFO 缓存的有效数据个数大于 RXTH 设置的值 **注:**如果 RXTHIEN = 1 且 RXTHIF = 1, SPI/I2S 控制器会产生一个 SPI 中断请求
[9]	**RXFULL**	**接收 FIFO 缓存满标志 (只读)** 0 = 接收 FIFO 缓存未满 1 = 接收 FIFO 缓存已满
[8]	**RXEMPTY**	**接收 FIFO 缓存空标志 (只读)** 0 = 接收 FIFO 缓存非空 1 = 接收 FIFO 缓存已空
[7:5]	**Reserved**	保留
[4]	**RIGHT**	**右声道 (只读)** 该位指示当前传输的数据是哪一个声道

		0 = 左声道 1 = 右声道
[3:0]	**Reserved**	保留

6.21 QSPI接口 (QSPI)

6.21.1 概述

QSPI接口是全双工同步串行数据通讯接口，可做为主机或从机，用4线双向通讯。M480包含2组QSPI控制器。

QSPI支持全双工的2位传输模式，也支持双I/O和四I/O传输模式。QSPI支持PDMA传输功能。

6.21.2 特征

- 支持主机模式和从机模式
- 主机模式最高到96MHz(芯片工作电压V_{DD}=2.7V~3.6V)
- 从机最高到96MHz当SPI主机支持RX数据采样时钟的对齐功能(芯片工作电压V_{DD}=2.7V~3.6V)
- 从机最高到48MHz当SPI主机不支持RX数据采样时钟的对齐功能(芯片工作电压V_{DD}=2.7V~3.6V)
- 支持2位传输模式
- 支持双I/O和四I/O传输模式
- 传输位长可为8~32
- 高低位在前可配置
- 支持字节重排功能
- 支持字节或字暂停模式
- 支持PDMA传输
- 支持三线模式，无从机片选的双向接口
- 支持一数据通道半双工传输
- 支持传输双倍发送率模式(TX DTR 模式)
- 支持只接收模式

6.21.3 框图

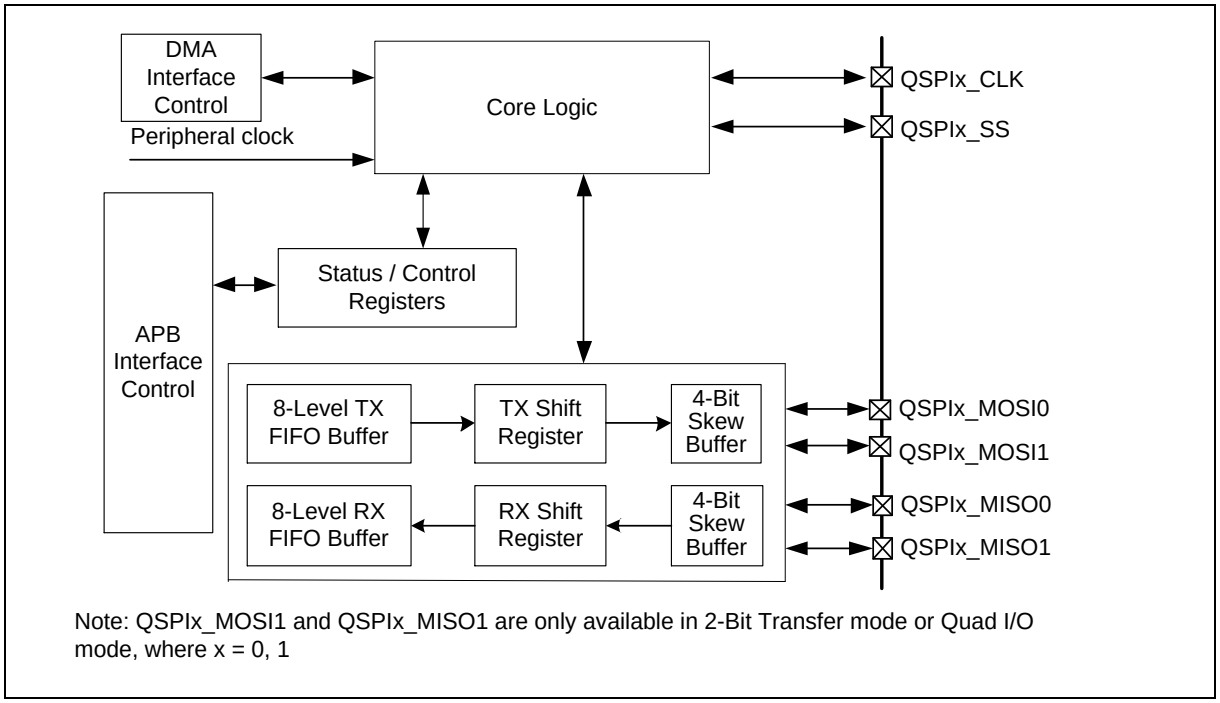


图 6.21-1QSPI 框架图

TX FIFO 缓存:

发送FIFO8级深度，32位宽，先进先出。数据通过软件写入QSPIx_TX寄存器后在写入FIFO缓存。

RX FIFO 缓存:

接收FIFO8级深度，32位宽，先进先出。接收控制逻辑器保存接收数据在FIFO缓存，软件通过读取QSPIx_RX寄存器可以得到FIFO缓存数据。

TX 移位寄存器:

发送移位寄存器是一个32位的寄存器。发送数据从TX FIFO缓存加载到这里，并一位一位的移到斜移缓存。

RX 移位寄存器:

接收移位寄存器是一个32位的寄存器。接收数据从斜移缓存一位一位的移到这里，当一个事务完成时再载入到RX FIFO。

斜移缓存:

斜移缓存是一个4级1位的缓存。总共有2个斜移缓存分别在发送和接收端。在接收端，斜移缓存用于把数据从QSPI总线移位到RX移位寄存器。在发送端，斜移缓存用于把数据从TX移位寄存器移位到QSPI总线。

6.21.4 基本配置

6.21.4.1 QSPI0基本配置

- 时钟配置

- 寄存器QSPI0SEL(CLK_CLKSEL2[3:2])配置QSPI0的时钟
- 寄存器QSPI0CKEN(CLK_APBCLK0[12])使能QSPI0的时钟

- 复位配置
  - 寄存器QSPI0RST(SYS_IPRST1[12])复位QSPI0控制器
- 管脚配置

组	管脚名	GPIO	MFP
QSPI0	QSPI0_CLK	PA.2, PH.8	MFP3
		PC.2	MFP4
		PF.2	MFP5
		PC.14	MFP6
	QSPI0_MISO0	PA.1, PE.1	MFP3
		PC.1	MFP4
	QSPI0_MISO1	PA.5, PH.10	MFP3
		PC.5	MFP4
	QSPI0_MOSI0	PA.0, PE.0	MFP3
		PC.0	MFP4
	QSPI0_MOSI1	PA.4, PH.11	MFP3
		PC.4	MFP4
	QSPI0_SS	PA.3, PH.9	MFP3
		PC.3	MFP4

6.21.4.2 QSPI1基本配置

- 时钟配置
  - 寄存器QSPI1SEL(CLK_CLKSEL2[13:12])配置QSPI1的时钟
  - 寄存器QSPI1CKEN(CLK_APBCLK1[4])使能QSPI1的时钟
- 复位配置
  - 寄存器QSPI1RST(SYS_IPRST2[4])复位QSPI1控制器
- 管脚配置

组	管脚名	GPIO	MFP
QSPI1	QSPI1_CLK	PG.12	MFP5
		PC.4	MFP14
	QSPI1_MISO0	PG.13	MFP5
		PC.3	MFP14
	QSPI1_MISO1	PG.9	MFP5

		PA.7	MFP9
	QSPI1_MOSI0	PG.14	MFP5
		PC.2	MFP14
	QSPI1_MOSI1	PG.10	MFP5
		PA.6	MFP9
	QSPI1_SS	PG.11	MFP5
		PC.5	MFP14

6.21.5 功能描述

6.21.5.1 术语

QSPI 外设时钟和 QSPI 总线时钟

QSPI控制器需要外设时钟来驱动QSPI逻辑单元执行数据传输。QSPI时钟决定于时钟分频器(SPIx_CLKDIV)和时钟源的设置，时钟源可以设置为HXT,HIRC,PLL时钟或PCLK.。寄存器CLK_CLKSEL2的QSPIxSEL位选择QSPI的时钟源。寄存器DIVIDER(QSPIx_CLKDIV[8:0])的值决定时钟的分频值。

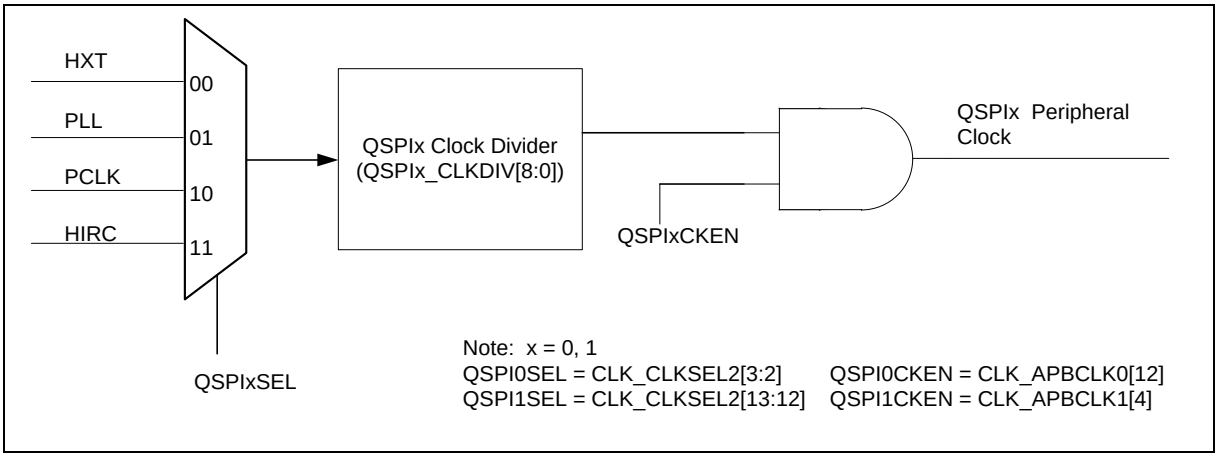


图 6.21-2QSPI 外设时钟

QSPI主机模式下，QSPI总线的时钟频率等于外设的时钟速率。从机模式下，QSPI总线时钟由主机设备提供。无论主机模式还是从机模式，QSPI时钟不能快于系统时钟。如果时钟源不是系统时钟，QSPI时钟不管是主机模式还是从机模式都必须慢于系统时钟频率。

主机/从机模式

可以通过寄存器SLAVE(QSPIx_CTL[18])设成主机模式或从机模式。QSPI传送过程中，通过HALFDPX(QSPIx_CTL[14])可以切换全双工模式或半双工模式。示意图如下：

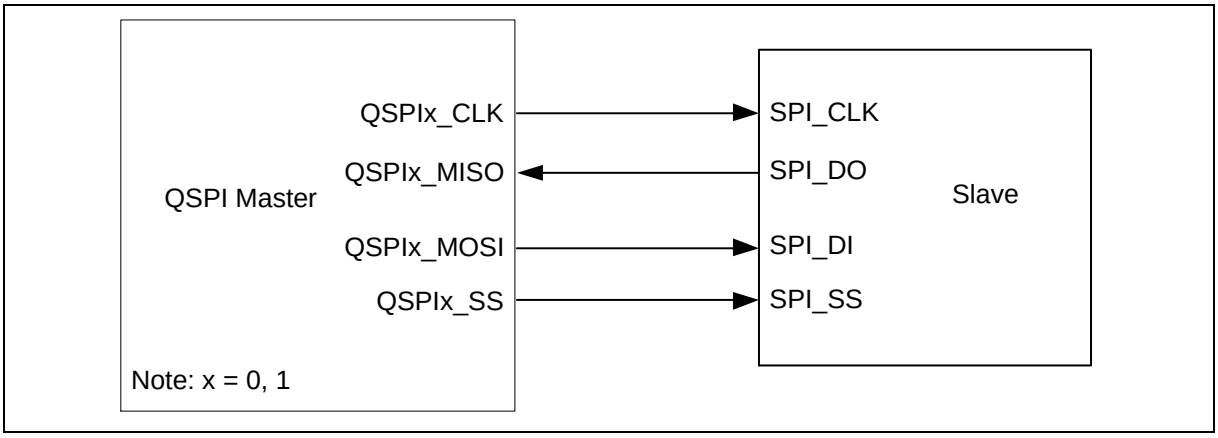


图 6.21-3QSPI 全双工主机模式

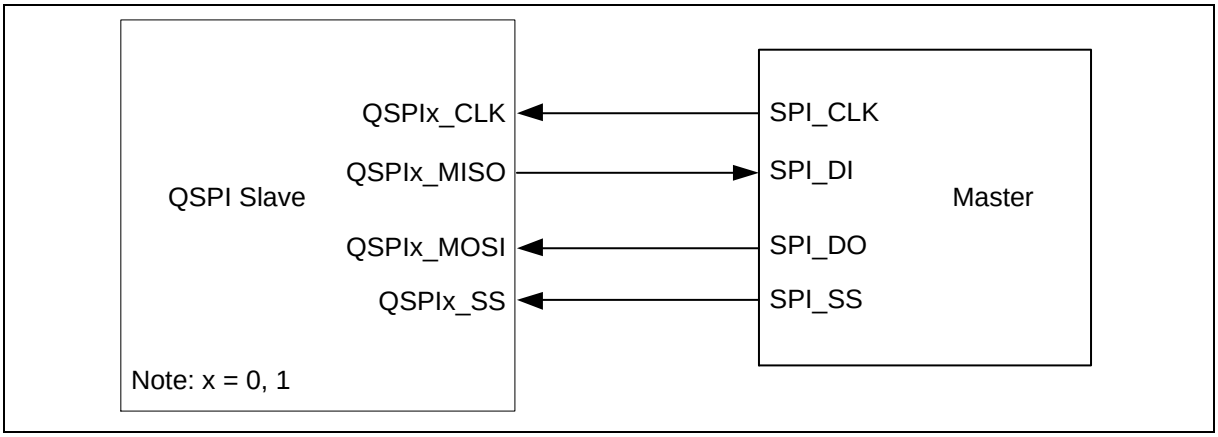


图 6.21-4QSPI 全双工从机模式

从机选择

主机模式下，QSPI通过从机片选QSPIx_SS（输出）使能从机设备。从机模式，其它主机通过QSPIx_SS（输入）使能QSPI控制器。片选有效边沿到第一个QSPI时钟输入应多于3个SPI外设时钟周期。

从机信号可通过寄存器SSACTPOL(QSPIx_SSCTL[2])设定为低有效或高有效。从机模式下，从机片选信号的无效时间必须大于等于3个外设时钟周期。

时序条件

CLKPOL(QSPIx_CTL[3])定义QSPI时钟空闲时的电平。如果CLKPOL=1，QSPI时钟空闲时输出高电平，如果CLKPOL=0，则QSPI时钟空闲时输出低电平。

TXNEG(QSPIx_CTL[2])定义数据是在QSPI时钟的下沿发送还是上沿发送。RXNEG(QSPIx_CTL[1])定义数据在QSPI时钟的下沿接收还是上沿接收。

注意:TXNEG和RXNEG的设置是互斥的，收发不能都配置为上沿或都是下沿。

接发位长

位长由DWIDTH(QSPIx_CTL[12:8])来配置，最多可为32位。

当QSPI完成一次DWIDTH(QSPIx_CTL[12:8])定义的位长的收/发时，传输中断标志将被置1。

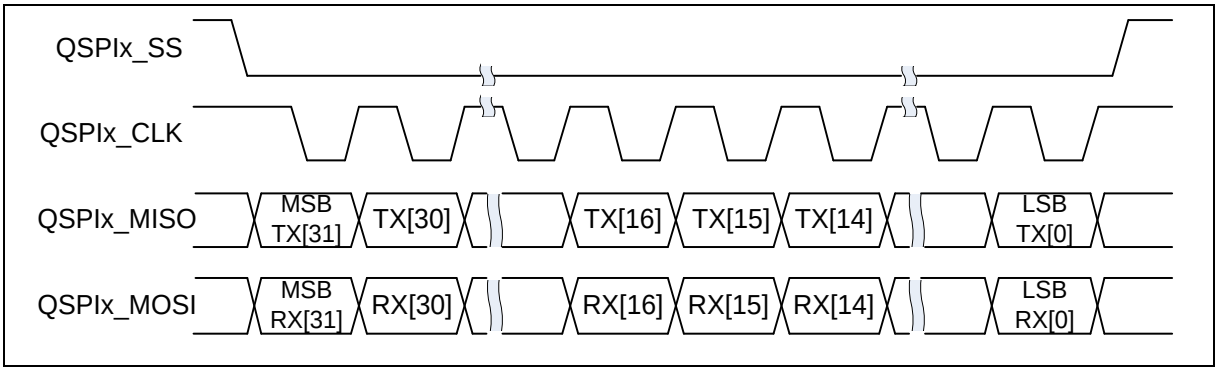


图 6.21-532 位的收发

高低位在前

LSB(QSPIx_CTL[13])定义传输顺序。LSB(QSPIx_CTL[13])=1时，低位会先传输。若(QSPIx_CTL[13])=0，则先传高位。

休眠间隔

主机模式下，SUSPITV(QSPIx_CTL[7:4])可配置在两次连续传输之间，插入0.5~15.5个QSPI时钟周期的空闲：空闲时间的计算是从前一次传输的最后一个时钟沿，到下一次传输的第一个时钟沿。SUSPITV的默认值是0x3(3.5个QSPI时钟周期)。

6.21.5.2 自动从机选择

主机模式下，如果AUTOSS(QSPIx_SSCTL[3])置位，从机选择信号根据SS(QSPIx_SSCTL[0])自动输出到QSPIx_SS管脚上。当数据写入FIFO数据传输启动，从机选择信号自动设置为有效状态。当QSPI总线空闲时，从机选择信号将自动变为无效状态。QSPI总线不空闲时——比如TX FIFO，TX移位寄存器或TX斜移缓存不空——在SUSPITV(QSPIx_CTL[7:4])的值大于或等于3的条件下，从机选择信号在两次传输之间将被置为无效状态。

主机模式下，如果SUSPITV的值小于3且AUTOSS置1时，两次连续传输之间，从机选择信号将保持有效。

如果AUTOSS被清零，从机选择输出信号的有效电平由SSACTPOL(QSPIx_SSCTL[2])来定义。

从机片选信号有效边沿，到第一个QSPI时钟边沿的间隔为1个QSPI总线时钟周期。QSPI总线的最后一个时钟，到从机选择信号无效边沿的间隔为1.5个QSPI总线时钟周期。

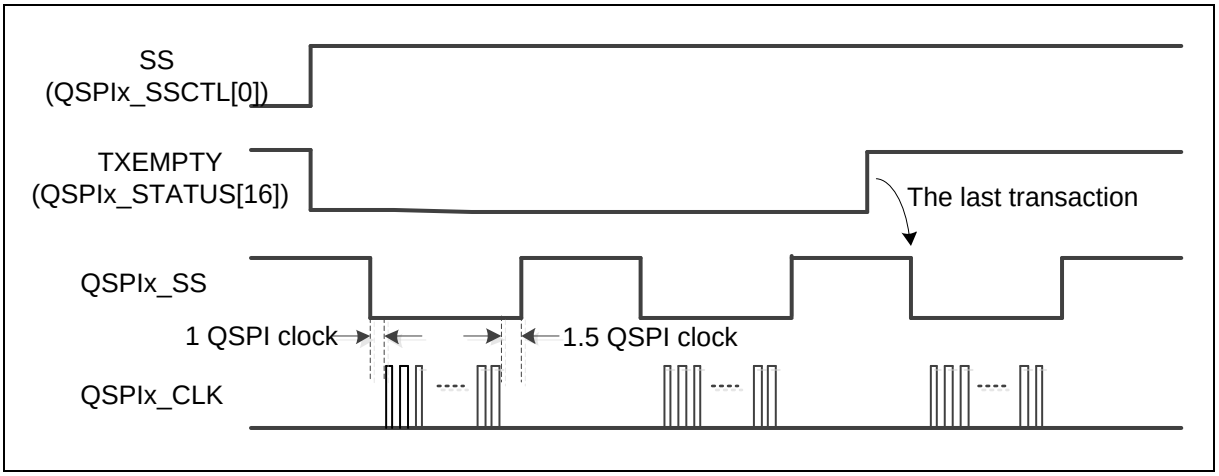


图 6.21-6 自动从机选择(SSACTPOL=0,SUSPITV>0x2)

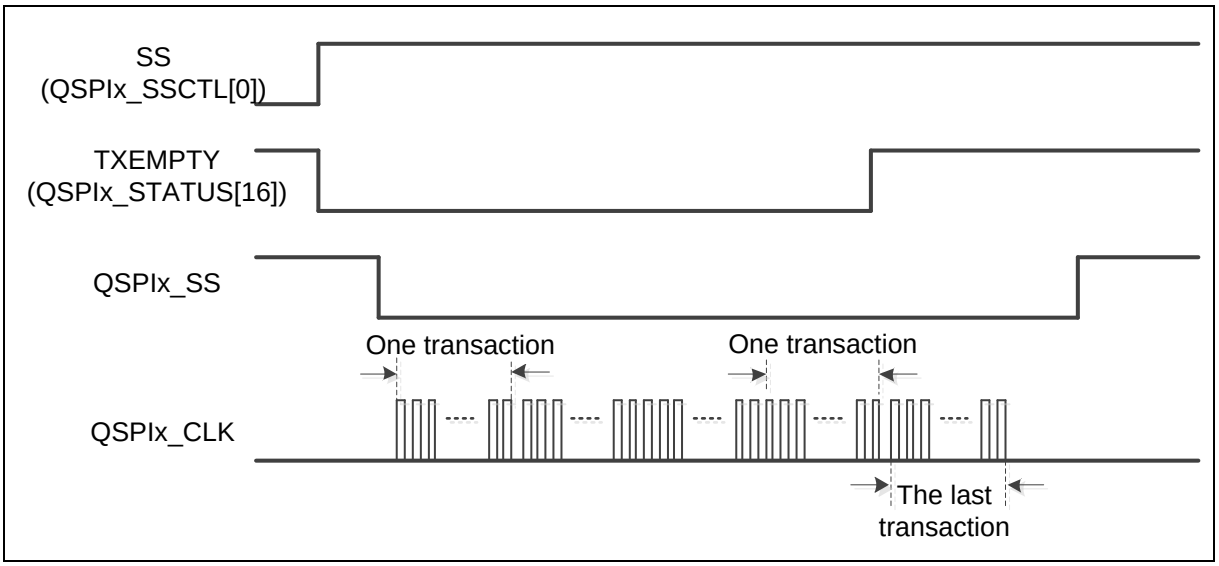


图 6.21-7 自动从机选择(SSACTPOL=0,SUSPITV<0x3)

6.21.5.3 字节重排和挂起功能

当传输设置为MSB优先(LSB=0)且使能了REORDER(QSPIx_CTL[19])，在32位传输模式(DWIDTH=0)下，存储在TX缓存与RX缓存的数据将按[BYTE0,BYTE1,BYTE2,BYTE3]的顺序重新排列。数据发送/接收的顺序为BYTE0,BYTE1,BYTE2,和BYTE3。如果DWIDTH设为24位传输模式，存储在TX缓存与RX缓存的数据将按[未知字节,BYTE0,BYTE1,BYTE2]的顺序重新排列。QSPI控制器将按照BYTE0,BYTE1,BYTE2的顺序发送/接收数据，每个字节MSB优先发送/接收。16位传输模式的规则与上面相同。字节重排序功能只适用于DWIDTH为16,24,和32位时。

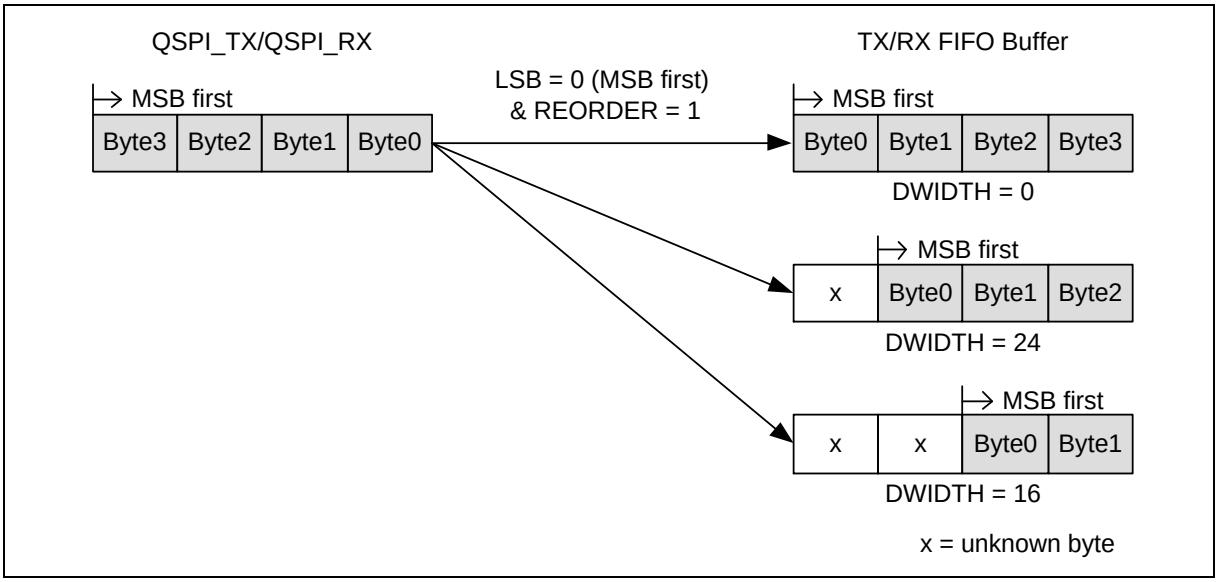


图 6.21-8 字节重排功能

主机模式下，如果REORDER(QSPIx_CTL[19])为1，两次连续传输字节之间将插入0.5~15.5个QSPI时钟周期的挂起间隔。挂起间隔时间由SUSPITV(QSPIx_CTL[7:4])设置。

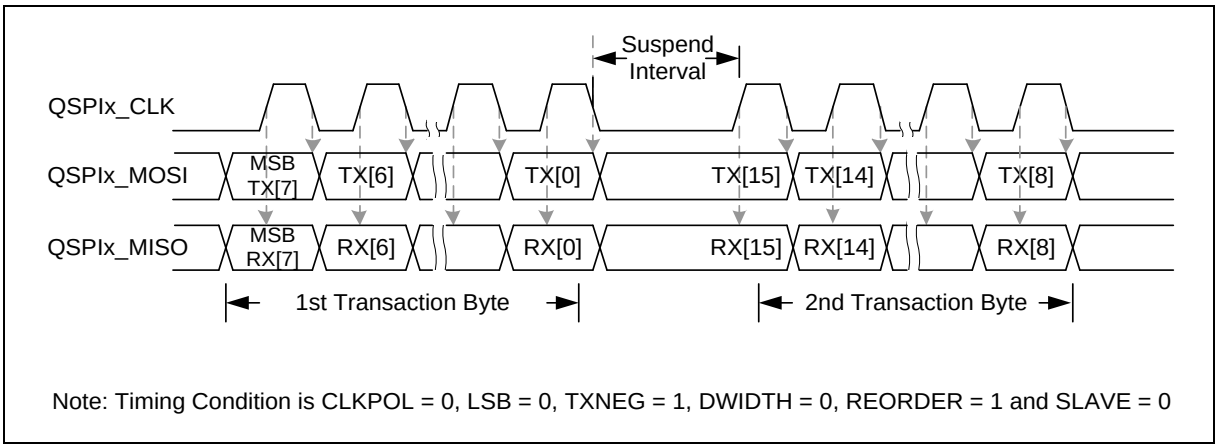


图 6.21-9 字节挂起波形图

6.21.5.4 半双工通讯

设置HALFDPX(QSPIx_CTL[14])位可以使能QSPI工作在半双工模式。半双工模式下，仅有一条数据线进行发送或接收，方向由DATDIR(QSPIx_CTL[20])配置。半双工模式下，没有用到的QSPIx_MISO管脚可以配置成GPIO供其他功能使用。使能或关闭HALFDPX(QSPIx_CTL[14])控制位将会同时产生TXFBCLR(QSPIx_FIFOCTL[9])和RXFBCLR(QSPIx_FIFOCTL[8])。

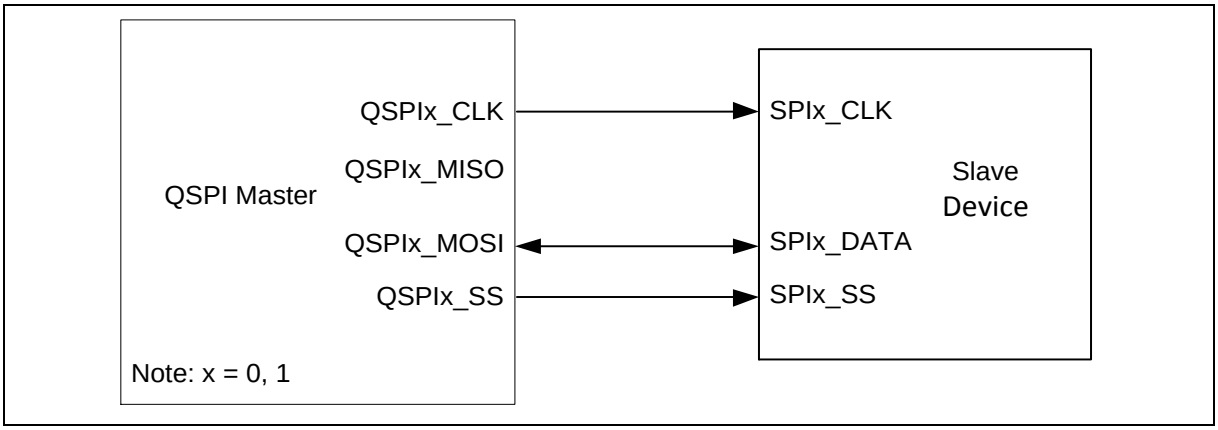


图 6.21-10QSPI 半双工主机模式

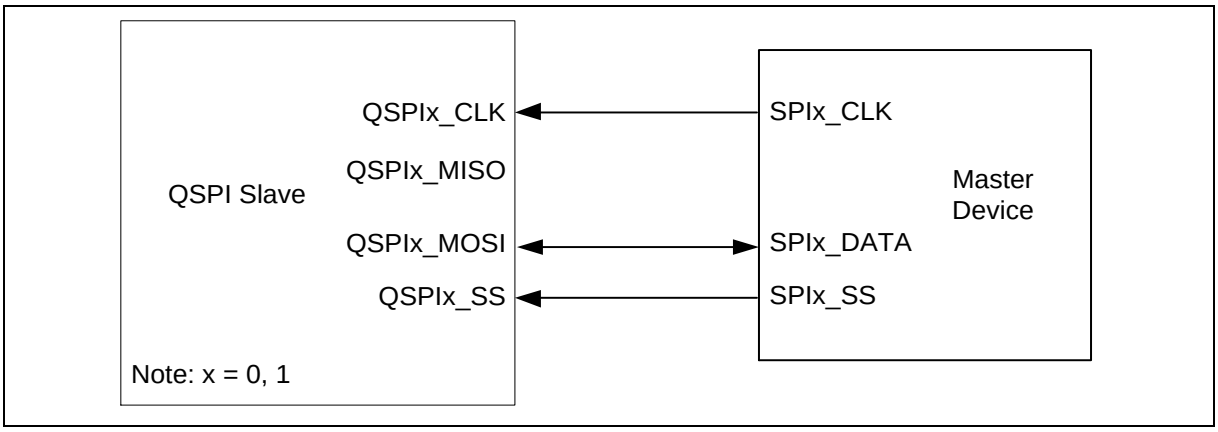


图 6.21-11QSPI 半双工从机模式

6.21.5.5 只接收模式

对于QSPI主机，可配置RXONLY(QSPIx_CTL[15])让其工作在只接收模式，QSPI主机一直不断地产生QSPI总线时钟。该模式下，如果使能了AUTOSS(QSPIx_SSCTL[3])，QSPI主机将一直保持从机片选信号有效。

在只接收模式下，没有用到的QSPIx_MOSI管脚可以配置成GPIO供其他功能使用。由于QSPI总线时钟一直存在，BUSY(QSPIx_STATUS[0])将会被置位。该模式下会同时产生TXFBCLR(QSPIx_FIFOCTL[9])和RXFBCLR(QSPIx_FIFOCTL[8])。使能该模式后，QSPI总线时钟会在6个外设时钟周期内发送出去。该模式下，写入发送FIFO的数据也会被载入发送移位寄存器发送出去。

当设置RXONLY(QSPIx_CTL[15])使能，QSPIRX接收到DWIDTH(QSPIx_CTL[12:8])设置的数据位数据后将接收到RX FIFO和QSPI时钟发送到SPI从机直到RX FIFO满。

2位传输模式TWOBIT(QSPIx_CTL[16])使能，QSPI主机发送输出时钟到从机接收RX数据直到RX FIFO满。在从RX FIFO中读出RX数据后，QSPI主机将发送输出时钟到SPI从机并再接收RX数据当RX FIFO计数器RXCNT小于或等于4。

6.21.5.6 从机三线模式

SLV3WIRE(QSPIx_SSCTL[4])置1就使能了从机3线模式QSPI控制器在无从机选择信号时正常工作。

SLV3WIRE(QSPIx_SSCTL[4])仅在从机模式有效。仅需三个管脚：QSPIx_CLK,QSPIx_MISO,和QSPIx_MOSI，QSPI控制器在无从机选择信号时正常工作，QSPIx_SS脚可配置成GPIO。在与QSPI主机通讯时，当SLV3WIRE(QSPIx_SSCTL[4])设为1时，在SPIEN(QSPIx_CTL[0])置1后就开始准备发送和接收数据。

6.21.5.7 PDMA传输功能

QSPI控制器支持PDMA传输功能。

当TXPDMAEN(QSPIx_PDMACTL[0])置1时，控制器会请求PDMA控制发送过程。

当RXPDMAEN(QSPIx_PDMACTL[1])置1时，将启动PDMA接收过程。当接收FIFO有数据时，控制器会请求PDMA接收数据。

注:QSPI只支持单一PDMA请求(读/写),不支持PDMA突发请求。

6.21.5.8 2位传输模式

TWOBIT(QSPIx_CTL[16])为1时，使能2位传输模式，QSPI进行全双工数据传输，可同时收发两位数据。

主机模式下，寄存器QSPIx_TX的偶数位(TX Data(n))通过QSPIx_MOSI0管脚发送，奇数位(TX Data(n+1))通过QSPIx_MOSI1管脚发送。同时，从QSPIx_MOSI0管脚接收到的偶数位数据将优先于从QSPIx_MOSI1管脚接收到的奇数位数据写入接收FIFO。

从机模式下，寄存器QSPI_TX中的偶数位和奇数位将分别从QSPIx_MISO0管脚和QSPIx_MISO1管脚发送。同时，从QSPIx_MISO0管脚接收到的偶数位和从QSPIx_MISO1管脚接收到的奇数位将分别写入QSPI_RX寄存器。FIFO缓存的数据顺序与主机模式是一样的。

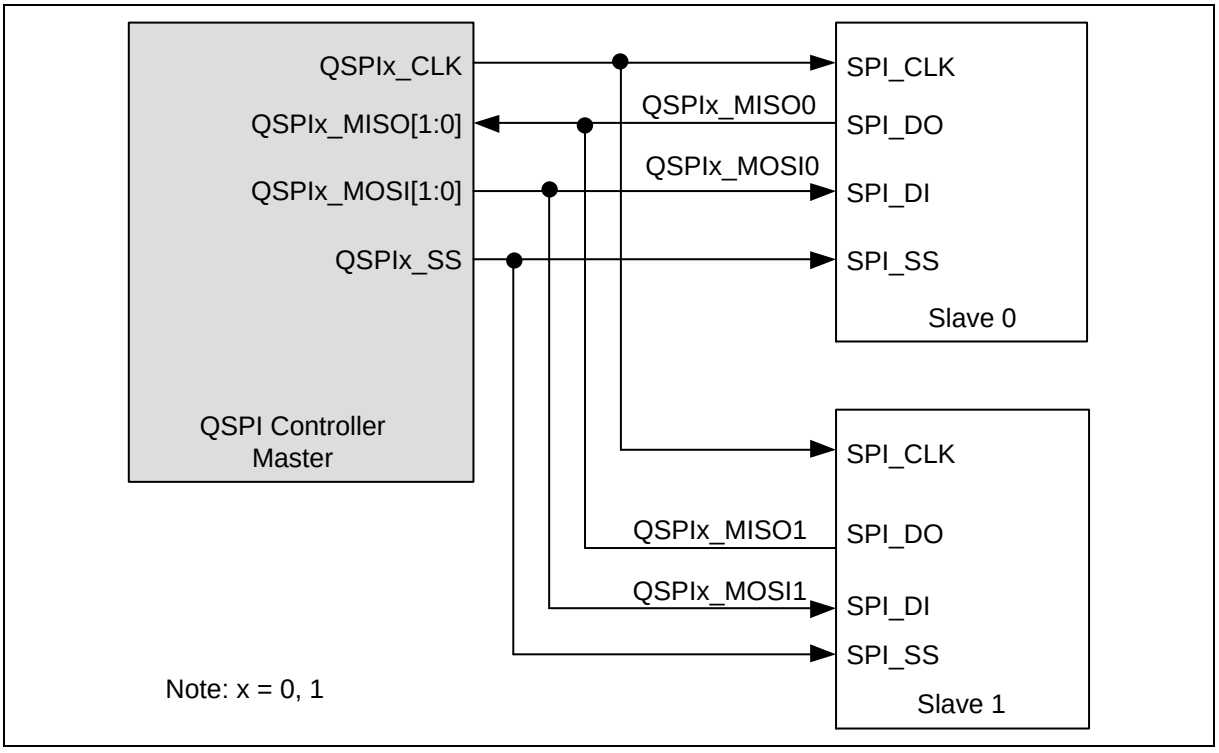


图 6.21-122 位传输模式

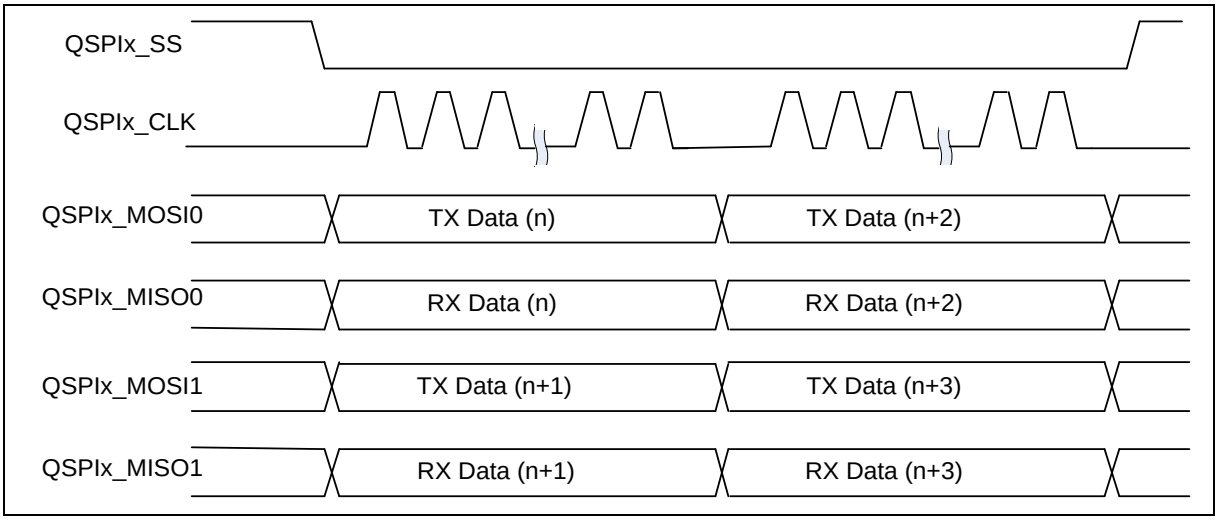


图 6.21-132 位传输模式的时序图(主机模式)

6.21.5.9 双I/O模式

DUALIOEN((QSPIx_CTL[21])置1，QSPI配置为双I/O传输。DATDIR(QSPIx_CTL[20])用来定义传输数据的方向。DATDIR=1时，为双路发送，反之为双路接收。该功能支持8,16,24,和32位长度。

从机三线模式或字节重排功能时，不支持双I/O模式。

针对双I/O模式，如果DUALIOEN(QSPIx_CTL[21])和DATDIR(QSPIx_CTL[20])都设置成1，则QSPIx_MISO0输出偶数位，QSPIx_MISO0输出奇数位。如果DUALIOEN(QSPIx_CTL[21])置1，DATDIR(QSPIx_CTL[20])置0，QSPIx_MISO0和QSPIx_MOSI0都为输入端口。

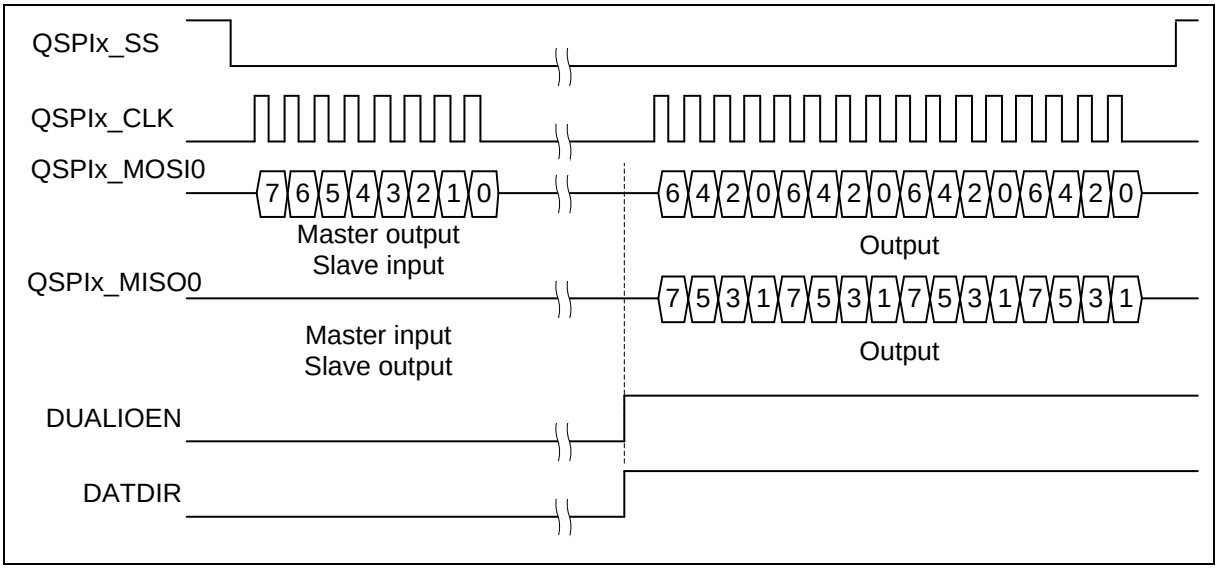


图6.21-14双输出模式的位时序

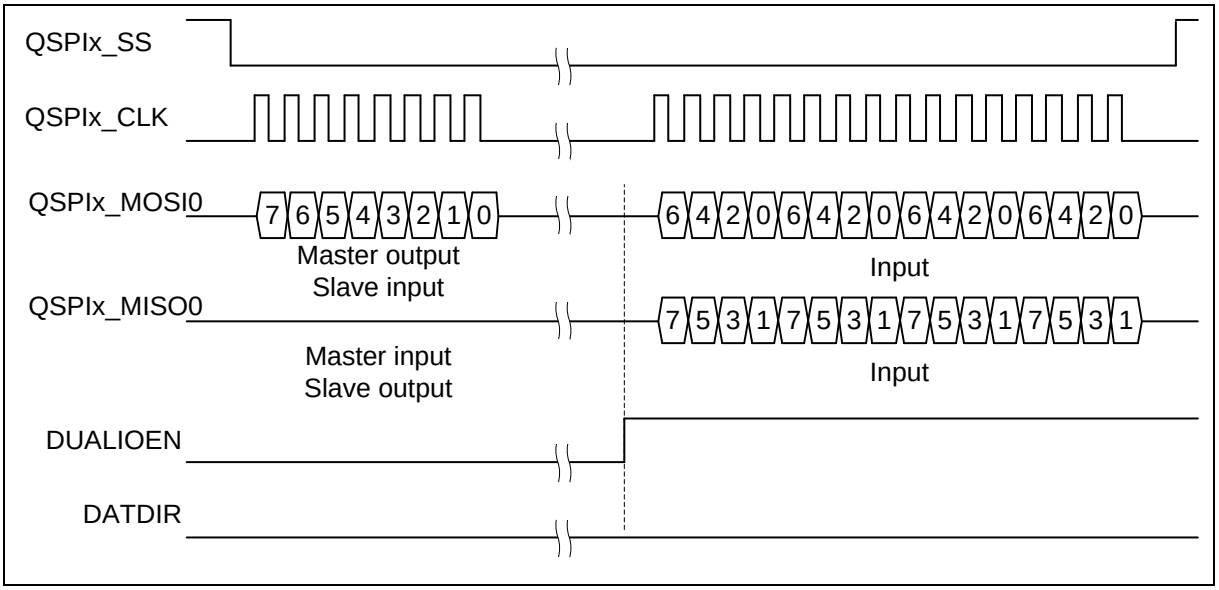


图 6.21-15 双输入模式的位时序

6.21.5.10 四I/O模式

定义传输方向。当DATDIR(QSPIx_CTL[20])=1，为发送，反之为接收。该功能支持8,16,24,和32位长度。

从机三线模式或字节重排功能时，不支持4 I/O模式。DUALIOEN(QSPIx_CTL[21])和QUADIOEN(QSPIx_CTL[22])不能同时设置为1。

对于四 I/O模式，如果QUADIOEN(QSPIx_CTL[22])和DATDIR(QSPIx_CTL[20])都设成1，则QSPIx_MOSI0和QSPIx_MOSI1输出偶数位，QSPIx_MOSI0和QSPIx_MOSI1输出奇数位。如果QUADIOEN(QSPIx_CTL[22])置1，DATDIR(QSPIx_CTL[20])置0，QSPIx_MISO0,QSPIx_MISO1,QSPIx_MOSI0和QSPIx_MOSI1都是输入。

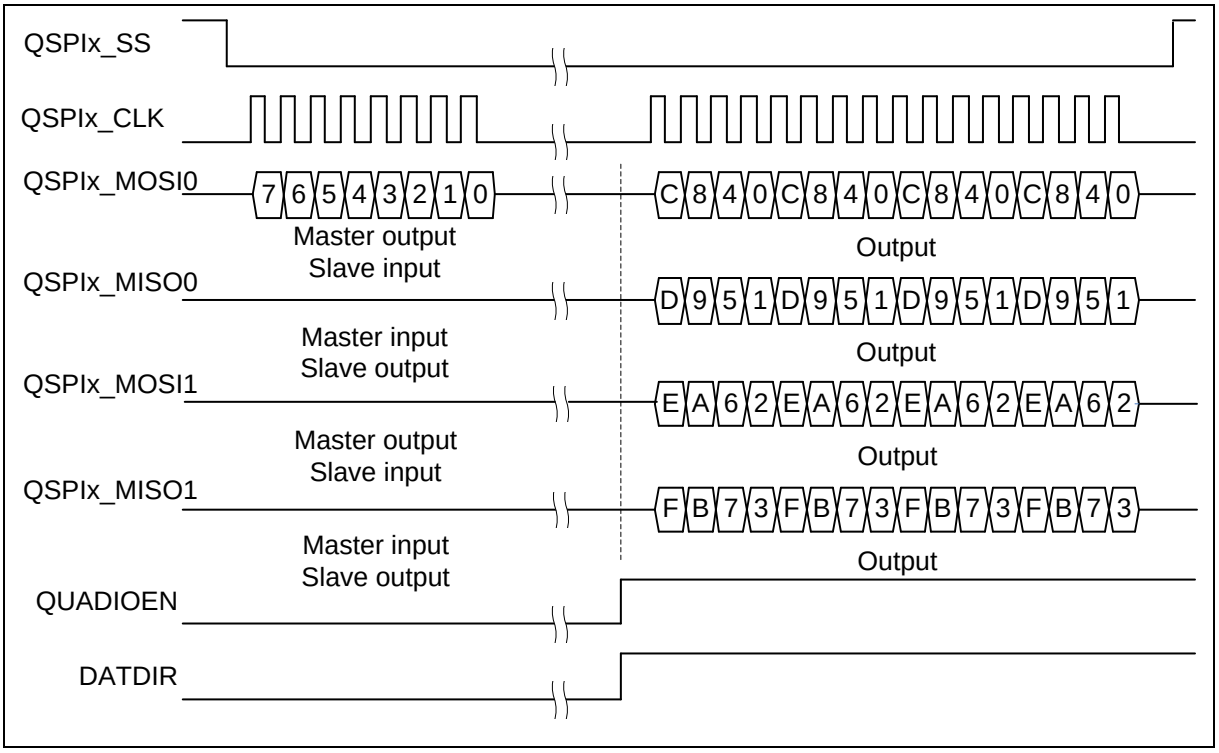


图 6.21-16 四 I/O 输出模式位顺序

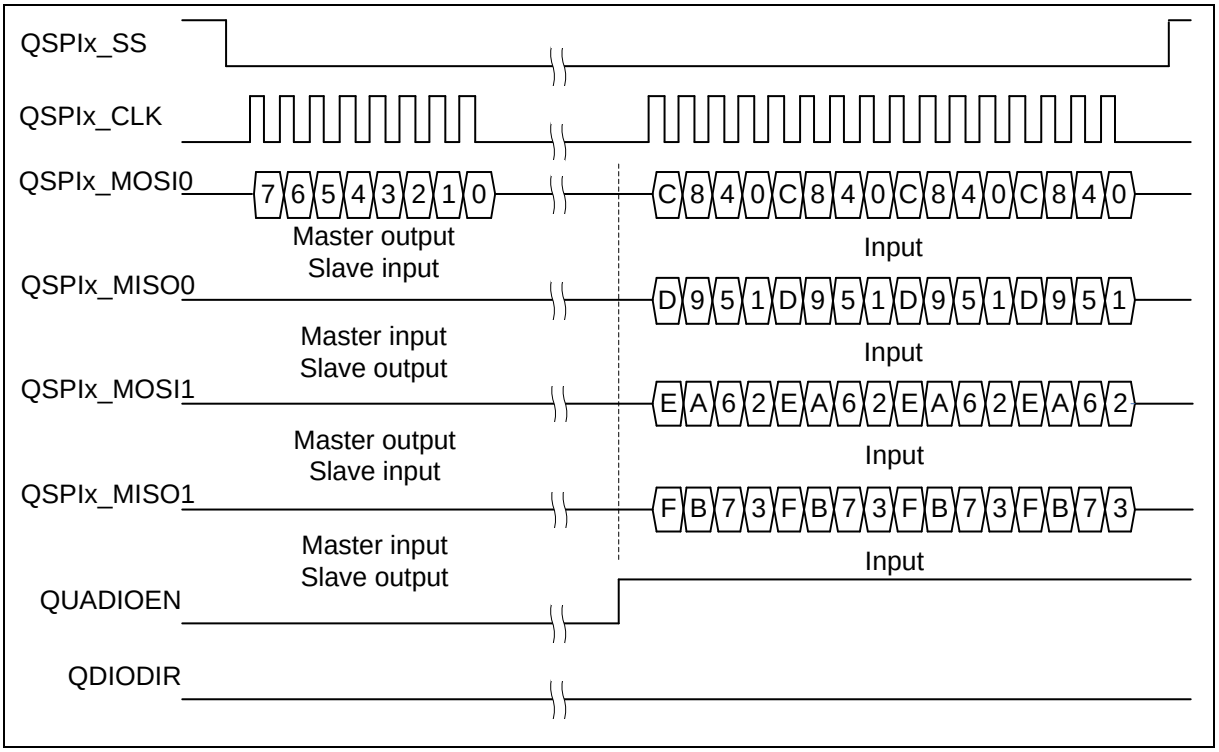


图 6.21-17 四 I/O 输入模式位顺序

6.21.5.11 传输双倍发送率模式

QSPI主机模式支持TXDTR模式（传输双倍发送率模式）当设置控制寄存器TXDTR_EN(QSPIx_CTL[23])为1。TX DTR模式时序如图6.21-18。

TX DTR模式，QSPI总线时钟(QSPIx_CLK)为QSPI外设时钟(QSPI_ECLK)频率的一半。TX DTR模式下，QSPITX输出数据将在QSPIx_CLK上升沿和下降沿附件，其中QSPIx_CLK的上升沿和下降沿是QSPI外设时钟(QSPI_ECLK)在QSPI输出数据QSPIx_MOSI的数据驱动边缘之后的半周期。

用户能选择TX DTR模式下QSPIx_CLK空闲电平，通过控制寄存器CLKPOL(QSPI_CTL[3。当设置CLKPOL为0,QSPIx_CLK的空闲电平为0。当设置CLKPOL为0,1SPIx_CLK的空闲电平为1。TX DTR模式，TXNEG(QSPI_CTL[2])等同CLKPOL(QSPI_CTL[3]),和SUSPITV(QSPI_CTL[7:4])等于0。SPI主机在TX DTR模式下，不能从SPI从机接收数据。

注：QSPI主机模式支持TX DTR模式，QSPI从机模式不支持。

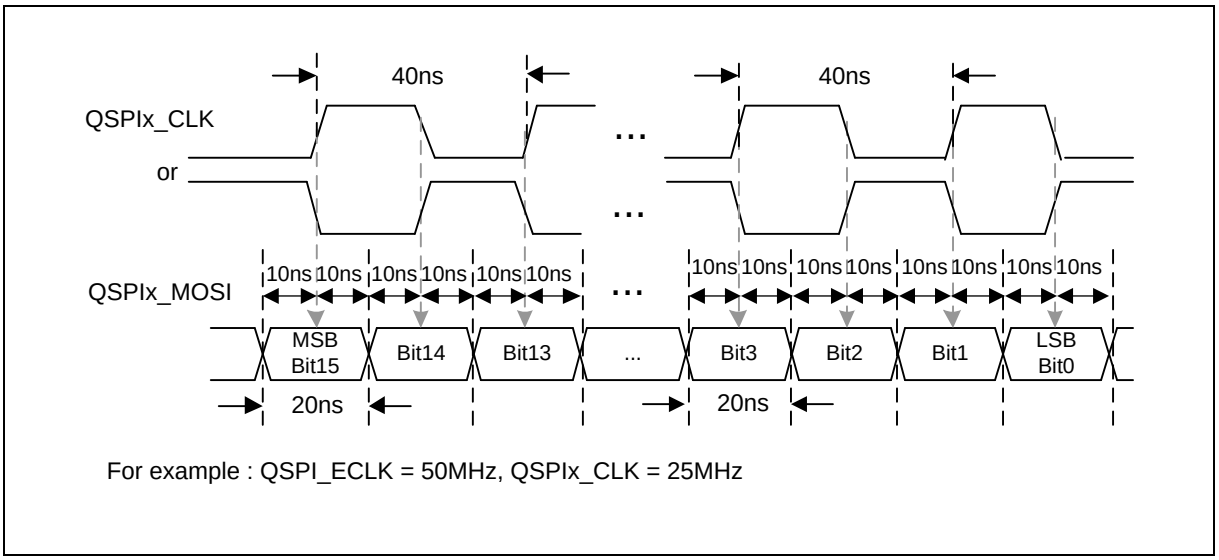


图 6.21-18TX DTR 模式下 QSPI 输出数据时序图

6.21.5.12 FIFO缓存操作

QSPI控制器配备了32位宽的发送和接收FIFO缓存。存放在发送FIFO缓存的数据会通过发送控制逻辑进行读取和发送。如果发送FIFO缓存满了，TXFULL(QSPIx_STATUS[17])会被置1。当QSPI传输逻辑单元抽出发送FIFO缓存的最后一个数据，发送FIFO缓存就为空了，TXEMPTY(QSPIx_STATUS[16])位被置1。注意TXEMPTY(QSPIx_STATUS[16])标志在最后一笔数据传输还在进行时就被置1了。在主机模式，当FIFO缓存写入数据或者QSPI总线上有任何事务，BUSY(QSPIx_STATUS[0])位被设置为1。(如：从机片选信号激活和QSPI控制器在从机模式正在接收数据)。当传送缓存为空且当前事务已经完成后，该位将设置为0。因此，软件可以检查BUSY(QSPIx_STATUS[0])位的状态以确认QSPI是否已经空闲。

接收控制逻辑存储QSPI接收到的数据到接收FIFO缓存。有FIFO相关的状态位，像RXEMPTY(QSPIx_STATUS[8])和RXFULL(QSPIx_STATUS[9])，来表明当前FIFO缓存的状态。

发送和接收的阀值可以通过设置TXTH(QSPIx_FIFOCTL[30:28])和RXTH(QSPIx_FIFOCTL[26:24])来设定。当存储在发送FIFO缓存的有效数据数小于或等于TXTH(QSPIx_FIFOCTL[30:28])的设定时，TXTHIF(QSPIx_STATUS[18])位会被置1。当存储在接收FIFO缓存的有效数据数大于RXTH(QSPIx_FIFOCTL[26:24])的设定，RXTHIF(QSPIx_STATUS[10])位会被置1。

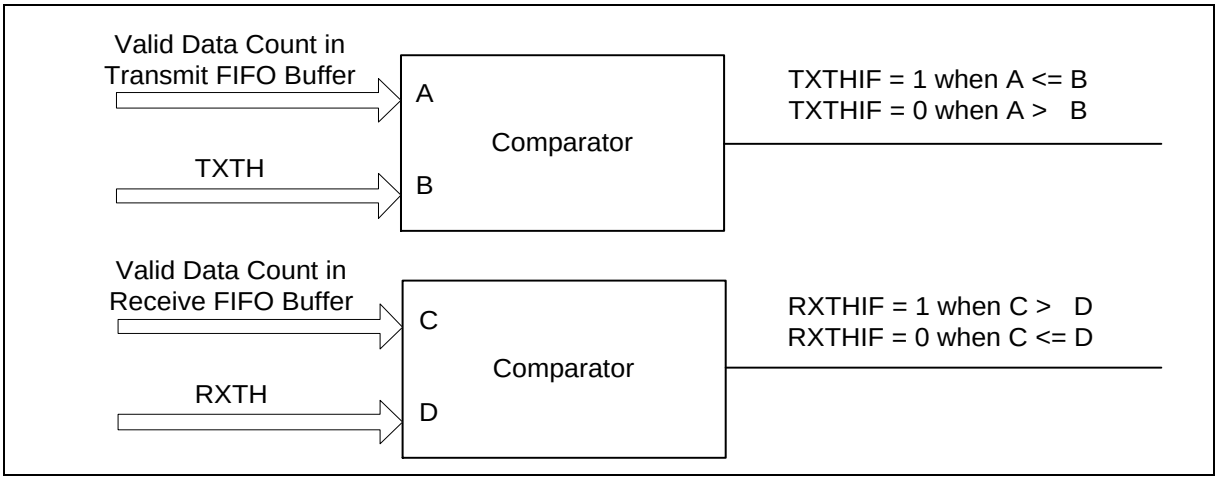


图 6.21-19 FIFO 阈值比较器

在主机模式，当第一个数据写入QSPIx_TX寄存器时，TXEMPTY(QSPIx_STATUS[16])标志将会被清除为0。在1个APB时钟周期和6个外设时钟周期后，发送将开始。用户可以立即写下一个数据到QSPIx_TX寄存器。QSPI控制器将会在两个连续的事务之间插入一个休眠间隔，休眠间隔的长度由SUSPITV(QSPIx_CTL[7:4])的设定值决定。如果SUSPITV(QSPIx_CTL[7:4])等于0，QSPI控制器可以执行连续发送。只要TXFULL(QSPIx_STATUS[17])为0，用户就可以向QSPIx_TX寄存器写入新的数据。.

如图6.21-20的例1所示，该图指明了TXEMPTY(QSPIx_STATUS[16])的更新条件以及FIFO缓存，移位寄存器和斜移缓存之间的关系。当Data0写入FIFO缓存时，TXEMPTY(QSPIx_STATUS[16])位被设置为0。Data0将由内核逻辑加载到移位寄存器，此时，TXEMPTY(QSPIx_STATUS[16])将设置为1。移位寄存器中Data0将逐位移入斜移缓存传输直到传输完成。

在例2中，该图为当FIFO缓存中有8个数据时，更新TXFULL(QSPIx_STATUS[17])的条件。当TXFULL=1时，下一数据Data9不能被写入FIFO缓存。

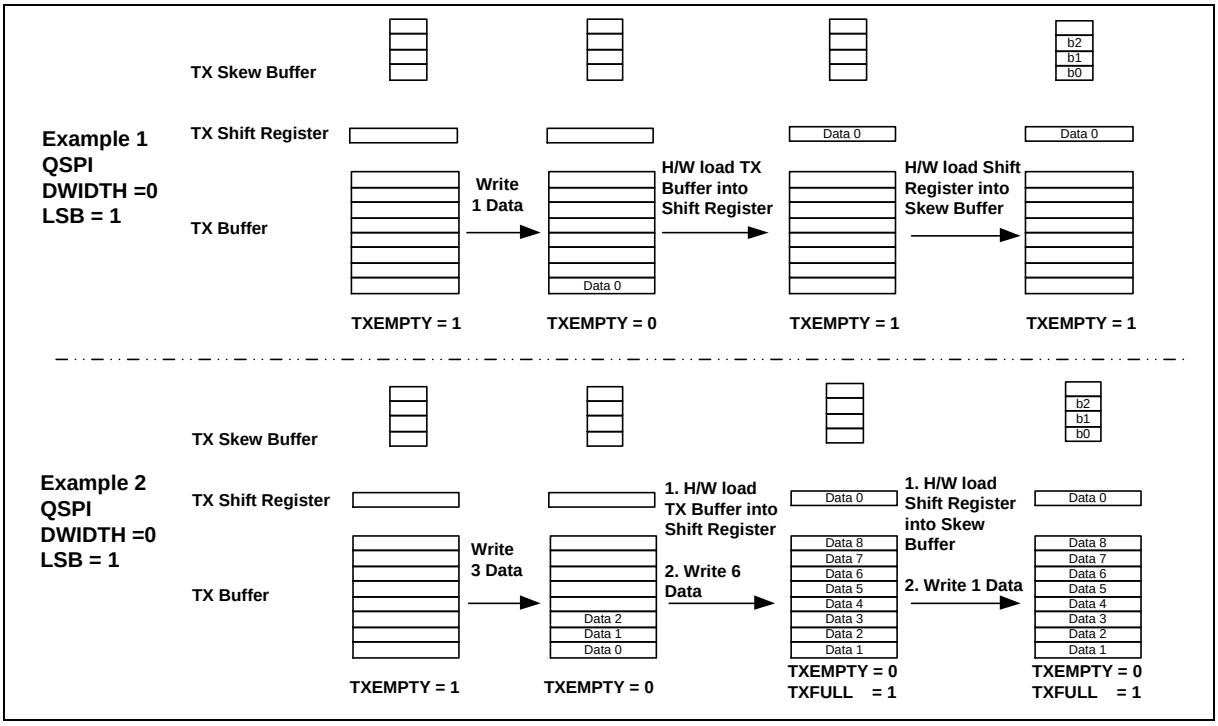


图 6.21-20 发送 FIFO 缓存示例

如果要发送的数据更新及时，接下来的事务将会被自动触发。如果在所有数据传输完成之后，QSPIx_TX寄存器没有被更新，则传输停止。

在主机模式接收操作中，串行数据从QSPIx_MISO管脚接收并被存储在接收FIFO缓存。

接收数据(Data0’sb0,b1,…b31)通过串行时钟(QSPIx_CLK)先存到斜移缓存，然后再逐位移到移位寄存器。当接收数据位达到DWIDTH(QSPIx_CTL[12:8])的值时，内部逻辑将移位寄存器中的数据加载到FIFO缓存。当接收FIFO缓存有未读数据时(参见图6.21-21示例1)，RXEMPTY(QSPIx_STATUS[8])将清0。只要RXEMPTY(QSPIx_STATUS[8])为0，用户就可以通过QSPIx_RX寄存器来读取接收的数据。如果接收FIFO缓存包含8个未读数据，RXFULL(QSPIx_STATUS[9])将设置为1(参见图6.21-21示例2)。

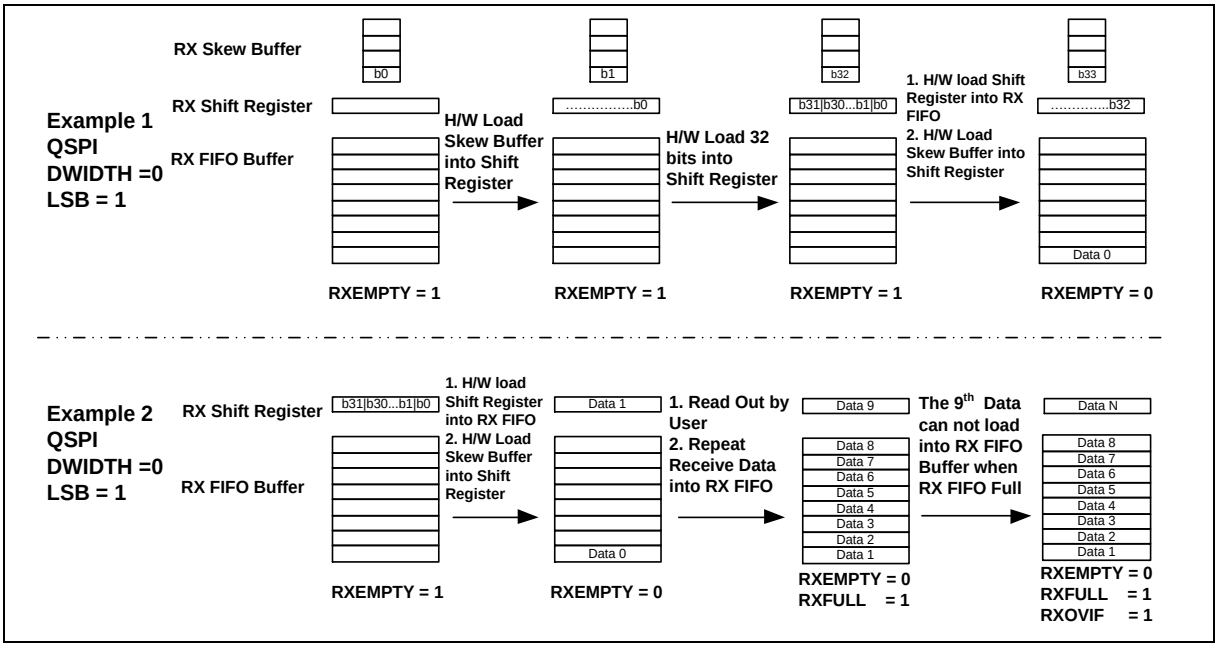


图 6.21-21 接收 FIFO 缓存示例

在从机模式中，当软件写数据到QSPIx_TX寄存器时，数据将会被加载到发送FIFO缓存，同时TXEMPTY(QSPIx_STATUS[16])标志也将被设置为0。当从设备从主机接收到时钟信号时，发送操作将会开始。只要TXFULL(QSPIx_STATUS[17])标志为0，用户就可以写数据到QSPIx_TX寄存器。所有数据都被QSPI发送逻辑单元发送出去后，而且软件没有再更新QSPIx_TX寄存器TXEMPTY(QSPIx_STATUS[16])标志将会被设置为1。

当从机片选信号有效时，如果没有任何数据写入QSPIx_TX寄存器，发送下溢标志TXUFIF(QSPIx_STATUS[19])将被设置为1。输出的数据在本次传输中将会通过设置TXUFPOL(QSPIx_FIFOCTL[6])保留，直到从机片选信号为无效状态。当传送下溢事件发生时，从机溢出运行标志SLVURIF(QSPIx_STATUS[7])将置为1。同时QSPIx_SS进入无效状态。

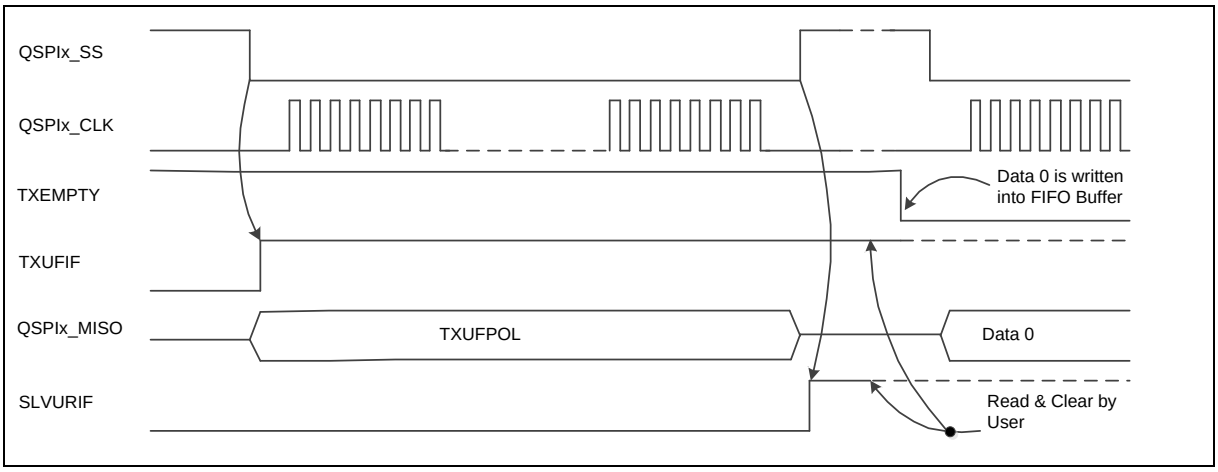


图 6.21-22 TX 下溢事件和从机溢出事件

在2位传输模式，在2个数据已经写入TX FIFO后，传送数据将被加载到移位寄存器。该模式同时使用两个移位寄存器和两个4级的斜移缓存。2位传送模式的详细时序，请参考2位传输模式的相关介绍。

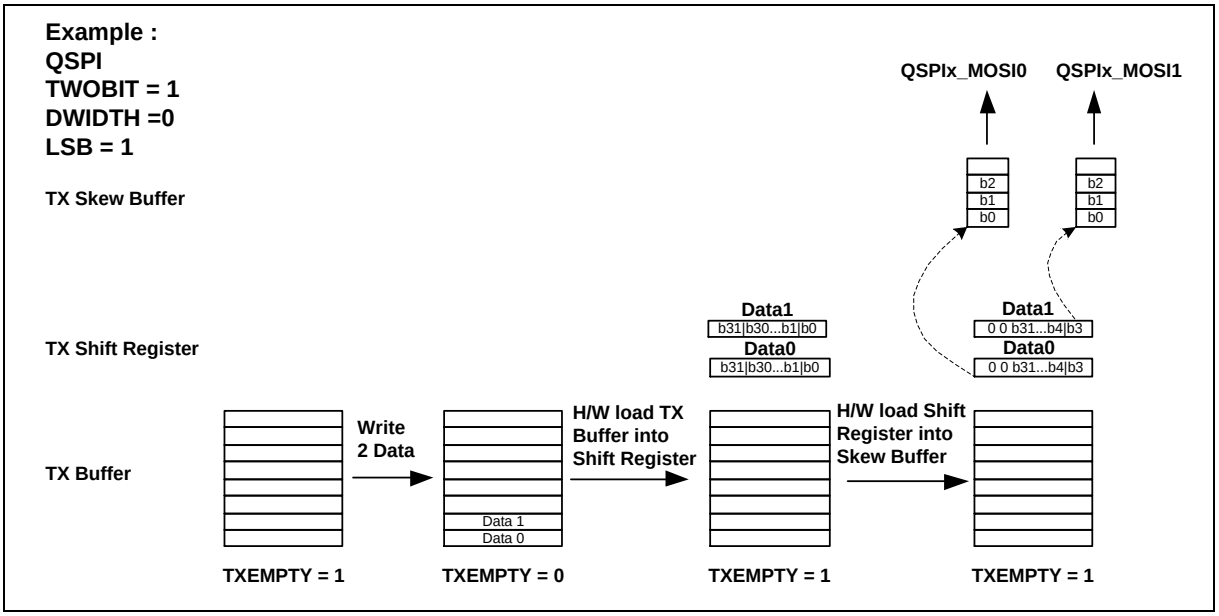


图 6.21-232 位传输模式 FIFO 缓存示例

在从机三线模式，如果在QSPI总线时钟出现前的3个外设时钟周期内，把数据写入发送FIFO缓存，则开始的2位数据是不可预知的(会保持上次传输最后一位的电平)。其它位则被TXUFPOL(QSPIx_FIFOCTL[6])保存。写入的数据将在下一次传输时发送。

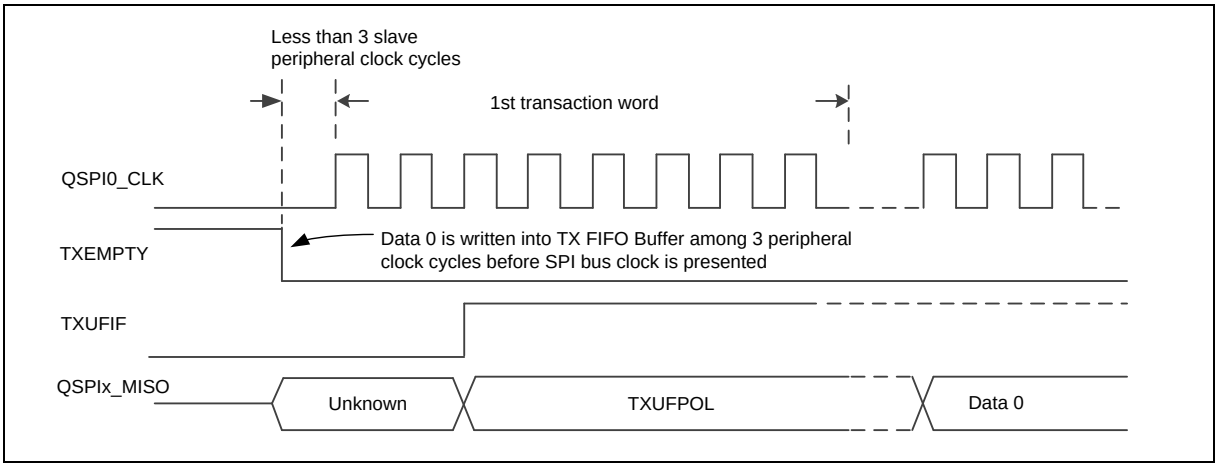


图 6.21-24 TX 下溢事件(QSPI0 从机三线模式使能)

在从机模式接收操作中，串行数据从QSPIx_MOSI管脚接收并存储到QSPIx_RX寄存器。接收机制类似于主机模式的接收操作。如果接收FIFO缓存包含8个未读数据，RXFULL(QSPIx_STATUS[9])将被置为1，RXOVIF(QSPIx_STATUS[11])也会被置为1。如果在QSPIx_MOSI管脚上有更多串行数据要接收，接下来的数据将被丢掉(参看接收FIFO缓存的示例图)。当从机片选线进入无效状态时，如果接收到位数数据与DWIDTH(QSPIx_CTL[12:8])设定的不一致，SLVBEIF(QSPIx_STATUS[6])将设置为1。

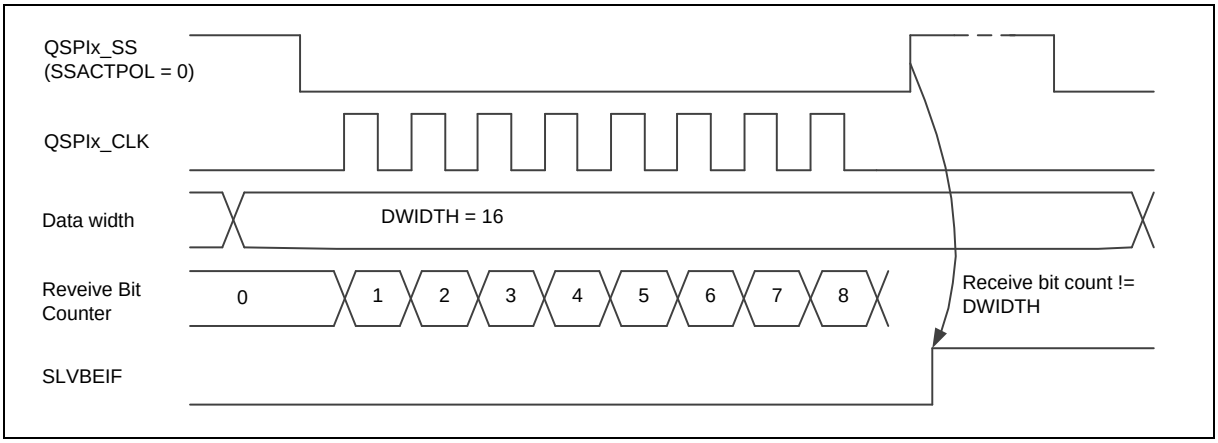


图 6.21-25 从机模式位计数错误

当从机选择信号有效且SLVTOCNT(QSPIx_SSCTL[31:16])的值不为0时，在串行时钟输入后，QSPI控制器逻辑的从机超时计数器开始计数。在一次事务完成后或者SLVTOCNT设置为0后，该计数器清除。如果超时计数器的值在该事务完成前大于或等于SLVTOCNT的值，从机超时事件发生，SLVTOIF(QSPIx_STATUS[5])将设置为1。

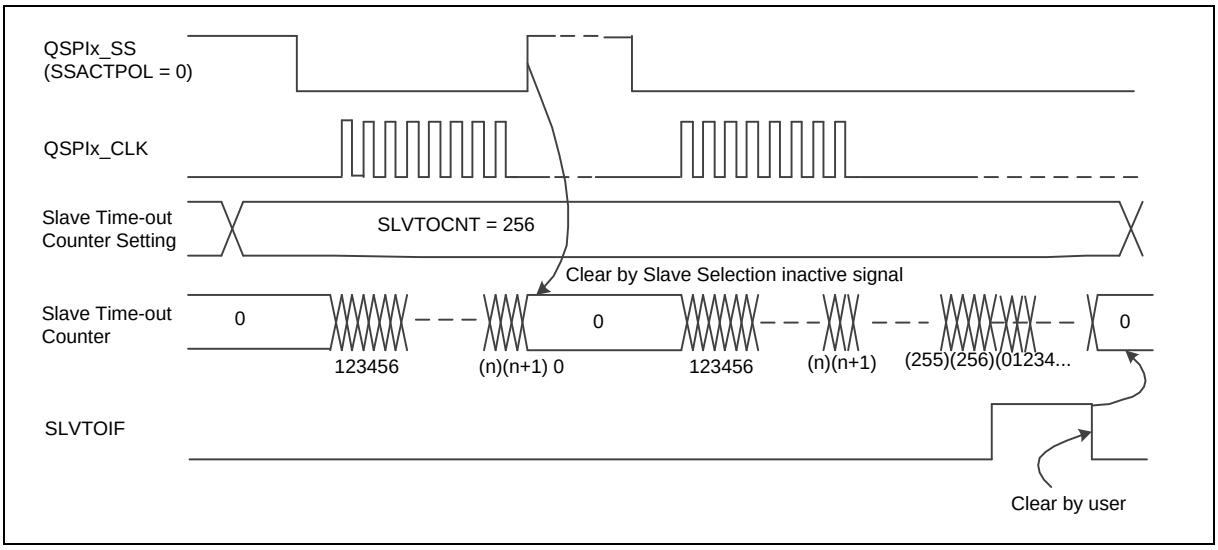


图 6.21-26 从机超时事件

控制器内置了一个接收超时功能模块。当接收FIFO非空，而且在主机模式时接收FIFO超过64个QSPI外设时钟周期或者从机模式时超过576个QSPI外设时钟周期没有读操作，则接收超时发生，RXTOIF(QSPIx_STATUS[12])置1。当接收FIFO有用户读取时，超时状态将自动清除。

6.21.5.13 中断

- QSPI单元传输中断

  当QSPI控制器完成一个单元传输，单元传输中断标志UNITIF(QSPIx_STATUS[1])将会被设为1。如果单元传输中断使能位UNITIEN(QSPIx_CTL[17])被置位，则单元传输中断事件将给CPU产生中断请求。单位传输中断标志位只能写1清零。

- QSPI从机片选有效/无效中断

从机模式下，如果SPIEN(QSPIx_CTL[0])和SLAVE(QSPIx_CTL[18])置1，从机片选信号进入有效/无效状态，从机片选有效/无效中断标志SSACTIF(QSPIx_STATUS[2])和SSINAIF(QSPIx_STATUS[3])会被置位。此时如果SSINAIEN(QSPIx_SSCTL[13])或SSACTIEN(QSPIx_SSCTL[12])为1，QSPI控制器会产生中断。

- 从机超时中断

  QSPI在从机模式时，用户可以通过从机超时功能识别有串行时钟输入但是超过了SLVTOCNT(QSPIx_SSCTL[31:16])中定义的从机外设时钟周期个数，传输事务还没有完成。

  当从机片选信号有效时，而且SLVTOCNT(QSPIx_SSCTL[31:16])的值不为0，在串行时钟输入后，QSPI控制器逻辑的从机超时计数器开始计数。在一次事务完成后或者SLVTOCNT(QSPIx_SSCTL[31:16])被置为0，该计数器将被清除。如果在一次事务传输完前，超时计数器的值大于或等于SLVTOCNT(QSPIx_SSCTL[31:16])的值，则从机超时事件发生，同时SLVTOIF(QSPIx_STATUS[5])将设置为1。如果SLVTOIEN(QSPIx_SSCTL[5])为1，SPI控制器将发生一个中断。

- 从机位计数错误中断

  在从机模式，当从机片选信号线进入无效状态时，如果发送或接收到位数据个数与DWIDTH(QSPIx_CTL[12:8])设置的不一致，SLVBEIF(QSPIx_STATUS[6])将被设置为1。未传输完成的数据将在TX和RX移位寄存器中弃除。如果SLVBEIEN(QSPIx_SSCTL[8])位为1，QSPI控制器将发生一个中断。

  注:如果从机片选信号激活，但是没有任何串行时钟输入，当从机片选信号进入无效状态时，SLVBEIF(QSPIx_STATUS[6])也将设置为1。

- TX下溢中断

  在QSPI从机模式，如果没有任何数据写入QSPIx_TX寄存器，当从机选择信号激活时，TXUFIF(QSPIx_STATUS[19])将设置为1。如果TXUFIEN(QSPIx_FIFOCTL[7])为1，QSPI控制器将发生发送下溢中断。

  注:QSPI从机模式发生下溢事件后，有两种方法可以使其恢复空闲状态并进入下个传输：(1)置TXRST为1(2)从机片选信号切为无效状态。

- 从机TX下溢运行中断

  当QSPIx_SS进入无效状态时，如果有TX下溢事件发生，SLVURIF(QSPIx_STATUS[7])将被设置为1。如果SLVURIEN(QSPIx_SSCTL[9])为1，QSPI控制器将发生发送溢出运行中断。

  注意:在从机三线模式，从机片选信号被认为是一直有效的，用户需要去查看TXUFIF(QSPIx_STATUS[19])位来确定是否有发生TX下溢事件。

- 接收溢出中断

  在从机模式，如果接收FIFO缓存已有8个未读数据，RXFULL(QSPIx_STATUS[9])和RXOVIF(QSPIx_STATUS[11])标志将会被设置为1。如果从QSPI总线上接收到更多串行数据，多余的数据将会丢失。如果RXOVIEN(QSPIx_FIFOCTL[5])为1，QSPI控制器将发生接收溢出中断。

- 接收FIFO超时中断

  如果在FIFO里有一个接收到的数据，在主机模式下用户超过64个QSPI外设时钟周期没有去读取，或者从机模式下超过576个QSPI外设时钟周期没有去读取，如果接收超时中断使

能位RXTOIEN(QSPIx_FIFOCTL[4])有设置为1，则会向系统发出一个RX超时中断。

- 发送FIFO中断

  在FIFO模式，如果发送FIFO缓存的有效数据少于或等于TXTH(QSPIx_FIFOCTL[30:28])的设定值，发送FIFO中断标志TXTHIF(QSPIx_STATUS[18])会被置1。如果发送FIFO中断位TXTHIEN(QSPIx_FIFOCTL[3])为1，则QSPI控制器会向系统产生一个发送FIFO中断。

- 接收FIFO中断

  在FIFO模式，如果接收FIFO缓存的有效数据大于RXTH(QSPIx_FIFOCTL[26:24])的设定值，接收FIFO中断标志RXTHIF(QSPIx_STATUS[10])会被设置为1。如果接收FIFO中断位RXTHIEN(QSPIx_FIFOCTL[2])有设置为1，QSPI控制器将会向系统产生一个接收FIFO中断。

6.21.6 时序图

从机选择信号的有效状态可以由SSACTPOL(QSPIx_SSCTL[2])配置。串行时钟的空闲状态可以通过CLKPOL(QSPIx_CTL[3])配置为高电平或低电平。传输字段长度在DWIDTH(QSPIx_CTL[12:8])中定义，发送/接收数据是以MSB还是LSB优先由LSB位(QSPIx_CTL[13])定义。用户也可以通过TXNEG/RXNEG(QSPIx_CTL[2:1])来选择发送/接收数据时串行时钟的边沿。四种QSPI主机/从机操作时序图以及相关设置如下图：

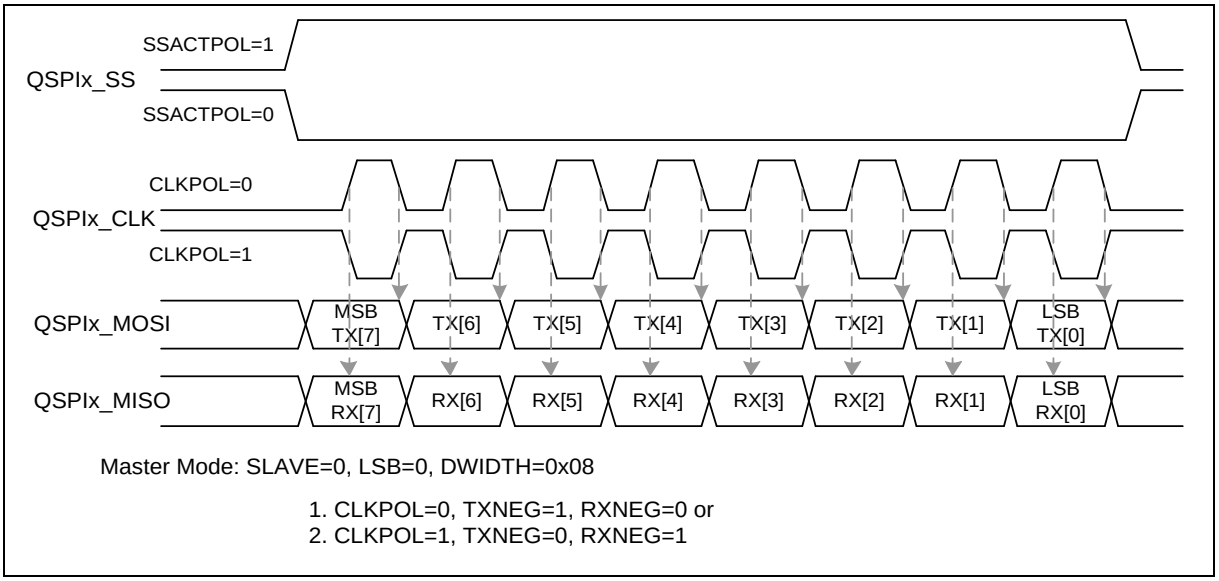


图 6.21-27QSPI 主机模式时序

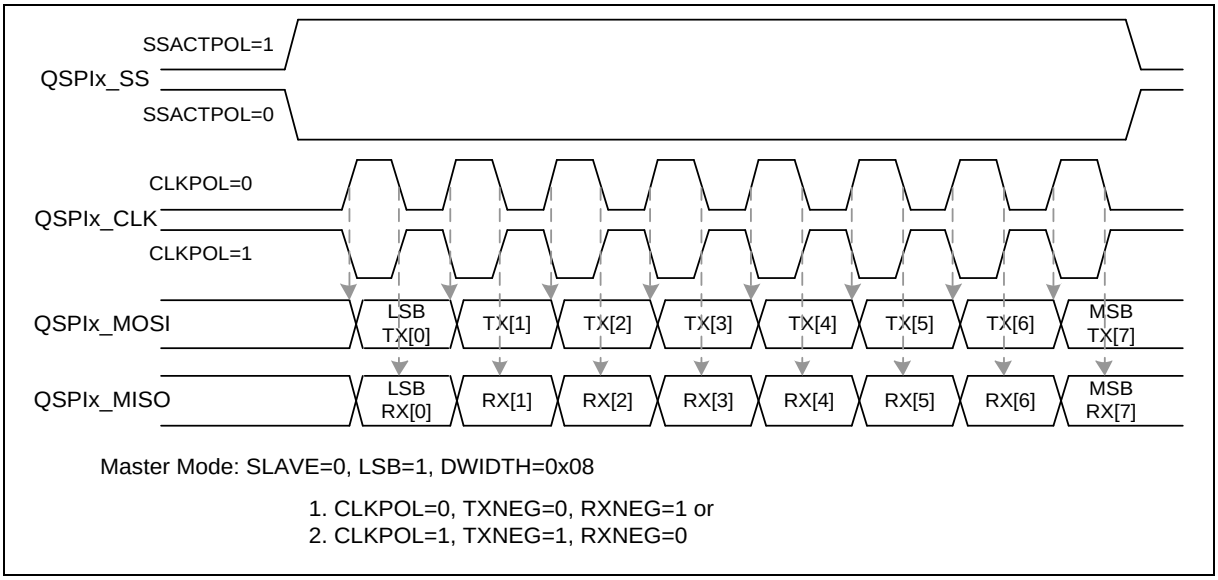


图 6.21-28QSPI 主机模式时序(QSPIx_CLK 交替相位)

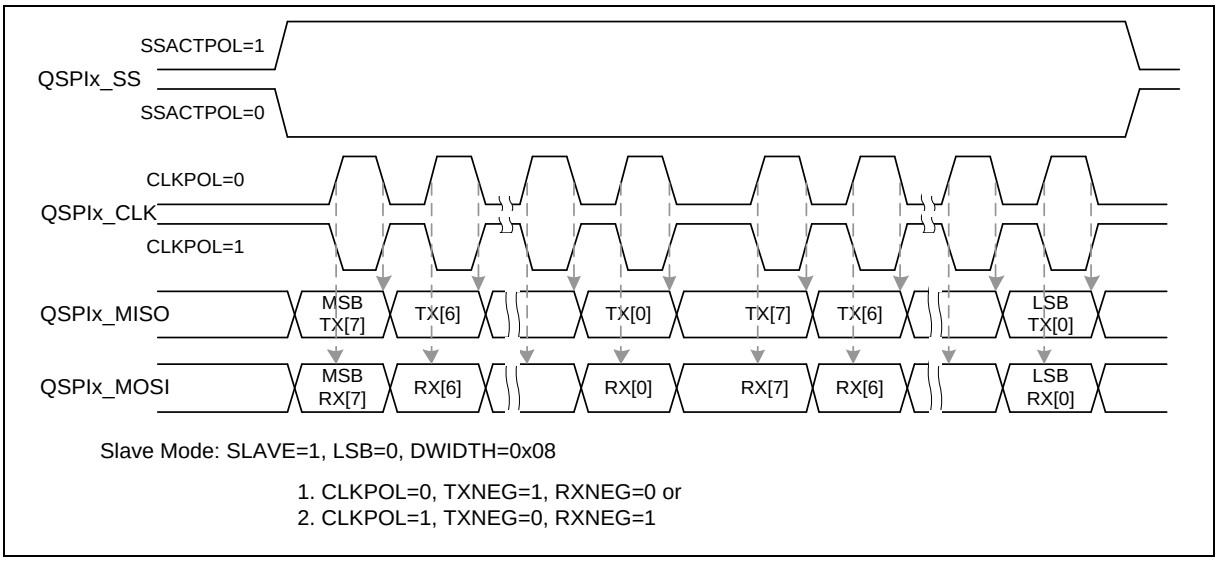


图 6.21-29QSPI 从机模式时序

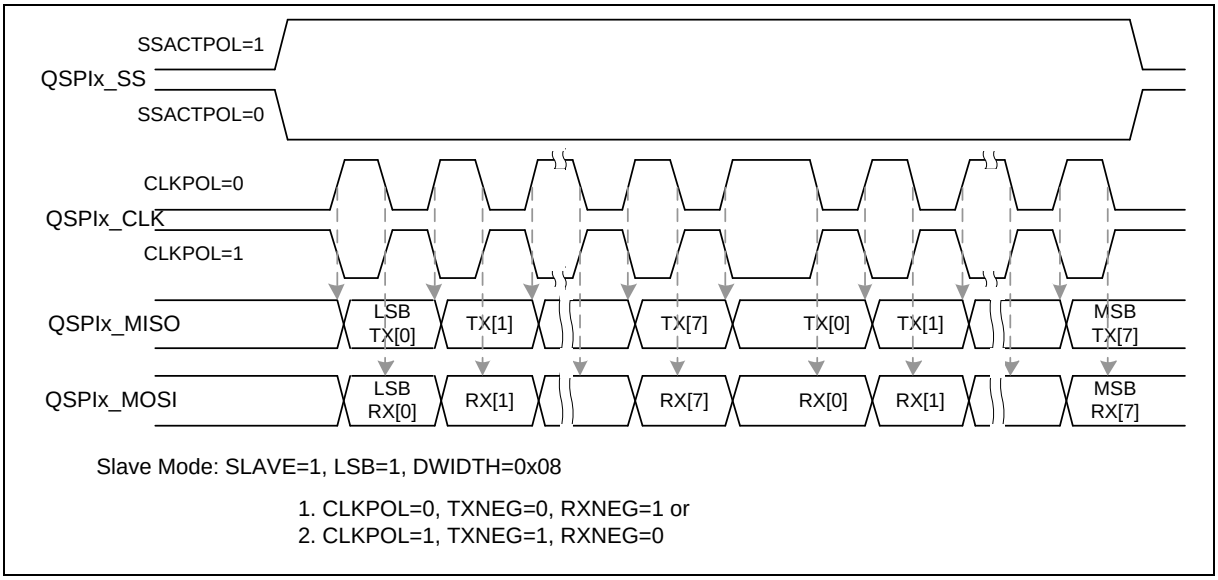


图 6.21-30 从机模式时序(QSPIx_CLK 交替相位)

6.21.7 编程示例

例 1:

QSPI配置为全双工主机时序如下：

- QSPI时钟上沿锁存数据
- QSPI时钟下沿发送数据
- MSB优先
- QSPI时钟空闲时处于低电平
- 每次只收发一个字节
- 片选信号低电平有效。

操作流程如下:

1. 在寄存器DIVIDER(SPIx_CLKDIV[8:0])配置SPI输出时钟频率
2. 写寄存器QSPIx_SSCTL配置QSPI主机相关模式:

   1)清0AUTOSS(QSPIx_SSCTL[3])关闭自动从机选择功能。

   2)清0 SSACTPOL(QSPIx_SSCTL[2])设置片选信号低电平有效。

   3)置SS(QSPIx_SSCTL[0])为1使能从机片选信号，激活片外从机设备。
3. 写寄存器QSPIx_CTL控制QSPI主机行为：

   1)设SLAVE(QSPIx_CTL[18])为0配置QSPI为主机设备。

   2)清0CLKPOL(QSPIx_CTL[3])强制QSPI空闲状态为低电平。

   3)置TXNEG(QSPIx_CTL[2])为1配置QSPI总线时钟下沿传输数据。

   4)清0RXNEG(QSPIx_CTL[1])配置QSPI总线时钟上沿锁存数据。

5)设置传输长度DWIDTH为8位(QSPIx_CTL[12:8]=0x08)。

6)清0LSB(QSPIx_CTL[13])配置MSB传输优先。

4. 置SPIEN(QSPIx_CTL[0])位1使能数据传输。
5. QSPI主机想要写一个字节的数据到片外设备时，将数据写到QSPIx_TX寄存器即可。
6. 等待QSPI中断发生(如果UNITIEN(QSPIx_CTL[17])为1)，或者轮询单元传输中断标志UNITIF(QSPIx_STATUS[1])。
7. 从QSPIx_RX读接收到的数据。
8. 重复步骤5继续其他数据的传输，或者配置SS(QSPIx_SSCTL[0])为0让片选信号无效以关闭片外从机设备。.

例2:

QSPI为全双工从机。片外主机按下面的规则通讯：

- QSPI时钟上沿锁存数据
- QSPI时钟下沿发送数据
- LSB优先
- QSPI时钟空闲时处于高电平
- 每次只收发一个字节
- 片选信号高电平有效。

操作流程如下:

1. 写寄存器QSPIx_SSCTL配置为从机模式.
2. 置SSACTPOL(QSPIx_SSCTL[2])为1配置片选信号高电平有效。.
3. 写QSPIx_CTL配置从机行为
   1) 设SLAVE(QSPIx_CTL[18])为1配置QSPI控制球为从机设备。
   2) 置CLKPOL(QSPIx_CTL[3])为1选择QSPI空闲状态为高电平。
   3) 置TXNEG(QSPIx_CTL[2])为1配置QSPI总线时钟下降沿传输数据。
   4) 清0RXNEG(QSPIx_CTL[1])配置QSPI总线时钟上升沿锁存数据。
   5) 设置传输长度DWIDTH为8位(QSPIx_CTL[12:8]=0x08)。
4. 置LSB(QSPIx_CTL[13])为1配置LSB传输优先
5. 置SPIEN(QSPIx_CTL[0])为1。等待片外主机设备的片选信号和QSPI时钟输入来启动数据传输
6. 如果QSPI从机想要发送一个字节的数据到片外主机，写数据到QSPIx_TX寄存器即可。
7. 如果QSPI从机想要从片外主机接收一个字的数据，且用户不关心什么数据被传输，则软件软件不需要去更新QSPIx_TX寄存器
8. 等待QSPI中等待QSPI中断发生(如果UNITIEN(QSPIx_CTL[17])为1)，或者轮询单元传输中断标志UNITIF(QSPIx_STATUS[1])。
9. 从QSPIx_RX寄存器读接收到的数据。
10. 回到7继续下个字节发送或停止发送。

6.21.8 寄存器映射

R: 只读，W: 只写，R/W: 读/写

寄存器	偏移量	R/W	描述	复位值
QSPI基地址: **QSPIx_BA = 0x4006_0000 + (0x0000_9000 *x)** **x = 0, 1**				
QSPIx_CTL	QSPIx_BA+0x00	R/W	QSPI控制寄存器	0x0000_0034
QSPIx_CLKDIV	QSPIx_BA+0x04	R/W	QSPI时钟分频寄存器	0x0000_0000
QSPIx_SSCTL	QSPIx_BA+0x08	R/W	QSPI从机选择控制寄存器	0x0000_0000
QSPIx_PDMACTL	QSPIx_BA+0x0C	R/W	QSPI PDMA控制寄存器	0x0000_0000
QSPIx_FIFOCTL	QSPIx_BA+0x10	R/W	QSPI FIFO控制寄存器	0x4400_0000
QSPIx_STATUS	QSPIx_BA+0x14	R/W	QSPI状态寄存器	0x0005_0110
QSPIx_TX	QSPIx_BA+0x20	W	QSPI数据发送寄存器	0x0000_0000
QSPIx_RX	QSPIx_BA+0x30	R	QSPI 数据接收寄存器	0x0000_0000

6.21.9 寄存器描述

QSPI控制寄存器(QSPIx_CTL)

寄存器	偏移量	R/W	描述	复位值
QSPIx_CTL	QSPIx_BA+0x00	R/W	QSPI控制寄存器	0x0000_0034

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
TXDTR_EN	QUADIOEN	DUALIOEN	DATDIR	REORDER	SLAVE	UNITIEN	TWOBIT
15	14	13	12	11	10	9	8
RXONLY	HALFDPX	LSB	DWIDTH				
7	6	5	4	3	2	1	0
SUSPITV				CLKPOL	TXNEG	RXNEG	SPIEN

位	描述	
[31:24]	**Reserved**	保留.
[23]	**TXDTR_EN**	**TX DTR(传输双倍发送率)模式使能位(主机有效)** 0=TX DTR模式关闭 1=TX DTR模式使能 **注:**QSPI主机模式支持TXDTR模式，从机模式不支持
[22]	**QUADIOEN**	**四I/O模式使能位** 0=四I/O模式关闭 1=四I/O模式使能
[21]	**DUALIOEN**	**双I/O模式使能位** 0=双I/O模式关闭 1=双I/O模式使能
[20]	**DATDIR**	**数据方向控制** 该位用来选则半双工模式和双/四I/O模式下数据输入/输出方向 0=QSPI数据输入 1=QSPI数据输出
[19]	**REORDER**	**字节重排功能使能位** 0=字节重排功能禁止 1=字节重排功能使能.每两个字节之间会被插入一个休眠间隔，休眠间隔的时间取决于SUSPITV的设置 **注:**仅在DWIDTH为16，24和32位时，该功能有效。
[18]	**SLAVE**	**从机模式控制**

		0=主机模式. 1=从机模式
[17]	**UNITIEN**	**单元传输中断使能位** 0=QSPI单元传输中断禁止 1=QSPI单元传输中断使能
[16]	**TWOBIT**	**2位传输模式使能位** 0=2位传输模式关闭 1=2位传输模式使能 **注:**当2-位传输模式使能时，第一个串行传输位数据来自第一个FIFO缓存数据，第二个串行传输位数据来自第二个FIFO缓存数据。同样，第一个接收位数据存储到第一个FIFO缓存，第二个接收位数据存放到第二个FIFO缓存。
[15]	**RXONLY**	**只接收模式使能位(仅主机)** 该只用于主机模式。在只接收模式，QSPI主机总线时钟会一直保持以接收数据，忙标志会一直置位。 0=只接收模式关闭 1=只接收模式使能
[14]	**HALFDPX**	**QSPI半双工传输使能位** 该位用来切换QSPI传输全双工模式和半双工模式。半双工模式下DATDIR(QSPIx_CTL[20])用来选择数据传输方向。 0=QSPI工作在全双工模式 1=QSPI工作在半双工模式
[13]	**LSB**	**LSB优先发送** 0=MSB,具体发送/接收寄存器的哪一位首先被发送/接收，取决于DWIDTH的设定值。 1=LSB,SPITX寄存器的低位，首先被发送到SPI数据输出管脚，从QSPI数据输入管脚上接收到的第一个数据位将会被放置到RX寄存器(QSPI_RX的位0)LSB的位置。
[12:8]	**DWIDTH**	**数据宽度** 该位用来标示每次传输时会有多少位被发送/接收。最低为8位，最高为32位。 DWIDTH=0x08….8位 DWIDTH=0x09….9位 …… DWIDTH=0x1F….31位 DWIDTH=0x00….32 位
[7:4]	**SUSPITV**	**休眠间隔 (仅主机)** 该四位用来配置在一次数据传输过程中连续两个发送/接收事务之间的休眠间隔。休眠间隔是从当前事务字的最后一个时钟边沿到接下来的事务的第一个边沿时钟。默认值是0x3.休眠间隔的周期可以根据下面公式获得： (SUSPITV[3:0]+0.5)*SPICLK时钟周期 例: SUSPITV=0x0….0.5SPICLK时钟周期 SUSPITV=0x1….1.5SPICLK时钟周期 …… SUSPITV=0xE….14.5SPICLK时钟周期 SUSPITV=0xF….15.5SPICLK时钟周期

[3]	**CLKPOL**	**时钟极性** 0=QSPI总线空闲时默认低 1=QSPI总线空闲时默认高
[2]	**TXNEG**	**下降沿发送** 0=在QSPI总线时钟的上升沿发送数据 1=在QSPI总线时钟的下降沿发送数据
[1]	**RXNEG**	**下降沿接收** 0=在QSPI总线时钟的上升沿锁存数据 1=在QSPI总线时钟的下降沿锁存数据
[0]	**SPIEN**	**QSPI传输控制使能位** 在主机模式下，FIFO缓存有数据时，该位设置为1后，开始传输。在从机模式下，该位置为1时，该设备已准备好接收数据。 0=禁止控制传输 1=使能控制传输 **注:**在更改SPIx_CTL,SPIx_CLKDIV,SPIx_SSCTL和SPIx_FIFOCTL寄存器的配置前，用户需清0SPIEN(SPIx_CTL[0])，并且确定SPIENSTS(SPI_STATUS[15])的值为0。

QSPI时钟分频寄存器(QSPIx_CLKDIV)

寄存器	偏移量	R/W	描述	复位值
QSPIx_CLKDIV	SPIx_BA+0x04	R/W	QSPI时钟分频寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							DIVIDER
7	6	5	4	3	2	1	0
DIVIDER							

位	描述	
[31:9]	**Reserved**	保留.
[8:0]	**DIVIDER**	**时钟除频** 该域的值是产生外设时钟，f_{spi_eclk}和QSPI主机的总线时钟的除频器的值。频率计算公式如下： $f_{spi_eclk} = \frac{f_{spi_clock_src}}{(DIVIDER+1)}$ 其中， $f_{spi_clock_src}$ 为外设时钟源，定义于时钟控制寄存器 CLK_CLKSEL2 中 **注:**在是否QSPI IP 软件复位和设置这个时钟除频寄存器之间的间隔时间必须大于或等于8个外设时钟周期。

注意:谨慎配置**DIVIDER**，QSPI时钟频率不能大于系统时钟频率

QSPI从机选择控制寄存器(QSPIx_SSCTL)

寄存器	偏移量	R/W	描述	复位值
QSPIx_SSCTL	SPIx_BA+0x08	R/W	SPI从机选择控制寄存器	0x0000_0000

31	30	29	28	27	26	25	24
SLVTOCNT							
23	22	21	20	19	18	17	16
SLVTOCNT							
15	14	13	12	11	10	9	8
Reserved		SSINAIEN	SSACTIEN	Reserved		SLVURIEN	SLVBEIEN
7	6	5	4	3	2	1	0
Reserved	SLVTORST	SLVTOIEN	SLV3WIRE	AUTOSS	SSACTPOL	Reserved	SS

位	描述	
[31:16]	SLVTOCNT	**从机模式超时周期** 在从机模式，当在从机片选信号有效期间有总线时钟输入时，这些位表示超时周期。超时计数器的时钟源是从机外设时钟。如果该位值为 0，表示禁止从机模式超时功能。
[15:14]	Reserved	保留
[13]	SSINAIEN	**从机片选无效中断使能位** 0=从机片选无效中断禁止 1=从机片选无效中断使能
[12]	SSACTIEN	**从机片选有效中断使能位** 0=从机片选有效中断禁止 1=从机片选有效中断使能
[11:10]	Reserved	保留
[9]	SLVURIEN	**从机模式TX下溢中断使能位** 0=从机模式TX下溢中断禁止 1=从机模式TX下溢中断使能
[8]	SLVBEIEN	**从机模式位计数错误中断使能位** 0=从机模式位计数错误中断禁止 1=从机模式位计数错误中断使能
[7]	Reserved	保留
[6]	SLVTORST	**从机模式超时复位控制** 0=从机模式超时事件发生时，TX和RX控制电路不会复位 1=从机模式超时事件发生时，TX和RX控制电路由硬件复位
[5]	SLVTOIEN	**从机模式超时中断使能位**

		0=从机模式超时中断禁止 1=从机模式超时中断使能
[4]	**SLV3WIRE**	**从机三线模式使能位** 只有QSPI支持从机三线模式。从机三线模式中，QSPI控制器可以工作在三线接口，QSPIx_CLK,QSPIx_MISO和SPIx_MOSI。 0=四线双向接口 1=三线双向接口
[3]	**AUTOSS**	**自动从机片选功能使能位(仅主机)** 0=自动从机片选功能禁止，从机片选信号由SS(QSPIx_SSCTL[0])控制 1=自动从机片选功能使能
[2]	**SSACTPOL**	**从机片选有效极性** 该位定义从机片选信号(QSPIx_SS)有效的极性 0=片选信号QSPIx_SS低电平有效 1=片选信号QSPIx_SS高电平有效
[1]	**Reserved**	保留
[0]	**SS**	**从机片选控制位(仅主机)** 如果AUTOSS为0, 0=设置SPIx_SS为无效状态 1=设置SPIx_SS为有效状态 如果AUTOSS置1, 0=保持SPIx_SS为无效状态 1=进行数据传输时，SPIx_SS会自动切换到有效状态；无数据传输时则会进入无效状态。SPIx_SS的有效状态定义于SSACTPOL(SPIx_SSCTL[2])。

QSPI PDMA控制寄存器(QSPIx_PDMACTL)

寄存器	偏移量	R/W	描述	复位值
QSPIx_PDMACTL	SPIx_BA+0x0C	R/W	QSPI PDMA控制寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved					PDMARST	RXPDMAEN	TXPDMAEN

位	描述	
[31:3]	Reserved	保留
[2]	PDMARST	**PDMA复位** 0=无效 1=复位QSPI控制器的PDMA控制逻辑。该位自动清0。
[1]	RXPDMAEN	**PDMA接收使能位** 0=PDMA接收禁止 1=PDMA接收使能
[0]	TXPDMAEN	**PDMA发送使能位** 0=PDMA发送禁止 1=PDMA发送使能 **注:**在QSPI主机模式支持全双工传输，如果发送和接收PDMA功能都使能，接收PDMA功能不能在发送PDMA功能之前使能。用户可以先使能发送PDMA功能或者同时使能两个功能。

QSPI FIFO控制寄存器(QSPIx_FIFOCTL)

寄存器	偏移量	R/W	描述	复位值
QSPIx_FIFOCTL	QSPIx_BA+0x10	R/W	QSPIFIFO控制寄存器	0x2200_0000

31	30	29	28	27	26	25	24
Reserved	TXTH			Reserved	RXTH		
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved						TXFBCLR	RXFBCLR
7	6	5	4	3	2	1	0
TXUFIEN	TXUFPOL	RXOVIEN	RXTOIEN	TXTHIEN	RXTHIEN	TXRST	RXRST

位	描述	
[31]	**Reserved**	保留
[30:28]	**TXTH**	**发送FIFO阈值** 如果发送FIFO缓存的有效数据个数小于或者等于TXTH的设置，TXTHIF将会被设置为1，否则TXTHIF将会被设置为0。
[27]	**Reserved**	保留
[26:24]	**RXTH**	**接收FIFO域值** 如果接收FIFO缓存的有效数据个数大于RXTH的设置，RXTHIF将会被设置为1，否则RXTHIF将会被设置为0。
[23:10]	**Reserved**	保留
[9]	**TXFBCLR**	**清发送FIFO缓存** 0=无效 1=清除发送FIFO指针，TXFULL位将清除为0而且TXEMPTY位也将设置为1。该位置1后，在一个系统时钟后，由硬件清0。 **注:**TX移位寄存器不会清空
[8]	**RXFBCLR**	**清接收FIFO缓存** 0=无效 1=清除接收FIFO指针，RXFULL位将清除为0而且RXEMPTY位将设置为1。该位设置为1后，在一个系统时钟后，由硬件清0。 **注:**RX移位寄存器不会清空
[7]	**TXUFIEN**	**TX下溢中断使能位** 从机模式下TX下溢中断事件发生时，TXUFIF(QSPIx_STATUS[19])会被置1。该位用来使能TX下溢中断。 0=从机TX下溢中断禁止

		1=从机TX下溢中断使能
[6]	**TXUFPOL**	**TX下溢数据极性** 0=在从机模式，如果有发送下溢事件发生，QSPI数据输出保持为0。 1=在从机模式，如果有发送下溢事件发生，QSPI数据输出保持为1。 **注:** 1.从机片选信号有效时如果没有任何数据发送则会发生下溢事件。 2.TX下溢事件发生时，QSPIx_MISO管脚状态将由该位决定而不管TXFIFO是否为空。存储在TX FIFO的数据会在下一个传输帧通过QSPIx_MISO发送出去。
[5]	**RXOVIEN**	**接收FIFO溢出中断使能位** 0=接收FIFO溢出中断禁止 1=接收FIFO溢出中断使能
[4]	**RXTOIEN**	**从机接收超时中断使能位** 0=从机接收超时中断禁止 1=从机接收超时中断使能
[3]	**TXTHIEN**	**发送FIFO阈值中断使能位** 0= TX FIFO阈值中断禁止 1=TX FIFO阈值中断使能
[2]	**RXTHIEN**	**接收FIFO阈值中断使能位** 0=RX FIFO阈值中断禁止 1=RX FIFO阈值中断使能
[1]	**TXRST**	**发送复位** 0=无效 1=复位发送FIFO指针和发送电路。TXFULL位将被清0，TXEMPTY位将被置1。该位置1后，大约经过3个系统时钟周期和2个外设时钟周期，硬件会将该位清除为0。如果用户想确认复位是否已经完成，可以读取TXRXRST(QSPIx_STATUS[23])来确认。 **注:**如果从机模式下发生TX下溢事件发生，该位也可以用来使QSPI返回空闲状态。
[0]	**RXRST**	**接收复位** 0=无效 1=复位接收FIFO指针和接收电路。RXFULL位将被清除为0，RXEMPTY位将被设置为1。该位置1后，大约经过3个系统时钟周期和2个外设时钟周期，硬件将该位清除为0。如果用户想确认复位是否已经完成，可以读TXRXRST(SPI_STATUS[23])来确认。

QSPI状态寄存器(QSPIx_STATUS)

寄存器	偏移量	R/W	描述	复位值
QSPIx_STATUS	QSPIx_BA+0x14	R/W	QSPI状态寄存器	0x0005_0110

31	30	29	28	27	26	25	24
TXCNT				RXCNT			
23	22	21	20	19	18	17	16
TXRXRST	Reserved			TXUFIF	TXTHIF	TXFULL	TXEMPTY
15	14	13	12	11	10	9	8
SPIENSTS	Reserved		RXTOIF	RXOVIF	RXTHIF	RXFULL	RXEMPTY
7	6	5	4	3	2	1	0
SLVURIF	SLVBEIF	SLVTOIF	SSLINE	SSINAIF	SSACTIF	UNITIF	BUSY

位	描述	
[31:28]	**TXCNT**	**发送FIFO数据个数(只读)** 该位域表明发送FIFO缓存的有效数据个数。
[27:24]	**RXCNT**	**接收FIFO数据个数(只读)** 该域表明接收FIFO缓存有效数据个数..
[23]	**TXRXRST**	**TX或RX复位状态(只读)** 0=TXRST或RXRST的复位功能已经完成 1=TXRST或RXRST正在复位 **注意:**TXRST和RXRST的复位操作都需要3个系统时钟周期+2个外设时钟周期。用户可以检查该状态位来检测复位功能是否完成。
[22:20]	**Reserved**	保留
[19]	**TXUFIF**	**TX下溢中断标志** 当发送下溢事件发生时，该位被设置为1，数据输出管脚的状态取决于TXUFPOL的设置。 0=无影响 1=当从机片选信号有效时，发送FIFO和发送移位寄存器中没有数据。 **注1**:该位写1清0。 **注2**:当从机片选信号有效时，如果复位从机的发送电路，在3个系统时钟周期和2个外设时钟周期后，此时复位操作已经完成，该位将被设置为1。
[18]	**TXTHIF**	**发送FIFO阈值中断标志(只读)** 0=发送FIFO缓存中的有效数据个数大于TXTH设置的值 1=发送FIFO缓存中的有效数据个数少于或等于TXTH设置的值
[17]	**TXFULL**	**发送FIFO缓存满标志(只读)** 0=发送FIFO缓存未满 1=发送FIFO缓存已满

[16]	**TXEMPTY**	**发送FIFO缓存空标志(只读)** 0=发送FIFO缓存不空 1=发送FIFO缓存已空
[15]	**SPIENSTS**	**QSPI使能状态位(只读)** 0=禁止QSPI控制器. 1=使能QSPI控制器. **注意:**QSPI外设时钟与系统时钟不同步。为了确保QSPI控制器已经被禁止，该位指示了QSPI控制器的真实状态。
[14:13]	**Reserved**	保留
[12]	**RXTOIF**	**接收超时中断标志** 0=没有接收FIFO超时事件 1=接收FIFO缓存非空且主机模式下，超过64个QSPI时钟周期或从机模式下超过576个QSPI引擎时钟周期，接收FIFO缓存上没有读操作。当接收FIFO缓存被软件读取，超时状态会自动清0。 **注:**该位写1清0
[11]	**RXOVIF**	**接收FIFO溢出中断标志** 当接收FIFO缓存已经满了，接下来的数据将被丢弃，同时该位被设置为1. 0=接收FIFO没有溢出 1=接收FIFO溢出 **注:**该位写1清0
[10]	**RXTHIF**	**接收FIFO阀值中断标志(只读)** 0=接收FIFO缓存的有效数据个数少于或等于RXTH设置的值 1=接收FIFO缓存的有效数据个数大于RXTH设置的值
[9]	**RXFULL**	**接收FIFO缓存满标志(只读)** 0=接收FIFO缓存未满 1=接收FIFO缓存已满
[8]	**RXEMPTY**	**接收FIFO缓存空标志(只读)** 0=接收FIFO缓存不空 1=接收FIFO缓存为空
[7]	**SLVURIF**	**从机模式发送下溢中断标志** 在从机模式，如果发送下溢事件发生，而且从机片选线进入无效状态，该中断标志将被设置为1。 0=未发生从机发送下溢事件 1=发生从机发送下溢事件 **注:**该位写1清0
[6]	**SLVBEIF**	**从机模式位计数错误中断标志** 在从机模式，当从机片选信号线进入无效状态时，如果位计数与DWIDTH不一致，该中断标志将被设置为1。 0=未发生从机模式位计数错误事件 1=发生从机模式位计数错误事件 **注:**如果从机片选激活，但是没有总线时钟输入，当从机片选进入无效状态时，SLVBEIF也将被设置为1.该位写1清0。

[5]	**SLVTOIF**	**从机超时中断标志** 当从机片选信号激活和SLVTOCNT的值不为0时，在检测到总线时钟时，QSPI控制器逻辑的从机超时计数器开始计数。在事务完成之前，如果超时计数器的值大于或等于SLVTOCNT(QSPIx_SSCTL[31:16])里的值。将发生从机超时中断事件。 0=从机超时未发生 1=发生从机超时 **注:**该位写1清0
[4]	**SSLINE**	**从机片选线总线状态(只读)** 0=从机片选线状态为0 1=从机片选线状态为1 **注:**该位只在从机模式有效。如果SSACTPOL(QSPIx_SSCTL[2])设置为0，而且SSLINE的值为1，QSPI从机片选处于无效状态。
[3]	**SSINAIF**	**从机片选无效中断标志** 0=从机片选无效中断被清除或没有发生 1=发生了从机片选无效中断 **注:**该位只在从机模式有效，该位写1清0。
[2]	**SSACTIF**	**从机片选激活中断标志** 0=从机片选激活中断被清除或未发生 1=发生了从机片选激活中断 **注:**该位只在从机模式有效，该位写1清0。
[1]	**UNITIF**	**单元传输中断标志** 0=该位清除后，没有事务完成 1=QSPI控制器已完成一个单元传输 **注:**该位写1清0。
[0]	**BUSY**	**忙状态(只读)** 0=QSPI控制器处于空闲状态 1=QSPI控制器处于忙状态. 如下是忙条件: a.QSPIx_CTL[0]=1且TXEMPTY=0。 b.对于QSPI主机模式，QSPIx_CTL[0]=1且TXEMPTY=1但是当前传输尚未完成。 c.对于QSPI主机模式，QSPIx_CTL[0]=1和RXONLY=1。 d.对于QSPI从机模式，SPIx_CTL[0]=1，从机选择有效时仍有串口时钟输入。 e.对于QSPI从机模式，SPIx_CTL[0]=1，从机选择无效时发送缓存或发送移位寄存器仍非空。

QSPI数据发送寄存器(QSPIx_TX)

寄存器	偏移量	R/W	描述	复位值
QSPIx_TX	QSPIx_BA+0x20	W	QSPI数据发送寄存器	0x0000_0000

31	30	29	28	27	26	25	24
TX							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
TX							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
TX							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
TX							

位	描述	
[31:0]	**TX**	**数据发送寄存器** 数据发送寄存器会把要发送的数据传入4级的发送FIFO缓存。该寄存器的有效位数在QSPI模式下取决于DWIDTH(QSPIx_CTL[12:8])，QSPI模式下，如果DWIDTH为0x08，TX[7:0]位会被发送。如果DWIDTH为0x00，QSPI控制器将发送32位。 **注:**主机模式下，QSPI控制器写该寄存器后，1个APB时钟加6个外设时钟后才会开始QSPI总线时钟的传输。

QSPI数据接收寄存器(QSPIx_RX)

寄存器	偏移量	R/W	描述	复位值
QSPIx_RX	QSPIx_BA+0x30	R	QSPI数据接收寄存器	0x0000_0000

31	30	29	28	27	26	25	24
RX							
23	22	21	20	19	18	17	16
RX							
15	14	13	12	11	10	9	8
RX							
7	6	5	4	3	2	1	0
RX							

位	描述	
[31:0]	RX	**数据接收寄存器** SPI控制器有8级FIFO缓存。从QSPI数据输入管脚接收到的数据会被保存到该寄存器。如果RXEMPTY(QSPIx_STATUS[8]不为1，软件可以通过读该寄存器获取FIFO缓存的数据。 该寄存器只读。

6.22 SPI同步串行接口控制器(SPI主机模式)

6.22.1 概述

SPI同步串行接口控制器在接收外围设备数据时执行串到并的转换，在收到MCU数据后执行并到串的转换。该控制器可以驱动一个外部设备（外部的SPI Flash），扮演SPI主机的角色。当数据传输完成，它可以产生中断信号，并且可以通过写1来清除中断标志。从机片选的有效电平根据外设可以选择低或高电平有效。写一个分频值到SPIM_CTL1寄存器设置输出到外围设备的串行输出时钟的频率。

SPIflash控制器普通I/O模式包含4个32位的发送/接收缓存，可以提供1到4个突发操作模式。每次传输的位数可以是8,16,24,或32位。一次传输中最大可以连续发送/接收四次数据。

通过DMA写模式，用户可以从SRAM传输数据到外部SPI flash器件。在DMA读模式下，可以将外部SPI flash器件数据传输到SRAM。在直接内存映射模式（DMM模式）下，在没有MCU设置相关SPI flash命令情况下，该SPI flash控制器会将AHB总线命令转换成SPI flash操作。因此用户可以将外部SPI flash当作一个ROM模块。

在关闭直接内存映射缓存的情况下，在控制器执行一个直接内存映射访问之后会预取4个字flash中的数据。在直接内存映射缓存开启模式下，它会使用32KB的缓存去减少访问外部SPI flash器件的次数，这样一来提高了SPI flash访问性能。在不增加串行时钟频率的情况下为了提高读SPI flash操作，这个SPI flash控制器支持DTR / DDR(双倍传输速率/双倍数据速率)读取命令指令，该指令支持标准/双/四数据SPI模式。第一字节的命令指令类似于其他所有的SPI命令被设备锁存在上升沿时钟。一旦DTR / DDR指令被设备接受，地址输入和数据输出将锁定在串行时钟的上升和下降的边沿。

在内核耦合存储器模式（CCM模式）下，缓存功能被硬件自动禁用，MCU可以把32KB的缓存当作一般的SRAM用。对于数据保护，在DMA 读/写模式和直接内存映射模式下，SPI flash 控制器支持密码的加密和解密电路来保护那些被用户放在外部SPI flash的数据。

6.22.2 特性

- 32MB的SPI flash
- 支持SPI主机模式
- 支持直接内存映射模式和普通I/O模式
- 普通I/O模式下支持8/16/24/32位传输
- 普通I/O模式下支持突发模式操作，一次传输中最大可以连续发送/接收四次数据
- 支持DMA下读/写
- 支持标准的1位、2位和4位I/O传输模式
- 支持DTR /DDR(双倍传输速率/双倍数据速率)传输模式
- 支持32KB缓存
- 在缓存禁用情况下支持32KB内核耦合存储器模式（CCM）
- 支持密码的加/解密
- 对于外部SPI flash器件有一根从机/设备选择线

6.22.3 框图

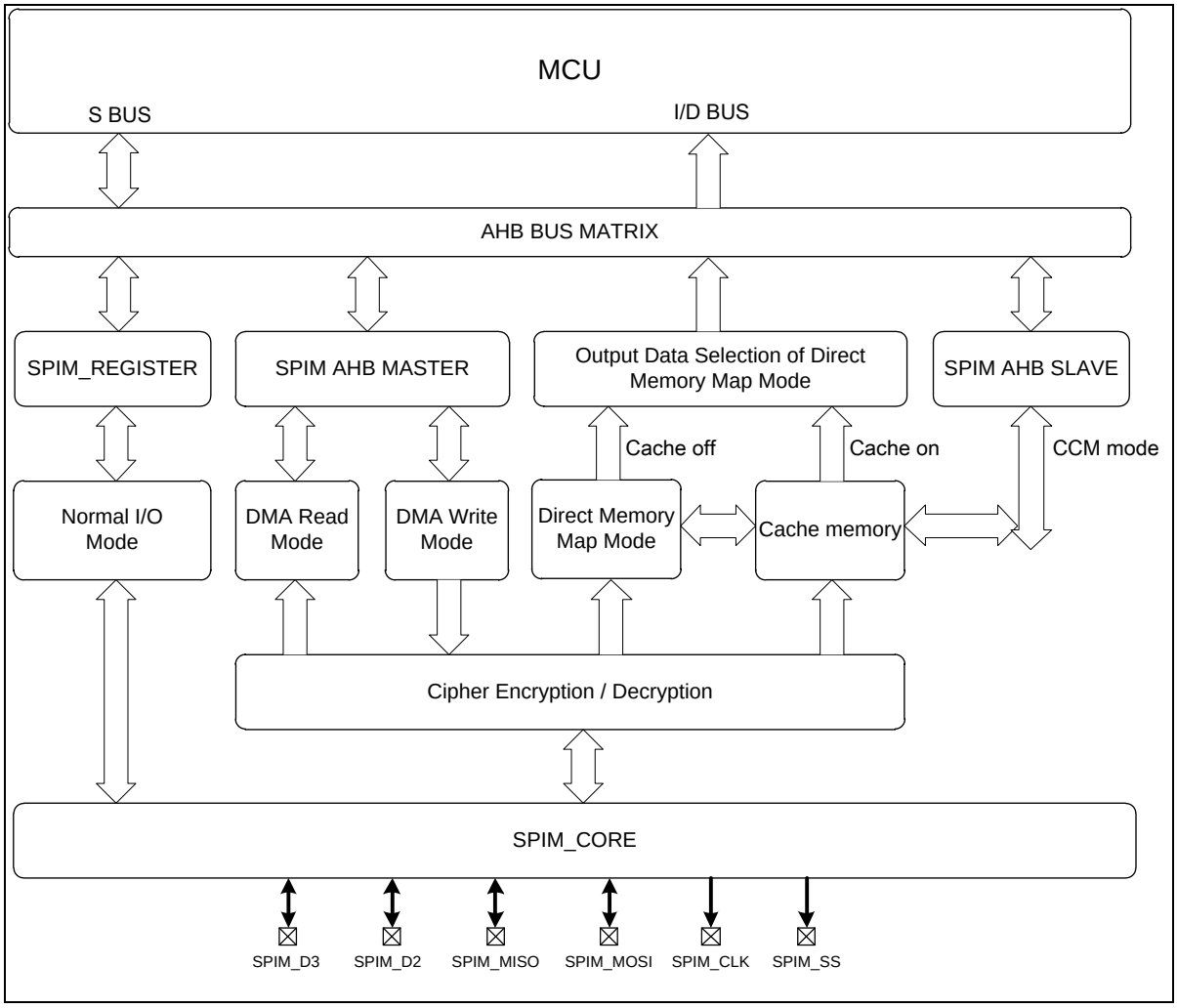


图 6.22-1SPIM 方框图

6.22.4 基本配置

- 时钟源配置
  - 使能SPIM外设时钟控制位是SPIMCKEN(CLK_AHBCLK[14])
- 复位配置
  - R复位SPIM控制器控制位是SPIMRST(SYS_IPRST0[14])
- 引脚配置

组	引脚名称	GPIO	MFP
SPIM	SPIM_CLK	PA.2	MFP2
		PC.2	MFP3

		PE.4, PG.12	MFP4
	SPIM_D2	PA.5	MFP2
		PC.5	MFP3
		PE.7, PG.9	MFP4
	SPIM_D3	PA.4	MFP2
		PC.4	MFP3
		PE.6, PG.10	MFP4
	SPIM_MISO	PA.1	MFP2
		PC.1	MFP3
		PE.3, PG.13	MFP4
	SPIM_MOSI	PA.0	MFP2
		PC.0	MFP3
		PE.2, PG.14	MFP4
	SPIM_SS	PA.3	MFP2
		PC.3	MFP3
		PE.5, PG.11	MFP4

6.22.5 功能描述

6.22.5.1 SPIM时序图

SPI传输时序图如下所示：

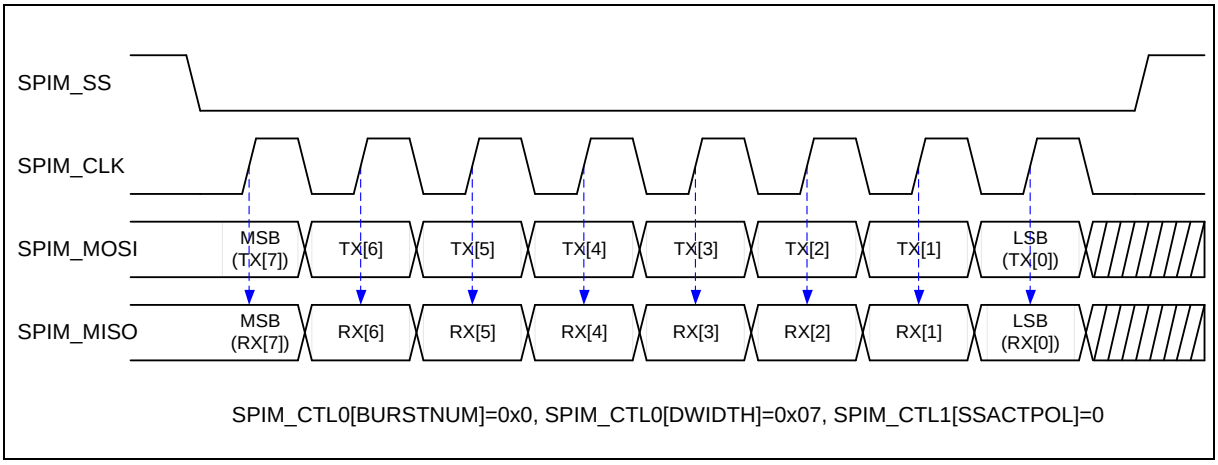


图 6.22-2SPIM 时序图

6.22.5.2 SPIM不带DMA的编程范例(普通 I/O模式)

依据如下的要求访问设备如图6.22-2:

- 数据高位先传

- 每次发送/接收只有一个字节
- 片选信号低有效
- 你应该做如下基本设置（具体步骤请参考设备的规格）：

1. 写一个分频值到DIVIDER(SPIM_CTL1[31:16])以确定串行时钟的频率。
2. 设置SSACTPOL(SPIM_CTL1[5])为0来激活你想要访问的设备。
3. 当发送(写)数据到设备：
   - QDIODIR(SPIM_CTL0[15])=1
   - BURSTNUM(SPIM_CTL0[14:13])=0x0
   - DWIDTH(SPIM_CTL0[12:8])=0x07
   - 写你要发送的数据到SPIM_TX0[7:0]
4. 当从设备接收（读）数据：
   - QDIODIR(SPIM_CTL0[15])=0
   - BURSTNUM(SPIM_CTL0[14:13])=0x0
   - DWIDTH(SPIM_CTL0[12:8])=0x07
5. 当从设备接收（读）数据
6. 等待中断或轮询SPIMEN(SPIM_CTL1[0])直到它变为0
7. 当接收（读）来自设备的数据：
   - 从SPIM_RX0[7:0]寄存器读出接收到的数据

6.22.5.3 SPIM带DMA的编程范例

如果用户想用DMA的功能访问设备，三个额外的寄存器需要被配置，包括SPIM_DMACNT, SPIM_SRAMADDR和SPIM_FADDR。DMA功能可以被用来支持加载启动代码（从外设读数据到系统内存）或从系统内存读数据保存到外设。用户必须定义源地址、长度和目的地址，然后硬件会自动搬运想要的数据长度到指定的目的地址。用户在SRAM和SPIflash之间使用DMA读/写操作之前，强烈建议向外部SPI flash发送禁用连续读模式和突发包模式的命令。

从SPI flash搬运数据到系统内存(DMA 读模式)

步骤 1:通过使用普通I/O模式检查设备ID确认设备已连接

步骤 2:

- 在SPIM_SRAMADDR寄存器设置目标内存地址
- 在SPIM_DMACNT寄存器设置代码的长度
- 在SPIM_FADDR寄存器设置 SPI Flash 的起始地址
- 设置OPMODE (SPIM_CTL0[23:22]) 为 DMA 读模式
- 设置 CMDCODE (SPIM_CTL0[31:24]) 为 0x03 命令码
- 设置SPIMEN (SPIM_CTL1[0])启动，然后 SPIM 会从SPIM_FADDR到SPIM_SRAMADDR搬运SPIM_DMACNT中设置的传输长度的代码块
- 等待中断或轮询SPIMEN (SPIM_CTL1[0])位直到它变为 0

如果使用的SPI Flash支持其他相关的读命令，用户也可以使用下面的读命令代码

- 快速读 (0Bh), 设置 CMDCODE (SPIM_CTL0[31:24])为 0x0B
- 快速读双线输出 (3Bh), 设置 CMDCODE (SPIM_CTL0[31:24]) 为0x3B
- 快速读双线I/O (BBh), 设置CMDCODE (SPIM_CTL0[31:24]) 为0xBB
- 快速读四线I/O (EBh), 设置CMDCODE (SPIM_CTL0[31:24]) 为0xEB
- 字读双线I/O (E7h), 设置CMDCODE (SPIM_CTL0[31:24]) 为0xE7
- DTR/DDR 快速读(0Dh), 设置CMDCODE (SPIM_CTL0[31:24]) 为0x0D
- DTR/DDR 快速读双线 I/O (BDh), 设置CMDCODE (SPIM_CTL0[31:24]) 为0xBD.
- DTR/DDR 快速读四线 I/O (EDh), 设置CMDCODE (SPIM_CTL0[31:24]) 为0xED。

6.22.5.4 从系统内存搬运数据到SPI flash (DMA 写模式)

步骤 1:写SPI Flash之前先擦除SPI Flash（使用普通I/O模式）

步骤 2:

- 发送写使能命令到 SPI Flash (使用普通I/O 模式).
- 在SPIM_SRAMADDR寄存器里设置源内存地址
- 在SPIM_DMACNT寄存器里设置传输数据个数
- 在SPIM_FADDR寄存器里设置 SPI Flash起始地址
- 设置OPMODE (SPIM_CTL0[23:22])为 DMA 写模式
- 设置CMDCODE (SPIM_CTL0[31:24])为 0x02 命令代码
- 设置SPIMEN (SPIM_CTL1[0])启动
- 等待中断或轮询SPIMEN (SPIM_CTL1[0])直到它变为 0.

如果使用的SPI Flash 支持其他相关的写命令，用户也可以使用下面命令代码（用户需要参考SPI Flash的规格来选择合适的四线写模式命令）

- 四线写, 设置CMDCODE (SPIM_CTL0[31:24]) 为0x32用于类型_1编程命令 (参考图 6.22-3).
- 四线写, 设置CMDCODE (SPIM_CTL0[31:24]) 为0x38用于类型_2编程命令 (参考图 6.22-4).
- 四线写, 设置CMDCODE (SPIM_CTL0[31:24]) 为0x40用于类型_3编程命令 (参考图 6.22-5).

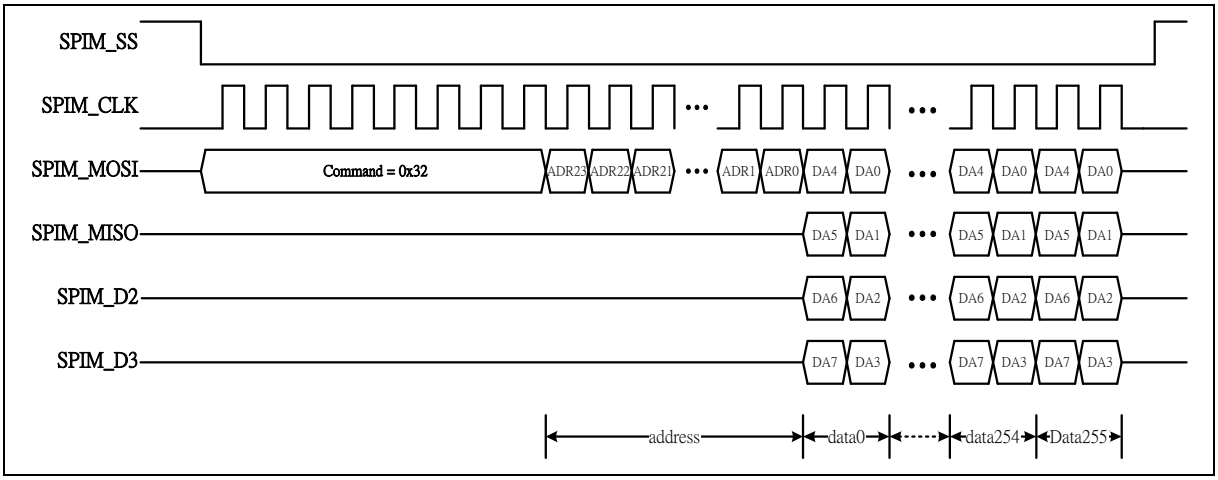


图 6.22-3 类型_1 四线模式写的编程命令模式

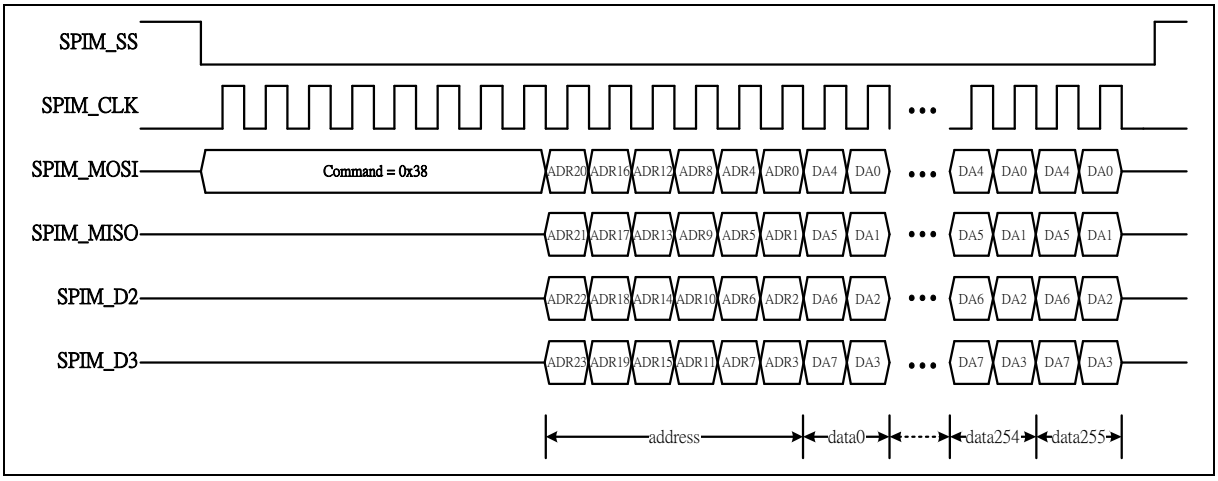


图 6.22-4 类型_2 四线模式写的编程命令模式

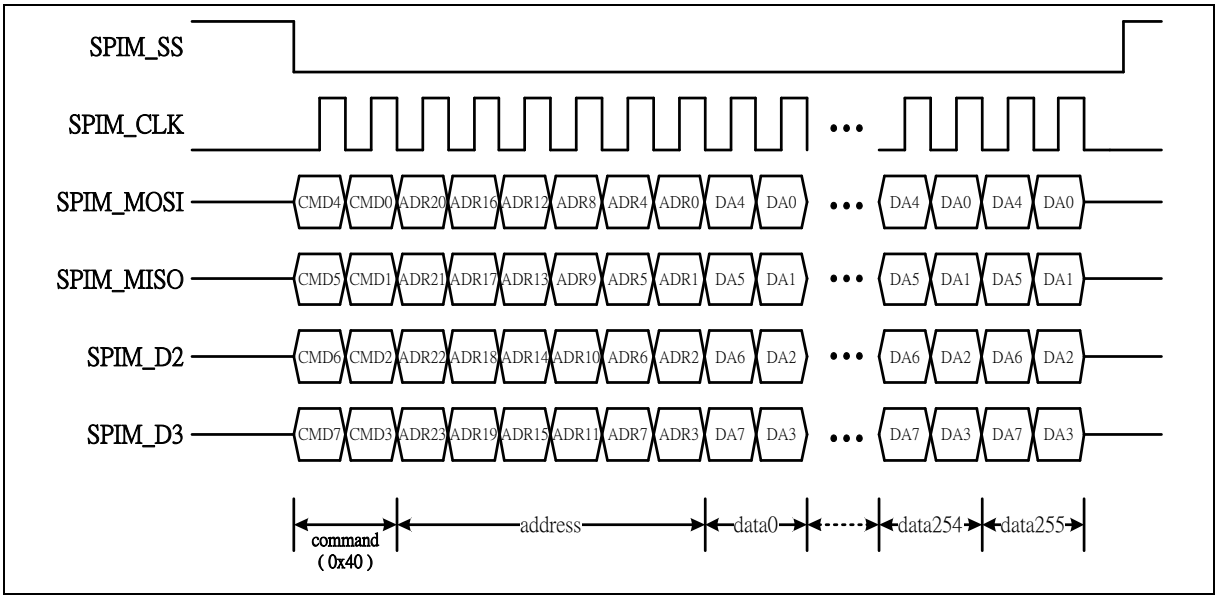


图 6.22-5 类型_3 四线模式写的编程命令模式

6.22.5.5 直接内存映射模式

在直接内存映射模式下， SPI Flash可以被当作一个ROM模块。控制器会通过AHB访问SPI Flash， 不需要MCU设置相关的SPI Flash命令。 MCU唯一需要做的是设置CMDCODE (SPIM_CTL0[31:24])为0x03, 0x0B, 0x3B, 0xBB, 0xEB, 0xE7, 0x0D, 0xBD,或 0xED.命令代码，然后用户就可以像一个ROM模块一样访问SPI Flash。

在直接内存映射缓存关闭模式下，一个直接内存映射访问后，会预先取四个字的Flash数据。在直接内存映射模式，带有缓存，并且内核耦合存储器模式（CCM）关闭的情况下，将会使用缓存内存来减少对外部SPI flash 器件的访问次数。因此对SPI flash访问可以改善。如果用户想要在直接内存映射访问之后修改SPIM硬件寄存器，需要等至少250个外设时钟周期(SPIM总线周期)。

在直接内存映射模式(DMM模式)下，双/四线读命令码“0xEB, 0xE7, 0xBB”和DTR/DDR 读命令码 “0x0D, 0xBD, 0xED”支持连续读模式(或性能增强模式)这样可以减少8个命令时序时钟，在SPI flash器件片选拉低后读地址数据被允许进入。(用户需要参考SPI Flash的规格书了解支持的信息)。图6.22-6中展示了在连续读模式禁用情况下地址模式是3字节的时候四线I/O模式SPI 数据传输。

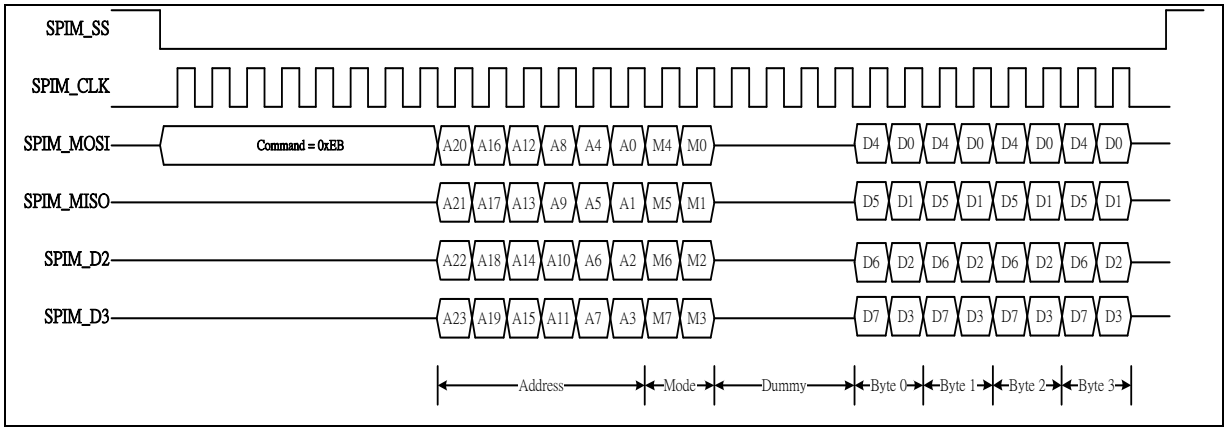


图 6.22-6 连续读模式被禁用下命令为 0xEB 的快速读四线 I/O

图6.22-7中展示了在连续读模式使能情况下地址模式是3字节的时候四线I/O模式SPI数据传输。

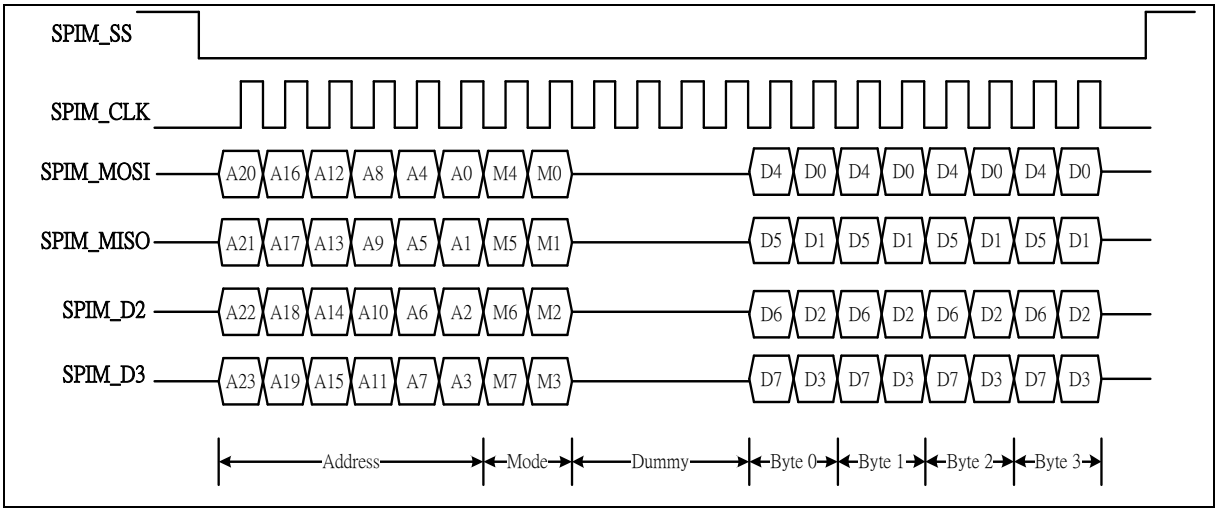


图 6.22-7 连续读模式使能下命令为 0xEB 的快速读四线

以下列出了在直接内存映射模式下SPI flash厂商连续读模式所能支持的读命令0xBB, 0xEB, 0xE7, 0x0D, 0xBD, 和 0xED：

- 支持读命令0xBB的SPI flash厂商：仅Winbond
- 支持读命令0xEB的SPI flash厂商：Winbond, Gigadevice, Spansion, EON, 和 MXIC
- 支持读命令0xE7的SPI flash厂商：仅Winbond
- 支持DTR/DDR读命令0x0D的SPI flash厂商：请检查所使用的SPI flash规格书
- 支持DTR/DDR双线I/O读命令0xBD的SPI flash厂商：请检查所使用的SPI flash规格书
- 支持DTR/DDR四线I/O读命令0xBD的SPI flash厂商：请参考所使用的SPI flash规格书

在 DMM 模式连续读模式开启情况下，用户在使用命令码 0xEB, 0xE7, 0x0D, 0xBD, 和 0xED 去读完 SPI flash 中最后所需的数据后，用户需要使用普通 I/O 模式发送 0xFF(3 字节地址模式模式下 8 个 SPI 时钟)或 0x3FF(4 字节地址模式 10 个 SPI 时钟)到外部 SPI flash 中，用 SPI 串口接口引脚 sdo_0，这样来禁用外部 SPI flash 连续读模式。了解连续读模式（或性能增强模式）的信息请参考所使用 SPI

flash 规格书。

在 DMM 模式连续读模式开启情况下，用户在使用命令码“0xBB”去读完 SPI flash 中最后所需的数据后，用户需要使用普通 I/O 模式发送 0xFFFF(3 字节地址模式模式下 16 个 SPI 时钟)或 0xFFFFF (4 字节地址模式 20 个 SPI 时钟)到外部 SPI flash 中，用 SPI 串口接口引脚 sdo_0。了解连续读模式（或性能增强模式）的信息请参考所使用 SPI flash 规格书。

用户可以使用下面普通I/O模式的设置步骤来发送连续读模式复位到外部SPI flash。想了解连续读模式（或性能增强模式）的信息请参考所使用SPI flash规格书

在直接内存映射模式下对于命令码 0xEB, 0xE7, 0x0D, 0xBD, 0xED 连续读模式复位（禁用）的步骤:

- 设置 OPMODE(SPIM_CTL0[23:22])为0x0 (普通 I/O 模式)
- 设置BITMODE(SPIM_CTL0[21:20]) 为0x2 (四线模式, 使用4个串行SPI信号发送数据到SPI flash)
- 设置QDIODIR(SPIM_CTL0[15]) 为0x1 (写模式)
- 设置BURSTNUM(SPIM_CTL0[14:13]) 为0x0 (在一次传输中只有一次发送/接收数据交换)
- 设置DWIDTH(SPIM_CTL0[12:8]) 为0x1F (发送 32位数据到SPI flash)
- 设置SPIM_TX0 寄存器为0xFFFFFFFF
- 置位SPIMEN(SPIM_CTL1[0]) 启动普通 I/O 操作
- 在普通I/O模式启动后，轮询 SPIMEN(SPIM_CTL1[0]) 检查运行状态 (1 = 忙, 0 = 完成)
- 设置SS(SPIM_CTL1[4]) 让 SPIM_SS 处在非激活电平

- **在直接内存映射模式下对于命令码0xBB连续读模式复位的步骤:**
  - 设置OPMODE(SPIM_CTL0[23:22]) 为 0x0 (普通 I/O 模式)
  - 设置BITMODE(SPIM_CTL0[21:20]) 为 0x1 (双线模式, 使用2个串行SPI信号发送数据到SPI flash)
  - 设置QDIODIR(SPIM_CTL0[15]) 为 0x1 (写模式)
  - 设置BURSTNUM(SPIM_CTL0[14:13]) 为 0x0 (在一次传输中只有一次发送/接收数据交换)
  - 设置DWIDTH(SPIM_CTL0[12:8]) 为 0x1F (发送 32位数据到SPI flash)
  - 设置SPIM_TX0 寄存器为0xFFFFFFFF
  - 置位SPIMEN(SPIM_CTL1[0]) 启动普通 I/O 操作
  - 在普通I/O模式启动后，轮询 SPIMEN(SPIM_CTL1[0]) 检查运行状态 (1 = 忙, 0 = 完成)
  - 设置SS(SPIM_CTL1[4]) 让 SPIM_SS 处在非激活电平
- 在直接内存映射模式下，四线读命令“0xEB” 和 “0xE7”支持突发包模式的缓存应用和连续读模式(或性能增强模式)。对于缓存应用，突发包模式可以快速地被用来填充缓存线(在该SPI flash控制器下，我们使用缓存数据线大小是16个字节)。对于直接内存映射模式和缓存使能条件下的连续读模式(或性能增强模式)，如果缓存没有命中，突发包模式可以快速的让MCU获得SPI flash数据请求。在突发包模式和连续读模式要从使能状态切换到禁用状态时，首先需要禁止外部SPI Flash的

连续读模式，然后禁止外部SPI Flash的突发包模式。

在用户使用 DMM 模式下读命令码“0xEB 或 0xE7”和突发包模式访问外部 flash 之前，请使用 SPI flash 控制器的普通 I/O 模式通过 SPI 的 sdo_0 ~ sdo_3 串行接口管脚发送“0x1101_1101 和 0x2000_0000”(外部 SPI flash 突发包使能命令需要 16 个 SPI 时钟，请参考使用的 SPI Flash 的规格书了解突发包模式的配置)到外部 SPI flash。在用户使用 DMM 模式下读命令码“0xEB 和 0xE7” 配合突发包模式读完外部 flash 最后需要的数据后，请使用 SPI flash 控制器的普通 I/O 模式通过 SPI 的 sdo_0 ~ sdo_3 串行接口管脚发送“0x1101_1101 和 0x0000_0000” (外部 SPI flash 突发包禁用命令需要 16 个 SPI 时钟，请参考使用的 SPI Flash 的规格书了解突发包模式的配置)到外部 SPI flash。

用户可以使用下面SPI flash控制器普通I/O模式的设置步骤来发送突发包模式使能或突发包模式禁用的 SPI 命令码到外部SPI flash

- **直接内存映射模式下使能突发包模式的步骤：**
  - 设置 OPMODE(SPIM_CTL0[23:22]) 为 0x0 (普通I/O)
  - 设置BITMODE(SPIM_CTL0[21:20]) 为 0x2 (四线模式, 使用4个串行SPI信号发送数据到SPI flash)
  - 设置QDIODIR(SPIM_CTL0[15]) 为0x1 (写模式)
  - 设置BURSTNUM(SPIM_CTL0[14:13]) 为0x1 (一次传输中有两次发送/接收数据交换)
  - 设置DWIDTH(SPIM_CTL0[12:8]) 为0x1F (发送 32位数据到SPI flash)
  - 设置SPIM_TX0 寄存器为 0x20000000, SPIM_TX1 寄存器为0x11011101
  - 设置SPIMEN(SPIM_CTL1[0]) 启动普通 I/O 操作
  - 在普通I/O模式启动后，轮询 SPIMEN(SPIM_CTL1[0]) 检查运行状态 (1 = 忙, 0 = 完成)
  - 设置SS(SPIM_CTL1[4]) 让 SPIM_SS 处在非激活电平

- **直接内存映射模式下禁用突发包模式的步骤：**
  - 设置OPMODE(SPIM_CTL0[23:22]) 为0x0 (普通I/O)
  - 设置BITMODE(SPIM_CTL0[21:20]) 为0x2 (四线模式, 使用4个串行SPI信号发送数据到SPI flash)
  - 设置QDIODIR(SPIM_CTL0[15]) 为0x1 (写模式)
  - 设置BURSTNUM(SPIM_CTL0[14:13]) 为0x1 (一次传输中有两次发送/接收数据交换)
  - 设置DWIDTH(SPIM_CTL0[12:8]) 为0x1F (发送 32位数据到SPI flash)
  - 设置SPIM_TX0寄存器为0x00000000, SPIM_TX1寄存器为0x11011101
  - 设置SPIMEN(SPIM_CTL1[0]) 启动普通 I/O 操作
  - 在普通I/O模式启动后，轮询 SPIMEN(SPIM_CTL1[0]) 检查运行状态 (1 = 忙, 0 = 完成)
  - 设置SS(SPIM_CTL1[4]) 让 SPIM_SS 处在非激活电平

为了在不增加串行时钟频率下提高读操作的吞吐量，该SPI flash控制器支持DTR/DDR (双倍传输速率/双倍数据速率)读命令码(i.e. 0x0D, 0xBD, 0xED)，该命令码在直接内存映射模式和DMA读模式下支持标

准/双线/四线SPI模式。一个字节的命令码依然是8个上升沿串行时钟被锁存到设备。一旦一条DTR/DDR指令码被设备接受，SPI flash读地址、模式位、空闲周期（dummy cycle）和输出数据，这些从SPI flash器件发送出数据会被在上升沿和下降沿锁存。下面为我们展示了DTR/DDR 读命令 0x0D, 0xBD, 和0xED的时序图。

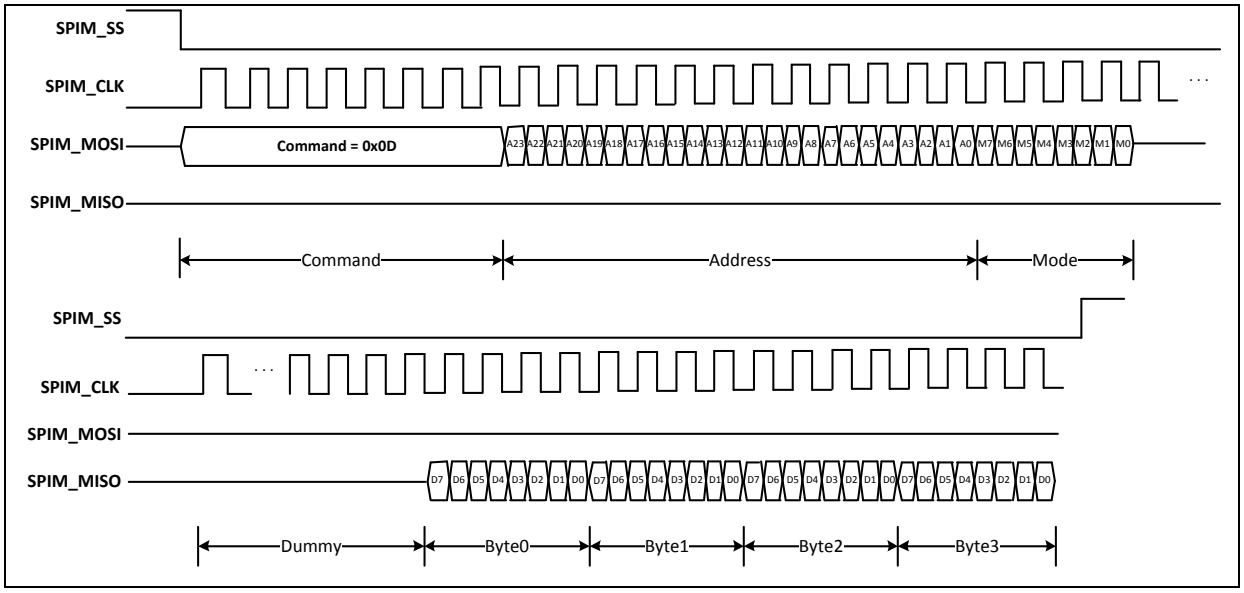


图 6.22-8 DTR/DDR 快速读命令 0x0D

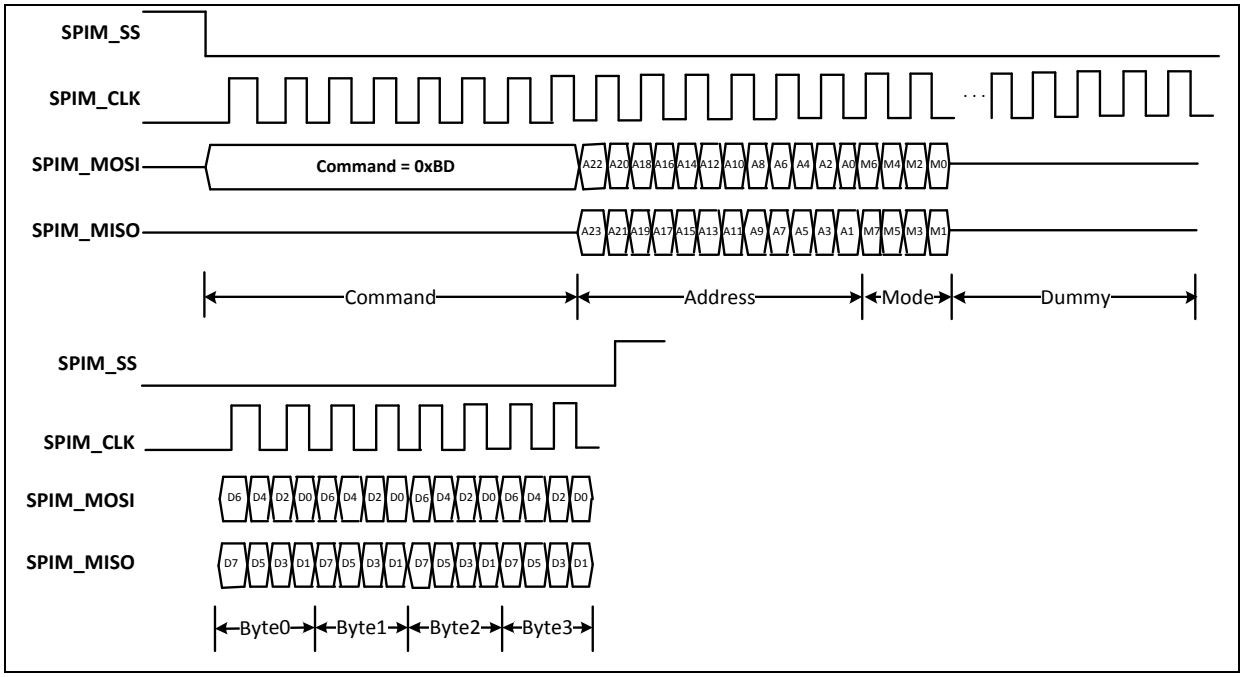


图 6.22-9 DTR/DDR 快速读双线 I/O 命令 0xBD

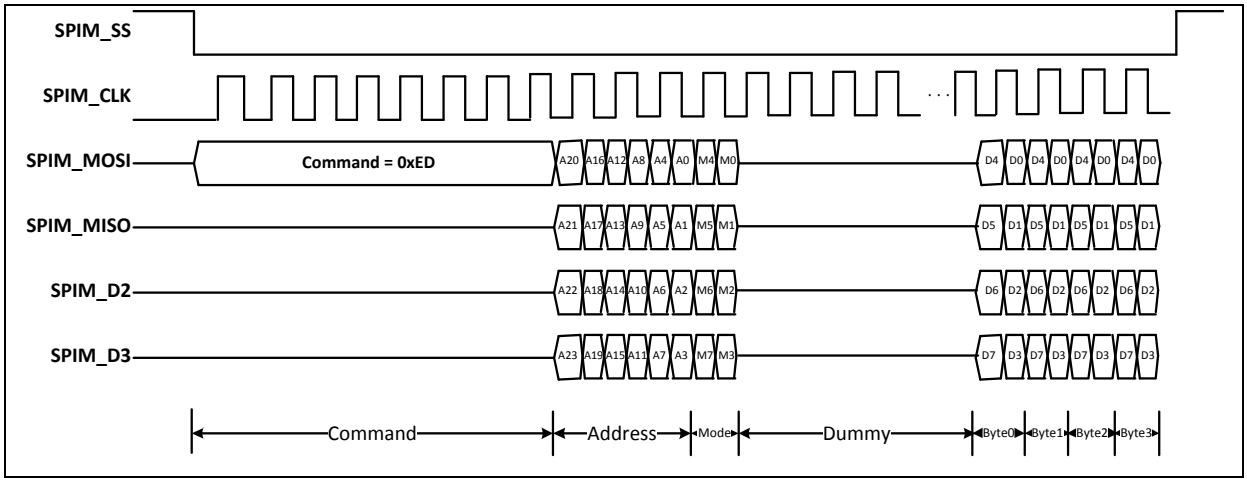


图 6.22-10 DTR/DDR 快速读四线 I/O 命令 0xED

空闲周期（dummy cycle）数取决于SPI flash 厂商、操作频率和命令类型，请参考所使用的SPI flash说明书了解准确的空闲周期（dummy cycle）数。

6.22.5.6 内核耦合存储器模式（CCM模式）

当用户设置CCMEN(SPIM_CTL1[2])为1时，缓存存储器操作模式会转变成内核耦合存储器模式(CCM模式)。在CCM模式下，SPI flash控制器的缓存功能会被硬件自动禁用，MCU访问内核耦合存储器就像是SRAM一样。当用户设置CCMEN(SPIM_CTL1[2])为0时MCU访问CCM地址范围，SPI flash控制器将会产生AHB错误给MCU

6.22.5.7 密码加密和解密

SPI flash控制器支持密码加密和解密功能，该功能可以用来保护用户放在外部SPI flash的数据。当用户使用DMA写模式并且密码功能使能时，密码电路会加密从SRAM传输到外部SPI flash的数据。当用户使用DMA读模式并且密码功能使能时，密码电路会解密从外部SPI flash传输到SRAM的数据。当用户使用直接内存映射模式并且密码功能使能时，密码电路会解密MCU从外部SPI flash读到的数据。在普通I/O模式，发送到外部SPI flash或接收外部SPI flash的数据都不会用到密码电路进行加密和解密

6.22.6 SPI flash控制器在AMBA 系统中映射地址

地址空间	标记	控制器
SPIM控制器映射地址，SPI flash内存空间，内核耦合存储空间		
0x4000_7000 – 0x4000_7FFF	SPIM_BA	SPIM 控制寄存器
(1) 当芯片 SRAM 大小为 96KB : 0x1001_0000 – 0x1001_7FFF 0x2001_0000 – 0x2001_7FFF **(2) 当芯片 SRAM 大小为128KB :** 0x1001_8000 – 0x1001_FFFF 0x2001_8000 – 0x2001_FFFF **(3) 当芯片 SRAM 大小为160KB :** 0x1002_0000 – 0x1002_7FFF 0x2002_0000 – 0x2002_7FFF	SRAM2_BA	仅对MCU的SRAM内存空间(32KB)和SPIM缓存共享(当CCM模式使能) **注：**芯片可以配置成三种大小的SRAM(160KB/128KB/96KB).在CCM模式下请检查SRAM的配置来确认MCU可以访问的地址范围。
0x0800_0000 – 0x09FF_FFFF	SPIM_FLASH_BA	对于直接内存映射模式(DMM模式)，SPIM内存空间(32MB)可以访问地址范围

表 6.22-1SPI flash 控制器在 AMBA 系统中映射地址

6.22.7 寄存器映射

R: 只读, **W**: 只写, **R/W**: 可读可写, **C**: 只能写0

寄存器	偏移地址	R/W	描述	复位值
SPIM基地址: **SPIM_BA = 0x4000_7000**				
SPIM_CTL0	SPIM_BA+0x00	R/W	控制和状态寄存器 0	0x00C0_0002
SPIM_CTL1	SPIM_BA+0x04	R/W	控制寄存器 1	0x0000_0010
SPIM_RXCLKDLY	SPIM_BA+0x0C	R/W	RX 时钟延时控制寄存器	0x0000_010F
SPIM_RX0	SPIM_BA+0x10	R	数据接收寄存器0	0x0000_0000
SPIM_RX1	SPIM_BA+0x14	R	数据接收寄存器1	0x0000_0000
SPIM_RX2	SPIM_BA+0x18	R	数据接收寄存器2	0x0000_0000
SPIM_RX3	SPIM_BA+0x1C	R	数据接收寄存器3	0x0000_0000
SPIM_TX0	SPIM_BA+0x20	R/W	数据发送寄存器 0	0x0000_0000
SPIM_TX1	SPIM_BA+0x24	R/W	数据发送寄存器1	0x0000_0000
SPIM_TX2	SPIM_BA+0x28	R/W	数据发送寄存器2	0x0000_0000
SPIM_TX3	SPIM_BA+0x2C	R/W	数据发送寄存器3	0x0000_0000
SPIM_SRAMADDR	SPIM_BA+0x30	R/W	SRAM 内存地址寄存器	0x0000_0000
SPIM_DMACNT	SPIM_BA+0x34	R/W	DMA 传输字节计数寄存器	0x0000_0000
SPIM_FADDR	SPIM_BA+0x38	R/W	SPI Flash 地址寄存器	0x0000_0000
SPIM_KEY1	SPIM_BA+0x3C	W	密钥1 寄存器	0x0000_0000
SPIM_KEY2	SPIM_BA+0x40	W	密钥2 寄存器	0x0000_0000
SPIM_DMMCTL	SPIM_BA+0x44	R/W	直接内存映射模式控制寄存器	0x0008_0000
SPIM_CTL2	SPIM_BA+0x48	R/W	控制寄存器 2	0x0804_0000

注：如果软件想写该外设的任何寄存器， SPIM_CTL1 寄存器的 SPIMEN 位应该要为低状态。

6.22.8 寄存器描述

控制和状态寄存器0 (SPIM_CTL0)

寄存器	偏移地址	R/W	描述	复位值
SPIM_CTL0	SPIM_BA+0x00	R/W	控制和状态寄存器0	0x00C0_0002

31	30	29	28	27	26	25	24
CMDCODE							
23	22	21	20	19	18	17	16
OPMODE		BITMODE		SUSPITV			
15	14	13	12	11	10	9	8
QDIODIR	BURSTNUM		DWIDTH				
7	6	5	4	3	2	1	0
IF	IEN	B4ADDREN	Reserved		BALEN	Reserved	CIPHOFF

位	描述	
[31:24]	CMDCODE	**页编程命令码 (注 4)** 0x02 = 页编程 (用于 DMA 写模式). 0x32 = 用类型_1编程流程的四线页编程(用于 DMA 写模式). (**注3)** 0x38 = 用类型_2编程流程的四线页编程(用于 DMA 写模式). (**注3)** 0x40 = 用类型_3 编程流程的四线页编程(用于 DMA 写模式). (**注 3)** 其它 = 保留. **读命令码 :** 0x03 = 标准读(用于 DMA 读/DMM 模式). 0x0B = 快速读 (用于 DMA 读/DMM 模式). 快速读命令码“0x0B”类似于标准读命令码“0x03”，除此之外它还可以工作在最高可能的频率(**注 2**) 0x3B = 快速读双线输出 (用于 DMA 读/DMM 模式). 0xBB = 快速读双线 I/O (用于 DMA 读/DMM 模式). 快速读双线 I/O 命令码“0xBB”类似于快速读双线输出命令码“0x3B”，但是它还具有在每个时钟下可以输入 2 位的地址位的能力 (**注 2**) 0xEB = 快速四线读 (用于 DMA 读/DMM 模式). 0xE7 = 字四线读 (用于 DMA 读/DMM 模式). 字四线读命令码“0xE7”类似于快速四线读命令码“0xEB”，除此之外它的最低地址位必须为 0 并且空闲周期数要比快速四线读少 (**注 2**) 0x0D = DTR/DDR 快速读 (用于 DMA 读/DMM 模式). 0xBD = DTR/DDR 双线读 (用于 DMA 读/DMM 模式). 0xED = DTR/DDR 四线读 (用于 DMA 读/DMM 模式). **其它命令码保留** DTR/DDR 读命令 0x0D,0xBD,0xED”通过在 SPI flash 时钟(SPIM_CLK)的上升沿和下降沿传输地址和数据来提高吞吐量。它类似于“0x0B, 0xBB, 0xEB”这些命令码，但是它可以在 SPI flash

		输出时钟的上升沿和下降沿都能够传输地址和数据。(**注 2**) **注 1:**在使用四线页编程/四线读命令之前，需要要先通过普通 I/O 模式使能 SPI flash 的四线模式。 **注 2:**参考 SPI flash 规格书所支持的命令码 **注 3:**对于页编程命令码的类型_1, 类型_2, 和类型_3，参考 图 6.22-3，图 6.22-4，和 图 6.22-5. **注 4:**在使用 SPI flash 控制器的 DMA 写模式去编程外部 SPI flash 之前，请禁用"连续读模式"和"突发包模式"。在使用 SPI flash 控制器 DMA 写模式去编程外部 SPI flash 的内容之后，请设置 CDINVAL(SPIM_CTL1[3])为 1（设置所有缓存数据无效）
[23:22]	**OPMODE**	**SPI 功能操作模式** 0x0 = 普通 I/O 模式. (**注 1**) (**注 3**) 0x1 = DMA 写模式. (**注 2**) (**注 3**) 0x2 = DMA 读模式. (**注 3**) 0x3 = 直接内存映射模式(DMM 模式) (默认). (**注 4**) **注 1 :**在使用 SPI flash 控制器普通 I/O 模式去编程外部 SPI flash 内容之后，请设置 CDINVAL(SPIM_CTL1[3])为 1（设置所有缓存数据无效） **注 2 :** 在 DMA 模式下，每次操作硬件会发送一页编程编程命令。用户必须要小心处理好跨页的情况。在使用 SPI flash 控制器 DMA 写模式去编程外部 SPI flash 的内容之后，请设置 CDINVAL(SPIM_CTL1[3])为 1（设置所有缓存数据无效） **注 3 :** 对于 32MB 的外部 SPI flash，在使用普通 I/O 模式、DMA 写模式和 DMA 读模式去写/读该外部 SPI flash 时，访问地址范围是 0x00000000 to 0x01FFFFFF。请用户核对好使用的 SPI flash 器件的容量，了解该外部 SPI flash 的访问地址范围。 **注 4 :** 对于 32MB 的外部 SPI flash，在使用直接内存映射模式（DMM 模式）去读该外部 SPI flash 时，访问地址范围是 0x08000000 to 0x09FFFFFF。请用户核对好使用的 SPI flash 器件的容量，了解该外部 SPI flash 的访问地址范围。
[21:20]	**BITMODE**	**SPI 接口位模式** 0x0 = 标准模式. 0x1 = 双线模式. 0x2 = 四线模式. 0x3 = 保留. **注:**仅用于普通 I/O 模式
[19:16]	**SUSPITV**	**暂停间隔** 这四个位域提供一次传输中两个连续发送/接收事务之间的挂起时间间隔的配置。默认值是 0x00。当 BURSTNUM = 00，设置该域没有影响。想要的时间间隔可以根据下面的等式获得（从当前时钟的最后一个下降沿到下一个时钟的上升沿） • (SUSPITV+2)* AHB 时钟周期 • 0x0 = 2 AHB时钟周期 • 0x1 = 3 AHB时钟周期 • …… • 0xE = 16 AHB时钟周期 • 0xF = 17 AHB时钟周期 • **注:**仅用于普通 I/O模式
[15]	**QDIODIR**	**四线/双线模式的 SPI 接口方向选择** 0 =接口信号为输入 1 =接口信号为输出

		注:仅用于普通 I/O 模式
[14:13]	**BURSTNUM**	**发送/接收突发个数** 该域指定在一次传输中多少个发送/接收事务应该被连续执行 0x0 = 在一次传输中，仅执行一个发送/接收事务 0x1 = 在一次传输中，执行两个连续的发送/接收事务. 0x2 = 在一次传输中，执行三个连续的发送/接收事务. 0x3 = 在一次传输中，执行四个连续的发送/接收事务. **注:**仅用于普通 I/O 模式
[12:8]	**DWIDTH**	**发送/接收位长度** 该域指定一次发送/接收事务中多少位被发送/接收 0x7 = 8 位. 0xF = 16 为. 0x17 = 24 为. 0x1F = 32 为. 其他 = 不正确的传输结果 **注 1:**仅用于普通 I/O 模式 **注 2:** 仅 8-, 16-, 24-, 和 32-位是允许的，其他位长度会导致不正确的传输
[7]	**IF**	**中断标志** **(1) 写操作：** 0 = 没有影响 1 = 写 1 清 0 **(2) 读操作：** 0 = 传输还未完成 1 = 传输已完成
[6]	**IEN**	**中断使能控制** 0 = SPIM 中断禁用 1 = SPIM 中断使能
[5]	**B4ADDREN**	**4-字节地址模式使能控制** 0 = 禁用 4-字节地址模式, 和使能 3-字节地址模式 1 = 使能 4-字节地址 **注:**用于 DMA 写/DMA 读/DMM 模式.
[4:3]	**Reserved**	保留
[2]	**BALEN**	**在密码使能和禁用控制之间平衡 AHB 控制时间** 当密码使能，AHB 控制信号根据编码解码的计算会延时一段时间，所以如果设置 BALEN 为 1，会使密码使能和密码禁止时的 AHB 信号处理时间相等 **注:**仅当密码禁用时有用
[1]	**Reserved**	保留
[0]	**CIPHOFF**	**密码禁用控制** 0 =密码功能使能 1 =密码功能禁用 **注 1:**如果 KEY1(SPIM_KEY1[31:0]) 或 KEY2(SPIM_KEY2[31:0])什么都没有（KEY1 为 0x0000_0000 或 KEY2 是 0x0000_0000），密码功能会被自动禁用。 **注 2:**当 CIPHOFF(SPIM_CTL0[0])为 0，KEY1(SPIM_KEY1[31:0])和 KEY2(SPIM_KEY2[31:0])都不为 0x0000_0000(i.e. KEY1≠ 0x0000_0000 和 KEY2≠0x0000_0000)，使能密码加密和解密

		注 3 : 当使能密码加密和解密时，请设置 DESELTIM(SPIM_DMMCTL[20:16]) >= 0x10。当禁用密码加密和解密时，请设置 DESELTIM(SPIM_DMMCTL[20:16]) >= 0x8。

控制寄存器1 (SPIM_CTL1)

寄存器	偏移地址	R/W	描述	复位值
SPIM_CTL1	SPIM_BA+0x04	R/W	控制寄存器1	0x0000_0010

31	30	29	28	27	26	25	24
DIVIDER							
23	22	21	20	19	18	17	16
DIVIDER							
15	14	13	12	11	10	9	8
Reserved				IDLETIME			
7	6	5	4	3	2	1	0
Reserved		SSACTPOL	SS	CDINVAL	CCMEN	CACHEOFF	SPIMEN

位	描述	
[31:16]	**DIVIDER**	**时钟分频寄存器** 该域的值是 AHB 时钟(HCLK)的频率分频值，用来产生 SPIM_CLK 引脚上输出的串行时钟，想要的频率可以根据下面的公式获得 $f_{SPIM_CLK} = \frac{f_{HCLK}}{(DIVIDER)*2}$ **注 1:** 当设置 DIVIDER 为 0，SPIM_CLK 的频率会等于 HCLK 的频率 **注 2:** SCLK 为串行 SPI 输出时钟 **注 3:**请参考所使用 SPI flash 器件的规格书决定 SPI flash 时钟频率 **注 4:**对于 DTR/DDR 读命令 “0x0D, 0xBD, 0xED”, DIVIDER 设置值仅为 1,2,4,8,16,32,…,2^n, 且 n = 0,1,2,3,4, ….
[15:12]	**Reserved**	保留
[11:8]	**IDLETIME**	**空闲时间间隔** 在 DMM 模式, IDLECNT 用来控制两次 SPI Flash 访问之间的最小的空闲时间. 最小时间 = (IDLETIME + 1) * AHB 时钟周期时间. **注 1:**仅使用于 DMM 模式. **注 2 :** AHB 时钟周期时间= 1/AHB 时钟频率
[7:6]	**Reserved**	保留
[5]	**SSACTPOL**	**从机选择激活电平** 该位定义了设备/从机选择信号的激活电平(SPIM_SS)，表 7.20-2 有列举 0 = SPIM_SS 从机选择信号低有效. 1 = SPIM_SS 从机选择信号高有效.
[4]	**SS**	**从机选择激活使能控制** 0 = SPIM_SS 处在激活电平

		1 = SPIM_SS 处在非激活电平(默认). **注:**在给定的时间，这个接口仅驱动一个设备/从机。所以在开始任何读或写传输之前，已选择的设备的从机选择必须设置成它的有效电平。SSACTPOL(SPIM_CTL1[5]) 和 SS 的功能说明见表 7.20-2
[3]	**CDINVAL**	**缓存数据无效使能控制** **(1) 写操作:** 0 = 无影响 1 = 设置所有缓存数据无效。该位被硬件自动清除。 **(2) 读操作 :**无影响 **注:** 当 SPI flash 内存进行页擦除或整个 flash 擦除时，请设置 CDINVAL 为 1.在使用 SPI flash 控制器的普通 I/O 模式或 DMA 写模式去编程或擦除外部 SPI flash 的内容之后，请设置 CDINVAL 为 1.
[2]	**CCMEN**	**CCM 模式使能控制** 0 = 禁用 CCM 模式. (默认值) 1 = 使能 CCM 模式 **注 1:**当 CCM 模式使能时，缓存功能会被硬件自动禁用。当 CCM 模式被禁用时，缓存功能被使能或是禁用是用户自己设置。 **注 2:**当 CCM 模式禁用时，用户通过总线主控访问内核耦合存储器。该情况下，SPI flash 控制器将会通过 HRESP 总线发送错误响应信号到总线主控。 **注 3:**当 CCM 模式使能，用户需要设置 CCMEN 为 1 并且需要需要读该寄存器显示当前硬件的状态。当读出 CCMEN 值为 1 时，MCU 就可以启动从 CCM 存储空间读数据或是写入数据到 CCM 存储空间。
[1]	**CACHEOFF**	**缓存存储功能禁用控制** 0 = 使能缓存存储功能. (默认值) 1 = 禁用缓存存储功能. **注:**当 CCM 模式使能时，缓存功能会被硬件自动禁用。当 CCM 模式被禁用时，缓存功能被使能或是禁用是用户自己设置。
[0]	**SPIMEN**	**执行和忙状态** **(1) 写操作 :** 0 = 无影响 1 = 启动传输，该位在传输期间保持为 1，传输完成后被自动清除 **(2) 读操作 :** 0 = 传输已完成 1 = 传输还未完成 **注:**在写 1 到 SPIMEN 位之前，所有寄存器应该设置好。在传输过程中，不应该写该外设的任何寄存器。

SSACTPOL	SS	功能描述
0	0	SPIM_SS 低电平有效, 并且设置SPIM_SS为 0
0	1	SPIM_SS低电平有效, 并且设置SPIM_SS 为 1
1	0	SPIM_SS 高电平有效,并且设置SPIM_SS 为1
1	1	SPIM_SS高电平有效,并且设置SPIM_SS 为 0

表 6.22-2 从机片选电平和激活使能的功能描述

RX时钟延时控制寄存器(SPIM_RXCLKDLY)

寄存器	偏移地址	R/W	描述	复位值
SPIM_RXCLKDLY	SPIM_BA+0x0C	R/W	RX时钟延时控制寄存器	0x0000_010F

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved			RDEDGE	Reserved	RDDLYSEL		
15	14	13	12	11	10	9	8
PHDELSEL							
7	6	5	4	3	2	1	0
DWDELSEL							

位	描述	
[31:21]	**Reserved**	保留
[20]	**RDEDGE**	**接收数据时的采样时钟边沿选择** **针对于普通 I/O 模式、DMA 读模式、DMA 写模式和直接内存映射模式** 0 : 使用 SPI 输入时钟上升沿去采样接收到数据 (默认值) 1 : 使用 SPI 输入时钟下降沿去采样接收到数据
[19]	**Reserved**	保留
[18:16]	**RDDLYSEL**	**接收数据时的采样时钟延时选择** **针对于普通 I/O 模式、DMA 读模式、DMA 写模式和直接内存映射模式** 决定插入延时周期的数目。用来调整接收数据的采样时钟来锁存正确的数据。 0x0 : 没有延时. (默认值) 0x1 : 延时 1 个 SPI flash 时钟 0x2 : 延时 2 个 SPI flash 时钟 0x3 : 延时 3 个 SPI flash 时钟 … 0x7 : 延时 7 个 SPI flash 时钟 **注 :**我们可以使用外部 SPI flash 器件的制造商 ID 或设备 ID 来决定 RDDLYSEL 正确的设置，请参考下面的范例。 例如，外部 SPI flash 器件的制造商 ID 和设备 ID 分别是 0xEF 和 0x1234。首先我们设置 RDDLYSEL 为 0，通过普通 I/O 模式使用读制造商 ID/设备 ID 命令去读制造商 ID（本例程的制造商 ID 是 0xEF (1110_1111)）。 如果从外部 SPI flash 读出的制造商 ID 是 0xF7 (1111_0111),这表明制造商 ID 被右移了一位并且大多数制造商 ID 的最高有效位(MSB)都是 1。依据从外部 SPI flash 读出的制造商 ID，我们需要设置 RDDLYSEL 为 1 来接收正确的 SPI flash 数据。
[15:8]	**PHDELSEL**	**DMA 写模式和 DMA 读模式下的 SPIFlash 的相位延时** 这一位设定当 SPI 控制器向 SPI Flash 发送数据时，指令数据，地址数据和空指令周期的相位

		差。 DMA 写模式和 DMA 读模式下的相位延时 = (PHDELSEL + 1) * AHB 时钟周期时间 (注). 注: AHB 时钟周期时间 = 1/AHB 时钟频率
[7:0]	**DWDELSEL**	**DMA 写模式下的 SPI flash 取消选择时间间隔(默认值 = 15)** **仅针对 DMA 写模式** 该寄存器设置 SPI flash 控制在 DMA 写模式下的取消选择时间间隔（也就是 SPIM_SS 非激活的时间间隔）**(注 1)** DMA 写模式的取消选择时间间隔= (DWDELSEL + 1) * AHB 时钟周期时间**(注 2)** **注 1:**请用户依据使用的外部 SPI flash 器件去设置该寄存器的值。通常，当 SPI flash 执行编程操作时 SPI flash 的取消选择时间间隔大于 50ns **注 2:** AHB 时钟周期时间= 1/AHB 时钟频率

数据接收寄存器0~3 (SPIM_RX0~3)

寄存器	偏移地址	R/W	描述	复位值
SPIM_RX0	SPIM_BA+0x10	R	数据接收寄存器0	0x0000_0000
SPIM_RX1	SPIM_BA+0x14	R	数据接收寄存器1	0x0000_0000
SPIM_RX2	SPIM_BA+0x18	R	数据接收寄存器2	0x0000_0000
SPIM_RX3	SPIM_BA+0x1C	R	数据接收寄存器3	0x0000_0000

31	30	29	28	27	26	25	24
RXDAT							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
RXDAT							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
RXDAT							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
RXDAT							

位	描述	
[31:0]	**RXDAT**	**数据接收寄存器** 数据接收寄存器保留最近一次已执行传输的接收到的数据。 . 有效RXDAT寄存器的个数在SPIM_CTL0[BURSTNUM]中指定，如果BURSTNUM > 0 ，接收到的数据先保存在最高有效RXDAT寄存器中。 有效位的个数在SPIM_CTL0[DWIDTH]中指定，如果DWIDTH是16, 24, 或 32，接收到的数据先保存在RXDAT寄存器的最低有效字节。 在一个字节中，接收到的数据先保存在RXDAT寄存器的最高有效位。 **例1:** 如果SPIM_CTL0[BURSTNUM] = 0x3 和 SPIM_CTL1[DWIDTH] = 0x17，接收到的数据会按顺序保存在SPIM_RX3[23:0], SPIM_RX2[23:0], SPIM_RX1[23:0], SPIM_RX0[23:0]. **例2:** 如果 SPIM_CTL0[BURSTNUM = 0x0 和 SPIM_CTL0[DWIDTH] = 0x17, 接收到的数据会按顺序保存在 SPIM_RX0[7:0], SPIM_RX0[15:8], SPIM_RX0[23:16]. **例3:** 如果 SPIM_CTL0[BURSTNUM = 0x0 和 SPIM_CTL0[DWIDTH] = 0x07, 接收到的数据会按顺序保存在SPIM_RX0[7], SPIM_RX0[6], …, SPIM_RX0[0].

数据发送寄存器0~3 (SPIM_TX0~3)

寄存器	偏移地址	R/W	描述	复位值
SPIM_TX0	SPIM_BA+0x20	R/W	数据发送寄存器0	0x0000_0000
SPIM_TX1	SPIM_BA+0x24	R/W	数据发送寄存器1	0x0000_0000
SPIM_TX2	SPIM_BA+0x28	R/W	数据发送寄存器2	0x0000_0000
SPIM_TX3	SPIM_BA+0x2C	R/W	数据发送寄存器3	0x0000_0000

31	30	29	28	27	26	25	24
TXDAT							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
TXDAT							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
TXDAT							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
TXDAT							

位	描述	
[31:0]	**TXDAT**	**数据发送寄存器** 数据发送寄存器保存下一次传输要发送的数据 有效TXDAT寄存器的个数在SPIM_CTL0[BURSTNUM]中指定，如果BURSTNUM > 0，数据会先发送最高有效TXDAT寄存器。 有效位的个数在SPIM_CTL0[DWIDTH]中指定。如果DWIDTH是16, 24, 或 32，会先发送TXDAT寄存器的最低有效字节 在一个字节中，先发送TX寄存器的最高有效位。 **例 1:** 如果 SPIM_CTL0[BURSTNUM] = 0x3 和 SPIM_CTL1[DWIDTH] = 0x17, 下次传输数据会按SPIM_TX3[23:0], SPIM_TX2[23:0], SPIM_TX1[23:0], SPIM_TX0[23:0] 顺序发送 **例 2:** 如果 SPIM_CTL0[BURSTNUM] = 0x0 和 SPIM_CTL0[DWIDTH] = 0x17, 下次传输数据会按SPIM_TX0[7:0], SPIM_TX0[15:8], SPIM_TX0[23:16] 顺序发送 **例3:** 如果 SPIM_CTL0[BURSTNUM] = 0x0 和 SPIM_CTL0[DWIDTH] = 0x07, 下次传输数据会按SPIM_TX0[7], SPIM_TX0[6], …, SPIM_TX0[0] 顺序发送

SRAM内存地址寄存器(SPIM_SRAMADDR)

寄存器	偏移地址	R/W	描述	复位值
SPIM_SRAMADDR	SPIM_BA+0x30	R/W	SRAM内存地址寄存器	0x0000_0000

31	30	29	28	27	26	25	24
ADDR							
23	22	21	20	19	18	17	16
ADDR							
15	14	13	12	11	10	9	8
ADDR							
7	6	5	4	3	2	1	0
ADDR							

位	描述	
[31:0]	ADDR	**SRAM内存地址** 对于 DMA 读模式,该寄存器是DMA传输的目的地址 对于 DMA 写模式, 该寄存器是DMA 传输的源地址 **注:**该地址必须是字对齐.

DMA传输字节计数寄存器(SPIM_DMACNT)

寄存器	偏移地址	R/W	描述	复位值
SPIM_DMACNT	SPIM_BA+0x34	R/W	DMA传输字节计数寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
DMACNT							
15	14	13	12	11	10	9	8
DMACNT							
7	6	5	4	3	2	1	0
DMACNT							

位	描述	
[31:24]	Reserved	保留
[23:0]	DMACNT	**DMA传输字节计数寄存器** 该位表示DMA传输的长度 **注1:**计数单位是字节 **注2:**.数值必须是4的倍数 **注3:**请核对所使用的SPI flash的规格书了解页编程最大字节长度

SPI Flash地址寄存器(SPIM_FADDR)

寄存器	偏移地址	R/W	描述	复位值
SPIM_FADDR	SPIM_BA+0x38	R/W	SPI地址寄存器	0x0000_0000

31	30	29	28	27	26	25	24
ADDR							
23	22	21	20	19	18	17	16
ADDR							
15	14	13	12	11	10	9	8
ADDR							
7	6	5	4	3	2	1	0
ADDR							

位	描述	
[31:0]	ADDR	**SPI Flash地址寄存器** 对于 DMA 读模式, 该寄存器是 DMA 传输的源地址. 对于 DMA 写模式, 该寄存器是DMA 传输的目的地址. **注1 :** 该地址必须是字对齐. **注2 :**对于32MB的外部SPI flash，当用户使用DMA写和DMA读模式去写/读外部SPI flash数据时，该SPI flash地址寄存器“ADDR”的值是从0x00000000 到 0x01FFFFFF。请检查所使用的SPI flash器件的大小了解外部SPI flash的访问地址范围。

密钥1 寄存器(SPIM_KEY1)

寄存器	偏移地址	R/W	描述	复位值
SPIM_KEY1	SPIM_BA+0x3C	W	密钥1 寄存器	0x0000_0000

31	30	29	28	27	26	25	24
KEY1							
23	22	21	20	19	18	17	16
KEY1							
15	14	13	12	11	10	9	8
KEY1							
7	6	5	4	3	2	1	0
KEY1							

位	描述	
[31:0]	KEY1	**密钥1 寄存器** 该域是密码功能KEY1的数据 **注 1:**如果 KEY1(SPIM_KEY1[31:0]) 或 KEY2(SPIM_KEY2[31:0])什么都没有（KEY1 为 0x0000_0000 或 KEY2 是 0x0000_0000），密码功能会被自动禁用。 **注2:**当CIPHOFF(SPIM_CTL0[0])为0，KEY1(SPIM_KEY1[31:0])和 KEY2(SPIM_KEY2[31:0])都不为0x0000_0000(也就是 KEY1≠ 0x0000_0000 和 KEY2≠0x0000_0000)，使能密码加密和解密

密钥2寄存器(SPIM_KEY2)

寄存器	偏移地址	R/W	描述	复位值
SPIM_KEY2	SPIM_BA+0x40	W	密钥2寄存器	0x0000_0000

31	30	29	28	27	26	25	24
KEY2							
23	22	21	20	19	18	17	16
KEY2							
15	14	13	12	11	10	9	8
KEY2							
7	6	5	4	3	2	1	0
KEY2							

位	描述	
[31:0]	KEY2	**密钥2 寄存器** 该域是密码功能KEY2的数据 **注 1:**如果 KEY1(SPIM_KEY1[31:0]) 或 KEY2(SPIM_KEY2[31:0])什么都没有（KEY1 为 0x0000_0000 或 KEY2 是 0x0000_0000），密码功能会被自动禁用。 **注2:**当CIPHOFF(SPIM_CTL0[0])为0，KEY1(SPIM_KEY1[31:0])和 KEY2(SPIM_KEY2[31:0])都不为0x0000_0000(也就是KEY1≠ 0x0000_0000 和 KEY2≠0x0000_0000)，使能密码加密和解密

直接内存映射模式控制寄存器(SPIM_DMMCTL)

寄存器	偏移地址	R/W	描述	复位值
SPIM_DMMCTL	SPIM_BA+0x44	R/W	直接内存映射模式控制寄存器	0x0008_0000

31	30	29	28	27	26	25	24
ACTSCLKT				Reserved	UACTSCLK	CREN	BWEN
23	22	21	20	19	18	17	16
Reserved			DESELTIM				
15	14	13	12	11	10	9	8
CRMDAT							
7	6	5	4	3	2	1	0
Reserved							

位	描述	
[31:28]	**ACTSCLKT**	**SPI Flash 激活 SCLK 时间** **仅针对直接内存映射模式、DMA 写模式和 DMA 读模式** 该寄存器设置 SPIM SS 激活边沿和第一个串行 SPI 输出时钟边沿之间的时间间隔，见图 6. 22-11 **(1) ACTSCLKT = 0 (功能禁用) :** 时间间隔 = 1 AHB 时钟周期时间. **(2) ACTSCLKT ≠ 0 (功能使能) :** 时间间隔 = (ACTSCLKT + 3) * AHB 时钟周期时间 **注 1 :** AHB 时钟周期时间.= 1/AHB 时钟频率 **注 2 :** SCLK 是 SPI 输出时钟 **注 3 :**请参考所使用的 SPI flash 规格书去设置该寄存器的值，不同 SPI flash 厂商设置值可能不同
[27]	**Reserved**	保留
[26]	**UACTSCLK**	**用户设置 SPI Flash 激活 SCLK 时间** **仅针对直接内存映射模式、DMA 写模式和 DMA 读模式** 0 = 依据 DIVIDER(SPIM_CTL1[31:16]), 由硬件自动设 ACTSCLKT(SPIM_DMMCTL[31:28]) (默认值) 1 = 用户手动设置 ACTSCLKT(SPIM_DMMCTL[31:28]) 当用户想手动设置 ACTSCLKT(SPIM_DMMCTL[31:28]) 时, 请设置 UACTSCLK 为 1.
[25]	**CREN**	**连续读模式使能控制** **仅针对直接内存映射模式下读命令码为 0xBB, 0xEB, 0xE7, 0x0D, 0xBD, 0xED(注 2)** 0 =禁用连续读模式 (默认) 1 =使能连续读模式 对于 SPI flash 读操作，快速四线读 I/O(0xEB)命令、四线 I/O 字读(0xE7 在 winbond SPI flash)、快速读双线 I/O(0xBB)、DTR/DDR 快速读（0x0D）、DTR/DDR 快速读双线 I/O

		（0xBD）和 DTR/DDR 快速读四线 I/O(0xED)可以在输入地址数据之后通过设置连续读模式（8 位）来减少命令的开销。 **注:**当用户使用连续读模式功能并且设置 USETEN (SPIM_CTL2[16])为 1 时，CRMDAT(SPIM_DMMCTL[15:8])必须依据使用的 SPI flash 规格设置。当用户使用连续读模式功能并且设置 USETEN (SPIM_CTL2[16])为 0 时，CRMDAT(SPIM_DMMCTL[15:8])被默认设置为 WINBOND SPI flash 的值
[24]	**BWEN**	**16 字节突发包模式使能控制寄存器(默认值 = 0)** **仅针对 WINBOND SPI flash，直接内存映射模式，缓存使能和读命令码“0xEB 和 0xE7”** 0 = 禁用突发包模式 (默认) 1 = 使能突发包模式 在直接内存映射模式下，四线读命令“0xEB” 和 “0xE7”支持突发包模式的缓存应用和连续读模式(或性能增强模式)。对于缓存应用，突发包模式可以快速地被用来填充缓存线(在该 SPI flash 控制器下，我们使用缓存数据线大小是 16 个字节)。对于直接内存映射模式和缓存使能条件下的连续读模式(或性能增强模式)，缓存数据丢失，突发包模式可以快速的让 MCU 获得 SPI flash 数据请求。
[23:21]	**Reserved**	保留
[20:16]	**DESELTIM**	**SPI Flash 取消选择时间** **仅针对直接内存映射模式** 设置 SPI flash 取消时间的最小时宽（就是说 SPIM_SS 最小取消选择时间），见图 6.22-11. **(1)缓存功能禁用：** SPIM_SS 取消选择时间的最小时间宽度= (DESELTIM + 1) * AHB 时钟周期 **(2)缓存功能使能：** SPIM_SS 取消选择时间的最小时间宽度= (DESELTIM + 4) * AHB 时钟周期 **注 1：**AHB 时钟周期 = 1/AHB 时钟频率 **注 2：**当使能密码加密和解密时，请设置 DESELTIM(SPIM_DMMCTL[20:16]) >= 0x10。当禁用密码加密和解密时，请设置 DESELTIM(SPIM_DMMCTL[20:16]) >= 0x8。 **注 3：**请参考所使用的 SPI flash 规格书去设置该寄存器的值，不同 SPI flash 厂商设置值可能不同
[15:8]	**CRMDAT**	**连续读模式（或性能增强模式）的模式位数据（默认值=0）** **仅针对直接内存映射模式** 针对连续读模式（性能增强模式）的模式位数据设置 当我们设置该模式位**(注 1)**和设置 CREN(SPIM_DMMCTL[25])时，可以减少 8 个命令相位时钟，在 SPIM_SS 被激活后可以立刻读取地址 **注 1：**请参考所使用的 SPI flash 规格书去设置该寄存器的值，不同 SPI flash 厂商设置值可能不同 **注 2：**CRMDAT 需要配合 CREN(SPIM_DMMCTL[25])使用
[7:0]	**Reserved**	保留

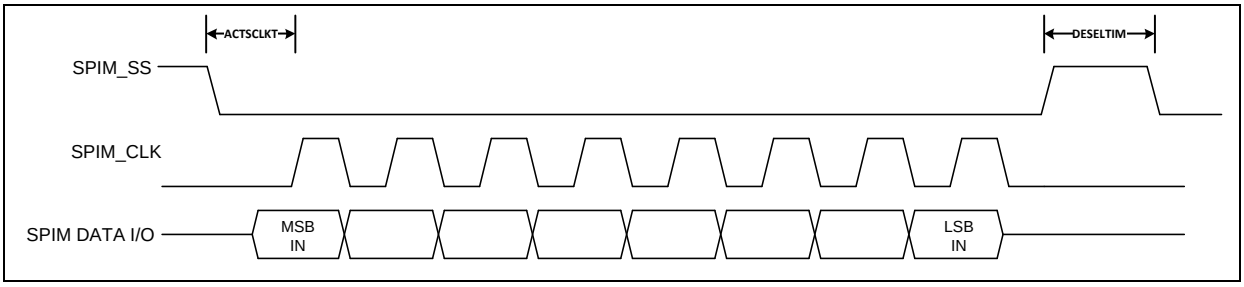


图 6.22-11SPIM_SS 激活建立时间与 SPIM_CLK 的关联和 SPIM_SS 取消选择时间

控制寄存器2 (SPIM_CTL2)

寄存器	偏移地址	R/W	描述	复位值
SPIM_CTL2	SPIM_BA+0x48	R/W	控制寄存器 2	0x0804_0000

31	30	29	28	27	26	25	24
Reserved			DCNUM				
23	22	21	20	19	18	17	16
Reserved			DTRMPOFF	Reserved			USETEN
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved							

位	描述	
[31:29]	**Reserved**	保留.
[28:24]	**DCNUM**	**空闲周期（Dummy Cycle）数** **仅针对直接内存映射模式和 DMA 读模式(注 1)** 设置空闲周期数 **(1)对于非 DTR/非 DDR 命令码 0x03, 0x0B, 0x3B, 0xBB, 0xEB, 和 0xE7 :** 当读命令码不需要任何空闲周期时（就是空闲周期数=0），用户必须设置 DCNUM 为 0x0. 对于命令码 0xBB，如果模式周期数（或性能增强周期数）和空闲周期数同时不为 0，用户必须依据使用的 SPI flash 规格设置 DCNUM 为**“模式周期数 + 空闲周期数”** 对于命令码 0xBB，如果只有空闲周期数（也就是空闲周期≠0x0 和模式周期数=0x0（或性能增强周期数=0x0）），用户必须依据使用的 SPI flash 规格设置 DCNUM 为**“空闲周期数”** 对于命令码 0x0B, 0x3B, 0xEB, 和 0xE7，用户只需依据使用的 SPI flash 规格设置 DCNUM 为**“空闲周期数”** **(2) 对于 DTR/DDR 命令码 0x0D, 0xBD, 和 0xED :** 用户只需依据使用的 SPI flash 规格设置 DCNUM 为**“空闲周期数”**和**DTRMPOFF(SPIM_CTL2[20])** **注 1 :** 空闲周期数取决于 SPI 输出的时钟频率、SPI flash 厂商和读命令类型。请参考所使用的 SPI flash 规格书去设置该寄存器的值。
[23:21]	**Reserved**	保留
[20]	**DTRMPOFF**	**对于 DTR/DDR 命令码为 0x0D, 0xBD, 和 0xED 的模式相位关闭** **仅针对直接内存映射模式和 DMA 读模式(注 1)** 0 =在 DTR/DDR 读命令码为 0x0D, 0xBD, 和 0xED 时，模式周期数（或性能增强周期数）不等于 0x0 1 =在 DTR/DDR 读命令码为 0x0D, 0xBD, 和 0xED 时，模式周期数（或性能增强周期数）等于 0x0 **注 1 :** 对于 DTR/DDR 命令码为 0x0D, 0xBD, 和 0xED，请参考所使用的 SPI flash 规格书去了解模式周期数（或性能增加周期数）

[19:17]	**Reserved**	保留
[16]	**USETEN**	**用户设置值使能控制** **仅针对于直接内存映射模式和读命令为 0x03,0x0B,0x3B,0xBB,0xEB,0xE7 的 DMA 读模式** 0 = SPI flash 控制器的硬件电路会使用一下 DCNUM(SPIM_CTL2[28:24]) 和 CRMDAT(SPIM_DMMCTL[15:8])的默认值去自动配置 SPI flash 操作 空闲周期数 (DCNUM) : 读命令为 0x03 的空闲周期数: 0x0 读命令为 0x0B 的空闲周期数: 0x8 读命令为 0x3B 的空闲周期数: 0x8 读命令为 0xBB 的空闲周期数: 0x0 读命令为 0xEB 的空闲周期数: 0x4 读命令为 0xE7 的空闲周期数: 0x2 连续读模式的模式位数据(CRMDAT) : 0x20 1 = 如果 DCNUM(SPIM_CTL2[28:24]) 和 CRMDAT(SPIM_DMMCTL[15:8])没有设置为以上的默认值，用户必须设置 USETEN 为 1、DCNUM(SPIM_CTL2[28:24]) 和 CRMDAT(SPIM_DMMCTL[15:8])手动配置 SPI flash 操作 对于 DTR/DDR 命令码为 0x0D, 0xBD, 和 0xED，请设置 USETEN 为 1
[15:0]	**Reserved**	保留

6.23 I²C接口

6.23.1 概述

I²C为双线，双向串行总线，通过简单有效的连线方式实现器件间的数据交换。I²C标准是多主机总线，包括冲突检测和仲裁，以防止在两个或多个主机同时尝试控制总线时发生数据损坏。

有两组I²C控制器，都支持掉电唤醒功能。

6.23.2 特性

I²C通过SDA 及SCL两条线与连接在总线上的器件传输信息，总线的主要特征有：

- 支持最多三组I²C接口
- 支持主机/从机模式
- 主从机之间双向数据传输
- 总线支持多主机(无中心主机)
- 支持高速3.4 Mbps模式
- 支持标准模式(100 kbps)，快速模式(400 kbps)和快速加模式(1 Mbps)
- 多主机间同时传输数据仲裁，避免总线上串行数据损坏
- 总线采用串行同步时钟，可实现设备之间以不同的速率传输
- 串行同步时钟被用作握手机制去挂起和恢复串行传输
- 内建14位溢出定时器，当I²C总线中止且定时器溢出时，产生I²C中断
- 可配置不同时钟以适用于可变速率控制
- 支持7位地址模式和10位地址模式
- 支持多地址识别（4组从机地址带mask选项）
- 支持掉电唤醒功能
- 支持带有一个缓冲能力的PDMA
- 支持建立/保持时间可编程
- 支持总线管理(SM/PM兼容)功能

6.23.3 框图

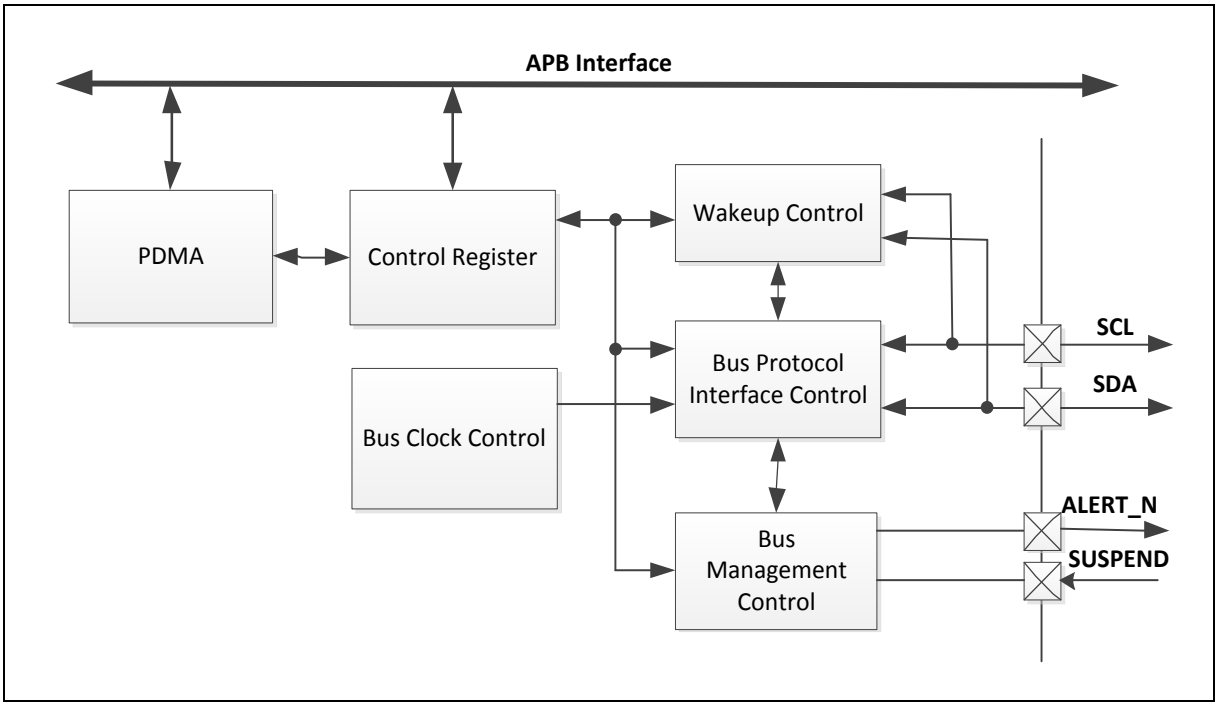


图 6.23-1 I²C 控制器框图

6.23.4 基本配置

6.23.4.1 I²C0 基本配置

- 时钟源配置
  - 使能I²C0时钟控制位I²C0CKEN (CLK_APBCLK0[8])。
- 复位配置
  - 复位I²C0控制器的控制位I²C0RST (SYS_IPRST1[8])。
- 管脚配置

组	管脚名称	GPIO	MFP
I²C0	I²C0_SCL	PC.12, PD.7, PE.13, PF.3, PG.0	MFP4
		PB.5, PH.2	MFP6
		PA.5, PB.9, PC.1	MFP9
	I²C0_SDA	PC.8, PC.11, PD.6, PF.2, PG.1	MFP4
		PB.4, PH.3	MFP6
		PA.4, PB.8, PC.0	MFP9
	I²C0_SMBAL	PG.2	MFP4
		PC.3	MFP9
		PA.3	MFP10

	I²C0_SMBSUS	PG.3	MFP4
		PC.2	MFP9
		PA.2	MFP10

6.23.4.2 I²C1 基本配置

- 时钟源配置
  - 使能I²C1时钟控制位I²C1CKEN (CLK_APBCLK0[9])。
- 复位配置
  - 复位I²C1控制器的控制位I²C1RST (SYS_IPRST1[9])。
- 管脚配置

组	管脚名称	GPIO	MFP
I²C1	I²C1_SCL	PF.0	MFP3
		PA.12, PD.5	MFP4
		PG.2	MFP5
		PB.11	MFP7
		PA.7, PE.1	MFP8
		PA.3, PB.1, PC.5	MFP9
		PB.3	MFP12
	I²C1_SDA	PF.1	MFP3
		PA.13, PD.4	MFP4
		PG.3	MFP5
		PB.10	MFP7
		PA.6, PE.0	MFP8
		PA.2, PB.0, PC.4	MFP9
		PB.2	MFP12
	I²C1_SMBAL	PG.0	MFP5
		PB.9	MFP7
		PC.7, PH.8	MFP8
	I²C1_SMBSUS	PG.1	MFP5
		PB.8	MFP7
		PC.6, PH.9	MFP8

6.23.4.3 I²C2基本配置

- 时钟源配置
  - 使能I²C2时钟控制位I²C2CKEN (CLK_APBCLK0[10])。
- 复位配置

- 复位I²C2控制器的控制位I²C2RST (SYS_IPRST1[10])。

- 管脚配置

组	管脚名称	GPIO	MFP
I²C2	I²C2_SCL	PD.9	MFP3
		PA.14, PD.1	MFP6
		PA.11	MFP7
		PB.13	MFP8
		PA.1, PH.8	MFP9
	I²C2_SDA	PD.8	MFP3
		PA.15, PD.0	MFP6
		PA.10	MFP7
		PB.12	MFP8
		PA.0, PH.9	MFP9
	I²C2_SMBAL	PB.15	MFP8
	I²C2_SMBSUS	PB.14	MFP8

6.23.5 功能描述

在I²C总线上，数据通过时钟线SCL和数据线SDA在主从机间逐一字节同步传送。每个字节数据长度是8位。一个SCL时钟脉冲传输一个数据位，数据由最高位MSB开始传输，每个字节传输后跟随一个应答位，每个位在SCL为高时采样；因此，SDA 线可能在SCL为低时改变，但在SCL 为高时必须保持稳定。当SCL为高时，SDA线上的跳变视为一个命令(START或STOP)。更多关于I²C总线时序的细节请参考图6.23-2。

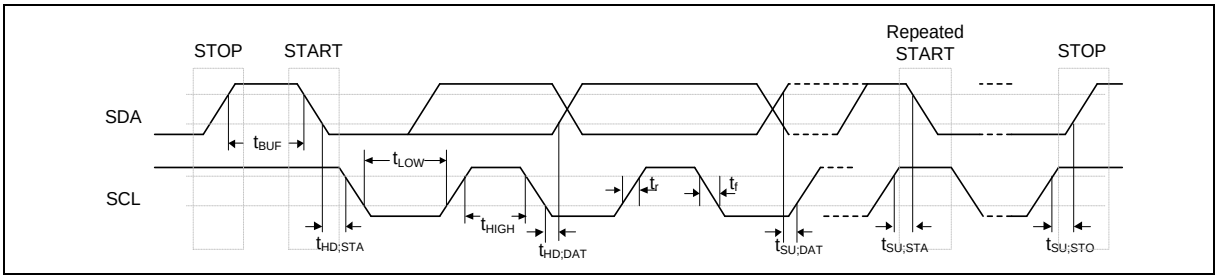


图 6.23-2 I²C 总线时序

片上I²C外设提供了一个符合I²C总线规范的串行接口。I²C端口自动处理字节传输。通过设置寄存器I2C_CTL0的I2CEN为'1'可使能该端口。I²C硬件接口通过数据线SDA 和时钟线 SCL两个管脚连到I²C总线。当I/O 管脚作为I²C 端口使用时，用户必须事先设定I/O 管脚为I²C功能。

注:这个两个管脚在I²C状态下为开漏脚，SDA和SCL两个管脚需要上拉电阻。

6.23.5.1 I²C 协议

标准I²C 协议如下图6.23-3，通常标准通讯有以下4部分:

- 起始(START)或者重复起始信号(Repeated START)
- 从机地址传输和R/W位传输
- 数据传输
- 停止信号(STOP)

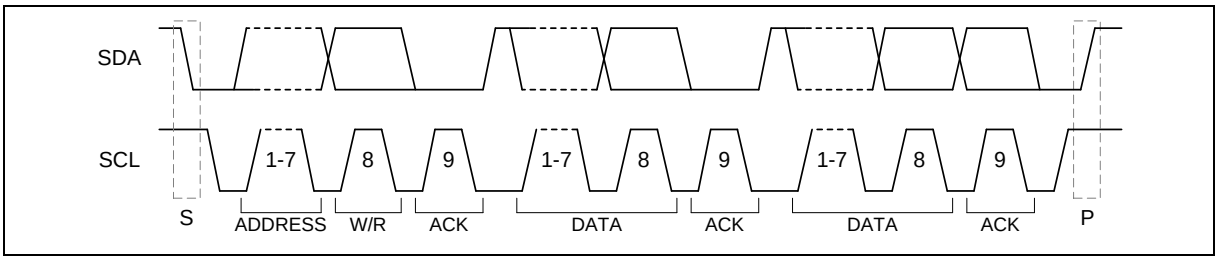


图 6.23-3 I^2C 协议

- 起始或重复起始信号

  当总线处于释放/空闲状态下，说明没有主机设备占用总线（SDA与SCL线同时为高），主机可以通过发送起始(START)信号来发起传输过程。起始信号，通常表示为"S"位，当SCL线为高时，SDA线上信号由高至低变化，就被定义为起始信号。起始信号表示一个新的数据传输的开始。

  主机发送完地址字节（地址和读/写位）后，可以发送任何数量的字节数据并带一个停止信号。也可以用另一个起始信号替代停止信号，随后是地址(包含读写位)和更多的数据。这个起始信号叫做重复起始。这个定义可以用来发送任意个起始信号，它的目的是在不释放总线情况下，对一个或多个设备能读/写操作，而不让操作被打断。用这个方法主机跟其他从机或同一个从机在不同方向传输（例如，写设备到读设备）不用释放总线。

- 停止信号

  主机可以通过产生一个停止信号来终止数据通信。停止信号，通常表示为P位，当SCL 线为高时，SDA线上信号由低至高变化，就被定义为停止信号。

  图6.23-4为起始，重复起始和停止信号的波形。

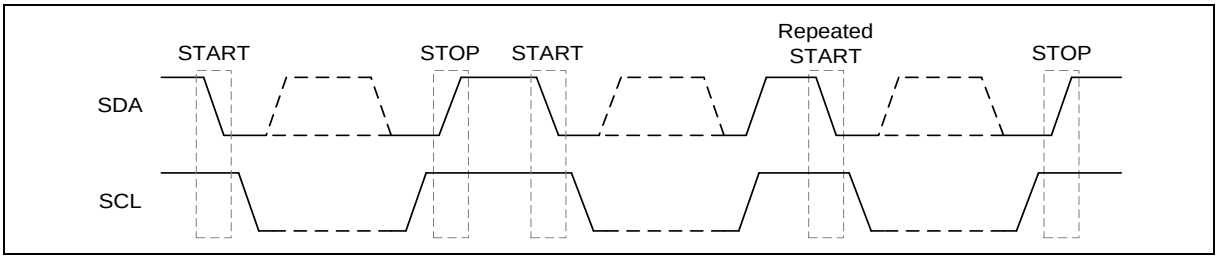


图 6.23-4 起始和停止状况

- 从地址传输

  在一个起始（或重复起始）信号后，主机发送一个从机地址来确认通信的目标设备。起始地址有一个或两个字节组成（针对7位或10位地址情况）。地址字节过后，从机会对传输的地址作应答反应。

  因此，从机地址可以被编程并且与接收到的地址比较。一旦匹配，从机会回一个应答(SDA = 0).如果与目标不匹配则无应答(SDA = 1)。除了编程地址匹配之外，如果从机有能力处理

类似的请求，那么另外的地址值也必须有应答回应。

- 数据传输

  当从机设备地址和R/W位被成功识别到，就可以根据R/W 位所决定的方向按一字节一字节方式进行数据传输。每个字节传输完后，紧接着的第9个SCL时钟周期会有一个应答信号位。如果从机上产生无应答信号(NACK)，主机可以产生停止信号来中止数据传输或者产生重复起始信号开始新一轮数据传输。

  当主机作为接收设备时，发生无应答信号(NACK)，则从机将释放SDA线，让主机产生停止信号或重复起始信号。图6.23-5和图6.23-6展示了传输和应答位的波形

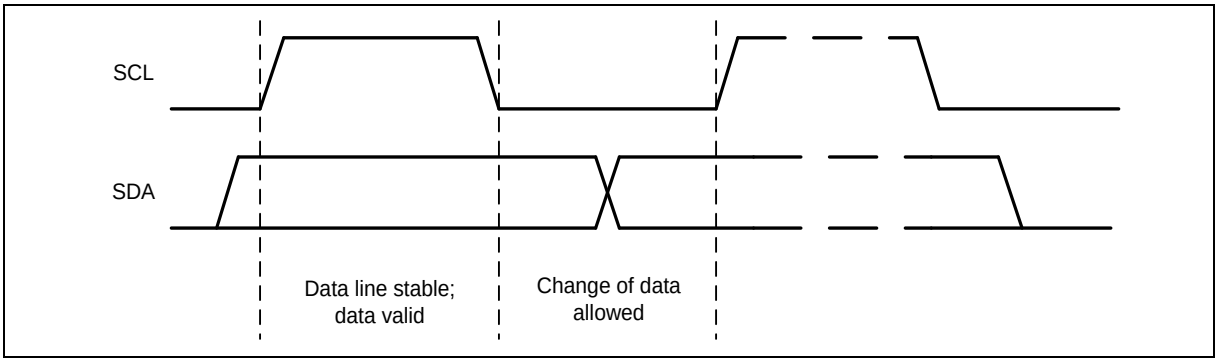


图 6.23-5 I²C 总线上的位传输

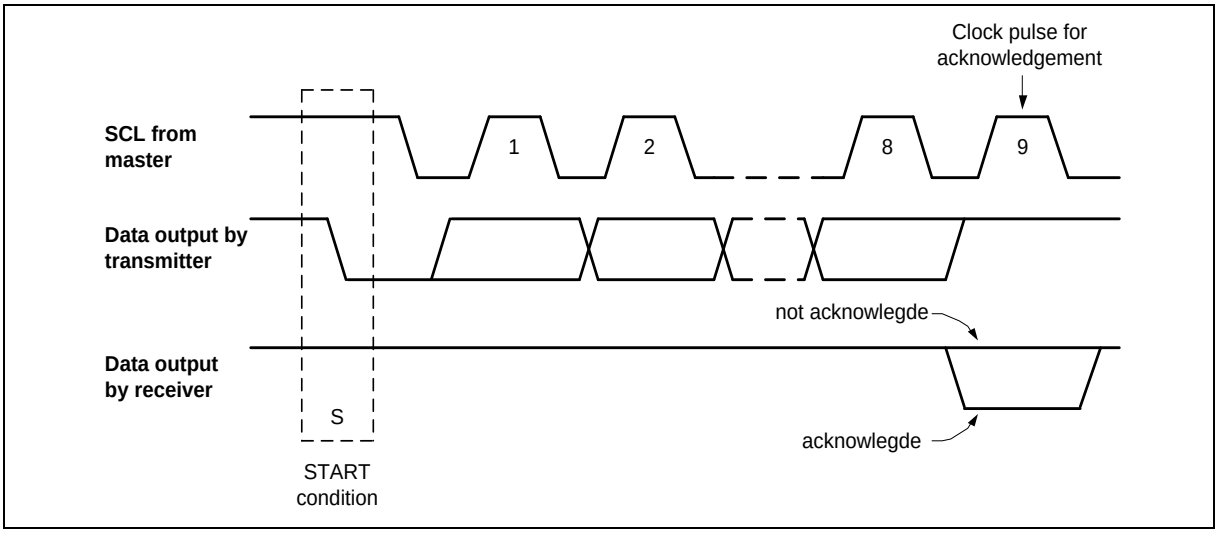


图 6.23-6 I²C 总线上的应答信号

- I²C总线上的数据传输

  图6.23-7表示7位地址情况下主机向从机传输数据。主机发出一个7-位地址和1-位写指示,表示主机想要传送数据给从机。从机回复应答给主机之后，主机继续传输数据。

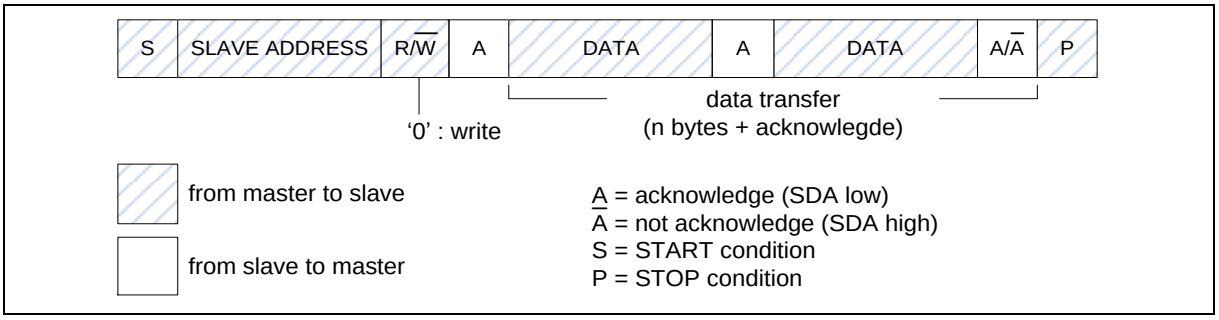


图 6.23-77 位地址情况下主机向从机传输数据

图6.23-8表示7位地址情况下主机向从机读取数据。主机发7-位地址寻址和1-位读指示，表示主机要向从机读取数据，从机返回应答给主机后，就开始给主机传输数据。

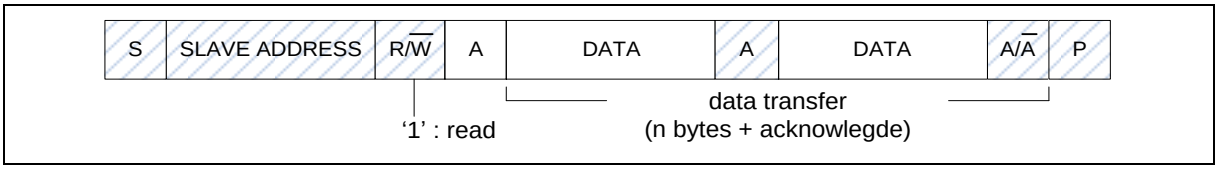


图 6.23-87 位地址情况下主机向从机读取数据

图6.23-9表示10位地址情况下主机向从机传输数据。主机发送10位地址。首字节是有10位地址指示(5'b11110)和2位地址再加上写指示，第二字节是剩下的8位地址。在第二字节之后，主机继续传输数据。注意7位和10位地址设备可以工作在同一总线上。

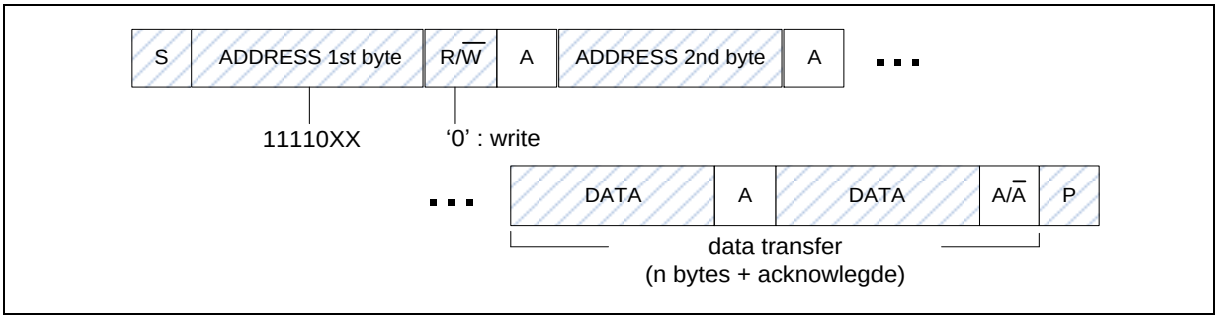


图 6.23-910 位地址情况下主机向从机传输数据

图6.23-10表示10位地址情况下主机向从机读取数据。首先主机发送10位地址给从机，在主机传输带有读指示的第一个字节之后。在带有读指示的第一个字节之后，从机就开始给主机传输数据

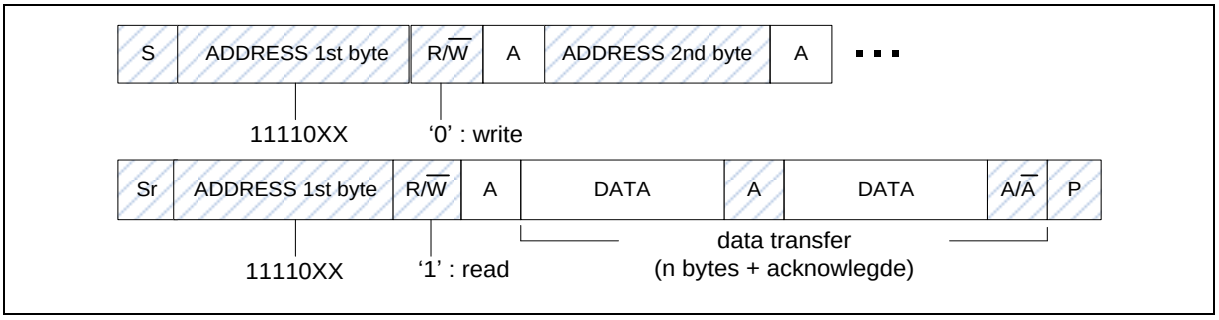


图 6.23-1010 位地址情况下主机向从机读取数据

6.23.5.2 操作模式

片上I²C端口支持三种工作模式：主机模式，从机模式和广播呼叫模式

应用中，I²C 端口可以作为主机和从机。在从机模式，I²C端口硬件会查找自身从机地址和广播呼叫地址，如果这两个地址的任一个被检测到，并且从机想要从主机接收或向主机发送数据(通过设置AA位)，应答脉冲将会在第9个时钟发出，此时，如果中断使能，则主机和从机设备上都会发生一次中断请求。在微控制器想要成为总线主机时，在进入主机模式之前，硬件需等待到总线空闲，以保证合理的从机动作不会被打断。在主机模式下，如果总线仲裁丢失，I²C端口立即切换到从机模式，并可以在同一次串行传输过程中检测自身。

为控制I²C总线的各种模式传输，用户需要按照寄存器I²C_STATUS0的当前状态码来设置I²C_CTL0，I²C_DAT寄存器。换句话说，每个I²C总线动作都要检查I²C_STATUS0寄存器的当前状态，然后再设置I²C_CTL0，I²C_DAT寄存器执行总线动作。最后，通过I²C_STATUS0检查响应状态。

在寄存器I²C_CTL0 [3]的SI标志清除后，寄存器I²C_CTL0的这些位STA，STO和AA用来控制I²C硬件的下一个状态。当完成一个新的动作，I²C_STATUS0的状态代码将被更新，I²C_CTL0寄存器的SI标志将被设置。如果I²C中断控制位INTEN (I²C_CTL0[7])被设置，新状态代码对应的动作或者软件将在中断服务程序中被执行。

图 6.23-11显示当前I²C状态码是0x08，然后通过设置I²C_DATA=SLA+W和(STA,STO,SI,AA) = (0,0,1,x)发送地址到I²C总线。如果总线上有从机匹配相应的地址且返回ACK，I²C_STATUS0状态码更新为0x18。

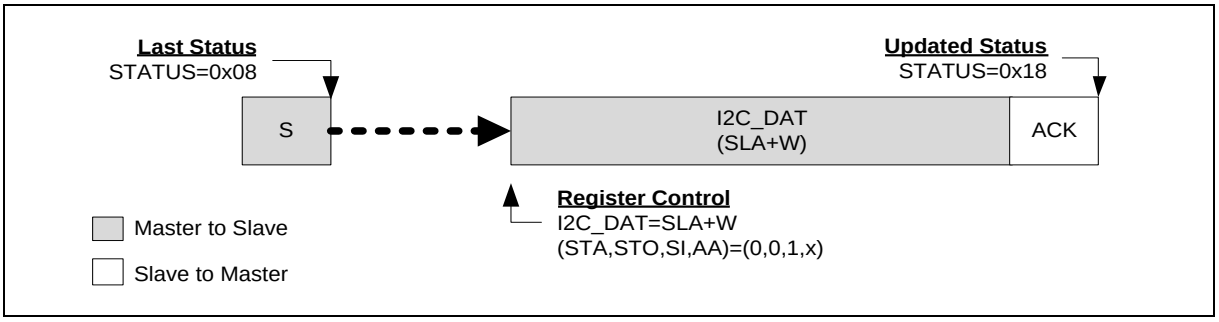


图 6.23-11 通过当前状态控制 I²C 总线

主机模式

图6.23-12和图6.23-13是I²C主机模式所有可能的协议展示。用户需要遵循恰当的流程来实现I²C协议。

换句话说，用户可以发送一个起始信号到总线，I²C总线将设置成主机传输(MT)模式(图6.23-12图6.23-12)，或主机接收(MR)模式(图6.23-13)。起始信号设置成功后新的状态码将是0x08。起始信号后，用户可以发送从机地址，读/写位，数据和重复起始，停止来执行I²C协议。

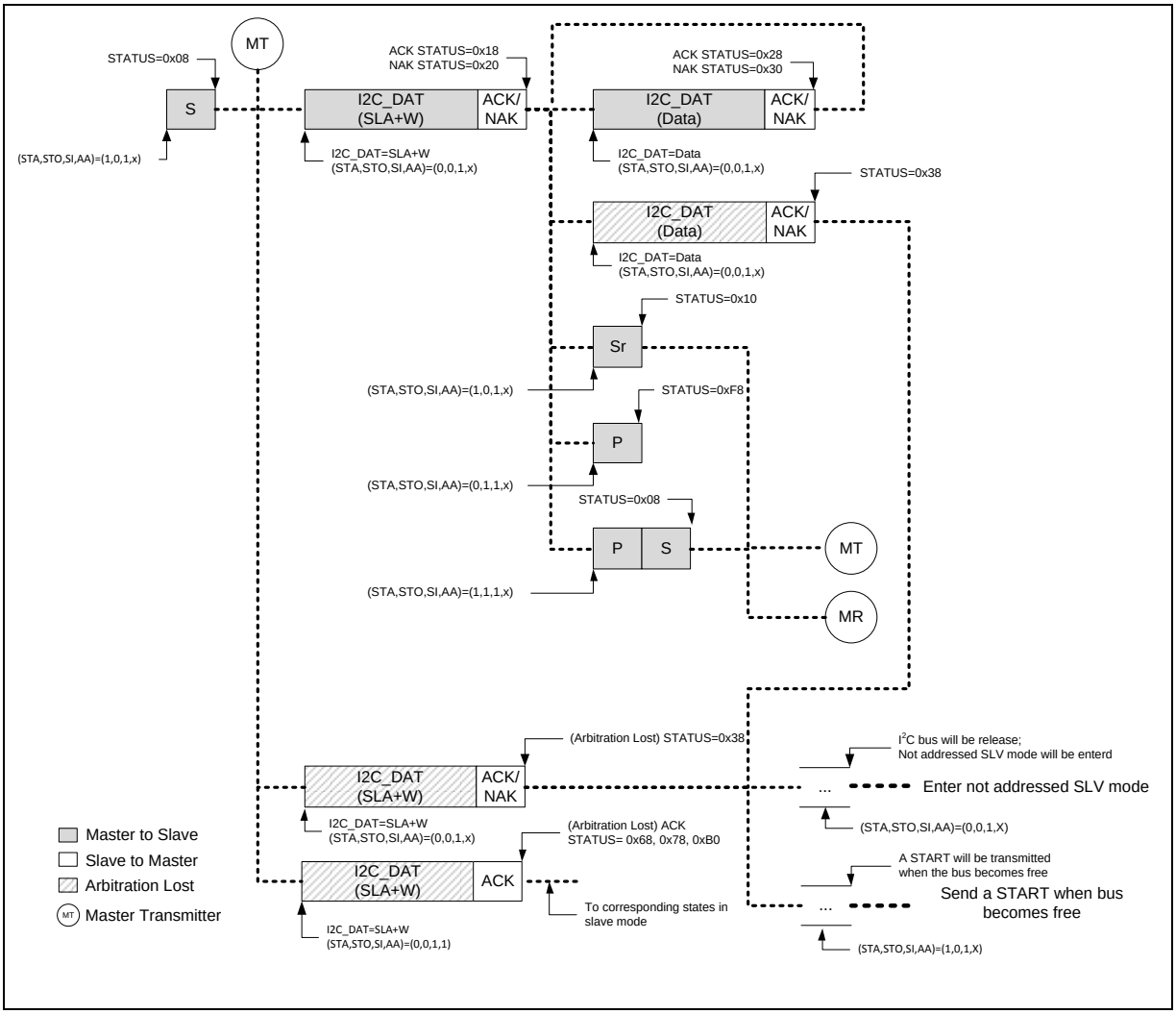


图 6.23-12 主机发送模式控制流程

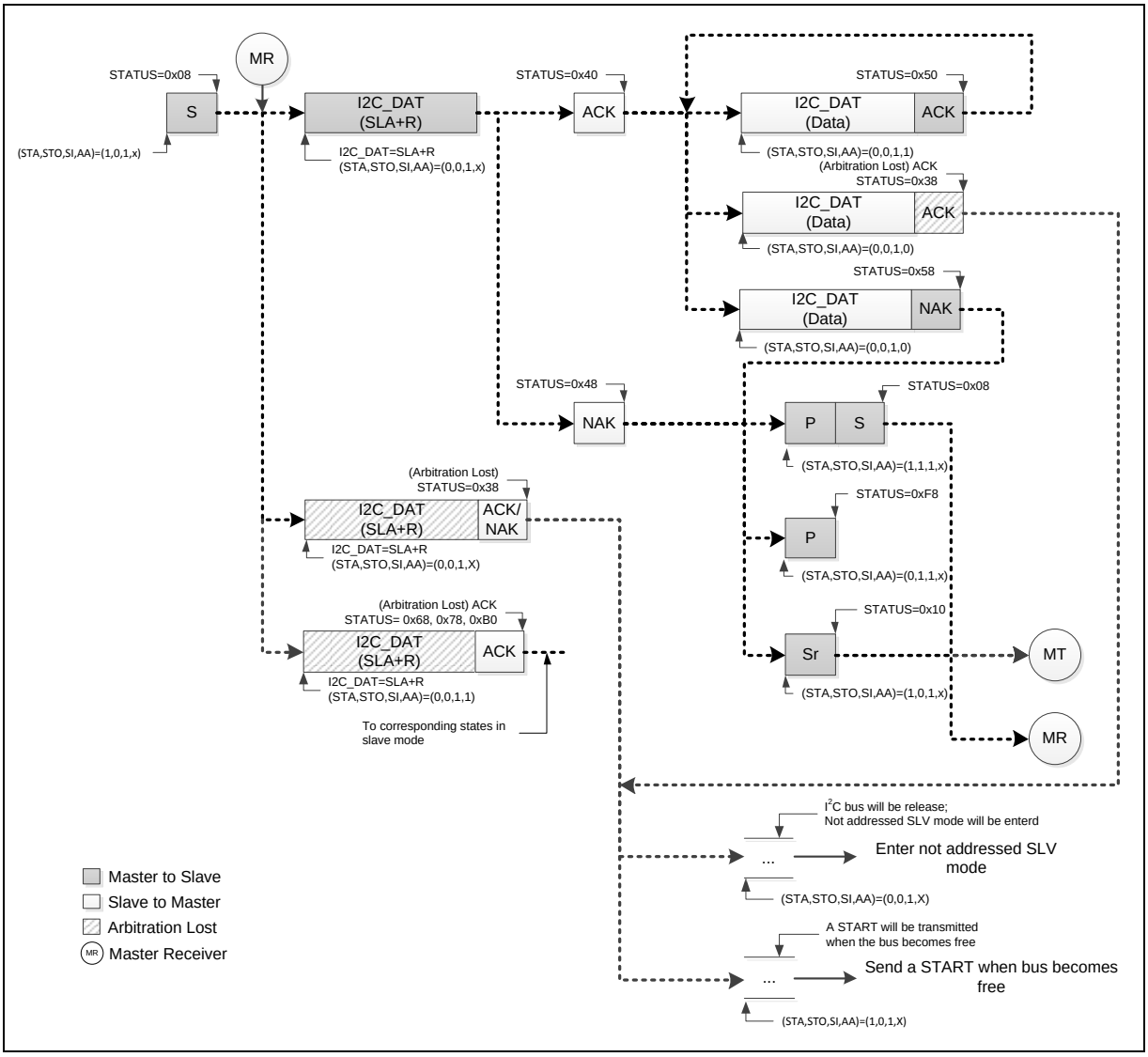


图 6.23-13 主机接收模式控制流程

如果I²C在主机模式并且仲裁丢失，状态码将置为0x38。在状态为0x38时，当总线空闲时，用户可以设置(STA, STO, SI, AA) = (1, 0, 1, X)发送起始信号来重新开始主机操作。否则，用户可以设置(STA, STO, SI, AA) = (0, 0, 1, X)来释放总线，并进入无地址从机模式。

从机模式

复位后默认情况下，I²C不会被寻址，并且不会识别I²C总线上的地址。用户可以通过I2C_ADDRn(n=0~3)设置从机地址和设置 (STA, STO, SI, AA) = (0, 0, 1, 1)来让I²C识别主机发送的地址。图6.23-14所示为I²C从机模式的所有可能的流程。用户需要遵循（图6.23-14）的流程来实现他们的I²C协议。

如果在主机模式总线仲裁丢失，I²C端口立即切换到从机模式，并且在同一串行传输中识别自有的从机地址。如果在仲裁丢失后识别到地址是SLA+W（主机想写数据到从机），状态码是0x68。如果在仲裁丢失后识别到地址是SLA+R（主机向从机读数据），状态码是0xB0。

注:I²C通信期间，在从机模式下，当对SI标志写'1'清除时，SCL 时钟将被释放。

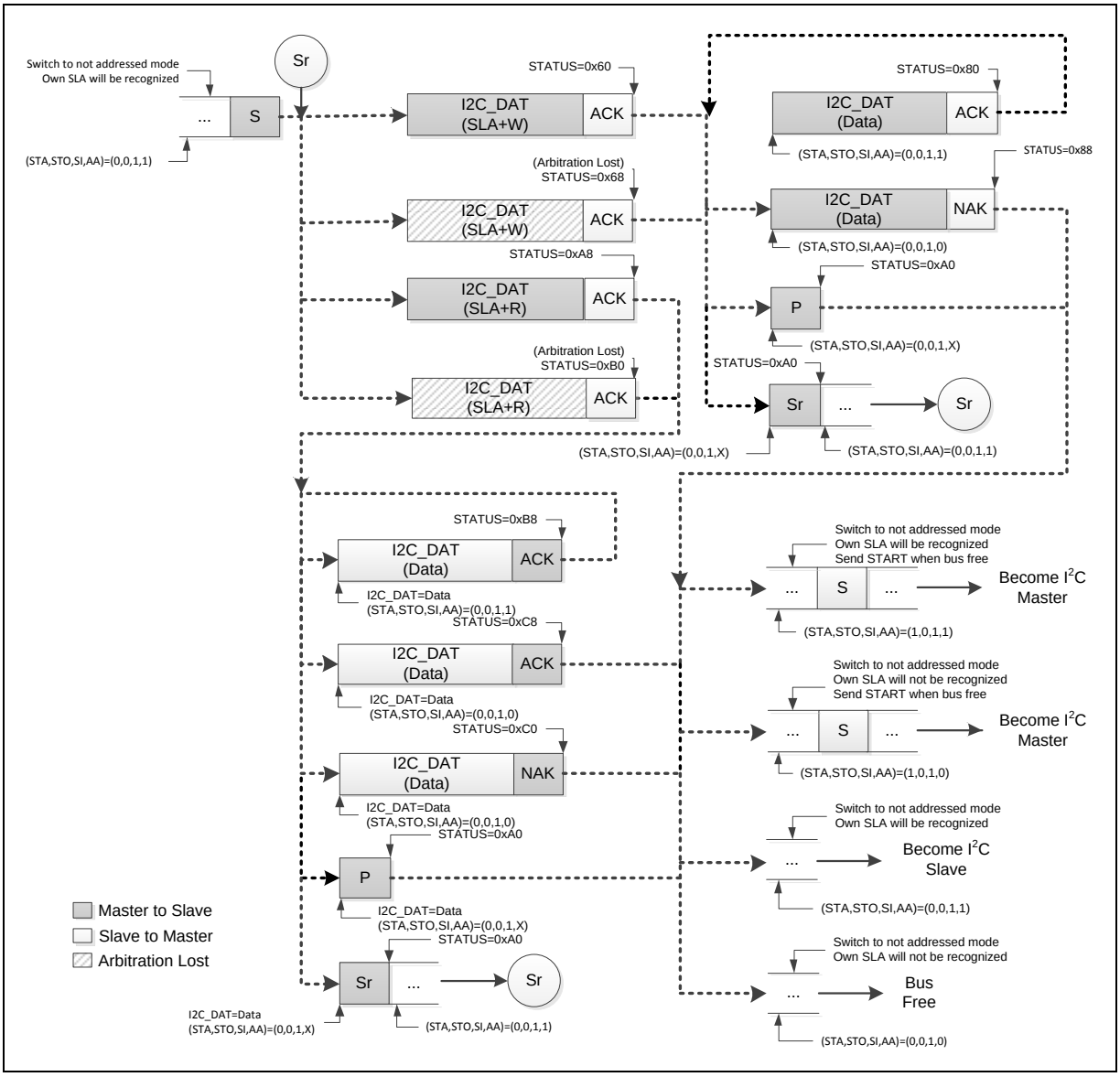


图 6.23-14 从机模式控制流程

如果I²C在可寻址从机模式接收数据时，收到停止或重复起始信号，状态码是0xA0。当状态码是0xA0时，用户可以遵循如上图状态码是0x88的操作。

如果I²C在可寻址从机模式传输数据时却收到停止或重复起始信号，状态码是0xA0。当状态码是0xA0时，用户可以遵循如上图状态码是0xC8的操作。

注:从机获得0x88, 0xC8, 0xC0 和0xA0状态后，从机可以切换到无地址模式，自身SLA不会被辨识。如果进入这种状态，从机不再接收主机任何信号或地址。在这种状态，I²C需要进入空闲模式。

广播呼叫模式(GC)

如果GC位(I2C_ADDRn [0])被置位，I²C端口硬件将响应广播呼叫地址(0x00).用户可以通过清GC位来禁

止广播呼叫功能。当I²C在从机模式时且GC位被设置，可以接收主机地址(0x00)的广播呼叫，将遵循广播模式状态。

当GC模式时地址匹配可以唤醒。注意默认地址是0x00，但是用户必须设置一个非0x00的地址

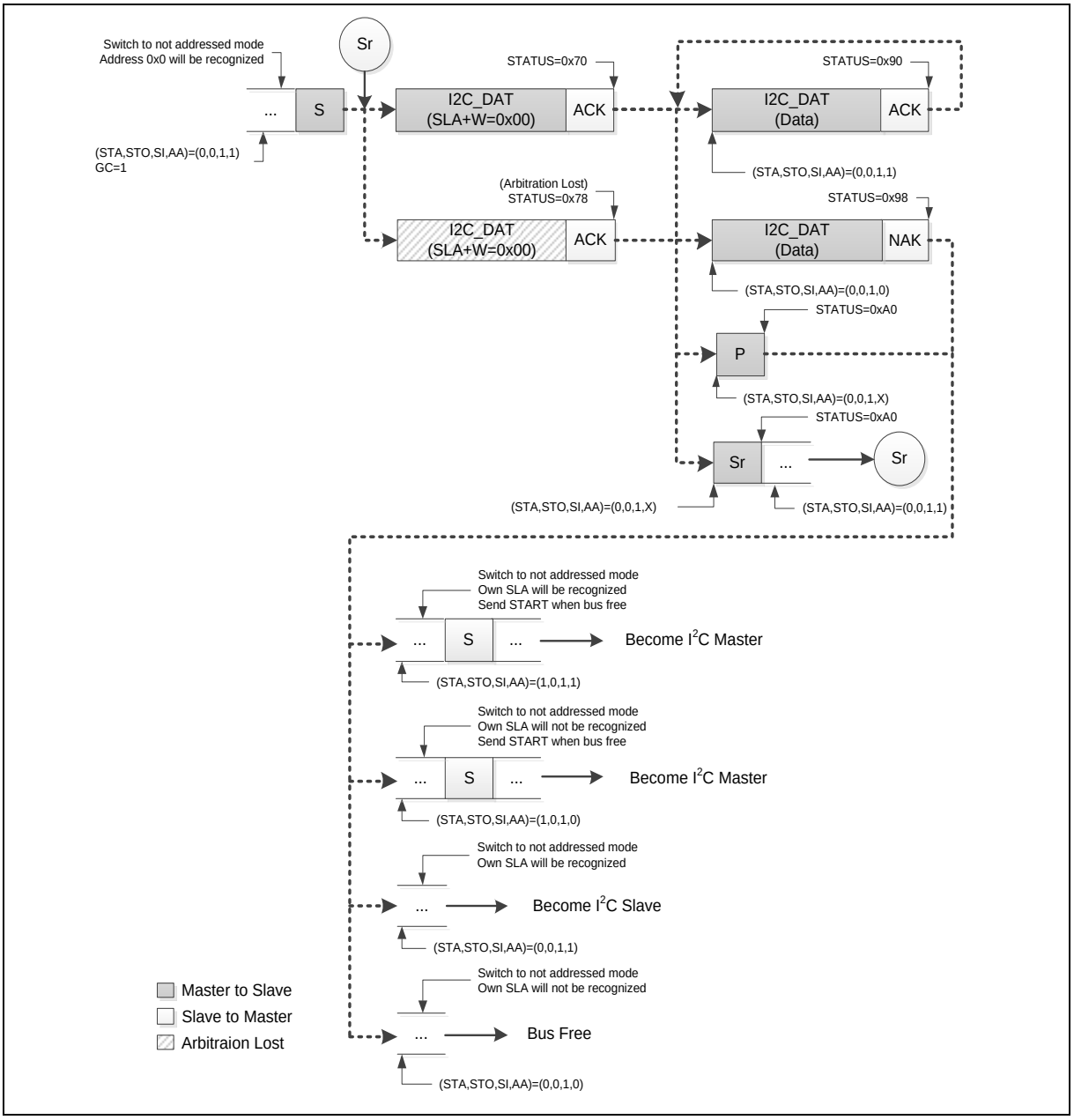


图 6.23-15 GC 模式

如果I²C在广播呼叫模式接收数据时，却收到停止或重复起始信号，状态码是0xA0。当状态码是0xA0时，用户可以遵循如上图状态码是0x98的流程处理。

注:从机获得0x98和0xA0状态后，从机会切换到无地址模式并且自身SLA将不被辨识。如果进入这种状态，从机不再接收主机的任何信号或地址。I²C需要进入空闲模式。

多主机模式

在一些应用中，一个I²C总线上有多个主机同时访问从机，并有可能同时在传送数据。I²C是支持多主机模式，并包含有冲突检测和仲裁，防止数据损坏。

如果两个主机同时发命令，通过仲裁来决定哪个优先并继续发命令。仲裁是在SCL为高时在SDA上执行的。每一个主机都会检测总线上的SDA信号是否符合它产生的SDA信号。如果检测到总线上SDA为低，但应该为高，则这个主机将失去仲裁。设备在仲裁丢失后会产生SCL脉冲直到本字节结束，然后释放总线进入从机模式。仲裁可以一直进行到所有数据被传输完。这样意味着多主机系统中主机必须监控总线和做相应的处理。

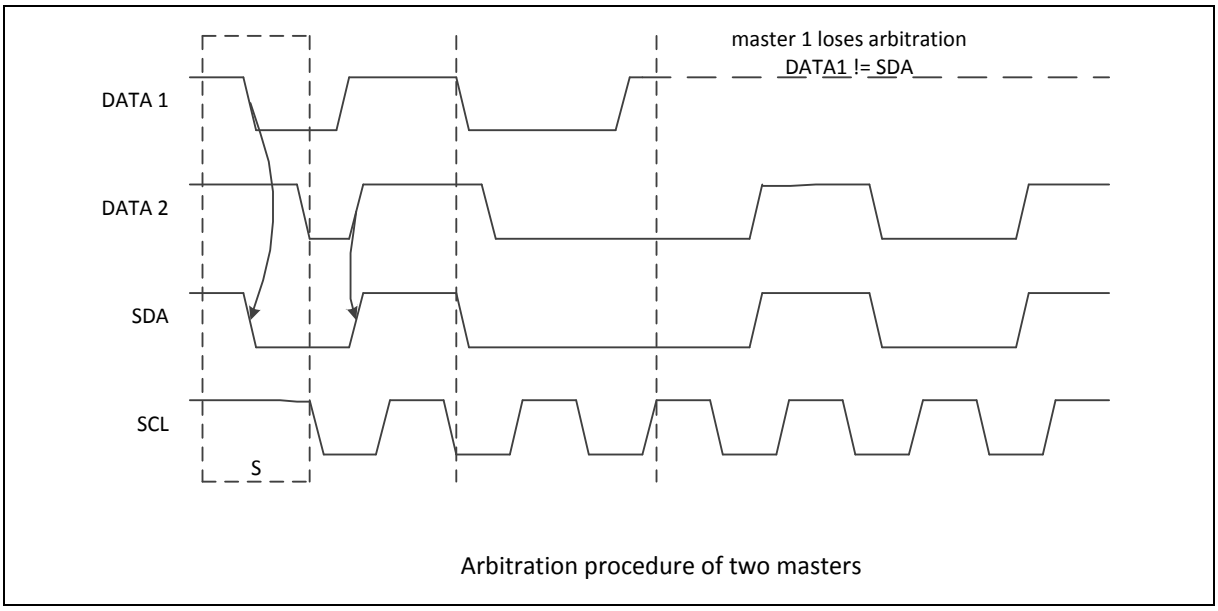


图 6.23-16 仲裁丢失

- 当I²C_STATUS0 = 0x38，代表接收到一个仲裁丢失.仲裁丢失事件可能发生在发送起始位，数据位或者停止位期间。用户可以在总线空闲时设置(STA, STO, SI, AA) = (1, 0, 1,X)来再次发送起始信号或者设置(STA, STO, SI, AA) = (0, 0, 1, X)发送停止信号来返回到无地址从机模式。用户可以通过ONBUSY (I2C_STATUS1 [8])侦测总线被释放。
- 当I²C_STATUS0 = 0x00，接收到一个“总线错误”，为了从错误总线恢复到I²C总线， STO应被设置且SI应将被清0，然后对STO清0来释放总线。
  - 设置(STA, STO, SI, AA) = (0, 1, 1, X)停止当前传输。
  - 设置(STA, STO, SI, AA) = (0, 0, 1, X)释放总线。

总线管理 (SMBus/PMBus 兼容)

该部分相关内容只适合支持总线管理特性的I²C接口。

简介

总线管理是个I²C接口，通过总线管理各种设备可以相互通信以及与系统的其它部分通信。它是基于I²C的操作规范。总线管理为系统和电源管理相关任务提供控制总线。

该外设与SMBUS规范2.0版本(http://smbus.org/specs/)以及PMBUS规范1.2版(http://pmbus.org/)相兼容。

系统管理总线规范涉及到3个类型的设备。

- 从设备，可以接收数据或响应命令。
- 主设备，可以产生命令，产生时钟和结束传输。
- 特定主设备的主机，可以给系统CPU提供主接口，主机必须是主-从设备而且必须支持SMbus主机识别协议。系统里只允许有一个主机。

总线管理设备基于I^2C规范2.1版。

设备识别 – 从机地址

任何存在总线管理的从机设备都有一个唯一的地址称为从机地址。下面的地址为保留而且禁止使用或者赋予任何总线管理设备。(详细请参看SMBus规范)。

从设备地址 位 7-1	读/写位 位 0	说明
0000 000	0	广播呼叫地址
0000 000	1	起始字节
0000 001	X	CBUS 地址
0000 010	X	不同总线格式保留地址
0000 011	X	保留将来使用
0000 1XX	X	保留将来使用
0101 000	X	保留访问总线主机
0110 111	X	保留访问总线默认地址
1111 0XX	X	10位从机寻址
1111 1XX	X	保留将来使用
0001 000	X	SMBus 主机
0001 100	X	SMBus警报相应地址
1100 001	X	SMBus设备默认地址

表 6.23-1 SMBus 保留地址

总线协议

任何设备都可能有11种命令协议。设备可以使用11种协议中某一或者所有来通信。这些协议为**Quick Command**, **Send Byte**, **Receive Byte**, **Write Byte**, **Write Word**, **Read Byte**, **Read Word**, **Process Call**, **Block Read**, **Block Write**和**Block Write-Block Read Process Call**。这些协议必须由软件来执行。(协议的更详细内容请参看SMBus规范2.0版本)

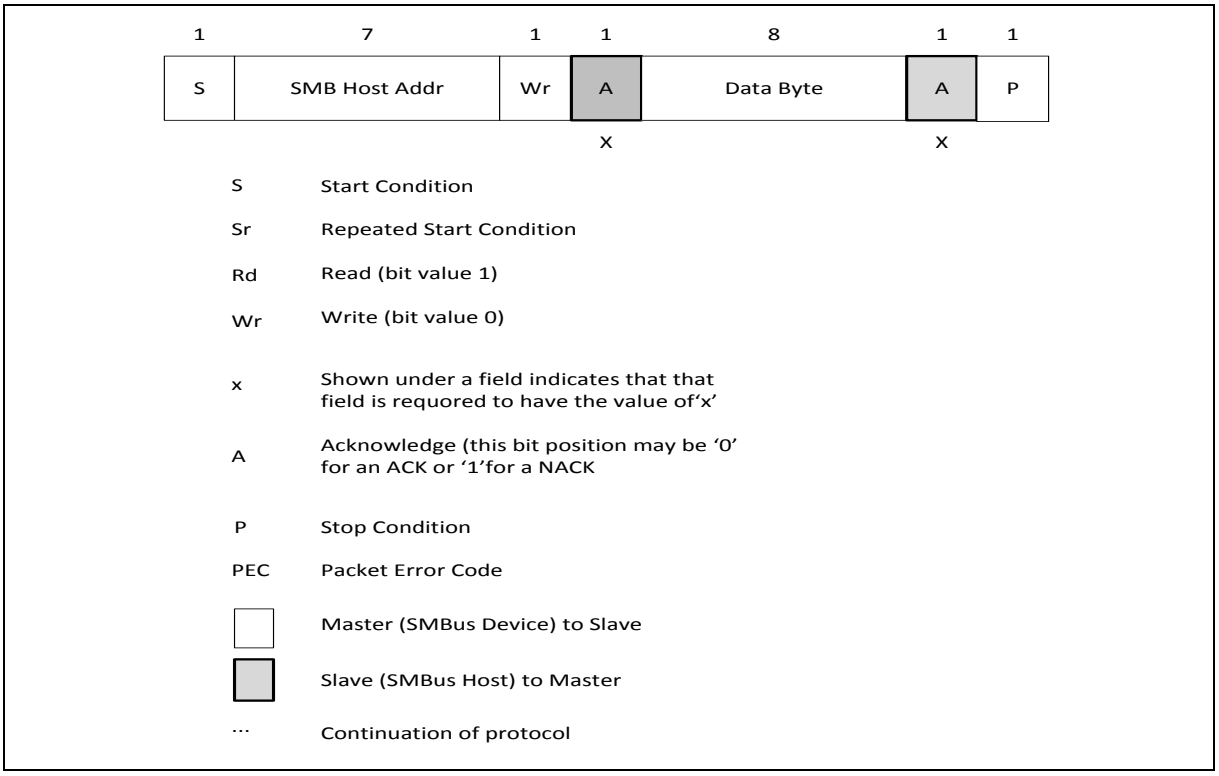


图 6.23-17 总线管理主要元素包协议图

地址解析协议(ARP)

从机地址冲突，总线管理可以通过动态给从设备赋予唯一的新地址解决。为了提供地址识别机制来区分各个设备，每个设备必须分配一个唯一设备标识符(UDID),由软件分配128位数字。

该设备支持地址解析协议(ARP)。通过设置BUSEN (I^{2}C_BUSCTL[7])，BMDEN (I^{2}C_BUSCTL[2]) 和ALERTEN (I^{2}C_BUSCTL[4])来使能总线管理设备默认地址(0b1100 001)。ARP命令必须由用户来执行，支持ARP的从机模式也可以执行仲裁。

接收命令和数据应答控制

总线管理接受器必须能够NACK每个收到的命令或数据。为了应许从机模式下的ACK控制，必须通过使能ACKMEN位(I^{2}C_BUSCTL[0])来设置成从设备字节控制模式。

主机通知协议

为防止未知设备未知信息的格式进入总线管理主机控制器，仅支持使用写字协议更改的协议。标准的写字协议通过替换命令代码与改变报警设备的地址来修改。

该外设支持主机通知协议，通过设置BUSEN (I^{2}C_BUSCTL[7]), BMHEN (I^{2}C_BUSCTL[3])和ALERTEN (I^{2}C_BUSCTL[4])实现。在该情况下，主机将会确认总线管理主机地址(0b0001000)。该协议工作时设备作为主机，而主机作为从机。

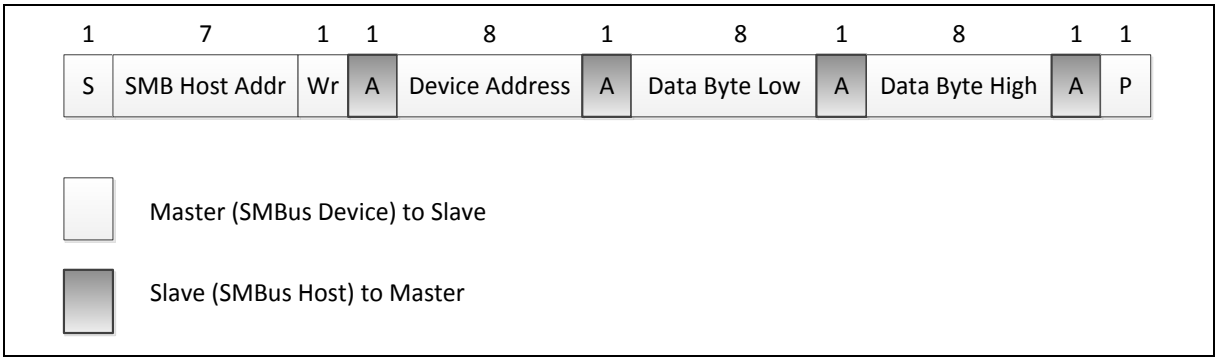


图 6.23-18 7 位可寻址的设备与主机通信

总线管理警报

总线管理支持可选警报信号。如果需要，从设备可以通过总线管理警报管脚(GPA[14]/GPE[10])向主机发信号。通过警报响应地址(0b0001 100)，主机可处理中断并可以同时访问所有总线管理警报管脚上的设备。只有拉低总线管理警报管脚的设备才能应答警报响应地址。

当配置成为从设备(BMHEN=0)时，总线管理警报管脚可以通过设置寄存器(I2C_BUSCTL[4]) ALERTEN 位拉低电平，与此同时警报响应地址(ARA)也使能了。

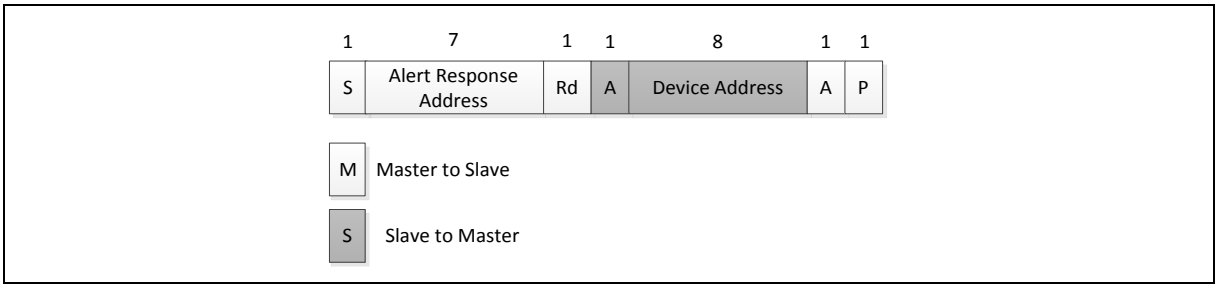


图 6.23-19 7 位可寻址的设备对警报响应地址的响应

配置成为主机(BMHEN=1)时，当总线管理警报管脚的一个下降沿检测到且ALERTEN=1时，警报标志(I^2C_BUSSTS[3])置位。当ALERTEN=0时，即使外部总线管理警报管脚为低电平，警报线还是认为为高。如果总线管理警报管脚不需要，ALERTEN = 0时，总线管理警报管脚可用作标准的GPIO口。

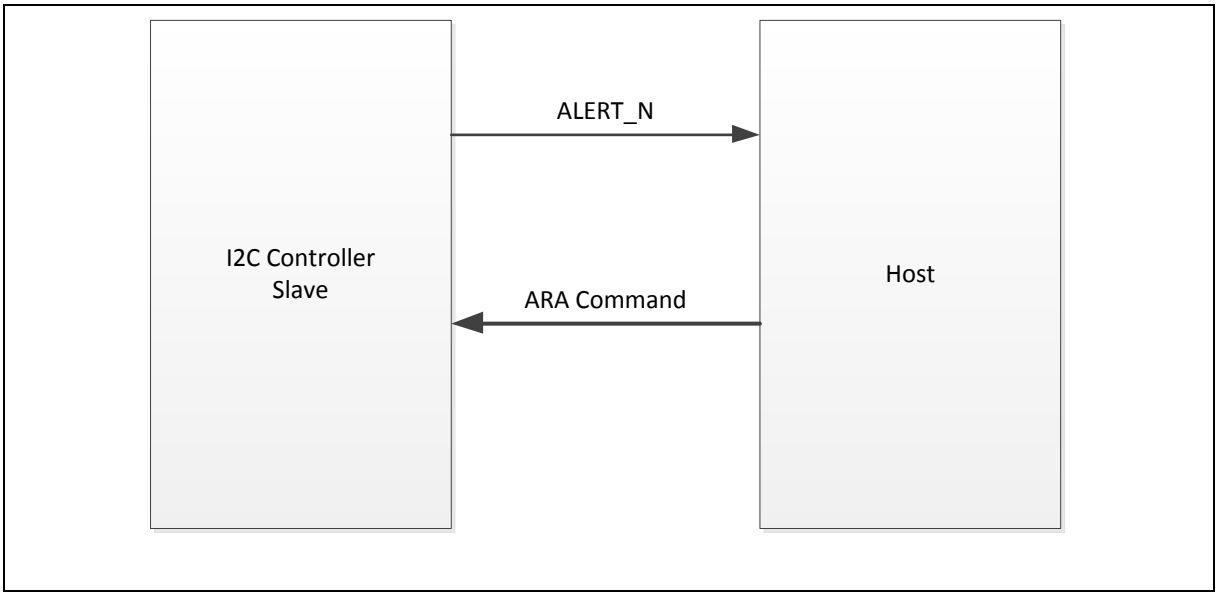


图 6.23-20 总线管理警报功能

包错误检查

包错误检查机制有在SMBus规范里介绍，是用来改进通信的可靠性和健壮性。包错误检查通过每次信息传输的最后增加一个错误包编码(PEC)来实现。PEC是通过使用C(x) = x8 + x2 + x + 1 CRC-8多项式来对所有信息字节(包括地址和读/写位)来计算。

当PECEN位(I^2C_BUSCTL[1])使能时，外设内嵌一个硬件PEC计算器，当收到字节数据与硬件计算的PEC值不匹配时，应许自动发送一个无应答信号。PEC计算的值也可以通过I^2C_PKTCRC读回。

超时

外设内嵌了硬件计数器为了与SMBus规范2.0版本里的3个超时相兼容。

总线管理超时：

总线忙时SCLK低电平超时条件

$T_{Time-out}$ = (BUSTO(I^2C_BUSTOUT[7:0]) +1) x 16x1024 (14-bit) x T_{PCLK} (如果 TOCDIV4 = 0).

= (BUSTO(I^2C_BUSTOUT[7:0])+1) x 16x1024 (14-bit) x 4 x T_{PCLK} (如果 TOCDIV4 = 1)

总线空闲条件(SCLK和SDA为高)

$T_{Time-out}$ = (BUSTO(I^2C_BUSTOUT[7:0])+ 1) x 4 x T_{PCLK}.

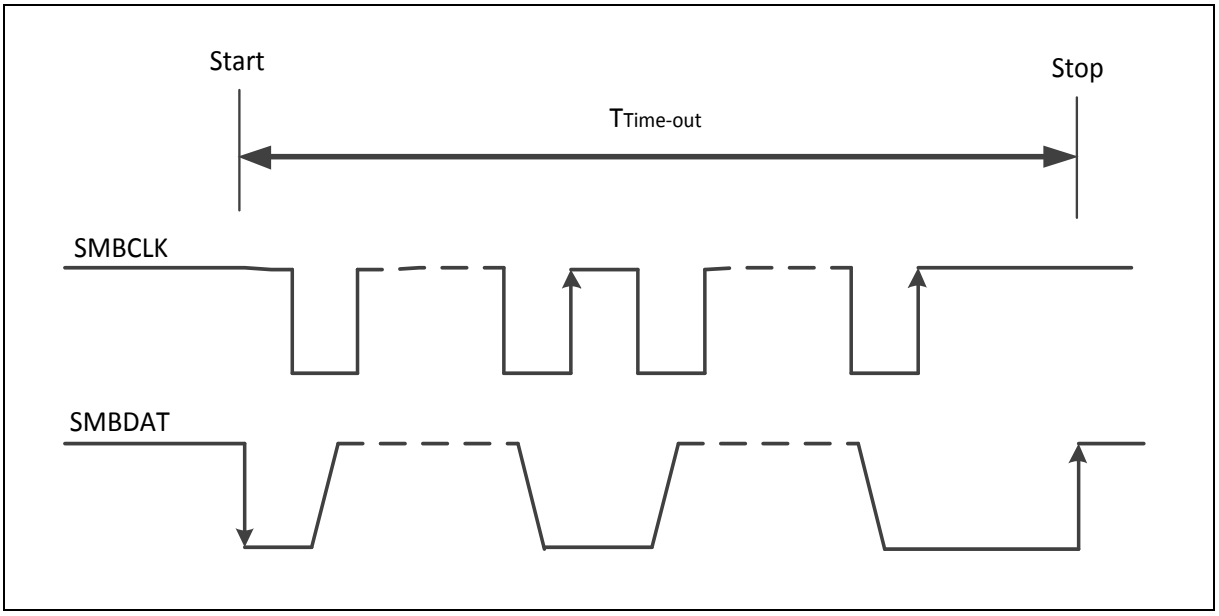


图 6.23-21 系统管理总线超时时序

总线时钟低电平超时：

在主机模式，侦测主机低电平延时时间($T_{LOW:MEXT}$)。

在从机模式，侦测从机低电平延时时间($T_{LOW:SEXT}$)。

$T_{TLOW:EXT}$ = (CLKTO (I^2C_CLKTOUT[7:0])+1) x 16x1024 (14-bit) x T_{PCLK} (如果 TOCDIV4= 0).

= (CLKTO (I^2C_CLKTOUT[7:0])+1) x 16x1024 (14-bit) x 4 x T_{PCLK} (如果 TOCDIV4= 1)

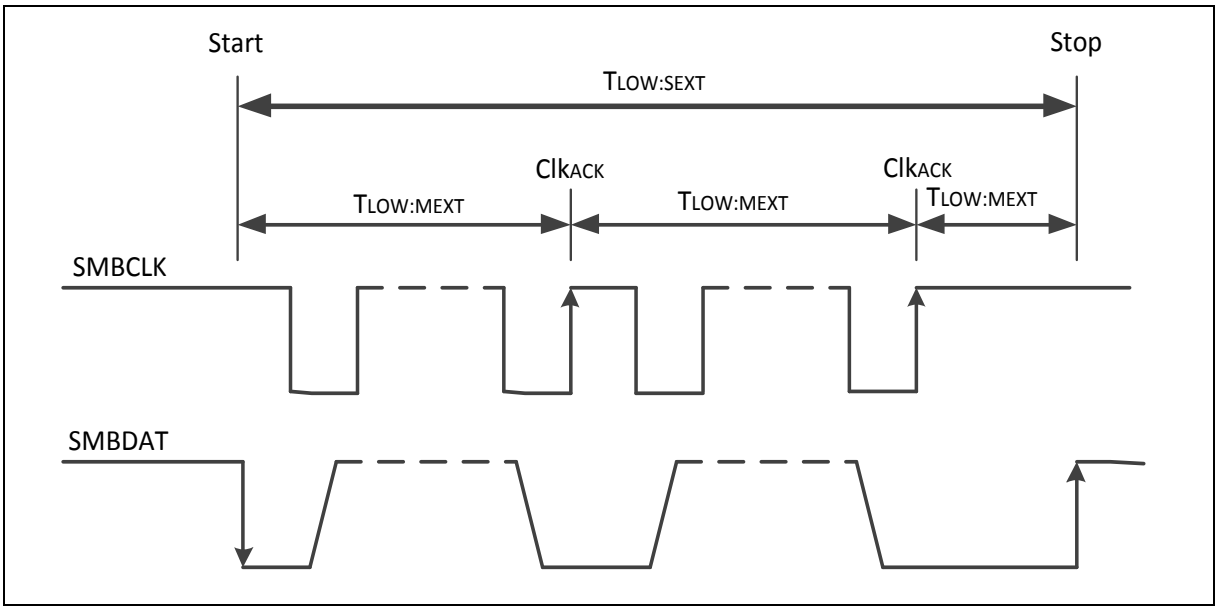


图 6.23-22 总线时钟低电平超时时序

总线空闲侦测

如果侦测到时钟和数据线上信号高电平时间已经T_{IDLE}大于$T_{HIGH,MAX}$时，主机可以认为总线已经空闲。时

序参数包括了一个主机已被动态加入到总线，可能没有检测到SMBCLK和SMBDAT线上状态的变化条件。在这种情况下，主机必须等待足够长的时间，以确保传输不在进行中。外设支持硬件空闲检测。

6.23.5.3 PDMA 传输

I²C控制器支持PDMA传输功能。当TXPDMAEN (I²C_CTL1 [0])被置1，I²C控制器会产生请求到PDMA控制器使其自动地启动DMA发送进程。

当RXPDMAEN (I²C_CTL1 [1])被置为1时，I²C控制器会启动PDMA接收进程。当有数据写入到接收BUFFER时，I²C控制器会自动地产生请求到PDMA控制。

当I²C进入到PDMA模式，大多数状态中断被屏蔽了。不让中断发生除了总线错误或NACK或STOP中断(0x20, 0x30, 0x38, 0x48, 0x58, 0x00, 0xA0, 0xC0, 0x88和0x98)。仅在I²C控制器处在主机TX模式下设置PDMASTR (I²C_CTL1 [8])。如果PDMASTR被清0，在PDMA传输完成和缓存为空之后I²C会自动地发送STOP。如果PDMASTR被置1，在PDMA传输完成和缓存为空之后SI会被置1并且I²C总线会被硬件钳位。

6.23.5.4 编程的建立和保持时间

为了保证一个正确数据的建立和保持时间，则时序必须可以配置。通过编程HTCTL (I²C_TMCTL[24:16])来配置保持时间，通过编程STCTL (I²C_TMCTL[8:0])来配置建立时间

延时时序参考外设时钟(PCLK)。当设备拉住主机时钟时，建立和保持时间的配置值不会被影响。

用户需要注意在建立和保持时间配置的极限上，时序设置必须遵从I²C协议。一旦建立时间配置大于设计极限，那就意味着如果设置的建立时间让SCL输出小于三个PCLK，由于SCL采样为三次那么I²C控制器就不能正常工作。一旦保持时间配置大于I²C时钟的极限，I²C将会发送总线错误。在设置时序前建议用户结合传输速率和协议计算好合理的时序。表6.23-2示了I²C传输速率与PCLK之间的关系，表格呈现的数是一个时钟间隙包含多少PCLK。建立和保持时间的配置哪怕在我们设计中编程一些极限值，但是用户必须要遵从I²C的标准协议。

I²C 传输速率 PCLK	100k	200k	400k	800k	1200k
12 MHz	120	60	30	15	10
24 MHz	240	120	60	30	20
48 MHz	480	240	120	60	40
72 MHz	720	360	180	90	60

表 6.23-2I²C 传输速率与 PCLK 之间的关系

对于建立时间错误调整范例，我们假设一个SCL周期包含5个PCLKs并且设置STCTL (I²C_TMCTL[8:0])为3那样对于建立时间设置拉伸三个PCLKs。建立时间最大设置值：ST_{limit} = (I²C_CLKDIV[7:0]+1) X 2 - 6.

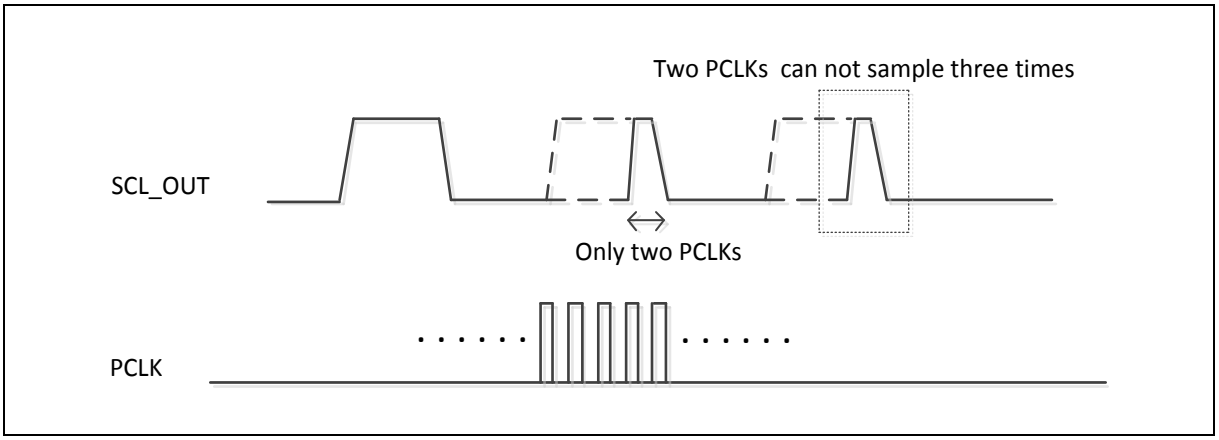


图 6.23-23 建立时间错误调整

对于保持时间错误调整范例，我们使用I²C传输速率=1200K和PCLK=72MHz，SCL的高电平/低电平空隙等于60PCLK。当我设置HTCTL (I²C_TMCTL[24:16])为61和STCTL (I²C_TMCTL[8:0])为0时，然后SDA输出延时将会越过SCL为高的空隙，以至产生总线错误。保持时间最大值的是设定：HT_{limit} = (I²C_CLKDIV[7:0]+1) X 2 – 9 。

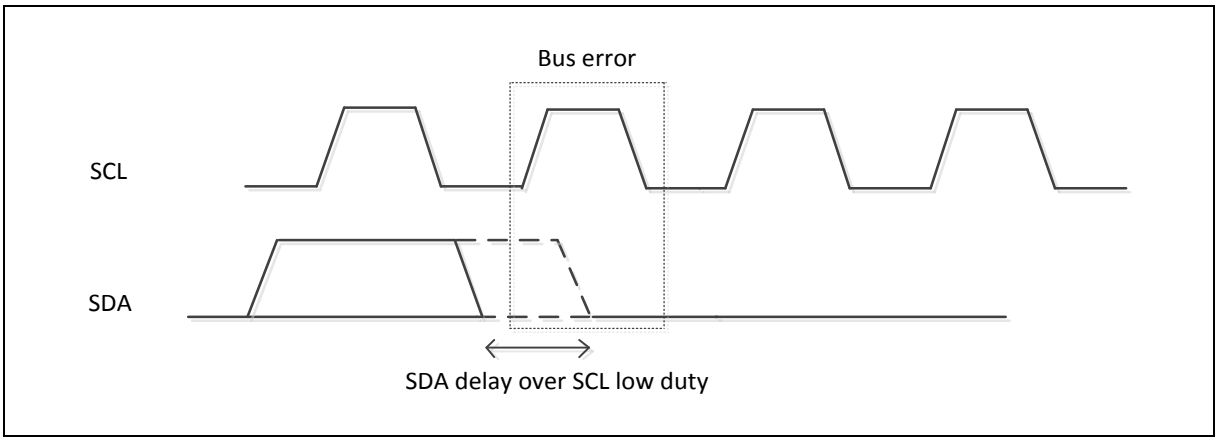


图 6.23-24 保持时间错误调整

6.23.5.5 I²C 协议寄存器

通过下列15个特殊功能寄存器来控制I²C端口：I²C_CTL（控制寄存器）I²C_STATUS（状态寄存器），I²C_DAT（数据寄存器），I²C_ADDRn（地址寄存器，n=0~3），I²C_ ADDRMSKn（地址掩码寄存器，n=0~3），I²C_CLKDIV（时钟速率寄存器），I²C_TOCTL（超时控制寄存器），I²C_WKCTL (唤醒控制寄存器)和I²C_WKSTS (唤醒状态寄存器)。

地址寄存器 (I²C_ADDR)

I²C端口内建4个从机地址寄存器I²C_ADDRn (n=0~3)。当I²C作为主机时，这四个寄存器的内容不相关。在从机模式下，位字段ADDR(I²C_ADDRn [7:1]) 必须装载芯片自己的从机地址。当I²C_ADDRn地址与接收到的从机地址符合时I²C硬件会作出反应。

I²C端口支持“广播呼叫”功能。当GC位(I²C_ADDRn [0]) 被设置，I²C端口硬件将应答广播呼叫地址(00H)。清GC位可禁用“广播呼叫”功能。

当GC位被置且I²C处于从机模式时，主机发出广播呼叫地址到I²C总线后，从机可以通过地址00H接收广播呼叫地址，然后它将跟随GC 模式的状态。

从机地址掩码寄存器 (I²C_ADDRMSK)

总线控制器带有四个地址掩码寄存器I2C_ADDRMSKn (n=0~3) 支持多地址识别。当地址掩码寄存器被置1，意味着收到的相应地址位是"无关的".如果置0则收到相应地址位应跟地址寄存器完全一样。

数据寄存器 (I²C_DAT)

该寄存器包含一个准备发送或刚接收到的一个字节的串行数据。当不在字节移位处理过程时，CPU 可以直接读写访问I²C_DAT[7:0]。当I²C处于一个确定的状态并且串行中断标志(SI)被置1，I²C_DAT[7:0]中的数据保持稳定。当数据被移出，总线上的数据同时被移入。I²C_DAT [7:0]的内容总是总线上传递的最后一个字节。

应答位由I²C硬件控制，不能被CPU 访问。串行数据在SCL线上串行时钟脉冲的上升沿移入I²C_DAT [7:0]。当一个字节被移入I²C_DAT [7:0]，则I²C_DAT [7:0]中的串行数据是可用的，应答位(ACK 或NACK)在第9个时钟脉冲由控制逻辑返回。为了在发送数据时监控总线的状态，当发送I²C_DAT [7:0]到总线时，总线数据将同时被移入I²C_DAT[7:0]。在发送数据过程中，串行数据在SCL 时钟脉冲的下降沿从I²C_DAT[7:0] 移出，在SCL时钟脉冲的上升沿数据移入I²C_DAT [7:0]。图6.23-25展示了I²C数据移位方向。

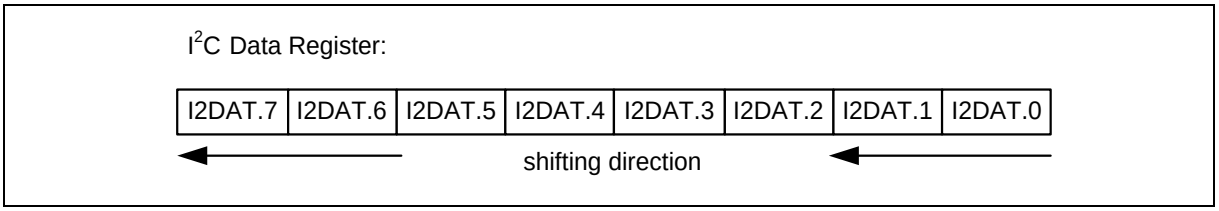


图 6.23-25 I²C 数据以为方向

控制寄存器 (I²C_CTL0)

CPU可以直接读或写寄存器I²C_CTL0[7:0]，当I²C端口通过设置I²CEN (I²C_CTL0 [6])为高使能，内部状态由I²C_CTL0和I²C逻辑硬件控制

有两个位会受硬件影响：当I²C硬件产生中断时SI被置位，当总线上产生停止信号时，STO位被清零，当I²CEN =0时STO也会被清零。

一旦有新的状态码产生并存储在I²C_STATUS0，I²C中断标志位SI(I²C_CTL0 [3])将被自动置位。若此时使能中断INTEN (I²C_CTL0 [7])位被设置，I²C中断将产生。I²C_STATUS0[7:0]用来存储内部状态码，SI被软件清除前内容一直保持。

状态寄存器 (I²C_STATUS0)

I²C_STATUS0 [7:0] 是一个8-位只读寄存器。I²C_STATUS0 [7:0]共有26 个可能的状态码。所有状态码由下面表6.23-3所列出。当I²C_STATUS0 [7:0] 的值为0xF8时，没有串行中断请求。所有其他的I²C_STATUS0 [7:0]的值对应I²C的状态。当进入其中任一状态时，就会产生状态中断请求(SI=1)。在SI被硬件置位或SI被软件复位后一个机器周期，有效状态码出现在I²C_STATUS [7:0]中。

此外，状态0x00表示总线错误，这会出现在起始或停止条件处在不正确的I²C格式帧。总线错误会发生在一个地址字节、一个数据字节或一个应答位的串行传输。从错误总线恢复时STO应被置位，SI应被清除，来进入无地址从机模式。然后STO清除释放总线并等待下一个传输。出现总线错误操作期间I²C总线不能识别停止条件。

主机模式	从机模式

状态	描述	状态	描述
0x08[1]	起始	0xA0	从机发送重新起始或停止
0x10[1]	主机重新起始	0xA8[1]	从机发送地址ACK
0x18[1]	主机发送地址ACK	0xB8[1]	从机发送数据ACK
0x20	主机发送地址NACK	0xC0	从机发送数据NACK
0x28[1]	主机发送数据ACK	0xC8[1]	从机发送最后数据ACK
0x30	主机发送数据NACK	0x60[1]	从机接收地址 ACK
0x38	主机仲裁丢失	0x68[1]	从机接收仲裁丢失
0x40[1]	主机接收地址ACK	0x80[1]	从机接收数据 ACK
0x48	主机接收地址NACK	0x88	从机接收数据NACK
0x50[1]	主机接收数据ACK	0x70[1]	广播模式地址ACK
0x58	主机接收数据NACK	0x78[1]	广播模式仲裁丢失
0x00	总线错误	0x90[1]	广播模式数据 ACK
		0x98	广播模式数据 NACK
		0xB0[1]	从机发送仲裁丢失
0xF0	如果BMDEN =1且ACKMEN位使能，I2C_STATUS0的值在从机接收条件时将固定为0xF0		
0xF8	总线释放 注: 主/从模式的“0xF8”状态，都不会产生中断 注 [1]: PDMA模式下没有中断		

表 6.23-3 I2C 状态码描述

时钟波特率位 (I2C_CLKDIV)

当I2C在主机模式下，I2C数据的波特率由CLK(I2C_CLKDIV [7:0])寄存器设定。在从机模式下时不需要。在从机模式下，I2C将自动与主机I2C设备时钟频率同步。在从机模式下，系统时钟频率应该大于I2C总线最大时钟的20倍。

I2C 数据波特率的设定：I2C数据波特率= (系统时钟) / (4x ((I2C_CLKDIV [7:0] +1))。如果系统时钟 = 16 MHz, I2C_CLKDIV [7:0] = 40 (0x28)，那么I2C数据波特率= 16 MHz/ (4x (40 +1)) = 97.5 Kbits/sec。

超时控制寄存器 (I2C_TOCTL)

系统提供一个14位超时的计数器来处理当I2C总线锁死时的情况。当超时计数器计数功能使能后，计数器开始计数直至溢出(TOIF=1)并向CPU产生I2C中断或者清除TOCEN为0停止计数。当超时计数器使能，对SI标志置高会使计数器复位，再对SI清零会重新开始计数。如果I2C总线锁死，会使I2C_STATUS及SI 标志不再更新，该14-位超时计数器会发生溢出从而产生I2C中断通知CPU。参考下图6.23-26超时计数器。用户可以写1清TOIF为0。

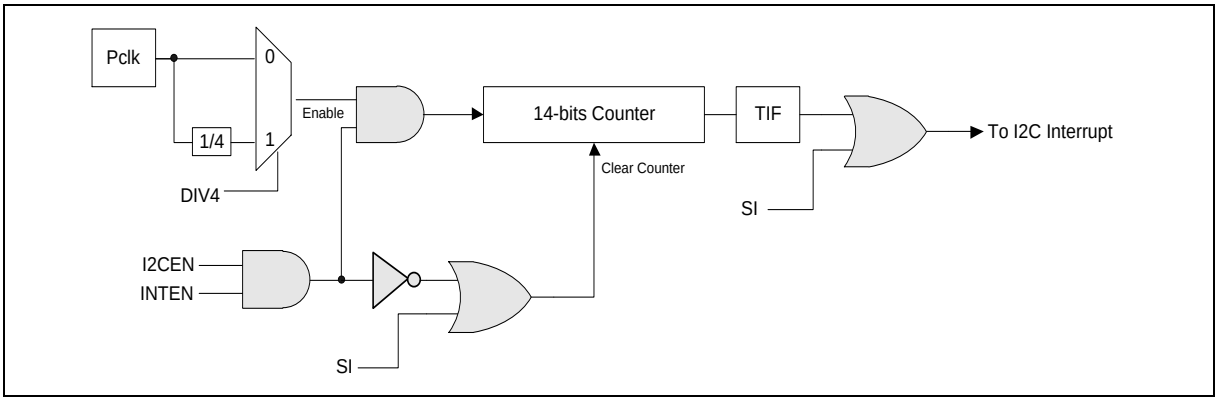


图 6.23-26 I²C 超时计数模块框图

唤醒控制寄存器 (I²C_WKCTL)

当芯片进入掉电模式并且WKEN (I²C_WKCTL [0])被设置为1时，其它I²C主机通过寻址我们I²C设备唤醒我们芯片，在进入睡眠前必须配置相关设置。当地址与设备地址匹配时并且ACK阶段过后，控制器会拉低SCL，然后控制器启动。如果NHDBUSEN (I²C_WKCTL [7])被设置，控制器将不会拉低SCL。请注意当控制器不能拉低SCL时，发送或接收数据会立即执行。如果发送或是接收数据后SI没有被清除，用户必须复位控制器并且重新执行原先的操作。

唤醒状态寄存器 (I²C_WKSTS)

当系统被其他的I²C主机设备唤醒时，WKIF被置位表示该事件发生。用户需要写“1”来清除此位。

当芯片被设备地址寄存器(I²C_ADDRn)其中之一地址匹配唤醒时，用户需要检查WKAKDONE (I²C_WKSTS [1])是否被置1来确认地址字节已完成。WKAKDONE位表明在掉电时候ACK阶段已完成。当地址与设备地址匹配时并且ACK阶段过后，控制器会拉低SCL，直到WKAKDONE被用户清除。如果SCL是低频速率并且系统被地址匹配帧唤醒，用户需要检查WKAKDONE来确认该帧已完成传输然后去做唤醒后的动作。请注意用户不能通过清除WKAKDONE为0来释放WKIF。

在控制器发送地址前，WRSTSWK (I²C_WKSTS [2])位用来记录读/写命令。在系统被地址匹配帧唤醒后，用户读该位的状态来准备下次传输数据(WRSTSWK = 0)或是等待到来的数据 (WRSTSWK = 1)及时地被存储。请注意当写1到WKAKDONE (I²C_WKSTS [1])时，WRSTSWK (I²C_WKSTS [2])被清除。

当系统被其他I²C主机唤醒后，WKIF被设定来指示此事件。用户需要写1来清除此位。

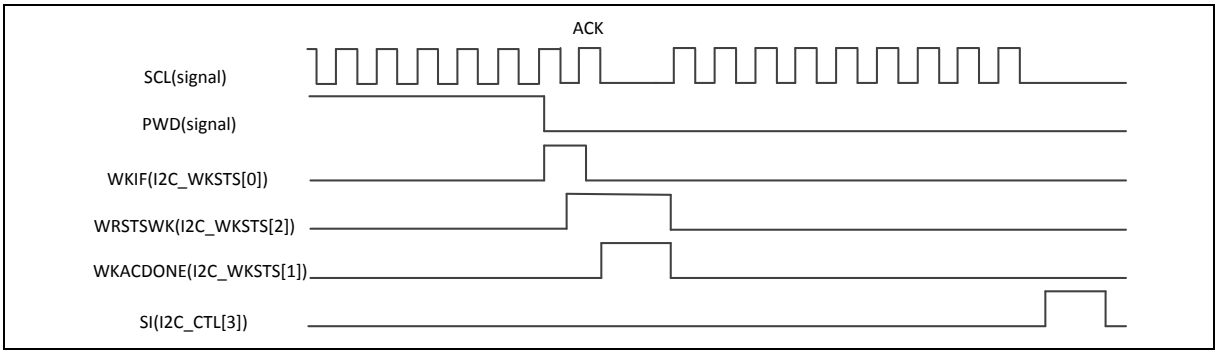


图 6.23-27 I²C 唤醒的相关信号波形

I²C 控制寄存器 1 (I²C_CTL1)

如果设置ADDR10EN (I2C_CTL1 [9])使能10位地址模式，I²C将会执行10位模式。

对于PDMA功能，设置TXPDMAEN (I²C_CTL1 [0])和RXPDMAEN (I²C_CTL1 [1])来启动。通过设置PDMARST (I²C_CTL1 [2])复位PDMA控制器。

I²C 状态寄存器 1 (I²C_STATUS1)

I²C控制器支持四组从机地址标志寄存器，ADMAT0, ADMAT1, ADMAT2 和 ADMAT3 (I²C_STATUS1[3:0])。每个寄存器代表该地址被使用并且被置1通知软件。

I²C 时序配置控制寄存器 (I²C_TMCTL)

为了配置建立/保证时间，要根据实际需求设置HTCTL (I2C_TMCTL[24:16]) 和 STCTL (I2C_TMCTL[8:0])。

总线管理控制寄存器 (I²C_BUSCTL)

SM总线管理控制事件在该寄存器定义。它包括手动控制应答 (ACKMEN (I²C_BUSCTL[0]))，包错误检测使能(PECEN (I²C_BUSCTL[1]))，设备默认地址使能 (BMDEN(I²C_BUSCTL[2])) 或者主机功能使能 (BMHEN (I²C_BUSCTL[3]))。警报和挂起功能可以由ALERTEN (I²C_BUSCTL[4])，SCTLOSTS (I²C_BUSCTL[5]))和SCTLOEN (I²C_BUSCTL[6])来设置。

系统总线管理允许通过BUSEN(I²CBUSCTL[7])位进行控制。BUSTOUT(I²CBUSCTL[9])用于计算有效总线上时钟低电平时间超时和总线空闲中的空闲周期。

可以通过控制PECTXEN(I²C_BUSCTL[8])位来传送或者接收计算PEC(如果PECEN置位)的值

该寄存器有个特别的位ACKM9SI (I²C_BUSCTL[11])。当ACKMEN使能时，在第8个时钟输入时，有SI中断产生，用户此时可以读取数据和状态寄存器。当SI中断清除时第8个时钟总线被释放，此时，如果ACKM9SI位设置为1，在第9个时钟周期时会有另一个SI中断事件发生，这样就知道传输帧完成后的总线状态。

对于PEC控制流程则是设置PECDIEN (I²C_BUSCTL[13]), BCDIEN (I²C_BUSCTL[12]) 或 PECCLR (I²C_BUSCTL[10])。

I²C 总线管理定时器控制寄存器 (I²C_BUSTCTL)

设置TORSTEN(I²C_BUSTCTL[4])，CLKTOIEN(I²C_BUSTCTL[3])，BUSTOIEN (I²C_BUSTCTL[2]), CLKTOEN (I²C_BUSTCTL[1])和BUSTOEN (I²C_BUSTCTL[0])来配置总线超时或时钟低定时器超时。

I²C 总线管理状态寄存器 (I²C_BUSSTS)

对于PEC控制流程，需要监控PECDONE (I²C_BUSSTS[7]), BCDONE (I²C_BUSSTS[1])或PECERR (I²C_BUSSTS[2])。

监控SCTLDIN (I²C_BUSSTS[4])来观察SUSCON输入状态。

I²C 字节数寄存器 (I²C_PKTSIZE)

当PECEN位 (I²C_BUSCTL[1]) 使能时，I²C控制器将计算总线上数据的PEC值。PLDSIZE (I²C_PKTSIZE[8:0])用来定义总线上数据需要传输的个数。当计数值达到PLDSIZE时，且PECTXEN (I²C_BUSCTL[8])有设置，最后的PEC值将会发送或者自动接收。

I²C PEC 值寄存器 (I²C_PKTCRC)

该寄存器为I²C总线上传输数据计算出来的PECCRC (I²C_PKTCRC[7:0])值。详细信息由在SM总线的PEC部分说明。

I²C 总线管理定时器和 I²C 时钟低定时器寄存器 (I²C_BUSTOUT/ I²C_CLKTOUT)

这两个寄存器的说明在总线管理 (SMBus/PMBus兼容)部分详细描述。

6.23.5.6 EEPROM随机读取范例

通过下面的步骤来配置I²C0相关寄存器，来使用I²C从EEPROM读取数据。

1. 设置I²C0的多功能引脚为SCL和SDA引脚。多功能配置请参考基本配置章节。
2. 使能I²C0 APB时钟。时钟配置请参考基本配置章节。
3. 通过设置I²C0RST=1来复位I²C0控制器，然后设置I²C0控制器为普通操作。复位控制器配置请参考基本配置章节。
4. 在“I²C_CTL0”寄存器中设置I²CEN=1来使能I²C0控制器。
5. 在“I²C_CLKDIV”寄存器中设置I²C0时钟分频值来设置I²C时钟频率。
6. 在系统“NVIC”控制寄存器中使能系统I²C0 IRQ。
7. 在“I²C_CTL0”寄存器中设置INTEN=1来使能I²C0中断。
8. 设置I²C0地址寄存器“I²C_ADDR0 ~ I²C_ADDR3”。

随机读操作是访问EEPROM的其中一种方法。这个方法是允许主机访问EEPROM的任何一个地址。图6.23-28展示了EEPROM随机读取操作。

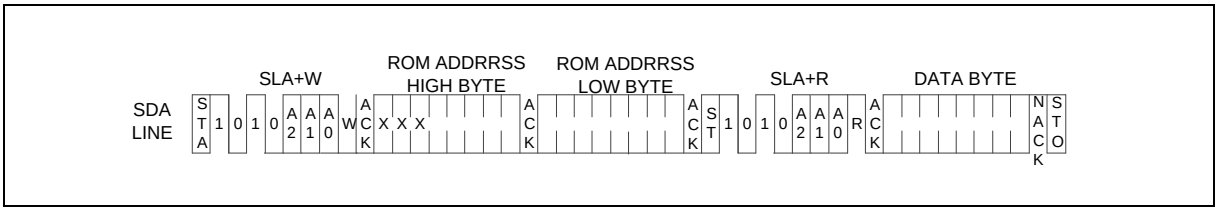


图 6.23-28 EEPROM 随机读取

图6.23-29展示了怎样使用I²C控制器执行EEPROM随机读取的协议。

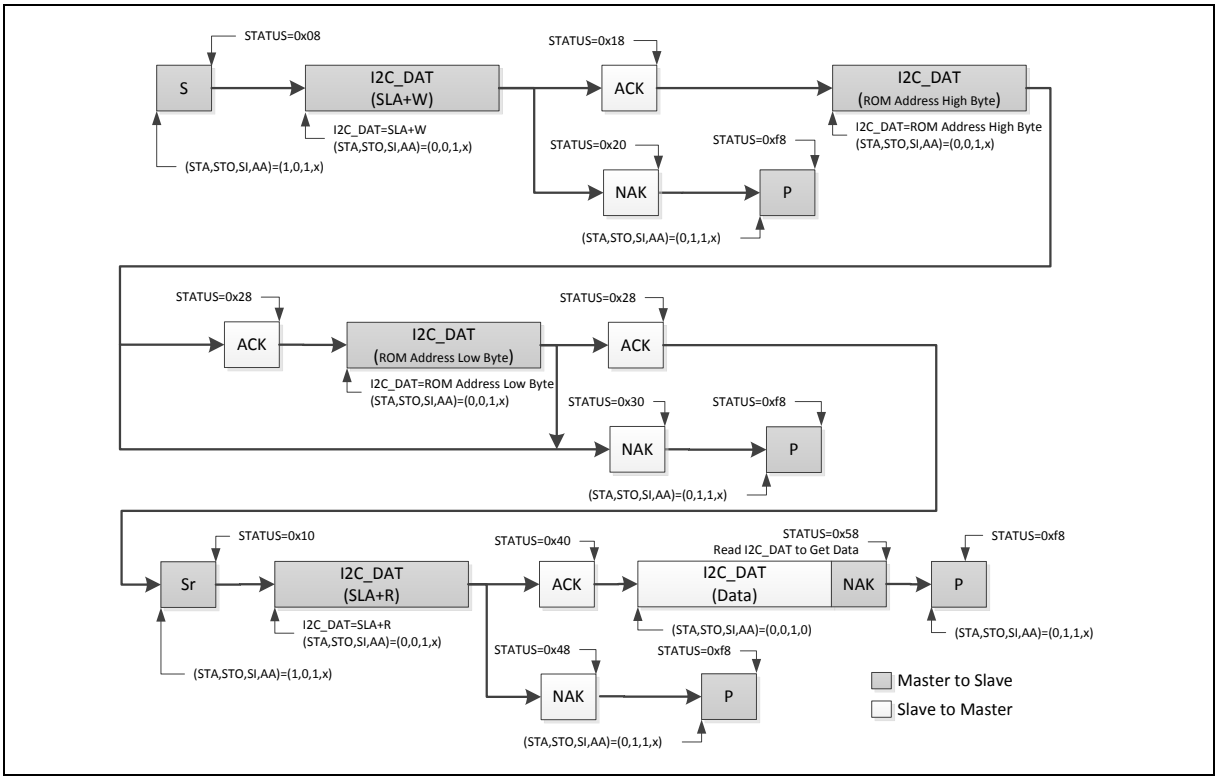


图 6.23-29 EEPROM 随机读取协议

I²C控制器作为主机发送起始信号到总线，然后发送SLA+W (从机地址 + 写位)到EERPOM，跟着由两个字节数据地址来设置EEPROM被读的地址。最后，重复起始信号跟着SLA+R被发送来向EEPROM读取数据。

6.23.6 寄存器映射

R:只读, **W:** 只写, **R/W:**读/写

寄存器	偏移地址	R/W	描述	复位值
I²C 基地址： **I²Cn_BA = 0x4008_0000 + (0x1000 *n)** **n= 0,1,2**				
I²C_CTL0	I²Cn_BA+0x00	R/W	I²C控制寄存器 0	0x0000_0000
I²C_ADDR0	I²Cn_BA+0x04	R/W	I²C从机地址寄存器0	0x0000_0000
I²C_DAT	I²Cn_BA+0x08	R/W	I²C数据寄存器	0x0000_0000
I²C_STATUS0	I²Cn_BA+0x0C	R	I²C状态寄存器 0	0x0000_00F8
I²C_CLKDIV	I²Cn_BA+0x10	R/W	I²C时钟分频寄存器	0x0000_0000
I²C_TOCTL	I²Cn_BA+0x14	R/W	I²C超时控制寄存器	0x0000_0000
I²C_ADDR1	I²Cn_BA+0x18	R/W	I²C从机地址寄存器1	0x0000_0000
I²C_ADDR2	I²Cn_BA+0x1C	R/W	I²C从机地址寄存器2	0x0000_0000
I²C_ADDR3	I²Cn_BA+0x20	R/W	I²C从机地址寄存器3	0x0000_0000
I²C_ADDRMSK0	I²Cn_BA+0x24	R/W	I²C从机地址掩码寄存器0	0x0000_0000
I²C_ADDRMSK1	I²Cn_BA+0x28	R/W	I²C从机地址掩码寄存器1	0x0000_0000
I²C_ADDRMSK2	I²Cn_BA+0x2C	R/W	I²C从机地址掩码寄存器2	0x0000_0000
I²C_ADDRMSK3	I²Cn_BA+0x30	R/W	I²C从机地址掩码寄存器3	0x0000_0000
I²C_WKCTL	I²Cn_BA+0x3C	R/W	I²C唤醒控制寄存器	0x0000_0000
I²C_WKSTS	I²Cn_BA+0x40	R/W	I²C唤醒状态寄存器	0x0000_0000
I²C_CTL1	I²Cn_BA+0x44	R/W	I²C控制寄存器 1	0x0000_0000
I²C_STATUS1	I²Cn_BA+0x48	R/W	I²C状态寄存器 1	0x0000_0000
I²C_TMCTL	I²Cn_BA+0x4C	R/W	I²C 时序配置控制寄存器	0x0000_0000
I²C_BUSCTL	I²Cn_BA+0x50	R/W	I²C总线管理控制寄存器	0x0000_0000
I²C_BUSTCTL	I²Cn_BA+0x54	R/W	I²C总线管理定时器控制寄存器	0x0000_0000
I²C_BUSSTS	I²Cn_BA+0x58	R/W	I²C总线管理状态寄存器	0x0000_0000
I²C_PKTSIZE	I²Cn_BA+0x5C	R/W	I²C包错误检查字节数寄存器	0x0000_0000
I²C_PKTCRC	I²Cn_BA+0x60	R	I²C包错误检查字节值寄存器	0x0000_0000
I²C_BUSTOUT	I²Cn_BA+0x64	R/W	I²C总线管理定时器寄存器	0x0000_0005
I²C_CLKTOUT	I²Cn_BA+0x68	R/W	I²C总线管理时钟低定时器寄存器	0x0000_0005

6.23.7 寄存器描述

I²C 控制寄存器 (I²C_CTL0)

寄存器	偏移地址	R/W	描述	复位值
I²C_CTL0	I²Cn_BA+0x00	R/W	I²C控制寄存器 0	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
INTEN	**I²CEN**	**STA**	**STO**	**SI**	**AA**	**Reserved**	

位	描述	
[31:8]	**Reserved**	保留
[7]	**INTEN**	**使能中断** 0 = 禁用 I²C 中断 1 = 使能 I²C 中断
[6]	**I²CEN**	**I²C控制器使能位** 设置使能 I²C 串行功能控制器。当I²CEN =1，I²C 串行功能使能。SDA和SCL对应的多功能管脚功能必须先设置为 I²C 功能。 0 = I²C控制器禁用 1 = I²C控制器使能
[5]	**STA**	**I²C起始控制位** 设置STA为1，进入主机模式，如果总线处于空闲状态，I²C 硬件会送出起始或重复起始条件。
[4]	**STO**	**I²C停止控制位** 在主机模式，设置STO来传送一个STOP条件到总线，然后I²C硬件将会检查总线状况，如果检测到一个STOP状况，这个标志会被硬件自动清除。
[3]	**SI**	**I²C中断标志** 当一个新的 I²C状态出现在寄存器I²C_STATUS0时，SI 标志由硬件置位，并且如果INTEN (I²C_CTL [7])位被置位，则产生I²C中断请求。SI必须由软件通过向该位写'1'清零。 如果在从机读模式且ACKMEN置位，SI标志在第8个时钟周期置位，此时用户确认应答位，在第9个时钟周期时，用户可以从数据缓冲器中读取数据。
[2]	**AA**	**应答控制位** 当AA=1先于地址或数据接收，在SCL线上的应答时钟脉冲期间将返回一个应答(SDA上为低

		电平)，有两种情况：1.) 从机正在应答主机发送的地址。2.) 接收设备正在应答发送设备发送的数据。当AA = 0先于地址或数据接收，则在SCL线上的应答时钟脉冲期间将返回一个非应答（SDA上为高电平）。
[1:0]	**Reserved**	保留

I²C 数据寄存器 (I²C_DAT)

寄存器	偏移地址	R/W	描述	复位值
I²C_DAT	I²Cn_BA+0x08	R/W	I²C数据寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
DAT							

位	描述	
[31:8]	**Reserved**	保留
[7:0]	**DAT**	**I²C数据** 该区域8位[7:0]存放I²C串行端口8位传输/接收的数据

I²C 状态寄存器 (I²C_STATUS0)

寄存器	偏移地址	R/W	描述	复位值
I²C_STATUS0	I²Cn_BA+0x0C	R	I²C状态寄存器0	0x0000_00F8

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
STATUS							

位	描述	
[31:8]	**Reserved**	保留
[7:0]	**STATUS**	**I²C状态** 低3位有效位一直为0,。高5位有效位包括状态码。有28种可能的状态码。当I2C_STATUS0的内容为0xF8时，没有中断产生。所有其他的I²C_STATUS0的值对应I²C的状态。当进入其中任一状态时，就会产生状态中断请求(SI=1)。在SI被硬件置位或SI被软件复位后一个机器周期，有效状态码出现在 I2C_STATUS0中。此外，00H状态表示总线错误。总线错误发生在START或STOP条件出现在帧结构不正确的位置。不正确的位置比如是在串行传输地址字节、数据字节或应答位期间。

I²C 时钟分频寄存器 (I²C_CLKDIV)

寄存器	偏移地址	R/W	描述	复位值
I²C_CLKDIV	I²Cn_BA+0x10	R/W	I²C时钟分频寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved						**DIVIDER**	
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
DIVIDER							

位	描述	
[31:10]	**Reserved**	保留
[9:0]	**DIVIDER**	**I²C时钟分频** I²C数据波特率= (系统时钟) / (4x (I²C_CLKDIV+1))。 **注:** I²C_CLKDIV的最小值是4。

I²C 超时控制寄存器 (I²C_TOCTL)

寄存器	偏移地址	R/W	描述	复位值
I²C_TOCTL	I²Cn_BA+0x14	R/W	I²C超时控制寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved					TOCEN	TOCDIV4	TOIF

位	描述	
[31:3]	**Reserved**	保留
[2]	**TOCEN**	**超时计数器使能位** 0 = 超时计数器禁用 1 = 超时计数器使能 当使能该位，则14位溢出定时器将会在SI清零后开始计数。设置SI位为高将会复位计数器，在 SI位清零之后，计数器会重新开始计数。
[1]	**TOCDIV4**	**溢出定时器输入时钟除以 4** 0 = 溢出定时器输入时钟禁用 1 = 溢出定时器输入时钟使能 该位使能后，超时时间扩大4倍。
[0]	**TOIF**	**超时溢出标志** 当超时发生时，该位由硬件置位，如果此时I²C的中断使能位INTEN置为1，则可引发 CPU 的中断。 **注意**：写1清除该位。

I²C 从机地址寄存器 (ADDRx)

寄存器	偏移地址	R/W	描述	复位值
I²C_ADDR0	I²Cn_BA+0x04	R/W	I²C从机地址寄存器0	0x0000_0000
I²C_ADDR1	I²Cn_BA+0x18	R/W	I²C从机地址寄存器1	0x0000_0000
I²C_ADDR2	I²Cn_BA+0x1C	R/W	I²C从机地址寄存器2	0x0000_0000
I²C_ADDR3	I²Cn_BA+0x20	R/W	I²C从机地址寄存器3	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved					ADDR		
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
ADDR							GC

位	描述	
[31:11]	**Reserved**	保留
[10:1]	**ADDR**	**I²C地址** 当I²C为主机模式时，该寄存器内容没有意义。从机模式下，七个重要的位必须载入到芯片的地址。如果地址符合，I²C硬件将会自动应答。 **注:**当软件设置为10'h000，地址无法使用
[0]	**GC**	**广播呼叫功能** 0 = 广播呼叫功能禁用 1 = 广播呼叫功能使能

I²C 从机地址掩码寄存器 (ADDRMSKx)

寄存器	偏移地址	R/W	描述	复位值
I²C_ADDRMSK0	I²Cn_BA+0x24	R/W	I²C从机地址掩码寄存器0	0x0000_0000
I²C_ADDRMSK1	I²Cn_BA+0x28	R/W	I²C从机地址掩码寄存器1	0x0000_0000
I²C_ADDRMSK2	I²Cn_BA+0x2C	R/W	I²C从机地址掩码寄存器2	0x0000_0000
I²C_ADDRMSK3	I²Cn_BA+0x30	R/W	I²C从机地址掩码寄存器3	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved					ADDRMSK		
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
ADDRMSK							Reserved

位	描述	
[31:11]	**Reserved**	保留
[10:1]	**ADDRMSK**	**I²C地址掩码** 0 = 掩码禁用（接收到的相应地址必须完全符合地址寄存器） 1 = 掩码使能（接收到的相应地址位不予辨识） I²C总线控制器有4个地址掩码寄存器，支持多地址识别。当地址掩码寄存器的某位被置'1'，表示接收到的地址的相应位可忽略。如果该位被置'0'，则表示接收到的地址的相应位必须和地址寄存器中的完全一致。 **注:**唤醒功能不能使用地址掩码
[0]	**Reserved**	保留

I²C 唤醒控制寄存器 (I²C_WKCTL)

寄存器	偏移地址	R/W	描述	复位值
I²C_WKCTL	I²Cn_BA+0x3C	R/W	I²C唤醒控制寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
NHDBUSEN	Reserved						WKEN

位	描述	
[31:8]	Reserved	保留
[7]	NHDBUSEN	**I²C总线保持禁用位** 0 = 唤醒后I²C总线保持 1= 唤醒后禁用I²C总线保持 **注:**当WKIF事件没有被清除时I²C控制器会产生响应，这样会导致数据发送或接收出错。如果数据发送或接收时WKIF事件没有清除，用户必须复位I²C控制器并且重新执行原先的操作。
[6:1]	Reserved	保留
[0]	WKEN	**I²C唤醒功能使能位** 0 = I²C唤醒功能禁止 1= I²C唤醒功能使能

I²C 唤醒状态寄存器 (I²C_WKSTS)

寄存器	偏移地址	R/W	描述	复位值
I²C_WKSTS	I²Cn_BA+0x40	R/W	I²C唤醒状态寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved					**WRSTSWK**	**WKAKDONE**	**WKIF**

位	描述	
[31:3]	**Reserved**	保留
[2]	**WRSTSWK**	**地址唤醒帧中读/写状态位** 0 = 在地址匹配唤醒帧中写命令被记录 1 = 在地址匹配唤醒帧中读命令被记录 **注:**当软件写1到WKAKDONE时该位被清除。
[1]	**WKAKDONE**	**唤醒地址帧应答位完成** 0 = 地址匹配帧ACK阶段没有完成 1 = 地址匹配帧ACK阶段在掉电时已完成 **注:**该位不能释放 WKIF。该位软件写1清0。
[0]	**WKIF**	**I²C唤醒标志** 当芯片从掉电模式下由I₂C唤醒时，该位设置为1，该位软件写1清0。

I²C 控制寄存器 1 (I²C_CTL1)

寄存器	偏移地址	R/W	描述	复位值
I²C_CTL1	I²Cn_BA+0x44	R/W	I²C控制寄存器1	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved						ADDR10EN	PDMASTR
7	6	5	4	3	2	1	0
Reserved					PDMARST	RXPDMAEN	TXPDMAEN

位	描述	
[31:8]	**Reserved**	保留
[9]	**ADDR10EN**	**10位地址功能使能** 0 = 禁用10位地址匹配功能 1 = 使能10位地址匹配功能
[8]	**PDMASTR**	**PDMA钳位** 0 = 在PDMA传输完成后I²C自动发送STOP (仅主机 TX) 1 = 在PDMA传输完成后如果SI没有被清除，I²C SCL总线会被硬件拉住(仅主机 TX)
[7:3]	**Reserved**	保留
[2]	**PDMARST**	**PDMA复位** 0 = 无影响 1 = I²C产生复位请求到PDMA.
[1]	**RXPDMAEN**	**PDMA接收通道有效** 0 = 禁止PDMA接收功能 1 = 使能PDMA接收功能
[0]	**TXPDMAEN**	**PDMA发送通道有效** 0 = 禁止PDMA发送功能 1 = 使能PDMA发送功能

I²C 状态寄存器 1 (I²C_STATUS1)

寄存器	偏移地址	R/W	描述	复位值
I²C_STATUS1	I²Cn_BA+0x48	R/W	I²C状态寄存器1	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							ONBUSY
7	6	5	4	3	2	1	0
Reserved				ADMAT3	ADMAT2	ADMAT1	ADMAT0

位	描述	
[31:8]	**Reserved**	保留
[8]	**ONBUSY**	**总线正忙(只读)** 表明在总线上在处理一个通信。当起始信号被侦测到时该位被硬件置位。当停止信号被侦测到时该位被硬件清除。 0 = 总线处在空闲状态(SCLK 和 SDA 都为高) 1 = 总线忙
[7:4]	**Reserved**	保留
[3]	**ADMAT3**	**I²C 地址3匹配状态** 当地址3匹配，硬件将指示哪个地址被使用。此位将置1，软件可写1清除该位。
[2]	**ADMAT2**	**I²C 地址2匹配状态** 当地址2匹配，硬件将指示哪个地址被使用。此位将置1，软件可写1清除该位。
[1]	**ADMAT1**	**I²C 地址1匹配状态** 当地址1匹配，硬件将指示哪个地址被使用。此位将置1，软件可写1清除该位。
[0]	**ADMAT0**	**I²C 地址0匹配状态** 当地址0匹配，硬件将指示哪个地址被使用。此位将置1，软件可写1清除该位。

I²C 时序配置控制寄存器 (I²C_TMCTL)

寄存器	偏移地址	R/W	描述	复位值
I²C_TMCTL	I²Cn_BA+0x4C	R/W	I²C时序配置控制寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							**HTCTL**
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
HTCTL							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							**STCTL**
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
STCTL							

位	描述	
[31:25]	**Reserved**	保留
[24:16]	**HTCTL**	**保持时间配置控制器** 该域是发送模式下用来在SCL下降沿和SDA上升沿之间产生延时序。 延迟保持时间外设时钟数 = HTCTL x PCLK.
[15:9]	**Reserved**	保留
[8:0]	**STCTL**	**建立时间配置控制器** 该域是发送模式下用来在SDA下降沿和SCL上升沿之间产生一个延时时序。 延时建立时间外设时钟数 = STCTL x PCLK. **注:**建立时间设置不能让SCL输出少于三个PCLKs。

I²C 总线管理控制寄存器 (I²C_BUSCTL)

寄存器	偏移地址	R/W	描述	复位值
I²C_BUSCTL	I²Cn_BA+0x50	R/W	I²C总线管理控制寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved		PECDIEN	BCDIEN	ACKM9SI	PECCLR	TIDLE	PECTXEN
7	6	5	4	3	2	1	0
BUSEN	SCTLOEN	SCTLOSTS	ALERTEN	BMHEN	BMDEN	PECEN	ACKMEN

位	描述	
[31:14]	**Reserved**	保留
[13]	**PECDIEN**	**包错误检测字节传输完成中断使能位** 0 = 禁用PEC传输完成中断 1 = 使能PEC传输完成中断 **注:**该位在 PECEN =1时使用。
[12]	**BCDIEN**	**包错误检测字节计数完成中断使能位** 0 = 禁用计数完成中断 1 = 使能计数完成中断 **注:**该位在 PECEN =1时使用。
[11]	**ACKM9SI**	**手工应答使能额外SI中断位** 0 = 当BUSEN =1 和 ACKMEN =1时，在第9个时钟周期没有SI中断 1 = 当BUSEN =1 和 ACKMEN =1时，在第9个时钟周期有SI中断
[10]	**PECCLR**	**在重新起始时清除PEC** 当PECEN设置为1时，PEC的计算开始。当检测到STA或STO位时清除。当发生REPEAT START时，PECCLR位用来使能清除PEC计算。 0 = 发生“Repeat Start”功能时禁止清除PEC计算 1 = 发生“Repeat Start”功能时使能清除PEC计算
[9]	**TIDLE**	**空闲状态时间检查** BUSTOUT用来计算有效总线上时钟低的时间和空闲总线上空闲的周期。该位用来定义使能哪个条件被使能。 0 = BUSTOUT用来计算有效总线上时钟低的时间。 1 = BUSTOUT用来计算空闲总线上空闲的周期。**注** BUSY (I2C_BUSSTS[0])位表示当前总线的状态。

[8]	**PECTXEN**	**包错误检查字节发送/接收使能位** 0 = 没有PEC传送 1 = 请求PEC传输 **注:**当ACKMEN =0时，在从机模式该位无影响。
[7]	**BUSEN**	**总线使能位** 0 = 禁止系统管理功能 1 = 使能系统管理功能 **注:**当该位使能,内部14位计数器用来计算时钟低条件的事件时间。
[6]	**SCTLOEN**	**挂起或控制管脚输出使能位** 0 = SUSCON管脚输入 1 = 输出使能SUSCON管脚有效
[5]	**SCTLOSTS**	**挂起/控制数据输出状态** 0 = SUSCON管脚输出为低 1 = SUSCON管脚输出为高
[4]	**ALERTEN**	**总线管脚警报使能位** 从模式(BMHEN =0) 0 = 释放BM_ALERT管脚为高而且禁止警报响应地址头：如果BMDEN和ACKMEN都使能，NACK后接着就是0001100x。 1 = 驱动BM_ALERT管脚为低而且使能警报响应地址头：如果BMDEN和ACKMEN都使能，ACK后接着就是0001100x。 主模式(BMHEN =1) 0 = BM_ALERT 管脚不支持 1 = BM_ALERT 管脚支持.
[3]	**BMHEN**	**总线管理主机使能位** 0 = 禁止主机功能 1 = 使能主机功能
[2]	**BMDEN**	**总线管理设备默认地址使能位** 0 = 禁止设备默认地址。当检测地址0’b1100001x且BMDEN和ACKMEN都使能，设备响应NACK。 1 = 使能设备默认地址。当检测地址0’b1100001x且BMDEN和ACKMEN都使能，设备响应ACK。
[1]	**PECEN**	**包错误检查计算使能位** 0 = 禁止包错误检查计算 1 = 使能包错误检查计算 **注:**当I^2C进入掉电模式，在唤醒之后如果需要PEC计算该位应该使能。
[0]	**ACKMEN**	**手动应答控制使能位** 为了从机接收包括命令和数据应许ACK控制，从机字节控制模式必须通过设置ACKMEN位使能。 0 = 禁止从机字节控制 1 = 使能从机字节控制。用户通过接收的数据在第9位可以响应ACK或NACK。当字节被接收

		时，第8个和第9个SCLK脉冲之间的SCLK低信号拉长 注意: 如果BMDEN =1且该位使能, 在从接收模式下I2C_STATUS的信息将固定在0xF0。

I²C 总线管理定时器控制寄存器 (I²C_BUSTCTL)

寄存器	偏移地址	R/W	描述	复位值
I²C_BUSTCTL	I²Cn_BA+0x54	R/W	I²C总线管理定时器控制寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved			TORSTEN	CLKTOIEN	BUSTOIEN	CLKTOEN	BUSTOEN

位	描述	
[31:5]	**Reserved**	保留
[4]	**TORSTEN**	**超时复位使能位** 0 = 禁用I²C状态机复位 1 = 使能I²C状态机复位(时钟和数据线将被放为高)
[3]	**CLKTOIEN**	**延长时钟超时中断使能位** 0 = 禁止延长时间中断 1 = 使能延长时间中断
[2]	**BUSTOIEN**	**超时中断使能位** BUSY =1 0 = 禁止SCLK 低超时中断 1 = 使能SCLK 低超时中断 BUSY =0 0 = 禁止总线空闲超时中断 1 = 使能总线空闲超时中断
[1]	**CLKTOEN**	**累积时钟低超时使能位** 0 = 禁止累积时钟低电平超时检测 1 = 使能累积时钟低电平超时检测 主机，计算从START到ACK周期 从机，计算从START 到STOP周期
[0]	**BUSTOEN**	**总线超时使能位** 0 = 禁止总线时钟低电平超时检测 1 = 使能总线时钟低电平超时检测(总线时钟为低电平大于T时间 (BIDLE=0) 或者高电平大与

		T时间 (BIDLE =1)

I²C 总线管理状态寄存器 (I²C_BUSSTS)

寄存器	偏移地址	R/W	描述	复位值
I²C_BUSSTS	I²Cn_BA+0x58	R/W	I²C总线管理状态寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
PECDONE	CLKTO	BUSTO	SCTLDIN	ALERT	PECERR	BCDONE	BUSY

位	描述	
[31:6]	**Reserved**	保留
[7]	**PECDONE**	**PEC字节发送/接收完成** 0 = PECEN置位时表明PEC发送/接收还未完成 1 = PECEN置位时表明PEC发送/接收已完成 **注:**该位软件写1清0.
[6]	**CLKTO**	**时钟累积低电平超时状态** 0 = 未发生时钟累积低电平超时 1 = 有发生时钟累积低电平超时 **注:**该位软件写1清0.
[5]	**BUSTO**	**总线超时状态** 0 = 未发生超时或未发生外部时钟超时 1 = 有发生超时或有发生外部时钟超时 在总线忙时该位指示时钟低电平总数发生超时事件，别的方式就指示总线空闲超时事件发生。 **注:**该位软件写1清0.
[4]	**SCTLDIN**	**总线挂起或控制信号输入状态** 0 = SUSCON管脚输入状态为0 1 = SUSCON管脚输入状态为1
[3]	**ALERT**	**SMBus警报状态** 从模式(BMHEN =0) 0 = 指示SMALERT管脚状态为低 1 = 指示SMALERT管脚状态为高 主模式(BMHEN =1)

		0 = 无SMBALERT事件 1 = 当BMHEN = 1(SMBus主机配置)和ALERTEN=1时，指示在SMALERT管脚上有检测到SMBALERT事件(下降沿)发生 **注意:** 1. SMALERT管脚是开漏管脚,使用时需要上拉电阻。2. 该位软件写1清0.
[2]	**PECERR**	**接收PEC错误状态位** 0 = 指示PEC值等于接收到PEC数据包 1 = 指示PEC值与接收到PEC数据包不匹配 **注:**该位软件写1清0
[1]	**BCDONE**	**字节数传送/接收完成状态位** 0 = 当PECEN置位时，传送/接收还没有完成 1 = 当PECEN置位时，传送/接收已经完成 **注:**该位软件写1清0.
[0]	**BUSY**	**总线忙状态位** 指示总线上通信的进程。当检测到起始条件时该位由硬件置位。当检测到停止条件时由硬件清除。 0 = 总线处于空闲状态(SCLK 和 SDA 都为高) 1 = 总线处于忙状态

I²C 包错误检查字节数寄存器 (I²C_PKTSIZE)

寄存器	偏移地址	R/W	描述	复位值
I²C_PKTSIZE	I²Cn_BA+0x5C	R/W	I²C包错误检查字节数寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							**PLDSIZE**
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
PLDSIZE							

位	描述	
[31:9]	**Reserved**	保留
[8:0]	**PLDSIZE**	**传输字节个数** 当PECEN置位时，指示一次事务中传送或接收数据的字节个数。最大传送或接受字节个数为256个字节。 **注：**字节计数包括地址、命令码和数据帧

I²C 包错误检查字节值寄存器 (I²C_PKTCRC)

寄存器	偏移地址	R/W	描述	复位值
I²C_PKTCRC	I²Cn_BA+0x60	R	I²C包错误检查字节值寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
PECCRC							

位	描述	
[31:8]	**Reserved**	保留
[7:0]	**PECCRC**	**包错误价差字节值** 该字节指示在传送或接收字节数据后，包错误检查内容通过使用C(x) = X8 + X2 + X + 1计数。该位域只读。

I²C 总线管理定时器寄存器 (I²C_BUSTOUT)

寄存器	偏移地址	R/W	描述	复位值
I²C_BUSTOUT	I²Cn_BA+0x64	R/W	I²C总线管理定时器寄存器	0x0000_0005

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
BUSTO							

位	描述	
[31:8]	**Reserved**	保留
[7:0]	**BUSTO**	**总线管理超时时间值** 该位域指示总线空闲或时钟低电平超时时间值 **注:**如果用户想修改BUSTOUT的值，先使能BUSEN(I2C_BUSCTL[7])，然后把TORSTEN(I2C_BUSTCTL[4])位必须设置为1，接着再清零。

I²C 总线管理时钟低定时器寄存器 (I²C_CLKTOUT)

寄存器	偏移地址	R/W	描述	复位值
I²C_CLKTOUT	I²Cn_BA+0x68	R/W	I²C总线管理时钟低定时器寄存器	0x0000_0005

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
CLKTO							

位	描述	
[31:8]	**Reserved**	保留
[7:0]	**CLKTO**	**总线时钟低电平时间** 该位域用来配置累积时钟延长时间 **注:**如果用户想要修改CLKLTOUT的值，先使能BUSEN，然后把TORSTEN需设置为1，接着再清零。

6.24 USCI – 通用串行接口

6.24.1 概述

通用串行控制接口(USCI)是一种灵活的接口模块，它集中了几种串行通信协议。用户可以配置该控制器作为UART、SPI或I^2C功能协议。

6.24.2 特性

控制器可以依据应用需求进行单独的配置。主要支持以下几种协议:

- UART
- SPI
- I^2C

6.24.3 框图

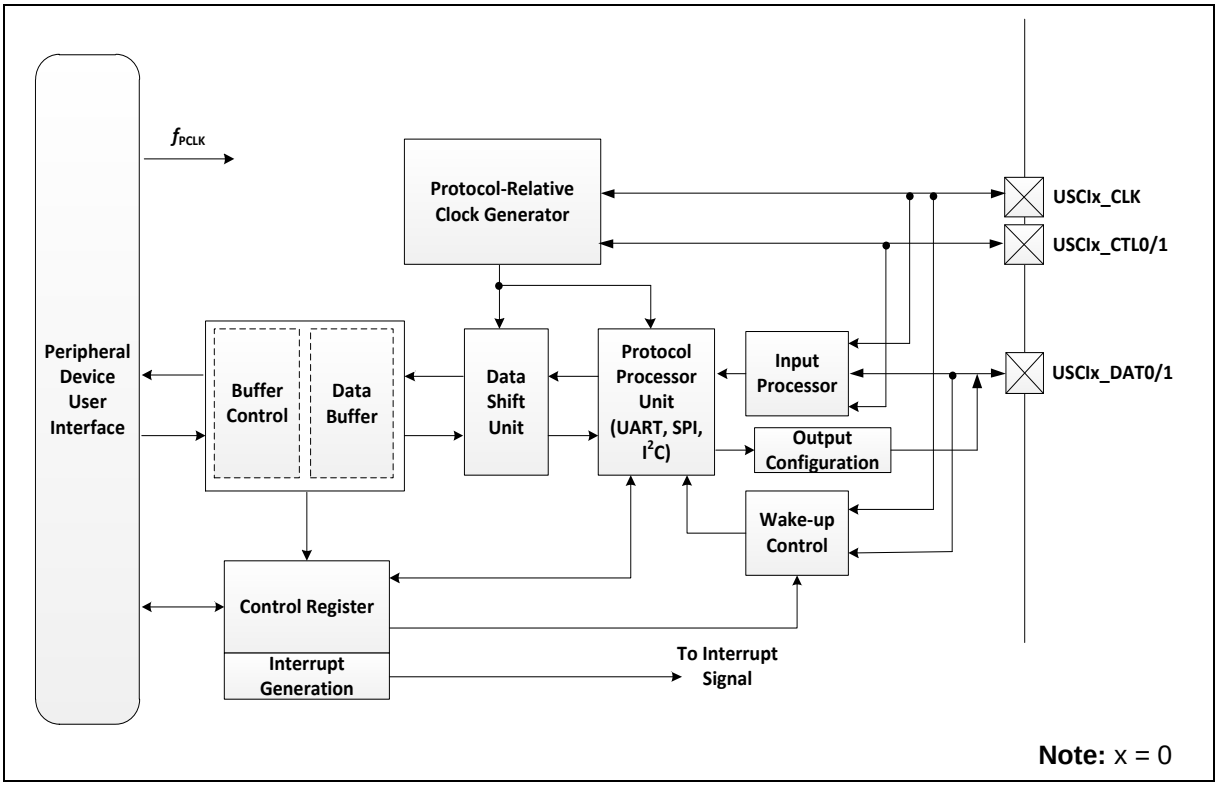


图 6.24-1USCI 框图

6.24.4 功能描述

USCI控制器的结构如图6.24-1USCI输入信号被输入处理器处理。数据缓冲和数据移位单元支持数据传输。每一个具体协议功能被协议处理单元处理。时序和具体协议的时间事件控制信号被协议相关时钟发生器处理。所有具体协议事件被中断产生单元处理。具体协议唤醒功能被唤醒控制器单元处理。

USCI具有三种协议分别是: UART, SPI, 和 I^2C.可以通过FUNMODE (USCI_CTL [2:0])选择。在改变协议前FUNMODE必须设置为0。

6.24.4.1 I/O处理

输入信号

所有输入阶段提供类似特性集合。可以用于所有协议。

表6.24-1为每种选择的协议列出了相关的输入信号。每种输入信号由输入处理器处理加工。比如信号反相选择控制或数字输入滤波。

选择协议		UART	SPI	I^2C
串行总线时钟输入	USCIx_CLK	-	SPI_CLK	SCL
控制输入	USCIx_CTL0	nCTS	SPI_SS	-
	USCIx_CTL1	-	-	-
数据输入	USCIx_DAT0	RX	SPI_MOSI_0	SDA
	USCIx_DAT1	-	SPI_MISO_0	-

表 6.24-1 不同协议的输入信号

具体协议的描述见相关的协议章节。

一般输入结构

数据输入结构和控制信号包含反相器、数字滤波器和边沿检测（仅数据信号）。

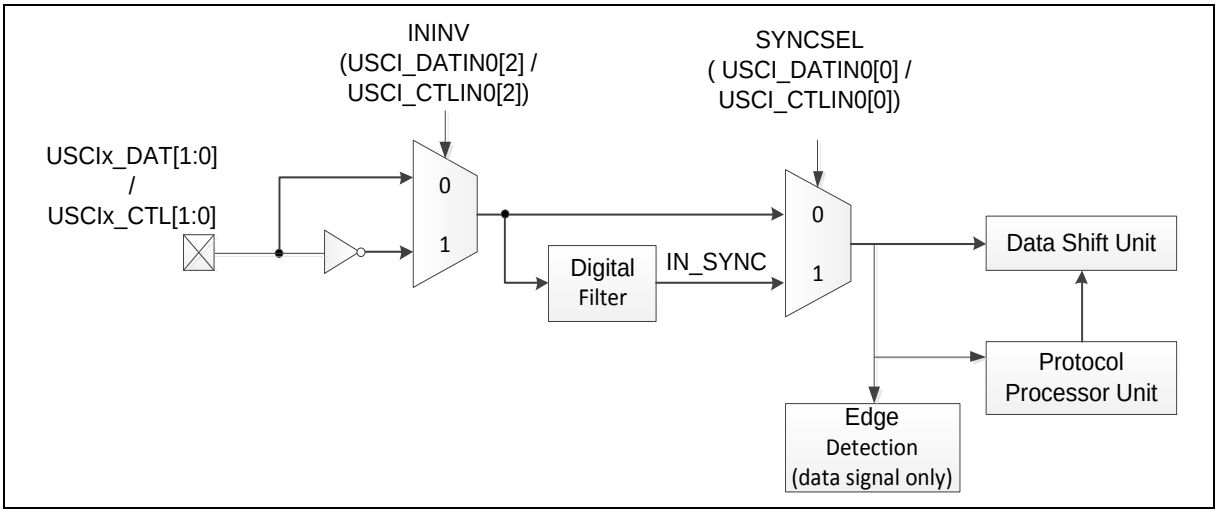


图 6.24-2 输入调节 USCIx_DAT[1:0]和 USCIx_CTL[1:0]

USCIx_CLK输入结构类似于USCIx_CTL[1:0]输入结构，除了不支持反相功能。

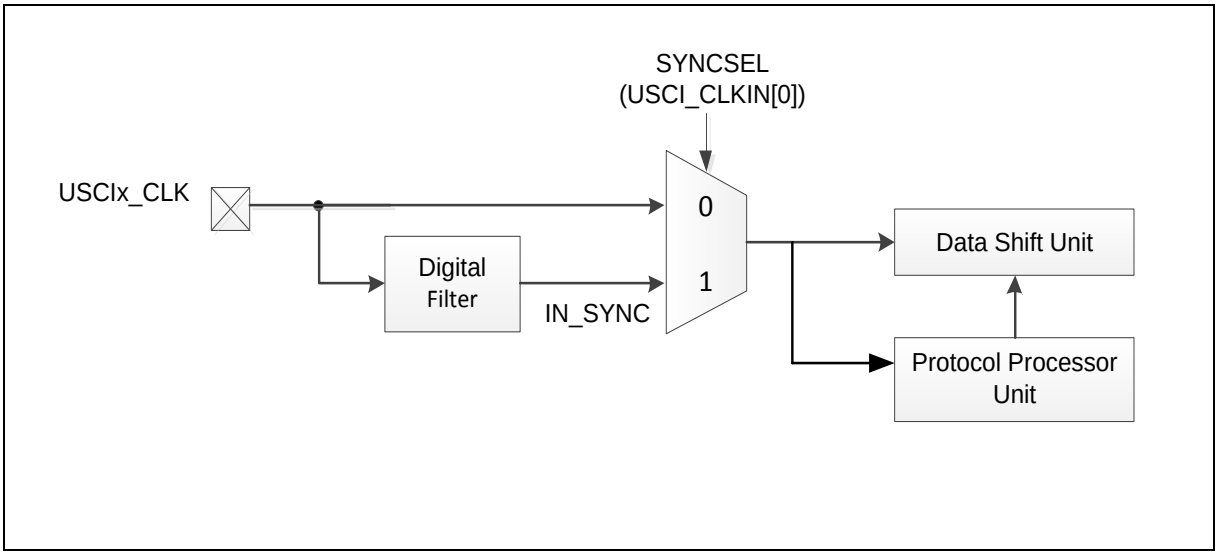


图 6.24-3 输入调节 USCIx_CLK

控制、时钟和数据输入结构的配置分别在USCI_CTLIN0, USCI_CLKIN和USCI_DATIN0寄存器中。EDGEDET (USCI_DATIN0[4:3]) 被用来选择边沿检测条件。在UART模式EDGEDET (USCI_DATIN0[4:3])必须设置为2'b10。可编程的边沿检测表明预期的事件被激活的触发信号触发了。

ININV (USCI_DATIN0[2] / USCI_CTLIN0[2])允许选择的输入信号被极性反转来适应输入信号到数据移位单元的内部极性和协议状态机。

如果SYNCSEL (USCI_DATIN0[0] / USCI_CTLIN0[0] / USCI_CLKIN[0])被设置为0，由于同步和过滤，输入信号的路径不会包含任何延时。如果信号中有噪声，有可能到同步信号（信号IN_SYNC同步到f_{PCLK}）SYNCSEL = 1时同步输入信号延时需要考虑进去。同步导致2-3倍的f_{PCLK}周期延时。

输出信号

表6.24-2为每种选择的协议列出了相关的输出信号。实际使用的输出数目依据所选的协议以及依据协议的意义来分类。

选择协议		UART	SPI	I^2C
串行总线 时钟输出	USCIx_CLK	-	SPI_CLK	SCL
控制输出	USCIx_CTL0	-	SPI_SS	-
	USCIx_CTL1	nRTS	-	-
数据输出	USCIx_DAT0	-	SPI_MOSI_0	SDA
	USCIx_DAT1	TX	SPI_MISO_0	-

表 6.24-2 不同协议输出信号

具体协议的描述见相关的协议章节。

6.24.4.2 数据缓冲

USCI控制器的数据处理是建立在数据移位单元(DSU)和缓冲结构基础上。数据移位和缓存寄存器都是16位的宽度。数据移位单元输入包含移位数据、串行总线时钟和移位控制。传送输出USCIx_DAT0管脚或USCIx_DAT1管脚取决于协议的选择。

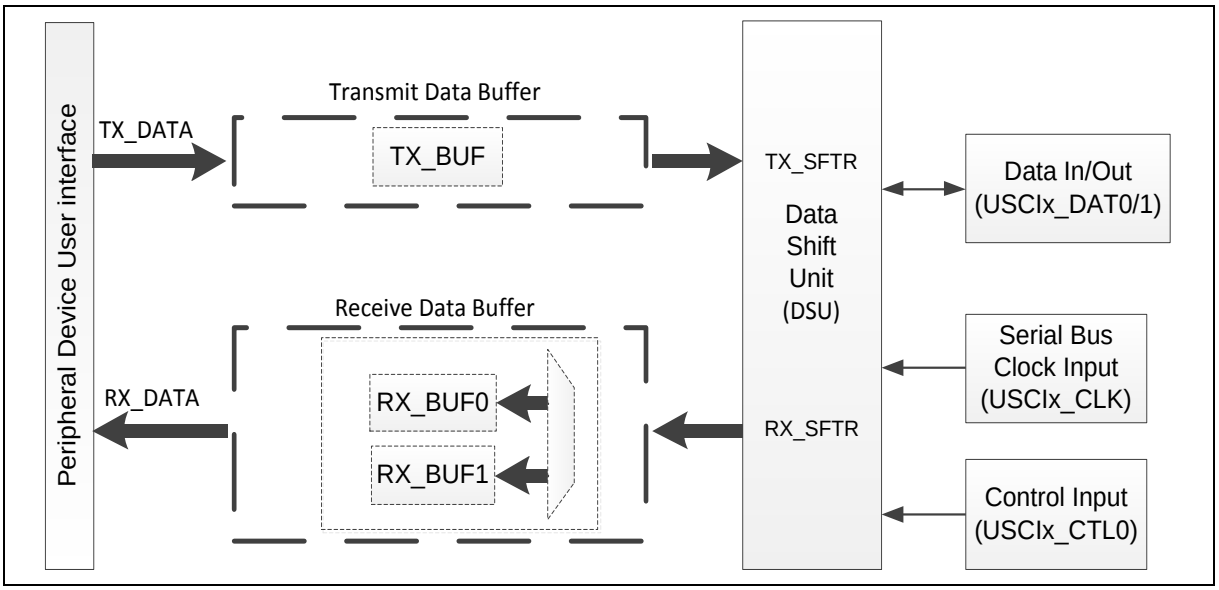


图 6.24-4 数据缓冲框图

数据处理操作包含:

- 外设用户接口(APB)被用作处理数据、中断、状态和控制信息。
- 发送器包含发送移位寄存器(TX_SFTR)和发送数据缓冲(TX_BUF). TXFULL / TXEMPTY (USCI_BUFSTS[9:8])和TXENDIF (USCI_PROTSTS[2])被用来指示发送的状态。
- 接收器包含接收移位寄存器(RX_SFTR)和双缓冲接收结构(RX_BUF0, RX_BUF1).。在双缓冲结构，用户不用关心接收顺序，如果用户没有及时读取USCI_RXDAT寄存器的数据，接收到两次数据会被保持住。

数据访问结构

数据访问结构包括对接收数据读访问和对传送数据写访问。接收到的数据被保存在接收缓冲RX_BUF0和 RX_BUF1中。用户不用关心接收顺序。通过读USCI_RXDAT寄存器来访问接收缓冲区。先接收到数据被先读出，然后在USCI_RXDAT可以查看下一个接收到的数据，可以再一次被读出。

通过写传送寄存器USCI_TXDAT来把发送数据加载到TX_BUF。

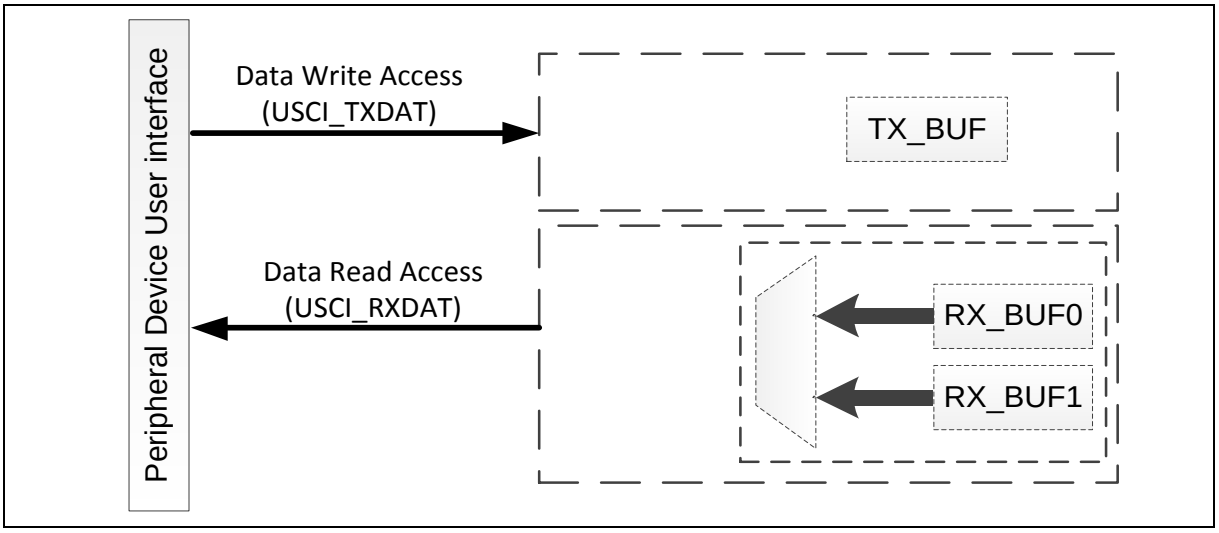


图 6.24-5 数据访问结构

传送数据路径

传送数据路径建立在16位宽度的传送移位寄存器(TX_SFTR)和传送缓冲TX_BUF。数据传输参数比如数据字长是通常是通过线控制寄存器USCI_LINECTL来控制发送和接收的。

传送缓冲

传送移位寄存器不能被用户直接访问。在数值存到到发送缓冲TX_BUF时会被自动的更新。如果当前传送数据完成则新数据将会被传送。

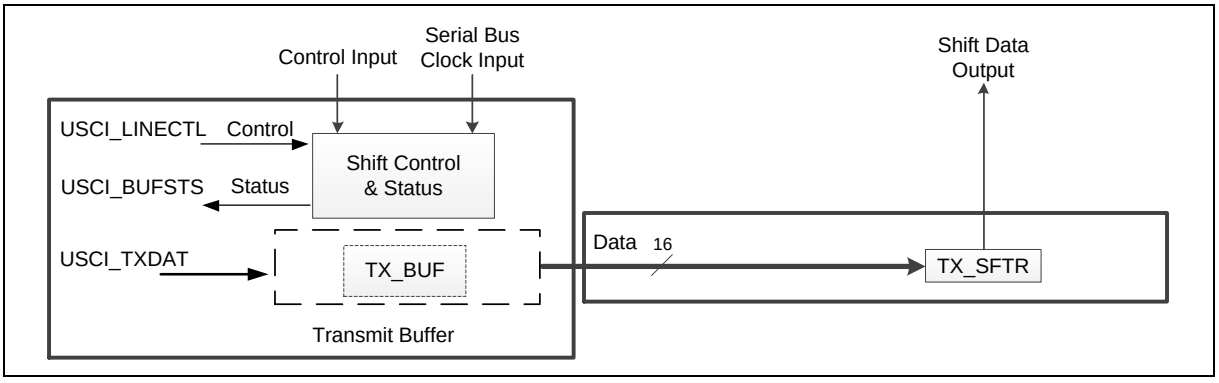


图 6.24-6 传送数据路径

传送数据的有效性

TXEMPTY (USCI_BUFSTS[8])的状态表明在传送缓冲(TX_BUF)中传送数据是有效的，还是没有准备好。TXSTIF (USCI_PROTSTS[1])在每次数据的启动过程时指示。

- 如果USCI控制器是主机，数据传输只能将有效数据载入到传送缓冲(TX_BUF)来启动。在这种情况下，传送移位寄存器会加载传送缓冲的内容。

  注:数据传输启动由主机决定。

- 如果USCI控制器是从机，由主机请求数据传输，从机的启动必须独立于传送缓冲(TX_BUF)的状态。如果一次数据的传输由主机发起，传送移位寄存器需依据具体协议控制信号加载有效的数据。

注:从机不能自己启动但是必须有回应。

- 从传送缓冲到数据移位单元加载数据时序取决于协议的配置。
- **UART:**在正常操作下，如果TXEMPTY = 0，在缓冲中数据字被传送。在自动流控下，当TXEMPTY = 0和 USCIx_CTL0在有效状态时在缓冲中数据字的传送可以启动。
- **SPI:**在主机模式下，当TXEMPTY (USCI_BUFSTS[8])为0时数据传送将会启动。在从机模式下，当从机选择信号处在激活状态和USCIx_CLK引脚出现时钟信号时，数据传送将会启动。
- **I2C:**如果TXEMPTY = 0，在缓冲中数据字被传送。
- 如果TXEMPTY (USCI_BUFSTS [8]) = 0在缓冲中传输数据会被传送。当一次有效的新的传送启动时，传送缓冲的内容(TX_BUF)不能被新的数据重新写入，如果TX_BUF的内容必须改变，在更新数据前用户必须设置TXRST (USCI_BUFCTL [16])为1去清除TX_BUF的内容。而且当传送缓冲(TX_BUF)被新数据更新时，TXEMPTY (USCI_BUFSTS [8])会被自动的清除。当一次传送正在进行时，TX_BUF可以加载新的数据。用户必须在新的传送前更新TX_BUF。

接收数据路径

接收数据路径建立在16位宽度的接收移位寄存器(RX_SFTR)和传送缓冲RX_BUF0和RX_BUF1。数据传输参数比如数据字长或移位方向通常是通过线控制寄存器USCI_LINECTL来控制发送和接收的。寄存器USCI_BUFSTS监控USCI_RXDAT数据的有效性。

接收缓冲

接收移位寄存器不能被用户直接访问。但是如果一个完整数据字被接收或是帧已完整会被自动地载入到接收缓冲中。缓冲的接收数据字可以通过寄存器USCI_RXDAT被读出

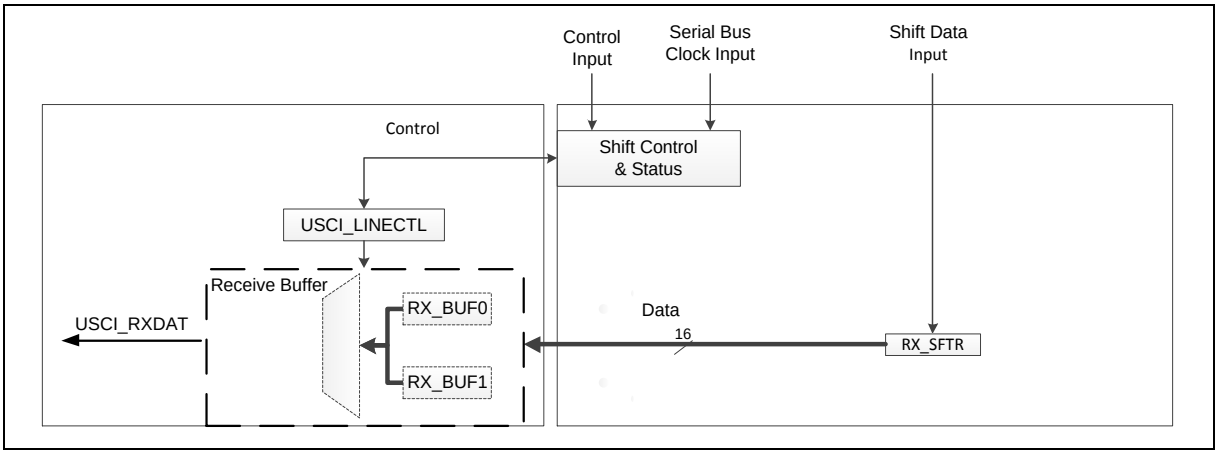


图 6.24-7 接收数据路径

6.24.4.3 端口方向控制

SPI协议是半双工配置，数据端口是双向的。端口方向控制通过专用的硬件接口来控制引脚方向。

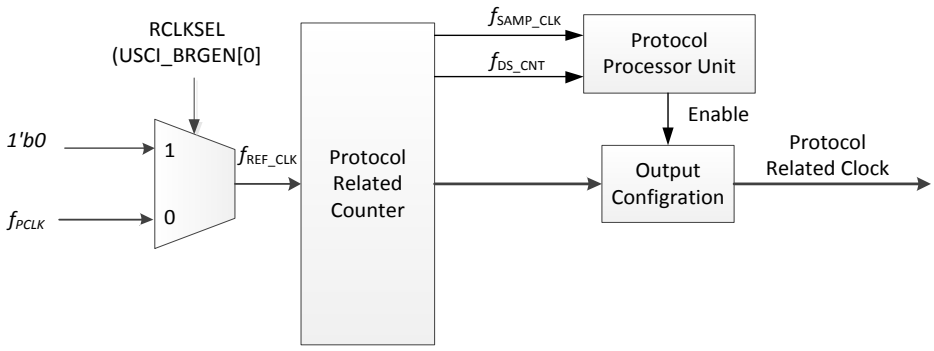
通过PORTDIR (USCI_TXDAT[16])来控制选中引脚的方向。当用户写USCI_TXDAT寄存器时，传送数据和它的端口方向同时被操作。

6.24.4.4 协议控制和状态

USCI控制器包含相关协议时钟发生器，由寄存器USCI_BRGEN控制。当USCI_BRGEN寄存器被写入时会产生复位。相关协议时钟发生器结构如下所示。

6.24.4.5 相关协议的时钟发生器

The USCI controller contains a protocol-relative clock generator and it is controlled by register USCI_BRGEN. It is reset when the USCI_BRGEN register is written. The structured of protocol-relative clock generator is shown below.



注:请参考**基本时钟分频计数器**章节了解更多f_{SAMP_CLK}

图 6.24-8 相关协议时钟发生器

协议相关的计数器包含基本时钟分频计数器和时序测量计数器。图是建立在分频阶段，为不同协议提供所需的频率。主要包含：

- 基本时钟分频计数器提供协议的相关时钟信号和其它相关协议信号(f_{SAMP_CLK}和f_{DS_CLK})。
- 时序测量计数器提供时间间隔测量，比如UART协议的波特率侦测。
- 相关协议时钟发生器输出信号在管脚上可获得（比如SPI的USCIx_CLK）。

基本时钟分频计数器

基本时钟分频计数器是来自对f_{REF_CLK2}，f_{REF_CLK}，f_{DIV_CLK}，f_{SCLK}，和f_{SAMP_CLK} 的分频。这些分频器的频率由PTCLKSEL (USCI_BRGEN [1]), CLKDIV (USCI_BRGEN [25:16]), SPCLKSEL (USCI_BRGEN [3:2])控制。

基本时钟分频计数器用来生成相关协议时序信号

$$f_{\mathrm{DIV_CLK}} = f_{REF_CLK} \times \frac{1}{\mathrm{CLKDIV} + 1} \text{如果} \mathrm{PTCLKSEL} = 0$$

$$f_{\mathrm{DIV_CLK}} = f_{REF_CLK} \times \frac{1}{(\mathrm{CLKDIV} + 1) \times 2} \text{如果} \mathrm{PTCLKSEL} = 1$$

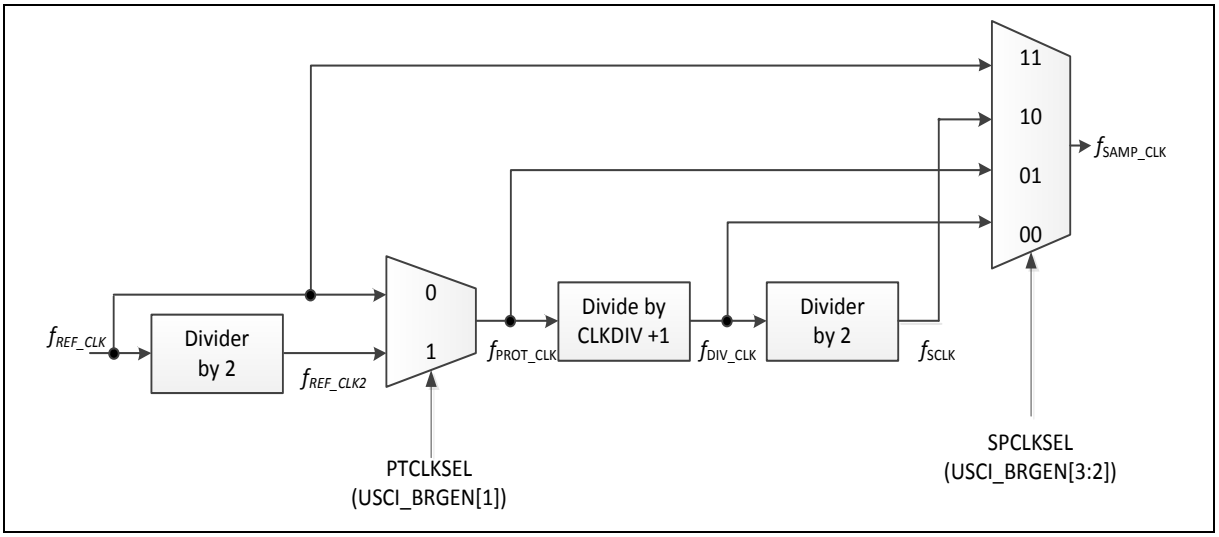


图 6.24-9 基本时钟分频计数器

时序测量计数器

时序测量计数器是用来对时间间隔的测量，通过TMCNTEN (USCI_BRGEN [4]) = 1使能。当TMCNTSRC (USCI_BRGEN [5])被置1，定时器时钟来自f_{DIV_CLK}, 否则来自f_{PROT_CLK}。因此在串行数据接收和发送的同时定时器可以执行时序测量。定时器计数的相关协议信号来自f_{PROT_CLK}或 f_{DIV_CLK}。当它达到用户指定的值时会停止计数。

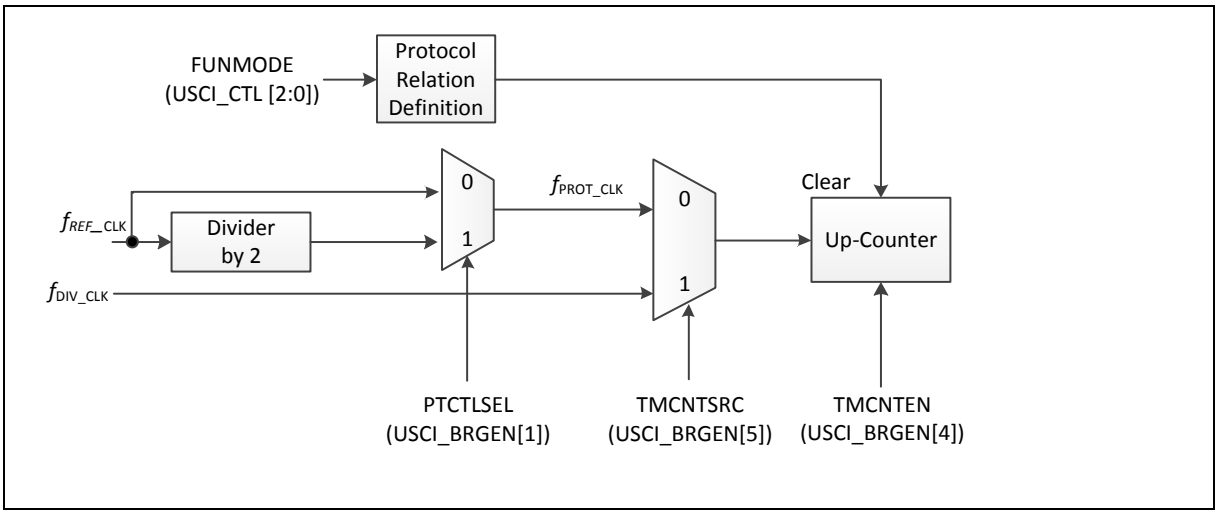


图 6.24-10 时序测量计数器框图

时序测量计数器是用来执行超时功能或自动波特率机制。该功能取决于所选用的协议如下所说。

- UART: 时序测量计数器被用作自动波特率侦测。
- SPI: 时序测量计数器被用作从机超时周期计数。
- I^2C: 时序测量计数器被用作超时时钟周期。

采样时间计数器

采样时间计数器与决定协议时序的协议相关计数器是有关联的。比如移位控制信号或位时序是建立在输

入频率f_{SAMP_CLK}基础上的。采样时间计数器位具体的协议产生时间间隔。采样频率f_{PDS_CNT} 周期是通过选择输入频率f_{SAMP_CLK} 和可编程与分频值(PDSCNT (USCI_BRGEN [9:8]))给定的。这就意味着采样时间取决于所选的协议。详细协议信息请参考相关的章节。

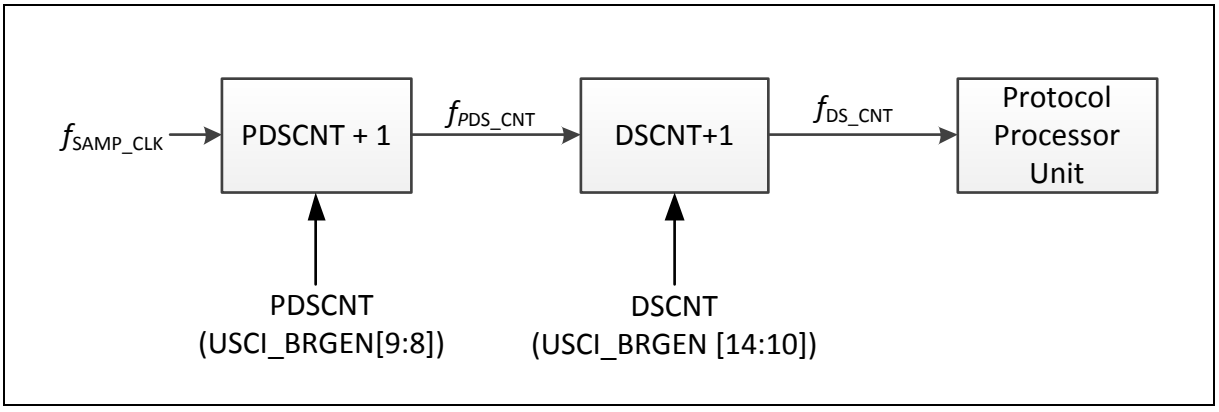


图 6.24-11 采样时间计数器

6.24.4.6 数据传输事件和中断

数据传输事件是建立在一个数据字的发送和接收基础上的。相关的标志在寄存器USCI_PROTSTS中。所有事件可以为中断产生被单独的使能。如果FUNMODE (USCI_CTL [2:0])被置0，USCI会被禁用。当FUNMODE (USCI_CTL [2:0])被设置为某个协议端口时，内部状态会被所选的协议的逻辑硬件所控制。

- 传送起始中断事件来表明一个数据字传送已经启动：

  当数据被载入到发送移位寄存器时一个传送起始中断发生。由TXSTIF (USCI_PROTSTS [1])标志指示，如果中断使能将会产生传送起始中断。

- 传送结束中断事件表明一个数据字传送已经完成：

  当在移位寄存器的发送数据已经完成时传送结束中断事件发生。由TXENDIF (USCI_PROTSTS [2])标志指示，如果中断使能将会产生传送结束中断。当移位控制设置（字长度、移位方向等等）时该事件同时可以指示当前数据字传输被内部“冻结”。在UART和I^2C模式下，依据TXEMPTY (USCI_BUFSTS [8])和协议相关内部信号传送结束中断事件来确定传送数据的有效性。

- 接收起始事件表明一个数据字接收已经启动：

  当接收时钟边沿在新数据字的第一个位移动被检测并且接收使能的时候，接收起始事件发生。由RXSTIF (USCI_PROTSTS [3])标志指示，如果中断使能将会产生接收起始中断。

- 接收事件表明一个数据字接收已经完成

  当一个新接收的字在接收缓冲中可用时，接收事件发生。由RXENDIF (USCI_PROTSTS [4])标志指示，如果中断使能将会产生接收中断。

- 数据丢失事件表明最新接收到的数据丢失：

  如果在寄存器USCI_RXDAT (旧的数据来自 RX_BUF0 或 RX_BUF1)的可用的数据没有被读出那么接收缓冲会溢出，新来的数据会丢失掉，该事件发生。由RXOVIF (USCI_BUFSTS[3])标志指示，如果中断使能将会产生协议中断。

事件产生和中断结构如下图6.24-12所示。

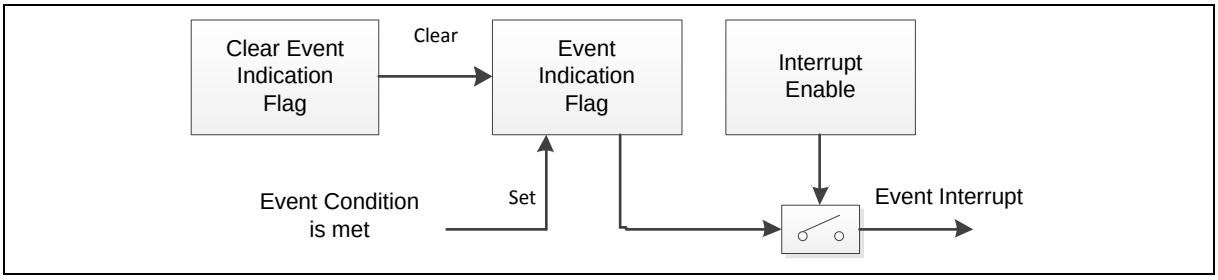


图 6.24-12 事件和中断结构

每个中断使能是通过设置USCI_INTEN [4:1].中的RXENDIEN, RXSTIEN, TXENDIEN, 和 TXSTIEN。这些事件包括接收结束中断事件、接收起始中断事件、传送结束中断事件和传送起始中断事件。对于具体的中断在每个协议中断使能寄存器中指定。

如果一个定义条件符合，一个事件被侦测到和事件指示标志自动被设置。该标志保持置位直到被软件清除。如果对应中断使能，在事件被侦测到时会产生中断。

寄存器、位和位域表明传输事件和控制产生下表6.24-3.所示的USCI中断。

事件	指示标志	指示清除	中断使能
传送结束中断事件	TXSTIF (USCI_PROTSTS [1])	由软件写1到USCI_PROTSTS对应的中断位清除	TXSTIEN (USCI_INTEN [1])
传送结束中断事件	TXENDIF (USCI_PROTSTS [2])		TXENDIEN (USCI_INTEN [2])
接收起始中断事件	RXSTIF (USCI_PROTSTS [3])		RXSTIEN (USCI_INTEN [3])
接收结束中断事件	RXENDIF (USCI_PROTSTS [4])		RXENDIEN (USCI_INTEN [4])

表 6.24-3 Data Transfer Events and Interrupt Handling

6.24.4.7 具体协议事件和中断

这些事件与在对应协议的章节描述的具体协议动作有关。相关的标志位在寄存器USCI_PROTSTS中。对于共同协议的中断，所有事件可以单独的使能。

事件	指示标志	指示清除	中断使能
UART模式下的协议	USCI_PROTSTS [17:16]和USCI_PROTSTS [11:5]	由软件写1到 USCI_PROTSTS对应的中断位清除	USCI_PROTIEN[2:1]
SPI模式下的协议	USCI_PROTSTS [9:8], USCI_PROTSTS [6:5]		USCI_PROTIEN [3:0]
I^2C模式下的协议	USCI_PROTSTS [13:8], USCI_PROTSTS [5]		USCI_PROTIEN [6:0]

表 6.24-4 具体协议事件和中断处理

6.24.4.8 唤醒

相关协议唤醒功能信息在唤醒控制寄存器(USCI_WKCTL)和唤醒状态寄存器(USCI_WKSTS)中。在可用

的协议之间可以共享。因此，这些寄存器的位代表的意义在不同协议之间会有所不同

6.24.4.9 PDMA

USCI支持PDMA传输。当PDMAEN (USCI_PDMACTL [3])置1时，PDMA功能使能。

当TXPDMAEN (USCI_PDMACTL [1])置1时，控制器会产生请求到PDMA控制器来自动地启动PDMA

当RXPDMAEN (USCI_PDMACTL [2]) 置1时，控制器会启动PDMA接收进程。当数据接收到FIFO缓冲时，USCI会自动地产生请求到PDMA控制器

在UART功能下，如果有错误事件（包括帧错误、奇偶校验错误和break侦测），RXPDMAEN需求会被清除并保持住。

6.25 USCI – UART模式

6.25.1 概述

异步串行通道UART能处理异步数据帧的接收和发送。接收过程是把外设的串行数据转为并行数据，发送过程是把CPU的并行数据转成串行数据发送出去。接收和发送帧是独立的，他们可以在不同时刻启动。

UART控制器也提供流控功能，有三种条件可以唤醒系统。

6.25.2 特性

- 有一个发送缓冲和两个接收缓冲来存放数据
- 支持硬件流控功能
- 支持可编程的波特率发生器
- 支持9位数据传输(支持9位RS-485)
- 内建波特率发生捕获事件为波特率侦测提供了可行性
- 支持PDMA传输
- 支持唤醒功能(仅传入数据和nCTS唤醒)

6.25.3 框图

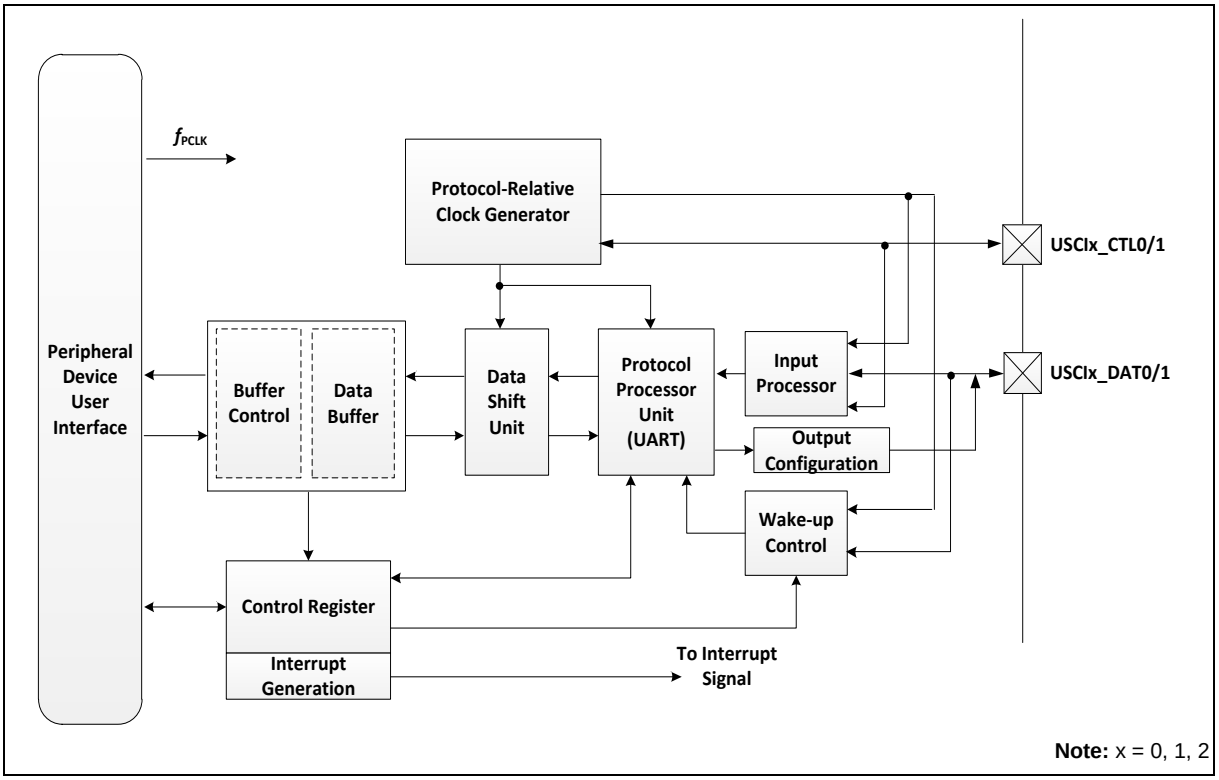


图 6.25-1USCI-UART 模式框图

6.25.4 基本配置

6.25.4.1 USCI0_UART基本配置

- 时钟源配置
- 使能USCI0时钟控制寄存器(USCI0CKEN)，在寄存器CLK_APBCLK1[8]。
- 复位USCI0控制寄存器(USCI0RST)，在寄存器SYS_IPRST2[8]。
- 使能USCI0_UART功能寄存器USCI0_CTL[2:0]，USCI1_CTL[2:0]=3’b010.
- 引脚配置。

组	管脚名称	GPIO	MFP
	USCI0_CTL0	PD.4	MFP3
		PC.14	MFP5
		PC.13	MFP6
		PE.6	MFP7
	USCI0_CTL1	PD.3	MFP3
		PB.15	MFP5
		PA.8	MFP6
		PE.5	MFP7
	USCI0_DAT0	PD.1	MFP3
		PB.13	MFP5
		PA.10	MFP6
		PE.3	MFP7
	USCI0_DAT1	PD.2	MFP3
		PB.14	MFP5
		PA.9	MFP6
		PE.4	MFP7

6.25.4.2 USCI1_UART基本配置

- 时钟源配置
- 使能USCI1时钟控制寄存器(USCI1CKEN)，在寄存器CLK_APBCLK1[9]。
- 复位USCI1控制寄存器(USCI1RST)，在寄存器SYS_IPRST2[9]。
- 使能USCI1_UART功能寄存器USCI1_CTL[2:0]，USCI1_CTL[2:0]=3’b010。
- 引脚功能

组	管脚名称	GPIO	MFP
	USCI1_CTL0	PB.10	MFP4
		PD.3, PE.9	MFP6

		PB.5	MFP8
	USCI1_CTL1	PB.9	MFP4
		PD.4, PE.8	MFP6
		PB.4	MFP8
	USCI1_DAT0	PB.7	MFP4
		PD.5, PE.10	MFP6
		PB.2	MFP8
	USCI1_DAT1	PB.6	MFP4
		PD.6, PE.11	MFP6
		PB.3	MFP8

6.25.5 功能描述

6.25.5.1 UUART 共同功能描述

请参考章节6.24.4详细描述。

6.25.5.2 信号描述

UART连接的特点是在发送和接收之间使用单根连接线。接收输入信号(RXD)由输入USCIx_DAT0处理，发送输出(TXD)信号由USCIx_DAT1处理。

对于全双工通信，独立的通信线在每个传输方向是必须的。图6.25-2展示的是UART模块A和UART模块B之间用点对点全双工通信的连接。

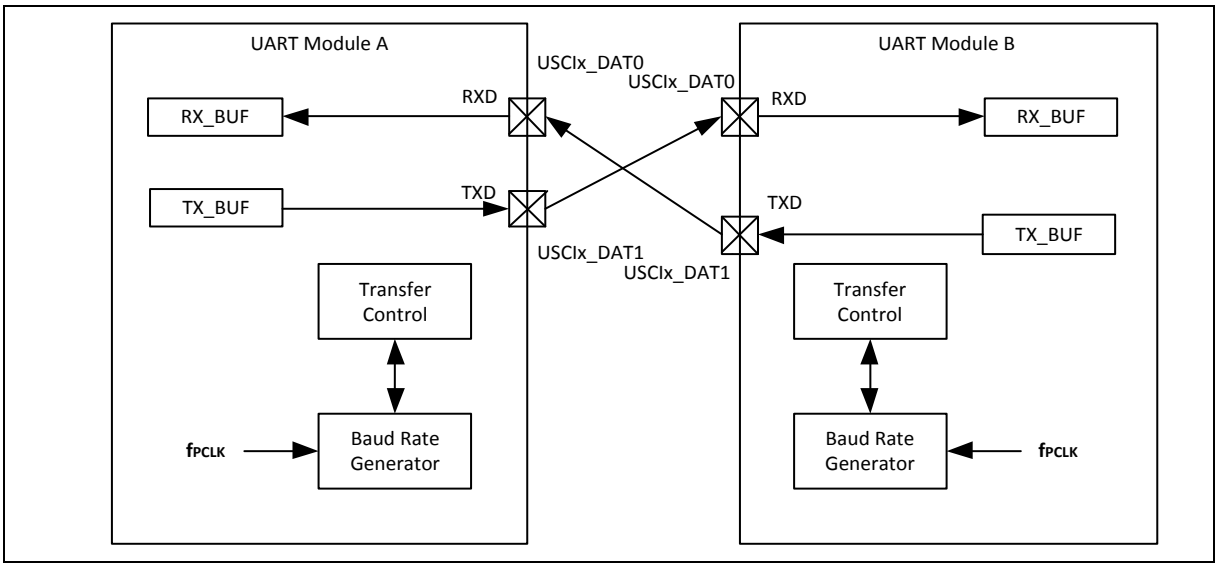


图 6.25-2 全双工通信的 UART 信号连接

输入信号

对于UART协议，表6.25-1中列出了输入信号的数目。每种输入信号由输入处理器处理加工。比如信号反相选择控制或数字输入滤波。他们可以依据在协议中的意义进行分类(见表6.25-1)。

选择协议		UART
控制输入	USCIx_CTL0	nCTS
	USCIx_CTL1	-
数据输入	USCIx_DAT0	RX
	USCIx_DAT1	-

表 6.25-1 UART 协议输入信号

输出信号

对于UART协议，每个相关协议输出信号是可用的。使用输出的数目实际是取决于所选的协议。他们可以依据在协议中的意义进行分类。

选择协议		UART
控制输入	USCIx_CTL0	-
	USCIx_CTL1	nRTS
数据输入	USCIx_DAT0	-
	USCIx_DAT1	TX

表 6.25-2UART 协议的输出信号

6.25.5.3 帧格式

标准UART帧如图6.25-3。包含：

- 信号为1的空闲时间。
- 信号为0的一个起始帧位(SOF)。
- 6~13位数据。
- 一个校验位(P)，可选择奇校验或是偶校验。还可以选择需要该位。
- 信号为1的一到两位的停止位。

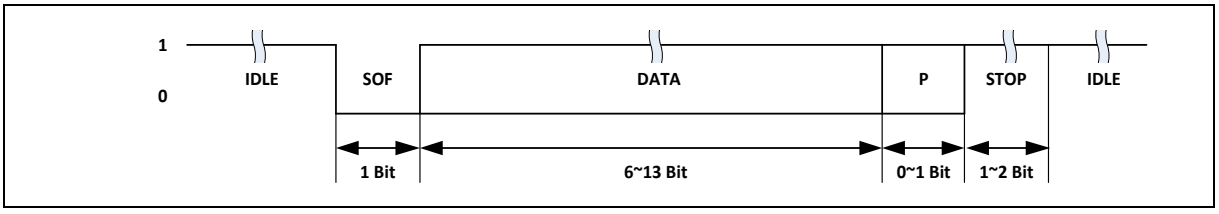


图 6.25-3 UART 标准帧格式

协议的特定位(SOF，P，STOP)由UART协议状态机自动地处理，不会通过接收和发送缓冲出现在数据流中。

起始位

接收器输入信号USCIx_DAT0检测是以一个下降沿开始的。当一个下降沿发生时同时接收器处于空闲时或是在最后一个停止位采样点之后，SOF位被侦测到。为了增强抗噪声能力，SOF位时序起始是在第一次下降沿被侦测到。如果采样位SOF的值为1，那么先前下降沿被当作是噪声并且接收器会再次进入空

闲。

数据域

数据域的长度（数据位数目）可以通过DWIDTH (UUART_LINECTL[11:8])编程。取值范围在6到13数据位。

注：在UART协议中，为了先传输LSB需设置LSB (USCI_LINECTL[0])为1。

校验位

UART可以在发送的时候生成校验位，在接收的时候检查校验位。通过PARITYEN (USCI_PROTCTL[1]) 和 EVENPARITY (USCI_PROTCTL[2])来选择校验的类型（无校验、奇校验或偶校验）。如果禁用校验功能，UART帧不会包含校验位。为了一致性，所有通信的设备必须配置成一样的校验类型。

在数据域的最后数据位之后，如果校验功能使能，发送器会自动地计算校验位并发出去。接收器解释该位作为接收校验并且和内部计算的相比较。校验检测和帧检测(停止位)在协议状态寄存器(USCI_PROTSTS)中被监测。该寄存器包含监测相关协议状态和相关协议错误指示(FRMERR, PARITYERR)这些位。

停止位

UART帧是以1到2位信号为1的停止位结束的(电平同空闲时电平)。停止位数目由STOPB (USCI_PROTCTL[0])位编程。在停止位过后一个新的起始位可以被直接的传输。

传输状态指示

RXBUSY (UUART_PROTSTS[10])指示接收状态。

通过RXBUSY位监测接收状态。在起始帧位开始到最后的停止位结束期间RXBUSY被置位。

6.25.5.4 操作模式

为了操作UART协议，下面遇到的问题必须考虑到:

选择 UART 模式

通过设置FUNMODOE(USCI_CTL[2:0])位为010B选择UART协议，通过设置PROTEN (USCI_PROTCTL [31])位为1使能UART协议。注意在改变协议前必须设置FUNMODE为0，强烈建议在使能UART协议前配置好UART的所有参数。

引脚连接

在UART协议中USCIx_DAT0引脚被用作接收数据输入信号(RX)。输入信号的属性在USCI_DATIN0中配置。起始位侦测建议设置EDGEDET (USCI_DATIN0[4:3])为10B。

在UART协议中USCIx_DAT1引脚被用作发送数据输出信号(TX)。输出信号的属性在USCI_LINECTL中配置。

在UART协议中USCIx_CTL0引脚被用作UART清除发送信号 (nCTS)。输入控制信号的属性在USCI_CTLIN0中配置。

在UART协议中USCIx_CTL1引脚被用作UART请求发送信号(nRTS)。输出控制信号的属性在USCI_LINECTL中配置。

位时序配置

预期波特率设置必须被选择包括波特率发生器和位时序在内。

帧格式配置

字长度、停止位数目和校验模式必须依据应用需求来设定，这些都是通过USCI_LINECTL和USCI_PROTCTL寄存器来编程。如果应用需要，数据输入和输出信号可以反转。通过设置LSB(USCI_LINECTL[0])为1来决定先传送LSB。

6.25.5.5 位时序

在UART模式，每一帧位被划分成数据采样时间是为了在sub-bit范围内提供间隔来调整应用需求的采样点。数据采样时间每一位数目在DSCNT (USCI_BRGEN[14:10])中定义并且数据采样时间长度由PDSCNT (USCI_BRGEN[9:8])给定。

在图6.25-4给出的范例中，一位时间是由16个数据采样时间组成（DSCNT(USCI_BRGEN[14:10]) = 15）。每一位时间不建议编程少于等于4个数据采样时间。

每位数值的采样点位置固定在1/2采样时间处。这样一来采样的平均值作为采样的值是可行的。

位时序建立(数据采样时间数目)对于发送器和接收器是一样的，因为他们的硬件电路是一样的。

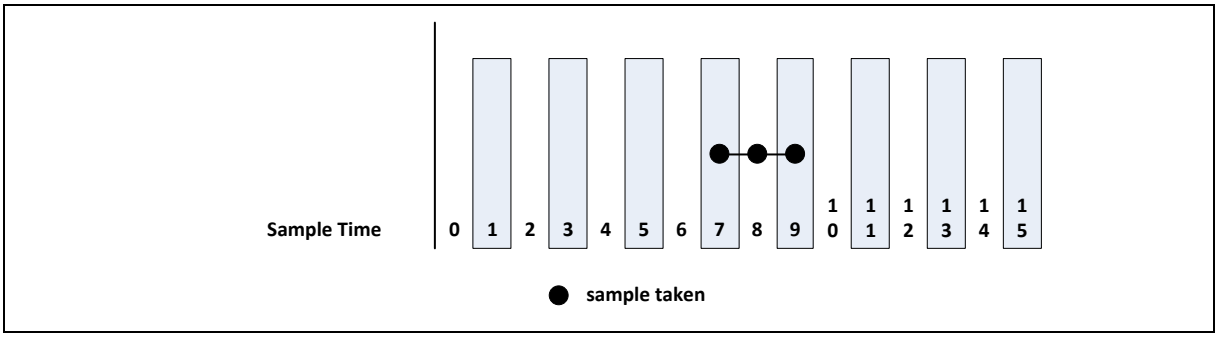


图 6.25-4 UART 位时序(数据采样时间)

6.25.5.6 波特率发生

在UART模式下波特率f_{UART}取决于数据采样时间每位时间的数目和他们的时序。波特率设定是在发送器和接收器空闲的时候改变。在RCLKSEL，SPCLKSEL，PDSCNT，和DSCNT中设置波特率：

RCLKSEL (UUART_BRGEN [0])

定义输入频率f_{REF_CLK}

SPCLKSEL (UUART_BRGEN [3:2])

定义采样时钟f_{SAM_CLK}的时钟源

PDSCNT (UUART_BRGEN [9:8])

定义数据采样时间的长度(由f_{REF_CLK}经1，2，3或4分频)

DSCNT (UUART_BRGEN [14:10])

定义采样时间中每位时间的数目

标准设定由RCLKSEL = 0 ($f_{REF_CLK} = f_{PCLK}$)，PTCLKSEL = 0 ($f_{PROT_CLK} = f_{REF_CLK}$)和SPCLKSEL = 2'b00 ($f_{SAM_CLK} = f_{DIV_CLK}$)决定。在这些条件下，波特率值如下：

$$f_{\mathrm{UART}} = f_{REF_CLK} \times \frac{1}{\mathrm{CLKDIV}+1} \times \frac{1}{\mathrm{PDSCNT}+1} \times \frac{1}{\mathrm{DSCNT}+1}$$

为了产生更低的频率，设置PTCLKSEL = 1 ($f_{PROT_CLK} = f_{REF_CLK} / 2$)，选择除2阶段，如下：

$$f_{\mathrm{UART}} = \frac{f_{REF_CLK}}{2} \times \frac{1}{\mathrm{CLKDIV}+1} \times \frac{1}{\mathrm{PDSCNT}+1} \times \frac{1}{\mathrm{DSCNT}+1}$$

如果SPCLKSEL = 2'b10 ($f_{SAMP_CLK} = f_{SCLK}$)，和RCLKSEL = 0 ($f_{REF_CLK} = f_{PCLK}$),，PTCLKSEL = 0 ($f_{PROT_CLK} = f_{REF_CLK}$)。波特率如下：

$$f_{\mathrm{UART}} = f_{REF_CLK} \times \frac{1}{\mathrm{CLKDIV}+1} \times \frac{1}{2} \times \frac{1}{\mathrm{PDSCNT}+1} \times \frac{1}{\mathrm{DSCNT}+1}$$

设置波特率参数之后UART波特率有一定的误差宽容度。表6.25-1 波特率关系列出了相关误差率对于使用计算相关波特率的设定的范例。表6.25-1 波特率关系**表6.27-1**

HCLK源	PCLK源	期待波特率	CLKDIV (UUART_BRGEN[25:16])	DSCNT (UUART_BRGEN[14:10])	PDSCNT	实际波特率	误差百分比
12 MHz	HCLK	115200	0xC	0x7	0x0	115384	0.16%
12 MHz	HCLK	9600	0x7C	0x9	0x0	9600	0%
12 MHz	HCLK	2400	0x1F3	0x9	0x0	2400	0%

表 6.25-1 波特率关系

注: {SPCLKSEL, PTCLKSEL, RCLKSEL = 2'b0,1'b0,1'b0}

6.25.5.7 自动波特率侦测

UART控制器支持自动波特率侦测功能。它被用来鉴别来自接收信号(USCIx_DAT0)的输入波特率，然后在满足侦测波特率功能完成后修改波特率时钟分频器CLKDIV (UUART_BRGEN[25:16])。依据时序测量计数器章节，时序测量计数器被用作输入信号(USCIx_DAT0)的时间间隔测量，在每次侦测信号的下降沿，实际定时器值被捕捉到BRDETITV (UUART_PROTCTL [24:16])中。

当ABREN (PROTOCOL[6])位被使能，必须使用0x55数据格式来进行自动波特率侦测。输入信号下降沿启动波特率计数器并且在下个下降沿时将会加载时序测量计数器值到BRDETITV (UUART_PROTCTL [24:16])中。建议使用f_{DIV_CLK} (TMCNTSRC (UUART_BRGENC[5]) =1)作为计数器源。

在自动波特率功能完成后（输入信号第四个下降沿的时间），BRDETITV (UUART_PROTCTL [25:16])将修改CLKDIV (UUART_BRGEN[25:16])。如果用户希望正确接收下一个连续帧，最好将CLKDIV (UUART_BRGEN[25:16])和DSCNT (UUART_BRGEN[14:10])的值设置为相同的值（该值应在0xF和0x5的列中，因为DSCNT用于定义每个位的样本计数器，而PDSCNT(UUART_BRGEN[9:8])为0x0。

在自动波特率侦测期间，在输入信号每次下降沿之后ABRDETIF (UUART_PROTSTS[9]) 和BRDETITV (UUART_PROTCTL [24:16])会被更新并且自动波特率模式0x55在帧传输完后不会被接收到接收器缓冲里。在第四个输入信号下降沿被侦测到时ABREN位会被硬件清除。因此用户可以读ABREN的状态了解自动波特率功能是否完成。

如果CLKDIV 和 DSCNT没有在自动波特率功能计算中设置为同样的值，用户应该在自动波特率功能完成之后通过BRDETITV和CLKDIV的值计算出合理的平均波特率。

如果输入信号的波特率非常慢并且时序测量计数器的位时间不能计算出输入位时间的正确周期，有一个ABERRSTS (UUART_PROTSTS[11])位来指示自动波特率侦测的错误信息。此刻，用户需要修改CLKDIV的值和要求主机再次发送0x55。

依据时序测量计数器的有效范围，对于BRDETITV最大自动波特率侦测是0x1FE。UART自动波特率控制如图6.25-5所示。

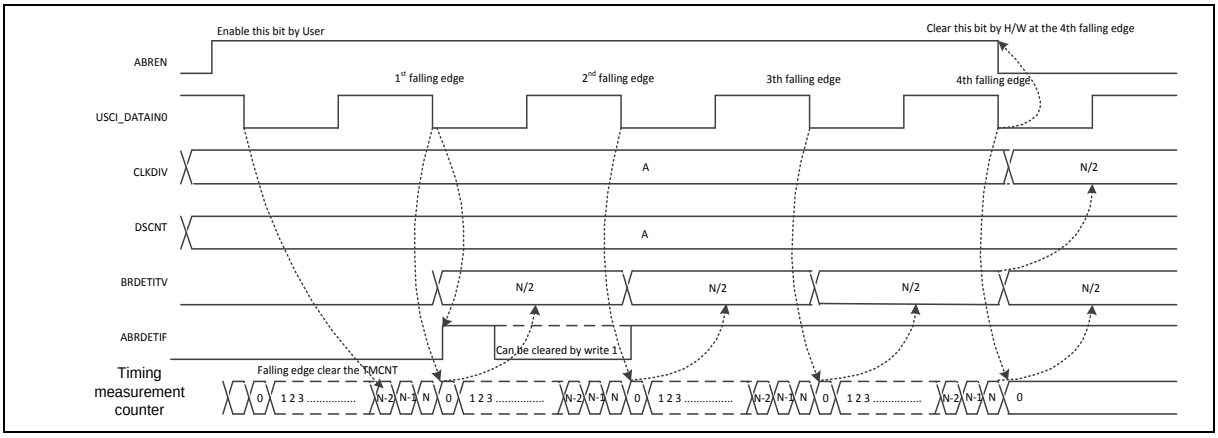


图 6.25-5 UART 自动波特率控制

6.25.5.8 自动流控制

UART支持硬件自动流控，为接收器缓冲提供nRTS流控是由RXFULL (UUART_BUFSTS[1])指示的。当缓冲满的时候(RXFULL = 1)，nRTS失效。

UART为发送器也提供了nCTS流控。当nCTS有效时，nCTS被用作控制数据发送。

6.25.5.9 RS-485支持

UART控制器可以扮演RS-485主机发送器的角色通过设置校验位（第9位）为1来鉴别地址帧。对于对于数据帧，校验位设置为0.当STICKEN (USCI_PROTCTL[26])被设置时，软件可以使用每一数据的15位来控制校验位（PARITYEN (USCI_PROTCTL[1])被设置）。例如，如果STICKEN被置1,并且数据序列是0x8015，0x8033，0x0055，0x0033和0x80AA数据0x15，0x33，0x55，0x33和0xAA发送校验将是1，1，0，0和1。

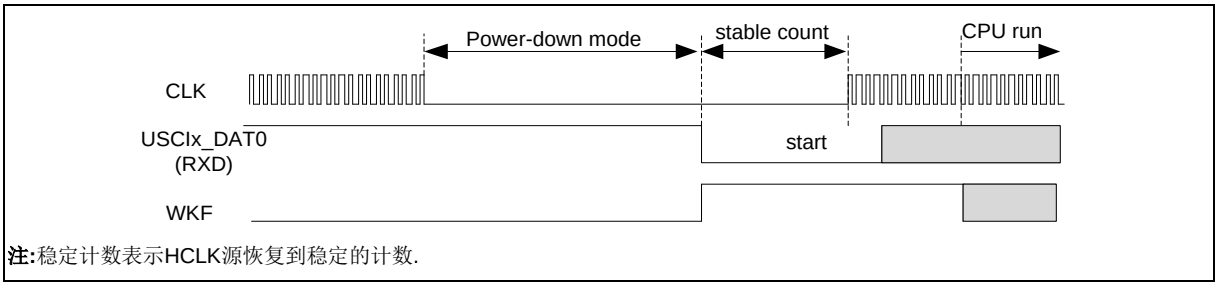
UART控制器可以扮演RS-485地址可寻从机角色，当PARITYEN(USCI_PROTCTL[1])，EVENPARITY (USCI_PROTCTL[2]) 和 STICKEN (USCI_PROTCTL[26]) 被设置时，PARITYERR (USCI_PROTSTS[5])相关协议错误被当作地址位侦测。如果PARITYERR被设置，意味着地址位在接收总线上的地址位被侦测到，否则数据被接收到缓冲中。

6.25.5.10 唤醒功能

USCI控制器在UART模式下支持唤醒系统功能。唤醒源包括输入数据和nCTS引脚。各唤醒源的描述如下：

(a) 输入数据唤醒

当系统处在掉电模式时并且WKEN (USCI_WKCTL [0]) 和 DATWKEN (USCI_PROTCTL[9])都被设置，输入数据引脚触发唤醒系统。在系统唤醒后为了接收输入的数据，WAKECNT(USCI_PROTCTL[14:11])需要被设置。WAKECNT (USCI_PROCTL[14:11])指明当设备被从掉电模式下唤醒时从机获得第一个位及起始位需要多少个时钟周期（该时钟是f_{PDS_CNT}）。输入数据唤醒如图6.25-6所示。



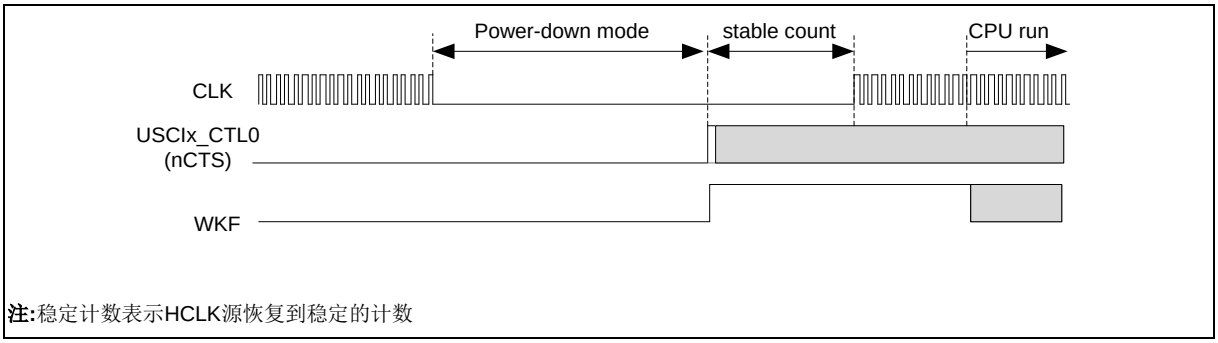
注:稳定计数表示HCLK源恢复到稳定的计数.

图 6.25-6 输入数据唤醒

(b) nCTS 引脚唤醒

当系统处在掉电模式下时并且WKEN (USCI_WKCTL [0]) 和 CTSWKEN (USCI_PROTCTL[10])被设置，nCTS引脚的变化可以唤醒系统。nCTS唤醒如图6.25-7和图6.25-8所示

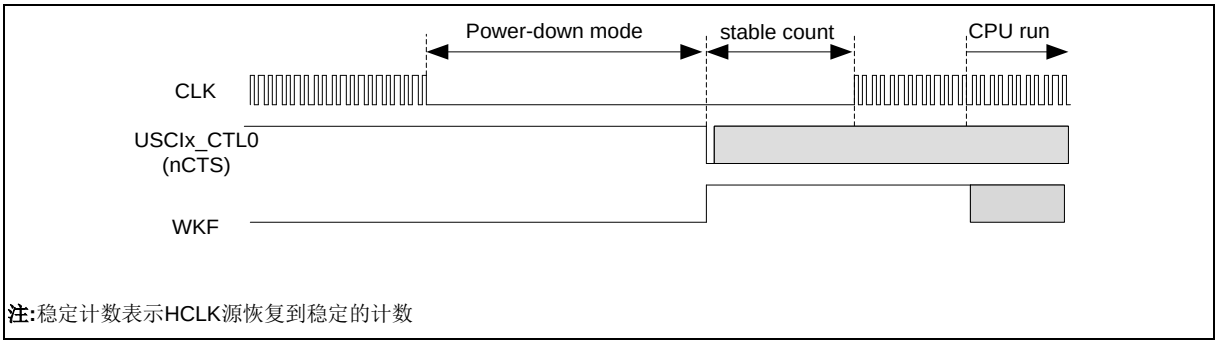
Case 1(nCTS transition from low to high):



注:稳定计数表示HCLK源恢复到稳定的计数

图 6.25-7nCTS 唤醒状态 1

Case 2 (nCTS transition from high to low):



注:稳定计数表示HCLK源恢复到稳定的计数

图 6.25-8nCTS 唤醒状态 2

6.25.5.11 中断事件

UART为协议事件和数据传输事件提供了中断。描述如下:

协议中断事件

在UART模式下，可以产生下面相关协议事件并且可以产生协议中断。

请注意USCI_PROTSTS位不会被硬件自动地清除，必须通过软件清除，这样是为了监控新到来的事件。

接收线状态

两个相关协议错误FRMERR (UUART_PROTSTS[6])和PARITYERR (UUART_PROTSTS[5])标志在相应的接收器缓冲状态寄存器中被置位指示每次接收到的数据

在UART模式下，校验位检查结果是通过相关协议错误指示的（0=接收校验位等于计算校验值），帧检查是通过相关协议错误指示的（0=接收停止位等于格式值‘1’）。

当接收数据为0、接收校验和停止位都为0时BREAK错误标志BREAK (UUART_PROTSTS[7])被置位。

在BREAK，FRMERR，PARITYERR (UUART_PROTSTS[7:5])中有校验错误、帧错误或BREAK数据侦测的中断指示。

自动波特率侦测

自动波特率中断ABRDETIF (UUART_PROTSTS [9])对于自动波特率捕获功能用来指示时序测量计数器已得到2位时间。

在接收信号的第一次下降沿是自动波特率侦测功能被使能。在下次下降沿被侦测之后自动波特率侦测功能进行测量。当帧传输完成时结束。在传输完成后，被除两次的时序测量计数器的值等于采样时间每位的数目。用户可以读出BRDETITV (UUART_PROTCTL[24:16])的值和写入波特率发生器寄存器CLKDIV (UUART_BRGEN[25:16])。

数据传输中断处理

数据传输中断指示相关UART帧处理事件。

发送开始中断

在数据起始位后TXSTIF (UUART_PROTSTS [1])位被置位。在缓冲区模式下，这是可以将新数据字写入UUART_TXDAT的最早时间点。

发送结束

该中断表明发送器已经完全的完成所有在缓冲中的数据传输。TXENDIF (UUART_PROTSTS [2])在最后停止位结束后被置位

接受开始中断

在采样到起始位的时刻RXSTIF (UUART_PROTSTS [3])被置位。

接受帧完成

该中断指示接收器已完全的完成一帧接收。在接收到最后位结束之后RXENDIF (UUART_PROTSTS [4])被置位。

6.25.5.12 编程范例

以下步骤是被配置为UART协议设置和数据的传输。

1. 设置FUNMODE (UUART_CTL[2:0])为0x2来选择UART接口。
2. 写入波特率寄存器UUART_BRGEN来选择期望的波特率。
   - 设置SPCLKSEL (UUART_BRGEN[3:2])，PTCLKSEL (UUART_BRGEN[1])和RCLKSEL (UUART_BRGEN[0])来选择时钟源。
   - 配置CLKDIV (UUART_BRGEN[25:16])，DSCNT (UUART_BRGEN[14:10])和PDSCNT (UUART_BRGEN[9:8])来确定波特率分辨率。

3. 写线控制寄存器UUART_LINECTL和协议控制寄存器UUART_PROTCTL来配置传输数据格式和UART协议设置。
   - 在DWIDTH (UUART_LINECTL[11:8])中编程数据字段长度。
   - 通过设置EVENPARITY (UUART_PROTCTL[2])和PARITYEN (UUART_PROTCTL[1])来使能校验位和确定校验类型。
   - 通过设置STOPB (UUART_PROTCTL[0])配置停止位长度。
   - 通过使能LSB (UUART_LINECTL[0])来选择UART协议LSB优先传输。
   - 设置EDGEDET (UUART_DATIN0[4:3])为“10”来选择下降沿作为接受起始位侦测。
4. 设置PROTEN (UUART_PROTCTL[31])为1来使能UART协议。
5. 发送和接收数据。
   - 写发送数据寄存器UUART_TXDAT来发送数据。
   - 等待直到TXSTIF(UUART_PROTSTS[1])被置位，用户才可以写入下一个数据到UUART_TXDAT。
   - 当 TXENDIF(UUART_PROTSTS[2])被置位，发送缓冲为空并且最后数据的停止位已经被发送。
   - 如果RXENDIF(UUART_PROTSTS[4])被置位，一个数据帧才完整接收完成。用户可以通过读取数据寄存器UUART_RXDAT获取数据。

6.25.6 寄存器映射

R: 只读，**W:**只写，**R/W:**读/写

寄存器	偏移地址	R/W	描述	复位值
UUART_UART 基地址： **UUARTn_BA = 0x400D_0000 + (0x1000 * n)** **n= 0, 1**				
UUART_CTL	UUARTn_BA+0x00	R/W	USCI 控制寄存器	0x0000_0000
UUART_INTEN	UUARTn_BA+0x04	R/W	USCI 中断使能寄存器	0x0000_0000
UUART_BRGEN	UUARTn_BA+0x08	R/W	USCI 波特率发生寄存器	0x0000_3C00
UUART_DATIN0	UUARTn_BA+0x10	R/W	USCI 输入数据信号配置寄存器0	0x0000_0000
UUART_CTLIN0	UUARTn_BA+0x20	R/W	USCI 输入控制信号配置寄存器0	0x0000_0000
UUART_CLKIN	UUARTn_BA+0x28	R/W	USCI 输入时钟信号配置寄存器	0x0000_0000
UUART_LINECTL	UUARTn_BA+0x2C	R/W	USCI 线控制寄存器	0x0000_0000
UUART_TXDAT	UUARTn_BA+0x30	W	USCI 发送数据寄存器	0x0000_0000
UUART_RXDAT	UUARTn_BA+0x34	R	USCI 接收数据寄存器	0x0000_0000
UUART_BUFCTL	UUARTn_BA+0x38	R/W	USCI 发送/接收缓冲控制寄存器	0x0000_0000
UUART_BUFSTS	UUARTn_BA+0x3C	R	USCI 发送/接收缓冲状态寄存器	0x0000_0101
UUART_PDMACTL	UUARTn_BA+0x40	R/W	USCI PDMA控制寄存器	0x0000_0000
UUART_WKCTL	UUARTn_BA+0x54	R/W	USCI 唤醒控制寄存器	0x0000_0000
UUART_WKSTS	UUARTn_BA+0x58	R/W	USCI 唤醒状态寄存器	0x0000_0000
UUART_PROTCTL	UUARTn_BA+0x5C	R/W	USCI 协议控制寄存器	0x0000_0000
UUART_PROTIEN	UUARTn_BA+0x60	R/W	USCI 协议中断使能寄存器	0x0000_0000
UUART_PROTSTS	UUARTn_BA+0x64	R/W	USCI 协议状态寄存器	0x0000_0000

6.25.7 寄存器描述

USCI 控制寄存器 (UUART_CTL)

寄存器	偏移地址	R/W	描述	复位值
UUART_CTL	UUARTn_BA+0x00	R/W	USCI 控制寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved					FUNMODE		

位	描述	
[31:3]	Reserved	保留
[2:0]	FUNMODE	**功能模式** 该域选择USCI控制器的协议。选择不可用的协议或保留的组合将禁用USCI。在两个协议之间切换时，必须在选择新协议之前禁用USCI。同时，当用户将000写入FUNMODE时，USCI将被重置。 000 = 禁用USCI。所有协议相关的状态机被设置为空闲状态。 001 = 选择SPI协议 010 = 选择UART 100 = 选择I^2C协议 **注:**其他组合值保留

USCI 中断使能寄存器 (UUART_INTEN)

寄存器	偏移地址	R/W	描述	复位值
UUART_INTEN	UUARTn_BA+0x04	R/W	USCI 中断使能寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved			RXENDIEN	RXSTIEN	TXENDIEN	TXSTIEN	Reserved

位	描述	
[31:5]	Reserved	保留
[4]	RXENDIEN	**接收结束中断使能位** 该位是使能发生接收结束事件时候产生中断 0 = 禁用接收结束中断 1 = 使能接收结束中断
[3]	RXSTIEN	**接收起始中断使能位** 该位是使能发生接收起始事件时候产生中断 0 = 禁用接收起始中断 1 = 使能接收起始中断
[2]	TXENDIEN	**发送结束中断使能位** 该位是使能发生发送结束事件时候产生中断 0 = 禁用发送结束中断 1 = 使能发送结束中断
[1]	TXSTIEN	**发送起始中断使能位** 该位是使能发生发送起始事件时候产生中断 0 = 禁用发送起始中断 1 = 使能发送起始中断
[0]	Reserved	保留

USCI 波特率发生寄存器 (UUART_BRGEN)

寄存器	偏移地址	R/W	描述	复位值
UUART_BRGEN	UUARTn_BA+0x08	R/W	USCI 波特率发生寄存器	0x0000_3C00

31	30	29	28	27	26	25	24
Reserved						CLKDIV	
23	22	21	20	19	18	17	16
CLKDIV							
15	14	13	12	11	10	9	8
Reserved	DSCNT					PDSCNT	
7	6	5	4	3	2	1	0
Reserved		TMCNTSRC	TMCNTEN	SPCLKSEL		PTCLKSEL	RCLKSEL

位	描述	
[31:26]	**Reserved**	保留
[25:16]	**CLKDIV**	**时钟分频器** 该域是定义协议时钟频率f_{PROT_CLK}和时钟分频器频率f_{DIV_CLK} ($f_{DIV_CLK} = f_{PROT_CLK} / (CLKDIV+1)$).之间的比率 **注:**在UART功能下，当自动波特率功能(ABREN(UUART _PROTCTL[6])) 使能时，会在输入数据0x55的第四个下降沿时被硬件更新。修改值是在位5和位6之间的时间平均值。用户可以使用修改的CLKDIV和新的BRDETITV (UUART _PROTCTL[24:16])来计算精确的波特率。
[15]	**Reserved**	保留
[14:10]	**DSCNT**	**采样计数器的分母** 该域定义采样时钟f_{SAMP_CLK} 的分频比 分频$f_{DS_CNT} = f_{PDS_CNT} / (DSCNT+1)$ **注:**在UART模式下，DSCNT最大值是0xF，建议设置4以上来保证接收数据的采样时正确的值。
[9:8]	**PDSCNT**	**采样计数器预分频器** 该域定义的是对来自采样时钟f_{SAMP_CLK}的分频比 分频$f_{PDS_CNT} = f_{SAMP_CLK} / (PDSCNT+1)$
[7:6]	**Reserved**	保留
[5]	**TMCNTSRC**	**时序测量计数器时钟源选择** 0 = 来自f_{PROT_CLK} 1 = 来自f_{DIV_CLK}
[4]	**TMCNTEN**	**时序测量计数器使能位** 该位是使能10位时序测量计数器 0 = 禁用时序测量计数器

		1 = 使能时序测量计数器
[3:2]	**SPCLKSEL**	**采样时钟源选择** 该域用于对协议采样时钟(f_{SAMP_CLK})的时钟源选择 00 = f_{SAMP_CLK} = f_{DIV_CLK}. 01 = f_{SAMP_CLK} = f_{PROT_CLK} 10 = f_{SAMP_CLK} = f_{SCLK}. 11 = f_{SAMP_CLK} = f_{REF_CLK}.
[1]	**PTCLKSEL**	**协议协议时钟源选择** 该位用于选择协议时钟(f_{PROT_CLK})的源信号 0 = 参考时钟f_{REF_CLK}. 1 = f_{REF_CLK2} (频率是f_{REF_CLK}一半)
[0]	**RCLKSEL**	**参考时钟源选择** 该位是选择参考时钟(f_{REF_CLK})源信号 0 = 外设设备时钟f_{PCLK}. 1 = 保留

USCI 输入数据信号配置寄存器 (UUART_DATIN0)

寄存器	偏移地址	R/W	描述	复位值
UUART_DATIN0	UUARTn_BA+0x10	R/W	USCI 输入数据信号配置寄存器0	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved			EDGEDET		ININV	Reserved	SYNCSEL

位	描述	
[31:5]	**Reserved**	保留
[4:3]	**EDGEDET**	**采样计数器的分母** 该域定义采样时钟f_{SAMP_CLK}的分频比 分频$f_{DS_CNT} = f_{PDS_CNT} / (DSCNT+1)$ **注:**在UART模式下，DSCNT最大值是0xF，建议设置4以上来保证接收数据的采样时正确的值。
[2]	**ININV**	**采样计数器预分频器** 该域定义的是对来自采样时钟f_{SAMP_CLK}的分频比 分频$f_{PDS_CNT} = f_{SAMP_CLK} / (PDSCNT+1)$
[1]	**Reserved**	保留
[0]	**SYNCSEL**	**时序测量计数器时钟源选择** 0 = 来自f_{PROT_CLK} 1 = 来自$f_{DIV_CLK.}$

USCI 输入控制信号配置寄存器 (UUART_CTLIN0)

寄存器	偏移地址	R/W	描述	复位值
UUART_CTLIN0	UUARTn_BA+0x20	R/W	USCI 输入控制信号配置寄存器0	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved					ININV	Reserved	SYNCSEL

位	描述	
[31:3]	Reserved	保留
[2]	ININV	**输入信号反相选择** 该域是定义使能对输入异步信号输入反相 0 = 对异步输入信号不反相 1 = 对异步输入信号反相
[1]	Reserved	保留
[0]	SYNCSEL	**输入同步信号选择** 该位是选择异步输入（带反相的）信号或是同步（带过滤的）信号被用作数据移位单元的输入 0 =异步信号作为信号被用作数据移位单元的输入 1 = 同步信号作为信号被用作数据移位单元的输入

USCI 输入时钟信号配置寄存器 (UUART_CLKIN)

寄存器	偏移地址	R/W	描述	复位值
UUART_CLKIN	UUARTn_BA+0x28	R/W	USCI输入时钟信号配置寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved							**SYNCSEL**

位	描述	
[31:1]	**Reserved**	保留
[0]	**SYNCSEL**	**输入同步信号选择** 该位是选择异步输入（带反相的）信号或是同步（带过滤的）信号被用作数据移位单元的输入 0 = 异步信号作为信号被用作数据移位单元的输入 1 = 同步信号作为信号被用作数据移位单元的输入

USCI 线控制寄存器 (UUART_LINECTL)

寄存器	偏移地址	R/W	描述	复位值
UUART_LINECTL	UUARTn_BA+0x2C	R/W	USCI 线控制寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved				DWIDTH			
7	6	5	4	3	2	1	0
CTLOINV	Reserved	DATOINV	Reserved				LSB

位	描述	
[31:12]	**Reserved**	保留
[11:8]	**DWIDTH**	**传输的字长** 该域定义发送和接收的数据字长。在数据缓冲中数据总是右对齐。USCI支持4到16位的字长 0x0:在数据字中包含16位，位于[15:0]这些位中 0x1: 保留 0x2: 保留 0x3: 保留 0x4: 在数据字中包含4位，位于[3:0]这些位中 0x5: 在数据字中包含5位，位于[4:0]这些位中 ... 0xF:在数据字中包含15位，位于[14:0]这些位中 **注:**在UART模式下，长度可配置为6~13 位。
[7]	**CTLOINV**	**控制信号输出反相选择** 该位定义了内部控制信号和输出控制信号之间的关系 0 = 无影响 1 = 在输出前控制信号被反相 **注:**在UART模式下，控制信号就是nRTS信号。
[6]	**Reserved**	保留
[5]	**DATOINV**	**数据输出反相选择** 该位定义了内部移位数据值和USCIx DAT1引脚输出数据信号之间的关系 0 = USCIx_DAT1的值等于数据移位寄存器 1 = USCIx_DAT1的值为数据移位寄存器的反相
[4:1]	**Reserved**	保留

[0]	**LSB**	**LSB优先传输选择** 0 = 取决于DWIDTH设置的接收/发送数据缓冲的MSB先发送/接收 1 = 数据缓冲的位0及LSB被先发送/接收

USCI 发送数据寄存器 (UUART_TXDAT)

寄存器	偏移地址	R/W	描述	复位值
UUART_TXDAT	UUARTn_BA+0x30	W	USCI 发送数据寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
TXDAT							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
TXDAT							

位	描述	
[31:16]	**Reserved**	保留
[15:0]	**TXDAT**	**发送数据** 软件可以写16位发送数据到该域用来发送

USCI 接收数据寄存器 (UUART_RXDAT)

寄存器	偏移地址	R/W	描述	复位值
UUART_RXDAT	UUARTn_BA+0x34	R	USCI 接收数据寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
RXDAT							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
RXDAT							

位	描述	
[31:16]	**Reserved**	保留
[15:0]	**RXDAT**	**接收的数据** 该域值监控存储在接收数据缓冲的接收到的数据 **注**: RXDAT[15:13]表明BREAK，FRMERR和PARITYERR (UUART_PROTSTS[7:5])相同帧状态

USCI 发送/接收缓冲控制寄存器(UUART_BUFCTL)

寄存器	偏移地址	R/W	描述	复位值
UUART_BUFCTL	UUARTn_BA+0x38	R/W	USCI 发送/接收缓冲控制寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved						RXRST	TXRST
15	14	13	12	11	10	9	8
RXCLR	RXOVIEN	Reserved					
7	6	5	4	3	2	1	0
TXCLR	Reserved						

位	描述	
[31:18]	**Reserved**	保留
[17]	**RXRST**	**接收复位** 0 = 无影响 1 = 复位接收相关计数器、状态机、接收移位寄存器的内容和数据缓冲。 **注1:** 1个PCLK周期后自动被清除 **注2:** 在该位被置1前，建议检查RXBUSY (UUART_PROTSTS[10])
[16]	**TXRST**	**发送复位** 0 = 无影响 1 = 复位发送相关计数器、状态机、发送移位寄存器的内容和数据缓冲。 **注:** 1个PCLK周期后自动被清除
[15]	**RXCLR**	**清接收缓冲** 0 = 无影响 1 = 接收缓冲被清除（填充被清除并且输出指针被设置为输入指针的值）仅当缓冲没有参与数据通信时被使用。 **注:** 1个PCLK周期后自动被清除
[14]	**RXOVIEN**	**接收缓冲溢出错误中断使能控制** 0 = 禁用接收溢出中断 1 = 使能接收溢出中断
[13:8]	**Reserved**	保留
[7]	**TXCLR**	**清发送缓冲** 0 = 无影响 1 = 发送缓冲被清除（填充被清除并且输出指针被设置为输入指针的值）仅当缓冲没有参与数据通信时被使用。

		注: 1个PCLK周期后自动被清除
[6:0]	**Reserved**	保留

USCI 发送/接收缓冲状态寄存器 (UUART_BUFSTS)

寄存器	偏移地址	R/W	描述	复位值
UUART_BUFSTS	UUARTn_BA+0x3C	R	USCI 发送/接收缓冲状态寄存器	0x0000_0101

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved						TXFULL	TXEMPTY
7	6	5	4	3	2	1	0
Reserved				RXOVIF	Reserved	RXFULL	RXEMPTY

位	描述	
[31:10]	**Reserved**	保留
[9]	**TXFULL**	**发送缓冲满指示** 0 = 发送缓冲还没满 1 = 发送缓冲满
[8]	**TXEMPTY**	**发送缓冲空指示** 0 = 发送缓冲非空 1 = 发送缓冲为空
[7:4]	**Reserved**	保留
[3]	**RXOVIF**	**接收缓冲溢出错误中断状态** 该位表示接收缓冲溢出错误事件被侦测到。如果RXOVIEN (UUART_BUFCTL[14])使能，会激活相应的中断请求。软件写**1**清除该位。 0 = 接收缓冲溢出错误事件没有被侦测到 1 = 接收缓冲溢出错误事件被侦测到
[2]	**Reserved**	保留
[1]	**RXFULL**	**接收缓冲满指示** 0 = 接收缓冲还没满 1 = 接收缓冲满
[0]	**RXEMPTY**	**接收缓冲空指示** 0 = 接收缓冲非空 1 = 接收缓冲为空

USCI PDMA控制寄存器 (UUART_PDMACTL)

寄存器	偏移地址	R/W	描述	复位值
UUART_PDMACTL	UUARTn_BA+0x40	R/W	USCI PDMA控制寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved				PDMAEN	RXPDMAEN	TXPDMAEN	PDMARST

位	描述	
[31:4]	**Reserved**	保留
[3]	**PDMAEN**	**PDMA模式使能位** 0 = 禁用PDMA功能 1 = 使能PDMA功能
[2]	**RXPDMAEN**	**PDMA接收通道有效** 0 = 禁用PDMA接收功能 1 = 使能PDMA接收功能
[1]	**TXPDMAEN**	**PDMA发送通道有效** 0 = 禁用PDMA发送功能 1 = 使能PDMA发送功能
[0]	**PDMARST**	**PDMA 复位** 0 = 无影响 1 = 复位USCI的PDMA控制逻辑。该位会自动地被清0

USCI 唤醒控制寄存器 (UUART_WKCTL)

寄存器	偏移地址	R/W	描述	复位值
UUART_WKCTL	UUARTn_BA+0x54	R/W	USCI 唤醒控制寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved					PDBOPT	Reserved	WKEN

位	描述	
[31:3]	Reserved	保留
[2]	PDBOPT	**掉电阻塞选择** 0 = 如果用户试图在协议正在传输的时候执行WFI来进入掉电模式，MCU会停止传输并且立即进入掉电模式 1 = 如果用户试图在协议正在传输的时候执行WFI来进入掉电模式，传输会继续进行并且MCU立即进入掉电模式
[1]	Reserved	保留
[0]	WKEN	**唤醒使能位** 0 = 禁用唤醒功能 1 = 使能唤醒功能

USCI 唤醒状态寄存器 (UUART_WKSTS)

寄存器	偏移地址	R/W	描述	复位值
UUART_WKSTS	UUARTn_BA+0x58	R/W	USCI 唤醒状态寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved							WKF

位	描述	
[31:1]	Reserved	保留
[0]	WKF	**唤醒标志** 当芯片被从掉电模式唤醒，该位置1.软件写1清除该位

USCI 协议控制寄存器 – UART (UUART_PROTCTL)

寄存器	偏移地址	R/W	描述	复位值
UUART_PROTCTL	UUARTn_BA+0x5C	R/W	USCI 协议控制寄存器	0x0000_0000

31	30	29	28	27	26	25	24
PROTEN	Reserved	BCEN	Reserved	Reserved	STICKEN	Reserved	BRDETITV
23	22	21	20	19	18	17	16
BRDETITV							
15	14	13	12	11	10	9	8
Reserved	WAKECNT				CTSWKEN	DATWKEN	Reserved
7	6	5	4	3	2	1	0
Reserved	ABREN	RTSAUDIREN	CTSAUTOEN	RTSAUTOEN	EVENPARITY	PARITYEN	STOPB

位	描述	
[31]	**PROTEN**	**UART协议使能位** 0 = 禁用UART 协议 1 = 使能UART 协议
[30]	**Reserved**	保留
[29]	**BCEN**	**发送Break控制使能位** 0 = 禁用发送Break控制 1 = 使能发送Break控制 **注:**当该位被置1，串行数据输出（TX）被强制为逻辑0状态。该位仅在TX线上有效，对发送逻辑单元没有影响
[27]	**Reserved**	保留
[26]	**STICKEN**	**Stick校验使能位** 0 = 禁用Stick校验 1 = 使能Stick校验 **注:**详细信息请参考RS-485章节
[25]	**Reserved**	保留
[24:16]	**BRDETITV**	**波特率侦测间隔** 该位指明从机在一个位时间内计算波特率需要多少时钟周期（该时钟由TMCNTSRC (UUART_BRGEN [5])决定）。总线的顺序应该是1和0这样交替（比如，输入数据格式应该是0x55）。用户可以读该值去了解当前总线输入波特率，不管ABRDETIF (UUART_PROTSTSL[9])何时置位。 **注:**软件写0到**BRDETITV**.清除为0。
[15]	**Reserved**	保留
[14:11]	**WAKECNT**	**唤醒计数器**

		该位指明当设备被从掉电模式下唤醒时从机获得第一个位及起始位需要多少个时钟周期（该时钟是f_{PDS_CNT}）。
[10]	**CTSWKEN**	**nCTS唤醒模式使能位** 0 = 禁用nCTS 唤醒模式 1 = 使能 nCTS唤醒模式
[9]	**DATWKEN**	**数据唤醒模式使能位** 0 = 禁用数据唤醒模式 1 = 使能数据唤醒模式
[6]	**ABREN**	**自动波特率侦测使能位** 0 = 禁用自动波特率侦测 1 = 使能自动波特率侦测 **注:**当自动波特率侦测操作结束时硬件清除该位。如果ARBIEN (UUART_PROTIEN [1])使能，将产生相关中断ABRDETIF (USCI_PROTST[9])
[5]	**RTSAUDIREN**	**nRTS自动方向使能位** 当nRTS自动方向使能，如果TX缓冲为空，UART自动激活nRTS信号 0 = 禁用nRTS自动方向控制 1 = 使能nRTS自动方向控制 **注 1:**该位是用在RS485下nRTS自动方向控制 **注 2:**该位仅在RTSAUTOEN没有置位情况下有作用
[4]	**CTSAUTOEN**	**nCTS自动流控使能位** 当nCTS自动流控使能时，当nCTS输入激活时UART会发送数据到外部设备（如nCTS输入没有激活UART不会发送数据到设备） 0 = 禁用 nCTS 自动流控 1 = 使能nCTS 自动流控
[3]	**RTSAUTOEN**	**nRTS自动流控使能位** 当nRTS自动流控使能时，如果接收缓冲没有满(RXFULL (UUART_BUFSTS[1] =1)，UART会解除nRTS信号。 0 = 禁用 nRTS 自动流控 1 = 使能nRTS 自动流控 **注 :**该位仅在RTSAUDIREN没有置位情况下有作用
[2]	**EVENPARITY**	**偶校验使能位** 0 = 发送和检测每一个字中逻辑1是奇数。 1 = 发送和检测每一个字中逻辑1是偶数。 **注:**仅当PARITYEN置位时该位有作用
[1]	**PARITYEN**	**校验使能位** 在UART帧中，该位定义校验位使能 0 = 禁用校验位 1 = 使能校验位
[0]	**STOPB**	**停止位** 在UART帧中，该位定义停止位数目 0 = 1位停止位 1 = 2位停止位

USCI 协议中断使能寄存器 – UART (UUART_PROTIEN)

寄存器	偏移地址	R/W	描述	复位值
UUART_PROTIEN	UUARTn_BA+0x60	R/W	USCI 协议中断使能寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved					RLSIEN	ABRIEN	Reserved

位	描述	
[31:3]	**Reserved**	保留
[2]	**RLSIEN**	**接收线状态中断使能** 0 = 禁用接收线状态中断 1 = 使能接收线状态中断 **注:** UART_PROTSTS[7:5]反映当前接收线状态中断事件
[1]	**ABRIEN**	**自动波特率中断使能位** 0 = 禁用自动波特率中断 1 = 使能自动波特率中断
[0]	**Reserved**	保留

USCI 协议状态寄存器 – UART (UUART_PROTSTS)

寄存器	偏移地址	R/W	描述	复位值
UUART_PROTSTS	UUARTn_BA+0x64	R/W	USCI 协议状态寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved						CTSLV	CTSSYNCLV
15	14	13	12	11	10	9	8
Reserved				ABERRSTS	RXBUSY	ABRDETIF	Reserved
7	6	5	4	3	2	1	0
BREAK	FRMERR	PARITYERR	RXENDIF	RXSTIF	TXENDIF	TXSTIF	Reserved

位	描述	
[31:18]	**Reserved**	保留
[17]	**CTSLV**	**nCTS引脚状态 (只读)** 该位被用作监控nCTS引脚输入的当前状态 0 = nCTS引脚为低电平 1 = nCTS引脚为高电平
[16]	**CTSSYNCLV**	**nCTS同步电平状态 (只读)** 该位被用作反映内部同步nCTS信号当前状态 0 = 内部同步nCTS为低 1 = 内部同步nCTS为高
[15:12]	**Reserved**	保留
[11]	**ABERRSTS**	**自动波特率错误状态** 当自动波特率侦测计数器溢出时该位被置位。当自动波特率计数器溢出时，用户需要修改CLKDIV (UUART_BRGEN[25:16])值并且使能ABREN (UUART_PROTCTL[6])来再次侦测正确的波特率 0 = 自动波特率计数器没有溢出 1 = 自动波特率计数器溢出 **注 1:**.该位在ABRDETIF相同时间被置位。 **注 2:** 写1到ABRDETIF 或 ABERRSTS，清除该位
[10]	**RXBUSY**	**RX总线状态标志 (只读)** 该位反映接收器忙状态 0 = 接收器处于空闲状态.

		1 = 接收器处于忙状态.
[9]	**ABRDETIF**	**自动波特率中断标志** 当自动波特率侦测在输入数据下降沿中间完成时，该位被置位。如果ABRIEN (UUART_PROTCTL[6])置位，会产生自动波特率中断。当输入数据为0x55时，该位会被设置4次并且在输入总线下次下降沿之前被清除。 0 = 自动波特率功能没有完成 1 = 一位自动波特率功能已完成 **注:**该位可以写1清除
[8]	**Reserved**	保留
[7]	**BREAK**	**Break标志** 每当接收数据输入（RX）保持在逻辑0状态，该状态持续时间超过了一整个字的传输（也就是：起始位+数据位+停止位的时间），该位被置1. 0 = 没有Break产生 1 = 接收总线上出现Break状态 **注:**写1到BREAK, FRMERR 和 PARITYERR中该位被清除
[6]	**FRMERR**	**帧错误标志** 每当接收字符没有有效的停止位时，该位被置1.（也就是，仅更随在最后数据位或是校验位之后的停止位被侦测到是逻辑0） 0 = 没有帧错误产生 1 = 有帧错误发生 **注:**写1到BREAK, FRMERR 和 PARITYERR中该位被清除
[5]	**PARITYERR**	**校验错误标志** 每当接收字符没有有效的校验位时，该位被置1. 0 = 没有校验错误 1 = 有校验错误发生 **注:**写1到BREAK, FRMERR 和 PARITYERR中该位被清除
[4]	**RXENDIF**	**接收结束中断标志** 0 = 没有接收结束中断状态发生 1 = 有接收结束中断状态发生 **注:**写1清除该位
[3]	**RXSTIF**	**接收起始中断标志** 0 = 没有接收起始中断状态发生 1 = 有接收起始中断状态发生 **注:**写1清除该位
[2]	**TXENDIF**	**发送结束中断标志** 0 = 没有发送结束中断状态发生 1 = 有发送结束中断状态发生 **注:**写1清除该位
[1]	**TXSTIF**	**发送起始中断标志** 0 = 没有发送起始中断状态发生 1 = 有发送起始中断状态发生 **注1:**写1清除该位

		注 2:当没有数据在发送缓冲时用户可以用此标志位来加载下次发送数据
[0]	**Reserved**	保留

6.26 USCI – SPI模式

6.26.1 概述

USCI控制器的SPI协议适用于同步串行数据通信并且支持全双工传输。支持主机和从机操作4线接口模式。USCI控制器的SPI模式在接收外设数据时执行的时串行到并行的转换，在发送数据到外设时执行的是并行到串行的转换。设置FUNMODE (USPI_CTL[2:0]) = 0x1时选择SPI模式。

通过设置SLAVE (USPI_PROTCTL[0])来选择SPI协议运行在主机或是从机模式。下图展示了主机和从机模式的应用。

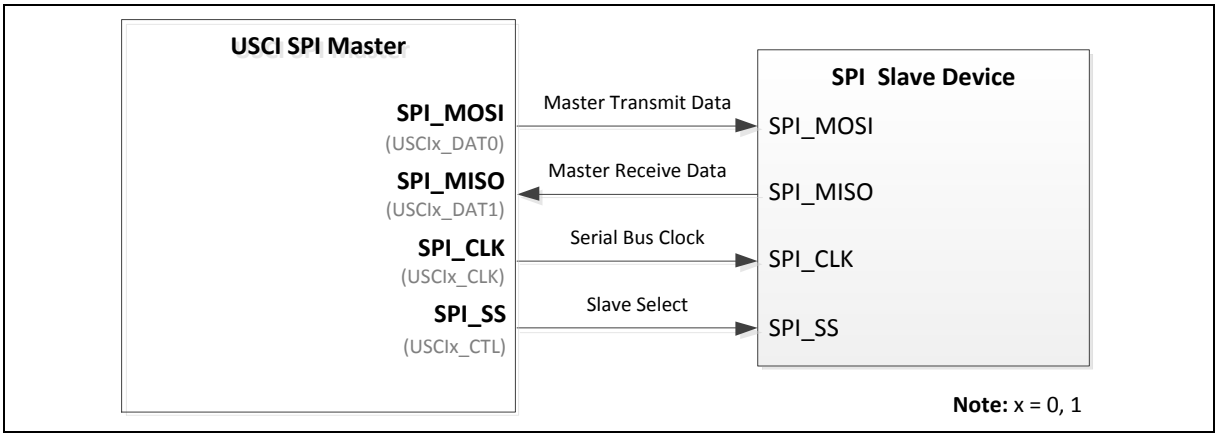


图 6.26-1 SPI 主机模式应用框图

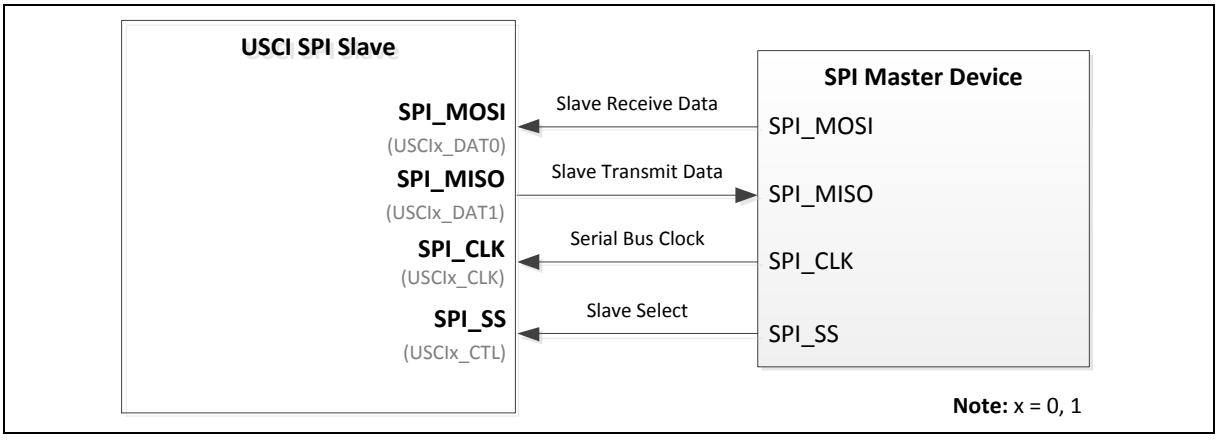


图 6.26-2 SPI 从机模式应用框图

6.26.2 特性

- 支持主机或从机模式 (最大频率 – 主机 = f_{PCLK} / 2, Slave <f_{PCLK} / 5)
- 可编程传输字的长度在4到16位
- 支持一组发送缓冲区和两组接收缓冲区
- 支持MSB优先或LSB优先传输
- 支持字传输暂停功能

- 支持PDMA传输
- 支持3线，无从机选择信号，双向接口
- 支持从机模式下从机选择信号的唤醒功能
- 支持一个数据通道半双工传输

6.26.3 框图

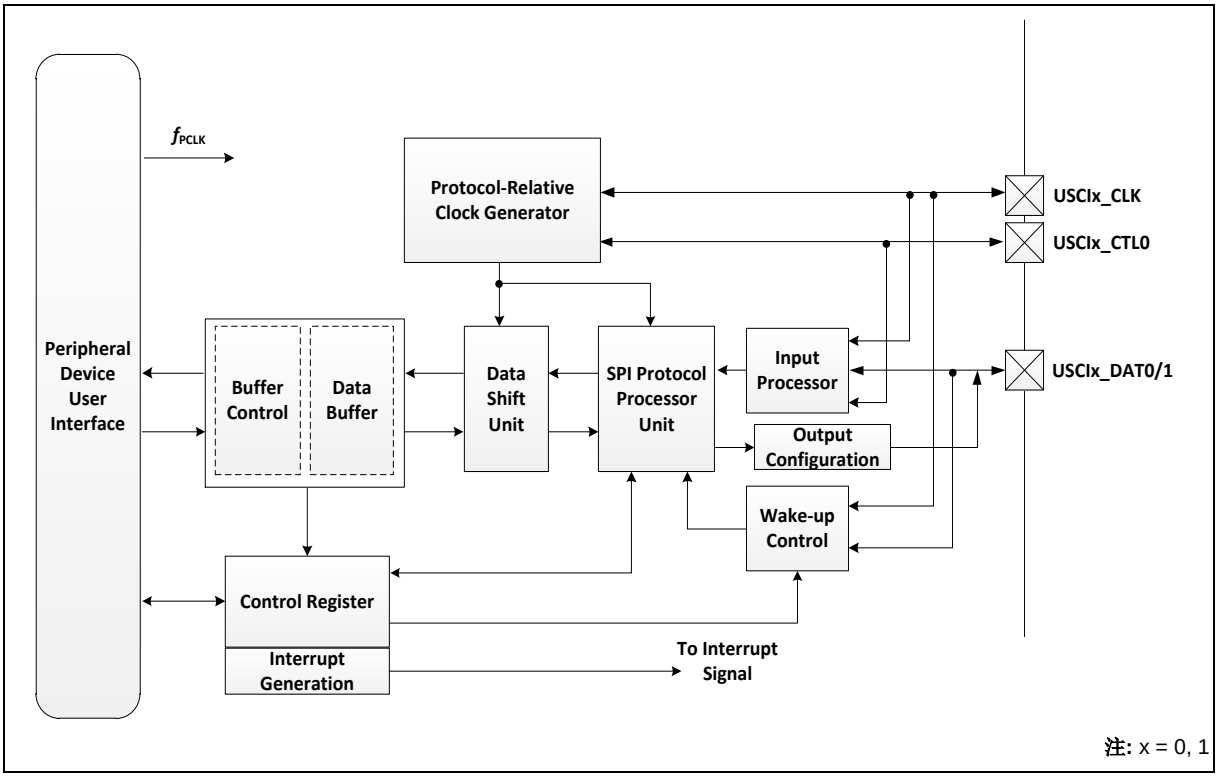


图 6.26-3USCI SPI 模式框图

6.26.4 基本配置

6.26.4.1 USCI0 SPI基本配置

- 时钟源配置
  - 使能USCI0外设时钟位USCI0CKEN (CLK_APBCLK1[8]).
  - 使能USCI0_SPI功能寄存器USPI_CTL[2:0]，USPI_CTL[2:0]=3’b001
- 复位配置
  - 复位USCI0控制器位USCI0RST (SYS_IPRST2[8]).
- 引脚配置

组	引脚名称	GPIO	MFP
USCI0	USCI0_CLK	PD.0	MFP3
		PB.12	MFP5

		PA.11	MFP6
		PE.2	MFP7
	USCI0_CTL0	PD.4	MFP3
		PC.14	MFP5
		PC.13	MFP6
		PE.6	MFP7
	USCI0_DAT0	PD.1	MFP3
		PB.13	MFP5
		PA.10	MFP6
		PE.3	MFP7
	USCI0_DAT1	PD.2	MFP3
		PB.14	MFP5
		PA.9	MFP6
		PE.4	MFP7

6.26.4.2 USCI1 SPI基本配置

- 时钟源配置
  - 使能USCI1外设时钟位USCI1CKEN (CLK_APBCLK1[9])。
  - 使能USCI1_SPI功能寄存器USPI_CTL[2:0])，USPI_CTL[2:0]=3’b001。
- 复位配置
  - 复位USCI1控制器位USCI1RST (SYS_IPRST2[9])。
- 引脚配置

组	引脚名称	GPIO	MFP
USCI1	USCI1_CLK	PB.8	MFP4
		PD.7, PE.12	MFP6
		PB.1	MFP8
	USCI1_CTL0	PB.10	MFP4
		PD.3, PE.9	MFP6
		PB.5	MFP8
	USCI1_DAT0	PB.7	MFP4
		PD.5, PE.10	MFP6
		PB.2	MFP8
	USCI1_DAT1	PB.6	MFP4
		PD.6, PE.11	MFP6

		PB.3	MFP8

6.26.5 功能描述

6.26.5.1 USCI共同功能描述

请参考章节6.24.4详细描述。

6.26.5.2 信号描述

在主机模式下的设备控制数据传输的开始和结束，并且SPI总线时钟和片选信号也是有主机设备产生。片选信号反映了一次数据的起始和结束，主机可以使用它使能对从机的发送或是接收操作。SPI通信信号如下。

SPI模式	接收数据	发送数据	串行总线时钟	从机选择
全双工SPI主机	SPI_MISO (USCIx_DAT1)	SPI_MOSI (USCIx_DAT0)	SPI_CLK (USCIx_CLK)	SPI_SS (USCIx_CTL0)
全双从SPI从机	SPI_MOSI (USCIx_DAT0)	SPI_MISO (USCIx_DAT1)	SPI_CLK (USCIx_CLK)	SPI_SS (USCIx_CTL0)
半双工SPI主机/从机	SPI_MOSI (USCIx_DAT0)	SPI_MOSI (USCIx_DAT0)	SPI_CLK (USCIx_CLK)	SPI_SS (USCIx_CTL0)

SPI 通信信号

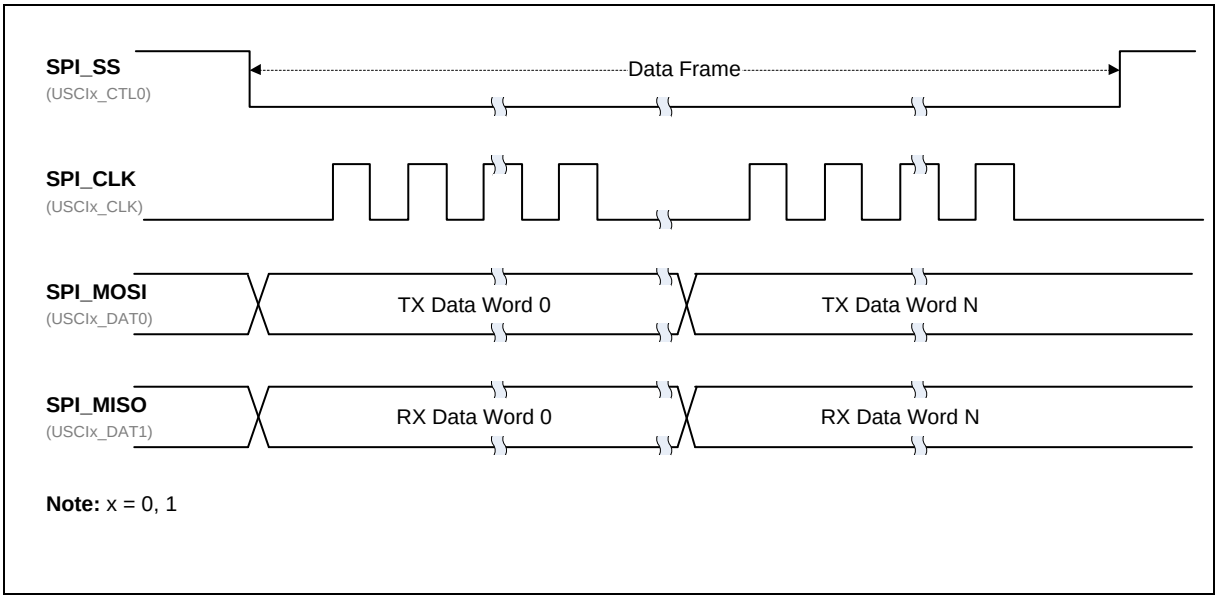


图 6.26-44-写全双工 SPI 通信信号（主机模式）

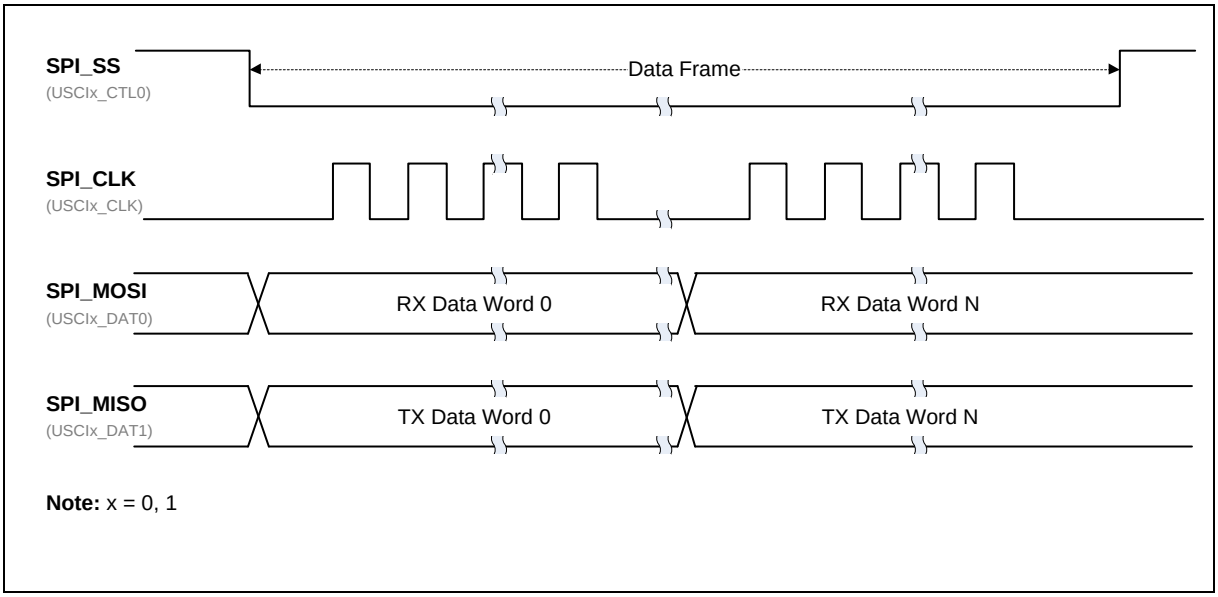


图 6.26-54-写全双工 SPI 通信信号（从机模式）

6.26.5.3 串行总线时钟配置

USCI控制器需要外设时钟来驱动USCI的逻辑单元执行数据传输。外设时钟频率等于PCLK频率。

在主机模式下，SPI总线时钟频率是由协议相关时钟发生器决定的。通常，SPI总线时钟被表示为SPI时钟。SPI时钟频率是f_{SAMP_CLK}的一半，f_{SAMP_CLK}是由SPCLKSEL (USPI_BRGEN[3:2])决定的。相关协议时钟发生器的详细内容请参考6.24.4。

在从机模式下，SPI总线时钟是有片外主机设备提供的。SPI从机外设时钟频率$_{PCLK}$必须快于SPI主机设备连接的串行总线时钟速率的5倍（也就是串行总线时钟速率< 1/5外设时钟f_{PCLK}）。

在SPI协议下，SCLKMODE (USPI_PROTCTL[7:6]) 既定义了串行总线时钟的空闲状态也被用作定义发送与接收数据的串行时钟边沿。SCLKMODE配置对主机和从机是一样的。串行总线时钟的四种模式配置如下。

SCLKMODE [1:0]	SPI时钟空闲状态	发送时序	接收时序
0x0	低	下降沿	上升沿
0x1	低	上升沿	下降沿
0x2	高	上升沿	下降沿
0x3	高	下降沿	上升沿

表 6.26-1 串行总线时钟配置

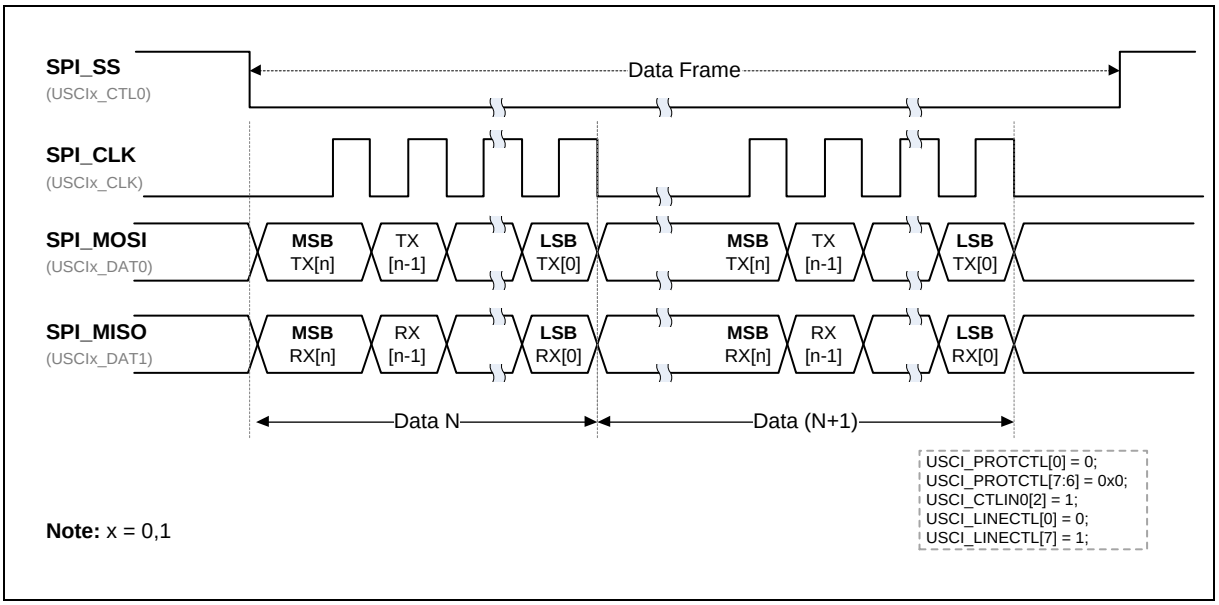


图 6.26-6 不同时钟模式下的 SPI 通信(SCLKMODE=0x0)

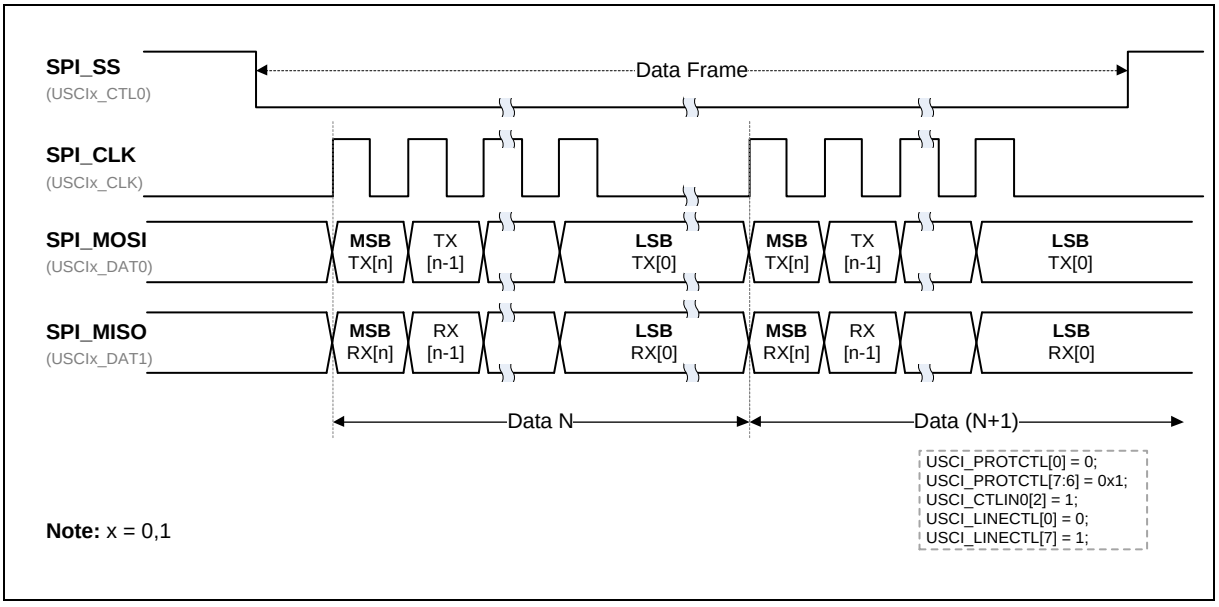


图 6.26-7 不同时钟模式下的 SPI 通信(SCLKMODE=0x1)

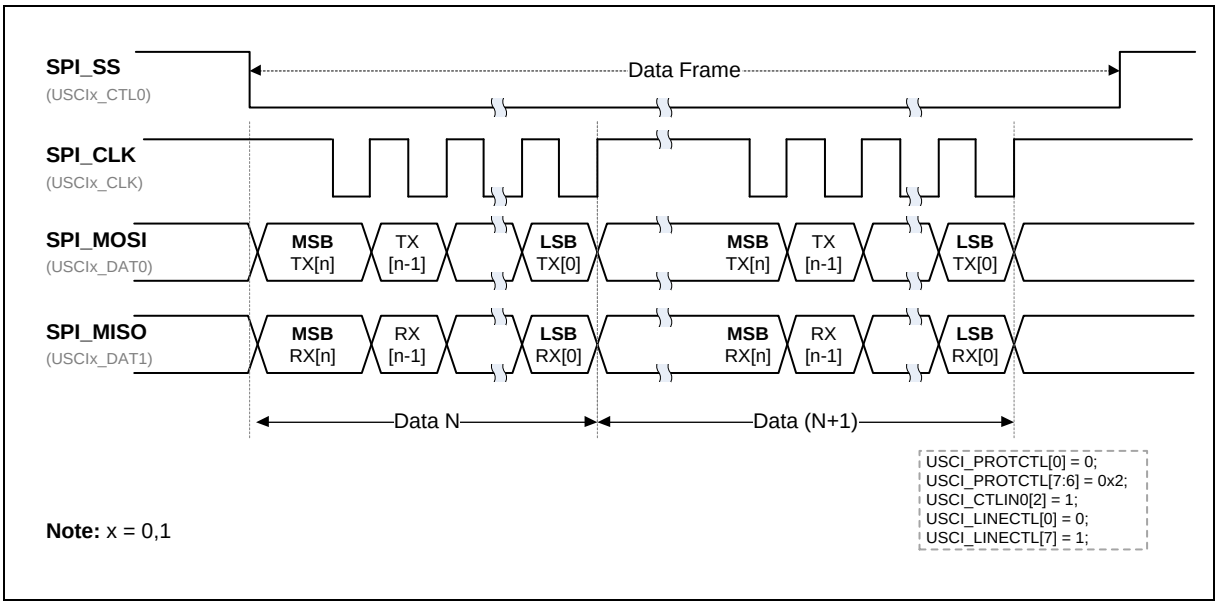


图 6.26-8 不同时钟模式下的 SPI 通信(SCLKMODE=0x2)

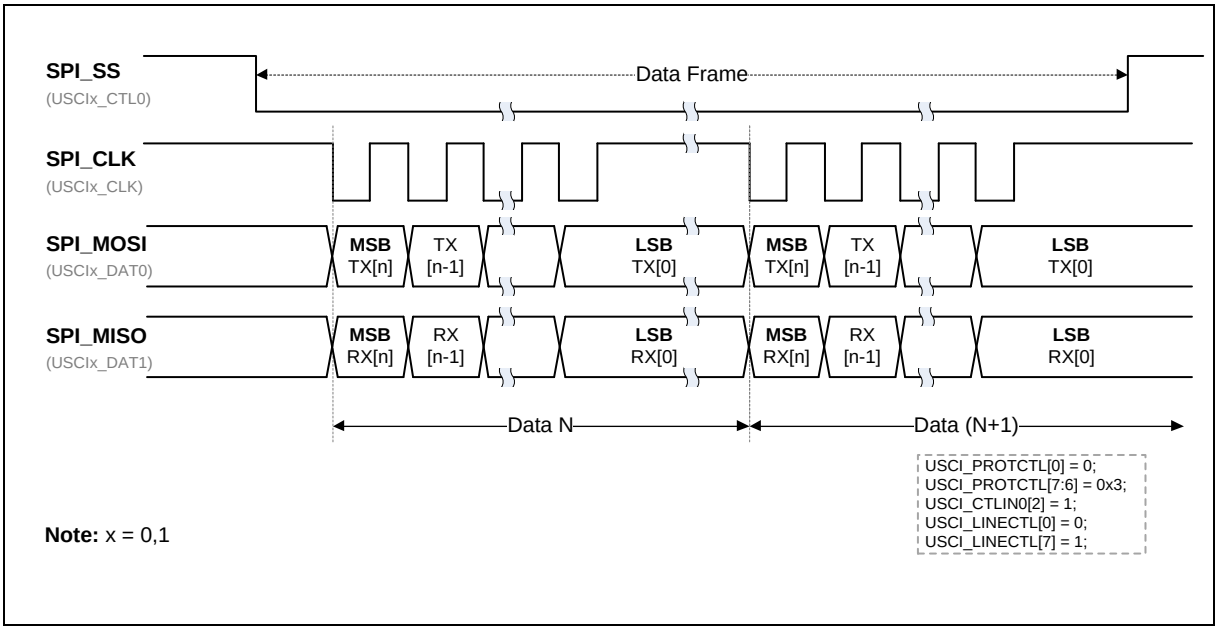


图 6.26-9 不同时钟模式下的 SPI 通信(SCLKMODE=0x3)

6.26.5.4 从机选择信号

SPI协议的从机片选信号默认是高电平激活状态。在SPI主机模式下，USCI控制器可以通过从机选择引脚SPI_SS (USPIx_CTL0)来驱动控制外部SPI从机设备。在SPI从机模式下，接收片选信号可以通过ININV (USPI_CTLIN0[2])反相。

如果外部SPI主机设备的片选信号是低电平激活，那么从机设备的ININV (USPI_CTLIN0[2])的设置应该设置为1来反相输入控制信号。如果USCI工作在SPI主机模式下，那些需要片选激活信号是低电平的外部SPI从机设备来说，主机输出从机选择反相CTLOINV (USPI_LINECTL[7])也要置1。

从机选择激活边沿与第一个SPI时钟输入边沿之间会超过2个USCI外设时钟周期。

为了正确地检测到帧结束，在连续两帧之间SPI从机输入片选信号必须保持非激活状态时间至少在2个USCI外设时钟周期

6.26.5.5 发送和接收数据

在USCI控制器SPI协议下，接收/发送数据位长度可以在DWIDTH (USPI_LINECTL[11:8])中定义，在SPI通信中发送和接收数据长度最大可以配置为16位。

LSB位(USPI_LINECTL[0])定义传输数据位的顺序。如果LSB位被置1，那么传输的数据顺序是LSB优先。如果LSB位被清0，那么传输的数据顺序是MSB优先。

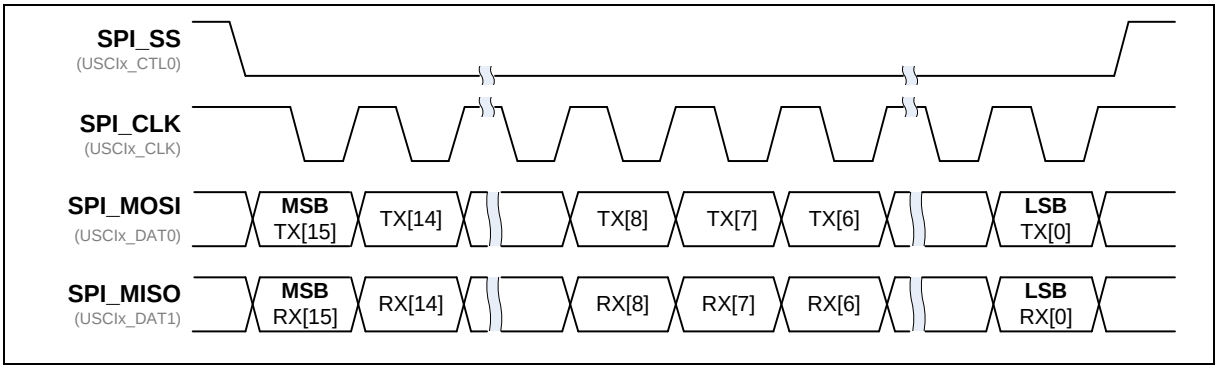


图 6.26-10 MSB 优先 16 位数据长度传输一个字的格式

6.26.5.6 字暂停

主机模式下两个连续字传输之间可以由SUSPITV (USPI_PROTCTL[11:8])提供一个0.5 ~ 15.5 SPI时钟周期可编程的暂停间隔的功能。暂停间隔指的是之前传输的最后一个时钟边沿到下次传输的第一个时钟边沿之间的间隔。SUSPITV (USPI_PROTCTL[11:8])默认值是0x3(3.5 SPI时钟周期)

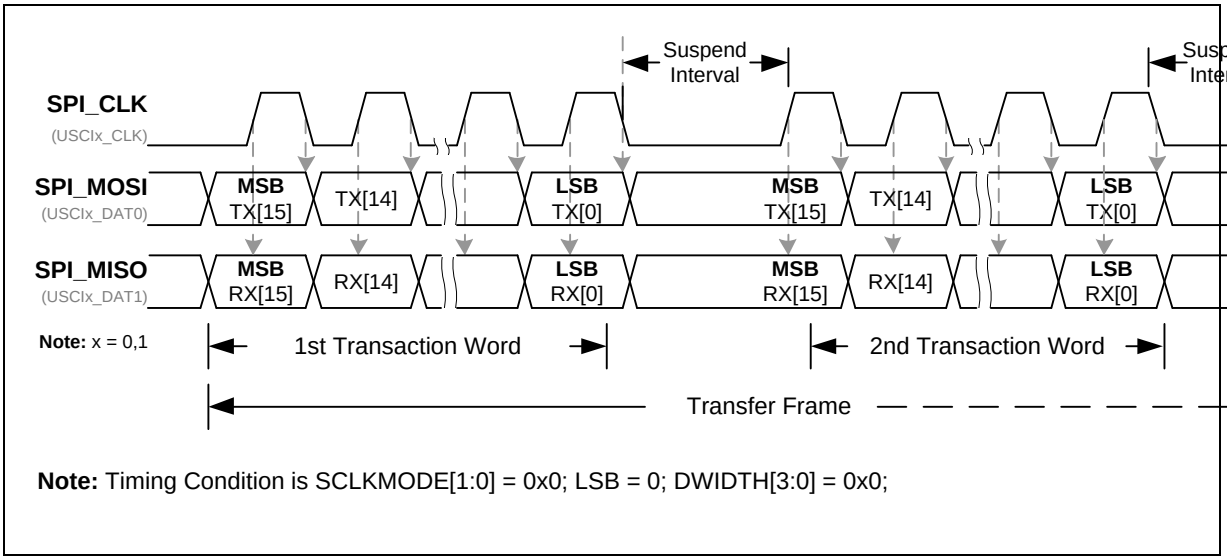


图 6.26-11 两个字传输间的字暂停

6.26.5.7 自动片选功能

AUTOSS (USPI_PROTCTL[3]) 被用作SPI主机模式来使能自动片选功能。如果AUTOSS (USPI_PROTCTL[3])被置位，片选信号会自动地产生并且SS (USPI_PROTCTL[2])设置值不会影响输出从机选择（通过USCIx_CTL0线）。也就意味着当向发送缓冲写入数据启动SPI数据传输的时候，片选信号会变成有效状态。在所有传输结束或是如果SUSPITV (USPI_PROTCTL[11:8])大于等于3时一次字传输完成之后，片选信号会失效。

如果AUTOSS 位 (USPI_PROTCTL[3])被清除，通过设置/清除SS (USPI_PROTCTL[2])来使片选信号USCIx_CTL0引脚有效/无效。内部片选信号激活电平是高，CTLOINV (USPI_LINECTL[7])可以被用作反相片选信号。

在SPI主机模式，如果SUSPITV (USPI_PROTCTL[11:8])值少于3并且AUTOSS (USPI_PROTCTL[3])被置1，在连续两周字传输之间片选信号保持有效状态。

在SPI从机模式，在两组连续传输之间为了识别片选信号的失效状态，接收片选信号非激活的周期必须大于2个外设时钟周期。

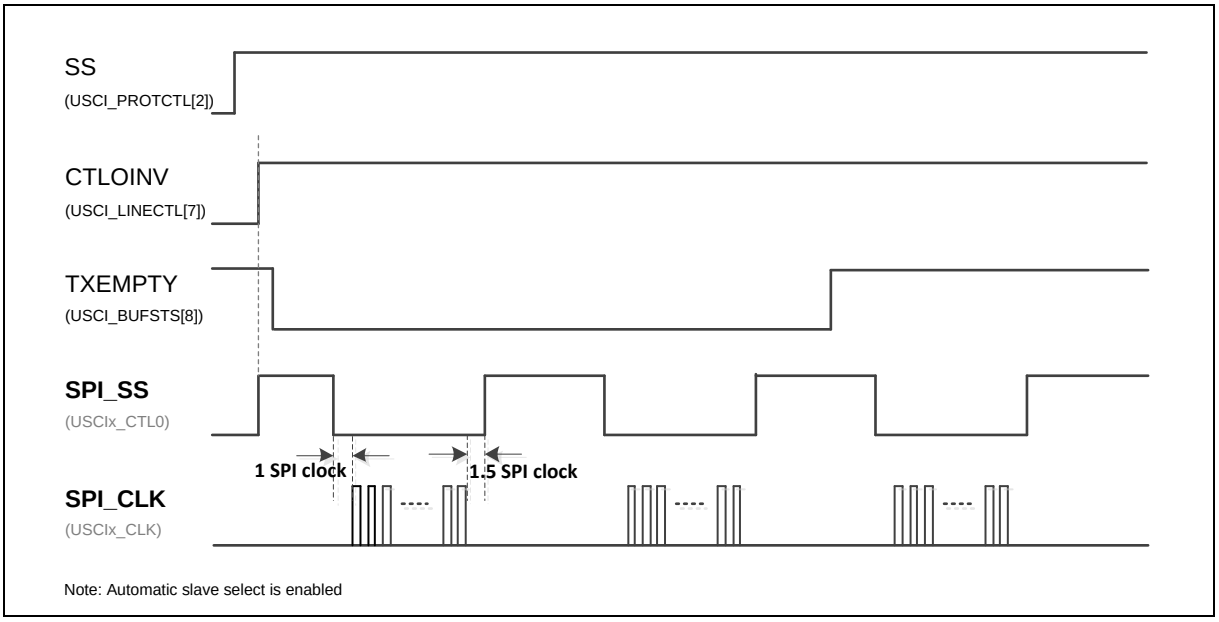


图 6.26-12 自动从机选择(SUSPITV ≧ 0x3)

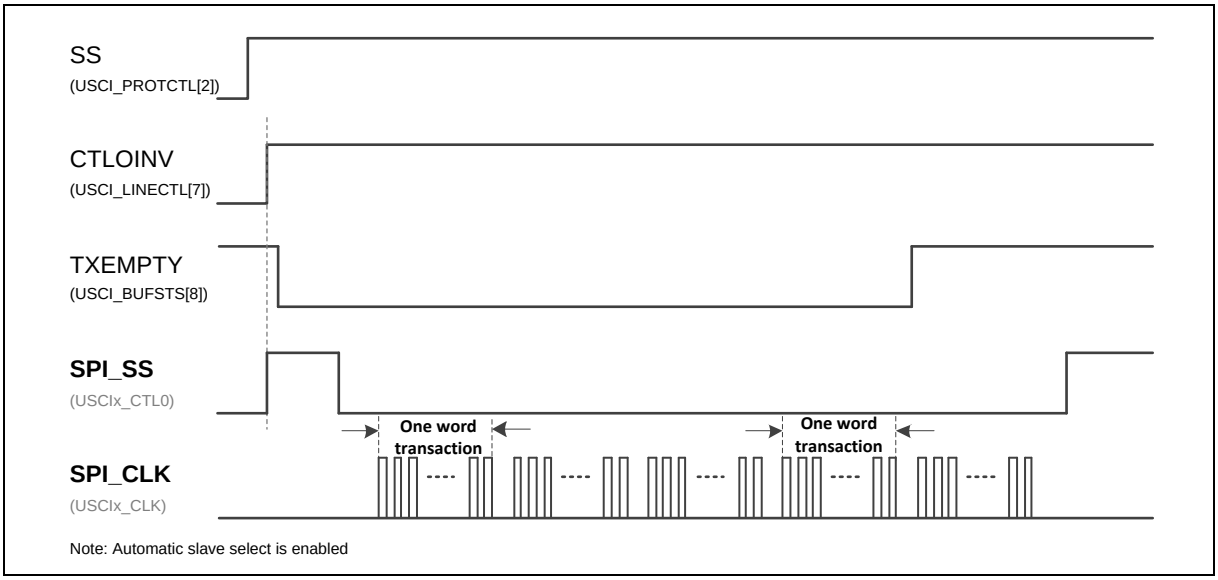


图 6.26-13 自动从机选择(SUSPITV < 0x3)

6.26.5.8 从机3线模式

当通过软件设置SLV3WIRE (USPI_PROTCTL[1])使能从机3线模式的时候，USCI SPI通信在从机模式下可以在没有片选信号下工作。仅三个引脚SPI_CLK (USCIx_CLK)，SPI_MOSI (USPIx_DAT0), 和SPI_MISO (USPIx_DAT1)被用作与SPI主机通信。当SLV3WIRE (USPI_PROTCTL[1])被置1时，在通过设置FUNMODE(USPI_CTL [2:0])为0x1使能SPI协议之后SPI从机就好发送/接收数。

6.26.5.9 数据传输模式

USCI控制器支持全双工SPI传输和一个数据通道半双工SPI传输。

- 全双工SPI传输

  在全双工SPI传输下，有两根数据引脚。一根用来发送数据，另外一根用来接收数据。因此数据发送和接收可以同时进行。

  SCLKMODE (USPI_PROTCTL[7:6])定义数据移位输出信号在USCIx_DAT0引脚上的转变时序。转变可能发生在相应的SPI总线时钟的边沿或是片选信号激活边沿。一个数据字的最后数据位的电平会一直保持在USCIx_DAT0引脚上直到下次串行总线时钟的边沿和下次数据开始。

- 一个数据通道半双工SPI传输

  在一个数据通道半双工SPI传输下，只有一根数据引脚用于数据传输。因此数据发送和接收不能同时进行。数据移位方向由PORTDIR (USPI_TXDAT[16])决定。更多信息参考寄存器描述

  一个数据通道半双工SPI传输类似于全双工SPI协议。所有传输数据时序等同全双工SPI传输

  图6.26-14是一个输出数据通道和一个输入数据通道于外部设备进行半双工传输的框图

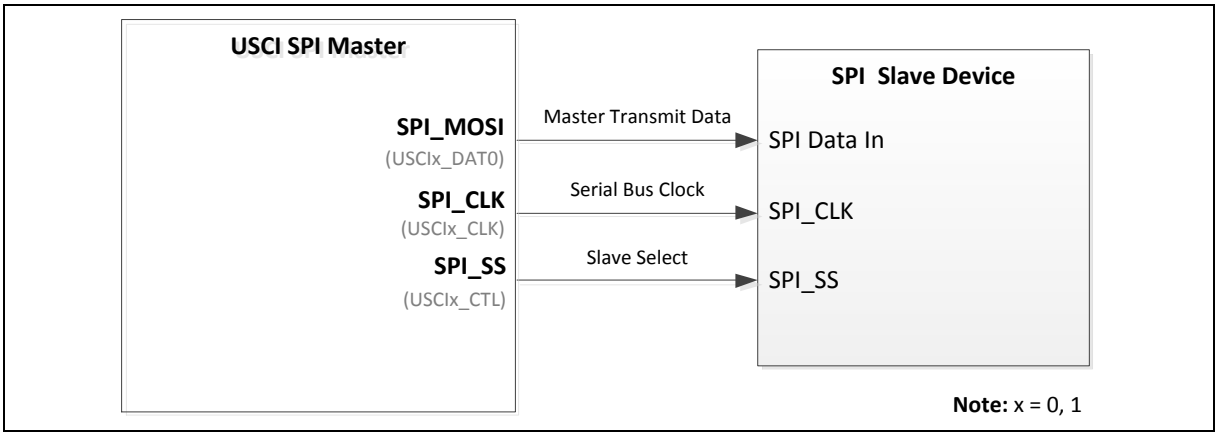


图 6.26-14 一个输出数据通道半双工通信(SPI 主机模式)

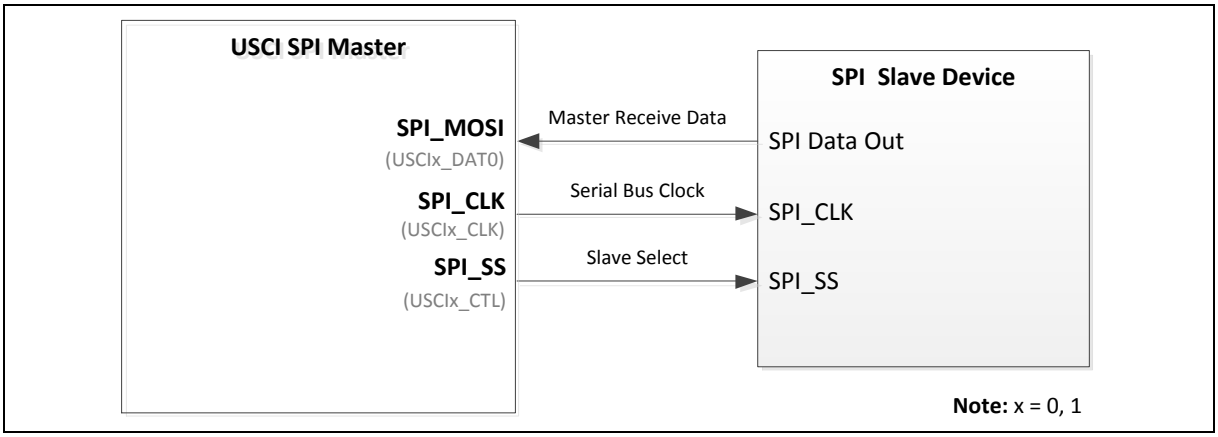


图 6.26-15 一个输入数据通道半双工通信(SPI 主机模式)

一个数据通道半双工传输模式是通过设置TSMSEL[2:0] (USPI_PROTCTL[14:12]) 和 PORTDIR (USCI_TXDAT[16])来配置的。当TSMSEL (USPI_PROTCTL[14:12])设置为0x4时，一个数据通道半双工传输模式被选择。PORTDIR (USPI_TXDAT[16])被用作定义相应传输数据的方向。当PORTDIR位被设置为0时，USCI控制器发送相应数据到外部SPI设备。当PORTDIR位被设置为1时，USCI控制器从外部SPI设备读相应的数据。

比如，在一个数据通道半双工传输模式下，如果PORTDIR=0，USCI SPI用USCIx_DAT0引脚发送数据；如果PORTDIR=1，USCI SPI用USCIx_DAT0引脚接收数据

6.26.5.10 中断

数据传输中断

- 发送起始中断

  一次数据发送的第一个数据位的起始之后，中断事件TXSTIF (USPI_PROTSTS[1])置位。该位可以写1清除。

- 发送结束中断

  在发送缓冲的最后发送数据的最后数据位结束之后，中断事件TXENDIF (USPI_PROTSTS[2])置位。该位可以写1清除。

- 接收起始中断

  一次接收数据的第一个数据位的起始之后，中断事件RXSTIF (USPI_PROTSTS[3])置位。该位可以写1清除。

- 接收结束中断

  接收数据的最后数据位结束之后，中断事件RXENDIF (USPI_PROTSTS[4])置位。该位可以写1清除。

相关协议中断

- SPI从机选择中断

  在SPI从机模式下，当SLAVE (USPI_PROTCTL [0])被置1并且从机检测到片选信号激活或非激活的时候，将会有从机选择激活SSACTIF (USPI_PROTSTS[9])和非激活SSINAIF (USPI_PROTSTS[8])中断标志被置1。如果SSINAIEN (USPI_PROTIEN[0])或SSACTIEN (USPI_PROTIEN[1])置1，SPI控制器会产生一个中断。因为SPI功能的内部从机片选信号是高电平激活，所以ININV (USPI_CTLIN0[2])可以被用作反相来自低电平激活设备的片选信号。

- 从机超时中断

  在SPI从机模式下，在一次字传输期间将会有从机超时功能被用于了解有没串行时钟输入。从机超时功能使用时序测量计数器来进行从机超时周期计算，该超时周期由SLVTOCNT (USPI_PROTCTL[25:16])定义。TMCNTSRC (USPI_BRGEN[5])被用于时序测量计数器时钟频率的选择。
  在SPI从机模式下当时序测量计数器由TMCNTEN (USPI_BRGEN[4])使能并且SLVTOCNT (USPI_PROTCTL[25:16])不为0的情况下，在每次接收字数据的第一个输入串行时钟之后，时序测量计数器会开始计数。计数器会在接收到随后的输入串行时钟时被复位，然后继续计数。最后，在当前字传输完成之后，时序测量计数器会被清除并且停止。在一个字传输完成之前，如果超时计数器的值大于等于SLVTOCNT (USPI_PROTCTL[25:16])的值，从机超时中断事件会发生并且SLVTOIF (USPI_PROTSTS[5])会被置1。

相关缓冲中断

如果USCI控制器中有发送/接收缓冲，相关缓冲中断有效。

- 接收缓冲上溢中断

  如果接收缓冲上溢事件发生，RXOVIF (USPI_BUFSTS[3])会被置1。该位写1清除。

- 发送缓冲下溢中断

  如果接收缓冲下溢事件发生，TXUDRIF (USPI_BUFSTS[11])会被置1。该位写1清除。

6.26.5.11 时序图

USCI SPI协议的从机片选信号默认值是高电平激活，可以设置CTLOINV (USPI_LINECTL[7])反相。

串行时钟的空闲状态和被用于发送/接收数据的串行总线时钟边沿可以通过设置SCLKMODE (USPI_PROTCTL[7:6])来配置。传输数据字的位宽度由DWIDTH (USPI_LINECTL[11:8])决定，数据位传输顺序由LSB (USPI_LINECTL[0])决定。主机/从机操作和相关设置如下面四组SPI时序图所示。

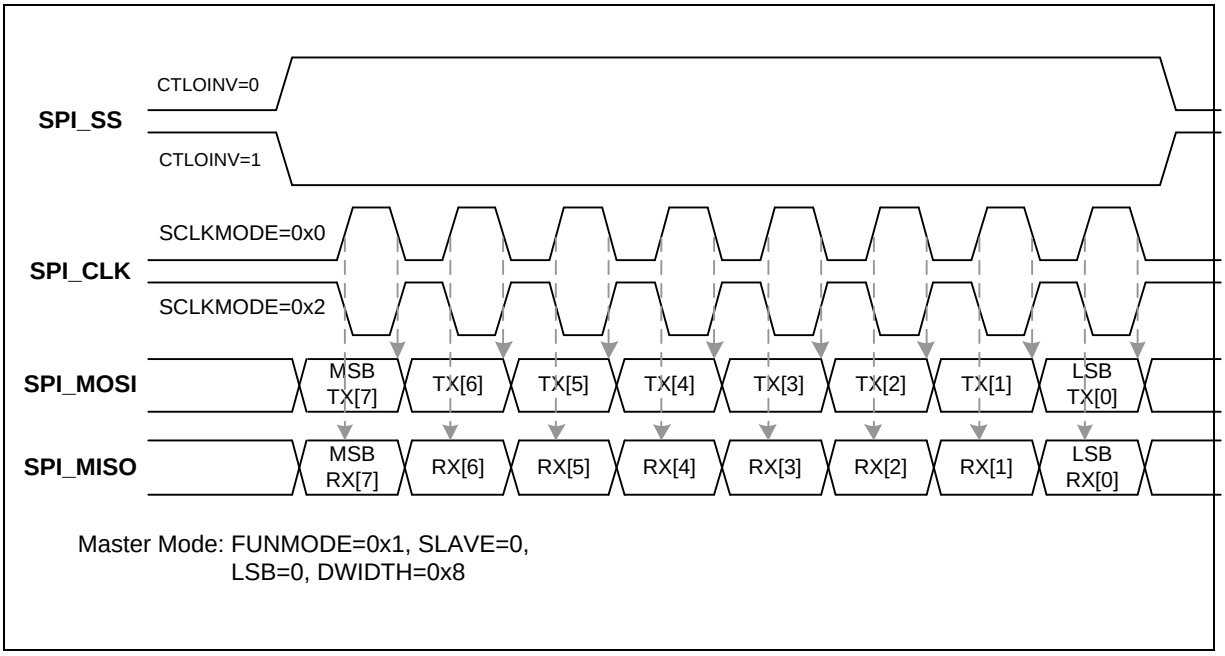


图 6.26-16 主机模式下 SPI 时序

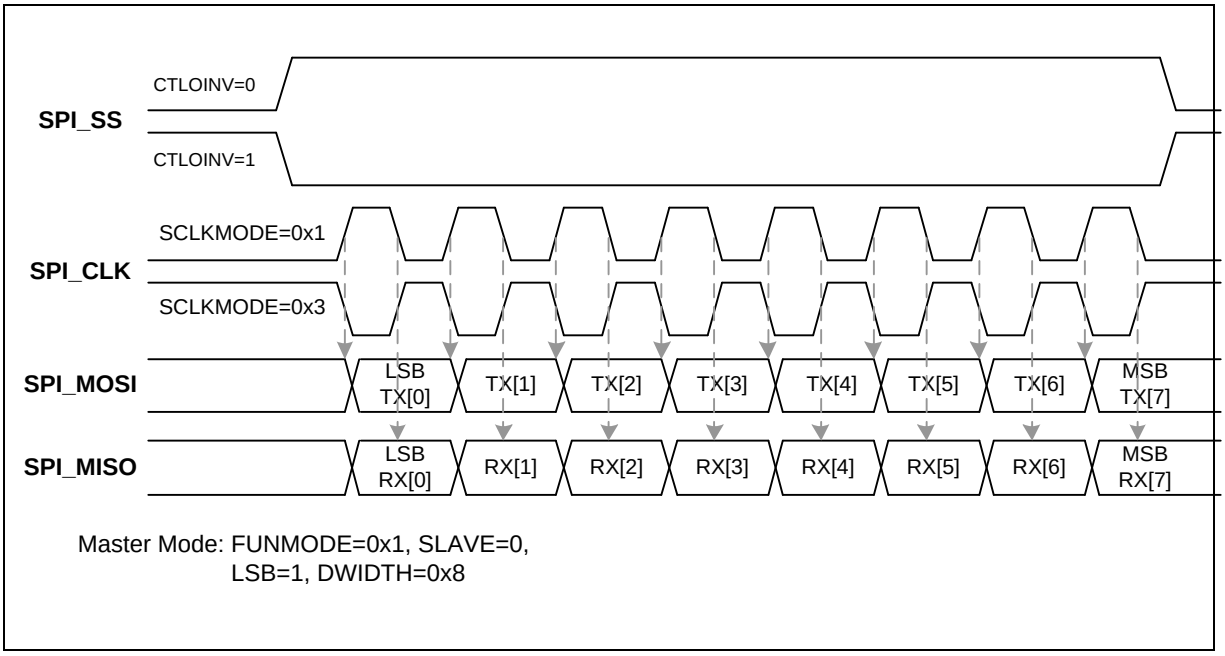


图 6.26-17 主机模式下 SPI 时序 (串行总线时钟的反相位)

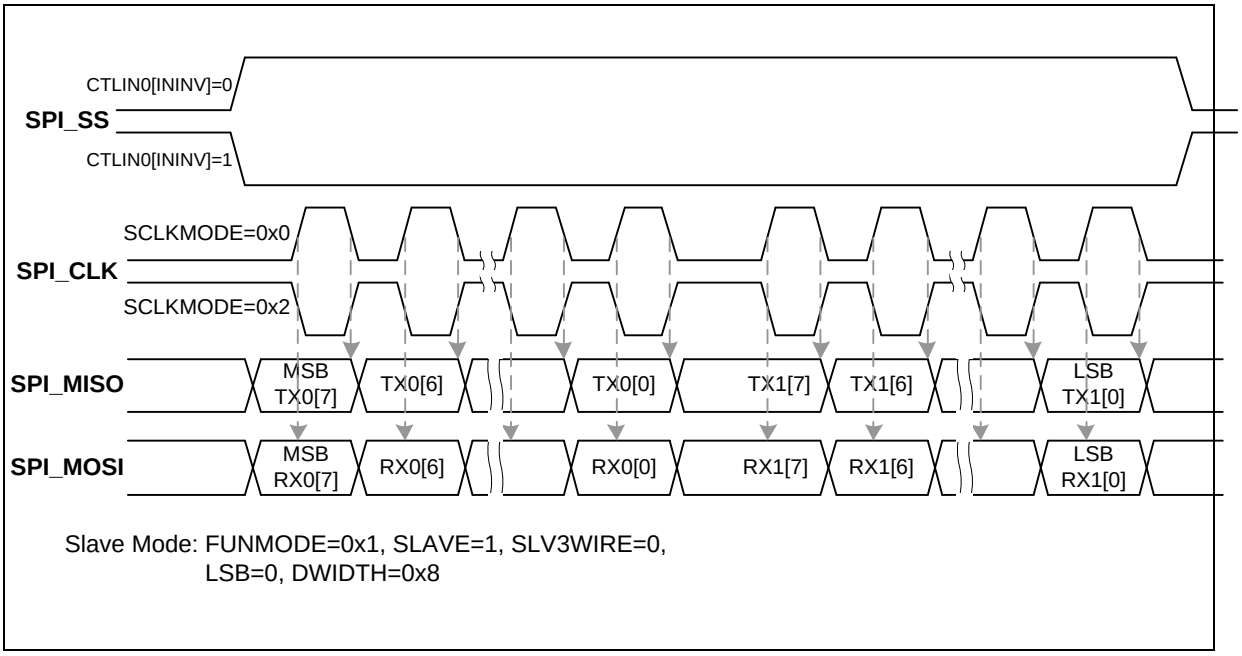


图 6.26-18 从机模式下 SPI 时序

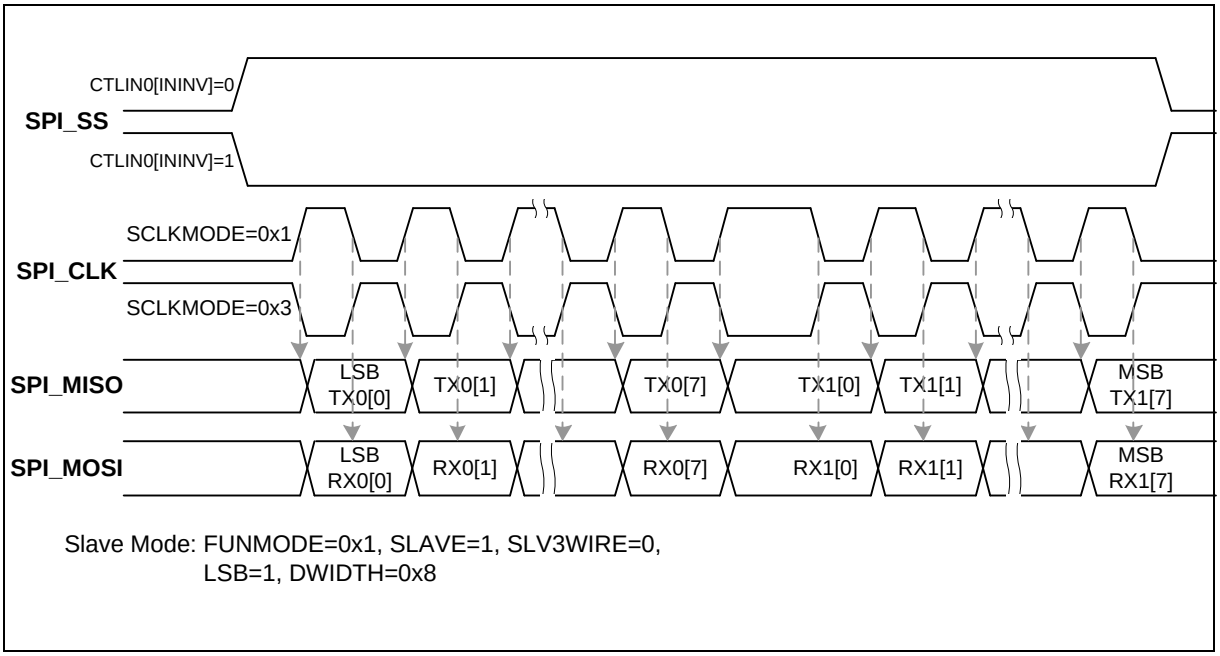


图 6.26-19 从机模式下 SPI 时序(串行总线时钟的反相位)

6.26.5.12 编程流程

该章节描述了USCI SPI数据传输的编程流程。

主机模式：

1. 设置CLK_APBCLK1寄存器，使能USCI外设时钟。

2. 通过设置相关多功能控制寄存器配置用配置用户指定的引脚作为USCI功能引脚。
3. 设置FUNMODE (USPI_CTL[2:0])为1来选择SPI模式。
4. 设置USPI_BRGEN寄存器来确定SPI总线时钟频率。
5. 根据用户应用需求，配置设置如下。
   - CTLOINV (USPI_LINECTL[7]):如果从机片选信号激活时低电平，设置该位为1，否则设置为0
   - DWIDTH (USPI_LINECTL[11:8]):数据宽度设置
   - LSB (USPI_LINECTL[0]): LSB优先还是MSB 优先
   - TSMSEL (USPI_PROTCTL[14:12]):全双工SPI传输或一个通道半双工SPI传
   - SCLKMODE (USPI_PROTCTL[7:6]):确定时钟时序
   - AUTOSS (USPI_PROTCTL[3]):是否使能自动从机片选功能
   - SLAVE (USPI_PROTCTL[0]):主机模式设置为0
   - 设置PROTEN (USPI_PROTCTL[31])为1来使能SPI协议
6. 如果禁用(AUTOSS=0)自动从机片选功能，在数据传输前设置SS (USPI_PROTCTL[2])为1；设置SS为0来取消从机选择信号。
7. 写USPI_TXDAT寄存器来触发SPI传输。在半双工SPI传输中，数据引脚方向通过设置PORTDIR (USPI_TXDAT[16])来确定。
8. 只要RXEMPTY (USPI_BUFSTS[0])为0，用户就可以通过读取USPI_RXDAT寄存器来获取接收的数据。只要TXFULL (USPI_BUFSTS[9])为0，就可以通过写入USPI_TXDAT寄存器来触发SPI数据传输。

从机模式:

1. 设置CLK_APBCLK1寄存器，使能USCI外设时钟。
2. 通过设置相关多功能控制寄存器来配置用户指定的引脚作为USCI功能引脚。
3. 设置FUNMODE (USPI_CTL[2:0])为1来选择SPI模式。
4. 根据用户应用需求，配置设置如下。
   - ININV (USPI_CTLIN0[2]):如果从机片选信号激活时低电平，设置该位为1，否则设置为0
   - DWIDTH (USPI_LINECTL[11:8]): 数据宽度设置
   - LSB (USPI_LINECTL[0]): LSB优先还是MSB 优先
   - TSMSEL (USPI_PROTCTL[14:12]):全双工SPI传输或一个通道半双工SPI传输
   - SCLKMODE (USPI_PROTCTL[7:6]):确定时钟时序
   - SLAVE (USPI_PROTCTL[0]):从机模式设置为1
5. 设置 PROTEN (USPI_PROTCTL[31])为1来使能SPI协议。
6. 写USPI_TXDAT寄存器来触发SPI传输。在半双工SPI传输中，数据引脚方向通过设置PORTDIR (USPI_TXDAT[16])来确定。
7. 只要RXEMPTY (USPI_BUFSTS[0])为0，用户就可以通过读USPI_RXDAT寄存器来获取接收的数据。只要TXFULL (USPI_BUFSTS[9])为0，下次发送的数据就可以写入到USPI_TXDAT寄存

器中。

6.26.5.13 唤醒功能

在SPI模式下USCI控制器支持唤醒系统功能。SPI协议中唤醒源是输入片选信号的转变。

6.26.6 寄存器映射

R:只读, **W:**只写, **R/W:**读/写

寄存器	寄存器	R/W	寄存器	寄存器
USCI_SPI基地址: **USPIn_BA = 0x400D_0000 + (0x1000 * n)** **n= 0, 1**				
USPI_CTL	USPIn_BA+0x00	R/W	USCI控制寄存器	0x0000_0000
USPI_INTEN	USPIn_BA+0x04	R/W	USCI中断使能寄存器	0x0000_0000
USPI_BRGEN	USPIn_BA+0x08	R/W	USCI波特率发生寄存器	0x0000_3C00
USPI_DATIN0	USPIn_BA+0x10	R/W	USCI输入数据信号配置寄存器0	0x0000_0000
USPI_CTLIN0	USPIn_BA+0x20	R/W	USCI输入控制信号配置寄存器0	0x0000_0000
USPI_CLKIN	USPIn_BA+0x28	R/W	USCI输入时钟信号配置寄存器	0x0000_0000
USPI_LINECTL	USPIn_BA+0x2C	R/W	USCI线控制寄存器	0x0000_0000
USPI_TXDAT	USPIn_BA+0x30	W	USCI发送数据寄存器	0x0000_0000
USPI_RXDAT	USPIn_BA+0x34	R	USCI接收数据寄存器	0x0000_0000
USPI_BUFCTL	USPIn_BA+0x38	R/W	USCI发送/接收缓冲控制寄存器	0x0000_0000
USPI_BUFSTS	USPIn_BA+0x3C	R	USCI发送/接收缓冲状态寄存器	0x0000_0101
USPI_PDMACTL	USPIn_BA+0x40	R/W	USCI PDMA控制寄存器	0x0000_0000
USPI_WKCTL	USPIn_BA+0x54	R/W	USCI唤醒控制寄存器	0x0000_0000
USPI_WKSTS	USPIn_BA+0x58	R/W	USCI唤醒状态寄存器	0x0000_0000
USPI_PROTCTL	USPIn_BA+0x5C	R/W	USCI协议控制寄存器	0x0000_0300
USPI_PROTIEN	USPIn_BA+0x60	R/W	USCI协议中断使能寄存器	0x0000_0000
USPI_PROTSTS	USPIn_BA+0x64	R/W	USCI协议状态寄存器	0x0000_0000

6.26.7 寄存器描述

USCI控制寄存器 (USPI_CTL)

寄存器	寄存器	R/W	寄存器	寄存器
USPI_CTL	USPIn_BA+0x00	R/W	USCI控制寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved					**FUNMODE**		

位	寄存器	
[31:3]	**Reserved**	保留
[2:0]	**FUNMODE**	**功能模式** 该域是选择USCI控制器的协议。待选的协议是不可用的。当两个协议之间切换时，在选择新协议前USCI必须先禁用。在用户写000到FUNMODE同时USCI会被复位。 000 = 禁用USCI，所有协议相关的状态机器被设置为空闲状态。 001 = 选择SPI协议 010 = 选择UART协议 100 = 选择I^2C协议 **注:**其它值保留。

USCI中断使能寄存器 (USPI_INTEN)

寄存器	寄存器	R/W	寄存器	寄存器
USPI_INTEN	USPIn_BA+0x04	R/W	USCI中断使能寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved			**RXENDIEN**	**RXSTIEN**	**TXENDIEN**	**TXSTIEN**	**Reserved**

位	寄存器	
[31:5]	**Reserved**	保留
[4]	**RXENDIEN**	**接收结束中断使能位** 该位是使能发生接收结束事件时候产生中断 0 = 禁用接收结束中断 1 = 使能接收结束中断
[3]	**RXSTIEN**	**接收起始中断使能位** 该位是使能发生接收起始事件时候产生中断 0 = 禁用接收起始中断 1 = 使能接收起始中断
[2]	**TXENDIEN**	**发送结束中断使能位** 该位是使能发生发送结束事件时候产生中断 0 = 禁用发送结束中断 1 = 使能发送结束中断
[1]	**TXSTIEN**	**发送起始中断使能位** 该位是使能发生发送起始事件时候产生中断 0 = 禁用发送起始中断 1 = 使能发送起始中断
[0]	**Reserved**	保留

USCI波特率发生寄存器 (USPI_BRGEN)

寄存器	寄存器	R/W	寄存器	寄存器
USPI_BRGEN	USPIn_BA+0x08	R/W	USCI波特率发生寄存器	0x0000_3C00

31	30	29	28	27	26	25	24
Reserved						**CLKDIV**	
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
CLKDIV							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved		**TMCNTSRC**	**TMCNTEN**	**SPCLKSEL**		**PTCLKSEL**	**RCLKSEL**

位	寄存器	
[31:26]	**Reserved**	保留
[25:16]	**CLKDIV**	**时钟分频器** 该域是定义协议时钟频率f_{PROT_CLK}和时钟分频器频率f_{DIV_CLK} (f_{DIV_CLK}= f_{PROT_CLK}/(CLKDIV+1)).之间的比例 **注:**在UART功能下，当自动波特率功能(ABREN(USCI_PROTCTL[6]))使能时，会在输入数据0x55的第四个下降沿时被硬件更新。修改值是在位5和位6之间的时间平均值。用户可以使用修改的CLKDIV和新的BRDETITV (USCI_PROTCTL[24:16])来计算精确的波特率。
[15:6]	**Reserved**	保留
[5]	**TMCNTSRC**	**时序测量计数器时钟源选择** 0 = 来自f_{PROT_CLK}. 1 = 来自f_{DIV_CLK}.
[4]	**TMCNTEN**	**时序测量计数器使能位** 该位是使能10位时序测量计数器 0 = 禁用时序测量计数器 1 = 使能时序测量计数器
[3:2]	**SPCLKSEL**	**采样时钟源选择** 该域用于对协议采样时钟(f_{SAMP_CLK}) 的时钟源选择 00 = f_{DIV_CLK}. 01 = f_{PROT_CLK}. 10 = f_{SCLK}. 11 = f_{REF_CLK}.
[1]	**PTCLKSEL**	**协议协议时钟源选择** 该位用于选择协议时钟(f_{PROT_CLK})的源信号 0 = 参考时钟f_{REF_CLK}.

		1 = f_{REF_CLK2} (频率是f_{REF_CLK}一半)
[0]	**RCLKSEL**	**参考时钟源选择** 该位是选择参考时钟(f_{REF_CLK})源信号 0 = 外设设备时钟$f_{PCLK.}$ 1 = 保留

USCI输入数据信号配置寄存器 (USPI_DATIN0)

寄存器	寄存器	R/W	寄存器	寄存器
USPI_DATIN0	USPIn_BA+0x10	R/W	USCI输入数据信号配置寄存器0	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved					**ININV**	**Reserved**	**SYNCSEL**

位	寄存器	
[31:3]	**Reserved**	保留
[2]	**ININV**	**输入信号反相选择** 该域是使能对输入异步信号的反相 0 = 对异步输入信号不反相 1 = 对异步输入信号反相 **注:**在 SPI协议下，我们建议该位设置为0。
[1]	**Reserved**	保留
[0]	**SYNCSEL**	**输入信号同步选择** 该位是选择异步输入（带反相的）信号或是同步（带过滤的）信号被用作数据移位单元的输入 0 = 异步信号作为信号被用作数据移位单元的输入 1 = 同步信号作为信号被用作数据移位单元的输入 **注:**在 SPI协议下，我们建议该位设置为0。

USCI输入控制信号配置寄存器(USPI_CTLIN0)

寄存器	寄存器	R/W	寄存器	寄存器
USPI_CTLIN0	USPIn_BA+0x20	R/W	USCI输入控制信号配置寄存器0	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved					**ININV**	**Reserved**	**SYNCSEL**

位	寄存器	
[31:3]	**Reserved**	保留
[2]	**ININV**	**输入信号反相选择** 该域是定义使能对输入异步信号输入反相 0 = 对异步输入信号不反相 1 = 对异步输入信号不反相
[1]	**Reserved**	保留
[0]	**SYNCSEL**	**输入同步信号选择** 该位是选择异步输入（带反相的）信号或是同步（带过滤的）信号被用作数据移位单元的输入 0 = 异步信号作为信号被用作数据移位单元的输入 1 = 同步信号作为信号被用作数据移位单元的输入 **注:**在 SPI协议下,我们建议该位设置为0。

USCI输入时钟信号配置寄存器 (USPI_CLKIN)

寄存器	寄存器	R/W	寄存器	寄存器
USPI_CLKIN	USPIn_BA+0x28	R/W	USCI输入时钟信号配置寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved							**SYNCSEL**

位	寄存器	
[31:1]	**Reserved**	保留
[0]	**SYNCSEL**	**输入同步信号选择** 该位是选择异步输入（带反相的）信号或是同步（带过滤的）信号被用作数据移位单元的输入 0 =异步信号作为信号被用作数据移位单元的输入 1 = 同步信号作为信号被用作数据移位单元的输入 **注:**在 SPI协议下,我们建议该位设置为0。

USCI线控制寄存器 (USPI_LINECTL)

寄存器	寄存器	R/W	寄存器	寄存器
USPI_LINECTL	USPIn_BA+0x2C	R/W	USCI线控制寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved				**DWIDTH**			
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
CTLOINV	**Reserved**	**DATOINV**	**Reserved**				**LSB**

位	寄存器	
[31:12]	**Reserved**	保留
[11:8]	**DWIDTH**	**传输的字长** 该域定义发送和接收的数据字长。在数据缓冲中数据总是右对齐。USCI支持4到16位的字长 0x0:在数据字中包含16位，位于[15:0]这些位中 0x1: 保留 0x2: 保留 0x3: 保留 0x4:在数据字中包含4位，位于[3:0]这些位中 0x5:在数据字中包含5位，位于[4:0]这些位中 ... 0xF:在数据字中包含15位，位于[14:0]这些位中
[7]	**CTLOINV**	**控制信号输出反相选择** 该位定义了内部控制信号和输出控制信号之间的关系 0 = 无影响 1 = 在输出前控制信号被反相 **注:**在不同协议中控制信号有不同的定义。SPI协议中，控制信号是从机片选信号。
[6]	**Reserved**	保留
[5]	**DATOINV**	**数据输出反相选择** 该位定义了内部移位数据值和USCIx_DAT0/1引脚输出数据信号之间的关系 0 = 数据输出电不反相 1 = 数据输出电反相
[4:1]	**Reserved**	保留
[0]	**LSB**	**LSB优先传输选择**

		0 = 取决于DWIDTH设置的接收/发送数据缓冲的MSB被先发送/接收 1 = 数据缓冲的位0及LSB被先发送/接收

USCI发送数据寄存器(USPI_TXDAT)

寄存器	寄存器	R/W	寄存器	寄存器
USPI_TXDAT	USPIn_BA+0x30	W	USCI发送数据寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							**PORTDIR**
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
TXDAT							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
TXDAT							

位	寄存器	
[31:17]	**Reserved**	保留
[16]	**PORTDIR**	**端口方向控制** 该位域仅在USCI工作在SPI的半双工传输下有效。它被用来定义数据端口引脚的方向。当软件写USCI_TXDAT寄存器时，发送数据和端口方向同时被处理 0 = 数据引脚被配置成输出模式 1 = 数据引脚被配置成输入模式
[15:0]	**TXDAT**	**发送数据** 软件可以写16位发送数据到该域用来发送为了避免发送数据写溢出，用户必须在写发送数据到该域之前检查TXEMPTY (USPI_BUFSTS[8])状态

USCI接收数据寄存器 (USPI_RXDAT)

寄存器	寄存器	R/W	寄存器	寄存器
USPI_RXDAT	USPIn_BA+0x34	R	USCI接收数据寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
RXDAT							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
RXDAT							

位	寄存器	
[31:16]	**Reserved**	保留
[15:0]	**RXDAT**	**接收的数据** 该域值监控存储在接收数据缓冲的接收到的数据

USCI发送/接收缓冲控制寄存器 (USPI_BUFCTL)

寄存器	寄存器	R/W	寄存器	寄存器
USPI_BUFCTL	USPIn_BA+0x38	R/W	USCI发送/接收缓冲控制寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved						RXRST	TXRST
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
RXCLR	RXOVIEN	Reserved					
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
TXCLR	TXUDRIEN	Reserved					

位	寄存器	
[31:18]	**Reserved**	保留
[17]	**RXRST**	**接收复位** 0 = 无影响 1 = 复位接收相关计数器、状态机、接收移位寄存器的内容和数据缓冲 **注:** 1个PCLK周期后自动被清除。
[16]	**TXRST**	**发送复位** 0 = 无影响 1 = 复位发送相关计数器、状态机、发送移位寄存器的内容和数据缓冲 **注1:** 1个PCLK周期后自动被清除。 **注2:**如果USPI_BUFCTL[5]=0，写1到该位将会拉低输出数据引脚。
[15]	**RXCLR**	**清接收缓冲** 0 = 无影响 1 = 接收缓冲被清除。仅当缓冲没有参与数据通信时被使用 **注:** 1个PCLK周期后自动被清除。
[14]	**RXOVIEN**	**接收缓冲溢出中断使能控制** 0 = 禁用接收上溢出中断 1 = 使能接收上溢出中断
[13:8]	**Reserved**	保留
[7]	**TXCLR**	**清发送缓冲** 0 = 无影响 1 = 发送缓冲被清除。仅当缓冲没有参与数据通信时被使用 **注:** 1个PCLK周期后自动被清除。

[6]	**TXUDRIEN**	**从机发送下溢出中断使能位** 0 =禁用发送下溢出中断 1 =使能发送下溢出中断
[5:0]	**Reserved**	保留

USCI发送/接收缓冲状态寄存器 (USPI_BUFSTS)

寄存器	寄存器	R/W	寄存器	寄存器
USPI_BUFSTS	USPIn_BA+0x3C	R	USCI发送/接收缓冲状态寄存器	0x0000_0101

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved				**TXUDRIF**	**Reserved**	**TXFULL**	**TXEMPTY**
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved				**RXOVIF**	**Reserved**	**RXFULL**	**RXEMPTY**

位	寄存器	
[31:12]	**Reserved**	**发送缓冲下溢出中断状态** 该位表示发送缓冲下溢出事件被侦测到。如果TXUDRIEN (USPI_BUFCTL[6])使能，会激活相应的中断请求。软件写**1**清除该位。 0 = 发送缓冲下溢出事件未被侦测到 1 = 发送缓冲下溢出事件被侦测到
[11]	**TXUDRIF**	保留
[10]	**Reserved**	**发送缓冲满指示** 0 = 发送缓冲还没满 1 = 发送缓冲满
[9]	**TXFULL**	**发送缓冲空指示** 0 = 发送缓冲非空 1 = 发送缓冲为空，可进行下次数据发送
[8]	**TXEMPTY**	保留
[7:4]	**Reserved**	**接收缓冲上溢出中断状态** 该位表示接收缓冲上溢出事件被侦测到。如果RXOVIEN (USPI_BUFCTL[14])使能，会激活相应的中断请求。软件写**1**清除该位。 0 = 接收缓冲上溢出事件未被侦测到 1 = 接收缓冲上溢出错件被侦测到
[3]	**RXOVIF**	保留
[2]	**Reserved**	**接收缓冲满指示** 0 =接收缓冲还没满 1 =接收缓冲满

[1]	**RXFULL**	**接收缓冲空指示** 0 =接收缓冲非空 1 =接收缓冲为空
[0]	**RXEMPTY**	**发送缓冲下溢出中断状态** 该位表示发送缓冲下溢出事件被侦测到。如果TXUDRIEN (USPI_BUFCTL[6])使能，会激活相应的中断请求。软件写**1**清除该位。 0 = 发送缓冲下溢出事件未被侦测到 1 = 发送缓冲下溢出事件被侦测到

USCI PDMA控制寄存器 (USPI_PDMACTL)

寄存器	寄存器	R/W	寄存器	寄存器
USPI_PDMACTL	USPIn_BA+0x40	R/W	USCI PDMA控制寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved				**PDMAEN**	**RXPDMAEN**	**TXPDMAEN**	**PDMARST**

位	寄存器	
[31:4]	**Reserved**	保留
[3]	**PDMAEN**	**PDMA模式使能位** 0 = 禁用PDMA功能 1 = 使能PDMA功能 注: I^2C 不支持 PDMA功能。
[2]	**RXPDMAEN**	**PDMA接收通道有效** 0 = 禁用PDMA接收功能 1 = 使能PDMA接收功能
[1]	**TXPDMAEN**	**PDMA发送通道有效** 0 = 禁用PDMA发送功能 1 = 使能PDMA发送功能
[0]	**PDMARST**	**PDMA 复位** 0 = 无影响 1 = 复位USCI的PDMA控制逻辑。该位会自动地被清0

USCI唤醒控制寄存器 (USPI_WKCTL)

寄存器	寄存器	R/W	寄存器	寄存器
USPI_WKCTL	USPIn_BA+0x54	R/W	USCI唤醒控制寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved					**PDBOPT**	**Reserved**	**WKEN**

位	寄存器	
[31:3]	**Reserved**	保留
[2]	**PDBOPT**	**掉电阻塞选择** 0 = 如果用户试图在协议正在传输的时候执行WFI来进入掉电模式，MCU会停止传输并且立即进入掉电模式 1 = 如果用户试图在协议正在传输的时候执行WFI来进入掉电模式，传输会继续进行并且MCU立即进入掉电模式
[1]	**Reserved**	**唤醒地址匹配使能位** 0 = 数据唤醒芯片 1 = 地址匹配唤醒芯片
[0]	**WKEN**	**唤醒使能位** 0 = 禁用唤醒功能 1 = 使能唤醒功能

USCI唤醒状态寄存器 (USPI_WKSTS)

寄存器	寄存器	R/W	寄存器	寄存器
USPI_WKSTS	USPIn_BA+0x58	R/W	USCI唤醒状态寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved							**WKF**

位	寄存器	
[31:1]	**Reserved**	保留
[0]	**WKF**	**唤醒标志** 当芯片被从掉电模式唤醒，该位置1.软件写1清除该位

USCI协议控制寄存器 – USPI_PROTCTL (SPI)

寄存器	寄存器	R/W	寄存器	寄存器
USPI_PROTCTL	USPIn_BA+0x5C	R/W	USCI协议控制寄存器	0x0000_0300

31	30	29	28	27	26	25	24
PROTEN	**Reserved**		**TXUDRPOL**	**Reserved**		**SLVTOCNT**	
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
SLVTOCNT							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved	**TSMSEL**			**SUSPITV**			
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
SCLKMODE		**Reserved**		**AUTOSS**	**SS**	**SLV3WIRE**	**SLAVE**

位	寄存器	
[31]	**PROTEN**	**SPI协议使能位** 0 = 禁用SPI协议 1 = 使能SPI协议
[30:29]	**Reserved**	保留
[28]	**TXUDRPOL**	**发送下溢出数据极性 (仅从机模式)** 该位定义在没有数据可发送时发送数据电平 0 = 如果TX下溢事件发生，输出数据电平为0 1 = 如果TX下溢事件发生，输出数据电平为1
[27:26]	**Reserved**	保留
[25:16]	**SLVTOCNT**	**从机模式超时周期(仅从机模式)** 在从机模式，该域被用作从机超时周期。该域表示在输入SCLK的两个边沿之间有多少时钟周期（时钟由TMCNTSRC，USPI_BRGEN[5]选择）产生从机超时事件。写0x0到该位域禁用从机超时功能 范例：SLVTOCNT为 0x0A 和 TMCNTSRC (USPI_BRGEN[5])为1 1，表示如果SPI总线时钟引脚超过（10+1）个f_{DIV_CLK}周期没有改变，将会有超时事件发生。
[15]	**Reserved**	保留
[14:12]	**TSMSEL**	**传输数据模式选择** 该位域定义发送和接收数据是如何移进和移出的。 TSMSEL = 000b: 全双工SPI TSMSEL = 100b: 半双工SPI 其它值保留 **注:**改变该位域的值将会自动地产生TXRST 和 RXRST来清除TX/RX数据缓冲。
[11:8]	**SUSPITV**	**暂停间隔(仅主模式)**

		该四位用来配置在一次数据传输过程中连续两个发送/接收事务之间的暂停间隔。暂停间隔是从当前事务字的最后一个时钟边沿到接下来的事务的第一个边沿时钟。默认值是0x3。暂停间隔的周期可以根据下面公式获得： (SUSPITV [3:0] + 0.5) * SPI_CLK时钟周期 例: SUSPITV = 0x0 … 0.5 SPI_CLK时钟周期 SUSPITV = 0x1 … 1.5 SPI_CLK时钟周期 …… SUSPITV = 0xE … 14.5 SPI_CLK时钟周期 SUSPITV = 0xF … 15.5 SPI_CLK时钟周期
[7:6]	**SCLKMODE**	**串行总线时钟模式** 该位域定义SCLK空闲状态、数据传输和接收边沿 MODE0 = 空闲状态时是低电平。下降沿时发送数据，上升沿时接收数据 MODE1 = 空闲状态时是低电平。上升沿时发送数据，下降沿时接收数据 MODE2 = 空闲状态时是高电平。上升沿时发送数据，下降沿时接收数据 MODE3 = 空闲状态时是高电平。下降沿时发送数据，上升沿时接收数据
[5:4]	**Reserved**	保留
[3]	**AUTOSS**	**自动从机片选功能使能(仅主模式)** 0 = 通过设置SS (USPI_PROTCTL[2])值来控制从机片选信号 1 = 从机片选信号自动地产生。当发送/接收启动时从机片选信号会被SPI控制激活，在每次发送/接收结束之后片选信号取消
[2]	**SS**	**从机选择控制(仅主模式)** 如果AUTOSS位被清除，设置该位为1将激活从机片选信号，设置设置该位为0将取消从机片选信号 如果AUTOSS功能使能(AUTOSS = 1),该位的设置值不会影响从机片选信号的当前的状态 **注:** SPI 协议下，内部从机片选信号激活状态时为高电平。
[1]	**SLV3WIRE**	**从机3-线模式选择(仅从机模式)** 从机模式下，SPI协议在3线接口（及没有片选信号）下工作 0 = 4线接口 1 = 3线接口
[0]	**SLAVE**	**从机模式选择** 0 = 主机模式 1 = 从机模式

USCI协议中断使能寄存器 – USPI_PROTIEN (SPI)

寄存器	寄存器	R/W	寄存器	寄存器
USPI_PROTIEN	USPIn_BA+0x60	R/W	USCI协议中断使能寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved				**SLVBEIEN**	**SLVTOIEN**	**SSACTIEN**	**SSINAIEN**

位	寄存器	
[31:5]	**Reserved**	保留
[3]	**SLVBEIEN**	**从机模式位计数错误中断使能控制** 如果在从机模式下数据传输由从机超时或是从机片选取消事件终止，这样一来发送/接收的数据位计数就会与DWIDTH (USPI_LINECTL[11:8])设置值不匹配。位计数错误事件发生。 0 = 禁用从机模式位计数错误中断 1 = 使能从机模式位计数错误中断
[2]	**SLVTOIEN**	**从机超时中断使能控制** SPI协议下，使能该中断以防从机超时事件 0 = 禁用从机超时中断 1 = 使能从机超时中断
[1]	**SSACTIEN**	**从机片选激活中断使能控制** 如果从机片选信号变为有效状态，该位是使能/禁用产生从机片选激活中断 0 = 禁用从机片选激活中断 1 = 使能从机片选激活中断
[0]	**SSINAIEN**	**从机片选取消中断使能控制** 如果从机片选信号变为无效状态，该位是使能/禁用产生从机片选取消中断 0 = 禁用从机片选取消中断 1 = 使能从机片选取消中断

USCI协议状态寄存器 – USPI_PROTSTS (SPI)

寄存器	寄存器	R/W	寄存器	寄存器
USPI_PROTSTS	USPIn_BA+0x64	R/W	USCI协议状态寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved					**SLVUDR**	**BUSY**	**SSLINE**
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved						**SSACTIF**	**SSINAIF**
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved	**SLVBEIF**	**SLVTOIF**	**RXENDIF**	**RXSTIF**	**TXENDIF**	**TXSTIF**	**Reserved**

位	寄存器	
[31:19]	**Reserved**	保留
[18]	**SLVUDR**	**从机模式发送下溢出状态（只读）** 在从机模式下，如果没有可用的发送数据在缓冲中同时输入串行总线时钟使发送数据继续移出，这时该位就会被置1.该位置位与字传输当前移出数据是否已切换到TXUDRPOL (USPI_PROTCTL[28])没有关系。 0 = 没有发生从机发送下溢出事件 1 = 发生从机发送下溢出事件
[17]	**BUSY**	**忙状态（只读）** 0 = SPI 处在空闲状态 1 = SPI 处在忙状态 下面列出了总线忙的状况： a. USPI_PROTCTL[31] = 1和TXEMPTY = 0 b. 对于SPI主机模式，TXEMPTY = 1但是当前传输还没完成 c. 对于SPI从机模式，USPI_PROTCTL[31] = 1并且在从机片选激活时有串行时钟输入到SPI内核逻辑单元 d. 对于SPI从机模式，USPI_PROTCTL[31] = 1并且即使在从机片选没有激活状况下发送缓冲或是发送移位寄存器为非空
[16]	**SSLINE**	**从机选择线总线状态（只读）** 仅在从模式下有效。该位是用来监控总线上输入从机片选信号当前的状态。 0 = 从机选择线状态为0 1 = 从机选择线状态为1
[15:10]	**Reserved**	保留
[9]	**SSACTIF**	**从机选择激活中断标志(仅从机模式)** 该位表示内部从机片选信号变为激活状态。软件写1清除该位。

		0 = 从机片选信号没有改变成激活状态 1 = 从机片选信号改变成激活状态 **注:**内部从机片选信号激活状态时为高电平。
[8]	**SSINAIF**	**从机选择取消中断标志(仅从机模式)** 该位表示内部从机片选信号变为非激活状态。软件写1清除该位。 0 = 从机片选信号没有改变成非激活状态 1 = 从机片选信号改变成非激活状态 **注:**内部从机片选信号激活状态时为高电平。
[7]	**Reserved**	保留
[6]	**SLVBEIF**	**从机位计数错误中断标志(仅从机模式)** 0 = 没有发生从机位计数错误事件 1 = 发生从机位计数错误事件 **注:**写1清除该位。
[5]	**SLVTOIF**	**从机超时中断标志(仅从机模式)** 0 = 没有发生从机超时事件 1 = 发生从机超时事件 **注:**写1清除该位。
[4]	**RXENDIF**	**接收结束中断标志** 0 = 没有接收结束中断状态发生 1 = 有接收结束中断状态发生 **注:**写1清除该位。
[3]	**RXSTIF**	**接收起始中断标志** 0 = 没有接收起始中断状态发生 1 = 有接收起始中断状态发生 **注:**写1清除该位。
[2]	**TXENDIF**	**发送结束中断标志** 0 = 没有发送结束中断状态发生 1 = 有发送结束中断状态发生 **注:**写1清除该位。
[1]	**TXSTIF**	**发送起始中断标志** 0 = 没有发送起始中断状态发生 1 = 有发送起始中断状态发生 **注:**写1清除该位。
[0]	**Reserved**	保留

6.27 USCI – I2C模式

6.27.1 概述

在I²C总线上，数据通过时钟线SCL和数据线SDA在主从机间逐一字节同步传送。每个字节数据长度是8位一个SCL时钟脉冲传输一个数据位，数据由最高位MSB 开始传输，每个字节传输后跟随一个应答位，每个位在SCL为高时采样；因此，SDA线可能在SCL为低时改变，但在SCL为高时必须保持稳定。当SCL为高时，SDA线上的跳变视为一个命令(START或STOP)。更多关于I²C总线时序的细节请参考下图6.27-1。

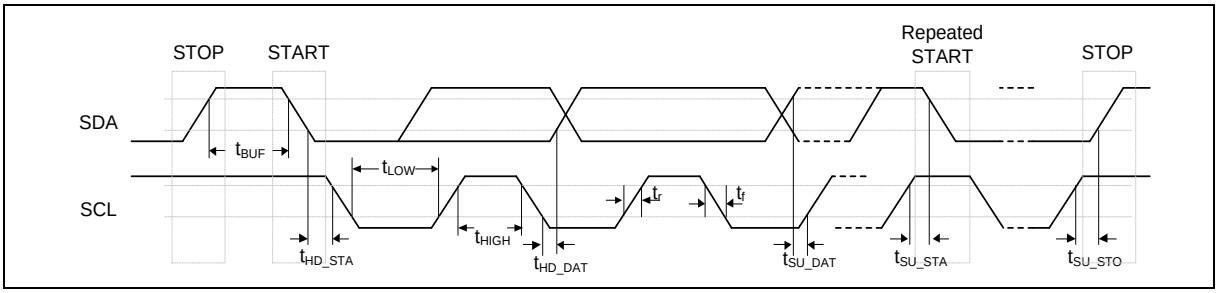


图 6.27-1 I²C 总线时序

设备片上I²C外设提供了一个符合I²C总线规范的串行接口。I²C端口自动处理字节传输。通过设置寄存FUNMODE (UI²C_CTL [2:0]) = 100B选择I²C模式。USCI硬件接口通过数据线SDA和时钟线SCL两个管脚连到I²C总线。当I/O管脚作为I²C端口使用时，用户必须先设定I/O管脚为I²C功能

注: SDA和SCL两个管脚需要上拉电阻，因为这两个管脚在USCI选作I²C模式时为开漏脚。

6.27.2 特性

- 支持主机/从机模式
- 支持7位地址模式和10位地址模式
- 支持标准模式(100 kbps)和快速模式(400 kbps)
- 支持多主机总线
- 支持一个发送缓冲和两个接收缓冲
- 支持10位总线超时
- 支持总线监控模式
- 支持掉电状况下由数据或是地址匹配唤醒
- 支持建立/保持时间可编程
- 支持多地址识别（2组从机地址带mask选项）

6.27.3 框图

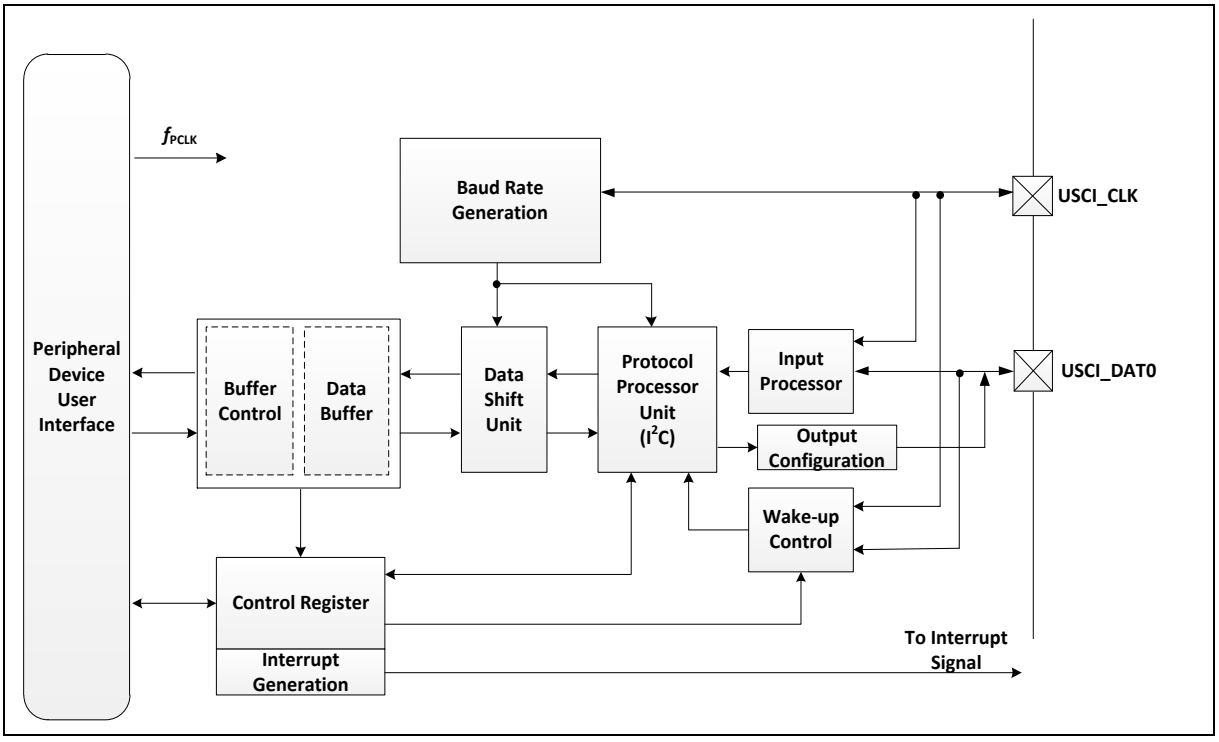


图 6.27-2USCI I²C 模式框图

6.27.4 基本配置

6.27.4.1 USCI0 I²C基本配置

- 时钟源配置
  - 使能USCI0外设时钟位USCI0CKEN (CLK_APBCLK1[8])
  - 使能USCI0_I2C功能寄存器UI²C_CTL[2:0])，UI²C_CTL[2:0]=3'b100
- 复位配置
  - 复位USCI0控制位USCI0RST (SYS_IPRST2[8])
- 引脚配置

组	引脚名称	GPIO	MFP
USCI0	USCI0_CLK	PD.0	MFP3
		PB.12	MFP5
		PA.11	MFP6
		PE.2	MFP7
	USCI0_DAT0	PD.1	MFP3
		PB.13	MFP5
		PA.10	MFP6

		PE.3	MFP7

6.27.4.2 USCI1 I²C基本配置

- 时钟源配置
  - 使能USCI1外设时钟位USCI1CKEN (CLK_APBCLK1[9])
  - 使能USCI1_I2C功能寄存器UI²C_CTL[2:0])，UI²C_CTL[2:0]=3'b100
- 复位配置
  - 复位USCI1控制位USCI1RST (SYS_IPRST2[9])
- 引脚配置

组	引脚名称	GPIO	MFP
USCI1	USCI1_CLK	PB.8	MFP4
		PD.7, PE.12	MFP6
		PB.1	MFP8
	USCI1_DAT0	PB.7	MFP4
		PD.5, PE.10	MFP6
		PB.2	MFP8

6.27.5 功能描述

6.27.5.1 起始或重复起始信号

图6.27-3展示了标准I²C协议。通常,标准通讯包含四个部分：

- 起始或重复起始信号
- 从机地址传输和读/写位传输
- 数据传输
- 停止信号

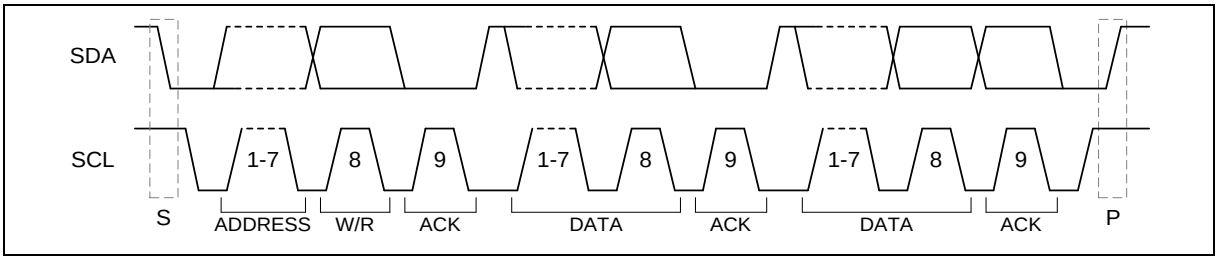


图 6.27-3I²C 协议

当总线处于释放/空闲状态下，说明没有主机设备占用总线（SDA与SCL线同时为高），主机可以通过发送起始(START)信号来发起传输过程。起始信号，通常表示为"S"位，当SCL线为高时，SDA线上信号由高至低变化，就被定义为起始信号。起始信号表示一个新的数据传输的开始。

重复起始不是两个启动信号之间的停止信号，通常称为"Sr"位。主机使用此方法与另一个从机或同一个从机以不同传输方向（例如从写入设备到读取设备）进行通信，而不释放总线空闲标志。

6.27.5.2 停止信号

主机可以通过产生一个停止信号来终止数据通信。停止信号，通常表示为“P”位，当SCL线为高时，SDA线上信号由低至高变化，就被定义为停止信号

图6.27-4展示了起始，重复起始和停止信号的波形。

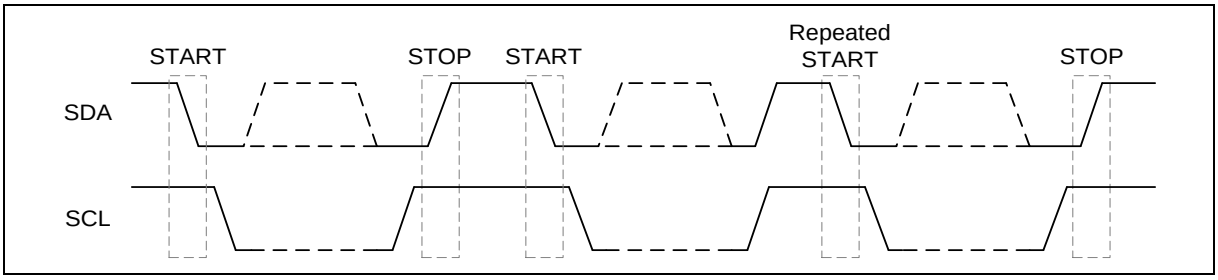


图 6.27-4START 和 STOP 状况

6.27.5.3 从地址传输

在一个起始（或重复起始）信号后，主机发送一个从机地址来确认通信的目标设备。起始地址有一个或两个字节组成（针对7位或10位地址情况）。地址字节过后，从机会对传输的地址作应答反应。

因此，从机地址可以被编程并且与接收到的地址比较。一旦匹配，从机会回一个应答(SDA = 0)如果与目标不匹配则无应答(SDA = 1)除了编程地址匹配之外，如果从机有能力处理类似的请求，那么另外的地址值也必须有应答回应。地址为00H是广播地址也能应答。

为了可选择应答不同的地址值，控制机制是按如下实现的：

- 如果GCFUNC 位(UI^{2}C_PROTCTL [0])被置位，I^{2}C端口硬件就会响应广播地址（00H）。清除GC位禁用广播功能。
- I^{2}C端口装备了一个设备地址寄存器UI^{2}C_DEVADDRn (n = 0~1)。在7位地址模式下，首先接收到的7位地址会与编程的从机地址(UI^{2}C_DEVADDRn [6:0])相比较。如果匹配，从机会回应答。
- 对于10位地址模式，当ADDR10EN (UI^{2}C_PROTCTL [4])被置位时，如果从机地址是1111 0XXB，XX位用于和UI^{2}C_DEVADDR [9:8]比较检测到地址匹配时将会产生应答。从机等待第二个地址字节然后与UI^{2}C_DEVADDR [7:0]比较，依据收到的10位地址来决定是否产生应答。用户必须处理好保留的的地址（详细描述请参考I^{2}C规格书）。仅支持地址1111 0XXB。在这些条件下，当地址匹配时，SLASEL (UI^{2}C_PROTSTS [14])会被置位。SLASEL (UI^{2}C_PROTSTS [14])会被一个（重复）起始或停止条件自动的清除。
- I^{2}C端口有两组地址掩码寄存器I^{2}C_ADDRMSKn (n = 0~1)，这样就能支持多地址识别。当地址掩码寄存器中位设置为1的时候，表示接收相应的地址为不需要关心，如果设置为0，表示接收相应地址位必须和地址寄存器是一样的。

6.27.5.4 数据传输

当从机设备地址和R/W位被成功识别到，就可以根据R/W位所决定的方向按一字节一字节方式进行数据传输。每个字节传输完后，紧接着的第9个SCL时钟周期会有一个应答信号位。如果从机上产生无应答信号(NACK)，主机可以产生停止信号来中止数据传输或者产生重复起始信号开始新一轮数据传输。

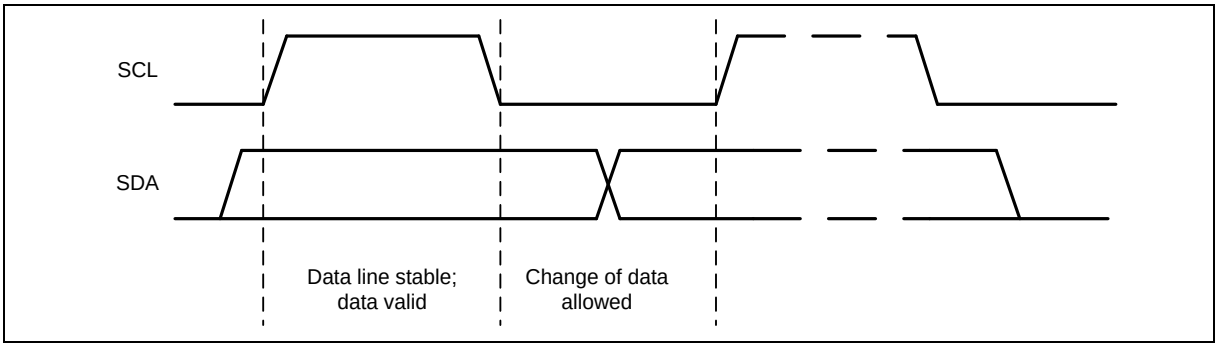


图 6.27-5 I²C 总线上的位传输

当主机作为接收设备时，发生无应答信号(NACK)，则从机将释放SDA线，让主机产生停止信号或重复起始信号。

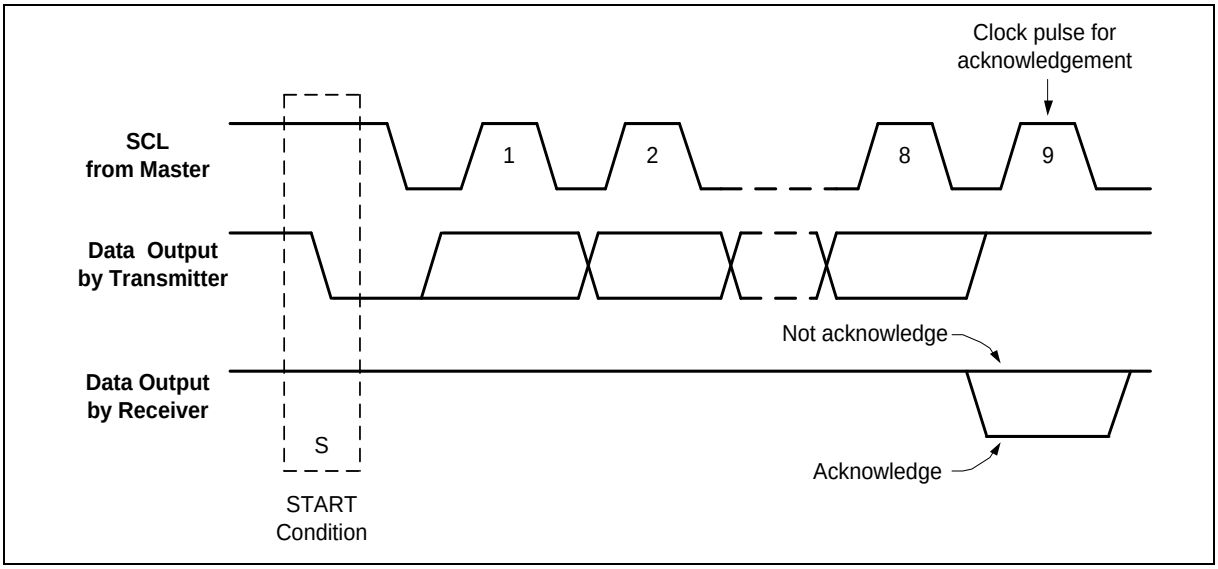


图 6.27-6 I²C 总线上的应答信号

6.27.5.5 始终波特率位

当I²C在主机模式下时，I²C的波特率由USCI_BRGEN寄存器决定，在从机模式下不用管。从机模式下，I²C会自动识别来自主机的时钟的任何频率。在RCLKSEL，SPCLKSEL，PDSCNT和DSCNT中定义波特率：

- RCLKSEL (UI²C_BRGEN [0])

  定义输入频率f_{REF_CLK}

- SPCLKSEL (UI²C_BRGEN[3:2])

  定义采样时钟f_{SAMP_CLK}的时钟源

- PDSCNT (UI²C_BRGEN [9:8])

  定义数据采样时间的长度(由f_{REF_CLK}经1，2，3或4分频)

- DSCNT (UI²C_BRGEN [14:10])

  定义采样时间每位时间的数目

标准设定由RCLKSEL = 0 (f_{REF_CLK} = f_{PCLK})，PTCLKSEL = 0 (f_{PROT_CLK} = f_{REF_CLK})和SPCLKSEL = 2'b00 (f_{SAMP_CLK} = f_{DIV_CLK})决定。在这些条件下，波特率值如下：

$$f_{I2C} = f_{REF_CLK} \times \frac{1}{\text{CLKDIV} + 1} \times \frac{1}{\text{PDSCNT} + 1} \times \frac{1}{\text{DSCNT} + 1}$$

为了产生更慢的频率，设置PTCLKSEL = 1 (f_{PROT_CLK} = fREF_CLK2)选择附加除2阶段，如下：

$$f_{I2C} = \frac{f_{REF_CLK}}{2} \times \frac{1}{\text{CLKDIV} + 1} \times \frac{1}{\text{PDSCNT} + 1} \times \frac{1}{\text{DSCNT} + 1}$$

如果SPCLKSEL = 2'b10 (f_{SAMP_CLK} = f_{SCLK})，和RCLKSEL = 0 (fREF_CLK = f_{PCLK}), PTCLKSEL = 0 (f_{PROT_CLK} = f_{REF_CLK})。波特率如下：

$$f_{I2C} = f_{REF_CLK} \times \frac{1}{\text{CLKDIV} + 1} \times \frac{1}{2} \times \frac{1}{\text{PDSCNT} + 1} \times \frac{1}{\text{DSCNT} + 1}$$

6.27.5.6 字节钳位

如果设备被选用为主机/从机发送模式，在需要发送数据的时候发送缓冲TXDAT没有有效的数据被传输，设备就会在ACK位前拉低SCL到0。在软件写1到PTRG (UI^{2}C_PROTCTL [5])中后等待周期结束。

6.27.5.7 多主机仲裁

在一些应用中，一个I^{2}C总线上有多个主机同时访问从机，并有可能同时在传送数据。I^{2}C是支持多主机模式，并包含有冲突检测和仲裁，防止数据损坏。

如果两个主机同时发I^{2}C命令，通过仲裁来决定哪个优先并继续发命令。仲裁是在SCL为高时在SDA上执行的。每一个主机都会检测总线上的SDA信号是否符合它产生的SDA信号。如果检测到总线上SDA为低，但应该为高，则这个主机将失去仲裁。设备在仲裁丢失后会产生SCL脉冲直到本字节结束，然后释放总线进入从机模式。仲裁可以一直进行到所有数据被传输完。这样意味着多主机系统中I^{2}C主机必须监控I^{2}C总线和做相应的处理。图6.27-7描述了Master1数据和Master2数据竞争仲裁。

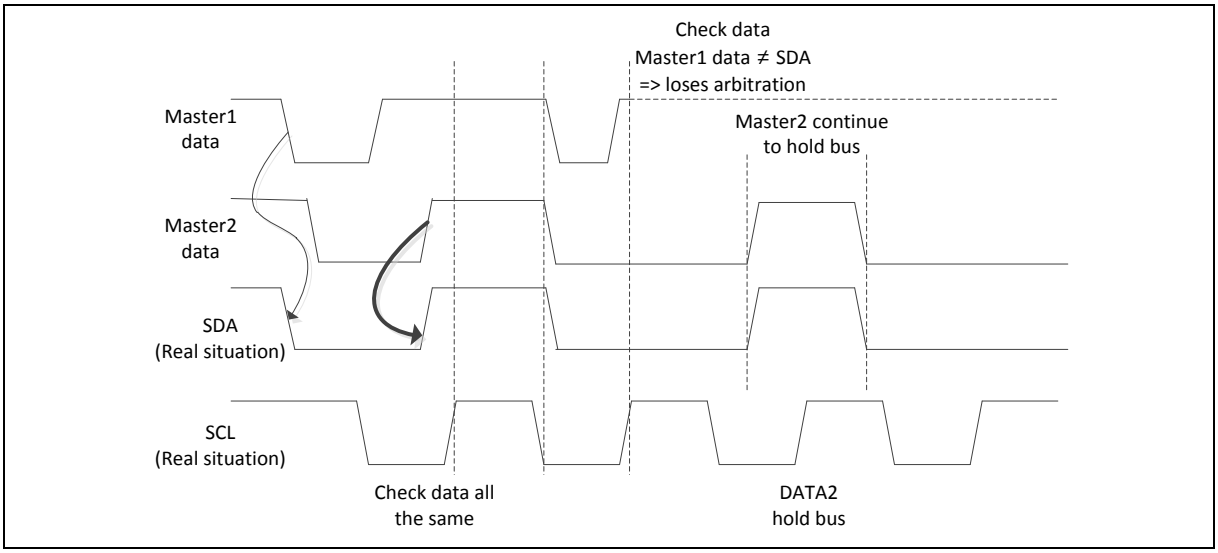


图 6.27-7 仲裁丢失

在地址和数据发送期间，主机发送器会对每个数据位在SCL的上升沿进行检查，如果发送的数据等于在SDA线上读到的数据，主机则继续保持总线，如果不等（发送值=1但是读的值是0），主机失去发送权

利。该事件可由ARBLOIF (USCI_PROTSTS [11])中断标志指示，如果使能ARBLOIEN (UI^{2}C_PROTIEN [4])则会产生协议中断

当发送仲裁丢失，软件必须再次初始化帧，为新的一次主机发送准备首地址字节的起始条件。仲裁也会发送在ACK位。如果主机仲裁丢失并且匹配地址时，那么主机会转变成从机角色

6.27.5.8 传输链

I^{2}C总线协议在仲裁阶段需要一个位内响应，而在从机发送时，传输链的环路延迟可以限制可到达的最大波特率，这很大程度上取决于总线特性（总线负载、模块频率等）。

移位时钟SCL由主设备产生，通过导线输出，然后经过输入级和输入滤波器。现在，可以检测边缘并且相应地生成SDA数据信号。SDA信号通过输出级和到主接收器部分的导线。在那里，它在采样前经过输入级和输入滤波器。

在SCL信号再次改变之前，必须完成这个完整的回路（包括获得稳定信号电平的所有稳定时间）。在计算作为f_{PCLK}和f_{PROT_CLK}函数的波特率时，必须考虑此路径中的延迟。我们建议用户采用f_{PCLK}。

6.27.5.9 非应答和错误条件

在一个非应答(NACKIF (UI^{2}C_PROTSTS [10]))或一个错误(ERRIF(UI^{2}C_PROTSTS [12]))状况下，不会有进一步的发送产生。在依据上次事件配置传输（写TXDAT）之前，软件不能使传输缓冲失效并且禁用传输。

6.27.5.10 I^{2}C协议中断事件

下面协议相关的事件是在I^{2}C模式下产生的，也可以产生协议中断

请注意在寄存器UI^{2}C_PROTSTS中的位不会被硬件自动清除，必须用软件清除这是为了监控新事件的的到来。

- 在帧的正确的位置处接收到起始条件(STARIF (UI^{2}C_PROTSTS [8]))
- 在帧的正确的位置处停止条件被传输(STORIF (UI^{2}C_PROTSTS [9]))
- 主机仲裁丢失(ARBLOIF (UI^{2}C_PROTSTS [11]))
- 从机读请求(SLAREAD (UI^{2}C_PROTSTS [15]))
- 应答接收(ACKIF (UI^{2}C_PROTSTS [13]))
- 非应答接收(NACKIF (UI^{2}C_PROTSTS [10]))
- 起始条件没有在帧期望的位置(ERRIF (UI^{2}C_PROTSTS [12]))
- 停止条件没有在帧期望的位置(ERRIF (UI^{2}C_PROTSTS [12]))

6.27.5.11 操作I^{2}C

操作I^{2}C协议需要考虑下面的问题：

选择I^{2}C 模式

运行并且FUNMODE (UI^{2}C_CTL [2:0]) = 000B期间，建议I^{2}C的配置参数不要改变。在FUNMODE (UI^{2}C_CTL [2:0]) = 000B时I^{2}C控制流程必须完成这样是为了避免意外的输入信号边沿，然后设置FUNMODE (UI^{2}C_CTL [2:0]) = 100B使能I^{2}C模式。

步骤 1.设置 FUNMODE (UI^{2}C_CTL [2:0]) = 000B

步骤 2.设置 FUNMODE (UI^{2}C_CTL [2:0]) = 100B

引脚连接

引脚被用作SDA和SCL时必须由USCI控制器配置成开漏模式来支持I²C总线的线与结构。

注:使能备用输出端口功能的步骤只能是在I²C使能之后，这是为了避免意想不到的脉冲输出。

位时序配置

在标准模式(100 kBit/s) 下模块的最小频率是2MHz,然而在快速模式(400 kBit/s)最小频率是10MHz。此外，被用于消除脉冲的数字滤波阶段达到50ns,滤波器的频率需有20MHz。如果另外一个I²C参与拉低SCL，在SCL最大1/f_{PROT_CLK} 为高的阶段可能会有一个不确定性。请注意SCL最大频率是SAMP_CLK/2并且必须设置SPCLKSEL (UI²C_BRGEN [3:2])为0选择f_{SAMP_CLK} = f_{DIV_CLK}

数据格式配置

数据格式必须配置为8数据位(DWIDTH (UI²C_LINECTL [11:8]) = 8)并且MSB优先(LSB (UI²C_LINECTL [0]) = 0).最后USCI_LINECTL必须设置为0x800.

控制流程

片上I²C端口支持三种模式主机模式、从机模式和广播模式。

应用中，I²C端口可以作为主机或从机。在从机模式，I²C端口硬件会查找自身从机地址和广播呼叫地址，如果这两个地址的任一个被检测到，并且从机想要从主机接收或向主机发送数据(通过设置AA位)，应答脉冲将会在第9个时钟发出，此时，如果中断使能，则主机和从机设备上都会发生一次中断请求。在微控制器想要成为总线主机时，在进入主机模式之前，硬件需等待到总线空闲，以保证合理的从机动作不会被打断。在主机模式下，如果总线仲裁丢失，I²C端口立即切换到从机模式，并可以在同一次串行传输过程中检测自身。

为控制I²C总线的各种模式传输，用户需要按照寄存器UI²C_PROTSTS的当前状态码来设置UI²C_PROTCTL，UI²C_PROTIEN，TXDAT寄存器。换句话说，每个I²C总线动作都要检查UI²C_PROTSTS寄存器的当前状态，然后再设置UI²C_PROTCTL，UI²C_PROTIEN，TXDAT寄存器执行总线动作。最后，通过UI²C_PROTSTS检查响应状态。

在中断标志清除后，寄存器UI²C_PROTCTL的这些位STA, STO 和 AA 用来控制I2C硬件的下一个状态。当完成一个新的动作，UI²C_PROTSTS的状态将被更新。如果I²C中断控制位UI²C_PROTIEN被设置，新状态对应的动作或者软件将在中断服务程序中被执行。

图6.27-8显示当前I²C STARIF (UI²C_PROTSTS [8])被硬件置1，然后设置TXDAT = SLA+W (从机地址 + 写位), (PTRG, STA, STO, AA) = (1, 0, 0, x)来发送地址到I²C总线，写1到STARIF (UI²C_PROTSTS [8])清标志。如果在总线上一个从机地址匹配并且应答ACK，ACKIF (UI²C_PROTSTS [13])会被更新。

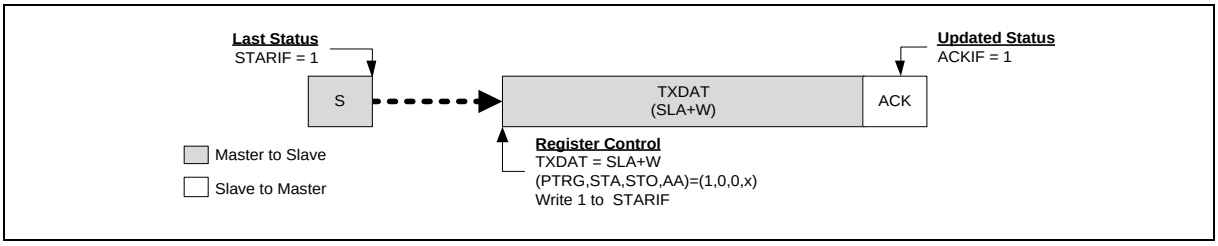


图 6.27-8 根据当前 I²C 状态控制 I²C 总线

I²C 总线上的数据传输

图6.27-9表示7位地址情况下主机向从机传输数据。主机发出一个7位地址和1位写指示,表示主机想要传送数据给从机。从机回复应答给主机之后，主机继续传输数据。

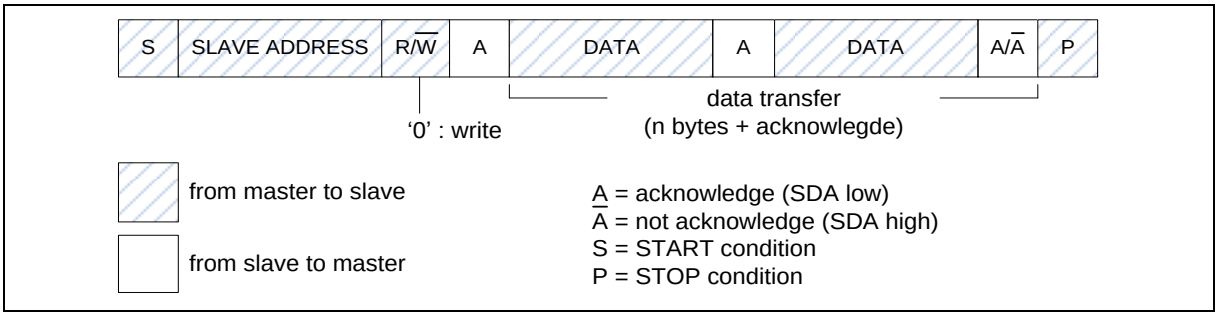


图 6.27-97 位地址格式主机向从机传输数据

图6.27-10表示7位地址情况下主机向从机读取数据。主机发7位地址寻址和1位读指示，表示主机要向从机读取数据，从机返回应答给主机后，就开始给主机传输数据。

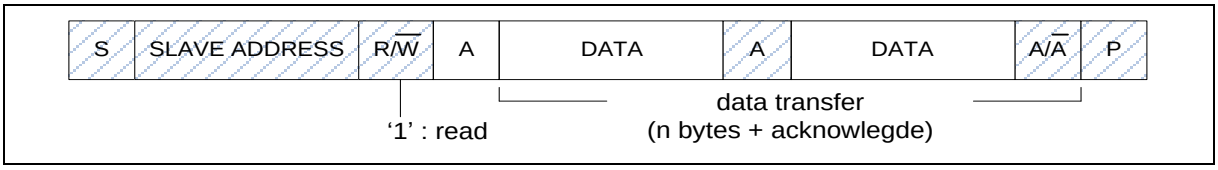


图 6.27-107 位地址情况下主机向从机读取数据

图6.27-11表示10位地址情况下主机向从机传输数据。主机发送10位地址。首字节是有10位地址指示(5'b11110)和2位地址再加上写指示，第二字节是剩下的8位地址。在第二字节之后，主机继续传输数据。注意7位和10位地址设备可以工作在同一总线上。

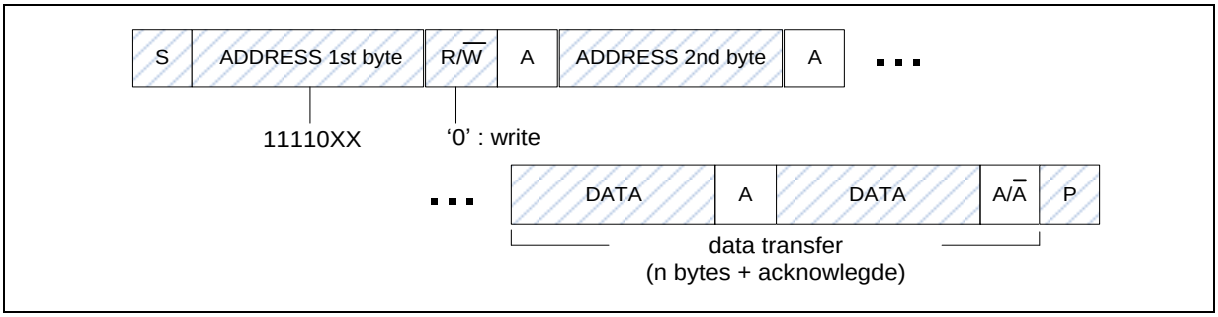


图 6.27-1110 位地址格式主机向从机传输数据

图6.27-12表示10位地址情况下主机向从机读取数据。首先主机发送10位地址给从机，在主机传输带有读指示的第一个字节之后。在带有读指示的第一个字节之后，从机就开始给主机传输数据。

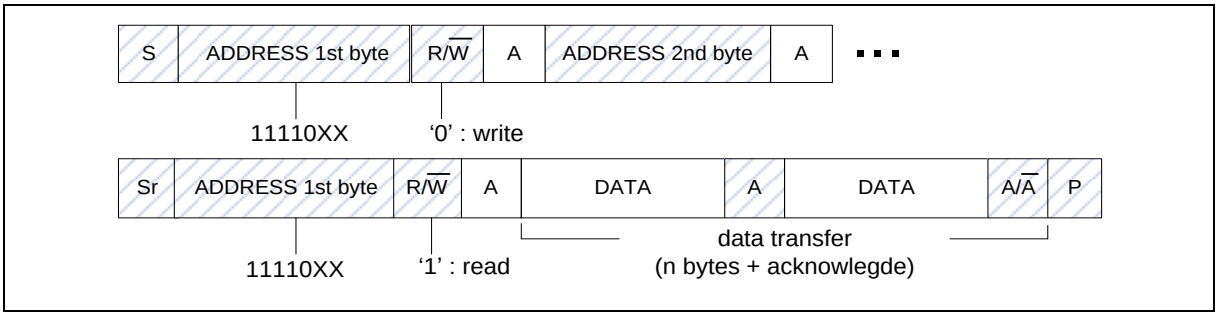


图 6.27-12 10 位地址格式主机向从机读取数据

主机模式

图6.27-13和图6.27-14是I²C主机模式所有可能的协议展示。用户需要遵循恰当的流程来实现I²C协议。

换句话说，用户可以发送一个起始信号到总线，I²C总线将设置成主机传输(MT)模式(图6.27-13)，或主机接收(MR)模式(图6.27-14)。起始信号设置成功后新的状态被设置STARIF (UI²C_PROTSTS [8])。起始信号后，用户可以发送从机地址，读/写位，数据和重复起始，停止来执行I²C协议。

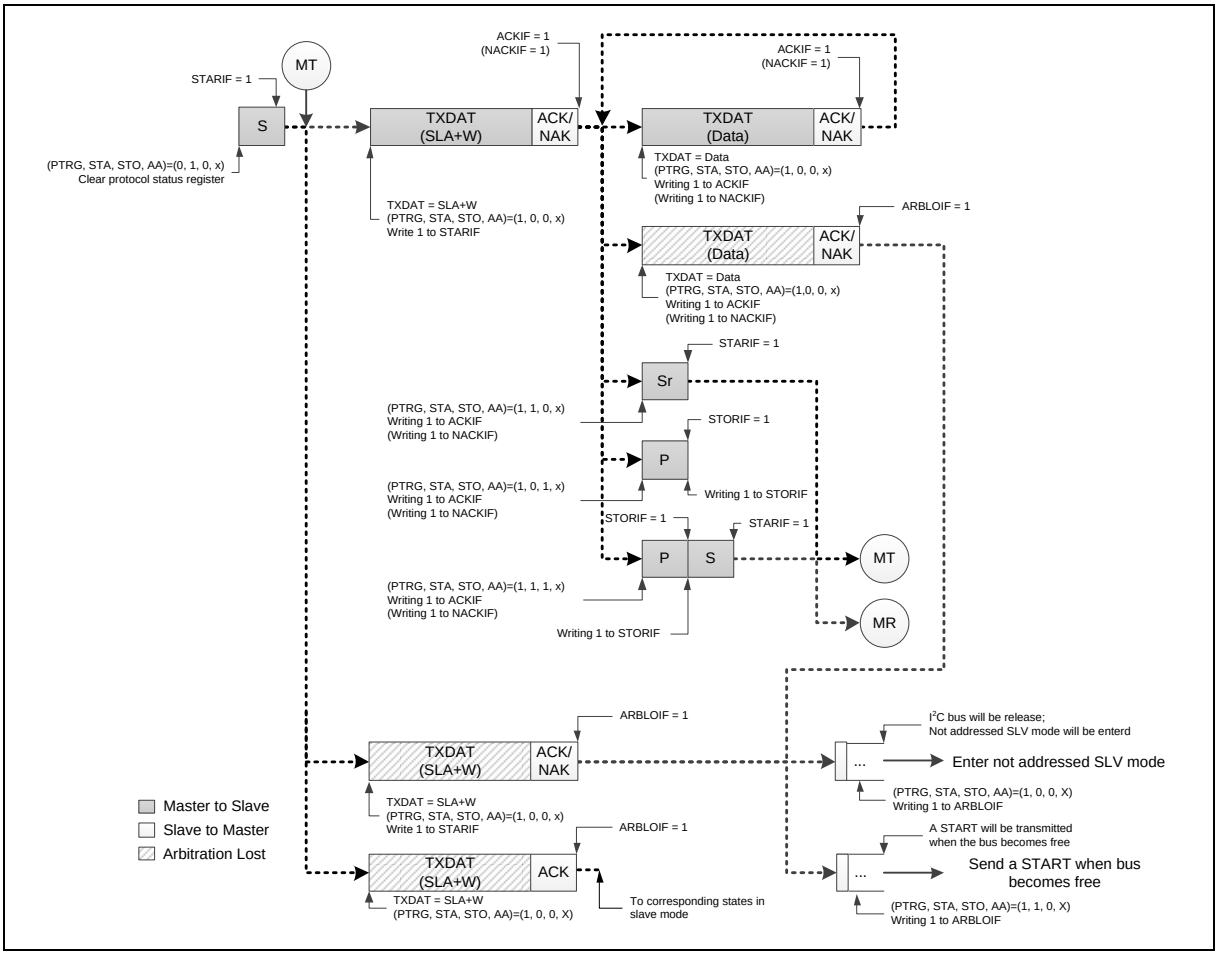


图 6.27-137 位地址主机发送模式控制流程

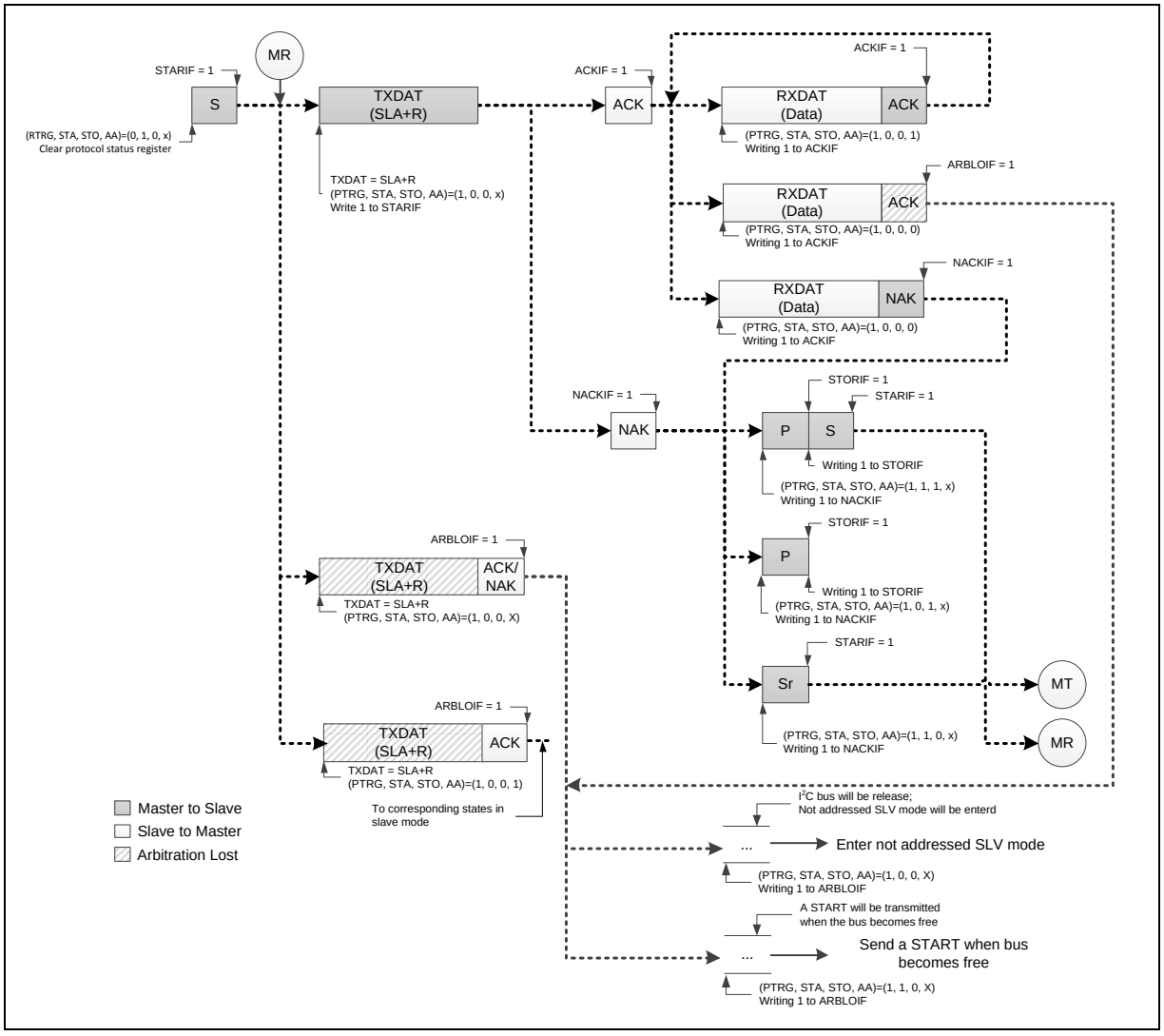


图 6.27-14 7 位地址主机接收模式控制流程

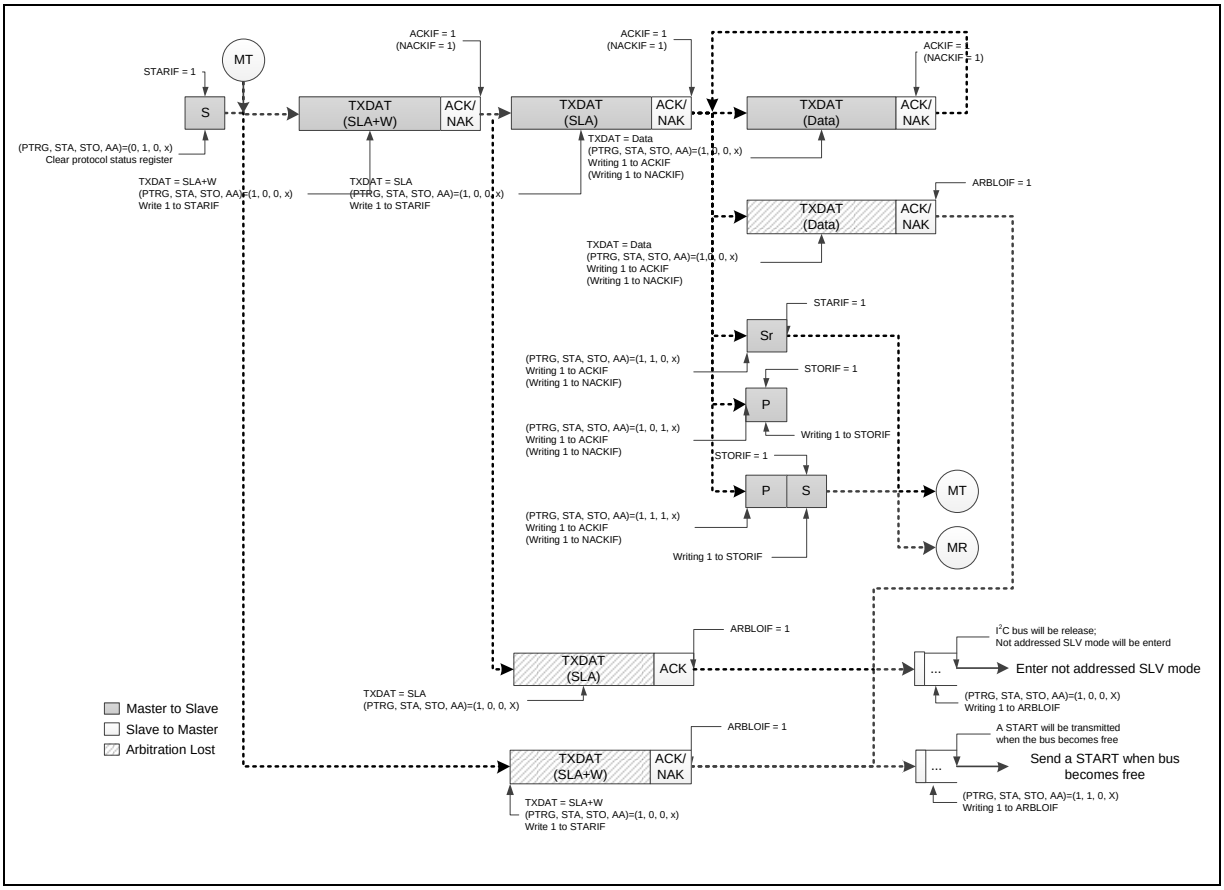


图 6.27-15 10 位地址主机发送模式控制流程

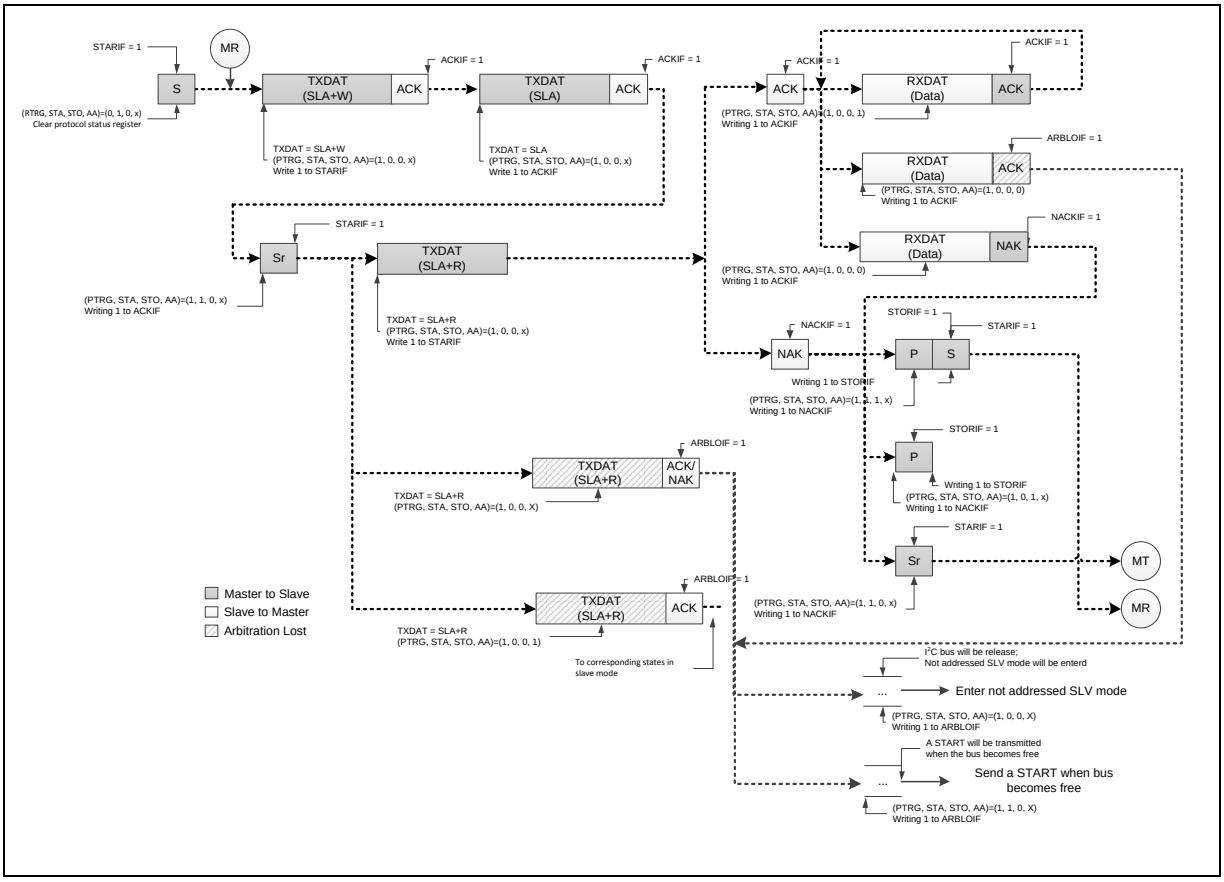


图 6.27-16 10 位地址接收模式控制流程

如果I²C在主机模式并且仲裁丢失，ARBLOIF (UI²C_PROTSTS [11])位被置位。当总线空闲时，用户可以写1到ARBLOIF (UI²C_PROTSTS [11])并且设置(PTRG, STA, STO, AA) = (1, 1, 0, X)发送起始信号来重新开始主机操作。否则，用户可以写1到ARBLOIF (UI²C_PROTSTS [11])并且设置(PTRG, STA, STO, AA) = (1, 0, 0, X)来释放总线，并进入无地址从机模式。

从机模式

当复位后，I²C不会被寻址，并且不会识别I²C总线上的地址。用户可以通过UI²C__DEVADDRn设置从机地址和设置(PTRG, STA, STO, AA) = (1, 0, 0, 1)来让I²C识别主机发送的地址。图6.27-17所示为I²C从机模式的所有可能的流程。用户需要遵循（图6.27-17）的流程来实现他们的I²C协议。

如果在主机模式总线仲裁丢失，I²C端口立即切换到从机模式，并且在同一串行传输中识别自有的从机地址。如果在仲裁丢失后识别到地址是SLA+W（主机想写数据到从机或是SLA+R（主机向从机读数据），ARBLOIF会被置1。

注: I²C通信期间，在从机模式下，当对PTRG (UI²C_PROTCTL [5])标志写‘1’时，SCL 时钟将被释放。

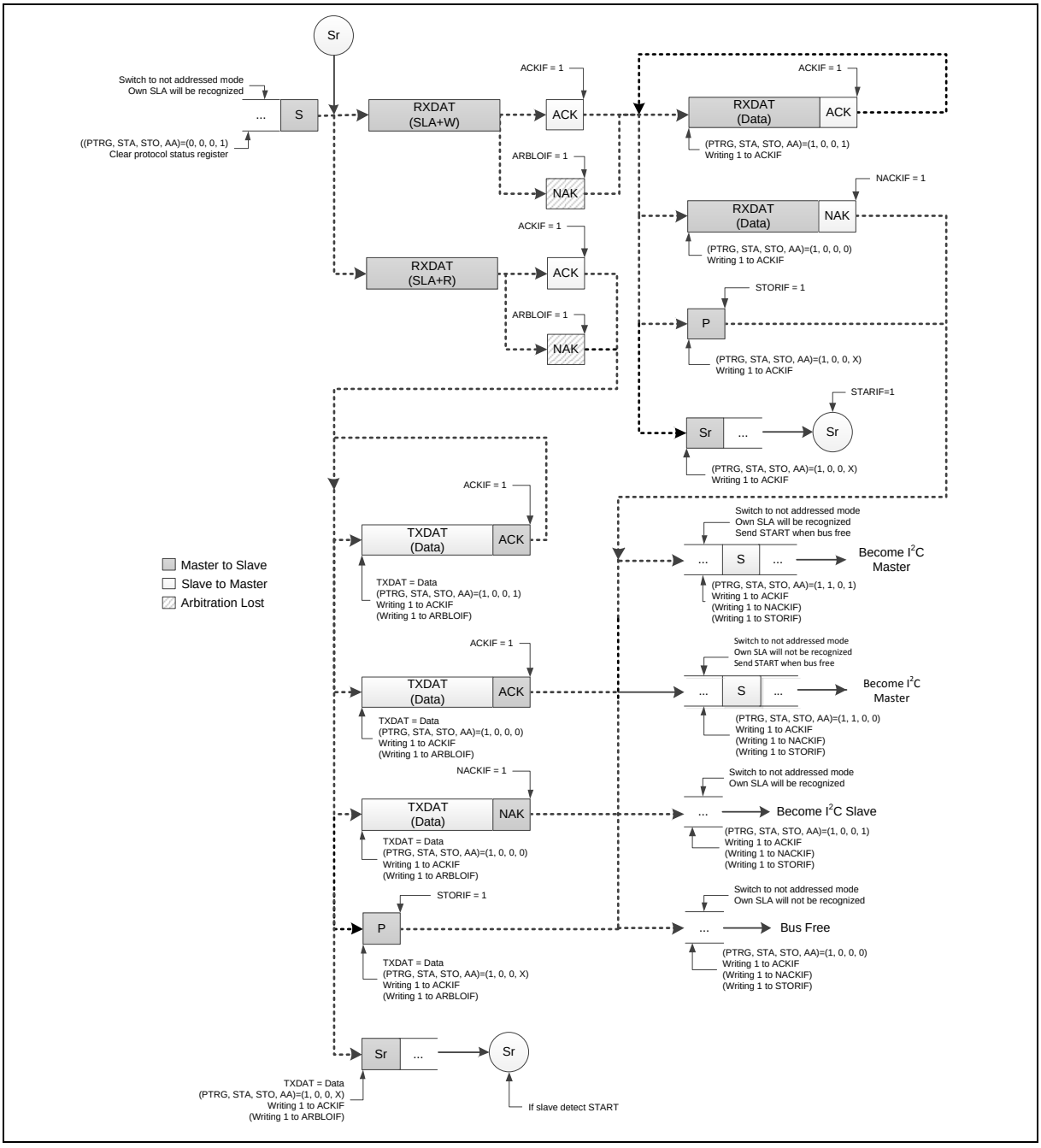


图 6.27-177 位地址从机模式控制流程

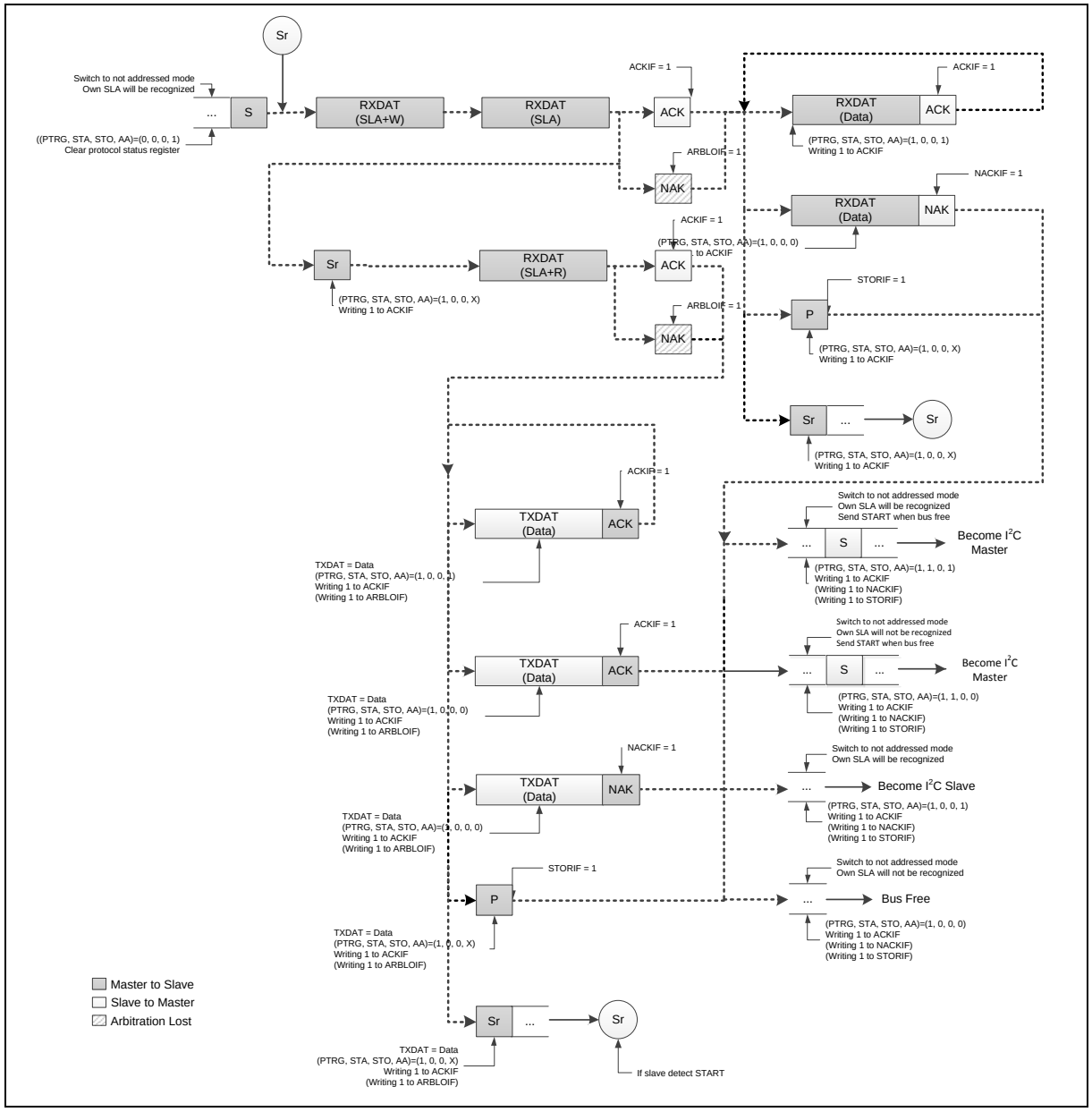


图 6.27-18 10 位地址从机控制流程

如果I²C在可寻址从机模式发送和接收数据时，收到停止或重复起始信号，STORIF (UI²C_PROTSTS [9])或STARIF (UI²C_PROTSTS [8])会被置位。当STARIF (UI²C_PROTSTS [8])被设置时，用户可以遵循如上图NACKIF (UI²C_PROTSTS [10])的操作

注:从机获得NACKIF (UI²C_PROTSTS [10])和起始/停止符号包括STARIF(UI²C_PROTSTS [8]) 和STORIF (UI²C_PROTSTS [9])中断状态标志后,从机可以切换到无地址模式，自身SLA不会被辨识。如果进入这种状态，从机不再接收主机任何信号或地址。在这种状态，I²C需要设置FUNMODE (UI²C_CTL [2:0]) = 000B离开这个状态

广播呼叫模式(GC)

如果GCFUNC位(UI²C_PROTCTL [0])被设置，I²C端口硬件将响应广播呼叫地址(0x00)。用户可以通过清GC位来禁止广播呼叫功能。当I²C在从机模式时且GC 位被设置，可以接收主机地址(0x00)的广播呼

叫，将遵循广播模式状态。

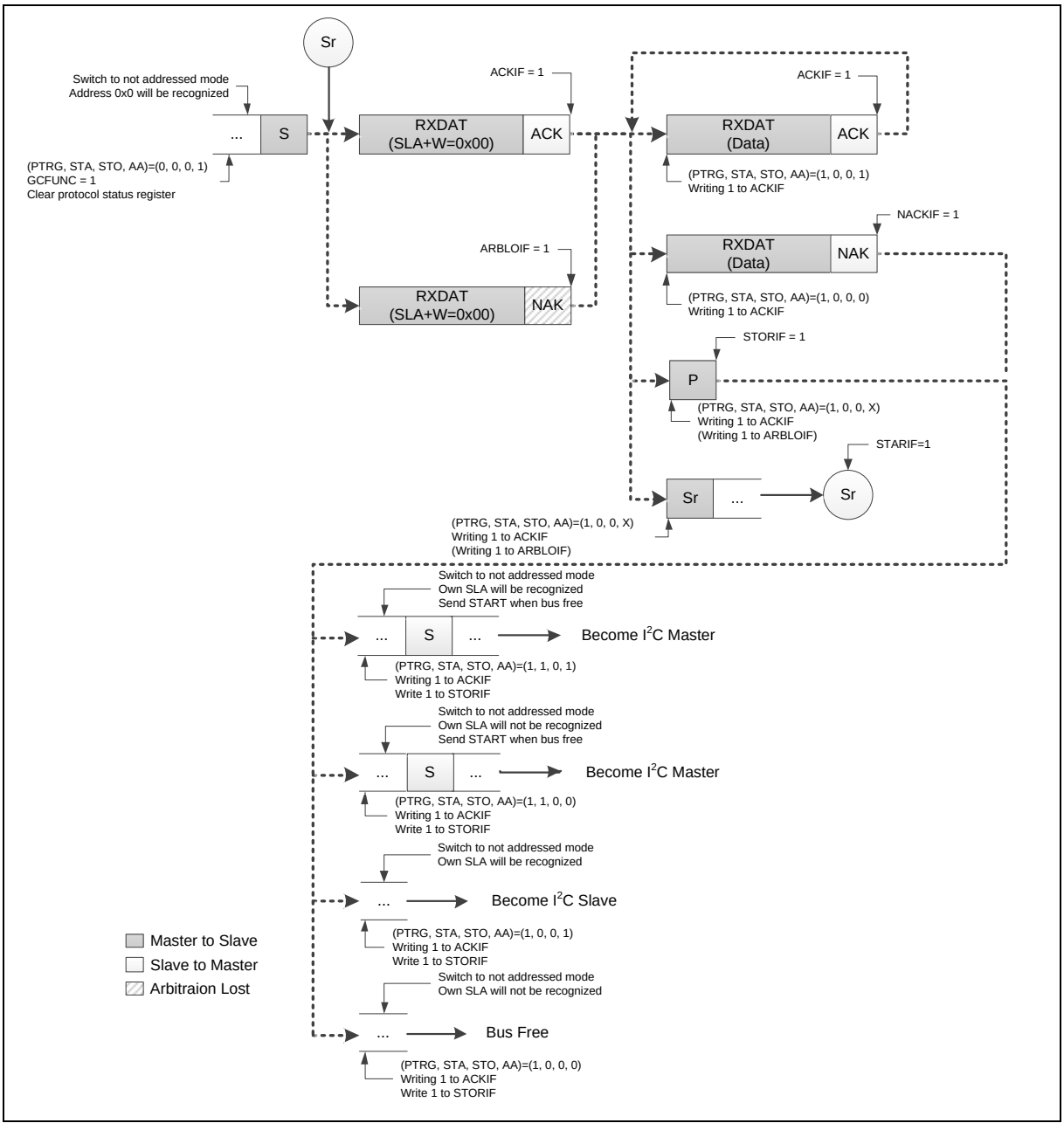


图 6.27-197 位地址的 GC 模式

如果I²C在广播呼叫模式接收数据时收到停止或重复起始信号，STORIF(UI²C_PROTSTS [9])或STARIF (UI²C_PROTSTS [8])会被置位。当得知STORIF(UI²C_PROTSTS[9])或STARIF (UI²C_PROTSTS [8]) 被置位时，用户可以遵循如上图NACKIF (UI²C_PROTSTS [10])的流程处理。

注:从机获得NACKIF(UI²C_PROTSTS [10])和起始/停止符号包括STARIF(UI²C_PROTSTS[8])和STORIF(UI²C_PROTSTS [9])中断状态标志后，从机可以切换到无地址模式，自身SLA不会被辨识。如果进入这种状态，从机不再接收主机任何信号或地址。在这种状态，I²C需要设置FUNMODE (USCI_CTL[2:0]) = 000B离开这个状态。

协议功能描述

监视器模式

当I²C进入监视器模式，在每次帧接收后设备总是返回NACK给主机即使是地址匹配。此外，设备会存储接收到的任何数据包括地址、命令码和数据。

监视器模式下的中断

当MONEN (UI²C_PROTCTL [9])被设置，所有中断处理与正常一样。注意当有初始化START时会发生第一个中断，这个与I²C从机不一样，但是其它中断是一样的。

地址匹配侦测之后，中断会在每次数据如同从机模式控制流程一样被接收之后或是再模块信任的每个字节作为从机读传输被传输之后产生。在第二个情况中，数据寄存器实际上会包含总线上一些其它被主机寻址的从机发送的数据。如果用户想监控其它设备，用户可以设置地址掩码和监视器。

如果监视器没有时间响应中断，当SCLOUTEN(UI²C_PROTCTL [8])被置1时SCL信号会被拉低。当SCLOUTEN(UI²C_PROTCTL [8])被置1时，用户必须设置PTRG(UI²C_PROTCTL [5])释放总线。如果SCLOUTEN(UI²C_PROTCTL [8])没有被置1，用户就不需要置PTRG(UI²C_PROTCTL [5])为1。

当设备地址匹配时，但是设备会响应NACK，该地址会被接收到缓冲中并且会产生NACK中断。

接下来的所有中断，处理器可能需要读数据寄存器来看总线上实际传输的是什么。

监视器模式下仲裁丢失

在监视器模式下，I²C模块不会响应由总线主机产生的ACK信息请求。总线上的其它从机会去响应。软件应该知道在模块处在监视器模式下并且不能检测到的任何仲裁丢失情况。

可编程的建立和保持时间

为了保证一个正确数据的建立和保持时间，则时序必须可以配置。通过编程HTCTL (UI²C_TMCTL[24:16])来配置保持时间，通过编程STCTL (UI²C_TMCTL[8:0])来配置建立时间。

延时时序参考外设时钟(PCLK)当设备拉住主机时钟时，建立和保持时间的配置值不会被影响。

用户需要集中精力在建立和保持时间配置的极限上，时序设置必须遵从I²C协议。一旦建立时间配置大于设计极限，那就意味着如果设置的建立时间让SCL输出小于三个PCLK，由于SCL采样为三次那么I²C控制器就不能正常工作。一旦保持时间配置大于I²C时钟的极限，I2C将会发送总线错误。在设置时序前建议用户结合传输速率和协议计算好合理的时序。表6.27-1显示了I²C传输速率与PCLK之间的关系，表格呈现的数是一个时钟间隙包含多少PCLK。建立和保持时间的配置哪怕在我们设计中编程一些极限值，但是用户必须要遵从I²C的标准协议。

I²C 传输速率 PCLK	100k	200k	400k	800k	1200k
12 MHz	120	60	30	15	10
24 MHz	240	120	60	30	20
48 MHz	480	240	120	60	40
72 MHz	720	360	180	90	60

表 6.27-1 I²C 传输速率与 PCLK 之间的关系

对于建立时间错误调整范例，假设一个SCL周期包含10个PCLKs，并将STCTL (UI²C_TMCTL[8:0])设置为3，拉伸3个PCLKs以进行设置时间设置。设置时间设置限制：

ST_{limit} = (UI²C_BRGEN[25:16]+1) - 6.

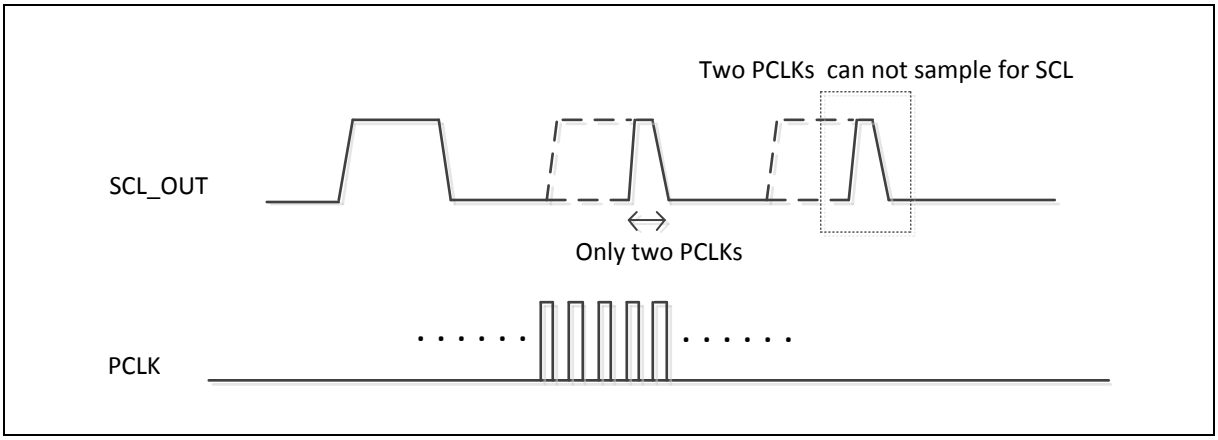


图 6.27-20 建立时间错误调整

对于保持时间错误调整范例，我们使用I²C传输速率=1200K和PCLK=72MHz，SCL的高电平/低电平空隙为60PCLK。当我设置HTCTL (UI²C_TMCTL[24:16])为63和STCTL (UI²C_TMCTL[8:0])为0时，然后SDA输出延时将会越过SCL为高的空隙，以至产生总线错误。保持时间最大值的是设定：

HT_{limit} = (UI²C_BRGEN[25:16]+1) - 9

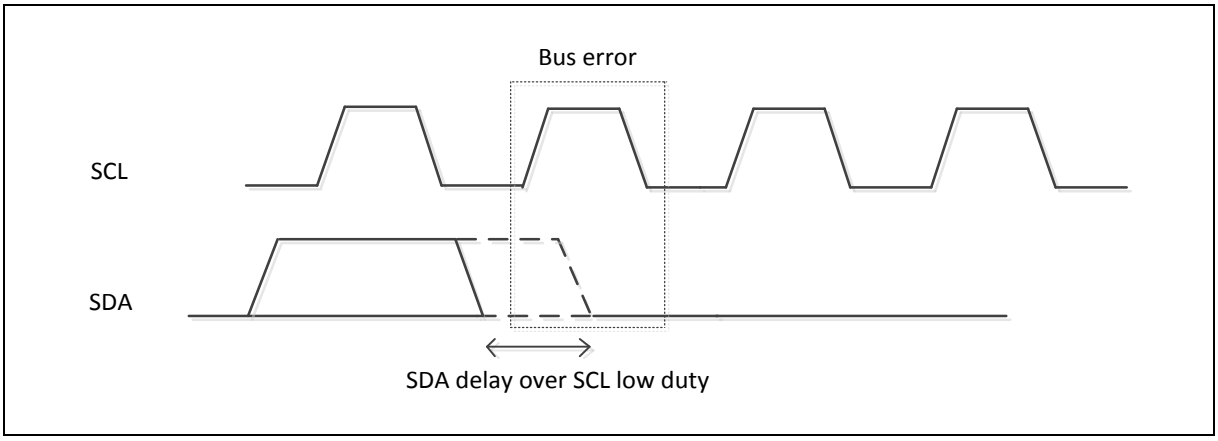


图 6.27-21 保持时间错误调整

I²C 超时功能

系统提供一个10位超时的计数器TOCNT (UI²C_PROTCTL [25:16])来处理当I²C总线锁死时的情况。当超时计数器计数功能使能后，计数器开始上计数直至等于TOCNT (UI²C_PROTCTL [25:16])并向CPU产生I²C中断或者清除TOIEN (UI²C_PROTIEN [0])为0停止计数或是设置所有的I²C中断信号(ACKIF, ERRIF, ARBLOIF, NACKIF, STORIF, STARIF)。用户需要写1清TOIF(UI²C_PROTSTS[5])为0。当超时计数器使能，对TOIF 写1会使计数器复位，TOIF被清除后重新开始计数。对于超时计数器TOCNT (UI²C_PROTCTL [25:16])、]). TTOCNT = (TOCNT (UI²C_PROTCTL [25:16]) +1) x32 (5-bit) x TPCLK 请参考图6.27-22。请注意时间计数器时钟源TMCNTSRC (UI²C_ BRGEN [5])必须设置为0。

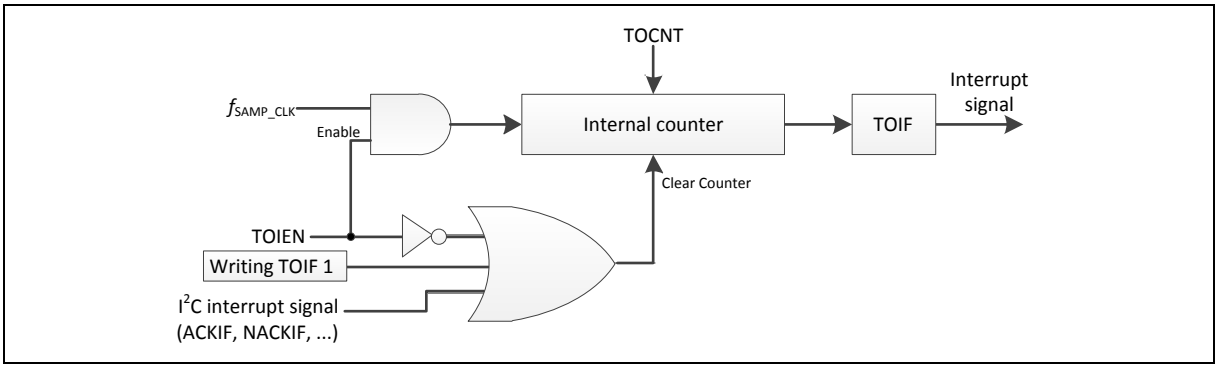


图 6.27-22 I²C 超时计数模式框图

唤醒功能

当芯片进入掉电模式并且WKEN (WKCTL[0])被设置为1时，其它I²C主机通过寻址我们I²C设备唤醒我们芯片，在进入睡眠前必须配置相关设置。地址匹配帧的ACK位阶段是在掉电情况下完成的。当地址与设备地址匹配时并且ACK阶段过后，控制器会拉低SCL。直到用户清除标志位SCL释放。如果SCL的频率是低速并且系统从地址匹配帧中唤醒，用户需要检查该位来确认该帧传输已经完成然后去处理唤醒的事务。因此，当芯片被设备地址寄存器(UI²C_DEVADDRn)其中之一地址匹配唤醒时，用户需要检查WKAKDONE (UI²C_PROTSTS [16])是否被置1来确认地址字节已完成。WKAKDONE位表明在掉电时候ACK阶段已完成。当地址与设备地址匹配时并且ACK阶段过后，控制器会拉低SCL，直到WKAKDONE被用户清除。如果SCL是低频速率并且系统被地址匹配帧唤醒，用户需要检查WKAKDONE来确认该帧已完成传输然后去做唤醒后的事务。请注意用户必须在清除WKAKDONE为0之后清除WKUPIF。

WRSTSWK (UI²C_PROTSTS [17])位用来记录在地址匹配唤醒帧中的读/写命令。在系统被地址匹配帧唤醒后，用户读该位的状态来准备下次传输数据(WRSTSWK = 0)或是等待到来的数据(WRSTSWK = 1)及时地被存储。

当系统被其他的I2C主机设备唤醒时，WKF(UI²C_WKSTS[0])被置位表示该事件发生。用户需要写“1”来清除此位。

EEPROM 随机读取范例

通过下面的步骤来配置USCI0_I2C相关寄存器，来使用I²C从EEPROM读取数据。

1. 设置USCI0_I2C多功能引脚为SCL和SDA引脚。请参考基本配置章节进行多功能配置。
2. 使能USCI0 APB时钟。请参考基本配置章节进行多功能配置。
3. 通过设置USCI0RST=1来复位USCI控制器，然后设置USCI0RST=0使USCI控制器为普通模式。请参考基本配置章节复位控制器配置。
4. 在寄存器“UI²C_CTL”中通过设置FUNMODE =100来使能USCI0_I2C控制器。
5. 在寄存器“UI²C_BRGEN”中设置USCI0_I2C时钟分频寄存器值来获取USCI0_I2C时钟频率。
6. 使能系统I²C0 IRQ 在系统“NVIC”控制寄存器。
7. 在寄存器“UI²C_PROTIEN”中设置ACKIEN, ERRIEN, ARBLOIEN, NACKIEN, STORIEN, STARIEN和TOIEN来使能I²C中断。
8. 设置USCI地址寄存器“UI²C_ADDR0 ~ UI²C_ADDR1”。

随机读操作时访问EEPROM的方法之一。此方法允许直接访问EEPROM空间的任何地址。图6.27-23展示了EEPROM随机读操作。

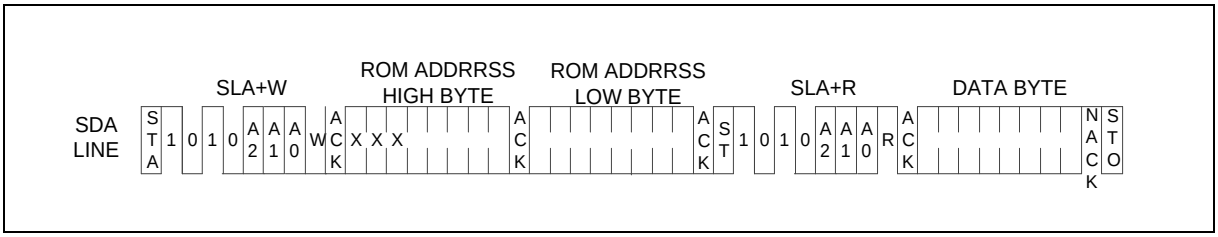


图 6.27-23 EEPROM 随机读取

图6.27-24展示了怎么使用I²C控制器执行EEPROM随机读取操作协议

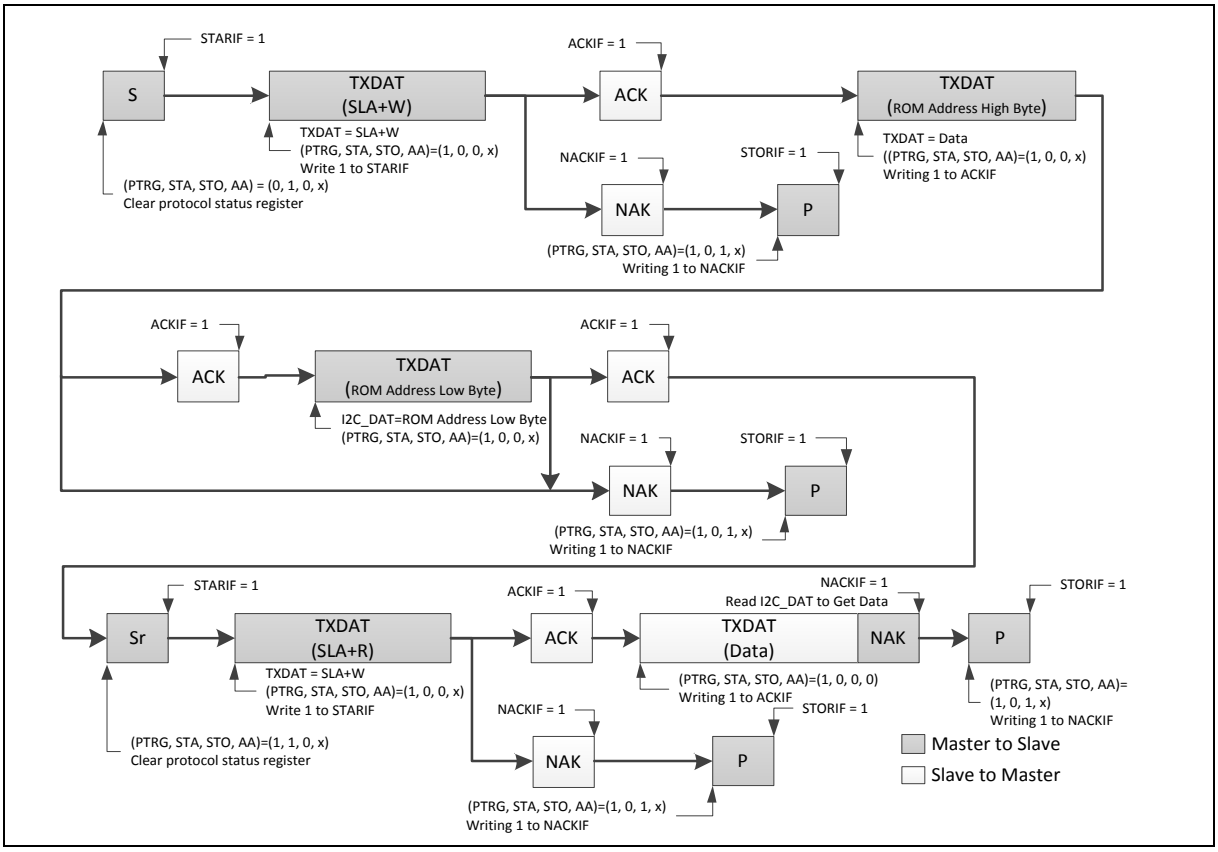


图 6.27-24 EEPROM 随机读取协议

I²C控制器作为主机发送起始信号到总线，然后发送SLA+W (从机地址 + 写位)到EERPOM，跟着由两个字节数据地址来设置EEPROM被读的地址。最后，重复起始信号跟着SLA+R被发送来向EEPROM 读取数据。

6.27.6 寄存器映射

R:只读, **W:**只写, **R/W:** 读写

寄存器	偏移地址	R/W	描述	复位值
UI²C_I2C 基地址： **UI²Cn_BA = 0x400D_0000 + (0x1000 * n)** **n= 0, 1**				
UI²C_CTL	UI²Cn_BA+0x00	R/W	USCI控制寄存器	0x0000_0000
UI²C_BRGEN	UI²Cn_BA+0x08	R/W	USCI波特率发生器寄存器	0x0000_3C00
UI²C_LINECTL	UI²Cn_BA+0x2C	R/W	USCI线控制寄存器	0x0000_0000
UI²C_TXDAT	UI²Cn_BA+0x30	W	USCI发送数据寄存器	0x0000_0000
UI²C_RXDAT	UI²Cn_BA+0x34	R	USCI接收数据寄存器	0x0000_0000
UI²C_DEVADDR0	UI²Cn_BA+0x44	R/W	USCI设备地址寄存器0	0x0000_0000
UI²C_DEVADDR1	UI²Cn_BA+0x48	R/W	USCI设备地址寄存器1	0x0000_0000
UI²C_ADDRMSK0	UI²Cn_BA+0x4C	R/W	USCI设备地址掩码寄存器0	0x0000_0000
UI²C_ADDRMSK1	UI²Cn_BA+0x50	R/W	USCI设备地址掩码寄存器1	0x0000_0000
UI²C_WKCTL	UI²Cn_BA+0x54	R/W	USCI唤醒控制寄存器	0x0000_0000
UI²C_WKSTS	UI²Cn_BA+0x58	R/W	USCI唤醒状态寄存器	0x0000_0000
UI²C_PROTCTL	UI²Cn_BA+0x5C	R/W	USCI协议控制寄存器	0x0000_0000
UI²C_PROTIEN	UI²Cn_BA+0x60	R/W	USCI协议中断使能寄存器	0x0000_0000
UI²C_PROTSTS	UI²Cn_BA+0x64	R/W	USCI协议中断状态寄存器	0x0000_0000
UI²C_TMCTL	UI²Cn_BA+0x8C	R/W	I²C时序配置寄存器	0x0000_0000

6.27.7 寄存器描述

USCI控制寄存器 (UI²C_CTL)

寄存器	偏移地址	R/W	描述	复位值
UI²C_CTL	UI²Cn_BA+0x00	R/W	USCI控制寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved					**FUNMODE**		

位	描述	
[31:3]	**Reserved**	保留
[2:0]	**FUNMODE**	**功能模式** 该域是选择USCI控制器的协议。选择一个不可用的协议或保留的组合会禁用USCI。当切换两个协议时，USCI在选择新的协议之前时禁用的。同时，当用户写000到FUNMODE将会复位USCI。 000 = USCI禁用，所以协议相关的状态机被设为空闲状态。 001 = SPI协议选择 010 = UART协议选择 100 = I²C协议选择 **注:**其他位组合保留

USCI波特率发生器寄存器 (UI^{2}C_BRGEN)

寄存器	偏移地址	R/W	描述	复位值
UI^{2}C_BRGEN	UI^{2}Cn_BA+0x08	R/W	USCI波特率发生器寄存器	0x0000_3C00

31	30	29	28	27	26	25	24
Reserved						CLKDIV	
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
CLKDIV							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved	DSCNT					PDSCNT	
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved		TMCNTSRC	TMCNTEN	SPCLKSEL		PTCLKSEL	RCLKSEL

位	描述	
[31:26]	**Reserved**	保留
[25:16]	**CLKDIV**	**时钟分频器** 该域是定义协议时钟频率f_{PROT_CLK}和时钟分频器频率之间的比率 f_{DIV_CLK} (f_{DIV_CLK}= f_{PROT_CLK}/ (CLKDIV+1))
[15]	**Reserved**	保留
[14:10]	**DSCNT**	**采样计数器的分比** 该域定义采样时钟f_{SAMP_CLK}的分频比。 分频f_{DS_CNT} = f_{PDS_CNT} / (DSCNT+1) **注:**在I^{2}C模式下，DSCNT最大值是0xF，建议设置4以上来保证接收数据的采样时正确的值。
[9:8]	**PDSCNT**	**采样计数器预分频** 该域定义来自采样时钟f_{SAMP_CLK}的分频比 分频f_{PDS_CNT} = f_{SAMP_CLK} / (PDSCNT+1)
[7:6]	**Reserved**	保留
[5]	**TMCNTSRC**	**时序测量计数器时钟源选择** 0 = 测量计数来自f_{PROT_CLK}. 1 = 测量计数来自f_{DIV_CLK}.
[4]	**TMCNTEN**	**时序测量计数器使能位** 该位使能10位时序测量计数器 0 = 测量计数来自禁止 1 = 测量计数来自使能
[3:2]	**SPCLKSEL**	**采样时钟源选择**

		该域用来为协议采样时钟源(f_{SAMP_CLK}) 选择时钟源 00 = f_{SAMP_CLK} = f_{DIV_CLK}. 01 = f_{SAMP_CLK} = f_{PROT_CLK}. 10 = f_{SAMP_CLK} = f_{SCLK}. 11 = f_{SAMP_CLK} = f_{REF_CLK}.
[1]	**PTCLKSEL**	**协议时钟源选择** 该位为协议时钟选择信号源(f_{PROT_CLK}) 0 = 参考时钟f_{REF_CLK}. 1 = f_{REF_CLK2} (频率为f_{REF_CLK}的一半)
[0]	**RCLKSEL**	**参考时钟源选择** 该位是选择参考时钟(f_{REF_CLK})源信号 0 = 外设设备时钟f_{PCLK}. 1 = 保留

USCI线控制寄存器 (UI^{2}C_LINECTL)

寄存器	偏移地址	R/W	描述	复位值
UI^{2}C_LINECTL	UI^{2}Cn_BA+0x2C	R/W	USCI线控制寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved				**DWIDTH**			
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved							**LSB**

位	描述	
[31:12]	**Reserved**	保留
[11:8]	**DWIDTH**	**传输的字长** 该域定义发送和接收的数据字长。在数据缓冲中数据总是右对齐。USCI支持4到16位的字长 0x0: 在数据字中包含16位，位于[15:0]这些位中 0x1: 保留 0x2: 保留 0x3: 保留 0x4: 在数据字中包含4位，位于[3:0]这些位中 0x5: 在数据字中包含5位，位于[4:0]这些位中 ... 0xF:在数据字中包含15位，位于[14:0]这些位中
[7:1]	**Reserved**	保留
[0]	**LSB**	**LSB优先传输选择** 0 =取决于DWIDTH设置的接收/发送数据缓冲的MSB先发送/接收 1 =数据缓冲的位0及LSB被先发送/接收

USCI发送数据寄存器 (UI^{2}C_TXDAT)

寄存器	偏移地址	R/W	描述	复位值
UI^{2}C_TXDAT	UI^{2}Cn_BA+0x30	W	USCI发送数据寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
TXDAT							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
TXDAT							

位	描述	
[31:16]	**Reserved**	保留
[15:0]	**TXDAT**	**发送数据** 软件可以写16位发送数据到该域用来发送

USCI接收数据寄存器 (UI^{2}C_RXDAT)

寄存器	偏移地址	R/W	描述	复位值
UI^{2}C_RXDAT	UI^{2}Cn_BA+0x34	R	USCI接收数据寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
RXDAT							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
RXDAT							

位	描述	
[31:16]	**Reserved**	保留
[15:0]	**RXDAT**	**接收的数据** 该域值监控存储在接收数据缓冲的接收到的数据 **注:**在I^{2}C协议下，RXDAT[12:8]表示不同的传输条件，该条件定义在I^{2}C中

USCI设备地址寄存器(UI^{2}C_DEVADDR)

寄存器	偏移地址	R/W	描述	复位值
UI^{2}C_DEVADDR0	UI^{2}Cn_BA+0x44	R/W	USCI设备地址寄存器0	0x0000_0000
UI^{2}C_DEVADDR1	UI^{2}Cn_BA+0x48	R/W	USCI设备地址寄存器1	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved						DEVADDR	
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
DEVADDR							

位	描述	
[31:10]	**Reserved**	保留
[9:0]	**DEVADDR**	**设备地址** 在I^{2}C协议下，该位域包含可编程的从机地址。如果第一个接收到地址字节是1111 0AAX$_B$，AA位会与 DEVADDR[9:8]比较检查地址地址匹配，X是读/写位。第二个地址字节会和DEVADDR[7:0]比较。 **注 1:**当 I^{2}C工作在7位地址模式时， DEVADDR [9:7]必须设置为3'b000。 **注 2:**当软件设置10'h000时，地址不能被使用。

USCI设备地址掩码寄存器(UI^{2}C_ADDRMSK) – 仅针对I^{2}C

寄存器	偏移地址	R/W	描述	复位值
UI^{2}C_ADDRMSK0	UI^{2}Cn_BA+0x4C	R/W	USCI设备地址掩码寄存器0	0x0000_0000
UI^{2}C_ADDRMSK1	UI^{2}Cn_BA+0x50	R/W	USCI设备地址掩码寄存器1	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved						ADDRMSK	
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
ADDRMSK							

位	描述	
[31:10]	**Reserved**	保留
[9:0]	**ADDRMSK**	**USCI设备地址掩码** 0 = 掩码禁用 (接收的相关地址位必须完全与地址寄存器一致) 1 = 使能掩码 (不用关心接收相关地址未) 2组地址掩码寄存器使得USCI支持多地址识别。当在地址掩码寄存器中的位被置1，意味着不用关心接收的相关地址位。如果设置为0，意味着接收的相关地址位必须和地址寄存器一致。 **注:**唤醒功能不能使用地址掩码

USCI喚醒控制寄存器 (UI^{2}C_WKCTL)

寄存器	偏移地址	R/W	描述	复位值
UI^{2}C_WKCTL	UI^{2}Cn_BA+0x54	R/W	USCI喚醒控制寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved						**WKADDREN**	**WKEN**

位	描述	
[31:2]	**Reserved**	保留
[1]	**WKADDREN**	**喚醒地址匹配使能位** 0 = 芯片依据起始信号喚醒 1 = 芯片依据地址匹配喚醒
[0]	**WKEN**	**喚醒使能位** 0 = 禁用喚醒功能 1 = 使能喚醒功能

USCI唤醒状态寄存器 (UI^{2}C_WKSTS)

寄存器	偏移地址	R/W	描述	复位值
UI^{2}C_WKSTS	UI^{2}Cn_BA+0x58	R/W	USCI唤醒状态寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved							**WKF**

位	描述	
[31:1]	**Reserved**	保留
[0]	**WKF**	**唤醒标志** 芯片被从掉电模式唤醒，该位置1软件写1清除该位。

USCI协议控制寄存器 – I²C (UI²C_PROTCTL)

寄存器	偏移地址	R/W	描述	复位值
UI²C_PROTCTL	UI²Cn_BA+0x5C	R/W	USCI协议控制寄存器	0x0000_0000

31	30	29	28	27	26	25	24
PROTEN	Reserved					TOCNT	
23	22	21	20	19	18	17	16
TOCNT							
15	14	13	12	11	10	9	8
Reserved						MONEN	SCLOUTEN
7	6	5	4	3	2	1	0
Reserved		PTRG	ADDR10EN	STA	STO	AA	GCFUNC

位	描述	
[31]	**PROTEN**	**I²C协议使能位** 0 = 禁用 I²C 协议 1 = 使能I²C 协议
[30:26]	**Reserved**	保留
[25:16]	**TOCNT**	**超时时钟周期** 当每个中断被清除的时候，该位域表示需要多少时钟周期（时钟源由TMCNTSRC (UI²C_BRGEN [5])选择）。当TOCNT大于0时超时使能。 **注:**在I²C模式下，TMCNTSRC (UI²C_BRGEN [5])必须设置为0
[15:10]	**Reserved**	保留
[9]	**MONEN**	**监视器模式使能位** 该位使能监视器模式。在监视器模式下，SDA会以高阻态输入。这样就可以阻止I²C模块输出任何数据（包括ACK）到I²C数据总线上。 0 = 禁用监视器模式 1 = 使能监视器模式 **注:**取决于SCLOUTEN位的状态，SCL输出被强制为高，阻止模块控制I²C时钟线
[8]	**SCLOUTEN**	**SCL输出使能位** 该位使能监视器SCL被拉低。监视器会拉低SCL直到有时间响应I²C中断 0 = 由于开漏机制SCL输出强制为高 1 = I²C模块扮演从机角色就像是正常操作，I²C保持时钟线为低直到有时间去清除I²C中断
[7:6]	**Reserved**	保留

[5]	**PTRG**	**I²C 协议触发 (只写)** 当新状态出现在USCI_PROTSTS寄存器中时，如果相关中断使能位被设置，将会产生中断请求。在相关中断位被置1之后必须软件写1到该位，然后继续执行I²C协议功能直到STOP激活或是PROTEN被禁用。 0 = I²C's钳位禁用，并且I²C协议功能继续执行 1 = I²C's钳位激活
[4]	**ADDR10EN**	**位地址功能使能位** 0 = 禁用10位地址匹配功能 1 = 使能10位地址匹配功能
[3]	**STA**	**I²C起始控制位** 设置STA为1，进入主机模式，如果总线处于空闲状态，I²C硬件会送出起始或重复起始条件到总线
[2]	**STO**	**I²C停止控制位** 在主机模式，设置STO来传送一个停止条件到总线，然后I²C硬件将会检查总线状况，如果检测到一个停止状况，这个标志会被硬件自动清除。在从机模式下，当总线错误(UI²C_PROTSTS.ERRIF = 1)的时候，设置STO复位I²C硬件来进入"无地址"从机模式
[1]	**AA**	**应答控制位** 当 AA=1 先于地址或数据接收，在SCL线上的应答时钟脉冲期间将返回一个应答(SDA上为低电平)，有两种情况： 1.)从机正在应答主机发送的地址。 2.)接收设备正在应答发送设备发送的数据。当 AA = 0 先于地址或数据接收，则在SCL线上的应答时钟脉冲期间将返回一个非应答（SDA上为高电平）。
[0]	**GCFUNC**	**广播呼叫功能** 0 = 广播呼叫功能禁用 1 = 广播呼叫功能使能 **注**：当设置了（USCI_PROTCTL [4]）时，不要设置该位。

USCI协议中断使能寄存器 – I²C (UI²C_PROTIEN)

寄存器	偏移地址	R/W	描述	复位值
UI²C_PROTIEN	UI²Cn_BA+0x60	R/W	USCI协议中断使能寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved	ACKIEN	ERRIEN	ARBLOIEN	NACKIEN	STORIEN	STARIEN	TOIEN

位	描述	
[31:7]	**Reserved**	保留
[6]	**ACKIEN**	**应答中断使能控制** 如果应答被主机检测到，该位是使能产生应答中断 0 = 禁用应答中断 1 = 使能应答中断
[5]	**ERRIEN**	**错误中断使能控制** 如果一个I²C错误条件被检测到(由 ERR (UI²C_PROTSTS [16])指示)，该位是使能产生错误中断 0 = 禁用错误中断 1 = 使能错误中断
[4]	**ARBLOIEN**	**仲裁丢失中断使能控制** 如果仲裁丢失事件被检测到，该位是使能产生仲裁丢失中断 0 = 禁用仲裁丢失中断 1 = 使能仲裁丢失中断
[3]	**NACKIEN**	**非应答中断使能控制** 如果非应答被主机检测到，该位是使能产生非应答中断 0 = 禁用非应答中断 1 = 使能非应答中断
[2]	**STORIEN**	**停止条件接收中断使能控制** 如果停止条件被检测到，该位是使能产生停止条件中断 0 = 禁用停止条件中断 1 = 使能停止条件中断

[1]	**STARIEN**	**起始条件接收中断使能控制** 如果起始条件被检测到，该位是使能产生起始条件中断 0 = 禁用起始条件中断 1 = 使能起始条件中断
[0]	**TOIEN**	**超时中断使能控制** 在I^2C协议下，该位是使能产生超时事件中断 0 = 禁用超时中断 1 = 使能超时中断

USCI协议中断状态寄存器 – I²C (UI²C_PROTSTS)

寄存器	偏移地址	R/W	描述	复位值
UI²C_PROTSTS	UI²Cn_BA+0x64	R/W	USCI协议中断状态寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved				ERRARBLO	BUSHANG	WRSTSWK	WKAKDONE
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
SLAREAD	SLASEL	ACKIF	ERRIF	ARBLOIF	NACKIF	STORIF	STARIF
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved	ONBUSY	TOIF	Reserved				

位	描述	
[31:20]	**Reserved**	保留
[19]	**ERRARBLO**	**仲裁丢失错误** 该位表示由于不能被输入处理器过滤的噪声引起总线仲裁丢失。当ERRARBLO被置位时I²C会发送起始条件。因此从机模式下无需关心。 0 = 总线传输状态正常 1 = 总线传输仲裁丢失错误 **注:**该位没有中断信号，并且当START条件出现时会被硬件自动清除
[18]	**BUSHANG**	**总线挂起** 该位指示总线挂起状态，当SCL保持高时4位计数器会计数并且参考源是f_{SAMP_CLK}.挂起计数器会计数到溢出并且在SDA位低时该位被置位。计数器由SCL下降沿信号复位。 0 = 总线传输正常 1 = 总线传输挂起 **注:**该位没有中断信号，并且当起始条件出现时会被硬件自动清除
[17]	**WRSTSWK**	**地址帧中的读/写状态位** 0 = 写命令被记录到地址匹配唤醒帧中 1 = 读命令被记录到地址匹配唤醒帧中
[16]	**WKAKDONE**	**唤醒地址帧应答位完成** 0 = 地址匹配帧ACK位阶段没有完成 1 = 掉电状况下地址匹配帧ACK位阶段已完成 **注:**当WKUPIF被置位时该位不会释放
[15]	**SLAREAD**	**从机读请求状态** 该位表示从机读请求被检测到 0 = 没有检测到从机R/W位是1

		1 = 检测到从机R/W位是1 **注:**该位没有中断信号，它会被硬件自动清除
[14]	**SLASEL**	**从机选择状态** 该位指示设备被选作从机 0 = 设备没有被选作从机 1 = 设备被选作从机 **注:**该位没有中断信号，它会被硬件自动清除
[13]	**ACKIF**	**应答接收中断标志** 主机模式下该位表示应答位已被接收。如果UI^{2}C_PROTCTL.ACKIEN = 1会产生中断 0 = 应答没有被接收 1 = 应答被接收 **注：** 软件写1清除
[12]	**ERRIF**	**错误中断标志** 当START或STOP条件出现在信息帧的不合理位置时该位用于指示发生总线错误。不合理位置比如在地址字节、数据字节或应答位这些串行传输。如果UI^{2}C_PROTCTL.ERRIEN = 1会产生中断 0 = I^{2}C错误没有被检测到 1 = I^{2}C 错误被检测到 **注1：** 软件写1清除 **注2：** 从机模式该位被置位时，用户必须写1到STO寄存器是设备进入“无地址”从机模式
[11]	**ARBLOIF**	**仲裁丢失中断标志** 该位表示发送仲裁丢失。如果UI^{2}C_PROTCTL.ARBLOIEN = 1会产生中断 0 = 没有发生仲裁丢失 1 = 仲裁丢失 **注：** 软件写1清除
[10]	**NACKIF**	**非应答接收中断标志** 主机模式下该位表示非应答位已被接收。如果UI^{2}C_PROTCTL.NACKIEN = 1会产生中断 0 = 非应答没有被接收 1 = 非应答被接收 **注：** 软件写1清除
[9]	**STORIF**	**停止条件接收中断标志** 该位表示停止条件在I^{2}C总线上已被检测到。如果UI^{2}C_PROTCTL.STORIEN = 1会产生中断 0 = 停止条件没有被检测到 1 = 停止条件被检测到 **注：** 软件写1清除
[8]	**STARIF**	**起始条件接收中断标志** 该位表示主机模式下起始条件或是重复起始条件在I^{2}C总线上已被检测到。然而在从机模式下表示重复起始条件被检测到。 如果UI^{2}C_PROTCTL.STARIEN = 1会产生中断 0 = 起始条件还没有被检测到 1 = 停止条件已被检测到

		注： 软件写1清除
[7]	**Reserved**	保留
[6]	**ONBUSY**	**总线忙** 表明总线上正在进行数据传输。当START条件被检测到时被硬件置位。当STOP被检测到时被硬件清除 0 = 总线空闲(SCLK和SDA都为高). 1 = 总线忙
[5]	**TOIF**	**超时中断标志** 0 = 没有发生超时中断 1 = 没有发生超时中断 **注:**软件写1清除
[4:0]	**Reserved**	保留

USCI时序配置寄存器 (UI^{2}C_TMCTL)

寄存器	偏移地址	R/W	描述	复位值
UI^{2}C_TMCTL	UI^{2}Cn_BA+0x8C	R/W	I^{2}C时序配置寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							**HTCTL**
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
HTCTL							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							**STCTL**
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
STCTL							

位	描述	
[31:25]	**Reserved**	保留
[24:16]	**HTCTL**	**保持时间配置控制寄存器** 该域是发送模式下用来在SCL下降沿和SDA边沿之间产生延时时序。 延时保持时间外设时钟数= HTCTL x f_{PCLK}.
[15:9]	**Reserved**	保留
[8:0]	**STCTL**	**建立时间配置控制寄存器** 该域是发送模式下用来在SDA边沿和SCL上升沿之间产生延时时序。 延时建立时间外设时钟数= STCTL x f_{PCLK}.

6.28 CAN接口

6.28.1 概述

C_CAN由CAN内核，报文RAM，报文处理器，控制寄存器和模块接口组成。CAN内核按照CAN协议2.0版本A部分和B部分规范执行通信。位速率最高可达1Mbit/s。为与物理层相连，还需另外外接硬件收发器。

在CAN网络中，各个报文对象是可以独立配置的。报文对象和用于在接收时进行报文过滤的标识符掩码都存储在报文RAM中。所有与报文处理相关的功能都在报文处理器中执行。这些功能包括接收过滤、CAN内核与报文RAM之间的报文传输、处理传送请求以及模块中断的产生。

C-CAN的寄存器组可以通过模块接口被软件直接访问。这些寄存器用来控制/配置CAN内核和报文处理器，以及访问报文RAM。

6.28.2 特性

- 支持CAN协议2.0版本A和B部分
- 位速率最高可达1 Mbit/s
- 32个报文对象
- 每个报文对象都有自己的标示符掩码
- 可编程FIFO模式（链接报文对象）
- 中断可屏蔽
- 禁用时间触发CAN应用下的自动重传模式
- 支持用于自检测的可编程环回模式
- 连接到AMBA APB总线上的16-位模块接口
- 支持唤醒功能

6.28.3 框图

C_CAN接口与AMBA APB 总线相连。图6.28-1展示了C_CAN框图。

- CAN内核
  - 包括CAN协议控制器和用于报文串/并数据转换的RX/TX移位寄存器
- 报文RAM
  - 用于保存报文对象和标识符掩码
- 寄存器组
  - 所有寄存器用来控制和配置C_CAN.
  - 报文处理模块
  - 用于控制CAN内核的RX/TX移位寄存器和报文RAM间的数据传输的状态机，以及按照控制和配置寄存器中设定产生中断

- 模块接口
  - C_CAN接口与AMBA APB 通过CPU的16位地址总线相连

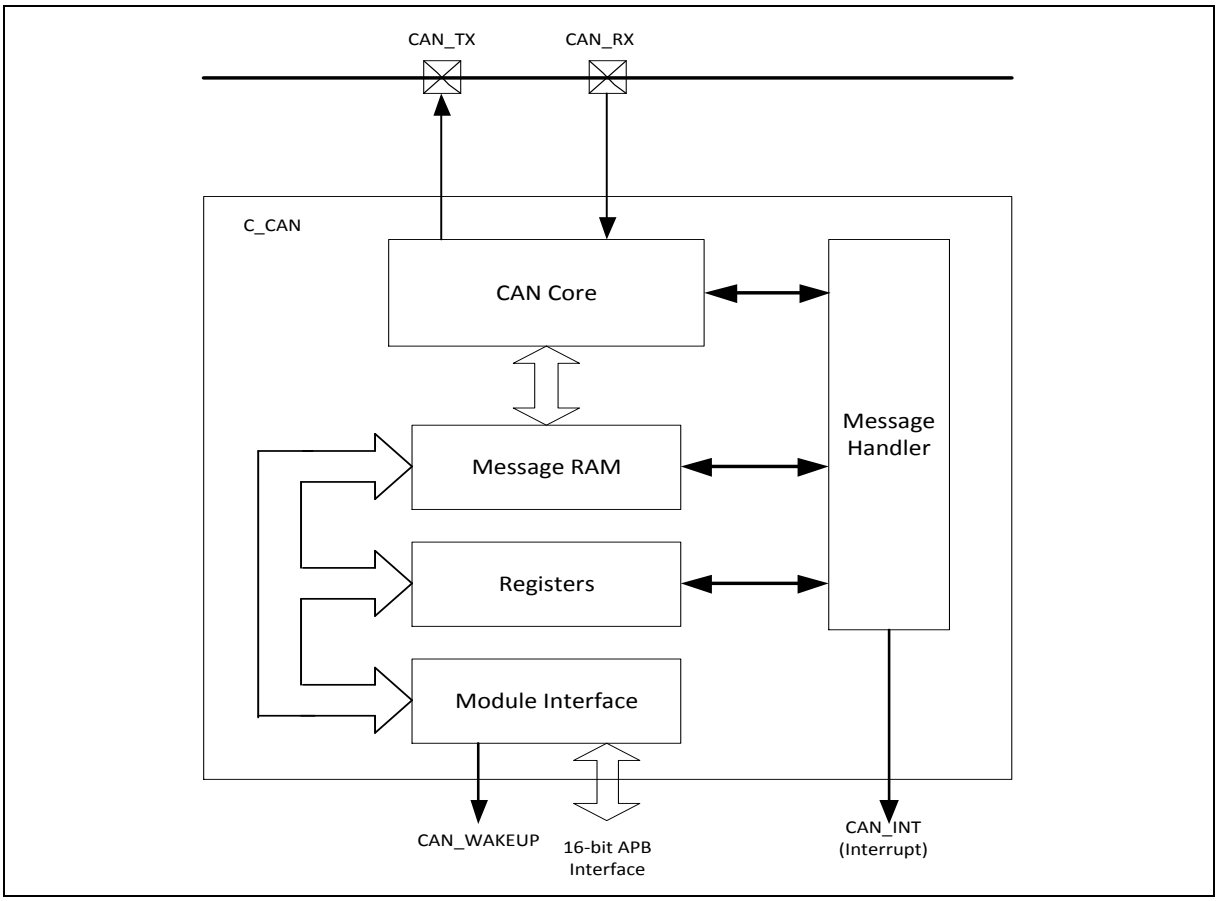


图 6.28-1 CAN 外设框图

6.28.4 基本配置

6.28.4.1 CAN0基本配置

- 时钟源配置
  - 使能CAN时钟(CAN_EN (APBCLK0[24])).
- 复位配置
  - 复位CAN控制器(CAN_RST (IPRSTC1[24])).
- 引脚配置

组	引脚名称	GPIO	MFP
CAN0	CAN0_RXD	PD.10, PE.15	MFP4
		PA.13	MFP6
		PB.10	MFP8

		PA.4, PC.4	MFP10
	CAN0_TXD	PD.11, PE.14	MFP4
		PA.12	MFP6
		PB.11	MFP8
		PA.5, PC.5	MFP10

6.28.4.2 CAN1基本配置

- 时钟源配置
  - 使能CAN时钟(CAN_EN (APBCLK0[25])).
- 复位配置
  - 复位CAN控制器(CAN_RST (IPRSTC1[25])).
- 引脚配置

组	引脚名称	GPIO	MFP
CAN1	CAN1_RXD	PB.6, PD.12	MFP5
		PG.1	MFP7
		PF.8	MFP8
		PC.9, PE.6	MFP9
		PC.2	MFP10
	CAN1_TXD	PB.7, PC.13	MFP5
		PG.0	MFP7
		PF9	MFP8
		PC.10, PE.7	MFP9
		PC.3	MFP10

6.28.4.3 CAN2基本配置

- 时钟源配置
  - 使能CAN时钟(CAN_EN (APBCLK1[28])).
- 复位配置
  - 复位CAN控制器(CAN_RST (IPRSTC2[28])).
- 引脚配置

组	引脚名称	GPIO	MFP
CAN2	CAN2_RXD	PD8	MFP6
		PF.6	MFP8
		PC.0	MFP10

		PB.8	MFP12
	CAN2_TXD	PD.9	MFP6
		PF.7	MFP8
		PC.1	MFP10
		PB.9	MFP12

6.28.5 功能描述

6.28.5.1 软件初始化

软件初始化可以通过软件或硬件复位，或者通过进入bus-off状态置位CAN控制寄存器的Init位(CAN_CON[0])来实现。

当Init位被置位后，所有CAN总线上的报文传输都会停止，CAN_TX输出管脚上的状态为隐性（高电平）。错误管理逻辑（EML）计数器将保持不变。设置Init位不会改变任何配置寄存器。

为了初始化CAN控制器，软件必须设置位定时寄存器和每个报文对象。如果不需要用到某个报文对象，那么必须清除相应的MsgVal位 (CAN_IFn_ARB2[15])，否则，整个报文对象都需要进行初始化。

当CAN控制寄存器的Init和CCE位(CAN_CON[6])都被置位时，用于配置位定时的位定时寄存器和波特率预分频扩展寄存器的访问才会被使能。

复位Init位（只能通过软件方式）完成软件初始化。接着位流处理器（BSP）（参看6.27.7.15：配置位定时）在等待 11个连续隐性位（总线空闲）的位序列后先将自身与CAN总线上数据实现同步，然后再参与总线活动和开始报文传输。

报文对象的初始化是独立于Init，可以在忙碌时完成。但在BSP开始传输报文之前，所有报文对象的标识符必须先配置成指定值或设成无效。

在正常工作期间改变一个报文对象的配置时，软件必须先复位相应的MsgVal位，然后再改变配置。当配置完成后，需再次设置MsgVal 位

6.28.5.2 CAN报文传输

一旦C_CAN被初始化以及Init位(CAN_CON[0])位被重置为0，C_CAN内核就会将自身与CAN总线同步，并开始报文传输。

如果接收到的报文通过了报文处理器的接收过滤，将会被存储在相应的报文对象中。保存在报文对象中的报文包括所有的仲裁位，DLC(CAN_IFn_MCON[3:0]) 和 8 个字节数据 (CAN_IFn_DAT_A1/2；CAN_IFn_DAT_B1/2)。如果使用了标识符掩码,那些被掩码为”无关”位的仲裁位会在报文对象中被覆盖。

软件可以通过接口寄存器在任何时间去读或写每条报文。如果同时访问，报文处理器将保证数据的一致性。

发送的报文由应用软件进行更新。如果一个报文永久性的存在某个报文对象中（仲裁位和控制位在配置时被设定），那么仅有数据字节会被更新，在置位 TxRqst 位 (CAN_IFn_MCON[8]) 和 NewDat(CAN_IFn_MCON[15])位会后就会开始传送。如果几条传输报文被指派给同一个报文对象(当报文对象的数量不够用时)，那么在发送这些报文前必须对整个报文对象进行配置。

任何数量的报文对象都可以在同一时间请求传送。报文对象会按照它们的内部优先级进行依次传送。报文可以随时更新或设置为无效，甚至在发送请求还在等待阶段也可进行。如果一条报文在其未发送前被更新，则旧的数据将会被丢弃。

根据报文对象的配置，在接收到匹配标识符的遥控帧后，会自动产生报文发送请求

禁用自动重传

依照CAN协议规范（见ISO11898，6.3.3恢复管理），C_CAN 为那些在传送过程中丢失仲裁或被错误干扰的帧，提供自动重传机制。在传送完全成功之前，帧传送服务是不能证实给用户。这也就意味着，默认情况下，自动重传是使能的。在C_CAN工作在时间触发CAN（TTCAN，见ISO11898-1）环境时，可以禁用自动重传功能。

通过设置CAN控制器中的禁用自动重传（DAR(CAN_CON[5])）位为1来禁用自动重传模式。在这种操作模式下，用户必须考虑报文缓存控制寄存器中TxRqst(CAN_IFn_MCON[8]) 和NewDat(CAN_IFn_MCON[15])位不同设置时的状况：

- 当传送开始时，各个报文缓存的TxRqst位被清除，而NewDat位一直置位
- 当传送完全成功时，NewDat位被清除
- 当传送失败（丢失仲裁或发生错误），NewDat位保持置位
- 为了重新开始传送，软件必须再将TxRqst位置位

6.28.6 测试模式

设定CAN控制寄存器的Test(CAN_CON[7])位进入测试模式。在测试模式下，测试寄存器的Tx1(CAN_TEST[6])， Tx0(CAN_TEST[5])， LBack(CAN_TEST[4])， Silent (CAN_TEST[3]) 和Basic (CAN_TEST[2])位可写。Rx位监视CAN_RX管脚的状态，因此该位只读。当Test位被清除时，所有的测试寄存器功能被禁用。

6.28.6.1 静默模式

通过设置测试寄存器中的Silent位(CAN_TEST[3])为1，CAN内核可被设为Silent模式。在Silent模式中，C_CAN能够接收有效的数据帧和有效的遥控帧，但是它在CAN总线上仅发送隐性位，而且它不能启动发送。如果CAN内核被要求发送一个显性位（ACK位，错误帧），那么该显性位将在内部自动改道以便CAN内核检测到该显性位，而CAN总线仍然保持在隐性状态。因为可以在传送显性位时不会影响到CAN总线，所以Silent模式可以用来分析CAN总线的通讯状况。下图6.28-2为静默模式下，CAN_TX和CAN_RX与CAN内核的连接信号。

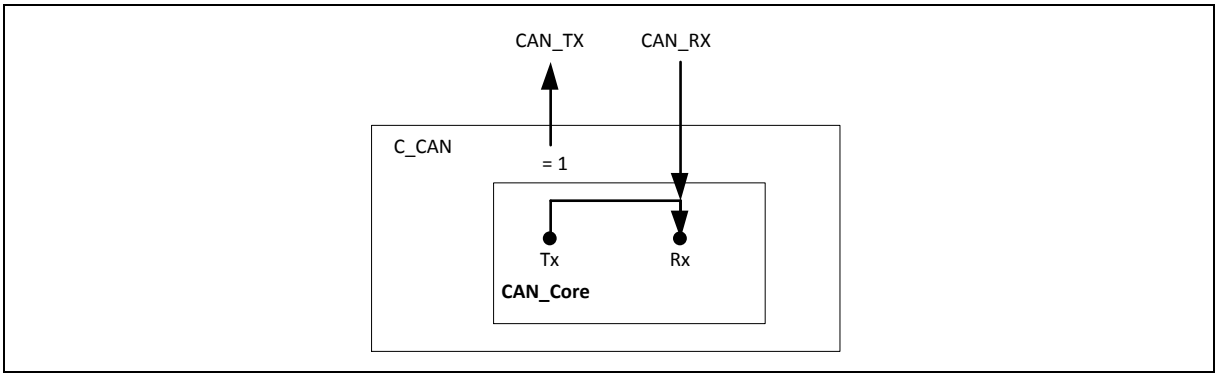


图 6.28-2 CAN 内核静默模式

6.28.6.2 回环模式

通过设定测试寄存器的LBack位(CAN_TEST[4])为1，CAN内核可被设为回环模式。在回环模式下，CAN内核把自己发送的报文作为接收到的报文对待，并将它们保存在接收缓存中（如果它们通过了接收

过滤）。图6.28-3为回环模式下，CAN_TX和CAN_RX 与CAN内核的连接信号

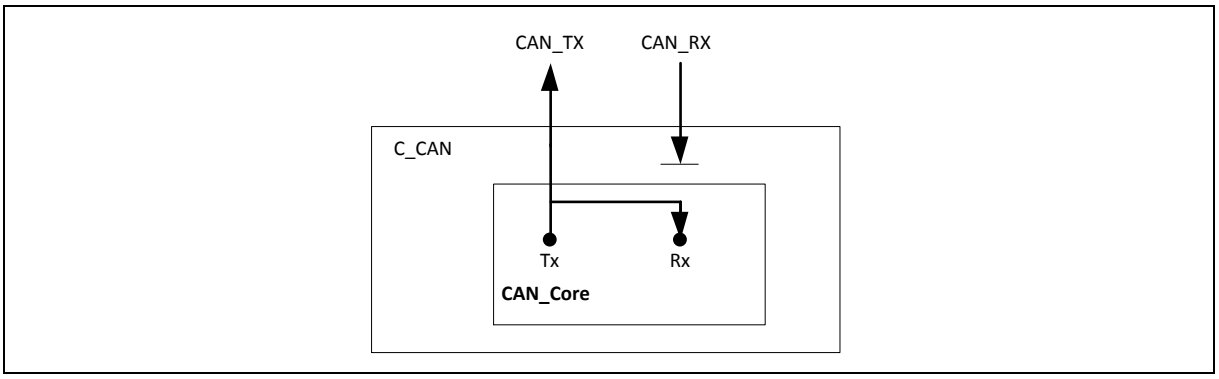


图 6.28-3 CAN 内核回环模式

该模式用于自检。为了不受外部的干扰，在回环模式下，CAN内核忽略应答错误（数据/远程帧应答槽内采样到隐性位）。在该模式下，CAN内核的Tx输出通过内部直接反馈到它的Rx输入上。CAN_RX输入管脚的真实值则被CAN内核忽略。发送的报文信号仍可通过 CAN_TX管脚进行监测。

6.28.6.3 回环模式和静默模式的组合

通过设定LBack(CAN_TEST[4])和Silent(CAN_TEST[3])位同时为1，还可以将回环模式和静默模式结合一起使用。该模式可用于 “热自测（Hot Selftest）”，即C_CAN可以在不影响已与CAN_TX和CAN_RX管脚相连的CAN系统的情况下进行测试。在该模式下，CAN_RX管脚与CAN内核断开，CAN_TX管脚保持隐性电平。图6.28-4为回环模式和静默模式整合下，CAN_TX 和CAN_RX与CAN内核的连接信号。

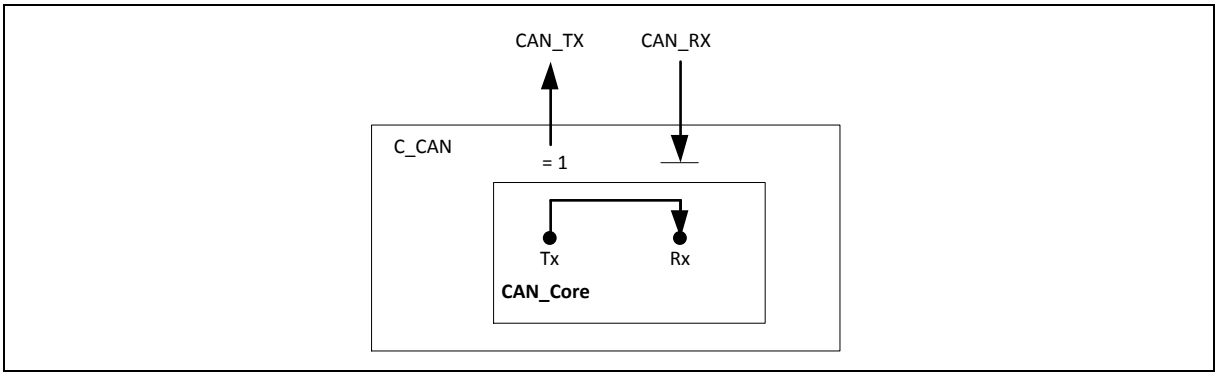


图 6.28-4 CAN 内核回环模式与静默模式组合

6.28.6.4 基本模式

通过设定测试寄存器的Basic位(CAN_TEST[2])为1可以设定CAN内核工作在Basic模式。该模式下C_CAN工作时没有报文RAM。

IF1寄存器组用作发送缓存。IF1寄存器组中内容通过设定IF1命令请求寄存器中的Busy位(CAN_IFn_CREQ[15])为1来请求发送。当Busy位置位时，IF1寄存器组锁定。Busy位指示传送正在挂起。

一旦CAN总线空闲，IF1寄存器组会被载入到CAN内核的移位寄存器，然后传送开始。当传输完成时，Busy位复位，并释放锁定的IF1寄存器组。

当IF1寄存器组锁定时，挂起的发送可以被随时中止，方法为复位IF1的命令请求寄存器的Busy位。如果软件已经复位了Busy位，那么为防止仲裁丢失或错误出现的重传是被禁用的。

IF2寄存器组用作接收缓存。收到报文后，移位寄存器的内容不经过任何接收过滤就存储到IF2寄存器组。

此外，在报文传输的过程中，移位寄存器的真实内容能够被监视。每次写IF2 命令请求寄存器中的Busy位为1，读报文对象就会被初始化，移位寄存器的内容则被保存在IF2寄存器组。

在基本模式下，所有与报文对象相关的控制和状态位以及IFn掩码寄存器的控制位都不能进行赋值。命令请求寄存器的报文编号也不能赋值。IF2 报文控制寄存器的NewDat(CAN_IFn_MCON[15])和MsgLst(CAN_IFn_MCON[14])位保留其功能。DLC3-0指示接收到的DLC(CAN_IFn_MCON[[3:0])，其他控制位读取值皆为‘0’。

6.28.6.5 软件控制CAN_TX管脚

CAN发送管脚CAN_TX上有四种输出功能可用。除了缺省功能（串行数据输出），CAN发送管脚还能驱动输出CAN采样点信号，用于监视CAN内核的位定时，而且CAN发送管脚还能持续驱动输出显性或隐性电平。后两种功能，结合可读的CAN接收管脚CAN_RX，能够用于检测CAN总线的物理层。

CAN_TX管脚的输出模式通过设定CAN测试寄存器的Tx1 (CAN_TEST[6])和Tx0(CAN_TEST[5])位进行选择。

CAN_TX管脚参与了三种测试模式所有的CAN协议功能。当CAN进行报文传输或者选择任一测试模式（回环模式，静默模式,或基本模式）时，CAN_TX 必须使用它的默认功能。

6.28.7 CAN通信

6.28.7.1 管理报文对象

报文RAM中的报文对象配置（除CAN控制寄存器的MsgVal, NewDat, IntPnd, 和TxRqst位外)不会受芯片复位影响。在应用软件清除Init位之前，所有的报文对象必须被应用软件初始化或是设置为 “无效”（MsgVal=0），而且位时序必须在应用软件清Init位(CAN_CON[0])之前被配置好。

通过配置两个接口寄存器中一个接口寄存器的屏蔽、仲裁、控制和数据域为期望值后，报文对象的配置完成。通过写相应的IFn命令请求寄存器，IFn报文缓存寄存器组会被载入到报文RAM中指定地址的报文对象中。

CAN控制寄存器的Init位被清除后，CAN内核的CAN协议控制状态机和报文处理器状态机控制C_CAN的内部数据流。收到的报文通过接收过滤后被保存在报文RAM中。挂起发送请求的报文被载入到CAN内核的移位寄存器，并通过CAN总线进行发送。

应用软件通过IFn接口寄存器组读取收到的报文以及更新发送的报文。依照配置，应用软件会被特定的CAN报文和CAN错误事件打断

6.28.7.2 报文处理器状态机

报文处理器控制CAN内核的Rx/Tx移位寄存器、报文RAM和IFn寄存器组之间的数据传输。

报文处理器FSM控制如下功能：

- 从IFn寄存器传送数据到报文RAM
- 从报文RAM传送数据到IFn寄存器
- 从移位寄存器传送数据到报文RAM
- 从报文RAM传送数据到移位寄存器
- 从移位寄存器传送数据到接收过滤单元

- 扫描报文RAM寻找匹配报文对象
- 处理TxRqst标志位
- 处理中断

6.28.7.3 报文RAM的数据传输

当应用软件对IFn寄存器组和报文RAM间的数据传输初始化后，报文处理器设置各自命令请求寄存器的Busy位（CAN_IFn_CREQ[15]）为‘1’。在传输完成后，Busy位会被再次清除（见图6.28-5）。

命令掩码寄存器可以指定是一个完整的报文对象还是报文对象的部分内容将被传输。因为报文RAM的结构，写操作不能仅仅针对报文对象的单独某位/字节，而是必须写入完整的报文对象到报文RAM。因此，从IFn寄存器到报文RAM的数据传输需要一个读-修改-写的周期。首先，将报文对象中不变的部分从报文RAM读出，然后将报文缓存寄存器的完整报文内容写入到报文对象

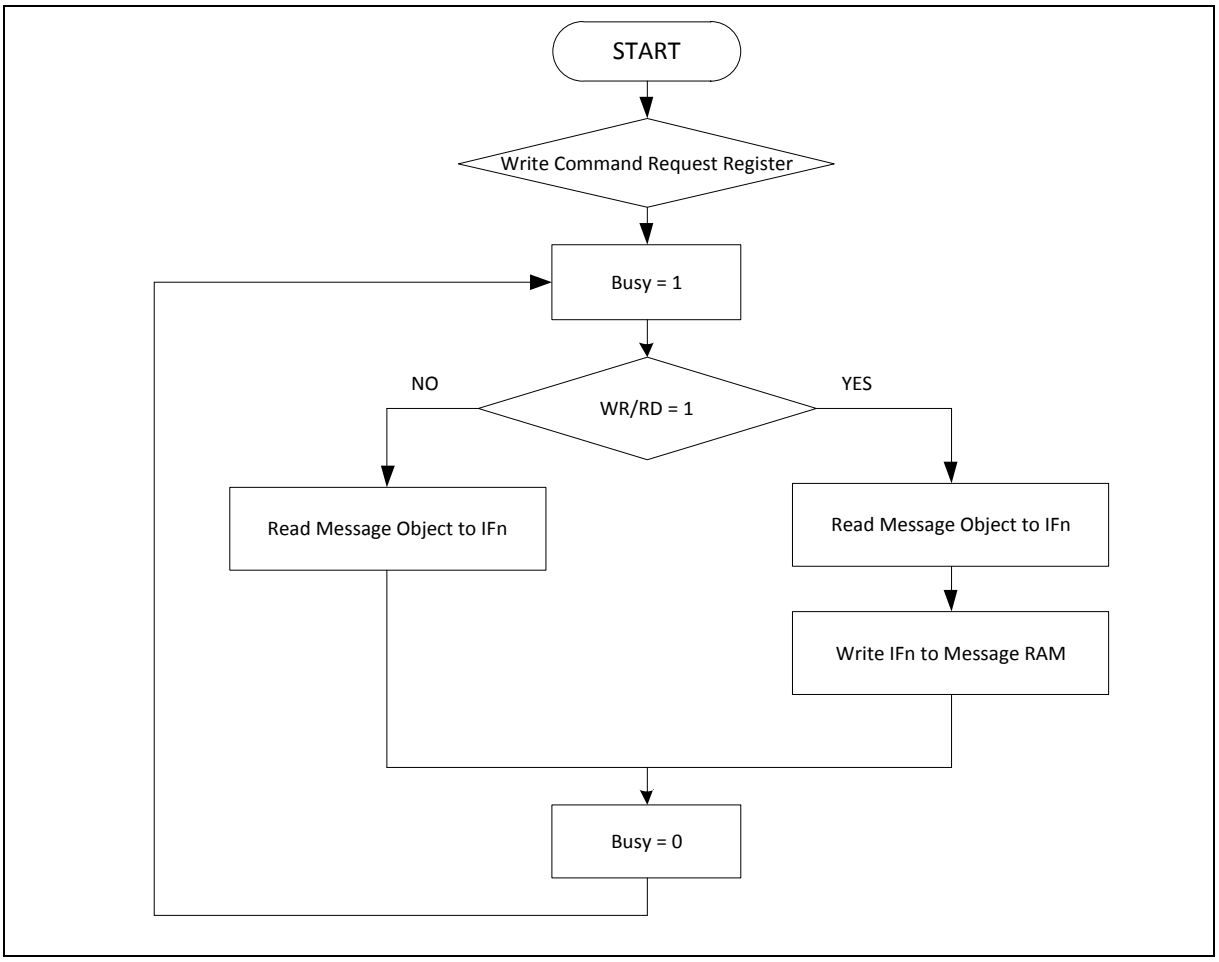


图 6.28-5 Ifn 寄存器和报文 RAM 间的数据传输

在写入一个报文对象部分内容后，没被命令掩码寄存器选中的报文缓存寄存器将设置选中报文对象的实际内容。

在读取一个报文对象部分内容后，没被命令掩码寄存器选中的报文缓存寄存器将保持不变。

6.28.7.4 报文传输

如果CAN内核单元的移位寄存器准备载入，但在IFn寄存器组和报文RAM之间没有数据传输，内核将重新评估MsgVal 位(CAN_IFn_ARB2[15]) 和TxRqst位(CAN_TXREQ1/2)。具有最高优先级发送请求的有效报文对象将被报文处理器载入到移位寄存器，开始发送。报文对象的NewDat (CAN_IFn_MCON[15])位被复位。

成功传送后或者自传送开始后没有新的数据写入报文对象(NewDat = ‘0’)，报文控制寄存器TxRqst 位(CAN_IFn_MCON[8])将被复位。如果报文控制寄存器TxIE位(CAN_IFn_MCON[11])被置位，中断标识符寄存器的IntPnd 位(CAN_IFn_MCON[13])在传送成功后将被置位。如果C_CAN仲裁失败或者在传送过程中有错误发生，报文将会在CAN总线空闲的时候进行重传。同时，如果有更高优先级的报文传送请求，则报文将会按照报文的优先级顺序进行传送。

6.28.7.5 收到报文的接收过滤

当一个输入报文的仲裁和控制域(Identifier + IDE + RTR + DLC)完全被移位到CAN内核的Rx/Tx移位寄存器时，报文处理器FSM开始扫描报文RAM，寻找匹配的有效报文对象。

扫描报文RAM寻找匹配的报文对象，需要将CAN内核移位寄存器的仲裁位载入到接收过滤单元。步骤为：先载入报文对象1的仲裁和掩码域（包括MsgVal (CAN_IFn_ARB2[15]), UMask (CAN_IFn_MCON[12]), NewDat (CAN_IFn_MCON[15]), 和EoB (CAN_IFn_MCON[7])）到接收过滤单元，再和移位寄存器的仲裁位进行比较。该步骤对接下来的每一个报文对象重复进行，直到匹配的报文对象出现或者到达报文RAM的末端。

如果匹配发生，扫描停止，报文处理器FSM将根据接收帧的类型（数据帧或远程帧）进行处理。

接收数据帧

报文处理器FSM将CAN内核移位寄存器中报文保存到报文RAM中的各个报文对象。不仅数据字节，所有仲裁位和数据长度码都被保存到相应的报文对象中。这是为了将数据字节与标识符继续保持关联性

置位NewDat 位(CAN_IFn_MCON[15])用于指示新数据（还没有被软件看到的）已经收到了。当报文对象被读取后，应用软件必须复位NewDat 位(CAN_IFn_MCON[15])。如果在接收的时候，NewDat 位已经被置位，MsgLst (CAN_IFn_MCON[14])被置位用来指示之前的数据（假定还没有被软件看到）已丢失。如果RxIE位 (CAN_IFn_MCON[10])被置位，IntPnd 位(CAN_IFn_MCON[13])也置位将使中断寄存器指向该报文对象

当请求的数据帧刚刚收到时，报文对象的TxRqst(CAN_IFn_MCON[8])位会被复位用来阻止传送远程帧。

接收远程帧

当收到一个远程帧时，必须考虑匹配的报文对象的三种配置：

1. Dir (CAN_IFn_ARB2[13]) = ‘1’ (方向=发送), RmtEn (CAN_IFn_MCON[9]) = ‘1’, UMask (CAN_IFn_MCON[12]) = ‘1’或’0’。

   当收到匹配的远程帧时，报文对象的TxRqst位被置位。报文对象的其他部分保持不变。

2. Dir = ‘1’ (方向= 发送), RmtEn = ‘0’, UMask =’0’

   当收到匹配的远程帧时，报文对象的TxRqst位保持不变，远程帧被忽略。

3. Dir = ‘1’ (方向= 发送), RmtEn = ‘0’, UMask =’1’

   当收到匹配的远程帧时，报文对象的TxRqst位被复位。移位寄存器中的仲裁和控制域(Identifier + IDE + RTR + DLC)被保存到报文RAM中的报文对象，报文对象的NewDat(CAN_IFn_MCON[15])

位被置位。报文对象的数据域保持不变。收到的远程帧处理方式与数据帧类似。

6.28.7.6 接收/发送优先级

报文对象接收/发送的优先级与报文号相关。报文对象1拥有最高优先级，报文对象32拥有最低优先级。如果有一个以上的发送请求挂起时，它们将根据相应报文对象的优先级来进行服务。

6.28.7.7 配置传送对象

表6.28-1展示了怎样初始化一个传送对象。

Ms	Arb	Data	Mask	EoB	Dir	NewDat	MsgLst	RxIE	TxIE	IntPnd	RmtEn	TxRqst
1	appl.	appl.	appl.	1	1	0	0	0	appl.	0	appl.	0

表 6.28-1 初始化传送对象

注: appl. = 应用软件。

仲裁寄存器(ID28-0 (CAN_IFn_ARB1/2) 和Xtd 位(CAN_IFn_ARB2[14]))由应用软件设置。它们定义标识符和发出的报文类型。如果11-位标识符（标准帧）被使用，它将被编程到ID28 - ID18。ID17 - ID0可以被忽略。

如果TxIE(CAN_IFn_MCON[11])位被置位，在报文对象成功发送之后，IntPnd(CAN_IFn_MCON[13])位将被置位。

如果RmtEn(CAN_IFn_MCON[9])位被置位，收到匹配的远程帧将置TxRqst(CAN_IFn_MCON[8])位，而且该远程帧会自动被一个数据帧应答。

数据寄存器的值(DLC3-0 (CAN_IFn_MCON[3:0]) , Data0-7)由应用软件提供。在数据有效之前，TxRqst和RmtEn可能不会被置位。

(UMask (CAN_IFn_MCON[12]) = '1')时,掩码寄存器(Msk28-0, UMask, MXtd, 和 MDir 位)可以用于允许有相似标识符的远程帧组设置TxRqst位。但Dir (CAN_IFn_ARB2[13])位不能被屏蔽

6.28.7.8 更新一个传送对象

不必复位MsgVal (CAN_IFn_ARB2[15])和TxRqst(CAN_IFn_MCON[8])，软件就可以通过IFn接口寄存器随时更新发送对象的数据字节。

即使只更新部分数据字节，在内容传输给报文对象之前，IFn数据A寄存器或B寄存器中的所有4个字节数据都必须有效。应用软件必须写所有4个字节到IFn数据寄存器或者在软件写新的数据字节前报文对象已被传输到IFn数据寄存器。

仅当（8个）数据字节更新时，先写0x0087到命令掩码寄存器，然后再写报文对象的编号到命令请求寄存器，同时更新数据字节及设置TxRqst

当数据更新时，为防止在传输的最后阶段TxRqst复位，必须同时置位NewDat (CAN_IFn_MCON[15])和TxRqst。

当NewDat和TxRqst一起被置位时，NewDat将会在新的传送开始时立即被复位。

6.28.7.9 配置一个接收对象

表6.28-2展示了怎样初始化一个接收对象。

MsgVal	Arb	Data	Mask	EoB	Dir	NewDat	MsgLst	RxIE	TxIE	IntPnd	RmtEn	TxRqst
1	appl.	appl.	appl.	1	0	0	0	appl.	0	0	0	0

表 6.28-2 初始化接收对象

仲裁寄存器的值(ID28-0 (CAN_IFn_ARB1/2)和Xtd位(CAN_IFn_ARB2[14]))由应用软件设置。它们定义标识符和接收报文的类型。如果11-位标识符（“标准帧”）被使用，它将会编程到ID28 - ID18。ID17 - ID0可以被忽略。当带有11-位标识符的数据帧被收到时，ID17 - ID0将被置0。

如果RxIE(CAN_IFn_MCON[10])位置位，当接收到数据帧并被保存到报文对象时，IntPnd (CAN_IFn_MCON[13])位将被置位。

数据长度码（(DLC3-0 (CAN_IFn_MCON[3:0])由应用软件设置。当报文处理器保存一个数据帧到报文对象时，它将会保存收到的数据长度码和8数据字节。如果数据长度码少于8，则报文对象余下的字节将被未指定的值覆盖。

(UMask (CAN_IFn_MCON[12]) = ’1’)时,掩码寄存器(Msk28-0, UMask, MXtd和MDir)可以被用于允许有相似标识符的数据帧组被接收。在典型的应用中，Dir (CAN_IFn_ARB2[13])位不能被屏蔽

6.28.7.10 处理接收报文

应用软件可以通过IFn接口寄存器随时读一条收到报文。报文处理器的状态机将保证数据的一致性。

通常的，软件先写0x007F到命令掩码寄存器，然后写报文对象编号到命令请求寄存器。这就会将收到的报文整个从报文RAM传输到报文缓存寄存器。并且，在报文RAM(不是报文缓存)中NewDat (CAN_IFn_MCON[15])和 IntPnd (CAN_IFn_MCON[13])位将被清除。

如果报文对象使用掩码进行接收过滤，则仲裁位表示哪条匹配的报文已经收到了。

NewDat的真实值（当前值）表示自上次该报文对象被读取后，是否收到新的报文。MsgLst (CAN_IFn_MCON[14])的真实值表示自上次该报文对象被读取后，收到的报文是否超过1条。MsgLst不会被自动复位。

利用远程帧，软件可以请求另一个CAN节点为接收报文对象提供新的数据。设定接收报文对象的TxRqst(CAN_IFn_MCON[8])位将引发带有接收报文对象的标识符的远程帧进行传送。该远程帧触发另一个CAN节点开始发送与之匹配的数据帧。如果在发送远程帧之前收到匹配的数据帧，TxRqst位将被自动复位

6.28.7.11 配置报文对象FIFO缓存

除EoB(CAN_IFn_MCON[7])位外，FIFO缓存的接收报文对象群的配置和单个接收报文对象的配置是一样的。参看章节6.5.7.9：配置接收对象。

为在FIFO缓存中关联2个或更多报文对象，这些报文对象的标识符和掩码（如使用）必须设置为要匹配的值。因为报文对象的默认优先级，最低编号的报文对象将会是FIFO缓存的第一个报文对象。FIFO缓存中（除了最后的报文对象外）所有报文对象的EoB位都必须设置为0。FIFO缓存中的最后一个报文对象的EoB位需设置为1，表示模块的结束。

6.28.7.12 用报文对象FIFO缓存接收报文

收到的报文如果匹配FIFO缓存中的标识符会被保存到该FIFO缓存中的一个报文对象中，报文存储的顺序为从最低编号的报文对象开始。

当一条报文保存到报文对象FIFO缓存中时，该报文对象的NewDat(CAN_IFn_MCON[15])位被置位。当EoB(CAN_IFn_MCON[7])为0时通过置位NewDat，报文对象将被锁定用于之后的报文处理器的写访问，直到应用软件将NewDat位写回0。

报文群将一直保存在FIFO缓存中直到FIFO缓存的最后一个报文对象。如果之前没有一个报文对象通过写NewDat位为0释放，则FIFO缓存之后收到的报文都将被写入到该FIFO缓存的最后一个报文对象中，当然之前的报文会被覆盖。

6.28.7.13 从报文对象FIFO中读数据

当应用软件通过写报文对象编号到IFn命令请求寄存器将报文对象的内容传输到IFn报文缓存寄存器时，相应的命令掩码寄存器设置为：(TxRqst/NewDat (CAN_IFn_CMASK[2]) = '1' 和 ClrIntPnd (CAN_IFn_CMASK[3]) = '1')时，NewDat (CAN_IFn_MCON[15]) 和 IntPnd (CAN_IFn_MCON[13])位复位为零。报文控制寄存器以上各个寄存器位一直反映复位前的状态。

为保证FIFO缓存的正确使用，应用软件须从FIFO缓存中最低报文编号的报文对象读起。

图6.28-6表示应用软件如何处理一组和FIFO缓存相连的报文对象。

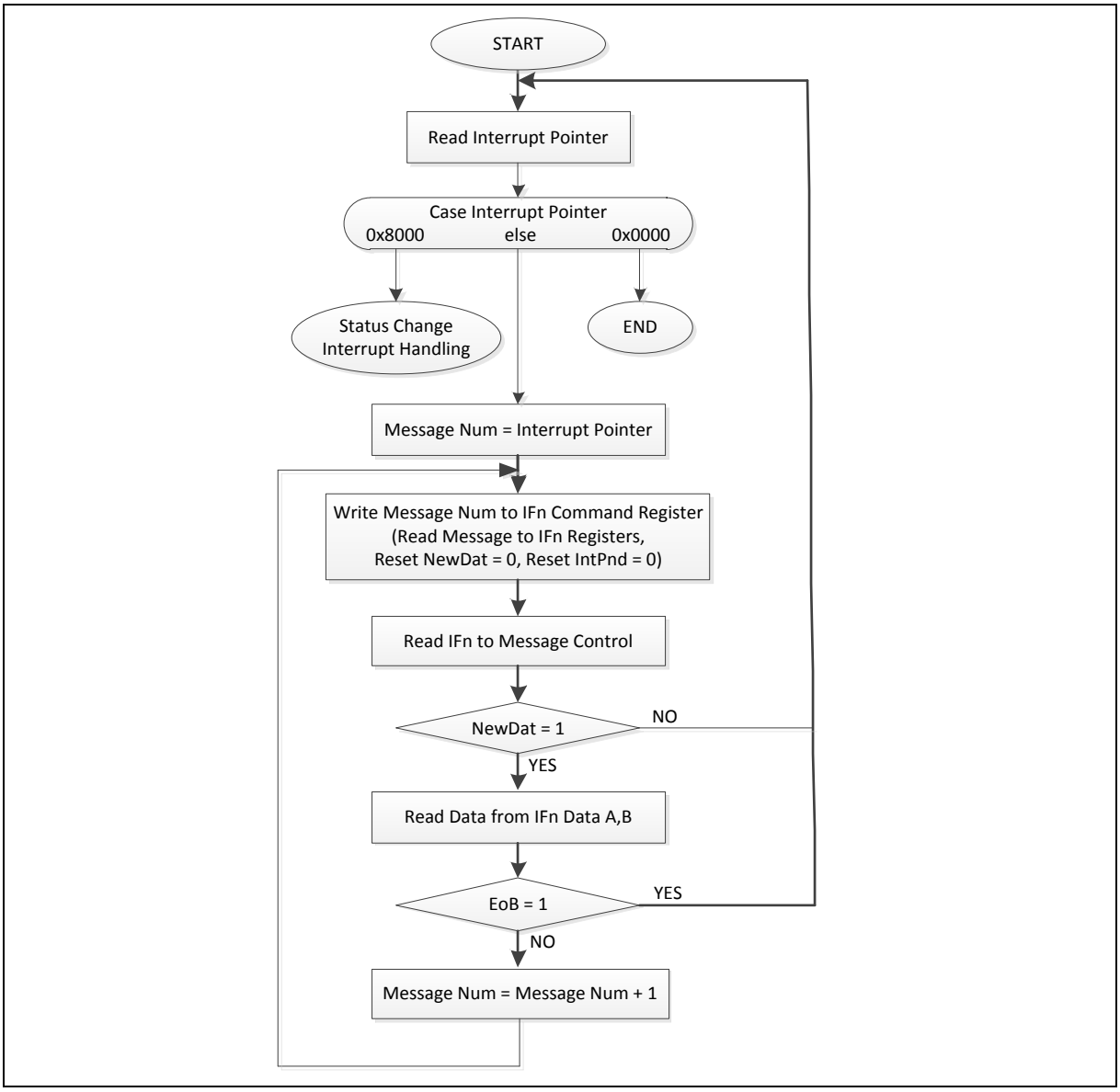


图 6.28-6 应用软件处理 FIFO 缓存

6.28.7.14 中断处理

如果几个中断同时挂起,则CAN的中断寄存器将指向具有最高优先级的挂起中断，而无需理会他们的时序。一个中断会保持挂起状态直到应用软件清除它。

状态中断拥有最高优先级。在报文中断中，报文对象的中断优先级随着报文编号的增加而降低

报文中断通过清除报文对象的IntPnd(CAN_IFn_MCON[13])位进行清除。状态中断通过读状态寄存器清除。

中断寄存器的中断标识符Intld指示引发中断的报文对象编号。当没有中断挂起时，寄存器将保持为0。如果中断寄存器的值不为0，则有中断挂起。如果IE(CAN_CON[1])被置位，则CAN_INT中断信号被激活。中断保持激活直到中断寄存器归零（中断源被复位）或直到IE被复位

CAN_IIDR的值为0x8000指示有中断正被挂起，原因是CAN内核更新了（不是必须改变）状态寄存器（

错误中断或状态中断）。该中断具有最高优先级，应用软件可以更新（复位）状态位RxOk (CAN_STATUS[4]), TxOk (CAN_STATUS[3]) 和 LEC (CAN_STATUS[2:0])，但是软件对状态寄存器的写访问不会产生或复位中断。

其他的值表示中断源是其中一个报文对象。IntId指向挂起中断中的具有最高中断优先级的报文中断

应用软件决定改变状态寄存器是否可能导致产生中断（EIE (CAN_MCON[3]) 和 SIE (CAN_MCON[2])），以及当中断寄存器的值不等于零时中断是否需要激活（CAN控制寄存器的IE位）。即使IE被复位，中断寄存器仍将会更新。

应用软件跟踪报文中断源有2种可能方式。一种是通过查看中断寄存器的IntId位，第二种是查询中断挂起寄存器。

中断处理服务惯例为：读触发中断源的报文并在读报文的同时复位报文对象的IntPnd（ClrIntPnd(CAN_IFn_CMASK[3])）位。当IntPnd被清除，中断寄存器将指向下一个带有挂起中断的报文对象。

6.28.7.15 配置位定时

CAN位定时配置的小错误不会导致通讯立即失败，但是会显著降低CAN网络的整体性能。

很多情况下，CAN位同步可以修补CAN位定时的错误配置，而这种错误有可能仅仅是偶发性发生一个错误帧。但是，在仲裁时，当两个或更多CAN节点同时试着发送一帧，那么一个错位的采样点可能会导致其中一个传送器变成被动错误状态。

如果要分析此类分散性错误，需要对CAN节点内CAN位同步以及CAN总线上CAN节点间相互作用有详细的了解。

6.28.7.16 位时间和位速率

CAN支持的位速率低至1 Kb/s，高至1000 Kb/s。CAN网络中的每个成员都有自己的时钟发生器，通常是石英振荡器。位时间的定时参数（例如：位速率的倒数）可以单独配置给每个CAN节点，这样尽管CAN节点的振荡器周期(f_{osc})可能不同，仍可创建一个公共的位速率。

这些振荡器的频率不是绝对的稳定，温度或电压的变化或者元件性能恶化会导致一些小的变化。只要这种变化保持在指定的振荡器容差范围（d_f）内，CAN节点就能够通过重同步对不同的位速率进行补偿。

依据CAN规范，位时间分为四段：同步段，传播时间段，相位缓冲段1和相位缓冲段2（见图6.28-7）。每段由一个特定的、可编程的时间片组成（表6.28-3）。时间片的长度(t_q),是位时间的基本时间单元，由CAN控制器的APB时钟f_{APB}和位定时寄存器（CAN_BTR）的BRP位定义：(CAN_BTIME[5:0])：t_q = BRP / f_{APB}。

同步段Sync_Seg，是位时间的一部分，是CAN总线电平跳变边沿发生的地方。Sync_Seg之外发生跳变边沿和Sync_Seg之间的距离称为边沿的相位误差。传播时间段Prop_Seg用于补偿CAN网络内的物理延时时间。相位缓冲段Phase_Seg1和Phase_Seg2围绕着采样点。（重）同步跳转宽度（SJW）定义了重同步将采样点在相位缓存段定义的范围内可能移动的距离，以此用于补偿边沿相位误差。

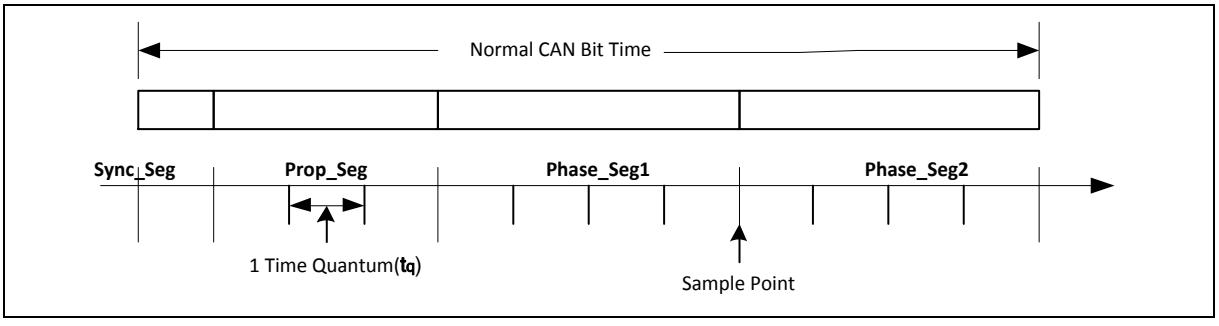


图 6.28-7 位时间

参数	范围	备注
BRP	[1..32]	定义时间片的长度t_q
Sync_Seg	1 t_q	固定长度，CAN总线输入到APB的同步
Prop_Seg	[1..8] t_q	补偿物理延时时间
Phase_Seg1	[1..8] t_q	同步功能会将此值暂时延长
Phase_Seg2	[1..8] t_q	同步功能会将此值暂时缩短
SJW	[1..4] t_q	不能比任一个相位缓存段长
该表格描述了CAN协议要求的最小可编程范围		

表 6.28-3 CAN 位时间参数

一个给定的位速率可以有不同的位时间配置，但是对于CAN网络的功能来说，物理延时时间和振荡器容错范围必须考虑。

6.28.7.17 传播时间段

该部分位时间用于补偿网络的物理延时时间。这些延时是由总线上的信号传播时间和CAN节点的内部延时时间组成。

CAN总线上，CAN节点间会发生相位出错的问题，这是由两个节点间的传播时间造成的。CAN协议非破坏性的逐位仲裁和CAN 接收器发出的显性响应位，要求发送位流的CAN节点必须同时能够接收其他CAN 节点发送的显性位。图6.28-8表示连两个CAN节点间的相位移动和传播时间。

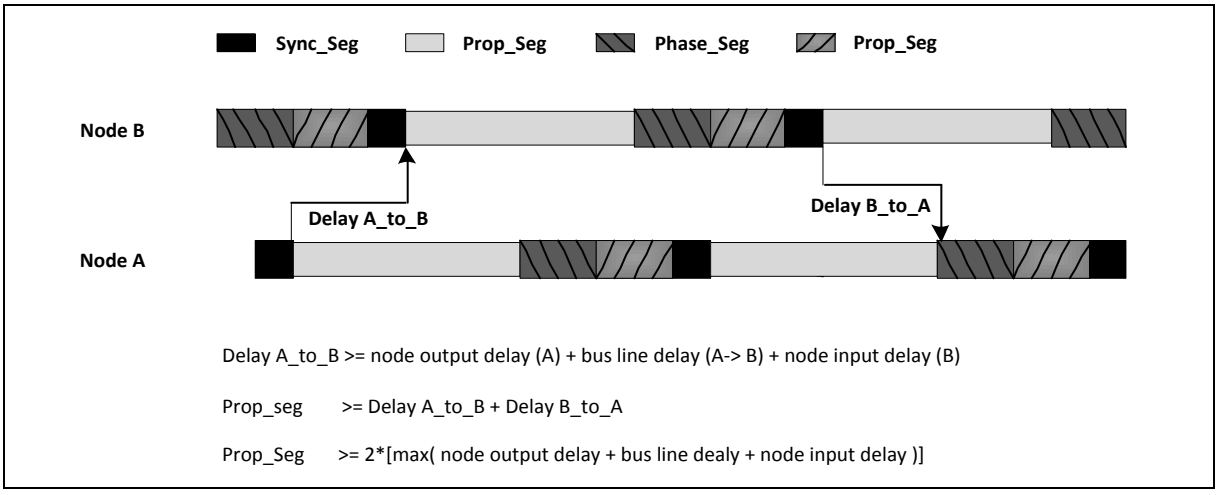


图 6.28-8 传播时间段

该示例中，节点A和B都是发送器，执行CAN 总线仲裁。节点A早于节点B发送它的起始帧位，时差小于1个位时间，节点B利用接收到的从隐性到显性的边沿与A节点同步。因为B节点是在边沿发出延时(A_to_B)后才收到，所以B的位时间段移位到与A同步。B节点发送的是一个高优先级的标识符，所以在B传送一个显性位，而A发送一个隐性位时，B将在这个标识位仲裁中胜出。节点B发送的显性位将延时(B_to_A)后到达在A。

由于振荡器容差功能，节点A采样点的实际位置可以在节点A相位缓存段标称的任何位置。所以节点B的位传送必须在Phase_Seg1开始之前到达节点A。该条件决定Prop_Seg的长度。

如果节点B传送的隐性到显性边沿在Phase_Seg1开始后才到达节点A，那么可能会出现节点A采样结果显性位变为隐性位的情况，这将导致一个位错误和错误标志并破坏当前帧。

这种错误只在两个节点都要求CAN总线进行仲裁，而两个节点的振荡器的偏差方向相反，且处于长总线两端的情况下才会发生。这是一个位定时配置错误(Prop_Seg 定义时间太短)导致偶发性总线错误的例子。

一些CAN硬件提供可选择的3次采样模式，但是C_CAN没有。在该模式下，CAN总线的输入信号通过一个低通滤波器，使用3个样本并根据大数逻辑来决定有效位值。这会导致额外的1个输入延时tq，因而需要更长的Prop_Seg。

6.28.7.18 相位缓冲及同步

相位缓冲段(Phase_Seg1 和 Phase_Seg2)和同步跳转宽度（SJW）用于振荡器误差补偿。通过同步，可以延长或缩短相位缓冲段

同步发生在从隐性到显性的边沿，目的是用来控制边沿和采样点之间的距离。

边沿检测是通过采样每个时间片的总线电平，并与前一个采样电平进行比较实现的。同步仅发生在总线电平从隐性到显性的变化时。

如果边沿发生在Sync_Seg里面，那么它是处于同步状态，否则边沿和Sync_Seg结束处之间的距离即为边沿相位误差，时间片为时间误差单位。如果边沿在Sync_Seg之前发生，相位误差为负，否则为正。

存在两种同步类型：硬同步和重同步。

硬同步在帧一开始时就会进行，而在帧内进行的是重同步

硬同步

硬同步之后，位时间将在Sync_Seg的结束处重新开始，而不用管边沿相位误差。因此硬同步强制导致硬同步的边沿处于同步段内，并重新开始位计时

位重同步

位重同步导致位时间的缩短或延长，以致采样点位置随边沿移动。

当引发重同步的边沿相位误差是正的，Phase_Seg1被延长。如果相位误差的值少于SJW，Phase_Seg1延长误差值个时间片，否则延长SJW时间片。

当造成重同步的边沿相位误差是负的，Phase_Seg2缩短。如果相位误差的值少于SJW，Phase_Seg2缩短误差值个时间片，否则缩短SJW个时间片。

当边沿相位误差的值小于或者等于SJW的设定值，硬同步和重同步作用相同。如果相位误差大于SJW，则重同步无法完全补偿相位误差，仍有误差（相位误差 - SJW）存在。

在两个采样点间只做一次同步，在边沿和采样点间最小的距离内完成，给总线电平稳定和过滤出小于(Prop_Seg + Phase_Seg1)的干扰脉冲的时间。

除了噪声脉冲，多数同步是由仲裁导致的。所有节点会“硬”同步在“领先”收发器（最先发送数据的收发器）的传送边沿，但是由于传播延时，这些节点不能达到理想的同步。“领头”发送器不一定获得仲裁，因此各接收器必须将自身同步到随后的“夺得领先”发送器(不同于之前的“领头”发生器)。同样的情况也发生在响应域，发送器和一些接收器必须要与在传送显性响应位时“夺得领先”的接收器取得同步。

当发送器和接收振荡器时钟周期的差异在同步时间内（最多10位）累加时，这些累加的差异不能比SJW长，这就限定了振荡器误差的范围。

图6.28-9为相位缓冲段是如何用于相位误差补偿的示例，其内有三张有两个连续的位定时的图。最上面一张表示同步在“late”边沿，最下面一张表示同步在“early”边沿，中间一张为无同步的参照图。

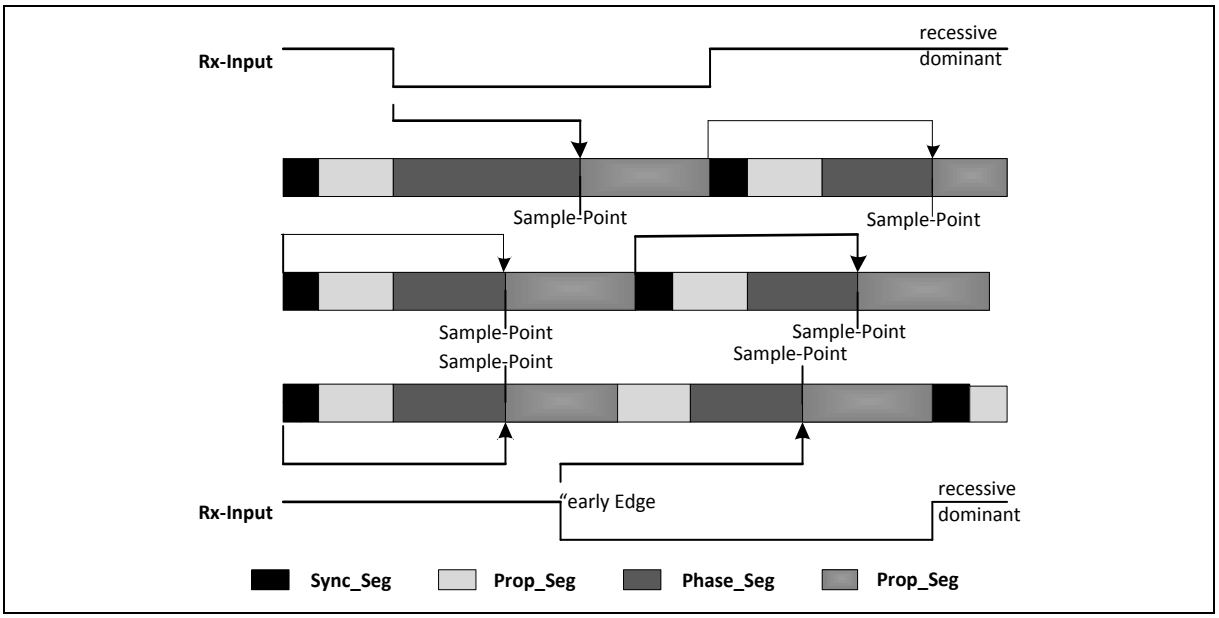


图 6.28-9 同步在“Late”及“Early”边沿

在第一个示例中隐性到显性的跳变边沿发生在Prop_Seg尾端。该边沿称之为“late”，因为它发生在Sync_Seg之后。为了应对“late”边沿，Phase_Seg1被延长，以使边沿到采样点的距离与没有边沿误差发生情况下Sync_Seg到采样点的距离保持一致。“late”边沿的相位误差小于SJW，所以能被完全补偿。

在第二个示例中为隐性到显性的跳变边沿发生在Phase_Seg2中。该边沿称之为“early”，因为它发生在Sync_Seg之前。为了应对 “early”边沿的情况，Phase_Seg2被缩短，Sync_Seg被忽略，以使边沿到采样点的距离与没有边沿误差发生的情况下Sync_Seg到采样点的距离保持一致。和上例相同，该“early”边沿的相位误差小于SJW，所以能被完全补偿。

相位缓冲段被延长或缩短只是暂时的，在下一个位时间，它们又将恢复成程序设置的值。

在以上例子中，是从CAN硬件状态机的观点去看位时序，包括位开始的位置和采样点结束的位置。当同步“early”边沿时，硬体状态机会忽略Sync_Seg，因为发生跳变的位置是在Phase_Seg2内不能再重新定义为Sync_Seg了。

图6.28-11所示为同步如何过滤短显性噪声脉冲。这些例子中噪声都在Prop_Seg尾端开始，长度为（Prop_Seg+ Phase_Seg1）。

在第一个例子中，同步跳转宽度大于或等于噪声脉冲的相位误差。因此采样点移动到干扰脉冲尾部后，采样到一个隐性总线电平。

在第二个例子中，SJW小于相位误差，所以采样点移动不够，显性噪声脉冲被当作总线电平采样。

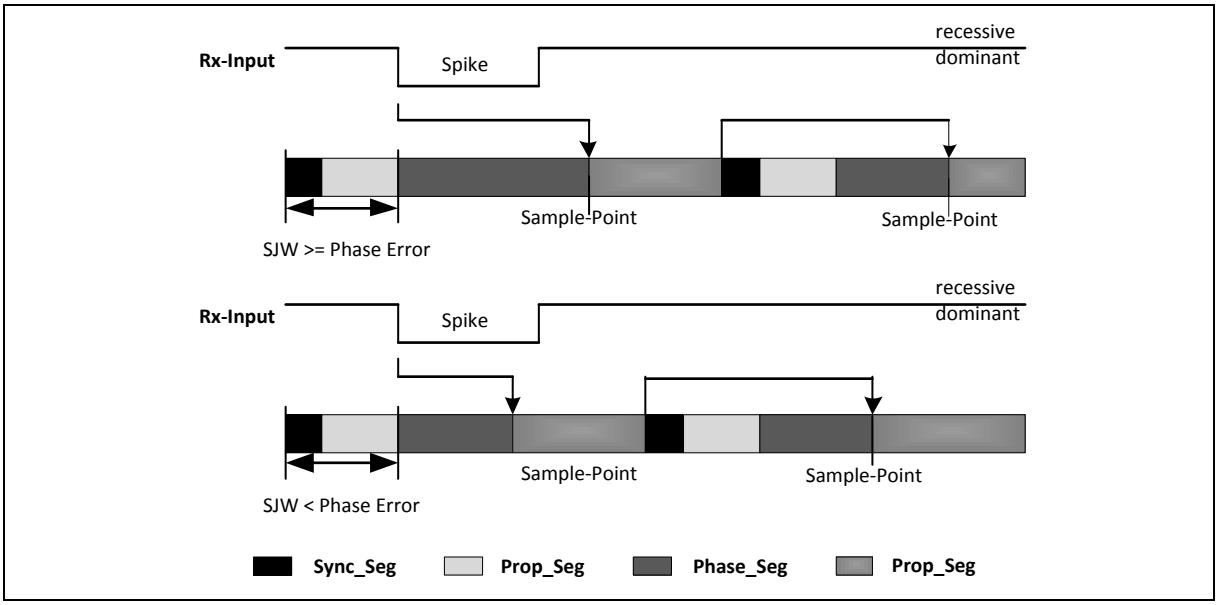


图 6.28-10 过滤显性干扰短脉冲

6.28.7.19 振荡器容差范围

CAN协议从版本1.1发展到版本1.2时，扩展了振荡器偏差范围。（版本1.0未在芯片应用中实现过）信号显性到隐性边沿进行同步的方式被放弃，仅有从隐性到显性的边沿被用于同步。协议更新到版本2.0（A和B），振荡器偏差范围没有更新。

振荡器频率f_{osc}基于标称频率f_{nom}的偏差范围d_f的公式为：

(1 – d*f*). fnom <= fosc <= (1 + d*f*) . fnom

该值取决于Phase_Seg1, Phase_Seg2的比例，SJW和位时间。最大偏差值df由两个条件决定（两个都必须满足）：

$$I: df <= \frac{\min(\text{Phase_Seg1}, \text{Phase_Seg2})}{2 * (13 * \text{bit_time} - Phase_Seg2}$$

$$II: df <= \frac{\text{SJW}}{20 * \text{bit_tim}}$$

注:这些条件基于APB时钟 = *f* osc

需要考虑SJW可能不会大于较小相位缓存段部分，传播时间段则会限制可被用于相位缓存段的位时间。

当Prop_Seg = 1， Phase_Seg1 = Phase_Seg2 = SJW = 4时，允许最大可能的振荡器偏差为1.58%。本组合中传播时间段仅占位时间10%不适用于短位时间，可被用于在总线40m长度上位速率达到125k Bit/s（位时间=8us）的情况。

6.28.7.20 配置CAN协议控制器

在大多数CAN控制器，也包括C_CAN，位定时配置设置在两个寄存器字节中。Prop_Seg，Phase_Seg1 （为TSEG1（CAN_BTIME[11:8]））和Phase_Seg2（TSEG2（CAN_BTIME[14:12]））组合在一个字节中。SJW和BRP组合在另一个字节中

在这些定时寄存器中，程序必须为TSEG1, TSEG2, SJW, 和 BRP这四个成员赋值，赋的值比它们的实际功能值小1。因此程序赋值范围为[0..n-1]，而不是[1..n]。举例：SJW（功能范围为[1..4]）仅用两个位

就可以表示

因此位时间的长度（程序设定值）为[TSEG1 + TSEG2 + 3] t_q或（功能值）为[Sync_Seg + Prop_Seg + Phase_Seg1 + Phase_Seg2] t_q

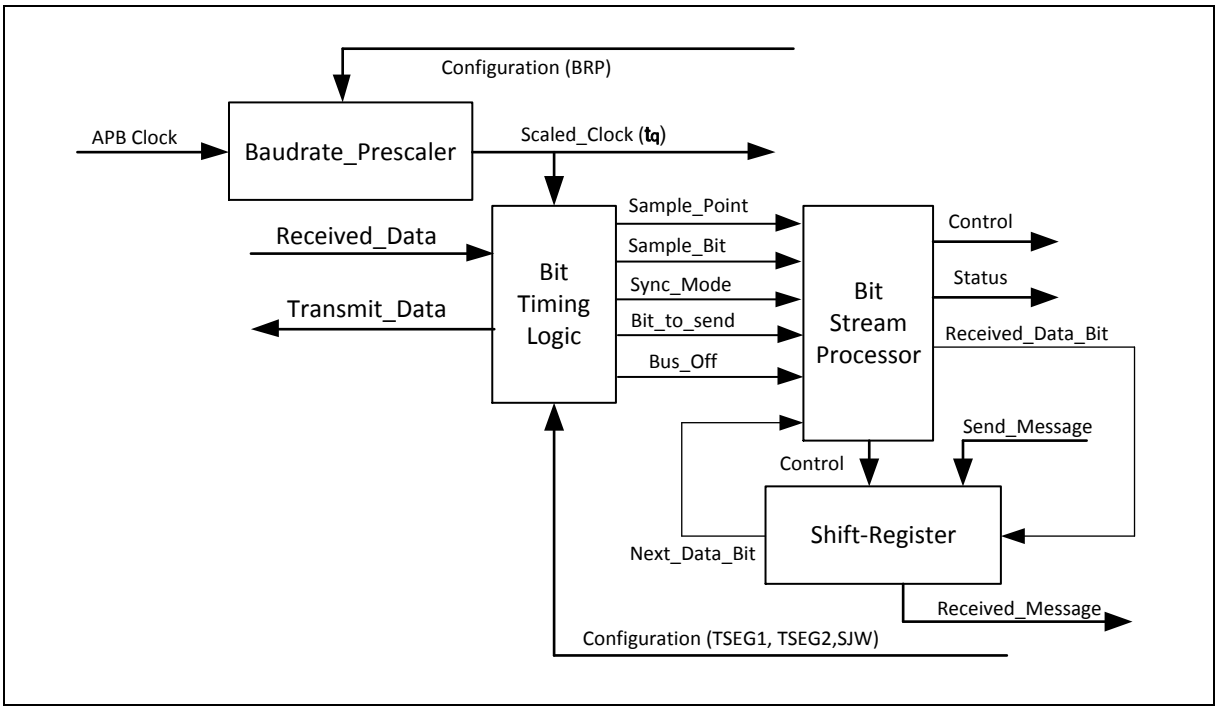


图 6.28-11 CAN 内核 CAN 协议控制器架构

根据CAN协议配置位定时寄存器中的数据。波特率分频器（用BRP配置）定义了时间片的长度，位时间的基本时间单元；位定时逻辑（由TSEG1, TSEG2, 和 SJW设置）定义了位时间中的时间片数目。

位时间的处理，采样点位置的计算和偶发性同步都是由位定时逻辑BTL(Bit Timing Logic) 控制，BTL每时间片就要计算一次。CAN协议控制器的其他部分，位流处理器BSP(Bit Stream Processor)每个位时间在采样点计算一次。

移位寄存器以串行方式发送报文，以并行方式接收报文。该寄存器的载入和移位由BSP控制。

BSP把报文变成帧，或反之。它产生和丢弃封包固定格式位，插入和提取填充位，计算和检查CRC码，执行错误管理，以及决定使用哪种同步类型。它在采样点进行计算并采样总线输入位，计算在采样点之后发送的下一位（如数据位，CRC位，填充位，错误标记或空闲）所需的时间被称为信息处理时间（IPT）。

IPT和应用有关，但是不会长于2 t_q；对C_CAN来说IPT为0 t_q。它的长度是Phase_Seg2程序设定长度的下限。在同步的情况下，Phase_Seg2可能被缩短到一个小于IPT的值，该值不会影响总线定时。

6.28.7.21 计算位定时参数

通常，位定时配置的计算从一个期望的位速率或位时间开始。位时间（1/位速率）的结果必须是APB时钟周期的整数倍。

位时间可能包括4到25个时间片，时间片的长度t_q由波特率分频器定义 t_q=(波特率预分频)/f_{apb_clk}。通过下面的步骤，几种组合都可达到期望的位时间

首先，位时间中需要定义的部分是Prop_Seg。它的长度取决于延时时间（用APB时钟测量）。对于可

扩展的CAN 总线系统，还要定义最大的总线长度和最大的节点延时。Prop_Seg的时间也要转化成时间片（向上取最接近的t_q的整数倍的值）。

Sync_Seg是1 t_q（固定）长度，留下（位时间- – Prop_Seg – 1) t_q给两个相位缓存段。如果剩下的t_q个数是偶数，相位缓存段有相同长度，Phase_Seg2 = Phase_Seg1, 如果是奇数情况，则 Phase_Seg2 = Phase_Seg1 + 1。

Phase_Seg2最小的标称值也必须要考虑一下。Phase_Seg2不能比CAN控制器的IPT短，而IPT的范围是[0..2] t_q.取决于控制器的实际硬件。

同步跳转宽度的长度要设为它的最大值，即4和Phase_Seg1中的较小值。

由振荡器偏差造成的最终配置结果由章节”振荡器偏差范围”中的方程式计算。

如果有多种配置可选，那么应该选允许最高振荡器偏差范围的配置。

带有多个不同系统时钟的CAN节点要求不同的配置来达到相同的位速率。CAN网络中的传播时间计算是基于最长延时时间的节点，整个网络只做一次传播时间计算。

CAN系统振荡器偏差范围由最低偏差容忍的节点决定。

根据计算结果，可能会要求减少总线长度或位速率，或者提高振荡器频率的稳定性，目的是寻找满足协议的CAN位定时配置。配置结果写入位定时寄存器：

(Phase_Seg2-1) & (Phase_Seg1+Prop_Seg-1) & (SynchronisationJumpWidth-1) & (Prescaler-1)

高分辨率位定时实例：

该例中，APB_CLK的时钟为10MHz，BRP (CAN_BTIME[5:0])为0，位速率为1 Mbps.

t_q	100 ns	= t_{APB_CLK}
总线驱动延时	50ns	
接收器电路延时	30ns	
总线延时 (40m)	220ns	
t_{Prop}	600ns	= 6 • t_q
t_{SJW}	100ns	= 1 • t_q
t_{TSeg1}	700ns = t_{Prop} + t_{SJW}	
t_{TSeg2}	200ns = 信息处理时间 + 1 • t_q	
$t_{Sync-Seg}$	100ns = 1 • t_q	
位时间	1000ns= $t_{Sync-Seg}$ + t_{TSeg1} + t_{TSeg2}	

APB_CLK容差0.39%= $\frac{Min(PB1, PB2)}{2 \times 13 \times (bit\ \ time - PB2))}$

$= \frac{0.1us}{2 \times 13 \times (1us - 0.2us))}$

这个例子里，位时钟参数$(2\text{-}1)_3$ & $(7\text{-}1)_4$ & $(1\text{-}1)_2$ & $(1\text{-}1)_6$, 位时钟寄存器设定为0x1600。

注: PB1/2: 表示相位缓冲段1/2

下标$(2\text{-}1)_3$表示位定时寄存器相关位的数值。

低分辨率位定时实例：

该例中，APB_CLK的时钟为2 MHz，BRP (CAN_BTIME[5:0])为1，位速率为100 Kbit/s。

t_q	1 us = 2 •t_{APB_CLK}
总线驱动延时	200ns
接收器电路延时	80ns
总线延时(40m)	220ns
t_{Prop}	1us = 1 • t_q
t_{SJW}	4us = 4 • t_q
t_{TSeg1}	5us = t_{Prop} + t_{SJW}
t_{TSeg2}	4us = Information Processing Time + 3 • t_q
$t_{Sync-Seg}$	1us = 1 • t_q
位时间	10us = $t_{Sync-Seg}$ + t_{TSeg1} + t_{TSeg2}

APB_CLK容差 1.58%

$$= \frac{Min(PB1, PB2)}{2 \times 13 \times (bit\ \ time - PB2))}$$

$$= \frac{4us}{2 \times (13 \times (10us - 4us))}$$

这个例子里，位时钟参数是$(4-1)_3$ & $(5-1)_4$ & $(4-1)_2$ & $(2-1)_6$,位时钟寄存器的值是=0x34C1。

6.28.7.22 CAN接口复位状态

在硬件复位后，C_CAN寄存器中值为“CAN寄存器映射寄存器描述”中的复位值

此外，复位后为Bus-off态，CAN_TX的输出被设置为隐性（HIGH）。CAN控制寄存器的值为0x0001（Init=‘1’）使能软件初始化。直到应用软件设置Init (CAN_CON[0])位为‘0’ C_CAN才会影响CAN总线。保存在报文RAM中的数据不受硬件复位影响。在上电后，报文RAM中的内容是不确定的。

6.28.8 寄存器映射

R:只读, **W:** 只写, **R/W:**读/写

描述	偏移量	R/W	描述	复位值
CAN 基地址： CAN_BA = 0x400A_0000				
CAN_CON	CAN_BA+0x00	R/W	CAN控制寄存器	0x0000_0001
CAN_STATUS	CAN_BA+0x04	R/W	CAN状态寄存器	0x0000_0000
CAN_ERR	CAN_BA+0x08	R	CAN错误计数寄存器	0x0000_0000
CAN_BTIME	CAN_BA+0x0C	R/W	位定时寄存器	0x0000_2301
CAN_IIDR	CAN_BA+0x10	R	中断标识符寄存器	0x0000_0000
CAN_TEST	CAN_BA+0x14	R/W	测试寄存器（寄存器表注1）	0x0000_0080
CAN_BRPE	CAN_BA+0x18	R/W	波特率预分频扩展寄存器	0x0000_0000
CAN_IFn_CREQ n = 1,2	CAN_BA+0x20 + (0x60 *(n-1))	R/W	IFn命令请求寄存器 (寄存器表注2)	0x0000_0001
CAN_IFn_CMASK n=1,2	CAN_BA+0x24 + (0x60 *(n-1))	R/W	IFn命令掩码寄存器	0x0000_0000
CAN_IFn_MASK1 n=1,2	CAN_BA+0x28 + (0x60 *(n-1))	R/W	IFn掩码1寄存器	0x0000_FFFF
CAN_IFn_MASK2 n=1,2	CAN_BA+0x2C + (0x60 *(n-1))	R/W	IFn掩码2寄存器	0x0000_FFFF
CAN_IFn_ARB1 n=1,2	CAN_BA+0x30 + (0x60 *(n-1))	R/W	IFn仲裁1寄存器	0x0000_0000
CAN_IFn_ARB2 n=1,2	CAN_BA+0x34 + (0x60 *(n-1))	R/W	IFn仲裁2寄存器	0x0000_0000
CAN_IFn_MCON n=1,2	CAN_BA+0x38 + (0x60 *(n-1))	R/W	IFn报文控制寄存器	0x0000_0000
CAN_IFn_DAT_A1 n=1,2	CAN_BA+0x3C + (0x60 *(n-1))	R/W	IFn数据A1寄存器 (寄存器表注3)	0x0000_0000
CAN_IFn_DAT_A2 n=1,2	CAN_BA+0x40 + (0x60 *(n-1))	R/W	IFn数据A2寄存器 (寄存器表注3)	0x0000_0000
CAN_IFn_DAT_B1 n=1,2	CAN_BA+0x44 + (0x60 *(n-1))	R/W	IFn数据B1寄存器 (寄存器表注 3)	0x0000_0000
CAN_IFn_DAT_B2 n=1,2	CAN_BA+0x48 + (0x60 *(n-1))	R/W	IFn数据B2寄存器 (寄存器表注3)	0x0000_0000
CAN_TXREQ1	CAN_BA+0x100	R	发送请求寄存器1	0x0000_0000
CAN_TXREQ2	CAN_BA+0x104	R	发送请求寄存器2	0x0000_0000

CAN_NDAT1	CAN_BA+0x120	R	新数据寄存器1	0x0000_0000
CAN_NDAT2	CAN_BA+0x124	R	新数据寄存器2	0x0000_0000
CAN_IPND1	CAN_BA+0x140	R	中断挂起寄存器1	0x0000_0000
CAN_IPND2	CAN_BA+0x144	R	中断挂起寄存器2	0x0000_0000
CAN_MVLD1	CAN_BA+0x160	R	报文有效寄存器1	0x0000_0000
CAN_MVLD2	CAN_BA+0x164	R	报文有效寄存器2	0x0000_0000
CAN_WU_EN	CAN_BA+0x168	R/W	唤醒使能控制寄存器	0x0000_0000
CAN_WU_STATUS	CAN_BA+0x16C	R/W	唤醒状态寄存器	0x0000_0000

注

1. 0x00 & 0br0000000, 其中R表示CAN_RX的当前值
2. IFn: 两组报文接口寄存器– IF1和IF2有相同的功能
3. An/Bn: 两组数据寄存器 – A1, A2和B1, B2.
4. CAN_BA, x = 0 或 1

CAN 寄存器表中各个位的功能

地址偏移量	寄存器名	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
00h	CAN_CON	Reserved								Test	CCE	DAR	Res	EIE	SIE	IE	Init
04h	CAN_STATUS	Reserved								BOff	EWarn	EPass	RxOk	TxOk	LEC		
08h	CAN_ERR	RP	REC6-0							TEC7-0							
0Ch	CAN_BTIME	Res	TSeg2			TSeg1				SJW			BRP				
10h	CAN_IIDR	IntId15-8								IntId7-0							
14h	CAN_TEST	Reserved								Rx	Tx1	Tx0	LBack	Silent	Basic	Reserved	
18h	CAN_BRPE	Reserved												BRPE			
20h	CAN_IF1_CREQ	Busy	Reserved									Message Number					
24h	CAN_IF1_CMASK	Reserved								WR/RD	Mask	Arb	Control	ClrIntPnd	TxRqst/	Data A	Data B
28h	CAN_IF1_MASK1	Msk15-0															
2Ch	CAN_IF1_MASK2	MXtd	MDir	Res	Msk28-16												
30h	CAN_IF1_ARB1	ID15-0															
34h	CAN_IF1_ARB2	MsgVal	Xtd	Dir	ID28-16												

地址偏移	寄存器名	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
38h	CAN_IF1_MCON	NewDat	MsgLst	IntPnd	UMask	TxIE	RxIE	RmtEn	TxRqst	EoB	Reserved			DLC3-0			
3Ch	CAN_IF1_DAT_A1	Data(1)								Data(0)							
40h	CAN_IF1_DAT_A2	Data(3)								Data(2)							
44h	CAN_IF1_DAT_B1	Data(5)								Data(4)							
48h	CAN_IF1_DAT_B2	Data(7)								Data(6)							
80h	CAN_IF2_CREQ	Busy	Reserved							Message Number							
84h	CAN_IF2_CMASK	Reserved								WR/RD	Mask	Arb	Control	ClrIntPnd	TxRqst/	Data A	Data B
88h	CAN_IF2_MASK1	Msk15-0															
8Ch	CAN_IF2_MASK2	MXtd	MDir	Res.	Msk28-16												
90h	CAN_IF2_ARB1	ID15-0															
94h	CAN_IF2_ARB2	MsgVal	Xtd	Dir	ID28-16												
98h	CAN_IF2_MCON	NewDat	MsgLst	IntPnd	UMask	TxIE	RxIE	RmtEn	TxRqst	EoB	Reserved			DLC3-0			
9Ch	CAN_IF2_DAT_A1	Data(1)								Data(0)							

地址偏移	寄存器名	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
A0h	CAN_IF2_DAT_A2	Data(3)								Data(2)							
A4h	CAN_IF2_DAT_B1	Data(5)								Data(4)							
A8h	CAN_IF2_DAT_B2	Data(7)								Data(6)							
100h	CAN_TXREQ1	TxRqst16-1															
104h	CAN_TXREQ2	TxRqst32-17															
120h	CAN_NDAT1	NewDat16-1															
124h	CAN_NDAT2	NewDat32-17															
140h	CAN_IPND1	IntPnd16-1															
144h	CAN_IPND2	IntPnd32-17															
160h	CAN_MVLD1	MsgVal16-1															
164h	CAN_MVLD2	MsgVal32-17															
168h	CAN_WU_EN	Reserved															WAKUP_EN
16Ch	CAN_WU_STATUS	Reserved															WAKUP_STS
170h	CAN_RAM_CEN	Reserved															RAM_CEN
Others	Reserved	Reserved															

表 6.28-4 CAN 寄存器中每个位的功能

注:读保留位都为‘0’，除了IFn Mask 2中保留位的值为‘1’.

Res. = 保留

6.28.9 寄存器描述

C_CAN 占用256字节的地址空间。这些寄存器按照16-位寄存器安排。

两组接口寄存器(IF1和IF2)控制软件对报文RAM的访问，它们是为报文RAM传输提供的缓存，避免软件访问和报文接收/发送之间发生冲突。

CAN控制寄存器(CAN_CON)

寄存器	偏移量	R/W	描述	复位值
CAN_CON	CAN_BA+0x00	R/W	CAN控制寄存器	0x0000_0001

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Test	CCE	DAR	Reserved	EIE	SIE	IE	Init

位	描述	
[31:8]	Reserved	保留
[7]	Test	**测试模式使能** 0= 正常操作模式 1= 测试模式
[6]	CCE	**配置改变使能** 0= 禁止写访问到位定时寄存器 1= 允许写访问到位定时寄存器(CAN_BTIME)(当Init(CAN_CON[0]) = 1)
[5]	DAR	**禁止自动重传** 0= 使能被干扰的报文自动重传 1= 禁止自动重传
[4]	Reserved	保留
[3]	EIE	**错误中断使能** 0= 禁止 -错误状态不会产生中断 1= 使能 - 改变状态寄存器的BOff (CAN_STATUS[7]) 或 EWarn (CAN_STATUS[6])位将产生中断
[2]	SIE	**状态改变中断使能** 0= 禁止 - 状态改变不会产生中断 1= 使能 - 当一条报文传输成功或一个CAN总线错误被检测到时将产生中断
[1]	IE	**模块中断使能** 0= 禁止 1= 使能
[0]	Init	**Init初始化**

		0= 正常操作 1= 初始化开始

注:不能通过设置或复位Init(CAN_CON[0])位来缩短bus-off修复时序(参见CAN规范REV.2.0)。如果设备进入bus-off状态，它将按照自己的步调设置Init位，停止所有总线的活动。一旦Init被CPU清除，设备将在恢复正常操作之前等待129个总线空闲（129*11个连续的隐性位）。在bus-off修复时序的最后，错误管理计数器将被复位。

在复位Init之后的等待过程中，每次检测到11个连续的隐性位，一个Bit0Error码就会写入状态寄存器，使能CPU立即去检查CAN总线是否被显性或连续的干扰困在，从而监控busoff修复时序的进程。

CAN状态寄存器(CAN_STATUS)

寄存器	偏移量	R/W	描述	复位值
CAN_STATUS	CAN_BA+0x04	R/W	CAN状态寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
BOff	EWarn	EPass	RxOK	TxOK	LEC		

位	描述	
[31:8]	Reserved	保留
[7]	BOff	**Bus-Off状态（只读）** 0= CAN模块没有处于Bus-Off状态 1= CAN模块处于Bus-Off状态
[6]	EWarn	**错误警告状态（只读）** 0= 所有错误计数器的数值都在96次错误警告以下 1= EML中至少有一个错误计数器达到了96次错误警告
[5]	EPass	**被动错误（只读）** 0= Can 内核处于主动错误状态 1= CAN内核处于CAN规范定义的被动错误状态
[4]	RxOK	**成功接收一条报文** 0= 自上次该位被CPU复位后,没有报文被成功接收到。CAN内核不能复位该位 1= 自上次该位被CPU复位后, 一条报文被成功接收到。(独立于接收过滤的结果)
[3]	TxOK	**成功发送一条报文** 0= 自上次该位被CPU复位后,没有报文被成功发送。CAN内核不能复位该位 1= 自上次该位被CPU复位后, 一条报文被成功发送。
[2:0]	LEC	**上一次的错误码（最近一次出现在CAN总线上的错误的类型）** LEC中的数值代表最近一次出现在CAN总线上错误的类型。当报文被成功传送，并且没有任何错误，LEC中的数值将被清零。 '7'为没有用到的代码， CPU 写入该数值到LEC用于检查是否有更新。下表详细描述错误代码的意义。

错误码	意义

0	无错误
1	无错误
2	填充错误：连续超过5个相同的位出现在接收报文中，这是不允许的。
3	格式错误：收到的帧中固定格式的部分出现了错误格式。
4	应答错误：CAN内核传送的报文没有被另一个节点应答。
5	Bit1错误：在发送一条报文的过程中（除仲裁域之外），设备想发送一个隐形位（逻辑值为‘1’的位），但是监测到总线的值为显性。.
6	Bit0错误：在发送一条报文的过程中（或应答位，或主动错误标志，或超载标志），设备想发送一个显性位（逻辑值为‘0’的位），但是监测到总线的值为隐性。在bus-off恢复的过程中，每检测到一个连续的11位隐性位，该状态设置一次。使能CPU监控bus-off恢复序列的进程。（提示总线没有被显性或连续的干扰困住）
7	CRC错误：收到的报文CRC检查和不正确，发过来的报文中接收的CRC和计算收到数据得到的CRC不一致。

表 6.28-5 错误码

状态中断

状态中断由BOff (CAN_STATUS[7]) 和EWarn(CAN_STATUS[6]) （错误中断）位或RxOK (CAN_STATUS[4])，TxOK (CAN_STATUS[3])和LEC (CAN_STATUS[2:0])（状态改变中断）位产生（假设CAN寄存器中相应的使能位被置位了）。改变EPass(CAN_STATUS[5])位或写入数据到RxOK，TxOK或LEC的不会产生状态中断。

如果该中断挂起，读状态寄存器将清除中断寄存器中的中断状态值（8000h）。

CAN错误计数寄存器(CAN_ERR)

寄存器	偏移量	R/W	描述	复位值
CAN_ERR	CAN_BA+0x08	R	CAN错误计数寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
RP	REC						
7	6	5	4	3	2	1	0
TEC							

位	描述	
[31:16]	Reserved	保留
[15]	RP	**接收错误认可** 0 = 接收错误计数器在被动错误标准以下 1 = 接收错误计数器已经达到CAN规范定义的被动错误标准
[14:8]	REC	**接收错误计数器** 接收错误计数器的当前状态。值到0到127之间。
[7:0]	TEC	**发送错误计数器** 发送错误计数器的当前状态。值到0到255之间。

位定时寄存器(CAN_BTIME)

寄存器	偏移量	R/W	描述	复位值
CAN_BTIME	CAN_BA+0x0C	R/W	位定时寄存器	0x0000_2301

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved	TSeg2			TSeg1			
7	6	5	4	3	2	1	0
SJW		BRP					

位	描述	
[31:15]	Reserved	保留
[14:12]	TSeg2	**采样点后的时间段** 0x0-0x7: TSeg2有效值为[0 … 7]。该位硬件实际解释值比程序设定值多1
[11:8]	TSeg1	**采样点前减Sync_Seg的时间段** 0x1-0x0f: TSeg1有效值为[1 … 15]. 该位硬件实际解释值比程序设定值多1
[7:6]	SJW	**(重)同步跳转宽度** 0x0-0x3: 有效值为[0 … 3]. 该位硬件实际解释值比程序设定值多1
[5:0]	BRP	**波特率预分频器** 0x01-0x3f: 该值用于振荡器频率除频产生位时间片。位时间是时间片的倍数累积。波特率分频器的有效值为[0 … 63]. 该位硬件实际解释值比程序设定值多1

注：复位值为0x2301，对应的模块时钟APB_CLK为8 MHz，C_CAN的位速率为500kBit/s。这些寄存器只在CCE (CAN_CON[6])位和CAN (CAN_CON[0])控制寄存器中Init位被置位后才可写。

中断标识寄存器(CAN_IIDR)

寄存器	偏移量	R/W	描述	复位值
CAN_IIDR	CAN_BA+0x10	R	中断标识寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
IntId							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
IntId							

位	描述	
[15:0]	**IntId**	**中断标识符（提示中断源）** 如果几个中断同时挂起，CAN 中断寄存器将忽略中断挂起时序而指向拥有最高优先级的挂起中断。中断将保持挂起直到应用软件清除它。如果IntId不为0x0000且IE(CAN_IFn_MCON[1])置位，则到IEC的IRQ中断信号会被激活。中断保持激活状态直到IntId恢复为0x0000（引起中断的原因已复位）或直到IE复位。 状态中断拥有最高优先级。在报文中断中，报文对象的优先级随着报文编号的增加而降低。 一个报文中断通过清除该报文对象的IntPnd(CAN_IFn_MCON[13])位清除。状态中断通过状态寄存器清除

IntId Value	意义
0x0000	没有中断挂起。
0x0001-0x0020	造成中断的报文对象编号
0x0021-0x7FFF	未使用
0x8000	状态中断
0x8001-0xFFFF	未使用

表 6.28-6 中断源

测试寄存器 (CAN_TEST)

寄存器	偏移量	R/W	描述	复位值
CAN_TEST	CAN_BA+0x14	R/W	测试寄存器 (寄存器映射标注1)	0x0000_0080

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Rx	Tx		LBack	Silent	Basic	Reserved	

位	描述	
[31:8]	**Reserved**	保留
[7]	**Rx**	**监测CAN_RX管脚的当前实际值（只读）*注1** 0 = CAN总线为显性(CAN_RX = ‘1’). 1 = CAN总线为隐性(CAN_RX = ‘0’).
[6:5]	**Tx**	**Tx[1:0]: 控制CAN_TX 管脚** 00 = 复位值，CAN_TX由CAN内核控制 01 = 采样点监测CAN_TX管脚 10 = CAN_TX管脚驱动一个显性值(‘0’) 11 = CAN_TX管脚驱动一个隐性值(‘1’)
[4]	**LBack**	**回环模式** 0= 回环模式禁止 1= 回环模式使能
[3]	**Silent**	**静默模式** 0 = 正常操作 1 = 模块工作在静默模式
[2]	**Basic**	**基本模式** 0 = 基本模式禁止 1 = IF1 寄存器用作TX Buffer，IF2寄存器用作RX Buffer.
[1:0]	**Reserved**	保留

复位值：0000 0000 R000 0000 b (R: RX管脚的当前值)

注:通过设置CAN控制寄存器的Test位来使能测试寄存器的写访问。不同的测试功能可以组合使用，但是Tx[1-0] (CAN_TEST[6:5]) “00”会干扰报文传输。

波特率预分频扩展寄存器 (CAN_BRPE)

寄存器	偏移量	R/W	描述	复位值
CAN_BRPE	CAN_BA+0x18	R/W	波特率预分频扩展寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved				**BRPE**			

位	描述	
[31:4]	**Reserved**	保留
[3:0]	**BRPE**	**BRPE: 波特率预分频扩展** 0x00-0x0F：通过编程BRPE，波特率分频可以被扩展到1023。该位硬件实际解释值比程序计算BRPE (MSBs) 和 BTIME (LSBs)的值要多1。

报文接口寄存器组

模块共有2组接口寄存器，用于控制CPU访问报文RAM。接口寄存器通过缓存传输避免CPU访问报文RAM时与CAN报文发送和接收发生冲突。一个完整的报文对象或部分报文对象在报文RAM和IFn报文缓存寄存器间只需一次传输就可以完成。

除基本测试模式外，2组接口寄存器的功能是相同的。它们可以这样用：一组寄存器用于传送数据到报文RAM，另一组用于接收来自报文RAM的数据，它们可以互相中断。下表6.28-7（IF1和IF2 报文接口寄存器组）提供了2组接口寄存器的概述。

每组接口寄存器由报文缓存寄存器构成，分别由各自的命令寄存器控制。命令掩码寄存器设定数据传输的方向和报文对象的哪个部分将要被传输。命令请求寄存器用于选择报文RAM中的一个报文对象作为传输的目标或来源，并开始执行命令掩码寄存器中指定的命令

地址	IF1寄存器组	地址	IF2寄存器组
CAN_BA+0x20	IF1命令请求	CAN_BA+0x80	IF2命令请求
CAN_BA+0x24	IF1命令掩码	CAN_BA+0x84	IF2命令掩码
CAN_BA+0x28	IF1掩码1	CAN_BA+0x88	IF2掩码1
CAN_BA+0x2C	IF1 掩码2	CAN_BA+0x8C	IF2掩码2
CAN_BA+0x30	IF1仲裁1	CAN_BA+0x90	IF2仲裁1
CAN_BA+0x34	IF1仲裁2	CAN_BA+0x94	IF2仲裁2
CAN_BA+0x38	IF1报文控制	CAN_BA+0x98	IF2报文控制
CAN_BA+0x3C	IF1数据A1	CAN_BA+0x9C	IF2数据A1
CAN_BA+0x40	IF1数据A2	CAN_BA+0xA0	IF2数据A2
CAN_BA+0x44	IF1数据B1	CAN_BA+0xA4	IF2数据B1
CAN_BA+0x48	IF1 数据B2	CAN_BA+0xA8	IF2数据B2

表 6.28-7 IF1 和 IF2 报文接口寄存器

IFn命令请求寄存器 (CAN_IFn_CREQ)

寄存器	偏移量	R/W	描述	复位值
CAN_IFn_CREQ	CAN_BA+0x20 + (0x60 *(n-1))	R/W	IFn命令请求寄存器 (寄存器表注2)	0x0000_0001

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Busy	Reserved						
7	6	5	4	3	2	1	0
Reserved		Message Number					

位	描述	
[31:16]	Reserved	保留
[15]	Busy	**忙标志** 0 = 读/写动作已经完成 1 = 写到IFn命令请求寄存器的动作正在进行。该位只能由软件读取。
[14:6]	Reserved	保留
[5:0]	Message Number	**报文编号** 0x01-0x20：有效的报文号，报文RAM中被选择用于数据传输的报文对象 0x00：非有效的报文号，视同0x20 0x21-0x3f：非有效的报文号，视同0x01-0x1F

一旦应用软件写报文编号到命令请求寄存器，报文传输就开始了。伴随着这个写操作，Busy位(CAN_IFn_CREQ[15])将自动被置位来通知CPU传输正在进行中。在等待3到6个APB_CLK 周期后，接口寄存器和报文RAM之间的传输完成了。Busy位被清除。

注意：当写入到命令请求寄存器中的报文编号为非有效值时，报文号将被解释转换为一个有效的值，对应报文对象也将会被传输。

IFn命令掩码寄存器(CAN_IFn_CMASK)

IFn命令掩码寄存器指定传输方向和选择哪个IFn报文缓存寄存器作为数据传输的源或目标。

寄存器	偏移量	R/W	描述	复位值
CAN_IFn_CMASK	CAN_BA+0x24 + (0x60 *(n-1))	R/W	IFn命令掩码寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
WR/RD	Mask	Arb	Control	ClrIntPnd	TxRqst/NewDat	DAT_A	DAT_B

位	描述	
[31:8]	**Reserved**	保留
[7]	**WR/RD**	**写/读模式** 0 = 读：将命令请求寄存器中指定地址的报文对象传输到选中的报文缓存寄存器中 1 = 写：将选中的报文缓存寄存器的数据传送到命令请求寄存器中指定地址的报文对象中
[6]	**Mask**	**访问掩码位** 写操作 0= 掩码位不变 1= 传输标识符 Mask + MDir + MXtd 到报文对象 读操作： 0= 掩码位不变 1= 传输标识符 Mask + MDir + MXtd 到IFn报文缓存寄存器
[5]	**Arb**	**访问仲裁位** 写操作： 0= 仲裁位不变 1= 传输 Identifier + Dir (CAN_IFn_ARB2[13]) + Xtd (CAN_IFn_ARB2[14]) + MsgVal (CAN_IFn_ARB2[15])到报文对象 读操作： 0= 仲裁位不变 1= 传输 Identifier + Dir (CAN_IFn_ARB2[13]) + Xtd (CAN_IFn_ARB2[14]) + MsgVal (CAN_IFn_ARB2[15])到IFn报文缓存寄存器
[4]	**Control**	**访问控制位** 写操作：

		0= 控制位不变 1= 传输控制位到报文对象 读操作： 0= 控制位不变 1= 传输控制位到IFn报文缓存寄存器
[3]	**ClrIntPnd**	**清除中断挂起位：** 写操作： 当写报文对象时，该位忽略。 读操作： 0=IntPnd(CAN_IFn_MCON[13])位保持不变 1= 清除报文对象中的IntPnd位
[2]	**TxRqst/NewDat**	**写操作时访问传送请求位** 0= TxRqst位不变 1= 设置TxRqst位 **注意：** 如果传送是通过设置IFn命令掩码寄存器中的TxRqst/NewDat位请求进行的，则 IFn报文控制寄存器中的TxRqst位将被忽略。 读操作时访问New Data位 0= NewDat位保持不变 1= 清除报文对象中的NewDat位 **注意：** 可通过复位控制位IntPnd和NewDat实现对一个报文对象的读取。传送到IFn报文控制寄存器的这些值总是反映其状态（在复位这些位之前）。
[1]	**DAT_A**	**访问数据字节[3：0]** 写： 0= 数据[3:0]未变 1= 写传输数据字节[3:0]到报文对象 读： 0= 数据字节[3:0]未变 1= 传送数据字节[3:0]到IFn报文缓存寄存器
[0]	**DAT_B**	**访问数据字节[7：4]** 写： 0= 数据[7:4]未变 1= 写传输数据字节[7:4]到报文对象 读： 0= 数据字节[7:4]未变 1= 传送数据字节[7:4]到IFn报文缓存寄存器

IFn掩码1寄存器(CAN_IFn_MASK1)

寄存器	偏移量	R/W	描述	复位值
CAN_IFn_MASK1	CAN_BA+0x28 + (0x60 *(n-1))	R/W	IFn掩码1寄存器	0x0000_FFFF

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Msk							
7	6	5	4	3	2	1	0
Msk							

位	描述	
[31:16]	Reserved	保留
[15:0]	Msk	标识符掩码15-0 0= 对应的报文对象标识符位不用于接收过滤 1= 对应的报文对象标识符位用于接收过滤

IFn掩码2寄存器(CAN_IFn_MASK2)

寄存器	偏移量	R/W	描述	复位值
CAN_IFn_MASK2	CAN_BA+0x2C + (0x60 *(n-1))	R/W	IFn掩码2寄存器	0x0000_FFFF

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
MXtd	MDir	Reserved	Msk				
7	6	5	4	3	2	1	0
Msk							

位	描述	
[31:16]	**Reserved**	保留
[15]	**MXtd**	**掩码扩展标识符** 0= 扩展标识符位（IDE）对于接收过滤没有影响 1= 扩展标识符位（IDE）用于接收过滤 **注意**：报文对象采用11-位（标准）标识符时，接收到的数据帧的标识符将会写入到ID28 - ID18 (CAN_IFn_ARB2[12:2]) 。 所有这些位以及掩码位 Msk28 - Msk18 (CAN_IFn_MASK2[12:2])，都要列入到接收过滤的考虑中。
[14]	**MDir**	**掩码报文方向** 1= 报文方向(Dir (CAN_IFn_ARB2[13]))位用于接收过滤 0= 报文方向位对于接收过滤没有影响
[13]	**Reserved**	保留
[12:0]	**Msk**	**标识符掩码28-16** 0= 报文对象标识符的相应位不用于接收过滤的匹配操作 1= 相应的标识符用于接收过滤

IFn仲裁1寄存器(CAN_IFn_ARB1)

寄存器	偏移量	R/W	描述	复位值
CAN_IFn_ARB1	CAN_BA+0x30 + (0x60 *(n-1))	R/W	IFn仲裁1寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
ID							
7	6	5	4	3	2	1	0
ID							

位	描述	
[31:16]	Reserved	保留
[15:0]	ID	报文标识符15-0 ID28 - ID0, 29-位标识符（“扩展帧”） ID28 - ID18, 11-位标识符（“标准帧”）

IFn仲裁2寄存器(CAN_IFn_ARB2)

寄存器	偏移量	R/W	描述	复位值
CAN_IFn_ARB2	CAN_BA+0x34 + (0x60 *(n-1))	R/W	IFn仲裁2寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
MsgVal	Xtd	Dir	ID				
7	6	5	4	3	2	1	0
ID							

位	描述	
[31:16]	**Reserved**	保留
[15]	**MsgVal**	**报文有效** 0= 报文对象无效，报文处理器将忽略此报文对象 1= 报文对象有效，报文对象已配置，而且报文处理器将处理此报文对象 **注意**：在初始化过程中，复位CAN控制寄存器的Init位之前，应用软件必须复位所有未使用的报文对象的MsgVal位。如果报文对象需要修改ID28-0，控制位Xtd，Dir，或数据长度码DLC23-0，那么在ID28-0，控制位Xtd，Dir，或数据长度码DLC23-0修改之前需将此位复位。
[14]	**Xtd**	**扩展标识符：** 0 = 11位（标准）标识符用于报文对象 1 = 29位（扩展）标识符用于报文对象
[13]	**Dir**	**报文方向：** 1= 方向为发送 TxRqst 端，各自的报文对象作为数据帧被传输。在收到带有匹配标识符的远程帧时，该报文对象的TxRqst位被置位。（如果RmtEn=1） 0= 方向为接收 TxRqst端，带有该报文对象标识符的远程帧被传输。在收到带有匹配标识符的数据帧时，报文保存在该报文对象内
[12:0]	**ID**	**报文标识符28-16** ID28 - ID0, 29位标识符（“扩展帧”） ID28 - ID18, 11-位标识符（“标准帧”）

IFn报文控制寄存器(CAN_IFn_MCON)

寄存器	偏移量	R/W	描述	复位值
CAN_IFn_MCON	CAN_BA+0x38 + (0x60 *(n-1))	R/W	IFn报文控制寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
NewDat	MsgLst	IntPnd	UMask	TxIE	RxIE	RmtEn	TxRqst
7	6	5	4	3	2	1	0
EoB	Reserved			DLC			

位	描述	
[31:16]	**Reserved**	保留
[15]	**NewDat**	**新数据** 0= 自上次该位被应用软件清除后，该报文对象的数据部分还没有由报文处理器写入新数据。 1= 报文处理器或应用软件已经写了新数据到该报文对象的数据部分。
[14]	**MsgLst**	**报文丢失** 0= 自上次该位被CPU复位后无报文丢失 1= 当报文处理器保存新的一个报文到报文对象，NewDat虽然被置位，但CPU已经丢失了一条报文。 **注：** 只接收报文有效。
[13]	**IntPnd**	**中断挂起** 0= 该报文对象不是中断源 1= 该报文对象是中断源。如果没有更高优先级的中断源存在时，中断寄存器中的中断标识符将指向该报文对象。
[12]	**UMask**	**使用验收掩码** 0= 忽略掩码 1= 使用掩码(Msk28-0, MXtd, and MDir)用于接收过滤 **注意：** 如果UMask位被置1，则在报文对象初始化的过程中，MsgVal(CAN_IFn_ARB2[15])位被置为1之前，报文对象的掩码位必须先被设定。
[11]	**TxIE**	**发送中断使能** **0=** IntPnd (CAN_IFn_MCON[13])在成功传送一帧后将保持不变 1= IntPnd (CAN_IFn_MCON[13])在成功传送一帧后将被置位
[10]	**RxIE**	**接收中断使能** **0=** IntPnd (CAN_IFn_MCON[13])在成功接收一帧后将保持不变

		1= IntPnd (CAN_IFn_MCON[13])在成功接收一帧后将被置位
[9]	**RmtEn**	**远程使能** 0= 在收到一远程帧后，TxRqst保持不变 1= 在收到一远程帧后，TxRqst被置位
[8]	**TxRqst**	**传送请求** 0= 该报文对象不在等待传送 1= 该报文对象已请求传送但还没有完成
[7]	**EoB**	**缓存结束** 0= 报文对象属于某FIFO缓存，但不是该FIFO缓存的最后一个报文对象。 1= 单报文对象或FIFO缓存的最后一个报文对象。 **注意：**该位用于关联两个或多个报文对象（最多32 个）组成一个FIFO 缓存。对于单个报文对象（不属于任何FIFO 缓存），该位必须始终设置为 1。
[6:4]	**Reserved**	保留.
[3:0]	**DLC**	**数据长度码** 0-8：数据帧有0-8个数据字节 9-15：数据帧有8个数据字节 **注意：**报文对象的数据长度码必须和其他节点中所有带有相同标识符的报文对象的数据长度码相同。当报文处理器保存一个数据帧时，它将把收到报文提供的数值写入DLC。 Data 0：CAN数据帧的1st数据字节 Data 1：CAN数据帧的2nd数据字节 Data 2：CAN数据帧的3rd数据字节 Data 3：CAN数据帧的4th数据字节 Data 4：CAN数据帧的5th数据字节 Data 5：CAN数据帧的6th数据字节 Data 5：CAN数据帧的7th数据字节 Data 7：CAN数据帧的8th数据字节 **注意：**Data 0字节是接收过程中，移入CAN内核的移位寄存器的第一个数据字节，Data 7字节是最后一个。当报文处理器保存一个数据帧，他将会写所有8个数据字节到报文对象中。如果数据长度码小于8，则报文对象剩下的字节将被非指定值覆盖。

IFn数据A1寄存器(CAN_IFn_DAT_A1)

寄存器	偏移量	R/W	描述	复位值
CAN_IFn_DAT_A1	CAN_BA+0x3C + (0x60 *(n-1))	R/W	IFn数据A1寄存器 (寄存器映射备注3)	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Data(1)							
7	6	5	4	3	2	1	0
Data(0)							

位	描述	
[31:16]	**Reserved**	保留
[15:8]	**Data(1)**	**数据字节1** CAN 数据帧的第2个数据字节
[7:0]	**Data(0)**	**数据字节0** CAN 数据帧的第1个数据字节

IFn数据A2寄存器(CAN_IFn_DAT_A2)

寄存器	偏移量	R/W	描述	复位值
CAN_IFn_DAT_A2	CAN_BA+0x40 + (0x60 *(n-1))	R/W	IFn 数据A2寄存器 (寄存器映射备注3)	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Data(3)							
7	6	5	4	3	2	1	0
Data(2)							

位	描述	
[31:16]	**Reserved**	保留
[15:8]	**Data(3)**	**数据字节3** CAN 数据帧的第4个数据字节
[7:0]	**Data(2)**	**数据字节2** CAN 数据帧的第3个数据字节

IFn数据B1寄存器(CAN_IFn_DAT_B1)

寄存器	偏移量	R/W	描述	复位值
CAN_IFn_DAT_B1	CAN_BA+0x44 + (0x60 *(n-1))	R/W	IFn数据B1寄存器 (寄存器映射备注3)	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Data(5)							
7	6	5	4	3	2	1	0
Data(4)							

位	描述	
[31:16]	**Reserved**	保留
[15:8]	**Data(5)**	**数据字节5** CAN 数据帧的第6个数据字节
[7:0]	**Data(4)**	**数据字节4** CAN 数据帧的第5个数据字节

IFn数据B2寄存器(CAN_IFn_DAT_B2)

寄存器	偏移量	R/W	描述	复位值
CAN_IFn_DAT_B2	CAN_BA+0x48 + (0x60 *(n-1))	R/W	IFn数据B2寄存器 (寄存器映射备注3)	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Data(7)							
7	6	5	4	3	2	1	0
Data(6)							

位	描述	
[31:16]	**Reserved**	保留
[15:8]	**Data(7)**	**数据字节7** CAN 数据帧的第8个数据字节
[7:0]	**Data(6)**	**数据字节6** CAN 数据帧的第7个数据字节

在一个CAN数据帧中，数据[0]是第一个，数据[7]是传送或接收的最后一个字节数据。在CAN的串行位流，每个字节的MSB将被先传送。

报文内存中的报文对象

在报文RAM中有32个报文对象。为了避免应用软件在访问报文RAM时与CAN报文接收发送发生冲突，CPU不能直接访问报文对象，所有访问读取都要通过IFn接口寄存器。下表 6.28 8详细列出了一个报文对象的结构。

报文对象												
UMask	Msk [28:0]	MXtd	MDir	EoB	NewDat		MsgLst	RxIE	TxIE	IntPnd	RmtEn	TxRqst
MsgVal	ID [28:0]	Xtd	Dir	DLC [3:0]	Data(0)	Data(1)	Data(2)	Data(3)	Data(4)	Data(5)	Data(6)	Data(7)

表 6.28-8 报文对象在内存中的结构

仲裁寄存器ID28-0(CAN_IFn_ARB1/2)，Xtd (CAN_IFn_ARB2[14])和Dir(CAN_IFn_ARB2[13])用于定义标识符，传出报文的传送类型以及传入报文的接收过滤（接收过滤还要用到屏蔽寄存器Msk28-0 (CAN_IFn_MASK1/2)，MXtd (CAN_IFn_MASK2[15])，和 MDir (CAN_IFn_MASK2[14]) ）。接收到的报文存储在对应标识符匹配，方向=接收（数据帧）或方向=发送（远程帧）的有效报文对象中。扩展帧只能保存在Xtd=1的报文对象中，标准帧保存在Xtd=0的报文对象中。如果一条接收到的报文（数据帧或远程帧）和1个以上的有效报文对象匹配，那么它将被保存在报文编号最低的报文对象中。

报文处理寄存器

所有的报文处理寄存器都是只读。报文处理寄存器内容（每个报文对象的TxRqst(CAN_IFn_MCON[8]), NewDat (CAN_IFn_MCON[15]), IntPnd(CAN_IFn_MCON[13]), 和 MsgVal (CAN_IFn_ARB2[15])，以及中断标示符）为报文处理器FSM提供的状态信息。

发送请求寄存器1 (CAN_TXREQ1)

这些寄存器保存32个报文对象的TxRqst位。通过读取TxRqst位，软件可以检查哪个报文对象的传送请求正在挂起。在收到一帧或者成功传送一帧后，应用软件可以通过IFn报文接口寄存器或者报文处理器，可以置位/复位指定报文对象的TxRqst位。

寄存器	偏移量	R/W	描述	复位值
CAN_TXREQ1	CAN_BA+0x100	R	发送请求寄存器1	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
TxRqst16-9							
7	6	5	4	3	2	1	0
TxRqst8-1							

位	描述	
[31:16]	Reserved	保留
[15:0]	TxRqst16-1	发送请求位16-1（所有报文对象）（只读） 0= 该报文对象没有等待传送 1= 该报文对象已有传送请求但还未完成。

发送请求寄存器2 (CAN_TXREQ2)

寄存器	偏移量	R/W	描述	复位值
CAN_TXREQ2	CAN_BA+0x104	R	发送请求寄存器2	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
TxRqst32-25							
7	6	5	4	3	2	1	0
TxRqst24-17							

位	描述	
[31:16]	Reserved	保留
[15:0]	TxRqst32-17	发送请求位32-17（所有报文对象）(只读) 0= 该报文对象没有等待传送 1= 该报文对象已有传送请求但还未完成

新数据寄存器1 (CAN_NDAT1)

这些寄存器保存32个报文对象的NewDat位。通过读取NewDat位，软件可以检查哪个报文对象的数据部分被更新了。在收到一帧或者成功传送一帧后，应用软件可以通过IFn报文接口寄存器或者报文处理器，置位/复位指定报文对象的NewDat位。

寄存器	偏移量	R/W	描述	复位值
CAN_NDAT1	CAN_BA+0x120	R	新数据寄存器 1	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
NewData16-9							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
NewData8-1							

位	描述	
[31:16]	**Reserved**	保留
[15:0]	**NewData16-1**	**新数据位16-1（所有报文对象）** 0= 自上次该标志被软件清除后，报文处理器没有写入新数据到该报文对象的数据部分 1= 报文处理器或应用软件已经写入了新数据到该报文对象的数据部分

新数据寄存器2 (CAN_NDAT2)

寄存器	偏移量	R/W	描述	复位值
CAN_NDAT2	CAN_BA+0x124	R	新数据寄存器 2	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
NewData32-25							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
NewData24-17							

位	描述	
[31:16]	**Reserved**	保留
[15:0]	**NewData32-17**	**新数据位32-17（所有报文对象）** 0= 自上次该标志被软件清除后，报文处理器没有写入新数据到该报文对象的数据部分 1= 报文处理器或应用软件已经写入了新数据到该报文对象的数据部分

中断挂起寄存器1 (CAN_IPND1)

这些寄存器保存32个报文对象的IntPnd位。通过读取IntPnd位，软件可以检查哪个报文对象的中断挂起。在收到一帧或者成功传送一帧后，应用软件可以通过IFn报文接口寄存器或者报文处理器，置位/复位指定报文对象的IntPnd位。这也将会影响中断寄存器的IntId的值。

寄存器	偏移量	R/W	描述	复位值
CAN_IPND1	CAN_BA+0x140	R	中断挂起寄存器1	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
IntPnd16-9							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
IntPnd8-1							

位	描述	
[31:16]	**Reserved**	保留
[15:0]	**IntPnd16-1**	**中断挂起位16-1(所有报文对象)** 0= 表示该报文对象不是中断源 1= 表示该报文对象是中断源

中断挂起寄存器2 (CAN_IPND2)

寄存器	偏移量	R/W	描述	复位值
CAN_IPND2	CAN_BA+0x144	R	中断挂起寄存器2	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
IntPnd32-25							
7	6	5	4	3	2	1	0
IntPnd24-17							

位	描述	
[31:16]	Reserved	保留
[15:0]	IntPnd32-17	中断挂起位32-17(所有报文对象) 0= 表示该报文对象不是中断源 1= 表示该报文对象是中断源

报文有效寄存器1 (CAN_MVLD1)

这些寄存器保存32个报文对象的MsgVal位。通过读取MsgVal位，应用软件可以检查哪个报文对象是有效的。应用软件（通过IFn报文接口寄存器）可以来置位/复位指定报文对象的MsgVal位。

寄存器	偏移量	R/W	描述	复位值
CAN_MVLD1	CAN_BA+0x160	R	报文有效寄存器1	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
MsgVal16- 9							
7	6	5	4	3	2	1	0
MsgVal8-1							

位	描述	
[31:16]	Reserved	保留
[15:0]	MsgVal16-1	报文有效位16-1（所有报文对象）（只读） 0= 该报文对象无效, 被报文处理器忽略 1= 该报文对象有效，可以被报文处理器考虑 例如：CAN_MVLD1[0]表示No.1对象是否有效，如果CAN_MVLD1[0]被置位，则报文对象No.1有效。

报文有效寄存器2 (CAN_MVLD2)

寄存器	偏移量	R/W	描述	复位值
CAN_MVLD2	CAN_BA+0x164	R	报文有效寄存器2	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
MsgVal32-25							
7	6	5	4	3	2	1	0
MsgVal24-17							

位	描述	
[31:16]	Reserved	保留
[15:0]	MsgVal32-17	报文有效位32-17（所有报文对象）（只读） 0= 该报文对象无效，被报文处理器忽略 1= 该报文对象有效，可以被报文处理器考虑 例如：.CAN_MVLD2[15]表示No.32对象是否有效，如果CAN_MVLD2[15]被置位，则报文对象No.32为有效

唤醒使能寄存器 (CAN_WU_EN)

寄存器	偏移量	R/W	描述	复位值
CAN_WU_EN	CAN_BA+0x168	R/W	唤醒使能状态寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved							WAKUP_EN

位	描述	
[31:1]	Reserved	保留
[0]	WAKUP_EN	**唤醒使能位** 0 = 禁止唤醒功能 1 = 使能唤醒功能 **注**: 当CAN_Rx管脚上有下降沿信号时,用户可唤醒系统

唤醒状态寄存器 (CAN_WU_STATUS)

寄存器	偏移量	R/W	描述	复位值
CAN_WU_STATUS	CAN_BA+0x16C	R/W	唤醒状态寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved							WAKUP_STS

位	描述	
[31:1]	Reserved	保留
[0]	WAKUP_STS	**唤醒状态** 0= 没有唤醒事件发生 1= 有唤醒事件发生了 **注：**该位可写‘0’清除

6.29 SD卡主机接口

6.29.1 概述

安全数字主机控制器(SD主机)包含DMAC单元和SD单元。DMAC单元提供SD共享缓存（128个字节）与系统内存间的交换数据DMA (直接内存访问)功能。SD单元控制SD/SDHC的接口。SDHOST控制器支持SD/SDHC并且有DMAC提供系统内存与卡之间快速的数据传输特性。

- 兼容AMBA AHB主/从接口，用于数据传输和寄存器读/写
- 支持单通道DMA
- 支持硬件分散收集功能
- 使用一个128字节共享缓存用做系统和卡之间的数据交换
- 单一时钟域AHB总线时钟(HCLK)的同步设计DMA
- 寄存器读写与数据传输的DMAC接口
- 支持SD/SDHC卡
- 两个时钟域完全异步设计，HCLK和引擎时钟，注意HCLK的频率需要比外设时钟快

	M48xID/M48xGA	M48xGC/M48xE8
安全数字主机控制器	2	1

表 6.29-1SDH 特性比较表

6.29.2 框图

SDHC框图和卡引脚分配如下：

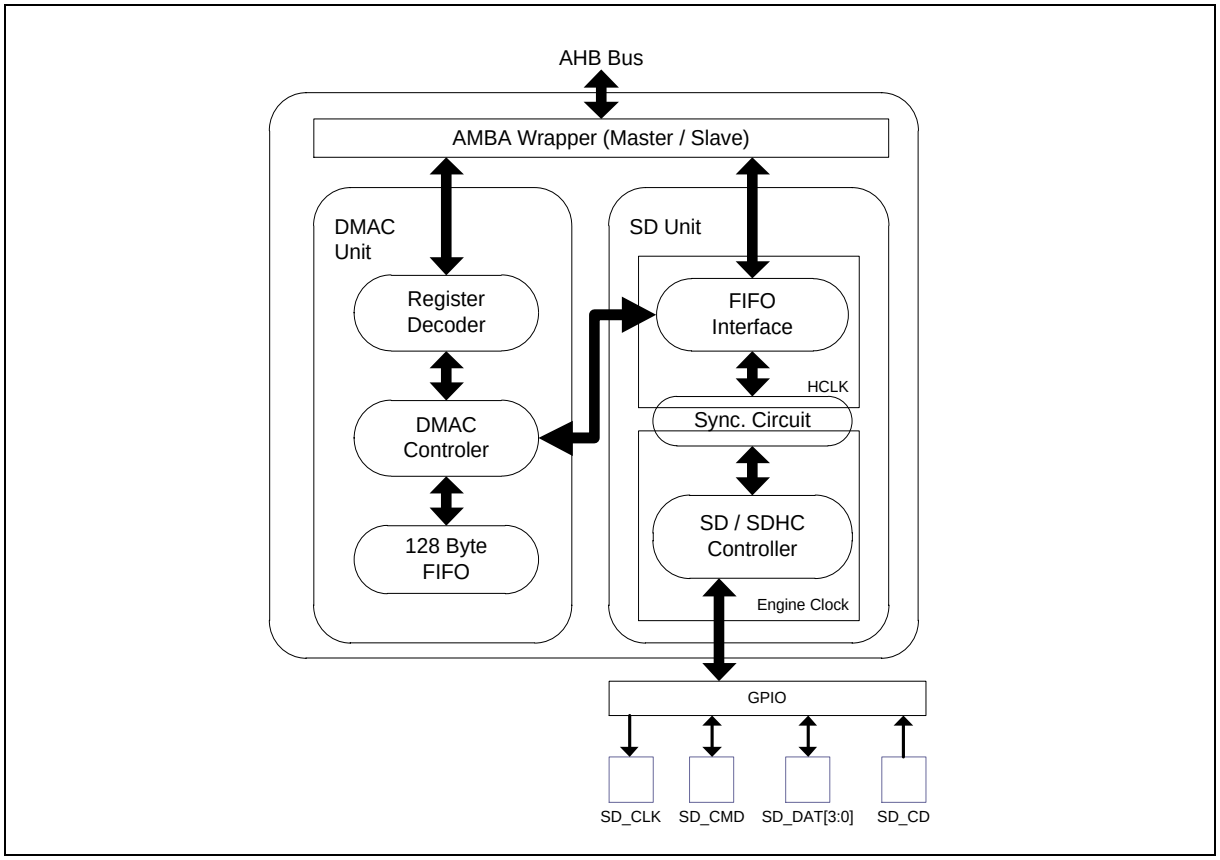


图 6.29-1 SD 主机控制器框图

6.29.3 基本配置

6.29.3.1 SD0基本配置

- 时钟源配置
  - 选择SD0时钟源寄存器SDH0SEL(CLK_CLKSEL0[21:20])
  - 选择SD0时钟分频数寄存器SDH0DIV(CLK_CLKDIV0[31:24])
  - 使能SD0时钟寄存器SDH0CKEN(CLK_AHBCLK[6])
- 复位配置
  - 复位SD0控制器寄存器SDH0RST(SYS_IPRST0[6])
- 引脚配置

组	引脚名称	GPIO	MFP
SD0	SD0_CLK	PB.1, PE.6	MFP3
	SD0_CMD	PB.0, PE.7	MFP3
	SD0_DAT0	PB.2, PE.2	MFP3

	SD0_DAT1	PB.3, PE.3	MFP3
	SD0_DAT2	PB.4, PE.4	MFP3
	SD0_DAT3	PB.5, PE.5	MFP3
	SD0_nCD	PD.13	MFP3
		PB.12	MFP9

表 6.29-2 SD0 引脚配置

6.29.3.2 SD1基本配置

- 时钟源配置
  - 选择SD1时钟源寄存器SDH1SEL(CLK_CLKSEL0[23:22])
  - 选择SD1时钟分频数寄存器SDH1DIV(CLK_CLKDIV3[31:24])
  - 使能SD1时钟寄存器SDH1CKEN(CLK_AHBCLK[17])
- 复位配置
  - 复位SD1控制器寄存器SDH1RST(SYS_IPRST0[17])
- 引脚配置

组	引脚名称	GPIO	MFP
SD1	SD1_CLK	PG.14	MFP3
		PA.4	MFP5
		PB.6	MFP7
	SD1_CMD	PG.13	MFP3
		PA.5	MFP5
		PB.7	MFP7
	SD1_DAT0	PG.12	MFP3
		PA.0, PA.8	MFP5
	SD1_DAT1	PG.11	MFP3
		PA.1, PA.9	MFP5
	SD1_DAT2	PG.10	MFP3
		PA.2, PA.10	MFP5
	SD1_DAT3	PG.9	MFP3
		PA.3, PA.11	MFP5
	SD1_nCD	PG.15	MFP3
		PA.6, PE.14	MFP5

表 6.29-3 SD1 引脚配置

6.29.4 功能描述

SD主机提供访问SD/SDHC卡的接口，该接口提供1位和4位数据总线模式。

SD控制器使用一个独立的时钟源，SDCLK，作为引擎时钟。SDCLK 可以与系统时钟HCLK完全异步。程序可以任意改变SD时钟，注意HCLK需要快于SDCLK。

6.29.4.1 基本操作

SD控制器可以产生所有类型48位的指令到SD卡，并且从SD卡获取所有类型的响应。收到响应后，响应的内容存入SDRSP0和SDRSP1。SD控制器会计算CRC7确认响应的正确性。如果CRC7错误, CDRIF (SDH_INTSTS[1])会置1且CRC7 (SDH_INTSTS[2])会清0。对于响应R1b，程序需要注意在收到响应后SD卡会在数据线DAT0产生忙信号; 程序可以通过时钟轮询检查这个状态直到它变高。对于响应R3, CRC7无效; 但是 SD 控制器仍然会计算CRC7并得到错误的结果，程序要忽略这个错误并清除CDRIF 标志(SDH_INTSTS[1])。

SD控制器包含2个状态机，指令/响应部分和数据部分。对于指令/响应部分，触发位是SDH_CTL寄存器的COEN，RIEN，R2_EN，CLK74OE和CLK8_OE位。如果程序使能所有这些位，处理优先级会是CLK74OE > COEN > RIEN/R2_EN > CLK8_OE，注意RIEN和R2_EN不能同时触发。对于数据部分，有DIEN和DOEN 可以选择。程序只能同时触发其中之一。如果 DIEN 被触发, SD 控制器会立即从DAT0数据线等待开始位，然后从SD卡得到特定数量的数据。收到数据后，SD控制器会检查CRC16的正确性；如果有错误, CDRIF (SDH_INTSTS[1])会置1且CRC16 (SDH_INTSTS[3])会清0。如果DOEN被触发，SD控制器会等待响应结束，然后发送特定数据到SD卡。数据发送之后SD控制器会从SD卡得到CRC状态并确认正确性;CRC 需要是‘010’，否则 CDRIF (SDH_INTSTS[1]) 会置 1 且 CRCSTAT (SDH_INTSTS[6:4]) 会是接收到的数值。

如果IR2_EN被触发，SD控制器会从SD卡接收响应R2 (136 位)，CRC7和结束位会丢弃。接收到的数据会放入DMA缓存，从地址偏移0x0开始。

6.29.4.2 多数据块传输

SD控制器提供多数据块传输功能 (改变 BLKLEN 来改变数据块长度)。程序可以用这个功能来加速数据传输。如果CRC7，CRC16或CRC状态错误，SD控制器会结束传输且置位CDRIF (SDH_INTSTS[1])，程序在这种状况发生时复位引擎。

在SD引擎里有一个硬件响应超时机制。程序可以设置一个24位超时值TOUT，SD控制器会根据这个值来决定什么时候超时。

6.29.4.3 DMA控制器

SD主机DMA控制器在SD主机控制器和系统内存（SRAM）和共享缓存（128个字节）之间提供DMA (直接内存访问)功能。SD主机间的DMA请求的仲裁由DMA总线管理器控制。程序只需要简单的填入开始地址和使能DMA，然后让DMA来自动控制数据传输。

DMA有128个字节的共享缓存，可以为SD主机提供多模块的数据传输。程序可以在SD主机不忙时直接访问缓存。

6.29.4.4 编程流程

这是一个简单的没有使能分散采集的 DMA 的编程例子。

1. 设置DMAEN (SDH_DMACTL[0])使能DMA
2. 在SD主机的SDH_DMASA填入相应的开始地址
3. 触发SD主机开始DMA传输
4. 等待传输完成

这是一个简单的使能分散采集的 DMA 的编程例子。

1. 设置DMAEN (SDH_DMACTL[0])使能DMA (SDH_DMACTL[0])且设置SGEN (SDH_DMACTL[3])使能分散采集功能
2. 在SD主机的SDH_DMASA填写物理地址描述符（PAD）表的开始地址
3. 当SDH_DMASA的第0位为1，不会按顺序读取PAD列表, 如果为0，会从PAD按序载入描述符。在这种功能下第一次在寄存器里写第0位是无效的，这一位要在PAD列表中设置
4. 触发SD主机开始DMA传输
5. 等待传输完成

PAD 表格式如图6.29-2所示。注意所有PAD的总共字节计数必须等于填写到SD主机的字节计数。EOF在最后的描述符中必须设置为1。

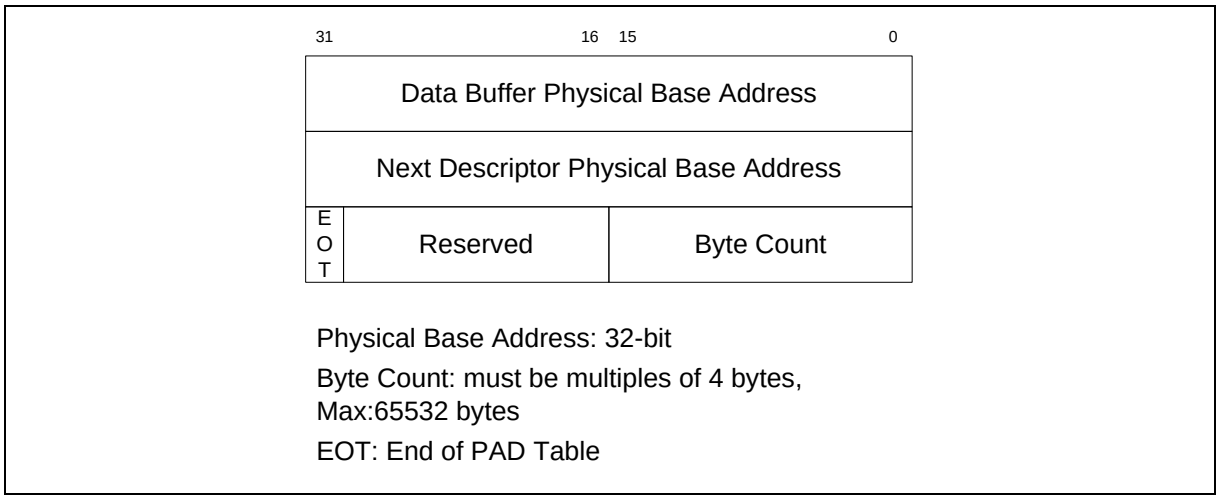


图 6.29-2 PAD (物理地址描述符) 表格式

6.29.5 寄存器映射

R：只读，**W**：只写，**R/W**：读/写

寄存器	偏移量	R/W	描述	复位值
SDH 基地址： SDHx_BA = 0x4000_D000 + (0x1000 * x) x=0,1				
SDH_FB_n n=0,1..31	SDHx_BA+0x000 + 0x4 * n	R/W	共享缓存(FIFO)	0x0000_0000
SDH_DMACTL	SDHx_BA+0x400	R/W	DMA控制和状态寄存器	0x0000_0000
SDH_DMASA	SDHx_BA+0x408	R/W	DMA传输开始地址寄存器	0x0000_0000
SDH_DMABCNT	SDHx_BA+0x40C	R	DMA传输字节数寄存器	0x0000_0000
SDH_DMAINTEN	SDHx_BA+0x410	R/W	DMA中断使能控制寄存器	0x0000_0001
SDH_DMAINTSTS	SDHx_BA+0x414	R/W	DMA中断状态寄存器	0x0000_0000
SDH_GCTL	SDHx_BA+0x800	R/W	全局控制和状态寄存器	0x0000_0000
SDH_GINTEN	SDHx_BA+0x804	R/W	全局中断控制寄存器	0x0000_0001
SDH_GINTSTS	SDHx_BA+0x808	R/W	全局中断状态寄存器	0x0000_0000
SDH_CTL	SDHx_BA+0x820	R/W	SD控制和状态寄存器	0x0101_0000
SDH_CMDARG	SDHx_BA+0x824	R/W	SD指令参数寄存器	0x0000_0000
SDH_INTEN	SDHx_BA+0x828	R/W	SD中断控制寄存器	0x0000_0A00
SDH_INTSTS	SDHx_BA+0x82C	R/W	SD中断状态寄存器	0x000X_008C
SDH_RESP0	SDHx_BA+0x830	R	SD接收响应标志寄存器0	0x0000_0000
SDH_RESP1	SDHx_BA+0x834	R	SD接收响应标志寄存器1	0x0000_0000
SDH_BLEN	SDHx_BA+0x838	R/W	SD数据块长度寄存器	0x0000_01FF
SDH_TOUT	SDHx_BA+0x83C	R/W	SD响应/数据输入超时寄存器	0x0000_0000

6.29.6 寄存器描述

共享缓存(SDH_FB_n)

寄存器	偏移量	R/W	描述	复位值
SDH_FB_n **n=0,1…31**	SDHx_BA+0x000 + 0x4 * n	R/W	共享缓存 (FIFO)	0x0000_0000

31	30	29	28	27	26	25	24
SB							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
SB							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
SB							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
SB							

位	描述	
[31:0]	**SB**	**SDH 共享缓存 (FIFO)** 一个128字节共享缓存用于系统内存和卡之间数据交换

DMA控制和状态寄存器(SDH_DMACTL)

寄存器	偏移量	R/W	描述	复位值
SDH_DMACTL	SDHx_BA+0x400	R/W	DMA控制和状态寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved						DMABUSY	Reserved
7	6	5	4	3	2	1	0
Reserved				SGEN	Reserved	DMARST	DMAEN

位	描述	
[31:10]	Reserved	保留
[9]	DMABUSY	**DMA 传输正在进行** 这一位指示SD主机被允许且正在进行DMA传输 0 = DMA传输不在进行 1 = DMA传输正在进行
[8:4]	Reserved	保留
[3]	SGEN	**分散采集功能使能位** 0 = 禁用分散采集功能 (DMA 会把DMASAR 里的起始地址作为一个单一内存块的开始指针) 1 = 使能分散采集功能 (DMA 会把 DMASAR 里的起始地址作为物理地址描述符（PAD）列表的起始地址，PAD的格式会在之后描述)
[2]	Reserved	保留
[1]	DMARST	**软件引擎复位** 0 = 没效果 1 = 复位内部状态机和指针。控制寄存器的值不会被清除。这一位会在几个时钟周期后自动清0。 **注:**程序复位DMA相关寄存器。
[0]	DMAEN	**DMA 引擎使能位** 0 = DMA 禁止. 1 = DMA 使能. 如果这一位为0，DMA会忽略所有SD主机的请求，强制总线管理器进入空闲状态。 **注:**如果发送目标异常, DMAEN 会清0。

DMA传输开始地址寄存器(SDH_DMASA)

寄存器	偏移量	R/W	描述	复位值
SDH_DMASA	SDHx_BA+0x408	R/W	DMA传输开始地址寄存器	0x0000_0000

31	30	29	28	27	26	25	24
DMASA							
23	22	21	20	19	18	17	16
DMASA							
15	14	13	12	11	10	9	8
DMASA							
7	6	5	4	3	2	1	0
DMASA							ORDER

位	描述	
[31:1]	DMASA	**DMA 传输开始地址** 这个区域是PAD0的起始地址指示32位的系统内存SRAM的起始地址，用作DMA发送或接收数据。如果 DMA 不在正常工作模式, 这个区域指向指向物理地址描述符（PAD）列表的起始地址. **注:** SRAM的起始地址必须字对齐，例如0x0000_0000, 0x0000_0004
[0]	ORDER	**决定PAD列表是否按顺序读取** 0 = PAD 列表按顺序读取 1 = PAD 列表不按顺序读取 **注:**分散采集模式下当SGEN = 1时bit0有效

DMA传输字节数寄存器(SDH_DMABCNT)

寄存器	偏移量	R/W	描述	复位值
SDH_DMABCNT	SDHx_BA+0x40C	R	DMA传输字节数寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved						BCNT	
23	22	21	20	19	18	17	16
BCNT							
15	14	13	12	11	10	9	8
BCNT							
7	6	5	4	3	2	1	0
BCNT							

位	描述	
[31:26]	Reserved	保留
[25:0]	BCNT	**DMA 传输字节数(只读)** 这个区域显示DMA传输剩下的字节数。这个值只在DMA忙碌时有效，其他情况，这个值是0。

DMA中断使能控制寄存器(SDH_DMAINTEN)

寄存器	偏移量	R/W	描述	复位值
SDH_DMAINTEN	SDHx_BA+0x410	R/W	DMA中断使能控制寄存器	0x0000_0001

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved						WEOTIEN	ABORTIEN

位	描述	
[31:2]	Reserved	保留
[1]	WEOTIEN	**遇到错误EOT中断使能位** 0 = 当遇到错误EOT时不产生中断 1 = 当遇到错误EOT时产生中断
[0]	ABORTIEN	**DMA读/写目标异常中断使能位** 0 = DMA传输时目标异常不产生中断 1 = DMA传输时目标异常产生中断

DMA中断状态寄存器(SDH_DMAINTSTS)

寄存器	偏移量	R/W	描述	复位值
SDH_DMAINTSTS	SDHx_BA+0x414	R/W	DMA中断状态寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved						WEOTIF	ABORTIF

位	描述	
[31:2]	Reserved	保留
[1]	WEOTIF	**遇到错误EOT中断标志（只读）** 当使能了分散采集DMA功能，在DMA传输完成前，描述符EOT位为1（PAD所有数据个数比SD主机设定的数目少），这一位会置1。 0 = DMA传输完成前没有遇到EOT位 1 = DMA传输完成前遇到了EOT位 **注意:**此位只读，写1清0
[0]	ABORTIF	**DMA读/写目标异常中断标志（只读）** 0 = 没有接收到总线错误响应 1 = 接收到总线错误响应 **注1:**此位只读，写1清0 **注2:**当DMA总线控制器收到错误响应，意味着发生了目标异常。DMA 会停止传输响应这个事件，总线进入空闲态。当发生了目标异常或WEOTIF为1，程序需要复位DMA 和 SD 主机，然后重新开始数据传输

全局控制和状态寄存器(SDH_GCTL)

寄存器	偏移量	R/W	描述	复位值
SDH_GCTL	SDHx_BA+0x800	R/W	全局控制和状态寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved						SDEN	GCTLRST

位	描述	
[31:2]	Reserved	保留
[1]	SDEN	**SD功能使能位** 0 = SD功能禁止 1 = SD功能使能
[0]	GCTLRST	**程序引擎复位** 0 = 没有影响 1 = 复位 SD 主机。控制寄存器不会被清除。此位在Reset完成后自动清0

全局中断控制寄存器(SDH_GINTEN)

寄存器	偏移量	R/W	描述	复位值
SDH_GINTEN	SDHx_BA+0x804	R/W	全局中断控制寄存器	0x0000_0001

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved							DTAIEN

位	描述	
[31:1]	Reserved	保留
[0]	DTAIEN	DMA 读/写目标异常中断使能位 0 = 禁止DMA 读/写目标异常中断 1 = 使能DMA 读/写目标异常中断

全局中断状态寄存器(SDH_GINTSTS)

寄存器	偏移量	R/W	描述	复位值
SDH_GINTSTS	SDHx_BA+0x808	R/W	全局中断状态寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved							DTAIF

位	描述	
[31:1]	Reserved	保留
[0]	DTAIF	**DMA读/写目标异常中断标志(只读)** 这一位指示DMA在读/写操作时接收到从内部AHB总线上收到错误响应。当发生了目标异常，请复位所有的引擎。 0 = 没有收到总线错误响应 1 = 接收到总线错误响应 **注:**此位只读，写1清0

SD控制和状态寄存器(SDH_CTL)

寄存器	偏移量	R/W	描述	复位值
SDH_CTL	SDHx_BA+0x820	R/W	SD控制和状态寄存器	0x0101_0000

31	30	29	28	27	26	25	24
Reserved				SDNWR			
23	22	21	20	19	18	17	16
BLKCNT							
15	14	13	12	11	10	9	8
DBW	CTLRST	CMDCODE					
7	6	5	4	3	2	1	0
CLKKEEP	CLK8OEN	CLK74OEN	R2EN	DOEN	DIEN	RIEN	COEN

位	描述	
[31:28]	**Reserved**	保留
[27:24]	**SDNWR**	**数据块写操作写信号参数** 这个值是数据块写操作时NWR参数，以SD时钟为单位，实际值是SDNWR+1.
[23:16]	**BLKCNT**	**需要发送或接收的数据块个数** 这个区域包含输入输出的数据块个数。用于READ_MULTIPLE_BLOCK 和 WRITE_MULTIPLE_BLOCK 指令, 程序可以使用这个功能加速数据传输提升性能。不要在这里填0x0。 **注:**对于READ_MULTIPLE_BLOCK 和 WRITE_MULTIPLE_BLOCK 指令,实际长度是 BLKCNT * (BLKLEN +1)。
[15]	**DBW**	**SD 数据总线宽度 (1线/4线选择)** 0 = 数据总线宽度是1位 1 = 数据总线宽度是4位
[14]	**CTLRST**	**软件引擎复位** 0 = 没影响 1 = 复位内部状态机和计数器。控制寄存器的值不会清除(但是 RIEN, DIEN, DOEN 和 R2_EN 位会清0). 这一位会在几个时钟后自动清0.
[13:8]	**CMDCODE**	**SD 指令码** T这个寄存器存放 SD 指令码 (0x00 – 0x3F).
[7]	**CLKKEEP**	**SD 端口时钟使能位** 0 = SD主机自动决定什么时候输出时钟，什么时候停止输出 1 = SD时钟总是输出

[6]	**CLK8OEN**	**产生8个时钟周期输出使能位** 0 = 没效果 (请通过 DMARST (SDH_CTL [0]) 清这一位) 1 =使能SD主机输出8个时钟周期 **注:**操作结束后，此位自动清0，不要写0到这个寄存器，否则控制器会工作不正常
[5]	**CLK74OEN**	**初始化 74 个时钟周期输出使能位** 0 = 没效果 (请通过 DMARST (SDH_CTL [0]) 清这一位) 1 = 使能，SD主机会输出74个时钟周期到SD卡。 **注:**操作结束后，此位自动清0，不要写0到这个寄存器，否则控制器会工作不正常
[4]	**R2EN**	**响应 R2 输入使能位** 0 = 没效果 (请通过 DMARST (SDH_CTL [0]) 清这一位) 1 = 使能，SD主机会等待从SD卡接收R2响应，并且存储响应数据到DMC的Flash缓存（不包含CRC7） **注:**操作结束后，此位自动清0，不要写0到这个寄存器，否则控制器会工作不正常
[3]	**DOEN**	**数据输出使能位** 0 = 没效果 (请通过 DMARST (SDH_CTL [0]) 清这一位) 1 = 使能, SD 主机会传输数据块和CRC16值到SD卡 **注:**操作结束后，此位自动清0，不要写0到这个寄存器，否则控制器会工作不正常
[2]	**DIEN**	**数据输入使能位** 0 = 没效果 (请通过 DMARST (SDH_CTL [0]) 清这一位) 1 = 使能，SD主机会等待从SD卡接收CRC16值 **注:**操作结束后，此位自动清0，不要写0到这个寄存器，否则控制器会工作不正常
[1]	**RIEN**	**响应输入使能位** 0 = 没效果(请通过 DMARST (SDH_CTL [0]) 清这一位) 1 = 使能，SD主机等待从SD卡接收响应 **注:**操作结束后，此位自动清0，不要写0到这个寄存器，否则控制器会工作不正常
[0]	**COEN**	**指令输出使能位** 0 = 没效果 (请通过 DMARST (SDH_CTL [0]) 清这一位) 1 = 使能，SD主机会输出指令到SD卡 **注:**操作结束后，此位自动清0，不要写0到这个寄存器，否则控制器会工作不正常

SD指令参数寄存器(SDH_CMDARG)

寄存器	偏移量	R/W	描述	复位值
SDH_CMDARG	SDHx_BA+0x824	R/W	SD指令参数寄存器	0x0000_0000

31	30	29	28	27	26	25	24
ARGUMENT							
23	22	21	20	19	18	17	16
ARGUMENT							
15	14	13	12	11	10	9	8
ARGUMENT							
7	6	5	4	3	2	1	0
ARGUMENT							

位	描述	
[31:0]	ARGUMENT	**SD指令参数** 这个寄存器包含从主控制器到SD卡的一个32位SD指令参数值，在触发COEN (SDH_CTL [0])前, 程序需要填这个区域

SD中断控制寄存器(SDH_INTEN)

寄存器	偏移量	R/W	描述	复位值
SDH_INTEN	SDHx_BA+0x828	R/W	SD中断控制寄存器	0x0000_0A00

31	30	29	28	27	26	25	24
Reserved	CDSRC	Reserved					
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved	WKIEN	DITOIEN	RTOIEN	Reserved		CDIEN	
7	6	5	4	3	2	1	0
Reserved						CRCIEN	BLKDIEN

位	描述	
[31]	Reserved	保留
[30]	CDSRC	**SD0卡片检测源选择** 0 = 从 SD 卡的 DAT3 管脚 主机需要时钟从DAT3读取数据。为了保证DAT3能产生时钟，请确定 CLKKEEP(SDH_CTL[7])是1。 1 = 从GPIO 管脚
[29:15]	Reserved	保留
[14]	WKIEN	**产生唤醒信号使能位** 使能或禁止当前SD卡通过DAT1管脚产生的中断（唤醒）信号 0 = 禁止 1 = 使能
[13]	DITOIEN	**数据输入超时中断使能位** 使能或禁止SD控制器输入超时时产生的中断，超时时间由TOUT 寄存器设置 0 = 禁止 1 = 使能
[12]	RTOIEN	**响应超时中断使能位** 使能或禁止当接收响应或R2超时时SD控制器产生的中断，超时时间由TOUT 寄存器设置 0 = 禁止 1 = 使能
[11:9]	Reserved	保留
[8]	CDIEN	保留

[7:2]	**Reserved**	**SD 卡片检测中断使能位** 使能或禁止当SD卡片插入或拔出时产生中断 0 = 禁止 1 = 使能
[1]	**CRCIEN**	保留
[0]	**BLKDIEN**	**CRC7, CRC16 和CRC状态错误中断使能位** 0 = SD主机当CRC7, CRC16 和 CRC状态错误时不产生中断 1 = SD主机当CRC7, CRC16 和 CRC状态错误时产生中断
		块传输完成中断使能位 0 = SD主机当数据输入输出传输完成后不产生中断 1 = SD主机当数据输入输出传输完成后产生中断

SD中断状态寄存器(SDH_INTSTS)

寄存器	偏移量	R/W	描述	复位值
SDH_INTSTS	SDHx_BA+0x82C	R/W	SD中断状态寄存器	0x000X_008C

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved					DAT1STS	Reserved	CDSTS
15	14	13	12	11	10	9	8
Reserved		DITOIF	RTOIF	Reserved		Reserved	CDIF
7	6	5	4	3	2	1	0
DAT0STS	CRCSTS			CRC16	CRC7	CRCIF	BLKDIF

位	描述	
[31:19]	**Reserved**	保留
[18]	**DAT1STS**	**SD端口的DAT1 管脚状态 (只读)** 这一位显示SD端口的DAT1 管脚状态
[17]	**Reserved**	保留
[16]	**CDSTS**	**SD卡片检测状态(只读)** 这一位只是SD卡片检测管脚的状态，用来卡片检测。当SD有卡片插入或拔出，程序需要检查这一位来确认卡片是否真的插入或拔出了。 如果 CDSRC (SDH_INTEN[30]) = 0,选择 DAT3作为卡片检测： 0 = 卡片拔出 1 = 卡片插入 如果CDSRC(SDH_INTEN[30]) = 1,选择 GPIO 作为卡片检测 0 = 卡片拔出 1 = 卡片插入
[15:14]	**Reserved**	保留
[13]	**DITOIF**	**数据输入超时中断标志(只读)** 这一位指示SD主机接收数据时发生了超时(等待开始位) 0 = 没有超时 1 = 数据输入超时 **注:**此位只读，写1清0

[12]	**RTOIF**	**响应超时中断标志(只读)** 这一位指示SD主机接收响应或R2时发生了超时(等待开始位) 0 = 没有超时 1 = 响应超时 **注:**此位只读，写1清0
[11:9]	**Reserved**	保留
[8]	**CDIF**	**SD卡片检测中断状态标志 (只读)** 这一位指示SD卡片插入或拔出。只有当CDIEN(SDH_INTEN[8]) 设为1, 这一位才有效。 0 = 没有卡片插入或拔出 1 = SD有卡片插入或拔出 **注:**此位只读，写1清0
[7]	**DAT0STS**	**目前选择的SD端口的DAT0 管脚状态 (只读)** 这一位指示目前选择的SD端口的 DAT0 管脚状态
[6:4]	**CRCSTS**	**数据输出传输CRC状态值(只读)** SD 主机会记录输出数据的CRC状态值。程序可以用这个值了解数据传输时发生了哪种错误。 010 = 正CRC 状态 101 = 负CRC状态 111 = 发生了SD卡编程错误
[3]	**CRC16**	**输入数据传输CRC16 状态值 (只读)** SD 主机在数据输入时确认CRC16 正确性。 0 = 错误 1 = 正确
[2]	**CRC7**	**CRC7 检测状态 (只读)** SD主机在每笔响应输入时会确认CRC7正确性。如果响应不包含 CRC7 信息 (如. R3), 那么程序需要关闭 CRCIEN (SDH_INTEN[1]) 然后忽略这一位。 0 = 错误 1 = 正确
[1]	**CRCIF**	**CRC7, CRC16 和 CRC 状态错误中断标志(只读)** 这一位指示SD 主机在响应输入，数据输入或数据输出（CRC状态错误）传输时发生了CRC错误。如果发生了CRC错误，程序需要复位SD引擎。有一些响应(例如 R3) 并没有CRC7域，SD主机仍然会计算CRC7，得到CRC错误并置位这个标志。在这种情况下程序需要忽略CRC错误并手动清除此位。 0 = 没有 CRC错误发生 1 = 发生了CRC错误 **注:**此位只读，写1清0
[0]	**BLKDIF**	**块传输完成中断标志 (只读)** 这一位指示SD主机完成了所有数据输入输出的传输。如果在块传输中产生了CRC16错误或不正确的CRC状态，传输会中断此位也会置1. 0 = 没有完成 1 = 完成了 **注:**此位只读，写1清0

SD接收响应标志寄存器0(SDH_RESP0)

寄存器	偏移量	R/W	描述	复位值
SDH_RESP0	SDHx_BA+0x830	R	SD接收响应标志寄存器0	0x0000_0000

31	30	29	28	27	26	25	24
RESPTK0							
23	22	21	20	19	18	17	16
RESPTK0							
15	14	13	12	11	10	9	8
RESPTK0							
7	6	5	4	3	2	1	0
RESPTK0							

位	描述	
[31:0]	RESPTK0	**SD接收响应标志 0** SD 主机控制器当RIEN (SDH_CTL[1])为1时会接收响应标志得到SD卡的回复。这个寄存器包含相应标识的47-16位

SD接收响应标志寄存器1(SDH_RESP1)

寄存器	偏移量	R/W	描述	复位值
SDH_RESP1	SDHx_BA+0x834	R	SD接收响应标志寄存器1	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
RESPTK1							

位	描述	
[7:0]	RESPTK1	**SD接收响应标志 1** SD 主机控制器当RIEN (SDH_CTL[1])为1时会接收响应标志得到SD卡的回复。这个寄存器包含相应标识的15-8位

SD数据块长度寄存器(SDH_BLEN)

寄存器	偏移量	R/W	描述	复位值
SDH_BLEN	SDHx_BA+0x838	R/W	SD数据块长度寄存器	0x0000_01FF

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved					BLKLEN		
7	6	5	4	3	2	1	0
BLKLEN							

位	描述	
[10:0]	BLKLEN	**SD 数据块长度字节数** 一个11位的值设定设定SD传输数据块的字节数。实际的字节数是BLKLEN+1. **注:** 默认的SD数据块长度是512字节

SD响应/数据输入超时寄存器(SDH_TOUT)

寄存器	偏移量	R/W	描述	复位值
SDH_TOUT	SDHx_BA+0x83C	R/W	SD响应/数据输入超时寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
TOUT							
15	14	13	12	11	10	9	8
TOUT							
7	6	5	4	3	2	1	0
TOUT							

位	描述	
[23:0]	TOUT	**SD响应/数据输入超时值** 一个24位的值设定响应/数据输入超时计数值。SD 主机控制器会等待输入数据或相应的开始位直到计数达到这个值。时钟周期依照SD引擎时钟频率。不要在这里写入过小的值，那样你会因为超时而收不到数据或响应。 **注:**填0x0将禁止硬件超时功能

6.30 EBI外部总线接口

6.30.1 概述

该芯片配有外部总线接口（EBI），供外部设备使用。为了保证外部设备和芯片之间的连接，EBI在地址总线和数据总线复用模式下工作。EBI支持三个片选，可以连接三个不同时序的外部设备。

6.30.2 特性

- 支持三路存储banks
- 为每个bank提供了有极性控制的片选引脚
- 每个bank访问空间达到1MB，实际上外部可寻址空间取决于芯片封装引脚
- 支持8位/16位数据宽度
- 16位数据宽度模式下支持字节写
- 支持地址/数据复用模式
- 每个存储bank支持时序参数独立调整
- 支持LCD i80接口模式
- 支持PDMA模式
- 支持基于HCLK所产生的不同频率的总线基本时钟(MCLK)
- 支持可配置的空闲周期以用于不同访问条件：空闲写命令完成(W2X)，空闲连续读(R2R)
- 支持地址总线个数据总线分离模式

6.30.3 框图

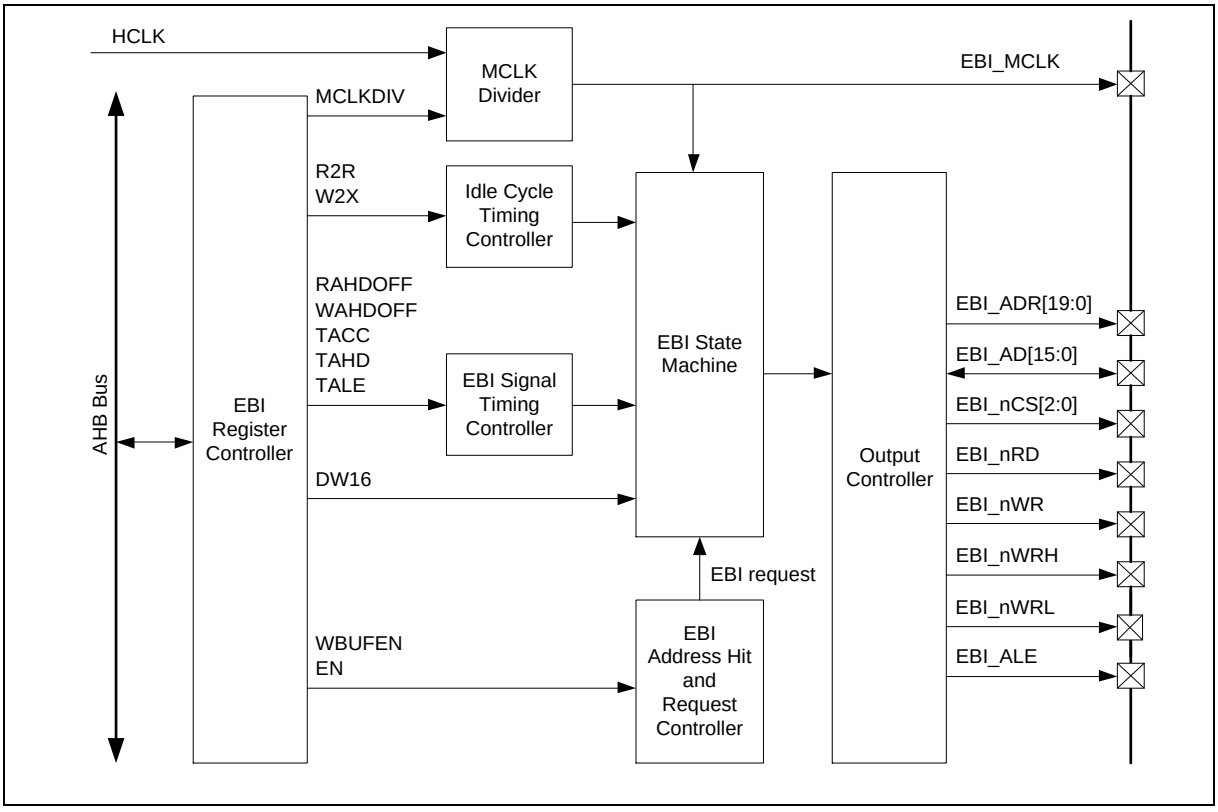


图 6.30-1 EBI 框图

6.30.4 基本配置

- 时钟源配置
  - 使能EBI控制器时钟位EBICKEN(CLK_AHBCLK[3])
- 复位配置
  - 复位EBI控制器时钟位SPI0RST(SYS_IPRST0[3])
- 引脚配置

组	引脚名称	GPIO	MFP
EBI	EBI_AD0	PC.0, PG.9	MFP2
	EBI_AD1	PC.1, PG.10	MFP2
	EBI_AD2	PC.2, PG.11	MFP2
	EBI_AD3	PC.3, PG.12	MFP2
	EBI_AD4	PC.4, PG.13	MFP2
	EBI_AD5	PC.5, PG.14	MFP2
	EBI_AD6	PA.6, PD.8	MFP2

	EBI_AD7	PA.7, PD.9	MFP2
	EBI_AD8	PC.6, PE.14	MFP2
	EBI_AD9	PC.7, PE.15	MFP2
	EBI_AD10	PD.3, PD.13, PE.1	MFP2
	EBI_AD11	PC.14, PD.2, PE.0	MFP2
	EBI_AD12	PB.15, PD.1, PH.8	MFP2
	EBI_AD13	PB.14, PD.0, PH.9	MFP2
	EBI_AD14	PB.13, PH.10	MFP2
	EBI_AD15	PB.12, PH.11	MFP2
	EBI_ADR0	PB.5, PH.7	MFP2
	EBI_ADR1	PB.4, PH.6	MFP2
	EBI_ADR2	PB.3, PH.5	MFP2
	EBI_ADR3	PB.2, PH.4	MFP2
	EBI_ADR4	PC.12, PH.3	MFP2
	EBI_ADR5	PC.11, PH.2	MFP2
	EBI_ADR6	PC.10, PH.1	MFP2
	EBI_ADR7	PC.9, PH.0	MFP2
	EBI_ADR8	PB.1, PG.0	MFP2
	EBI_ADR9	PB.0, PG.1	MFP2
	EBI_ADR10	PC.13, PE.8	MFP2
	EBI_ADR11	PE.9, PG.2	MFP2
	EBI_ADR12	PE.10, PG.3	MFP2
	EBI_ADR13	PE.11, PG.4	MFP2
	EBI_ADR14	PE.12, PF.11	MFP2
	EBI_ADR15	PE.13, PF.10	MFP2
	EBI_ADR16	PB.11, PC.8, PF.9	MFP2
	EBI_ADR17	PB.10, PF.8	MFP2
	EBI_ADR18	PB.9, PF.7	MFP2
	EBI_ADR19	PB.8, PF.6	MFP2
	EBI_ALE	PA.8, PE.2	MFP2
	EBI_MCLK	PA.9, PE.3	MFP2
	EBI_nCS0	PD.12, PD.14, PF.3	MFP2
		PF.6	MFP7
		PB.7	MFP8

	EBI_nCS1	PD.11, PF.2, PG.5	MFP2
		PB.6	MFP8
	EBI_nCS2	PD.10, PG.6	MFP2
	EBI_nRD	PA.11, PE.5	MFP2
	EBI_nWR	PA.10, PE.4	MFP2
	EBI_nWRH	PB.6, PG.8	MFP2
	EBI_nWRL	PB.7, PG.7	MFP2

6.30.5 功能描述

6.30.5.1 EBI区域和地址

EBI映像地址分布在0x6000_0000 ~ 0x602F_FFFF，内存空间总共3Mbytes。当系统请求地址到达EBI的存储空间时，相应的EBI芯片选择信号被断言，EBI状态机工作。

片选信号	地址映像
EBI_nCS0	0x6000_0000 ~ 0x600F_FFFF
EBI_nCS1	0x6010_0000 ~ 0x601F_FFFF
EBI_nCS2	0x6020_0000 ~ 0x602F_FFFF

表 6.30-1 EBI 地址映像

为了映射整个EBI 内存空间，8位设备需20位地址宽度，16位设备需19位地址宽度。小于20-位地址的设备， EBI 将映射设备到镜像空间。例如，一个18-位 EBI地址设备，EBI （Bank0/EBI_nCS0) 将外设同时映射到0x6000_0000~0x6003_FFFF，0x6004_0000~0x6007_FFFF，0x6008_0000~ 0x600B_FFFF和0x600C_0000~0x600F_FFFF。

6.30.5.2 EBI数据宽度连接-地址与数据总线复用模式

EBI 支持地址总线和数据总线复用。对于地址总线和数据总线分开的外部设备，与设备的连接需要额外的逻辑单元锁存地址。这样，管脚EBI_ALE需要连接到锁存器上锁存地址值。管脚EBI_AD为锁存器的输入，锁存器的输出连接到外部设备的地址总线上。

对于16位设备， EBI_AD[15:0]是地址与数据复用线， EBI_ADR [18:16]仅为地址，直接与16位设备连接。当EBI 数据宽度设置为16位， EBI_ADR[19]无作用。对于8-位设备，仅EBI_AD [7:0]地址与数据复用，EBI_AD[15:8] 和 EBI_ADR[19:16]为地址，直接与8-位设备连接。

图6.30-2描述16位EBI 数据宽度与16-位设备的连接，图6.30-3描述8位EBI数据宽度与8位设备的连接

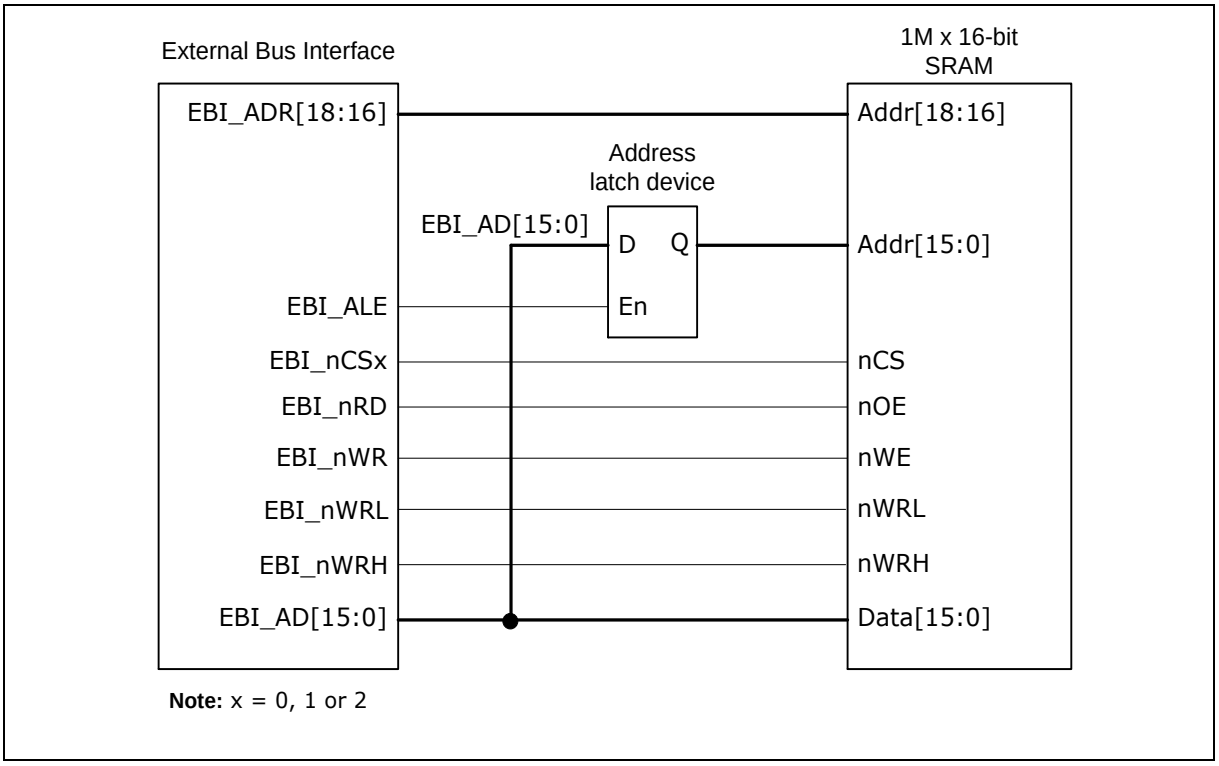


图 6.30-2 16 位 EBI 数据宽度与 16 位设备的连接

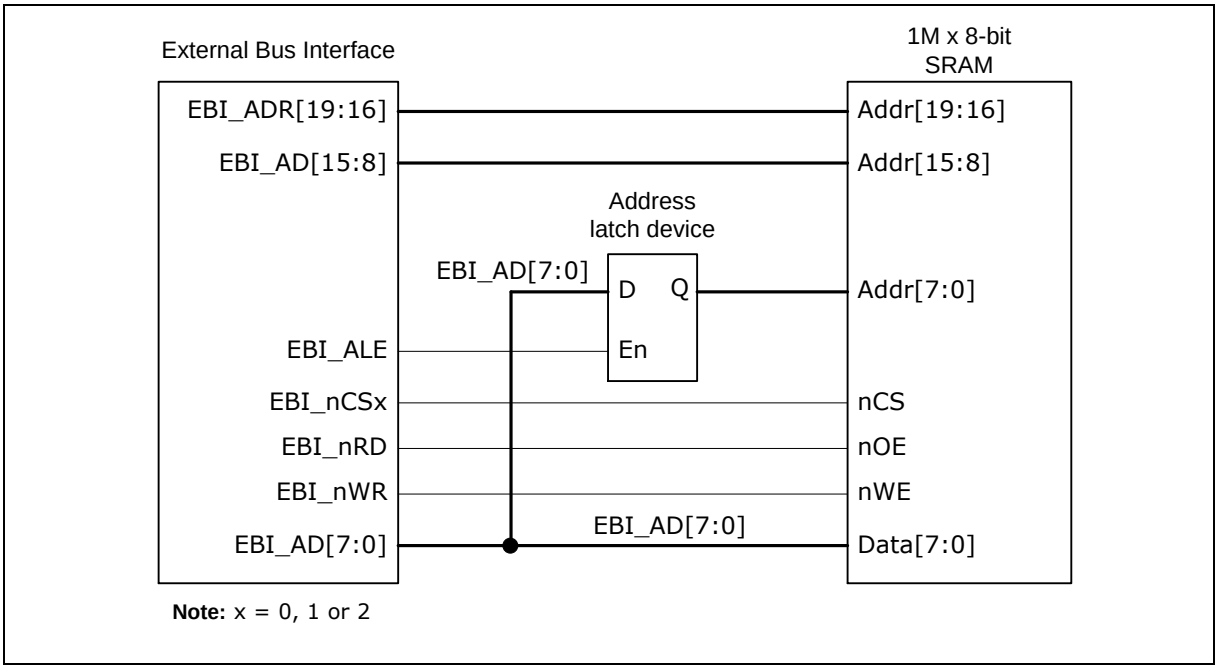


图 6.30-3 8 位 EBI 数据宽度与 8 位设备的连接

当系统访问数据宽度大于EBI数据宽度时，EBI控制器将通过多次操作EBI访问来完成系统访问命令。例如，如果系统通过EBI设备请求32位数据，当设置EBI数据宽度为8位时，EBI控制器将操作访问4次。

6.30.5.3 EBI数据宽度连接-地址与数据总线分离模式

EBI支持地址和数据总线分离模式。用户设置 ADSEPEN (EBI_CTLx[3]) 使能该模式：EBI_AD数据总线直接连接到设备数据总线，EBI_ADR直接连到设备地址总线。

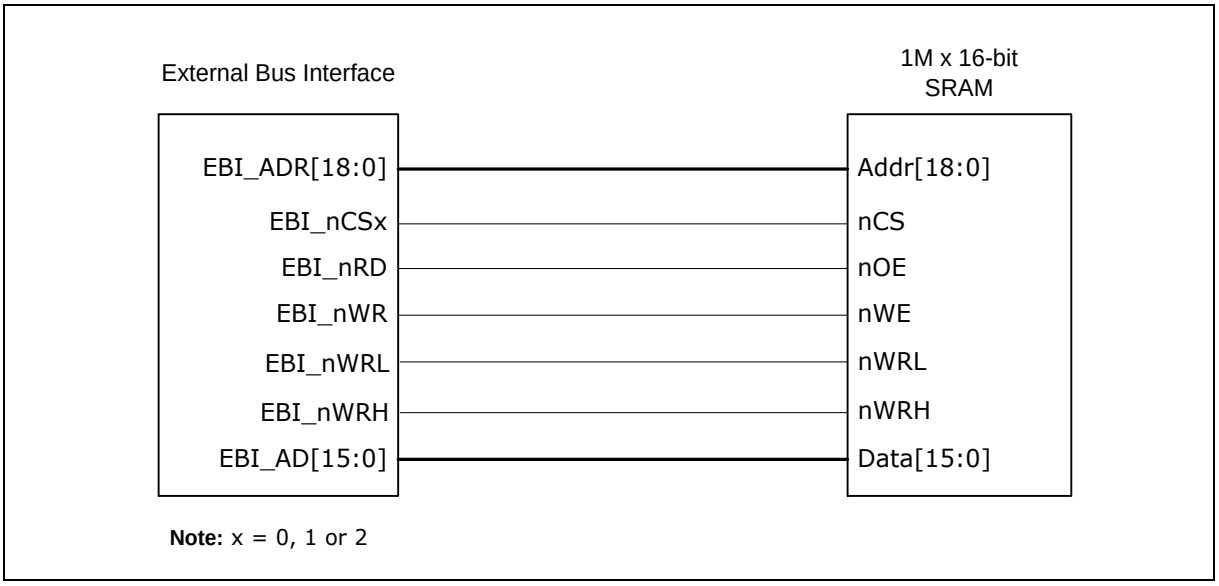


图 6.30-4 分离模式下 16 位 EBI 数据宽度与 16 位设备的连接

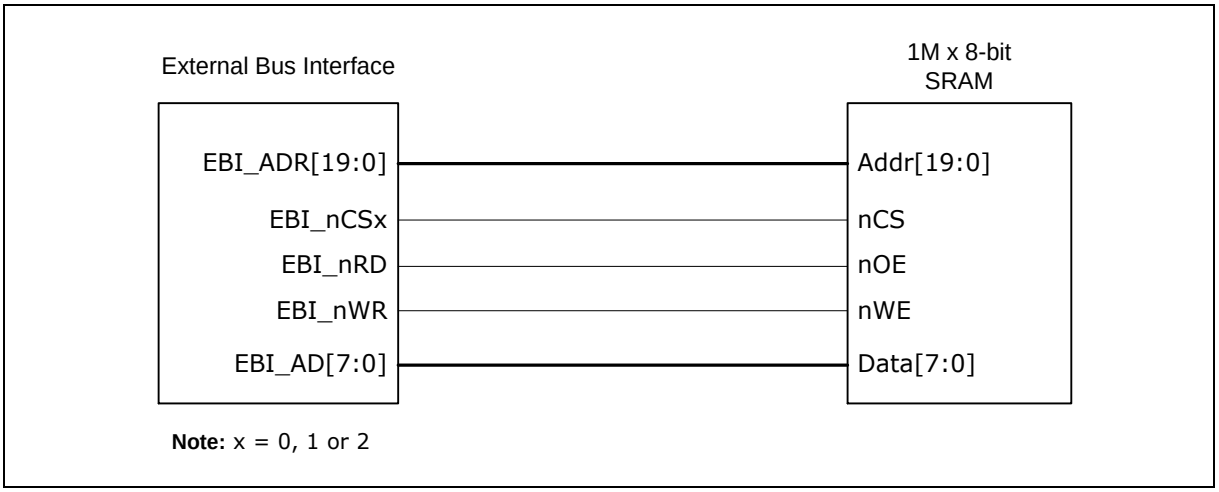


图 6.30-5 分离模式下 8 位 EBI 数据宽度与 8 位设备的连接

6.30.5.4 EBI操作控制

MCLK 控制

EBI工作时，所有EBI信号与EBI_MCLK同步。以低工作频率连接到外部设备时， EBI_MCLK可以通过寄存器MCLKDIV (EBI_CTLx[10:8])最多分频到HCLK/32。如果EBI_MCLK设置为HCLK/1，则EBI信号与EBI_MCLK的正边沿同步，否则与EBI_MCLK的负边沿同步。

操作和访问时序控制

在EBI接入开始时，片选(EBI_nCS0、EBI_nCS1和EBI_nCS2)置低并等待一个EBI_MCLK时间(tASU)以

使地址稳定，然后EBI_ALE置高并保持一段时间(tALE)用于地址锁存。地址锁存后，EBI_ALE置低并等待一个EBI_MCLK时间(tLHD)，再插入一个EBI_MCLK时间(tA2D)用于总线转换(地址到数据)。读时EBI_nRD置低，写时EBI_nWR置低。在保持访问时间(tACC)后EBI_nRD或EBI_nWR置高完成读或写。之后，EBI信号在数据保持时间(tAHD)之后片选置高，地址由当前访问控制释放。

EBI控制器为不同的外部设备提供灵活的时序控制。在EBI时序控制中，tASU, tLHD 和tA2D固定为1个EBI_MCLK时间，tAHD通过设定TAHD (EBI_TCTLx[10:8]) 寄存器可以在1~8个EBI_MCLK周期内调节， tACC通过设定TACC(EBI_TCTLx[7:3]) 寄存器可以在1~32个EBI_MCLK 周期内调节，tALE通过TALE(EBI_CTL0[18:16] 寄存器可以在1~8个EBI_MCLK周期内调节。一些外设可以支持无数据访问保持时间，EBI控制器可以跳过tAHD, 通过寄存器设定WAHDOFF (EBI_TCTLx[23]) 和RAHDOFF (EBI_TCTLx[22]) 提高访问速度。

对每个芯片的片选，除了tALE只由EBI_CTL0控制，EBI提供带有时序控制的单独的寄存器。

参数	值	单元	描述
tASU	1	MCLK	地址锁存建立时间
tALE	1 ~ 8	MCLK	ALE 高时间，由TALE (EBI_CTL0[18:16])控制
tLHD	1	MCLK	地址锁存保持时间
tA2D	1	MCLK	地址到数据的延时 (总线转换时间)
tACC	1 ~ 32	MCLK	数据访问时间，由TACC (EBI_TCTLx[7:3])控制
tAHD	1 ~ 8	MCLK	数据访问保持时间，由TAHD (EBI_TCTLx[10:8])控制
IDLE	0 ~ 15	MCLK	空闲周期. 由 R2R (EBI_TCTLx[27:24])和W2X (EBI_TCTLx[15:12])控制

表 6.30-2 时序控制参数

图6.30-6是一个设置16位数据宽度的例子。在该例中，EBI_AD总线用作地址[15:0]和数据[15:0]。当EBI_ALE置高，EBI_AD为地址输出。在地址锁存后，EBI_ALE置低并且在读取访问操作时，EBI_AD总线转换成高阻以等待设备输出数据，或用于写数据输出。

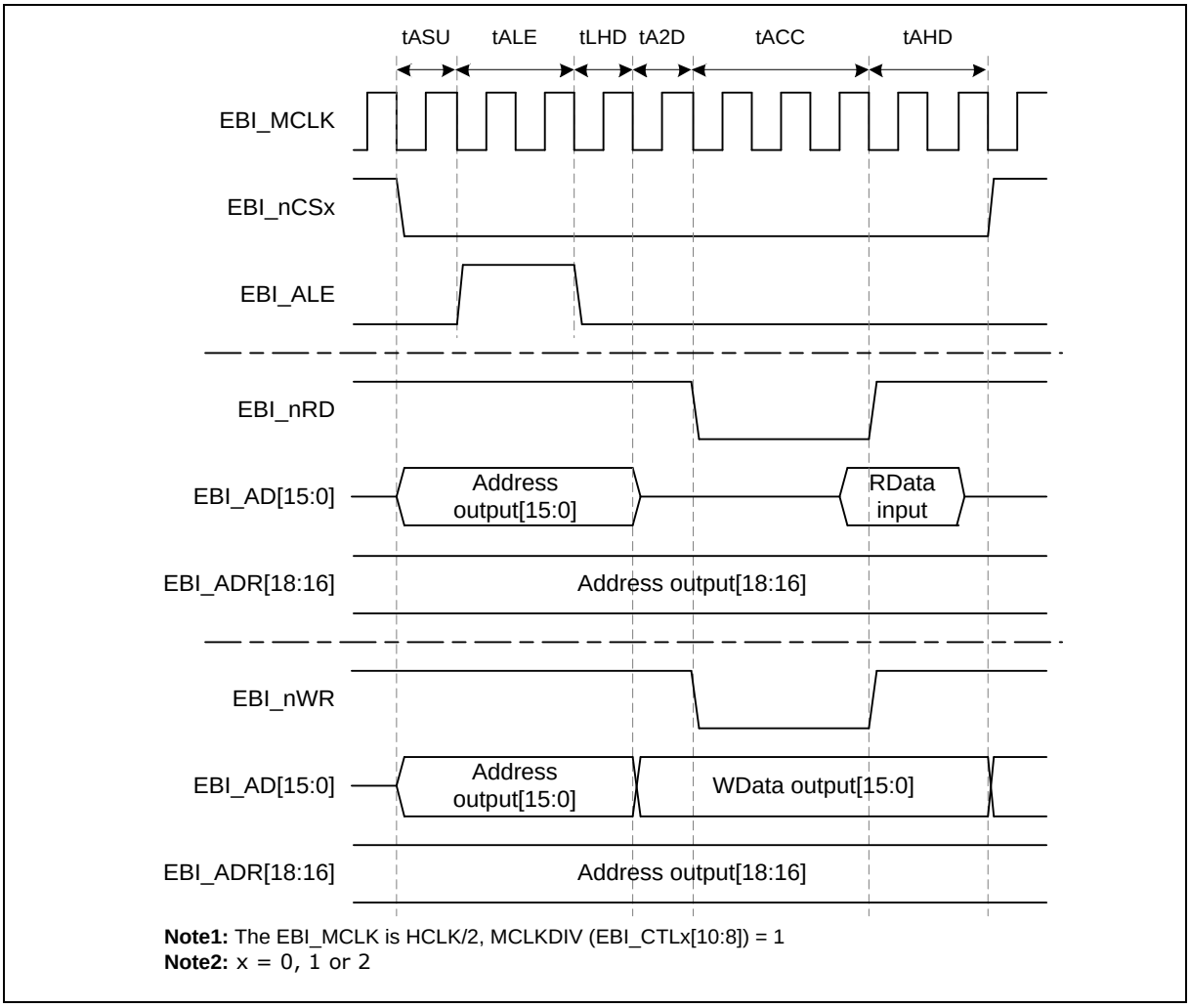
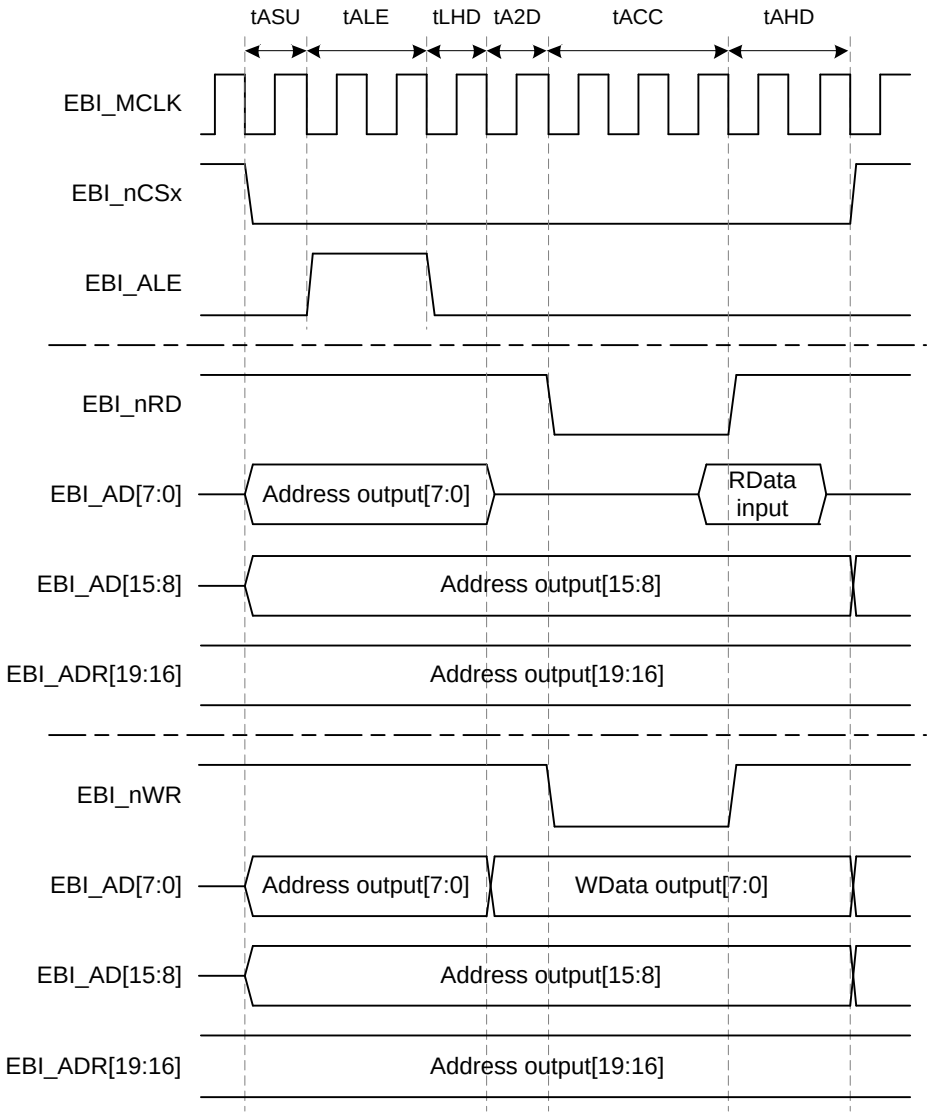


图 6.30-6 16 位数据宽度的时序控制波形

图6.30-7是一个设置8位数据宽度的例子。8位和16位数据宽度的不同之处在于EBI_AD[15:8]。在8位数据宽度的设置中，EBI_AD[15:8]总为地址[15:8]输出，因此外部锁存只需8-位宽度。

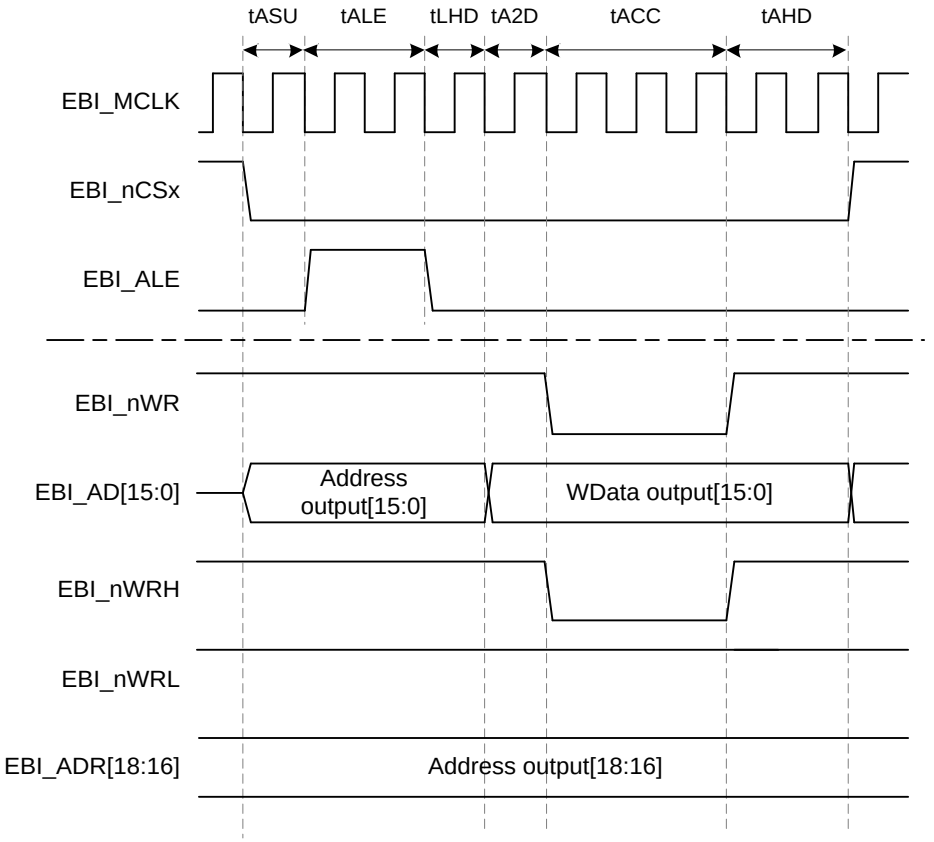


Note1: The EBI_MCLK is HCLK/2, MCLKDIV (EBI_CTLx[10:8]) = 1
Note2: x = 0, 1 or 2

图 6.30-7 8 位数据宽度的时序控制波形

字节访问

当连接16位设备时EBI支持字节访问。16位数据总线上用引脚EBI_nWRH和EBI_nWRL指示高字节使能和低字节使能。图6.30-8是展示用EBI_nWRH指示在EBI_AD[15:8]上8位数据宽度的写操作。



Note1: The EBI_MCLK is HCLK/2, MCLKDIV (EBI_CTLx[10:8]) = 1
Note2: x = 0, 1 or 2

图 6.30-816 位数据宽度字节写的时序控制波形

插入空闲周期

当EBI连续访问时，如果器件访问时间较慢，可能会有总线冲突。EBI控制器支持额外空闲周期以解决该问题。在空闲周期，所有EBI的控制信号无效。如下图6.30-9空闲周期。

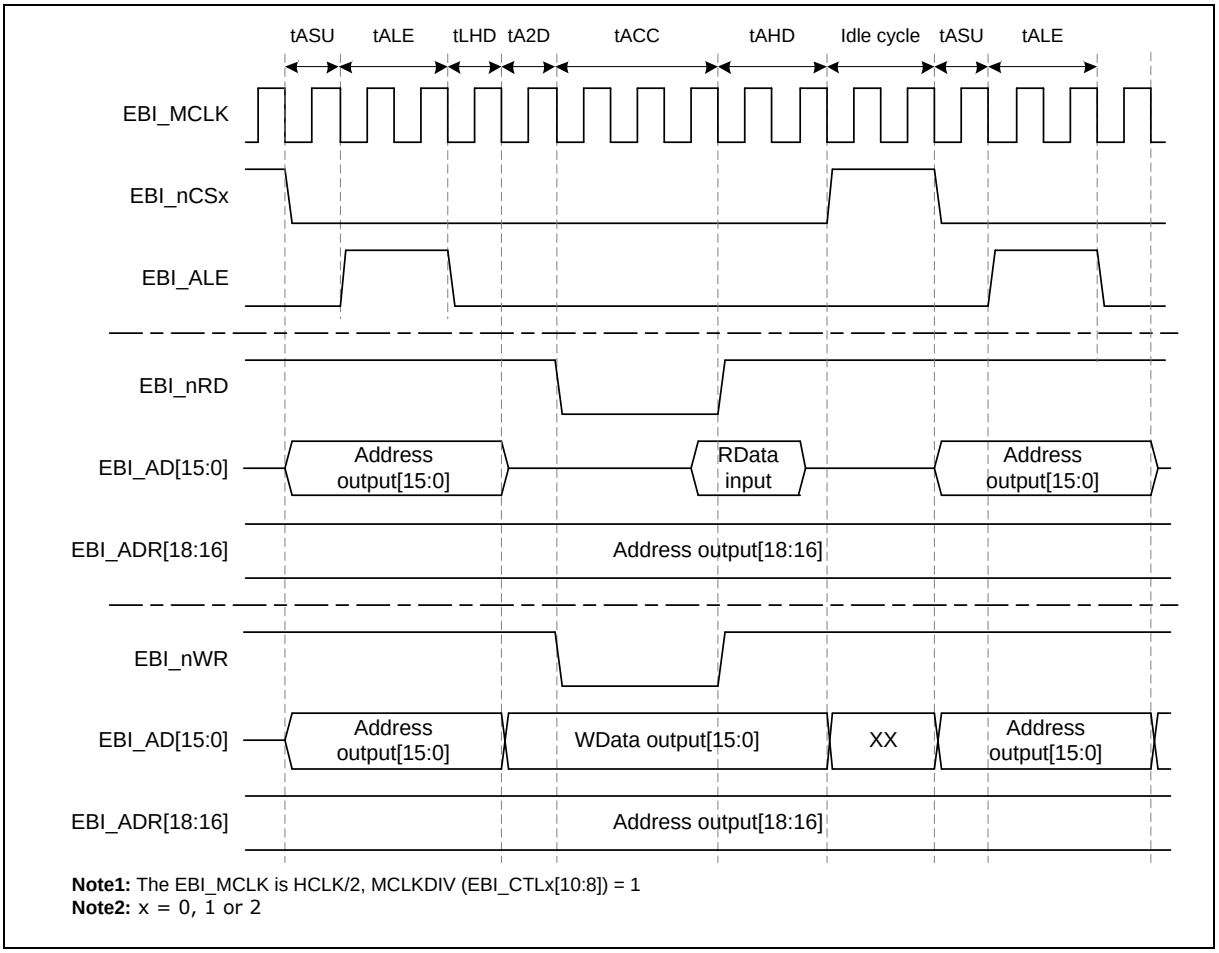


图 6.30-9 插入空闲周期的时序控制

以下两个条件，EBI可插入空闲周期：

1. 写访问之后
2. 读访问之后与下一个读访问之前(R2R 空闲周期)

通过设定寄存器W2X (EBI_TCTLx[15:12])，和R2R (EBI_TCTLx[27:24])，空闲周期可设定在0~15 EBI_MCLK。

片选极性控制

EBI支持芯片选择极性控制，用于连接可变外部设备。当CSPOLINV (EBI_CTLx[2])置0，片选(EBI_nCSx)低电平有效。这意味着在EBI_nCSx低电平时访问外部设备。当CSPOLINV (EBI_CTLx[2])置1，片选(EBI_nCSx)高电平有效。这意味着当在EBI_nCSx高电平时访问外部设备。

写缓存

当用户通过EBI总线写一个数据到外设时，EBI控制器将立刻开始写动作，CPU是保持状态，直到EBI写

动作结束。用户可以使能写缓存功能来提高CPU和EBI的访问效率。当EBI写缓存功能使能后，在EBI向外设写数据期间，CPU还可以持续执行其他指令。例外的情况，当EBI执行写动作时，如果CPU通过EBI执行令另一个数据访问，CPU会是保持状态。

用户可以通过设置 WBUFEN (EBI_CTL0[24])使能写缓存。

地址数据分离模式

当EBI设置为分离模式， tALE, tLHD, tA2D周期会被忽略。EBI_AD 和 EBI_ADR分别各自用于数据和地址总线。

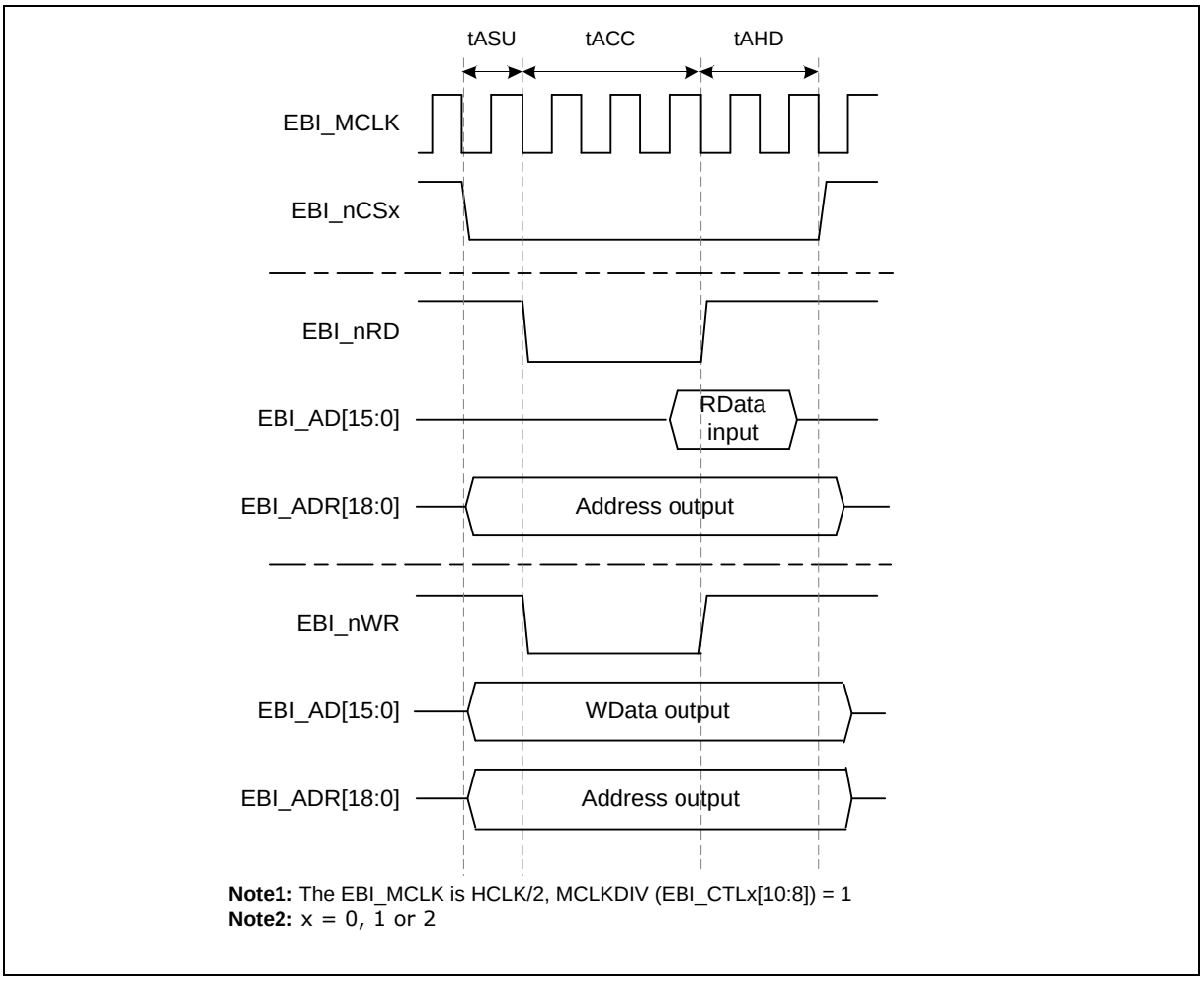


图 6.30-10 位数据宽度在分离模式下的时序控制波形

连续数据访问模式

对于需要更快的数据访问且不需要地址控制的接口，EBI支持连续数据访问。每个bank，用户可以通过设置CACCESS (EBI_CTLX[4])使能该模式。当EBI设置为连续数据访问模式，tASU, tALE, tLHD周期会被忽略，且EBI可以在一个读或写命令内连续地访问数据。在每个访问命令之间会有无效的周期时间。时序如下图6.30-11。

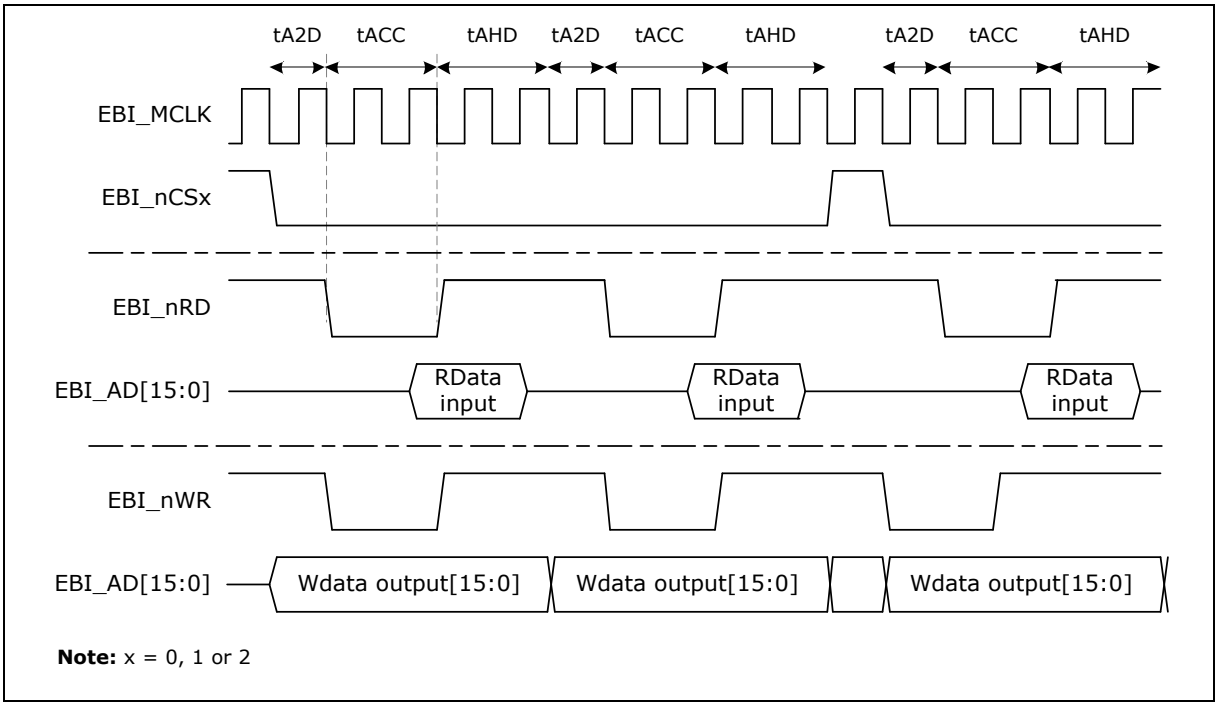


图 6.30-11 连续数据访问模式的时序控制波形

6.30.6 寄存器映射

R: 只读, **W**: 只写, **R/W**: 读/写

寄存器	偏移量	R/W	描述	复位值
EBI基地址: **EBI_BA = 0x4001_0000**				
EBI_CTL0	EBI_BA+0x00	R/W	外部总线接口Bank0控制寄存器	0x0000_0000
EBI_TCTL0	EBI_BA+0x04	R/W	外部总线接口Bank0时序控制寄存器	0x0000_0000
EBI_CTL1	EBI_BA+0x10	R/W	外部总线接口Bank1控制寄存器	0x0000_0000
EBI_TCTL1	EBI_BA+0x14	R/W	外部总线接口Bank1时序控制寄存器	0x0000_0000
EBI_CTL2	EBI_BA+0x20	R/W	外部总线接口Bank2控制寄存器	0x0000_0000
EBI_TCTL2	EBI_BA+0x24	R/W	外部总线接口Bank2时序控制寄存器	0x0000_0000

6.30.7 寄存器描述

外部总线接口控制寄存器 (EBI_CTLx)

寄存器	偏移地址	R/W	描述	复位值
EBI_CTL0	EBI_BA+0x00	R/W	外部总线接口Bank0控制寄存器	0x0000_0000
EBI_CTL1	EBI_BA+0x10	R/W	外部总线接口Bank1控制寄存器	0x0000_0000
EBI_CTL2	EBI_BA+0x20	R/W	外部总线接口Bank2控制寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reversed							WBUFEN
23	22	21	20	19	18	17	16
Reversed					TALE		
15	14	13	12	11	10	9	8
Reversed					MCLKDIV		
7	6	5	4	3	2	1	0
Reversed			CACCESS	ADSEPEN	CSPOLINV	DW16	EN

位	描述	
[31:25]	**Reserved**	保留
[24]	**WBUFEN**	**EBI 写缓存使能位** 0 = 禁用EBI 写缓存 1 = 使能EBI 写缓存 **注意:**该位仅在EBI_CTL0寄存器中有效
[23:19]	**Reserved**	保留
[18:16]	**TALE**	**ALE的扩展时间** 控制EBI_ALE高电平时间(tALE) tALE = (TALE+1)*EBI_MCLK. **注意:**该位仅在EBI_CTL0寄存器中有效
[15:11]	**Reserved**	保留
[10:8]	**MCLKDIV**	**外部输出时钟分频器** MCLK频率如下： 000 = HCLK/1. 001 = HCLK/2. 010 = HCLK/4. 011 = HCLK/8. 100 = HCLK/16. 101 = HCLK/32.

		110 = HCLK/64. 111 = HCLK/128.
[7:5]	**Reserved**	保留
[4]	**CACCESS**	**连续数据访问模式** 当连续访问模式使能，为了连续数据传输需要，tASU, tALE 和 tLHD周期会被忽略。 0 =连续数据访问模式禁止. 1 =连续数据访问模式使能.
[3]	**ADSEPEN**	**EBI地址/数据总线分离模式使能位** 0 = EBI地址/数据总线复用模式 1 = EBI地址/数据总线分离模式
[2]	**CSPOLINV**	**片选管脚极性反转** 0 = 片选管脚 (EBI_nCS) 低电平有效 1 = 片选管脚 (EBI_nCS) 高电平有效
[1]	**DW16**	**EBI 数据宽度16-位选择** 0 = EBI 数据宽度是 8-位 1 = EBI 数据宽度是 16-位
[0]	**EN**	**EBI 使能位** 该位是EBI的功能使能位 0 = 禁用EBI 功能 1 = 使能EBI 功能

外部总线接口时序控制寄存器(EBI_TCTLx)

寄存器	偏移地址	R/W	描述	复位值
EBI_TCTL0	EBI_BA+0x04	R/W	外接总线接口Bank0时序控制寄存器	0x0000_0000
EBI_TCTL1	EBI_BA+0x14	R/W	外接总线接口Bank1时序控制寄存器	0x0000_0000
EBI_TCTL2	EBI_BA+0x24	R/W	外接总线接口Bank2时序控制寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved				R2R			
23	22	21	20	19	18	17	16
WAHDOFF	RAHDOFF	Reserved					
15	14	13	12	11	10	9	8
W2X				Reversed	TAHD		
7	6	5	4	3	2	1	0
TACC					Reserved		

位	描述	
[31:30]	**Reserved**	保留.
[27:24]	**R2R**	**读-读之间的空闲状态周期** 读-读(R2R)空闲状态周期= (R2R * EBI_MCLK). 当读动作完成，下一个动作也是读时，插入读-读(R2R)空闲状态周期后，EBI_nCS回空闲状态
[23]	**WAHDOFF**	**EBI写时，访问保持时间禁用** 0 = 在EBI写期间，使能数据访问保持时间(tAHD) 1 = 在EBI写期间，禁用数据访问保持时间(tAHD).
[22]	**RAHDOFF**	**EBI读时，访问保持时间禁用** 0 = 在EBI读期间，使能数据访问保持时间(tAHD) 1 = 在EBI读期间，禁用数据访问保持时间(tAHD).
[21:16]	**Reserved**	保留.
[15:12]	**W2X**	**写之后的空闲状态周期** W2X 空闲状态周期 = (W2X * EBI_MCLK). 当写动作完成，插入W2X空闲状态周期后，EBI_nCS回空闲状态
[11]	**Reserved**	保留.
[10:8]	**TAHD**	**EBI 数据访问保持时间** TAHD 定义数据访问保持时间(tAHD). tAHD = (TAHD +1) * EBI_MCLK.

[7:3]	**TACC**	**EBI 数据访问时间** TACC 定义数据访问时间 (tACC). tACC = (TACC +1) * EBI_MCLK.
[2:0]	**Reserved**	保留.

6.31 USBD USB 1.1设备控制器

6.31.1 概述

此设备中有一套USB 2.0全速设备控制器和收发器。它符合USB 2.0全速设备规范，支持控制/批量/中断/同步传输类型。

在这个设备控制器中，有两个主要接口：APB总线和USB总线，它们来自USB PHY收发器。对于APB总线，CPU可以通过它编程控制寄存器。在这个控制器中有1K字节的内部SRAM作为数据缓冲区。对于输入或输出传输，需要将数据写入SRAM或通过APB接口或SIE从SRAM读取数据。用户需要通过缓冲区分段寄存器（USBD_BUFSEGx）为每个端点缓冲区设置SRAM的有效起始地址。

此控制器中有12个端点。每个端点都可以配置为输入或输出端点。所有的操作包括控制、批量、中断和同步传输都在这个模块中实现。“端点控制”块还用于管理每个端点的数据同步、端点状态、当前起始地址、传输状态和数据缓冲区状态。

这个控制器有四种不同的中断事件。它们是无事件唤醒、设备插件或插件事件、USB事件，如IN ACK, OUT ACK等，和总线事件，如Suspend和Resume等。任何事件都会导致中断，用户只需检查中断事件状态寄存器（USBD_INTSTS）中的相关事件标志，以确认发生了何种中断，然后检查相关的USB端点状态寄存器（USBD_EPSTS0和USBD_EPSTS1）来确认此端点中发生的事件类型。

此USB控制器还支持软件断开连接功能。它用于模拟此设备与主机的断开连接。如果用户启用SE0位（USBD_SE0），USB控制器将强制USB_D+和USB_D-的输出为低电平，并且其功能被禁用。禁用SE0位后，主机将再次枚举USB设备。

更多关于USB的信息，请参考*Universal Serial Bus Specification Revision 1.1*。

6.31.2 特性

- 兼容USB 2.0全速规范
- 提供一个中断向量及5中不同的中断事件(SOF, NEVWK, VBUSDET, USB和BUS)
- 支持控制/批量/中断/同步四种传输
- 支持总线闲置3ms以上切换到总线挂起功能
- 支持12个可配置为控制/批量/中断/同步传输类型的端点和最大1K字节缓冲区
- 提供远程唤醒功能

6.31.3 框图

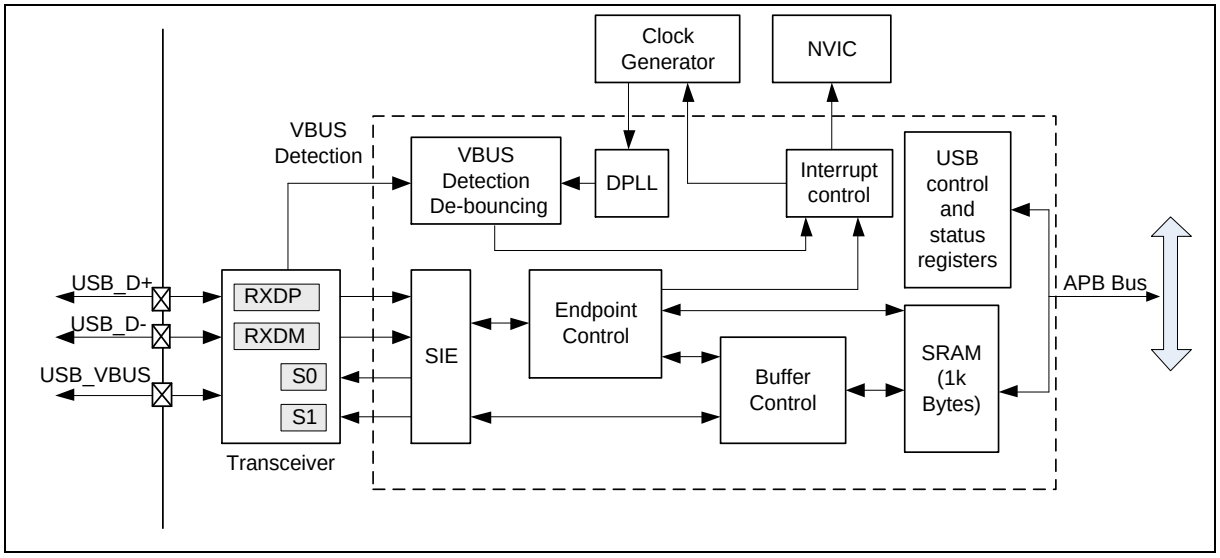


图 6.31-1 USB 框图

6.31.4 M48xID/M48xGA基本配置

USB的帧的功能由USBROLE (SYS_USBPHY[1:0])来设置。内部USB 3.3V LDO可通过使能LDO33EN (SYS_USBPHY[8])打开。这两个配置都是写保护位。当写这些位前，用户必须禁止寄存器保护功能。请参考SYS_REGLCTL寄存器详细描述。USBD设备的时钟由PLL时钟分频所得。用户必须在USB设备控制器使能前，先配置PLL时钟。设置USBDCKEN (CLK_APBCLK0[27])位可使能USBD时钟，设置USBDIV (CLK_CLKDIV0[7:4])对USB时钟预分频，产生合适的时钟频率。

6.31.4.1 USB 1.1设备基本配置

- 时钟源配置
  - 设置PLL控制器(CLK_PLLCTL)
  - 选择USBD设备时钟分频数USBDIV(CLK_CLKDIV0[7:4])
  - 使能USBD外设时钟位USBDCKEN(CLK_APBCLK0[27])
- 复位配置
  - 复位 USBD 控制器位USBDRST(SYS_IPRST1[27])
- 引脚配置

6.31.5 M48xGC/M48xE8基本配置

USB帧的作用由USBROLE (SYS_USBPHY[1:0])决定。内部USB 3.3V LDO可以由LDO33EN (SYS_USBPHY[8])使能。这两个位是写保护位。在写入这些位前，用户必须禁止寄存器保护功能。参考SYS_REGLCTL寄存器详细描述。USBD时钟源来自HIRC48M或PLL。如果用户想要达到无晶振化，必须选择HIRC作为USB时钟源，否则用户必须在启用USB设备控制器之前启用cyrstal的PLL相关配置。设置USBDCKEN (CLK_APBCLK0[27])位来使能USBD时钟和4位预分频USBDIV (CLK_CLKDIV0[7:4])来生产正确的USBD时钟速率。

6.31.5.1 USB 1.1设备基本配置

- 时钟源配置
  - 使能HIRC48M (CLK_PWRCTL[18])或PLL控制器(CLK_PLLCTL)
  - 选择USBD外设时钟的时钟分频器数USBDIV(CLK_CLKDIV0[7:4])
  - 使能USBD外设时钟位USBDCKEN(CLK_APBCLK0[27]).
- 复位配置
  - 复位USBD控制器位USBDRST(SYS_IPRST1[27])
- 引脚配置

组	引脚名称	GPIO	MFP
USB	USB_D+	PA.14	MFP14
	USB_D-	PA.13	MFP14
	USB_OTG_ID	PA.15	MFP14
	USB_VBUS	PA.12	MFP14

6.31.6 功能描述

6.31.6.1 串行接口引擎(SIE)

SIE是设备控制器的前端，用于处理USB协议。SIE的主要工作是向上发送信号到事务处理层.功能包括：

- 数据包识别和事务排序
- SOP,EOP,RESET,RESUME信号侦测/产生
- 时钟/数据分离
- NRZI数据编解码和位传输
- CRC产生和检查(令牌和数据)
- 包ID(PID)生成和检查/解码
- 串-并/并-串转换

6.31.6.2 端点控制

此控制器支持12个端点。每个端点都可以配置为控制、批量、中断或同步传输类型。所有的操作包括控制、批量、中断和同步传输都在这个模块中实现。它还用于管理每个端点的数据顺序同步、端点状态控制、当前端点起始地址、当前事务状态和数据缓冲区状态。

6.31.6.3 数字锁相环(DPLL)

USB数据的传输比特率是12MHz，DPLL使用来自时钟控制器的48MHz时钟源来锁定RXDP和RXDM上的输入数据。12M的比特率时钟也是由DPLL转换而来。

6.31.6.4 VBUS侦测去抖

USB设备有可能经常在主机上被插拔。当USB设备从主机上拔下后，为了监测到它的状态，设备控制器提供了一个硬件去抖USB VBUS检测中断，来避免USB插拔时的抖动问题。USB设备插拔操作10ms后会产生一个VBUS检测中断。用户可以通过读USBD_VBUSDET寄存器的内容知道USB设备的插拔状态

。VBUSDET标志反应了USB BUS 没有消抖情况下的当前状态。如果VBUSDET为1，表示USB线被插入。如果用户轮询该标志来检测USB的状态，需要通过软件来做消抖。

6.31.6.5 中断控制

USB控制器带有一个USB中断向量，该中断包含了四个中断事件（唤醒事件，中断事件，USB事件，BUS事件）。其中唤醒中断事件(NEVWK)发生在当系统从Power-down低功耗模式中唤醒时，（低功耗模式在Power-down控制寄存器CLK_PWRCTL做了定义）。检测中断事件(VBDET)用于USB设备检测中断。USB中断事件用于告知MCU产生了一些USB请求，如IN ACK, OUT ACK等。BUS中断事件用来告知MCU产生了总线事件，如挂起，恢复等。当需要用到这些中断时，必须在USB设备控制器的中断使能控制寄存器(USBD_INTEN)中打开相应位。

当系统在Power- down模式下唤醒后，如果20ms内没有其它USB中断事件发生，则发生唤醒中断。系统进入Power-down后，如果USB唤醒功能使能，USB_VBUS,USB_D+ 和 USB_D-上的任何变化可以唤醒MCU。如果这个变化不是有意而为的，那么只有唤醒中断会发生。若 USB 唤醒超过 20 ms 后，如果没有其他 USB 中断事件发生，则唤醒中断将发生。图6.31-2为唤醒中断的控制流程。

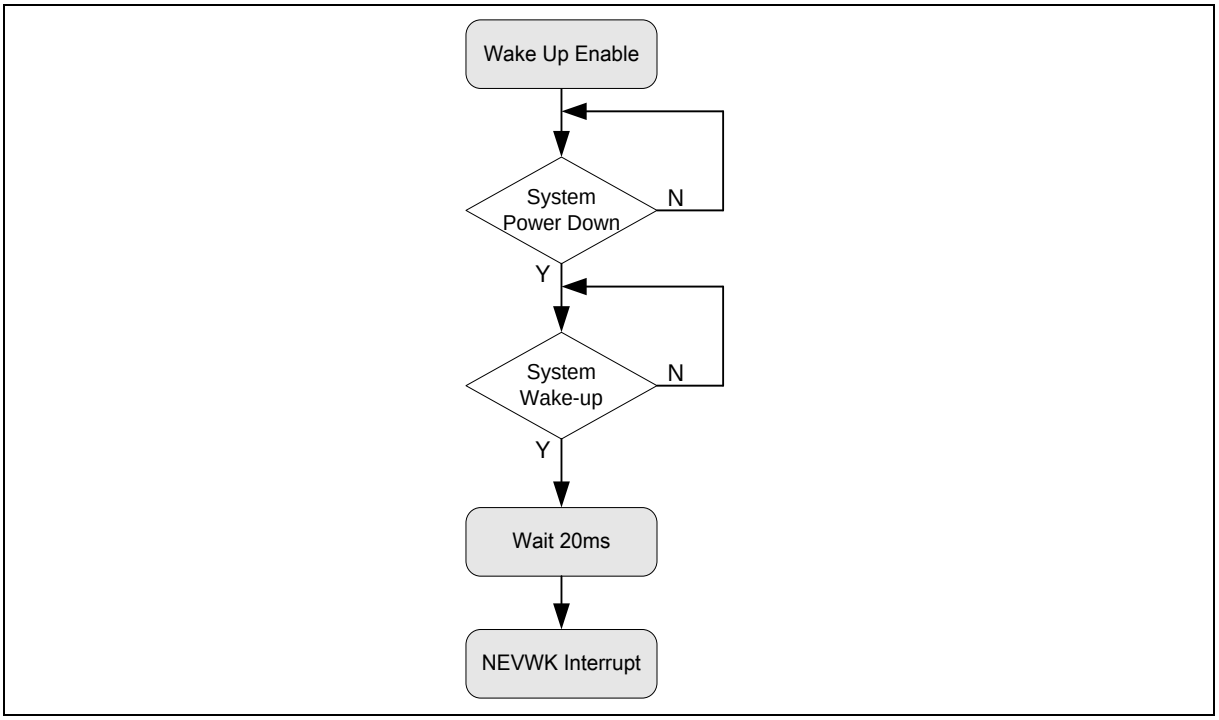


图 6.31-2 唤醒中断操作流程

USB中断事件用于告知MCU USB总线上所发生的事件，MCU可以读取EPSTS位(USBD_EPSTS0和USBD_EPSTS1)和EPEVT11~0 (USBD_INTSTS[27:16])的内容，来知道USB当前的状态以便进行下一步操作。

和USB中断事件一样，BUS中断事件是用于告知MCU一些BUS事件。比如USB复位，挂起，超时溢出以及总线恢复等。用户可以读USBD_ATTR寄存器内容，从而知道USB的总线的状态。

6.31.6.6 省电

在一些特殊情况下，比如挂起，用户可以写0到USBD_ATTR[4]位来手动禁止PHY来省电。

6.31.6.7 缓存控制

USB控制器中总共有1K字节的SRAM，12个端点可以共享这些缓存。在USB模块功能使能前，应先在缓冲段寄存器配置每个端点的有效起始地址。”缓冲区控制”模块就是用于控制每个端点的有效起始地址和它所分配的SRAM缓冲区间大小(在USBD_MXPLDx寄存器定义)。

图6.31-3描述了，根据USBD_BUFSEGx 和 USBD_MXPLDx寄存器中所定义的，各个端点缓冲区起始地址和大小。如果USBD_BUFSEG0被设置为0x08h, USBD_MXPLD0被设为0x40h,那么端点0 所分配的缓冲区的大小就是从USBD_BA+0x108h开始，到USBD_BA+0x148h结束。(注:USBD的SRAM起始地址是从USBD_BA+0x100h开始)。

注: USBD SRAM基地址USBD_BA+0x100h.

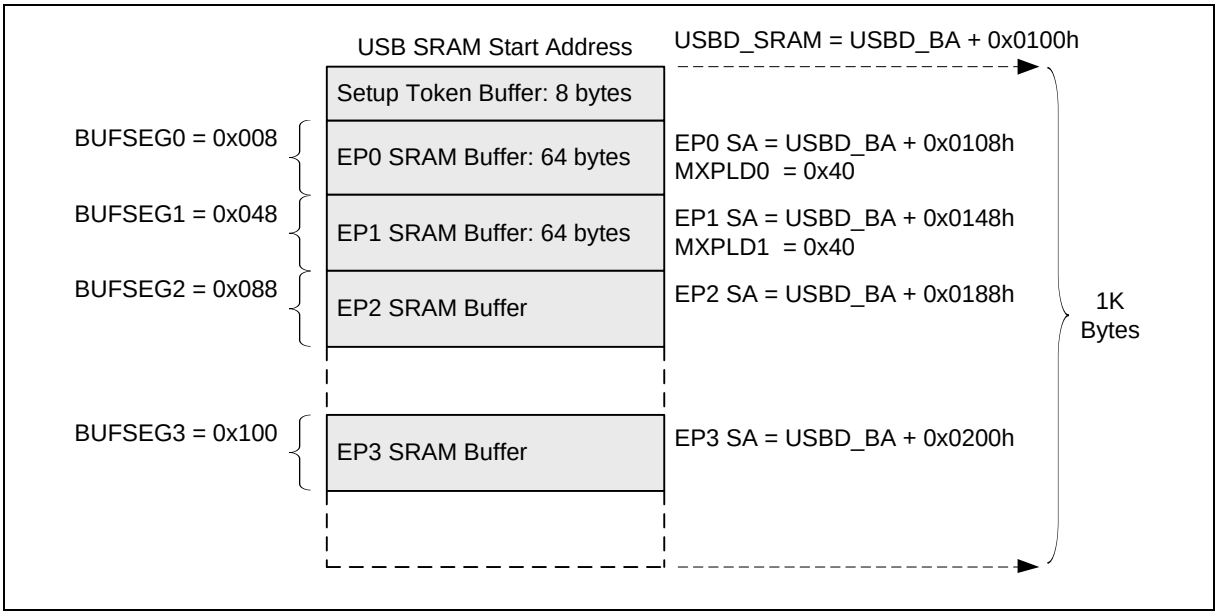


图 6.31-3 端点 SRAM 结构

6.31.6.8 与USB外设通信处理

用户可以使用中断或轮询USBD_INTSTS寄存器来监测USB的数据通信。当通信发生时，USBD_INTSTS寄存器被硬件置位，并向CPU产生一个中断请求(如果相关中断打开)，或者也可以不使用中断方式，用轮询USBD_INTSTS寄存器的相应位的方法来获取事件信息。以下是使用中断方式的控制流程。

当USB主机向设备控制器请求数据时，用户需要预先把相关数据放到指定的端点缓存。填充完数据后，用户需要写实际数据长度到USBD_MXPLDx这个寄存器当中。一旦这个寄存器数据被写入，内部信号“In_Rdy”信号会被设置，当接收到主机发来的IN token信号后，缓冲区数据会被立即传送出去。需要注意的在指定数据发送完成后，“In_Rdy”信号会由硬件自动清除。

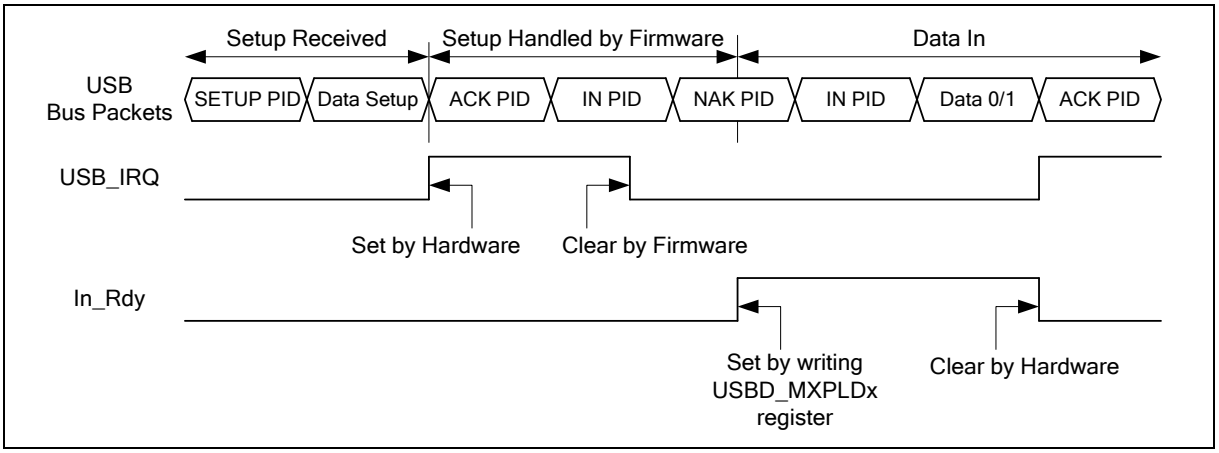


图 6.31-4 Setup 事务后接着是 Data In 事务

相应的，当USB主机想要发送数据到从设备控制器的输出端点时，硬件会把数据填充到指定的端点缓冲区。通信完成后，硬件会在端点对应的USBD_MXPLDx寄存器中自动记录数据长度，并清除“Out_Rdy”信号。这将会避免硬件在用户没有取走当前数据时又接收到下一笔数据。一旦用户处理了这次通信时，由软件写入特定的寄存器USBD_MXPLDx，以再次产生“Out_Rdy”信号来接收下一次通信。

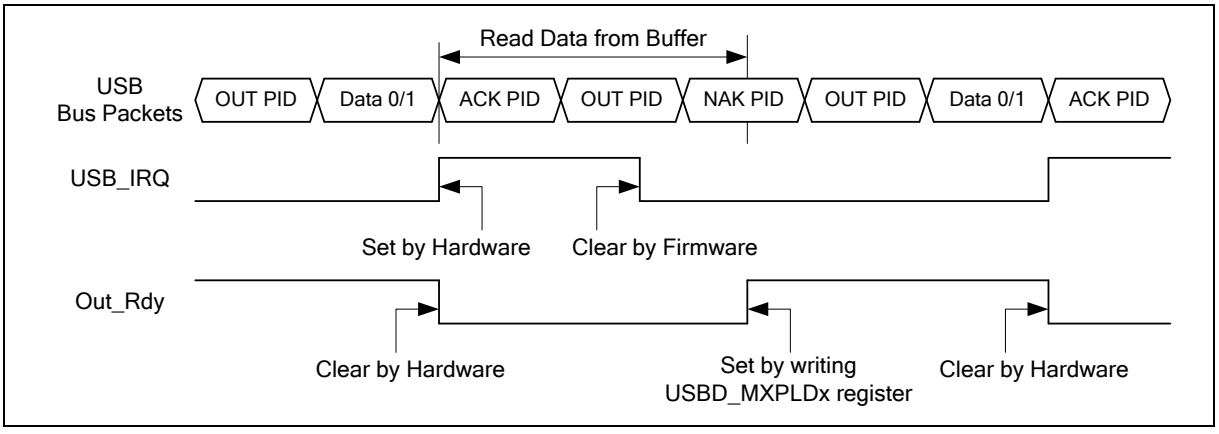


图 6.31-5 数据输出

6.31.6.9 链路电源管理(LPM)

电源管理就是类似于挂起/恢复功能，但是电源状态过渡延时只有10us（取代了USB2.0挂起/恢复大于20ms的延时）。

新快速机制用于总线上根端口从使能状态（L0）到新休眠状态（L1）的转换。L0和L1状态详见表6.31-1，寄存器 USBD_ATTR & USBD_LPMATTR可以让用户了解LPM机制的当前电源状态/

LPM 状态	描述
L0(开)	在这个状态，端口被使能可以进行信号的传输。L0下的端口不是在启用状态就是在空闲状态。所谓启用就是正在发送或是接收数据。这种状态下开始帧包（SOF）由主机产生，其速率与客户设备一致。
L1(睡眠)	类似于下面的L2使用，但是还是有一点差别。进入L1需要向hub或是主机发出请求。一个LPM事务下发到设备。仅当在设备相应ACK握手后请求事务才能发生。通过远程唤醒、恢复信号、复位信号或是断开连接离开L1状态。连接的设备如L2一样但是L1不会从VBUS消耗多少功耗。当在L1时，主机或是设备可以初始化复位信号。尽管信号恢复如同L2，信号周期和L1到L0过渡延时更短。
L2(挂起)	该状态就是USB2.0挂起状态。向hub或是主机端口发送命令进入L2。设备发现挂起条件是观察到3ms非活动状态。结果状态不是低速就是全速空闲。L2会从VBUS消耗很多功耗。通过远程唤醒、恢复信号、复位信号或是断开连接离开L2状态。
L3(关闭)	这种状态下，端口不能进行数据传输。相当于断电，断开连接和禁用。

表 6.31-1 USB 链路电源管理(Lx)状态

状态转化过程请参考图6.31-6，更多的USB链路电源管理（LPM）信息请参考USB2.0链路电源管理ECN。

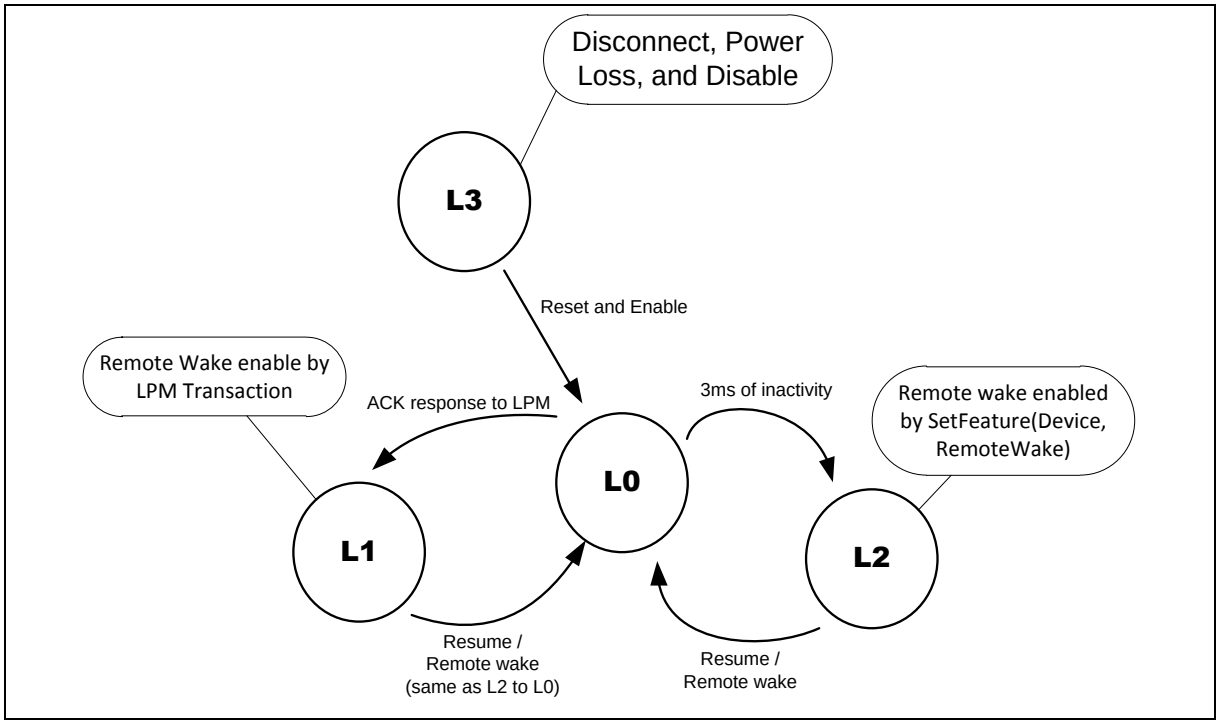


图 6.31-6LPM 状态转换图

6.31.7 寄存器映射

R: 只读, **W**: 只写, **R/W**: 读/写

寄存器	地址偏移	R/W	描述	复位值
USBD基地址： USBD_BA = 0x400C_0000				
USBD_INTEN	USBD_BA+0x000	R/W	USB设备中断使能寄存器	0x0000_0000
USBD_INTSTS	USBD_BA+0x004	R/W	USB设备中断事件状态寄存器	0x0000_0000
USBD_FADDR	USBD_BA+0x008	R/W	USB设备功能地址寄存器	0x0000_0000
USBD_EPSTS	USBD_BA+0x00C	R	USB设备端点状态寄存器	0x0000_0000
USBD_ATTR	USBD_BA+0x010	R/W	USB设备总线状态和属性寄存器	0x0000_0040
USBD_VBUSDET	USBD_BA+0x014	R	USB设备VBUS检测寄存器	0x0000_0000
USBD_STBUFSEG	USBD_BA+0x018	R/W	SETUP令牌包缓存段寄存器	0x0000_0000
USBD_EPSTS0	USBD_BA+0x020	R	USB设备端点状态寄存器0	0x0000_0000
USBD_EPSTS1	USBD_BA+0x024	R	USB设备端点状态寄存器1	0x0000_0000
USBD_LPMATTR	USBD_BA+0x088	R	USB LPM属性寄存器	0x0000_0000
USBD_FN	USBD_BA+0x08C	R	USB帧号寄存器	0x0000_0XXX
USBD_SE0	USBD_BA+0x090	R/W	USB设备驱动SE0控制寄存器	0x0000_0001
USBD_BUFSEG0	USBD_BA+0x500	R/W	端点0缓存段寄存器	0x0000_0000
USBD_MXPLD0	USBD_BA+0x504	R/W	端点0最大载荷寄存器	0x0000_0000
USBD_CFG0	USBD_BA+0x508	R/W	端点0配置寄存器	0x0000_0000
USBD_CFGP0	USBD_BA+0x50C	R/W	端点0设置Stall和清除In/Out 准备控制寄存器	0x0000_0000
USBD_BUFSEG1	USBD_BA+0x510	R/W	端点1缓存段寄存器	0x0000_0000
USBD_MXPLD1	USBD_BA+0x514	R/W	端点1最大载荷寄存器	0x0000_0000
USBD_CFG1	USBD_BA+0x518	R/W	端点1配置寄存器	0x0000_0000
USBD_CFGP1	USBD_BA+0x51C	R/W	端点1设置Stall和清除In/Out 准备控制寄存器	0x0000_0000
USBD_BUFSEG2	USBD_BA+0x520	R/W	端点2缓存段寄存器	0x0000_0000
USBD_MXPLD2	USBD_BA+0x524	R/W	端点2最大载荷寄存器	0x0000_0000
USBD_CFG2	USBD_BA+0x528	R/W	端点2配置寄存器	0x0000_0000
USBD_CFGP2	USBD_BA+0x52C	R/W	端点2设置Stall和清除In/Out 准备控制寄存器	0x0000_0000
USBD_BUFSEG3	USBD_BA+0x530	R/W	端点3缓存段寄存器	0x0000_0000

USBD_MXPLD3	USBD_BA+0x534	R/W	端点3最大载荷寄存器	0x0000_0000
USBD_CFG3	USBD_BA+0x538	R/W	端点3配置寄存器	0x0000_0000
USBD_CFGP3	USBD_BA+0x53C	R/W	端点3设置Stall和清除In/Out 准备控制寄存器	0x0000_0000
USBD_BUFSEG4	USBD_BA+0x540	R/W	端点4缓存段寄存器	0x0000_0000
USBD_MXPLD4	USBD_BA+0x544	R/W	端点4最大载荷寄存器	0x0000_0000
USBD_CFG4	USBD_BA+0x548	R/W	端点4配置寄存器	0x0000_0000
USBD_CFGP4	USBD_BA+0x54C	R/W	端点4设置Stall和清除In/Out 准备控制寄存器	0x0000_0000
USBD_BUFSEG5	USBD_BA+0x550	R/W	端点5缓存段寄存器	0x0000_0000
USBD_MXPLD5	USBD_BA+0x554	R/W	端点5最大载荷寄存器	0x0000_0000
USBD_CFG5	USBD_BA+0x558	R/W	端点5配置寄存器	0x0000_0000
USBD_CFGP5	USBD_BA+0x55C	R/W	端点5设置Stall和清除In/Out 准备控制寄存器	0x0000_0000
USBD_BUFSEG6	USBD_BA+0x560	R/W	端点6缓存段寄存器	0x0000_0000
USBD_MXPLD6	USBD_BA+0x564	R/W	端点6最大载荷寄存器	0x0000_0000
USBD_CFG6	USBD_BA+0x568	R/W	端点6配置寄存器	0x0000_0000
USBD_CFGP6	USBD_BA+0x56C	R/W	端点6设置Stall和清除In/Out 准备控制寄存器	0x0000_0000
USBD_BUFSEG7	USBD_BA+0x570	R/W	端点7缓存段寄存器	0x0000_0000
USBD_MXPLD7	USBD_BA+0x574	R/W	端点7最大载荷寄存器	0x0000_0000
USBD_CFG7	USBD_BA+0x578	R/W	端点7配置寄存器	0x0000_0000
USBD_CFGP7	USBD_BA+0x57C	R/W	端点7设置Stall和清除In/Out准备控制寄存器	0x0000_0000
USBD_BUFSEG8	USBD_BA+0x580	R/W	端点8缓存段寄存器	0x0000_0000
USBD_MXPLD8	USBD_BA+0x584	R/W	端点8最大载荷寄存器	0x0000_0000
USBD_CFG8	USBD_BA+0x588	R/W	端点8配置寄存器	0x0000_0000
USBD_CFGP8	USBD_BA+0x58C	R/W	端点8设置Stall和清除In/Out准备控制寄存器	0x0000_0000
USBD_BUFSEG9	USBD_BA+0x590	R/W	端点9缓存段寄存器	0x0000_0000
USBD_MXPLD9	USBD_BA+0x594	R/W	端点9最大载荷寄存器	0x0000_0000
USBD_CFG9	USBD_BA+0x598	R/W	端点9配置寄存器	0x0000_0000
USBD_CFGP9	USBD_BA+0x59C	R/W	端点9设置Stall和清除In/Out准备控制寄存器	0x0000_0000
USBD_BUFSEG10	USBD_BA+0x5A0	R/W	端点10缓存段寄存器	0x0000_0000
USBD_MXPLD10	USBD_BA+0x5A4	R/W	端点10最大载荷寄存器	0x0000_0000
USBD_CFG10	USBD_BA+0x5A8	R/W	端点10配置寄存器	0x0000_0000

USBD_CFGP10	USBD_BA+0x5AC	R/W	端点10设置Stall和清除In/Out准备控制寄存器	0x0000_0000
USBD_BUFSEG11	USBD_BA+0x5B0	R/W	端点11缓存段寄存器	0x0000_0000
USBD_MXPLD11	USBD_BA+0x5B4	R/W	端点11最大载荷寄存器	0x0000_0000
USBD_CFG11	USBD_BA+0x5B8	R/W	端点11配置寄存器	0x0000_0000
USBD_CFGP11	USBD_BA+0x5BC	R/W	端点11设置Stall和清除In/Out准备控制寄存器	0x0000_0000

内存类型	**地址**	**大小**	**描述**
USBD_BA = 0x400C_0000			
USBD_SRAM	USBD_BA+0x100 ~ USBD_BA+0x4FF	1024 Bytes	SRAM用于整个端点缓冲区。 详细内容请参考章节6.31.6.7，端点SRAM结构和描述

6.31.8 寄存器描述

USB中断使能寄存器 (USBD_INTEN)

寄存器	地址偏移	R/W	描述	复位值
USBD_INTEN	USBD_BA+0x000	R/W	USB设备中断使能寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
INNAKEN	Reserved						WKEN
7	6	5	4	3	2	1	0
Reserved			SOFIEN	NEVWKIEN	VBDETIEN	USBIEN	BUSIEN

位	描述	
[31:16]	Reserved	保留
[15]	INNAKEN	**收到IN Token 时，激活的 NAK功能及状态** 0 = 当从设备收到IN token 应答 NAK 时，IN NAK 状态不会被更新到端点状态寄存器 USBD_EPSTS0 和 USBD_EPSTS1，所以不会产生USB中断事件 1 = 当从设备收到IN token 时应答NAK，IN NAK 的状态被更新到端点状态寄存器 USBD_EPSTS0 和 USBD_EPSTS1，并发生 USB 中断事件
[14:9]	Reserved	保留
[8]	WKEN	**唤醒功能使能位** 0 = USB唤醒功能禁止 1 = USB唤醒功能使能
[7:5]	Reserved	保留
[4]	SOFIEN	**开始帧中断使能位** 0 = SOF中断禁止 1 = SOF中断使能
[3]	NEVWKIEN	**USB 无事件唤醒中断使能位** 0 = 无事件唤醒中断禁止 1 = 无事件唤醒中断使能
[2]	VBDETIEN	**VBUS检测中断使能位** 0 = VBUS检测中断禁止 1 = VBUS检测中断使能
[1]	USBIEN	**USB事件中断使能位**

		0 = USB事件中断禁止 1 = USB事件中断使能
[0]	**BUSIEN**	**Bus 事件中断使能位** 0 = BUS 事件中断禁止 1 = BUS 事件中断使能

USB中断事件状态寄存器 (USBD_INTSTS)

寄存器	地址偏移	R/W	描述	复位值
USBD_INTSTS	USBD_BA+0x004	R/W	USB设备中断事件状态寄存器	0x0000_0000

31	30	29	28	27	26	25	24
SETUP	Reserved			EPEVT11	EPEVT10	EPEVT9	EPEVT8
23	22	21	20	19	18	17	16
EPEVT7	EPEVT6	EPEVT5	EPEVT4	EPEVT3	EPEVT2	EPEVT1	EPEVT0
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved		SOFIF		NEVWKIF	VBDETIF	USBIF	BUSIF

位	描述	
[31]	**SETUP**	**Setup事件状态** 0 = 没有SETUP事件 1 = 发生SETUP事件, 写1 到 USBD_INTSTS[31]位清零
[30:28]	**Reserved**	保留
[27]	**EPEVT11**	**端点11的USB事件状态** 0 = 端点11没有事件发生 1 = 端点11有USB 事件发生, 检查USBD_EPSTS1[15:12]位可以知道发生了哪种USB事件,可以对USBD_INTSTS[27]位或USBD_INTSTS[1]位写1来清零
[26]	**EPEVT10**	**端点10的USB事件状态** 0 = 端点10没有事件发生 1 = 端点10有USB 事件发生, 检查USBD_EPSTS1[11 :8]位可以知道发生了哪种USB事件,可以对USBD_INTSTS[26]位或USBD_INTSTS[1]位写1来清零
[25]	**EPEVT9**	**端点9的USB事件状态** 0 = 端点9没有事件发生 1 = 端点9有USB 事件发生, 检查USBD_EPSTS1[7 :4]位可以知道发生了哪种USB事件,可以对USBD_INTSTS[25]位或USBD_INTSTS[1]位写1来清零
[24]	**EPEVT8**	**端点8的USB事件状态** 0 = 端点8没有事件发生 1 = 端点8有USB 事件发生, 检查USBD_EPSTS1[3 :0]位可以知道发生了哪种USB事件,可以对USBD_INTSTS[24]位或USBD_INTSTS[1]位写1来清零
[23]	**EPEVT7**	**端点7的USB事件状态** 0 = 端点7没有事件发生 1 = 端点7有USB 事件发生, 检查USBD_EPSTS0[31:28]位可以知道发生了哪种USB事件,可以对USBD_INTSTS[23] 位或USBD_INTSTS[1]位写1来清零

[22]	**EPEVT6**	**端点 6的USB 事件状态** 0 = 端点6没有事件发生. 1 = 端点6有USB 事件发生, 检查USBD_EPSTS0[27:24]位可以知道发生了哪种USB事件, 可以对 USBD_INTSTS[22] 位或 USBD_INTSTS[1]位写1来清零
[21]	**EPEVT5**	**端点 5的USB 事件状态** 0 = 端点5没有事件发生 1 = 端点5有USB 事件发生, 检查USBD_EPSTS0[23:20]位可以知道发生了哪种USB事件, 可以对 USBD_INTSTS[21]位或 USBD_INTSTS[1]位写1来清零
[20]	**EPEVT4**	**端点4的USB 事件状态** 0 = 端点4没有事件发生 1 = 端点4有USB 事件发生, 检查USBD_EPSTS0[19:16]位可以知道发生了哪种USB事件, 可以对USBD_INTSTS[20] 位或 USBD_INTSTS[1]位写1来清零
[19]	**EPEVT3**	**端点 3的USB 事件状态** 0 = 端点3没有事件发生 1 = 端点3有USB 事件发生, 检查USBD_EPSTS0[15:12]位可以知道发生了哪种USB事件,, 可以对USBD_INTSTS[19] 位或USBD_INTSTS[1]位写1来清零
[18]	**EPEVT2**	**端点2的USB 事件状态** 0 = 端点2没有事件发生 1 = 端点2有USB 事件发生, 检查USBD_EPSTS0[11:8]位可以知道发生了哪种USB事件, 可以对USBD_INTSTS[18] 位或USBD_INTSTS[1]位写1来清零
[17]	**EPEVT1**	**端点 1的USB 事件状态** 0 = 端点1没有事件发生 1 = 端点1有USB 事件发生, 检查USBD_EPSTS0[7:4]位可以知道发生了哪种USB事件, 可以对USBD_INTSTS[17]位或USBD_INTSTS[1]位写1来清零
[16]	**EPEVT0**	**端点 0的USB 事件状态** 0 = 端点0没有事件发生. 1 = 端点0有USB 事件发生, 检查USBD_EPSTS0[3:0]位可以知道发生了哪种USB事件,可以对USBD_INTSTS[16] 位或USBD_INTSTS[1]位写1来清零
[15:5]	**Reserved**	保留
[4]	**SOFIF**	**开始帧中断状态** 0 = 没有SOF事件发生 1 = 有SOF事件发生, 对USBD_INTSTS[4]位写1来清零
[3]	**NEVWKIF**	**无事件唤醒中断状态** 0 = 没有唤醒事件发生 1 = 有无唤醒事件发生, 对USBD_INTSTS[3]位写1来清零
[2]	**VBDETIF**	**VBUS检测中断状态** 0 = 没有USB设备连接/分离事件发生 1 = USB总线上有连接/分离事件发生，对USBD_INTSTS[2]位写1清零
[1]	**USBIF**	**USB 中断事件状态** USB 事件包括在总线上的SETUP Token, IN Token, OUT ACK, ISO IN, 或 ISO OUT事件 0 = 没有USB事件发生 1 = 有USB事件发生，检查EPSTS0~5[2:0]位可以知道发生了哪种USB事件。对

		USBD_INTSTS[1] 位或对EPSTS0~11和SETUP (USBD_INTSTS[31])位写1清零
[0]	**BUSIF**	**BUS中断事件状态** BUS中断事件说明总线上发生了挂起或恢复功能. 0 = 没有BUS中断事件发生 1 = 发生了Bus中断事件; 检查 USBD_ATTR[3:0] 位可以知道发生了哪种BUS中断事件，对USBD_INTSTS[0]位写1清零

USB设备功能地址寄存器 (USBD_FADDR)

一个7位值USB总线上设备地址

寄存器	地址偏移	R/W	描述	复位值
USBD_FADDR	USBD_BA+0x008	R/W	USB设备功能地址寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved	FADDR						

位	描述	
[31:7]	Reserved	保留
[6:0]	FADDR	USB设备功能地址

USB设备端点状态寄存器 (USBD_EPSTS)

寄存器	地址偏移	R/W	描述	复位值
USBD_EPSTS	USBD_BA+0x00C	R	USB设备端点状态寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
OV	**Reserved**						

位	描述	
[31:8]	**Reserved**	保留
[7]	**OV**	**溢出** 用于指示所接收数据长度是否超过最大负荷范围 0 = 未超出 1 = 主机发送数据超出MXPLD寄存器设置范围，或Setup 数据超过8个字节
[6:0]	**Reserved**	保留

USB总线状态和属性寄存器 (USBD_ATTR)

寄存器	地址偏移	R/W	描述	复位值
USBD_ATTR	USBD_BA+0x010	R/W	USB设备总线状态和属性寄存器	0x0000_0040

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved		L1RESUME	L1SUSPEND	LPMACK	BYTEM	PWRDN	DPPUEN
7	6	5	4	3	2	1	0
USBEN	Reserved	RWAKEUP	PHYEN	TOUT	RESUME	SUSPEND	USBRST

位	描述	
[31:14]	**Reserved**	保留
[13]	**L1RESUME**	**LPM L1 恢复(只读)** 0 = 总线没有LPM L1状态恢复 1 = LPM L1状态从挂起恢复
[12]	**L1SUSPEND**	**LPM L1 挂起(只读)** 0 = 总线没有LPM L1状态挂起 1 = 该位在LPM命令进入L1状态被成功接收和应答的时候由硬件置位
[11]	**LPMACK**	**LPM Token 应答使能位** NYET/ACK仅在一个成功的LPM事务后被返回也就是在EXT token 和LPM token和有效的bLinkState = 0001 (L1)被接收时没有出错，否则ERROR 和 STALL分别会自动地被返回 0=有效的 LPM Token 不会被应答 1=有效的 LPM Token 会被应答
[10]	**BYTEM**	**CPU 存取 USB SRAM 大小模式选择** 0 = 字模式: CPU到 USB SRAM 的数据传输必须以字为单位 1 = 字节模式: CPU到 USB SRAM 的数据传输必须以字节为单位
[9]	**PWRDN**	保留
[8]	**DPPUEN**	**USB_DP管脚上拉电阻使能位** 0 = USB_D+ 脚上的上拉电阻禁止. 1 = USB_D+ 脚上的上拉电阻打开.
[7]	**USBEN**	**USB 控制器使能位** 0 = USB 控制器禁止. 1 = USB 控制器使能

[6]	**Reserved**	保留
[5]	**RWAKEUP**	**远程唤醒** 0 = 把USB总线从 K 状态释放 1 = 强迫USB 总线到 K (USB_D+ low, USB_D- high) 状态, 用于远程唤醒
[4]	**PHYEN**	**PHY 收发器使能位** 0 = PHY 收发器功能禁止. 1 = PHY收发器功能使能.
[3]	**TOUT**	**时间溢出状态(只读)** 0 = 没有时间溢出情况发生 1 = 超过18个位长度计数器时间，总线上仍没有数据回复
[2]	**RESUME**	**总线恢复状态(只读)** 0 = 总线没有恢复. 1 = 总线从挂起状态恢复
[1]	**SUSPEND**	**挂起状态(只读)** 0 = 总线没有挂起事件发生. 1 = 总线闲置状态超过3ms, 有可能是从设备被拔下或主机进入了睡眠模式
[0]	**USBRST**	**USB 总线复位状态(只读)** 0 = 总线没有复位 1 = 总线有复位，此时SE0 (single-ended 0)已超过 2.5us

USB设备VBUS检测寄存器(USBD_VBUSDET)

寄存器	地址偏移	R/W	描述	复位值
USBD_VBUSDET	USBD_BA+0x014	R	USB设备VBUS检测寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved							**VBUSDET**

位	描述	
[31:1]	**Reserved**	保留
[0]	**VBUSDET**	**设备VBUS 检测** 0 = 控制器没有连接到USB主机 1 = 控制器连接到到USB主机

USB SETUP令牌包缓存段寄存器(USBD_STBUFSEG)

寄存器	地址偏移	R/W	描述	复位值
USBD_STBUFSEG	USBD_BA+0x018	R/W	SETUP令牌包缓存段寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							**STBUFSEG**
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
STBUFSEG					**Reserved**		

位	描述	
[31:9]	**Reserved**	保留
[8:3]	**STBUFSEG**	**令牌包缓存段设置位** 用于指示 SETUP 令牌包在USB设备SRAM中的起始地址，它的有效起始地址是USBD_SRAM 地址 + {STBUFSEG[8:3], 3'b000} 此处USBD_SRAM 地址 = USBD_BA+0x100h 注:该内容只适用于SETUP令牌包
[2:0]	**Reserved**	保留

USB设备端点状态寄存器0 (USBD_EPSTS0)

寄存器	地址偏移	R/W	描述	复位值
USBD_EPSTS0	USBD_BA+0x020	R	USB设备端点状态寄存器0	0x0000_0000

31	30	29	28	27	26	25	24
EPSTS7				**EPSTS6**			
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
EPSTS5				**EPSTS4**			
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
EPSTS3				**EPSTS2**			
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
EPSTS1				**EPSTS0**			

位	描述	
[31:28]	**EPSTS7**	**端点7状态** 这些位用于指示该端点的当前状态 0000 = In ACK 0001 = In NAK 0010 = Out Packet Data0 ACK 0011 = 设置ACK 0110 = Out Packet Data1 ACK 0111 = 同步传输结束
[27:24]	**EPSTS6**	**端点6 状态** 这些位用于指示该端点的当前状态 0000 = In ACK 0001 = In NAK 0010 = Out Packet Data0 ACK 0011 = Setup ACK 0110 = Out Packet Data1 ACK 0111 = 同步传输结束
[23:20]	**EPSTS5**	**端点5状态** 这些位用于指示该端点的当前状态 0000 = In ACK 0001 = In NAK 0010 = Out Packet Data0 ACK 0011 = Setup ACK 0110 = Out Packet Data1 ACK 0111 = 同步传输结束

[19:16]	**EPSTS4**	**端点4 状态** 这些位用于指示该端点的当前状态 0000 = In ACK. 0001 = In NAK. 0010 = Out Packet Data0 ACK. 0011 = Setup ACK. 0110 = Out Packet Data1 ACK. 0111 = 同步传输结束.
[15:12]	**EPSTS3**	**端点3状态** 这些位用于指示该端点的当前状态 0000 = In ACK. 0001 = In NAK. 0010 = Out Packet Data0 ACK. 0011 = Setup ACK. 0110 = Out Packet Data1 ACK. 0111 = 同步传输结束.
[11:8]	**EPSTS2**	**端点2状态** 这些位用于指示该端点的当前状态 0000 = In ACK. 0001 = In NAK. 0010 = Out Packet Data0 ACK. 0011 = Setup ACK. 0110 = Out Packet Data1 ACK. 0111 = 同步传输结束.
[7:4]	**EPSTS1**	**端点1状态** 这些位用于指示该端点的当前状态 0000 = In ACK. 0001 = In NAK. 0010 = Out Packet Data0 ACK. 0011 = Setup ACK. 0110 = Out Packet Data1 ACK. 0111 = 同步传输结束.
[3:0]	**EPSTS0**	**端点0状态** 这些位用于指示该端点的当前状态 0000 = In ACK. 0001 = In NAK. 0010 = Out Packet Data0 ACK. 0011 = Setup ACK. 0110 = Out Packet Data1 ACK. 0111 = 同步传输结束.

USB设备端点状态寄存器1 (USBD_EPSTS1)

寄存器	地址偏移	R/W	描述	复位值
USBD_EPSTS1	USBD_BA+0x024	R	USB设备端点状态寄存器1	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
EPSTS11				**EPSTS10**			
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
EPSTS9				**EPSTS8**			

位	描述	
[31:16]	**Reserved**	保留
[15:12]	**EPSTS11**	**端点11 状态** 这些位用于指示该端点的当前状态 0000 = In ACK. 0001 = In NAK. 0010 = Out Packet Data0 ACK. 0011 = Setup ACK. 0110 = Out Packet Data1 ACK. 0111 =同步传输结束.
[11:8]	**EPSTS10**	**端点10 状态** 这些位用于指示该端点的当前状态 0000 = In ACK. 0001 = In NAK. 0010 = Out Packet Data0 ACK. 0011 = Setup ACK. 0110 = Out Packet Data1 ACK. 0111 =同步传输结束.
[7:4]	**EPSTS9**	**端点9状态** 这些位用于指示该端点的当前状态 0000 = In ACK. 0001 = In NAK. 0010 = Out Packet Data0 ACK. 0011 = Setup ACK. 0110 = Out Packet Data1 ACK.

		0111 =同步传输结束.
[3:0]	**EPSTS8**	**端点8状态** 这些位用于指示该端点的当前状态 0000 = In ACK. 0001 = In NAK. 0010 = Out Packet Data0 ACK. 0011 = Setup ACK. 0110 = Out Packet Data1 ACK. 0111 =同步传输结束.

USB LPM属性寄存器(USBD_LPMATTR)

寄存器	地址偏移	R/W	描述	复位值
USBD_LPMATTR	USBD_BA+0x088	R	USB LPM属性寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							**LPMRWAKUP**
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
LPMBESL				**LPMLINKSTS**			

位	描述	
[31:9]	**Reserved**	保留
[8]	**LPMRWAKUP**	**LPM 远程唤醒** 该位包含接收的最后ACK LPM Token的远程唤醒值
[7:4]	**LPMBESL**	**LPM最大力度服务延时** 该位包含接收的最后ACK LPM Token的BESL值
[3:0]	**LPMLINKSTS**	**LPM 链路状态** 该位包含接收的最后ACK LPM Token的链路状态

USB帧号寄存器(USBD_FN)

寄存器	地址偏移	R/W	描述	复位值
USBD_FN	USBD_BA+0x08C	R	USB帧号寄存器	0x0000_0XXX

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved					**FN**		
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
FN							

位	描述	
[31:11]	**Reserved**	保留
[10:0]	**FN**	**帧编号** 该域包含了最近一次收到的SOF包的11位的帧号

USB设备驱动SE0控制寄存器(USBD_SE0)

寄存器	地址偏移	R/W	描述	复位值
USBD_SE0	USBD_BA+0x090	R/W	USB设备驱动SE0控制寄存器	0x0000_0001

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved							**SE0**

位	描述	
[31:1]	**Reserved**	保留
[0]	**SE0**	**在USB总线上驱动单端0控制位** 当USB_D+和USB_D-都拉低时，为单端0(SE0) 0 = 正常操作 1 = 强制USB PHY 收发器驱动SE0

USB缓存段寄存器(USB_BUFSEGx)

寄存器	地址偏移	R/W	描述	复位值
USBD_BUFSEG0	USBD_BA+0x500	R/W	端点0缓存段寄存器	0x0000_0000
USBD_BUFSEG1	USBD_BA+0x510	R/W	端点1缓存段寄存器	0x0000_0000
USBD_BUFSEG2	USBD_BA+0x520	R/W	端点2缓存段寄存器	0x0000_0000
USBD_BUFSEG3	USBD_BA+0x530	R/W	端点3缓存段寄存器	0x0000_0000
USBD_BUFSEG4	USBD_BA+0x540	R/W	端点4缓存段寄存器	0x0000_0000
USBD_BUFSEG5	USBD_BA+0x550	R/W	端点5缓存段寄存器	0x0000_0000
USBD_BUFSEG6	USBD_BA+0x560	R/W	端点6缓存段寄存器	0x0000_0000
USBD_BUFSEG7	USBD_BA+0x570	R/W	端点7缓存段寄存器	0x0000_0000
USBD_BUFSEG8	USBD_BA+0x580	R/W	端点8缓存段寄存器	0x0000_0000
USBD_BUFSEG9	USBD_BA+0x590	R/W	端点9缓存段寄存器	0x0000_0000
USBD_BUFSEG10	USBD_BA+0x5A0	R/W	端点10缓存段寄存器	0x0000_0000
USBD_BUFSEG11	USBD_BA+0x5B0	R/W	端点11缓存段寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							**BUFSEG**
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
BUFSEG					**Reserved**		

位	描述	
[31:9]	**Reserved**	保留
[8:3]	**BUFSEG**	**端点缓存段设置位** 用于指示每个端点在USB SRAM 的偏移地址，端点的有效起始地址是 USBD_SRAM 地址 + { BUFSEG[8:3], 3'b000} 此处USBD_SRAM 地址 = USBD_BA+0x100h 请参考章节6.31.6.7，查阅端点SRAM的结构和相关描述
[2:0]	**Reserved**	保留

USB最大负荷设置寄存器(USB_MXPLDx)

寄存器	地址偏移	R/W	描述	复位值
USBD_MXPLD0	USBD_BA+0x504	R/W	端点0最大负荷设置寄存器	0x0000_0000
USBD_MXPLD1	USBD_BA+0x514	R/W	端点1最大负荷设置寄存器	0x0000_0000
USBD_MXPLD2	USBD_BA+0x524	R/W	端点2最大负荷设置寄存器	0x0000_0000
USBD_MXPLD3	USBD_BA+0x534	R/W	端点3最大负荷设置寄存器	0x0000_0000
USBD_MXPLD4	USBD_BA+0x544	R/W	端点4最大负荷设置寄存器	0x0000_0000
USBD_MXPLD5	USBD_BA+0x554	R/W	端点5最大负荷设置寄存器	0x0000_0000
USBD_MXPLD6	USBD_BA+0x564	R/W	端点6最大负荷设置寄存器	0x0000_0000
USBD_MXPLD7	USBD_BA+0x574	R/W	端点7最大负荷设置寄存器	0x0000_0000
USBD_MXPLD8	USBD_BA+0x584	R/W	端点8最大负荷设置寄存器	0x0000_0000
USBD_MXPLD9	USBD_BA+0x594	R/W	端点9最大负荷设置寄存器	0x0000_0000
USBD_MXPLD10	USBD_BA+0x5A4	R/W	端点10最大负荷设置寄存器	0x0000_0000
USBD_MXPLD11	USBD_BA+0x5B4	R/W	端点11最大负荷设置寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							**MXPLD**
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
MXPLD							

位	描述	
[31:9]	**Reserved**	保留
[8:0]	**MXPLD**	**最大有效负荷位** 定义了发送到主机(IN token)的实际数据长度，或从主机接收到数据 (OUT token)的实际长度. 也用于指示IN token时做好了发送数据准备，或OUT token时做好了接收数据准备. 1) 当CPU写入值到该寄存器后, 对 IN token, MXPLD寄存器的值用于指示要发送的数据长度，且已做好发送准备。 对OUT token,设备已经做好了从主机接收数据的准备。MXPLD的值就表示从主机所接收数据的最大长度。

		(2) 当CPU读该寄存器时, 对IN token, MXPLD 的值表示要发送到主机的数据长度 对OUT token, MXPLD 的值表示从主机接收到的实际数据长度 **注:**一旦 MXPLD 的值被写入, 收到IN/OUT token后.数据包可以立即收发

USB配置寄存器(USB_CFGx)

寄存器	地址偏移	R/W	描述	复位值
USBD_CFG0	USBD_BA+0x508	R/W	端点0配置寄存器	0x0000_0000
USBD_CFG1	USBD_BA+0x518	R/W	端点1配置寄存器	0x0000_0000
USBD_CFG2	USBD_BA+0x528	R/W	端点2配置寄存器	0x0000_0000
USBD_CFG3	USBD_BA+0x538	R/W	端点3配置寄存器	0x0000_0000
USBD_CFG4	USBD_BA+0x548	R/W	端点4配置寄存器	0x0000_0000
USBD_CFG5	USBD_BA+0x558	R/W	端点5配置寄存器	0x0000_0000
USBD_CFG6	USBD_BA+0x568	R/W	端点6配置寄存器	0x0000_0000
USBD_CFG7	USBD_BA+0x578	R/W	端点7配置寄存器	0x0000_0000
USBD_CFG8	USBD_BA+0x588	R/W	端点8配置寄存器	0x0000_0000
USBD_CFG9	USBD_BA+0x598	R/W	端点9配置寄存器	0x0000_0000
USBD_CFG10	USBD_BA+0x5A8	R/W	端点10配置寄存器	0x0000_0000
USBD_CFG11	USBD_BA+0x5B8	R/W	端点11配置寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved						**CSTALL**	**Reserved**
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
DSQSYNC	**STATE**		**ISOCH**	**EPNUM**			

位	描述	
[31:10]	**Reserved**	保留
[9]	**CSTALL**	**清STALL 响应位** 0 =在setup阶段禁止设备清除STALL 1 =在setup阶段应许设备清除STALL
[8]	**Reserved**	保留
[7]	**DSQSYNC**	**数据时序同步位** 0 = DATA0 PID 1 = DATA1 PID

		注:该位用于指定在接下来的IN token 传输过程是DATA0还是DATA1 PID. 在IN token 过程，硬件会基于该位硬件自动触发选择
[6:5]	**STATE**	**端点状态** 00 =端点被禁止 01 =输出端点 10 =输入端点 11 =未定义
[4]	**ISOCH**	**同步端点设置** 该位用于设置该端点为同步端点。无握手信号 0 = 非同步端点. 1 = 同步端点
[3:0]	**EPNUM**	**端点号** 用于定义当前端点的端点号

USB扩展配置寄存器(USB_CFGPx)

寄存器	地址偏移	R/W	描述	复位值
USBD_CFGP0	USBD_BA+0x50C	R/W	端点0设置Stall和清In/Out准备控制寄存器	0x0000_0000
USBD_CFGP1	USBD_BA+0x51C	R/W	端点1设置Stall和清In/Out准备控制寄存器	0x0000_0000
USBD_CFGP2	USBD_BA+0x52C	R/W	端点2设置Stall和清In/Out准备控制寄存器	0x0000_0000
USBD_CFGP3	USBD_BA+0x53C	R/W	端点3设置Stall和清In/Out准备控制寄存器	0x0000_0000
USBD_CFGP4	USBD_BA+0x54C	R/W	端点4设置Stall和清In/Out准备控制寄存器	0x0000_0000
USBD_CFGP5	USBD_BA+0x55C	R/W	端点5设置Stall和清In/Out准备控制寄存器	0x0000_0000
USBD_CFGP6	USBD_BA+0x56C	R/W	端点6设置Stall和清In/Out准备控制寄存器	0x0000_0000
USBD_CFGP7	USBD_BA+0x57C	R/W	端点7设置Stall和清In/Out准备控制寄存器	0x0000_0000
USBD_CFGP8	USBD_BA+0x58C	R/W	端点8设置Stall和清In/Out准备控制寄存器	0x0000_0000
USBD_CFGP9	USBD_BA+0x59C	R/W	端点9设置Stall和清In/Out准备控制寄存器	0x0000_0000
USBD_CFGP10	USBD_BA+0x5AC	R/W	端点10设置Stall和清In/Out准备控制寄存器	0x0000_0000
USBD_CFGP11	USBD_BA+0x5BC	R/W	端点11设置Stall和清In/Out准备控制寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved						**SSTALL**	**CLRRDY**

位	描述	
[31:2]	**Reserved**	保留
[1]	**SSTALL**	**STALL设置位** 0 = 禁止设备响应STALL 1 = 设置设备自动响应STALL
[0]	**CLRRDY**	**清除准备位** 当USBD_ MXPLDx寄存器被设置后，表示该端点准备好可以发送或接收数据。如果用户想在传输开始前关闭传输，需要设置该位为1来进行关闭，该位会自动清零。 对IN token，写 ‘1’ 清除 IN token 时发送数据到 USB 的准备信号

		对OUT token，写 ‘1’ 清除 OUT token 时从 USB 接收数据的准备信号 该位只能写 1，读数据返回值总是 0。

6.32 HSUSBD 高速USB2.0设备接口

6.32.1 概述

USB 设备控制器与AHB总线和UTMI总线都有接口。USB控制器同时包含AHB主接口和AHB从接口。CPU通过AHB从接口编写USB控制器寄存器。在收发传输中，USB设备控制器需要通过AHB主接口来从内存读写数据。USB设备控制器符合USB2.0协议规范，包含12个可配置的端点以及一个控制端点。这些端点可以配置成批量模式、中断模式或等时模式。USB设备控制器内建DMA以降低CPU负担。

6.32.2 特性

- 符合 USB2.0协议规范
- 支持12个可配置的端点以及一个控制端点
- 在收发方向，每个端点均可配置成等时传输，批量传输或中断传输
- 输入端点有3种工作模式-自动确认模式，手动确认模式和自由模式
- 支持 DMA 操作
- 4092个字节的可配置 RAM 作为端点缓存
- 端点最大支持1024个字节大小的数据包

6.32.3 框图

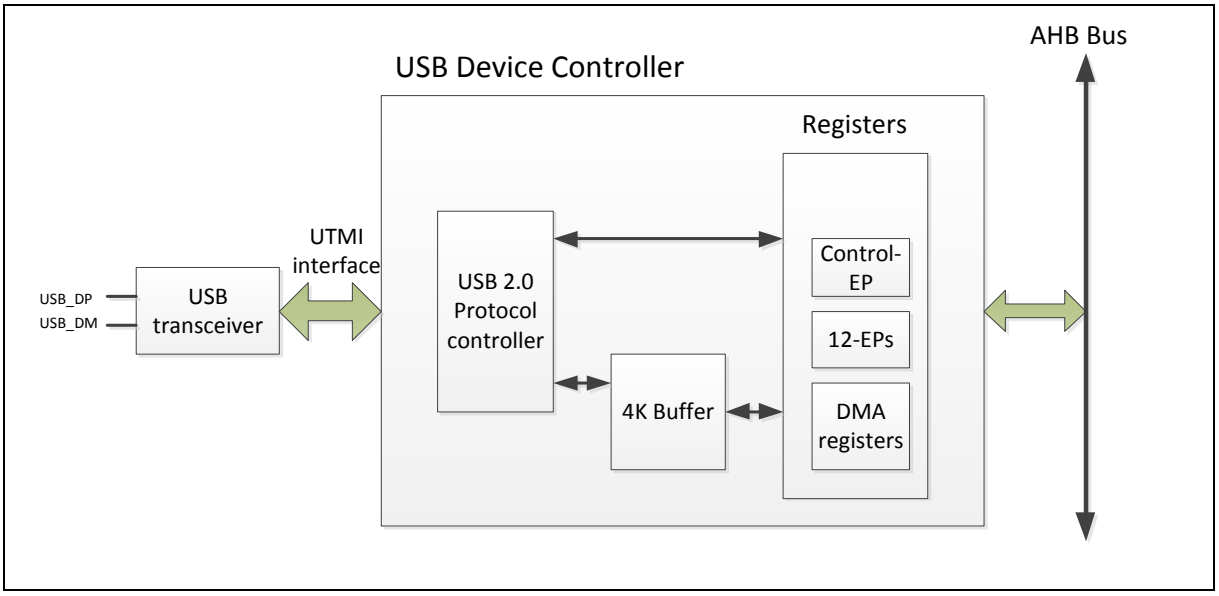


图 6.32-1 USB 设备控制器框图

6.32.4 基本配置

- 时钟源控制
  - 使能HSUSBD时钟HSUSBDCKEN(CLK_AHBCLK[10])。
- 复位控制
  - 复位HSUSBD控制器HSUSBDRST(SYS_IPRST0[10])。

6.32.5 功能描述

6.32.5.1 发送模式的不同操作模式

当接收到一个in-token，数据就可以写入缓存做发送准备。CPU对发送数据的确认有3种方式：

- 自动确认模式
- 手动确认模式
- 自由模式

6.32.5.2 自动确认模式

如果端点配置为自动确认模式，端点只会在数据个数等于寄存器EPMPS配置的个数时响应in-token。如果CPU需要在最后一笔传输中发送一个短包，需要对位SHORTTXEN（HSUSBD_EPxRSPCTL[6]）置1，缓存里剩下的数据就会在收到下一个in-token后发送给主机。

这种模式CPU的介入较少，主要由USB设备控制器完成。这种模式一般在发送给主机的数据量大于最大包时使用。

SHORTTXEN	缓存里的有效数据	数据发送/NAK发送
0	< 最大包大小	NAK 发送
0	>=最大包大小	发送最大包数据
1	< 最大包大小	发送所有的有效数据
1	>=最大包大小	发送最大包数据

6.32.5.3 手动确认模式

如果选择端点以手动验证模式操作，则端点仅在每次CPU验证缓冲区中的数据时才应答。CPU必须将数据写入缓冲区，然后将数据的计数写入EPxTXCNT寄存器。一旦通过向EPxTXCNT注册表中写入计数来验证数据，数据将在收到in-token时发送到主机。

此模式要求每次传输都有CPU干预。但是，如果每次发送的数据计数不是固定的，而是由CPU决定的，那么这将非常有用。

EPxTXCNT 写入	缓存里数据有效个数	数据发送/NAK发送
NO	-	NAK
YES	EPxTXCNT	发送EPxTXCNT长度的数据

6.32.5.4 自由模式

自由模式是最简单的操作模式，没有确认的过程。CPU向缓存写入数据后，如果接收到主机发送的in-token，缓存里的数据会自动确认然后发送给主机。如果缓存里的数据个数多于最大包时，控制器自动发送最大包的数据给主机。

这种模式需要CPU最少的参与，适合传输速率比包大小重要的等时传输模式的发送。

缓存里的有效数据	数据发送
< 最大包大小	发送数据有效
>= 最大包大小	

6.32.5.5 分散-收集功能

用户使能DMA分散收集功能通过设置SGEN(HSUSBD_DMACTL[6])=1,然后设置HSUSBD_DMACNT为8个字节，内存地址和长度会在描述符里重排序，格式如表表6.32-1。

	格式			
	[31]	[30]	[29:0]	
Word0	MEM_ADDR[31:0]			
Word1	EOT	RD	保留	计数[19:0]

表 6.32-1 分散收集描述符格式

MEM_ADDR: 传输结束。当此位为高，意味着是最后一个描述符。

EOT: 意味着从内存读取数据到缓存。“0”意味着从缓存读取数据到内存。

RD:“1”表示从内存读到缓存。“0”意味着从缓存读取数据到内存。

6.32.6 寄存器映射

R: 只读, **W:** 只写, **R/W:**读/写

寄存器	偏移地址	R/W	描述	复位值
USBD 基地址: **HSUSBD_BA = 0x4001_9000**				
HSUSBD_GINTSTS	HSUSBD_BA+0x000	R	总体中断状态寄存器	0x0000_0000
HSUSBD_GINTEN	HSUSBD_BA+0x008	R/W	总体中断使能寄存器	0x0000_0001
HSUSBD_BUSINTSTS	HSUSBD_BA+0x010	R/W	USB总线中断状态寄存器	0x0000_0000
HSUSBD_BUSINTEN	HSUSBD_BA+0x014	R/W	USB总线中断使能寄存器	0x0000_0040
HSUSBD_OPER	HSUSBD_BA+0x018	R/W	USB操作寄存器	0x0000_0002
HSUSBD_FRAMECNT	HSUSBD_BA+0x01C	R	USB帧计数寄存器	0x0000_0000
HSUSBD_FADDR	HSUSBD_BA+0x020	R/W	USB功能地址寄存器	0x0000_0000
HSUSBD_TEST	HSUSBD_BA+0x024	R/W	USB测试模式寄存器	0x0000_0000
HSUSBD_CEPDAT	HSUSBD_BA+0x028	R/W	控制端点数据缓存	0x0000_0000
HSUSBD_CEPCTL	HSUSBD_BA+0x02C	R/W	控制端点控制寄存器	0x0000_0000
HSUSBD_CEPINTEN	HSUSBD_BA+0x030	R/W	控制端点中断使能	0x0000_0000
HSUSBD_CEPINTSTS	HSUSBD_BA+0x034	R/W	控制端点中断状态	0x0000_1800
HSUSBD_CEPTXCNT	HSUSBD_BA+0x038	R/W	控制端点输入传输数据个数	0x0000_0000
HSUSBD_CEPRXCNT	HSUSBD_BA+0x03C	R	控制端点输出传输数据个数	0x0000_0000
HSUSBD_CEPDATCNT	HSUSBD_BA+0x040	R	控制端点数据个数	0x0000_0000
HSUSBD_SETUP1_0	HSUSBD_BA+0x044	R	Setup1&Setup0字节	0x0000_0000
HSUSBD_SETUP3_2	HSUSBD_BA+0x048	R	Setup3&Setup2字节	0x0000_0000
HSUSBD_SETUP5_4	HSUSBD_BA+0x04C	R	Setup5&Setup4字节	0x0000_0000
HSUSBD_SETUP7_6	HSUSBD_BA+0x050	R	Setup7&Setup6字节	0x0000_0000
HSUSBD_CEPBUFSTART	HSUSBD_BA+0x054	R/W	控制端点内存起始地址寄存器	0x0000_0000
HSUSBD_CEPBUFEND	HSUSBD_BA+0x058	R/W	控制端点内存结束地址寄存器	0x0000_0000
HSUSBD_DMACTL	HSUSBD_BA+0x05C	R/W	DMA控制状态寄存器	0x0000_0000
HSUSBD_DMACNT	HSUSBD_BA+0x060	R/W	DMA计数寄存器	0x0000_0000
HSUSBD_EPADAT	HSUSBD_BA+0x064	R/W	端点A数据寄存器	0x0000_0000
HSUSBD_EPAINTSTS	HSUSBD_BA+0x068	R/W	端点A中断状态寄存器	0x0000_0003
HSUSBD_EPAINTEN	HSUSBD_BA+0x06C	R/W	端点A中断使能寄存器	0x0000_0000
HSUSBD_EPADATCNT	HSUSBD_BA+0x070	R	端点A数据有效个数寄存器	0x0000_0000
HSUSBD_EPARSPCTL	HSUSBD_BA+0x074	R/W	端点A响应控制寄存器	0x0000_0000
HSUSBD_EPAMPS	HSUSBD_BA+0x078	R/W	端点A最大包大小寄存器	0x0000_0000

HSUSBD_EPATXCNT	HSUSBD_BA+0x07C	R/W	端点A发送字节个数寄存器	0x0000_0000
HSUSBD_EPACFG	HSUSBD_BA+0x080	R/W	端点A配置寄存器	0x0000_0012
HSUSBD_EPABUFSTART	HSUSBD_BA+0x084	R/W	端点A内存起始地址寄存器	0x0000_0000
HSUSBD_EPABUFEND	HSUSBD_BA+0x088	R/W	端点A内存结束地址寄存器	0x0000_0000
HSUSBD_EPBDAT	HSUSBD_BA+0x08C	R/W	端点B数据寄存器	0x0000_0000
HSUSBD_EPBINTSTS	HSUSBD_BA+0x090	R/W	端点B中断状态寄存器	0x0000_0003
HSUSBD_EPBINTEN	HSUSBD_BA+0x094	R/W	端点B中断使能寄存器	0x0000_0000
HSUSBD_EPBDATCNT	HSUSBD_BA+0x098	R	端点B数据有效个数寄存器	0x0000_0000
HSUSBD_EPBRSPCTL	HSUSBD_BA+0x09C	R/W	端点B响应控制寄存器	0x0000_0000
HSUSBD_EPBMPS	HSUSBD_BA+0x0A0	R/W	端点B最大包大小寄存器	0x0000_0000
HSUSBD_EPBTXCNT	HSUSBD_BA+0x0A4	R/W	端点B传输字节数寄存器	0x0000_0000
HSUSBD_EPBCFG	HSUSBD_BA+0x0A8	R/W	端点B配置寄存器	0x0000_0022
HSUSBD_EPBBUFSTART	HSUSBD_BA+0x0AC	R/W	端点B内存起始地址寄存器	0x0000_0000
HSUSBD_EPBBUFEND	HSUSBD_BA+0x0B0	R/W	端点B内存结束地址寄存器	0x0000_0000
HSUSBD_EPCDAT	HSUSBD_BA+0x0B4	R/W	端点C数据寄存器	0x0000_0000
HSUSBD_EPCINTSTS	HSUSBD_BA+0x0B8	R/W	端点C中断状态寄存器	0x0000_0003
HSUSBD_EPCINTEN	HSUSBD_BA+0x0BC	R/W	端点C中断使能寄存器	0x0000_0000
HSUSBD_EPCDATCNT	HSUSBD_BA+0x0C0	R	端点C数据有效字节数寄存器	0x0000_0000
HSUSBD_EPCRSPCTL	HSUSBD_BA+0x0C4	R/W	端点C响应控制寄存器	0x0000_0000
HSUSBD_EPCMPS	HSUSBD_BA+0x0C8	R/W	端点C最大包大小寄存器	0x0000_0000
HSUSBD_EPCTXCNT	HSUSBD_BA+0x0CC	R/W	端点C传输字节数寄存器	0x0000_0000
HSUSBD_EPCCFG	HSUSBD_BA+0x0D0	R/W	端点C配置寄存器	0x0000_0032
HSUSBD_EPCBUFSTART	HSUSBD_BA+0x0D4	R/W	端点C内存起始地址寄存器	0x0000_0000
HSUSBD_EPCBUFEND	HSUSBD_BA+0x0D8	R/W	端点C内存结束地址寄存器	0x0000_0000
HSUSBD_EPDDAT	HSUSBD_BA+0x0DC	R/W	端点D数据寄存器	0x0000_0000
HSUSBD_EPDINTSTS	HSUSBD_BA+0x0E0	R/W	端点D中断状态寄存器	0x0000_0003
HSUSBD_EPDINTEN	HSUSBD_BA+0x0E4	R/W	端点D中断使能寄存器	0x0000_0000
HSUSBD_EPDDATCNT	HSUSBD_BA+0x0E8	R	端点D数据有效字节数寄存器	0x0000_0000
HSUSBD_EPDRSPCTL	HSUSBD_BA+0x0EC	R/W	端点D响应控制寄存器	0x0000_0000
HSUSBD_EPDMPS	HSUSBD_BA+0x0F0	R/W	端点D最大包大小寄存器	0x0000_0000
HSUSBD_EPDTXCNT	HSUSBD_BA+0x0F4	R/W	端点D传输计数器寄存器	0x0000_0000
HSUSBD_EPDCFG	HSUSBD_BA+0x0F8	R/W	端点D配置寄存器	0x0000_0042
HSUSBD_EPDBUFSTART	HSUSBD_BA+0x0FC	R/W	端点D内存起始地址寄存器	0x0000_0000

HSUSBD_EPDBUFEND	HSUSBD_BA+0x100	R/W	端点D内存结束地址寄存器	0x0000_0000
HSUSBD_EPEDAT	HSUSBD_BA+0x104	R/W	端点D数据寄存器	0x0000_0000
HSUSBD_EPEINTSTS	HSUSBD_BA+0x108	R/W	端点E数据寄存器	0x0000_0003
HSUSBD_EPEINTEN	HSUSBD_BA+0x10C	R/W	端点E中断状态寄存器	0x0000_0000
HSUSBD_EPEDATCNT	HSUSBD_BA+0x110	R	端点E中断使能寄存器	0x0000_0000
HSUSBD_EPERSPCTL	HSUSBD_BA+0x114	R/W	端点E数据有效字节数寄存器	0x0000_0000
HSUSBD_EPEMPS	HSUSBD_BA+0x118	R/W	端点E响应控制寄存器	0x0000_0000
HSUSBD_EPETXCNT	HSUSBD_BA+0x11C	R/W	端点E最大包大小寄存器	0x0000_0000
HSUSBD_EPECFG	HSUSBD_BA+0x120	R/W	端点E传输计数器寄存器	0x0000_0052
HSUSBD_EPEBUFSTART	HSUSBD_BA+0x124	R/W	端点E配置寄存器	0x0000_0000
HSUSBD_EPEBUFEND	HSUSBD_BA+0x128	R/W	端点E内存起始地址寄存器	0x0000_0000
HSUSBD_EPFDAT	HSUSBD_BA+0x12C	R/W	端点F数据寄存器	0x0000_0000
HSUSBD_EPFINTSTS	HSUSBD_BA+0x130	R/W	端点F中断状态寄存器	0x0000_0003
HSUSBD_EPFINTEN	HSUSBD_BA+0x134	R/W	端点F中断使能寄存器	0x0000_0000
HSUSBD_EPFDATCNT	HSUSBD_BA+0x138	R	端点F数据有效字节数寄存器	0x0000_0000
HSUSBD_EPFRSPCTL	HSUSBD_BA+0x13C	R/W	端点F响应控制寄存器	0x0000_0000
HSUSBD_EPFMPS	HSUSBD_BA+0x140	R/W	端点F最大包大小寄存器	0x0000_0000
HSUSBD_EPFTXCNT	HSUSBD_BA+0x144	R/W	端点F传输计数器寄存器	0x0000_0000
HSUSBD_EPFCFG	HSUSBD_BA+0x148	R/W	端点F配置寄存器	0x0000_0062
HSUSBD_EPFBUFSTART	HSUSBD_BA+0x14C	R/W	端点F内存起始地址寄存器	0x0000_0000
HSUSBD_EPFBUFEND	HSUSBD_BA+0x150	R/W	端点F内存结束地址寄存器	0x0000_0000
HSUSBD_EPGDAT	HSUSBD_BA+0x154	R/W	端点G数据寄存器	0x0000_0000
HSUSBD_EPGINTSTS	HSUSBD_BA+0x158	R/W	端点G中断状态寄存器	0x0000_0003
HSUSBD_EPGINTEN	HSUSBD_BA+0x15C	R/W	端点G中断使能寄存器	0x0000_0000
HSUSBD_EPGDATCNT	HSUSBD_BA+0x160	R	端点G数据有效字节数寄存器	0x0000_0000
HSUSBD_EPGRSPCTL	HSUSBD_BA+0x164	R/W	端点G响应控制寄存器	0x0000_0000
HSUSBD_EPGMPS	HSUSBD_BA+0x168	R/W	端点G最大包大小寄存器	0x0000_0000
HSUSBD_EPGTXCNT	HSUSBD_BA+0x16C	R/W	端点G传输计数器寄存器	0x0000_0000
HSUSBD_EPGCFG	HSUSBD_BA+0x170	R/W	端点G配置寄存器	0x0000_0072
HSUSBD_EPGBUFSTART	HSUSBD_BA+0x174	R/W	端点G内存起始地址寄存器	0x0000_0000
HSUSBD_EPGBUFEND	HSUSBD_BA+0x178	R/W	端点G内存结束地址寄存器	0x0000_0000
HSUSBD_EPHDAT	HSUSBD_BA+0x17C	R/W	端点H数据寄存器	0x0000_0000
HSUSBD_EPHINTSTS	HSUSBD_BA+0x180	R/W	端点H中断状态寄存器	0x0000_0003

HSUSBD_EPHINTEN	HSUSBD_BA+0x184	R/W	端点H中断使能寄存器	0x0000_0000
HSUSBD_EPHDATCNT	HSUSBD_BA+0x188	R	端点H数据有效字节数寄存器	0x0000_0000
HSUSBD_EPHRSPCTL	HSUSBD_BA+0x18C	R/W	端点H响应控制寄存器	0x0000_0000
HSUSBD_EPHMPS	HSUSBD_BA+0x190	R/W	端点H最大包大小寄存器	0x0000_0000
HSUSBD_EPHTXCNT	HSUSBD_BA+0x194	R/W	端点H传输计数器寄存器	0x0000_0000
HSUSBD_EPHCFG	HSUSBD_BA+0x198	R/W	端点H配置寄存器	0x0000_0082
HSUSBD_EPHBUFSTART	HSUSBD_BA+0x19C	R/W	端点H内存起始地址寄存器	0x0000_0000
HSUSBD_EPHBUFEND	HSUSBD_BA+0x1A0	R/W	端点H内存结束地址寄存器	0x0000_0000
HSUSBD_EPIDAT	HSUSBD_BA+0x1A4	R/W	端点I数据寄存器	0x0000_0000
HSUSBD_EPIINTSTS	HSUSBD_BA+0x1A8	R/W	端点I中断状态寄存器	0x0000_0003
HSUSBD_EPIINTEN	HSUSBD_BA+0x1AC	R/W	端点I中断使能寄存器	0x0000_0000
HSUSBD_EPIDATCNT	HSUSBD_BA+0x1B0	R	端点I数据有效字节数寄存器	0x0000_0000
HSUSBD_EPIRSPCTL	HSUSBD_BA+0x1B4	R/W	端点I响应控制寄存器	0x0000_0000
HSUSBD_EPIMPS	HSUSBD_BA+0x1B8	R/W	端点I最大包大小寄存器	0x0000_0000
HSUSBD_EPITXCNT	HSUSBD_BA+0x1BC	R/W	端点I传输计数器寄存器	0x0000_0000
HSUSBD_EPICFG	HSUSBD_BA+0x1C0	R/W	端点I配置寄存器	0x0000_0092
HSUSBD_EPIBUFSTART	HSUSBD_BA+0x1C4	R/W	端点I内存起始地址寄存器	0x0000_0000
HSUSBD_EPIBUFEND	HSUSBD_BA+0x1C8	R/W	端点I内存结束地址寄存器	0x0000_0000
HSUSBD_EPJDAT	HSUSBD_BA+0x1CC	R/W	端点J数据寄存器	0x0000_0000
HSUSBD_EPJINTSTS	HSUSBD_BA+0x1D0	R/W	端点J中断状态寄存器	0x0000_0003
HSUSBD_EPJINTEN	HSUSBD_BA+0x1D4	R/W	端点J中断使能寄存器	0x0000_0000
HSUSBD_EPJDATCNT	HSUSBD_BA+0x1D8	R	端点J数据有效字节数寄存器	0x0000_0000
HSUSBD_EPJRSPCTL	HSUSBD_BA+0x1DC	R/W	端点J响应控制寄存器	0x0000_0000
HSUSBD_EPJMPS	HSUSBD_BA+0x1E0	R/W	端点J最大包大小寄存器	0x0000_0000
HSUSBD_EPJTXCNT	HSUSBD_BA+0x1E4	R/W	端点J传输计数器寄存器	0x0000_0000
HSUSBD_EPJCFG	HSUSBD_BA+0x1E8	R/W	端点J配置寄存器	0x0000_00A2
HSUSBD_EPJBUFSTART	HSUSBD_BA+0x1EC	R/W	端点J内存起始地址寄存器	0x0000_0000
HSUSBD_EPJBUFEND	HSUSBD_BA+0x1F0	R/W	端点J内存结束地址寄存器	0x0000_0000
HSUSBD_EPKDAT	HSUSBD_BA+0x1F4	R/W	端点K数据寄存器	0x0000_0000
HSUSBD_EPKINTSTS	HSUSBD_BA+0x1F8	R/W	端点K中断状态寄存器	0x0000_0003
HSUSBD_EPKINTEN	HSUSBD_BA+0x1FC	R/W	端点K中断使能寄存器	0x0000_0000
HSUSBD_EPKDATCNT	HSUSBD_BA+0x200	R	端点K数据有效字节数寄存器	0x0000_0000
HSUSBD_EPKRSPCTL	HSUSBD_BA+0x204	R/W	端点K响应控制寄存器	0x0000_0000

HSUSBD_EPKMPS	HSUSBD_BA+0x208	R/W	端点K最大包大小寄存器	0x0000_0000
HSUSBD_EPKTXCNT	HSUSBD_BA+0x20C	R/W	端点K传输计数器寄存器	0x0000_0000
HSUSBD_EPKCFG	HSUSBD_BA+0x210	R/W	端点K配置寄存器	0x0000_00B2
HSUSBD_EPKBUFSTART	HSUSBD_BA+0x214	R/W	端点K内存起始地址寄存器	0x0000_0000
HSUSBD_EPKBUFEND	HSUSBD_BA+0x218	R/W	端点K内存结束地址寄存器	0x0000_0000
HSUSBD_EPLDAT	HSUSBD_BA+0x21C	R/W	端点L数据寄存器	0x0000_0000
HSUSBD_EPLINTSTS	HSUSBD_BA+0x220	R/W	端点L中断状态寄存器	0x0000_0003
HSUSBD_EPLINTEN	HSUSBD_BA+0x224	R/W	端点L中断使能寄存器	0x0000_0000
HSUSBD_EPLDATCNT	HSUSBD_BA+0x228	R	端点L数据有效字节数寄存器	0x0000_0000
HSUSBD_EPLRSPCTL	HSUSBD_BA+0x22C	R/W	端点L响应控制寄存器	0x0000_0000
HSUSBD_EPLMPS	HSUSBD_BA+0x230	R/W	端点L最大包大小寄存器	0x0000_0000
HSUSBD_EPLTXCNT	HSUSBD_BA+0x234	R/W	端点L传输计数器寄存器	0x0000_0000
HSUSBD_EPLCFG	HSUSBD_BA+0x238	R/W	端点L配置寄存器	0x0000_00C2
HSUSBD_EPLBUFSTART	HSUSBD_BA+0x23C	R/W	端点L内存起始地址寄存器	0x0000_0000
HSUSBD_EPLBUFEND	HSUSBD_BA+0x240	R/W	端点L内存结束地址寄存器	0x0000_0000
HSUSBD_DMAADDR	HSUSBD_BA+0x700	R/W	AHBDMA地址寄存器	0x0000_0000
HSUSBD_PHYCTL	HSUSBD_BA+0x704	R/W	USBPHY控制寄存器	0x0000_0420

6.32.7 寄存器描述

总体中断状态寄存器 (HSUSBD_GINTSTS)

寄存器	偏移量	R/W	描述	复位值
HSUSBD_GINTSTS	HSUSBD_BA+0x000	R	总体中断状态寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved		EPLIF	EPKIF	EPJIF	EPIIF	EPHIF	EPGIF
7	6	5	4	3	2	1	0
EPFIF	EPEIF	EPDIF	EPCIF	EPBIF	EPAIF	CEPIF	USBIF

位	描述	
[31:14]	**Reserved**	保留
[13]	**EPLIF**	**端点 L中断** 此位置1后，需读端点 L 的中断状态寄存器以了解产生中断的原因 0 = 没发生中断 1 = 发生了中断
[12]	**EPKIF**	**端点 K中断** 当此位为1，需要去读相应的端点 K 的中断状态寄存器以了解产生中断的原因 0 = 没发生中断 1 = 发生了中断
[11]	**EPJIF**	**端点 J中断** 当此位为1，需要去读相应的端点 J 的中断状态寄存器以了解产生中断的原因 0 = 没发生中断 1 = 发生了中断
[10]	**EPIIF**	**端点 I中断** 当此位为1，需要去读相应的端点 I 的中断状态寄存器以了解产生中断的原因 0 = 没发生中断 1 = 发生了中断
[9]	**EPHIF**	**端点 H 中断** 当此位为1，需要去读相应的端点 H 的中断状态寄存器以了解产生中断的原因 0 = 没发生中断 1 = 发生了中断

[8]	**EPGIF**	**端点 G中断** 当此位为1，需要去读相应的端点 G 的中断状态寄存器以了解产生中断的原因 0 = 没发生中断 1 = 发生了中断
[7]	**EPFIF**	**端点 F中断** 当此位为1，需要去读相应的端点 F 的中断状态寄存器以了解产生中断的原因 0 = 没发生中断 1 = 发生了中断
[6]	**EPEIF**	**端点 E中断** 当此位为1，需要去读相应的端点 E 的中断状态寄存器以了解产生中断的原因 0 = 没发生中断 1 = 发生了中断
[5]	**EPDIF**	**端点 D中断** 当此位为1，需要去读相应的端点 D 的中断状态寄存器以了解产生中断的原因 0 = 没发生中断 1 = 发生了中断
[4]	**EPCIF**	**端点 C中断** 当此位为1，需要去读相应的端点 C 的中断状态寄存器以了解产生中断的原因 0 = 没发生中断 1 = 发生了中断
[3]	**EPBIF**	**端点 B中断** 当此位为1，需要去读相应的端点 B 的中断状态寄存器以了解产生中断的原因 0 = 没发生中断 1 = 发生了中断
[2]	**EPAIF**	**端点 A中断** 当此位为1，需要去读相应的端点 A 的中断状态寄存器以了解产生中断的原因 0 = 没发生中断 1 = 发生了中断
[1]	**CEPIF**	**控制端点中断** 当此位为1，需要去读相应的控制端点的中断状态寄存器以了解产生中断的原因 0 = 没发生中断 1 = 发生了中断
[0]	**USBIF**	**USB 中断** 此位是USB特殊事件端点的中断表达，当此位置1，需要去读USB中断状态寄存器来了解产生中断的原因 0 = 没发生中断 1 = 发生了中断

总体中断使能寄存器(HSUSBD_GINTEN)

寄存器	偏移量	R/W	描述	复位值
HSUSBD_GINTEN	HSUSBD_BA+0x008	R/W	总体中断使能寄存器	0x0000_0001

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved		EPLIEN	EPKIEN	EPJIEN	EPIIEN	EPHIEN	EPGIEN
7	6	5	4	3	2	1	0
EPFIEN	EPEIEN	EPDIEN	EPCIEN	EPBIEN	EPAIEN	CEPIEN	USBIEN

位	描述	
[31:14]	**Reserved**	保留
[13]	**EPLIEN**	**端点 L 中断使能位** 0 = 禁止端点L中断 1 = 使能端点L中断
[12]	**EPKIEN**	**端点 K 中断使能位** 0 = 禁止端点K中断 1 = 使能端点K中断
[11]	**EPJIEN**	**端点 J 中断使能位** 0 = 禁止端点J中断 1 = 使能端点J中断
[10]	**EPIIEN**	**端点 I 中断使能位** 0 = 禁止端点I中断 1 = 使能端点I中断
[9]	**EPHIEN**	**端点 H 中断使能位** 0 = 禁止端点H中断 1 = 使能端点H中断
[8]	**EPGIEN**	**端点 G 中断使能位** 0 = 禁止端点G中断 1 = 使能端点G中断
[7]	**EPFIEN**	**端点 F 中断使能位** 0 = 禁止端点F中断 1 = 使能端点F中断

[6]	**EPEIEN**	**端点 E 中断使能位** 0 = 禁止端点E中断 1 = 使能端点E中断
[5]	**EPDIEN**	**端点 D 中断使能位** 0 = 禁止端点D中断 1 = 使能端点D中断
[4]	**EPCIEN**	**端点 C 中断使能位** 0 = 禁止端点C中断 1 = 使能端点C中断
[3]	**EPBIEN**	**端点 B 中断使能位** 0 = 禁止端点B中断 1 = 使能端点B中断
[2]	**EPAIEN**	**端点 A 中断使能位** 0 = 禁止端点A中断 1 = 使能端点A中断
[1]	**CEPIEN**	**控制端点中断使能位** 0 = 禁止控制端点中断 1 = 使能控制端点中断
[0]	**USBIEN**	**USB 中断使能位** 0 = 禁止中断 1 = 使能中断

USB总线中断状态寄存器(HSUSBD_BUSINTSTS)

寄存器	偏移量	R/W	描述	复位值
HSUSBD_BUSINTSTS	HSUSBD_BA+0x010	R/W	USB总线中断状态寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							VBUSDETIF
7	6	5	4	3	2	1	0
Reserved	PHYCLKVLDIF	DMADONEIF	HISPDIF	SUSPENDIF	RESUMEIF	RSTIF	SOFIF

位	描述	
[31:9]	**Reserved**	保留
[8]	**VBUSDETIF**	**VBUS 检测中断状态** 0 = 没有 VBUS 插入 1 = VBUS已插入 **注:**此位写1清0。。
[7]	**Reserved**	保留
[6]	**PHYCLKVLDIF**	**可用时钟中断** 0 = 可用时钟未就绪 1 = 传输器提供的可用时钟有效 **注:**此位写1清0。。
[5]	**DMADONEIF**	**DMA 完成中断** 0 = 没有DMA传输完成 1 = DMA传输完成 **注:**此位写1清0。。
[4]	**HISPDIF**	**高速设置** 0 = 没有检测到有效的高速复位协议 1 = 检测到有效的高速复位协议，并且设备被设置为高速 **注:**此位写1清0。。
[3]	**SUSPENDIF**	**挂起请求** 此位默认是1，要在USB复位前写1清0，此位在检测到主机发送的 USB 挂起请求后置1 0 = 没有检测到主机发送的USB挂起请求 1=检测到主机发送的USB挂起请求 **注:**此位写1清0。。

[2]	**RESUMEIF**	**恢复状态** 0 =没发生总线恢复 1 =发生总线恢复 **注:**此位写1清0。。
[1]	**RSTIF**	**复位状态** 当此位置1，指示 USB 根端口的复位结束了 0 = USB 根端口复位没有结束 1 = USB根端口复位结束 **注:**此位写1清0。。
[0]	**SOFIF**	**SOF接受控制** 0 = 没有接收到帧开始包 1 = 接收到帧开始包 **注:**此位写1清0。。

USB总线中断使能寄存器(HSUSBD_BUSINTEN)

寄存器	偏移量	R/W	描述	复位值
HSUSBD_BUSINTEN	HSUSBD_BA+0x014	R/W	USB总线中断使能寄存器	0x0000_0040

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							VBUSDETIEN
7	6	5	4	3	2	1	0
Reserved	PHYCLKVLDIEN	DMADONEIEN	HISPDIEN	SUSPENDIEN	RESUMEIEN	RSTIEN	SOFIEN

位	描述	
[31:9]	**Reserved**	保留
[8]	**VBUSDETIEN**	**VBUS 检测中断使能位** 0 = 禁止VBUS浮空检测中断. 1 = 使能VBUS 浮空检测中断.
[7]	**Reserved**	保留
[6]	**PHYCLKVLDIEN**	**可用时钟中断使能位** 0 = 禁止可用时钟中断. 1 = 使能可用时钟中断.
[5]	**DMADONEIEN**	**DMA 完成中断 使能** 0 = 禁止 DMA 完成中断 1 = 使能 DMA 完成中断.
[4]	**HISPDIEN**	**高速设置中断使能** 0 = 禁止高速设置中断. 1 = 使能高速设置中断.
[3]	**SUSPENDIEN**	**挂起请求中断使能** 0 = 禁止挂起中断. 1 = 使能挂起中断
[2]	**RESUMEIEN**	**恢复中断使能** 0 = 禁止恢复中断 1 = 使能恢复中断.
[1]	**RSTIEN**	**复位状态** 0 = 禁止USB复位中断 1 = 使能USB复位中断.

[0]	**SOFIEN**	**SOF 中断使能** 0 = 禁止 SOF中断 1 = 使能 SOF中断

USB操作寄存器(HSUSBD_OPER)

寄存器	偏移量	R/W	描述	复位值
HSUSBD_OPER	HSUSBD_BA+0x018	R/W	USB操作寄存器	0x0000_0002

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved					CURSPD	HISPDEN	RESUMEEN

位	描述	
[31:3]	Reserved	保留
[2]	CURSPD	USB 目前速度 0 = 设备工作在全速模式. 1 = USB设备工作在高速模式
[1]	HISPDEN	USB 高速 0 = USB 设备在复位协议中不理会线性调频序列，意味着即使连上了USB2.0 的主机也只允许USB 设备控制器工作在全速模式，. 1 = USB 设备控制器在复位时，发起线性调频序列.
[0]	RESUMEEN	产生恢复序列 0 = 没有向主机发起恢复序列 1 = 当使能设备远程唤醒，向主机发起恢复序列。此位自动清0。

USB帧计数寄存器(HSUSBD_FRAMECNT)

寄存器	偏移量	R/W	描述	复位值
HSUSBD_FRAMECNT	HSUSBD_BA+0x01C	R	USB帧计数寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved		**FRAMECNT**					
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
FRAMECNT					**MFRAMECNT**		

位	描述	
[31:14]	**Reserved**	保留
[13:3]	**FRAMECNT**	**帧计数** 这里包含最近的帧开始包的计数值
[2:0]	**MFRAMECNT**	**微帧计数** 这里包含帧计数域里的微帧数量

USB功能地址寄存器(HSUSBD_FADDR)

寄存器	偏移量	R/W	描述	复位值
HSUSBD_FADDR	HSUSBD_BA+0x020	R/W	USB功能地址寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved	**FADDR**						

位	描述	
[31:7]	**Reserved**	保留
[6:0]	**FADDR**	**USB功能地址** 这里包含USB设备的配置地址，在检测到根端口复位后清0。

USB测试模式寄存器(HSUSBD_TEST)

寄存器	偏移量	R/W	描述	复位值
HSUSBD_TEST	HSUSBD_BA+0x024	R/W	USB测试模式寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved					**TESTMODE**		

位	描述	
[31:3]	**Reserved**	保留
[2:0]	**TESTMODE**	**测试模式选择** 000 = 正常操作 001 = Test_J 010 = Test_K 011 = Test_SE0_NAK 100 = Test_Packet 101 = Test_Force_Enable 110 = 保留 111 = 保留 **注：**在检测到根端口复位后清0。

控制端点数据缓存(HSUSBD_CEPDAT)

寄存器	偏移量	R/W	描述	复位值
HSUSBD_CEPDAT	HSUSBD_BA+0x028	R/W	控制端点数据缓存	0x0000_0000

31	30	29	28	27	26	25	24
DAT							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
DAT							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
DAT							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
DAT							

位	描述	
[31:0]	**DAT**	**控制端点数据缓存** 控制端点读写的数据缓存 **注：**只支持字或字节访问。

控制断点控制寄存器(HSUSBD_CEPCTL)

寄存器	偏移量	R/W	描述	复位值
HSUSBD_CEPCTL	HSUSBD_BA+0x02C	R/W	控制断点控制寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved				FLUSH	ZEROLEN	STALLEN	NAKCLR

位	描述	
[31:4]	Reserved	保留
[3]	FLUSH	**CEP清除位** 0 = 无效 1 = 清除包缓存及HSUSBD_CEPDATCNT寄存器，此位自动清0
[2]	ZEROLEN	**0长度包** 这一位只在自动确认模式下有效。 0 = 无效 1 = USB 设备收到IN token 后向主机发送0长度数据包。这一位在发送0长度的包之后自动清0，所以CPU不需要清除此位
[1]	STALLEN	**STALL使能位** stall 位置1通常用来响应无效的或不支持的请求。此位置1必须同时把. NAKCLR位清0，因为NAKCLR位的优先级比STALL位高。在接收到下一个setup-token后此位自动清除，所以，CPU不需要再清除此位。 0 = 控制端点收到令牌，不回复 STALL 1 = 控制端点收到任何令牌，都回复 STALL **注:** 只有当CPU对HSUSBD_CEPCTL [1:0] 写入 2'b10 或2'b00, 此位才会更新。
[0]	NAKCLR	**无响应控制** 0 = 此位由CPU写0清除，表示 CPU 已对 USB Device 配置好并已做好接收数据、响应 ACK 的准备 1 = USB 设备收到Setpup token后置1，USB 设备对接下来的主机请求响应 NAK 。本地CPU可以用这个时间完成基于这个请求的相关工作之后清除这一位 **注:** 只有当CPU对HSUSBD_CEPCTL [1:0] 写入 2'b10 或2'b00, 此位才会更新。

控制端点中断使能(HSUSBD_CEPINTEN)

寄存器	偏移量	R/W	描述	复位值
HSUSBD_CEPINTEN	HSUSBD_BA+0x030	R/W	控制端点中断使能	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved			BUFEMPTYIEN	BUFFULLIEN	STSDONEIEN	ERRIEN	STALLIEN
7	6	5	4	3	2	1	0
NAKIEN	RXPKIEN	TXPKIEN	PINGIEN	INTKIEN	OUTTKIEN	SETUPPKIEN	SETUPTKIEN

位	描述	
[31:13]	**Reserved**	保留
[12]	**BUFEMPTYIEN**	**缓存空中断** 0 = 禁止控制端点缓存空中断 1 = 使能控制端点缓存空中断
[11]	**BUFFULLIEN**	**缓存满中断** 0 = 禁止控制端点缓存满中断 1 = 使能控制端点缓存满中断
[10]	**STSDONEIEN**	**状态完成中断** 0 = 禁止控制端点状态完成中断 1 = 使能控制端点状态完成中断
[9]	**ERRIEN**	**USB 错误中断** 0 = 禁止控制端点USB 错误中断 1 = 使能控制端点USB 错误中断
[8]	**STALLIEN**	**STALL发送中断** 0 = 禁止控制端点STALL发送中断 1 = 使能控制端点STALL 发送中断
[7]	**NAKIEN**	**NAK 发送中断** 0 = 禁止控制端点NAK发送中断 1 = 使能控制端点NAK 发送中断
[6]	**RXPKIEN**	**数据包接收中断** 0 = 禁止控制端点数据接收中断 1 = 使能控制端点数据接收中断

[5]	**TXPKIEN**	**数据包发送中断** 0 = 禁止控制端点数据包发送中断 1 = 使能控制端点数据包发送中断
[4]	**PINGIEN**	**Ping Token 中断** 0 = 禁止控制端点ping token中断 1 = 使能控制端点ping token中断
[3]	**INTKIEN**	**In Token 中断** 0 = 禁止控制端点IN token中断 1 = 使能控制端点IN token中断
[2]	**OUTTKIEN**	**Out Token 中断** 0 = 禁止控制端点OUT toke中断 1 = 使能控制端点OUT toke中断
[1]	**SETUPPKIEN**	**Setup包中断** 0 = 禁止控制端点SETUP包中断 1 = 使能控制端点SETUP包中断
[0]	**SETUPTKIEN**	**Setup Token 中断使能位** 0 = 禁止控制端点SETUP token中断 1 = 使能控制端点SETUP token中断

控制端点中断状态(HSUSBD_CEPINTSTS)

寄存器	偏移量	R/W	描述	复位值
HSUSBD_CEPINTSTS	HSUSBD_BA+0x034	R/W	控制端点中断状态	0x0000_1800

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved			BUFEMPTYIF	BUFFULLIF	STSDONEIF	ERRIF	STALLIF
7	6	5	4	3	2	1	0
NAKIF	RXPKIF	TXPKIF	PINGIF	INTKIF	OUTTKIF	SETUPPKIF	SETUPTKIF

位	描述	
[31:13]	Reserved	保留
[12]	BUFEMPTYIF	**缓存空中断** 0 = 控制端点缓存非空 1 = 控制端点缓存空 **注：**此位写1清0。
[11]	BUFFULLIF	**缓存满中断** 0 = 控制端点缓存未满. 1 = 控制端点缓存满. **注：**此位写1清0。
[10]	STSDONEIF	**状态完成中断** 0 = 没有USB事务成功结束 1 = USB 事务的状态阶段成功完成. **注：**此位写1清0。
[9]	ERRIF	**USB 错误中断** 0 = 事务中没有错误发生. 1 = 事务中发生了一个错误. **注：**此位写1清0。
[8]	STALLIF	**STALL发送中断** 0 = 没有发送stall-token 响应收发令牌 1 = 发送stall-token 响应收发令牌. **注：**此位写1清0。

[7]	**NAKIF**	**NAK 发送中断** 0 = 没有发送 NAK-token响应令牌 1 = 发送 NAK-token响应令牌 **注：**此位写1清0。
[6]	**RXPKIF**	**数据包接受中断** 0 = 没有发生这样的状况：在OUT-token 时从主机成功接收数据并发送 ACK给主机 1 = 在OUT-token 时从主机成功接收数据并发送 ACK给主机 **注：**此位写1清0。
[5]	**TXPKIF**	**数据包发送中断** 0 = 没有发生这样的状况：在IN-token 时向主机成功发送数据并接收到ACK-token 1 = 在IN-token 时向主机成功发送数据并接收到ACK-token. **注：**此位写1清0。
[4]	**PINGIF**	**Ping Token中断** 0 = 控制端点没有从主机接收到ping tokent. 1 = 控制端点从主机接收到ping tokent. **注：**此位写1清0。
[3]	**INTKIF**	**in Token中断** 0 = 控制端点没有从主机接收到 IN token. 1 = 控制端点从主机接收到 IN token. **注：**此位写1清0。
[2]	**OUTTKIF**	**Out Token中断** 0 = 控制端点没有从主机接收到 OUT token 1 = 控制端点从主机接收到 OUT token **注：**此位写1清0。
[1]	**SETUPPKIF**	**Setup Packet中断** 这一位必须在收到下一个setup packet 前清除（写1）。如果这一位没有清除，那么连续的 setup packets 会覆盖setup packet 缓存 0 = 没有从主机接收到 Setup packet 1 = 从主机接收到 Setup packet. **注：**此位写1清0。
[0]	**SETUPTKIF**	**Setup Token中断** 0 = 没有接收到Setup token. 1 = 接收到Setup token **注：**此位写1清0。

控制端点输入传输数据个数(HSUSBD_CEPTXCNT)

寄存器	偏移量	R/W	描述	复位值
HSUSBD_CEPTXCNT	HSUSBD_BA+0x038	R/W	控制端点输入传输数据个数	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
TXCNT							

位	描述	
[31:8]	**Reserved**	保留
[7:0]	**TXCNT**	**IN 数据计数** 控制端点没有操作模式选择（但是类似手动模式）。本地CPU接受到in-token 向控制传输缓存写入数据，并向这个寄存器写入发送字节数。如果这里是0，会向主机发送一个0长度的包。如果写入的个数大于最大包的长度，只会发送最大包长度的数据

控制端点输出传输数据个数(HSUSBD_CEPRXCNT)

寄存器	偏移量	R/W	描述	复位值
HSUSBD_CEPRXCNT	HSUSBD_BA+0x03C	R	控制端点输出传输数据个数	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
RXCNT							

位	描述	
[31:8]	**Reserved**	保留
[7:0]	**RXCNT**	**OUT 数据个数** 控制传输，在OUT传输中收到的数据个数

控制端点数据个数(HSUSBD_CEPDATCNT)

寄存器	偏移量	R/W	描述	复位值
HSUSBD_CEPDATCNT	HSUSBD_BA+0x040	R	控制端点数据个数	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
DATCNT							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
DATCNT							

位	描述	
[31:16]	**Reserved**	保留
[15:0]	**DATCNT**	**控制端点数据个数** OUT时，控制端点缓存中收到的数据个数；IN 时，控制端点尚未发出去的数据个数。

Setup1 & Setup0字节(HSUSBD_SETUP1_0)

寄存器	偏移量	R/W	描述	复位值
HSUSBD_SETUP1_0	HSUSBD_BA+0x044	R	Setup1 & Setup0字节	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
SETUP1							
7	6	5	4	3	2	1	0
SETUP0							

位	描述	
[31:16]	Reserved	保留
[15:8]	SETUP1	**Setup 字节 1[15:8]** 这个字节的内容是最近接收到的setup packet的字节1，对于标准的设备请求，下面是bRequest的对应信息. 00000000 = 读取状态 00000001 =清除特征字 00000010 = 保留 00000011 = 设置特征字 00000100 = 保留 00000101 = 设置地址 00000110 = 读取描述符 00000111 = 设置描述符 00001000 = 读取陪住 00001001 = 设置配置 00001010 = 读取接口 00001011 = 设置接口 00001100 = 同步帧

[7:0]	**SETUP0**	**Setup 字节 0[7:0]** 这个寄存器是最近接收到的setup packet的字节0。对于标准的设备请求，下面是bmRequestType的对应信息. Bit 7(方向): 0: 主机到设备 1: 设备到主机 Bit 6-5 (类型): 00: 标准 01: 类 10: 厂家 11: 保留 Bit 4-0 (接收者) 00000**:** 设备 00001: 接口 00010: 端点 00011: 其他 Others: 保留

Setup3 & Setup2字节(HSUSBD_SETUP3_2)

寄存器	偏移量	R/W	描述	复位值
HSUSBD_SETUP3_2	HSUSBD_BA+0x048	R	Setup3 & Setup2字节	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
SETUP3							
7	6	5	4	3	2	1	0
SETUP2							

位	描述	
[31:16]	Reserved	保留
[15:8]	SETUP3	**Setup 字节 3 [15:8]** 这个寄存器是最近接收到的 setup packet 的字节3. 对于标准的设备请求，这里是wValue域的高字节。
[7:0]	SETUP2	**Setup 字节 2 [7:0]** 这个寄存器是最近接收到的 setup packet 的字节2. 对于标准的设备请求，这里是wValue域的低字节。

Setup5 & Setup4字节(HSUSBD_SETUP5_4)

寄存器	偏移量	R/W	描述	复位值
HSUSBD_SETUP5_4	HSUSBD_BA+0x04C	R	Setup5 & Setup4字节	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
SETUP5							
7	6	5	4	3	2	1	0
SETUP4							

位	描述	
[31:16]	Reserved	保留
[15:8]	SETUP5	**Setup 字节 5[15:8]** 这个寄存器是最近接收到的 setup packet 的字节5。对于标准的设备请求，这里是wIndex域的高字节。
[7:0]	SETUP4	**Setup 字节 4[7:0]** 这个寄存器是最近接收到的 setup packet 的字节4。对于标准的设备请求，这里是wIndex域的低字节。

Setup7 & Setup6字节(HSUSBD_SETUP7_6)

寄存器	偏移量	R/W	描述	复位值
HSUSBD_SETUP7_6	HSUSBD_BA+0x050	R	Setup7 & Setup6字节	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
SETUP7							
7	6	5	4	3	2	1	0
SETUP6							

位	描述	
[31:16]	Reserved	保留
[15:8]	SETUP7	**Setup 字节 7[15:8]** 这个寄存器是最近接收到的 setup packet 的字节7. 对于标准的设备请求，这里是wLength域的高字节。
[7:0]	SETUP6	**Setup 字节 6[7:0]** 这个寄存器是最近接收到的 setup packet 的字节6. 对于标准的设备请求，这里是wLength域的低字节。

控制端点内存起始地址(HSUSBD_CEPBUFSTART)

寄存器	偏移量	R/W	描述	复位值
HSUSBD_CEPBUFSTART	HSUSBD_BA+0x054	R/W	控制端点内存起始地址	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved				SADDR			
7	6	5	4	3	2	1	0
SADDR							

位	描述	
[31:12]	Reserved	保留
[11:0]	SADDR	**控制端点起始地址** 这里是控制端点内存的起始地址。

控制端点内存结束地址寄存器(HSUSBD_CEPBUFEND)

寄存器	偏移量	R/W	描述	复位值
HSUSBD_CEPBUFEND	HSUSBD_BA+0x058	R/W	控制端点内存结束地址寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved				EADDR			
7	6	5	4	3	2	1	0
EADDR							

位	描述	
[31:12]	**Reserved**	保留
[11:0]	**EADDR**	**控制端点结束地址** 这里是控制端点内存的结束地址。

DMA控制状态寄存器(HSUSBD_DMACTL)

寄存器	偏移量	R/W	描述	复位值
HSUSBD_DMACTL	HSUSBD_BA+0x05C	R/W	DMA控制状态寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							SVINEP
7	6	5	4	3	2	1	0
DMARST	SGEN	DMAEN	DMARD	EPNUM			

位	描述	
[31:9]	Reserved	保留
[8]	SVINEP	**端点方向** 这里决定 DMA 服务于输入端点或输出端点 0: DMA服务输出端点 1: DMA 服务输入端点
[7]	DMARST	**复位 DMA 状态机** 0 : 无效 1 : 复位DMA状态机
[6]	SGEN	**分散-收集 功能使能位** 0 : 禁止分散-收集功能 1 : 使能分散-收集功能
[5]	DMAEN	**DMA 使能位** 0 : 禁止DMA功能. 1 : 使能DMA功能
[4]	DMARD	**DMA 操作** 0 : DMA 写操作 (从 USB 缓存读取). DMA 会在执行写操作之前根据EPNM的设定确认端点数据的有效个数HSUSBD_EPxDATCNT 1 : DMA 读操作 (写入 USB缓存)
[3:0]	EPNUM	**DMA端点地址位** 设定端点地址

DMA计数寄存器(HSUSBD_DMACNT)

寄存器	偏移量	R/W	描述	复位值
HSUSBD_DMACNT	HSUSBD_BA+0x060	R/W	DMA计数寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved				**DMACNT**			
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
DMACNT							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
DMACNT							

位	描述	
[31:20]	**Reserved**	保留
[19:0]	**DMACNT**	**DMA 传输计数** 在这里写入需要DMA 传输的数据个数.

端点A~L数据寄存器(HSUSBD_EPADAT~ HSUSBD_EPLDAT)

寄存器	偏移量	R/W	描述	复位值
HSUSBD_EPADAT	HSUSBD_BA+0x064	R/W	端点A数据寄存器	0x0000_0000
HSUSBD_EPBDAT	HSUSBD_BA+0x08C	R/W	端点B数据寄存器	0x0000_0000
HSUSBD_EPCDAT	HSUSBD_BA+0x0B4	R/W	端点C数据寄存器	0x0000_0000
HSUSBD_EPDDAT	HSUSBD_BA+0x0DC	R/W	端点D数据寄存器	0x0000_0000
HSUSBD_EPEDAT	HSUSBD_BA+0x104	R/W	端点E数据寄存器	0x0000_0000
HSUSBD_EPFDAT	HSUSBD_BA+0x12C	R/W	端点F数据寄存器	0x0000_0000
HSUSBD_EPGDAT	HSUSBD_BA+0x154	R/W	端点G数据寄存器	0x0000_0000
HSUSBD_EPHDAT	HSUSBD_BA+0x17C	R/W	端点H数据寄存器	0x0000_0000
HSUSBD_EPIDAT	HSUSBD_BA+0x1A4	R/W	端点I数据寄存器	0x0000_0000
HSUSBD_EPJDAT	HSUSBD_BA+0x1CC	R/W	端点J数据寄存器	0x0000_0000
HSUSBD_EPKDAT	HSUSBD_BA+0x1F4	R/W	端点K数据寄存器	0x0000_0000
HSUSBD_EPLDAT	HSUSBD_BA+0x21C	R/W	端点L数据寄存器	0x0000_0000

31	30	29	28	27	26	25	24
EPDAT							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
EPDAT							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
EPDAT							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
EPDAT							

位	描述	
[31:0]	**EPDAT**	**端点A~L数据寄存器** 端点A~L 读写数据缓存 **注：**只支持字节或字访问

端点A~L中断状态寄存器(HSUSBD_EPAINTSTS~ HSUSBD_EPLINTSTS)

寄存器	偏移量	R/W	描述	复位值
HSUSBD_EPAINTSTS	HSUSBD_BA+0x068	R/W	端点A中断状态寄存器	0x0000_0003
HSUSBD_EPBINTSTS	HSUSBD_BA+0x090	R/W	端点B中断状态寄存器	0x0000_0003
HSUSBD_EPCINTSTS	HSUSBD_BA+0x0B8	R/W	端点C中断状态寄存器	0x0000_0003
HSUSBD_EPDINTSTS	HSUSBD_BA+0x0E0	R/W	端点D中断状态寄存器	0x0000_0003
HSUSBD_EPEINTSTS	HSUSBD_BA+0x108	R/W	端点E中断状态寄存器	0x0000_0003
HSUSBD_EPFINTSTS	HSUSBD_BA+0x130	R/W	端点F中断状态寄存器	0x0000_0003
HSUSBD_EPGINTSTS	HSUSBD_BA+0x158	R/W	端点G中断状态寄存器	0x0000_0003
HSUSBD_EPHINTSTS	HSUSBD_BA+0x180	R/W	端点H中断状态寄存器	0x0000_0003
HSUSBD_EPIINTSTS	HSUSBD_BA+0x1A8	R/W	端点I中断状态寄存器	0x0000_0003
HSUSBD_EPJINTSTS	HSUSBD_BA+0x1D0	R/W	端点J中断状态寄存器	0x0000_0003
HSUSBD_EPKINTSTS	HSUSBD_BA+0x1F8	R/W	端点K中断状态寄存器	0x0000_0003
HSUSBD_EPLINTSTS	HSUSBD_BA+0x220	R/W	端点L中断状态寄存器	0x0000_0003

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved			SHORTRXIF	ERRIF	NYETIF	STALLIF	NAKIF
7	6	5	4	3	2	1	0
PINGIF	INTKIF	OUTTKIF	RXPKIF	TXPKIF	SHORTTXIF	BUFEMPTYIF	BUFFULLIF

位	描述	
[31:13]	Reserved	保留

[12]	**SHORTRXIF**	**接收到 Bulk Out 短包** 0 = 没有接收到 bulk out 短包. 1 = 接收到 bulk out 短包 (包含0长度的包). **注:**此位写1清0。
[11]	**ERRIF**	**错误发送** 0 = 传输中没有错误 1 = 传输中发生错误 **注:**此位写1清0。
[10]	**NYETIF**	**NYET 发送** 0 = 内存中有足够的空间存放下一个数据包 1 = 内存中没有足够的空间存放下一个数据包 **注:**此位写1清0。
[9]	**STALLIF**	**USB STALL 发送** 0 = 上一个数据包已被接受或提供，没有响应STALL 1 = 上一个数据包没有被接受或提供，响应了STALL **注:**此位写1清0。
[8]	**NAKIF**	**USB NAK 发送** 0 = 上一个IN-token 提供了数据（未发NAK） 1 = 上一个IN-token未提供数据，响应了NAK **注:**此位写1清0。
[7]	**PINGIF**	**PING Token 中断** 0 = 从机没有接收到数据PING token 1 = 从机接收到一个数据PING token **注:**此位写1清0。
[6]	**INTKIF**	**数据 IN Token 中断** 0 =从机没收到IN token 1 =从机收到一个IN token **注:**此位写1清0。
[5]	**OUTTKIF**	**数据 OUT Token 中断** 0 = 从机没收到 OUT token 1 = 从机收到 OUT token。这一位在高速模式下也可以被 PING token 置位 **注:**此位写1清0。
[4]	**RXPKIF**	**数据包接收中断** 0 = 端点没有收到数据包 1 = 端点收到一个数据包 **注:**此位写1清0。
[3]	**TXPKIF**	**数据包发送中断** 0 = 端点没有向主机发送数据包 1 = 端点向主机发送一个数据包. **注:**此位写1清0。

[2]	**SHORTTXIF**	**短包发送中断** 0 = 上一个数据包不小于最大包 (EPMPS) 1 = 上一个数据包小于最大包 (EPMPS) **注:**此位写1清0。
[1]	**BUFEMPTYIF**	**缓存空** 对于输入端点 0 = 端点缓存非空 1 = 端点缓存空 对于输出端点 0 = 缓存里可读数据不是0 1 = 缓存里可读数据是0，或本地没有可用缓存（没有可读数据） **注:**此位只读。
[0]	**BUFFULLIF**	**缓存满** 对于输入缓存，目前选择的缓存满了，或者本地没有空间可写，对于输出缓存，FIFO已满，整个数据包都可以读取 0 = 端点包缓存未满 1 = 端点包缓存已满 **注:**此位只读。

端点A~L中断使能寄存器(HSUSBD_EPAINTEN~ HSUSBD_EPLINTEN)

寄存器	偏移量	R/W	描述	复位值
HSUSBD_EPAINTEN	HSUSBD_BA+0x06C	R/W	端点A中断使能寄存器	0x0000_0000
HSUSBD_EPBINTEN	HSUSBD_BA+0x094	R/W	端点B中断使能寄存器	0x0000_0000
HSUSBD_EPCINTEN	HSUSBD_BA+0x0BC	R/W	端点C中断使能寄存器	0x0000_0000
HSUSBD_EPDINTEN	HSUSBD_BA+0x0E4	R/W	端点D中断使能寄存器	0x0000_0000
HSUSBD_EPEINTEN	HSUSBD_BA+0x10C	R/W	端点E中断使能寄存器	0x0000_0000
HSUSBD_EPFINTEN	HSUSBD_BA+0x134	R/W	端点F中断使能寄存器	0x0000_0000
HSUSBD_EPGINTEN	HSUSBD_BA+0x15C	R/W	端点G中断使能寄存器	0x0000_0000
HSUSBD_EPHINTEN	HSUSBD_BA+0x184	R/W	端点H中断使能寄存器	0x0000_0000
HSUSBD_EPIINTEN	HSUSBD_BA+0x1AC	R/W	端点I中断使能寄存器	0x0000_0000
HSUSBD_EPJINTEN	HSUSBD_BA+0x1D4	R/W	端点J中断使能寄存器	0x0000_0000
HSUSBD_EPKINTEN	HSUSBD_BA+0x1FC	R/W	端点K中断使能寄存器	0x0000_0000
HSUSBD_EPLINTEN	HSUSBD_BA+0x224	R/W	端点L中断使能寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved			SHORTRXIEN	ERRIEN	NYETIEN	STALLIEN	NAKIEN
7	6	5	4	3	2	1	0
PINGIEN	INTKIEN	OUTTKIEN	RXPKIEN	TXPKIEN	SHORTTXIEN	BUFEMPTYIEN	BUFFULLIEN

位	描述	
[31:13]	**Reserved**	保留
[12]	**SHORTRXIEN**	**Bulk Out 短包中断使能位** 0 = 禁止 Bulk out 短包中断 1 = 使能 Bulk out 短包中断.

[11]	**ERRIEN**	**错误中断使能位** 0 = 禁止错误事件中断. 1 = 使能错误事件中断.
[10]	**NYETIEN**	**NYET 中断使能位** 0 = 禁止 NYET 情况的中断. 1 = 使能 NYET 情况的中断.
[9]	**STALLIEN**	**STALL 发送中断使能位** 当此位置1，端点向主机发送 stall 时产生中断. 0 = 禁止 STALL 中断 1 = 使能 STALL 中断
[8]	**NAKIEN**	**NAK 发送中断使能位** 当此位置1，端点向主机发送 NAK 时产生中断. 0 = 禁止 NAK 中断 1 = 使能 NAK 中断
[7]	**PINGIEN**	**PING Token中断使能位** 当此位置1，端点收到 PING token时产生中断. 0 = 禁止PING token中断 1 = 使能PING token中断.
[6]	**INTKIEN**	**IN Token 中断使能位** 当此位置1，端点收到 IN token 时产生中断. 0 = 禁止IN token中断 1 = 使能IN token中断.
[5]	**OUTTKIEN**	**OUT Token 中断使能位** 当此位置1，端点收到 OUT token时产生中断. 0 = 禁止 OUT token中断 1 = 使能 OUT token中断
[4]	**RXPKIEN**	**数据包接收中断使能位** 当此位置1，端点收到一个数据包时产生中断. 0 = 禁止数据包接收中断 1 = 使能数据包接收中断.
[3]	**TXPKIEN**	**数据包发送中断使能位** 当此位置1，端点向主机发送一个数据包时产生中断. 0 = 禁止向主机发送数据包的中断 1 = 使能向主机发送数据包的中断.
[2]	**SHORTTXIEN**	**短包发送中断使能位** 当此位置1，当这个端点向主机发送或接收到短包时产生中断 0 = 禁止短包中断. 1 = 使能短包中断.
[1]	**BUFEMPTYIEN**	**缓存空中断** 0 = 禁止缓存空中断. 1 = 使能缓存空中断.

[0]	**BUFFULLIEN**	**缓存满中断** 0 = 禁止缓存满中断 1 = 使能缓存满中断.

端点A~L数据有效个数寄存器(HSUSBD_EPADATCNT~ HSUSBD_EPLDATCNT)

寄存器	偏移量	R/W	描述	复位值
HSUSBD_EPADATCNT	HSUSBD_BA+0x070	R	端点A数据有效个数寄存器	0x0000_0000
HSUSBD_EPBDATCNT	HSUSBD_BA+0x098	R	端点B数据有效个数寄存器	0x0000_0000
HSUSBD_EPCDATCNT	HSUSBD_BA+0x0C0	R	端点C数据有效个数寄存器	0x0000_0000
HSUSBD_EPDDATCNT	HSUSBD_BA+0x0E8	R	端点D数据有效个数寄存器	0x0000_0000
HSUSBD_EPEDATCNT	HSUSBD_BA+0x110	R	端点E数据有效个数寄存器	0x0000_0000
HSUSBD_EPFDATCNT	HSUSBD_BA+0x138	R	端点F数据有效个数寄存器	0x0000_0000
HSUSBD_EPGDATCNT	HSUSBD_BA+0x160	R	端点G数据有效个数寄存器	0x0000_0000
HSUSBD_EPHDATCNT	HSUSBD_BA+0x188	R	端点H数据有效个数寄存器	0x0000_0000
HSUSBD_EPIDATCNT	HSUSBD_BA+0x1B0	R	端点I数据有效个数寄存器	0x0000_0000
HSUSBD_EPJDATCNT	HSUSBD_BA+0x1D8	R	端点J数据有效个数寄存器	0x0000_0000
HSUSBD_EPKDATCNT	HSUSBD_BA+0x200	R	端点K数据有效个数寄存器	0x0000_0000
HSUSBD_EPLDATCNT	HSUSBD_BA+0x228	R	端点L数据有效个数寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved	DMALOOP						
23	22	21	20	19	18	17	16
DMALOOP							
15	14	13	12	11	10	9	8
DATCNT							
7	6	5	4	3	2	1	0
DATCNT							

位	描述	
[31]	**Reserved**	保留
[30:16]	**DMALOOP**	**DMA 循环** 这个字节是未完成 DMA 循环. 每个循环传输32个字节

[15:0]	**DATCNT**	**数据个数** 对于输入端点(EPDIR(HSUSBD_EPxCFG[3] 为高.), 这里是输入端点缓存里的数据有效个数 对于输出端点(EPDIR(HSUSBD_EPxCFG[3]为低.), 这里是主机传送来的数据有效个数.

端点A~L响应控制寄存器(HSUSBD_EPARSPCTL~ HSUSBD_EPLRSPCTL)

寄存器	偏移量	R/W	描述	复位值
HSUSBD_EPARSPCTL	HSUSBD_BA+0x074	R/W	端点A响应控制寄存器	0x0000_0000
HSUSBD_EPBRSPCTL	HSUSBD_BA+0x09C	R/W	端点B响应控制寄存器	0x0000_0000
HSUSBD_EPCRSPCTL	HSUSBD_BA+0x0C4	R/W	端点C响应控制寄存器	0x0000_0000
HSUSBD_EPDRSPCTL	HSUSBD_BA+0x0EC	R/W	端点D响应控制寄存器	0x0000_0000
HSUSBD_EPERSPCTL	HSUSBD_BA+0x114	R/W	端点E响应控制寄存器	0x0000_0000
HSUSBD_EPFRSPCTL	HSUSBD_BA+0x13C	R/W	端点F响应控制寄存器	0x0000_0000
HSUSBD_EPGRSPCTL	HSUSBD_BA+0x164	R/W	端点G响应控制寄存器	0x0000_0000
HSUSBD_EPHRSPCTL	HSUSBD_BA+0x18C	R/W	端点H响应控制寄存器	0x0000_0000
HSUSBD_EPIRSPCTL	HSUSBD_BA+0x1B4	R/W	端点I响应控制寄存器	0x0000_0000
HSUSBD_EPJRSPCTL	HSUSBD_BA+0x1DC	R/W	端点J响应控制寄存器	0x0000_0000
HSUSBD_EPKRSPCTL	HSUSBD_BA+0x204	R/W	端点K响应控制寄存器	0x0000_0000
HSUSBD_EPLRSPCTL	HSUSBD_BA+0x22C	R/W	端点L响应控制寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
DISBUF	SHORTTXEN	ZEROLEN	HALT	TOGGLE	MODE		FLUSH

位	描述	
[31:8]	Reserved	保留

[7]	**DISBUF**	**缓存禁止位** 这一位用来接收不知道大小的OUT短包，接收包的大小依据HSUSBD_EPxDATCNT寄存器. 0 = 收到 Bulk-OUT 短包时收下数据. 1 = 收到 Bulk-OUT 短包时禁止缓存.
[6]	**SHORTTXEN**	**短包传输使能位** 此位只在自动确认模式下有效。如果此位置1，缓存里剩下的小于最大包大小的数据被确认作为最后一笔传输。当数据包发送后此位自动清0 0 = 缓存里数据个数小于最大包个数时，收到 IN token 不发出去(回NAK) 1 = 缓存里数据个数小于最大包个数时，收到 IN token 就发出去.
[5]	**ZEROLEN**	**0长度** 这一位用来发送0长度的包来响应 IN-token. 当此位置1，接收到IN-token 后会发送0长度的包。此位在发送完0长度的包之后自动清除 0 =收到IN-token 不发送0长度的包 1 =收到IN-token 发送0长度的包.
[4]	**HALT**	**端点挂起** 这一位用来收到主机令牌后发送 STALL握手 。当本地CPU检测到(ep_halt)特性后，需要写1清0.. 0 =从主机收到令牌后不发送 STALL 1 =从主机收到令牌后发送 STALL.
[3]	**TOGGLE**	**端点交替** 这一位用来清除端点数据交替位，读得到目前端点数据交替位 本地CPU在接收到主机的总线请求或清除特征字(ep_halt)请求时可以用这一位来建立端点交替。只有当交替位为1，这一位才可以写入 0 = 不清除端点交替位. 1 = 清除端点交替位
[2:1]	**MODE**	**模式控制** 这两位决定输入端点的操作模式. 00: 自动确认模式 01: 手动确认模式 10: 自由模式 11: 保留 这两位对输出端点无效，设为11会选择自动确认模式.
[0]	**FLUSH**	**清空缓存** 写1清除缓存以及相应的EP_AVAIL 寄存器。此位自动清0.这一位常在配置后写1. 0 = 包缓存不被清空. 1 = 清空包缓存.

端点A~L最大包长度寄存器(HSUSBD_EPAMPS~ HSUSBD_EPLMPS)

寄存器	偏移量	R/W	描述	复位值
HSUSBD_EPAMPS	HSUSBD_BA+0x078	R/W	端点A最大包长度寄存器	0x0000_0000
HSUSBD_EPBMPS	HSUSBD_BA+0x0A0	R/W	端点B最大包长度寄存器	0x0000_0000
HSUSBD_EPCMPS	HSUSBD_BA+0x0C8	R/W	端点C最大包长度寄存器	0x0000_0000
HSUSBD_EPDMPS	HSUSBD_BA+0x0F0	R/W	端点D最大包长度寄存器	0x0000_0000
HSUSBD_EPEMPS	HSUSBD_BA+0x118	R/W	端点E最大包长度寄存器	0x0000_0000
HSUSBD_EPFMPS	HSUSBD_BA+0x140	R/W	端点F最大包长度寄存器	0x0000_0000
HSUSBD_EPGMPS	HSUSBD_BA+0x168	R/W	端点G最大包长度寄存器	0x0000_0000
HSUSBD_EPHMPS	HSUSBD_BA+0x190	R/W	端点H最大包长度寄存器	0x0000_0000
HSUSBD_EPIMPS	HSUSBD_BA+0x1B8	R/W	端点I最大包长度寄存器	0x0000_0000
HSUSBD_EPJMPS	HSUSBD_BA+0x1E0	R/W	端点J最大包长度寄存器	0x0000_0000
HSUSBD_EPKMPS	HSUSBD_BA+0x208	R/W	端点K最大包长度寄存器	0x0000_0000
HSUSBD_EPLMPS	HSUSBD_BA+0x230	R/W	端点L最大包长度寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved					EPMPS		
7	6	5	4	3	2	1	0
EPMPS							

位	描述	
[31:11]	**Reserved**	保留
[10:0]	**EPMPS**	**端点最大包大小** 这里决定端点的最大包大小

端点A~L传输个数寄存器(HSUSBD_EPATXCNT~ HSUSBD_EPLTXCNT)

寄存器	偏移量	R/W	描述	复位值
HSUSBD_EPATXCNT	HSUSBD_BA+0x07C	R/W	端点A传输个数寄存器	0x0000_0000
HSUSBD_EPBTXCNT	HSUSBD_BA+0x0A4	R/W	端点B传输个数寄存器	0x0000_0000
HSUSBD_EPCTXCNT	HSUSBD_BA+0x0CC	R/W	端点C传输个数寄存器	0x0000_0000
HSUSBD_EPDTXCNT	HSUSBD_BA+0x0F4	R/W	端点D传输个数寄存器	0x0000_0000
HSUSBD_EPETXCNT	HSUSBD_BA+0x11C	R/W	端点E传输个数寄存器	0x0000_0000
HSUSBD_EPFTXCNT	HSUSBD_BA+0x144	R/W	端点F传输个数寄存器	0x0000_0000
HSUSBD_EPGTXCNT	HSUSBD_BA+0x16C	R/W	端点G传输个数寄存器	0x0000_0000
HSUSBD_EPHTXCNT	HSUSBD_BA+0x194	R/W	端点H传输个数寄存器	0x0000_0000
HSUSBD_EPITXCNT	HSUSBD_BA+0x1BC	R/W	端点I传输个数寄存器	0x0000_0000
HSUSBD_EPJTXCNT	HSUSBD_BA+0x1E4	R/W	端点J传输个数寄存器	0x0000_0000
HSUSBD_EPKTXCNT	HSUSBD_BA+0x20C	R/W	端点K传输个数寄存器	0x0000_0000
HSUSBD_EPLTXCNT	HSUSBD_BA+0x234	R/W	端点L传输个数寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved					TXCNT		
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
TXCNT							

位	描述	
[31:11]	**Reserved**	保留
[10:0]	**TXCNT**	**端点传输个数** 对于输入端点，手动确认模式这里决定发送给主机的数据个数 对输出端点无效.

端点A~L配置寄存器(HSUSBD_EPACFG~ HSUSBD_EPLCFG)

寄存器	偏移量	R/W	描述	复位值
HSUSBD_EPACFG	HSUSBD_BA+0x080	R/W	端点A配置寄存器	0x0000_0012
HSUSBD_EPBCFG	HSUSBD_BA+0x0A8	R/W	端点B配置寄存器	0x0000_0022
HSUSBD_EPCCFG	HSUSBD_BA+0x0D0	R/W	端点C配置寄存器	0x0000_0032
HSUSBD_EPDCFG	HSUSBD_BA+0x0F8	R/W	端点D配置寄存器	0x0000_0042
HSUSBD_EPECFG	HSUSBD_BA+0x120	R/W	端点E配置寄存器	0x0000_0052
HSUSBD_EPFCFG	HSUSBD_BA+0x148	R/W	端点F配置寄存器	0x0000_0062
HSUSBD_EPGCFG	HSUSBD_BA+0x170	R/W	端点G配置寄存器	0x0000_0072
HSUSBD_EPHCFG	HSUSBD_BA+0x198	R/W	端点H配置寄存器	0x0000_0082
HSUSBD_EPICFG	HSUSBD_BA+0x1C0	R/W	端点I配置寄存器	0x0000_0092
HSUSBD_EPJCFG	HSUSBD_BA+0x1E8	R/W	端点J配置寄存器	0x0000_00A2
HSUSBD_EPKCFG	HSUSBD_BA+0x210	R/W	端点K配置寄存器	0x0000_00B2
HSUSBD_EPLCFG	HSUSBD_BA+0x238	R/W	端点L配置寄存器	0x0000_00C2

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
EPNUM				**EPDIR**	**EPTYPE**		**EPEN**

位	描述	
[31:6]	**Reserved**	保留
[7:4]	**EPNUM**	**端点号** 这里选择端点号，有效值是1到 15 **注:** 不支持两个端点有同一个端点号

[3]	**EPDIR**	**端点方向** 0 = 输出端点(主机输出到设备) 1 = 输入端点(主机从设备输入) **注：**每个端点号只支持一个OUT和IN端点
[2:1]	**EPTYPE**	**端点类型** 这里选择端点类型，端点0是控制端点 00 = 保留 01 = 批量传输 10 = 中断传输 11 = 等时传输
[0]	**EPEN**	**端点有效** 置1使能端点，这一位对端点0无效，端点0总是使能 0 = 禁止端点 1 = 使能端点

端点A~L内存起始地址寄存器(HSUSBD_EPABUFSTART~ HSUSBD_EPLBUFSTART)

寄存器	偏移量	R/W	描述	复位值
HSUSBD_EPABUFSTART	HSUSBD_BA+0x084	R/W	端点A内存起始地址寄存器	0x0000_0000
HSUSBD_EPBBUFSTART	HSUSBD_BA+0x0AC	R/W	端点B内存起始地址寄存器	0x0000_0000
HSUSBD_EPCBUFSTART	HSUSBD_BA+0x0D4	R/W	端点C内存起始地址寄存器	0x0000_0000
HSUSBD_EPDBUFSTART	HSUSBD_BA+0x0FC	R/W	端点D内存起始地址寄存器	0x0000_0000
HSUSBD_EPEBUFSTART	HSUSBD_BA+0x124	R/W	端点E内存起始地址寄存器	0x0000_0000
HSUSBD_EPFBUFSTART	HSUSBD_BA+0x14C	R/W	端点F内存起始地址寄存器	0x0000_0000
HSUSBD_EPGBUFSTART	HSUSBD_BA+0x174	R/W	端点G内存起始地址寄存器	0x0000_0000
HSUSBD_EPHBUFSTART	HSUSBD_BA+0x19C	R/W	端点H内存起始地址寄存器	0x0000_0000
HSUSBD_EPIBUFSTART	HSUSBD_BA+0x1C4	R/W	端点I内存起始地址寄存器	0x0000_0000
HSUSBD_EPJBUFSTART	HSUSBD_BA+0x1EC	R/W	端点J内存起始地址寄存器	0x0000_0000
HSUSBD_EPKBUFSTART	HSUSBD_BA+0x214	R/W	端点K内存起始地址寄存器	0x0000_0000
HSUSBD_EPLBUFSTART	HSUSBD_BA+0x23C	R/W	端点L内存起始地址寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved				**SADDR**			
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
SADDR							

位	描述	
[31:12]	**Reserved**	保留
[11:0]	**SADDR**	**端点起始地址** 这里是对应端点的内存起始地址

端点A~L 内存结束地址寄存器(HSUSBD_EPABUFEND~ HSUSBD_EPLBUFEND)

寄存器	偏移量	R/W	描述	复位值
HSUSBD_EPABUFEND	HSUSBD_BA+0x088	R/W	端点A内存结束地址寄存器	0x0000_0000
HSUSBD_EPBBUFEND	HSUSBD_BA+0x0B0	R/W	端点B内存结束地址寄存器	0x0000_0000
HSUSBD_EPCBUFEND	HSUSBD_BA+0x0D8	R/W	端点C内存结束地址寄存器	0x0000_0000
HSUSBD_EPDBUFEND	HSUSBD_BA+0x100	R/W	端点D内存结束地址寄存器	0x0000_0000
HSUSBD_EPEBUFEND	HSUSBD_BA+0x128	R/W	端点E内存结束地址寄存器	0x0000_0000
HSUSBD_EPFBUFEND	HSUSBD_BA+0x150	R/W	端点F内存结束地址寄存器	0x0000_0000
HSUSBD_EPGBUFEND	HSUSBD_BA+0x178	R/W	端点G内存结束地址寄存器	0x0000_0000
HSUSBD_EPHBUFEND	HSUSBD_BA+0x1A0	R/W	端点H内存结束地址寄存器	0x0000_0000
HSUSBD_EPIBUFEND	HSUSBD_BA+0x1C8	R/W	端点I内存结束地址寄存器	0x0000_0000
HSUSBD_EPJBUFEND	HSUSBD_BA+0x1F0	R/W	端点J内存结束地址寄存器	0x0000_0000
HSUSBD_EPKBUFEND	HSUSBD_BA+0x218	R/W	端点K内存结束地址寄存器	0x0000_0000
HSUSBD_EPLBUFEND	HSUSBD_BA+0x240	R/W	端点L内存结束地址寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved				**EADDR**			
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
EADDR							

位	描述	
[31:12]	**Reserved**	保留
[11:0]	**EADDR**	**端点结束地址** 这里是对应端点的内存结束地址

AHB地址寄存器(HSUSBD_DMAADDR)

寄存器	偏移量	R/W	描述	复位值
HSUSBD_DMAADDR	HSUSBD_BA+0x700	R/W	AHB DMA地址寄存器	0x0000_0000

31	30	29	28	27	26	25	24
DMAADDR							
23	22	21	20	19	18	17	16
DMAADDR							
15	14	13	12	11	10	9	8
DMAADDR							
7	6	5	4	3	2	1	0
DMAADDR							

位	描述	
[31:0]	DMAADDR	DMAADDR 这里指定 DMA 的读写地址，地址必须32位对齐

USB PHY控制寄存器(HSUSBD_PHYCTL)

寄存器	偏移量	R/W	描述	复位值
HSUSBD_PHYCTL	HSUSBD_BA+0x704	R/W	USB PHY控制寄存器	0x0000_0420

31	30	29	28	27	26	25	24
VBUSDET	**Reserved**						**WKEN**
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved						**PHYEN**	**DPPUEN**
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved							

位	描述	
[31]	**VBUSDET**	**VBUS 状态** 0 = 没有检测到VBUS 1 = 检测到 VBUS
[30:25]	**Reserved**	保留
[24]	**WKEN**	**唤醒使能位** 0 = 禁止唤醒功能 1 = 使能唤醒功能
[23:10]	**Reserved**	保留
[9]	**PHYEN**	**PHY 挂起使能位** 0 = USB PHY挂起 1 = USB PHY未挂起
[8]	**DPPUEN**	**DP 上拉** 0 = 禁止 D+上拉电阻 1 = 使能D+上拉电阻
[7:0]	**Reserved**	保留

6.33 USB 1.1 主控制器 (USBH)

6.33.1 概述

本芯片带有USB1.1主机控制器(USBH)，支持Open Host Controller Interface (OpenHCI, OHCI)规范，主控制器寄存器用来管理设备和USB总线数据传输。

USBH集成一个根集线器和一个USB端口，有一个DMA用来在系统内存和USB总线之间传送实时数据，端口有电源控制和过电流检测功能。

USBH负责检测USB设备的插拔，管理数据传输，收集USB状态信息，激活USB总线，提供电源控制以及检测USB设备的电流。

6.33.2 特性

- 支持USB总线规范1.1
- 支持Open Host Controller Interface (OpenHCI) 规范 1.0.
- 支持全速 (12Mbps) 及低速 (1.5Mbps) USB 设备
- 支持控制，批量，中断和等时传输
- 支持一个集成的根集线器
- 支持一个USB主/从机共享的端口(OTG 功能).
- 支持电源控制及端口过电流检测
- 支持DMA实时传输数据

6.33.3 框图

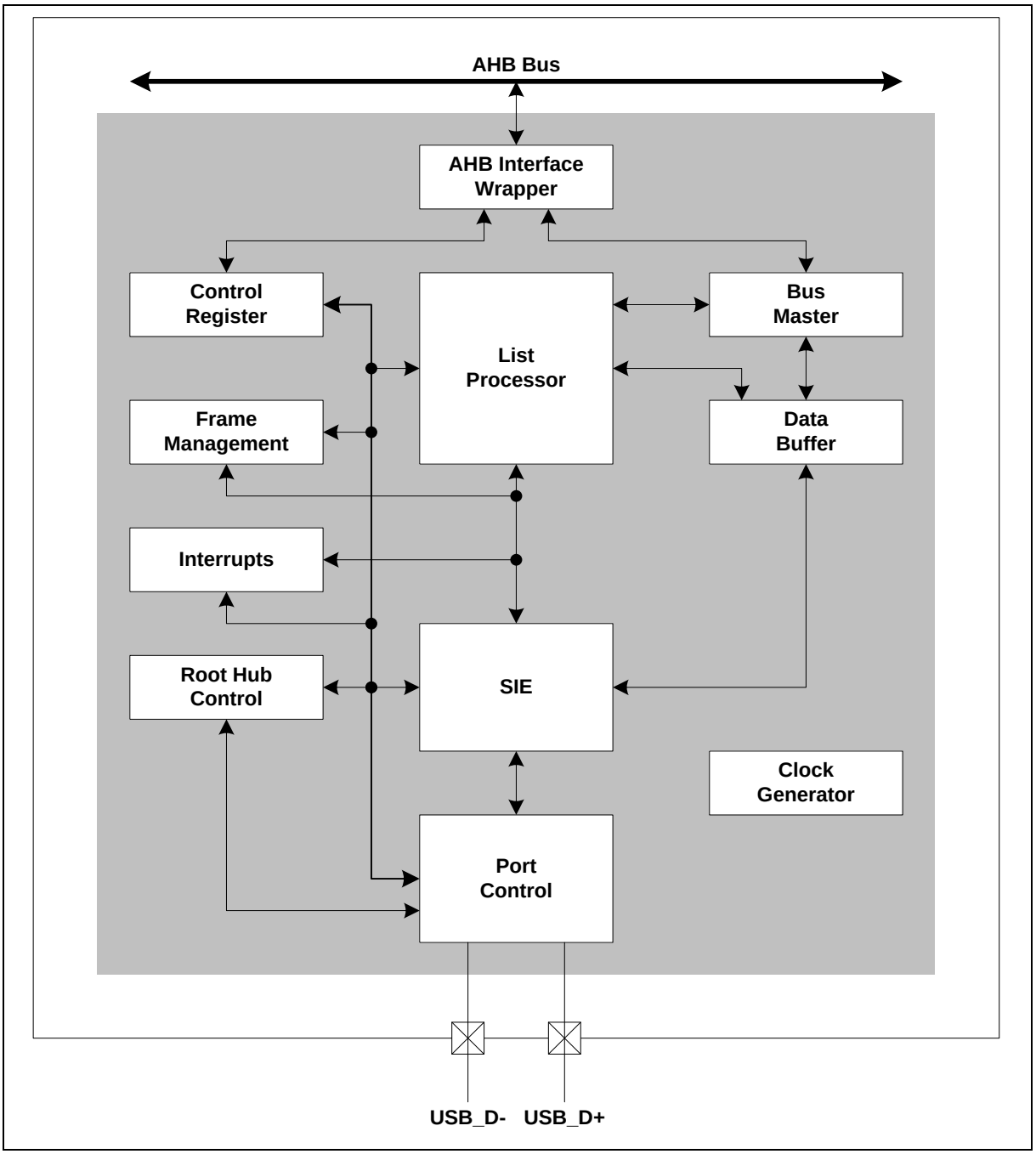


图 6.33-1 USB 1.1 主控制器框图

6.33.4 基本配置

6.33.4.1 USBH OHCI 控制器基本配置

- 时钟源配置
  - USBSEL (CLK_CLKSEL0[8])选择USBH的时钟源
  - USBDIV (CLK_CLKDIV0[7:4])设置USBH的时钟除频
  - USBHCKEN (CLK_AHBCLK[16])使能USBH时钟
  - USBDCKEN(CLK_APBCLK0[27]) 使能USBD时钟
- 复位配置
  - USBHRST (SYS_IPRST0[4])复位USBH控制器
- 管脚配置

组	管脚名称	GPIO	MFP
USB	USB_D+	PA.14	MFP14
	USB_D-	PA.13	MFP14
	USB_OTG_ID	PA.15	MFP14
	USB_VBUS	PA.12	MFP14
	USB_VBUS_EN	PB.6, PB.15	MFP14
	USB_VBUS_ST	PB.14	MFP15
		PB.7, PC.14	MFP14

表 6.33-1 USB 1.1 主控制器管脚配置

6.33.5 功能描述

6.33.5.1 OHCI 控制器

AHB 接口

OpenHCI主控制器通过AHB总线接入系统，要求同时支持主/从总线操作。作为主机时，主控制器负责AHB总线和端点描述符（EDs）及传输描述符(TDs)的数据传输，以及内存与片上数据缓存之间的数据传输。作为从机时，主控制器监控AHB总线上的数据命令，并决定何时回应这些命令。通过AHB总线的从机接口，可以配置和非实时操作主控制器的操作寄存器

主控制器

主控制器有5个功能模块，包括列表处理，帧管理，中断处理，主控制器总线主机以及数据缓存。

列表处理功能会管理主控制器驱动程序的数据结构，并协调主控制器内的所有动作。

帧管理负责管理USB和OpenHCI规格规定的帧的特定任务，这些任务是：

- 管理OpenHCI帧特定的操作寄存器
- 最大数据包计数器操作
- USB对SIE传送请求的帧验证
- 向SIE产生SOF token 请求

主控制器与主控制器驱动间的通讯是通过中断方式实现的，主控制器有几种事件可以触发中断，每个事

件会在HcInterruptStatus寄存器设置相应的标志位。

在数据通路中主机总线控制器是核心单元，它协调所有与AHB接口的操作，总线控制器有两个数据来源，列表处理器和数据缓存引擎。

数据缓存作为总线控制器和SIE间的数据接口，是一个64字节双向异步FIFO和一个DWORD AHB保持寄存器的组合。

M48xID/M48xGA 的 USB 接口

USB接口集成了USB端口和根集线器，端口1也是串行接口引擎(SIE)和USB时钟发生器，负责主控制器发出的总线传输请求及USB协议定义的集线器和端口的管理。

SIE负责管理所有USB的传输，控制总线协议，数据包的打包/拆包，数据的并串转换，CRC校验，位填充以及NRZI 编码，所有USB的传输都是由列表处理器和帧管理器发起的。

根集线器是一个分别控制的端口集合，它通过一些相同的功能模块维持所有端口的控制/状态。

M48xGC/M48xE8 的 USB 接口

USB接口集成了USB端口和根集线器，端口2也是串行接口引擎(SIE)和USB时钟发生器，负责主控制器发出的总线传输请求及USB协议定义的集线器和端口的管理。

SIE负责管理所有USB的传输，控制总线协议，数据包的打包/拆包，数据的并串转换，CRC校验，位填充以及NRZI 编码，所有USB的传输都是由列表处理器和帧管理器发起的。

根集线器是一个分别控制的端口集合，它通过一些相同的功能模块维持所有端口的控制/状态。

6.33.6 寄存器表

R: 只读, **W**: 只写, **R/W**: 可读写

寄存器	偏移量	R/W	描述	复位值
USBH 基地址: **USBH_BA = 0x4000_9000**				
HcRevision	USBH_BA+0x000	R	主控制器版本寄存器	0x0000_0110
HcControl	USBH_BA+0x004	R/W	主控制器控制寄存器	0x0000_0000
HcCommandStatus	USBH_BA+0x008	R/W	主控制器命令状态寄存器	0x0000_0000
HcInterruptStatus	USBH_BA+0x00C	R/W	主控制器中断状态寄存器	0x0000_0000
HcInterruptEnable	USBH_BA+0x010	R/W	主控制器中断使能寄存器	0x0000_0000
HcInterruptDisable	USBH_BA+0x014	R/W	主控制器中断禁止寄存器	0x0000_0000
HcHCCA	USBH_BA+0x018	R/W	主控制器通信控制区内存地址寄存器	0x0000_0000
HcPeriodCurrentED	USBH_BA+0x01C	R/W	主控制器当前周期端点描述符地址寄存器	0x0000_0000
HcControlHeadED	USBH_BA+0x020	R/W	主控制器控制端点描述符首地址寄存器	0x0000_0000
HcControlCurrentED	USBH_BA+0x024	R/W	主控制器控制端点描述符当前地址寄存器	0x0000_0000
HcBulkHeadED	USBH_BA+0x028	R/W	主控制器批量端点描述符首地址寄存器	0x0000_0000
HcBulkCurrentED	USBH_BA+0x02C	R/W	主控制器批量端点描述符当前地址寄存器	0x0000_0000
HcDoneHead	USBH_BA+0x030	R/W	主控制器完成了的传输描述符首地址寄存器	0x0000_0000
HcFmInterval	USBH_BA+0x034	R/W	主控制器帧间隔寄存器	0x0000_2EDF
HcFmRemaining	USBH_BA+0x038	R	主控制器帧剩余时间寄存器	0x0000_0000
HcFmNumber	USBH_BA+0x03C	R	主控制器帧计数器	0x0000_0000
HcPeriodicStart	USBH_BA+0x040	R/W	主控制器周期列表开始处理寄存器	0x0000_0000
HcLSThreshold	USBH_BA+0x044	R/W	主控制器低速度门槛寄存器	0x0000_0628
HcRhDescriptorA	USBH_BA+0x048	R/W	主控制器根集线器描述符A寄存器	0x0100_0001
HcRhDescriptorB	USBH_BA+0x04C	R/W	主控制器根集线器描述符B寄存器	0x0000_0000
HcRhStatus	USBH_BA+0x050	R/W	主控制器根集线器状态寄存器	0x0000_0000
HcRhPortStatus1	USBH_BA+0x058	R/W	主控制器根集线器端口状态[1]	0x0000_0000
HcPhyControl	USBH_BA+0x200	R/W	主控制器PHY 控制寄存器	0x080F_0000
HcMiscControl	USBH_BA+0x204	R/W	主控制器多功能控制寄存器	0x0000_0000

6.33.7 寄存器描述

主控制器版本寄存器(HcRevision)

寄存器	偏移量	R/W	描述	复位值
HcRevision	USBH_BA+0x000	R	主控制器版本寄存器	0x0000_0110

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
REV							

位	描述	
[31:8]	**Reserved**	保留
[7:0]	**REV**	**版本号** 表示硬件匹配的OpenHCI规格版本号 主控制器支持1.1版本规范 (X.Y = XYh).

主控制器控制寄存器(HcControl)

寄存器	偏移量	R/W	描述	复位值
HcControl	USBH_BA+0x004	R/W	主控制器控制寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
HCFS		**BLE**	**CLE**	**IE**	**PLE**	**CBSR**	

位	描述	
[31:8]	**Reserved**	保留
[7:6]	**HCFS**	**主控制器功能状态** 设置主控制器状态，当控制器从下行端口收到RESUME信号后，可以强制将状态从USBSUSPEND改变为USBRESUME 可以有如下状态： 00 = USBSUSPEND. 01 = USBOPERATIONAL. 10 = USBRESUME. 11 = USBRESET.
[5]	**BLE**	**批量列表使能位** 0 = 下一个SOF (Start-Of-Frame)后禁止处理批量列表 1 = 使能下一帧处理批量列表.
[4]	**CLE**	**控制列表使能位** 0 = 下一个SOF (Start-Of-Frame)后禁止处理控制列表 1 = 使能下一帧处理控制列表
[3]	**IE**	**同步列表使能位** ISOEn 及 PLE (HcControl[2]) 为高使能主控制器处理同步列表， ISOEn 或 PLE (HcControl[2])为低禁止主控制器处理同步列表 0 = 下一个SOF (Start-Of-Frame)后禁止处理同步列表 1 = 如果PLE (HcControl[2])也为高，使能下一帧处理同步列表

[2]	**PLE**	**周期列表使能位** 为高时，使能处理周期（中断和同步）列表，主控制器在每一帧要开始周期传输之前都会检测该位 0 = 下一个SOF (Start-Of-Frame)后禁止处理周期（中断和同步）列表 1 = 使能下一帧处理周期列表 **注:**如果用户想使能同步列表，必须设置PLE 及IE (HcControl[3])为高.
[1:0]	**CBSR**	**控制/批量处理比例** 这里定义了控制/批量端点描述符的处理时间比例。在处理非周期列表之前，主控制器(HC)会将代表已处理过的控制端点描述符的内部计数器跟这里定义的比例做比对，以决定是继续处理控制端点描述符还是切换到处理批量端点描述符。当处理跨越多个帧时，内部计数器会被保存。如果发生复位，主控制器驱动(HCD)负责保存这个值。 00 = 控制端点描述符与批量端点描述符处理时间的比值是1:1. 01 = 控制端点描述符与批量端点描述符处理时间的比值是 2:1. 10 = 控制端点描述符与批量端点描述符处理时间的比值是 3:1. 11 = 控制端点描述符与批量端点描述符处理时间的比值是4:1.

主控制器命令状态寄存器(HcCommandStatus)

寄存器	偏移量	R/W	描述	复位值
HcCommandStatus	USBH_BA+0x008	R/W	主控制器命令状态寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved						SOC	
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved					BLF	CLF	HCR

位	描述	
[31:18]	**Reserved**	保留
[17:16]	**SOC**	**调度超时计数** 每次发生调度超时错误这里会加一，复位为00b, 计数到11b后复位。就算SO(HcInterruptStatus[0])被置位了，发生调度超时错误时这里还会加一
[15:3]	**Reserved**	保留
[2]	**BLF**	**批量列表满** 设置为1表示批量列表上有使能的传输描述符。此位可能被软件或者主控制器设置，主控制器开始处理批量列表时将此位清零 0 = 没有发现使能的传输描述符或者主控制器开始处理批量列表了. 1 = 在批量列表发现或者添加了使能的传输描述符.
[1]	**CLF**	**控制列表满** 设置为1表示控制列表上有使能的传输描述符。此位可能被软件或者主控制器设置，主控制器开始处理控制列表时将此位清零 0 = 没有发现使能的传输描述符或者主控制器开始处理控制列表了 1 = 在控制列表发现或者添加了使能的传输描述符..
[0]	**HCR**	**主控制器复位** 软件设置此位为1复位主控制器，复位操作完成后主控制器将此位清零 设置此位时不会复位根集线器，也没有复位信号发到下行端口 0 = 主控制器不在软件复位状态 1 = 主控制器在软件复位状态

主控制器中断状态寄存器(HcInterruptStatus)

寄存器	偏移量	R/W	描述	复位值
HcInterruptStatus	USBH_BA+0x00C	R/W	主控制器中断状态寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved	**RHSC**	**FNO**	**Reserved**	**RD**	**SF**	**WDH**	**SO**

位	描述	
[31:7]	**Reserved**	保留
[6]	**RHSC**	**根集线器状态改变** 当寄存器HcRhStatus或HcRhPortStatus1的内容改变时，该位被设置 0 = 寄存器HcRhStatus和HcRhPortStatus1的内容没有改变. 1 = 寄存器HcRhStatus或HcRhPortStatus1的内容改变了.
[5]	**FNO**	**帧计数器溢出** 当帧计数器的Bit15从1变为0，或者从0变为1时该位被设置 0 = 帧计数器的Bit15没有变化 1 = 帧计数器的Bit15产生了从1到0，或者从0到1的变化.
[4]	**Reserved**	保留
[3]	**RD**	**唤醒检测** 主控制器检测到下行端口有唤醒信号时设置该位. 0 = 下行端口没有唤醒信号 1 = 下行端口有唤醒信号.
[2]	**SF**	**帧开始** 帧管理模块发出'帧开始'信号时设置此位，同时主控制器将产生一个SOF token 0 = 没有帧开始信号 1 = 有帧开始信号，同时主控制器产生一个SOF token
[1]	**WDH**	**完成的传输描述符头地址写回** 主控制器将完成了的传输描述符头地址HcDoneHead写到寄存器HccaDoneHead时此位置高，此位被清零前HccaDoneHead不能再次被更新 0 = 主控制器没有更新HccaDoneHead. 1 = 主控制器写HcDoneHead 到 HccaDoneHead.

[0]	**SO**	**调度溢出** 当列表处理器调度溢出时设置此位. 0 = 没有发生调度溢出 1 = 发生了调度溢出

主控制器中断使能寄存器(HcInterruptEnable)

寄存器	偏移量	R/W	描述	复位值
HcInterruptEnable	USBH_BA+0x010	R/W	主控制器中断使能寄存器	0x0000_0000

31	30	29	28	27	26	25	24
MIE	Reserved						
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved	RHSC	FNO	Reserved	RD	SF	WDH	SO

位	描述	
[31]	**MIE**	**主中断使能位** 此位是全局中断使能位，写1后使能本寄存器其他位的中断使能 写操作: 0 = 没有影响 1 = 如果寄存器HcInterruptEnable相应的位为1，则由以下原因造成的中断会被使能：RHSC (HcInterruptStatus[6]), FNO (HcInterruptStatus[5]), RD (HcInterruptStatus[3]), SF (HcInterruptStatus[2]), WDH (HcInterruptStatus[1]) 以及 SO (HcInterruptStatus[0]) 读操作: 0 = 就算HcInterruptEnable相应位为1，由以下原因造成的中断也不会被使能： RHSC (HcInterruptStatus[6]), FNO (HcInterruptStatus[5]), RD (HcInterruptStatus[3]), SF (HcInterruptStatus[2]), WDH (HcInterruptStatus[1]) 以及 SO (HcInterruptStatus[0]) 1 = 如果HcInterruptEnable相应位为1，由以下原因造成的中断将被使能： RHSC (HcInterruptStatus[6]), FNO (HcInterruptStatus[5]), RD (HcInterruptStatus[3]), SF (HcInterruptStatus[2]), WDH (HcInterruptStatus[1]) 以及 SO (HcInterruptStatus[0])
[30:7]	**Reserved**	保留
[6]	**RHSC**	**根集线器状态改变中断使能位** 写操作: 0 = 没有影响. 1 = RHSC (HcInterruptStatus[6])产生的中断将被使能 读操作: 0 = RHSC (HcInterruptStatus[6])产生的中断将被禁止 1 = RHSC (HcInterruptStatus[6])产生的中断将被使能

[5]	**FNO**	**帧计数溢出中断使能位** 写操作: 0 = 没有影响. 1 = FNO (HcInterruptStatus[5]) 产生的中断使能 读操作: 0 = FNO (HcInterruptStatus[5])产生的中断被禁止. 1 = FNO (HcInterruptStatus[5])产生的中断被使能
[4]	**Reserved**	保留
[3]	**RD**	**唤醒检测中断使能位** 写操作: 0 = 没有影响. 1 = RD (HcInterruptStatus[3]) 产生的中断使能 读操作: 0 = RD (HcInterruptStatus[3])产生的中断被禁止. 1 = RD (HcInterruptStatus[3])产生的中断被使能
[2]	**SF**	**帧开始中断使能** 写操作: 0 = 没有影响. 1 = SF (HcInterruptStatus[2]) 产生的中断使能 读操作: 0 = SF (HcInterruptStatus[2])产生的中断被禁止. 1 = SF (HcInterruptStatus[2])产生的中断被使能
[1]	**WDH**	**完成的传输描述符头地址写回中断使能位** 写操作: 0 = 没有影响. 1 = WDH (HcInterruptStatus[1])产生的中断使能 读操作: 0 = WDH (HcInterruptStatus[1])产生的中断被禁止. 1 = WDH (HcInterruptStatus[1])产生的中断被使能
[0]	**SO**	**调度溢出中断使能位** 写操作: 0 = 没有影响. 1 = SO (HcInterruptStatus[0]) 产生的中断使能 读操作: 0 = SO (HcInterruptStatus[0]) 产生的中断被禁止. 1 = SO (HcInterruptStatus[0]) 产生的中断被使能

主控制器中断禁止寄存器(HcInterruptDisable)

寄存器	偏移量	R/W	描述	复位值
HcInterruptDisable	USBH_BA+0x014	R/W	主控制器中断禁止寄存器	0x0000_0000

31	30	29	28	27	26	25	24
MIE	Reserved						
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved	RHSC	FNO	Reserved	RD	SF	WDH	SO

位	描述	
[31]	MIE	**主中断禁止位** 此位是全局中断禁止位，写1后禁止所有中断 写操作: 0 = 没有影响 1 = 就算寄存器HcInterruptEnable相应的位为1，由以下原因造成的中断也会被禁止：RHSC (HcInterruptStatus[6]), FNO (HcInterruptStatus[5]), RD (HcInterruptStatus[3]), SF (HcInterruptStatus[2]), WDH (HcInterruptStatus[1]) 以及 SO (HcInterruptStatus[0]) 读操作: 0 = 就算HcInterruptEnable相应位为1，由以下原因造成的中断也不会被使能： RHSC (HcInterruptStatus[6]), FNO (HcInterruptStatus[5]), RD (HcInterruptStatus[3]), SF (HcInterruptStatus[2]), WDH (HcInterruptStatus[1]) 以及 SO (HcInterruptStatus[0]) 1 = 如果HcInterruptEnable相应位为1，由以下原因造成的中断被使能： RHSC (HcInterruptStatus[6]), FNO (HcInterruptStatus[5]), RD (HcInterruptStatus[3]), SF (HcInterruptStatus[2]), WDH (HcInterruptStatus[1]) 以及 SO (HcInterruptStatus[0])
[30:7]	Reserved	保留
[6]	RHSC	**根集线器状态改变中断禁止位** 写操作: 0 = 没有影响. 1 = RHSC (HcInterruptStatus[6])产生的中断将被禁止 读操作: 0 = RHSC (HcInterruptStatus[6])产生的中断被禁止 1 = RHSC (HcInterruptStatus[6])产生的中断被使能

[5]	**FNO**	**帧计数溢出中断禁止位** 写操作: 0 = 没有影响. 1 = FNO (HcInterruptStatus[5]) 产生的中断禁止 读操作: 0 = FNO (HcInterruptStatus[5])产生的中断被禁止 1 = FNO (HcInterruptStatus[5])产生的中断被使能.
[4]	**Reserved**	保留
[3]	**RD**	**唤醒检测中断禁止位** 写操作: 0 = 没有影响. 1 = RD (HcInterruptStatus[3]) 产生的中断禁止 读操作: 1 = RD (HcInterruptStatus[3])产生的中断被使能. 0 = RD (HcInterruptStatus[3])产生的中断被禁止
[2]	**SF**	**帧开始中断禁止** 写操作: 0 = 没有影响. 1 = SF (HcInterruptStatus[2]) 产生的中断禁止 读操作: 1 = SF (HcInterruptStatus[2])产生的中断被使能. 0 = SF (HcInterruptStatus[2])产生的中断被禁止
[1]	**WDH**	**完成的传输描述符头地址写回中断禁止位** 写操作: 0 = 没有影响. 1 = WDH (HcInterruptStatus[1])产生的中断禁止 读操作: 1 = WDH (HcInterruptStatus[1])产生的中断被使能. 0 = WDH (HcInterruptStatus[1])产生的中断被禁止
[0]	**SO**	**调度溢出中断禁止位** 写操作: 0 = 没有影响. 1 = SO (HcInterruptStatus[0]) 产生的中断禁止 读操作: 1 = SO (HcInterruptStatus[0]) 产生的中断被使能. 0 = SO (HcInterruptStatus[0]) 产生的中断被禁止

主控制器通信控制区内存地址寄存器(HcHCCA)

寄存器	偏移量	R/W	描述	复位值
HcHCCA	USBH_BA+0x018	R/W	主控制器通信控制区内存地址寄存器	0x0000_0000

31	30	29	28	27	26	25	24
HCCA							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
HCCA							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
HCCA							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved							

位	描述	
[31:8]	**HCCA**	**主控制器通信控制区** 主控制器通信控制区的基地址指针
[7:0]	**Reserved**	保留

主控制器当前周期端点描述符地址寄存器(HcPeriodCurrentED)

寄存器	偏移量	R/W	描述	复位值
HcPeriodCurrentED	USBH_BA+0x01C	R/W	主控制器当前周期端点描述符地址寄存器	0x0000_0000

31	30	29	28	27	26	25	24
PCED							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
PCED							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
PCED							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
PCED				**Reserved**			

位	描述	
[31:4]	**PCED**	**当前周期端点描述符** 当前同步或中断端点描述符的物理地址指针
[3:0]	**Reserved**	保留

主控制器控制端点描述符首地址寄存器(HcControlHeadED)

寄存器	偏移量	R/W	描述	复位值
HcControlHeadED	USBH_BA+0x020	R/W	主控制器控制端点描述符首地址寄存器	0x0000_0000

31	30	29	28	27	26	25	24
CHED							
23	22	21	20	19	18	17	16
CHED							
15	14	13	12	11	10	9	8
CHED							
7	6	5	4	3	2	1	0
CHED				Reserved			

位	描述	
[31:4]	CHED	**控制描述符首地址** 控制列表第一个端点描述符的物理地址指针
[3:0]	Reserved	保留

主控制器控制端点描述符当前地址寄存器(HcControlCurrentED)

寄存器	偏移量	R/W	描述	复位值
HcControlCurrentED	USBH_BA+0x024	R/W	主控制器控制端点描述符当前地址寄存器	0x0000_0000

31	30	29	28	27	26	25	24
CCED							
23	22	21	20	19	18	17	16
CCED							
15	14	13	12	11	10	9	8
CCED							
7	6	5	4	3	2	1	0
CCED				Reserved			

位	描述	
[31:4]	CCED	**控制端点描述符当前地址** 控制列表当前端点描述符物理地址指针
[3:0]	Reserved	保留

主控制器批量端点描述符首地址寄存器(HcBulkHeadED)

寄存器	偏移量	R/W	描述	复位值
HcBulkHeadED	USBH_BA+0x028	R/W	主控制器批量端点描述符首地址寄存器	0x0000_0000

31	30	29	28	27	26	25	24
BHED							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
BHED							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
BHED							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
BHED				**Reserved**			

位	描述	
[31:4]	**BHED**	**批量端点描述符首地址** 批量列表端点描述符首地址指针
[3:0]	**Reserved**	保留

主控制器批量端点描述符当前地址寄存器(HcBulkCurrentED)

寄存器	偏移量	R/W	描述	复位值
HcBulkCurrentED	USBH_BA+0x02C	R/W	主控制器批量端点描述符当前地址寄存器	0x0000_0000

31	30	29	28	27	26	25	24
BCED							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
BCED							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
BCED							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
BCED				**Reserved**			

位	描述	
[31:4]	**BCED**	**批量端点描述符当前地址** 批量列表当前描述符物理地址指针
[3:0]	**Reserved**	保留

主控制器已完成传输描述符首地址寄存器(HcDoneHead)

寄存器	偏移量	R/W	描述	复位值
HcDoneHead	USBH_BA+0x030	R/W	主控制器已完成传输描述符首地址寄存器	0x0000_0000

31	30	29	28	27	26	25	24
DH							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
DH							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
DH							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
DH				**Reserved**			

位	描述	
[31:4]	**DH**	**已完成的传输描述符首地址** 加入完成队列的最新完成的传输描述符物理地址指针
[3:0]	**Reserved**	保留

主控制器帧间隔寄存器(HcFmInterval)

寄存器	偏移量	R/W	描述	复位值
HcFmInterval	USBH_BA+0x034	R/W	主控制器帧间隔寄存器	0x0000_2EDF

31	30	29	28	27	26	25	24
FIT	**FSMPS**						
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
FSMPS							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved		**FI**					
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
FI							

位	描述	
[31]	**FIT**	**帧间隔反转** 当主控制器驱动加载一个新的值到FI (HcFmInterval[13:0])中，此位反转 0 = 主控制驱动没有加载新的值到FI (HcFmInterval[13:0]). 1 = 主控制驱动加载新的值到FI (HcFmInterval[13:0]).
[30:16]	**FSMPS**	**最大数据包** 这里定义了帧开始时加载到最大数据包计数器中的值
[15:14]	**Reserved**	保留
[13:0]	**FI**	**帧间隔** 这里以位时间（bit time）定义了帧的长度为（位时间-1），例如，一帧12,000位时间这里应当存储11999

主控制器帧剩余时间寄存器(HcFmRemaining)

寄存器	偏移量	R/W	描述	复位值
HcFmRemaining	USBH_BA+0x038	R	主控制器帧剩余时间寄存器	0x0000_0000

31	30	29	28	27	26	25	24
FRT	**Reserved**						
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved		**FR**					
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
FR							

位	描述	
[31]	**FRT**	**帧剩余反转** 当FR (HcFmRemaining[13:0])到0时，此位从FIT (HcFmInterval[31])加载
[30:14]	**Reserved**	保留
[13:0]	**FR**	**帧剩余** 当主控制器在USBOPERATIONAL状态时，FR每个12MHz时钟周期减1，当减到0时（帧结束）计数器重新装载帧间隔，此外，当主控制器从其他状态切换到USBOPERATIONAL时计数器也会重新装载

主控制器帧计数器(HcFmNumber)

寄存器	偏移量	R/W	描述	复位值
HcFmNumber	USBH_BA+0x03C	R	主控制器帧计数器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
FN							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
FN							

位	描述	
[31:16]	**Reserved**	保留
[15:0]	**FN**	**帧计数器** 在FR (HcFmRemaining[13:0])重新装载时此计数器加一，计数从‘FFFFh’ 到 ‘0h’循环

主控制器周期列表开始处理寄存器(HcPeriodicStart)

寄存器	偏移量	R/W	描述	复位值
HcPeriodicStart	USBH_BA+0x040	R/W	主控制器周期列表开始处理寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved		**PS**					
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
PS							

位	描述	
[31:14]	**Reserved**	保留
[13:0]	**PS**	**周期列表开始处理** 一个帧中，列表处理器检测PS决定何时开始处理周期列表

主控制器低速度门槛寄存器(HcLSThreshold)

寄存器	偏移量	R/W	描述	复位值
HcLSThreshold	USBH_BA+0x044	R/W	主控制器低速度门槛寄存器	0x0000_0628

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved				**LST**			
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
LST							

位	描述	
[31:12]	**Reserved**	保留
[11:0]	**LST**	**低速门槛** 主控制器驱动在启动低速传输前先将LST的值与FR (HcFmRemaining[13:0])比较只有当FR (HcFmRemaining[13:0])>=LST的时候才会启动低速传输

主控制器根集线器描述符A寄存器(HcRhDescriptorA)

寄存器	偏移量	R/W	描述	复位值
HcRhDescriptorA	USBH_BA+0x048	R/W	主控制器根集线器描述符A寄存器	0x0100_0001

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved			NOCP	OCPM	Reserved		PSM
7	6	5	4	3	2	1	0
NDP							

位	描述	
[31:13]	**Reserved**	保留
[12]	**NOCP**	**无过电流保护** 此位描述了当根集线器端口过电流是否会被报告 0 = 过电流状态被报告 1 = 过电流状态不会被报告
[11]	**OCPM**	**过电流保护模式** 此位描述了根集线器过电流状态如何被报告，此位只有在NOCP (HcRhDescriptorA[12])被清零时有效 0 = 所有下行端口过电流状态都会被报告 1 = 每个端口过电流状态被单独报告
[10:9]	**Reserved**	保留
[8]	**PSM**	**电源切换模式** 此位用来定义根集线器电源切换的方式 0 = 所有端口同时供电 1 = 每个端口可单独供电
[7:0]	**NDP**	**下行端口数量** 主控制器支持2个下行端口，但是在此芯片中只有1个端口可用

主控制器根集线器描述符B寄存器(HcRhDescriptorB)

寄存器	偏移量	R/W	描述	复位值
HcRhDescriptorB	USBH_BA+0x04C	R/W	主控制器根集线器描述符B寄存器	0x0000_0000

31	30	29	28	27	26	25	24
PPCM							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
PPCM							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved							

位	描述	
[31:16]	**PPCM**	**端口电源控制屏蔽** 电源切换模式PSM为1时，这里每一位决定一个端口是否受所有端口同时供电命令的影响，为1时各个端口受单独供电命令影响，为0时所有端口受同时供电命令影响。 0 = 所有端口受同时供电命令影响 1 = 各个端口受单独供电命令影响 **注:** PPCM[15:2] 及 PPCM[0] 是保留位.
[15:0]	**Reserved**	保留

主控制器根集线器状态寄存器(HcRhStatus)

寄存器	偏移量	R/W	描述	复位值
HcRhStatus	USBH_BA+0x050	R/W	主控制器根集线器状态寄存器	0x0000_0000

31	30	29	28	27	26	25	24
CRWE	**Reserved**						
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved						**OCIC**	**LPSC**
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
DRWE	**Reserved**						
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved						**OCI**	**LPS**

位	描述	
[31]	**CRWE**	**清除远程唤醒使能位** 此位用来清除DRWE (HcRhStatus[15]). 此位只能读到0 写操作: 0 = 没有影响 1 = 清除 DRWE (HcRhStatus[15]).
[31:18]	**Reserved**	保留
[17]	**OCIC**	**过电流显示改变** 当OCI (HcRhStatus[1])发生变化时此位由硬件置1，此位写1清0 0 = OCI (HcRhStatus[1]) 没有变化. 1 = OCI (HcRhStatus[1]) 改变了
[16]	**LPSC**	**打开所有端口供电** 当(PSM (HcRhDescriptorA[8]) = 0)时，此位用来使能所有端口供电 此位总是读为0 写操作: 0 = 没有影响 1 = 所有端口供电
[15]	**DRWE**	**设备远程唤醒使能位** 此位使能连接状态改变（HcRhPortStatus1[16]）作为远程唤醒事件 写操作： 0 = 没有影响. 1 = 使能连接状态改变作为远程唤醒事件 读操作: 0 = 禁止连接状态改变作为远程唤醒事件 1 = 使能连接状态改变作为远程唤醒事件

[14:2]	**Reserved**	保留
[1]	**OCI**	**过电流显示** 此位显示了过电流的状态，只有在NOCP (HcRhDesA[12]) 及 OCPM (HcRhDesA[11])为0时有效 0 = 没有过电流状况 1 = 出现过电流状况
[0]	**LPS**	**切断所有端口电源** 当电源模式为(PSM (HcRhDescriptorA[8]) = 0)时，此位写1将切断所有端口电源 此位只能读为0. 写操作: 0 = 没影响. 1 = 切断所有端口电源.

主控制器根集线器端口状态(HcRhPrt [1])

寄存器	偏移量	R/W	描述	复位值
HcRhPortStatus1	USBH_BA+0x058	R/W	主控制器根集线器端口状态[1]	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved			PRSC	OCIC	PSSC	PESC	CSC
15	14	13	12	11	10	9	8
Reserved						LSDA	PPS
7	6	5	4	3	2	1	0
Reserved			PRS	POCI	PSS	PES	CCS

位	描述	
[31:21]	**Reserved**	保留
[20]	**PRSC**	**端口复位状态改变** 此位显示端口复位信号完成 此位写1清0 0 = 端口复位信号没有完成 1 = 端口复位信号完成
[19]	**OCIC**	**端口过电流显示改变** 当 POCI (HcRhPortStatus1[3])改变时，此位置1 写1清0 0 = POCI (HcRhPortStatus1[3]) 没有改变 1 = POCI (HcRhPortStatus1[3]) 改变了
[18]	**PSSC**	**端口暂停状态改变** 此位表示端口有完整的唤醒时序 写1清0 0 = 端口唤醒没有发生 1 = 端口唤醒
[17]	**PESC**	**端口使能状态改变** 该位表示端口由于硬件事件而被禁止（PES (HcRhPortStatus1[1])=0）了 写1清0 0 = PES (HcRhPortStatus1[1]) 没有改变. 1 = PES (HcRhPortStatus1[1]) 改变了

[16]	**CSC**	**连接状态改变** 此位显示检测到连接或者断开事件发生 (CCS (HcRhPortStatus1[0]) 改变). 写1清0 0 = 没有连接/断开事件发生(CCS (HcRhPortStatus1[0]) 不变). 1 = 硬件检测到连接/断开事件发生(CCS (HcRhPortStatus1[0]) 改变).
[15:10]	**Reserved**	保留
[9]	**LSDA**	**低速设备插入（读）或切断端口电源（写）** 该位定义了插入设备的速度，只有在CCS (HcRhPortStatus1[0])为1是有效。此位也用来切断端口电源 写操作: 0 = 没有影响. 1 = 清除 PPS (HcRhPortStatus1[8]). 读操作: 0 = 全速设备 1 = 低速设备
[8]	**PPS**	**端口电源状态** 此位显示了端口的电源状态，不管电源切换模式 写操作: 0 = 没有影响. 1 = 端口电源使能. 读操作: 0 = 端口电源禁止. 1 = 端口电源使能.
[7:5]	**Reserved**	保留
[4]	**PRS**	**端口复位状态** 此位反映了端口的复位状态 写操作: 0 = 没有影响. 1 = 设置端口复位. 读操作： 0 = 端口复位信号没有使能. 1 = 端口复位信号使能.
[3]	**POCI**	**端口过电流显示（读）或清除暂停状态（写）** 此位反应了端口过电流的状态，只有在NOCP (HcRhDescriptorA[12]) 为 0 且OCPM (HcRhDescriptorA[11]) 为1时有效 该位也可用于启动端口原状态。 写操作: 0 = 没有影响 1 = 清除暂停状态. 读操作: 0 = 没有过电流状况 1 = 有过电流状况

[2]	**PSS**	**端口暂停状态** 此位显示端口是暂停的 写操作: 0 = 没有影响 1 = 设置端口暂停 读操作: 0 = 端口没有暂停 1 = 端口暂停
[1]	**PES**	**端口使能状态** 写操作： 0 = 没有影响 1 = 设置端口使能. 读操作: 0 = 端口禁止. 1 = 端口使能.
[0]	**CCS**	**当前连接状态（读）或清除端口使能位（写）** 写操作: 0 = 没有影响 1 = 清除端口使能. 读操作: 0 = 没有连接的设备. 1 = 设备连接了

主控制器PHY 控制寄存器(HcPhyControl)

寄存器	偏移量	R/W	描述	复位值
HcPhyControl	USBH_BA+0x200	R/W	主控制器PHY 控制寄存器	0x080F_0000

31	30	29	28	27	26	25	24
Reserved				**STBYEN**	**Reserved**		
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved							

位	描述	
[31:28]	**Reserved**	保留
[27]	**STBYEN**	**USB 收发器待机使能位** 此位控制USB收发器是否可以进入待机模式以减小功耗 0 = USB收发器永远不会进入待机模式 1 = 当端口在断电模式下（端口电源不使能），USB收发器将进入待机模式
[26:0]	**Reserved**	保留

主控制器多功能控制寄存器(HcMiscControl)

寄存器	偏移量	R/W	描述	复位值
HcMiscControl	USBH_BA+0x204	R/W	主控制器多功能控制寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							DPRT1
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved			PPCAL	OCAL	Reserved	ABORT	Reserved

位	描述	
[31:17]	Reserved	保留
[16]	DPRT1	**禁止端口 1** 此位控制连接USB主机与收发器之间的端口1是否使能，如果连接被禁止，USB主机将不会收到任何USB总线事件 设置此位为1，不管USB主机作何操作，端口1的收发器将强制进入待机模式 0 = USB主机和端口1的收发器间的连接使能 1 = USB主机和端口1的收发器间的连接禁止，收发器将强制进入待机模式
[15:5]	Reserved	保留
[4]	PPCAL	**端口电源控制** 此位设置端口外部电源IC的控制极性 0 = 端口电源控制高有效 1 = 端口电源控制低有效
[3]	OCAL	**过电流低有效** 此位控制外部电源IC过电流指示的极性 0 = 过电流标志高有效 1 = 过电流标志低有效
[2]	Reserved	保留
[1]	ABORT	**AHB 总线错误响应** 此位显示AHB总线收到错误响应 0 = 没有收到错误响应 1 = 收到了错误响应
[0]	Reserved	保留

6.34 USB 2.0 主控制器(USBH)

6.34.1 概述

这颗芯片配有USB 2.0 HS/FS 主机控制器 (USBH) ，支持增强型主机控制器接口(EHCI) 以及开放主机控制器接口(OpenHCI, OHCI) 协议,主机控制器寄存器级的操作以管理设备以及USB通讯。

USBH 内建一个根 Hub 以及一个 USB 端口, 实时在系统内存与USB总线间传输数据的 DMA，端口电源控制以及过流检测功能。

USBH 负责检测USB 设备的插入和拔出，管理数据传输，收集状态以及使能 USB 总线, 提供电源控制以及检测插入设备的过流的情况。

6.34.2 特性

- 支持USB规范2.0版本
- 支持EHCI规范 1.0版本
- 支持OpenHCI规范 1.0版本
- 支持高速 (480Mbps), 全速(12Mbps) 以及低速 (1.5Mbps) USB 设备.
- 支持控制, 批量, 中断, 等时以及拆分传输.
- 内建根 Hub.
- 支持端口路径逻辑将全速，低速设备接入OHCI 控制器.
- 支持 USB 主机端口与USB 设备端口共享 (OTG 功能).
- 支持端口电源控制以及端口过流检测
- 支持实时 DMA 数据传输.

6.34.3 框图

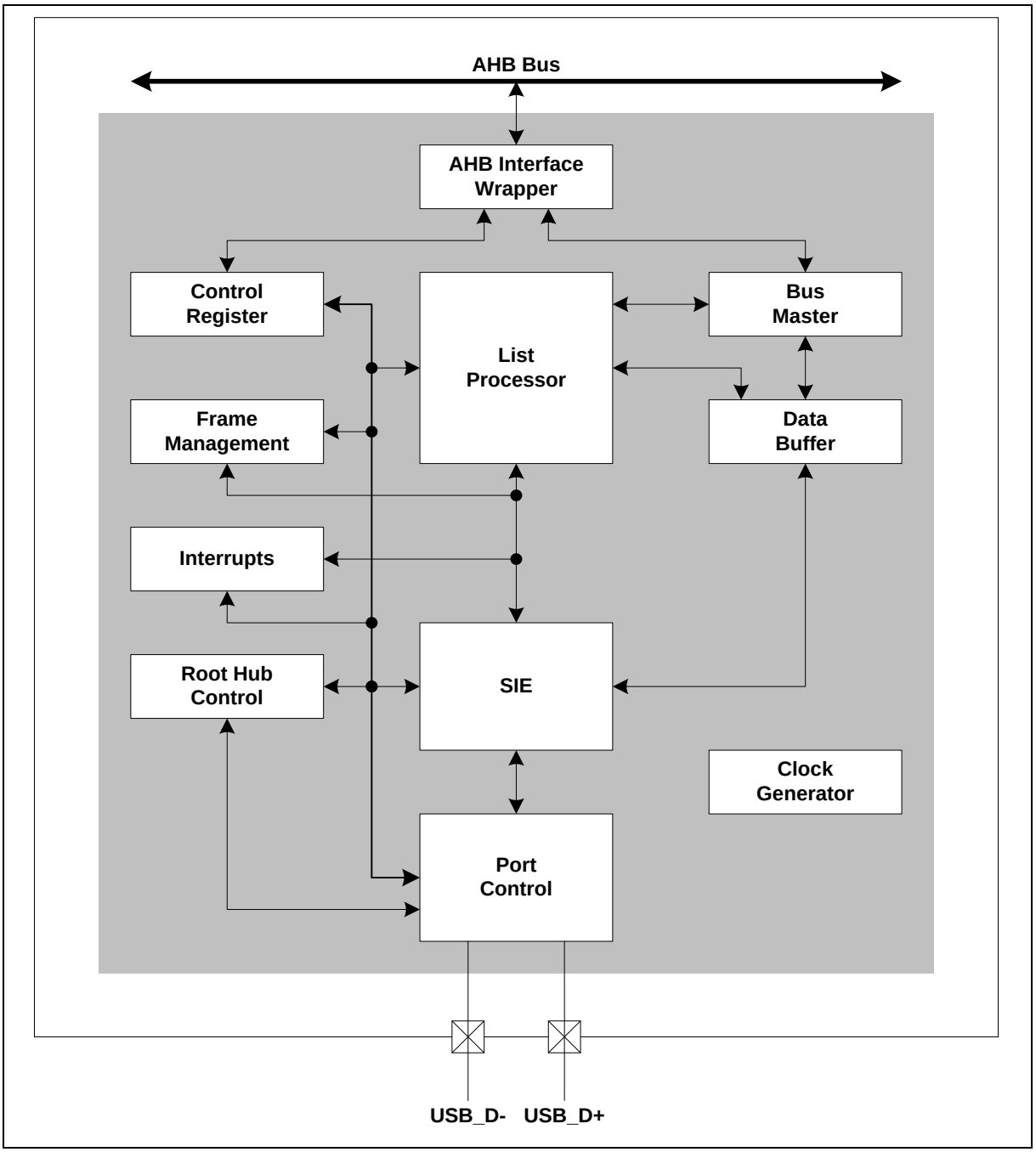


图 6.34-1 USB 2.0 FS 主机控制器框图

6.34.4 基本配置

6.34.4.1 USBH EHCI控制器基本配置

- 时钟源配置
  - 配置USBSEL (CLK_CLKSEL0[8])选择USBH时钟源
  - 配置USBDIV (CLK_CLKDIV0[7:4]).设置USBH除频
  - 配置HSUSBHCKEN (CLK_AHBCLK[16])使能 USBH时钟
- 复位配置
  - 通过UHC20RST (SYS_IPRST0[16]) 复位USBH 控制器

6.34.4.2 USBH OHCI控制器基本配置

- 时钟源配置
  - 配置USBSEL (CLK_CLKSEL0[8])选择USBH时钟源
  - 配置USBDIV (CLK_CLKDIV0[7:4]).设置USBH除频
  - 配置HSUSBHCKEN (CLK_AHBCLK[16])使能 USBH时钟
- 复位配置
  - 通过UHC20RST (SYS_IPRST0[16]) 复位USBH 控制器
- 管脚配置

组	管脚名称	GPIO	MFP
USB	USB_D+	PA.14	MFP14
	USB_D-	PA.13	MFP14
	USB_OTG_ID	PA.15	MFP14
	USB_VBUS	PA.12	MFP14
	USB_VBUS_EN	PB.6, PB.15	MFP14
	USB_VBUS_ST	PB.7, PC.14, PD.4	MFP14

6.34.5 功能描述

6.34.5.1 EHCI 控制器

EHCI 与系统通过AHB接口连接， CPU 通过AHB读写寄存器。CPU 作为总线管理者发起传输，EHCI做出响应。例如，如果 CPU 要向EHCI寄存器写数据, CPU提供地址和数据值。EHCI 通过该地址找到寄存器填入数据值。如果读寄存器，EHCI 通过该地址找到寄存器读到值并放在系统总线上。

同样的, 当 EHCI 要执行数据传输, 它就作为管理者发起数据传输，这时系统内存作为总线目标. EHCI, 作为管理者可以执行两种数据传输：从EHCI到系统内存,或从系统内存到EHCI.当 EHCI 想要将从USB2.0 设备来的数据存入系统内存，它就通过内存接口信号，配置EHCI 写控制字 (write), 数据以及要写入内存的数据个数，内存控制器收到数据将其存入内存。如果数据从内存传到设备， EHCI 执行向系统总线的读操作。内存控制器通过内存接口信号提供数据， EHCI 接收到数据然后将其传向设备

6.34.5.2 OHCI 控制器

AHB 接口

OpenHCI 主机控制器与系统通过AHB 总线连接。需要主从两种总线操作。作主，主机控制器响应 AHB 总线上的时钟周期通过EDs 和 TDs 在内存和本地数据缓存间传输数据。做从，主机控制器监控AHB 总线的时钟周期决定何时响应这些信号。通过AHB 总线从机接口非实时的访问主机控制器寄存器。

主控制器

主机控制器包含 5 个功能模块，包括列表处理器，帧管理，中断执行，主控制器总线管理器以及数据缓存。

列表处理器管理主机控制器驱动的数据结构以及主机控制器其他协同事项。

帧管理模块负责管理USB 协议以及 OpenHCI协议要求的帧相关事务，包括：

- 管理 OpenHCI 帧特殊操作寄存器
- 管理最大数据包计数.
- 执行SIE要求的USB 传输帧格式
- 产生SIE要求的 SOF token

中断是主机控制器驱动HC-initiated 通讯的一种通讯手段。有一些事件会触发主机控制器中断，每个特殊事件置位HcInterruptStatus寄存器的特定位.

主控制器总线管理是数据通道的核心单元，它联系各种访问到AHB总线上..。主控制有两个总线管理员：列表处理器以及数据缓存引擎。

数据缓存作为总线管理者与SIE之间的数据接口，它包含64个字节的固定基地址的双向异步FIFO，以及一个双字长AHB 保持寄存器

USB 接口

USB 接口包含内建的根以及一个 USB 端口, 端口 1和串行接口引擎 (SIE) 和USB 时钟发生器。接口执行主控制器需求的总线传输以及USB的hub以及端口管理。

SIE 负责管理所有的USB事物，控制总线协议，包的生成与解包，数据的并串转换，CRC码，位填充，以及不归零码的编码。所有的USB 事物由列表处理器以及帧管理单元提出。

根Hub 是独立控制的端口集， hub 管理所有端口的功能的控制与状态。

6.34.6 寄存器映射

R: 只读, **W**: 只写, **R/W**: 可读可写

寄存器	偏移量	R/W	描述	复位值
USBH 基地址: **USBH_BA = 0x4000_9000** **HSUSBH_BA = 0x4001_A000**				
HcRevision	USBH_BA+0x000	R	主机控制器版本寄存器	0x0000_0110
HcControl	USBH_BA+0x004	R/W	主机控制器控制寄存器	0x0000_0000
HcCommandStatus	USBH_BA+0x008	R/W	主机控制器指令状态寄存器	0x0000_0000
HcInterruptStatus	USBH_BA+0x00C	R/W	主机控制器中断状态寄存器	0x0000_0000
HcInterruptEnable	USBH_BA+0x010	R/W	主机控制器中断使能寄存器	0x0000_0000
HcInterruptDisable	USBH_BA+0x014	R/W	主机控制器中断禁止寄存器	0x0000_0000
HcHCCA	USBH_BA+0x018	R/W	主机控制器通讯范围寄存器	0x0000_0000
HcPeriodCurrentED	USBH_BA+0x01C	R/W	主机控制器周期传输目前 ED寄存器	0x0000_0000
HcControlHeadED	USBH_BA+0x020	R/W	主机控制器控制传输首个 ED寄存器	0x0000_0000
HcControlCurrentED	USBH_BA+0x024	R/W	主机控制器控制传输目前ED 寄存器	0x0000_0000
HcBulkHeadED	USBH_BA+0x028	R/W	主机控制器块传输首个 ED 寄存器	0x0000_0000
HcBulkCurrentED	USBH_BA+0x02C	R/W	主控制器块传输目前 ED 寄存器	0x0000_0000
HcDoneHead	USBH_BA+0x030	R/W	主控制器完成首个寄存器	0x0000_0000
HcFmInterval	USBH_BA+0x034	R/W	主机控制器帧间隔寄存器	0x0000_2EDF
HcFmRemaining	USBH_BA+0x038	R	主机控制器帧剩余寄存器	0x0000_0000
HcFmNumber	USBH_BA+0x03C	R	主机控制器帧号码寄存器	0x0000_0000
HcPeriodicStart	USBH_BA+0x040	R/W	主机控制器周期开始寄存器	0x0000_0000
HcLSThreshold	USBH_BA+0x044	R/W	主机控制器低速阈值寄存器	0x0000_0628
HcRhDescriptorA	USBH_BA+0x048	R/W	主机控制器根 Hub 描述符 A 寄存器	0x0000_0902
HcRhDescriptorB	USBH_BA+0x04C	R/W	主机控制器根 Hub 描述符 B 寄存器	0x0000_0000
HcRhStatus	USBH_BA+0x050	R/W	主机控制器根 Hub 状态寄存器	0x0000_0000
HcRhPortStatus1	USBH_BA+0x058	R/W	主机控制器根 Hub 端口状态寄存器1	0x0000_0000
HcPhyControl	USBH_BA+0x200	R/W	主机控制器 PHY 控制寄存器	0x0000_0000
HcMiscControl	USBH_BA+0x204	R/W	主机控制器杂项控制寄存器	0x0000_0000
EHCVNR	HSUSBH_BA+0x000	R	EHCI 版本号寄存器	0x0095_0020
EHCSPR	HSUSBH_BA+0x004	R	EHCI 结构参数寄存器	0x0000_0012

寄存器	偏移量	R/W	描述	复位值
EHCCPR	HSUSBH_BA+0x008	R	EHCI 性能参数寄存器	0x0000_0000
UCMDR	HSUSBH_BA+0x020	R/W	USB 指令寄存器	0x0008_0000
USTSR	HSUSBH_BA+0x024	R/W	USB 状态寄存器	0x0000_1000
UIENR	HSUSBH_BA+0x028	R/W	USB 中断使能寄存器	0x0000_0000
UFINDR	HSUSBH_BA+0x02C	R/W	USB 帧索引寄存器	0x0000_0000
UPFLBAR	HSUSBH_BA+0x034	R/W	USB 周期帧列表基地址寄存器	0x0000_0000
UCALAR	HSUSBH_BA+0x038	R/W	USB目前异步列表地址寄存器	0x0000_0000
UASSTR	HSUSBH_BA+0x03C	R/W	USB 异步计划休眠定时器寄存器	0x0000_0BD6
UCFGR	HSUSBH_BA+0x060	R/W	USB 配置标志寄存器	0x0000_0000
UPSCR0	HSUSBH_BA+0x064	R/W	USB 端口 0 状态和控制寄存器	0x0000_2000
UPSCR1	HSUSBH_BA+0x068	R/W	USB 端口 1 状态和控制寄存器	0x0000_2000
USBPCR0	HSUSBH_BA+0x0C4	R/W	USB PHY 0 控制寄存器	0x0000_0060
USBPCR1	HSUSBH_BA+0x0C8	R/W	USB PHY 1 控制寄存器	0x0000_0020

6.34.7 寄存器描述

HcRevision 主机控制器版本寄存器

寄存器	偏移量	R/W	描述	复位值
HcRevision	USBH_BA+0x000	R	主机控制器版本寄存器	0x0000_0110

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
REV							

位	描述	
[31:8]	Reserved	保留.
[7:0]	REV	**版本号** 标明硬件的 Open HCI 协议版本号. 主控制器支持 1.1 标准. (X.Y = XYh).

HcControl 主控制器控制寄存器

寄存器	偏移量	R/W	描述	复位值
HcControl	USBH_BA+0x004	R/W	主控制器控制寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
HCFS		BLE	CLE	IE	PLE	CBSR	

位	描述	
[31:8]	Reserved	保留.
[7:6]	HCFS	主控制器功能状态 这里设定主机控制器状态. 控制器当检测到下游端口发来的恢复信号后会从USBSUSPEND状态变为 USBRESUME 状态。状态包括： 00 = USB 挂起. 01 = USB 操作. 10 = USB 恢复. 11 = USB 复位.
[5]	BLE	批量列表使能位 0 = 禁止在下一个SOF ((帧起始)处理批量列表 1 = 在下一帧处理批量列表.
[4]	CLE	控制列表使能位 0 = 禁止在下一个SOF ((帧起始)处理控制列表 1 = 在下一帧处理控制列表.
[3]	IE	同步列表使能位 ISOEn 和 PLE (HcControl[2]) 都为高的时候使能控制器处理同步列表。ISOEn或PLE (HcControl[2]) 有一个为低禁止主机控制器处理同步列表 0 = 禁止在下一个SOF (Start-Of-Frame)处理同步列表 1 = 如果PLE (HcControl[2]) 也为高，在下一帧处理同步列表.
[2]	PLE	周期列表使能位 使能处理周期（中断和同步）列表。主控制器在处理周期传输帧前确认此位。 0 = 禁止在下一个SOF (帧起始)处理中断和同步列表 1 = 在下一帧处理中断和同步列表.. **注意:**要处理同步列表, PLE 和 IE (HcControl[3]) 都要为高.

[1:0]	**CBSR**	**控制与批量服务比例** 控制端点与批量端点的服务比例。在处理非周期列表之前，控制器根据有多少非空的控制端点与批量端点的的比例，决定是继续服务其他的控制端点，还是切换到批量端点。帧切换的时候内部计数器会保持。当复位的时候HCD 负责恢复这个值 值. 00 = 控制端点与批量端点服务比例是 1:1. 01 = 控制端点与批量端点服务比例是 2:1. 10 = 控制端点与批量端点服务比例是 3:1. 11 = 控制端点与批量端点服务比例是 4:1.

HcCommandStatus 主控制器指令状态寄存器

寄存器	偏移量	R/W	描述	复位值
HcCommandStatus	USBH_BA+0x008	R/W	主控制器指令状态寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved						SOC	
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved					BLF	CLF	HCR

位	描述	
[31:18]	**Reserved**	保留.
[17:16]	**SOC**	**流程重写计数** 每次流程重写错误时加1，初始化是00b 在11b回绕。即使SO (HcInterruptStatus[0])已经置1，当检测到流程重写错误时也会加1.
[15:3]	**Reserved**	保留.
[2]	**BLF**	**填入块列表** 为1指示批量列表里有活动的 TD。此位可以由软件或主机控制器来置1，在主控制器每次处理批量列表头之后清0. 0 = 在批量列表里没有发现活动的 TD或主机控制器处理到批量列表头. 1 = 块列表加入了活动的 TD.
[1]	**CLF**	**填入控制列表** 为1指示控制列表里有活动的 TD。此位可以由软件或主机控制器来置1，在主控制器每次处理到控制列表头之后清0. 0 = 在控制列表里没有发现活动的 TD或主机控制器处理到控制列表头. 1 = 控制列表加入了活动的 TD.
[0]	**HCR**	**主机控制器复位** 这里设置主机控制器软复位，完成复位操作后此位由主机控制器清除. 此位置1后不会复位根Hub ，也不会向下游设备发送复位信号. 0 = 主机控制器不在软复位状态. 1 = 主机控制器在软复位状态.

HcInterruptStatus 主机控制器中断状态寄存器

寄存器	偏移量	R/W	描述	复位值
HcInterruptStatus	USBH_BA+0x00C	R/W	主机控制器中断状态寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved	**RHSC**	**FNO**	**Reserved**	**RD**	**SF**	**WDH**	**SO**

位	描述	
[31:7]	**Reserved**	保留.
[6]	**RHSC**	**根 Hub 状态改变** 这一位当HcRhStatus的内容或HcRhPortStatus1内容改变时置位 0 = HcRhStatus 与 HcRhPortStatus1寄存器的值没有改变. 1 = HcRhStatus 与 HcRhPortStatus1寄存器的值改变了.
[5]	**FNO**	**帧编号溢出** 这一位当帧编号的bit 15 由 1 变 0 或由 0 变 1 时置位. 0 = 帧编号的bit 15没变 1 = 帧编号的bit 15 由 1 变 0 或由 0 变 1.
[4]	**Reserved**	保留.
[3]	**RD**	**恢复信号检测** 当从下游端口检测到恢复信号时置位. 0 = 没有从下游端口检测到恢复信号. 1 = 从下游端口检测到恢复信号.
[2]	**SF**	**帧起始** 当帧管理功能单元产生帧起始事件时置位，主机控制器同时产生一个 SOF token. 0 = .没有帧起始事件. 1 = .产生帧起始事件，主机控制器同时产生一个 SOF token
[1]	**WDH**	**回写完成头** 当主控制器将HcDoneHead 写入 HccaDoneHead 时置位，之后在此位清除之前 HccaDoneHead 不会更新 0 = .主机控制器没有更新 HccaDoneHead. 1 = .主控制器将HcDoneHead 写入 HccaDoneHead.

[0]	**SO**	**计划表超期** 当列表处理器确定发生计划表超期时置位 0 = 计划表超期没有发生. 1 = 发生计划表超期.

HcInterruptEnable 主机控制器中断使能寄存器

寄存器	偏移量	R/W	描述	复位值
HcInterruptEnable	USBH_BA+0x010	R/W	主机控制器中断使能寄存器	0x0000_0000

31	30	29	28	27	26	25	24
MIE	Reserved						
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved	RHSC	FNO	Reserved	RD	SF	WDH	SO

位	描述	
[31]	**MIE**	**管理者中断使能位** 这一位是整体中断使能，写1允许其他位使能后产生中断. 写操作: 0 = 没影响. 1 = 使能当 RHSC (HcInterruptStatus[6]), FNO (HcInterruptStatus[5]), RD (HcInterruptStatus[3]), SF (HcInterruptStatus[2]), WDH (HcInterruptStatus[1])或 SO (HcInterruptStatus[0]) 使能如果相应的 HcInterruptEnable的对应位为高时产生中断. 读操作: 0 = 当RHSC (HcInterruptStatus[6]), FNO (HcInterruptStatus[5]), RD (HcInterruptStatus[3]), SF (HcInterruptStatus[2]), WDH (HcInterruptStatus[1])或 SO (HcInterruptStatus[0]) 使能如果相应的 HcInterruptEnable的对应位为高时不会产生中断.. 1 = 当 RHSC (HcInterruptStatus[6]), FNO (HcInterruptStatus[5]), RD (HcInterruptStatus[3]), SF (HcInterruptStatus[2]), WDH (HcInterruptStatus[1])或 SO (HcInterruptStatus[0]) 使能如果相应的 HcInterruptEnable的对应位为高时产生中断..
[30:7]	**Reserved**	保留.
[6]	**RHSC**	**根 Hub 状态改变使能位** 写操作: 0 = 没影响. 1 = 使能 RHSC (HcInterruptStatus[6]) 引起的中断. 读操作: 0 = RHSC (HcInterruptStatus[6]) 引起的中断已禁止. 1 = RHSC (HcInterruptStatus[6]) 引起的中断已使能.

[5]	**FNO**	**帧编号溢出使能位** 写操作: 0 = 没影响. 1 = 使能 FNO (HcInterruptStatus[5])引起的中断. 读操作: 0 = FNO (HcInterruptStatus[5])引起的中断已禁止. 1 = FNO (HcInterruptStatus[5])引起的中断已使能.
[4]	**Reserved**	保留.
[3]	**RD**	**Resume 检测使能位** 写操作: 0 = 没影响. 1 = 使能由 RD (HcInterruptStatus[3]) 引起的中断. 读操作: 0 = 由 RD (HcInterruptStatus[3]) 引起的中断已禁止. 1 = 由 RD (HcInterruptStatus[3]) 引起的中断已使能.
[2]	**SF**	**帧起始使能位** 写操作: 0 = 没影响. 1 = 使能由 SF (HcInterruptStatus[2])引起的中断. 读操作: 0 = 由 SF (HcInterruptStatus[2])引起的中断已禁止. 1 = 由 SF (HcInterruptStatus[2])引起的中断已使能
[1]	**WDH**	**回写完成头使能位** 写操作: 0 = 没影响. 1 = 使能由 WDH (HcInterruptStatus[1]) 引起的中断. 读操作: 0 = 由 WDH (HcInterruptStatus[1]) 引起的中断已禁止. 1 = 由 WDH (HcInterruptStatus[1]) 引起的中断已使能.
[0]	**SO**	**计划表超期使能位** 写操作: 0 = 没影响. 1 = 使能由 SO (HcInterruptStatus[0]) 引起的中断. 读操作: 0 = 由 SO (HcInterruptStatus[0]) 引起的中断已禁止. 1 = 由 SO (HcInterruptStatus[0]) 引起的中断已使能.

HcInterruptDisable 主机控制器中断禁止寄存器

寄存器	偏移量	R/W	描述	复位值
HcInterruptDisable	USBH_BA+0x014	R/W	主机控制器中断禁止寄存器	0x0000_0000

31	30	29	28	27	26	25	24
MIE	Reserved						
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved	RHSC	FNO	Reserved	RD	SF	WDH	SO

位	描述	
[31]	**MIE**	**管理者中断禁止位** 总的中断禁止，写1禁用所有中断 写操作: 0 = 没影响. 1 = 禁止当 RHSC (HcInterruptStatus[6]), FNO (HcInterruptStatus[5]), RD (HcInterruptStatus[3]), SF (HcInterruptStatus[2]), WDH (HcInterruptStatus[1])或 SO (HcInterruptStatus[0]) 使能如果相应的 HcInterruptEnable的对应位为高时产生中断. 读操作: 0 = 当 RHSC (HcInterruptStatus[6]), FNO (HcInterruptStatus[5]), RD (HcInterruptStatus[3]), SF (HcInterruptStatus[2]), WDH (HcInterruptStatus[1])或 SO (HcInterruptStatus[0]) 使能如果相应的 HcInterruptEnable的对应位为高时不会产生中断.. 1 = 当 RHSC (HcInterruptStatus[6]), FNO (HcInterruptStatus[5]), RD (HcInterruptStatus[3]), SF (HcInterruptStatus[2]), WDH (HcInterruptStatus[1])或 SO (HcInterruptStatus[0]) 使能如果相应的 HcInterruptEnable的对应位为高时产生中断..
[30:7]	**Reserved**	保留.
[6]	**RHSC**	**根 Hub 状态改变禁止位** 写操作: 0 = 没影响. 1 = 禁止 RHSC (HcInterruptStatus[6]) 引起的中断. 读操作: 0 = RHSC (HcInterruptStatus[6]) 引起的中断已禁止. 1 = RHSC (HcInterruptStatus[6]) 引起的中断已使能

[5]	**FNO**	**帧编号溢出禁止位** 写操作: 0 = 没影响. 1 = 禁止 FNO (HcInterruptStatus[5])引起的中断. 读操作: 0 = FNO (HcInterruptStatus[5])引起的中断已禁止. 1 = FNO (HcInterruptStatus[5])引起的中断已使能.
[4]	**Reserved**	保留.
[3]	**RD**	**Resume 检测禁止位** 写操作: 0 = 没影响. 1 = 禁止由 RD (HcInterruptStatus[3]) 引起的中断. 读操作: 0 = 由 RD (HcInterruptStatus[3]) 引起的中断已禁止. 1 = 由 RD (HcInterruptStatus[3]) 引起的中断已使能.
[2]	**SF**	**帧起始禁止位** 写操作: 0 = 没影响. 1 = 禁止由 SF (HcInterruptStatus[2])引起的中断. 读操作: 0 = 由 SF (HcInterruptStatus[2])引起的中断已禁止. 1 = 由 SF (HcInterruptStatus[2])引起的中断已使能
[1]	**WDH**	**回写完成头禁止位** 写操作: 0 = 没影响. 1 = 禁止由 WDH (HcInterruptStatus[1]) 引起的中断. 读操作: 0 = 由 WDH (HcInterruptStatus[1]) 引起的中断已禁止. 1 = 由 WDH (HcInterruptStatus[1]) 引起的中断已使能.
[0]	**SO**	**计划表超期禁止位** 写操作: 0 = 没影响. 1 = 禁止由 SO (HcInterruptStatus[0]) 引起的中断. 读操作: 0 = 由 SO (HcInterruptStatus[0]) 引起的中断已禁止. 1 = 由 SO (HcInterruptStatus[0]) 引起的中断已使能.

HcHCCA 主机控制器通讯范围寄存器

寄存器	偏移量	R/W	描述	复位值
HcHCCA	USBH_BA+0x018	R/W	主机控制器通讯范围寄存器	0x0000_0000

31	30	29	28	27	26	25	24
HCCA							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
HCCA							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
HCCA							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved							

位	描述	
[31:8]	**HCCA**	**主机控制器通讯范围** 指向主机控制器通讯范围 (HCCA)的基地址.
[7:0]	**Reserved**	保留.

HcPeriodCurrentED 主机控制器周期目前 ED 寄存器

寄存器	偏移量	R/W	描述	复位值
HcPeriodCurrentED	USBH_BA+0x01C	R/W	主机控制器周期目前 ED 寄存器	0x0000_0000

31	30	29	28	27	26	25	24
PCED							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
PCED							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
PCED							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
PCED				**Reserved**			

位	描述	
[31:4]	**PCED**	**周期目前 ED** 指向目前同步或中断端点描述符的物理地址.
[3:0]	**Reserved**	保留.

HcControlHeadED 主机控制器控制头 ED 寄存器

寄存器	偏移量	R/W	描述	复位值
HcControlHeadED	USBH_BA+0x020	R/W	主机控制器控制头 ED 寄存器	0x0000_0000

31	30	29	28	27	26	25	24
CHED							
23	22	21	20	19	18	17	16
CHED							
15	14	13	12	11	10	9	8
CHED							
7	6	5	4	3	2	1	0
CHED				Reserved			

位	描述	
[31:4]	CHED	**控制头 ED** 指向控制列表的第一个端点描述符的物理地址
[3:0]	Reserved	保留.

HcControlCurrentED 主机控制器控制目前 ED 寄存器

寄存器	偏移量	R/W	描述	复位值
HcControlCurrentED	USBH_BA+0x024	R/W	主机控制器控制目前 ED 寄存器	0x0000_0000

31	30	29	28	27	26	25	24
CCED							
23	22	21	20	19	18	17	16
CCED							
15	14	13	12	11	10	9	8
CCED							
7	6	5	4	3	2	1	0
CCED				Reserved			

位	描述	
[31:4]	CCED	**控制目前头 ED** 指向控制列表里目前端点描述符的物理地址
[3:0]	Reserved	保留.

HcBulkHeadED 主机控制器块头 ED 寄存器

寄存器	偏移量	R/W	描述	复位值
HcBulkHeadED	USBH_BA+0x028	R/W	主机控制器块头 ED 寄存器	0x0000_0000

31	30	29	28	27	26	25	24
BHED							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
BHED							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
BHED							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
BHED				**Reserved**			

位	描述	
[31:4]	**BHED**	**块头 ED** 指向块列表第一个端点描述符的物理地址.
[3:0]	**Reserved**	保留.

HcBulkCurrentED 主机控制器块目前 ED 寄存器

寄存器	偏移量	R/W	描述	复位值
HcBulkCurrentED	USBH_BA+0x02C	R/W	主机控制器块目前 ED寄存器	0x0000_0000

31	30	29	28	27	26	25	24
BCED							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
BCED							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
BCED							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
BCED				**Reserved**			

位	描述	
[31:4]	**BCED**	**块目前 ED** 指向块列表目前端点描述符的物理地址
[3:0]	**Reserved**	保留.

HcDoneHead 主机控制器完成头寄存器

寄存器	偏移量	R/W	描述	复位值
HcDoneHead	USBH_BA+0x030	R/W	主机控制器完成头寄存器	0x0000_0000

31	30	29	28	27	26	25	24
DH							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
DH							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
DH							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
DH				**Reserved**			

位	描述	
[31:4]	**DH**	**完成头** 指向刚刚传输完成加入完成队列的描述符的物理地址.
[3:0]	**Reserved**	保留.

HcFmInterval 主机控制器帧间隔寄存器

寄存器	偏移量	R/W	描述	复位值
HcFmInterval	USBH_BA+0x034	R/W	主机控制器帧间隔寄存器	0x0000_2EDF

31	30	29	28	27	26	25	24
FIT	**FSMPS**						
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
FSMPS							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved		**FI**					
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
FI							

位	描述	
[31]	**FIT**	**帧间隔交替** 当载入新的值到FI (HcFmInterval[13:0])，主机控制器驱动会反转此位 0 = 主机控制器没有载入新的值到 FI (HcFmInterval[13:0]). 1 = 主机控制器载入新的值到FI (HcFmInterval[13:0]).
[30:16]	**FSMPS**	**FS 最大数据包** 这里的值在每一帧的开始载入最大数据包计数器.
[15:14]	**Reserved**	保留.
[13:0]	**FI**	**帧间隔** 这里设定帧时间长度 (bit times - 1). 例如一个帧 12,000 bit times，这里的值是11,999.

HcFmRemaining 主机控制器帧剩余寄存器

寄存器	偏移量	R/W	描述	复位值
HcFmRemaining	USBH_BA+0x038	R	主机控制器帧剩余寄存器	0x0000_0000

31	30	29	28	27	26	25	24
FRT	Reserved						
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved		FR					
7	6	5	4	3	2	1	0
FR							

位	描述	
[31]	FRT	**帧剩余反转** 这一位当FR (HcFmRemaining[13:0])到0之后从FIT (HcFmInterval[31]) 载入.
[30:14]	Reserved	保留.
[13:0]	FR	**帧剩余** 当主机控制器在 USBOPERATIONAL 状态，这个14位递减的计数器在每个12MHz的时钟周期递减。当计数器的值减到0（帧结束），计数器载入帧间隔，另外，计数器也在主机控制器进入USBOPERATIONAL.状态时载入

HcFmNumber 主机控制器帧编号寄存器

寄存器	偏移量	R/W	描述	复位值
HcFmNumber	USBH_BA+0x03C	R	主机控制器帧编号寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
FN							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
FN							

位	描述	
[31:16]	**Reserved**	保留
[15:0]	**FN**	**帧编号** 16位上数计数器在每次重载FR (HcFmRemaining[13:0])时加1，溢出时由‘FFFFh’变为‘0h.’

HcPeriodicStart 主机控制器周期开始寄存器

寄存器	偏移量	R/W	描述	复位值
HcPeriodicStart	USBH_BA+0x040	R/W	主机控制器周期开始寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved		PS					
7	6	5	4	3	2	1	0
PS							

位	描述	
[31:14]	Reserved	保留
[13:0]	PS	周期开始 这里的值被列表处理器用来决定周期列表处理需要在帧的哪里开始.

HcLSThreshold 主控制器低速阈值寄存器

寄存器	偏移量	R/W	描述	复位值
HcLSThreshold	USBH_BA+0x044	R/W	主控制器低速阈值寄存器	0x0000_0628

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved				LST			
7	6	5	4	3	2	1	0
LST							

位	描述	
[31:12]	Reserved	保留.
[11:0]	LST	低速阈值 在建立低速传输前比较这里与FR (HcFmRemaining[13:0]) 的值。传输只会在 FR (HcFmRemaining[13:0]) >= 这里的值的时候开始. 这个值由主机控制器驱动考虑传输条件设定并预先初始化。.

HcRhDescriptorA 主机控制器根 Hub 描述符 A 寄存器

寄存器	偏移量	R/W	描述	复位值
HcRhDescriptorA	USBH_BA+0x048	R/W	主机控制器根 Hub 描述符 A 寄存器	0x0000_0902

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved			NOCP	OCPM	Reserved		PSM
7	6	5	4	3	2	1	0
NDP							

位	描述	
[31:13]	**Reserved**	保留.
[12]	**NOCP**	**没有过流保护** 这一位描述根Hub端口报告的过流状态 0 = 报告过流状态. 1 = 没有报告过流状态.
[11]	**OCPM**	**过流保护模式** 这一位描述根Hub端口报告的过流状态，这一位只在NOCP (HcRhDescriptorA[12]) 为0时有效. 0 = 整体过流. 1 = 单独过流.
[10:9]	**Reserved**	保留.
[8]	**PSM**	**电源切换模式** 这一位决定根Hub端口如何控制电源切换 0 = 整体切换 1 = 单独切换
[7:0]	**NDP**	**下游端口数量** USB 主机控制器支持两个下游端口，本系列芯片支持1个

HcRhDescriptorB 主机控制器根 Hub 描述符 B 寄存器

寄存器	偏移量	R/W	描述	复位值
HcRhDescriptorB	USBH_BA+0x04C	R/W	主机控制器根 Hub 描述符 B 寄存器	0x0000_0000

31	30	29	28	27	26	25	24
PPCM							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
PPCM							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved							

位	描述	
[31:16]	**PPCM**	**端口电源控制屏蔽** 这里仅在PowerSwitchingMode 为1 (单独端口切换)时有效。当为1，端口只响应独立端口电源切换指令 (Set/ClearPortPower)。当为0，端口只响应整体电源切换指令(Set/ClearGlobalPower). 0 = 端口电源由整体电源切换控制. 1 = 端口电源由单独电源切换控制 **注意:** PPCM[15:2] 以及 PPCM[0] 保留
[15:0]	**Reserved**	保留.

HcRhStatus 主机控制器根 Hub 状态寄存器

寄存器	偏移量	R/W	描述	复位值
HcRhStatus	USBH_BA+0x050	R/W	主机控制器根Hub状态寄存器	0x0000_0000

31	30	29	28	27	26	25	24
CRWE	Reserved						
23	22	21	20	19	18	17	16
Reserved						OCIC	LPSC
15	14	13	12	11	10	9	8
DRWE	Reserved						
7	6	5	4	3	2	1	0
Reserved						OCI	LPS

位	描述	
[31]	CRWE	**清除远端唤醒使能位** 这一位用来清除 DRWE (HcRhStatus[15]). 这一位永远读到0. 写操作: 0 = 没影响. 1 = 清除 DRWE (HcRhStatus[15]).
[31:18]	Reserved	保留.
[17]	OCIC	**过流指示改变** 这一位由硬件当OCI (HcRhStatus[1])改变时设置 此位写1清0. 0 = OCI (HcRhStatus[1]) 没有改变. 1 = OCI (HcRhStatus[1]) 改变了.
[16]	LPSC	**设置整体电源** 在整体电源模式 (PSM (HcRhDescriptorA[8]) = 0),这一位写1使能所有端口的电源 这一位总是读为0. 写操作: 0 = 没影响. 1 = 设置整体电源.

[15]	**DRWE**	**设备远程唤醒使能位** 这一位控制在远端唤醒事件时端口连接状态是否改变 写操作: 0 = 没影响. 1 = 使能在远端唤醒事件时端口连接状态改变. 读操作: 0 = 在远端唤醒事件时端口连接状态改变已禁止. 1 = 在远端唤醒事件时端口连接状态改变已使能.
[14:2]	**Reserved**	保留.
[1]	**OCI**	**过流指示** 这一位反映过流状态管脚的状态，仅在NOCP (HcRhDesA[12]) 和 OCPM (HcRhDesA[11]) 为0时有效 0 = 没有过流状态. 1 = 过流状态.
[0]	**LPS**	**清除整体电源** 在整体电源模式 (PSM (HcRhDescriptorA[8]) = 0), 这一位写1清除所有端口的电源 这一位读总是为0. 写操作: 0 = 没影响. 1 = 清除整体电源

HcRhPrt [1] 主机控制器根 Hub 端口状态

寄存器	偏移量	R/W	描述	复位值
HcRhPortStatus1	USBH_BA+0x058	R/W	主机控制器根 Hub 端口状态 [1]	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved			PRSC	OCIC	PSSC	PESC	CSC
15	14	13	12	11	10	9	8
Reserved						LSDA	PPS
7	6	5	4	3	2	1	0
Reserved			PRS	POCI	PSS	PES	CCS

位	描述	
[31:21]	**Reserved**	保留.
[20]	**PRSC**	**端口复位状态改变** 这一位指示端口复位信号已经完成. 此位写1清0. 0 = 端口复位没有完成. 1 = 端口复位已完成.
[19]	**OCIC**	**端口过流指示改变** T这一位当 POCI (HcRhPortStatus1[3])改变时置1. 此位写1清0. 0 = POCI (HcRhPortStatus1[3]) 没有改变. 1 = POCI (HcRhPortStatus1[3]) 改变了.
[18]	**PSSC**	**端口挂起状态改变** 这一位指示端口的选择恢复流程完成 此位写1清0 0 = 端口恢复没有完成. 1 = 端口恢复已完成.
[17]	**PESC**	**端口使能状态改变** 这一位指示因为硬件事件端口被禁止了(PES (HcRhPortStatus1[1])为0). 此位写1清0. 0 = PES (HcRhPortStatus1[1]) 没有改变. 1 = PES (HcRhPortStatus1[1]) 已改变.

[16]	**CSC**	**连接状态改变** 这一位指示检测到连接或断开事件(CCS (HcRhPortStatus1[0]) 改变). 此位写1清0. 0 = 没有连接或断开事件t (CCS (HcRhPortStatus1[0]) 没有改变). 1 = 硬件检测到连接或断开事件 (CCS (HcRhPortStatus1[0]) 改变了).
[15:10]	**Reserved**	保留.
[9]	**LSDA**	**低速设备 (读) 或清除端点电源 (写)** 这一位决定插入设备的速度 (和 bud空闲) 。这一位仅在 CCS (HcRhPortStatus1[0]) 为1时有效. 这一位也用来清除端口电源 写操作: 0 = 没影响. 1 = 清除 PPS (HcRhPortStatus1[8]). 读操作: 0 = 全速设备. 1 = 低速设备.
[8]	**PPS**	**端口电源状态** 这一位反映端口电源状态，不管电源切换模式 写操作: 0 = 没影响. 1 = 端口电源使能. 读操作: 0 = 端口电源禁止. 1 = 端口电源使能.
[7:5]	**Reserved**	保留.
[4]	**PRS**	**端口复位状态** 这一位反映端口的复位状态. 写操作: 0 = 没影响. 1 = 设置端口复位. 读操作 0 = 端口复位信号无效. 1 = 端口复位信号有效.
[3]	**POCI**	**端口过流指示 (读) 或清除端口挂起 (写)** 这一位反映端口过流状态管脚的状态，这里仅当NOCP (HcRhDescriptorA[12]) 为0且OCPM (HcRhDescriptorA[11]) 为1时有效. 这一位也用来建立端口的选择结果流程 写操作: 0 = 没影响. 1 = 清除端口挂起. 读操作: 0 = 没有过流状态. 1 = 有过流状态.

[2]	**PSS**	**端口挂起状态** 此位指示端口已挂起 写操作: 0 = 没影响. 1 = 设置端口挂起. 都操作: 0 = 端口没有挂起. 1 = 端口选择挂起.
[1]	**PES**	**端口使能状态** 写操作: 0 = 没影响. 1 = 设置端口使能. 读操作: 0 = 端口禁止 1 = 端口使能.
[0]	**CCS**	**目前连接状态（读)或清除端口使能位(写)** 写操作: 0 = 没影响. 1 = 清除端口使能. 读操作: 0 = 没有设备连接. 1 = 设备已连接.

HcPhyControl 主机控制器 PHY 控制寄存器

寄存器	偏移量	R/W	描述	复位值
HcPhyControl	USBH_BA+0x200	R/W	主机控制器 PHY 控制寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved				**STBYEN**	**Reserved**		
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved							

位	描述	
[31:28]	**Reserved**	保留.
[27]	**STBYEN**	**USB 收发器待命使能位** 这意味控制USB 收发器是否进入待命模式以降低电源功耗 0 = USB 收发器不会进入待命模式. 1 = USB 收发器在端口掉电状态（端口电源无效）会进入待命模式
[26:0]	**Reserved**	保留.

HcMiscControl 主机控制器杂项控制寄存器

寄存器	偏移量	R/W	描述	复位值
HcMiscControl	USBH_BA+0x204	R/W	主机控制器杂项控制寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							DPRT1
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved				OCAL	Reserved	ABORT	Reserved

位	描述	
[31:17]	Reserved	保留.
[16]	DPRT1	**禁止端口 1** 这一位控制是否禁止端口1的USB 主机控制器与收发器之间的连接。如果连接被禁止，USB 主机控制器不会确认USB总线事件. 此位置1，不管USB主控制如何工作，端口1的收发器也会强制进入待命模式. 0 = 端口1的USB 主机控制器与收发器之间的连接使能. 1 = 端口1的USB 主机控制器与收发器之间的连接禁止，端口1的收发器也会强制进入待命模式.
[15:4]	Reserved	保留.
[3]	OCAL	**过流低有效** 这里设定外部电源IC过流标识的极性. 0 = 过流标志高有效. 1 = 过流标志低有效.
[2]	Reserved	保留.
[1]	ABORT	**AHB 总线错误响应** 这一位指示AHB 总线上收到一个错误响应 0 = 没有收到错误响应. 1 = 收到错误响应.
[0]	Reserved	保留.

EHCVNR EHCI 版本号寄存器

寄存器	偏移量	R/W	描述	复位值
EHCVNR	HSUSBH_BA+0x000	R	EHCI 版本号寄存器	0x0095_0020

31	30	29	28	27	26	25	24
VERSION							
23	22	21	20	19	18	17	16
VERSION							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
CRLEN							

位	描述	
[31:16]	VERSION	**主控制器接口版本号** 这里两个字节组合的BCD码是主机控制器支持的 EHCI 版本号。高字节是主版本，低字节是小版本。.
[15:8]	Reserved	保留.
[7:0]	CRLEN	**性能寄存器长度** 这个寄存器的值作为一个偏移量加到寄存器基地址得到操作寄存器空间的开始地址.

EHCSPR EHCI 结构参数寄存器

寄存器	偏移量	R/W	描述	复位值
EHCSPR	HSUSBH_BA+0x004	R	EHCI 结构参数寄存器	0x0000_0012

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
N_CC				N_PCC			
7	6	5	4	3	2	1	0
Reserved			PPC	N_PORTS			

位	描述	
[31:16]	Reserved	保留.
[15:12]	N_CC	**配套控制器个数** 这里指定 USB 2.0主控制器的配套控制器的个数 这里为0指示没有配有配套的主机控制器。不支持端口所有权变更，主机控制器根端口只支持高速设备. 大于0的值表示有配套的 USB 1.1 主机控制器(s). 支持端口所有权变更。主控制器根端口支持高速，全速和低速设备。
[11:8]	N_PCC	**每个配套处理器的端口数** 这里设定每个配套处理器支持的端口数，用来决定系统程序端口出线配置 例如, 如果 N_PORTS值是6 而 N_CC 的值是 2 ，那么 N_PCC的值可以是3。约定是前3个 N_PCC 端口被设置连接配套处理器1，接下来的N_PCC 端口连接配套处理器2，在前一个例子中， N_PCC 可以是 4, 前 4 个连接到配套处理器1，后2个链接到配套处理器2. 这里的值必须要与 N_PORTS 以及 N_CC一致.
[7:5]	Reserved	保留.
[4]	PPC	**端口电源控制** 这里决定是否主控制器贯彻端口电源控制。这里是1表明端口有端口电源切换，是0表示端口没有端口电源切换。这里的值影响到每个端口状态和控制寄存器的功能.
[3:0]	N_PORTS	**物理下游端口数量** 这是指定这个主机控制器实现的物理下行端口数。这里的值决定操作寄存器空间有多少个端口寄存器地址(见表 2-8). 有效值范围是 1H 到 FH. 这里填0未定义.

EHCCPR EHCI 性能参数寄存器

寄存器	偏移量	R/W	描述	复位值
EHCCPR	HSUSBH_BA+0x008	R	EHCI 性能参数寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
EECP							
7	6	5	4	3	2	1	0
IST				Reserved	ASPC	PFLF	AC64

位	描述	
[31:16]	Reserved	保留
[15:8]	EECP	**EHCI 扩展容量指针 (EECP)** 0 = 不支持扩展容量.
[7:4]	IST	**同步列表阈值** 软件依据这里的值，以及主机控制器目前的执行位置，可以可靠的更新同步列表 当 bit [7] 是0，最低3位指示微帧数量，主控制器可以在清除状态前持有一套同步数据结构（一个或多个）。
[3]	Reserved	保留.
[2]	ASPC	**同步列表栈能力** 0 = EHCI 主机控制器不支持同步列表高速队列头栈特性.
[1]	PFLF	**可编程帧列表标志** 0 = EHCI 主机控制器里，系统软件必须使用1024个元素的帧列表
[0]	AC64	**64位寻址能力** 0 = 数据结构使用 32位地址内存指针.

UCMDR USB 指令寄存器

寄存器	偏移量	R/W	描述	复位值
UCMDR	HSUSBH_BA+0x020	R/W	USB 指令寄存器	0x0008_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
ITC							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved	IAAD	ASEN	PSEN	FLSZ		HCRST	RUN

位	描述	
[31:24]	**Reserved**	保留.
[23:16]	**ITC**	**中断阈值控制 (读写)** 系统软件用这里来选择主机控制器执行中断的最大频率。有效的值在下表列出。如果软件写入一个未确定的值，结果不可知 最大中断间隔： 0x00 = 保留. 0x01 = 1 微帧. 0x02 = 2微帧. 0x04 = 4微帧.. 0x08 = 8微帧. (默认, 相当于 1 ms). 0x10 = 16微帧. (2 ms). 0x20 = 32微帧. (4 ms). 0x40 = 64微帧. (8 ms). 这个寄存器填入其他值结果未定义. 程序当HCHalted 为0时修改这一位结果不可知.
[15:7]	**Reserved**	保留.
[6]	**IAAD**	**同步增强型门铃中断(读写)** 这一位被程序用作门铃去告诉主机控制器执行中断，下一次同步计划表递进。程序写1按响门铃。. 当主机控制器已经驱逐所有相关的暂存的计划表状态，设定USBSTS 寄存器同步递进状态中断位为1，主机控制器会在下个中断时隙执行终端。 设定USBSTS 寄存器同步递进状态中断位为1后主机控制器设置这位为0。 程序不能在禁止同步计划表时写这里为1，那样做会导致不可预知的后果。

[5]	**ASEN**	**同步计划表使能 (读写)** 这一位控制主机控制器是否跳过执行同步计划表: 0 = 不要执行同步计划表. 1 = 使用 ASYNCLISTADDR 寄存器执行同步计划表.
[4]	**PSEN**	**周期计划表使能 (读写)** 这一位控制主机控制器是否跳过执行周期计划表: 0 = 不执行周期计划表. 1 = 使用 PERIODICLISTBASE 寄存器执行周期计划表.
[3:2]	**FLSZ**	**帧列表大小 (读写或只读)** 这里仅在HCCPARAMS 寄存器的可编程帧列表标志为1时可读写。这里决定帧列表的大小。帧列表大小控制帧索引寄存器的哪些位用作目前的帧列表索引: 00 = 1024 个元素 (4096字节) 默认值. 01 = 512 元素 (2048 字节). 10 = 256 元素 (1024 字节) – 资源紧张的环境下. 11 = 保留.
[1]	**HCRST**	**主机控制器复位 (HCRESET) (读写)** 这个控制位是主机控制器的软件复位。这一位对于根Hub寄存器的效果与芯片硬件复位类似 当软件向此位写1，主机控制器复位内部流水线，定时器，计数器，状态机等到它们的默认值。任何USB正在执行的流程会被终止， USB 复位不会去复位下游端口。 所有的操作寄存器，包括端口寄存器和端口状态机会回复初始化的值。端口所有权会更改为配套控制器。软件需要重新初始化主机控制器来回到操作状态。 这一位当复位流程结束后会由硬件自动清0，程序不要写0打断复位流程。 程序不能在USBSTS 寄存.
[0]	**RUN**	**运行/停止 (读写)** 写1，主机控制器执行计划表，主机控制器会在此位为1时一直执行。当此位为0，主机控制器完成目前的以及所有有效的流水线的USB传输后挂起。主机控制器在此位清0后需要在16个微帧的时间内挂起。当主机控制器完成流水线传输进入停止态后状态寄存器HC Halted位置1.程序不要在主机控制器在挂起状态（USBSTS寄存器的HCHalted 位为1）时向此位写1，否则将不可预知后果。 0 = 停止. 1 = 运行.

USTSR USB 状态寄存器

寄存器	偏移量	R/W	描述	复位值
USTSR	HSUSBH_BA+0x024	R/W	USB 状态寄存器	0x0000_1000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
ASS	PSS	RECLA	HCHalted	Reserved			
7	6	5	4	3	2	1	0
Reserved		IAA	HSERR	FLR	PCD	UERRINT	USBINT

位	描述	
[31:16]	Reserved	保留.
[15]	ASS	**同步计划表状态 (只读)** 这一位报告目前同步计划表的状态，如果这一位是0，那么同步计划表禁止。如果这一位是1，那么同步计划表使能。.主机控制器并不需要在程序修改USBCMD寄存器的同步计划表使能位后立即禁止或使能。当这里和同步计划表使能位是相同的值，同步计划表是使能或禁止。
[14]	PSS	**周期计划表状态 (只读)** 这一位报告目前周期计划表的状态，如果这一位是0，那么周期计划表禁止。如果这一位是1，那么周期计划表使能。.主机控制器并不需要在程序修改USBCMD寄存器的周期计划表使能位后立即禁止或使能。当这里和周期计划表使能位是相同的值，周期计划表是使能或禁止。
[13]	RECLA	**开拓 (只读)** 这是只读的状态位，用来检测同步计划表空.
[12]	HCHalted	**HCHalted (只读)** 这一位当运行/停止位是1时为0.主机控制器当软件或硬件（例如内部错误）将运行/停止位设为0后将此位置1
[11:6]	Reserved	保留.
[5]	IAA	**同步递进中断 (R/WC)** 系统程序可以向USBCMD寄存器的同步递进门铃位写1强制主机控制器执行下一个同步计划表，这一位是中断源的状态位。.
[4]	HSERR	**主机系统错误 (R/WC)** 主机控制器当主机系统访问主机控制模块发生严重错误时将此位置1

[3]	**FLR**	**帧列表轮转 (R/WC)** 主机控制器当帧列表索引从最大值轮转到0时将此位置1。轮转发生的值取决于帧列表大小。例如, 如果帧列表大小 (USBCMD 寄存器的帧列表大小域) 是 1024, the帧索引寄存器每次在 FRINDEX[13] 反转时回转. 同样的，如果大小是512,主机控制器每次在 FRINDEX[12] 反转时将此位置1.
[2]	**PCD**	**端口改变检测(R/WC)** 主机控制器当任何端口的端口所有者位设为0，位传输从0变1，或者强制端口恢复位从0变1导致挂起端口检测到J-K 跳变时将此位置1。这一位也在系统软件向端口的的端口所有者位写1让出所有权导致连接状态改变时设为1。 这一位运行在辅助电源正常时保持，或者，也接受EHCI 主机控制器设备D3 到 D0 转变这一位又OR 载入所有的PORTSC改变位 (包括: 强制端口恢复, 过流改变, 使能禁止改变和连接状态改变).
[1]	**UERRINT**	**USB 错误中断 (USBERRINT) (R/WC)** 主机控制器当完成的USB事物结果是错误状况（例如计数器下溢错误)时置1，如果没有错误中断发生的 TD 的 IOC位为1, 这一位和USBINT 位都会置1.
[0]	**USBINT**	**USB 中断 (USBINT) (R/WC)** 主机控制器当完成USB事物后将此位置1，隐退的传输描述符的IOC位会置1. 主机控制器也当检测到短包时将此位置1 (实际字节数少于期望字节数).

UIENR USB 中断使能寄存器

寄存器	偏移量	R/W	描述	复位值
UIENR	HSUSBH_BA+0x028	R/W	USB 中断使能寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved		IAAEN	HSERREN	FLREN	PCIEN	UERRIEN	USBIEN

位	描述	
[31:6]	Reserved	**同步递进中断使能进制位** 当此位置1，且USBSTS 寄存器的同步递进中断位是1，主机控制器会在下一个中断入口执行中断. 中断在软件清除同步递进中断位后确认 0 = 禁止同步递进中断. 1 = 使能同步递进中断
[5]	IAAEN	**主机控制器错误使能禁止位** 当此位为1，并且USBSTS 寄存器的主机系统错误状态是1，主机控制器会执行一个中断。中断当软件清除主机错误中断位后确认 0 = 主机系统错误中断禁止. 1 = 主机系统错误中断使能
[4]	HSERREN	**帧列表回转使能禁止位** 当此位为1，且USBSTS寄存器的帧列表回转位为1，主机控制器会执行一个中断。中断当软件清除帧列表回转位后确认. 0 = 帧列表回转中断禁止. 1 = 帧列表回转中断使能.
[3]	FLREN	**端口改变中断使能禁止位** 当此位为1，且USBSTS寄存器的端口改变检测位是1，主机控制器会执行一个中断。中断当软件清除端口中断检测位后确认. 0 = 端口改变中断禁止. 1 = 端口改变中断使能.
[2]	PCIEN	**同步递进中断使能进制位** 当此位置1，且USBSTS 寄存器的同步递进中断位是1，主机控制器会在下一个中断入口执行中断. 中断在软件清除同步递进中断位后确认 0 = 禁止同步递进中断. 1 = 使能同步递进中断

[1]	**UERRIEN**	**USB 错误中断使能禁止位** 当此位为1，且USBSTS寄存器的USBER-RINT 位为1,主机控制器会在下一个中断入口执行一个中断。中断当软件清除 USBERRINT 位后确认. 0 = USB 错误中断禁止. 1 = USB 错误中断使能.
[0]	**USBIEN**	**USB 中断使能禁止位** 当此位为1，且USBSTS 的USBINT位为1,主机控制器会在下一个中断入口执行一个中断，中断在软件清除USBINT位后确认. 0 = USB 中断禁止. 1 = USB 中断使能.

UFINDR USB 帧索引寄存器

寄存器	偏移量	R/W	描述	复位值
UFINDR	HSUSBH_BA+0x02C	R/W	USB 帧索引寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved		FI					
7	6	5	4	3	2	1	0
FI							

位	描述	
[31:14]	Reserved	保留.
[13:0]	FI	**帧索引** 这里的值在每个时间帧（如微帧）结束时加1， Bits [N:3] 用做帧列表目前索引。这意味着帧列表的每个位置在移到下一个索引前会访问8个时间周期（帧或微帧）。下表是根据USBCMD 帧列表大小的对应值. **FLSZ (UCMDR[3:2] Number Elements N** 0x0 1024 12 0x1 512 11 0x2 256 10 0x3 保留

UPFLBAR USB 周期帧列表基地址寄存器

寄存器	偏移量	R/W	描述	复位值
UPFLBAR	HSUSBH_BA+0x034	R/W	USB 周期帧列表基地址寄存器	0x0000_0000

31	30	29	28	27	26	25	24
BADDR							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
BADDR							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
BADDR				**Reserved**			
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved							

位	描述	
[31:12]	**BADDR**	**基地址** 这些位对应内存地址信号 [31:12]
[11:0]	**Reserved**	保留.

UCALAR USB 目前同步列表地址寄存器

寄存器	偏移量	R/W	描述	复位值
UCALAR	HSUSBH_BA+0x038	R/W	USB 目前同步列表地址寄存器	0x0000_0000

31	30	29	28	27	26	25	24
LPL							
23	22	21	20	19	18	17	16
LPL							
15	14	13	12	11	10	9	8
LPL							
7	6	5	4	3	2	1	0
LPL			Reserved				

位	描述	
[31:5]	LPL	**链接低指针 (LPL)** 这些位对应与内存地址信号[31:5], 只被 Queue Head (QH)参考.
[4:0]	Reserved	保留.

UASSTR USB 异步计划表休眠定时器寄存器

寄存器	偏移量	R/W	描述	复位值
UASSTR	HSUSBH_BA+0x03C	R/W	USB 异步计划表休眠定时器寄存器	0x0000_0BD6

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved				ASSTMR			
7	6	5	4	3	2	1	0
ASSTMR							

位	描述	
[31:11]	Reserved	保留
[11:0]	ASSTMR	异步计划表休眠定时器 这里决定EHCI AsyncSchedSleep时间. 这里用来控制当异步计划表为空。主机控制器多久去系统内存搬运异步计划表 这个定时器的默认值是 12'hBD6. 因为这个定时器用 UTMI 时钟 (30MHz) 休眠时间差不多 100us.

UCFGR USB 配置标识寄存器

寄存器	偏移量	R/W	描述	复位值
UCFGR	HSUSBH_BA+0x060	R/W	USB 配置标志寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved							CF

位	描述	
[31:1]	Reserved	保留
[0]	CF	**配置标识 (CF)** 主机程序设置此位作为配置主机定时器的最后行动。这一位控制默认端口出线控制逻辑 0 = 每个端口出线逻辑默认贯彻标准控制器默认逻辑. 1 = P每个端口出线逻辑默认贯彻此控制器默认逻辑

UPSCR USB 端口状态和控制寄存器

寄存器	偏移量	R/W	描述	复位值
UPSCR0	HSUSBH_BA+0x064	R/W	USB 端口 0 端口和控制寄存器	0x0000_2000
UPSCR1	HSUSBH_BA+0x068	R/W	USB 端口 1 端口和控制寄存器	0x0000_2000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved				PTC			
15	14	13	12	11	10	9	8
Reserved		PO	PP	LSTS		Reserved	PRST
7	6	5	4	3	2	1	0
SUSPEND	FPR	OCC	OCA	PEC	PE	CSC	CCS

位	描述	
[31:20]	**Reserved**	保留.
[19:16]	**PTC**	**端口测试控制 (读写)** 当这里是0，端口不会工作在测试模式。非零值表示工作特定的测试模式，测试模式编码如下(0x6 ~ 0xF 保留): 测试模式位 0x0 = 测试模式关闭 0x1 = 测试J_STATE. 0x2 =测试K_STATE. 0x3 =测试SE0_NAK. 0x4 =测试包. 0x5 =测试FORCE_ENABLE.
[15:14]	**Reserved**	保留.
[13]	**PO**	**端口所有者 (读写)** 当CONFIGFLAG 寄存器的控制位由0变1时，这一位无条件的变为0b 。当控制位为1时，此位无条件变1.. 系统软件用这一位来释放端口所有权来选择主机控制器（如果插入的设备不是高速设备）当插入的不是高速设备时程序向这里写1，这里是1意味着配套控制器控制这个端口
[12]	**PP**	**端口电源 (PP)** 主机控制器有电源控制切换，这一位反映目前的开关设定（0关闭，1打开），当端口没有有效电源（PP为0），端口无法工作，且不会报告插入，拔出等。. 当供电端口检测到过流状况，且PPC 是1，主机控制器会设置 PP位由1变0（关闭端口电源）

[11:10]	**LSTS**	**线路状态(RO)** 这里反映目前信号线 D+ (bit 11) 与 D- (bit 10) 信号线的逻辑电平。这两位用来在端口复位前检测低速设备以及使能序列。这里仅在使能位为0且目前连接状态位是1时有效. 位编码如下: Bits[11:10] USB 状态翻译 00 = SE0 不是低速设备, 执行 EHCI 复位. 01 = K-state 低速设备, 释放端口所有权. 10 = J-state不是低速设备, 执行 EHCI 复位. 11 = 未定义的不是低速设备，执行 EHCI 复位. 在端口电源是0，这里的值没有定义
[9]	**Reserved**	保留.
[8]	**PRST**	**端口复位 (读写)** 当软件向这里写1 (之前是0), 就开始 USB2.0标准定义的复位序列。程序向这里写0中断复位序列。程序需要保持这一位为1足够的时间来确保 USB2.0协议要求的复位序列能够完成。. 注意:当程序向这里写1，也必须想端口使能位写0。 注意当程序向这里写0，到这一位变为0会有一个延时。这一位直到复位完成才会读为0。如果复位完成端口会是高速模式，主机控制器会自动使能这个端口 (设置端口使能位为1). 这一位由1变0，主机控制器终止复位且保留端口状态2mS。例如：如果端口在复位时检测到插入设备是高速设备，那么程序向这一位写0，主机控制器保证端口在使能状态2 mS 在程序尝试使用这一位之前，USBSTS寄存器的 HCHalted 位需要是0。主机控制器在HCHalted为1时保持端口复位。. 当端口电源是0时，此位是0. 0 = 端口没有复位. 1 = 端口复位.
[7]	**SUSPEND**	**挂起 (读写)** 这个寄存器的端口使能位和挂起位定义端口状态如下： 00 = 端口禁止. 01 = 端口禁止. 10 = 端口使能. 11 = 端口挂起. 在挂起态，下游传播数据在这个端口屏蔽，除了端口复位。如果传输正在进行时此位被置1，屏蔽将在传输结束时发生。在挂起态，端口检测恢复信号。注意这一位的状态直到端口进入挂机态后才会改变，如果设置此位时有USB事物正在进行，那么就会有一个延时。 向此位写0会被主机控制器忽略，主机控制器会在下列条件将此位置0： 程序设置强制端口恢复位为0（从1） 程序设置端口复位位为1（从0）. 如果设置此位为1，但是端口没有使能（端口使能位为0），结果未知。 当端口电源为0，此位为0. 0 = 端口不在挂起态. 1 = 端口在挂起态.

[6]	**FPR**	**强制端口恢复 (读写)** 操作这一位要依据Suspend 位的值。例如，如果端口没有挂起（挂起位和使能位都是1），而程序修改此位为1，对总线的影响不可知。. 程序设置此位为1驱动恢复信号。当端口在挂起态，主机控制器检测到J-to-K 跳变时设置此位为1。当是因为检测到J-to-K跳变设置此位是1，USBSTS寄存器的端口改变检测位也会也会置1。如果是程序设置此位为1，主机控制器不会设置端口改变检测位。. 注意当EHCI 控制器拥有端口，回复序列依照 USB 2.0协议规定。在此位为1时端口持续产生恢复信号(全速 'K') 。程序需要评估恢复信号的时间然后将此位置0。将此位从1写0是端口恢复到高速模式（强制总线下游端口进入高速空闲）这一位保持为1直到端口切换到高速空闲。在软件将此位写0后主机控制器需要在2mS内完成这些. 当端口电源是0此位是0. 0 = 端口没有检测到或驱动恢复 (Kstate) . 1 = 端口检测到或驱动恢复.
[5]	**OCC**	**过流改变 (R/WC)** 1 = 当改变为过流有效时此位为1，程序写1清除此位.
[4]	**OCA**	**过流有效 (RO)** 此位在过流状况消除后自动由1变0. 0 = 这个端口没有过流状况. 1 = 端口有过流状况.
[3]	**PEC**	**端口使能/禁止改变 (R/WC)** 对于根hub, 这一位仅在端口因为EOF2点上的相应条件被禁止时置1（见USB规范第11章端口错误定义），程序写1清除此位. 如果端口电源是0此位是0. 0 = 没有改变. 1 = 端口使能禁止状态改变了.
[2]	**PE**	**Port 使能/禁止 (R/W)** 端口仅在作为复位与使能的一部分有主机控制器使能。程序不能向这里写1使能端口。主机控制器仅在复位序列中决定插入的设备是高速设备时将此位置1。. 端口可以由任意错误条件（未连接事件或其他错误条件）禁止或由程序禁止。注意这一位在端口状态真正改变后才会改变，可能在禁止使能端口时由于其他控制器或总线事件造成延时。 当端口被禁止 (0b)，下游数据会在这个端口被屏蔽，除了复位. 如果端口电源是0这里是0. 0 = 端口禁止. 1 = 端口使能.
[1]	**CSC**	**连接状态改变 (R/W)** 指示端口目前的连接状态改变了。主机控制器会在任何端口设备连接状态改变时设置此位，即使系统软件没有清除一个存在的连接状态改变。例如，在系统软件清除改变状态前插入状态改变了两次，硬件会设置已经设置的位。程序对此位写1清0。 . 当端口电源是0这里是0. 0 = 没有改变. 1 = 目前连接状态改变了.

[0]	**CCS**	**目前连接状态 (只读)** 这位反映目前端口的状态，不一定直接引起连接状态改变位(Bit 1) 为1. 如果端口电源是0此位是0. 0 = 没有设备连接 1 = 端口上有设备

USBPCR0 USB PHY 0 控制寄存器

寄存器	偏移量	R/W	描述	复位值
USBPCR0	HSUSBH_BA+0x0C4	R/W	USB PHY 0 控制寄存器	0x0000_0060

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved				**CLKVALID**	**Reserved**		**SUSPEND**
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved							

位	描述	
[31:12]	**Reserved**	保留.
[11]	**CLKVALID**	**UTMI 时钟有效** 这一位指示从USB 2.0 PHY来的 UTMI时钟有效. 程序在这个标识有效前不要去写控制寄存器. 0 = UTMI 时钟无效. 1 = UTMI 时钟有效.
[10:9]	**Reserved**	保留.
[8]	**SUSPEND**	**挂起声明** 这一位控制 USB PHY 0.挂起模式 当 PHY 已挂起, PHY 的所有电路会掉电，输出三态。 这一位默认是 1’b0 ， 意味着 USB PHY 0默认是挂起。需要在配置 USB主机控制器之前设置此位为1’b1 让 USB PHY 0 离开挂起状态. 0 = USB PHY 0 已挂起. 1 = USB PHY 0 没有挂起
[7:0]	**Reserved**	保留.

USBPCR1 USB PHY 1 控制寄存器

寄存器	偏移量	R/W	描述	复位值
USBPCR1	HSUSBH_BA+0x0C8	R/W	USB PHY 1 控制寄存器	0x0000_0020

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							**SUSPEND**
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved							

位	描述	
[31:9]	**Reserved**	保留.
[8]	**SUSPEND**	**挂起声明** 这一位控制 USB PHY 1.挂起模式 当 PHY 已挂起, PHY 的所有电路会掉电，输出三态。 这一位默认是 1'b0 ， 意味着 USB PHY 1默认是挂起。需要在配置 USB主机控制器之前设置此位为1'b1 让 USB PHY 1 离开挂起状态. 0 = USB PHY 1 已挂起. 1 = USB PHY 1 没有挂起
[7:0]	**Reserved**	保留.

6.35 USB On-The-Go (OTG)

6.35.1 概述

OTG 控制器连接 USB PHY 和 USB 控制器，包含USB 1.1 主机控制器和 USB 2.0 全速设备控制器。OTG 控制器支持“On-The-Go和集成主机 USB 2.0协议版本2.0 补充规范”定义的HNP 和 SRP 协议

USB 框架, 包括 USB 主机, USB 设备, 和 OTG 控制器, 可以配置为仅主机, 仅设备, 依据ID或USBROLE (SYS_USBPHY[1:0])定义的OTG 设备模式。在仅主机模式下， USB 框架工作在USB主机。 USB 框架都支持全速和低速传输。在仅从机模式, USB 工作在 USB 设备. USB 框架只支持全速传输。在依据 ID 模式, USB 框架依据USB_ID 管脚状态可以工作在 USB 主机和 USB 设备模式。在 OTG 设备模式, USB 框架的角色依据 OTG 标准的定义。USB 框架在OTG 设备作为外设时只支持全速传输。

6.35.2 特性

- 内建 USB PHY
- 可配置工作在:
  - 仅主机
  - 仅设备
  - 依据ID: USB框架的角色仅依据USB_ID 管脚状态—作为 USB 主机 (USB_ID 管脚是低)或USB 设备 (USB_ID 管脚是高). 不支持 HNP或SRP 协议.
  - OTG设备:依据 USB_ID 管脚状态作为 A-device (USB_ID 管脚是低)或B-device (USB_ID 管脚是高).支持 HNP 和 SRP 协议.

6.35.3 框图

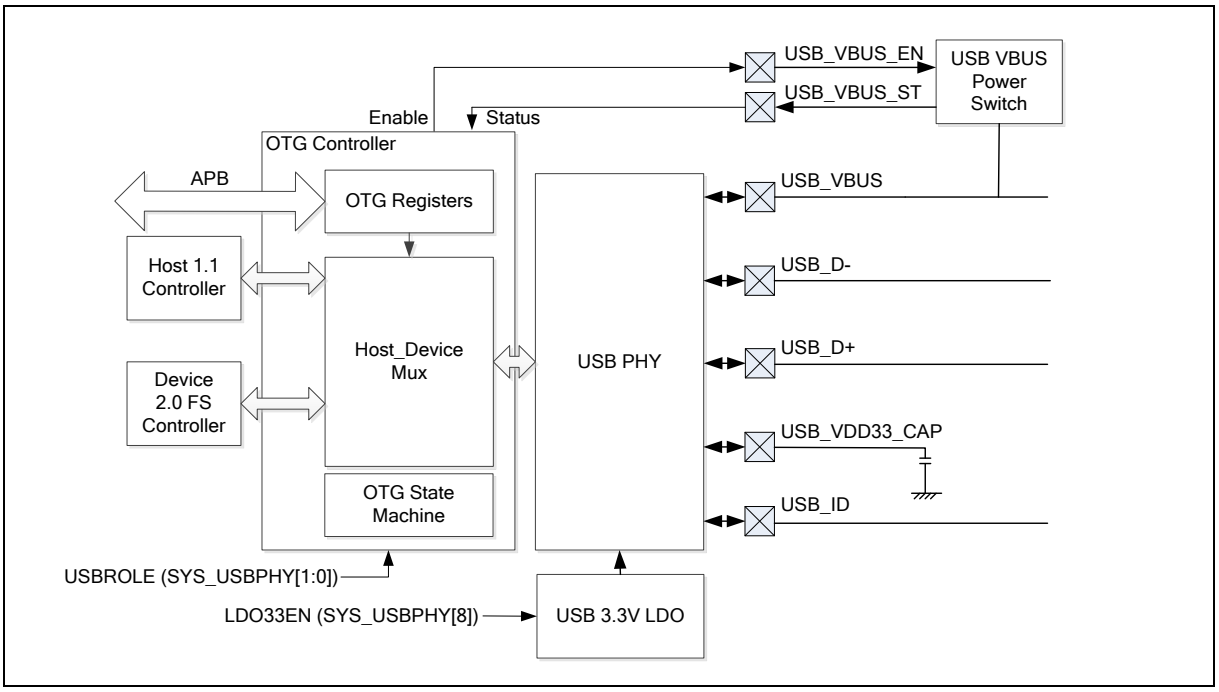


图6.35-1 USB OTG 框图

6.35.4 基本配置

OTG 外设时钟由 OTGCKEN (CLK_APBCLK0[26]) 使能。USB 框架的角色由 USBROLE (SYS_USBPHY[1:0])设置。内部 USB 3.3V LDO 由 LDO33EN (SYS_USBPHY[8])使能。这两个配置是写保护位，在写这些位之前，用户需要禁用寄存器保护功能。参考 SYS_REGLCTL 寄存器的描述。USB_VBUS_EN 和 USB_VBUS_ST 管脚功能由SYS_GPA_MFPL或SYS_GPC_MFPL 寄存器配置。

6.35.5 功能描述

USB 框架的角色依据USBROLE (SYS_USBPHY[1:0]) 的设定以及USB_ID 管脚状态决定。 USBROLE 的配置优先于USB_ID 管脚的状态。用户可以配置 OTG 控制器到 USB 主机模式, USB 设备模式,依据 ID 模式和 OTG 设备模式。在 USB 主机模式, 主机控制器直接与USB PHY 互动。在 USB 设备模式, 设备控制器直接与 USB PHY 互动。在这些情况下， OTG 控制器作为简单的多路复用器使用。在依据 ID 模式, USB_ID 管脚的状态决定 USB 帧充当 USB 主机或 USB 设备。如果USB_ID 管脚在 FALSE 状态 (低电平), USB 框架充当 USB 主机。如果 USB_ID 管脚在 TRUE 状态 (高电平), USB 帧充当 USB 设备。在OTG 设备模式, OTG 控制器会执行OTG HNP 和 SRP 协议，如果USB_ID 管脚是 FALSE 状态 (低电平), OTG 控制器充当 OTG A-设备，如果 USB_ID 管脚是 TRUE 状态 (高电平), OTG 控制器充当 OTG B-设备.

6.35.5.1 USB框架的角色

USB 设备模式

当 USBROLE (SYS_USBPHY[1:0]) 设为 0, USB框架充当 USB 设备. USB 主机功能不可用.

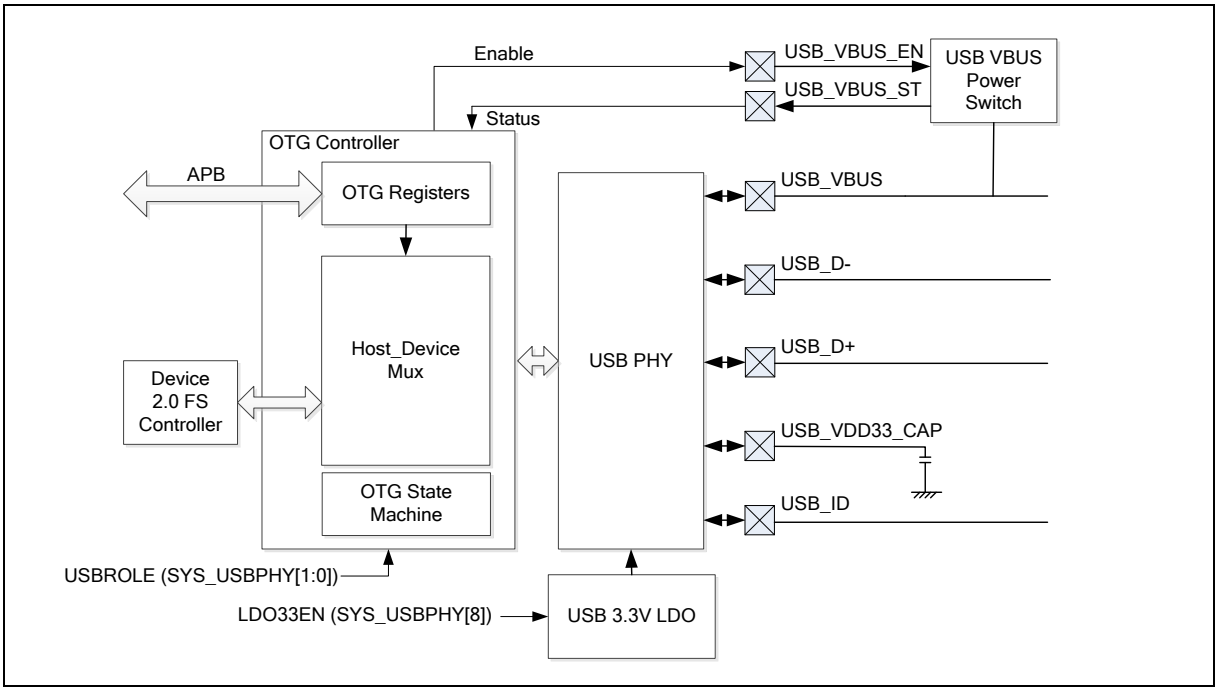


图 6.35-2 USB 设备模式

USB主机模式

当 USBROLE (SYS_USBPHY[1:0])设为1, USB 框架充当 USB 主机. USB 设备功能不可用.

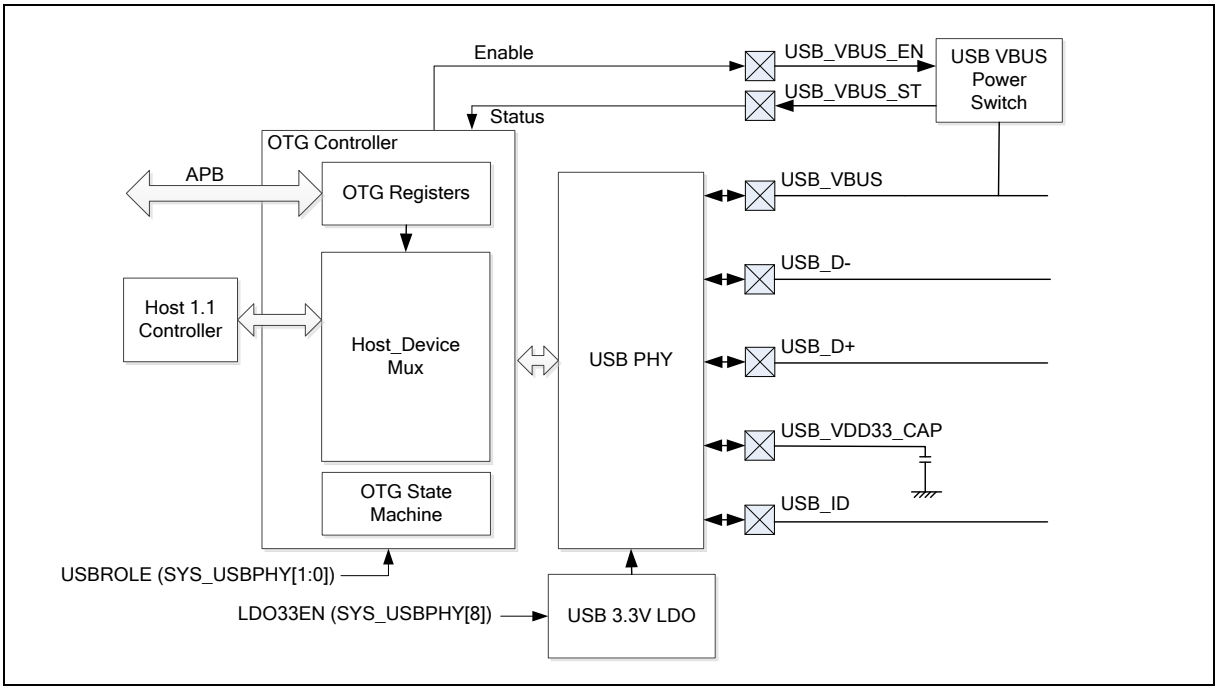


图 6.35-3 USB 主机模式

依据ID 模式

当 USBROLE (SYS_USBPHY[1:0])设为 2, USB 框架的角色依据 USB_ID 管脚的状态. ID 检测功能由 IDDETEN (OTG_PHYCTL[1]) 使能。 USB_ID 管脚状态影响IDSTS (OTG_STATUS[1]). 当 USB 框架充当USB 主机 (USB_ID 管脚低), 框图与 USB 主机模式相同。当 USB框架充当 USB 设备 (USB_ID管脚是高), 框图与 USB 设备模式相同。

OTG 设备模式

当 USBROLE (SYS_USBPHY[1:0]) 等于 3, USB框架的角色取决于USB_ID 管脚的状态。ID 检测功能由 IDDETEN (OTG_PHYCTL[1]) 使能。 USB_ID 管脚状态影响 IDSTS (OTG_STATUS[1]). 当 USB_ID 管脚是低电平, OTG 控制器充当 OTG A-device. 当 USB_ID 管脚是高电平, OTG 控制器充当 OTG B-device. 请参考OTG 标准得到详细的A-device 和B-device的行为描述

6.35.5.2 会话请求协议(SRP)

当 USB 框架配置为OTG 设备模式, OTG 控制器支持 SRP来保护电源，参考OTG 标准得到SRP的详细描述.

A-device会话请求协议

1. A-device 关闭 USB 总线电源保护电源， B-device 通过检查 VBUS 的状态识别到这一状况。
2. 当 B-device 想要连接到 A-device，B-device 在数据线产生脉冲请求 A-device 提供 USB 总线电源
3. A-device 通过检查 SRPDETIF (OTG_INTSTS[13])认到 USB 总线电源请求
4. A-device 当 SRPDETIF (OTG_INTSTS[13])由硬件置 1 时通过设置 BUSREG (OTG_CTL[1])为 1 开始驱动 VBUS，如果 VBUS 在特定时间内达到有效电平，并且 B-device 已连接, A-device 会变为 USB 主机, HOSTIF (OTG_INTSTS[7]) 会设为 1. 如果 VBUS 在特定时间内不能达到有效电平，这意味着电流事件发生，那么 VBUS 错误位, VBEIF (OTG_IS[1]), 会设为 1.

B-device会话请求协议

1. A-device 关闭 USB 总线电源保护电源. B-device 通过检查 VBUSVLD (OTG_STATUS[5])确认这一事件.
2. B-device 可以通过设定 BUSREQ(OTG_CTL[1])请求 A-device 提供 USB 总线电源.
3. B-device 会产生 OTG 定义的数据线脉冲
4. A-device 检测到数据线脉冲后会驱动 VBUS， B-device 可以通过确认 VBUSVLD (OTG_STATUS[5])识别这一状况。如果 A-device 在特定时间内驱动 VBUS 到有效电平, B-device 变为 USB 外设, PDEVIF (OTG_INTSTS[6]) 设为 1. 如果 A-device 没有在特定时间内驱动 VBUS 驱动到有效电平, SRP 错误标志, SRPFIF (OTG_INTSTS[2]), 会置 1 且 B-device 会进入 OTG 协议定义的空闲态.

6.35.5.3 主机谈判协议(HNP)

当USB 框架配置为 OTG 设备模式, 主机功能可以在两个直连的OTG 设备间切换，而不要改变接头连接，参考OTG 标准得到HNP的详细介绍.

A-device 主机谈判协议

1. OTG 协议定义的 A-Host 发送 SetFeature b_hnp_enable 指令使能 B-device HNP 功能. B-device 响应 ACK 表明 B-device 支持 HNP. 用户需要设定 HNPREQEN (OTG_CTL[2]) 为 1 使能 HNP 协议.
2. A-Host 完成所有必须的操作之后设置 BUSREQ (OTG_CTL[1])为 0 进入挂起状态，使 USB 总线进入 J-state (USB_D+ 高且 USB_D- 低)
3. A-Host 通过在特定的时间内检测 USB_D+ 和 USB_D- low 为低检测到 B-peripheral 断开连接，A-Host 变为 A-Peripheral。如果 A-Host 在特定时间内不能检测到 B-Peripheral 断开连接, A-Host 会回到空闲态.

B-device 主机谈判协议

1. 当 B-Peripheral 成功接收到 SetFeature b_hnp_enable 指令, user 通过设定 HNPREQEN (OTG_CTL[2])为 1 使能 B-peripheral HNP 功能.
2. 在检测到 USB 总线在 J-state(USB_D+ 高且 USB_D- 低)，用户设置 BUSREG (OTG_CTL[1])为 1. 接着 USB_D+ 上拉电阻移除导致 USB 断开连接状态 (USB_D+ 低且 USB_D- 低).
3. 如果 B-device 在特定时间内检测到 A-device 已连接 (USB_D+ 高), B-device 变为 B-Host. 如果 B-device 在特定时间内无法检测到 A-device 已连接 (USB_D+ 高), HNP 错误标志, HNPFIF (OTG_INTSTS[3]), 会设为 1.

6.35.6 寄存器映射

R:只读, **W:**只写, **R/W:** 可读可写

寄存器	偏移量	R/W	描述	复位值
OTG 基地址: **OTG_BA = 0x4004_D000**				
OTG_CTL	OTG_BA+0x00	R/W	OTG 控制寄存器	0x0000_0000
OTG_PHYCTL	OTG_BA+0x04	R/W	OTG PHY控制寄存器	0x0000_0000
OTG_INTEN	OTG_BA+0x08	R/W	OTG 中断使能寄存器	0x0000_0000
OTG_INTSTS	OTG_BA+0x0C	R/W	OTG 中断状态寄存器	0x0000_0000
OTG_STATUS	OTG_BA+0x10	R	OTG 状态寄存器	0x0000_0006

6.35.7 寄存器描述

OTG_CTL OTG 控制寄存器

寄存器	偏移量	R/W	描述	复位值
OTG_CTL	OTG_BA+0x00	R/W	OTG控制寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved		WKEN	OTGEN	Reserved	HNPREQEN	BUSREQ	VBUSDROP

位	描述	
[31:6]	**Reserved**	保留
[5]	**WKEN**	**OTG ID管脚唤醒使能位** 0 = OTG ID 管脚状态改变唤醒功能禁止. 1 = OTG ID 管脚状态改变唤醒功能使能
[4]	**OTGEN**	**OTG 功能使能位** 用户需要当 USB框架设为 OTG设备时设置此位使能OTG功能. 当 USB框架没有设为 OTG 设备时，此位需要设为0. 0= OTG 功能禁止. 1 = OTG 功能使能.
[3]	**Reserved**	保留
[2]	**HNPREQEN**	**OTG HNP 请求使能位** 当USB框架作为 A-device, 设置此位当A-device 允许执行 HNP协议—A-device 改变角色从主机变为外设。当OTG状态从a_suspend变为a_peripheral或回到a_idle状态时此位会清0 当 USB 框架作为 B-device, 设置此位在 OTG A-device 成功发送 SetFeature (b_hnp_enable) 指令到 OTG B-device开始角色转换后—B-device改变角色从外设到主机。这一位会在OTG 状态从b_peripheral 变为b_wait_acon 或回到 b_idle 状态后清0. 0 = HNP 请求禁止. 1 = HNP 请求使能 (A-device 改变角色从主机到设备， B-device 可以改变角色从外设到主机). **注意:**参考 OTG 协议了解 a_suspend, a_peripheral, a_idle 和 b_idle状态.
[1]	**BUSREQ**	**OTG 总线请求** 如果OTG A-device 通过USB总线传输数据, 设定此位驱动 VBUS位高检测 USB 设备连接. 如果用户不再想使用总线，清除此位拉低VBUS 来节省电源. 这一位当 A-device 变为 A_wait_vfall 态后清除. 当 VBUSDROP (OTG_CTL[0]) 置1或IDSTS (OTG_STATUS[1]) 改变后清除. 如果OTG-B 设备想要请求VBUS, 设定这一位运行SRP协议. 这一位当SRP错误 (在B-

位	描述	
		device 在特定时间内执行OTG 协议定义的ARP ， OTG A-设备不提供VBUS)后清除。此位也在 VBUSDROP (OTG_CTL[0]) 位为1且 IDSTS (OTG_STATUS[1]) 改变后清除. 0 = OTG A-device 没有发动VBUS 或OTG B-device 没有请求SRP 1 = OTG A-device 发动VBUS 或OTG B-device 请求SRP.
[0]	**VBUSDROP**	**拉低 VBUS控制** 如果OTG A-device上的用户应用想要保护电源，设置这一位拉低VBUS. 不管是A-device或B-device BUSREQ (OTG_CTL[1])都会清除. 0 = 没有拉低 VBUS. 1 = 拉低 VBUS.

OTG_PHYCTL OTG PHY 控制寄存器

寄存器	偏移量	R/W	描述	复位值
OTG_PHYCTL	OTG_BA+0x04	R/W	OTG PHY 控制寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved		VBSTSPOL	VBENPOL	Reserved		IDDETEN	OTGPHYEN

位	描述	
[31:6]	Reserved	保留.
[5]	VBSTSPOL	**片外 USB VBUS 电源开关状态极性** 片外USB VBUS 电源开关信号的有效极性依据所选择的组件。USB_VBUS_ST 管脚用来监控外设 USB VBUS 电源开关的有效信号. 依据外部USB VBUS 电源开关的极性设置此位. 0 = 片外USB VBUS 电源开关信号的有效极性是高. 1 = 片外USB VBUS 电源开关信号的有效极性是低.
[4]	VBENPOL	**片外 USB VBUS 电源开关使能极性** OTG 控制器需要在需要时会使能片外USB VBUS 电源开关提供VBUS 电源. USB_VBUS_EN 管脚用来控制片外USB VBUS 电源开关. 使能外部USB VBUS 电源开关的极性(高有效或低有效) 依据所选择的器件。设置这一位选择合适的极性. 0 = 片外 USB VBUS 电源开关使能高有效. 1 = 片外 USB VBUS 电源开关使能低有效
[3:2]	Reserved	保留.
[1]	IDDETEN	**ID 检测使能位** 0 = 禁止检测 ID 管脚状态 1 = 使能检测 ID 管脚状态.
[0]	OTGPHYEN	**OTG PHY 使能** 当 USB 框架配置为OTG-device或ID-dependent, 用户需要在使用OTG 功能前设置此位。如果设备没有配置为 OTG-device 或ID-dependent , 此位不重要 0 = OTG PHY 禁止. 1 = OTG PHY 使能.

OTG_INTEN OTG 中断使能寄存器

寄存器	偏移量	R/W	描述	复位值
OTG_INTEN	OTG_BA+0x08	R/W	OTG中断使能寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved		SRPDETIEN	Reserved	SECHGIEN	VBCHGIEN	AVLDCHGIEN	BVLDCHGIEN
7	6	5	4	3	2	1	0
HOSTIEN	PDEVIEN	IDCHGIEN	GOIDLEIEN	HNPFIEN	SRPFIEN	VBEIEN	ROLECHGIEN

位	描述	
[31:14]	**Reserved**	保留.
[13]	**SRPDETIEN**	**SRP 检测中断使能位** 0 = 中断禁止. 1 = 中断使能.
[12]	**Reserved**	保留.
[11]	**SECHGIEN**	**SESSEND 状态改变中断使能位** 如果此位是 1 且 SESSEND (OTG_STATUS[2]) 状态改变，中断会产生 0 = 中断禁止. 1 = 中断使能.
[10]	**VBCHGIEN**	**VBUSVLD 状态改变中断使能位** 如果此位是 1 且 VBUSVLD (OTG_STATUS[5])状态改变，中断会产生 0 = 中断禁止. 1 = 中断使能.
[9]	**AVLDCHGIEN**	**A-device 会话有效状态改变中断使能位** 如果此位是 1 且 AVLD (OTG_STATUS[4])状态改变，中断会产生 0 = 中断禁止. 1 = 中断使能.
[8]	**BVLDCHGIEN**	**B-device 会话有效状态改变中断使能位** 如果此位是 1 且 BVLD (OTG_STATUS[3]) 状态改变，中断会产生. 0 = 中断禁止. 1 = 中断使能.
[7]	**HOSTIEN**	**充当主机中断使能位** 如果此位是 1 且设备作为主机，中断会产生.

位	描述	
		0 = 设备变主机中断禁止. 1 = 设备变主机终端使能.
[6]	**PDEVIEN**	**充当外设中断使能位** 如果此位为1，设备变为外设，中断会产生 0 = 设备变为外设中断禁止. 1 = 设备变为外设中断使能.
[5]	**IDCHGIEN**	**IDSTS 改变中断使能** 如果此位为 1 且 IDSTS (OTG_STATUS[1]) 状态改变，中断会产生. 0 = 中断禁止. 1 = 中断使能.
[4]	**GOIDLEIEN**	**OTG 设备变为 IDLE 状态中断使能位** 0 = 中断禁止. 1 = 中断使能. **注意:**进入空闲态意味着进入a_idle或b_idle 状态. 请参考OTG标准A-device 状态图和 B-device 状态图.
[3]	**HNPFIEN**	**HNP 失败中断使能位** 0 = 中断禁止. 1 = 中断使能.
[2]	**SRPFIEN**	**SRP 失败中断使能位** 0 = 中断禁止. 1 = 中断使能.
[1]	**VBEIEN**	**VBUS 错误中断使能位** 0 = 中断禁止. 1 = 中断使能. **注意:** VBUS 错误意味着进入a_vbus_err状态. 请参考OTG标准A-device 状态图.
[0]	**ROLECHGIEN**	**角色 (主机或外设) 改变中断使能位** 0 = 中断禁止. 1 = 中断使能.

OTG_INTSTS OTG 中断状态寄存器

寄存器	偏移量	R/W	描述	复位值
OTG_INTSTS	OTG_BA+0x0C	R/W	OTG 中断状态寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved		SRPDETIF	Reserved	SECHGIF	VBCHGIF	AVLDCHGIF	BVLDCHGIF
7	6	5	4	3	2	1	0
HOSTIF	PDEVIF	IDCHGIF	GOIDLEIF	HNPFIF	SRPFIF	VBEIF	ROLECHGIF

位	描述	
[31:14]	**Reserved**	保留.
[13]	**SRPDETIF**	**SRP 检测中断状态** 0 = SRP 没有检测到. 1 = SRP 检测到. **注意: 写1清除这个状态**.
[12]	**Reserved**	**保留**
[11]	**SECHGIF**	**SESSEND 状态改变中断状态** 0 = SESSEND (OTG_STATUS[2]) 没有翻转. 1 = SESSEND (OTG_STATUS[2]) 改变. **注意: 写1清这个状态**.
[10]	**VBCHGIF**	**VBUSVLD 状态改变中断状态** 0 = VBUSVLD (OTG_STATUS[5]) 没有翻转. 1 = VBUSVLD (OTG_STATUS[5]) 改变. **注意: 写1清这个状态**.
[9]	**AVLDCHGIF**	**A-device 会话有效状态改变中断状态** 0 = AVLD (OTG_STATUS[4]) 没有翻转. 1 = AVLD (OTG_STATUS[4]) 改变. **注意: 写1清这个状态**.
[8]	**BVLDCHGIF**	**B-device 会话有效状态改变中断状态** 0 = BVLD (OTG_STATUS[3]) 没有翻转. 1 = BVLD (OTG_STATUS[3])改变. **注意: 写1清这个状态**.
[7]	**HOSTIF**	**作为主机中断状态**

位	描述	
		0= 设备没有作为主机. 1 = 设备作为主机. **注意: 写1清这个状态**.
[6]	**PDEVIF**	**充当外设中断状态** 0= 设备没有充当外设. 1 = 这个设备充当外设. **注意: 写1清这个标志**.
[5]	**IDCHGIF**	**ID 状态改变中断状态** 0 = IDSTS (OTG_STATUS[1]) 没有翻转. 1 = IDSTS (OTG_STATUS[1]) 改变. **注意: 写1清这个标志**.
[4]	**GOIDLEIF**	**OTG 进入 IDLE 中断状态** 当OTG设备从非空闲变为空闲态此位置1， OTG设备可以是主机或者是外设l. 0 = OTG 设备没有回到空闲态 (a_idle或b_idle). 1 = OTG 设备回到空闲态(a_idle或b_idle). **注意 1:**进入空闲态就是进入a_idle或b_idle 状态. 请参考 OTG 规范. **注意 2:**写1清这个标志.
[3]	**HNPFIF**	**HNP 错误中断状态** 当 A-device 已授予 B-device作为主机，且 USB 总线在 SE0 (USB_D+ 和 USB_D-都是低)状态, 当 A-device 在期望时间内没有连接此位置1. 0 = A-device在特定时间内连上了 B-device. 1 = A-device 在特定时间内没有连上B-device. **注意:**写1清这个标志.
[2]	**SRPFIF**	**SRP 错误中断状态** 初始化 SRP 之后，OTG B-device 会等待OTG A-device至少在OTG规范定义的TB_SRP_FAIL minimum的时间内拉高VBUS, 当 OTG B-device 在这个时间内没有检测到VBUS 拉高，这个标志置1. 0 = OTG B-device 在这时间内检测到 VBUS 变高. 1 = OTG B-device 没有在这时间内检测到 VBUS 变高. **注意:**写1清这个标志.
[1]	**VBEIF**	**VBUS 错误中断状态** 当OTG A-device 开始驱动VBUS 为高之后，在最大时间 100ms后VBUS 的电平仍无法达到最低阈值4.4V 时此位置1. 0 = OTG A-device 驱动 VBUS 在期望时间内达到阈值电压. 1 = OTG A-device 驱动 VBUS 在期望时间内没有达到阈值电压. **注意:**写 1 清这个标志并恢复VBUS 错误状态.
[0]	**ROLECHGIF**	**OTG 角色改变中断状态** 当USB_ID 管脚没有改变时，OTG 设备由主机变为外设或由外设变为主机时此位置1. 0 = OTG 设备角色没有改变. 1 = OTG设备角色改变了. **注意:**写1清这个标志.

OTG_STATUS OTG 功能状态寄存器

寄存器	偏移量	R/W	描述	复位值
OTG_STATUS	OTG_BA+0x10	R	OTG 状态寄存器	0x0000_0006

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
ASHOST	ASPERI	VBUSVLD	AVLD	BVLD	SESSEND	IDSTS	OVERCUR

位	描述	
[31:8]	**Reserved**	保留.
[7]	**ASHOST**	**作为主机状态** 当 OTG 作为主机, 此位置1. 0: OTG 没有作为主机 1: OTG 作为主机
[6]	**ASPERI**	**作为外设状态** 当 OTG作为外设，此位置1. 0: OTG 没有作为外设 1: OTG 作为外设
[5]	**VBUSVLD**	**VBUS 有效状态** 当 VBUS高于 4.7V,此位置 1. 0 = VBUS还未有效. 1 = VBUS有效.
[4]	**AVLD**	**A-device 会话有效状态** 0 = A-device 会话无效. 1 = A-device 会话有效.
[3]	**BVLD**	**B-device 会话有效状态** 0 = B-device 会话无效. 1 = B-device会话有效.
[2]	**SESSEND**	**会话结束状态** 当 VBUS 电压低于 0.4V, t此位置 1. 会话结束意味这 VBUS上没有电压. 0 = 会话没有结束. 1 = 会话已结束.

位	描述	
[1]	**IDSTS**	**USB_ID Mini-b/Micro-plug 管脚状态** 0 = Mini-A/Micro-A 插入. 1 = Mini-B/Micro-B 插入.
[0]	**OVERCUR**	**过流状况** 当OTG A-device 开始驱动VBUS 为高之后，在最大时间 100ms后VBUS 的电平仍无法达到最低阈值4.4V时此位置1. 0 = OTG A-device 成功驱动VBUS. 1 = OTG A-device 在特定时间内无法拉高 VBUS

6.36 HSOTG 高速 USB OTG

6.36.1 概述

HSOTG 控制器连接USB PHY 和 USB 控制器，包含USB 2.0主机控制器和USB 2.0 高速设备控制器. OTG 控制器支持“On-The-Go and Embedded Host Supplement to the USB 2.0 Revision 1.3 Specification”定义的 HNP 和 SRP 协议.

USB 框架, 包括 USB 主机, USB 设备, 和 OTG 控制器, 可以配置为仅主机, 仅设备, 依据ID或HSUSBROLE (SYS_USBPHY[17:16])定义的OTG 设备模式。在仅主机模式下， USB 框架工作在USB 主机。 USB 框架都支持高速，全速和低速传输。在仅从机模式, USB 工作在 USB 设备. USB 框架支持高速和全速传输。在依据 ID模式, USB 框架依据USB_ID 管脚状态可以工作在 USB 主机和 USB 设备模式。在 OTG 设备模式, USB 框架的角色依据 OTG 标准的定义。USB 框架在OTG 设备作为外设时支持高速和全速传输。

6.36.2 特性

- 内建 USB PHY
- 可配置工作在:
  - 仅主机
  - 仅设备
  - 依据ID: USB 框架的角色仅依据USB_ID 管脚的值—作为 USB 主机 (USB_ID管脚为低)或USB 设备 (USB_ID 管脚为高). 不支持 HNP或SRP 协议
  - OTG 设备: 依据 USB_ID 管脚状态作为 A-device (USB_ID 管脚为低)或B-device (USB_ID 管脚为高). 支持 HNP 和 SRP 协议.

6.36.3 框图

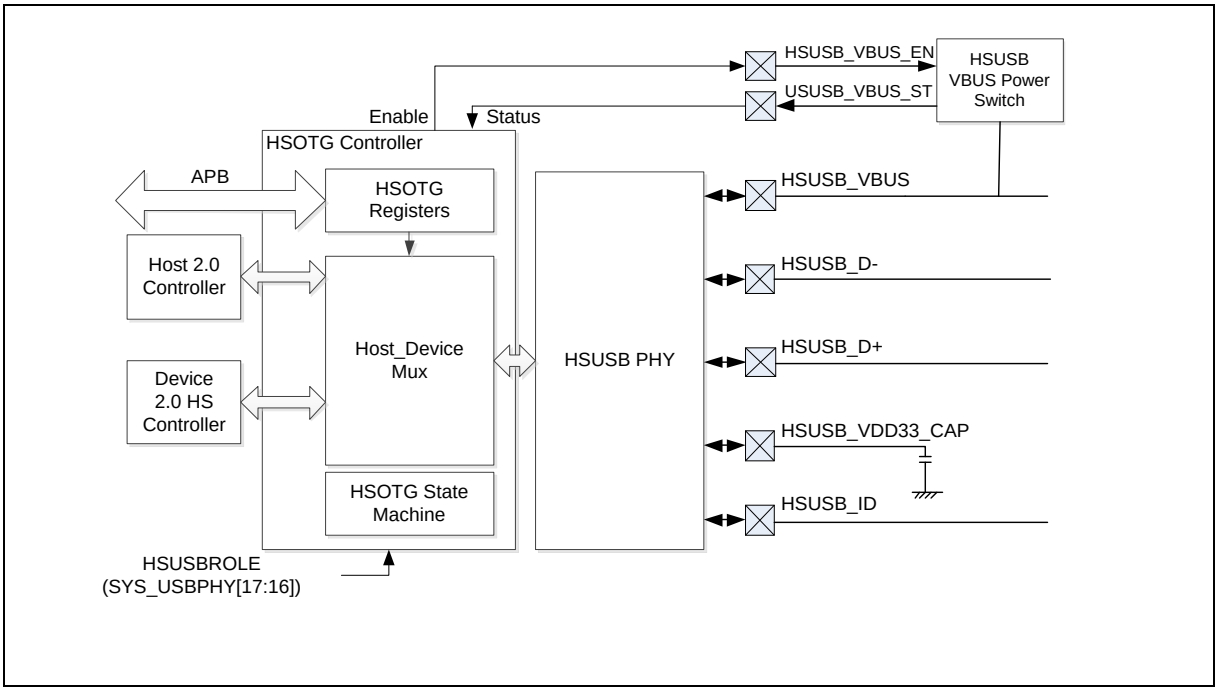


图6.36-1 USB OTG 框图

6.36.4 基本配置

- 时钟源配置
  - 在HSOTGCKEN(CLK_AHBCLK[30])使能HSOTG 时钟
- 复位配置
  - 在HSOTGRST (SYS_IPRST2[30])复位 HSOTG.

OTG 外设时钟可以由HSOTGCKEN (CLK_APBCLK0[30])使能. USB 框架的结构由 HSUSBROLE (SYS_USBPHY[17:16])来决定。这两个配置是写保护的，在写这些位之前，用户需要禁止寄存器保护功能。参考SYS_REGLCTL 寄存器的描述。 USB_VBUS_EN 和 USB_VBUS_ST 管脚功能由SYS_GPA_MFPL或SYS_GPC_MFPL 寄存器定义.

6.36.5 功能描述

USB 框架的角色由HSUSBROLE (SYS_USBPHY[17:16])和USB_ID 管脚状态来决定。 HSUSBROLE 的配置优先与USB_ID 管脚的状态。用户客户配置 OTG 控制器作为 USB 主机模式, USB 设备模式, 依据ID 模式或OTG 设备模式。在 USB 主机模式, 主机控制器直接与USB PHY 互动。在 USB 设备模式, t 设备控制器直接与 USB PHY 互动. 在这些情况下， OTG 控制器作为简单的多路复用器使用。在依据 ID 模式, USB_ID 管脚的状态决定 USB 帧充当 USB 主机或 USB 设备。如果USB_ID 管脚在 FALSE 状态 (低电平), USB 框架充当 USB 主机。如果 USB_ID 管脚在 TRUE 状态 (高电平), USB 帧充当 USB 设备。在OTG 设备模式, OTG 控制器会执行OTG HNP 和 SRP 协议，如果USB_ID 管脚是 FALSE 状态 (低电平), OTG 控制器充当 OTG A-device，如果 USB_ID 管脚是 TRUE 状态 (高电平), OTG 控制器充当 OTG B-device.

6.36.5.1 USB框架角色

USB 设备模式

当 USBROLE (SYS_USBPHY[1:0])为 0, USB 框架充当 USB 设备. USB主机功能不可用.

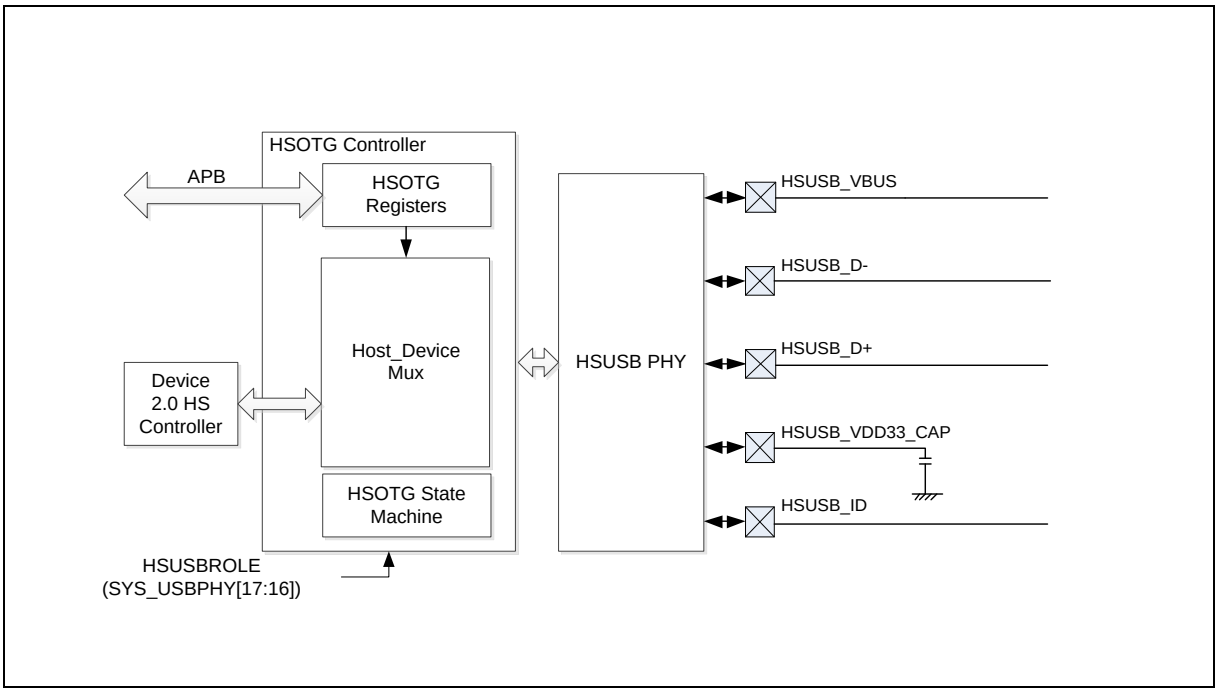


图 6.36-2 USB 设备模式

USB 主机模式

当 HSUSBROLE (SYS_USBPHY[17:16])为 1, USB 框架充当 USB 主机. USB 设备功能不可用

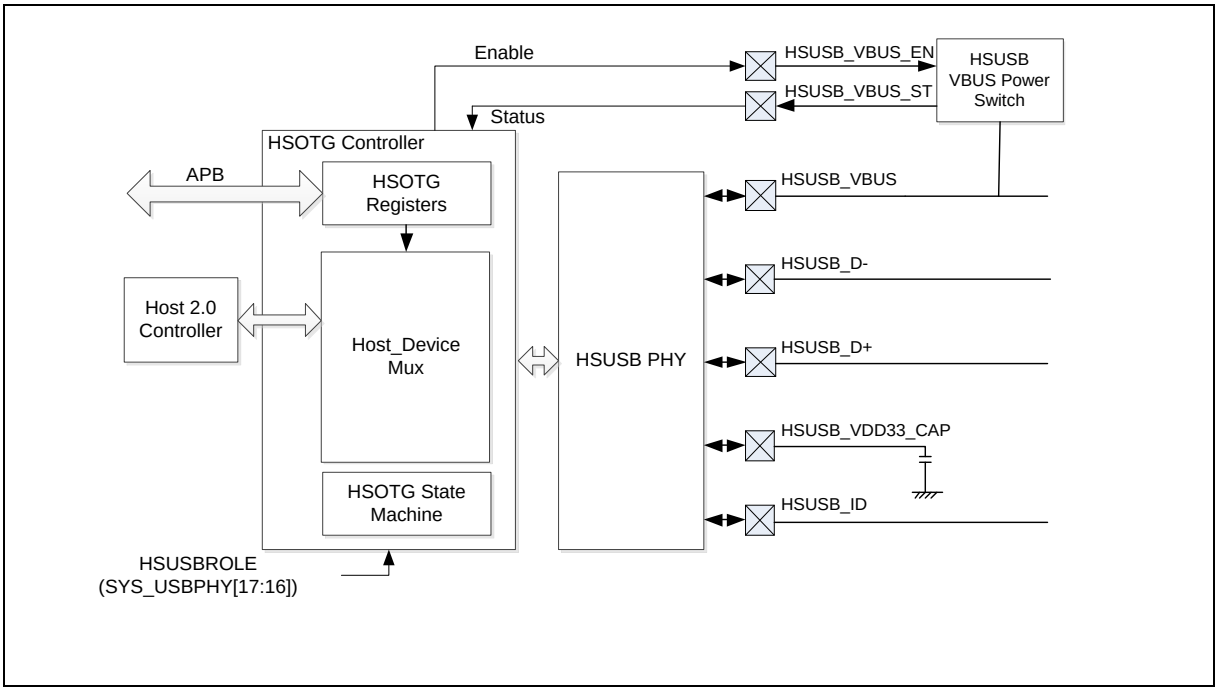


图 6.36-3 USB 主机模式

依据ID 模式

当 HSUSBROLE (SYS_USBPHY[17:16])设为 2, USB 框架的角色依据 USB_ID 管脚的状态. ID 检测功能由IDDETEN (HSOTG_PHYCTL[1]) 使能。 USB_ID 管脚状态影响IDSTS (HSOTG_STATUS[1]). 当 USB 框架充当USB 主机 (USB_ID 管脚低), 框图与 USB 主机模式相同。当 USB框架充当 USB 设备 (USB_ID管脚是高), 框图与 USB 设备模式相同。

OTG 设备模式

当 HSUSBROLE (SYS_USBPHY[17:16]) 等于 3, USB框架的角色取决于USB_ID 管脚的状态。ID 检测功能由IDDETEN (HSOTG_PHYCTL[1]) 使能。 USB_ID 管脚状态影响IDSTS (HSOTG_STATUS[1]). 当 USB_ID 管脚是低电平, OTG 控制器充当 OTG A-device. 当 USB_ID 管脚是高电平, OTG 控制器充当 OTG B-device. 请参考OTG 标准得到详细的A-device 和B-device的行为描述

6.36.5.2 会话请求协议(SRP)

当 USB 框架配置为OTG 设备模式, OTG 控制器支持 SRP来保护电源，参考OTG 标准得到SRP的详细描述.

A-Device 会话请求协议

1. A-device 关闭 USB 总线电源保护电源， B-device 通过检查 VBUS 的状态识别到这一状况。
2. 当 B-device 想要连接到 A-device，B-device 在数据线产生脉冲请求 A-device 提供 USB 总线电源
3. A-device 通过检查 SRPDETIF (HSOTG_INTSTS[13])认到 USB 总线电源请求
4. A-device 当 SRPDETIF (HSOTG_INTSTS[13])由硬件置1时通过设置 BUSREG (HSOTG_CTL[1]) 为1开始驱动 VBUS，如果 VBUS 在特定时间内达到有效电平，并且 B-device 已连接, A-device 会变为 USB 主机, HOSTIF (HSOTG_INTSTS[7]) 会设为 1. 如果 VBUS 在特定时间内不能达到有效电平，这意味着电流事件发生，那么 VBUS 错误位, VBEIF (HSOTG_IS[1]), 会设为 1.

B-device 会话请求协议

1. A-device 关闭 USB 总线电源保护电源. B-device 通过检查 VBUSVLD (HSOTG_STATUS[5])确认这一事件.
2. B-device 可以通过设定 BUSREQ(HSOTG_CTL[1])请求 A-device 提供 USB 总线电源.
3. B-device 会产生 OTG 定义的数据线脉冲
4. A-device 检测到数据线脉冲后会驱动 VBUS， B-device 可以通过确认 VBUSVLD (HSOTG_STATUS[5])识别这一状况。如果 A-device 在特定时间内驱动 VBUS 到有效电平, B-device 变为 USB 外设, PDEVIF (HSOTG_INTSTS[6]) 设为 1. 如果 A-device 没有在特定时间内驱动 VBUS 驱动到有效电平, SRP 错误标志, SRPFIF (HSOTG_INTSTS[2]), 会置 1 且 B-device 会进入 OTG 协议定义的空闲态.

6.36.5.3 主机谈判协议(HNP)

当USB 框架配置为 OTG 设备模式, 主机功能可以在两个直连的OTG 设备间切换，而不要改变接头连接，参考OTG 标准得到HNP的详细介绍.

A-device 主机谈判协议

1. OTG 协议定义的 A-Host 发送 SetFeature b_hnp_enable 指令使能 B-device HNP 功能. B-device 响应 ACK 表明 B-device 支持 HNP. 用户需要设定 HNPREQEN (HSOTG_CTL[2])为 1 使能 HNP 协议.
2. A-Host 完成所有必须的操作之后设置 BUSREQ (HSOTG_CTL[1])为0进入挂起状态，使 USB 总线进入 J-state (USB_D+ 高且 USB_D- 低)
3. A-Host 通过在特定的时间内检测 USB_D+ 和 USB_D- low 为低检测到 B-peripheral 断开连接，A-Host 变为 A-Peripheral。如果 A-Host 在特定时间内不能检测到 B-Peripheral 断开连接, A-Host 会回到空闲态.

B-device主机谈判协议

1. 当 B-Peripheral 成功接收到 SetFeature b_hnp_enable 指令, user 通过设定 HNPREQEN (HSOTG_CTL[2])为1使能 B-peripheral HNP 功能.
2. 在检测到 USB 总线在 J-state(USB_D+ 高且 USB_D- 低)，用户设置 BUSREG (HSOTG_CTL[1]) 为1. 接着 USB_D+ 上拉电阻移除导致 USB 断开连接状态 (USB_D+ 低且 USB_D- 低).
3. 如果 B-device 在特定时间内检测到 A-device 已连接 (USB_D+ 高), B-device 变为 B-Host. 如果 B-device 在特定时间内无法检测到 A-device 已连接 (USB_D+ 高), HNP 错误标志, HNPFIF (HSOTG_INTSTS[3]), 会设为 1

6.36.6 寄存器映射

R:只读, **W:**只写, **R/W:** 可读可写

寄存器	偏移量	R/W	描述	复位值
HSOTG 基地址: **HSOTG_BA = 0x4004_F000**				
HSOTG_CTL	HSOTG_BA+0x00	R/W	HSOTG 控制寄存器	0x0000_0000
HSOTG_PHYCTL	HSOTG_BA+0x04	R/W	HSOTG PHY 控制寄存器	0x0000_0000
HSOTG_INTEN	HSOTG_BA+0x08	R/W	HSOTG 中断使能寄存器	0x0000_0000
HSOTG_INTSTS	HSOTG_BA+0x0C	R/W	HSOTG 中断状态寄存器	0x0000_0000
HSOTG_STATUS	HSOTG_BA+0x10	R	HSOTG 状态寄存器	0x0000_0006

6.36.7 寄存器描述

HSOTG_CTL HSOTG 控制寄存器

寄存器	偏移量	R/W	描述	复位值
HSOTG_CTL	HSOTG_BA+0x00	R/W	HSOTG控制寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved		WKEN	OTGEN	Reserved	HNPREQEN	BUSREQ	VBUSDROP

位	描述	
[31:6]	**Reserved**	保留.
[5]	**WKEN**	**OTG ID 管脚唤醒使能位** 0 = OTG ID管脚状态改变唤醒功能禁止. 1 = OTG ID 管脚状态改变唤醒功能使能.
[4]	**OTGEN**	**OTG 功能使能位** 当USB框架被设为OTG设备时，用户需要设置此位使能OTG功能。如果没有配置成OTG设备，这里需要是0. 0= OTG 功能禁止. 1 = OTG 功能使能.
[3]	**Reserved**	保留.
[2]	**HNPREQEN**	**OTG HNP 请求使能位** 当USB框架作为 A-device, 设置此位当A-device 允许执行 HNP协议—A-device 改变角色从主机变为外设。当OTG状态从a_suspend变为a_peripheral或回到a_idle状态时此位会清0 当 USB框架作为 B-device, 设置此位在OTG A-device 成功发送SetFeature (b_hnp_enable) 指令到 OTG B-device开始角色转换后—B-device改变角色从外设到主机。这一位会在OTG 状态从b_peripheral 变为b_wait_acon 或回到 b_idle 状态后清0. 0 = HNP 请求禁止. 1 = HNP 请求使能 (A-device 改变角色从主机到设备， B-device 可以改变角色从外设到主机). **注意:**参考 OTG 协议了解 a_suspend, a_peripheral, a_idle 和 b_idle状态.
[1]	**BUSREQ**	**OTG 总线请求** 如果OTG A-device 通过USB总线传输数据, 设定此位驱动 VBUS位高检测 USB 设备连接. 如果用户不再想使用总线，清除此位拉低VBUS 来节省电源. 这一位当 A-device 变为 A_wait_vfall 态后清除. 当 VBUSDROP (OTG_CTL[0]) 置1或IDSTS (OTG_STATUS[1]) 改变后清除.

位	描述	
		如果OTG-B 设备想要请求VBUS, 设定这一位运行SRP协议. 这一位当SRP错误 (在B-device 在特定时间内执行OTG 协议定义的ARP ， OTG A-设备不提供VBUS)后清除。此位也在 VBUSDROP (OTG_CTL[0]) 位为1且 IDSTS (OTG_STATUS[1]) 改变后清除. 0 = OTG A-device 没有发动VBUS 或OTG B-device 没有请求SRP 1 = OTG A-device 发动VBUS 或OTG B-device 请求SRP.
[0]	**VBUSDROP**	**拉低 VBUS控制** 如果OTG A-device上的用户应用想要保护电源，设置这一位拉低VBUS. 不管是A-device或B-device BUSREQ (OTG_CTL[1])都会清除. 0 = 没有拉低 VBUS. 1 = 拉低 VBUS.

HSOTG_PHYCTL HSOTG PHY 控制寄存器

寄存器	偏移量	R/W	描述	复位值
HSOTG_PHYCTL	HSOTG_BA+0x04	R/W	HSOTG PHY 控制寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved		VBSTSPOL	VBENPOL	Reserved		IDDETEN	OTGPHYEN

位	描述	
[31:6]	**Reserved**	保留.
[5]	**VBSTSPOL**	**片外 USB VBUS 电源开关状态极性** 片外USB VBUS 电源开关信号的有效极性依据所选择的组件。USB_VBUS_ST 管脚用来监控外设 USB VBUS 电源开关的有效信号. 依据外部USB VBUS 电源开关的极性设置此位. 0 = 片外USB VBUS 电源开关信号的有效极性是高. 1 = 片外USB VBUS 电源开关信号的有效极性是低.
[4]	**VBENPOL**	**片外 USB VBUS 电源开关使能极性** OTG 控制器需要在需要时会使能片外USB VBUS 电源开关提供VBUS 电源. USB_VBUS_EN 管脚用来控制片外ρ USB VBUS 电源开关. 使能外部USB VBUS 电源开关的极性 (高有效或低有效) 依据所选择的器件。设置这一位选择合适的极性. 0 = 片外 USB VBUS 电源开关使能高有效. 1 = 片外 USB VBUS 电源开关使能低有效
[3:2]	**Reserved**	保留.
[1]	**IDDETEN**	**ID 检测使能位** 0 = 禁止检测 ID 管脚状态 1 = 使能检测 ID 管脚状态.
[0]	**OTGPHYEN**	**OTG PHY 使能** 当 USB 框架配置为OTG-device或ID-dependent, 用户需要在使用OTG 功能前设置此位。如果设备没有配置为 OTG-device 或ID-dependent , 此位不重要 0 = OTG PHY 禁止. 1 = OTG PHY 使能.

HSOTG_INTEN HSOTG 中断使能寄存器

寄存器	偏移量	R/W	描述	复位值
HSOTG_INTEN	HSOTG_BA+0x08	R/W	HSOTG中断使能寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved		SRPDETIEN	Reserved	SECHGIEN	VBCHGIEN	AVLDCHGIEN	BVLDCHGIEN
7	6	5	4	3	2	1	0
HOSTIEN	PDEVIEN	IDCHGIEN	GOIDLEIEN	HNPFIEN	SRPFIEN	VBEIEN	ROLECHGIEN

位	描述	
[31:14]	**Reserved**	保留.
[13]	**SRPDETIEN**	**SRP 检测中断使能位** 0 = 中断禁止. 1 = 中断使能.
[12]	**Reserved**	保留.
[11]	**SECHGIEN**	**SESSEND 状态改变中断使能位** 如果此位是 1 且 SESSEND (OTG_STATUS[2]) 状态改变，中断会产生 0 = 中断禁止. 1 = 中断使能.
[10]	**VBCHGIEN**	**VBUSVLD 状态改变中断使能位** 如果此位是 1 且 VBUSVLD (OTG_STATUS[5])状态改变，中断会产生 0 = 中断禁止. 1 = 中断使能.
[9]	**AVLDCHGIEN**	**A-device 会话有效状态改变中断使能位** 如果此位是 1 且 AVLD (OTG_STATUS[4])状态改变，中断会产生 0 = 中断禁止. 1 = 中断使能.
[8]	**BVLDCHGIEN**	**B-device 会话有效状态改变中断使能位** 如果此位是 1 且 BVLD (OTG_STATUS[3]) 状态改变，中断会产生. 0 = 中断禁止. 1 = 中断使能.
[7]	**HOSTIEN**	**充当主机中断使能位** 如果此位是 1 且设备作为主机，中断会产生.

位	描述	
		0 = 设备变主机中断禁止. 1 = 设备变主机终端使能.
[6]	**PDEVIEN**	**充当外设中断使能位** 如果此位为1，设备变为外设，中断会产生 0 = 设备变为外设中断禁止. 1 = 设备变为外设中断使能.
[5]	**IDCHGIEN**	**IDSTS 改变中断使能** 如果此位为 1 且 IDSTS (OTG_STATUS[1]) 状态改变，中断会产生. 0 = 中断禁止. 1 = 中断使能.
[4]	**GOIDLEIEN**	**OTG 设备变为 IDLE 状态中断使能位** 0 = 中断禁止. 1 = 中断使能. **注意:**进入空闲态意味着进入a_idle或b_idle 状态. 请参考OTG标准A-device 状态图和 B-device 状态图.
[3]	**HNPFIEN**	**HNP 失败中断使能位** 0 = 中断禁止. 1 = 中断使能.
[2]	**SRPFIEN**	**SRP 失败中断使能位** 0 = 中断禁止. 1 = 中断使能.
[1]	**VBEIEN**	**VBUS 错误中断使能位** 0 = 中断禁止. 1 = 中断使能. **注意:** VBUS 错误意味着进入a_vbus_err状态. 请参考OTG标准A-device 状态图.
[0]	**ROLECHGIEN**	**角色 (主机或外设) 改变中断使能位** 0 = 中断禁止. 1 = 中断使能.

HSOTG_INTSTS HSOTG 中断状态寄存器

寄存器	偏移量	R/W	描述	复位值
HSOTG_INTSTS	HSOTG_BA+0x0C	R/W	HSOTG 中断状态寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved		SRPDETIF	Reserved	SECHGIF	VBCHGIF	AVLDCHGIF	BVLDCHGIF
7	6	5	4	3	2	1	0
HOSTIF	PDEVIF	IDCHGIF	GOIDLEIF	HNPFIF	SRPFIF	VBEIF	ROLECHGIF

位	描述	
[31:14]	**Reserved**	保留.
[13]	**SRPDETIF**	**SRP 检测中断状态** 0 = SRP 没有检测到. 1 = SRP 检测到. **注意: 写1清除这个状态**.
[12]	**Reserved**	**保留**
[11]	**SECHGIF**	**SESSEND 状态改变中断状态** 0 = SESSEND (OTG_STATUS[2]) 没有翻转. 1 = SESSEND (OTG_STATUS[2]) 改变. **注意: 写1清这个状态**.
[10]	**VBCHGIF**	**VBUSVLD 状态改变中断状态** 0 = VBUSVLD (OTG_STATUS[5]) 没有翻转. 1 = VBUSVLD (OTG_STATUS[5]) 改变. **注意: 写1清这个状态**.
[9]	**AVLDCHGIF**	**A-device 会话有效状态改变中断状态** 0 = AVLD (OTG_STATUS[4]) 没有翻转. 1 = AVLD (OTG_STATUS[4]) 改变. **注意: 写1清这个状态**.
[8]	**BVLDCHGIF**	**B-device 会话有效状态改变中断状态** 0 = BVLD (OTG_STATUS[3]) 没有翻转. 1 = BVLD (OTG_STATUS[3])改变. **注意: 写1清这个状态**.
[7]	**HOSTIF**	**作为主机中断状态**

位	描述	
		0= 设备没有作为主机. 1 = 设备作为主机. **注意: 写1清这个状态**.
[6]	**PDEVIF**	**充当外设中断状态** 0= 设备没有充当外设. 1 = 这个设备充当外设. **注意: 写1清这个标志**.
[5]	**IDCHGIF**	**ID 状态改变中断状态** 0 = IDSTS (OTG_STATUS[1]) 没有翻转. 1 = IDSTS (OTG_STATUS[1]) 改变. **注意: 写1清这个标志**.
[4]	**GOIDLEIF**	**OTG 进入 IDLE 中断状态** 当OTG设备从非空闲变为空闲态此位置1， OTG设备可以是主机或者是外设I. 0 = OTG 设备没有回到空闲态 (a_idle或b_idle). 1 = OTG 设备回到空闲态(a_idle或b_idle). **注意 1:**进入空闲态就是进入a_idle或b_idle 状态. 请参考 OTG 规范. **注意 2:**写1清这个标志.
[3]	**HNPFIF**	**HNP 错误中断状态** 当 A-device 已授予 B-device作为主机，且 USB 总线在 SE0 (USB_D+ 和 USB_D-都是低)状态, 当 A-device 在期望时间内没有连接此位置1. 0 = A-device在特定时间内连上了 B-device. 1 = A-device 在特定时间内没有连上B-device. **注意:**写1清这个标志.
[2]	**SRPFIF**	**SRP 错误中断状态** 初始化 SRP之后，OTG B-device 会等待OTG A-device至少在OTG规范定义的TB_SRP_FAIL minimum的时间内拉高VBUS, 当 OTG B-device 在这个时间内没有检测到VBUS 拉高，这个标志置1. 0 = OTG B-device 在这时间内检测到 VBUS 变高. 1 = OTG B-device 没有在这时间内检测到 VBUS 变高. **注意:**写1清这个标志.
[1]	**VBEIF**	**VBUS 错误中断状态** 当OTG A-device 开始驱动VBUS 为高之后，在最大时间 100ms后VBUS 的电平仍无法达到最低阈值4.4V 时此位置1. 0 = OTG A-device 驱动 VBUS 在期望时间内达到阈值电压. 1 = OTG A-device 驱动 VBUS 在期望时间内没有达到阈值电压. **注意:**写 1 清这个标志并恢复VBUS 错误状态.
[0]	**ROLECHGIF**	**OTG 角色改变中断状态** 当USB_ID 管脚没有改变时，OTG 设备由主机变为外设或由外设变为主机时此位置1. 0 = OTG 设备角色没有改变. 1 = OTG设备角色改变了. **注意:**写1清这个标志.

HSOTG_STATUS HSOTG 功能状态寄存器

寄存器	偏移量	R/W	描述	复位值
HSOTG_STATUS	HSOTG_BA+0x10	R	HSOTG 状态寄存器	0x0000_0006

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
ASHOST	ASPERI	VBUSVLD	AVLD	BVLD	SESSEND	IDSTS	OVERCUR

位	描述	
[31:8]	**Reserved**	保留
[7]	**ASHOST**	**作为主机状态** 当 OTG 作为主机, 此位置1. 0: OTG 没有作为主机 1: OTG 作为主机
[6]	**ASPERI**	**作为外设状态** 当 OTG作为外设，此位置1. 0: OTG 没有作为外设 1: OTG 作为外设
[5]	**VBUSVLD**	**VBUS 有效状态** 当 VBUS高于 4.7V,此位置 1. 0 = VBUS还未有效. 1 = VBUS有效.
[4]	**AVLD**	**A-device 会话有效状态** 0 = A-device 会话无效. 1 = A-device 会话有效.
[3]	**BVLD**	**B-device 会话有效状态** 0 = B-device 会话无效. 1 = B-device会话有效.
[2]	**SESSEND**	**会话结束状态** 当 VBUS 电压低于 0.4V, t此位置 1. 会话结束意味这 VBUS上没有电压. 0 = 会话没有结束. 1 = 会话已结束.

位	描述	
[1]	**IDSTS**	**USB_ID Mini-b/Micro-plug 管脚状态** 0 = Mini-A/Micro-A 插入. 1 = Mini-B/Micro-B 插入.
[0]	**OVERCUR**	**过流状况** 当OTG A-device 开始驱动VBUS 为高之后，在最大时间 100ms后VBUS 的电平仍无法达到最低阈值4.4V时此位置1. 0 = OTG A-device 成功驱动VBUS. 1 = OTG A-device 在特定时间内无法拉高 VBUS

6.37 CRC 控制器

6.37.1 概述

CRC循环冗余发生器使用4种常见的多项式CRC-CCITT, CRC-8, CRC-16, 和 CRC-32 执行 CRC 计算

6.37.2 特性

- 支持4种常见的多项式 CRC-CCITT, CRC-8, CRC-16, 和 CRC-32
  - CRC-CCITT: $X^{16} + X^{12} + X^5 + 1$
  - CRC-8: $X^8 + X^2 + X + 1$
  - CRC-16: $X^{16} + X^{15} + X^2 + 1$
  - CRC-32: $X^{32} + X^{26} + X^{23} + X^{22} + X^{16} + X^{12} + X^{11} + X^{10} + X^8 + X^7 + X^5 + X^4 + X^2 + X + 1$
- 可编程种子值
- 对于输入数据和CRC 校验和，支持可编程的位顺序反转设定
- 对于输入数据和CRC 校验和，支持可编程的补码设定
- 支持 8/16/32-bit 数据宽度
  - 8-bit 写模式: 1-AHB 时钟周期操作
  - 16-bit 写模式: 2-AHB 时钟周期操作
  - 32-bit 写模式: 4-AHB 时钟周期操作
- 支持使用 PDMA 写数据执行 CRC 操作

6.37.3 框图

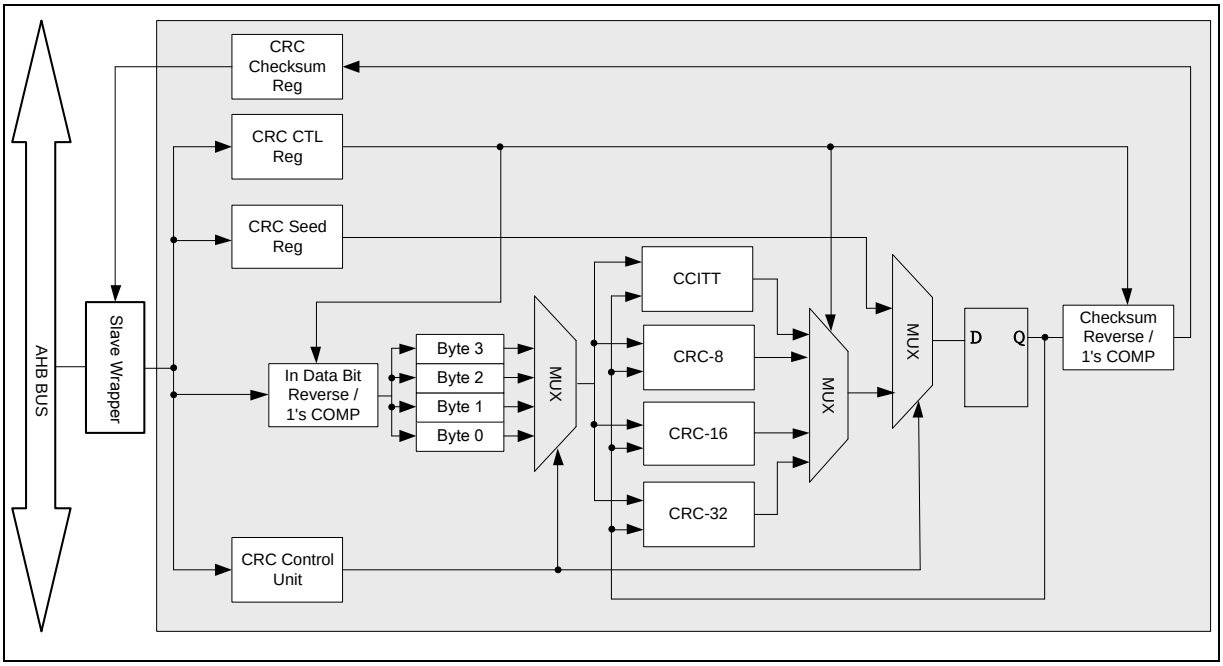


图 6.37-1CRC 发生器框图

6.37.4 基本配置

- 时钟源配置
  - 在CRCCKEN (CLK_AHBCLK[7])使能CRC 外设时钟
- 复位配置
  - 在CRCRST (SYS_IPRST0[7])复位CRC 控制器.

6.37.5 功能描述

CRC 计算可选择4中常见多项式：CRC-CCITT, CRC-8, CRC-16 和 CRC-32；通过 CRCMODE[1:0] (CRC_CTL[31:30])选择 CRC多项式

下面是编程流程示例：

1. 设置CRCEN (CRC_CTL[0])使能CRC 发生器
   1) CRC计算的初始化设定
2. 初始化计算设定.
   1) 设置CHKSFMT (CRC_CTL[27])配置CRC 校验和补码.

      设置 CHKSREV (CRC_CTL[25])配置CRC校验和位顺序反转见图6.37-2
   2) 配置DATFMT (CRC_CTL[26])配置CRC 写数据补码
   3) 设置DATREV (CRC_CTL[24])配置 CRC 写数据每个字节的位顺序反转，功能框图见图 6.37-3.
3. 执行 CHKSINIT (CRC_CTL[1]) 从CRC_SEED寄存器中载入检验和初值.
4. 写数据到 CRC_DAT 寄存器计算 CRC 校验和.
5. 读CRC_CHECKSUM 寄存器得到CRC 校验和结果

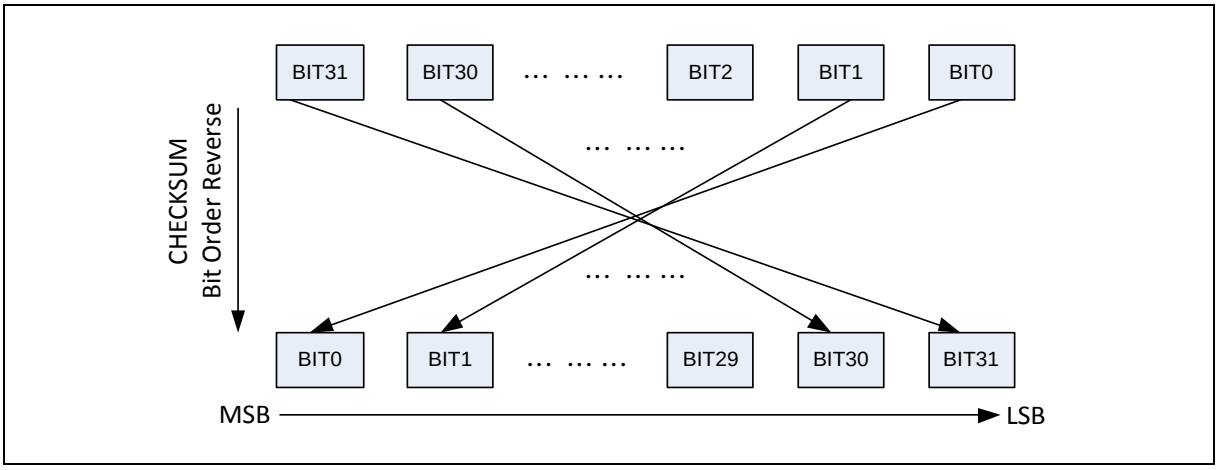


图 6.37-2 校验和位顺序反转功能框图

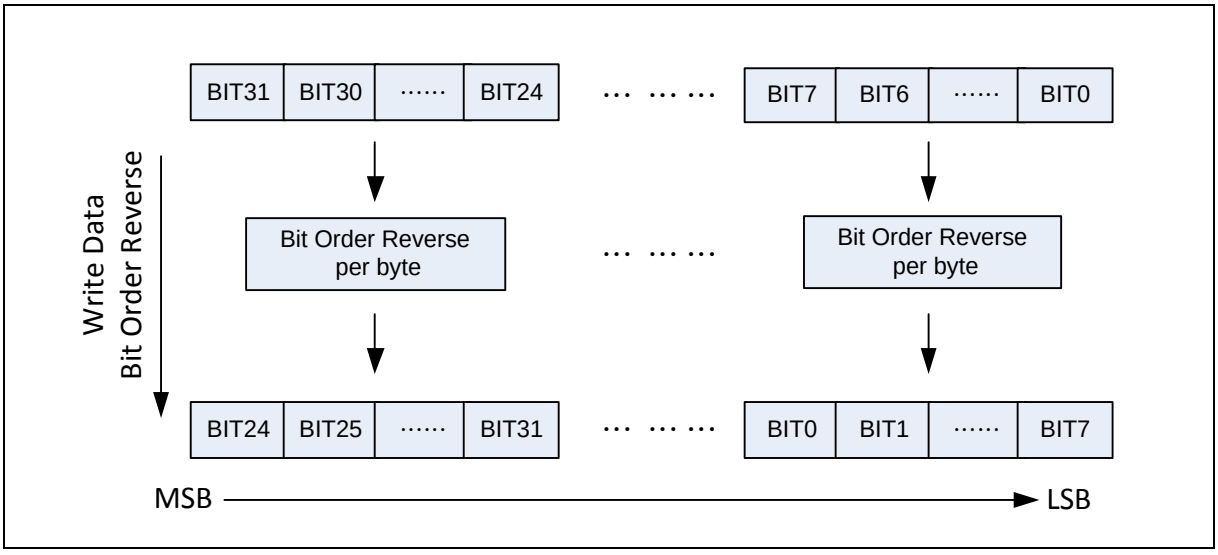


图 6.37-3 写数据位顺序反转功能框图

6.37.6 寄存器映射

R:只读, **W:**只写, **R/W:** 可读可写

寄存器	偏移量	R/W	描述	复位值
CRC 基地址: **CRC_BA = 0x4003_1000**				
CRC_CTL	CRC_BA+0x00	R/W	CRC 控制寄存器	0x2000_0000
CRC_DAT	CRC_BA+0x04	R/W	CRC写数据寄存器	0x0000_0000
CRC_SEED	CRC_BA+0x08	R/W	CRC 种子寄存器	0xFFFF_FFFF
CRC_CHECKSUM	CRC_BA+0x0C	R	CRC 校验和寄存器	0xFFFF_FFFF

6.37.7 寄存器描述

CRC_CTL CRC 控制寄存器

寄存器	偏移量	R/W	描述	复位值
CRC_CTL	CRC_BA+0x00	R/W	CRC控制寄存器	0x2000_0000

31	30	29	28	27	26	25	24
CRCMODE		DATLEN		CHKSFMT	DATFMT	CHKSREV	DATREV
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved						CHKSINIT	CRCEN

位	描述	
[31:30]	**CRCMODE**	**CRC 多项式模式** 这里选择 CRC 多项式模式. 00 = CRC-CCITT多项式模式. 01 = CRC-8多项式模式. 10 = CRC-16多项式模式. 11 = CRC-32多项式模式.
[29:28]	**DATLEN**	**CPU 写数据长度** 这里设置写数据长度. 00 = 数据长度是 8-bit模式. 01 = 数据长度是 16-bit 模式. 1x = 数据长度是 32-bit 模式. **注意:**当写数据长度是8-bit 模式， CRC_DAT 寄存器的有效数据是DATA[7:0]位; 如果写数据长度是 16-bit 模式, CRC_DAT寄存器的有效数据是 DATA[15:0].
[27]	**CHKSFMT**	**校验和补码** 0 = 禁止 CRC校验和补码. 1 = 使能 CRC 校验和补码.
[26]	**DATFMT**	**写数据补码** 0 = 禁止 CRC 写入数据做补码. 1 = 使能CRC 写入数据做补码.
[25]	**CHKSREV**	**校验和位顺序反转**. 0 = 禁止 CRC 校验和位顺序反转. 1 = 使能CRC 校验和位顺序反转. **注意:**如果校验和结果是0xDD7B0F2E, 位顺序反转的CRC 校验和是0x74F0DEBB.

[24]	**DATREV**	**写数据位顺序反转** 这一位用来使能写入CRC_DAT 寄存器的值位顺序反转. 0 = CRC 写入数据位顺序反转禁止. 1 = CRC 写入数据位顺序反转使能 (按字节看). **注意:** 如果写入数据是0xAABBCCDD,位顺序反转过的 CRC 写入数据是 0x55DD33BB.
[23:2]	**Reserved**	保留.
[1]	**CHKSINIT**	**校验和初始化** 0 = 没影响. 1 = 初始化校验和的值，载入 CRC_SEED 寄存器的值到CRC_CHECKSUM 寄存器. **注意:** 这一位会自动清除
[0]	**CRCEN**	**CRC 操作使能位** 0 = 没影响 1 = CRC 操作使能.

CRC_DAT CRC 写数据寄存器

寄存器	偏移量	R/W	描述	复位值
CRC_DAT	CRC_BA+0x04	R/W	CRC写数据寄存器	0x0000_0000

31	30	29	28	27	26	25	24
DATA							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
DATA							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
DATA							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
DATA							

位	描述	
[31:0]	**DATA**	**CRC 写数据位** 可以CPU直接写入，或者使用PDMA功能写数据到这个寄存器执行CRC操作 **注意:**当数据长度是8-bit 模式， CRC_DAT 寄存器的有效数据是DATA[7:0]位; 如果写数据长度是 16-bit 模式, CRC_DAT寄存器的有效数据是 DATA[15:0].

CRC_SEED CRC 种子寄存器

寄存器	偏移量	R/W	描述	复位值
CRC_SEED	CRC_BA+0x08	R/W	CRC 种子寄存器	0xFFFF_FFFF

31	30	29	28	27	26	25	24
SEED							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
SEED							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
SEED							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
SEED							

位	描述	
[31:0]	**SEED**	**CRC 种子值** 这里是 CRC 种子值. **注意:** 这里在执行CHKSINIT (CRC_CTL[1])后会重载作为校验和的初始值 (CRC_CHECKSUM 寄存器)

CRC_CHECKSUM CRC 校验和寄存器

寄存器	偏移量	R/W	描述	复位值
CRC_CHECKSUM	CRC_BA+0x0C	R	CRC校验和寄存器	0xFFFF_FFFF

31	30	29	28	27	26	25	24
CHECKSUM							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
CHECKSUM							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
CHECKSUM							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
CHECKSUM							

位	描述	
[31:0]	**CHECKSUM**	**CRC 校验和结果(只读)** 这里是CRC校验和结果

6.38 CRYPTO 加解密算法加速器

6.38.1 概述

支持 AES, DES/TDES, SHA，HMAC 和 ECC 算法，包含一个伪随机数生成器(PRNG)，可产生 64 位, 128 位, 192 位, 和 256 位随机数。

AES引擎支持ECB、CBC、CFB、OFB、 CTR、CBC-CS1、CBC-CS2、和 CBC-CS3 模式。

DES/TDES引擎支持ECB、CBC、CFB，OFB和CTR模式。

SHA引擎完全兼容SHA -160、SHA-224、SHA-256和相应的 HMAC 算法

ECC 加速器在二元域和素数域中用多项式基实现完全符合椭圆曲线密码体制

6.38.2 特性

- PRNG
  - 支持 64位, 128 位 , 192 位, 和 256位随机数的产生
- AES
  - 支持 FIPS NIST 197
  - 支持 SP800-38A 和 addendum
  - 支持 128, 192, 和 256 位密钥
  - 支持 ECB, CBC, CFB, OFB, CTR, CBC-CS1, CBC-CS2, 和 CBC-CS3 模式
  - 支持密钥扩展
- DES
  - 支持 FIPS 46-3
  - 支持加密和解密
  - 支持 ECB, CBC, CFB, OFB, 和 CTR 模式
- TDES
  - 支持 FIPS NIST 800-67
  - 根据 X9.52 标准执行
  - 支持双密钥或三密钥模式
  - 支持加密和解密
  - 支持 ECB, CBC, CFB, OFB, 和 CTR 模式
- SHA
  - 支持 FIPS NIST 180, 180-2
  - 支持 SHA-160, SHA-224, SHA-256, SHA-384, 和 SHA-512
- HMAC
  - 支持 FIPS NIST 180, 180-2
  - 支持 HMAC-SHA-160, HMAC-SHA-224, HMAC-SHA-256, HMAC-SHA-384, 和 HMAC-

SHA-512

- ECC
  - 同时支持素数域 GF(p) 和二进制域 GF(2^m)
  - 支持 NIST P-192, P-224, P-256, P-384, 和 P-521
  - 支持 NIST B-163, B-233, B-283, B-409, 和 B-571
  - 支持 NIST K-163, K-233, K-283, K-409, 和 K-571
  - 在GF(p) 和 GF(2^m)支持点乘法，加法和加倍运算在 GF(p) 和 GF(2^m)
  - 在GF(p) 支持模划分，乘法，加法和减法运算

引擎	模式	M48xID/M48xGA	M48xGC/M48xE8
PRNG		●	●
AES		●	●
DES/TDES		●	
SHA/HMAC	SHA-160	●	
	SHA-224	●	
	SHA-256	●	
	SHA-384	●	
	SHA-512	●	
ECC	P-192/224/256	●	
	P-384/521	●	
	B-163/233	●	
	B-283/409/571	●	
	K-163/233	●	
	K-283/409/571	●	
	Curve25519		
	Side-channel attack protection		

表 6.38-1 不同芯片加密特性比较表

6.38.3 框图

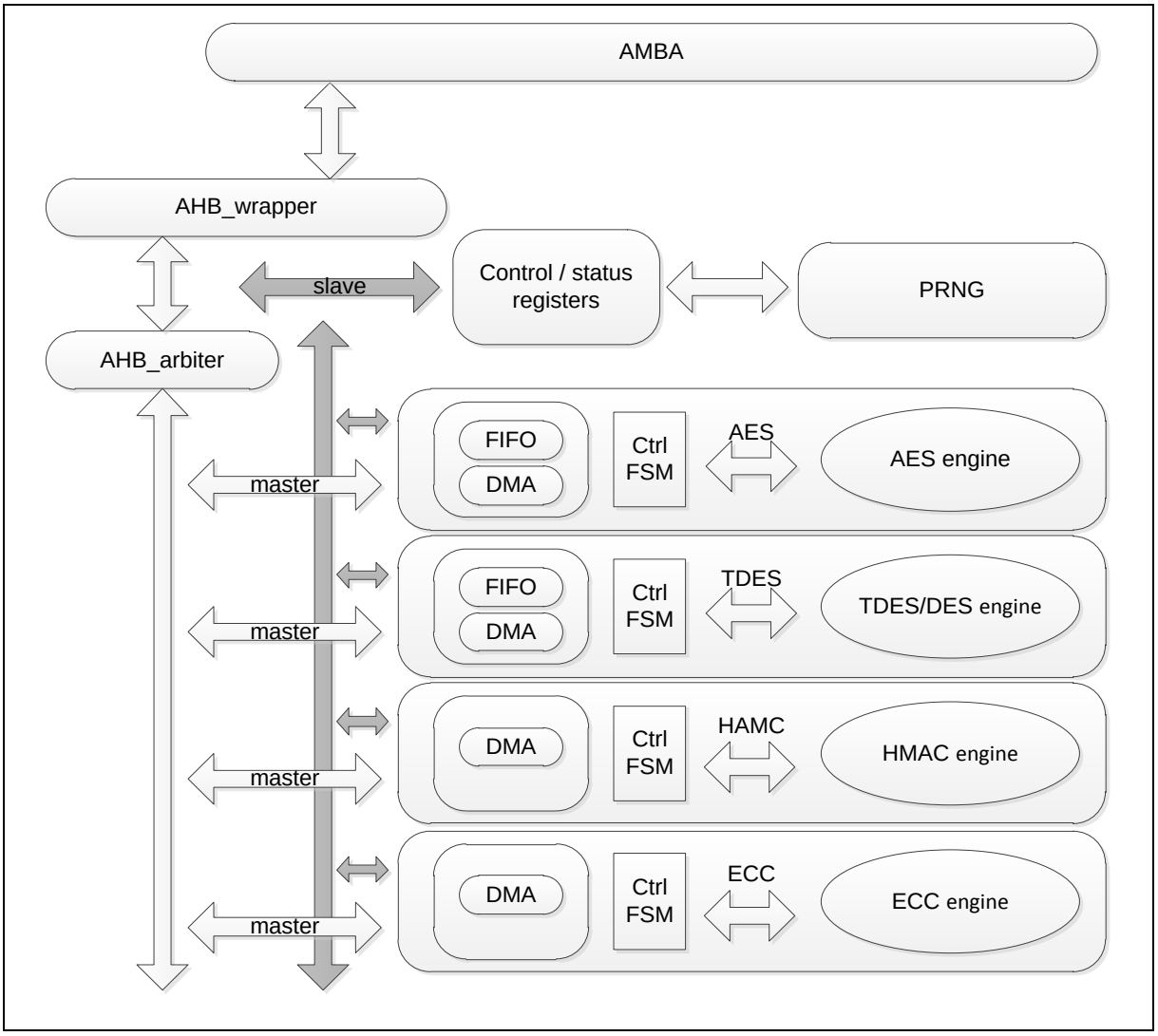


图 6.38-1 加解算法加速器框图

6.38.4 基本配置

- 时钟源使能
  - CRYPTO时钟源使能位：CRPTCKEN (CLK_AHBCLK[12])
- 复位使能
  - 复位CRYPTO 控制在CRPTRST (SYS_IPRST0[12])

6.38.5 功能描述

CRYPTO加解算法加速器包含一个安全的伪随机数生成器（PRNG）内核，支持AES、DES/TDES、SHA和ECC的算法。该引擎可用于不同的数据安全应用，例如需要加密保护和完整性的安全通信。

1. PRNG 内核依据支持 KEYSZ(CRYPTO_PRNG_CTL[3:2])的设定，支持 64 位, 128 位, 192 位, 以及 256 位随机数的生成.
2. AES 加速器完全兼容实现 AES (高级加密标准) 加解密的算法。. AES 加速器支持 ECB, CBC, CFB, OFB, CTR, CBC-CS1, CBC-CS2, 和 CBC-CS3 模式. AES 加速器提供 DMA 功能降低 CPU 负担, 并且支持 3 种突发长度, 16 字, 8 字, 和 4 字.
3. DES/TDES 加速器完全兼容实现 DES 和三倍 DES 加解密算法. DES/TDES 加速器支持 ECB, CBC, CFB, OFB,和 CTR 模式. DES/TDES 也支持 DMA 功能降低 CPU 负担. 支持两种突发长度, 4 字和 8 字。5
4. SHA/HMAC 加速器完全兼容实现 SHA-160, SHA-224, SHA-256, SHA-384, SHA-512, 和相应的 HMAC 算法。SHA/HMAC 加速器也支持 DMA 功能降低 CPU 参与. 支持 3 种, 16 字，8 字和 4 字.
5. ECC 加速器完全兼容实现素数域 GF(p) 和二进制域 GF(2^m) 算法. 素数域 GF(p) 支持 NIST P-192, P-224, P-256, P-384 ， P-521 及曲线 25519。二进制域 GF(2^m)支持 NIST B-163, B-233, B-283, B-409, B-571 和 NIST K-163, K-233, K-283, K-409 和 K-571.

程序可以使能CRYPTO_INTEN 控制数据流，检查CRYPTO_INTSTS 监控加速器状态。当任意引擎发生操作错误或缓存错误，相应的错误标识会置1，如果中断使能会通知CPU。如果想知道错误的详细信息，程序可以确认每个引擎的状态标志寄存器。表6.38-2列出所有引擎错误使能位，错误标志位和错误条件.

引擎	错误中断使能	错误中断标志	错误条件
AES	AESEIEN (CRYPTO_INTEN[1])	AESEIF (CRYPTO_INTSTS[1])	INBUFERR (CRYPTO_AES_STS[10]) OUTBUFERR (CRYPTO_AES_STS[18]) BUSERR (CRYPTO_AES_STS[20])
TDES/DES	TDESEIEN (CRYPTO_INTEN[9])	TDESEIF (CRYPTO_INTSTS[9])	INBUFERR (CRYPTO_TDES_STS[10]) OUTBUFERR (CRYPTO_TDES_STS[18]) BUSERR (CRYPTO_TDES_STS[20])
SHA/HMAC	HMACEIEN (CRYPTO_INTEN[25])	HMACEIF (CRYPTO_INTSTS[25])	DMAERR (CRYPTO_HMAC_STS[8])
ECC	ECCEIEN (CRYPTO_INTEN[23])	ECCEIF (CRYPTO_INTSTS[23])	BUSERR (CRYPTO_ECC_STS[16])

表 6.38-2 每个引擎错误条件和错误标识

加解算法加速器支持下列特性提升性能

DMA 模式

当CPU配置好了DMA 源地址寄存器, 目的地址寄存器, 和字节数寄存器，对加速器搬运数据完全由DMA逻辑完成. 这个模式可以降低 CPU的负担. 加解算法加速器为AES引擎内建1个硬件DMA信道, DES/TDES引擎有1个硬件DMA 信道, SHA/HMAC 引擎有1个硬件DMA 通道.

引擎	DMA 使能位
AES	DMAEN (CRYPTO_AES_CTL[7])
TDES/DES	DMAEN (CRYPTO_TDES_CTL[7])
SHA/HMAC	DMAEN (CRYPTO_HMAC_CTL[7])
ECC	DMAEN (CRYPTO_ECC_CTL[7])

表 6.38-3 DMA 使能位列表

DMA 串流模式

当 SRAM 内存不够使用，或者是其他周边装置等待处理的情况下，可以采用分次串流处理的方式来完成加解密操作。在 DMA 串流模式下，不需要一次完成加密操作，应用程序可以加解密一段数据，更新 DMA 来源、目的地址、及数据量计数，然后再接续加解密操作。过程中，AES硬件会记住当前状态，应用程序可以加解密一段数据，去服务其他周边装置，然后再回来继续加解密操作，直到全部数据处理完成。另一个好处是，内存的需求也会大幅度地降低。

引擎	DMA 串流位
AES	DMACSCAD (CRYPTO_AES_CTL[6])
TDES/DES	DMACSCAD (CRYPTO_TDES_CTL[6])
SHA/HAMC	DMACSCAD (CRYPTO_HMAC_CTL[6])

表 6.38-4 DMA 串流位表

无 DMA 模式

输入数据量比较小的情况，DMA模式是不可取的。这种非DMA模式可以减少对引擎的处理时间，因为没有DMA相关的寄存器需要配置，且没有引入DMA逻辑的延迟。输入数据通过写数据输入寄存器传入到加密引擎。

信道扩展模式

在这种模式下，在AES或DES/TDES模式，多个虚拟信道用四个DMA信道的其中一个是可行的。总的信道数可以超过四个DMA通道的限制。来自反馈寄存器 (CRYPTO_AES_FDBCKx, CRYPTO_TDES_FDBCKH, 和 CRYPTO_TDES_FDBCKL) 的中间数据应该暂时储存在数据SRAM。并切换到另一个引擎操作的配置设置，包括操作模式，加密/解密，密钥，密钥大小，IV，和其他参数。一旦切换回来，来自反馈寄存器的中间数据应写入初始向量 (CRYPTO_AESn_IVx, CRYPTO_TDESx_IVH, 和 CRYPTO_TDESx_IVL) 用于引擎带初始配置设定继续操作。请注，在ECB模式，没有必要将中间数据从反馈寄存器移到IV。

引擎	通道选择位
AES	CHANNEL (CRYPTO_AES_CTL[25,24])
TDES/DES	CHANNEL (CRYPTO_TDES_CTL[25,24])

表 6.38-5 通道选择位表

6.38.5.2 PRNG (伪随机数生成器)

PRNG 框图见图 6.38‑2. 内核支持 64 位，128 位，192 位，和 256 位随机数的生成，由 KEYSZ(CRYPTO_PRNG_CTL[3:2])配置。

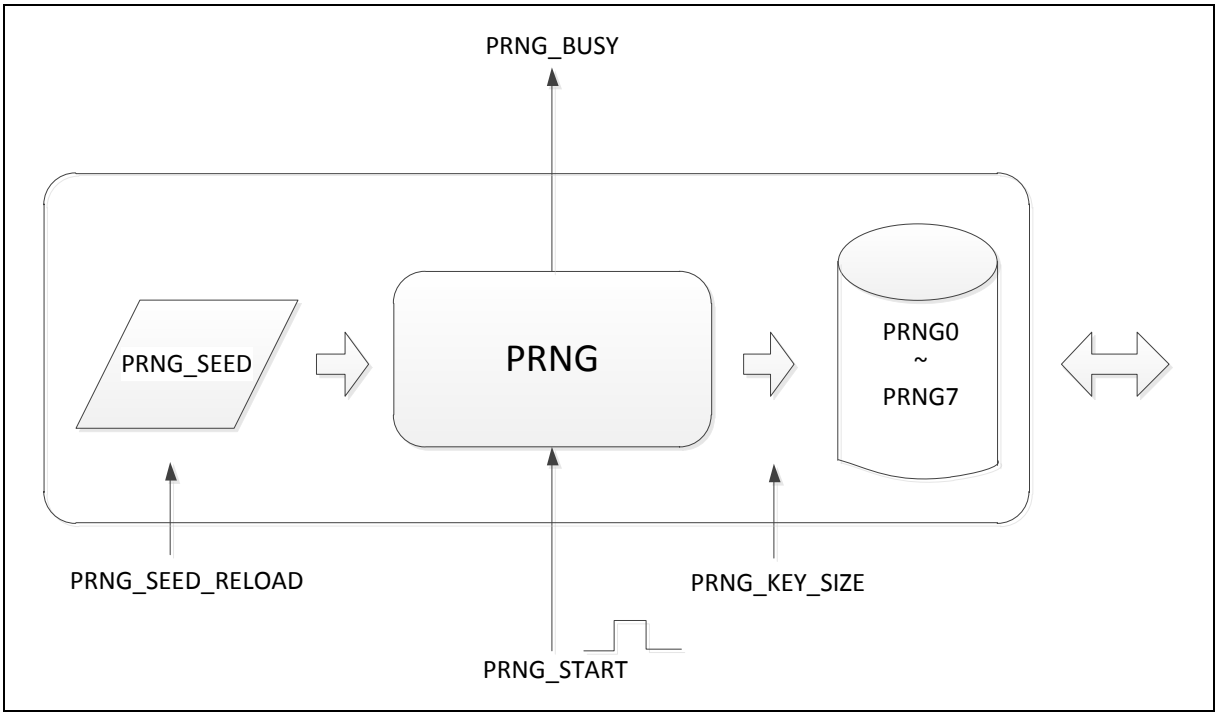


图 6.38‑2 PRNG 功能框图

得到一个伪随机数的编程步骤如下描述

1. 检查 BUSY(CRYPTO_PRNG_CTL[8]) 直到它变 0.
2. 初始化 PRNG 参数. 配置KEYSZ (CRYPTO_PRNG_CTL[3:2]), 写一个随机种子到 CRYPTO_PRNG _SEED. 注意CRYPTO_PRNG_SEED 需要初始化，因为芯片上电并不会初始化这里.
3. 配置 PRNG 控制寄存器KEYSZ(CRYPTO_PRNG_CTL[3:2]), 种子重加载 SEEDRLD(CRYPTO_PRNG_CTL[1]), 和START(CRYPTO_PRNG_CTL[0])
4. 软件检查 BUSY(CRYPTO_PRNG_CTL[8]) 直到变成 0, 或是等待 PRNGIF (CRYPTO_INTSTS[16]) (必须使能 PRNGIEN (CRYPTO_INTEN[16])). 然后软件从 CRYPTO_PRNG_KEY0 ~ CRYPTO_PRNG_KEY7中读出产生的随机数（KEY）

6.38.5.3 AES (高级加密标准)

电子密码本模式

电子密码本（ECB）模式是一种保密模式，特征是一个固定的密文块分配给每个明文块，对于一个给定的密匙。它与码本中的码字分配类似。ECB加密，每一组明文块都直接独立地应用到了正向密码函数 $CIPH_k$。产生的序列输出块是密文。ECB解密，每一组密文块都直接独立地应用到了反向密码函数

$CIPH^{-1}_k$。产生的序列的输出块是明文。

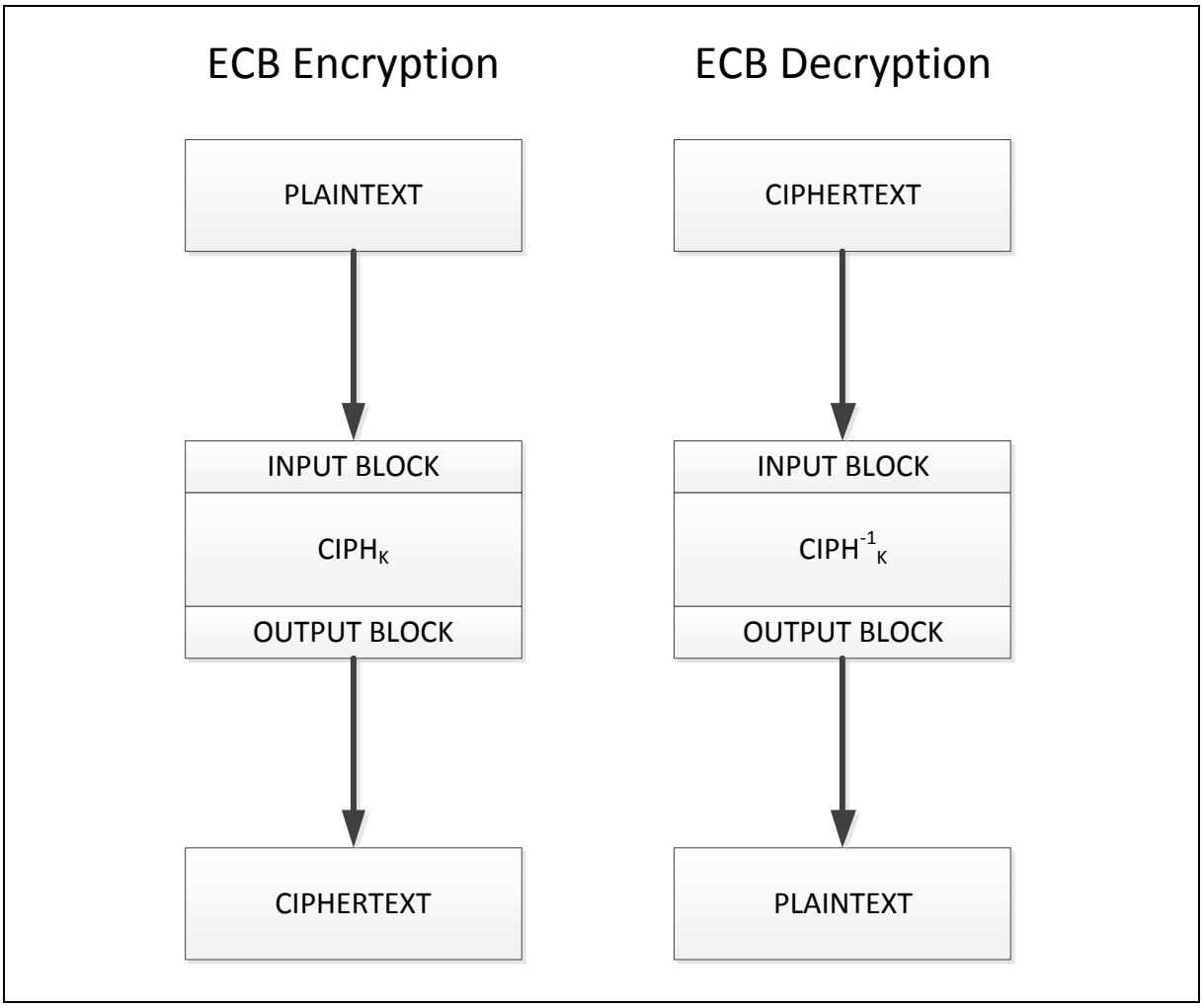


图 6.38-3 电子密码本模式

在ECB模式，任何给定的明文块总是在一个给定的密匙下被加码成密文块的。如果在特定的应用中这个属性是不可行的，则不应该使用ECB模式。

密码块链模式

密码块链（CBC）模式是一种保密模式，它的加密处理特点结合明文块链和先前的密文块。CBC模式需要一个初始化向量（IV）来结合第一个明文块。IV不需要保密，但它必须是不可预测的。

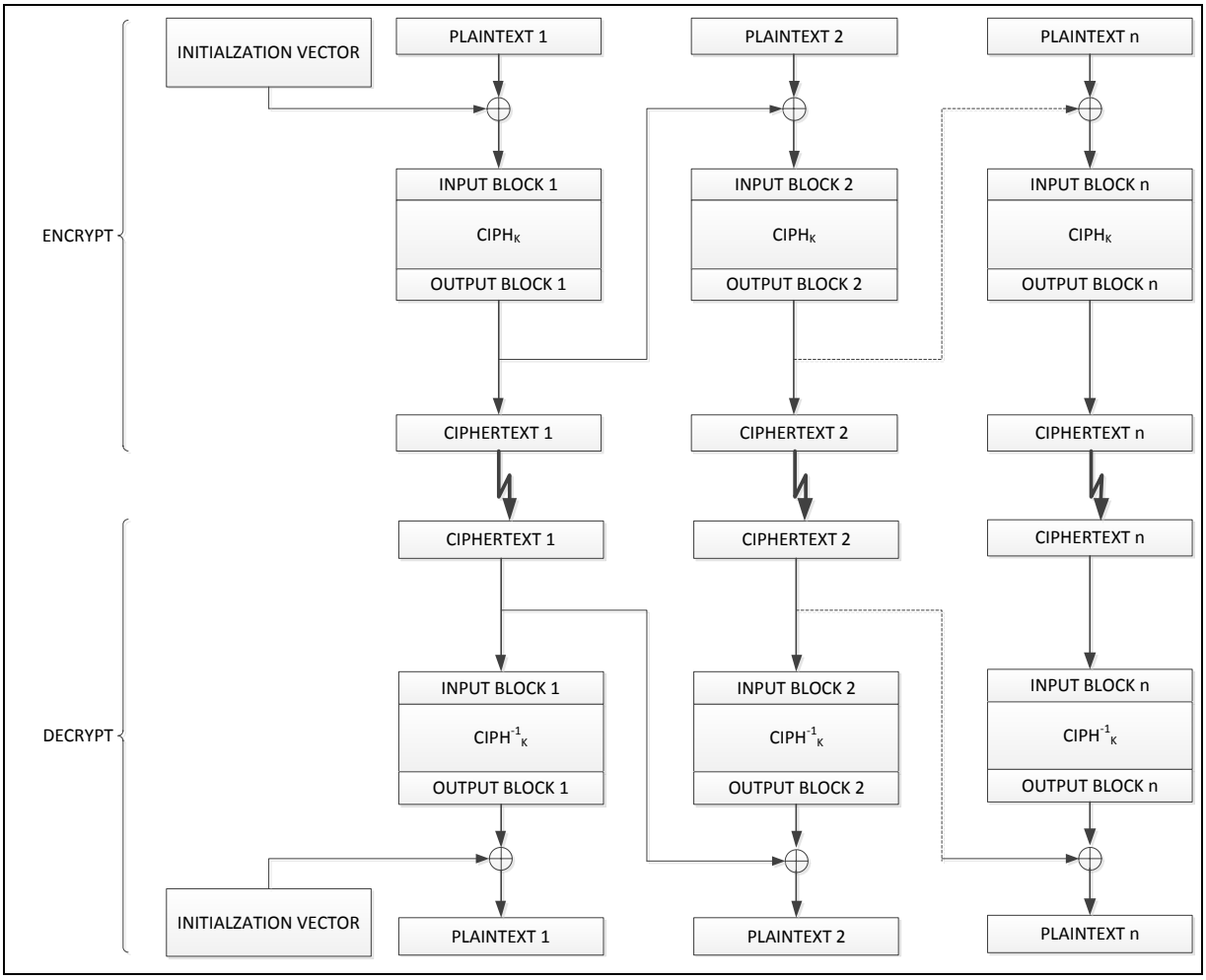


图 6.38-4 密码块链模式

密码反馈模式

密码反馈（CFB）模式是一种保密模式，特征是连续密文段反馈到了正向密码输入块来生成输出块，这是与明文进行异或运算产生密文，反之亦然。CFB模式需要一个IV作为初始输入块，IV不需要保密，但它必须是不可预测的。CFB模式下AES仅支持128bit段长度

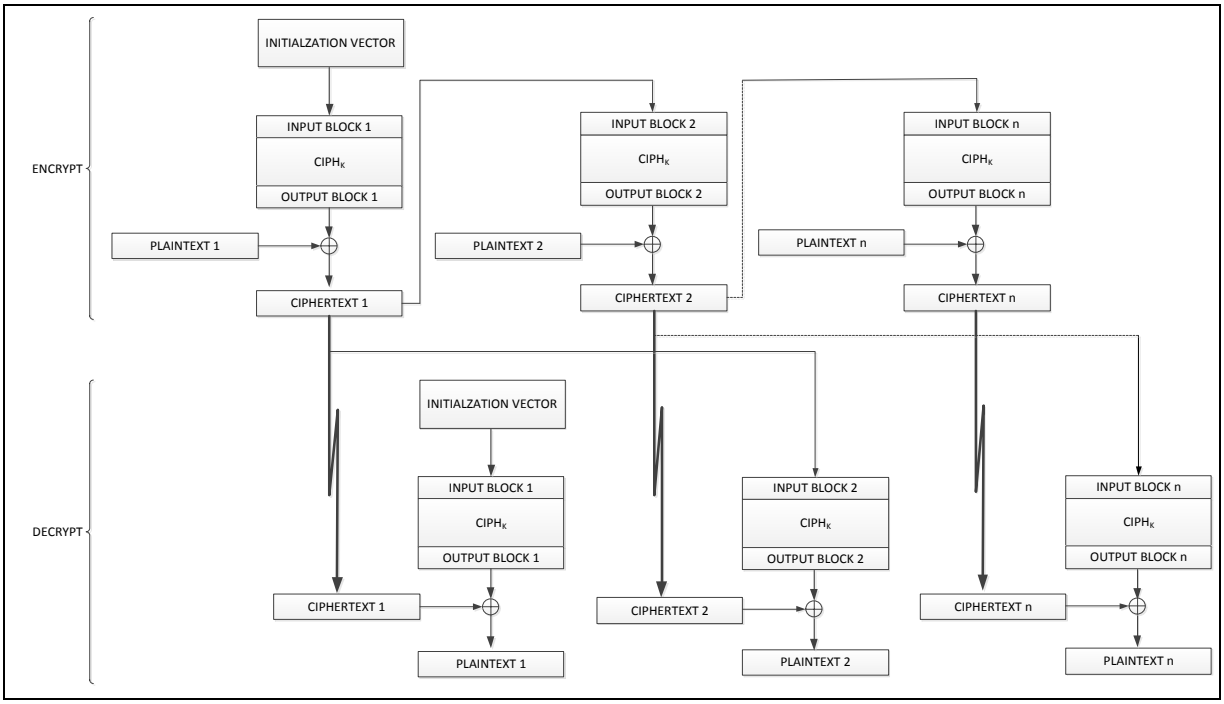


图 6.38-5 密码反馈模式

输出反馈模式

输出反馈（OFB）模式是一种保密模式，特征是正向密码迭代一个IV来生产一个输出块的序列。这是与明文进行异或运算产生密文，反之亦然。OFB模式要求IV是一个唯一数，即在一个给定的密匙下，对于该模式的每一次执行IV必须是唯一的。

在一个给定的密匙下，对于已加密的每一个消息，OFB模式需要一个唯一的IV。如果违反了这一要求，相同的IV用于多条消息的加密，那么这些消息的机密性可能会受到损害。在给定密钥下，对于一个消息的加密，如果到正向密码函数的任何输入块被指定作为另一个消息加密的IV，则保密性可能同样被损害。

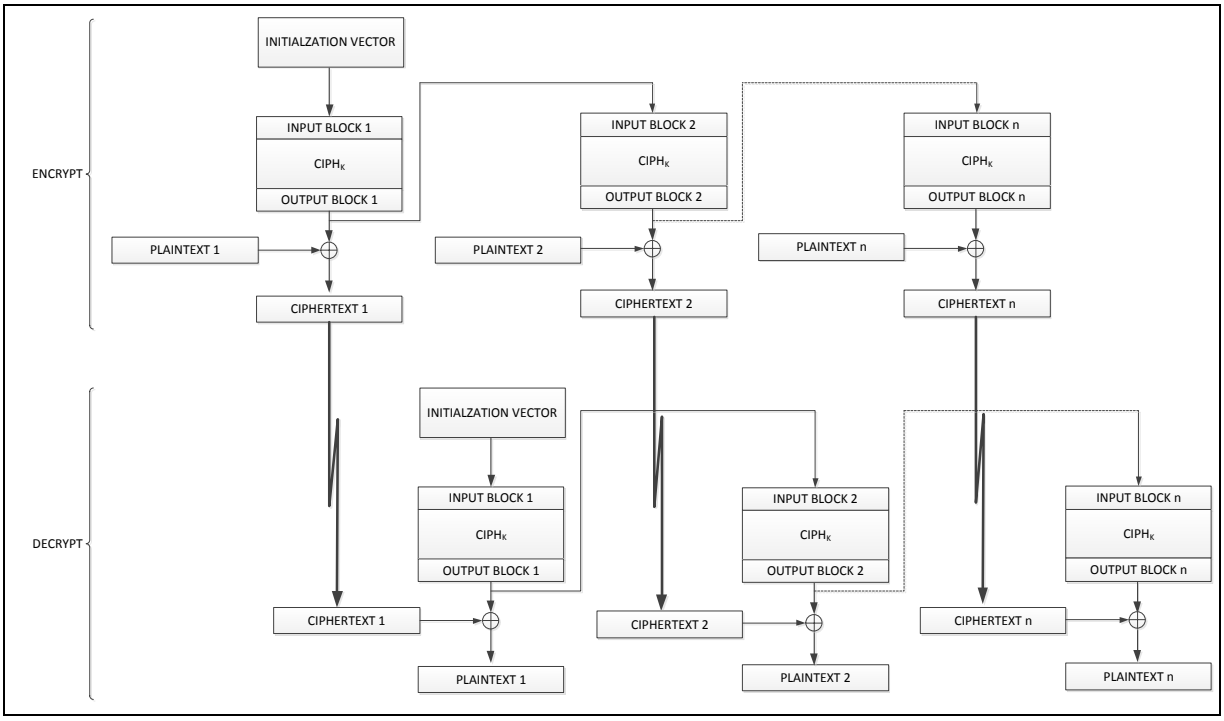


图 6.38-6 输出反馈模式

计数器模式

计数器（CTR）模式是一种保密模式，特点是正向密码应用到输入块组，叫做计数器，来产生一个输出块序列，这用于和明文异或运算来产生密文，反之亦然。计数器序列必须有这样的属性，序列中每一块不同于每一个其他块。此条件并不局限于单个消息，在给定的密钥下所有被加密的消息，所有的计数器必须是不同的。

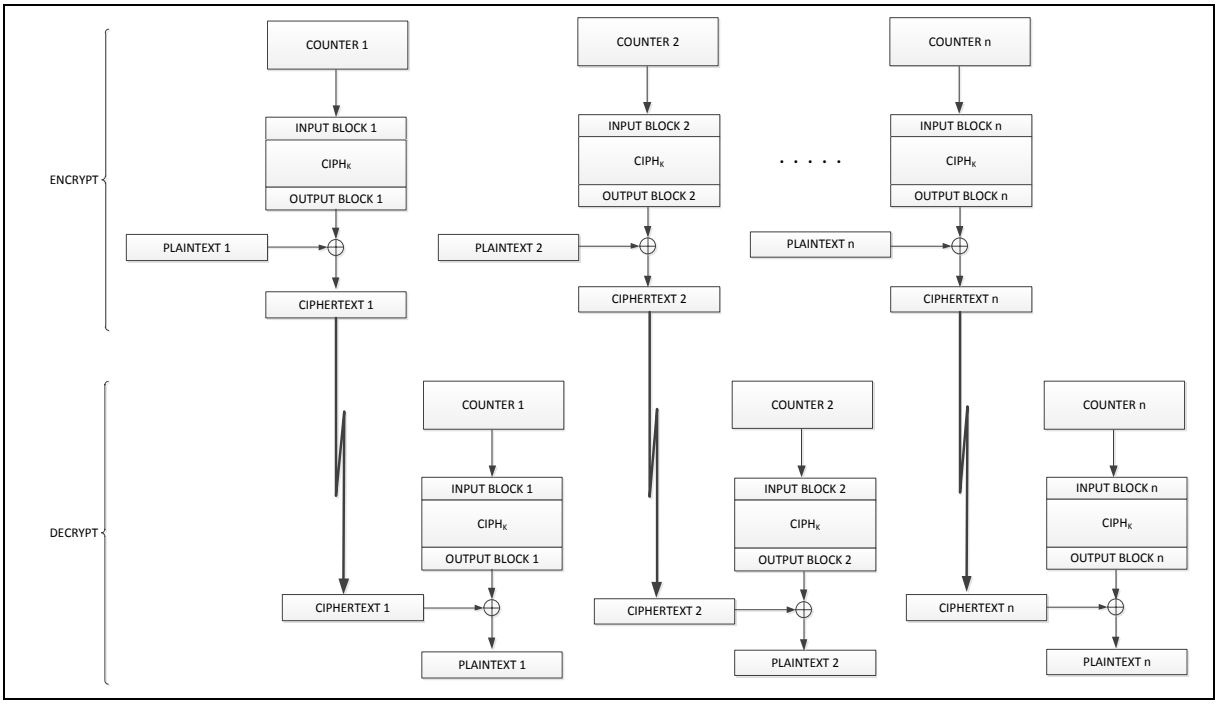


图 6.38-7 计数器模式

CBC 密文窃取 1 模式(CBC-CS1)

图6.38-8展示P_n^*为局部块的情况下的 CBC-CS1-Encrypt 算法. 加解密加速器会添加0到P_n^* 组成完整的P_n.块

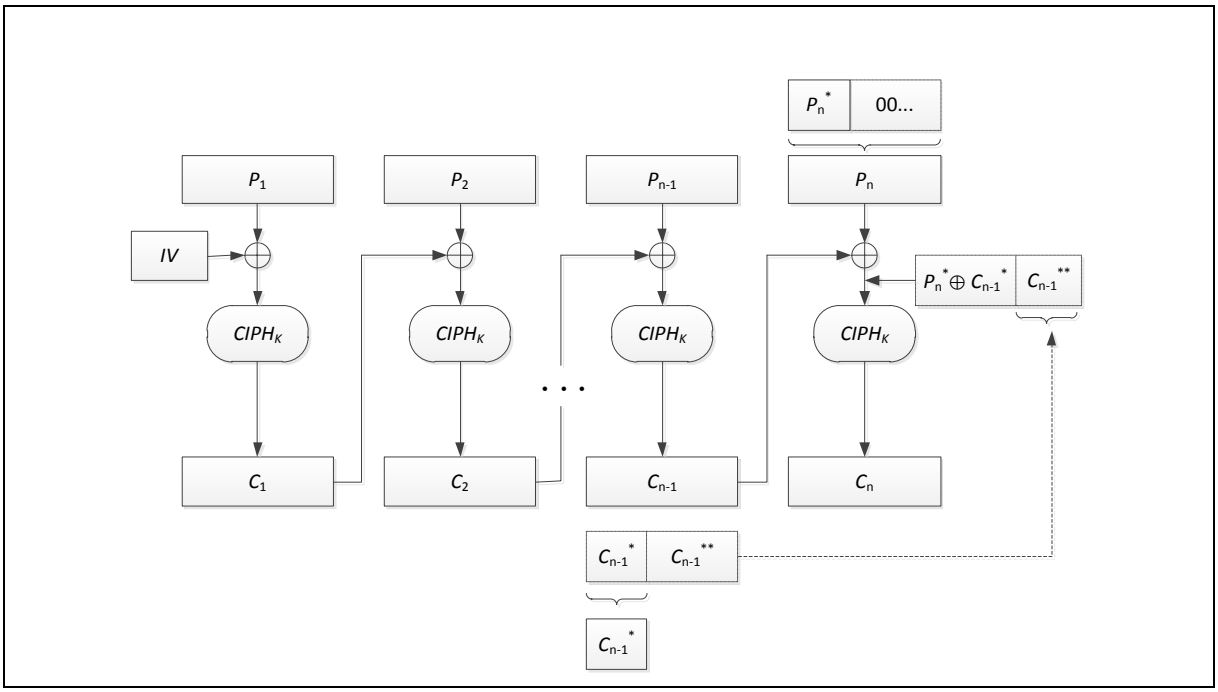


图 6.38-8 CBC-CS1 加密

图6.38-9列出C_{n-1}^*是局部块的情况下CBC-CS1-Decrypt 算法

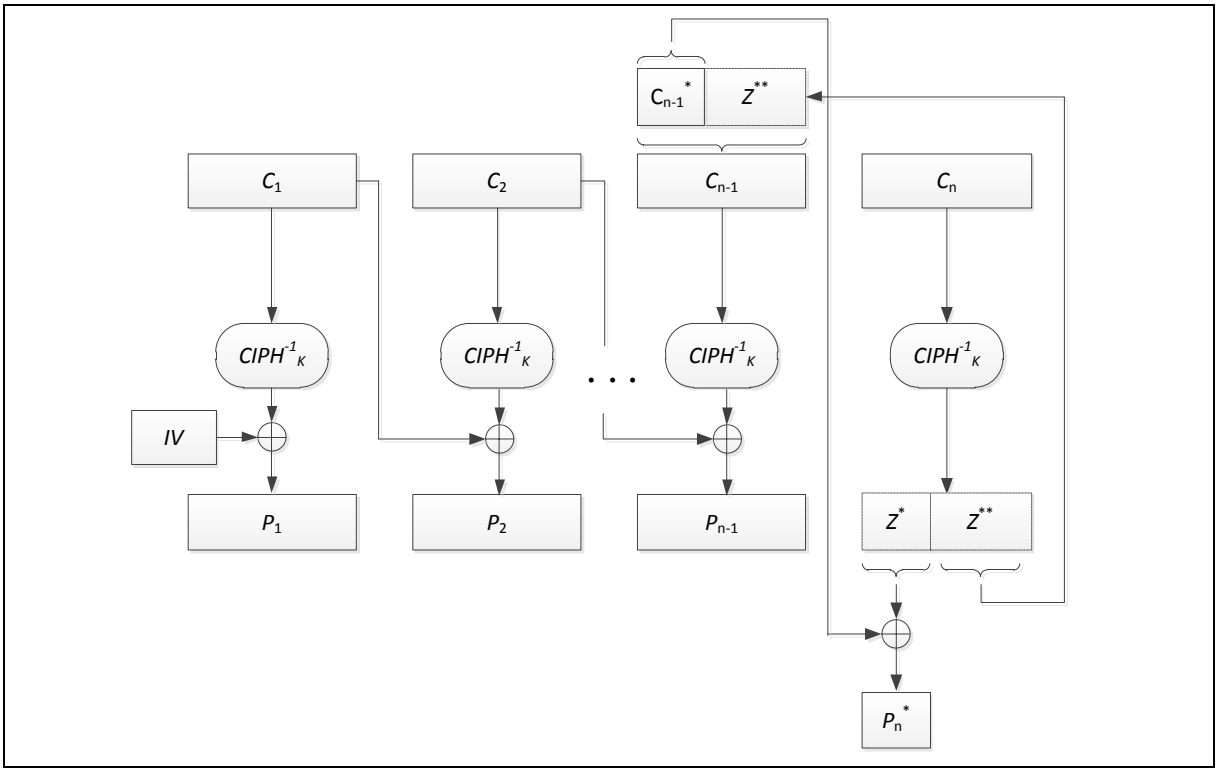


图 6.38-10 CBC-CS1 解密

CBC 密文窃取 2 模式(CBC-CS2)

当P_n^*是局部块, 那么 CBC-CS2-Encrypt 与 CBC-CS1-Encrypt 区别仅在于C_{n-1}^*和C_n.的排序不同.

CBC 密文窃取 3 模式(CBC-CS3)

C_{n-1}^* 和C_n是无条件交换，即使当C_{n-1}^*是一个完整块; 那么, CBC-CS不是严格的CBC模式的延伸. 在其他情况下,如果C_{n-1}^* 是一个非空的局部块, CBC-CS3-Encrypt 和 CBC-CS2-Encrypt 相同.

如何编程AES相关的寄存器，请参考下面编程步骤.

AES DMA 模式编程流程

1. 写 1 到 AESIEN(CRYPTO_INTEN [0]) 使能AES 中断.
2. 从四个DMA 信道中选择一个通道.
3. 编程 AES 密匙到寄存器CRYPTO_AESn_KEY0 ~ CRYPTO_ AESn_KEY7. (这里的n是选择的通道号)
4. 编程初始化向量到寄存器CRYPTO_AESn_IV0 ~ CRYPTO_AESn_IV3.
5. 编程 DMA 源地址到寄存器CRYPTO_AESn_SADDR.
6. 编程 DMA 目的地址到寄存器CRYPTO_AESn_DADDR.
7. 编程 DMA 传输字节数到寄存器CRYPTO_AESn_CNT.
8. 配置 KEYPRT(CRYPTO_TDES_CTL[31]), INSWAP(CRYPTO_TDES_CTL[23]), OUTSWAP(CRYPTO_TDES_CTL[22]), BLKSWAP(CRYPTO_TDES_CTL[21]), CHANNEL(CRYPTO_TDES_CTL[25:24]), ENCRYPTO(CRYPTO_TDES_CTL[16]), OPMODE(CRYPTO_TDES_CTL[10:8]), DMAEN(CRYPTO_TDES_CTL[7]),

3KEYS(CRYPTO_TDES_CTL[3]), 及 TMODE(CRYPTO_TDES_CTL[2]).

9. 写选择的DMA字节数个输入数据到DMA源地址
10. 写 1 到 START(CRYPTO_AES_CTL[0]) 开始 AES 加密/解密.
11. 等待AES 中断标志AESIF(CRYPTO_INTSTS[0]) 置位.
12. 从DMA目的地址读取选择的DMA字节数个数据。
13. 如果使能了DMACSCAD (CRYPTO_AES_CTL[6])，重复步骤9 到步骤12 直到所有的数据处理完.

AES 非-DMA 模式编程流程

1. 写1 到AESIEN(CRYPTO_INTEN[0]) 使能AES 中断.
2. 编程 AES 密匙到寄存器CRYPTO_AESn_KEY0 ~ CRYPTO_ AESn_KEY7. (这里的n是选择的通道号)
3. 编程初始化向量到寄存器CRYPTO_AESn_IV0 ~ CRYPTO_AESn_IV3.
4. 配置 KEYPRT(CRYPTO_AES_CTL[31]), ENCRYPTO(CRYPTO_AES_CTL[16]), OPMODE(CRYPTO_AES_CTL[15:8]), 及 KEYSZ(CRYPTO_AES_CTL[3:2]).
5. 写 1 到 START(CRYPTO_AES_CTL[0]) 来开始 AES 加密/解密.
6. 查询 INBUFFULL(CRYPTO_AES_STS[9]) 和 OUTBUFEMPTY(CRYPTO_AES_STS[16]). 如果 INBUFFULL(CRYPTO_AES_STS[9]) 是 0，写 32 位输入数据到CRYPTO_AES_DATIN，如果 OUTBUFEMPTY(CRYPTO_AES_STS[16]) 是 0，从CRYPTO_AES_DATOUT读取 32 位数据。
7. 重复步骤 6 直到 128 位数据 (16 字节) 写入AES引擎或从AES引擎读出。
8. 写 1 到 DMALAST(CRYPTO_AES_CTL[5]).
9. 如果当前操作是最后的操作，那么最后需要往寄存器CRYPTO_AES_CNT中写入数据字节计数
10. 重复步骤 6 到步骤 9 直到所有的数据处理完.

6.38.5.4 SHA (安全 Hash 算法)

用户可以参考下面步骤来理解如何编程SHA相关寄存器.

SHA DMA 模式编程流程

1. 写1到 HMACIEN(CRYPTO_INTEN[24]) 使能SHA/HMAC 中断.
2. 配置 INSWAP(CRYPTO_HMAC_CTL[23]), OUTSWAP(CRYPTO_HMAC_CTL[22]), DMAEN(CRYPTO_HMAC_CTL[7]), 和 OPMODE(CRYPTO_HMAC_CTL[10:8]). 清除 HMACEN(CRYPTO_HMAC_CTL[4]) 来选择 SHA 模式
3. 该步骤只针对串流模式。第一次串流模式运算时写1到DMACSCAD (CRYPTO_HMAC_CTL[6]) 及DMAFIRST (CRYPTO_HMAC_CTL[4])，最后一次串流模式运算时写1到DMACSCAD (CRYPTO_HMAC_CTL[6]) 及 DMALAST (CRYPTO_HMAC_CTL[5])，其他情况下只写1到DMACSCAD(CRYPTO_HMAC_CTL[6])即可
4. 填写 DMA 源地址寄存器CRYPTO_HMAC_SADDR.
5. 填写 DMA 字节数寄存器CRYPTO_HMAC_DMACNT.

6. 向 DMA 源地址写入选定的DMA 字节数的数据.
7. 写1到START(CRYPTO_HMAC_CTL[0]) 开始 SHA 加密.
8. 等待 SHA 终端标志HMACIF(CRYPTO_INTSTS[24]) 置位.
9. 读输出数据 (SHA160: CRYPTO_HMAC_DGST0 ~ CRYPTO_HMAC_DGST4, SHA224: CRYPTO_HMAC_DGST0 ~ CRYPTO_HMAC_DGST6, SHA256: CRYPTO_HMAC_DGST0 ~ CRYPTO_HMAC_DGST7, SHA384: CRYPTO_HMAC_DGST0 ~ CRYPTO_HMAC_DGST11, SHA512: CRYPTO_HMAC_DGST0 ~ CRYPTO_HMAC_DGST15).

注意: KEYCNT(CRYPTO_HMAC_KEYCNT[31:0]) 保持SHA/HMAC 引擎操作的种子的字节数。SHA DMA串流模式下，DMACNT(CRYPTO_HMAC_DMACNT[31:0])必须保持区块空间对齐，最后一次串流模式计算除外

SHA 非-DMA 模式编程流程

1. 配置 INSWAP(CRYPTO_HMAC_CTL[23]), OUTSWAP(CRYPTO_HMAC_CTL[22]), 和 OPMODE(CRYPTO_HMAC_CTL[10:8]). 清除 HMACEN(CRYPTO_HMAC_CTL[4])选择 SHA 模式
2. 如果是最后的输入字，设置DMALAST(CRYPTO_HMAC_CTL[5]).
3. 写 1到 START(CRYPTO_HMAC_CTL[0]) 开始 SHA 加密.
4. 等待 SHA 数据输入请求DATINREQ(CRYPTO_HMAC_STS[16]) 置位.
5. 写一笔数据到 CRYPTO_HMAC_DATIN.
6. 重复步骤 2 到 5 直到所有的输入数据写入到SHA 引擎.
7. 等待 BUSY (CRYPTO_HMAC_STS[0]) 清除.
8. 读取输出数据 (SHA160: CRYPTO_HMAC_DGST0 ~ CRYPTO_HMAC_DGST4, SHA224: CRYPTO_HMAC_DGST0 ~ CRYPTO_HMAC_DGST6, SHA256: CRYPTO_HMAC_DGST0 ~ CRYPTO_HMAC_DGST7, SHA384: CRYPTO_HMAC_DGST0 ~ CRYPTO_HMAC_DGST11, SHA512: CRYPTO_HMAC_DGST0 ~ CRYPTO_HMAC_DGST15).

注意: KEYCNT(CRYPTO_HMAC_KEYCNT[31:0]) 保持 SHA/HMAC 引擎操作的种子的字节数.

6.38.5.5 HMAC (Keyed-Hash消息认证码)

Keyed-Hash 消息认证码是一个特殊的架构来用于计算涉及密码散列函数与秘密加密密钥结合的消息验证码的特定结构。任意加密散列函数，例如SHA-1, 可能用于计算一个 HMAC; 由此产生的MAC 相应的称为 HMAC-SHA1

用户可以参考下面的步骤理解如何编写HMAC 相关寄存器.

HMAC DMA 模式编程流程

1. 写 1 到 HMACIEN(CRYPTO_INTEN[24]) 使能 HMAC中断.
2. 配置 INSWAP(CRYPTO_HMAC_CTL[23]), OUTSWAP(CRYPTO_HMAC_CTL[22]), DMAEN(CRYPTO_HMAC_CTL[7]), 和 OPMODE(CRYPTO_HMAC_CTL[10:8]). 设置

HMACEN(CRYPTO_HMAC_CTL[4]) 选择HMAC 模式

3. 该步骤只针对串流模式。第一次串流模式运算时写1到DMACSCAD (CRYPTO_HMAC_CTL[6]) 及 DMAFIRST (CRYPTO_HMAC_CTL[4])，最后一次串流模式运算时写1到DMACSCAD (CRYPTO_HMAC_CTL[6]) 和 DMALAST (CRYPTO_HMAC_CTL[5])，其他情况下只写1到 DMACSCAD(CRYPTO_HMAC_CTL[6]) 即可
4. 填写 DMA 源地址寄存器SADDR(CRYPTO_HMAC_SADDR[31:0]).
5. 填写 DMA 字节数寄存器DMACNT(CRYPTO_HMAC_DMACNT[31:0])
6. 向 DMA 源地址写入选择DMA字节数的数据.
7. 写 1 到 START(CRYPTO_HMAC_CTL[0]) 开始HMAC 加密.
8. 等待 HMAC中断标识HMACIF(CRYPTO_INTSTS[24]) 置位
9. 读取输出数据 (HMAC-SHA160: CRYPTO_HMAC_DGST0 ~ CRYPTO_HMAC_DGST4, HMAC-SHA224: CRYPTO_HMAC_DGST0 ~ CRYPTO_HMAC_DGST6, HMAC-SHA256: CRYPTO_HMAC_DGST0 ~ CRYPTO_HMAC_DGST7, HMAC-SHA384: CRYPTO_HMAC_DGST0 ~ CRYPTO_HMAC_DGST11, HMAC-SHA512: CRYPTO_HMAC_DGST0 ~ CRYPTO_HMAC_DGST15).

注意: KEYCNT(CRYPTO_HMAC_KEYCNT[31:0]) 保持SHA/HMAC 引擎操作的种子的字节数.

HMAC 非-DMA 模式编程流程

1. 配置 INSWAP(CRYPTO_HMAC_CTL[23]), OUTSWAP(CRYPTO_HMAC_CTL[22]), 和 OPMODE(CRYPTO_HMAC_CTL[10:8]). 设置 HMACEN(CRYPTO_HMAC_CTL[4])选择 HMAC 模式
2. 如果是最后的输入字，设置DMALAST(CRYPTO_HMAC_CTL[5]).
3. 写 1 到 START(CRYPTO_HMAC_CTL[0]) 开始 HMAC 加密.
4. 等待 HMAC 数据输入请求 DATINREQ(CRYPTO_HMAC_STS[16]) 置位.
5. 写一笔数据到 CRYPTO_HMAC_DATIN.
6. 重复步骤 2 到 5 直到所有的输入数据写入SHA 引擎.
7. 等待 BUSY (CRYPTO_HMAC_STS[0]) 清除.
8. 读到输出数据 (HMAC-SHA160: CRYPTO_HMAC_DGST0 ~ CRYPTO_HMAC_DGST4, HMAC-SHA224: CRYPTO_HMAC_DGST0 ~ CRYPTO_HMAC_DGST6, HMAC-SHA256: CRYPTO_HMAC_DGST0 ~ CRYPTO_HMAC_DGST7, HMAC-SHA384: CRYPTO_HMAC_DGST0 ~ CRYPTO_HMAC_DGST11, HMAC-SHA512: CRYPTO_HMAC_DGST0 ~ CRYPTO_HMAC_DGST15).
9.
10. **注意:** KEYCNT(CRYPTO_HMAC_KEYCNT[31:0]) 保持SHA/HMAC 引擎操作的种子的字节数.

6.38.5.6 ECC (椭圆曲线密码体制)

椭圆曲线密码体制 (ECC) 是一种著名的公钥密码体制的方法。近年来，许多协议和应用都利用有限域上椭圆曲线的代数循环群特征构造密码系统. 椭圆曲线的所有点都遵循椭圆曲线的公式：y2≡x3+A*x+B (mod N) 在 GF(p) 和 y2+x*y≡x3+A*x2+B (mod N)在 GF(2m)，图6.38-10显示ECC应用的主要层次结构图。经常出现的参数和相应的寄存器如表6.38-6所示。

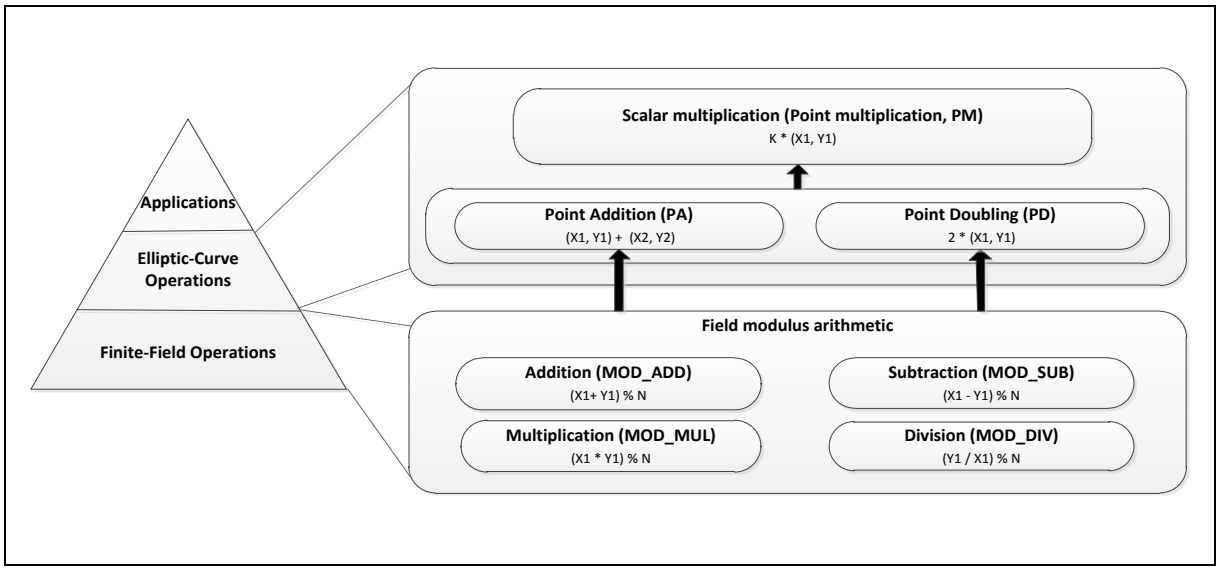


图 6.38-11ECC 的主要层次结构图

参数	描述	相应寄存器
X1	点 1 的 X 坐标	CRYPTO_ECC_X1_00~ CRYPTO_ECC_X1_17
Y1	点 1 的 Y 坐标	CRYPTO_ECC_Y1_00~ CRYPTO_ECC_Y1_17
X2	点 2 的 X 坐标	CRYPTO_ECC_X2_00~ CRYPTO_ECC_X2_17
Y2	点 2 的 Y 坐标	CRYPTO_ECC_Y2_00~ CRYPTO_ECC_Y2_17
A	曲线参数 A	CRYPTO_ECC_A_00~ CRYPTO_ECC_A_17
B	曲线参数 B	CRYPTO_ECC_B_00~ CRYPTO_ECC_B_17
N	曲线参数 N	CRYPTO_ECC_N_00~ CRYPTO_ECC_N_17
M	曲线长度	CRYPTO_ECC_CTL[31:22]
K	标量常数	CRYPTO_ECC_K_00~ CRYPTO_ECC_K_17

表 6.38-6 ECC 参数和相应寄存器列表

标量乘法（点乘）是ECC应用中的核心运算。标量乘法的计算由点加法和点乘运算组成。此外，在点加法和倍运算公式中存在许多有限域模运算。为了加速 ECC 应用, 我们提出了一个椭圆曲线加密加速器，可以处理不仅在GF(p) 和 GF(2m)的三点操作也支持在GF(p)的四模操作。在开始ECC 加速器之前, 用户必须提供所需的ECC操作输入数据包括点坐标 (X1, Y1, X2, Y2), 曲线参数 (A, B, N, M) 和标量数据 (K) 如表6.38-7 标记“√”表示输入数据对这个操作是必须的。在我们的ECC加速器的输入数据和相应的寄存器的详细定义显示在下一节寄存器映射

在 ECC 加速器完成之后，所有的点操作会产生一个输出点，包含 CRYPTO_ECC_X1_00 到 CRYPTO_ECC_X1_17 寄存器的 X 坐标和 CRYPTO_ECC_Y1_00 到 CRYPTO_ECC_Y1_17 寄存器的 Y 坐标。在所有的模操作中， ECC 加速器只产生结果到寄存器 CRYPTO_ECC_X1_00 搭配 CRYPTO_ECC_X1_17.

操作	PM	PA	PD	MOD_DIV	MOD_MUL	MOD_ADD	MOD_SUB
ECCOP[1:0]	00	10	11	01	01	01	01
MODOP[1:0]	XX	XX	XX	00	01	10	11
X1	√	√	√	√	√	√	√
Y1	√	√	√	√	√	√	√
X2		√					
Y2		√					
A	√	√	√				
B	√		√				
N	√	√	√	√	√	√	√
M	√	√	√		√		
K	√						

表 6.38-7 各种操作所需的输入数据

用户可以参考以下步骤来了解如何编写ECC相关寄存器。

ECC DMA 模式编程流程

1. 写 1到 ECCIEN(CRYPTO_INTEN[22]) 使能 ECC 中断.
2. 填写 DMA源地址寄存器CRYPTO_ECC_SADDR.
3. 填写 DMA 目的地址寄存器CRYPTO_ECC_DADDR.
4. 填写 DMA 字数寄存器CRYPTO_ECC_WORDCNT.
5. 向开始寄存器地址填写所有的输入数据表6.38-7展示我们要更新到寄存器 CRYPTO_ECC_STARTREG的内容.
6. 向 DMA 源地址寄存器写入设定DMA 字数的数据
7. 配置 CURVEM(CRYPTO_ECC_CTL[31:22]), LDK(CRYPTO_ECC_CTL[21]), LDN(CRYPTO_ECC_CTL[20]), LDB(CRYPTO_ECC_CTL[19]), LDA(CRYPTO_ECC_CTL[18]), LDP2(CRYPTO_ECC_CTL[17]), LDP1(CRYPTO_ECC_CTL[16]), SBM(CRYPTO_ECC_CTL[15]), SCAP(CRYPTO_ECC_CTL[14]), CSEL(CRYPTO_ECC_CTL[13]), MODOP(CRYPTO_ECC_CTL[12:11]), ECCOP(CRYPTO_ECC_CTL[10:9]), FSEL(CRYPTO_ECC_CTL[8]), DMAEN(CRYPTO_ECC_CTL[7]), 及 START(CRYPTO_ECC_CTL[0])
8. 等待 ECC 中断标志 ECCIF(CRYPTO_INTSTS[22]) 置位.
9. 读取输出数据，清除 ECC 中断标志ECCIF(CRYPTO_INTSTS[22]).

ECC 非-DMA 模式编程流程

1. 依据表6.38-7写数据到相应的寄存器，比如CURVEA, CURVEB, CURVEN, SCALARK

2. 配置 CURVEM(CRYPTO_ECC_CTL[31:22]), LDK(CRYPTO_ECC_CTL[21]), LDN(CRYPTO_ECC_CTL[20]), LDB(CRYPTO_ECC_CTL[19]), LDA(CRYPTO_ECC_CTL[18]), LDP2(CRYPTO_ECC_CTL[17]), LDP1(CRYPTO_ECC_CTL[16]), SBM(CRYPTO_ECC_CTL[15]), SCAP(CRYPTO_ECC_CTL[14]), CSEL(CRYPTO_ECC_CTL[13]), MODOP(CRYPTO_ECC_CTL[12:11]), ECCOP(CRYPTO_ECC_CTL[10:9]), FSEL(CRYPTO_ECC_CTL[8]), 及 START(CRYPTO_ECC_CTL[0])
3. 等待 BUSY (CRYPTO_ECC_STS[0]) 位清0.
4. 读取输出数据，清除 ECC 中断标志 ECCIF(CRYPTO_INTSTS[22]).

ECC 加速器的一些注意事项如下

1. ECC 加速器支持的密钥长度从163 到 256位.
2. 所有的输入输出数据都必须是正数 (如果输入数据是负数,需要加上N).
3. GF(2^m) 不可约多项式必须采用HP里最小的，请参考表6.38-8

	2,1	**3,1**	**4,1**	**5,2**
6,1	7,1	8,4,3,1	9,1	10,3
11,2	12,3	13,4,3,1	14,5	15,1
16,5,3,1	17,3	18,3	19,5,2,1	20,3
21,2	22,1	23,5	24,4,3,1	25,3
26,4,3,1	27,5,2,1	28,1	29,2	30,1
31,3	32,7,3,2	33,10	34,7	35,2
36,9	37,6,4,1	38,6,5,1	39,4	40,5,4,3
41,3	42,7	43,6,4,3	44,5	45,4,3,1
46,1	47,5	48,5,3,2	49,9	50,4,3,2
51,6,3,1	52,3	53,6,2,1	54,9	55,7
56,7,4,2	57,4	58,19	59,7,4,2	60,1
61,5,2,1	62,29	63,1	64,4,3,1	65,18
66,3	67,5,2,1	68,9	69,6,5,2	70,5,3,1
71,6	72,10,9,3	73,25	74,35	75,6,3,1
76,21	77,6,5,2	78,6,5,3	79,9	80,9,4,2
81,4	82,8,3,1	83,7,4,2	84,5	85,8,2,1
86,21	87,13	88,7,6,2	89,38	90,27
91,8,5,1	92,21	93,2	94,21	95,11
96,10,9,6	97,6	98,11	99,6,3,1	100,15
101,7,6,1	102,29	103,9	104,4,3,1	105,4
106,15	107,9,7,4	108,17	109,5,4,2	110,33
111,10	112,5,4,3	113,9	114,5,3,2	115,8,7,5
116,4,2,1	117,5,2,1	118,33	119,8	120,4,3,1

121,18	122,6,2,1	123,2	124,19	125,7,6,5
126,21	127,1	128,7,2,1	129,5	130,3
131,8,3,2	132,17	133,9,8,2	134,57	135,11
136,5,3,2	137,21	138,8,7,1	139,8,5,3	140,15
141,10,4,1	142,21	143,5,3,2	144,7,4,2	145,52
146,71	147,14	148,27	149,10,9,7	150,53
151,3	152,6,3,2	153,1	154,15	155,62
156,9	157,6,5,2	158,8,6,5	159,31	160,5,3,2
161,18	162,27	163,7,6,3	164,10,8,7	165,9,8,3
166,37	167,6	168,15,3,2	169,34	170,11
171,6,5,2	172,1	173,8,5,2	174,13	175,6
176,11,3,2	177,8	178,31	179,4,2,1	180,3
181,7,6,1	182,81	183,56	184,9,8,7	185,24
186,11	187,7,6,5	188,6,5,2	189,6,5,2	190,8,7,6
191,9	192,7,2,1	193,15	194,87	195,8,3,2
196,3	197,9,4,2	198,9	199,34	200,5,3,2
201,14	202,55	203,8,7,1	204,27	205,9,5,2
206,10,9,5	207,43	208,9,3,1	209,6	210,7
211,11,10,8	212,105	213,6,5,2	214,73	215,23
216,7,3,1	217,45	218,11	219,8,4,1	220,7
221,8,6,2	222,5,4,2	223,33	224,9,8,3	225,32
226,10,7,3	227,10,9,4	228,113	229,10,4,1	230,8,7,6
231,26	232,9,4,2	233,74	234,31	235,9,6,1
236,5	237,7,4,1	238,73	239,36	240,8,5,3
241,70	242,95	243,8,5,1	244,111	245,6,4,1
246,11,2,1	247,82	248,15,14,10	249,35	250,103
251,7,4,2	252,15	253,46	254,7,2,1	255,52
256,10,5,2	257,12	258,71	259,10,6,2	260,15
261,7,6,4	262,9,8,4	263,93	264,9,6,2	265,42
266,47	267,8,6,3	268,25	269,7,6,1	270,53
271,58	272,9,3,2	273,23	274,67	275,11,10,9
276,63	277,12,6,3	278,5	279,5	280,9,5,2
281,93	282,35	283,12,7,5	284,53	285,10,7,5
286,69	287,71	288,11,10,1	289,21	290,5,3,2
291,12,11,5	292,37	293,11,6,1	294,33	295,48

296,7,3,2	297,5	298,11,8,4	299,11,6,4	300,5
301,9,5,2	302,41	303,1	304,11,2,1	305,102
306,7,3,1	307,8,4,2	308,15	309,10,6,4	310,93
311,7,5,3	312,9,7,4	313,79	314,15	315,10,9,1
316,63	317,7,4,2	318,45	319,36	320,4,3,1
321,31	322,67	323,10,3,1	324,51	325,10,5,2
326,10,3,1	327,34	328,8,3,1	329,50	330,99
331,10,6,2	332,89	333,2	334,5,2,1	335,10,7,2
336,7,4,1	337,55	338,4,3,1	339,16,10,7	340,45
341,10,8,6	342,125	343,75	344,7,2,1	345,22
346,63	347,11,10,3	348,103	349,6,5,2	350,53
351,34	352,13,11,6	353,69	354,99	355,6,5,1
356,10,9,7	357,11,10,2	358,57	359,68	360,5,3,2
361,7,4,1	362,63	363,8,5,3	364,9	365,9,6,5
366,29	367,21	368,7,3,2	369,91	370,139
371,8,3,2	372,111	373,8,7,2	374,8,6,5	375,16
376,8,7,5	377,41	378,43	379,10,8,5	380,47
381,5,2,1	382,81	383,90	384,12,3,2	385,6
386,83	387,8,7,1	388,159	389,10,9,5	390,9
391,28	392,13,10,6	393,7	394,135	395,11,6,5
396,25	397,12,7,6	398,7,6,2	399,26	400,5,3,2
401,152	402,171	403,9,8,5	404,65	405,13,8,2
406,141	407,71	408,5,3,2	409,87	410,10,4,3
411,12,10,3	412,147	413,10,7,6	414,13	415,102
416,9,5,2	417,107	418,199	419,15,5,4	420,7
421,5,4,2	422,149	423,25	424,9,7,2	425,12
426,63	427,11,6,5	428,105	429,10,8,7	430,14,6,1
431,120	432,13,4,3	433,33	434,12,11,5	435,12,9,5
436,165	437,6,2,1	438,65	439,49	440,4,3,1
441,7	442,7,5,2	443,10,6,1	444,81	445,7,6,4
446,105	447,73	448,11,6,4	449,134	450,47
451,16,10,1	452,6,5,4	453,15,6,4	454,8,6,1	455,38
456,18,9,6	457,16	458,203	459,13,5,2	460,19
461,7,6,1	462,73	463,93	464,19,18,13	465,31
466,14,11,6	467,11,6,1	468,27	469,9,5,2	470,9

471,1	472,11,3,2	473,200	474,191	475,9,8,4
476,9	477,16,15,7	478,121	479,104	480,15,9,6
481,138	482,9,6,5	483,9,6,4	484,105	485,17,16,6
486,81	487,94	488,4,3,1	489,83	490,219
491,11,6,3	492,7	493,10,5,3	494,17	495,76
496,16,5,2	497,78	498,155	499,11,6,5	500,27
501,5,4,2	502,8,5,4	503,3	504,15,14,6	505,156
506,23	507,13,6,3	508,9	509,8,7,3	510,69
511,10	512,8,5,2	513,26	514,67	515,14,7,4
516,21	517,12,10,2	518,33	519,79	520,15,11,2
521,32	522,39	523,13,6,2	524,167	525,6,4,1
526,97	527,47	528,11,6,2	529,42	530,10,7,3
531,10,5,4	532,1	533,4,3,2	534,161	535,8,6,2
536,7,5,3	537,94	538,195	539,10,5,4	540,9
541,13,10,4	542,8,6,1	543,16	544,8,3,1	545,122
546,8,2,1	547,13,7,4	548,10,5,3	549,16,4,3	550,193
551,135	552,19,16,9	553,39	554,10,8,7	555,10,9,4
556,153	557,7,6,5	558,73	559,34	560,11,9,6
561,71	562,11,4,2	563,14,7,3	564,163	565,11,6,1
566,153	567,28	568,15,7,6	569,77	570,67
571,10,5,2	572,12,8,1	573,10,6,4	574,13	575,146
576,13,4,3	577,25	578,23,22,16	579,12,9,7	580,237

表 6.38-8 轻量级二进制不可约多项式

4. 只有当 START 和 DMAEN (CRYPTO_ECC_CTL[0] 和 [7]) 同时为1，ECC DMA 模式才有效.
5. 当 ECC 引擎正在工作 (BUSY(CRYPTO_ECC_STS[0])是 1 且 DMABUSY (CRYPTO_ECC_STS[1]) 是 0), 用户不能修改所有的输入数据寄存器 (CRYPTO_ECC_X1_00 ~ CRYPTO_ECC_K_17).
6. 如果用户想要停止 ECC 加速器, 请配置 STOP (CRYPTO_ECC_CTL[1]) 为 1. 注意: 为了避免下次操作的传输错误, BUSY 信号不会立即清除，直到DMA操作完成
7. 在二进制域不支持模操作.
8. PF的模乘运算和除法运算的输入数据必须小于n。
9. K是私有密钥, 所以这个寄存器只写.

以下是密钥对产生和 ECDSA 应用的方法.

密钥对的产生

公用密钥产生方程 : Q = dG (mod N)

1. 依据表6.38-6写曲线参数 A, B, N, 和曲线长度 M 到相应的寄存器
2. 依据表6.38-6写点G(x, y) 到 X1, Y1 寄存器
3. 依据表6.38-6写私有密钥 d 到 K 寄存器
4. 设置 ECCOP(CRYPTO_ECC_CTL[10:9]) 为 00
5. 设置 FSEL(CRYPTO_ECC_CTL[8]) 依据使用的素数域或二进制域的曲线
6. 设置 START(CRYPTO_ECC_CTL[0]) 为 1
7. 等待 BUSY(CRYPTO_ECC_STS[0])清0
8. 从X1, Y1 寄存器读公共密钥 Q

椭圆曲线数字签名算法(ECDSA)

ECDSA 签名产生流程:

1. 计算 e = HASH(m), 这里HASH 是密码哈希算法(即 SHA-256)
   1) 使用 SHA 计算 e
2. 从[1, n-1]选择随机整数k
   1) 注意n是数阶，不是素数模或不可约多项式函数
3. 计算 r = x1 (mod n), 这里 (x1, y1) = k * G. 如果 r = 0, 跳到步骤 2
   1) 依据表6.38-6写入曲线参数 A, B, N 和曲线长度 M 到相应的寄存器
   2) 依据表6.38-6将素数模或不可约多项式函数写到 N 寄存器
   3) 依据表6.38-6写点 G(x, y) 到 X1, Y1 寄存器
   4) 依据表6.38-6写随机整数 k 到 K 寄存器
   5) 设置 ECCOP(CRYPTO_ECC_CTL[10:9]) 为 00
   6) 设置 FSEL(CRYPTO_ECC_CTL[8]) 依据使用的素数域或二进制域的曲线
   7) 设置 START(CRYPTO_ECC_CTL[0]) 为 1
   8) 等待 BUSY(CRYPTO_ECC_STS[0]) 清0
   9) 依据表6.38-6将曲线顺序和曲线长度写入 N ,M 寄存器
   10) 写 0x0 到 Y1 寄存器
   11) 设置 ECCOP(CRYPTO_ECC_CTL[10:9]) 为 01
   12) 设置 MOPOP(CRYPTO_ECC_CTL[12:11]) 为 10
   13) 设置 START(CRYPTO_ECC_CTL[0]) 为 1
   14) 等待 BUSY(CRYPTO_ECC_STS[0]) 清0
   15) 读 X1 寄存器得到 r
4. 计算 $s = k^{-1} \times (e + d \times r) (\mathrm{mod}\ n)$. 如果 s = 0, 跳到步骤 2
   1) 依据表6.38-6写曲线顺序到 N 寄存器

2) 依据表6.38-6写随机整数 k 到 X1 寄存器
3) 设置 ECCOP(CRYPTO_ECC_CTL[10:9]) 为 01
4) 设置 MOPOP(CRYPTO_ECC_CTL[12:11]) 为 00
5) 设置 START(CRYPTO_ECC_CTL[0]) 到 1
6) 等待 BUSY(CRYPTO_ECC_STS[0]) 清0
7) 读 X1 寄存器得到 k^{-1}
8) 依据表6.38-6写曲线顺序和曲线长度到 N ,M寄存器
9) 写 r, d 到 X1, Y1 寄存器
10) 设置 ECCOP(CRYPTO_ECC_CTL[10:9]) 为 01
11) 设置 MOPOP(CRYPTO_ECC_CTL[12:11]) 为 01
12) 设置 START(CRYPTO_ECC_CTL[0]) 到 1
13) 等待 BUSY(CRYPTO_ECC_STS[0]) 清0
14) 依据表6.38-6写曲线顺序到N 寄存器
15) 写 e 到 Y1 寄存器
16) 设置 ECCOP(CRYPTO_ECC_CTL[10:9]) 到 01
17) 设置MOPOP(CRYPTO_ECC_CTL[12:11]) 到 10
18) 设置 START(CRYPTO_ECC_CTL[0]) 为 1
19) 等待BUSY(CRYPTO_ECC_STS[0]) 清0
20) 依据表6.38-6写曲线顺序和曲线长度到N ,M 寄存器
21) 写 k^{-1}到 Y1 寄存器
22) 设置 ECCOP(CRYPTO_ECC_CTL[10:9]) 为 01
23) 设置 MOPOP(CRYPTO_ECC_CTL[12:11]) 为 01
24) 设置 START(CRYPTO_ECC_CTL[0]) 为 1
25) 等待 BUSY(CRYPTO_ECC_STS[0]) 清0
26) 读 X1 寄存器得到 s

5. 签名是 pair (r, s)

ECDSA 签名验证步骤:

1. 校验r 和 s 是 [1, n-1]范围内的整数. 如果不是，签名无效
   1) 计算 e = HASH (m), 这里 HASH 签名生成中的哈希算法
2. 用 SHA 来计算 e
3. 计算 $w = s^{-1} \pmod n$
   1) 依据表6.38-6写曲线顺序到N 寄存器
   2) 依据表6.38-6写 s 到 X1 寄存器
   3) 依据表6.38-6写 0x1 到 Y1寄存器

4) 设置 ECCOP(CRYPTO_ECC_CTL[10:9]) 为 01
5) 设置 MOPOP(CRYPTO_ECC_CTL[12:11]) 为 00
6) 设置 FSEL(CRYPTO_ECC_CTL[8]) 依据使用的素数域或二进制域曲线a
7) 设置 START(CRYPTO_ECC_CTL[0]) 为 1
8) 等待 BUSY(CRYPTO_ECC_STS[0]) 清0
9) 读 X1 寄存器得到 w

4. 计算 u1 = e × w (mod n) 和 u2 = r × w (mod n)
   1) 依据表6.38-6写曲线顺序和曲线长度到N ,M 寄存器
   2) 写 e, w到 X1, Y1 寄存器
   3) 设置 ECCOP(CRYPTO_ECC_CTL[10:9]) 为 01
   4) 设置 MOPOP(CRYPTO_ECC_CTL[12:11]) 为 01
   5) 设置 START(CRYPTO_ECC_CTL[0]) 为 1
   6) 等待 BUSY(CRYPTO_ECC_STS[0]) 清0
   7) 读 X1 寄存器得到 u1
   8) 依据表6.38-6写曲线顺序和曲线长度到 N ,M 寄存器
   9) 写 r, w 到 X1, Y1 寄存器
   10) 设置 ECCOP(CRYPTO_ECC_CTL[10:9]) 为 01
   11) 设置 MOPOP(CRYPTO_ECC_CTL[12:11])为 01
   12) 设置START(CRYPTO_ECC_CTL[0]) 为 1
   13) 等待 BUSY(CRYPTO_ECC_STS[0]) 清0
   14) 读 X1 寄存器得到 u2
5. 计算 X' (x1', y1') = u1 * G + u2 * Q
   1) 依据表6.38-6写曲线参数 A, B, N, 和曲线长度 M 到相应寄存器
   2) 写点 G(x, y) 到 X1, Y1 寄存器
   3) 写 u1 到 K 寄存器
   4) 设置 ECCOP(CRYPTO_ECC_CTL[10:9]) 为 00
   5) 设置 START(CRYPTO_ECC_CTL[0]) 为 1
   6) 等待 BUSY(CRYPTO_ECC_STS[0]) 清0
   7) 读 X1, Y1寄存器得到 u1*G
   8) 依据表6.38-6写曲线参数 A, B, N, 和曲线长度 M 到相应寄存器
   9) 写公共密钥 Q(x,y) 到 X1, Y1 寄存器
   10) 写 u2到 K 寄存器
   11) 设置 ECCOP(CRYPTO_ECC_CTL[10:9]) 为 00
   12) 设置 START(CRYPTO_ECC_CTL[0]) 为 1

13) 等待 BUSY(CRYPTO_ECC_STS[0]) 清0
14) 依据表6.38-6写曲线参数 A, B, N, 和曲线长度 M 到相应寄存器
15) 写结果数据 u1*G 到 X2, Y2 寄存器
16) 设置 ECCOP(CRYPTO_ECC_CTL[10:9]) 为 10
17) 设置 START(CRYPTO_ECC_CTL[0]) 为 1
18) 等待 BUSY(CRYPTO_ECC_STS[0]) 清0
19) 读 X1, Y1寄存器得到X'(x1', y1')
20) 依据表6.38-6写曲线顺序和曲线长度到 N ,M 寄存器
21) 写 x1' 到 X1寄存器
22) 写 0x0到 Y1 寄存器
23) 设置 ECCOP(CRYPTO_ECC_CTL[10:9]) 为 01
24) 设置 MOPOP(CRYPTO_ECC_CTL[12:11])为 10
25) 设置 START(CRYPTO_ECC_CTL[0]) 为 1
26) 等待BUSY(CRYPTO_ECC_STS[0]) 清0
27) 读 X1 寄存器得到 x1' (mod n)

6. 签名当x1' = r时有效, 其他无效

椭圆曲线 Diffie-Hellman 带辅助秘钥

共享密钥产生公式: Z是P的x-坐标，P = hdQ (mod N)

1. 依据表6.38-6写曲线参数A,B,N和曲线长度M到相应的寄存器
2. 据表6.38-6写接收方公共密钥 Q(x,y) 到 X1, Y1 寄存器
3. 据表6.38-6写 (辅因子 h * 我的私钥 d) 到 K 寄存器
4. h=1 在 P-192, P-224, P-256, P-384 和 P-521中
5. h=2 在 B-163, B-233, B-283, B-409, B-571 和 K-163中
6. h=4 在 K-233, K-283, K-409 和 K-571中
7. 设置 ECCOP(CRYPTO_ECC_CTL[10:9]) 为 00
8. 设置 FSEL(CRYPTO_ECC_CTL[8]) 依据使用的素数域或二进制域曲线
9. 设置 START(CRYPTO_ECC_CTL[0]) 为 1
10. 等待 BUSY(CRYPTO_ECC_STS[0]) 清0
11. 读 X1, Y1寄存器得到公钥Q

基于哈希密钥推导公式: DerivedKeyingMaterial = KDF(Z, 其它输入)
步骤1 For i = 1 to reps, 其中reps = ceil((Z长度, 其它输入)/哈希长度)

1. 依据表6.38-6写曲线参数A,B,N和曲线长度M到相应的寄存器
2. 据表6.38-6写接收方公共密钥 Q(x,y) 到 X1, Y1 寄存器

步骤 2 返回 DerivedKeyingMaterial = $Hash_1$ || $Hash_2$ || … || $Hash_{reps}$

注:

1. 其它输入详情请参考NIST SP 800-56A中的第46页, 有关 Pair-Wise Key Establishment Schemes Using Discrete Logarithm Cryptography
2. 有关辅因子详情请参考 “RECOMMENDED ELLIPTIC CURVES FOR FEDERAL GOVERNMENT USE”

6.38.6 寄存器映射

R: 只读, **W**: 只写, **R/W**: 可读可写

寄存器	偏移量	R/W	描述	复位值
CRYPTO 基地址: **CRYPTO_BA = 0x5008_0000**				
CRYPTO_INTEN	CRYPTO_BA+0x000	R/W	加密中断使能控制寄存器	0x0000_0000
CRYPTO_INTSTS	CRYPTO_BA+0x004	R/W	加密中断标志	0x0000_0000
CRYPTO_PRNG_CTL	CRYPTO_BA+0x008	R/W	PRNG 控制寄存器	0x0000_0000
CRYPTO_PRNG_SEED	CRYPTO_BA+0x00C	W	PRNG种子	Undefined
CRYPTO_PRNG_KEY0	CRYPTO_BA+0x010	R	PRNG 产生密钥0	Undefined
CRYPTO_PRNG_KEY1	CRYPTO_BA+0x014	R	PRNG产生密钥1	Undefined
CRYPTO_PRNG_KEY2	CRYPTO_BA+0x018	R	PRNG产生密钥2	Undefined
CRYPTO_PRNG_KEY3	CRYPTO_BA+0x01C	R	PRNG产生密钥3	Undefined
CRYPTO_PRNG_KEY4	CRYPTO_BA+0x020	R	PRNG产生密钥4	Undefined
CRYPTO_PRNG_KEY5	CRYPTO_BA+0x024	R	PRNG产生密钥5	Undefined
CRYPTO_PRNG_KEY6	CRYPTO_BA+0x028	R	PRNG产生密钥6	Undefined
CRYPTO_PRNG_KEY7	CRYPTO_BA+0x02C	R	PRNG产生密钥7	Undefined
CRYPTO_AES_FDBCK0	CRYPTO_BA+0x050	R	AES引擎加密操作后输出反馈数据	0x0000_0000
CRYPTO_AES_FDBCK1	CRYPTO_BA+0x054	R	AES引擎加密操作后输出反馈数据	0x0000_0000
CRYPTO_AES_FDBCK2	CRYPTO_BA+0x058	R	AES引擎加密操作后输出反馈数据	0x0000_0000
CRYPTO_AES_FDBCK3	CRYPTO_BA+0x05C	R	AES引擎加密操作后输出反馈数据	0x0000_0000
CRYPTO_TDES_FDBCKH	CRYPTO_BA+0x060	R	TDES/DES 引擎加密操作后输出反馈数据高位	0x0000_0000
CRYPTO_TDES_FDBCKL	CRYPTO_BA+0x064	R	TDES/DES 引擎加密操作后输出反馈数据低位	0x0000_0000
CRYPTO_AES_CTL	CRYPTO_BA+0x100	R/W	AES 控制寄存器	0x0000_0000
CRYPTO_AES_STS	CRYPTO_BA+0x104	R	AES 引擎标志	0x0001_0100
CRYPTO_AES_DATIN	CRYPTO_BA+0x108	R/W	AES 引擎数据输入端口寄存器	0x0000_0000
CRYPTO_AES_DATOUT	CRYPTO_BA+0x10C	R	AES 引擎数据输出端口寄存器	0x0000_0000
CRYPTO_AES0_KEY0	CRYPTO_BA+0x110	R/W	AES 通道0 密钥字0 寄存器	0x0000_0000
CRYPTO_AES0_KEY1	CRYPTO_BA+0x114	R/W	AES 通道0 密钥字1寄存器	0x0000_0000
CRYPTO_AES0_KEY2	CRYPTO_BA+0x118	R/W	AES 通道0 密钥字2 寄存器	0x0000_0000
CRYPTO_AES0_KEY3	CRYPTO_BA+0x11C	R/W	AES 通道0 密钥字3 寄存器	0x0000_0000
CRYPTO_AES0_KEY4	CRYPTO_BA+0x120	R/W	AES 通道0 密钥字4 寄存器	0x0000_0000
CRYPTO_AES0_KEY5	CRYPTO_BA+0x124	R/W	AES 通道0 密钥字5 寄存器	0x0000_0000

CRYPTO_AES0_KEY6	CRYPTO_BA+0x128	R/W	AES 通道0 密钥字6 寄存器	0x0000_0000
CRYPTO_AES0_KEY7	CRYPTO_BA+0x12C	R/W	AES 通道0 密钥字7 寄存器	0x0000_0000
CRYPTO_AES0_IV0	CRYPTO_BA+0x130	R/W	AES 通道0初始化向量字0 寄存器	0x0000_0000
CRYPTO_AES0_IV1	CRYPTO_BA+0x134	R/W	AES 通道0初始化向量字1 寄存器	0x0000_0000
CRYPTO_AES0_IV2	CRYPTO_BA+0x138	R/W	AES 通道0初始化向量字2 寄存器	0x0000_0000
CRYPTO_AES0_IV3	CRYPTO_BA+0x13C	R/W	AES 通道0初始化向量字3 寄存器	0x0000_0000
CRYPTO_AES0_SADDR	CRYPTO_BA+0x140	R/W	AES 信道0 DMA源地址寄存器	0x0000_0000
CRYPTO_AES0_DADDR	CRYPTO_BA+0x144	R/W	AES 通道0 DMA 目的地址寄存器	0x0000_0000
CRYPTO_AES0_CNT	CRYPTO_BA+0x148	R/W	AES 通道0字节数寄存器	0x0000_0000
CRYPTO_AES1_KEY0	CRYPTO_BA+0x14C	R/W	AES 通道1 密钥字0 寄存器	0x0000_0000
CRYPTO_AES1_KEY1	CRYPTO_BA+0x150	R/W	AES 通道1 密钥字1寄存器	0x0000_0000
CRYPTO_AES1_KEY2	CRYPTO_BA+0x154	R/W	AES 通道1 密钥字2 寄存器	0x0000_0000
CRYPTO_AES1_KEY3	CRYPTO_BA+0x158	R/W	AES 通道1 密钥字3 寄存器	0x0000_0000
CRYPTO_AES1_KEY4	CRYPTO_BA+0x15C	R/W	AES 通道1 密钥字4 寄存器	0x0000_0000
CRYPTO_AES1_KEY5	CRYPTO_BA+0x160	R/W	AES 通道1 密钥字5 寄存器	0x0000_0000
CRYPTO_AES1_KEY6	CRYPTO_BA+0x164	R/W	AES 通道1 密钥字6 寄存器	0x0000_0000
CRYPTO_AES1_KEY7	CRYPTO_BA+0x168	R/W	AES 通道1 密钥字7 寄存器	0x0000_0000
CRYPTO_AES1_IV0	CRYPTO_BA+0x16C	R/W	AES 通道1初始化向量字0 寄存器	0x0000_0000
CRYPTO_AES1_IV1	CRYPTO_BA+0x170	R/W	AES 通道1初始化向量字1 寄存器	0x0000_0000
CRYPTO_AES1_IV2	CRYPTO_BA+0x174	R/W	AES 通道1初始化向量字2 寄存器	0x0000_0000
CRYPTO_AES1_IV3	CRYPTO_BA+0x178	R/W	AES 通道1初始化向量字3 寄存器	0x0000_0000
CRYPTO_AES1_SADDR	CRYPTO_BA+0x17C	R/W	AES 信道1 DMA源地址寄存器	0x0000_0000
CRYPTO_AES1_DADDR	CRYPTO_BA+0x180	R/W	AES 通道1 DMA 目的地址寄存器	0x0000_0000
CRYPTO_AES1_CNT	CRYPTO_BA+0x184	R/W	AES 通道1字节数寄存器	0x0000_0000
CRYPTO_AES2_KEY0	CRYPTO_BA+0x188	R/W	AES 通道2 密钥字0 寄存器	0x0000_0000
CRYPTO_AES2_KEY1	CRYPTO_BA+0x18C	R/W	AES 通道2 密钥字1寄存器	0x0000_0000
CRYPTO_AES2_KEY2	CRYPTO_BA+0x190	R/W	AES 通道2 密钥字2 寄存器	0x0000_0000
CRYPTO_AES2_KEY3	CRYPTO_BA+0x194	R/W	AES 通道2 密钥字3 寄存器	0x0000_0000
CRYPTO_AES2_KEY4	CRYPTO_BA+0x198	R/W	AES 通道2 密钥字4 寄存器	0x0000_0000
CRYPTO_AES2_KEY5	CRYPTO_BA+0x19C	R/W	AES 通道2 密钥字5 寄存器	0x0000_0000
CRYPTO_AES2_KEY6	CRYPTO_BA+0x1A0	R/W	AES 通道2 密钥字6 寄存器	0x0000_0000
CRYPTO_AES2_KEY7	CRYPTO_BA+0x1A4	R/W	AES 通道2 密钥字7 寄存器	0x0000_0000
CRYPTO_AES2_IV0	CRYPTO_BA+0x1A8	R/W	AES 通道2初始化向量字0 寄存器	0x0000_0000

CRYPTO_AES2_IV1	CRYPTO_BA+0x1AC	R/W	AES 通道2初始化向量字1 寄存器	0x0000_0000
CRYPTO_AES2_IV2	CRYPTO_BA+0x1B0	R/W	AES 通道2初始化向量字2 寄存器	0x0000_0000
CRYPTO_AES2_IV3	CRYPTO_BA+0x1B4	R/W	AES 通道2初始化向量字3 寄存器	0x0000_0000
CRYPTO_AES2_SADDR	CRYPTO_BA+0x1B8	R/W	AES 信道2 DMA源地址寄存器	0x0000_0000
CRYPTO_AES2_DADDR	CRYPTO_BA+0x1BC	R/W	AES 通道2 DMA 目的地址寄存器	0x0000_0000
CRYPTO_AES2_CNT	CRYPTO_BA+0x1C0	R/W	AES 通道2字节数寄存器	0x0000_0000
CRYPTO_AES3_KEY0	CRYPTO_BA+0x1C4	R/W	AES 通道3 密钥字0 寄存器	0x0000_0000
CRYPTO_AES3_KEY1	CRYPTO_BA+0x1C8	R/W	AES 通道3 密钥字1寄存器	0x0000_0000
CRYPTO_AES3_KEY2	CRYPTO_BA+0x1CC	R/W	AES 通道3 密钥字2 寄存器	0x0000_0000
CRYPTO_AES3_KEY3	CRYPTO_BA+0x1D0	R/W	AES 通道3 密钥字3 寄存器	0x0000_0000
CRYPTO_AES3_KEY4	CRYPTO_BA+0x1D4	R/W	AES 通道3 密钥字4 寄存器	0x0000_0000
CRYPTO_AES3_KEY5	CRYPTO_BA+0x1D8	R/W	AES 通道3 密钥字5 寄存器	0x0000_0000
CRYPTO_AES3_KEY6	CRYPTO_BA+0x1DC	R/W	AES 通道3 密钥字6 寄存器	0x0000_0000
CRYPTO_AES3_KEY7	CRYPTO_BA+0x1E0	R/W	AES 通道3 密钥字7 寄存器	0x0000_0000
CRYPTO_AES3_IV0	CRYPTO_BA+0x1E4	R/W	AES 通道3初始化向量字0 寄存器	0x0000_0000
CRYPTO_AES3_IV1	CRYPTO_BA+0x1E8	R/W	AES 通道3初始化向量字1 寄存器	0x0000_0000
CRYPTO_AES3_IV2	CRYPTO_BA+0x1EC	R/W	AES 通道3初始化向量字2 寄存器	0x0000_0000
CRYPTO_AES3_IV3	CRYPTO_BA+0x1F0	R/W	AES 通道3初始化向量字3 寄存器	0x0000_0000
CRYPTO_AES3_SADDR	CRYPTO_BA+0x1F4	R/W	AES 信道3 DMA源地址寄存器	0x0000_0000
CRYPTO_AES3_DADDR	CRYPTO_BA+0x1F8	R/W	AES 通道3 DMA 目的地址寄存器	0x0000_0000
CRYPTO_AES3_CNT	CRYPTO_BA+0x1FC	R/W	AES 通道3字节数寄存器	0x0000_0000
CRYPTO_TDES_CTL	CRYPTO_BA+0x200	R/W	TDES/DES 控制寄存器	0x0000_0000
CRYPTO_TDES_STS	CRYPTO_BA+0x204	R	TDES/DES 引擎标志	0x0001_0100
CRYPTO_TDES0_KEY1H	CRYPTO_BA+0x208	R/W	TDES/DES 通道0 密钥 1 高位寄存器	0x0000_0000
CRYPTO_TDES0_KEY1L	CRYPTO_BA+0x20C	R/W	TDES/DES 通道0 密钥 1 低位寄存器	0x0000_0000
CRYPTO_TDES0_KEY2H	CRYPTO_BA+0x210	R/W	TDES/DES 通道0 密钥 2 高位寄存器	0x0000_0000
CRYPTO_TDES0_KEY2L	CRYPTO_BA+0x214	R/W	TDES/DES 通道0 密钥 2 低位寄存器	0x0000_0000
CRYPTO_TDES0_KEY3H	CRYPTO_BA+0x218	R/W	TDES/DES 通道0 密钥 3高位寄存器	0x0000_0000
CRYPTO_TDES0_KEY3L	CRYPTO_BA+0x21C	R/W	TDES/DES 通道0 密钥 3 低位寄存器	0x0000_0000
CRYPTO_TDES0_IVH	CRYPTO_BA+0x220	R/W	TDES/DES 通道0初始化向量高位寄存器	0x0000_0000

CRYPTO_TDES0_IVL	CRYPTO_BA+0x224	R/W	TDES/DES 通道0初始化向量低位寄存器	0x0000_0000
CRYPTO_TDES0_SADDR	CRYPTO_BA+0x228	R/W	TDES/DES 信道0 DMA 源地址寄存器	0x0000_0000
CRYPTO_TDES0_DADDR	CRYPTO_BA+0x22C	R/W	TDES/DES 通道0 DMA 目的地址寄存器	0x0000_0000
CRYPTO_TDES0_CNT	CRYPTO_BA+0x230	R/W	TDES/DES 通道0字节数寄存器	0x0000_0000
CRYPTO_TDES_DATIN	CRYPTO_BA+0x234	R/W	TDES/DES 引擎输入数据寄存器	0x0000_0000
CRYPTO_TDES_DATOUT	CRYPTO_BA+0x238	R	TDES/DES 引起输出数据寄存器	0x0000_0000
CRYPTO_TDES1_KEY1H	CRYPTO_BA+0x248	R/W	TDES/DES 通道1密钥 1 高位寄存器	0x0000_0000
CRYPTO_TDES1_KEY1L	CRYPTO_BA+0x24C	R/W	TDES/DES 通道1 密钥 1 低位寄存器	0x0000_0000
CRYPTO_TDES1_KEY2H	CRYPTO_BA+0x250	R/W	TDES/DES 通道1 密钥 2 高位寄存器	0x0000_0000
CRYPTO_TDES1_KEY2L	CRYPTO_BA+0x254	R/W	TDES/DES 通道1 密钥 2 低位寄存器	0x0000_0000
CRYPTO_TDES1_KEY3H	CRYPTO_BA+0x258	R/W	TDES/DES 通道1 密钥 3高位寄存器	0x0000_0000
CRYPTO_TDES1_KEY3L	CRYPTO_BA+0x25C	R/W	TDES/DES 通道1 密钥 3 低位寄存器	0x0000_0000
CRYPTO_TDES1_IVH	CRYPTO_BA+0x260	R/W	TDES/DES 通道1初始化向量高位寄存器	0x0000_0000
CRYPTO_TDES1_IVL	CRYPTO_BA+0x264	R/W	TDES/DES 通道1初始化向量低位寄存器	0x0000_0000
CRYPTO_TDES1_SADDR	CRYPTO_BA+0x268	R/W	TDES/DES 信道1 DMA 源地址寄存器	0x0000_0000
CRYPTO_TDES1_DADDR	CRYPTO_BA+0x26C	R/W	TDES/DES 通道1 DMA 目的地址寄存器	0x0000_0000
CRYPTO_TDES1_CNT	CRYPTO_BA+0x270	R/W	TDES/DES 通道1字节数寄存器	0x0000_0000
CRYPTO_TDES2_KEY1H	CRYPTO_BA+0x288	R/W	TDES/DES 通道2密钥 1 高位寄存器	0x0000_0000
CRYPTO_TDES2_KEY1L	CRYPTO_BA+0x28C	R/W	TDES/DES 通道2 密钥 1 低位寄存器	0x0000_0000
CRYPTO_TDES2_KEY2H	CRYPTO_BA+0x290	R/W	TDES/DES 通道2 密钥 2 高位寄存器	0x0000_0000
CRYPTO_TDES2_KEY2L	CRYPTO_BA+0x294	R/W	TDES/DES 通道2 密钥 2 低位寄存器	0x0000_0000
CRYPTO_TDES2_KEY3H	CRYPTO_BA+0x298	R/W	TDES/DES 通道2 密钥 3高位寄存器	0x0000_0000
CRYPTO_TDES2_KEY3L	CRYPTO_BA+0x29C	R/W	TDES/DES 通道2 密钥 3 低位寄存器	0x0000_0000
CRYPTO_TDES2_IVH	CRYPTO_BA+0x2A0	R/W	TDES/DES 通道2初始化向量高位寄存器	0x0000_0000
CRYPTO_TDES2_IVL	CRYPTO_BA+0x2A4	R/W	TDES/DES 通道2初始化向量低位寄存器	0x0000_0000
CRYPTO_TDES2_SADDR	CRYPTO_BA+0x2A8	R/W	TDES/DES 信道2 DMA 源地址寄存器	0x0000_0000

CRYPTO_TDES2_DADDR	CRYPTO_BA+0x2AC	R/W	TDES/DES 通道2 DMA 目的地址寄存器	0x0000_0000
CRYPTO_TDES2_CNT	CRYPTO_BA+0x2B0	R/W	TDES/DES 通道2字节数寄存器	0x0000_0000
CRYPTO_TDES3_KEY1H	CRYPTO_BA+0x2C8	R/W	TDES/DES 通道3 密钥 1 高位寄存器	0x0000_0000
CRYPTO_TDES3_KEY1L	CRYPTO_BA+0x2CC	R/W	TDES/DES 通道3 密钥 1 低位寄存器	0x0000_0000
CRYPTO_TDES3_KEY2H	CRYPTO_BA+0x2D0	R/W	TDES/DES 通道3 密钥 2 高位寄存器	0x0000_0000
CRYPTO_TDES3_KEY2L	CRYPTO_BA+0x2D4	R/W	TDES/DES 通道3 密钥 2 低位寄存器	0x0000_0000
CRYPTO_TDES3_KEY3H	CRYPTO_BA+0x2D8	R/W	TDES/DES 通道3 密钥 3高位寄存器	0x0000_0000
CRYPTO_TDES3_KEY3L	CRYPTO_BA+0x2DC	R/W	TDES/DES 通道3 密钥 3 低位寄存器	0x0000_0000
CRYPTO_TDES3_IVH	CRYPTO_BA+0x2E0	R/W	TDES/DES 通道3初始化向量高位寄存器	0x0000_0000
CRYPTO_TDES3_IVL	CRYPTO_BA+0x2E4	R/W	TDES/DES 通道3初始化向量低位寄存器	0x0000_0000
CRYPTO_TDES3_SADDR	CRYPTO_BA+0x2E8	R/W	TDES/DES 信道3 DMA 源地址寄存器	0x0000_0000
CRYPTO_TDES3_DADDR	CRYPTO_BA+0x2EC	R/W	TDES/DES 通道3 DMA 目的地址寄存器	0x0000_0000
CRYPTO_TDES3_CNT	CRYPTO_BA+0x2F0	R/W	TDES/DES 通道3字节数寄存器	0x0000_0000
CRYPTO_HMAC_CTL	CRYPTO_BA+0x300	R/W	SHA/HMAC 控制寄存器	0x0000_0000
CRYPTO_HMAC_STS	CRYPTO_BA+0x304	R	SHA/HMAC 状态标志	0x0000_0000
CRYPTO_HMAC_DGST0	CRYPTO_BA+0x308	R	SHA/HMAC 信息摘要 0	0x0000_0000
CRYPTO_HMAC_DGST1	CRYPTO_BA+0x30C	R	SHA/HMAC信息摘要1	0x0000_0000
CRYPTO_HMAC_DGST2	CRYPTO_BA+0x310	R	SHA/HMAC信息摘要2	0x0000_0000
CRYPTO_HMAC_DGST3	CRYPTO_BA+0x314	R	SHA/HMAC信息摘要3	0x0000_0000
CRYPTO_HMAC_DGST4	CRYPTO_BA+0x318	R	SHA/HMAC信息摘要4	0x0000_0000
CRYPTO_HMAC_DGST5	CRYPTO_BA+0x31C	R	SHA/HMAC信息摘要5	0x0000_0000
CRYPTO_HMAC_DGST6	CRYPTO_BA+0x320	R	SHA/HMAC信息摘要6	0x0000_0000
CRYPTO_HMAC_DGST7	CRYPTO_BA+0x324	R	SHA/HMAC信息摘要7	0x0000_0000
CRYPTO_HMAC_DGST8	CRYPTO_BA+0x328	R	SHA/HMAC信息摘要8	0x0000_0000
CRYPTO_HMAC_DGST9	CRYPTO_BA+0x32C	R	SHA/HMAC信息摘要9	0x0000_0000

CRYPTO_HMAC_DGST10	CRYPTO_BA+0x330	R	SHA/HMAC信息摘要10	0x0000_0000
CRYPTO_HMAC_DGST11	CRYPTO_BA+0x334	R	SHA/HMAC信息摘要11	0x0000_0000
CRYPTO_HMAC_DGST12	CRYPTO_BA+0x338	R	SHA/HMAC信息摘要12	0x0000_0000
CRYPTO_HMAC_DGST13	CRYPTO_BA+0x33C	R	SHA/HMAC信息摘要13	0x0000_0000
CRYPTO_HMAC_DGST14	CRYPTO_BA+0x340	R	SHA/HMAC信息摘要14	0x0000_0000
CRYPTO_HMAC_DGST15	CRYPTO_BA+0x344	R	SHA/HMAC信息摘要15	0x0000_0000
CRYPTO_HMAC_KEYCNT	CRYPTO_BA+0x348	R/W	SHA/HMAC 密钥字节数寄存器	0x0000_0000
CRYPTO_HMAC_SADDR	CRYPTO_BA+0x34C	R/W	SHA/HMAC DMA 源地址寄存器	0x0000_0000
CRYPTO_HMAC_DMACNT	CRYPTO_BA+0x350	R/W	SHA/HMAC 字节数寄存器	0x0000_0000
CRYPTO_HMAC_DATIN	CRYPTO_BA+0x354	R/W	SHA/HMAC引擎非-DMA 模式数据输入端口寄存器	0x0000_0000
CRYPTO_ECC_CTL	CRYPTO_BA+0x800	R/W	ECC 控制寄存器	0x0000_0000
CRYPTO_ECC_STS	CRYPTO_BA+0x804	R	ECC 状态寄存器	0x0000_0000
CRYPTO_ECC_X1_00	CRYPTO_BA+0x808	R/W	ECC 第一点的 X坐标字0	0x0000_0000
CRYPTO_ECC_X1_01	CRYPTO_BA+0x80C	R/W	ECC 第一点的 X坐标字1	0x0000_0000
CRYPTO_ECC_X1_02	CRYPTO_BA+0x810	R/W	ECC 第一点的 X坐标字2	0x0000_0000
CRYPTO_ECC_X1_03	CRYPTO_BA+0x814	R/W	ECC 第一点的 X坐标字3	0x0000_0000
CRYPTO_ECC_X1_04	CRYPTO_BA+0x818	R/W	ECC 第一点的 X坐标字4	0x0000_0000
CRYPTO_ECC_X1_05	CRYPTO_BA+0x81C	R/W	ECC 第一点的 X坐标字5	0x0000_0000
CRYPTO_ECC_X1_06	CRYPTO_BA+0x820	R/W	ECC 第一点的 X坐标字6	0x0000_0000
CRYPTO_ECC_X1_07	CRYPTO_BA+0x824	R/W	ECC 第一点的 X坐标字7	0x0000_0000
CRYPTO_ECC_X1_08	CRYPTO_BA+0x828	R/W	ECC 第一点的 X坐标字8	0x0000_0000
CRYPTO_ECC_X1_09	CRYPTO_BA+0x82C	R/W	ECC 第一点的 X坐标字9	0x0000_0000
CRYPTO_ECC_X1_10	CRYPTO_BA+0x830	R/W	ECC 第一点的 X坐标字10	0x0000_0000
CRYPTO_ECC_X1_11	CRYPTO_BA+0x834	R/W	ECC 第一点的 X坐标字11	0x0000_0000
CRYPTO_ECC_X1_12	CRYPTO_BA+0x838	R/W	ECC 第一点的 X坐标字12	0x0000_0000
CRYPTO_ECC_X1_13	CRYPTO_BA+0x83C	R/W	ECC 第一点的 X坐标字13	0x0000_0000
CRYPTO_ECC_X1_14	CRYPTO_BA+0x840	R/W	ECC 第一点的 X坐标字14	0x0000_0000
CRYPTO_ECC_X1_15	CRYPTO_BA+0x844	R/W	ECC 第一点的 X坐标字15	0x0000_0000
CRYPTO_ECC_X1_16	CRYPTO_BA+0x848	R/W	ECC 第一点的 X坐标字16	0x0000_0000
CRYPTO_ECC_X1_17	CRYPTO_BA+0x84C	R/W	ECC 第一点的 X坐标字17	0x0000_0000

CRYPTO_ECC_Y1_00	CRYPTO_BA+0x850	R/W	ECC 第一点的 Y坐标字0	0x0000_0000
CRYPTO_ECC_Y1_01	CRYPTO_BA+0x854	R/W	ECC 第一点的 Y坐标字1	0x0000_0000
CRYPTO_ECC_Y1_02	CRYPTO_BA+0x858	R/W	ECC 第一点的 Y坐标字2	0x0000_0000
CRYPTO_ECC_Y1_03	CRYPTO_BA+0x85C	R/W	ECC 第一点的 Y坐标字3	0x0000_0000
CRYPTO_ECC_Y1_04	CRYPTO_BA+0x860	R/W	ECC 第一点的 Y坐标字4	0x0000_0000
CRYPTO_ECC_Y1_05	CRYPTO_BA+0x864	R/W	ECC 第一点的 Y坐标字5	0x0000_0000
CRYPTO_ECC_Y1_06	CRYPTO_BA+0x868	R/W	ECC 第一点的 Y坐标字6	0x0000_0000
CRYPTO_ECC_Y1_07	CRYPTO_BA+0x86C	R/W	ECC 第一点的 Y坐标字7	0x0000_0000
CRYPTO_ECC_Y1_08	CRYPTO_BA+0x870	R/W	ECC 第一点的 Y坐标字8	0x0000_0000
CRYPTO_ECC_Y1_09	CRYPTO_BA+0x874	R/W	ECC 第一点的 Y坐标字9	0x0000_0000
CRYPTO_ECC_Y1_10	CRYPTO_BA+0x878	R/W	ECC 第一点的 Y坐标字10	0x0000_0000
CRYPTO_ECC_Y1_11	CRYPTO_BA+0x87C	R/W	ECC 第一点的 Y坐标字11	0x0000_0000
CRYPTO_ECC_Y1_12	CRYPTO_BA+0x880	R/W	ECC 第一点的 Y坐标字12	0x0000_0000
CRYPTO_ECC_Y1_13	CRYPTO_BA+0x884	R/W	ECC 第一点的 Y坐标字13	0x0000_0000
CRYPTO_ECC_Y1_14	CRYPTO_BA+0x888	R/W	ECC 第一点的 Y坐标字14	0x0000_0000
CRYPTO_ECC_Y1_15	CRYPTO_BA+0x88C	R/W	ECC 第一点的 Y坐标字15	0x0000_0000
CRYPTO_ECC_Y1_16	CRYPTO_BA+0x890	R/W	ECC 第一点的 Y坐标字16	0x0000_0000
CRYPTO_ECC_Y1_17	CRYPTO_BA+0x894	R/W	ECC 第一点的 Y坐标字17	0x0000_0000
CRYPTO_ECC_X2_00	CRYPTO_BA+0x898	R/W	ECC 第二点的 X坐标字0	0x0000_0000
CRYPTO_ECC_X2_01	CRYPTO_BA+0x89C	R/W	ECC 第二点的 X坐标字1	0x0000_0000
CRYPTO_ECC_X2_02	CRYPTO_BA+0x8A0	R/W	ECC 第二点的 X坐标字2	0x0000_0000
CRYPTO_ECC_X2_03	CRYPTO_BA+0x8A4	R/W	ECC 第二点的 X坐标字3	0x0000_0000
CRYPTO_ECC_X2_04	CRYPTO_BA+0x8A8	R/W	ECC 第二点的 X坐标字4	0x0000_0000
CRYPTO_ECC_X2_05	CRYPTO_BA+0x8AC	R/W	ECC 第二点的 X坐标字5	0x0000_0000
CRYPTO_ECC_X2_06	CRYPTO_BA+0x8B0	R/W	ECC 第二点的 X坐标字6	0x0000_0000
CRYPTO_ECC_X2_07	CRYPTO_BA+0x8B4	R/W	ECC 第二点的 X坐标字7	0x0000_0000
CRYPTO_ECC_X2_08	CRYPTO_BA+0x8B8	R/W	ECC 第二点的 X坐标字8	0x0000_0000
CRYPTO_ECC_X2_09	CRYPTO_BA+0x8BC	R/W	ECC 第二点的 X坐标字9	0x0000_0000
CRYPTO_ECC_X2_10	CRYPTO_BA+0x8C0	R/W	ECC 第二点的 X坐标字10	0x0000_0000
CRYPTO_ECC_X2_11	CRYPTO_BA+0x8C4	R/W	ECC 第二点的 X坐标字11	0x0000_0000
CRYPTO_ECC_X2_12	CRYPTO_BA+0x8C8	R/W	ECC 第二点的 X坐标字12	0x0000_0000
CRYPTO_ECC_X2_13	CRYPTO_BA+0x8CC	R/W	ECC 第二点的 X坐标字13	0x0000_0000
CRYPTO_ECC_X2_14	CRYPTO_BA+0x8D0	R/W	ECC 第二点的 X坐标字14	0x0000_0000

CRYPTO_ECC_X2_15	CRYPTO_BA+0x8D4	R/W	ECC 第二点的 X坐标字15	0x0000_0000
CRYPTO_ECC_X2_16	CRYPTO_BA+0x8D8	R/W	ECC 第二点的 X坐标字16	0x0000_0000
CRYPTO_ECC_X2_17	CRYPTO_BA+0x8DC	R/W	ECC 第二点的 X坐标字17	0x0000_0000
CRYPTO_ECC_Y2_00	CRYPTO_BA+0x8E0	R/W	ECC 第二点的 Y坐标字0	0x0000_0000
CRYPTO_ECC_Y2_01	CRYPTO_BA+0x8E4	R/W	ECC 第二点的 Y坐标字1	0x0000_0000
CRYPTO_ECC_Y2_02	CRYPTO_BA+0x8E8	R/W	ECC 第二点的 Y坐标字2	0x0000_0000
CRYPTO_ECC_Y2_03	CRYPTO_BA+0x8EC	R/W	ECC 第二点的 Y坐标字3	0x0000_0000
CRYPTO_ECC_Y2_04	CRYPTO_BA+0x8F0	R/W	ECC 第二点的 Y坐标字4	0x0000_0000
CRYPTO_ECC_Y2_05	CRYPTO_BA+0x8F4	R/W	ECC 第二点的 Y坐标字5	0x0000_0000
CRYPTO_ECC_Y2_06	CRYPTO_BA+0x8F8	R/W	ECC 第二点的 Y坐标字6	0x0000_0000
CRYPTO_ECC_Y2_07	CRYPTO_BA+0x8FC	R/W	ECC 第二点的 Y坐标字7	0x0000_0000
CRYPTO_ECC_Y2_08	CRYPTO_BA+0x900	R/W	ECC 第二点的 Y坐标字8	0x0000_0000
CRYPTO_ECC_Y2_09	CRYPTO_BA+0x904	R/W	ECC 第二点的 Y坐标字9	0x0000_0000
CRYPTO_ECC_Y2_10	CRYPTO_BA+0x908	R/W	ECC 第二点的 Y坐标字10	0x0000_0000
CRYPTO_ECC_Y2_11	CRYPTO_BA+0x90C	R/W	ECC 第二点的 Y坐标字11	0x0000_0000
CRYPTO_ECC_Y2_12	CRYPTO_BA+0x910	R/W	ECC 第二点的 Y坐标字12	0x0000_0000
CRYPTO_ECC_Y2_13	CRYPTO_BA+0x914	R/W	ECC 第二点的 Y坐标字13	0x0000_0000
CRYPTO_ECC_Y2_14	CRYPTO_BA+0x918	R/W	ECC 第二点的 Y坐标字14	0x0000_0000
CRYPTO_ECC_Y2_15	CRYPTO_BA+0x91C	R/W	ECC 第二点的 Y坐标字15	0x0000_0000
CRYPTO_ECC_Y2_16	CRYPTO_BA+0x920	R/W	ECC 第二点的 Y坐标字16	0x0000_0000
CRYPTO_ECC_Y2_17	CRYPTO_BA+0x924	R/W	ECC 第二点的 Y坐标字17	0x0000_0000
CRYPTO_ECC_A_00	CRYPTO_BA+0x928	R/W	ECC 椭圆曲线参数CURVEA 字 0	0x0000_0000
CRYPTO_ECC_A_01	CRYPTO_BA+0x92C	R/W	ECC 椭圆曲线参数CURVEA 字 1	0x0000_0000
CRYPTO_ECC_A_02	CRYPTO_BA+0x930	R/W	ECC 椭圆曲线参数CURVEA 字 2	0x0000_0000
CRYPTO_ECC_A_03	CRYPTO_BA+0x934	R/W	ECC 椭圆曲线参数CURVEA 字 3	0x0000_0000
CRYPTO_ECC_A_04	CRYPTO_BA+0x938	R/W	ECC 椭圆曲线参数CURVEA 字 4	0x0000_0000
CRYPTO_ECC_A_05	CRYPTO_BA+0x93C	R/W	ECC 椭圆曲线参数CURVEA 字 5	0x0000_0000
CRYPTO_ECC_A_06	CRYPTO_BA+0x940	R/W	ECC 椭圆曲线参数CURVEA 字 6	0x0000_0000
CRYPTO_ECC_A_07	CRYPTO_BA+0x944	R/W	ECC 椭圆曲线参数CURVEA 字 7	0x0000_0000
CRYPTO_ECC_A_08	CRYPTO_BA+0x948	R/W	ECC 椭圆曲线参数CURVEA 字 8	0x0000_0000
CRYPTO_ECC_A_09	CRYPTO_BA+0x94C	R/W	ECC 椭圆曲线参数CURVEA 字9	0x0000_0000
CRYPTO_ECC_A_10	CRYPTO_BA+0x950	R/W	ECC 椭圆曲线参数CURVEA 字 10	0x0000_0000
CRYPTO_ECC_A_11	CRYPTO_BA+0x954	R/W	ECC 椭圆曲线参数CURVEA 字 11	0x0000_0000

CRYPTO_ECC_A_12	CRYPTO_BA+0x958	R/W	ECC 椭圆曲线参数CURVEA 字 12	0x0000_0000
CRYPTO_ECC_A_13	CRYPTO_BA+0x95C	R/W	ECC 椭圆曲线参数CURVEA 字 13	0x0000_0000
CRYPTO_ECC_A_14	CRYPTO_BA+0x960	R/W	ECC 椭圆曲线参数CURVEA 字 14	0x0000_0000
CRYPTO_ECC_A_15	CRYPTO_BA+0x964	R/W	ECC 椭圆曲线参数CURVEA 字 15	0x0000_0000
CRYPTO_ECC_A_16	CRYPTO_BA+0x968	R/W	ECC 椭圆曲线参数CURVEA 字 16	0x0000_0000
CRYPTO_ECC_A_17	CRYPTO_BA+0x96C	R/W	ECC 椭圆曲线参数CURVEA 字 17	0x0000_0000
CRYPTO_ECC_B_00	CRYPTO_BA+0x970	R/W	ECC 椭圆曲线参数CURVEB 字 0	0x0000_0000
CRYPTO_ECC_B_01	CRYPTO_BA+0x974	R/W	ECC 椭圆曲线参数CURVEB 字 1	0x0000_0000
CRYPTO_ECC_B_02	CRYPTO_BA+0x978	R/W	ECC 椭圆曲线参数CURVEB 字 2	0x0000_0000
CRYPTO_ECC_B_03	CRYPTO_BA+0x97C	R/W	ECC 椭圆曲线参数CURVEB 字 3	0x0000_0000
CRYPTO_ECC_B_04	CRYPTO_BA+0x980	R/W	ECC 椭圆曲线参数CURVEB 字 4	0x0000_0000
CRYPTO_ECC_B_05	CRYPTO_BA+0x984	R/W	ECC 椭圆曲线参数CURVEB 字 5	0x0000_0000
CRYPTO_ECC_B_06	CRYPTO_BA+0x988	R/W	ECC 椭圆曲线参数CURVEB 字 6	0x0000_0000
CRYPTO_ECC_B_07	CRYPTO_BA+0x98C	R/W	ECC 椭圆曲线参数CURVEB 字 7	0x0000_0000
CRYPTO_ECC_B_08	CRYPTO_BA+0x990	R/W	ECC 椭圆曲线参数CURVEB 字 8	0x0000_0000
CRYPTO_ECC_B_09	CRYPTO_BA+0x994	R/W	ECC 椭圆曲线参数CURVEB 字9	0x0000_0000
CRYPTO_ECC_B_10	CRYPTO_BA+0x998	R/W	ECC 椭圆曲线参数CURVEB 字 10	0x0000_0000
CRYPTO_ECC_B_11	CRYPTO_BA+0x99C	R/W	ECC 椭圆曲线参数CURVEB 字 11	0x0000_0000
CRYPTO_ECC_B_12	CRYPTO_BA+0x9A0	R/W	ECC 椭圆曲线参数CURVEB 字 12	0x0000_0000
CRYPTO_ECC_B_13	CRYPTO_BA+0x9A4	R/W	ECC 椭圆曲线参数CURVEB 字 13	0x0000_0000
CRYPTO_ECC_B_14	CRYPTO_BA+0x9A8	R/W	ECC 椭圆曲线参数CURVEB 字 14	0x0000_0000
CRYPTO_ECC_B_15	CRYPTO_BA+0x9AC	R/W	ECC 椭圆曲线参数CURVEB 字 15	0x0000_0000
CRYPTO_ECC_B_16	CRYPTO_BA+0x9B0	R/W	ECC 椭圆曲线参数CURVEB 字 16	0x0000_0000
CRYPTO_ECC_B_17	CRYPTO_BA+0x9B4	R/W	ECC 椭圆曲线参数CURVEB 字 17	0x0000_0000
CRYPTO_ECC_N_00	CRYPTO_BA+0x9B8	R/W	ECC 椭圆曲线参数CURVEN 字 0	0x0000_0000
CRYPTO_ECC_N_01	CRYPTO_BA+0x9BC	R/W	ECC 椭圆曲线参数CURVEN 字 1	0x0000_0000
CRYPTO_ECC_N_02	CRYPTO_BA+0x9C0	R/W	ECC 椭圆曲线参数CURVEN 字 2	0x0000_0000
CRYPTO_ECC_N_03	CRYPTO_BA+0x9C4	R/W	ECC 椭圆曲线参数CURVEN 字 3	0x0000_0000
CRYPTO_ECC_N_04	CRYPTO_BA+0x9C8	R/W	ECC 椭圆曲线参数CURVEN 字 4	0x0000_0000
CRYPTO_ECC_N_05	CRYPTO_BA+0x9CC	R/W	ECC 椭圆曲线参数CURVEN 字 5	0x0000_0000
CRYPTO_ECC_N_06	CRYPTO_BA+0x9D0	R/W	ECC 椭圆曲线参数CURVEN 字 6	0x0000_0000
CRYPTO_ECC_N_07	CRYPTO_BA+0x9D4	R/W	ECC 椭圆曲线参数CURVEN 字 7	0x0000_0000
CRYPTO_ECC_N_08	CRYPTO_BA+0x9D8	R/W	ECC 椭圆曲线参数CURVEN 字 8	0x0000_0000

CRYPTO_ECC_N_09	CRYPTO_BA+0x9DC	R/W	ECC 椭圆曲线参数CURVEN 字9	0x0000_0000
CRYPTO_ECC_N_10	CRYPTO_BA+0x9E0	R/W	ECC 椭圆曲线参数CURVEN 字 10	0x0000_0000
CRYPTO_ECC_N_11	CRYPTO_BA+0x9E4	R/W	ECC 椭圆曲线参数CURVEN 字 11	0x0000_0000
CRYPTO_ECC_N_12	CRYPTO_BA+0x9E8	R/W	ECC 椭圆曲线参数CURVEN 字 12	0x0000_0000
CRYPTO_ECC_N_13	CRYPTO_BA+0x9EC	R/W	ECC 椭圆曲线参数CURVEN 字 13	0x0000_0000
CRYPTO_ECC_N_14	CRYPTO_BA+0x9F0	R/W	ECC 椭圆曲线参数CURVEN 字 14	0x0000_0000
CRYPTO_ECC_N_15	CRYPTO_BA+0x9F4	R/W	ECC 椭圆曲线参数CURVEN 字 15	0x0000_0000
CRYPTO_ECC_N_16	CRYPTO_BA+0x9F8	R/W	ECC 椭圆曲线参数CURVEN 字 16	0x0000_0000
CRYPTO_ECC_N_17	CRYPTO_BA+0x9FC	R/W	ECC 椭圆曲线参数CURVEN 字 17	0x0000_0000
CRYPTO_ECC_K_00	CRYPTO_BA+0xA00	W	ECC 点乘标量 SCALARK 字 0	0x0000_0000
CRYPTO_ECC_K_01	CRYPTO_BA+0xA04	W	ECC 点乘标量 SCALARK 字 1	0x0000_0000
CRYPTO_ECC_K_02	CRYPTO_BA+0xA08	W	ECC 点乘标量 SCALARK 字 2	0x0000_0000
CRYPTO_ECC_K_03	CRYPTO_BA+0xA0C	W	ECC 点乘标量 SCALARK 字 3	0x0000_0000
CRYPTO_ECC_K_04	CRYPTO_BA+0xA10	W	ECC 点乘标量 SCALARK 字 4	0x0000_0000
CRYPTO_ECC_K_05	CRYPTO_BA+0xA14	W	ECC 点乘标量 SCALARK 字 5	0x0000_0000
CRYPTO_ECC_K_06	CRYPTO_BA+0xA18	W	ECC 点乘标量 SCALARK 字 6	0x0000_0000
CRYPTO_ECC_K_07	CRYPTO_BA+0xA1C	W	ECC 点乘标量 SCALARK 字 7	0x0000_0000
CRYPTO_ECC_K_08	CRYPTO_BA+0xA20	W	ECC 点乘标量 SCALARK 字 8	0x0000_0000
CRYPTO_ECC_K_09	CRYPTO_BA+0xA24	W	ECC 点乘标量 SCALARK 字 9	0x0000_0000
CRYPTO_ECC_K_10	CRYPTO_BA+0xA28	W	ECC 点乘标量 SCALARK 字 10	0x0000_0000
CRYPTO_ECC_K_11	CRYPTO_BA+0xA2C	W	ECC 点乘标量 SCALARK 字 11	0x0000_0000
CRYPTO_ECC_K_12	CRYPTO_BA+0xA30	W	ECC 点乘标量 SCALARK 字 12	0x0000_0000
CRYPTO_ECC_K_13	CRYPTO_BA+0xA34	W	ECC 点乘标量 SCALARK 字 13	0x0000_0000
CRYPTO_ECC_K_14	CRYPTO_BA+0xA38	W	ECC 点乘标量 SCALARK 字 14	0x0000_0000
CRYPTO_ECC_K_15	CRYPTO_BA+0xA3C	W	ECC 点乘标量 SCALARK 字 15	0x0000_0000
CRYPTO_ECC_K_16	CRYPTO_BA+0xA40	W	ECC 点乘标量 SCALARK 字 16	0x0000_0000
CRYPTO_ECC_K_17	CRYPTO_BA+0xA44	W	ECC 点乘标量 SCALARK 字 17	0x0000_0000
CRYPTO_ECC_SADDR	CRYPTO_BA+0xA48	R/W	ECC DMA源地址寄存器	0x0000_0000
CRYPTO_ECC_DADDR	CRYPTO_BA+0xA4C	R/W	ECC DMA 目的地址寄存器	0x0000_0000
CRYPTO_ECC_STARTREG	CRYPTO_BA+0xA50	R/W	ECC 更新起始地址寄存器	0x0000_0000
CRYPTO_ECC_WORDCNT	CRYPTO_BA+0xA54	R/W	ECC DMA 字数寄存器	0x0000_0000

6.38.7 寄存器描述

6.38.7.1 加密寄存器

CRYPTO_INTEN 加密中断使能控制寄存器

寄存器	偏移量	R/W	描述	复位值
CRYPTO_INTEN	CRYPTO_BA+0x000	R/W	加密中断使能控制寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved						HMACEIEN	HMACIEN
23	22	21	20	19	18	17	16
ECCEIEN	ECCIEN	Reserved					PRNGIEN
15	14	13	12	11	10	9	8
Reserved						TDESEIEN	TDESIEN
7	6	5	4	3	2	1	0
Reserved						AESEIEN	AESIEN

位	描述	
[31:26]	Reserved	保留.
[25]	HMACEIEN	SHA/HMAC 错误中断使能控制 0 = SHA/HMAC 错误中断标志禁止. 1 = SHA/HMAC 错误中断标志使能
[24]	HMACIEN	SHA/HMAC 中断使能控制 0 = SHA/HMAC中断禁止. 1 = SHA/HMAC中断使能. 在 DMA 模式,当SHA _DMA_CNT设定数量的数据存入SHA/HMAC引擎后会产生一个中断，在非-DMA模式, 当SHA/HMAC引擎完成操作以后会触发一个中断.
[23]	ECCEIEN	ECC 错误中断使能控制 0 = ECC 错误终端标志禁止 1 = ECC 错误中断标志使能
[22]	ECCIEN	ECC 中断使能控制 0 = ECC中断禁止. 1 = ECC中断使能. 在 DMA 模式，当ECC_DMA_CNT 设定数量的数据参入ECC 引擎后会触发一个中断 在非-DMA 模式, 当 ECC 引擎完成操作后会触发一个中断
[21:17]	Reserved	保留.
[16]	PRNGIEN	PRNG 中断使能位 0 = PRNG 中断禁止. 1 = PRNG 中断使能.

[15:10]	**Reserved**	**TDES/DES 错误标志使能控制** 0 = TDES/DES 错误中断标志禁止. 1 = TDES/DES 错误中断标志使能
[9]	**TDESEIEN**	**TDES/DES 中断使能控制** 0 = TDES/DES 中断禁止. 1 = TDES/DES 中断使能. 在 DMA 模式, 当TDES_DMA_CNT 设定数量的数据存入 TDES 引擎后会触发一个中断 在非-DMA 模式, 当 TDES 引擎完成操作后会触发一个中断.
[8]	**TDESIEN**	保留.
[7:2]	**Reserved**	**AES 错误标志使能控制** 0 = AES 错误中断标志禁止. 1 = AES 错误中断标志使能.
[1]	**AESEIEN**	**AES 中断使能控制** 0 = AES 中断禁止. 1 = AES 中断使能. 在 DMA模式, 当AES_DMA_CNT 设定数量的数据存入AES 引擎会触发一个中断 在非-DMA 模式, 当AES 引擎完成操作以后会触发一个中断.
[0]	**AESIEN**	保留.

CRYPTO_INTSTS 加密中断标志寄存器

寄存器	偏移量	R/W	描述	复位值
CRYPTO_INTSTS	CRYPTO_BA+0x004	R/W	加密中断标志寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved						HMACEIF	HMACIF
23	22	21	20	19	18	17	16
ECCEIF	ECCIF	Reserved					PRNGIF
15	14	13	12	11	10	9	8
Reserved						TDESEIF	TDESIF
7	6	5	4	3	2	1	0
Reserved						AESEIF	AESIF

位	描述	
[31:26]	**Reserved**	保留
[25]	**HMACEIF**	**SHA/HMAC 错误标志** 这个寄存器包括操作和设置错误，具体的标志见CRYPTO_HMAC_STS寄存器. 此位写1清0，写0无效. 0 = 没有 SHA/HMAC 错误. 1 = SHA/HMAC 错误中断.
[24]	**HMACIF**	**SHA/HMAC 完成中断标志** 此位写1清0，写0无效. 0 = 没有 SHA/HMAC中断 1 = SHA/HMAC操作完成中断.
[23]	**ECCEIF**	**ECC 错误标志** 这个寄存器包括操作和设置错误，具体的标志见CRYPTO_ECC_STS寄存器. 此位写1清0，写0无效 0 = 没有 ECC 错误. 1 = ECC 错误中断.
[22]	**ECCIF**	**ECC 完成中断标志** 此位写1清0，写0无效. 0 = 没有 ECC中断. 1 = ECC 操作完成中断.
[21:17]	**Reserved**	保留.
[16]	**PRNGIF**	**PRNG完成中断标志** 此位写1清0，写0无效. 0 = 没有 PRNG 中断. 1 = PRNG 密钥产生完成中断.

[15:10]	**Reserved**	保留.
[9]	**TDESEIF**	**TDES/DES 错误标志** 这个寄存器包括操作和设置错误，具体的标志见CRYPTO_TDES_STS寄存器. 此位写1清0，写0无效 0 = 没有 TDES/DES 错误. 1 = TDES/DES 加解密错误中断.
[8]	**TDESIF**	**TDES/DES 完成中断标志** 此位写1清0，写0无效 0 = 没有 TDES/DES中断 1 = TDES/DES 加解密完成中断
[7:2]	**Reserved**	保留.
[1]	**AESEIF**	**AES 错误标志** 此位写1清0，写0无效. 0 = 没有 AES错误. 1 = AES 加解密错误中断.
[0]	**AESIF**	**AES 完成中断标志** 此位写1清0，写0无效. 0 = 没有 AES中断 1= AES加解密完成中断.

6.38.7.2 PRNG 寄存器

CRYPTO_PRNG_CTL PRNG 控制寄存器

寄存器	偏移量	R/W	描述	复位值
CRYPTO_PRNG_CTL	CRYPTO_BA+0x008	R/W	PRNG控制寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							BUSY
7	6	5	4	3	2	1	0
Reserved				KEYSZ		SEEDRLD	START

位	描述	
[31:9]	Reserved	保留.
[8]	BUSY	PRNG 忙 (只读) 0 = PRNG 引擎空闲. 1 = 表明 PRNG 引擎正在产生 CRYPTO_PRNG_KEYx.
[7:4]	Reserved	保留
[3:2]	KEYSZ	PRNG 产生密钥大小 00 = 64位. 01 = 128位. 10 = 192位. 11 = 256位.
[1]	SEEDRLD	PRNG 引擎重载新的种子 0 = 根据目前种子产生密钥. 1 = 重载新的种子
[0]	START	开始 PRNG引擎 0 = 停止 PRNG 引擎. 1 = 产生新的密钥，并把密钥存入寄存器CRYPTO_PRNG_KEYx（会在新密钥产生时清除）

CRYPTO_PRNG_SEED PRNG 种子寄存器

寄存器	偏移量	R/W	描述	复位值
CRYPTO_PRNG_SEED	CRYPTO_BA+0x00C	W	PRNG 种子	Undefined

31	30	29	28	27	26	25	24
SEED							
23	22	21	20	19	18	17	16
SEED							
15	14	13	12	11	10	9	8
SEED							
7	6	5	4	3	2	1	0
SEED							

位	描述	
[31:0]	SEED	PRNG种子 (只写) 这里存放 PRNG 引擎的种子

CRYPTO_PRNG_KEYx PRNG 密钥 x 寄存器

寄存器	偏移量	R/W	描述	复位值
CRYPTO_PRNG_KEY0	CRYPTO_BA+0x010	R	PRNG 产生密钥 0	Undefined
CRYPTO_PRNG_KEY1	CRYPTO_BA+0x014	R	PRNG产生密钥 1	Undefined
CRYPTO_PRNG_KEY2	CRYPTO_BA+0x018	R	PRNG产生密钥 2	Undefined
CRYPTO_PRNG_KEY3	CRYPTO_BA+0x01C	R	PRNG产生密钥 3	Undefined
CRYPTO_PRNG_KEY4	CRYPTO_BA+0x020	R	PRNG产生密钥 4	Undefined
CRYPTO_PRNG_KEY5	CRYPTO_BA+0x024	R	PRNG产生密钥 5	Undefined
CRYPTO_PRNG_KEY6	CRYPTO_BA+0x028	R	PRNG产生密钥 6	Undefined
CRYPTO_PRNG_KEY7	CRYPTO_BA+0x02C	R	PRNG产生密钥 7	Undefined

31	30	29	28	27	26	25	24
KEY							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
KEY							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
KEY							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
KEY							

位	描述	
[31:0]	**KEY**	**存储 PRNG 产生的密钥 (只读)** 这里存储 PRNG产生的密钥.

6.38.7.3 AES 寄存器

AES 控制寄存器 (CRYPTO_AES_CTL)

寄存器	偏移量	R/W	描述	复位值
CRYPTO_AES_CTL	CRYPTO_BA+0x100	R/W	AES 控制寄存器	0x0000_0000

31	30	29	28	27	26	25	24
KEYPRT	KEYUNPRT					CHANNEL	
23	22	21	20	19	18	17	16
INSWAP	OUTSWAP	Reserved					ENCRYPTO
15	14	13	12	11	10	9	8
OPMODE							
7	6	5	4	3	2	1	0
DMAEN	DMACSCAD	DMALAST	Reserved	KEYSZ		STOP	START

位	描述	
[31]	**KEYPRT**	**保护密钥** 读到一个标志反映 KEYPRT. 0 = 没影响. 1 = 保护AES密钥的内容不被读取。读 CRYPTO_AESn_KEYx 寄存器的值不是CRYPTO_AESn_KEYx寄存器的内容.当此位置1，可以被KEYUNPRT清除，但密钥的内容也会被清除
[30:26]	**KEYUNPRT**	**不保护密钥** 写 0 到 CRYPTO_AES_CTL[31]且写 “10110” 到 CRYPTO_AES_CTL[30:26]来解除保护AES密钥. KEYUNPRT可以读可以写，当AES 引擎正在操作时写入, BUSY标志是 1, 写入无效.
[25:24]	**CHANNEL**	**AES 引擎工作信道** 00 = 目前控制寄存器设定用于通道 0. 01 = 目前控制寄存器设定用于通道1. 10 = 目前控制寄存器设定用于通道2. 11 = 目前控制寄存器设定用于通道 3.
[23]	**INSWAP**	**AES引擎输入数据交换** 0 = 保持原有顺序. 1 = CPU 存入加速器的数据顺序从{byte3, byte2, byte1, byte0} 变为 {byte0, byte1, byte2, byte3}.
[22]	**OUTSWAP**	**AES 引擎输出数据交换** 0 = 保持原有顺序. 1 = 从加速器存入 CPU 的数据顺序从 {byte3, byte2, byte1, byte0} 变为 {byte0, byte1, byte2, byte3}.
[17]	**Reserved**	保留.

[16]	**ENCRYPTO**	**AES 加密/解密** 0 = AES 引擎执行解密操作. 1 = AES 引擎执行加密操作.
[15:8]	**OPMODE**	**AES 引擎操作模式** 0x00 = ECB (电子密码本模式) 0x01 = CBC (密码块链接模式). 0x02 = CFB (密码反馈模式). 0x03 = OFB (输出反馈模式). 0x04 = CTR (计数器模式). 0x10 = CBC-CS1 (CBC 密文窃取模式 1). 0x11 = CBC-CS2 (CBC密文窃取模式2). 0x12 = CBC-CS3 (CBC密文窃取模式3).
[7]	**DMAEN**	**AES引擎 DMA 使能控制** 0 = AES DMA引擎禁止 AES 引擎工作在非-DMA模式, 从端口 CRYPTO_AES_DATIN获取数据. 1 = AES_DMA引擎使能. AES 引擎工作在DMA模式,从引擎写入读出数据由DMA逻辑完成.
[6]	**DMACSCAD**	**AES 引擎 DMA 级联模式** 0 = DMA 级联模式禁止 1 = 在 DMA 级联模式,程序可以一次级联操作中更新 DMA s源地址寄存器, 目的地址寄存器, 和字节数寄存器,不需要等加速器操作完成
[5]	**DMALAST**	**AES 最后一个数据块** 在 DMA 模式,这一位需要在在开始最后DMA级联循环前置1. 在非-DMA 模式, 在ECB, CBC, CTR, OFB, 和 CFB 模式下存入最后一笔数据时置1, 在 CBC-CS1, CBC-CS2, 个 CBC-CS3 模式下存入最后一笔的前一笔数据时置1. 此位总是读为0，.当 START 触发需要重新写入
[3:2]	**KEYSZ**	**AES 密钥大小** 这里定义三种不同 AES操作密钥大小. 2'b00 = 128 位密钥. 2'b01 = 192位密钥. 2'b10 = 256位密钥. 2'b11 = 保留. 如果 AES 加速器正在操作且相应的标志BUSY为 1, 更新此位无效
[1]	**STOP**	**AES 引擎停止** 0 = 没效果. 1 = 停止 AES 引擎. **注意:** 此位总是读为0.
[0]	**START**	**AES 引擎开始** 0 = 没效果. 1 = 开始t AES 引擎. BUSY 标志置1. **注意:** 此位总是读为0.

CRYPTO_AES_STS AES 状态标志寄存器

寄存器	偏移量	R/W	描述	复位值
CRYPTO_AES_STS	CRYPTO_BA+0x104	R	AES引擎标志	0x0001_0100

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved			BUSERR	Reserved	OUTBUFERR	OUTBUFFULL	OUTBUFEMPTY
15	14	13	12	11	10	9	8
Reserved			CNTERR	Reserved	INBUFERR	INBUFFULL	INBUFEMPTY
7	6	5	4	3	2	1	0
Reserved							BUSY

位	描述	
[31:21]	**Reserved**	保留.
[20]	**BUSERR**	**AES DMA 访问总线错误标志** 0 = 没有错误. 1 = 总线错误会停止 DMA操作和 AES 引擎.
[19]	**Reserved**	保留
[18]	**OUTBUFERR**	**AES输出缓存错误标志** 0 = 没有错误. 1 = 从 AES 引擎获取数据时发生错误.
[17]	**OUTBUFFULL**	**AES 输出缓存满标志** 0 = AES 输出缓存未满. 1 = AES 输出缓存已满, 程序需要从 CRYPTO_AES_DATOUT读取数据，不然. AES 引擎会因为输出缓存满而等待
[16]	**OUTBUFEMPTY**	**AES输出缓存空** 0 = AES 输出缓存非空。输出缓存里仍有有效数据. 1 = AES 输出缓存已空。程序不能再从CRYPTO_AES_DATOUT读数据，.标志 OUTBUFERR 当输出缓存空时会置1.
[15:13]	**Reserved**	保留
[12]	**CNTERR**	**CRYPTO_AESn_CNT 设置错误** 0 = CRYPTO_AESn_CNT 设置没有错误. 1 = CRYPTO_AESn_CNT在 ECB, CBC, CFB, OFB, 和 CTR 模式下（如果CRYPTO_AES_CTL[7]使能）没有乘以16.
[11]	**Reserved**	保留.

[10]	**INBUFERR**	**AES 输入缓存错误标志** 0 = 没有错误. 1 = 写数据到 AES 引擎时发生错误.
[9]	**INBUFFULL**	**AES输入缓存满标志** 0 = AES 输入缓存未满. 程序可以向 AES 引擎写入数据. 1 = AES 输入缓存已满. 程序不能向 AES 引擎写入数据. 标志 INBUFERR 会置 1.
[8]	**INBUFEMPTY**	**AES 输入缓存空** 0 = 输入缓存里有一些数据等待 AES 引擎执行. 1 = AES 输入缓存空. 程序需要向 AES引擎写入数据. 另外, AES 引擎会挂起等待输入数据.
[7:1]	**Reserved**	保留.
[0]	**BUSY**	**AES 引擎忙** 0 = AES 引擎在空闲或完成态. 1 = AES 引擎正在执行.

CRYPTO_AES_DATIN AES 数据输入端口寄存器

寄存器	偏移量	R/W	描述	复位值
CRYPTO_AES_DATIN	CRYPTO_BA+0x108	R/W	AES引擎数据输入端口寄存器	0x0000_0000

31	30	29	28	27	26	25	24
DATIN							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
DATIN							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
DATIN							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
DATIN							

位	描述	
[31:0]	**DATIN**	**AES 引擎输入端口** CPU 通过这个端口向 AES 引擎存入数据，确认CRYPTO_AES_STS的状态，当 INBUFFULL 是 0时存入数据.

CRYPTO_AES_DATOUT AES 数据输出端口寄存器

寄存器	偏移量	R/W	描述	复位值
CRYPTO_AES_DATOUT	CRYPTO_BA+0x10C	R	AES 引擎数据输出端口寄存器	0x0000_0000

31	30	29	28	27	26	25	24
DATOUT							
23	22	21	20	19	18	17	16
DATOUT							
15	14	13	12	11	10	9	8
DATOUT							
7	6	5	4	3	2	1	0
DATOUT							

位	描述	
[31:0]	DATOUT	**AES 引擎输出端口** CPU 从这里得到 AES 引擎输出的数据，需确认CRYPTO_AES_STS寄存器，当OUTBUFEMPTY 为 0时读取数据.

AES密钥字 x 寄存器(CRYPTO_AES0_KEYx)

寄存器	偏移量	R/W	描述	复位值
CRYPTO_AES0_KEY0	CRYPTO_BA+0x110	R/W	AES 通道0 密钥字0 寄存器	0x0000_0000
CRYPTO_AES0_KEY1	CRYPTO_BA+0x114	R/W	AES 通道0 密钥字1寄存器	0x0000_0000
CRYPTO_AES0_KEY2	CRYPTO_BA+0x118	R/W	AES 通道0 密钥字2 寄存器	0x0000_0000
CRYPTO_AES0_KEY3	CRYPTO_BA+0x11C	R/W	AES 通道0 密钥字3 寄存器	0x0000_0000
CRYPTO_AES0_KEY4	CRYPTO_BA+0x120	R/W	AES 通道0 密钥字4 寄存器	0x0000_0000
CRYPTO_AES0_KEY5	CRYPTO_BA+0x124	R/W	AES 通道0 密钥字5 寄存器	0x0000_0000
CRYPTO_AES0_KEY6	CRYPTO_BA+0x128	R/W	AES 通道0 密钥字6 寄存器	0x0000_0000
CRYPTO_AES0_KEY7	CRYPTO_BA+0x12C	R/W	AES 通道0 密钥字7 寄存器	0x0000_0000
CRYPTO_AES1_KEY0	CRYPTO_BA+0x14C	R/W	AES 通道1 密钥字0 寄存器	0x0000_0000
CRYPTO_AES1_KEY1	CRYPTO_BA+0x150	R/W	AES 通道1 密钥字1寄存器	0x0000_0000
CRYPTO_AES1_KEY2	CRYPTO_BA+0x154	R/W	AES 通道1 密钥字2 寄存器	0x0000_0000
CRYPTO_AES1_KEY3	CRYPTO_BA+0x158	R/W	AES 通道1 密钥字3 寄存器	0x0000_0000
CRYPTO_AES1_KEY4	CRYPTO_BA+0x15C	R/W	AES 通道1 密钥字4 寄存器	0x0000_0000
CRYPTO_AES1_KEY5	CRYPTO_BA+0x160	R/W	AES 通道1 密钥字5 寄存器	0x0000_0000
CRYPTO_AES1_KEY6	CRYPTO_BA+0x164	R/W	AES 通道1 密钥字6 寄存器	0x0000_0000
CRYPTO_AES1_KEY7	CRYPTO_BA+0x168	R/W	AES 通道1 密钥字7 寄存器	0x0000_0000
CRYPTO_AES2_KEY0	CRYPTO_BA+0x188	R/W	AES 通道2 密钥字0 寄存器	0x0000_0000
CRYPTO_AES2_KEY1	CRYPTO_BA+0x18C	R/W	AES 通道2 密钥字1寄存器	0x0000_0000
CRYPTO_AES2_KEY2	CRYPTO_BA+0x190	R/W	AES 通道2 密钥字2 寄存器	0x0000_0000
CRYPTO_AES2_KEY3	CRYPTO_BA+0x194	R/W	AES 通道2 密钥字3 寄存器	0x0000_0000
CRYPTO_AES2_KEY4	CRYPTO_BA+0x198	R/W	AES 通道2 密钥字4 寄存器	0x0000_0000
CRYPTO_AES2_KEY5	CRYPTO_BA+0x19C	R/W	AES 通道2 密钥字5 寄存器	0x0000_0000

CRYPTO_AES2_KEY6	CRYPTO_BA+0x1A0	R/W	AES 通道2 密钥字6 寄存器	0x0000_0000
CRYPTO_AES2_KEY7	CRYPTO_BA+0x1A4	R/W	AES 通道2 密钥字7 寄存器	0x0000_0000
CRYPTO_AES3_KEY0	CRYPTO_BA+0x1C4	R/W	AES 通道3 密钥字0 寄存器	0x0000_0000
CRYPTO_AES3_KEY1	CRYPTO_BA+0x1C8	R/W	AES 通道3 密钥字1寄存器	0x0000_0000
CRYPTO_AES3_KEY2	CRYPTO_BA+0x1CC	R/W	AES 通道3 密钥字2 寄存器	0x0000_0000
CRYPTO_AES3_KEY3	CRYPTO_BA+0x1D0	R/W	AES 通道3 密钥字3 寄存器	0x0000_0000
CRYPTO_AES3_KEY4	CRYPTO_BA+0x1D4	R/W	AES 通道3 密钥字4 寄存器	0x0000_0000
CRYPTO_AES3_KEY5	CRYPTO_BA+0x1D8	R/W	AES 通道3 密钥字5 寄存器	0x0000_0000
CRYPTO_AES3_KEY6	CRYPTO_BA+0x1DC	R/W	AES 通道3 密钥字6 寄存器	0x0000_0000
CRYPTO_AES3_KEY7	CRYPTO_BA+0x1E0	R/W	AES 通道3 密钥字7 寄存器	0x0000_0000

31	30	29	28	27	26	25	24
KEY							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
KEY							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
KEY							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
KEY							

位	描述	
[31:0]	**KEY**	**CRYPTO_AESn_KEYx** KEY 保存AES 操作的安全密钥. n = 0, 1..3. x = 0, 1..7. AES 加速器的安全密钥可以是128, 192,或256位，需要4，6，8个32位寄存器来存放。{CRYPTO_AESn_KEY3, CRYPTO_AESn_KEY2, CRYPTO_AESn_KEY1, CRYPTO_AESn_KEY0}存放128位的AES操作安全密钥. {CRYPTO_AESn_KEY5, CRYPTO_AESn_KEY4, CRYPTO_AESn_KEY3, CRYPTO_AESn_KEY2, CRYPTO_AESn_KEY1, CRYPTO_AESn_KEY0} 存放 192位AES 操作安全密钥. {CRYPTO_AESn_KEY7, CRYPTO_AESn_KEY6, CRYPTO_AESn_KEY5, CRYPTO_AESn_KEY4, CRYPTO_AESn_KEY3, CRYPTO_AESn_KEY2, CRYPTO_AESn_KEY1, CRYPTO_AESn_KEY0} 存放 256位 AES操作安全密钥.

AES 初始化向量字 x 寄存器(CRYPTO_AES0_IVx)

寄存器	偏移量	R/W	描述	复位值
CRYPTO_AES0_IV0	CRYPTO_BA+0x130	R/W	AES 通道0初始化向量字0寄存器	0x0000_0000
CRYPTO_AES0_IV1	CRYPTO_BA+0x134	R/W	AES 通道0初始化向量字1寄存器	0x0000_0000
CRYPTO_AES0_IV2	CRYPTO_BA+0x138	R/W	AES 通道0初始化向量字2寄存器	0x0000_0000
CRYPTO_AES0_IV3	CRYPTO_BA+0x13C	R/W	AES 通道0初始化向量字3寄存器	0x0000_0000
CRYPTO_AES1_IV0	CRYPTO_BA+0x16C	R/W	AES 通道1初始化向量字0寄存器	0x0000_0000
CRYPTO_AES1_IV1	CRYPTO_BA+0x170	R/W	AES 通道1初始化向量字1寄存器	0x0000_0000
CRYPTO_AES1_IV2	CRYPTO_BA+0x174	R/W	AES 通道1初始化向量字2寄存器	0x0000_0000
CRYPTO_AES1_IV3	CRYPTO_BA+0x178	R/W	AES 通道1初始化向量字3寄存器	0x0000_0000
CRYPTO_AES2_IV0	CRYPTO_BA+0x1A8	R/W	AES 通道2初始化向量字0寄存器	0x0000_0000
CRYPTO_AES2_IV1	CRYPTO_BA+0x1AC	R/W	AES 通道2初始化向量字1寄存器	0x0000_0000
CRYPTO_AES2_IV2	CRYPTO_BA+0x1B0	R/W	AES 通道2初始化向量字2寄存器	0x0000_0000
CRYPTO_AES2_IV3	CRYPTO_BA+0x1B4	R/W	AES 通道2初始化向量字3寄存器	0x0000_0000
CRYPTO_AES3_IV0	CRYPTO_BA+0x1E4	R/W	AES 通道3初始化向量字0寄存器	0x0000_0000
CRYPTO_AES3_IV1	CRYPTO_BA+0x1E8	R/W	AES 通道3初始化向量字1寄存器	0x0000_0000
CRYPTO_AES3_IV2	CRYPTO_BA+0x1EC	R/W	AES 通道3初始化向量字2寄存器	0x0000_0000
CRYPTO_AES3_IV3	CRYPTO_BA+0x1F0	R/W	AES 通道3初始化向量字3寄存器	0x0000_0000

31	30	29	28	27	26	25	24
IV							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
IV							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
IV							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
IV							

位	描述	
[31:0]	**IV**	**AES 初始向量** n = 0, 1..3. x = 0, 1..3. 4个初始化向量 (CRYPTO_AESn_IV0, CRYPTO_AESn_IV1, CRYPTO_AESn_IV2, 和 CRYPTO_AESn_IV3) 是AES工作于CBC, CFB, 和 OFB模式需要的. 4个寄存器 (CRYPTO_AESn_IV0, CRYPTO_AESn_IV1, CRYPTO_AESn_IV2, 和 CRYPTO_AESn_IV3) 在 AES 工作在 CTR 模式时做为临时计数器.

AES DMA 源地址寄存器(CRYPTO_AES0_SADDR)

寄存器	偏移量	R/W	描述	复位值
CRYPTO_AES0_SADDR	CRYPTO_BA+0x140	R/W	AES DMA信道 0源地址寄存器	0x0000_0000
CRYPTO_AES1_SADDR	CRYPTO_BA+0x17C	R/W	AES DMA信道 1源地址寄存器	0x0000_0000
CRYPTO_AES2_SADDR	CRYPTO_BA+0x1B8	R/W	AES DMA信道 2源地址寄存器	0x0000_0000
CRYPTO_AES3_SADDR	CRYPTO_BA+0x1F4	R/W	AES DMA信道 3源地址寄存器	0x0000_0000

31	30	29	28	27	26	25	24
SADDR							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
SADDR							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
SADDR							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
SADDR							

位	描述	
[31:0]	**SADDR**	**AES DMA 源地址** AES 加速器支持 DMA 功能在SRAM内存和内建FIFO之间传输原始文本。 SADDR存放源文本存放的数据缓存的源地址。基于源地址, AES 加速器可以从SRAM内存读到原始文本(加密)/加密文本(解密)来执行AES 操作. 源地址必须是字对齐，也就是说，SADDR位0和位1会被忽略。 SADDR 可读可写，当AES正在操作时写SADDR 不会影响目前的 AES操作，但 SADDR 寄存器的值会在稍后更新。因此，程序可以为下一次AES 操作准备源地址. 在 DMA 模式, 程序可以在触发START 前更新下一笔CRYPTO_AESn_SADDR. CRYPTO_AESn_SADDR 和 CRYPTO_AESn_DADDR 的值可以相同.

AES DMA目的地址寄存器(CRYPTO_AES0_DADDR)

寄存器	偏移量	R/W	描述	复位值
CRYPTO_AES0_DADDR	CRYPTO_BA+0x144	R/W	AES通道0 DMA目的地址寄存器	0x0000_0000
CRYPTO_AES1_DADDR	CRYPTO_BA+0x180	R/W	AES通道1DMA目的地址寄存器	0x0000_0000
CRYPTO_AES2_DADDR	CRYPTO_BA+0x1BC	R/W	AES通道2 DMA目的地址寄存器	0x0000_0000
CRYPTO_AES3_DADDR	CRYPTO_BA+0x1F8	R/W	AES通道3 DMA目的地址寄存器	0x0000_0000

31	30	29	28	27	26	25	24
DADDR							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
DADDR							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
DADDR							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
DADDR							

位	描述	
[31:0]	**DADDR**	**AES DMA 目的地址** AES 加速器支持DMA 功能来在SRAM内存和内建FIFO间传输加密文本。. DADDR存放引擎输出文本存放的数据缓存的目的地址。基于目的地址， AES 加速器可以在AES 操作完成之后将加密文本(加密)/原始文本(解密)写回SRAM内存。目的地址的开始必须是字对齐，DADDR 的位0和位1会被忽略. DADDR 可读可写。在AES正在操作时写入DADDR 不会影响目前的 AES操作，但DADDR 会在稍后更新，因此程序可以为下次AES操作准备目的地址的值.. 在DMA 模式, 程序可以在触发START 前更新下一个CRYPTO_AESn_DADDR. CRYPTO_AESn_SADDR 和 CRYPTO_AESn_DADDR 的值可以相同.

AES 字节数寄存器 (CRYPTO_AES0_CNT)

寄存器	偏移量	R/W	描述	复位值
CRYPTO_AES0_CNT	CRYPTO_BA+0x148	R/W	AES 通道0字节数寄存器	0x0000_0000
CRYPTO_AES1_CNT	CRYPTO_BA+0x184	R/W	AES 通道1字节数寄存器	0x0000_0000
CRYPTO_AES2_CNT	CRYPTO_BA+0x1C0	R/W	AES 通道2字节数寄存器	0x0000_0000
CRYPTO_AES3_CNT	CRYPTO_BA+0x1FC	R/W	AES 通道3字节数寄存器	0x0000_0000

31	30	29	28	27	26	25	24
CNT							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
CNT							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
CNT							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
CNT							

位	描述	
[31:0]	**CNT**	**AES 字节数** 在 AES 引擎工作在 DMA 模式时，CRYPTO_AESn_CNT 存放源文本的字节数，CRYPTO_AESn_CNT 是32位，所以最大字节数是 4G. CRYPTO_AESn_CNT 可读可写。当 AES 加速器正在操作时写入CRYPTO_AESn_CNT 不会影响目前的AES操作。但是CRYPTO_AESn_CNT 会在稍后更新。因此, 程序可以为下一次AES操作准备字节数. 依照 CBC-CS1, CBC-CS2, 和 CBC-CS3 标准, 操作数据的个数最少要一个块。少于一个块会导致不可预料的结果. 在非-DMA ECB, CBC, CFB, OFB, 和 CTR 模式, CRYPTO_AESn_CNT在存入最后一个数据块的数据前必须写入最后一个数据块的字节数。在非-DMA CBC-CS1, CBC-CS2, 和 CBC-CS3 模式, CRYPTO_AESn_CNT在存入最后两个数据块的数据前必须写入最后两个数据块的字节数.

AES反馈 x 寄存器 (CRYPTO_AES_FDBCKx)

寄存器	偏移量	R/W	描述	复位值
CRYPTO_AES_FDBCK0	CRYPTO_BA+0x050	R	AES 引擎加密操作后输出反馈数据	0x0000_0000
CRYPTO_AES_FDBCK1	CRYPTO_BA+0x054	R	AES 引擎加密操作后输出反馈数据	0x0000_0000
CRYPTO_AES_FDBCK2	CRYPTO_BA+0x058	R	AES 引擎加密操作后输出反馈数据	0x0000_0000
CRYPTO_AES_FDBCK3	CRYPTO_BA+0x05C	R	AES 引擎加密操作后输出反馈数据	0x0000_0000

31	30	29	28	27	26	25	24
FDBCK							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
FDBCK							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
FDBCK							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
FDBCK							

位	描述	
[31:0]	**FDBCK**	**AES反馈信息** 反馈值是 128 位大小. 在 DMA 级联模式 AES 引擎使用 CRYPTO_AES_FDBCKx 的数据作为输入到 CRYPTO_AESn_IVx的下一个块 . AES 引擎输出下次快操作的IV的反馈信息..。程序可以用这个反馈信息实现多于4个DMA 通道。程序可以暂存反馈值。当切换回来，向同一个通道的这个寄存器写入这个暂存值，下面的操作会依照当初的设定.

6.38.7.4 TDES/DES 寄存器

CRYPTO_TDES_CTL TDES/DES 控制寄存器

寄存器	偏移量	R/W	描述	复位值
CRYPTO_TDES_CTL	CRYPTO_BA+0x200	R/W	TDES/DES 控制寄存器	0x0000_0000

31	30	29	28	27	26	25	24
KEYPRT	KEYUNPRT					CHANNEL	
23	22	21	20	19	18	17	16
INSWAP	OUTSWAP	BLKSWAP	Reserved				ENCRYPTO
15	14	13	12	11	10	9	8
Reserved					OPMODE		
7	6	5	4	3	2	1	0
DMAEN	DMACSCAD	DMALAST	Reserved	3KEYS	TMODE	STOP	START

位	描述	
[31]	**KEYPRT**	**保护密钥** 读到 KEYPRT的标志. 0 = 没影响. 1 = 这一位保护TDES密钥不被读出。读 CRYPTO_ TDESn_KEYxH/L 读到的值不是 CRYPTO_ TDESn_KEYxH/的内容. 当此位置1，可以通过访问KEYUNPRT清0，密钥的内容也同时被清除.
[30:26]	**KEYUNPRT**	**不保护密钥** 写 0到 CRYPTO_TDES_CTL [31] 写 “10110” 到 CRYPTO_TDES_CTL [30:26] 解保护 TDES 密钥. KEYUNPRT 可读写，当TDES 引擎正在操作时写入， BUSY 标志是 1,不会改变 KEYUNPRT.
[25:24]	**CHANNEL**	**TDES/DES 引擎工作信道** 00 = 目前控制寄存器设定为通道 0. 01 = 目前控制寄存器设定为通道 1. 10 = 目前控制寄存器设定为通道 2. 11 = 目前控制寄存器设定为通道 3.
[23]	**INSWAP**	**TDES/DES 引擎输入数据交换** 0 = 保持原始顺序. 1 = CPU 存入加速器的数据顺序从{byte3, byte2, byte1, byte0} 变为 {byte0, byte1, byte2, byte3}.
[22]	**OUTSWAP**	**TDES/DES 引擎输出数据交换** 0 = 保持原始顺序. 1 = CPU 输出数据到加速器的顺序从 {byte3, byte2, byte1, byte0} 变为 {byte0, byte1, byte2, byte3}.

[21]	**BLKSWAP**	**TDES/DES 引擎模块双字交换** 0 = 保持原始顺序. **{WORD_H, WORD_L}**. 1 = 当此位置1, TDES 改变高低位顺序**{WORD_L, WORD_H}.**
[20:17]	**Reserved**	保留.
[16]	**ENCRYPTO**	**TDES/DES 加密/解密** 0 = TDES引擎执行解密操作. 1 = TDES 引擎执行加密操作.
[15:11]	**Reserved**	保留.
[10:8]	**OPMODE**	**TDES/DES 引擎工作模式** 0x00 = ECB (电子密码本模式). 0x01 = CBC (密码块链接模式). 0x02 = CFB (密码反馈模式). 0x03 = OFB (输出反馈模式). 0x04 = CTR (计数器模式). Others = CTR (计数器模式).
[7]	**DMAEN**	**TDES/DES 引擎 DMA 使能控制** 0 = TDES_DMA引擎禁止. TDES 引擎工作在非-DMA 模式, 从端口CRYPTO_TDES_DATIN读取数据. 1 = TDES_DMA 引擎使能. TDES 引擎工作在 DMA 模式, 数据搬运由 DMA 逻辑完成.
[6]	**DMACSCAD**	**TDES/DES 引擎 DMA 级联模式** 0 = DMA 级联功能禁止. 1 = 在 DMA 级联模式, 程序可以在没有完成加速器操作前更新DMA 源地址寄存器, 目的地址寄存器, 字节数寄存器
[5]	**DMALAST**	**TDES/DES 引起开始最后一个块** 在 DMA 模式，这一位必须在开始最后一笔DMA 级联循环前置1. 在非-DMA 模式, 这一位在存入最后一个块前置1.
[4]	**Reserved**	保留.
[3]	**3KEYS**	**TDES/DES 密钥编号** 0 = TDES/DES引擎选择KEY1 和 KEY2. 1 = TDES/DES引擎三个密钥都使能.
[2]	**TMODE**	**TDES/DES 引擎操作模式** 0 = TDES/DES引擎设置 DES模式. 1 = TDES/DES引擎设置三重 DES 模式.
[1]	**STOP**	**TDES/DES 引擎停止** 0 = 没影响. 1 = 停止 TDES/DES 引擎. **注意:** 读此位总为0.

[0]	**START**	**TDES/DES 引擎开始** 0 = 没影响. 1 = 开始 TDES/DES 引擎.标志 BUSY 会置1. **注意: 读此位总为0**.

CRYPTO_TDES0_STS TDES/DES 状态标志寄存器

寄存器	偏移量	R/W	描述	复位值
CRYPTO_TDES_STS	CRYPTO_BA+0x204	R	TDES/DES 引擎标志	0x0001_0100

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved			BUSERR	Reserved	OUTBUFERR	OUTBUFFULL	OUTBUFEMPTY
15	14	13	12	11	10	9	8
Reserved					INBUFERR	INBUFFULL	INBUFEMPTY
7	6	5	4	3	2	1	0
Reserved							BUSY

位	描述	
[31:21]	**Reserved**	保留.
[20]	**BUSERR**	**TDES/DES DMA 访问总线出错标志** 0 = 没有错误 1 = 总线错误会停止 DMA操作和 TDES/DES引擎.
[19]	**Reserved**	保留.
[18]	**OUTBUFERR**	**TDES/DES输出缓存错误标志** 0 = 没有错误. 1 = 从TDES/DES 引擎读出测试结果时出错.
[17]	**OUTBUFFULL**	**TDES/DES 输出缓存满标志** 0 = TDES/DES 输出缓存未满. 1 = TDES/DES 输出缓存满了,程序需要从 TDES_DATA_OUT读出数据。. DES/DES 引擎会挂起因为输出缓存满了.
[16]	**OUTBUFEMPTY**	**TDES/DES 输出换成空标志** 0 = TDES/DES 输出缓存非空. 输出缓存里有一些有效数据. 1 = TDES/DES 输出缓存空, 程序不能从 TDES_DATA_OUT读取数据。标志 OUTBUFERR会置 1, 因为输出缓存空了.
[15:11]	**Reserved**	保留.
[10]	**INBUFERR**	**TDES/DES输入缓存错误标志** 0 = 没有错误. 1 = 数据存入 TDES/DES引擎.时发生错误
[9]	**INBUFFULL**	**TDES/DES输入缓存满标志** 0 = TDES/DES输入缓存未满. 程序可以向 TDES/DES引擎填入数据. 1 = TDES输入缓存满了. 程序不能向 TDES/DES 引擎填入数据. 标志 INBUFERR 会置 1.

[8]	**INBUFEMPTY**	**TDES/DES 输入缓存空** 0 = 输入缓存中仍有一些数据等待 TDES/DES引擎执行. 1 = TDES/DES 输入缓存空. 程序需要填入数据到TDES/DES引擎 TDES/DES 会挂起等待输入数据
[7:1]	**Reserved**	保留.
[0]	**BUSY**	**TDES/DES 引擎忙** 0 = TDES/DES 引擎在空闲或完成态. 1 = TDES/DES 引擎正在执行.

TDES_KEY1H/L, TDES_KEY2H/L, TDES_KEY3H/L TDES/DES 密钥 1, 2, 3 高低字寄存器

寄存器	偏移量	R/W	描述	复位值
CRYPTO_TDES0_KEY1H	CRYPTO_BA+0x208	R/W	TDES/DES 通道0 密钥 1 高位字	0x0000_0000
CRYPTO_TDES0_KEY1L	CRYPTO_BA+0x20C	R/W	TDES/DES 通道0 密钥 1 低位字	0x0000_0000
CRYPTO_TDES0_KEY2H	CRYPTO_BA+0x210	R/W	TDES/DES 通道0 密钥 2 高位字	0x0000_0000
CRYPTO_TDES0_KEY2L	CRYPTO_BA+0x214	R/W	TDES/DES 通道0 密钥 2 低位字	0x0000_0000
CRYPTO_TDES0_KEY3H	CRYPTO_BA+0x218	R/W	TDES/DES 通道0 密钥 3 高位字	0x0000_0000
CRYPTO_TDES0_KEY3L	CRYPTO_BA+0x21C	R/W	TDES/DES 通道0 密钥 3 低位字	0x0000_0000
CRYPTO_TDES1_KEY1H	CRYPTO_BA+0x248	R/W	TDES/DES 通道1 密钥 1 高位字	0x0000_0000
CRYPTO_TDES1_KEY1L	CRYPTO_BA+0x24C	R/W	TDES/DES 通道1 密钥 1 低位字	0x0000_0000
CRYPTO_TDES1_KEY2H	CRYPTO_BA+0x250	R/W	TDES/DES 通道1 密钥 2 高位字	0x0000_0000
CRYPTO_TDES1_KEY2L	CRYPTO_BA+0x254	R/W	TDES/DES 通道1 密钥 2 低位字	0x0000_0000
CRYPTO_TDES1_KEY3H	CRYPTO_BA+0x258	R/W	TDES/DES 通道1 密钥 3 高位字	0x0000_0000
CRYPTO_TDES1_KEY3L	CRYPTO_BA+0x25C	R/W	TDES/DES 通道1 密钥 3 低位字	0x0000_0000
CRYPTO_TDES2_KEY1H	CRYPTO_BA+0x288	R/W	TDES/DES 通道2 密钥 1 高位字	0x0000_0000
CRYPTO_TDES2_KEY1L	CRYPTO_BA+0x28C	R/W	TDES/DES 通道2 密钥 1 低位字	0x0000_0000
CRYPTO_TDES2_KEY2H	CRYPTO_BA+0x290	R/W	TDES/DES 通道2 密钥 2 高位字	0x0000_0000
CRYPTO_TDES2_KEY2L	CRYPTO_BA+0x294	R/W	TDES/DES 通道2 密钥 2 低位字	0x0000_0000
CRYPTO_TDES2_KEY3H	CRYPTO_BA+0x298	R/W	TDES/DES 通道2 密钥 3 高位字	0x0000_0000
CRYPTO_TDES2_KEY3L	CRYPTO_BA+0x29C	R/W	TDES/DES 通道2 密钥 3 低位字	0x0000_0000
CRYPTO_TDES3_KEY1H	CRYPTO_BA+0x2C8	R/W	TDES/DES 通道3 密钥 1 高位字	0x0000_0000
CRYPTO_TDES3_KEY1L	CRYPTO_BA+0x2CC	R/W	TDES/DES 通道3 密钥 1 低位字	0x0000_0000
CRYPTO_TDES3_KEY2H	CRYPTO_BA+0x2D0	R/W	TDES/DES 通道3 密钥 2 高位字	0x0000_0000
CRYPTO_TDES3_KEY2L	CRYPTO_BA+0x2D4	R/W	TDES/DES 通道3 密钥 2 低位字	0x0000_0000

CRYPTO_TDES3_KEY3H	CRYPTO_BA+0x2D8	R/W	TDES/DES 通道3 密钥 3 高位字	0x0000_0000
CRYPTO_TDES3_KEY3L	CRYPTO_BA+0x2DC	R/W	TDES/DES 通道3 密钥 3 低位字	0x0000_0000

31	30	29	28	27	26	25	24
KEY							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
KEY							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
KEY							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
KEY							

位	描述	
[31:0]	**KEY**	**TDES/DES 密钥高/低字** TDES/DES 算法运算密钥寄存器 TDES/DES 加速器的安全密钥是64位. T需要两个 32-位寄存器存储安全密钥. 寄存器 CRYPTO_TDESn_KEYxH用来存放位 [63:32]， 寄存器 CRYPTO_TDESn_KEYxL用来存放位 [31:0].

CRYPTO_TDES0_IVH/L, CRYPTO_TDES1_IVH/L, CRYPTO_TDES2_IVH/L, CRYPTO_TDES3_IVH/L TDES/DES IV 高低字寄存器

寄存器	偏移量	R/W	描述	复位值
CRYPTO_TDES0_IVH	CRYPTO_BA+0x220	R/W	TDES/DES 通道0 初始化向量高字寄存器	0x0000_0000
CRYPTO_TDES0_IVL	CRYPTO_BA+0x224	R/W	TDES/DES 通道0 初始化向量低字寄存器	0x0000_0000
CRYPTO_TDES1_IVH	CRYPTO_BA+0x260	R/W	TDES/DES 通道1 初始化向量高字寄存器	0x0000_0000
CRYPTO_TDES1_IVL	CRYPTO_BA+0x264	R/W	TDES/DES 通道1 初始化向量低字寄存器	0x0000_0000
CRYPTO_TDES2_IVH	CRYPTO_BA+0x2A0	R/W	TDES/DES 通道2 初始化向量高字寄存器	0x0000_0000
CRYPTO_TDES2_IVL	CRYPTO_BA+0x2A4	R/W	TDES/DES 通道2 初始化向量低字寄存器	0x0000_0000
CRYPTO_TDES3_IVH	CRYPTO_BA+0x2E0	R/W	TDES/DES 通道3 初始化向量高字寄存器	0x0000_0000
CRYPTO_TDES3_IVL	CRYPTO_BA+0x2E4	R/W	TDES/DES 通道3 初始化向量低字寄存器	0x0000_0000

31	30	29	28	27	26	25	24
IV							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
IV							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
IV							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
IV							

位	描述	
[31:0]	**IV**	**TDES/DES 初始化向量高/低字** 初始化向量 (IV) 在 TDES/DES引擎的 CBC, CFB, 和 OFB模式时使用. IV在CTR模式作为临时计数器使用.

CRYPTO_TDES0_SADDR, CRYPTO_TDES1_SADDR, CRYPTO_TDES2_SADDR, CRYPTO_TDES3_SADDR TDES/DES DMA 源地址寄存器

寄存器	偏移量	R/W	描述	复位值
CRYPTO_TDES0_SADDR	CRYPTO_BA+0x228	R/W	TDES/DES 信道0 DMA源地址寄存器	0x0000_0000
CRYPTO_TDES1_SADDR	CRYPTO_BA+0x268	R/W	TDES/DES 信道1 DMA源地址寄存器	0x0000_0000
CRYPTO_TDES2_SADDR	CRYPTO_BA+0x2A8	R/W	TDES/DES 信道2 DMA源地址寄存器	0x0000_0000
CRYPTO_TDES3_SADDR	CRYPTO_BA+0x2E8	R/W	TDES/DES 信道3 DMA源地址寄存器	0x0000_0000

31	30	29	28	27	26	25	24
SADDR							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
SADDR							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
SADDR							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
SADDR							

位	描述	
[31:0]	**SADDR**	**TDES/DES DMA 源地址** TDES/DES 加速器支持DMA 功能来在SRAM内存和内建FIFO间传输原始文本。CRYPTO_TDESn_SADDR 存放原始文本存放的数据缓存的源地址。基于这个源地址，TDES/DES 加速器可以从SRAM内存读到原始文本(加密)/加密文本(解密)并且执行 TDES/DES 操作. 源地址起始地址需要字对齐，就是说CRYPTO_TDESn_SADDR的位1和位0会被忽略. CRYPTO_TDESn_SADDR 可读可写，在TDES/DES 加速器正在操作时写CRYPTO_TDESn_SADDR不会影响目前的 TDES/DES 操作. 但是 CRYPTO_TDESn_SADDR 的值会在稍后更新，因此，程序可以为下次TDES/DES 操作准备 DMA 源地址. 在 DMA 模式, 程序可以在触发START 之前更新下一笔 CRYPTO_TDESn_SADDR. CRYPTO_TDESn_SADDR 和 CRYPTO_TDESn_DADDR 值可以相同.

CRYPTO_TDES0_DADDR, CRYPTO_TDES1_DADDR, CRYPTO_TDES2_DADDR, CRYPTO_TDES3_DADDR TDES/DES DMA 目的地址寄存器

寄存器	偏移量	R/W	描述	复位值
CRYPTO_TDES0_DADDR	CRYPTO_BA+0x22C	R/W	TDES/DES 通道0 DMA 目的地址寄存器	0x0000_0000
CRYPTO_TDES1_DADDR	CRYPTO_BA+0x26C	R/W	TDES/DES 通道1 DMA 目的地址寄存器	0x0000_0000
CRYPTO_TDES2_DADDR	CRYPTO_BA+0x2AC	R/W	TDES/DES 通道2 DMA 目的地址寄存器	0x0000_0000
CRYPTO_TDES3_DADDR	CRYPTO_BA+0x2EC	R/W	TDES/DES 通道3 DMA 目的地址寄存器	0x0000_0000

31	30	29	28	27	26	25	24
DADDR							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
DADDR							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
DADDR							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
DADDR							

位	描述	
[31:0]	**DADDR**	**TDES/DES DMA 目的地址** TDES/DES 加速器支持DMA 功能在SRAM内存和内建FIFO间传输加密文本。CRYPTO_TDESn_DADDR 存放引擎输出文本存放的数据缓存的目的地址. 基于这个目的地址, TDES/DES 加速器可以在TDES/DES 操作完成之后把加密文本(加密)/原始文本(解密)写入SRAM内存。.目的地址的开始位置必须是字对齐，就是说 CRYPTO_TDESn_DADDR 的位0和位1会被忽略. CRYPTO_TDESn_DADDR 可读可写。.在TDES/DES操作时写入 CRYPTO_TDESn_DADDR 不会影响当前的 TDES/DES 操作，但CRYPTO_TDESn_DADDR的值会在稍后更新，因此，程序可以为下一笔TDES/DES操作准备目的地址。. 在 DMA 模式, 程序可以在触发START 前更新系一个CRYPTO_TDESn_DA DDR CRYPTO_TDESn_SADDDR 和 CRYPTO_TDESn_DA DDR的值可以相同.

CRYPTO_TDES0_CNT, CRYPTO_TDES1_CNT, CRYPTO_TDES2_CNT, CRYPTO_TDES3_CNT TDES/DES 块计数寄存器

寄存器	偏移量	R/W	描述	复位值
CRYPTO_TDES0_CNT	CRYPTO_BA+0x230	R/W	TDES/DES 通道0字节计数寄存器	0x0000_0000
CRYPTO_TDES1_CNT	CRYPTO_BA+0x270	R/W	TDES/DES 通道1字节计数寄存器	0x0000_0000
CRYPTO_TDES2_CNT	CRYPTO_BA+0x2B0	R/W	TDES/DES 通道2字节计数寄存器	0x0000_0000
CRYPTO_TDES3_CNT	CRYPTO_BA+0x2F0	R/W	TDES/DES 通道3字节计数寄存器	0x0000_0000

31	30	29	28	27	26	25	24
CNT							
23	22	21	20	19	18	17	16
CNT							
15	14	13	12	11	10	9	8
CNT							
7	6	5	4	3	2	1	0
CNT							

位	描述	
[31:0]	**CNT**	**TDES/DES 字节计数** TDES/DES 工作在DMA 模式，CRYPTO_TDESn_CNT 存放原始文本的字节数，CRYPTO_TDESn_CNT 是32位，所以最大字节数是 4G 字节 . CRYPTO_TDESn_CNT 可读可写。当TDES/DES 正在工作时写CRYPTO_TDESn_CNT 不会影响目前的 TDES/DES 操作。但CRYPTO_TDESn_CNT 的值会在稍后更新，因此程序可以为下次TDES/DES操作准备好字节数. 在非-DMA ECB, CBC, CFB, OFB, 和 CTR 模式, CRYPTO_TDESn_CNT 在存储最后一个数据块之前设置最后一个数据块的字节数.

CRYPTO_TDES_DATIN TDES/DES 数据输入端口寄存器

寄存器	偏移量	R/W	描述	复位值
CRYPTO_TDES_DATIN	CRYPTO_BA+0x234	R/W	TDES/DES 引擎输入数据字寄存器	0x0000_0000

31	30	29	28	27	26	25	24
DATIN							
23	22	21	20	19	18	17	16
DATIN							
15	14	13	12	11	10	9	8
DATIN							
7	6	5	4	3	2	1	0
DATIN							

位	描述	
[31:0]	DATIN	**TDES/DES 引擎输入端口** CPU 通过这个端口向TDES/DES 引擎填写数据，确认CRYPTO_TDES_STS寄存器，在INBUFFULL 为 0时填写数据.

CRYPTO_TDES_DATOUT TDES/DES 数据输出端口寄存器

寄存器	偏移量	R/W	描述	复位值
CRYPTO_TDES_DATOUT	CRYPTO_BA+0x238	R	TDES/DES引擎输出数据字寄存器	0x0000_0000

31	30	29	28	27	26	25	24
DATOUT							
23	22	21	20	19	18	17	16
DATOUT							
15	14	13	12	11	10	9	8
DATOUT							
7	6	5	4	3	2	1	0
DATOUT							

位	描述	
[31:0]	DATOUT	**TDES/DES 引擎输出端口** CPU 通过这个端口从TDES/DES 引擎得到结果，确认CRYPTO_TDES_STS寄存器，在OUTBUFEMPTY 为 0时读取数据.

CRYPTO_TDES_FDBCKx TDES/DES 反馈 x 寄存器

寄存器	偏移量	R/W	描述	复位值
CRYPTO_TDES_FDBCKH	CRYPTO_BA+0x060	R	TDES/DES 引擎加密操作后输出反馈高位数据	0x0000_0000
CRYPTO_TDES_FDBCKL	CRYPTO_BA+0x064	R	TDES/DES 引擎加密操作后输出反馈低位数据	0x0000_0000

31	30	29	28	27	26	25	24
FDBCK							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
FDBCK							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
FDBCK							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
FDBCK							

位	描述	
[31:0]	**FDBCK**	**TDES/DES 反馈** 反馈值是 64 位. 在 DMA 级联模式下，TDES/DES 引擎用 {CRYPTO_TDES_FDBCKH, CRYPTO_TDES_FDBCKL} 的数据作为下一个数据块的 {CRYPTO_TDESn_IVH, CRYPTO_TDESn_IVL}。反馈寄存器在CBC, CFB, 和 OFB模式下使用. TDES/DES 引擎输出反馈信息作为下一个块操作的初始化向量。程序可以使用这个反馈信息实现多于4个DMA通道. 程序可以暂存反馈值，当切换回来，将反馈值填入同一个通道的CRYPTO_TDESn_IVH/L寄存器，可以依照原来的设定继续执行.

6.38.7.5 SHA/HMAC 寄存器

CRYPTO_HMAC_CTL SHA/HMAC 控制寄存器

寄存器	偏移量	R/W	描述	复位值
CRYPTO_HMAC_CTL	CRYPTO_BA+0x300	R/W	SHA/HMAC 控制寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
INSWAP	OUTSWAP	Reserved					
15	14	13	12	11	10	9	8
Reserved				HMACEN	OPMODE		
7	6	5	4	3	2	1	0
DMAEN	DMACSCAD	DMALAST	DMAFIRST	Reserved		STOP	START

位	描述	
[31:24]	**Reserved**	保留
[23]	**INSWAP**	**SHA/HMAC引擎输入数据交换** 0 = 保持原有顺序. 1 = CPU存入加速器的数据顺序从{byte3, byte2, byte1, byte0} 变为 {byte0, byte1, byte2, byte3}.
[22]	**OUTSWAP**	**SHA/HMAC 引擎输出数据交换** 0 = 保持原有顺序. 1 = CPU从加速器读取数据的顺序从{byte3, byte2, byte1, byte0} 变为 {byte0, byte1, byte2, byte3}.
[21:12]	**Reserved**	保留
[11]	**HMACEN**	**HMAC_SHA 引擎操作模式** 0 = 执行 SHA 功能 1 = 执行 HMAC 功能
[10:8]	**OPMODE**	**SHA/HMAC 引擎操作模式** 0x0xx: SHA160 0x100: SHA256 0x101: SHA224 0x110: SHA512 0x111: SHA384 这些位可读写. 但当BUSY为 1时写入没有效果.

[7]	**DMAEN**	**SHA/HMAC 引擎 DMA 使能控制** 0 = SHA/HMAC DMA 引擎禁止. SHA/HMAC 引擎工作在非-DMA 模式, 从端口 CRYPTO_HMAC_DATIN得到数据. 1 = SHA/HMAC DMA 引擎使能. SHA/HMAC 引擎工作在 DMA 模式, 数据存取由 DMA 逻辑完成.
[6]	**DMACSCAD**	**SHA/HMAC 引擎 DMA串流模式** 0 = DMA串流模式禁止 1 = 在DMA串流模式下，软件可以在串流操作时更新DAM源地址，目的地址及DMA计数寄存器，不需要等待操作完成
[5]	**DMALAST**	**SHA/HMAC 串流模式最后区块** 此位在最后一个数据块时置1.
[4]	**DMAFIRST**	**SHA/HMAC 串流模式第一个区块** 此位必须在输入第一个数据时置1
[3:2]	**Reserved**	保留
[1]	**STOP**	**SHA/HMAC 引擎停止** 0 = 没影响. 1 = 停止 SHA/HMAC 引擎. 此位总是读为0
[0]	**START**	**SHA/HMAC 引擎开始** 0 = 没影响. 1 = 开始 SHA/HMAC 引擎. BUSY标志置1. 此位总是读为0.

CRYPTO_HMAC_STS SHA/HMAC 状态寄存器

寄存器	偏移量	R/W	描述	复位值
CRYPTO_HMAC_STS	CRYPTO_BA+0x304	R	SHA/HMAC 状态标志	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							DATINREQ
15	14	13	12	11	10	9	8
Reserved							DMAERR
7	6	5	4	3	2	1	0
Reserved						DMABUSY	BUSY

位	描述	
[31:16]	**Reserved**	保留
[16]	**DATINREQ**	**SHA/HMAC 非-DMA模式数据输入请求** 0 = 没影响. 1 = 请求SHA/HMAC 非-DMA 模式数据输入.
[15:9]	**Reserved**	保留.
[8]	**DMAERR**	**SHA/HMAC引擎 DMA 错误标志** 0 = 表明SHA/HMAC 引擎正常访问. 1 = 表明 SHA/HMAC引擎访问错误
[7:2]	**Reserved**	保留.
[1]	**DMABUSY**	**SHA/HMAC 引擎 DMA 忙标志** 0 = SHA/HMAC DMA 引擎在空闲或完成态. 1 = SHA/HMAC DMA 引擎忙.
[0]	**BUSY**	**SHA/HMAC 引擎忙** 0 = SHA/HMAC 在空闲或完成态 1 = SHA/HMAC 引擎忙.

CRYPTO_HMAC_DGSTx SHA/HMAC输出摘要字寄存器

寄存器	偏移量	R/W	描述	复位值
CRYPTO_HMAC_DGST0	CRYPTO_BA+0x308	R	SHA/HMAC 摘要信息 0	0x0000_0000
CRYPTO_HMAC_DGST1	CRYPTO_BA+0x30C	R	SHA/HMAC摘要信息1	0x0000_0000
CRYPTO_HMAC_DGST2	CRYPTO_BA+0x310	R	SHA/HMAC摘要信息2	0x0000_0000
CRYPTO_HMAC_DGST3	CRYPTO_BA+0x314	R	SHA/HMAC摘要信息3	0x0000_0000
CRYPTO_HMAC_DGST4	CRYPTO_BA+0x318	R	SHA/HMAC摘要信息4	0x0000_0000
CRYPTO_HMAC_DGST5	CRYPTO_BA+0x31C	R	SHA/HMAC摘要信息5	0x0000_0000
CRYPTO_HMAC_DGST6	CRYPTO_BA+0x320	R	SHA/HMAC摘要信息6	0x0000_0000
CRYPTO_HMAC_DGST7	CRYPTO_BA+0x324	R	SHA/HMAC摘要信息7	0x0000_0000
CRYPTO_HMAC_DGST8	CRYPTO_BA+0x328	R	SHA/HMAC摘要信息8	0x0000_0000
CRYPTO_HMAC_DGST9	CRYPTO_BA+0x32C	R	SHA/HMAC摘要信息9	0x0000_0000
CRYPTO_HMAC_DGST10	CRYPTO_BA+0x330	R	SHA/HMAC摘要信息10	0x0000_0000
CRYPTO_HMAC_DGST11	CRYPTO_BA+0x334	R	SHA/HMAC摘要信息11	0x0000_0000
CRYPTO_HMAC_DGST12	CRYPTO_BA+0x338	R	SHA/HMAC摘要信息12	0x0000_0000
CRYPTO_HMAC_DGST13	CRYPTO_BA+0x33C	R	SHA/HMAC摘要信息13	0x0000_0000
CRYPTO_HMAC_DGST14	CRYPTO_BA+0x340	R	SHA/HMAC摘要信息14	0x0000_0000
CRYPTO_HMAC_DGST15	CRYPTO_BA+0x344	R	SHA/HMAC摘要信息15	0x0000_0000

31	30	29	28	27	26	25	24
DGST							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
DGST							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
DGST							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
DGST							

位	描述	
[31:0]	**DGST**	**SHA/HMAC 摘要信息输出寄存器** 对于 SHA-160, 信息摘要存于 CRYPTO_HMAC_DGST0 ~ CRYPTO_HMAC_DGST4. 对于 SHA-224, 信息摘要存于CRYPTO_HMAC_DGST0 ~ CRYPTO_HMAC_DGST6. 对于SHA-256, 信息摘要存于CRYPTO_HMAC_DGST0 ~ CRYPTO_HMAC_DGST7. 对于SHA-384, 信息摘要存于CRYPTO_HMAC_DGST0 ~ CRYPTO_HMAC_DGST11. 对于SHA-512, 信息摘要存于CRYPTO_HMAC_DGST0 ~ CRYPTO_HMAC_DGST15.

CRYPTO_HMAC_KEYCNT SHA/HMAC 密钥字节计数寄存器

寄存器	偏移量	R/W	描述	复位值
CRYPTO_HMAC_KEYCNT	CRYPTO_BA+0x348	R/W	SHA/HMAC密钥字节计数寄存器	0x0000_0000

31	30	29	28	27	26	25	24
KEYCNT							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
KEYCNT							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
KEYCNT							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
KEYCNT							

位	描述	
[31:0]	**KEYCNT**	**SHA/HMAC 密钥字节计数** CRYPTO_HMAC_KEYCNT 保存SHA/HMAC 引擎操作的密钥字节个数。这个寄存器是32-位，最大的字节数是4G字节. 这里可读可写. 在SHA/HMAC操作时写 CRYPTO_HMAC_KEYCNT 不会影响当前的 SHA/HMAC操作，但是CRYPTO_SHA _KEYCNT的值会在稍后更新，所以程序可以为下次SHA/HMAC操作准备密钥个数

CRYPTO_HMAC_SADDR SHA/HMAC DMA 源地址寄存器

寄存器	偏移量	R/W	描述	复位值
CRYPTO_HMAC_SADDR	CRYPTO_BA+0x34C	R/W	SHA/HMAC DMA 源地址寄存器	0x0000_0000

31	30	29	28	27	26	25	24
SADDR							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
SADDR							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
SADDR							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
SADDR							

位	描述	
[31:0]	**SADDR**	**SHA/HMAC DMA源地址** SHA/HMAC 加速器支持 DMA 功能在SRAM内存和内建FIFO之间传输原始文本CRYPTO_HMAC_SADDR 存放原始文本存放数据缓存的源地址。基于这个源地址，SHA/HMAC 加速器可以从SRAM中读到原始文本然后执行 SHA/HMAC 操作.。源地址的开始地址必须是字对齐的，就是说CRYPTO_HMAC_SADDR的位1和位0会被忽略 CRYPTO_HMAC_SADDR 可读可写。当SHA/HMAC加速器正在工作时写CRYPTO_HMAC_SADDR 不会影响当前的 SHA/HMAC 操作，但CRYPTO_HMAC_SADDR 的值会在稍后更新。程序可以为下次SHA/HMAC操作准备 DMA 源地址. 在 DMA 模式, 程序可以在触发START 更新下一笔CRYPTO_HMAC_SADDR. CRYPTO_HMAC_SADDR 和 CRYPTO_HMAC_DADDR 值可以相同.

CRYPTO_HMAC_DMACNT SHA/HMAC 字节计数寄存器

寄存器	偏移量	R/W	描述	复位值
CRYPTO_HMAC_DMACNT	CRYPTO_BA+0x350	R/W	SHA/HMAC 字节计数寄存器	0x0000_0000

31	30	29	28	27	26	25	24
DMACNT							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
DMACNT							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
DMACNT							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
DMACNT							

位	描述	
[31:0]	**DMACNT**	**SHA/HMAC 操作字节数** CRYPTO_HMAC_DMACNT存放SHA/HMAC引擎工作在DMA模式时原始文本的字节数 . CRYPTO_HMAC_DMACNT 是32位，最大字节数是4G 字节. CRYPTO_HMAC_DMACNT 可读可写，在SHA/HMAC加速器正在操作时写CRYPTO_HMAC_DMACNT不会影响目前的SHA/HMAC操作。但CRYPTO_HMAC_DMACNT的值会在稍后更新，程序可以为下次SHA/HMAC操作准备数据字节数. 在非-DMA模式, CRYPTO_HMAC_DMACNT 必须在填写最后一个数据块之前设置

CRYPTO_HMAC_DATIN SHA/HMAC 数据输入端口寄存器

寄存器	偏移量	R/W	描述	复位值
CRYPTO_HMAC_DATIN	CRYPTO_BA+0x354	R/W	SHA/HMAC 引擎非-DMA 模式数据输入端口寄存器	0x0000_0000

31	30	29	28	27	26	25	24
DATIN							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
DATIN							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
DATIN							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
DATIN							

位	描述	
[31:0]	**DATIN**	**SHA/HMAC引擎输入端口** CPU 通过这个端口填写数据到SHA/HMAC 引擎，确认CRYPTO_HMAC_STS的状态，在DATINREQ 为 1时填写数据.

6.38.7.6 ECC 寄存器

CRYPTO_ECC_CTL ECC 控制寄存器

寄存器	偏移量	R/W	描述	复位值
CRYPTO_ECC_CTL	CRYPTO_BA+0x800	R/W	ECC 控制寄存器	0x0000_0000

31	30	29	28	27	26	25	24
CURVEM							
23	22	21	20	19	18	17	16
CURVEM		LDK	LDN	LDB	LDA	LDP2	LDP1
15	14	13	12	11	10	9	8
Reserved	SCAP	CSEL	MODOP		ECCOP		FSEL
7	6	5	4	3	2	1	0
DMAEN	Reserved					STOP	START

Bits	Description	
[31:22]	**CURVEM**	椭圆曲线密钥长度
[21]	**LDK**	**SCALARK 寄存器控制信号** 0 = SCALARK 的寄存器不被DMA或用户修改 1 = SCALARK 的寄存器被DMA或用户修改.
[20]	**LDN**	**椭圆曲线参数CURVEN 寄存器控制信号** 0 = CURVEN 的寄存器不被DMA或用户修改. 1 = CURVEN 的寄存器被DMA或用户修改
[19]	**LDB**	**椭圆曲线参数CURVEB 寄存器控制信号** 0 = CURVEB 的寄存器不被DMA或用户修改. 1 = CURVEB 的寄存器被DMA或用户修改
[18]	**LDA**	**椭圆曲线参数CURVEA 寄存器控制信号** 0 = CURVEA 的寄存器不被DMA或用户修改. 1 = CURVEA 的寄存器被DMA或用户修改
[17]	**LDP2**	**第二点(POINTX2, POINTY2)的X，Y坐标的寄存器控制信号** 0 = POINTX2 和 POINTY2的寄存器不被DMA或用户修改. 1 = POINTX2 和 POINTY2的寄存器被DMA或用户修改.
[16]	**LDP1**	**第一点(POINTX2, POINTY2)的X，Y坐标的寄存器控制信号** 0 = POINTX1 和 POINTY1的寄存器不被DMA或用户修改. 1 = POINTX1 和 POINTY1的寄存器被DMA或用户修改.
[15]	**Reserved**	保留

[14]	**SCAP**	**侧路攻击保护** 0 = 全速没有侧路保护 1 = 慢速有侧路保护
[13]	**CSEL**	**曲线选择** 0 = NIST建议曲线 1 = 蒙哥马利曲线
[12:11]	**MODOP**	**PF的模操作** 00 = 除 :. POINTX1 = (POINTY1 / POINTX1) % CURVEN. 01 = 乘 :. POINTX1 = (POINTX1 * POINTY1) % CURVEN. 10 = 加 :. POINTX1 = (POINTX1 + POINTY1) % CURVEN. 11 = 减 :. POINTX1 = (POINTX1 - POINTY1) % CURVEN. MODOP只当 ECCOP = 01时有效.
[10:9]	**ECCOP**	**BF 和 PF的点操作** 00 = 点乘 :. (POINTX1, POINTY1) = SCALARK * (POINTX1, POINTY1). 01 = 模操作 : 由 MODOP (CRYPTO_ECC_CTL[12:11])选择. 10 = 点加 :. (POINTX1, POINTY1) = (POINTX1, POINTY1) +. (POINTX2, POINTY2) 11 = 点翻倍 :. (POINTX1, POINTY1) = 2 * (POINTX1, POINTY1). 除了上面3个输入数据，点操作还需要椭圆曲线的其他参数(CURVEA, CURVEB, CURVEN 和 CURVEM) 入图 6.27-11.
[8]	**FSEL**	**域选择** 0 = 二进制域 (GF(2^m)). 1 = 素数域 (GF(p)).
[7]	**DMAEN**	**ECC 加速器 DMA 使能控制** 0 = ECC DMA 引擎禁止. 1 = ECC DMA 引擎使能 只有当 START 和 DMAEN 为 1, ECC DMA 引擎才会使能
[6:2]	**Reserved**	保留.
[1]	**STOP**	**ECC 加速器停止** 0 = 没影响. 1 = 中断 ECC 加速器并让它进入空闲态 读此位总为0. 在停止ECC 加速器之后记得清 ECC 中断标志.

[0]	**START**	**ECC加速器开始** 0 = 没影响. 1 = 开始 ECC 加速器，. BUSY标志会置1 读此位总为0.. 当BUSY 标志是1.ECC 加速器会忽略这次START 信号 .

CRYPTO_ECC_STS ECC 状态寄存器

寄存器	偏移量	R/W	描述	复位值
CRYPTO_ECC_STS	CRYPTO_BA+0x804	R	ECC 状态寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							BUSERR
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved						DMABUSY	BUSY

位	描述	
[31:17]	**Reserved**	保留
[16]	**BUSERR**	**ECC DMA 访问总线错误标志** 0 = 没有错误. 1 = 总线错误会停止 DMA 操作和 ECC 加速器.
[15:2]	**Reserved**	保留.
[1]	**DMABUSY**	**ECC DMA 忙标志** 0 = ECC DMA在空闲或完成态 1 = ECC DMA忙
[0]	**BUSY**	**ECC 加速器忙标志** 0 = ECC 加速器在空闲或完成态. 1 = ECC 加速器正在执行且保护所有寄存器. ECC 加速器完成之后记得清 ECC 中断标志

CRYPTO_ECC_X1 ECC第一点 X坐标值寄存器

寄存器	偏移量	R/W	描述	复位值
CRYPTO_ECC_X1_00	CRYPTO_BA+0x808	R/W	ECC第一点 X坐标值字0	0x0000_0000
CRYPTO_ECC_X1_01	CRYPTO_BA+0x80C	R/W	ECC第一点 X坐标值字1	0x0000_0000
CRYPTO_ECC_X1_02	CRYPTO_BA+0x810	R/W	ECC第一点 X坐标值字2	0x0000_0000
CRYPTO_ECC_X1_03	CRYPTO_BA+0x814	R/W	ECC第一点 X坐标值字3	0x0000_0000
CRYPTO_ECC_X1_04	CRYPTO_BA+0x818	R/W	ECC第一点 X坐标值字4	0x0000_0000
CRYPTO_ECC_X1_05	CRYPTO_BA+0x81C	R/W	ECC第一点 X坐标值字5	0x0000_0000
CRYPTO_ECC_X1_06	CRYPTO_BA+0x820	R/W	ECC第一点 X坐标值字6	0x0000_0000
CRYPTO_ECC_X1_07	CRYPTO_BA+0x824	R/W	ECC第一点 X坐标值字7	0x0000_0000
CRYPTO_ECC_X1_08	CRYPTO_BA+0x828	R/W	ECC第一点 X坐标值字8	0x0000_0000
CRYPTO_ECC_X1_09	CRYPTO_BA+0x82C	R/W	ECC第一点 X坐标值字9	0x0000_0000
CRYPTO_ECC_X1_10	CRYPTO_BA+0x830	R/W	ECC第一点 X坐标值字10	0x0000_0000
CRYPTO_ECC_X1_11	CRYPTO_BA+0x834	R/W	ECC第一点 X坐标值字11	0x0000_0000
CRYPTO_ECC_X1_12	CRYPTO_BA+0x838	R/W	ECC第一点 X坐标值字12	0x0000_0000
CRYPTO_ECC_X1_13	CRYPTO_BA+0x83C	R/W	ECC第一点 X坐标值字13	0x0000_0000
CRYPTO_ECC_X1_14	CRYPTO_BA+0x840	R/W	ECC第一点 X坐标值字14	0x0000_0000
CRYPTO_ECC_X1_15	CRYPTO_BA+0x844	R/W	ECC第一点 X坐标值字15	0x0000_0000
CRYPTO_ECC_X1_16	CRYPTO_BA+0x848	R/W	ECC第一点 X坐标值字16	0x0000_0000
CRYPTO_ECC_X1_17	CRYPTO_BA+0x84C	R/W	ECC第一点 X坐标值字17	0x0000_0000

31	30	29	28	27	26	25	24
POINTX1							
23	22	21	20	19	18	17	16
POINTX1							
15	14	13	12	11	10	9	8

POINTX1							
7	6	5	4	3	2	1	0
POINTX1							

位	描述	
[31:0]	**POINTX1**	**ECC第一点 X坐标值 (POINTX1)** 对于 B-163或 K-163, POINTX1 存储于 CRYPTO_ECC_X1_00~CRYPTO_ECC_X1_05 对于 B-233或K-233, POINTX1 存储于 CRYPTO_ECC_X1_00~CRYPTO_ECC_X1_07 对于B-283或K-283, POINTX1存储于 CRYPTO_ECC_X1_00~CRYPTO_ECC_X1_08 对于B-409或K-409, POINTX1 存储于 CRYPTO_ECC_X1_00~CRYPTO_ECC_X1_12 对于B-571或K-571, POINTX1存储于 CRYPTO_ECC_X1_00~CRYPTO_ECC_X1_17 对于P-192, POINTX1 存储于 CRYPTO_ECC_X1_00~CRYPTO_ECC_X1_05 对于P-224, POINTX1 存储于 CRYPTO_ECC_X1_00~CRYPTO_ECC_X1_06 对于P-256, POINTX1 存储于 CRYPTO_ECC_X1_00~CRYPTO_ECC_X1_07 对于P-384, POINTX1 存储于 CRYPTO_ECC_X1_00~CRYPTO_ECC_X1_11 对于P-521, POINTX1 存储于 CRYPTO_ECC_X1_00~CRYPTO_ECC_X1_16

CRYPTO_ECC_Y1ECC第一点 Y坐标值寄存器

寄存器	偏移量	R/W	描述	复位值
CRYPTO_ECC_Y1_00	CRYPTO_BA+0x850	R/W	ECC第一点 Y坐标值字0	0x0000_0000
CRYPTO_ECC_Y1_01	CRYPTO_BA+0x854	R/W	ECC第一点 Y坐标值字1	0x0000_0000
CRYPTO_ECC_Y1_02	CRYPTO_BA+0x858	R/W	ECC第一点 Y坐标值字2	0x0000_0000
CRYPTO_ECC_Y1_03	CRYPTO_BA+0x85C	R/W	ECC第一点 Y坐标值字3	0x0000_0000
CRYPTO_ECC_Y1_04	CRYPTO_BA+0x860	R/W	ECC第一点 Y坐标值字4	0x0000_0000
CRYPTO_ECC_Y1_05	CRYPTO_BA+0x864	R/W	ECC第一点 Y坐标值字5	0x0000_0000
CRYPTO_ECC_Y1_06	CRYPTO_BA+0x868	R/W	ECC第一点 Y坐标值字6	0x0000_0000
CRYPTO_ECC_Y1_07	CRYPTO_BA+0x86C	R/W	ECC第一点 Y坐标值字7	0x0000_0000
CRYPTO_ECC_Y1_08	CRYPTO_BA+0x870	R/W	ECC第一点 Y坐标值字8	0x0000_0000
CRYPTO_ECC_Y1_09	CRYPTO_BA+0x874	R/W	ECC第一点 Y坐标值字9	0x0000_0000
CRYPTO_ECC_Y1_10	CRYPTO_BA+0x878	R/W	ECC第一点 Y坐标值字10	0x0000_0000
CRYPTO_ECC_Y1_11	CRYPTO_BA+0x87C	R/W	ECC第一点 Y坐标值字11	0x0000_0000
CRYPTO_ECC_Y1_12	CRYPTO_BA+0x880	R/W	ECC第一点 Y坐标值字12	0x0000_0000
CRYPTO_ECC_Y1_13	CRYPTO_BA+0x884	R/W	ECC第一点 Y坐标值字13	0x0000_0000
CRYPTO_ECC_Y1_14	CRYPTO_BA+0x888	R/W	ECC第一点 Y坐标值字14	0x0000_0000
CRYPTO_ECC_Y1_15	CRYPTO_BA+0x88C	R/W	ECC第一点 Y坐标值字15	0x0000_0000
CRYPTO_ECC_Y1_16	CRYPTO_BA+0x890	R/W	ECC第一点 Y坐标值字16	0x0000_0000
CRYPTO_ECC_Y1_17	CRYPTO_BA+0x894	R/W	ECC第一点 Y坐标值字17	0x0000_0000

31	30	29	28	27	26	25	24
POINTY1							
23	22	21	20	19	18	17	16
POINTY1							
15	14	13	12	11	10	9	8

POINTY1							
7	6	5	4	3	2	1	0
POINTY1							

位	描述	
[31:0]	**POINTY1**	**ECC第一点 Y坐标值寄存器 (POINTY1)** 对于 B-163或K-163, POINTY1存储于 CRYPTO_ECC_Y1_00~CRYPTO_ECC_Y1_05 对于B-233或K-233, POINTY1 存储于 CRYPTO_ECC_Y1_00~CRYPTO_ECC_Y1_07 对于B-283或K-283, POINTY1存储于 CRYPTO_ECC_Y1_00~CRYPTO_ECC_Y1_08 对于B-409或K-409, POINTY1 存储于 CRYPTO_ECC_Y1_00~CRYPTO_ECC_Y1_12 对于B-571或K-571, POINTY1 存储于 CRYPTO_ECC_Y1_00~CRYPTO_ECC_Y1_17 对于P-192, POINTY1 存储于 CRYPTO_ECC_Y1_00~CRYPTO_ECC_Y1_05 对于P-224, POINTY1 存储于 CRYPTO_ECC_Y1_00~CRYPTO_ECC_Y1_06 对于P-256, POINTY1 存储于 CRYPTO_ECC_Y1_00~CRYPTO_ECC_Y1_07 对于P-384, POINTY1 存储于 CRYPTO_ECC_Y1_00~CRYPTO_ECC_Y1_11 对于P-521, POINTY1 存储于 CRYPTO_ECC_Y1_00~CRYPTO_ECC_Y1_16

CRYPTO_ECC_X2ECC第二点 X坐标值寄存器

寄存器	偏移量	R/W	描述	复位值
CRYPTO_ECC_X2_00	CRYPTO_BA+0x898	R/W	ECC第二点 X坐标值字0	0x0000_0000
CRYPTO_ECC_X2_01	CRYPTO_BA+0x89C	R/W	ECC第二点 X坐标值字1	0x0000_0000
CRYPTO_ECC_X2_02	CRYPTO_BA+0x8A0	R/W	ECC第二点 X坐标值字2	0x0000_0000
CRYPTO_ECC_X2_03	CRYPTO_BA+0x8A4	R/W	ECC第二点 X坐标值字3	0x0000_0000
CRYPTO_ECC_X2_04	CRYPTO_BA+0x8A8	R/W	ECC第二点 X坐标值字4	0x0000_0000
CRYPTO_ECC_X2_05	CRYPTO_BA+0x8AC	R/W	ECC第二点 X坐标值字5	0x0000_0000
CRYPTO_ECC_X2_06	CRYPTO_BA+0x8B0	R/W	ECC第二点 X坐标值字6	0x0000_0000
CRYPTO_ECC_X2_07	CRYPTO_BA+0x8B4	R/W	ECC第二点 X坐标值字7	0x0000_0000
CRYPTO_ECC_X2_08	CRYPTO_BA+0x8B8	R/W	ECC第二点 X坐标值字8	0x0000_0000
CRYPTO_ECC_X2_09	CRYPTO_BA+0x8BC	R/W	ECC第二点 X坐标值字9	0x0000_0000
CRYPTO_ECC_X2_10	CRYPTO_BA+0x8C0	R/W	ECC第二点 X坐标值字10	0x0000_0000
CRYPTO_ECC_X2_11	CRYPTO_BA+0x8C4	R/W	ECC第二点 X坐标值字11	0x0000_0000
CRYPTO_ECC_X2_12	CRYPTO_BA+0x8C8	R/W	ECC第二点 X坐标值字12	0x0000_0000
CRYPTO_ECC_X2_13	CRYPTO_BA+0x8CC	R/W	ECC第二点 X坐标值字13	0x0000_0000
CRYPTO_ECC_X2_14	CRYPTO_BA+0x8D0	R/W	ECC第二点 X坐标值字14	0x0000_0000
CRYPTO_ECC_X2_15	CRYPTO_BA+0x8D4	R/W	ECC第二点 X坐标值字15	0x0000_0000
CRYPTO_ECC_X2_16	CRYPTO_BA+0x8D8	R/W	ECC第二点 X坐标值字16	0x0000_0000
CRYPTO_ECC_X2_17	CRYPTO_BA+0x8DC	R/W	ECC第二点 X坐标值字17	0x0000_0000

31	30	29	28	27	26	25	24
POINTX2							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
POINTX2							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**

POINTX2							
7	6	5	4	3	2	1	0
POINTX2							

位	描述	
[31:0]	**POINTX2**	**ECC第二点 X坐标值寄存器 (POINTX2)** 对于 B-163或K-163, POINTX2 存储于 CRYPTO_ECC_X2_00~CRYPTO_ECC_X2_05 对于B-233或K-233, POINTX2 存储于 CRYPTO_ECC_X2_00~CRYPTO_ECC_X2_07 对于B-283或K-283, POINTX2 存储于 CRYPTO_ECC_X2_00~CRYPTO_ECC_X2_08 对于B-409或K-409, POINTX2 存储于 CRYPTO_ECC_X2_00~CRYPTO_ECC_X2_12 对于B-571或K-571, POINTX2 存储于 CRYPTO_ECC_X2_00~CRYPTO_ECC_X2_17 对于P-192, POINTX2 存储于 CRYPTO_ECC_X2_00~CRYPTO_ECC_X2_05 对于P-224, POINTX2 存储于 CRYPTO_ECC_X2_00~CRYPTO_ECC_X2_06 对于P-256, POINTX2 存储于 CRYPTO_ECC_X2_00~CRYPTO_ECC_X2_07 对于P-384, POINTX2 存储于 CRYPTO_ECC_X2_00~CRYPTO_ECC_X2_11 对于P-521, POINTX2 存储于 CRYPTO_ECC_X2_00~CRYPTO_ECC_X2_16

CRYPTO_ECC_Y2ECC第二点Y坐标值寄存器

寄存器	偏移量	R/W	描述	复位值
CRYPTO_ECC_Y2_00	CRYPTO_BA+0x8E0	R/W	ECC第二点 Y坐标值字0	0x0000_0000
CRYPTO_ECC_Y2_01	CRYPTO_BA+0x8E4	R/W	ECC第二点 Y坐标值字1	0x0000_0000
CRYPTO_ECC_Y2_02	CRYPTO_BA+0x8E8	R/W	ECC第二点 Y坐标值字2	0x0000_0000
CRYPTO_ECC_Y2_03	CRYPTO_BA+0x8EC	R/W	ECC第二点 Y坐标值字3	0x0000_0000
CRYPTO_ECC_Y2_04	CRYPTO_BA+0x8F0	R/W	ECC第二点 Y坐标值字4	0x0000_0000
CRYPTO_ECC_Y2_05	CRYPTO_BA+0x8F4	R/W	ECC第二点 Y坐标值字5	0x0000_0000
CRYPTO_ECC_Y2_06	CRYPTO_BA+0x8F8	R/W	ECC第二点 Y坐标值字6	0x0000_0000
CRYPTO_ECC_Y2_07	CRYPTO_BA+0x8FC	R/W	ECC第二点 Y坐标值字7	0x0000_0000
CRYPTO_ECC_Y2_08	CRYPTO_BA+0x900	R/W	ECC第二点 Y坐标值字8	0x0000_0000
CRYPTO_ECC_Y2_09	CRYPTO_BA+0x904	R/W	ECC第二点 Y坐标值字9	0x0000_0000
CRYPTO_ECC_Y2_10	CRYPTO_BA+0x908	R/W	ECC第二点 Y坐标值字10	0x0000_0000
CRYPTO_ECC_Y2_11	CRYPTO_BA+0x90C	R/W	ECC第二点 Y坐标值字11	0x0000_0000
CRYPTO_ECC_Y2_12	CRYPTO_BA+0x910	R/W	ECC第二点 Y坐标值字12	0x0000_0000
CRYPTO_ECC_Y2_13	CRYPTO_BA+0x914	R/W	ECC第二点 Y坐标值字13	0x0000_0000
CRYPTO_ECC_Y2_14	CRYPTO_BA+0x918	R/W	ECC第二点 Y坐标值字14	0x0000_0000
CRYPTO_ECC_Y2_15	CRYPTO_BA+0x91C	R/W	ECC第二点 Y坐标值字15	0x0000_0000
CRYPTO_ECC_Y2_16	CRYPTO_BA+0x920	R/W	ECC第二点 Y坐标值字16	0x0000_0000
CRYPTO_ECC_Y2_17	CRYPTO_BA+0x924	R/W	ECC第二点 Y坐标值字17	0x0000_0000

31	30	29	28	27	26	25	24
POINTY2							
23	22	21	20	19	18	17	16
POINTY2							
15	14	13	12	11	10	9	8

POINTY2							
7	6	5	4	3	2	1	0
POINTY2							

位	描述	
[31:0]	**POINTY2**	**ECC第二点 Y坐标值寄存器(POINTY2)** 对于 B-163或K-163, POINTY2存储于 CRYPTO_ECC_Y2_00~CRYPTO_ECC_Y2_05 对于B-233或K-233, POINTY2存储于 CRYPTO_ECC_Y2_00~CRYPTO_ECC_Y2_07 对于B-283或K-283, POINTY2存储于 CRYPTO_ECC_Y2_00~CRYPTO_ECC_Y2_08 对于B-409或K-409, POINTY2存储于 CRYPTO_ECC_Y2_00~CRYPTO_ECC_Y2_12 对于B-571或K-571, POINTY2存储于 CRYPTO_ECC_Y2_00~CRYPTO_ECC_Y2_17 对于P-192, POINTY2存储于 CRYPTO_ECC_Y2_00~CRYPTO_ECC_Y2_05 对于P-224, POINTY2存储于 CRYPTO_ECC_Y2_00~CRYPTO_ECC_Y2_06 对于P-256, POINTY2存储于 CRYPTO_ECC_Y2_00~CRYPTO_ECC_Y2_07 对于P-384, POINTY2存储于 CRYPTO_ECC_Y2_00~CRYPTO_ECC_Y2_11 对于P-521, POINTY2存储于 CRYPTO_ECC_Y2_00~CRYPTO_ECC_Y2_16

CRYPTO_ECC_A ECC 椭圆曲线参数 CURVEA 值寄存器

寄存器	偏移量	R/W	描述	复位值
CRYPTO_ECC_A_00	CRYPTO_BA+0x928	R/W	ECC 椭圆曲线参数 CURVEA 字0	0x0000_0000
CRYPTO_ECC_A_01	CRYPTO_BA+0x92C	R/W	ECC 椭圆曲线参数 CURVEA 字1	0x0000_0000
CRYPTO_ECC_A_02	CRYPTO_BA+0x930	R/W	ECC 椭圆曲线参数 CURVEA 字2	0x0000_0000
CRYPTO_ECC_A_03	CRYPTO_BA+0x934	R/W	ECC 椭圆曲线参数 CURVEA 字3	0x0000_0000
CRYPTO_ECC_A_04	CRYPTO_BA+0x938	R/W	ECC 椭圆曲线参数 CURVEA 字4	0x0000_0000
CRYPTO_ECC_A_05	CRYPTO_BA+0x93C	R/W	ECC 椭圆曲线参数 CURVEA 字5	0x0000_0000
CRYPTO_ECC_A_06	CRYPTO_BA+0x940	R/W	ECC 椭圆曲线参数 CURVEA 字6	0x0000_0000
CRYPTO_ECC_A_07	CRYPTO_BA+0x944	R/W	ECC 椭圆曲线参数 CURVEA 字7	0x0000_0000
CRYPTO_ECC_A_08	CRYPTO_BA+0x948	R/W	ECC 椭圆曲线参数 CURVEA 字8	0x0000_0000
CRYPTO_ECC_A_09	CRYPTO_BA+0x94C	R/W	ECC 椭圆曲线参数 CURVEA 字9	0x0000_0000
CRYPTO_ECC_A_10	CRYPTO_BA+0x950	R/W	ECC 椭圆曲线参数 CURVEA 字10	0x0000_0000
CRYPTO_ECC_A_11	CRYPTO_BA+0x954	R/W	ECC 椭圆曲线参数 CURVEA 字11	0x0000_0000
CRYPTO_ECC_A_12	CRYPTO_BA+0x958	R/W	ECC 椭圆曲线参数 CURVEA 字12	0x0000_0000
CRYPTO_ECC_A_13	CRYPTO_BA+0x95C	R/W	ECC 椭圆曲线参数 CURVEA 字13	0x0000_0000
CRYPTO_ECC_A_14	CRYPTO_BA+0x960	R/W	ECC 椭圆曲线参数 CURVEA 字14	0x0000_0000
CRYPTO_ECC_A_15	CRYPTO_BA+0x964	R/W	ECC 椭圆曲线参数 CURVEA 字15	0x0000_0000
CRYPTO_ECC_A_16	CRYPTO_BA+0x968	R/W	ECC 椭圆曲线参数 CURVEA 字16	0x0000_0000
CRYPTO_ECC_A_17	CRYPTO_BA+0x96C	R/W	ECC 椭圆曲线参数 CURVEA 字17	0x0000_0000

31	30	29	28	27	26	25	24
CURVEA							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
CURVEA							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**

CURVEA							
7	6	5	4	3	2	1	0
CURVEA							

位	描述	
[31:0]	**CURVEA**	**ECC 椭圆曲线参数 CURVEA 值 (CURVEA)** 椭圆曲线公式是 $y^2=x^3+CURVEA*x+CURVEB$ 在 GF(p) 以及 $y^2+x*y=x^3+CURVEA*x^2+CURVEB$ i在 GF(2^m). 对于 B-163或K-163, CURVEA存储于 CRYPTO_ECC_A_00~CRYPTO_ECC_A_05 对于 B-233或K-233, CURVEA存储于 CRYPTO_ECC_A_00~CRYPTO_ECC_A_07 对于 B-283或K-283, CURVEA存储于 CRYPTO_ECC_A_00~CRYPTO_ECC_A_08 对于 B-409或K-409, CURVEA存储于 CRYPTO_ECC_A_00~CRYPTO_ECC_A_12 对于 B-571或K-571, CURVEA存储于 CRYPTO_ECC_A_00~CRYPTO_ECC_A_17 对于 P-192, CURVEA存储于CRYPTO_ECC_A_00~CRYPTO_ECC_A_05 对于 P-224, CURVEA存储于CRYPTO_ECC_A_00~CRYPTO_ECC_A_06 对于 P-256, CURVEA存储于CRYPTO_ECC_A_00~CRYPTO_ECC_A_07 对于 P-384, CURVEA存储于CRYPTO_ECC_A_00~CRYPTO_ECC_A_11 对于 P-521, CURVEA存储于CRYPTO_ECC_A_00~CRYPTO_ECC_A_16

CRYPTO_ECC_BECC 椭圆曲线参数 CURVEB 值寄存器

寄存器	偏移量	R/W	描述	复位值
CRYPTO_ECC_B_00	CRYPTO_BA+0x970	R/W	ECC 椭圆曲线参数 CURVEB 字0	0x0000_0000
CRYPTO_ECC_B_01	CRYPTO_BA+0x974	R/W	ECC 椭圆曲线参数 CURVEB 字1	0x0000_0000
CRYPTO_ECC_B_02	CRYPTO_BA+0x978	R/W	ECC 椭圆曲线参数 CURVEB 字2	0x0000_0000
CRYPTO_ECC_B_03	CRYPTO_BA+0x97C	R/W	ECC 椭圆曲线参数 CURVEB 字3	0x0000_0000
CRYPTO_ECC_B_04	CRYPTO_BA+0x980	R/W	ECC 椭圆曲线参数 CURVEB 字4	0x0000_0000
CRYPTO_ECC_B_05	CRYPTO_BA+0x984	R/W	ECC 椭圆曲线参数 CURVEB 字5	0x0000_0000
CRYPTO_ECC_B_06	CRYPTO_BA+0x988	R/W	ECC 椭圆曲线参数 CURVEB 字6	0x0000_0000
CRYPTO_ECC_B_07	CRYPTO_BA+0x98C	R/W	ECC 椭圆曲线参数 CURVEB 字7	0x0000_0000
CRYPTO_ECC_B_08	CRYPTO_BA+0x990	R/W	ECC 椭圆曲线参数 CURVEB 字8	0x0000_0000
CRYPTO_ECC_B_09	CRYPTO_BA+0x994	R/W	ECC 椭圆曲线参数 CURVEB 字9	0x0000_0000
CRYPTO_ECC_B_10	CRYPTO_BA+0x998	R/W	ECC 椭圆曲线参数 CURVEB 字10	0x0000_0000
CRYPTO_ECC_B_11	CRYPTO_BA+0x99C	R/W	ECC 椭圆曲线参数 CURVEB 字11	0x0000_0000
CRYPTO_ECC_B_12	CRYPTO_BA+0x9A0	R/W	ECC 椭圆曲线参数 CURVEB 字12	0x0000_0000
CRYPTO_ECC_B_13	CRYPTO_BA+0x9A4	R/W	ECC 椭圆曲线参数 CURVEB 字13	0x0000_0000
CRYPTO_ECC_B_14	CRYPTO_BA+0x9A8	R/W	ECC 椭圆曲线参数 CURVEB 字14	0x0000_0000
CRYPTO_ECC_B_15	CRYPTO_BA+0x9AC	R/W	ECC 椭圆曲线参数 CURVEB 字15	0x0000_0000
CRYPTO_ECC_B_16	CRYPTO_BA+0x9B0	R/W	ECC 椭圆曲线参数 CURVEB 字16	0x0000_0000
CRYPTO_ECC_B_17	CRYPTO_BA+0x9B4	R/W	ECC 椭圆曲线参数 CURVEB 字17	0x0000_0000

31	30	29	28	27	26	25	24
CURVEB							
23	22	21	20	19	18	17	16
CURVEB							
15	14	13	12	11	10	9	8

CURVEB							
7	6	5	4	3	2	1	0
CURVEB							

位	描述	
[31:0]	**CURVEB**	**ECC 椭圆曲线参数 CURVEB 值(CURVEB)** 椭圆曲线公式是 $y^2=x^3+CURVEA*x+CURVEB$ 在 GF(p) 和 $y^2+x*y=x^3+CURVEA*x^2+CURVEB$ 在 GF(2^m). 对于 B-163或K-163, CURVEB存储于 CRYPTO_ECC_B_00~CRYPTO_ECC_B_05 对于 B-233或K-233, CURVEB存储于 CRYPTO_ECC_B_00~CRYPTO_ECC_B_07 对于 B-283或K-283, CURVEB存储于 CRYPTO_ECC_B_00~CRYPTO_ECC_B_08 对于 B-409或K-409, CURVEB存储于 CRYPTO_ECC_B_00~CRYPTO_ECC_B_12 对于 B-521或K-521, CURVEB存储于 CRYPTO_ECC_B_00~CRYPTO_ECC_B_17 对于 P-192, CURVEB存储于CRYPTO_ECC_B_00~CRYPTO_ECC_B_05 对于 P-224, CURVEB存储于CRYPTO_ECC_B_00~CRYPTO_ECC_B_06 对于 P-256, CURVEB存储于CRYPTO_ECC_B_00~CRYPTO_ECC_B_07 对于 P-384, CURVEB存储于CRYPTO_ECC_B_00~CRYPTO_ECC_B_11 对于 P-521, CURVEB存储于CRYPTO_ECC_B_00~CRYPTO_ECC_B_16

CRYPTO_ECC_NECC 椭圆曲线参数 CURVEN值寄存器

寄存器	偏移量	R/W	描述	复位值
CRYPTO_ECC_N_00	CRYPTO_BA+0x9B8	R/W	ECC 椭圆曲线参数 CURVEN 字0	0x0000_0000
CRYPTO_ECC_N_01	CRYPTO_BA+0x9BC	R/W	ECC 椭圆曲线参数 CURVEN 字1	0x0000_0000
CRYPTO_ECC_N_02	CRYPTO_BA+0x9C0	R/W	ECC 椭圆曲线参数 CURVEN 字2	0x0000_0000
CRYPTO_ECC_N_03	CRYPTO_BA+0x9C4	R/W	ECC 椭圆曲线参数 CURVEN 字3	0x0000_0000
CRYPTO_ECC_N_04	CRYPTO_BA+0x9C8	R/W	ECC 椭圆曲线参数 CURVEN 字4	0x0000_0000
CRYPTO_ECC_N_05	CRYPTO_BA+0x9CC	R/W	ECC 椭圆曲线参数 CURVEN 字5	0x0000_0000
CRYPTO_ECC_N_06	CRYPTO_BA+0x9D0	R/W	ECC 椭圆曲线参数 CURVEN 字6	0x0000_0000
CRYPTO_ECC_N_07	CRYPTO_BA+0x9D4	R/W	ECC 椭圆曲线参数 CURVEN 字7	0x0000_0000
CRYPTO_ECC_N_08	CRYPTO_BA+0x9D8	R/W	ECC 椭圆曲线参数 CURVEN 字8	0x0000_0000
CRYPTO_ECC_N_09	CRYPTO_BA+0x9DC	R/W	ECC 椭圆曲线参数 CURVEN 字9	0x0000_0000
CRYPTO_ECC_N_10	CRYPTO_BA+0x9E0	R/W	ECC 椭圆曲线参数 CURVEN 字10	0x0000_0000
CRYPTO_ECC_N_11	CRYPTO_BA+0x9E4	R/W	ECC 椭圆曲线参数 CURVEN 字11	0x0000_0000
CRYPTO_ECC_N_12	CRYPTO_BA+0x9E8	R/W	ECC 椭圆曲线参数 CURVEN 字12	0x0000_0000
CRYPTO_ECC_N_13	CRYPTO_BA+0x9EC	R/W	ECC 椭圆曲线参数 CURVEN 字13	0x0000_0000
CRYPTO_ECC_N_14	CRYPTO_BA+0x9F0	R/W	ECC 椭圆曲线参数 CURVEN 字14	0x0000_0000
CRYPTO_ECC_N_15	CRYPTO_BA+0x9F4	R/W	ECC 椭圆曲线参数 CURVEN 字15	0x0000_0000
CRYPTO_ECC_N_16	CRYPTO_BA+0x9F8	R/W	ECC 椭圆曲线参数 CURVEN 字16	0x0000_0000
CRYPTO_ECC_N_17	CRYPTO_BA+0x9FC	R/W	ECC 椭圆曲线参数 CURVEN 字17	0x0000_0000

31	30	29	28	27	26	25	24
CURVEN							
23	22	21	20	19	18	17	16
CURVEN							
15	14	13	12	11	10	9	8

CURVEN							
7	6	5	4	3	2	1	0
CURVEN							

位	描述	
[31:0]	**CURVEN**	**ECC 椭圆曲线参数 CURVEN 值(CURVEN)** 在GF(p), CURVEN 是素数多项式. 在GF(2^m), CURVEN 是不可约多项式. 对于 B-163或K-163, CURVEN存储于 CRYPTO_ECC_N_00~CRYPTO_ECC_N_05 对于 B-233或K-233, CURVEN存储于 CRYPTO_ECC_N_00~CRYPTO_ECC_N_07 对于 B-283或K-283, CURVEN存储于 CRYPTO_ECC_N_00~CRYPTO_ECC_N_08 对于 B-409或K-409, CURVEN存储于 CRYPTO_ECC_N_00~CRYPTO_ECC_N_12 对于 B-571或K-571, CURVEN存储于 CRYPTO_ECC_N_00~CRYPTO_ECC_N_17 对于 P-192, CURVEN存储于CRYPTO_ECC_N_00~CRYPTO_ECC_N_05 对于 P-224, CURVEN存储于CRYPTO_ECC_N_00~CRYPTO_ECC_N_06 对于 P-256, CURVEN存储于CRYPTO_ECC_N_00~CRYPTO_ECC_N_07 对于 P-384, CURVEN存储于CRYPTO_ECC_N_00~CRYPTO_ECC_N_11 对于 P-521, CURVEN存储于CRYPTO_ECC_N_00~CRYPTO_ECC_N_16

CRYPTO_ECC_KECC 椭圆曲线标量 K 值寄存器

寄存器	偏移量	R/W	描述	复位值
CRYPTO_ECC_K_00	CRYPTO_BA+0xA00	W	ECC 点乘标量 SCALARK 字0	0x0000_0000
CRYPTO_ECC_K_01	CRYPTO_BA+0xA04	W	ECC 点乘标量 SCALARK 字1	0x0000_0000
CRYPTO_ECC_K_02	CRYPTO_BA+0xA08	W	ECC 点乘标量 SCALARK 字2	0x0000_0000
CRYPTO_ECC_K_03	CRYPTO_BA+0xA0C	W	ECC 点乘标量 SCALARK 字3	0x0000_0000
CRYPTO_ECC_K_04	CRYPTO_BA+0xA10	W	ECC 点乘标量 SCALARK 字4	0x0000_0000
CRYPTO_ECC_K_05	CRYPTO_BA+0xA14	W	ECC 点乘标量 SCALARK 字5	0x0000_0000
CRYPTO_ECC_K_06	CRYPTO_BA+0xA18	W	ECC 点乘标量 SCALARK 字6	0x0000_0000
CRYPTO_ECC_K_07	CRYPTO_BA+0xA1C	W	ECC 点乘标量 SCALARK 字7	0x0000_0000
CRYPTO_ECC_K_08	CRYPTO_BA+0xA20	W	ECC 点乘标量 SCALARK 字8	0x0000_0000
CRYPTO_ECC_K_09	CRYPTO_BA+0xA24	W	ECC 点乘标量 SCALARK 字9	0x0000_0000
CRYPTO_ECC_K_10	CRYPTO_BA+0xA28	W	ECC 点乘标量 SCALARK 字10	0x0000_0000
CRYPTO_ECC_K_11	CRYPTO_BA+0xA2C	W	ECC 点乘标量 SCALARK 字11	0x0000_0000
CRYPTO_ECC_K_12	CRYPTO_BA+0xA30	W	ECC 点乘标量 SCALARK 字12	0x0000_0000
CRYPTO_ECC_K_13	CRYPTO_BA+0xA34	W	ECC 点乘标量 SCALARK 字13	0x0000_0000
CRYPTO_ECC_K_14	CRYPTO_BA+0xA38	W	ECC 点乘标量 SCALARK 字14	0x0000_0000
CRYPTO_ECC_K_15	CRYPTO_BA+0xA3C	W	ECC 点乘标量 SCALARK 字15	0x0000_0000
CRYPTO_ECC_K_16	CRYPTO_BA+0xA40	W	ECC 点乘标量 SCALARK 字16	0x0000_0000
CRYPTO_ECC_K_17	CRYPTO_BA+0xA44	W	ECC 点乘标量 SCALARK 字17	0x0000_0000

31	30	29	28	27	26	25	24
SCALARK							
23	22	21	20	19	18	17	16
SCALARK							
15	14	13	12	11	10	9	8

SCALARK							
7	6	5	4	3	2	1	0
SCALARK							

位	描述	
[31:0]	**SCALARK**	**ECC 椭圆曲线标量SCALARK 值(SCALARK)** 因为 SCALARK 一般存储私有密钥, ECC 加速器不允许读SCALARK寄存器. 对于 B-163或K-163, SCALARK存储于 CRYPTO_ECC_K_00~CRYPTO_ECC_K_05 对于 B-233或K-233, SCALARK存储于 CRYPTO_ECC_K_00~CRYPTO_ECC_K_07 对于 B-283或K-283, SCALARK存储于 CRYPTO_ECC_K_00~CRYPTO_ECC_K_08 对于 B-409或K-409, SCALARK存储于 CRYPTO_ECC_K_00~CRYPTO_ECC_K_12 对于 B-571或K-571, SCALARK存储于 CRYPTO_ECC_K_00~CRYPTO_ECC_K_17 对于 P-192, SCALARK存储于CRYPTO_ECC_K_00~CRYPTO_ECC_K_05 对于 P-224, SCALARK存储于CRYPTO_ECC_K_00~CRYPTO_ECC_K_06 对于 P-256, SCALARK存储于CRYPTO_ECC_K_00~CRYPTO_ECC_K_07 对于 P-384, SCALARK存储于CRYPTO_ECC_K_00~CRYPTO_ECC_K_11 对于 P-521, SCALARK存储于CRYPTO_ECC_K_00~CRYPTO_ECC_K_16

CRYPTO_ECC_SADDRECC DMA 源地址寄存器

寄存器	偏移量	R/W	描述	复位值
CRYPTO_ECC_SADDR	CRYPTO_BA+0xA48	R/W	ECC DMA 源地址寄存器	0x0000_0000

31	30	29	28	27	26	25	24
SADDR							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
SADDR							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
SADDR							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
SADDR							

位	描述	
[31:0]	**Reserved**	**ECC DMA 源地址** ECC 加速器支持DMA功能在SRAM内存和ECC加速器之间传输数据和参数. SADDR 存放原始文本存放的数据缓存的源地址，基于源地址, ECC 加速器可以从SRAM内存读到数据和参数执行ECC 操作. 起始地址的开始必须是字对齐的， SADDR 的位1和位0会被忽略。SADDR 可读可写. 在DMA 模式，程序需要在触发START前更新CRYPTO_ECC_SADDR.

CRYPTO_ECC_DADDR ECC DMA 目的地址寄存器

寄存器	偏移量	R/W	描述	复位值
CRYPTO_ECC_DADDR	CRYPTO_BA+0xA4C	R/W	ECC DMA 目的地址寄存器	0x0000_0000

31	30	29	28	27	26	25	24
DADDR							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
DADDR							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
DADDR							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
DADDR							

位	描述	
[31:0]	**DADDR**	**ECC DMA 目的地址** ECC 加速器支持 DMA 功能在SRAM内存和ECC 加速器之间传输数据和参数。 DADDR 存放ECC 引擎输出数据存放的数据缓存的地址，基于目的地址， ECC 加速器可以在ECC操作完成之后把结果数据写回SRAM内存。目的地址的其实位置必须是字对齐的，DADDR的位1和位0会被忽略。 DADDR 可读可写. 在 DMA 模式, 程序需要在触发START 前更新 CRYPTO_ECC_DADDR

CRYPTO_ECC_STARTREGECC 更新寄存器起始地址

寄存器	偏移量	R/W	描述	复位值
CRYPTO_ECC_STARTREG	CRYPTO_BA+0xA50	R/W	ECC 更新寄存器起始地址	0x0000_0000

31	30	29	28	27	26	25	24
STARTREG							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
STARTREG							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
STARTREG							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
STARTREG							

位	描述	
[31:0]	**STARTREG**	**ECC 更新寄存器起始地址** DMA从更新寄存器的地址向ECC 引擎填写第一笔数据或参数. 当 ECC 引擎正在工作, ECC 加速器不允许用户修改STARTREG.例如, 我们想要更新CRYPTO_ECC POINTX1的输入数据， STARTREG 的值是 0x808.

CRYPTO_ECC_WORDCNTECC DMA 字计数

寄存器	偏移量	R/W	描述	复位值
CRYPTO_ECC_WORDCNT	CRYPTO_BA+0xA54	R/W	ECC DMA 字计数	0x0000_0000

31	30	29	28	27	26	25	24
WORDCNT							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
WORDCNT							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
WORDCNT							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
WORDCNT							

位	描述	
[31:0]	**WORDCNT**	**ECC DMA 字计数** CRYPTO_ECC_WORDCNT 存放ECC 加速器在各种DMA 模式下的操作需要的输入数据的源地址。尽管 CRYPTO_ECC_WORDCNT 是 32-位, ECC 加速器的最大字数是144个字. CRYPTO_ECC_WORDCNT 可读可写.

6.39 摄像头接口控制器(CCAP)

6.39.1 概述

摄像头接口(CCAP)用来从图像传感器获取图片数据。获取图像数据后它会处理这些数据，然后DMA控制器将会通过AHB总线把这些数据从内部FIFO搬运到系统内存

6.39.2 特性

- 支持CCIR601 & CCIR656 & 4位接口接到CMOS图像传感器
- 支持CMOS传感器输入YUV422 及 RGB565颜色格式的数据
- 支持YUV422, RGB565, RGB555 和只有Y图像数据包输出
- 多个可屏蔽中断源：地址匹配，总线匹配，传输错误，视频帧结束，所有中断源共享一个中断入口
- 内置DMA控制器，支持通过AHB总线将数据从内部FIFO传送到系统内存
- 支持裁剪功能，将输入图片裁剪到需要的尺寸，方便处理
- 支持降低帧率
- 支持图像缩放
- 支持8位门槛设置的亮度输出

6.39.3 框图

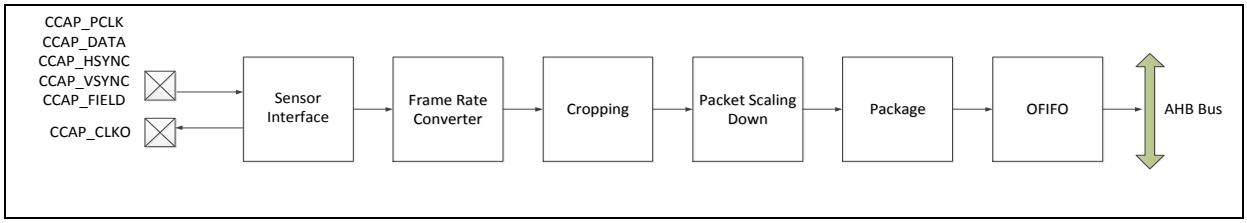


图 6.39-1 摄像头接口功能框图

6.39.4 基本配置

6.39.4.1 CCAP 管脚描述

名称	位	I/O 类型	管脚描述
CCAP_DATA	8	输入	传感器像素数据
CCAP_HSYNC	1	输入	传感器水平同步
CCAP_VSYNC	1	输入	传感器垂直同步
CCAP_PCLK	1	输入	传感器像素时钟（从传感器输出）
CCAP_FIELD	1	输入	奇偶标志
CCAP_CLKO	1	输出	传感器像素时钟（输出到传感器）

- CCAP_DATA跟CCAP_PCLK同步，你可以设置寄存器PCLKP (CCAP_PAR[8])改变上升/下降沿的极性

- CCAP_HSYNC信号表示一条线的开始/结束，可以通过HSP (CCAP_PAR[9])改变极性
- CCAP_VSYNC信号表示一帧数据的开始/结束，可以通过VSP (CCAP_PAR[10])改变极性
- CCAP_FIELD信号表示奇数还是偶数区域
- CCAP_CLKO信号表示输入传感器的像素时钟

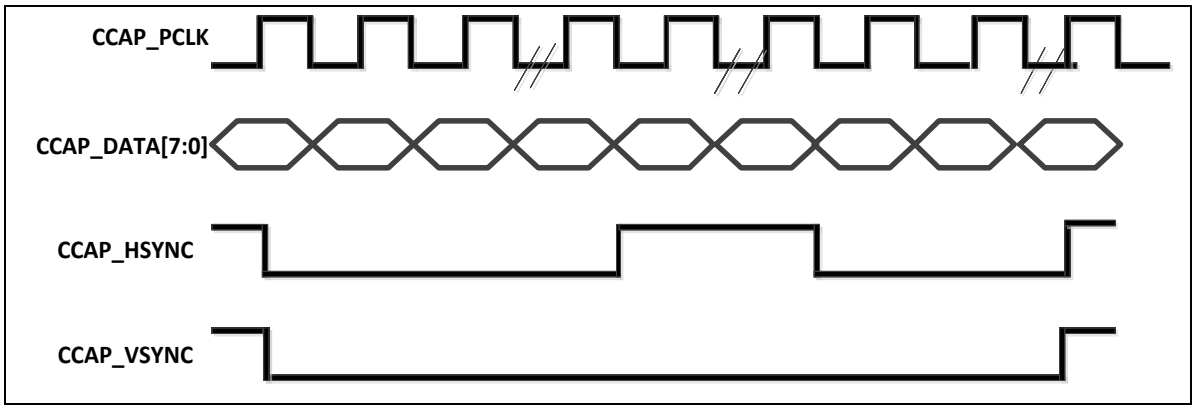


图 6.39-2 CCAP 基本管脚图

6.39.4.2 CCAP 管脚配置

组	管脚名称	GPIO	MFP
CCAP	CCAP_CLKO (输出到 Sensor)	GPG[10]	MFP7
		GPB[12]	MFP10
	CCAP_DATA0	GPF[7]	MFP7
		GPC[0]	MFP13
	CCAP_DATA1	GPF[8]	MFP7
		GPC[1]	MFP13
	CCAP_DATA2	GPF[9]	MFP7
		GPC[2]	MFP13
	CCAP_DATA3	GPF[10]	MFP7
		GPC[3]	MFP13
	CCAP_DATA4	GPF[11]	MFP7
		GPC[4]	MFP13
	CCAP_DATA5	GPG[4]	MFP7
		GPC[5]	MFP13
	CCAP_DATA6	GPG[3]	MFP7
		GPA[0]	MFP10
	CCAP_DATA7	GPG[2]	MFP7
		GPA[1]	MFP10
	CCAP_FIELD	GPG[11]	MFP7

		GPB[11]	MFP12
	CCAP_HSYNC	GPG[13]	MFP7
		GPB[9]	MFP14
	CCAP_PCLK (从Sensor输入)	GPG[9]	MFP7
		GPB[13]	MFP10
	CCAP_VSYNC	GPG[12]	MFP7
		GPB[10]	MFP12

6.39.5 功能描述

6.39.5.1 图像获取流程

用户需要下面设定才能用CCAP功能从图像传感器采集一幅图像数据到内存

- 配置GPIO为MFP
- 配置并使能CCAP的时钟
- 根据CMOS传感器的规格配置相关控制寄存器
- 配置中断相关寄存器，使能或屏蔽中断
- 配置内置DMA相关寄存器，移动数据到系统内存
- 配置CCAP控制寄存器（裁剪，快门模式，亮度8位转1位，帧率下降，数据包水平/垂直缩放）
- 配置控制寄存器使能图像数据采集输出

6.39.5.2 传感器类型及输出格式

CCAP支持两种传感器：黑白CMOS传感器及彩色CMOS传感器，用户要根据传感器类型设置MONO(CCAP_CTL[7])

CCAP支持两种传感器数据输入模式(YCbCr422,RGB565)，用户需要根据所使用的传感器类型设置INFMT(CCAP_PAR[0])

CCAP支持两种传感器数据输入类型(CCIR601,CCIR656)，用户需要根据传感器设置SENTYPE(CCAP_PAR[1])

CCAP支持不同的传感器数据输入顺序，用户要根据传感器设置INDATORD(CCAP_PAR[3:2])

INFMT(CCAP_PAR[0])	INDATORD(CCAP_PAR[3:2])	传感器输入数据顺序
0 (YCbCr)	0	Byte 0 1 2 3 : Y0 U0 Y1 V0
0 (YCbCr)	1	Byte 0 1 2 3 : Y0 V0 Y1 U0
0 (YCbCr)	2	Byte 0 1 2 3 : U0 Y0 V0 Y1
0 (YCbCr)	3	Byte 0 1 2 3 : V0 Y0 U0 Y1
1 (RGB565)	0	Byte 0 1 :{R[4:0],G[5:3]} {G[2:0],B[4:0]}
1 (RGB565)	1	Byte 0 1 :{B[4:0],G[5:3]} {G[2:0],R[4:0]}
1 (RGB565)	2	Byte 0 1 :{G[2:0],B[4:0]} {R[4:0],G[5:3]}
1 (RGB565)	3	Byte 0 1 :{G[2:0],R[4:0]} {B[4:0],G[5:3]}

CCAP 支持不同的传感器信号极性(VSYNC,HSYNC,PCLKP)，用户需要根据传感器设置 VSP(CCAP_PAR[10]), HSP(CCAP_PAR[9]), PLKP(CCAP_PAR[8])

对于黑白CMOS传感器，CCAP支持两种数据I/O接口和数据顺序，客户需要根据所选传感器设置 MY8_MY4(CCAP_CTL[18]) 和 MY4_SWAP(CCAP_CTL[17])

CCAP支持4种图像数据格式输出到内存(YCbCr422，只有Y,RGB555,RGB565)，用户需要根据需要设置 OUTFMT(CCAP_PAR[5:4])

用户如果设置UPDATE(CCAP_CTL[24])为1，那么在新的帧开始前这些寄存器将会被更新

6.39.5.3 CCAP中断使能及屏蔽

CCAP支持三个中断源：

- 地址匹配中断:当前数据包内存地址寄存器(CCAP_CURADDRP)等于比较内存地址寄存器(CCAP_CMPADDR)将产生中断
- 总线传输错误中断：CCAP发出传输请求给AHB得到一个错误响应时将触发此中断
- 视频帧结束中断：当采集一帧数据结束的时候将触发这个中断

用户可以通过设置摄像头采集接口中断寄存器(CCAP_INT)使能或屏蔽这些中断

6.39.5.4 通过内置DMA控制器传输数据

CCAP 拥有内置的DMA控制器可以将数据从CCAP内部FIFO传送到系统内存。用户可以通过 CCAP_PKTBA0控制寄存器设置系统内存的基地址，通过PKTSTRIDE设置两条线间跨越的像素，用户也可以读取CCAP_CURADDRP获知CCAP将写数据到哪个地址

用户可以通过PKTFTH(CCAP_FIFOTH[28:24])设置门槛值，当内部FIFO收到的数据大于PKTFTH将触发内嵌DMA搬移数据到系统内存，用户也可以通过OVF(CCAP_FIFOTH[31]) 知道内部FIFO是否溢出

6.39.5.5 裁剪(CROP)功能

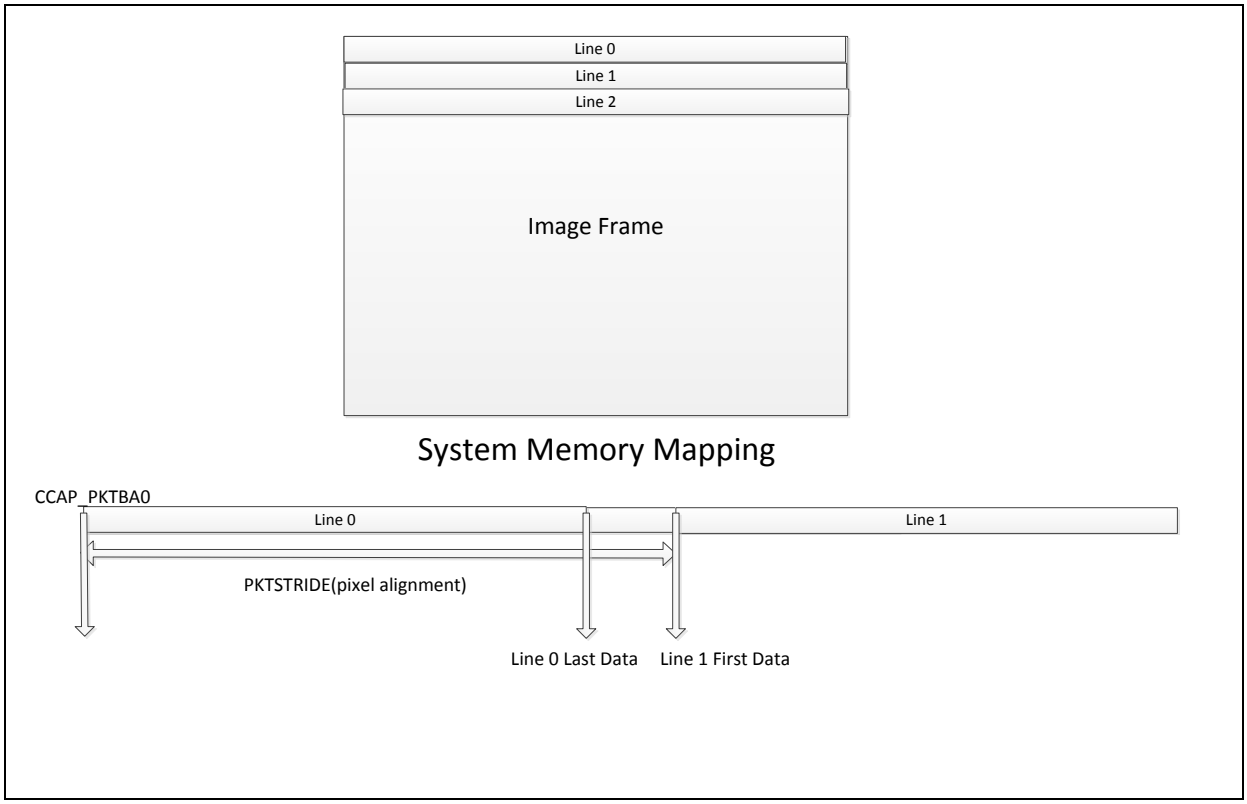


图 6.39-3 区域裁剪内存映射

数据采集接口可以从接收到的图片中选择一个窗口，窗口的尺寸由像素时钟(水平方向)及像素线数（垂直方向）决定。开始坐标（左上角）由CCAP_CWSP定义，尺寸由CCAP_CWS定义（垂直方向的像素线数及水平方向的像素数据）

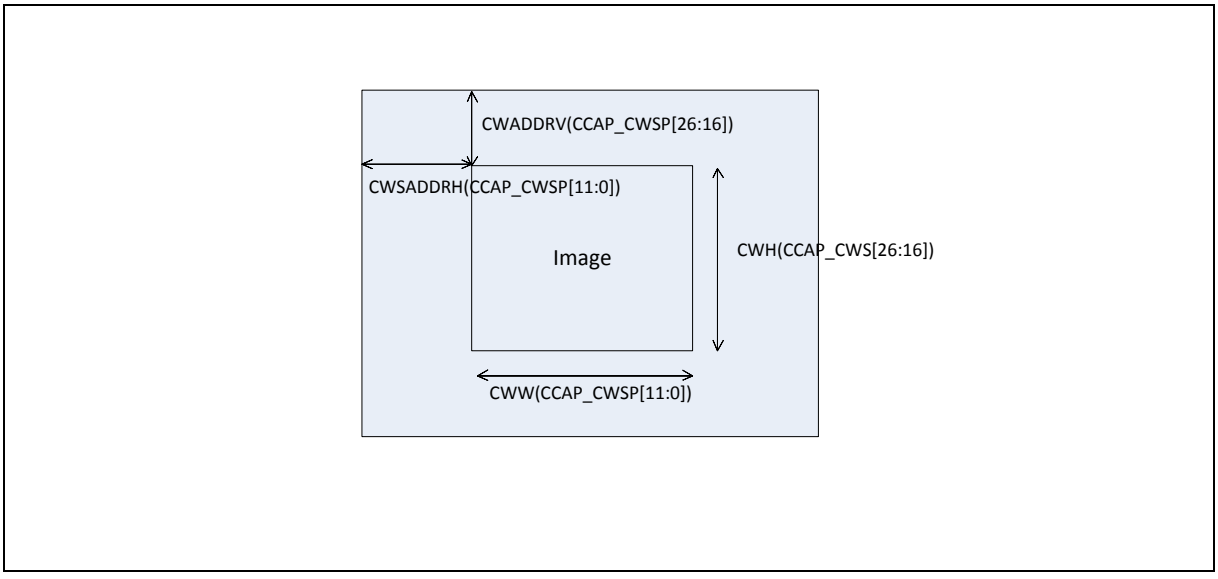


图 6.39-4 区域裁剪后的窗口图像开始及尺寸

6.39.5.6 快门模式 (单帧)

此模式下只采集一帧图像，SHUTTER (CCAP_CTL[16])置1，摄像头接口在采集一帧数据后自动停止

6.39.5.7 亮度8位转1位

当Luma_Y_one (CCAP_CTL[19])置1，将根据LUMA_Y1_THRESH (CCAP_LUMA_Y1_THD[7:0])的设定将8位亮度数据转成1位，当数据大于LUMA_Y1_THRESH将输出1，反之输出0

6.39.5.8 帧率下降

CCAP可以通过FRM(CCAP_FRCTL[5:0]) 和 FRN(CCAP_FRCTL[13:8])实现帧率下降功能，输出图像的帧率等于输入帧率*(FRN/FRM)

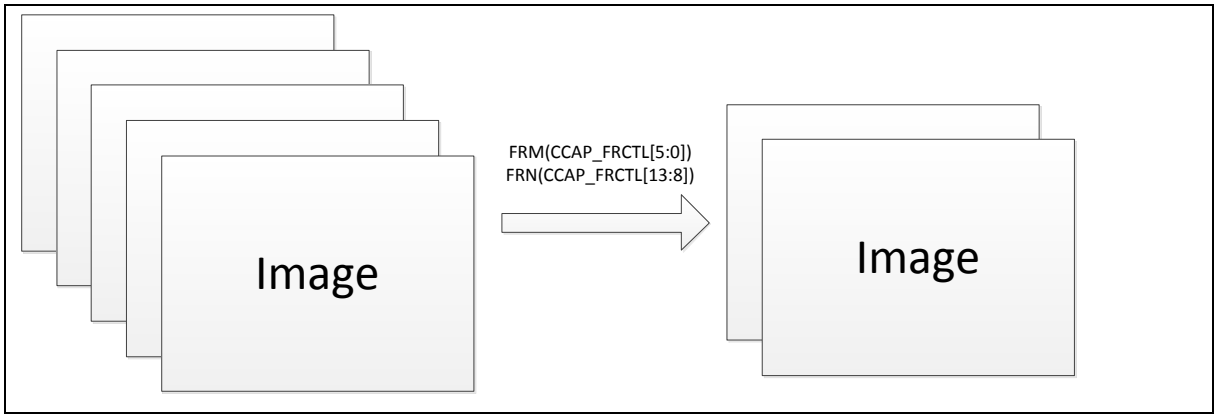


图 6.39-5 帧率下降框图

6.39.5.9 数据包垂直/水平缩放

CCAP可以设置CCAP_PKTSM 和 CCAP_PKTSL达到输入图像的垂直/水平缩放

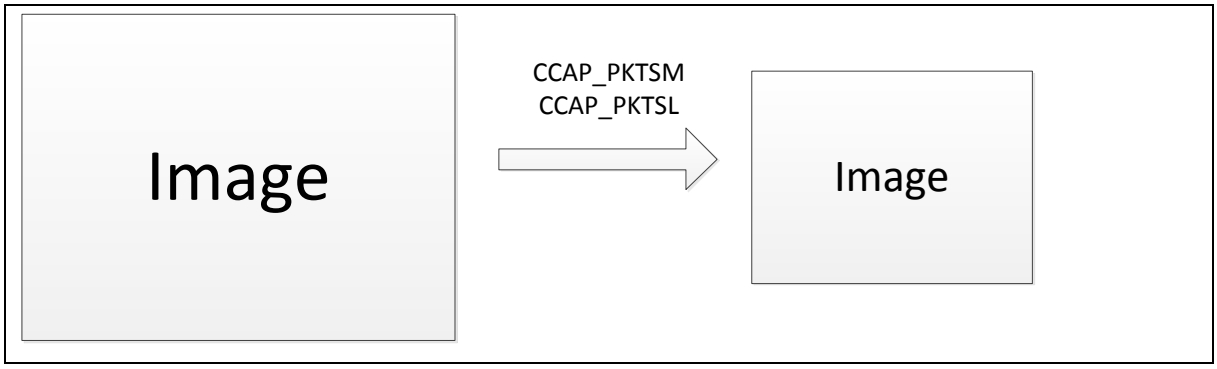


图 6.39-6 帧数据包缩放框图

6.39.5.10 使能采集接口

用户可以设置CCAPEN(CCAP_CTL[0])使能采集接口，设置PKTEN (CCAP_CTL[6])使能数据包输出

6.39.6 寄存器表

R: 只读, **W**: 只写, **R/W**: 可读写

寄存器	偏移量	R/W	描述	复位值
CCAP 基地址: **CCAP_BA = 0x4003_0000**				
CCAP_CTL	CCAP_BA+0x00	R/W	摄像头接口控制寄存器	0x0000_0050
CCAP_PAR	CCAP_BA+0x04	R/W	摄像头接口参数寄存器	0x0000_0000
CCAP_INT	CCAP_BA+0x08	R/W	摄像头接口中断寄存器	0x0000_0000
CCAP_CWSP	CCAP_BA+0x20	R/W	窗口裁剪开始地址寄存器	0x0000_0000
CCAP_CWS	CCAP_BA+0x24	R/W	裁剪窗口尺寸寄存器	0x0000_0000
CCAP_PKTSL	CCAP_BA+0x28	R/W	数据包垂直/水平缩放因子寄存器（低位）	0x0000_0000
CCAP_FRCTL	CCAP_BA+0x30	R/W	降低帧率因子寄存器	0x0000_0000
CCAP_STRIDE	CCAP_BA+0x34	R/W	帧输出像素跨越宽度寄存器	0x0000_0000
CCAP_FIFOTH	CCAP_BA+0x3C	R/W	FIFO门槛寄存器	0x070D_0507
CCAP_CMPADDR	CCAP_BA+0x40	R/W	比较内存基地址寄存器	0xFFFF_FFFC
CCAP_LUMA_Y1_THD	CCAP_BA+0x44	R/W	亮度数据8位转1位门槛寄存器	0x0000_0080
CCAP_PKTSM	CCAP_BA+0x48	R/W	数据包垂直/水平缩放因子寄存器（高位）	0x0000_0000
CCAP_CURADDRP	CCAP_BA+0x50	R	当前数据包系统内存地址寄存器	0x0000_0000
CCAP_PKTBA0	CCAP_BA+0x60	R/W	系统内存数据包基地址0寄存器	0x0000_0000

6.39.7 寄存器描述

摄像头接口控制寄存器(CCAP_CTL)

寄存器	偏移量	R/W	描述	复位值
CCAP_CTL	CCAP_BA+0x00	R/W	摄像头接口控制寄存器	0x0000_0050

31	30	29	28	27	26	25	24
Reserved							VPRST
23	22	21	20	19	18	17	16
Reserved			UPDATE	Luma_Y_One	MY8_MY4	MY4_SWAP	SHUTTER
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
MONO	PKTEN	Reserved					CCAPEN

位	描述	
[31:25]	**Reserved**	保留
[24]	**VPRST**	**采集接口复位** 0 = 禁止采集接口复位 1 = 使能采集接口复位
[23:21]	**Reserved**	保留
[20]	**UPDATE**	**新的帧更新寄存器** 0 = 禁止新的帧更新寄存器 1 = 新的帧更新寄存器（寄存器更新后自动清0）
[19]	**Luma_Y_One**	**彩色/黑白CMOS传感器亮度8位转1位** 0 = 彩色/黑白CMOS传感器1位亮度禁止 1 = 彩色/黑白CMOS传感器1位亮度使能 **注:**彩色CMOS传感器在此位为1时U/V信号被忽略
[18]	**MY8_MY4**	**黑白CMOS传感器数据I/O接口** 0 = 黑白CMOS传感器使用4位数据I/O接口 1 = 黑白CMOS传感器使用8位数据I/O接口
[17]	**MY4_SWAP**	**黑白CMOS传感器4位数据顺序** 0 = 4位数据顺序：第一个像素在高4位 1 = 4位数据顺序：第一个像素在低4位
[16]	**SHUTTER**	**摄像头采集接口采集完一帧数据后自动禁止采集接口（快门模式）** 0 = 快门模式禁止 1 = 快门模式使能
[15:8]	**Reserved**	保留

[7]	**MONO**	**黑白CMOS传感器选择** 0 = 彩色CMOS传感器 1 = 黑白CMOS传感器，当MONO使能后U/V数据被忽略
[6]	**PKTEN**	**数据包输出使能位** 0 = 数据包输出禁止 1 = 数据包输出使能
[5:1]	**Reserved**	保留
[0]	**CCAPEN**	**摄像头采集接口使能位** 0 = 摄像头采集接口禁止 1 = 摄像头采集接口使能

摄像头接口参数寄存器 (CCAP_PAR)

寄存器	偏移量	R/W	描述	复位值
CCAP_PAR	CCAP_BA+0x04	R/W	摄像头接口参数寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved					FBB	Reserved	
15	14	13	12	11	10	9	8
Reserved					VSP	HSP	PCLKP
7	6	5	4	3	2	1	0
Reserved	RANGE	OUTFMT		INDATORD		SENTYPE	INFMT

位	描述	
[31:19]	**Reserved**	保留
[18]	**FBB**	**空白域（只在CCIR656模式下）** 空白域表示空白像素数据(0x80108010)是否必须传送到系统内存 0 = 空白域禁止（空白像素数据将传送到系统内存） 1 = 空白域使能（仅有效数据会被传送到系统内存）
[10]	**VSP**	**传感器Vsync极性** 0 = Vsync 低有效 1 = Vsync 高有效
[9]	**HSP**	**传感器Hsync极性** 0 = Hsync 低有效 1 = Hsync 高有效
[8]	**PCLKP**	**传感器像素时钟极性** 0 = 输入视频数据信号在像素时钟下降沿锁存 1 = 输入视频数据信号在像素时钟上升沿锁存
[7]	**PLNFMT**	**平面输出YUV格式** 0 = YUV422. 1 = YUV420.
[6]	**RANGE**	**输入YUV CCIR601颜色映射到全彩** 0 = 默认 1 = 映射到全彩
[5:4]	**OUTFMT**	**输出到系统内存的图像数据格式** 00 = YCbCr422. 01 = 只有Y（当CCAP_CTL “Luma_Y_One”或“MONO”使能选择此格式）

		10 = RGB555. 11 = RGB565.
[3:2]	**INDATORD**	**传感器输入数据顺序** 如果INFMT (CCAP_PAR[0]) = 0 (YCbCr) 00 = 传感器输入数据(字节 0 1 2 3) is Y0 U0 Y1 V0. 01 = 传感器输入数据(字节0 1 2 3) is Y0 V0 Y1 U0. 10 = 传感器输入数据(字节0 1 2 3) is U0 Y0 V0 Y1. 11 = 传感器输入数据(字节0 1 2 3) is V0 Y0 U0 Y1. 如果 INFMT (CCAP_PAR[0]) = 1 (RGB565). 00 = 传感器输入数据(Byte 0) is {R[4:0],G[5:3]},. 传感器输入数据(字节1) is {G[2:0],B[4:0]}. 01 = 传感器输入数据(Byte 0) is {B[4:0],G[5:3]},. 传感器输入数据(字节1) is {G[2:0], R[4:0]}. 10 = 传感器输入数据(Byte 0) is {G[2:0],B[4:0]},. 传感器输入数据(字节1) is {R[4:0], G[5:3]}. 11 = 传感器输入数据(Byte 0) is {G[2:0],R[4:0]},. 传感器输入数据(字节1) is {B[4:0], G[5:3]}.
[1]	**SENTYPE**	**传感器输入类型** 0 = CCIR601. 1 = CCIR656, Vsync & Hsync 嵌入在数据信号中
[0]	**INFMT**	**传感器输入数据格式** 0 = YCbCr422. 1 = RGB565.

摄像头接口中断寄存器(CCAP_INT)

寄存器	偏移量	R/W	描述	复位值
CCAP_INT	CCAP_BA+0x08	R/W	摄像头接口中断寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved				ADDRMIEN	Reserved	MEIEN	VIEN
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved				ADDRMINTF	Reserved	MEINTF	VINTF

位	描述	
[31:20]	**Reserved**	保留
[19]	**ADDRMIEN**	**地址匹配中断使能** 0 = 地址匹配中断禁止 1 = 地址匹配中断使能
[18]	**Reserved**	保留
[17]	**MEIEN**	**总线传输错误中断使能** 0 = 总线传输错误中断禁止 1 = 总线传输错误中断使能
[16]	**VIEN**	**视频帧结束中断使能** 0 = 视频帧结束中断禁止 1 = 视频帧结束中断使能
[15:4]	**Reserved**	保留
[3]	**ADDRMINTF**	**内存地址匹配中断** 0 = 内存地址匹配中断没有发生 1 = 内存地址匹配中断发生了 **注:**此位写1清零
[2]	**Reserved**	保留
[1]	**MEINTF**	**总线错误中断** 0 = 总线错误没有发生 1 = 发生总线传输错误 **注:**此位写1清零
[0]	**VINTF**	**视频帧结束中断**

		0 = 没有收到一个完整帧 1 = 收到一个完整帧 **注:**此位写1清零

窗口裁剪开始地址寄存器(CCAP_CWSP)

寄存器	偏移量	R/W	描述	复位值
CCAP_CWSP	CCAP_BA+0x20	R/W	窗口裁剪开始地址寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved					CWSADDRV		
23	22	21	20	19	18	17	16
CWSADDRV							
15	14	13	12	11	10	9	8
Reserved				CWSADDRH			
7	6	5	4	3	2	1	0
CWSADDRH							

位	描述	
[31:27]	Reserved	保留
[26:16]	CWSADDRV	**裁剪窗口垂直开始地址** 指定裁剪窗口的垂直开始地址
[15:12]	Reserved	保留
[11:0]	CWSADDRH	**裁剪窗口水平开始地址** 指定裁剪窗口的水平开始地址

裁剪窗口尺寸寄存器(CCAP_CWS)

寄存器	偏移量	R/W	描述	复位值
CCAP_CWS	CCAP_BA+0x24	R/W	裁剪窗口尺寸寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved					CWH		
23	22	21	20	19	18	17	16
CWH							
15	14	13	12	11	10	9	8
Reserved				CWW			
7	6	5	4	3	2	1	0
CWW							

位	描述	
[31:27]	**Reserved**	保留
[26:16]	**CWH**	**裁剪窗口高度** 指定裁剪窗口的高度
[15:12]	**Reserved**	保留
[11:0]	**CWW**	**裁剪窗口宽度** 指定裁剪窗口的宽度

数据包垂直/水平缩放因子寄存器 (低位) (CCAP_PKTSL)

寄存器	偏移量	R/W	描述	复位值
CCAP_PKTSL	CCAP_BA+0x28	R/W	数据包垂直/水平缩放因子寄存器（低位）	0x0000_0000

31	30	29	28	27	26	25	24
PKTSVNL							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
PKTSVML							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
PKTSHNL							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
PKTSHML							

位	描述	
[31:24]	**PKTSVNL**	**数据包缩放垂直因子N（低8位）** 垂直缩放因子N的低8位数据 此低8位数据将与高8位数据(PKDSVNH)组成16位垂直缩放因子N
[23:16]	**PKTSVML**	**数据包缩放垂直因子M(低8位)** 垂直缩放因子M的低8位 此低8位数据将与高8位(PKDSVMH)数据组成16位的垂直缩放因子M 输出图像的高度将等于原来图像高度* * N/M. **注:**N必须小于等于M
[15:8]	**PKTSHNL**	**数据包缩放水平因子N(低8位)** 水平缩放因子N的低8位 此低8位数据将与高8位数据(PKDSHNH)组成16位水平缩放因子N
[7:0]	**PKTSHML**	**数据包缩放水平因子M(低8位)** 水平缩放因子M的低8位 此低8位数据将与高8位数据(PKDSHMH)组成16位缩放因子M 输出图像的宽度将是原来图片宽度* N/M. **注:** N必须小于等于M

降低帧率因子寄存器(CCAP_FRCTL)

寄存器	偏移量	R/W	描述	复位值
CCAP_FRCTL	CCAP_BA+0x30	R/W	降低帧率因子寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	28	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved		FRN					
7	6	5	4	3	2	1	0
Reserved		FRM					

位	描述	
[31:14]	Reserved	保留
[13:8]	FRN	**降低帧率因子N** 降低帧率公式中的因子N
[7:6]	Reserved	保留
[5:0]	FRM	**降低帧率因子M** 降低帧率公式中的因子M 输出的图像帧率将等于原来帧率*（N/M） **注: N必须小于等于M**

帧输出像素跨越宽度寄存器(CCAP_STRIDE)

寄存器	偏移量	R/W	描述	复位值
CCAP_STRIDE	CCAP_BA+0x34	R/W	帧输出像素跨越宽度寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	28	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved		PKTSTRIDE					
7	6	5	4	3	2	1	0
PKTSTRIDE							

位	描述	
[31:14]	Reserved	保留
[13:0]	PKTSTRIDE	**帧数据包输出像素跨越** 数据包输出像素跨越数量 数据格式为1位时（黑白）输出跨越宽度32像素对齐，数据格式为8位（彩色或黑白）时4像素对齐。这意味着不管彩色还是黑白传感器，每条像素线开始地址都是字对齐

FIFO门槛寄存器(CCAP_FIFOTH)

寄存器	偏移量	R/W	描述	复位值
CCAP_FIFOTH	CCAP_BA+0x3C	R/W	FIFO门槛寄存器	0x070D_0507

31	30	29	28	27	26	25	24
OVF	Reserved		PKTFTH				
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved							

位	描述	
[31]	OVF	**FIFO溢出标志** 表示FIFO溢出标志
[30:29]	Reserved	保留
[28:24]	PKTFTH	**数据包FIFO门槛** 设定5位数据的FIFO门槛
[23:0]	Reserved	保留

比较内存基地址寄存器(CCAP_CMPADDR)

寄存器	偏移量	R/W	描述	复位值
CCAP_CMPADDR	CCAP_BA+0x40	R/W	比较内存基地址寄存器	0xFFFF_FFFC

31	30	29	28	27	26	25	24
CMPADDR							
23	22	21	20	19	18	17	16
CMPADDR							
15	14	13	12	11	10	9	8
CMPADDR							
7	6	5	4	3	2	1	0
CMPADDR							

位	描述	
[31:0]	CMPADDR	比较内存基地址寄存器 这是一个字对齐的地址，也就是地址的低2位被忽略

亮度数据8位转1位门槛寄存器(CCAP_LUMA_Y1_THD)

寄存器	偏移量	R/W	描述	复位值
CCAP_LUMA_Y1_THD	CCAP_BA+0x44	R/W	亮度数据8位转1位门槛寄存器	0x0000_0080

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
LUMA_Y1_THRESH							

位	描述	
[31:8]	Reserved	保留
[7:0]	LUMA_Y1_THRESH	亮度数据8位转1位门槛寄存器 亮度信息8位转1位的门槛值

数据包垂直/水平缩放因子寄存器（高位） (CCAP_PKTSM)

寄存器	偏移量	R/W	描述	复位值
CCAP_PKTSM	CCAP_BA+0x48	R/W	数据包垂直/水平缩放因子寄存器（高位）	0x0000_0000

31	30	29	28	27	26	25	24
PKTSVNH							
23	22	21	20	19	18	17	16
PKTSVMH							
15	14	13	12	11	10	9	8
PKTSHNH							
7	6	5	4	3	2	1	0
PKTSHMH							

位	描述	
[31:24]	PKTSVNH	**数据包缩放垂直因子N（高8位）** 垂直缩放因子N的高8位数据 请参考寄存器“CCAP_PKTSL”以了解这两个寄存器间的协作
[23:16]	PKTSVMH	**数据包缩放垂直因子M(高8位)** 垂直缩放因子M的高8位 请参考寄存器“CCAP_PKTSL”以了解这两个寄存器间的协作
[15:8]	PKTSHNH	**数据包缩放水平因子N(高8位)** 水平缩放因子N的高8位 请参考寄存器“CCAP_PKTSL”以了解这两个寄存器间的协作
[7:0]	PKTSHMH	**数据包缩放水平因子M(低8位)** 水平缩放因子M的低8位 请参考寄存器“CCAP_PKTSL”以了解这两个寄存器间的协作

Current Packet System Memory Address Register (CCAP_CURADDRP)

Register	Offset	R/W	Description	Reset Value
CCAP_CURADDRP	CCAP_BA+0x50	R	Current Packet System Memory Address Register	0x0000_0000

31	30	29	28	27	26	25	24
CURADDR							
23	22	21	20	19	18	17	16
CURADDR							
15	14	13	12	11	10	9	8
CURADDR							
7	6	5	4	3	2	1	0
CURADDR							

Bits	Description	
[31:0]	CURADDR	**Current Packet Output Memory Address** Speficy the 32-bit value of the current packet output memory address.

系统内存数据包基地址0寄存器(CCAP_PKTBA0)

寄存器	偏移量	R/W	描述	复位值
CCAP_PKTBA0	CCAP_BA+0x60	R/W	系统内存数据包基地址0寄存器	0x0000_0000

31	30	29	28	27	26	25	24
BASEADDR							
23	22	21	20	19	18	17	16
BASEADDR							
15	14	13	12	11	10	9	8
BASEADDR							
7	6	5	4	3	2	1	0
BASEADDR							

位	描述	
[31:0]	BASEADDR	系统内存数据包基地址0寄存器 这是一个字对齐的地址，也就是地址的低2位被忽略

6.40 EADC 增强型 12位模数转换器

6.40.1 概述

M480 含一个或两个12位逐次比较式的 EADC，有16个外部输入和3个内部输入。转换可由软件触发, EPWM0/1 触发, BPWM0/1 触发, timer0~3 触发， ADINT0, ADINT1 中断 EOC (转换结束)触发和外部管脚(EADC0/1_ST) 触发。

6.40.2 特性

- 输入电压范围: 0~ V_{REF} (最大到 3.6V)
- 参考电压： V_{REF}或AV_{DD}
- 12位分辨率， 10位精度保证
- 16 路单端输入，或8 对差分输入
- 3 路内部输入分别是内部带隙电压 (V_{BG}), 温度传感器 (V_{TEMP}), 和电池电压(V_{BAT})
- 4 个 EADC 中断 (ADINT0~3) ，都有独立的中断向量地址
- 时钟频率最大 72 MHz
- 每个EADC的转换率高达 5.14 MSPS
- 采样时间可配置.
- 可配置12-位, 10-位, 8-位, 6-位模式.
- 支持校准和载入较准字
- 内部参考电压 VREF: 1.6V, 2.0V, 2.5V, 和 3.0V.
- 支持3种省电模式:
  - 深度掉电模式
  - 掉电模式
  - 待机模式
- 多达 19 个采样模块
  - 每个采样模块可配置EADC 0/1 通道 EADC_CH0~15 ，触发源和采样时间
  - EADC0 16, 17, 18通道固定为：带隙电压，温度传感器和电池电压(V_{BAT})
  - 采样模块0~3双数据缓存
  - 转换结果保存在 19 个数据寄存器，有有效位和覆盖标志
- 一次 EADC 可以由下列条件触发:
  - 写 1 到 SWTRGn (EADC0/1_SWTRG[n] , n = 0~18)
  - 外部管脚 EADC0/1_ST
  - Timer0~3 溢出脉冲
  - ADINT0 和 ADINT1 的EOC (转换结束)中断脉冲
  - EPWM / BPWM

- 支持 PDMA 传输
- 有转换结果监控比较模式

	M48xID/M48xGA	M48xGC/M48xE8
EADC数量	1	2

表 1.1-1 EADC 性能比较表

6.40.3 框图

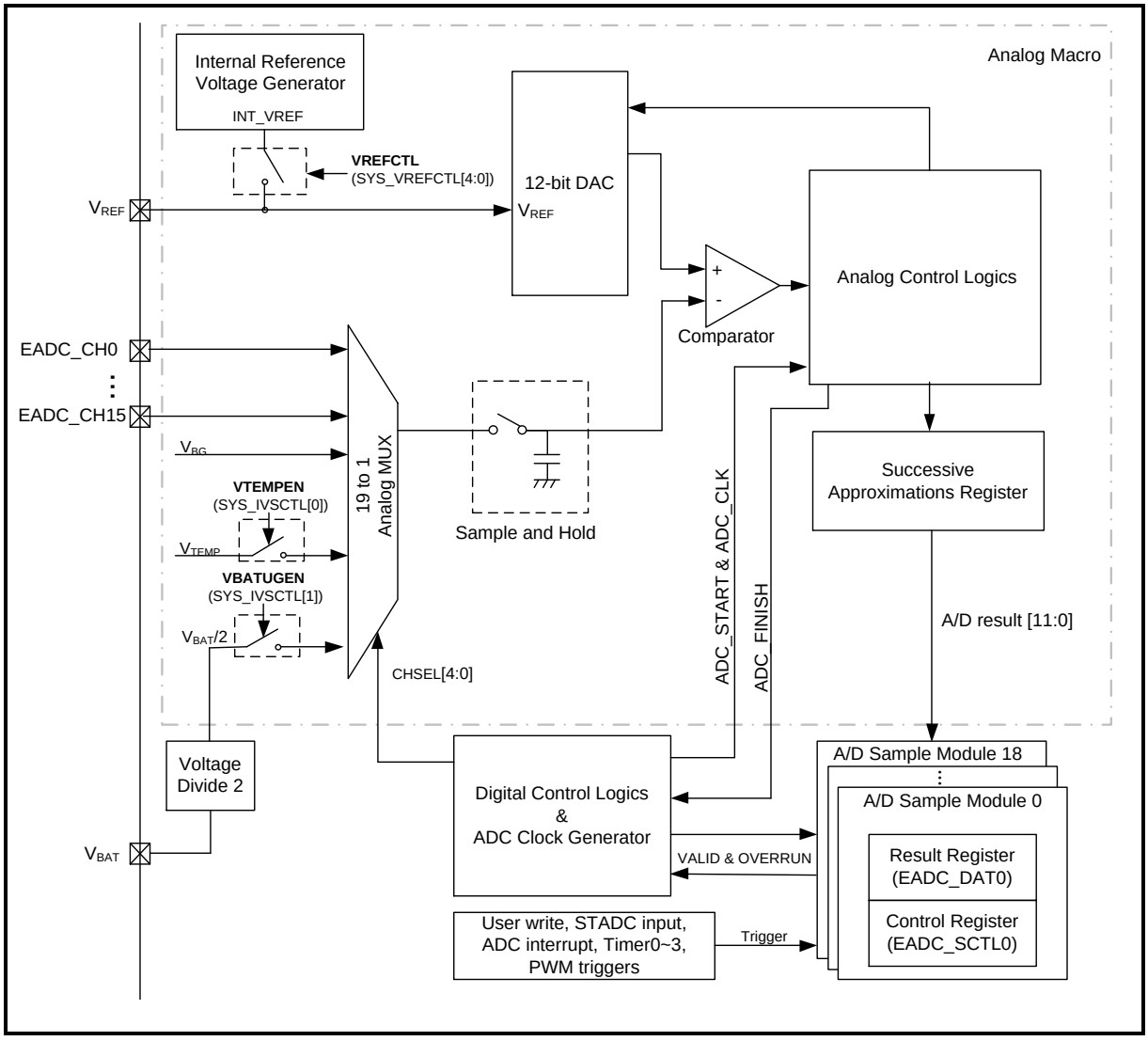


图 6.40-1 EADC 转换框图

6.40.4 基本配置

6.40.4.1 EADC0 基本配置

- 时钟源配置
  - 在 EADC0DIV (CKL_CLKDIV0[23:16])选择时钟除频
  - 在EADC0CKEN (CLK_APBCLK0[28])使能EADC 0外设时钟.
- 复位配置
  - 复位 EADC0 控制器在 EADCRST(SYS_IPRST1[28])
- 管脚配置

组	管脚名	GPIO	MFP
EADC0	EADC0_CH0	PB.0	MFP1
	EADC0_CH1	PB.1	MFP1
	EADC0_CH2	PB.2	MFP1
	EADC0_CH3	PB.3	MFP1
	EADC0_CH4	PB.4	MFP1
	EADC0_CH5	PB.5	MFP1
	EADC0_CH6	PB.6	MFP1
	EADC0_CH7	PB.7	MFP1
	EADC0_CH8	PB.8	MFP1
	EADC0_CH9	PB.9	MFP1
	EADC0_CH10	PB.10	MFP1
	EADC0_CH11	PB.11	MFP1
	EADC0_CH12	PB.12	MFP1
	EADC0_CH13	PB.13	MFP1
	EADC0_CH14	PB.14	MFP1
	EADC0_CH15	PB.15	MFP1
	EADC0_ST	PF.5	MFP11
		PC.13, PD.12	MFP14
		PG.15	MFP15

6.40.4.2 EADC1 基本配置

- 时钟源配置
  - 在 EADC1DIV (CKL_CLKDIV2[31:24])选择时钟除频
  - EADC1CKEN (CLK_APBCLK1[31])使能EADC1外设时钟
- 复位配置
  - EADC1RST (SYS_IPRST2[31])复位EADC1

- 管脚配置

组	管脚名	GPIO	MFP
EADC1	EADC1_CH0	PD.10	MFP1
	EADC1_CH1	PD.11	MFP1
	EADC1_CH2	PD.12	MFP1
	EADC1_CH3	PC.13	MFP1
	EADC1_CH4	PA.8	MFP1
	EADC1_CH5	PA.9	MFP1
	EADC1_CH6	PA.10	MFP1
	EADC1_CH7	PA.11	MFP1
	EADC1_CH8	PB.0	MFP1
	EADC1_CH9	PB.1	MFP1
	EADC1_CH10	PB.2	MFP1
	EADC1_CH11	PB.3	MFP1
	EADC1_CH12	PB.12	MFP1
	EADC1_CH13	PB.13	MFP1
	EADC1_CH14	PB.14	MFP1
	EADC1_CH15	PB.15	MFP1
	EADC1_ST	PF.4	MFP11
		PC.9, PC.10	MFP14
		PG.0	MFP15

6.40.5 功能描述

EADC0/1 包含19个模拟开关，19个采样模块以及一个12位转换器。模块0~18, 每一个都可以配置触发源，转换通道。模块 0~15 可以配置到信道 EADC0/1_CH0~15, 以及不同的触发源。模块 0~3 和模块 4~15 如下图所示.

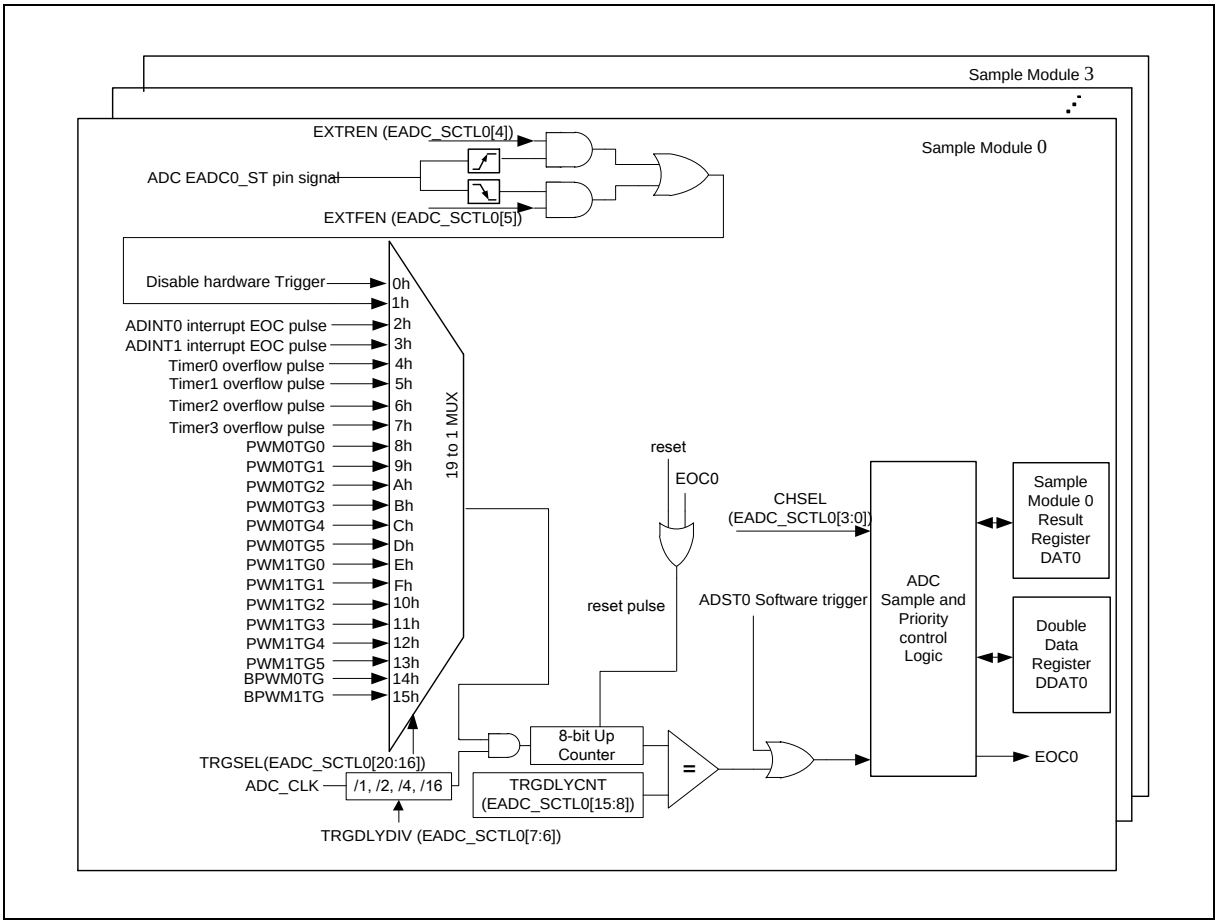


图 6.40-2 模块 0~3 框图

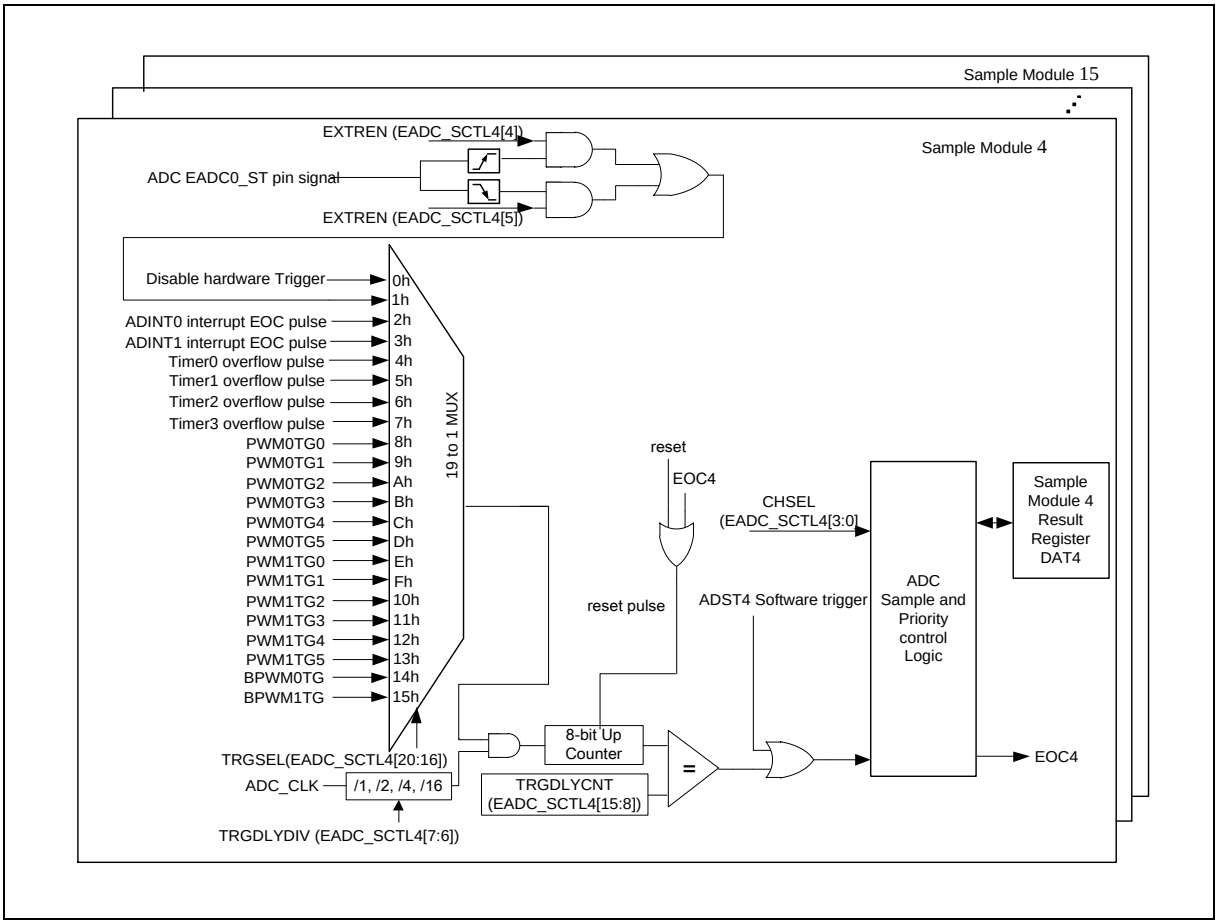


图 6.40-3 模块 4~15 框图

模块 16~18 转换内部通道 (V_{BG}, V_{TEMP}, V_{BAT})，可以通过写SWTRGn (EADC0/1_SWTRG[n], n = 16~18)触发。图6.40-4展示模块 16~18.

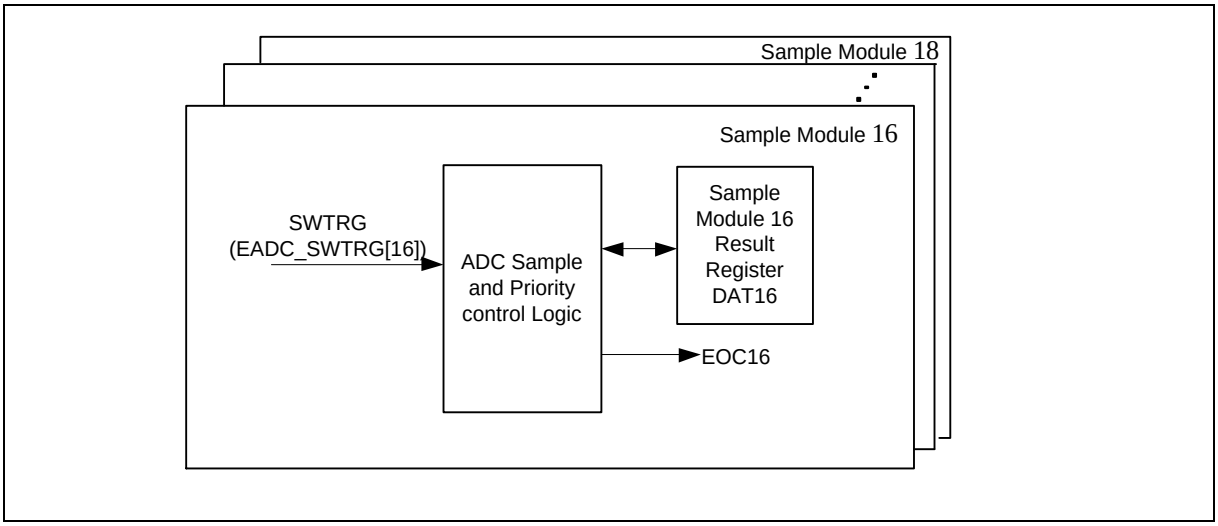


图 6.40-4 模块 16~18 框图

- 模块0~15的EADC0/1 触发源如下:
- 写 1 到 SWTRGn (EADC0/1_SWTRG[n], n = 0~15)
- 外部管脚 EADC0/1_ST
- Timer0~3 溢出脉冲
- ADINT0, ADINT1 EADC0/1 的 EOC (转换结束)中断
- EPWM/BPWM

当指定的采样模块转换结束（产生EADC0/1 EOC信号），将产生 ADINT0或ADINT1 中断。在连续扫描模式下此中断信号可以用来触发其他EADC0/1通道的转换。

6.40.5.1 EADC0/1时钟产生

EADC0/1时钟频率最大可达 72 MHz，采样率最高可达 5.14 MSPS.

EADC0/1 时钟控制如图6.40‑5. EADC 0/1时钟源来自 HCLK, EADC0/1 时钟被一个8位计数器分频:

EADC0/1 时钟频率 = (PCLK1) / (EADCDIV0/1 (CKL_CLKDIV0[23:16])+1)

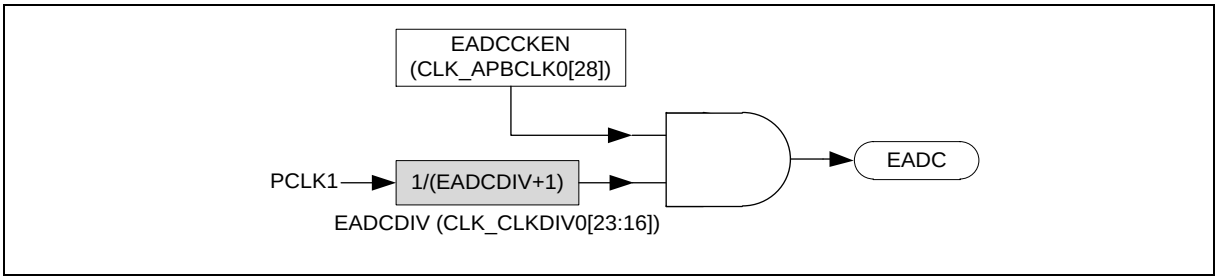


图 6.40‑5 EADC0/1 时钟控制

6.40.5.2 EADC0/1软件触发模式

在指定的EADC0/1通道上开始单端转换,操作如下:

1. EADC0/1 转换由SWTRGn (EADC0/1_SWTRG[n], n=0~18) 置 1 触发或其他输入触发.
2. 当 EADC0/1 转换完成，12 位结果将储存在 EADC0/1 相应采样模块寄存器中 EADC0/1_DATn (n=0~18)
3. 当 EADC0/1 转换已完成，ADIFn (EADC0/1_STATUS2[3:0], n=0~3)将被置1，如果中断使能位EADCIENn (EADC_CTL[5:2], n=0~3) 为1，将产生EADC0/1 中断 (ADINTn, n=0~3)
4. SWTRGn (n=0~18) 在EADC0/1转换过程中一直保持为1，当EADC0/1转换结束，SWTRGn (n=0~18)自动清零，EADC0/1将做其他等待的转换

转换周期的时序图如图6.40‑6.

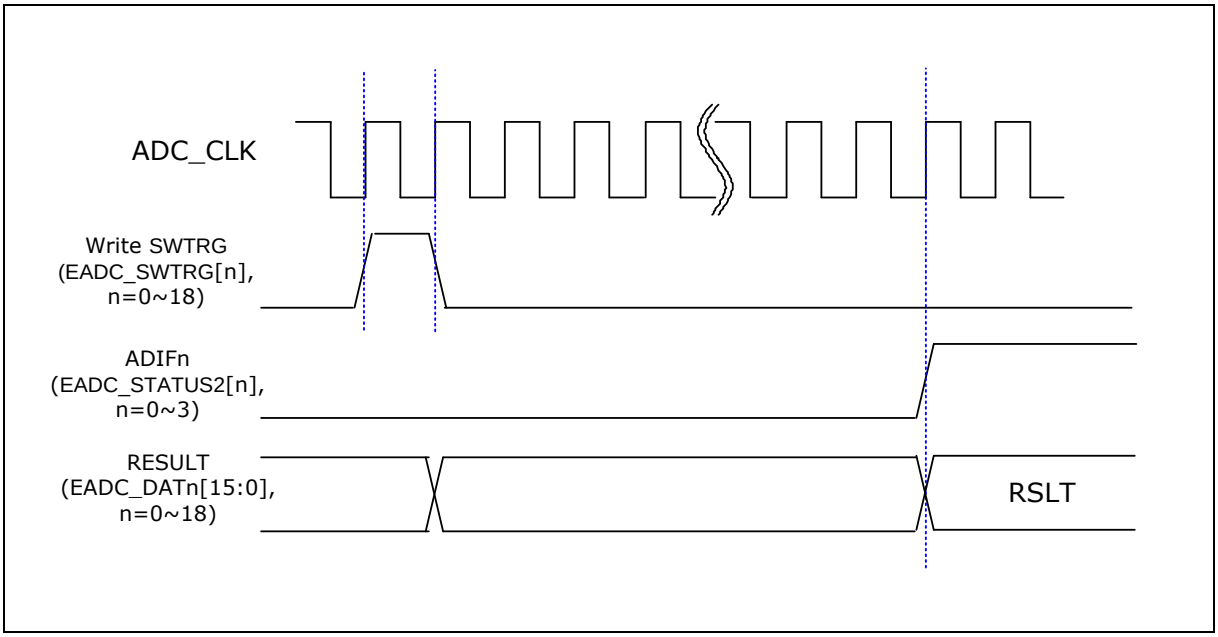


图 6.40-6 EADC0/1 转换时序图, n=0~18

如果多于1个模块使能转换，高优先级的模块先转换，低优先级的等待。模块编号较低的有较高的优先级。模块0 有最高的优先级，采样模块 18优先级最低.

注:如果下次转换间隔超过100us，EADC0/1会自动地进入空闲状态。使用前用户需要执行一次空转换，也就是当EADC0/1处在空闲状态时第一次转换的结果是不准确的。

6.40.5.3 EADC0/1 转换优先级

模块优先级见图6.40-7。当多个模块同时启动转换，编号较低的先转换，其他模块进入等待队列，等待标志 STPF(EADC0/1_PENDSTS[n], n=0~18) 由硬件置 1 。当模块完成转换，STPF(EADC0/1_PENDSTS[n], n=0~18) 自动清0. 如果队列里的采样模块被触发多于一次，覆盖标志 SPOVF(EADC0/1_OVSTS[n], n=0~18)由硬件置 1 .

例如, 采样模块 0, 2, 3, 5 同时触发。采样模块 0 的通道会先转换。模块 2, 3, 5 会挂起，且 STPF (EADC0/1_PENDSTS[2], EADC0/1_PENDSTS [3], EADC0/1_PENDSTS [5]) 置 1. 如果模块 5 同时触发2次, SPOVF(EADC0/1_OVSTS[5]) 置 1.

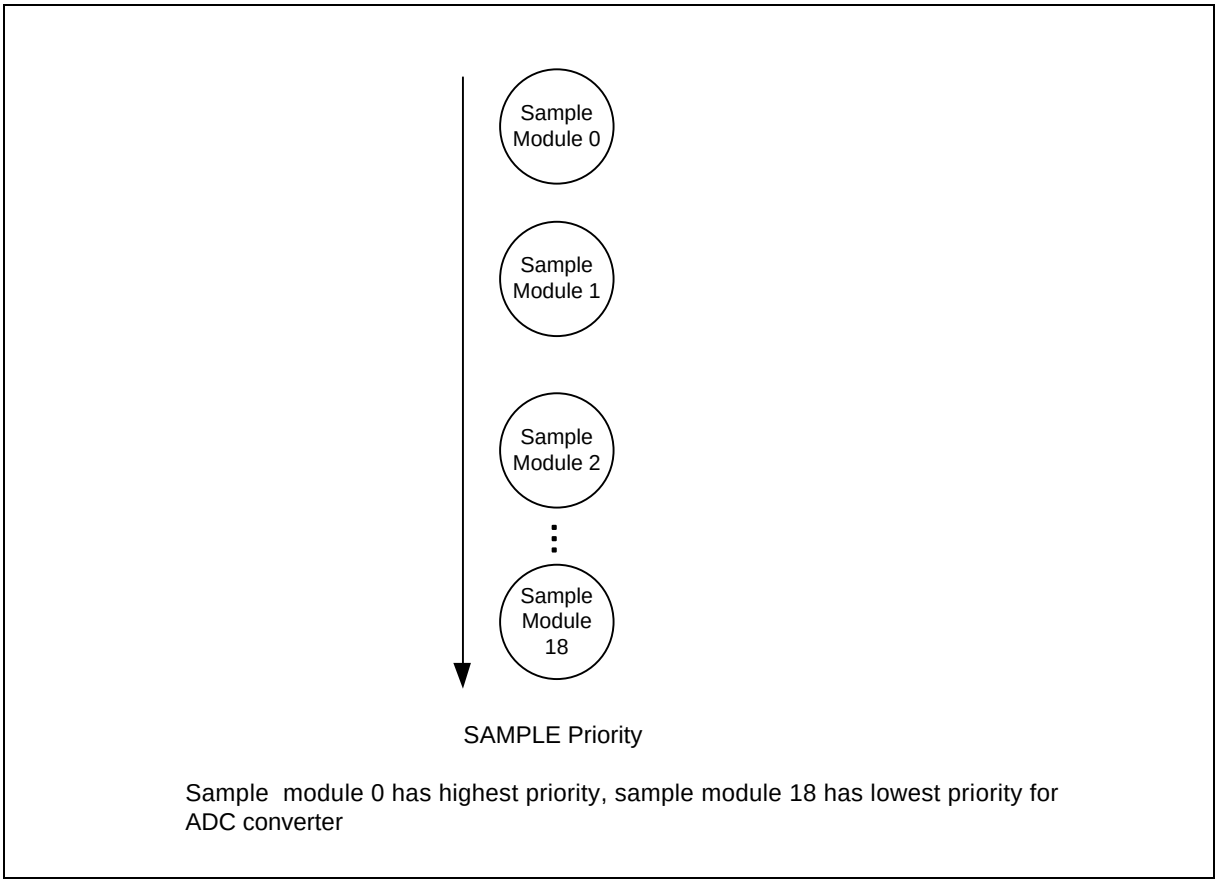


图6.40-7模块转换优先级仲裁框图

6.40.5.4 转换周期和采样率

RESSEL (EADC0/1_CTL[7:6])可配置EADC0/1结果的位数，对应不同的转换周期。如表6.40-1.

分辨率	最小转换周期
6位	8 EADC_CLK
8 位	10 EADC_CLK
10 位	12 EADC_CLK
12位	14 EADC_CLK

表 6.40-1 分辨率与转换周期的关系

有高速信道和低速信道。EADC0/1_CH10~15是高速通道， EADC0/1_CH0~9 是低速通道。高速通道的最大采样率是 5.14 MSPS ，低速通道是 2.14 MSPS. 超过这个转换率会得到错误的结果。采样率由下面的公式计算：

采样率 = (EADC0/1 时钟频率) / (转换周期)

6.40.5.5 软件触发最大采样率

如果用户需要以最大采样率检测快速通道，转换需要在如下条件下：多采样模块, 由软件触发, 在上次转换后连续触发. 连续转换的例子如下：

1. 使用模块 0~15 来执行连续的转换. 设置 CHSEL (EADC0/1 _SCTL0~15[3:0]) 选择通道 (EADC0/1 _CH10~ EADC0/1 _CH15). 设置 EXTSMPT (EADC0/1 _SCTL0~15[31:24]) 和 TRGDLYCNT (EADC0/1 _SCTL0~15[15:8]) 为 0x00 选择最小采样时间.
2. 设置 SWTRG (EADC0/1 _SWTRG[18:0]) 为 0xffff 来触发模块 0~15.
3. 等待 CURSPL (EADC0/1 _STATUS3[4:0]) 变为 0xf 意味着模块 0~14 已经执行，而模块15正在执行. 设置SWTRG (EADC0/1 _SWTRG[18:0]) 为 0x7fff 来触发模块 0~14 下一轮转换.
4. 等待 CURSPL (EADC0/1 _STATUS3[4:0]) 变为 0x1, 设置 SWTRG (EADC0/1 _SWTRG[18:0]) 为 0x8000 触发模块 15.
5. 重复步骤 3~4 来连续转换.

6.40.5.6 EADC0/1 采样模块转换完成中断操作

EADC0/1 有 4个中断 ADINT0~3, 每个中断有自己的中断向量，可以设置多个模块EOC脉冲 (转换完成脉冲) 作为中断源。图6.40‑8展示中断控制逻辑。选 ADINT0 为例, 当 EADC0/1 IEN0 (EADC0/1 _CTL[2]) = 1 且 SPLIEn (EADC0/1 _INTSRC0[n]) = 1 (n=0~18), 选择模块的EOC (转换完成) 脉冲会设置标志 ADIF0 (EADC0/1 _STATUS[0]) 为 1 然后中断 (ADINT0) 会产生.

中断脉冲 (ADINT0/1)在指定模块 EOC产生时产生，可以作为采样模块转换的触发源。用户使用它实现 EADC0/1 连续转换.

中断触发连续转换的例子如下：

1. 如果EADC0/1 采样模块2的 EOC2作为 ADINT0中断触发源，SPLIE2 (EADC0/1 _INTSRC0[2]) = 1 且 ADINT0作为模块 0, 1, 2转换触发源.
2. 设置软件触发 SWTRG2 (EADC0/1 _SWTRG[2]) 为 1 开始模块 2 转换,转换完成后, 产生转换结束信号和 ADINT0中断脉冲, ADINT0 中断脉冲会触发采样模块 0, 1, 2 开始 EADC0/1 转换。
3. ADINT0 中断脉冲重复触发采样模块0, 1, 2连续EADC0/1 转换.
4. 当需要停在连续转换时，清除 TRGSEL (EADC0/1 _SCTL2[20:16]) 为 0 ，禁止模块 2 的 ADINT0 中断脉冲触发转换

注意：因为系统需要在中断脉冲后消耗3 EADC0/1 _CLK 触发下一个模块, 中断触发连续采样周期是17 EADC0/1 _CLK.

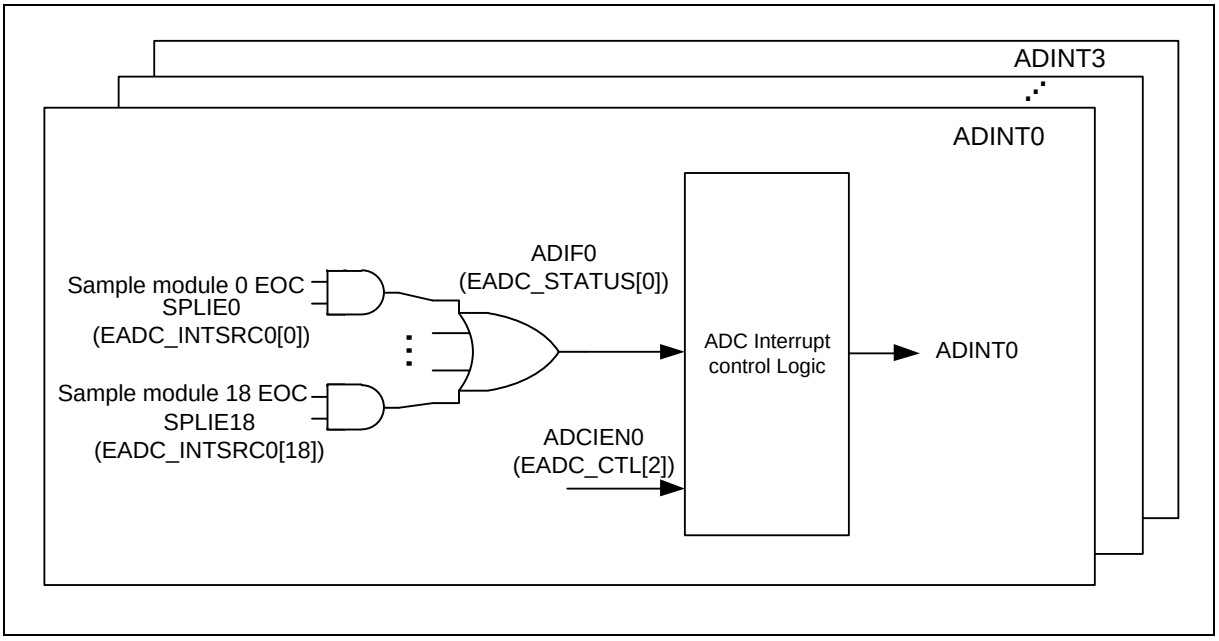


图 6.40-8 选定模块 EADC0/1 EOC 信号 ADINT0~3 中断

6.40.5.7 EADC0/1 由定时器溢出触发或由外部管脚 EADC0/1_ST 触发

有4个定时器和外部管脚EADC0/1_ST可以配置为模块0~15的触发源.

6.40.5.8 EADC0/1 由 EPWM / BPWM 同步触发

除了软件触发, ADINT0/1 中断脉冲, 外部管脚EADC0/1_ST和Timer0~3 溢出脉冲都可以启动 EADC0/1 转换之外, 新的特性允许 EPWM/BPWM 触发启动 EADC0/1 转换. 要配置 EPWM/BPWM 触发类型: EPWM/BPWM 的上升沿，下降沿或 EPWM/BPWM (仅中心对齐模式) 的中点触发EADC0/1 . 以及从触发到 EADC0/1 开始的延时. 通过设置 TRGDLYCNT (EADC0/1 _SCTLn[15:8], n=0~15) 和 TRGDLYDIV (EADC0/1 _SCTLn[7:6], n=8~15)配置延迟时间。延时可配置的EPWM/BPWM触发 EADC0/1 转换时序图见图6.40-9

图6.40-10展示其他触发源配置延时时间

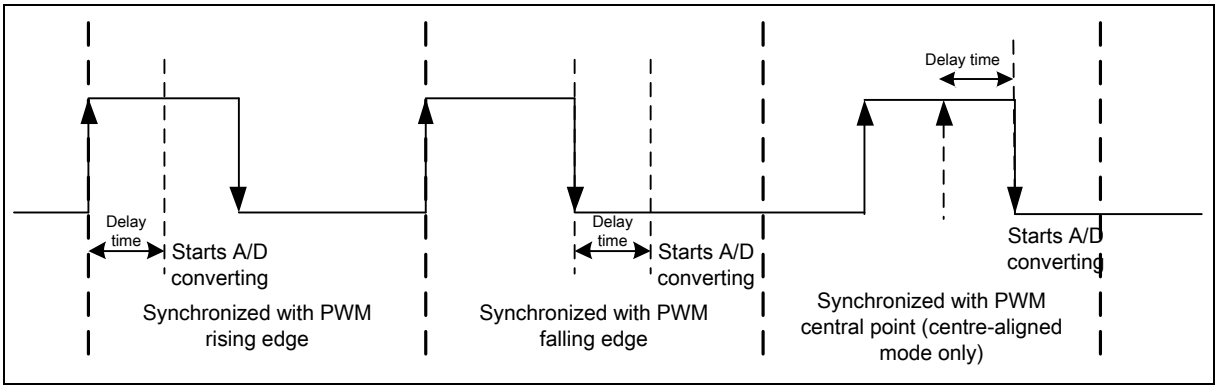


图 6.40-9 EPWM 触发 EADC0/1 开始转换

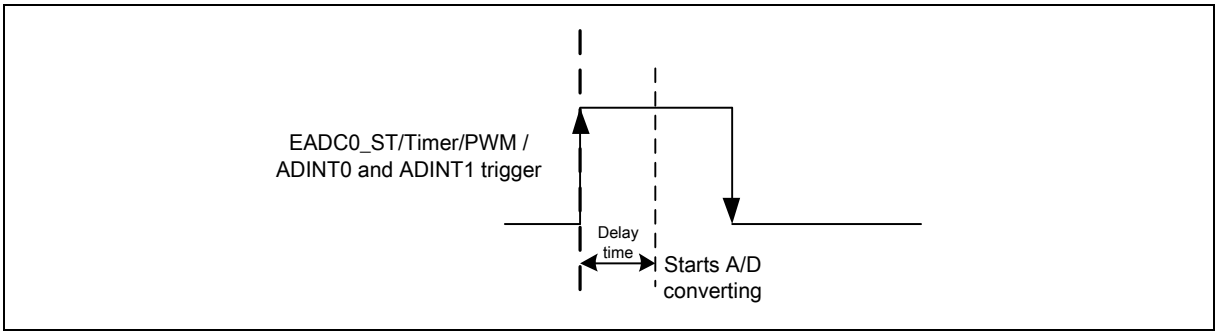


图6.40-10外部触发 EADC0/1 开始转换

6.40.5.9 EADC0/1 转换时间和外部触发

当SWTRGn (EADC0/1 _SWTRG[n], n=0~18)设为1，延时(Td)后采样输入然后开始转换。EADC0/1 时钟由 PCLK 除以 (EADC0/1 (CLKDIV[23:16])+1)得到，从写 SWTRGn 到 EADC0/1 开始采样输入的最小延时是2个 ADC 时钟周期. 延时时间如图6.40-11.

设置 TRGSEL (EADC0/1 _SCTLn[20:16], n=0~15)为 0x01 选择由EADC0/1_ST引脚输入信号触发ADC。可以设置EXTFEN (EADC0/1 _SCTLn[5], n=0~15) 和 EXTREN (EADC0/1 _SCTLn[4], n=0~15)使能引脚EADC0/1_ST触发条件是上升沿还是下降沿。如果选择上升沿触发, 低电平最少保持 2 PCLK 周期且接下来的高电平至少保持 3 PCLK 周期。如果选择下降沿触发, 高电平至少保持 2 PCLK 周期且接下来的下降沿至少保持 3 PCLK 周期. 短于设定的脉冲会被忽略. 外部管脚触发时序见图6.40-12.

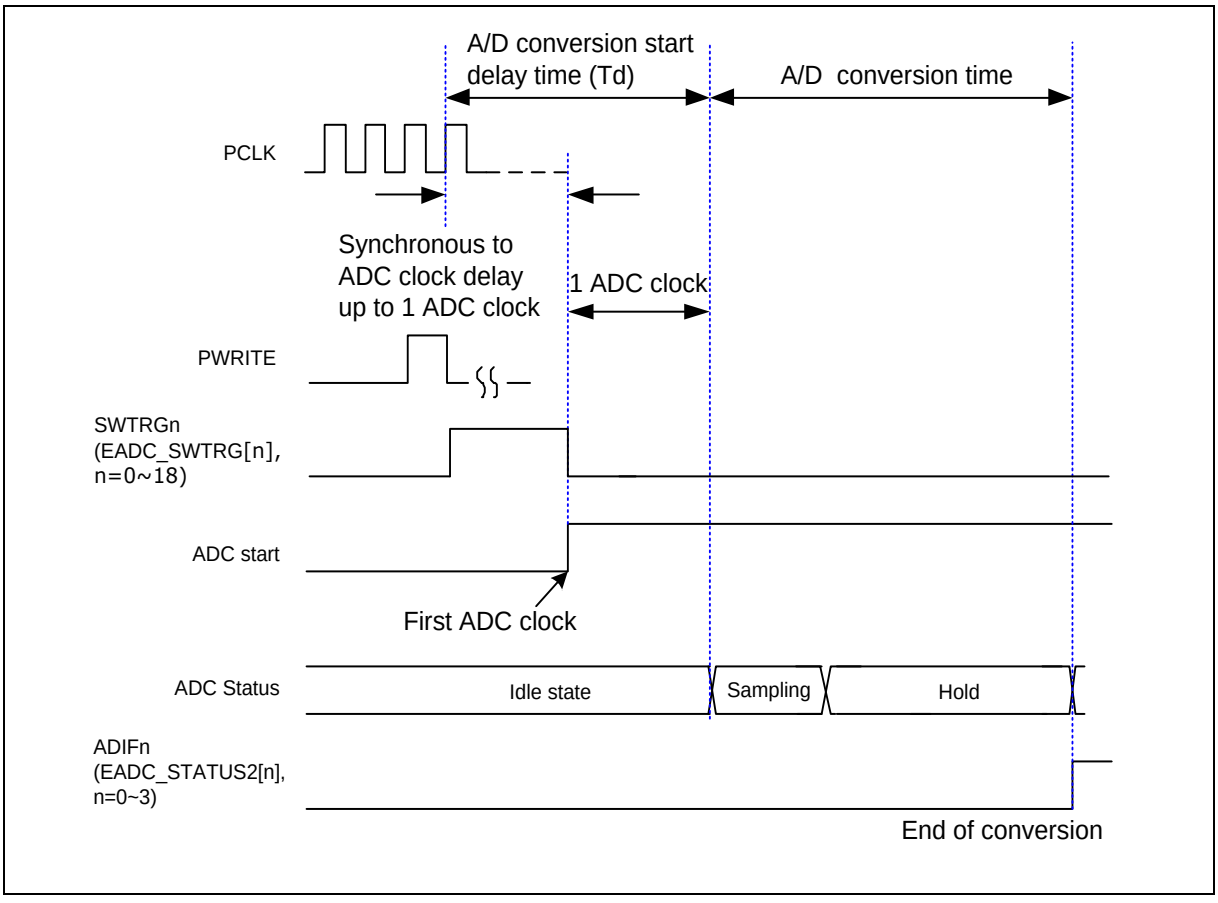


图 6.40-11 转换开始延时时间

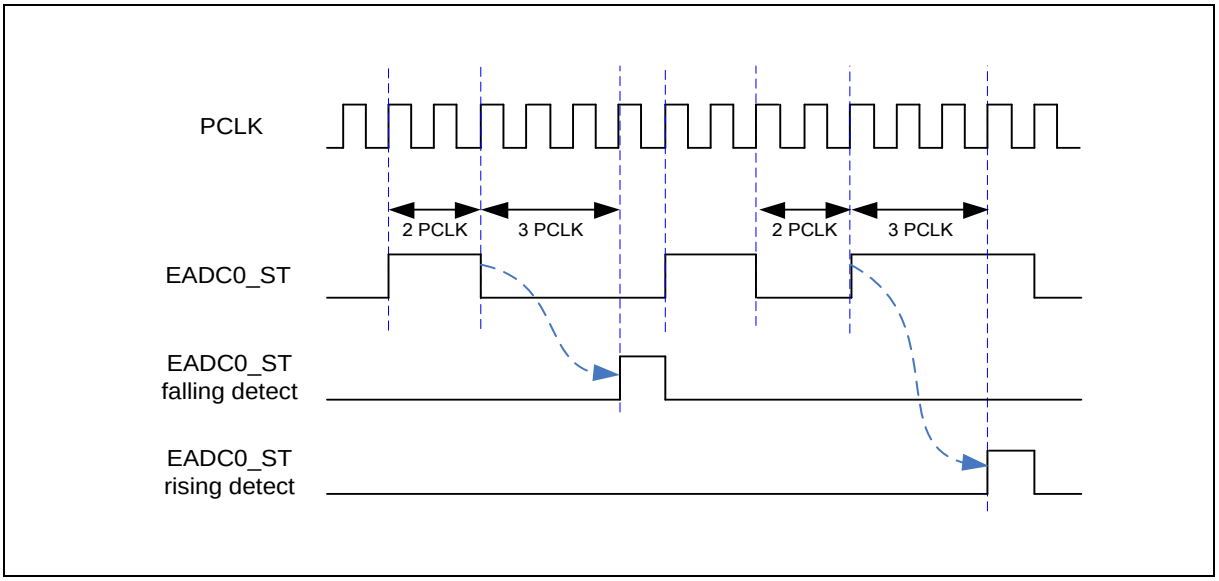
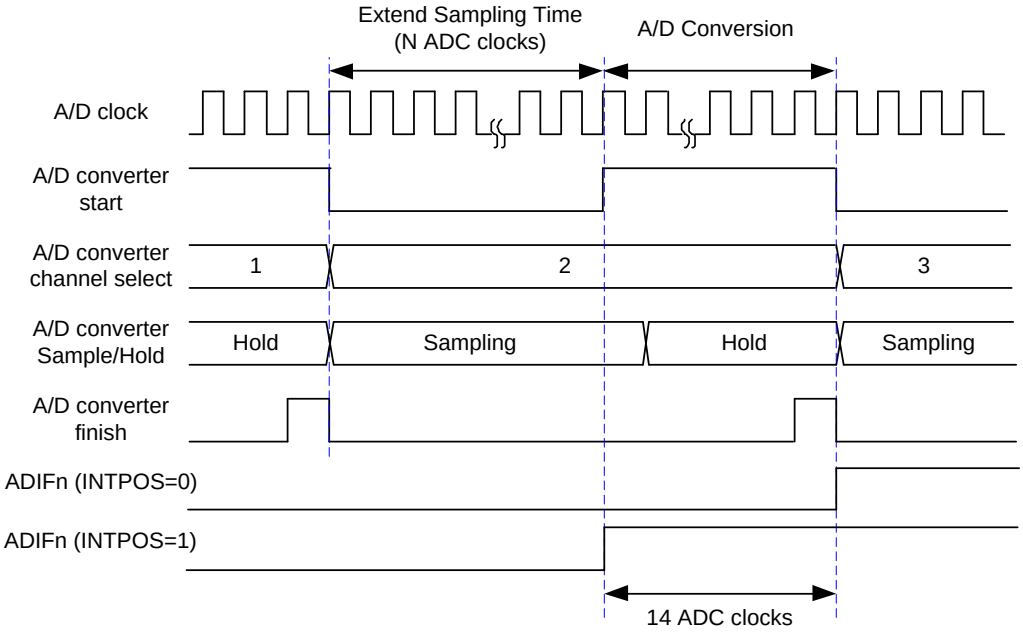


图 6.40-12 EADC0/1_ST 防抖时序

6.40.5.10 ADC延长采样时间

当 ADC 采样率较高时, 如果模拟电路的输出阻抗很高，导致充电时间过长，模拟输入电压的采样时间可能不够。可以对每个采样通道写EXTSMPT (EADC0/1 _SCTLn[31:24], n=0~15) 延长采样时间。延长采样时间的范围是 0 ~255 ADC 时钟。延长采样时间如图6.40-13.



Note: N = EXTSMPT (EADC_SCTLn[31:24], n=0~15) (Extend sampling cycle for each sample module)
ADIFn (EADC_STATUS2[3:0], n=0~3)
INTPOS (EADC_SCTLn[22], n=0~18)

图 6.40-13 ADC 延长采样时间时序

6.40.5.11 转换结果比较模式

有4个比较寄存器EADC0/1 _CMP0 ~ EADC0/1 _CMP3监控模块0~18中4个转换结果，如图6.40-14. 用户设定 CMPSPL (EADC0/1 _CMPn[7:3], n =0~3)选择监控的模块， CMPCOND (EADC0/1 _CMPn[2] , n =0~3) 配置监控条件是结果小于，还是大于等于CMPDAT (EADC0/1 _CMPn[27:16] , n =0~3)设定的值。当 CMPSPL 选择的模块转换完成，就自动比较一次。若比较结果满足比较条件，内部比较符合计数器加1。如果比较结果不满足条件比较符合计数器重置为 0。当计数器的值大到 (CMPMCNT (EADC0/1 _CMPn[11:8])+1, n =0~3)的设定时， ADCMPFn (EADC0/1 _STATUS2[7:4] , n =0~3)位置1, 如果 ADCMPIE (EADC0/1 _CMPn[1] , n =0~3)为1使能中断，那么 ADINT3 中断会产生。用户可以用这个功能监控外部模拟电压变化。详细的逻辑框图如图6.40-14.

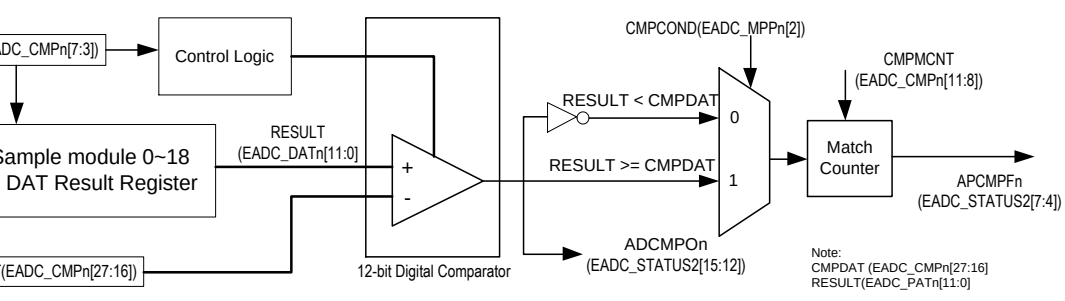


图 6.40-14 ADC 转换结果监控逻辑框图

配置 CMPWEN (EADC0/1 _CMP0[15]/ EADC0/1 _CMP2[15]) 使能窗口比较模式：当 EADC0/1 _CMP0 和 EADC0/1 _CMP1 比较条件都满足时 ADCMPF0 (EADC0/1 _STATUS2[4]) 置 1。当 EADC0/1 _CMP2 和 EADC0/1 _CMP3 比较条件都满足的时候 ADCMPF2 (EADC0/1 _STATUS2[6])置 1。

6.40.5.12 差分模式

EADC控制器支持全差分模式，全差分SAR将差分电压输入转成数据；但是这种情况下两个输入信号是动态改变的，或者互补相反的，因此VINP 和 VINM管脚相位必须以共模电压为中心相差180°，例如VREF/2。全差分EADC输入共模电压(VCM)波动有一个规格，VCM定义为输入信号的平均电压：VCM = (VINP + VINM) / 2，绝大多数全差分输入SAR ADC要求输入共模电压与V_{REF} /2的差别不能超过10%

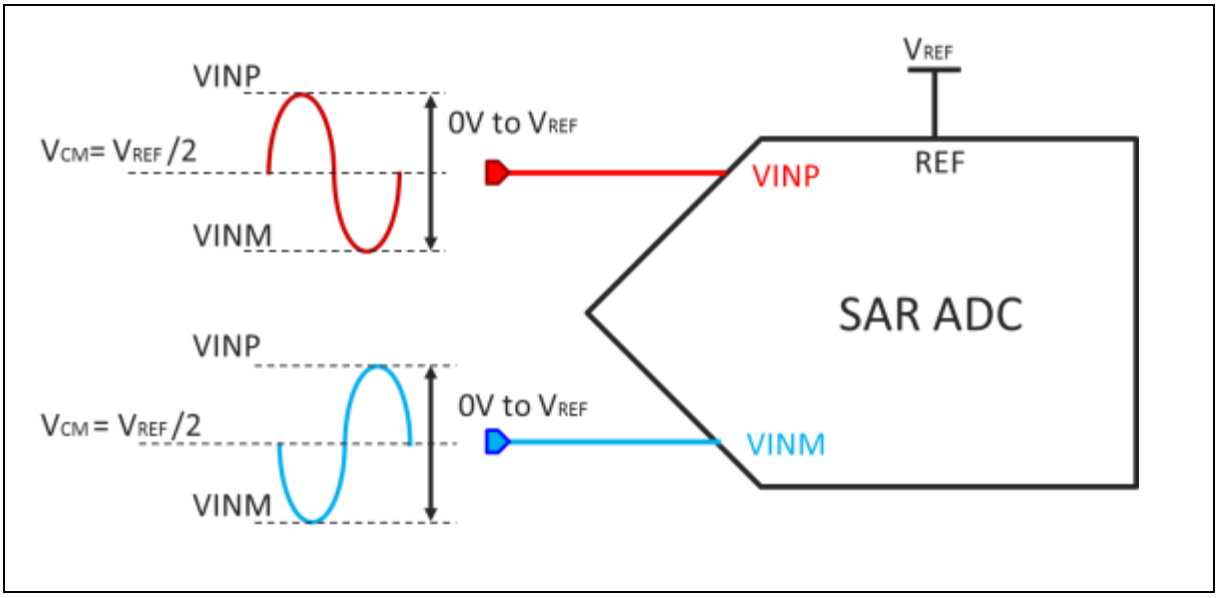


图6.40-16 全差分输入 EADC0/1

如果用户使能了DIFFEN (EADC_CTL[8])，差分模式使能，模拟差分输入通道对如表6.40-2.

差分模拟输入通道对	ADC 模拟输入	
	V_{plus}	V_{minus}
0	EADC_CH0	EADC_CH1
1	EADC_CH2	EADC_CH3
2	EADC_CH4	EADC_CH5
3	EADC_CH6	EADC_CH7
4	EADC_CH8	EADC_CH9
5	EADC_CH10	EADC_CH11
6	EADC_CH12	EADC_CH13
7	EADC_CH14	EADC_CH15

表 6.40-2 EADC0/1 差分模式信道选择

差分输入模式，要在 CHSEL (EADC0/1 _SCTLn[3:0])中使能偶数编号通道。转换结果存放在对应使能的偶数编号通道寄存器中。若DMOF (EADC0/1 _CTL[9]) = 1转换结果是二进制补码形式。

6.40.5.13 双缓存模式

模块0~3支持双数据缓存，使能位在 DBMEN (EADC0/1 _SCTLn[23], n=0~3)，此模式，第一次 ADC 转换结束, VALID (EADC0/1 _DATn[17], n=0~3) 置1, 但 VALID (EADC0/1 _DDATn[17], n=0~3) 保持0。第二次ADC 转换完成后VALID (EADC0/1 _DDATn[17], n=0~3) 置1，用户可以在 EADC0/1 _DATn 和 EADC0/1 _DDATn 寄存器得到两次转换结果。

6.40.5.14 同步模式

EADC0/1包含EADC0和EADC1，用户可以选择同一个触发源触发两个EADC，有EPWM0/1, BPWM0/1 及 timer0~3可被选作共同触发源。例如，用户使能EADC0采样模块0和EADC1采样模块1，共享定时器触发源，EADC0,EADC1,将在收到触发信号后同时开始转换，注意如果其中一个EADC在收到定时器触发时正处于数据转换状态，则无法同时完成同步转换

注意:在同步模式下，ADC时钟必须等于PCLK

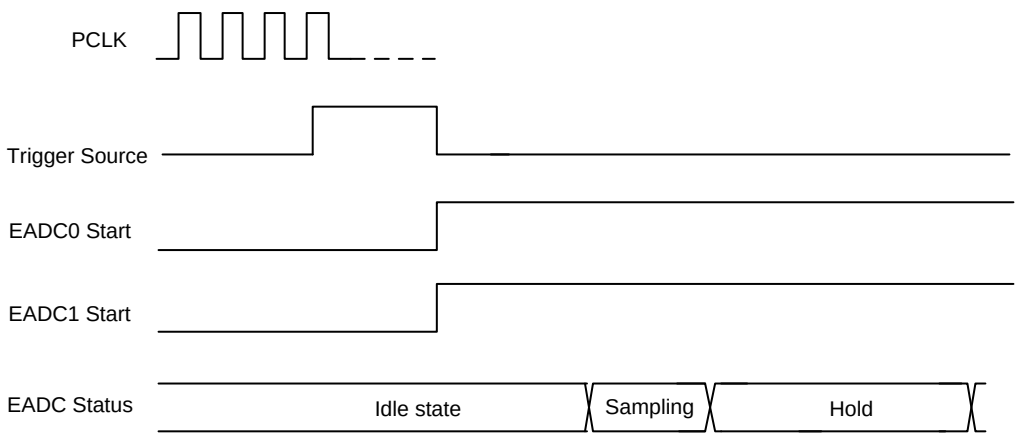


图 6.40-15 EADC0/1 同步触发

6.40.5.15 PDMA 请求

EADC0/1控制器支持PDMA，使用PDMA传输ADC结果，要使能 PDMAEN (EADC0/1 _CTL[11]) 且配置 PDMA 信道的源地址是EADC0/1 _CURDAT (EADC0/1 _BA+0x4C)。任意 VALID (EADC0/1 _DATn[17],n=0~18) 为1, ADC 都会发出PDMA请求，PDMA会读取EADC0/1_CURDAT获得ADC数值。 EADC0/1 _CURDAT 寄存器是 EADC0/1 _DAT 寄存器的影子寄存器。编号较低的模块有更高的优先级。 PDMA 读EADC0/1 _CURDAT寄存器之后，相应EADC0/1 _DAT寄存器的 VAILD 位自动清除.

6.40.5.16 中断源

在转换开始或转换完成产生 ADIFn (EADC0/1 _STATUS2[3:0], n=0~3) 由 INTPOS (EADC0/1 _SCTLn[22], n=0~15)决定。如果 ADCIENn (EADC0/1 _CTL[5:2], n=0~3) 为1，转换完成产生中断请求 ADINTn (n=0~3) 。中断控制器如图6.40-16。

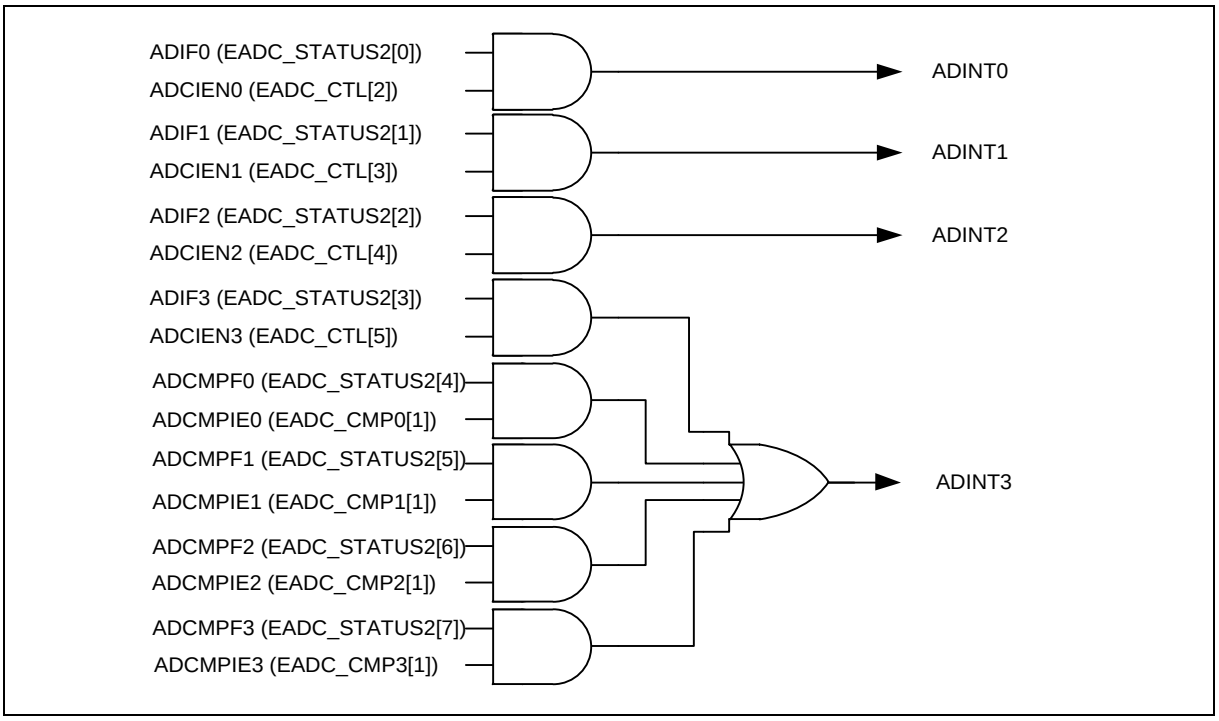


图 6.40-16 ADC 控制器中断

6.40.5.17 电源管理和校准

有3种省电模式：深度掉电模式，掉电模式和待机模式。要在ADCEN (EADC0/1 _CTL[0])=0时设定PWDMOD (EADC0/1 _PWRM[3:2]) 选择模式。3种模式的不同如表6.40-3. 因为内部 LDO 会在深度掉电和掉电模时关闭，再次使能需要较长时间恢复。恢复时间由 LDOSUT (EADC0/1 _PWRM[19:8]) 设定，需要长于 20 us。在待机模式, LDO 保持使能，不需要恢复时间.

电源供应	深度掉电	掉电	待机
内部 LDO	禁止	禁止	使能
内部电源开关	禁止	使能	使能

表 6.40-3 EADC0/1 省电模式

当ADCEN(EADC0/1 _CTL[0]) 为 1设定EADC0/1 有效, 便执行开始序列。开始序列完成后, PWUPRDY (EADC0/1 _PWRM[0]) 由硬件置 1 意味着准备好转换。在开始序列中，ADCEN (EADC0/1 _CTL[0]) 需要保持为 1 直到 PWUPRDY (EADC0/1 _PWRM[0]) 设为 1。在开始序列中修改 ADCEN (EADC0/1 _CTL[0]) 会导致 EADC0/1 功能失败。

ADC 的转换结果可以通过校准来更加精确。用户可以设置 PWUCALEN (EADC0/1 _PWRM[1]) 为 1 在启动时执行校准。这一位需要与 CALSEL (EADC0/1 _CALCTL [3])配套设置, {PWUCALEN, CALSEL} 配置如表6.40-4. 启动校准的例子如图6.40-17.

PWUCALEN	CALSEL	配置
0	0	无校准启动
0	1	无校准启动
1	0	启动时载入校准值
1	1	启动校准

表 6.40-4 EADC0/1 校准启动

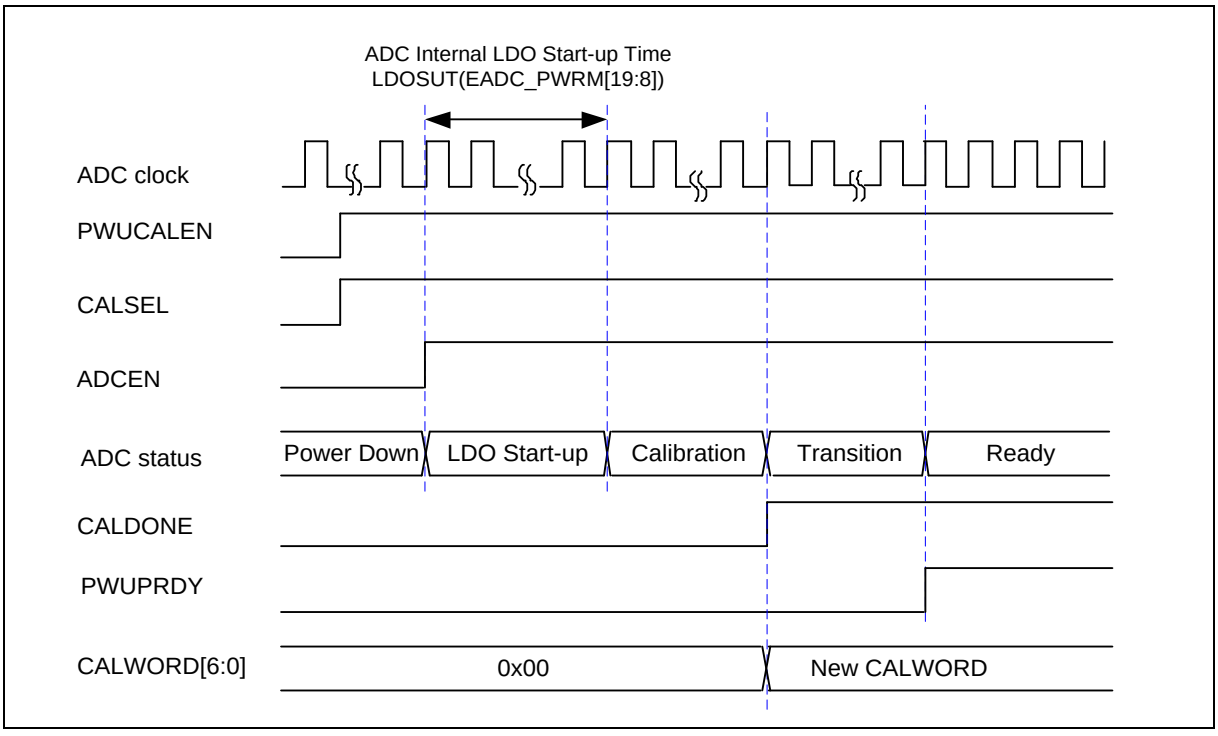


图 6.40-17 ADC 带校准的启动序列

为了得到更精密的结果，用户可以在数次转换之后再次校准。设置 CALSTART (EADC0/1_CALCTL[1]) 为 1 使能再次校准，这一位需要与 CALSEL (EADC0/1_CALCTL[3])同时操作。设置 CALSTART (EADC0/1_CALCTL[1]) 为 1 且 CALSEL (EADC0/1_CALCTL[3]) 为 1 会执行再次校准，然后更新 CALWORD (EADC0/1_CALDWRD[6:0])。设置 CALSTART (EADC0/1_CALCTL[1]) 为 1 且 CALSEL (EADC0/1_CALCTL[3]) 为 0 会载入用户定义的 CALWORD (EADC0/1_CALDWRD[6:0]) 。重新校准的流程如下所示：

1. 设置CALSEL (EADC0/1 _CALCTL[3]) 为 1或0 选择校准功能
2. 设置 CALSTART (EADC0/1 _CALCTL[1]) 为 1 来使能校准
3. CALDONE (EADC0/1 _CALCTL[2]) 会在校准中由硬件设为 0
4. 等待 CALDONE (EADC0/1 _CALCTL[2]) 由硬件设为 1。如果 CALSEL (EADC0/1 _CALCTL[3]) 设为 1, 新的校准值会更新到 CALWORD (EADC0/1 _CALDWRD[6:0])。如果 CALSEL (EADC0/1 _CALCTL[3])设为0, 设定的 CALWORD (EADC0/1 _CALDWRD[6:0]) 会载入而不需要执行校准。

注: 注意：在将CALSTART（EADC0/1 _CALCTL [1]）设置为1以再次开始校准之前，EADC0/1 DIV（CKL_CLKDIV0 [23:16]）必须为0。

6.40.6 寄存器映射

R: 只读, **W**: 只写, **R/W**: 可读可写

寄存器	偏移量	R/W	描述	复位值
EADC 基地址: **EADCx_BA = 0x4004_3000 + (0x8000 * x)** **x=0,1**				
EADCx_DAT0 **x=0,1**	EADCx_BA+0x00	R	EADC 采样模块 0 数据寄存器 0	0x0000_0000
EADCx_DAT1 **x=0,1**	EADCx_BA+0x04	R	EADC采样模块 1 数据寄存器 1	0x0000_0000
EADCx_DAT2 **x=0,1**	EADCx_BA+0x08	R	EADC 采样模块 2 数据寄存器 2	0x0000_0000
EADCx_DAT3 **x=0,1**	EADCx_BA+0x0C	R	EADC 采样模块 3 数据寄存器 3	0x0000_0000
EADCx_DAT4 **x=0,1**	EADCx_BA+0x10	R	EADC 采样模块 4 数据寄存器 4	0x0000_0000
EADCx_DAT5 **x=0,1**	EADCx _BA+0x14	R	EADC 采样模块 5 数据寄存器 5	0x0000_0000
EADCx_DAT6 **x=0,1**	EADCx _BA+0x18	R	EADC 采样模块 6 数据寄存器 6	0x0000_0000
EADCx_DAT7 **x=0,1**	EADCx _BA+0x1C	R	EADC 采样模块 7 数据寄存器 7	0x0000_0000
EADCx_DAT8 **x=0,1**	EADCx_BA+0x20	R	EADC 采样模块 8 数据寄存器 8	0x0000_0000
EADCx_DAT9 **x=0,1**	EADCx_BA+0x24	R	EADC 采样模块 9 数据寄存器 9	0x0000_0000
EADCx_DAT10 **x=0,1**	EADCx_BA+0x28	R	EADC 采样模块 10 数据寄存器 10	0x0000_0000
EADCx_DAT11 **x=0,1**	EADCx_BA+0x2C	R	EADC 采样模块 11 数据寄存器 11	0x0000_0000
EADCx_DAT12 **x=0,1**	EADCx_BA+0x30	R	EADC 采样模块 12 数据寄存器 12	0x0000_0000
EADCx_DAT13 **x=0,1**	EADCx_BA+0x34	R	EADC 采样模块 13 数据寄存器 13	0x0000_0000
EADCx_DAT14 **x=0,1**	EADCx_BA+0x38	R	EADC 采样模块 14 数据寄存器 14	0x0000_0000
EADCx_DAT15 **x=0,1**	EADCx_BA+0x3C	R	EADC 采样模块 15 数据寄存器 15	0x0000_0000

寄存器	偏移量	R/W	描述	复位值
EADC 基地址: **EADCx_BA = 0x4004_3000 + (0x8000 * x)** **x=0,1**				
EADCx_DAT16 **x=0,1**	EADCx_BA+0x40	R	EADC 采样模块 16 数据寄存器 16	0x0000_0000
EADCx_DAT17 **x=0,1**	EADCx_BA+0x44	R	EADC 采样模块 17 数据寄存器 17	0x0000_0000
EADCx_DAT18 **x=0,1**	EADCx_BA+0x48	R	EADC 采样模块 18 数据寄存器 18	0x0000_0000
EADCx_CURDAT **x=0,1**	EADCx_BA+0x4C	R	EADC PDMA 目前传输数据寄存器	0x0000_0000
EADCx_CTL **x=0,1**	EADCx_BA+0x50	R/W	EADC 控制寄存器	0x0000_00C0
EADCx_SWTRG **x=0,1**	EADCx_BA+0x54	W	EADC 采样模块软件触发寄存器	0x0000_0000
EADCx_PENDSTS **x=0,1**	EADCx_BA+0x58	R/W	EADC 开始转换等待标志寄存器	0x0000_0000
EADCx_OVSTS **x=0,1**	EADCx_BA+0x5C	R/W	EADC 采样模块开始转换覆盖标志寄存器	0x0000_0000
EADCx_SCTL0 **x=0,1**	EADCx_BA+0x80	R/W	EADC 采样模块 0 控制寄存器	0x0000_0000
EADCx_SCTL1 **x=0,1**	EADCx_BA+0x84	R/W	EADC 采样模块 1 控制寄存器	0x0000_0000
EADCx_SCTL2 **x=0,1**	EADCx_BA+0x88	R/W	EADC 采样模块 2 控制寄存器	0x0000_0000
EADCx_SCTL3 **x=0,1**	EADCx_BA+0x8C	R/W	EADC 采样模块 3 控制寄存器	0x0000_0000
EADCx_SCTL4 **x=0,1**	EADCx_BA+0x90	R/W	EADC 采样模块 4 控制寄存器	0x0000_0000
EADCx_SCTL5 **x=0,1**	EADCx_BA+0x94	R/W	EADC 采样模块 5 控制寄存器	0x0000_0000
EADCx_SCTL6 **x=0,1**	EADCx_BA+0x98	R/W	EADC 采样模块 6 控制寄存器	0x0000_0000
EADCx_SCTL7 **x=0,1**	EADCx_BA+0x9C	R/W	EADC 采样模块 7 控制寄存器	0x0000_0000
EADCx_SCTL8 **x=0,1**	EADCx_BA+0xA0	R/W	EADC 采样模块 8 控制寄存器	0x0000_0000
EADCx_SCTL9 **x=0,1**	EADCx_BA+0xA4	R/W	EADC 采样模块 9 控制寄存器	0x0000_0000
EADCx_SCTL10	EADCx_BA+0xA8	R/W	EADC 采样模块 10 控制寄存器	0x0000_0000

寄存器	偏移量	R/W	描述	复位值
EADC 基地址: **EADCx_BA = 0x4004_3000 + (0x8000 * x)** **x=0,1**				
x=0,1				
EADCx_SCTL11	EADCx_BA+0xAC	R/W	EADC 采样模块 11 控制寄存器	0x0000_0000
EADCx_SCTL12 **x=0,1**	EADCx_BA+0xB0	R/W	EADC 采样模块 12 控制寄存器	0x0000_0000
EADCx_SCTL13 **x=0,1**	EADCx_BA+0xB4	R/W	EADC 采样模块 13 控制寄存器	0x0000_0000
EADCx_SCTL14 **x=0,1**	EADCx_BA+0xB8	R/W	EADC 采样模块 14 控制寄存器	0x0000_0000
EADCx_SCTL15 **x=0,1**	EADCx_BA+0xBC	R/W	EADC 采样模块 15 控制寄存器	0x0000_0000
EADCx_SCTL16 **x=0,1**	EADCx_BA+0xC0	R/W	EADC 采样模块 16 控制寄存器	0x0000_0000
EADCx_SCTL17 **x=0,1**	EADCx_BA+0xC4	R/W	EADC 采样模块 17 控制寄存器	0x0000_0000
EADCx_SCTL18 **x=0,1**	EADCx_BA+0xC8	R/W	EADC 采样模块 18 控制寄存器	0x0000_0000
EADCx_INTSRC0 **x=0,1**	EADCx_BA+0xD0	R/W	EADC中断 0 源使能控制寄存器.	0x0000_0000
EADCx_INTSRC1 **x=0,1**	EADCx_BA+0xD4	R/W	EADC中断 1 源使能控制寄存器.	0x0000_0000
EADCx_INTSRC2 **x=0,1**	EADCx_BA+0xD8	R/W	EADC中断 2 源使能控制寄存器.	0x0000_0000
EADCx_INTSRC3 **x=0,1**	EADCx_BA+0xDC	R/W	EADC中断 3 源使能控制寄存器.	0x0000_0000
EADCx_CMP0 **x=0,1**	EADCx_BA+0xE0	R/W	EADC 结果比较寄存器 0	0x0000_0000
EADCx_CMP1 **x=0,1**	EADCx_BA+0xE4	R/W	EADC 结果比较寄存器 1	0x0000_0000
EADCx_CMP2 **x=0,1**	EADCx_BA+0xE8	R/W	EADC 结果比较寄存器 2	0x0000_0000
EADCx_CMP3 **x=0,1**	EADCx_BA+0xEC	R/W	EADC 结果比较寄存器 3	0x0000_0000
EADCx_STATUS0 **x=0,1**	EADCx_BA+0xF0	R	EADC 状态寄存器 0	0x0000_0000
EADCx_STATUS1 **x=0,1**	EADCx_BA+0xF4	R	EADC 状态寄存器 1	0x0000_0000

寄存器	偏移量	R/W	描述	复位值
EADC 基地址: **EADCx_BA = 0x4004_3000 + (0x8000 * x)** **x=0,1**				
EADCx_STATUS2 **x=0,1**	EADCx_BA+0xF8	R/W	EADC 状态寄存器 2	0x0011_0000
EADCx_STATUS3 **x=0,1**	EADCx_BA+0xFC	R	EADC 状态寄存器 3	0x0000_001F
EADCx_DDAT0 **x=0,1**	EADCx_BA+0x100	R	EADC 采样模块 0 双数据寄存器 0	0x0000_0000
EADCx_DDAT1 **x=0,1**	EADCx_BA+0x104	R	EADC 采样模块 1 双数据寄存器 1	0x0000_0000
EADCx_DDAT2 **x=0,1**	EADCx_BA+0x108	R	EADC 采样模块 2 双数据寄存器 2	0x0000_0000
EADCx_DDAT3 **x=0,1**	EADCx_BA+0x10C	R	EADC 采样模块 3 双数据寄存器 3	0x0000_0000
EADCx_PWRM **x=0,1**	EADCx_BA+0x110	R/W	EADC 电源管理寄存器	0x0006_E012
EADCx_CALCTL **x=0,1**	EADCx_BA+0x114	R/W	EADC 校准控制寄存器	0x0000_0008
EADCx_CALDWRD **x=0,1**	EADCx_BA+0x118	R/W	EADC 校准载入值寄存器	0x0000_00XX
EADCx_PDMACTL **x=0,1**	EADCx_BA+0x130	R/W	EADC PDMA控制寄存器	0x0000_0000

6.40.7 寄存器描述

EADC_DAT0~18 EADC 数据寄存器

寄存器	偏移量	R/W	描述	复位值
EADCx_DAT0 x=0,1	EADCx_BA+0x00	R	EADC 采样模块 0 数据寄存器	0x0000_0000
EADCx_DAT1 x=0,1	EADCx_BA+0x04	R	EADC 采样模块 1 数据寄存器	0x0000_0000
EADCx_DAT2 x=0,1	EADCx_BA+0x08	R	EADC 采样模块 2 数据寄存器	0x0000_0000
EADCx_DAT3 x=0,1	EADCx_BA+0x0C	R	EADC 采样模块 3 数据寄存器	0x0000_0000
EADCx_DAT4 x=0,1	EADCx_BA+0x10	R	EADC 采样模块 4 数据寄存器	0x0000_0000
EADCx_DAT5 x=0,1	EADCx_BA+0x14	R	EADC 采样模块 5 数据寄存器	0x0000_0000
EADCx_DAT6 x=0,1	EADCx_BA+0x18	R	EADC 采样模块 6 数据寄存器	0x0000_0000
EADCx_DAT7 x=0,1	EADCx_BA+0x1C	R	EADC 采样模块 7 数据寄存器	0x0000_0000
EADCx_DAT8 x=0,1	EADCx_BA+0x20	R	EADC 采样模块 8 数据寄存器	0x0000_0000
EADCx_DAT9 x=0,1	EADCx_BA+0x24	R	EADC 采样模块 9 数据寄存器	0x0000_0000
EADCx_DAT10 x=0,1	EADCx_BA+0x28	R	EADC 采样模块 10 数据寄存器	0x0000_0000
EADCx_DAT11 x=0,1	EADCx_BA+0x2C	R	EADC 采样模块 11 数据寄存器	0x0000_0000
EADCx_DAT12 x=0,1	EADCx_BA+0x30	R	EADC 采样模块 12 数据寄存器	0x0000_0000
EADCx_DAT13 x=0,1	EADCx_BA+0x34	R	EADC 采样模块 13 数据寄存器	0x0000_0000
EADCx_DAT14 x=0,1	EADCx_BA+0x38	R	EADC 采样模块 14 数据寄存器	0x0000_0000
EADCx_DAT15 x=0,1	EADCx_BA+0x3C	R	EADC 采样模块 15 数据寄存器	0x0000_0000
EADCx_DAT16	EADCx_BA+0x40	R	EADC 采样模块 16 数据寄存器	0x0000_0000

x=0,1				
EADCx_DAT17 **x=0,1**	EADCx_BA+0x44	R	EADC 采样模块 17 数据寄存器	0x0000_0000
EADCx_DAT18 **x=0,1**	EADCx_BA+0x48	R	EADC 采样模块 18 数据寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved						VALID	OV
15	14	13	12	11	10	9	8
RESULT							
7	6	5	4	3	2	1	0
RESULT							

位	描述	
[31:18]	**Reserved**	保留.
[17]	**VALID**	**有效标志** 转换完成后置 1，EADC_DAT被读取后硬件清0 0 = RESULT[11:0]数据无效. 1 = RESULT[11:0] 数据有效.
[16]	**OV**	**覆盖标志** RESULT[11:0] 里的转换结果在被读出前，又有新的转换结果存入， OV会设为 1. 0 = RESULT[11:0]的结果是最近的转换结果. 1 = RESULT[11:0] 被覆盖. **注意:** EADC_DAT寄存器被读取后此位被硬件清0.
[15:0]	**RESULT**	**EADC 转换结果** 这里是 12 位的转换结果. 当 DMOF (EADC_CTL[9])设为 0, 12-位 EADC 转换结果是 RESULT[11:0] 的无符号数，RESULT[15:12]会填0. 当 DMOF (EADC_CTL[9])设为 1, 12-位EADC 转换结果是RESULT[11:0] 的有符号二进制补码， RESULT[15:12]填符号位.

EADC_CURDAT EADC PDMA 目前传输数据寄存器

寄存器	偏移量	R/W	描述	复位值
EADCx_CURDAT x=0,1	EADCx_BA+0x4C	R	EADC PDMA目前传输数据寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved						**CURDAT**	
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
CURDAT							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
CURDAT							

位	描述	
[31:18]	**Reserved**	保留.
[17:0]	**CURDAT**	**EADC PDMA 目前传输数据寄存器（只读）** 这个寄存器是 EADC_DATn (n=0~18) 的影子寄存器，为了支持 PDMA .

EADC_CTL EADC 控制寄存器

寄存器	偏移量	R/W	描述	复位值
EADCx_CTL x=0,1	EADCx_BA+0x50	R/W	EADC控制寄存器	0x0000_00C0

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved						**DMOF**	**DIFFEN**
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
RESSEL		**ADCIEN3**	**ADCIEN2**	**ADCIEN1**	**ADCIEN0**	**ADCRST**	**ADCEN**

位	描述	
[31:10]	**Reserved**	保留.
[9]	**DMOF**	**差分模式输出格式** 0 = EADC 转换结果是无符号数的格式. 1 = EADC转换结果是二进制补码的格式
[8]	**DIFFEN**	**差分模式使能位** 0 = 单端模拟输入模式. 1 = 差分模拟输入模式.
[7:6]	**RESSEL**	**EADC 结果位数选择** 00 = 6位 EADC 结果存入 RESULT (EADC_DATn[5:0]). 01 = 8位 EADC 结果存入 RESULT (EADC_DATn[7:0]). 10 = 10位 EADC结果存入RESULT (EADC_DATn[9:0]). 11 = 12位 EADC 结果存入RESULT (EADC_DATn[11:0]).
[5]	**ADCIEN3**	**EADINT3中断使能位** 选定的通道转换完成后，完成标志EADIF3 (EADC_STATUS2[3]) 置1，如果EADCIEN3 为1会产生转换完成中断EADINT3 0 = EADINT3 中断禁止 1 = EADINT3中断使能.
[4]	**ADCIEN2**	**EADINT2中断使能位** 选定的通道转换完成后，完成标志EADIF2 (EADC_STATUS2[2]) 置1，如果EADCIEN2为1会产生转换完成中断EADINT2 0 = EADINT2 中断禁止 1 = EADINT2中断使能.

位	描述	
[3]	**ADCIEN1**	**EADINT1中断使能位** 选定的通道转换完成后，完成标志EADIF1 (EADC_STATUS2[1]) 置1，如果EADCIEN1为1会产生转换完成中断EADINT1 0 = EADINT1 中断禁止 1 = EADINT1中断使能
[2]	**ADCIEN0**	**EADINT0中断使能位** 选定的通道转换完成后，完成标志EADIF0 (EADC_STATUS2[0]) 置1，如果EADCIEN0 为1会产生转换完成中断EADINT0 0 = EADINT0 中断禁止 1 = EADINT0中断使能
[1]	**ADCRST**	**EADC 复位** 0 = 没影响. 1 = EADC 复位到初始态, 但不会改变 EADC 寄存器的值. **注意: E**ADCRST 在 EADC 复位时保持为1, 当 EADC复位结束, EADCRST位会自动清 0.
[0]	**ADCEN**	**EADC使能位** 0 = 禁止 EADC. 1 = 使能 EADC. **注意:**开始转换前, 这一位要设为 1。此位清0 EADC 模拟电路断电节省功耗.

EADC_SWTRG EADC 软件启动寄存器

寄存器	偏移量	R/W	描述	复位值
EADCx_SWTRG x=0,1	EADCx_BA+0x54	W	EADC采样模块软件启动寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved					**SWTRG**		
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
SWTRG							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
SWTRG							

位	描述	
[31:19]	**Reserved**	保留.
[18:0]	**SWTRG**	**采样模块 0~18 软件启动** 0 = 没影响. 1 = 当该模块获得优先级时，启动 EADC 转换 **注意：**写这个寄存器启动 EADC转换后， EADC_PENDSTS 寄存器显示哪一个通道会被转换。如果用户想要禁止采样模块转换，可以写EADC_PENDSTS 寄存器清除.

EADC_PENDSTS EADC模块转换等待标志寄存器

寄存器	偏移量	R/W	描述	复位值
EADCx_PENDSTS x=0,1	EADCx_BA+0x58	R/W	EADC开始转换等待标志寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved					STPF		
15	14	13	12	11	10	9	8
STPF							
7	6	5	4	3	2	1	0
STPF							

位	描述	
[31:19]	**Reserved**	保留.
[18:0]	**STPF**	**EADC 模块 0~18 转换等待标志** **读:** 0 = 模块没有等待转换. 1 = 模块在等待转换. **写:** 1 = 清等待标志且取消采样模块转换. **注意:**在等待转换期间对应位为1，当对应的 EADC 转换完成自动清 0.

EADC_OVSTS EADC 模块覆盖标志寄存器

寄存器	偏移量	R/W	描述	复位值
EADCx_OVSTS x=0,1	EADCx_BA+0x5C	R/W	EADC 模块开始转换覆盖标志寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved					**SPOVF**		
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
SPOVF							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
SPOVF							

位	描述	
[31:19]	**Reserved**	保留.
[18:0]	**SPOVF**	**EADC 模块0~18 覆盖标志** 0 = 没有覆盖事件. 1 = 新的事件已产生而旧的事件还在等待 **注意:**此位写1清0.

EADC_SCTL0~3 EADC 模块 0~3 控制寄存器

寄存器	偏移量	R/W	描述	复位值
EADCx_SCTL0 x=0,1	EADCx_BA+0x80	R/W	EADC 模块 0 控制寄存器	0x0000_0000
EADCx_SCTL1 x=0,1	EADCx_BA+0x84	R/W	EADC 模块 1 控制寄存器	0x0000_0000
EADCx_SCTL2 x=0,1	EADCx_BA+0x88	R/W	EADC 模块 2 控制寄存器	0x0000_0000
EADCx_SCTL3 x=0,1	EADCx_BA+0x8C	R/W	EADC 模块 3 控制寄存器	0x0000_0000

31	30	29	28	27	26	25	24
EXTSMPT							
23	22	21	20	19	18	17	16
DBMEN	INTPOS	Reserved	TRGSEL				
15	14	13	12	11	10	9	8
TRGDLYCNT							
7	6	5	4	3	2	1	0
TRGDLYDIV		EXTFEN	EXTREN	CHSEL			

位	描述	
[31:24]	**EXTSMPT**	**EADC 采样时间加长** 当 EADC转换速率较高，并且外部模拟电路的输出阻抗较高时，采样时间短，可能采样保持电容还没充完电就开始转换，EADC结果会偏小。这种情况可以加长采样时间。 延时范围从 0~255 EADC时钟.
[23]	**DBMEN**	**双缓存模式使能位** 0 = 只有一个结果寄存器 (默认). 1 = 有2个结果寄存器.
[22]	**INTPOS**	**中断标志位置选择** 0 = 在EADC转换结束设置中断标志 ADIFn (EADC_STATUS2[n], n=0~3). 1 = 在EADC转换开始设置中断标志ADIFn (EADC_STATUS2[n], n=0~3)
[21]	**Reserved**	保留.

位	描述	
[20:16]	**TRGSEL**	**EADC 模块转换触发源选择** 0H = 禁止触发. 1H = EADC0_ST外部管脚触发. 2H = EADC ADINT0 中断 EOC (转换完成) 脉冲触发. 3H = EADC ADINT1 中断 EOC (转换完成) 脉冲触发. 4H = Timer0 溢出脉冲触发. 5H = Timer1 溢出脉冲触发. 6H = Timer2溢出脉冲触发. 7H = Timer3溢出脉冲触发. 8H = EPWM0TG0. 9H = EPWM0TG1. AH = EPWM0TG2. BH = EPWM0TG3. CH = EPWM0TG4. DH = EPWM0TG5. EH = EPWM1TG0. FH = EPWM1TG1. 10H = EPWM1TG2. 11H = EPWM1TG3. 12H = EPWM1TG4. 13H = EPWM1TG5. 14H = BPWM0TG. 15H = BPWM1TG. 其他 = 保留.
[15:8]	**TRGDLYCNT**	**EADC 模块转换触发延迟时间** 触发延时 = TRGDLYCNT x EADC_CLK x n (n=1,2,4,16 由 TRGDLYDIV设定).
[7:6]	**TRGDLYDIV**	**EADC 模块转换触发延时除频选择** 延时时钟频率: 00 = EADC_CLK/1. 01 = EADC_CLK/2. 10 = EADC_CLK/4. 11 = EADC_CLK/16.
[5]	**EXTFEN**	**EADC 外部触发下降沿使能位** 0 = 当 EADC 选择 EADC0_ST 作为触发源，禁止下降沿触发. 1 = 当 EADC 选择 EADC0_ST 作为触发源，使能下降沿触发.
[4]	**EXTREN**	**EADC 外部触发上升沿使能位** 0 = 当 EADC 选择 EADC0_ST 作为触发源，禁止上升沿触发. 1 = 当 EADC 选择 EADC0_ST 作为触发源，使能上升沿触发.

位	描述	
[3:0]	**CHSEL**	**EADC 模块通道选择** 00H = EADC_CH0 (慢速通道). 01H = EADC_CH1 (慢速通道). 02H = EADC_CH2 (慢速通道). 03H = EADC_CH3 (慢速通道). 04H = EADC_CH4 (慢速通道). 05H = EADC_CH5 (慢速通道). 06H = EADC_CH6 (慢速通道). 07H = EADC_CH7 (慢速通道). 08H = EADC_CH8 (慢速通道). 09H = EADC_CH9 (慢速通道). 0AH = EADC_CH10 (快速通道). 0BH = EADC_CH11 (快速通道). 0CH = EADC_CH12 (快速通道). 0DH = EADC_CH13 (快速通道). 0EH = EADC_CH14 (快速通道). 0FH = EADC_CH15 (快速通道).

EADC_SCTL4~15 EADC 模块 4~15 控制寄存器

寄存器	偏移量	R/W	描述	复位值
EADCx_SCTL4 x=0,1	EADCx_BA+0x90	R/W	EADC 采样模块 4 控制寄存器	0x0000_0000
EADCx_SCTL5 x=0,1	EADCx_BA+0x94	R/W	EADC 采样模块 5 控制寄存器	0x0000_0000
EADCx_SCTL6 x=0,1	EADCx_BA+0x98	R/W	EADC 采样模块 6 控制寄存器	0x0000_0000
EADCx_SCTL7 x=0,1	EADCx_BA+0x9C	R/W	EADC 采样模块 7 控制寄存器	0x0000_0000
EADCx_SCTL8 x=0,1	EADCx_BA+0xA0	R/W	EADC 采样模块 8 控制寄存器	0x0000_0000
EADCx_SCTL9 x=0,1	EADCx_BA+0xA4	R/W	EADC 采样模块 9 控制寄存器	0x0000_0000
EADCx_SCTL10 x=0,1	EADCx_BA+0xA8	R/W	EADC 采样模块 10 控制寄存器	0x0000_0000
EADCx_SCTL11 x=0,1	EADCx_BA+0xAC	R/W	EADC 采样模块 11 控制寄存器	0x0000_0000
EADCx_SCTL12 x=0,1	EADCx_BA+0xB0	R/W	EADC 采样模块 12 控制寄存器	0x0000_0000
EADCx_SCTL13 x=0,1	EADCx_BA+0xB4	R/W	EADC 采样模块 13 控制寄存器	0x0000_0000
EADCx_SCTL14 x=0,1	EADCx_BA+0xB8	R/W	EADC 采样模块 14 控制寄存器	0x0000_0000
EADCx_SCTL15 x=0,1	EADCx_BA+0xBC	R/W	EADC 采样模块 15 控制寄存器	0x0000_0000

31	30	29	28	27	26	25	24
EXTSMPT							
23	22	21	20	19	18	17	16
Reserved	INTPOS	Reserved	TRGSEL				
15	14	13	12	11	10	9	8
TRGDLYCNT							
7	6	5	4	3	2	1	0
TRGDLYDIV		EXTFEN	EXTREN	CHSEL			

位	描述

位	描述	
[31:24]	**EXTSMPT**	**EADC 采样时间加长** 当 EADC转换速率较高，并且外部模拟电路的输出阻抗较高时，采样时间短，可能采样保持电容还没充完电就开始转换，EADC结果会偏小。这种情况可以加长采样时间。延时范围从 0~255 EADC时钟.
[23]	**Reserved**	保留.
[22]	**INTPOS**	**中断标志位置选择** 0 = 在EADC转换结束设置中断标志ADIFn (EADC_STATUS2[n], n=0~3). 1 = 在EADC转换开始设置中断标志ADIFn (EADC_STATUS2[n], n=0~3)
[21]	**Reserved**	保留.
[20:16]	**TRGSEL**	**EADC 模块转换触发源选择** 0H = 禁止触发. 1H = EADC0_ST外部管脚触发. 2H = EADC ADINT0 中断 EOC (转换完成) 脉冲触发. 3H = EADC ADINT1 中断 EOC (转换完成) 脉冲触发. 4H = Timer0 溢出脉冲触发. 5H = Timer1 溢出脉冲触发. 6H = Timer2溢出脉冲触发. 7H = Timer3溢出脉冲触发. 8H = EPWM0TG0. 9H = EPWM0TG1. AH = EPWM0TG2. BH = EPWM0TG3. CH = EPWM0TG4. DH = EPWM0TG5. EH = EPWM1TG0. FH = EPWM1TG1. 10H = EPWM1TG2. 11H = EPWM1TG3. 12H = EPWM1TG4. 13H = EPWM1TG5. 14H = BPWM0TG. 15H = BPWM1TG. 其他 = 保留.
[15:8]	**TRGDLYCNT**	**EADC 模块转换触发延迟时间** 触发延时 = TRGDLYCNT x EADC_CLK x n (n=1,2,4,16 由 TRGDLYDIV设定).
[7:6]	**TRGDLYDIV**	**EADC 模块转换触发延时除频选择** 延时时钟频率: 00 = EADC_CLK/1. 01 = EADC_CLK/2. 10 = EADC_CLK/4. 11 = EADC_CLK/16.

位	描述	
[5]	**EXTFEN**	**EADC 外部触发下降沿使能位** 0 = 当EADC 选择 EADC0_ST 作为触发源，禁止下降沿触发. 1 = 当 EADC 选择 EADC0_ST 作为触发源，使能下降沿触发.
[4]	**EXTREN**	**EADC 外部触发上升沿使能位** 0 = 当 EADC 选择 EADC0_ST 作为触发源，禁止上升沿触发. 1 = 当 EADC 选择 EADC0_ST 作为触发源，使能上升沿触发.
[3:0]	**CHSEL**	**EADC 模块通道选择** 00H = EADC_CH0 (慢速通道). 01H = EADC_CH1 (慢速通道). 02H = EADC_CH2 (慢速通道). 03H = EADC_CH3 (慢速通道). 04H = EADC_CH4 (慢速通道). 05H = EADC_CH5 (慢速通道). 06H = EADC_CH6 (慢速通道). 07H = EADC_CH7 (慢速通道). 08H = EADC_CH8 (慢速通道). 09H = EADC_CH9 (慢速通道). 0AH = EADC_CH10 (快速通道). 0BH = EADC_CH11 (快速通道). 0CH = EADC_CH12 (快速通道). 0DH = EADC_CH13 (快速通道). 0EH = EADC_CH14 (快速通道). 0FH = EADC_CH15 (快速通道).

EADC_SCTL16~18 EADC 模块 16~18 控制寄存器

寄存器	偏移量	R/W	描述	复位值
EADCx_SCTL16 x=0,1	EADCx_BA+0xC0	R/W	EADC 采样模块 16 控制寄存器	0x0000_0000
EADCx_SCTL17 x=0,1	EADCx_BA+0xC4	R/W	EADC 采样模块 17控制寄存器	0x0000_0000
EADCx_SCTL18 x=0,1	EADCx_BA+0xC8	R/W	EADC 采样模块 18 控制寄存器	0x0000_0000

31	30	29	28	27	26	25	24
EXTSMPT							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved							

位	描述	
[31:24]	**EXTSMPT**	**EADC 采样时间延长** 当 EADC转换速率较高时，采样时间短，可能采样保持电容还没充完电就开始转换，EADC结果会偏小。这种情况可以加长采样时间。 延时范围从 0~255 EADC时钟.
[23:0]	**Reserved**	保留.

EADC_INTSRC0~3 EADC 中断源使能控制寄存器

寄存器	偏移量	R/W	描述	复位值
EADCx_INTSRC0 x=0,1	EADCx_BA+0xD0	R/W	EADC中断 0 源使能控制寄存器.	0x0000_0000
EADCx_INTSRC1 x=0,1	EADCx_BA+0xD4	R/W	EADC中断 1 源使能控制寄存器.	0x0000_0000
EADCx_INTSRC2 x=0,1	EADCx_BA+0xD8	R/W	EADC中断 2 源使能控制寄存器.	0x0000_0000
EADCx_INTSRC3 x=0,1	EADCx_BA+0xDC	R/W	EADC中断 3 源使能控制寄存器.	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved					SPLIE18	SPLIE17	SPLIE16
15	14	13	12	11	10	9	8
SPLIE15	SPLIE14	SPLIE13	SPLIE12	SPLIE11	SPLIE10	SPLIE9	SPLIE8
7	6	5	4	3	2	1	0
SPLIE7	SPLIE6	SPLIE5	SPLIE4	SPLIE3	SPLIE2	SPLIE1	SPLIE0

位	描述	
[18]	**SPLIE18**	**模块 18 中断使能位** 0 = 模块 18 中断禁止. 1 = 模块 18 中断使能.
[17]	**SPLIE17**	**模块 17 中断使能位** 0 = 模块 17 中断禁止. 1 = 模块 17 中断使能.
[16]	**SPLIE16**	**模块 16 中断使能位** 0 = 模块 16 中断禁止. 1 = 模块 16 中断使能.
[15]	**SPLIE15**	**模块 15 中断使能位** 0 = 模块 15 中断禁止. 1 = 模块 15 中断使能.
[14]	**SPLIE14**	**模块 14 中断使能位** 0 = 模块 14 中断禁止. 1 = 模块 14 中断使能.

位	描述	
[13]	**SPLIE13**	**模块 13 中断使能位** 0 = 模块 13 中断禁止. 1 = 模块 13 中断使能.
[12]	**SPLIE12**	**模块 12 中断使能位** 0 = 模块 12 中断禁止. 1 = 模块 12 中断使能.
[11]	**SPLIE11**	**模块 11 中断使能位** 0 = 模块 11 中断禁止. 1 = 模块 11 中断使能.
[10]	**SPLIE10**	**模块 10 中断使能位** 0 = 模块 10 中断禁止. 1 = 模块 10 中断使能.
[9]	**SPLIE9**	**模块 9 中断使能位** 0 = 模块 9 中断禁止. 1 = 模块 9 中断使能.
[8]	**SPLIE8**	**模块 8 中断使能位** 0 = 模块 8 中断禁止. 1 = 模块 8 中断使能.
[7]	**SPLIE7**	**模块 7 中断使能位** 0 = 模块 7 中断禁止. 1 = 模块 7 中断使能.
[6]	**SPLIE6**	**模块 6 中断使能位** 0 = 模块 6 中断禁止. 1 = 模块 6 中断使能.
[5]	**SPLIE5**	**模块 5 中断使能位** 0 = 模块 5 中断禁止. 1 = 模块 5 中断使能.
[4]	**SPLIE4**	**模块 4 中断使能位** 0 = 模块 4 中断禁止. 1 = 模块 4 中断使能.
[3]	**SPLIE3**	**模块 3 中断使能位** 0 = 模块 3 中断禁止. 1 = 模块 3 中断使能.
[2]	**SPLIE2**	**模块 2 中断使能位** 0 = 模块 2 中断禁止. 1 = 模块 2 中断使能.
[1]	**SPLIE1**	**模块 1 中断使能位** 0 = 模块 1 中断禁止. 1 = 模块 1 中断使能.

位	描述	
[0]	**SPLIE0**	**模块 0 中断使能位** 0 = 模块 0 中断禁止. 1 = 模块 0 中断使能.

EADC_CMP0~3 EADC 结果比较寄存器 0/1/2/3

寄存器	偏移量	R/W	描述	复位值
EADCx_CMP0 x=0,1	EADCx_BA+0xE0	R/W	EADC 结果比较寄存器 0	0x0000_0000
EADCx_CMP1 x=0,1	EADCx_BA+0xE4	R/W	EADC 结果比较寄存器 1	0x0000_0000
EADCx_CMP2 x=0,1	EADCx_BA+0xE8	R/W	EADC 结果比较寄存器 2	0x0000_0000
EADCx_CMP3 x=0,1	EADCx_BA+0xEC	R/W	EADC 结果比较寄存器 3	0x0000_0000

31	30	29	28	27	26	25	24
Reserved				CMPDAT			
23	22	21	20	19	18	17	16
CMPDAT							
15	14	13	12	11	10	9	8
CMPWEN	Reserved			CMPMCNT			
7	6	5	4	3	2	1	0
CMPSPL					CMPCOND	ADCMPIE	ADCMPEN

位	描述	
[31:28]	**Reserved**	保留
[27:16]	**CMPDAT**	**比较数据** 这里 12 位的值用来和选定采样模块的转换结果比较。用户可以监控外部模拟输入管脚电压的变化，而不用增加软件负担.
[15]	**CMPWEN**	**窗口模式比较使能位** 0 = EADCMPF0 (EADC_STATUS2[4]) 当EADC_CMP0比较条件满足时置1。EADCMPF2 (EADC_STATUS2[6]) 当EADC_CMP2 比较条件满足时置1 。 1 = EADCMPF0 (EADC_STATUS2[4]) 当EADC_CMP0 和 EADC_CMP1 的比较条件都满足时置1。EADCMPF2 (EADC_STATUS2[6]) 当EADC_CMP2 和 EADC_CMP3 比较条件都满足时置1. **注意:** 这一位只在EADC_CMP0 和 EADC_CMP2 寄存器里存在.
[14:12]	**Reserved**	保留
[11:8]	**CMPMCNT**	**比较满足计数** 当选定的 EADC 通道转换结果满足CMPCOND (EADC_CMPn[2], n=0~3)定义的比较条件, 比较计数器材加 1.如果比较结果没有满足比较条件, 比较计数器重置为0. 当计数器计数到 (CMPMCNT +1)时，EADCMPFn (EADC_STATUS2[7:4], n=0~3) 会置1.

位	描述	
[7:3]	**CMPSPL**	**比较采样模块选择** 00000 = 采样模块 0 转换结果 EADC_DAT0被选择用来比较. 00001 = 采样模块 1 转换结果 EADC_DAT1被选择用来比较. 00010 = 采样模块 2 转换结果 EADC_DAT2被选择用来比较. 00011 = 采样模块 3 转换结果 EADC_DAT3被选择用来比较. 00100 = 采样模块 4 转换结果 EADC_DAT4被选择用来比较. 00101 = 采样模块 5 转换结果 EADC_DAT5被选择用来比较. 00110 = 采样模块 6 转换结果 EADC_DAT6被选择用来比较. 00111 = 采样模块 7 转换结果 EADC_DAT7被选择用来比较. 01000 = 采样模块 8 转换结果 EADC_DAT8被选择用来比较. 01001 = 采样模块 9 转换结果 EADC_DAT9被选择用来比较. 01010 = 采样模块 10 转换结果 EADC_DAT10被选择用来比较. 01011 = 采样模块 11 转换结果 EADC_DAT11被选择用来比较. 01100 = 采样模块 12 转换结果 EADC_DAT12被选择用来比较. 01101 = 采样模块 13 转换结果 EADC_DAT13被选择用来比较. 01110 = 采样模块 14 转换结果 EADC_DAT14被选择用来比较. 01111 = 采样模块 15 转换结果 EADC_DAT15被选择用来比较. 10000 = 采样模块 16 转换结果 EADC_DAT16被选择用来比较. 10001 = 采样模块 17 转换结果 EADC_DAT17被选择用来比较. 10010 = 采样模块 18 转换结果 EADC_DAT18被选择用来比较.
[2]	**CMPCOND**	**比较条件** 0= 当 12位EADC转换结果小于12位 CMPDAT (EADC_CMPn [27:16]),比较计数器加1 1=当 12位EADC转换结果大于等于12位 CMPDAT (EADC_CMPn [27:16]),比较计数器加1. **注意:**当比较计数器等于 (CMPMCNT (EADC_CMPn[11:8], n=0~3) +1), CMPF位置1
[1]	**ADCMPIE**	**EADC结果比较中断使能位** 0 = 比较功能中断禁止. 1 = 比较功能中断使能. 如果比较功能中断使能且转换结果满足 CMPCOND (EADC_CMPn[2], n=0~3) 和 CMPMCNT (EADC_CMPn[11:8], n=0~3)的条件 ，EADCMPFn (EADC_STATUS2[7:4], n=0~3) 会置1,如果 EADCMPIE设为1, 比较中断请求会产生.
[0]	**ADCMPEN**	**EADC 结果比较使能位** 0 = 禁止比较. 1 = 使能比较. 设置此位为 1 ， 当转换结果存入寄存器EADC_DAT，使能CMPDAT (EADC_CMPn[27:16], n=0~3) 与选定通道的转换结果比较.

EADC_STATUS0 EADC 状态寄存器 0

寄存器	偏移量	R/W	描述	复位值
EADCx_STATUS0 **x=0,1**	EADCx_BA+0xF0	R	EADC状态寄存器 0	0x0000_0000

31	30	29	28	27	26	25	24
OV							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
OV							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
VALID							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
VALID							

位	描述	
[31:16]	**OV**	**EADC_DAT0~15 覆盖标志** 这一位是EADC模块数据寄存器EADC_DATn. (n=0~15)中 OV位的镜像.
[15:0]	**VALID**	**EADC_DAT0~15 数据有效标志** 这一位是EADC模块数据寄存器EADC_DATn. (n=0~15)中 VALID位的镜像

EADC_STATUS1 EADC 状态寄存器1

寄存器	偏移量	R/W	描述	复位值
EADCx_STATUS1 x=0,1	EADCx_BA+0xF4	R	EADC 状态寄存器 1	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved					OV		
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved					VALID		

位	描述	
[31:19]	**Reserved**	保留.
[18:16]	**OV**	**EADC_DAT16~18 覆盖标志** 这一位是EADC模块数据寄存器EADC_DATn. (n=16~18)中 OV位的镜像
[15:3]	**Reserved**	保留.
[2:0]	**VALID**	**EADC_DAT16~18 数据有效标志** 这一位是EADC模块数据寄存器EADC_DATn. (n=16~18)中VALID位的镜像

EADC_STATUS2 EADC 状态寄存器 2

寄存器	偏移量	R/W	描述	复位值
EADCx_STATUS2 x=0,1	EADCx_BA+0xF8	R/W	EADC 状态寄存器 2	0x0000_0000

31	30	29	28	27	26	25	24
Reserved				AOV	AVALID	STOVF	ADOVIF
23	22	21	20	19	18	17	16
BUSY	Reserved		CHANNEL				
15	14	13	12	11	10	9	8
ADCMPO3	ADCMPO2	ADCMPO1	ADCMPO0	ADOVIF3	ADOVIF2	ADOVIF1	ADOVIF0
7	6	5	4	3	2	1	0
ADCMPF3	ADCMPF2	ADCMPF1	ADCMPF0	ADIF3	ADIF2	ADIF1	ADIF0

位	描述	
[31:28]	**Reserved**	保留.
[27]	**AOV**	**所有采样模块 EADC 结果数据寄存器覆盖标志确认(只读)** n=0~18. 0 = 没有采样模块 ADC 结果数据寄存器覆盖标志 OVn (EADC_DATn[16])设为 1. 1 = 有采样模块 ADC 结果数据寄存器覆盖标志 OVn (EADC_DATn[16])设为 1. **注意:** 任意 OVn 标志为1 此位保持为 1.
[26]	**AVALID**	**所有采样模块 EADC 结果数据寄存器数据有效标志确认(只读)** n=0~18. 0 = 没有有采样模块 ADC 结果数据寄存器数据有效标志VALIDn (EADC_DATn[17]) 设为 1. 1 = 有有采样模块 ADC 结果数据寄存器数据有效标志VALIDn (EADC_DATn[17]) 设为 1. **注意:** 任意 VALIDn标志为1 此位保持为 1..
[25]	**STOVF**	**所有 EADC 采样模块开始转换覆盖标志确认(只读)** n=0~18. 0 = 没有 ADC 采样模块开始转换覆盖标志 SPOVFn (EADC_OVSTS[n])设为 1. 1 = 有 ADC 采样模块开始转换覆盖标志 SPOVFn (EADC_OVSTS[n])设为 1. **注意:** 任意 SPOVFn 标志为1此位保持为1.
[24]	**ADOVIF**	**所有 EADC 中断标志覆盖位确认 (只读)** n=0~3. 0 = 没有ADINT 终端标识 ADOVIFn (EADC_STATUS2[11:8])被覆盖为 1. 1 = 有ADINT 终端标识 ADOVIFn (EADC_STATUS2[11:8])被覆盖为 1 **注意:** 当任意ADOVIFn 标志为 1，此位保持为1.

位	描述	
[23]	**BUSY**	**忙/闲 (只读)** 0 = EADC在空闲态. 1 = EADC在忙于转换
[22:21]	**Reserved**	保留.
[20:16]	**CHANNEL**	**目前转换通道(只读)** 这里反映当BUSY=1， EADC 目前的转换通道. 此位只读. 00H = EADC_CH0. 01H = EADC_CH1. 02H = EADC_CH2. 03H = EADC_CH3. 04H = EADC_CH4. 05H = EADC_CH5. 06H = EADC_CH6. 07H = EADC_CH7. 08H = EADC_CH8. 09H = EADC_CH9. 0AH = EADC_CH10. 0BH = EADC_CH11. 0CH = EADC_CH12. 0DH = EADC_CH13. 0EH = EADC_CH14. 0FH = EADC_CH15. 10H = V_{BG}. 11H = V_{TEMP}. 12H = V_{BAT}/4.
[15]	**ADCMPO3**	**EADC 比较模块 3 输出状态(只读)** 12 位比较模块3数据CMPDAT3 (EADC_CMP3[27:16]) 用来比较指定转换模块的转换结果. 可以用来监控模拟输入电压状态. 0 = EADC_DAT 的转换结果小于CMPDAT3的设置. 1 = EADC_DAT 的转换结果大于等于CMPDAT3的设置.
[14]	**ADCMPO2**	**EADC 比较模块 2 输出状态(只读)** 12 位比较模块2数据CMPDAT2 (EADC_CMP3[27:16]) 用来比较指定转换模块的转换结果. 可以用来监控模拟输入电压状态. 0 = EADC_DAT 的转换结果小于CMPDAT2的设置. 1 = EADC_DAT 的转换结果大于等于CMPDAT2的设置.
[13]	**ADCMPO1**	**EADC 比较模块 1 输出状态(只读)** 12 位比较模块1数据CMPDAT1 (EADC_CMP3[27:16]) 用来比较指定转换模块的转换结果. 可以用来监控模拟输入电压状态. 0 = EADC_DAT 的转换结果小于CMPDAT1的设置. 1 = EADC_DAT 的转换结果大于等于CMPDAT1的设置.

位	描述	
[12]	**ADCMPO0**	**EADC 比较模块 0 输出状态(只读)** 12 位比较模块0数据CMPDAT0 (EADC_CMP3[27:16]) 用来比较指定转换模块的转换结果. 可以用来监控模拟输入电压状态. 0 = EADC_DAT 的转换结果小于CMPDAT0的设置. 1 = EADC_DAT 的转换结果大于等于CMPDAT0的设置.
[11]	**ADOVIF3**	**EADC ADINT3 中断标志覆盖** 0 = ADINT3 中断标志没有覆盖到 1. 1 = ADINT3中断标志覆盖到 1. **注意:**此位写1清0.
[10]	**ADOVIF2**	**EADC ADINT2 中断标志覆盖** 0 = ADINT2 中断标志没有覆盖到 1. 1 = ADINT2中断标志覆盖到 1. **注意:**此位写1清0.
[9]	**ADOVIF1**	**EADC ADINT1 中断标志覆盖** 0 = ADINT1 中断标志没有覆盖到 1. 1 = ADINT1中断标志覆盖到 1. **注意:**此位写1清0.
[8]	**ADOVIF0**	**EADC ADINT0 中断标志覆盖** 0 = ADINT0 中断标志没有覆盖到 1. 1 = ADINT0中断标志覆盖到 1. **注意:**此位写1清0.
[7]	**ADCMPF3**	**EADC 比较模块 3 标志** 当选定的比较模块EADC 转换结果满足 EADC_CMP3 的条件，那么此位置 1. 0 = EADC_DAT的转换结果没有符合 EADC_CMP3寄存器的设定 1 = EADC_DAT的转换结果符合 EADC_CMP3寄存器的设定. **注意:**此位写1清0.
[6]	**ADCMPF2**	**EADC 比较模块 2 标志** 当选定的比较模块EADC 转换结果满足 EADC_CMP2 的条件，那么此位置 1. 0 = EADC_DAT的转换结果没有符合 EADC_CMP2寄存器的设定 1 = EADC_DAT的转换结果符合 EADC_CMP2寄存器的设定. **注意:**此位写1清0.
[5]	**ADCMPF1**	**EADC 比较模块 1 标志** 当选定的比较模块EADC 转换结果满足 EADC_CMP1 的条件，那么此位置 1. 0 = EADC_DAT的转换结果没有符合 EADC_CMP1寄存器的设定 1 = EADC_DAT的转换结果符合 EADC_CMP1寄存器的设定. **注意:**此位写1清0.

位	描述	
[4]	**ADCMPF0**	**EADC 比较模块 0 标志** 当选定的比较模块EADC 转换结果满足 EADC_CMP0 的条件，那么此位置 1. 0 = EADC_DAT的转换结果没有符合 EADC_CMP0寄存器的设定 1 = EADC_DAT的转换结果符合 EADC_CMP0寄存器的设定. **注意:**此位写1清0.
[3]	**ADIF3**	**EADC ADINT3中断标志** 0 = 没收到 ADINT3 中断脉冲. 1 = 收到 ADINT3 中断脉冲. **注意1:** 此位写1清0 **注意2:**此位反映选择的采样模块EADC 是否转换完成
[2]	**ADIF2**	**EADC ADINT2中断标志** 0 = 没收到 ADINT2 中断脉冲. 1 = 收到 ADINT2 中断脉冲. **注意1:** 此位写1清0 **注意2:**此位反映选择的采样模块 EADC 是否转换完成
[1]	**ADIF1**	**EADC ADINT1中断标志** 0 = 没收到 ADINT1 中断脉冲. 1 = 收到 ADINT1 中断脉冲. **注意1:** 此位写1清0 **注意2:**此位反映选择的采样模块 EADC 是否转换完成
[0]	**ADIF0**	**EADC ADINT0中断标志** 0 = 没收到 ADINT0 中断脉冲. 1 = 收到 ADINT0 中断脉冲. **注意1:** 此位写1清0 **注意2:**此位反映选择的采样模块 EADC 是否转换完成

EADC_STATUS3 EADC 状态寄存器 3

寄存器	偏移量	R/W	描述	复位值
EADCx_STATUS3 x=0,1	EADCx_BA+0xFC	R	EADC 状态寄存器 3	0x0000_001F

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved			**CURSPL**				

位	描述	
[31:5]	**Reserved**	保留.
[4:0]	**CURSPL**	**EADC 目前采样模块（只读）** 这个寄存器反映目前哪个模块执行EADC. 如果EADC空闲，这里读到 0x1F.

EADC_DDAT0~3 EADC 模块双数据寄存器

寄存器	偏移量	R/W	描述	复位值
EADCx_DDAT0 x=0,1	EADCx_BA+0x100	R	EADC 采样模块0双数据寄存器	0x0000_0000
EADCx_DDAT1 x=0,1	EADCx_BA+0x104	R	EADC 采样模块1双数据寄存器	0x0000_0000
EADCx_DDAT2 x=0,1	EADCx_BA+0x108	R	EADC 采样模块2双数据寄存器	0x0000_0000
EADCx_DDAT3 x=0,1	EADCx_BA+0x10C	R	EADC 采样模块3双数据寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved						**VALID**	**OV**
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
RESULT							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
RESULT							

位	描述	
[31:17]	**Reserved**	保留.
[17]	**VALID**	**有效标志** 0 = RESULT (EADC_DDATn[15:0])里的数据无效 1 = RESULT (EADC_DDATn[15:0])里的数据有效. 此位当相应的模块转换完成时置 1， 寄存器当EADC_DDATn. (n=0~3)被读取后由硬件清0.
[16]	**OV**	**覆盖标志** 0 = RESULT (EADC_DATn[15:0], n=0~3)里数据未被覆盖 1 = RESULT (EADC_DATn[15:0], n=0~3)里数据被覆盖. 如果RESULT[15:0]的数据在载入新值之前没有被读出， OV会置 1. 读EADC_DDAT 后硬件自动清0.
[15:0]	**RESULT**	**EADC 转换结果** 这里是 12位转换结果. 当 DMOF (EADC_CTL[9])为 0, 12-位 EADC 转换结果以无符号数存在RESULT [11:0] ，RESULT [15:12]填0. W当DMOF (EADC_CTL[9]) 为 1, 12位 EADC 转换结果以二进制补码存在RESULT [11:0] RESULT [15:12]填符号位.

EADC_PWRM EADC 电源管理寄存器

寄存器	偏移量	R/W	描述	复位值
EADCx_PWRM x=0,1	EADCx_BA+0x110	R/W	EADC电源管理寄存器	0x0006_E012

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved				**LDOSUT**			
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
LDOSUT							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved				**PWDMOD**		**PWUCALEN**	**PWUPRDY**

位	描述	
[31:20]	**Reserved**	保留.
[19:8]	**LDOSUT**	**EADC 内部 LDO 启动时间** 设置这里控制LDO 启动时间. LDO 需要的最短的启动时间是20us. LDO 启动时间 = (1/ADC_CLK) x LDOSUT.
[7:4]	**Reserved**	保留.
[3:2]	**PWDMOD**	**EADC掉电模式** 这里设置EADC 系统掉电时的模式. 00 = EADC 深度掉电模式. 01 = EADC 掉电. 10 = EADC 等待. 11 = EADC 深度掉电模式. **注意:**不同的 PWDMOD 有不同的上下电序列,为了避免错误的EADC序列，需要在下电和上电的时候保持 PWMOD的值不变.
[1]	**PWUCALEN**	**上电校验功能使能控制** 0 = 禁止上电校验功能. 1 = 使能上电校验功能. **注意:** 这一位与 CALSEL (EADC_CALCTL [3])一起起作用, {PWUCALEN,CALSEL } 描述: PWUCALEN 是 0 且 CALSEL 是 0: 不需要校准. PWUCALEN 是 0 且 CALSEL 是 1: 不需要校准. PWUCALEN 是 1 且 CALSEL 是 0: 启动载入校准值. PWUCALEN 是 1 且 CALSEL 是 1: 上电校准

位	描述	
[0]	**PWUPRDY**	**EADC 启动序列完成(只读)** 0 = EADC没有准备好，可能处在掉电状态或正在启动序列中. 1 = EADC准备好转换.

EADC_CALCTL EADC 校准控制寄存器

寄存器	偏移量	R/W	描述	复位值
EADCx_CALCTL x=0,1	EADCx_BA+0x114	R/W	EADC 校准控制寄存器	0x0000_0008

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved				CALSEL	CALDONE	CALSTART	Reserved

位	描述	
[31:4]	**Reserved**	保留.
[3]	**CALSEL**	**选择校准功能** 0 = 在校准功能有效时载入校准值. 1 = 在校准功能有效时执行校准.
[2]	**CALDONE**	**校准功能完成(只读)** 0 = 正在校准. 1 = 校准完成.
[1]	**CALSTART**	**校准功能启动** 0 = 停止校准功能. 1 = 启动校准功能. **注意1:** 此位由软件置1，重校准完成后由硬件清0 注意2: 将CALSTART（EADC_CALCTL [1]）设置为1再次开始校准之前，EADCDIV（CKL_CLKDIV0 [23:16]）必须为0。
[0]	**Reserved**	保留.

EADC_CALDWORD EADC 校准载入字寄存器

寄存器	偏移量	R/W	描述	复位值
EADCx_CALDWRD x=0,1	EADCx_BA+0x118	R/W	EADC校准载入字寄存器	0x0000_00XX

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved	**CALWORD**						

位	描述	
[31:7]	**Reserved**	保留.
[6:0]	**CALWORD**	**校准字** 在载入校准前写入预校准的值 校准完成后读这个寄存器的值. **注意:** 校准包括两个部分 “校准“ 和 “载入校准”; 如果配置为 “校准“; 这个寄存器存放之前校准完成后的结果; 如果配置为 “载入校准” ; 在载入之前配置这个寄存器, 载入校准完成后, 载入的较准字会应用到EADC; 载入校准功能中， 载入的值直到校准完成后才会等于原始的CALWORD

EADC PDMA控制寄存器(EADC_PDMACTL)

寄存器	偏移量	R/W	描述	复位值
EADCx_PDMACTL x=0,1	EADCx_BA+0x130	R/W	EADC PDMA 控制寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved					**PDMATEN**		
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
PDMATEN							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
PDMATEN							

位	描述	
[31:19]	**保留**	保留
[18:0]	**PDMATEN**	**PDMA 传输使能位** 当EADC转换结束，EADC的值将被存储到EADC_DATn (n: 0 ~ 18)寄存器，用户使能此位可以触发PDMA搬运数据 0 = PDMA 数据传输禁止 1 = PDMA 数据传输使能 **注:** . 当此位为1，用户必须设置EADCIENn (EADC_CTL[5:2], n=0~3) = 0禁止中断

6.41 DAC 数模转换器

6.41.1 概述

DAC是一个12位数模转换器，可配置为12位或 8位输出模式，并且可以由PDMA 控制。集成了一个电压输出缓冲器，可以减小输出阻抗并且可以直接驱动外部负载无需再加外部运算放大器。

6.41.2 特性

- 输出电压范围: 0~AVDD.
- 支持 12位或8位输出模式.
- 轨到轨建立时间 8us.
- 支持 12-位 1 MSPS 电压类型 DAC.
- 参考电压可以选择内部参考电压 (INT_VREF)或VREF管脚.
- DAC 最大更新率 1 MSPS.
- 支持电压输出缓存模式和旁路电压输出缓存模式.
- 支持软件和硬件触发, 包括 Timer0~3, EPWM0, EPWM1,和外部触发管脚触发 DAC 转换.
- 支持 PDMA 模式.
- 两个DAC支持组模式，同步更新.

6.41.3 框图

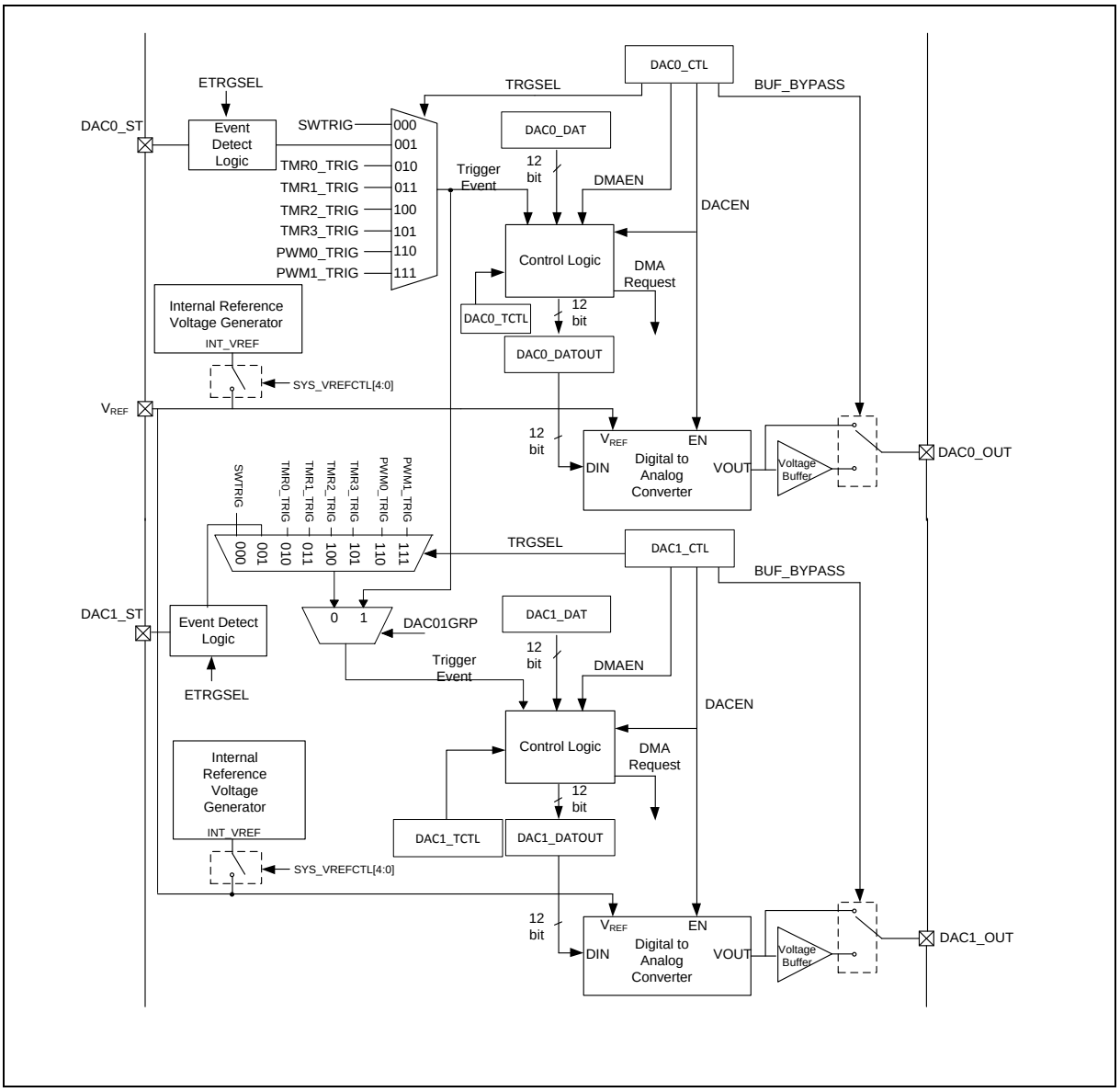


图 6.41-1 数模转换器框图

6.41.4 基本配置

6.41.4.1 DAC0 基本配置

- 时钟源配置
  - 在DACCKEN (CLK_APBCLK1[12])使能 DAC0 外设时钟.
- 复位设置
  - 在DACRST (SYS_IPRST2[12])复位 DAC0 控制器.
- 管脚配置

组	管脚名称	GPIO	MFP
DAC0	DAC0_OUT	PB.12	MFP1
	DAC0_ST	PA.10	MFP14
		PA.0	MFP15

6.41.4.2 DAC1 基本配置

- 时钟源配置
  - 在DACCKEN (CLK_APBCLK1[12])使能 DAC1 外设时钟.
- 复位设置
  - 在DACRST (SYS_IPRST2[12])复位 DAC1 控制器.
- 管脚配置

组	管脚名称	GPIO	MFP
DAC1	DAC1_OUT	PB.13	MFP1
	DAC1_ST	PA.11	MFP14
		PA.1	MFP15

6.41.5 功能描述

6.41.5.1 DAC 输出

DAC 集成集成了一个电压输出缓冲器来降低输出阻抗，无需外加运放可直接驱动负载，输出缓存由BYPASS (DACn_CTL[8])n=0, 1使能或禁用。DAC最大输出电压受到所选择的参考源限制。

6.41.5.2 DAC参考电压

DAC 的参考电压与 EADC 共享，由系统管理控制寄存器VREFCTL (SYS_VREFCTL[4:0]) 配置。可配置为外部参考电压 (V_{REF}) 或内部参考电压 (INT_VREF).

6.41.5.3 DAC 数据格式

支持数据左对齐或右对齐模式。需要设置下列寄存器:

- 12-位左对齐: 用户把数据存入 DACn_DAT[15:4] 位. DACn_DAT[31:16] 和 DACn_DAT[3:0] 在 DAC 转换中被忽略.
- 12-位右对齐: 用户把数据存入 DACn_DAT[11:0] 位, DACn_DAT[31:12]在 DAC 转换中被忽略

当 DAC 工作在 8-位模式,对齐设定无效. 要设定 8-位模式, 设置 BWSEL(DACn_CTL[15:14])为 01. 否则保持BWSEL 为 00.

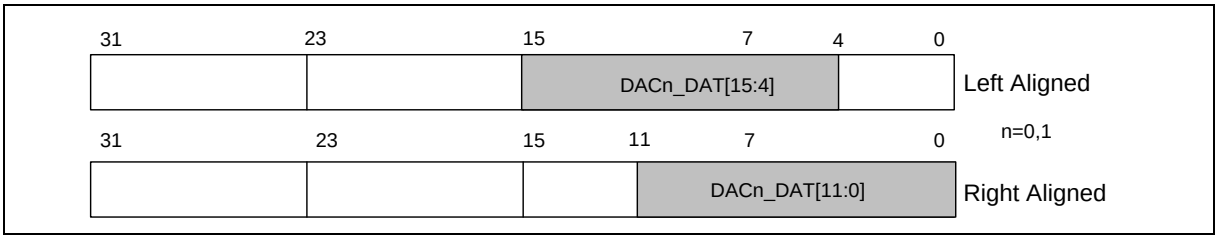


图 6.41-2 数据保持寄存器格式

6.41.5.4 DAC 转换

图6.41-3展示软件启动 DAC 转换过程。写数据到寄存器 DACn_DAT, 数据会载入输出寄存器 DACn_DATOUT， DAC 在一个PCLK (APB clock) 时钟后开始转换. 图6.41-4展示硬件触发开始DAC (外部管脚 DACn_ST, 定时器触发事件或EPWM 触发事件). DACn_DAT 寄存器里写入的数据在一个 PCLK (APB clock)后，传输到输出缓存DACn_DATOUT.

输出寄存器 DACn_DATOUT载入 DACn_DAT 的内容,转换建立时间后变为有效。输入从最低 (0x000) 到最高 (0xFFF)需要8us 。相邻的数值转换建立时间是1us。 DAC 提供一个 10-位计数器计算转换时间。连续操作时，需要往寄存器 SETTLET (DACn_TCTL[9:0])配置 DAC 转换时间。这个值必须大于 DAC电器特性表列出的 DAC 转换建立时间。例如，当 DAC APB 时钟速度 80MHz 且 DAC 转换建立时间是8us, 选择的 SETTLET 值必须大于 0x280。转换开始后，转换完成标志 FINISH (DACn_STATUS[0]) 会被硬件清 0，当计数器计数到 SETTLET时置1

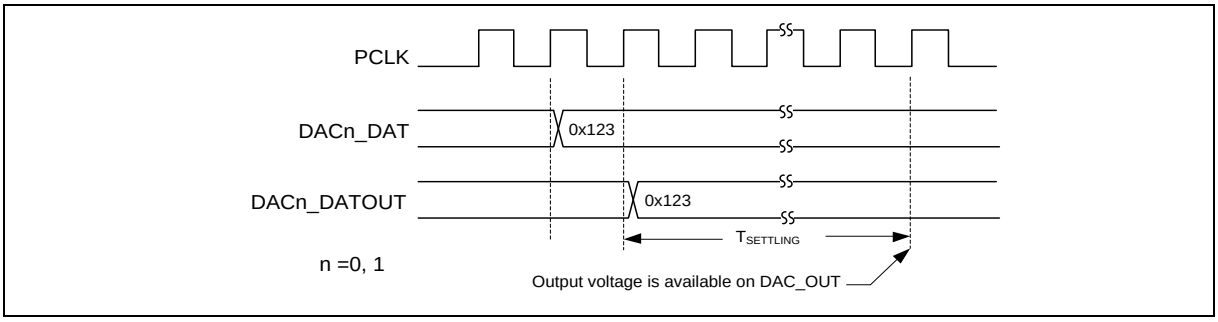


图 6.41-3DAC 转换由软件写触发

6.41.5.5 DAC输出电压

DAC输出电压范围是0到V_{REF}，管脚输出的模拟电压由下列公式计算:

DACOUT = V_{REF} * $\frac{DATnOUT[11:0]}{4096}$, n=0,1

6.41.5.6 DAC触发选择

DAC 转换可以由写DACn_DAT触发, 软件触发或硬件触发。TRGEN (DACn_CTL[4]) = 0时，转换由写寄存器DACn_DAT开始， TRGEN (DACn_CTL[4])= 1时, 转换由外部 DACn_ST 管脚,定时器,或EPWM 触发。若选择软件触发, 每次写 SWTRG (DACn_SWTRG[0])为 1触发一次转换。当 DACn_DATOUT 载入DACDAT 内容后SWTRG 由硬件清0。TRGSEL (DACn_CTL[7:5]) 选择8种触发转换方式。

当 DAC 检测到触发事件输入, 之前写到 DACDAT 的数据传输到 DACn_DATOUT[11:0]， DAC 转换在 1 个 PCLK (APB 时钟) 后开始.

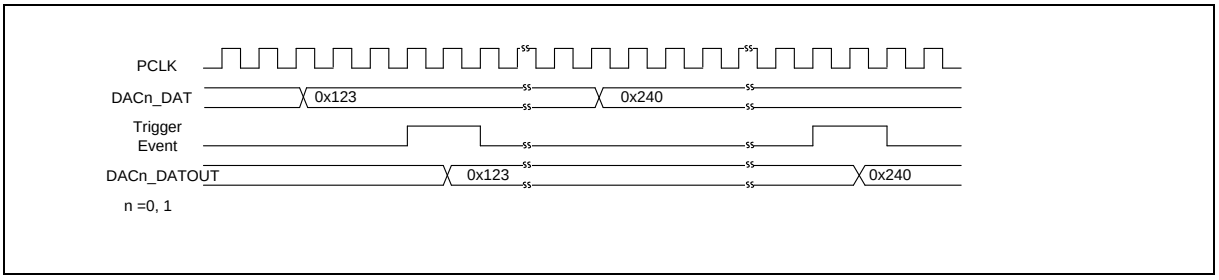


图 6.41-4 DAC 转换由硬件触发事件开始

6.41.5.7 DAC组模式

GRPEN (DAC0_CTL[16]) 可配置DAC0和DAC1为组模式，硬件保证两个 DAC 同时更新。组模式下DAC1的行为由DAC0_CTL 和 DAC0_TCTL控制，DAC1_CTL 和 DAC1_TCTL 不起作用。图6.41-5展示组模式与普通模式的比较.

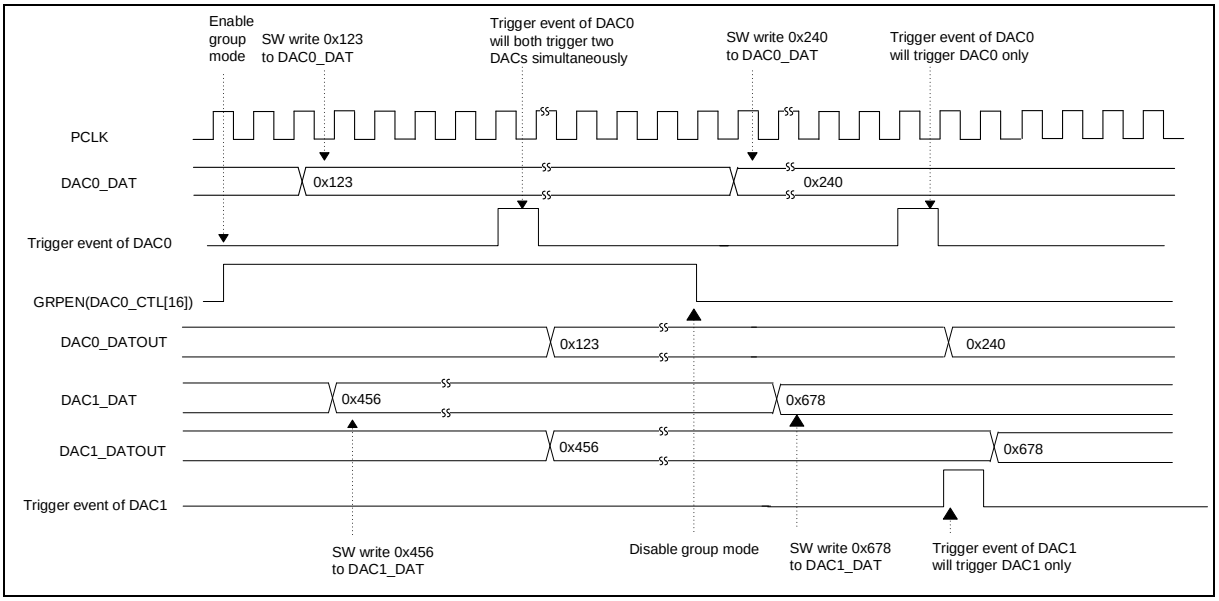


图 6.41-5DAC0 和 DAC1 组模式和非组模式更新示例

6.41.5.8 DMA操作

DMAEN (DACn_CTL[2]) =1使能DMA功能，DAC DMA 请求在硬件触发时产生。DACn_DAT 的内容传到 DACn_DATOUT[11:0] ， DAC 在1个PCLK (APB 时钟)后开始转换。写到DACn_DAT的新数据在下次触发时转换。图6.41-6展示 DAC PDMA下溢的情形 ，当第二个 DMA 请求在第一笔转换完成前到来，新的PDMA 请求不会执行， DMA 下溢标志 DMAUDR (DACn_STATUS[1])置 1 报告错误情况。DMA 数据传输禁止且不触发新的 DMA 请求，DAC 继续转换最后的数据。如果相应的DMAURIEN (DACn_CTL[3]) 使能就会产生中断。用户需要及时调整触发的频率或定时器、EPWM等，然后重新开始 DAC 转换.

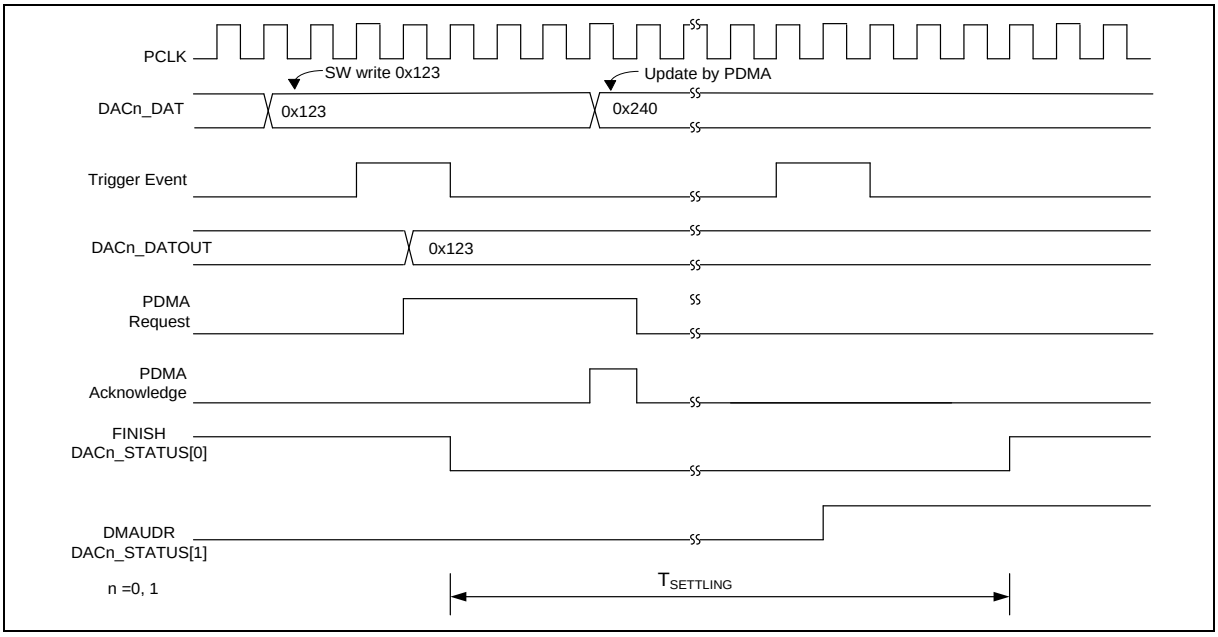


图 6.41-6DAC PDMA 下溢条件示例

DMA 请求也可以由软件使能产生，用户设置 DMAEN (DACn_CTL[2]) 为 1 以及 TRGEN (DACn_CTL[4]) 为 0, DMA 请求周期产生，周期依据 SETTLET (DACn_TCTL[9:0]) 的值. DAC输出周期性更新。当用户清 DMAEN (DACn_CTL[2]) 为 0, DAC 控制器会停止产生新的 PDMA 传输请求。图 6.41-7提供软件PDMA模式 DAC 连续转换的示例。

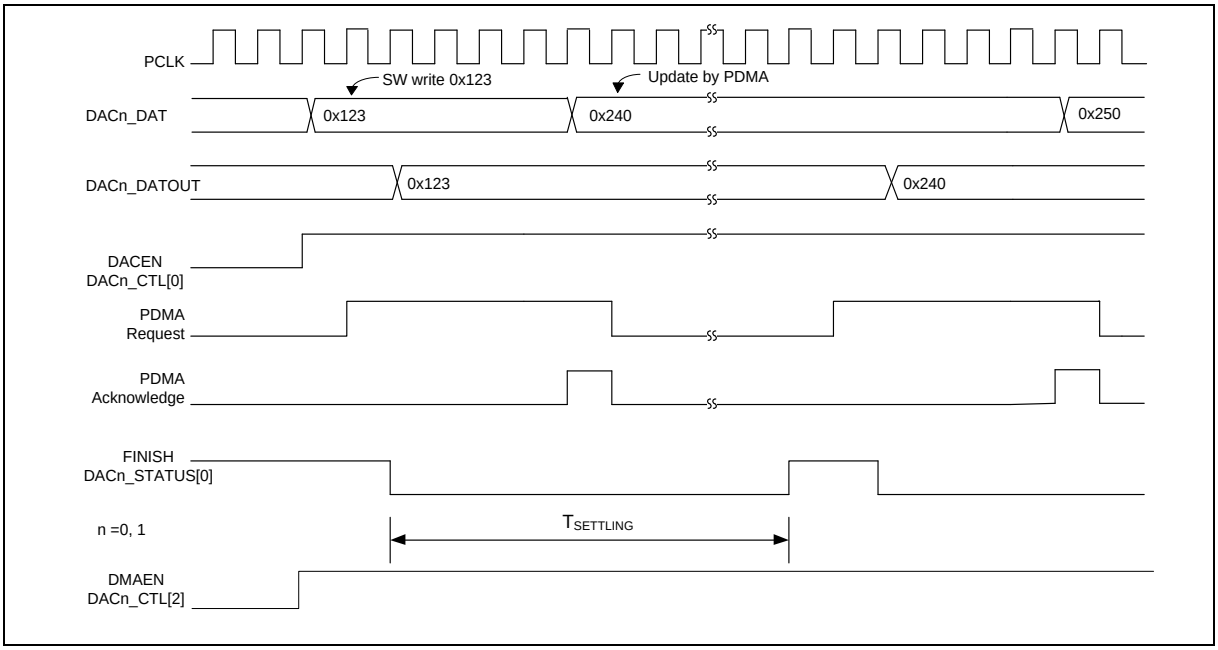


图 6.41-7 软件 PDMA 模式 DAC 连续转换

6.41.5.9 中断源

DAC 控制器有两个中断源, 一个是DAC 数据转换完成中断，另一个是 DMA下溢中断，如图6.41-8. DAC 转换完成, FINISH (DACn_STATUS[0])位会置 1 ，如果DACIEN (DACn_CTL[1])=1则产生中断。如果新的 DMA 触发事件在DAC 数据转换完成之前产生, DMA 下溢标志 DMAUDR (DACn_STATUS[1])

置1，若 DMAURIEN (DACn_CTL[3])=1使能了中断，会申请中断。

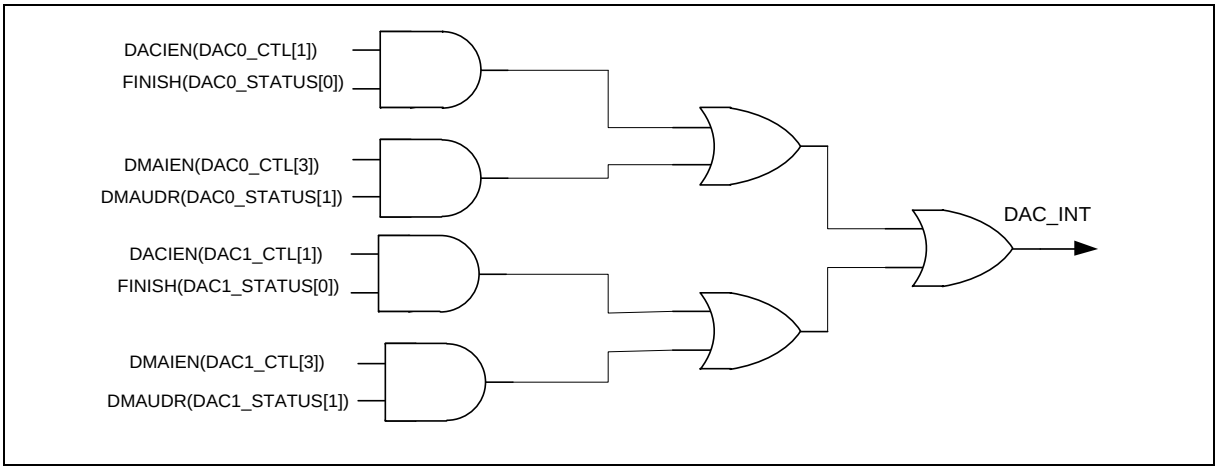


图 6.41-8 DAC 中断源

6.41.6 寄存器映射

R: 只读, **W**: 只写, **R/W**: 可读可写

寄存器	偏移量	R/W	描述	复位值
DAC 基地址: **DAC_BA = 0x4004_7000**				
DAC0_CTL	DAC_BA+0x00	R/W	DAC0 控制寄存器	0x0000_0000
DAC0_SWTRG	DAC_BA+0x04	R/W	DAC0 软件触发控制寄存器	0x0000_0000
DAC0_DAT	DAC_BA+0x08	R/W	DAC0 数据保持寄存器	0x0000_0000
DAC0_DATOUT	DAC_BA+0x0C	R	DAC0 数据输出寄存器	0x0000_0000
DAC0_STATUS	DAC_BA+0x10	R/W	DAC0 状态寄存器	0x0000_0000
DAC0_TCTL	DAC_BA+0x14	R/W	DAC0 时序控制寄存器	0x0000_0000
DAC1_CTL	DAC_BA+0x40	R/W	DAC1 控制寄存器	0x0000_0000
DAC1_SWTRG	DAC_BA+0x44	R/W	DAC1 软件触发控制寄存器	0x0000_0000
DAC1_DAT	DAC_BA+0x48	R/W	DAC1 数据保持寄存器	0x0000_0000
DAC1_DATOUT	DAC_BA+0x4C	R	DAC1 数据输出寄存器	0x0000_0000
DAC1_STATUS	DAC_BA+0x50	R/W	DAC1 状态寄存器	0x0000_0000
DAC1_TCTL	DAC_BA+0x54	R/W	DAC1 时序控制寄存器	0x0000_0000

6.41.7 寄存器描述

DAC0_CTL DAC0 控制寄存器

寄存器	偏移量	R/W	描述	复位值
DAC0_CTL	DAC_BA+0x00	R/W	DAC0 控制寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							GRPEN
15	14	13	12	11	10	9	8
BWSEL		ETRGSEL		Reserved	LALIGN	Reserved	BYPASS
7	6	5	4	3	2	1	0
TRGSEL			TRGEN	DMAURIEN	DMAEN	DACIEN	DACEN

位	描述	
[31:17]	**Reserved**	保留.
[16]	**GRPEN**	**DAC 组模式使能位** 0 = DAC0 和 DAC1 组模式禁止. 1 = DAC0 和 DAC1 组模式使能.
[15:14]	**BWSEL**	**DAC 数据位宽选择** 00 = 数据是 12 位. 01 = 数据是 8 位. 其他 = 保留.
[13:12]	**ETRGSEL**	**外部管脚触发选择** 00 = 低电平触发. 01 = 高电平触发. 10 = 下降沿触发 11 = 上升沿触发
[11]	**Reserved**	保留.
[10]	**LALIGN**	**DAC数据左对齐使能控制位** 0 = 右对齐. 1 = 左对齐
[9]	**Reserved**	保留.
[8]	**BYPASS**	**输出缓存选择** 0 = 输出电压缓存使能. 1 = 输出电压缓存禁止.
[7:5]	**TRGSEL**	**触发源选择**

		000 = 软件触发. 001 = 外部管脚 DAC0_ST触发 010 = 定时器 0 触发. 011 = 定时器1 触发. 100 = 定时器 2 触发. 101 = 定时器 3 触发. 110 = EPWM0 触发. 111 = EPWM1 触发.
[4]	**TRGEN**	**触发模式使能位** 0 = DAC 事件触发模式禁止. 1 = DAC 事件触发模式使能.
[3]	**DMAURIEN**	**DMA 下溢中断使能位** 0 = DMA 下溢中断禁止. 1 = DMA 下溢中断使能.
[2]	**DMAEN**	**DMA 模式使能位** 0 = DMA 模式禁止. 1 = DMA 模式使能.
[1]	**DACIEN**	**DAC 中断使能位** 0 = 中断禁止. 1 = 中断使能.
[0]	**DACEN**	**DAC 使能位** 0 = DAC 禁止. 1 = DAC 使能.

DAC0_SWTRG DAC0 软件触发控制寄存器

寄存器	偏移量	R/W	描述	复位值
DAC0_SWTRG	DAC_BA+0x04	R/W	DAC0 软件触发控制寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved							**SWTRG**

位	描述	
[31:1]	**Reserved**	保留
[0]	**SWTRG**	**软件触发** 0 = 禁止软件触发. 1 = 使能软件触发. 用户向这一位写1产生一个一次性脉冲，然后由硬件清0；读此位总为0

DAC0_DAT DAC0 数据保持寄存器

寄存器	偏移量	R/W	描述	复位值
DAC0_DAT	DAC_BA+0x08	R/W	DAC0数据保持寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
DACDAT							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
DACDAT							

位	描述	
[31:16]	**Reserved**	保留.
[15:0]	**DACDAT**	**DAC 12-位保持数据** 写DAC 输出的12-位数据. 不使用的位 (左对齐DAC_DAT[3:0]，右对齐DAC_DAT[15:12]) 会被 DAC控制器硬件忽略. 12 位左对齐:数据填到 DAC_DAT[15:4]. 12 位右对齐:数据填到 DAC_DAT[11:0].

DAC0_DATOUT DAC0数据输出寄存器

寄存器	偏移量	R/W	描述	复位值
DAC0_DATOUT	DAC_BA+0x0C	R	DAC0 数据输出寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved				DATOUT			
7	6	5	4	3	2	1	0
DATOUT							

位	描述	
[31:12]	**Reserved**	保留
[11:0]	**DATOUT**	**DAC 12-位输出数据** 这里是目前数据用来 DAC 输出转换. 从 DAC_DAT 寄存器搬来，用户不能直接改写

DAC0_STATUS DAC0 状态寄存器

寄存器	偏移量	R/W	描述	复位值
DAC0_STATUS	DAC_BA+0x10	R/W	DAC0 状态寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							BUSY
7	6	5	4	3	2	1	0
Reserved						DMAUDR	FINISH

位	描述	
[31:9]	**Reserved**	保留.
[8]	**BUSY**	**DAC 忙标志 (只读)** 0 = DAC准备好下一次转换. 1 = DAC正忙于转换.
[7:2]	**Reserved**	保留.
[1]	**DMAUDR**	**DMA 下溢中断标志** 0 = 没有 DMA 下溢错误状况发生. 1 = 有 DMA 下溢错误状况发生. 此位写1清0.
[0]	**FINISH**	**DAC 转换完成中断** 0 = DAC正在转换. 1 = DAC转换完成. 转换计数器计到 SETTLET时此位置1 . 当 DAC开始新的转换时清0. 用户写1可以清0.

DAC0_TCTL DAC0 时序控制寄存器

寄存器	偏移量	R/W	描述	复位值
DAC0_TCTL	DAC_BA+0x14	R/W	DAC0 时序控制寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved						**SETTLET**	
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
SETTLET							

位	描述	
[31:10]	**Reserved**	保留.
[9:0]	**SETTLET**	**DAC输出建立时间** 填的值必须满足DAC 转换建立时间，单位是 PCLK (APB时钟) 例如, DAC 时钟是 80MHz 且 DAC 转换建立时间是 1 us, SETTLET值必须大于 0x50. SELTTLET = DAC控制器时钟频率 x 建立时间

DAC1_CTL DAC1 控制寄存器

寄存器	偏移量	R/W	描述	复位值
DAC1_CTL	DAC_BA+0x40	R/W	DAC1 控制寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
BWSEL		ETRGSEL		Reserved	LALIGN	Reserved	BYPASS
7	6	5	4	3	2	1	0
TRGSEL			TRGEN	DMAURIEN	DMAEN	DACIEN	DACEN

位	描述	
[31:16]	**Reserved**	保留
[15:14]	**BWSEL**	**DAC 数据位宽选择** 00 = 数据是 12 位. 01 = 数据是 8 位. 其他 = 保留.
[13:12]	**ETRGSEL**	**外部管脚触发选择** 00 = 低电平触发. 01 = 高电平触发. 10 = 下降沿触发 11 = 上升沿触发
[11]	**Reserved**	保留.
[10]	**LALIGN**	**DAC数据左对齐使能控制** 0 = 右对齐. 1 = 左对齐
[9]	**Reserved**	保留.
[8]	**BYPASS**	**输出缓存选择** 0 = 输出电压缓存使能. 1 = 输出电压缓存禁止.
[7:5]	**TRGSEL**	**触发源选择** 000 = 软件触发. 001 = 外部管脚 DAC0_ST触发 010 = 定时器 0 触发. 011 = 定时器1 触发.

		100 = 定时器 2 触发. 101 = 定时器 3 触发. 110 = EPWM0 触发. 111 = EPWM1 触发.
[4]	**TRGEN**	**触发模式使能位** 0 = DAC 事件触发模式禁止. 1 = DAC 事件触发模式使能.
[3]	**DMAURIEN**	**DMA 下溢中断使能位** 0 = DMA 下溢中断禁止. 1 = DMA 下溢中断使能.
[2]	**DMAEN**	**DMA 模式使能位** 0 = DMA 模式禁止. 1 = DMA 模式使能.
[1]	**DACIEN**	**DAC 中断使能位** 0 = 中断禁止. 1 = 中断使能.
[0]	**DACEN**	**DAC 使能位** 0 = DAC 禁止. 1 = DAC 使能.

DAC1_SWTRG DAC1 软件触发控制寄存器

寄存器	偏移量	R/W	描述	复位值
DAC1_SWTRG	DAC_BA+0x44	R/W	DAC1 软件触发控制寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved							SWTRG

位	描述	
[31:1]	Reserved	保留
[0]	SWTRG	**软件触发** 0 = 禁止软件触发. 1 = 使能软件触发. 用户向这一位写1产生一个一次性脉冲，然后由硬件清0；读此位总为0

DAC1_DAT DAC1 数据保持寄存器

寄存器	偏移量	R/W	描述	复位值
DAC1_DAT	DAC_BA+0x48	R/W	DAC1数据保持寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
DACDAT							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
DACDAT							

位	描述	
[31:16]	**Reserved**	保留.
[15:0]	**DACDAT**	**DAC 12-位保持数据** 写DAC 输出的12-位数据. 不使用的位 (左对齐DAC_DAT[3:0]，右对齐DAC_DAT[15:12]) 会被 DAC控制器硬件忽略. 12 位左对齐:数据填到 DAC_DAT[15:4]. 12 位右对齐:数据填到 DAC_DAT[11:0].

DAC1_DATOUT DAC1 数据输出寄存器

寄存器	偏移量	R/W	描述	复位值
DAC1_DATOUT	DAC_BA+0x4C	R	DAC1 数据输出寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved				DATOUT			
7	6	5	4	3	2	1	0
DATOUT							

位	描述	
[31:12]	Reserved	保留
[11:0]	DATOUT	DAC 12-位输出数据 这里是目前数据用来 DAC 输出转换. 从 DAC_DAT 寄存器搬来，用户不能直接改写

DAC1_STATUS DAC1 状态寄存器

寄存器	偏移量	R/W	描述	复位值
DAC1_STATUS	DAC_BA+0x50	R/W	DAC1 状态寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							BUSY
7	6	5	4	3	2	1	0
Reserved						DMAUDR	FINISH

位	描述	
[31:9]	**Reserved**	保留.
[8]	**BUSY**	**DAC 忙标志 (只读)** 0 = DAC准备好下一次转换. 1 = DAC正忙于转换.
[7:2]	**Reserved**	保留.
[1]	**DMAUDR**	**DMA 下溢中断标志** 0 = 没有 DMA 下溢. 1 = 有 DMA 下溢错误状况发生. 此位写1清0.
[0]	**FINISH**	**DAC 转换完成中断** 0 = DAC正在转换. 1 = DAC转换完成. 转换计数器计到 SETTLET时此位置1 . 当 DAC开始新的转换时清0. 用户写1可以清0.

DAC1_TCTL DAC1 时序控制寄存器

寄存器	偏移量	R/W	描述	复位值
DAC1_TCTL	DAC_BA+0x54	R/W	DAC1 时序控制寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved						**SETTLET**	
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
SETTLET							

位	描述	
[31:10]	**Reserved**	保留.
[9:0]	**SETTLET**	**DAC输出建立时间** 填的值必须满足DAC 转换建立时间，单位是 PCLK (APB时钟) 例如, DAC 时钟是 80MHz 且 DAC 转换建立时间是 1 us, SETTLET值必须大于 0x50. SELTTLET = DAC控制器时钟频率 x 建立时间

6.42 ACMP 模拟比较器控制器

6.42.1 概述

芯片提供两个比较器。当正输入大于负输入时比较器输出逻辑1。每个比较器在输出翻转时都可以配置产生中断。

6.42.2 特性

- 输入电压范围: 0 ~ V_{DDA} (AV_{DD}管脚电压)
- 两组轨到轨模拟比较器
- 支持迟滞回差功能
  - 迟滞回差电压: 0mV, 10mV, 20mV 和 30mV
- 支持唤醒功能
- 支持传输延时和低功耗模式
- 可配置正输入和负输入引脚
- ACMP0 支持:
  - 可选4个 I/O 管脚做正输入源:
    - ACMP0_P0, ACMP0_P1, ACMP0_P2,或ACMP0_P3
  - 4 个负端输入源:
    - ACMP0_N
    - 比较器参考电压(CRV)
    - 内部带隙电压(VBG)
    - DAC0输出(DAC0_OUT)
- ACMP1 支持
  - 可选4个 I/O 管脚做正输入源:
    - ACMP1_P0, ACMP1_P1, ACMP1_P2,或ACMP1_P3
  - 4 个负端输入源:
    - ACMP1_N
    - 比较器参考电压(CRV)
    - 内部带隙电压(VBG)
    - DAC0输出(DAC0_OUT)
- 共享一个 ACMP 中断向量
- 比较结果改变时产生中断 (中断条件可配置)
- 支持触发 EPWM 急停事件
- 支持窗口比较模式和窗口锁存模式

6.42.3 框图

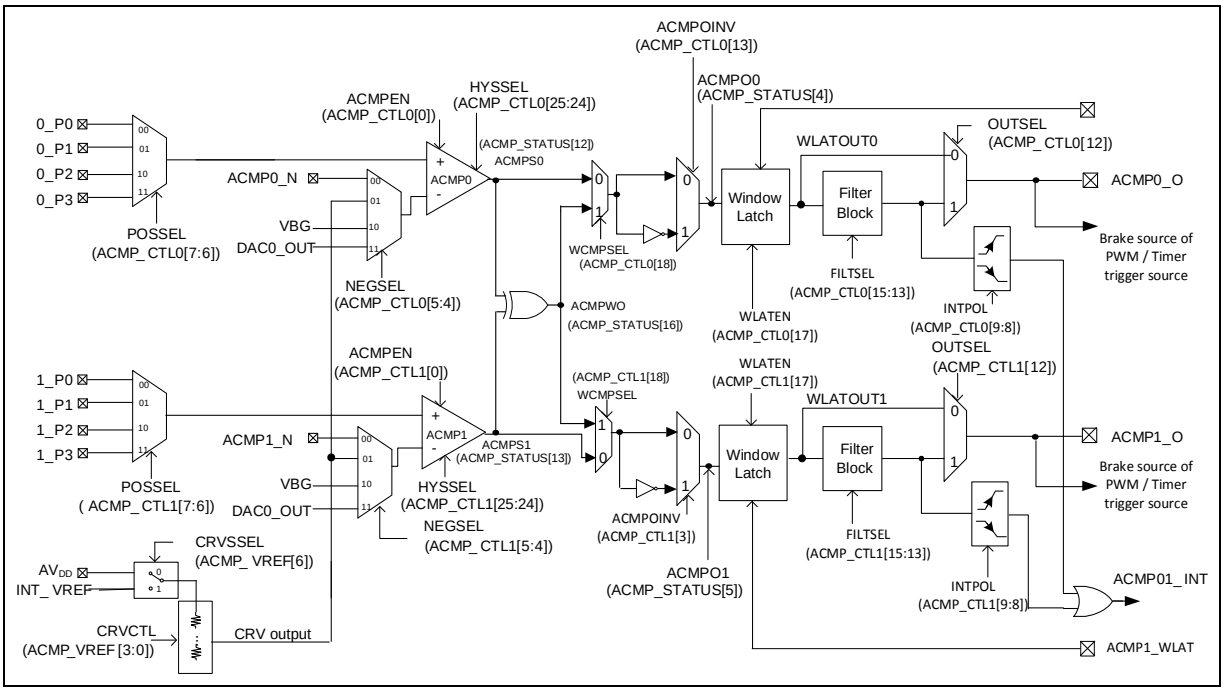


图 6.42-1 模拟比较器框图

6.42.4 基本配置

6.42.4.1 ACMP0 基本配置

- 时钟源配置
  - 在ACMP01CKEN (CLK_APBCLK0[7])使能 ACMP0外设时钟.
- 复位配置
  - 在ACMP01RST (SYS_IPRST1[7]) 复位 ACMP0 控制器.
- 管脚配置

组	管脚名称	GPIO	MFP
ACMP0	ACMP0_N	PB.3	MFP1
	ACMP0_O	PC.1, PC.12	MFP14
		PB.7	MFP15
	ACMP0_P0	PA.11	MFP1
	ACMP0_P1	PB.2	MFP1
	ACMP0_P2	PB.12	MFP1
	ACMP0_P3	PB.13	MFP1
	ACMP0_WLAT	PA.7	MFP13

6.42.4.2 ACMP1 基本配置

- 时钟源配置

- 在ACMP01CKEN (CLK_APBCLK0[7]) 使能 ACMP1 外设时钟.

- 复位配置
  - 在ACMP01RST (SYS_IPRST1[7]) 复位 ACMP1 控制器.
- 管脚配置

组	管脚名称	GPIO	MFP
ACMP1	ACMP1_N	PB.5	MFP1
	ACMP1_O	PC.0, PC.11	MFP14
		PB.6	MFP15
	ACMP1_P0	PA.10	MFP1
	ACMP1_P1	PB.4	MFP1
	ACMP1_P2	PB.12	MFP1
	ACMP1_P3	PB.13	MFP1
	ACMP1_WLAT	PA.6	MFP13

6.42.5 功能描述

6.42.5.1 迟滞回差功能

模拟比较器输出支持迟滞回差功能，保证输出的稳定，参考图6.42-2. 如果比较器输出 0, 直到正端电压比负端电压高于阈值电压时输出才会变为1。同样的，如果比较器输出为1，直到正端电压比负端电压低，且压差大于阈值电压时才会变0。

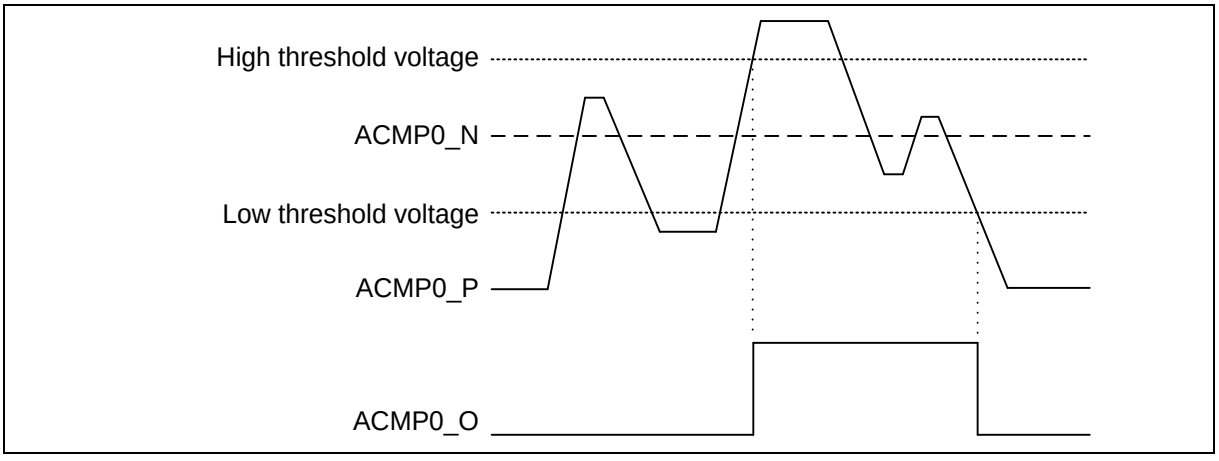


图 6.42-2 ACMP0 比较器迟滞回差功能

6.42.5.2 窗口锁存模式

图6.42-3未比较器工作在窗口锁存模式。设置WLATEN (ACMP_CTL0/1[17]) 为 1使能窗口锁存模式。当使能窗口锁存功能时，ACMP0/1_WLAT管脚用来控制输出WLATOUT0/1。当ACMP0/1_WLAT管脚是高, ACMPO0/1 输出到WLATOUT0/1；当ACMP0/1_WLAT管脚是低, WLATOUT0/1保持WLATOUT0/1上一次的状态。

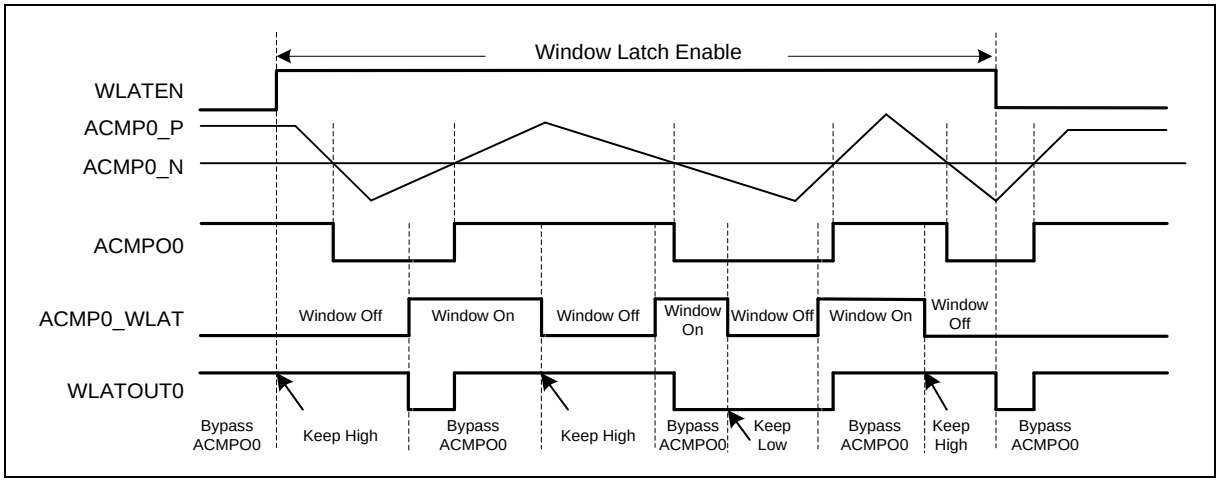


图 6.42-3 窗口锁存模式

6.42.5.3 滤波功能

为了避免比较器输出不稳定的状态，模拟比较器提供了滤波功能。

设定 FILTSEL (ACMP_CTL0[15:13], ACMP_CTL1[15:13]), 模拟比较器的输出会由连续的 PCLKs时钟采样。采样间隔越长，比较器的输出越稳定。但是比较器的敏感性也越低。

图6.42-4为ACMP0的FILTSEL = 3 (4 PCLK)时的例子。所有的结果在输出到ACMPO0前必须保持4 个 PCLK 时钟。如果比较结果短于 4个 PCLK,会被滤掉.

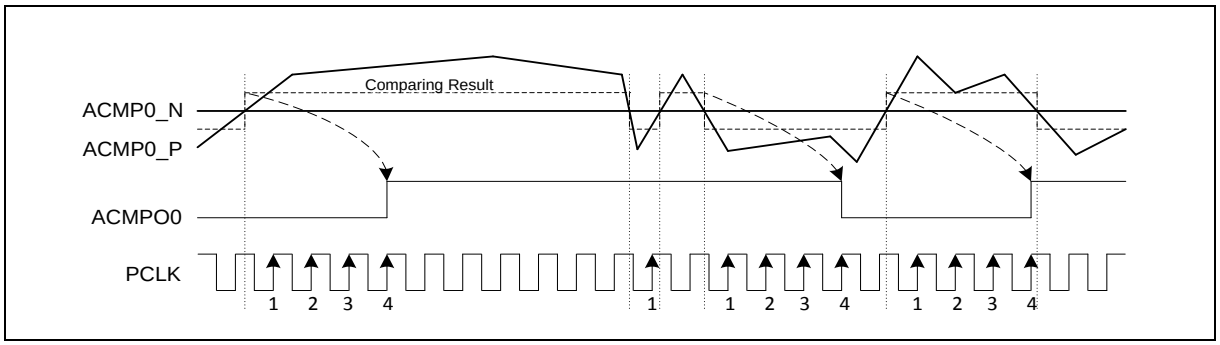


图 6.42-4 滤波功能示例

6.42.5.4 中断源

ACMP0 和 ACMP1 的输出值分别反映在 ACMPO0 (ACMP_STATUS[4]) 和 ACMPO1 (ACMP_STATUS[5])，该值是被窗口锁存和滤波处理过后的值，可以去触发中断。参考图6.42-5, 如果寄存器ACMP_CTL0/1的ACMPIE位设为 1, 将使能中断。如果ACMPO0/1输出改变符合INTPOL (ACMP_CTL0/1[9:8]) 配置的，中断标志 ACMPIF0 (ACMP_STATUS[0]) 或 ACMPIF1 (ACMP_STATUS[1])置 1. 中断标志写1清0。

如果ACMP中断使能并且ACMP唤醒功能使能，系统被ACMP唤醒后，WKIF (ACMP_STATUS[8], ACMP_STATUS[9]) 会置1并且产生中断申请。

图6.42-5展示ACMP的中断源，由 ACMPIE使能或禁止。

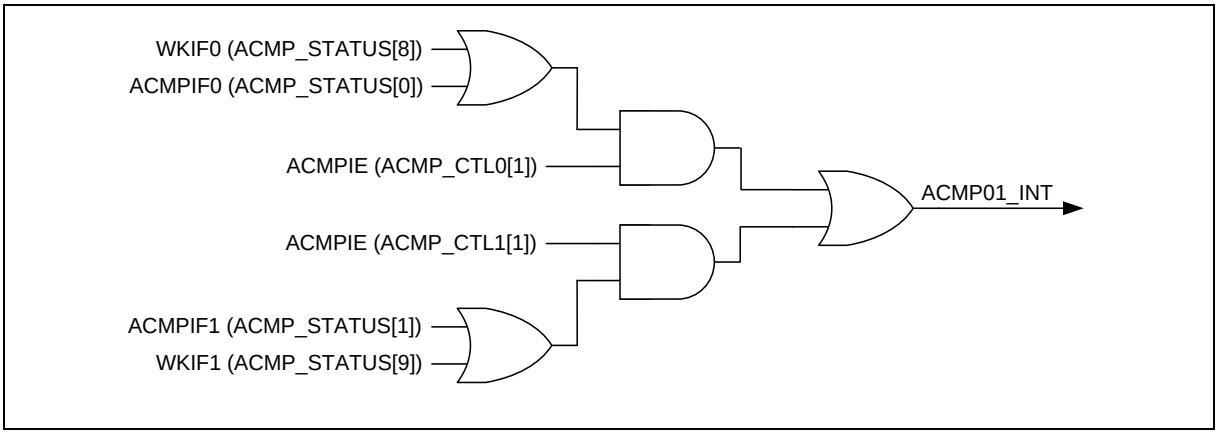


图 6.42-5 比较器控制器中断

6.42.5.5 比较器参考电压(CRV)

CRV 模块包含电阻梯和模拟开关。设置CRVCTL (ACMP_VREF[3:0]) 选择CRV输出电压。配置NEGSEL (ACMP_CTL0[5:4], ACMP_CTL1[5:4])可将此电压连接到比较器负输入端。图6.42-6展示模拟参考电压的框图。

NEGSEL (ACMP_CTL0[5:4], ACMP_CTL1[5:4]) 若没有选择CRV模块，电阻梯会被硬件关闭以降低功耗。电阻梯的参考电压可以通过SYS_VREFCTL 选择AVDD 管脚或内部参考电压。

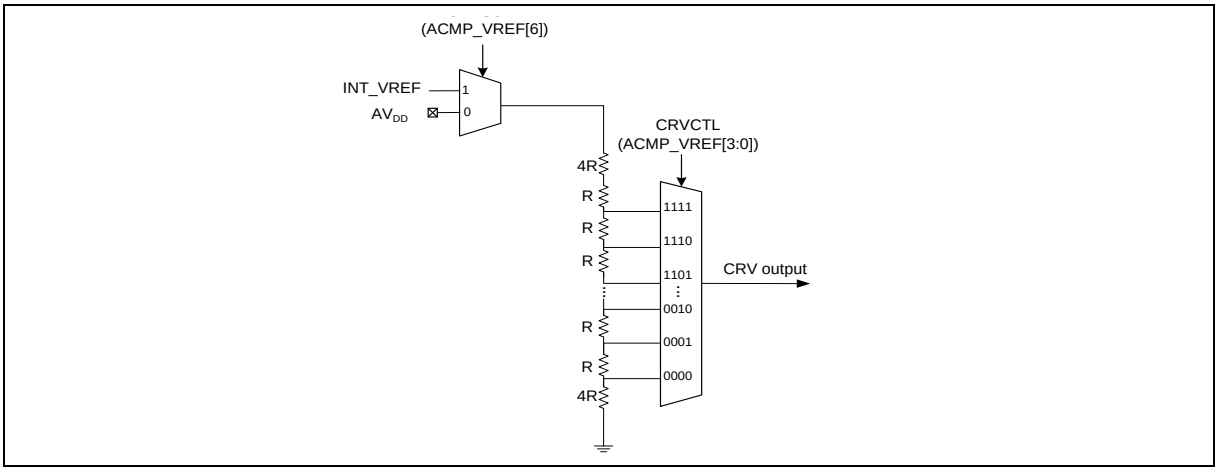


图 6.42-6 比较器参考电压框图

6.42.5.6 窗口比较模式

设置WCMPSEL (ACMP_CTL0/1[18])为1使能窗口比较器模式，可监控某个电压是否在特定范围内。用户可以连接特定的模拟电压源到两个比较器的正输入端或是负输入端。监测电压范围的上、下限取决于两个比较器的另外的输入端。如果一个比较器的输出为低，而另一个比较器的输出为高这就意味着一个比较器监测的是上限，另一个比较器监测的是下限。用户可以通过ACMPWO (ACMP_STATUS[16])直接监控模拟输入电压，如果 ACMPWO 是高, 意味着输入电压在上限和下限的范围之内, 叫做模拟电压在窗口里。

图6.42-7给出窗口比较模式的例子，该例程中窗口比较模式被选中，用户选择每个比较器四路正输入端中的一路，然后在芯片外部将该两组比较器的输入连接到一起。

如果 ACMPS0 输出高且 ACMPS1 输出低, 意味着电压源在上限和下限之内，叫做电压源在窗口里，否则，电压源在窗口外。

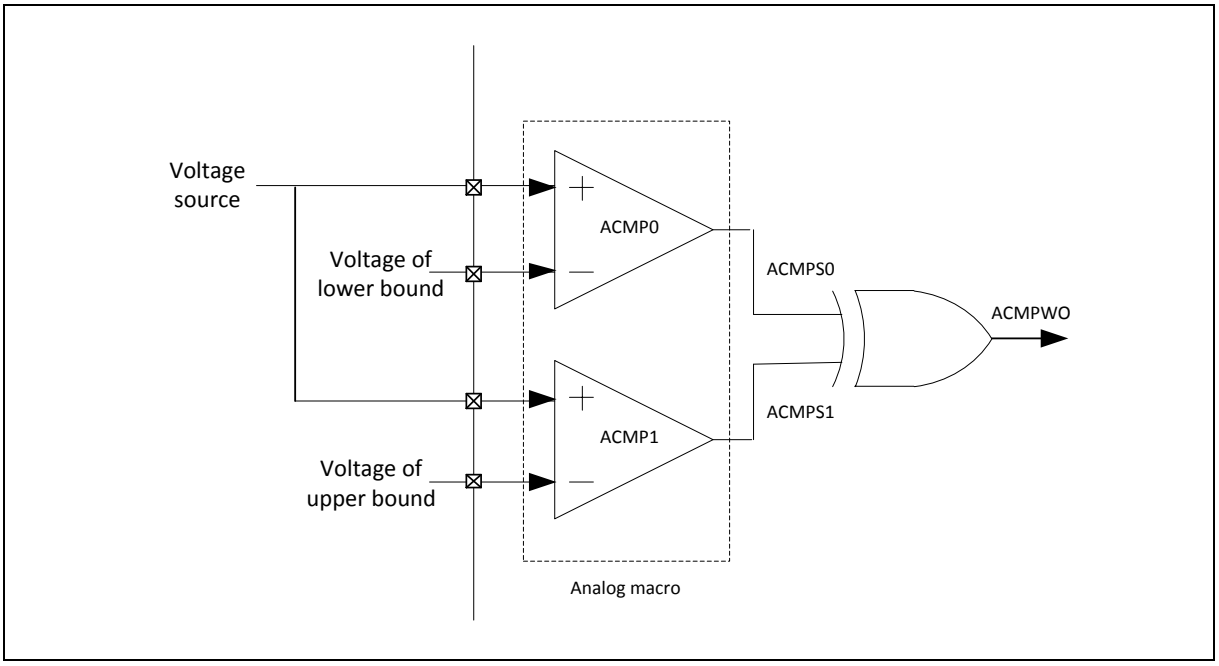


图 6.42-7 窗口比较模式示例

比较器窗口输出 (ACMPWO)在 ACMP_STATUS[16]， 窗口比较逻辑的真值表如表6.42-1.

ACMPS0	ACMPS1	ACMPWO
0	0	0
0	1	1
1	1	0
1	0	1

表 6.42-1 窗口比较逻辑真值表

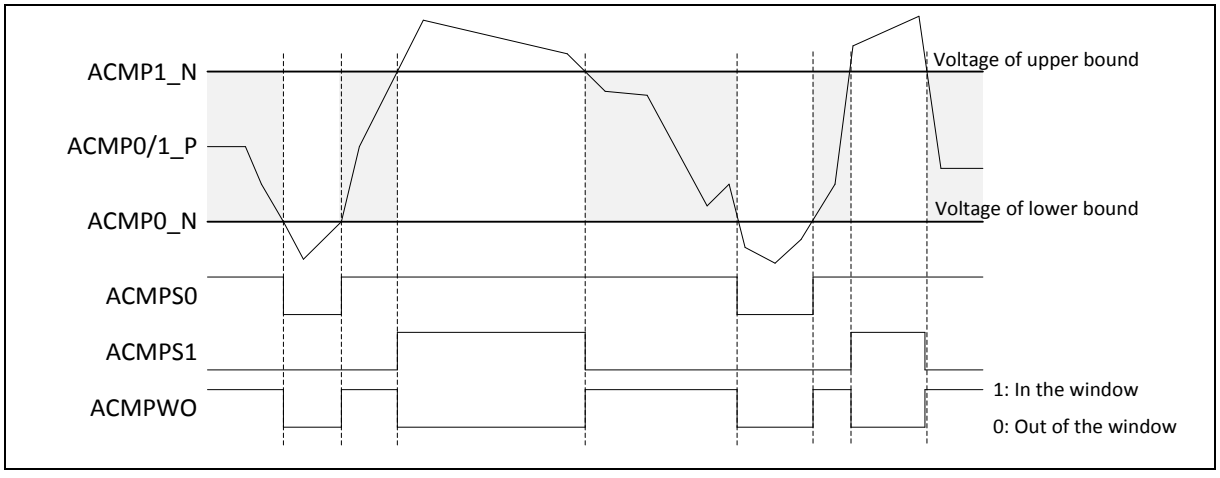


图 6.42-8 窗口比较模式示例

如图6.42-8所示，如果ACMPWO为1，这就意味着正端输入电压在窗口内，否则就是在窗口外。因此ACMPWO可以被用作监控外部模拟管脚的电压转变。更进一步，ACMPWO 也可以用做窗口锁存，过滤功能和ACMP中断.

注意负端输入必须是不同的电压源，否则该功能就没有任何意义。

6.42.6 寄存器映射

R: 只读, **W**: 只写, **R/W**: 可读可写

寄存器	偏移量	R/W	描述	复位值
ACMP 基地址: **ACMP01_BA = 0x4004_5000**				
ACMP_CTL0	ACMP01_BA+0x00	R/W	模拟比较器 0 控制寄存器	0x0000_0000
ACMP_CTL1	ACMP01_BA+0x04	R/W	模拟比较器 1 控制寄存器	0x0000_0000
ACMP_STATUS	ACMP01_BA+0x08	R/W	模拟比较器状态寄存器	0x0000_0000
ACMP_VREF	ACMP01_BA+0x0C	R/W	模拟比较器参考电压控制寄存器	0x0000_0000

6.42.7 寄存器描述

ACMP_CTL0 模拟比较器 0 控制寄存器

寄存器	偏移量	R/W	描述	复位值
ACMP_CTL0	ACMP01_BA+0x00	R/W	模拟比较器 0 控制寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved		MODESEL		Reserved		HYSSEL	
23	22	21	20	19	18	17	16
Reserved					WCMPSEL	WLATEN	WKEN
15	14	13	12	11	10	9	8
FILTSEL			OUTSEL	Reserved		INTPOL	
7	6	5	4	3	2	1	0
POSSEL		NEGSEL		ACMPOINV	Reserved	ACMPIE	ACMPEN

位	描述	
[31:30]	**Reserved**	保留.
[29:28]	**MODESEL**	**传输延时模式选择** 00 = 最大传输延时是 4.5uS, 操作电流是 1.2uA. 01 = 最大传输延时是 2uS, 操作电流是 3uA. 10 = 最大传输延时是 600nS, 操作电流是 10uA. 11 = 最大传输延时是 200nS, 操作电流是 75uA.
[27:26]	**Reserved**	保留.
[25:24]	**HYSSEL**	**迟滞回差电压选择** 00 = 0mV. 01 = 10mV. 10 = 20mV. 11 = 30mV.
[23:19]	**Reserved**	保留
[18]	**WCMPSEL**	**窗口比较模式选择** 0 = 不选择窗口比较模式. 1 = 选择窗口比较模式.
[17]	**WLATEN**	**窗口锁存模式使能** 0 = 窗口锁存模式禁止 1 = 窗口锁存模式使能
[16]	**WKEN**	**掉电唤醒使能位** 0 = 唤醒功能禁止. 1 = 唤醒功能使能
[15:13]	**FILTSEL**	**比较器输出滤波采样时长选择**

		000 = 滤波功能禁止. 001 = ACMP0输出经过 1 个连续的 PCLK 采样. 010 = ACMP0输出经过 2个连续的 PCLKs采样. 011 = ACMP0输出经过 4个连续的PCLKs采样. 100 = ACMP0输出经过 8个连续的PCLKs采样. 101 = ACMP0输出经过 16个连续的PCLKs采样. 110 = ACMP0输出经过 32个连续的PCLKs采样. 111 = ACMP0输出经过 64 个连续的PCLKs采样.
[12]	**OUTSEL**	**比较器滤波输出选择** 0 = 比较器 0 输出到 ACMP0_O管脚不经过滤波 1 = 比较器 0 输出到 ACMP0_O管脚经过滤波
[11:10]	**Reserved**	保留.
[9:8]	**INTPOL**	**中断条件极性选择** ACMPIF0 当下列输出条件被检测到时设为1 00 = 上升沿或下降沿. 01 = 上升沿. 10 = 下降沿. 11 = 保留.
[7:6]	**POSSEL**	**比较器正端输入选择** 00 = 从 ACMP0_P0输入. 01 = 从 ACMP0_P1输入. 10 = 从 ACMP0_P2输入. 11 = 从ACMP0_P3输入.
[5:4]	**NEGSEL**	**比较器负端输入选择** 00 = ACMP0_N 管脚. 01 = 内部比较器参考电压 (CRV). 10 = 带隙电压. 11 = DAC 输出.
[3]	**ACMPOINV**	**比较器输出反相** 0 = 禁止比较器 0 输出反相. 1 = 使能比较器 0 输出反相.
[2]	**Reserved**	保留.
[1]	**ACMPIE**	**比较器中断使能位** 0 = 比较器 0 中断禁止. 1 = 比较器 0 中断使能.如果 WKEN (ACMP_CTL0[16])设为 1, 中断唤醒功能也会使能.
[0]	**ACMPEN**	**比较器使能位** 0 = 比较器 0 禁止. 1 = 比较器 0 使能.

ACMP_CTL1 模拟比较器 1 控制寄存器

寄存器	偏移量	R/W	描述	复位值
ACMP_CTL1	ACMP01_BA+0x04	R/W	模拟比较器 1 控制寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved		MODESEL		Reserved		HYSSEL	
23	22	21	20	19	18	17	16
Reserved					WCMPSEL	WLATEN	WKEN
15	14	13	12	11	10	9	8
FILTSEL			OUTSEL	Reserved		INTPOL	
7	6	5	4	3	2	1	0
POSSEL		NEGSEL		ACMPOINV	Reserved	ACMPIE	ACMPEN

位	描述	
[31:30]	**Reserved**	保留.
[29:28]	**MODESEL**	**传输延时模式选择** 00 = 最大传输延时是 4.5uS, 操作电流是 1.2uA. 01 = 最大传输延时是 2uS, 操作电流是 3uA. 10 = 最大传输延时是 600nS, 操作电流是 10uA. 11 = 最大传输延时是 200nS, 操作电流是 75uA.
[27:26]	**Reserved**	保留.
[25:24]	**HYSSEL**	**迟滞回差电压选择** 00 = 0mV. 01 = 10mV. 10 = 20mV. 11 = 30mV.
[23:19]	**Reserved**	保留.
[18]	**WCMPSEL**	**窗口比较模式选择** 0 = 不选择窗口比较模式. 1 = 选择窗口比较模式.
[17]	**WLATEN**	**窗口锁存模式使能** 0 = 窗口锁存模式禁止 1 = 窗口锁存模式使能
[16]	**WKEN**	**掉电唤醒使能位** 0 = 唤醒功能禁止. 1 = 唤醒功能使能

位	描述	
[15:13]	**FILTSEL**	**比较器输出滤波采样时长选择** 000 = 滤波功能禁止. 001 = ACMP1输出经过 1 个连续的 PCLK 采样. 010 = ACMP1输出经过 2个连续的 PCLKs采样. 011 = ACMP1输出经过 4个连续的PCLKs采样. 100 = ACMP1输出经过 8个连续的PCLKs采样. 101 = ACMP1输出经过 16个连续的PCLKs采样. 110 = ACMP1输出经过 32个连续的PCLKs采样. 111 = ACMP1输出经过 64 c个连续的PCLKs采样.
[12]	**OUTSEL**	**比较器输出选择** 0 = 比较器 1 输出到 ACMP1_O管脚不经过滤波. 1 = 比较器 1 输出到 ACMP1_O管脚经过滤波.
[11:10]	**Reserved**	保留.
[9:8]	**INTPOL**	**中断极性选择** ACMPIF1 当下列输出条件被检测到时设为1 00 = 上升沿或下降沿. 01 = 上升沿. 10 = 下降沿. 11 = 保留.
[7:6]	**POSSEL**	**比较器正输入端选择** 00 = 从 ACMP1_P0输入. 01 = 从 ACMP1_P1输入. 10 = 从 ACMP1_P2输入. 11 = 从ACMP1_P3输入.
[5:4]	**NEGSEL**	**比较器负输入端选择** 00 = ACMP1_N 管脚. 01 = 内部比较器参考电压 (CRV). 10 = 带隙电压. 11 = DAC 输出.
[3]	**ACMPOINV**	**比较器输出反相** 0 = 禁止比较器 1 输出反相. 1 = 使能比较器 1 输出反相.
[2]	**Reserved**	保留.
[1]	**ACMPIE**	**比较器中断使能位** 0 = 比较器 1 中断禁止. 1 = 比较器 1 中断使能.如果 WKEN (ACMP_CTL1[16])设为 1, 中断唤醒功能也会使能.
[0]	**ACMPEN**	**比较器使能位** 0 = 比较器 1 禁止. 1 = 比较器 1 使能.

ACMP_STATUS 模拟比较器状态寄存器

寄存器	偏移量	R/W	描述	复位值
ACMP_STATUS	ACMP01_BA+0x08	R/W	模拟比较器状态寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							ACMPWO
15	14	13	12	11	10	9	8
Reserved		ACMPS1	ACMPS0	Reserved		WKIF1	WKIF0
7	6	5	4	3	2	1	0
Reserved		ACMPO1	ACMPO0	Reserved		ACMPIF1	ACMPIF0

位	描述	
[31:17]	**Reserved**	保留
[16]	**ACMPWO**	**比较器窗口输出** 这一位展示窗口比较模式的输出状态 0 = 正向输入电压在窗口外 1 = 正向输入电压在窗口里
[15:14]	**Reserved**	保留.
[13]	**ACMPS1**	**比较器 1 状态** 允许程序同步于PCLK 读取.，当比较器 1 禁止（ACMPEN (ACMP_CTL1[0])=0）时清0
[12]	**ACMPS0**	**比较器 0 状态** 允许程序同步于PCLK 读取.，当比较器 0 禁止（ACMPEN (ACMP_CTL0[0])=0）时清0
[11:10]	**Reserved**	保留.
[9]	**WKIF1**	保留.
[8]	**WKIF0**	**比较器 1 掉电唤醒中断标志** 当ACMP1 唤醒中断事件发生时，这一位置 1. 0 = 没有掉电唤醒发生. 1 = 发生掉电唤醒. **注意:**此位写1清0.
[7:6]	**Reserved**	**比较器 0 掉电唤醒中断标志** 当ACMP0 唤醒中断事件发生时，这一位置 1. 0 = 没有掉电唤醒发生. 1 = 发生掉电唤醒. **注意:**此位写1清0.
[5]	**ACMPO1**	**比较器 1 输出**

位	描述	
		允许程序同步于PCLK 读取.，当比较器 1 禁止（ACMPEN (ACMP_CTL1[0])=0）时清0
[4]	**ACMPO0**	**比较器 0 输出** 允许程序同步于PCLK 读取.，当比较器 0 禁止（ACMPEN (ACMP_CTL0[0])=0）时清0
[3:2]	**Reserved**	保留.
[1]	**ACMPIF1**	**比较器 1 中断标志** 当比较器1的输出检测到INTPOL (ACMP_CTL1[9:8])定义的条件成立，此位由硬件置1，当ACMPIE (ACMP_CTL1[1])为1会产生中断. **注意:**此位写1清0..
[0]	**ACMPIF0**	**比较器 0 中断标志** 当比较器0的输出检测到INTPOL (ACMP_CTL0[9:8])定义的条件成立，此位由硬件置1，当ACMPIE (ACMP_CTL0[1])为1会产生中断. **注意:**此位写1清0..

ACMP_VREF 模拟比较器参考电压控制寄存器

寄存器	偏移量	R/W	描述	复位值
ACMP_VREF	ACMP01_BA+0x0C	R/W	模拟比较器参考电压控制寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved	CRVSSEL	Reserved		CRVCTL			

位	描述	
[31:7]	Reserved	保留.
[6]	CRVSSEL	CRV 电压源选择 0 = V_{DDA}选择作为 CRV电压源. 1 = SYS_VREFCTL 寄存器选择的参考电压作为 CRV 电压源.
[5:4]	Reserved	保留.
[3:0]	CRVCTL	比较器参考电压设置 CRV = CRV 电压源 * (1/6+CRVCTL/24).

6.43 OPA 运算放大器

6.43.1 概述

芯片集成3个运算放大器。用户依据不同的应用独立的使能它们。为了测量需求有一个运算放大器输出和ADC的通道相连。运算放大器电路增益可编程。

6.43.2 特性

- 模拟电压输入范围: 0~V_{DD}
- 3个运算放大器
- 施密特输出可做基本比较器功能
- 施密特输出可触发中断

6.43.3 框图

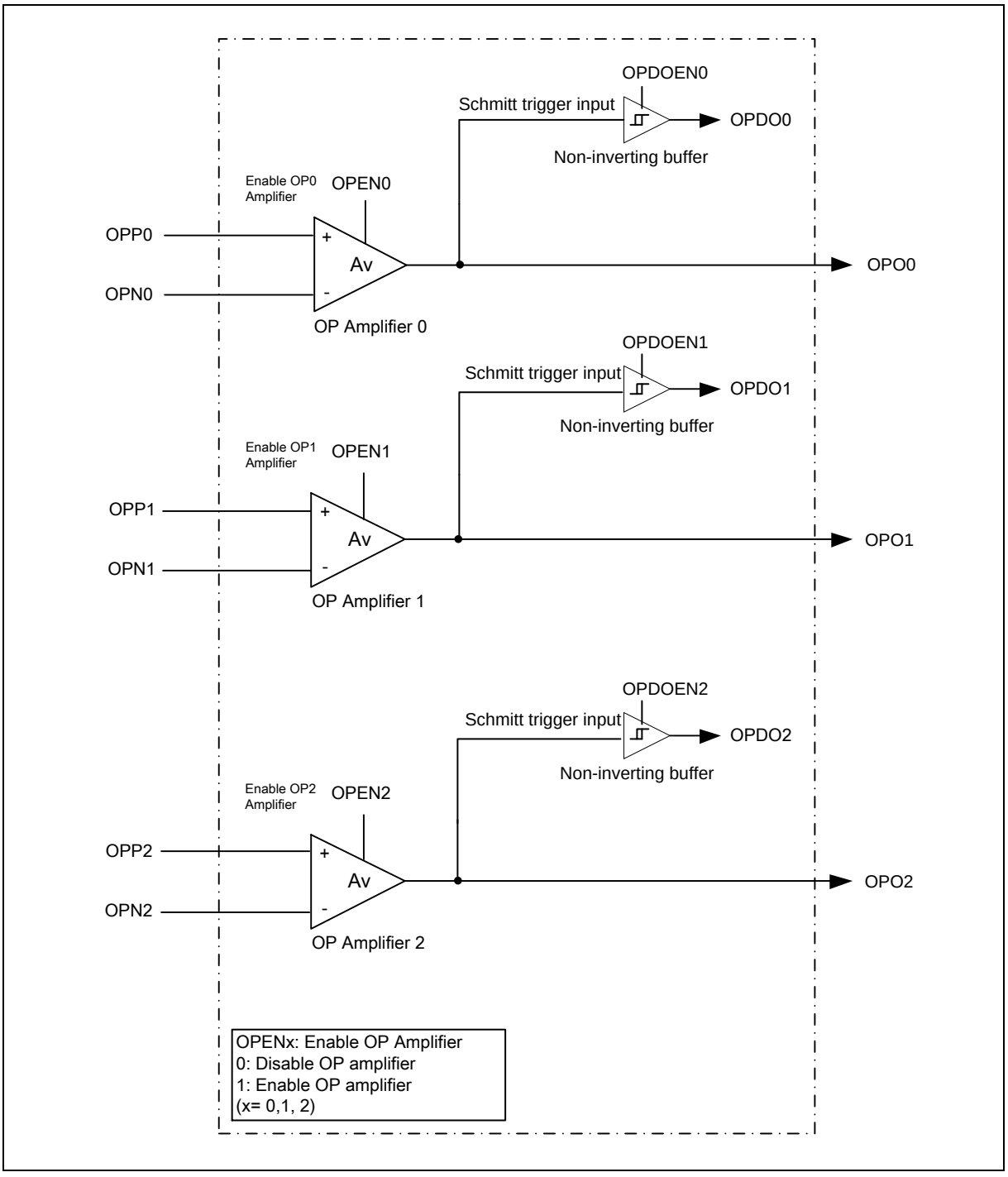


图 6.43-1 运算放大器框图

6.43.4 基本配置

6.43.4.1 OPA0 基本配置

- 时钟源配置
  - 使能 OPA0 外设时钟在OPACKEN (CLK_APBCLK1[30])

- 复位配置
  - 复位OPA0 控制在OPARST (SYS_IPRST2[30])
- 管脚配置

组	管脚名称	GPIO	MFP
OPA0	OPA0_N	PB.1	MFP1
	OPA0_O	PB.2	MFP1
	OPA0_P	PB.0	MFP1

6.43.4.2 OPA1基本配置

- 时钟源配置
  - 使能 OPA1外设时钟在OPACKEN (CLK_APBCLK1[30])
- 复位配置
  - 复位OPA1 控制在OPARST (SYS_IPRST2[30])
- 管脚配置

组	管脚名称	GPIO	MFP
OPA1	OPA1_N	PA.9	MFP1
	OPA1_O	PA.10	MFP1
	OPA1_P	PA.8	MFP1

6.43.4.3 OPA2 基本配置

- 时钟源配置
  - 使能 OPA2 外设时钟在OPACKEN (CLK_APBCLK1[30]).
- 复位配置
  - 复位OPA2 控制在OPARST (SYS_IPRST2[30])
- 管脚配置

组	管脚名称	GPIO	MFP
OPA2	OPA2_N	PD.11	MFP1
	OPA2_O	PD.12	MFP1
	OPA2_P	PD.10	MFP1

6.43.5 功能描述

6.43.5.1 OP 放大器功能

运放通过 OPENx (OPA_CTL[1:0]) 位使能，x=0, 1, 2 对应 OPA0, OPA1, OPA2 。OPA 管脚功能参考图6.43-1。为了测量OPA0的输出同时也内部连接到ADC通道。设置OPDOENx (OPA_CTL[6:4])使能施密特触发。施密特输出使能并且OP输出改变时，数字输出OPDOx (OPA_STATUS[2:0])置1。施密特输出禁止时, 总是输出0。

6.43.5.2 校准功能

OP 放大器的增益可调为 2, 4, 8倍... .该电路有五个校准位，用于校准失调电压。校准可以通过设置 CALTRGx (OPA_CALCTL[2:0])启动。用户可以根据应用设置 CALCLKx (OPA_CALCTL[9:4]) 选择校准时钟， x=0, 1, 2。校准之后, 输入失调电压可以降到 ±1.6mV以内。读 OPA_CALST 寄存器可以监控校准状态，包含校准完成状态， NMOS, 和 PMOS 校准结果状态。轨到轨共模输入范围是通过使用 NMOS 和 PMOS差分对并联连接来实现的。

6.43.5.3 中断源

当施密特输出中断 OPDOIEN0 (OPA_CTL[8]), OPDOIEN1 (OPA_CTL[9]), 和 OPDOIEN2 (OPA_CTL[10]) 使能时，如果 OPA0, 1, 2 施密特输出发生改变，中断标志 OPDOIF0 (OPA_STATUS[4]) , OPDOIF1 (OPA_STATUS[5]) 和OPDOIF2 (OPA_STATUS[6])会置1。这些标志写1清0，施密特输出可以作为比较器中断源。

6.43.6 寄存器映射

R: 只读, **W**: 只写, **R/W**: 可读可写, **C**:只能写0

寄存器	偏移量	R/W	描述	复位值
OPA 基地址: **OPA_BA = 0x4004_6000**				
OPA_CTL	OPA_BA+0x00	R/W	OP 放大器控制寄存器	0x0000_0000
OPA_STATUS	OPA_BA+0x04	R/W	OP 放大器状态寄存器	0x0000_0000
OPA_CALCTL	OPA_BA+0x08	R/W	OP 放大器校准控制寄存器	0x0000_0000
OPA_CALST	OPA_BA+0x0C	R	OP 放大器校准状态寄存器	0x0000_0000

6.43.7 寄存器描述

OPA_CTL OPA 控制寄存器

寄存器	偏移量	R/W	描述	复位值
OPA_CTL	OPA_BA+0x00	R/W	OP放大器控制寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved					OPDOIEN2	OPDOIEN1	OPDOIEN0
7	6	5	4	3	2	1	0
Reserved	OPDOEN2	OPDOEN1	OPDOEN0	Reserved	OPEN2	OPEN1	OPEN0

位	描述	
[31:11]	**Reserved**	保留.
[10]	**OPDOIEN2**	**运放 2 施密特输出中断使能位** 0 = 禁止运放 2 施密特输出中断. 1 = 使能运放 2 施密特输出中断 当运放2施密特输出改变时，中断标志OPDOIF2置1，就是说 OPDOIEN2 为 1, 会产生比较器中断请求
[9]	**OPDOIEN1**	**运放 1 施密特输出中断使能位** 0 = 禁止运放 1 施密特输出中断. 1 = 使能运放 1 施密特输出中断 当运放1施密特输出改变时，中断标志OPDOIF1置1，就是说 OPDOIEN1 为 1, 会产生比较器中断请求
[8]	**OPDOIEN0**	**运放 0 施密特输出中断使能位** 0 = 禁止运放 0施密特输出中断. 1 = 使能运放 0 施密特输出中断 当运放0施密特输出改变时，中断标志OPDOIF0置1，就是说 OPDOIEN0 为 1, 会产生比较器中断请求
[7]	**Reserved**	保留.
[6]	**OPDOEN2**	**运放2 施密特输出使能位** 0 = 禁止OP2 施密特输出. 1 = 使能OP2 施密特输出.
[5]	**OPDOEN1**	**运放1 施密特输出使能位** 0 = 禁止OP1施密特输出. 1 = 使能OP1 施密特输出.

位	描述	
[4]	**OPDOEN0**	**运放2 施密特输出使能位** 0 = 禁止OP0 施密特输出. 1 = 使能OP0 施密特输出.
[3]	**Reserved**	保留.
[2]	**OPEN2**	**放大器 2 使能位** 0 = OP 放大器 2 禁止. 1 = OP 放大器 2 使能. **注意:** OP 放大器 2 在OPEN2置1后需要等待 20μs输出稳定.
[1]	**OPEN1**	**放大器 1 使能位** 0 = OP 放大器 1 禁止. 1 = OP 放大器 1 使能. **注意:** OP 放大器 1 在OPEN1置1后需要等待 20μs输出稳定.
[0]	**OPEN0**	**放大器 0 使能位** 0 = OP 放大器 0 禁止. 1 = OP 放大器 0 使能. **注意:** OP 放大器 0 在OPEN0置1后需要等待 20μs输出稳定.

OPA_STATUS OPA 状态寄存器

寄存器	偏移量	R/W	描述	复位值
OPA_STATUS	OPA_BA+0x04	R/W	OP 放大器状态寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved	OPDOIF2	OPDOIF1	OPDOIF0	Reserved	OPDO2	OPDO1	OPDO0

位	描述	
[31:7]	Reserved	保留
[6]	OPDOIF2	**OP2 施密特输出中断标志** 在OP2施密特输出改变时置1。此位写1清0.
[5]	OPDOIF1	**OP1 施密特输出中断标志** 在OP1施密特输出改变时置1。此位写1清0.
[4]	OPDOIF0	**OP0 施密特输出中断标志** 在OP0施密特输出改变时置1。此位写1清0.
[3]	Reserved	保留.
[2]	OPDO2	**OP2施密特输出** 施密特输出禁止时(OPDOEN2 = 0) 为 0.
[1]	OPDO1	**OP1施密特输出** 施密特输出禁止时(OPDOEN1 = 0) 为 0 .
[0]	OPDO0	**OP0施密特输出** 施密特输出禁止时(OPDOEN0 = 0) 为 0

OPA_CALCTL OPA 校准控制寄存器

寄存器	偏移量	R/W	描述	复位值
OPA_CALCTL	OPA_BA+0x08	R/W	OP 放大器校准控制寄存器	0x0000_03F0

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved					CALRVS2	CALRVS1	CALRVS0
15	14	13	12	11	10	9	8
Reserved						CALCLK2	
7	6	5	4	3	2	1	0
CALCLK1		CALCLK0		Reserved	CALTRG2	CALTRG1	CALTRG0

位	描述	
[31:19]	**Reserved**	保留.
[18]	**CALRVS2**	**OPA2 校准参考电压选择** 0 = VREF 是$\frac{1}{2}$ AVDD. 1 = VREF 从高 vcm 到低 vcm.
[17]	**CALRVS1**	**OPA1 校准参考电压选择** 0 = VREF 是$\frac{1}{2}$ AVDD. 1 = VREF 从高 vcm 到低 vcm.
[16]	**CALRVS0**	**OPA0 校准参考电压选择** 0 = VREF 是$\frac{1}{2}$ AVDD. 1 = VREF 从高 vcm 到低 vcm.
[15:10]	**Reserved**	保留.
[9:8]	**CALCLK2**	**OPA2** 校准时钟速率选择 00 = 1KHz 01 = 保留 10 = 保留 11 = 保留
[7:6]	**CALCLK1**	**OPA1** 校准时钟速率选择 00 = 1KHz 01 = 保留 10 = 保留 11 = 保留

位	描述	
[5:4]	**CALCLK0**	**OPA0** 校准时钟速率选择 00 = 1KHz 01 = 保留 10 = 保留 11 = 保留
[3]	**Reserved**	保留.
[2]	**CALTRG2**	**OP 放大器 2校准触发位** 0 = 停止校准 1 = 触发校准 **注1:**在使能该位前，OPEN2应该被置位并且内部HIRC应该提前使能。 **注2:**校准结束后硬件自动清除该位
[1]	**CALTRG1**	**OP 放大器 1校准触发位** 0 = 停止校准 1 = 触发校准 **注1:**在使能该位前，OPEN1应该被置位并且内部HIRC应该提前使能。 **注2:**校准结束后硬件自动清除该位
[0]	**CALTRG0**	**OP 放大器 0校准触发位** 0 = 停止校准 1 = 触发校准 **注1:**在使能该位前，OPEN0应该被置位并且内部HIRC应该提前使能。 **注2:**校准结束后硬件自动清除该位

OPA_CALST OPA 校准状态寄存器

寄存器	偏移量	R/W	描述	复位值
OPA_CALST	OPA_BA+0x0C	R	OP 放大器校准状态寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved					CALPS2	CALNS2	DONE2
7	6	5	4	3	2	1	0
Reserved	CALPS1	CALNS1	DONE1	Reserved	CALPS0	CALNS0	DONE0

位	描述	
[31:11]	**Reserved**	保留.
[10]	**CALPS2**	**OP2 PMOS校准结果** 0 = 通过. 1 = 失败.
[9]	**CALNS2**	**OP2 NMOS校准结果** 0 = 通过. 1 = 失败.
[8]	**DONE2**	**OP2 校准完成状态** 0 = 正在校准. 1 = 校准完成.
[7]	**Reserved**	保留.
[6]	**CALPS1**	**OP1 PMOS校准结果** 0 = 通过. 1 = 失败.
[5]	**CALNS1**	**OP1 NMOS校准结果** 0 = 通过. 1 = 失败.
[4]	**DONE1**	**OP1 校准完成状态** 0 = 正在校准. 1 = 校准完成.
[3]	**Reserved**	保留.

位	描述	
[2]	**CALPS0**	**OP0 PMOS校准结果** 0 = 通过. 1 = 失败.
[1]	**CALNS0**	**OP0 NMOS校准结果** 0 = 通过. 1 = 失败.
[0]	**DONE0**	**OP0 校准完成状态** 0 = 正在校准. 1 = 校准完成.

6.44 外设互动

6.44.1 概述

一些外设可以互动，可以在CPU不介入的情况下实现外设自动通信或同步动作。外设互动不需CPU介入，节省CPU资源降低功耗，动作反应快没有软件延时。

6.44.2 外设互动矩阵表

源外设	目的外设									
	BPWM	DAC	EADC	ECAP	EPWM	HIRC	IRCTRIM	RC48M	TIMERPWM	QEI
ACMP	-	-	-	-	3	-	-	-	3,6	
BOD	-	-	-	-	3	-	-	-	3	
BPWM	4	-	1	-	4	-	-	-	-	
Clock Fail	-	-	-	-	3	-	-	-	3	
CPU Lockup	-	-	-	-	3	-	-	-	3	
EADC	-	-	-	-	3	-	-	-	-	
EPWM	4	1	1	-	4	-	-	-	-	
IRCTRIM	-	-	-	-	-	2	-	2	-	
LIRC	-	-	-	-	-	-	-	-	6	
LXT	-	-	-	-	-	-	2	-	-	
QEI	-	-	-	8	-	-	-	-	-	
SRAM	-	-	-	-	3	-	-	-	3	
TIMERPWM	5	1	1	-	5	-	-	-	7	9
USB11Device	-	-	-	-	-	-	2	-	-	

表6.44-1外设互动矩阵表

6.44.3 功能描述

6.44.3.1 从 EPWM, TIMER 到 EADC/DAC

EPWM 触发 EADC 转换

EPWM 可以作为EADC转换触发源

设置EPWM触发为EADC外部硬件触发输入源，6.40.5.8.章节由详细描述

EPWM 触发 DAC 转换

EPWM也可以用来触发DAC转换

设置EPWM触发DAC在章节6.41.5.6.有描述

详细EPWM触发条件在章节6.12.5.27有描述

BPWM 触发 EADC 转换

BPWM可以作为EADC触发源之一

BPWM触发EADC在章节6.40.5.8.描述

BPWM触发条件在章节6.13.5.16描述

定时器触发 EADC 转换

定时器0-3可以作为EADC的触发源之一，当定时器的值匹配比较值或当TMx_EXT管脚变化满足设定条件，定时器将触发EADC开始转换

设置定时器作为EADC的触发源，在章节6.40.5.9描述

定时器触发 DAC 转换

设置定时器为DAC的触发源，在章节6.41.5.6描述

定时器触发条件在章节6.8.5.10.描述

6.44.3.2 从LXT及USB1.1设备到HIRC TRM & RC 48MHz

使用 LXT 或USB SOF 信号自动校准 HIRC

此芯片支持自动校准功能： HIRC(12MHz RC振荡器)及48MHz振荡器可以通过外部32.768 kHz晶振或USB SOF信号自动校准到全温度范围0.25 %的精度

详细的HIRC校准设定在章节6.2.9描述

6.44.3.3 从 ACMP, BOD, Clock 失效,SRAM 校验错误及CPU 锁死到 EPWM/ TIMERPWM

EPWM 刹车源

EPWM刹车源可以是ACMP0/1_O输出信号，EADC结果监控或几个系统失效条件，包括时钟失效，掉电检测，内核锁死，SRAM校验错误。当系统失效时，EPWM刹车信号产生，EPWM输出将停止并保持在可以保护EPWM控制的电源器件状态

详细EPWM刹车设置在章节6.12.5.23.描述

定时器 PWM 刹车源

定时器PWM刹车源可以是ACMP0/1_O输出信号，EADC结果监控或几个系统失效条件，包括时钟失效，掉电检测，内核锁死，SRAM校验错误。当系统失效时，定时器PWM刹车信号产生，定时器PWM输出将停止并保持在可以保护定时器PWM控制的电源器件状态

详细的定时器PWM刹车功能设置，请参考6.8.6.17章节

6.44.3.4 从 EPWM/ BPWM 到 EPWM/ BPWM

EPWM 同步开始功能

选择EPWM0或EPWM1或 BPWM0或 BPWM1为EPWM同步源，打开EPWM的同步功能并设置CNTSEN(EPWM_SSTRG[0])，所选择的EPWM通道将会与同步源同时开始计数。

EPWM 同步开始功能在章节6.12.5.19.描述

BPWM 同步开始功能

选择EPWM0或EPWM1或 BPWM0或 BPWM1为BPWM同步源，打开BPWM的同步功能并设置CNTSEN(BPWM_SSTRG[0])，所选择的BPWM通道将会与同步源同时开始计数。

BPWM 同步开始功能在章节6.13.5.11描述

6.44.3.5 从 TIMER到 EPWM/BPWM

定时器产生触发脉冲作为 EPWM 外部时钟源

定时器0-3可以产生触发脉冲作为EPWM/BPWM外部时钟源

当定时器计数值等于比较值或者当TMx_EXT管脚信号变化满足设置，定时器可以产生触发脉冲，具体描述请参考章节6.8.5.9

EPWM/BPWM时钟源设置请参考6.12.3 / 6.13.3.

6.44.3.6 从 ACMP 和 LIRC 到定时器捕获功能

测量 ACMP0/1 输出信号间隔或 LIRC 时钟速度

设置ACMP0/1输出信号或LIRC时钟作为定时器的捕获信号可以测量信号频率。用户软件可以用捕获的结果来校准LIRC或计算ACMP0/1输出的脉冲宽度

定时器捕获的详细描述在6.8.5.8和6.8.5.9章节

6.44.3.7 从 Timer0/2 到 Timer1/3

定时器间触发捕获模式

定时器0/2设置为外部计数模式，计数外部脉冲，并且可以产生内部信号(INTR_TMR_TRG)触发Timer1/3开始或停止计数。

详细的定时器间触发捕获模式在章节6.8.5.11描述

6.44.3.8 从 QEI 到 ECAP

ECAP 输入滤波

ECAP输入滤波架构类似QEI，有6个采样率选择，支持宽范围的滤波，保证信号的采样

详细设置请参考章节6.14.5.1

6.44.3.9 从 TIMER 到 QEI

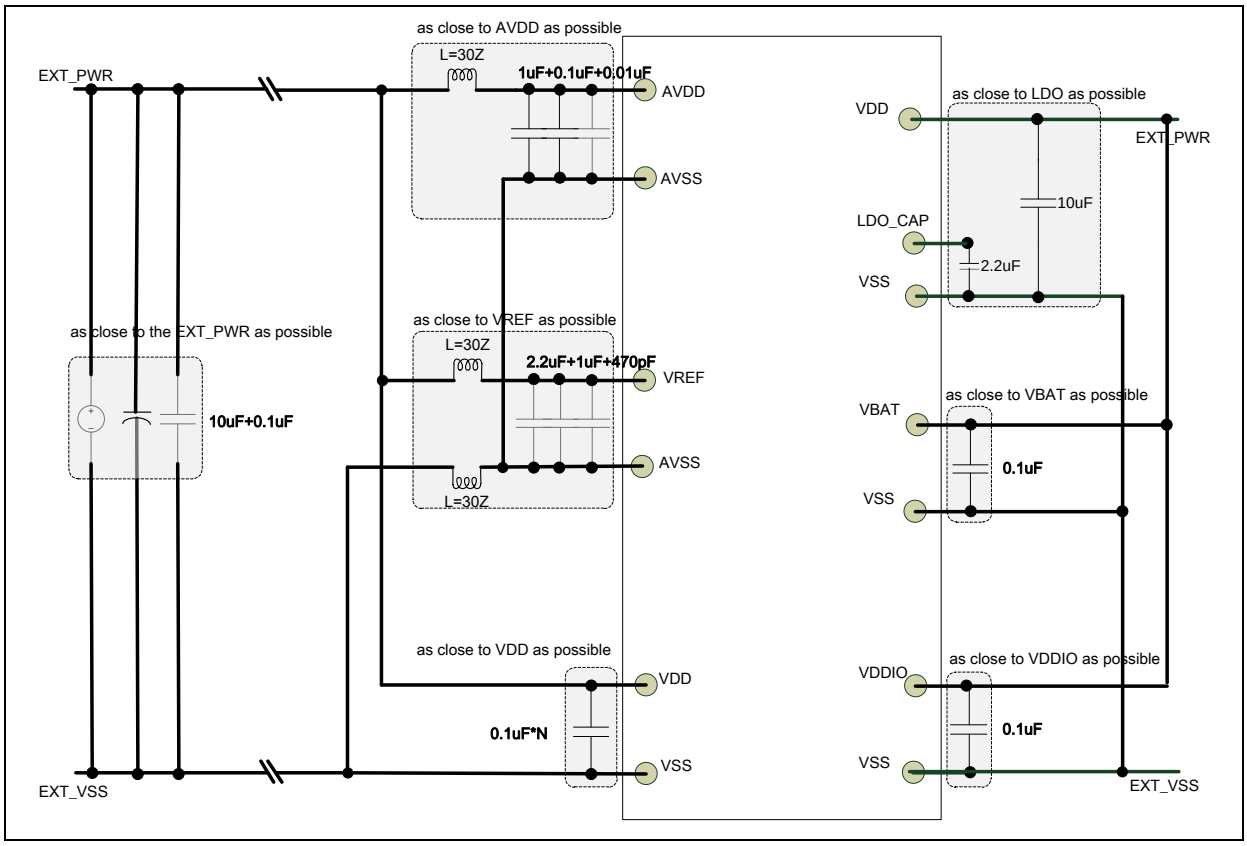
TIMER TIF 事件到 QEI

当QEI位HOLDCNT(QEI_CTL[24])置位，CNT(QEI_CNT[31:0])的值将被锁存到QEI计数保存寄存器CNTHOLD(QEI_CNTHOLD[31:0])，这个值将一直保持到下一个HOLDCNT (QEI_CTL[24])触发到来。HOLDCNT位可以通过写1置位也可以由定时器中断标志TIF (TIMERx_INTSTS[0])的上升沿置位，具体的描述在章节6.14.5.11

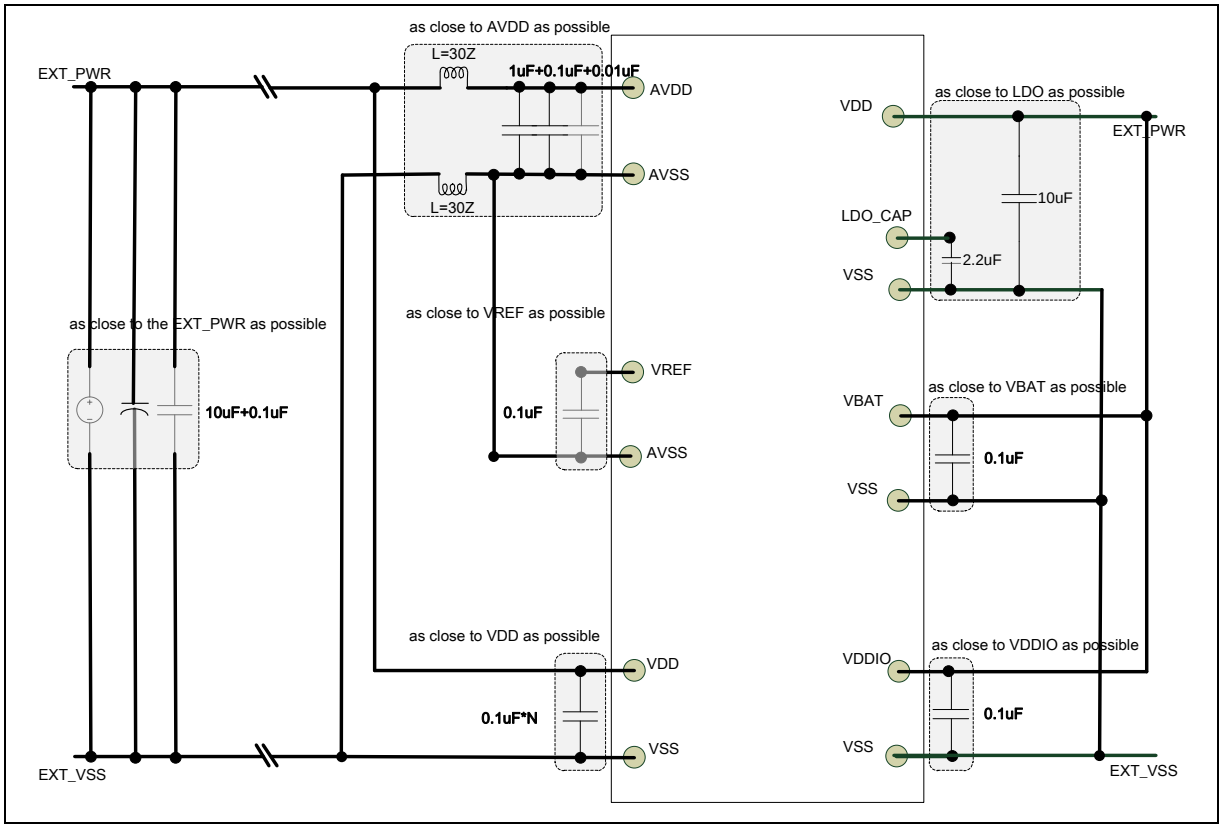
详细设置请参考章节6.8.5.1

7 应用电路

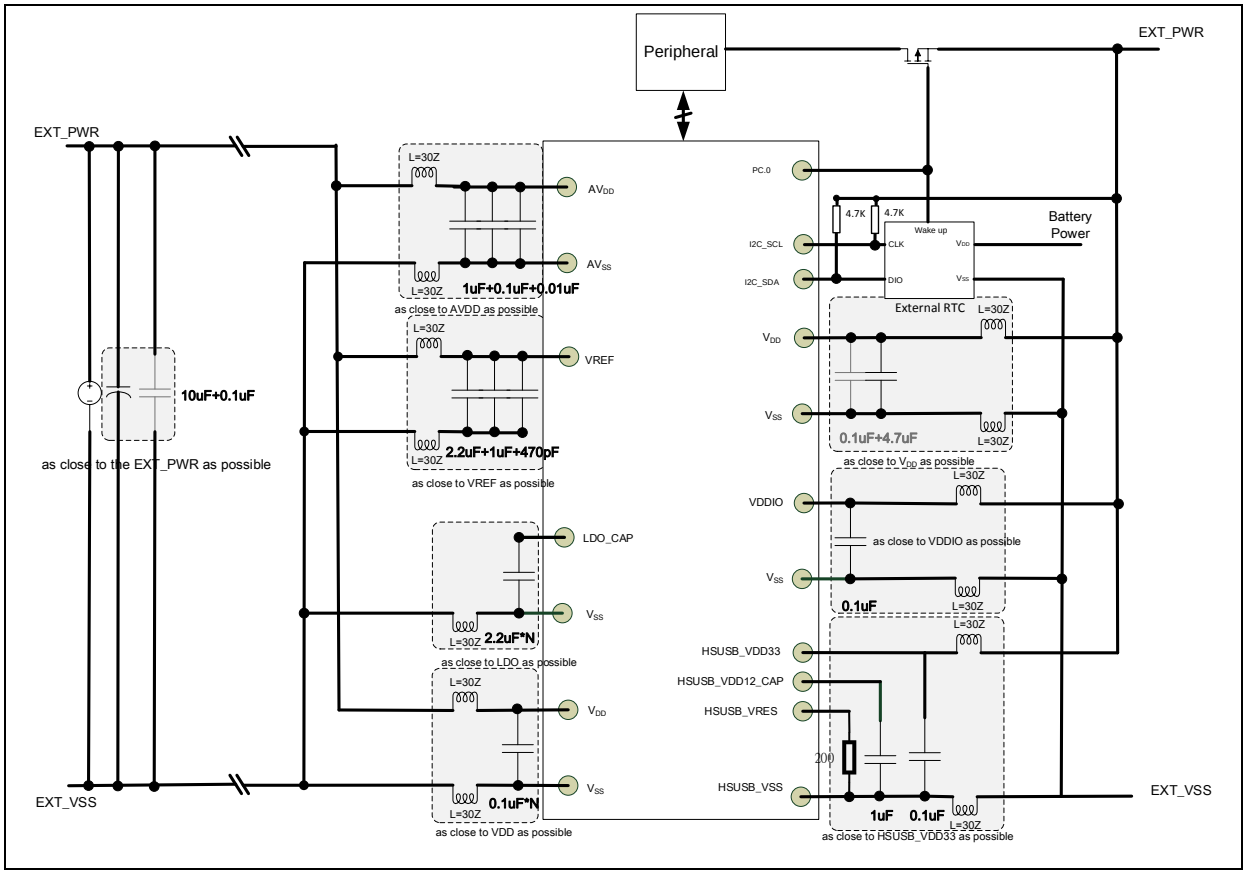
7.1 电源供电及外部V_{REF}



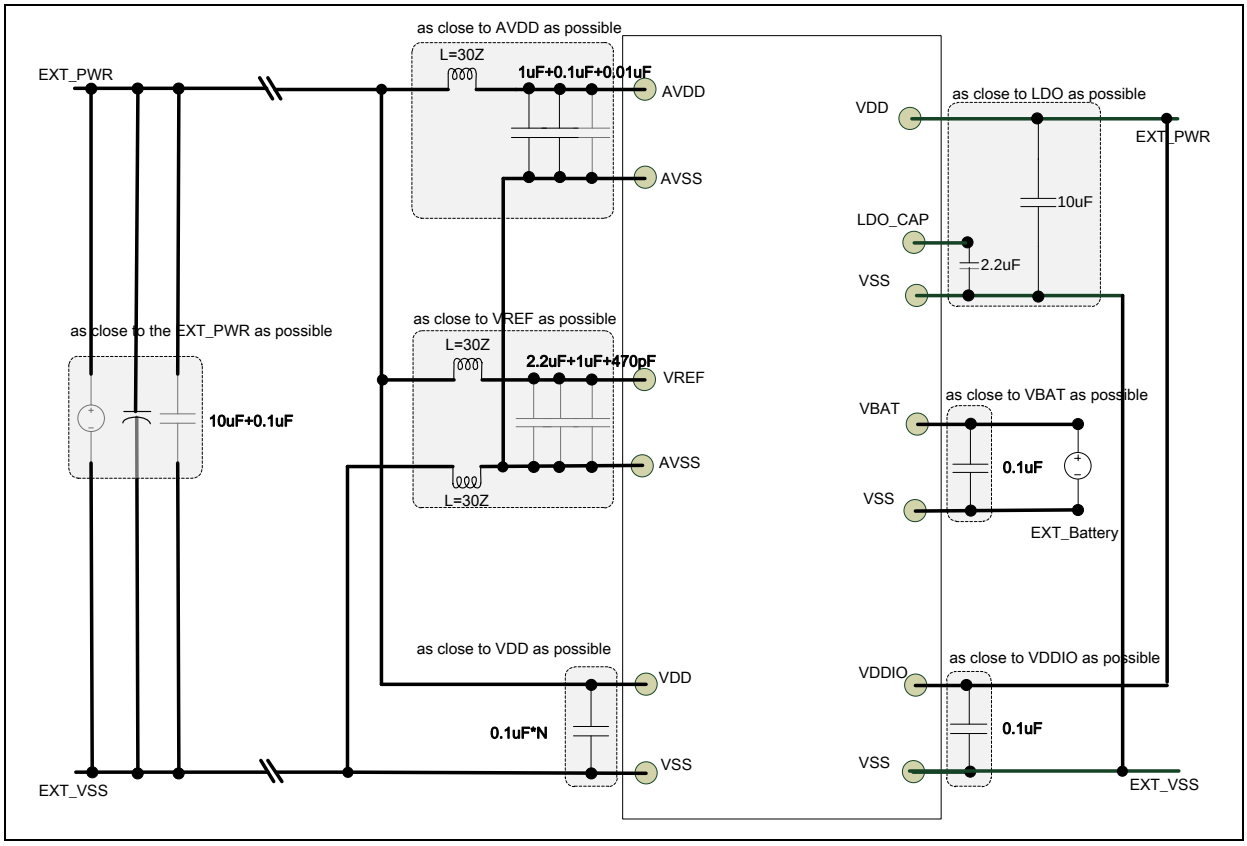
7.2 电源供电及内部 V_{REF}



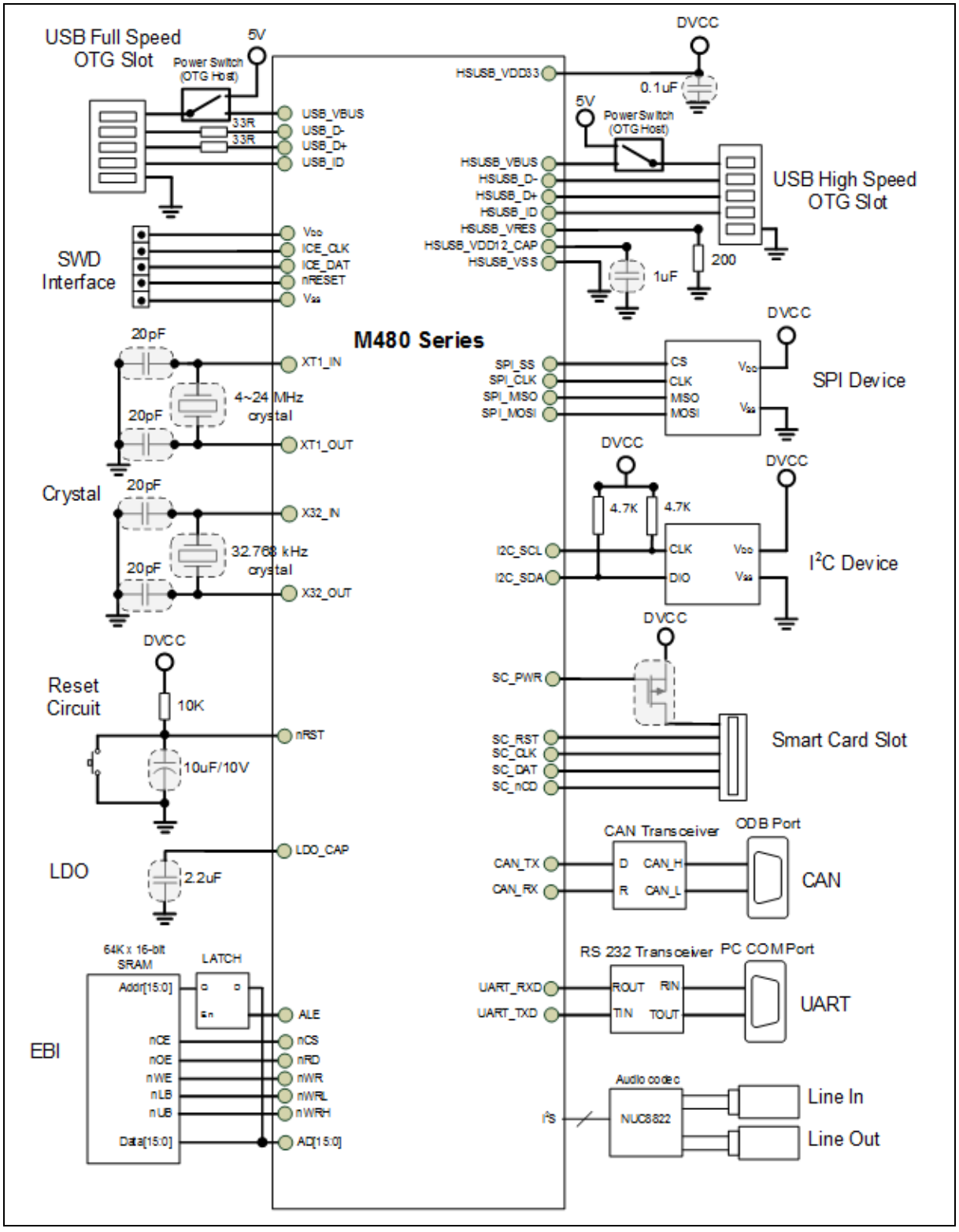
7.3 电源和 V_{REF}以及外部RTC使用电池电源



7.4 电源和 V_{REF}以及内部RTC使用电池电源



7.5 外设应用电路



*注: 使用USB或USB HS 不带 OTG时 USB_ID, HSUSB _ID 可以浮空

8 电气特性

请参考相关技术手册了解M480系列的详细电气特性信息

9 缩写

9.1 缩写

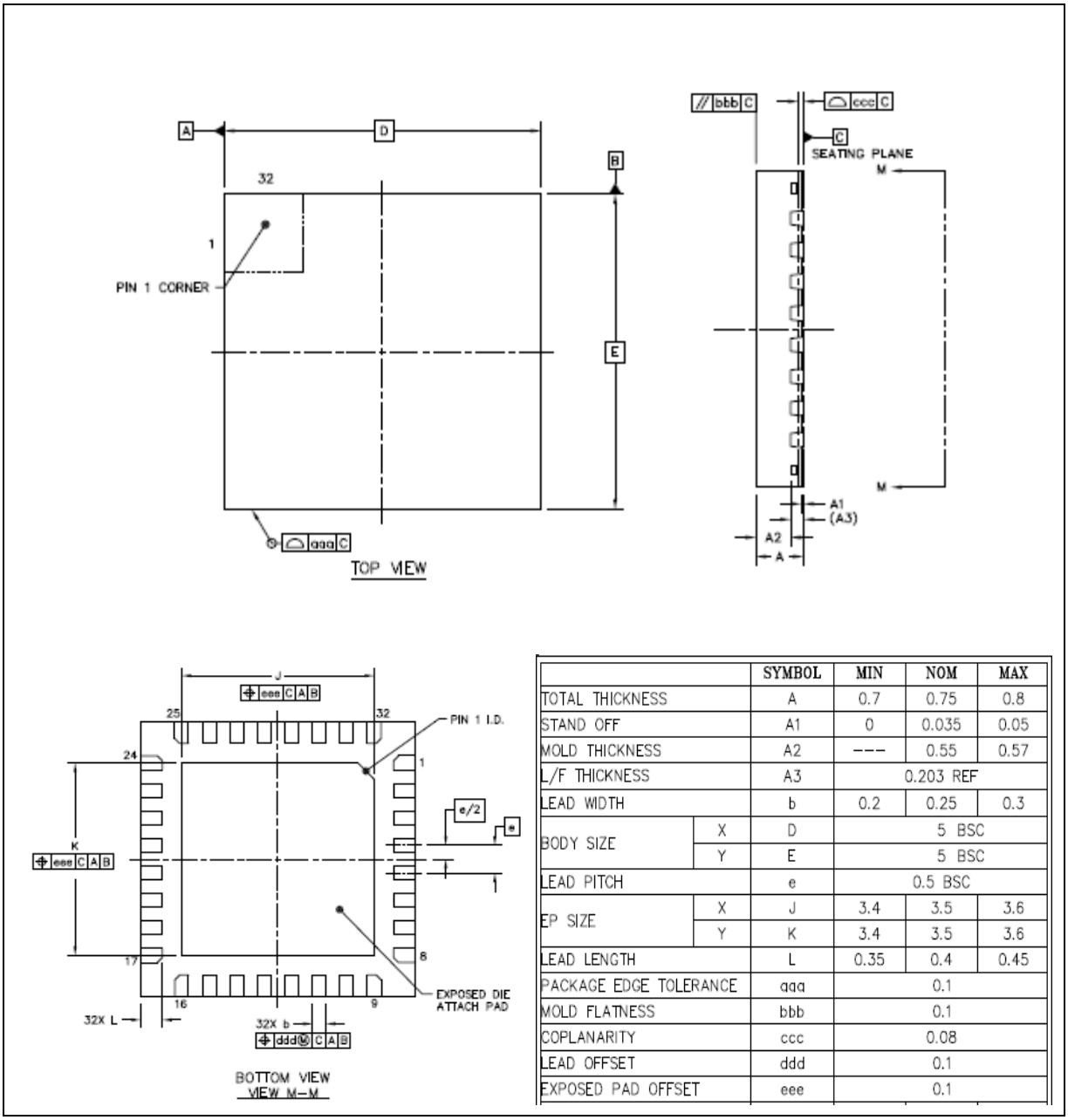
首字母缩写	描述
ACMP	Analog Comparator Controller
ADC	Analog-to-Digital Converter
AES	Advanced Encryption Standard
APB	Advanced Peripheral Bus
AHB	Advanced High-Performance Bus
BOD	Brown-out Detection
CAN	Controller Area Network
CCAP	Camera Capture Interface
DAP	Debug Access Port
DES	Data Encryption Standard
EADC	Enhanced Analog-to-Digital Converter
EBI	External Bus Interface
EMAC	Ethernet MAC Controller
EPWM	Enhanced Pulse Width Modulation
FIFO	First In, First Out
FMC	Flash Memory Controller
FPU	Floating-point Unit
GPIO	General-Purpose Input/Output
HCLK	The Clock of Advanced High-Performance Bus
HIRC	12 MHz Internal High Speed RC Oscillator
HXT	4~24 MHz External High Speed Crystal Oscillator
IAP	In Application Programming
ICP	In Circuit Programming
ISP	In System Programming
LDO	Low Dropout Regulator
LIN	Local Interconnect Network
LIRC	10 kHz internal low speed RC oscillator (LIRC)
MPU	Memory Protection Unit
NVIC	Nested Vectored Interrupt Controller
PCLK	The Clock of Advanced Peripheral Bus
PDMA	Peripheral Direct Memory Access

PLL	Phase-Locked Loop
PWM	Pulse Width Modulation
QEI	Quadrature Encoder Interface
SD	Secure Digital
SPI	Serial Peripheral Interface
SPS	Samples per Second
TDES	Triple Data Encryption Standard
TK	Touch Key
TMR	Timer Controller
UART	Universal Asynchronous Receiver/Transmitter
UCID	Unique Customer ID
USB	Universal Serial Bus
WDT	Watchdog Timer
WWDT	Window Watchdog Timer

表 9.1-1 缩写表

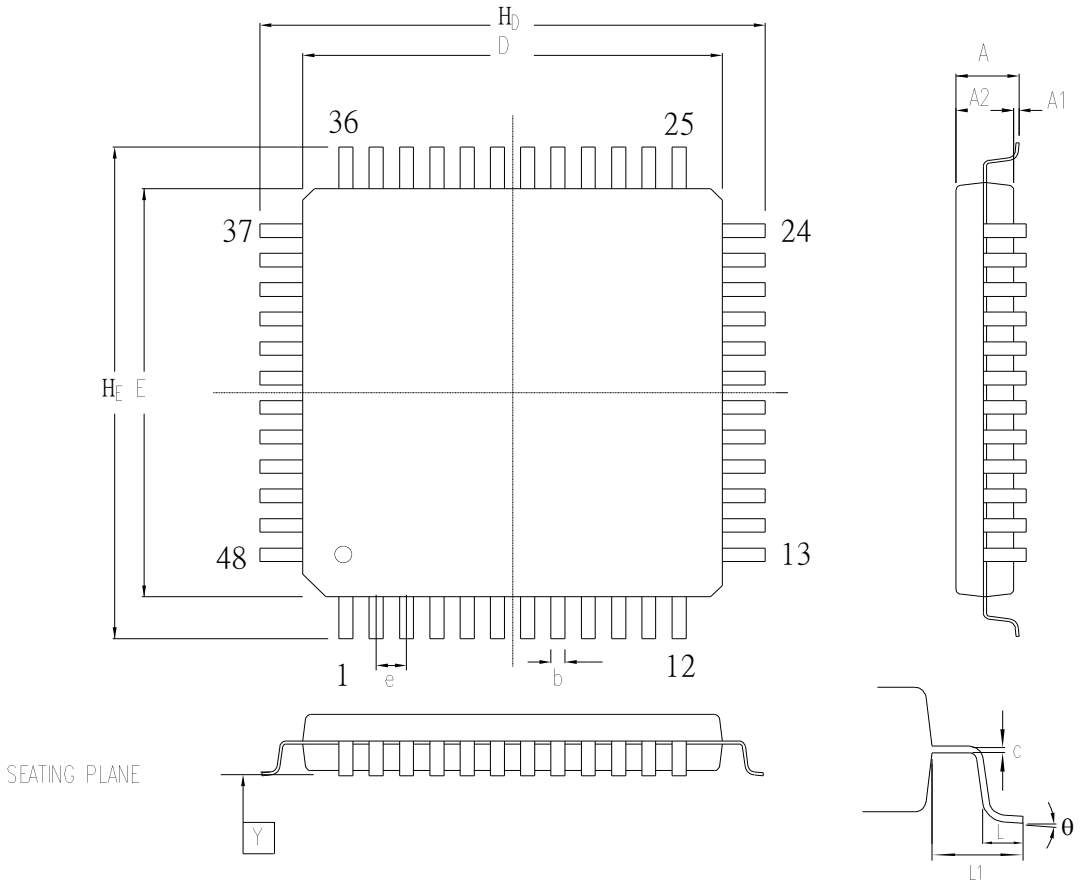
10 封装尺寸

10.1 QFN 33L (5x5x0.8 mm^3 Pitch 0.5 mm)



		SYMBOL	MIN	NOM	MAX
TOTAL THICKNESS		A	0.7	0.75	0.8
STAND OFF		A1	0	0.035	0.05
MOLD THICKNESS		A2	---	0.55	0.57
L/F THICKNESS		A3		0.203 REF	
LEAD WIDTH		b	0.2	0.25	0.3
BODY SIZE	X	D		5 BSC	
	Y	E		5 BSC	
LEAD PITCH		e		0.5 BSC	
EP SIZE	X	J	3.4	3.5	3.6
	Y	K	3.4	3.5	3.6
LEAD LENGTH		L	0.35	0.4	0.45
PACKAGE EDGE TOLERANCE		aaa		0.1	
MOLD FLATNESS		bbb		0.1	
COPLANARITY		ccc		0.08	
LEAD OFFSET		ddd		0.1	
EXPOSED PAD OFFSET		eee		0.1	

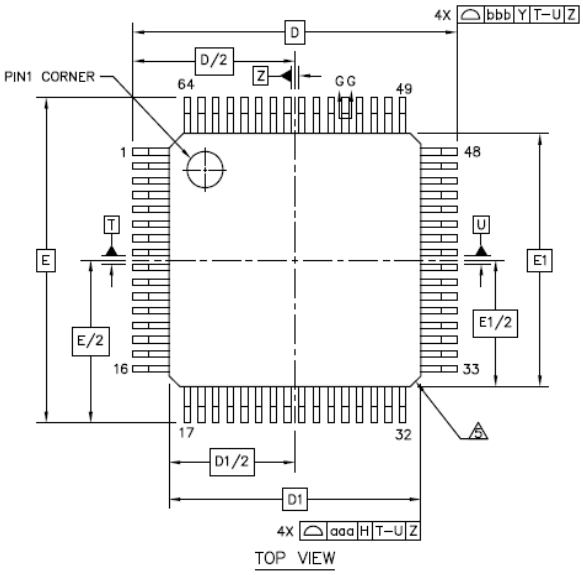
10.2 LQFP 48L (7x7x1.4 mm^3 Footprint 2.0mm)



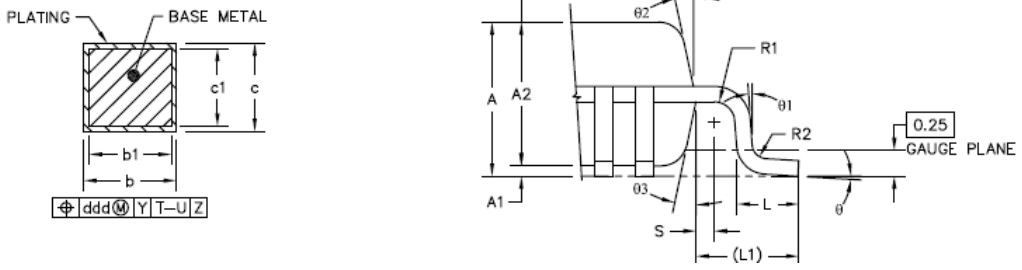
Controlling dimension : Millimeters

Symbol	Dimension in inch			Dimension in mm		
	Min	Nom	Max	Min	Nom	Max
A	—	—	—	—	—	—
A_1	0.002	0.004	0.006	0.05	0.10	0.15
A_2	0.053	0.055	0.057	1.35	1.40	1.45
b	0.006	0.008	0.010	0.15	0.20	0.25
c	0.004	0.006	0.008	0.10	0.15	0.20
D	0.272	0.276	0.280	6.90	7.00	7.10
E	0.272	0.276	0.280	6.90	7.00	7.10
e	0.014	0.020	0.026	0.35	0.50	0.65
H_D	0.350	0.354	0.358	8.90	9.00	9.10
H_E	0.350	0.354	0.358	8.90	9.00	9.10
L	0.018	0.024	0.030	0.45	0.60	0.75
L_1	—	0.039	—	—	1.00	—
Y	—	—	0.004	—	—	0.10
θ	0	—	7	0	—	7

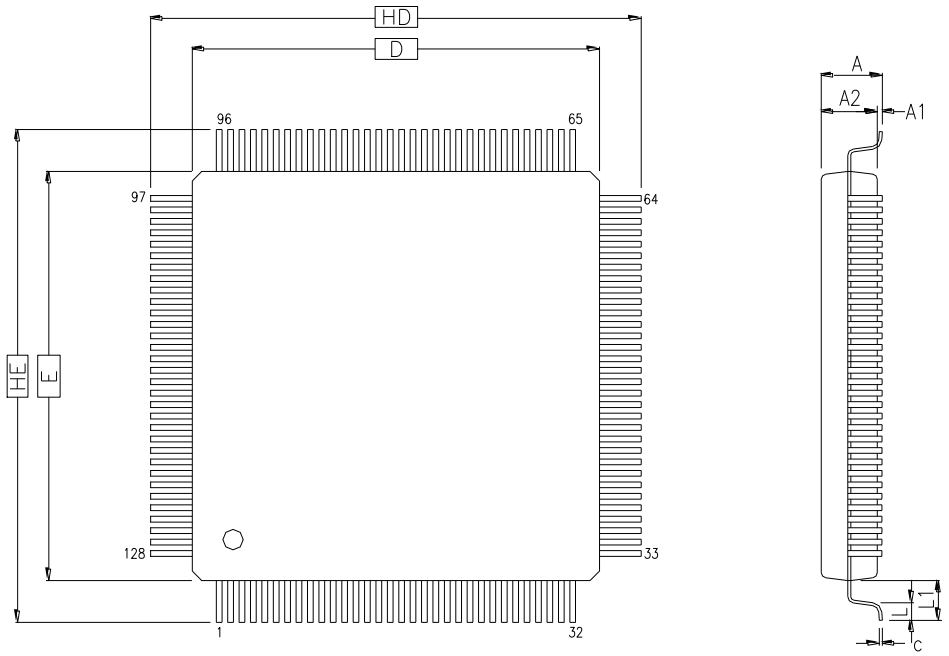
10.3 LQFP 64L (7x7x1.4 mm^3 footprint 2.0 mm)



		SYMBOL	MIN	NOM	MAX
TOTAL THICKNESS		A	---	---	1.6
STAND OFF		A1	0.05	---	0.15
MOLD THICKNESS		A2	1.35	1.4	1.45
LEAD WIDTH(PLATING)		b	0.13	0.18	0.23
LEAD WIDTH		b1	0.13	0.16	0.19
L/F THICKNESS(PLATING)		c	0.09	---	0.2
L/F THICKNESS		c1	0.09	---	0.16
BODY SIZE	X	D	9 BSC		
	Y	E	9 BSC		
	X	D1	7 BSC		
	Y	E1	7 BSC		
LEAD PITCH		e	0.4 BSC		
		L	0.45	0.6	0.75
FOOTPRINT		L1	1 REF		
		θ	0°	3.5°	7°
		θ1	0°	---	---
		θ2	11°	12°	13°
		θ3	11°	12°	13°
		R1	0.08	---	---
		R2	0.08	---	0.2
		S	0.2	---	---
PACKAGE EDGE TOLERANCE		aaa	0.2		
LEAD EDGE TOLERANCE		bbb	0.2		
COPLANARITY		ccc	0.08		
LEAD OFFSET		ddd	0.07		
MOLD FLATNESS		eee	0.05		



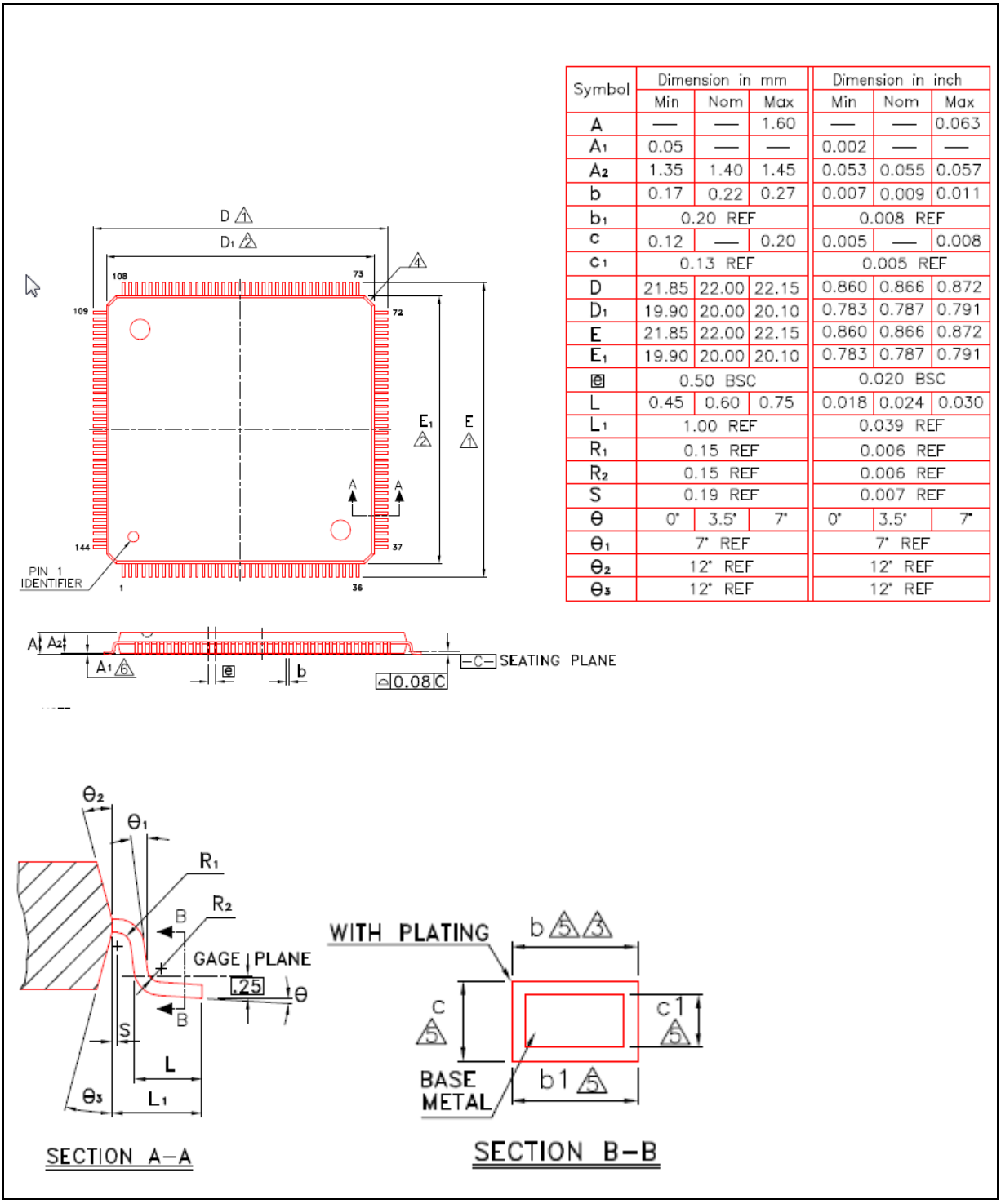
10.4 LQFP 128L (14x14x1.4 mm^3 footprint 2.0 mm)



COTROL DIMENSIONS ARE IN MILLIMETERS.

SYMBOL	MILLIMETER			INCH		
	MIN.	NOM.	MAX.	MIN.	NOM.	MAX.
A	—	—	1.60	—	—	0.063
A1	0.05	—	0.15	0.002	—	0.006
A2	1.35	1.40	1.45	0.053	0.055	0.057
HD	16.00 BSC.			0.630 BSC.		
D	14.00 BSC.			0.551 BSC.		
HE	16.00 BSC.			0.630 BSC.		
E	14.00 BSC.			0.551 BSC.		
b	0.13	0.16	0.23	0.005	0.006	0.009
e	0.40 BSC.			0.016 BSC.		
θ	0°	3.5°	7°	0°	3.5°	7°
c	0.09	—	0.20	0.004	—	0.008
L	0.45	0.60	0.75	0.018	0.024	0.030
L_1	1.00 REF			0.039 REF		
y	–	–	0.1	–	–	0.004

10.5 LQFP 144L (20x20x1.4 mm^3 footprint 2.0 mm)



Symbol	Dimension in mm			Dimension in inch		
	Min	Nom	Max	Min	Nom	Max
A	—	—	1.60	—	—	0.063
A_1	0.05	—	—	0.002	—	—
A_2	1.35	1.40	1.45	0.053	0.055	0.057
b	0.17	0.22	0.27	0.007	0.009	0.011
b_1	0.20 REF			0.008 REF		
c	0.12	—	0.20	0.005	—	0.008
c_1	0.13 REF			0.005 REF		
D	21.85	22.00	22.15	0.860	0.866	0.872
D_1	19.90	20.00	20.10	0.783	0.787	0.791
E	21.85	22.00	22.15	0.860	0.866	0.872
E_1	19.90	20.00	20.10	0.783	0.787	0.791
e	0.50 BSC			0.020 BSC		
L	0.45	0.60	0.75	0.018	0.024	0.030
L_1	1.00 REF			0.039 REF		
R_1	0.15 REF			0.006 REF		
R_2	0.15 REF			0.006 REF		
S	0.19 REF			0.007 REF		
θ	0°	3.5°	7°	0°	3.5°	7°
$θ_1$	7° REF			7° REF		
$θ_2$	12° REF			12° REF		
$θ_3$	12° REF			12° REF		

11 修订历史

日期	版本	描述
2018.03.30	1.00	初始版本。
2018.07.16	1.01	1. 章节6.2.12在寄存器SYS_IPRST1中增加HSOTGRST，OTGRST。 2. 章节6.2.7增加注释“SRAM bank2当读数据时会有额外两个等待周期”。 3. 图6.3-1增加注释USB高速应用中HXT必须为12 MHz。 4. 章节6.3.9增加注释“在CLK_PCLKDIV寄存器中PCLK必须小于 96 MHz”。
2019.12.20	2.00	1. 增加新M480 256 /128 KB 内存产品线。 2. 移除选型表M48xGAAE型号信息。 3. 增加了FMC、BPWM、SDH、Crypto和EADC外设的功能比较表。 4. 增加章节6.44外设互动允许外设之间的自主通信或同步动作。

Important Notice

Nuvoton Products are neither intended nor warranted for usage in systems or equipment, any malfunction or failure of which may cause loss of human life, bodily injury or severe property damage. Such applications are deemed, "Insecure Usage".

Insecure usage includes, but is not limited to: equipment for surgical implementation, atomic energy control instruments, airplane or spaceship instruments, the control or operation of dynamic, brake or safety systems designed for vehicular use, traffic signal instruments, all types of safety devices, and other applications intended to support or sustain life.

All Insecure Usage shall be made at customer's risk, and in the event that third parties lay claims to Nuvoton as a result of customer's Insecure Usage, customer shall indemnify the damages and liabilities thus incurred by Nuvoton.

Please note that all data and specifications are subject to change without notice.
All the trademarks of products and companies mentioned in this datasheet belong to their respective owners.