以 SPIM 读取外部 SPI Flash 图片

NuMicro® 32 位系列微控制器范例代码介绍

文件信息

应用简述	本范例代码使用 NuMaker-M467HJ 来实作以 SPIM 读取存于微控制器外部 SPI flash 中的图文件并显示于 NuMaker-M467HJ 的 LCD 上，并比较两线与四线模式的 SPIM 在读取速度的差异。
BSP 版本	M460_Series_BSP_CMSIS_V3.00.004 emWin Software Package for M460 V7.1
开发平台	NuMaker--M467HJ V1.1

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 and microprocessor 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

1. 概述

本范例使用 NuMaker-M467HJ 来实作以 SPIM(SPI Synchronous Serial Interface)读取存于微控制器外部 SPI flash 中的图文件并显示于 NuMaker-M467HJ 的 LCD 上，同时亦比较两线与四线模式的 SPIM 在读取速度的差异。

在人机接口(HMI)或需要丰富图形显示的项目中，受限于微控制器内建 SRAM 空间不足，设计者时常会将大量的图形资源（如图标、背景图片...等）储存在外部的 SPI Flash 中。当需要显示时，再透过 SPIM 接口高速传输到 MCU，并最终推送到 LCD 驱动器。

1.1 原理

本范例的核心在于利用 SPIM(SPI Synchronous Serial Interface)的多 I/O(Multi-I/O)能力，高效地将储存于外部 SPI Flash 的图像数据读取出来，并显示在 LCD 屏幕上。

1.1.1 SPIM 与 Direct Memory Mapping 功能

Nuvoton NuMicro M467 系列微控制器的 SPIM 模块是专为高速存取外部 SPI Flash 而设计的高效能接口。它不仅完整支持标准 SPI 模式，更提供了 Dual 和 Quad 的多 I/O 模式，同时具备强大的直接内存映像模式(Direct Memory Mapping Mode，简称 DMM mode)功能。

整个流程始于以 MCU 作为 Master，向外部 SPI Flash 发出读取指令与图像的目标地址。采用 SPIM 的 DMM mode，其运作原理是，SPIM Controller 会将外部的 SPI Flash 内存空间映像到 MCU 的内部地址空间中，如表 1-1。对于 CPU 而言，这块外部 Flash 就像是内部存储器的一部分，其存取地址被统一纳入 MCU 的内存分区规划。

Memory Address	Size	Functions
0x40007000-0x40007FFF	4 Kbytes	SPIM control registers：储存 SPIM 配置相关参数
0x100000-0x020FFFFF	32 Mbytes	外部 SPI flash 被映像到的 MCU 内部地址空间(注：此地址空间仅适用于 Flash 大小为 1024 Kbytes 的型号)

表 1-1 SPIM 内存映像地址

一旦进入此模式，当 CPU 尝试从这块映像的内存地址(0x100000)执行读取操作时，SPIM Controller 会自动地执行完整的 SPI Flash 读取时序，包括发送读取指令、地址，并启动高速数据传输，而不需要 CPU 介入或额外的函数调用。这极大地简化了图像数据的获取流程，尤其适用于需要频繁读取图像数据的应用场景。

SPIM 的相关软件设定请参考代码介绍。

1.1.2 Dual Mode and Quad Mode

要提升效能，SPI Flash 读取效率亦是一大关键，以标准 SPI 接口是使用一对数据线(MISO 与 MOSI)进行数据输入和输出，为了增加效率，大多数的 SPI Flash 能支持双线模式(Dual Mode)与四线模式(Quad Mode)的 Multi-I/O 模式。

在 Dual Mode 下，传统的 MOSI 和 MISO 脚位被配置成双向的 IO0 与 IO1 数据线，让 SPIM 在每个时钟周期内并行传输 2 个位(2-bit)，数据吞吐量便提高一倍。更进一步，Quad Mode 模式则将 Flash 芯片上的写保护(WP#)和暂停(HOLD#)脚位重新定义为额外的数据线 IO2 和 IO3。当 Flash 芯片被设定启用 Quad Enable (QE)位后，SPIM 便可以在每个时钟周期内利用这四条数据线并行传输 4 个位(4-bit)，理论带宽提升至 Dual Mode 的两倍。

在设定 SPIM 进入 DMM 模式并启用 Dual/Quad Mode 时，必须注意 MCU 与外部 SPI Flash 的指令兼容性，不同的 Flash 芯片厂商或型号，其高速连续读取指令(Fast Read Opcode)可能不同。

此范例所使用的开发板为 NuMaker-M467HJ V1.1，从电路图可以看到选用的是 Winbond SPI flash 型号为 W25Q32JVSSIQ，可以支持 M467 series SPIM 所设定的读取指令，如指令 0x3B 为 Fast Read Dual Output、指令 0xEB 为 Fast Quad Read，并且如 Dummy clocks 等，需对照 SPI flash 规格所述的波型设定，如图 1-1，以符合时序要求。

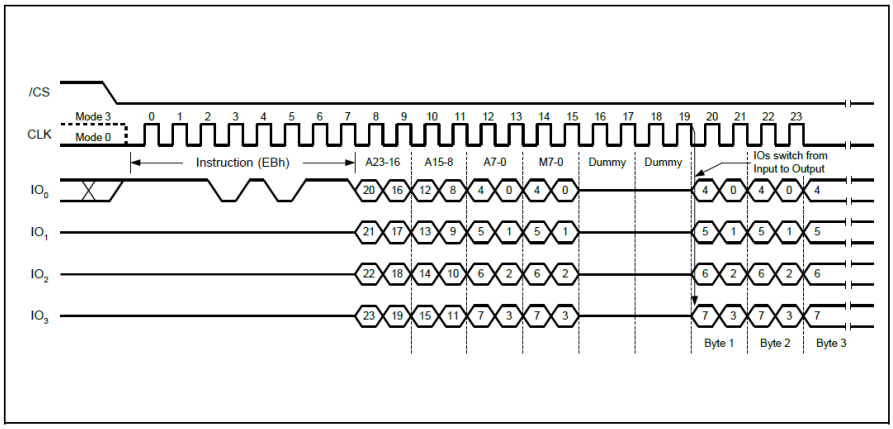

Figure 24. Fast Read Quad I/O Instruction (M7-M0 should be set to Fxh)

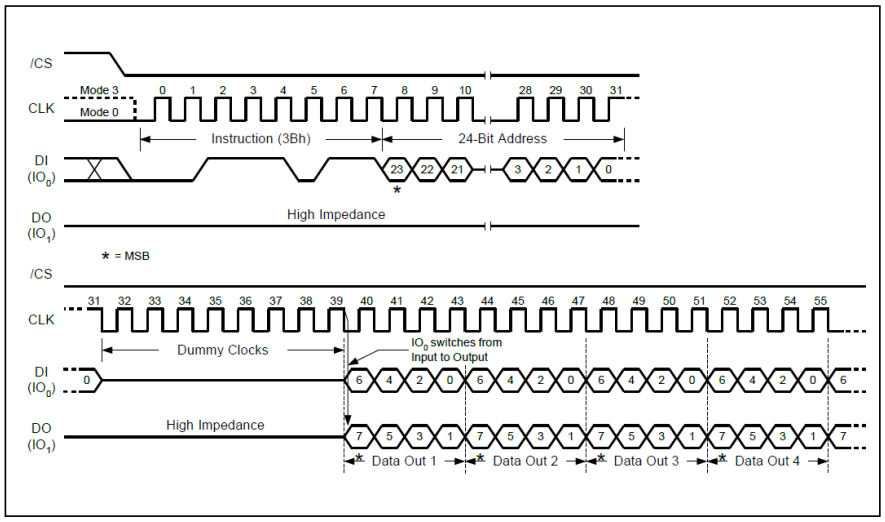

Figure 18. Fast Read Dual Output Instruction

Source: Datasheet of Winbond W25Q32JV

图 1-1 SPI Flash 规格书

而硬件脚位配置则如表 1-2，电路连接方式如图 1-2 所示。

Pin Number	I/O Function
PI12	SPIM_SS
PI13	SPIM_MISO
PI14	SPIM_D2
PI15	SPIM_D3
PJ0	SPIM_CLK
PJ1	SPIM_MOSI

表 1-2 脚位配置表

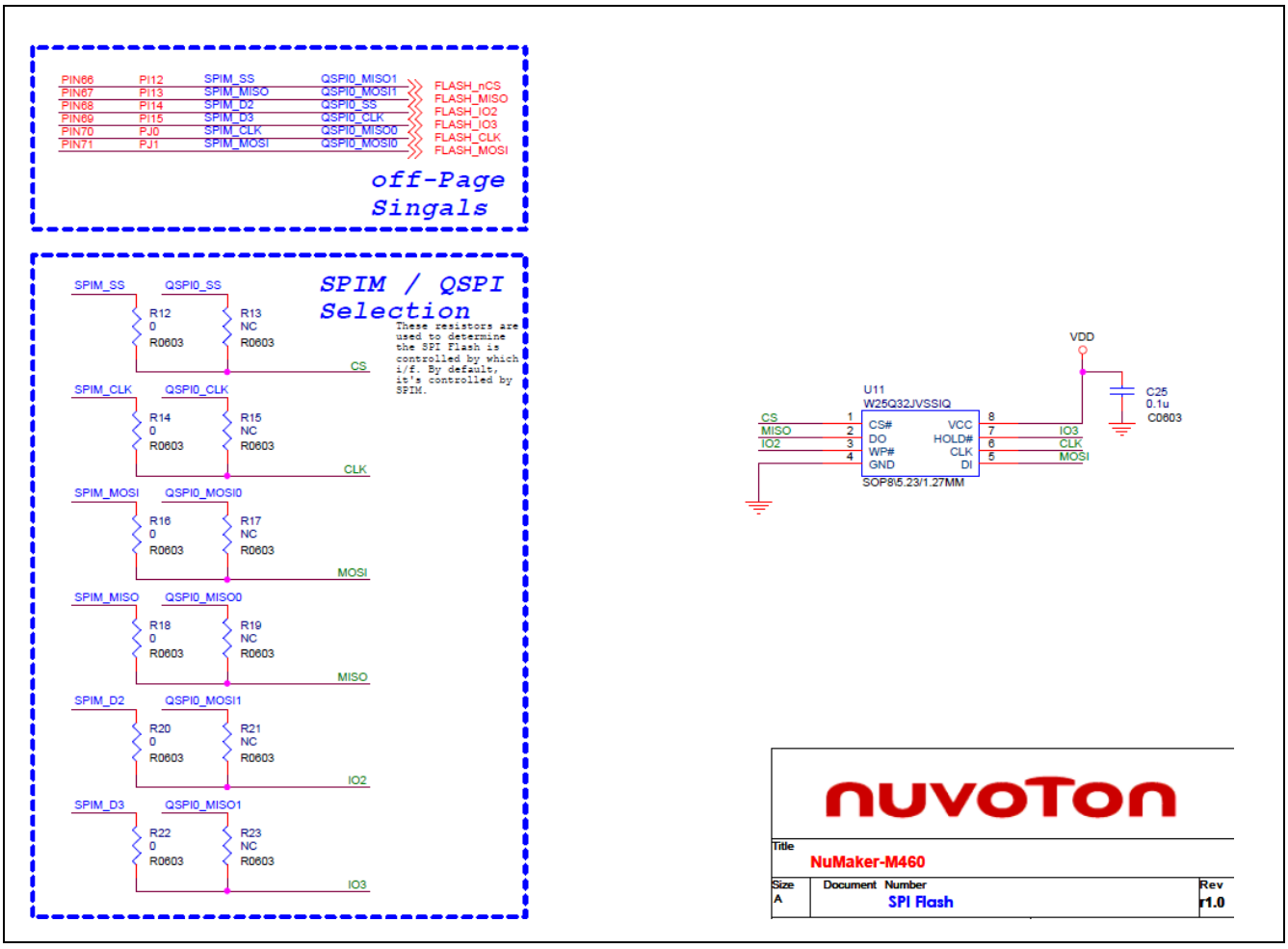

图 1-2 SPI Flash 电路图连接方式

1.2 执行结果

1.2.1 烧录

在烧录 SPI flash 时，需要于烧录设定中指定烧录脚位，以 Keil MDK 为范例，需参考图 1-3 设定，按照表 1-1 脚位规划，选定对应的脚位。

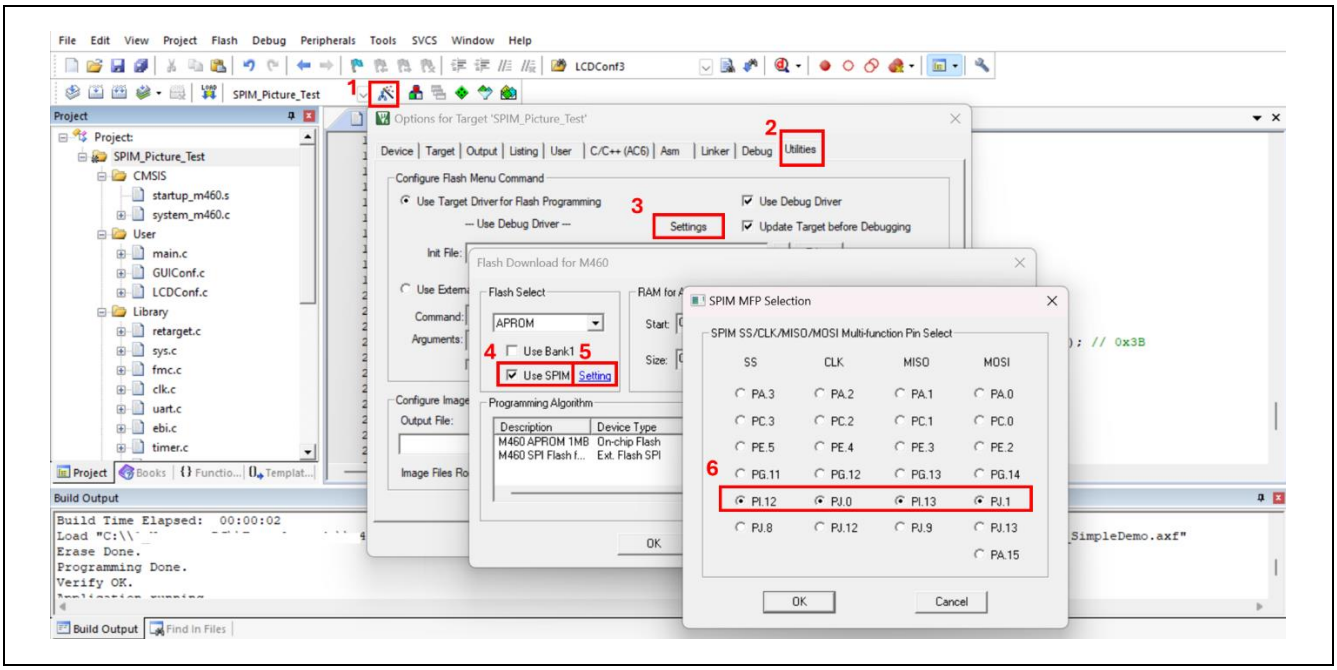

图 1-3 SPI Flash 烧录设定

1.2.2 执行

执行编译和烧录的结果如图 1-4 与图 1-5 所示。初始化完成后，使用 PC UART 终端机输入 0 或 1 以进入 Quad Mode 或 Dual Mode 两种测试模式，执行完成后，会在 LCD 上看到图片成功显示，并透过打印出的信息查看从 SPI Flash 加载一张分辨率为 480x272 的图所需耗费的时间。

透过软件时间的计算，可以看到绘制一张分辨率为 480x272 的图片，Quad Mode 仅需约 18 ms，而 Dual Mode 需约 35 ms，证实 Quad Mode 的数据带宽优势，能大幅缩短图像加载的时间，对于人机接口的流畅度亦会有决定性的影响。

```
+-----------------------------------------------------------------------+
  emWin AppWizard SPIM Read Mode Test
  This example project is based on emWin AppWizard.
  It demonstrates the performance and initialization
  differences between SPIM Fast Quad Read mode and
  Fast Dual Read mode.

  Please select the test mode:
    Enter 0 -> Test Fast 'Quad' Read mode
    Enter 1 -> Test Fast 'Dual' Read mode
+-----------------------------------------------------------------------+

Setting up Fast 'Quad' Read mode...
== Test completed in [18] ms with Mode 0. Picture showed on LCD. ==|
```

图 1-4 执行编译和烧录结果– Quad Mode

```
+-----------------------------------------------------------------------+
  emWin AppWizard SPIM Read Mode Test
  This example project is based on emWin AppWizard.
  It demonstrates the performance and initialization
  differences between SPIM Fast Quad Read mode and
  Fast Dual Read mode.

  Please select the test mode:
    Enter 0 -> Test Fast 'Quad' Read mode
    Enter 1 -> Test Fast 'Dual' Read mode
+-----------------------------------------------------------------------+

Setting up Fast 'Dual' Read mode...
SPIM get JEDEC ID=0xEF, 0x40, 0x16
== Test completed in [35] ms with Mode 1. Picture showed on LCD. ==
```

图 1-5 执行编译和烧录结果– Dual Mode

另外，亦可使用市面上的逻辑分析仪实际量测图片读取的时间，此范例使用仪器 Acute MSO3124V，如图 1-6，选择 Serial Flash Bus Decode 模式分析，可以得到测量结果如图 1-7 所示，除了测量时间，亦可从 Bus Decode 结果验证是否有正确设定到对应的模式。

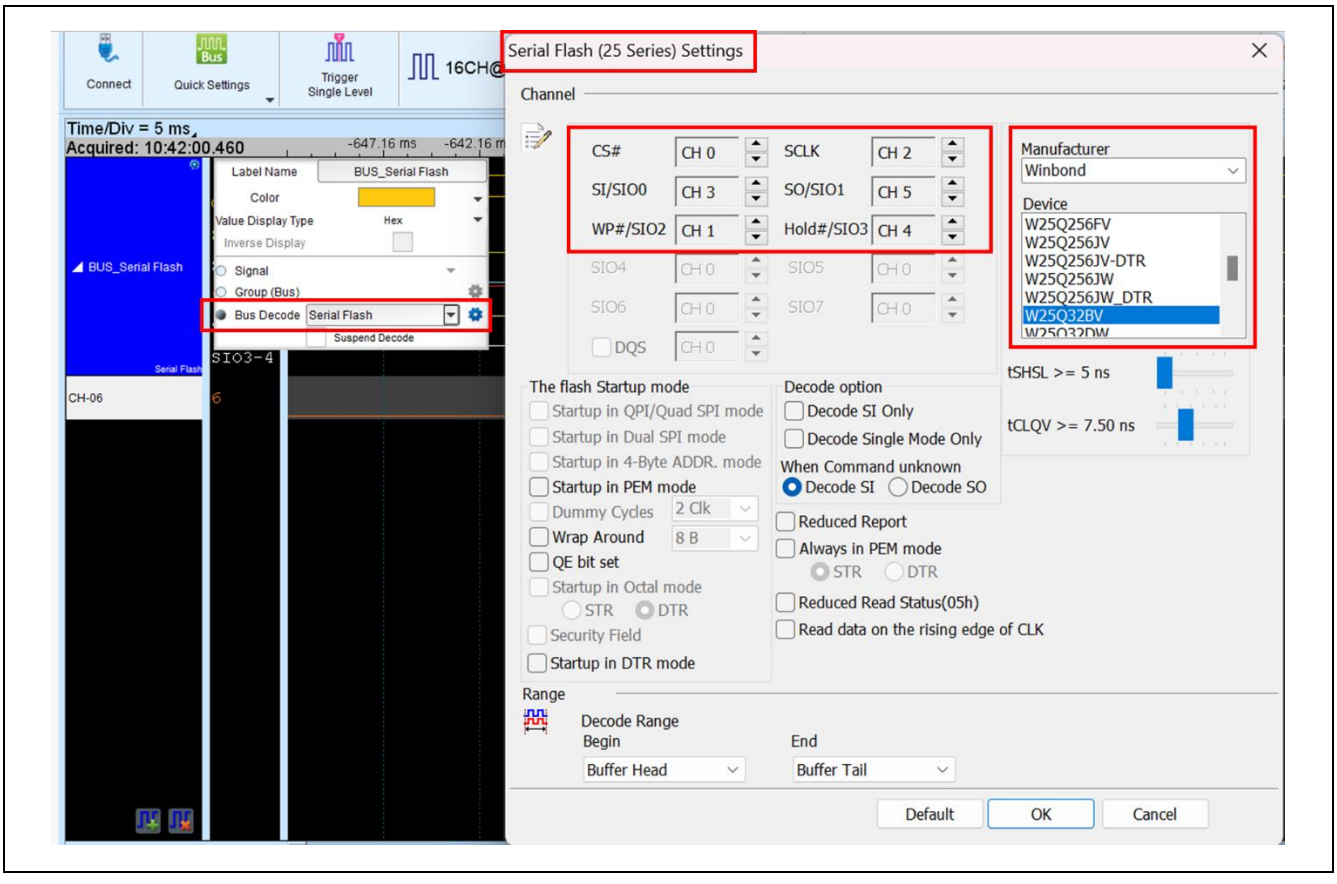

图 1-6 逻辑分析仪 Bus Decode 设定

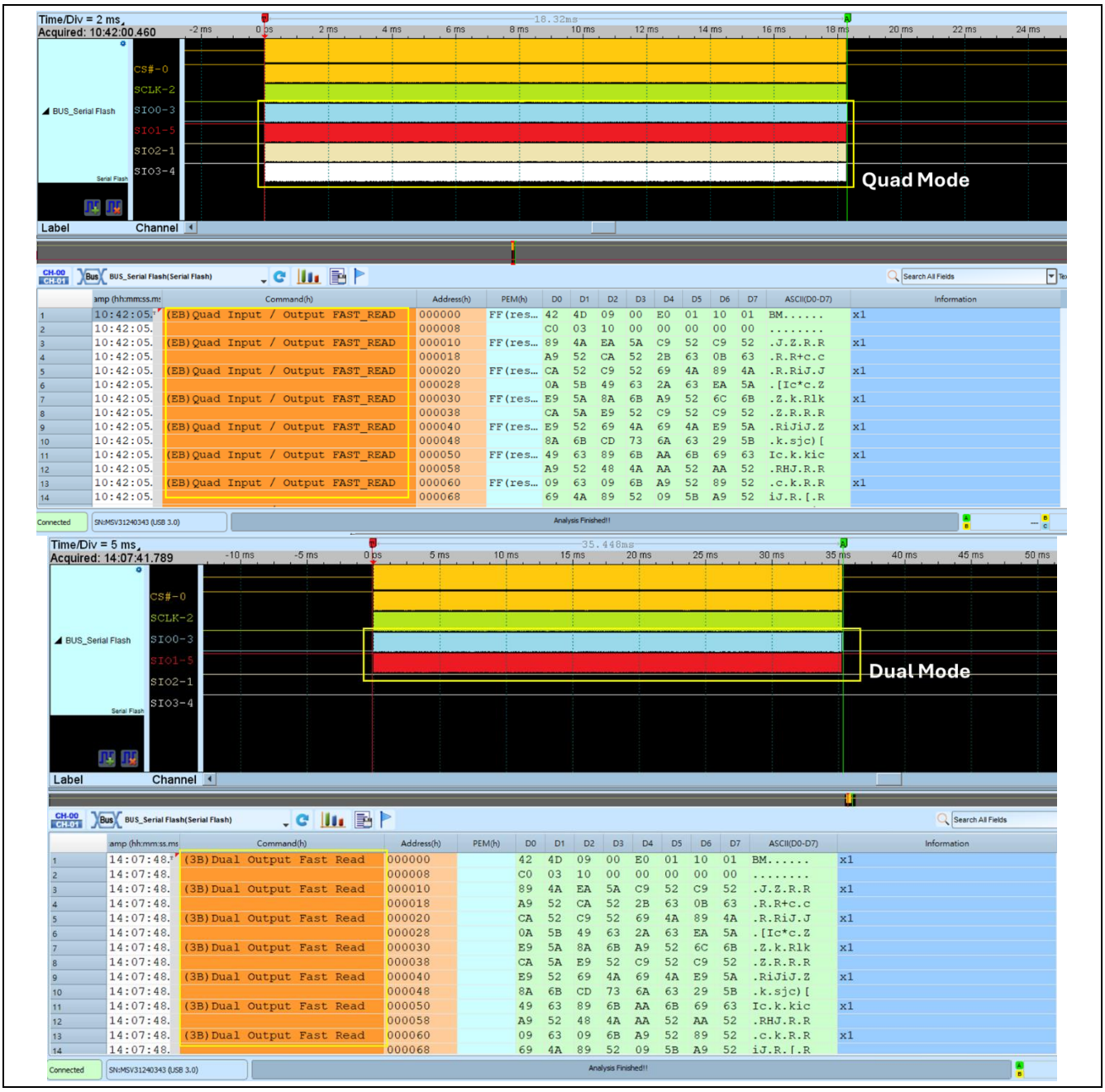
Time/Div = 2 ms
Acquired: 10:42:00.460
CS#-0
SCLK-2
SIO0-3
SIO1-5
SIO2-1
SIO3-4
BUS_Serial Flash
Quad Mode
Label
Channel
Time/Div = 5 ms
Acquired: 14:07:41.789
Dual Mode

图 1-7 逻辑分析结果

2. 代码介绍

SPIM 的主设定代码位于 *main.c*。

SPIM 初始化分为脚位设置与参数设置。脚位设置位于 sys_init()函数中，对应表 1-2 设定 SPIM multi-function pin，其中 clock pin(PJ0)设定为 schmitt trigger 提高抗噪声能力，以确保频率信号稳定性，并对 SPIM I/O 设定为高转换率模式，使电压变化速度更快，在高速通信时可以保持信号完整性。

```
void sys_init(void)
{
    …
    /* Init SPIM multi-function pins, MOSI(PJ.1), MISO(PI.13), CLK(PJ.0), SS(PI.12),
D3(PI.15), and D2(PI.14) */
    SET_SPIM_MOSI_PJ1();
    SET_SPIM_MISO_PI13();
    SET_SPIM_CLK_PJ0();
    SET_SPIM_SS_PI12();
    SET_SPIM_D3_PI15();
    SET_SPIM_D2_PI14();
    PJ->SMTEN |= GPIO_SMTEN_SMTEN0_Msk;

    /* Set SPIM I/O pins as high slew rate up to 80 MHz. */
    PI->SLEWCTL = (PI->SLEWCTL & 0x0CFFFFFF) | (0x1<<GPIO_SLEWCTL_HSREN12_Pos) |
                  (0x1<<GPIO_SLEWCTL_HSREN13_Pos) | (0x1<<GPIO_SLEWCTL_HSREN14_Pos) |
                  (0x1<<GPIO_SLEWCTL_HSREN15_Pos) ;
    PJ->SLEWCTL = (PJ->SLEWCTL & 0xF3FFFFF0) | (0x1<<GPIO_SLEWCTL_HSREN0_Pos) |
                  (0x1<<GPIO_SLEWCTL_HSREN13_Pos) ;
    …
}
```

SPIM 的参数设置，则于 main()函数中，设定说明如下：

1. SPIM 频率：SPIM_SET_CLOCK_DIVIDER 为分频器设定，计算公式为 HCLK/(Divider)^2，此范例 HCLK 频率为 200 MHz，因此值为 2 表示 SPIM 频率为 50 MHz。

2. 接收频率延迟：SPIM_SET_RXCLKDLY_RDDLYSEL 与 SPIM_SET_RXCLKDLY_RDEDGE 为接收频率延迟设定，此处设定为无延迟。

3. Dummy Cycles：SPIM_SET_DCNUM 为在 MCU 发送完指令后到 SPI Flash 开始输出有效数据前的缓冲时间，根据图 1-1 规格书波型，若指令为 0xEB(Fast Read Quad Output)，需有 4 个 clocks 的 dummy cycles，而指令为 0x3B(Fast Read Dual Output)，需有 8 个 clocks 的 dummy cycles。

4. 初始化 SPI Flash 并验证 ID：SPIM_InitFlash 是对 SPI Flash 初始化的 API，也透过 SPIM_ReadJedecId 印出 SPI Flash 的 JEDEC ID。

5. SPIM 内部快取：DMM mode 下可以启用 SPIM 内部 16 Kbytes 快取，此快取区块会成为在 MCU 与 SPI Flash 间的缓冲区，若连续存取相近的数据，将会直接从内部快取提供，能再减少到 SPI Flash 存取的次数。

6. 加密位：Chiper Disable Bit 是决定是否要对传输的数据进行实时加解密， 由于加解密亦会需要额外处理时间，因此此范例设定宏定义 SPIM_CIPHER_ON 为 0，关闭加解密功能。

7. 4 bytes mode ： W25Q32JVSSIQ 并无支持 4 bytes mode ， 因此宏定义 USE_4_BYTES_MODE 为 0，使用 3 bytes mode。

8. SPIM DMM mode：SPIM_EnterDirectMapMode 是 SPIM 进入 DMM mode 的 API，其中的 u32RdCmd 参数即是用于指定采用的 I/O 模式，如指令 0xEB 为 Fast Read Quad Output，指令 0x3B 为 Fast Read Dual Output。

```
int main(void)
{
   …
   if(mode == '0')
   {
      printf("Setting up Fast 'Quad' Read mode...\n");
      uint8_t idBuf[3];
      SYS_UnlockReg();
      SPIM_SET_CLOCK_DIVIDER(2);
      SPIM_SET_RXCLKDLY_RDDLYSEL(0);
      SPIM_SET_RXCLKDLY_RDEDGE();
      SPIM_SET_DCNUM(4);

      if (SPIM_InitFlash(1) != 0)
      {
        printf("SPIM flash initialize failed!\n");
      }
      SPIM_ReadJedecId(idBuf, sizeof (idBuf), 1);
      printf("SPIM get JEDEC ID=0x%02X, 0x%02X, 0x%02X\n", idBuf[0], idBuf[1], idBuf[2]);

      SPIM_ENABLE_CACHE();

      if (SPIM_CIPHER_ON)
        SPIM_ENABLE_CIPHER();
      else
        SPIM_DISABLE_CIPHER();

      if (SPIM_Enable_4Bytes_Mode(USE_4_BYTES_MODE, 1) != 0)
      {
        printf("SPIM_Enable_4Bytes_Mode failed!\n");
      }
```

```
        /* 0xEB */
        SPIM_EnterDirectMapMode(USE_4_BYTES_MODE, CMD_DMA_FAST_QUAD_READ, 0);
    }
    else if(mode == '1')
    {
        printf("Setting up Fast 'Dual' Read mode...\n");
        uint8_t idBuf[3];
        SYS_UnlockReg();
        SPIM_SET_CLOCK_DIVIDER(2);
        SPIM_SET_RXCLKDLY_RDDLYSEL(0);
        SPIM_SET_RXCLKDLY_RDEDGE();

        SPIM_SET_DCNUM(8);

        if (SPIM_InitFlash(1) != 0)
        {
          printf("SPIM flash initialize failed!\n");
        }
        SPIM_ReadJedecId(idBuf, sizeof (idBuf), 1);
        printf("SPIM get JEDEC ID=0x%02X, 0x%02X, 0x%02X\n", idBuf[0], idBuf[1], idBuf[2]);

        SPIM_ENABLE_CACHE();

        if (SPIM_CIPHER_ON)
          SPIM_ENABLE_CIPHER();
        else
          SPIM_DISABLE_CIPHER();

        if (SPIM_Enable_4Bytes_Mode(USE_4_BYTES_MODE, 1) != 0)
        {
           printf("SPIM_Enable_4Bytes_Mode failed!\n");
        }
        /* Disable quad mode bit */
        SPIM_SetQuadEnable(0, 2);
        /* 0x3B */
        SPIM_EnterDirectMapMode(USE_4_BYTES_MODE, CMD_DMA_FAST_READ_DUAL_OUTPUT, 0);
    }
    …
}
```

最后，藉由 emWin Library 的 API GUI_DrawStreamedBitmapAuto()绘制一张分辨率 480*272 的图，在绘图前后纪录时间戳，并由 UART 打印出计算结果。

```
int main(void)
{
    …
    GUI_MULTIBUF_Begin();
    tStart = OS_TimeMS;
    GUI_DrawStreamedBitmapAuto(acEbike_Demo_BG0, 0, 0);
    tEnd = OS_TimeMS;
    GUI_MULTIBUF_End();
    tElapsed = tEnd - tStart;
    printf("== Test completed in [%d] ms with Mode %c. Picture showed on LCD. ==\r\n",
    tElapsed, mode);
    GUI_Delay(2000);
```

```
    …
}
```

如同前面所述，SPI Flash 内存空间虚拟地映像到 MCU 的 0x100000 地址空间中，在 Keil MDK 可以透过 Scatter File 精确地规划内存区块，开启 Scatter File 的步骤如图 2-1，设定范例如下，在此范例中定义了起始位置为 0x100000 的 LOAD_ROM_2 区域，也就是 SPI Flash 虚拟映像的地址，而图片 Ebike_Demo_BG0.o 被明确分配到此区块中，使得 MCU 可以直接到 0x100000 存取此图片，实现高效率的外部数据存取功能。

```
LOAD_ROM_1  0x0
{
    APROM.bin  0x0
    {
     startup_M460.o (RESET, +FIRST)
     *(+RO)
    }

    SRAM  0x20000000 0x80000
    {
     * (+RW, +ZI)
    }
}

LOAD_ROM_2  0x100000
{
    SPIM.bin  0x100000
    {
     Ebike_Demo_BG0.o (+RO)
    }
}
```

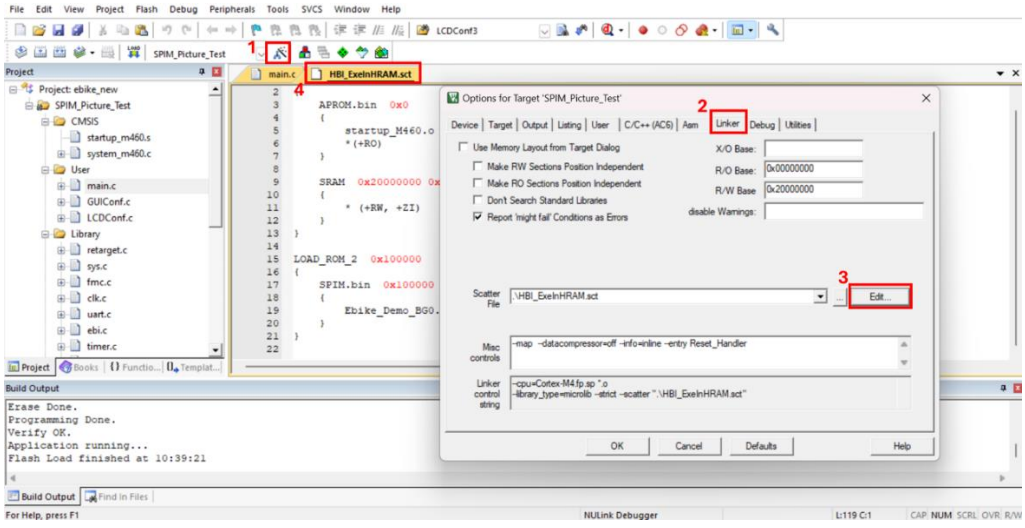

图 2-1 Scatter File 开启方式

3. 软件与硬件需求

3.1 软件需求

- BSP 版本
  - M460_Series_BSP_CMSIS_V3.00.004
  - emWin Software Package for M460 V7.1
- IDE 版本
  - Keil uVision 5.43

3.2 硬件需求

- 电路组件
  - NuMaker-M467HJ V1.1

4. 目录信息

目录结构如下图所示。

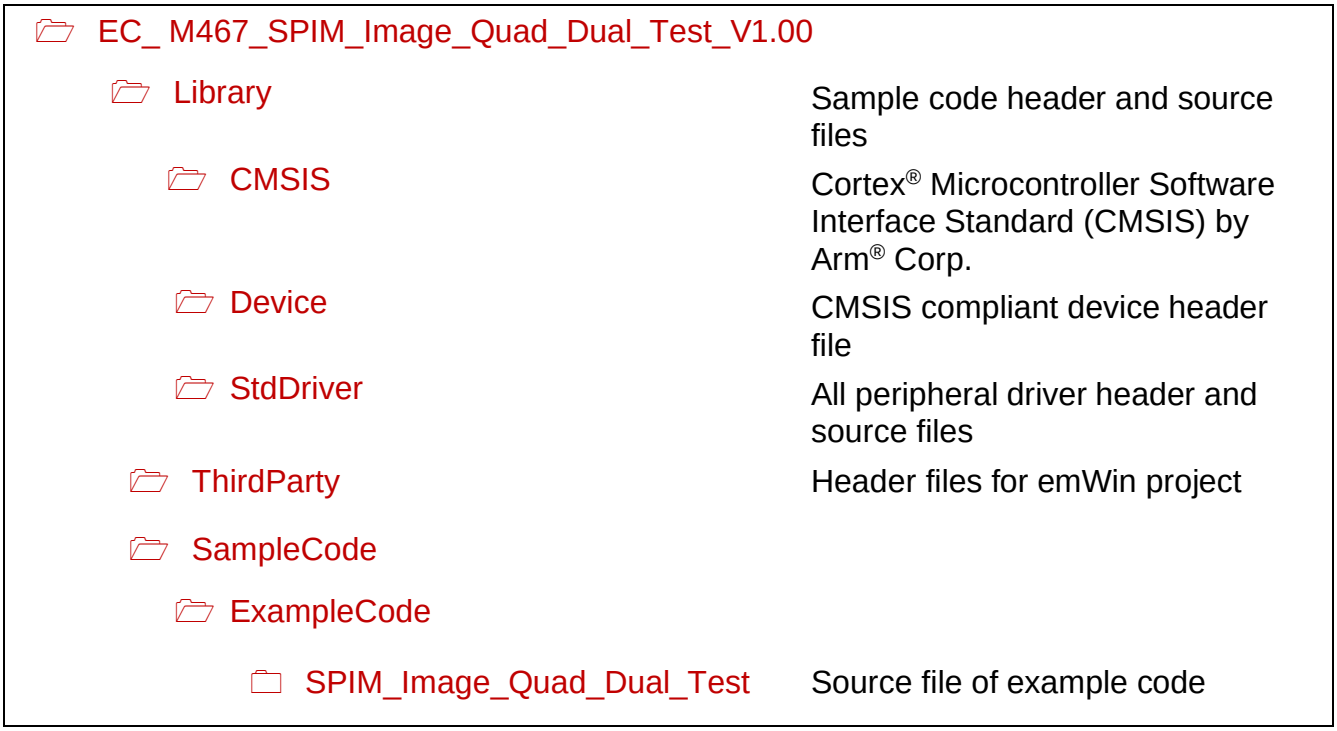

图 4-1 目录信息

5. 范例程序执行

1. 根据目录信息章节进入 ExampleCode 路径中的 KEIL 文件夹，双击 *SPIM_Image_Quad_Dual_Test.uvprojx*。
2. 进入编译模式界面
   - 编译
   - 下载代码至内存
   - 进入 / 离开仿真模式
3. 进入仿真模式界面
   - 执行代码

6. 修订纪录

Date	Revision	Description
2025.11.01	1.00	初始发布。

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.