M467 ADC 10.5 Msps 取樣速率

NuMicro® 32 位元系列微控制器範例代碼介紹

文件資訊

應用簡述	本範例代碼使用 M467 的 EADC0/EADC1/EADC2 交替採樣來實現 10.5 Msps 的取樣速率。
BSP 版本	M460_Series_BSP_CMSIS_V3.00.004
開發平臺	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

本範例代碼將示範如何使用 M467 的 EADC 來實現 10.5 Msps 的取樣速率。

M467 有三個取樣速率高達 4.5 Msps 的 EADC。EADC0/EADC1/EADC2 交替採樣，取樣速率達到三倍，如圖 1-1 所示。

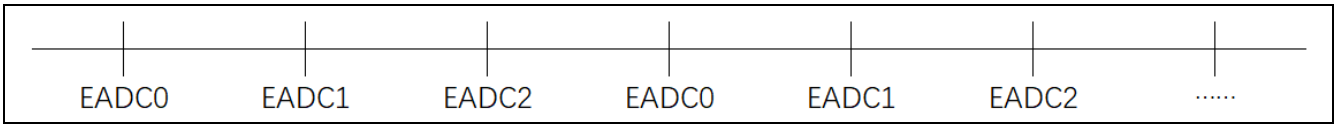


圖 1-1 EADC0/EADC1/EADC2 交替採樣

要實現高速採樣且不丟資料，需要多個外設協同工作。本範例通過 EPWM 觸發 EADC 採樣，EADC 再觸發 PDMA 搬運採樣資料，如圖 1-2 所示。因為觸發延時、PDMA 搬運都需要時間，最終取樣速率可以達到 10.5 Msps。

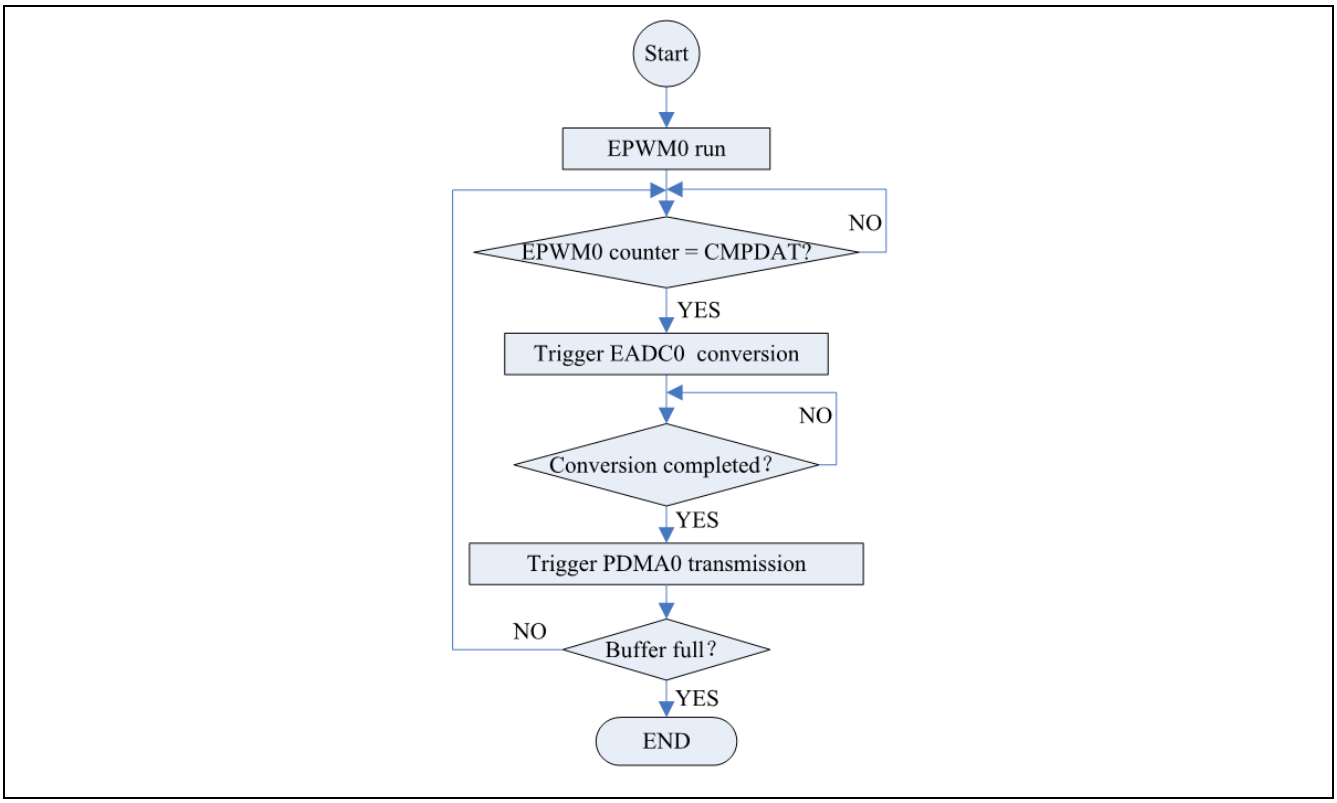


圖 1-2 EPWM/EADC/PDMA 工作流程

1.1 原理

本範例使用 3.5 MHz 的 EPWM，每週期產生三個觸發點分別用於觸發三個 EADC 採樣。每個 EPWM 通道每週期只能產生一個觸發點。採用三路 EPWM 通道同步啟動，設置不同的比較值 (0 點、1/3 週期點、2/3 週期點)，來產生三個不同位置的採樣點，如圖 1-3 所示。再利用 EPWM 的累加器停止模式控制採樣個數。

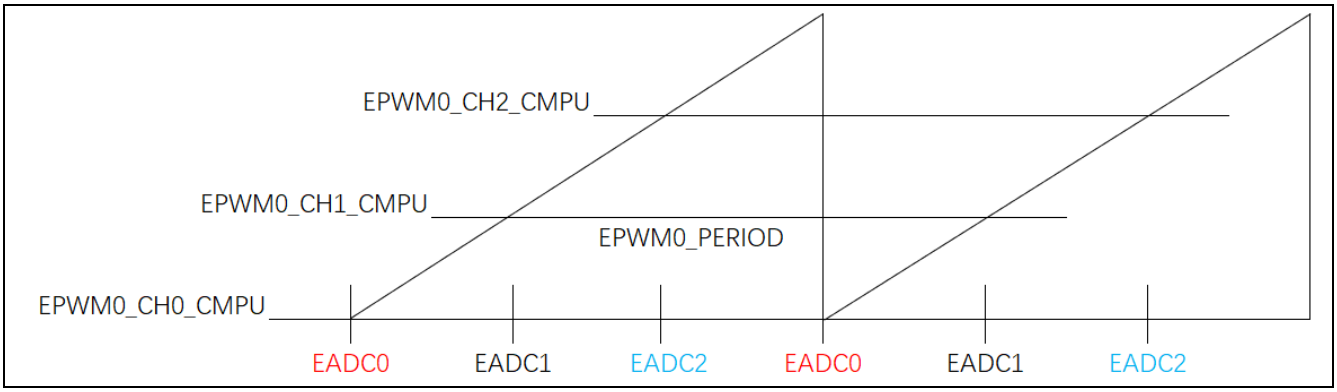


圖 1-3 EADC0/EADC1/EADC2 採樣時序

EADC 採樣完成自動觸發 PDMA 搬運採樣資料。使用 PDMA 的三路通道分別搬運三個 EADC 的資料。另外利用 PDMA 跨步功能，實現直接將三個 EADC 的採樣值依次放入同一個陣列裡，如圖 1-4 所示。

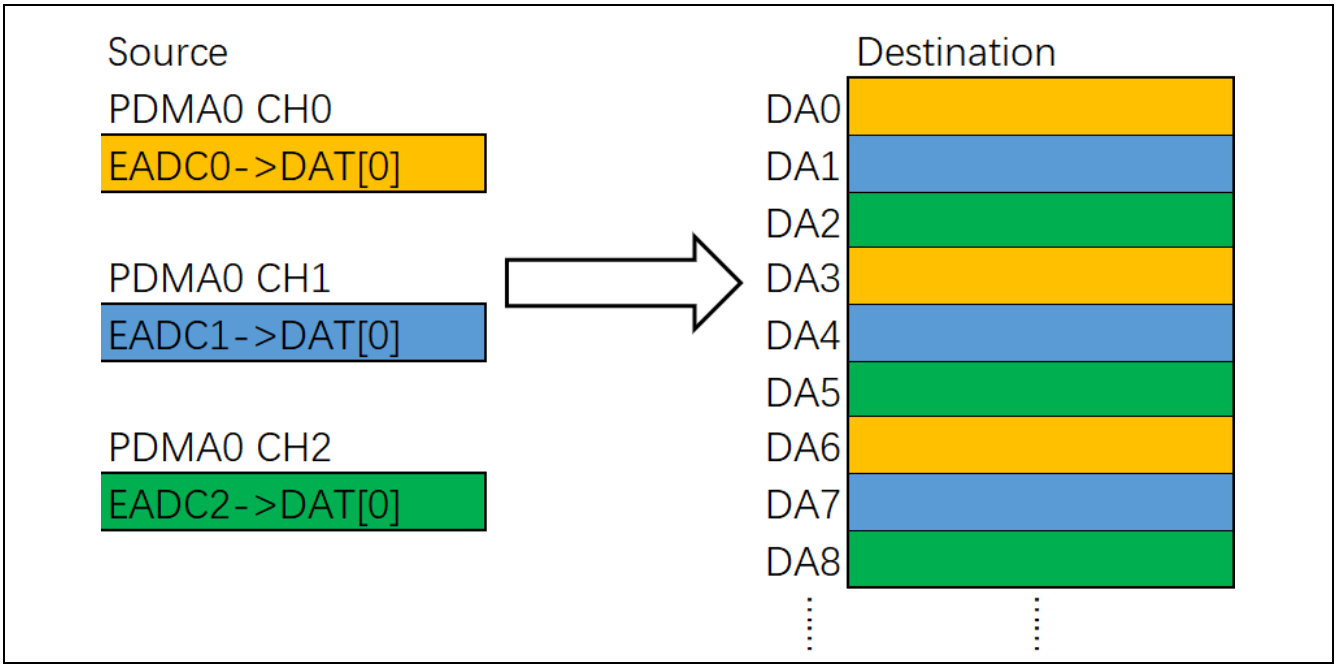


圖 1-4 PDMA 跨步傳輸

1.2 執行結果

如圖 1-5 所示是波形發生器產生的 1.32 MHz 的正弦波，連接到 PB0 引腳作為 ADC 採樣的輸入信號，PB0 可以同時配置為 EADC0_CH0/EADC1_CH8/ EADC2_CH8。

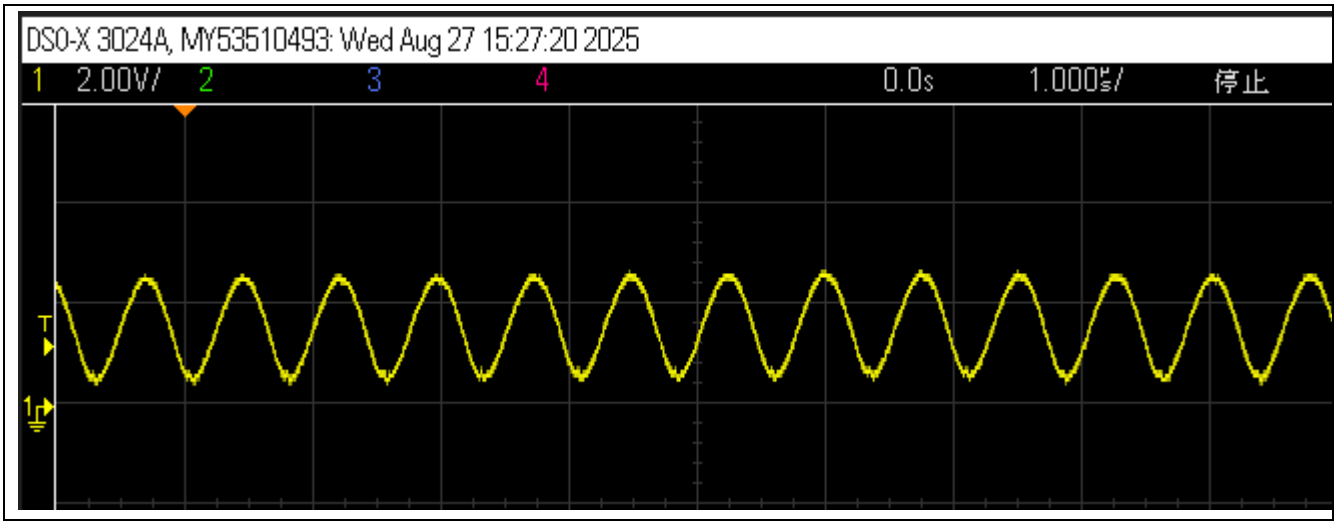


圖 1-5 1.32 MHz 正弦波

PC 上打開串口工具，連接 UART0 用於列印調試資訊。圖 1-6 所示是 M467 列印的 300 個採樣資料(EADC0/EADC1/EADC2 各 100 個) 。

```
+------------------------------------------------------------+
|              M467 EADC maximum sampling rate               |
+------------------------------------------------------------+
EPWM trigger mode.
EADC0/EADC1/EADC2 alternate sampling.
Transfer conversion data by PDMA.
The I/O connection for EADC0~2:
Connect the input signal to PB0.
+------------------------------------------------------------+

Press any key to continue.
Conversion result:
720  1066  1896  2974  3542  3330  2496  1336  728  1104  1960  3006  3572  3418  2440  1270  732  1142  2014  3054  3578  3386  2536  1326  730  1102  2056  3058  3594  3354  2472  1286  730  1142  2012  3046  3574
3270  2382  1222  728  1178  2074  3098  3592  3362  2484  1280  732  1226  2136  3150  3608  3310  2398  1222  730  1194  2084  3146  3560  3258  2312  1152  732  1238  2150  3158  3610  3310  2394  1110  744  1294
2216  3210  3626  3278  2346  1174  734  1248  2234  3182  3640  3214  2250  1110  750  1302  2232  3210  3616  3130  2184  1056  748  1350  2270  3254  3626  3214  2250  1102  742  1394  2312  3290  3626  3166
2184  1046  762  1350  2292  3252  3556  3078  2110  998  766  1402  2344  3298  3624  3164  2172  1052  780  1450  2410  3310  3626  3118  2110  998  772  1400  2426  3310  3610  3062  2042  954  776  1454  2392
3322  3624  3120  1960  922  790  1510  2460  3348  3620  3074  2048  958  814  1542  2528  3384  3604  3018  1992  922  800  1504  2536  3362  3608  2970  1920  886  804  1550  2506  3374  3611  3022  1840  850
832  1594  2568  3406  3594  2974  1928  886  852  1642  2624  3456  3596  2918  1862  862  834  1610  2650  3406  3576  2858  1786  818  862  1662  2622  3438  3582  2920  1720  802  888  1710  2682  3478  3578
2862  1794  830  910  1750  2730  3486  3562  2824  1744  802  886  1702  2676  3464  3480  2714  1674  774  892  1744  2700  3490  3560  2810  1598  782  936  1818  2776  3512  3552  2768  1684  790  914  1856
2790  3542  3518  2704  1624  772  944  1818  2776  3502  3452  2630  1540  758  976  1882  2844  3540  3530  2702  1610  750  1010  1942  2904  3562  3492  2616  1540  740  972  1878  2840  3544  3416  2518  1462
742  1022  1968  2916  3562
Press any key to continue.
```

圖 1-6 調試串口列印的資料

如圖 1-7 所示是 10.5 Msps 的 ADC 採樣資料組成的波形。

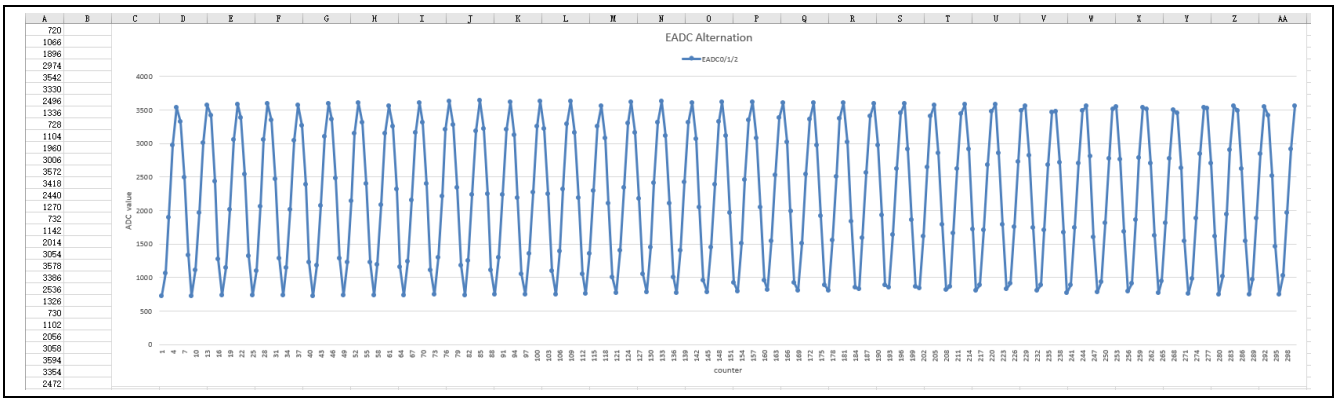


圖 1-7 三組資料組成的波形

2. 代碼介紹

代碼位於 main.c。

巨集定義

定義 EADC 的觸發源、EADC 的通道和採樣模組、PDMA 通道、ADC 採樣個數、PWM 週期和上計數比較點等。

```
/*---------------------------------------------------------------------------------------------------------*/
/* Define                                                                                                  */
/*---------------------------------------------------------------------------------------------------------*/
/* ADC trigger source */
#define EADC0_TRIGGER_CH 0 /* EPWM Trigger channel for EADC0 */
#define EADC1_TRIGGER_CH 1 /* EPWM Trigger channel for EADC1 */
#define EADC2_TRIGGER_CH 2 /* EPWM Trigger channel for EADC2 */

/* ADC channel & module */
#define EADC0_CH         0   /* EADC0 input channel */
#define EADC0_MODULE_NUM 0   /* EADC0 sample module */

#define EADC1_CH         8   /* EADC1 input channel */
#define EADC1_MODULE_NUM 0   /* EADC1 sample module */

#define EADC2_CH         8   /* EADC2 input channel */
#define EADC2_MODULE_NUM 0   /* EADC2 sample module */

/* PDMA channel */
#define EADC0_PDMA_CH    0   /* PDMA channel of EADC0 */
#define EADC1_PDMA_CH    1   /* PDMA channel of EADC1 */
#define EADC2_PDMA_CH    2   /* PDMA channel of EADC2 */

/* ADC sampling times */
#define ADC_SAMPLING_TIMES 100

/* PWM frequency */
#define PWM_FREQ  3500000

/* PWM CMR */
#define PWM_START_POINT    0
#define PWM_1_3_POINT      1
#define PWM_2_3_POINT      2
```

Buffer 定義

定義採樣資料存放的 buffer。

```
static int16_t  DataBuffer[ADC_SAMPLING_TIMES * 3] = {0};
```

系統初始化

使能 PLLFN，設置為 180 MHz，作為 EADC 90 MHz 的時鐘源。

```
void SYS_Init(void)
{
    ......
    /* Set core clock to 200MHz */
    CLK_SetCoreClock(200000000);

    /* Set PLLFN clock to 180MHz */
    CLK_EnablePLLFN(CLK_PLLFNCTL1_PLLSRC_HIRC, 180000000);

    ......

    /* Enable EADC0 module clock */
    CLK_EnableModuleClock(EADC0_MODULE);

    /* Set EADC0 clock divider as 1 */
    CLK_SetModuleClock(EADC0_MODULE, CLK_CLKSEL0_EADC0SEL_PLLFN_DIV2,
                       CLK_CLKDIV0_EADC0(1));

    /* Enable EADC1 module clock */
    CLK_EnableModuleClock(EADC1_MODULE);

    /* Set EADC1 clock divider as 1 */
    CLK_SetModuleClock(EADC1_MODULE, CLK_CLKSEL0_EADC1SEL_PLLFN_DIV2,
                       CLK_CLKDIV2_EADC1(1));

    /* Enable EADC2 module clock */
    CLK_EnableModuleClock(EADC2_MODULE);

    /* Set EADC2 clock divider as 1 */
    CLK_SetModuleClock(EADC2_MODULE, CLK_CLKSEL0_EADC2SEL_PLLFN_DIV2,
                       CLK_CLKDIV5_EADC2(1));

    /* Enable EPWM0 module clock */
    CLK_EnableModuleClock(EPWM0_MODULE);

    /* Select EPWM0 module clock source as HCLK */
    CLK_SetModuleClock(EPWM0_MODULE, CLK_CLKSEL2_EPWM0SEL_HCLK, 0);

    /* Enable PDMA0 clock source */
    CLK_EnableModuleClock(PDMA0_MODULE);

    ......
    /* Set multi-function pins for EADC0~2 channels. */
    SET_EADC0_CH0_PB0();

    /* Disable digital input path of EADC analog pin to prevent leakage */
    GPIO_DISABLE_DIGITAL_PATH(PB, BIT0);
}
```

EPWM 初始化

設置 EPWM0 CH0/CH1/CH2 通道頻率為 3.5 MHz、上計數模式、上計數比較點 (0 點、1/3 週期點、2/3 週期點) 觸發 EADC。使能累加器停止模式，累加器計數值等於設定值後自動停止 EPWM0，用於實現控制 ADC 的採樣次數。設置三個 PWM 通道同步啟動，保證三通道計數同步。

```
void EPWM0_Init(void)
{
    /* Reset EPWM0 module */
    SYS_ResetModule(EPWM0_RST);

    /* EPWM0 channel 0~2 frequency and duty configuration */
    EPWM_ConfigOutputCh(EPWM0, EADC0_TRIGGER_CH, PWM_FREQ, PWM_START_POINT);

    EPWM_ConfigOutputCh(EPWM0, EADC1_TRIGGER_CH, PWM_FREQ, PWM_1_3_POINT);

    EPWM_ConfigOutputCh(EPWM0, EADC2_TRIGGER_CH, PWM_FREQ, PWM_2_3_POINT);

    /* Enable EPWM accumulator */
    EPWM_EnableAcc(EPWM0,
                   EADC0_TRIGGER_CH,   /* Channel = ch0 */
                   ADC_SAMPLING_TIMES, /* Count = ADC_SAMPLING_TIMES */
                   EPWM_IFA_COMPARE_UP_COUNT_POINT); /* Accumulator source = up-count
                                                        compared point */

    EPWM_EnableAcc(EPWM0,
                   EADC1_TRIGGER_CH,   /* Channel = ch1 */
                   ADC_SAMPLING_TIMES, /* Count = ADC_SAMPLING_TIMES */
                   EPWM_IFA_COMPARE_UP_COUNT_POINT); /* Accumulator source = up-count
                                                        compared point */

    EPWM_EnableAcc(EPWM0,
                   EADC2_TRIGGER_CH,   /* Channel = ch2 */
                   ADC_SAMPLING_TIMES, /* Count = ADC_SAMPLING_TIMES */
                   EPWM_IFA_COMPARE_UP_COUNT_POINT); /* Accumulator source = up-count
                                                        compared point */

    /* Enable EPWM0 accumulator stop mode */
    EPWM_EnableAccStopMode(EPWM0, EADC0_TRIGGER_CH);

    EPWM_EnableAccStopMode(EPWM0, EADC1_TRIGGER_CH);

    EPWM_EnableAccStopMode(EPWM0, EADC2_TRIGGER_CH);

    /* EPWM up-count compared point trigger EADC enable */
    EPWM_EnableADCTrigger(EPWM0, EADC0_TRIGGER_CH, EPWM_TRG_ADC_EVEN_COMPARE_UP);

    EPWM_EnableADCTrigger(EPWM0, EADC1_TRIGGER_CH, EPWM_TRG_ADC_ODD_COMPARE_UP);

    EPWM_EnableADCTrigger(EPWM0, EADC2_TRIGGER_CH, EPWM_TRG_ADC_EVEN_COMPARE_UP);

    /* Enable counter synchronous start function for EPWM0 channel 0~2 */
```

```
    EPWM_ENABLE_TIMER_SYNC(EPWM0,
                           (1 << EADC0_TRIGGER_CH) |
                           (1 << EADC1_TRIGGER_CH) |
                           (1 << EADC2_TRIGGER_CH),
                           EPWM_SSCTL_SSRC_EPWM0);
}
```

PDMA 初始化

PDMA0 的通道 CH0/CH1/CH2 分別用於搬運 EADC0/EADC1/EADC2 的採樣資料到記憶體，由 EADC 觸發。源位址分別為三個 EADC 的資料暫存器，目標位址分別為同一個陣列的成員 0/1/2 的位址，源位址固定，目標位址遞增。使能跨步功能，每搬運一個資料，源位址不變，目標位址跳過兩個地址，這樣就能實現直接將三個 EADC 的採樣資料搬入同一個陣列。最後設置通道 CH2 搬運完後產生中斷。

```
void PDMA_Init(void)
{
    /* Reset PDMA module */
    SYS_ResetModule(PDMA0_RST);

    /* Configure PDMA peripheral mode form EADC to memory */
    /* Open Channel 0 */
    PDMA_Open(PDMA0, 1 << EADC0_PDMA_CH);

    /* Transfer width = 16 bit and transfer count = ADC_SAMPLING_TIMES */
    PDMA_SetTransferCnt(PDMA0, EADC0_PDMA_CH, PDMA_WIDTH_16, ADC_SAMPLING_TIMES);

    /* Set source address and destination address */
    PDMA_SetTransferAddr(PDMA0,
                         EADC0_PDMA_CH,                              /* PDMA channel */
                         (uint32_t)&EADC0->DAT[EADC0_MODULE_NUM], /* Source address */
                         PDMA_SAR_FIX,                               /* Source address fix */
                         (uint32_t)DataBuffer,                       /* Destination address*/
                         PDMA_DAR_INC);                 /* Destination address increment */

    /* Set PDMA Stride Mode */
    /* Destination stride count = 2, Source stride count = 0, Transfer count of per
       Stride = 1 */
    PDMA_SetStride(PDMA0, EADC0_PDMA_CH, 2, 0, 0);

    /* Select PDMA request source as ADC RX */
    PDMA_SetTransferMode(PDMA0, EADC0_PDMA_CH, PDMA_EADC0_RX, FALSE, 0);

    /* Set PDMA as single request type for EADC */
    PDMA_SetBurstType(PDMA0, EADC0_PDMA_CH, PDMA_REQ_SINGLE, PDMA_BURST_4);

    /* Open Channel 1 */
    PDMA_Open(PDMA0, 1 << EADC1_PDMA_CH);

    /* Transfer width = 16 bit and transfer count = ADC_SAMPLING_TIMES */
    PDMA_SetTransferCnt(PDMA0, EADC1_PDMA_CH, PDMA_WIDTH_16, ADC_SAMPLING_TIMES);
```

```
    /* Set source address and destination address */
    PDMA_SetTransferAddr(PDMA0,
                         EADC1_PDMA_CH,                              /* PDMA channel */
                         (uint32_t)&EADC1->DAT[EADC1_MODULE_NUM], /* Source address */
                         PDMA_SAR_FIX,                               /* Source address fix */
                         (uint32_t)&DataBuffer[1],                   /* Destination address*/
                         PDMA_DAR_INC);                 /* Destination address increment */

    /* Set PDMA Stride Mode */
    /* Destination stride count is 2, Source stride count is 0, Transfer count of per
       stride is 1 */
    PDMA_SetStride(PDMA0, EADC1_PDMA_CH, 2, 0, 0);

    /* Select PDMA request source as ADC RX */
    PDMA_SetTransferMode(PDMA0, EADC1_PDMA_CH, PDMA_EADC1_RX, FALSE, 0);

    /* Set PDMA as single request type for EADC */
    PDMA_SetBurstType(PDMA0, EADC1_PDMA_CH, PDMA_REQ_SINGLE, PDMA_BURST_4);

    /* Open Channel 2 */
    PDMA_Open(PDMA0, 1 << EADC2_PDMA_CH);

    /* Transfer width = 16 bit and transfer count = ADC_SAMPLING_TIMES */
    PDMA_SetTransferCnt(PDMA0, EADC2_PDMA_CH, PDMA_WIDTH_16, ADC_SAMPLING_TIMES);

    /* Set source address and destination address */
    PDMA_SetTransferAddr(PDMA0,
                         EADC2_PDMA_CH,                              /* PDMA channel */
                         (uint32_t)&EADC2->DAT[EADC2_MODULE_NUM], /* Source address */
                         PDMA_SAR_FIX,                               /* Source address fix */
                         (uint32_t)&DataBuffer[2],                   /* Destination address*/
                         PDMA_DAR_INC);                 /* Destination address increment */

    /* Set PDMA Stride Mode */
    /* Destination stride count = 2, Source stride count == 0, Transfer count of per
       stride = 1 */
    PDMA_SetStride(PDMA0, EADC2_PDMA_CH, 2, 0, 0);

    /* Select PDMA request source as ADC RX */
    PDMA_SetTransferMode(PDMA0, EADC2_PDMA_CH, PDMA_EADC2_RX, FALSE, 0);

    /* Set PDMA as single request type for EADC */
    PDMA_SetBurstType(PDMA0, EADC2_PDMA_CH, PDMA_REQ_SINGLE, PDMA_BURST_4);

    /* Enable channel 2 transportation completed interrupt */
    PDMA_EnableInt(PDMA0, EADC2_PDMA_CH, PDMA_INT_TRANS_DONE);

    /* Enable NVIC for PDMA0 */
    NVIC_EnableIRQ(PDMA0_IRQn);
}
```

PDMA 中斷處理函數

當 CH2 傳輸完成後設置資料搬運完成標誌，用於主函數裡判斷 ADC 轉換完成。

```
void PDMA0_IRQHandler(void)
{
    ……
    else if (status & PDMA_INTSTS_TDIF_Msk)     /* done */
    {
        if (PDMA_GET_TD_STS(PDMA0) & (1 << EADC2_PDMA_CH))
        {
            g_u32IsTestOver = 1;
        }

        PDMA_CLR_TD_FLAG(PDMA0, 1 << EADC2_PDMA_CH);
    }
    ……
}
```

EADC 初始化

將 EADC0/EADC1/EADC2 配置為單端模式。設置採樣模組、觸發源以及使能 EADC 觸發 PDMA。

```
void EADC_Init(void)
{
    /* Configure EADC0 */
    /* Set input mode as single-end and enable the A/D converter */
    EADC_Open(EADC0, EADC_CTL_DIFFEN_SINGLE_END);

    /* Configure the sample module for analog input channel and enable EPWM0 trigger
       source */
    EADC_ConfigSampleModule(EADC0, EADC0_MODULE_NUM, EADC_EPWM0TG0_TRIGGER, EADC0_CH);

    /* Enable Sample Module PDMA transfer. */
    EADC_ENABLE_SAMPLE_MODULE_PDMA(EADC0, 1 << EADC0_MODULE_NUM);

    /* Configure EADC1 */
    /* Set input mode as single-end and enable the A/D converter */
    EADC_Open(EADC1, EADC_CTL_DIFFEN_SINGLE_END);

    /* Configure the sample module for analog input channel and enable EPWM0 trigger
       source */
    EADC_ConfigSampleModule(EADC1, EADC1_MODULE_NUM, EADC_EPWM0TG1_TRIGGER, EADC1_CH);

    /* Enable Sample Module PDMA transfer. */
    EADC_ENABLE_SAMPLE_MODULE_PDMA(EADC1, 1 << EADC1_MODULE_NUM);

    /* Configure EADC2 */
    /* Set input mode as single-end and enable the A/D converter */
    EADC_Open(EADC2, EADC_CTL_DIFFEN_SINGLE_END);

    /* Configure the sample module for analog input channel and enable EPWM0 trigger
       source */
    EADC_ConfigSampleModule(EADC2, EADC2_MODULE_NUM, EADC_EPWM0TG2_TRIGGER, EADC2_CH);

    /* Enable Sample Module PDMA transfer. */
    EADC_ENABLE_SAMPLE_MODULE_PDMA(EADC2, 1 << EADC2_MODULE_NUM);
```

```
}
```

主函數

```
int main(void)
{
    ……

    /* Init EPWM for EADC */
    EPWM0_Init();

    /* Init PDMA for EADC */
    PDMA_Init();

    ……

    /* EADC function test */
    while (1)
    {
        ……

        /* Reload PDMA configuration for next transmission */
        ReloadPDMA();

        EADC_Init();

        /* Trigger EPWM counter synchronous start by EPWM0 */
        EPWM_TRIGGER_SYNC_START(EPWM0);

        ……

        while (g_u32IsTestOver == 0)
        {
            ……
        }

        g_u32IsTestOver = 0;

        printf("Conversion result:\n");

        for (Index = 0; (Index) < (ADC_SAMPLING_TIMES * 3); Index++)
        {
            printf("%d  ", DataBuffer[Index]);
        }

        ……
    }

    ……
}
```

3. 軟體與硬體需求

3.1 軟體需求

- BSP 版本
  - M460_Series_BSP_CMSIS_V3.00.004
- IDE 版本
  - Keil uVision 5.40

3.2 硬體需求

- 電路元件
  - NuMaker- M467HJ V1.1
- 線路示意圖
  - 將波形發生器的輸出連接到 PB.0。
  - 將 UART0 TX (PB.13) pin 通過 UART 轉 USB 工具連接到 PC。
  - 將 UART0 RX (PB.12) pin 通過 UART 轉 USB 工具連接到 PC 。

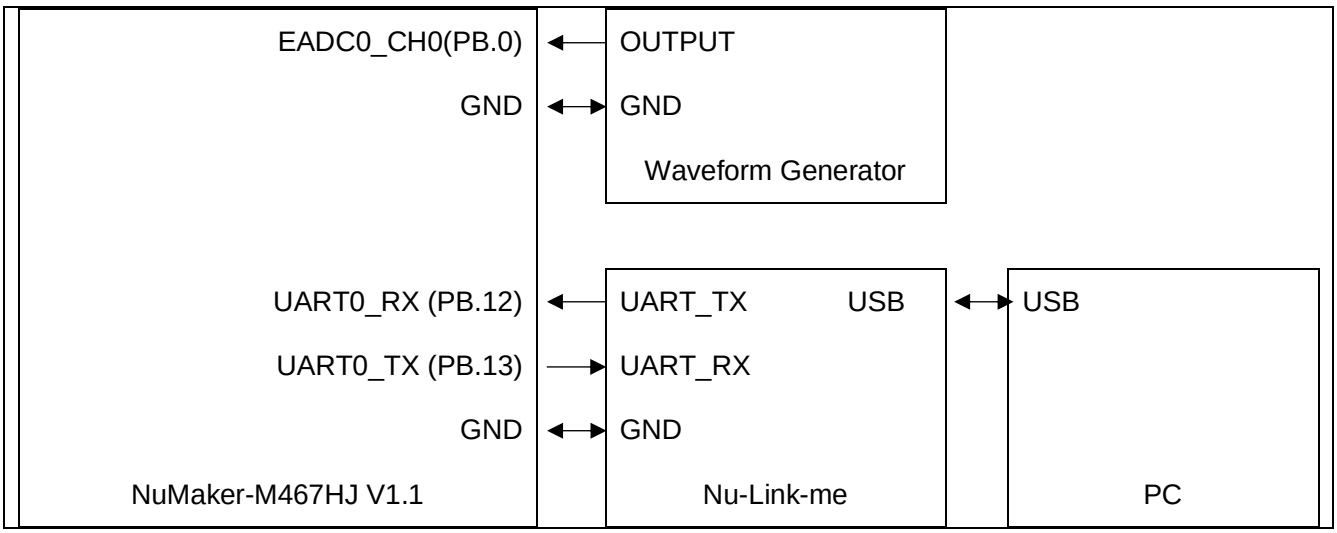


圖 3-1 線路示意圖

4. 目錄資訊

目錄結構如下圖所示。

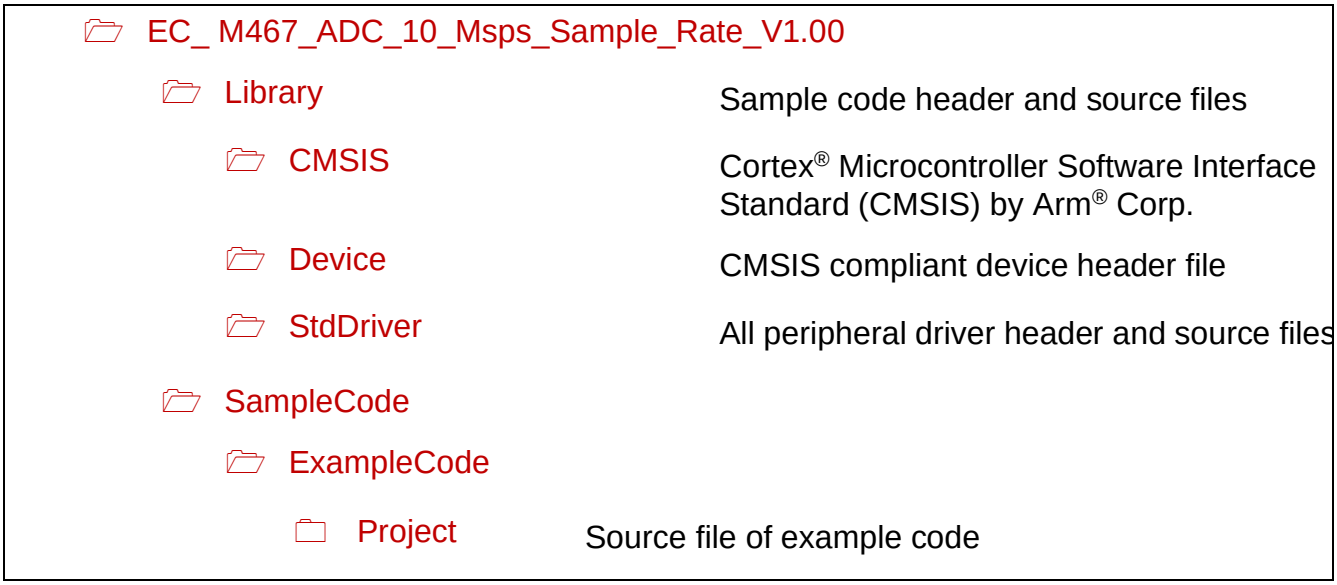


圖 4-1 目錄資訊

5. 範例程式執行

1. 根據目錄資訊章節進入 ExampleCode 路徑中的 KEIL 資料夾，按兩下 *M467_ADC_10_Msps_Sample_Rate.uvprojx*。
2. 進入編譯模式介面
   - 編譯
   - 下載代碼至記憶體
   - 進入 / 離開模擬模式
3. 進入模擬模式介面
   - 執行代碼

6. 修訂紀錄

Date	Revision	Description
2025.08.25	1.00	初始發佈。

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.