以 SPIM 讀取外部 SPI Flash 圖片

NuMicro® 32 位元系列微控制器範例代碼介紹

文件資訊

應用簡述	本範例程式使用 NuMaker-M467HJ 來實作以 SPIM 讀取存於微控制器外部 SPI flash 中的圖檔並顯示於 NuMaker-M467HJ 的 LCD 上，並比較兩線與四線模式的 SPIM 在讀取速度的差異。
BSP 版本	M460_Series_BSP_CMSIS_V3.00.004 emWin Software Package for M460 V7.1
開發平台	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

本範例使用 NuMaker-M467HJ 來實作以 SPIM(SPI Synchronous Serial Interface)讀取存於微控制器外部 SPI flash 中的圖檔並顯示於 NuMaker-M467HJ 的 LCD 上，同時亦比較兩線與四線模式的 SPIM 在讀取速度的差異。

在人機介面(HMI)或需要豐富圖形顯示的專案中，受限於微控制器內建 SRAM 空間不足，設計者時常會將大量的圖形資源（如介面圖標、背景圖片...等）儲存在外部的 SPI Flash 中。當需要顯示時，再透過 SPIM 介面高速傳輸到 MCU，並最終推送到 LCD 驅動器。

1.1 原理

本範例的核心在於利用 SPIM(SPI Synchronous Serial Interface)的多 I/O(Multi-I/O)能力，高效地將儲存於外部 SPI Flash 的圖像數據讀取出來，並顯示在 LCD 螢幕上。

1.1.1 SPIM 與 Direct Memory Mapping 功能

Nuvoton NuMicro M467 系列微控制器的 SPIM 模組是專為高速存取外部 SPI Flash 而設計的高效能介面。它不僅完整支援標準 SPI 模式，更提供了 Dual 和 Quad 的多 I/O 模式，同時具備強大的直接記憶體映射模式(Direct Memory Mapping Mode，簡稱 DMM mode)功能。

整個流程始於以 MCU 作為 Master，向外部 SPI Flash 發出讀取指令與圖像的目標位址。採用 SPIM 的 DMM mode，其運作原理是，SPIM Controller 會將外部的 SPI Flash 記憶體空間映射到 MCU 的內部位址空間中，如表 1-1。對於 CPU 而言，這塊外部 Flash 就像是內部記憶體的一部分，其存取位址被統一納入 MCU 的記憶體分區規劃。

Memory Address	Size	Functions
0x40007000-0x40007FFF	4 Kbytes	SPIM control registers：儲存 SPIM 配置相關參數
0x100000-0x020FFFFF	32 Mbytes	外部 SPI flash 被映射到的 MCU 內部位址空間(註：此地址空間僅適用於 Flash 大小為 1024 Kbytes 的型號)

表 1-1 SPIM 記憶體映射位址

一旦進入此模式，當 CPU 嘗試從這塊映射的記憶體位址(0x100000)執行讀取操作時，SPIM Controller 會自動地執行完整的 SPI Flash 讀取時序，包括發送讀取指令、位址，並啟動高速數據傳輸，而不需要 CPU 介入或額外的函數呼叫。這極大地簡化了圖像數據的獲取流程，尤其適用於需要頻繁讀取圖像數據的應用場景。

SPIM 的相關軟體設定請參考代碼介紹。

1.1.2 Dual Mode and Quad Mode

要提升效能，SPI Flash 讀取效率亦是一大關鍵，以標準 SPI 介面是使用一對數據線(MISO 與 MOSI)進行數據輸入和輸出，為了增加效率，大多數的 SPI Flash 能支援雙線模式(Dual Mode)與四線模式(Quad Mode)的 Multi-I/O 模式。

在 Dual Mode 下，傳統的 MOSI 和 MISO 腳位被配置成雙向的 IO0 與 IO1 數據線，讓 SPIM 在每個時鐘周期內並行傳輸 2 個位元(2-bit)，數據吞吐量便提高一倍。更進一步，Quad Mode 模式則將 Flash 晶片上的寫保護(WP#)和暫停(HOLD#)腳位重新定義為額外的數據線 IO2 和 IO3。當 Flash 晶片被設定啟用 Quad Enable (QE)位元後，SPIM 便可以在每個時鐘週期內利用這四條數據線並行傳輸 4 個位元(4-bit)，理論頻寬提升至 Dual Mode 的兩倍。

在設定 SPIM 進入 DMM 模式並啟用 Dual/Quad Mode 時，必須注意 MCU 與外部 SPI Flash 的指令兼容性，不同的 Flash 晶片廠商或型號，其高速連續讀取指令(Fast Read Opcode)可能不同。

此範例所使用的開發板為 NuMaker-M467HJ V1.1，從電路圖可以看到選用的是 Winbond SPI flash 型號為 W25Q32JVSSIQ，可以支援 M467 series SPIM 所設定的讀取指令，如指令 0x3B 為 Fast Read Dual Output、指令 0xEB 為 Fast Quad Read，並且如 Dummy clocks 等，需對照 SPI flash 規格所述的波型設定，如圖 1-1，以符合時序要求。

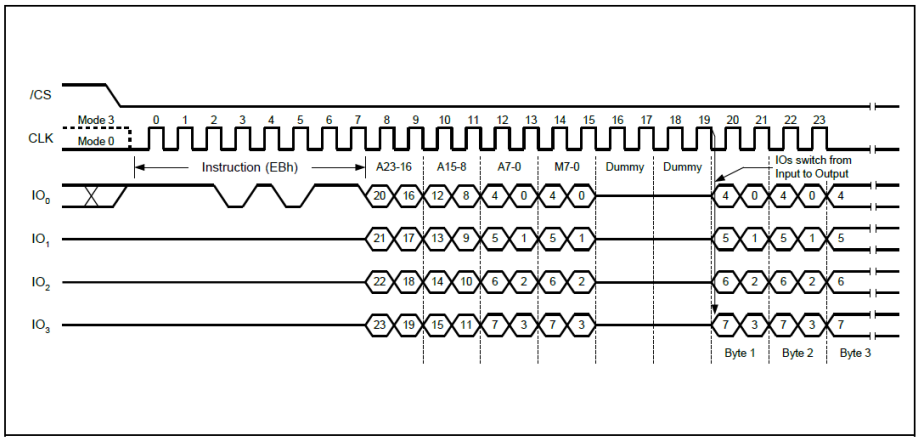

Figure 24. Fast Read Quad I/O Instruction (M7-M0 should be set to Fxh)

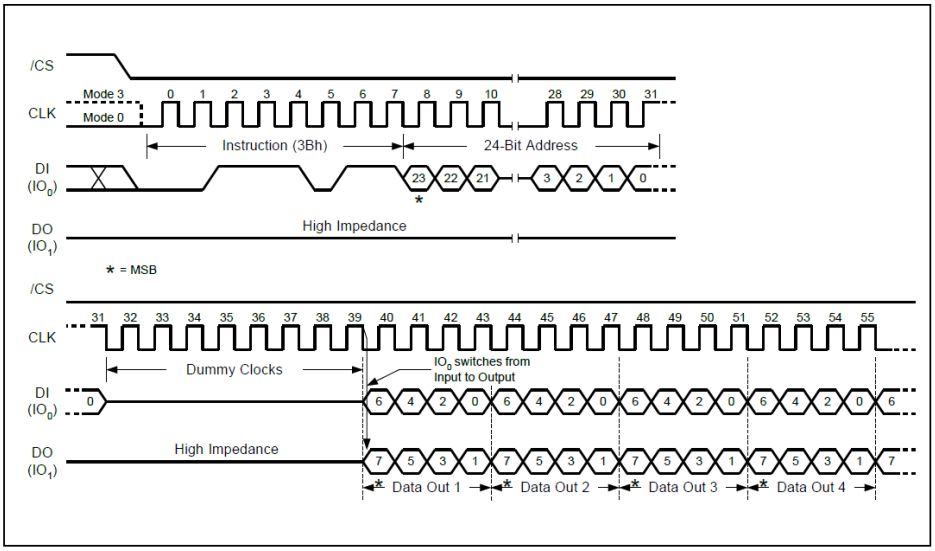

Figure 18. Fast Read Dual Output Instruction

Source: Datasheet of Winbond W25Q32JV

圖 1-1 SPI Flash 規格書

而硬體腳位配置則如表 1-2，電路連接方式如圖 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 腳位配置表

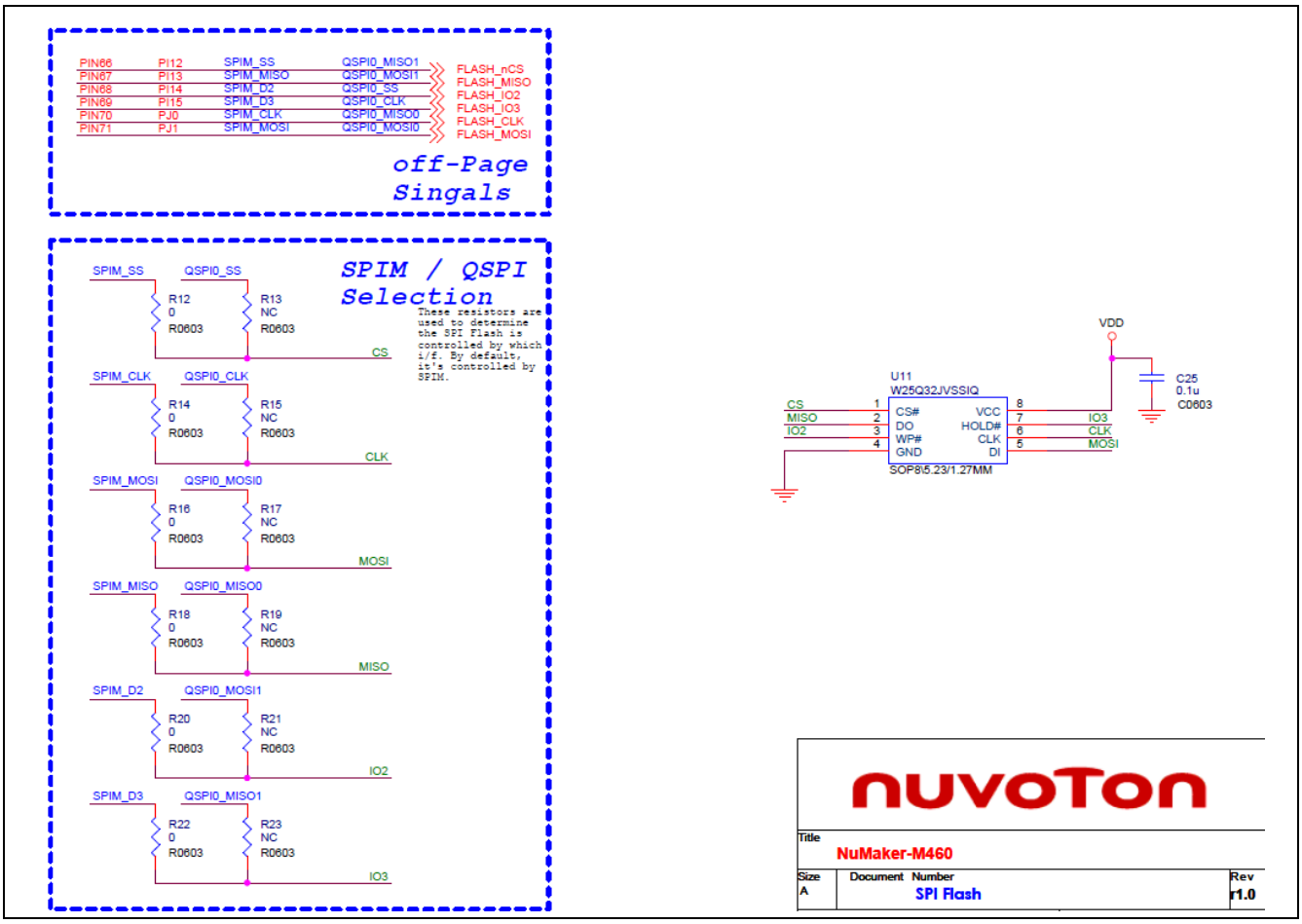

圖 1-2 SPI Flash 電路圖連接方式

1.2 執行結果

1.2.1 燒錄

在燒錄 SPI flash 時，需要於燒錄設定中指定燒錄腳位，以 Keil MDK 為範例，需參考圖 1-3 設定，按照表 1-2 腳位規劃，選定對應的腳位。

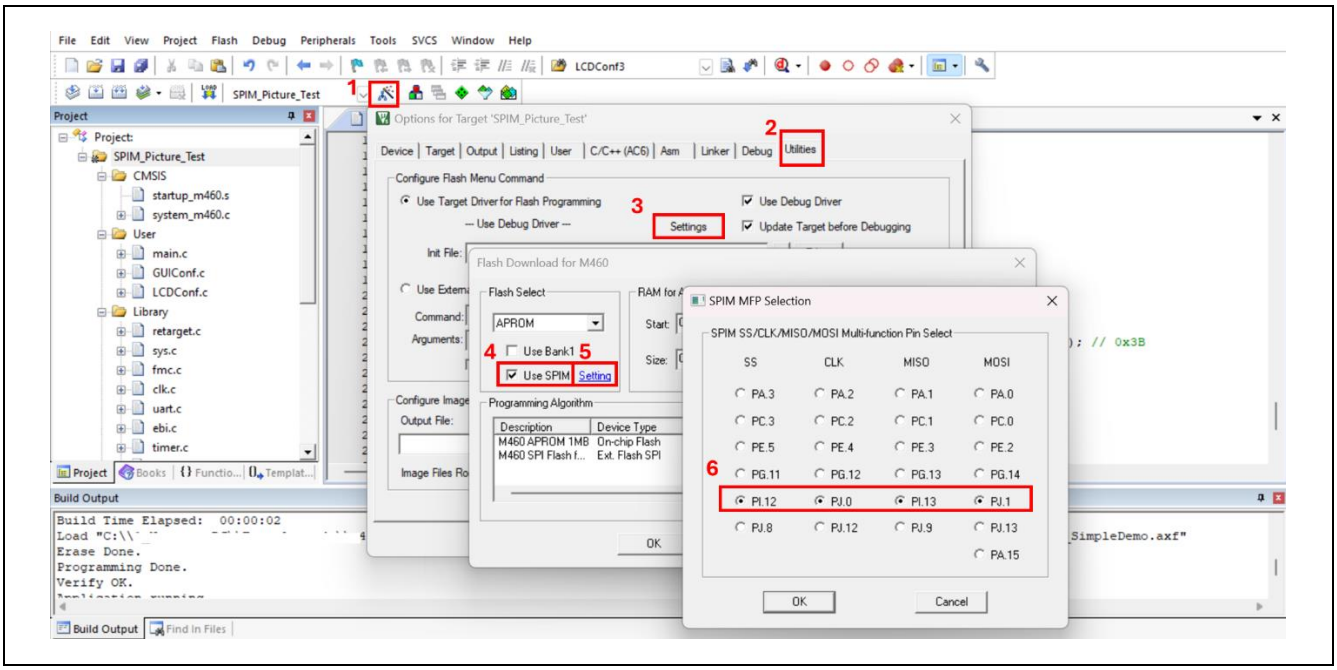

圖 1-3 SPI Flash 燒錄設定

1.2.2 執行

執行編譯和燒錄的結果如圖 1-4 與圖 1-5 所示。初始化完成後，使用 PC UART 終端機輸入 0 或 1 以進入 Quad Mode 或 Dual Mode 兩種測試模式，執行完成後，會在 LCD 上看到圖片成功顯示，並透過打印出的資訊查看從 SPI Flash 載入一張解析度為 480x272 的圖所需耗費的時間。

透過軟體時間的計算，可以看到繪製一張解析度為 480x272 的圖片，Quad Mode 僅需約 18 ms，而 Dual Mode 需約 35 ms，證實 Quad Mode 的數據頻寬優勢，能大幅縮短圖像載入的時間，對於人機介面的流暢度亦會有決定性的影響。

```
+-----------------------------------------------------------------------+
  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. ==|
```

圖 1-4 執行編譯和燒錄結果 – 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. ==
```

圖 1-5 執行編譯和燒錄結果 – Dual Mode

另外，亦可使用市面上的邏輯分析儀實際量測圖片讀取的時間，此範例使用儀器 Acute MSO3124V，如圖 1-6，選擇 Serial Flash Bus Decode 模式分析，可以得到測量結果如圖 1-7 所示，除了測量時間，亦可從 Bus Decode 結果驗證是否有正確設定到對應的模式。

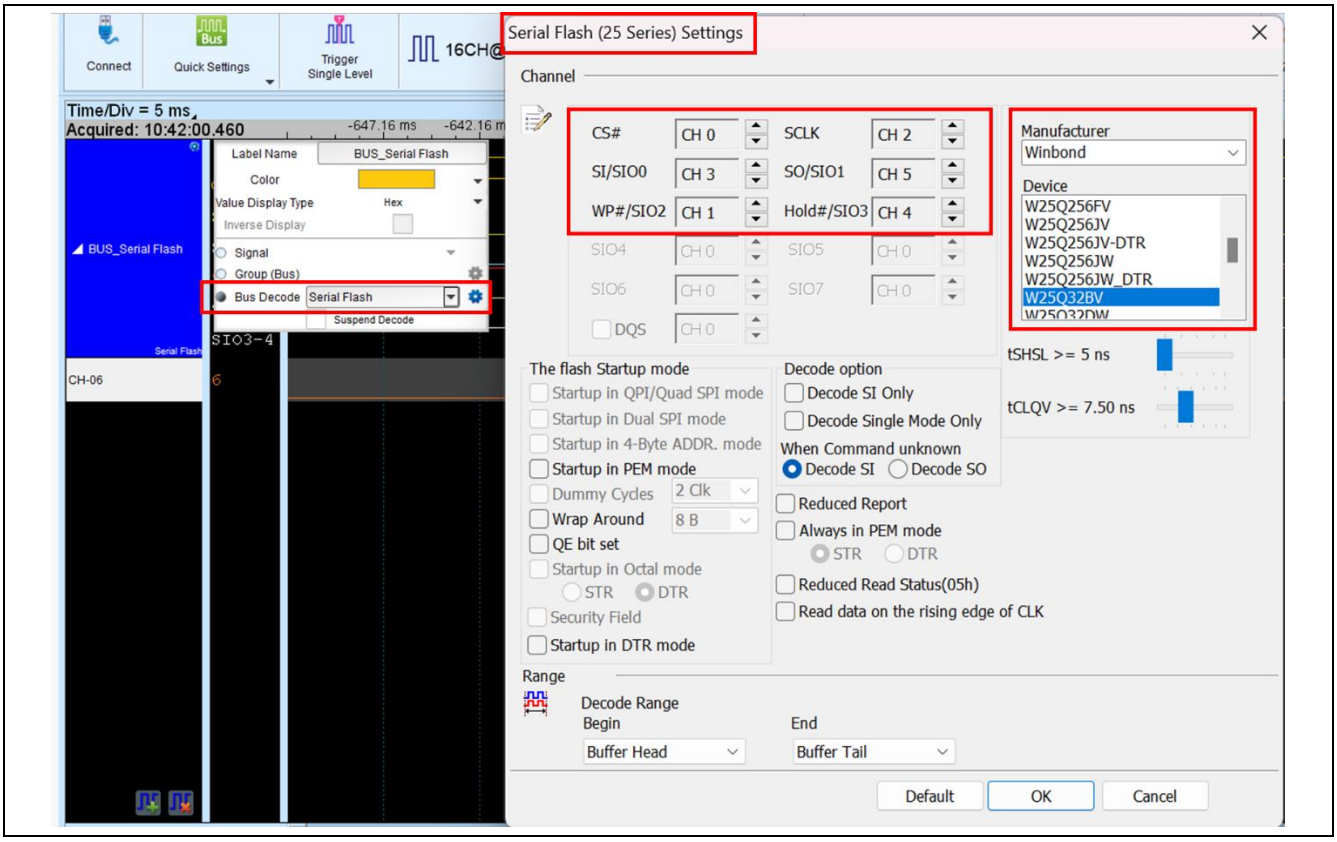

圖 1-6 邏輯分析儀 Bus Decode 設定

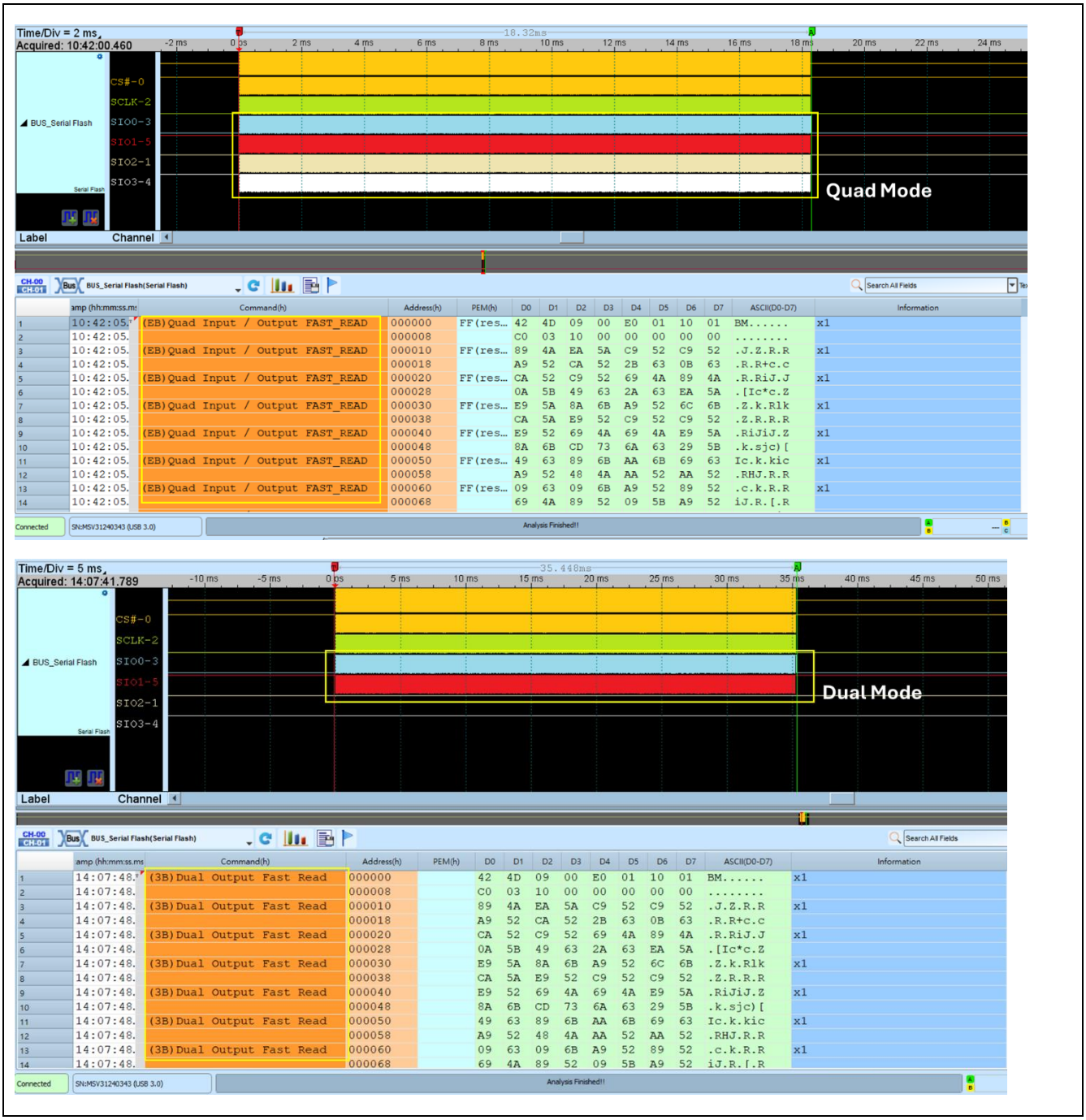
Time/Div = 2 ms
Acquired: 10:42:00.460
CS#-0
SCLK-2
SIO0-3
SIO1-5
SIO2-1
SIO3-4
BUS_Serial Flash
Quad Mode
Label
Channel
BUS_Serial Flash(Serial Flash)
Time/Div = 5 ms
Acquired: 14:07:41.789
Dual Mode

圖 1-7 邏輯分析結果

2. 代碼介紹

SPIM 的主設定代碼位於 *main.c*。

SPIM 初始化分為腳位設置與參數設置。腳位設置位於 sys_init()函式中，對應表 1-2 設定 SPIM multi-function pin，其中 clock pin(PJ0)設定為 Schmitt trigger 提高抗雜訊能力，以確保時脈訊號穩定性，並對 SPIM I/O 設定為高轉換率模式，使電壓變化速度更快，在高速通訊時可以保持訊號完整性。

```
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 的參數設置，則於 main()函示中，設定說明如下：

1. SPIM 時脈：SPIM_SET_CLOCK_DIVIDER 為分頻器設定，計算公式為 HCLK/(Divider)^2，此範例 HCLK 頻率為 200 MHz，因此值為 2 表示 SPIM 時脈為 50 MHz。
2. 接收時脈延遲：SPIM_SET_RXCLKDLY_RDDLYSEL 與 SPIM_SET_RXCLKDLY_RDEDGE 為接收時脈延遲設定，此處設定為無延遲。
3. Dummy Cycles：SPIM_SET_DCNUM 為在 MCU 發送完指令後到 SPI Flash 開始輸出有效資料前的緩衝時間，根據圖 1-1 規格書波型，若指令為 0xEB(Fast Read Quad Output)，需有 4 個 clocks 的 dummy cycles，而指令為 0x3B(Fast Read Dual Output)，需有 8 個 clocks 的 dummy cycles。

4. 初始化 SPI Flash 並驗證 ID：SPIM_InitFlash 是對 SPI Flash 初始化的 API，也透過 SPIM_ReadJedecId 印出 SPI Flash 的 JEDEC ID。

5. SPIM 內部快取：DMM mode 下可以啟用 SPIM 內部 16 Kbytes 快取，此快取區塊會成為在 MCU 與 SPI Flash 間的緩衝區，若連續存取相近的數據，將會直接從內部快取提供，能再減少到 SPI Flash 存取的次數。

6. 加密位元：Chiper Disable Bit 是決定是否要對傳輸的數據進行即時加解密， 由於加解密亦會需要額外處理時間，因此此範例於巨集定義 SPIM_CIPHER_ON 為 0，關閉加解密功能。

7. 4 bytes mode ： W25Q32JVSSIQ 並無支援 4 bytes mode ，因此巨集定義 USE_4_BYTES_MODE 為 0，使用 3 bytes mode。

8. SPIM DMM mode：SPIM_EnterDirectMapMode 是 SPIM 進入 DMM mode 的 API，其中的 u32RdCmd 參數即是用於指定採用的 I/O 模式，如指令 0xEB 為 Fast Read Quad Output，指令 0x3B 為 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);
   }
   …
}
```

最後，藉由 emWin Library 的 API GUI_DrawStreamedBitmapAuto()繪製一張解析度 480*272 的圖，在繪圖前後紀錄時間戳記，並由 UART 打印出計算結果。

```
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 記憶體空間虛擬地映射到 MCU 的 0x100000 位址空間中，在 Keil MDK 可以透過 Scatter File 精確地規劃記憶體區塊，開啟 Scatter File 的步驟如圖 2-1，設定範例如下，在此範例中定義了起始位置為 0x100000 的 LOAD_ROM_2 區域，也就是 SPI Flash 虛擬映射的位址，而圖片 Ebike_Demo_BG0.o 被明確分配到此區塊中，使得 MCU 可以直接到 0x100000 存取此圖片，實現高效率的外部數據存取功能。

```
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)
    }
}
```

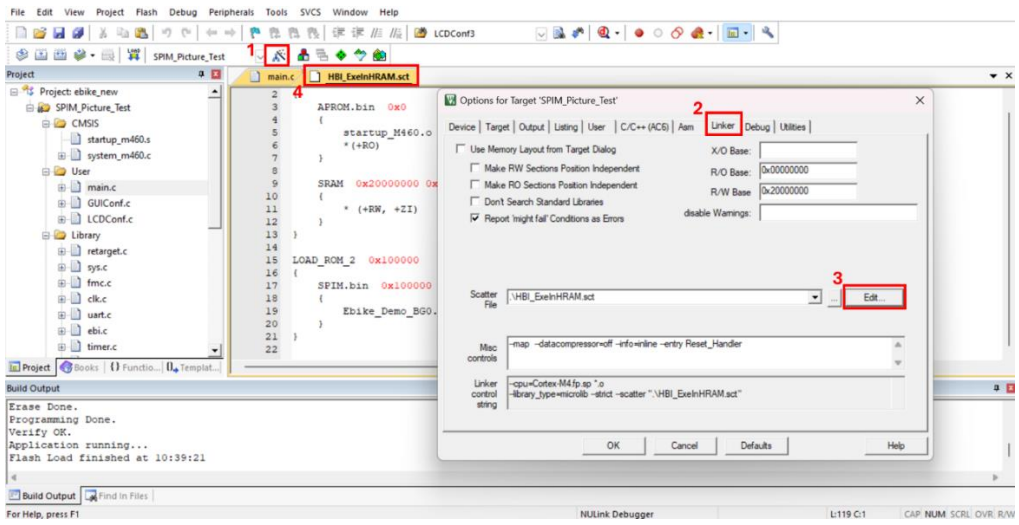

圖 2-1 Scatter File 開啟方式

3. 軟體與硬體需求

3.1 軟體需求

- 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 硬體需求

- 電路元件
  - NuMaker-M467HJ V1.1

4. 目錄資訊

目錄結構如下圖所示。

EC_M467_SPIM_Image_Quad_Dual_Test_V1.00	
Library	Sample code header and source files
CMSIS	Cortex® Microcontroller Software Interface Standard (CMSIS) by Arm® Corp.
Device	CMSIS compliant device header file
StdDriver	All peripheral driver header and source files
ThirdParty	Header files for emWin project
SampleCode	
ExampleCode	
SPIM_Image_Quad_Dual_Test	Source file of example code

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，雙擊 *SPIM_Image_Quad_Dual_Test.uvprojx*。
2. 進入編譯模式介面
   - 編譯
   - 下載代碼至記憶體
   - 進入 / 離開除錯模式
3. 進入除錯模式介面
   - 執行代碼

6. 修訂紀錄

Date	Revision	Description
2025.11.01	1.00	初始發布。

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.