PDMA 搬移 FLASH 數據到 TFT 屏顯示

NuMicro® 32 位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 M031 系列芯片將 SPI FLASH 中圖片數據通過 PDMA 搬移到 TFT 屏顯示
BSP 版本	M031_Series_BSP_CMSIS_V3.06.000
開發平台	NuMaker-M031KI V1.0 NuTFT-SPI_320x240 Daughter Board_V1.3

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

本範例代碼使用 M031KIAAE 的 SPI、USPI 外設 PDMA 功能將 SPI FLASH 中圖片數據搬移到 TFT 屏進行顯示。

首先通過 SPI 外設將圖片數據寫入到 FLASH 中，然後通過 SPI 外設 RX PDMA 功能將 FLASH 中圖片數據傳輸到 USPI 外設 TX，由 TX 驅動 TFT 屏進行顯示。

1.1 原理

320x240 TFT 以 RGB565 模式顯示，顯示一屏圖片需要 153600 (320*240*2) 個位字節數據。FLASH (W25Q16) 存儲容量為 2M Bytes，上電後將圖片數據保存到 FLASH。M031KIAAE 通過 SPI 與 FLASH 進行通信，通過 USPI 與 TFT 控制芯片 (ILI9341) 進行通信。SPI 外設及 USPI 外設都支持 PDMA 功能。 通過使能 SPI 外設 RX PDMA 功能實現將 FLASH 中的圖片數據直接傳輸到 USPI 外設 TX，由 TX 驅動 TFT 進行顯示。除了設置 FLASH 讀數據起始位址和設置 TFT 顯示初始位置之外，其他數據傳輸都由 PDMA 完成，進而提升 CPU 的運行效率。

1.2 執行結果

本範例代碼執行編譯和燒錄後，TFT 屏顯示如圖 1-1 所示。



圖 1-1 執行編譯和燒錄結果

串口列印消息如圖 1-2 所示。

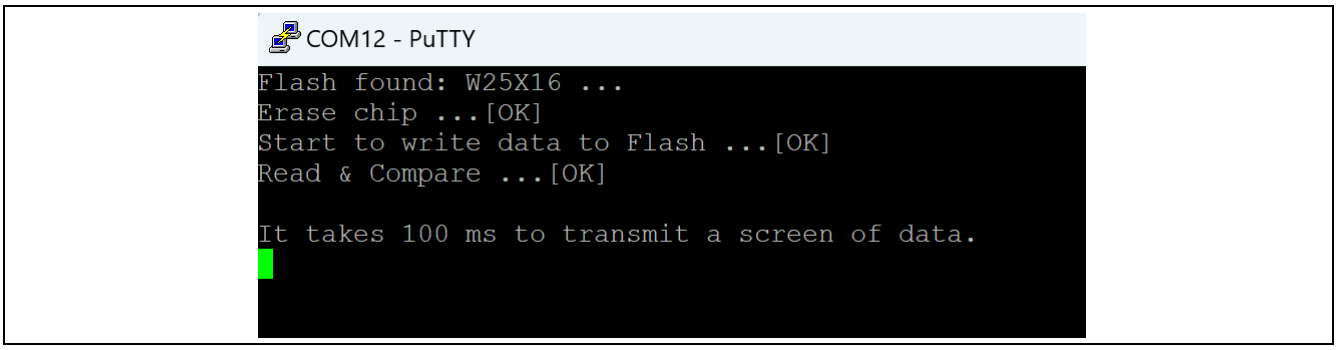


圖 1-2 串口列印消息

2. 代碼介紹

圖片數據數組定義，代碼位於 *Image.c*。

```
const uint8_t u8Image[DATA_PER_SCREEN] =
{
    0XF7,0XBE,0XF7,0XDE,0XF7,0XDE,0XF7,0XDE,0XF7,0XDE,0XF7,0XBE,0XF7,0XBE,0XF7,0XBE,
    0XEF,0X9D,0XE7,0X7C,0XE7,0X7C,0XE7,0X7C,0XE7,0X7C,0XE7,0X7C,0XE7,0X7C,0XE7,0X7C,
    0XE7,0X7C,0XE7,0X7C,0XE7,0X7C,0XE7,0X9C,0XE7,0X9D,0XE7,0X9D,0XE7,0X9C,0XE7,0X9C,
    0XE7,0X7C,0XE7,0X9C,0XE7,0X9C,0XE7,0X9D,0XE7,0X9D,0XE7,0X9D,0XE7,0X9D,0XEF,0X9D,
    0XEF,0X9D,0XEF,0X9E,0XEF,0XBD,0XEF,0XBE,0XEF,0XBE,0XEF,0XBD,0XEF,0XBD,0XEF,0X9D,
    ……
    0X3B,0X67,0X32,0XC6,0X2A,0X24,0X19,0X83,0X19,0X83,0X21,0XA4,0X2A,0X05,0X2A,0X25,
    0X21,0XE4,0X21,0XC3,0X21,0XC3,0X21,0X83,0X19,0X43,0X21,0X63,0X2A,0X04,0X32,0X45,
    0X2A,0X04,0X32,0X65,0X3A,0XC7,0X2A,0X25,0X19,0X83,0X21,0XC4,0X29,0XE5,0X21,0XA4,
    0X19,0X43,0X19,0X43,0X21,0X83,0X21,0X63,0X19,0X22,0X19,0X02,0X10,0XE2,0X10,0XC2,
    0X10,0XC2,0X10,0XC2,0X10,0XC2,0X10,0XC2,0X10,0XC2,0X10,0XC2,0X10,0XC2,0X10,0XA2,
};
```

圖片數據數組大小定義，代碼位於 *Image.h*。

```
#define DATA_PER_SCREEN   (153600UL)
extern const uint8_t u8Image[DATA_PER_SCREEN];
```

主函數包括對系統，對 UART0、USCI_SPI0、PWM、SPI0，對 TFT 驅動芯片，對 SPI FLASH，對 Timer0 初始化，最後執行主循環。代碼位於 main 函數。

```
    int main()
    {
        /* System initialization */
        SYS_Init();
        /* Initial UART0 to 115200-8n1 for print message */
        UART_Open(UART0, 115200);
        /* Initial USCI_SPI0 */
        USCI_SPI_Init();
        /* Initial PWM for backlight brightness control */
        PWM_Init();
        /* Initial SPI0 */
        SPI_Init();
        /* Initial ILI9341 LCD Panel Hardware */
        Drv_ILI9341_LCD_Init();
       /* Identify flash type, erase flash, write picture datas to flash,
       check if write successful */
        SpiFlash_Setting();
         /* Initial Timer0 */
         Timer0_Init();
        while (1)
        {
            /* Data read from FLASH is sent to TFT via PDMA */
            Copy_FlashData_To_TFT_WithPDMA();
        }
    }
```

主循環函數，用於實現將 FLASH 中數據搬移到 TFT 顯示。當一屏數據傳輸完成後，串口輸出傳輸所花的時間。代碼位於 Copy_FlashData_To_TFT_WithPDMA 函數。

```
/* Data read from FLASH is sent to TFT via PDMA */
void Copy_FlashData_To_TFT_WithPDMA(void)
{
    uint32_t u32RegValue, u32Abort;

    /* Data Width is swtich to 8 Bits */
    Switch_DataWidth_Is_Byte(TRUE);
    /* Reset LCD display position */
    LCD_Set_Pos();
    /* Reset read start address */
    SpiFlash_SetReadAddress(0x000000);
    /* Data Width is swtich to 16 Bits */
    Switch_DataWidth_Is_Byte(FALSE);
    /* Init PDMA */
    PDMA_Init();
    /* CS = 0 : select chip */
    USPI_CS_CLR;
    /* DC = 1 : write data */
    LCM_DC_SET;
    /* Enable SPI RX PDMA function */
    SPI_TRIGGER_RX_PDMA(SPI0);
    while (1)
    {
        /* Get interrupt status */
        u32RegValue = PDMA_GET_INT_STATUS(PDMA);
        /* Check the PDMA transfer done interrupt flag */
        if (u32RegValue &PDMA_INTSTS_TDIF_Msk)
        {
            /* Check the PDMA transfer done flags */
            if ((PDMA_GET_TD_STS(PDMA) & (0x01 << SPI_MASTER_RX_DMA_CH)) ==
                    (0x01 << SPI_MASTER_RX_DMA_CH))
            {
                /* Clear the PDMA transfer done flags */
                PDMA_CLR_TD_FLAG(PDMA, (0x01 << SPI_MASTER_RX_DMA_CH));
                /* Enable SPI RX PDMA function */
                SPI_TRIGGER_RX_PDMA(SPI0);
                u8TransferCnt++;
                if (u8TransferCnt >= TRANSFER_TIMES)
                {
                    u8TransferCnt = 0;
                    printf("\nIt takes %d ms to transmit a screen of data. \n",
                           (u8TimerCnt*10));
                    /* Disable Timer0 NVIC */
                    NVIC_DisableIRQ(TMR0_IRQn);
                    while (1);
                }
            }
        }
        /* Check the DMA transfer abort interrupt flag */
        if (u32RegValue &PDMA_INTSTS_ABTIF_Msk)
        {
            /* Get the target abort flag */
            u32Abort = PDMA_GET_ABORT_STS(PDMA);
            /* Clear the target abort flag */
            PDMA_CLR_ABORT_FLAG(PDMA, u32Abort);
            printf("\nabort happen\n");
```

```
        }

        /* Check the DMA time-out interrupt flag */
        if (u32RegValue &(PDMA_INTSTS_REQTOF0_Msk | PDMA_INTSTS_REQTOF1_Msk))
        {
            /* Clear the time-out flag */
            PDMA->INTSTS = u32RegValue &(PDMA_INTSTS_REQTOF0_Msk |
                                         PDMA_INTSTS_REQTOF1_Msk);
            printf("\ntime-out happen\n");
        }
    }
}
```

設置命令的時候將傳輸位寬設為 8 bits，傳輸数据的時候將位寬設置為 16 bits。16 bits 傳輸位寬比 8 bits 傳輸位寬更高效。代碼位於 Switch_DataWidth_Is_Byte 函數。

```
void Switch_DataWidth_Is_Byte(_Bool bFlag)
{
    if (bFlag)
    {
        /* USPI0 Transfer Data width is 8 Bits */
        USPI0->LINECTL &= ~USPI_LINECTL_DWIDTH_Msk;
        USPI0->LINECTL |= (0x08 << USPI_LINECTL_DWIDTH_Pos);
        /* SPI0 Transfer Data width is 8 Bits*/
        SPI0->CTL &= ~SPI_CTL_DWIDTH_Msk;
        SPI0->CTL |= (0x08 << SPI_CTL_DWIDTH_Pos);
    }
    else
    {
        /* USPI0 Transfer Data width is 16 Bits */
        USPI0->LINECTL &= ~USPI_LINECTL_DWIDTH_Msk;
        USPI0->LINECTL |= (0 << USPI_LINECTL_DWIDTH_Pos);
        /* SPI0 Transfer Data width is 16 Bits */
        SPI0->CTL &= ~SPI_CTL_DWIDTH_Msk;
        SPI0->CTL |= (0x10 << SPI_CTL_DWIDTH_Pos);
    }
}
```

Timer0 初始化，設置每隔 10 ms 進入一次中斷，計數器初始化為 0，代碼位於 Timer0_Init 函數中。定時器中斷函數，每進入中斷對計數器進行加 1，代碼位於 TMR0_IRQHandler 函數。

```
void Timer0_Init(void)
{
    /* Open Timer0 in periodic mode, enable interrupt
    and 1 interrupt tick per 10 millisecond */
    TIMER_Open(TIMER0, TIMER_PERIODIC_MODE, 100);
    TIMER_EnableInt(TIMER0);
    /* Start Timer0 counting */
    TIMER_Start(TIMER0);
    /* Clear timer counter */
    u8TimerCnt = 0;
    /* Enable Timer0 NVIC */
    NVIC_EnableIRQ(TMR0_IRQn);
}

void TMR0_IRQHandler(void)
{
    if (TIMER_GetIntFlag(TIMER0) == 1)
    {
        /* Clear Timer0 time-out interrupt flag */
        TIMER_ClearIntFlag(TIMER0);
```

```
        /* Timer counter plus one */
        u8TimerCnt++;
    }
}
```

因為 TFT 以 RGB565 模式顯示，一個像素點需要 16 bits 來表示，且 MCU 與 TFT 之間傳輸基本單位也是 16 bits，因此傳輸一屏圖片即 320x240 像素點需要傳輸 76800x16 bits 數據。因為一個 PDMA table 傳輸數量是用 16 bits 定義，最大傳輸數據為 65536 x16 bits，因此傳輸一屏圖片需要分 2 次傳輸，分別傳輸 38400 x16 bits 數據。PDMA 操作模式設置為 Scatter-Gather。傳輸數量設置為 38400。傳輸位寬設置為 16 bits。傳輸類型設置為 signal。源地址固定為 SPI0 RX 寄存器，目標位址固定為 USPI0 TX 寄存器。代碼位於 PDMA_Init 函數。

```
void PDMA_Init(void)
{
    /* Descriptor table 1 configuration */
    DMA_DESC[0].ctl = ((TRANSFER_LEN - 1) << PDMA_DSCT_CTL_TXCNT_Pos) | PDMA_WIDTH_16 |
                      PDMA_SAR_FIX | PDMA_DAR_FIX | PDMA_REQ_SINGLE | PDMA_OP_SCATTER;
    DMA_DESC[0].src = (uint32_t) &SPI0->RX;
    DMA_DESC[0].dest = (uint32_t) &USPI0->TXDAT;
    DMA_DESC[0].offset = (uint32_t) &DMA_DESC[1] - (PDMA->SCATBA);
    /* Descriptor table 2 configuration */
    DMA_DESC[1].ctl = ((TRANSFER_LEN - 1) << PDMA_DSCT_CTL_TXCNT_Pos) | PDMA_WIDTH_16 |
                      PDMA_SAR_FIX | PDMA_DAR_FIX | PDMA_REQ_SINGLE | PDMA_OP_BASIC;
    DMA_DESC[1].src = (uint32_t) &SPI0->RX;
    DMA_DESC[1].dest = (uint32_t) &USPI0->TXDAT;
    DMA_DESC[1].offset = 0;
    /* Open PDMA channel 1 for SPI RX */
    PDMA_Open(PDMA, (1 << SPI_MASTER_RX_DMA_CH));
    /* Configure PDMA transfer mode */
    PDMA_SetTransferMode(PDMA, SPI_MASTER_RX_DMA_CH, PDMA_SPI0_RX, 1, (uint32_t)
                         &DMA_DESC[0]);
}
```

3. 軟體與硬體需求

3.1 軟體需求

- BSP 版本
  - M031_Series_BSP_CMSIS_V3.06.000
- IDE 版本
  - Keil uVersion 5.36

3.2 硬體需求

- 電路元件
  - NuMaker-M031KI V1.0
  - NuTFT-SPI_320x240 Daughter Board_V1.3
- 線路示意圖
  - FLASH 和 TFT Control Chip 位於 NuTFT-SPI_320x240 Daughter Board_V1.3 上
  - M031KIAAE 位於 NuMaker-M031KI V1.0 上

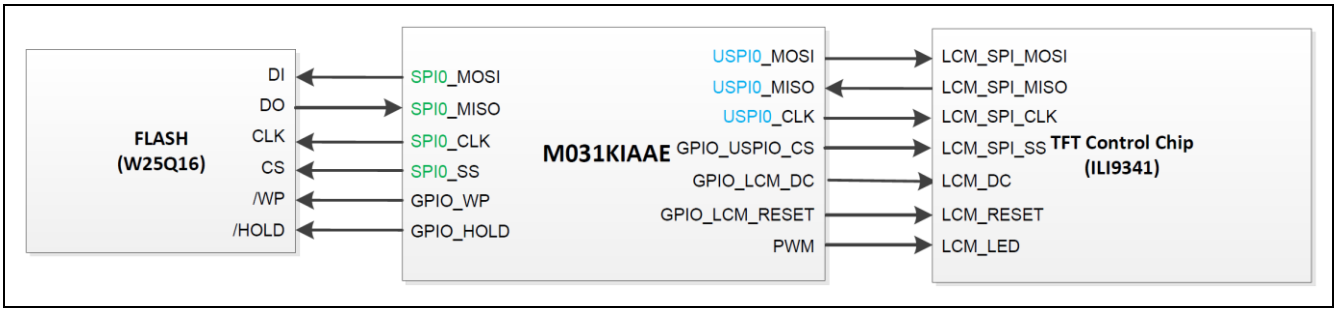


圖 3-1 線路示意圖

4. 目錄資訊

EC_ M031_PDMA_FLASH_TO_TFT_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
SampleCode	
ExampleCode	
Project	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 *M031_PDMA_FLASH_TO_TFT.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
2024.07.10	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.