M251 DMX512 控制範例

NuMicro® 32 位元系列微控制器範例代碼

文件資訊

應用簡述	本範例代碼使用 M251 實現 DMX512 控制 ARGB2 LED
BSP 版本	M251_M252_M254_M256_M258_Series_BSP_CMSIS_V3.02.007
開發平台	NuMaker-M251KG 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

DMX512 協議是由美國舞臺燈光協會（USITT）提出的一種數據調光協議，它給出了燈光控制器與燈具設備之間通信的標準協議。應用它能夠對舞臺，劇場，攝影棚等場所的調光器及其他控制設備進行數位控制，適用於一主多從式控制系統，其連線簡單可靠性高。本文描述如何使用 M251 UART 解碼 DMX512 數據封包，並控制 ARGB2 LED 燈。

1.1 原理

1.1.1 ARGB2 LED 數據傳輸時間 (TH + TL = 1.2 µs ± 600 ns)

ARGB2 LED 時序圖如圖 1-1 與圖 1-2 所示。邏輯 ‘0’ 定義為高準位 0.3 µs ± 0.15 µs 與低準位 0.9 µs ± 0.15 µs，邏輯 ‘1’ 定義為高準位 0.9 µs ± 0.15 µs 與低準位 0.3 µs ± 0.15 µs。

在使用 PSIO 控制 ARGB2 LED，必須先設定好時序規則(slot)，按照預期的時序運作，將時間基本單位劃分約 0.3 µs 可以造出所需的波形。時序計算公式如下：

- PSIO clock source 選擇 PCLK1，PCLK1 clock 計算方式 HIRC (48 MHz) / 4 = 12 MHz
- PSIO clock 使用 PCLK1 除頻 4 = 3 MHz
- PSIO one slot cycle time = 1 / 3 MHz = 0.33 µs

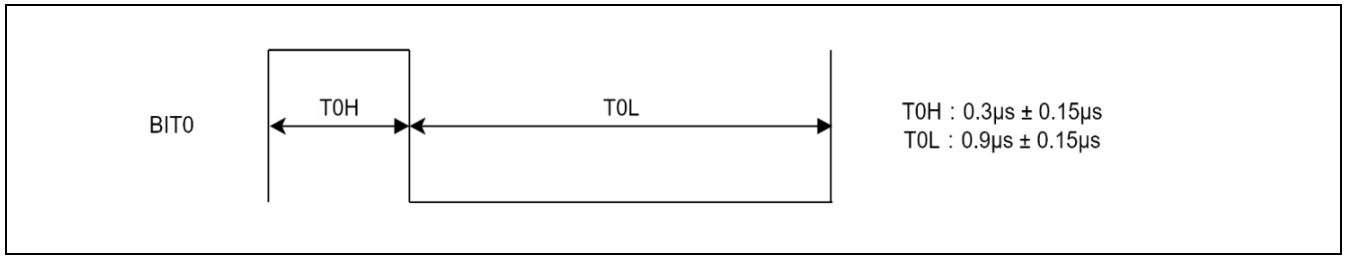

圖 1-1 邏輯 ‘0’ (bit 0)

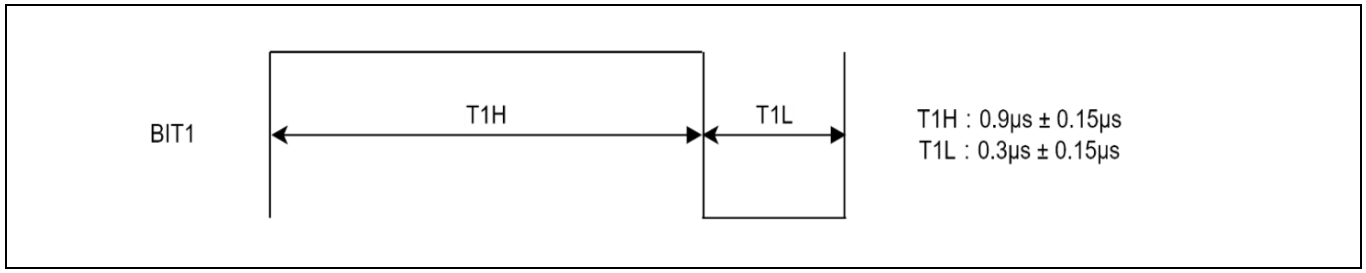

圖 1-2 邏輯 ‘1’ (bit 1)

1.1.2 DMX512 數據幀

一個位元組為一個數據幀，稱作一個控制通道，可以控制燈光設備的一個或多個功能。一個數據幀由 1 個開始位元（S），8 個數據位元（D0~D7）和 2 個結束位元（E）共 11 位元構成，採用單向異步串列傳輸，如圖 1-3 所示：

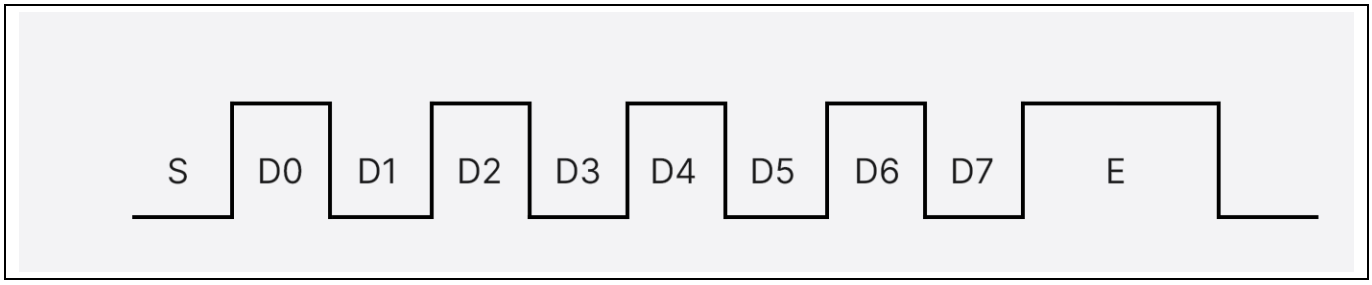

圖 1-3 DMX512 數據幀

D0~D7 為 8 位元控制數據，控制燈光亮度時可產生 256 個亮度等級，0 對應燈光最暗，255 對應最亮。位元寬 4 us，每幀 11 位元，指令幀寬度為 44 us，傳輸速率為 1/44=250 kbps。

1.1.3 DMX512 數據封包

一個完整的 DMX512 數據封包由一個 MTBP 位元，一個 Break 位元，一個 MAB 位元，一個 SC 幀和 512 個數據幀構成。如圖 1-4 所示：

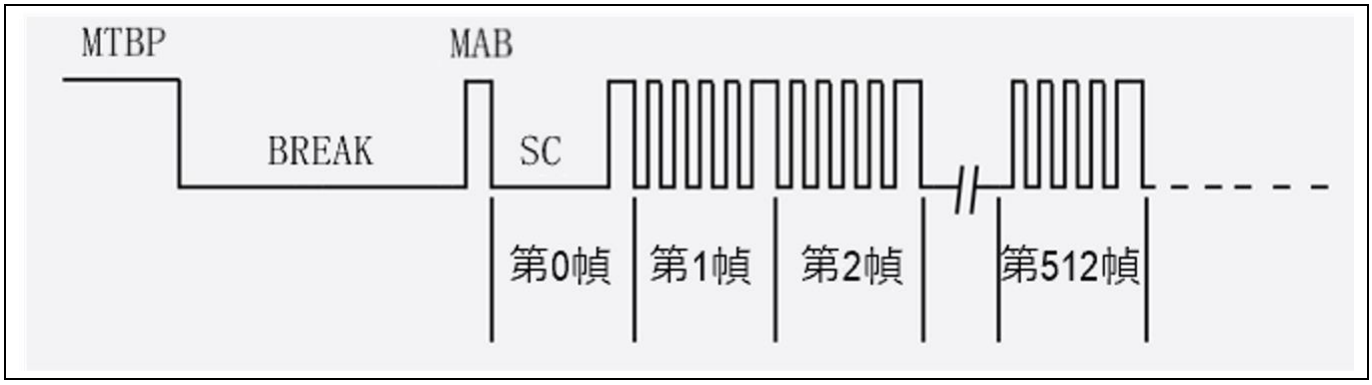

圖 1-4 DMX512 數據封包

- MTBP(Mark Time Between Packets)位元，代表著一個完整的數據封包發送完畢，高準位有效。
- Break 位元，對應一個數據封包結束後的程式復位階段，寬度不少於兩幀(22 位元)。

- MAB 位元，由於每一個數據幀的第一位元(即開始位元)為低準位，所以必須用一個高準位脈衝間隔前後兩個低準位脈衝，這個起間隔、分離作用的高準位脈衝即 MAB(Mark After Break)。
- SC(Start Code)幀，同數據幀一樣，也是由 11 位元構成，除最後的兩個高準位的結束位元之外，其他 9 位元全部是低準位，通常將其叫做開始幀、第 0 幀或第 0 通道，不控制設備。

1.1.4 UART 解碼 DMX512 數據

1. 設置 UART 串列傳輸速率為 250 kbps，8 位元數據長度，2 個 STOP 位元。
2. 使能 RDA/RLS 中斷。
3. 接收到 BREAK 位元和 SC 幀後開始接收控制通道數據幀（1~512）。
4. 控制通道數據幀數據用於控制 LED 亮度。

2. 代碼介紹

PSIO_StripInit()分別初始化 2 條燈帶對應的 PSIO Slot Control 和 PSIO PIN。

```
void PSIO_StripInit(void)
{
    uint8_t i;

    gsPSIO_ARGB2_Cfg[0].u8SlotCtrl = PSIO_SC0;
    gsPSIO_ARGB2_Cfg[0].u8DataPin = PSIO_PIN0;

    gsPSIO_ARGB2_Cfg[1].u8SlotCtrl = PSIO_SC1;
    gsPSIO_ARGB2_Cfg[1].u8DataPin = PSIO_PIN1;

    for (i = 0; i < 2; i++)
    {
        PSIO_Strip_OpenOutput(gsPSIO_ARGB2_Cfg[i], PSIO_PIN_ENABLE);
    }
}
```

PSIO_Strip_OpenOutput()在初始化 Output Mode 時，使用 PSIO_SET_GENCTL()設定 Pin 的狀態，內容為使用的 Pin、Slot controller、模式、初始化狀態及結束狀態等。PSIO_SET_WIDTH()與 PSIO_SET_OUTPUT_DEPTH()則是與數據的格式有關，此範例初始設定輸出寬度為 8 bit、輸入寬度為 1 bit，深度皆為 1。

```
static void PSIO_Strip_OpenOutput(S_PSIO_ARGB2_STRIP_CFG_T gsPSIO_Cfg, uint32_t u32Switch)
{
    /* PSIO UART TX PIN general setting */
    PSIO_SET_GENCTL(PSIO,
                    gsPSIO_Cfg.u8DataPin,
                    u32Switch, //PSIO_PIN_ENABLE
                    gsPSIO_Cfg.u8SlotCtrl,
                    PSIO_OUTPUT_MODE,
                    PSIO_LOW_LEVEL,
                    PSIO_LOW_LEVEL);

    /* Set data order ad MSB */
    PSIO_SET_ORDER(PSIO, gsPSIO_Cfg.u8DataPin, PSIO_MSB);

    /* Set output data buffer width as 8 bit, input data width as 1 bit*/
    PSIO_SET_WIDTH(PSIO, gsPSIO_Cfg.u8DataPin, 1, 8);

    /* Set output data buffer depth as 1 */
    PSIO_SET_OUTPUT_DEPTH(PSIO, gsPSIO_Cfg.u8DataPin, PSIO_DEPTH1);

    /* Set slot controller trigger source as software trigger */
    PSIO_SET_TRIGSRC(PSIO, gsPSIO_Cfg.u8SlotCtrl, PSIO_SW_TRIGGER);
}
```

PSIO_SendDataWithSize()在輸出邏輯 ‘0’ 、邏輯 ‘1’ 波形信號需要三種不同的動作，分別是輸出高準位、從緩衝區輸出數據及輸出低準位，時序的規劃如圖 1-1 和圖 1-2 邏輯 ‘1’ (bit 1)所述。Slot 0 ~ Slot 2 可視為 ARGB2 LED 協議的 1 bit。PSIO_SET_WIDTH()根據輸出的數據大小設定輸出寬度，例如設定 ARGB2 LED 的顏色需要設定輸出寬度為 24 bit。PSIO_SET_SCCTL()利用 PSIO 的循環功能，將此行為再複製 23 次，即輸出 24 bit 數據。設定完成後，用軟體觸發 PSIO_START_SC()的方式，讓 slot controller 開始執行。

```
void PSIO_SendDataWithSize(S_PSIO_ARGB2_STRIP_CFG_T gsPSIO_Cfg, uint32_t u32DataSize,
uint32_t u32Data)
{
    S_PSIO_CP_CONFIG sDataConfig = {0};

    /* Slot 0 ~ 7 */
    sDataConfig.CKPT0SLT = PSIO_SLOT0;
    sDataConfig.CKPT1SLT = PSIO_SLOT1;
    sDataConfig.CKPT2SLT = PSIO_SLOT2;
    sDataConfig.CKPT3SLT = PSIO_SLOT_DISABLE;
    sDataConfig.CKPT4SLT = PSIO_SLOT_DISABLE;
    sDataConfig.CKPT5SLT = PSIO_SLOT_DISABLE;
    sDataConfig.CKPT6SLT = PSIO_SLOT_DISABLE;
    sDataConfig.CKPT7SLT = PSIO_SLOT_DISABLE;

    /* Actions 0 ~ 7 */
    sDataConfig.CKPT0ACT = PSIO_OUT_HIGH;
    sDataConfig.CKPT1ACT = PSIO_OUT_BUFFER;
    sDataConfig.CKPT2ACT = PSIO_OUT_LOW;
    sDataConfig.CKPT3ACT = PSIO_NO_ACTION;
    sDataConfig.CKPT4ACT = PSIO_NO_ACTION;
    sDataConfig.CKPT5ACT = PSIO_NO_ACTION;
    sDataConfig.CKPT6ACT = PSIO_NO_ACTION;
    sDataConfig.CKPT7ACT = PSIO_NO_ACTION;

    /* Set check point configuration */
    PSIO_SET_CP_CONFIG(PSIO, gsPSIO_Cfg.u8DataPin, &sDataConfig);

    /* Set slot count */
    PSIO_SCSLOT_SET_SLOT(PSIO, gsPSIO_Cfg.u8SlotCtrl, PSIO_SLOT0, 1);
    PSIO_SCSLOT_SET_SLOT(PSIO, gsPSIO_Cfg.u8SlotCtrl, PSIO_SLOT1, 2);
    PSIO_SCSLOT_SET_SLOT(PSIO, gsPSIO_Cfg.u8SlotCtrl, PSIO_SLOT2, 1);

    /* Enable repeat slot0 ~ slot2 u32DataSize - 1 times */
    PSIO_SET_SCCTL(PSIO, gsPSIO_Cfg.u8SlotCtrl, PSIO_SLOT0, PSIO_SLOT2,
                   u32DataSize - 1, PSIO_REPEAT_DISABLE);

    /* Set out data width as u32DataSize bit */
    PSIO_SET_WIDTH(PSIO, gsPSIO_Cfg.u8DataPin, 0, u32DataSize);

    /* Set out data depth as 1 */
    PSIO_SET_OUTPUT_DEPTH(PSIO, gsPSIO_Cfg.u8DataPin, PSIO_DEPTH1);
```

```
    /* Set output data to buffer */
    PSIO_SET_OUTPUT_DATA(PSIO, gsPSIO_Cfg.u8DataPin, u32Data);

    /* Trigger slot controller */
    PSIO_START_SC(PSIO, gsPSIO_Cfg.u8SlotCtrl);

    /* Wait for slot controller is not busy */
    while (PSIO_GET_BUSY_FLAG(PSIO, gsPSIO_Cfg.u8SlotCtrl));

    /* Stop slot controller */
    PSIO_STOP_SC(PSIO, gsPSIO_Cfg.u8SlotCtrl);

    /* Disable repeat slot0 ~ slot2 */
    PSIO_SET_SCCTL(PSIO, gsPSIO_Cfg.u8SlotCtrl, PSIO_SLOT_DISABLE, PSIO_SLOT_DISABLE,
                   0, PSIO_REPEAT_DISABLE);
}
```

DMX512_UART_Init()對 UART 初始化。

```
void DMX512_UART_Init(void)
{
    /* Configure DMX512_UART_nRTS as OUTPUT mode */
    GPIO_SetMode(DMX512_UART_nRTS_PORT, DMX512_UART_nRTS_BIT, GPIO_MODE_OUTPUT);
    DMX512_UART_nRTS_PIN = 0;

    /* Set multi-function pins for DMX512_UART RX and TX */
    DMX512_UART_RX_MFP &= ~(DMX512_UART_RX_MFP_MSK | DMX512_UART_TX_MFP_MSK);
    DMX512_UART_TX_MFP |= (DMX512_UART_RX_MFP_FUNC | DMX512_UART_TX_MFP_FUNC);

    /* Configure DMX512_UART and set DMX512_UART baud rate */
    UART_Open(DMX512_UART, DMX512_UART_BAUDRATE);
    DMX512_UART->LINE = UART_WORD_LEN_8 | UART_PARITY_NONE | UART_STOP_BIT_2;

    /* Enable RDA\RLS Interrupt */
    NVIC_EnableIRQ(DMX512_UART_IRQn);
    UART_EnableInt(DMX512_UART, (UART_INTEN_RDAIEN_Msk | UART_INTEN_RLSIEN_Msk));
}
```

DMX512_UART_IRQHandler()為 UART 中斷處理程式。

```
void DMX512_UART_IRQHandler(void)
{
    uint32_t u32IntSts = DMX512_UART->INTSTS;
    uint8_t u8InChar;

    if (u32IntSts & UART_INTSTS_RLSIF_Msk)
    {
        /* Break Interrupt Flag */
        /* This bit is set to logic "1" whenever the received data input (RX) is held in
           the "spacing state" (logic 0) for longer than a full word transmission time
```

```
           (that is, the total time of "START bit" + data bits + parity + STOP bits). */
        if (DMX512_UART->FIFOSTS & UART_FIFOSTS_BIF_Msk)
        {
            u8BreakFlag = 1;
            u16RecvLen = 0;
            u8ScFlag = 0;
        }

        /* Framing Error Flag */
        /* This bit is set to logic "1" whenever the received character does not have a
           valid "STOP bit" (that is, the STOP bit following the last data bit or PARITY
           bit is detected as logic 0). */
        else if (DMX512_UART->FIFOSTS & UART_FIFOSTS_FEF_Msk)
        {
            u8BreakFlag = 0;
            u16RecvLen = 0;
            u8ScFlag = 0;
        }

        u8InChar = UART_READ(DMX512_UART);
    }

    else if (u32IntSts & UART_INTSTS_RDAINT_Msk)
    {
        /* Get all the input characters */
        while (UART_GET_RX_EMPTY(DMX512_UART) == 0)
        {
            /* Get the character from UART Buffer */
            u8InChar = UART_READ(DMX512_UART);

            if (u8BreakFlag)
            {
                if (u8ScFlag == 0)
                {
                    if (u8InChar == 0x00)
                    {
                        u8ScFlag = 1;
                        g_u8Dmx512Buff[u16RecvLen++] = u8InChar;
                    }
                    else
                        u8BreakFlag = 0;
                }
                else
                {
                    g_u8Dmx512Buff[u16RecvLen++] = u8InChar;
                }
            }

            if (u16RecvLen>512)
                u16RecvLen = 0;
        }

        /*Forces a write of all user-space buffered data for the given output*/
        fflush(stdout);
    }
```

```
    if (DMX512_UART->FIFOSTS & (UART_FIFOSTS_BIF_Msk | UART_FIFOSTS_FEF_Msk |
                                UART_FIFOSTS_PEF_Msk | UART_FIFOSTS_RXOVIF_Msk))
    {
        DMX512_UART->FIFOSTS = (UART_FIFOSTS_BIF_Msk | UART_FIFOSTS_FEF_Msk |
                                UART_FIFOSTS_PEF_Msk | UART_FIFOSTS_RXOVIF_Msk);
    }
}
```

Strip_PSIO_DMX512_Test()對 DMX512 數據幀的數據進行處理，透過 PSIO 模塊控制 ARGB2 LED。g_u8Dmx512Buff[1]數據幀 1 到 g_u8Dmx512Buff[512]數據幀 512 分別對應第 1-512 顆燈的亮度，g_u8Addr 為 DMX512 控制板的設備位址。

```
void Strip_PSIO_DMX512_Test(void)
{
    uint8_t i, j;
    uint8_t u8addr;

    u8addr = g_u8Addr;

    for (i = 0; i < sizeof(g_u8TransferTbl); i++)
    {
        g_u8TransferTbl[i] = 0;
    }

    for (i = 0; i < sizeof(LED_POS_TBL); i++)
    {
        g_u8TransferTbl[(LED_POS_TBL[i]-1)*3] = g_u8Dmx512Buff[i*3+1+u8addr];
        g_u8TransferTbl[(LED_POS_TBL[i]-1)*3+1] = g_u8Dmx512Buff[i*3+2+u8addr];
        g_u8TransferTbl[(LED_POS_TBL[i]-1)*3+2] = g_u8Dmx512Buff[i*3+3+u8addr];
    }

    for (i = 0; i < STRIP_NUM; i++)
    {
        uint8_t channel = (i == 0) ? 1 : 0;

        for (j = 0; j < LED_NUM; j++)
        {
            PSIO_StripSetRGBColor(channel, g_u8TransferTbl[i * 36+j*3],
                 g_u8TransferTbl[i * 36+j*3+1], g_u8TransferTbl[i * 36+j*3+2]);
        }
    }
}
```

3. 軟體與硬體需求

3.1 軟體需求

- BSP 版本
  - M251_M252_M254_M256_M258_Series_BSP_CMSIS_V3.02.007
- IDE 版本
  - Keil uVersion V5.41.0.0

3.2 硬體需求

- 電路元件
  - NuMaker-M251KG V1.1
  - NuMaker-Lighting-ARGB
- USB 轉 DMX512 Bridge

  用戶透過指撥開關設置設備位址，如圖 3-1 所述，設備位址（0~255）分別對應 DMX512 控制通道數據幀（1~256）。

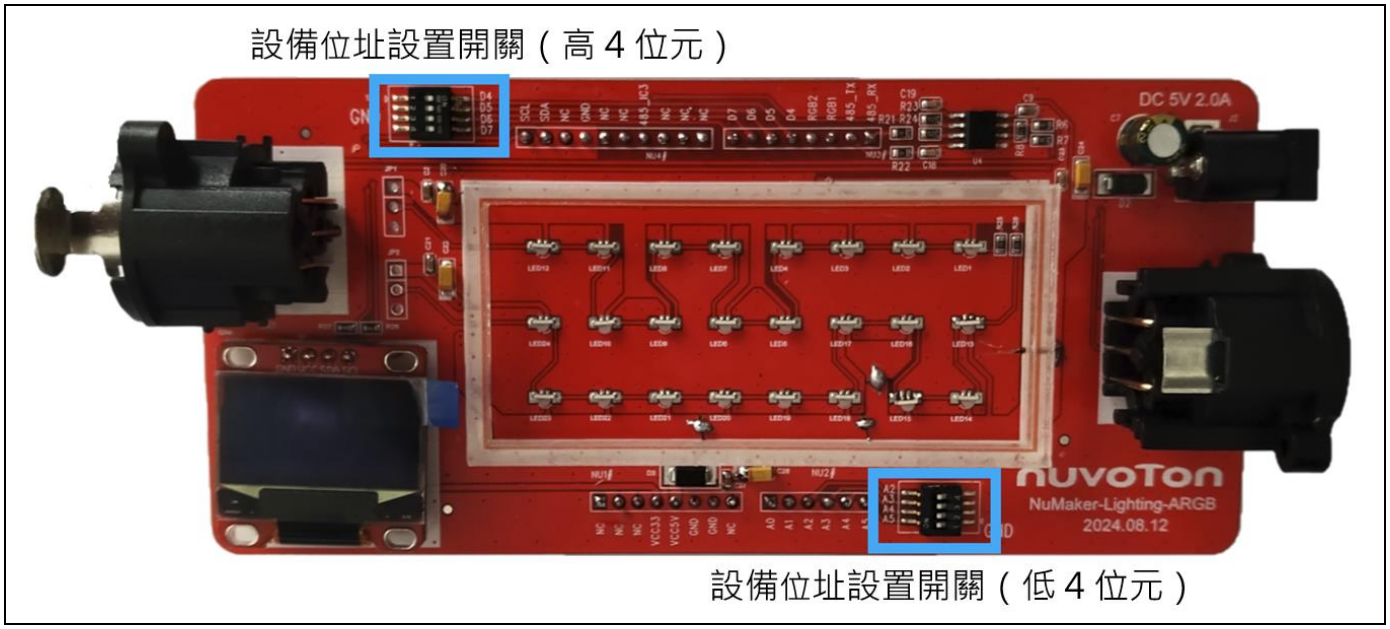

圖 3-1 備位址設置開關

4. 目錄資訊

目錄結構如下圖所示。

EC_M251_DMX512_Control_V1.00	
Tool	Tools
DMX512_Controller	Source file of DMX512_Controller
Library	Sample code header and source file
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. 將 USB 轉 DMX512 Bridge 插入電腦 USB 口，XLR 插入 NuMaker-Lighting-ARGB 板，如圖 5-1：

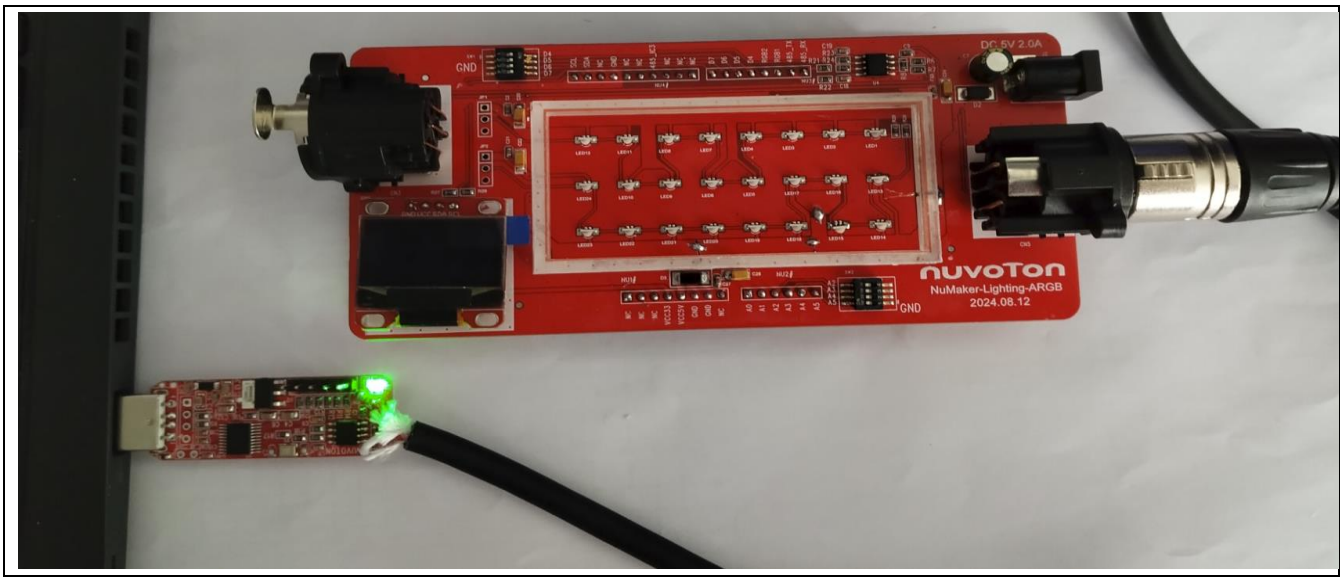

圖 5-1 USB 轉 DMX512 Bridge 接線圖

2. 根據目錄資訊章節進入 *Tool\DMX512_Controller\bin\Debug* 路徑中打開 DMX512 Controller，點擊 Search->下拉選擇框中選擇 PID：5020->點擊 Connect。根據 NuMaker-Lighting-ARGB 設置的設備位址，點擊 Selection 右邊下拉，選擇對應的 Board，如圖 5-2：

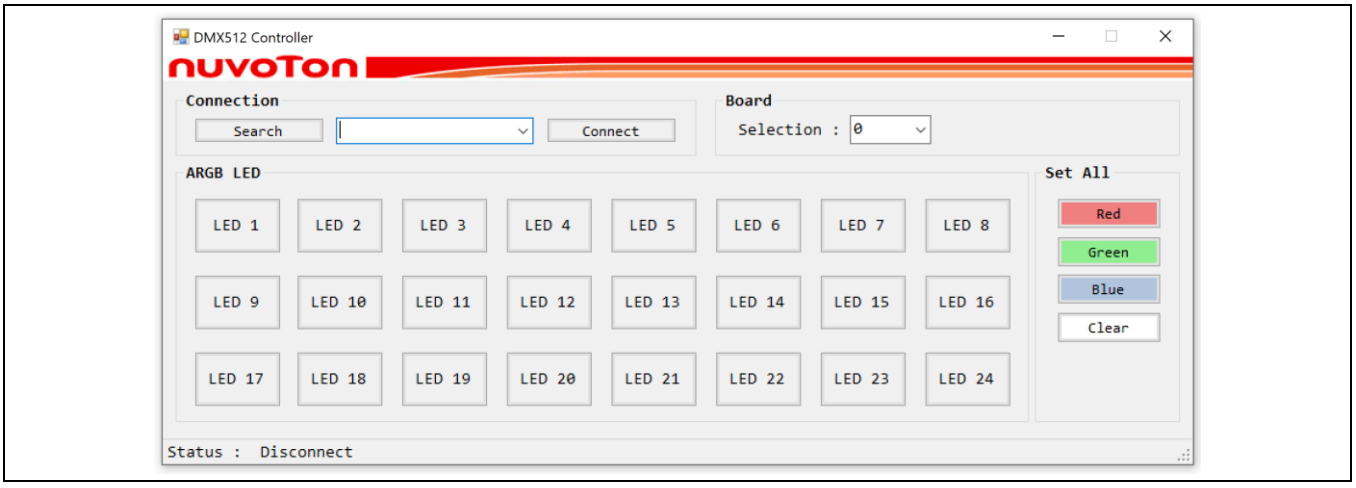

圖 5-2 DMX512 Controller 界面圖

3. 根據目錄資訊章節進入 ExampleCode 路徑中的 KEIL 文件夾，雙擊 *M251_DMX512_Control.uvprojx*。

4. 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

5. 進入模擬模式介面

   - 執行代碼
   - 在 DMX512 Controller 軟體介面中，透過滑鼠左鍵選擇對應的 LED，並調整其亮度。

6. 修訂記錄

Date	Revision	Description
2024.10.25	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.