# 以 emWin 設計示波器 GUI 專案

NuMicro® 32 位系列微控制器範例程式介紹

## 文件資訊

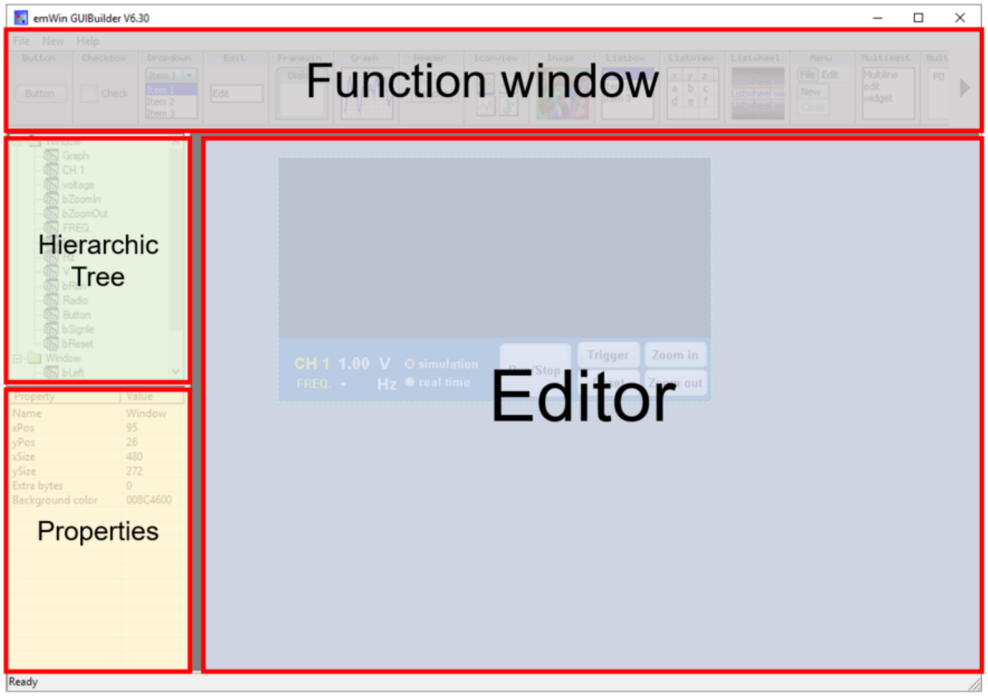
| | |
|---|---|
| 應用簡述 | 本範例程式使用 SEGGER emWin GUIBuilder 建立圖形化使用者介面(GUI)，結合 M467 系列開發平台，展示示波器圖形化介面。 |
| BSP 版本 | M460_Series_BSP_CMSIS_V3.00.001 |
| 開發平台 | NuMaker-HMI-M467 |

# 1. 概述

本範例程式將展示使用 SEGGER emWin 於 NuMaker-HMI-M467 開發平台，逐步完成示波器專案的 GUI(Graphical User Interface)介面設計，互動上設計 Run/Stop 按鈕及 Trigger、Reset、Zoom in、Zoom out 等相應按鈕點擊功能。使用中斷 EADC 擷取類比訊號並於畫面上顯示，EADC 的取樣頻率為 1kHz，最高支持 3.3v 訊號源，同步啟動 DAC 輸出 30Hz 的弦波驗證。

## 1.1 UI 設計

本範例使用 emWin GUIBuilder V6.30 做畫面設計，GUIBuilder IDE 開發上可區分為 Function window、Editor window、Hierarchic Tree、Properties，如圖 1-1 所示。使用者只需透過 Function window 上的元件拖曳至 Editor window 就能設計 UI 畫面。



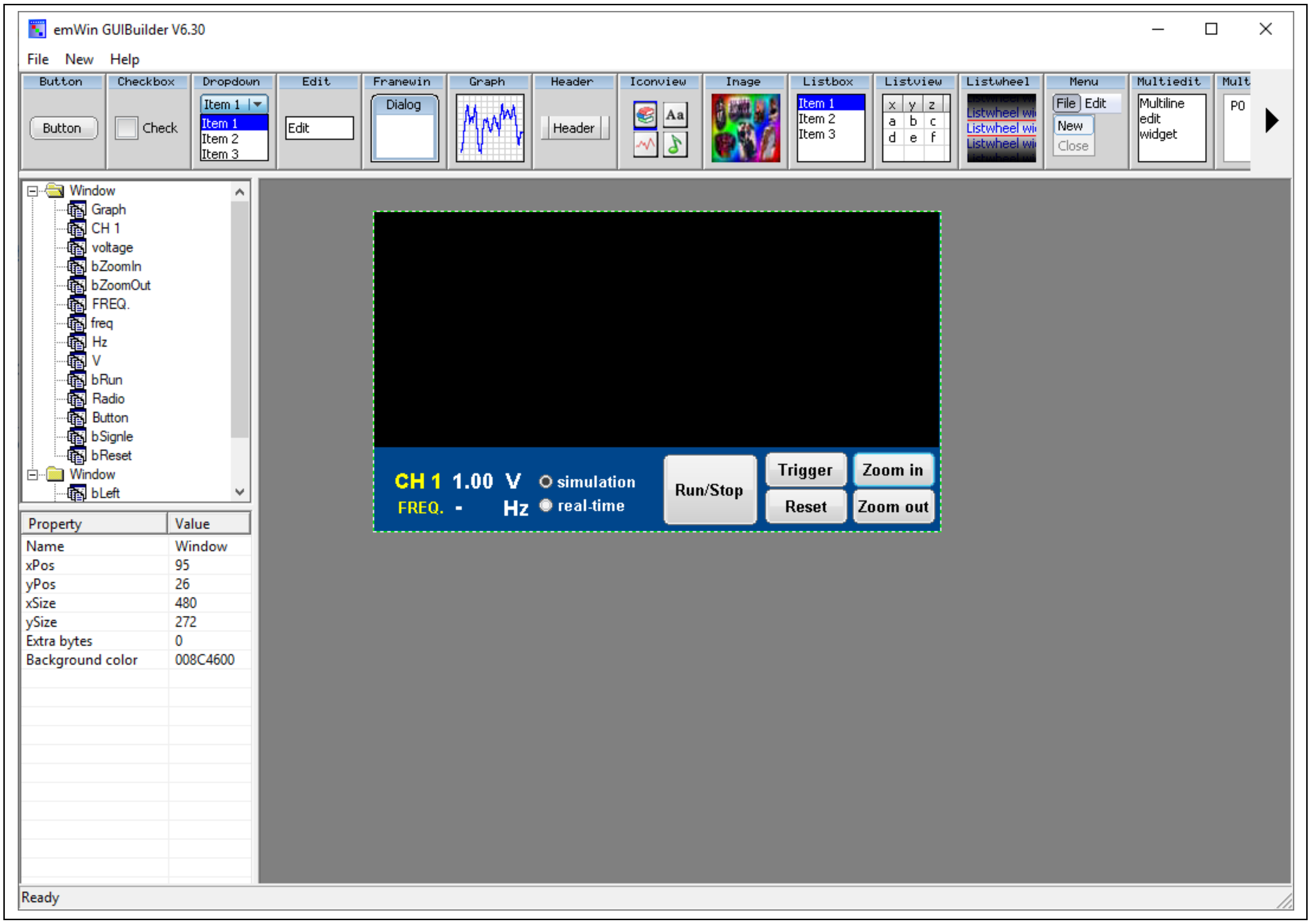
| Property | Value |
|---|---|
| Name | Window |
| xPos | 95 |
| yPos | 26 |
| xSize | 480 |
| ySize | 272 |
| Extra bytes | 0 |
| Background color | 008C4600 |

圖 1-1 emWin GUIBuilder IDE

NuMaker-HMI-M467 開發板採用 NuMaker-THT-LCD43，window 大小為 480x272 pixels。設置示波器專案會需要 Graph 元件繪製波形，新增 Text 欄位顯示波形頻率資訊，Radio 單選當前畫面顯示的模式，分為 simulation 及 real-time 模式，其中 simulation 模式會模擬一個弦波輸出，

real-time 模式會真實取樣 EADC 資訊，按鈕則繪製 Run/Stop 按鈕及 Trigger、Reset、Zoom in、Zoom out 等相應按鈕點擊功能。Run/Stop 按鈕可以設置示波器畫面開啟捲動或停止捲動功能；Trigger 按鈕可以擷取第一段訊號源，與畫面的大小配置相同；Reset 按鈕會清除當前頁面的數據及重置功能；Zoom in/out 會放大或縮小波形的振幅，設計至多 3 個 level。

元件的顏色大小配置都可以透過 **Properties** 中配置，GUI 畫面設計及屬性可參考圖 1-3 至圖 1-5。



| Property | Value |
|---|---|
| Name | Window |
| xPos | 95 |
| yPos | 26 |
| xSize | 480 |
| ySize | 272 |
| Extra bytes | 0 |
| Background color | 008C4600 |

圖 1-2 示波器專案格局配置圖

存檔後會更新 *FramewinDLG.c* 檔案，需要將變更後的檔案覆蓋到示波器專案，此小節即完成初步的 UI 設計。於此範例程式中，*FramewinDLG.c* 存放於 Application 資料夾中，已設計好 UI 的框架及按鈕的配置等設定，用戶可以自行設計自己的 GUI。

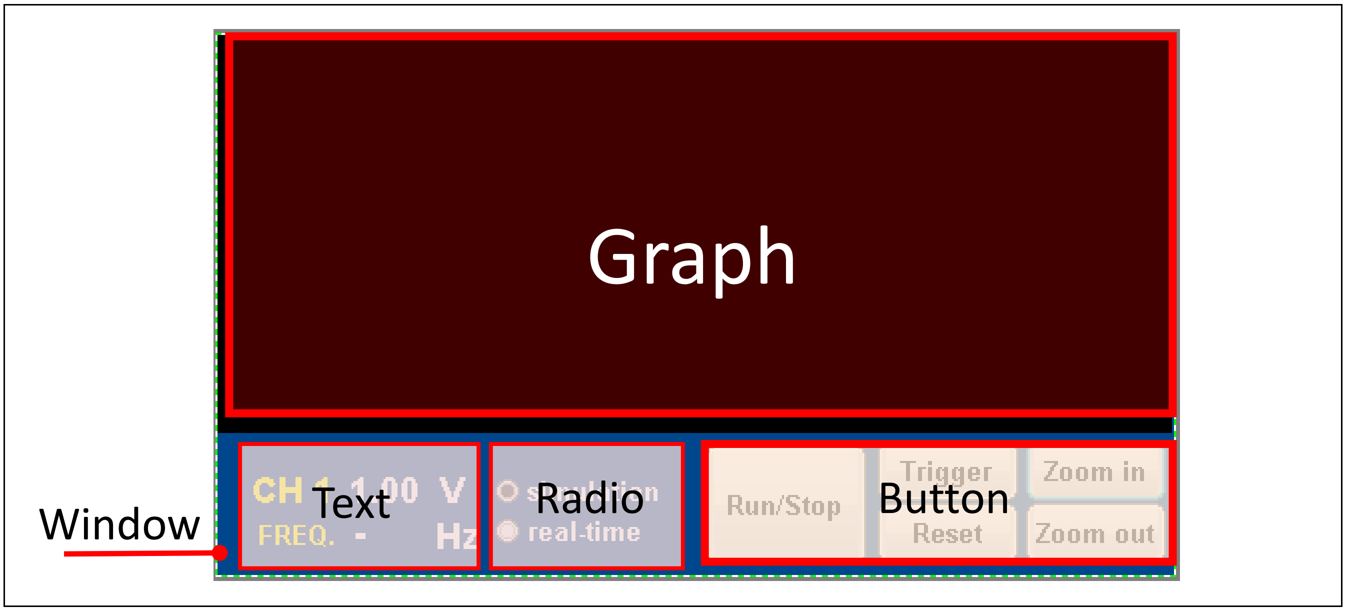


圖 1-3 示波器元件規劃配置圖

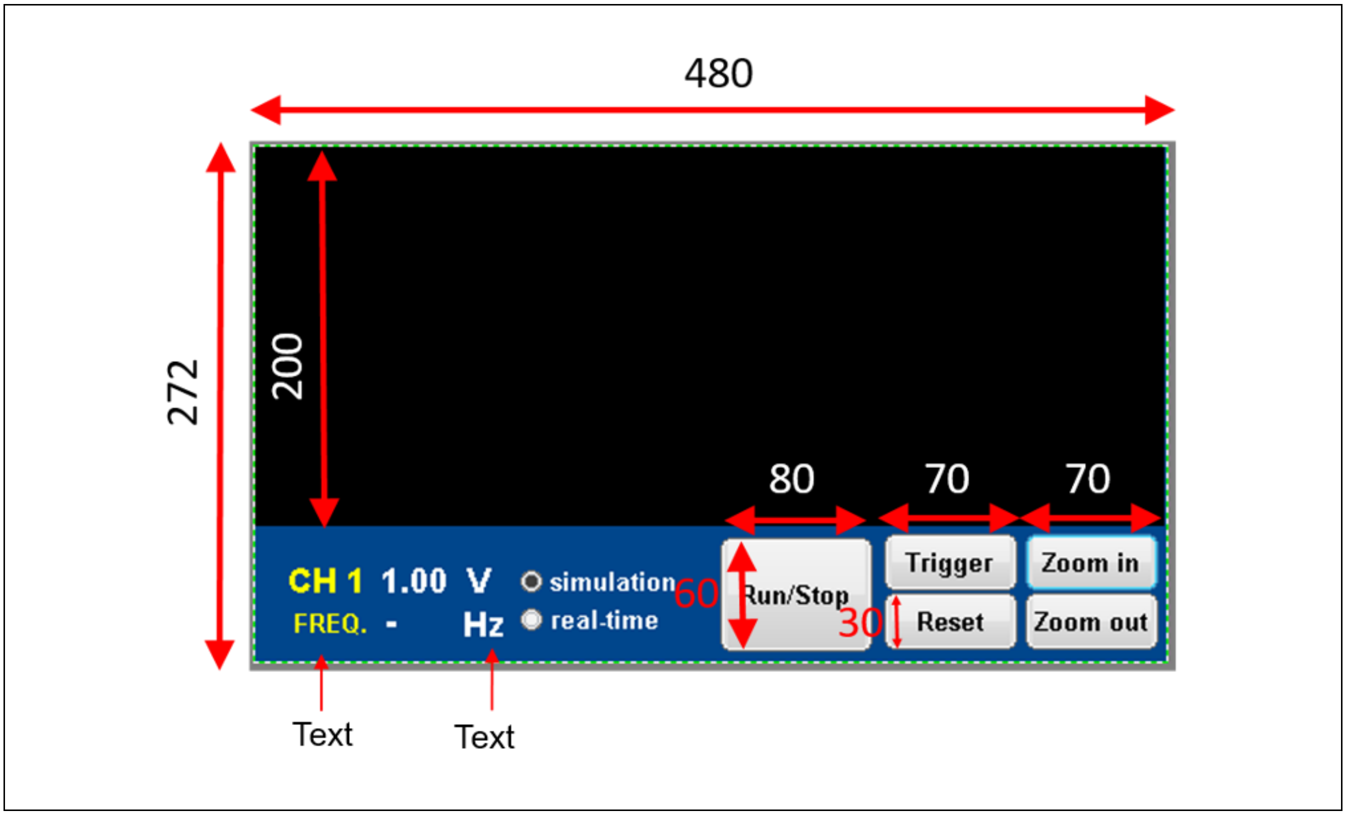


圖 1-4 示波器元件大小圖

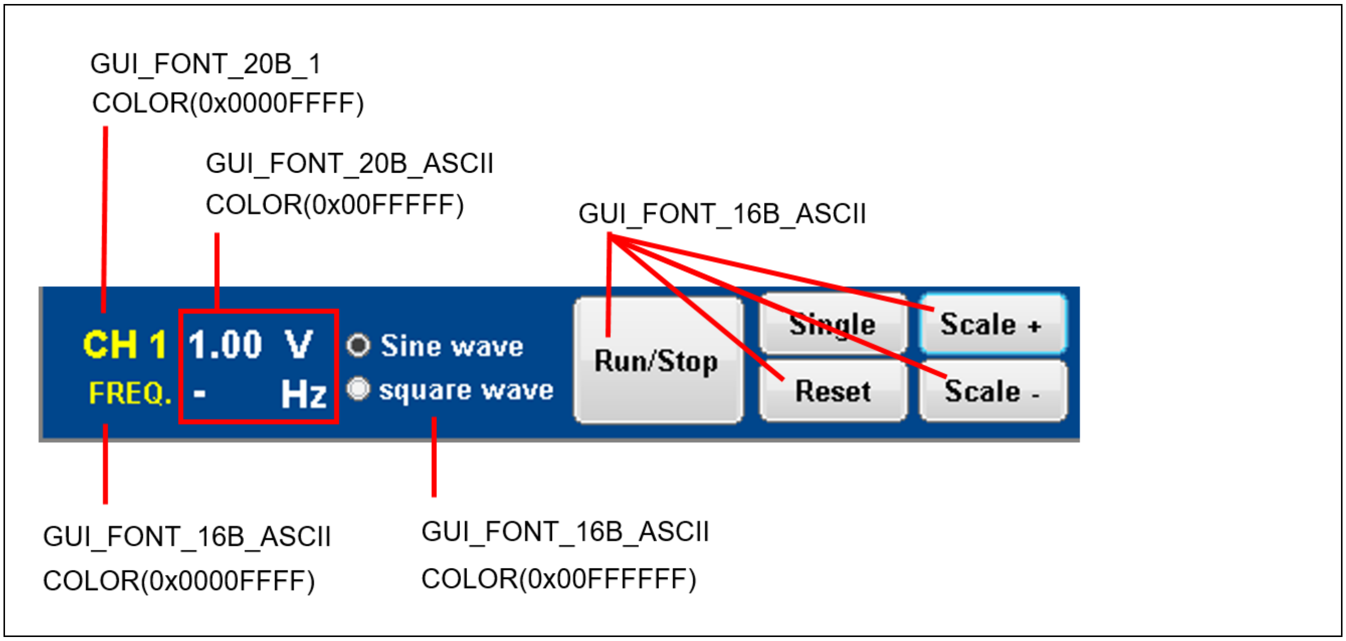


圖 1-5 示波器元件字型色彩圖

## 1.2 UI 程式原理

系統的框架定義，如圖 1-6 所示。

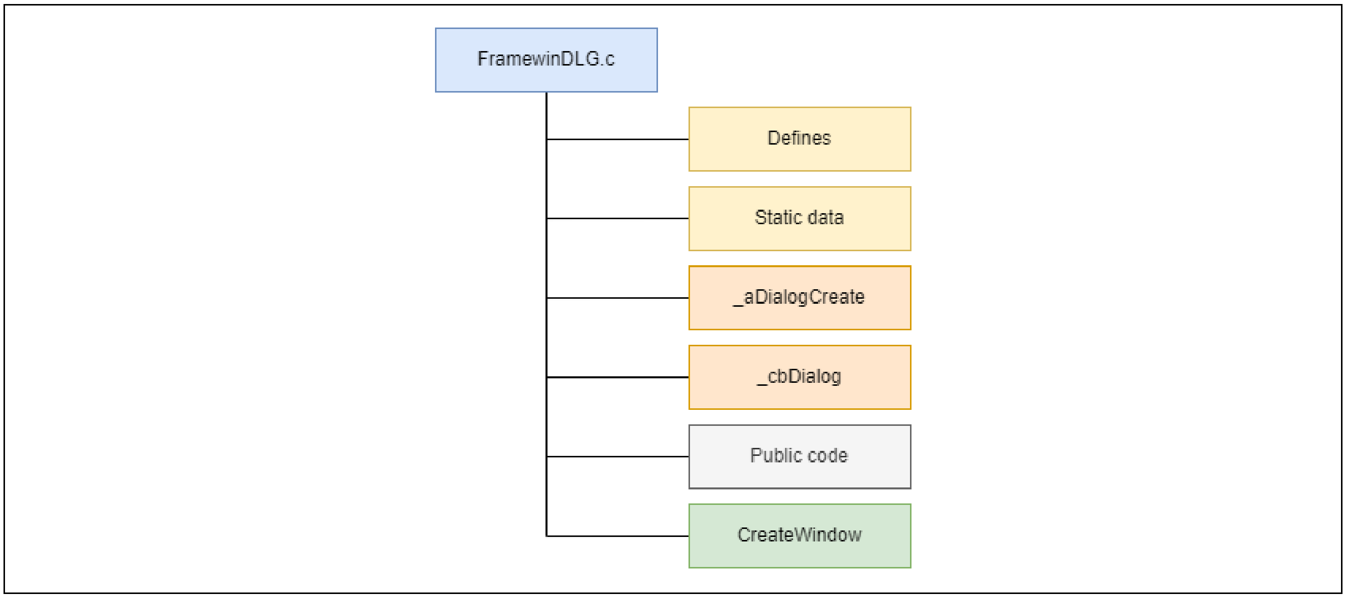


圖 1-6 UX 框架

*FramewinDLG.c* 中以 API 方式來設定元件的外觀，可以在 **_aDialogCreate** 中發現到 GUIBuilder 於 Editor 中的位置等配置:

```
/*********************************************************************
*
*    _aDialogCreate
*/

static const GUI_WIDGET_CREATE_INFO _aDialogCreate[] =
{
    { WINDOW_CreateIndirect, "Window",  ID_WINDOW_0,  0,   0, 480, 272, 0, 0x0, 0 },
    { GRAPH_CreateIndirect,  "Graph",   ID_GRAPH_0,   0,   0, 480, 200, 0, 0x0, 0 },
    … …
    { BUTTON_CreateIndirect, "Button",  ID_BUTTON_3, 318, 314, 80, 30, 0, 0x0, 0 },
    { BUTTON_CreateIndirect, "bSingle", ID_BUTTON_4, 331, 204, 70, 30, 0, 0x0, 0 },
    { BUTTON_CreateIndirect, "bReset",  ID_BUTTON_5, 331, 235, 70, 30, 0, 0x0, 0 },
};
```

*FramewinDLG.c* 中 **_cbDialog** 處理 GUI 生成 D 及對應事件，而其中 GRAPH widget 設定網格格線的配置:

```
/*********************************************************************
*
*    _cbDialog
*/

static void _cbDialog(WM_MESSAGE * pMsg)
{
… …
    switch (pMsg->MsgId)
    {

        case WM_INIT_DIALOG:
            // Initialization of 'Window'
            hItem = pMsg->hWin;
            WINDOW_SetBkColor(hItem, GUI_MAKE_COLOR(0x008C4600));

            // Enable multi-buffering
            WM_MULTIBUF_Enable(1);

            … …

            // Initialize the GRAPH widget
            hItem = WM_GetDialogItem(pMsg->hWin, ID_GRAPH_0);

            // Set the appearance and grid properties of the graph
            GRAPH_SetBorder(hItem, 1,1,1,1);
            GRAPH_SetColor(hItem, GUI_BLACK, GRAPH_CI_BK);
            GRAPH_SetColor(hItem, GUI_GRAY_E7, GRAPH_CI_BORDER);
            GRAPH_SetColor(hItem, GUI_GRAY_E7, GRAPH_CI_FRAME);
            GRAPH_SetColor(hItem, GUI_WHITE, GRAPH_CI_GRID);
            GRAPH_SetGridVis(hItem, 1);
            GRAPH_SetGridDistX(hItem, 50);
            GRAPH_SetGridDistY(hItem, 25);
            GRAPH_SetLineStyleH(hItem, GUI_LS_DOT);
            GRAPH_SetLineStyleV(hItem, GUI_LS_DOT);

            // Create and configure a vertical scale, then attach it to the graph
            hScaleHandle = GRAPH_SCALE_Create(15, GUI_TA_HCENTER | GUI_TA_VCENTER,
GRAPH_SCALE_CF_VERTICAL, 1);
            GRAPH_SCALE_SetFont(hScaleHandle, &GUI_Font8x8);
            GRAPH_SCALE_SetTextColor(hScaleHandle, GUI_BLACK);
            GRAPH_SCALE_SetNumDecs(hScaleHandle, 0);
            GRAPH_SCALE_SetTickDist(hScaleHandle, 20);
            GRAPH_SCALE_SetOff(hScaleHandle, 100);
            GRAPH_AttachScale(hItem,hScaleHandle);

            // Introduce a delay of 500 milliseconds
            GUI_Delay(500);

            // Create and configure graph data, then attach it to the graph
            hGraphData = GRAPH_DATA_YT_Create(GUI_YELLOW,LCD_GetXSize(),DrawData,0);
            GRAPH_DATA_YT_SetAlign(hGraphData, GRAPH_ALIGN_RIGHT);
            GRAPH_DATA_YT_SetOffY(hGraphData, 100);
            GRAPH_AttachData(hItem, hGraphData);


            // Set a custom callback, here we want to draw a cursor
            // WM_SetCallback(hItem, _cbGraph);
            break;


        //
        // event
        //
        case WM_NOTIFY_PARENT:
            // Retrieve the ID and notification code of the event sender
            Id    = WM_GetId(pMsg->hWinSrc);
            NCode = pMsg->Data.v;

            switch(Id)
            {
                case ID_BUTTON_0: // Notifications sent by 'bZoomIn'
                    switch(NCode)
                    {
                        case WM_NOTIFICATION_CLICKED:
                            // Clear existing graph data
                            GRAPH_DATA_YT_Clear(hGraphData);

                            // Increment the 'multiple' factor for zooming in
                            multiple++;
                            if(multiple > 2) multiple=2;

                            // Update graph data with adjusted values
                            for(i=0; i<480; i++)
                            {
                                GRAPH_DATA_YT_AddValue(hGraphData,  DrawData[i] *
multipleArray[multiple]);
                            }
                            break;
                        case WM_NOTIFICATION_RELEASED:
                            break;
                    }
                    break;

                case ID_BUTTON_1: // Notifications sent by 'bZoomOut'
                    switch(NCode)
                    {
                        case WM_NOTIFICATION_CLICKED:
                            // Clear existing graph data
                            GRAPH_DATA_YT_Clear(hGraphData);

                            if(multiple == 0)
                                multiple=0;
                            else
                                multiple--;

                            for(i=0; i<480; i++)
                            {
                                GRAPH_DATA_YT_AddValue(hGraphData,  DrawData[i] *
multipleArray[multiple]);
                            }
                            break;

                        case WM_NOTIFICATION_RELEASED:
                            // USER START (Optionally insert code for reacting on
notification message)
                            break;
                    }
                    break;
            … …

            }
            break;

… …
}
```

**CreateFramewin()**來執行 GUI 的創建，並設置 multi-buffering。

```
/*********************************************************************
*
*    CreateWindow
*/
WM_HWIN CreateFramewin(void)
{

    WM_HWIN hWin;

    // Enable multi-buffering for the window
    WM_MULTIBUF_Enable(1);
    // Create a dialog box (FrameWin) using the specified dialog creation parameters
    hWin = GUI_CreateDialogBox(_aDialogCreate, GUI_COUNTOF(_aDialogCreate), _cbDialog,
WM_HBKWIN, 0, 0);

    return hWin;
}
```

## 2. 程式介紹

*main.c* 中先對系統初始化，設定觸控屏幕後執行 MainTask()以啟動 emWin 顯示 GUI。

於 **MainTask()** 中，**Oscilloscope_init()**會啟動示波器取得 EADC 的資料，取樣率約 1kHz，同步輸出一個弦波，並更新 GUI 上的數據資料。

```
/*--------------------------------------------------------------------------------------*/
/*   Main Function                                                                      */
/*--------------------------------------------------------------------------------------*/

int main(void)
{
    /* init */
    SYS_Init();

    // Initialize the touch functionality.
    g_enable_Touch = 0;
    tmr_init();
    st1633i_low_level_init();
    st1633i_startup_sequence();
    g_enable_Touch = 1;

    // Print a message indicating the CPU frequency.
    printf("[M]Oscilloscope Demo - CPU @ %d Hz \r\n", SystemCoreClock);

    /* Main Loop  */
    MainTask();


    // This loop will keep the program running indefinitely.
    while(1) {}

    return 0;
}
```

```
void MainTask(void)
{
    // Initialize variables
    int i=0;
    int32_t bSingleCount = 0;
    int32_t bSingleFlag = 0;

    // Create a window handle
    WM_HWIN hWin;
    printf("Main Task -> \n");

    // Enable memory device for GUI rendering
    WM_SetCreateFlags(WM_CF_MEMDEV);

    // Initialize the GUI
    GUI_Init();
    WM_MULTIBUF_Enable(1);   // Enable multi-buffering for better performance
    hWin = CreateFramewin(); // Create a window using CreateFramewin function

    // Initialize the Oscilloscope
    Oscilloscope_init();

    // Enter the main loop of the task
    while (1)
    {
        GUI_Delay(1);
         …
    }
}
```

Oscilloscope.c 中初始化就設置 EADC(量測引腳為 PB0)、Timer1、Timer2、EPWM(輸出腳位為 PC12)和 DAC(輸出腳位為 PB12)。屏幕更新率同為 100Hz 刷新一次。

```
/****************************************************************************************/
/* Oscilloscope init                                                                    */
/*   Use EADC  -> TMR1 1kHz                                                             */
/*       DAC    -> TMR2 2kHz                                                            */
/*       EPWM1 -> 30 Hz                                                                 */
/****************************************************************************************/
void Oscilloscope_init(void)
{
     SYS_UnlockReg();

    /* Enable module clock */
    CLK_EnableModuleClock(EADC0_MODULE);
    CLK_EnableModuleClock(TMR1_MODULE);
    CLK_EnableModuleClock(TMR2_MODULE);
    CLK_EnableModuleClock(EPWM1_MODULE);
    CLK_EnableModuleClock(DAC_MODULE);

    /* Set module clock */
    CLK_SetModuleClock(EADC0_MODULE, CLK_CLKSEL0_EADC0SEL_PLL_DIV2,
CLK_CLKDIV0_EADC0(12));
    CLK_SetModuleClock(EPWM1_MODULE, CLK_CLKSEL2_EPWM1SEL_PCLK1, 0);
    CLK_SetModuleClock(TMR1_MODULE, CLK_CLKSEL1_TMR1SEL_HIRC, NULL);
    CLK_SetModuleClock(TMR2_MODULE, CLK_CLKSEL1_TMR2SEL_HIRC, 0);

    /* EADC0 channels. */
    SET_EADC0_CH0_PB0();
    SET_DAC0_OUT_PB12();

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

    /* EPWM channels. */
    SET_EPWM1_CH0_PC12();
    EPWM_ConfigOutputChannel(EPWM1, 0, 30, 50);
    EPWM_EnableOutput(EPWM1, 0x3F);
    EPWM_Start(EPWM1, 0x3F);

    /* module init */
    TMR1_Init();
    EADC_Init();
    DAC_Init();
    SYS_LockReg();
}
```

若要變更 EADC 取樣時間可以變更 Timer1 中設置，目前設置的取樣頻率是 1kHz 並觸發 EADC 中斷測量。

```
/*****************************************************************************************/
/* TIMER1 init */
/*****************************************************************************************/

void TMR1_Init(void)
{
    TIMER_Open(TIMER1, TIMER_PERIODIC_MODE, 1000);
    TIMER_SetTriggerTarget(TIMER1, TIMER_TRG_TO_EADC);

     // Start Timer 1
     TIMER_Start(TIMER1);
}
```

```
/*****************************************************************************************/
/* EADC  init */
/*****************************************************************************************/
void EADC_Init(void)
{
    int32_t i32Err;

    i32Err = EADC_Open(EADC0, EADC_CTL_DIFFEN_SINGLE_END);

    … …

    /* Configure the sample module 0 for analog input channel 0 and enable Timer1 trigger 
source */
     EADC_ConfigSampleModule(EADC0, 0, EADC_TIMER1_TRIGGER, 0);

     /* Set sample module external sampling time to 0 */
     EADC_SetExtendSampleTime(EADC0, 0, 0);

     /* Enable Accumulate feature */
     EADC_ENABLE_ACU(EADC0, 0, EADC_MCTL1_ACU_32);

     /* Enable Average feature */
     EADC_ENABLE_AVG(EADC0, 0);

     /* Clear the A/D ADINT0 interrupt flag for safe */
     EADC_CLR_INT_FLAG(EADC0, EADC_STATUS2_ADIF0_Msk);

     /* Enable the sample module interrupt.  */
     EADC_ENABLE_INT(EADC0, BIT0);
     EADC_ENABLE_SAMPLE_MODULE_INT(EADC0, 0, BIT0);
     NVIC_EnableIRQ(EADC00_IRQn);

      /* Reset the ADC interrupt indicator and trigger sample module to start A/D 
conversion */
     EADC_START_CONV(EADC0, BIT0);

}
```

# 3. 軟體與硬體需求

## 3.1 軟體需求

- BSP 版本
  - M460_Series_BSP_CMSIS_V3.00.001
- IDE 版本
  - Keil uVersion 5.36

## 3.2 硬體需求

- 電路元件
  - NuMaker-HMI-M467HJ V1.1
- 線路示意圖
  - 將 NuMaker-HMI-M467HJ 透過 USB 線材與 PC 相接。
  - 輸入腳位: EADC (PB0)，請使用杜邦線接線連接至 PB12 或 PC12。
  - 輸出腳位: DAC (PB12)、EPWM (PC12)。

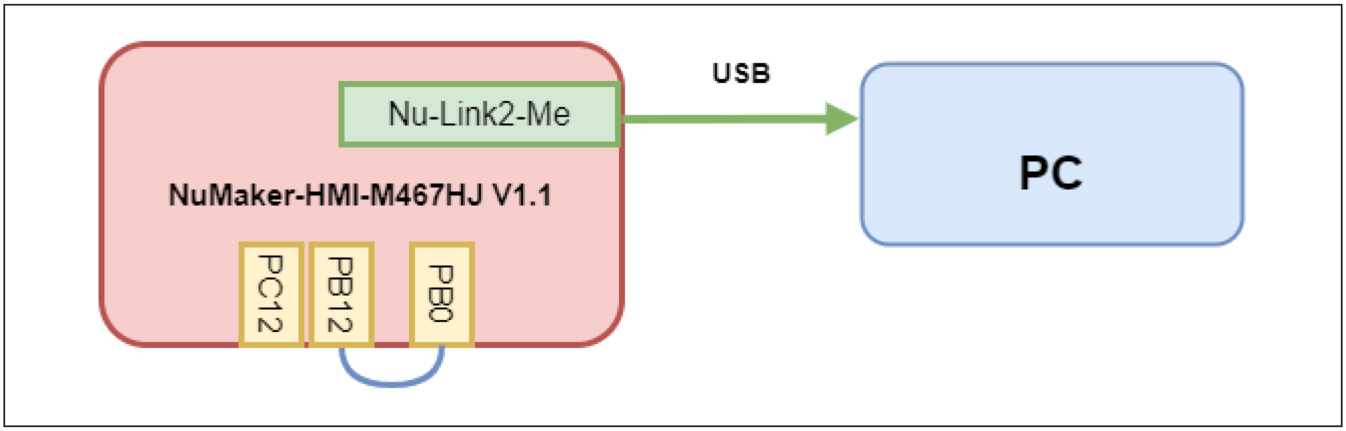


圖 3-1 NuMaker-HMI-M467HJ 和 PC 連接示意圖

# 4. 目錄資訊

| | |
|---|---|
| 🗁 EC_M467_emWin_Oscilloscope_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 | |
| 🗁 emWin_Oscilloscope | Source file of example code |
| 🗁 ThirdParty | Header files for emWin project |
| 🗁 Tool | Tool : GUIBuilder.exe |

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 *emWin_Oscilloscope.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
   - 執行結果

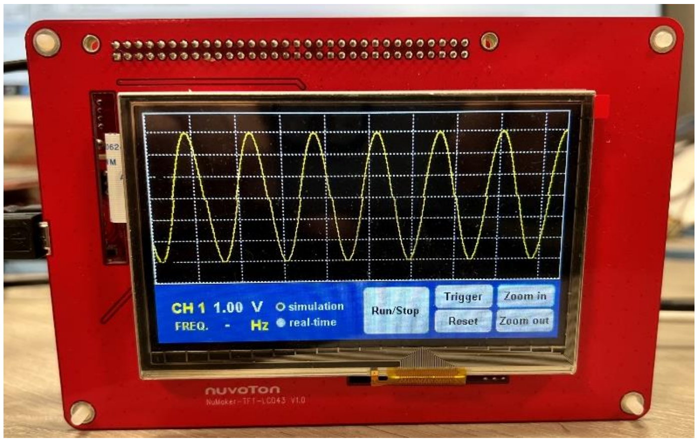


圖 5-1 NuMaker-HMI-M467HJ 執行結果圖

# 6. 修訂紀錄

| Date | Revision | Description |
|---|---|---|
| 2023.08.13 | 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.**