M480 emWin Graphic to Widget

Example Code Introduction for 32-bit NuMicro® Family

Information

Application	This example code utilizes API of emWin to draw graphic that can be used as a button to change value of text edit on the panel.
BSP Version	M480 Series BSP CMSIS V3.03.001
Hardware	NuMaker-PFM-M487 V3.0 & M487 Advance Ver4.0

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 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 Function Description

1.1 Introduction

Nuvoton has license emWin library from Segger. It is really simple to use API of emWin to develop human machine interface(HMI) on the product. This example code uses M480 MCU and utilizes API of emWin to draw graphic that can be used as a button to change value of text edit on the panel. emWin can easily implement how to use graphic as a widget, button to adjust the value of text edit on the NuMaker-PFM-M487 V3.0 board.

For more information on the NuMaker-PFM-M487 board and NuMaker-M487-Advance board platform for emWin, please visit Nuvoton website to download the User Manual document UM_NuMaker-PFM_M487_User_Manual and UM_NuMaker_M487_Advance_User_Manual, more detail about emWin. Please download the UM_NuMaker-emWin-M487_User_Manual:

https://www.nuvoton.com/hq/products/microcontrollers/arm-cortex-m4-mcus/User-Manual/?__locale=en&resourcePage=Y&category=&pageIndex=3

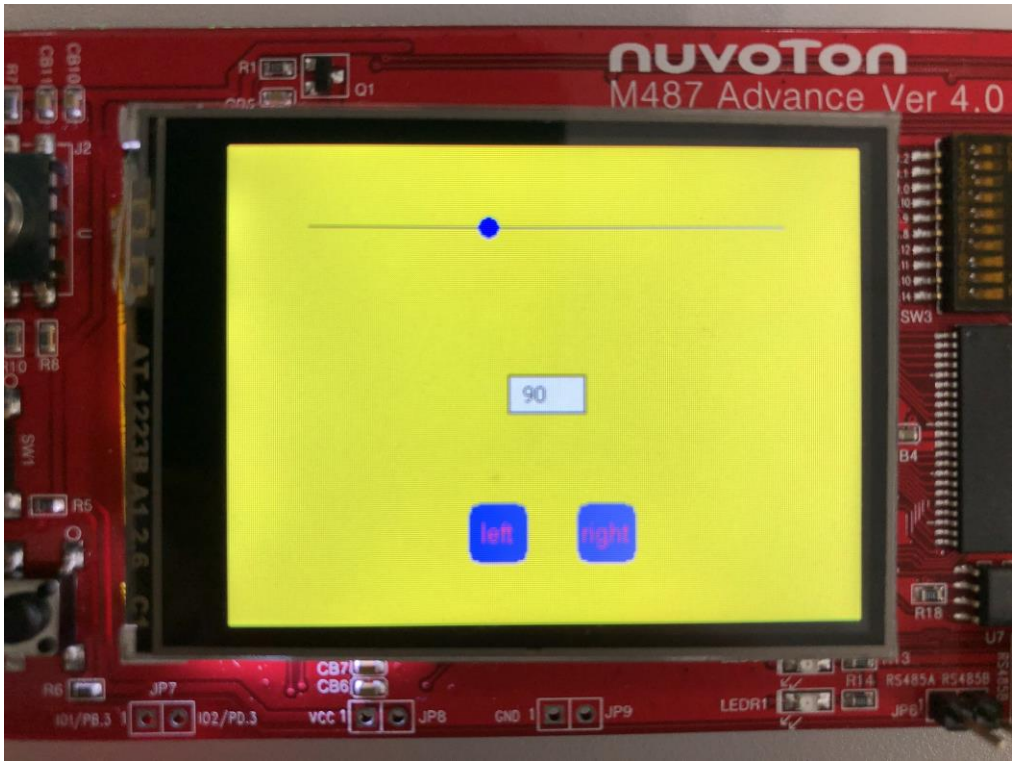
NuMaker emWin M487 Board

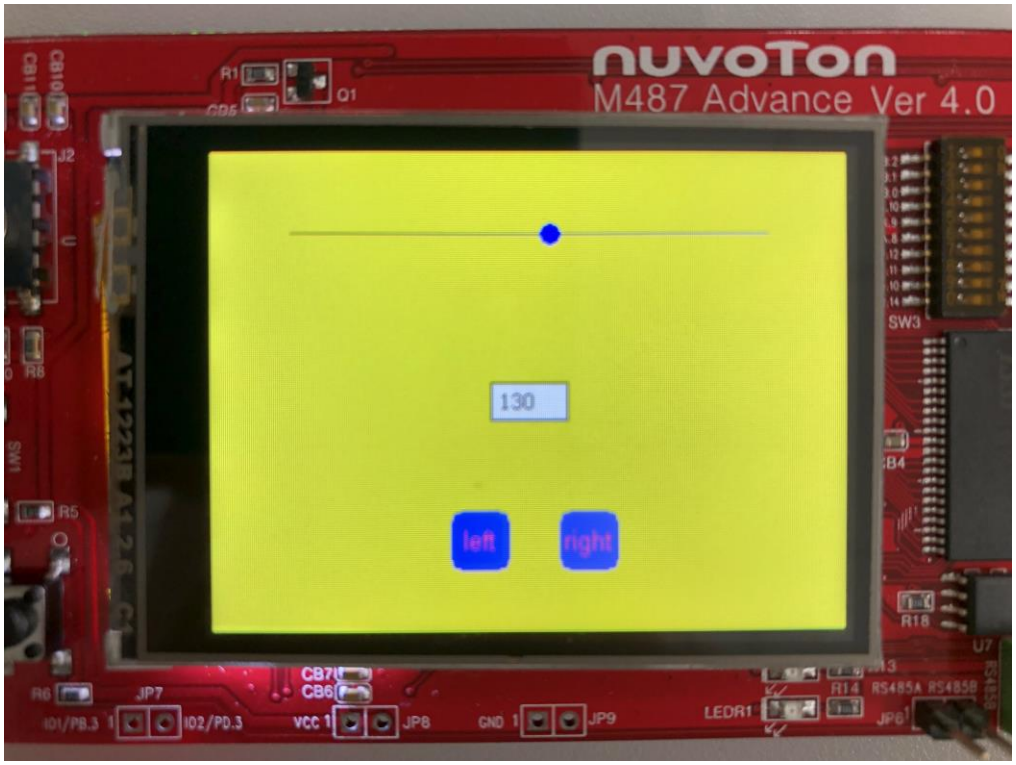
1.2 Principle

The mainly principle for user need to know how to use API to draw graphic. The sample code use window manager to create a frame window. User need to announce Callback function of button in the frame window. The Callback function need to draw the graphic as the button, controlling the value of text edit. As before creating a frame window, user need to initialize emWin first then can call any emWin API to arrive regarding emWin functions.

1.3 Demo Result

After executed this example code, user will be seeing two graphic buttons, controlling value of text edit and changing the position of circle on the line.





2 Code Description

This main() function is used to initialize platform for emWin include PLL clock to 192MHz, Uart clock, EBI, Timer and multi-function pins.

```
/**
 * @brief       main
 *
 * @param[in]   None
 *
 * @return      None
 *
 * @details     This function is used to initialize platform for emWin
 */

void main(void)
{
    /* Init System, IP clock and multi-function I/O */
    _SYS_Init();

    /* Init UART to 115200-8n1 for print message */
    UART_Open(UART0, 115200);

    /* Enable Timer0 clock and select Timer0 clock source */
    CLK_EnableModuleClock(TMR0_MODULE);
    CLK_SetModuleClock(TMR0_MODULE, CLK_CLKSEL1_TMR0SEL_HXT, 0);

     /* Initial Timer0 to periodic mode with 1000Hz */
    TIMER_Open(TIMER0, TIMER_PERIODIC_MODE, 1000);

     /* Enable Timer0 interrupt */
    TIMER_EnableInt(TIMER0);
    NVIC_SetPriority(TMR0_IRQn, 1);
    NVIC_EnableIRQ(TMR0_IRQn);

    /* Start Timer0 */
    TIMER_Start(TIMER0);
    /* Start application */
    GUI_Main();
    while(1);
```

This GUI_Main () function is used to emWin API to set parameter of emWin. First, GUI_Init() function is initialized for setting regarding emWin configuration and display driver link and emWin initialization in GUI_Main () function. Second, the frame window is used CreateWindow() function to create a window for parsing all action of wundow.

```
WM_HWIN CreateWindow(void)
{
  WM_HWIN hWin;


  hWin = GUI_CreateDialogBox(_aDialogCreate, GUI_COUNTOF(_aDialogCreate), _cbDialog,
WM_HBKWIN, 0, 0);
  return hWin;
}


void GUI_Main(void)
{
    GUI_Init();
    ts_calibrate(320, 240);
    g_enable_Touch = 1;
    WM_SetCreateFlags(WM_CF_MEMDEV);
    WM_MULTIBUF_Enable(1);
    GUI_SetTextMode(GUI_TM_TRANS);
    CreateWindow();
    while (1)
    {
        GUI_Exec();
    }
```

This _cbDialog () function is to handle all of actions of frame window, including window initialization, graphic updating, buttons parsing and so on.

```
static void _cbButton(WM_MESSAGE * pMsg)
{
     WM_HWIN  hWin;
     GUI_RECT Rect;
     hWin  = pMsg->hWin;
     switch(pMsg->MsgId)
     {
         case WM_PAINT:
              // Returns the size of the active window
              WM_GetClientRect(&Rect);
              if(BUTTON_IsPressed(hWin))
              {
                    // Set the current foreground color
                   GUI_SetColor(GUI_DARKGRAY);
                    // Draws a filled rectangle with rounded corners
                   GUI_FillRoundedRect(Rect.x0,Rect.y0, Rect.x1, Rect.y1, 5);
                    // Set the current background color
                   GUI_SetBkColor(GUI_DARKGRAY);
                   GUI_SetColor(GUI_LIGHTBLUE);
              }
              else
              {
                   ……
              }
              GUI_SetFont(&GUI_Font16_ASCII);
              GUI_DispStringInRect("right",&Rect, GUI_TA_HCENTER | GUI_TA_VCENTER);
              break;
}
static void _cbDialog(WM_MESSAGE * pMsg) {
  WM_HWIN hItem;
  int     NCode;
  int     Id;
  char sBuf[20];
  switch (pMsg->MsgId) {
  case WM_INIT_DIALOG:
    // buttons initialization
    hItem = WM_GetDialogItem(pMsg->hWin, ID_BUTTON_0);
    // Sets the callback routine for a window
    WM_SetCallback(hItem, cbButton);
```

```
    ……
    break;
  case WM_PAINT:
    // graphic updating
    GUI_SetBkColor(GUI_YELLOW);
    GUI_Clear();
    break;
  case WM_NOTIFY_PARENT:
    // widgets action parsing
    Id    = WM_GetId(pMsg->hWinSrc);
    NCode = pMsg->Data.v;
    switch(Id) {
    case ID_BUTTON_0: // Notifications sent by 'Button'
    // buttons processing
      switch(NCode) {
      case WM_NOTIFICATION_CLICKED:
        Draw_degree+=10;
        if(Draw_degree>=240) Draw_degree=240;
        sprintf(sBuf,"%d    ", Draw_degree);
        hItem = WM_GetDialogItem(pMsg->hWin, ID_EDIT_0);
        // Sets the text
        EDIT_SetText(hItem, sBuf);
        // Invalidates a window
        WM_InvalidateWindow(pMsg->hWin);
        break;
      }
        ……
}
```

WM_GetClientRect()

Description

Returns the coordinates of the client area in the active window in window coordinates. That means x0 and y0 of the GUI_RECT structure will be 0, x1 and y1 corresponds to the size - 1.

Prototype

```
void WM_GetClientRect(GUI_RECT * pRect);
```

Parameter	Description
pRect	Pointer to a GUI_RECT structure.

GUI_SetBkColor()

Description

Sets the default background color.

Prototype

```
GUI_COLOR GUI_SetBkColor(GUI_COLOR Color);
```

Parameter	Description
Color	Color for background, 24-bit RGB value.

GUI_FillRoundedRect()

Description

Draws a filled rectangle at a specified position in the current window with rounded corners.

Prototype

```
void GUI_FillRoundedRect(int x0, int y0, int x1, int y1, int r);
```

Parameter	Description
x0	Upper left X-position.
y0	Upper left Y-position.
x1	Lower right X-position.
y1	Lower right Y-position.
r	Radius to be used for the rounded corners.

GUI_SetColor()

Description

Sets the current foreground color.

Prototype

```
void GUI_SetColor(GUI_COLOR Color);
```

Parameter	Description
Color	Color for foreground, 24-bit RGB value.

WM_GetDialogItem()

Description

Returns the window handle of a dialog box item (widget).

Prototype

```
WM_HWIN WM_GetDialogItem(WM_HWIN hDialog, int Id);
```

Parameter	Description
hDialog	Handle of dialog box.
Id	Window Id of the widget.

WM_SetCallback()

Description

Sets a callback routine to be executed by the Window Manager.

Prototype

```
WM_CALLBACK * WM_SetCallback(WM_HWIN hWin, WM_CALLBACK * cb);
```

Parameter	Description
hWin	Window handle.
cb	Pointer to callback routine.

EDIT_SetText()

Description

Sets the text to be displayed in the EDIT widget.

Prototype

```
void EDIT_SetText(EDIT_Handle hObj, const char * s)
```

Parameter	Description
hObj	Handle of EDIT widget.
s	Text to display.

WM_InvalidateWindow()

Description

Invalidates a specified window.

Prototype

```
void WM_InvalidateWindow(WM_HWIN hWin);
```

Parameter	Description
hWin	Window handle.

3 Software and Hardware Environment

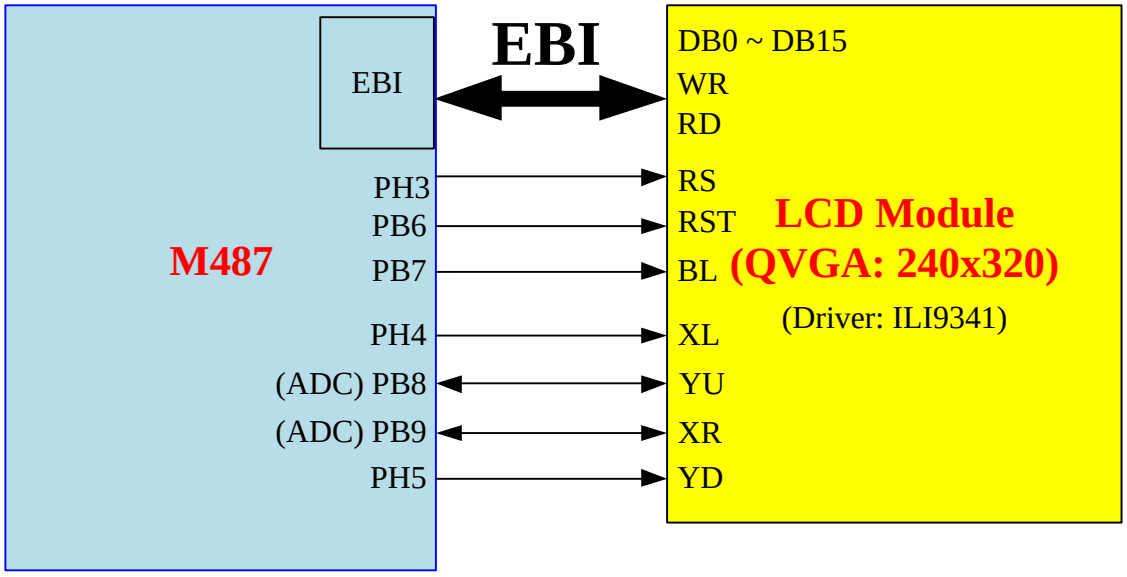
Software Environment

- BSP version
  - M480 Series BSP CMSIS V3.03.001
- IDE version
  - Keil uVersion 5.26

Hardware Environment

- Circuit components
  - NuMaker-PFM-M487 V3.0 board
  - M487 Advance Ver4.0
- Diagram

The below diagram is to show how to connect LCD module with M487 including EBI interface for communication, determination data or command by RS pin, module reset by RST pin, backlight control by BL pin and two ADC pin for measuring x and y coordinate。



4 Directory Information

- EC_M480_emWin_SimpleDemo_ GraphicToWidget_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 — Source file of example code
  - ThirdParty
    - emWin — emWin is designed to provide an efficient, processor- and display controller-independent graphical user interface for any application that operates with a graphical display.

5 How to Execute Example Code?

1. Browsing into sample code folder by Directory Information (section 4) and double click emWin_SimpleDemo.uvproj.
2. Enter Keil compile mode
   a. Build
   b. Download
   c. Start/Stop debug session
3. Enter debug mode
   a. Run

6 Revision History

Date	Revision	Description
Jul. 18, 2019	1.00	1. Initially issued.

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.