EmbeddedGUI Quick Start Guide

Version 0.6.2

Wang Chao

eluneyun@gmail.com

Aug 2011

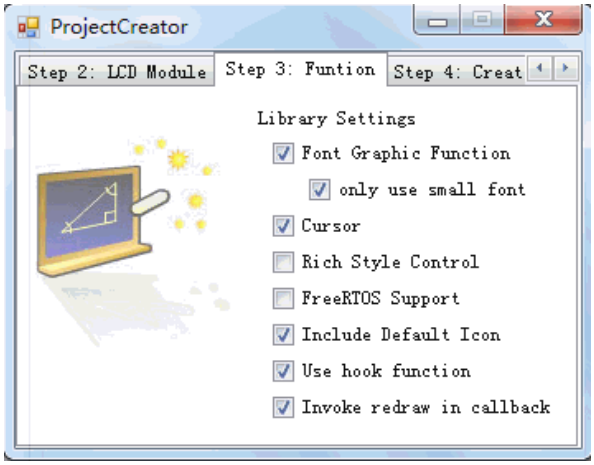
1. Introduction

The EmbeddedGUI Library is designed for embedded systems with very small RAM and flash space. The LCD interfaces and graphic functions are implemented in C. It also provides many controls and windows implemented in C++.This library is designed without any dynamic memory allocation. So it is easy to be transplanted to any embedded systems based on ARM or DSP. It also supports FreeRTOS and can be easily transplanted to any other RTOS. EmbeddedGUI only takes about 5.5KBytes flash and 2KBytes RAM space for a simple application using 128*64 resolution monochrome LCD.

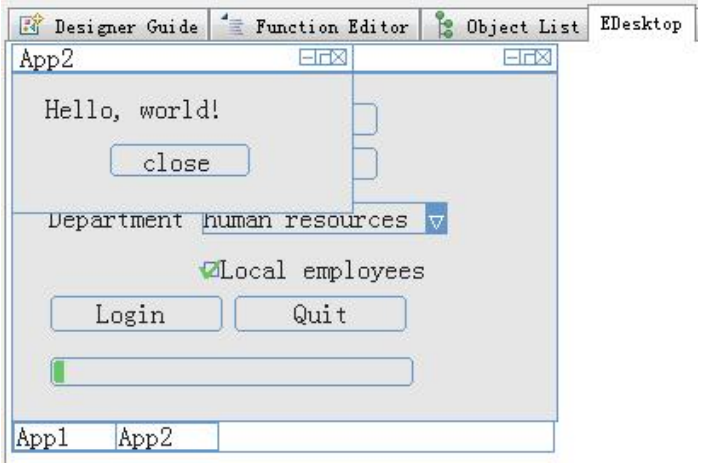
2. EmbeddedGUI Designer

EmbeddedGUI Designer is a layout tool which can generate code automatically. Note that EmbeddedGUI Designer 0.6 requires Microsoft .NET Framework4.

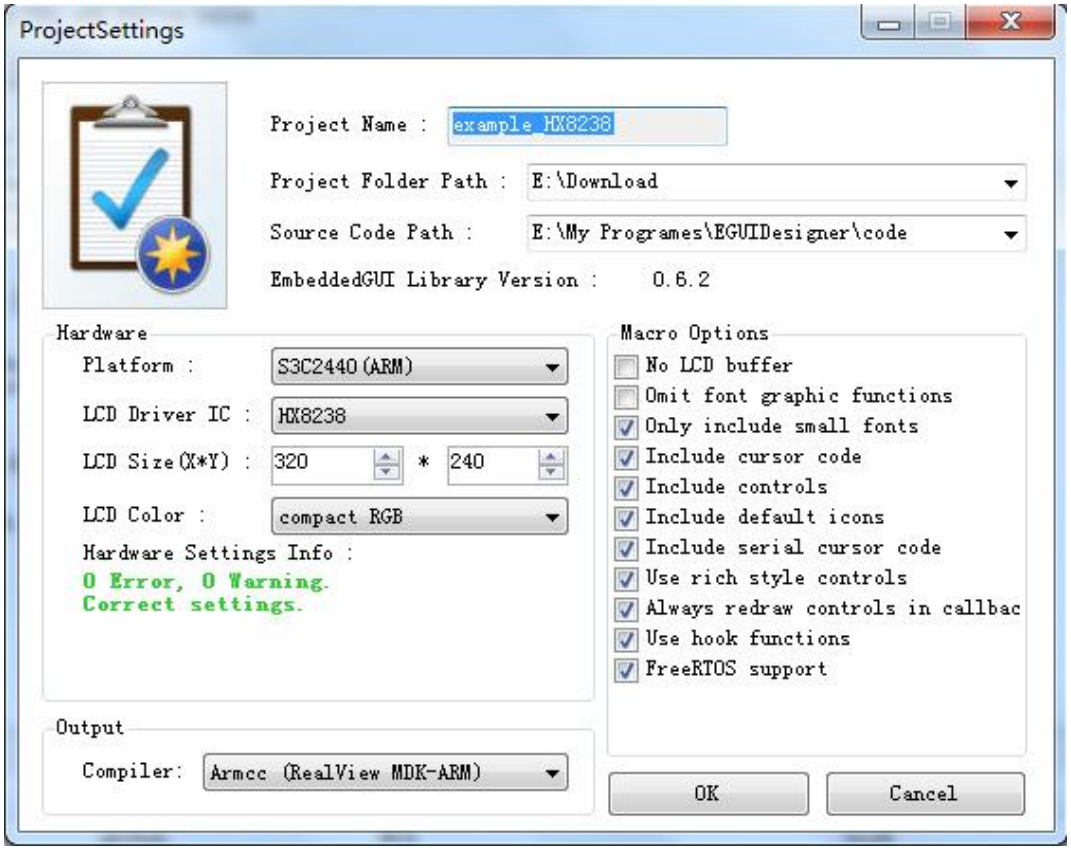
Step 1　Create a new project



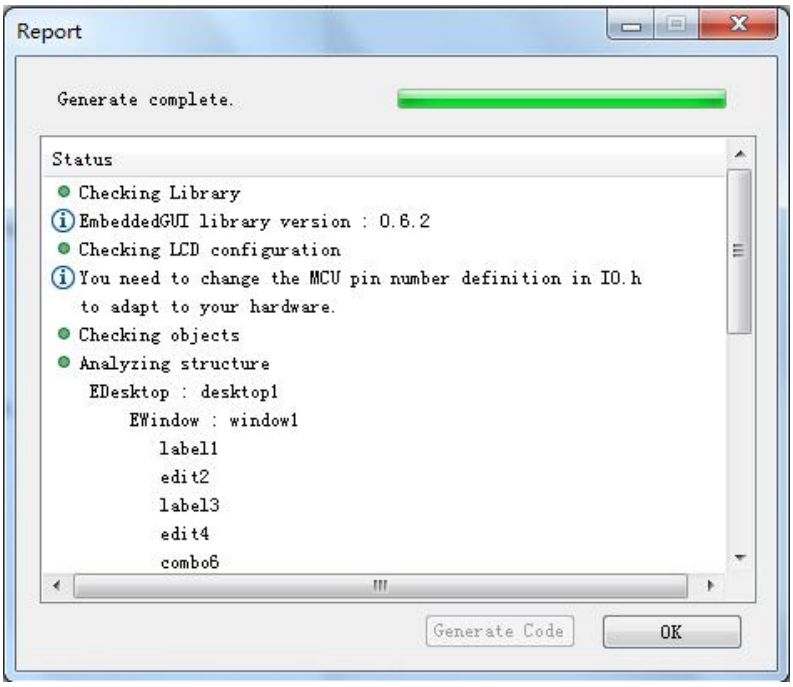
Step 2　Add GUI object



Step 3　Project configuration



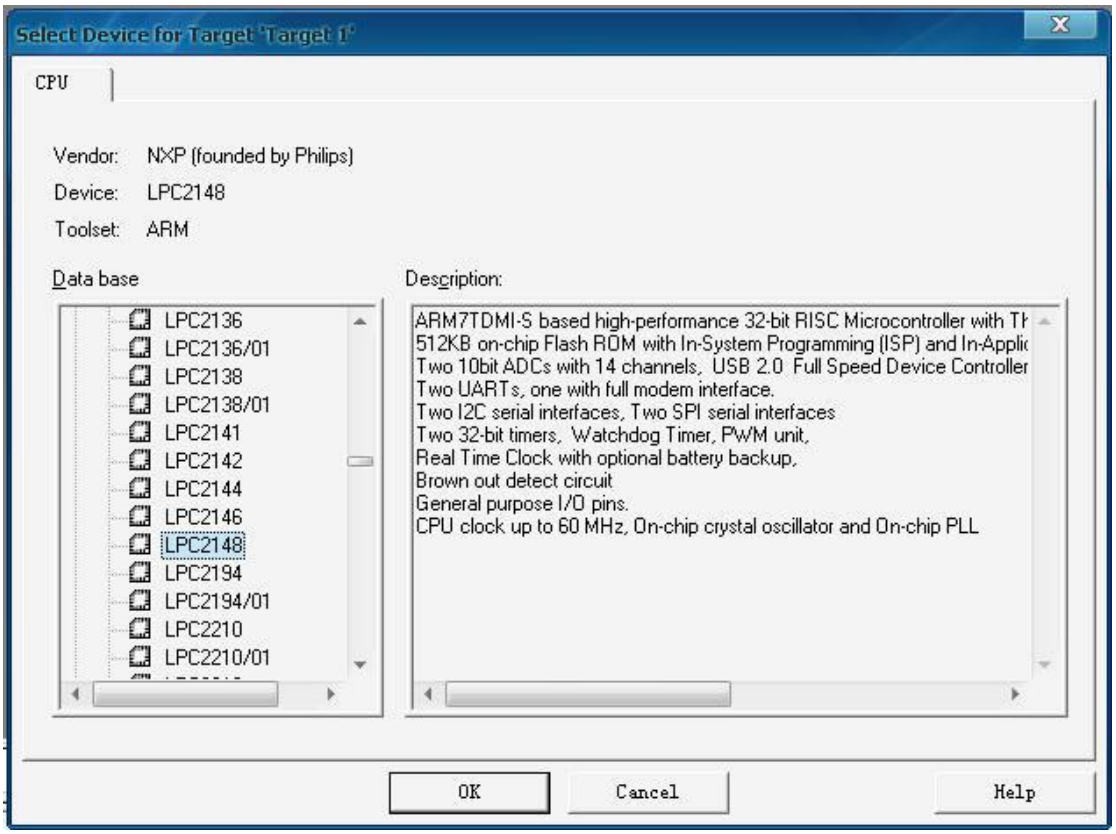
Step 4　Generate code



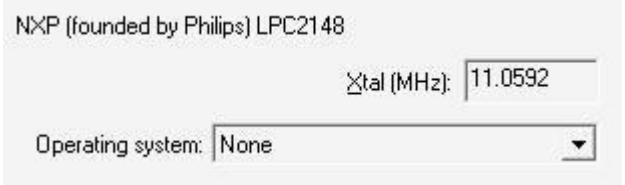
3. EmbeddedGUI Quick Start

Step 1　Create a new project

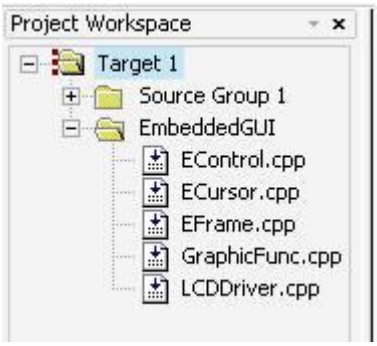
In this section, we will use the MDK μVision3 as the IDE. First of all,we need to create a new project. Choose the MCU of your evaluation board, which is LPC2148 in this example shown in Figure 1.



On my evaluation board, the crystal oscillator is 11.0592MHz. So I also need to edit the Xtal value.



Add the source code to the project.



Step 2 Configure the EmbeddedGUI

You can skip this step, if the EmbeddedGUI files are generated by EmbeddedGUI Designer.

Open the head file EGUIMacro.h and edit the macro definitions according to your hardware. After this configuration, you can start to write the main program now. Note that EmbeddedGUI contains C++ code. You need to create a CPP file for main function in order to invoke C++ compiler. If your main.c is already existed, please refer to the FreeRTOS example.

Include LCDDriver.h, EControl.h and EFrame.h at the very beginning of main.cpp. If your main function wants to call any C function, make sure you have use the extern"C".

```
    ESimpleDesktop desk;
    EWindow introWin("EGUIGuide\0",EPosition(0,0),ESize(127,63),true);
extern "C"{
int main(){
    EEvent e;
    //init the LCD
    EG_LCDInit();
    //Clear the buffer, set the buffer to 0
    EG_LCDClearBuffer();
    desk.AddWindow(&introWin);
    // Global Redraw upon the buffer
```

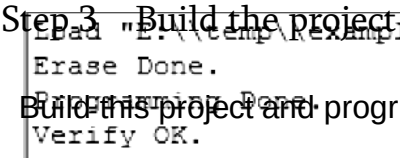
```
        desk.GlobalRedraw();
        while(1){
            // edit the event here...
            //e.m_uSource=1;
            //e.m_uMessage=1;
            //e.m_pPosition.m_uX=20;
            //e.m_pPosition.m_uY=20;
            //dispatch this event
            desk.DispatchEvent(&e);
        }
        return 0;
    }
    }
```

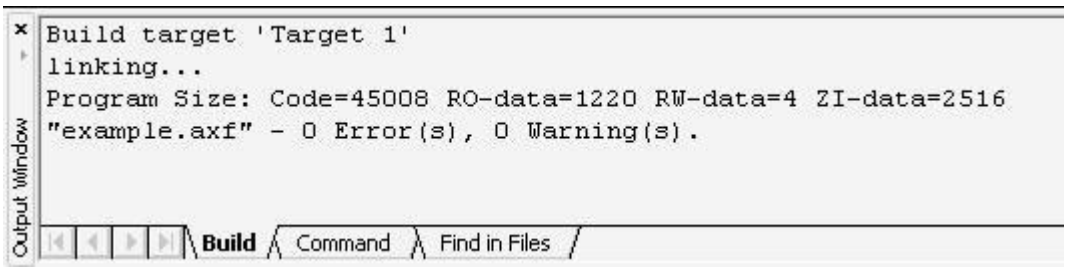
Remember to invoke the ESimpleDesktop::ScheduleRedraw() in the ISR of a timer. In this function, GUI system will redraw itself. Failing to call this function, LCD will display the same image all the time.

```
__irq void timerISR(){
    unsigned int* upoint=(unsigned int*)0xE0008000;
    *upoint=0x01;   //reset the timer1 ISR
    upoint=(unsigned int*)0xFFFFF030;
    *upoint=0x00;   /*Vector Address register should be written near
the end of an ISR to update the priority hardware.*/
    desk.ScheduleRedraw();
}
```

Step 3. Build the project and debug

Build this project and program the Hex File into your device.





4. FAQ

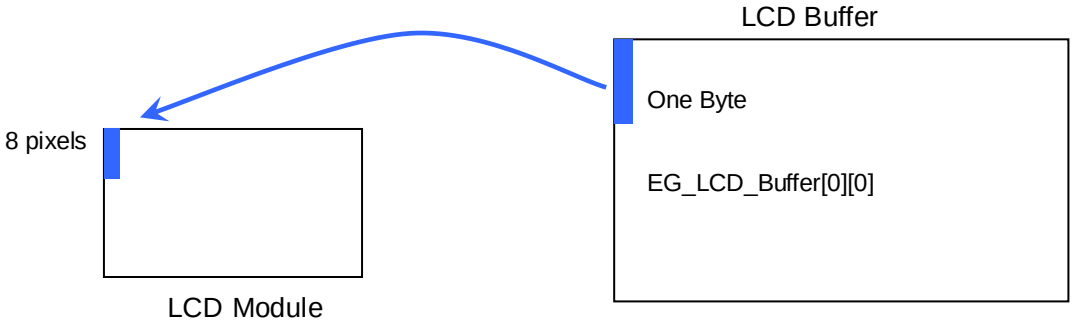
Question 1 :

How many color modes does EmbeddedGUI support?

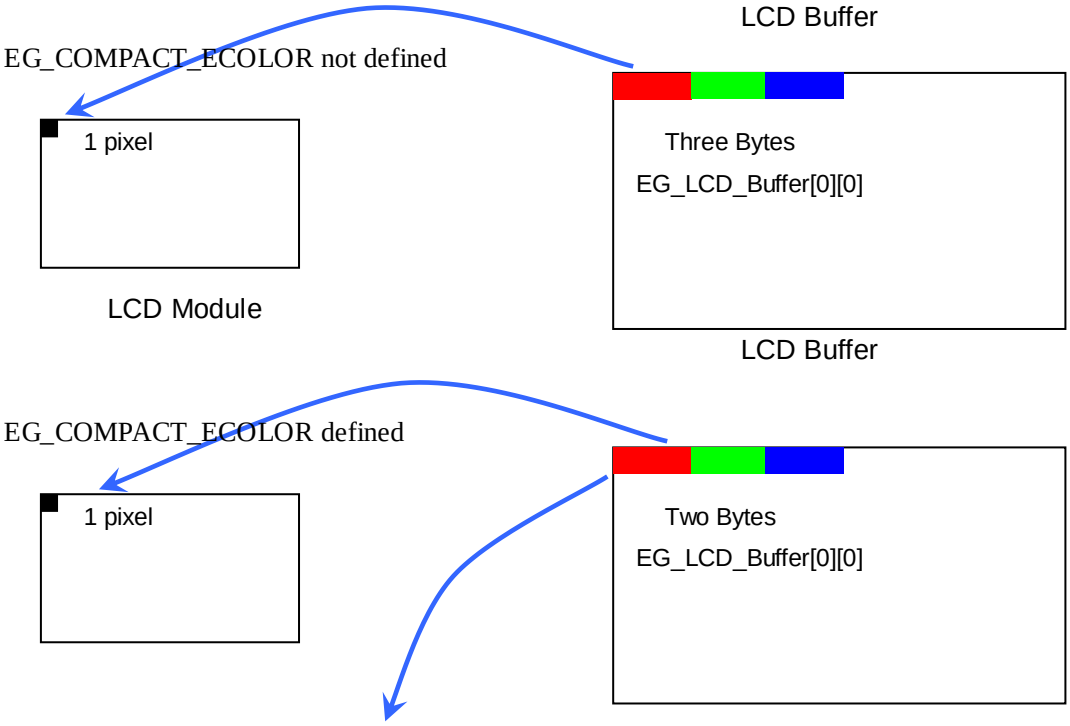
Answer :

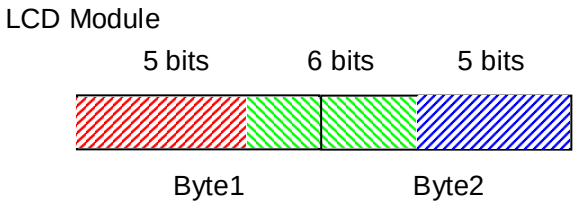
In current system, we have three color modes:

1 Single color mode , which is used by the most monochrome LCD module.In this mode, every pixel takes only one bit. The memory organization is as follows:



2 RGB color mode, every pixel takes two or three byte,depending on the memory size. Some LCD module uses two bytes to represent a pixel in order to save more RAM space.





Question 2:

How many LCD drivers can EmbeddedGUI support? How do I use them?

Answer:

There are six LCD drivers supported now:

KS0107 & KS0108
T6393C
S6B33C
RA8803
S1D13305
HX8238A

If your LCD module uses one of these driver IC, you can use them directly. Include the LCDDriver.h to your project and define one of these macros in EGUIMacro.h.

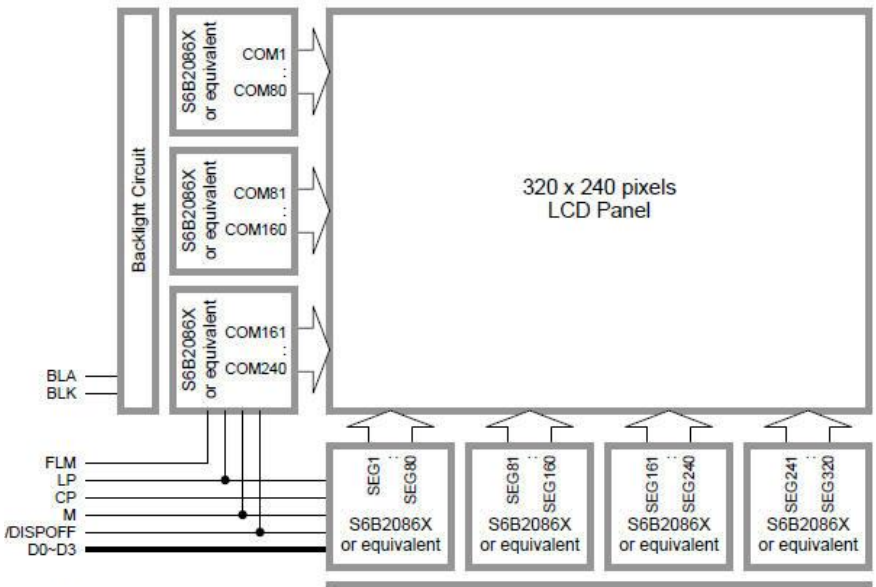
EG_KS010X_LCD_DRIVER
EG_T6963C_LCD_DRIVER
EG_S6B33C_LCD_DRIVER
EG_RA8803_LCD_DRIVER
EG_S1D13305_LCD_DRIVER
EG_HX8238_LCD_DRIVER

Question 3:

I don't know the LCD driver IC of my LCD module.

Answer:

In the datasheet or manual of your LCD module, the manufacturer will give you its schematic. This schematic will probably be like this :



The S6B2086X which directly connected to LCD panel is the LCD driver IC.

Question 4:

How to program my own LCD driver?

Answer:

If your LCD module uses other driver IC, you have to program your own driver. There are only three APIs you need to implement.

```
void EG_LCDInit(void);      // initialize the LCD module.
void EG_LCDClearScreen(void);      // clear the LCD screen
// copy the data from EG_LCD_Buffer to LCD and display it
void EG_LCDFlushBuffer(void);
```

Question 5:

I don't need a desktop in my application.

Answer:

Use the ESimpleDesktop instead. The EDesktop provides a GUI interface of a desktop, while the ESimpleDesktop doesn't. ESimpleDesktop only contains the functions to manage the windows. Its redraw function only erase the background.

Question 6:

I don't understand the event mechanism of EmbeddedGUI.

Answer:

The event mechanism is same as the message in Win32 API. It also has lots similarities with MFC. The event will be dispatched from desktop to window and handled by the callback function of control at last. In the message mapping in MFC, messages will also be sent to bottom level as well.

In Win32 programming, most of the messages are received from the

operating system. In the EmbeddedGUI, however, the event is generated by user. The disadvantage is that you need to generate all the events by yourself.The advantage is that you can generate the custom events, which is usually very useful in the embedded system.

Eg:

```
EEvent e;
e.m_uSource=1;
e.m_uMessage=1;
e.m_pPosition.m_uX=20;
e.m_pPosition.m_uY=20;
// This event indicates that a cursor click happens at (20,20)
desk.DispatchEvent(&e);
```

Question 7:

What do I need to configure?

Answer:

All the macros you need to configure are in the EGUIMacro.h . You need to define some of them according to the hardware platform. In the layout tool , we will provide a convenient way to configure all these macros.