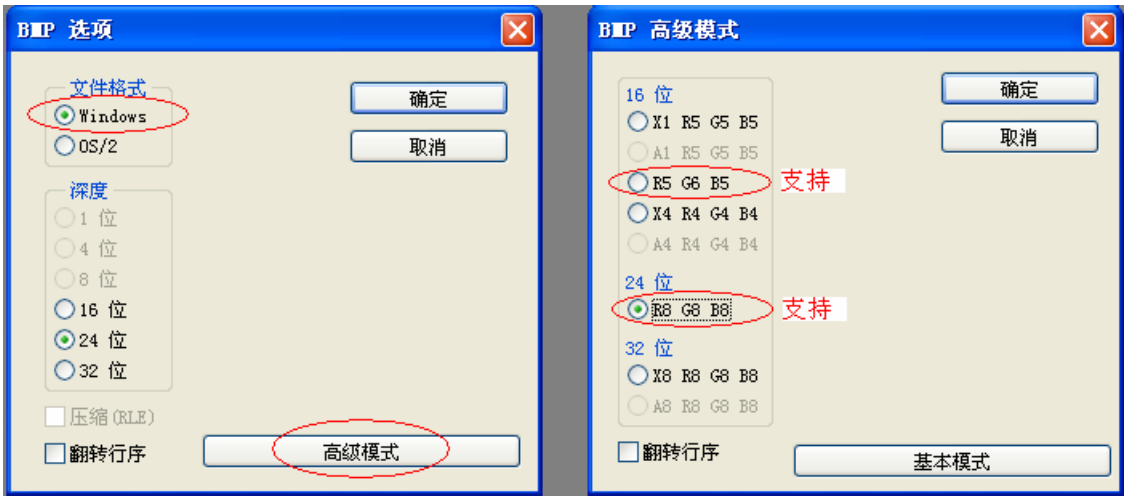
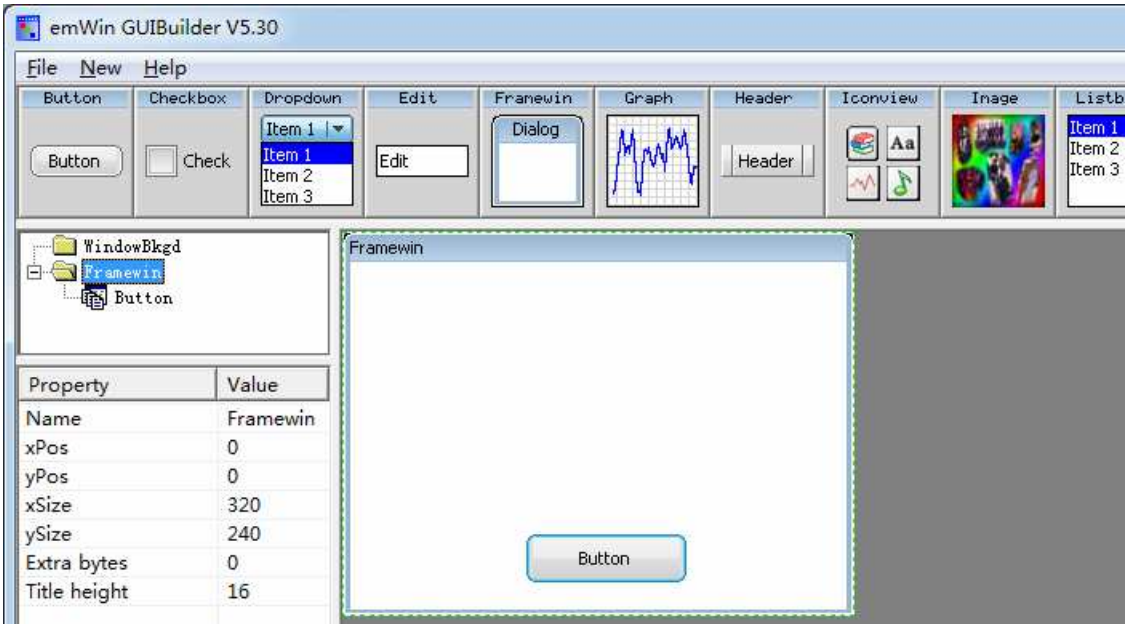
emwin 2-day quick tutorial 009_Bitmap 位图显示以及位图皮肤的使用方法

1. 在 CodeBlocks 中使用位图

备注:GLCD 屏支持 24bbp(RGB888)、16bbp(RGB565)格式的 BMP 位图(不支持其他格式的位图),16bbp(RGB565)BMP 位图可以通过 photoshop 保存得到.因此,在 CodeBlocks 中也需要使用这种格式的位图,以 photoshop 为例:



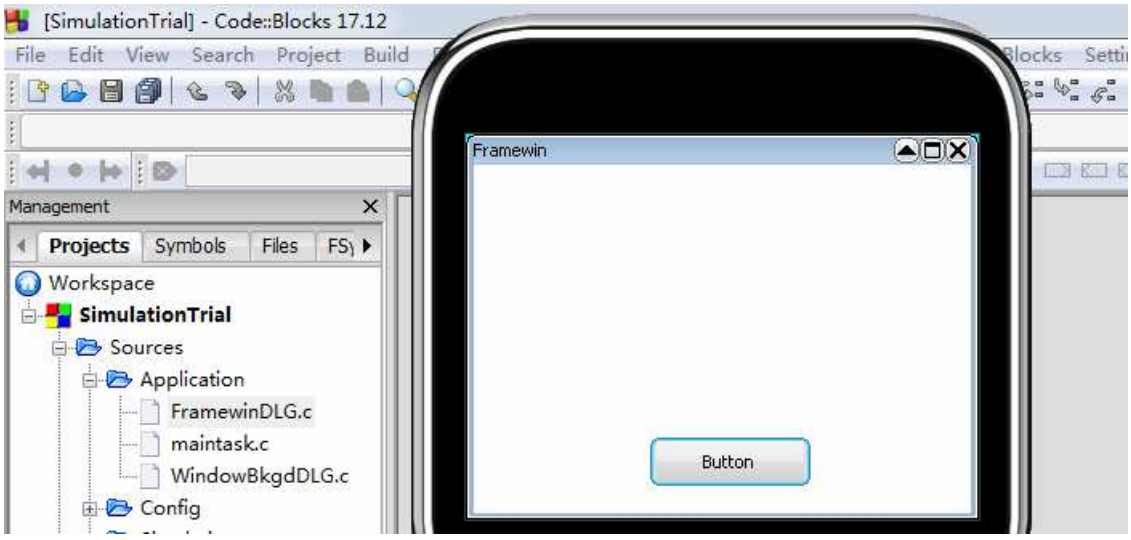
(1)使用 GUIBuilder V5.30 创建如下页面并保存为 c 文件,然后添加到 CodeBlocks 工程:



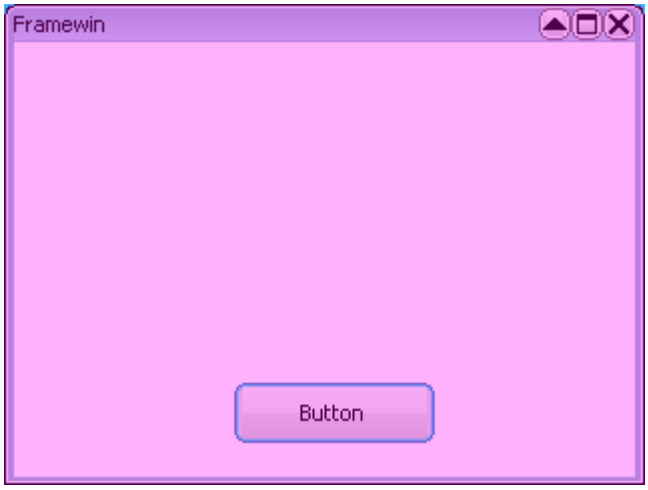
小技巧:放置 xxxx 控件之后不要用鼠标移动,用上下左右建移动更容易对齐(步进是 5).

备注:GUIBuilder 生成的 c 文件,用户代码最好加在"USER START"和"USER END"之间,其他地方除了数字以外,不要做任何修改,否则 GUIBuilder 将无法再次打开此 c 文件;另外, GUIBuilder 再次打开编辑并保存时,"USER START"和"USER END"之间的内容将不会被更改;还有 c 文件不要在 GUIBuilder 打开状态直接去修改 c 代码,否则点 GUIBuilder 保存之后修改的内容将会丢失.

编译运行:



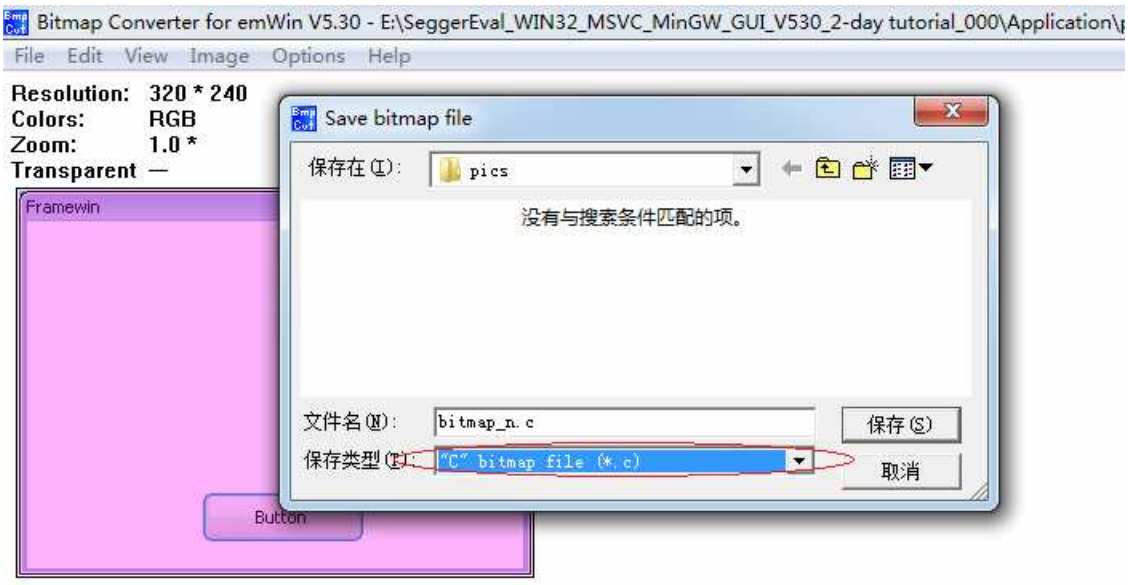
(2)截屏模拟器并保存为以下两个 bmp 文件,一个是界面截屏(为了区别截屏界面,特地变了下颜色),另一个是界面截屏的反色,文件名对应为 bitmap_n.bmp、bitmap_p.bmp:

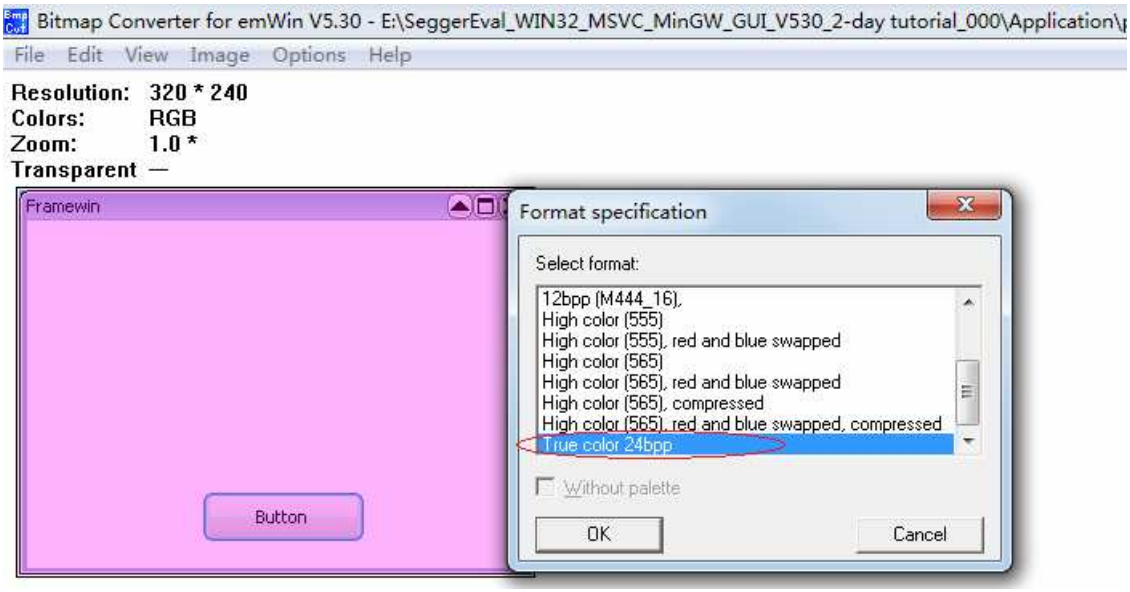


变色的用于正常背景显示,反色的用于按键点击.

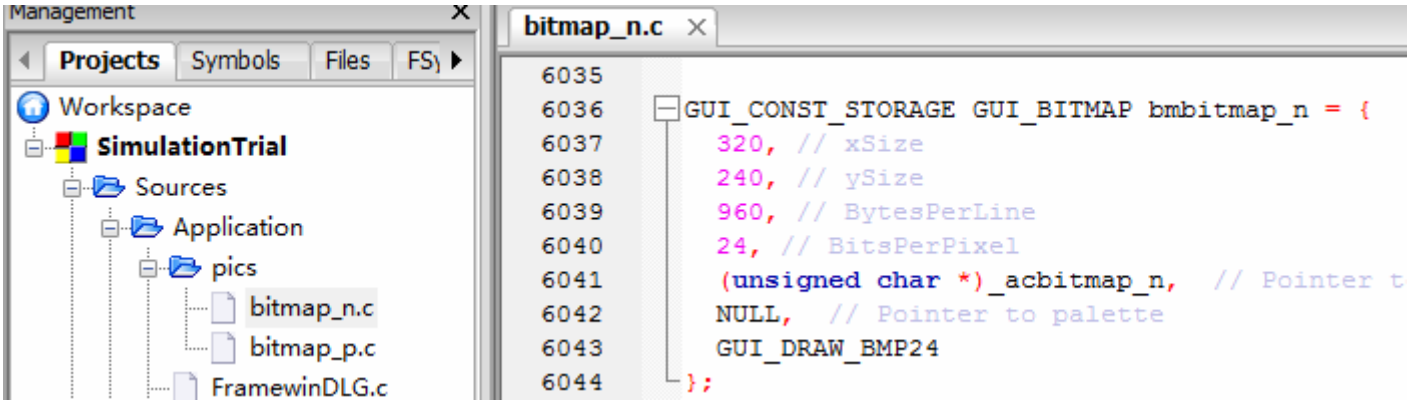
备注:如果你想要精美华丽的界面效果,也是按照此方法:截屏原始界面->美工处理->设置 emWin 使用图片皮肤(整幅图片,不需要切片分解).

(3)用 BmpCvt_V530.exe(Tool 目录下)将位图转换为 c 文件:bitmap_n.c、bitmap_p.c





(4)将 bitmap_n.c、bitmap_p.c 拷贝并添加到 CodeBlocks 工程:



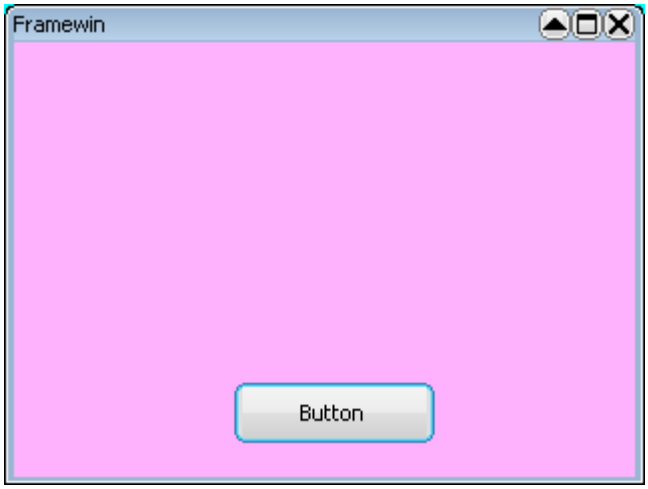
看到这里,你可能会想到如果是 MCU 平台,图片数据存到哪里?怎样读取图片数据?储存数据可以考虑 NAND-FLASH、SPI-FLASH、NOR-FLASH 等等...,至于怎样读取图片数据,这是 emWin 底层驱动程序的事情,在这里我们只说怎样做 emWin 的界面应用程序而已.GLCD 的图片是通过 USB 下载到屏端的 Nand-Flash 中,调用显示已经在 emWin 例程中的底层驱动做好.

(5)修改程序显示背景位图:

DLG.c

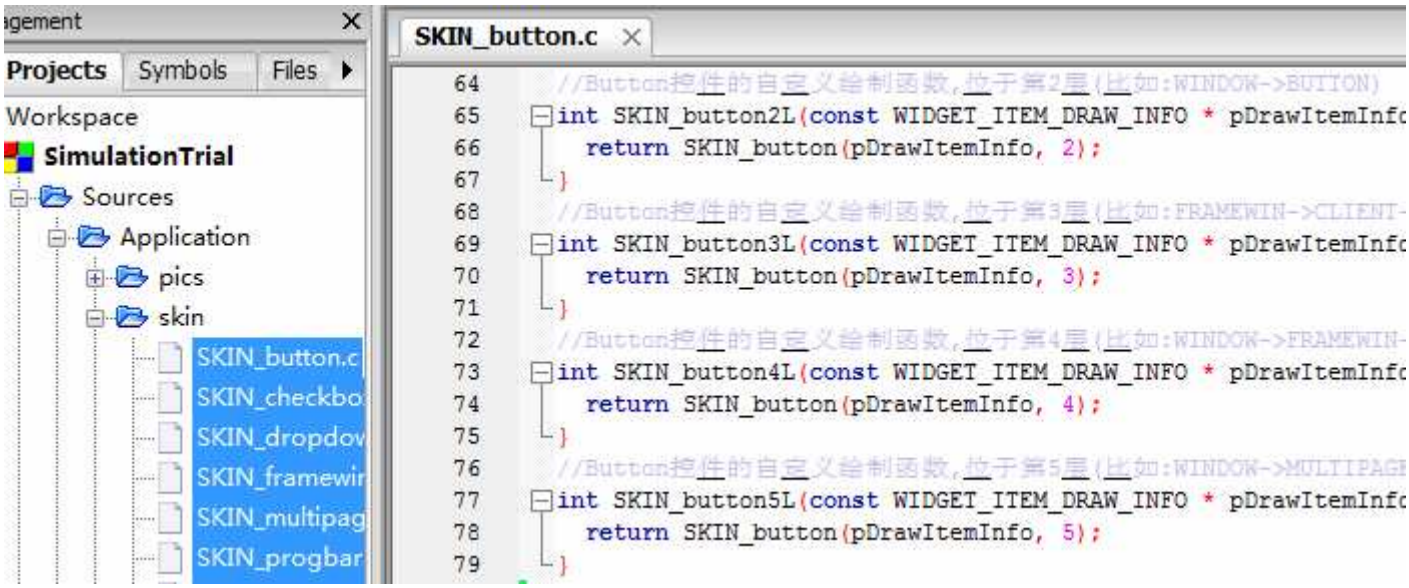
```
// USER START (Optionally insert additional message handling)
//窗口重绘消息,这个比较难说明白,反正在Framewin或Window窗口之中我们一般是用控件,如果
//Window redraw message, this is difficult to explain,anyway,we usually use widget
case WM_PAINT:
  posiX = WM_GetWindowOrgX(WM_GetParent(pMsg->hWin))-WM_GetWindowOrgX(pMsg->hWin);
  posiY = WM_GetWindowOrgY(WM_GetParent(pMsg->hWin))-WM_GetWindowOrgY(pMsg->hWin);
  GUI_DrawBitmap(&bmbitmap_n, posiX, posiY);
  break;
// USER END
default:
```

备注:因为 GUI_DrawBitmap()只显示在 Client 区,posiX 和 posiY 是利用坐标计算偏移掉图片的边框,让其对准.当然,你也可以做成不含边框只留 Client 区的图片.

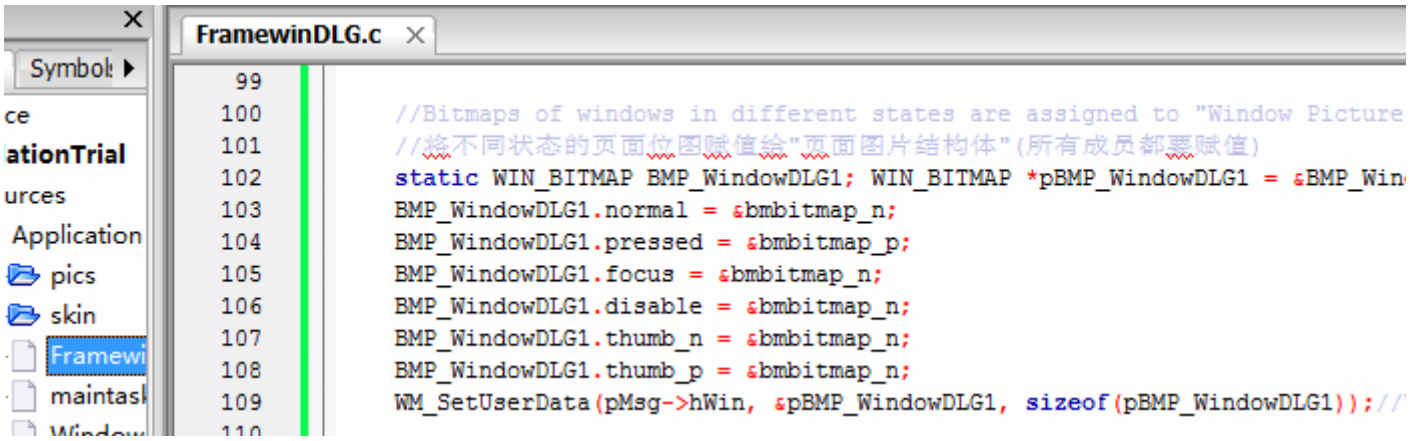


(6)按钮位图皮肤的实现:

将 skinPRJ_WINemWin530_NoOS_CodeBlocks_GBK.zip 例程(在 GLCD demo 代码里面可以找到)的 Application/skin 目录拷贝到工程目录并添加到 CodeBlocks 工程,skin 目录的程序是尼奇光电编写的 skinning 方式实现位图皮肤程序(关于实现原理可以看文章的最后):

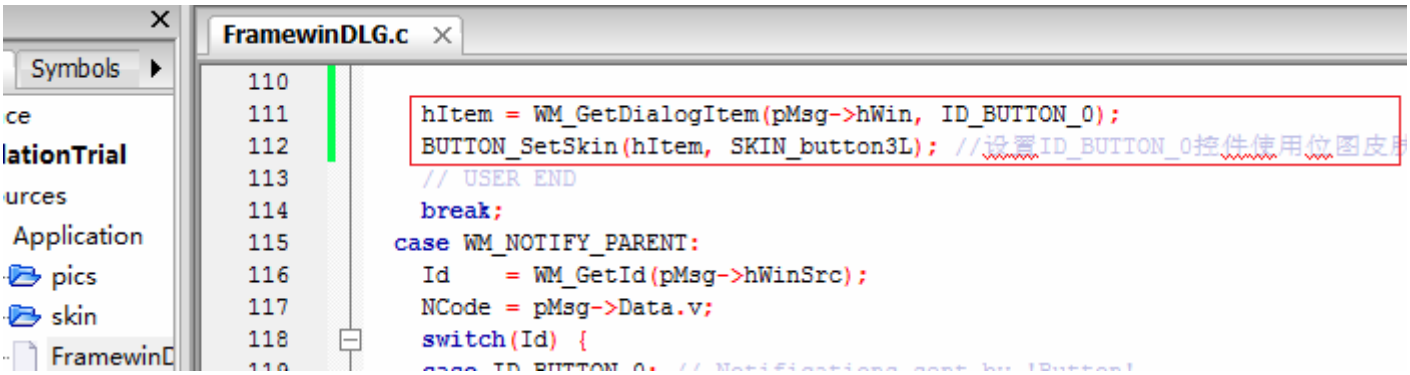


在 WM_INIT_DIALOG 消息中(窗口初始化时程序跑到这里)将不同状态的页面图片赋值给"页面图片结构体":



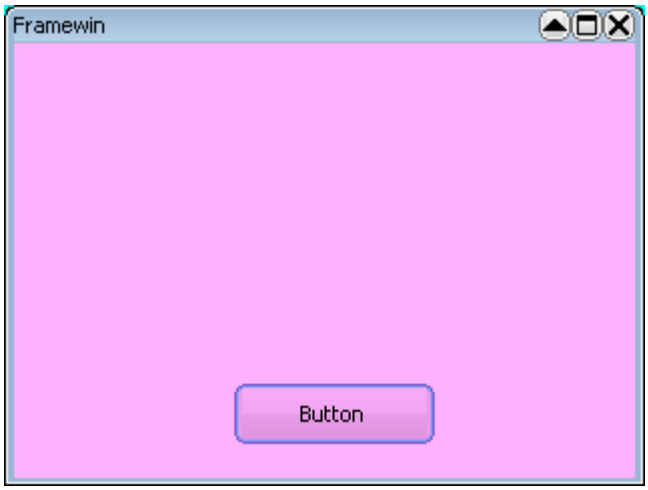
这个步骤的目的是将 2 个状态的皮肤图片的"指针"存于父页面之中,这个父页面之中的所有子控件就可以共享这幅图片(这就是整幅图片不需要切片分解的原理),如果将所有子控件设置为使用该皮肤图片时,emWin 内部会自动调用图片作为皮肤(不要想着还要自己去实现).2 个状态的图片包括:正常、按下/标记(比如选择控件的打勾状态),还有"禁用"和"聚焦"是可选状态.

设置控件使用位图皮肤:



SKIN_button3L "3L"的意思是存有图片"指针"的父页面到这个控件的层数,在这里 BUTTON 控件是 Framewin 页面的子控件,因此他们的子父关系是 FRAMEWIN->CLIENT->BUTTON,即 BUTTON 位于 Framewin 父窗体的第 3 层.

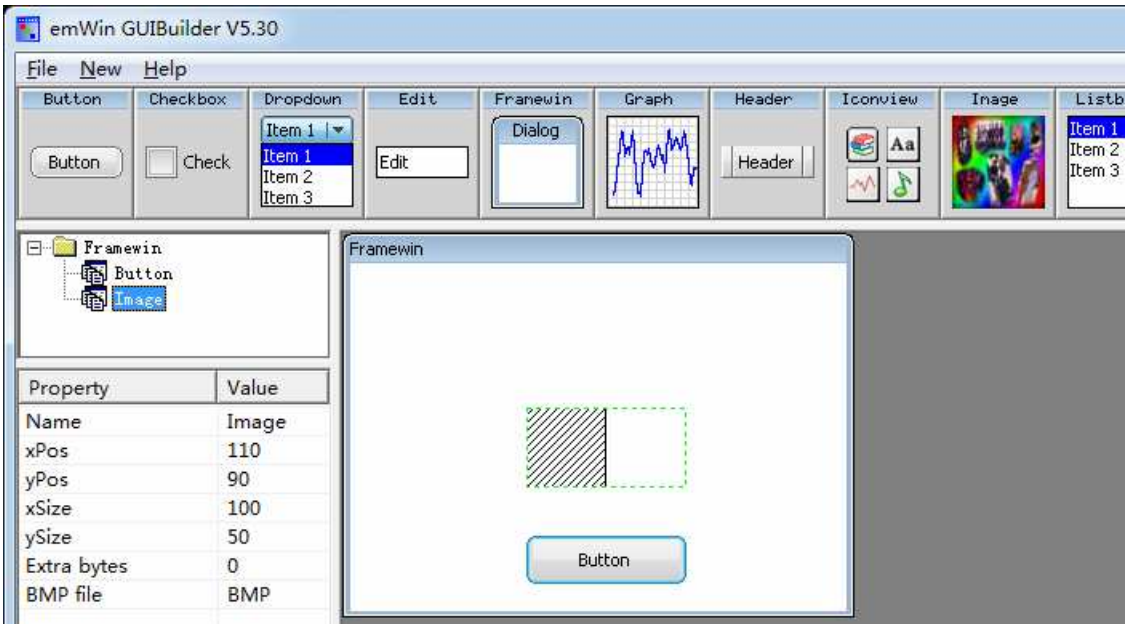
编译运行:



看到这里你可能会想,如果我想连 Framewin 的边框也使用图片皮肤怎办?那么你可以使用 Window 窗口,再做有边框的位图皮肤.

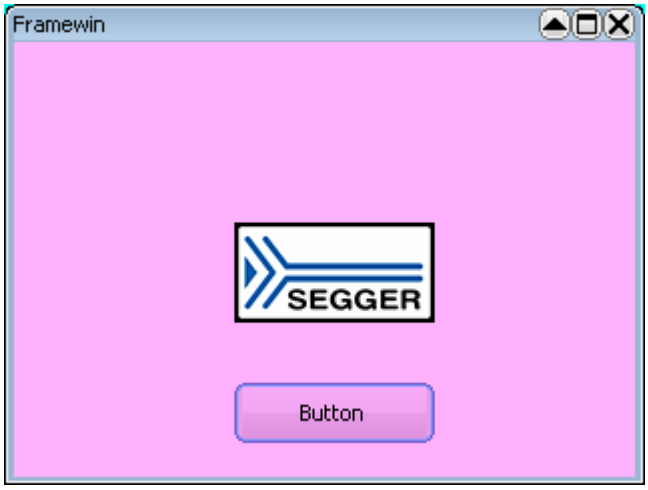
2. 使用 Image 控件显示图片:

除了背景和位图皮肤以外,要在界面某个位置显示一张图片,用 Image 控件会方便很多.使用 GUIBuilder V5.30 打开之前生成的 c 文件,并添加一个 Image 控件然后点保存,之前加在"USER START"和"USER END"之间的用户代码是不会被更改的:



在 WM_INIT_DIALOG 消息中(窗口初始化时程序跑到这里)将图片指针赋值给 Image 控件:

```
IMAGE_SetBitmap(WM_GetDialogItem(pMsg->hWin, ID_IMAGE_0), &bmlogo);
```



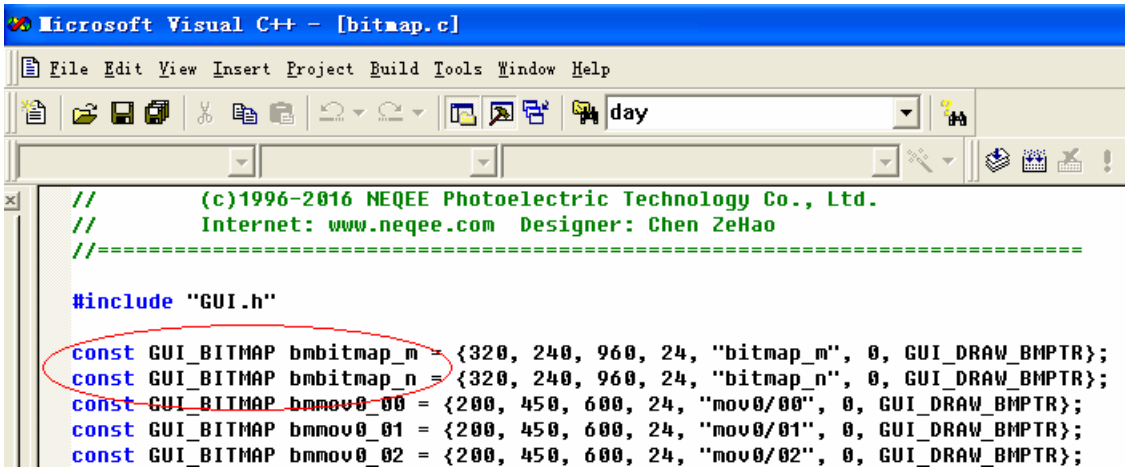
3. 在 GLCD 屏中使用位图

在 GLCD 屏中使用位图,除了生成 bitmap_n.c、bitmap_p.c 不同以外,其他和 CodeBlocks 是一样的.

1)将 bitmap_n.bmp 和 bitmap_p.bmp 拷贝到 GLCD 的 Nand-Flash 的 pics 目录.

2)双击 Nand-Flash 根目录下的 bmpGen_v121.exe 工具,将生成 bitmap.c 和 bitmap.h 两个文件.

3)将 bitmap.c 和 bitmap.h 加到 Keil MDK 工程,即可像 CodeBlocks 一样使用 bitmap_n、bitmap_p 两个位图:



4. emWin 通过 skinning 方式实现控件位图皮肤的原理

①emWin 传统贴皮肤图片的方法是调用 APP 函数 XXXX_SetBitmap()方式实现,但非常非常麻烦.

②要想把控件显示出来,emWin 每种控件都有 1 个控件绘制函数,而把这个控件绘制函数改成我们自己编写的"自定义绘制函数",这样我们想把这个控件画成什么样都行,emWin 已经不参与这个控件的绘制工作了;在这个自定义绘制函数里面我们什么都不干,只显示出这个控件的图片,这就是用 skinning 方式实现位图皮肤.

③支持整幅图片贴图,不需要对整幅图片进行切片分解,可以非常快速做界面贴图.

如上图所示,(x0,y0)为整幅图片的显示坐标,(x1,y1)和(x2,y2)是 emWin 为控件自动生成的剪切显示坐标,我们只需要调用 GUI_DrawBitmap()函数在(x0,y0)显示整幅图片即可,emWin 会自动剪切显示出该控件的位图皮肤.

Skinning 方式实现位图皮肤的实施步骤:

1)在 WM_INIT_DIALOG 消息中(窗口初始化时程序跑到这里),将"页面图片结构体指针"保存于父窗体中:

```
//Bitmaps of windows in different states are assigned to "Window Picture Structures" (all members must be assigned values)
//将不同状态的页面位图赋值给"页面图片结构体"(所有成员都要赋值)
static WIN_BITMAP BMP_WindowDLG1; WIN_BITMAP *pBMP_WindowDLG1 = &BMP_WindowDLG1;
BMP_WindowDLG1.normal = &bmbitmap_n;
BMP_WindowDLG1.pressed = &bmbitmap_p;
BMP_WindowDLG1.focus = &bmbitmap_n;
BMP_WindowDLG1.disable = &bmbitmap_n;
BMP_WindowDLG1.thumb_n = &bmbitmap_n;
BMP_WindowDLG1.thumb_p = &bmbitmap_n;
WM_SetUserData(pMsg->hWin, &pBMP_WindowDLG1, sizeof(pBMP_WindowDLG1));//Write the pointer to the user data
```

2)在 WM_INIT_DIALOG 消息中(窗口初始化时程序跑到这里),把 BUTTON 控件的绘制函数改成自定义绘制函数:

```
hItem = WM_GetDialogItem(pMsg->hWin, ID_BUTTON_0);
BUTTON_SetSkin(hItem, SKIN_button3L);//设置 ID_BUTTON_0 控件使用位图皮肤 Set ID_BUTTON_0 to used bitmap skinning
```

备注:BUTTON 控件的自定义绘制函数 SKIN_button3L()是在 SKIN_button.c 文件中实现的,"页面图片结构体"是在 skin.h 中定义的.

5.总结:

以整幅图片实现 emWin 位图皮肤应该是尼奇光电业界首创的方法,可以大大减轻控件贴图的工作量,保守估计贴图工作效率至少提高 10 倍以上.如果把每个控件对应的图片先切片分解下来,然后再转换成 c 文件,最后再贴图到控件上面去,那样工作量将是非常恐怖的!