本集视频正在制作中，敬请期待…

第四十四集

基于 uC/OS-II 和 uC/GUI 的 UI 系统工程讲解

到了这里，我们之前说的三个方面，也就是我们的内置 IP 核，自定义 IP 核、操作系统及 UI 界面，这三个方面的内容就讲解完了，我们 A4 开发板上的外设都已经会使用了，屏幕也可以点亮了，如果现在，我们想要实现一个漂亮的图形界面系统，那么想必也是轻松加愉快，我们只需要把屏幕和外设相结合，然后再设计一个非常酷炫的界面，就能够实现和我们开发板开机所自带的 UI 系统一样的效果了，说是这样说，但是做起来，有的朋友也许就不知道如何下手了，为了能够让大家有直观的理解，我们就以A4开发板的开机测试界面为例，给大家去讲一讲我们到底是怎样完成这样一个 UI 系统的。

§ 1.33 功能概述

废话不多说，首先我们先来给大家讲解一下该工程主要实现了哪些功能，该工程主要实现了如图 1.325 所示内容。

图 1.325 测试板载功能

你没有看错，该工程主要就是设计出这样一个测试板载功能的 UI 界面，对于这个界面相信大家已经很熟了，它主要是用来测试我们按键、数码管、LED 等和蜂鸣器四个外设的。我们只要知道了该页面是如何设计的，至于其他页面，我们就可以依葫芦画瓢一一设计出来。

§ 1.34 硬件框架

讲完了功能概述，接下来我们就来讲解硬件框架，该工程的硬件框架，如图 1.326 所示。

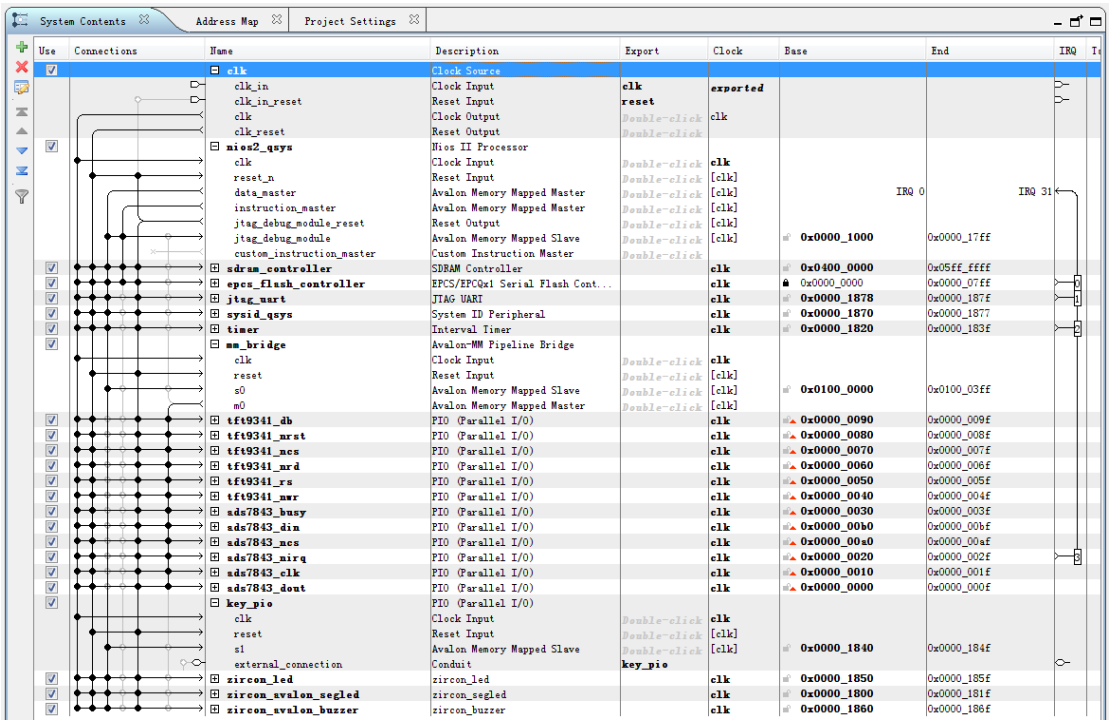

图 1.326 基于 uC/OS II 和 uC/GUI 的 UI 系统硬件框架图

从该图中我们可以看出，我们这里的硬件框架实验是在uC/GUI系统的硬件框架基础上多添加了 3 个 IP 核和 1 个 PIO，其中，3 个 IP 核分别对应着我们的 LED 外设，数码管外设个蜂鸣器外设，1 个 PIO 对应着我们的按键外设。当然，我们这里也可以不使用 IP 核，和按键一样使用 PIO 的方式来实现也是可以的，不过，如果我们使用的是 PIO IP 核实现的话，那么我们在软件编程中将要编写 LED、数码管和蜂鸣器的应用程序。如果我们使用 IP 核的话，那么我们之前已经编写的 IP 核寄存器头文件和 IP 核底层驱动文件就不需要再编写了，我们只需要将这些 IP 核添加至 Qsys 软件中，这些应用程序会就会自动添加到我们的 BSP 板级支持包中，我们在应用程序中只需要调用这些函数就可以使用了。现在我们能够发现，之前选择使用 IP 核是多么明智的选择。此时此刻，即使我们不说，想必大家也是都能明白的，因为这些 IP 核都是我们学过的。

§ 1.35 软件工程

讲完了硬件框架，接下来我们就来讲解软件工程，该实验的软件工程代码，如代码 1.99 所示。

代码 1.99 Qsys_Finish.c 代码

```
1  //--------------------------------------------------------------------------------
2  //-- 文件名    :  Qsys_Finish.c
3  //-- 描述      :  基于 uC/OS-II 和 uC/GUI 的 UI 系统
4  //-- 修订历史  :  2014-1-1
5  //-- 作者      :  Zircon Opto-Electronic Technology CO.,Ltd.
6  //--------------------------------------------------------------------------------
7  #include <stdio.h>
```

```c
#include "GUI.h"
#include "ZIRCON_UI.h"
#include "ADS7843.h"

//-------------------------------------------------------------------------------
//-- 名称     :  task1()
//-- 功能     :  图形界面程序
//-- 输入参数  :  无
//-- 输出参数  :  无
//-------------------------------------------------------------------------------
void task1(void* pdata)
{
   GUI_Init();
   ZIRCON_Test01();
}

//-------------------------------------------------------------------------------
//-- 名称     :  task2()
//-- 功能     :  触摸检测程序
//-- 输入参数  :  无
//-- 输出参数  :  无
//-------------------------------------------------------------------------------
void task2(void* pdata)
{
   ADS7843_Init();
   while (1)
   {
      OSTimeDlyHMSM(0, 0, 0, 5);
      GUI_TOUCH_Exec();
   }
}

//-------------------------------------------------------------------------------
//-- 名称     :  main()
//-- 功能     :  程序入口
//-- 输入参数  :  无
//-- 输出参数  :  无
//-------------------------------------------------------------------------------
int main(void)
{

  OSTaskCreateExt(task1,
                  NULL,
                  (void *)&task1_stk[TASK_STACKSIZE-1],
```

```
                TASK1_PRIORITY,
                TASK1_PRIORITY,
                task1_stk,
                TASK_STACKSIZE,
                NULL,
                0);


  OSTaskCreateExt(task2,
                NULL,
                (void *)&task2_stk[TASK_STACKSIZE-1],
                TASK2_PRIORITY,
                TASK2_PRIORITY,
                task2_stk,
                TASK_STACKSIZE,
                NULL,
                0);
  OSStart();
  return 0;
}
```

从该代码中我们可以看出，代码第 21 行是最关键的一行，也是我们之前没有见过的。下面我们就来看下第 21 行代码究竟实现了怎样的功能，该函数代码 1.100 所示。

代码 1.100 ZIRCON_Test01 代码

```
//-------------------------------------------------------------------------
//-- 名称        : ZIRCON_Test01()
//-- 功能        : 图形主界面程序
//-- 输入参数    : 无
//-- 输出参数    : 无
//-------------------------------------------------------------------------
void ZIRCON_Test01()
{
  /* 显示一张图片 ,该代码主要是利用了 ILI9341 底层驱动来实现的 */
  ILI9341_DisplayPic(Test_01);
  /* 在所有窗口上自动使用存储设备 */
  WM_SetCreateFlags(WM_CF_MEMDEV);
  /* 声明 7 个按钮 */
  BUTTON_Handle Test_01_BUTTON[7];
  /* 创建底部菜单栏左键 */
  Test_01_BUTTON[0] = BUTTON_Create(0  , 270, 80 , 50, GUI_ID_BUTTON0, WM_CF_SHOW);
  BUTTON_SetBitmap(Test_01_BUTTON[0], 0, &bmMenu_Left);
  /* 创建底部菜单栏中键 */
  Test_01_BUTTON[1] = BUTTON_Create(80 , 270, 80 , 50, GUI_ID_BUTTON1, WM_CF_SHOW);
  BUTTON_SetBitmap(Test_01_BUTTON[1], 0, &bmMenu_Home);
```

```
    /* 创建底部菜单栏右键 */
    Test_01_BUTTON[2] = BUTTON_Create(160, 270, 80 , 50, GUI_ID_BUTTON2, WM_CF_SHOW);
    BUTTON_SetBitmap(Test_01_BUTTON[2], 0, &bmMenu_Right);
    /* 创建按键菜单栏 */
    Test_01_BUTTON[3] = BUTTON_Create(0  , 51 , 240, 50, GUI_ID_BUTTON3, WM_CF_SHOW);
    BUTTON_SetBitmap(Test_01_BUTTON[3], 0, &bmTest_01_Key);
    /* 创建数码管菜单栏 */
    Test_01_BUTTON[4] = BUTTON_Create(0  , 100, 240, 50, GUI_ID_BUTTON4, WM_CF_SHOW);
    BUTTON_SetBitmap(Test_01_BUTTON[4], 0, &bmTest_01_DigitalLed);
    /* 创建 LED 菜单栏 */
    Test_01_BUTTON[5] = BUTTON_Create(0  , 150, 240, 50, GUI_ID_BUTTON5, WM_CF_SHOW);
    BUTTON_SetBitmap(Test_01_BUTTON[5], 0, &bmTest_01_Led);
    /* 创建蜂鸣器菜单栏 */
    Test_01_BUTTON[6] = BUTTON_Create(0  , 201, 240, 50, GUI_ID_BUTTON6, WM_CF_SHOW);
    BUTTON_SetBitmap(Test_01_BUTTON[6], 0, &bmTest_01_Beep);
    /* 用来去掉按钮的焦点 */
    BUTTON_SetFocussable(Test_01_BUTTON[0],0);    /* 用来去掉底部菜单栏左键的焦点  */
    BUTTON_SetFocussable(Test_01_BUTTON[1],0);    /* 用来去掉底部菜单栏中键的焦点  */
    BUTTON_SetFocussable(Test_01_BUTTON[2],0);    /* 用来去掉底部菜单栏右键的焦点  */
    BUTTON_SetFocussable(Test_01_BUTTON[3],0);    /* 用来去掉按键菜单栏的焦点      */
    BUTTON_SetFocussable(Test_01_BUTTON[4],0);    /* 用来去掉数码管菜单栏 的焦点   */
    BUTTON_SetFocussable(Test_01_BUTTON[5],0);    /* 用来去掉 LED 菜单栏的焦点     */
    BUTTON_SetFocussable(Test_01_BUTTON[6],0);    /* 用来去掉蜂鸣器菜单栏的焦点    */
    /* 判断按钮按下并执行相应的程序 */
    switch(GUI_WaitKey())
    {  /* 如果按下底部菜单栏左键,将执行下列程序  */
       case GUI_ID_BUTTON0 : {     BUTTON_Delete(Test_01_BUTTON[0]);     /* 删掉底部菜单栏左键    */
                                   BUTTON_Delete(Test_01_BUTTON[1]);     /* 删掉底部菜单栏中键    */
                                   BUTTON_Delete(Test_01_BUTTON[2]);     /* 删掉底部菜单栏右键    */
                                   BUTTON_Delete(Test_01_BUTTON[3]);     /* 删掉按键菜单栏        */
                                   BUTTON_Delete(Test_01_BUTTON[4]);     /* 删掉数码管菜单栏      */
                                   BUTTON_Delete(Test_01_BUTTON[5]);     /* 删掉 LED 菜单栏       */
                                   BUTTON_Delete(Test_01_BUTTON[6]);     /* 删掉蜂鸣器菜单栏      */
                                   ZIRCON_Test01();break;}               /* 进入测试主页面        */
       /* 如果按下底部菜单栏中键,将执行下列程序  */
       case GUI_ID_BUTTON1 : {     BUTTON_Delete(Test_01_BUTTON[0]);     /* 删掉底部菜单栏左键    */
                                   BUTTON_Delete(Test_01_BUTTON[1]);     /* 删掉底部菜单栏中键    */
                                   BUTTON_Delete(Test_01_BUTTON[2]);     /* 删掉底部菜单栏右键    */
                                   BUTTON_Delete(Test_01_BUTTON[3]);     /* 删掉按键菜单栏        */
                                   BUTTON_Delete(Test_01_BUTTON[4]);     /* 删掉数码管菜单栏      */
                                   BUTTON_Delete(Test_01_BUTTON[5]);     /* 删掉 LED 菜单栏       */
                                   BUTTON_Delete(Test_01_BUTTON[6]);     /* 删掉蜂鸣器菜单栏      */
                                   ZIRCON_Test01();break;}               /* 进入测试主页面        */
       /* 如果按下底部菜单栏右键,将执行下列程序  */
```

```
65      case GUI_ID_BUTTON2 : {     BUTTON_Delete(Test_01_BUTTON[0]);      /* 删掉底部菜单栏左键   */
66                                  BUTTON_Delete(Test_01_BUTTON[1]);     /* 删掉底部菜单栏中键    */
67                                  BUTTON_Delete(Test_01_BUTTON[2]);     /* 删掉底部菜单栏右键   */
68                                  BUTTON_Delete(Test_01_BUTTON[3]);     /* 删掉按键菜单栏       */
69                                  BUTTON_Delete(Test_01_BUTTON[4]);     /* 删掉数码管菜单栏     */
70                                  BUTTON_Delete(Test_01_BUTTON[5]);     /* 删掉 LED 菜单栏      */
71                                  BUTTON_Delete(Test_01_BUTTON[6]);     /* 删掉蜂鸣器菜单栏     */
72                                  ZIRCON_Test01();break;}               /* 进入测试主页面       */
73      /* 如果按下按键菜单栏,将执行下列程序 */
74      case GUI_ID_BUTTON3 : {     BUTTON_Delete(Test_01_BUTTON[0]);     /* 删掉底部菜单栏左键   */
75                                  BUTTON_Delete(Test_01_BUTTON[1]);     /* 删掉底部菜单栏中键   */
76                                  BUTTON_Delete(Test_01_BUTTON[2]);     /* 删掉底部菜单栏右键   */
77                                  BUTTON_Delete(Test_01_BUTTON[3]);     /* 删掉按键菜单栏       */
78                                  BUTTON_Delete(Test_01_BUTTON[4]);     /* 删掉数码管菜单栏     */
79                                  BUTTON_Delete(Test_01_BUTTON[5]);     /* 删掉 LED 菜单栏      */
80                                  BUTTON_Delete(Test_01_BUTTON[6]);     /* 删掉蜂鸣器菜单栏     */
81                                  Test01_Key_Home();break;}             /* 进入按键测试页面     */
82      /* 如果按下数码管菜单栏,将执行下列程序 */
83      case GUI_ID_BUTTON4 : {     BUTTON_Delete(Test_01_BUTTON[0]);     /* 删掉底部菜单栏左键   */
84                                  BUTTON_Delete(Test_01_BUTTON[1]);     /* 删掉底部菜单栏中键   */
85                                  BUTTON_Delete(Test_01_BUTTON[2]);     /* 删掉底部菜单栏右键   */
86                                  BUTTON_Delete(Test_01_BUTTON[3]);     /* 删掉按键菜单栏       */
87                                  BUTTON_Delete(Test_01_BUTTON[4]);     /* 删掉数码管菜单栏     */
88                                  BUTTON_Delete(Test_01_BUTTON[5]);     /* 删掉 LED 菜单栏      */
89                                  BUTTON_Delete(Test_01_BUTTON[6]);     /* 删掉蜂鸣器菜单栏     */
90                                  Test01_Segled_Home();break;}          /* 进入数码管测试页面   */
91      /* 如果按下 LED 菜单栏,将执行下列程序 */
92      case GUI_ID_BUTTON5 : {     BUTTON_Delete(Test_01_BUTTON[0]);     /* 删掉底部菜单栏左键   */
93                                  BUTTON_Delete(Test_01_BUTTON[1]);     /* 删掉底部菜单栏中键   */
94                                  BUTTON_Delete(Test_01_BUTTON[2]);     /* 删掉底部菜单栏右键   */
95                                  BUTTON_Delete(Test_01_BUTTON[3]);     /* 删掉按键菜单栏       */
96                                  BUTTON_Delete(Test_01_BUTTON[4]);     /* 删掉数码管菜单栏     */
97                                  BUTTON_Delete(Test_01_BUTTON[5]);     /* 删掉 LED 菜单栏      */
98                                  BUTTON_Delete(Test_01_BUTTON[6]);     /* 删掉蜂鸣器菜单栏     */
99                                  Test01_Led_Home();break;}             /* 进入 LED 测试页面    */
100     /* 如果按下蜂鸣器菜单栏,将执行下列程序 */
101     case GUI_ID_BUTTON6 : {     BUTTON_Delete(Test_01_BUTTON[0]);     /* 删掉底部菜单栏左键   */
102                                 BUTTON_Delete(Test_01_BUTTON[1]);     /* 删掉底部菜单栏中键   */
103                                 BUTTON_Delete(Test_01_BUTTON[2]);     /* 删掉底部菜单栏右键   */
104                                 BUTTON_Delete(Test_01_BUTTON[3]);     /* 删掉按键菜单栏       */
105                                 BUTTON_Delete(Test_01_BUTTON[4]);     /* 删掉数码管菜单栏     */
106                                 BUTTON_Delete(Test_01_BUTTON[5]);     /* 删掉 LED 菜单栏      */
107                                 BUTTON_Delete(Test_01_BUTTON[6]);     /* 删掉蜂鸣器菜单栏     */
108                                 Test01_Beep_Home();break;}            /* 进入蜂鸣器测试页面   */
```

```
109     }
120 }
```

该代码注释比较详细，代码比较简单，这里我们就不对代码进行介绍了，下面我们主要来讲解一下该代码所实现的效果，这里我们就以按键菜单为例，如图 1.327 所示。

图 1.327 ZIRCON_Test01 函数实现的效果图

数码管菜单、LED 灯菜单、蜂鸣器菜单，以及底部菜单栏三个按键也都是利用这种方法实现的，当然了实现这种效果的方法不止一种，我们还使用触摸，判断位置来达到同样的效果。说完了程序，接下来我们再来说一说，图片是如何转换成数组的。这里我们使用了两种工具，第一种工具是 Image2Lcd，该工具可以将.bmp 和.jpg 格式的图片直接转换称.c 文件，然后我们再将.c文件修改成.h文件添加至Eclipse软件工程中就可以直接使用了，该工具转换的图片我们主要是利用 ILI9341 底层驱动直接进行显示图片。第二种工具是 uCGUI 自带的位图转换器（uC-GUI-BitmapConvert），这个工具我们可以在 uC/GUI 的源码中找到，关于这个工具的使用，uC/GUI的中文手册第八章中有详细的讲解，我们这里就不在进一步说明了。该工具转换的图片我们主要用在了按钮上。最后我们再来补充说明两点，第一点就是：如果你的按钮制作出来不是平面的按钮，而是一个 3D 按钮，那么你需要在 uCGUI 文件夹下的 Widget 中的 BUTTON.c 中修改如下：

```
1 /* Support for 3D effects */
2 #ifndef   BUTTON_USE_3D
3   #define BUTTON_USE_3D 0  //将 1 改为 0 关闭 3D 按钮显示
4 #endif
```

第二点就是：当你进入测试页面以后，如果你发现图片切换会有黑白交替闪烁效果，那么你需要在 uCGUI 下的 GUI_X 文件夹中的 GUI_X_uCOS.C 中修改如下：

```
1 void GUI_X_ExecIdle (void)
2 {
```

```
//    OS_X_Delay(1);
   OSTimeDly(5);
}
```