

数字信号处理实验指导书

实验二 键盘输入液晶屏显示实验

一、实验目的

- 1.通过实验学习使用 F28335 DSP 的扩展 I/O 端口控制外围设备的方法，了解液晶显示器的显示控制原理及编程方法。
2. 通过实验学习使用 F28335 DSP 的扩展端口接收外围设备信息的方法，了解键盘的使用原理及编程方法。

二、实验内容

- 1、基本内容：通过在 CCS 的编程，在液晶显示屏上显示“ICETEK-F28335-AF 液晶显示”和计时时钟，精确到秒，形式为“时时：分分：秒秒”。5 分
- 2、基本内容：获取键值，并显示键值。3 分
- 3、拓展内容：结合键盘和显示，自主设计，完成某一功能。根据难易程度及特殊性。可获得 1~2 分

三、实验设备

计算机， ICETEK-F28335-AF 实验箱。

四、实验准备

ICETEK-DSP 教学实验箱的硬件连接，同实验一

五、实验原理

1.扩展 IO 接口：

ICETEK-F28335-AF 是一块以 TMS320F28335DSP 为核心的 DSP 扩展评估板，它通过扩展接口与实验箱的显示/控制模块连接，可以控制其各种外围设备。

2. 显示控制方法：

本实验中使用已写好的库函数对液晶屏幕进行操作。需要在工程文件中加入库 ICETEK-CTR.lib 以及头文件 ICETEK-CTR.h。

下面给出 ICETEK-CTR.lib 的控制液晶屏幕的接口函数及其功能描述：

```
void ICETEKCTR_InitLCD(); //初始化液晶显示屏
```

```
void ICETEKCTR_LCDCMD(Byte dbCommand); //向 LCD 发送指令
void ICETEKCTR_LCDDAT(Byte cData); //向 LCD 发送数据
void ICETEKCTR_LCDCLS(); //LCD 清屏
void ICETEKCTR_LCDPutString(char *sString,int x,int y); //在 LCD 屏幕上显示字符串

void ICETEKCTR_LCDDrawPixel(int x,int y,Byte cColor); //写点到屏幕，输入参数坐标值和颜色，颜色 0 消点,1 画点,2 异或画点
```

3. **键盘连接原理：**键盘的扫描码由 DSP 的扩展地址 0x208001 给出，当有键盘输入时，读此端口得到扫描码，当无键被按下时读此端口的结果为 0。这功能由 ICETEKCTR_GetKey()实现，函数返回值就是键盘的扫描码。9 个按键分别回读回 1-9 这九个数字。

4. 实验程序流程图

在实验报告中给出流程。

六、实验步骤

- 1、启动 CCS
- 2、建立目标配置文件（若已建立则可跳过此步骤）
- 3、创建工程（详见指示灯实验）
- 4、修改 main.c 的内容为所编写代码，点击 File->Save 保存

（1）液晶显示

```
#include "DSP2833x_Device.h"
#include "DSP2833x_Examples.h"
#include "ICETEK-CTR.h"

void InitICETEK28335Ae(); //初始化：DSP主频、GPIO、中断向量、中断使能
void main() { int bSuccess,h,m,s; char buffer[10]={ "00:00:00" };
    InitICETEK28335Ae();
    bSuccess=ICETEKCTR_InitCTR(ICETEKCTRModeTeachingResearch); //初始化
    ICETEK-CTR: 教研模式
    while ( bSuccess ); // 如果初始化ICETEK-CTR错误，停止运行，可观察bSuccess取值查找初始化失败原因
    ICETEKCTR_LCDPutString("ICETEK-F28335-AF",0,LCDLINE0);
    ICETEKCTR_LCDPutString("液晶显示",2,LCDLINE1);
```

```

h=m=s=0;
for(;;){
    s++; if ( s>59 ) { s=0; m++; if ( m>59 ) { m=0; h++; h%=24; } }
    buffer[0]=h/10+'0'; buffer[1]=h%10+'0';
    buffer[3]=m/10+'0'; buffer[4]=m%10+'0';
    buffer[6]=s/10+'0'; buffer[7]=s%10+'0';
    ICETEKCTR_LCDPutString(buffer,2,LCDLINE3);
    ICETEKCTR_Delays(1000);
}
}

void InitICETEKF28335Ae() {
    // Step 1. Initialize System Control:
    // PLL, WatchDog, enable Peripheral Clocks
    // This example function is found in the DSP2833x_SysCtrl.c file.
    InitSysCtrl();

    // Step 2. Initialize GPIO:
    // This example function is found in the DSP2833x_Gpio.c file and
    // illustrates how to set the GPIO to it's default state.
    // InitGpio(); Skipped for this example
    InitXintf16Gpio(); //初始化扩展空间接口管脚，以便与ICETEK-CTR通讯

    // Step 3. Clear all interrupts and initialize PIE vector table:
    // Disable CPU interrupts
    DINT;

    // Initialize PIE control registers to their default state.
    // The default state is all PIE interrupts disabled and flags
    // are cleared.
    // This function is found in the DSP2833x_PieCtrl.c file.
    InitPieCtrl();

    // Disable CPU interrupts and clear all CPU interrupt flags:
    IER = 0x0000;
    IFR = 0x0000;

    // Initialize the PIE vector table with pointers to the shell Interrupt
    // Service Routines (ISR).
    // This will populate the entire table, even if the interrupt
    // is not used in this example. This is useful for debug purposes.
    // The shell ISR routines are found in DSP2833x_DefaultIsr.c.
    // This function is found in DSP2833x_PieVect.c.
    InitPieVectTable();
}

```

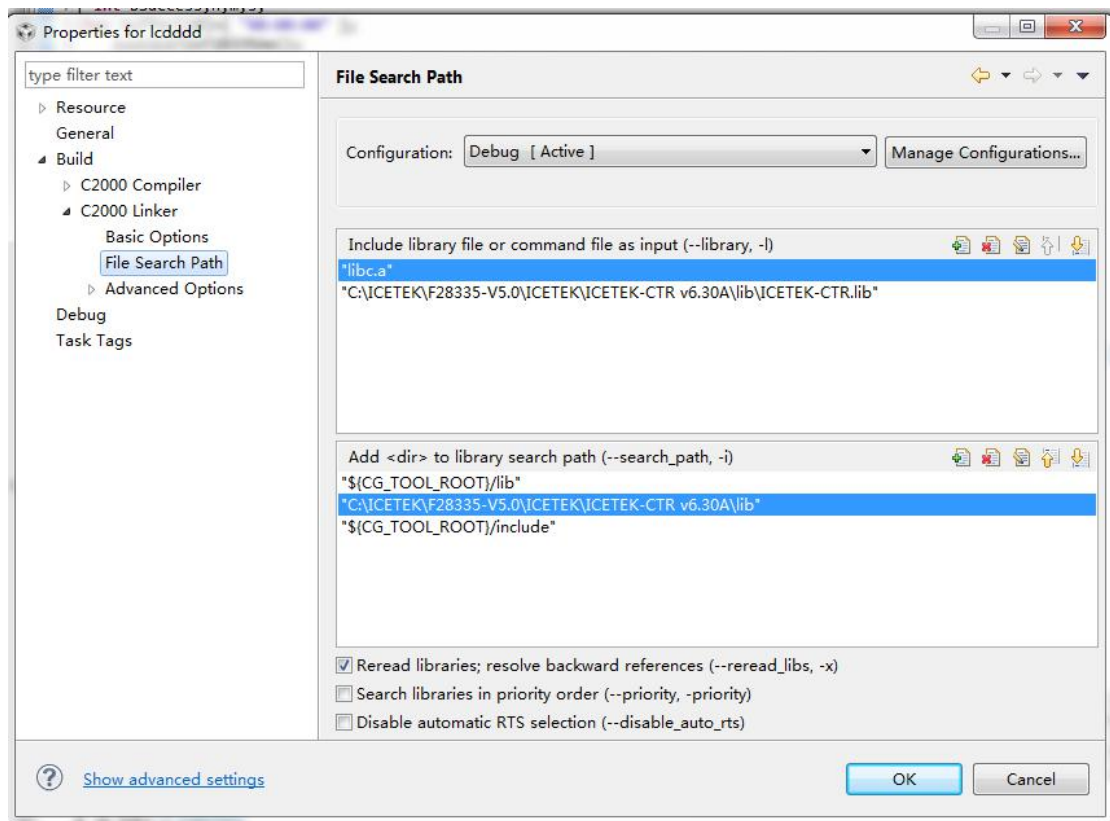
```
}
```

(2) 键盘

```
void main() { int bSuccess; unsigned int uKeyCode,uDelayCount;  
    InitICETEKv28335Ae();  
    bSuccess=ICETEKCTR_InitCTR(ICETEKCTRModeTeachingResearch); //初始化  
    ICETEK-CTR: 教研模式  
    while ( bSuccess ); // 如果初始化ICETEK-CTR错误, 停止运行, 可观察bSuccess取  
    值查找初始化失败原因  
    uKeyCode=ICETEKCTR_GetKey();  
}
```

5、添加头文件（添加方法见实验一）


6、添加函数库如下图



7、连接配置文件

右侧 Target Configuration 栏中选择之前设置好的硬件配置文件，右键点击
->Link File To Project->CProgram（或任何用户自定义的名称）。

8、点击按钮 ，CCS 会自动编译、连接和下载程序

9、点击菜单 Run->Resume，运行程序，或者直接点击  按钮，观察 LED 指示灯显示情况

10、退出 CCS

七、实验报告

参照自动控制原理的实验报告模板。也可以自由发挥。