**《嵌入式系统》课程**

**作业一：**

**GPIO实验——用按键控制LED**

姓名： 阮玉斌

学号： 2019112043

班级： 自动化3班

一、任务要求：

1、掌握GPIO引脚原理、功能使用和相应寄存器配置；

2、实现如下功能：

① 按键1，点亮三盏灯；

② 按键2，熄灭三盏灯；

③ 按键3，按照123、321的顺序依次点亮三盏灯。

二、思路：

1、通过开发板电路图可知，S3C2440有130个I/O口，其中GPG0、GPG2、GPG3、GPG11依次控制独立按键S2-S5，本实验要求使用三个独立按键实现控制功能，则配置S2-S4相应GPFCON寄存器将GPG0、GPG2、GPG3设为输入功能，的通过读出GPFDAT/GPGDAT寄存器判断相应位是0还是1判断按键是否按下，按键按下，相应引脚电平为低电平，寄存器中相应位为0，否则为1；

2、编写子函数key( ) 用于实现独立按键扫描功能，扫描到相应按键的寄存器位为0，则将按键标志设置成相应的数，其中通过延时子程序实现按键的去抖动功能；

3、通过查询按键标志判断按下哪个按键，在相应的条件语句中执行LED的相应操作，其中由于人眼的视觉滞留效应，按键3对应的依次点亮灯的功能要使用wait( ) 子程序延时大于20ms以上时间，否则无法被人眼捕捉到；

4、相应LED的点亮和熄灭也是通过对相应寄存器的相应位通过逻辑运算进行的，使用前需要先配置相应的寄存器设置。

5、在无穷循环中，通过调用key(keyflag) 函数进行按键扫描，从中判断出S2、S3、S4是否按下，若按下则执行相应按键对应的LED操作；若无按键按下，则重复执行上一次按下按键对应的LED操作，直至有按键按下。复位初始化时，按键标志位为0，LED全灭。

三、步骤：（含代码分析、调试过程分析及验证过程图片）

1、源代码分析

key\_led.c程序的源代码和注释分析如下：

#define GPFCON (\*(volatile unsigned long \*)0x56000050)

#define GPFDAT (\*(volatile unsigned long \*)0x56000054)

#define GPGCON (\*(volatile unsigned long \*)0x56000060)

#define GPGDAT (\*(volatile unsigned long \*)0x56000064)

/\* LED1,LED2,LED4对应GPF4、GPF5、GPF6 \*/

#define GPF4\_out (1<<(4\*2))

#define GPF5\_out (1<<(5\*2))

#define GPF6\_out (1<<(6\*2))

#define GPF4\_msk (3<<(4\*2))

#define GPF5\_msk (3<<(5\*2))

#define GPF6\_msk (3<<(6\*2))

/\* 因s1是对应复位按键，因此本案例使用s2-s4进行控制

\* S2,S3,S4对应GPF0、GPF2、GPG3 \*/

#define GPF0\_in (0<<(0\*2))

#define GPF2\_in (0<<(2\*2))

#define GPG3\_in (0<<(3\*2))

#define GPF0\_msk (3<<(0\*2))

#define GPF2\_msk (3<<(2\*2))

#define GPG3\_msk (3<<(3\*2))

/\* 延时子程序和按键扫描子程序 \*/

void wait(volatile unsigned long dly);

int key(int flag);

int main()

{

int keyflag = 0; //按键标志

// LED1,LED2,LED4对应的3根引脚设为输出

GPFCON &= ~(GPF4\_msk | GPF5\_msk | GPF6\_msk);

GPFCON |= GPF4\_out | GPF5\_out | GPF6\_out;

// S2,S3对应的2根引脚设为输入

GPFCON &= ~(GPF0\_msk | GPF2\_msk);

GPFCON |= GPF0\_in | GPF2\_in;

// S4对应的引脚设为输入

GPGCON &= ~GPG3\_msk;

GPGCON |= GPG3\_in;

while(1)

{

keyflag = key(keyflag);

// 若按键S\_n按下，则相应标志keyflag表示按键编号n

// 然后执行相应LED操作

if(keyflag==0) //按键未按下

GPFDAT |= ((1<<4)|(1<<5)|(1<<6)); //全灭

if(keyflag==2) //按下按键S2

{

GPFDAT &= ((~(1<<4))&(~(1<<5))&(~(1<<6)));

} //实现3盏灯全亮

if(keyflag==3) //按下按键S3

{

GPFDAT |= ((1<<4)|(1<<5)|(1<<6));

} //实现三盏灯全灭

if(keyflag==4) //按下按键S4

{

GPFDAT |= ((1<<6)|(1<<5)|(1<<4));

wait(30000); //全灭

GPFDAT &= ~(1<<4); // LED1点亮

wait(30000);

GPFDAT &= ~(1<<5); // LED2点亮

wait(30000);

GPFDAT &= ~(1<<6); // LED3点亮

wait(30000);

GPFDAT |= ((1<<6)|(1<<5)|(1<<4));

wait(30000); //全灭

GPFDAT &= ~(1<<6); // LED3点亮

wait(30000);

GPFDAT &= ~(1<<5); // LED2点亮

wait(30000);

GPFDAT &= ~(1<<4); // LED1点亮

wait(30000);

}

wait(30000);

}

return 0;

}

/\*--------------延时程序---------------\*/

void wait(volatile unsigned long dly)

{

for(; dly > 0; dly--);

}

/\*----------------按键扫描程序--------------------\*/

/\* 功能：含去抖在内进行按键扫描 \*/

int key(int flag)

{

if((GPFDAT & (1<<0))==0) // S2按下

{

wait(200);

if((GPFDAT & (1<<0))==0) //去抖动，下同

{

return 2;

}

}

if((GPFDAT & (1<<2))==0) // S3按下

{

wait(200);

if((GPFDAT & (1<<2))==0)

{

return 3;

}

}

if((GPGDAT & (1<<3))==0) // S4按下

{

wait(200);

if((GPGDAT & (1<<3))==0)

{

return 4;

}

}

else

return flag; //当没有按键按下时，保留之前按键状态

}

2、调试过程分析

（1）将key\_led.c和启动文件代码crt0.S以及Makefile指令文件放在同一目录下，其中crt0.S和Makefile分别如下图3-1和3-2所示：

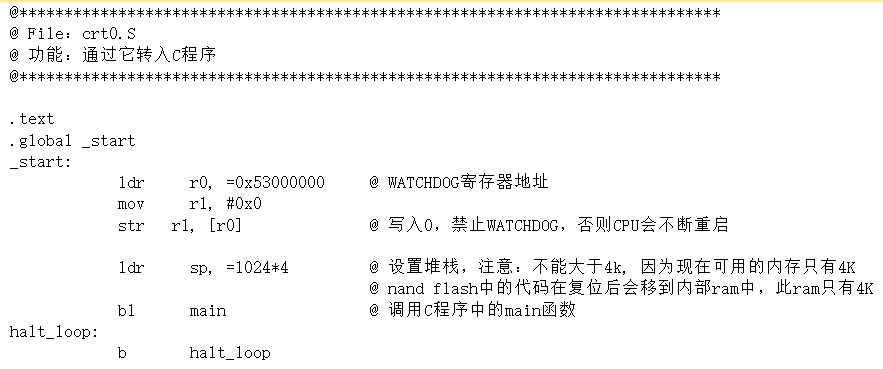


图3-1 crt0.S源程序

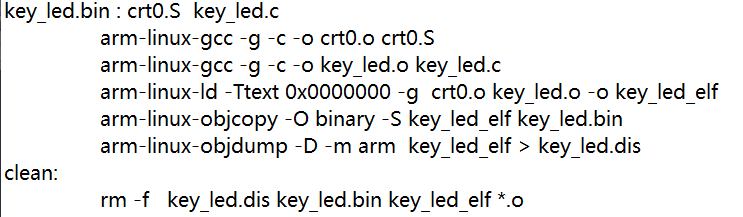


图3-2 Makefile指令

（2）通过Filezilla上传整个文件夹到ubuntu虚拟机，通过命令行进入到对应的文件目录下，执行make指令，make指令中第2、3行分别编译源程序crt0.S和key\_led.c（还没有连接），第4行将编译得到结果连接起来，第5行得到对应的二进制.bin格式文件，第6行将结果转换为汇编码查看，编译结果如下图3-3所示：

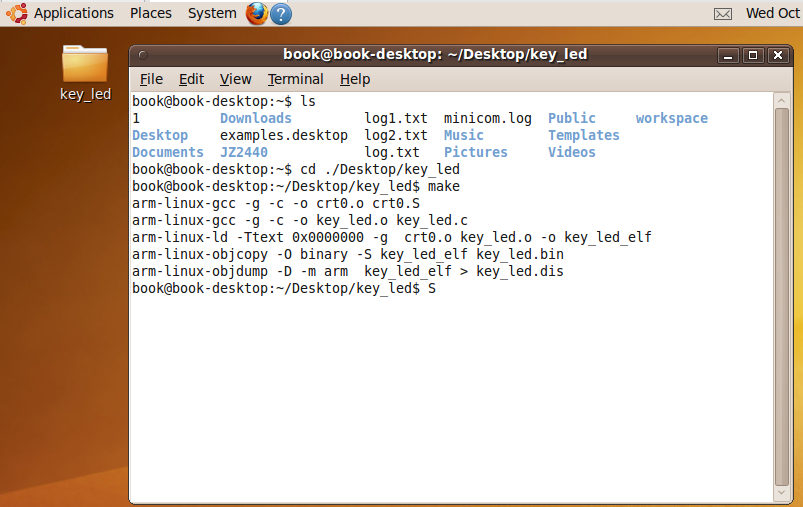


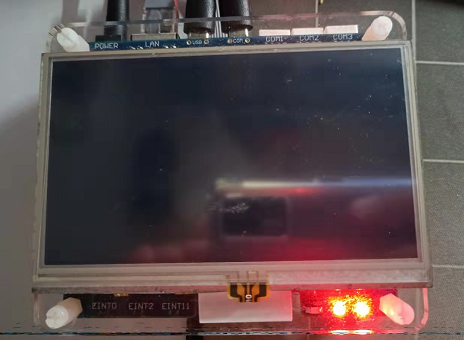
图3-3 调试编译结果

3、验证过程

将编译后转换生成的key\_led.bin文件烧写到裸板后，结果图如下：

图3-4 K2按下全亮 图3-5 K3按下全灭



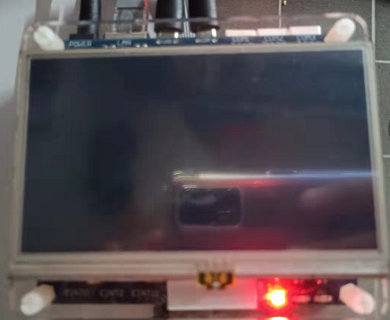


图3-5 K4按下LED走马灯点亮效果

实验结果满足要求。

四、总结：

本实验通过GPIO实现了独立按键控制不同LED点亮效果，实验结果满足实验要求。实验过程中掌握了根据原理图和寄存器进行寄存器配置，通过位操作读取按键对应Key的值，根据值设置引脚，同时熟练掌握了主机开发流程。