实验二:GPIO 输入/输出实验

1. 实验目的

- 了解 Mini2440 ARM 实验板的功能和使用。
- 掌握 J-link 仿真器的用法，并能连接实验板调试程序。
- 掌握 GPIO 外设的操作原理和编程。

2. 实验设备

- 硬件：PC 机一台
  Mini2440 ARM 实验板一套
  J-link 仿真器一套
- 软件：WindowsXP 系统，Keil uVision 4.0 集成开发环境

3. 实验内容

（1）使用 GPIO 控制 Mini2440 实验板上的 LED 指示灯的亮/灭，使用 Keil uVision 的调试功能单步、全速运行程序，设置断点，打开寄存器窗口监视寄存器，观察实验板上的 LED 指示灯的状态。

（2）使用 GPIO 读取 Mini2440 实验板上的按键状态，观察按键输入的抖动现象。

4. 实验预习要求

(1)学习 GPIO 外设的操作原理和编程方法;
(2)查阅 JTAG 的介绍，了解使用仿真器联机调试的原理。

5. 实验步骤

(1)认识 Mini2440 ARM 实验板。见图 2-1。

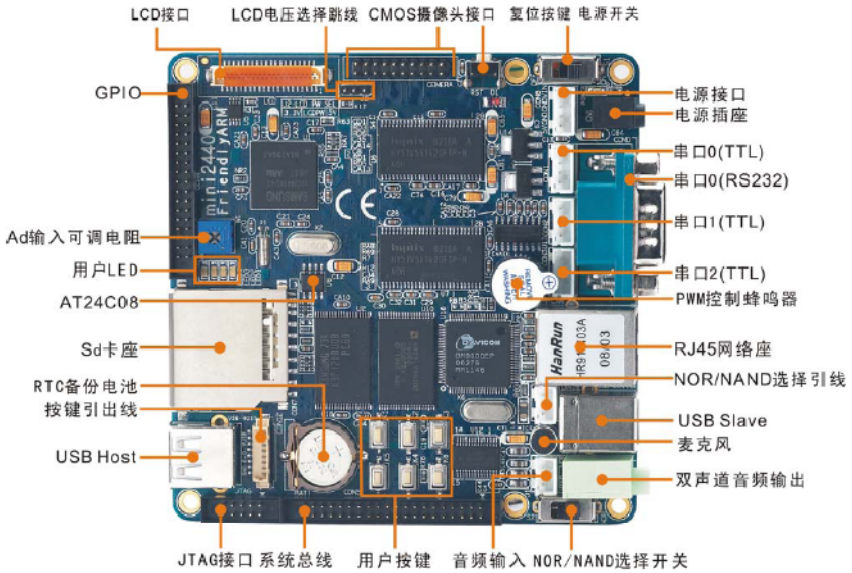

图 2-1 实验板及主要功能模块

图 2-2 电源接口与电源开关

LED 是开发中最常用的状态指示设备，本开发板具有4个用户可编程LED，它们直接与CPU 的GPIO 相连接，**低电平有效(点亮)**，详细电路连接关系见图2-3、图2-4。

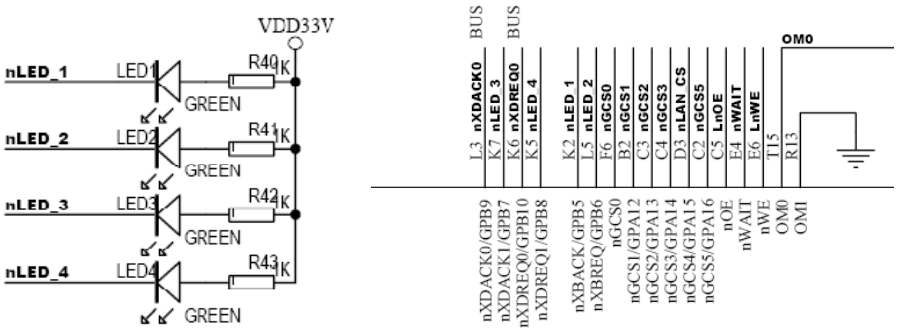

图 2-3 LED 指示灯

开发板总共有6个用户测试用按键，它们均从CPU 中断引脚直接引出，低电平触发，这些引脚也可以复用为GPIO和特殊功能口。

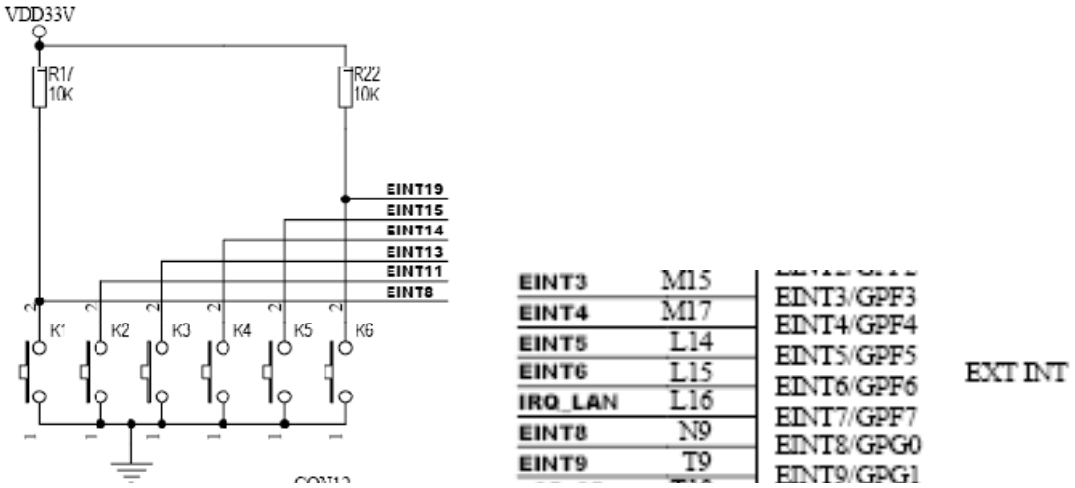

图 2-4 按键输入

JTAG接口是开发中最重要的接口，用途是调试，市面上常见的JLINK、ULINK，以及其他的仿真调试器，最终都是通过JTAG 接口连接的。标准的JTAG 接口是4 线：TMS、 TCK、TDI、TDO，分别为模式选择、时钟、数据输入和数据输出线，加上电源和地，一般总共6 条线就够了；为了方便调试，大部分仿真器还提供了一个复位信号。见图2-5。

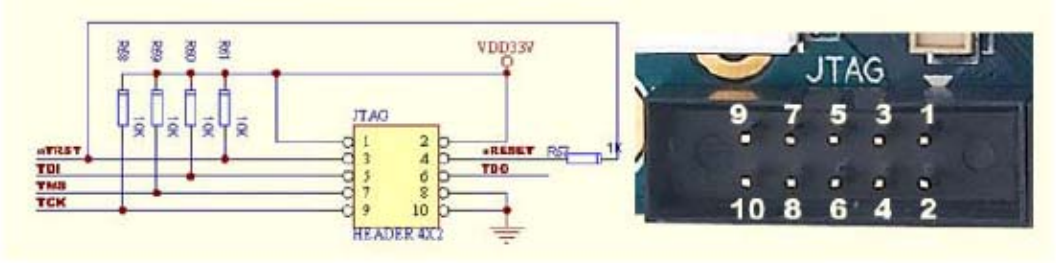

图 2-5 JTAG 接口

(2)启动 Keil uVision，新建一个工程 ex02-1。不需要系统提供的 Startup 文件。建立汇编源文件 ex02-1.s，编写实验程序，然后添加到工程中。

(3)设置工程选项，存储器映射。见图 2-6。

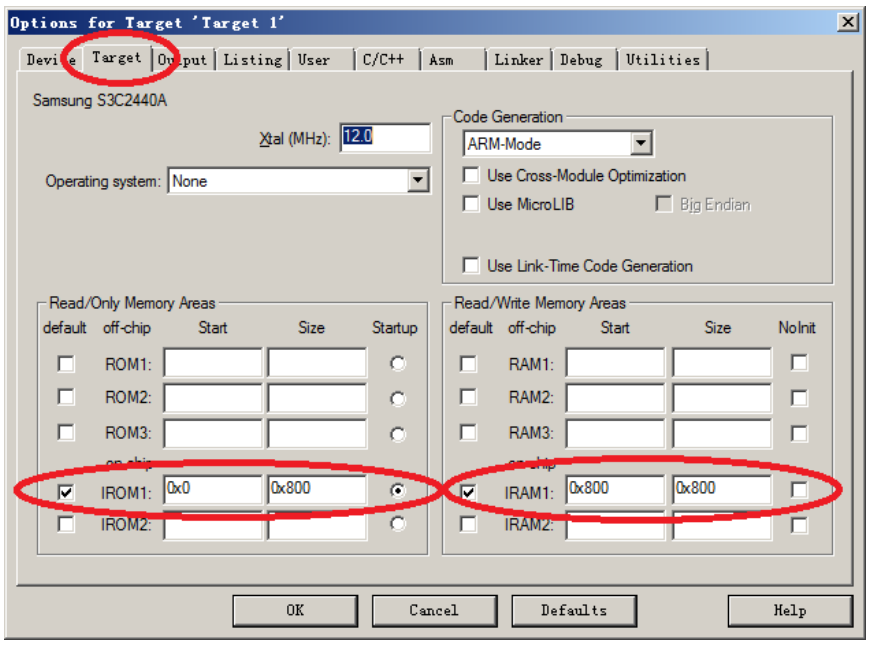

图 2-6 设置存储器映射

(4) 设置工程调试选项。见图 2-7。建立仿真初始化文件 RAM.ini，内容如下：

```
_WDWORD(0x53000000, 0x0);   //disable watch dog
LOAD %L INCREMENTAL
```

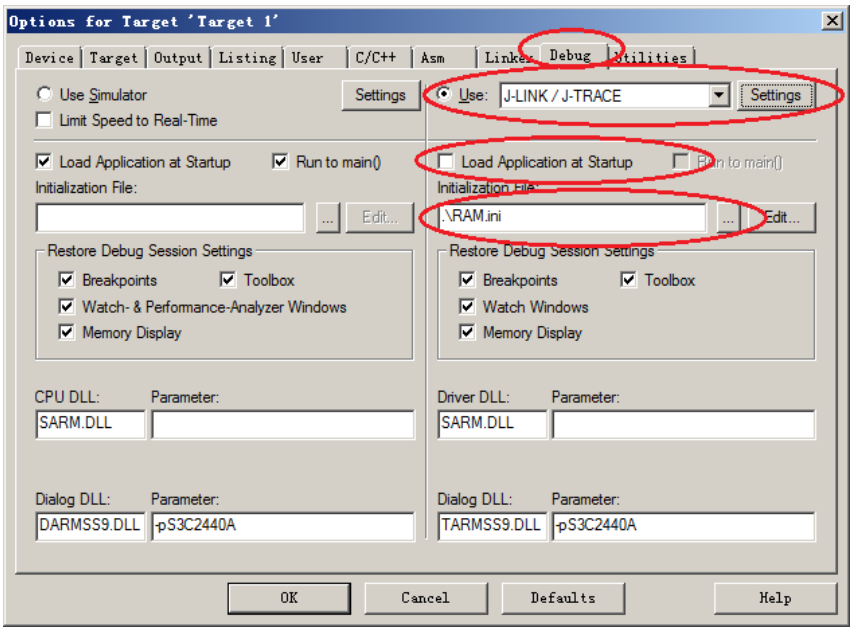

图 2-7 设置仿真调试选项

(5)编译链接工程。连接实验板电源、J-link 仿真器，进行仿真调试。见图 2-7。

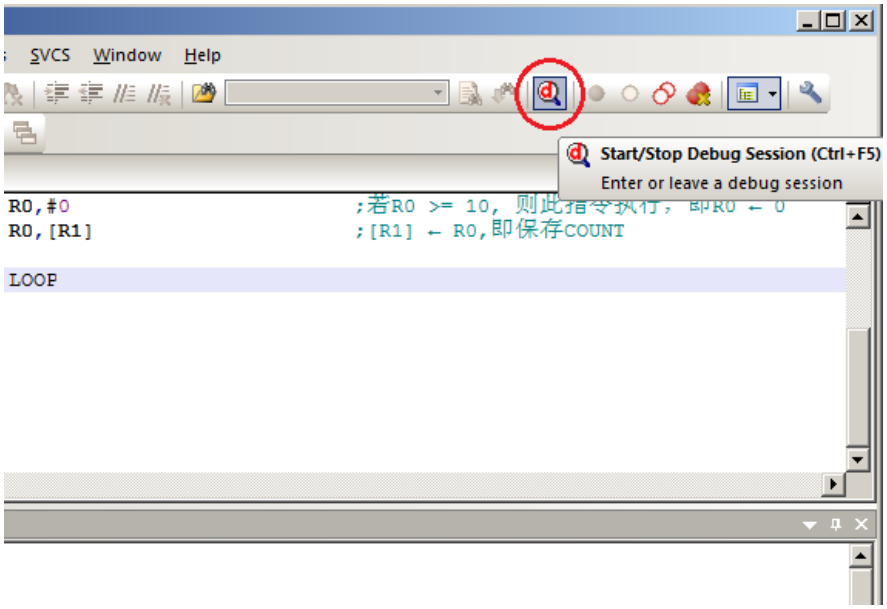

图 2-7 调试运行

(6)单步执行程序，观察板上 LED 灯的变化。

参考：GPIOB 控制寄存器

Register	Address	R/W	Description	Reset Value
GPBCON	0x56000010	R/W	Configures the pins of port B	0x0
GPBDAT	0x56000014	R/W	The data register for port B	Undef.

PBCON	Bit	Description
GPB10	[21:20]	00 = Input 01 = Output 10 = nXDREQ0 11 = reserved
GPB9	[19:18]	00 = Input 01 = Output 10 = nXDACK0 11 = reserved
GPB8	[17:16]	00 = Input 01 = Output 10 = nXDREQ1 11 = Reserved
GPB7	[15:14]	00 = Input 01 = Output 10 = nXDACK1 11 = Reserved
GPB6	[13:12]	00 = Input 01 = Output 10 = nXBREQ 11 = reserved
GPB5	[11:10]	00 = Input 01 = Output 10 = nXBACK 11 = reserved
GPB4	[9:8]	00 = Input 01 = Output 10 = TCLK [0] 11 = reserved
GPB3	[7:6]	00 = Input 01 = Output 10 = TOUT3 11 = reserved
GPB2	[5:4]	00 = Input 01 = Output 10 = TOUT2 11 = reserved]
GPB1	[3:2]	00 = Input 01 = Output 10 = TOUT1 11 = reserved
GPB0	[1:0]	00 = Input 01 = Output 10 = TOUT0 11 = reserved

Register	Address	R/W	Description	Reset Value
GPGCON	0x56000060	R/W	Configures the pins of port G	0x0
GPGDAT	0x56000064	R/W	The data register for port G	Undef.

GPG4	[9:8]	00 = Input 10 = EINT[12]	01 = Output 11 = LCD_PWRDN
GPG3	[7:6]	00 = Input 10 = EINT[11]	01 = Output 11 = nSS1
GPG2	[5:4]	00 = Input 10 = EINT[10]	01 = Output 11 = nSS0
GPG1	[3:2]	00 = Input 10 = EINT[9]	01 = Output 11 = Reserved
GPG0	[1:0]	00 = Input 10 = EINT[8]	01 = Output 11 = Reserved

6. 实验参考程序

GPIO 输出实验的参考程序见程序清单 2.1。GPIO 输入实验的参考程序见程序清单 2.2。

程序清单 2.1 GPIO 输出实验参考程序

```
rGPBCON   EQU       0x56000010              ;Port B control register
rGPBDAT   EQU       0x56000014              ;Port B data register

          AREA RESET, CODE, READONLY
          ENTRY
          CODE32

START     LDR       R1,=rGPBCON             ;set GPIO portB(5,6,7,8) as output
          LDR       R0,=0x15400
```

```
        STR     R0,[R1]

        LDR     R1,=rGPBDAT             ;R1 ← GPIO portB data register

LOOP    MOV     R0,#0x1e0               ;set GPIO portB(5,6,7,8) output high (LEDs off)
        STR     R0,[R1]

        MOV     R0,#0x00                ;set GPIO portB(5,6,7,8) output high (LEDs on)
        STR     R0,[R1]

        B       LOOP

        END
```

程序清单 2.2 GPIO 输入实验参考程序

```
rGPGCON EQU     0x56000060              ;Port G control register
rGPGDAT EQU     0x56000064              ;Port G data register

        AREA RESET, CODE, READONLY      ;声明代码段 RESET
        ENTRY                           ;表示程序入口
        CODE32                          ;声明 32 位 ARM 指令

START   LDR     R1,=rGPGCON             ;set GPIO portG(0) as input
        LDR     R0,=0x0
        STR     R0,[R1]

        MOV     R3, #0x0                ;initialize counter to 0
        LDR     R0,=0x7fff              ;initialize to default state
        LDR     R1,=rGPGDAT             ;R1 ← GPIO portG data register

LOOP    MOV     R2,R0                   ;save the old state
        LDR     R0,[R1]                 ;read GPIO portG state
        CMP     R0, R2                  ;detect state change
        BEQ     LOOP
        ADD     R3, #0x1                ;increase the counter

        B       LOOP

        END
```

7. 思　考

(1)GPIO 输出实验程序全速执行时能看到指示灯的闪烁吗,为了控制闪烁速

度程序要如何扩充？

(2)GPIO 输入实验中，如果信号即不接高电平，也不接低电平，读入的状态是什么呢？

8. 选作内容

(1)增加延时操作控制指示灯的闪烁速度。

(2)用按键控制指示灯的状态。