

第4章 单片机的“触角”——I/O口

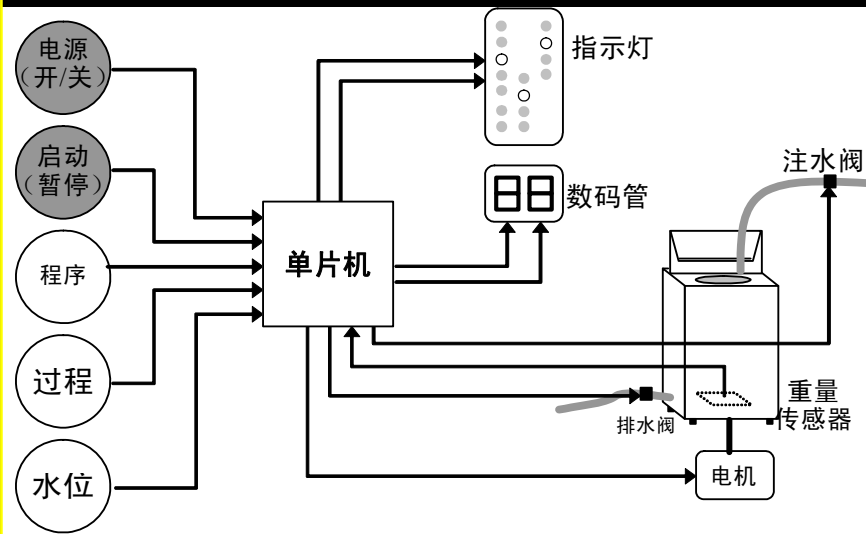
欢迎访问 电路飞翔网

<http://www.circuitfly.com> 获取更多信息

- 4.1 解读AT89S51的I/O口
- 4.2 I/O口作输入端口使用——流水控制灯
- 4.3 七段数码管的控制——秒表
- 4.4 小键盘的控制
- 4.5 实例点拨——计时提醒器

4.1 解读AT89S51的I/O口

• 4.1.1 I/O口的整体印象

系统示意图	功 能	部 件	I/O功能
	电源	电源开关	输入
	启动/暂停	启动/暂停按钮	输入
	过程指示	指示灯	输出
	时间显示	七段数码管	输出
	程序控制	程序按钮	输入
	水位控制	水位按钮	输入
	注水控制	注水阀	输出
	称衣物重量	重量传感器	输入
	排水控制	排水阀	输出
	波轮旋转	电动机	输出

洗衣机与单片机的I/O控制

4.1 解读AT89S51的I/O口

• 4.1.1 I/O口的整体印象(89S51的I/O口)

P3口具有
双重功能

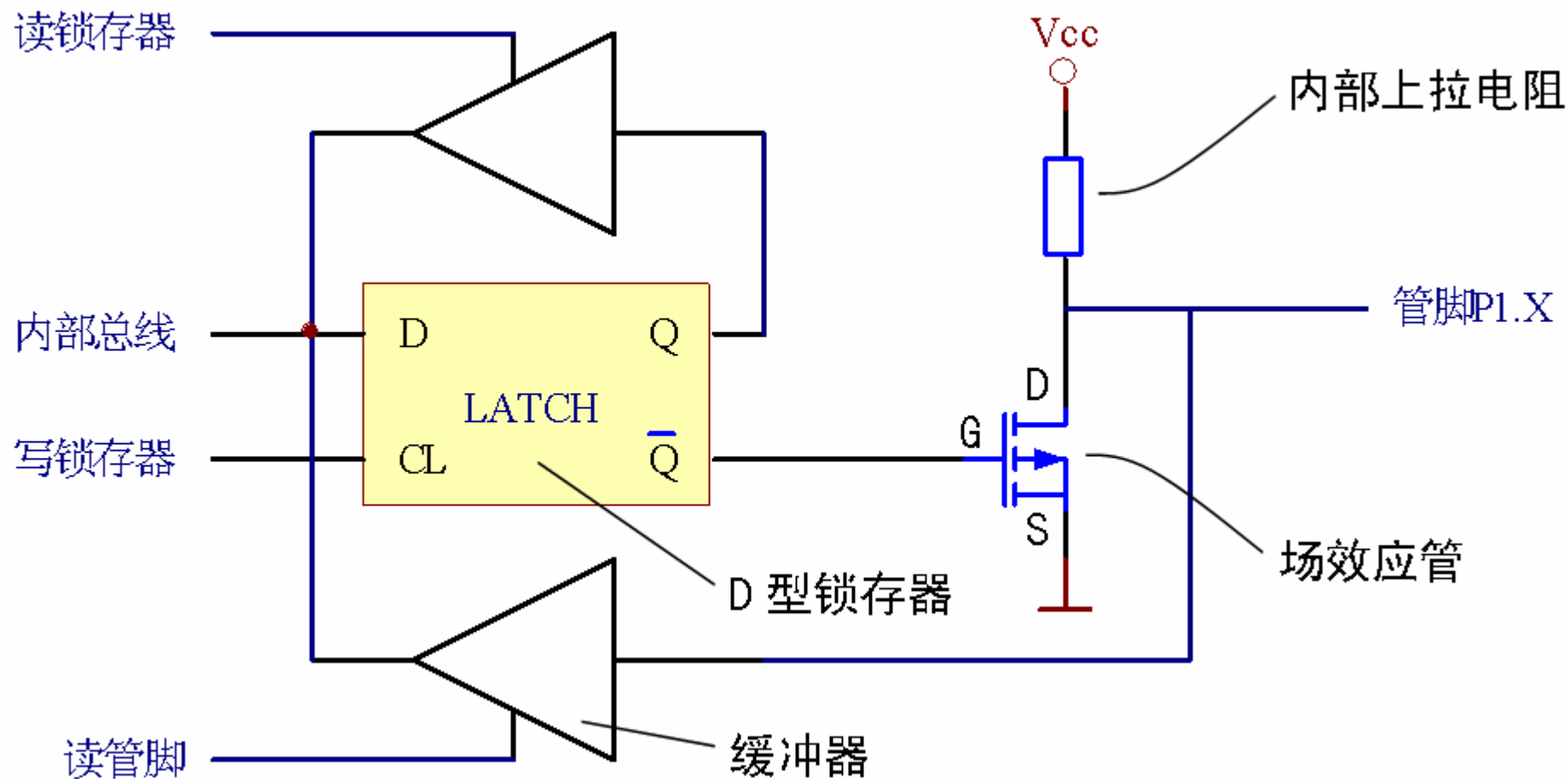
31	<u>EA/VPP</u>	Vcc	40
19	XTAL1	AD0/P0.0	39
		AD1/P0.1	38
18	XTAL2	AD2/P0.2	37
		AD3/P0.3	36
9	RST	AD4/P0.4	35
		AD5/P0.5	34
10	P3.0/RXD	AD6/P0.6	33
11	P3.1/TXD	AD7/P0.7	32
12	P3.2/ <u>INT0</u>		
13	P3.3/ <u>INT1</u>	A8/P2.0	21
14	P3.4/T0	A9/P2.1	22
15	P3.5/T1	A10/P2.2	23
		A11/P2.3	24
1	P1.0	A12/P2.4	25
2	P1.1	A13/P2.5	26
3	P1.2	A14/P2.6	27
4	P1.3	A15/P2.7	28
5	P1.4		
6	P1.5/MOSI	<u>PSEN</u>	29
7	P1.6/MISO	ALE/PROG	30
8	P1.7/SCK		
		P36/ <u>WR</u>	16
20	GND	P37/ <u>RD</u>	17

P0和P2口
访问外部存储器

P3口具有双重功能

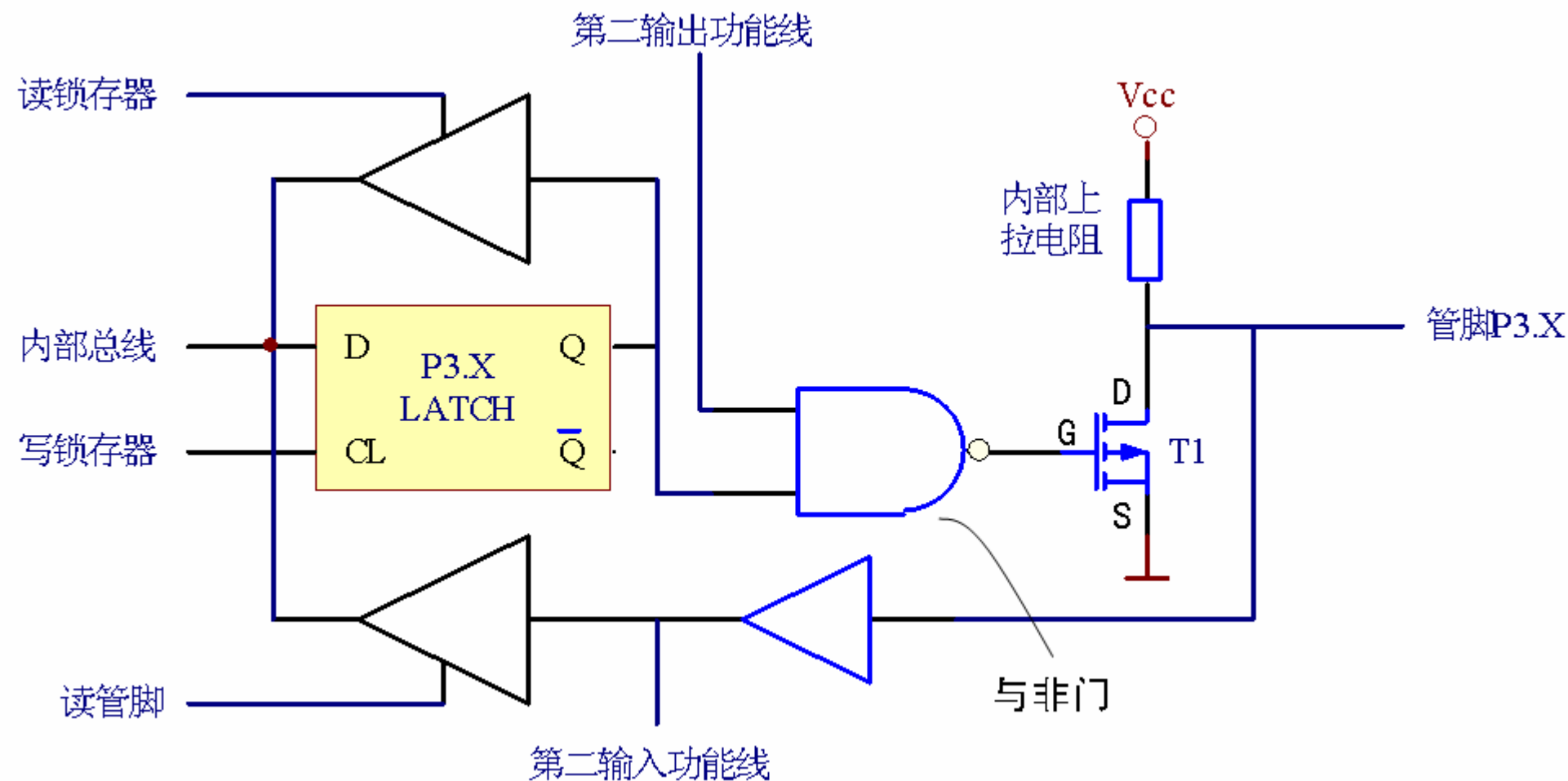
4.1 解读AT89S51的I/O口

• 4.1.2 深入观察I/O口的结构（P1口结构）



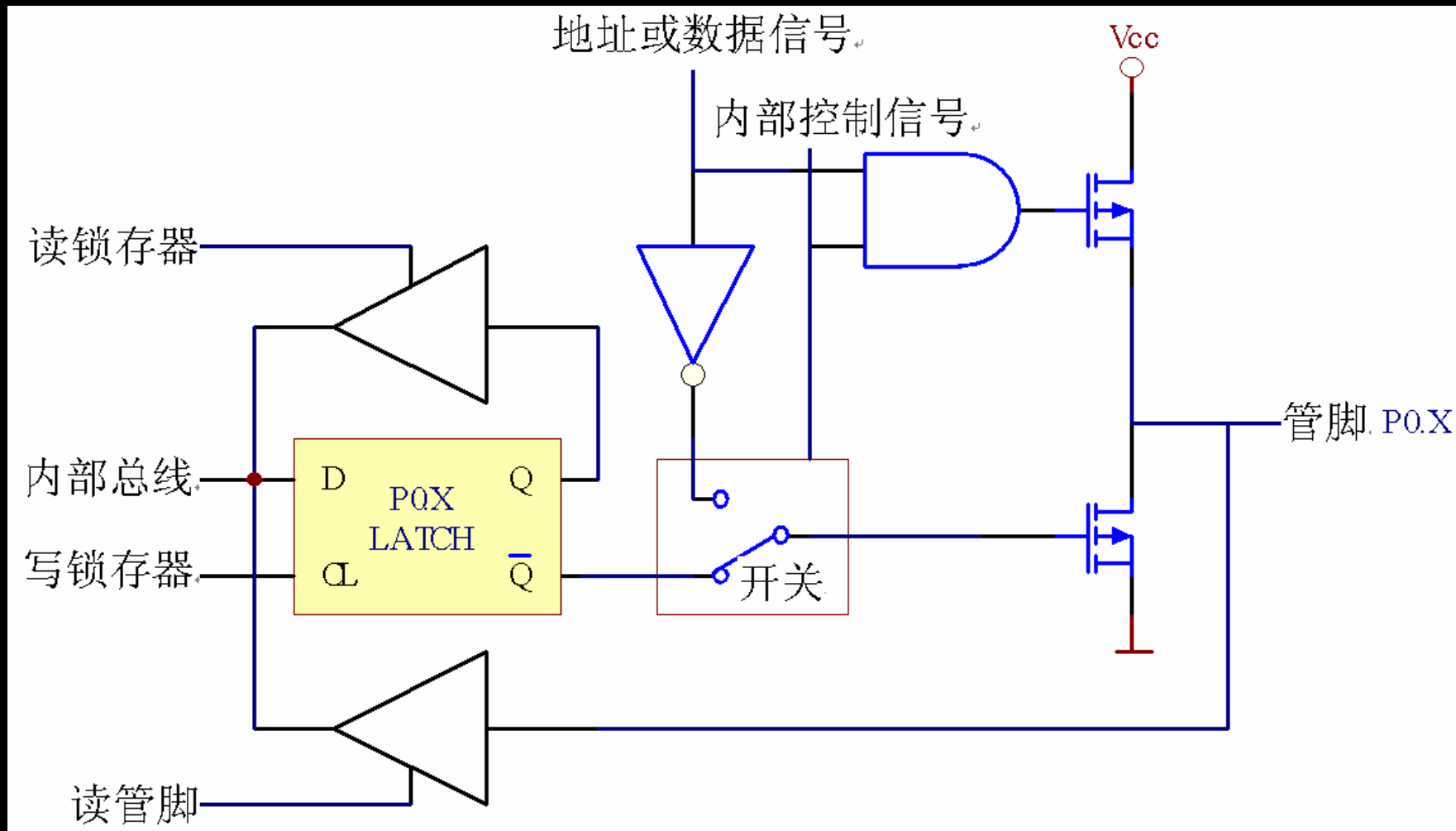
4.1 解读AT89S51的I/O口

• 4.1.2 深入观察I/O口的结构（P3口结构）



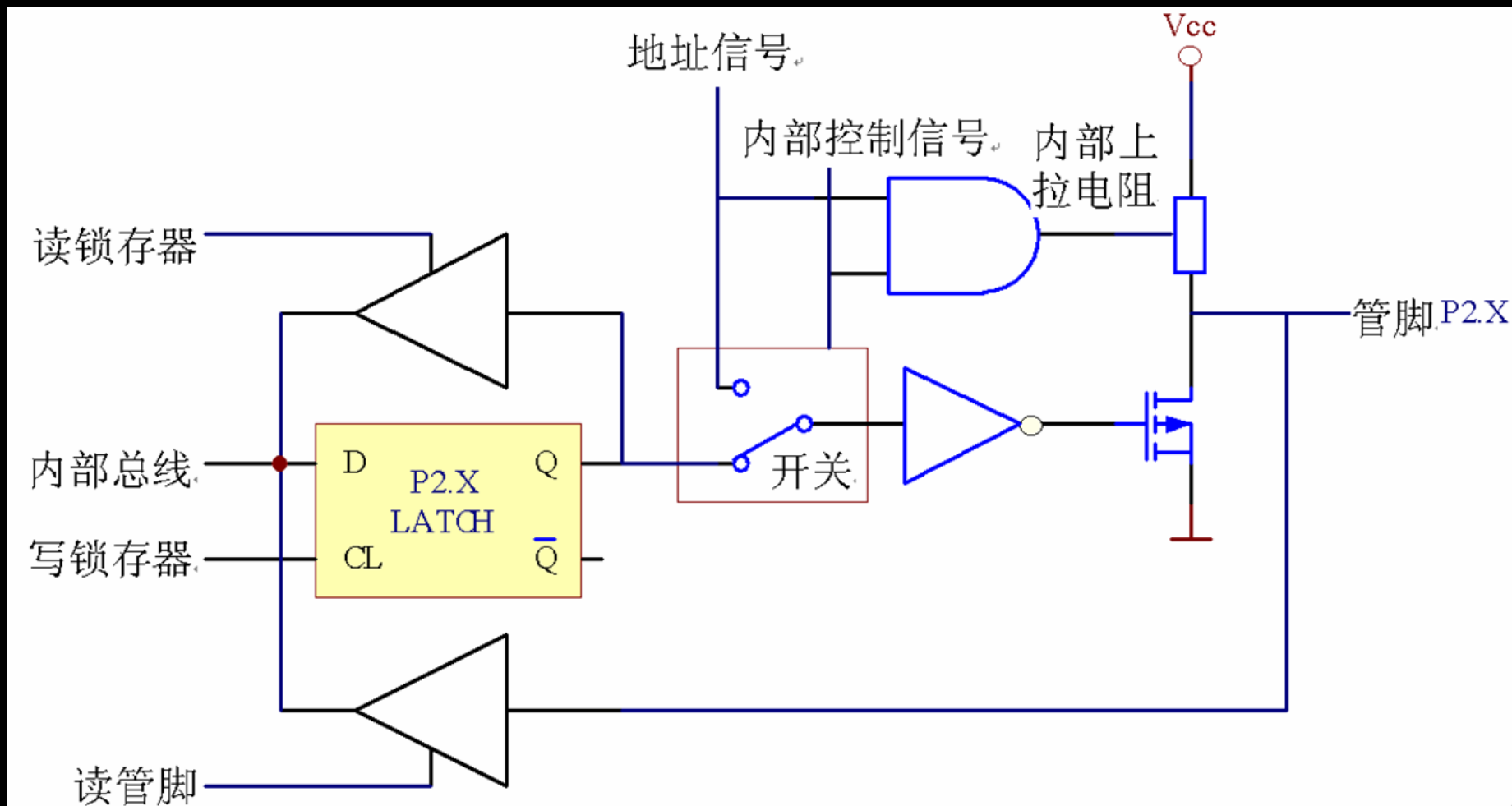
4.1 解读AT89S51的I/O口

• 4.1.2 深入观察I/O口的结构（P0口结构）



4.1 解读AT89S51的I/O口

• 4.1.2 深入观察I/O口的结构（P2口结构）



4.2 I/O口作输入端口使用

——流水控制灯

- 明确系统功能

8支发光二极管作为显示器件， 两个按钮开关A和B，

一开始， 8支发光二极管为全亮状态，

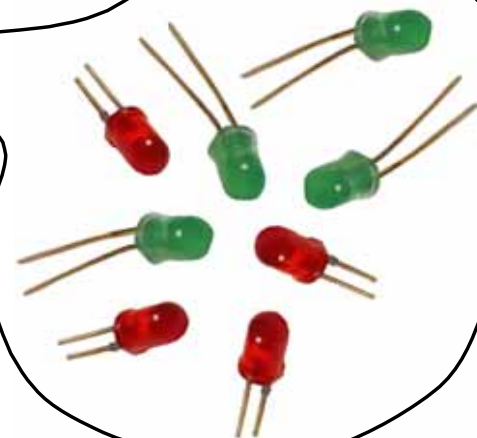
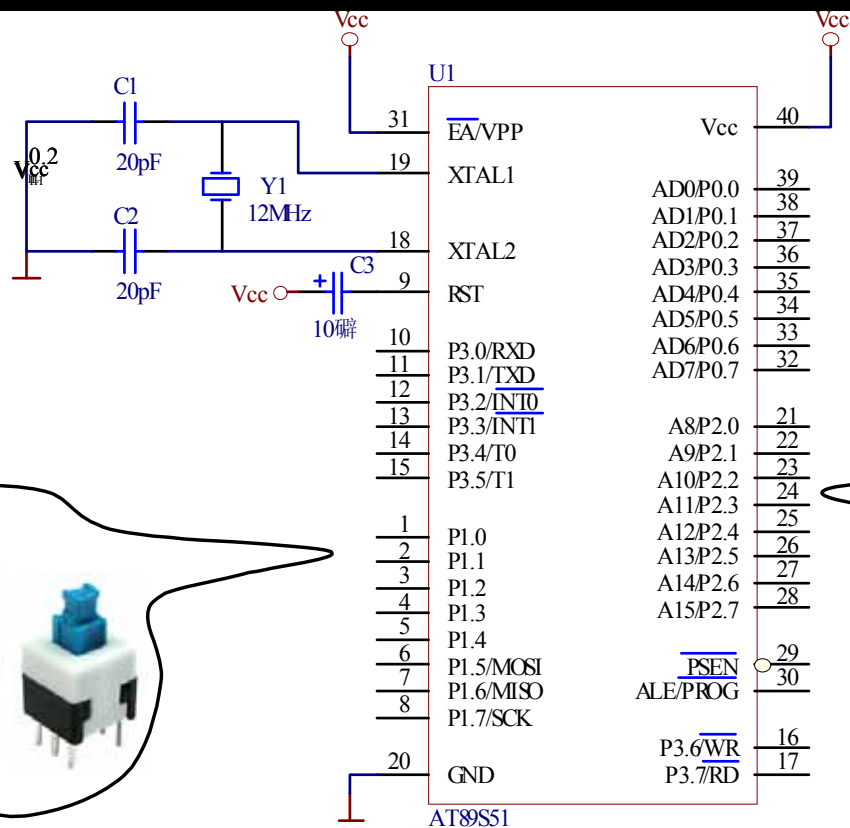
当按下按钮A时， 8支发光二极管做单一灯的从右向左流动，

当按下按钮B时， 8支发光二极管做单一灯的从左向右流动。

4.2 I/O口作输入端口使用

——流水控制灯

• 硬件设计

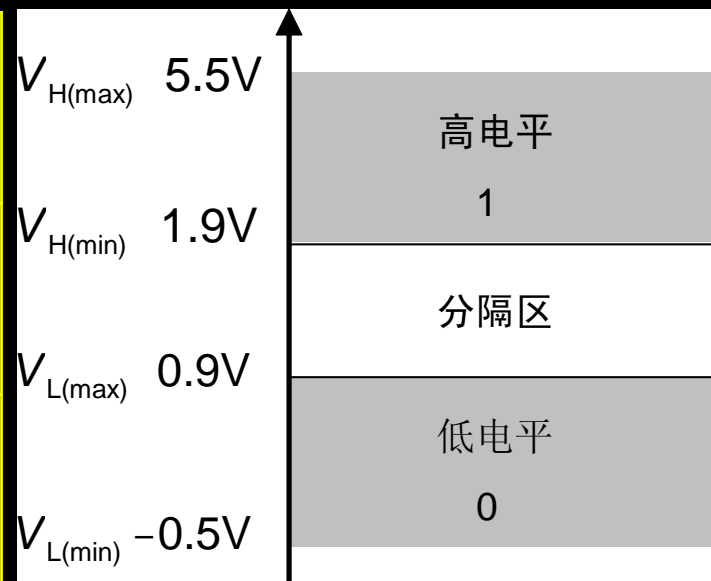


4.2 I/O口作输入端口使用

——流水控制灯

- 硬件设计 （高低电平）

表示符号	参数	最小值	最大值	单位
V_{IL}	输入低电平	-0.5	-0.1	V
V_{IH}	输入高电平	+0.9	+0.5	V

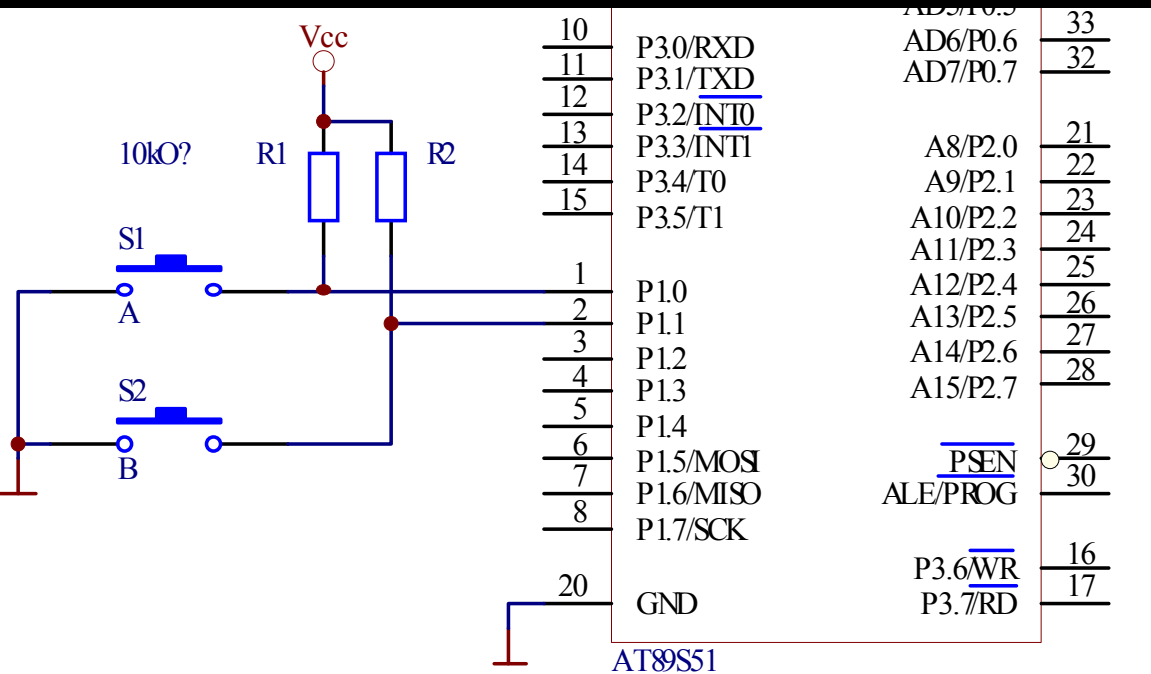


AT89S51单片机的逻辑电平描述

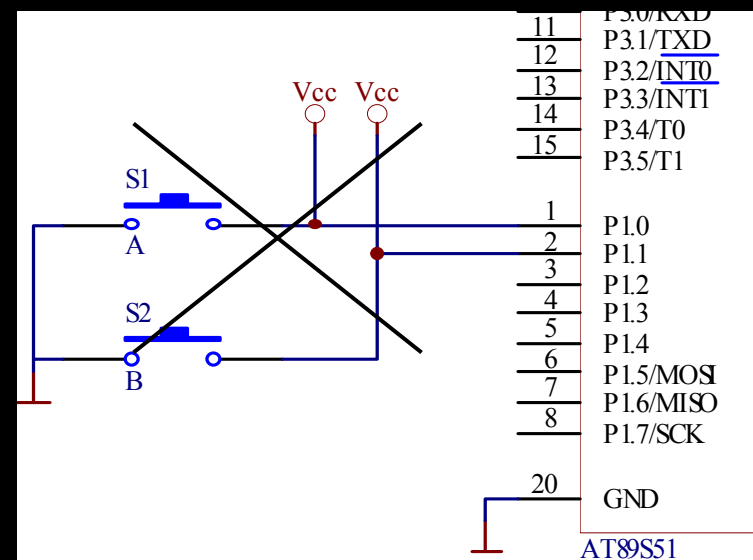
4.2 I/O口作输入端口使用

——流水控制灯

- 硬件设计（I/O口作为输入口）



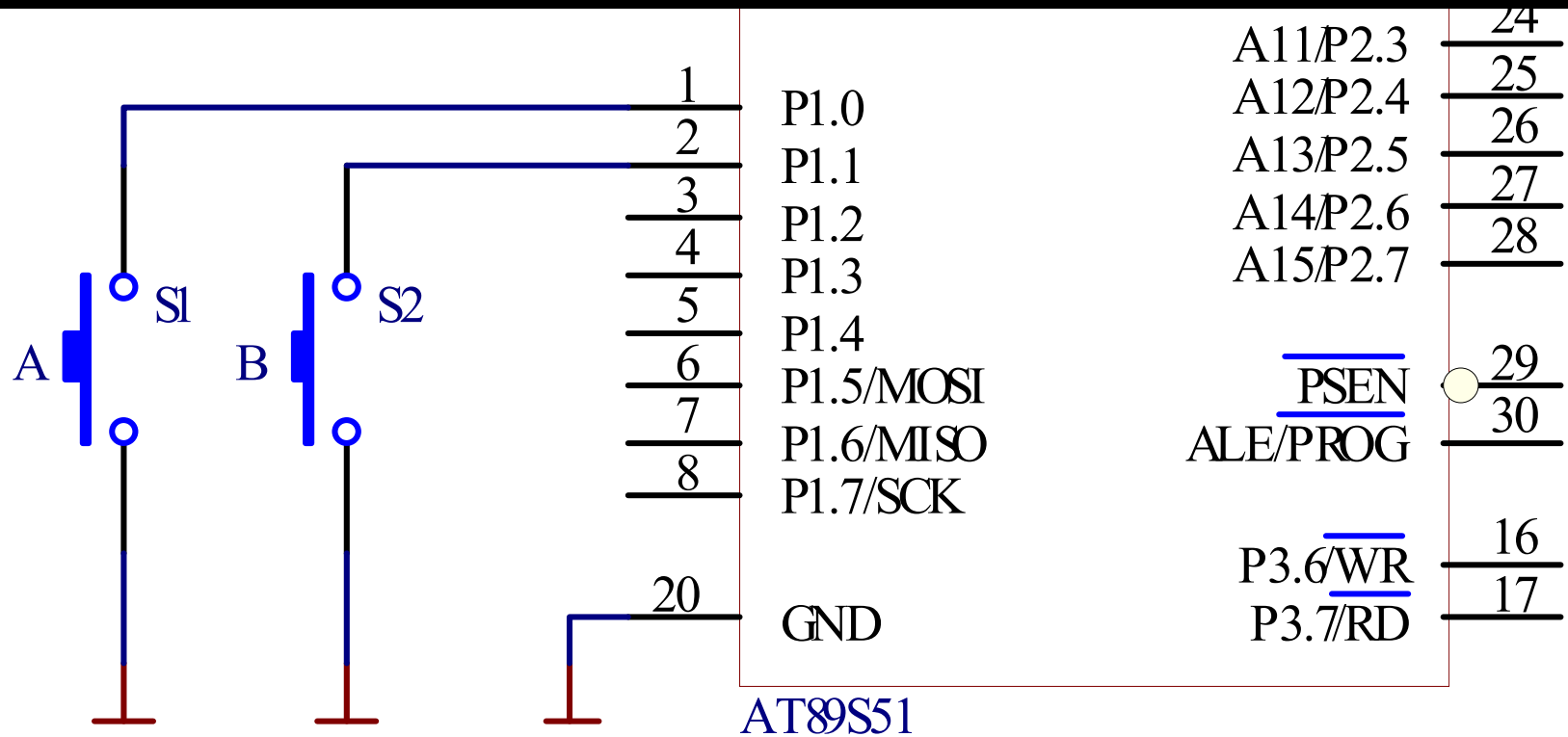
(a) 上拉电阻方案



4.2 I/O口作输入端口使用

——流水控制灯

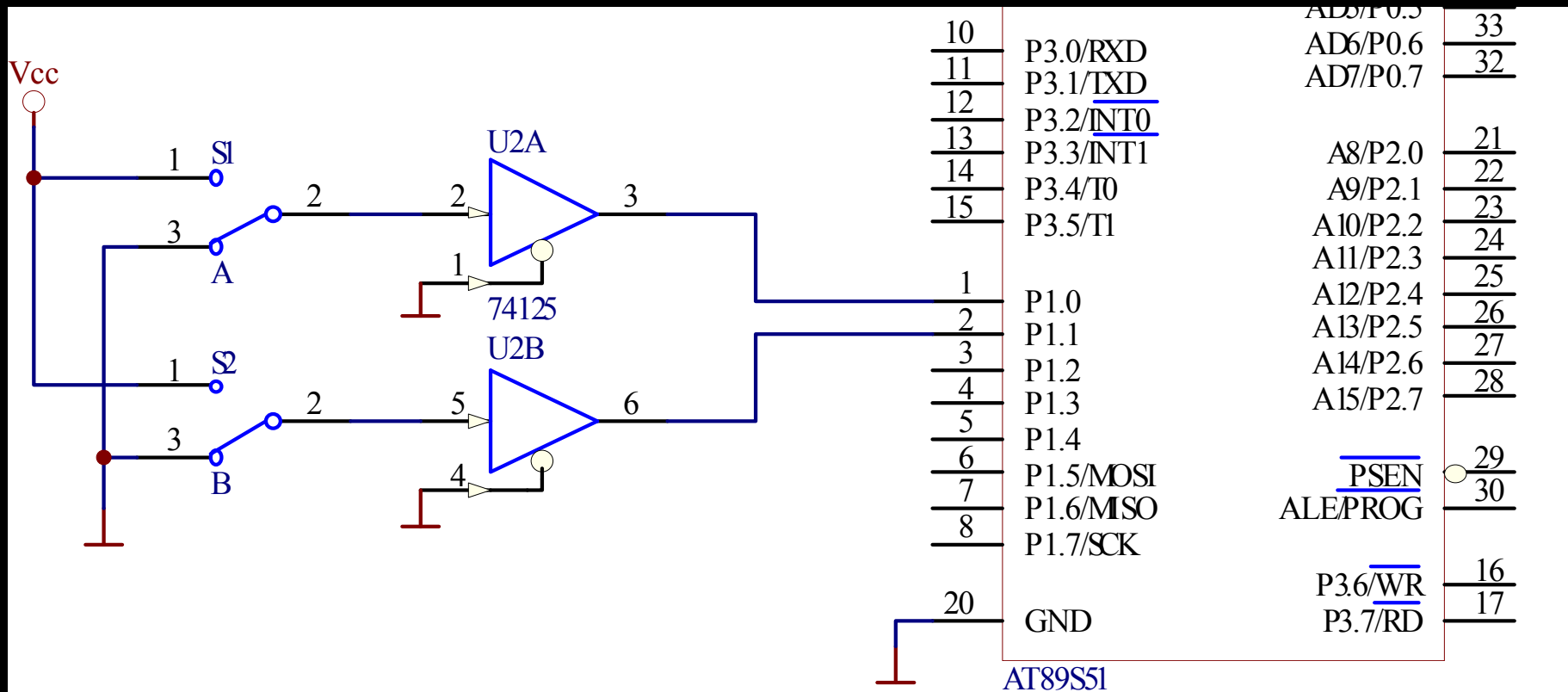
- 硬件设计（I/O口作为输入口）



(b) 简易方案

——流水控制灯

- 硬件设计（I/O口作为输入口）

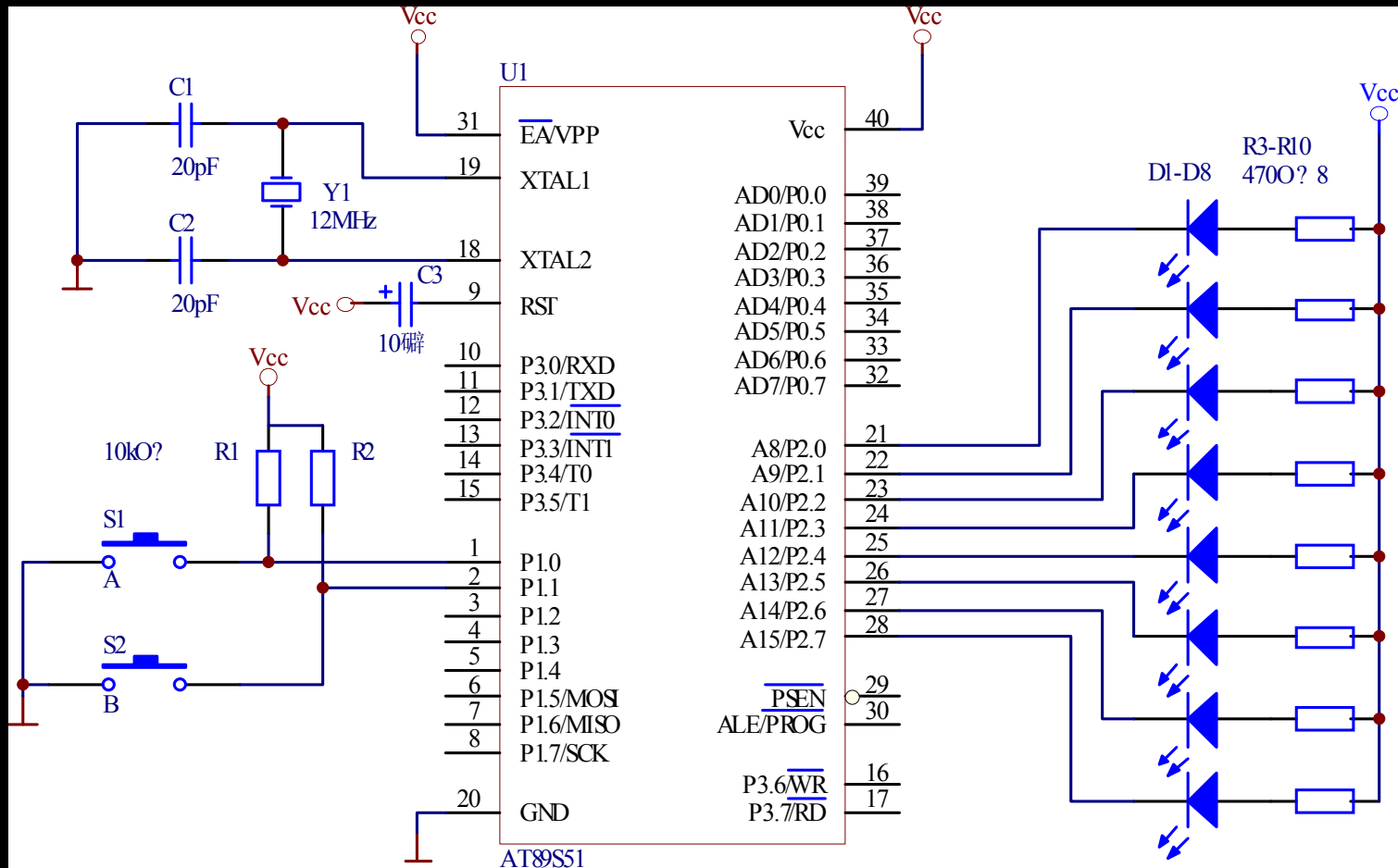


(c) 缓冲器方案

4.2 I/O口作输入端口使用

——流水控制灯

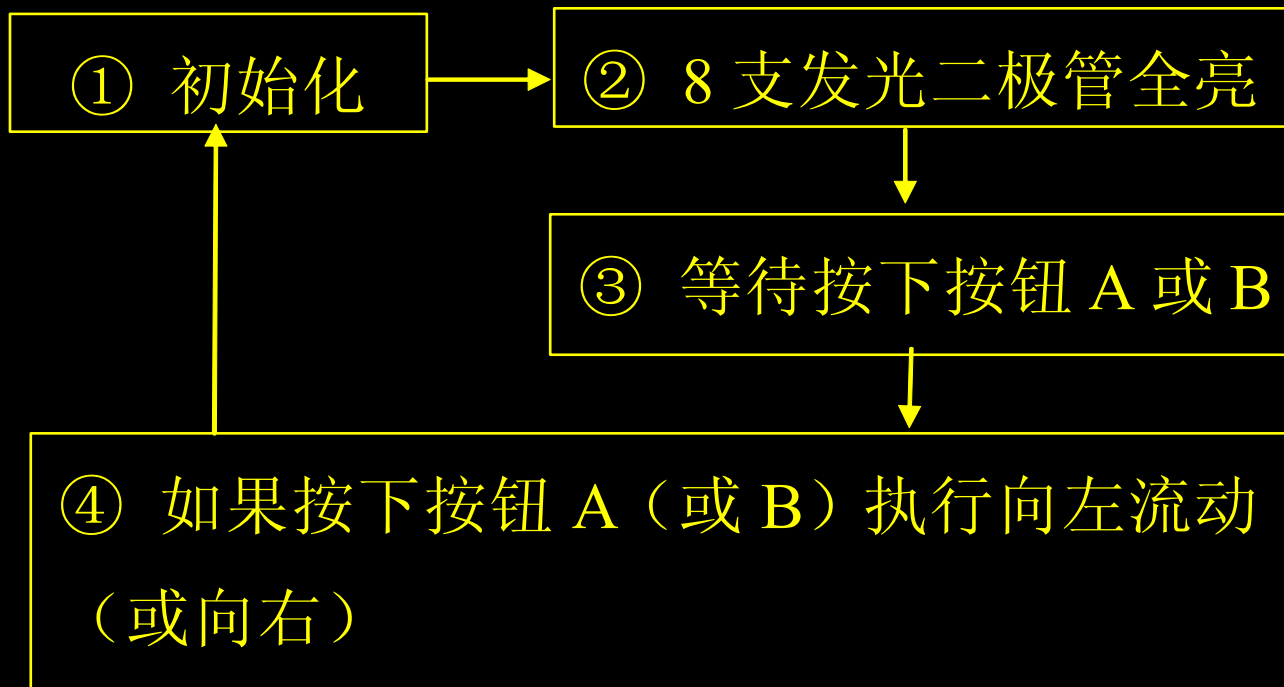
• 硬件设计 （系统电路图）



4.2 I/O口作输入端口使用

——流水控制灯

- 搭建硬件平台
- 软件设计



程序设计思路

4.2 I/O口作输入端口使用

——流水控制灯

- 软件设计（指令简析“JB P1.0, WAIT_A”）

.....

WAIT_A:

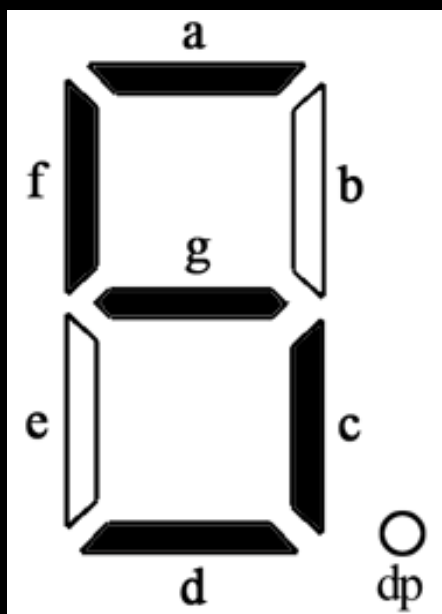
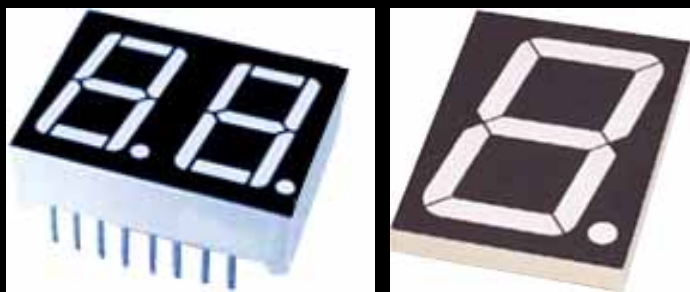
JB P1.0, WAIT_A ;如果按钮A没有按下,
;循环执行本行以继续判断

JMP LEFT

.....

4.3 七段数码管的控制——秒表

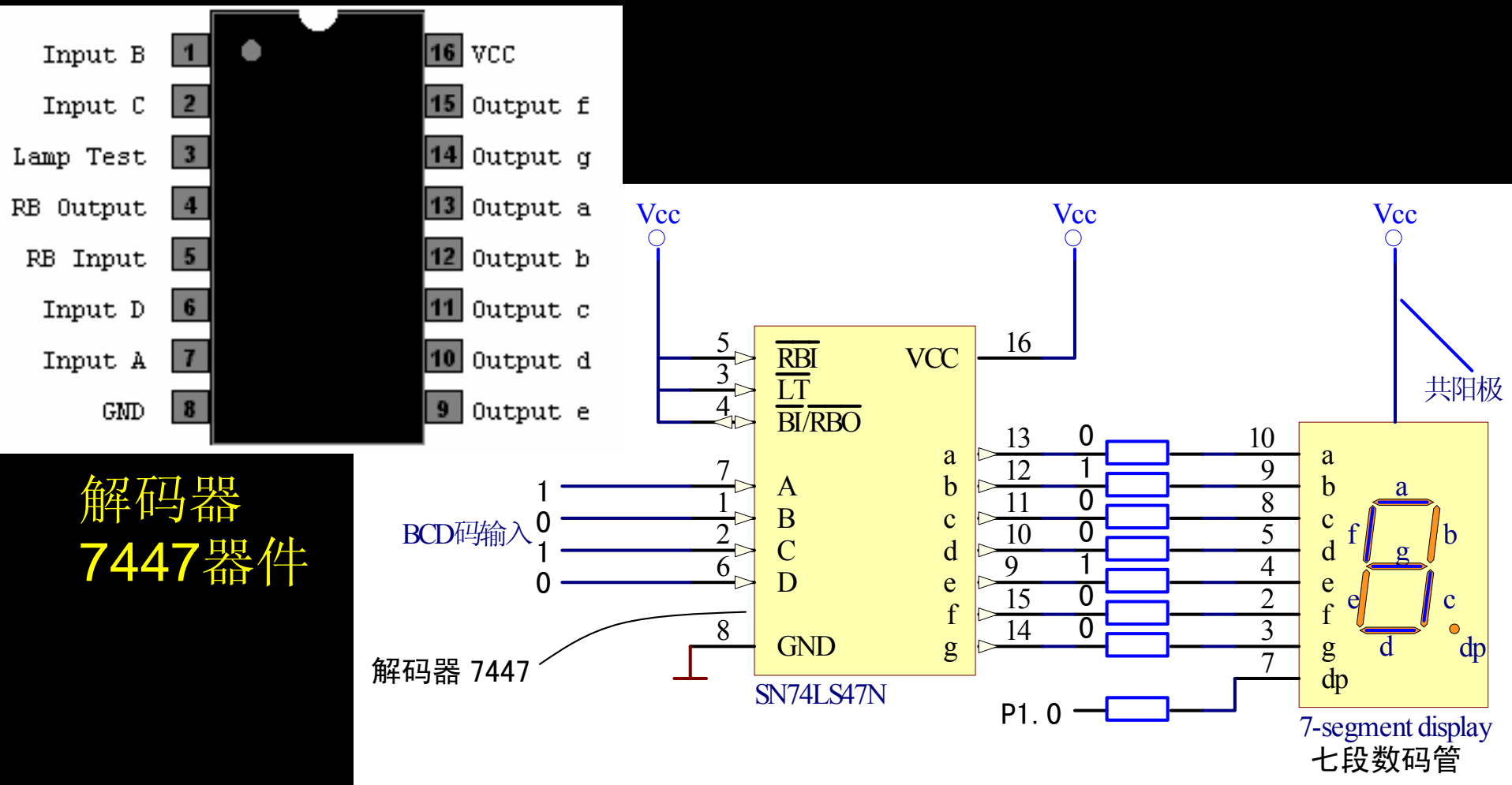
• 七段数码管介绍



数 字	亮 段
0	a,b,c,d,e,f
1	b,c
2	a,b,d,e,g
3	a,b,c,d,g
4	b,c,f,g
5	a,c,d,f,g
6	a,c,d,e,f,g
7	a,b,c
8	a,b,c,d,e,f,g
9	a,b,c,d,f,g

4.3 七段数码管的控制——秒表

• 七段数码管介绍 (解码器7447)



4.3 七段数码管的控制——秒表

- 七段数码管介绍 (解码器7447)

显 示	BCD码输入端				亮段控制输出端						
	D	C	B	A	a	b	c	d	e	f	g
0	0	0	0	0	0	0	0	0	0	0	1
1	0	0	0	1	1	0	0	1	1	1	1
2	0	0	1	0	0	0	1	0	0	1	0
3	0	0	1	1	0	0	0	0	1	1	0
4	0	1	0	0	1	0	0	1	1	0	0
5	0	1	0	1	0	1	0	0	1	0	0
6	0	1	1	0	0	1	0	0	0	0	0
7	0	1	1	1	0	0	0	1	1	1	1
8	1	0	0	0	0	0	0	0	0	0	0
9	1	0	0	1	0	0	0	0	1	0	0

4.3 七段数码管的控制——秒表

- 明确系统功能

两位七段数码管在开机时显示“00”，

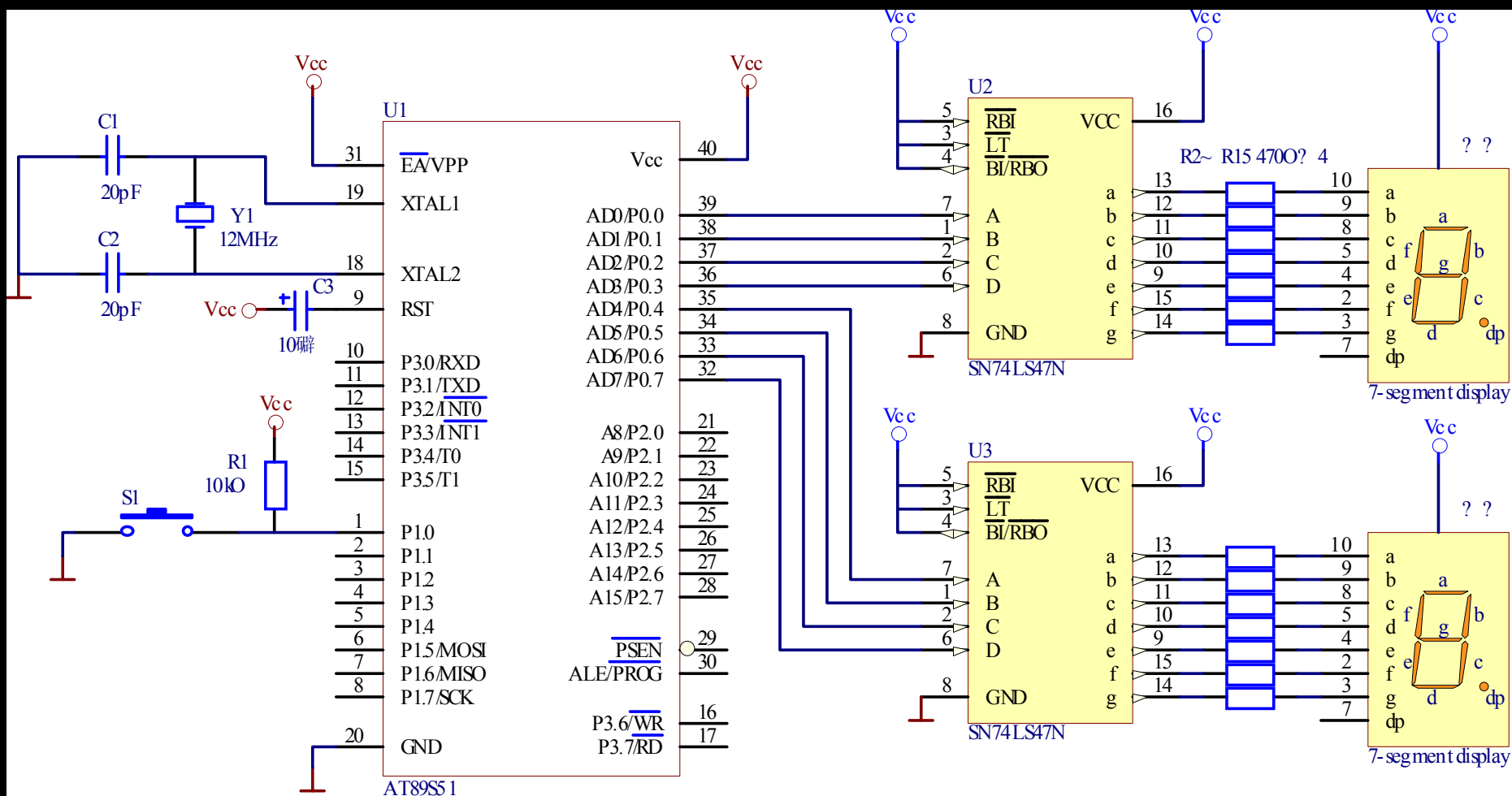
第1次按下按钮开关后秒表开始计时，

第2次按下后计时停止，

第3次按下后两个数码管清0，并回到一开始的计时状态。

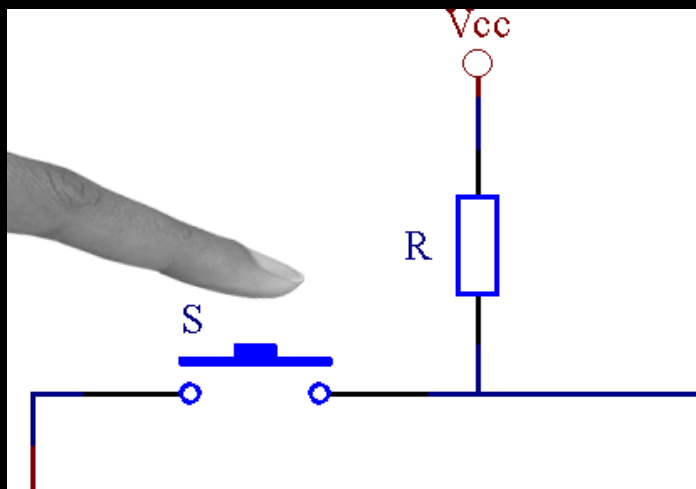
4.3 七段数码管的控制——秒表

• 硬件设计(系统电路图)

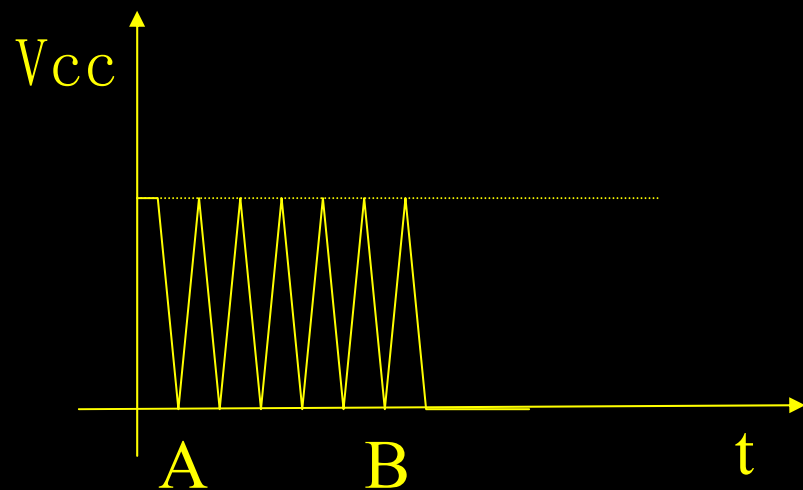


4.3 七段数码管的控制——秒表

- 软件设计前奏 (解决开关抖动)



△按下开关时手指的抖动

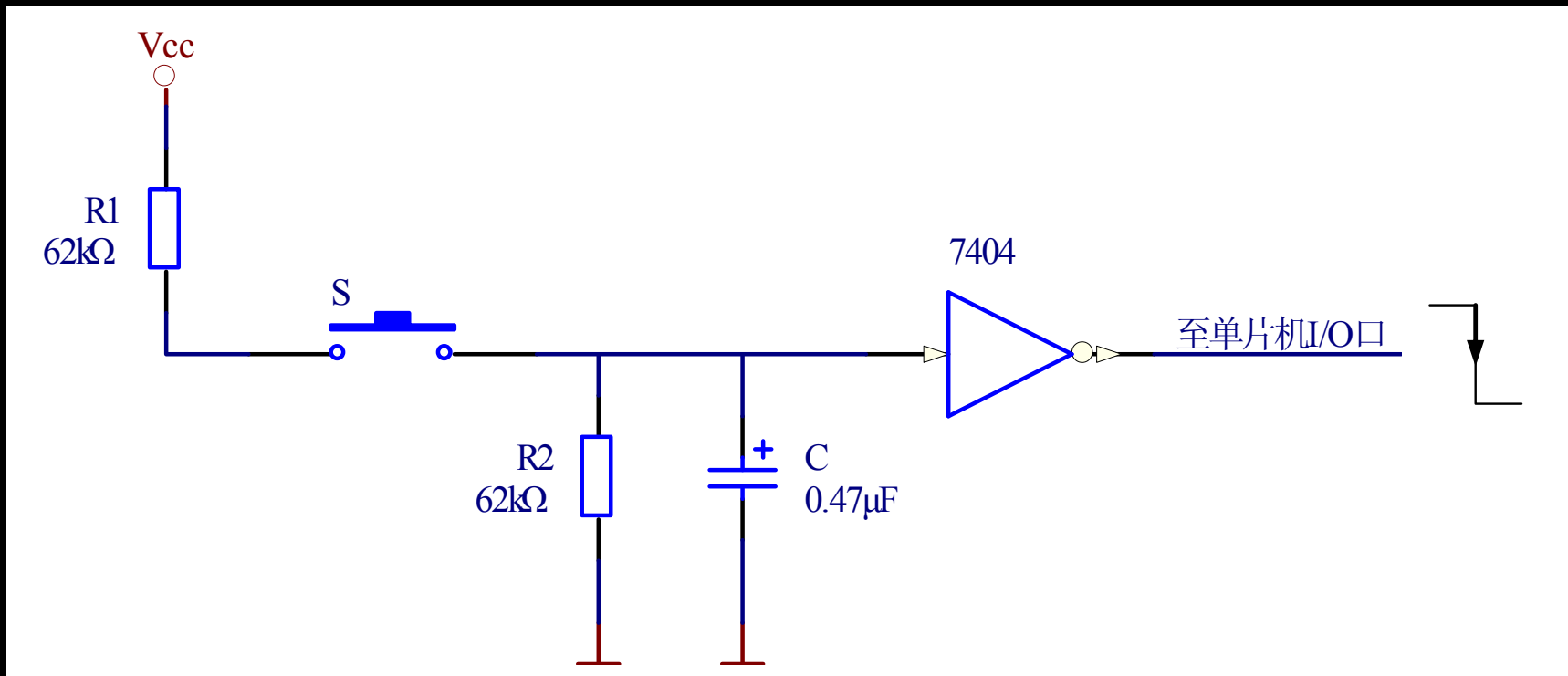


△输出端的电平变化

解决开关抖动的办法： 硬件的方法 软件的方法

4.3 七段数码管的控制——秒表

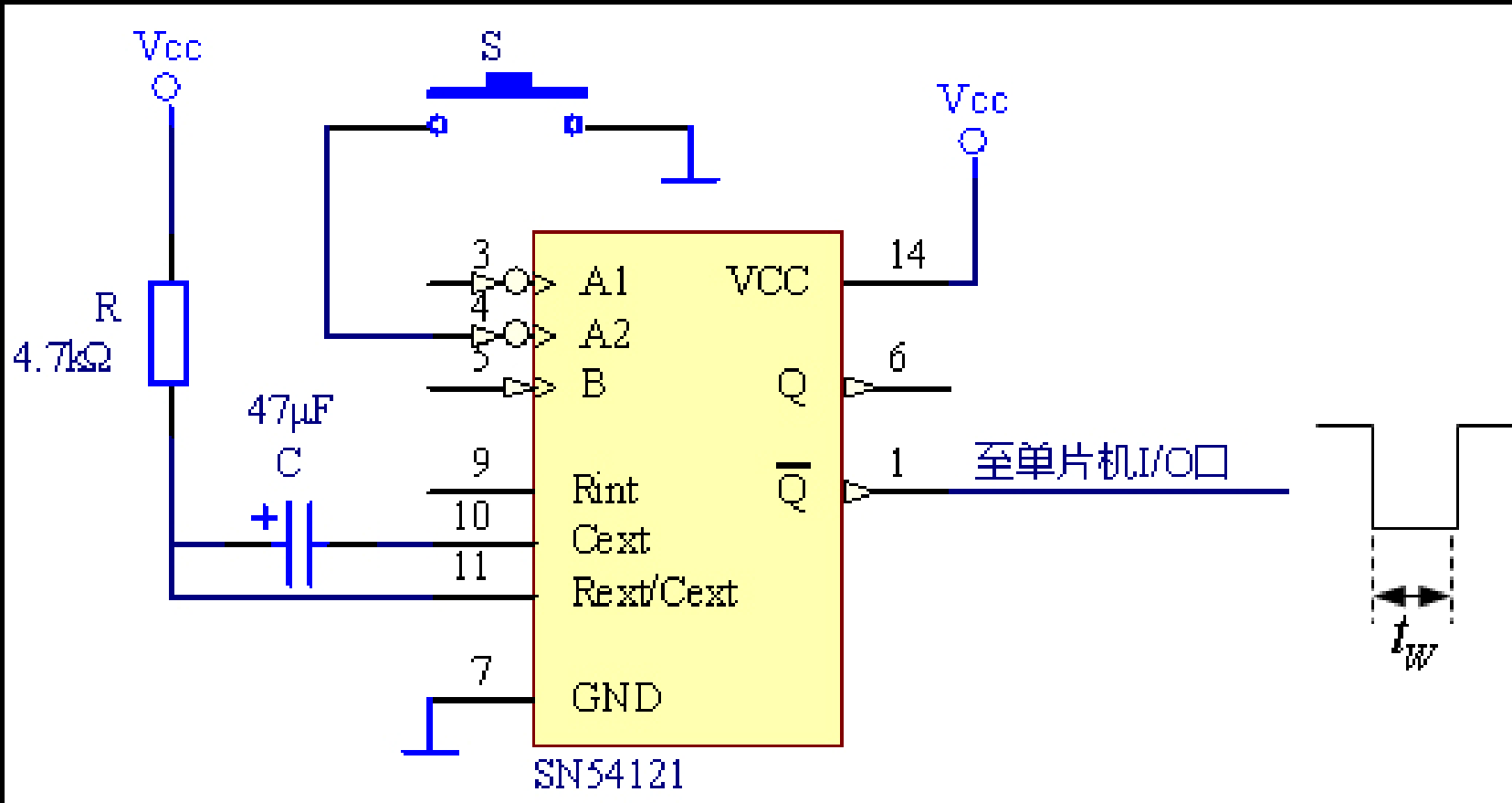
- 软件设计前奏 (硬件的方法解决开关抖动)



门控滤波电路

4.3 七段数码管的控制——秒表

- 软件设计前奏 (硬件的方法解决开关抖动)



振荡器滤波电路

4.3 七段数码管的控制——秒表

- 软件设计前奏 (软件的方法解决开关抖动)

JB P1.0, \$;如果P1.0为高电平, 就重复执行本行

CALL FILTER ;调消除开关抖动的子程序

JNB P1.0, \$;如果P1.0为低电平, 就重复执行本行

注释:

“CALL FILTER ”调用消除开关抖动的延时子程序

4.3 七段数码管的控制——秒表

- 软件设计

启动时，显示00和等待按钮按下

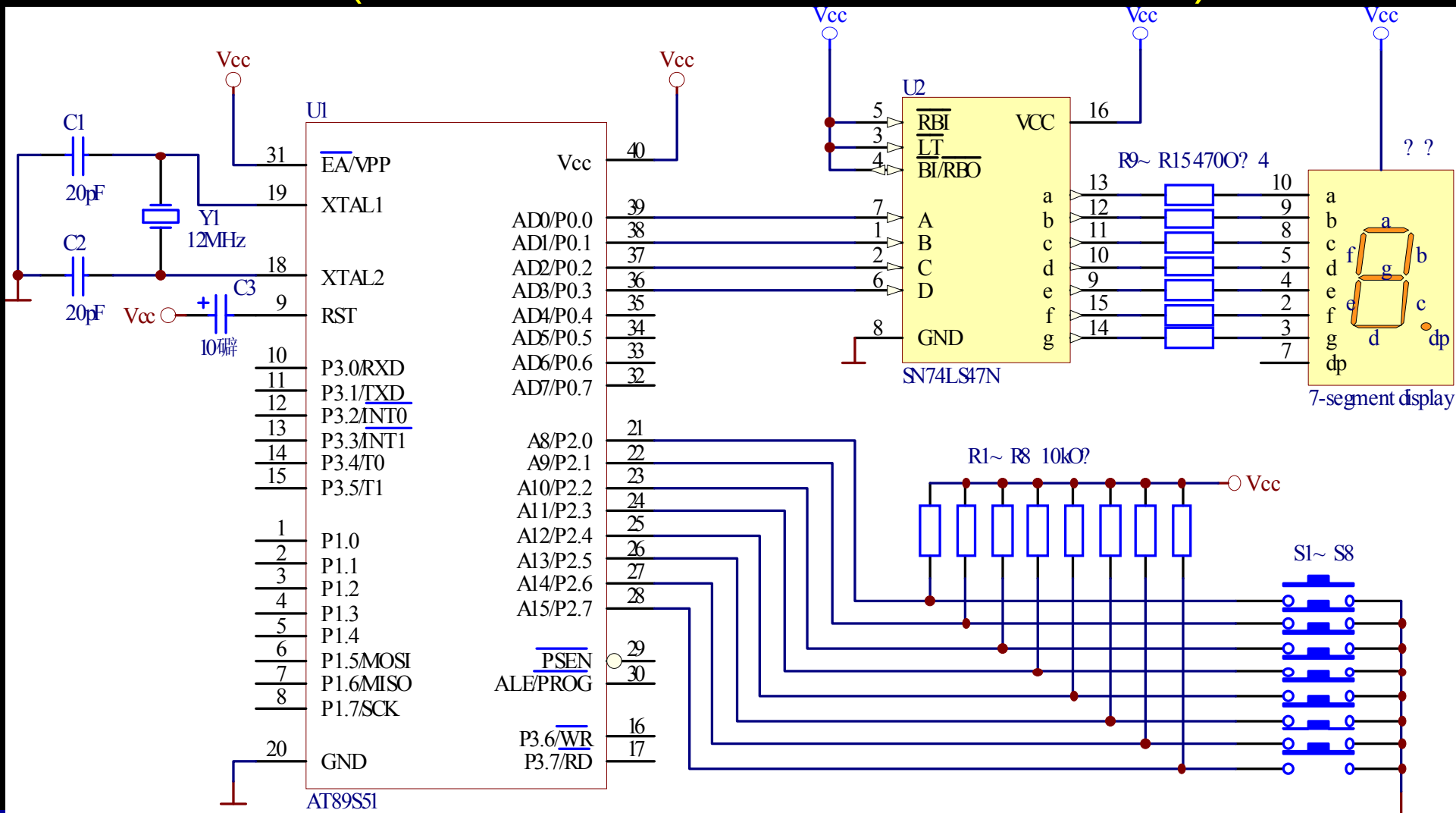
秒表计时显示和1秒延时

处理第2次、第3次按下按钮的处理

延时程序

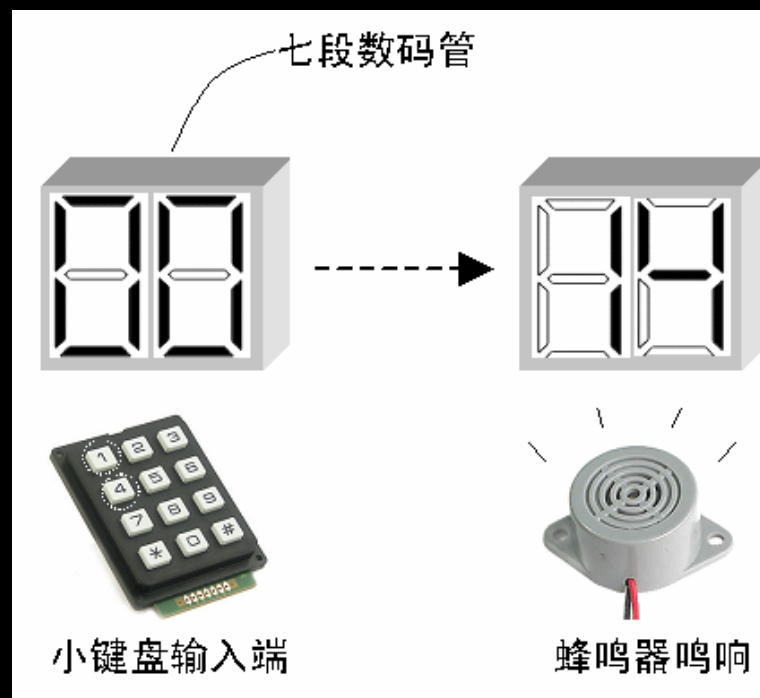
4.4 小键盘的控制

- 上拉电阻(避免与Vcc直连, 开关断路无电流)



4.5 实例点拨——计时提醒器

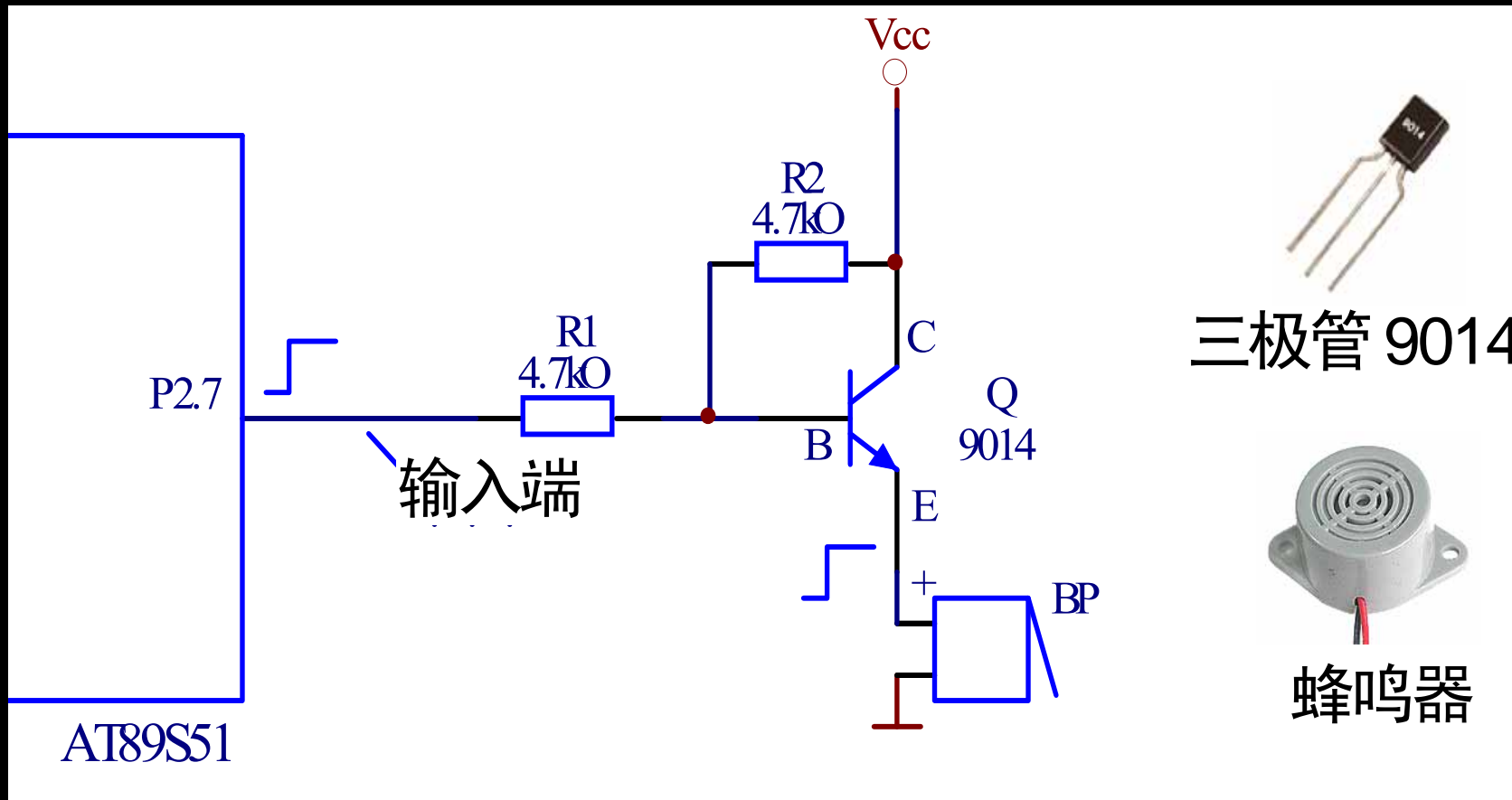
- 明确系统功能
 - ✓ 开机时，七段数码管显示“00”，等待输入计时时间，
 - ✓ 输入有误，按“#”键取消重新输入，
 - ✓ 按“*”键则确认，计时提醒器开始工作，
 - ✓ 计时完时，蜂鸣器发出“嘀、嘀……”的提示音，
 - ✓ 在计时过程中，如果按下“#”键则取消计时，系统回到开机时的状态。



4.5 实例点拨——计时提醒器

- 硬件设计

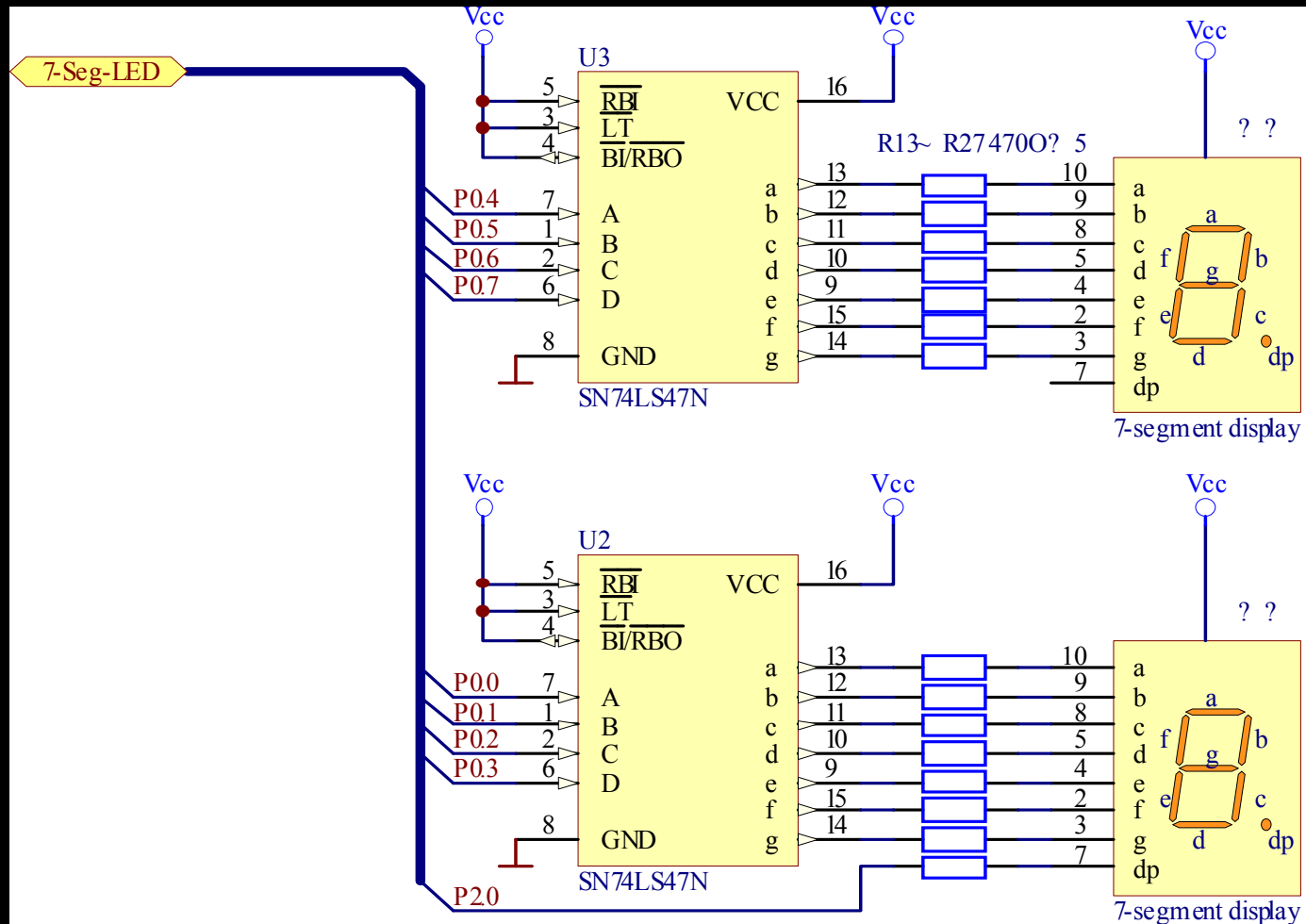
12个按键，两位七段数码管，蜂鸣器和单片机最简系统



4.5 实例点拨——计时提醒器

• 硬件设计

两位七
段数码
管和解
码器
7447



- 硬件设计



4.5 实例点拨——计时提醒器

- 软件设计（SWAP指令介绍）

MOV A, #11110000B

；向累加器ACC中载入二进制数11110000，后缀B
代表二进制数

SWAP A

；累加器ACC中的低4位与高4位对调，
(A) = 0000 1111

4.5 实例点拨——计时提醒器

• 软件设计（好习惯1/2）

✓ 阅读程序的好习惯

.....

MOV R1, #0F2H

MOV A, R1

SWAP A

ADD A, #01H

MOV P0, A

.....

.....

MOV R1, #0F2H → R1 = 0F2H

MOV A, R1 → A = R1 = 0F2H

SWAP A → A = 2FH

ADD A, #01H → A = 2F + 01 = 30H

MOV P0, A → P0 = A = 30H

.....

在纸上用笔记录每一行
指令产生的影响和结果

✓ 程序中的标号应该尽量起得有实际、具体、确切的含义，这样能提高程序的书写、编辑、调试的效率。

4.5 实例点拨——计时提醒器

- 软件设计（好习惯2/2）
- ✓ 一般来说，标号写在最左边，最好给标号单独占一行。在它的下面接着写程序。每一行指令的起始点，也就是汇编指令的助记符应该比标号偏右一些，以利于区别标号和指令。

```

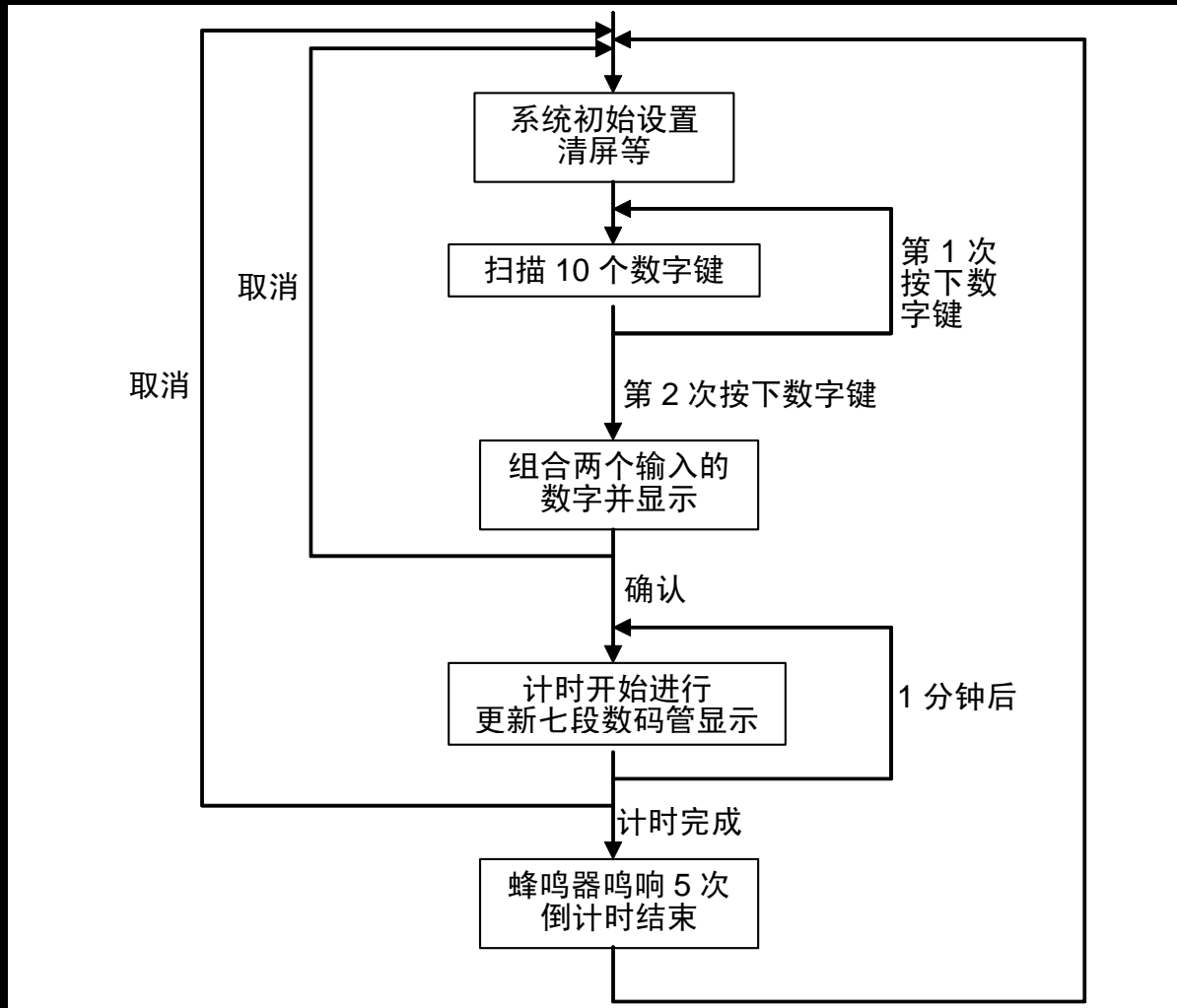
ORG      00H
START:   MOV      P0,#00H
          MOV      R0,#2
          CLR      C
          CLR      P2.7
SETTING:  CHK_0:
          JB       P1.0,CHK_1
          CALL     FILTER
          JNB      P1.0,$
          MOV      R1,#0
          JMP      TIMERSET
  
```

```

                                ORG      00H
START:
                                MOV      P0,#00H
                                MOV      R0,#2
                                CLR      C
                                CLR      P2.7
SETTING:
                                CHK_0:
                                JB       P1.0,CHK_1
                                CALL     FILTER
                                JNB      P1.0,$
                                MOV      R1,#0
                                JMP      TIMERSET
  
```

4.5 实例点拨——计时提醒器

- 软件设计（程序流程图）



4.5 实例点拨——计时提醒器

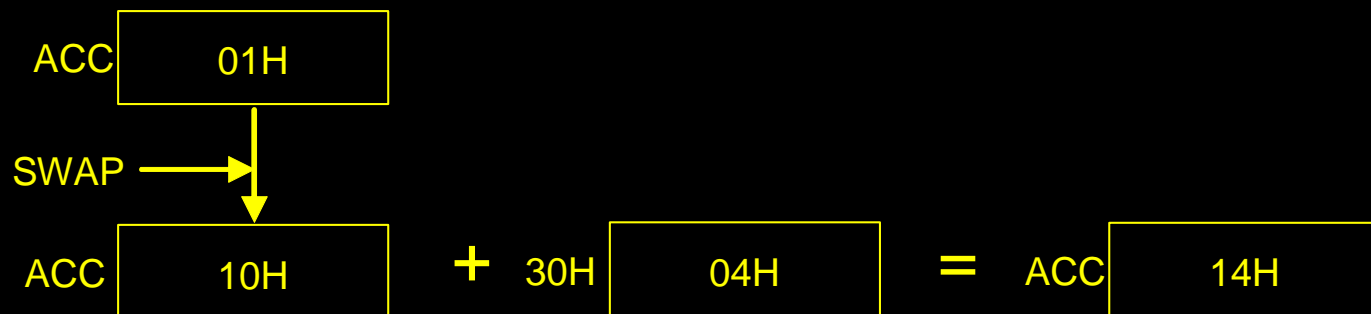
- 软件设计（程序解析）
 1. 计时提醒器接受键盘输入

CHK_0:

JB	P1.0,CHK_1	;判断按钮按下的JB
CALL	FILTER	;消除抖动子程序的CALL
JNB	P1.0, \$;判断按钮松开的JNB
MOV	R1, #0	
JMP	TIMERSET	

4.5 实例点拨——计时提醒器

2. 显示和存储所设定的时间



```

TIMERSET:      ;这是显示和存储所设定的时间的程序段
               DJNZ  R0, LSB      ; 如果 R0 不等于 0, 说明按键设置的是时间的低位
               JMP    MSB
LSB:           MOV    30H, R1      ; 把低位时间值存储到地址 30H 中
               JMP    SETTING     ; 跳回扫描数字键程序段, 继续扫描高位时间的输入
MSB:           MOV    A, R1        ; 把高位时间载入累加器 ACC 中
               SWAP    A           ; 累加器 ACC 的高 4 位与低 4 位互换
               ADD     A, 30H      ; 累加器 ACC 与地址 30H 中的数据相加, 得到两位时间值
               MOV     31H, A      ; 把设定好的时间保存到 31H 中
               MOV     P0, A       ; 把设定好的时间输出到七段数码管显示
  
```

4.5 实例点拨——计时提醒器

3. *号键确认，或“#”号键取消

CONFIRM: ;这是确认设置时间的程序段

OK:

JB P2.3,CANCEL ; 判断是否按下确认键"*"

CALL FILTER

JNB P2.3,\$

MOV A, #00H ; 计时初始值载入 ACC 中

JMP TIMING ; 如果确认，就跳到 TIMING 程序段开始计时

CANCEL:

JB P2.4,OK

CALL FILTER

JNB P2.4,\$

RESTART:

JMP START ; 如果按取消键"#", 则跳回一开始重新扫描设置时间

4.5 实例点拨——计时提醒器

4. 正常的计时阶段

TIMING:

```
        MOV        P0, A            ; 更新显示时间
MIN:
        MOV        R0, #60          ; 60 个 1 秒为 1 分钟, R0 作为分钟计数
SEC:
        MOV        R1, #10          ; 延时 1 秒
SEC_10:
        MOV        R2, #200         ; 延时 1/10 秒
        CALL       DELAY
        DJNZ       R1, SEC_10
        DJNZ       R0, SEC
        ADD        A, #1             ; 计时值加 1
        DA         A                ; 十进制调整
        CJNE       A, 31H, TIMING    ; 如果计时值不等于 31H 保存的设置值,
                                     就继续计时
```

4.5 实例点拨——计时提醒器

5. 蜂鸣器的“嘀、嘀……”提示音

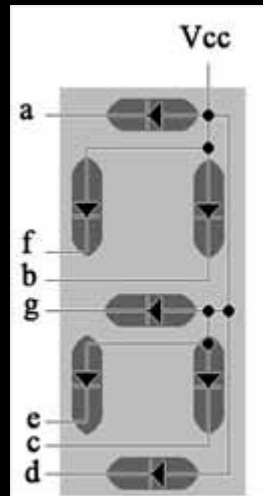
BEEP:

```
        MOV        R2, #5           ; 设置蜂鸣器鸣响 5 次
REPEAT:
        SETB       P2.7             ; 把 P2.7 脚置高, 蜂鸣器鸣响
        CALL        LONGDELAY
        CLR        P2.7             ; 把 P2.7 脚置低, 蜂鸣器关闭
        CALL        LONGDELAY
        DJNZ       R2, REPEAT
        JMP        START            ; 完成后跳回一开始扫描数字键
```

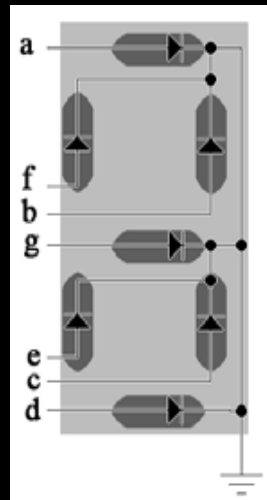

器件介绍

1. 七段数码管

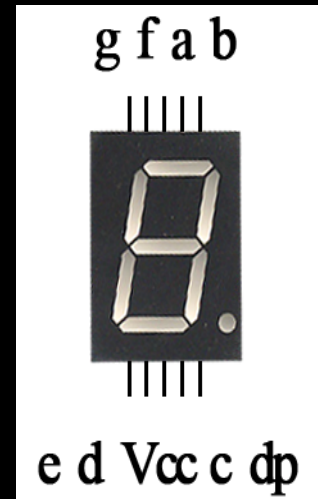
共阳



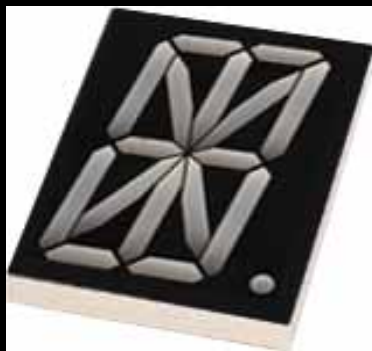
共阴



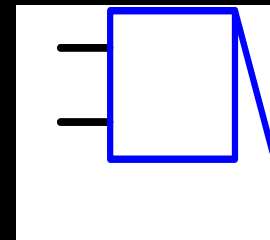
管脚排布



2. 米字数码管



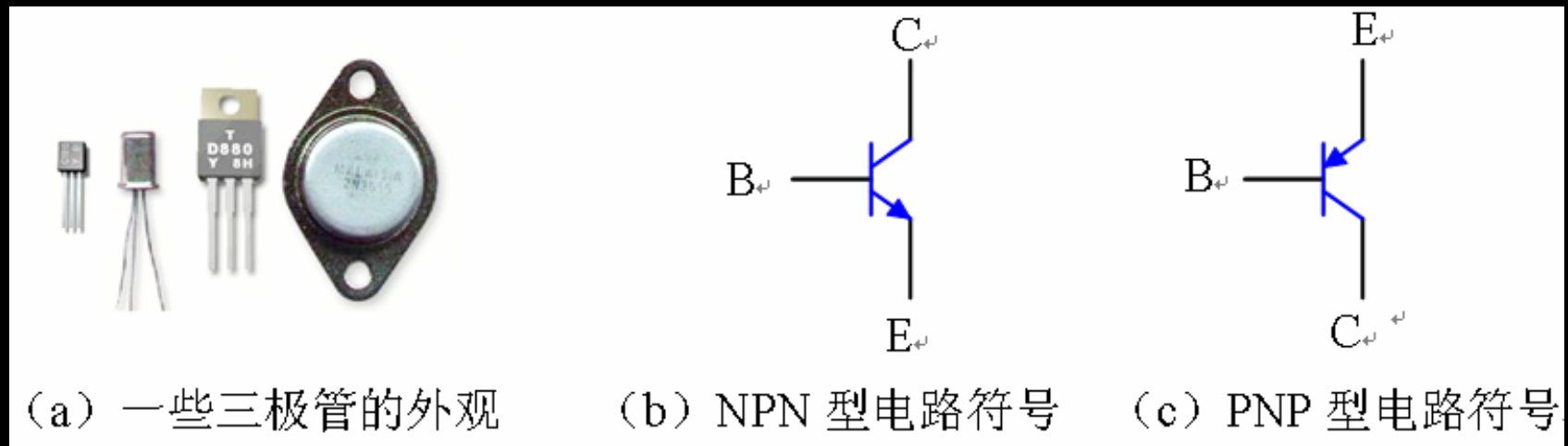
3. 蜂鸣器



蜂鸣器及其电路符号

器件介绍

4. 三极管



5. 三态缓冲器74125

