单片机应用技术（C51）

主讲：陈丽娟

项目二：花样流水LED灯设计

任务目标：

任务一：点亮单个LED
（认识单片机的输出端口、掌握C语言编程的基本思路）
任务二：用开关控制LED
（认识单片机的输入端口）
任务三：单个LED闪烁
（了解C语言的基本语句、函数）
任务四：花样流水灯设计
（掌握C语言的基本语句、函数）

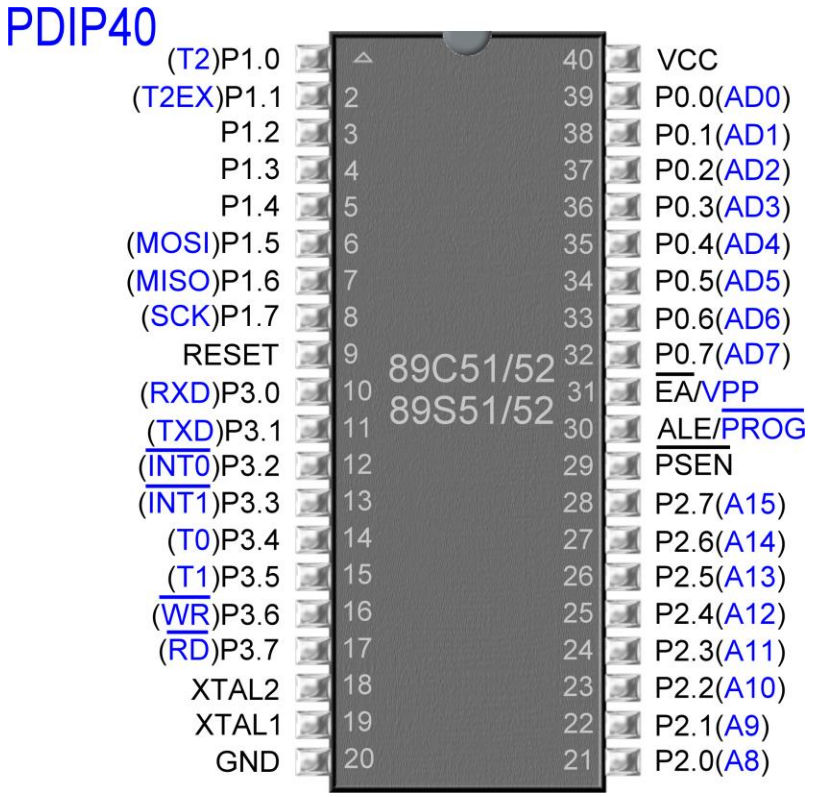
单片机的触角——I/O口

单片机管脚是控制外设或与外设交换信息的通道。

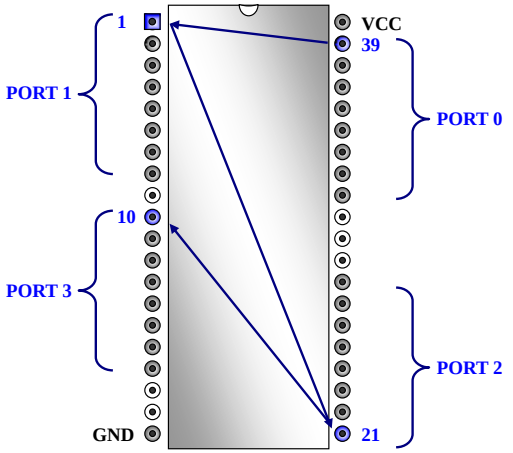
主要的外设有：发光二极管、开关、数码管、液晶屏、电机、继电器等。外设都是经单片机的管脚下工作。

I/O是输入/输出的意思，单片机的I/O口肩负着控制外设和接收信号的责任。

理解I/O口是如何在程序的控制下工作是学习单片机首要解决的问题。



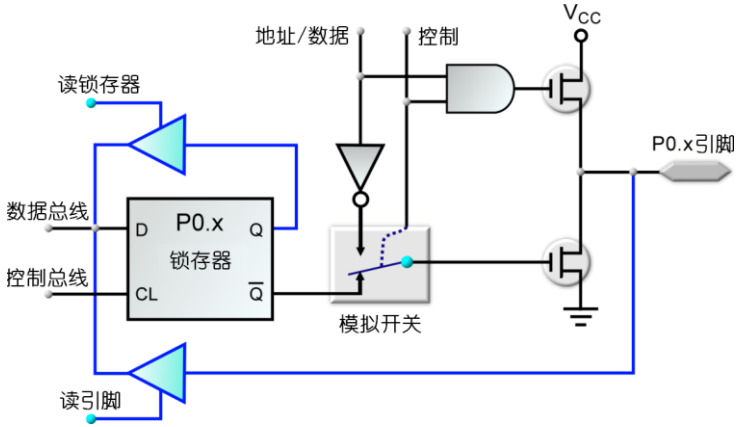
单片机的触角——I/O口　P0口



P0口（32~39管脚）

——是一个8位的开漏型双向I/O口。P0口在作输入/输出口使用时需要添加外部上拉电阻。

单片机上电复位时，P0口默认作为输出口，如果需要作输入口使用，需要先用程序向每个I/O口写入。



PORT 0 内部电路结构（1 位）

单片机的触角——I/O口　P0口

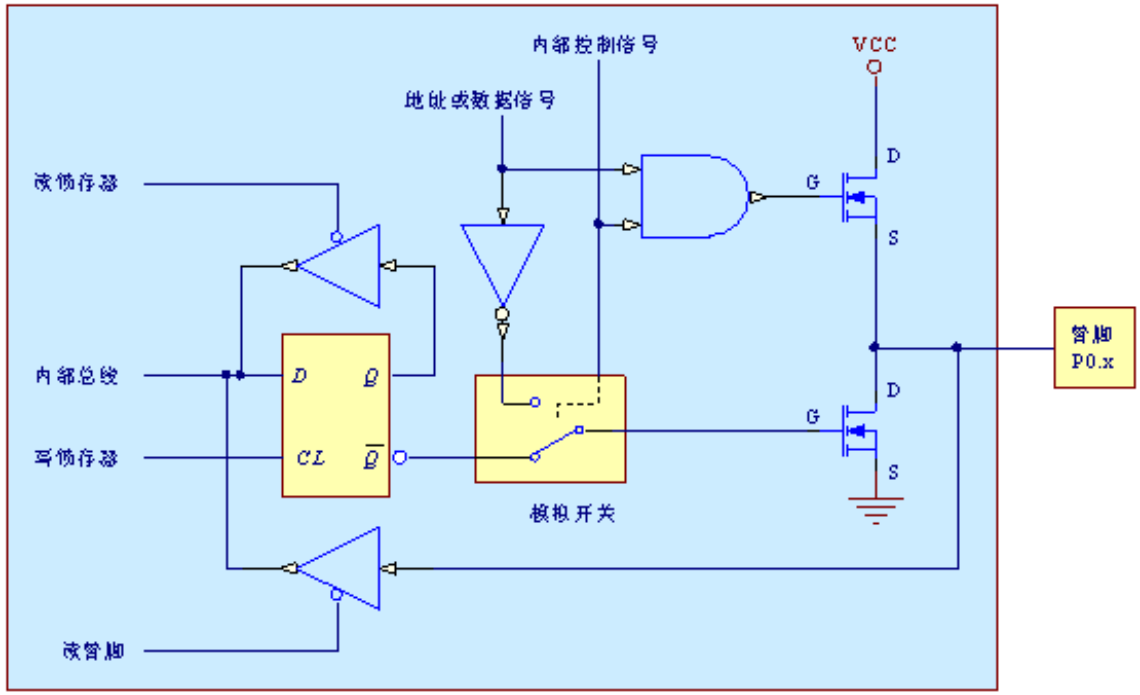
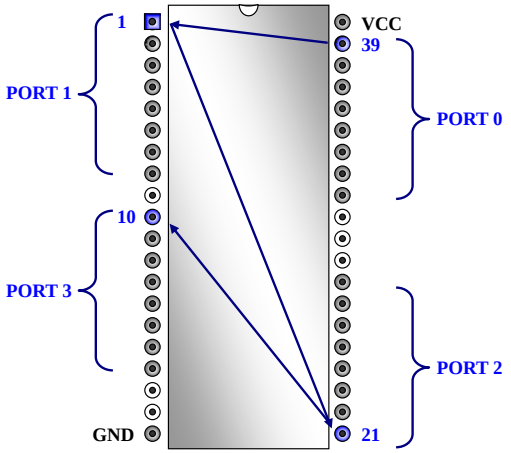


图 5-19 P0 口结构

P0中的模拟开关，由“内部控制信号”控制，以选通P0口所要交换的是一般I/O口数据或是访问外部存储器的地址/数据。

由于P0口是没有内部上拉电阻的，所以它作一般I/O口使用时需要添加外部上拉电阻。不过在访问外部存储器时上拉FET（受与门控制的FET）会导通而不需要再添加外部上拉电阻。

单片机的触角——I/O口　P1口

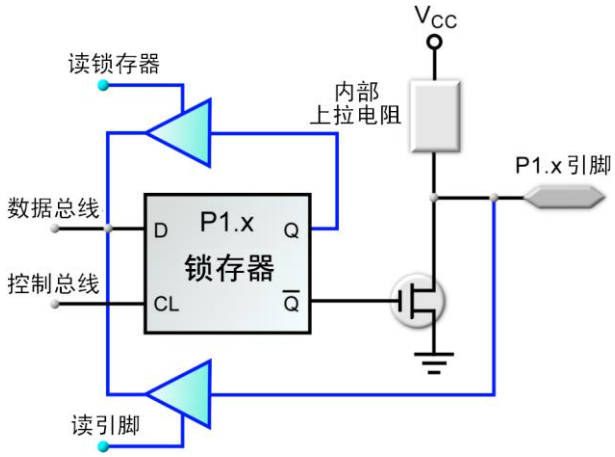


P1口（1~8管脚）

——是一组带内部上拉电阻的双向I/O口，由于P1口内置有上拉电阻，于是在作输入/输出口时不再需要添加外置上拉电阻。

作输入口时，也需要向每位写入1。

P1.5、P1.6、P1.7除作一般I/O口外，还作为下载接口用于向单片机下载程序。



PORT 1 内部电路结构（1 位）

单片机的触角——I/O口　P1口

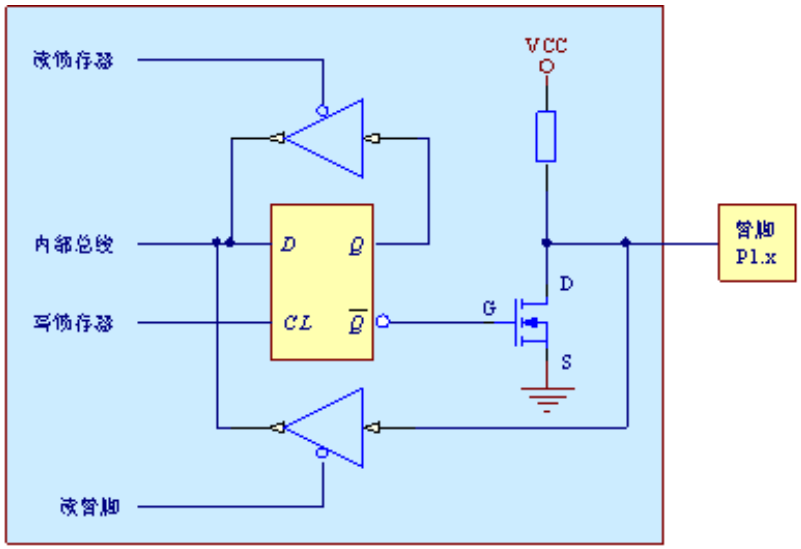


图 5-17　P1 口结构

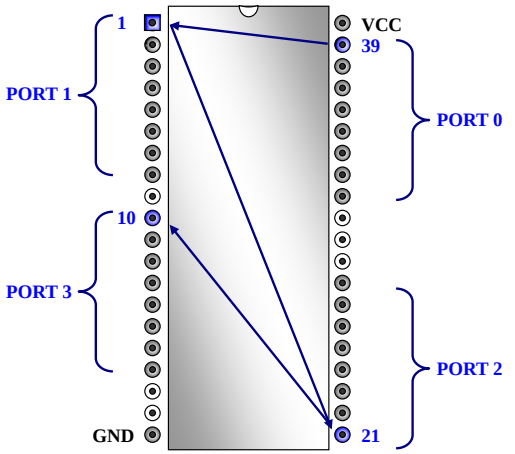
P1口主要由D锁存器、两个缓冲器、场效应管、内部上拉电阻等组成。

当程序让该I/O口输出1时，“内部总线”出现1，当“写锁存器”信号到来时，输出端*Q*就输出1，*/Q*输出0。于是FET截止，于是“管脚P1.x”因电阻的上拉也输出1。

当程序让该I/O口输出0时，“内部总线”出现0使/Q =1，于是FET导通，“管脚P1.x”接地而呈现0。

如果I/O口作输入时，与“管脚P1.x”相连的外设使该管脚出现1或0，程序控制“读管脚”使能缓冲器，则“管脚P1.x”的状态就通过缓冲器进入“内部总线”上，指令就可接收到“内部总线”上的数据了。

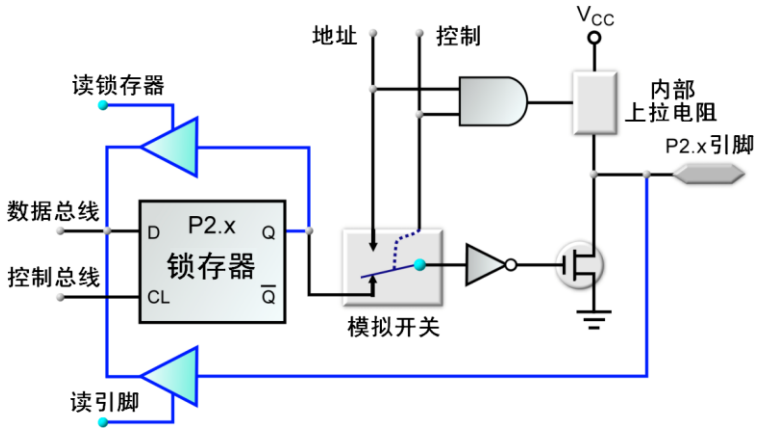
单片机的触角——I/O口　P2口



P2口（21~28管脚）

——一组带内部上拉电阻的双向I/O口。

由于P2口内置有上拉电阻，于是在作输入/输出口时不再需要添加外置上拉电阻。当P2口作输入时，需要写入1。



PORT 2 内部电路结构（1位）

单片机的触角——I/O口　P2口

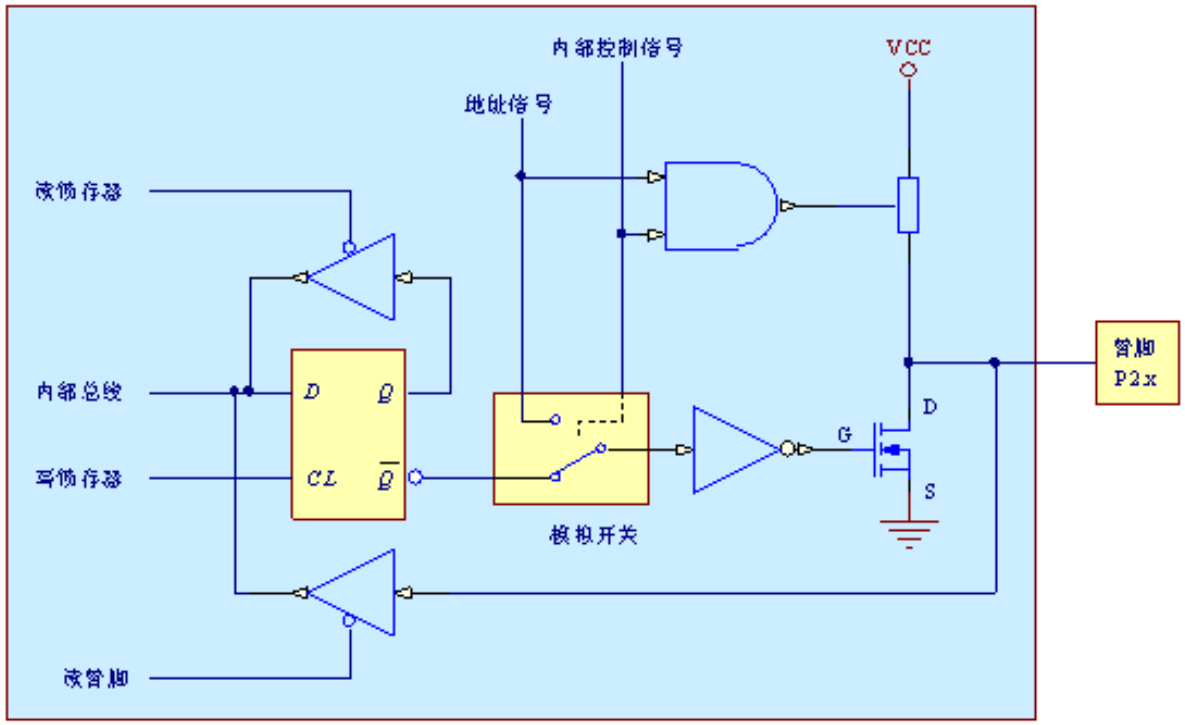


图 5-20 P2 口结构

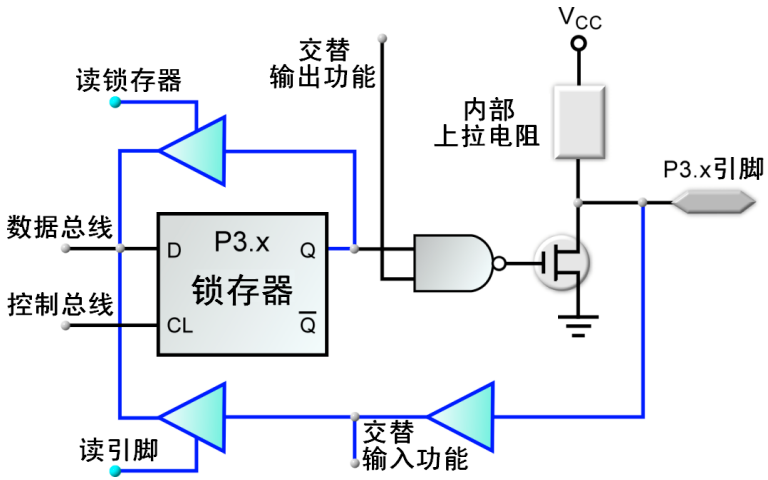
P2口由于在访问外部存储器时作为高位地址使用，所以它与P0口结构相似。它也由“内部控制信号”控制模拟开关的切换从而实现P2口的功能转换。只不过P2口具有内部上拉电阻，在作一般I/O口使用时不需要外部再添加。

单片机的触角——I/O口　P3口

P3口（10~17管脚）

同样是一组带内部上拉电阻的双向I/O口。由于P3口内置有上拉电阻，于是在作输入/输出口时不再需要添加外置上拉电阻。当P3口作输入时，需要写入1。

PORT 3	其它功能	说　　明
P3.0	**RXD**	串行口的接收引脚
P3.1	**TXD**	串行口的传送引脚
P3.2	**INT0**	INT0 中断输入
P3.3	**INT1**	INT1 中断输入
P3.4	**T0**	Timer/Counter 0 输入
P3.5	**T1**	Timer/Counter 1 输入
P3.6	**WR**	写入外部存储器控制引脚
P3.7	**RD**	读取外部存储器控制引脚



PORT 3 内部电路结构（1 位）

单片机的触角——I/O口　P3口

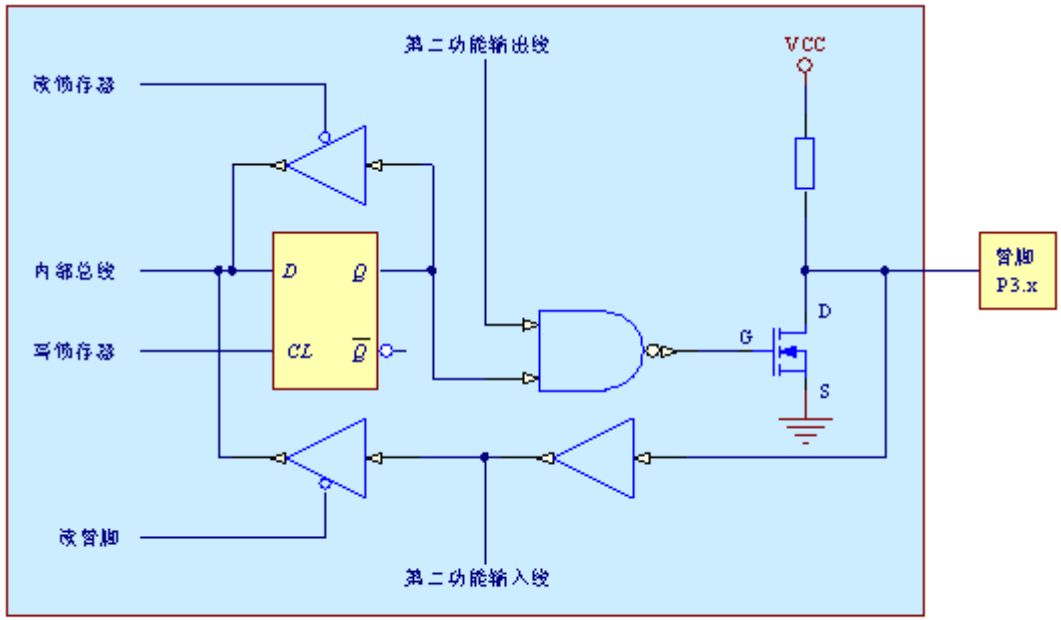


图 5-18　P3 口结构

P3口比P1口多出了“第二功能输入线”和“第二功能输出线”。“第二功能输出线”与锁存器的输出端Q通过一个与非门和FET的G极连接。这样，如果“内部总线”=1时，与非门的输出端与“第二功能输出线”相反。比如“第二功能输出线”=1，FET的G极为0，所以FET截止，“管脚P3.x”=1。

当P3口作第二功能输入时，“管脚P3.x”上的信号通过一个缓冲器出现在“第二功能输入线”，程序只要把这个信号读走即得到第二功能的输入数据。

单片机的触角——I/O口

I/O口小结

•每一个I/O口都可以独立地作输入或输出口使用，但P0和P2在访问外部存储器时作地址/数据总线，此时它们将不能再作为I/O口使用。

•AT89S51单片机复位时每一个I/O口的“内部总线”=1，如果随后程序使“内部总线”=0，那么当I/O口作为输入时，必须通过程序通过输出1使FET截止，这样从“管脚Px.x”输入的信号才能在“读管脚”信号的帮助下被正确读走。

•P1、P2、P3因为内部上拉电阻而被称为“准双向口”。在作输入时，上拉电阻将“管脚Px.x”拉高并在外设输入低电平时向外输出电流。

•P0口没有内部上拉电阻，是一个真正的双向口。作输入时因开漏结构而浮地。

任务一：用单片机点亮单个LED灯

基本步骤：

1、需求分析：单片机上电，发光二极管导通。

2、电路设计（硬件）

3、程序设计（软件）

4、系统调试

任务一：单片机点亮单个LED灯

2、电路设计（硬件）

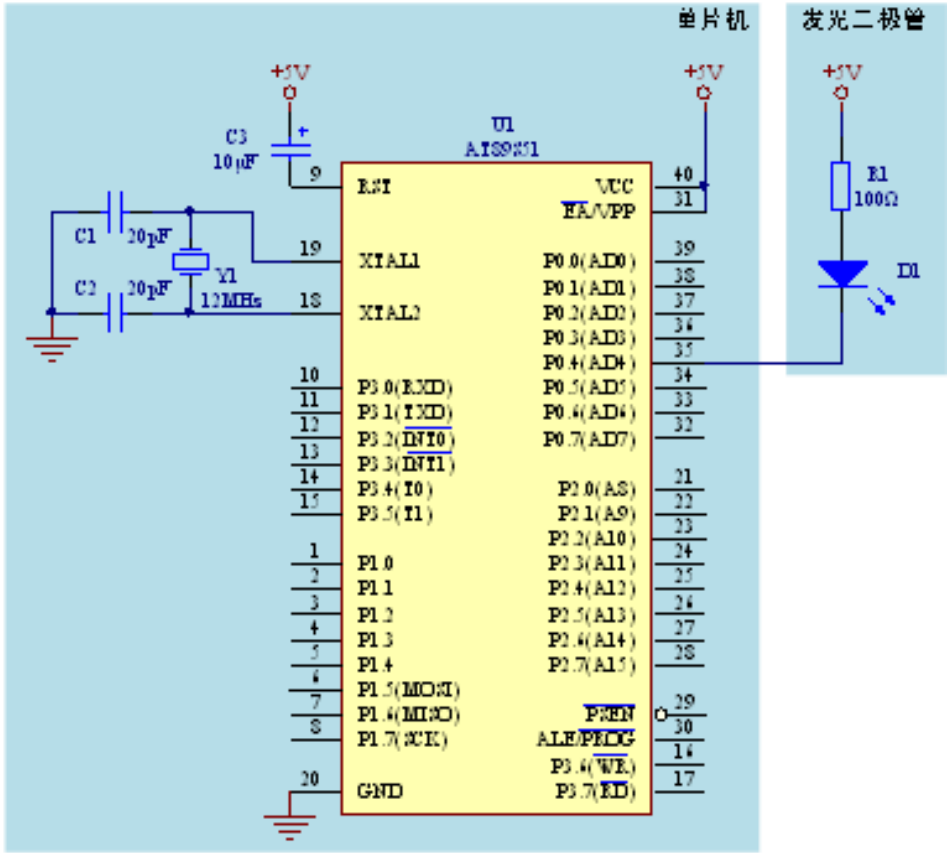
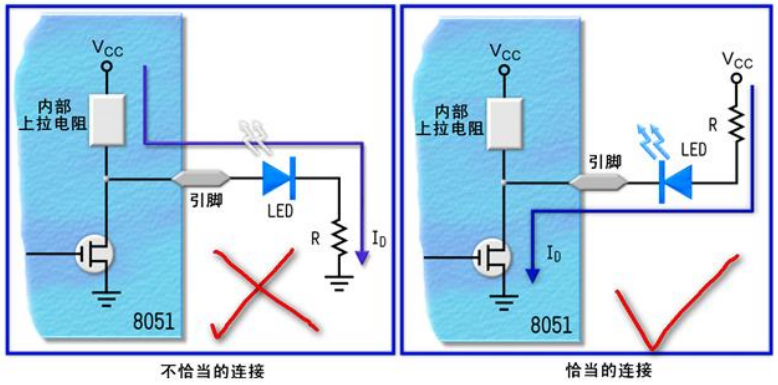


图 2-19 单片机控制一个发光二极管的电路图

分配好I/O端口，在单片机最简系统基础上增加输出对象：发光二极管



输出 LED 的电路连接方式

任务一：单片机点亮单个LED灯

3、程序设计（软件）

```
#include <reg51.h>
sbit LED=P0^4;
main()
{
        LED=0;
}
```

4、系统调试：使用protues、keil vision进行联调。

任务二：开关控制LED灯

1、需求分析：按下开关，灯亮；不按开关，灯不亮。

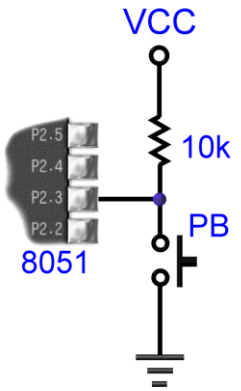
2、电路设计（硬件）：

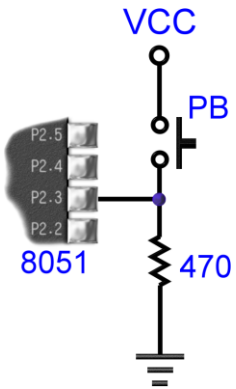
3、程序设计（软件）：

4、系统调试

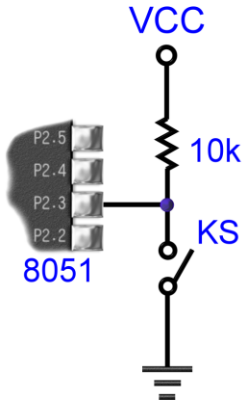
任务二：开关控制LED灯

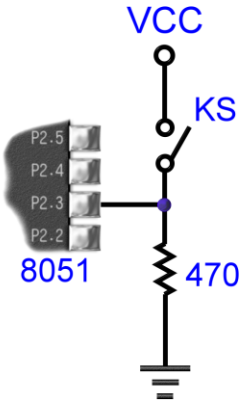
2、电路设计（硬件）：输入部分电路设计





按钮开关输入电路

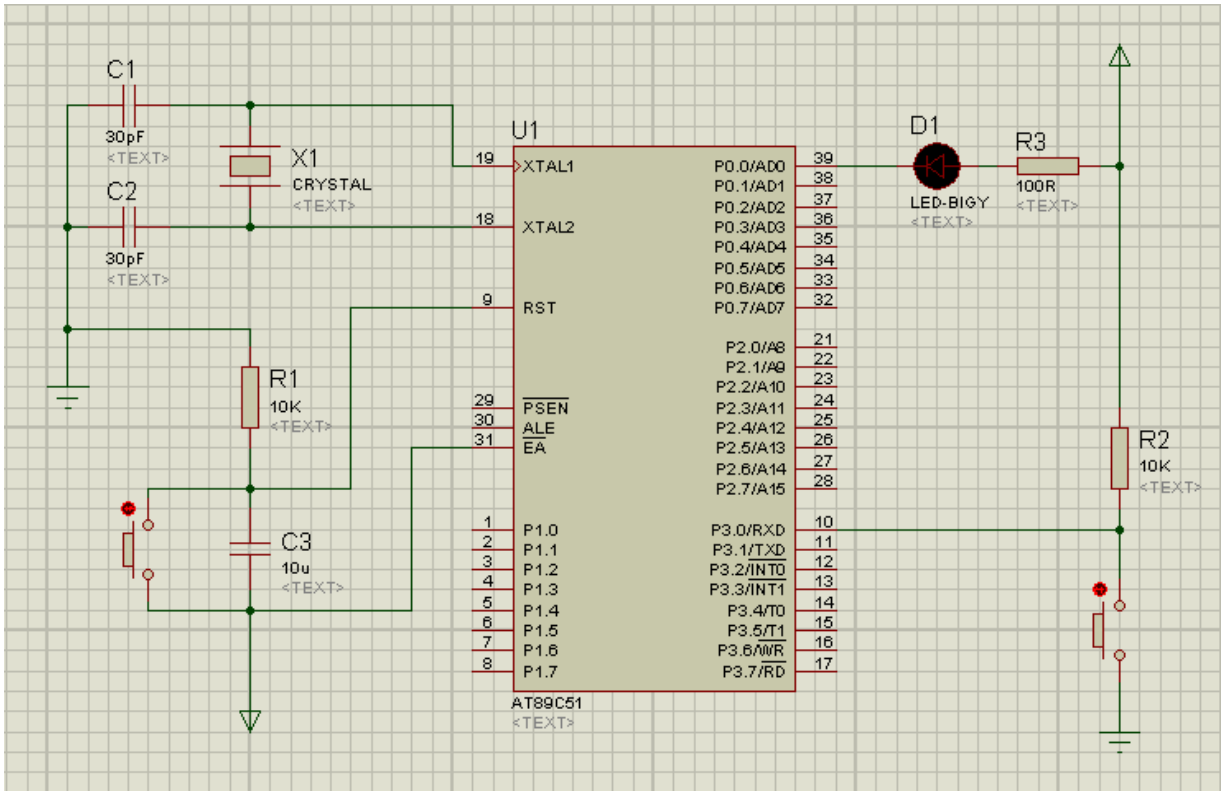




闸刀开关输入电路

任务二：开关控制LED灯

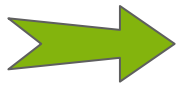
2、电路设计（硬件）：



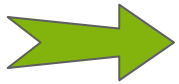
任务二：开关控制LED灯

3、程序设计（软件）：编程思路

如果按下开关，灯亮；
如果不按开关，灯不亮。



如果K1=1；
那么LED=0；
如果K1=0；
那么LED=1；



```
if (K1==1)
{LED=0 ;}
else
{LED=1 ;}
```

任务二：开关控制LED灯

3、程序设计（软件）：完整程序

```
#include<reg51.h> //定义reg51.h头文件。
sbit led=P1^0;    //定义LED位置，即把LED灯接到单片机的P1.0端口
sbit k1=P3^0;       // 定义开关的位置，即把开关K1接到单片机的P3.0端口
Main()              //主程序开始，
{                   //大括号，跟最后一个大括号配合，所有程序都被这一对括号括起来
    If (K1==1)      // 判断开关K1是不是被按下
    {LED=0 ;}       // 如果开关K1被按下，LED灯亮，采用低电平导通方式
    Else            // 否则，即：开关没有被按下
    {LED=1 ;}       // 那么led灯不亮
}                   // 主程序结束。
```

任务二：开关控制LED灯

4、系统调试：使用protues、keil vision进行联调。

任务三：单个LED闪烁

1、需求分析：让LED每隔一段时间闪烁一次

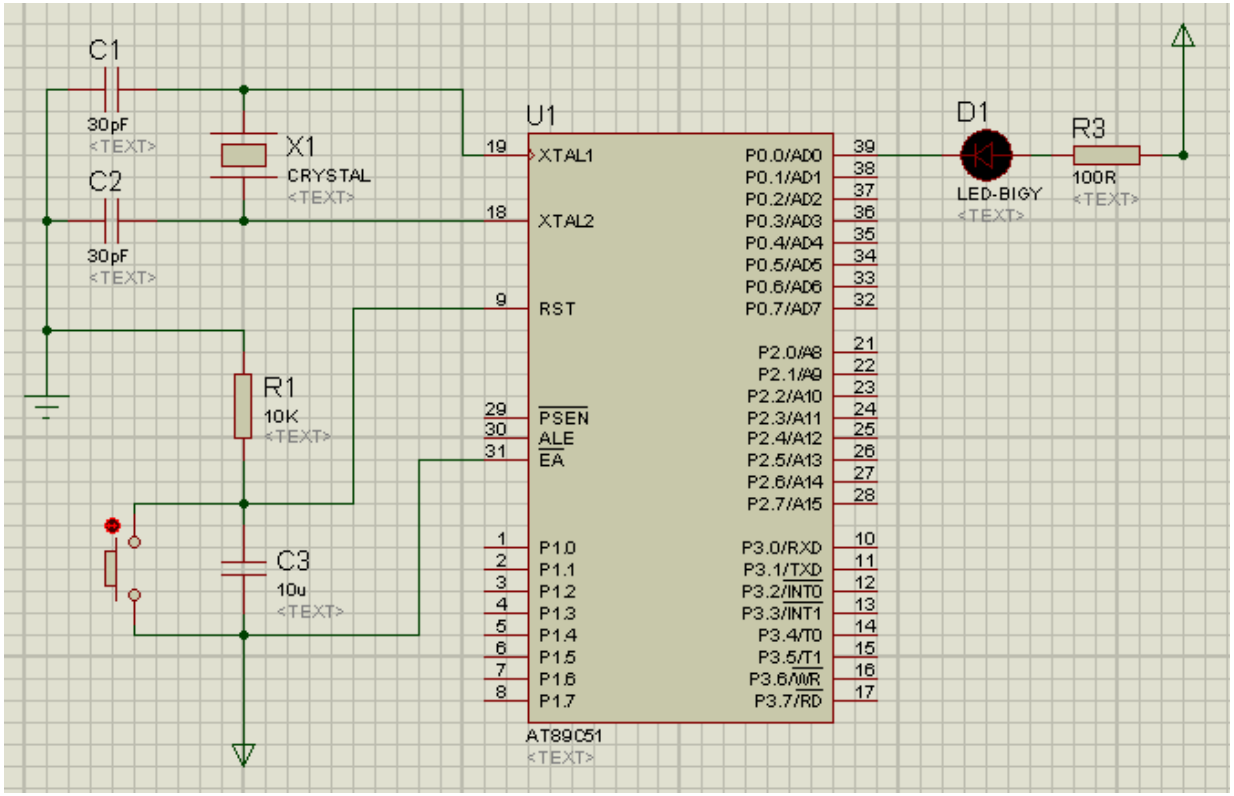
2、电路设计（硬件）：

3、程序设计（软件）：

4、系统调试

任务三：单个LED闪烁

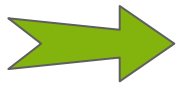
2、电路设计（硬件）：



任务三：单个LED闪烁

3、程序设计（软件）：编程思路

第一步程序开始，led 亮，
过一段时间，led熄灭。
又过一段时间，led亮
再过一段时间，led 熄灭。
无限循环



P1.0=0;
过一段时间；
P1.0=1;
又过一段时间；
P1.0=0;
无限循环；

任务三：单个LED闪烁

3、程序设计（软件）：

实现单片机的延时方法

（1）、for语句

（2）、延时函数

a、不带参数的延时函数

b、带参数的延时函数

实现单片机的循环方法

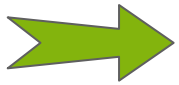
while语句

任务三：单个LED闪烁

3、程序设计（软件）：方法一　for语句实现延时

格式：for（i=0;i<10000;i++）；

```
P1.0=0;
过一段时间；
P1.0=1;
又过一段时间；
P1.0=0;
无限循环；
```



```
led=0;
for（i=0;i<10000;i++）；；
led=1;
for（i=0;i<10000;i++）；
led=0;
for（i=0;i<10000;i++）；
无限循坏；
```

任务三：单个LED闪烁

3、程序设计（软件）：方法一　for语句实现延时

```
#include<reg51.h>   //定义reg51.h头文件。
sbit led=P0^0;      //定义LED位置，把LED灯接到P0.0端口
main ()             //主程序开始，
{
int i;              //定义变量i
while(1)            //无限循环
{  led=0;           // 点亮led灯
for（i=0;i<10000;i++）;  // 延时约0.1s
led=1;              // 熄灭led
for（i=0;i<10000;i++）; // 延时约0.1s
}                   // while无限循环结束
}                   //主程序结束
```

任务三：单个LED闪烁

3、程序设计（软件）：方法二　延时函数实现延时

函数：是一种独立功能的程序。

不带参数的延时函数

```
delay()
{
    int i;
    for(i=0;i<10000;i++);
}
```

函数的定义

```
main ()
{
   while(1)
   {    led=0;
        delay ();
        led=1;
        delay();
    }
}
```

函数的调用

任务三：单个LED闪烁

3、程序设计（软件）：方法二 延时函数实现延时

```
#include<reg51.h>     //定义reg51.h头文件。
sbit led=P0^0;        //定义LED位置，即把LED灯接到单片机的P0.0端口
delay（int）;         //声明延时函数
main ()               //主程序开始，
{
    while(1)          //无限循环
    {    led=0;       // 点亮led灯
         delay ();    // 延时
         led=1;       //熄灭led灯
         delay();     // 延时
     }                //结束无限循环
}                     //主程序结束
delay()           //  延时函数
{
    int i;                          //定义整型变量
    for(i=0;i<10000;i++);           //计数10000次
}
```

不带参数的延时函数

任务三：单个LED闪烁

3、程序设计（软件）：方法二　延时函数实现延时

函数：是一种独立功能的程序。

带参数的延时函数

```
delay (int x)
{
        int i;
        for(i=0;i<x;i++);
}
```

函数的定义

```
main ()
{
   while(1)
   {    led=0;
        delay (10000);
        led=1;
        delay(5000);
    }
}
```

函数的调用

任务三：单个LED闪烁

3、程序设计（软件）：方法二 延时函数实现延时

```
#include<reg51.h>     //定义reg51.h头文件。
sbit led=P0^0;        //定义LED位置，即把LED灯接到单片机的P0.0端口
delay（int x）;        //声明延时函数
main ()               //主程序开始，
{
    while(1)                    //无限循环
    {     led=0;                // 点亮led灯
          delay (10000);        // 延时
          led=1;                //熄灭led灯
          delay(5000);          // 延时
     }                          //结束无限循环
}                               //主程序结束
delay(int x)                    // 延时函数
{
    int i;                      //定义整型变量
    for(i=0;i<x;i++);           //计数x次
}
```

带参数的延时函数

任务三：单个LED闪烁

4、系统调试：使用protues、keil vision进行联调。

专题一　Keil C语言

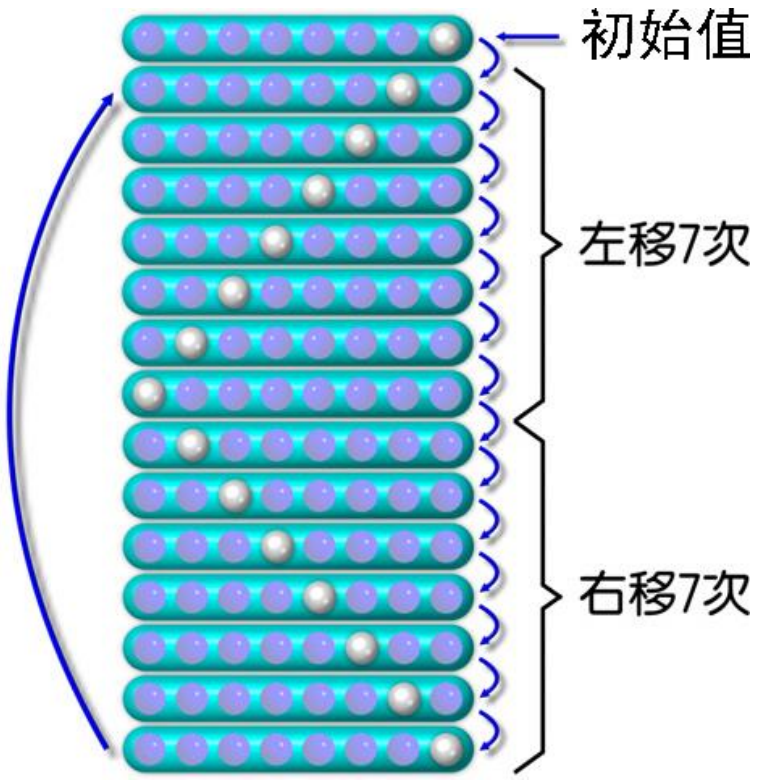
任务四：花样流水灯设计

1、需求分析：霹雳灯

2、电路设计（硬件）：

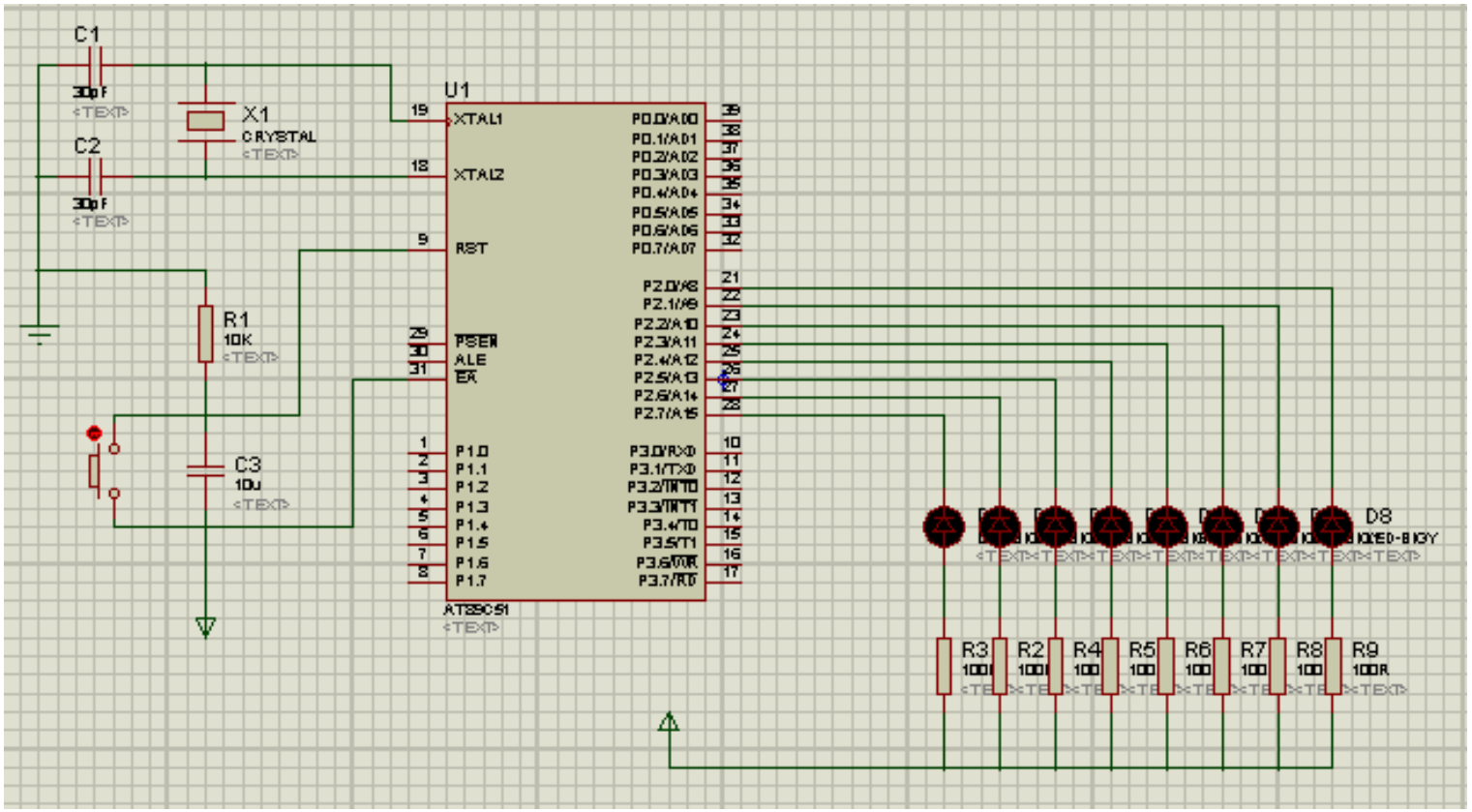
3、程序设计（软件）：

4、系统调试



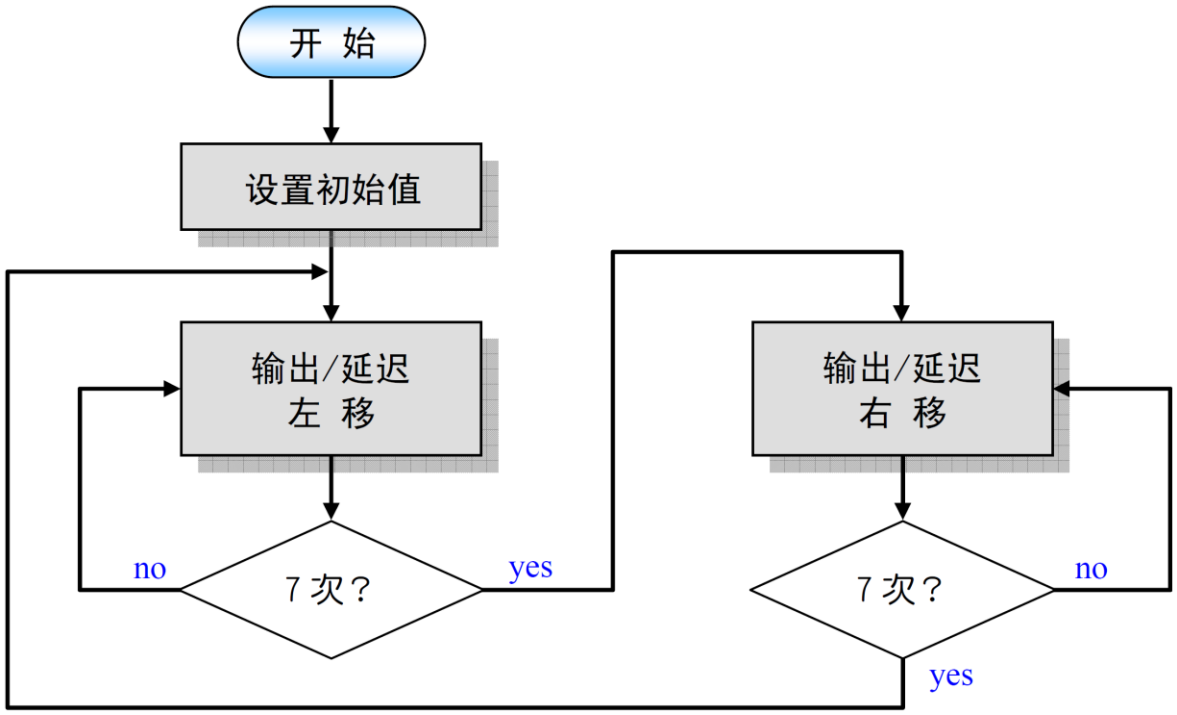
任务四：花样流水灯设计

2、电路设计（硬件）：



任务四：花样流水灯设计

3、程序设计（软件）：程序流程图



任务四：花样流水灯设计

3、程序设计（软件）：

```
/* ch03-3-3.c - 霹雳灯实验程序   */
//==声明区=============================================
#include <reg51.h>                    //  定义8051寄存器的头文件
#define  LED     P2                   //  定义LED接至Port 1
void delay（int）;                    // 声明延迟函数
//==主程序=============================================
main（）                              // 主程序开始
{ unsigned char i;                    // 声明无符号变量i（占1Bytes）
  LED=0xfe;                           // 初值=1111 1110,只有最右1 灯亮
  while（1）                          // 无穷循环,程序一直跑
  { for（i=0;i<7;i++）                 // 左移7次
    { delay（100）;                   // 延迟100×5m=0.5s
      LED=（LED<<1）|0x01;            // 左移1位,并设置最低位为1
    }                                 // 左移结束,只有最左1灯亮
    for（i=0;i<7;i++）                // 右移7次
      { delay（100）;                 // 延迟100×5m=0.5s
        LED=（LED>>1）|0x80;          // 右移1位,并设置最高位为1
      }                               // 结束右移,只有最右1灯亮
  }                                   // while循环结束
}                                     // 主程序结束
```

任务四：花样流水灯设计

3、程序设计（软件）：

```
//==子程序==========================================
/* 延迟函数,延迟约x×5ms */
void delay（int x）                    // 延迟函数开始
  { int i,j;                           // 声明整数变量i,j
      for （i=0;i<x;i++）              // 计数x次,延迟x×5ms
        for （j=0;j<600;j++）;         // 计数600次，延迟5ms
  }                                    // 延迟函数结束
```

思考：如何修改程序，使它编程双灯的霹雳灯？

任务四：花样流水灯设计——跑马灯

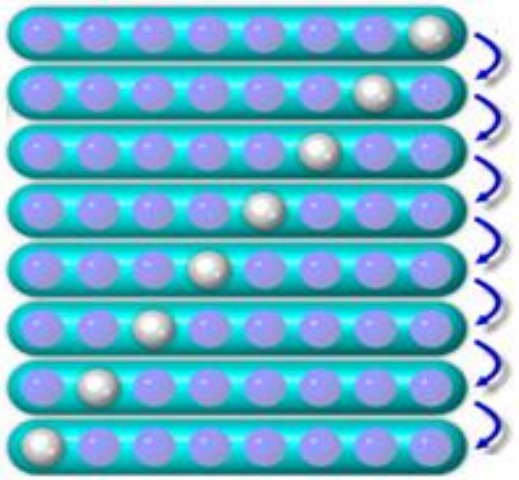
思考题：

1、需求分析：

2、电路设计（硬件）：

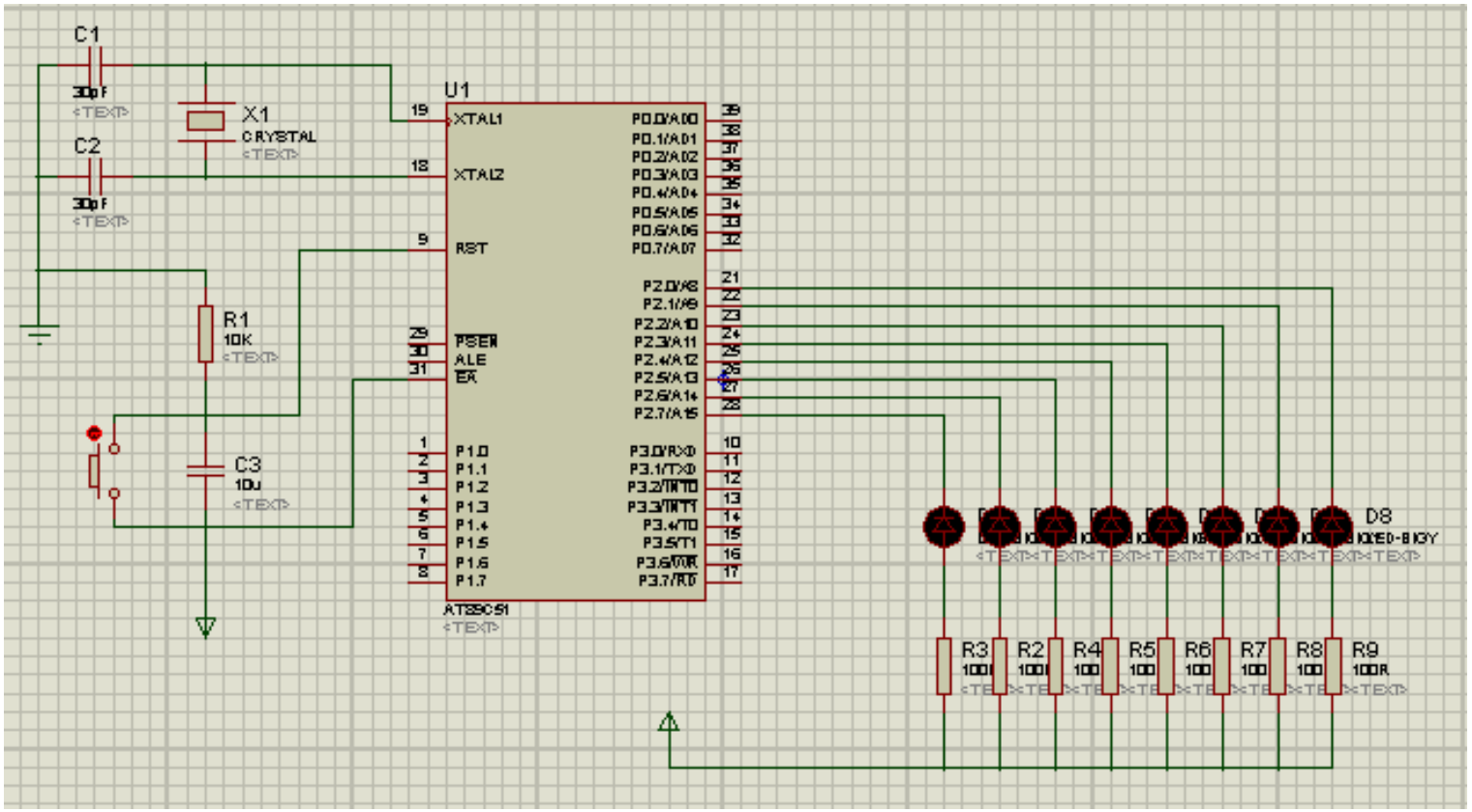
3、程序设计（软件）：

4、系统调试



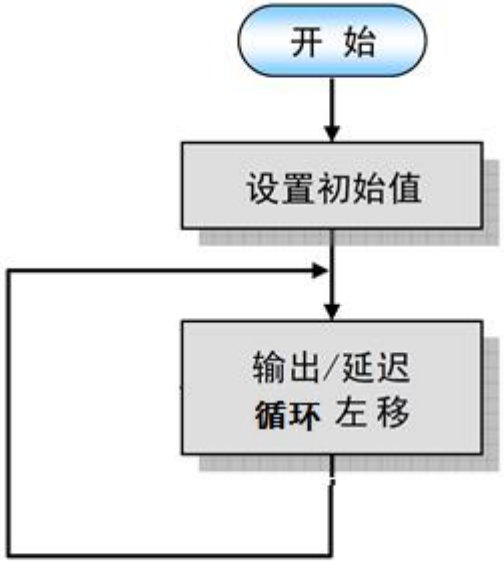
任务四：花样流水灯设计——跑马灯：

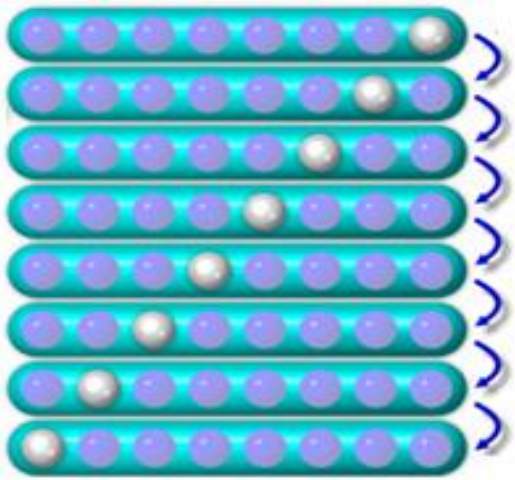
2、电路设计（硬件）：



任务四：花样流水灯设计

3、程序设计（软件）：程序流程图





任务四：花样流水灯设计——跑马灯：

3、程序设计（软件）：

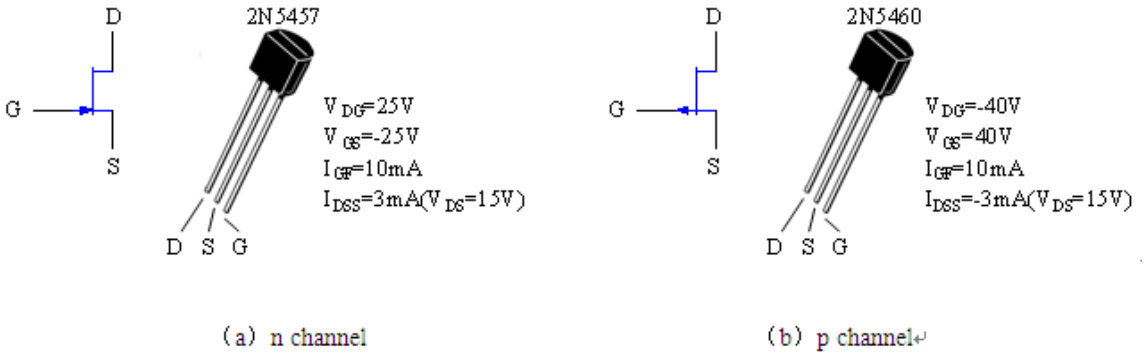
```
#include <reg51.h>
#include <intrins.h>
void Delay10ms(unsigned int c);
void main(void)
{
    unsigned char LED;
    LED = 0x01;
    while(1)
    {
        P2 = LED;
        Delay10ms(50);
        LED = _crol_(LED,1);
    }
}
```

主函数

```
void Delay10ms(unsigned int c)
{
    unsigned char a, b;
    for (;c>0;c--)
    {
        for (b=38;b>0;b--)
        {
            for (a=130;a>0;a--);
        }
    }
}
```

延时函数

补充知识：场效应管



场效应管有两类，一类称为结型场效应管，英文缩写JFET。另一类称为金属氧化物半导体场效应管，英文缩写MOSFET。图示为n channel和p channel两种JFET的电路符号和代表器件的外观，JFET有三个管脚——D极（漏极）、S极（源极）、G极（栅极）。

补充知识：场效应管

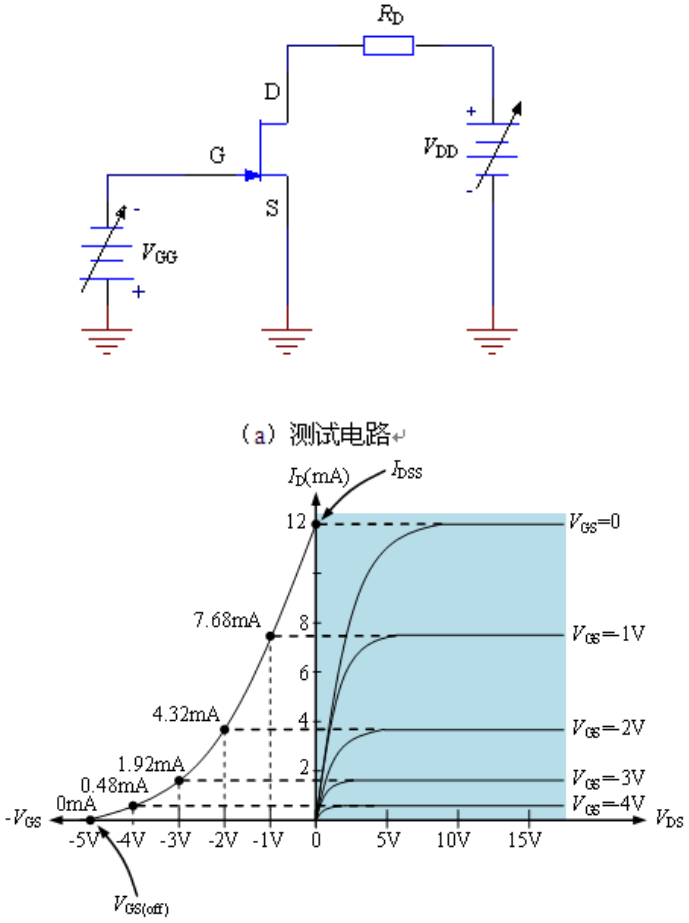


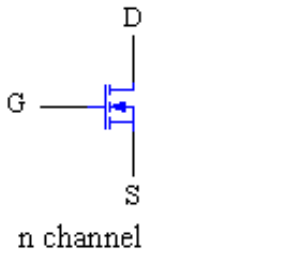
图 5-5 n-channel 型 JFET 的特性曲线

JFET

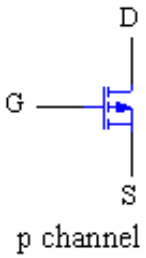
首先，调节电源V_{GG}电压至0V，相当于JFET的G极接地，G极与S极之间的电压V_{GS}=0。此时增加D极电压V_{DD}，也就是增加JFET的D极与S极之间的电压V_{DS}。在增加的过程中，流过D极的电流I_D渐渐增大，当I_D增大到12mA时就不再增大，如图所示的I_D=I_{DSS}对应右侧的曲线。至于I_{DSS}的数值可从器件的技术手册获得。

当V_{GG}不等于0，即V_{GS}不等于0，对应图左侧曲线部分，发现随着V_{GS}的增大，JFET的D极电流I_D在减小。所以对于JFET来说，V_{GS}控制着D极电流I_D，且V_{GS}越大，I_D越小。

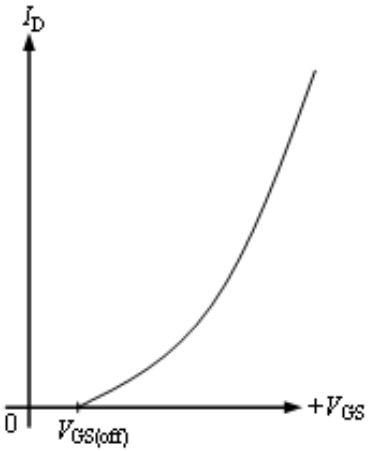
补充知识：场效应管

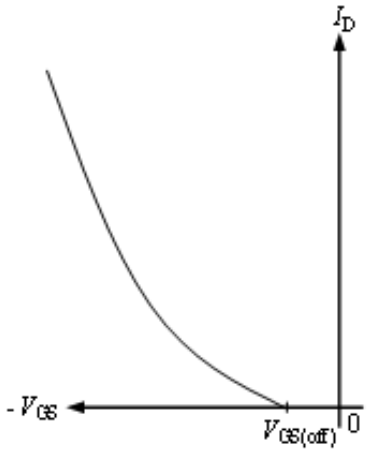


n channel



p channel





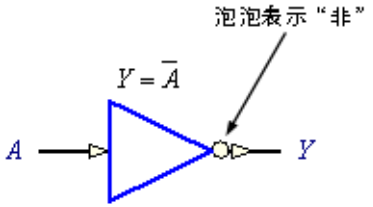
MOSFET

MOSFET是另一类场效应管。MOSFET分为耗尽型MOSFET（英文缩写D-MOSFET）和增强型MOSFET（英文缩写E-MOSFET）两种，MOSFET也有三个管脚——D极（漏极）、S极（源极）、G极（栅极）。

对于n channel的FET来说，当G极得到高电平时产生D极电流I_D，即FET导通；当G极为低电平时截止。而p channel的FET正好相反，当G极为低电平时产生D极电流I_D，即FET导通；当G极为高电平时截止。

补充知识：逻辑门

1 非门

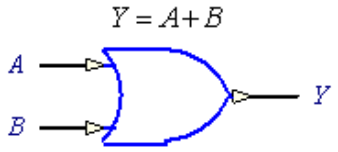


真值表

输入 A	输出 Y
0	1
1	0

图 5-7 非门

2 或门

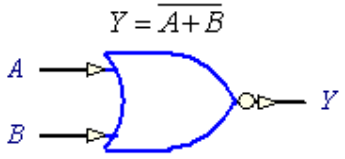


真值表

输入 A	输入 B	输出 Y
0	0	0
0	1	1
1	0	1
1	1	1

图 5-8 或门

3 或非门

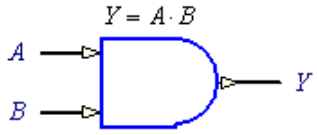


真值表

输入 A	输入 B	输出 Y
0	0	1
0	1	0
1	0	0
1	1	0

图 5-9 或非门

4 与门



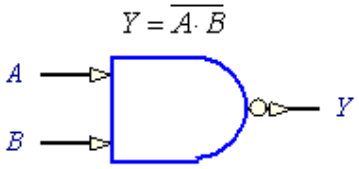
真值表

输入 A	输入 B	输出 Y
0	0	0
0	1	0
1	0	0
1	1	1

图 5-10 与门

补充知识：逻辑门

5 与非门



真值表

输入 A	输入 B	输出 Y
0	0	1
0	1	1
1	0	1
1	1	0

图 5-11　与非门

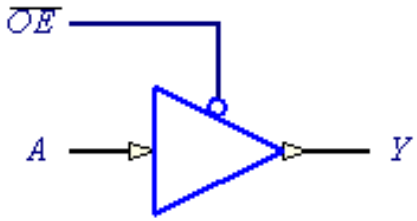
6 异或门

真值表

输入 A	输入 B	输出 Y
0	0	0
0	1	1
1	0	1
1	1	0

图 5-12　异或门

7 缓冲器



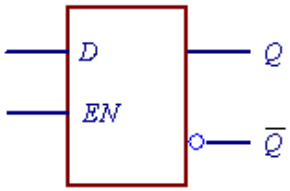
真值表

使能 $\overline{OE}$	输入 A	输出 Y
0	0	0
0	1	1

图 5-13　缓冲器

补充知识：逻辑门

8 门控D锁存器



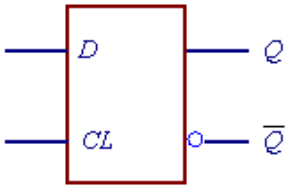
真值表

输入		输出		说明
D	*EN*	*Q*	$\overline{Q}$	
0	1	0	1	复位
1	1	1	0	置位
X	0	Q_0	$\overline{Q}_0$	不变

图 5-14 门控 D 锁存器

门控D锁存器：*EN*端是门控端，高电平使能。*D*为输入端，当*EN*＝1时，锁存器的输出端*Q*与输入端*D*状态相同。当*EN*=0时，输出*Q*将维持原来的状态不变。

9 边沿D触发器



真值表

输入		输出		说明
D	*CL*	*Q*	$\overline{Q}$	
1	↑	1	0	置位
0	↑	0	1	复位

图 5-15 边沿 D 触发器

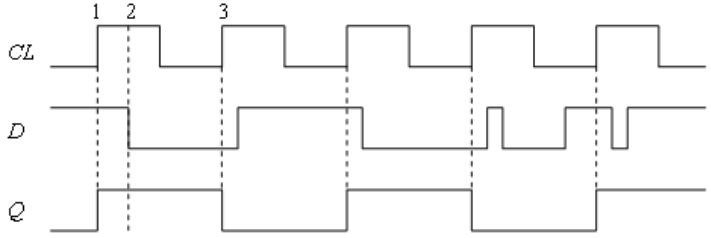


图 5-16 边沿 D 触发器的输入输出波形