第4章 并行接口系统

内容提要

★单片机的并行接口P0～P3

★ P0～P3端口的功能和内部结构

★ P0～P3端口的编程

★ 用并行口设计LED数码显示和键盘电路

★ 并行接口小结

由于CPU的数据线是外设或存贮器和CPU进行数据传输的唯一公共通道，为了使数据线的使用对象不产生使用总线的冲突，以及快速的CPU和慢速的外设时间上协调，CPU和外设之间必须有接口电路（简称接口或I/O口），接口起着缓冲、锁存数据，地址译码、信息格式转换、传递状态（外设状态），发布命令等功能。

I/O接口有并行接口、串行接口、定时/计数器、A/D、D/A等，根据外设的不同情况和要求选择不同的接口。本章介绍并行接口，用于和外设的并行数据通信。

4.1单片机的并行接口P0～P3

MCS-51单片机有P0、P1、P2、P3四个8位双向I/O口，每个端口可以按字节输入或输出，也可以按位进行输入或输出，四个口共32根口线，用作位控制十分方便。P0口为三态双向口，能带8个TTL电路；P1、P2、P3口为准双向口，负载能力为4个TTL电路。

4.1.1 P0～P3端口的功能和内部结构

4.1.1.1 P0～P3接口功能

大多数口线都有双重功能，具体介绍如下：

P0口—1. 作为输入/输出口。

2. 作为地址/数据总线 ，接外围芯片时P0口分时输出低 8 位地址与数据信号。

P1口—1. 作为输入/输出口。

2. 作为SPI接口的管脚及T2输入管脚等

P2口— 1. 作为输入/输出口。

2. 作为高8位地址总线。

P3口—P3口为双功能

1. 作第一功能使用时，其功能为输入/输出口。

2. 作第二功能使用时，每一位功能定义如下表所示：

端口引脚	第 二 功 能
P3.0	RXD （串行输入线）
P3.1	TXD （串行输出线）
P3.2	$\overline{\text{INT0}}$（外部中断0输入线）
P3.3	$\overline{\text{INT1}}$ （外部中断1输入线）
P3.4	T0 （定时器0外部计数脉冲输入）
P3.5	T1 （定时器1外部计数脉冲输入）
P3.6	$\overline{\text{WR}}$ （外部数据存储器写选通信号输入）
P3.7	$\overline{\text{RD}}$ （外部数据存储器读选通信号输入）

4.1.1.2 端口的内部结构

四个端口的一位结构见图5.1，同一个端口的各位具有相同的结构。由图可见，四个端口的结构有相同之处：

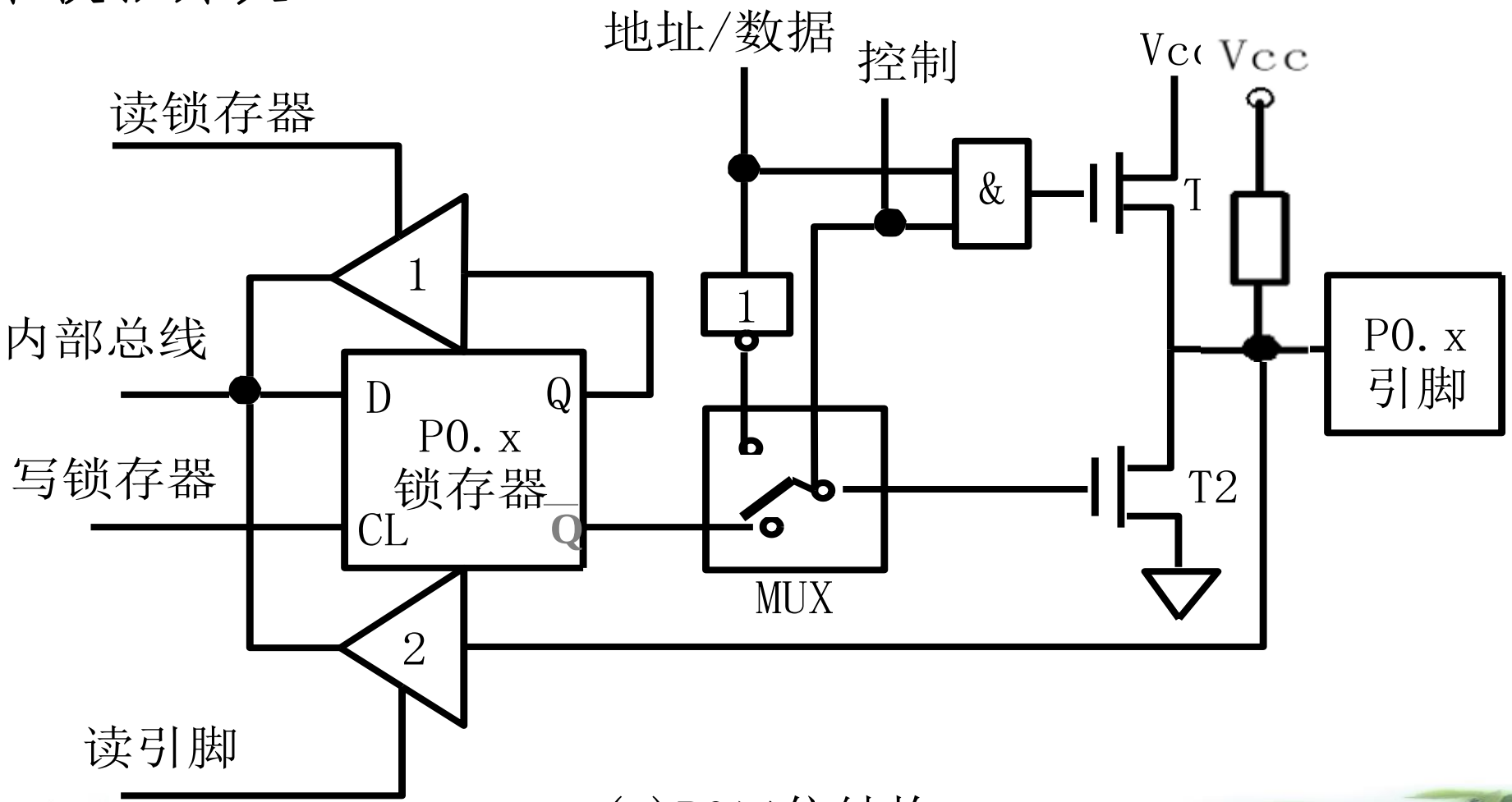
★ 都有两个输入缓冲器，分别受内部读锁存器和读引脚控制信号的控制。

★ 都有锁存器(即专用寄存器P0～P3)

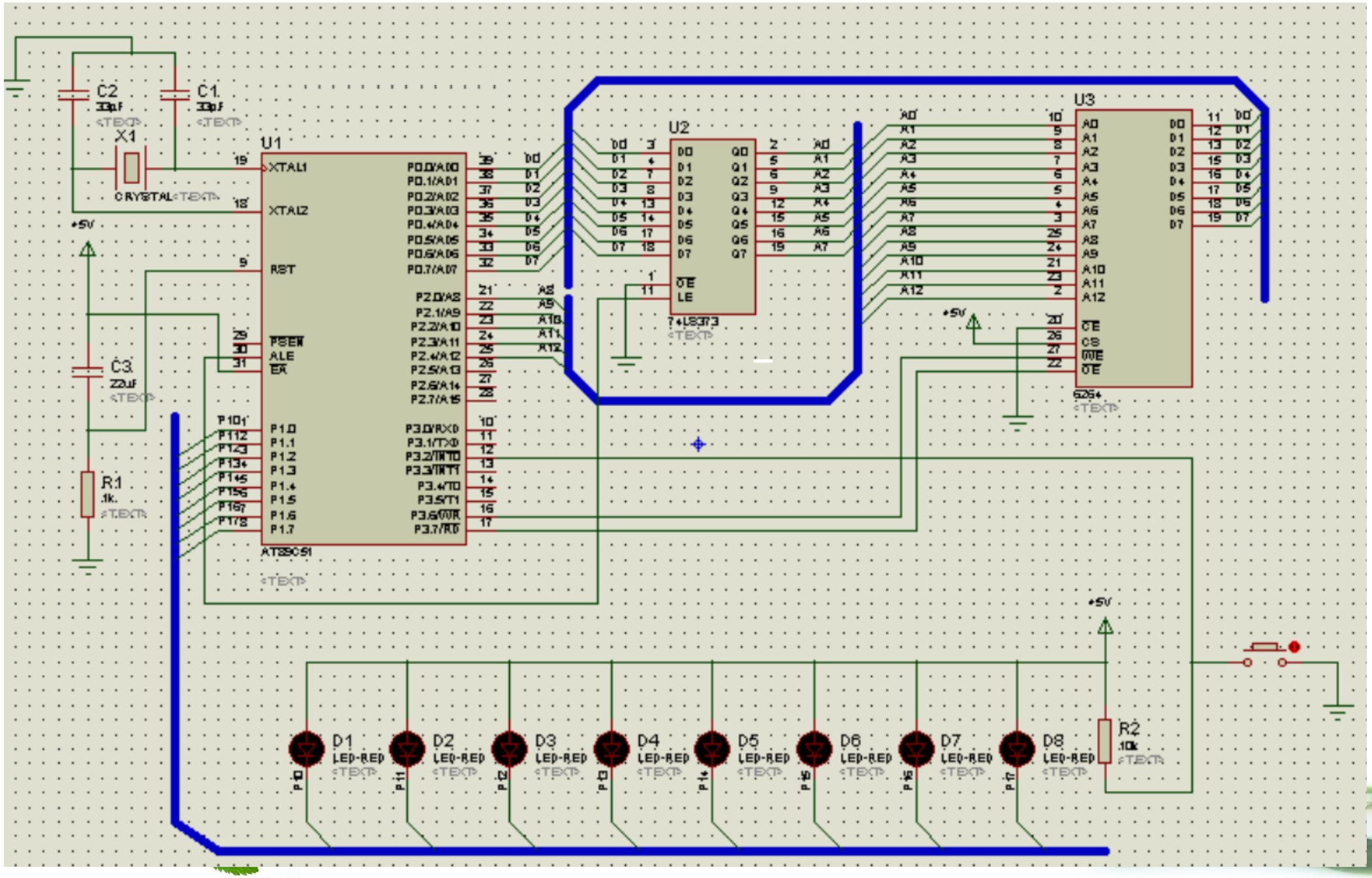
★都是场效应管输出驱动。

依据每个端口的不同功能，内部结构亦有不同之处，以下重点介绍不同之处。

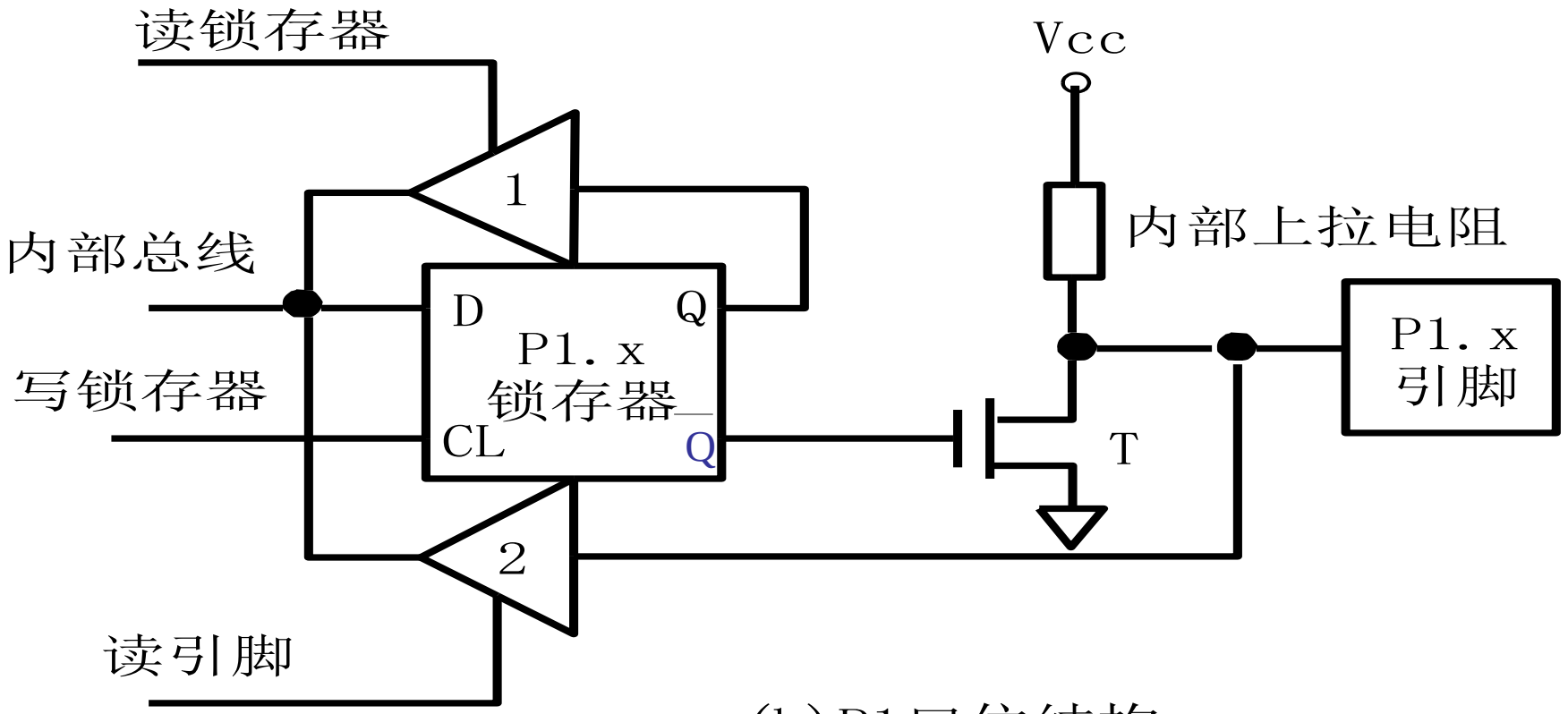
1.P0口　P0口的输出驱动电路由上拉场效应管T1和驱动场效应T2组成，控制电路包括一个与门，一个非门和一个模拟开关MUX。



(a)P0口位结构


C2
33pF
C1
33pF
X1
CRYSTAL
U1
XTAL1
XTAL2
RST
PSEN
ALE
EA
P0.0/AD0
P0.1/AD1
P0.2/AD2
P0.3/AD3
P0.4/AD4
P0.5/AD5
P0.6/AD6
P0.7/AD7
P2.0/A8
P2.1/A9
P2.2/A10
P2.3/A11
P2.4/A12
P2.5/A13
P2.6/A14
P2.7/A15
P1.0
P1.1
P1.2
P1.3
P1.4
P1.5
P1.6
P1.7
P3.0/RXD
P3.1/TXD
P3.2/INT0
P3.3/INT1
P3.4/T0
P3.5/T1
P3.6/WR
P3.7/RD
AT89C51
C3
22uF
R1
1k
+5V
U2
74LS373
OE
LE
U3
6264
CE
CS
WE
OE
D1
D2
D3
D4
D5
D6
D7
D8
LED-RED
R2
10k

2.P1口 P1口的结构见下图

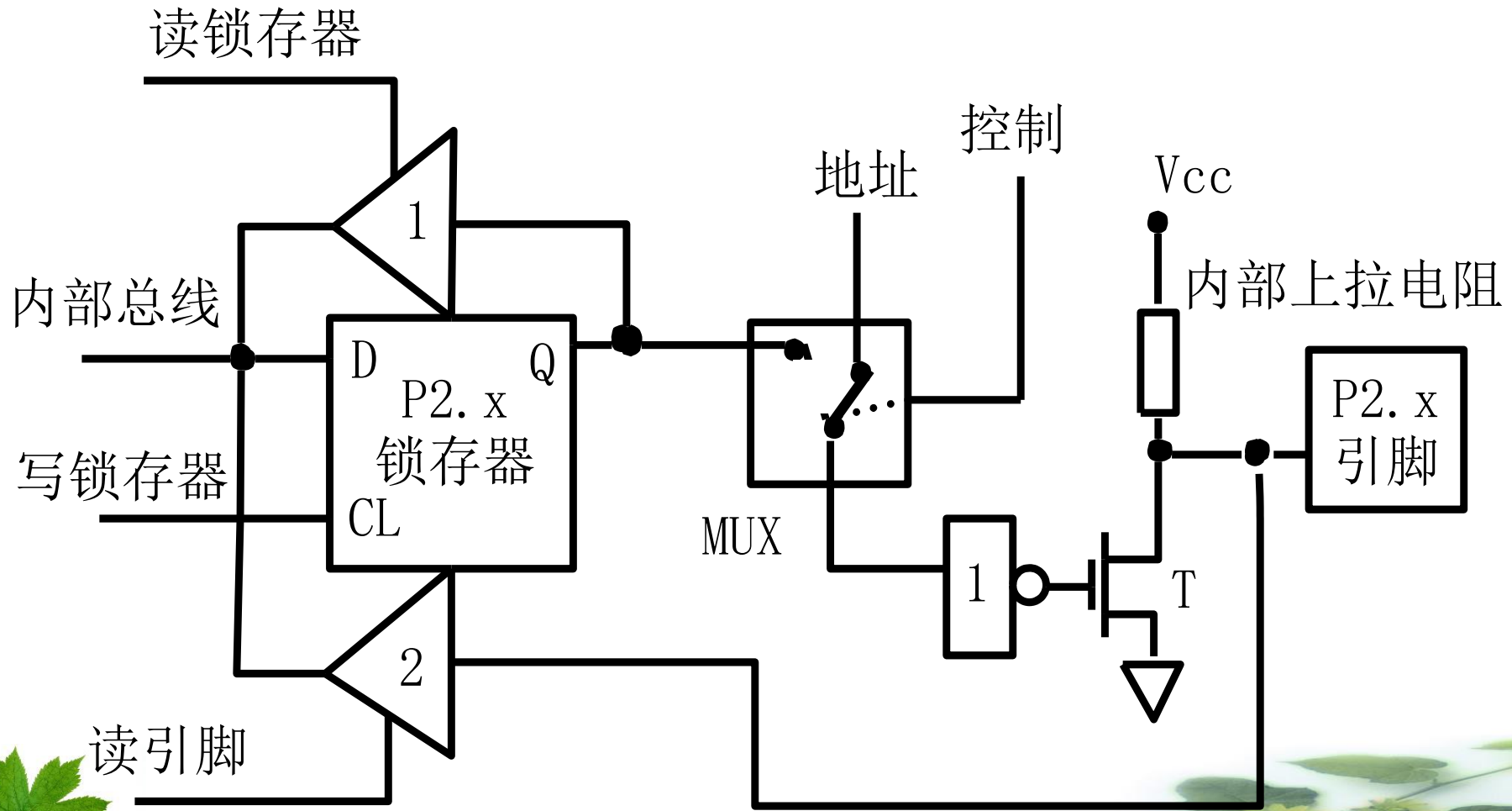


(b) P1口位结构

P1口作通用I/O口使用，因电路结构上输出驱动部分接有上拉电阻。当作输入时，同P0一样，要先对该口写“1”。

3.P2口

P2口的位结构比P1多了一个转换控制部分，当P2口作通用I/O口时，多路开关MUX倒向左；



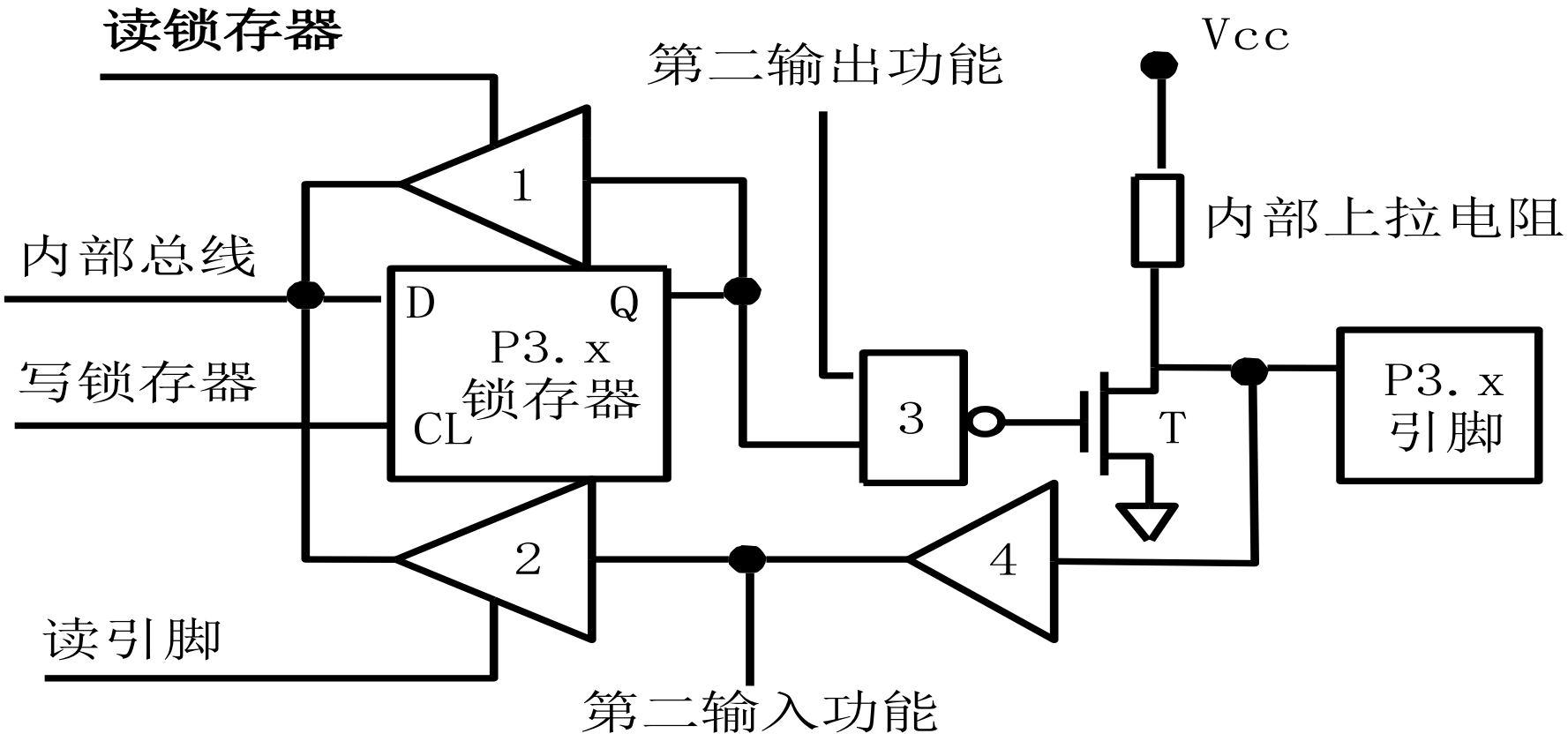
当扩展片外存贮器时，MUX开关打向右，P2口作高八位地址线输出高八位地址信号。

其MUX的的倒向选择是受CPU内部控制的。

应当注意：当P2口的几位作地址线使用时，剩下的P2口线不能作I /O口线使用。

4.P3口

P3口 为双功能I/O口，内部结构中增加了第二输入/输出功能。



(d)P3口位结构

P3的各位如不设定为第二功能则自动处于第一功能，在更多情况下，根据需要， 把几条口线设为第二功能，剩下的口线可作第一功能(I/O)使用，此时，宜采用位操作形式 。

归纳四个并行口使用的注意事项如下：

1. 如果单片机内部有程序存贮器，不需要扩展外部存贮器和I/O接口，单片机的四个口均可作I/O口用。
2. 四个口在作输入口使用时，均应先对其写“1”，以避免误读。
3. P0口作I/O口使用时应外接10K的上拉电阻，其它口则可不必。
4. P2可某几根线作地址使用时，剩下的线不能作I/O口线使用。
5. P3口的某些口线作第二功能时，剩下的口线可以单独作I/O口线使用。

4.1.2 编程举例

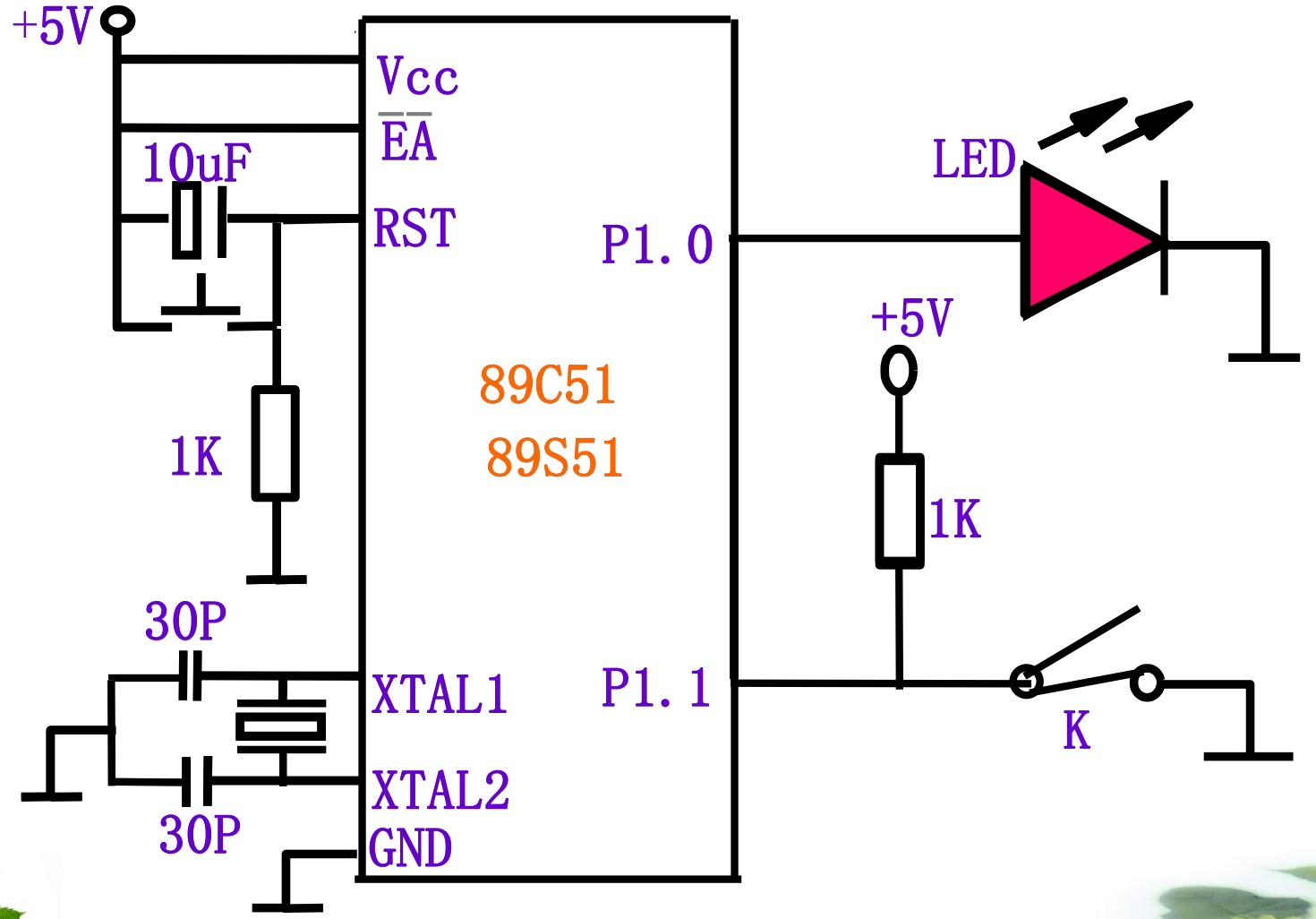
下面举例说明端口的输入、输出功能，其他功能的应用实例在后面章节说明。

例4-1.设计一电路，监视某开关K，用发光二极管LED显示开关状态，如果开关合上，LED 亮、 开关打开，LED熄灭。

分析：设计电路如图5.2如示。

开关接在P1.1口线，LED接P1.0口线， 当开关断开时， P1.1为+5V，对应数字量为“1”，开 关合上时P1.1电平为0V，对应数字量为“0”，这样就可以用JB指令对开关状态进行检测 。

LED正偏时才能发亮，按电路接法，当P1.0输出“1”，LED正偏而发亮，当P1.0 输出“0”，LED 的两端电压为 0 而熄灭。

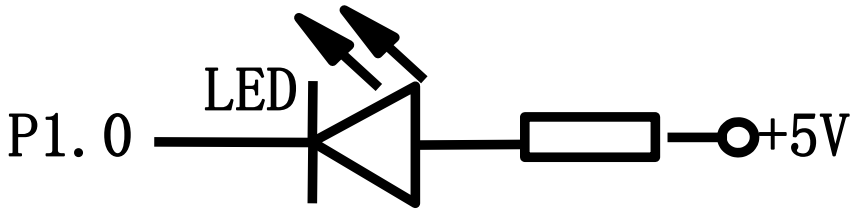


编程如下：

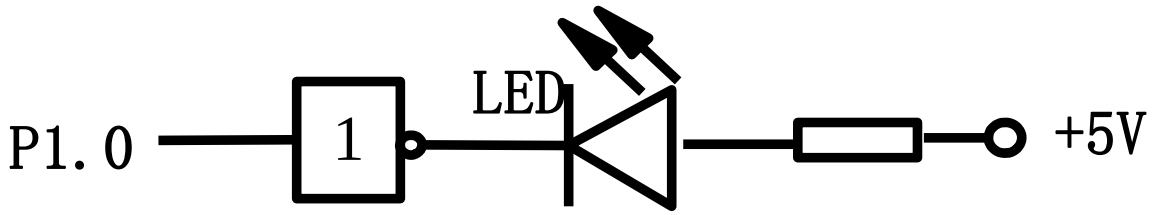
```
        CLR   P1.0        ；使发光二极管灭
AGA:SETB  P1.1        ；先对P1口写入“1”
        JB   P1.1，LIG   ；开关开，转LIG
        SETB  P1.0        ；开关合上，二极管亮
        SJMP  AGA
LIG:   CLR   P1.0          ；开关开，二极管灭
        SJMP  AGA
```

在上述电路图中二极管亮度不够，按下面两种电路接法，增加了驱动能力，二极管更亮些。

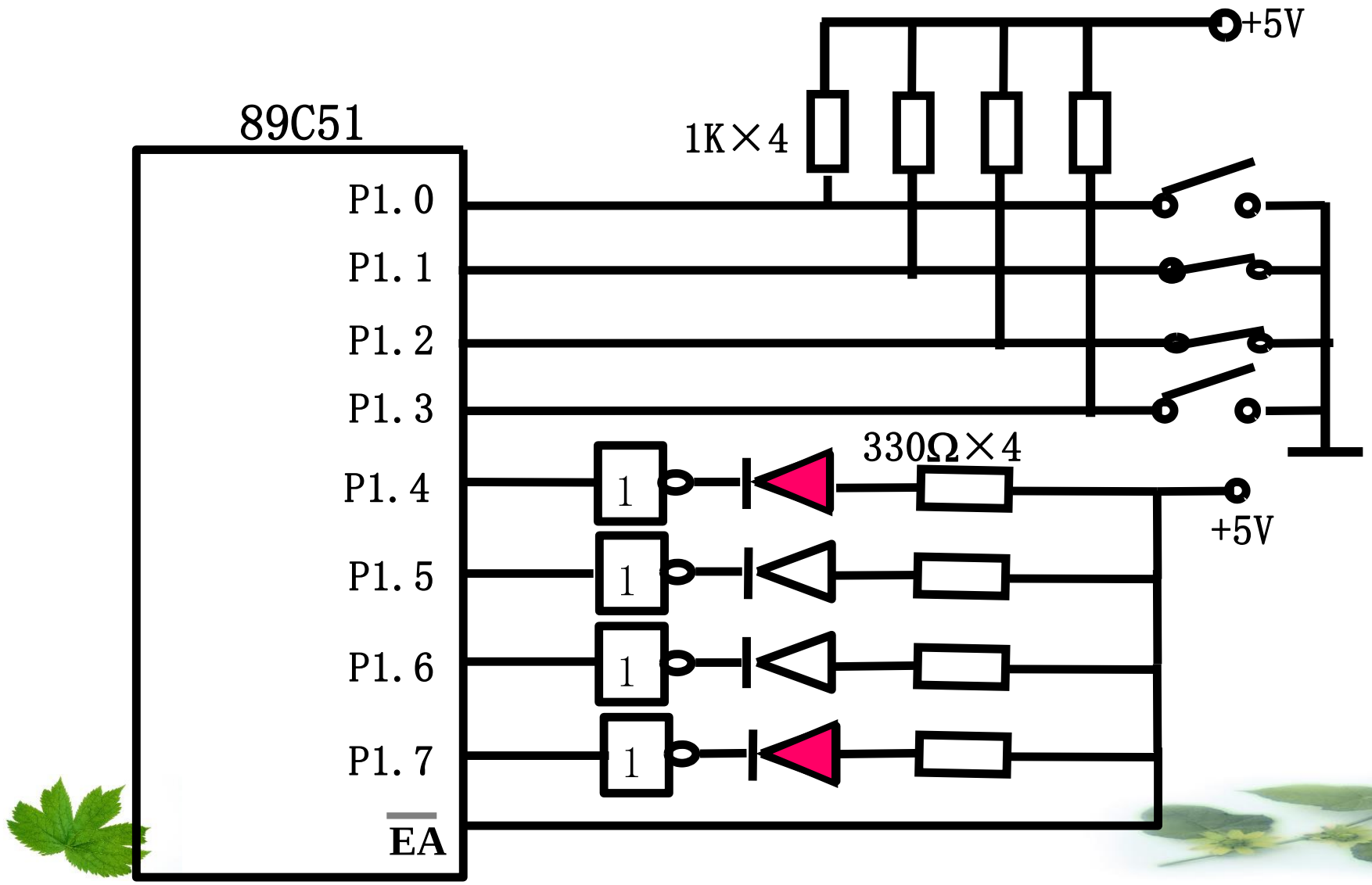
接成灌电流形式：



加驱动电路：



例4-2. 在下图中P1.4～P1.7接四个发光二极管LED, P1.0～P1.3接四个开关，编程将开关的状态反映到发光二极管上。



上述程序中每次读开关之前，输入位都先置“1”，保证了开关状态的正确读入。

编程如下：

```
        ORG   0000H
ABC: MOV  P1, #0FH    ;高四位灭，低四位送“1”
        MOV  A, P1          ;读P1口引脚开关状态至A
        SWAP  A          ;低四位开关状态转换到高四位
        ANL  A, #0F0H       ;保留高四位
        MOV  P1, A          ;从P1口输出
        SJMP  ABC           ;循环
```

4.1.3 用并行口设计LED数码显示器和键盘电路

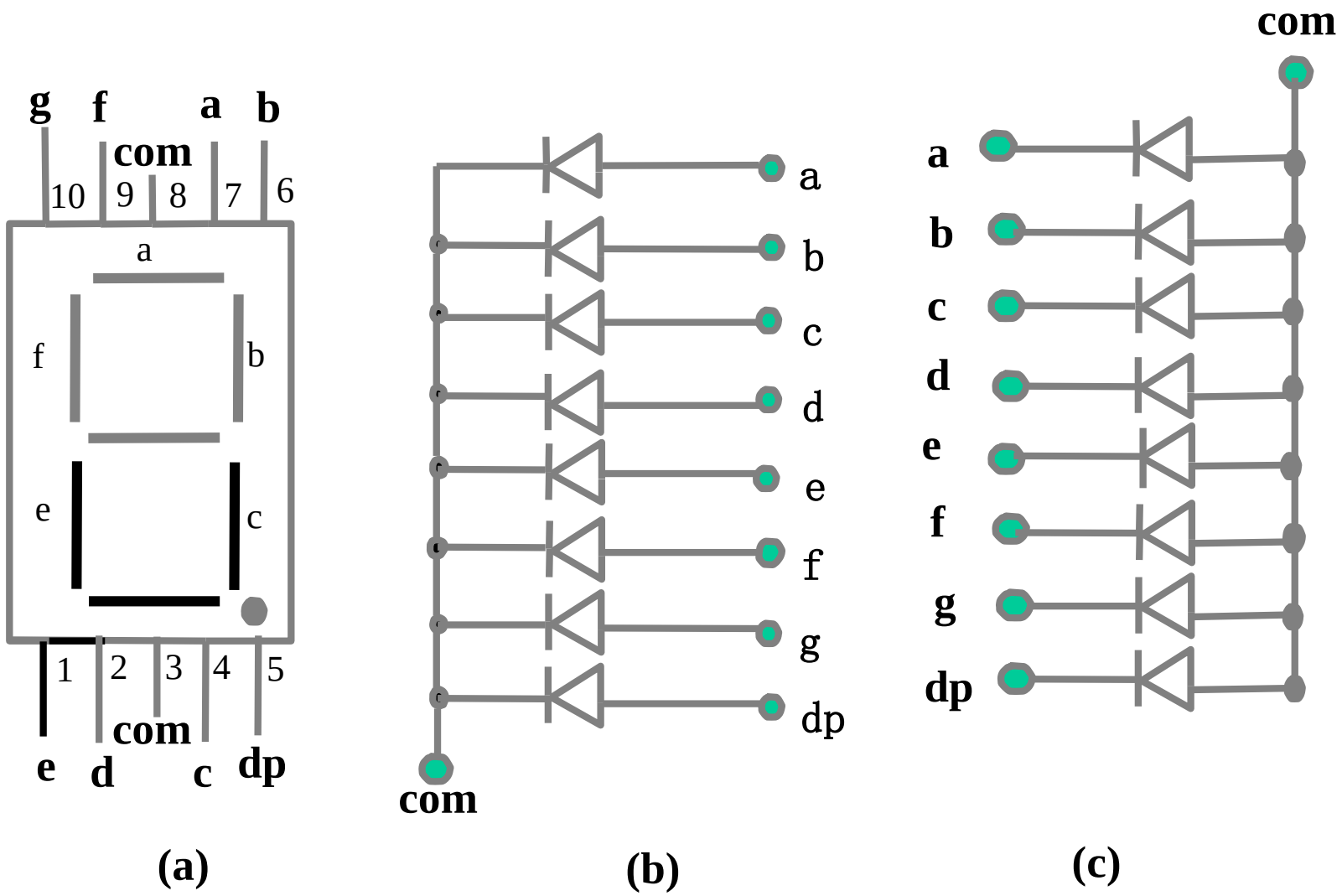
键盘和显示器是单片机应用系统中常用的输入输出装置。LED数码显示器是常用的显示器之一，下面介绍用单片机并行口设计LED数码显示电路和键盘电路的方法。

4.1.3.1用并行口设计LED显示电路

1. LED显示器及其原理

LED有着显示亮度高，响应速度快的特点，最常用的是七段式LED显示器，又称数码管。

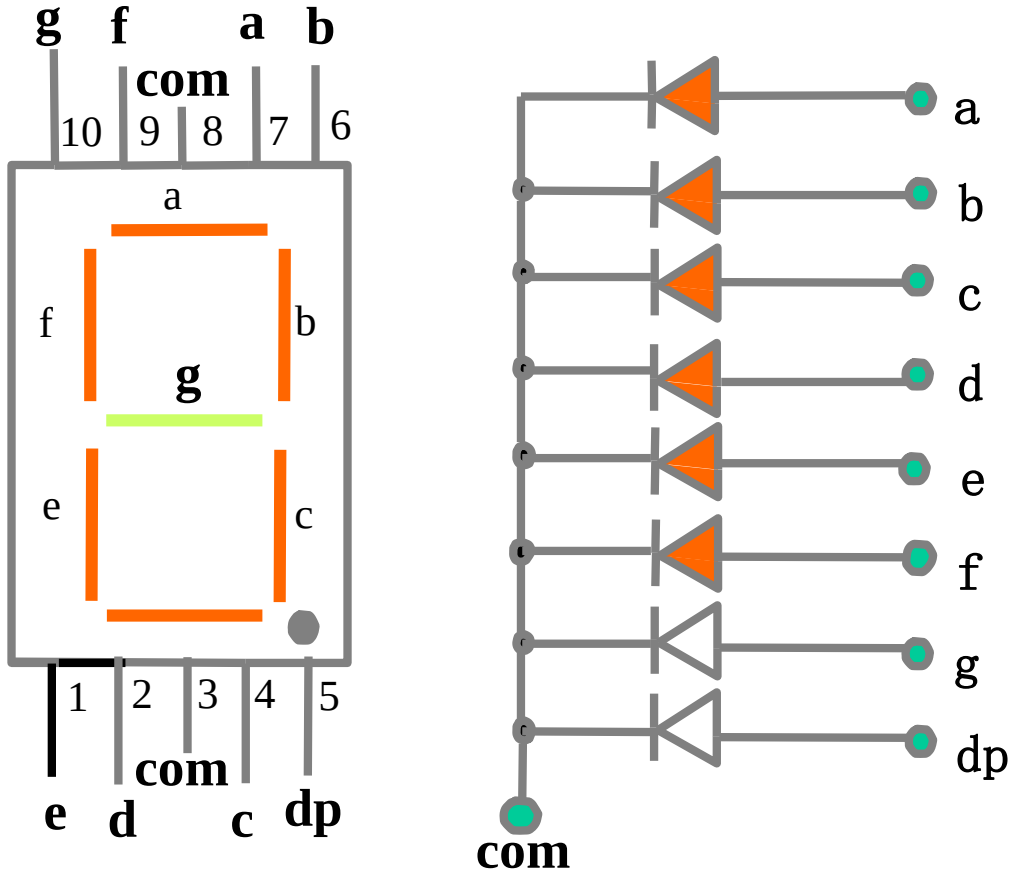
七段LED显示器内部由七个条形发光二极管和一个小圆点发光二极管组成，根据各管的亮暗组合成字符。常见LED的管脚排列见图4(a)。其中COM为公共点，根据内部发光二极管的接线 形式，可分成共阴极型图4(b)和共阳极型图4(c)。


g f com a b
10 9 8 7 6
a
f b
e c
1 2 3 4 5
e d com c dp
com
a
b
c
d
e
f
g
dp
com
a
b
c
d
e
f
g
dp

(a) (b) (c)

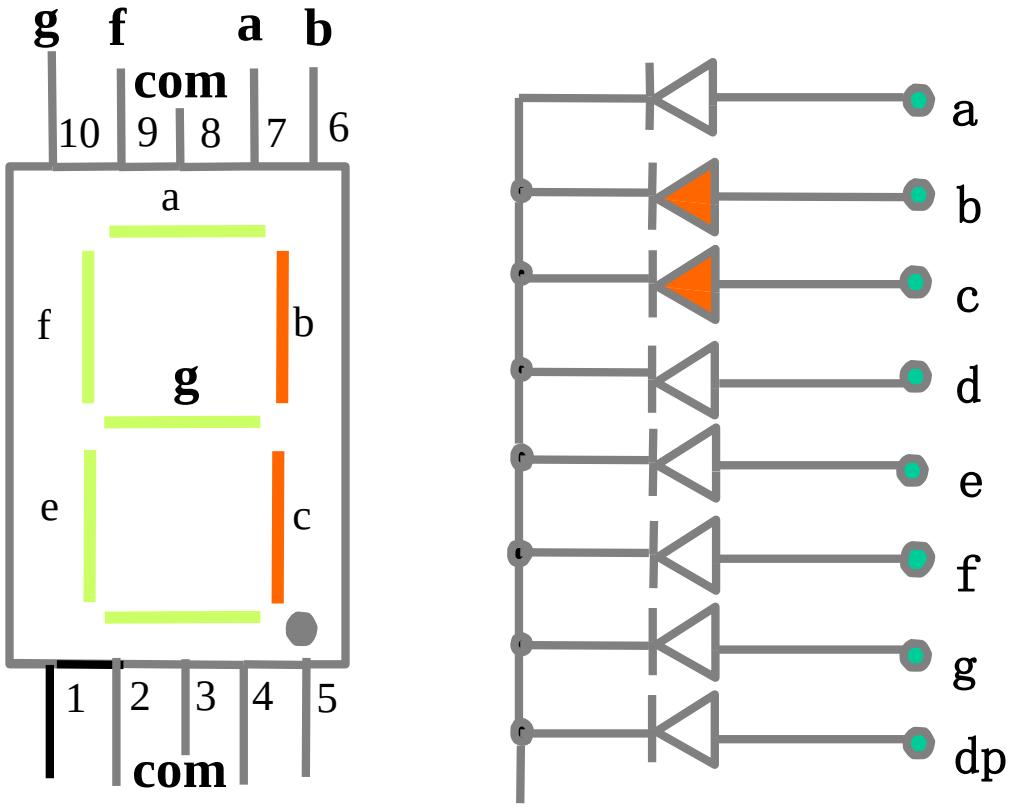
图4

LED数码管的g～a七个发光二极管因加正电压而发亮，因加零电压而不能发亮，不同亮暗的组合就能形成不同的字形，这种组合称之为字形码，例如显示”0”



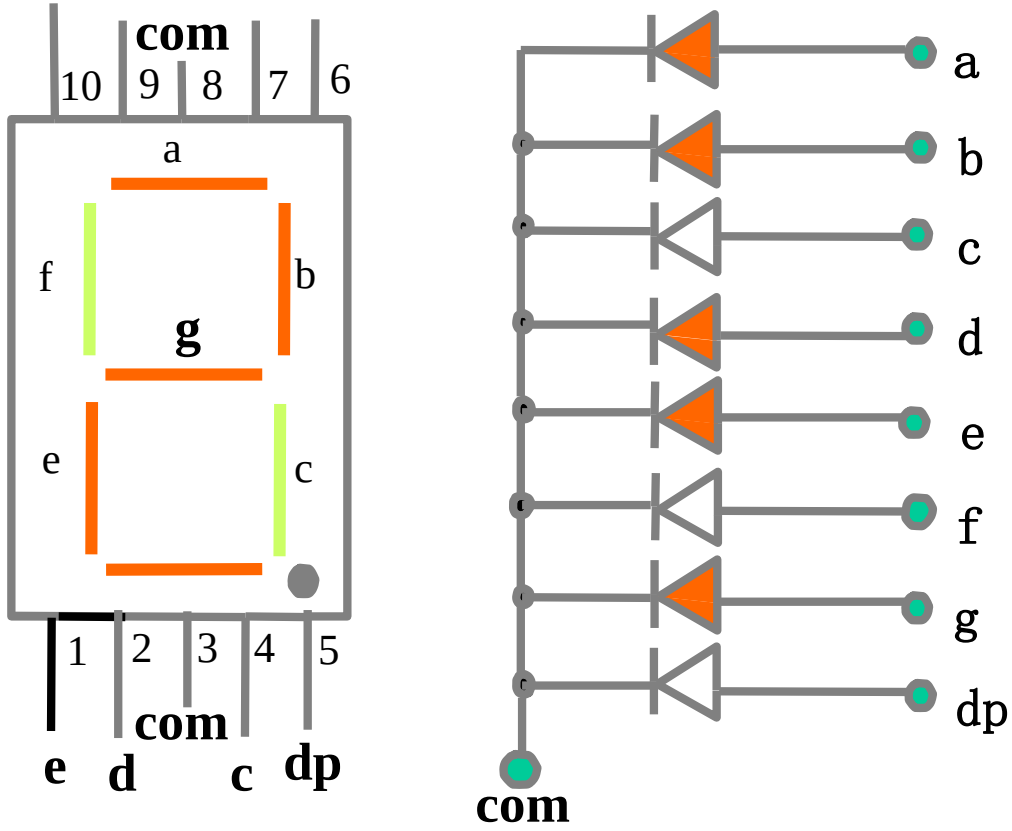
x	g	f	e	d	c	b	a	
0	0	1	1	1	1	1	1	3fh

例如显示”1”



x	g	f	e	d	c	b	a	
0	0	0	0	0	1	1	0	06h

例如显示”2”



x g f e d c b a

0 1 0 1 1 0 1 1 （5bh）

显然共阳极和共阴极的字形码是不同的，其字形码见下表。LED数码管每段需10～20ma的驱动电流，可用TTL或CMOS器件驱动。

字形码的控制输出

1、采用硬件译码方式，如采用BCD 7段译码/驱动器74LS48、74LS49、CD4511(共阴极)或74LS46、74LS47、CD4513(其阳极)

2、可用软件查表方式输出。比较经济！

显示字符	段　符　号								十六进制代码	
	dp	g	f	e	d	c	b	a	共阴	共阳
0	0	0	1	1	1	1	1	1	3FH	C0
1	0	0	0	0	0	1	1	0	06H	F9
2	0	1	0	1	1	0	1	1	5BH	A4
3	0	1	0	0	1	1	1	1	4FH	B0
4	0	1	1	0	0	1	1	0	66H	99
5	0	1	1	0	1	1	0	1	6DH	92
6	0	1	1	1	1	1	0	1	7DH	82
7	0	0	0	0	0	1	1	1	07H	F8
8	0	1	1	1	1	1	1	1	7FH	80
9	0	1	1	0	1	1	1	1	6FH	90
A	0	1	1	1	0	1	1	1	77H	88
B	0	1	1	1	1	1	0	0	7CH	83
C	0	0	1	1	1	0	0	1	39H	C6
D	0	1	0	1	1	1	1	0	5EH	A1
E	0	1	1	1	1	0	0	1	79H	86
F	0	1	1	1	0	0	0	1	71H	84
H	0	1	1	1	0	0	0	1	76H	FF
P	1	1	1	1	0	0	1	1	F3H	BF

2. **数码管直接**接法占用接口多，如果P0口和P2口要用作数据线和地址线，仅用单片机的并行口就只能接二个数码管。也可以用串行接口的方法接多个数码管， 使之静态显示.

动态接口采用各数码管循环轮流显示的方法，当循环显示频率较高时，利用人眼的暂留特性，看不出闪烁显示现象，这种显示需要一个接口完成字形码的输出(字形选择)，另一接口完 成各数码管的轮流点亮(数位选择)。

例如图5是接有五个共阴极数码管的动态显示接口电路，用74LS373接成直通的方式作驱动 电路，阴极用非门74LS04反相门驱动，字形选择由P1口提供，位选择由P3口控制。

当P3.0～P3.4轮流输出1时，五个数码管轮流显示。P1.7接开关，当开关打向位置“1”时， 显示“12345”字样，当开关打向“2”时，显示“HELLO”字样，程序清单如下：

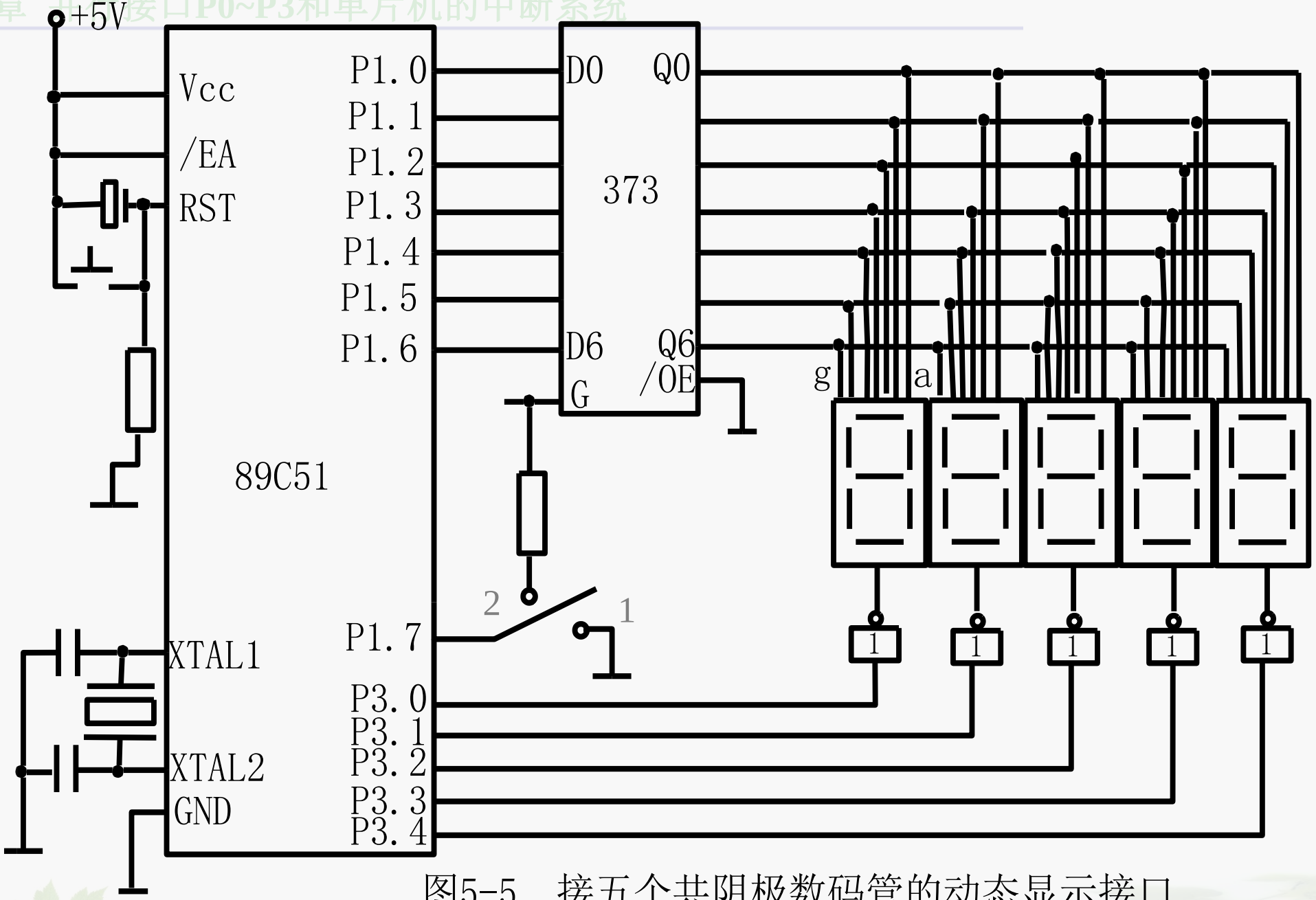


图5-5　接五个共阴极数码管的动态显示接口

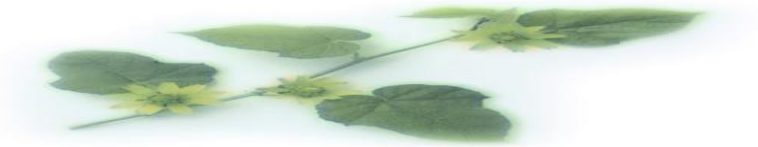
用汇编语言编程

```
        ORG   0000H
        MOV   P3，#0              ；清显示
TEST:SETB   P1.7
        JB   P1.7，DIR1           ；检测开关
        MOV   DPTR，#TAB1   ;开关置于1，‘12345’字形表头地址
        SJMP   DIR
DIR1: MOV   DPTR，#TAB2    ;开关置于2，“HELLO”字形表头
DIR:   MOV   R0，#0               ；R0存字形表偏移量
        MOV   R1，#01             ；R1置数码表位选代码
NEXT:MOV   A，R0
        MOVC   A，@A+DPTR   ;查字形码表1
        MOV   P1，A               ；送P1口输出
```

```
        MOV   A，R1
        MOV   P3，A     ；输出位选码
       ACALL   DAY     ；延时
        INC   R0          ；指向下一位字形
        RL   A            ；指向下一位
        MOV   R1，A
       CJNE   R1，#20H，NEXT ;五个 数码管显示完?
        SJMP   TEST
DAY:MOV   R6，#20   ；  延时20ms子程序
DL2: MOV   R7，#7DH
```

```
DL1: NOP
     NOP
     DJNZ  R7，DL1
     DJNZ  R6，DL2
     RET
TAB1:db 06H,5BH,4FH,66H,6DH   ；“1～5”的字形码
TAB2:db 76H,79H,38H,38H,5CH   ；“HELLO”的字形码
     END
```

```
#include<reg 51.h>
#define uint unsigned int
#define uchar unsigned char
sbit P17=P1^7;
main ( ){
uchar code tab1[5]={0x06,0x5B,0x4F,0x66,0x6D} ;   /*“1~5”的字形码，因P1.7接的开关，最高位送的“1”*/
uchar code tab2[5]={0x76,0x79,0x38,0x38,0x5C};   /*“HELLO”的段码 “1”*/
uchar i;uint j;
while(1)
 {      P3=0x01;//如果不加反向就P3=0xfe;
   for(i=0;i<5;i++)
   {    P1=0x00;     //仿真中必须加上清除以前显示
     if(P17==1)  P1=tab1[i];
          else P1=tab2[i];
         P3<<=1;  //如果不加反向就P3=(P3<<1)|0x01;
         for(j=0;j<=250;j++); //delay();延时
   }}}
```



4.1.3.2用并行口设计键盘电路

键盘是计算机系统中不可缺少的输入设备，当按键少时可接成线性键盘(如图5.3中的按键)，当按键较多时，这样的接法占用口线较多。将按键接成矩阵的形式，可以节省口线，例如两个接口可按8*8的形式接64个按键。每个按键有它的行值和列值，行值和列值的组合就是识别这个按键的编码。矩阵的行线和列线分别通过两并行接口和CPU通信。每个按键的状态同样需变成数字量“0”和“1”，开关的一端通过电阻接Vcc(列)、而接地是通过程序输出数字“0”实现的。

键盘处理程序的任务是：

确定有无键按下；

判哪一个键按下，

键的功能是什么；

还要消除按键在闭合或断开时的抖动。

两个并行口中，一个输出扫描码， 使按键逐行动态接地(称行扫描)，另一个并行口输入按键状态(称回馈信号，键盘的列值)， 由行扫描值和回馈信号共同形成键编码而识别按键、通过软件查表，查出该键的功能。也可由硬件编码器完成键的编码。

下图中，用8XX51的并行口P1接4×4矩阵键盘，以P1.0～P1.3作输出线，以P1.4～P1.7作输入线，键盘扫描程序的流程如图5.7所示。

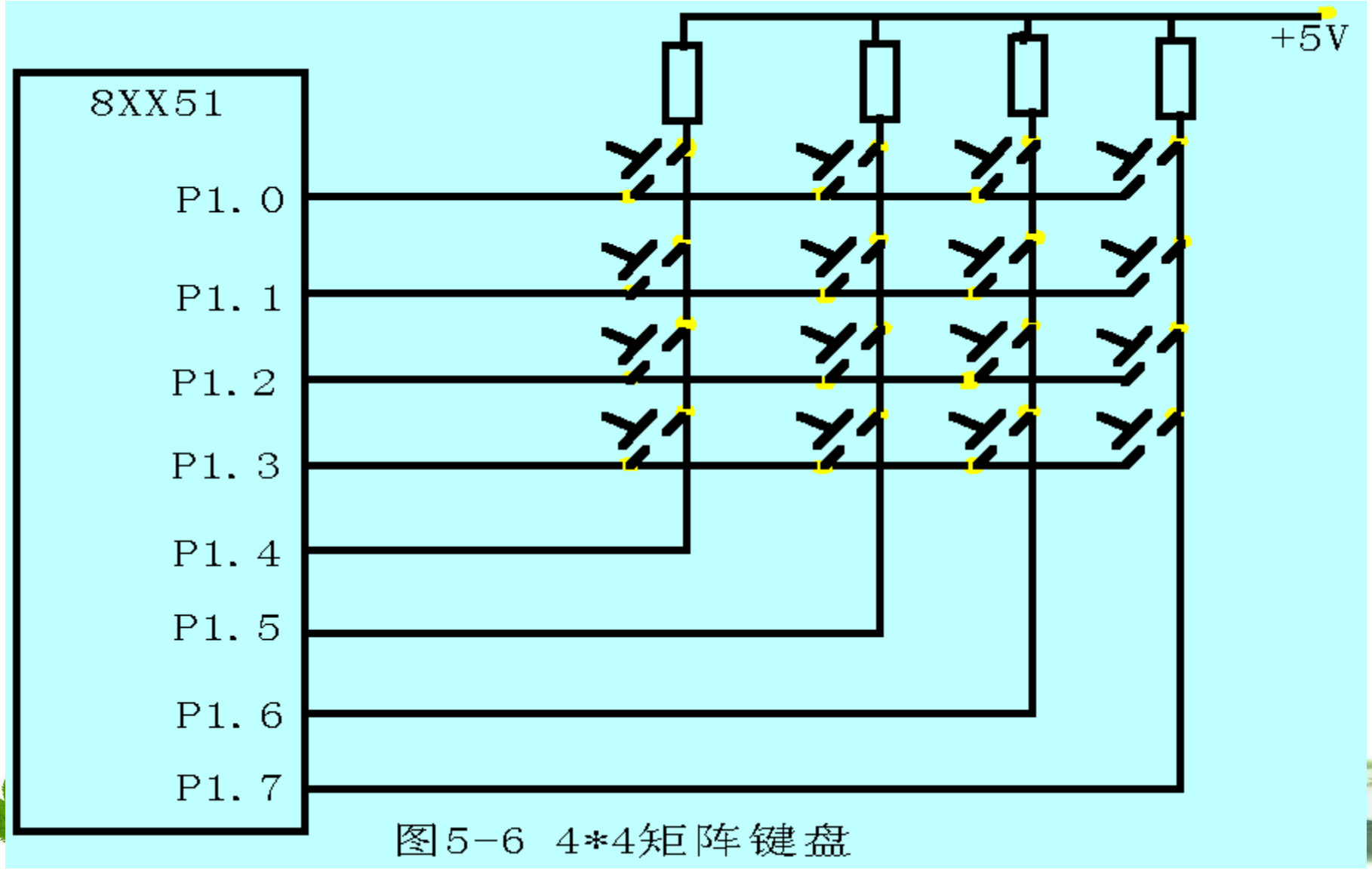


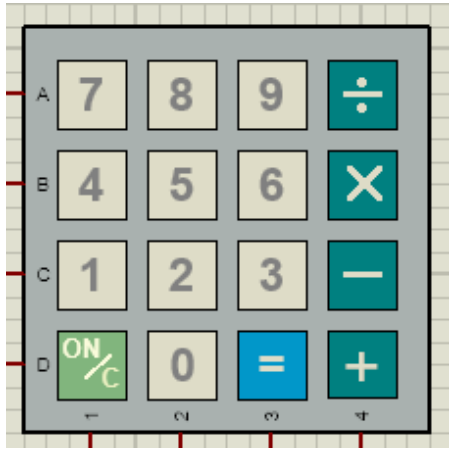
图5-6 4*4矩阵键盘

EE DE BE 7E

ED DD BD 7D

EB DB BB 7B

E7 D7 B7 77



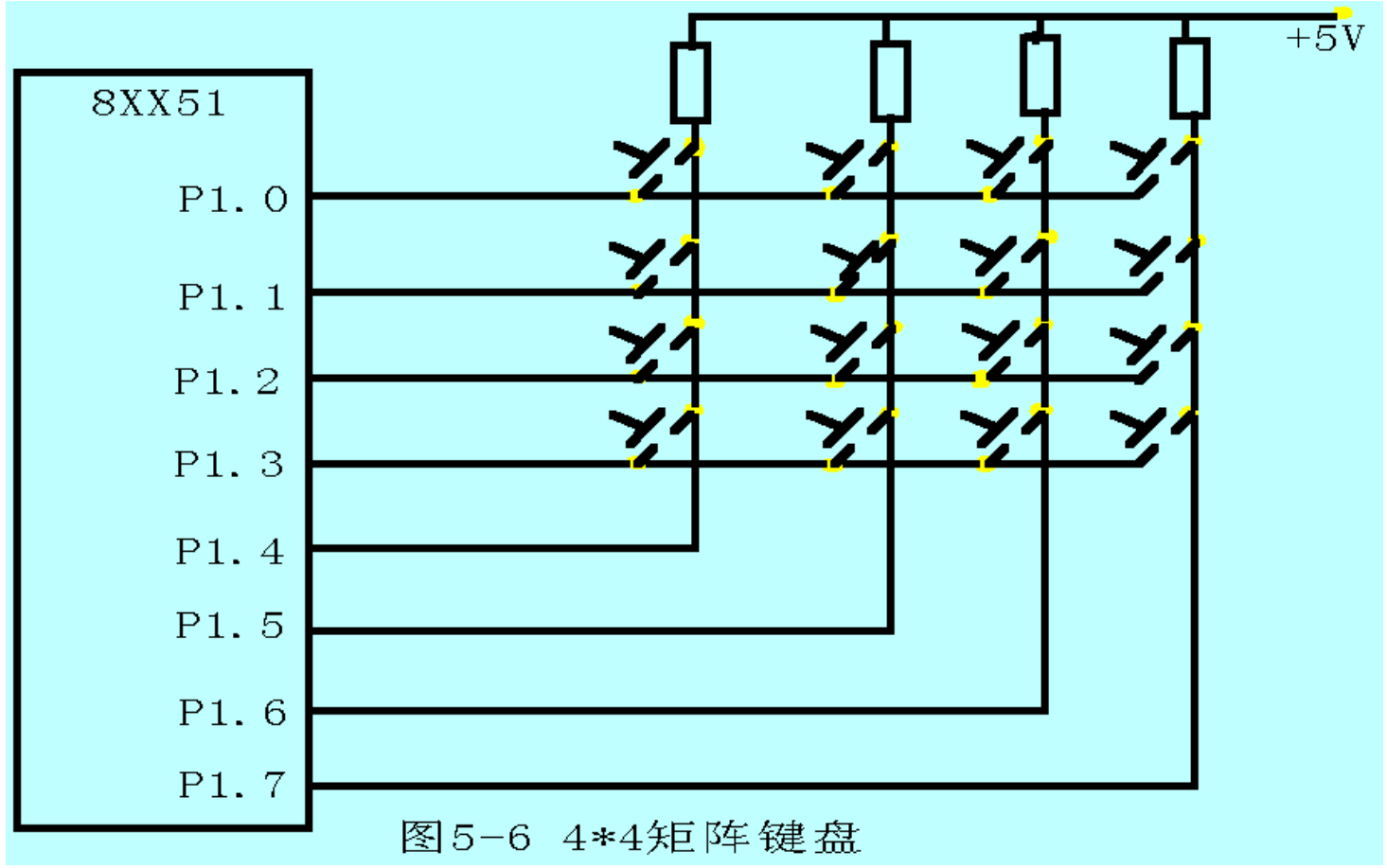
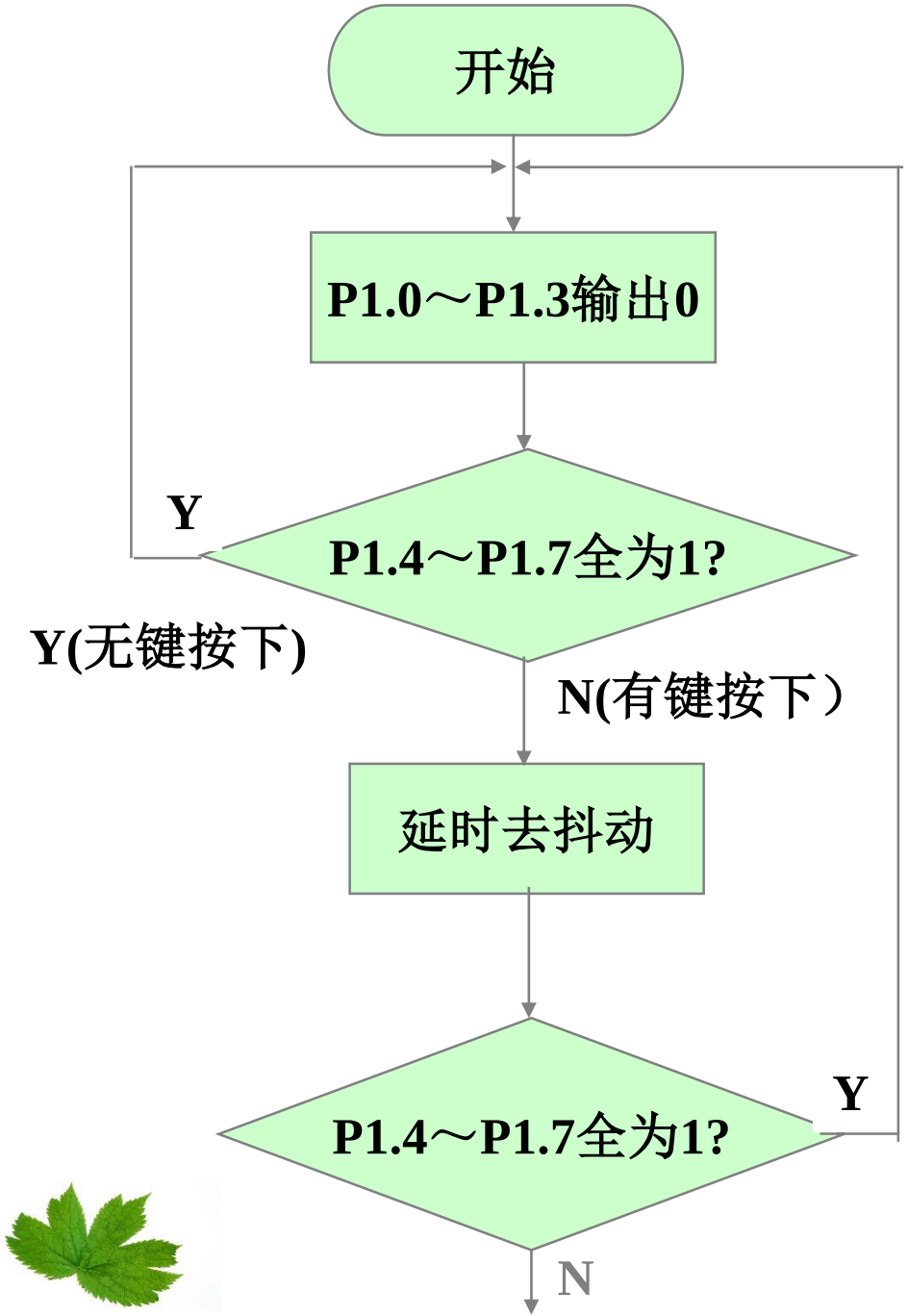
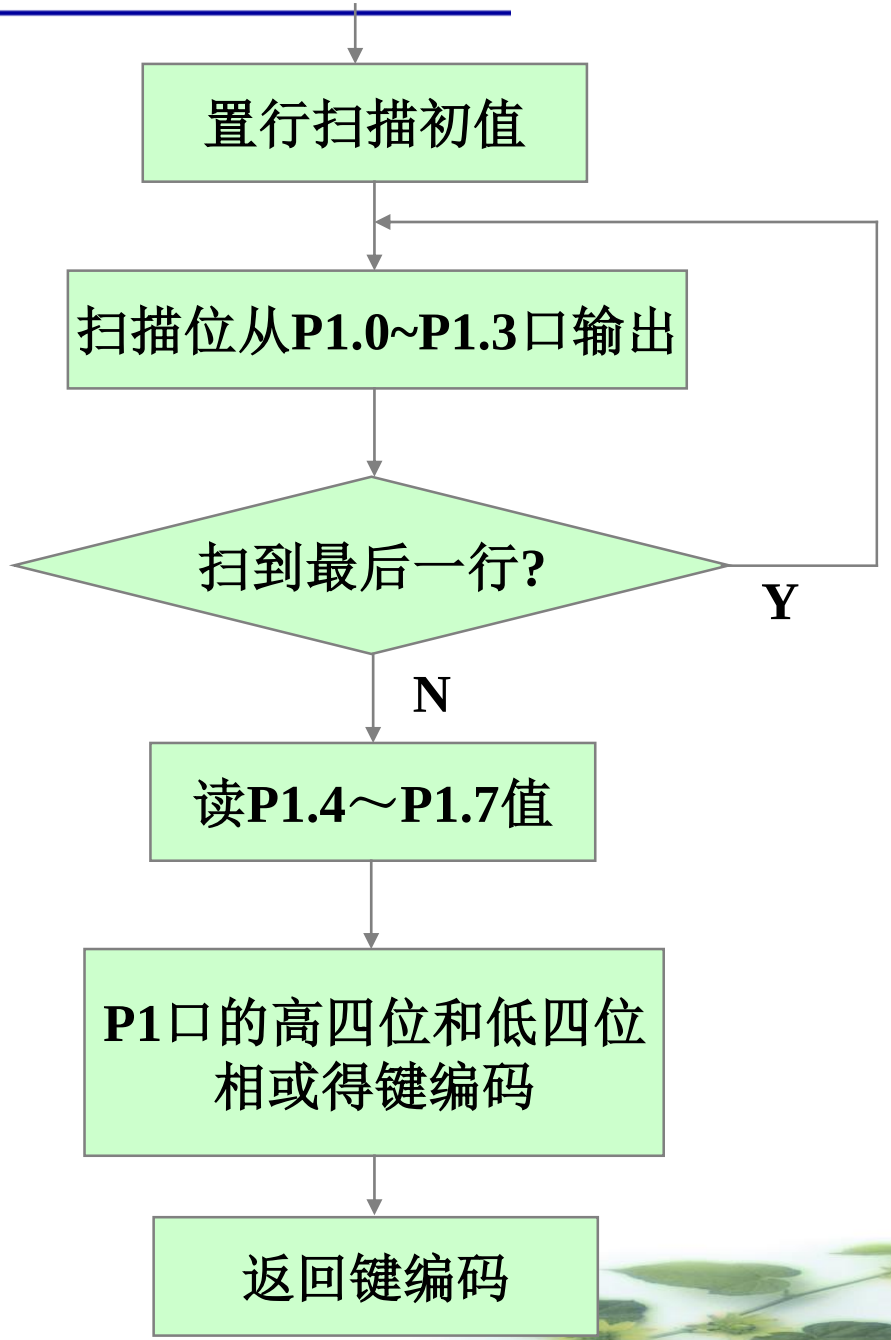


图5-6 4*4矩阵键盘

```
uchar code
key_value[16]={0xD7,0xEB,0xDB,0xBB,0xED,0xDD,0xBD,0xEE,0xDE,0xBE,0x77,0x7B,0x7D,0x7E,
0xB7,0xE7};//键盘标识对应的键盘值
```


开始
P1.0～P1.3输出0
P1.4～P1.7全为1?
Y
Y(无键按下)
N(有键按下）
延时去抖动
P1.4～P1.7全为1?
Y
N


置行扫描初值
扫描位从P1.0~P1.3口输出
扫到最后一行?
Y
N
读P1.4～P1.7值
P1口的高四位和低四位
相或得键编码
返回键编码

程序清单如下：

```
ORG  0000H
TEST:
MOV   P1,#0F0H     ; P1.0～P1.3输出0, P1.4～P1.7输出1,作输入位
MOV  A,P1          ;读键盘，检测有无键按下
ANL  A,#0F0H       ; 屏蔽P1.0～P1.3，检测P1.4～P1.是否全为1
CJNE  A, #0F0H,HAVE ; P1.4～P1.7不全为1，有键按下
SJMP  TEST         ;P1.4～P1.7全为1，无键按下，重检测键盘
HAVE:
MOV  A,#0FE        ;有键按下，逐行扫描键盘，置扫描初值
```

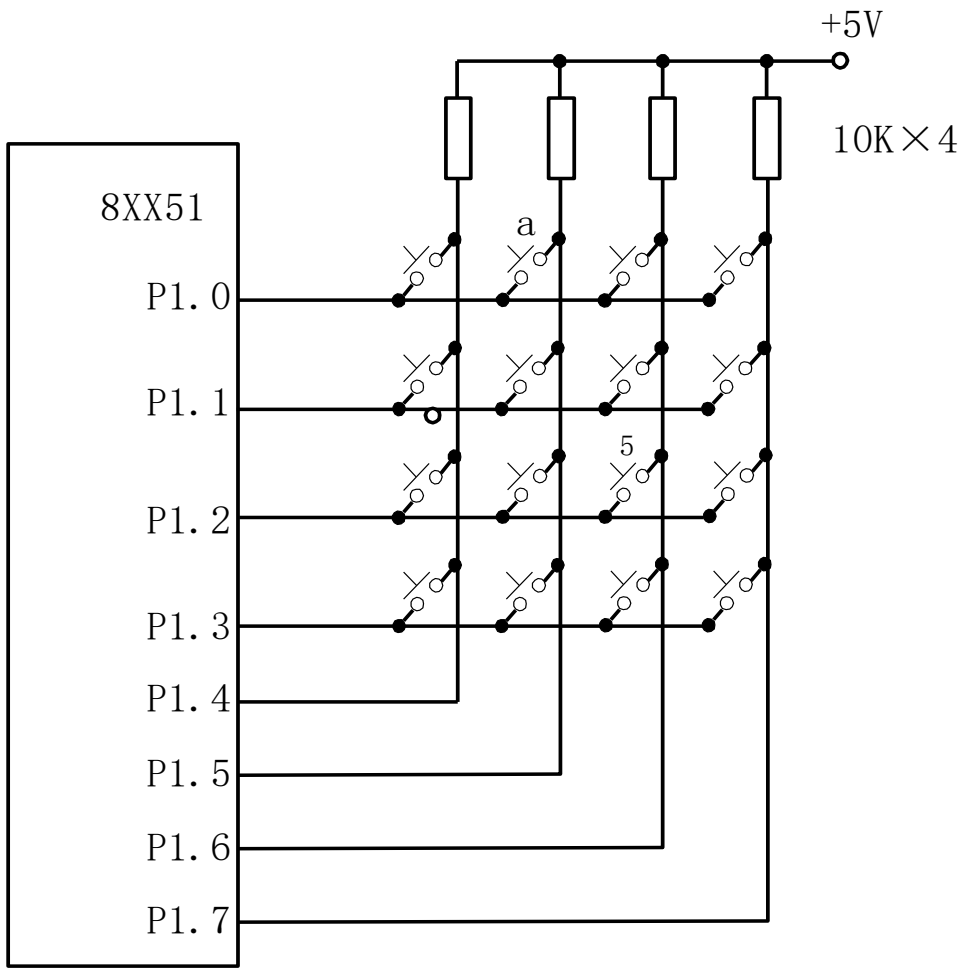
```
NEXT: MOV  B,A          ；扫描码暂存于B
      MOV  P1,A         ；输出扫描码
READ: MOV  A,P1         ；读键盘
      ANL  A,#0F0H      ；屏蔽P1.0～P1.3，
                          检测P1.4～P1.7是否全为1
      CJNE A,＃0F0H,YES ；P1.4～P1.7不全为1，
                          该行有键按下
      MOV  A,B          ；被扫行无键按下，准备查下一行
      RL   A            ；置下一行扫描码
      CJNE A,#0EFH,NEXT ；未扫到到最后一行循环
YES:  ACALL DAY         ；延时去抖动
```

```
MOV  A,P1          ；再读键盘
ANL  A,#0F0H       ；屏蔽P1.0～P1.3，保留P1.4～P1.7(列码)
MOV  R2,A          ；暂存列码
MOV  A, B
ANL  A,#0FH        ；取行扫描码
ORL  A,R2          ；行码、列码合并为键编码
MOV  B,A           ；键编码存于B
LJMP  SAM38        ；转键分析处理程序（见例3-8）
…
```

例10-15 以P1.0~P1.3作输出线，以P1.4~P1.7作输入线，如图4-6所示。

C语言编程程序清单如下：

```
#include<reg51.h>
#define uchar unsigned char
#define uint unsigned int
void delayms (void);
uchar kbscan(void);          /* 函数说明 */
void main (void)
{ uchar key;
while (1)
 {key=kbscan( );    /* 键盘扫描函数 */
  delayms( ); } }
void delayms (void)          /* 延时 */
{uchar i;
 for (i=200;i>0;i- -）{ ;}
}
```



```
uchar kbscan(void)                    / * 键盘扫描函数 * /
{uchar sccode,recode;
P1=0xf0;                              / *P1.0~P1.3发全0，P1.4~P1.7输入 * /
if ( (P1 & 0xf0)! =0xf0)              / * 如P1口高四位不全为1有键按下 * /
  {delayms( );                        / *  延时去抖动 * /
  if ( (P1 & 0xf0) ! =0xf0)           / * 在读输入值 * /
   {sccode =0xfe                      / * 最低位置0 * /
   while ( ( sccode & 0x10)! =0)      / * 不到最后一行循环 * /
    {P1 =sccode;                      /*P1口输出扫描码 * /
    if ( (P1 & 0xf0)! =0xf0)          / * 如P1.4~P1.7不全为1，该行有键按下 * /
     {recode = P1 & 0xf0;             / * 保留P1口高四位值，低四位变为全1, 作为列值 * /
   return( (sccode&0x0f) | (recode) ); }  /*行码+列值=键编码返回主程序 * /
   else
    sccode =(sccode<<1) | 0x01;       / * 如该行无键按下，查下一行，行扫描值左移一位*/
    }}}
      return( 0 );  }                 / * 无键按下，返回值为0 * /
```

4.1.4 并行接口小结

并行接口是单片机用得最多的部分，可直接接外部设备(要注意电平的匹配)。