

实用单片机讲座：

□周兴华

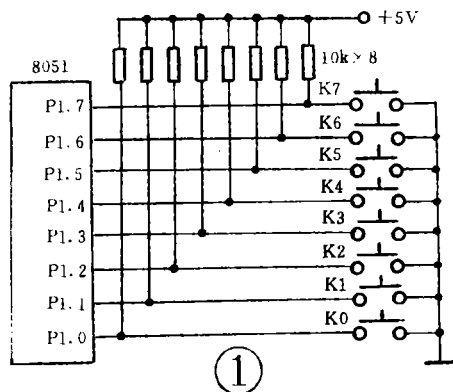
手把手教你学单片机(十四)

——单片机的键盘接口技术

键盘是单片机不可缺少的输入设备,是实现人机对话的纽带。键盘按结构形式可分为非编码键盘和编码键盘,前者是用软件方法产生键码,而后者则用硬件方法来产生键码。在单片机中使用的都是非编码键盘,因为非编码键盘结构简单、成本低廉。非编码键盘的类型很多,常用的有独立式键盘、行列式键盘等。

独立式键盘

独立式键盘是指将每个按键按一-对一的方式直接连接到 I/O 输入线上所构成的键盘,如图 1 所示。



在图 1 中,键盘接口中使用多少根 I/O 线,键盘中就有几个按键。键盘接口使用了 8 根 I/O 口线,该键盘就有 8 个按键。这种类型的键盘,键盘的按键比较少,且键盘中各个按键的工作互不干扰。因此,用户可以根据实际需要对该键盘中的按键灵活地编码。

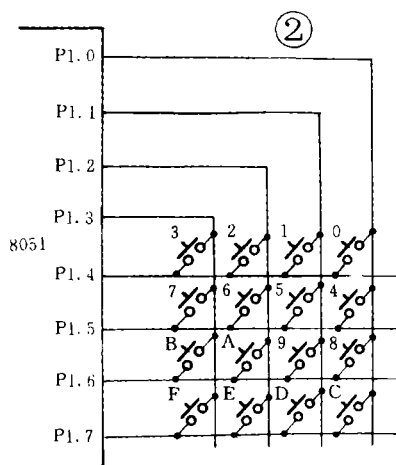
最简单的编码方式就是根据 I/O 输入口所直接反映的相应按键按下的状态进行编码,称按键直接状态码。假如图 1 中的 K0 键被按下,则 P1 口的输入状态是 11111110,则 K0 键的直接状态编码就是 FEH。对于这样编码的独立式键盘,CPU 可以通过直接读取 I/O 口的状态来获取按键的直接状态编码值,根据这个值直接进行按键识别。这种形

式的键盘结构简单,按键的识别容易。

独立式键盘的缺点是需要占用较多的 I/O 口线。当单片机应用系统键盘中需要的按键比较少或 I/O 口线比较富余时,可以采用这种类型键盘。

行列式键盘

行列式键盘是用 n 条 I/O 线作为行线,m 条 I/O 线作为列线组成的键盘。在行线和列线的每一个交叉点上,设置一个按键。这样,键盘中按键的个数是 mxn 个。这种形式的键盘结构,能够有效地提高单片机系统中 I/O 口的利用率。图 2 为行列式键盘输入示意图,列



线接 P1.0~P1.3,行线接 P1.4~P1.7。行列式键盘适合于按键输入多的情况。

独立式键盘接口的编程模式

在确定了键盘的编程结构后,就可以编制键盘接口程序。键盘接口程序的功能实际上就是驱动键盘工作,完成按键的识别,根据所识别按键的键值,完成按键子程序的正确散转,从而完成单片机应用系统对用户按键动作的预定义的响应。

由于独立式键盘的每一个按键占用一条 I/O 口线,每个按键的工作不影响其它按键,因此,可以直接依据每个 I/O 口线的状态来进行子程序散转,使

程序编制简练一些。

也可以使用键盘编码值来进行按键子程序的散转,程序更具有通用性。通用的独立式键盘接口程序由键盘管理程序、散转表和键盘处理子程序三部分组成。独立式键盘接口程序各个部分的原理如下:

1. **键盘管理程序:** 担负键盘工作时的循环监测(看是否有键被按下)、键盘去抖动、按键识别、子程序散转(根据所识别的按键进行转子程序处理)等基本工作。

2. **散转表:** 支持应用程序根据按键值进行正确的按键子程序跳转。

3. **键盘处理子程序:** 负责对具体按键的系统定义功能执行。

行列式键盘接口的编程模式

行列式键盘具有更加广泛的应用,可采用计算的方法来求出键值,以得到按键特征码。现以图 2 为例加以说明。

1. 检测出是否有键按下。方法是 P1.4~P1.7 输出全 0,然后读 P1.0~P1.3 的状态,若为全 1 则无键闭合,否则表示有键闭合。

2. 有键闭合后,调用 10~20ms 延时子程序避开按键抖动。

3. 确认键已稳定闭合后,接着判断为哪一个键闭合?方法是对键盘进行扫描,即依次给每一列线送 0,其余各列都为 1,并检测每次扫描的行状态。每当扫描输出某一系列为 0 时,相继读入行线状态。若为全 1,表示为 0 的这一列上没有键闭合,否则不为全 1,表示为 0 的这一列上有键闭合。确定了闭合键的位置后,就可计算出键值,即产生键码。

行列式键盘也可采用查表法求得键值,这样,键盘接口程序更具有通用性。

它的基本编程模式与独立式键盘一致,都是由键盘管理程序、散转表和按键子程序三部分组成。但是,行列式

键盘的按键编码方式与独立式键盘不一样,所以,其键盘管理程序需完成键盘驱动和按键识别两项工作,而独立式键盘的管理程序只需要完成按键识别一项工作。行列式键盘如按键盘按键的直接工作状态进行编码,可以使得单片机系统方便地用键盘工作时端口的输入输出状态获得按键编码,按键检测容易。但直接状态编码的码值的离散性比较大,若直接用它的值进行子程序跳转,在编程时,不好处理。因此,可以用键盘直接状态扫描码与键盘特征码一起组成一个键码查询表,程序中根据得到的键盘直接状态扫描码,用查表法查询键码查询表,得到按键特征码,程序按特征码散转。所谓的特征码就是根据子程序散转的需要,程序员自己设定的与直接状态扫描码对应的按键特征码。

键盘工作方式

CPU对键盘的扫描可以采取程序控制的随机方式,即只有在CPU空闲时才去扫描键盘,响应操作员的键盘输入,但CPU在执行应用程序(若应用程序中没有键扫描程序)的过程中,不能响应键盘输入。对键盘的扫描也可采用定时方式,即利用单片机内部定时器,每隔一定时间(如10ms)对键盘扫描一次,这种控制方式,不管键盘上有无键闭合,CPU总是定时地关心键盘状态。在大多数情况下,CPU对键盘可能进行空扫描。为了提高CPU的效率而又能及时响应键盘输入,可以采用中断方式,即CPU平时不必扫描键盘,只要当键盘上有键闭合时就产生中断请求,向CPU申请中断,CPU响应键盘中断后立即对键盘进行扫描,识别闭合键,并作相应的处理。

下面在S1板做一个独立式键盘输入实验,P1口作输入,P2口作输出。P1口的P1.0输入低电平后,P2口的P1.0输出低电平,点亮1位发光管;P1口的P1.1输入低电平后,P2口的P1.0、P1.1输出低电平,点亮2位发光管;P1口的P1.2输入低电平后,P2口的P1.0、P1.1、P1.2输出低电平,点亮3位发光管;……P1口的P1.7输入低电平后,P2口的P1.0、P1.1、P1.2、P1.3、P1.4、P1.5、P1.6、P1.7输出低电平,点亮8位发光管

在我的文档中建立一个文件目录(S26),然后建立一个S26.uv2的工程项目,最后建立源程序文件(S26.asm)。

输入下面的程序:

```

序号: 1      ORG 0000H
2      AJMP MAIN
3      ORG 030H
4      MAIN:MOV P2,#0FFH
5      MOV A,#0FFH
6      MOV P1,A
7      MOV A,P1
8      CJNE A,#0FFH,GO1
9      AJMP MAIN
10     GO1:ACALL DEL
11     CJNE A,#0FFH,GO2
12     AJMP MAIN
13     GO2:MOV DPTR,#TAB
14     MOV R0,#00H
15     L1: RRC A
16     JNC N1
17     INC R0
18     SJMP L1
19     N1: MOV A,R0
20     RLC A
21     JMP @A+DPTR
22     TAB:AJMP PR0
23     AJMP PR1
24     AJMP PR2
25     AJMP PR3
26     AJMP PR4
27     AJMP PR5
28     AJMP PR6
29     AJMP PR7
30     PR0:MOV P2,#0FEH
31     ACALL DEL
32     AJMP MAIN
33     PR1:MOV P2,#0FCH
34     ACALL DEL
35     AJMP MAIN
36     PR2:MOV P2,#0F8H
37     ACALL DEL
38     AJMP MAIN
39     PR3:MOV P2,#0F0H
40     ACALL DEL
41     AJMP MAIN
42     PR4:MOV P2,#0E0H
43     ACALL DEL
44     AJMP MAIN

```

```

45     PR5:MOV P2,#0C0H
46     ACALL DEL
47     AJMP MAIN
48     PR6:MOV P2,#80H
49     ACALL DEL
50     JMP MAIN
51     PR7:MOV P2,#00H
52     ACALL DEL
53     AJMP MAIN
54     DEL:MOV R7,#0FFH
55     DEL1:MOV R6,#0FFH
56     DEL2:DJNZ R6,DEL2
57     DJNZ R7,DEL1
58     RET
59     END

```

编译通过后,将S26文件夹中的hex文件烧录到89C51芯片中,将芯片插入到S1型LED试验板上,取下P1口的短路块,接上5V稳压电源,这时P2口外接的8个LED均不亮。用一根试验线一端接地,另一端依次触碰P1.0~P1.7,会发现P2口的发光管点亮情况从1个变化到8个。

下面我们对程序进行分析解释。

序号1(程序解释,以下同):程序开始。

序号2:跳转到MAIN主程序处。

序号3:主程序从地址30H开始。

序号4:将立即数FFH传送至P2口,关闭P2口的8个发光管。

序号5:将立即数FFH传送至累加器A。

序号6:累加器的数送P1口(全1),准备读取输入状态。

序号7:读取P1口输入状态。

序号8:若A不等于FFH,说明有键闭合,转GO1;否则无键闭合,顺序执行。

序号9:跳转至主程序MAIN处。

序号10:调用延时子程序,避开按键抖动干扰。

序号11:再判键闭合,若A不等于FFH,说明有键闭合,转GO2;否则无键闭合,顺序执行。

序号12:跳转至主程序MAIN处。

序号13:散转表首地址送DPTR数据指针。

序号14:设置初始键号(0)送寄存器R0。

序号15:累加器A的内容右移一位,这

样最低位首先移入进位寄存器 CY(即从最低位 P1.0 开始寻找闭合键)。
 序号 16: CY 不等于 1, 说明有键按下, 转 N1。否则顺序执行。
 序号 17: 键号加 1。
 序号 18: 跳转至 L1 处继续寻找闭合键。
 序号 19: 将键号送累加器中。
 序号 20: 键号乘 2, 修正变址值。
 序号 21: 散转形成的键值入口地址表。
 序号 22: 转向 P1.0 号键功能程序 PR0。
 序号 23: 转向 P1.1 号键功能程序 PR1。
 序号 24: 转向 P1.2 号键功能程序 PR2。
 序号 25: 转向 P1.3 号键功能程序 PR3。
 序号 26: 转向 P1.4 号键功能程序 PR4。
 序号 27: 转向 P1.5 号键功能程序 PR5。
 序号 28: 转向 P1.6 号键功能程序 PR6。
 序号 29: 转向 P1.7 号键功能程序 PR7。
 序号 30: PR0 功能程序, 点亮 P2.0 发光管。
 序号 31: 调用延时子程序, 维持发光管点亮。
 序号 32: 跳转至主程序循环运行。
 序号 33: PR1 功能程序, 点亮 P2.0、P2.1 发光管。
 序号 34: 调用延时子程序, 维持发光管点亮。
 序号 35: 跳转至主程序循环运行。
 序号 36: PR2 功能程序, 点亮 P2.0、P2.1、P2.2 发光管。
 序号 37: 调用延时子程序, 维持发光管点亮。
 序号 38: 跳转至主程序循环运行。
 序号 39: PR3 功能程序, 点亮 P2.0、P2.1、P2.2、P2.3 发光管。
 序号 40: 调用延时子程序, 维持发光管点亮。
 序号 41: 跳转至主程序循环运行。
 序号 42: PR4 功能程序, 点亮 P2.0、P2.1、P2.2、P2.3、P2.4 发光管。
 序号 43: 调用延时子程序, 维持发光管

点亮。
 序号 44: 跳转至主程序循环运行。
 序号 45: PR5 功能程序, 点亮 P2.0、P2.1、P2.2、P2.3、P2.4、P2.5 发光管。
 序号 46: 调用延时子程序, 维持发光管点亮。
 序号 47: 跳转至主程序循环运行。
 序号 48: PR6 功能程序, 点亮 P2.0、P2.1、P2.2、P2.3、P2.4、P2.5、P2.6 发光管。
 序号 49: 调用延时子程序, 维持发光管点亮。
 序号 50: 跳转至主程序循环运行。
 序号 51: PR7 功能程序, 点亮 P2.0、P2.1、P2.2、P2.3、P2.4、P2.5、P2.6、P2.7 发光管。
 序号 52: 调用延时子程序, 维持发光管点亮。
 序号 53: 跳转至主程序循环运行。
 序号 54~58: 延时子程序。
 序号 59: 程序结束。

下面在 S2 试验板做一个行列式键盘输入实验。通电后 P0 口的数码管显示“0”, 按下 0 号键, P0 口的数码管显示“0”; 按下 1 号键, P0 口的数码管显示“1”; ……按下 9 号键, P0 口的数码管显示“9”; 按下 *、# 键, P0 口的数码管显示熄灭。S2 试验板的电路原理图刊登在《电子制作》2003 年第 1 期上。

在我的文档中建立一个文件目录 (S27), 然后建立一个 S27.uv2 的工程项目, 最后建立源程序文件 (S27.asm)。

输入下面的程序:

```

序号:1   ORG 0000H
2       AJMP MAIN
3       ORG 030H
4   MAIN:MOV P0,#0FFH
5   START:MOV P3,#0FH
6         MOV A,P3
7         CJNE A,#0FH,GO1
8         MOV A,R1
9         MOV DPTR,#TAB
10        MOVC A,@A+DPTR
11        MOV P0,A
12        ACALL DEL
13        AJMP START
14   GO1:  ACALL DEL
15        CJNE A,#0FH,GO2
16        AJMP START
    
```

```

17   GO2:  MOV R2,#0DFH
18         MOV R0,#00H
19   ST:   MOV P3,R2
20         MOV A,P3
21         JB ACC.0,ONE
22         MOV A,#01H
23         AJMP LKP
24   ONE:  JB ACC.1,TWO
25         MOV A,#04H
26         AJMP LKP
27   TWO:  JB ACC.2,THR
28         MOV A,#07H
29         AJMP LKP
30   THR:  JB ACC.3,NEXT
31         MOV A,#0AH
32   LKP:  ADD A,R0
33         CJNE A,#0BH,LKP1
34         MOV A,#00H
35   LKP1:MOV R1,A
36         AJMP START
37   NEXT:INC R0
38         CJNE R0,#03H,L1
39         MOV R0,#00H
40         AJMP ST
41   L1:  CJNE R2,#0DFH,L2
42         MOV R2,#0BFH
43         AJMP ST
44   L2:  CJNE R2,#0BFH,L3
45         MOV R2,#7FH
46         AJMP ST
47   L3:  CJNE R2,#7FH,RE
48         MOV R2,#0DFH
49   RE:  AJMP START
50   DEL:  MOV R7,#0DH
51   DEL1:MOV R6,#0FFH
52   DEL2:DJNZ R6,DEL2
53         DJNZ R7,DEL1
54         RET
55   TAB:DB 0C0H,0F9H,
          0A4H,0B0H,99H
56   DB 92H,82H,0F8H,80H,
          90H
57         END
    
```

编译通过后, 将 S27 文件夹中的 hex 文件烧录到 89C51 芯片中, 将芯片插入到 S2 型试验板上, 通电后 P0 口的数码管显示“0”, 按下 0 号键, P0 口的数码管显示“0”; 按下 1 号键, P0 口的数码管显示“1”; ……按下 9 号

MCU Application

键,P0口的数码管显示“9”;按下*、#键,P0口的数码管显示熄灭。完全达到设计目的。

下面我们对程序进行分析解释。

序号1(程序解释,以下同):程序开始。

序号2:跳转到MAIN主程序处。

序号3:主程序从地址30H开始。

序号4:熄灭P0口数码管。

序号5:置P3口为0FH,准备读取输入状态。

序号6:读取P3口输入状态。

序号7:若A不等于FFH,说明有键闭合,转GO1;否则无键闭合,顺序执行。

序号8:将寄存器R1内容送累加器A。上电时,R1内容为00H。

序号9:数码管的字段码数据表格首地址送数据指针DPTR。

序号10:查表获得的数据存累加器A。

序号11:将累加器A的内容送P0口显示。

序号12:调用延时子程序,维持数码管点亮。

序号13:跳转到START处循环执行。

序号14:调用延时子程序,避开按键抖动干扰。

序号15:再判键闭合,若A不等于FFH,说明有键闭合,转GO2;否则无键闭合,顺序执行。

序号16:跳转至主程序START处循环执行。

序号17:向寄存器R2送立即数DFH。

序号18:向寄存器R0送立即数00H。

序号19:R2内容送P3口,准备读取最左列的按键输入。

序号20:读取P3口输入至累加器A中。

序号21:1号键没有闭合转ONE处,否则顺序执行。

序号22:向累加器A送立即数01H。

序号23:跳转至LKP处。

序号24:4号键没有闭合转TWO处,否则顺序执行。

序号25:向累加器A送立即数04H。

序号26:跳转至LKP处。

序号27:7号键没有闭合转THR处,否则顺序执行。

序号28:向累加器A送立即数07H。

序号29:跳转至LKP处。

序号30:*号键没有闭合转NEXT处,否则顺序执行。

序号31:向累加器A送立即数0AH。

序号32:将R0内容与A内容相加,计算出键值,结果存A。

序号33:若A内容不为0BH,转LKP1;否则顺序执行。

序号34:清除累加器A。

序号35:累加器A内容送寄存器R1。

序号36:跳转到START循环执行。

序号37:R0加1。

序号38:若R0内容不为03H,转L1。

序号39:清除R0内容。

序号40:跳转到ST处进行下一列按键输入的判断。

序号41:若R2不等于DFH,转L2;否则顺序执行。

序号42:向R2送立即数BFH。

序号43:跳转到ST处。

序号44:若R2不等于BFH,转L3;否则顺序执行。

序号45:向R2送立即数7FH。

序号46:跳转到ST处。

序号47:若R2不等于7FH,转RE;否则顺序执行。

序号48:向R2送立即数DFH。

序号49:跳转到START处循环执行。

序号50~54:延时子程序。

序号55~56:数码管字段数据表。

序号57:程序结束。

下来再做两个关于键盘工作方式的实验。前面已介绍了键盘工作方式,但许多读者缺少感性认识,通过这两个实验可理解CPU对键盘的程序控制扫描与定时中断扫描的区别。

在S1试验板做一个键盘输入实验。通电后P0口的8个发光管点亮10秒钟,熄灭10秒钟,反复循环。用试验线置P3.0低电平后,要等待P0口的发光管在亮、灭转换期间,CPU才读入P3.0状态,使P1口的8个发光管点亮或熄灭。这种CPU对键盘的程序控制扫描方式,有时会因CPU在忙于处理其它事情而延误或遗漏了对键盘输入的反应。

在我的文档中建立一个文件目录(S28),然后建立一个S28.uv2的工程项目,最后建立源程序文件(S28.asm)。

输入下面的程序:

```

序号:1      ORG 0000H
2          AJMP MAIN
3          ORG 030H
4          MAIN:MOV P0,#00H
5          ACALL DEL10S
6          ACALL KEY
7          MOV P0,#0FFH
8          ACALL DEL10S
9          ACALL KEY
10         AJMP MAIN
11        KEY:JB P3.0,RE
12         CPL C
13         JC NEXT1
14         MOV P1,#00H
15         AJMP NEXT2
16        NEXT1:MOV P1,#0FFH
17        NEXT2:ACALL DEL1S
18        RE:  RET
19        DEL1S:MOV R5,#0FFH
20        F1:  MOV R6,#0FFH
21        F2:  DJNZ R6,F2
22         DJNZ R5,F1
23         RET
24        DEL10S:MOV R0,0AH
25        LOOP:ACALL DEL1S
26         DJNZ R0,LOOP
27         RET
28        END

```

编译通过后,将S28文件夹中的hex文件烧录到89C51芯片中,将芯片插入到S1型试验板上,通电后P0口的8个发光管点亮10秒钟,熄灭10秒钟,反复循环。用试验线一端置“0”,另一端触碰P3.0进行试验。试验中发现,只有连续触碰P3.0且要等待P0口的发光管10秒到后在亮、灭转换期间,才能使P1口的发光管点亮或熄灭。如果仅仅短暂地触碰P3.0,不能使P1口的发光管状态发生变化。

下面我们对程序进行分析解释。

序号1(程序解释,以下同):程序开始。
 序号2:跳转到MAIN主程序处。
 序号3:主程序从地址30H开始。
 序号4:点亮P0口发光二极管。
 序号5:调用延时子程序,维持点亮P0口发光二极管10秒。
 序号6:调用键盘扫描子程序。
 序号7:熄灭P0口发光二极管。
 序号8:调用延时子程序,维持点亮P0

口发光二极管 10 秒。
 序号 9:调用键盘扫描子程序。
 序号 10:跳转到 MAIN 处循环运行。
 序号 11:键盘扫描子程序开始,若 P3.0 为高电平(无键闭合)转标号 RE 处,否则顺序执行。
 序号 12:进位位 CY 取反。
 序号 13:若 CY 为 1 转 NEXT1,否则顺序执行。
 序号 14:P1 口置全 0, 点亮 8 个发光管。
 序号 15:跳转至 NEXT2。
 序号 16:P1 口置全 1, 熄灭 8 个发光管。
 序号 17:调用 1 秒延时子程序,维持发光管熄灭或点亮。
 序号 18:子程序返回。
 序号 19~23:1 秒延时子程序。
 序号 24~27:10 秒延时子程序。
 序号 28:程序结束。

下来在 S1 试验板做一个定时中断键盘输入实验。通电后 P0 口的 8 个发光管点亮 25 秒钟,熄灭 25 秒钟,反复循环。用试验线置 P3.0 低电平后,P1 口的 8 个发光管状态马上发生变化(点亮或熄灭)。这种 CPU 对键盘的定时中断扫描方式,只要定时时间足够短(几十 mS),就不会因 CPU 在忙于处理其它事情而延误或遗漏了对键盘输入的反应。

在我的文档中建立一个文件目录(S29),然后建立一个 S29.uv2 的工程项目,最后建立源程序文件(S29.asm)。

输入下面的程序:

```

序号:1      ORG 0000H
2          AJMP MAIN
3          ORG 000BH
4          AJMP CTC0
5          ORG 0030H
6  MAIN:MOV TMOD,#00H
7          MOV TL0,#0FFH
8          MOV TH0,#0FFH
9          SETB EA
10         SETB ET0
11         SETB TR0
12  GOON:MOV P0,#00H
13         ACALL DEL25S
14         MOV P0,#0FFH
15         ACALL DEL25S
    
```

```

16         AJMP GOON
17  CTC0:MOV TL0,#0FFH
18         MOV TH0,#0FFH
19         JB P3.0,RE
20         CLR TR0
21         CPL C
22         JC NEXT
23         MOV P1,#00H
24         AJMP NEXT1
25  NEXT:MOV P1,#0FFH
26  NEXT1:ACALL DEL1S
27         SETB TR0
28  RE:  RETI
29  DEL1S:MOV R5,#0FFH
30  F1:  MOV R6,#0FFH
31  F2:  DJNZ R6,F2
32         DJNZ R5,F1
33         RET
34  DEL25S:MOV R0,0AH
35  LOOP:ACALL DEL1S
36         DJNZ R0,LOOP
37         RET
38  END
    
```

编译通过后,将 S29 文件夹中的 hex 文件烧录到 89C51 芯片中,将芯片插入到 S1 型试验板上,通电后 P0 口的 8 个发光管点亮 25 秒钟,熄灭 25 秒钟,反复循环。用试验线一端置 '0',另一端触碰 P3.0 进行试验。试验发现,只要一触碰 P3.0,P1 口的发光管状态马上发生变化(点亮或熄灭),说明 CPU 很快就对键盘输入进行反应,这样就不会延误或遗漏了每一次的键盘输入事件。有些读者会问,上一个实验和这一个实验的延时子程序完全一样,为什么 10 秒变成了 25 秒呢?这主要由于在延时过程中 CPU 不断被 T0 定时中断打断而去判断键盘输入,这样就使延时时间加长。

下面我们对程序进行分析解释。

序号 1(程序解释,以下同):程序开始。
 序号 2:跳转到 MAIN 主程序处。
 序号 3:定时器 T0 的中断服务子程序入口地址。
 序号 4:跳转到定时器 T0 中断服务子程序(标号 CTC0 处)。
 序号 5:主程序从地址 30H 开始。
 序号 6:设置定时器 T0 方式 0。

序号 7、8:T0 载入定时初值(在晶振为 12MHz 定时约 8mS)。
 序号 9:总中断允许。
 序号 10:T0 开中断。
 序号 11:启动 T0。
 序号 12:点亮 P0 口发光二极管。
 序号 13:调用延时子程序,维持点亮 P0 口发光二极管 25 秒。
 序号 14:熄灭 P0 口发光二极管。
 序号 15:调用延时子程序,维持点亮 P0 口发光二极管 25 秒。
 序号 16:跳转到标号 GOON 处循环运行。
 序号 17:T0 定时中断子程序开始。
 序号 17、18:重装定时初值。
 序号 19:若 P3.0 为高电平(无键闭合)转标号 RE 处,否则顺序执行。
 序号 20:关闭定时器 T0。
 序号 21:进位位 CY 取反。
 序号 22:若 CY 为 1 转 NEXT,否则顺序执行。
 序号 23:P1 口置全 0, 点亮 8 个发光管。
 序号 24:跳转至 NEXT1。
 序号 25:P1 口置全 1, 熄灭 8 个发光管。
 序号 26:调用 1 秒延时子程序,维持发光管熄灭或点亮。
 序号 27:启动定时器 T0。
 序号 28:定时中断子程序返回。
 序号 29~33:1 秒延时子程序。
 序号 34~37:25 秒延时子程序。
 序号 38:程序结束。

(下一讲学习 LED 显示器接口技术的程序设计及有关实验)

配文优惠邮购(每次邮费保价费 12 元):Keil 51 Windows 集成开发环境(已汉化光盘,邮购代号:K1):46 元。TOP851 多功能编程器(邮购代号:B1):400 元。LED 输出试验板(邮购代号:S1):90 元。LED 数码管输出试验板(邮购代号:S2):140 元。5V 高稳定专用稳压电源(邮购代号:D1):35 元。。

邮购时只需在附言栏中写明邮购代号及数量并附上联系电话即可。

邮购地址:201103 上海市闵行区莲花路 2151 弄 57 号 201 室

联系人:吕超亚

电话:021-64066571 13044152947

技术支持 E-mail:zxh2151@sohu.com ◀