实验 6: 熟悉和掌握定时器/计数器应用操作

姓名：朱勇椿　学号：201411213004

2016 年 11 月 26 日

1　实验题目

1. 掌握微计算机定时器/计数器的硬、软件的设计。
2. 学习设计程序框图，编写控制定时器/计数器的应用程序。

2　实验内容

1. 设计外部按键电路图（参考实验 5 原理图），按键模拟产生外部计数脉冲。
2. 设计数器/定时器控制程序（优先选择汇编语言），要求如下：
   (a) 交通灯控制，南北车流量大，绿灯 10 秒，东西绿灯 1 秒，当东西车辆大于 3 个时，东西绿灯 5 秒，要求所有定时使用定时器 T1，对东西车辆的计数使用 T0，计数可为按键或红外模拟产生，仿真或写入验证是否正确。
   (b) 跑马灯程序，转圈依次亮一个灯，红、绿交替亮，每隔 2 秒换一个灯，每个灯亮时以 0.2 秒闪烁。要求 T0 定时 2 秒，T1 定时 0.2 秒，仿真或写入验证是否正确。
3. 设计蜂鸣器硬件电路图（参考实验 5 原理图）。
4. 编写演奏音乐程序，循环演奏一段音乐，仿真或写入检查是否正确。

3　实验过程

3.1　实验 1

交通灯控制，南北车流量大，绿灯 10 秒，东西绿灯 1 秒，当东西车辆大于 3 个时，东西绿灯 5 秒，要求所有定时使用定时器 T1, 对东西车辆的计数使用 T0 计数可为按键或红外模拟产生，仿真或写入验证是否正确。

```
01  TMOD_M  EQU 00010101B
02  I_TH0   EQU 0FFH
03  I_TL0   EQU 0FDH
04  I_TH1   EQU 03CH
05  I_TL1   EQU 0B0H
06  A_TIME1 EQU 20
07  A_TIME5 EQU 100
08  A_TIME10    EQU 200
09  A_LAMP  EQU P2
10  SN_G    EQU 10011001B
11  EW_G    EQU 01100110B
12  MARK1   bit 0h  ;0表示东西绿灯1表示南北绿灯
13  MARK2   bit 1h  ;0表示东西1s，1表示东西5s
14  DEL_LEN EQU 10H
15      ORG 0000H
16      LJMP    START
17      ORG 000BH
18      LJMP    T0_S
19      ORG 001BH
20      LJMP    T1_S
21      ORG 0030H
22  START:  LCALL   INIT
23  MLOOP:
24      SJMP    MLOOP
25  INIT:
```

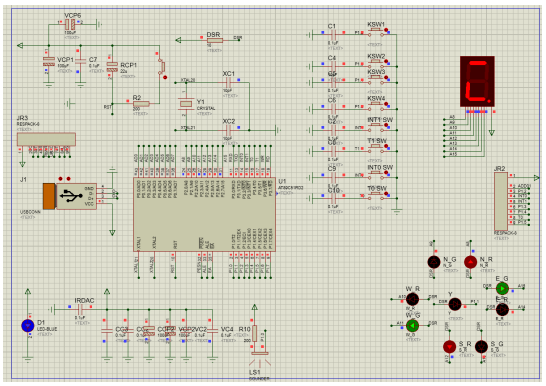
(a) 程序

```
26      MOV TMOD,#TMOD_M
27      MOV TH0,#I_TH0
28      MOV TL0,#I_TL0
29      MOV R0,#0
30      MOV TH1,#I_TH1
31      MOV TL1,#I_TL1
32      MOV A_LAMP,#EW_G
33      SETB    ET0
34      SETB    EA
35      SETB    TR0
36      SETB    ET1
37      SETB    TR1
38      RET
39  T0_S:
40      CLR EA
41      SETB    MARK2
42      MOV TH0,#I_TH0
43      MOV TL0,#I_TL0
44      SETB    EA
45      RETI
46  T1_S:
47      INC R0
48      JB  MARK1,TEN
49      JB  MARK2,FIVE
50  ONE:    CJNE    R0,#A_TIME1,EXIT
```

(b) 程序

```
51      SJMP    TIME
52  FIVE:   CJNE    R0,#A_TIME5,EXIT
53      SJMP    TIME
54  TEN:    CJNE    R0,#A_TIME10,EXIT
55  TIME:
56      JB  MARK1,EW
57  SN: MOV A_LAMP,#SN_G
58      SETB    MARK1
59      CLR MARK2
60      SJMP    EX
61  EW: MOV A_LAMP,#EW_G
62      CLR     MARK1
63  EX:
64      MOV R0,#0
65  EXIT:   MOV TH1,#I_TH1
66      MOV TL1,#I_TL1
67      RETI
68      END
```

(c) 程序



(d) 程序

正常状态东西绿灯亮 1s，南北绿灯亮 10s，用 t0 对东西绿灯的车辆进行计数，用按钮模拟车辆到来，如果车辆数量大于等于 3，那么东西绿灯此次亮 5s。在实验中一直结果不如预期，后来问了老师得知关于 EQU 这条指令其实就是把两个东西画上等号，我写的 MARK1 EQU 0B MARK2 EQU 0B，这样的操作实际上把 bit 存储区中的 0 的位置同时给了这两个标记，导致程序运行不正确。深刻理解了赋值，bit 存储区等概念。

3.2 实验 2

跑马灯程序，转圈依次亮一个灯，红、绿交替亮，每隔 2 秒换一个灯，每个灯亮时以 0.2 秒闪烁。要求 T0 定时 2 秒，T1 定时 0.2 秒，仿真或写入验证是否正确。

```
TMOD_W  EQU 00010001B
        ;GATE1=1
        ;C/T1=0
        ;M1=0,M0=1
I_TH0   EQU 03CH
I_TL0   EQU 0B0H
I_TH1   EQU 03CH
I_TL1   EQU 0B0H
A_TIME1 EQU 4
A_TIME0 EQU 40
A_LAMP  EQU P2
MARK    bit 0h  ;0表示灭1表示亮
LED     EQU 01111111B
    ORG 0000H
    LJMP    START
    ORG 000BH
    LJMP    T0_S
    ORG 001BH
    LJMP    T1_S
    ORG 0030H
START:  LCALL   INIT
MLOOP:
    SJMP    MLOOP
INIT:   MOV A_LAMP,#LED
    MOV R0,#0
```

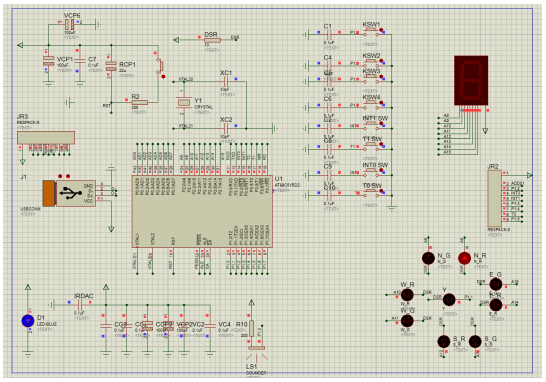
(e) 代码

```
    MOV R2,A_LAMP
    MOV R1,#0
    MOV TMOD,#TMOD_W
    MOV TH0,#I_TH0
    MOV TL0,#I_TL0
    MOV TH1,#I_TH1
    MOV TL1,#I_TL1
    SETB    ET0
    SETB    EA
    SETB    TR0
    SETB    MARK
    SETB    ET1
    SETB    TR1
    RET
T0_S:
    INC R0
    CJNE    R0,#A_TIME0,EXIT0
    MOV R0,#0
    MOV     A,R2
    RL      A
    MOV     R2,A
EXIT0:  MOV TH1,#I_TH1
    MOV TL1,#I_TL1
    RETI
T1_S:
```

(f) 代码

```
    INC R1
    CJNE    R1,#A_TIME1,EXIT1
    MOV R1,#0
    JB  MARK,MIE
    MOV A_LAMP,R2
    SETB    MARK
    SJMP    EXIT1
MIE:
    MOV A_LAMP,#0FFH
    CLR MARK
EXIT1:
    MOV TH1,#I_TH1
    MOV TL1,#I_TL1
    RETI
    END
```

(g) 代码



(h) 运行结果

设计思路为用 T0 来定时 2S，用 T1 来定时 0.2s，用 R2 来存储当前跑马灯应该处于的状态，全灭的状态不存储，当 T0 发生中断时，对 R2 进行移位操作，当 T1 发生中断时，进行亮灭切换。最终结果就是亮 0.2s 灭 0.2s 的交替，同时灯循环着亮灭。

3.3 实验 3

编写演奏音乐程序，循环演奏一段音乐，仿真或写入检查是否正确。

```c
#include<reg51.h>
#define uchar unsigned char
sbit speaker=P1^0;
uchar T_Mod=0x11;
uchar T10msL=0xD8;
uchar T10msH=0xEF;
uchar index;
uchar tint_c;
uchar note_l;
uchar note_h;
uchar ic_tmp=0;
uchar temp=0;
code uchar musicfreqh[]={
    0xF2,0xF3,0xF5,0xF5,0xF6,0xF7,0xF8,
    0xF9,0xF9,0xFA,0xFA,0xFB,0xFB,0xFC,0xFC,
    0xFC,0xFD,0xFD,0xFD,0xFD,0xFE,
    0xFE,0xFE,0xFE,0xFE,0xFE,0xFE,0xFE
    };
code uchar musicfreql[]={
    0x42,0xC1,0x17,0xB6,0xD0,0xD1,0xB6,
    0x21,0xE1,0x8C,0xD8,0x68,0xE9,0x5B,0x8F,
    0xEE,0x44,0x6B,0xB4,0xF4,0x2D,
    0x47,0x77,0xA2,0xB6,0xDA,0xFA,0x16};
code uchar musicn_del[]={100,100,100,100,100,100,100,100,100};
code uchar musicnotet[]={7,8,9,10,11,12,13,14,00};
```

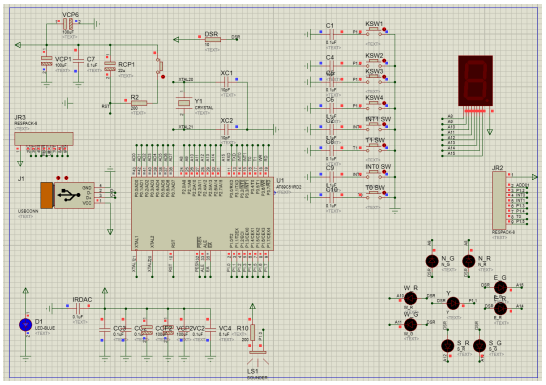
(i) 代码

```c
void timer_initial(void){
    TMOD=T_Mod;
    TL1=T10msL;
    TH1=T10msH;
    ET0=1;
    ET1=1;
}
void Time0(void) interrupt 1 using 1{
    TR0=0;
    TL0=note_l;
    TH0=note_h;
    speaker=~speaker;
    TR0=1;
}
void Time1(void) interrupt 3 using 2{
    TR1=0;
    TL1=T10msL;
    TH1=T10msH;
    ic_tmp++;
    TR1=1;
}
void main()
{
    while(1){
        index=0;
```

(j) 代码

```c
        timer_initial();
        for(index=0;musicnotet[index]!=0;index++){
            EA=0;
            TR0=0;
            TR1=0;
            ic_tmp=0;
            tint_c=musicn_del[index];
            temp=musicnotet[index];
            note_l=musicfreql[temp];
            note_h=musicfreqh[temp];
            TR0=1;
            TR1=1;
            EA=1;
            while(tint_c!=ic_tmp);
        }
    }
}
```

(k) 代码



(l) 运行结果

图中看不出程序运行的结果，仿真时可以听到声音，程序用的书上的 C 程序，没做改动，在选作 1 中进行音乐改动。仿真过程对蜂鸣器好像有要求，最开始用的 buzzer 蜂鸣器没有声音，换用了 sounder 蜂鸣器才有声音，网上说 sounder 蜂鸣器是专用于 51 单片机仿真。

4 选做题

4.1 4 段音乐

```c
#include<reg51.h>
#define uchar unsigned char
sbit speaker=P1^0;
sbit sw1=P1^2;
sbit sw2=P1^3;
sbit sw3=P1^4;
sbit sw4=P1^5;
uchar T_Mod=0x11;
uchar T10msL=0xD8;
uchar T10msH=0xEF;
uchar index;
uchar tint_c;
uchar note_l;
uchar note_h;
uchar ic_tmp=0;
uchar temp=0;
code uchar musicfreqh[]={
    0xF2,0xF3,0xF5,0xF5,0xF6,0xF7,0xF8,
    0xF9,0xF9,0xFA,0xFA,0xFB,0xFB,0xFC,0xFC,
    0xFC,0xFD,0xFD,0xFD,0xFD,0xFE,
    0xFE,0xFE,0xFE,0xFE,0xFE,0xFE,0xFE
    };
code uchar musicfreql[]={
    0x42,0xC1,0x17,0xB6,0xD0,0xD1,0xB6,
    0x21,0xE1,0x8C,0xD8,0x68,0xE9,0x5B,0x8F,
```

(m) 代码

```c
    0xEE,0x44,0x6B,0xB4,0xF4,0x2D,
    0x47,0x77,0xA2,0xB6,0xDA,0xFA,0x16);
code uchar musicn_del1[]={50,50,50,50,50,50,50,50,50,50,100,50,50,
100,50,50,50,50,100,100,50,50,50,50,100,100,100,100,100,100,100,100};
code uchar musicnotet1[]={7,8,9,7,7,8,9,7,9,10,11,9,10,11,11,12,
11,10,9,7,11,12,11,10,9,7,8,4,7,8,4,7,00}; //两只老虎
code uchar musicn_del2[]={50,50,100,50,50,100,50,50,50,25,25,50,50,100,50,50,50,25,25,
50,50,100,50,50,100,50,50,100,100,50,50,50,50,100,50,50,50,50,
100,50,50,50,50,50,50,50,50,50,50,50,50,100};
code uchar musicnotet2[]={11,11,11,11,11,11,11,9,11,12,11,9,8,9,11,11,9,11,9,8,
7,9,8,9,9,8,12,7,8,12,12,11,9,12,11,11,9,8,9,
11,11,9,8,9,11,9,8,9,5,7,8,9,7,00};   //卖报歌
code uchar musicn_del3[]={50,50,50,50,50,50,100,50,50,50,50,50,100,50,50,50,50,50,50,100,
50,50,50,50,100,50,50,50,50,50,50,100,50,50,50,50,100,
50,50,50,50,50,50,100,50,50,50,50,100};
code uchar musicnotet3[]={11,9,11,9,11,9,7,8,10,9,8,11,11,9,11,9,11,9,7,
8,10,9,8,7,8,8,10,10,9,7,11,8,10,9,8,11,
11,9,11,9,11,9,7,8,10,9,8,7,00};    //粉刷匠
code uchar musicn_del4[]={50,50,50,50,50,50,100,50,50,50,50,100,100,
50,50,50,50,50,50,100,50,50,50,50,50,100,100,
50,50,50,50,50,50,100};
code uchar musicnotet4[]={11,12,11,12,11,12,11,11,14,13,12,11,9,
11,11,9,10,11,11,9,11,11,9,10,11,11,9,
7,10,9,8,7,8,7,00};   //我朋友
void timer_initial(void){
```

(n) 代码

```c
    TMOD=T_Mod;
    TL1=T10msL;
    TH1=T10msH;
    ET0=1;
    ET1=1;
}
void Time0(void) interrupt 1 using 1{
    TR0=0;
    TL0=note_l;
    TH0=note_h;
    speaker=~speaker;
    TR0=1;
}
void Time1(void) interrupt 3 using 2{
    TR1=0;
    TL1=T10msL;
    TH1=T10msH;
    ic_tmp++;
    TR1=1;
}
void main()
{
    int mark=1;
    int keep=1;
    int t=0;
```

(o) 代码

```c
    while(1){
        index=0;
        timer_initial();
        if(mark==1){keep=1;t=musicnotet1[index];}
        else if(mark==2){keep=2;t=musicnotet2[index];}
        else if(mark==3){keep=3;t=musicnotet3[index];}
        else if(mark==4){keep=4;t=musicnotet4[index];}
        for(index=0;musicnotet3[index]!=0;index++){
            if(!sw1){
               mark=1;
            }
            if(!sw2){
               mark=2;
            }
            if(!sw3){
               mark=3;
            }
            if(!sw4){
               mark=4;
            }
            EA=0;
            TR0=0;
            TR1=0;
            ic_tmp=0;
            if(keep==1){
```

(p) 代码

```c
                tint_c=musicn_del1[index];
                temp=musicnotet1[index];
                t=musicnotet1[index];
            }
            else if(keep==2){
                tint_c=musicn_del2[index];
                temp=musicnotet2[index];
                t=musicnotet2[index];
            }
            else if(keep==3){
                tint_c=musicn_del3[index];
                temp=musicnotet3[index];
                t=musicnotet3[index];
            }
            else if(keep==4){
                tint_c=musicn_del4[index];
                temp=musicnotet4[index];
                t=musicnotet4[index];
            }
            note_l=musicfreql[temp];
            note_h=musicfreqh[temp];
            TR0=1;
            TR1=1;
            EA=1;
            while(tint_c!=ic_tmp);
```

(q) 代码

程序中用 sw_1-4 来分别对应 4 个开关，用 mark 来标记下一首歌，用 keep 来标记当前歌曲，歌曲分别是两只老虎，卖报歌，粉刷匠，找朋友，按下按钮时不会立即切歌，而是等这首歌放完，下一首歌进行改变。

5 问题反思

要理解汇编中的 EQU 指令具体实现的是什么操作，理解 51 单片机中的存储区，bit 存储区和寄存器存储器是分开的，以及定时器和计数器的工作原理，定时器是当时间到了就产生中断，计数器是数字溢出产生中断。

6 经验感想

上课时问了老师蜂鸣器和定时器联合工作的原理，明白了定时器让蜂鸣器工作实际上就是通过改变信号周期来改变频率来控制蜂鸣器发出不同的声音。本次实验后两个程序用的 C 语言，感觉 C 语言确实比汇编更容易看懂，逻辑更清楚。