

华中科技大学

《开放式单片机应用系统专题设计》

专题名称：演讲定时器

班级：电子信息类光电 1802 班

姓名：马笑天

学号：U201813654

指导教师：吴裕斌 曹丹华

光学与电子信息学院

2020.5

一、任务要求

1. 计时的精度为 1 秒。
2. 可通过键盘设置规定讲演时间，最大讲演时间为 99 分 59 秒。
3. 有启动键、暂停键
4. 时间过半时有声音提示。（1 秒）
5. 正常讲演时黄灯（D9）每隔 3 秒闪烁一次（亮 1 秒，熄 2 秒），同时数码管剩余时间。
6. 讲演结束前一分钟有声音提示，同时黄灯闪烁频率逐渐加快，到达定时时间时黄灯变成常亮，蜂鸣器长鸣一声（2 秒）。
7. 计时到达 0 后，计时方式变为加计数，黄灯每隔 2 秒闪烁 1 次，直到计数值到最大值。

二、功能特点与使用说明

1. 功能特点

（1）整个程序通电后可以循环使用，无需每次使用结束进行 RESET 功能；通过 KINT 键集成了长短按，实现功能的跳转，并辅以 0-9 键盘的快速输入，这些大大方便了用户操作；

（2）在按 KINT 键时有声音和灯光的反馈，设定时间时选位的闪烁等功能，在操作时具有提示用户的功能；

（3）输入时间过程中的多键位同时按下的保护、在 0000 状态下禁止启动倒计时保护以及超时最大值防溢出保护，最大限度防止用户误操作；

（4）将 F1-F6 的入口也一并给出，方便将来功能的拓展。

2. 使用说明

通电后，数码管显示“0000”，长按 KINT 键唤起时间设定，从最高位开始，待设定位呈现闪烁状，需要用户依据自身需求按下 0-9 中一个键，该位即设定完成，系统将自动跳转到下一位。高 2 位为分钟，低 2 位为秒。若依照这种模式完成 4 位的设定后，系统仍会自动跳转回最高位循环往复，直至再次长按 KINT 键完成时间的设定。

只有在完成时间设定后，单次短按 KINT 键，系统才开始倒计时。倒计时过程中可以通过单次短按 KINT 键来启动、暂停计时，也可以在暂停的情况下通过长按该键重新设定时间。在运行过程中，处于特定时间点和时间段时，D9 黄灯以及蜂鸣器会由相应的闪烁和鸣响。超时计时模式下，当时间大于等于 99 分 59 秒时，系统采取溢出保护提醒，此时数码管显示“End. ”，同时 D9 灯常亮，蜂鸣器长响。此时可以 RESET 使系统复位，也可以长按 KINT 键进行时间的重设。

注：（1）短按 KINT，蜂鸣器会微微短暂响起；长按 KINT，蜂鸣器微微短暂响起的同时，D9 会微微短暂亮起，用以提示用户；

（2）输入时间过程中，按下 F1-F6 以及短按 KINT 键均视为无效操作；

（3）若在输入 0-9 时有多个按键按下，则视第一个按下的键为输入的值；

（4）无法在设定时间为 0000 的状态下长按 KINT 开始倒计时；

（5）在倒计时的状态下无法通过长按 KINT 进行时间的设定。

三、方案设计（含设计思路、分析与计算、资源分配）

1. 设计思路

在宏观上，整个程序作为一个由 4 状态的状态机组成，循环往复。图 1 为程序的状态图。应当指出的是，在各个状态中，可能出现不同的函数，但是它们共同构成了这个状态的完整性。

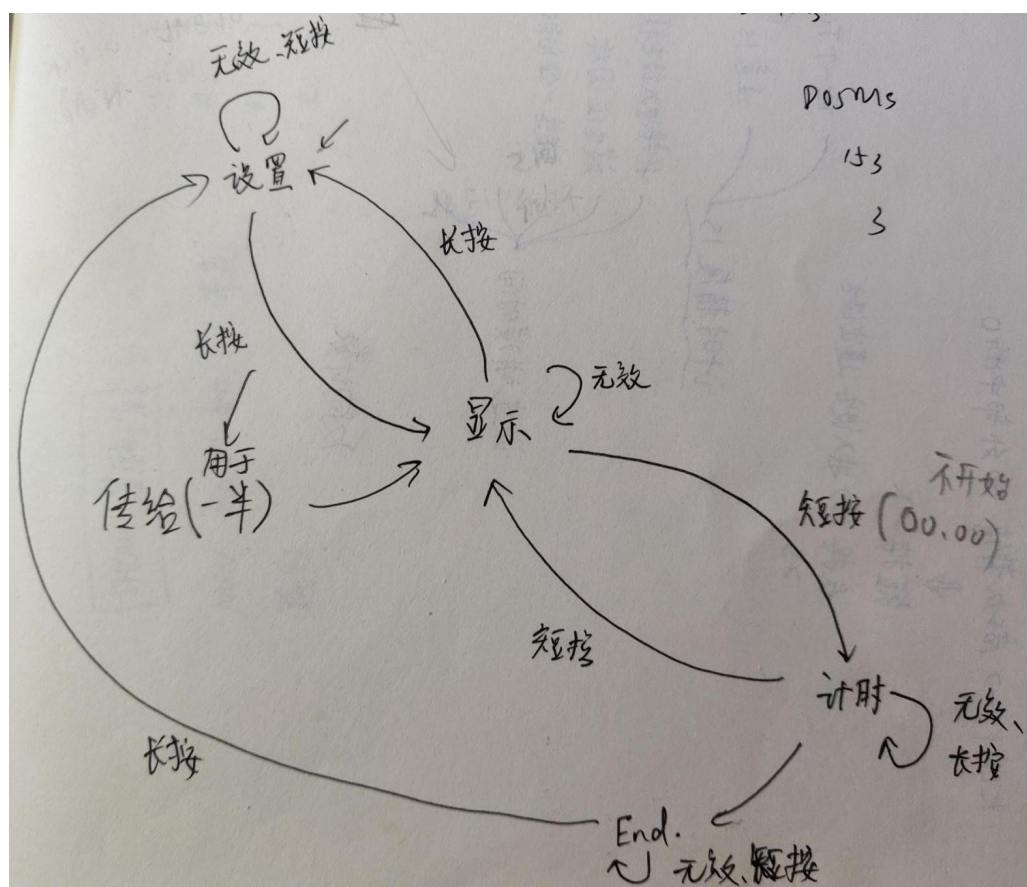


图 1

更具体的，根据任务要求，思路如下。本系统要求计时精度为 1s，则设计单独用 1 个定时器循环计时，应注意 1s 的精度。本着精简、易用的设计理念，将 0-9 用于时间设定，用 KINT 键的短按来执行启动、暂停，长按来执行唤起时间设定、完成时间设定的任务，同时有声光的按键反馈。为了防止

因误操作导致的问题，应当设定相应的保护机制。硬件上保留出 F1-F6，代码段给出其相应的接口，可用于未来其他功能拓展。

2. 分析与计算

本系统采用内部晶振，频率为 24.5MHz，进行 4 分频后作为系统时钟频率 6.125MHz。

在进行 1s 定时过程中，使用 Timer0，原因是它在定时器中具有最高的优先级，这将保证它的请求能够及时得到处理，确保没有等待时间，进而保证精度，并且它可以自动清理中断请求标志信号。F310 中允许的定时器 0 的分频情况如下图 2 所示。原则上，用尽可能少的循环次数，甚至是仅用一次就完成 1s 的计时是首选方案，但考虑到 6.125MHz 的时钟频率以及最大 16 位的计时容量，我们最终采用了 SYSCLK/4 的选项，这样在最少 24 次循环下便可完成 1s 定时。考虑到整除等因素，最终选择 1s 内循环 25 次，单次计数 61250 次，则计数初值为 $(65536-61250)_D=(4286)_D=(10BE)_H$ ，在随后的测试中，发现当 0FB4H 时，定时的精度最高，故最终设定为 0FB4H。

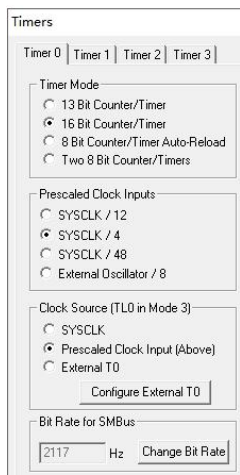


图 2

在数码管刷新速度方面，利用人眼的视觉暂留效应，能够实现动态扫描。设计的扫描时间 0.5ms，4 个管扫描完后，再等待 3ms，最终的一个刷新周期是 5ms，实现了 200Hz 的扫描频率，保障了视觉效果的同时，也尽可能降低在拍摄视频时的闪烁感。

在判断 KINT 是否按下，长按还是短按的问题上，我们利用外部中断 1 和 Timer1 的配合实现。Timer1 设置为 10ms 的模式，这样不仅能够提供 KINT 键消抖，而且也能够通过时间的累加来判断进行了长按还是短按。通过 PSW 中的 F0 与 F1 判断，默认状态时均为零。KINT 键按下时开始计时，如果发现小于 10ms，则视作无效，不改变 F0、F1 的值；KINT 键按下时间在 10ms—2s 之间时，F0 改为 1，F1 仍为 0，表示一次短按；KINT 键按下事件在 2s 以上，F0 改为 1 的同时，F1 也变为 1，表示一次长按。然后对 F0、F1 进行条件判断，从而提示程序进行相应的跳转，之后这些标志位又会被重新置零。Timer1 的 10ms，同样考虑到 6.125MHz 的时钟频率以及最大 16 位的计时容量，F310 中允许

的定时器 1 分频如下图 3 所示，我们最终采用了 SYSCLK/4 的选项，定时器频率为 1.53125MHz，使其一次即完成 10ms 的定时，单次计数 15312，则计数初值为 $(65536-15312)_D=(50224)_D=(C430)_H$ ，在随后的测试中，发现当 C2F7H 时，效果更好，故最终设定为 C2F7H。

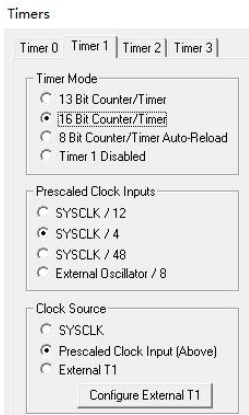


图 3

在选位闪烁方面，是在 100 个显示周期即 0.5 秒内被选中位亮，在另外的 0.5 秒内灭。这个实现方法是赋值一个变量为 200，并在每个显示周期内与 100 相减，大于 0 亮，小于 0 灭，并且这个变量每个显示周期都会减一，一直循环，直到变量等于 0，此时重新赋值为 200，即可完成选位闪烁功能。

在倒计时 1 分之内的 D9 渐快闪烁过程中，采用两秒一加速的策略，这里充分利用了 Timer0 中的 25 循环为 1s 的特点，实现渐快闪烁的效果。

在处理时间过半蜂鸣 1s 时，应当在用户设定好时间后，计算并存入其半值，在之后的倒计时模块中，将现在时间与半值进行比较。当到达半值时，蜂鸣 1s 即可实现该功能。这种方法适用于各种时间触发功能的实现，因此略过其他类似功能的实现方法的描述。

3. 资源分配

概述:堆栈指针 SP 从 50H 开始,使用了第 0 组寄存器中的所有工作寄存器(8 个),RAM 区的 30H-40H (17 个),PSW 中的 F0、F1 以及位寻址区的 20H.0-20H.5 (8 个),通过编译后的信息得知: Code=1934.

图 4 详细地描述了各个存储单元的使用情况，不再赘述。

中断方面，使用了外部中断 1 以及 Timer0、Timer1 两个定时器。

```

;初始化
MOV SP, #50H
MOV R7, #0      ;R7 临时变量 SETTING部分使用
MOV R6, #0      ;R6 临时变量 DETECT部分使用
MOV R5, #150    ;R5 用于记录KINT是否长按
MOV R4, #0      ;R4 临时变量 显示部分共同使用
MOV R3, #00H    ;R3 分钟
MOV R2, #00H    ;R2 秒钟
MOV R1, #0      ;R1 临时变量
MOV R0, #30H    ;R0 指向30H-33H, 显示缓冲区

MOV 30H, #0
MOV 31H, #0      ;31H&30H分别为秒钟的十位、个位
MOV 32H, #0
MOV 33H, #0      ;33H&32H分别为分钟的十位、个位
MOV 34H, #200    ;用于控制用户设定时的位选闪烁
MOV 35H, #25     ;用于1s的定时
MOV 36H, #0      ;用于10ms的按键延时
MOV 37H, #0      ;用于存放用户设定的按键值
MOV 38H, #3      ;用来指示设定位的位选, 默认最先设定最高位
MOV 39H, #0      ;用来指示数字键按下状态
MOV 3AH, #0
MOV 3BH, #0      ;用来存放时间的半值, 3BH用来存分, 3AH用来存秒
MOV 3CH, #0      ;临时变量
MOV 3DH, #0      ;普通闪烁临时变量
MOV 3EH, #1      ;超时闪烁变量
MOV 3FH, #31     ;渐快闪烁变量
MOV 40H, #31     ;渐快闪烁变量
CLR F0           ;用户标志位F0, 用于提示是(1)否(0)按下按键
CLR F1           ;用户标志位F1, 用于提示长按(1)或短按(0)键
CLR 20H.0        ;暂存用户标志位F0, 用于提示是否按下按键
CLR 20H.1        ;暂存用户标志位F1, 用于提示长按或短按键
CLR 20H.2        ;提示是(1)否(0)进入超时模式
CLR 20H.3        ;提示是(1)否(0)进入渐快模式
CLR 20H.4        ;渐快模式的加速标志, 2秒一周期
CLR 20H.5        ;渐快模式亮灭的控制位

```

图 4

四、流程图（主程序、子程序的详细流程）

图 5 为主程序，图 6-图 9 分别为子程序模块

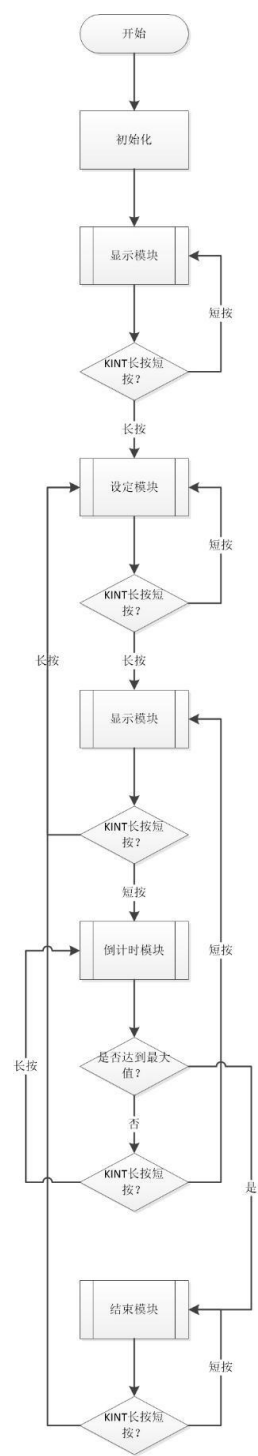


图 5

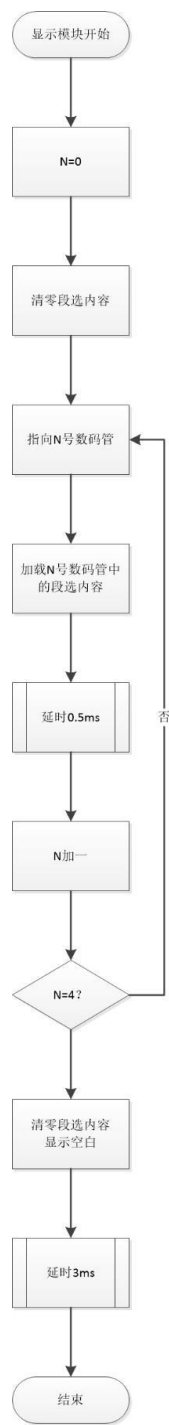


图 6

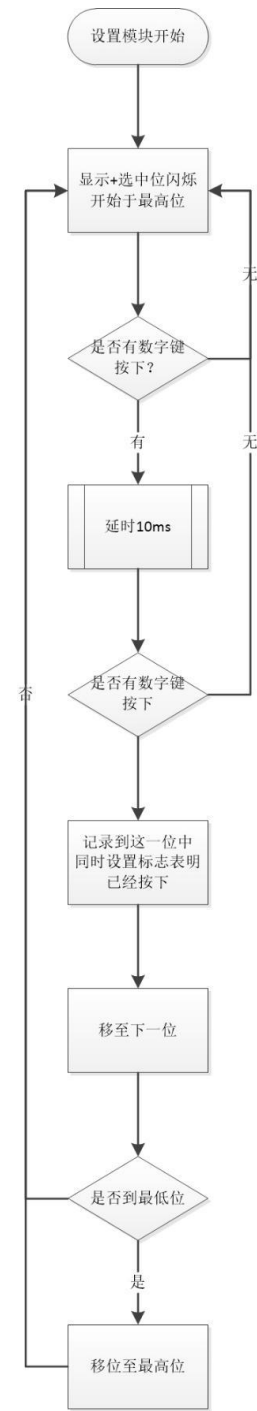


图 7

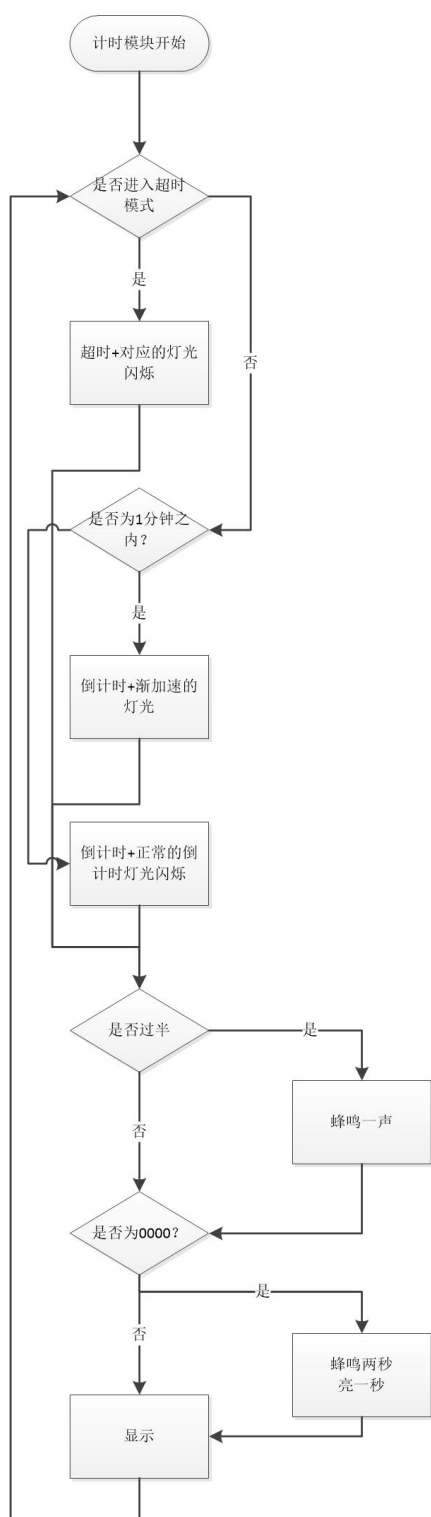


图 8

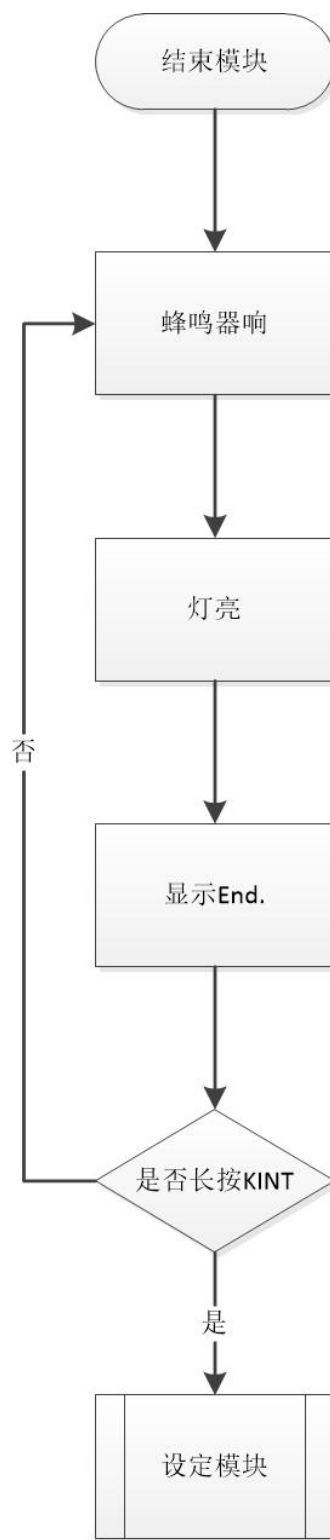


图 9

五、源程序（含文件头说明、资源使用说明、语句行注释）

```
;File name:    Timer.asm
;Description:  完成了任务三当中的功能
;Designed by:  Ma Xiaotian
;Date:2020-5-30
;-----
#include (C8051F310.inc)
;D9 BEEP KINT
LED      BIT P0.0      ;D9 黄灯，低电平有效
BEEP     BIT P3.1      ;蜂鸣器，高电平有效
KINT     BIT P0.1      ;独立按键，按下为低电平
;LED 指示灯阵列
DAT_IN   BIT P3.3
CLK      BIT P3.4

    ORG 0000H
    LJMP HERE

    ORG 000BH
    LJMP TIMER0      ;TIMER0 1s 的授时 设置高优先级

    ORG 0013H
    LJMP DETECT      ;/INT1 检测 KINT 键

    ORG 001BH
    LJMP TIMER1      ;TIMER1 检测按键的下按 单次循环约 10ms

    ORG 0100H
;-----

;初始化 C8051F310, SYSCLK=24.5/4=6.125MHz（内部晶振）
;关 WDT
;推挽、开漏设定

;T1,16bits Timer, TCLK=SYSCLK/4, TH1=88H, TL1=05FH
;T1 一个周期近似为 20ms, 为按键消抖以及长短按授时
;T0,16bits Timer, TCLK=SYSCLK/4, TH0=10H, TL0=0DDH
;T0 为数码管提供 1s 的授时, 精度约是每小时快 8s
;（以手机内置定时器为金标设备型号 OnePlus 6, 固件氢 OS 10.0.4）
;-----
//初始化
HERE: LCALL Init_Device
      ;关闭 D1-D8, 关闭蜂鸣器
      CLR BEEP
      SETB DAT_IN
      MOV R1, #8
INIT: CLR CLK
      SETB CLK
      DJNZ R1, INIT

;资源使用说明
```

;使用了片内晶振，并 4 分频作为系统时钟。1 个外部中断，2 个定时器
;其余存储空间的使用情况如下

```
MOV SP, #50H
MOV R7, #0           ;R7  临时变量 SETTING 部分使用
MOV R6, #0           ;R6  临时变量 DETECT 部分使用
MOV R5, #150         ;R5  用于记录 KINT 是否长按
MOV R4, #0           ;R4  临时变量 显示部分共同使用
MOV R3, #00H         ;R3  分钟
MOV R2, #00H         ;R2  秒钟
MOV R1, #0           ;R1  临时变量
MOV R0, #30H         ;R0  指向 30H-33H, 显示缓冲区

MOV 30H, #0
MOV 31H, #0           ;31H&30H 分别为秒钟的十位、个位
MOV 32H, #0
MOV 33H, #0           ;33H&32H 分别为分钟的十位、个位
MOV 34H, #200         ;用于控制用户设定时的位选闪烁
MOV 35H, #25          ;用于 1s 的定时
MOV 36H, #0           ;用于 10ms 的按键延时
MOV 37H, #0           ;用于存放用户设定的按键值
MOV 38H, #3           ;用来指示设定位的位选，默认最先设定最高位
MOV 39H, #0           ;用来指示数字键按下状态
MOV 3AH, #0
MOV 3BH, #0           ;用来存放时间的半值，3BH 用来存分，3AH 用来存秒
MOV 3CH, #0           ;临时变量
MOV 3DH, #0           ;普通闪烁临时变量
MOV 3EH, #1           ;超时闪烁变量
MOV 3FH, #31          ;渐快闪烁变量
MOV 40H, #31          ;渐快闪烁变量
CLR F0                ;用户标志位 F0，用于提示是(1)否(0)按下按键
CLR F1                ;用户标志位 F1，用于提示长按(1)或短按(0)键
CLR 20H.0             ;暂存用户标志位 F0，用于提示是否按下按键
CLR 20H.1             ;暂存用户标志位 F1，用于提示长按或短按键
CLR 20H.2             ;提示是(1)否(0)进入超时模式
CLR 20H.3             ;提示是(1)否(0)进入渐快模式
CLR 20H.4             ;渐快模式的加速标志, 2 秒一周期
CLR 20H.5             ;渐快模式亮灭的控制位
```

;F0 为 0 时，没有按下；F0 为 1 时且 F1 为 0 时，短按；F0 为 1 时且 F1 为 1 时，长按

;

//主程序块

```
MAIN: LCALL SHOW
      JNB F0, MAIN
      JBC F1, JUMP2
      SJMP JUMP1
```

```
JUMP2: MOV 38H, #3
      LCALL SETTING
      MOV 3FH, #31
```

```

    MOV 40H, #31
//求取设定时间的一半, 存入 3BH, 3AH 中
    MOV A, 31H
    MOV B, #10
    MUL AB
    ADD A, 30H           ;秒钟的 10 进制数
    MOV B, #2
    DIV AB               ;商在 A 中, 是十进制数
    MOV B, #10
    DIV AB               ;十位在 A 中, 个位在 B 中
    MOV 3AH, B
    MOV B, #16
    MUL AB               ;结果在 A 中
    ADD A, 3AH
    MOV 3AH, A           ;暂时算出的秒钟的结果以 BCD 形式存入了 3AH 中
    MOV A, 33H
    MOV B, #10
    MUL AB
    ADD A, 32H           ;分钟的 10 进制数
    MOV B, #2
    DIV AB               ;商在 A 中, 余数在 B 中(十进制数)
    MOV 3CH, A
    MOV R1, B
    CJNE R1, #1, NEXT6;整除
    MOV A, 3AH           ;余 1, 则秒钟加 30H
    ADD A, #30H
    MOV 3AH, A           ;更新了 3AH 中的 BCD 码
NEXT6:MOV B, #10
    MOV A, 3CH
    DIV AB               ;十位在 A 中, 个位在 B 中
    MOV 3BH, B
    MOV B, #16
    MUL AB               ;结果在 A 中
    ADD A, 3BH
    MOV 3BH, A           ;秒钟的结果以 BCD 码形式存入了 3BH 中
//
    CLR 20H.2           ;将超时计数标志位清零
    JBC F1, MAIN

JUMP1:LCALL COUNT
    JNB 20H.2, HERE21   ;最后的结束
    CJNE R3, #99H, HERE21
    CJNE R2, #59H, HERE21
    SJMP JUMP3
HERE21:JNB F0, JUMP1
    JBC F1, JUMP1
    SJMP MAIN

JUMP3:CLR TR0
    LCALL BYE
    JNB F0, JUMP3
    JBC F1, JUMP2

```

SJMP JUMP3

//子程序块

SHOW: ;固定值显示模式

MOV P1, #0

CLR F0

MOV A, R0

ANL A, #0FH

SWAP A

RL A

RL A

MOV R1, A

MOV A, P0

ANL A, #3FH

ORL A, R1

MOV P0, A ;位选传送, 显示

MOV A, @R0

MOV DPTR, #0700H

MOVC A, @A+DPTR ;选择相应的数字

MOV P1, A ;段选传送

LCALL D05MS

INC R0

MOV A, R0

CJNE A, #34H, SHOW

MOV P1, #0

MOV R0, #30H

MOV R4, #6 ;再延时 3ms, 凑够一个周期 5ms

LOOP1:LCALL D05MS

DJNZ R4, LOOP1

JNB F0, EOC3 ;禁止从 0000 开始

CJNE R3, #0, EOC3

CJNE R2, #0, EOC3

JB F1, EOC3

CLR F0

EOC3:

RET

SETTING: ;显示+闪烁位选设定

;显示

CLR F0

STA1:MOV P1, #0

MOV A, R0

ANL A, #0FH

SWAP A

RL A

RL A

MOV R1, A

MOV A, P0

ANL A, #3FH

ORL A, R1

```

MOV P0, A           ;位选传送，显示
MOV A, R0
ANL A, #0FH
CJNE A, 38H, HERE15;无需闪烁的位则跳过下面的语句块
DEC 34H
MOV A, 34H
CLR C
SUBB A, #100
JC HERE15
MOV P1, #0          ;闪烁
CJNE A, #9CH, HERE16
MOV 34H, #200
AJMP HERE16

HERE15:MOV A, @R0
MOV DPTR, #0700H
MOVC A, @A+DPTR      ;选择相应的数字
MOV P1, A            ;段选传送

HERE16:
LCALL D05MS
INC R0
MOV A, R0
CJNE A, #34H, STA1
MOV P1, #0
MOV R0, #30H
MOV R4, #6           ;再延时 3ms，凑够一个周期 5ms

LOOP2:
LCALL D05MS
DJNZ R4, LOOP2

;按键检测部分
MOV A, 39H
JNZ RELAY1           ;如果 39H=1（按下），就不再进行数字键值检测，防串键
ANL P2, #0F0H        ;行线同时输出低电平
MOV A, P2
CJNE A, #0F0H, TEST1;如果 KI 出现了低电平，则说明可能有按键按下
JB F1, RELAY3
SJMP STA1            ;否则都为高电平，则说明没有按键按下，返回开头

TEST1:
LCALL SHOW
LCALL SHOW
MOV A, P2
CJNE A, #0F0H, NEXT1 ;KI 仍出现低电平，则说明确实有按键按下
SJMP STA1            ;否则都为高电平，则说明误触，再次返回开头

NEXT1:
JNB P2.4, COLUMN0
JNB P2.5, COLUMN1
JNB P2.6, COLUMN2
JNB P2.7, COLUMN3

COLUMN0:
ORL P2, #0FH
CLR P2.0
JNB P2.4, K0

```

```

SETB P2.0
CLR P2.1
JNB P2.4, K1
SETB P2.1
CLR P2.2
JNB P2.4, K2
SETB P2.2
CLR P2.3
JNB P2.4, K3
RELAY1:AJMP HERE9      ;接续跳转
COLUMN1:
    ORL P2, #0FH
    CLR P2.0
    JNB P2.5, K4
    SETB P2.0
    CLR P2.1
    JNB P2.5, K5
    SETB P2.1
    CLR P2.2
    JNB P2.5, K6
    SETB P2.2
    CLR P2.3
    JNB P2.5, K7
RELAY3:AJMP HERE11
COLUMN2:
    ORL P2, #0FH
    CLR P2.0
    JNB P2.6, K8
    SETB P2.0
    CLR P2.1
    JNB P2.6, K9
    SETB P2.1
    CLR P2.2
    JNB P2.6, KA
    SETB P2.2
    CLR P2.3
    JNB P2.6, KB
COLUMN3:
    ORL P2, #0FH
    CLR P2.0
    JNB P2.7, KC
    SETB P2.0
    CLR P2.1
    JNB P2.7, KD
    SETB P2.1
    CLR P2.2
    JNB P2.7, KE
    SETB P2.2
    CLR P2.3
    JNB P2.7, KF
K0:    MOV 37H, #0
        LJMP HERE8
K1:    MOV 37H, #1
        LJMP HERE8
K2:    MOV 37H, #2
        LJMP HERE8

```

```

K3:    MOV 37H, #3
        LJMP HERE8
K4:    MOV 37H, #4
        LJMP HERE8
K5:    MOV 37H, #5
        LJMP HERE8
K6:    MOV 37H, #6
        LJMP HERE8
K7:    MOV 37H, #7
        LJMP HERE8
K8:    MOV 37H, #8
        LJMP HERE8
K9:    MOV 37H, #9
        LJMP HERE8
KA:    MOV 37H, #0EH
        LJMP HERE9
KB:    MOV 37H, #0EH
        LJMP HERE9
KC:    MOV 37H, #0EH
        LJMP HERE9
KD:    MOV 37H, #0EH
        LJMP HERE9
KE:    MOV 37H, #0EH
        LJMP HERE9
KF:    MOV 37H, #0EH ;这里 KA-KF 没有合并写, 为了未来添加功能时找入口方便
        LJMP HERE9 ;统一设为 0EH 的输出, 方便下面的判断
RELAY2: AJMP STA1

```

```

HERE8: MOV A, 38H
        CJNE A, #1, HERE12 ;控制当 38H=1 时, 只有 0-5 有效, 其余的均无效
        MOV A, 37H
        CJNE A, #5, HERE13
        SJMP HERE12 ;5 合法
HERE13: JC HERE12 ;0-4 合法
        SJMP HERE9 ;其余的不合法
HERE12: MOV 39H, #1 ;已按下有效的键, 此时 37H 和 39H 都填入了相应的值
HERE9:  ANL P2, #0F0H ;行线同时输出低电平
        MOV A, P2
        CJNE A, #0F0H, RELAY2 ;当有按键按下时, 为了保证显示, 仍返回 SETTING

        MOV 39H, #0 ;恢复为未按下的状态
        MOV A, 37H ;ABCDEF 的筛选
        CJNE A, #0EH, HERE14
        SJMP RELAY2 ;ABCDEF 则回到开头, 视作无效
HERE14: MOV A, 38H ;当松开后, 根据键入值修改相应位 (3210)
        JZ ZERO
        CJNE A, #3, NEXT2
THREE: MOV 33H, 37H ;37H 写入 33H 中 (分钟高位)
        MOV A, 37H
        SWAP A
        ANL 03H, #0FH
        ORL 03H, A ;37H 写入 R3 高 4 位中
        SJMP HERE10

```

```

ZERO: MOV 30H, 37H    ;37H 写入 30H 中 (秒钟低位)
      MOV A, 37H
      ANL 02H, #0F0H
      ORL 02H, A      ;37H 写入 R2 低 4 位中
      SJMP HERE10
NEXT2: CJNE A, #2, ONE
TWO:  MOV 32H, 37H    ;37H 写入 32H 中 (分钟低位)
      MOV A, 37H
      ANL 03H, #0F0H
      ORL 03H, A      ;37H 写入 R3 低 4 位中
      SJMP HERE10
ONE:  MOV 31H, 37H
      MOV A, 37H
      SWAP A
      ANL 02H, #0FH   ;37H 写入 R2 高 4 位中
      ORL 02H, A
HERE10: DEC 38H
      MOV A, 38H
      CJNE A, #0FFH, HERE11
      MOV 38H, #3      ;等于 0 时要注意 38H 恢复为 3, 进行循环输入
HERE11: JNB F0, RELAY2 ;无效 KINT 将继续程序内循环, 从而保证选位的连续
      JNB F1, RELAY2 ;短按 KINT 将继续程序内循环
      RET              ;长按 KINT 则会跳出程序

```

```

COUNT: ; 倒计时+超时计数
      SETB TR0
      JB 20H.2, TIMEOUT
      CJNE R3, #1, NEXT7
      CJNE R2, #0, NEXT7
      SETB BEEP      ;一分钟的时候响一下
NEXT7: CJNE R3, #0, FLASH;跳到普通闪烁
      SJMP ACCEL      ;跳到渐快闪烁
TIMEOUT: MOV A, 3EH   ;超时模式
      CJNE A, #2, NEXT8
      MOV 3EH, #0
      CLR LED
NEXT8: CJNE R3, #0, DISP
      CJNE R2, #1, DISP
      SETB BEEP      ;超时 1s 响一下
      SJMP DISP
HALF:  MOV A, R3
      CJNE A, 3BH, ZER
      MOV A, R2
      CJNE A, 3AH, ZER
      SETB BEEP      ;时间过半响一下
ZER:   CJNE R3, #0, DISP
      CJNE R2, #0, DISP
      SETB 20H.2     ;到了 00: 00 置位超时标志
      SETB BEEP      ;响一下
      CLR LED        ;亮一下

```



```

        SJMP DISP
FLASH:MOV A, 3DH      ;普通模式
        CJNE A, #3, NEXT9
        MOV 3DH, #0
        CLR LED
NEXT9: SJMP HALF
RELAY4: SJMP COUNT    ;上行接力
ACCEL:MOV A, R2
        JZ HERE20
        SETB 20H.3    ;渐快模式
        SJMP NEXT10
HERE20:CLR 20H.3
NEXT10: SJMP HALF

;显示部分
DISP: MOV P1, #0
        CLR F0
        MOV A, R0
        ANL A, #0FH
        SWAP A
        RL A
        RL A
        MOV R1, A
        MOV A, P0
        ANL A, #3FH
        ORL A, R1
        MOV P0, A      ;位选传送, 显示
        MOV A, @R0
        MOV DPTR, #0700H
        MOVC A, @A+DPTR;选择相应的数字
        MOV P1, A      ;段选传送

        LCALL D05MS
        SETB TR0

        MOV P1, #0
        INC R0
        MOV A, R0
        CJNE A, #34H, DISP
        MOV P1, #0
        MOV R0, #30H
        MOV R1, #6      ;再延时 3ms, 凑够一个周期 5ms
LOOP3: LCALL D05MS
        SETB TR0
        DJNZ R1, LOOP3
RET

BYE:    ;结束语
        MOV P1, #0
        MOV R2, #0
        MOV R3, #0
        MOV 30H, #0

```

```

MOV 31H, #0
MOV 32H, #0
MOV 33H, #0
SETB BEEP
CLR LED
CLR F0
MOV A, R0
ANL A, #0FH
SWAP A
RL A
RL A
MOV R1, A
MOV A, P0
ANL A, #3FH
ORL A, R1
MOV P0, A      ;位选传送, 显示
MOV A, R0
MOV DPTR, #0700H
MOVC A, @A+DPTR ;选择相应的数字
MOV P1, A      ;段选传送
LCALL D05MS

INC R0
MOV A, R0
CJNE A, #34H, BYE
MOV P1, #0
MOV R0, #30H
MOV R4, #6      ;再延时 3ms, 凑够一个周期 5ms
LOOP15:LCALL D05MS
DJNZ R4, LOOP15
JB F1, EOC4
CLR F0
EOC4:
RET

;D10MS 10ms 按键间隔
D10MS:PUSH ACC
MOV R7, #250
LOOP7:DJNZ R7, LOOP8
POP ACC
RET
LOOP8:MOV 36H, #60
LOOP6:DJNZ 36H, LOOP6

SJMP LOOP7

;D1MS 1ms 间隔
D1MS: PUSH ACC
MOV R7, #25

```

```

LOOP13:DJNZ R7, LOOP14
        POP ACC
        RET
LOOP14:MOV 36H, #60
LOOP12:DJNZ 36H, LOOP12

        SJMP LOOP13

```

```

;D05MS 0.5ms 显示间隔
D05MS:PUSH ACC
        MOV R7, #167
LOOP10:DJNZ R7, LOOP11
        POP ACC
        RET
LOOP11:MOV 36H, #3
LOOP9:DJNZ 36H, LOOP9

        SJMP LOOP10

```

```

;-----
//ISR

```

```

;TIMER0 1s 定时，需循环 25 次
TIMER0:
        CLR TR0
        MOV TH0, #0FH
        MOV TL0, #0B5H
        JNB 20H.3, NEXT13 ;渐快模式涉及下面的语句块
        DEC 40H
        MOV A, 40H
        CJNE A, #1, JUDGE
        MOV 40H, 3FH ;重置为零
        CPL 20H.5 ;亮灭切换
JUDGE:JB 20H.5, NEXT11
        SETB LED ;20H.5=0, 灯灭
        SJMP NEXT13
NEXT11:CLR LED ;20H.5=1, 灯亮
NEXT13:MOV A, 35H
        DEC A
        MOV 35H, A
        CJNE A, #0, HERE7 ;这里是小循环
        SETB TR0
        CPL 20H.4
        JNB 20H.2, NEXT12
        INC 3EH ;超时模式
        CLR 20H.3 ;清除 20H.3 的渐快模式的置位
        SJMP NEXT18
NEXT12:JNB 20H.3, NEXT17
        JNB 20H.4, NEXT14

```

```

        DEC 3FH
        SJMP NEXT14
NEXT17: INC 3DH          ;普通模式
NEXT18: SETB LED        ;关 LED
NEXT14: MOV P1, #0
        CLR BEEP
NEXT16: MOV 35H, #25
NEXT15: JB 20H.2, HERE17 ; 20H.2 置位时进入超时计数模式
        CJNE R2, #0, HERE6 ; 倒计时模式
        MOV R2, #59H
        MOV A, R3
        ADD A, #99H
        DA A
        MOV R3, A
        SJMP HERE5
HERE6:  MOV A, R2
        ADD A, #99H
        DA A
        MOV R2, A
HERE5:  MOV A, R2          ;对 30H-33H 进行更改
        ANL A, #0FH
        MOV 30H, A
        MOV A, R2
        ANL A, #0F0H
        SWAP A
        MOV 31H, A
        MOV A, R3
        ANL A, #0FH
        MOV 32H, A
        MOV A, R3
        ANL A, #0F0H
        SWAP A
        MOV 33H, A
        SJMP HERE7
HERE17: CJNE R2, #59H, HERE19
        MOV R2, #00H
        MOV A, R3
        ADD A, #01H
        DA A
        MOV R3, A
        SJMP HERE18
HERE19: MOV A, R2
        ADD A, #01H
        DA A
        MOV R2, A
HERE18: SJMP HERE5
HERE7:
        RETI

;TIMER1 消抖、长按授时
TIMER1:
        CLR TR1

```

```

        JB 20H.0, HERE1;已经按下, 20H.0=1, 就跳到 HERE1
        CJNE R5, #150, HERE3;这行指令是为了重装 R5 的值到 150
HERE4: JB KINT, HERE1;假按, 第一次时间短于 10ms, 不置 20H.0 为 1
        SETB 20H.0;20H.0=1
HERE1: MOV TH1, #0C4H
        MOV TL1, #030H
        JB KINT, EOCO
        DJNZ R5, EOCD;按下, 暂记为短按, 置 20H.0 为 1
        SJMP EOCC
HERE3: MOV R5, #150
        SJMP HERE4;重装完再跳回正常执行
EOCD: SETB 20H.0
        SJMP EOCD
EOCC: SETB 20H.1
        SJMP EOCD
EOCO: MOV C, 20H.0;统一输出
        JNC NEXT19
        SETB BEEP
        LCALL D05MS
        CLR BEEP
NEXT19: MOV F0, C
        MOV C, 20H.1
        JNC NEXT20
        CLR LED
        SETB BEEP
        LCALL D1MS
        SETB LED
        CLR BEEP
NEXT20: MOV F1, C
        CLR C;清理缓存
        CLR 20H.0
        CLR 20H.1
EOCR: ANL P0, #0FFH;通过读-改-写刷新一下
        SETB EX1
        RETI

```

;DETECT 相应 KINT 键

```

DETECT:
        CLR EX1;关掉外部中断 1 使能, 防止多次进入对显示的干扰
        SETB TR1;TIMER1 开始计时, 支持按键去抖和长短按识别等功能
        SETB KINT;置 1
        RETI

```

;-----定义段码表-----

```

        ORG 0700H
LUT1: DB 0FCH, 60H, 0DAH, 0F2H;0-3
        DB 66H, 0B6H, 0BEH, 0E0H;4-7
        DB 0FEH, 0F6H;8,9

```

```

        ORG 0730H
LUT2: DB 01H, 7AH, 2AH, 9EH        ;End.

; Peripheral specific initialization functions,
; Called from the Init_Device label
;-----
;-  Generated Initialization File  --
;-----
PCA_Init:
    anl  PCA0MD,    #0BFh
    mov  PCA0MD,    #000h
    ret

Timer_Init:
    mov  TMOD,      #011h
    mov  CKCON,     #001h
    mov  TL0,       #0B4h
    mov  TL1,       #0F7h
    mov  TH0,       #00Fh
    mov  TH1,       #0C2h
    mov  TMR2RLL,   #0F0h
    mov  TMR2L,     #0F0h
    mov  TMR3RLH,   #0DCh
    mov  TMR3H,     #0DCh
    ret

Port_IO_Init:
    ; P0.0 - Unassigned, Push-Pull, Digital
    ; P0.1 - Unassigned, Open-Drain, Digital
    ; P0.2 - Skipped, Push-Pull, Analog
    ; P0.3 - Skipped, Push-Pull, Analog
    ; P0.4 - Unassigned, Open-Drain, Digital
    ; P0.5 - Unassigned, Open-Drain, Digital
    ; P0.6 - Unassigned, Push-Pull, Digital
    ; P0.7 - Unassigned, Push-Pull, Digital

    ; P1.0 - Unassigned, Push-Pull, Digital
    ; P1.1 - Unassigned, Push-Pull, Digital
    ; P1.2 - Unassigned, Push-Pull, Digital
    ; P1.3 - Unassigned, Push-Pull, Digital
    ; P1.4 - Unassigned, Push-Pull, Digital
    ; P1.5 - Unassigned, Push-Pull, Digital
    ; P1.6 - Unassigned, Push-Pull, Digital
    ; P1.7 - Unassigned, Push-Pull, Digital
    ; P2.0 - Unassigned, Push-Pull, Digital
    ; P2.1 - Unassigned, Push-Pull, Digital
    ; P2.2 - Unassigned, Push-Pull, Digital
    ; P2.3 - Unassigned, Push-Pull, Digital

    mov  P0MDIN,    #0F3h
    mov  P0MDOUT,   #0CDh
    mov  P1MDOUT,   #0FFh
    mov  P2MDOUT,   #00Fh
    mov  P3MDOUT,   #002h
    mov  P0SKIP,    #00Ch

```

```

    mov  XBR1,      #040h
    ret

Oscillator_Init:
    mov  OSCICN,    #081h
    ret

Interrupts_Init:
    mov  IP,        #02Eh
    mov  IT01CF,    #014h
    mov  IE,        #0AEh
    ret

; Initialization function for device,
; Call Init_Device from your main program
Init_Device:
    lcall PCA_Init
    lcall Timer_Init
    lcall Port_IO_Init
    lcall Oscillator_Init
    lcall Interrupts_Init
    ret

end

```

六、程序测试方法与结果

本程序采用 KEIL 软件自带的 Simulation 功能仿真与硬件运行结果相结合。（注：为了测试方便，下面图片中的函数所在位置不代表实际代码中出现的位置）

需要测试的内容如下：

- （1）各个子程序模块、函数、中断的延时情况
- （2）数码管与 D9 灯的实现情况
- （3）KINT 键长短按识别情况
- （4）矩阵式数字键盘识别情况
- （5）蜂鸣器的运行情况

测试的结果如下：

(1) 各个子程序模块、函数、中断的延时情况

D05MS 延时函数设想的理论值应为 0.5ms，Simulation 模式下，如图 10 所示，实际运行情况约为 0.545ms。考虑到该函数为单个数码管显示的时间，这样的数值是可以接受的。

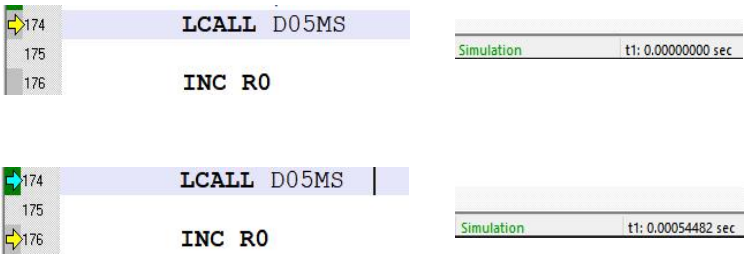


图 10

D1MS 延时函数设想的理论值应为 1ms，Simulation 模式下，如图 11 所示，实际运行情况约为 0.975ms。考虑到该函数为按键按下时蜂鸣器反馈鸣响延迟的时间，数值是可以接受的。

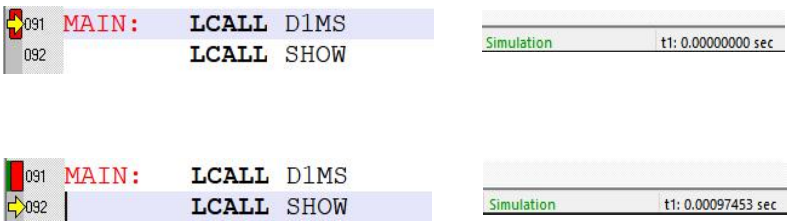


图 11

D10MS 延时函数设想的理论值应为 10ms，Simulation 模式下，如图 12 所示，实际运行情况约为 10.08ms。考虑到该函数为矩阵式数字键盘按键按下后消抖的时间，这样的数值是可以接受的。

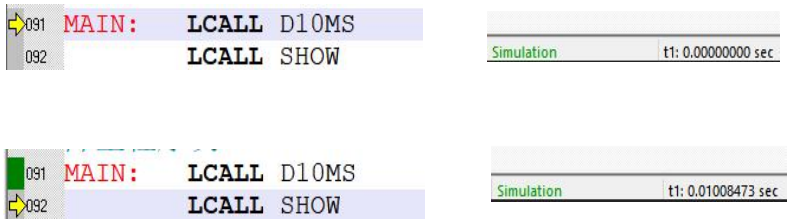


图 12

TIMER0 的授时设想的是 25 个循环为 1s，则单次中断的时间为 40ms，如图 13 所示，实际运行情况约 40.17ms，应当指出的是，这里的仿真结果并不代表实际情况。在随后的实际测试中，发现当 0FB4H 时，定时的精度最高，故最终设定为 0FB4H。

实测的误差为一小时快 8 秒。

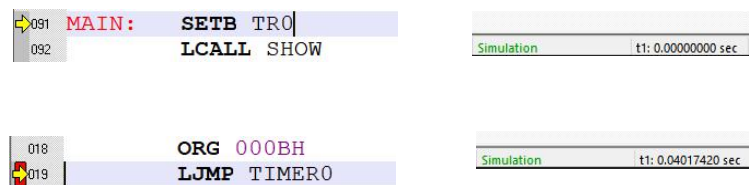


图 13

TIMER1 的授时设想的是 10ms，如图 14 所示，实际运行情况约 10.20ms。考虑到该定时器用于 KINT 按键按下后消抖的时间，这样的数值是可以接受的。

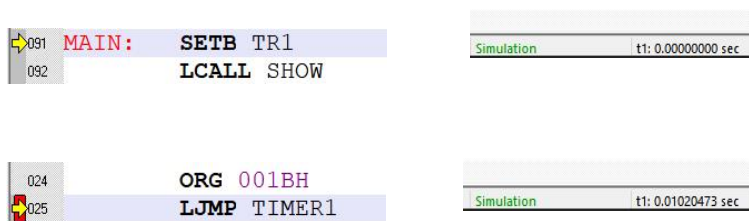


图 14

(2) 数码管与 D9 灯的实现情况

如图 15 所示，初始状态下，四个数码管均为 0。

如图 16 所示，输入状态下，待输入位呈闪烁状。

如图 17 所示，正在进行输入。

如图 18 所示，倒计时模式下，数码管与 D9 均在正常工作。

如图 19 所示，超时的结束，数码管显示“End.”，并且 D9 常亮。

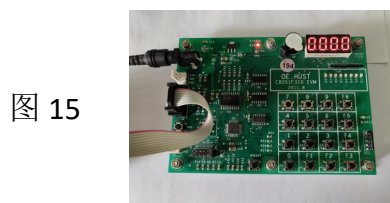


图 15

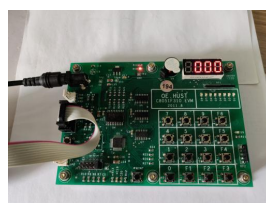


图 16

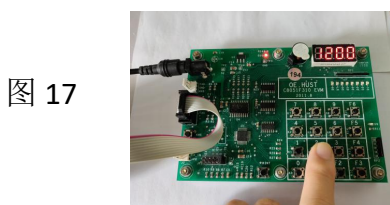


图 17

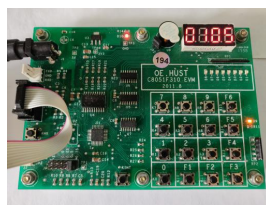


图 18

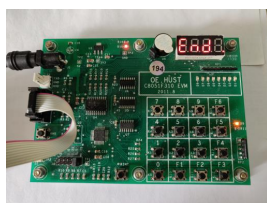
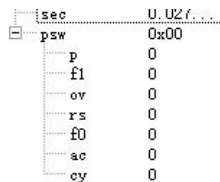


图 19

(3) KINT 键长短按识别情况

Simulation 模式下，手动勾选 P0.1，测试结果如下图 20 所示。

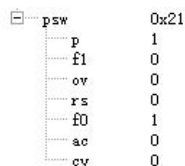
当 P0.1 被“按下”的时间小于 10ms 时，F0=F1=0



sec	0.0027...
psw	0x00
p	0
f1	0
ov	0
rs	0
f0	0
ac	0
cy	0

图 20

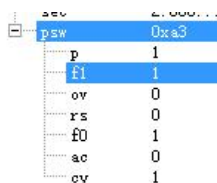
当 P0.1 被“按下”的时间大于 10ms 且小于 2s 时，F0=1，F1=0，测试结果如下图 21 所示。



psw	0x21
p	1
f1	0
ov	0
rs	0
f0	1
ac	0
cy	0

图 21

当 P0.1 被“按下”的时间大于 2s，F0=F1=1，测试结果如下图 22 所示。



psw	0xa3
p	1
f1	1
ov	0
rs	0
f0	1
ac	0
cy	1

图 22

(4) 矩阵式数字键盘识别情况

经过实际测试，矩阵式键盘能够在的保证输入准确、防止串键、不识别 F1-F6、秒钟输入限制的同时，不干扰数码管显示。

(5) 蜂鸣器的运行情况

经过实际测试，蜂鸣器在 KINT 键按下均会清脆的响一声，并且在应该鸣响的时间点均能正常运行。

七、心得与体会

抱着试试看的心态，我报名了微机实验专题设计，很幸运能成功入围。从5月12日收到板子开工，到如今已步入六月，半个多月的时间里，我专心做好这一件事，事后回想，无论是从技术上还是心理上，都感觉收获满满。

朝来夕往，筚路蓝缕。

在技术上，我很开心能够对书本上所学的知识加以综合运用，虽然这不是什么特别复杂的大项目，但麻雀虽小，五脏俱全，理论课程学习到的大部分知识都用上了。这一点我觉得很重要，因为只有通过一个个项目，才能将书本上的理论体系变为真正实际能够使用的功能，让它变得不再“虚无缥缈”，而是脚踏实地。这样也会使得课本上理想的丰满迎接现实的骨感，但是却更是富有挑战与趣味的。PBL模式是区别于其他课程的重要特色，我也衷心希望这种模式能够在更多的课程中得到推广。

我有两个肤浅的思考：一是局部与整体的关系。整个程序犹如整个宇宙，而构成整个程序的指令犹如构成整个世界的元素。元素并不纷繁，但是通过无穷种可能的结合与排列，呈现在我们面前的是一个超乎想象的绚烂世界。因此，就像发明一种新功能的材料，我们应该从事物的功能、性质来反推它的结构，思考运用什么样的元素，而这个创造的过程正如同我们的编程。我们应当具有整体观，想清楚各个模块之间的联系，这样会有助于程序的编写，而避免掉入琐碎的陷阱中。二是离散与连续的关系。直觉上，人类普遍认为世界是连续的，就如同我们看到数码管均匀的亮着。而电子技术从某种意义上很大程度上利用了人们的这种不总是正确的感觉，将离散变为感官上的连续。我们可以让数码管在大部分时间内都是关闭的状态，但却看上去一直在亮。我有一个关于单片机高手的认识，即运用单片机的最高境界就是充分用好每段离散的时间，使得更多的功能能够在感光上连续的实现。这需要华罗庚提到的“接待客人烧水沏茶”的统筹规划思想。

空山无人，流水花开。

在心理上的成长，我认为是更为重要的。调一段代码遇到困难，可能一个上午都卡在了一个字母的错用，难受吗？当然，但是那又能怎样？还是得沉住气接着弄。和老师同学之间的交流万分重要，甚至有时候能够改变你的设计思路。在最后这周里，由于没有实验板，很多同学找我来跑他们的程序，我都会耐心的和他们视频在线调试，为他们的成功而高兴，也会为他们的bug提出自己的想法，这都是在相互促进相互鼓励。我收获的其实远远不止这些，它们都会潜移默化地在未来生活中影响我。

最后，我要感谢曹老师给我这次机会参与到这样一个意义非凡的项目，感谢父母在家听我的板子嗡嗡的响个不停，感谢二位老师整理出来的系统详尽的资料，感谢吴老师验收时的辛苦，感谢助教的操劳，感谢交流群里的各种经验分享，感谢两次答疑会给予了我信心，感谢自己的付出。

二〇二〇年六月二日于北京家中

设计人签名： 马笑天