单片机原理与接口技术实验

学　　院	自动化学院
专　　业	电子信息科学与技术
年级班别	2017 级 1 班
学　　号	3117001295
学生姓名	方智威
指导教师	王倩雪

2019 年 9 月 28 日

实验一

实验项目名称：PIC 单片机汇编语言编程及硬件电路调试

一、报告内容

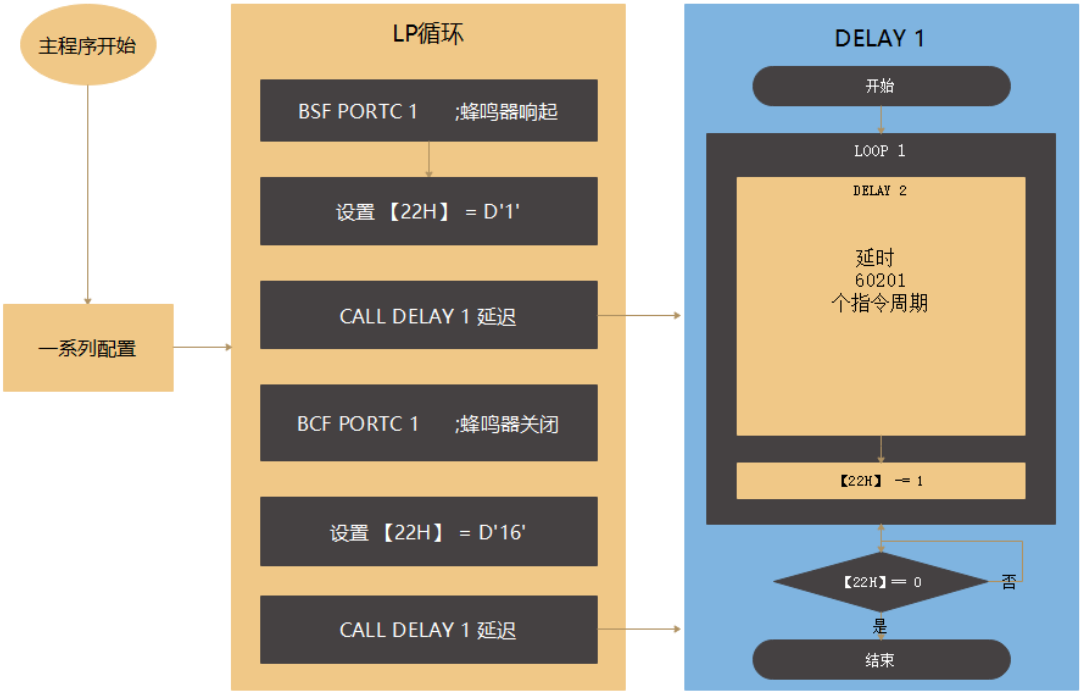
1、对例程 1 的每一句指令进行注释，描述该句指令的作用以及如何影响寄存器结果。

```
      INCLUDE "P16F877.INC"  ;PIC16F877A包含的头文件
      ORG  0000H             ;伪指令，即下面的程序从程序存储器的地址0000H开始存放

      NOP               ; 空操作

      BANKSEL TRISC     ; 选择TRISC所在的体域
      CLRF TRISC        ; TRISC=00000000，即将RC全设为输出
      BANKSEL PORTC     ; 选择PORTC所在体域
      CLRF PORTC        ; PORTC=00000000，即将输出全设为低电平
      BANKSEL PORTC     ; 选择PORTC所在体域
      BSF PORTC,1       ; RC1=1，蜂鸣器响起
LP    GOTO LP           ; 循环
      END
```

2、标出例程 2 每一句指令的指令周期，伪指令除外。

```
      INCLUDE "P16F877.INC"  ;PIC16F877A包含的头文件
      ORG  0000H             ;伪指令，即下面的程序从程序存储器的地址0000H开始存放

      NOP                          ;指令周期1
      BANKSEL TRISC                ;指令周期1
      CLRF TRISC                   ;指令周期1
      BANKSEL PORTC                ;指令周期1
      CLRF PORTC                   ;指令周期1
      BANKSEL PORTC                ;指令周期1

LP    BSF PORTC,1                  ;指令周期1
      BANKSEL 22H                  ;指令周期1
      MOVLW D'1'                   ;指令周期1
      MOVWF 22H                    ;指令周期1
      CALL DELAY1                  ;指令周期2

      BCF PORTC,1                  ;指令周期1
      BANKSEL 22H                  ;指令周期1

      MOVLW D'16'    ;正确设置参数，实现蜂鸣器以1秒为周期鸣叫，  指令周期1

      MOVWF 22H                    ;指令周期1
      CALL DELAY1                  ;指令周期2
      GOTO LP                      ;指令周期2

DELAY1  NOP                        ;指令周期1
LOOP1   CALL DELAY2                ;指令周期2
        DECFSZ 22H                 ;指令周期1（2）
        GOTO LOOP1                 ;指令周期2
        RETURN

DELAY2  MOVLW     D'100'           ;指令周期1
        MOVWF     20H              ;指令周期1
LOOP2   MOVLW     D'200'           ;指令周期1
        MOVWF     21H              ;指令周期1
LOOP3   DECFSZ    21H              ;指令周期1（2）
        GOTO      LOOP3            ;指令周期2
        DECFSZ    20H              ;指令周期1（2）
        GOTO      LOOP2            ;指令周期2
        RETURN
END
```

3、写出例程 2 的程序流程图。



4、写出方框内参数的计算过程。

DELAY2 = 1+1【 (【(1+2)x200 – 1】 + 1 + 2) x 100 – 1】 = 60201

DELAY1 = (【DELAY2】 +1+2) * 【22H】 = 1000000

解得 【22H】= 16.610192

二、思考题

待补

三、实验中遇到的问题和解决途径

问题一：计算指令周期，在多个循环时容易出错混乱

解决：仔细看代码，逐行剖析

实验二

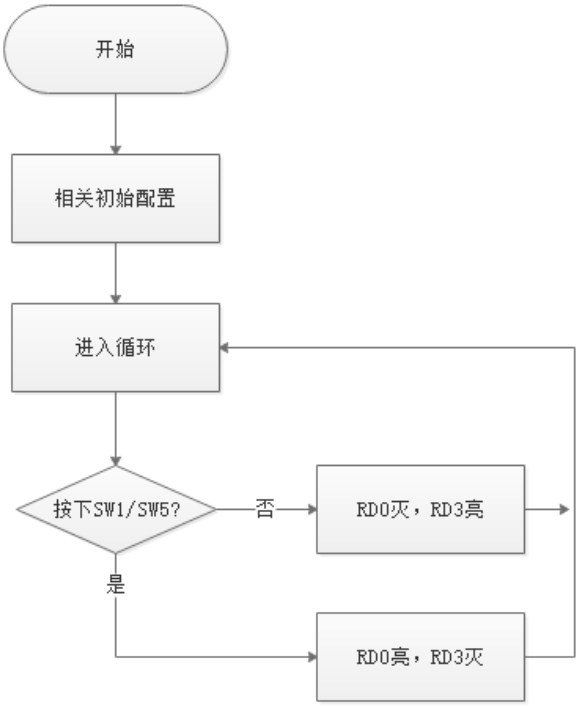
实验项目名称：PICC 编译环境及 PIC 单片机的通用 IO 口编程

一、报告要求

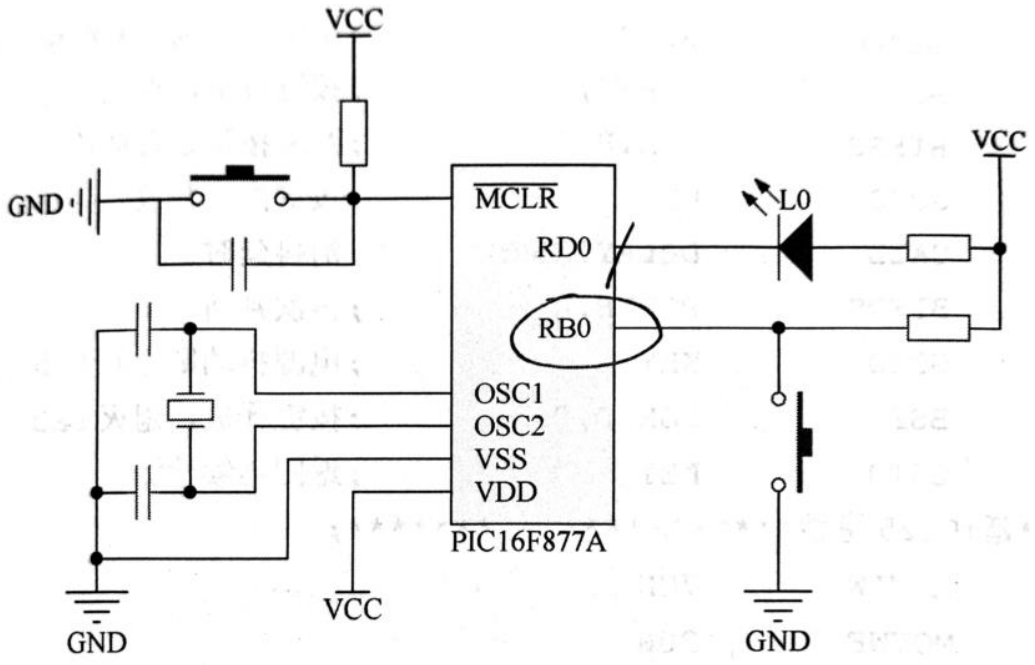
1. 对例程 1 的每一句程序进行注释。描述如何改变寄存器及其对 IO 的影响。

```
#include <pic.h>     //加载头文件
 __CONFIG(XT&WDTDIS&LVPDIS);
 //设置配置字
//名称: int const d=0B000000001;

 //设置配置字
//名称: 主函数
void main(void)
{
     TRISE=TRISE&(~(0x01<<4));        // TRISE[4]=0, 即PSPMODE=0, D端口工作模式为串行
     ADCON1=(ADCON1&0xf0 )|0x07;      // ADCON1[3:0] = 0111, 将RA0~RA5, RE0~RE2 改为数字IO
     TRISA=TRISA&(~(0X01<<1));        // TRISA[1] = 0, RA1输出
     TRISA=TRISA&(~(0X02<<1));        // TRISA[2] = 0, RA2输出
     TRISD=0X00;                      // TRISD[7:0] = 00000000, RD全输出
     TRISB=TRISB&(0X01<<2);           // TRISB[7:0] = 00000?00, 除RB2, RB其他全输出
     PORTD=0B00000000;                // 灭掉所有LED
     PORTA=0B00000110;                // 选择了COL1, COL@, 即灯与按键的第一排和第二排
     while(1){
         if(RB2==0) {                // 如果SW1或SW5值为0, RD0亮, RD3灭
             RD0=1;
             RD3=0;
                    }
         else {                      // 如果SW1或SW5值为1, RD3亮, RD1灭
                RD0=0;
                RD3=1;
                     }
     }
}
```

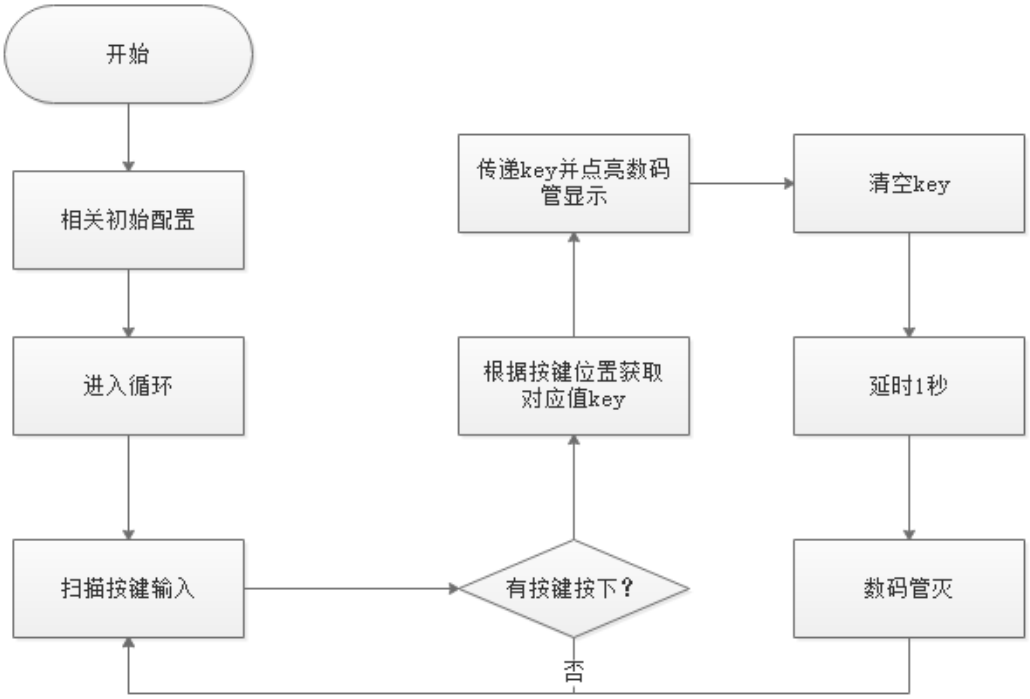
2. 画出例程 1 的程序流程图。



3. 画出例程 2 所涉及的硬件电路图,要求包括最小系统,IO 口,LED,键盘电路。



4. 画出例程 2 的程序流程图，包括主程序及子程序。



5.讨论红色表及部分的 delaytime 的作用以及其参数的影响。

因为电路设计的原因，多个数码管共用 PORTD 数据口，所以正常情况下只能多个数码管显示同一个值。为能充分利用数码管，我们可以设置极短延时让不同数据在不同数码管间快速切换，达到一种像是显示着不同数值的现象。就好像该题的 delaytime 延时，若 delaytime 过长，则会导致数码管与数据的切换过慢，导致两个数码管来回闪烁而不是同时显示，达不到我们想要的效果。因此我们需要这个 delaytime 尽可能的短但不可以没有

二、思考题

```
#include<pic.h>
__CONFIG(XT&WDTDIS&LVPDIS);
#define uchar unsigned char
#define uint unsigned int

void delay(uint x)
{
    int i, j;
    for(i=x;i>0;i--)
        for(j=110;j>0;j--);
}
```

```
void main(void)
{
    int i;
    TRISE = TRISE&(~(0x01<<4));
    ADCON1=(ADCON1&0xf0)|0x07;
    TRISA=TRISA&(~(0x01<<1));
    TRISA=TRISA&(~(0x01<<2));
    TRISA=TRISA&(~(0x01<<3));
    TRISA=TRISA&(~(0x01<<4));
    TRISA=TRISA&(~(0x01<<5));
    TRISD=0x00;
    PORTA = 0B00000010;
    PORTD = 0B00000001;
    delay(100);
    while(1)
    {
        for(i=0;i<7;i++)
        {
            PORTD = PORTD << 1;
            delay(100);
        }

        for(i=0;i<3;i++)
        {
            if(i<2)
                PORTA = PORTA << 1;
            else
                PORTA = PORTA << 2;
            delay(100);
        }

        for(i=0;i<7;i++)
        {
            PORTD = PORTD >> 1;
            delay(100);
        }

        for(i=0;i<3;i++)
        {
            if(!i)
                PORTA = PORTA >> 2;
            else
                PORTA = PORTA >> 1;
            delay(100);
        }
    }
}
```

三、实验中遇到的问题和解决途径

问题一：计算指令周期，在多个循环时容易出错混乱

解决：仔细看代码，逐行剖析

实验三

实验项目名称：PIC 单片机中断实验

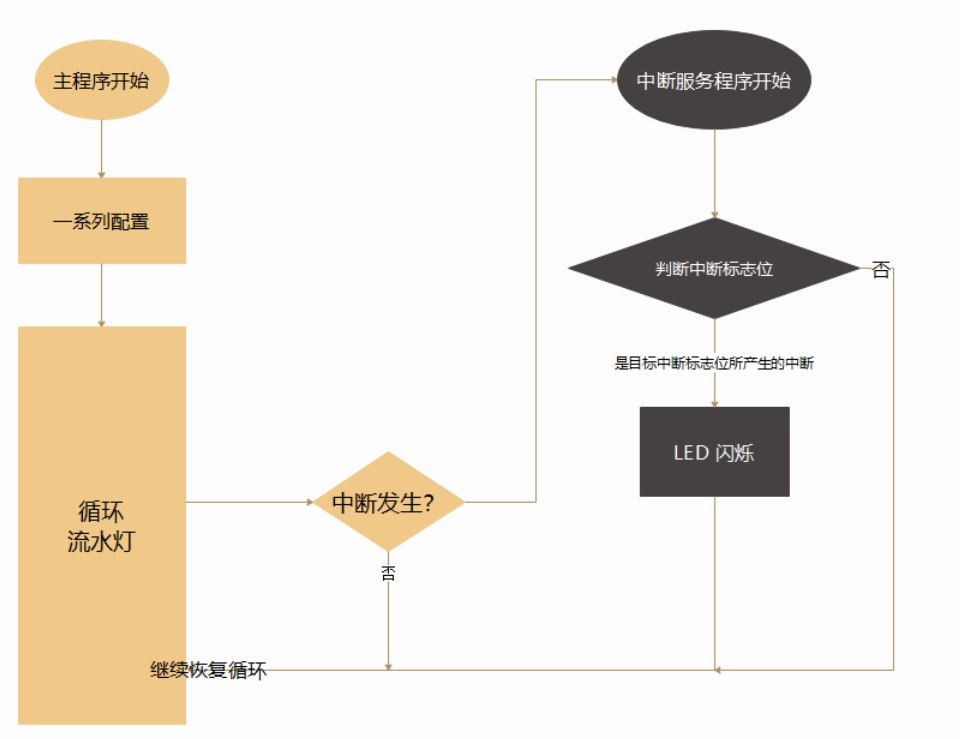
一、报告要求

1. 对例程 1 的每一句程序进行注释。描述如何改变寄存器及其对 IO 的影响。

```
#include<pic.h>   //加载头文件;

#define uint8 unsigned char
void Delay(unsigned int delay);     //声明延时函数;
void Delay1(unsigned int delay);    //声明延时函数;

void main()
{
   ADCON1=(ADCON1&0xf0)|0x07;        //设置ADCON1低四位为0111,
                                     //将A口或E口作为普通I/O口使用;
   TRISB0=1;                         //初始化I/O口，将RB0设置为输入;
   TRISD=0x00;                       //将RD全部设置为输出;
   TRISA=TRISA&(~(0x01<<1));         //将RA1设置为输出;
   INTEDG=1;    //当INTEDG=1时，RB0/INT引脚上升沿触发；当INTEDG=0时，RB0/INT引脚下降沿触发;
   INTE=1;      //使能外部触发中断，允许外部触发中断请求;
   INTF=0;      //清除标志位，未发生外部触发中断;
   GIE=1;       //使能总中断，允许所有中断源的中断请求;
   RA1=1;       //选择第一列LED灯，即选通数管的第一位;
   while(1){
       unsigned char i=0;  //定义无符号变量i，i=0;
       for(i=0; i<8; i++)    //for循环语句，i<8时自加1，否则跳出循环
        {
           PORTD=(0x01<<i);   //将PORTD的第i位设置为高电平
           Delay(3000);       //设置延时参数为3000
        }
   }
}

void interrupt PIC_Int(void)     // 开始中断服务程序
{
 uint8 i;                       //定义变量i
 if (INTF)                      //RB0电平变化触发中断
 {
    PORTD=0x00;              //灭掉所有LED
    for(i=0; i<10; i++)          // for循环语句，i<10时自加1，否则跳出循环
        {
           PORTD=~PORTD;     //点亮所有LED
           Delay1(150);       //调用Delay1延时函数，设置延时参数为150
        }
        INTF=0;               // 清除标志位
 }
}
```

```
void Delay (unsigned int delay)     //定义延时函数Delay
{
 unsigned int i;                 //定义无符号整型变量i
  for (;delay>0; delay--)        // for循环语句，delay>0时自减1，否则跳出循环
    for (i=0; i<45; i++);          // for循环语句，i<10时自加1，否则跳出循环

}
 void Delay1 (unsigned int delay)     //定义延时函数Delay1
{
 unsigned int i;     //定义无符号整型变量i
  for (;delay>0; delay--)    // for循环语句，delay>0时自减1，否则跳出循环
    for (i=0; i<45; i++);     // for循环语句，i<10时自加1，否则跳出循环
}
```

2. 画出例程 1 的程序流程图。



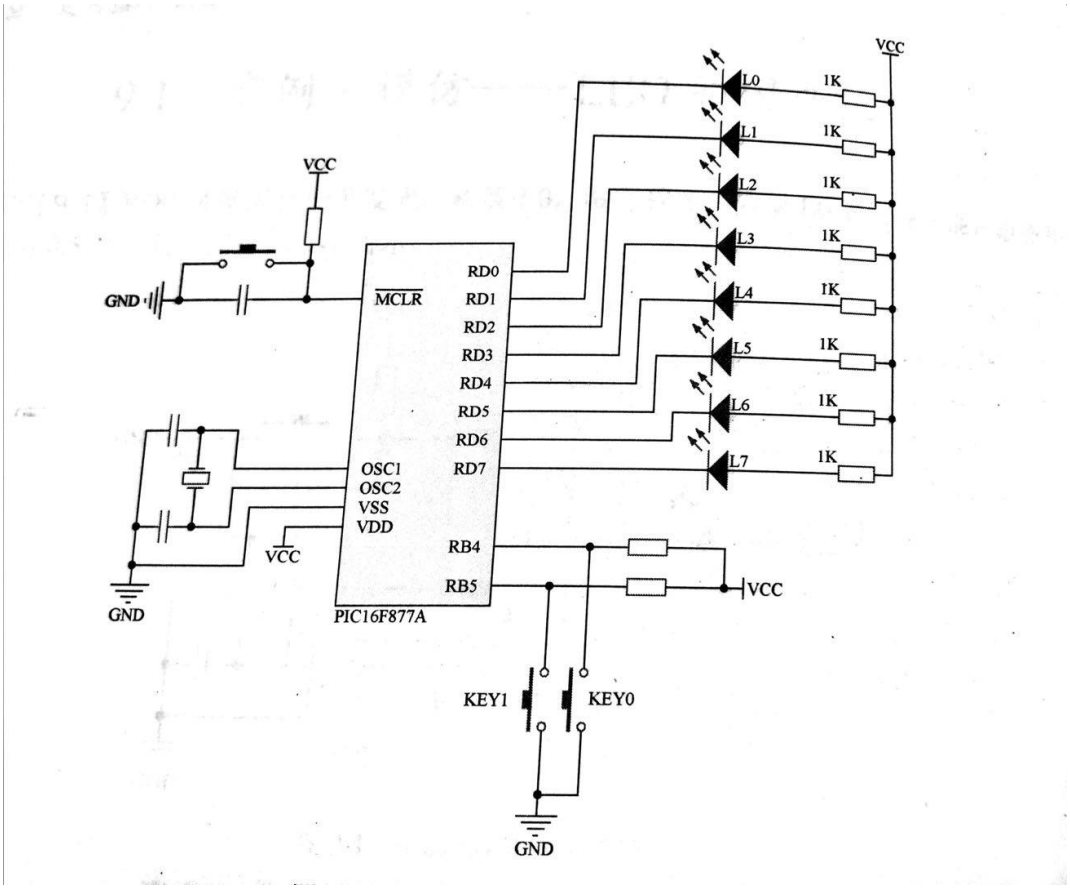
3. 讨论中断方式与查询方式的区别。

通过查询方式进入中断服务程序，则需要在程序中不断去扫描标志位。而通过中断方式来执行中断服务程序，能避免一些持续扫描而造成不必要的资源浪费，而且在任何时候只要中断标志位更改，能及时跳转的中断服务程序，不像查询方式有相对延时，需要等到扫描步骤执行才能去判断中断是否触发。

4. 讨论 INTEDG 的作用和对 INT 中断触发的效果。

INTEDG 位是用来配置 RB0/INT 引脚输入中断信号是上升沿触发还是下降沿触发，若 INTEDG = 1，则 RB0 的信号为上升沿时触发中断；若 INTEDG = 0，则 RB0 的信号为下降沿时触发中断。

5. 画出任务 2 所涉及到的硬件电路图，要求包括最小系统，IO 口，LED，相关键盘电路。



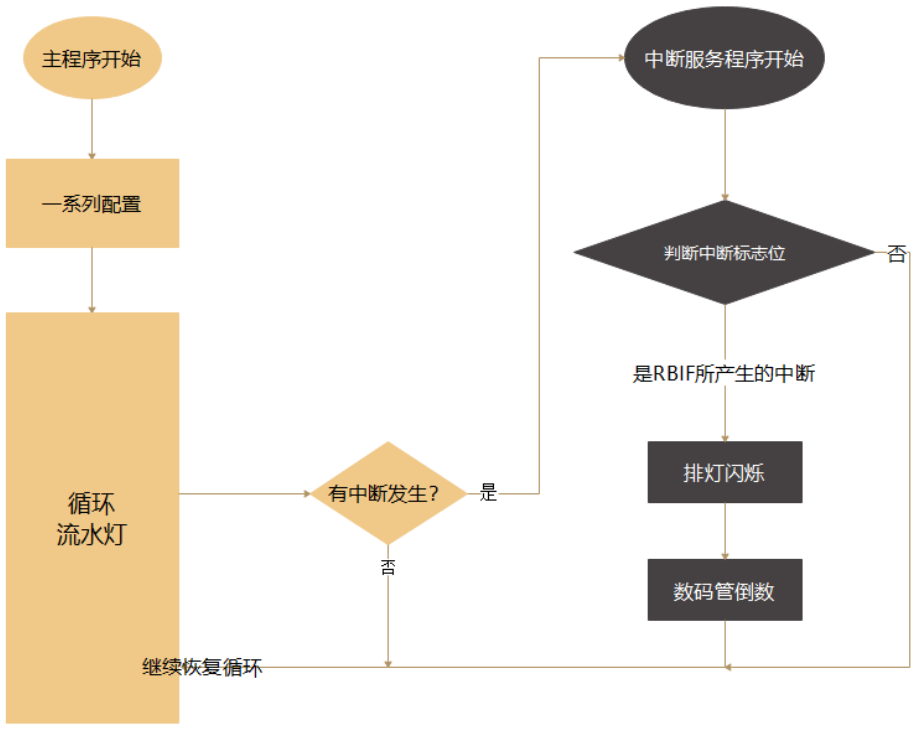
6. 编写出任务 2 的中断服务程序以及程序流程图。

```
#include<pic.h>
#define uint8 unsigned char

void Delay(unsigned int delay);   //声明延时函数;
void Delay1(unsigned int delay);  //声明延时函数;
const unsigned char LED[10]=
{
    0x3f,     //0
    0x06,     //1
    0x5b,     //2
    0x4f,     //3
    0x66,     //4
    0x6d,     //5
    0x7d,     //6
    0x07,     //7
    0x7f,     //8
    0x6f,     //9
};
```

```
void main() {
    ADCON1 = (ADCON1&0b11110000)|0b00000111;          // 设置ADCON1低四位为0111，将A口或E口作为
全数电普通I/O口使用;
    TRISB = (TRISB&0b11001111)|0b00110000;            // RB4(SW2),RB5(SW3) 输入
    TRISD = 0x00;                                     // RD0~RD7(LED)，全输出
    TRISA = TRISA&0b11011101;                         // RA1(COL1,排灯1，倒数第一个
LED),RA5(COL4,排灯4，倒数第四个LED)，设置为输出;


    RBIE = 1;         // SW2,SW3 变化中断使能
    RBIF = 0;         // SW2,SW3 变化中断标志清除
    GIE = 1;          // 使能总中断

    RA1 = 1;          // RA1(COL1,排灯)，使能
    RA5 = 0;          // RA5(COL4,倒数第四个LED)，关;

    while(1){         //执行while语句
        unsigned int i;
        for(i=0; i<8; i++) {
            PORTD=(0x01<<i);    //将PORTD的第i位设置为高电平
            Delay(1000);        //设置延时参数为3000
         }
    }
}
```

```
void interrupt PIC_Int(void) {
    unsigned int i;

    // SW2 or SW3 触发中断
    if(RBIF) {
        PORTD=0x00;              //灭掉所有LED


        for(i=0; i<10; i++)  {
           PORTD=~PORTD;     //点亮所有LED
           Delay1(150);        //调用Delay1延时函数，设置延时参数为150
        }
```

```
        RA1 = 0;                    // RA1(COL1,排灯), 关
        RA5 = 1;                    // RA5(COL4,倒数第四个LED), 使能;

        for(i=0; i<6; i++) {
            PORTD = LED[5-i];
            Delay1(300);
        }

        PORTD=0x00;                 //灭掉所有LED

        RA1 = 1;                    // RA1(COL1,排灯), 使能
        RA5 = 0;                    // RA5(COL4,倒数第四个LED), 关;

    }
    RBIF = 0;
}


void Delay (unsigned int delay) {
    unsigned int i;                 //定义无符号整型变量i
    for (;delay>0; delay--)        // for循环语句, delay>0时自减1, 否则跳出循环
        for (i=0; i<45; i++);          // for循环语句, i<10时自加1, 否则跳出循环

}

void Delay1 (unsigned int delay) {
    unsigned int i;     //定义无符号整型变量i
    for (;delay>0; delay--)   // for循环语句, delay>0时自减1, 否则跳出循环
        for (i=0; i<45; i++);     // for循环语句, i<10时自加1, 否则跳出循环
}
```



二、思考题

编写中断服务程序，实现当 SW3 按下时，触发 RB 电平变化中断，LED 数码管第四位倒数 5,4,3,2,1,0，当 SW0 按下时，触发 INT 中断，第一列 LED 闪烁 5 次。

```
#include<pic.h>
#define uint8 unsigned char

void Delay(unsigned int delay);   //声明延时函数;
void Delay1(unsigned int delay);  //声明延时函数;
const unsigned char LED[10]=
{
    0x3f,     //0
    0x06,     //1
    0x5b,     //2
    0x4f,     //3
    0x66,     //4
    0x6d,     //5
    0x7d,     //6
    0x07,     //7
    0x7f,     //8
    0x6f,     //9
};

void main() {
    ADCON1 = (ADCON1&0b11110000)|0b00000111;        // 设置ADCON1低四位为0111，将A口或E口作为
全数电普通I/O口使用;
    TRISB = (TRISB&0b11011110)|0b00100001;          // RB0(SW0)，RB5(SW3) 输入
    TRISD = 0x00;                                   // RD0~RD7(LED)，全输出
    TRISA = TRISA&0b11011101;                       // RA1(COL1,排灯1，倒数第一个
LED),RA5(COL4,排灯4，倒数第四个LED)，设置为输出;


    INTEDG = 1;     // SW0 中断上升沿触发
    RBIE = 1;       // SW3 变化中断使能
    RBIF = 0;       // SW3 变化中断标志清除
    INTE = 1;       // SW0 变化中断使能
    INTF = 0;       // SW0 变化中断标志清除;
    GIE = 1;        // 使能总中断
    RA1 = 1;        // RA1(COL1,排灯)，使能
    RA5 = 0;        // RA5(COL4,倒数第四个LED)，关;

    while(1){       //执行while语句
        unsigned int i;
        for(i=0; i<8; i++)    //for循环语句，i<8时自加1，否则跳出循环
         {
            PORTD=(0x01<<i);   //将PORTD的第i位设置为高电平
            Delay(1000);       //设置延时参数为3000
         }
    }
}
```

```
void interrupt PIC_Int(void) {
    unsigned int i;

    // SW0触发中断
    if(INTF) {
        PORTD=0x00;             //灭掉所有LED

        for(i=0; i<10; i++)          // for循环语句，i<10时自加1，否则跳出循环
            {
               PORTD=~PORTD;    //点亮所有LED
               Delay1(150);       //调用Delay1延时函数，设置延时参数为150
            }

    }
    // SW3触发中断
    else if(RBIF && !RB5) {
        PORTD=0x00;             //灭掉所有LED
        RA1 = 0;                // RA1(COL1,排灯)，关
        RA5 = 1;                // RA5(COL4,倒数第四个LED)，使能;

        for(i=0; i<6; i++) {
            PORTD = LED[5-i];
            Delay1(300);
        }
        PORTD=0x00;             //灭掉所有LED
        RA1 = 1;                // RA1(COL1,排灯)，使能
        RA5 = 0;                // RA5(COL4,倒数第四个LED)，关;


    }
    INTF = 0;
    RBIF = 0;
}
```

```
void Delay (unsigned int delay)     //定义延时函数Delay
{
    unsigned int i;                //定义无符号整型变量i
        for (;delay>0; delay--)       // for循环语句，delay>0时自减1，否则跳出循环
        for (i=0; i<45; i++);         // for循环语句，i<10时自加1，否则跳出循环

}
 void Delay1 (unsigned int delay)     //定义延时函数Delay1
{
    unsigned int i;     //定义无符号整型变量i
        for (;delay>0; delay--)   // for循环语句，delay>0时自减1，否则跳出循环
        for (i=0; i<45; i++);    // for循环语句，i<10时自加1，否则跳出循环
}
```

三、实验中遇到的问题和解决途径

问题一：思考题中两种不同触发中断下的中断清除会有问题。

解决：把中断标志位清除放在判断语句外。

实验四

实验项目名称：PIC 单片机定时器实验

一、报告要求

1. 对例程 1、2、3 的每一句程序进行注释。描述如何改变寄存器及其对 IO 的影响。

- 例一

```
#include <pic.h>
// __CONFIG(XT&WDTDIS&LVPDIS);


#define CONST_T 0xD9;

void delay_10ms (unsigned int n);

void main() {
    TRISC1 = 0;             // RC1输出

    while(1) {

        RC1 = 0;            // 关蜂鸣器
        delay_10ms(25);     // 900ms
        RC1 = 1;            // 开蜂鸣器
        delay_10ms(25);     // 100ms
        RC1 = 0;            // 关蜂鸣器
        delay_10ms(25);     // 900ms
        RC1 = 1;            // 开蜂鸣器
        delay_10ms(25);     // 100ms

    }

}

void delay_10ms (unsigned int n) {
    unsigned int i;
    OPTION = 0x07;      // 分频比1:256
    T0IF = 0;           // 清除标志位

    for (i=0; i<n; i++) {
        TMR0 = CONST_T;     // 重新赋初值
        while(!T0IF);       // 溢出
        T0IF = 0;           // 清除标志位
    }
}
```

- 例二

```
#include <pic.h>
// __CONFIG(XT&WDTDIS&LVPDIS);


#define CONST_T 0xD9;
#define N_10 0x19;
int n;                  // 定义全局变量 n，用于标志TRM0中断发生次数
void main() {
    TRISC1 = 0;         // RC1输出
    OPTION = 0x07;      // 分频比 1:256
    T0IE = 1;           // TMR0中断使能
    T0IF = 0;           // TMR0中断标志位清除
    GIE = 1;            // 总中断使能

    TMR0 = CONST_T;     // TMR0寄存器赋初值
    RC1 = 0;            // RC1输出低电平

    n = N_10;           // n = 0x19
    while(1);           // 循环
}

void interrupt PIC_Int(void) {

    if(n == 0) {
        RC1 = !RC1;     // RC1取反
        n = N_10;       // n重新赋值
    }
    else {
        n = n - 1;      // 中断次数+1，n-1
    }

    TMR0 = CONST_T;     // TMR0寄存器赋初值
    T0IF = 0;           // TMR0中断标志位清除
}
```

- 例三

```
#include<pic.h>
void main(){
    ADCON1=(ADCON1&0xf0)|0x07;              // ADCON1低四位为0111，RA为数字IO
    TRISD=0x00;                             // D端口设置为输出
    TRISA=TRISA&(~(0x01<<1));               // RA1设置为输出
    TRISA4=1;                               // RA4设置为输入，计数按键
    RA1=1;                                  // RA1设为高电平，选中按键，通过跳线把按键信号传给RA4
    OPTION_REG=0B00101000;                  // 工作于计数器模式，分频取2
    T0IF=0;                                 // 清除标志位
    TMR0=0x00;                              //TMR0初始值为0

    while(1){
        PORTD=TMR0;                         // 数码管显示计数了几次
    }
}
```

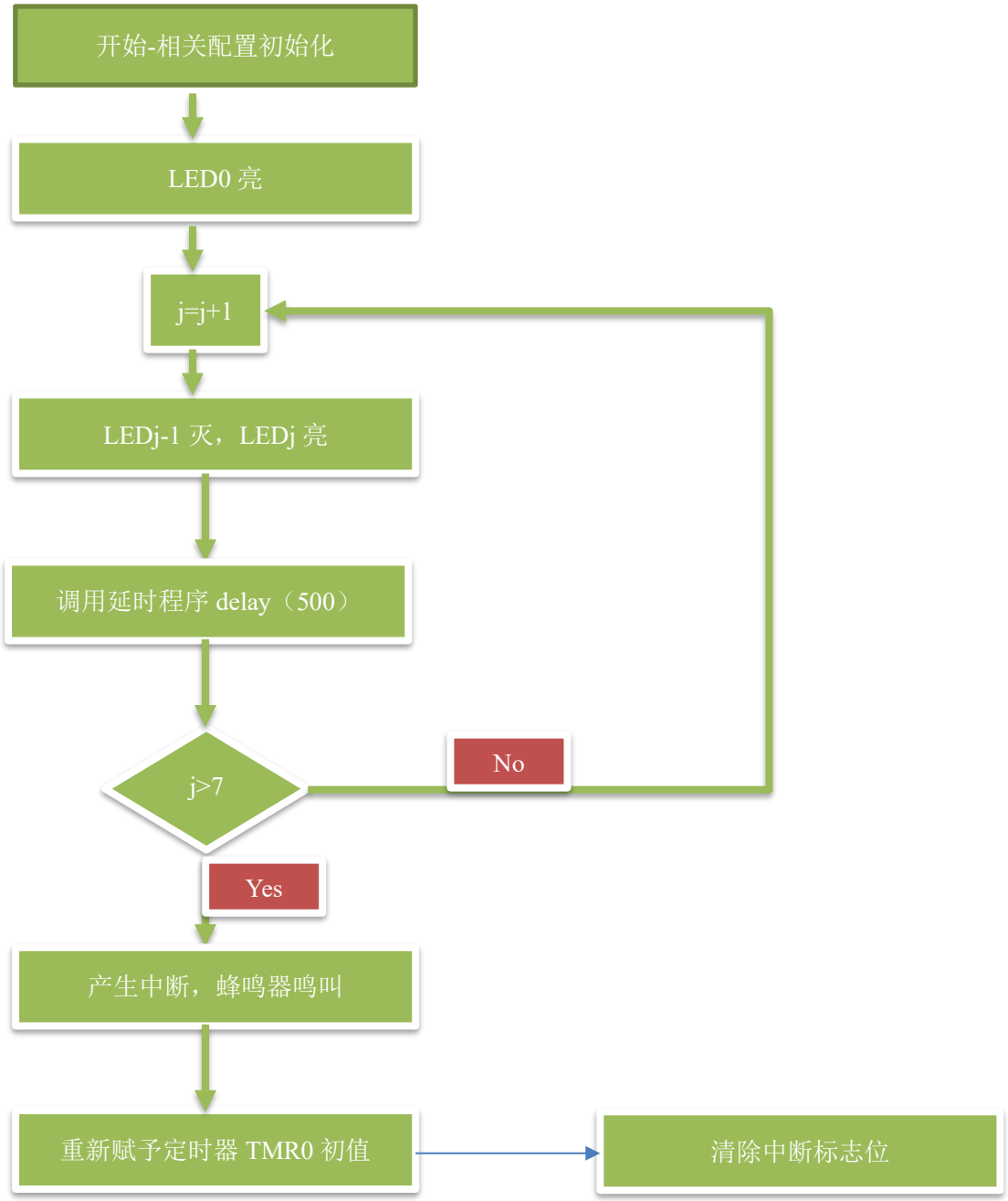
2，详细写出任务 1 中定时器初值的计算过程。

选用 12M 晶振

256*（256-Y）*（4/12）=10000

解得 Y=139=0x8B

3，画出例程 2 的程序流程图。



4，讨论定时器查询方式和中断方式的差异和优缺点。

中断方式会自动对中断标志进行扫描和捕获，而查询则需自己实现对中断标志的扫描。

中断方式的优点在于，能较为实时地捕获到中断标志位，而不需要像查询方式那样需要等到查询程序的开始才能进行中断捕获。

实验五

实验项目名称：PIC 单片机 PWM 实验

一、报告要求

1，对例程 1 的每一句程序进行注释。描述如何改变寄存器及其作用。

```
#include <pic.h>

void PWMIni(void);

void main(void) {
    TRISD = 0x00;        // D端口全输出
    PORTD = 0xff;        // D端口全输出高电平
    PWMIni();            // 初始化配置
    while(1);            // 进入循环
}

void PWMIni(void) {
    TRISC2 = 0;                      // RC2=0,设为输出模式
    T2CON = (T2CON & 0xfc) | 0x03;   // T2CON[2:0]=1x, TRM2分频比设为16:1
    TMR2=0;                          // 清空TMR2[7:0]
    PR2=187;                         // [4M]=62, [12M]=187
    CCP1CON = 0b00111111;            // CCP1CON[3:0]=11xx, CCP1-PWM模式
                                     // CCP1CON[5:4]=11, 脉宽寄存器低二位设为11
    CCPR1L = 46;                     // [4M]=15, [12M]=46

    TMR2ON = 1;                      // TMR2 开始计时
}
```

2，详细写出任务 1 中定时器初值的计算过程。

【4M 晶振, 1KHZ, 25%占空比】

1x10^(-3) = 4x(1/4)x10^(-6) x 16 x (1+PR2)

1x10^(-3) x25%= 4x(1/4)x10^(-6) x 16 x CCPR1L

解得 PR2 = 62, CCPR1L = 15

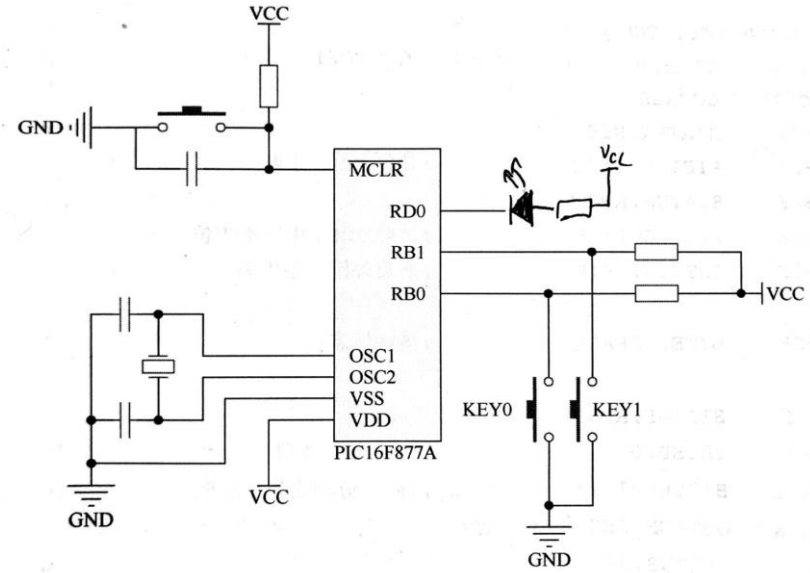
【12M 晶振, 1KHZ, 25%占空比】

1x10^(-3) = 4x(1/12)x10^(-6) x 16 x (1+PR2)

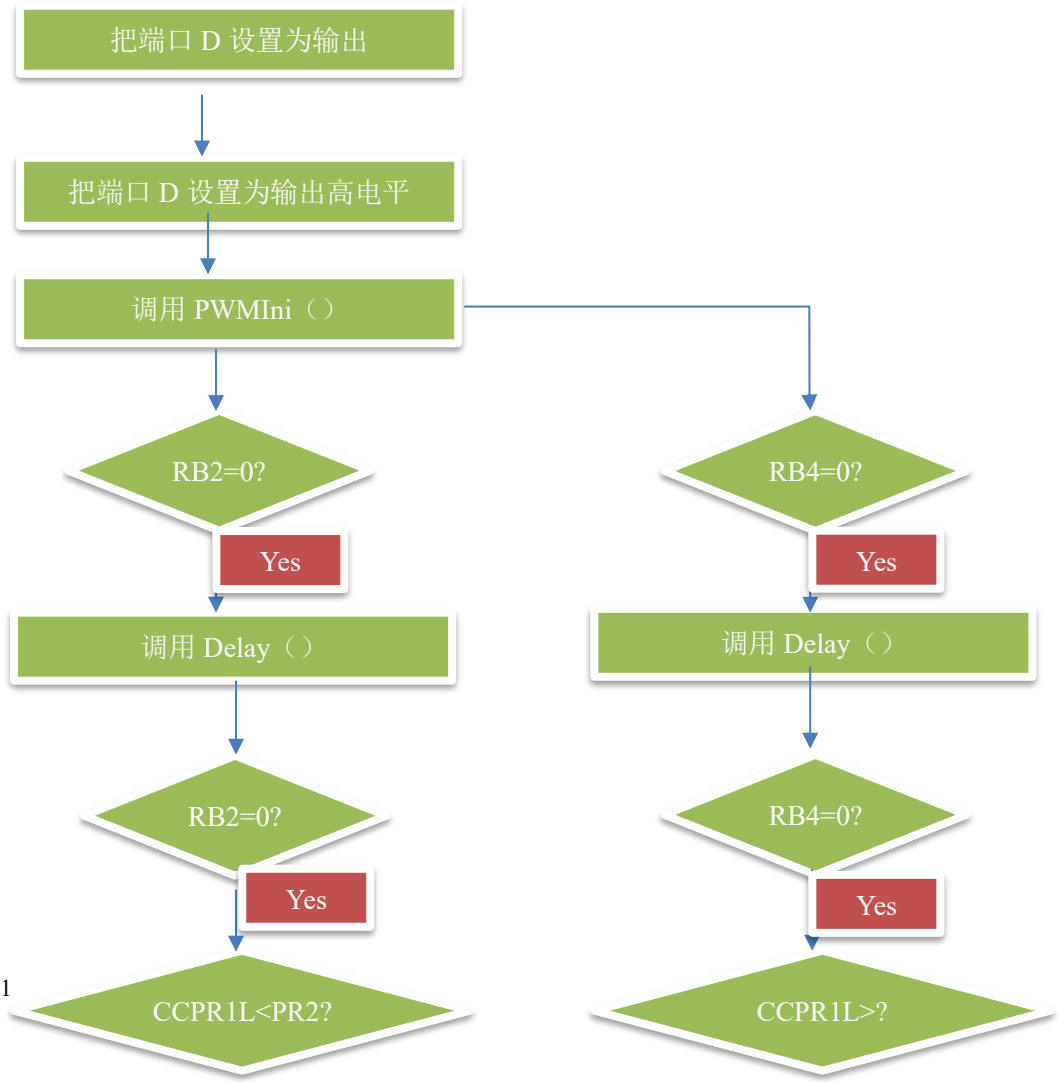
1x10^(-3) x25%= 4x(1/12)x10^(-6) x 16 x CCPR1L

解得 PR2 = 187, CCPR1L =46

3，画出任务 2 的电路图。



4，画出任务 2 的程序流程图。



实验六

实验项目名称：PIC 单片机 AD 采样实验

一、报告要求

1，对例程 1 的每一句程序进行注释。

```
#include  <pic.h> //加载头文件CONFIG (XTWDTDIS ELVPDIS)
unsigned  char GetKeyValue (void) ;
void Delay (int t) ;
void INIAD (void) ;
void main (woid )
{
    unsigned char RH, RL;
    TRISD=0X00;//设置RD为输出
    INIAD();//调用INIAD函数
    TRISA1=0;//设置RA1为输出
    TRISA2=0;//设置RA2为输出
    RA2=1;//设置RA2为输出高电平
    PORTD=0X12;//点亮LED灯
    RH=0;
    RL=0;
    while (1)
    {
        GO=1;//启动A/D转换
        while(!ADIF);//等待A/D转换结束
        RH= ADRESH;//把转换结果送至RH和RL
        RL= ADRESL;
        RA1=1;RA2=0;//选择第一位数码管
        PORTD=RL;//把低八位送至端口D
        Delay(1) ;
        RA1=0;RA2=1;//选择第二位数码管
        PORTD=RH;//把高八位送至端口D
        Delay(1);
    }
}
void INIAD(void)
{
    TRISA0=1;      //设置RA0为输入
    ADCON1=0x0E;   //选择左对齐方式，并选择RA0为模拟输入通道
    ADCON0=0x01;   //系统时钟频率fosc/2，选择RA0为模拟输入通道，ADC进入准备
    ADIF=0;        //未发生A/D转换中断
}

void Delay(int t) //定义延时函数
{
    int j,i;
    for(i=t; i>0;i--)
    {
       for(j=0;j<100;j++);
    }
}
```

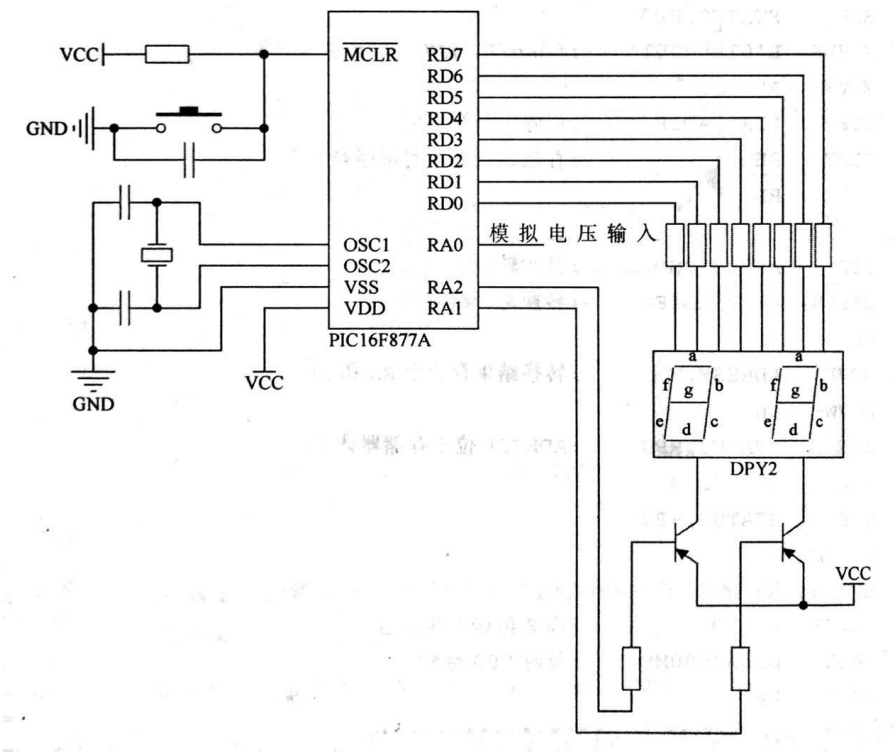
2，写出例程 1 中，对 AD 相关寄存器进行初始化的值（括号处）。并解释为什么这样设置。

GO=1 时要启动 A/D 转换，

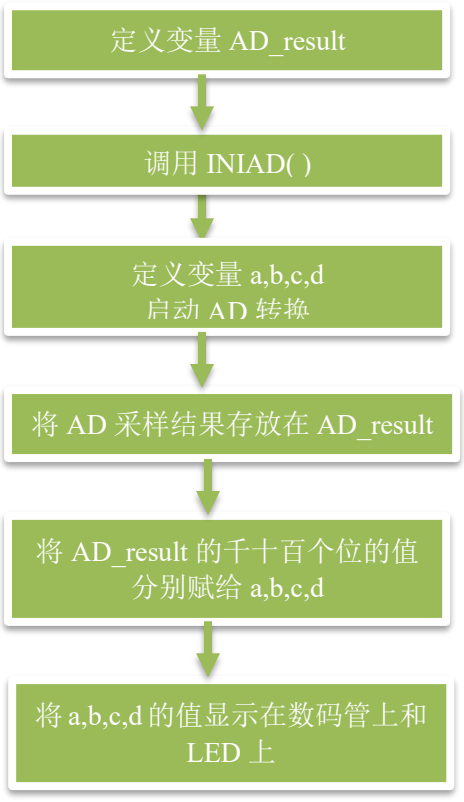
ADCON1=0x0E 是要选择左对齐方式，并选择 RA0 为模拟输入通道

ADCON0=0x01 是要选择系统时钟频率 fosc/2，选择 RA0 为模拟输入通道，ADC 进入准备

3，画出 2 的硬件电路图。



4，画出任务 2 的程序流程图，包括主程序，子程序。



实验七、八

实验项目名称：PIC 单片机串行通信

一、报告要求

1，填空完成例程 1。

```
#include <pic.h>

void delay(int t);
void init_USART();

void main(void) {
    init_USART();
    while(1) {
        TXREG = 0x32;
        while(!TRMT);
        delay(10);
    }
}

void init_USART() {
    TRISC6 = 0;
    TRISC7 = 1;
    TXSTA = 0b00100100;
    RCSTA = 0b10010000;
    SPBRG = 25;
    INTCON = 0x00;
}

void delay(int t) {
    int i, j;
    for(i=t; i>0; i--) {
        for(j=0; j<100; j++);
    }
}
```

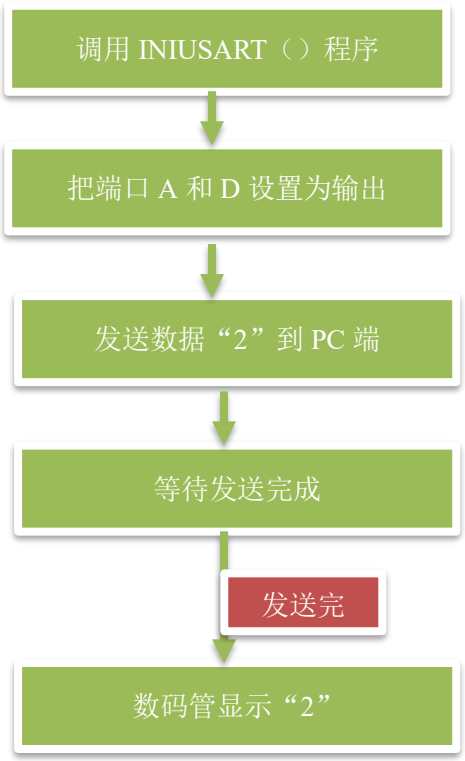
2，写出成功通信的波特率值以及其对应的 SPBRG 值。

【4M 晶振-高速挡】

9600 = (1/4)x10^(-6)/[16x(SPBRG+1)]

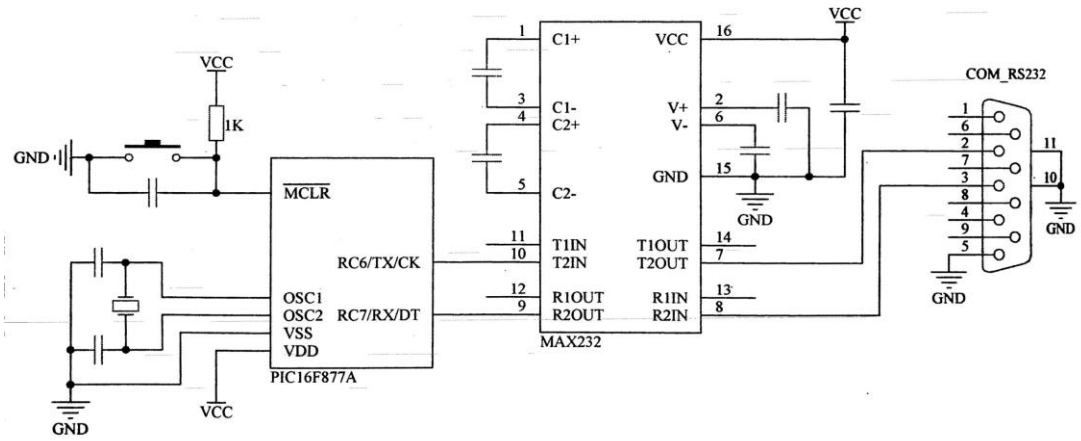
解得 SPBRG = 25

3，写出任务 3 程序流程图，并贴出程序




```
#include <pic.h>

void delay(int t);
void init_USART();

void main(void) {
    init_USART();
    while(1) {
        if(RCIF) {
            PORTD = RCREG-0x30;
            RCIF=0;
        }
    }
}

void init_USART() {
    TRISC6 = 0;
    TRISC7 = 1;
    TRISD = 0;
    TRISA1 = 0;
    RA1 = 1;
    TXSTA = 0b00100100;
    RCSTA = 0b10010000;
    SPBRG = 25;
    INTCON = 0x00;

}

void delay(int t) {
    int i, j;
    for(i=t; i>0; i--) {
        for(j=0; j<100; j++);
    }
}
```

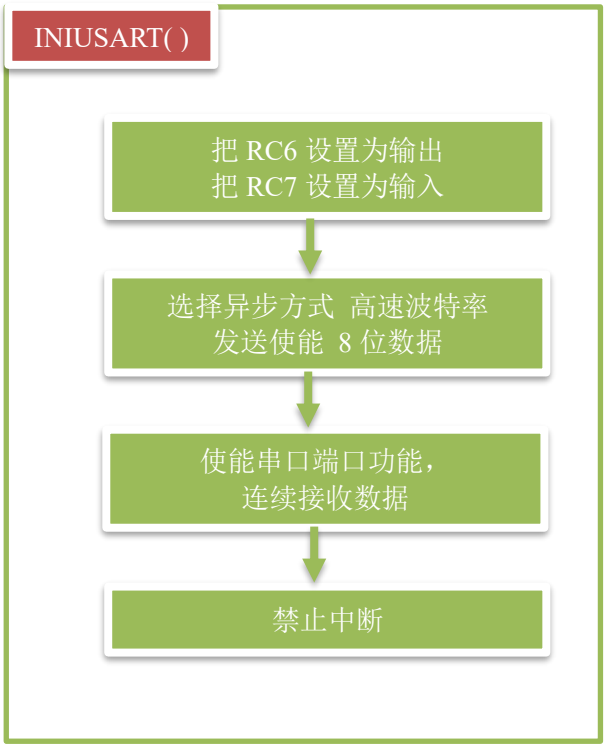
4，画出任务 4 的硬件电路图及程序流程图，包括主程序，子程序。




```
#include <pic.h>

void delay(int t);
void init();
unsigned int RH, RL;
unsigned int temp;

void main(void) {
    init();

    while(1) {
        ADGO=1;
        while(!ADIF);
        RH=ADRESH;
        RL=ADRESL;

        temp = RH*256 + RL;  // 【10 bit】 0~1024

        // Y***
        TXREG=(temp/1000) + 0x30;
        while(!TRMT);
        delay(5);

        // *Y**
        TXREG=(temp/100)%10 + 0x30;
        while(!TRMT);
        delay(5);

        // **Y*
        TXREG=(temp/10)%10 + 0x30;
        while(!TRMT);
        delay(5);

        //  ***Y
        TXREG=temp%10 + 0x30;
        while(!TRMT);
        delay(5);
```

```
        // \n
        TXREG=0x0A;
        while(!TRMT);
        delay(5);

        delay(400);
    }
}

void init() {
    // IO_configure
    TRISC6 = 0;
    TRISC7 = 1;

    // USART_configure
    TXSTA = 0b00100100;
    RCSTA = 0b10010000;
    SPBRG = 25;
    INTCON = 0x00;

    // AD_configure
    TRISA0=1;         //RA0 as AD input
    ADCON1=0x8E;
    ADCON0=0x41;
    ADIF=0;
}


void delay(int t) {
    int i, j;
    for(i=t; i>0; i--) {
        for(j=0; j<100; j++);
    }
}
```


INIAD( )
将 RA0 设置成输入
将 RA1,RA2,RA3,RA5,RD 设置成输出
选择右对齐方式，并选择
RA0/AN0 为模拟输入通道
选择频率 focs/2 并开启 AD 转换
清除标志位


INIUSART( )
把 RC6 设置为输出
把 RC7 设置为输入
选择异步方式 高速波特率
发送使能 8 位数据
使能串口端口功能，
连续接收数据
禁止中断