# 华中科技大学

## 《8051单片机》实验报告

题目： 频率计功能

院 （系）：电子信息与通信学院

专业班级：电信卓越1701

姓 名：

学 号：

2020 年 6 月 6 日

一、实验目的

请将给出的Proteus图用51单片机完成一个频率计功能。

1、设置一个或几个显示缓冲区为DISPBUFF[]，存储显示的频率，占空比，周期等。

2、用输入设备切换显示频率及相应参数，模式0：显示频率，模式1：显示周期，模式2：显示占空比。4位数码管的可以加上三个LED来指示单位。

3、频率测量范围200Hz-10KHz。（可以采用测量多个周期取平均的方法减少误差）

4、自己完成外部信号源的连接。用7414来整形。

5、主程序用C编写，另外必须包括一个汇编子程序。

6、可以增加部分功能。（必须在文档中说明）

按要求上交实验报告（几种模式对应Proteus仿真截图一定要有）。

二、实验步骤

1、中断相关设置

本次实验要求使用四个中断方式，即定时器T0、T1中断，两个外部中断。首先分析四个中断功能如下：

定时器T0：gate们置高，主要用来测量1s中高电平出现的时间，用来求占空比。

定时器T1: 用来定时，1s到达时根据测量时间求频率、占空比、周期。

外部INT0：接输入信号，高电平到达中断(下降沿中断)

外部INT1：输出模式选择，含有模式0，1，2三种选择。

根据上面各中断的功能和作用，结合IE和TCON以及TMOD寄存器结构可以设置中断初始化程序：

IE=0x8f;

TCON=0x55; //默认把外部中断也打开。

TMOD=0x19; //定时器0gate位置1

TH0=0;

TL0=0; //计时初始为0

TH1=0x3c;

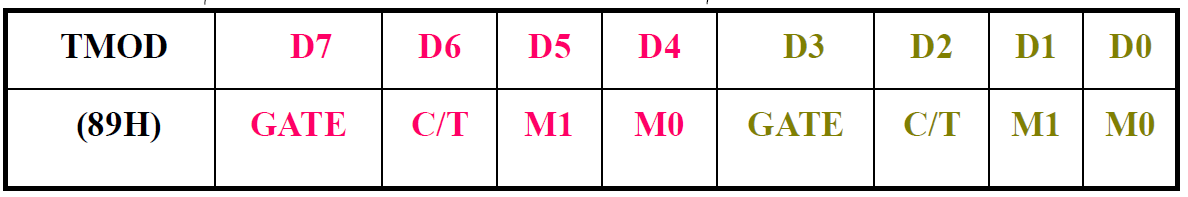
TL1=0xb0; //定时器初始值，定时50ms，20次为1s



图一：中断控制寄存器IE



图二：中断方式选择寄存器TCON



图三：工作模式寄存器TMOD

2、程序实现思路

本次实验通过汇编子程序实现显示缓冲区BCD到数码管显示的译码功能，通过汇编子程序实现输入数字输出相应BCD码，具体实现参考源代码(汇编子程序)。

本次实验主要是设计频率和占空比测量程序，最后将所得结果经过转换传入给四位数码管显示程序得以显示结果。

Main()函数、初始化函数、显示函数display()、数值转化函数

main()函数中主要是初始化中断、显示结果display()。

所得数字结果需要经过buffer\_display()函数转化成数组，再调用四位数码管显示函数输出结果。Display()函数中需要根据mode的不同，设计三种输出模式，再将结果输出。具体实现见最后程序源代码。

中断服务程序

1、int timer0() interrupt 1 //定时器0中断

该中断主要是计算1s中正电平所占时常，在计数器溢出时才能中断，且因为gate设为1，只有高电平时，该定时器的值才会增加。具体实现见源程序。

2、int timer1() interrupt 3 //定时器1中断，用来计算1s时间

该中断主要是计算是否达到1s达到1s后根据测量结果计算频率、占空比、周期等。

3、int MODE1() interrupt 0 //外部按键中断

该中断用于切换显示模式，含有0，1，2三种模式，具体实现见源程序。

4、int mode0() interrupt 2 //外部中断

该中断在输入信号为正脉冲时发生，首次中断时应该开启两个计数器开始计数，并且每次中断将脉冲数量加1.用于计算频率。

结果计算和转化

结果计算中：1s中脉冲数量和频率相等。占空比等于高电平时间所占比例。周期为频率的倒数。具体实现可见源程序。

数值结果需要转化为四位数码管输入程序所要求的形式，这里考虑将结果转化为一个个值存入数组中，数组可以直接通过输入函数和汇编子程序输入。程序设计如下：

void buffer\_display(uint x) //输入频率等结果，将结果显示在频幕上

{

int i;

for(i=3;i>=0;i--)

{

DISPBUFF[i]=x%10;

x=x/10;

}//将数值换成buffer

for (i=0;i<=3;i++)

{

serial\_data(bcd\_to\_led(DISPBUFF[3-i])&0xff); //调用汇编子程序

}

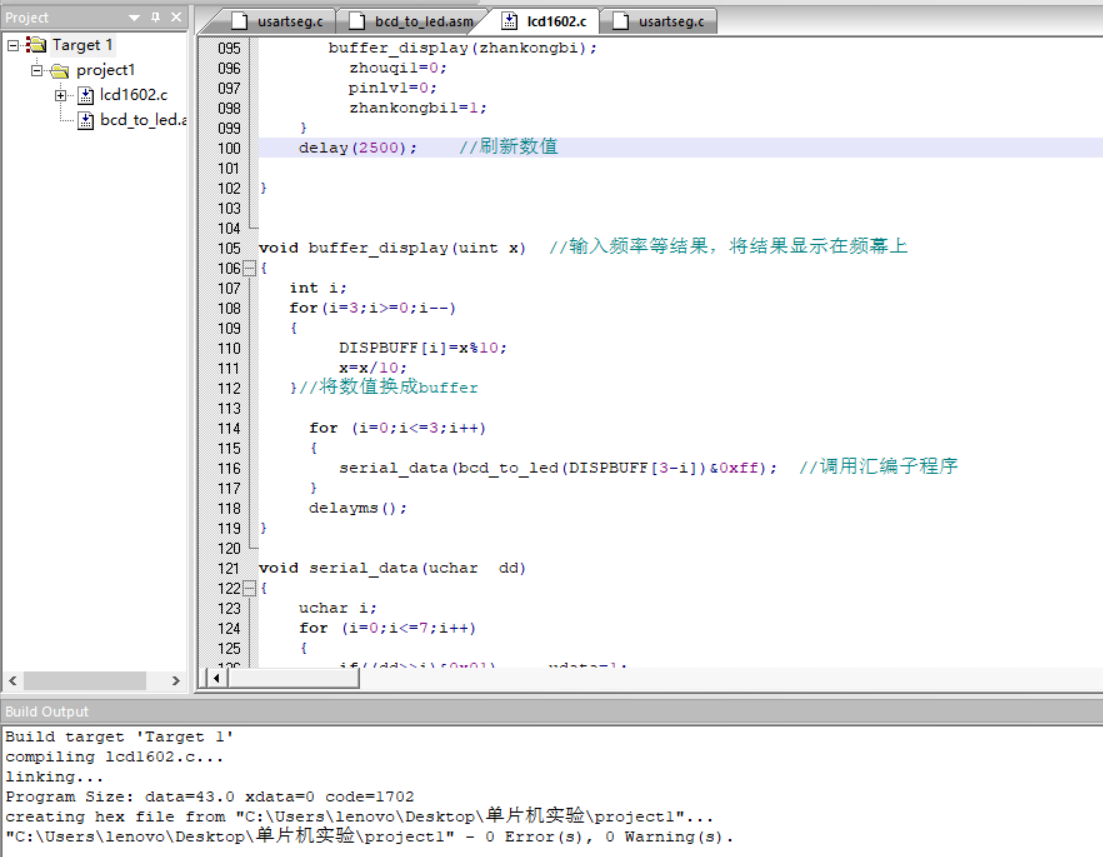
delayms();

}

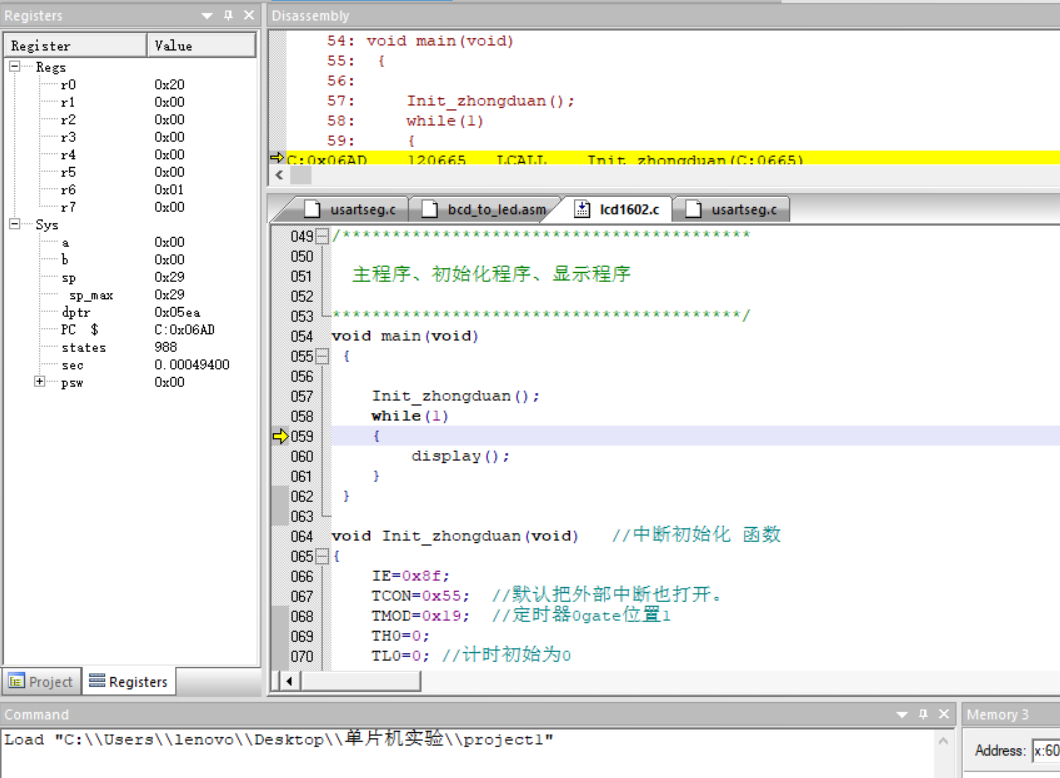
具体实现上述程序最终形成完整程序即可得到满足要求的结果。

1. 实验结果

**1、编译成功界面、调试界面**



图四：编译成功界面

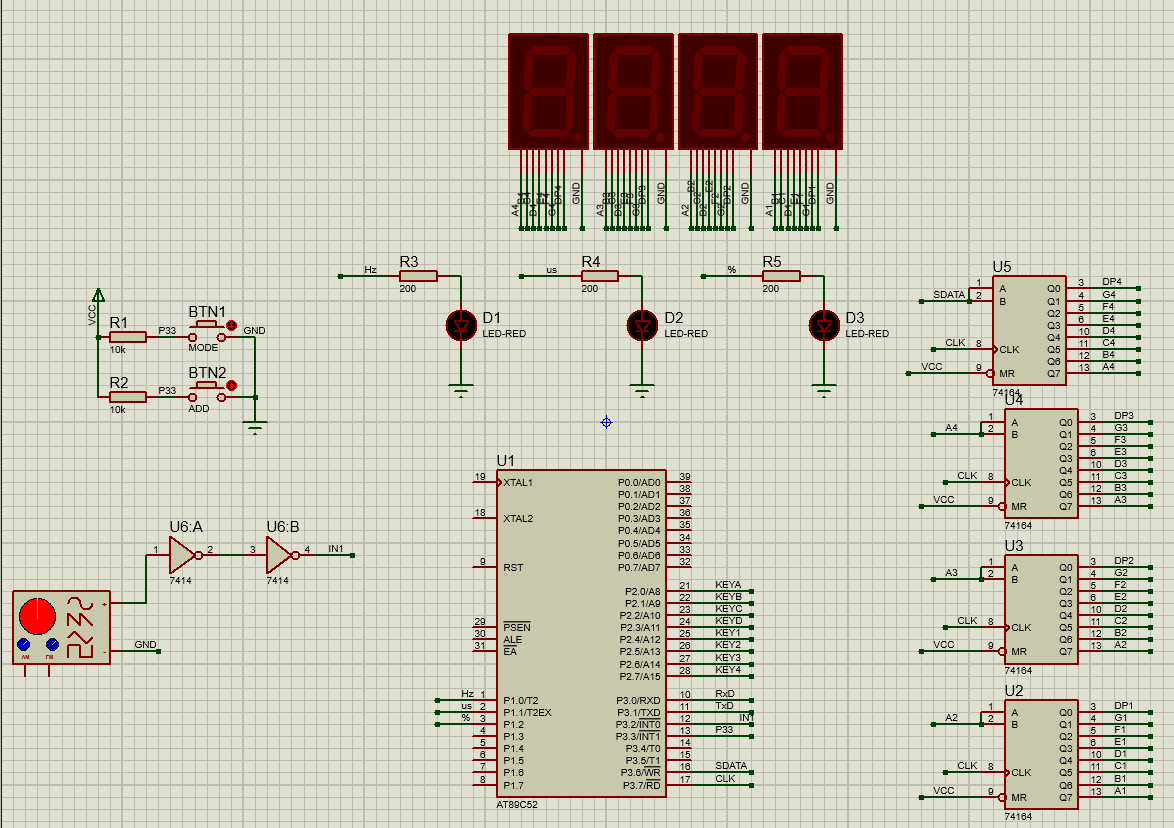


图五：调试界面

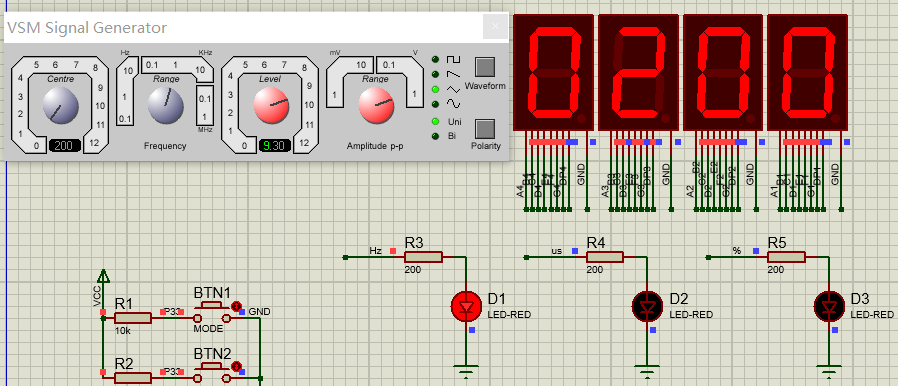
**2、仿真结果**

Proteus整体仿真截图如下所示：

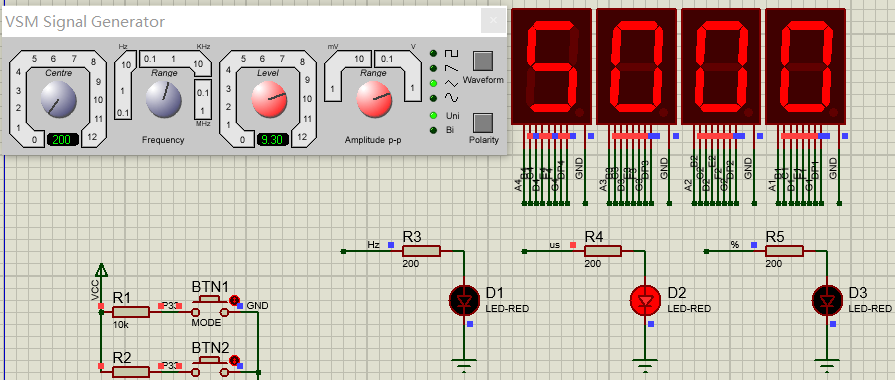
下图中添加了题目要求的外部信号源的连接以及三个指示单位的二极管。



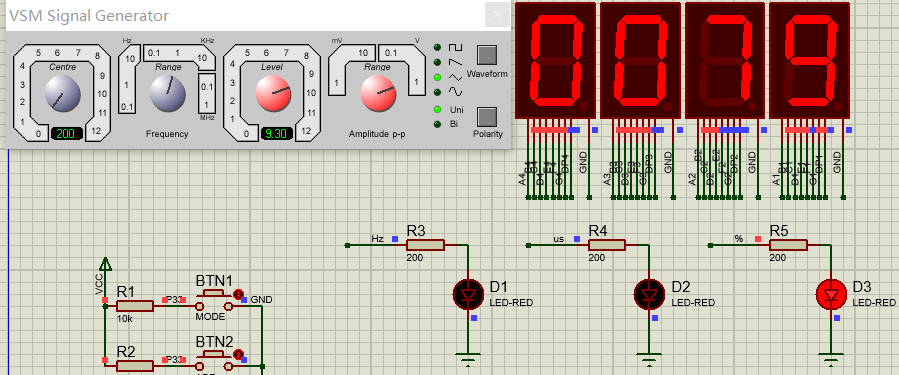
**200Hz三角波信号的三种输出模式结果图**



频率(Hz):第一个三极管发光

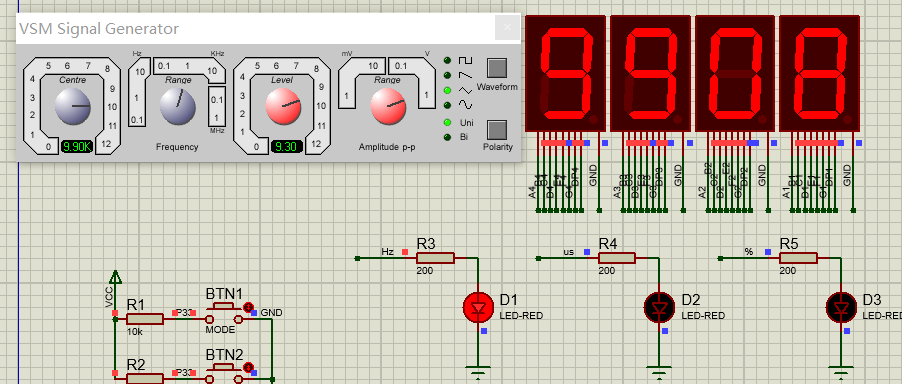


周期(us):第二个三极管发光

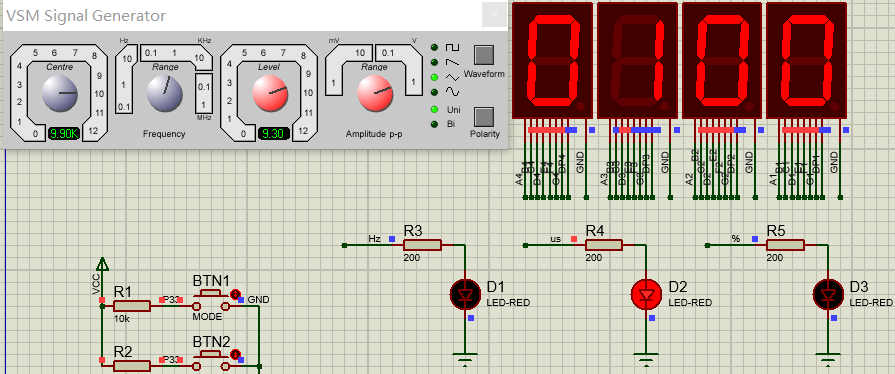


占空比(us):第三个三极管发光

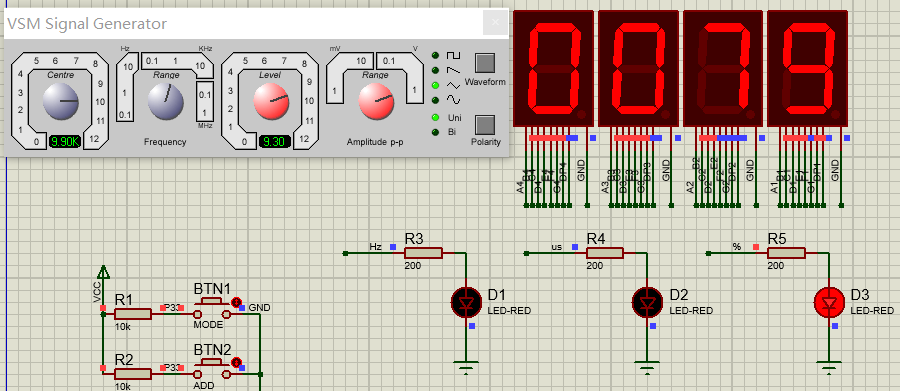
**9.9KHz三角波信号的三种输出模式结果图(四个数码管最多显示10K)**



频率(Hz):第一个三极管发光

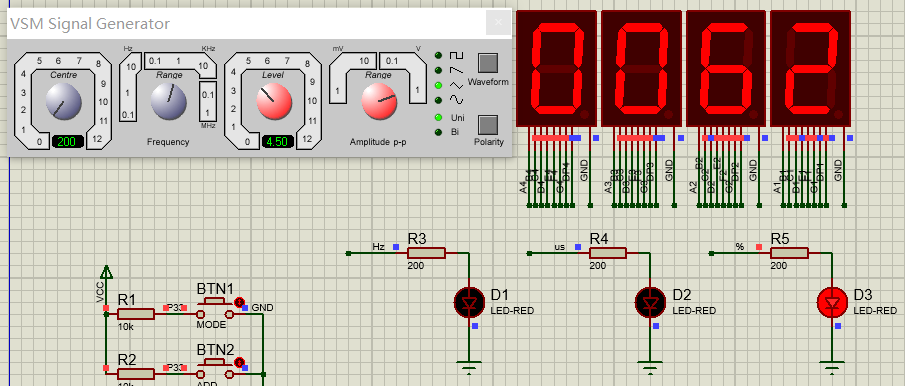


周期(us):第二个三极管发光

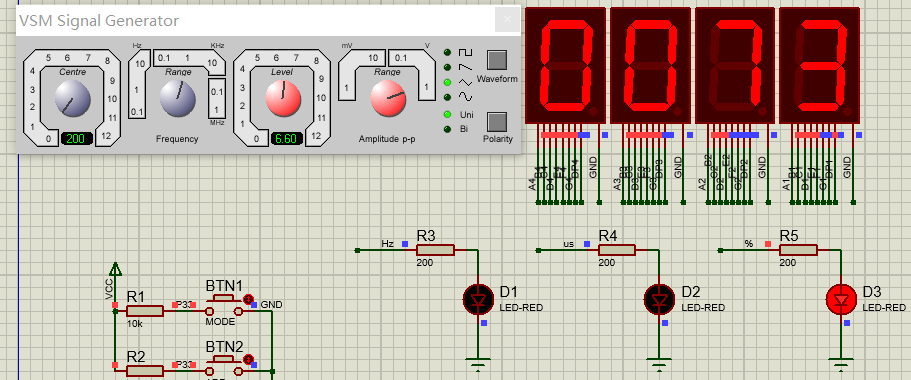


占空比(us):第三个三极管发光

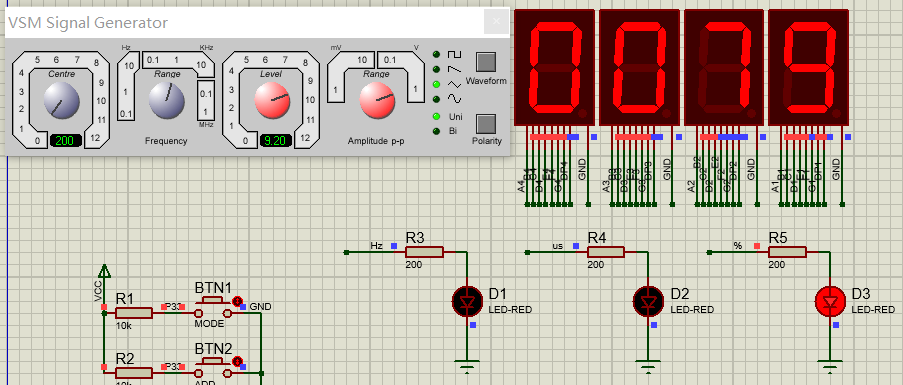
**调节幅值改变占空比(固定频率为200Hz)**



4.5V-----62%



6.6V----73%



9.2V-----79%

四、实验总结

该实验通过中断实现频率和占空比的测量。通过本次实验掌握了外部中断的使用和定时器的使用，加强了自己利用C51编程的能力。同时也掌握了数码管的显示等等操作，能够将结果显示在四位数码管上。进一步加强了利用keil4调试的能力。Proteus的掌握也更上一层楼。

五、实验说明

最后结果需要等到稳定显示时才会显示正确，该程序可以实现数码管结果动态更新，调节输入信号结果会动态更新但大概会有1s左右延迟。

实验是出现了占空比计算不出的问题，主要原因是中间的计算结果过大超出了C51的支持范围，后来通过排查找到原因，最终设法改变式子，将中间结果变小得到理想结果。

实验仍然存在一定误差，由于时间问题误差未能完全消除，只通过表达式消除部分误差。

六、程序源代码

1. Lcd1602.c文件

// 该程序实现频率计和测占空比，开始数据可能不准，等待数据刷新稳定即可。

#include "reg51.h"

#include "intrins.h"

// Define new types

typedef unsigned char uchar;

typedef unsigned int uint;

extern int bcd\_to\_led(uint x); //汇编子程序，传入DISPBUFF[4]中的数据

sbit udata=P3^6;

sbit uclk=P3^7;

sbit zhouqi1=P1^1;

sbit pinlv1=P1^0;

sbit zhankongbi1=P1^2;

// 程序定义

void serial\_data(uchar dd);

void Init\_zhongduan(void); //中断初始化

void get\_result(); //1s时计算频率，占空比，周期。

void delay(uint); //延时函数

void display(void); //显示函数

void buffer\_display(uint x); // 输出缓冲区数据

//全局变量定义

//button1\_mode->输出模式选择，外部中断1调节，初始化为0.

//maichong\_number-》1s内的脉冲数量，用于计算频率、周期

//timer0\_number->定时器0溢出数量

//timer1\_number->定时器1中断次数，20次到1s

//timer1\_flag->达到1s的标志，到1s才可输出结果，删了此变量

uint button1\_mode = 0,maichong\_number=0,timer0\_number=0,timer1\_number=0,timer1\_flag = 0;

uint frequent=0; //频率，占空比，周期

unsigned long int zhankongbi=0;

unsigned long zhouqi=0;

uint DISPBUFF[] = {0,0,0,0}; //设置6位数进行显示

void delayms()

{

uint j;

for(j=0;j<=30000;j++) ;

}

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

主程序、初始化程序、显示程序

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

void main(void)

{

Init\_zhongduan();

while(1)

{

display();

}

}

void Init\_zhongduan(void) //中断初始化 函数

{

IE=0x8f;

TCON=0x55; //默认把外部中断也打开。

TMOD=0x19; //定时器0gate位置1

TH0=0;

TL0=0; //计时初始为0

TH1=0x3c;

TL1=0xb0; //定时器初始值，定时50ms

}

void display() //显示函数，不同模式显示不同结果

{

if(button1\_mode == 0) //频率

{

buffer\_display(frequent);

zhouqi1=0;

pinlv1=1;

zhankongbi1=0;

}

else if(button1\_mode == 1) //周期

{

buffer\_display(zhouqi);

zhouqi1=1;

pinlv1=0;

zhankongbi1=0;

}

else if(button1\_mode == 2) //占空比

{

buffer\_display(zhankongbi);

zhouqi1=0;

pinlv1=0;

zhankongbi1=1;

}

delay(2500); //刷新数值

}

void buffer\_display(uint x) //输入频率等结果，将结果显示在频幕上

{

int i;

for(i=3;i>=0;i--)

{

DISPBUFF[i]=x%10;

x=x/10;

}//将数值换成buffer

for (i=0;i<=3;i++)

{

serial\_data(bcd\_to\_led(DISPBUFF[3-i])&0xff); //调用汇编子程序

}

delayms();

}

void serial\_data(uchar dd)

{

uchar i;

for (i=0;i<=7;i++)

{

if((dd>>i)&0x01) udata=1;

else udata=0;

uclk=1; uclk=0;

}

}

void delay(uint j)

{

uchar i = 60;

for(; j>0; j--)

{

while(--i);

i = 59;

while(--i);

i = 60;

}

}

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

中断处理函数-----四个中断都用了

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

int MODE1() interrupt 2 //按键控制外部1中断,切换显示模式

{

if(button1\_mode < 2) button1\_mode++;

else button1\_mode = 0;

}

int mode0() interrupt 0 //外部中断0，输入正电平时产生中断。

{ //中断正电平数目+1

if(maichong\_number == 0) //0个周期了。最开始重置定时器

{

TH0=0;

TL0=0; //计时初始为0

TH1=0x3c;

TL1=0xb0; //定时器初始值，定时50ms

}

maichong\_number++; //脉冲数量加一

}

int timer0() interrupt 1 //计数器0中断，溢出时才能中断，一旦来正脉冲就会计时。

{

TH0=0;

TL0=0; //计时初始为0。

timer0\_number++;

} //可以计算1s钟正脉冲所占时常

int timer1() interrupt 3 //定时器1中断，用来弄1s时间

{

TH1=0x3c;

TL1=0xb0; //重新设置定时初值，50ms ，20次为1s

timer1\_number++;

if(timer1\_number == 20) //到达1s，计算频率占空比。

{

get\_result();

timer1\_flag = 1;

maichong\_number = 0; //脉冲数置0

timer1\_number = 0; //计数置0

timer0\_number = 0; //溢出次数置零

Init\_zhongduan(); //中断初始化

TH0=0;

TL0=0; //计时初始为0。

}

}

void get\_result(void) //根据中断得出的数据得处结果

{

//这边数字太大了，不能直接256\*256，拆开算，可能单片机自己的问题。

float t1,t2;

t1 = (timer0\_number\*256/100)\*256/100;//

t2 = (TH0\*(256)+TL0)/10000;

zhankongbi =(t1+t2);//结果为百分比

frequent = maichong\_number - maichong\_number/200\*18 + maichong\_number/5000\*56; //频率，含有修正

zhouqi = 1000000/frequent;

}

1. bcd\_to\_led文件

DE SEGMENT CODE

PUBLIC \_bcd\_to\_led

RSEG DE

\_bcd\_to\_led: MOV A, R7

MOV DPTR, #TAB

MOVC A, @A+DPTR

MOV R7,A

RET

TAB:DB 3fH,06H,5bH,4FH,66H,6dH,7dH,07H,7fH,6fH

END