به نام خدا

پروژه پایانی درس ساختار کامپیوتر و میکروپروسسور

جناب آقای دکتر جاهد

سرکار خانم مهندس کشوری

رادمهر کریمیان

98103556

ترم 2–1399

فهرست:

- **مقدمه،موضوع و اهداف کامل مسئله و معرفی ادوات مورد نظر**

در این پروژه هدف ما ساخت و تولید سیستمی برپایه سیستم 8051 اینتل بوده است که در آن دمای کوره ای روی ال سی دی نشان داده شده و همچنین در حافظه ثبت شود.

در این سیستم ما از ماژول های زیر استفاده میکنیم:

1. 8031

این میکروکنترلر از خانواده هشت بیتی Intel MCS-51 بوده است که درواقع شامل128 بایت رم داخلی و تا 64 کیلوبایت رم خارجی را دارد.

2. 8255

PPI 8255 یک دستگاه ورودی و خروجی قابل برنامه ریزی با هدف کلی است که برای ارتباط CPU با دنیای خارج آن مانند ADC، DAC ، صفحه کلید و غیره طراحی شده است. ما می توانیم آن را با توجه به شرایط داده شده برنامه ریزی کنیم. تقریباً با هر ریز پردازنده قابل استفاده است. از سه درگاه ورودی / خروجی دو طرفه 8 بیتی یعنی PORT A، PORT B و PORT C تشکیل شده است.

3. **رم و رام خارجی**

برای اینکه دستورات و همچنین دمای خوانده شده را در سیستم ذخیره کنیم از رم و رام های دو کیلوبایتی استفاده میکنیم.

4. 74LS373

IC 74LS373 یک لچ است که از هشت قفل با سه خروجی برای برنامه های سیستم های سازمان یافته باس تشکیل شده است. این یک IC 20 پین است که از هشت خط داده ورودی (D0-D7) و هشت خط خروجی (O0-O7) تشکیل شده است.

5. 74Ls235

SN54 / 74LS245 یک فرستنده / گیرنده اتوبوس هشتتایی است که برای ارتباط داده ای 2 طرفه ناهمزمان 8 خطی بین گذرگاه های داده طراحی شده است. Direction Input (DR) **بسته به سطح منطقی بودن ، انتقال داده ها را از گذرگاه A به گذرگاه B یا گذرگاه B به گذرگاه A کنترل می کند. از** Enable input (E) **می توان برای جداسازی اتوبوس ها استفاده کرد.**

6. LCD

ال سی دی همان طور که میشناسیم نمایشگر LCD 16x2 ماژول بسیار اساسی است و به طور معمول در دستگاه ها و مدارهای مختلف استفاده می شود. یک LCD 16x2 به این معنی است که می تواند 16 حرف در هر خط نمایش دهد و 2 خط از این دست وجود دارد. در این LCD هر کاراکتر در ماتریس پیکسل 75x نمایش داده می شود.

7. سنسور دما

LM35 دستگاه اندازه گیری دما با ولتاژ خروجی آنالوگ متناسب با دما است. ولتاژ خروجی را در سانتیگراد (سانتیگراد) تأمین می کند. به مدار کالیبراسیون خارجی احتیاج ندارد.

8. A/D

ADC0804 یک IC (مدار مجتمع) است که ولتاژ آنالوگ ورودی را به خروجی دیجیتال معادل آن تبدیل می کند. مبدل دیجیتال آنالوگ به صورت مستقل است. ... نه تنها به اندازه گیری دما محدود می شود بلکه می تواند در هرجایی که برای اندازه گیری سیگنال آنالوگ لازم است استفاده شود.

- **فرضیات و روش طراحی**

در این مسئله و پروژه از آنجایی که ما توانایی و آزادی های موجود در دنیا ی واقعی را نداشتیم برخی تمهیدات به شکل زیر درنظر گرفتیم:

مشکل اول و بزرگ عدم وجود چیپ رام استاندارد در نرم افزار پروتئوس بود بنابر این ما به جای 8031 از 8051 استفاده کردیم،علت استفاده از 8051 به این دلیل است که از رام داخلی 8051 استفاده کنیم و دستورات خود را به شکل فایل هگز در آن ذخیره کنیم.

مشکل دوم نبود کوره در نرم افزار پروتئوس بود،برای حل این مشکل نیز ما به شکل بسیار بسیار ساده فرض کردیم که کوره ما در اصل چیزی جز یک سنسور دمایی lm35 نیست و درواقع با بررسی و اتصال این سنسور به ای سی adc0804 و دریافت دما استفاده کردیم.

نکته ای دیگر که در نتیجه این اقدام رخ میداد عدم استفاده از قطار پالس است.

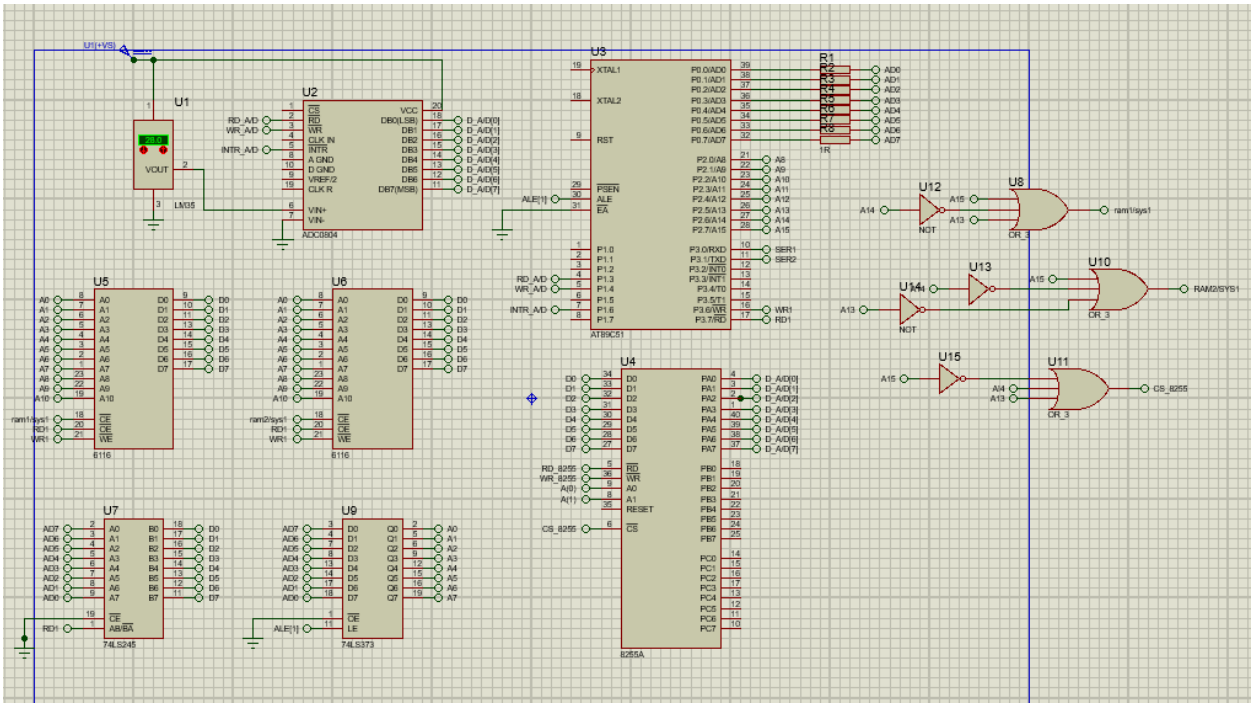
البته ما این پالس را از یکی از پین ها ارسال کردیم تا ارسال شود ولی استفاده ای در کوره نداشت.

نکته دیگر ارسال داده ها با باد ریت 19200 بود که در این حالت باید SMOD را یک قرار میدادیم که در قسمت های بعدی این نکته به خوبی بررسی شده است.

نکات دیگر اعم از چگونه ارسال،چگونی سیو داده ها و .. در قسمت کد آمده است.

- **بلوک دیاگرام سخت افزار:**

راجب اتصالات و بلوک های سیستم یک داریم:



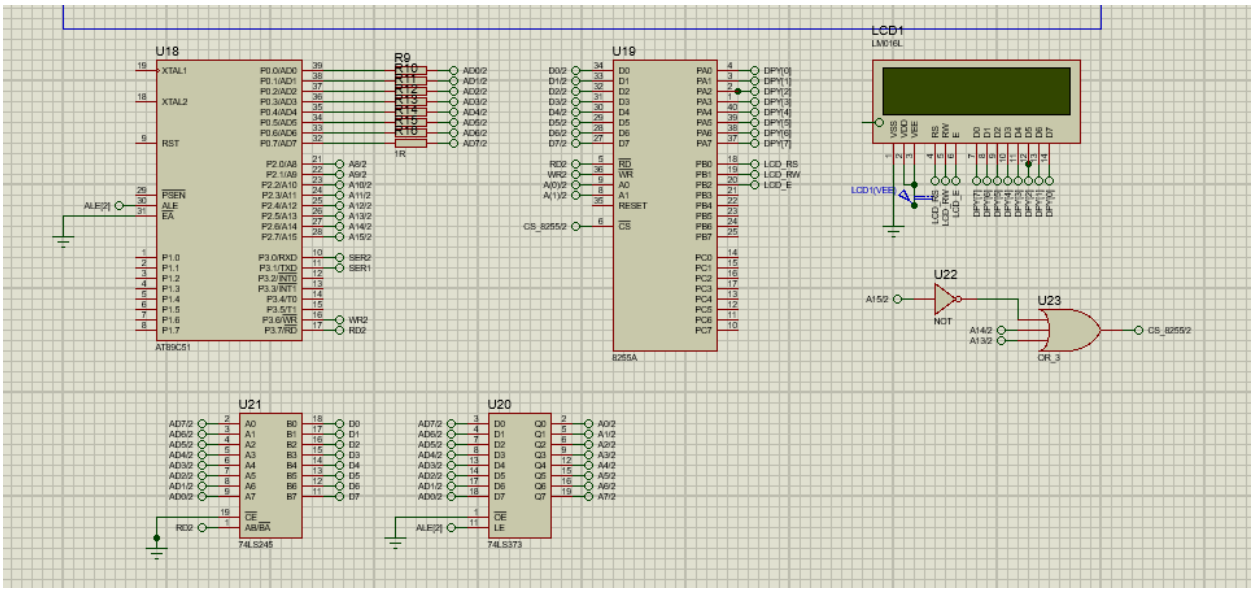
روش اتصال به شکل زیر است

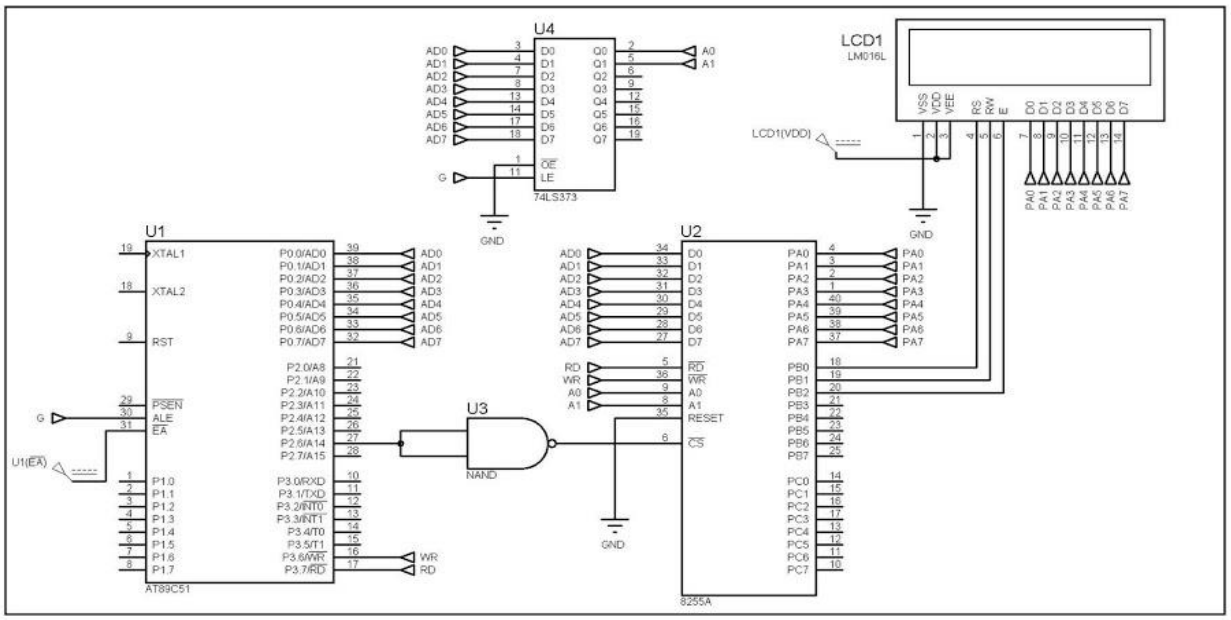
سیگنال دیجیتال حاصل از acd0804 به 8255 متصل می شود.سپس داده ها از 8255 به 8051 منتقل می شود.

سپس در 8051 در رم خارجی سیو میشود و پس از آن به شکل سریالی به سیستم 2 انتقال پیدا میکند.

همچنین تعدادی گیت منطقی برای آدرس دهی پاره ای و سیگنال چیپ سلکت قرار داده شده اند.

راجب اتصالات سیستم دو داریم:



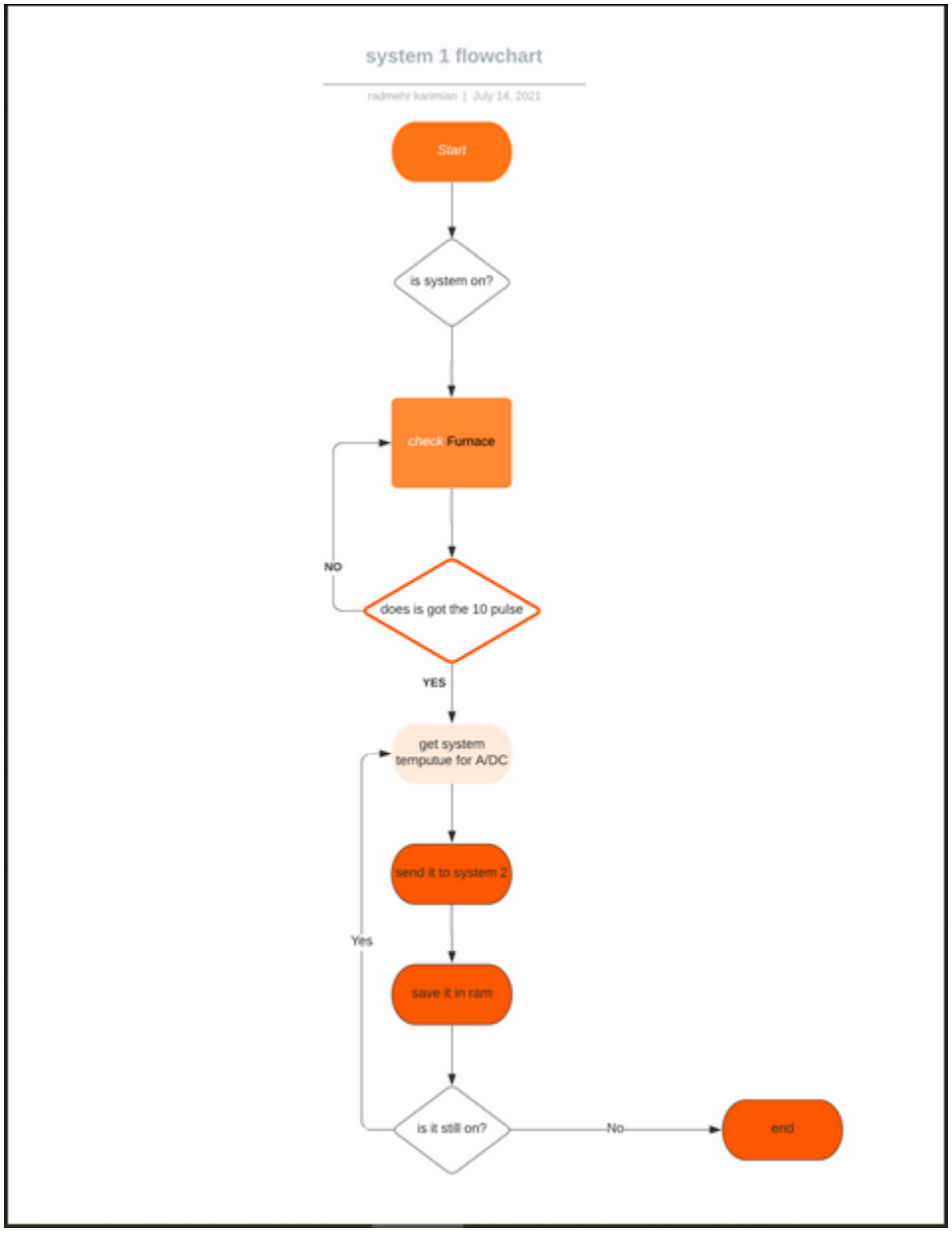


که همانطور که واضح است داده ها از طریق 8051 به 8255 منتقل می شوند و سپس پورت B با ارائه تنظیمات lcd داده ها را نمایش می دهد.

دقت کنید که در این حالت از mode صفر استفاده شده است.

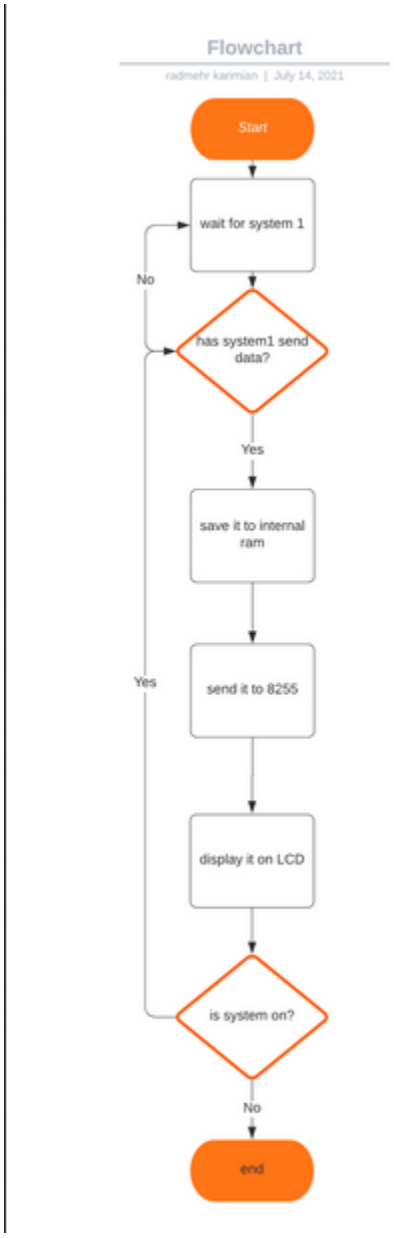
- **فلوچارت و نرم افزار**

راجب فلوچارت سیستم یک داریم:



که هر قسمت از عملیات های ما در هر تابع و به شکل کلی در قسمت مین اجرا می شود.

راجب فلوچارت سیستم دو داریم:

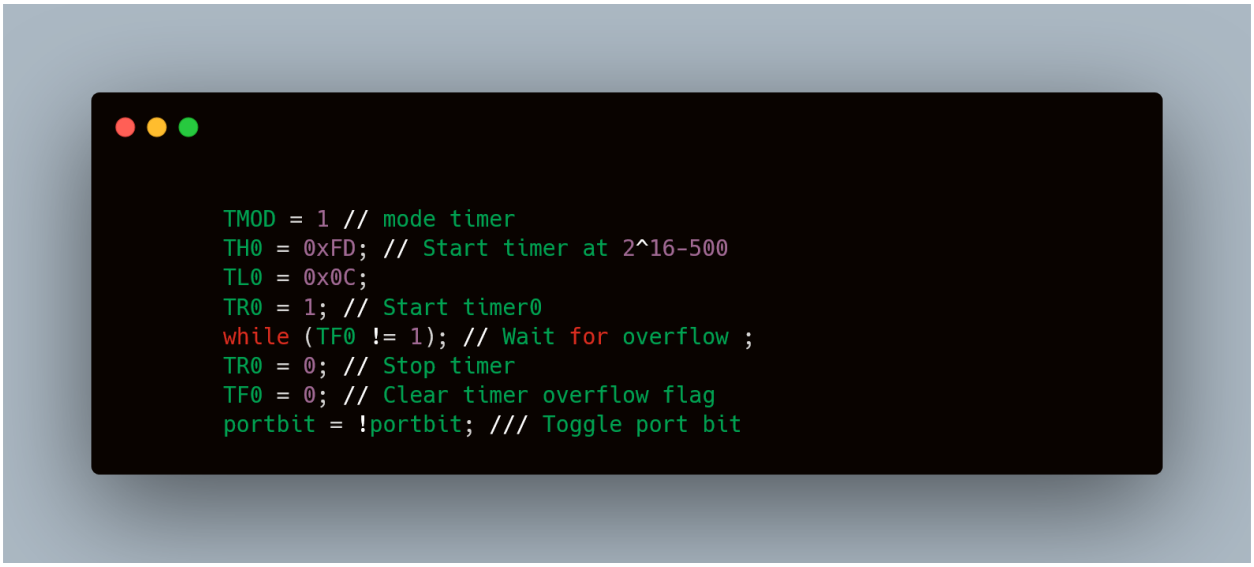


که هر قسمت آن به تفاضیل در قسمت بعد توضیح داده می شود.

- **معرفی مدولار در بخش های مختلف نرم افزار**
- **سیستم یک:**

1. **تابع پالس**
2. **تابع ارسال سریال**
3. **تابع تبدیل سیگنال**
4. **تابع ذخیره داده**
5. **تابع تاخیر**

1. **راجع به تابع پالس داریم که:**



```
TMOD = 1 // mode timer
TH0 = 0xFD; // Start timer at 2^16-500
TL0 = 0x0C;
TR0 = 1; // Start timer0
while (TF0 != 1); // Wait for overflow ;
TR0 = 0; // Stop timer
TF0 = 0; // Clear timer overflow flag
portbit = !portbit; /// Toggle port bit
```

که همان طور که مسخص است در تابع مین با استفاده از تایمر ها وست کردن آن ها در زمان بالا و پایین و مدت زمانی که طول میکشد تا تایمر اجرا شود بیت portbit را عوض میکنیم اگر چه که این بار بیش از 10 بار در زمان اجرا می شود اما چون مشکلی از نظر کوره نیست شکلی نخواهیم داشت.

همچنین دقت کنید که از قسمت سخت افزاری در کد صرف نظر شده است چراکه پین های مان پر شده بود و به خطای پروتئوس برخورد می کردیم.

```
    void save2RAM(char Value)
{
    if (ramLocation >= 0x2000)
        ramLocation = 0x1000;    //reset the RAM location

    XBYTE[ramLocation] = Value;    //save the Temperature into External RAM
    ramLocation = ramLocation + 1;    //Increase the RAM location
}
```

تابع سیو داده ها در رام به این شکل استفاده می شود که با دستور xbyte که برای ذخیره در رم خارجی است استفاده می شود دقت کنید که چون رام ما از 1000 شروع شده است اگر راممان پر شود آن را از اول شروع به پر کردن میکنیم.

```
char adc()
{
    char TempValue;
    //Applying appropirate settings for ADC to start and convert the analog signal
    WR_ADC = 0;
    RD_ADC = 1;
    WR_ADC = 1;
    while (INTR_ADC == 1);      //Wait till conversion is complete
    RD_ADC = 0;                 //Read from ADC

    set8255();         //Apply the 8255 settings

    TempValue = PortA;          //Read the Temperature value
    //TempValue = P0;

    return TempValue;
}
```

در این قسمت نیز ابتدا با دستورات 0804 حالاتی که آماده دریافت باشد را استفاده میکنیم سپس داده های بدست آمده را از این طریق به 8255 و بعد آن به 8051 انتقال می دهیم.

```
void serial_init()
{
    TMOD = 0x20; //use timer1

    PCON = 0x80; //SCON = 1; Baud Rate Doubler
    TH1 = 0xFD;  //baud-rate = 19200;

    SCON = 0x50;
    TR1 = 1;
}

void serial_transmit(char val)
{
    /*
    This function Sends a char (val) serially from System 1 by applying appropirate setting of TxD
    */
    SBUF = val;
    while (TI == 0);
    TI = 0;
}
```

این تابع برای تولید شرایط ارسال به شکل سریال استفاده می شود.البته دقت کنید که چون در باد ریت 19200 بایدSCON یک باشد باید PCON را مقدار دهی کنیم چرا که scon بیت بالای pcon می باشد.علت این کار عدم توانایی در مقدار دهی به طور یکسان است.

این تابع به طور کامل در جلسات آزمایشگاه تولید و بررسی شده است.

تابع بعدی که شکل آن در زیر آمده است به این شکل است که ما میخواهیم ابتدا در 8255 توسط A/D مقدار را دریافت کرده و سپس آن را ارسال کنیم برای همین تنظیمات 8255 را به گونه ای تعریف میکنیم که ابتدا دریافت و سپس ارسال کنیم.

```
void Set8255 ()
{

    A0 = 1;  A1 = 1;
    RD_SYS1 = 1;
    WR_SYS1 = 0;
    CWR = 0x9B; //Port A as input Port from ADC

    /*A0 = 0;
    A1 = 0;
    */
    delay(30);
    //So we have Temperature in Port A of 8255


    /*
    A0 = 1;
    A1 = 1;
    RD_SYS1 = 1;
    WR_SYS1 = 0;
    CWR = 0x8B;  //Port A as output Port to Data Bus
    */

    //Set the Value of Port A on Data Bus
    A0 = 0;  A1 = 0;
    RD_SYS1 = 0;
    WR_SYS1 = 1;
    delay(30);
    //So we have Port A on Data Bus

}
```

تابع آخر نیز تابع دیلی می باشد که از بررسی آن در این قسمت به علت آشنایی کامل با آن صرف نظر کردیم.

- **سیستم 2 دارای توابع زیر می باشد**
  1. **تابع دریافت اطلاعات سری**
  2. **تابع پرینت و چاپ بر روی LCD**

در این قسمت طبق تابع های بررسی شده سر کلاس داریم:

```
void serial_init()
{

    TMOD = 0x20; //use timer1

    PCON = 0x80;
    TH1 = 0xFD;  //baud-rate = 19200;

    SCON = 0x50;
    TR1 = 1;
}

char serial_receive()
{

    unsigned char val;
    while(RI == 0);
    val = SBUF;
    RI = 0;
    return val;
}
```

که تابع اینیشیال برای تولید شرایط دریافت و قسمت بعد با توجه به باد ریت تعریف شده است که توضیحات آن مشابه توضیحات سیستم یک می باشد.

```
void save2RAM(char val)
{
    if (ramLocation >= 100)
        ramLocation = 0;

    TemperatureVal[ramLocation] = val;          //Save the Temperature value into Interal RAM space
    ramLocation++;
}
```

تابع سیو در رم داخلی که به سادگی تعریف شده است فقط برای اینکه اورفلوی داده ای نداشته باشیم از 100 به بعد آن را صفر کردیم که مجددا از اول شروع به پرکردن بکند.

تابع های lcd نیز به دو مدل تعریف می شود یکی برای استفاده دستوراتی که به ال سی دی میدهیم چرا که ذات چاپ دستورات و ذات چاپ عبارات با یک دیگر فرق دارد.

```
void Lcd_CmdWrite(char Command_Val)
{
   PortA = Command_Val;     //Command on Port A

   //Command Control Signals

   //Port B = 0100B  = 0x04 => Enable: 1, RW:0, RS:0  (Command Mode)
   PortB  = 0x04;
   delay(100);

   //Port B = 0000B  = 0x00 => Enable: 0, RW:0, RS:0  (Command Mode)
   PortB  = 0x00;
   delay(100);
}

void Lcd_DataWrite(char Data_Val)
{

   PortA = Data_Val;

   //Data Control Signals

   //Port B = 0101B  = 0x05 => Enable: 1, RW:0, RS:1  (Data Mode)
   PortB  = 0x05;
   delay(100);

   //Port B = 0001B  = 0x01 => Enable: 0, RW:0, RS:1  (Data Mode)
   PortB  = 0x01;
   delay(100);
}
```

در پایان نیز با توجه به شرایطمان از توابع گفته شده استفاده می کنیم:

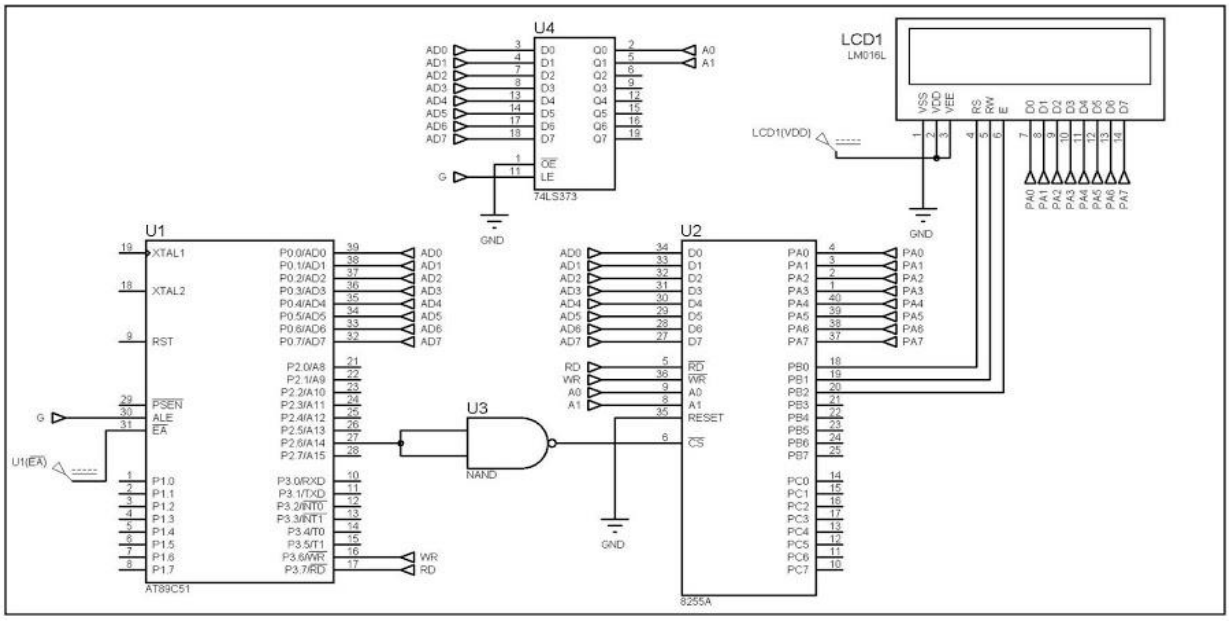
```c
void main()
{
    char START_phrase[] = "START:";  //Start phrase to be sent
    int i = 0;
    char j,a[]={"The temputure is :"};
    serial_init();            //Initialize the Serial communicatin
    while(1)
    {
        Temp = serial_receive();             //Receive Temperature from System1
        save2RAM(Temp);             //Save the Temperature value into Internal RAM

        Lcd_CmdWrite(0x38);   //Initialize LCD IN 2x16 MATRIX
        Lcd_CmdWrite(0x0e);   //Display ON, Cursor ON
        Lcd_CmdWrite(0x01);   //Clear LCD
        Lcd_CmdWrite(0x80);   //Sets line & Position

        //Display "START:" on LCD
        for (i = 0; i < 6; i++)
        {
        Lcd_DataWrite(START_phrase[i]);
        }
        Lcd_CmdWrite(0xC0);       //Sets the second line
        for(j=0;a[i]!=0;j++)
        {
        Lcd_DataWrite(a[i]);
        }
        Lcd_DataWrite(Temp);   //Display Temperature Value on LCD
    }
}
```

- **مدار ها و برنامه های مورد استفاده در شبیه سازی با توضیحات**

مدارات و اتصالات در قسمت شماتیک سخت افزار توضیح داده شدند اما برای بار دیگر شکل آن ها در قسمت زیر آمده است.



که همان طور که مشخص است داده های ما از طریق پورت صفر به لچ و سپس به 8255 انتقال پیدا میکند.

تنظیمات نیز به گونه ای می باشد که از پورت B به lcd انتقال داده شود و سپس نمایش داده شوند.

در سیستم یک نیز به همین شکل می باشد ولی ابتدا باید از ظریق a/dc سیگنال به دیجیتال ترجمه شود و سپس از 8255 به 8051 به طور مشابه انتقال پیدا میکند.

که توضیحات آن در قسمت های قبلی داده شد.

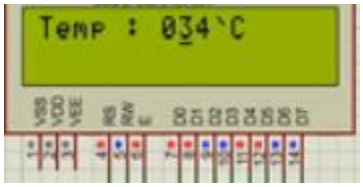
- **نتایج شبیه سازی سخت افزار و نرم افزار**

در قسمت نرم افزار مشکلی نداشتیم و توسط keil فایل ها ترجمه و تدوین شده اند:

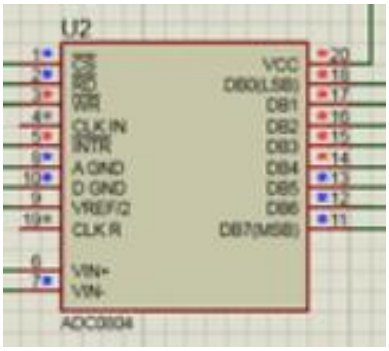
فایل ها به شکل کامل توسط نرم افزار کامپایل شده و دو سیستم ما در دو پروژه مختلف تعریف شده اند.

در قسمت سخت افزار ابتدا به طور تکی هر سیستم را تست میکنیم:

در سیستم دو به درستی توانایی چاپ را داریم :



در سیستم یک نیز توانایی دریافت را داریم اما به دلیل ارتباطات پروتئوس و همچنین باگ های آن سیستم ما به طور کامل ران نشد.



امید است که تا زمان ارائه این مشکل حل و به شکل کامل سیستم کار کند.

- **جمع بندی و برآورد نتیجه بر حسب خواسته ها:**

هر کدام از سیستم های ما به شکل جداگانه به طور صحیح و درست کار میکنند ولی زمان اتصال به یک دیگر با ارور هایی از سمت پروتئوس مواجه می شویم که در حل آن به مشکل خورده ایم.

علت این اتفاق همچنان برای ما غیر قابل توجیه است.

بالاخص که سیستم ما از نظر کد و ساختار نرم افزاری و همچنین ساختار سخت افزاری بدون مشکل است!

همانطور که دیدیم و جز به جز توابع و کد سخت افزار را بررسی کردیم دیدیم که به چه نحو باید نرم افزار و سخت افزار مربوطه را طراحی کرد.

متاسفانه در پروتئوس پروژه ران نشد. که علت اصلی آن با توجه به کلاس های آزمایشگاه ضعف نرم افزار پروتئوس می باشد.

منابع:

1. اسلاید های استاد
2. این لینک
3. این لینک
4. این لینک
5. این لینک
6. سایت http://what-when-how.com/