



راه اندازی و اندازه گیری دما با سنسور دمای DS18B20 به صورت تک و چند سنسور با AVR



تاریخ انتشار ۱۵ تیر، ۱۴۰۰ توسط آرش فتاحی

با عرض سلام خدمت همراهان سایت میکروالکام. در این مطلب قصد داریم ابتدا یک سنسور دمای DS18B20 را به میکروکنترلر Atmega32 متصل کرده و سپس با کامپایلر کدویژن برنامه مربوط به آن را جهت قرائت دما و نمایش آن بر روی LCD کاراکتری انجام دهیم. در مرحله بعد به جای 1 عدد سنسور دما، 8 عدد از آن را به میکرو متصل کرده و برنامه‌ای خواهیم نوشت که با نگه داشتن یک کلید که به میکرو متصل است، به ترتیب دمای سنسور 1 تا 8 بر روی



LCD کاراکتری نمایش داده شود. پس با من تا انتهای مطلب همراه باشید. همچنین میتونید سایر مطالب من رو از [این قسمت](#) مطالعه کنید.

معرفی سنسور دما DS18B20

سنسور DS18B20، یک سنسور دمای دیجیتال دقیق و ارزان قیمت بوده که می‌توان از آن برای سنجش دمای محیط‌های گوناگون استفاده نمود. این سنسور دقتی 12 بیتی و حدود 0.0625 درجه سانتی گراد داشته و براساس دیتاشیت ارائه شده توسط شرکت سازنده، می‌تواند دمای بین 55- تا 125+ درجه سانتی‌گراد را اندازه‌گیری کند.



سنسور دما DS18B20

برای اتصال DS18B20 به میکروکنترلر، از پروتکل یک سیمه (One Wire) استفاده می‌شود. از آنجایی که هر کدام از سنسورهای DS18B20 آدرس اختصاصی خود را دارند، می‌توان تا 8 عدد سنسور را از طریق یک سیم کنترل کرده و دمای هر کدام از این سنسورها را به صورت مجزا توسط میکروکنترلر قرائت کرد.

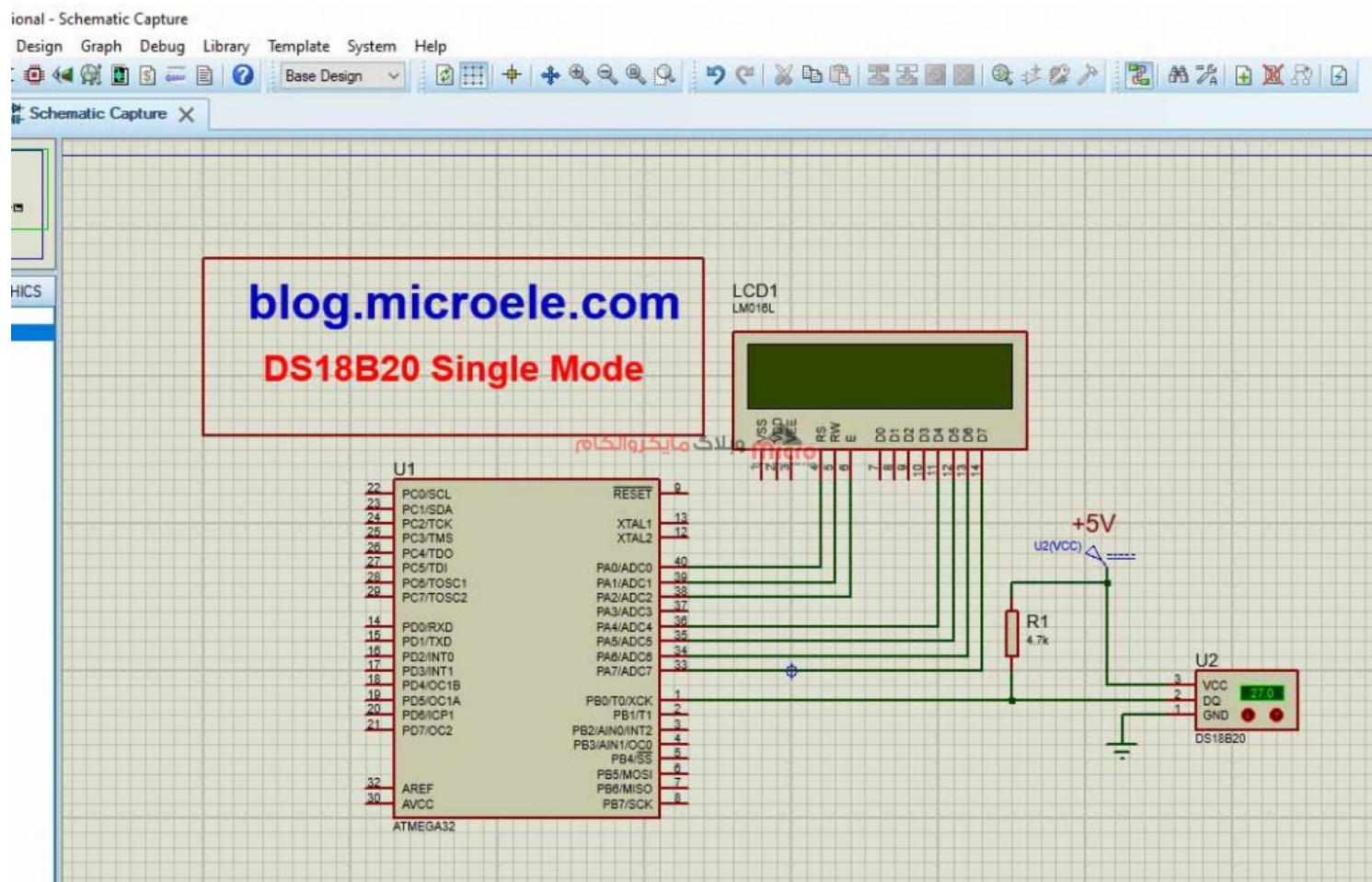


وسایل مورد نیاز

- میکروکنترلر Atmega32
- سنسور دمای DS18B20
- مقاومت 4.7 کیلوهم
- **LCD کاراکتری**
- پتانسیومتر 10 کیلوهم برای تنظیم کنتراست LCD
- **برد برد**
- **سیم برد بردی**

بخش اول: راه اندازی سنسور DS18B20 به صورت تک سنسور (Single)

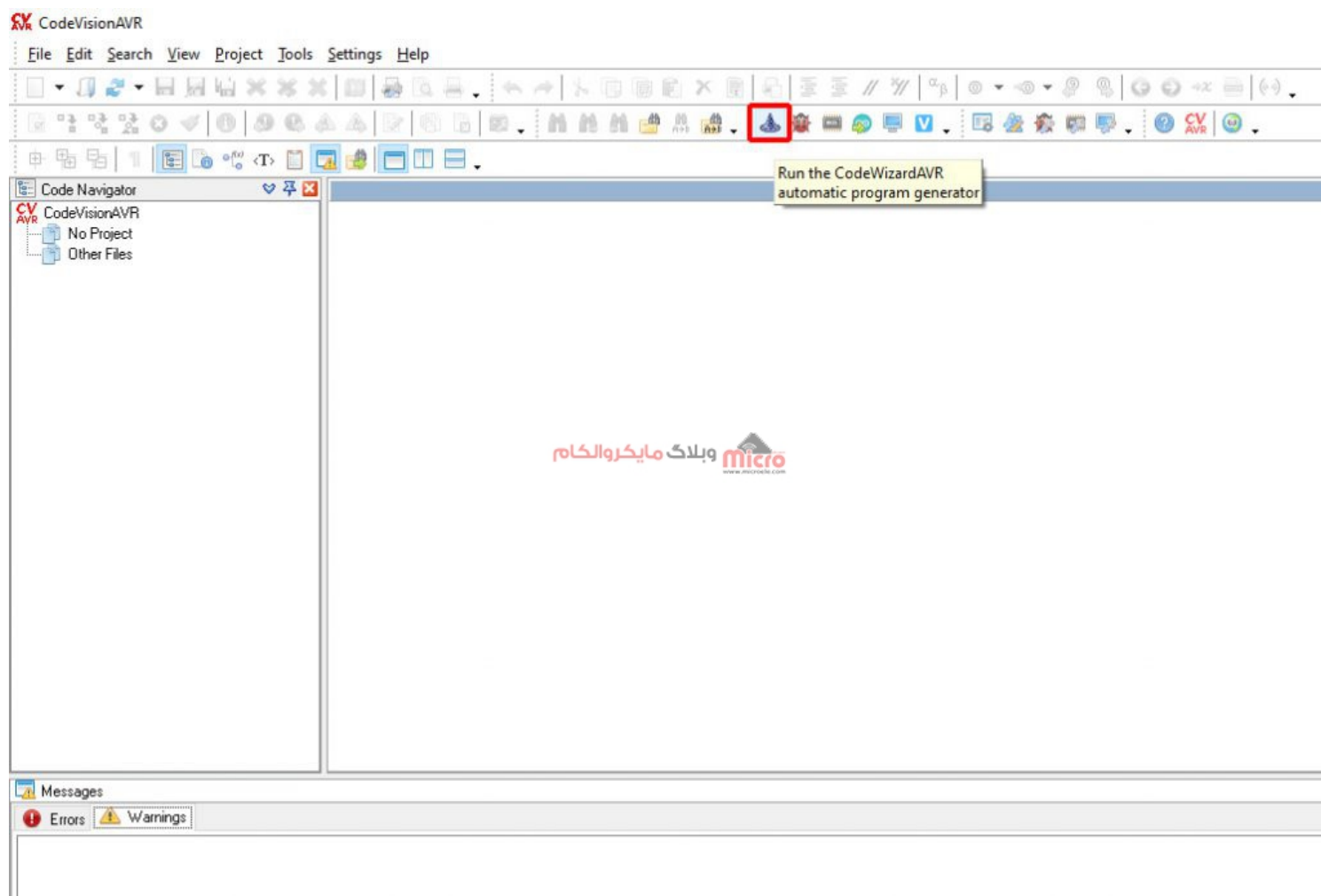
ابتدا مدار مورد نظر را مانند تصویر زیر در نرم افزار پروتئوس پیاده سازی می کنیم. برای اتصال سنسور دما، حتما باید پایه ای که قرار است برای داده های سنسور اختصاص دهیم، با یک مقاومت 4.7 کیلوهم Pull-Up گردد.



مدار بسته شده در پروتئوس برای شبیه سازی DS18B20

تظیمات CodeWizard در نرم افزار CodeVision

نرم افزار کدویژن را باز کرده و Wizard را اجرا می کنیم.



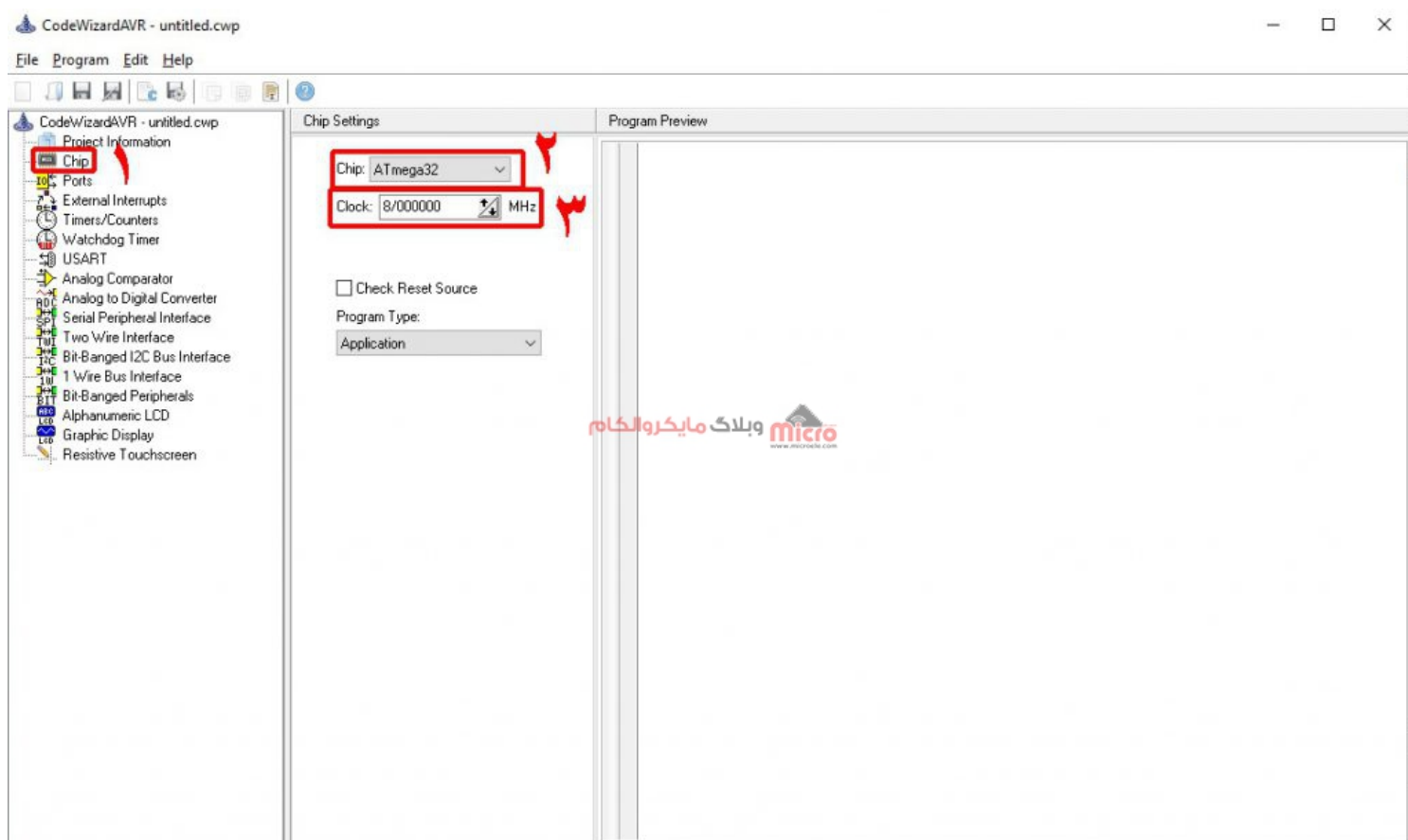
CodeWizard

در پنجره باز شده برای انتخاب Target، گزینه اول را انتخاب کرده و بر روی OK کلیک کنید تا وارد محیط Wizard شوید.



CodeWizard Target

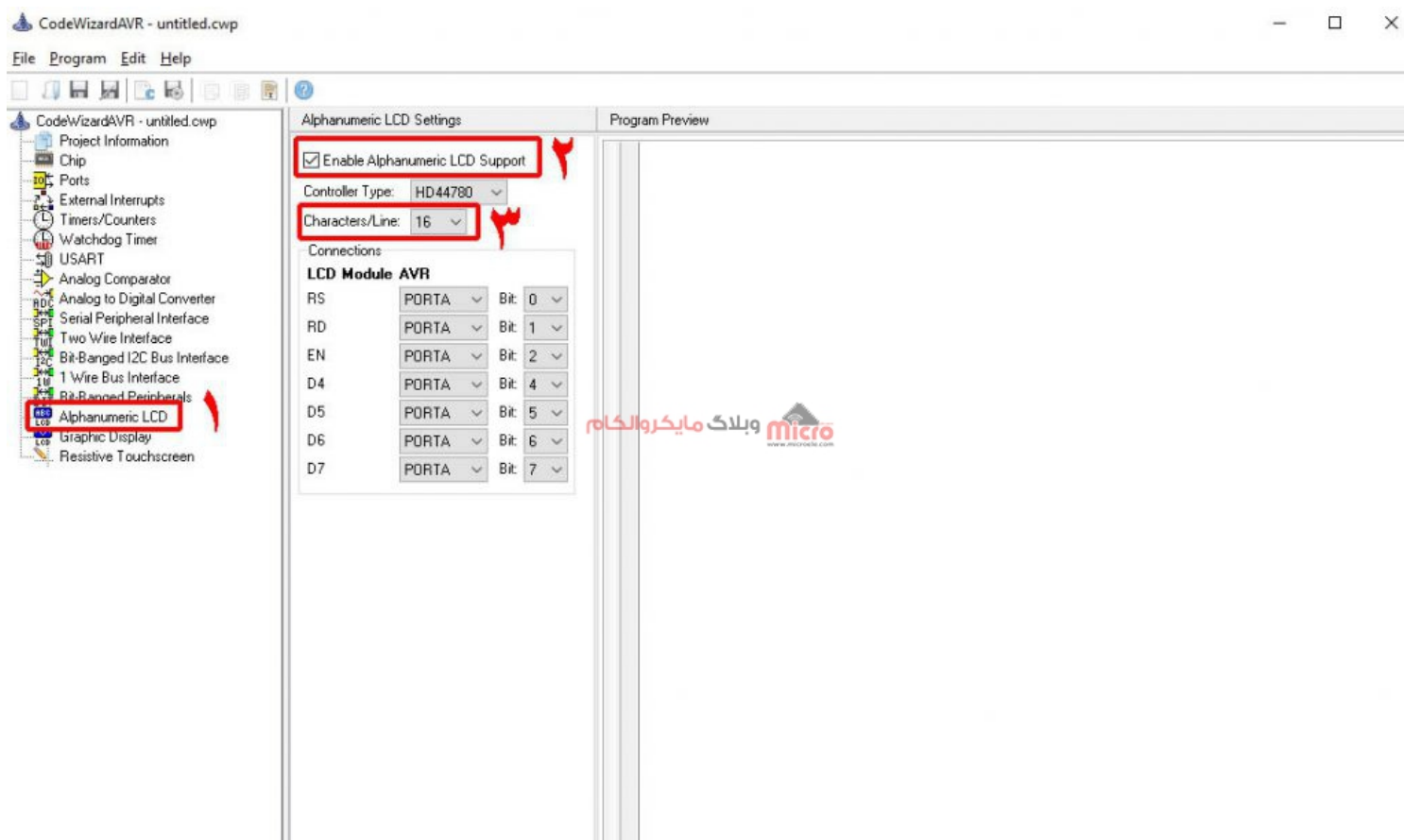
در Wizard، ابتدا چیپ مورد نظر خود، که در اینجا برای ما Atmega32 می باشد را انتخاب کرده و سپس در قسمت Clock، فرکانس کاری میکرو را تعیین می کنیم. در این پروژه می خواهیم میکرو با فرکانس داخلی 8 مگاهرتز کار کند، بنابراین Clock را بر روی 8/000000 تنظیم می کنیم. شما می توانید فرکانس های بالاتر و یا پایین تر را انتخاب کنید.



تنظیمات میکرو در CodeWizard

تنظیمات LCD کاراکتری

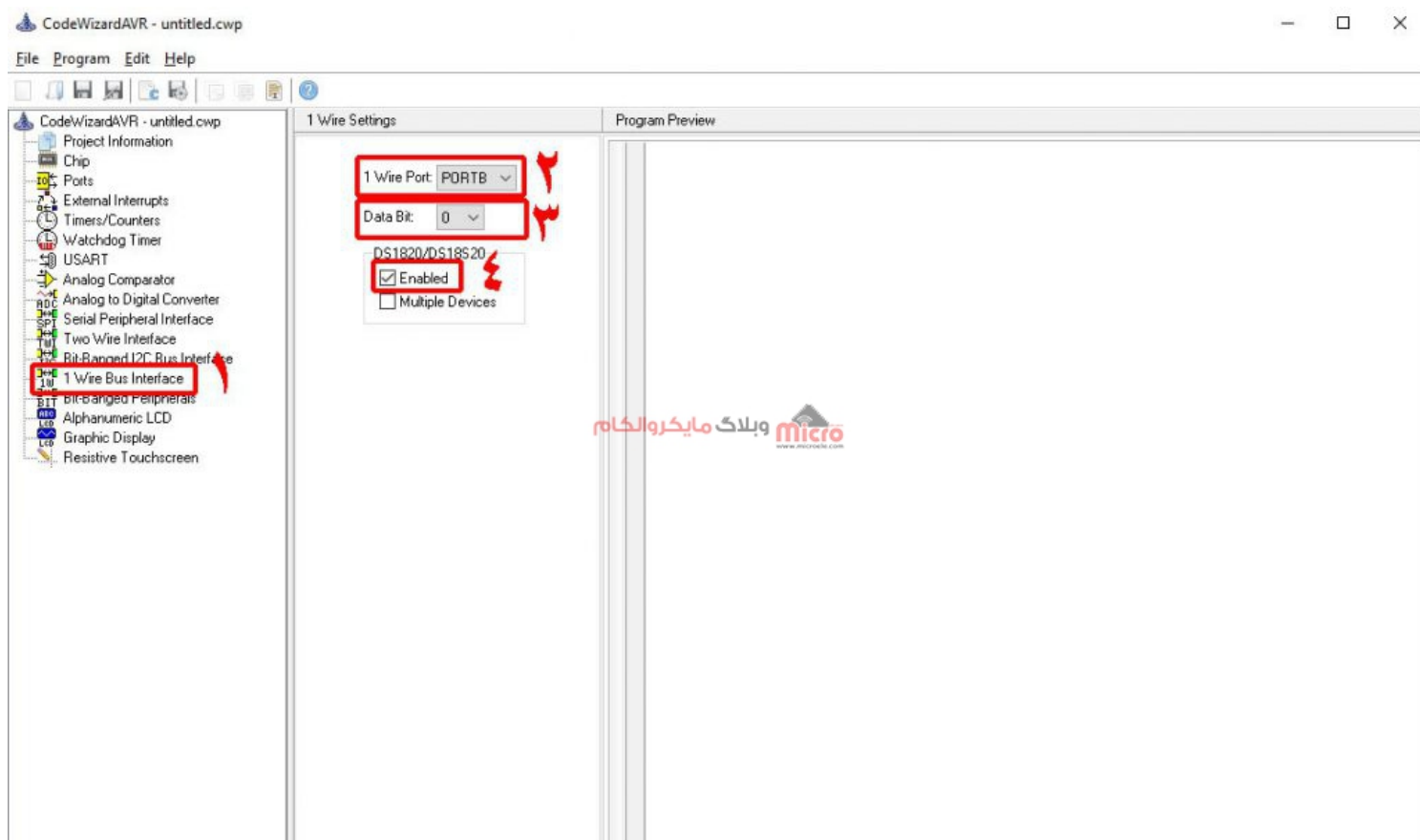
برای راه اندازی LCD، از بخش Alphanumeric LCD گزینه Enable Alphanumeric LCD Support را فعال کنید. از آنجایی که LCD استفاده شده، 16 کاراکتری می‌باشد. Characters/Line را روی 16 قرار می‌دهیم.



تنظیمات LCD در CodeWizard

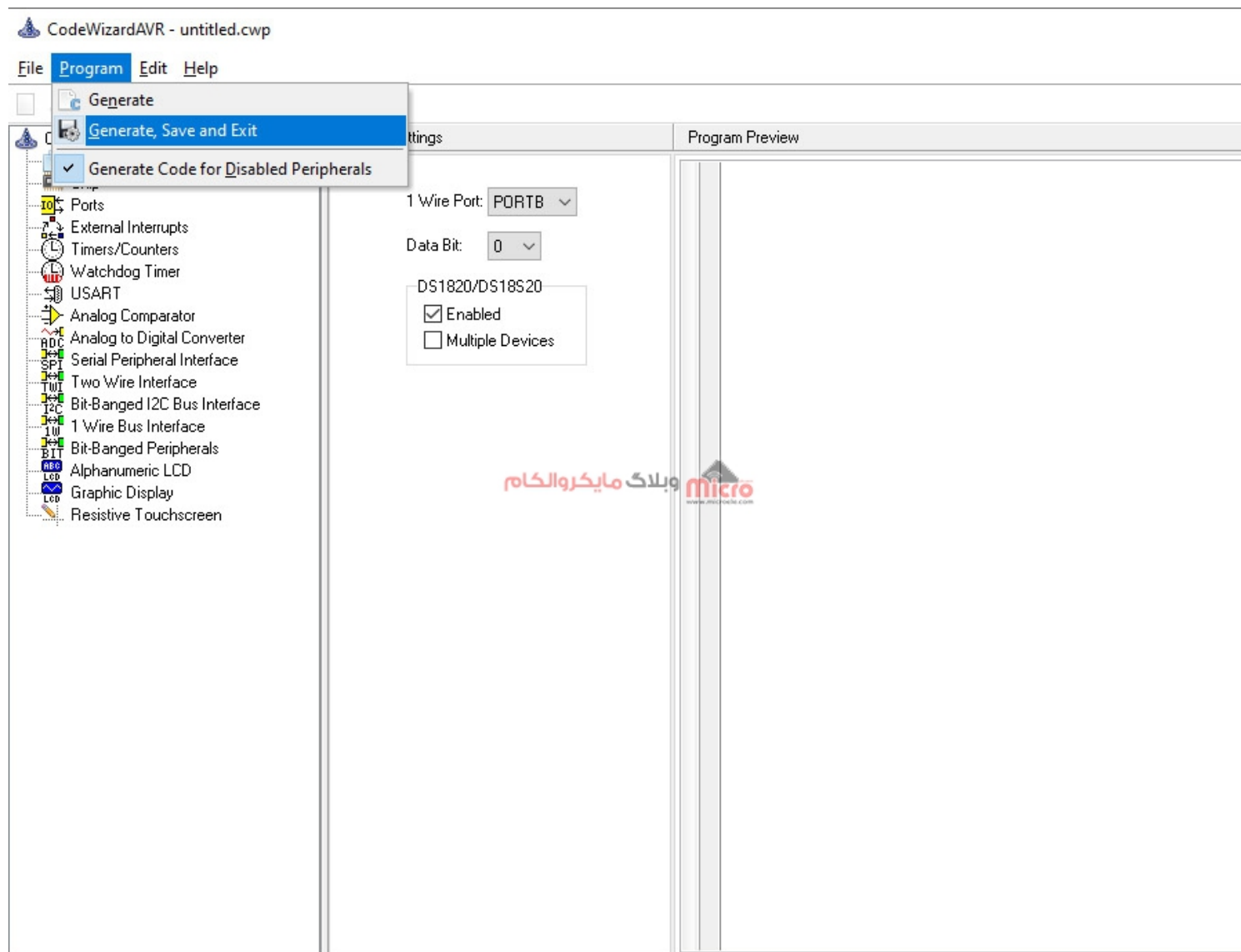
تنظیمات One Wire

بخش آخر تنظیمات مربوط به راه اندازی DS18B20 است. برای این کار به بخش 1 Wire Bus Interface رفته و از قسمت 1 Wire Port B، پورتی که سنسور به آن متصل است را انتخاب کنید. در این جا ما سنسور را به PORTB متصل کرده ایم. در قسمت Data Bit، شماره پایه ای که سنسور به میکرو متصل شده است را انتخاب کنید، که برای ما پایه صفرم پورت B می باشد. در نهایت از بخش DS1820/DS18B20 گزینه Enable را فعال کنید.



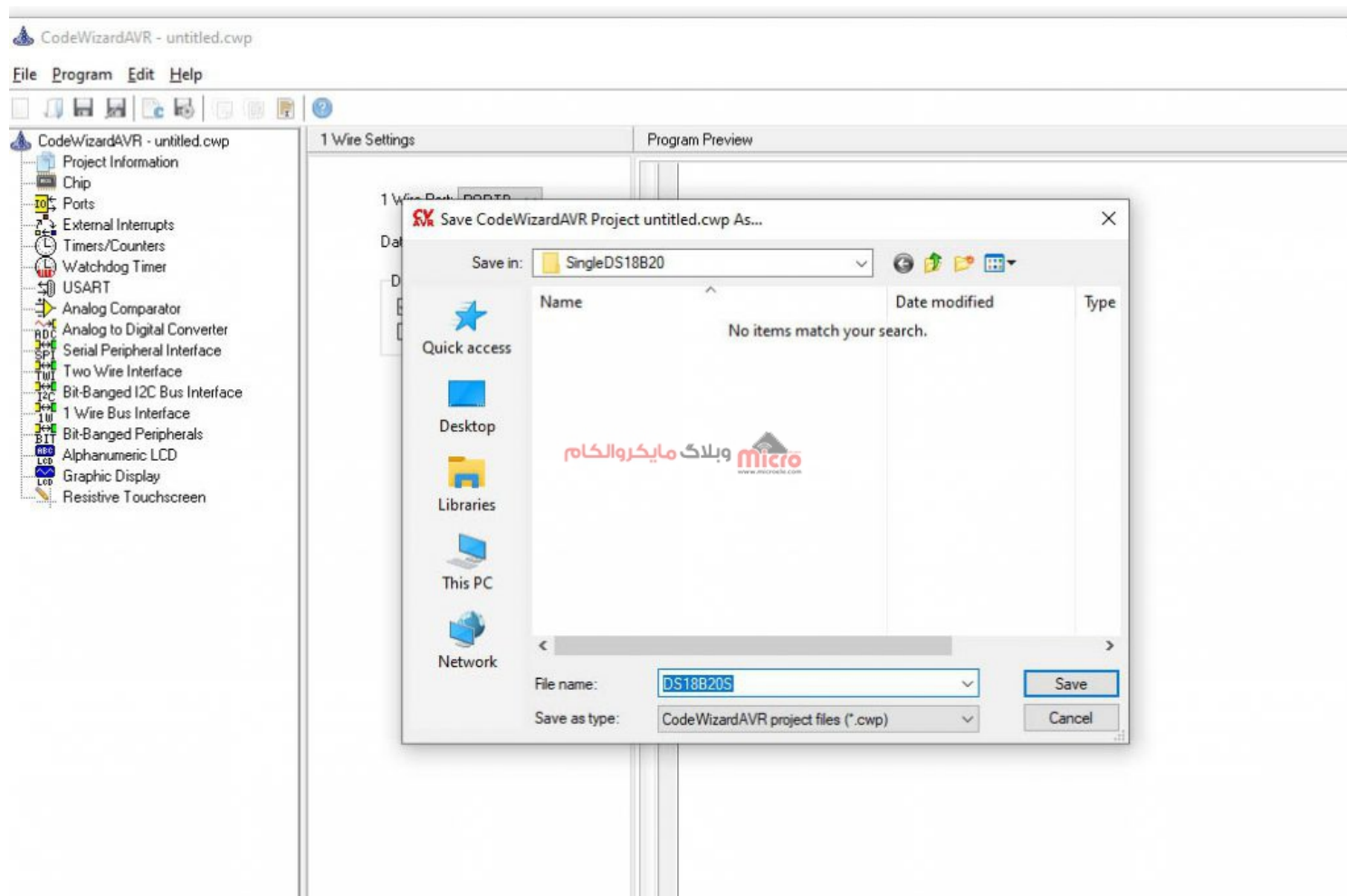
تنظیمات One Wire و DS18B20 در CodeWizard

حال نوبت به ذخیره سازی و Generate کردن پروژه می‌رسد. برای این کار از منوی Program در کدویزارد گزینه Generate, Save and Exit را انتخاب کنید.



تولید کد در CodeWizard در کدویژن

در نهایت برای ذخیره سازی فایل‌ها نام انتخاب کرده و آن‌ها را Save کنید. در این‌جا، فایل .c را به نام main ذخیره کردیم.



نخیره سازی پروژه در CodeWizard

شروع کد نویسی برای حالت تک سنسور در کدویژن

بعد از باز شدن فایل c. که در اینجا برای ما main.c می باشد، ابتدا باید نحوه فراخوانی کتابخانه سنسور دما را اصلاح کنیم. از آنجایی که ما از سنسور DS18B20 استفاده می کنیم، بنابراین ds1820.h را باید به ds18B20.h تغییر دهیم. حال برنامه را Build کرده تا فایل ds18B20.h توسط کامپایلر به پروژه اضافه گردد.

```
#include <ds18B20.h>
```



```
CodeVisionAVR - F:\microele\DS18B20CV\Project\SingleDS18B20\DS18B20S.prj
File Edit Search View Project Tools Settings Help

Code Navigator
Project: DS18B20S
  Notes
  main.c
  Headers
  List Files
    DS18B20S.asm
    DS18B20S.lst
    DS18B20S.map
  Other Files

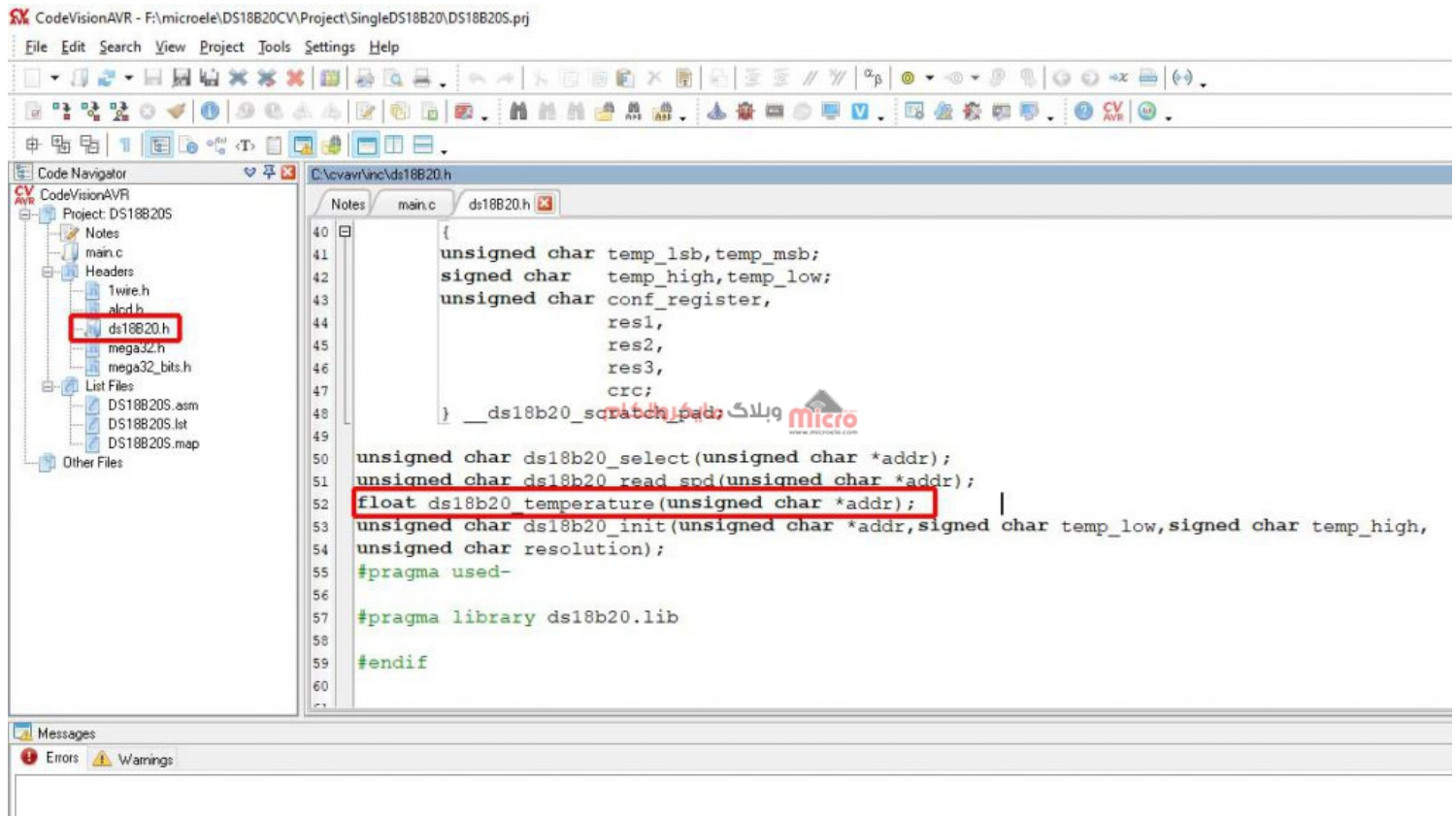
main.c
19 Memory model : Small
20 External RAM size : 0
21 Data Stack size : 512
22 *****/
23
24 #include <mega32.h>
25
26 // 1 Wire Bus interface functions
27 #include <1wire.h>
28 // DS1820 Temperature
29 #include <ds18B20.h>
30
31 // Alphanumeric LCD functions
32 #include <alcd.h>
33
34 // Declare your global variables here
35
36 void main(void)
37 {
38 // Declare your local variables here
39
```

جایگزینی کتابخانه DS18B20

بعد از Build کردن برنامه، در قسمت Code Navigator در سمت چپ، ds18B20.h توسط کامپایلر لود شده و نمایان می‌شود.

اضافه کردن کتابخانه‌ها و متغیرها

با کلیک بر روی آن می‌توانید توابع این کتابخانه را مشاهده کنید. برای خواندن مقادیر دما، تابع ds18b20_temperature که خروجی آن به صورت Float است را استفاده خواهیم کرد.



کتابخانه DS18B20

```
ds18b20_temperature();
```

مانند تصویر زیر، در فایل main.c کتابخانه‌های delay.h و stdio.h را اضافه کنید. از کتابخانه stdio جهت استفاده از تابع sprintf برای تبدیل مقادیر float به کاراکتر جهت چاپ مقادیر دما بر روی LCD استفاده خواهیم کرد.

```
#include <delay.h>
#include <stdio.h>
```

همچنین متغیرهای Temp از نوع Float جهت انتقال مقادیر خوانده شده از سنسور و آرایه رشته‌ای str جهت تبدیل مقادیر عددی به رشته برای نمایش بر روی LCD کاراکتری در برنامه تعریف می‌کنیم.



```
CodeVisionAVR - F:\microele\DS18B20CV\Project\SingleDS18B20\DS18B20S.prj
File Edit Search View Project Tools Settings Help

Code Navigator
Project: DS18B20S
  Notes
  main.c
  Headers
    1wire.h
    alcd.h
    delay.h
    ds18B20.h
    io.h
    mega32.h
    mega32_bits.h
    stdarg.h
    stdio.h
  List Files
    DS18B20S.asm
    DS18B20S.lst
    DS18B20S.map
  Other Files

F:\microele\DS18B20CV\Project\SingleDS18B20\main.c
Notes main.c * ds18B20.h
22  *****/
23
24  #include <mega32.h>
25
26  #include <delay.h>
27  #include <stdio.h>
28
29  // 1 Wire Bus interface functions
30  #include <lwire.h>
31
32  // DS1820 Temperature Sensor functions
33  #include <ds18B20.h>
34
35  // Alphanumeric LCD functions
36  #include <alcd.h>
37
38  // Declare your global variables here
39  float Temp=0;
40  char str[20];
41
42  void main(void)
43  {
44  }
```

افزودن کتابخانه‌ها در فایل c.

```
float Temp=0;
char str[20];
```

تکمیل برنامه در حلقه While در کدویژن

حال برای نوشتن ادامه برنامه به حلقه While می رویم. تابع ds18b20_temprature دارای یک ورودی آدرس است. از آنجایی که هدف ما در این بخش راه اندازی یک تک سنسور است، ورودی این تابع را مقدار 0 قرار می‌دهیم. در آخر



برای انتقال خروجی این تابع، آن را برابر با متغیر Temp که قبلاً آن را تعریف کرده بودیم قرار می‌دهیم.

```
166 lcd_init(16);
167 lcd_gotoxy(0,0);
168 lcd_putsf("www.microele.com");
169 delay_ms(500);
170 lcd_gotoxy(2,1);
171 lcd_putsf("Arash Fattahi");
172 delay_ms(2000);
173 lcd_clear();
174 while (1)
175 {
176     // Place your code here
177     Temp = ds18b20_temperature(0);
178     sprintf(str, "Temp:%4.2f", Temp);
179     lcd_gotoxy(0,0);
180     lcd_puts(str);
181     delay_ms(300);
182 }
183
184
185
```

تکمیل برنامه نوشته شده در حلقه while

```
Temp = ds18b20_temperature(0);
sprintf(str, "Temp:%4.2f", Temp);
lcd_gotoxy(0,0);
lcd_puts(str);
```



```
delay_ms(300);
```

از تابع `sprintf` جهت تبدیل مقدار `Float` به رشته استفاده کرده و مقدار متغیر `Temp` را به صورت دو رقمی با دو رقم اعشار از طریق المان `4.2f` به متغیر رشته‌ای `str` تبدیل و به آن منتقل می‌کنیم. در صورتی که می‌خواهید مقادیر سه رقمی نیز قابل چاپ بر روی `LCD` باشند، کافیسست به جای عبارت `4.2f` از `5.2f` در تابع `sprintf` استفاده کنید. در نهایت با دستور `lcd_gotoxy(0,0)` به ابتدای سطر اول `LCD` رفته و از طریق دستور `lcd_puts(str)` مقادیر دما را بر روی `LCD` چاپ می‌کنیم.

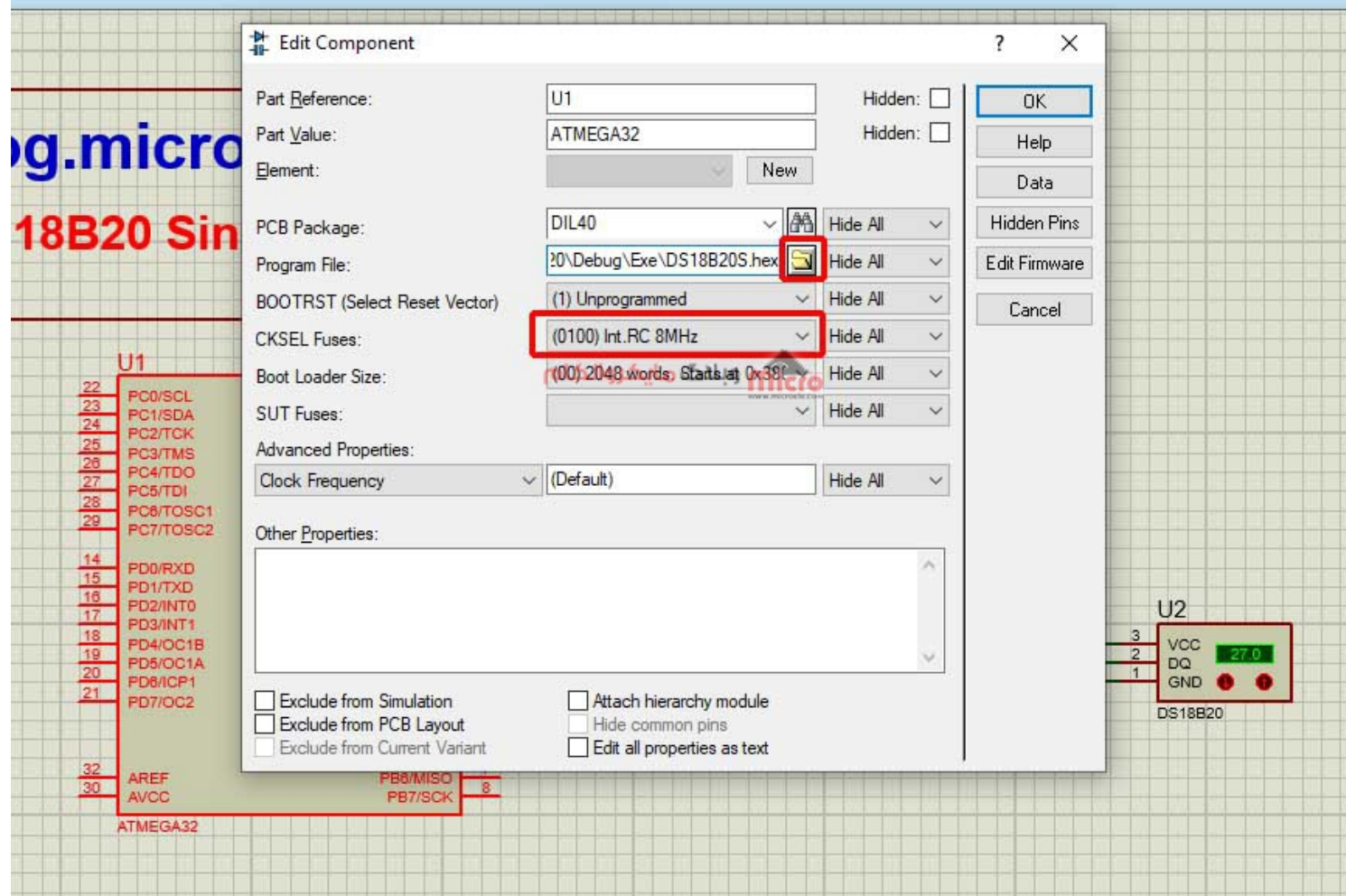
شبیه سازی حالت تک سنسور DS18B20

به محیط پروتئوس رفته و با دو بار کلیک بر روی میکرو، برنامه را به آن لود کرده و همچنین فرکانس داخلی میکرو را بر روی `8Mhz` داخلی تنظیم می‌کنیم.



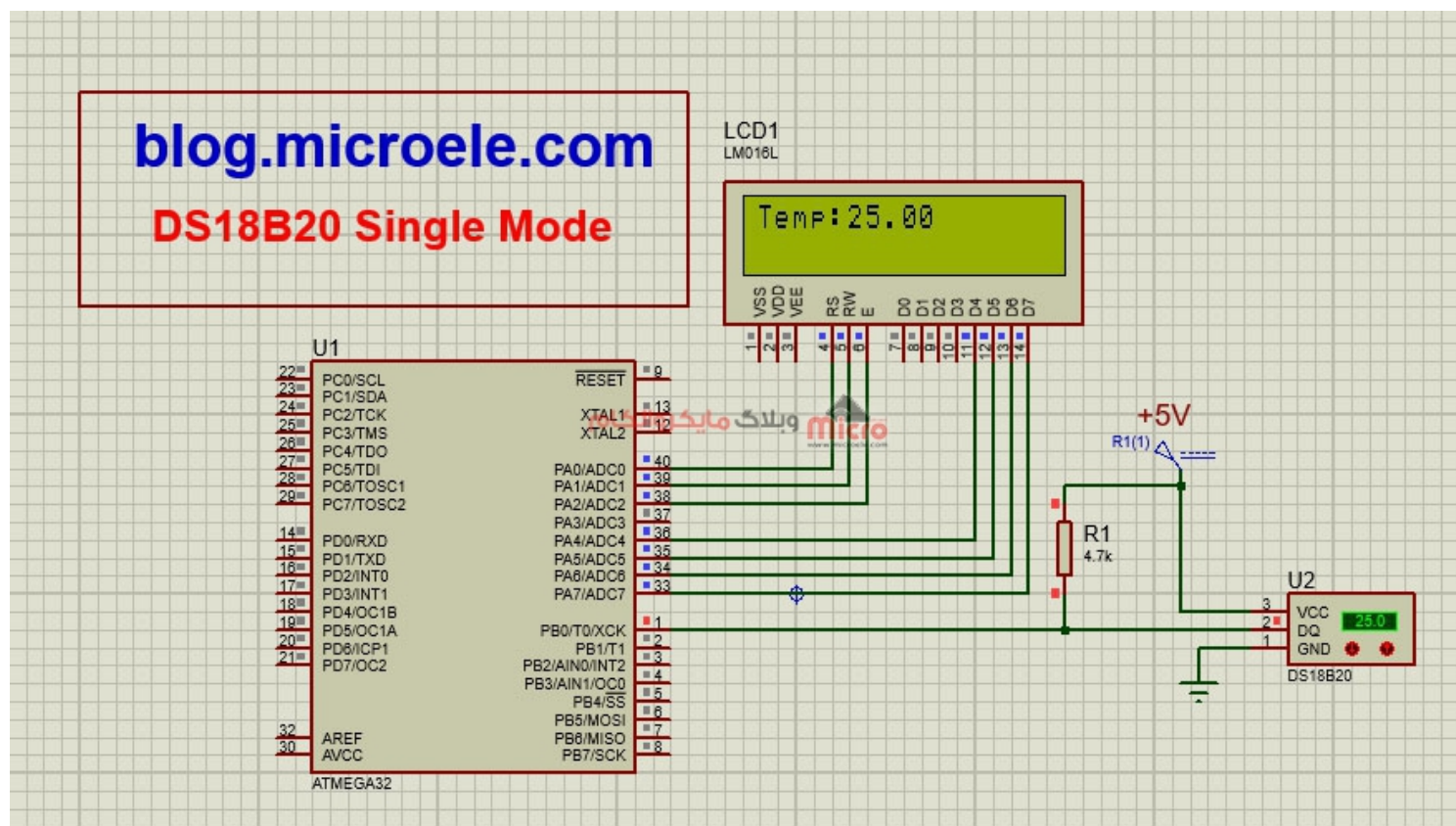
apture

Debug Library Template System Help



لود کردن برنامه به میکرو در محیط پروتئوس

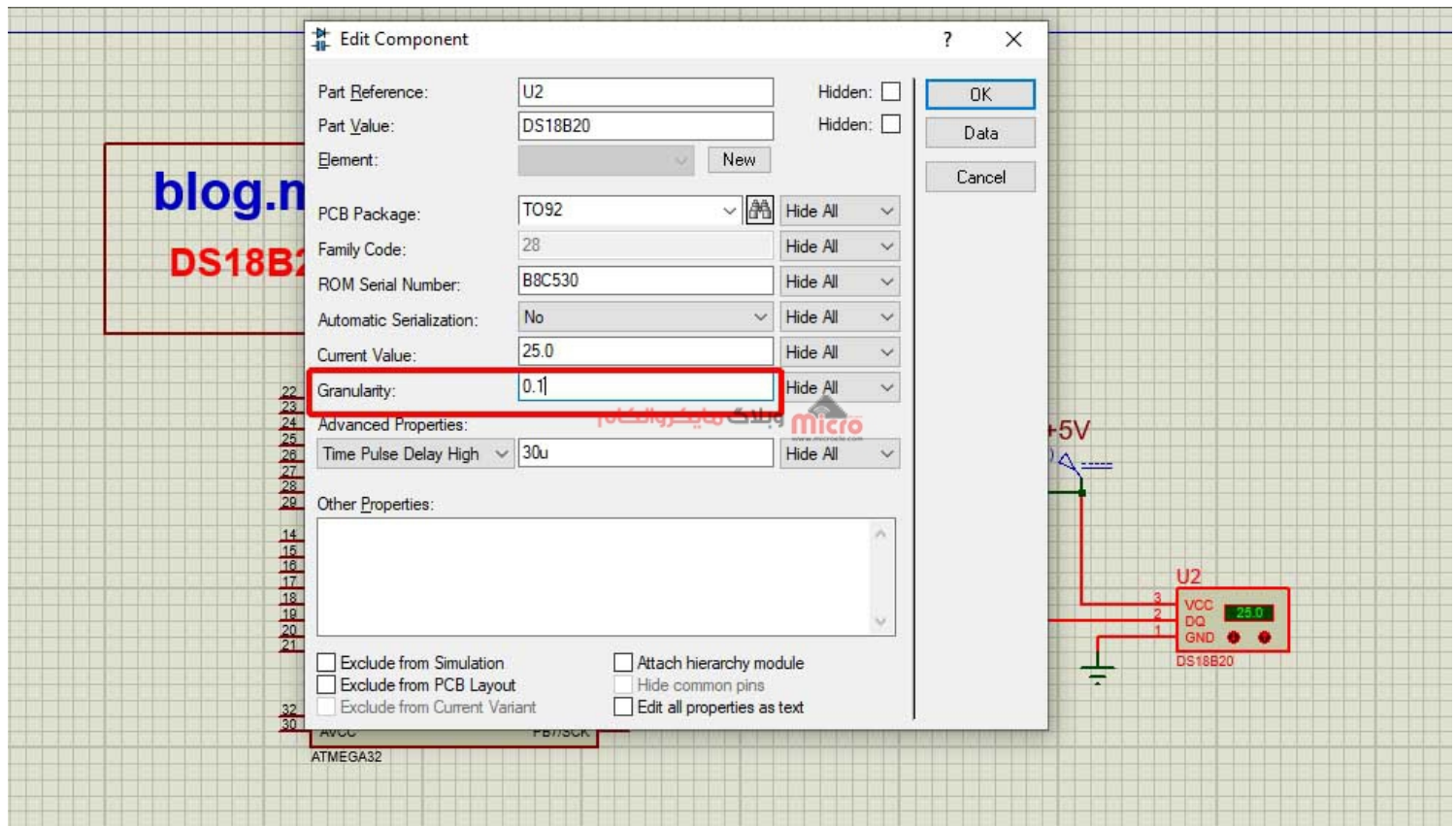
پروتئوس را Run کرده و شبیه سازی را اجرا می‌کنیم. همچنین برای تغییر دما می‌توانید از دو کلید قرمز رنگِ فلش مانند بر روی سنسور استفاده کنید.



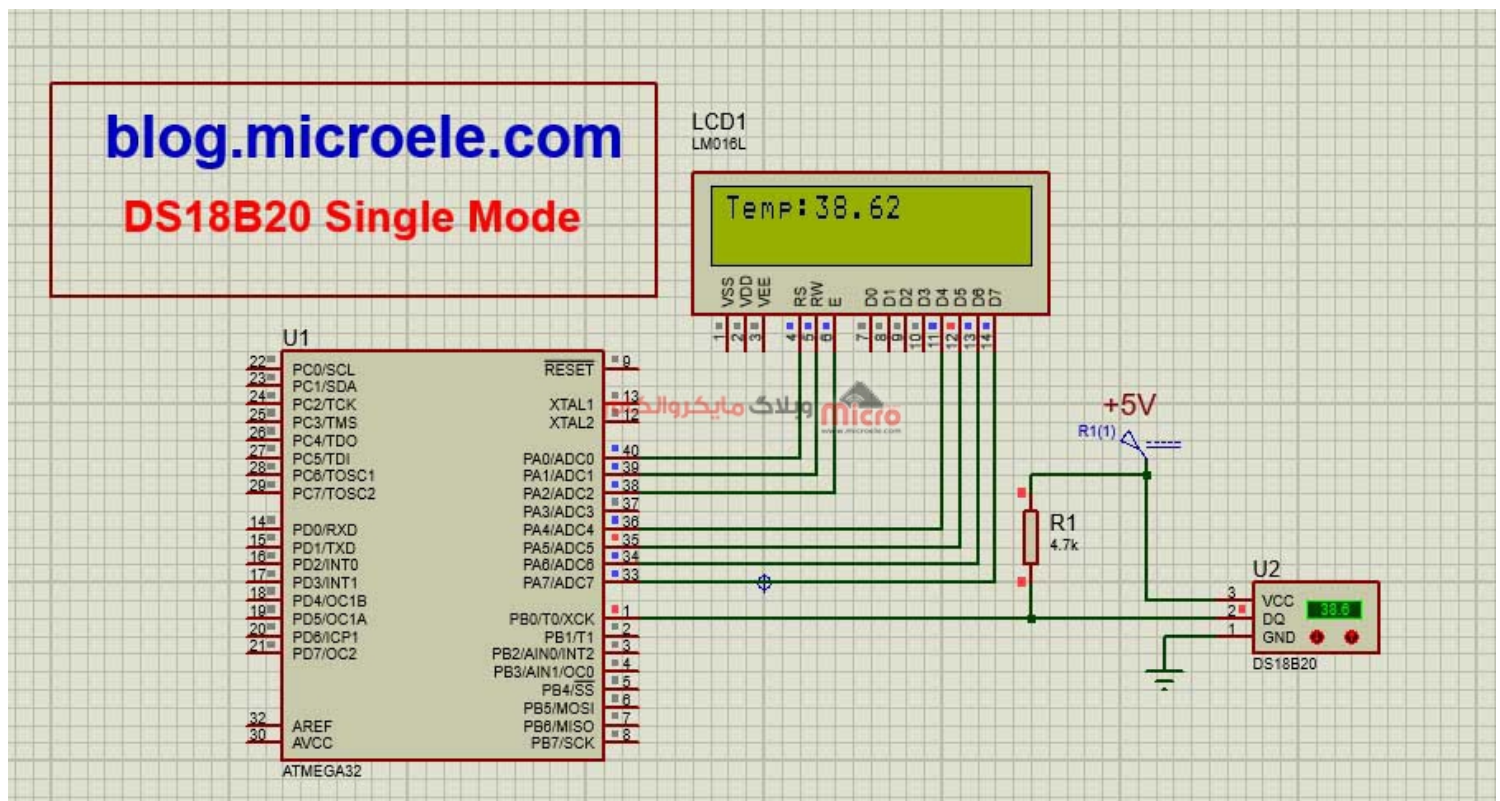
شیبه سازی Single Mode در پروتئوس

تنظیم Granularity در پروتئوس

برای بالاتر بردن دقت سنسور DS18B20 در پروتئوس و تولید مقادیر اعشاری، کافیسیت بر روی سنسور DS18B20 کلیک راست، و سپس کلیک چپ کرده تا پنجره تنظیمات سنسور باز شود. با قرار دادن مقادیر 0.1 یا 0.01 در قسمت Granularity می‌توان دقت اعمال دما بر روی سنسور را افزایش داد.



تغییر Granularity سنسور DS18B20 در پروتئوس

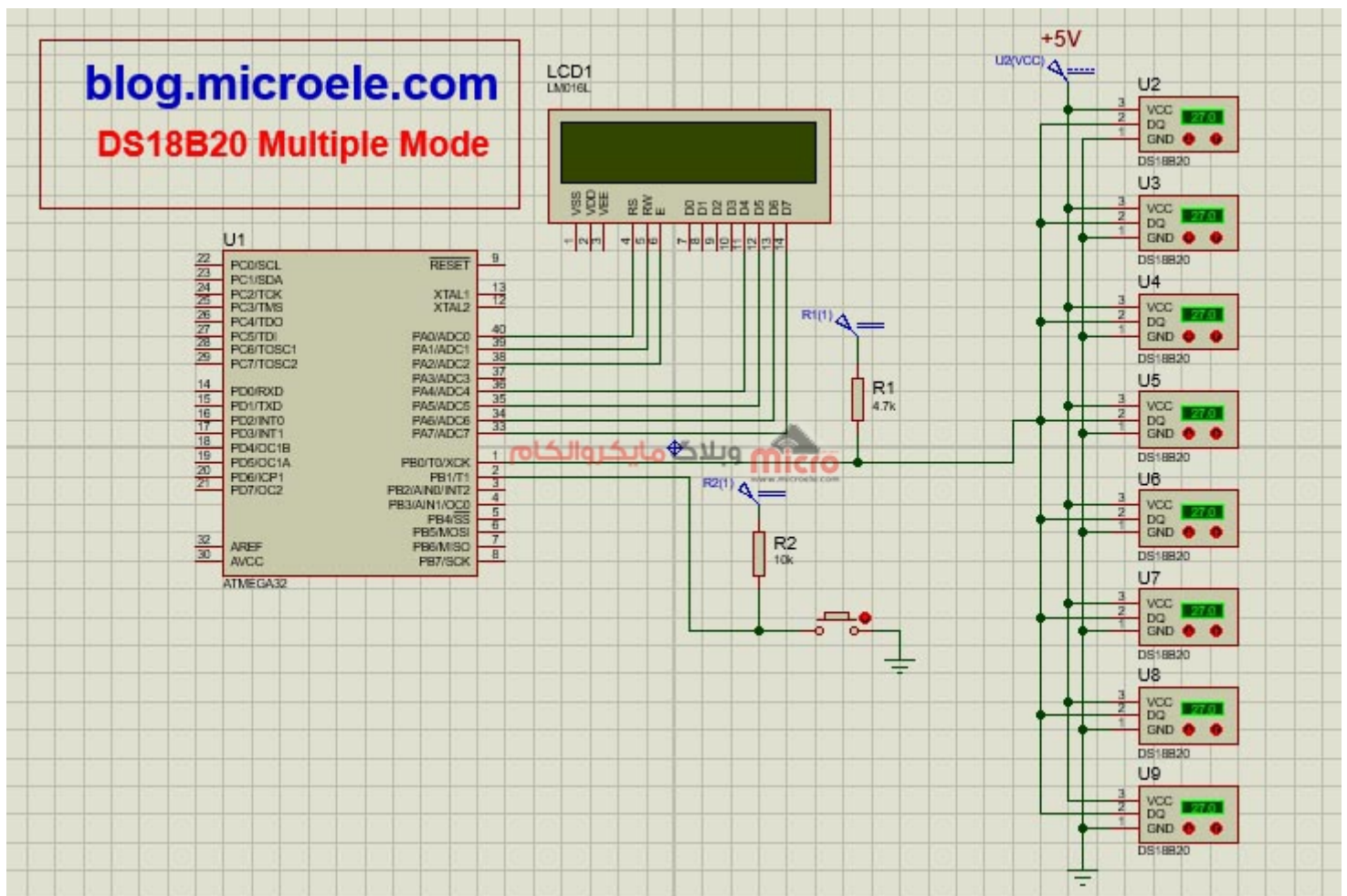


شبیه سازی DS18B20 در پروتئوس

بخش دوم: راه اندازی سنسور DS18B20 به صورت چند سنسور (Multiple)

در این بخش به راه اندازی تعداد بیشتری سنسور دمای DS18B20 به صورت هم زمان بر روی یک پین میکرو خواهیم پرداخت.

مطابق تصویر زیر مدار را در نرم افزار پروتئوس می‌بندیم:



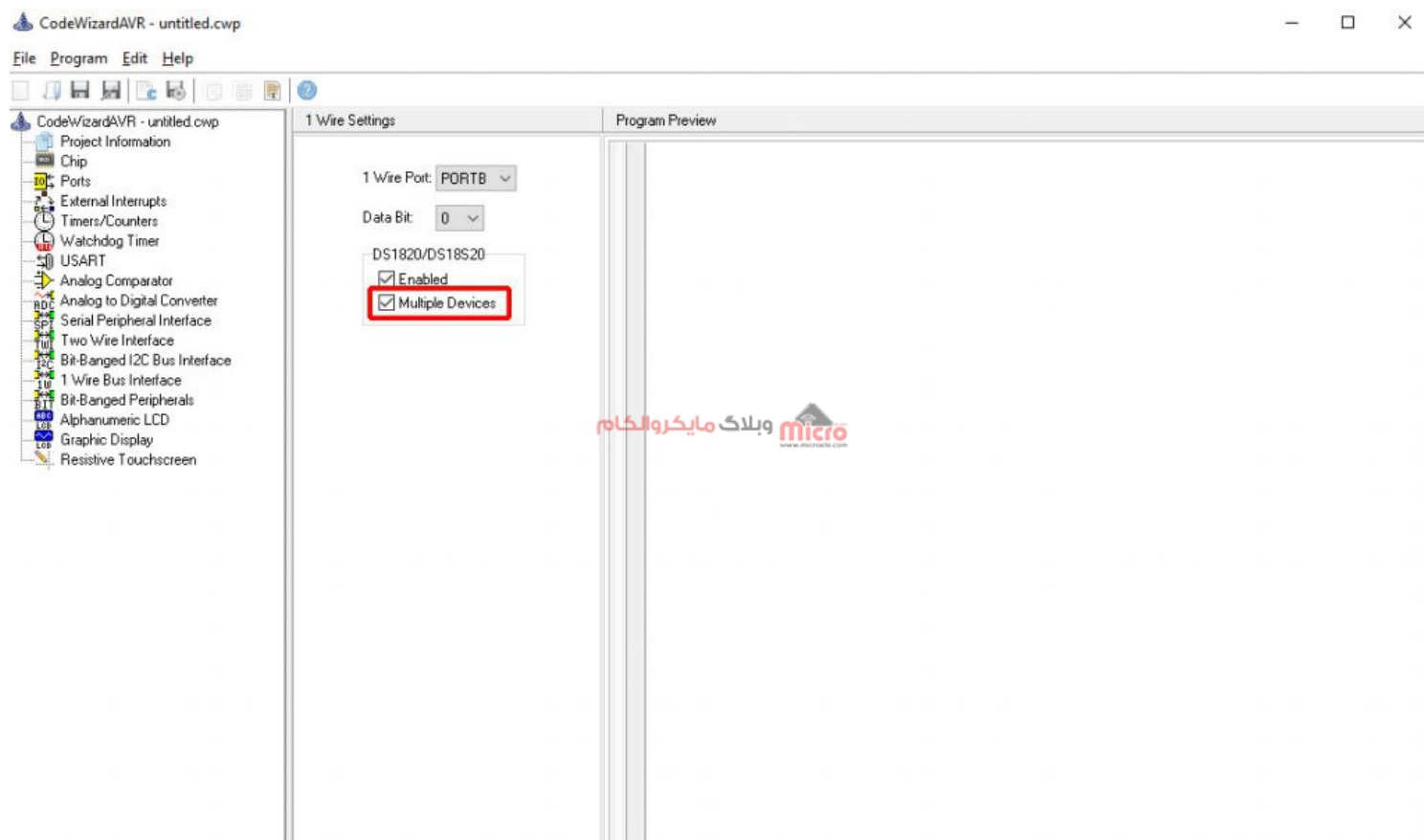
بستن مدار DS18B20 به صورت چند سنسور

در این مدار، یک کلید به پایه PB1 میکرو اضافه می‌کنیم که با نگه داشتن آن، نمایش دمای مرتبط با هر سنسور بر روی LCD به صورت چرخشی عوض شود.

اعمال تنظیمات Code Wizard برای حالت چند سنسور

مطابق بخش قبل، پیکربندی‌های مربوط به انتخاب میکرو و LCD را در کدویزارد نرم افزار کدویژن انجام داده و به بخش مربوط به تنظیمات One Wire می‌رویم.

تنظیمات را مانند دفعه قبل انجام داده با این تفاوت که این بار گزینه Multiple Devices را نیز فعال می‌کنیم. در آخر پروژه را Generate کرده و Save کنید.



فعال کردن گزینه Multiple Devices در CodeWizard

شروع کد نویسی برای حالت چند سنسور DS18B20 در کدویژن

مانند تصویر زیر مجدداً در فایل main.c، کتابخانه‌ها را اضافه کنید. همچنین حرف B را در کتابخانه ds1820 اضافه کنید. به صورت پیش فرض یک تعریف با عنوان MAX_DS1820 توسط کدویژن ایجاد شده است که ماکزیمم تعداد سنسورهای DS18B20، که می‌توانیم به میکرو متصل کنیم را نشان می‌دهد. در حال حاضر این تعریف برای 8 عدد سنسور تنظیم شده است. اگر تعداد سنسورهایی که به میکرو متصل کرده‌اید کمتر است، می‌توانید عدد متناظر با تعداد سنسورهای مدنظر خود را به جای 8 قرار دهید. در قسمت بعدی کد، یک متغیر به نام ds1820_devices به صورت پیش فرض تعریف شده است که آدرس سنسورهای DS18B20 در آن قرار می‌گیرد.



```
22  *****/
23
24  #include <mega32.h>
25
26  #include <delay.h>
27  #include <stdio.h>
28
29  // 1 Wire Bus interface functions
30  #include <1wire.h>
31
32  // DS1820 Temperature Sensor function
33  #include <ds18B20.h>
34
35  // maximum number of DS1820 devices
36  // connected to the 1 Wire bus
37  #define MAX_DS1820 8
38  // number of DS1820 devices
39  // connected to the 1 Wire bus
40  unsigned char ds1820_devices;
41  // DS1820 devices ROM code storage area,
42  // 9 bytes are used for each device
43  // see the 1-wire search function description in the 1-wire
```

اضافه کردن کتابخانه به فایل main.c

آرایه ds1820_rom_codes

کمی پایین‌تر، یک آرایه به نام `ds1820_rom_codes[MAX_DS1820][9]` به صورت پیش فرض توسط کدویژن در کد قرار داده شده است که تمامی آدرس‌های مربوط به سنسورها، در این آرایه ذخیره می‌گردد. در تصویر زیر، یک آرایه به نام Temp با 8 خانه جهت ذخیره سازی مقادیر دمای سنسورها تعریف کردیم. همچنین یک متغیر i به عنوان شمارنده جهت گزینش سنسورها هنگام فشردن کلید تعریف شده است.



```
40 unsigned char ds1820_devices;
41 // DS1820 devices ROM code storage area,
42 // 9 bytes are used for each device
43 // (see the w1_search function description in the help)
44 unsigned char ds1820_rom_codes[MAX_DS1820][9];
45
46 // Alphanumeric LCD functions
47 #include <alcd.h>
48
49 // Declare your global variables here
50 float Temp[8]={0,0,0,0,0,0,0,0};
51 char str[20];
52 unsigned char i=0;
53
54 void main(void)
55 {
56 // Declare your local variables here
57
58 // Input/Output Ports initialization
59 // Port A initialization
60 // Function: Bit7=In Bit6=In Bit5=In Bit4=In Bit3=In Bit2=In Bit1=In Bit0=In
```

تعریف متغیرها در فایل main

```
float Temp[8]={0,0,0,0,0,0,0,0};
char str[20];
unsigned char i=0;
```

نحوه جست و جوی آدرس‌های سنسورهای DS18B20 توسط کدویژن

از طریق دستور (ds1820_devices=w1_search(0xf0,ds1820_rom_codes) به صورت پیش فرض توسط کدویژن در فایل



main.c ایجاد شده است، عملیات جست و جوی آدرس‌های سنسورهای DS18B20 متصل شده به میکرو انجام شده و در نهایت این آدرس‌ها در آرایه ds1820_rom_codes ذخیره می‌گردند.

```
CodeVisionAVR - F:\microele\DS18B20CV\Project\MultipleDS18B20\mds.prj
File Edit Search View Project Tools Settings Help

Code Navigator
CodeVisionAVR
Project: mds
  Notes
  main.c
  Headers
    1wire.h
    alcd.h
    delay.h
    ds18B20.h
    io.h
    mega32.h
    mega32_bits.h
    stdarg.h
    stdio.h
  List Files
    mds.asm
    mds.lst
    mds.map
  Other Files

Notes main.c
160 // 1 Wire Bus initialization
161 // 1 Wire Data port: PORTB
162 // 1 Wire Data bit: 0
163 // Note: 1 Wire port settings are specified in the
164 // Project/Configure/C Compiler/Libraries/1 Wire menu.
165 w1_init();
166
167 // Determine the number of DS1820 devices
168 // connected to the 1 Wire bus
169 ds1820_devices = w1_search(0xff, ds1820_rom_codes);
170
171 // Alphanumeric LCD initialization
172 // Connections are specified in the
173 // Project/Configure/C Compiler/Libraries/Alphanumeric LCD menu:
174 // RS - PORTA Bit 0
175 // RD - PORTA Bit 1
176 // EN - PORTA Bit 2
177 // D4 - PORTA Bit 4
178 // D5 - PORTA Bit 5
179 // D6 - PORTA Bit 6
180 // D7 - PORTA Bit 7
181 // Characters/lines: 16
```

جست و جوی آدرس‌ها در DS18B20

قرائت دما با سنسور دما DS18B20 در حالت Multiple

مجدداً با استفاده از تابع ds1820_temperature می‌توانیم مقادیر سنسورها را بخوانیم. با این تفاوت که برای ورودی این



تابع به جای عدد 0، آرایه ds18B20_rom_codes، که حاوی آدرس‌های سنسورها می‌باشد را قرار می‌دهیم.

```
187 lcd_putsf("Arash Fattahi");
188 delay_ms(2000);
189 lcd_clear();
190 while (1)
191 {
192     // Place your code here
193     if(PINB.1 == 0)
194     {
195         if(i<7) i++;
196         else if(i==7) i=0;
197     }
198     Temp[i] = ds18b20_temperature(ds1820_rom_codes[i]);
199     sprintf(str, "Temp%01d:%4.2f", i, Temp[i]);
200     lcd_gotoxy(0,0);
201     lcd_puts(str);
202     delay_ms(50);
203 }
204
205
206
```

تکمیل برنامه در حلقه اصلی While

```
if(PINB.1 == 0)
{
    if(i<7) i++;
    else if(i==7) i=0;
}
```

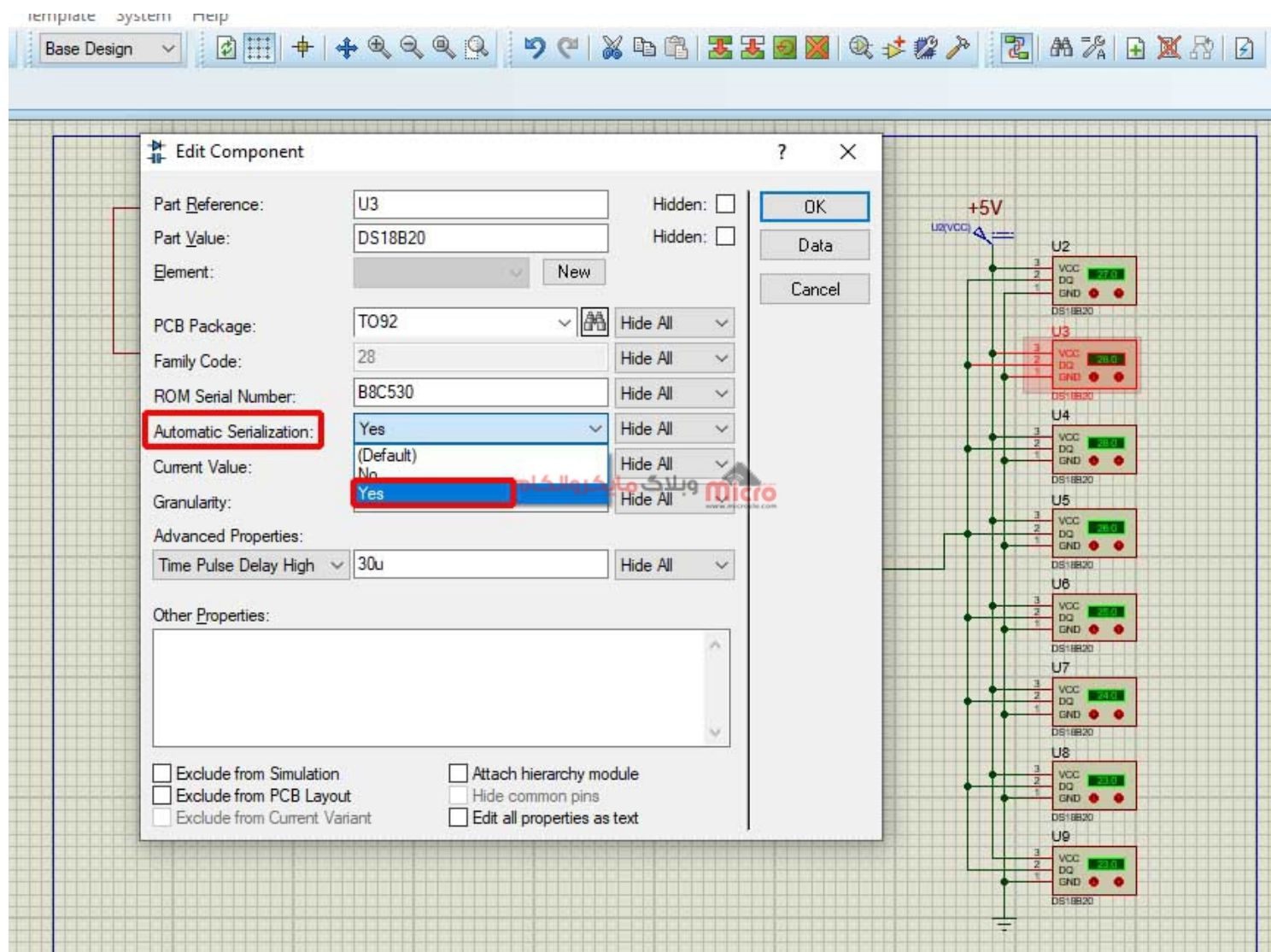


```
Temp[i] = ds18b20_temperature(ds1820_rom_codes[i]);  
sprintf(str, "Temp%01d:%4.2f", i, Temp[i]);  
lcd_gotoxy(0,0);  
lcd_puts(str);  
delay_ms(50);
```

برنامه را Build کرده و در محیط پروتئوس برنامه را بر روی میکرو لود می‌کنیم.

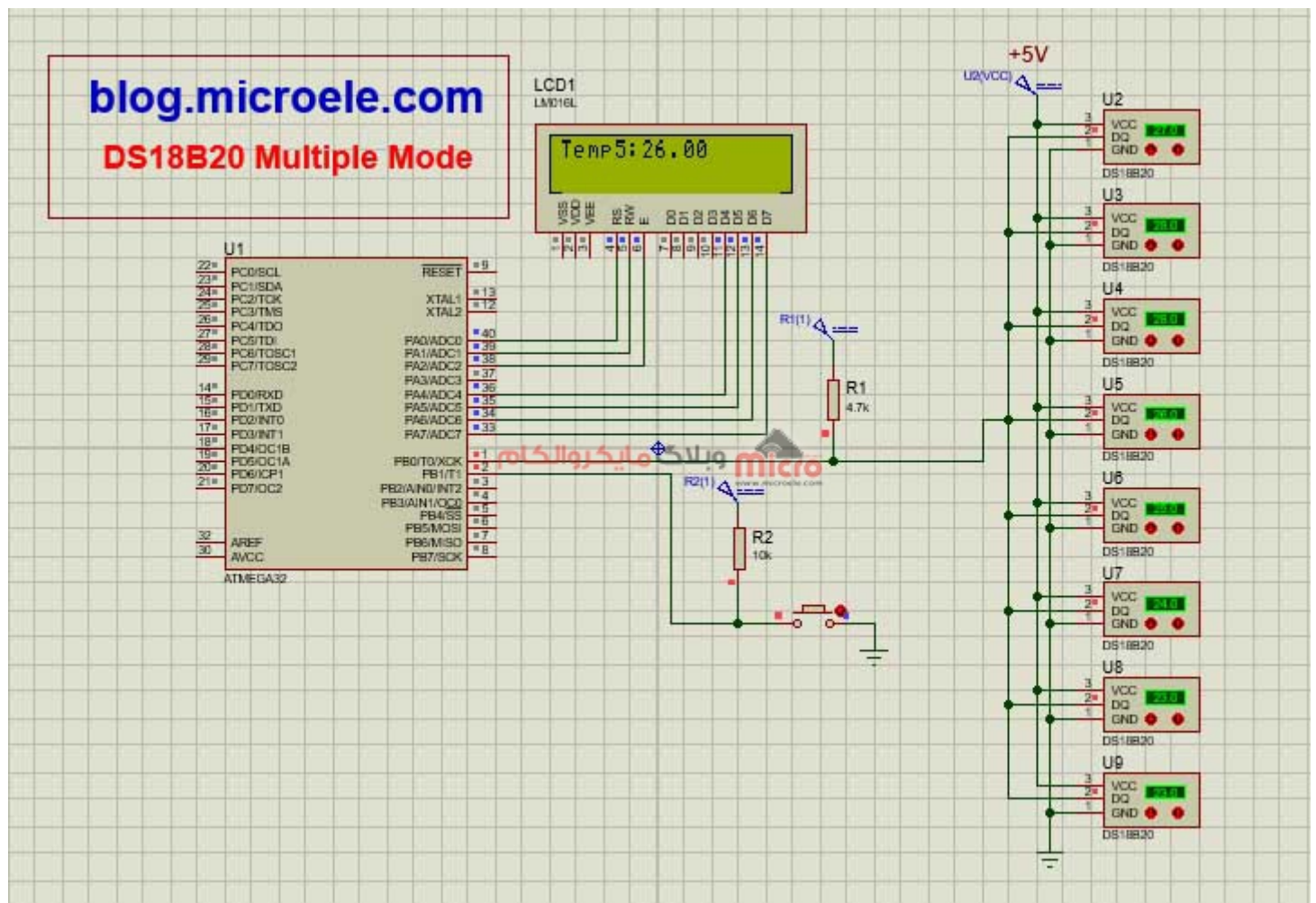
شبیه سازی حالت Multiple سنسور دما DS18B20 در پروتئوس

در محیط پروتئوس بر روی سنسور های دما یک بار کلیک راست و سپس کلیک چپ کرده تا پنجره تنظیمات سنسور باز شود. منوی Automatic Serialization را باز کرده و آن را بر روی Yes قرار دهید. این تنظیم باید برای تمامی سنسورها اعمال شود.



فعال سازی Automatic Serialization سنسور دما در پروتئوس

حال با نگه داشتن کلید در پروتئوس، دمای سنسورها به ترتیب بر روی LCD نمایش داده خواهد شد.



شبیه سازی Multiple Mode در پروتئوس

برای دریافت فایل‌های پروژه و شبیه سازی سنسور دمای DS18B20، از طریق [این لینک](#) اقدام نمایید.

توجه شود که برای اجرای فایل‌های پروتئوس، از ورژن 8.11 به بالا استفاده شود.

جمع بندی

در این مطلب ویژگی های سنسور دمای DS18B20 که یکی از پرکاربردترین سنسورهای دما در پروژه‌های الکترونیکی و صنعتی می‌باشد معرفی و نحوه راه اندازی سنسور دمای DS18B20، با میکروکنترلر AVR و کامپایلر کدویژن، به صورت



تک سنسور و چند سنسور و نحوه شبیه سازی آن در محیط پروتئوس شرح داده شد.

امیدوارم از این آموزش کمال بهره را برده باشید. در صورتی که هرگونه نظر یا سوال داشتید درباره این آموزش لطفاً اون رو در انتهای همین صفحه در قسمت دیدگاه ها قرار بدید. در کوتاه ترین زمان ممکن به اون ها پاسخ خواهم داد. اگر این مطلب براتون مفید بود، اون رو حتماً به اشتراک بگذارید. همینطور میتونید این آموزش را پس از اجرای عملی توی اینستاگرام با هشتگ #microelecom به اشتراک بگذارید و **پیج مایکروالکام** (@microelecom) رو هم منشن کنید.