**آنالوگ به دیجیتال در آردوینو**

در این آموزش مفهوم ADC (تبدیل آنالوگ به دیجیتال) را در ARDUINO UNO معرفی می کنیم. همانطور که در شکل زیر نشان داده شده است، برد آردوینو شش کانال ADC دارد. یک یا همه آنها می تواند به عنوان ورودی برای ولتاژ آنالوگ استفاده شود. آنالوگ به دیجیتال در برد Arduini Uno با دقت 10 بیتی (یعنی مقادیر عدد صحیح بین (0- (2 ^ 10) 1023)) است. به این معنی که ولتاژهای ورودی بین 0 تا 5 ولت را به مقادیر عدد صحیح بین 0 تا 1023 تبدیل میکند. بنابراین برای هر واحد 5/1024 = 4.9mV است.



در این آموزش ما پتانسیومتر را به کانال A0 متصل میکنیم و با تغییر آن ، نتیجه تبدیل آنالوگ به دیجیتال را روی ال سی دی کاراکتری نمایش مدهیم.

**دستورات آنالوگ به دیجیتال در آردوینو**

کانال های Analog Digital Converter دارای مقدار مرجع پیش فرض 5V هستند. این بدان معنی است که ما می توانیم حداکثر ولتاژ ورودی 5 ولت را برای تبدیل ADC در هر کانال ورودی قرار بدهیم. از آنجا که برخی از سنسورها [ولتاژ](https://irenx.ir/electronic/voltage-current/) های 0-2.5 ولت را ارائه می دهند ، با استفاده از یک ولتاژ مرجع 5 ولت دقت کمتری می گیریم، بنابراین ما یک دستورالعمل داریم که به ما امکان تغییر این مقدار مرجع را می دهد. بنابراین برای تغییر مقدار مرجع ما از دستور زیر استفاده میکنیم. ولتاژ مرجع مورد نظر را در بین پرانتز قرار دهید.

("analogReference ()")

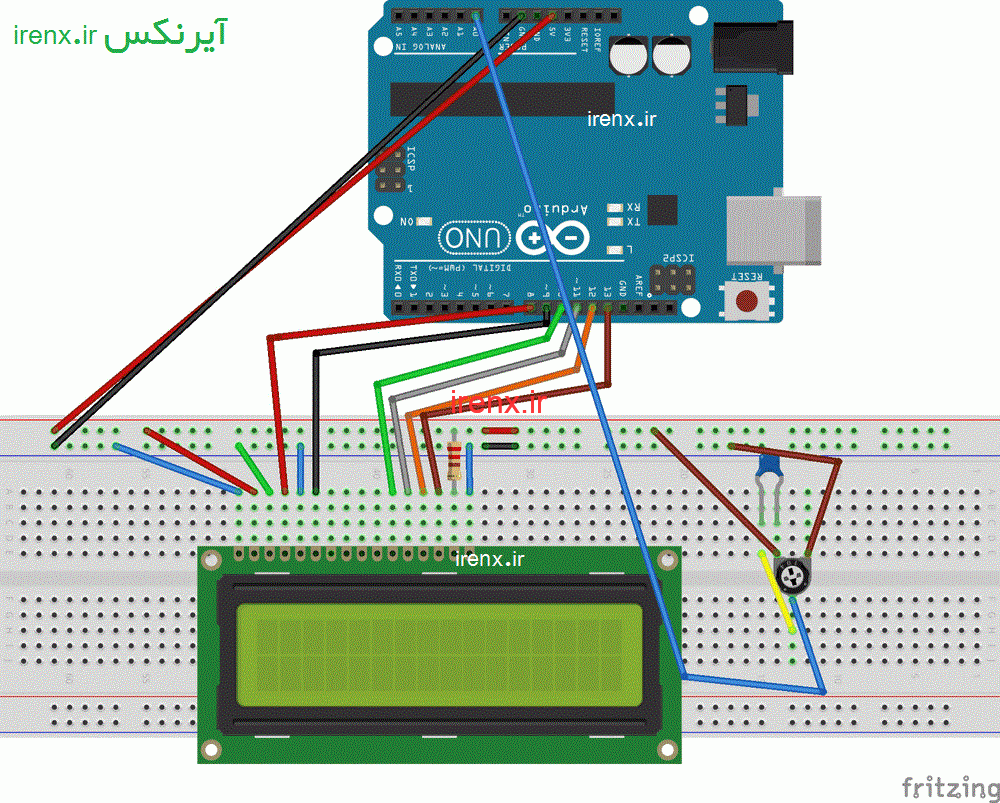
به صورت پیش فرض ما حداکثر دقت ADC را که در برد آردوینو یونو 10 بیت است دریافت می کنیم، این دقت را می توان با استفاده از دستورالعمل زیر تغییر داد. این تغییر دقت ممکن است برای برخی موارد مفید واقع شود. در دستور زیر بین پرانتز بیت مورد نظر را وارد کنید.

(analogReadResolution(bit))

حالا اگر نمیخواهید این دو دستور که در بالا گفتیم را تغییر دهید نیاز به وارد کردن آنها در کد نیست و میتوانید از همان مقدار پیشفرض آن استفاده کنید. حالا می توانیم مقدار را از کانال آنالوگ به دیجیتال ‘0″ به صورت مستقیم  باتابع “((analogRead(pin)” بخوانیم ، در اینجا “pin” نشانگر پینی است که به آن سیگنال آنالوگ را وصل کردیم، در آموزش “A0” است. مقدار ADC را می توان به یک عدد صحیح تبدیل کرد. به دستور العمل زیر دقت کنید، با استفاده از دستور العمل زیر ما مقدار آنالوگ را در متغیر ADCVALUE ذخیره میکنیم.

int ADCVALUE = analogRead(A0);

**توضیحات مدار لازم برای تبدیل آنالوگ به دیجیتال**



آردوینو به کاربر اجازه میدهد تا از ال سی دی کاراکتری در حالت 4 بیتی استفاده کند. این نوع ارتباط کاربر را قادر میسازد تا از پین های کمتری در برد آردوینو استفاده کند. این حالت به صورت پیشفرض در Arduino قرار داده شده است.

مقاله پیشنهادی: [ارتباط سریال SPI بین میکروکنترلر نوتوون nuvoton و آردوینو](https://irenx.ir/arduino/spi-communication-nuvoton-arduino/)

**قطعات مورد نیاز این آموزش Arduino**

1. برد آردوینو Uno
2. منبع تغذیه 5 ولت
3. ال سی دی کاراکتری 16\*2
4. خازن 100uf
5. خازن 100nf
6. پتانسیومتر 100 کیلو اهم

قطعات مورد نیاز را از [**فروشگاه قطعات آیرنکس**](https://irenx.ir/shop/) تهیه کنید.

**کد آنالوگ به دیجیتال در Arduino**

در ابتدای کد ما کتابخانه ال سی دی کاراکتری را فراخوانی میکنیم. در خط دوم پین های LCD کاراکتری را پیکربندی میکنیم. یعنی پین های Enable , d4 , d5 , d6 , d7. سپس مقدار دهی اولیه کاراکتر ADC را انجام میدهیم.

#include <LiquidCrystal.h> // فراخوانی کتابخانه ال سی دی کاراکتری

LiquidCrystal lcd(8, 9, 10, 11, 12, 13); // پین های ENABLE PIN,D4 PIN,D5 PIN, D6 PIN, D7 PIN

char ADCSHOW[5]; //سایز کاراکتر آ.ب.د برای نمایش در ال سی دی

سپس به تابع void setup میرسیم. همانطور که در قسمت های قبل آموزش آردوینو گفتیم، در این قسمت پیکربندی پروژه را انجام میدهیم. در این پروژه و در این قسمت ما تعداد ستون ها و ردیف های LCD کاراکتری را تنظیم میکنیم.

void setup()

{

lcd.begin(16, 2); // تنظیم ستون ها و ردیف های LCD

}

سپس نوبت به تابع void loop میرسد. همانطور که قبلا گفتیم در این قسمت ما یک چرخه تکرار شونده را تنظیم میکنیم. دستورات این تابع به طور مداوم انجام میشوند.

همچنین اگر در مورد این مطلب سوالی داشتید در انتهای صفحه در قسمت نظرات بپرسید

در ابتدا ما در خط اول ال سی دی کاراکتری کلمه دلخواه خود را نشان میدهیم. این کلمه میتواند هرچیزی باشد ما از irenx.ir استفاده کرده ایم. در خط بعدی کد ما مکان متن irenx.ir را تنظیم میکنیم. و در خط سوم این کد نیز ، ردیف دوم ال سی دی کاراکتری را تنظیم میکنیم. تا بنویسد **Natije ADC :**

void loop()

{

lcd.print("IRENX.IR"); //نمایش نام اولیه

lcd.setCursor(0, 1); // تنظیم مکان نام

lcd.print("Natije ADC:"); //نمایش جمله نتیجه ADC

در قسمت بعدی ما در خط اول کد ، مقدار آنالوگ را از پین A0 دریافت میکنیم. در خط دوم با توجه به سایز کاراکتر که در قسمت اول کد انتخاب کردیم این مقدار آنالوگ را به دیجیتال تبدیل میکنیم. و در خط سوم مقدار ADC را در جلوی Natije ADC : نمایش میدهیم. در خط پنجم نیز مکان نمایش ADC را تنظیم میکنیم.

String ADCVALUE = String(analogRead(A0)); //دریافت مقدار اولیه آنالوگ

ADCVALUE.toCharArray(ADCSHOW, 5); //تبدیل مقدار خوانده شده به دیجیتال

lcd.print(ADCSHOW); //نمایش مقدار ADC

lcd.print(" ");

lcd.setCursor(0, 0); // تنظیم مکان نمایش ADC

}

**در باکس زیر کد کامل به همراه توضیح را میبینید.**

#include <LiquidCrystal.h> // فراخوانی کتابخانه ال سی دی کاراکتری

LiquidCrystal lcd(8, 9, 10, 11, 12, 13); // پین های ENABLE PIN,D4 PIN,D5 PIN, D6 PIN, D7 PIN

char ADCSHOW[5]; //سایز کاراکتر آ.ب.د برای نمایش در ال سی دی

void setup()

{

lcd.begin(16, 2); // تنظیم ستون ها و ردیف های LCD

}

void loop()

{

lcd.print(" IRENX.IR"); //نمایش نام اولیه

lcd.setCursor(0, 1); // تنظیم مکان نام

lcd.print("Natije ADC:"); //نمایش جمله نتیجه ADC

String ADCVALUE = String(analogRead(A0)); //دریافت مقدار اولیه آنالوگ

ADCVALUE.toCharArray(ADCSHOW, 5); //تبدیل مقدار خوانده شده به دیجیتال

lcd.print(ADCSHOW); //نمایش مقدار ADC

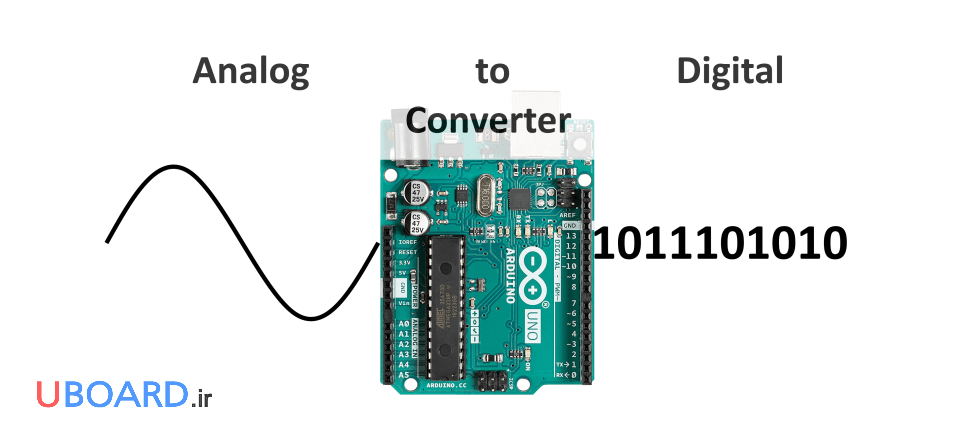
lcd.print(" ");

lcd.setCursor(0, 0); // تنظیم مکان نمایش ADC

}

# [راه اندازی ADC در آردوینو](https://uboard.ir/blog/topic/run-adc-arduino.p114)

## راه اندازی ADC در آردوینو، راه اندازی LCD گرافیکی KS0108 با کتابخانۀ U8g2، اتصالات لازم برای راه اندازی ADC و LCD گرافیکی KS0108، نمونه کد راه اندازی ADC آردوینو، فرمول محاسبۀ دما با TMP36، راه اندازی سنسور دما با آردوینو، راه اندازی فتوسل (LDR) با آردوینو



رضا بداغی

۰۰/۰۶/۱۲

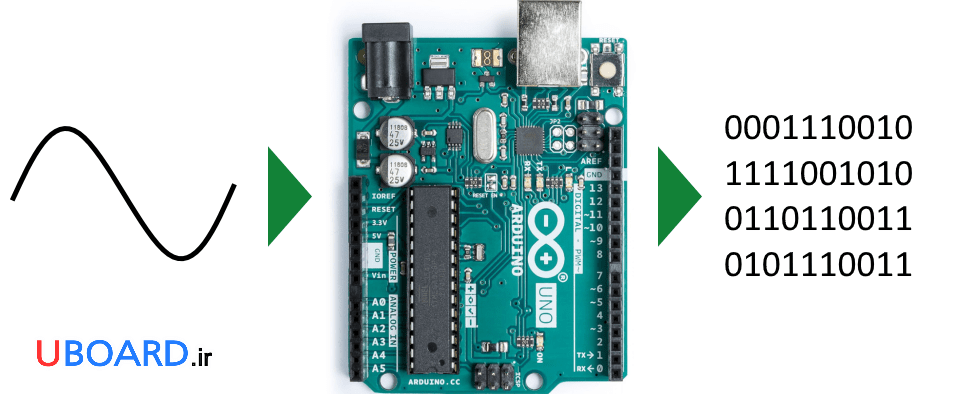
زمان موردنیاز برای مطالعه ۱۵ دقیقه

راه اندازی ADC در [آردوینو](https://uboard.ir/blog/topic/arduino.p104) مانند آنچه در نوشته های «[راه اندازی ADC در AVR](https://uboard.ir/blog/topic/run-adc-ks108-avr.p31)» و «[راه اندازی ADC در STM32](https://uboard.ir/blog/topic/run-adc-stm32.p109)» داشتیم، است. تفاوت راه اندازی ADC آردوینو با راه اندازی ADC میکروکنترلرهای AVR یا میکروکنترلرهای دیگر، راحتی کار با آردوینو است. در پلتفرم آردوینو، با اجرای تنها یک تابع، تبدیل ADC آغاز و نتیجه در خروجی تابع قرار می گیرد. مانند مباحث نوشته های مذکور، مباحث این نوشته هم عبارتند از:

* راه اندازی مبدل آنالوگ به دیجیتال آردوینو؛
* اندازه گیری ولتاژ با آردوینو؛
* راه اندازی سنسور دما با آردوینو؛
* راه اندازی فتوسل با آردوینو؛
* راه اندازی LCD گرافیکی KS0108 با آردوینو.

در اینجا برای راه اندازی KS0108، از کتابخانۀ U8g2 استفاده کرده ایم. راه اندازی ADC آردوینو، در یک شکلِ ساده است. یعنی مد تحریک Single است، رزولوشن و فرکانس نمونه برداری ثابت هستند و از وقفۀ ADC استفاده نمی شود. در بردهای آردوینو مبتنی بر میکروکنترلرهای 8 بیتی AVR، رزولوشن 10 بیتی و فرکانس نمونه برداری حدود 9.6 کیلوهرتز است. در برخی بردهای آردوینو دیگر می توان رزولوشن را تغییر داد. ما در این نوشته از برد Arduino UNO استفاده می کنیم. تنها کاری که نیاز است انجام دهیم، اجرای تابع خواندن کانال ورودی است. خروجی تابع نیز مقدار دیجیتال تبدیل شده است. خواهیم دید که از خروجی این تابع، در توابعی دیگر، به ولتاژ، دما و شدت نور می رسیم. و آنها را در LCD گرافیکی KS0108 نمایش می دهیم. کدهای پروژۀ این نوشته را در Arduino IDE نوشته و پروژه را در [نرم افزار پروتئوس](https://uboard.ir/blog/topic/make-project-proteus.p71) شبیه سازی کرده ایم. فایل های Arduino IDE و پروتئوس و همچنین کتابخانۀ U8g2 در پیوست قرار دارند.

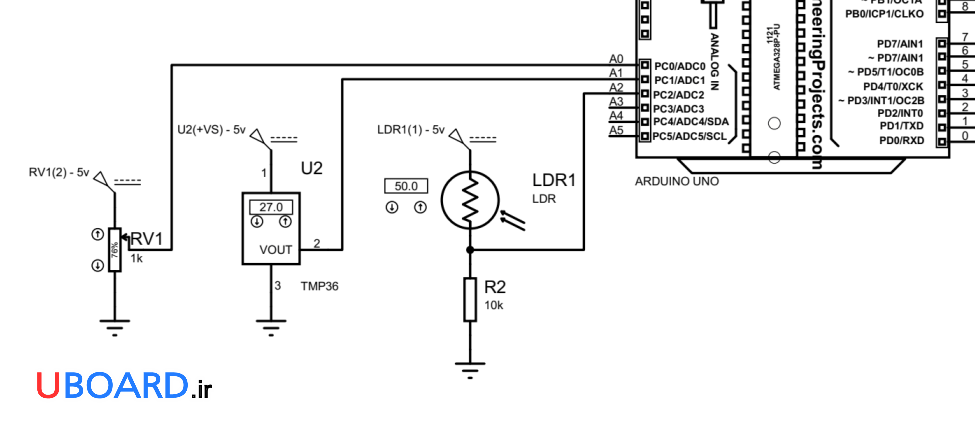
**نکته:** KS0108 را می توان با میکروکنترلرهای STM32 در پلتفرم آردوینو و با استفاده از کتابخانۀ U8g2 راه اندازی کرد. به نوشتۀ «[راه اندازی STM32 با آردینو](https://uboard.ir/blog/topic/stm32-in-arduino.p110)» مراجعه نمایید.



تصویر 1 – راه اندازی ADC در آردوینو

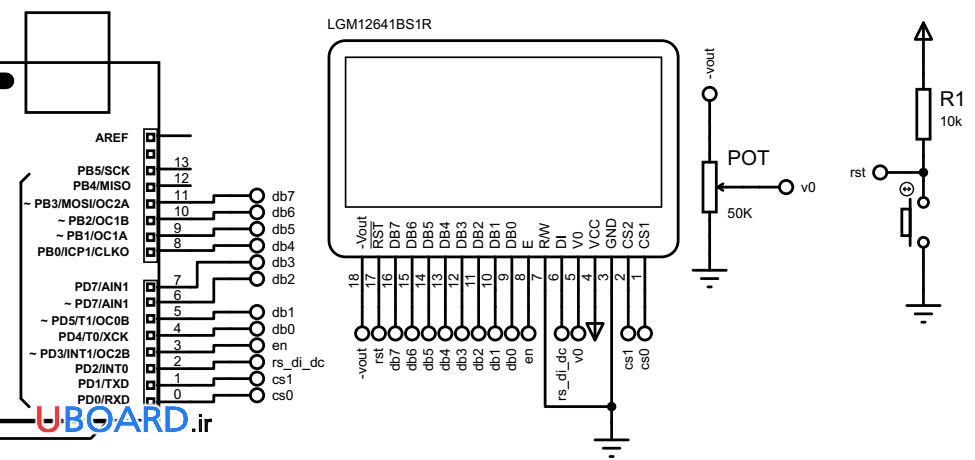
## ****اتصالات لازم برای راه اندازی ADC و LCD گرافیکی KS0108****

در پروژۀ این نوشته، خروجی پتانسیومتر را به پایۀ A0 آردوینو متصل کرده ایم. خروجی سنسور دمای TMP36 و خروجی مدار فتوسل را هم به ترتیب به A1 و A2 وصل کرده ایم. ولتاژ تغذیه برای پتانسیومتر، سنسور دما و فتوسل، 5 ولت است. بنابراین ولتاژ خروجی پتانسیومتر می تواند بین صفر تا 5 ولت باشد. خروجی مدار فتوسل و سنسور دما هم در همین محدوده است. با تعیین ولتاژ مرجع 5 ولت برای ADC، می توانیم ولتاژ خروجی پتانسیومتر، سنسور دما و مدار فتوسل را بخوانیم. در نوشته هایی که در مقدمه به آنها اشاره کردیم، خروجی سنسور دمای LM35 را اندازه گرفته ایم. در اینجا از سنسور TMP36 استفاده کرده ایم. راه اندازی LM35 با آردوینو تفاوت زیادی با راه اندازی سنسور TMP36 ندارد.



تصویر 2 – اتصال فتوسل، سنسور دما و پتانسیومتر به Arduino UNO

برای نمایش نتیجۀ راه اندازی ADC در آردوینو در این نوشته از LCD گرافیکی KS0108 استفاده می کنیم. نحوۀ اتصال KS0108 به آردوینو به صورت زیر است. دقت داشته باشید که پایۀ DI نام های دیگری هم دارد. RS و DC نام های دیگر پایۀ DI هستند. در برخی LCDهای KS0108 ممکن است نام پایۀ CS0 و CS1، به ترتیب CS1 و CS2 باشد. همچنین برخی از این LCDها پایۀ CS2 هم دارند. در LCD گرافیکی که ما استفاده کرده ایم، پایه های CS1 و CS2 وجود دارند که در کد باید CS0 و CS1 فرض شوند. پایۀ RW نیز نمایشگر زمین شده است. توجه کنید که در عمل دو پایۀ دیگر در LCD وجود دارد که مربوط به LED پس زمینه هستند. در LCD مورد استفادۀ ما، این پایه ها، پایه های 19 و 20 هستند. که آنها را به ترتیب به 5 ولت و زمین متصل می کنیم.



تصویر 3 – اتصالات لازم برای راه اندازی KS0108 با آردوینو

## ****نمونه کد راه اندازی ADC در آردوینو****

ابتدا پایه هایی که به خروجی پتانسیومتر، سنسور دمای TMP36 و فتوسل متصل است، دیفاین کرده ایم. ولتاژ مرجع پیش فرض برابر ولتاژ پایۀ AVCC [میکروکنترلر](https://uboard.ir/blog/topic/microcontroller-introduction.p69) روی برد آردوینو یعنی 5 ولت است. روزولوشن ADC برابر 10 بیت است. 210 – 1 که در فرمول ولتاژ استفاده می کنیم، برابر 1023 است. آنها را دیفان کرده ایم. سپس کتابخانۀ U8g2 را به پروژه اضافه کرده ایم. در کتابخانۀ U8g2 برای LCD گرافیکی KS0108 کلاس هایی وجود دارد. که ما از کلاس U8G2\_KS0108\_128X64\_F شیء my\_glcd را ایجاد کرده ایم. پارامتر ورودی اول، تعیین کنندۀ Rotation یا چرخش است. که می خواهیم جهت نمایش، افقی رو به پایین باشد. بنابراین آن را برابر U8G2\_R0 قرار داده ایم. U8G2\_R0 تا U8G2\_R3 در کتابخانۀ U8G2 دیفاین شده اند. پارامترهای ورودی دوم تا نهم، شمارۀ پایه های D0 تا D7 نمایشگر هستند. پارامتر ورودی دهم تا سیزدهم نیز پایه های دیگر LCD هستند. دو پارامتر ورودی آخر هم به ترتیب پایه های CS2 و Reset نمایشگر هستند. LCD مورد استفادۀ ما پایۀ CS2 ندارد و پایۀ Reset هم به آردوینو متصل نمی کنیم. بنابراین مقدار آنها را برابر U8X8\_PIN\_NONE که یک عبارت دیفاین شده در کتابخانۀ U8g2 است، قرار داده ایم.

#define potentiometer\_output\_pin A0

#define tmp36\_output\_pin A1

#define photocell\_output\_pin A2

#define vref 5.0 // Default

#define resolution 1023 // 10 bit

#define d0\_pin 4

#define d1\_pin 5

#define d2\_pin 6

#define d3\_pin 7

#define d4\_pin 8

#define d5\_pin 9

#define d6\_pin 10

#define d7\_pin 11

#define en\_pin 3

#define rs\_pin 2 // or di\_pin or dc\_pin (dc = data / command)

#define cs0\_pin 0 // or cs1\_pin (On my GLCD, the cs1 pin is the same as cs0.)

#define cs1\_pin 1 // or cs2\_pin (On my GLCD, the cs2 pin is the same as cs1.)

//#define cs2\_pin ... // or U8X8\_PIN\_NONE (There is no CS2 pin on my GLCD.)

#include

U8G2\_KS0108\_128X64\_F my\_glcd(U8G2\_R0, // Rotation

d0\_pin,

d1\_pin,

d2\_pin,

d3\_pin,

d4\_pin,

d5\_pin,

d6\_pin,

d7\_pin,

en\_pin,

rs\_pin,

cs0\_pin,

cs1\_pin,

U8X8\_PIN\_NONE, // CS2\_pin

U8X8\_PIN\_NONE); // Reset\_pin

در ادامه توابع محاسبۀ ولتاژ پتانسیومتر، دمای اندازه گیری شده توسط TMP36 و شدت نور بر حسب درصد، آورده ایم. مقدار ولتاژ ورودی برابر است با:

(digital\_rsult \* Vref) / 1023

که آن را در تابع ولتاژ آورده ایم. فرمول محاسبۀ دما از سنسور TMP36 هم به این صورت است:

(Volatge - 0.5)\*100

که Voltage را با تابع ولتاژ محاسبه کرده ایم و در فرمول دما در تابع دما قرار داده ایم. تابع شدت نور نیز نتیجۀ تبدیل خروجی مدار فتوسل را که از صفر تا 1023 است، با تابع map، به مقداری بین صفر تا 100 تبدیل می کند.

float voltage(unsigned int voltage\_d)

{

return (((float)voltage\_d \* vref)/resolution);

}

float temperature(unsigned int temp\_d)

{

float temp = voltage(temp\_d);

temp -= 0.5; // 0.5 is offset of TMP36

temp \*= 100; // from TMP36 datasheet

return temp;

}

unsigned int light(unsigned int light\_d)

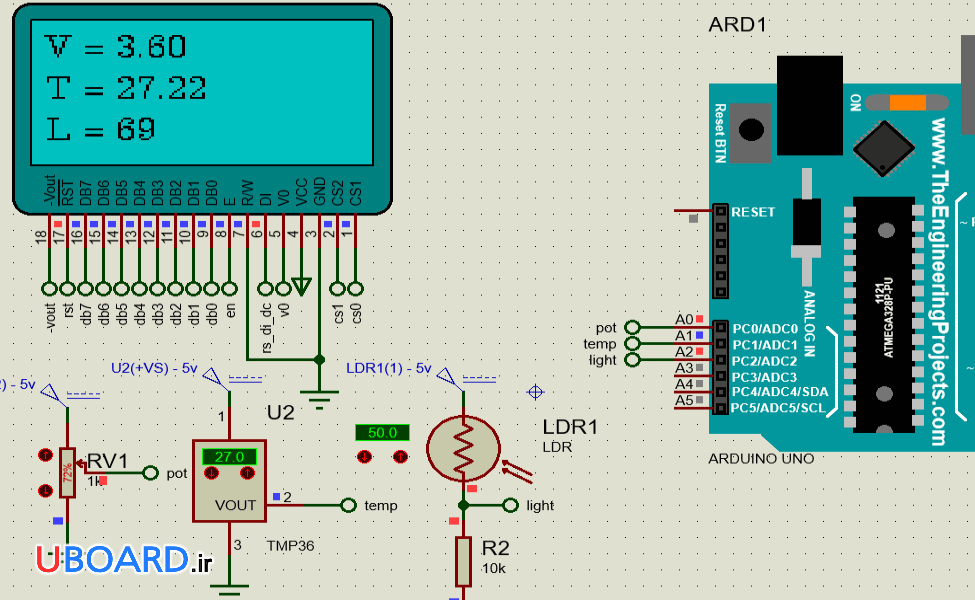
{

unsigned int light\_percent = 0;

light\_percent = map(light\_d, 0, 1023, 0, 100);

return light\_percent;

}



تصویر 4 – نتیجۀ راه اندازی ADC در آردوینو در نرم افزار پروتئوس

### ****راه اندازی KS0108، پتانسیومتر، TMP36 و فتوسل با آردوینو****

در تابع setup ابتدا تابع begin را فراخوانی کرده ایم. سپس با تابع analogReference، ولتاژ مرجع ADC را روی حالت پیش فرض (ولتاژ AVCC) قرار داده ایم. بعد با تابع clear نمایشگر را پاک و یک فونت را برای نمایش تعیین کرده ایم. بعد از آن، رشتۀ UBOARD.ir را به تابع drawStr ارسال کرده ایم. در سطرهای بعد فونت را تغییر و رشته های دیگر  را در تابع drawStr قرار داده ایم. ورودی اول و دوم drawStr ستون و سطری از LCD است که رشته در آن نمایش داده می شود. سپس با اجرای تابع sendBuffer، رشته هایی را که در تابع drawStr قرار داده ایم، در LCD نمایش داده ایم. در نهایت یک تأخیر 2.5 ثانیه ای ایجاد کرده ایم.

void setup(void) {

my\_glcd.begin();

analogReference(DEFAULT);

my\_glcd.clear();

my\_glcd.setFont(u8g2\_font\_ncenB14\_tr);

my\_glcd.drawStr(8, 15, "UBOARD.ir");

my\_glcd.setFont(u8g2\_font\_ncenR10\_tr);

my\_glcd.drawStr(0, 32, "This project is a");

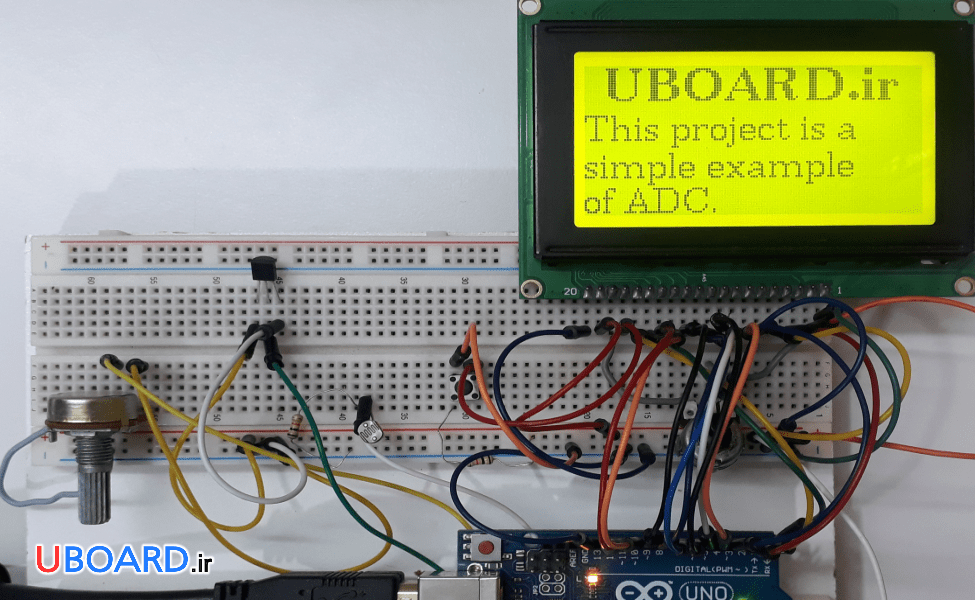
my\_glcd.drawStr(0, 48, "simple example");

my\_glcd.drawStr(0, 63, "of ADC.");

my\_glcd.sendBuffer();

delay(2500);

}



تصویر 5 – راه اندازی KS0108 با آردوینو با کتابخانه U8g2

در ادامه در تابع loop با تابع analogRead ولتاژ ورودی ها را به مقادیر دیجیتال تبدیل کرده ایم. این مقادیر را در channel\_x\_value قرار داده ایم. سپس LCD را پاک کرده ایم. بعد از آن مقدار channel\_1\_value و channel\_2\_value را به ترتیب در توابع voltage و temperature قرار داده ایم. خروجی این توابع را که از نوع float است، به تابع dtostrf ارسال کرده ایم. تابع dtostrf، مقادیر float را به رشته تبدیل می کند. ورودی اول آن، مقدار float مورد نظر است. ورودی دوم، سایز رشته، ورودی سوم، تعداد ارقام اعشار و ورودی چهارم نیز یک رشته است. سپس با تابع sprint رشتۀ str\_float را در رشتۀ glcd\_str قرار داده ایم. بعد تابع drawStr را اجرا کرده ایم. سپس channel\_2\_value را در تابع light و خروجی light را با spintf در glcd\_str قرار داده ایم. تابع drawStr و sendBuffer را فراخوانی کرده و در نهایت یک تأخیر 500 میلی ثانیه ای به وجود آورده ایم. نتیجۀ راه اندازی ADC در آردوینو در این چند سطر کد، در تصویر زیر آمده است.

[انواع متغیر در زبان C](https://uboard.ir/blog/topic/variable-types-c-language.p76)

void loop(void) {

unsigned int channel\_0\_value, channel\_1\_value, channel\_2\_value;

char glcd\_str[25], str\_float[10];

channel\_0\_value = analogRead(potentiometer\_output\_pin);

channel\_1\_value = analogRead(tmp36\_output\_pin);

channel\_2\_value = analogRead(photocell\_output\_pin);

my\_glcd.clear();

dtostrf(voltage(channel\_0\_value), 4, 2, str\_float);

sprintf(glcd\_str, "V = %s", str\_float);

my\_glcd.setFont(u8g2\_font\_ncenR10\_tr);

my\_glcd.drawStr(3, 15, glcd\_str);

dtostrf(temperature(channel\_1\_value), 4, 2, str\_float);

sprintf(glcd\_str, "T = %s", str\_float);

my\_glcd.setFont(u8g2\_font\_ncenR10\_tr);

my\_glcd.drawStr(3, 35, glcd\_str);

sprintf(glcd\_str, "L = %d", light(channel\_2\_value));

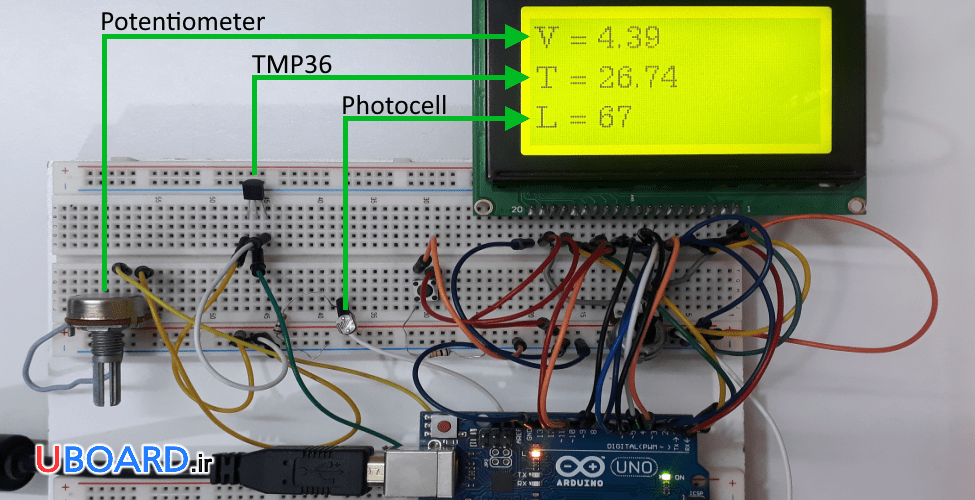
my\_glcd.setFont(u8g2\_font\_ncenR10\_tr);

my\_glcd.drawStr(3, 55, glcd\_str);

my\_glcd.sendBuffer();

delay(500);

}



تصویر 6 – نتیجۀ عملی راه اندازی ADC در آردوینو و نمایش در LCD گرافیکی KS0108

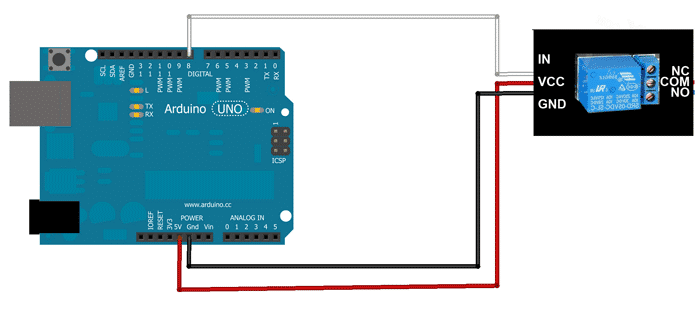
## ****نتایج راه اندازی ADC در آردوینو****

1. برنامه نویسی ADC در آردوینو، نسبت به برنامه نویسی آن با [میکروکنترلرهای AVR](https://uboard.ir/blog/topic/avr-microcontroller.p96) و میکروکنترلرهای دیگر، ساده تر است. مقدار آنالوگ روی پایۀ ورودی، با اجرای تنها یک تابع، به دیجیتال تبدیل می شود.
2. در ADC آردوینو پیچیدگی ها و جزئیات وجود ندارد و بیشتر تنظیمات ADC غیر قابل تغییرند. مثلاً رزولوشن 10 بیتی و مد تحریک Single است. اما در راه اندازی ADC در میکروکنترلرها می توان تنظیمات متعددی داشت. مثلاً در [ADC در AVR](https://uboard.ir/blog/topic/%D8%A2%D9%85%D9%88%D8%B2%D8%B4-adc-%D8%AF%D8%B1-avr.p52) می توان رزولوشن را 8 بیتی یا 10 بیتی تنظیم کرد.
3. راه اندازی TMP36 با آردوینو تقریباً مشابه راه اندازی LM35است. فرمول های محاسبۀ دما در راه اندازی این دو سنسور متفاوت است.
4. برای خواندن ولتاژ با آردوینو یا میکروکنترلرها باید دقت داشت که محدودۀ ولتاژی را که می توان با ADC اندازه گرفت، صفر تا ولتاژ مرجع ADC است.
5. همۀ LCDهای گرافیکی KS0108، ترتیب و نوع پایه های یکسانی ندارند. از این لحاظ ممکن است انواع متفاوتی وجود داشته باشد.
6. ولتاژ روی کانال ورودی ADC از این فرمول به دست می آید: (digital\_result \* Vref)/(2Resolution - 1)
7. تابع map مقدار یک عدد را که در یک محدوده است، به یک عدد متناسب با محدوده ای دیگر تبدیل می کند.
8. تابع analogReference برای تعیین ولتاژ مرجع ADC است. که برای برد Arduino UNO می تواند در حالت DEFAULT (ولتاژ پایۀ AVCC) یا EXTERNAL (ولتاژ خارجی روی پایۀ AREF) یا INTERNAL (ولتاژ مرجع 1.1 ولت داخلی) باشد.
9. برای راه اندازی مبدل آنالوگ به دیجیتال در آردوینو کافی است تابع analogRead اجرا شود.
10. برای تبدیل مقادیر float به رشته در آردوینو نمی توان مستقیماً از تابع sprintf استفاده کرد. در آردوینو تابع dtostrf استفاده می شود.

# راه اندازی رله با آردوینو

توسط [حمیدرضا مودت](http://levinic.ir/author/hamidr/)

راه اندازی رله با آردوینو

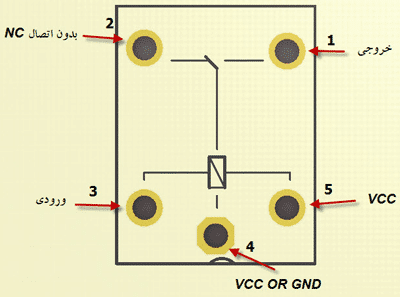
[](http://levinic.ir/wp-content/uploads/set-up-a-5v-relay-on-the-arduino.png)

قبل از اینکه بریم سراغ راه اندازی رله با آردوینو بهتره که اول تعریف کنیم این قطعه چیه، عموما در بحث راه اندازی قطعات با آردوینو از ماژول ها یا شیلد ها استفاده میشه ، در این بخش هم ما ابتدا توضیحی درباره ی رله و خود این قطعه میدیم و بعد میریم سراغ ماژول رله 5 ولتی.

## رله Relay چیست ؟

رله یک کلید یا سوئیچ الکترونیکیه ، مثل کلیدی که لامپ رو باهاش روشن میکنین! فقط این قطعه با اعمال ولتاژ این کار رو براتون انجام میده و مشخصه که خیلی قطعه پرکاربردی میتونه باشه. مثلا وقتی که طبق برنامه ی خاص و دقیقی قرار باشه موتوری روشن/خاموش بشه یا به وسیله پیامک چراغی روشن/خاموش بشه میشه از رله استفاده کرد و  …

عموما رله ها 4 یا 5 پایه هستن ، که اینجا بیشتر هدف مون روی معرفی پرکاربرد ترین رله بازار ینی رله 5 ولتی (مدل 12 ولتی هم تفاوتی توی ساختار نداره) و 5 پایه هستش که به شکل زیره.

[](http://levinic.ir/wp-content/uploads/relay-5-volt-pin.png)

1- پایه یک خروجی ما هستش که اون رو به LED ، موتور یا هر وسیله ای که قراره روشن بشه وصل میکنیم. این پایه در حالت عادی خاموشه. یا اصطلاحا N.C (Normally closed) هستش.

2-این پایه مثل پایه یکه فقط در حالت عادی همیشه وصله و با اعمال فرمان خروجی قطع میشه.این پایه اصطلاحا  N.O (Normally open) هستش.

اگر این پایه رو استفاده نکنیم بدون اتصال باقی میمونه

ممکنه بخوایم با وصل شدن یک وسیله به برق، همون لحظه وسیله روشن بشه، در این صورت از پایه 2 استفاده میکنیم. اما بیشتر مواقع یک وسیله رو به برق میزنیم و تا فرمان روشن شدن بهش ندیم (مثلا یک موتور) انتظار نداریم که روشن بشه! در نتیجه بیشتر از پایه یک استفاده میکنیم(ما هم در ادامه از پایه یک استفاده میکنیم!)

3- پایه 3 ، پایه ای هستش که قراره ما به اون فرمان بدیم، یعنی با صفر و یک شدن اون تغییر وضعیت در خروجی ایجاد میشه.

4- برای قطع و وصل کردن جریان led یا موتور ، کافیه از سمت سیم فاز یا نول سیم رو قطع کنیم و کلید رو در اون جا قرار بدیم ، این پایه هم همین کاربرد رو داره.

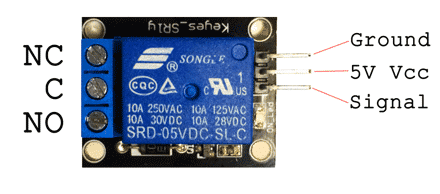
یعنی شما میتونین پایه VCC یا GND رو به این پایه وصل کنید و در صورت اعمال فرمان به رله، در خروجی رله،  VCC یا GND رو دریافت کنید و به موتور یا LED بدین ! البته توصیه میشه که سعی کنید VCC  رو به این پایه بدین…

5- پایه پنج هم VCC مدار وصل میشه و با پایه 3 یک میدان مغناطیسی برای اتصال داخلی رله فراهم می کنه تا فرمان اجرا بشه.

**توجه:**بین پایه 3 و 5 یک دیود قرار می گیره تا از جریان برگشتی جلوگیری بشه(پایه 3(کاتد) و پایه 5(آند))

## راه اندازی ماژول رله با آردوینو

خب توضیحات تا اینجا درباره ی قطعه ی رله بود ، اما ماژول هایی هستنن که کار مدار بستن رو هم ساده تر میکنن! ماژول رله پایه ها به شرح زیره :

[](http://levinic.ir/wp-content/uploads/relay-module-5-volt-pin.png)

پایه ی NC:  که همون پایه ی Normally closed هستش

پایه ی NO: که همون پایه ی Normally open هست.

پایه ی C: پایه ی مشترک ( ولتاژ VCC یا GND ای که قراره به LED موتور یا .. بره رو به این وصل میکنیم!)

پایه ی  Ground: به GND آردوینو وصل میشه.

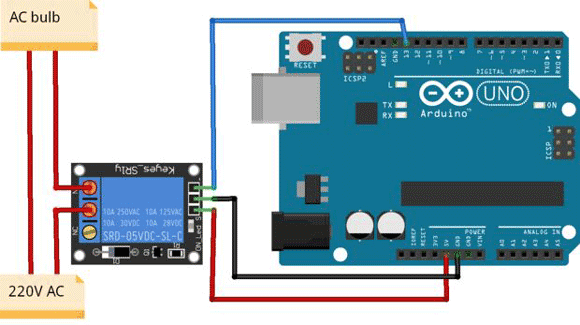
پایه ی  5V Vcc: به 5 ولت آردوینو وصل میشه.

پایه ی Signal: با فرمان آردوینو به این پایه خروجی تغییر وضعیت میده.(به پایه های دیجتال وصل میشه)

بعضی از ماژول ها یک جامبر دارن که پایه های H و L داره ، اون پایه برای تعیین وضیعت نوع تحریک رله هستش! حالا این یعنی چی؟ یعنی اینکه اگر پایه ی Signal یک شد رله عمل کنه یا اگر صفر شد؟ که میشه با جامپر تعیین کرد روی کدوم حالت ، رله تغییر وضیعت بده.

این بخش بیشتر برای معرفی بود و فقط به توضیخ یک پروژه ساده در ادامه اکتفا میکنم ، اما [پروژه آردوینو](http://levinic.ir) بیشتری در بخش [پروژه های آردوینو](http://levinic.ir/category/پروژه-های-آردوینو/) توسط رله با آردوینو به زودی قرار میگیره.

### پروژه ی آردوینو با رله

[](http://levinic.ir/wp-content/uploads/relay-arduino-schematic.png)

پروژه اول که خیلی سادست و جلسات اول [آموزش آردوینو](http://levinic.ir) اون رو توضیح دادیم

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11 | void setup()  {    pinMode(13, OUTPUT);  }    void loop() {    digitalWrite(13,LOW);    delay(4000);              // wait for 5 seconds    digitalWrite(13,HIGH);    delay(4000);              // wait for 5 seconds  } |

و پروژه دوم رله با آردوینو :

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14 | int relay=13;    void setup() {    // put your setup code here, to run once:  pinMode(13, OUTPUT);  Serial.begin(9600);  }    void loop() {    if(Serial.available()>0){      int dat=Serial.read()-48;      Serial.println(dat);      digitalWrite(relay,dat);  }} |

این پروژه از ارتباط سریال فرمان میگیره ، در ارتباط سریال (فرض میکنیم جامبر نداریم و اگر هم داریم روی ماژول روی حالت H یا HIGH تنظیم شده) اگر عدد یک فرستاده بشه خروجی وصل میشه و اگر عدد 0 ارسال بشه خروجی قطع میشه.

طبق [جدول کدهای اسکی](http://levinic.ir/wp-content/uploads/Ascii-table.png) با ارسال عدد 0  میکرو عدد 48 رو برگشت میده ، که برای برابر صفر شدن وقرار دادن در دستور digitalWrite اون رو بعد از خوندن منهای مقدار 48 میکنیم.

این قضیه برای مقدار 1 هم صادقه و با ارسال عدد 1 در پورت سریال عدد 49 دیده میشه که او« هم از 48 کم میشه و خروجی برابر با 1 میشه.

# در آردوینو – آموزش آردوینو جلسه ششم

توسط [حمیدرضا مودت](http://levinic.ir/author/hamidr/)

خیلی از سیگنال ها توی دنیای واقعی پیوسته و آنالوگ هستند اما ما از قطعات دیجیتال استفاده میکنیم و به طبع دوسداریم سیگنال های پیوسته رو به دیجیتال تبدیل کنیم !برای تبدیل این سیگنال ها از مبدل های انالوگ به دیجیتال یا ADC استفاده میکنیم.ADC آی سی هم داره ، ولی توی میکروکنترلر ها این واحد وجود داره و درنتیجه ADC در آردوینو هم وجود داره و لازم به استفاده از آی سی دیگه ای نداریم.

### کاربرد ADC در آردوینو – مبدل آنالوگ به دیجیتال

بعنوان مثال یک سری از سنسور های دما مثل سری LM و به طور خاص LM35 که خیلی پر مصرفه توی پروژه ها ، خروجی ای با ولتاژ آنالوگ دارن و با تغییر دما ، ولتاژ پایه های خروجیشون تغییر میکنه و ما نیاز داریم برای خوندنشون خروجی رو تبدیل به دیجیتال کنیم.

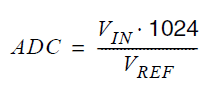
### استفاده و راه اندازی واحد ADC در آردوینو یا مبدل آنالوگ به دیجیتال

توی مبدل های آنالوگ به دیجیتال 2 مشخصه داریم که خیلی مهم

1. دقت
2. سرعت

دقت تراشه برای اندازه گیری کمترین مقدار سیگنال آنالوگ مهمه . این مقدار تناسب داره با تعداد  بیت های خروجی داده های دیجیتال (زمانی که تبدیل انجام شده)

و سرعت که به فرکانس نمونه برداری مربوط میشه و هرچقدر فاصله زمانی بین نمونه برداری کمتر باشه(همون دوره تناوب) سرعت هم بیشتر میشه.مقدار داده ی دیجیتالی خروجی هم با فرمول زیر حساب میشه:



VIN: ولتاژ ورودیمونه که قراره تبدیل بشه.

VREF : ولتاز رفرنسه که جلوتر توضیح میدم . ولی درحالت دیفالت و معمول 5 ولته.

طبق قانون نایکوئیست فرکانس نمونه برداری باید حداقل 2 برابر بیشترین فرکانس موجود سیگنال باشه. باید حواسمون باشه به اینکه بالا بودن فرکانس نمونه برداری باعث بالارفتن توان مصرفی میشه و این برای سیستم هایی که قراره با باتری کار کنن خیلی مهمه. مثلا وقتی داریم از یک سنسور دما یا رطوبت استفاده میکنیم زیاد سرعت نمونه برداری ی بالا مهم نیست چون دما و رطوبت توی زمان پایین تغییر وضعیت نمیدن.(برای هر سنسوری میشه زمان بین تغییرات خروجیش رو توی دیتاشیت نگاه کرد)

میکورکنترلر های avr حداکثر 10 بیت دقت دارن و بازه ی نمونه برداری از 0 تا ولتاژ مرجع (فرض میکنیم 5ولت) رو به 1024 قسمت تقسیم می کنند.بنابراین با اضافه شدن حدود 5 میلی ولت به سیگنال آنالوگ یک بیت به مقدار دیجیتال اضافه میشه!(توی فرمول بالا خودتون حساب کنید به این نتیجه میرسین)

ADC در آردوینو UNO و بقیه ی آردوینو ها هم 10 بیتی ای هست چون هسته avr دارن ، در سری هایی که هسته ARM دارن هم ای مقدار 12 بیته.

فک کنم بهتره بقیه توضیح رو با یک مثال ادامه بدم :

مثلا میخوایم با یک LM35 کار کنیم که یک سنسور دماست، توی دیتاشیت این آی سی که نگاه میکنیم میبینیم که گفته 10 میلی ولت به ازای هر درجه سانتی گراد به خروجیش اضافه میشه.دقت آی سی هم تغریبا 0.5 درجه هست ، یعنی توانایی اندازه گیری 0.5 درجه سانتیگراد رو با دقت خوبی داره، پس به ازای هر 0.5 درجه 5 میلی ولت در خروجی افزایش داریم که برابر افزایش یک بیت در مقدار دیجیتال هست.بنابراین ، مبدل کاملا سیگنال آنالوگ رو ردیابی میکنه و دقت خوبی داره.البته اگه قرار باشه دما رو بدون اعشار نشون بدیم دقت حتی بیش از نیاز ما هم هست

برای انتخاب ولتاژ مرجع هم باید اینو بدونیم که ولتاژ مرجع به دامنه سیگنال آنالوگ ورودی وابسته هستش، آردوینو ، 2 ولتاژ مرجع 5 ولت و داخلی رو بدون نیاز به قطعه خارجی فراهم کرده(دمش گرم )

برای استفاده از ولتاژ های مرجع باید از تابع زیر استفاده کنیم:

### تابع ()analogReference

که ورودی اون شامل متغیرهای زیره:

DEFAULT: ولتاژ 5  (3.3)

INTERNAL: ولتاژ 1.1 ولت در تراشه های ATMEGA168 & ATMEGA328

INTERNAL1V1: ولتاژ 1.1 ولت تنها در بردهای مگا

INTERNAL3V56: ولتاژ 2.56 ولت تنها در بردهای مگا

EXTERNAL: ولتاژ مرجع دلخواه بین صفر تا 5 ولت در پایهAREF

### تابع analogRead()

برای تبدیل و خوندن داده، تابع analogRead رو استفاده میکنیم ومتغیر ورودی تابع ، شماره پایه ی ورودی هستش و خروجی تابع یک عدد ده بیتی صفر تا 1023 هستش که مقدار دیجیتال و متناظر سیگنال آنالوگ در لحظه نمونه برداری هست.

(یک نکته هم اینکه هر تبدیل در مبدل آنالوگ به دیجیتال به جز تبدیل اول ، سیزده پالس ساعت طول میکشه)

تابع analogReadResolution()

(مخصوص برد های DUE و ZERO (M0 PRO))

یک تابع هم مخصوص برد های DUE و ZERO (M0 PRO)وجود داره ، اول مطلب گفتم که میکروهای AVR دقت 10 بیتی دارن ، اما این برد ها چون هسته ARM دارن دارای دقت بالاتری و 12 بیتی هستند وخروجی بین 0 تا 4095 دارن.البته با این کد میشه دقتی 16 بیتی هم داشت و درکل وظیفه این تابع اینه که دقت خوندن ورودی آنالوگ رو تنظیم میکنه.

اینم یک مثال برای این تابع:

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27  28  29  30  31  32  33  34  35  36  37  38  39  40  41  42  43  44  45  46  47  48  49  50  51  52  53  54 | void setup() {      // open a serial connection      Serial.begin(9600);    }      void loop() {      // read the input on A0 at default resolution (10 bits)      // and send it out the serial connection      analogReadResolution(10);      Serial.print("ADC 10-bit (default) : ");      Serial.print(analogRead(A0));        // change the resolution to 12 bits and read A0      analogReadResolution(12);      Serial.print(", 12-bit : ");      Serial.print(analogRead(A0));        // change the resolution to 16 bits and read A0      analogReadResolution(16);      Serial.print(", 16-bit : ");      Serial.print(analogRead(A0));        // change the resolution to 8 bits and read A0      analogReadResolution(8);      Serial.print(", 8-bit : ");      Serial.println(analogRead(A0));        // a little delay to not hog serial monitor      delay(100);    } |

توضیح بسه دیگه !بریم سراغ چند تا مثال! سوال داشیتن اول توی گوگل بگردین (اینطوری بهتر یاد میگیرین)و اگر به جواب نرسیدین بپرسین ، حتما سعی میکنم کمک کنم   
یک برنامه بنویسید که از پایه ی A0 سیگنال آنالوگ رو بخونه و مقدار اون رو به شکل سریال بفرسته : ([ارتباط سریال در آردوینو – آموزش برنامه نویسی آردوینو – جلسه چهارم](http://levinic.ir/%d8%a7%d8%b1%d8%aa%d8%a8%d8%a7%d8%b7-%d8%b3%d8%b1%db%8c%d8%a7%d9%84-%d8%af%d8%b1-%d8%a2%d8%b1%d8%af%d9%88%db%8c%d9%86%d9%88/) رو اگر یادتون رفته میتونید نگاه بندازین)



|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16 | void setup() {      Serial.begin(9600);      analogReference(DEFAULT);    }      void loop() {      int sensorValue = analogRead(A0);//"int sensorValue" ro mishod balaye void setup ham nevesht .      Serial.println(sensorValue);    } |

برنامه ای بنویسید که یک led رو با توجه با مقدار خوانده شده از حسگری که به پایه ی صفر آنالوگ وصل شده روشن و خاموش کنه.

با این مشخصات که اگر مقدار خوانده شده از پایه ی انالوگ کمتر از 800 بود پایه روشن بشه و اگر نبود خاموش بشه. (مقدار 800 معادل 3.9 ولت میشه)

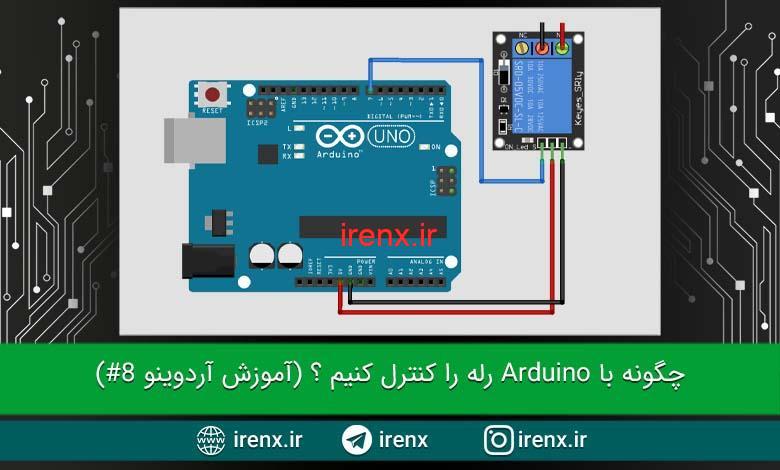


|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27  28  29  30  31  32  33  34  35  36  37 | int pininput=A0;    int ledpin=9;        void setup() {    pinMode(ledpin,OUTPUT);    analogReference(DEFAULT);      }      void loop() {    int val=analogRead(pininput);      if(val<800)      {        digitalWrite(ledpin,HIGH);      }      else        {          digitalWrite(ledpin,LOW);      }    } |

# آموزش کنترل رله با برد آردوینو (کنترل وسایل)

محمد رحیمی1399/03/16

زمان تقریبی مطالعه 4 دقیقه



سلام. آموزش نحوه کنترل رله با آردوینو را آماده کرده ایم. (آموزش Arduino قسمت 8) را آماده کرده ایم.

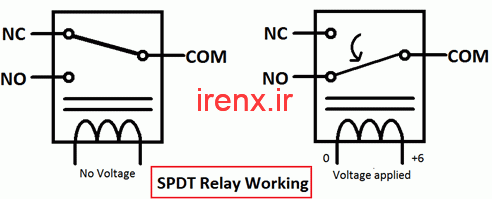
محتویات [[نمایش](https://irenx.ir/arduino/relay-control-with-arduino/)]

### رله چیست و چگونه کار میکند ؟

رله یک سوئیچ الکترومغناطیسی است که توسط جریان کم کنترل می شود و برای روشن و خاموش کردن جریان بزرگتر مورد استفاده قرار می گیرد. با استفاده از جریان کوچک می توان حالت رله را تغییر داد. رله گزینه خوبی برای کنترل دستگاه های AC ([**جریان متناوب**](https://irenx.ir/electronic/alternating-current/)) با استفاده از یک جریان DC بسیار کوچکتر است. رله دارای پنج پایه به شرح زیر است :

برای یادگیری آردوینو (برنامه نویسی، ساخت ربات، ارتباط با اندروید) روی [**دوره آموزش آردوینو**](https://irenx.ir/arduino-course/) کلیک کنید.

برای یادگیری کامل الکترونیک روی[**دوره آموزش الکترونیک**](https://irenx.ir/electronic-course/) کلیک کنید.



هنگامی که هیچ ولتاژی به سیم پیچ اعمال نشود، Com به NC متصل است. و هنگامی که مقدار ولتاژ به سیم پیچ اعمال شود، میدان الکترو مغناطیسی تولید میشود که آرماتور متصل به NC را به سمت خود جذب میکند و باعث میشود COM و NO به هم متصل شوند. این کار باعث میشود بتوانیم با استفاده از جریان کمتر مثل جریان برد آردوینو که 5 ولت است، ورود جریان به طور مثال 220 ولت را کنترل کنیم.

رله ها انواع بسیار مختلفی دارند، در اینجا ما از رله با ولتاژ عامل 6 ولت استفاده میکنیم که به ما اجازه میدهد عبور جریان 7 آمپر 250 ولت AC را کنترل کنیم.

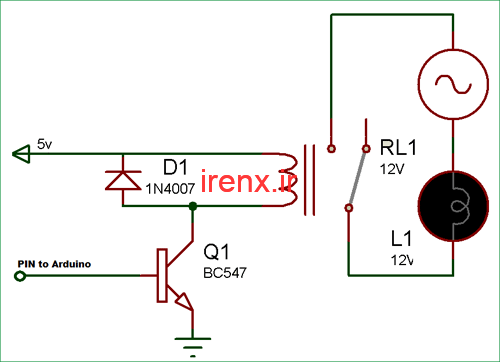
جهت کسب اطلاعات بیشتر، مقاله [رله چیست](https://irenx.ir/electronic/what-is-relay/) را بخوانید.

رله ها اغلب با استفاده از یک مدار درایور کوچک که شامل یک ترانزیستور، دیود و یک مقاومت ترکیب میشوند. ترانزیستور برای تقویت جریان استفاده می شود به طوری که جریان کامل (از منبع DC) می تواند از طریق یک سیم پیچ جریان یابد تا انرژی آن را به طور کامل استفاده کند. از مقاومت برای تهیه بایاس به ترانزیستور استفاده می شود. و هنگام خاموش شدن ترانزیستور از دیود برای جلوگیری از جریان معکوس استفاده می شود.

هر سیم پیچ Inductor با خاموش شدن ناگهانی جریان متضاد و قوی ای تولید میکند بنابراین باید از دیود برای جلوگیری از جریان معکوس استفاده می شود. ماژول های رله با دیود و قطعات لازم به صورت آماده در بازار موجود است اما در صورتی که میخواهید از رله تنها استفاده کنید باید خودتان دیود ، ترانزیستور و … را به آن متصل کنید.

مدار درایور رله به همراه قطعات مورد نیاز آن در تصویر زیر نشان داده شده است.

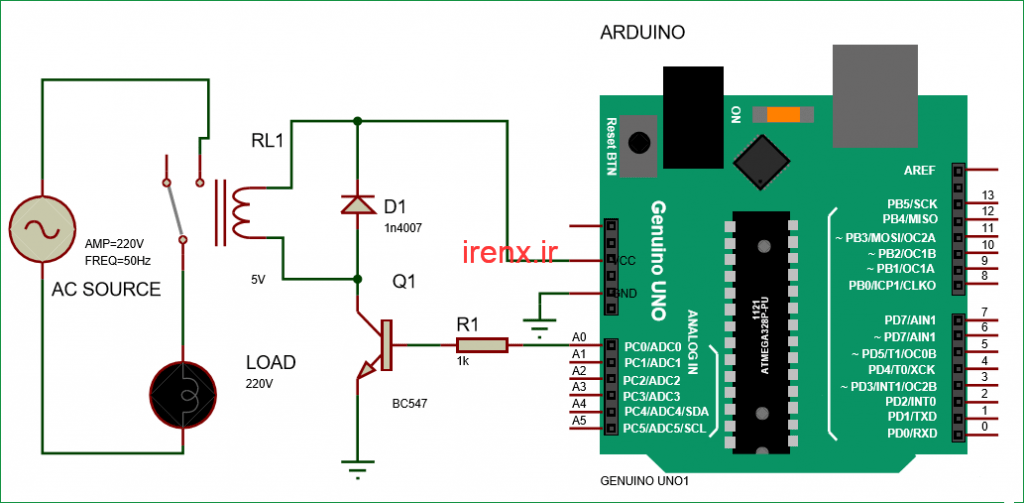
مقاله پیشنهادی: [پروژه تشخیص حرکت با آردوینو و سنسور PIR](https://irenx.ir/arduino/motion-detection-arduino/)



## نحوه اتصال رله به برد آردوینو

در تصویر زیر نحوه اتصال یک رله به برد Arduino را مشاهده میکنید. همانطور که میبینید ما از پین A0 آردوینو برای کنترل رله استفاده کرده ایم.

قطعات مورد نیاز را از [**فروشگاه قطعات آیرنکس**](https://irenx.ir/shop/) تهیه کنید.



در این مدار کنترل رله ما از آردوینو برای کنترل رله از طریق [ترانزیستور](https://irenx.ir/electronic/what-is-transistor/) BC547 استفاده کرده ایم. ما پایه ترانزیستور را از طریق یک مقاومت 1k به pin A0 وصل کرده ایم. یک لامپ AC برای نمایش عملکرد استفاده می شود. از آداپتور 12v برای تغذیه مدار استفاده می شود.

همچنین اگر در مورد این مطلب سوالی داشتید در انتهای صفحه در قسمت نظرات بپرسید

کار ساده است ، ما باید پین RELAY یعنی PIN A0 را فعال کنیم تا ماژول رله روشن شود و پین رله را غیرفعال کنیم تا ماژول رله خاموش شود. چراغ / وسیله AC طبق رله روشن و خاموش می شود.

## برنامه نویسی رله در آردوینو

در این قسمت ما به طور کامل با نحوه برنامه نویسی رله در نرم افزار Arduino آشنا میشویم.

در قسمت اول کد ما پین رله را تعریف میکنیم که همانطور که گفتیم A0. همچنین برای اینکه برنامه از سادگی در بیاید ما یک متغیر تعریف میکنیم به نام interval که نمایان گر 1000 است. کمی جلوتر میفهمید به چه کار می آید.

#define relay A0 // تعریف پین رله

#define interval 1000 // تعریف 1000

سپس به تابع void setup میرسیم، در این قسمت پین رله (A0) را به عنوان پین خروجی تعریف میکنیم.

void setup() {

pinMode(relay, OUTPUT); // تنظیم پین رله به عنوان خروجی

}

سپس به اصل کار میرسیم که تابع void loop است. هرچه در این تابع قرار دهیم پی در پی تکرار میشود. ما در این قسمت کد ها را طوری مینویسیم که رله ما خاموش و روشن شود.

void loop()

{

  digitalWrite(relay, HIGH); // رله روشن

  delay(interval); // به مدت تعریف شده 1 ثانیه

  digitalWrite(relay, LOW); // رله خاموش

  delay(interval); // به مدت تعریف شده 1 ثانیه

}

در قسمت بالا میبینید که در روبروی Delay که همان تاخیر است به جای عدد از کلمه interval استفاده کرده ایم. این کار باعث میشود در برنامه هایی که تعداد زیادی تاخیر شبیه به هم وجود دارد ، تغییر زمان انتظار راحت تر شود.

به طور مثال ما میخواهیم 10 ال ای دی را به فلاشر تبدیل کنیم که هر 1 ثانیه چشمک بزنند. اینجا اگر برای هر ال ای دی ، یک Delay عددی بنویسیم، در صورتی که بخواهیم مقدار تاخیر را به 2 ثانیه تبدیل کنیم باید 10 قسمت از کد را تغییر دهیم. اما اگر مثل اینجا یک کلمه را به جای عدد تعریف کنیم فقط با تغییر عدد تاخیر در اول کد، تاخیر برای هر 10 ال ای دی تغییر میکند.

**کد کامل را در باکس زیر آورده ایم.**

#define relay A0

#define interval 1000

void setup() {

  pinMode(relay, OUTPUT);

}

void loop()

{

   digitalWrite(relay, HIGH);

   delay(interval);

   digitalWrite(relay, LOW);

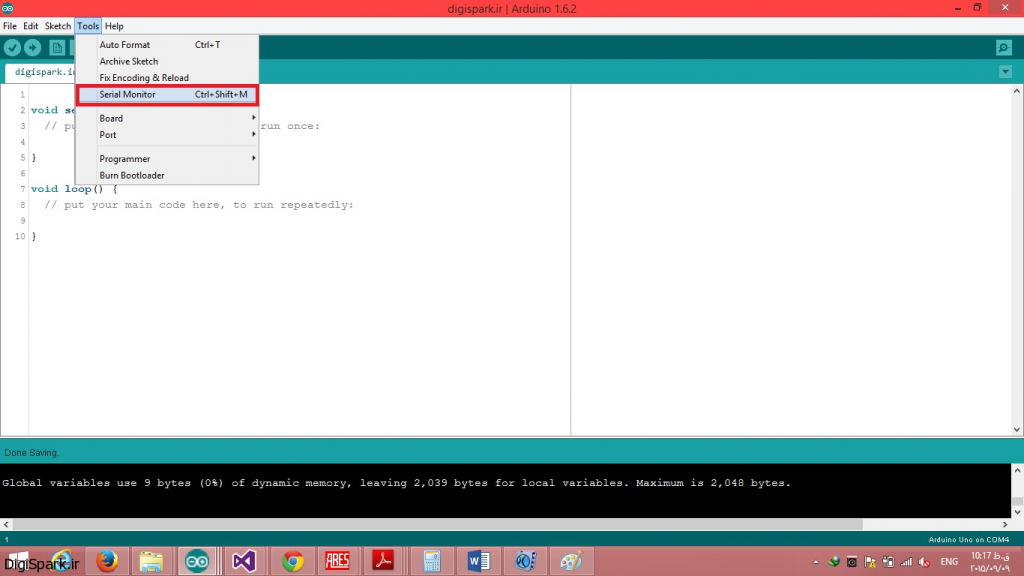
   delay(interval);

}

## آموزش ارتباط سریال آردوینو بخش اول

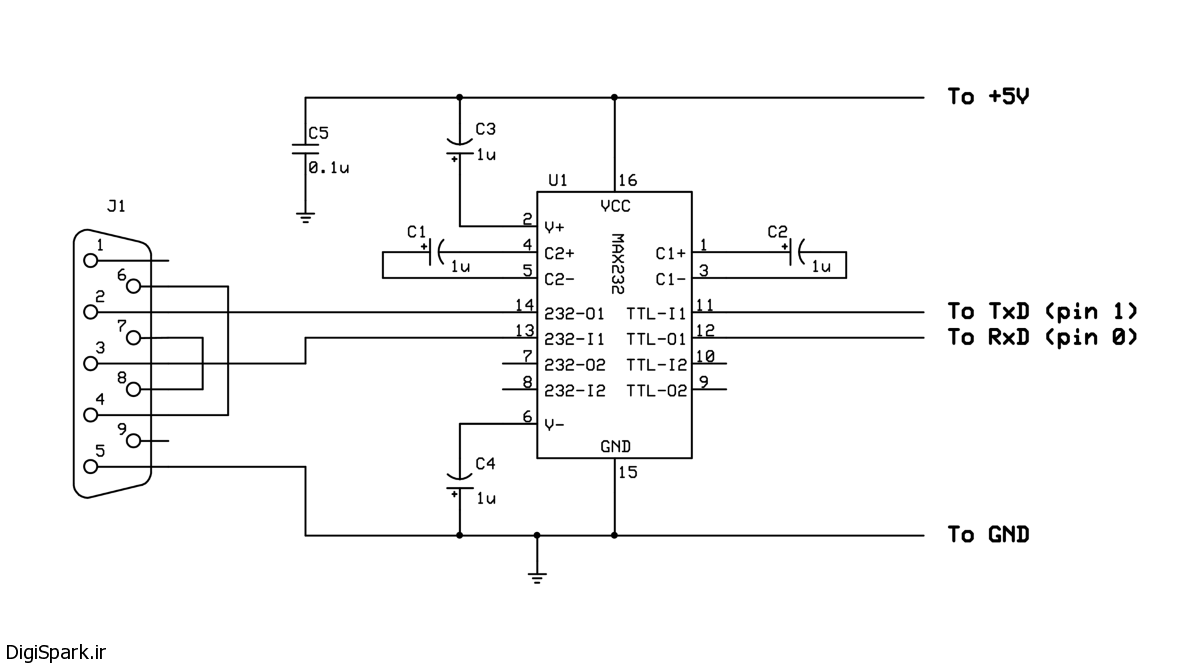
تمام برد های آردوینو دارای حداقل یک عدد  پورت سریال می باشد که  اختصارا به آن ها UART ,USART گفته می‌شود. در آردوینو از پین‌های دیجیتال ۰ و ۱ برای راه اندازی پورت سریال استفاده می‌شود و میتوان به وسیله‌ی درگاه USB بر روی برد ،آردوینو را به کامپیوتر متصل نمود( پین ۰   RX(گیرنده اطلاعات)  و پین ۱   TX (فرستنده اطلاعات )می باشد).هنگامی که از پین های ۰ و ۱ به عنوان پروت سریال استفاده شود ، دیگر نمی‌توان از این پین ها به عنوان ورودی خروجی دیجیتال استفاده نمود.

شما می‌توانید از سریال مانیتوری که در محیط نرم افزار آردوینو در نظر گرفته شده است برای انتقال دیتا میان کامپیوتر و آردوینو استفاده کنید.برای باز کردن سریال مانیتور همانند تصویر زیر عمل کنید.

[](http://digispark.ir/wp-content/uploads/2015/09/serial.png)

ارتباط سریال آردوینو

پس از باز کردن این پنجره در قسمت boude rate ، مقداری که در هنگام استفاده از تابع begin وراد کرده اید را  در این قسمت وارد کنید( این قسمت به طور کامل در طول آموزش توضیح داده می‌شود). آردوینو Uno فقط دارای یک پورت ارتباط سریال می‌باشد که از پین های ۰ و ۱ برای این منظور استفاده شده است. آردوینو سری MEGA علاوه بر پورت سریالی که بر روی پین های ۰ و ۱ دارد ، دارای سه پورت سریال دیگر به نام‌های Serial1 , Serial2 , Serial3 می‌باشد.اگر شما قصد برقراری ارتباط آردوینو با کامپیوتر به وسیله‌ی این سه پورت ( Serial1 , Serial2 , Serial3) را دارید باید از یک مبدل USB TO Serial استفاده کنید.همچنین می‌توانید برای متصل کردن پورت‌های سریال آردوینو به کابل  DB9 ، مطابق مدار زیر از یک آی‌سی MAX232 استفاده کنید.

[](http://digispark.ir/wp-content/uploads/2015/09/max-232-arduino-lg.png)

مدار ای سی max232 برای اتصال آردوینو DB9

[](http://digispark.ir/wp-content/uploads/2015/09/DB9.jpg)

کابل DB9

پلت فرم آردوینو دارای توابع و دستورالعمل های بسیار کاملی برای دریافت و ارسال دیتا توسط پروتکل سریال می‌باشد از این رو ارتباط سریال در آردوینو بسیار ساده می‌باشد .در ادامه به شرح دستور عمل‌های مربوط به پروتکل ارتباط سریال در آردوینو می‌پردازیم.

## برنامه نویسی آردوینو

#### جهت دسترسی به این بخش، اشتراک [اسپارکر پلاس](https://digispark.ir/sparker-plus-register/) تهیه کنید.

مثال:فرض می‌کنیم از این کد قطعه کد برای تنظیم کردن پورت  سریال  استفاده کرده ایم و قصد داریم کاراکتر رشته OK را با استفاده از پورت سریال به کامپیوتر منتقل کنیم:



|  |  |
| --- | --- |
| 1 | Serial.begin(9600 , SERIAL\_8N1 ); |

این کد به این معناست که از بادریت ۹۶۰۰ و از مقدار SERIAL\_8N1 به عنوان config   استفاده کرده ایم .

به این علت که کاراکتر O و K هر کدام ۸ بیت و همچنین در کانفیگ از  SERIAL\_8N1 که به معنای۸ بیت داده ، بدون بیت توازن  و ۱ عدد stop bit می‌باشد ، باید در ۲ بسته اطلاعات را  ارسال کنیم . هم اکنون به شرح ارسال کاراکتر O می‌پردازیم. کاراکتر O دارای کد اسکی ۷۹  می باشد و در مبنای دو دارای مقدار ۰۱۰۰۱۱۱۱ می‌باشد.

بنابراین نقشه ی ارسال کاراکتر O به صورت زیر می‌باشد.

[](http://digispark.ir/wp-content/uploads/2015/09/oooooooooo.png)

نقشه‌ی ارسال کاراکتر O

## end()

ارتباط سریال را قطع می‌کند و اجازه میدهد که پین هایی که بر روی حالت ارتباط سریال قرار داشتند ، بتوانند به صورت ورودی خروجی دیجیتال مورد استفاده قرار بگیرند .  این تابع دارای ساختار مانند قطعه کد زیر می‌باشد :



|  |  |
| --- | --- |
| 1 | Serial.end(); |

برای راه اندازی مجدد می‌توان از تابع **Begin**استفاده نمود.

## print()

این تابع دیتا را به پورت سریال ارسال می‌کند.تابع **print**دارای فرم های مختلفی می‌باشد.این تابع اعداد صحیح  ، متن ، کاراکتر را به صورت رشته اعداد اعشاری را به صورت رشته  با دو عدد اعشار ارسال می‌کند.به مثال زیر توجه کنید:



|  |  |
| --- | --- |
| 1  2  3  4 | Serial.print( 78 );// gives "78"  Serial.print(1.23456);// gives "1.23"  Serial.print('N');// gives "N"  Serial.print("Hello world.");// gives "Hello world." |

این تابع دارای دو ساختار به صورت زیر می‌باشد:



|  |  |
| --- | --- |
| 1  2 | Serial.print( val);  Serial.print( val, format); |

**val :**دیتایی می باشد که می‌خواهیم آن را به پروت سریال ارسال کنیم

**format :**فرمت نمایش و تعداد ممیز اعشار نمایش داده را مشخص می‌کند.

برای ارسال اطلاعات به فرمت باینری ( مبنای ۲ )  از BIN ، مبنای ۸ از OCT ، مبنای ۱۰ از DEC و مبنای ۱۶ از HEX  استفاده می‌کنیم. توجه شما را به مثال زیر جلب می‌کنم :



|  |  |
| --- | --- |
| 1  2  3  4 | Serial.print(78, BIN); // gives "1001110"       مبنای 2  Serial.print(78, OCT); // gives "116"           مبنای 8  Serial.print(78, DEC); // gives "78"            مبنای 10  Serial.print(78, HEX); // gives "4E"            مبنای 16 |

همچنین می‌توان تعداد ممیز اعشار را توسط format تعیین نمود.برای این کار کافی است تنها تعداد اعشار مورد نیاز را به جای format قرار دهیم همانند مثال زیر:



|  |  |
| --- | --- |
| 1  2  3 | Serial.print(1.23456, 0); // gives "1"  Serial.print(1.23456, 2); // gives "1.23"  Serial.print(1.23456, 4); // gives "1.2346" |

## println()

این تابع همانند تابع   **print** عمل می‌کند با این تفاوت که با هر بار ارسال اطلاعات دستور رفتن به خط بعد را نیز اجرا می‌کند.

به زبانی ساده تر دو نمونه کد زیر با هم یکسان می‌باشند:



|  |  |
| --- | --- |
| 1  2 | Serial.print("");    Serial.print('n'); |



|  |  |
| --- | --- |
| 1 | Serial.println(""); |

## available()

به وسیله‌ی این تابع تعداد کاراکتر هایی که آماده خواندن از پورت سریال می‌باشند مشخص می‌شود.همچنین این تابع این امکان را فراهم می‌کند که ۶۴ بایت در بافر دریافت کننده ی پورت سریال ذخیره شود. این تابع تعداد کاراکتر های موجود در بافر را بر می‌گرداند. ساختار کلی تابع به صورت زیر می‌باشد:



|  |  |
| --- | --- |
| 1 | Serial.available(); |

به مثال زیر توجه کنید:



|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18 | int incomingByte = 0;   // for incoming serial data    void setup() {          Serial.begin(9600);     // opens serial port, sets data rate to 9600 bps  }    void loop() {            // send data only when you receive data:          if (Serial.available() > 0) {                  // read the incoming byte:                  incomingByte = Serial.read();                    // say what you got:                  Serial.print("I received: ");                  Serial.println(incomingByte, DEC);          }  } |

در این مثال اگر اطلاعاتی به پورت سریال فرستاده شد **Serial.available**مقدار true را برمی‌گرداند و برنامه وارد شرط شده و عملیاتی را انجام می‌دهد.برنامه همچنان ادامه پیدا می‌کند و اگر در بافر سریال اطلاعی برای خواندن وجود داشت ، دوباره برنامه وارد شرط می‌شود.توصیه می‌شود در برنامه نویسی حتما از این تابع استفاده کنید.

## read()

به وسیله‌ی این تابع دیتایی که بر روی پورت سریال قرار دارد خوانده می‌شود.این تابع اگر دیتایی دریافت نکند مقدار ۱- و اگر دیتایی دریافت کند ، همان دیتا را برمی‌گرداند.

ساختار کلی تابع به صورت زیر می‌باشد:



|  |  |
| --- | --- |
| 1 | Serial.read(); |

مثال:



|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21 | void loop()  {    if (Serial.available())    {      char ch = Serial.read();      if (ch >= '0' && ch <= '7')      {        int led = ch - '0';        bitSet(leds, led);        updateShiftRegister();        Serial.print("Turned on LED ");        Serial.println(led);      }      if (ch == 'x')      {        leds = 0;        updateShiftRegister();        Serial.println("Cleared");      }    }  } |

در مثال بالا به وسیله‌ی تابع  available برنامه به گونه‌ای نوشته شده است که فط در هنگام ارسال  دیتا وارد پردازش شده و عملیات انجام می‌دهد و در صورتی که دیتایی بر روی پورت سریال دریافت نشود برنامه در عمل بدون پردازش می‌باشد.درصورتی که دیتایی دریافت شود تابه available مقدار true را برمی‌گرداند و برنامه وارد شرط شده و مابقی دیتاهایی که به درحال ارسال به سمت گیرنده هستند به وسیله تابع available در بافر گیرنده پورت سریال قرار می‌گیرند.سپس توسط قطعه کد زیر



|  |  |
| --- | --- |
| 1 | char ch = Serial.read(); |

دیتای دریافتی درون متغیر ch ذخیره می‌شود و بعد از خوانده شدن توسط تابع **read** این دیتا از بافر گیرنده پورت سریال پاک می‌شود  و در ادامه نیز این عملیات در صورت وجود دیتا در بافر سریال تکرار می‌شود.

## Serial.readBytes()

به وسیله‌ی این تابع مقادیری از جنس بایت یا کاراکتر از روی بافر پورت سریال خوانده می‌شود و درون یک آرایه ( بافر ) دخیره می‌شود و تعداد بایت دریافتی را برمی‌گرداند

این تابع دارای ساختار زیر است:



|  |  |
| --- | --- |
| 1 | Serial.readBytes(buffer, length); |

**lenght :**تعداد بایتی است که قرار است دریافت شود.

**buffer:**نام آرایه‌ای که قرار است اطلاعات خوانده شده از پورت سریال درون آن دخیره شود.این آرایه می‌تواند از جنس []char یا []byte باشد

این تابع هنگامی که به مقدار lenght دیتا دریافت کند عملیات خواندن از پورت سریال را متوقف می‌کند.

به مثال زیر توجه کنید :



|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21 | char buffer[3];  int lenght;  void setup()  {    Serial.begin(9600);    }      void loop()  {    if (Serial.available() > 0)    {      lenght = Serial.readBytes(buffer , 3);      for (int i = 0 ; i <= lenght- 1 ; i++)      {        Serial.print(buffer[i]);      }      Serial.println();    }  } |

در این مثال برنامه به شروع خواندن پورت سریال می‌کندو هنگامی که ۳ بایت داده دریافت کرد متوقف شده و سه بایت دریافتی را چاپ می‌کند و دوباره به عملیات خود ادامه می‌دهد.

## Serial.readBytesUntil()

دارای ساختار زیر می‌باشد :



|  |  |
| --- | --- |
| 1 | Serial.readBytesUntil(character, buffer, length); |

**lenght :**تعداد بایت مورد نظر برای دریافت

**character :**کاراکتری که به محض  دیدن آن ، عملیات خواندن را متوقف می‌کند.

**buffer :**آرایه‌ای از جنس []char یا []byte

این تابع عمل کردی مشابه تابع **readBytes**دارد با ایت تفاوت که  دیتا را از بافر پورت سریال می‌خواند و به محض رسیدن به یک کاراکتر  خاص ( character ) و  یا تعداد تعیین شده‌ی بایت ( length )  عملیات خواندن را متوقف نموده و تعداد بایت دریافت شده قبل از کاراکتر مورد نظر را برمی‌گرداند ( اگر دیتایی موجود نباشد عدد ۰ را برمی‌گرداند )  .

به عنوان مثال در قطعه کد زیر برنامه شروع به خواندن از بافر پورت سریال می‌کند و هنگامی که به کاراکتر  e  رسید و یا هنگامی که تعداد بایت دریافتی به ۸ عدد رسید عملیات خواندن را متوقف می‌کند . و دیتای دریافتی را بر روی سریال مانیتور نرم افزار نمایش می‌دهد.



|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21 | char buffer[8];  int lenght;  void setup()  {    Serial.begin(9600);    }      void loop()  {    if (Serial.available() &gt; 0)    {      lenght = Serial.readBytesUntil('e',buffer , 8);      for (int i = 0 ; i &lt;= lenght- 1 ; i++)      {        Serial.print(buffer[i]);      }      Serial.println();    }  } |

در قسمت بعدی آموزش به شرح توابعی دیگر از کتابخانه پورت سریال arduino خواهیم پرداخت . با [دیجی اسپارک](http://digispark.ir/) ، بزرگترین [مرجع تخصصی آموزش آردوینو](http://daneshjookit.com/) در سراسر کشور همراه باشید.

# آموزش جامع آردوینو جلسه ۱

 [مهدی دمیرچیلو](https://dmf313.ir/author/dmf313/)

  [2015/07/03](https://dmf313.ir/2015/)

  [Arduino train](https://dmf313.ir/arduino/arduino-train/)

* 

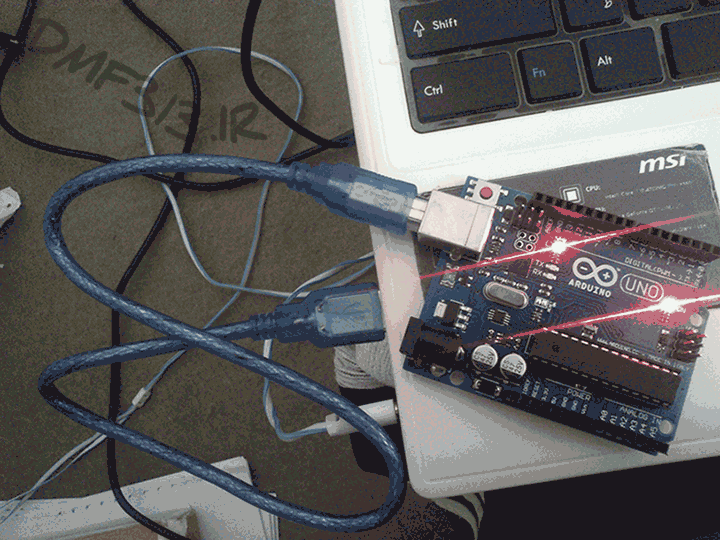
[](https://www.eca.ir/forums/)

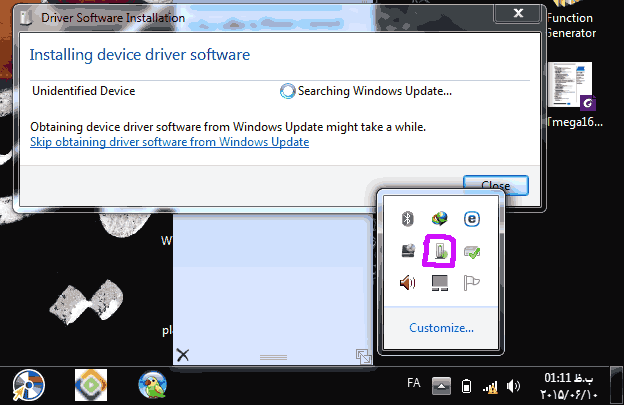
### به نام خدا

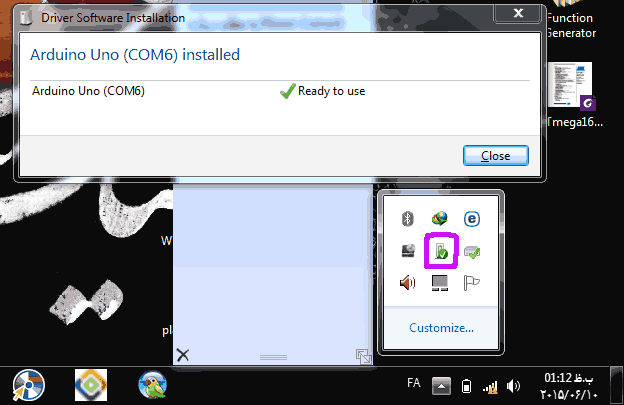
**آموزش جامع آردوینو جلسه ۱ :** سلام خدمت شما دوستان گل ، تو این مطلب از [آموزش آردوینو](https://dmf313.ir/electronics/arduino/arduino-train/) که جلسه اولش هم هست (و انشاالله تا آخرش میریم) میخوام به [توضیح توابع آردوینو](https://dmf313.ir/electronics/arduino/arduino-train/) بپردازم و یه [پروژه آردوینو](https://dmf313.ir/electronics/arduino/arduino-projects/) هم براتون گزاشتم.  
[](https://dmf313.ir/wp-content/uploads/arduino1.jpg)  
**توجه :** من از برد UNO آردینو استفاده میکنم ، به شما هم پیشنهاد میکنم همین برد رو بخرید.(از ما گفتن cool) شماتیک های پروژه رو هم که میبنید همین برد آردینو UNO هستش.  
[](https://dmf313.ir/wp-content/uploads/11021b-00.jpg)

اولین باری که آردوینو رو به سیستم وصل میکنیم ^\_^

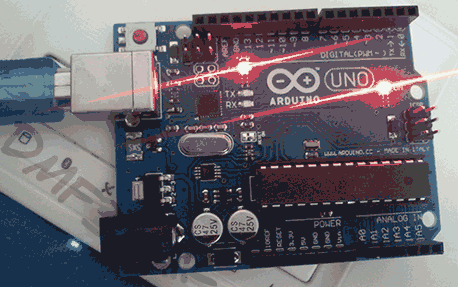
بنا به درخواست یکی از دوستان این قسمت رو اضافه میکنم blush  
اولین باری که آردوینو رو به کامپیوتر/لبتاب وصل میکنید دنبال درایور برد مورد نظر میگرده و پیداش میکنه و نصبش میکنه، اگه به این ترنت وصل باشید که از نت دانلود میکنه درایور رو و خودش نصب میکنه و اگه وصل نبودید به اینترنت و درایور هم نصب نشد کارتون یکم سخت میشه و باس بگیردید و درایورش رو پیدا کنید و نصب کنید diabloکه تولید لینک مقابل(گوگل) میتونید بگردید و پیدا کنید درایور مورد نظر رو : [**دانلود درایور uno**](https://www.google.com/search?q=%D8%AF%D8%A7%D9%86%D9%84%D9%88%D8%AF+%D8%AF%D8%B1%D8%A7%DB%8C%D9%88%D8%B1+uno&oq=%D8%AF%D8%A7%D9%86%D9%84%D9%88%D8%AF+%D8%AF%D8%B1%D8%A7%DB%8C%D9%88%D8%B1+uno&aqs=chrome..69i57.2614j0j7&sourceid=chrome&es_sm=93&ie=UTF-8)

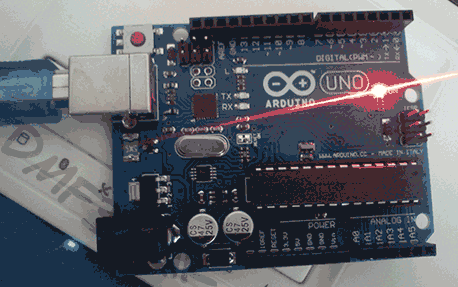
**۱) وصل کردن آردوینو uno به کامپیوتر/لبتاب**  
[](https://dmf313.ir/wp-content/uploads/0.png)

**۲) وای میستیم(میشینیم، صبر میکنیم و... laugh) تا درایور مطابق دو شکل زیر نصب بشه.(اول دنبال درایور میگرده و بعد نصبش میکنه....الله اکبر...عجب جمله خفنی laugh....بهتره به اینترنت وصل باشید.)**  
[](https://dmf313.ir/wp-content/uploads/130.png)

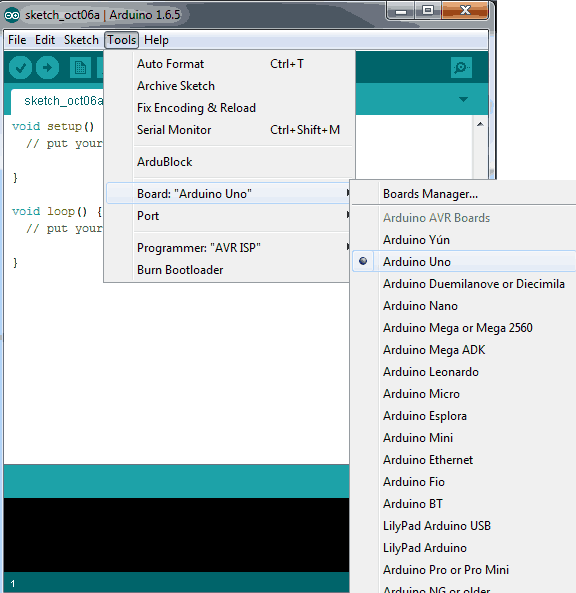
[](https://dmf313.ir/wp-content/uploads/213.png)

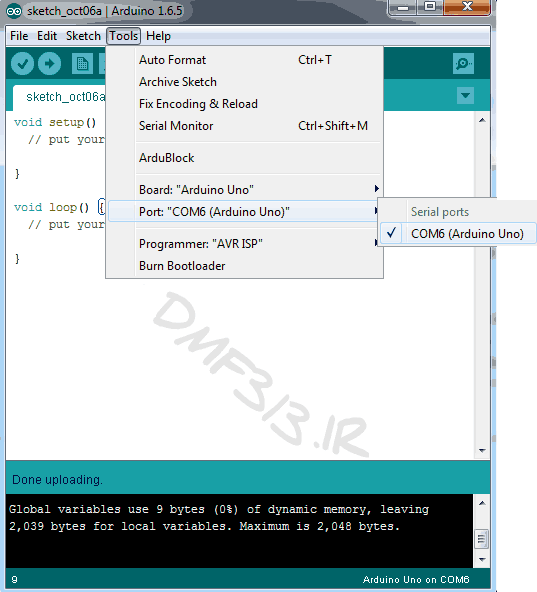
**۳) خوب درایور آردوینو هم نصب شد**، الان میبینید که تو برد آردوینو ۲ تا led داریم که یکیش برا تغزیه هستش فک کنم و همیشه روشنه و یکی از led ها به پایه ۱۳ آردوینو وصل هستش و تقریبا هر ۱ ثانیه یکبار خاموش روشن میشه.(که یعنی از قبل یه برنامه چشمک زدن led رو ریختن تو برد آردوینو تا مطمئن بشن و شما هم مطمئن بشید که برد سالمه، البته ما که شانس نداریم یه وقت دیدی اون led هه چشمک نمیزنه، در این حالت باید یه برنامه خودتون بریزید تو آردوینو و تست کنید که در ادامه میگم....خیالتون راحت...حواتون رو دارم heart)----این چشمک زدن اون led هه رو در زیر با ۲ تا عکس نشون میدم بهتون.

[](https://dmf313.ir/wp-content/uploads/on.png)

[](https://dmf313.ir/wp-content/uploads/off.png)

**۴) ریختن یه برنامه ساده تو آردوینو** و تست کردن و نحوه ریختن برنامه داخل آردوینو، نحوه آپلود کردن کدها داخل آردوینو، نحوه پروژگرام کردن آردوینو، برنامه تست برای آردوینو و...(تمام عناوینی که میشید به این قسمت انتصاب داد رو فک کنم گفتم crazy)  
خب بعد این که **نرم افزار آردوینو** رو دانلود کردید، اجراش میکنید([**دانلودنرم افزار آردوینو**](https://www.arduino.cc/en/Main/Software)) و طبق مراحل زیر پیش میرید.

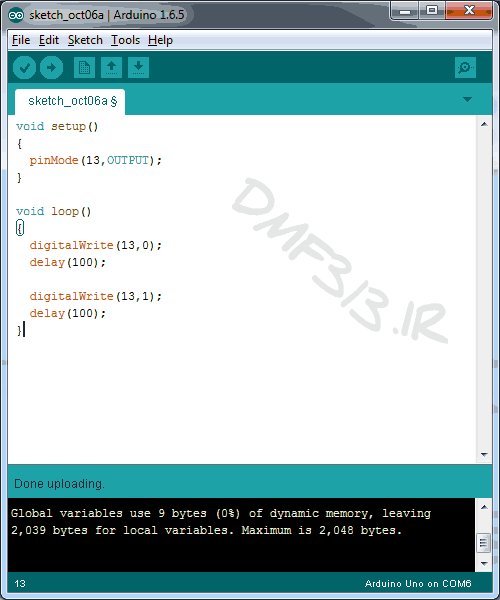
**۴٫۱)** ابتدا باید برد آردوینو مورد نظر رو انتخاب کنیم.  
[](https://dmf313.ir/wp-content/uploads/Untitled-127.png)

**۴٫۲)** بعد باید پورت مورد نظر رو انتخاب کنید(حالا ممکنه این پورت برا کامپیوتر من پورت ۵ باشه و برا شما پورت ۷ بشه و برا یکی دیگه متفاوت باشه، که زیاد مهم نیست.)  
[](https://dmf313.ir/wp-content/uploads/Untitled-214.png)

**۴٫۳)** حالا کدهای زیر رو به نرم افزار اضافه میکنیم(در محیط نرم افزار قرار میدیم.)، فعلا به این که این کد ها چی هستش و توضیحات و... گیر ندید در ادامه و مطلب بعد قشنگ متوجه میشید، تنها چیزی که باید بدونید و باهاش فعلا ور برید کد **;(delay(100**  هستش، که اون عدد داخل پرانتز رو تغییر بدی زمان روشن و خاموش شدن LED روی برد آردوینو تغییر میکنه، و از این طریق هم یه برنامه نمونه ریختید تو بردتون و از سالم بودنش مطمئن شدید و هم نحوه پروگرام کردن رو یاد گیرفتید و هم ...دیگه چیزی به فکرم نمیرسه focus، فقط حواستون باشه که **;(delay(100**در کد های زیر ۲ بار نوشته شده که اولی زمان خاموش بودن و دومی زمان روشن بودن LED هستش.

Arduino

|  |
| --- |
|  |
| void setup()  {    pinMode(13,OUTPUT);  }  void loop()  {    digitalWrite(13,0);    delay(100);    digitalWrite(13,1);    delay(100);  } | |

[](https://dmf313.ir/wp-content/uploads/Untitled-36.png)

**۴٫۴)** بعد از انجام کارهایی که در بالا گفتم، پروژتون رو در یه جایی ذخیره میکنید(وای بحالتون اگه بفهمم نحوه ذخیره کردن رو بلد نیستید. bomb) حالا روی دکمه ای که به شکل فلش هستش کلیک میکنید(شکل زیر، اون دکمه ای که سفید رنگ هستش) تا برنامه کد هاتون رو بررسی کنه(و اگه مشکلی بود بهتون بگه) و بعد که نرمافزار دید کد هاتون مشکلی نداره میاد و اونو میریزه تو برد و شما اولین پروژتون رو میبینید و کیف میکنید(مثل من هنگام بستن اولین پروژه با آردوینو blush)  
[نرم افزار آردوینو](https://dmf313.ir/wp-content/uploads/Untitled-52.png)

**۴٫۵)** در این مرحله پروژتون رو میبینید laugh، و اگه کار کرد من رو دعا میکنید و اگه کار نکرد بازم منو دعا میکنید، فقط با این تفاوت که در قسمت نظرات مشکلتون رو هم میگید تا با هم، دو نفری مشکل رو حلش کنیم.

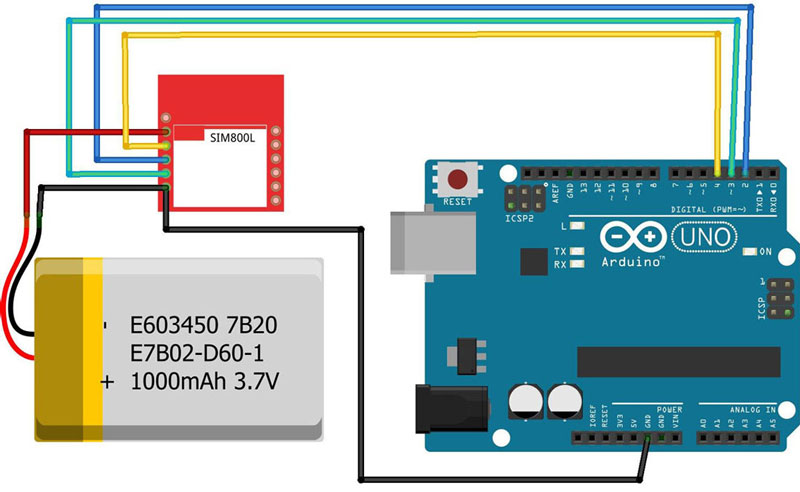
# راه اندازی ماژول SIM800L با آردوینو UNO

توسط [ادمین 2](https://118elec.com/author/elham-h/)

[8 اردیبهشت 1400](https://118elec.com/%d8%b1%d8%a7%d9%87-%d8%a7%d9%86%d8%af%d8%a7%d8%b2%db%8c-%d9%85%d8%a7%da%98%d9%88%d9%84-sim800l-%d8%a8%d8%a7-%d8%a2%d8%b1%d8%af%d9%88%db%8c%d9%86%d9%88/)

در [پروژه آردوینو](https://118elec.com/category/%d9%be%d8%b1%d9%88%da%98%d9%87-%d8%a2%d8%b1%d8%af%d9%88%db%8c%d9%86%d9%88/)

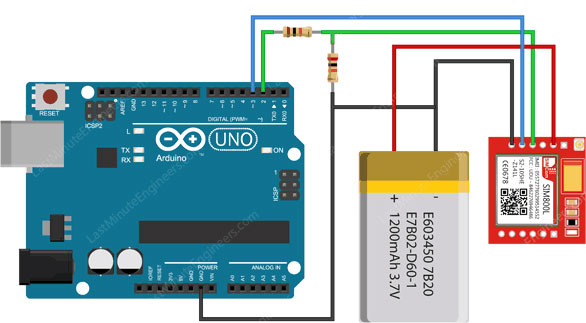
[7](https://118elec.com/%d8%b1%d8%a7%d9%87-%d8%a7%d9%86%d8%af%d8%a7%d8%b2%db%8c-%d9%85%d8%a7%da%98%d9%88%d9%84-sim800l-%d8%a8%d8%a7-%d8%a2%d8%b1%d8%af%d9%88%db%8c%d9%86%d9%88/#comments)

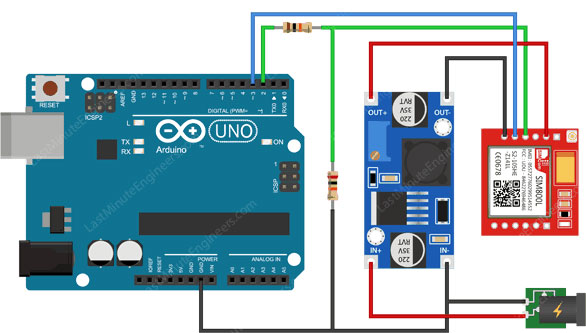
[](https://118elec.com/wp-content/uploads/2021/01/804ef97ac311ccdb7eea54763ea00f27.jpg)

برای راه اندازی ماژول SIM800L با آردوینو ، ابتدا باید آنتن را به ماژول GSM وصل کرد. سپس سیم کارت میکرو فعال شده را در سوکت سیم کارت قرار داد. با اتصال پین TX ماژول به پین 3 آردوینو می توان از طریق مانیتور سریال آردوینو با ماژول ارتباط برقرار کرد.

باید به این نکته توجه شود که نمی توان پین Rx ماژول را به صورت مستقیم به پین دیجیتال آردوینو وصل کرد. زیرا ولتاژ روی پایه های GPIO در آردوینو Uno پنج ولت است، در حالیکه ماژول SIM800L با 3.3 ولت کار می کند و تحمل 5 ولت را ندارد. بنابراین سطح ولتاژی که از آردوینو به ماژول وارد می شود باید به 3.3 کاهش یابد تا به ماژول آسیبی نرسد. چندین راه حل برای کاهش سطح سیگنال  وجود دارد که ساده ترین آن استفاده از یک تقسیم مقاومتی ساده است. می توان یک مقاومت 10 کیلو اهم بین پین Rx ماژول SIM800L GSM و پین 2 آردوینو Uno و همچنین یک مقاومت 20 کیلو اهم بین پین RX ماژول و زمین قرار داد.

حال باید پین های منبع تغذیه وصل شوند. همانطور که در [مقاله معرفی ماژول SIM800L GSM](https://118elec.com/معرفی-ماژول-sim800l-gsm/) توضیح داده شد، می توان برای تغذیه ماژول به دو روش عمل کرد که در شکل های زیر شماتیک مدار و نحوه اتصال آن آورده شده است. در شکل اول از باتری Li-Po  با ظرفیت 1200mAh و در شکل  دوم از مبدل کاهنده باک LM2596 برای تغذیه ماژول استفاده شده است.

[](https://118elec.com/wp-content/uploads/2021/01/Arduino-Wiring-Fritzing-Connections-with-SIM800L-GSM-GPRS-Module-3.jpg)راه اندازی ماژول SIM800l با آردوینو UNO و استفاده از باتری Li-Po 3.7 ولت برای تغذیه ماژول

[](https://118elec.com/wp-content/uploads/2021/01/Arduino-Wiring-Fritzing-Connections-with-SIM800L-GSM-GPRS-Module-LM2596.jpg)راه اندازی ماژول SIM800l با آردوینو UNO و استفاده از مبدل کاهنده باک برای تغذیه ماژول

**توجه:** **هنگامیکه از مبدل کاهنده LM2596 به عنوان منبع تغذیه ماژول استفاده می شود، زمین های مدار باید به یکدیگر متصل باشند.**

هنگامیکه تمام اتصالات انجام شد، می توان آردوینو را برای ارسال و دریافت تماس و پیامک برنامه ریزی کرد.

## بررسی دستوراتAT برای راه اندازی ماژول SIM800l با آردوینو

برای ارسال دستورات AT و ارتباط با ماژول SIM800L از مانیتور سریال نرم افزار آردوینو استفاده می شود. کد زیر برای ارتباط آردوینو با ماژول SIM800L بر روی مانیتور سریال ، نوشته شده است. . برای برنامه ریزی آردوینو ، ابتدا باید آردوینو به کامپیوتر وصل شود و سپس کد زیر اجرا و در آردوینو آپلود شود.

لازم به ذکر است که در پنجره مانیتور سریال  ، باید گزینه NL&CR انتخاب شود. حال به شرح قسمت های مختلف کد پرداخته می شود.

#include < SoftwareSerial.h >

//Create software serial object to communicate with SIM800L

SoftwareSerial mySerial(3, 2); //SIM800L Tx & Rx is connected to Arduino #3 & #2

void setup()

{

//Begin serial communication with Arduino and Arduino IDE (Serial Monitor)

Serial.begin(9600);

//Begin serial communication with Arduino and SIM800L

mySerial.begin(9600);

Serial.println("Initializing...");

delay(1000);

mySerial.println("AT"); //Once the handshake test is successful, it will back to OK

updateSerial();

mySerial.println("AT+CSQ"); //Signal quality test, value range is 0-31 , 31 is the best

updateSerial();

mySerial.println("AT+CCID"); //Read SIM information to confirm whether the SIM is plugged

updateSerial();

mySerial.println("AT+CREG?"); //Check whether it has registered in the network

updateSerial();

}

void loop()

{

updateSerial();

}

void updateSerial()

{

delay(500);

while (Serial.available())

{

mySerial.write(Serial.read());//Forward what Serial received to Software Serial Port

}

while(mySerial.available())

{

Serial.write(mySerial.read());//Forward what Software Serial received to Serial Port

}

}

در ابتدای کد کتابخانه SoftwareSerial.h برای ارتباط سریال میان پین های آردوینو و پورت های Rx /Tx ماژول SIM800L افزوده  شده است.

#include < SoftwareSerial.h >

//Create software serial object to communicate with SIM800L

SoftwareSerial mySerial(3, 2); //SIM800L Tx & Rx is connected to Arduino #3 & #2

در تابع setup ، تنظیمات اولیه ارتباط سریال بین آردوینو، محیط آردوینو IDE و ماژول SIM800L با نرخ ارسال 9600 آورده شده است.

//Begin serial communication with Arduino and Arduino IDE (Serial Monitor)

Serial.begin(9600);

//Begin serial communication with Arduino and SIM800L

mySerial.begin(9600);

پس از انجام تنظیمات اولیه ، با ارسال دستورات AT به ماژول، ارتباط برقرار می شود. همچنین ارسال بصورت خودکار است. در صورتیکه کاراکترهای AT دیده شوند به این معناست که کد به درستی اجرا شده است. سپس کد دستوری برای درخواست ارتباط با ماژول داده می شود که حاوی اطلاعات زیر است:

دستور AT : پایه ترین دستور AT است که با ارسال آن و دریافت OK از سمت ماژول می توان از برقراری ارتباط بین ماژول و آردوینو اطمینان یافت. سپس می توان دستوراتی را مانند دستورات زیر، برای ماژول ارسال کرد و اطلاعاتی را در مورد آن بدست آورد:

دستور AT+CSQ : شدت سیگنال را بررسی می کند. عدد اول، شدت سیگنال به دسی بل را بیان می کند که باید مقدارش از 5 بیشتر باشد. هر چه این مقدار بیشتر باشد، بهتر است. البته این مقدار به نوع آنتن مورد استفاده و موقعیت مکانی ماژول بستگی دارد.

دستور AT+CCID : با ارسال این دستور و دریافت OK می توان از فعال بودن سیم کارت اطمینان پیدا کرد. به علاوه با این دستور شماره سیم کارت قابل دریافت است.

دستور ?AT+CREG : بررسی می کند که آیا سیم کارت در شبکه ثبت شده یا خیر. عدد دوم که در پاسخ به این دستور دریافت می شود، نوع شبکه ای که سیم کارت در آن ثبت شده است را نشان می دهد. این عدد می تواند 1 یا 5 باشد. عدد 1 نشان دهنده شبکه خانگی و عدد 5 نشان دهنده شبکه رومینگ است. هر عدد دیگری به جز 1 و 5 ، نشان می دهد که سیم کارت در هیچ شبکه ای ثبت نشده است.

delay(1000);

mySerial.println("AT"); //Once the handshake test is successful, it will back to OK

updateSerial();

mySerial.println("AT+CSQ"); //Signal quality test, value range is 0-31 , 31 is the best

updateSerial();

mySerial.println("AT+CCID"); //Read SIM information to confirm whether the SIM is plugged

updateSerial();

mySerial.println("AT+CREG?"); //Check whether it has registered in the network

updateSerial();

در تابع loop ، تابع ()updateSerial قرار دارد که دائماً منتظر ورودی های مانیتور سریال است تا آن را از طریق پین 2 (پین Rx ماژول) به ماژول SIM800L ارسال کند. همچنین دائماً پین D3 (پین Tx ماژول) را چک می کند تا اگر ماژول پاسخی به ورودی داشته باشد، آن را در مانیتور سریال نشان دهد.

void updateSerial()

{

delay(500);

while (Serial.available())

{

mySerial.write(Serial.read());//Forward what Serial received to Software Serial Port

}

while(mySerial.available())

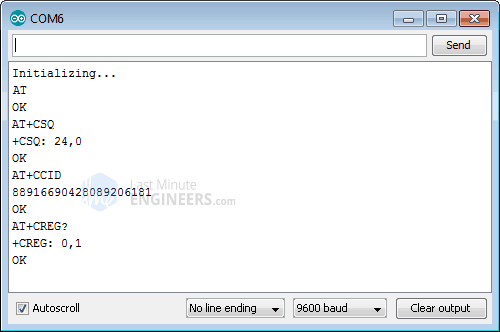
{

Serial.write(mySerial.read());//Forward what Software Serial received to Serial Port

}

}

با اجرای کد، شکل زیر بر روی مانیتور سریال دیده می شود.

دستورات پایه AT برای ارتباط با ماژول SIM800

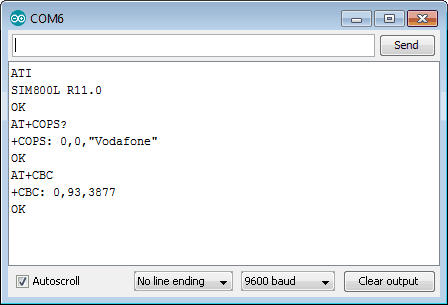
علاوه بر دستوراتی که بیان شد، دستورات دیگری نیز می توان برای دریافت  اطلاعات بیشتری در مورد اتصال شبکه و وضعیت باتری از طریق مانیتور سریال به ماژول ارسال کرد. در ادامه چند دستور آورده شده است.

دستور ATI : نام و نسخه ماژول را می دهد.

دستور ?AT+COPS : اتصال به شبکه را بررسی می کند. .

دستور AT+COPS=? : لیست اُپراتورهای موجود در شبکه را نشان می دهد.

دستور AT+CBC : وضعیت باتری Li-Po را نشان می دهد. عدد دوم میزان شارژ باتری (که در اینجا 93%) و عدد سوم ولتاژ واقعی به میلی ولت (در اینجا 3.877 ولت) است.

دستورات AT برای اتصال شبکه در ماژول SIM800L GSM

## ارسال پیام با ماژول SIM800L GSM

پس از آماده سازی آردوینو و SIM800 ، می توان ماژول آردوینو را برای ارسال پیام به شماره موبایل مورد نظر کد نویسی کرد. قبل از کد نویسی باید شماره موبایل را وارد کرد. در رشته ZZxxxxxxxxxx ، کد کشور باید جایگزین ZZ و شماره 10 رقمی به جای xxxxxxxxxx قرار گیرد.

#include < SoftwareSerial.h >

//Create software serial object to communicate with SIM800L

SoftwareSerial mySerial(3, 2); //SIM800L Tx & Rx is connected to Arduino #3 & #2

void setup()

{

//Begin serial communication with Arduino and Arduino IDE (Serial Monitor)

Serial.begin(9600);

//Begin serial communication with Arduino and SIM800L

mySerial.begin(9600);

Serial.println("Initializing...");

delay(1000);

mySerial.println("AT"); //Once the handshake test is successful, it will back to OK

updateSerial();

mySerial.println("AT+CMGF=1"); // Configuring TEXT mode

updateSerial();

mySerial.println("AT+CMGS=\"+ZZxxxxxxxxxx\"");//change ZZ with country code and xxxxxxxxxxx with phone number to sms

updateSerial();

mySerial.print("Last Minute Engineers | lastminuteengineers.com"); //text content

updateSerial();

mySerial.write(26);

}

void loop()

{

}

void updateSerial()

{

delay(500);

while (Serial.available())

{

mySerial.write(Serial.read());//Forward what Serial received to Software Serial Port

}

while(mySerial.available())

{

Serial.write(mySerial.read());//Forward what Software Serial received to Serial Port

}

}

این کد تقریبا شبیه کد قسمت قبل است. در اینجا به محض اینکه اتصال برقرار شود، دستورات زیر ارسال می شوند:

دستور AT+CMGF=1 : قالب پیام را از نوع “متن” قرار می دهد. قالب پیشفرض PDU (پروتکل انتقال داده) است.

دستور AT+CMGS=+ZZxxxxxxxxxx : پیام را به شماره مشخص شده ارسال می کند. پیامی که به دنبال آن کلید ‘Ctrl+z’ وارد شود به عنوان یک پیام متنی در نظر گرفته می شود. کلید ‘Ctrl+z’ بیست و ششمین کاراکتر جدول ASCII است. بنابراین، لازم است که هنگام ارسال پیام متنی 26 دسیمال یا 1A هگز ارسال شود.

println("AT+CMGF=1"); // Configuring TEXT mode

updateSerial();

mySerial.println("AT+CMGS=\"+ZZxxxxxxxxxx\"");//change ZZ with country code and xxxxxxxxxxx with phone number to sms

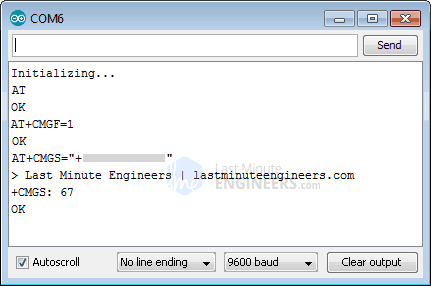
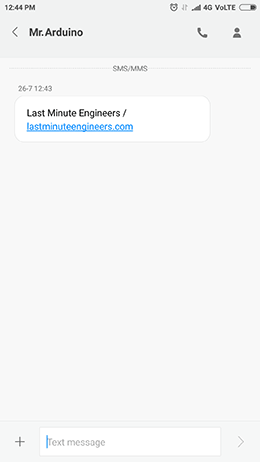
updateSerial();

mySerial.print("Last Minute Engineers | lastminuteengineers.com"); //text content

updateSerial();

mySerial.write(26);

در صورتی که فقط یک بار پیام ارسال شود، تابع loop خالی می ماند. در صورتی که لازم باشد که یک بار دیگر نیز پیام ارسال شود، تنها کافیست که کلید RESET در آردوینو فشار داده شود. تصویر زیر پیام ارسالی به  ماژول SIM800l را نشان می دهد.

* 
* 

پیام ارسالی به ماژول SIM800

## دریافت پیام با SIM800L

برای دریافت پیام می توان آردوینو را بصورت زیر کد نویسی کرد. کد زیر برای زمانی که لازم باشد هنگام دریافت پیام، عمل خاصی انجام گیرد نیز کاربرد دارد. به عنوان مثال، هنگامی که آردوینو پیامی را دریافت کند، رله خاموش یا روشن شود.

#include < SoftwareSerial.h >

//Create software serial object to communicate with SIM800L

SoftwareSerial mySerial(3, 2); //SIM800L Tx & Rx is connected to Arduino #3 & #2

void setup()

{

//Begin serial communication with Arduino and Arduino IDE (Serial Monitor)

Serial.begin(9600);

//Begin serial communication with Arduino and SIM800L

mySerial.begin(9600);

Serial.println("Initializing...");

delay(1000);

mySerial.println("AT"); //Once the handshake test is successful, it will back to OK

updateSerial();

mySerial.println("AT+CMGF=1"); // Configuring TEXT mode

updateSerial();

mySerial.println("AT+CNMI=1,2,0,0,0"); // Decides how newly arrived SMS messages should be handled

updateSerial();

}

void loop()

{

updateSerial();

}

void updateSerial()

{

delay(500);

while (Serial.available())

{

mySerial.write(Serial.read());//Forward what Serial received to Software Serial Port

}

while(mySerial.available())

{

Serial.write(mySerial.read());//Forward what Software Serial received to Serial Port

}

}

این کد نیز مشابه کدهای قبل است و تنها در چندین دستور AT متفاوت است. در این کد ، هنگامی که اتصال برقرار می شود، دستورات AT زیر اجرا می شوند:

دستور AT+CMGF=1 : قالب پیام را از نوع ‘متنی’ قرار می دهد. قالب پیشفرض PDU است.

دستور AT+CNMI=1,2,0,0,0 : چگونگی دستیابی کامپیوتر به پیام جدید را تعیین می کند. با استفاده از این دستور می توان تعیین کرد که ماژول پیام های ورودی جدید را مستقیما به کامپیوتر ارسال کند یا آنها را در حافظه خود ذخیره کند و سپس مکان ذخیره سازی پیام را به کامپیوتر اطلاع دهد.

پاسخ ماژول با کد دستوری CMT+ شروع می شود: تمام مقادیر در آن با استفاده از کاما ‘,’ از یکدیگر جدا شده اند. اولین مقدار شماره موبایل، مقدار دوم نام شخصی است که پیام را فرستاده، مقدار سوم تاریخ و زمان دریافت پیام و مقدار چهارم متن پیام است.

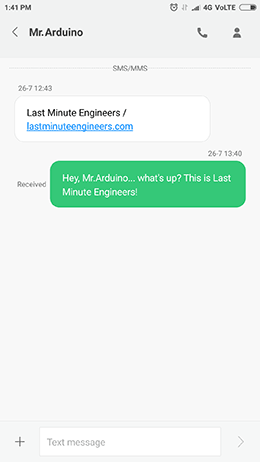
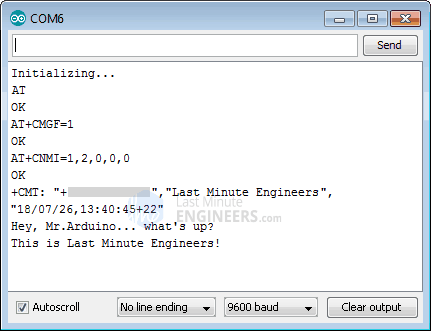
println("AT+CMGF=1"); // Configuring TEXT mode

updateSerial();

mySerial.println("AT+CNMI=1,2,0,0,0"); // Decides how newly arrived SMS messages should be handled

updateSerial();

این بار تابع loop (حلقه) خالی نیست  تا هر زمان که پیامی به ماژول ارسال شود، بتوان آن را خواند. به محض ارسال پیام به ماژول SIM800 ، می توان آن را بر روی مانیتور سریال مشاهده کرد.

* 
* 

دریافت پیام توسط ماژول SIM800 و نمایش خروجی بر روی مانیتور سریال

### افزایش اندازه بافر SoftwareSerial آردوینو برای دریافت پیام

در صورتی که پیام طولانی باشد، ممکن است که برخی از کاراکترها حذف شوند. این مساله به دلیل خطای کد نویسی نمی باشد بلکه بافر دریافتی SoftwareSerial پر شده و به همین دلیل، برخی از کاراکترها حذف می شوند.

ساده ترین راه حل افزایش اندازه بافر SoftwareSerial است. اندازه پیشفرض بافر 64 بایت است که می توان آن را تا 256 بایت (این مقدار بسته به کاربرد میتواند کمتر انتخاب شود) افزایش داد. برای این منظور، در کامپیوتر در صورتی که دارای سیستم عامل ویندوز باشد، به آدرس زیر رفته :

C:\Program Files (x86) -> Arduino -> hardware -> Arduino -> avr -> libraries -> SoftwareSerial

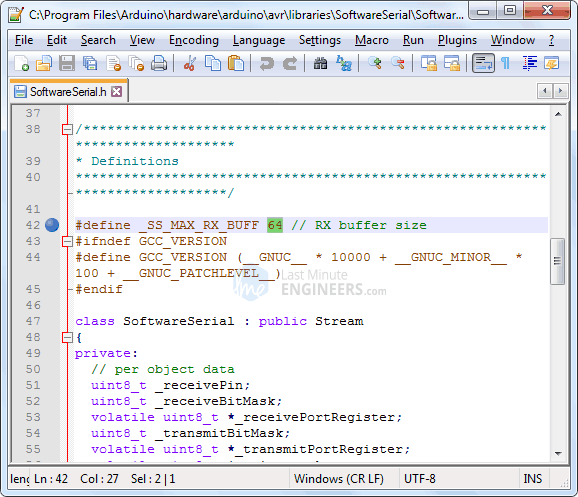
(در صورتی که از نسخه های جدیدتر آردوینو IDE استفاده می شود به پوشه src رفته)، سپس کتابخانه SoftwareSerial.h را باز کرده و دستور مربوط به اندازه بافر مانند زیر تغییر کند:

// RX buffer size  
#define \_SS\_MAX\_RX\_BUFF 64

باید به صورت زیر تغییر کند:

// RX buffer size  
#define \_SS\_MAX\_RX\_BUFF 256

پس از ذخیره کردن فایل، باید کد دوباره اجرا شود.

افزایش اندازه بافر SoftwareSerial آردوینو

## برقراری تماس با ماژول SIM800L و آردوینو

این دستورات برای برقراری تماس کمکی (SoS) در موقعیت های خطرناک و اضطراری بطور مثال، زمانیکه آتش سوزی یا سرقت در منزل رخ بدهد، کاربرد دارد.

قبل از اجرای برنامه، ابتدا باید شماره موبایل مورد نظر وارد شود به این صورت که در رشته ZZxxxxxxxxxx کد کشور با ZZ و شماره موبایل 10 رقمی مورد نظر با xxxxxxxxxx  جایگزین شوند.

#include < SoftwareSerial.h >

//Create software serial object to communicate with SIM800L

SoftwareSerial mySerial(3, 2); //SIM800L Tx & Rx is connected to Arduino #3 & #2

void setup()

{

//Begin serial communication with Arduino and Arduino IDE (Serial Monitor)

Serial.begin(9600);

//Begin serial communication with Arduino and SIM800L

mySerial.begin(9600);

Serial.println("Initializing...");

delay(1000);

mySerial.println("AT"); //Once the handshake test is successful, i t will back to OK

updateSerial();

mySerial.println("ATD+ +ZZxxxxxxxxxx;"); // change ZZ with country code and xxxxxxxxxxx with phone number to dial

updateSerial();

delay(20000); // wait for 20 seconds...

mySerial.println("ATH"); //hang up

updateSerial();

}

void loop()

{

}

void updateSerial()

{

delay(500);

while (Serial.available())

{

mySerial.write(Serial.read());//Forward what Serial received to Software Serial Port

}

while(mySerial.available())

{

Serial.write(mySerial.read());//Forward what Software Serial received to Serial Port

}

}

برای برقراری تماس باید دستورات AT زیر را به کار برد:

دستور ;ATD+ +ZZxxxxxxxxxx : با شماره وارد شده تماس می گیرد. کاراکتر (;) نشان دهنده پایان دستور است و باید در انتهای تمام خط کدها قرار بگیرد.

دستور ATH : تماس را قطع می کند.

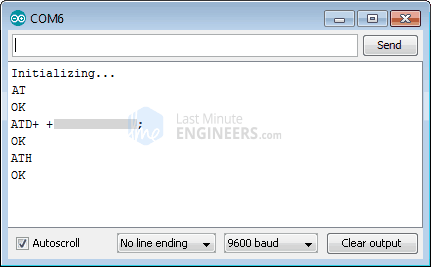
println("ATD+ +ZZxxxxxxxxxx;"); // change ZZ with country code and xxxxxxxxxxx with phone number to dial

updateSerial();

delay(20000); // wait for 20 seconds...

mySerial.println("ATH"); //hang up

updateSerial();





برقراری تماس با ماژول SIM800 و نمایش دستورات AT در مانیتور سریال

## دریافت تماس با آردوینو و ماژول SIM800L

برای دریافت تماس، تنها کافیست که به ماژول SIM800L گوش داده شود. کد زیر برای زمانی که لازم باشد هنگام دریافت تماس از یک شماره بخصوص، کار خاصی انجام گیرد، کاربرد دارد.

#include < SoftwareSerial.h >

//Create software serial object to communicate with SIM800L

SoftwareSerial mySerial(3, 2); //SIM800L Tx & Rx is connected to Arduino #3 & #2

void setup()

{

//Begin serial communication with Arduino and Arduino IDE (Serial Monitor)

Serial.begin(9600);

//Begin serial communication with Arduino and SIM800L

mySerial.begin(9600);

Serial.println("Initializing...");

}

void loop()

{

updateSerial();

}

void updateSerial()

{

delay(500);

while (Serial.available())

{

mySerial.write(Serial.read());//Forward what Serial received to Software Serial Port

}

while(mySerial.available())

{

Serial.write(mySerial.read());//Forward what Software Serial received to Serial Port

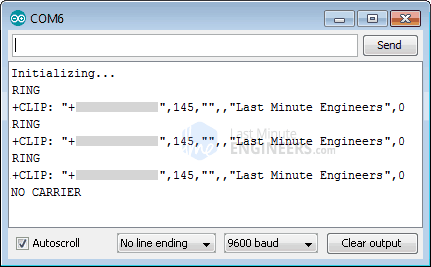
}

}

زمانی که تماس دریافت می شود، “RING” در مانیتور سریال مشاهده می شود. به دنبال “RING” شماره تلفن و شناسه تماس گیرنده نیز در مانیتور سریال نرم افزار آردوینو دیده می شود. برای پذیرش یا قطع تماس ورودی، از دستورات AT زیراستفاده می شود:

دستور ATA  : برای پذیرش تماس ورودی به کار میرود.

دستور ATH : تماس ورودی را قطع می کند. هنگام قطع تماس، “NO CARRIER” را بر روی مانیتور سریال می فرستد و نشان دهنده این است که تماسی برقرار نشده است.

* 
* 

خروجی مانیتور سریال زمانی که تماس توسط ماژول SIM800L GSM دریافت می شود

# آموزش راه اندازی ماژول Sim800L بخش اول تست شبکه GSM با سیم کارت

4 سال قبل

[۳۶۹ دیدگاه](https://digispark.ir/network-test-gsm-sim800l-module-arduino/#comments)

توسط [اروند طباطبایی](https://digispark.ir/author/arvand/)

42,670 بازدید

3 دقیقه زمان مطالعه



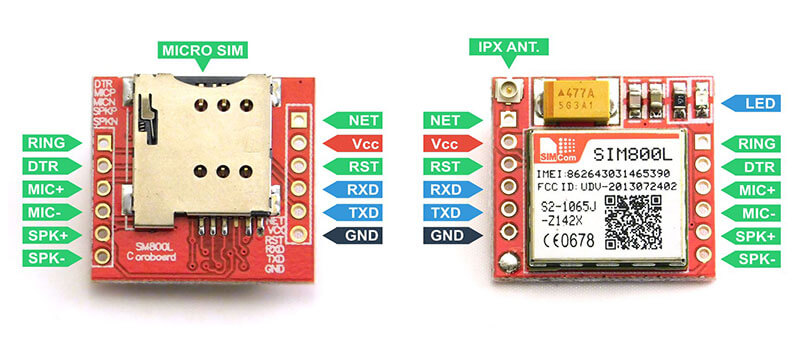


نوشته شده توسط [اروند طباطبایی](https://digispark.ir/author/arvand/)

حدود ۲ دهه از زمان ارایه اولین تلفن همراه می‌گذرد. در این مدت شاهد رشد و پیشرفت این کوچولوی همه کاره بوده‌ایم. با پیشرفته‌تر شدن و در دسترس عموم قرار گرفتن تلفن همراه، کاربری‌های جدید هم برای آن تعریف شد. از این پروتکل برای مدیریت یک مدار فرمان می‌توان استفاده کرد. **ماژول‌های GSM** عملکرد یکسانی با تلفن همراه دارند. پروتکل ارتباطی آن‌ها دقیقا یکسان است. در بستر مخابراتی می‌توان فرمان و انتقال دیتا انجام داد. **ماژول‌های GSM** انواع مختلف با ویژگی‌های متنوع دارند. **تراشه ماژول Sim800L** یکی از مقرون به صرفه‌ترین و کم حجم‌ترین این ماژول‌ها است. در این مجموعه آموزش شیوه کار با **تراشه Sim800L** را یاد خواهید گرفت. در ادامه با [مرجع تخصصی آردوینو به زبان فارسی](https://daneshjookit.com/121-%D8%A8%D8%B1%D8%AF-%D8%A2%D8%B1%D8%AF%D9%88%DB%8C%D9%86%D9%88-arduino-board)، [دیجی اسپارک](http://digispark.ir) همراه باشید.

## ساختار تراشه GSM Sim800L

اینکه درون این تراشه‌ها چه اتفاقی می‌افتد، مبحث تخصصی مخابراتی و برای برخی از کاربران سردرگمی ایجاد می‌کند. ولی به صورت کلی این تراشه مانند یک درگاه است که مفاهیم **شبکه مخابراتی** را دریافت و تبدیل می‌کند. سپس مقادیر مورد نیاز را تبدیل و در شبکه **مخابراتی** ارسال می‌کند. عمدتا ماژول‌های GSM با اسلات سیم کارت همراه هستند. در [ماژول Sim800L](https://daneshjookit.com/module/%D9%85%D8%AE%D8%A7%D8%A8%D8%B1%D8%A7%D8%AA%DB%8C/%D8%B3%DB%8C%D9%85-%DA%A9%D8%A7%D8%B1%D8%AA-gsm/1980-sim800l-gsm-gprs.html) از سیم کارت سایز مایکرو Micro Sim Card  استفاده می‌شود. پس قدم اول در استفاده از این ماژول داشتن سیم کارت مایکرو است.



## ویژگی‌های ماژول sim800L

[ماژول جی اس ام Sim800L](https://daneshjookit.com/143-%D8%B3%DB%8C%D9%85-%DA%A9%D8%A7%D8%B1%D8%AA-gsm) پشتیبانی از چهار باند ۸۵۰/۹۰۰/۱۸۰۰/۱۹۰۰ مگاهرتز

[پروتکل ارتباطی سریال (UART)](http://digispark.ir/%D8%A2%D9%85%D9%88%D8%B2%D8%B4-%D8%AC%D8%A7%D9%85%D8%B9-%D8%A7%D8%B1%D8%AA%D8%A8%D8%A7%D8%B7-%D8%B3%D8%B1%DB%8C%D8%A7%D9%84-%D8%A2%D8%B1%D8%AF%D9%88%DB%8C%D9%86%D9%88/)

کنترل از طریق AT Command و لایبرری

قابلیت اتصال میکروفن و اسپیکر

جربان مصرفی حداکثر ۲ آمپر

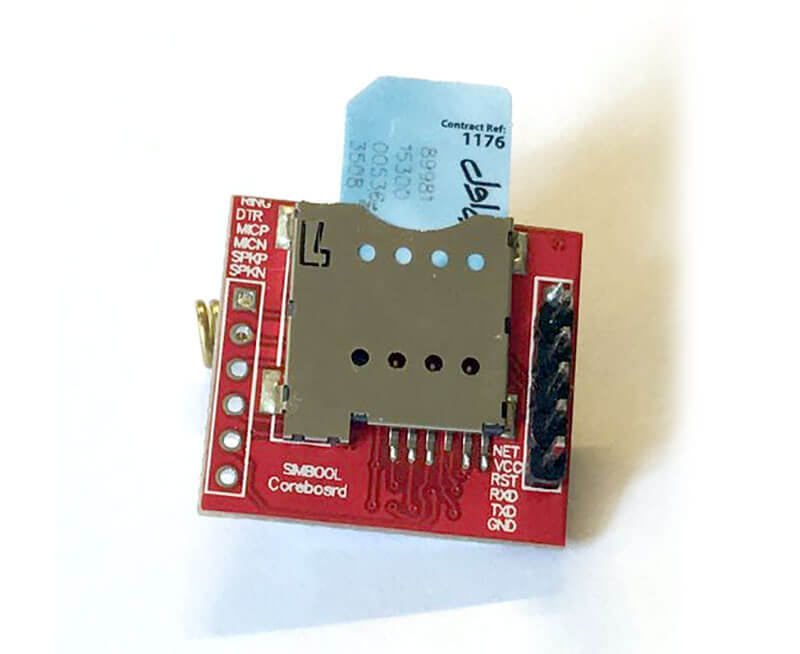
تغذیه ۳٫۸ تا ۴٫۲ ولت DC

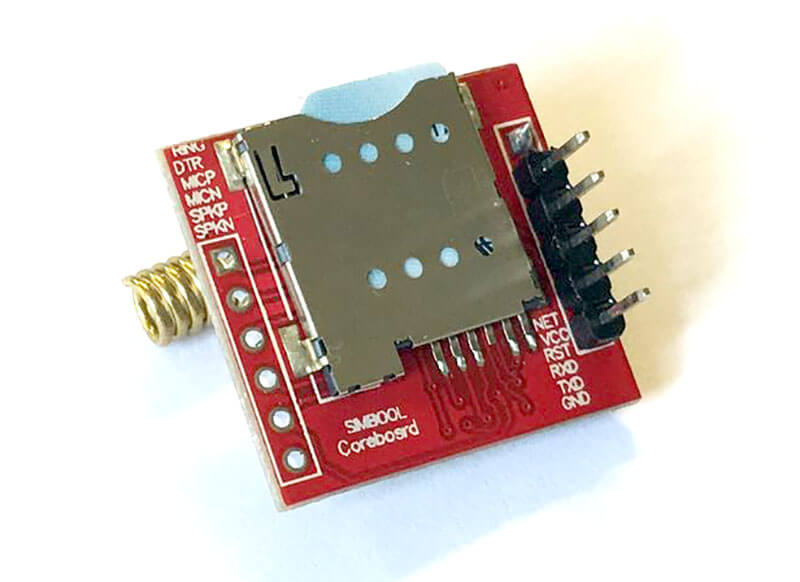
اسلات سیم‌کارت میکرو

دمای کاری ۴۰- تا ۸۵+ سانتی‌گراد

## راه اندازی ماژول Sim800L

ابتدا سیم کارت مایکرو را مانند تصویر زیر در اسلات سیم کارت **ماژول GSM** قرار دهید. همانطور که در بالا ذکر شد تغذیه این مدار حداکثر ۴٫۲ ولت می‌باشد. این محدوده ولتاژ کاری سلول لیتیومی است. از اینرو برای راه اندازی پیشنهاد می‌شود که از یک سلول باتری لیتیوم پلیمر استفاده شود. البته ابعاد کوچک و مصرف جریان ناچیز این ماژول هم دلیلی بر انتخاب این محدوده ولتاژ کاری است. برای استفاده از این ماژول نیاز به رابط سریال داریم. بنابراین پایه RX از ماژول را به پین tx برد آردوینو، سپس پایه TX از ماژول را به پین rx [برد آردوینو Arduino](http://digispark.ir/arduino-rfid-rc522-tag-number/) وصل کنید. پایه ریست را در این آموزش نیاز نداریم. ولی در حالت کلی باید این پایه هم به پین ریست آردوینو متصل شود. برای استفاده از [رابط سریال برد آردوینو Arduino](http://digispark.ir/%d8%a2%d9%85%d9%88%d8%b2%d8%b4-%d8%a7%d8%b1%d8%aa%d8%a8%d8%a7%d8%b7-%d8%b3%d8%b1%db%8c%d8%a7%d9%84-arduino/) نیاز به کتابخانه سریال داریم، این کتابخانه به صورت پیش فرض در سورس آردوینو Arduino قرار دارد.



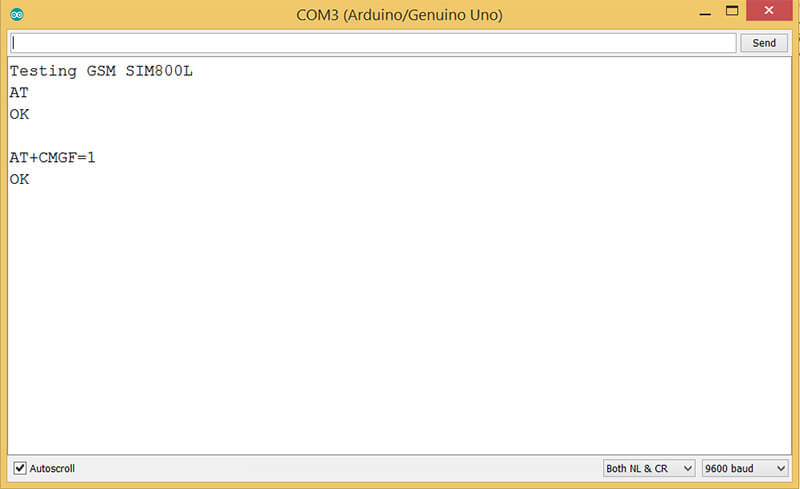


## کدهای برنامه ماژول Sim800L



|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27  28  29  30 | #include <SoftwareSerial.h>    String Arsp, Grsp;  SoftwareSerial gsm(10, 11); // RX, TX    void setup() {    // put your setup code here, to run once:      Serial.begin(9600);    Serial.println("Testing GSM SIM800L");    gsm.begin(4800);    }    void loop() {    // put your main code here, to run repeatedly:      if(gsm.available())    {      Grsp = gsm.readString();      Serial.println(Grsp);    }      if(Serial.available())    {      Arsp = Serial.readString();      gsm.println(Arsp);    }    } |

پس از انتقال کدها در نرم افزار آردوینو و آپلود آن روی برد، کافیست که سریال مانیتور نرم افزار را باز کنید. تست ماژول به صورت اتوماتیک انجام شده و نتایج را از همان جا می‌توانید ببینید. همچنین در این حالت دستورات AT را میتوانید در سریال مانیتور تست کنید. همانند تصویر زیر:



### لوازم مورد نیاز

[برد آردوینو Arduino](https://daneshjookit.com/%D8%A8%D8%B1%D8%AF-%D8%A2%D8%B1%D8%AF%D9%88%DB%8C%D9%86%D9%88-arduino-board/1691-arduino-uno-r3.html)

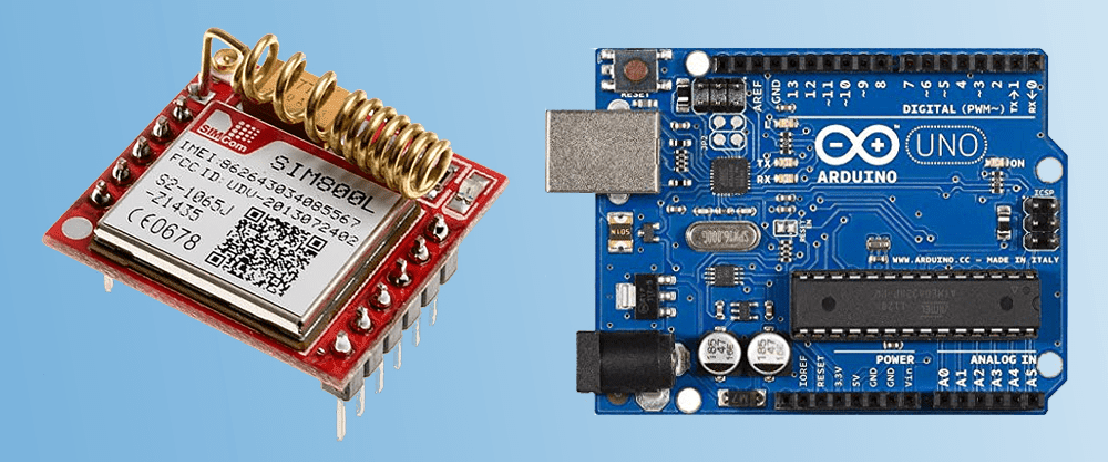
[ماژول GSM Sim800L](https://daneshjookit.com/%D9%85%D8%A7%DA%98%D9%88%D9%84-module/1980-sim800l.html)

[باتری لیتیوم پلیمر](https://daneshjookit.com/%D9%84%DB%8C%D8%AA%DB%8C%D9%88%D9%85/1293-%D8%A8%D8%A7%D8%AA%D8%B1%DB%8C-%D9%84%DB%8C%D8%AA%DB%8C%D9%88%D9%85-%D9%BE%D9%84%DB%8C%D9%85%D8%B1-li-po-750mah.html)

# راه اندازی و تست ماژول SIM800L با اردوینو

خرداد 6, 1400

[اموزش آردوینو ARDUINO](https://makingprojec.com/blog/category/weblog/arduino-training/), [وبلاگ](https://makingprojec.com/blog/category/weblog/)

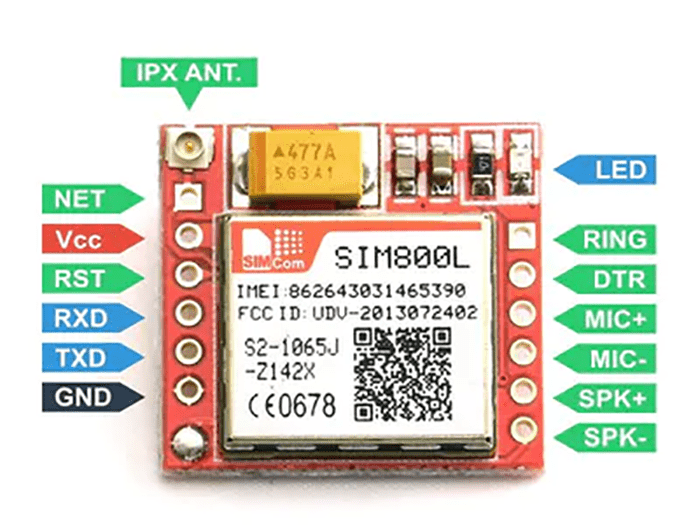


در این آموزش از ساخت پروژه نحوه راه اندازی و تست ماژول SIM800L با اردوینو را توضیح خواهیم داد. ماژول sim800l یکی از پر کاربرد ترین ماژول های gsm هست و یکی از رایج ترین کاربرد این ماژول کنترل لوازم با sms و تماس هست.

تغذیه این ماژول 3.4 تا 4.4 ولت و جریان مصرفی این ماژول 2 امپر و این ماژول تشنه تغذیه هست این مورد باعث شده که خیلی از افراد توانایی راه اندازی این ماژول رو نداشتن و اینکه نمیشه تغذیه این ماژول رو با اردوینو تامین کرد چون خروجی 5 ولت آردینو حدود 500 میلی آمپر شدت جریان داره که مناسب این ماژول نیست و باعث میشه ماژول اسیب ببینه

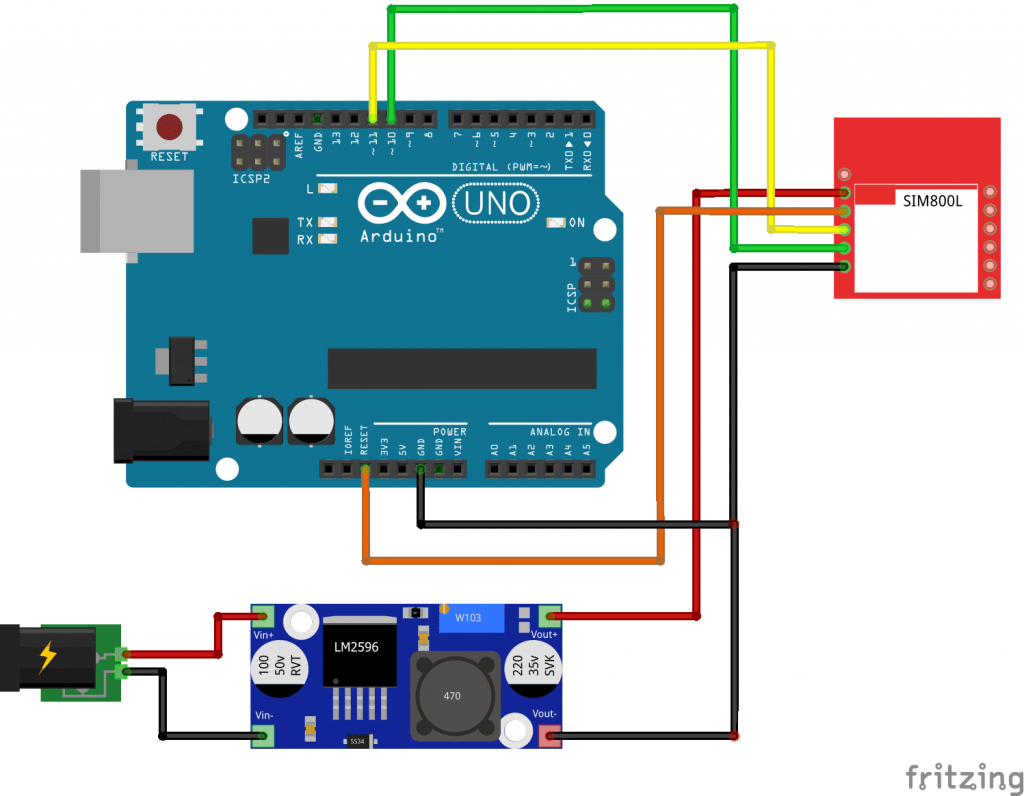
## پین های ماژول sim800l

این ماژول دارای 12 پایه هست ما برای راه اندازی ماژول sim800l فقط به پین RST , TXD , RXD نیاز داریم که به اردوینو متصل میشه

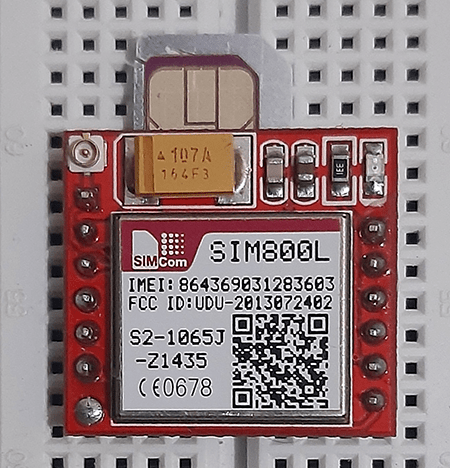


## شماتیک تست و راه اندازی ماژول SIM800L

طبق شماتیک مدار زیر را ببندید توجه داشته باشبد که ولتاژ ماژول lm2596 توسط مولتی ترن بین 3.4 تا 4.4 ولت تنظیم شده باشه و حتی شما میتوانید تغذیه ماژول رو توسط باتری لتیوم یونی تامین کنید



مانند تصویر زیر سیمکارت رو داخل sim800l قرار میدیم



## کد اردوینو تست و راه اندازی ماژول SIM800L

کد زیر را کپی کنید و در نرم افزار اردوینو پیست کنید و روی اردوینو پروگرام کنید

#include <SoftwareSerial.h>

String txt1,txt2;

SoftwareSerial gsm(10,11); //10 --> Rx , 11 --> Tx

void setup() {

Serial.begin(9600);

Serial.println("Testing GSM SIM800L");

gsm.begin(9600);

}

void loop() {

if(gsm.available())

{

txt1 = gsm.readString();

Serial.println(txt1);

}

if(Serial.available())

{

txt2 = Serial.readString();

gsm.println(txt2);

}

}

بعد از اینکه کد رو روی اردونیو پروگرام کردید سریال مانیتور اردوینو رو اجرا کنید و سرعت سریال مانیتور رو روی 9600 قرار دهید و اینکه ماژول دارای یک led هست که با چشمک زدن ماژول عملکرد ماژول مشخص میشه

در صورتی که LED هر 1 ثانیه یک بار چشمک زد ممکنه مشکل از تغذیه ماژول یا نبود سیمکارت در ماژول باشه

## دستور های At Command ماژول SIM800L

ما برخی از دستورات مهم AT COMMAND ماژول رو قرار دادیم و شما این دستورات رو در سریال مانیتور اردوینو میزنید و به این صورت متوجه میشد که ماژول به درستی عمل میکنه یا نه

|  |  |
| --- | --- |
| **عملکرد** | **AT COMMAND** |
| چک کردن ارتباط با ماژول ( بررسی سالم بودن ماژول ) | AT |
| نمایش ورژن FIRMWARE ماژول | ATI |
| نمایش ورژن کامل FIRMWARE ماژول | AT+GMR |
| نمایش مدل ماژول | AT+GMM |
| نمایش قدرت سیگنال ( اگر صفر باشد یعنی قادر به دریافت سیگنال و اتصال به شبکه نیست ) | AT+CSQ |
| نمایش نام اپراتور شبکه | AT+CSPN? |
| تست سیمکارت ( در صورت صفر بودن سیمکارت قطع و در صورت 1 بودن سیمکارت متصل می باشد و ماژول سیمکارت را شناسایی کرده است. ) | AT+CSMINS? |
| بررسی اینکه سیمکارت در حالت پرواز قراردارد یا خیر؟ ( باید 1 برگرداند؛ در غیر این صورت با دستور AT+CFUN=1 فعال می کنیم ) | AT+CFUN? |
| نمایش زمان ماژول | AT+CCLK? |
| برای تنظیم باودریت ماژول استفاده می شود ( باودریت ماژول به صورت خودکار بوده و خود را با تنظیمات سریال هماهنگ می کند، ولی میتوان به صورت دستی نیز تغییر داد. ) | AT+IPR=x AT+IPR=9600 |
| نمایش اطلاعات شناسایی محصول | AT+GSV |
| بررسی وضعیت آماده بودن سیمکارت جهت تماس و ارسال پیام ( باید وضعیت READY نمایش داده شود ) | AT+CPIN? |
| نمایش شماره سریال سیمکارت ( IMEI ) | AT+CGSN |

# چطور با سنسور رطوبت خاک و برد آردوینو گلدان هوشمند بسازیم؟

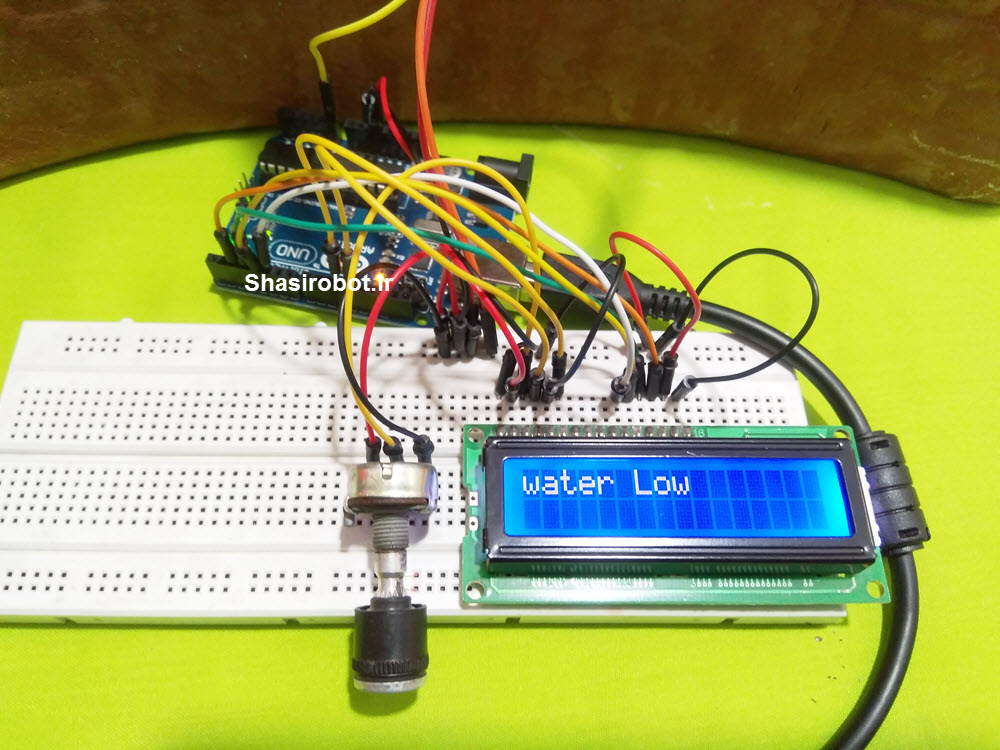
[حمید شهریاری](http://shasirobot.ir/author/shasirobotali/)دسامبر 29, 2020آخرین بروزرسانی: دسامبر 30, 2020

30 771 زمان تقریبی مطالعه 7 دقیقه

برد آردوینو گلدان هوشمند و سنسور رطوبت خاک

# چطور با سنسور رطوبت خاک و برد آردوینو گلدان هوشمند بسازیم؟

حتما شما هم گلخونه های هوشمند رو دیدید که بصورت اتوماتیک بمحض کاهش رطوبت خاک عمل آبیاری رو انجام میده.سنسوری که برای اینکار استفاده میشه یه سنسور رطوبت خاک هست که توی این مقاله نحوه راه اندازی این سنور رو یاد میگیریم



توی این پروژه یاد میگیریم با استفاده از یه سنسور رطوبت خاک یک گلدان هوشمند بسازیم ، بطوریکه وقتی رطوبت خاک از مقدار معنی کمتر بشه اونوقت یه پمپ کوچیک گلدان یا گلخونه رو آبیاری کنه (البته چون من توی وسایلام پم آب مناسب نداشتم از یه LED توی خروجی استفاده میکنم)

# سنسور رطوبت خاک چطور کار میکند؟

نحوه کار سنسور رطوبت خاک که اغلب برای ساخت گلدان هوشمند و گلخانه هوشمند استفاده میشه خیلی سادست.



این سنسور تشکیل شده از یه صفحه U شکل که دقیقا مثل یک پتانسومتر عمل میکنه و با تغییر رطوبت خاک میزان مقاومت خروجی دوسر این سنسور هم تغییر میکنه که با اندازه گیری این مقاومت توسط برد اردوینو میتونیم از میزان رطوبت خاک اطلاع پیدا میکنیم

### رابطه میزان رطوبت با مقاومت خروجی سنسور رطوبت:

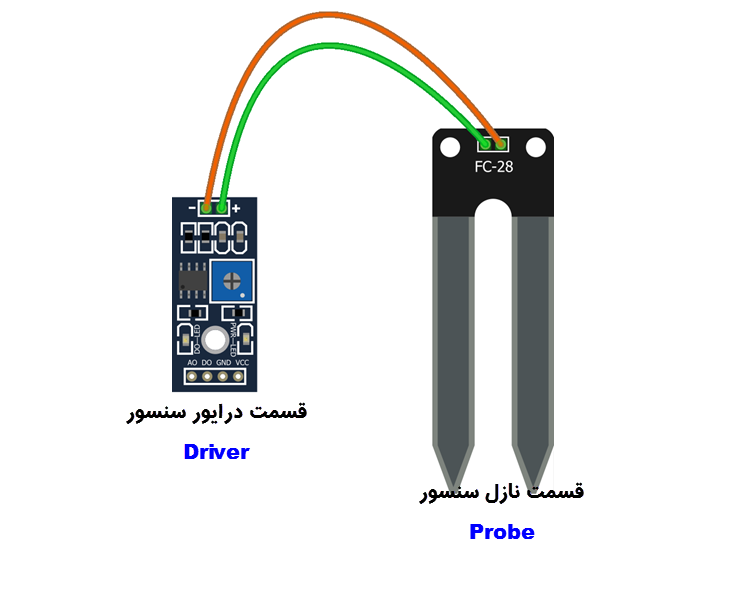
هر چه رطوبت خاک بیشتر بشه میزان هدایت پذیری بیشتر و در نتیجه مقاومت خروجی سنسور رطوبت کمتر میشه و برعکس ، هنگامی که گلدون آب کمتری داره هدایت پذیری کمتر و مقاومت خروجی سنسور رطوبت بیشتر میشه

با تغییر مقاومت خروجی ولتاژ دو سر سنسور تغییر میکنه که براحتی با استفاده از مبدل آنالوگ به دیجیتال برد آردوینو میتونیم این مقدار رو اندازه گیری کنیم.

توی شکل زیر رابطه میزان رطوبت با مقاومت خروجی رو میبینید:

# بررسی اجزای ماژول رطوبت خاک

هر ماژول رطوبت خاک از دو قسمت جدا از همدیگه تشکیل شده:



### 1-قسمت نازل سنسور(Probe)

نازل سنسور رطوبت خاک یک قطعه رسانا چنگال مانند هست که توی جایی که میخواهیم میزان رطوبتشو اندازه بگیریم قرار میدیم

این نازل دقیقا مثل یک مقاومت متغیر عمل میکنه و با کم و زیاد شدن رطوبت خاک مقاومت خروجیش کم و زیاد میشه

### 2-قسمت درایور (ِDriver)

برای اتصال نازل به برد آردوینو 2 راه داریم:

برای اتصال پراب سنسور به برد آردوینو از یه مدار راه انداز یا همون درایور استفاده میکنیم.

ابتدا پراب به مدار راه انداز متصل میشه و خروجی های مدار راه انداز به برد اردوینو میره تا اندازه گیری میزان رطوبت خاک با دقت انجام بشه.

هر ماژول سنسور خاک 2 تا خروجی در اختیار ما قرار میده:

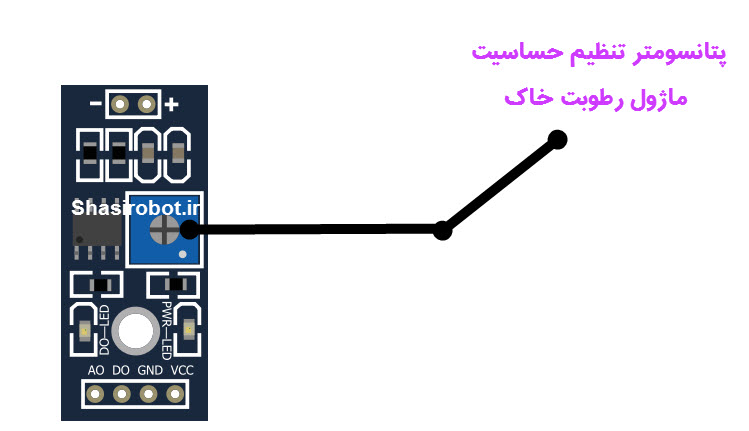
1- خروجی آنالوگ یا همون پایه A0 که میزان رطوبت هر لحظه رو در اختیار ما قرار میده و میتونیم با اتصال به یکی از پایه های A0…………A5 برد آردوینو میزان دقیق رطوبت رواندازه گیری کنیم

2-خروجی دیجیتال یا همون D0 ماژول که  فقط 2 حالت خشک یا مرطوب بودن خاک رو نمایش بده.توی خروجی دیجیتال دیگه نمتونیم میزان دقیق رطوبت خاک رو اندازه گیری کنیم بلکه فقط میتونیم بفهمیم خاک گلدون خشکیه یا نه ….

[مطلب پیشنهادی  چطوربا برد آردوینو ماژول فرستنده گیرنده مادون قرمز TCRT5000 راه اندازی کنیم؟](http://shasirobot.ir/2021/01/04/%da%86%d8%b7%d9%88%d8%b1%d8%a8%d8%a7-%d8%a8%d8%b1%d8%af-%d8%a2%d8%b1%d8%af%d9%88%db%8c%d9%86%d9%88-%d9%85%d8%a7%da%98%d9%88%d9%84-%d9%81%d8%b1%d8%b3%d8%aa%d9%86%d8%af%d9%87-%da%af%db%8c%d8%b1%d9%86/" \t "_blank)

## چطور حساسیت ماژول رطوبت سنج رو کم یا زیاد کنیم؟

هر ماژول رطوبت سنج مجهز به یک پتانسومتر هست که برای تعریف میزان استانه رطوبت مورد استفاده قرار میگیره ، بطوریکه بالاتر از اون آستانه رو مرطوب و پایین تر از اون استانه رو خشک فرض میکنه .



بمحض اینکه مقدار خروجی سنسور رطوبت از آستانه ای که شما با پتانسومتر تعریف کردید بیشتر شد خروجی دیجیتال ماژول D0 فعال میشه و شما مثلا میتونید یک رله رو فعال کنید(حالا این رله میتونه به یک پمپ آب وصل شده باشه)

دقت کنید که تنظیم میزان حساسیت ماژول تنها در حالتی که شما دارید از خروجی دیجیتال(D0) استفاده میکنید مورد استفاده قرار میگیره نه وقتی که دارید خروجی آنالوگ() رو اندازه گیری میکنید.

با چرخش پتانسیومتر بصورت ساعت گرد حساسیت ماژول افزایش پیدا میکنه

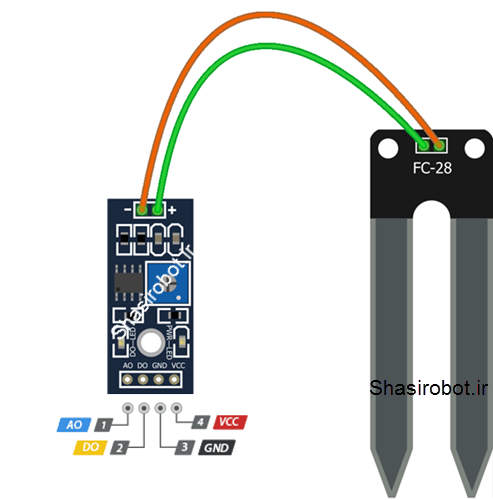
با چرخش پتانسیومتر بصورت پاد ساعت گرد حساسیت ماژول کاهش پیدا میکنه

### بر روی هر ماژول رطوبت سنج 2 تا LED قرار گرفته:

LED اول که بمحض اتصال ماژول رطوبت به منبع تغذیه ، روشن میشه

LED دوم فقط زمانی که خروجی دیجیتال رو انتخاب میکنیم وقتی ماژول بحالت LOW میره این LED هم روشن میشه.

## آرایش پایه های ماژول رطوبت خاک:



AO:

خروجی آنالوگ(پیوسته)ماژول رطوبت خاک هست که باید به یکی از پایه های A0…………A5 برد آردوینو متصل کنیم

DO:

خروجی دیجیتال ماژول رطوبت خاک هست که میتونیم به یک رله یا برد آردوینو متصل کنیم

VCC:

این پایه بعنوان تغذیه ماژول استفاده میشه و به ولتاژ 3.3 تا 5 ولت متصل میکنیم

GND:

این پایه به زمین مدار متصل میشه

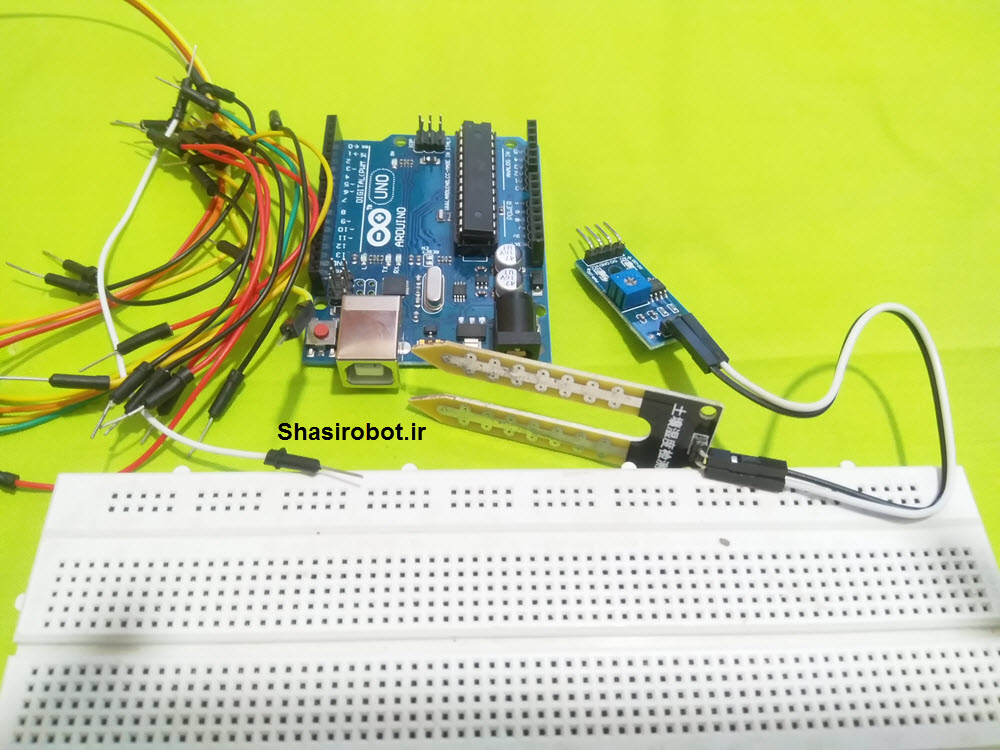
## پروژه اول:

## راه اندازی ماژول رطوبت خاک با استفاده از خروجی آنالوگ و نمایش داده ها روی سریال مانیتور

همونطور که میدونید ماژول رطوبت خاک هم خروجی دیجیتال و هم خروجی آنالوگ رو در اختیار ما قرار میده.توی آزمایش اول میریم و خروجی آنالوگ سنسور رطوبت خاک رو به بردآردوینو متصل میکنیم و خروجی ها رو توی سریال مانیتور مشاهده میکنیم

اینکار باعث میشه که بدونیم میزان خروجی سنسور بازای وقتی که خاک گلدون ما خشک ، نیمه مرطوب و کاملا خیس هست چقدر تغییر میکنه

## قطعات مورد نیاز پروژه راه اندازی ماژول رطوبت خاک با استفاده از خروجی آنالوگ و نمایش داده ها روی سریال مانیتور



1-برد آردوینو یونو(برد آردوینو نانو یا هر برد دیگه از خانواده آردوینو)

2-سنسور رطوبت خاک

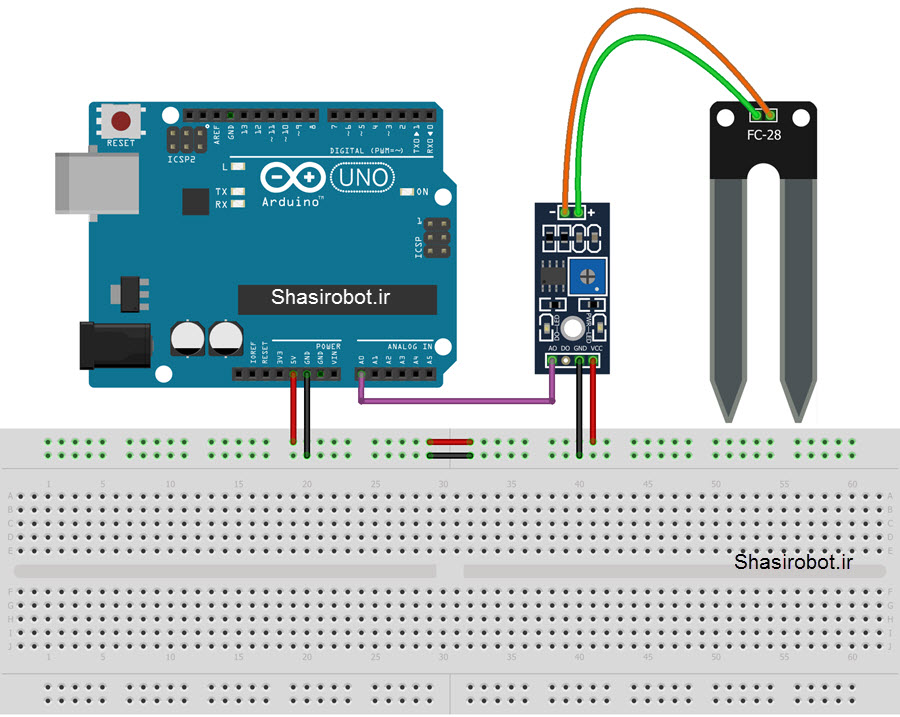
3-برد بورد

4-سیم برد بوردی

5-یه گلدون

## شماتیک برد بوردی پروژه راه اندازی ماژول رطوبت خاک با استفاده از خروجی آنالوگ و نمایش داده ها روی سریال مانیتور

مدار راه اندازی سنسور رطوبت خاک خیلی سادست .مدار رو مطابق شکل زیر ببنید



## سورس کد پروژه راه اندازی ماژول رطوبت خاک با استفاده از خروجی آنالوگ و نمایش داده ها روی سریال مانیتور

سورس زیر توی محیط Arduino IDE کپی کنید و روی دکمه Compile بزنید تا روی برد آردوینو یونو منتقل بشه



#define sensorPin A0

​

void setup()

{

Serial.begin(9600);

}

​

void loop() {

//get the reading from the function below and print it

Serial.print("Analog output: ");

Serial.println(readSensor());

delay(1000);

}

​

​

int readSensor()

{

int val = analogRead(sensorPin); // Read the analog value form sensor

return val; // Return value

}

نکته مهم:

[مطلب پیشنهادی  چطور با آردوینو بازی نجات دایناسور گوگل Chrome را هک کنیم؟](http://shasirobot.ir/2021/02/26/%da%86%d8%b7%d9%88%d8%b1-%d8%a8%d8%a7-%d8%a2%d8%b1%d8%af%d9%88%db%8c%d9%86%d9%88-%d8%a8%d8%a7%d8%b2%db%8c-%d9%86%d8%ac%d8%a7%d8%aa-%d8%af%d8%a7%db%8c%d9%86%d8%a7%d8%b3%d9%88%d8%b1-%da%af%d9%88%da%af/" \t "_blank)

با توجه به اینکه مقدار خروجی سنسور با تغییر نوع خاک ممکنه تغییر کنه شما قبل از اینکه سنسور رو برای پروژه های بعدیتون استفاده کنید باید مقادیر خروجی رو از برنامه بالا برای 3 حالت:

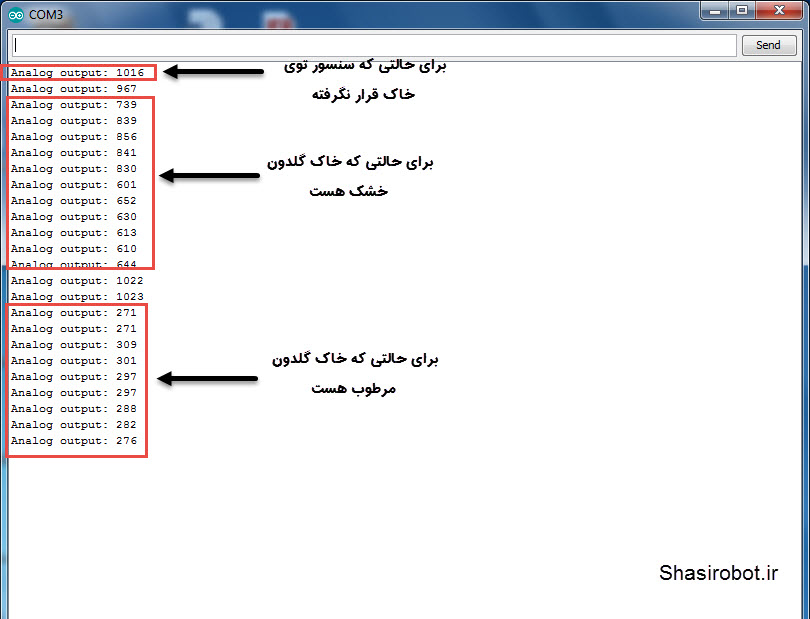
خاک غرغاب

خاک مرطوب

خاک خشک

بدست بیارید.

برای خاک گلدون من مقادیر زیر بدست اومد که ممکنه برای شما کمی تفاوت داشته باشه



خاک غرغاب در حدود (کمتر از 500)

خاک مرطوب(بین 650 تا 800)

خاک خشک(بین 250تا 350)



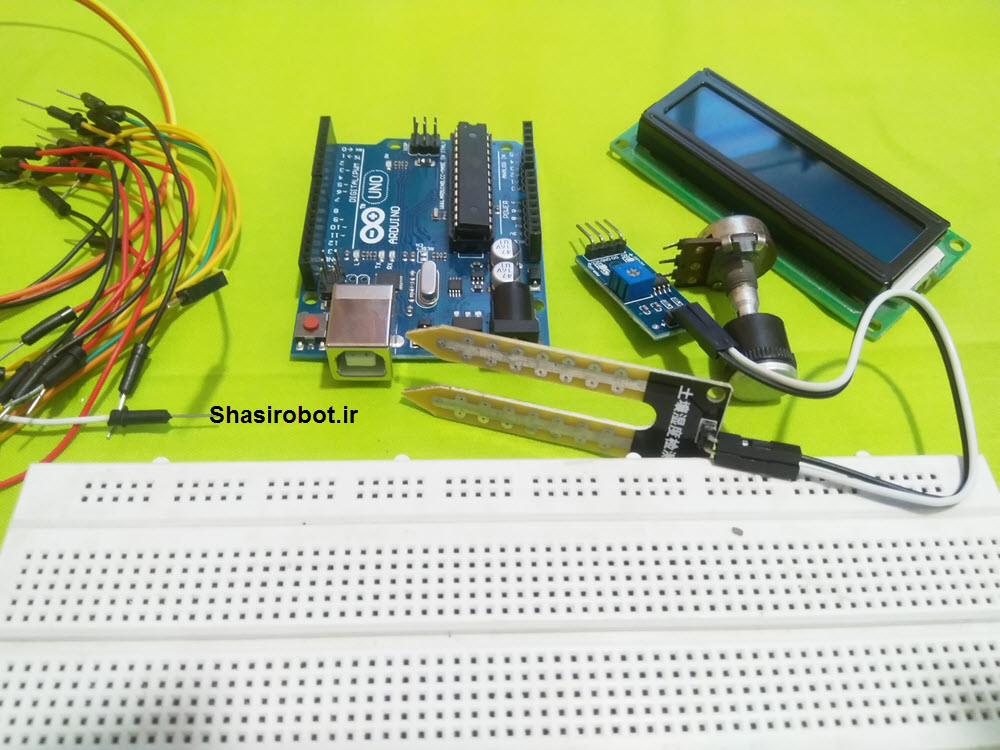
خب دیگه الان که متوجه شدیدم این سنسور توی شرایط خاک مرطوب و خشک و غرغاب خروجیش چطور تغییر میکنه بریم و مقادیر رو روی ال سی دی کاراکتری نمایش بدیم….

## پروژه دوم:

## راه اندازی ماژول رطوبت خاک و نمایش داده ها روی LCD کاراکتری 2\*16

توی این پروژه با توجه به مقادیر بدست آمده از آزمایش اول میتونیم مقادیر خروجی رو روی LCD نمایش بدیم.البته ممکنه مقادیر خروجی ازمایش شما کمی متفاوت تر باشه که براحتی با استفاده از تکنیک بالا میتونید به دقیق ترین حالت برسید

## قطعات مورد نیاز پروژه راه اندازی ماژول رطوبت خاک و نمایش داده ها روی LCD کاراکتری 2\*16



1-برد آردوینو یونو(برد آردوینو نانو یا هر برد دیگه از خانواده آردوینو)

2-سنسور رطوبت خاک

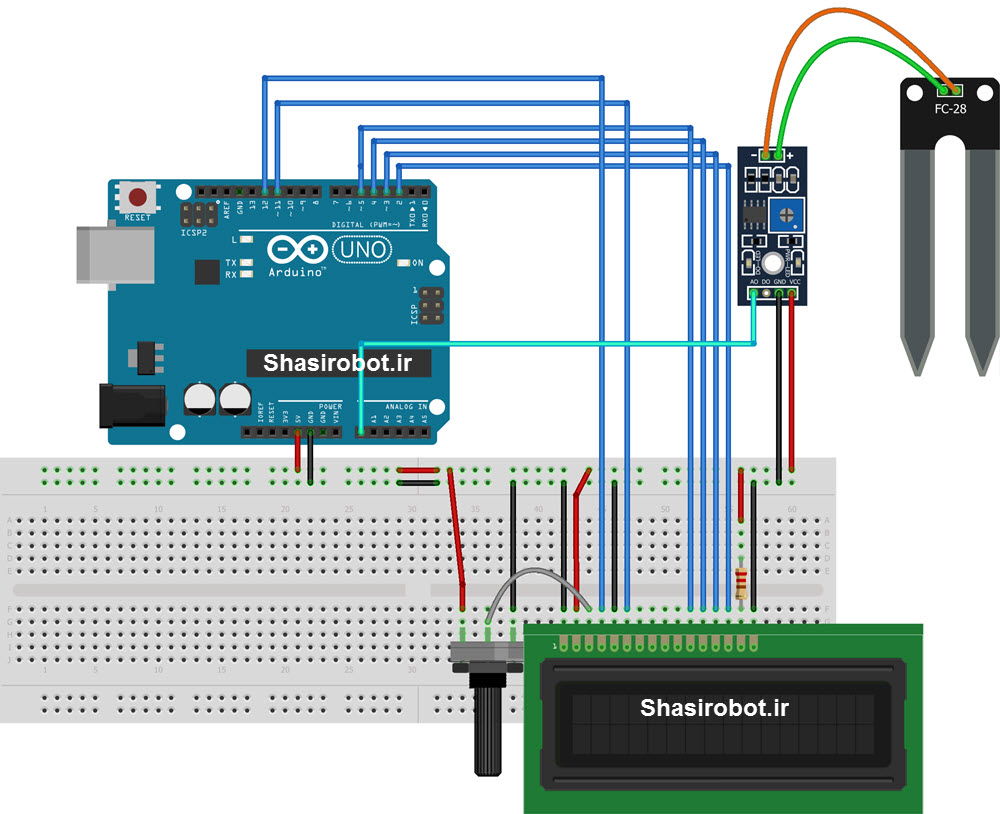
3-برد بورد

4-سیم برد بوردی

5-یه گلدون

6- LCD کاراکتری 2\*16

## شماتیک برد بوردی پروژه راه اندازی ماژول رطوبت خاک و نمایش داده ها روی LCD کاراکتری 2\*16



## سورس کد پروژه راه اندازی ماژول رطوبت خاک و نمایش داده ها روی LCD کاراکتری 2\*16

کد سورس زیر رو توی محیط برنامه نویسی Arduino IDE کپی کنید و کلید کامپایل Compile رو بزنید تا برنامه روی برد آردوینو شما منتقل بشه

#define sensorPin A0

#include <LiquidCrystal.h> // Include LiquidCrystal Library

#define soilWet 600 // Define max value we consider soil 'wet'

#define soilDry 800 // Define min value we consider soil 'dry'

#define outPin 8

LiquidCrystal lcd(12, 11, 5, 4, 3, 2); // Create an LCD object.

void setup()

{

lcd.begin(16, 2);

Serial.begin(9600);

lcd.clear();

lcd.print(" shasirobot.ir");

delay(2000);

lcd.clear();

}

void loop() {

int moisture = readSensor();

if (moisture < soilWet)

{

lcd.print("water Full");

}

else if (moisture >= soilWet && moisture < soilDry)

{

lcd.print("water Perfect");

}

else

{

lcd.print("water Low");

}

delay(1000); // Take a reading every second for testing

lcd.clear();

}

int readSensor()

{

int val = analogRead(sensorPin); // Read the analog value form sensor

return val; // Return value

}

## تحلیل کد پروژه راه اندازی ماژول رطوبت خاک و نمایش داده ها روی LCD کاراکتری 2\*16

توضیحات برنامه خیلی پیچیده نیست ، اگه به سورس یه نگاه بندازید متوجه میشید بازم من توضیح میدم تا جای ابهامی برای شما عزیزانم باقی  نمونه:

[مطلب پیشنهادی  آموزش راه اندازی روتاری انکودر(Rotary Encoder)با سروو موتور و برد آردوینو](http://shasirobot.ir/2021/02/14/%d8%a2%d9%85%d9%88%d8%b2%d8%b4-%d8%b1%d8%a7%d9%87-%d8%a7%d9%86%d8%af%d8%a7%d8%b2%db%8c-%d8%b1%d9%88%d8%aa%d8%a7%d8%b1%db%8c-%d8%a7%d9%86%da%a9%d9%88%d8%af%d8%b1rotary-encoder%d8%a8%d8%a7-%d8%b3/" \t "_blank)

1- پایه A0 برد اردوینو رو برای اتصال به پایه آنالوک سنسور رطوبت سنج در نظر گرفتیم

2- کتابخانه LiquidCrystal رو برای اتصال به LCD کاراکتری فراخوانی کردیم

3- مقادیر بدست آمده بازای خاک مرطوب و خاک خشک رو که در آزمایش اول بدست آورده بودیم توی دو تا متغیر ثابت soilWet  وsoilDry تعریف کردیم

4- یه خروجی برای اتصال به رله یا LED در نظر گرفتیم که میتونید طوری سورس برنامه رو تعریف کنید که بمحض اینکه رطوبت گلدان از مقدار خاصی کمتر شد رله عمل کنه و یه پمپ اب یا یه زنگ هشدار به صدا در بیاد

5- برنامه یک شی بنام lcd تعریف کردیم و محل قرار گرفتن پایه های LCD کاراکتری رو روی برد اردوینو تعریف کردیم

#define sensorPin A0

#include <LiquidCrystal.h> // Include LiquidCrystal Library

#define soilWet 600 // Define max value we consider soil 'wet'

#define soilDry 800 // Define min value we consider soil 'dry'

#define outPin 8

LiquidCrystal lcd(12, 11, 5, 4, 3, 2); // Create an LCD object.

توی بلوک void setup برنامه:

نوع LCD کاراکتری خودمون که 16\*2 هست رو تعریف میکنیم

LCD کاراکتری رو با دستور lcd.clear پاک میکنیم

با دستور lcd.print متن مورد نظر خودمونو  ” shasirobot.ir ” روی LCD کاراکتری نمایش میدیدم

یه تاخیر کوتاه ایجاد میکنیم و متن” shasirobot.ir “رو از روی ال سی دی پاک میکنیم

void setup()

{

lcd.begin(16, 2);

Serial.begin(9600);

lcd.clear();

lcd.print(" shasirobot.ir");

delay(2000);

lcd.clear();

}

توی بلوک void loop برنامه:

ابتدا مقدار پایه A0 برد آردوینو رو میخونیم و توی متغیر moisture  ذخیره میکنیم.

این مقدار رطوبت بدست آمده سنسور رو با مقادیر بدست آمده از آزمایش قبل توسط دستور if……..else مقایسه میکنیم و بازای 3 حالت خشکی گلدان ، رطوبت معمولی و غرغاب شدن گلدان پیام های خاصی رو روی ال اسی دی نمایش میدیم

یک تاخیر کوتاه ایجاد میکنیم تا داده ها روی LCD کاراکتری بدرستی نمایش داده بشن

در نهایت داده ها رو پاک میکنیم و این حلقه بارها و بارها تکرار میشه

void loop() {

int moisture = readSensor();

if (moisture < soilWet)

{

lcd.print("water Full");

}

else if (moisture >= soilWet && moisture < soilDry)

{

lcd.print("water Perfect");

}

else

{

lcd.print("water Low");

}

delay(1000); // Take a reading every second for testing

lcd.clear();

}