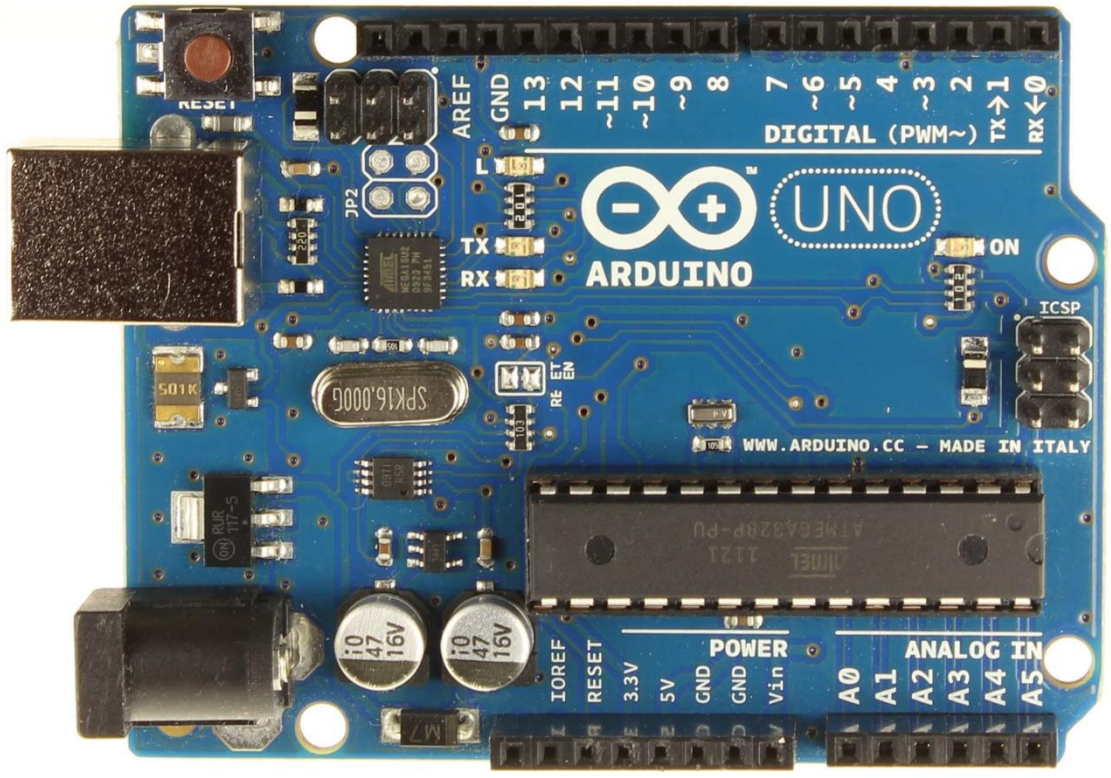
**آموزش اردوینو – فصل دوم ( اولین برنامه شما ، کار با LCD ، کار با پورت ها)**

**موضوعات فصل ۲**

- ۸ – دستورات قواعدی زبان آردوینو (سی/سی پلاس پلاس)
- ۹ – مراحل نوشتن یک برنامه جدید در آردوینو (بدنه یک برنامه)
- ۱۰ – دستورات مربوط به پورت ها در اردوینو (کار با پورت ها )
- ۱۱ – دستورات تاخیر در اردوینو
- ۱۲ –کار با LCD کارکتری در اردوینو (دستورات مربوط به راه اندازی و ... )

**تصویر زیر یک برد Arduino uno است ، پایه های این برد به سه قسمت تقسیم شده اند**



**1 – DIGITAL :** پایه های دیجیتال **14** تا هستند و از **0** شروع می شوند و با **13** به پایان می رسد در متن برنامه ها این پایه ها با اعدد **0** تا **13** فراخوانی و پیکربندی می شوند . در بین این **14** پایه بعضی از پایه ها **PWM** و یا خروجی ، ورودی سریال هستند . این پایه ها را می توان به صورت خروجی و وردی تعریف کرد . پایه های **3** ، **5** ، **6** ، **9** ، **10** ، **11** را می توان بعنوان **PWM** تعریف کرد . پایه **0** بعنوان ورودی سریال و پایه **1** بعنوان خروجی می تواند تعریف شود .

**2 – ANALOG IN :** این پایه ها ورودی مبدل انالوگ به دیجیتال هستند . **6** عدد ورودی آنالود داریم که از **A0** شروع می شود و با **A5** خاتمه می یابند ، از بین این **6** پایه ، پایه **A5** بعنوان **SCL** و پایه **A4** بعنوان **SDA** تعریف می شوند که در ارتباط **I2C** از آن استفاده می شود ، بعضی از اردوینو ها این دو پایه را جدا کرده و دو پین مستقل به آن اختصاص داده اند

**3 – POWER :** این پایه ها برای تامین ولتاژ قطعات و ماژول های جانبی مورد استفاده است

در تمام اردوینو ها ارایش پایه های به شکل بالا است تنها ممکن است شکل فیزیکی برد ها با هم تفاوت داشته باشد

**توجه :** برای شبیه سازی اردوینو با نرم افزار پروتیوس از این آموزش استفاده کنید .

## ۸ – دستورات قواعدی زبان آردوینو (سی/سی پلاس پلاس):

در هر کامپایلری که برای برنامه نویسی تعریف می شود مجموعه قواعدی وجود دارد که باید آن ها را رعایت کرد وگرنا کامپایلر از شما ایراد خواهد گرفت به قواعد زیر توجه کند .

**قانون ۱ :** بزرگی و کوچکی **:** کامپایر اردوینو نسبت به کوچکی و بزرگی متغیر ها حساس است و باید کوچکی و بزرگی متغییر ها و دستورات را رعایت کنیم

**قانون ۲ :** سمیکالون ; **:** باید در انتهای هر خط از برنامه نویسی سمیکالون (;) قرار دهیم **مثال :**

```
int a = 13;
```

**قانون ۳ :** آکولاد {} **:** در درون اکولاد {} دستورات اصلی نوشته می شود **مثال :**

```
void setup() {
  pinMode(13, OUTPUT);
}
```

**قانون ۴ :** تک خط کامنت // **:** برای اینکه در یک خط بعداز دستور مد نظر یا در خط جدید توضیحاتی را ذکر کنیم ابتدا // قرار می دهیم سپس توضیحات را بعداز ان می نویسیم و مجاز هستیم تنها یک خط توضیحات بنویسیم و اگر رفتیم خط بعدی مجدد // را در ابتدا بنویسیم **مثال :**

```
x = 5;  // This is a single line comment. Anything after the slashes is a comment
        // to the end of the line
```

**قانون ۵ :** چند خط کامنت /* */ **:** برای توضیحات تولانی که چندین سطر است از دستور چند خط کامنت استفاده می کنیم **مثال :**

```
x = 3;  /* but not another multiline comment - this is invalid */
```

## ۹ – مراحل نوشتن یک برنامه جدید در آردوینو ( بدنه یک برنامه ) :

پیکره اصلی یک برنامه در کامپایر آردوینو به شکل زیر است

```
void setup() {
  //(حلقه اول) تعریف متغیر
}

void loop() {
  // حلقه اصلی
}
```

**تعریف متغیر (حلقه اول) :** در این قسمت اطلاعاتی که یک بار مورد بررسی قرار میگیرد جای دارد . بعنوان مثال ، تعریف متغییر ، وضعیت پایه ها ، تعریف کتابخانه و ... در این بخش قرار می گیرند ، این بخش فقط یک بار و هنگام روشن کرد برد یا ریستارد کردن برد اجرا می شود . **مثال :**

```
void setup()
{
  pinMode(1, INPUT);
}
```

در مثال بالا پایه 1 بعنوان ورودی تعریف شده است ،

**حلقه اصلی :** در این حلقه برنامه اصلی که مرتبا تکرار می شود ، قرار می گیرد این حقه تا بی نهایت ادامه دارد و همیشه تکرار می شود . **مثال :**

```
void loop()
{
  if (digitalRead(1) == HIGH)
    Serial.write('H');
  else
    Serial.write('L');

  delay(1000);
}
```

در مثال بالا اگر کلید متصل به پایه 1 برابر یک شد در خروجی سریال حرف H ارسال می شود و اگر مساوی یک نباشد حرف L ارسال می شود .

## ۱۰ – دستورات مربوط به پورت ها در اردوینو (کار با پورت ها ) :

برای استفاده از پایه ها در آردوینو باید آن ها را به صورت خروجی و وردی پیکربندی کرد .

یک پایه هنگامی به عنوان خروجی تعریف می شود که بخواهیم از آن ولتاژ بگیریم و یک پایه هنگامی به عنوان ورودی قرار می گیرد که بخواهیم به آن ولتاژ بدهیم ، ( مثلا هنگامی که قصد داریم کلیدی را به میکرو متصل کنیم ، باید یک سر کلید را به ۵ ولت و سر دیگر آن را به پایه های میکرو متصل نماییم ، هنگامی که کلیدی فشرده می شود ولتاژ ۵ ولت به پایه ی میکرو اعمال شده و آردوینو متوجه تغییر وضعیت می شود ، گرفتن ولتاژ نیز همچون یک LED (چراغ) است که بین گراند (صفر ولت) و یکی از پایه های آردوینو متصل شده و می تواند یک وضعیت خروجی را نمایش دهد )

### برای خروجی یا ورودی قرار دادن یکی از پایه ها از دستور زیر استفاده می شود :

```
pinMode(pin, mode)
```

این دستور در حلقه اول قرار میگیرد و متغییر های آن به شکل زیر هستند

Pin : یکی از پایه های اردوینو است و می تواند عدد ۰ تا ۱۳ باشد . ( البته در صورت نیاز می تواند از ۰ تا ۲۴ باشد)

Mode : وضعیت پایه اردوینو می باشد که می خواهیم بعنوان ورودی یا خروجی تعریف کنیم و می تواند به شکل های زیر تعریف شود

INPUT : هنگامی که بخواهیم پایه ای را بعنوان ورودی تعریف کنیم

OUTPUT : هنگامی که بخواهیم پایه ای را بعنوان خروجی تعریف کنیم

INPUT_PULLUP : هنگامی که به پایه ای بعنوان ورودی و برای اتصال کلید به کار برود

### مثال :

```
pinMode(0, INPUT_PULLUP);
pinMode(13, INPUT);
pinMode(1, OUTPUT);
```

### دستورات مربوط به پورت ها :

| دستور digitalWrite |
|---|

digitalWrite() : از این دستور هنگامی استفاده می شود که یک پایه بعنوان خروجی تعریف شده باشد ، به وسیله این دستور می توان یک پایه خاص ( پایه 0 تا 13 ) را 0 ولت یا 5 ولت کرد (HIGH or LOW) ، هنگامی از این دستور استفاده می شود که اگر یک مصرف کننده مانند LED به پایه مد نظر وصل شده باشد

r

فرم کلی دستور به شکل زیر است :

digitalWrite(pin, value)

pin : شماره پایه اردوینو ( از 0 تا 13 )

Value : وضعیت پایه که می تواند HIGH یا LOW باشد (HIGH= 5V and LOW=0V)

## دستور digitalRead

digitalRead() : این دستور وقتی استفاده می شود که یک پایه بعنوان ورودی تعریف شده باشد و وظیفه آن دریافت مقدار دیجیتال ( 0 یا 1) از بیرون است ، بعنوان مثال وقتی کلیدی به پایه ای از اردوینو وصل می شود و می خواهیم از وضعیت فشره شدن کلید مطلع شویم از این دستور استفاده می کنیم

فرم کلی دستور به شکل زیر است :

digitalRead(pin)

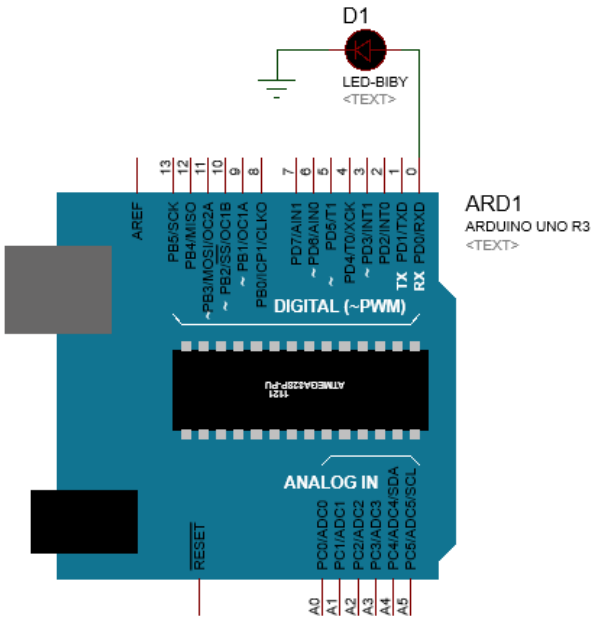
pin : پایه ای از اردوینو ( 0 تا 13 ) که به آن کلید (یا وسیله ای دیگر) وصل است

## مثال های مربوط به پورت ها

**مثال ۱ :** استفاده از دستور OUTPUT و ()digitalWrite

در مثال زیر نحوه استفاده از دستور OUTPUT و digitalWrite() آموزش داده شده است ، در این مثال پایه 0 اردوینو را بعنوان خروجی معرفی می کنیم سپس یک led را به پایه 0 اردوینو وصل می کنیم و آن را روشن و خاموش می کنیم . (چشمک زن)

```
void setup() {
  pinMode(0, OUTPUT);
}
void loop() {
  digitalWrite(0, HIGH);
  delay(1000);
  digitalWrite(0, LOW);
  delay(1000);
}
```



در مثال بالا ابتدا با دستور `pinMode(0, OUTPUT)` پایه 0 از آردوینو بعنوان خروجی تعریف می شود

با دستور `digitalWrite(0, HIGH)` پایه 0 از اردوینو 5 ولت (HIGH) می شود .

با دستور `delay(1000)` به مدت ۱۰۰۰ میلی ثانیه یا ۱ ثانیه تاخیر ایجاد می کنیم

با دستور `digitalWrite(0, LOW)` پایه 0 از اردوینو 0 ولت (LOW) می شود .

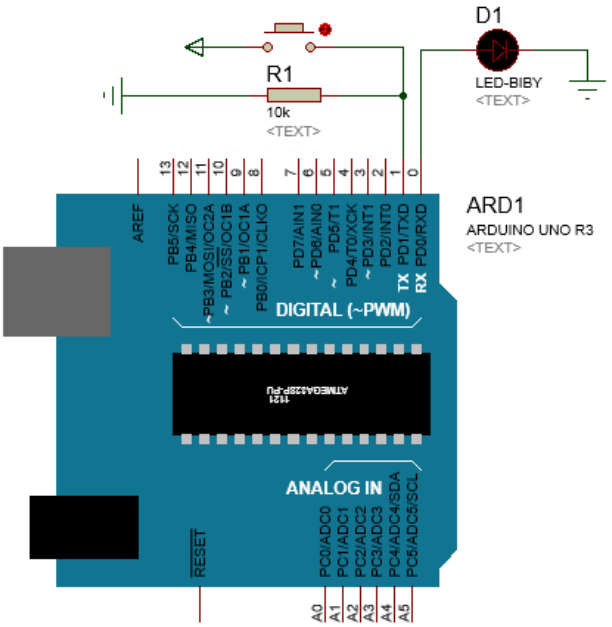
**مثال ۲ :** استفاده از دستور INPUT و () digitalRead

در مثال زیر نحوه استفاده از دستور INPUT و () digitalRead اموزش داده می شود در این مثال پایه 0 اردوینو را بعنوان خروجی معرفی می کنیم سپس یک led را به پایه 0 اردوینو وصل می کنیم سپس پایه 1 اردوینو را بعنوان وردی تعریف می کنیم و به آن یک کلید وصل می کنیم و برنامه ای می نویسیم که با فشردن کلید LED روشن شود

```
void setup() {
  pinMode(0, OUTPUT);
  pinMode(1, INPUT);
}

void loop() {
if ( digitalRead(1) == HIGH)
{
    digitalWrite(0, HIGH);
  }

}
```




در مثال بالا ابتدا با دستور `pinMode(0, OUTPUT)` پایه 0 از آردوینو بعنوان خروجی تعریف می شود

سپس با دستور `pinMode(1, INPUT)` پایه 1 از آردوینو بعنوان ورودی تعریف می شود

با دستور `digitalRead(1)` وضعیت پایه ۱ که به آن کلید وصل است معلوم میشود و با استفاده از دستور شرطی به فرم زیر در صورت فشرن کلید LED روشن می شود (در باره دستورات شرطی در جلسات اینده بصورت کامل توضیح داده خواهد شد)

```
if ( digitalRead(1) ==
HIGH) {
    digitalWrite(0, HIGH);
  }
```

در مثال بالا می بینید که همراه کلید یک مقاومت ۱۰ کیلو به پین مد نظر وصل شده است ، به این مقاومت مقاومت پول اپ می گویند و در صورت فشره نبودن کلید، کلید را در حالت 0 ولت نگه میدارد ، در صورتی که این مقاومت

r

وصل نباشد کلید در اثر نویز محیط به حالت نامتعادل 0 و ولتاژ بالاتر نوسان می کند و باعث می شود کلید عمل نکند و مدار دچار اختلال شود . در اردوینو دستوری وجود داره که اگر از ان استفاده کنیم پول اپ اخلی میکرو را فعال می کند و لازم نیست دیگه به کلید مقاومت وصل کنیم به مثال زیر توجه کنید .

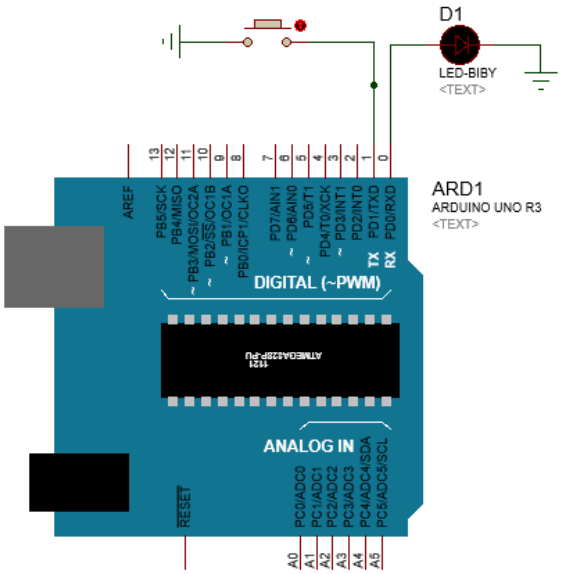
**مثال ۳ :** استفاده از دستور INPUT_PULLUP و () digitalRead

این مثال دقیقا مثال ۲ است که بجای استفاده از دستور INPUT از دستور INPUT_PULLUP استفاده می شود و مقاومت پول اپ حذف شده و کلید به GND وصل می شود .

```
void setup() {
  pinMode(0, OUTPUT);
  pinMode(1, INPUT_PULLUP);
}

void loop() {
if ( digitalRead(1) == LOW) {
    digitalWrite(0, HIGH);
  }

}
```




دستور شرطی در این مثال این گونه است که اگر پایه ۱ صفر ولت شود انگاه LED روشن شود .

## ۱۱ – دستورات تاخیر در اردوینو

از دستورات تاخیر برای ایجاد توقف موقت در برنامه استفاده می شود

| دستور delay |
|---|

delay(ms) : از این دستور برای ایجاد تاخیر میلی ثانیه ای ایجاد می شود

( یک ثانیه ۱۰۰۰ میلی ثانیه است )

ms : تعدا میلی ثانیه می باشد ، مثال:

delay(1000); تاخیر یک ثانیه ای

delay(500); تاخیر نیم ثانیه ای

r

## دستور delayMicroseconds

**delayMicroseconds(us) : از این دستور برای ایجاد تاخیر میکرو ثانیه ای ایجاد می شود**

**us : تعداد میکرو ثانیه می باشد ، مثال:**

delayMicroseconds (1000); **تاخیر ۱۰۰۰ میکرو ثانیه ای**

delayMicroseconds (500); **تاخیر ۵۰۰ میکرو ثانیه ای**

## دستور micros

**micros () : این دستور بعداز پروگرام و روشن شدن اردوینو شروع به شمردن می کند و بصورت مستقل کار می کند و بعداز ۷۰ دقیقه سر ریز می شود ( دوباره صفر می شود)**

## دستور millis

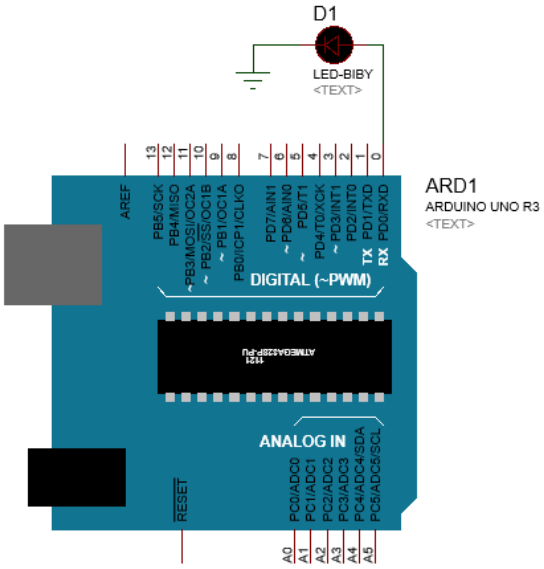
**millis () : این دستور بعداز پروگرام و روشن شدن اردوینو شروع به شمردن می کند و بصورت مستقل کار می کند و بعداز ۵۰ روز سر ریز می شود ( دوباره صفر می شود)**

**از این دو دستور millis و micros مانند تایمر های میکرو در بیسیک و کدویژن استفاده می شود**

## مثال های مربوط به دستورات تاخیر

**مثال ۴ :** استفاده از دستور delay

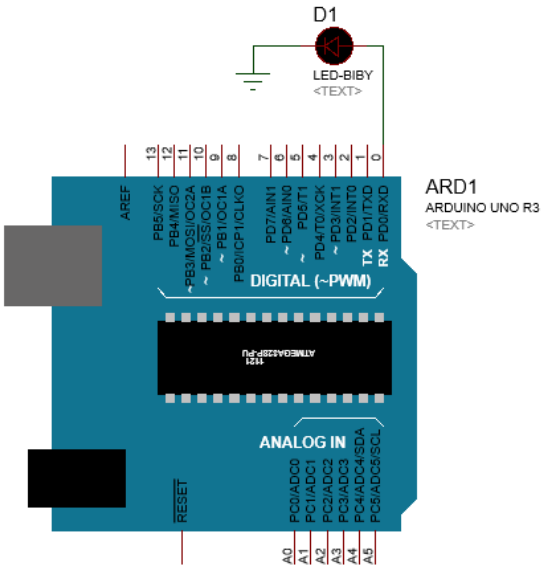
```
void setup() {
  pinMode(0, OUTPUT);
}
void loop() {
  digitalWrite(0, HIGH);
  delay(200);
  digitalWrite(0, LOW);
  delay(200);
}
```




**در مثال بالا LED با سرعت ۲۰۰ میلی ثانیه روشن و خاموش می شود .**

**مثال ۵ :** استفاده از دستور delayMicroseconds

```
void setup() {
  pinMode(0, OUTPUT);
}
void loop() {
  digitalWrite(0, HIGH);
DelayMicroseconds(1500);
  digitalWrite(0, LOW);
DelayMicroseconds(1500);
}
```




**در مثال بالا LED با سرعت ۱۵۰۰ میکرو ثانیه روشن و خاموش می شود .**

**مثال ۶ :** استفاده از دستور **millis** و **micros** **بعنوان** **مثال در مورد micros :**

**این دو دستور مشابه هم هستند و تنها زمان سر ریز شدن (صفر شدن ) انها با هم تفاوت دارد . اگر A را بعنوان یک متغییر تعریف کنیم از این دستور به شکل زیر استفاده می شود**

```
A = micros();
```

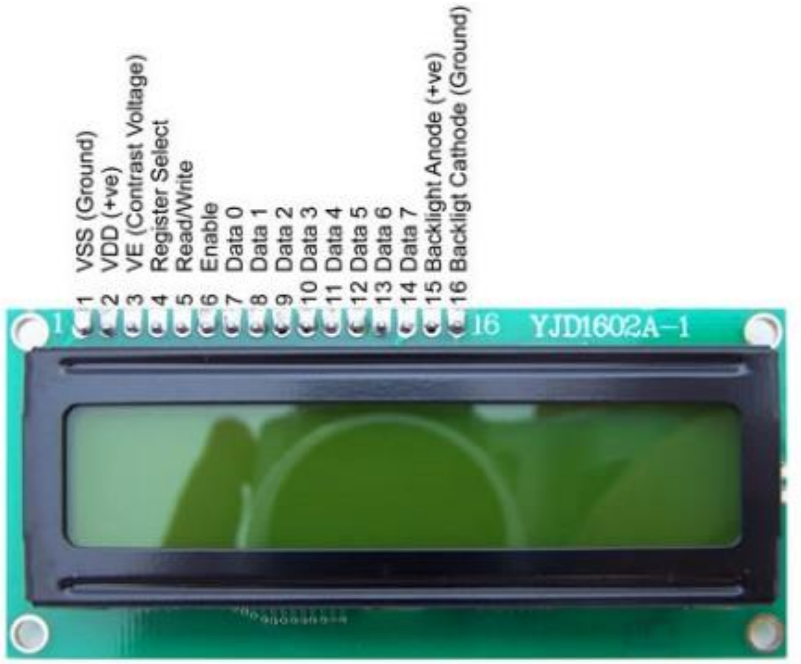
**در دستور بالا بعداز روشن شدن اردوینو زمان در متغییر A ریخته می شود.**

**برای این دستور در جلسات بعدی مثال عملی آورده می شود تا برای شما ملموس تر گردد .**

## ۱۲ –کار با LCD کارکتری در اردوینو (دستورات مربوط به راه اندازی و ... )

**از ال سی های کارکتری برای نمایش متن و کارکتر و ... استفاده می شود ال سی دی های کارکتری در اندازه های مختلف وجود دارند و مبنای اندازه انها سطر و ستون LCD است که در**

**اندازه های 1*16 ، 2*16 ، 2*20 ، 4*20 و ... وجود دارند برای راه اندازی LCD کارکتری به شکل زیر عمل می شود ، پایه های هر ال سی دی کارکتری به شکل زیر هستند**




پایه هایLCD

| پایه | نام | عملکرد |
|---|---|---|
| 1 | VSS | زمین |
| 2 | VCC | 5V+ |
| 3 | VEE | کنترل درخشندگی (می توانید با یک مقاومت ۲ کیلو یا پتانسومتر ۱۰ کیلو آن را زمین کنید) |
| 4 | RS | اگر این پایه ۰ باشد اطلاعات روی DB0-DB7 به عنوان فرمان و اگر ۱ باشد به عنوان کاراکتر پذیرفته می شود |
| 5 | R/W | اگر این پایه ۰ باشد LCD برای نوشتن آماده می شود و اگر ۱ باشد برای خواندن آماده می شود |
| 6 | E | فعال سازی LCD که با یک لبه پایین رونده می باشد |
| 7 | DB0 | استفاده نمیشود |
| 8 | DB1 | استفاده نمیشود |
| 9 | DB2 | استفاده نمیشود |
| 10 | DB3 | استفاده نمیشود |
| 11 | DB4 | خطوط دیتا |
| 12 | DB5 | خطوط دیتا |
| 13 | DB6 | خطوط دیتا |
| 14 | DB7 | خطوط دیتا |
| 15 | A | 5V+ از پایه ۱۵ و ۱۶ برای روشن کردن LED پس زمینه استفاده می شود |
| 16 | K | زمین |

r

برای راه اندازی ال سی های کارکتری در اردوینو به شکل زیر عمل می شود ،

ابتدا باید تابع مربوط به ال سی دی را با دستور زیر فراخوانی کنیم این دستور نباید داخل حلقه ها باشد و باید در ابتدای کد ها نویشته شود :

```
#include <LiquidCrystal.h>
```

بعداز دستور بالا پایه هایی از اردوینو که باید به ال سی دی متصل شوند مشخص شود :

```
LiquidCrystal lcd(RS, E, D4, D5, D6, D7);
```

در دستور بالا ترتیب پایه های ال سی دی نوشته شده است ، شما بجای نام پایه های lcd شماره پایه های اردوینو که پین ال سی دی به آن متصل شده است را وصل کنید . بعداز دستور بالا شما در حلقه اول دستور زیر را قرار دهید که مشخص کننده اندازه ال سی دی است مثلا مثال زیر مربوط به LCD2*16 است .

```
lcd.begin(16, 2);
```

کار های بالا را باید برای هر ال سی دی که بخواهیم فعال کنیم انجام دهیم ، حالا نوبت به دستورات مربوط به ال سی دی است که در حلقه اصلی و بی نهایت برنامه نوشته می شود.

## دستورات مربوط به کار با LCD های کارکتری :

`lcd.print("TEXT")` : از این دستور برای نمایش اعداد و حروف و علایم و ... استفاده می شود

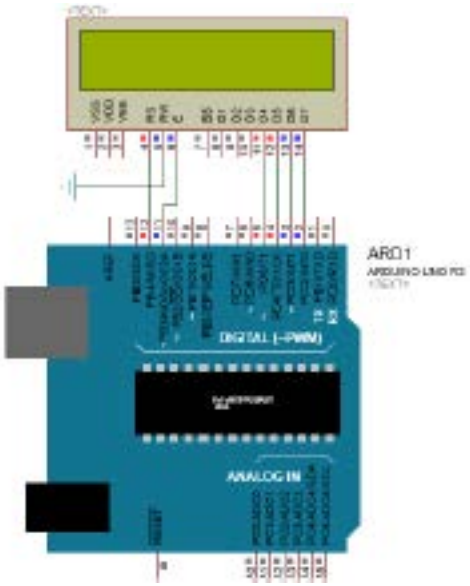
`lcd.setCursor(X, Y)` : از این دستور برای مشخص کردن محل متن در ال سی دی استفاده می شود که عدد اول (X) مربوط به ستون و بسته به نوع ال سی دی که چند در چند است تعیین می شود ، مثلا در LCD2*16 می تواند از ۰ تا ۱۶ باشد و عدد دوم (Y) مربوط به سطر است و بسته به نوع ال سی دارد ، مثلا در مثال ذکر شده که تنها دو سطر دارد می تواند ۰ یا ۱ باشد که ۰ یعنی سطر اول و ۱ یعنی سطر دوم

**مثال ۷ :** نمایش متن و کارکتر و اعداد و ... در LCD

عبارت computer را در سطر اول و electronic را در سطر دوم ال سی دی نمایش دهید . **در برنامه زیر خط**

```
LiquidCrystal lcd(12, 11, 5, 4, 3, 2);
```

مشخص می کنند که پایه های ال سی دی به کدام پایه های اردوینو متصل می شوند .



```
#include <LiquidCrystal.h>
LiquidCrystal lcd(12, 11, 5, 4, 3, 2);

void setup() {
lcd.begin(16, 2);
}

void loop() {
  lcd.setCursor(0, 0);
   lcd.print("computer");

  lcd.setCursor(0, 1);
 lcd.print("electronic");

 delay(500);
}
```

## سایر دستورات ال سی دی :

**lcd.blink()** : ا قرار دادن این دستور در حلقه اصلی مکان نما شروع به چشمک زدن می کند

**lcd.noBlink()** : با قرار دادن این دستور در حلقه اصلی مکان نما خاموش می شود

**lcd.clear()** : با این دستور همه کارکتر های ال سی دی پاک می شود (همان دستور CLS در بسکام است)

**lcd.noDisplay()** : برای خاموش کردن ال سی دی استفاده می شود و در حلقه اصلی قرار می گیرد

**lcd.display()** : برای روشن کردن ال سی دی استفاده می شود و در حلقه اصلی قرار می گیرد

**lcd.noCursor()** : برای خاموش کردن مکان نما اسفاده می شود

**lcd.cursor()** : برای روشن کردن مکان نما استفاده می شود

**lcd.scrollDisplayLeft()** : برای حرکت متن بصورت افقی از سمت راست به سمت چپ

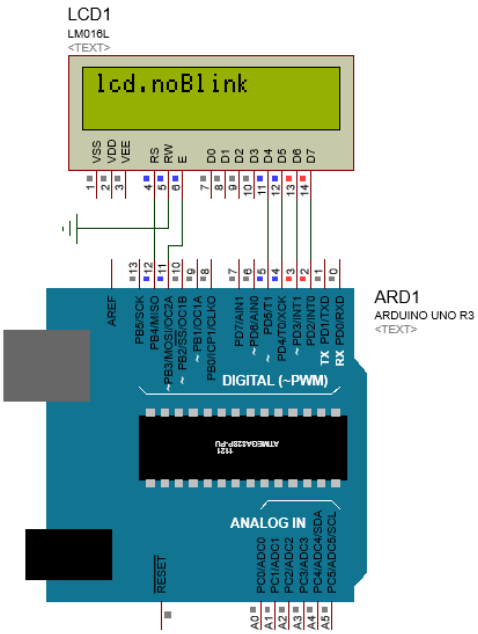
**lcd.scrollDisplayRight()** : برای حرکت متن بصورت افقی از سمت چپ به سمت راست

**مثال ۸ :** در مثال زیر دستورات **lcd.blink** ، **lcd.noBlink** ، **lcd.clear** ، **lcd.noDisplay** ، **lcd.display** ، **lcd.noCursor** و **lcd.cursor** مورد استفاده قرار گرفته اند ، کد های مربوط به هر دسته را با رنگ متفاوت مشخص کرده ایم

```
#include <LiquidCrystal.h>
LiquidCrystal lcd(12, 11, 5, 4, 3, 2);

void setup() {
lcd.begin(16, 2);
}

void loop() {
// lcd.blink
lcd.setCursor(0, 0);
lcd.print("lcd.blink");
lcd.blink();
delay(2000);
lcd.clear();
//lcd.noBlink
lcd.setCursor(0, 0);
lcd.print("lcd.noBlink");
lcd.noBlink();
delay(2000);
lcd.clear();
//lcd.cursor
lcd.setCursor(0, 0);
lcd.print("lcd.cursor");
lcd.cursor();
delay(2000);
lcd.clear();
//lcd.nocursor
lcd.setCursor(0, 0);
lcd.print("lcd.nocursor");
lcd.noCursor();
delay(2000);
lcd.clear();
//noDisplay
lcd.noDisplay();
delay(2000);
//Display
lcd.display() ;
}
```




**مثال ۹ :** در مثال زیر دستورات **lcd.scrollDisplayRight()** و **lcd.scrollDisplayLeft()** مورد استفاده قرار گرفته ، البته کار کرد این دو دستور دقیقا شبیه هم است واولی کارکتر هارو به راست حرکت می دهد و دومی کارکتر ها را به چپ حرکت می دهد ، در انجا مثالی از دستور اول اورده شده است .

r

```
#include <LiquidCrystal.h>
LiquidCrystal lcd(12, 11, 5, 4, 3, 2);

void setup() {
lcd.begin(16, 2);
}

void loop() {
 lcd.setCursor(0, 0);
lcd.print("Computer ");
lcd.scrollDisplayRight();
 delay(200);
}
```