جَامعة
المَنارة
MANARA UNIVERSITY

المتحكمات الصغرية والنظم المضمنة

محاضرة 2

الإخراج التشابهي

تعديل عرض النبضة PWM

د. فادي متوج

تعديل عرض النبضة
Pulse Width Modulation
analogWrite()

- يعد التحكم الرقمي في الأرجل أمرًا رائعًا لإضاءة الليدات، والتحكم في الريليهات ، و دوران المحركات بسرعة ثابتة.
- ولكن ماذا لو أردنا إخراج جهد غير 0 فولت أو 5 فولت؟

  لا يمكننا ذلك - ما لم نستخدم دارة خارجية DAC للتحويل من رقمي إلى تشابهي.
- ومع ذلك، يمكننا الاقتراب جدًا من توليد قيم خرج تشابهية باستخدام خدعة تسمى تعديل عرض النبضة PWM .

تعديل عرض النبضة
Pulse Width Modulation
analogWrite()

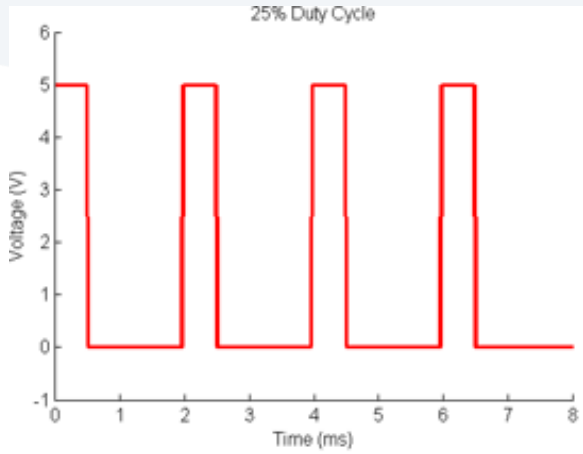
- في كل بورد Arduino، توجد أرجل يمكنها استخدام الأمر **analogWrite ()** لتوليد إشارات PWM التي يمكنها محاكاة إشارة تشابهية نقية عند استخدامها مع أجهزة طرفية معينة.
- يتم تمييز هذه الأرجل بعلامة ~ على البورد.
- على Arduino Uno، الأرجل 3 و 5 و 6 و 9 و 10 و 11 هي أرجل PWM
- يقبل الأمر **analogWrite ()** معاملين : الرجل التي يجب التحكم فيها والقيمة المراد إخراجها عليها.
- الخرج PWM هو قيمة تقع في المجال من 0 إلى 255.

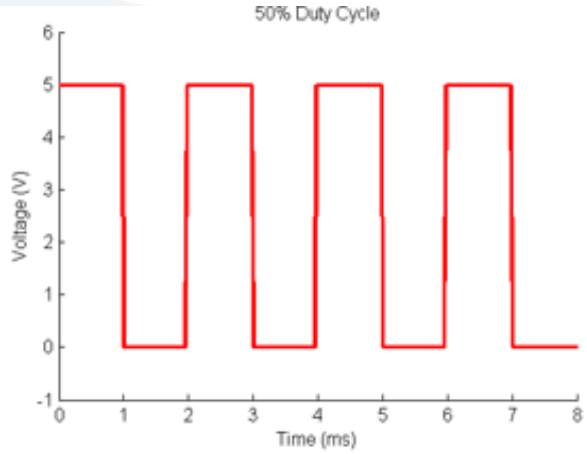
استخدم PWM لتغيير شدة سطوع LED

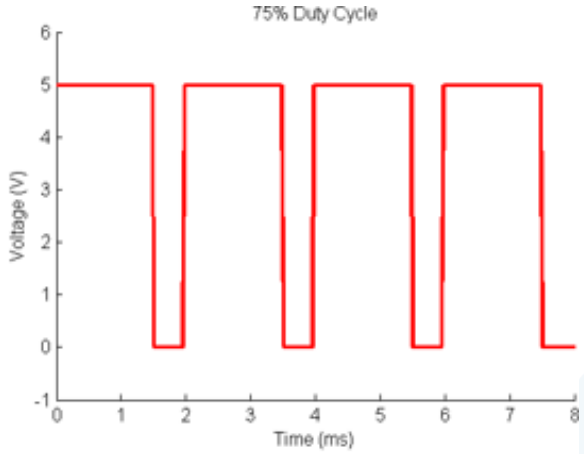
```
const int LED=9;    //define LED for Pin 9
void setup() {
pinMode (LED, OUTPUT);     //Set the LED pin as an output
}
void loop() {
for (int i=0; i<256; i++)
{
analogWrite(LED, i);
delay(10);
}
for (int i=255; i>=0; i--)
{
analogWrite(LED, i);
delay(10); } }
```

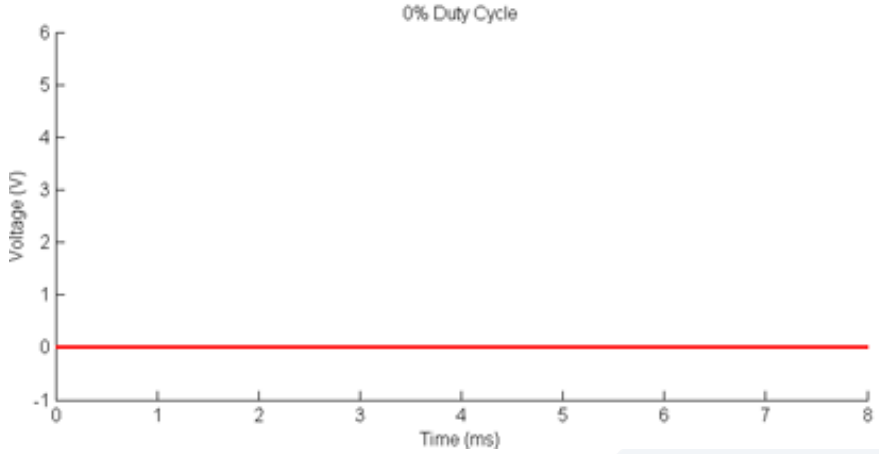
ماذا يفعل الـ LED عند تنفيذ الكود السابق؟

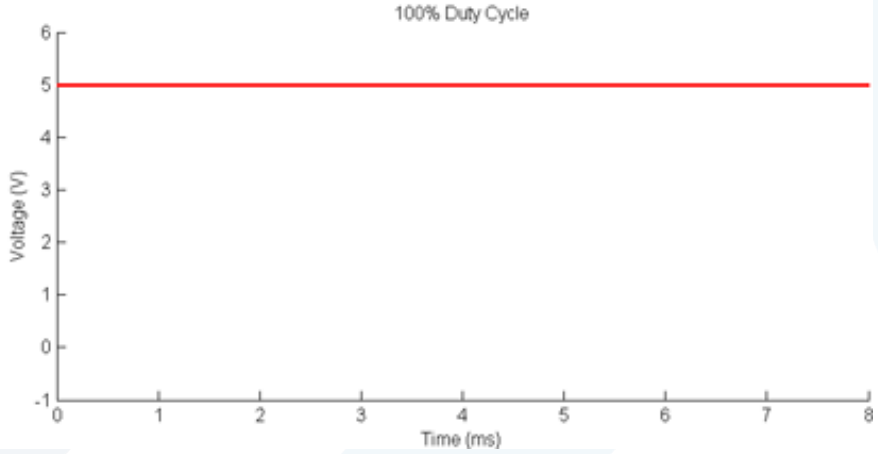
- يجب أن نلاحظ ازدياد تدريجي في شدة سطوع الليد من كونه مطفأ في البداية وصولاً إلى السطوع الكامل، ثم بالعكس انخفاض تدريجي في شدة سطوع الليد من حالة السطوع الكامل وصولاً إلى انطفائه بشكل كامل.
- بالطبع ، لأن التعليمات كلها موجودة في الحلقة الرئيسية loop، فإن هذا السيناريو يتكرر.











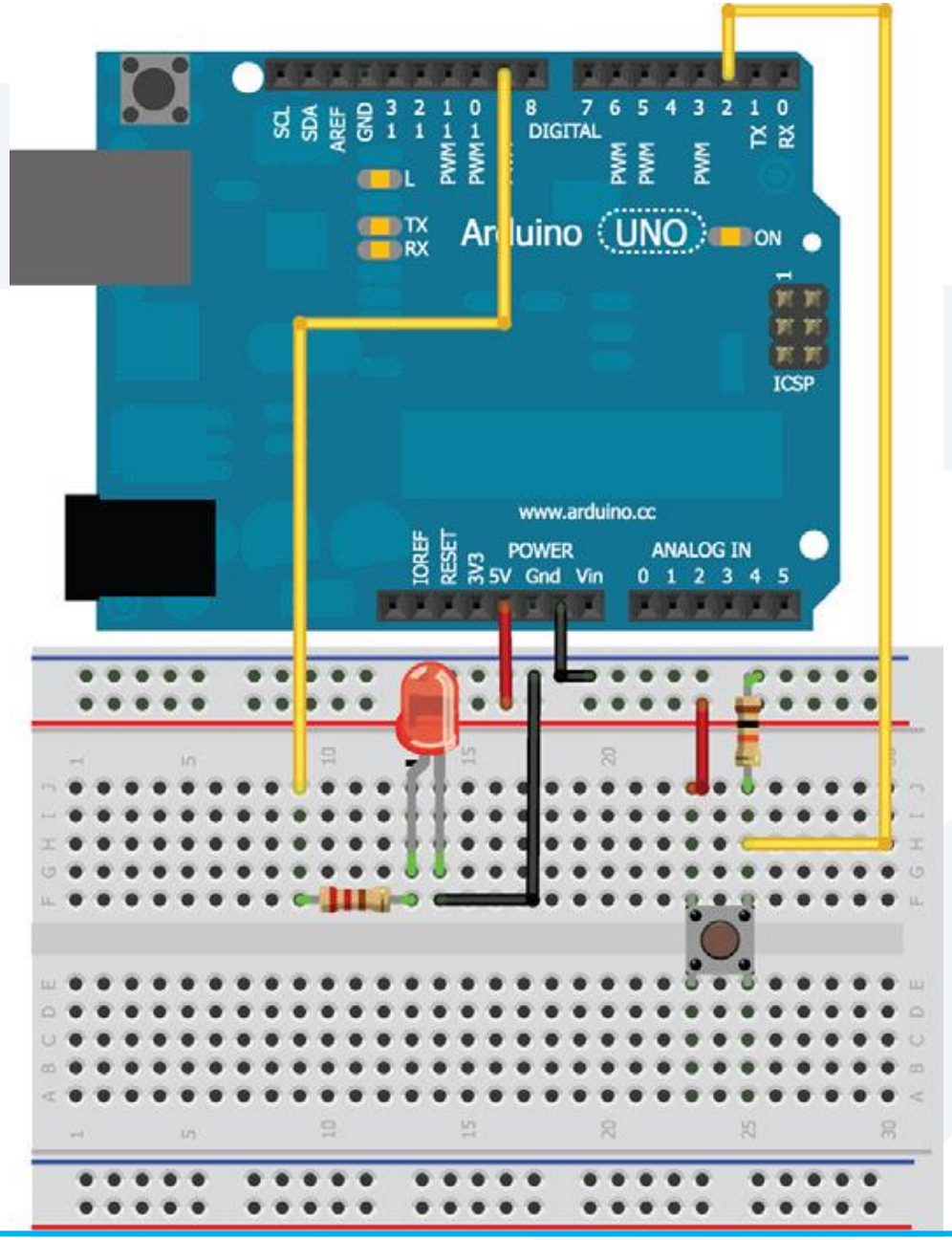
- تعمل تقنية PWMعن طريق تعديل دورة عمل الموجة المربعة.
- تشير دورة العمل (Duty cycle) إلى النسبة المئوية من الزمن التي تكون فيها الموجة المربعة بحالة HIGH بالنسبة لكامل دور الإشارة.
- مثلاً الموجة المربعة التي لها دورة عمل بنسبة 50٪ تكون في حالة HIGH خلال نصف الدور، وفي حالة LOW في نصف الدور.

- يحدد الأمر **analogWrite ()** دورة عمل الموجة المربعة اعتمادًا على القيمة التي نمررها إليه:
  - كتابة قيمة 0 باستخدام analogWrite () تشير إلى موجة مربعة مع دورة عمل بنسبة 0 بالمائة (دائمًا LOW).
  - تشير كتابة 255 إلى موجة مربعة مع دورة عمل بنسبة 100 بالمائة (دائمًاHIGH).
  - تشير كتابة 127 إلى موجة مربعة مع دورة عمل بنسبة 50 بالمائة (نصف الزمن HIGH و نصف الزمن LOW).
- بالنسبة للإشارة ذات دورة العمل بنسبة 25 بالمائة ، تكون بحالة HIGH بنسبة 25 بالمائة من الزمن، وبحالة LOW بنسبة 75 بالمائة من الزمن.
- تردد هذه الموجة المربعة ، في حالة Arduino، يبلغ حوالي **490 هرتز**. بمعنى آخر ، تتناوب الإشارة بين HIGH (5 فولت) وLOW (0 فولت) حوالي 490 مرة كل ثانية.
- لنطرح السؤال التالي : طالما لم نقم بالفعل بتغيير الجهد الذي يتم توصيله إلى LED، فلماذا نرى أن الـ LED يصبح باهتًا عندما نخفض دورة العمل؟ في الحقيقة يحصل هذا نتيجة لعب أعيننا خدعة علينا! إذا كان الـ LED يضيء ويطفئ كل 1 مللي ثانية (وهذا هو الحال مع دورة العمل بنسبة 50 بالمائة) ، فيبدو أنه يعمل بنصف سطوع تقريبًا لأنه يومض بشكل أسرع مما يمكن أن تدركه أعيننا. لذلك ، يقوم دماغنا في الواقع بتقدير متوسط الإشارة ويخدعنا للاعتقاد بأن LEDيعمل بنصف سطوع.

قراءة المدخلات الرقمية

- بعد أن درسنا كل من الخرج الرقمي والتشابهي. سنقوم بدراسة كيفية قراءة المدخلات الرقمية ، مثل المفاتيح والأزرار.

توصيل السويتش باستخدام مقاومة Pulldown



```
const int LED=9;          //The LED is connected to pin 9
const int BUTTON=2;   //The Button is connected to pin 2
void setup() {
pinMode (LED, OUTPUT);       //Set the LED pin as an output
pinMode (BUTTON, INPUT);   //Set button as input (not required
}
void loop(){
if (digitalRead(BUTTON) == LOW)
   {
     digitalWrite(LED, LOW);
    }
else
   {digitalWrite(LED, HIGH); }
}
```

قراءة قيم الحساسات التشابهية

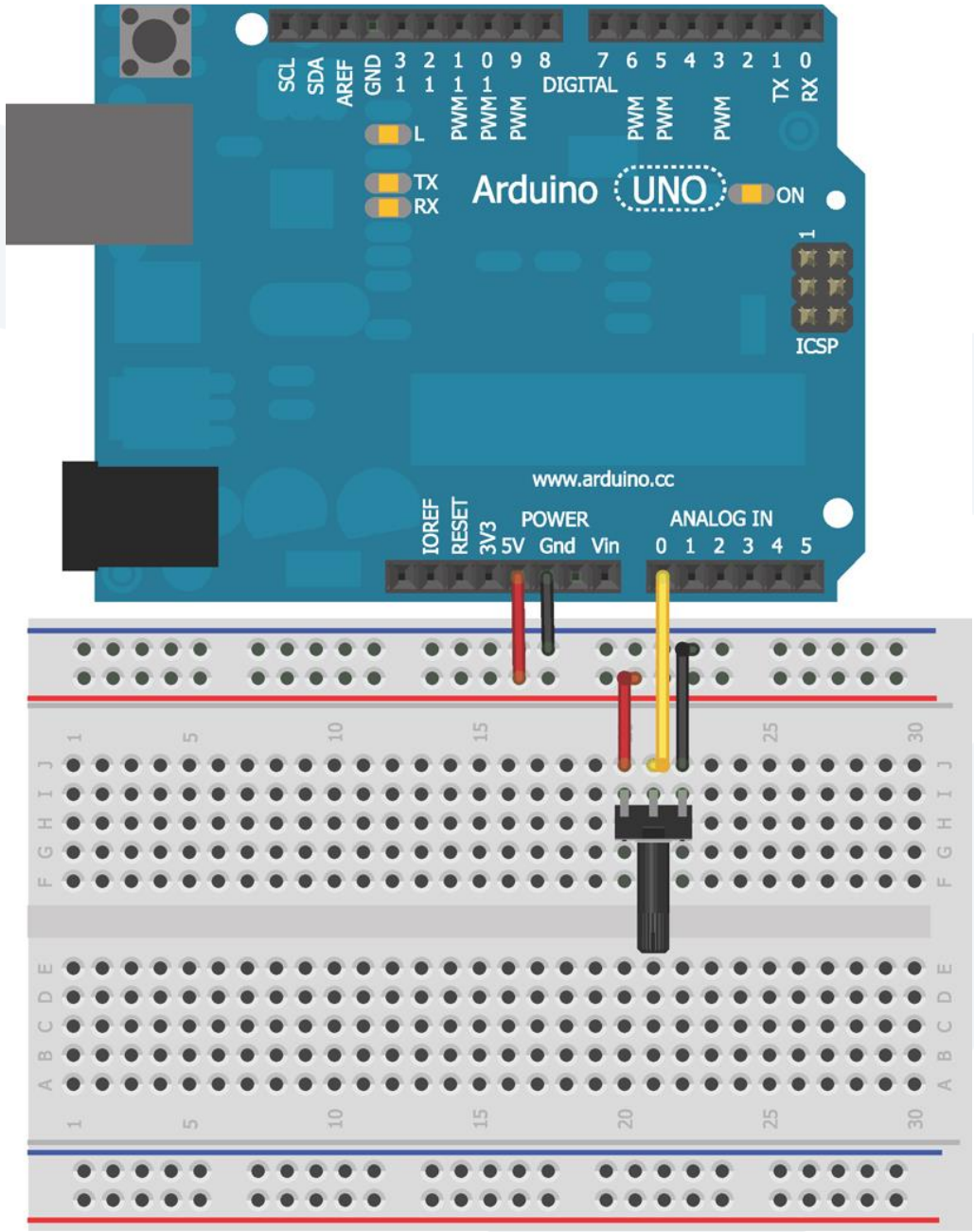
- العالم من حولنا تشابهي. على الرغم من أننا قد نسمع أن العالم "يتحول إلى عالم رقمي" ، فإن غالبية الظواهر التي يمكن ملاحظتها في عالمنا هي دائمًا تشابهية في طبيعتها.
- يمكن أن نذكر عدة أمثلة عن الإشارات التشابهية مثل شدة إشعاع الشمس، أو درجة حرارة مياه المحيط ، أو تركيز الملوثات في الهواء.
- للتعامل مع هذه الإشارات التشابهية يجب تحويلها إلى قيم رقمية يمكن معالجتها باستخدام متحكم مثل Arduino

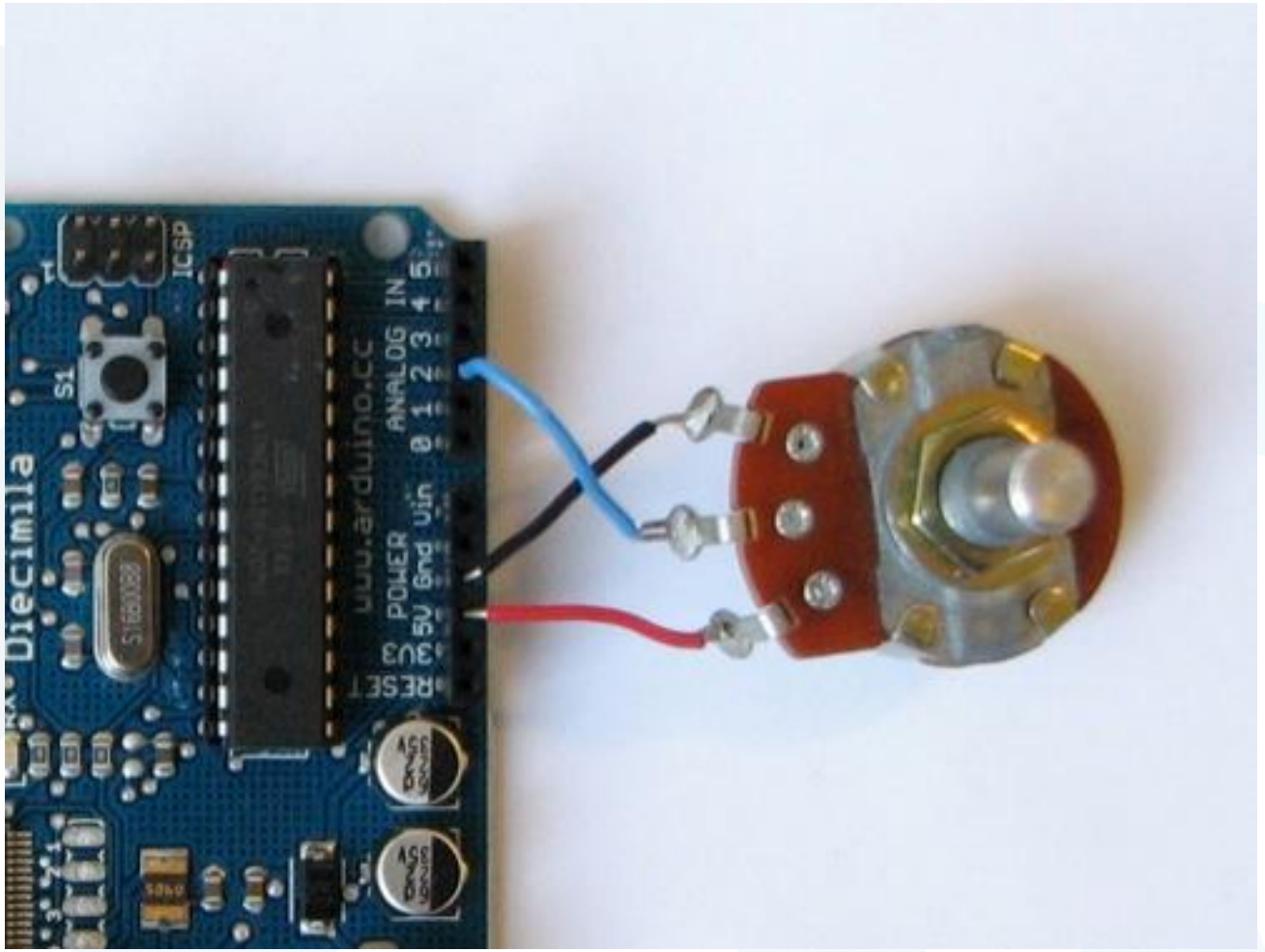
تحويل الإشارة التشابهية إلى رقمية

- يمكننا استخدام المحول التشابهي إلى رقمي (ADC) المتضمن في Arduino لتحويل قيم الجهد التشابهي إلى تمثيل رقمي يمكننا العمل معه.
- في حالة Arduino Uno، يوجد ADC بدقة 10بت لإجراء التحويل من تشابهي إلى رقمي.
- "10 بت" يعني أن ADC يمكنه تقسيم إشارة تشابهية إلى 1024 قيمة مختلفة، وبالتالي يمكن لـ Arduinoتحديد قيمة رقمية من **0** إلى **1023** لأي قيمة تشابهية نعطيها له.
  - يؤدي وضع 0 فولت على دخل ADC إلى إرجاع قيمة 0 رقمي،
  - يؤدي وضع 2.5 فولت على دخل ADC إلى إرجاع قيمة 512 (نصف 1024)
  - يؤدي وضع 5 فولت على دخل ADC إلى إرجاع قيمة 1023.

مثال1 : قراءة قيمة مقاومة متغيرة (إشارة تشابهية)

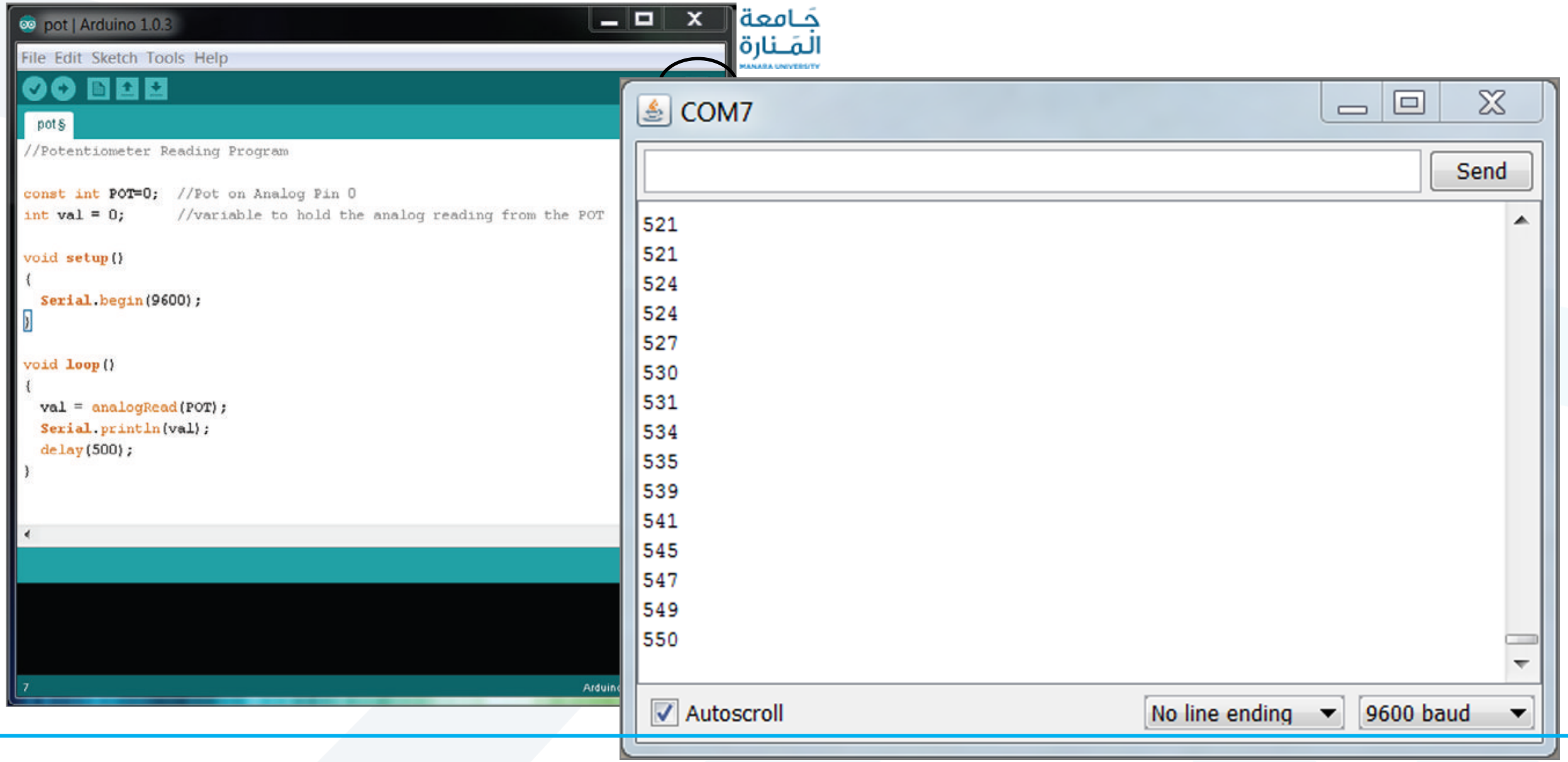
- المقاومات المتغيرة هي مقسمات جهد متغيرة تأتي بأحجام وأشكال كثيرة، لكن لديها جميعًا ثلاثة أرجل.
  - نقوم بتوصيل أحد الأرجل الخارجية بالأرضي والآخر بـ 5 فولت.
  - المقاومات المتغيرة متناظرة ، لذلك لا يهم الجانب الذي توصل به 5 فولت والأرضي.
  - نقوم بتوصيل الرجل الوسطى بأحد أرجل الدخل التشابهي (A0 مثلاً) على Arduino.

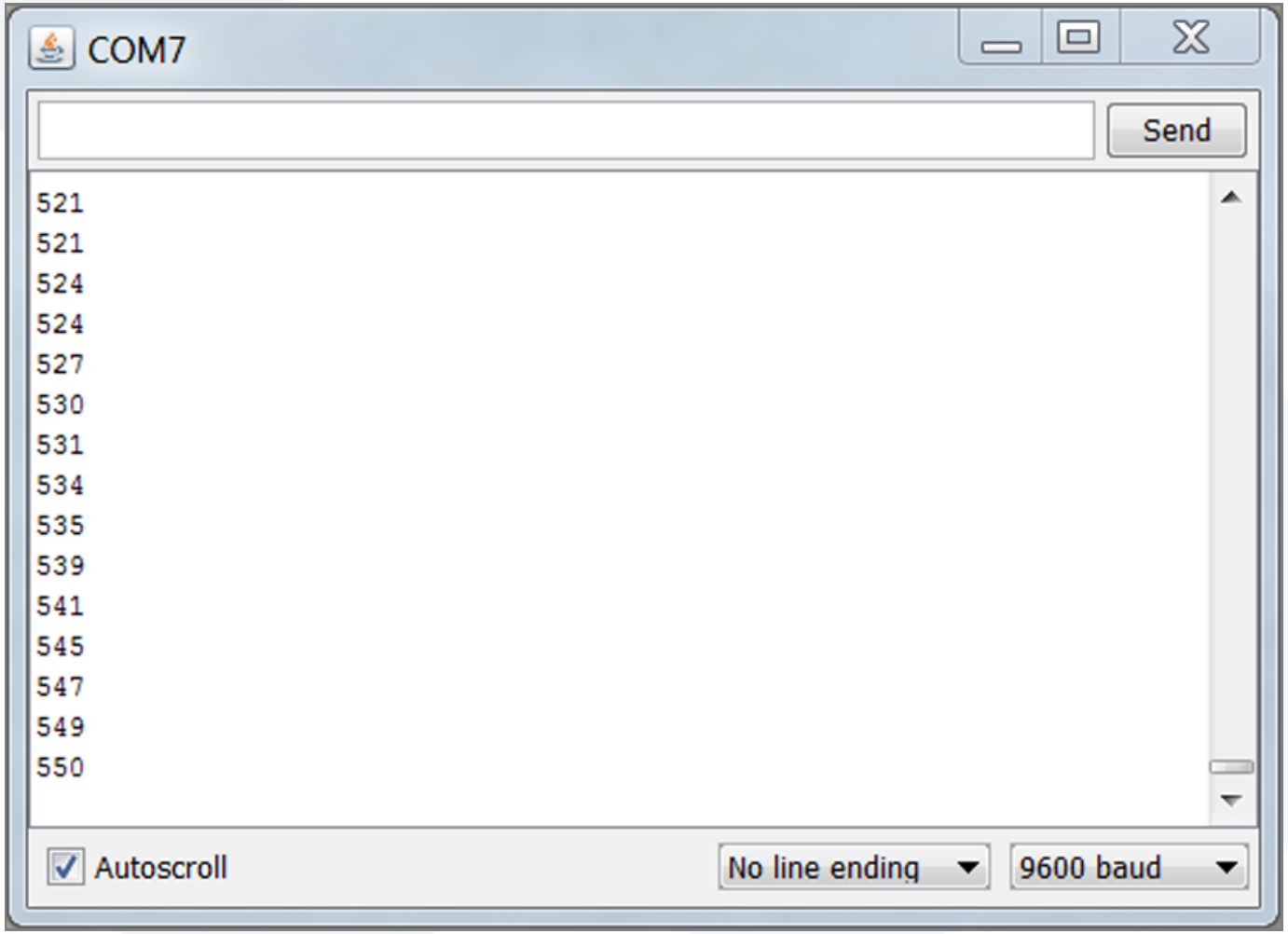

SCL
SDA
AREF
GND
DIGITAL
L
TX
RX
Arduino
UNO
ON
ICSP
www.arduino.cc
IOREF
RESET
3V3
POWER
5V
Gnd
Vin
ANALOG IN
0 1 2 3 4 5


ICSP
S1
ANALOG IN
POWER
RESET

- أثناء قيامنا بتحريك مقبض المقاومة المتغيرة، نقوم بتغيير الجهد الذي نقوم بإدخاله في المدخل التشابهي A0 بين 0 فولت و 5 فولت .
- سنستخدم وظيفة **الاتصال التسلسلي** في Arduino لطباعة قيمة خرج المحول ADC للمقاومة المتغيرة على جهاز الكمبيوتر الخاص بنا أثناء تغيير قيمتها.
- سنستخدم التابع **analogRead ()** لقراءة قيمة رجل الدخل التشابهية المتصلة ببورد Arduino
- سنستخدم التابع **Serial.println ()** لطباعة القيمة المقروءة على شاشة Arduino IDE حيث نقوم بفتح نافذة الإظهار عبر النقر على زر serial monitor.
- يجب أن نقوم بتهيئة الواجهة التسلسلية للكمبيوتر وذلك باستخدام تعليمة Serial.begin () ضمن التابع setup().
- يأخذ التابع Serial.begin () معامل واحد يحدد سرعة الاتصال أو معدل نقل البيانات .
- يحدد معدل نقل البيانات عدد البتات التي يتم نقلها في الثانية.
- تمكننا معدلات النقل الأسرع من إرسال المزيد من البيانات في وقت أقل ، ولكنها قد تؤدي أيضًا إلى حدوث أخطاء في الإرسال في بعض أنظمة الاتصال.

```
//Potentiometer Reading Program
const int POT=0;   //Pot on analog pin 0
int val = 0; //variable to hold the analog reading from the POT
void setup()
{
Serial.begin(9600);
}
void loop()
{
val = analogRead(POT);
Serial.println(val);
delay(500);
}
```


pot | Arduino 1.0.3
File Edit Sketch Tools Help
pot §
//Potentiometer Reading Program
const int POT=0; //Pot on Analog Pin 0
int val = 0; //variable to hold the analog reading from the POT
void setup()
{
Serial.begin(9600);
}
void loop()
{
val = analogRead(POT);
Serial.println(val);
delay(500);
}
COM7
Send
521
521
524
524
527
530
531
534
535
539
541
545
547
549
550
Autoscroll
No line ending
9600 baud


COM7
Send
521
521
524
524
527
530
531
534
535
539
541
545
547
549
550
Autoscroll
No line ending
9600 baud

استخدام الحساسات التشابهية

- جميع أنواع الحساسات تنتج قيم خرج تشابهية (Analog) تتوافق مع أحداث "العالم الحقيقي". تشمل الأمثلة على ذلك ما يلي:

- ❑ **حساسات التسارع (Accelerometers)** التي تكشف عن الميل أو الانحراف (توجد في العديد من الهواتف الذكية والأجهزة اللوحية اليوم).
- ❑ **الحساسات المغناطيسية (Magnetometers)** التي تكشف عن الحقول المغناطيسية (تُستخدم في صنع البوصلات الرقمية).
- ❑ **الحساسات بالأشعة تحت الحمراء (Infrared Sensors)** التي تكشف عن المسافة إلى عائق معين.
- ❑ **حساسات الحرارة (Temperature Sensors)**

- تم تصميم العديد من هذه الحساسات للعمل بطريقة مشابهة للمقاومة المتغيرة (Potentiometer)، حيث نقوم بتزويدها بتغذية كهربائية (VCC) ووصلة تأريض (GND)، وتقوم بتوليد جهد تشابهي (Analog Voltage) عبر الرجل الثالثة للحساس، والتي يتم توصيلها إلى محول الإشارة التشابهية إلى رقمية (ADC) في الأردوينو.

استخدام الحساسات التشابهية

- يربط حساس درجة الحرارة TMP36بسهولة قراءات درجة الحرارة بالدرجة المئوية (سلسيوس) مع مستويات جهد الخرج.
- بما أن كل 10 ملي فولت تقابل 1 درجة مئوية C°، يمكننا بسهولة إنشاء علاقة خطية للتحويل من الجهد الذي نقيسه على خرج الحساس إلى درجة الحرارة المطلقة للبيئة المحيطة:

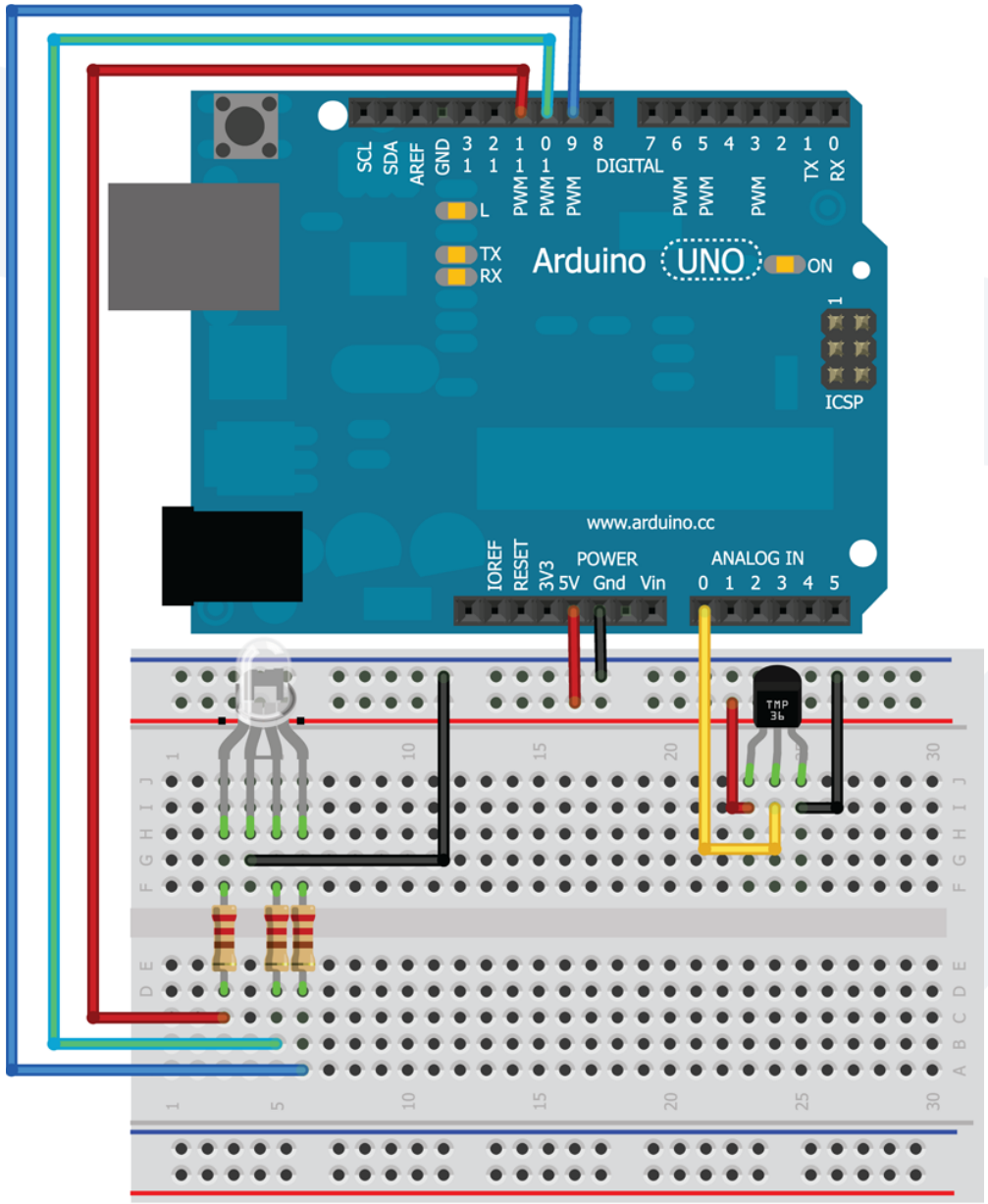
$$C = [(Vout\ in\ mV) - 500]/10$$

- قيمة الإزاحةالبالغة "-500" هي للتعامل مع درجات الحرارة الأقل من 0 درجة مئوية.

العمل مع الحساسات التشابهية لاستشعار درجة الحرارة

- هذا المثال البسيط يستخدم حساس درجة الحرارة .TMP36.
- سنقوم بإنشاء نظام تنبيه بسيط لدرجة الحرارة. سيضيء المصباح (أو الضوء) باللون الأخضر عندما تكون درجة الحرارة ضمن النطاق المقبول، وسيتحول إلى اللون الأحمر عندما تصبح ساخنة جداً، وإلى اللون الأزرق عندما تصبح باردة جداً.
- الخطوات التالية هي نفسها بشكل أساسي لأي حساس تشابهي قد نرغب في استخدامه.

العمل مع الحساسات التشابهية
لاستشعار درجة الحرارة



العمل مع الحساسات التشابهية
لاستشعار درجة الحرارة

- وفقًا لورقة البيانات (Datasheet)، يمكن استخدام المعادلة التالية للتحويل بين درجة الحرارة C ° والجهد الكهربائي(mV).

Temperature(C) * 10 = voltage (mV) – 500

- عند إدخال قيمة700 mV في المعادلة، يمكننا التأكد من أنها تقابل درجة حرارة 20C°.
- باستخدام نفس المنطق: درجة حرارة 22C ° تقابل قيمة رقمية 147ودرجة حرارة 18C ° تقابل قيمة رقمية 139.ستُستخدم هذه القيم الرقمية كعتبات لتغيير لون إضاءة الـ LED، بحيث يشير إلى أن درجة الحرارة أصبحت مرتفعة جدًا أو منخفضة جدًا.

```
//Temperature Alert!
const int BLED=9; //Blue LED on pin 9
const int GLED=10; //Green LED on pin 10
const int RLED=11; //Red LED on pin 11
const int TEMP=0; //Temp Sensor is on pin A0
const int LOWER_BOUND=139; //Lower Threshold
const int UPPER_BOUND=147; //Upper Threshold
int val = 0; //Variable to hold analog reading
void setup()
{
pinMode (BLED, OUTPUT); //Set Blue LED as Output
pinMode (GLED, OUTPUT); //Set Green LED as Output
pinMode (RLED, OUTPUT); //Set Red LED as Output
}
```

```
void loop() {
val = analogRead(TEMP);
if (val < LOWER_BOUND)
{
digitalWrite(RLED, LOW);
digitalWrite(GLED, LOW);
digitalWrite(BLED, HIGH);
}
else if (val > UPPER_BOUND)
{
digitalWrite(RLED, HIGH);
digitalWrite(GLED, LOW);
digitalWrite(BLED, LOW);
}
else{
digitalWrite(RLED, LOW);
digitalWrite(GLED, HIGH);
digitalWrite(BLED, LOW);
}}
```

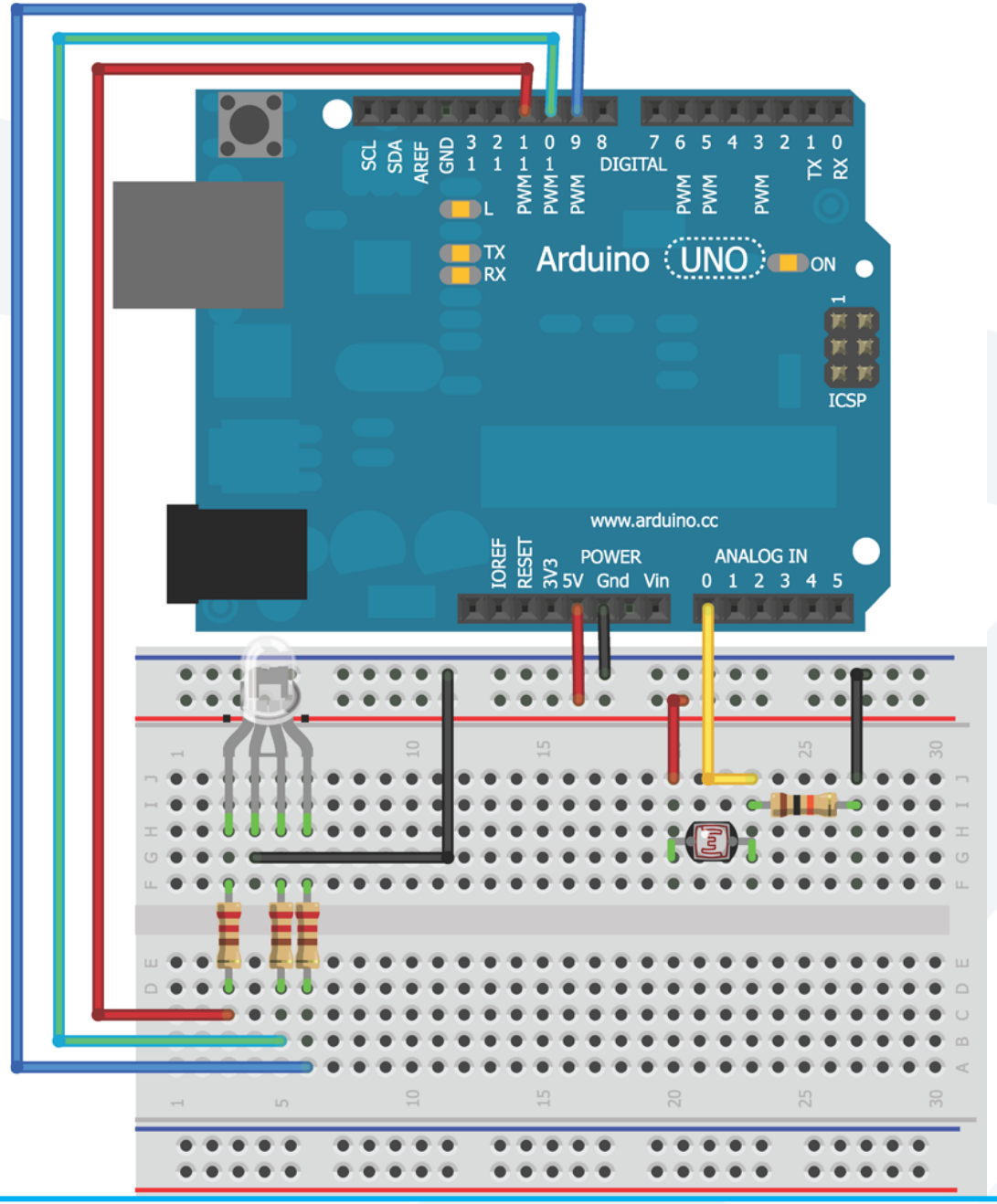
مثال2 : المقاومات الضوئية

- تغير المقاومات الضوئية مقاومتها اعتمادًا على كمية الضوء التي تسقط عليها.
- مثلاً المقاومة الضوئية 200 kΩ عندما تكون في الظلام الدامس، تكون مقاومتها حوالي 200kΩ أما عند التشبع بالضوء، تنخفض المقاومة تقريبًا إلى الصفر.
- يمكننا استخدام المقاومة الضوئية لبناء نظام إنارة ذكي (يجب أن يصبح الضوء أكثر سطوعًا حين تصبح الغرفة مظلمة والعكس صحيح).

بناء نظام إنارة ذكي باستخدام المقاومة الضوئية

- في البداية نحدد القيم المقابلة لمستوى إنارة الغرفة في السطوع الكامل و الظلام التام.
- يمكن أن نستخدم برنامج serial monitor القراءة التسلسلية الذي مر معنا سابقاً و الذي يمكننا من خلاله تحديد قيم الإنارة عندما تكون غرفتنا في السطوع الكامل أو الظلام التام.
- على سبيل المثال، وجدنا أن الغرفة المظلمة تقابل قراءة قيمة 200 رقمي وأن الغرفة المضيئة تمامًا تقابل قراءة قيمة 900 رقمي.
- ستختلف هذه القيم بناءً على ظروف الإضاءة الخاصة بكل مكان يراد تصميم نظام الإنارة له وقيمة المقاومة الضوئية التي نستخدمها.

توصيل الدارة



استخدام الدخل التشابهي للتحكم في الخرج التشابهي

- يمكننا استخدام الأمر analogWrite () للتحكم بشدة سطوع LED الذي يمثل نظام الإنارة لدينا.
- يقبل الأمر analogWrite () قيم بين 0 و 255 فقط ، بينما تقوم تعليمة القراءة التشابهية analogRead() بإرجاع قيم تصل إلى 1023 كقيمة عظمى .
- تحتوي لغة برمجة Arduino على تابعين مفيدين في الربط بين مجالين من القيم: تابع map () و تابع constrain() .
- يستخدم التابع map () على النحو التالي: output = map(value, fromLow, fromHigh, toLow, toHigh)
- value : هي المعلومات التي نبدأ بها و الواقعة في المجال الأول (بين 0 و 1023). في حالتنا، هذه هي أحدث قراءة من الإدخال التشابهي و التي تمثل مستوى إضاءة الغرفة.
- fromLow و fromHigh : هي حدود الإدخال (حدود المجال الأول). و هي القيم التي وجدنا أنها تتوافق مع الحد الأدنى والحد الأقصى للسطوع في غرفتنا. على سبيل المثال ، كانت 200 و 900 على الترتيب.
- toLow و to High : هي حدود المجال الثاني و التي تمثل القيم التي نريد ربط القيمة من المجال الأول بها و التي تمثل المستوى الذي يجب أن يضيء به نظام الإنارة. و نظرًا لأن تعليمة analogWrite () تتوقع أن تأخذ قيمة بين 0 و 255 ، فإننا نستخدمها كحدود لهذا المجال.
- في مشروع نظام الإنارة الذكي كلما ازداد ظلام الغرفة يجب أن يعمل النظام على زيادة شدة إضاءة الـ LED بحيث يصبح أكثر سطوعاً . لذلك، عندما يكون الإدخال من ADC ذو قيمة منخفضة، فإننا نريد أن يكون الخرج على الـ LED عالي القيمة، والعكس صحيح.

```
const int RLED=9;      //Red LED on pin 9 (PWM)
const int LIGHT=0;     //Light Sensor on analog pin 0
const int MIN_LIGHT=200;    //Minimum expected light value
const int MAX_LIGHT=900;    //Maximum Expected Light value
int val = 0;          //variable to hold the analog reading
void setup()
{
pinMode(RLED, OUTPUT);      //Set LED pin as output
}
void loop()
{
val = analogRead(LIGHT);              //Read the light sensor
val = map(val, MIN_LIGHT, MAX_LIGHT, 255, 0);   //Map the light reading
val = constrain(val, 0, 255);         //Constrain light value
analogWrite(RLED, val);               //Control the LED
}
```