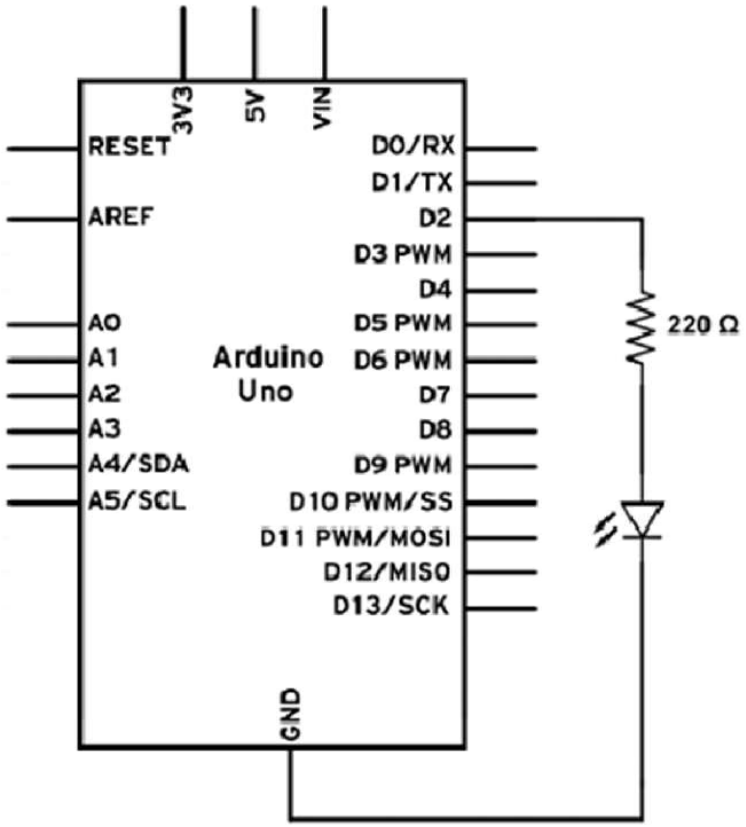
Blinking LED without delay()

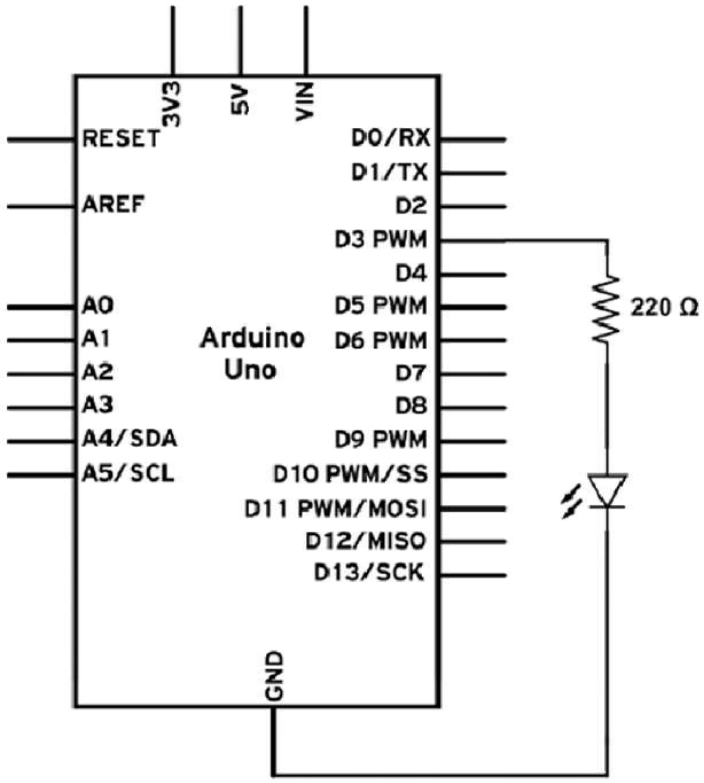
```
// Variable for keeping the previous LED state
int previousLEDstate = LOW;

unsigned long lastTime = 0; // Last time the LED changed state
int interval = 200; // interval between the blinks in milliseconds

void setup() {
  // Declare the pin for the LED as Output
  pinMode(LED_BUILTIN, OUTPUT);
}

void loop(){

  // Here we can write any code we want to execute continuously
  // Read the current time
  unsigned long currentTime = millis();

  // Compare the current time with the last time
  if (currentTime - lastTime >= interval){

    // First we set the previous time to the current time
    lastTime = currentTime;

    // Then we inverse the state of the LED
    if (previousLEDstate == HIGH) {
      digitalWrite(LED_BUILTIN, LOW);
      previousLEDstate = LOW;
    } else {
      digitalWrite(LED_BUILTIN, HIGH);
      previousLEDstate = HIGH;
    }
  }
}
```

Connecting an external LED



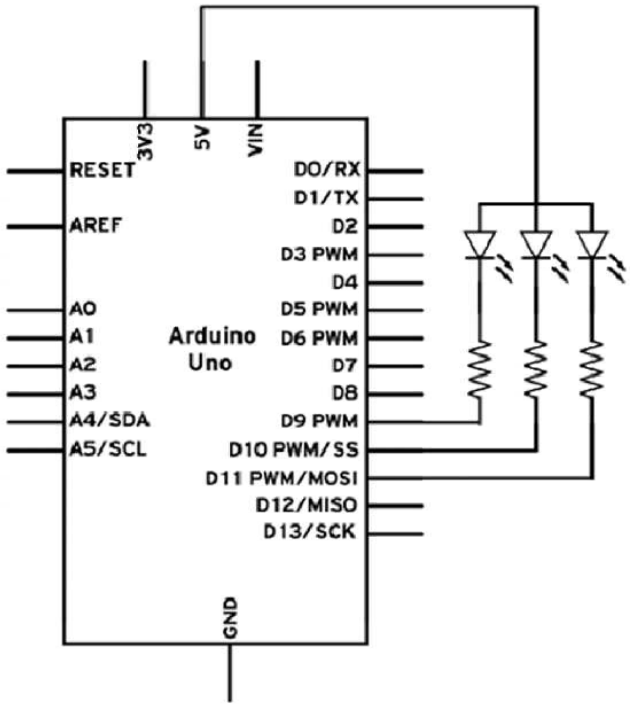

```
// Declare the LED pin
int LED = 2;

void setup() {
  // Declare the pin for the LED as Output
  pinMode(LED, OUTPUT);
}

void loop(){
  // Here we will turn the LED ON and wait 200 milliseconds
  digitalWrite(LED, HIGH);
  delay(200);
  // Here we will turn the LED OFF and wait 200 milliseconds
  digitalWrite(LED, LOW);
  delay(200);
}
```

Fading the external LED




```
// Declare the LED pin with PWM
int LED = 3;

void setup() {
  // Declare the pin for the LED as Output
  pinMode(LED, OUTPUT);
}

void loop(){
  // Here we will fade the LED from 0 to maximum, 255
  for (int i = 0; i < 256; i++){
    analogWrite(LED, i);
    delay(5);
  }
  // Fade the LED from maximum to 0
  for (int i = 255; i >= 0; i--){
    analogWrite(LED, i);
    delay(5);
  }
}
```

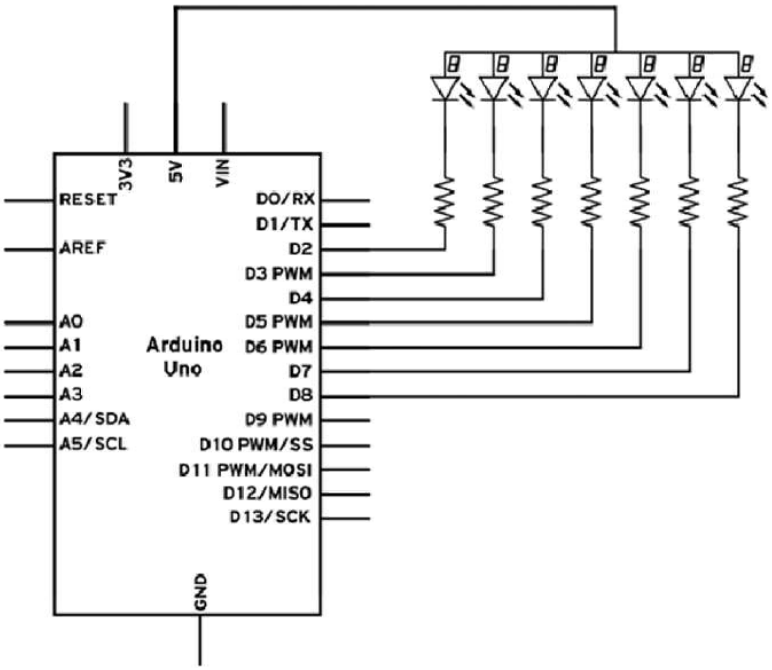
RGB LED




```
// Declare the PWM LED pins
int redLED = 9;
int greenLED = 10;
int blueLED = 11;

void setup() {
  // Declare the pins for the LED as Output
  pinMode(redLED, OUTPUT);
  pinMode(greenLED, OUTPUT);
  pinMode(blueLED, OUTPUT);
}

// A simple function to set the level for each color from 0 to 255
void setColor(int redValue, int greenValue, int blueValue){
  analogWrite(redLED, 255 - redValue);
  analogWrite(greenLED, 255 - greenValue);
  analogWrite(blueLED, 255 - blueValue);
}
```

```
void loop(){
  // Change a few colors

  setColor(255, 0, 0); // Red Color
  delay(500);

  setColor(0, 255, 0); // Green Color
  delay(500);

  setColor(0, 0, 255); // Blue Color
  delay(500);

  setColor(255, 255, 0); // Yellow
  delay(500);

  setColor(0, 255, 255); // Cyan
  delay(500);

  setColor(255, 0, 255); // Magenta

    delay(500);

    setColor(255, 255, 255); // White
    delay(500);
 }
```

The 7-segment display

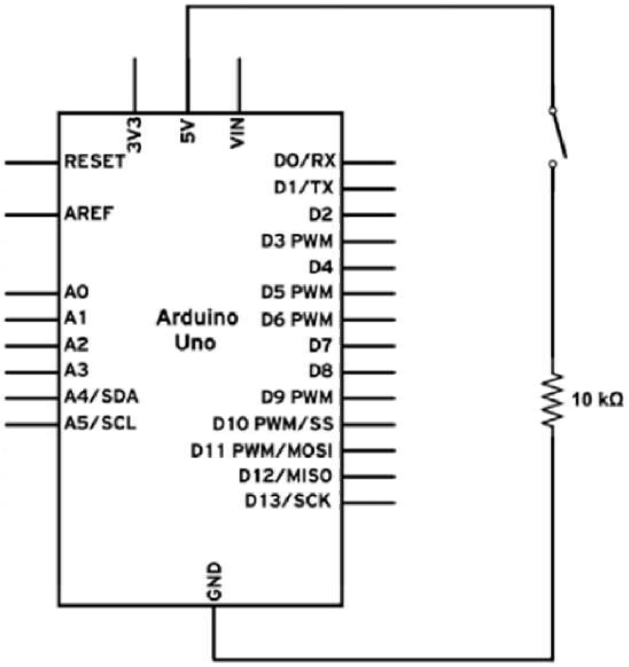



```
// Declare the pins for the Segment display
int pinUP = 2;  // Upper segment
int pinUPR = 3; // Up-right segment
int pinDWR = 4; // Down-right segment
int pinDW = 5;  // Down segment
int pinDWL = 6; // Down-left segment
int pinUPL = 7; // Up-left segment
int pinCT = 8;  // Center segment

void setup() {
  // Declare the pins as Outputs
  pinMode(pinUP, OUTPUT);
  pinMode(pinUPR, OUTPUT);
  pinMode(pinDWR, OUTPUT);
  pinMode(pinDW, OUTPUT);
  pinMode(pinDWL, OUTPUT);
  pinMode(pinUPL, OUTPUT);
  pinMode(pinCT, OUTPUT);
}
```

```
void writeNumber(int value){
  // First we erase the previous value
  digitalWrite(pinUP, HIGH);
  digitalWrite(pinUPR, HIGH);
  digitalWrite(pinDWR, HIGH);
  digitalWrite(pinDW, HIGH);
  digitalWrite(pinDWL, HIGH);
  digitalWrite(pinUPL, HIGH);
  digitalWrite(pinCT, HIGH);
  // If we want to write 0
  if (value == 0){
    digitalWrite(pinUP, LOW);
    digitalWrite(pinUPR, LOW);
    digitalWrite(pinDWR, LOW);
    digitalWrite(pinDW, LOW);
    digitalWrite(pinDWL, LOW);
    digitalWrite(pinUPL, LOW);
  }

  // If we want to write 1

     if (value == 1){
       digitalWrite(pinUPR, LOW);
       digitalWrite(pinDWR, LOW);
     }

     // If we want to write 2
     if (value == 2){
       digitalWrite(pinUP, LOW);
       digitalWrite(pinUPR, LOW);
       digitalWrite(pinCT, LOW);
       digitalWrite(pinDWL, LOW);
       digitalWrite(pinDW, LOW);
     }
```

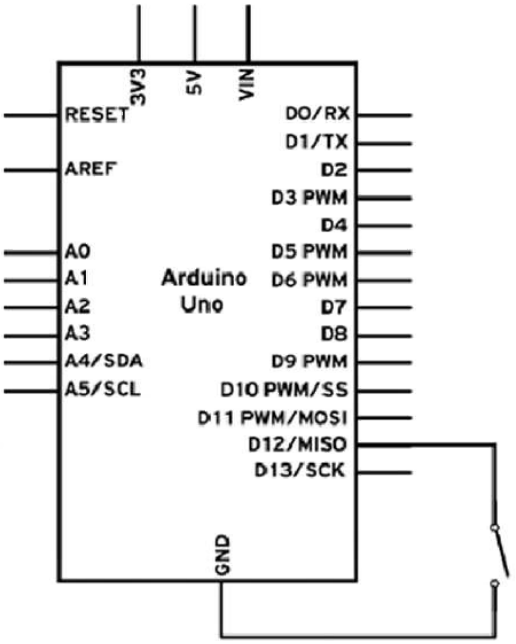
```
  // If we want to write 3
  if (value == 3){
    digitalWrite(pinUP, LOW);
    digitalWrite(pinUPR, LOW);
    digitalWrite(pinCT, LOW);
    digitalWrite(pinDWR, LOW);
    digitalWrite(pinDW, LOW);
  }
}


void loop(){
  // A resetting count-down
  writeNumber(3);
  delay(1000);

  writeNumber(2);
  delay(1000);

  writeNumber(1);
  delay(1000);

  writeNumber(0);
  delay(1000);
}
```

Connecting a button



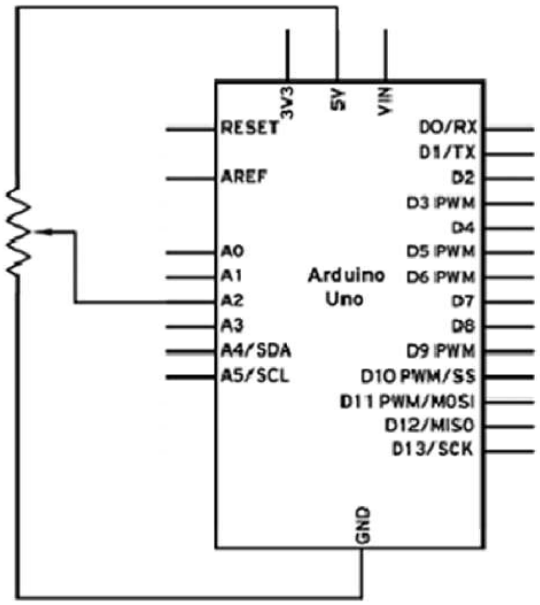

```
// Declare the pins for the Button and the LED
int buttonPin = 2;
int LED = 13;

void setup() {
  // Define pin #2 as input
  pinMode(buttonPin, INPUT);
  // Define pin #13 as output, for the LED
  pinMode(LED, OUTPUT);
}

void loop() {
  // Read the value of the input. It can either be 1 or 0.
  int buttonValue = digitalRead(buttonPin);

  if (buttonValue == HIGH) {
    // If button pushed, turn LED on
    digitalWrite(LED,HIGH);
  } else {
    // Otherwise, turn the LED off
    digitalWrite(LED, LOW);
  }
}
```

Button with no resistor



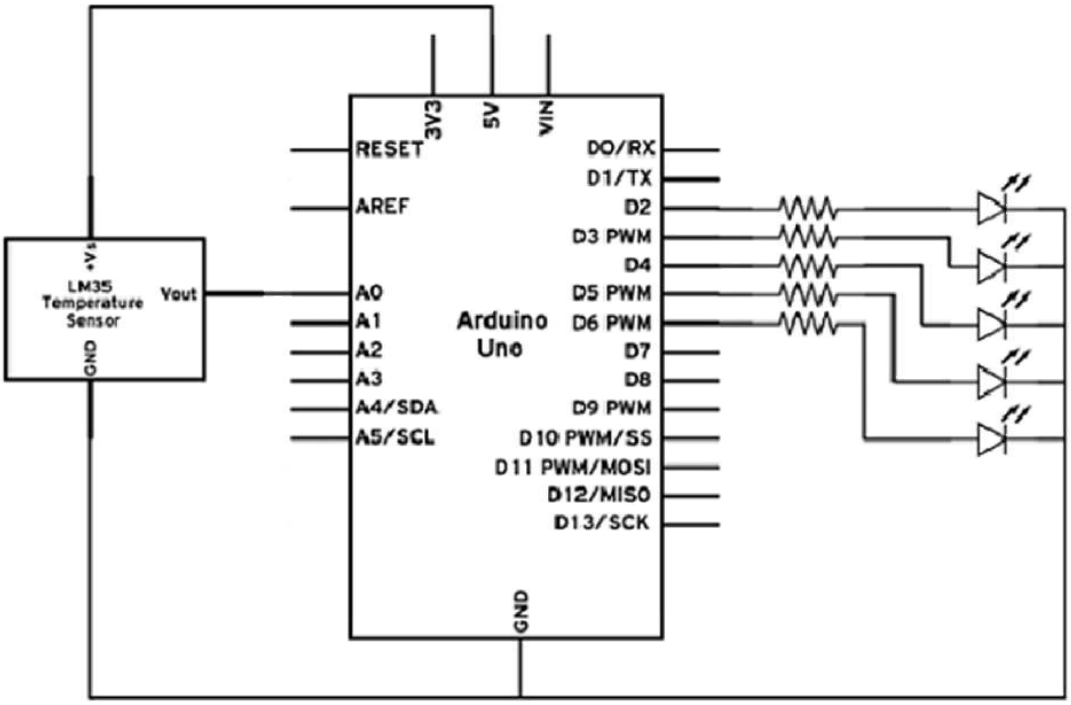

```
// Declare the pins for the Button and the LED
int buttonPin = 12;
int LED = 13;

void setup() {
  // Define pin #12 as input and activate the internal pull-u
resistor
  pinMode(buttonPin, INPUT_PULLUP);
  // Define pin #13 as output, for the LED
  pinMode(LED, OUTPUT);
}

void loop(){
  // Read the value of the input. It can either be 1 or 0
  int buttonValue = digitalRead(buttonPin);

  if (buttonValue == LOW){
    // If button pushed, turn LED on
    digitalWrite(LED,HIGH);
  } else {
    // Otherwise, turn the LED off
    digitalWrite(LED, LOW);
  }
}
```

Simple sensor – potentiometer




```
int LED = 13;        // Declare the built-in LED
int sensorPin = A2; // Declare the analog port we connected

void setup(){
  // Start the Serial connection
  Serial.begin(9600);
  // Set the built in LED pin as OUTPUT
  pinMode(LED, OUTPUT);
}


void loop(){
  // Read the value of the sensor
  int val = analogRead(sensorPin);
  // Print it to the Serial
  Serial.println(val);

  // Blink the LED with a delay of a forth of the sensor value
  digitalWrite(LED, HIGH);
  delay(val/4);
  digitalWrite(LED, LOW);
  delay(val/4);
}
```

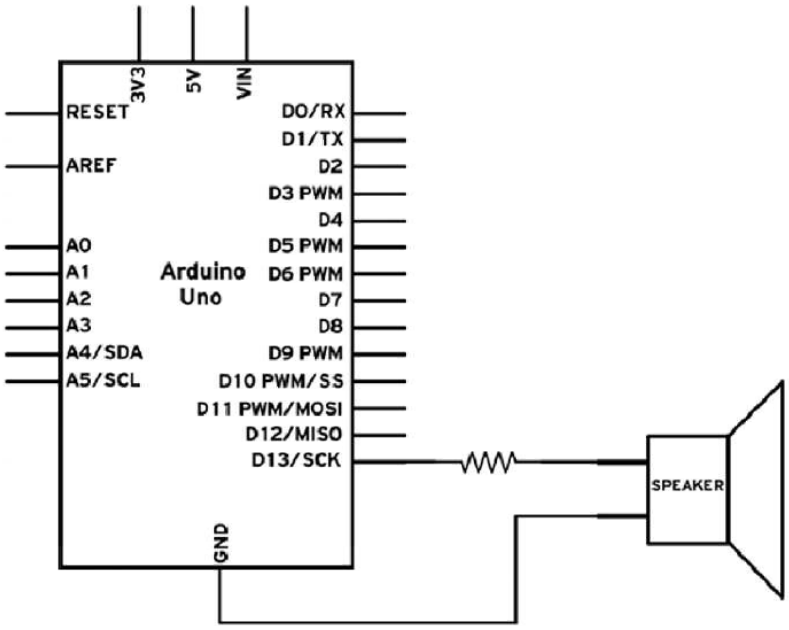
Temperature sensor




```
// Declare the LEDs in an array
int LED [5] = {2, 3, 4, 5, 6};
int sensorPin = A0; // Declare the used sensor pin

void setup(){
   // Start the Serial connection
  Serial.begin(9600);
  // Set all LEDs as OUTPUTS
  for (int i = 0; i < 5; i++){
    pinMode(LED[i], OUTPUT);
  }
}

void loop(){
  // Read the value of the sensor
  int val = analogRead(sensorPin);
  Serial.println(val); // Print it to the Serial

  // On the LM35 each degree Celsius equals 10 mV
  // 20C is represented by 200 mV which means 0.2 V / 5 V * 1023 = 41
  // Each degree is represented by an analogue value change of
  approximately 2

  // Set all LEDs off
  for (int i = 0; i < 5; i++){
    digitalWrite(LED[i], LOW);
  }
```

```
if (val > 40 && val < 45){ // 20 - 22 C
   digitalWrite( LED[0], HIGH);
 } else if (val > 45 && val < 49){ // 22 - 24 C
   digitalWrite( LED[0], HIGH);
   digitalWrite( LED[1], HIGH);
 } else if (val > 49 && val < 53){ // 24 - 26 C
   digitalWrite( LED[0], HIGH);
   digitalWrite( LED[1], HIGH);
   digitalWrite( LED[2], HIGH);
 } else if (val > 53 && val < 57){ // 26 - 28 C
   digitalWrite( LED[0], HIGH);


     digitalWrite( LED[1], HIGH);
     digitalWrite( LED[2], HIGH);
     digitalWrite( LED[3], HIGH);
   } else if (val > 57){ // Over 28 C
     digitalWrite( LED[0], HIGH);
     digitalWrite( LED[1], HIGH);
     digitalWrite( LED[2], HIGH);
     digitalWrite( LED[3], HIGH);
     digitalWrite( LED[4], HIGH);
   }
   delay(100); // Small delay for the Serial to send
 }
```

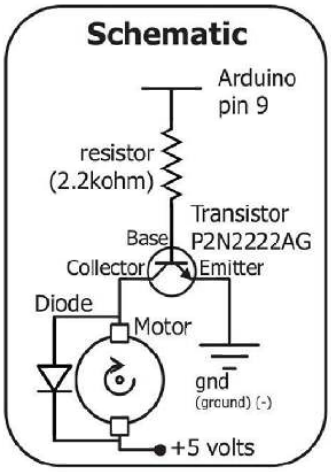
Creating sound




```
// Defining the 8 frequencies that make the 7 notes and one
repetition in the Solfeggio
#define Do  131
#define Re  147
#define Mi  165
#define Fa  175
#define Sol 196
#define La  220
#define Ti  247
#define Do2 262

// Defining the pin connected to the speaker
int tonePin = 13;

void setup(){
```

```
   // Tone pins don't need to be declared
}

void loop(){
  // Do
  tone(tonePin, Do, 125);
  delay(125);
  // Re
  tone(tonePin, Re, 125);
  delay(125);
  // Mi
  tone(tonePin, Mi, 125);
  delay(125);
  // Fa
  tone(tonePin, Fa, 125);
  delay(125);
  // Sol
  tone(tonePin, Sol, 125);
  delay(125);
  // La
  tone(tonePin, La, 125);
  delay(125);
  // Ti
  tone(tonePin, Ti, 125);
  delay(125);
  // Higher Do
  tone(tonePin, Do2, 125);
  delay(125);
}
```

استخدام محرك التيار المستمر DC

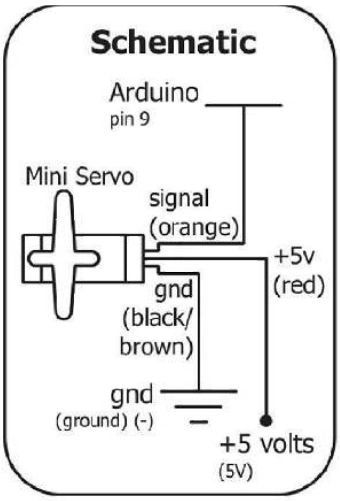


مكونات المثال (7):

- ✓ بورده اردوينو Arduino Uno
- ✓ لوحه تجارب Breadboard
- ✓ محرك تيار مُستَمرصغير DC motor
- ✓ ترانزستور 2N2222 او PN2222
- ✓ دايود 1N4001 او اى بديل
- ✓ مقاومه 2.2 كيلو اوم
- ✓ أسلاك توصيل
- ✓ كابل التوصيل بالUSB

```
//Example_10_DC_Motor
int motorPin = 9 ;
int onTime = 2500 ;
int offTime = 1000 ;

void setup ( )
{pinMode(motorPin, OUTPUT); }

void loop ( )
{
analogWrite(motorPin,100);
delay(onTime);
digitalWrite(motorPin, LOW);
delay(offTime);

analogWrite(motorPin,190);
delay(onTime);
digitalWrite(motorPin, LOW);
delay(offTime);

analogWrite(motorPin,255);
delay(onTime);
digitalWrite(motorPin, LOW);
delay(offTime);
}
```

استخدام محرك سيرفو



مكونات المثال (7):

- ✓ بورده اردوينو Arduino Uno
- ✓ لوحه تجارب Breadboard
- ✓ محرك تيار مُستَمرصغير DC motor
- ✓ ترانزستور 2N2222 او PN2222
- ✓ دايود 1N4001 او اى بديل
- ✓ مقاومه 2.2 كيلو اوم
- ✓ أسلاك توصيل
- ✓ كابل التوصيل بالUSB

مكتبه اضافيه

```
//Example_11_Servo_Motor
#include <Servo.h>
Servo myservo;
int pos = 0;

void setup()
{
  myservo.attach(9);
}

  void loop()
{

  for(pos = 0; pos < 180; pos += 1)
  {
    myservo.write(pos);
    delay(15);
  }
  for(pos = 180; pos>=1; pos-=1)
  {
    myservo.write(pos);
    delay(15);
  }

}
```

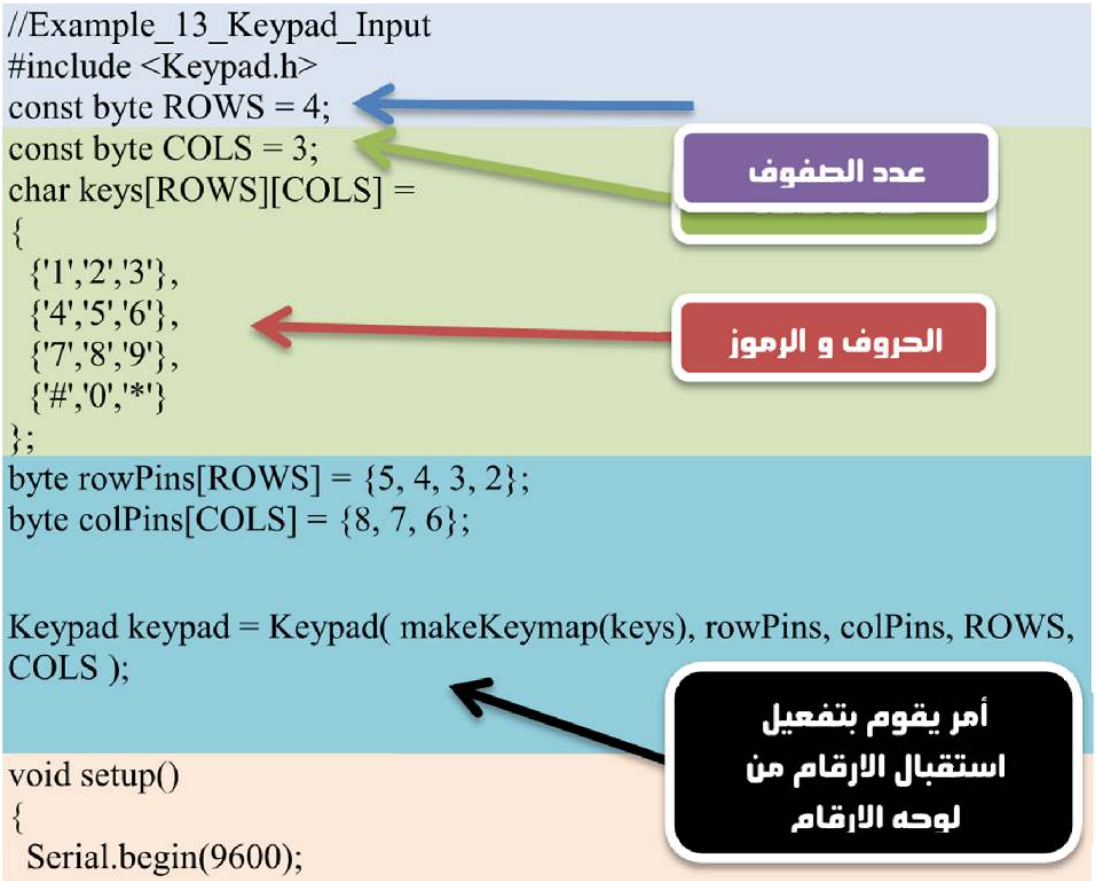
توصيل شاشه Character

LCD بمقاس 16x2

```
//Example_12_LCD_16x2

#include <LiquidCrystal.h>

LiquidCrystal lcd(7, 8, 9, 10, 11, 12);


void setup()

 {

  lcd.begin(16, 2);

  lcd.print("hello, world!");

}


void loop()

 {

  lcd.setCursor(0, 1);

  lcd.print(millis()/1000);

}
```

استخدام لوحه الأرقام Keypad مع اردوينو

```
//Example_13_Keypad_Input
#include <Keypad.h>
const byte ROWS = 4;
const byte COLS = 3;
char keys[ROWS][COLS] =
{
  {'1','2','3'},
  {'4','5','6'},
  {'7','8','9'},
  {'#','0','*'}
};
byte rowPins[ROWS] = {5, 4, 3, 2};
byte colPins[COLS] = {8, 7, 6};


Keypad keypad = Keypad( makeKeymap(keys), rowPins, colPins, ROWS,
COLS );


void setup()
{
  Serial.begin(9600);
}


void loop()
{
  char key = keypad.getKey();


  if (key != NO_KEY) { Serial.println(key); }
}
```



تشغيل وإطفاء LED
بواسطة لوحة المفاتيح

```
#include <Keypad.h>                                  <<------- إضافة مكتبة Keypad إلى البرنامج

const byte ROWS = 4;                                 <<------------------------------ أربعة صفوف
const byte COLS = 4;                                 <<------------------------------ أربعة أعمدة

char hexaKeys[ROWS][COLS] = {                        <<--------- تعريف رموز المفاتيح في اللوحة
  {'1','2','3','A'},
  {'4','5','6','B'},
  {'7','8','9','C'},
  {'*','0','#','D'}
};
byte rowPins[ROWS] = {3, 4, 5, 6};                   << أرقام الأرجل التي يتم توصيلها مع الصفوف
byte colPins[COLS] = {7, 8, 9, 10};                  <<  أرقام الأرجل التي يتم توصيلها مع الأعمدة

Keypad customKeypad = Keypad( makeKeymap(hexaKeys),
 rowPins, colPins, ROWS, COLS);

int LED=13;                                          <<----------------- ترميز مجموعة من الأرجل
char customKey;                                      <<---------- تعريف متغير باسم customKey
void setup()  {

  pinMode(LED,OUTPUT);
  Serial.begin(9600);
}

void loop()
{                                                    قراءة رمز المفتاح الذي تم ضغطه وتخزين
  customKey = customKeypad.getKey();                 الرمز في متغير <<---------------------------
                                                     الجملة الشرطة المرتبطة بعرض الرقم في
  if (customKey != NO_KEY)                           حالة الضغط على أي مفتاح <<---------------
  {
    Serial.println(customKey);                       كتابة المتغير على المراقب التسلسلي ----<<

    switch (customKey)
    {
      case '1':                                      الجملة الشرطية في حالة الضغط على الرقم 1 <<
       digitalWrite(LED,HIGH);                       تشغيل LED ------------------------------<<
       break;

      case '2':                                      الجملة الشرطية في حالة الضغط على الرقم 2 <<
       digitalWrite(LED,LOW);                        إطفاء LED -------------------------------<<
       break;

      default: ;
    }
  }
}
```

قياس درجة الحرارة وعرضها على شاشة العرض

```
#include<LiquidCrystal.h>                    <<---------- إضافة مكتبة LCD إلى البرنامج
LiquidCrystal lcd(8, 9, 4, 5, 6, 7);        <<---- تهيئة المكتبة مع المنافذ المستخدمة
float C;                                              (RS,E,D4,D5,D6,D7)
int tempPin = A0;
int LED= 13;

void setup() {
 {
  pinMode(LED,OUTPUT);
  Serial.begin(9600);
  lcd.begin(16,2);          <<-- تهيئة شاشة LCD وتحديد ابعادها (الاعمدة والصفوف)
  lcd.setCursor(0,0);       <<-- إعادة مؤشر الكتابة إلى بداية السطر الأول
  lcd.print("hello");       <<--------- طباعة النص على شاشة العرض
  lcd.setCursor(0,1);       <<-- نقل مؤشر الكتابة إلى بداية السطر الثاني
  lcd.print("I'm programmer");
  delay(2000);
  lcd.clear();
  lcd.setCursor(0,0);
  lcd.print("C=");
 }

void loop()
 {

 C=analogRead(tempPin);

                                          X/= هذه المعادلة لتحويل درجة الحرارة إلى
  C = C*0.48828125;                       درجة سيليزية <<----------

  if (C>=30){                             الجملة الشرطية في حالة إرتفاع درجة الحرارة-->>
    digitalWrite(LED,HIGH);}
    else{
     digitalWrite(LED,LOW);}
     lcd.setCursor(3,0);
     lcd.print(C);

 }
```