Al-Mustaqbal University

Science College

Dep. Medical Physics

Medical Physics

Second Stage

Lab4

Computer 4

M.S.c Mortada Sabri
M.S.c Noor Mohammed

Serial monitor

To make electronic circuits transmit information to each other, you must share a common communication protocol. There are two types of communication:

1- Parallel communication

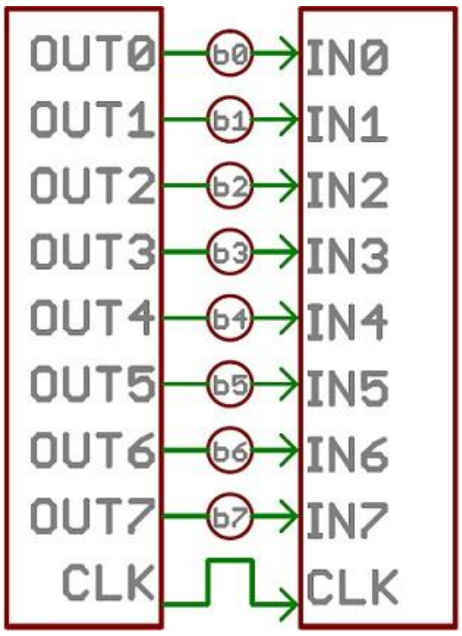
It is transfers several bits at the same time. Uses a number of wires of eight or sixteen or more, and the data are transmitted in huge waves of values 0 and 1, the bytes are transmitted through each pulse of the clock.

2- Serial communication

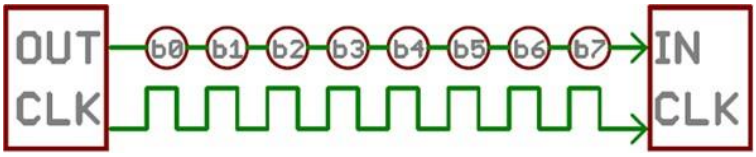
It is transferring data bit by bit. These interfaces can work using a few wires up to one wire and usually do not exceed four wires, one bit transfer each pulse clock. Divided into two types:

a- **Synchronous Serial:** All devices connected to a synchronous serial bus share a common clock, requiring at least one additional wire between connected devices.

b- **Asynchronous Serial:** Data is transferred without the use of an external clock signal. It is the best way to reduce the number of wires required.



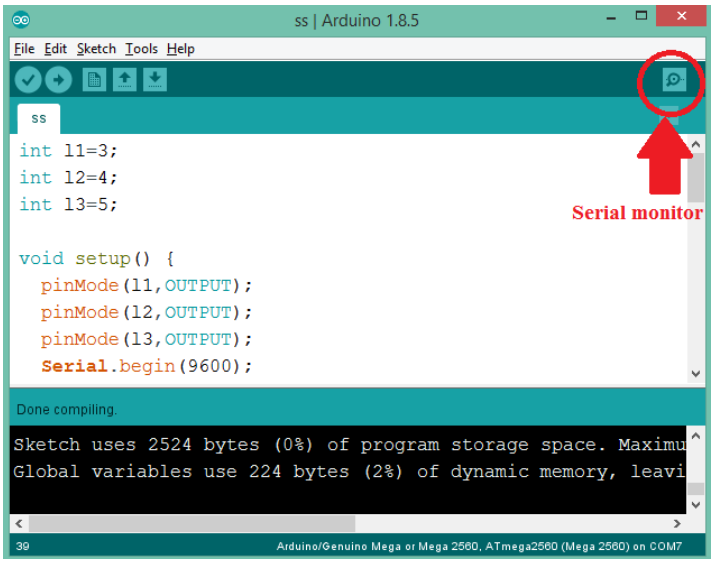
Parallel communication



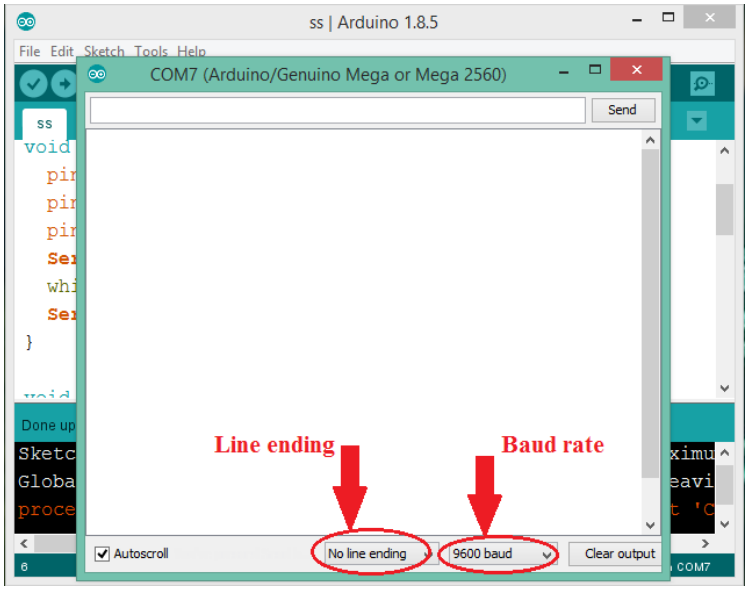
Serial communication

Serial monitor: The Serial Monitor is a separate pop-up window that acts as a separate terminal that communicates by receiving and sending Serial Data. It is the link between the computer and the Arduino.

Serial Data is sent as Asynchronous Serial over a single wire and consists of a series of 1's and 0's sent over the wire. Data can be sent in both directions (In our case on two wires).

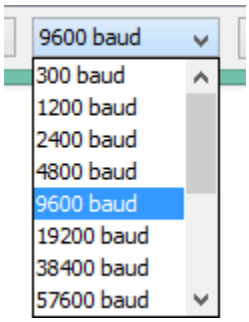


When open the window will find that it contains the following:

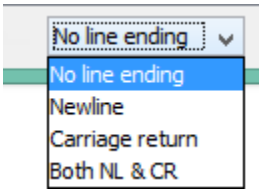


Baud rate: Determines the speed of data transmission over the serial line, expressed in bits / sec (bps).

The Baud rate can be any possible value. The only thing that is required is that both devices work at the same rate. Of the most common baud rates especially in simple devices that do not require high speed is 9600bps. Other standard rates include 300, 1200, 2400, 4800, 19200, 34800, 57600, 115200 and so on...



Line ending: Specifies what sends after the value entered to the Arduino by pressing the send button.



No line ending	Do not send anything
Newline	\n
Carriage return	\r
Both NL & CR	\r\n

Serial monitor Functions

Serial.begin(): Sets the data rate in bits per second (baud) for serial data transmission, For communicating between the computer and Arduino.

```
Serial.begin(speed, config)
```

Ex: Serial.begin(9600);

Serial.available(): Get the number of bytes (characters) available for reading from the serial port.

```
Serial. available():
```

Serial.read(): Reads incoming serial data.

```
Serial. read():
```

Serial.print: Prints data to the serial port as human-readable ASCII text.

```
Serial. Print(val)
```

Val: The value to be printed.
(**EX:** Serial.print("Hello World")→ gives Hello world)

Serial.println(): Prints data to the serial port and a newline character (or '\n'). This command takes the same forms as Serial.print().

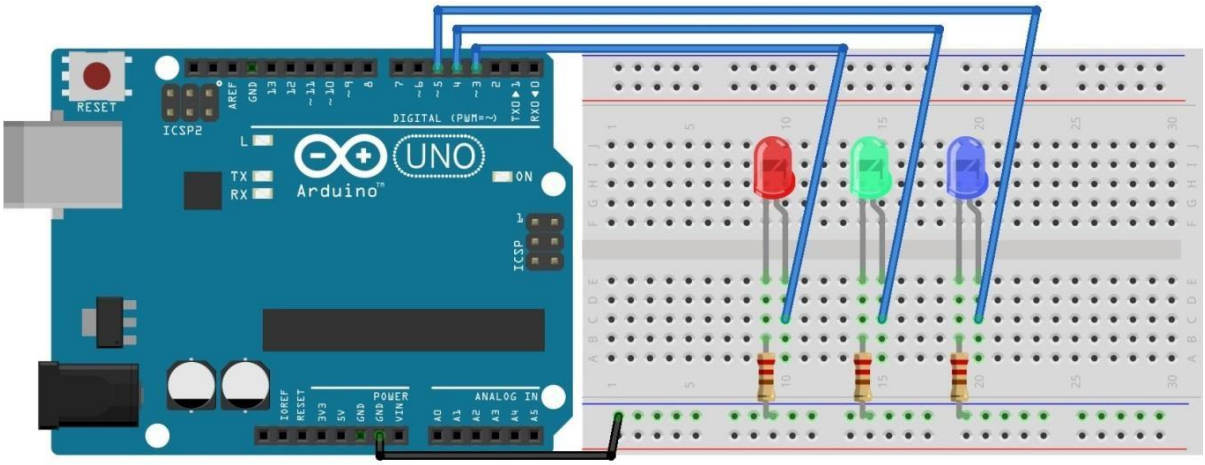
```
Serial.println(val)
```

Example1

(Turn on Leds by serial monitor)

Requirements: Arduino, BreadBoard, 3Resistor, 3 Led, wires.

Connection map:



Code:

```
int led1=3;
int led2=4;
int led3=5;

void setup() {
  pinMode(led1,OUTPUT);
  pinMode(led2,OUTPUT);
  pinMode(led3,OUTPUT);
  Serial.begin(9600);
  Serial.println("Enter LED Number 3,4,5 to opened up & 0 to Close all");
}
```

```
void loop() {
  if  (Serial.available()){
    char c =Serial.read();
    if(c=='3'){
      digitalWrite(led1,HIGH);
      digitalWrite(led2,LOW);
      digitalWrite(led3,LOW);
    }
    else if(c=='4'){
      digitalWrite(led2,HIGH);
      digitalWrite(led1,LOW);
      digitalWrite(led3,LOW);
    }
    else if(c=='5'){
      digitalWrite(led3,HIGH);
      digitalWrite(led1,LOW);
      digitalWrite(led2,LOW);
    }
    else if(c=='0'){
      digitalWrite(led3,LOW);
      digitalWrite(led1,LOW);
      digitalWrite(led2,LOW);
    }
  }
}
```

Buzzer: A device that converts electrical power to audible sound. Typical uses of buzzers include alarm devices, timers and so on.

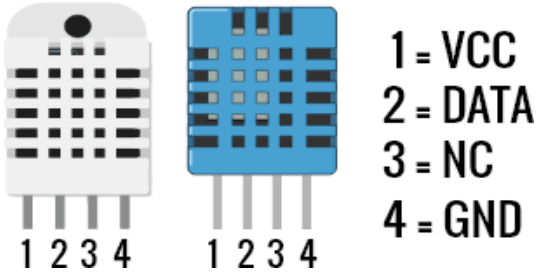
Sensors

What is a sensor?

A sensor is an object (device, module, or subsystem) whose purpose is to detect events or changes in its environment, and then send the information to other electronics to provide a corresponding output.

Most used sensors for Arduino:

1- **DHT11 sensor:** stand for Digital Humidity & Temperature sensor. Used for measure temperature and humidity. There is another type of it known as DHT22.

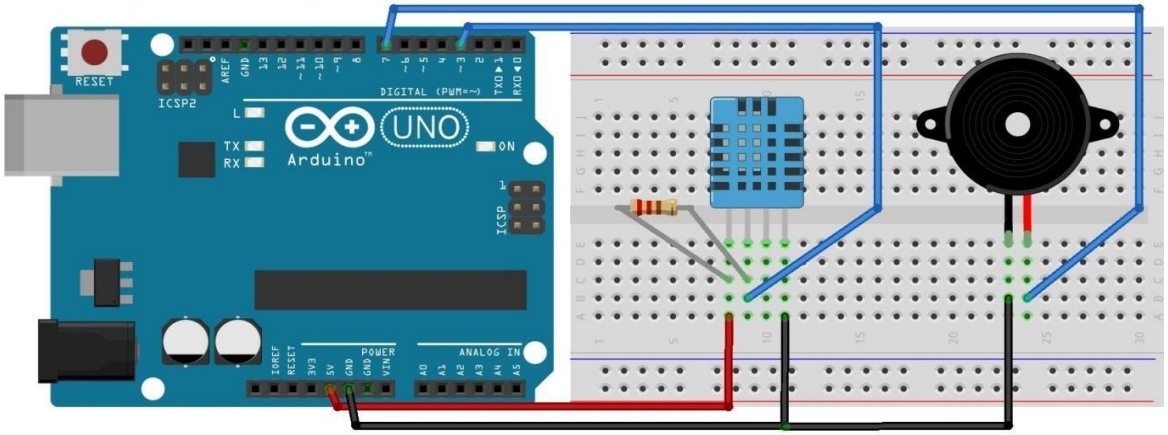


Example

(Measurement of temperature and humidity)

Requirements: Arduino, BreadBoard, Resistor, dht11 Sensor,Buzzer, wires.

Connection map:



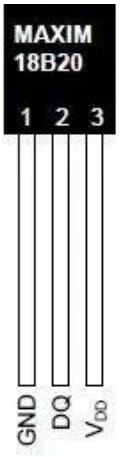
Code:

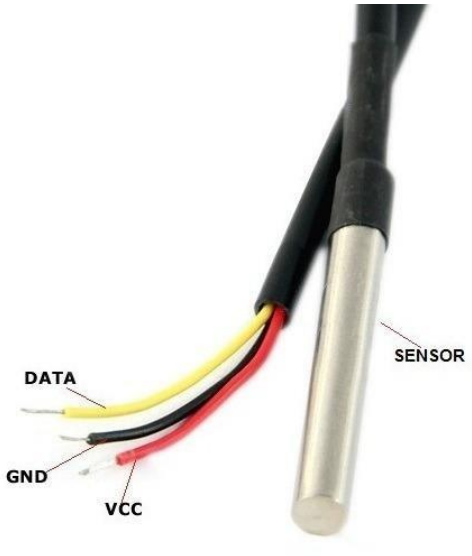
(*first must add dht11 library - can download it from github)

```
#include <dht11.h>

dht11 DHT;
int dht11 = 3;
int buzzer = 7;
void setup() {
    pinMode(buzzer, OUTPUT);
   Serial.begin(9600);
}
void loop() {
  DHT.read(dht11);
  if (DHT.temperature >= 23) {
    Serial.println("Warring!");
    tone(buzzer, 1000);
    delay(1000);
    noTone(buzzer);
  Serial.println("Temperature: ");
  Serial.print("C°= ");
  Serial.println(DHT.temperature);
}
```

2- **Waterproof (DS18b20) sensor:** The DS18b20 is a digital sensor that operates through a single wire and provides accurate temperature readings. It also has water resistance and the reading is not affected by the long distance between the sensor and the control board.



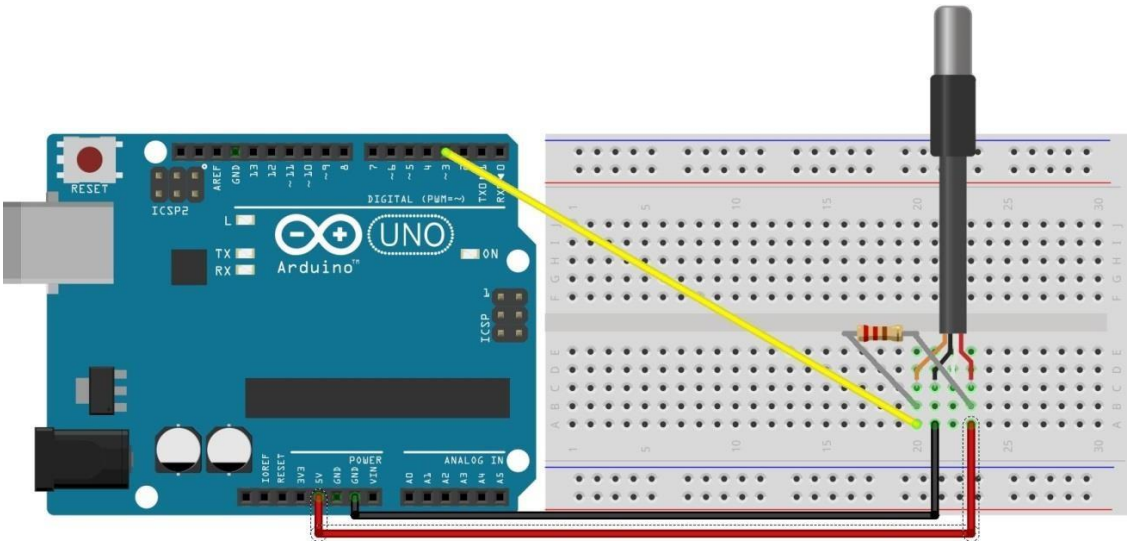


Example

(Measurement of water temperature)

Requirements: Arduino, BreadBoard, Resistor, waterproof Sensor, wires.

Connection map:




THANK YOU