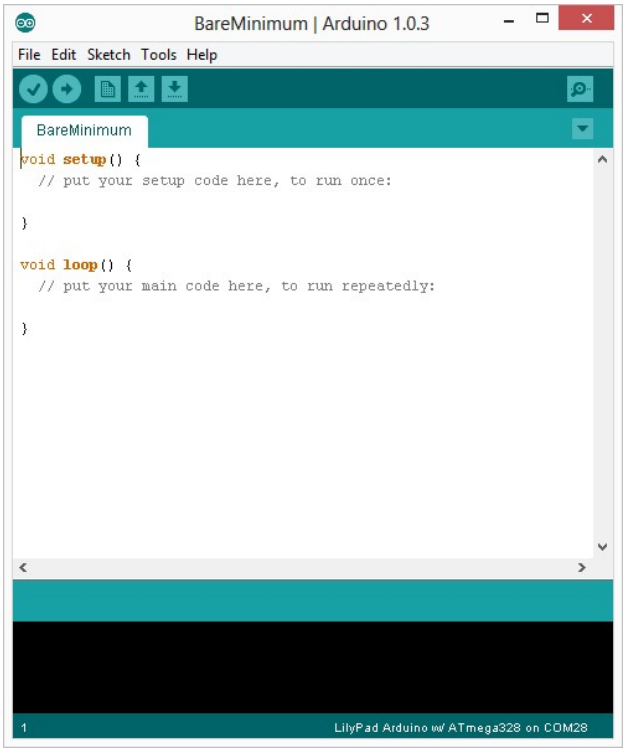
United International University
QUEST FOR EXCELLENCE

CSE 426 Experiment 8: Introduction to Arduino Uno board

Objective:
• To understand the configuration of Arduino board and simulation in proteus.

Equipment needed:
- Arduino Uno board
- Arduino IDE (Compiler)
- Proteus (Simulator)

Part A: Arduino IDE:



Two required functions / methods / routines:

```
void setup()
{
        // runs once
}
void loop()
{
        // repeats
}
```

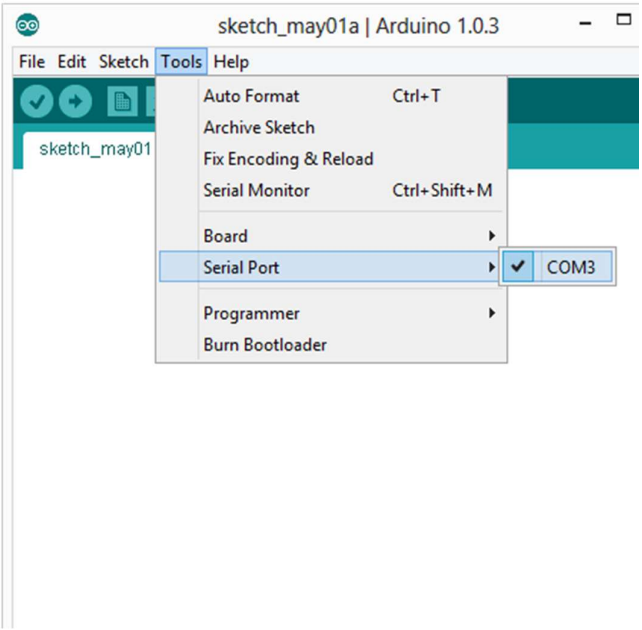


Figure: Settings: Tools → Serial Port

Your computer communicates to the Arduino microcontroller via a serial port → through a USB-Serial adapter. Check to make sure that the drivers are properly installed.

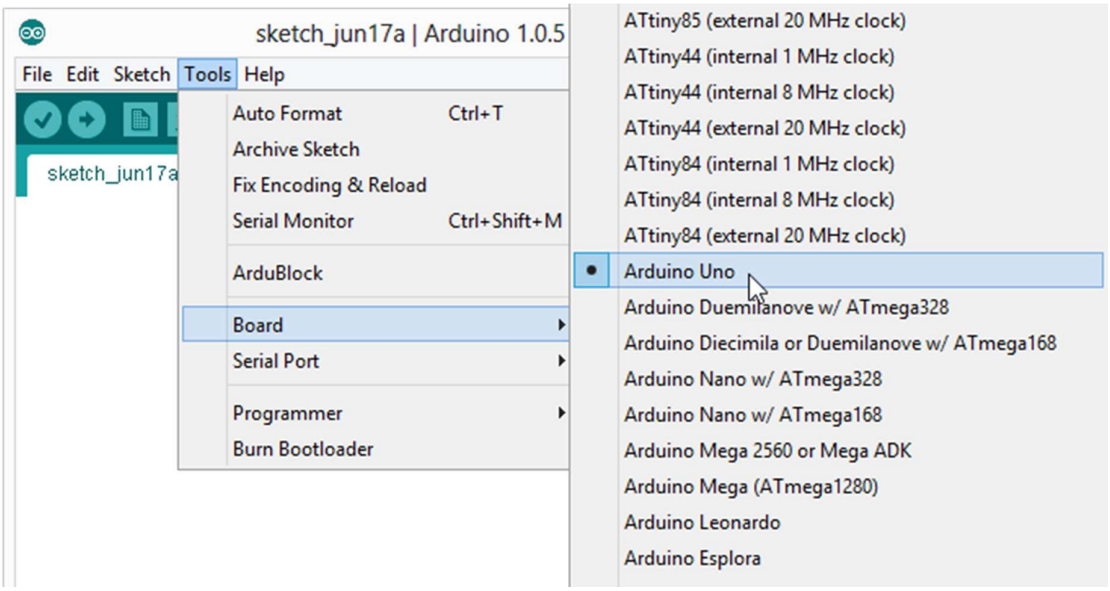


Figure : Settings: Tools → Board

Next, double-check that the proper board is selected under the Tools→Board menu

pinMode(pin, INPUT/OUTPUT);

ex: pinMode(13, OUTPUT);

digitalWrite(pin, HIGH/LOW);
ex: digitalWrite(13, HIGH);

digitalRead(pin);
ex: digitalRead(13);

delay(time_ms);
ex: delay(2500); // delay of 2.5 sec.

Code: Turns on an LED on for one second, then off for one second, repeatedly.

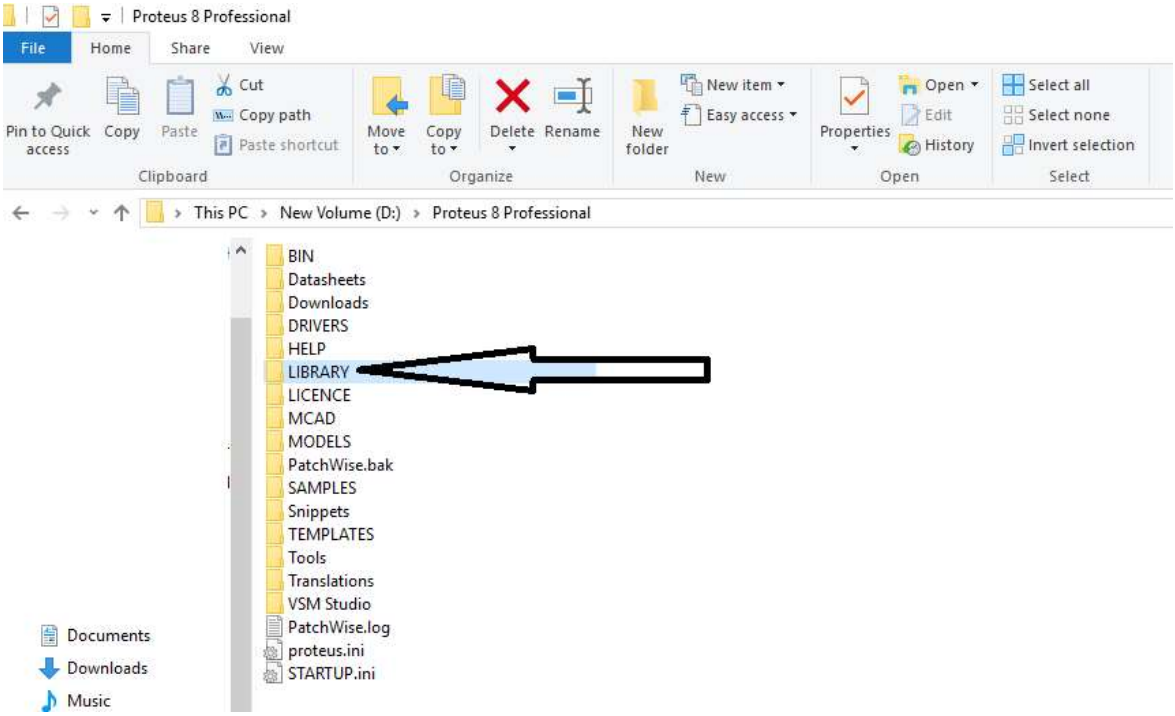
```
// the setup function runs once when you press reset or power the board
void setup()
{
  // initialize digital pin 13 as an output.
  pinMode(13, OUTPUT);
}

// the loop function runs over and over again forever
void loop()
{
  digitalWrite(13, HIGH);   // turn the LED on (HIGH is the voltage level)
  delay(1000);              // wait for a second
  digitalWrite(13, LOW);    // turn the LED off by making the voltage LOW
  delay(1000);              // wait for a second
}
```

How to add Arduino Library into Proteus 8:
Download Arduino
https://www.arduino.cc/en/Main/Software

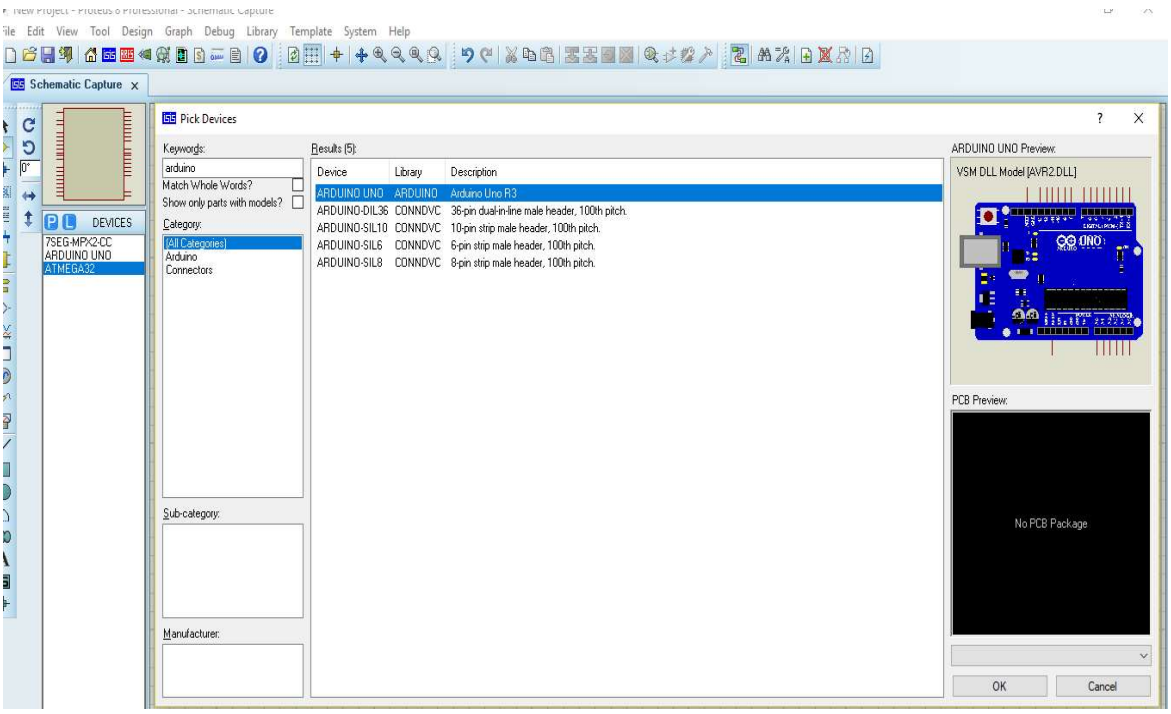
Download arduino library file
ELMS course website → Click course material →Arduino library for proteus→ARDUINO.LIB

Step 1:
Copy ARDUINO.LIB in the in to Library folder;

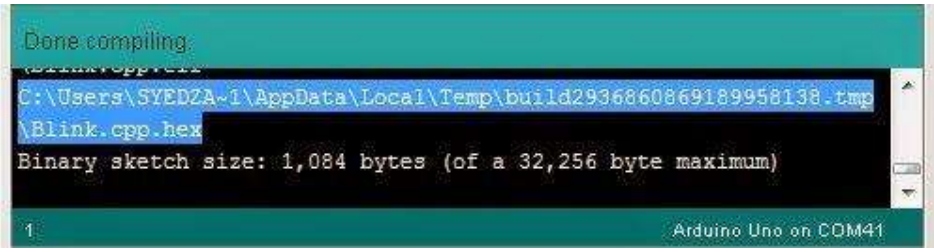
If you are using Proteus 8 then Library folder will be within the data Folder (Proteus 8 Professional\Data\LIBRARY)



Step 2: Now open Your Proteus and Search Arduino

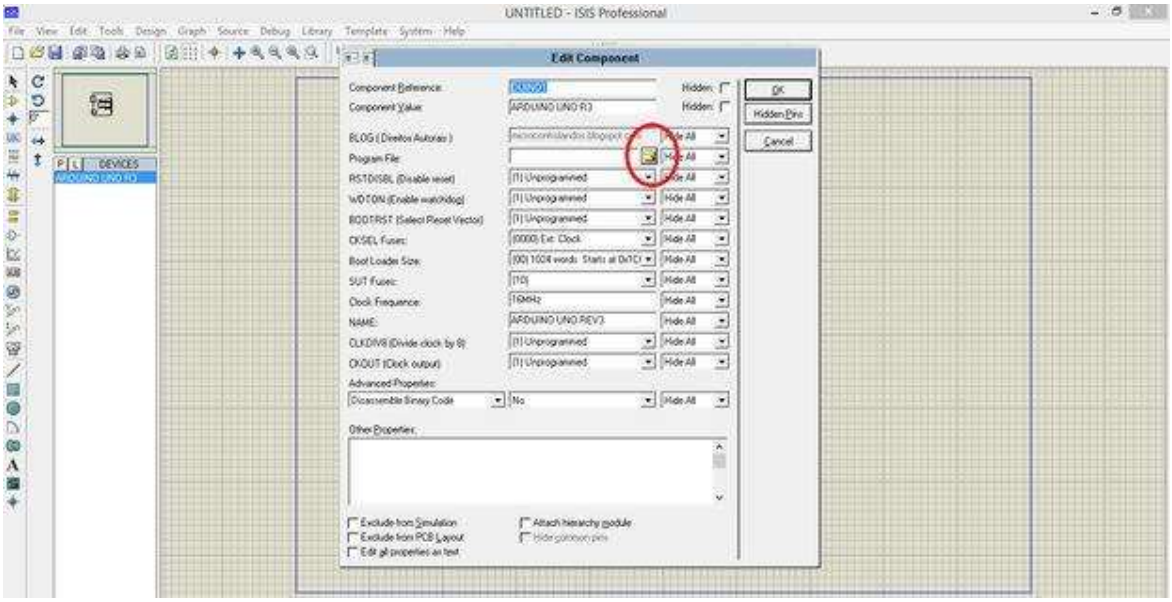


Step 3: Make Hex file



Open arduino Software and Click on file and then Preference and tick both of the Option "Compilation and Upload " After writing your code, click on compile then you will get the Link of Your Hex file at the Output go to that place and get your Hex file.

Step 4: Upload Code



it's Time To upload Code to Arduino.

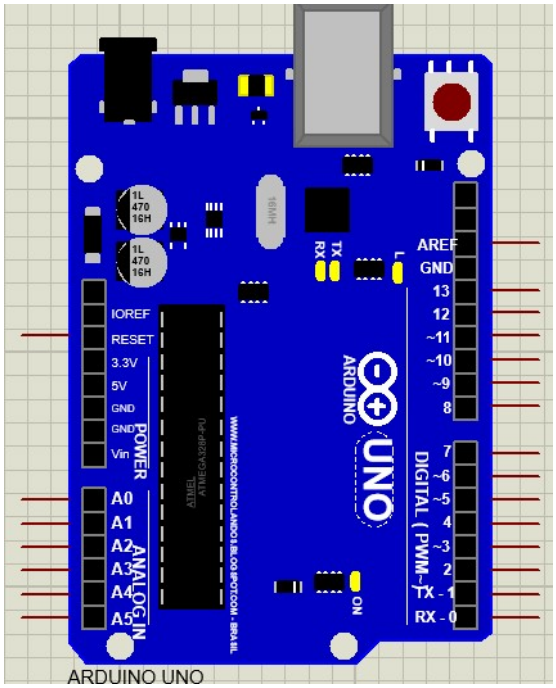


Figure : Arduino uno in proteus

<u>Part B: Serial communication with Arduino.</u>

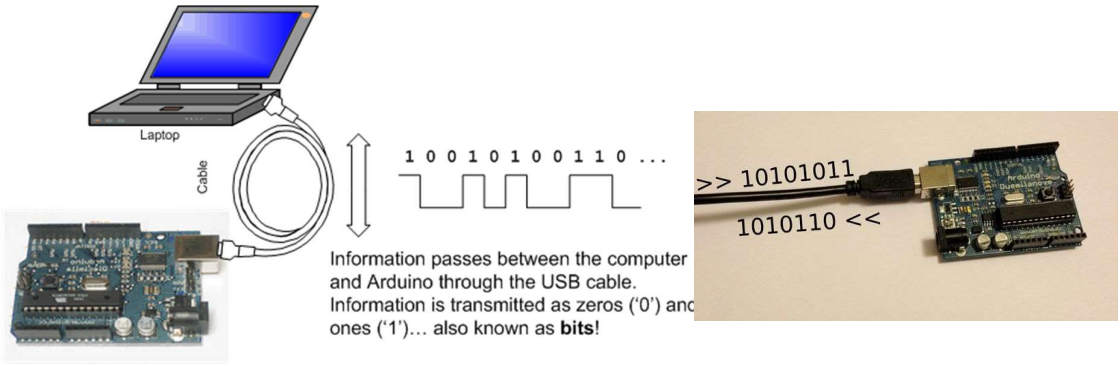


Figure: Serial communication from Arduino to PC.

If you use Arduino connected to a **sensor** (see Fig.1), Arduino produce a series of data that may be sent to a computer to be stored in a file, displayed or processed in some way. If you use the Arduino connected to an **actuator** (see Fig.2), such as a stepper motor, most likely, the computer will send a series of data to the Arduino. The latter will process the received data by converting suitably into commands to send to the motor to make it move in the amount of necessary steps.

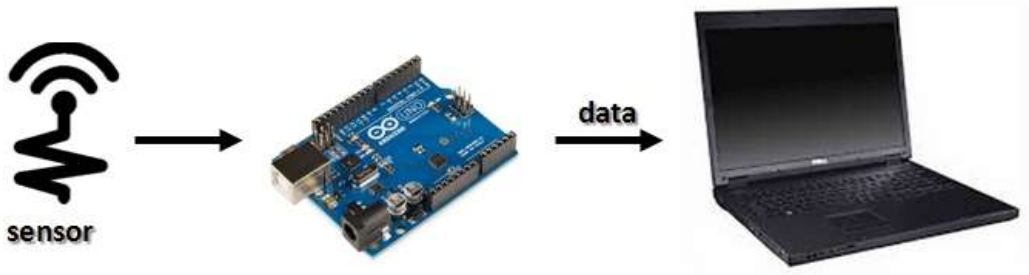


Figure 1

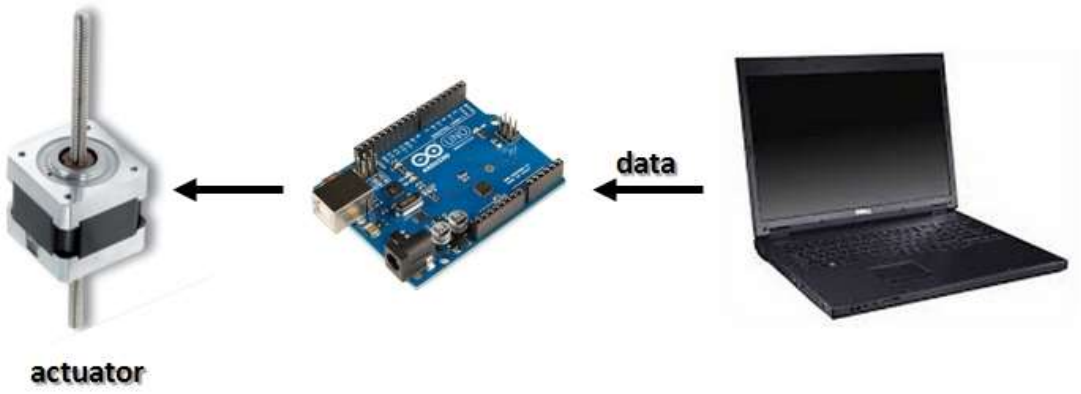
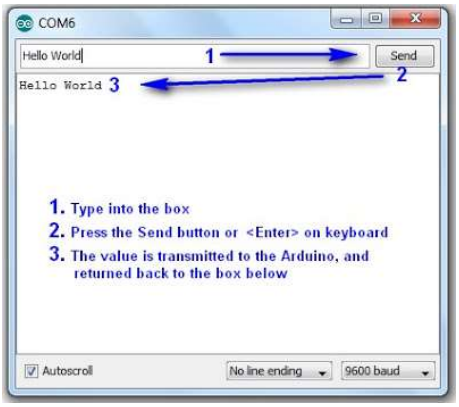
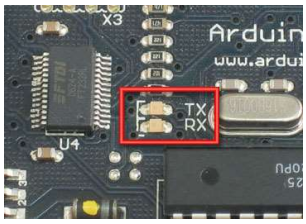


Figure 2





1. Once the Arduino sketch has been uploaded to the Arduino. Open the Serial monitor, which looks like a magnifying glass at the top right section of the Arduino IDE. Please note, that you need to keep the USB connected to the Arduino during this process, as the USB cable is your communication link between your computer and the Arduino.

2. Type anything into the top box of the Serial Monitor and press <Enter> on your keyboard. This will send a series of bytes to the Arduino. The Arduino will respond by sending back your typed message in the larger textbox.

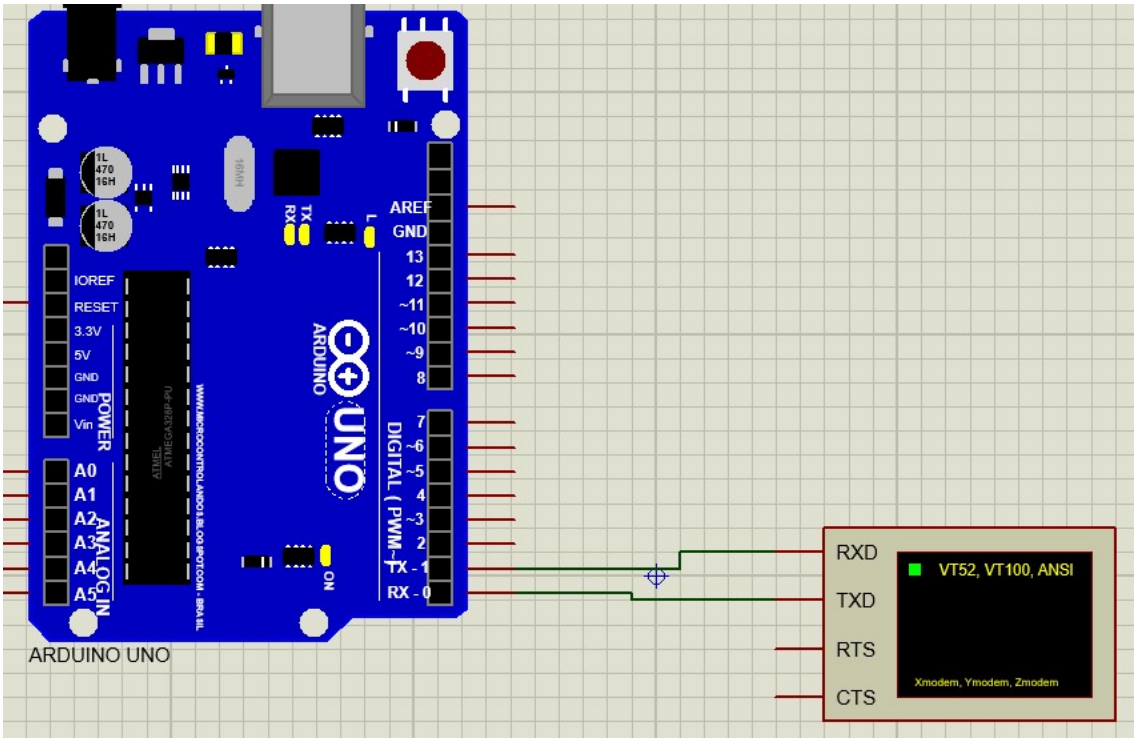


Figure: Proterus “virtual terminal” serial communication simulation

Coode:

```
byte byteRead;

void setup()
{
// Turn the Serial Protocol ON
  Serial.begin(9600);
}

void loop()
{
   /*  check if data has been sent from the computer: */
  if (Serial.available())
  {
    /* read the most recent byte */
    byteRead = Serial.read();
    /*ECHO the value that was read, back to the serial port. */
    Serial.write(byteRead);
    Serial.write("#");

  }
}
```

Part C: Push button interfacing with Arduino

Internal to the Arduino are pullup resistors with a value around 50k-ohm. These resistors can be optionally connected internally using INPUT_PULLUP. This is functionally (and electrically) equivalent to connecting a 50k-ohm resistor between the pin and +5V, the only difference is that it requires no external components and you can turn it on and off in software during the execution of the program.

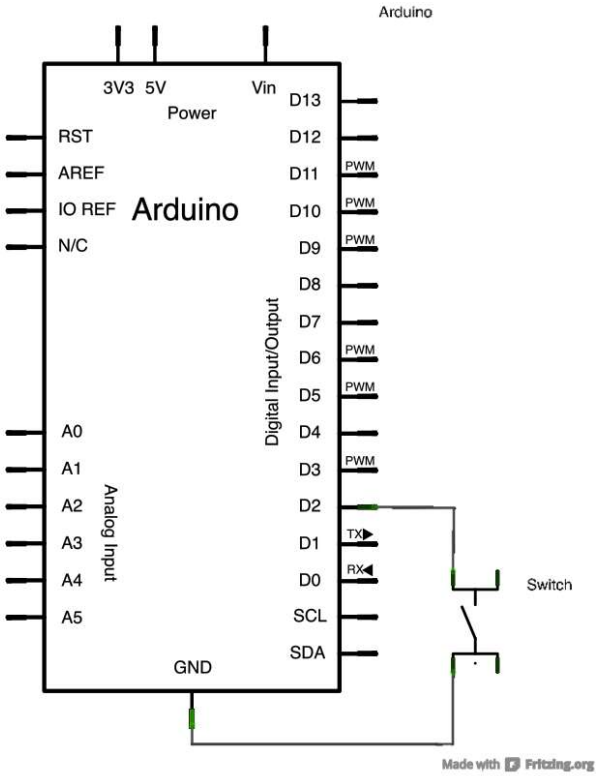


Figure : Push button with Arduino

This example demonstrates the use of INPUT_PULLUP with pinMode().
Code:

```
void setup()
{
  //start serial connection
  Serial.begin(9600);
  //configure pin2 as an input and enable the internal pull-up resistor
  pinMode(2, INPUT_PULLUP);
  pinMode(13, OUTPUT);
}


void loop() {
  //read the pushbutton value into a variable
  int sensorVal = digitalRead(2);
  //print out the value of the pushbutton
  Serial.println(sensorVal);
```

```
 // Keep in mind the pullup means the pushbutton's
 // logic is inverted. It goes HIGH when it's open,
 // and LOW when it's pressed. Turn on pin 13 when the
 // button's pressed, and off when it's not:
 if (sensorVal == LOW)
{
   digitalWrite(13, HIGH);
 } else {
   digitalWrite(13, LOW);
 }
}
```

Part D: Bluetooth Module HC-05 interfacing with Arduino.

Bluetooth is a **standardized protocol (**IEEE 802.15.1**)** for sending and receiving data via a 2.4GHz wireless link. It's a secure protocol, and it's perfect for short-range, low-power, low-cost, wireless transmissions between electronic devices. The range is approximately 10 Meters (30 feet).

HC-05 Specifications

- 2.45Ghz Frequency
- Asynchronous Speed 2.1Mbps (max) .160Kbps
- Security: Authentication
- Profile: Bluetooth Serial Port
- Power Supply: +3.3/5 VDc
- Working Temperature: >20C
- It will communicate at 38400 baud rate.
- Default password of the HC-05 Bluetooth Module: 1234

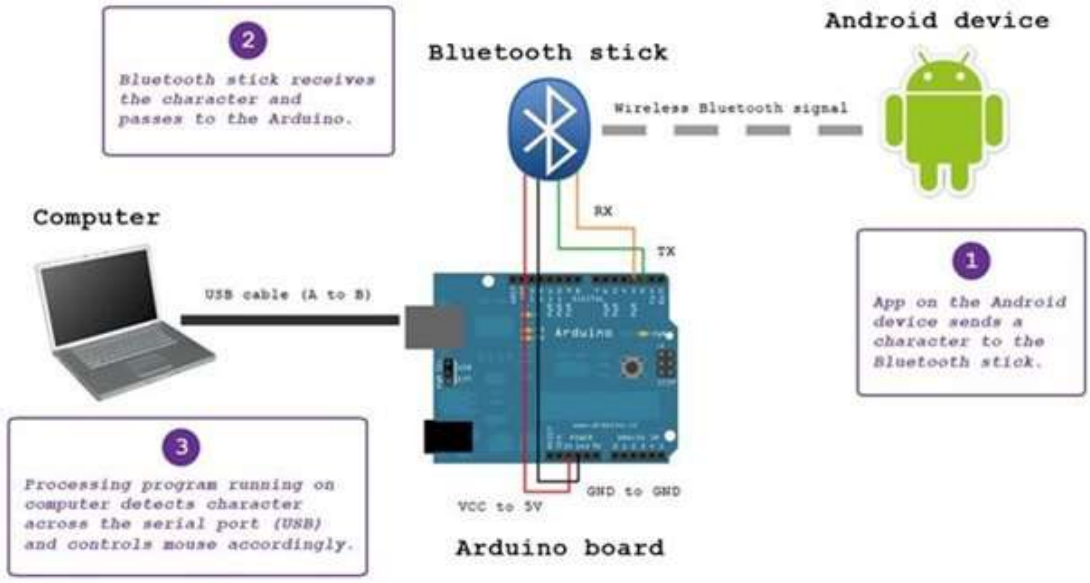


Figure : Smart phone communicating with Arduino board via Bluetooth

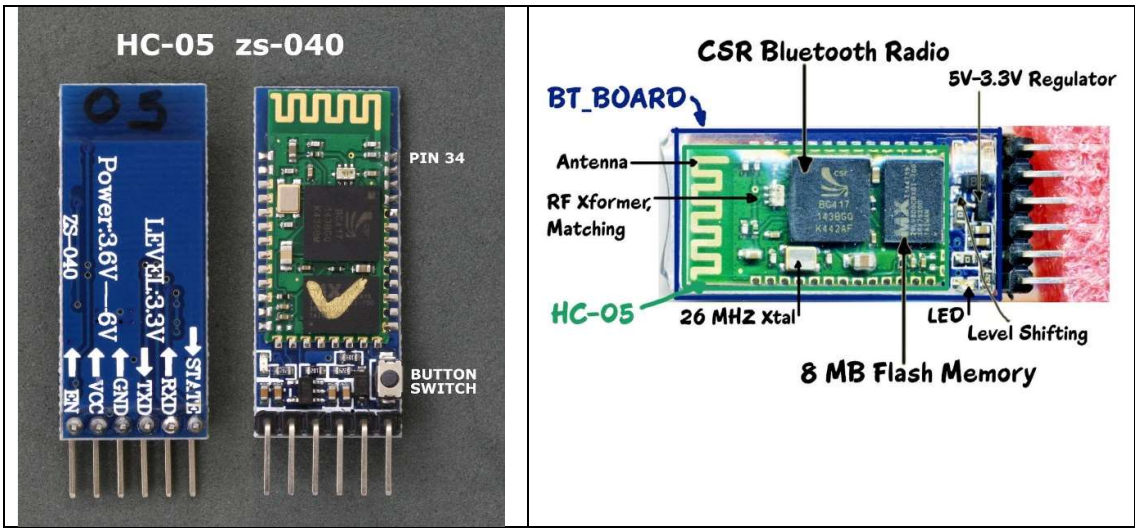


Figure : HC-05 Bluetooth module

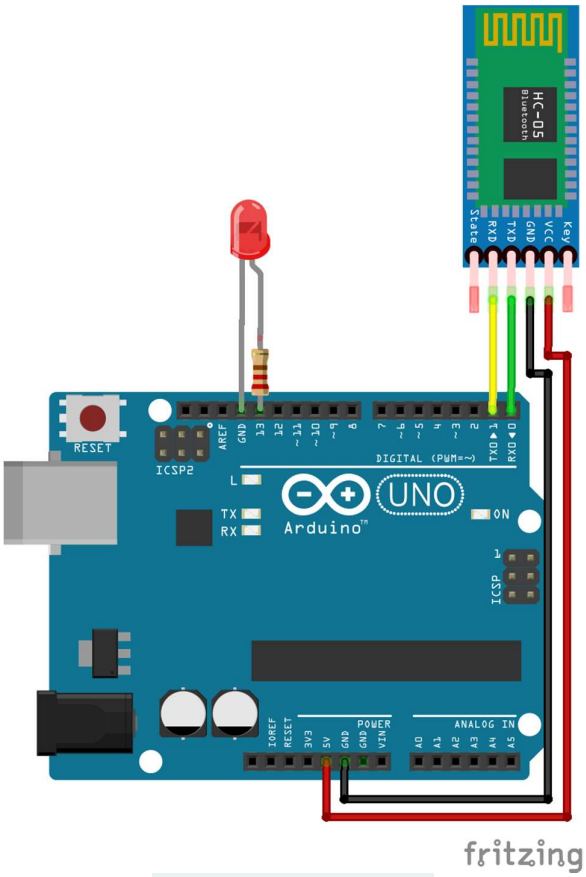
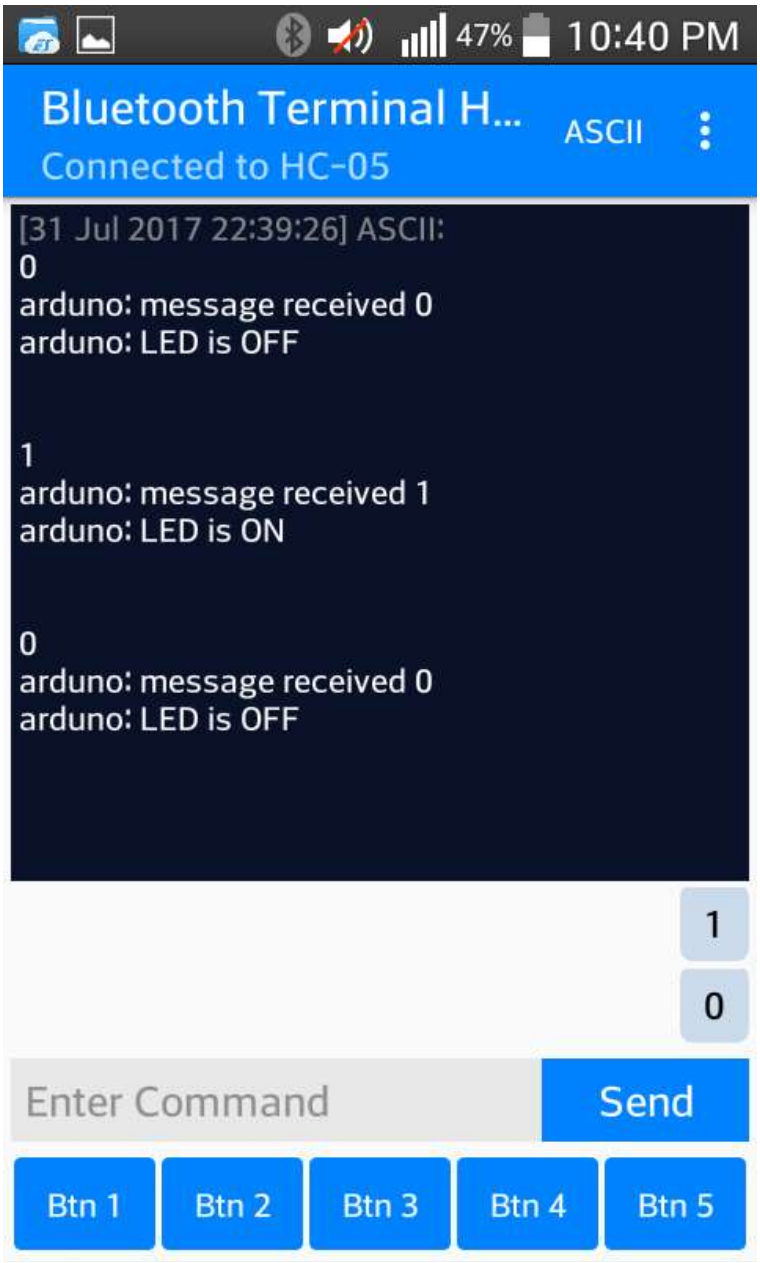


Figure : schematic diag

Code:

```
char data = 0;                //Variable for storing received data

void setup()
{
  Serial.begin(38400);         //Sets the data rate in bits per second (baud) for serial data
transmission
  pinMode(13, OUTPUT);         //Sets digital pin 13 as output pin
}


void loop()
{

  if(Serial.available() > 0)  // Send data only when you receive data:
  {
    data = Serial.read();      //Read the incoming data and store it into variable data
    Serial.write(data);        //Print Value inside data in Serial monitor
    Serial.print("\n");        //New line

    if(data == '1')            //Checks whether value of data is equal to 1
    {
      digitalWrite(13, HIGH);  //If value is 1 then LED turns ON
      Serial.write("arduno: message received ");
      Serial.write(data);
      Serial.print("\n");        //New line
      Serial.write("arduno: LED is ON");
    }
    else if(data == '0')       //Checks whether value of data is equal to 0
    {
     digitalWrite(13, LOW);   //If value is 0 then LED turns OFF
     Serial.write("arduno: message received ");
     Serial.write(data);
     Serial.print("\n");        //New line
     Serial.write("arduno: LED is OFF");
    }
  }

}
```



Reference:

https://www.arduino.cc/
https://www.arduino.cc/en/Reference/Serial
https://www.arduino.cc/en/Tutorial/InputPullupSerial
http://www.meccanismocomplesso.org/en/arduino-tutorial-serial-data-actuator/