IEEE QP Wiki

Engineer. Design. Innovate. Mentor. Showcase

Bluetooth

Prerequisites

- Arduino Uno
- HC-05 Bluetooth Module
- Putty
- Computer with Bluetooth Connection
- For Example Use:
  - MPU-6050
  - Python 3
  - Python IDE of your choice (Sublime, Atom, Visual Studio Code, etc.)

Setting Up Your Bluetooth Module with AT Commands

The first thing that you need to do is wire up your HC-05 with your arduino. Don't plug in your arduino before you plug in your HC-05, you will see why in a bit!

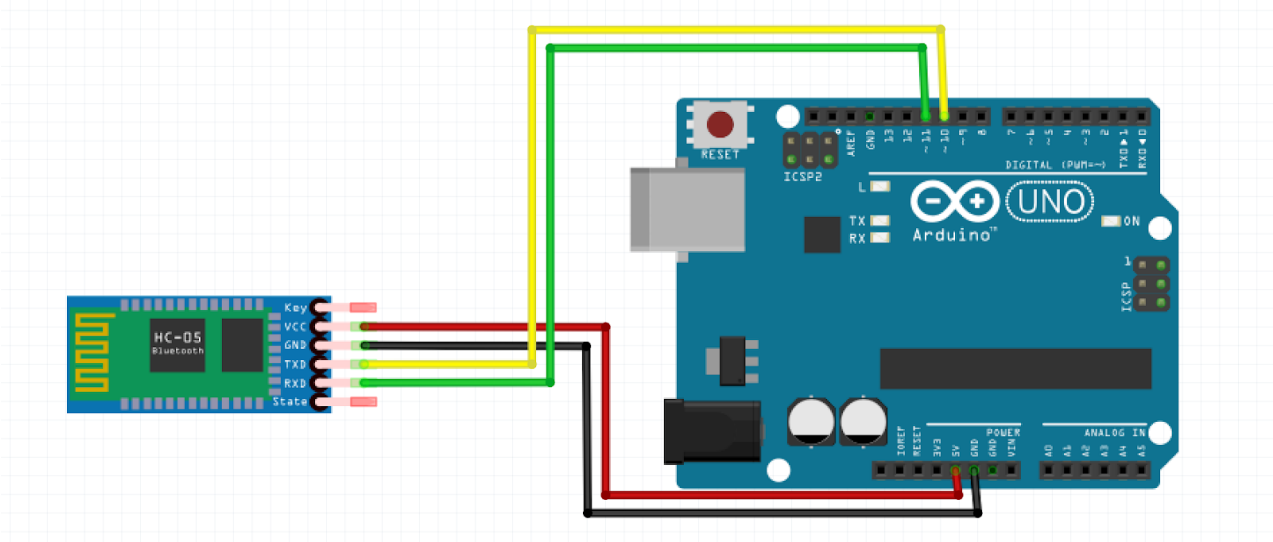
Connect the HC-05 to the Arduino as such:

TXD > D10

RXD > D11

+5V > 5V

GND > GND



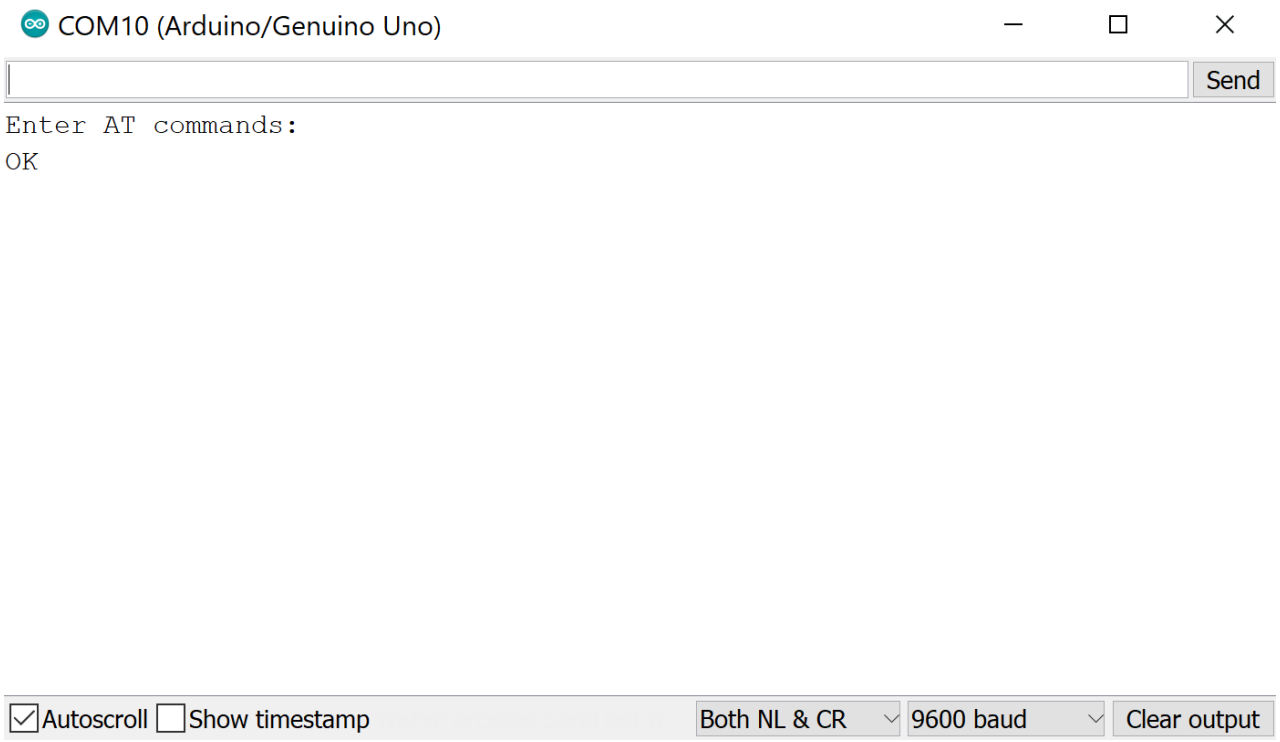
Wiring Diagram for HC-05

Now that you have plugged everything, hold the button on the HC-05 as you are connecting power to your arduino, until the LED onboard the HC-05 starts blinking approximately every 2 seconds. This puts the HC-05 into AT mode, which is very useful for configuring your arduino! Using AT mode you can configure the name of your module, so you don't confuse your module with every other HC-05, and be able to change things like the password and data transfer rates of the HC-05 as well!

CODE FOR ENTERING AT COMMANDS

```
#include <SoftwareSerial.h>
SoftwareSerial BTSerial(10, 11); // RX | TX
void setup()
{
  Serial.begin(9600);
  Serial.println("Enter AT commands:");
  BTSerial.begin(38400);  // HC-05 default speed in AT command more
}
void loop()
{
  // Keep reading from HC-05 and send to Arduino Serial Monitor
  if (BTSerial.available())
  Serial.write(BTSerial.read());
  // Keep reading from Arduino Serial Monitor and send to HC-05
  if (Serial.available())
  BTSerial.write(Serial.read());
}
```

After you have uploaded the code above, go to the Serial Monitor, and select 9600 Baud for your baud rate and Both NL and CR as your output mode from Newline. Type into the serial input, AT. If you set up everything correctly, the bluetooth sensor should respond and say OK



Bluetooth!

Now that this works, you can change a bunch of other settings using AT commands!

Useful AT Commands:

To test whether AT communication is working:

AT

To change the name of your module to "Quarterly_Projects"

AT+NAME = *Quarterly_Projects*

To change the password of your module from 1234 to "7890"

AT+PSWD = 7890

Here you can find a lot more useful commands to further configure your HC-05!

https://www.itead.cc/wiki/Serial_Port_Bluetooth_Module_(Master/Slave)_:_HC-05#AT_command_Default

TL;DR – Setting Up Your Bluetooth Module with AT Commands

1. Set up the circuit as shown in the picture above
2. Hold the Button on the HC-05
3. As you are holding the button plug in the arduino, until the HC-05 starts blinking in 2 second intervals for AT mode
4. Run the code
5. Go to the Serial monitor and switch the baud rate to 9600 and output to Both NL and CR
6. Type AT into the Serial monitor and (hopefully) get an OK back

Controlling your Arduino with Bluetooth Serial.read()

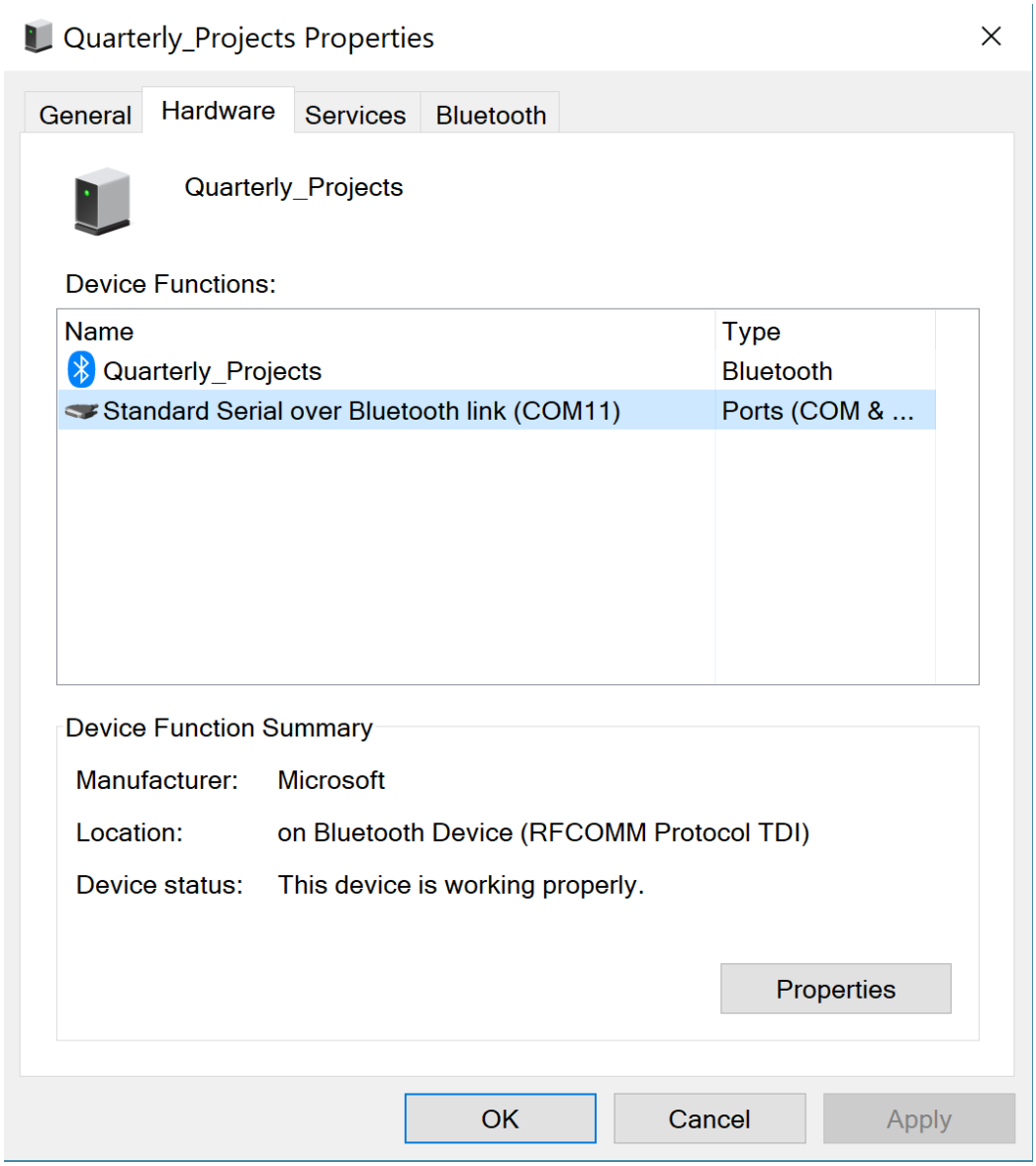
If you are wondering how bluetooth with the HC-05 works, it sends serial data (sending data one bit at a time) over the bluetooth frequency. Since the HC-05 is just sending and receiving bits sequentially, it's very easy to use this to your advantage for communication between external devices.

Just like how you communicated with the HC-05 via serial with the Arduino IDE, you can also use other programs, with your phone, but with this example, your computer. This example requires that your computer has a bluetooth connection, although most computers should have a bluetooth communication. You could do this part with really any device that is compatible with the HC-05 (Not iOS devices because of MFi Licensing)

Now that you have configured your HC-05 you can now connect to it on your device. Check the COM Port in the device settings of your laptop. Make sure to unplug and replug the HC-05 into your Arduino without holding the button on the HC-05 to start it in the standard communication mode!

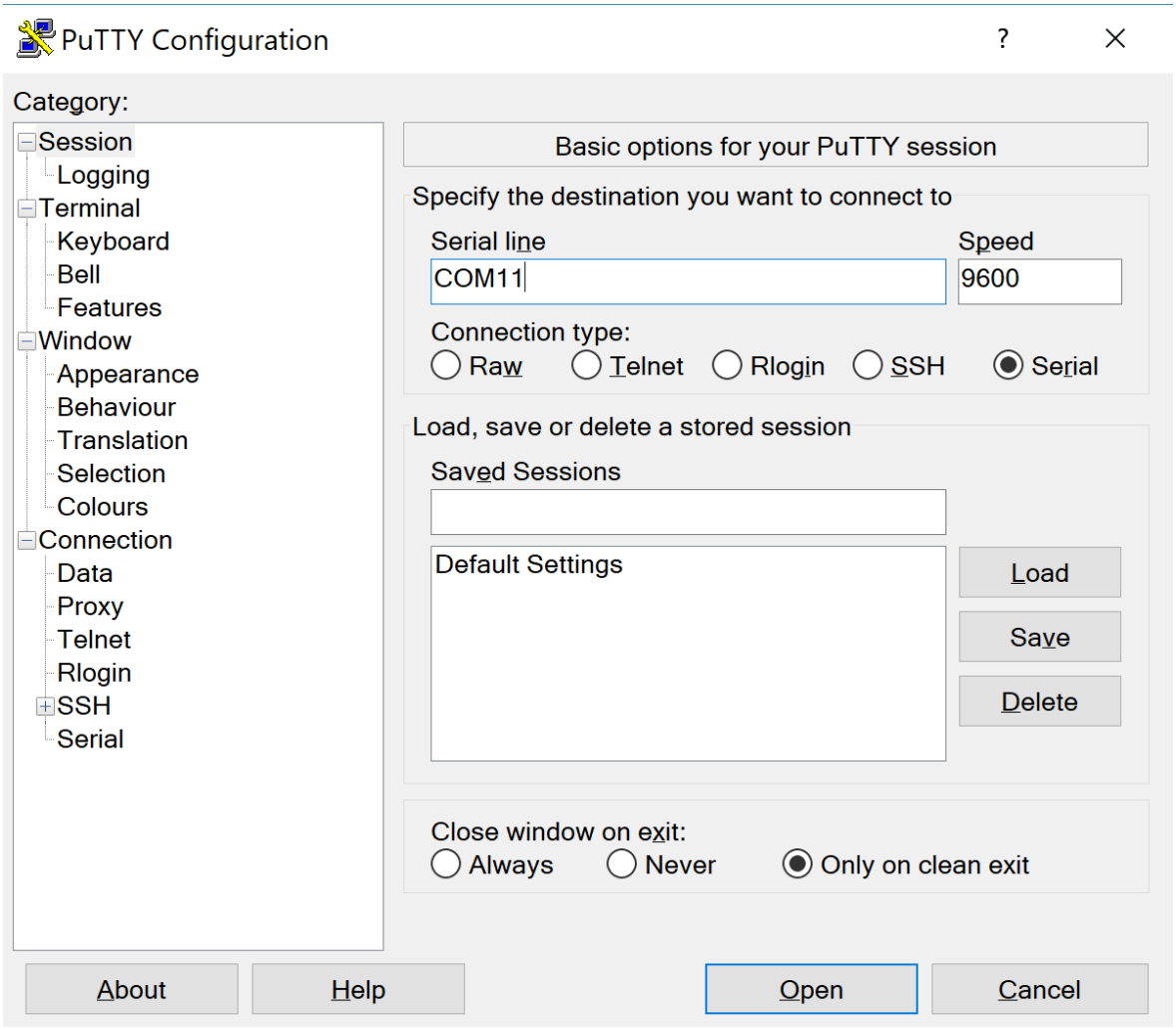
(Windows) **https://www.fugawi.com/knowledge_base/document/FF12906**

(OSX) **https://todbot.com/blog/2006/02/23/howto-mac-os-x-bluetooth-serial-port/**



Now you know the COM port of the HC-05 you can now connect to it via serial with your SSH program of choice. I am using Putty in this example, but it should be similar for connecting it with any similar program. As you can see above, the COM port for the HC-05 is COM11.

Go to the Serial Connection type, and use COM11 (The COM port with my HC-05 is COM11), with 9600 Baud, which is the default for the HC-05 for reading and writing.

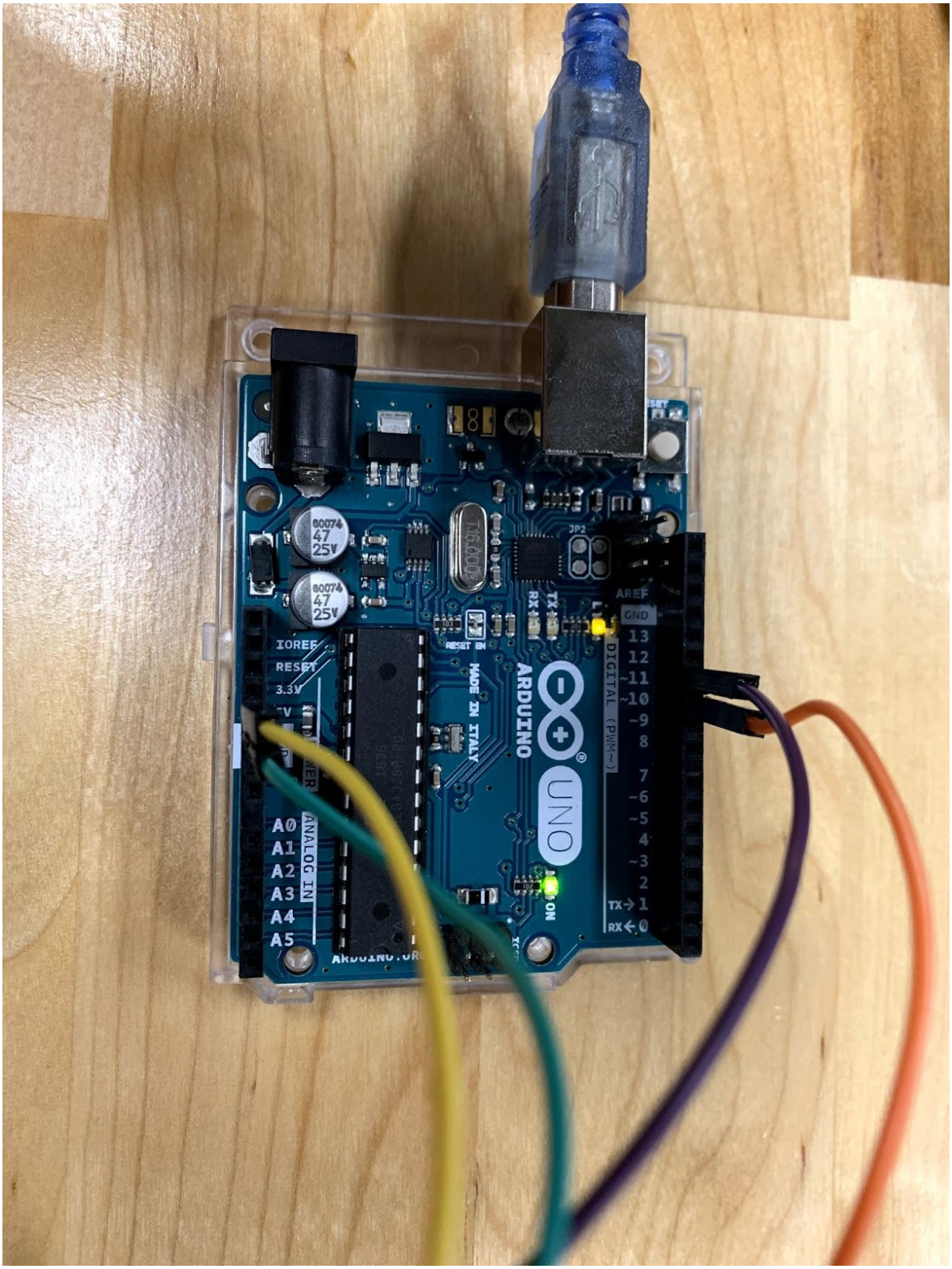


Your Putty should then open successfully and open a terminal! Then upload the code on the next page, and read the comments for how the code works!

CODE FOR CONTROLLING LED WITH BLUETOOTH

```
#include <SoftwareSerial.h>
SoftwareSerial BTSerial(10, 11); // RX | TX
void setup()
{
  pinMode(LED_BUILTIN, OUTPUT);
  Serial.begin(9600);
  BTSerial.begin(9600);  // HC-05 default speed in Standard mode
}
void loop()
{
  // Keep reading from HC-05 and send to Arduino Serial Monitor
  if (BTSerial.available()){ // Check if the Serial is reading properly
    char c = BTSerial.read();
    if(c == '1'){ //If 1 is detected in Putty
      digitalWrite(LED_BUILTIN, HIGH); //Set the LED to High
      BTSerial.println("LED On!"); //Write this to the HC-05's serial
    }
    if(c == '0'){ //If 0 is detected in Putty
      digitalWrite(LED_BUILTIN, LOW);
      BTSerial.println("LED Off!");
    }
  }
}
```

Then, you should see the LED turn on and off as you enter in 0 or 1 into the terminal and also see status updates printing to the terminal as well!



Now that you have this simple example working, you can now apply this to many other devices connected to your arduino, for example motors and displays!

TL;DR – Controlling your Arduino with Bluetooth Serial.read()

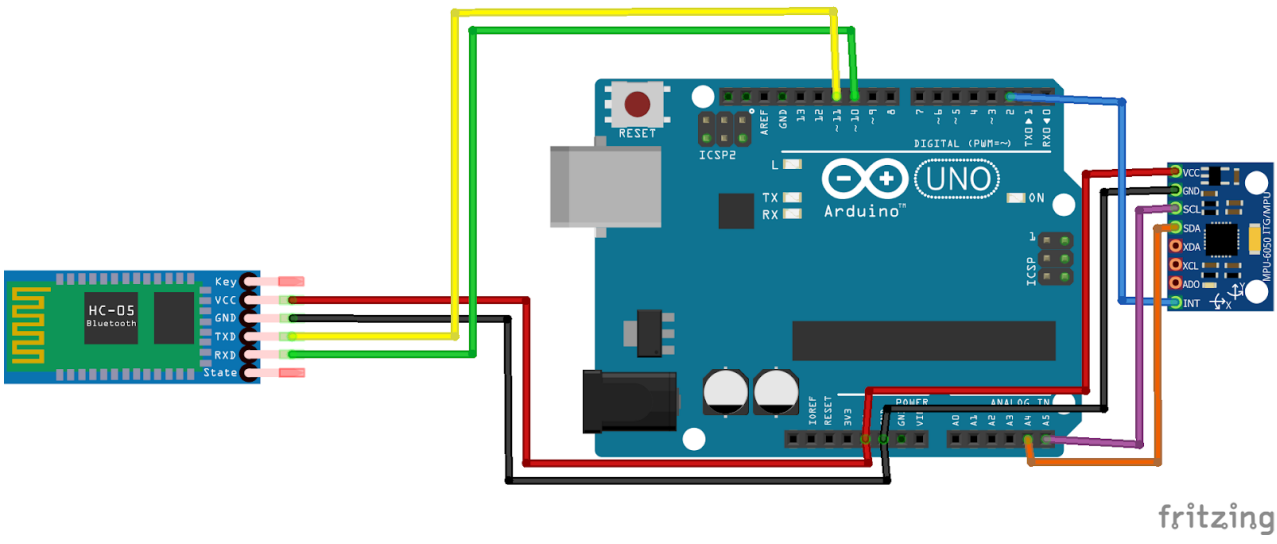
1. Start the HC-05 in standard mode by not holding the button on startup
2. Connect to your bluetooth module
3. Find the COM Port of the bluetooth
4. Connect to the Bluetooth module using any serial terminal
5. Upload the code for controlling the LED
6. Press 0 or 1 on your keyboard
7. Watch the LED turn on and off

Using the HC-05 with Bluetooth Serial.print() and Python

Another thing that you can do with the HC-05 is send data and other information over from the Arduino to your device of choice. This allows you to display or store the data that you received from an arduino!

In this example, we are using an MPU-6050, which is a very versatile sensor as it has both Accelerometer/Gyrometer capabilities, and a temperature sensor, making it very useful for a wearable! Hook it up like the diagram below to connect the MPU-6050 to the arduino.

Wiring Diagram



Wiring Diagram

For HC-05

TXD > D10

RXD > D11

+5V > 5V

GND > GND

For MPU-6050

INT > D2

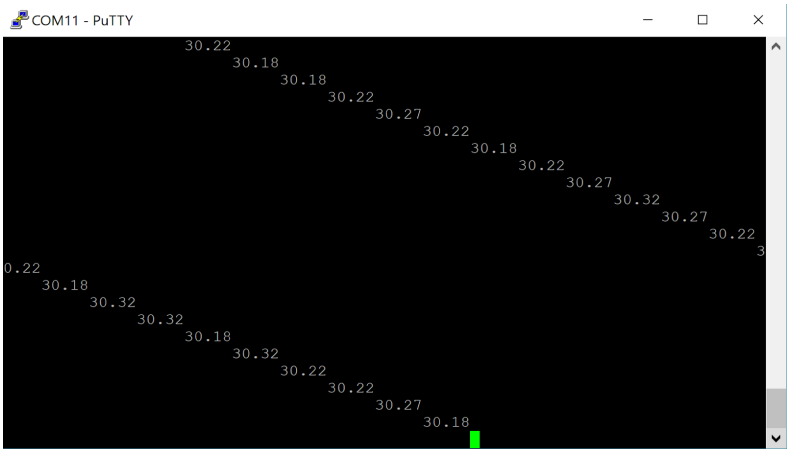
SCL > A5

SDA > A4

VCC > 5V

GND > GND

The MPU 6050 outputs its data to registers on the arduino, and the code below reads the data from each register representing the acceleration, orientation, and temperature. Then it sends the temperature to the computer using bluetooth serial, which you can in fact read with putty as shown in the previous example! The output from putty should look a little bit funky, but that's okay, the python code should parse this with no problem.



ARDUINO CODE SENDING TEMPERATURE WITH SERIAL.PRINT

```
#include <SoftwareSerial.h>
SoftwareSerial BTSerial(10, 11); // RX | TX
#include<Wire.h>
const int MPU_addr=0x68;// I2C address of the MPU-6050
int16_t AcX,AcY,AcZ,Tmp,GyX,GyY,GyZ;
String newTmp;
void setup()
{
  pinMode(LED_BUILTIN, OUTPUT);
  Serial.begin(9600);
  BTSerial.begin(9600);  // HC-05 default speed in AT command more
    Wire.begin();
    Wire.beginTransmission(MPU_addr);
    Wire.write(0x6B);  // PWR_MGMT_1 register
    Wire.write(0);     // set to zero (wakes up the MPU-6050)
    Wire.endTransmission(true);
}
void loop()
{
  Wire.beginTransmission(MPU_addr);
  Wire.write(0x3B);  // starting with register 0x3B (ACCEL_XOUT_H)
  Wire.endTransmission(false);
  Wire.requestFrom(MPU_addr,14,true);  // request a total of 14 registers
  //Values for the acceleration in each direction
  AcX=Wire.read()<<8|Wire.read();  // 0x3B (ACCEL_XOUT_H) & 0x3C (ACCEL_XOUT
  AcY=Wire.read()<<8|Wire.read();  // 0x3D (ACCEL_YOUT_H) & 0x3E (ACCEL_YOUT
  AcZ=Wire.read()<<8|Wire.read();  // 0x3F (ACCEL_ZOUT_H) & 0x40 (ACCEL_ZOUT
  //Values for the temperature read from the register
  Tmp=Wire.read()<<8|Wire.read();  // 0x41 (TEMP_OUT_H) & 0x42 (TEMP_OUT_L)
  //Values for the orientation in each direction
  GyX=Wire.read()<<8|Wire.read();  // 0x43 (GYRO_XOUT_H) & 0x44 (GYRO_XOUT_L
  GyY=Wire.read()<<8|Wire.read();  // 0x45 (GYRO_YOUT_H) & 0x46 (GYRO_YOUT_L
  GyZ=Wire.read()<<8|Wire.read();  // 0x47 (GYRO_ZOUT_H) & 0x48 (GYRO_ZOUT_L
  newTmp = Tmp/340.00+36.53; //equation for temperature in celsius from MPU6
  if (BTSerial.available()){
  BTSerial.print(newTmp);  //Write temperature values to the HC-05
  BTSerial.print('n');
  }
   delay(500);
 }
```

Now that you are outputting the values from the MPU-6050, you can read the serial output using Python, and then plot it! We are using Python's serial library to read the output from bluetooth and then plotting the temperature data using Matplotlib and NumPy.

If you have python installed, the only setup that you need for this is to install Matplotlib, which is a dependency for this program in order to plot the temperature values. Go into powershell or any other terminal and install it using the python package manager, pip.

```
pip install matplotlib
```

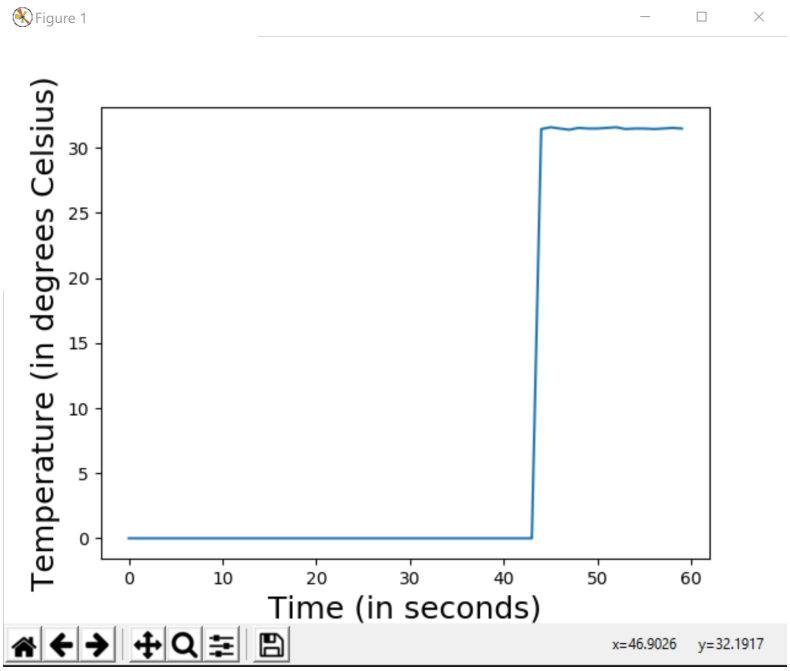
Run the code below when you save it as a .py file. For example if you save the file as plottemp.py, run it with

```
python plottemp.py
```

PYTHON CODE FOR DISPLAYING TEMPERATURE

```
import serial
import time
import matplotlib
matplotlib.use("tkAgg")
import matplotlib.pyplot as plt
import numpy as np
ser = serial.Serial('COM11', 9600, timeout=5) #timeout of 5 to give python p
ser.flushInput()
#plot window of 60 seconds
plot_window = 60
y_var = np.array(np.zeros([plot_window]))
plt.ion()
fig, ax = plt.subplots()
plt.xlabel('Time (in seconds)',fontsize = 18)
plt.ylabel('Temperature (in degrees Celsius)', fontsize=18)
line, = ax.plot(y_var)
while True:
   try:
       #read bits from serial
       ser_bytes = ser.readline()
       try:
           #transform bytes into ascii code
           decoded_bytes = float(ser_bytes.decode("ascii"))
           print(ser_bytes.decode("ascii"))
       except:
           continue
       #graph data from MPU6050
       y_var = np.append(y_var,decoded_bytes)
       y_var = y_var[1:plot_window+1]
       line.set_ydata(y_var)
       ax.relim()
       ax.autoscale_view()
       fig.canvas.draw()
       fig.canvas.flush_events()
   except:
       print("Keyboard Interrupt")
       break
```

After running the code above, you should get a graph of the temperature from the MPU-6050, plotted via Matplotlib!



Now play around with the code to apply it to your own use case!

References

Arduino and HC-05 Bluetooth Module Tutorial

https://www.itead.cc/wiki/Serial_Port_Bluetooth_Module_(Master/Slave)_:_HC-05#AT_command_Default

https://playground.arduino.cc/Main/MPU-6050

https://matplotlib.org/contents.html

Edit