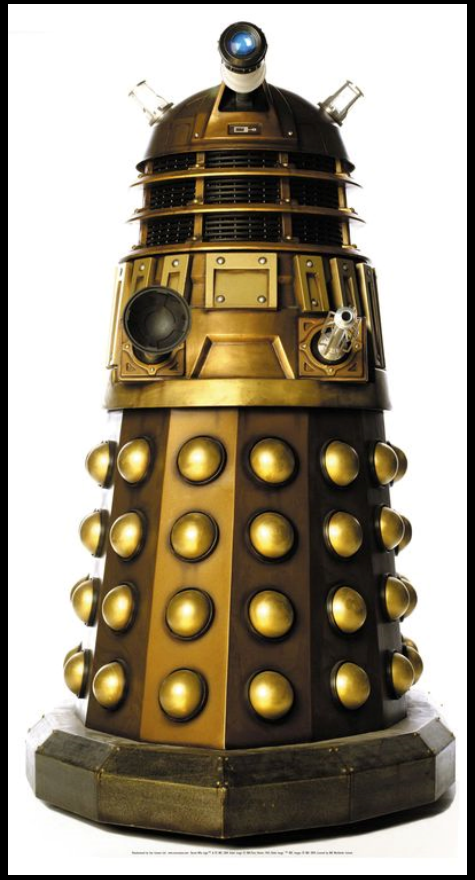
Hardware and Python

Steve Granda

Why would I want to do this?

- You want to make something interactive.
- You want to recreate sputnik.
- You want to measure data from hardware or sensor.
- You want to begin reverse engineering hardware.
- You want to create a robotic army of minions.

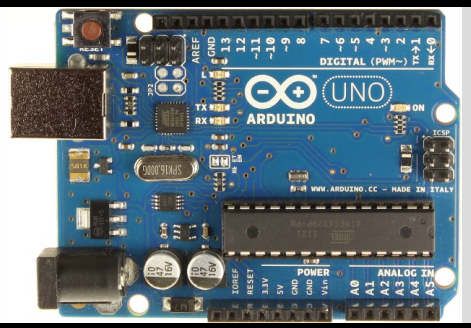
Yeah, well, you know, that's just, like, your opinion, man.

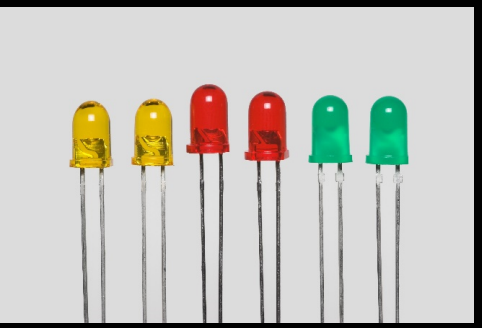


What are we using today?

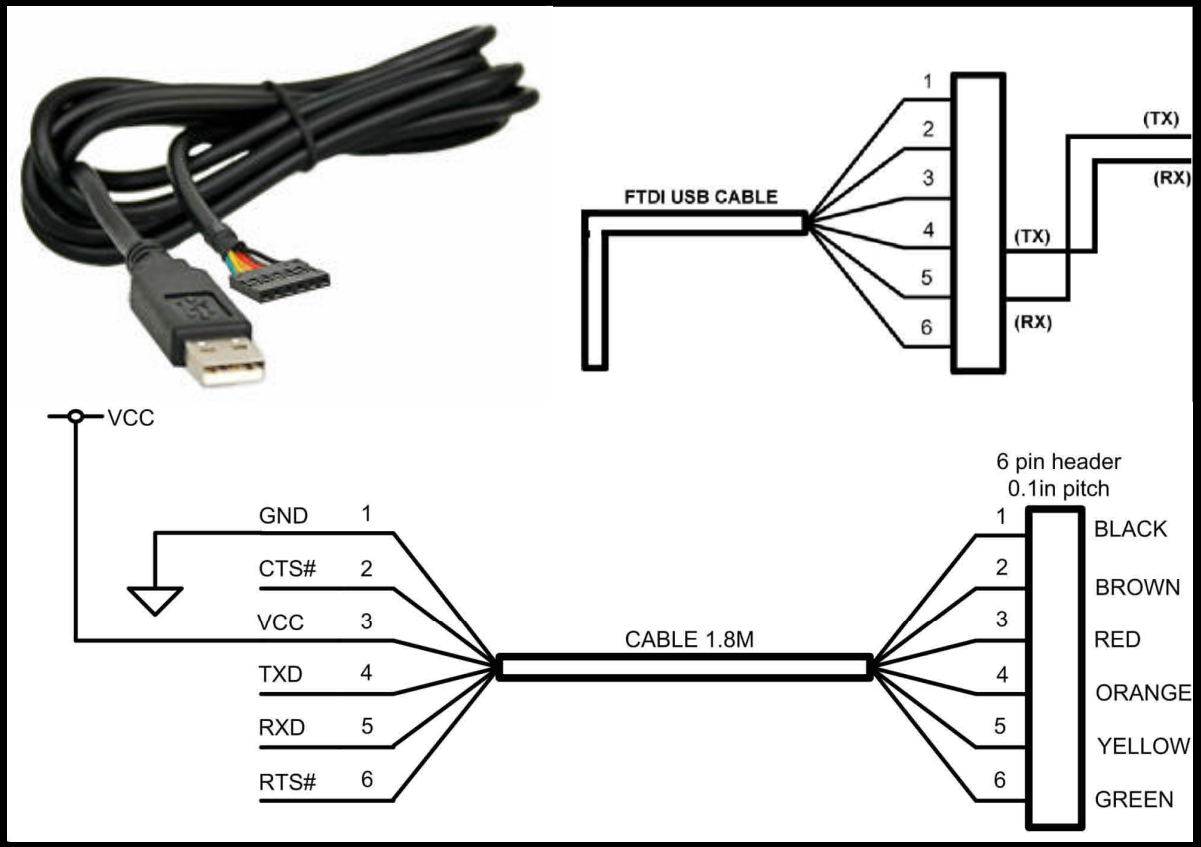
- Python
- FTDI Cable (Serial to USB)
- Arduino Duemilanove
- LED’s

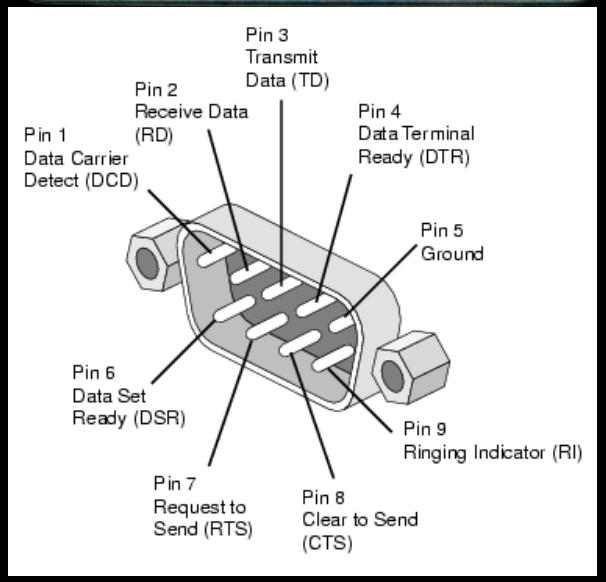




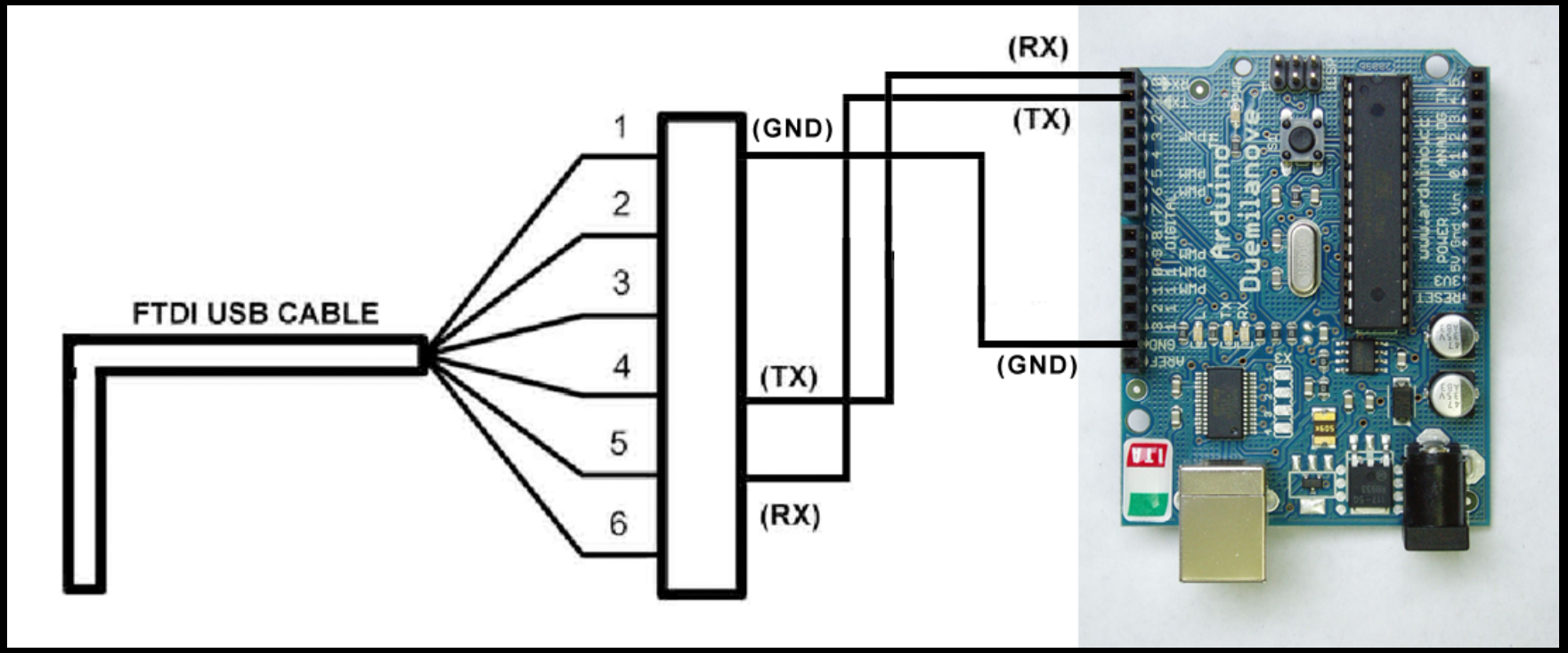


Wired Hardware Communication

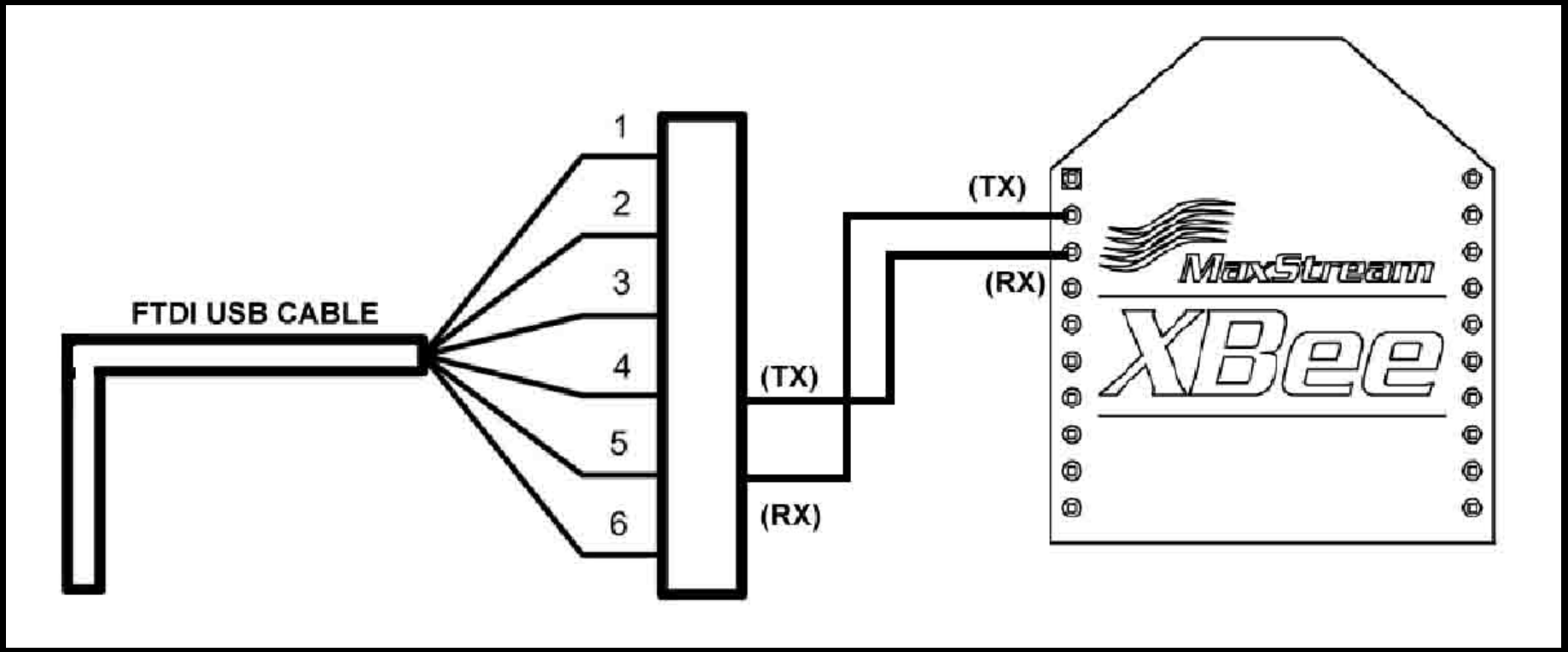




FTDI to Python



Wireless Hardware Communication



Programming the Arduino

- You need to tell the Arduino how to interpret what it receives on the FTDI cable by Python.
- A few simple Serial Commands for the Arduino
  More can be found here:
  http://arduino.cc/en/reference/serial
- Serial port is set at 9600

```
...

//SETUP PINS and UART
void setup()
{
  //Begin Serial Communication
  Serial.begin(9600);
...
Serial.println("INITIALIZING.....");
  Serial.print("\n");
  Serial.print("\r");
  Serial.println("BOARD WAS RESET");
```

Finally Python

- On Linux/Mac machines, this will be a bit easier as most of the time you'll just talk to the FTDI by opening a file:

  /dev/ttyUSB0

- Windows users might have a bit more trouble as the port isn't a file and gets enumerated via Com 1, Com 3, etc.

- Make sure to set the baud correctly otherwise you'll just see garbage printed out (if you're lucky).

```
import serial, time

#Different for Linux, Mac, Win
port= '/dev/ttyUSB0‘
baud= 9600  #Remember this from Arduino
times = 1       #The time we take to timeout (1 second)

device = serial.Serial(port, baud, timeout = times)

while 1 :
                input = raw_input(">> ")
                device.write(input + '\r\n')
...
```

Other Ways

- There are various ways of communicating with the device over serial since you're technically just writing out to a file.
- In the previous example we used the serial library and opened:
  /dev/ttyUSB0
- Now try opening /dev/ttyUSB0 as a file and write something to it.

```
import serial

#Different for Linux, Mac, Win
port= '/dev/ttyUSB0‘
baud= 9600  #Remember this from Arduino
times = 1       #The time we take to timeout (1 second)

device = open(port, “r+”)

while 1 :
            input = “your input here”
            device.write(input)
…
```

Oddities

- Sometimes you will:
  - See garbage when reading the line.
    (Try wiggling the TX/RX cables while reading the “/dev/ttyUSB0” file)
    (Baud Setting, cables are loose, or you’re listening to dubstep next to the board)
  - Words will get chopped off.
    (The hardware/file buffers aren’t full or got flushed early.)
  - It will be very slow.
    (9600 baud man. 9600 bits per second)
    (8 bits = 1 byte. An ASCII character is about 8 bits)
  - It just wont work.
    (Solar Flares, Cables popped out, you don’t have administrative rights to open the file, ???)

Where to go from here

- With this basic information you can now buy an FTDI cable and connect it to random devices that have UART/Serial pins.

  This means you can now read data from hardware or sensors and write it out to a file, parse it, or manipulate it.

- There are many protocols that you can also manipulate in a similar way requiring other hardware similar to an FTDI cable which can be read by python

  CAN BUS (Controller Area Network) -- Hack your car's sensors.
  I²C (Inter-Integrated Circuit) -- Read other sensors
  Devices in /dev/* -- Write py-scripts to annoy co-workers
  or just mess up your computer horribly