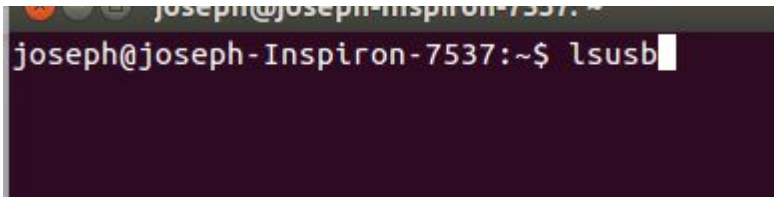
This is a step-by-step guide for establishing a connection with an Olimex ESP8266-EVB using Ubuntu.

1. Testing connection

Firstly we're going to make sure that our device correctly connects with the computer, and establish a way of communicating with it over the serial line. Go to the Ubuntu Software Centre, type "serial" into the search bar at the top right. The first result should be "Serial port terminal". Install it.

Plug in USB-serial converter (for Ubuntu there shouldn't be any drivers needed). To make sure it connected properly go to a terminal and type in: `lsusb`



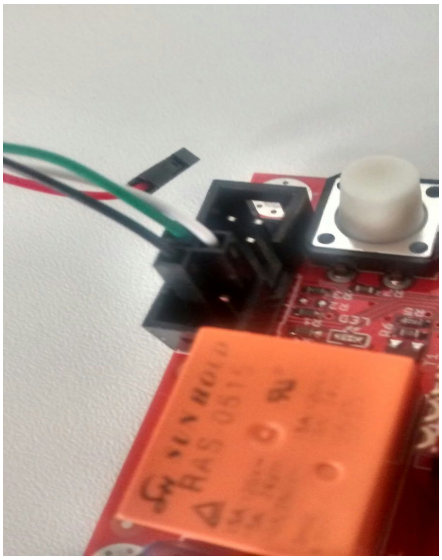
You should see something like:

```
joseph@joseph-Inspiron-7537:~$ lsusb
Bus 001 Device 002: ID 8087:8000 Intel Corp.
Bus 001 Device 001: ID 1d6b:0002 Linux Foundation 2.0 root hub
Bus 003 Device 001: ID 1d6b:0003 Linux Foundation 3.0 root hub
Bus 002 Device 005: ID 04f3:0206 Elan Microelectronics Corp.
Bus 002 Device 004: ID 8087:07dc Intel Corp.
Bus 002 Device 003: ID 0c45:6705 Microdia
Bus 002 Device 002: ID 067b:2303 Prolific Technology, Inc. PL2303 Serial Port
Bus 002 Device 001: ID 1d6b:0002 Linux Foundation 2.0 root hub
joseph@joseph-Inspiron-7537:~$
```

In my case the usb serial converter has come up as "Prolific Technology, Inc. PL2303 Serial Port". Then type in the comand: `dmesg | grep tty`

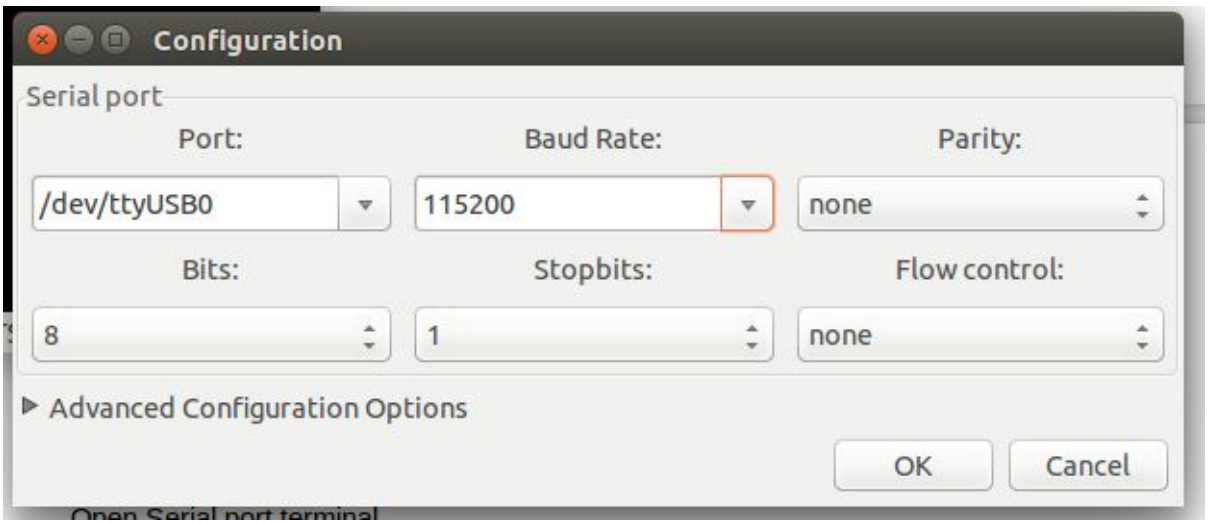
```
joseph@joseph-Inspiron-7537:~$ dmesg | grep tty
[    0.000000] console [tty0] enabled
[    4.306090] usb 2-3: pl2303 converter now attached to ttyUSB0
joseph@joseph-Inspiron-7537:~$
```

This shows which USB port the converter is connected to. As you can see this device is connected to ttyUSB0. Plug in the other end of the cable to the board like this:



Black goes to ground, Green goes to Rx, White goes to Tx. Schematics of the board can be seen here.

Now, open the Serial port terminal you downloaded earlier. Go to the top of the screen, then click Configuration -> Port. For Port, choose the port that came up on the list you saw before (e.g. /dev/ttyUSB0). For the Baud rate, choose 115200.



Click OK, and press Enter. Type in AT, press Enter. It should say OK. If it gives gibberish, try setting the baud rate to 9600, and try again. Other commands are located here. Try some of them out.
If none of the AT commands are working it is possible that the device you are using has already had its firmware overwritten. Continue on and set up the toolchain; if you like at the end you can easily rewrite the AT software to the device.

2. Setting up ESP8266 toolchain:

This section is based on https://github.com/esp8266/esp8266-wiki/wiki/Toolchain and https://github.com/pfalcon/esp-open-sdk

Navigate to /opt:

```
cd /opt
```

Download some tools needed for setup:

```
sudo apt-get install make unrar autoconf automake libtool gcc g++ gperf flex bison texinfo gawk ncurses-dev libexpat-dev python sed
```

clone the git repository:

```
git clone https://github.com/pfalcon/esp-open-sdk.git
```

navigate to the new folder:

```
cd esp-open-sdk
```

build the project:

```
make
```

Note that this "make" command can take a while (over twenty minutes).

If you are having problems with permissions (you may need sudo to run some aspects of make, but other aspects won't run properly with sudo) then try the whole thing cloning the repo somewhere like /home/$username

If you are still having problems, delete everything and go here and follow the instructions exactly.

Now go here and follow the instructions. Before doing anything in Uploading, power off the ESP8266-EVB, then hold down the on-board button and power it back on again. This will boot it in flash programming mode. Now follow the instructions to upload the demo on to the board.

From here making and uploading your own programs is straightforward. To make things easy you can in the beginning keep your projects in the same form as the at or IoT demos, with *driver*, *include*, and *user* folders, and the "main" of the system being the *user_main* and *user_config* files in the user folder. By doing this you can keep all the Makefiles from the demo, so you can focus purely on the C code of your project.

Once you have a project ready, navigate to the project folder, `make` the project, then run the following commands:

```
cd .output/eagle/debug/image
```

```
esptool -eo eagle.app.v6.out -bo eagle.app.v6.flash.bin -bs .text -bs .data -bs .rodata -bc -ec
```

```
xtensa-lx106-elf-objcopy --only-section .irom0.text -O binary
eagle.app.v6.out eagle.app.v6.irom0text.bin
```

```
cp eagle.app.v6.flash.bin ../../../../../bin/
```

```
cp eagle.app.v6.irom0text.bin ../../../../../bin/
```

```
/opt/Espressif/esptool-py/esptool.py --port /dev/ttyUSB0
write_flash 0x00000 eagle.app.v6.flash.bin 0x40000
eagle.app.v6.irom0text.bin
```

like before.