What is Beaglebone Black?

The Beaglebone Black (BB) is a development platform for embedded system developers. To make a long story short, BB is a board similar to Phytec-Wega-AM335x that is catered towards hobbyists and designers to accomplish early development stage in embedded systems projects. BB runs on the AM335x processor based on ARM Cortex-A8, which is a similar processor that the Phytec board uses. This board also uses the Linux OS as its operating system. The most commonly used Linux OS distribution that the board uses is Angstrom.

Here is the pinout of a common BB:

![Beaglebone Black Pinout Diagram](image)

**Figure 1. Beaglebone Black pinout**

How do you connect to BB?

Since this board has a running operating system, one way to connect to the board is through SSH, which is similar to the Phytec board. However, the awesome thing about this board is that we only need one cable: the serial USB cable. This cable can provide power to the board, establish network connection, and provide serial communication between the board and the host computer. So, you don’t need a separate power and Ethernet cables. This board can be accessed from most host computers with Linux, Windows, and Mac OSX operating systems. However, the new update of Mac OSX (El
Capitan) has a little bug that does not allow us to connect to the board after installing the necessary drivers. Unfortunately, we can only connect from Linux and Windows at the moment, unless you found a way to connect using Mac OSX. Before connecting to the board, please follow the following link to install the drivers for your host computer:

http://beagleboard.org/getting-started (Link 1)

You can skip the section “Update to the latest software” and go straight to connecting to the board. To connect to your BB from Linux, you can:

```
ssh root@192.168.7.2
```

There is no password associated with the board so you can just press enter when asked for password.

For Windows, you have to download an SSH client software that establishes an SSH connection to the board called Putty. You can download Putty from the following website:

http://www.putty.org/ (Link 2)

Once you installed Putty, you can enter the board IP address “Host Name” box and click “Open” as shown in the following window.

![Figure 2. Putty main prompt for SSH configuration](image)

Once you click “Open,” there will be a window that asks you about a Putty security alert, just simply say yes. Once a black window pops up, it will show a text “login as: ” Just simply type “root” and hit enter. Again, there is no password associated with this board, just press enter when the window asks for a password.
Voila! Now, you are connected to the board. The file system of BB is very similar to the Phytec board. So, simple ls, cd, echo, grep, and other Linux terminal commands work just fine with this board.

**Code Compilation**

One thing that is different about BB and the Phytec board is that code compilation is usually done inside BB, not the host computer. So, to compile your codes, you have to first transfer your codes to the board, not your binaries. For your C codes, once you are in the board, you can compile the codes using “gcc”, not “arm-linux-gnueabihf-gcc” because the board is preinstalled with its own processor-specific compiler. If you are using makefiles, make sure to check your compiler type before proceeding to compile your codes.

Other than gcc, BB also has Python and C++ compilers. So, image and signal processing can be done more easily if you want to analyze your signals using Python or C++ for your final project.

**Hardware Peripherals**

Embedded systems is not embedded systems without GPIO, PWM, ADC, and other hardware peripheral pins. As you can see from Figure 1, the board has quite a lot of hardware peripheral pins. GPIO pins are in multiple places across the board. PWM is usually referred as EHRPWMxy and ECAPWM0 where x can be 1 or 2 and y can be A or B. ADC is usually referred as AINx where x can be anywhere from 0 to 6. One thing to note, you will be using a communication protocol called Universal Asynchronous Receiver/Transmitter (UART) if you want to communicate your board with another board. For the Bluetooth module, you also have to learn about the UART protocol to communicate with it.

The following website condensed the examples of BB hardware peripheral codes into one web page:

https://press3.mcs.anl.gov/forest/hardware/beaglebone-black-software/examples-and-tutorials/ (Link 3)

Since the board and OS versions provided by the examples are older, the paths for the device files might be slightly different (not that much). For example, PWM’s enabler might be located at

“/sys/devices/bone_capemgr.7/slots”

instead of

“/sys/devices/bone_capemgr.9/slots”
You might want to check the exact paths of the device files from the terminal first before coding. You might run into unexpected “Segmentation Fault” if you don’t check them first.

If you have been Googling and using the command “find” to find the device file, but there is no luck, you can ask one of the TAs, and we can try to help you find it. Again, you have to be comfortable facing this type of issues because it will not be as easy as Googling or “asking a TA for help” if you get stuck in the real world.

Feel free to discover about more hardware peripherals. The final project is not limited to only knowing the peripherals you’ve already known from previous labs (PWM, GPIO, and ADC). You are welcome to discover other peripherals such as the I2C, SPI, the SD card module, and other awesome peripherals.

**UART on Beaglebone Black**

If you want to communicate your BB or Phytec with another device (BB to BB, BB to Phytec, etc.), I suggest you guys to use a protocol called Universal Asynchronous Receiver/Transmitter (UART) protocol. This protocol sends bytes serially using two data lines: TX and RX. Please read more about it in the following link:

https://learn.sparkfun.com/tutorials/serial-communication (Link 4)

In EE 472, what device has this kind of protocol? It’s for you to figure out. It starts with ‘b’ and ends with ‘h’. If you read this document through the end, you will know what it is.

The way to set up UART should be the same for both Phytec and BB. However, I highly recommend switching to BB for the remaining part of the project because BB has a lot of supports regarding embedded Linux.

For BB, here is how to setup UART:

http://tinkernow.com/2015/01/beaglebone-black-rs232-uart-setup/ (Link 5)

The datasheets and documents for the device that you are going to communicate with will be in the References section and Lab 4 documentations.

**Bluetooth Module and Further UART Configuration**

For the UART setup and configuration on Link 5 above, Section 3 is intended to guide you if you need to test whether the UART is working with the host computer. Instead of using the MAX232 chip, you guys have the DB9 (RS232 to USB) converter cable that looks like:
In Figure 4, you only really care about TXD, RXD, and GND pins. So, if you want to test your UART with the computer, you can just connect the USB end to the computer, connect the GND to GND of BB, RX of DB9 to TX of BB, and TX of DB9 to RX of BB. You probably will need a female-to-male wires to accomplish this. After that, you can test your UART using Minicom. Don’t forget that the baud rates for the UART protocol for both devices have to match.

If you decided to use the MAX232 chip, EE store has it, and Link 5 should be able to guide you on how to configure the chip.

For Bluetooth configuration, it’s up to you to use Phytec or BB. However, if you decided to use BB for Bluetooth, you can just connect directly the RX-BB to TX-BT and TX-BB to RX-BT (RX-BB and TX-BB are for Beaglebone Black, and RX-BT and TX-BT are for Bluetooth module).

Once you configured the Bluetooth module with the Beaglebone Black, the Bluetooth module must also then communicate with a device that supports Bluetooth. It is usually a phone with an Android OS, or a Windows PC with Bluetooth. So far, no Macs or iPhones support Bluetooth SPP, sorry. Again, let us know if there is a way to go around it. Remember, the Bluetooth module will only work as a serial interface if it is in SPP mode.
References and Good Help


UART basics: [https://learn.sparkfun.com/tutorials/serial-communication](https://learn.sparkfun.com/tutorials/serial-communication)

