Project 4A
Beaglebone Bring-Up

INTRODUCTION:

The venerable Arduino, while fun and useful, is an old and weak system:

- Limited CPU power greatly restricted the possible processing.
- Limited memory greatly restricted the type of software that could be run.
- The available libraries were minimal, and mostly device related.
- Development had to be done on another system, cross-compiled, and then up-loaded and flashed onto the Arduino.

The Beaglebone is a much more powerful platform. It runs a full-blown Linux system:

- it hosts remote terminal access via both USB and WIFI.
- it has a package manager that can be used to find and download software from the internet.
- it can host its own complier and development tools.

Other than size and power, the biggest difference between an Beaglebone and a desktop computer is that the Beaglebone has a great many programmed I/O pins that can be used to interact with a wide range of sensors and actuators. This difference makes it a very reasonable platform for both Embedded Systems and Internet of Things projects.

The Beaglebone comes in several models. You will be working with the Beaglebone Green Wireless model. Obtaining this model is important, since other Beaglebone models do not have the wireless networking capability required for the project you will be performing. You will be provided with detailed instructions on obtaining the right model of Beaglebone for this project.

Currently the version of software commonly installed on Beaglebones is not the best version for this project. In the section describing the project below, we will describe how you should update this software for the project.

RELATION TO READING AND LECTURES:

None. All of this work is preliminary to embedded system software development.

PROJECT OBJECTIVES:

- ensure students have a working Beaglebone.
- ensure that students can log into their Beaglebone via both USB and WIFI.
- ensure that students have the ability to transfer files to/from their Beaglebone.
- ensure students have a working Beaglebone development environment.
- ensure students have the ability to install new packages on their Beaglebone.

This should be a simple matter of following simple steps. But if you do not have communication with a working Beaglebone development environment, you will be unable to complete the remaining embedded systems projects.

DELIVERABLES:

A single compressed tarball (`.tar.gz`) containing:

- a selfie (`my_beaglebone.jpg`) of you holding your assembled Beaglebone (showing the front of the board).
- a screen shot (`connected.png`) from a USB terminal command session connected to your system, showing that you gave it a name and configured network access.
- a screen shot (`copy.png`) from a local session in which you copy a hello world program from your laptop/desktop to your Beaglebone.
- a screen shot (`build_run.png`) from a WIFI SSH session in which you build and run a trivial (hello world) program.
- a screen shot (`git.png`) from a WIFI SSH session in which you install *git* and clone a repository.
- a `Makefile` to build the tarball.
- a README file containing the serial number of your Beaglebone and the WiFi (unless you are using a hard-wired ethernet connnection) MAC address, and a description of the above files.

Most phones can produce `.jpg` images and Linux screen shots are usually `.png` but any of these images can be submitted in `.jpg`, `.png` or `.gif` format.

PROJECT DESCRIPTION:

There are a number of useful tutorials for working with the Beaglebone, which are referenced below in the various steps of the project. We strongly recommend that you read them before performing each step.

1. Flash the Beaglebone with an updated version of the system software. Your TA will help you perform this operation, so consult your TA on this step. Reflashing the software has proven particularly important if you are using a Mac computer to interact with it. This quarter we will be using version 9.5 Debian IoT for the Beaglebones.
2. Read Introduction to the Beaglebone. This tutorial will provide information on configuring your BeagleBone's hardware and software. Follow its instructions on how to assemble it, power it up and connect it (via USB) to your personal notebook or desktop. Establish a USB terminal login session from your notebook/desktop to your Beaglebone. If you run MacOS and have not, for some reason, been able to reflash your Beaglebone, as described above, the Screen program might allow you to access the Beaglebone via USB.
3. Read Linux on the Beaglebone. This provides some basic information on using Linux on the Beaglebone.
4. Read WiFi, SSH, and SFTP on the Beaglebone. Follow the instructions in this tutorial to create a password for your Beaglebone and enable WiFi access to it. Change the name of your Beaglebone to *studentID*.lasr.cs.ucla.edu (instructions on how to do so can be found here). Then run the *ifconfig* command to display the IP and MAC addresses, and take a screen shot.
5. Read Beaglebone Tutorial: GPIO, Interrupt, Analog and PWM. This tutorial will provide information on how to use the Grove Base Cape and sensors with your Beaglebone.
6. We have also included some guidelines for working with the Beaglebone on this particular project. These guidelines are specific to our project 4 at UCLA, and may not be helpful to others using Beaglebones, but they should help you work through some common problems we have observed in using these devices.
7. Log in to your Beaglebone (over WIFI) via *ssh* (or *putty* or other equivalent tool).
8. On your notebook/laptop, write a trivial C "Hello World" program. Follow the instructions in WiFi, SSH, and SFTP on the Beaglebone to use *sftp* to copy that program from your notebook/desktop to your Beaglebone. Alternately, figure out how to use *scp(1)* to transfer your program to the Beaglebone. If your notebook/desktop runs Windows, use PSCP to transfer the program. Take a screen shot (on your desktop/laptop) of the successful copy command.
9. Use *ssh(1)* to connect (via WIFI) to your Beaglebone, and compile and run a simple C "Hello world." program. Take a screen shot of the compilation and execution.
10. Download the *git* version control system onto your Beaglebone and use it to create a code repository on that Beaglebone. Use *apt-get* to get the *git* package onto the Beaglebone. You can find instructions on using *apt-get* at the Debian apt-get description page. Once you have installed *git*, use it to `clone` a local

copy of a repository (e.g. your own github repo). Take a screen shot of the clone command and directory listing.

SUBMISSION:

Your tarball should have a name of the form `lab4a-`*studentID*`.tar.gz`. Your ID should be in the XXXXXXXXX format, not the XXX-XXX-XXX format. You are strongly advised to sanity check your submission with this test script. Much of this project will be auto-graded. Passing the sanity check does not guarantee a high score, but submissions that do not pass the sanity check should expect to receive very low scores!

Your **README** file must include lines of the form:

NAME: *your name*
EMAIL: *your email*
ID: *your student ID*

You may add files not specified in this page into the tarball for your submission, if you feel they are helpful. If you do so, be sure to mention each such file by name in your README file. Also be sure they are properly handled during the dist and clean operations in your Makefile.

GRADING:

Points for this project will be awarded:

Value	Feature
5%	untars expected contents
5%	obtain and assemble your own Beaglebone
10%	successfully establish USB terminal session
20%	successfully establish *ssh* session
20%	successfully copy a file via scp/sftp protocol
20%	successful compilation and execution
20%	successful *git* install and `clone`