**Memo**

**Senior Design**

ENG EC 463

To: Professor Pisano

From: Brian Kane, Katherine Murphy, Ben Paolillo, Jiwon Song, Todd Sukolsky

Team: 14: Bikeputer (Re.Cycle)

Date: 11/15/2012

Subject: First Deliverable Testing Plan

1. For the first deliverable, we plan to demonstrate the functionality of an LCD Touch Screen with an Arduino, the functionality of the Beaglebone and the power management circuit (on a demo board). For the LCD screen, we plan to show that we can navigate between nine different screens and maintain data corresponding to the current screen position; additionally, we are able to calibrate the screen whenever the user needs to do so. This component is significant because the screen is the main user interface and will eventually display the data supplied by the user, GPS, and various sensors.

The BeagleBone is the main processing unit on the board and is responsible for communication with all of the modules. This first deliverable requires root access to the processor, the ability to connect and stream data via USB and transfer UART strings via wire.

The Power Management system is described as the control block of all power. The system should bring in a voltage from a battery, in this case a LiPo 2-cell, and run an ADC conversion. The output of this ADC converter will trigger the main regulator to provide power to the rest of the board or go send the uC into a sleep state along with the rest of the board. The regulator should be shut off when the voltage is below 7.43 volts for this battery.

The Touch Screen portion of the deliverable signifies the main user interface ability. The screen will eventually display data supplied by the user, GPS location, heart rate and other cycling statistics. Without this component the system has no user interaction. The BeagleBone functionality, being the MPU, is the most critical part of the system and showing root connectivity, pin manipulation and data streaming is key. The feature that seperates our project from competing technologies is our use of supplemental power. The power management system is essential for this to conserve power in a low-voltage case and to simply provide power to the rest of the system.

2.0 LCD Screen:

2.1 Measureable Criteria:

For the LCD screen we will demonstrate navigation between the nine possible screens. We will demonstrate calibration by running a finger around the perimeter; an analog to digital conversion of the voltage signal from the touchscreen will result in the coordinates best corresponding to the position of the users touch. In order to maintain an accurate measurement of these coordinates, we need to store the minimum and maximum values of the results of these conversions as parameters for both the x and y variables. During calibration, if the ADC results in a value smaller than the currently stored minimum or greater than the currently stored maximum, the new value will be tentatively stored as a replacement for the calibration parameters. The user will then be prompted on whether to save these new settings, or restore the calibration to its state prior to calibration.

* 1. Equipment Setup/Measurement Plan/Data Acquisition:

The touchscreen LCD are set in a breadboard with a soldered on header, then wired to the Arduino board. The board includes a boot loader, which programs the Arduino with our most updated module, generating the graphics and user interface environment. For accuracy feedback, the board is connected to the USB port of a computer, then the Serial Monitor is executed to show UART sent from the Arduino. This data, in this demonstration, consists of the coordinates of the users touch (optionally) as well as the configuration parameters, when in configuration mode.

3.0 Beaglebone and Power Management:

3.1 Measureable Criteria:

The measurable criteria for the bone and power management system is the ability to

gain root access to the bone, stream live data through it’s USB and UART ports and toggle a GPIO pin high or low depending on the input (as well as blink an LED). The power management system is measurable with the voltage calculated by the ATMEGA168A, to within .075 volts, and by the ability to turn off the main regulators to the development board when the low battery threshold is met (7.43 volts for normal case. During test it will be changed to the current battery level. ADC reading should be < voltage\*1000/8.78). See “Assessment” section for further details.

3.2 Equipment Setup:

Beaglebone is connected to a Linux, Windows or Mac environment via USB and Ethernet.

The USB allows connectivity on the local subnet and the Ethernet is used to stream the shared wireless or LAN connection to the bone. In Linux “ssh” command can be implemented to connect to the board, always at local IP 192.168.7.2. In Windows a client is used such as “Putty”. Attached to the bone, on the UART2 pins, is a development board Todd previously produced for a different project. The board contains the power management system minus the backup watch battery for the Real-Time Clock. Also attached is a mini USB -> USB cable that is used to stream NMEA strings from the onboard GPS. This GPS is a test unit for the NMEA string parser (currently under development in Python). Also attached to the Bone is a small circuit made to toggle an LED on and off, and with a python script blink. The circuit consists of one led and a 220 ohm resistor. The anode of the LED is tied to one end of the resister, the other end tied to ground. The second terminal of the resister is tied to the GPIO pin that will output 3.3V.

The power management system is implemented on the development board and a schematic of the module on the board is provided in this packet. It uses an ATMEGA168A, 8-bit, microcontroller to take ADC readings, print UART strings, and communicate with other modules on the board. The uart lines are fed to the header pins 0 -> AVR\_RXD and 1 -> AVR\_TXD. The ground pin for the connection is located at pin 17 -> GND.

3.3 Measurement Plan and Data Collection:

Step one is to root connect to the bone using the command “ssh [root@192.168.7.2](mailto:root@192.168.7.2)”. The password for the root user is null. Next we will execute a test C++ script to show functionality of the Angstrom Software Distribution. The program should output a string to the terminal “Hello from Beagle Bone! So nice to speak to you today.” This is done through command line text editors and installed compilers. Proceeding the C++ execution will be proof of live stream through the onboard USB cable. The stream will be live NMEA strings from the GPS on the development board. To execute, the “minicom” command will be used which is native to the linux operating system. The baud rate for the GPS is 9600 so the command for this is “minicom –b 9600 –D /dev/ttyUSB0”. The “-D /dev/ttyUSB0” string portion tells minicom to listen over the USB port for data. The output of this command should be strings of the format “$GPxxx x,x,x,x,x,x,x,x,x,x,x,x”. There are numerous types of strings. “$GPGGA” strings contain location (longitude, latitude) data. These strings are blank in this case because of the null location. After this demonstration the GPIO functionality of the bone will be demonstrated by toggling an LED on/off given input in the terminal. After a toggle is shown, a python script will be executed to blink the LED at 1Hz.

The last test of the bone will implement it’s UART2 and UART1 pins. This is also a the test of the battery management system. First a python script will be run setting up the correct UART ports. This will be viewed in the terminal, then executed. Following this a three line connection will be made to the development board for UART\_TXD, UART\_RXD, and GND. Once plugged in we will stream UART data, using minicom once again, using the command “minicom –b 115200 –D /dev/ttyOx” where “x” is either 1 or 2, depending on which UART pins we are going to test. The baud rate for the UART data is 115200. From this we should see battery voltages streaming on the screen. After a successful transmission is completed, the minimum threshold voltage for the ADC conversion will be changed to show the functionality of the regulator enable lines. After changing, the microcontroller should take an ADC reading, find a voltage lower than the prescribed minimum, and shut off the main regulator on the board. This will trigger the onboard LED’s on the development board to shut off.

3.4 Assessment:

The first test of the bone will be judged on the simple process of gaining root access to the board and running the C++ program with the correct output to the terminal. The USB stream test will be considered a success if the NMEA strings appear in the minicom terminal. If the strings are not present, then streaming is not taking place and there is an internal error. The final test is deemed successful based on two collections: The first requirement is the ATMEGA168A needs to transmit the UART strings to the bone correctly on both UART1 and UART2 ports. The second requirement is the ADC reading from the microcontroller needs to be within .075 volts of the actual voltage measured by a multimeter. With a nominal voltage of 7.4V, an error of .075 volts represents ~1% error worst case.