**Battery Health Management (BHM) on BeagleBone Black**

The primary objective for the internship was to develop BHM software that used an Unscented Kalman Filter (UKF) routine to estimate battery voltage and charge in the presence of occasional measurement error. In the UAV lab of the Safety-Avionics Systems branch, BHM software exists for fixed wing aircraft and on the PC104 DOS computer, but does not currently exist for multi-rotor vehicles nor on the Beagle Bone Black Linux computer. The challenge has been to take the initial hardware and efforts of my mentor Dr. Ken Eure in order to further a BHM software solution on the Beagle Bone Black computer. BHM To that end, subprojects were completed along the way as an aid in developing the final solution.

A great resource that Dr. Eure provided was a book called “Exploring BeagleBone: Tools and Techniques for Building with Embedded Linux” by Derek Molloy. This was an extremely key resource that I referred to on a daily basis. In addition, Dr. Eure provided me with an example script written in the Matlab environment that implemented a UKF routine. I also frequently consulted papers written by Simon J. Julier and Jeffrey K. Uhlmann whom appear to be the authors of UKF technique.

In a notable completed subproject, a scalar UKF routine estimated the voltage across a capacitor after being charged and then discharged. This made use of basic RC circuit model equations. These equations were used in the scalar UKF and were programmed into an Arduino that monitored and predicted voltage as well as the remaining time to discharge to a specific voltage. While the Arduino was connected to a PC, the voltage and time data was sent to an Excel spreadsheet and displayed on a chart in real-time using the PLX-DAQ VBA script.

The next notable achievement was the Matlab simulation. I was able to take battery current from a figure from the paper “Battery Charge Depletion Prediction on an Electric Aircraft” as input to the dynamic equations and UKF to replicate output voltage, state of charge, and End of Discharge (EOD) predicted time. I was able reproduce similar output curves shown in the paper.

The final milestone was the production of the following items: A vector UKF routine was written into a Matlab function. This Matlab function was then automatically translated into C code. A main function was written in C on the Beagle Bone Black in order to read in voltages and battery current from an analog-to-digital converter (ADC) board and uses these as inputs to the UKF function. The UKF function then returns an improved estimate of the state of charge and voltage of the battery. The program writes this data (several times a second) to a file stored onto an SD card that can be removed for post-processing. The final system contains a research test battery connected to an ADC board connected to a Linduino (like an Arduino) which is connected to the Beagle Bone Black using a USB Serial connection.