Bryan Friestad
Last edited 5/5/20

OBC Subsystem - Next Steps

Pivotal Tasks

- Interrupt based serial communications (I2C and UART)
    - Benefit: Using interrupts over polling mode communications can improve response time and ensure messages aren't missed.
    - Method: There are HAL (hardware abstraction layer) library functions already available that perform interrupt based I2C and UART communications. The interrupts must simply be set up.
- Code to handle new ground station commands as they are developed
    - Method: using the same format already existing in Main.c, you can add additional functionality to handle new custom CySat Packet Protocol commands. Some suggested commands include, but are not limited to:
        - Alter beacon (text and period)
        - Get battery capacity
        - Get attitude telemetry data
        - Perform health checks
- Implementing code to complete Mock Launch and Mock Mission
    - Info: The general progress of the mission is described in the Concept of Operations document, which the overall team leader of CySat manages. This document walks through the main phases of the satellite's life: 1) Startup Mode, 2) Diagnostic Mode, 3) Main Operating Mode, and 4) End of Life. The "Mock Launch" includes phases one and two, while the "Mock Mission" handles phase 3 and optionally phase 4.
    - Method: Using the "Mock" Mock launch plans found in the doc folder of the repository, determine a list of tasks that need to be added to complete the full Mock Launch. This "Mock" Mock launch includes steps and alterations due to our switch to remote work (because of COVID-19). Complete those additional steps (mostly just communicating to the components of the satellite instead of just printing to the screen that you would) and then attempt the full Mock Launch, after that, you can begin to put together a plan for the Mock Mission, which should look similar.

Bryan Friestad
Last edited 5/5/20

Stretch Goals

- Real time operating system
    - Benefit: In order to ensure longevity of the satellite, and provide graceful degradation, a RTOS should be implemented
    - Method: There already exists a FreeRTOS middleware in the OBC SDK, which can be utilized to make tasks with a requested period.
- In-flight reprogrammability
    - Benefit: The satellite may not have all possible software written by the time the launch is arranged.Being able to reprogram in flight would allow you to add new functionality and fix unexpected bugs
    - Method: This requires use of HAL library access to the SD card on board. Getting these working and testing them would be the first step. Then, I would work on making a first stage of the application, which loads the second stage from a particular section of the SD card. Finally, keeping track of which section of the SD card to boot from would essentially give you a bootloader. Other options that don’t entail complete bootloaders could be feasible as well.