Winter Progress Report for RockSat-X Payload - Hephaestus
Group 26

Amber Horvath

CS462 - Winter 2017

March 23, 2017

1 Introduction

The Hephaestus Payload is a rocketry payload that will fly onboard the 2016-2017 RockSat-X rocket. The rocket will be launched from the Wallops Flight Facility filled with student-made payloads from various institutions. The Hephaestus payload will consist of a deployable arm and a video camera. The arm shall extend and move to a series of pre-placed sensors and make contact with the sensors. The arm will then contract and retract back into the rocket. The video camera shall record the arm's movement. Data about the flight, such as error codes, shall be sent via a telemetry port and written onto a microSD card. The Hephaestus mission will be Oregon State University's first space mission and will prove not only our ability to develop a space-ready payload, but also the viability of construction in space using a robotic arm.

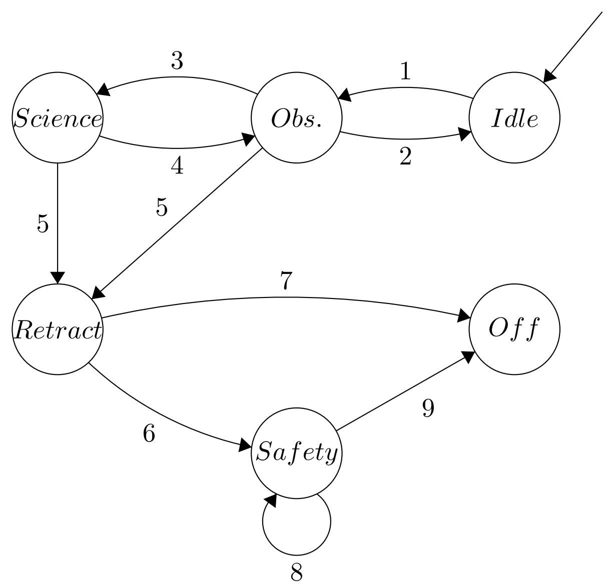
2 Individual Pieces

The individual pieces of the project I am in charge of for the project include Emergency Retraction, Modes of Operation, and Touch Sensor Viability. I also took on the Data Storage task by the request of the Electrical Engineering team.

2.1 Emergency Retraction

In the case of the arm getting caught in a bad state, we shall retract the arm back into a safe state This will be accomplished by testing for a bad state. We will retract the arm and test whether or not we?re in our calibrated normal position If we are not, we are in a bad state and need to execute the emergency arm retraction. We will then disable each motor except the one attached to the base of the payload. Once each motor has been disabled, we will retract the base of the payload.

2.2 Modes of Operation



The Modes of Operation detail the states the program will be in during the course of the flight, as seen in the included state diagram. The transitions are as such:

1. **Apogee is reached.** The experiment begins; the camera is powered on; and the on-board computer is on. A camera sweep is performed.
2. **Error: Return to Idle.** If an error occurs, we shall transition back to Idle mode.
3. **Payload Assembly and Camera have been deployed.** The software shall enter Science mode once the payload assembly has been deployed and the camera sweep has completed.
4. **Error: Return to Observation.** If an error occurs, we transition back to Observation mode.
5. **Timer switch to end apogee period.** Experiment time has ended, proceed to Retract mode to prepare for descent.

6. **Error: Proceed to Safety.** If the arm fails to retract, proceed to Safety mode.
7. **Power off.** Arm is retracted, payload is retracted, computer is powered off, camera is powered off. Payload is ready for descent.
8. **Cycle in Safety.** Continually attempt to retract the arm and payload and power off the computer and camera.
9. **Power off.** Payload is now ready for descent.

2.3 Touch Sensor Viability

Within the body of the payload, two touch sensors will be placed at predetermined locations. The arm shall make contact with the touch sensor and press it. Upon being pressed, the touch sensor will go high and send a signal over telemetry and be written to an SD card. The arm will then recalibrate before moving to the next sensor. If no sensors remain, the arm will retract and await shutdown.

3 Progress and Problems

3.1 Emergency Retraction

Due to the testing of emergency retraction being dependent upon the payload being assembled and the motors being attached to the payload, this has been a lower priority task. Pseudocode was written that details how this component of the project interacts with the other components of the program, but the code has not yet been completed. Code has been written for the interrupt subroutine that will notify the on-board computer to halt the current process and power off the specified motors.

3.2 Modes of Operation

The Modes of Operation requirement was technically completed last term, as it was essentially a state diagram that detailed how the program would operate throughout the flight. A state was added during the middle of the term to account for the time when the payload is being retracted from its extended state, but no other changes were made. As more code is being written, the modes of operation requirement will entail me ensuring that the design of the program upholds the design we originally developed last term.

"sdcardtest.c", which is the current file being used to test the core functionality our system requires. Our system must be able to open a file, write or append some data to the opened file, and close the file. These functionalities map to the API calls specified in "ff.c", the currently open file in Figure 1. The main focus of my effort has been on getting these API calls to work as we intend for them to. Currently, calls to the "fmount" function, which mounts the filesystem onto the micro controller, and the "fopen" function, which creates a file pointer, work. However, we have not been able to get writing to a file to work. This has been a large blocker for progress as the errors thrown are difficult to follow and, as evident within Figure 1 there is a large amount of code that must be sifted through.