Aydin O'Leary Spring '21 ThinSat Work

Mission description and constraints

The main goal of the ThinSat mission (at least the part that I worked on) is to fire four dipoles out of a satellite at a specific location over Massachusetts (one per day for four days). Once this is done, the satellite should deorbit. The satellite completes about 16 full orbits per day.

The dipoles are activated through a burn wire, so the only thing that the onboard computer needs to do to send a dipole is to send a high signal out of a specific port. The challenge comes with sending that signal at the right time and place above the Earth's surface.

The onboard computer figures out when to shoot dipoles and deorbit by comparing its time to a predetermined list of events generated before launch and stored on the satellite. The real-time clock (RTC) that we are using, the DS3231, is pretty precise (drifting a maximum of 15 seconds in 2 months) but not quite precise enough for our applications; to remedy this, we are using an odometry algorithm based on detecting sunrises and sunsets (method 2 in these slides).

The satellite will spend an unknown time on the ground before being launched from the ISS; this creates some issues to be discussed in the next section.

Things still-to-do (as of mid May 2021)

- There is tension between the fact that our location-finding algorithm relies on a precise lookup table and the fact that we don't know when we'll launch. This still needs to be resolved, as far as I know.
- Once that's figured out, the task breakdown and data flow can be finalized and the tasks can be programmed.
- I recommend testing tasks and software cohesion with the Arduino FreeRTOS library first, then transferring to the more complicated AVR toolchain once you're more comfortable with FreeRTOS conceptually.

Design Decisions

I picked an RTOS primarily because they see a lot of use in aerospace applications and I wanted some experience with one; what we need the satellite to do is just barely not complex enough to not *need* an RTOS. That being said, it is still very well-suited for the task (ha) at hand.

Useful Resources
RTC datasheet
Odometry Algorithm Slides
Microcontroller Datasheet
Arduino FreeRTOS Library
FreeRTOS AVR port
Summer '19 Github
Summer '19 Google Drive
Kalman Filtering Tutorial
FreeRTOS Kernel Guide
FreeRTOS Documentation
Youtube Guide to RTOS concepts

Long Links (in order)
https://datasheets.maximintegrated.com/en/ds/DS3231.pdf
https://docs.google.com/presentation/d/1XInfQLXaoFCPvZmZP7s5ru3Ck-q_vFBR/edit#slide=id.p1
http://ww1.microchip.com/downloads/en/DeviceDoc/Atmel-7766-8-bit-AVR-ATmega16U4-32U4_Datasheet.pdf
https://www.arduino.cc/reference/en/libraries/freertos/
https://www.freertos.org/a00098.html
https://github.com/dharaspatel/NASA-ThinSat/wiki
https://drive.google.com/drive/folders/1ahHINVxbvm4W4_QKbZe77RymcYQGK34d
https://www.kalmanfilter.net/default.aspx
https://www.freertos.org/features.html
https://www.youtube.com/watch?v=F321087yYy4

Various Emails
christopher.lee@olin.edu (Chris Lee, the Olin professor in charge of this research)
aoleary@olin.edu (myself, if you have any questions about what I worked on)
ericjs65@gmail.com (Eric Steinberg, worked on programming the sun-event detection algorithm)
dpatel1@olin.edu (Dhara Patel, worked on the project during Summer 2019)
john.bacon-1@nasa.gov (Jack Bacon, our NASA contact for this project)

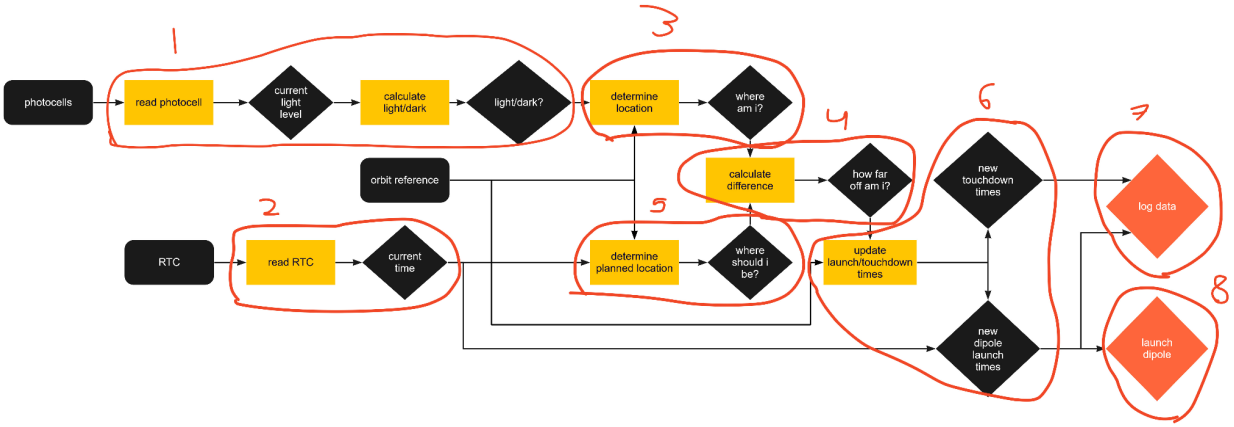
Rough timeline of work I did

- Schematic investigation / problem understanding (early-mid February)
- RTOS / arduino / C investigation / learning (mid February to mid March)
- Data flow design (mid March to mid April)
- Tracking down previous project resources (mid-late April)
- Trying to figure out answers to the Main Question (early May)
- Writing this up/collecting resources (mid May)

Note: towards the end of April my responsibilities for other jobs/classes increased drastically so I did not work on this as much as I'd have liked

Some work I did

My final revision of the data flow diagram and task division before I ran into the Main Problem:



A video showing a functioning RTOS blinker on an ATMega32u4 board:
https://youtu.be/FUXdBo1TIr8