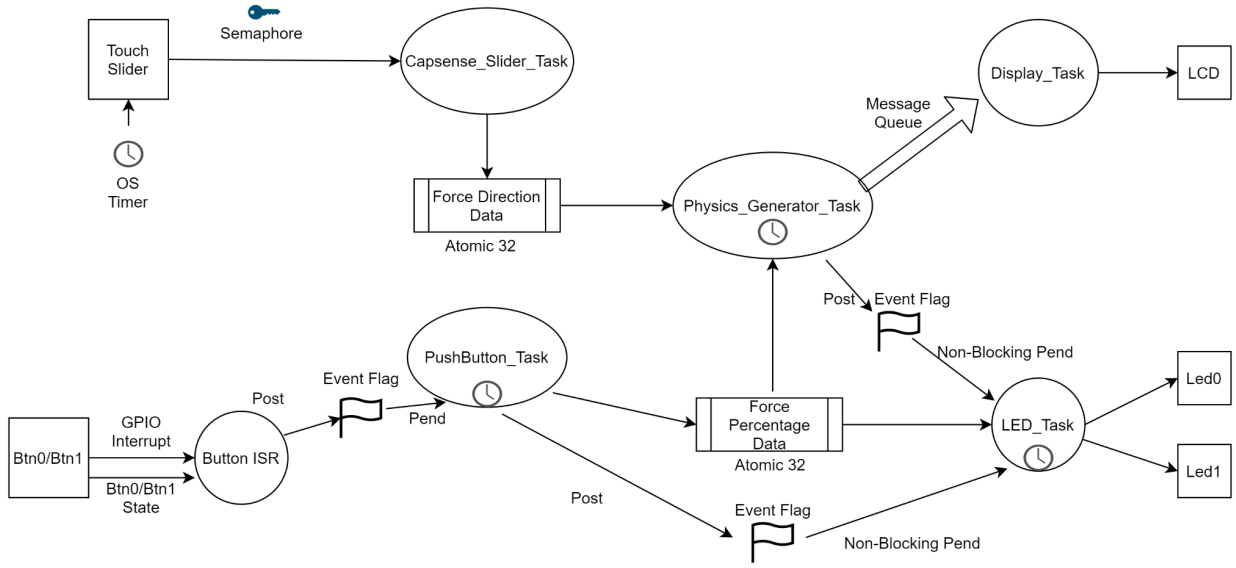
Justin Thwaites
ECEN 3753
Final Project

Week 2 Report

I am currently 38% complete with the project.

Updated Task Diagram



Unit Tests and Functional tests:

As of now, of the 12 units, 4 of them are passing. These unit tests are currently only passing because they are testing many edge and default cases and the functions currently just outputting default values. This is why 33.3% of the values are passing. The unit tests are as follows:

1. Button 0 reads to ensure a button0 pressed event and then a button0 released event is sent into the pushbutton task the button recording function of PushButton_task will return that pushbutton 0 is pressed. This is cut by the Button ISR and before the writing of force percentage to the shared Force Percentage Data.
2. Button 1 reads to ensure a button1 pressed event and then a button1 released event is sent into the pushbutton task the button recording function of PushButton_task will return that pushbutton 1 is pressed. This is cut by the Button ISR and before the writing of force percentage to the shared Force Percentage Data.
3. Button error read to insure a button1 pressed event and then a button0 pressed event and then a button1 released event are sent into the pushbutton task the button recording function of PushButton_task will return that no push buttons are pressed. This is cut by

the Button ISR and before the writing of force percentage to the shared Force Percentage Data.

4. Button error read to ensure a button0 pressed event and then a button1 pressed event and then a button0 released event are sent into the pushbutton task the button recording function of PushButton_task will return that no push buttons are pressed. This is cut by the Button ISR and before the writing of force percentage to the shared Force Percentage Data.
5. Slider Reading pressed test, ensures that given values of the normalized slider positions function with all below the pressed threshold, it will return the lowest slider position. This unit test is cut by the Touch slider semaphore and the input to the force direction data.
6. Slider Reading not pressed, ensures that given values of the normalized slider positions function with all values above the pressed threshold, it will return that no slider is pressed. This unit test is cut by the Touch slider semaphore and the input to the force direction data.
7. Slider Reading out of bounds error, ensures that given values of the normalized slider positions are all 0 or all above the given threshold, it will return an error code that can be caught to prevent buggy performance. This unit test is cut by the Touch slider semaphore and the input to the force direction data.
8. LED1 Control, this unit test looks at the handling logic and makes sure if a stub provides the Event Flag for LED1 to be turned on, the handling logic will turn on LED1. This Unit test is cut Around the LED Task by the Event Flag data structure and the BSP LED functions.
9. LED0 Control, this unit test looks at the handling logic and makes sure if a stub provides the Force percentage at 100, the logic will return that LED0 should be high. This Unit test is cut Around the LED Task by the force percentage data structure and the BSP LED functions.
10. LED0 Control, this unit test looks at the handling logic and makes sure if a stub provides the Force percentage at 50%, the logic will return that LED0 should be low. This Unit test is cut Around the LED Task by the force percentage data structure and the BSP LED functions.
11. LED0 Control, this unit test looks at the handling logic and makes sure if a stub provides the Force percentage at 0, the logic will return that LED0 should be low. This Unit test is cut Around the LED Task by the force percentage data structure and the BSP LED functions.
12. Physics_generator task, this unit test tests the physics engine with given global inputs as well as the force direction and force percentage and makes sure that the correct force is computed as a result of the global constants as well as the force direction and force percentage.

Testing Status Summary

Of all the unit tests 1-12, 8 are currently at status “NotRun”. These tests are running and passing because the default values are being returned.

Project Summary Statement

This week I designed the layout of my project and set up the project code in Simplicity Studio. Additionally, I implemented the unit testing suite and implemented all of the unit tests.

I have completed 38% (6.5/17hrs) of the project scope by this second week. This has taken me 50% (8.5/17hrs) of the time. I may want to adjust my amount of hours by 1.25x this will keep me on track for the rest of the project. I will be accounting for this because all of the tasks have been taking a slight bit more time on average.

Project Scope

- Task Diagram
    - Complete
    - Takes 1 hour
    - Spent 1 hour
- Unit Test generating
    - Complete
    - Takes 0.5 hours
    - Spent 1 hour
- Project Scoping
    - Complete
    - Takes 1 hours
    - Spent 0.5 hours
- Risks
    - Complete
    - Takes 0.5 hours
    - Spent 0.5 hours
- Setting up the Project In simplicity with skeleton tasks and stub functions
    - Complete
    - Takes 1 hour
    - Spent 1.5 hours

This part of the project seems to take quite a while to set up, however since this is set up it does provide a much better sense of planning of the project. Much of the implementation should go much faster now that the project can compile and the skeleton of the project has been built out. I think it took extra time because I needed more time planning.

- Data Structure Definitions
    - Complete
    - Takes 1 hour
    - Spent 1 hour

This part of the project was important because it, along with the task of setting up the project allowed for a greater preparedness and planning of the project. I feel that it will set me up much better in the long run.

- Unit testing implementation of all unit tests
    - Complete
    - Takes 2 hours
    - Spent 2.5 hours

My unit testing implementation provided a bit more granularity in my project because it made me think heavily bout the inputs and outputs of the logic functions so that they have a clean interface to interact with the tasks as well as the unit testing suite.

- Pushbutton Task
    - Yet to complete
    - Takes 1.5 hours
- Slider Task implementation
    - Yet to complete
    - Takes 1 hour
- Physics Engine Task
    - Yet to complete
    - Takes 2.5 hours
- Display Task
    - Yet to complete
    - Takes 2 hours
- LED Task
    - Yet to complete
    - Takes 1 hours
- Functional Tests
    - Yet to complete
    - Takes 1 hour
- Integrating project subsystems together for complete functionality
    - Yet to complete
    - Takes 1 hour
- Final quality inspection
    - Yet to complete
    - Takes 0.5 hours

Total Complete/ Total Required = 6.5/17 hours 38%

Risks

See the attached spreadsheet for my filled out Risks register.