Unit testing:

Cutting point-After button task reads from the fifo:

1. Test that a fifo read from an empty fifo returns false.-Pass
2. Test that a fifo read from a fifo returns the correct value.-Pass
3. Test that the rail gun charge gets calculated correctly.-Pass

Cutting point-After physics task updates values periodically.

1. If capsense force is not equal to 0, check if the horizontal position of the platform is correct.-Pass
2. For satchels, check that the proper number of satchels are in the air at all times.-Pass
3. Check that the vertical position of the satchels was updated correctly.-Pass
4. If something collides with a wall, check that its velocity changed sign.-Pass
5. If the shield was activated during the physics update, check if any satchels in range were properly destroyed.-Fail
6. When a satchel reaches the ground, check that the satchel’s x-position is on target. Pass

Cutting point-after the display/LED task updates periodically

1. Check that the left LED turns on and off at the correct duty cycle.-Fail
2. After the castle evacuation time expires, check the left LED is constantly on.-Fail
3. Check that the Pulse width for the right LED is relatively equal to the current force magnitude.-Fail

Functional Tests:

1. Press the left/right side of the CAPSENSE slider and check if the platform moves to the left/right. Pass
2. The outer quarters should cause the right LED to stay constantly lit while the inner quarters should have the LED blink on and off evenly. -Pass
3. Pressing two sections of the CAPSENSE slider on opposite sides combines their effective force, potentially canceling each other out. -Pass
4. If BTN1 is pressed, a force field should appear on the display and destroy any satchels in range.
5. Pressing and holding BTN0 then releasing should fire a projectile with speed increasing as you hold the button, maxing out after 5 seconds. -NotRun
6. If the generator does not have enough power for the shield or the railgun nothing should happen. -NotRun
7. Hitting the Castle wall should break a piece of the foundation. -NotRun
8. After foundation 3 hits, the left LED should start blinking. -NotRun
9. Either hitting the castle 5 times or 2 minutes after the evacuation sequence begins should take the user to a victory screen. -NotRun
10. The platform hitting a satchel or railgun shot, a wall too fast, or running out of energy while the castle is not in an escape function should take the user to a game over screen. Pass

Project Summary:

This week, I finalized my functional tests and implemented the button reading task and railgun physics and display for my game.

Summary Effort/Estimate:

I have 73.3% of my current work (33 estimated hours out of 45 total) in 51.1% of the budgeted time (23 hours spent out of 45 hour estimate) For the work that has been completed, I took 0.697x (13.5 actual hours/15 estimated hours) as much time as I estimated.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Task | Estimated Effort | %Estimate | Actual Effort | Status |
| Task Diagram | 3 hours | 6.66 | 2.75 hours | Complete |
| **Unit Testing** | **8 hours** | **17.77** | **3.5 hours** | **Complete** |
| Platform Phys | 5 hours | 11.11 | 5 hours | Complete |
| Satchel Phys | 5 hours | 11.11 | 4.75 hours | Complete |
| **Railgun Phys** | **5 hours** | **11.11** | **6 hours** | **Complete** |
| Display Task | 12 hours | 26.66 | 7 hours | In progress |
| **Button Task** | **5 hours** | **11.11** | **1.5 hours** | **In progress** |
| CAPSENSE Task | 2 hours | 4.44 | 1 hour | Complete |
| Total Complete | 33 hours | 73.33 | 23 hours | 45 hours budgeted |

* Unit testing
  + Although we are not actually implementing the unit tests as I had initially expected, I think that having the unit tests as a general baseline for what I expect the code to do has been helpful for debugging issues as I work through the project. In addition, the functional tests I’ve written help demonstrate how large scale items of my game should be working so that someone else is able to see how my game is supposed to function on their own.
* Railgun Task
  + This implementation was a little bit tougher than I had initially expected. The main issue I was running into was using an event flag between my button read task and my physics task to tell the physics task when to start tracking a railgun shot. I was struggling to find a way to have the event flag pend in the physics task run in such a way that it would not block the physics task for a long period of time, since it has to run fairly frequently for the game to function properly. I was trying to run the Flag Pend function as a non-blocking operation but kept getting error from my code with this setup. I eventually decided to have the Pend block the task, but timeout after a couple of ticks so the code still mostly runs at a periodic rate. For the final week I will need to do some testing to make sure that this stalling doesn’t end up causing my game to operate in a weird way.