Task Diagram

Test Plan and Results

“Cutting Points” for Testing

1. ISR button 1 test
   a. Test to see is button one interrupt is interrupting when button 1 is being pressed and sends a flag.
2. ISR butt0n 0 test
   a. Test to see that the interrupt is happening on both the rising and falling edge. When it's on the rising edge it will send a charge start flag and when it interrupts on the falling edge it will send a charge end flag.
3. Railgun Task is updating appropriately
   a. When receiving a flag from button 0 it will start and stop a timer. It will start on the start charge flag and end on the end charge flag. The total time charged is stored in a structure that is mutex and used in physics task later.
4. Movement Task is updating properly
5. Physics Task is calculating correctly
6. LED output is correct
7. LCD output is correct

Unit Tests

1. Button 1 Test:
   a. The test for button one will be comprised of 4 sections.
      i. Button 1 ISR test to see if the button is being pressed correctly and that it is posting a flag properly
      ii. Physics task pend flag. Make sure that the pend flag works for button 1.
      iii. Physics task calculations. Make sure the calculation for how much the energy is being used per shield and the x and y of where the shield is activating
      iv. Shield activities. Make sure the if statement are correct for thing such as satchel hit but shield activation and railgun hit but shield activation.
2. Button 0 Test:
   a. The test for button zero is comprised of 5 sections
      i. Button 0 ISR. Test that the ISR is hit on both rising and falling edges. When rising it flags the railgun to start charging. When on a falling edge tells the railgun to stop charging. These are done using flags.
      ii. In the Railgun task send an LED 0 Timer Start Flag for the light to start increasing in brightness and to get time. When receiving a charge end flag send a LED 0 stop/turn off a flag and get time.
      iii. In the Railgun task calculate time. Use the start and end times gotten from when flags were enabled for start and stop. Make sure the value is passed into the railgun structure.
      iv. In Physics task. Make sure it gets the correct railgun charge time and calculates the right energy and power usage.

            v. In Physics task. Make sure the shot trajectory is calculated according to the formulas

3. Movement/slider Test
    a. There are 4 checkpoints for this.
        i. Test the movement task. When in the movement taks make sure it pends for the timer flag and goes off every 100 ms.
        ii. When getting the capsense value make sure that it is the correct value.
        iii. After getting the capsense value check if it is already that value and if is the value and set a new slider position value listed as -2, -1, 1, 2, or 0.
        iv. Test that the slider value is being passed to the movement structure.

Functional Tests

1. Bttn 0 activates led0 that blinks and gets longer/ led gets brighter. Pass (1)
    a. Press Bttn0, and led0 should blink on and off with it staying on longer the longer you hold it
2. Bttn 0 charges a charge and displays a shot with proper physics. Pass (2)
    a. Upon releasing bttn0 a shot will be fired from the cannon on the lcd screen and follow a parabolic trace
3. The Shot damages the castle appropriately. Pass (5)
    a. When a shot hits the castle the castle foundation should break apart. When hit the required amount of time the foundation should be fully broken and activate led1 to blink
4. Bttn 1 activates a shield (circle) on the LCD. Pass (2)
    a. Bttn1 should cause a circle on the lcd to flash and destroy satchels in a range
5. Shield destroys satchels that is within range. Pass (4)
6. Slider movement causes a platform to move around on the screen. Pass (1)
    a. Moving your finger on the slider should cause a small platform on the lcd to have a force applied in that direction
7. Slider movement is constricted to the screen and the platform bounces off the edges. Pass (2)
    a. When the slider hits the edge of the screen it will bounce off. If the velocity exceeds the max bounce velocity the platform with bounce with the max velocity
8. Castle is displayed on the screen. Pass (2)
    a. The castle should appear on the top left of the lcd
9. Satchels are thrown one at a time on the screen. Pass (4)
    a. There should be 1 satchel charge on the screen at a time and they are released from the castle at various velocities. When one hits the platform a game over the display is shown.
10. Satchels have n amount on the screen at once. Fail (2)
    a. There are n amount for satchels on the screen at once
11. Satchels are thrown periodically. Fail (2)
    a. Satchels appear on the screen periodically,

Summary

<u>Project Progress</u>

Summary Statements

Week 1:

- This week I made the Task diagram and identified cutting points to test the functionality of my project at certain points.

Week 2:

- This week I got the bttn0 to activate led0 properly, got the slider to move a platform, and identified unit tests.

Week 3:

- This week I got the slider fully functional, the railgun shoots correctly with a shot displayed, the castle drawn, and functionality tests identified.

Week 4:

- This week I got the satchels to work and have one on the screen at a time. The satchels also interacted with the slider and the shield. Also, the railgun shot does damage to the castle.

Week 5:

- This week I got the right parameters to work on the game, the castle to destroy properly when hit, and the winner and loser screens.

Week Final:

- This week was the demo and I got the shot hitting yourself working along with all previous weeks having other functionality working.

Estimated Completion and Effort Summary

Week 1:

- I have Completed 1% of the currently-scoped, estimated (1hrs estimated for work completed thus far 1/ 40hrs total estimate) in 2.5% of the total budgeted total-project time. (1hrs spent, of 40hrs total estimate). The work done took 1x (1/1) as much time as I estimated.

Week 2:

- I have completed about 25% of the currently-scoped, estimated (7hrs estimated for work completed thus far 7/ 40hrs total estimate) in 17.5% of the total budgeted total-project time. (8hrs spent, of 40hrs total estimate). The work done took 0.875x (7/8) as much time as I estimated.

Week 3:

- I have completed about 50% of the currently-scoped, estimated (15hrs estimated for work completed thus far 15 / 40hrs total estimate) in 37.5% of the total budgeted total-project time. (15hrs spent, of 40hrs total estimate). The work done took 1x (15/15) as much time as I estimated.

Week 4;

- I have completed about 75% of the currently-scoped, estimated (30hrs estimated for work completed thus far 23 / 40hrs total estimate) in 57.5% of the total budgeted total-project time. (23hrs spent, of 40hrs total estimate). The work done took 1.3x (30/23) as much time as I estimated.

Week 5:

- I have completed about 95-100% of the currently-scoped, estimated (32hrs estimated for work completed thus far 32/ 40hrs total estimate) in 57.5% of the total budgeted total-project time. (32hrs spent, of 40hrs total estimate). The work done took 1x (32/32) as much time as I estimated.

Week Final:

- I have completed about 100% of the currently-scoped, estimated (35hrs estimated for work completed thus far 35/ 40hrs total estimate) in 87.5% of the total budgeted total-project time. (40hrs spent, of 40hrs total estimate). The work done took 1.14x (35/30) as much time as I estimated.

<u>In-Scope Work Item Completion</u>

Not-yet Completed

- Structures

- Button ISR
  - Bttn0
  - Bttn 1

- Railgun Task

- Movement Task

- Physics Task
  - Platform
  - Railgun
  - Castle
  - Satchels
  - Interaction between items

- LED Task
  - LED 0
  - LED 1

- LCD Task
  - Castle
  - Satchels
  - Shot
  - Railgun
  - Platform

Completed

Week 1:

- Task Diagram
  - The task diagram shows the general outline of intertask communication and the data flow between interrupts, tasks, and other tasks. It also contains all the necessary tasks needed in order to carry out the game's functionality.

Week 2:

- Button 0 ISR

        - The Button 0 ISR interrupt on both rising and falling edges and send flags accordingly. When on the rising edge it send the railgun start charging flag and when on the falling edge it send the railgun stop flag.
    - Railgun Task
        - The Railgun task sends flags to the LED 0 to start getting brighter when the task receives a railgun start charging flag. When the task receives a railgun stop charging flag it send a flag to stop the led. The railgun task also gets the start and stop times and calculates the charge time. Once calculated it passes the value into the railgun structure.
    - Movement Task
        - The movement task uses a timer the flags it every 100 ms. When activated it uses the slider function to get the position. Then it checks what position is touches and if it has already been activated. This sets a variable to the direction the slider will move the platform. The direction variable is then passed to the movement structure.

Week 3:

- Physics Task
    - Platform
        - In the physics task, the platform movement physics are completed with it moving in the direction when pressing the slider and bouncing off the walls when hit. Also has a max force and max bounce speed set.
    - Railgun
        - The railgun/shot physics is complete with the shot being shot from the end of the railgun displayed on the LCD. Also the physics of the shot work and is shot parabolically.
- LCD Task
    - Castle
        - The castle and the foundation is drawn onto the top left corner of the screen.
    - Shot
        - The shot from the railgun starts at the end of the railgun and goes in a parabolic motion on the screen and when it falls out of the screen it is set back to 0, waiting for the next shot.
    - Railgun
        - Railgun is drawn onto the platform on the screen and moves with the platform.
    - Platform
        - The platform moves properly on the screen and does not clip off the screen.

Week 4:

- Physics Task
    - Satchels
        - The satchels are released from the castle edge with one on the screen at a time. The satchels have random velocity when thrown and bound off the

    - canyon walls. Currently, I have a placeholder that the satchels bounce off the platform for now before I change it to an end screen. Satchels also reset when they fall of the screen or are destroyed by the shield
  - Castle
    - The castle counts when it has been hit and increases the hit counter. It also breaks apart when hit.
  - Shield
    - The shield activates when bttn1 is pressed and destroys satchels when active and satchels are within range.

Week 5:
- Physics Task
  - Interactions
    - All of the different items in the game are able to interact with each other correctly with the satchel causing a game over when hitting the platform, the platform hitting the walls signifies a game over, hitting the castle doing damage, and using the shield on the satchel destroys them if the castle is hit after the foundation is destroyed you lose.
- LED Task
  - LED1
    - LED1 is working when the castle foundation is fully destroyed the light blinks for 5 seconds before displaying the winner screen.

Week Final
- Physics Task
  - Made the shot able to hit the platform and cause a game over.

Risk Registers
- In excel sheet

Analysis

The priorities of the task are as follows physics, slider, LCD, railgun, and LED. The highest being physics and the lowest being LED. This allowed for the task that needed the highest priority to be completed first. In the segger view screenshots in the git hub, it is clear that the LCD updates every 300 ms and physics updates every 100 ms. This is not the given value because the os_tick is in 100ms and the ticks are uint so the smallest is 1 tick. Instead, I used 1, 2, and 3, tick which is the same ratio between tasks. Since the update speed is double the ideal update, the game doesn't run very smoothly. I also had to increase movement values to compensate for that. Going back to the segger view, the movement task does not seem to be updated every 200 ms and instead is going in every 10 ms. I do not know why that is happening, maybe it's because it is trying to sample but it shouldn't because there is a mutex. Also while debugging when turning off the timer for movement it does not access the movement task.

In the physics task, I grouped things based on interactions and what items are being used. So for the slider, I used the slider values and physics along with the edges of the LCD for the collisions. For the railgun, I calculated the shot movement and used some of the slider values to incorporate, such as slider position and slider velocity. I also used an if statement to check if the shot hit the platform. For the shield, I just set the conditions when active. In the satchel section, I put the satchel movement and conditions for when interacting with the wall and shield, platform. For the castle, I have all the conditions for when it is hit in there, along with the game-over condition if hit after the foundation is destroyed. As for the storage, the section only refills the storage or when full stays at that value. All the decreased energy conditions occur in the respective places that are decreasing the energy. I did this order because to me it makes the most sense to section items and what they interact with as a single section. Some difficulty with doing this was that I realized I was using 3 different axes for the satchel, shots, and platform. This made it hard to get everything to interact because I had to shift values to get them to line up with each other.

Regarding scalability, I made a ratio value that is the ratio of pixels to meters and used that as my space conversion value. This allows for pretty good scalability when changing game values. However, I did not scale the cannon so that does not change. Besides that, the castle size should update to an appropriate size when compared to the canyon size. Also, all the movement data should update pretty well with some adjustments to some data values. As for time scalability, that is dependent on the value used as the change in time. This value is directly used in the calculations and is not a variable, but changing this value helps with increasing and decreasing time updates and allowing the game to still be playable. I found the castle size to not be that important since in my game it scales in proportion to the length of the canyon. For extra constraints, I feel like making a space ratio value for pixels to meters would have been helpful.

If I had more time on this project I would probably end up adding a menu, leaderboard, better animation, and maybe try to change the os tick so it runs smoother. The ticks are one I am invested in because the way the game runs right now does not look good because it refresh is too slow. As for the other addition, these would make the game more competitive and look nicer.