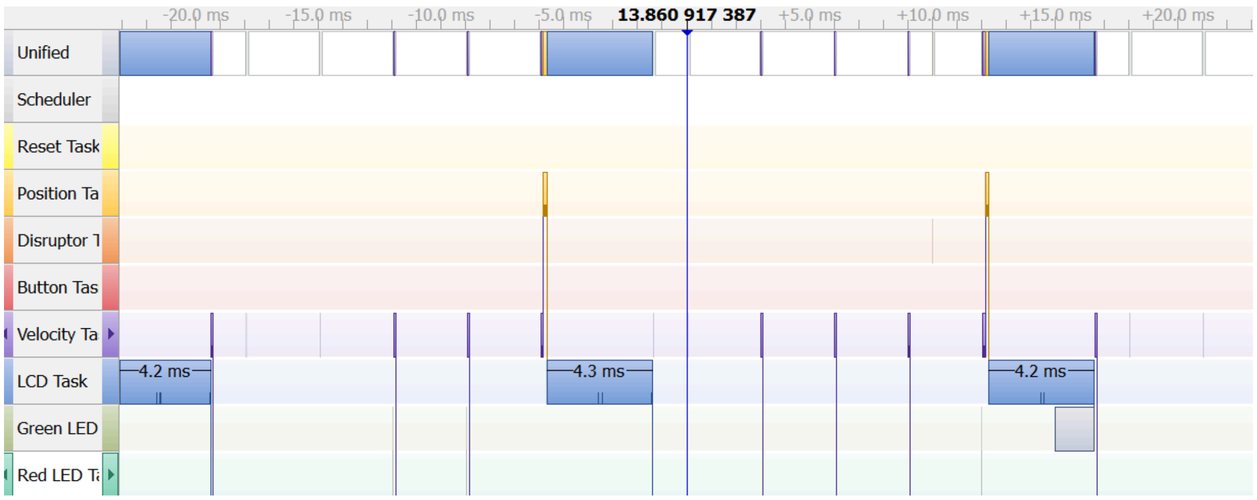
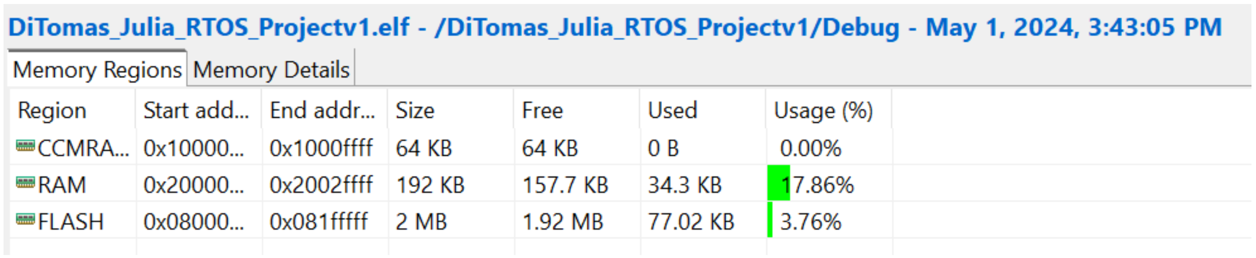
- RT tasks: What are your priorities, execution times vs. deadlines as seen with Segger SystemView or other analysis? (Include screenshots) Conflicts seen that kept it from operating as you planned? (2)



To avoid tasks conflicting, I attempted to make each task only execute as frequently as needed. This was not really a concern for tasks that are triggered by external events (such as a button press or winning the game) but rather for the LED tasks, the velocity task, and the disruptor task. For the green LED, I made the frequency 40 Hz, since I found that to be enough to hide the flickering. For the red LED, I have it calculate how long the light will be on or off (based on the energy level), and then delay for that long. The disruptor task's frequency just gave resolution to the energy level, so it did not need to be very high since all that might change is an extra second fraction that the disruptor can be on or off. Finally, I made the velocity update frequency pretty high, to smooth out the gyro motion, without causing the map to flicker too much. (I was not able to get only a portion of the map to be redrawn each time, which is why the LCD task takes awhile, but it only executes every 5th time that the velocity task does.) After adjusting the task priorities, and more importantly, the frequencies, I did not have any conflicts prevent the planned operation.

- Code Space: how much, and evaluative comments (2)



DiTomas_Julia_RTOS_Projectv1.elf - /DiTomas_Julia_RTOS_Projectv1/Debug - May 1, 2024, 3:43:05 PM

Memory Regions | Memory Details

Region	Start add...	End addr...	Size	Free	Used	Usage (%)
CCMRA...	0x10000...	0x1000ffff	64 KB	64 KB	0 B	0.00%
RAM	0x20000...	0x2002ffff	192 KB	157.7 KB	34.3 KB	17.86%
FLASH	0x08000...	0x081fffff	2 MB	1.92 MB	77.02 KB	3.76%

The build analyzer screenshot is shown above for the final project submission. For the beginning of the project, I was using the original Lab 7 LCD driver, and about halfway through, I ran out of RAM (I think it was about 97%). That is when I switched to the SDRAM LCD driver and was able to continue with the project. Now, for the complete project, RAM is only at 17.68%.

- Evaluation of your approach(es) to the physics update requirement. Explain your rationale for grouping the physics updates and edge case handling the way you did (2). What limitations did you have to deal with that were not obvious at first? (1)

I created one task to sample the gyroscope, calculate the angles, and update the velocity. It is a fairly simple task that occurs on a timer and averages a customizable number of gyroscope samples before updating the position task with the new velocity (which allows these events to happen at different frequencies). The position task is a lot more complex and requires much more data to update the ball's position and check for collisions. It checks for waypoints and holes first, since if the game ends, then the rest is unnecessary. It then checks for simple collisions (the ball moved into one of the walls bordering the cell that it started in before that motion). After that, if there has not yet been a collision, it checks for the complicated corner collisions. This allows it to execute the simpler and better-working code if it can (since the bounce off the corners is a little funky). The biggest non-obvious limitation was the gyro drift. The gyro seems to have an error that can grow over time substantially into a large angle, even when the board is not actually tilting. This was somewhat mitigated by setting the angles back to 0 when the game resets (which assumes the user holds it flat when they press the reset button).

- Configuration Data (think back to the lecture including "corner testing"): what ranges did you find playable? Which configuration data did you find unnecessary to either have, or to have any flexibility with? (1)

I think that playable ranges for each configurable item depend on the other configurations. For example, the ball could not have a diameter of 16 pixels if the walls were only 10 pixels apart, or else you would just be waiting for the disruptor the entire time. Since the random map might require use of the disruptor to win, there must be at least enough disruptor energy to use it when fully charged. I personally found the edges really hard to avoid, the game more fun with more cells in the maze (which implies a smaller ball), and enough time to get to all the waypoints without feeling insanely rushed. I think that the frequency configuration data is probably unnecessary, because those don't really change the gameplay itself. I also think that the pin maze items might be unnecessary, because when playing the game you typically don’t give it a super hard twist unless you are trying to throw the ball.

- What would be your next steps if you had just another 2 weeks to work on your project, and why? (2)

While there are things that I could do to improve what I already have (interactions with walls, the maze being redrawn more efficiently), I think it would be more fun to add more to the game. With two more weeks I would probably add Pac-man style enemies (in the shape of ghosts or something else) that move around the maze (but cannot tunnel through walls) and must be avoided. I also might see if there are better ways to fix the gyro’s weird errors, besides just averaging samples, because that does affect the gameplay quite a bit and make it a little frustrating.