ECE 4550/5550G Project 1

Due on Thursday February 16, 2023 at 11:59PM in Canvas (200 Points)

Objective
To become familiar with FreeRTOS.

1 Getting Started

For this and future projects, you will need an Arduino Mega (or another compatible board such as ELEGOO Mega 2560, see the project FAQ page `https://canvas.vt.edu/courses/165624/pages/project-faq`). If you have not worked with an Arduino before and would like to learn more, the vendor website (`https://www.arduino.cc/`) is a great place to start. Much like Linux, Arduino is very popular and plenty of online resources are available.

Along with having your Mega, you will need to download the Arduino IDE:
`https://www.arduino.cc/en/main/software`. The IDE allows you to write code, compile and debug it, and upload it onto the Mega itself.

For the rest of the semester, you will be working with a real-time operating system (RTOS). We will use FreeRTOS, which is often the RTOS of choice in the real world. Much like Arduino, FreeRTOS has a large online community. The starting point is: `https://www.freertos.org/`.

Your first task is to install the FreeRTOS libary and run the `Blink_AnalogRead` demo program. You can do so by following the steps outlined in this blog post: `https://feilipu.me/2015/11/24/arduino_freertos/`. To open the library manager, go to `Arduino -> Tools -> Manage Libraries`. Be sure to select version 10.4.3-6 instead of the newest version.

To try the `Blink_AnalogRead` demo, select `Arduino -> File -> Examples -> Examples from Custom Libraries -> FreeRTOS -> Blink_AnalogRead`. Once you upload said program to your Mega, you should see the LED next to pin 13 blinking on and off at a one second interval. Also, open up the `Serial Plotter` (under `Tools`) and observe the output.

2 Tasks & Rubrics

1. (20 points) Study the code in the `Blink_AnalogRead` demo program and answer the following questions. Note that you will likely need to refer to the FreeRTOS tutorial and reference manual (`https://www.freertos.org/Documentation/RTOS_book.html`) to understand some of the code. The `Blink_AnalogRead` implements two periodic tasks. What are the names of these tasks, their periods, and their priorities? Copy and paste the relevant segment(s) of the code if it helps you to answer the question.

2. (40 points) One nice feature of the Arduino IDE is that you can communicate with the Mega in real time. This is done via the `Serial Monitor` (under `Tools`). Read the following reference on serial communications in Arduino: `https://www.arduino.cc/reference/en/language/functions/communication/serial/`, paying a particular attention to the `print` and `println`

function. Then, modify the demo program so that, using the `Serial Monitor`, you can see which task is executing (and when, by enabling the timestamp in the `Serial Monitor`). In your report, include the relevant code snippet(s) and a screenshot of the output of the `Serial Monitor`. You may change the blinking frequency if that makes it easier for you to debug your code. Note that you can also input data to the Mega in real-time using the various read functions. Feel free to try that!

3. (60 points) By default, FreeRTOS uses a simple priority-based scheduler. Now, let us consider the following task set consisting of three dummy tasks where π_i denotes the priority of task τ_i (a higher value means a higher priority). All time are in seconds.

Task	T	π
τ_1	2	2
τ_2	3	3
τ_3	5	1

Create a program to implement these tasks in FreeRTOS. The body of these dummy tasks don't have to do anything (but can if you wish). Include the entire code in your report, with helpful comments where appropriate, along with a screenshot of the output of the `Serial Monitor`. Briefly explain why your implementation is correct.

4. (80 points) One of the most fun things about being in this class is that you can make change to the RTOS itself and see what happens. For this, you will need to modify the FreeRTOS library that you have downloaded. Note that the source code for FreeRTOS should be inside `Arduino -> libraries`, where the `Arduino` directory is most likely in your home directory (please see the documentation on the Arduino IDE if you cannot locate this directory). Open the file `tasks.c` and skim through it. Note that the FreeRTOS reference manual contains a detailed description of each of the function. Your last task is to locate the part of the code where the highest-priority job is selected for execution, and modify the code so that the lowest-priority job is selected instead. Run the same program from #3 and observe the output. In your report, describe the changes you made to `tasks.c` and explain what you observe in the output and why. You do not need to include a screenshot of the `Serial Monitor` in this question.

NOTE: the `tasks.c` file is quite long, so please start the project early enough.

3 Turning In Your Work

You are to upload the following documents on Canvas.

- Your report in PDF format. At the beginning of the assignment, please list the following statement along with your **signature**.

  `I have neither given nor received unauthorized assistance on this assignment.`

- The modified demo program from #2.
- Your program from #3.
- The modified `tasks.c` file.

Please remember that you must turn in your own work. Failure to do so will result in you being reported to the university.