Assignment 4: RTOS and Priority Preemption - Part 2

Group 20:

Ashlesha Deokar: G01374665
Aniket Anil Raut: G01387118
Mandar Chaudhari: G01393699
Ganesh Madarasu: G01413183

Github Repository: https://github.com/mandarc64/CS692_001_G20

Real-Time Stopwatch Application Thread Priority Documentation

In this real-time stopwatch application, we have three main threads:

1. Button Status Thread: Monitors the state of the start/stop button.
2. Reset Button Thread: Monitors the state of the reset button.
3. Time Printer Thread: Displays the elapsed time on the terminal.

Priority Assignments

The priorities for these threads are assigned based on the Rate Monotonic Scheduling (RMS) principle, which states that the thread with the shortest period (highest frequency) should have the highest priority.

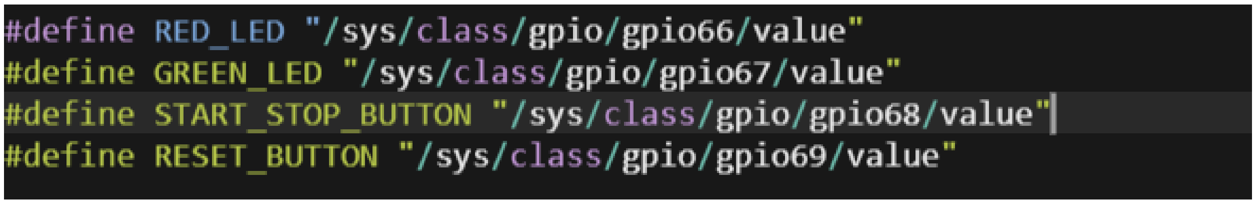
1. Button Status Thread
Role: Toggles the stopwatch state and updates the start/stop flag.
Priority: Highest
Reasoning: This thread has the highest frequency as it needs to poll the start/stop button every 10 ms to ensure a responsive user interface. A higher priority ensures that the stopwatch can be started/stopped with minimal latency.

2. Reset Button Thread
Role: Resets the stopwatch and updates the elapsed time.
Priority: High
Reasoning: Although this thread has the same frequency as the Button Status Thread, it is given a slightly lower priority because resetting the stopwatch is typically a less frequent operation than starting/stopping it. However, it still needs to be responsive, hence the high priority.

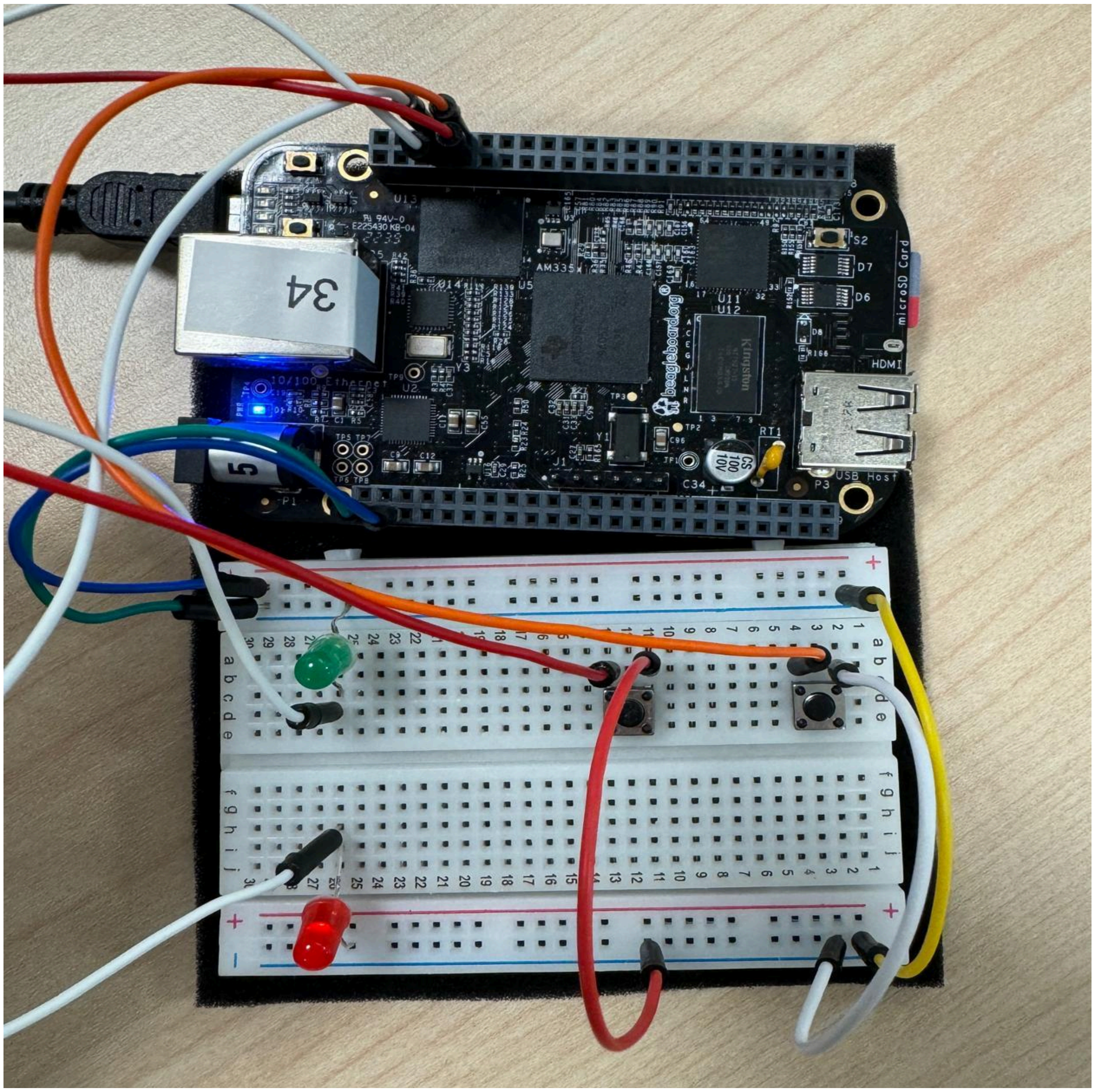
3. Time Printer Thread
Role: Displays the elapsed time on the terminal.
Priority: Lowest
Reasoning: This thread has the lowest frequency and is responsible for updating the terminal display, which is less time-critical than handling button presses. Therefore, it is assigned the lowest priority.

Supporting Material

Connections:



```
#define RED_LED "/sys/class/gpio/gpio66/value"
#define GREEN_LED "/sys/class/gpio/gpio67/value"
#define START_STOP_BUTTON "/sys/class/gpio/gpio68/value"
#define RESET_BUTTON "/sys/class/gpio/gpio69/value"
```



Conclusion

The priority assignments for the threads in this real-time stopwatch application are based on the principles of Rate Monotonic Scheduling and the specific roles and frequencies of each thread. It ensures that the application remains responsive and functions correctly.