**Lab 2 Report**

Tyler Crabb

A01389040

Carter Nettesheim

A02248990

Discuss the differences in the schedules generated using RM and EDF in terms of preemptions and deadline misses, along with any other conclusions and observations.

Both the rate monotonic (RM) and the earliest deadline first (EDF) algorithms are good algorithms, which can schedule tasks in a priority list. However, because earliest deadline first is a dynamic algorithm it often finds a better schedule than rate monotonic can. This happens because of how EDF works. In fact, the name EDF, or earliest deadline first, is the whole reason that EDF is faster than rate monotonic. Unlike RM, EDF prioritizes the next deadline. This could be a short, periodic task, or an aperiodic task. This allows the algorithm to always find the best utilization making it an optimized algorithm. For the same reason, EDF is harder to implement and can take more processing power. RM, on the other hand, always prioritizes periodic tasks. The priority is set beforehand and only the periodic tasks are known beforehand. Because RM always prioritizes periodic tasks it is easier to implement and takes less processing power, ideal for smaller RTOS applications, and lower utilizations.

As we built both EDF and RM we realized there were many ways to implement them. We went for an easier, but process heavy version, using lots of for loops. We implemented a queue. In rate monotonic, the priorities were set beforehand based on period. At every time slot, RM checked the priority list to see if any tasks needed to be released. If they did, it would release them, or add them to the queue. As time went on, it would continue to add new tasks to the queue in the order of the set priority. We made a queue for both aperiodic and periodic. Since periodic has higher priority in RM, the periodic queue always was checked first. If nothing was there, then the aperiodic tasks were run. This is the biggest downfall of RM, and we noticed it in tests.

EDF allows both aperiodic tasks and periodic tasks to have priority. It’s like the Martin Luther King speech for tasks, all tasks are created equal. Whatever task needs to be done first, will assume priority. In RM we noticed if there wasn’t much slack time the aperiodic tasks would never get done, and end up missing their deadline, even though the deadline is 500 ms after release time. This is plenty of time to get the task done, but because of the algorithm it never gets done.

In conclusion, we found RM works great for periodic tasks where the priority can be known from the beginning, and the utilization never changed because of aperiodic tasks. It works perfectly well, is simpler to implement, and takes less processing power. EDF on the other hand handles aperiodic tasks much better. Outperforming RM in high utilization with aperiodic tasks. RM is a great algorithm and will work for many situations, but EDF will need to be implemented in the case of a regular everyday computer, because of how many tasks need to be handled with equal priority, not to mention the aperiodic tasks from mouse and keyboard input. EDF is an optimized algorithm and if feasible, will find a schedule fitting all tasks within. RM may not find as good of a schedule due to the pre-prioritized tasks.