COMP 3510 – Embedded Systems Development  
Rain Li, David Harris

**Lab 1 Report**

Our implementation of the Lab 1 assignment successfully detects when a device has generated an event and processes events in the buffer to make way for new events. Put simply, our code works.

Below is the information we collected by testing our code with preset values:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Number of Devices | lambda | mu | Avg Missed Events % | Avg Response Time (s) | Avg Turn Around Time (s) |
| 2 | 2 | 10 | 0.5 | 0.028097 | 0.13624 |
| 2 | 2 | 30 | 14.5 | 0.099073 | 0.305058 |
| 2 | 2 | 60 | 30 | 0.19745 | 0.474583 |
| 2 | 2 | 90 | 46 | 0.293826 | 0.570439 |
| 4 | 2 | 10 | 0.5 | 0.013155 | 0.070574 |
| 4 | 2 | 30 | 4.5 | 0.09666 | 0.230552 |
| 4 | 2 | 60 | 14.75 | 0.2313945 | 0.443452 |
| 4 | 2 | 90 | 26.75 | 0.394554 | 0.61029525 |
| 8 | 4 | 10 | 0.25 | 0.011846375 | 0.070233125 |
| 8 | 4 | 30 | 1.375 | 0.089242625 | 0.247915625 |
| 8 | 6 | 60 | 6.77125 | 0.538235875 | 0.947026125 |
| 8 | 6 | 90 | 17.6525 | 1.144154 | 1.543744375 |

Our results indicate that, with optimal service time, our code is able to process generated event with over a 99% hit rate. As the time it takes to service each event (**mu**) increases, our code struggles to process each of the events in a enough time to ensure that each event is cleared from the buffer to make way for the event that follows it. Furthermore, as the average interval time between arrivals (**lambda**) increases, the percentage of missed events falls, as there is more time for the program to react to new events that arrive before another event comes after it.

The code is not perfect. The program still misses events. With the round-robin method of execution, device n can be serviced, and then as the program moves on to n+1 and n+2 and so on, depending on the time it takes to service each event, device n may have time to generate more than one event, which will result in a miss.

Our code minimizes the time it takes to completely service all buffered events in one cycle by only cycling through the number of devices that are actively generating events, not by checking each flag individually on each cycle.