Carlos A. Rodriguez San Miguel March 24th, 2025

CIIC4050-096

Programming Project 1 – Conclusion of Results

1. Which method maintained a more accurate 50ms interval?

We collected the time it took for each process to update the values of the coordinates into logs, and inserted these values into an excel file to analyze them closer. This data and more can be found in the elapsed\_times.xlsx file.

Here are the resulting calculations of the average time in each process:

  
  
  
  
  
We can see that the POSIX timers method, on average, held a more accurate interval of 50ms and having only 0.1% of error. On the other hand, utilizing threads and usleep(50000) held a less accurate interval of 50ms, having 4.4% of error.

2. Which method consumed fewer system resources?

Logically, the method with the thread should use more resources due to a thread creating a stack for itself and such. Both the processes call a function which writes on a log file, so that doesn’t really have to be considered much. The process with the timers does everything on main with no threads launched, so that is its advantage. Generally, using HTOP on the terminal, we saw that %CPU bounced between 5-15% and %MEM bounced between 2-3%, but no discernable patterns were found between the processes.

Overall, due to the use of threads, we can say that the POSIX timers process used less resources while maintaining a tighter interval.

3. Which one would I use on a real-time system?

Real-time systems require heavy constraints as well as strict time intervals. The percent of error between the two processes is small, but regardless, the accuracy in intervals on the process that utilizes POSIX timers is far closer to the real value compared to the one using the thread. Therefore, due to its slight less usage of resources and better compliance with the time interval, I would choose the POSIX timers process to use on a real-time system.