Assignment 1 - Simulation Analysis
SYSC4001 - L1
Kalyah Mckesey - 101307188
Sarah Andrew - 101303468
https://github.com/kalyah-ui/SYSC4001_A1

Purpose

This simulation demonstrates the interrupt process. It displays the average time required for a device to complete a task, simulating each execution step. The program uses a vector table to locate and load the address of the ISR into the PC. It also utilizes a device table that gives the average time it would take for the device to complete its I/O. The trace file contains a sequence of events that occur during the execution process. Once the events are finished, the device sends an interrupt via END_IO to the CPU. An output trace file is then provided that includes all the steps in the interrupt process. We used 20 different trace files, varying the delay times, number of devices, and ISR durations, to analyze the effects of changing these values.

Varying save/restore context time

When changing the save/restore value from 10 ms to 20 ms, the restore/save value increases during each cycle. This is because the CPU spends a significant amount of time simulating the restoration and saving of values as the number of activities associated with the interrupts increases. Thus, the interrupt service routine takes a longer time, at 20 ms. When analyzing the execution data for 20 ms, the CPU takes a longer time to save the current state after the ISR is completed. Moreover, the save/restore value is changed to 30 ms, which takes even longer than 10 ms and 20 ms. This indicates that the CPU is occupied for a longer period during each interrupt, and the overall program is delayed. To summarize, the higher the context/save value is, the longer the CPU is occupied.

Varying ISR activity time

When varying the ISR between 40 and 200, the CPU takes longer simulated time during the interrupt service routine. This is because when an interrupt occurs, the CPU services the interrupt and suspends the program execution. At this stage, the CPU is managing the interrupt activities and cannot perform additional instructions. The program remains halted until the service routine is completed and the CPU is restored using the IRET instruction. When analyzing the execution data for 40 ISRs for device 10, the CPU processes the interrupt for 156 steps.

In contrast, when the ISR is set to 98, the CPU simulates taking more time to process the interrupt. For device 7, it executes three ISR activities, with each being 98, 98, and 69 steps, resulting in a total of 265 steps. Once the ISR value increases, the CPU will simulate allocating additional time to handle the interrupt. This shows that a higher ISR value will cause the CPU to spend more cycles handling the interrupt. If an ISR execution takes too long, the CPU will continue to handle the interrupt. This may cause the overall program to be delayed, and the

interrupt latency would increase for additional interrupts. The difference in the speed of these steps affects the overall execution time of the process, as the CPU will spend more time processing the ISR. If the ISR is a high value, that increases the total execution time. When the total execution time increases, it may delay other processes, resulting in the CPU being occupied for a longer period. In summary, a lower ISR value results in the CPU taking less time to deal with an interrupt.

Varying CPU Time

While varying the CPU time is somewhat similar to varying the ISR time, as they both affect the CPU's speed, they also impact the amount of simulation time spent in different modes. Increasing CPU time effectively increases the amount of time spent in user mode throughout the process. Conversely, increasing ISR time increases the amount of time spent in kernel mode. The CPU time has a direct effect on how long the interrupt is serviced, as the CPU spends more time on user processes.