SYSC 4001

Assignment 1 Report

Sydney Kuhn - 101302923
(Student 1)

Keon Foster - 101303908
(Student 2)

Trace Files

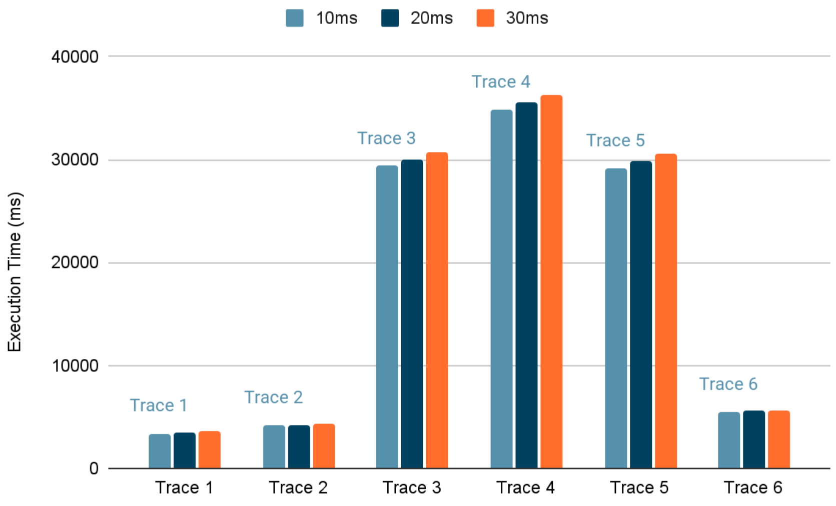
[Input Trace Files - Github]

Test Cases

Save/Restore Context Time

Execution Time of Traces with varying Context Times

Trace	10ms	20ms	30ms
1	3402	3502	3602
2	4172	4232	4292
3	29390	30050	30710
4	34851	35531	36211
5	29172	29892	30612
6	5490	5590	5690

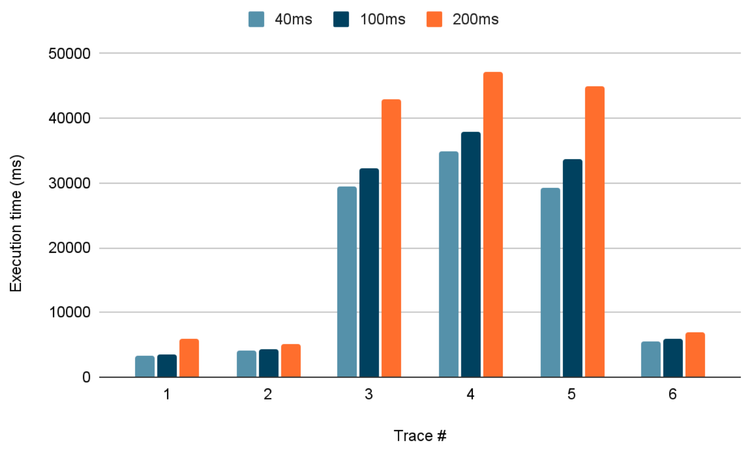


The execution time results above show simulations run with three different save/restore context durations: 10ms, 20ms, and 30ms. As demonstrated in the graph, increasing the context time leads to a corresponding increase in overall execution time. This is because every time an Interrupt Service Routine is triggered, the system must save the current process state and restore it afterward. The longer this save/restore operation takes, the more overhead is added to each interrupt. As a result, traces with more ISR events experience a greater impact. For example, if a trace contains 20 ISR's, and the context save/restore takes 10ms longer than usual, that trace will now take 200ms longer than ideal. This is especially evident in trace 4, which contains the highest number of ISR-triggering activities and shows the most significant increase in execution time. The simulation clearly demonstrates that the more frequent the interrupts and the longer the context switch duration, the longer the total program execution time becomes.

ISR Activity Time

Total execution time of traces with varying activity times

Trace #	40ms	100ms	200ms
1	3402	3564	5838
2	4172	4341	5030
3	29390	32359	42883
4	34851	37852	47104
5	29172	33597	45013
6	5490	5879	6989



As seen from the data, increasing the execution time of the ISR activities significantly increases the overall execution times. This increase can be most prominently seen in the longer traces which have a larger amount of SYSCALL and END_IO activities. The larger amount of ISR activities all experiencing an increase in execution time add up to a much greater increase in the overall runtime than compared to the smaller traces. The change in execution times also cause drastic changes within a single trace as well. This can be seen very well in traces 3, 4, and 5 who all noticed a small difference in total execution time when going to 100ms (2.5x increase in activity time) compared to a very large difference when going to 200ms(2x increase in activity time). If we were to take an example trace with 10 SYSCALLs/END_IOs we would see a total ISR execution time of 400ms, 1000ms, and 2000ms respectively. Increase the number of activities from 10 to 20 and the execution times become 800ms, 2000ms, and 4000ms respectively. This example showcases how the longer activity times impact longer traces far more severely, indicating a need for long programs to properly optimize their processes to avoid extremely slow performance.