GitHub: https://github.com/Hey1121/SYSC4001_A1

Part 2:

Trace file 0-5 demonstrate the default behaviour of the simulator.

Trace file 19 contains the default version of a trace file that will be executed with various modifiers.

It has a duration of 3102 ms

Trace file 6 behaves as if the CPU is 50% slower than default, meaning that all CPU bursts take 1.5 times the time to execute.

It has a duration of 3295 ms, 106% the default time, meaning that the duration of the CPU burst has a minor effect on the total time taken by the program.

Trace file 7 has a context switching time of 20 ms instead of the default 10ms

It has a duration of 3162 ms, 101% the default time, meaning that the duration of the context switch has a minor effect on the total time taken by the program.

Trace file 8 has a context switching time of 30 ms instead of the default 10ms

It has a duration of 3222ms, 103% the default time, meaning that the duration of the context switch has a minor effect on the total time taken by the program.

Trace file 9 has the ISRs have a flat 100 ms increase in time taken.

It has a duration of 3702, 119% the default time, meaning that the ISR has a noticeable effect on the total time taken by the program.

Trace file 10 has the ISRs have a flat 200 ms increase in time taken.

It has a duration of 4302, 138% the default time, meaning that the ISR has a noticeable effect on the total time taken by the program.

Trace file 11 has the kernel mode operation time increased to 2ms (from 1ms).

It has a duration of 3108, a negligible impact on the time taken.

Trace file 12 has several I/O operations per CPU burst. This is to simulate an operation that, for example, takes many inputs for a given CPU action.

Trace file 13 has the opposite, with several CPU bursts per I/O cycle. This is to simulate an operation that, for example, has low amount of data, but needs several complex math operations applied.

Trace files 14-20 show the behaviour of the simulator in various situations with default settings.

In conclusion, in most operations most of the time taken by programs of this nature are taken by the ISRs of the I/O devices. Therefore, a good way of speeding up these programs is by minimizing the I/O calls or by having the CPU perform useful actions during the ISR (by way of multithreading).