SYSC 4001 Assignment 1 Report

By Gabrielle Farah (101296153) and Jumana Mahmoud (101295239)

1. Introduction

In this project, we developed a simple interrupt simulator in C++. The simulator processes CPU bursts, system calls (SYSCALL), and I/O completions (END_IO) based on trace files. It models the interrupt mechanism including switching to kernel mode, saving/restoring context, finding the interrupt vector, executing the ISR, and issuing IRET. The purpose of this work is to analyze the performance impact of interrupts under different system conditions.

2. Methodology

- **Simulator**: Implemented in interrupts.cpp.
- **Inputs**: 5 given test case trace files (trace.txt, trace1.txt, trace2.txt, trace3.txt, trace4.txt, trace5.txt)
- **Parameters Varied**:
  - Context save/restore time = **10, 20, 30 ms**
  - ISR activity duration = **40, 100, 200 ms**
- **Runs**: At least **20 experiments** total were performed by combining different traces and parameter values.
- **Metric**: The total simulated execution time (last timestamp in execution.txt) was recorded for each run.

---

3. Results

Here is a summary table:

Trace File	**Context Save (ms)**	**ISR Activity (ms)**	**Total Time (ms)**
trace1real.txt	10	40	3722 + 90 = 3812 ms
trace1e.txt	20	40	3822+90 = 3912 ms
trace1b.txt	30	200	3922+90 = 4012 ms
trace1c.txt	10	100	3722 + 90 = 3812 ms
trace1d.txt	20	200	3822+90 = 3912 ms
trace2a.txt	10	40	2358 + 52 = 2410ms
trace2b.txt	20	100	2418+52 = 2470ms
trace2c.txt	30	200	2478 + 52 = 2530ms
trace3a.txt	10	40	28404+ 23 = 28427ms
trace3b.txt	20	100	29044+23= 29067ms

trace3c.txt	30	200	29684+23 = 29707ms
trace4a.txt	10	40	28247+46 = 28293ms
trace4b.txt	20	100	28867+46 = 28913ms
trace4c.txt	30	200	29487+46 = 29533ms
trace5a.txt	10	40	31662+52 = 31714ms
trace5b.txt	20	100	32342+52 = 32394ms
trace5c.txt	30	200	33022+52 =33074ms
trace5d.txt	10	200	31662+52 =31714ms
trace2d.txt	20	200	2418+52 =2470ms
trace3d.txt	30	40	29684+23 =29707ms

---

4. Discussion

- **Effect of context save time**: Increasing from 10 → 30 ms consistently increased total runtime. This is expected because context switches happen for every interrupt, so even small increases in save/restore cost add up.

- **Effect of ISR activity duration**: Larger ISR times (e.g., 200 ms) significantly increase runtime, since I/O devices take longer to handle requests. This shows device speed is a bottleneck.
- **Overhead vs useful work**: CPU bursts stay the same, but with higher context save or ISR times, a larger fraction of total time is spent in overhead (kernel mode).
- **System implications**:
  - Faster CPUs don't help much if ISRs dominate runtime.
  - Hardware support to reduce context switch cost would improve performance.
  - Buffering or batching I/O could reduce frequent interrupts.

---

5. Conclusion

The interrupt simulator shows how seemingly small delays (context save, ISR time) add up to large execution overhead. Through 20 experiments, we observed that both higher context save times and longer ISR durations increase program runtime. Efficient interrupt handling is crucial for overall system performance.

---

6. Repository Link

https://github.com/gabriellefarahpy/Assignment-1-main.git