Assignment 1

SYSC 4001A (L3)
Student 1: Nicholas Dorscht (101298132)
Student 2: Eshal Kashif (101297950)

Part II - Design and Implementation of an Interrupts Simulator

Introduction

This assignment served as a way to discover and explore the concept of hardware interrupts by going through the steps a short program goes through when it is executed. It goes through the different states of executing a system call, including switching execution modes, saving context, and processing values from the ISR table. Additionally, it explores how each of these steps affects the runtime of the program in different ways, and how changing some values may have a more dramatic effect than others.

Discussion

By varying the values of certain variables, such as the length of time the ISR takes to run, as well as the amount of time the CPU takes to save/restore context, we were able to gather data on how these changes affect the overall runtime in different ways. Additionally, we asked ourselves some questions as to how different factors that are not necessarily software related can also affect execution time.

One of the first changes we explored was the effect of changing the time it takes to save and restore context. While this may seem trivial, as its execution time is short compared to other delays such as device drivers, for every interrupt context must be saved and restored. This means that the effect of doubling the time is seen twice which can lengthen execution of a program by entire seconds depending on the length of the program. Another change we tested was the time it takes for the processor to execute the ISR. While this change is only seen once per interrupt, a much longer ISR execution time delays the entire system as the processor has to wait to execute the ISR before it can move onto the next interrupt. This means that having longer execution times can result in more and more interrupts stacking up. Overall, we found that these changes both lengthened the amount of time it took for the program to execute then they were increased.

Examples of a test case where we changed the context save/restore time can be seen below. In Figure 1, we use a context save/restore time of 10ms, which results in a total time of 524ms at the end of the screenshot. In Figure 2, a context save/restore time of 30ms is used resulting in a total time at the end of the image of 584ms (60ms more, since 30-10=20 and we save twice and restore once here), causing longer execution of time.

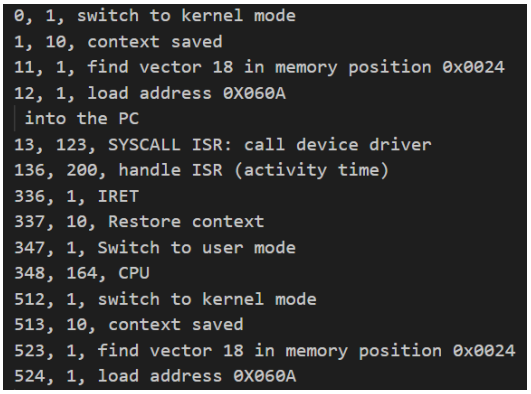

Fig. 1: Trace file showing longer ISR and context times (context time=10ms).

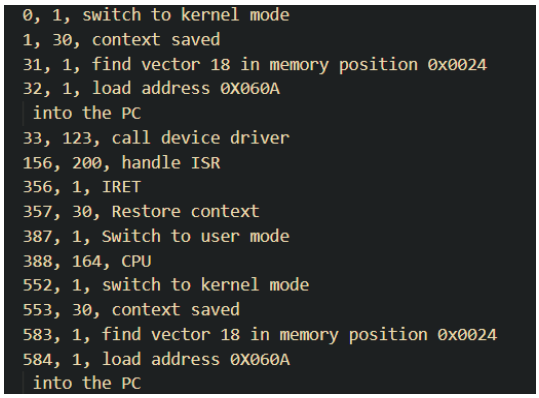

Fig. 2: Trace file showing longer ISR and context times (context time=30ms).

This shows that context switching is a significant contributor to interrupt-handling overhead and that optimizing it or using faster context save hardware could noticeably improve overall system performance. We achieve a similar result by changing ISR activity times. In some cases, long ISRs can cause a timeout, potentially causing data loss or crashes.

Another non-software factor that could affect execution time is the speed of the CPU being used. Having a faster CPU could shorten execution time, but not in the same way as the software. While better software can decrease the amount of clock cycles needed to run the program, a faster CPU will run the same amount of cycles as a slower one. Instead, it shortens execution time with a faster clock, allowing it to process commands at a faster rate.

Additionally, if the vector table used 4-byte addresses instead of 2, each entry in memory would take twice as much space, so the interrupt vectors would be spaced further apart (i.e. device 7's vector would move from 0x000E to 0x001C). This may increase the total size of the table and could slightly raise lookup time if we model each byte access taking time.

Lastly, another interesting question that was explored was the effect of I/O devices becoming slower. Increasing device delays in the table to simulate slower hardware resulted in increased execution time. The CPU spent time waiting for End of I/O events, despite the interrupt handling events remaining the same, demonstrating that system performance is heavily influenced by I/O speed. Slow devices can create bottlenecks, reducing output throughput despite a fast CPU.

Github

https://github.com/nicdorscht/SYSC4001_A1