

CS4348 : Operating Systems Concepts

Homework Assignment 1

Matthew McMillian
mgm160130@utdallas.edu

August 28, 2018

1. Figure 1.11 on slide 26 is displaying what happens when an interrupt occurs within the OS, and how the system handles the memory and registers. In the figure, part (a) shows what happens when an interrupt happens at an arbitrary location 'n'. During this process, the current program counter + 1, general registers, and stack pointer get pushed to the control stack so that the task that triggered the interrupt doesn't overwrite anything from the inside of the task that the outside environment was using. Part (b) describes the events that occur when the interrupted event ends. After the event ends, the program counter, general registers, and stack pointer get updated with the values stored on the control stack.
2. Memory hierarchies are important for a system. Memory comes in many different sizes and speeds, and there are trade-offs with each one. Typically, faster memory is more expensive and it has less of a size, and slower memory is cheaper but it has a larger size. Therefore, in order to optimize our available options, we use both fast and slow memory in various sizes and speeds to get the most out of the system. A good example of this is the cache system and disk storage. For the most part, you access cached elements very frequently. For this reason you would need fast access speed which in turn means the cache must be smaller in size. With disk storage, you most likely won't access it very frequently, and thus you can forfeit your read/write speed for more storage space.
3. Proven elsewhere about caches, if you fetch an instruction from a block there is a high probability that the next instruction will be in that same block. Similarly to the last question, if you read a small block into the cache, you will be able to read from it very fast instead of having to do another slow fetch from the memory. This combination of periodic slow fetching from disk to gain, in general, higher performance is the goal of the blocking size in caches. This size is very important; if you make the cache size is too large, you lose your performance, but if it's too small you may not gain as much performance as you would like.
4. A monitor program is somewhat like a queuing system. that is, it accepts a 'batch' of jobs and runs them one at a time. In the 1940s - 1950s, people had to reserve times

on terminals to run their code since machines were so expensive and slow, and it was a large inconvenience to developers. The monitor system was a solution to this since it allowed for a solid work flow as people were able to drop off their code to be put into a batch to be run, and they could come back later and get the results. In conclusion, the transition from one person at a time whom had to reserve time slots to the ability to drop off your code without having to wait allowed for fluidity within the enterprise environment.

5. TBD

6. TBD