Daniel Crawford

2018-08-29

Operating System Concepts Assignment 1

1) Explain figure 1.11 on slide 26 in the slides. (Note this is not figure 1.1 on slide 5.)

This figure demonstrates how a program handles an interrupt using available memory space in the control stack. Also, this is not an illustration of a program in order, it is everything that is being held in main memory at a moment in time. Currently, the user’s program is at instruction N, and this particular instruction causes an interrupt. The processor is now holding the program counter value at N + 1, but the interrupt will cause the program counter to change. So the program needs to make some stack space available so it adjusts the stack pointer to point from T(the top of the stack) to T – M where M is how many registers need to be saved. Now, the program counter, and data within the program will be saved within the stack. After the data has been saved, the program counter is changed to Y which is the start of the interrupt service routine. Once this is completed, the program counter has now reached Y + L + 1 where L is the length of the interrupt service routine. Now the processor can return to the user program and restore all data/registers being used before from the control stack. This will return the program counter to N + 1 and the stack pointer to T. The program can now complete unless there is another instruction that would cause another interrupt.

2) Why is a memory hierarchy of different memory types used instead of only one kind of memory?

A couple different ideas for each type of memory. There are trade-offs being done when deciding how fast memory is, how large it is and how cost effective it is. If memory is very fast, then its significantly more expensive. Vise versa, if memory is very large, then it is very slow. Still, larger sources of memory will have lower cost per bit. If only one kind of memory was used, there would be two typical scenarios. The first being that you would have extremely expensive memory that transfers very quickly. This is assuming that speeds as fast as DRAM would be used for both volatile and non-volatile memory. Even so, latency would increase with significant amounts of fast access memory, ultimately ruining the speed that was promised. On the other hand, a computer could use only a solid-state drive or hard-drive. The benefit of this is that you have massive amounts of memory, but accessing this memory would be very slow. Thankfully, the memory hierarchy takes advantage of all types of memory’s strengths and combines them. A user would want to use the faster memory as much as possible, while limiting accesses to storage such as the hard-drive. The principal of locality of reference is used heavily in the memory hierarchy, so the data that is being used most frequently plus the data around it will be held in the fastest memory and if it is not found there, it would be called a miss. However, it would simply search in the next level on the memory hierarchy for the data such as the cache, and if not found there, it would move on to the main memory. Worst case, it will shift down to a storage device. Hit ratios however have been improved significantly, so the memory a program most often wants to access will likely be found in either the registers, cache or main memory. So generally, the memory hierarchy is used because programs and users want memory to be accessed as quick as possible, and users want lots of storage to keep their data. Because the memory hierarchy uses all kinds of memory instead of one, it can accomplish both of these things by placing the most needed data in the fastest memory, and other data that has not been used in a while in slower memory.

3) What is the effect of block size in cache design?

A block is a unit of data exchanged between the cache and the main memory. The idea of the block is to bring in data that is likely to be accessed based on the locality of reference. This, in turn, improves the hit ratio which provides faster access times on average. However, as these block sizes increase, the hit ratio will begin to decrease. If a block size is too large, the probability of using the newly fetched data from a block decreases and becomes less than the probability of reusing the data that has just been moved out of the cache to make room for the large block size. So in terms of cache design, a smaller block size can improve performance assuming that the block size is choosing the most frequently accessed data. To add to block size, the ability of a processor to replace a block which is least likely to be accessed again is very important to maintaining performance. This is called a replacement algorithm, and it chooses which block will be moved into main memory from the cache. The difficulty with this algorithm is it has to decide which block is least likely to be accessed, which is impossible to accomplish. However, an effective strategy is to determine the least frequently used block, which is detected by the least-recently-used algorithm. Assuming this works effectively, it will limit the amount of times a block needs to be moved out of the cache, which means that every block inside of the cache is being frequently accessed. Lastly, if a block’s contents are changed, then it needs to be written back to main memory before replacing it. The write policy will decide when a memory write operation will take place. There are two sides, one where the writing occurs after each alter, and another only when the block is replaced. The issue with the last one is that it will leave the main memory unused.

4) Explain how the monitor program improved on the early systems of the 1940's and 50's.

Before the monitor was created, a user would have to manage two fundamental steps: Scheduling and setup. This meant that the user would designate a time for a program to run, such as 1 hour or 45 minutes, and then if the program finished early it would wait until the time interval is complete. Even worse, each user had to completely setup each program carefully so that it works properly and if one step was imperfect, the user would have to go back to the beginning of the setup sequence. At this point, the user is completely interacting with the hardware, which is difficult. These operations are referred to as serial processing, and it means that the computer takes on a program one instruction at a time in series. This has been improved by including libraries of linker, loaders, debuggers, others as common software for all users to use. However, this system was still very flawed, and was improved upon by developing a batch OS. The batch OS contains a monitor, which takes away the users direct access to the hardware. Now, the monitor will place jobs together sequentially, and will communicate with the monitor once it is done so the monitor can load the next program. This is a massive improvement from what users were working with before, because now instead of designating how long a program will run, it will simply run until the program is complete, and then shift to the next program quickly. The batch OS therefore creates a significant improvement in efficiency by allowing the processor to be constantly working on the next job.

5) What is memory paging, and how can it facilitate virtual memory?

Memory paging is a concept that was introduced to represent a process as a fixed-sized block, pages. The issue with these blocks going into main memory is that it is difficult to make sure physical memory is not already taken up. Thanks to virtual memory, physical memory does not need to be accounted for when using pages. Virtual memory can be used to address memory for programs without considering the physical memory. Pages can be referred to using a virtual address, which consists of the page number and its offset. Afterwards, the page of a process can be located anywhere in main memory. The virtual address is also turned into a physical address, by using the virtual address and adjusting it using an offset. Virtual memory is capable of managing a reference to main memory, and if it does not exist, it will find the missing page and load it. This is referred to as a page fault, and while it is slow, it is still very effective with high hit ratios at finding memory fast.

6) How is a microkernel architecture different from a monolithic architecture?

A monolithic kernel included everything such as scheduling, a file system, networking, device drivers, and memory management. This kernel is implemented as a single process, where all of these things listed share the same address space. The microkernel on the other hand assigns only essential functions to the kernel. Now the services that were originally included in the kernel are now treated as a user mode process. These are called servers. Because of this design, servers can now be customized to a specific environment or applications requirements. Another benefit to the microkernel is that is simplifies implementation and works better for a distributed environment. Since some of these applications have been moved from the kernel, this allows for them to be better fitted for whatever environment they are being used on.