Operating System Questions

Matthew Johnson, Trinity Hall

May 6, 2002

1

1.1 Von Neumann Architecture

The Von Neumann Architecture is used in all modern computers - it is the basis for a stored program machine. At the simplest it defines a control unit, that converns how the machine runs, and an arithmetic unit. Commands and data are stored in memory, which the control unit can cause to be loaded into the arithmetic unit, and the result stored back in memory.

1.2 Registers

Registers use much faster, more expensive memory. This makes operations on the data in the registers quicker, since the registers are within the processor core itself, and data does not have to travel over the buses from main memory. Also, the fixed size and properties of the registers make various optimazations possible. Buffereing and cacheing tequniques can be used to improve the speed of loading data into the registers from memory.

1.3 Cache

Cache is used as an intermediate step between the memory and the registers. Since most memory accesses occur sequntially, it is possible to predictively load ahead of the program, and have the data already in the cache, rather than the memory. In modern computers there are typically 2 levels of cache, one on the die itself, and the other (larger) cache further away (logically) from the core, althought possibly still on the chip. Cache is made of faster memory technology than main memory, and this, coupled with the proximity to the processor core, improves the access speed.

2

2.1 Convert binary to decimal / hex / 2's complement decimal

$$
\begin{aligned}
10000001_2 &= 1 + 2^7 = 129_{10} \\
&= 81_{16} \\
&= 1 - 2^7 = -127_{2's} \\
10000000_2 &= 2^7 = 128_{10} \\
&= 80_{16} \\
&= -2^7 = -128_{2's} \\
00000001_2 &= 1_{10} = 01_{16} = 1_{2's} \\
00000000_2 &= 0_{10} = 00_{16} = 0_{2's}
\end{aligned}
$$

2.2 Subtractions in 2's complement

$11010 - 10000 = 01010$
$11010 - 01101 = 11010 + 10011 = (1)01101$
$000100 - 110000 = 000100 + 010000 = 010100$
$1010100 - 1010100 = 1010100 + 0101100 = (1)000000$

3

3.1 Condition codes

Condition codes / flags are bits in one of the registers in the CPU, which are used to indicate, possibly exceptional, conditions in the CPU that the programmer might want to be aware of. For example, if the result of a subtract operation is zero the zero flag might be set. This is useful in equality testing.

3.2 Big- / little-endian

$0x7BDCE823$ breaks down into the bytes: $7B\ DC\ E8\ 23$. This is the little-endian representation, which in binary is
$0111, 1011, 1101, 1100, 1110, 1000, 0010, 0011_2$. Big endian, the bytes are reversed, so we get $23\ E8\ DC\ 78$ and
$0010, 0011, 1110, 1000, 1101, 1100, 0111, 1001_2$ respectively

4

4.1 Buses

Buses are the method by which a computer transfers data between the various components. They are a number of communication wires, group into a logical collection, that operate simultaneously. They are usually synchronous, because it is much easier to build them them like that, and also there may be many devices connected to the same bus, and so a synchronous master-slave configuration can

4.2 Interrupts

Interrupts are used to help free up the CPU while it is waiting for I/O. The processor sends a request for (for example) data from the hard disk. The CPU can put that process on hold, and continue executing something else, until the HDD control sets an interrupt (one of the input lines to the CPU) which tells the CPU that the data is ready, and it can resume execution of the original process. On modern computers, the interrupt takes the form of only a few wires into the CPU. The interrupt details are encoded onto these, and when they are asserted, the CPU looks up in a table of memory addresses which bit of code should be called for that interrupt. It will then save the state of the registers, and jump to that memory location, to handle the interrupt.

4.3 Traps

Traps are software triggered interrupts, where as interrupts are generated by hardware, over dedicated interrupt lines into the processor. *Is this right? I don't believe its been covered in lectures using that terminology*

5

5.1 Dual-mode CPUs

The CPU has 2 seperate modes of execution, kernel & user modes, as a protection against malicious / badly written code. User mode programs can be restricted in what sorts of instructions they can run, and what memory they can access. Typically, they can only access the memory containing their own data, and jump to their own instructions. Other access are tightly controlled using Operating System APIs, where you can ask the Operating System to perform some controlled command for you, and it can make sure it is not run improperly

5.2 Restricted Instructions

Disable all interrupts	restricted	This command can lock the processor into a blocked state on all processes, since it cannot recieve interrupts to continue.
Read the clock	unrestricted	this is fine!
Set the clock	restricted	You don't want random user processes to be able to change the time. This can create unpredictable effects
Change the memory map	restricted	
Update base & limit registers	restricted	otherwise it is possible to read & write other processes' memory

6 Threads / Process & the DECSYSTEM-20

I don't think we have reached this point in the lectures yet, and from a quick flick through the notes, I can't see the answers