University of Stirling Computing Science and Mathematics

CSCU9A1 Autumn 2016

Tutorial 6

1. Modern computers are *digital* computers. What does that mean, and how do these compare with analog computers (what's an analog computer? See https://en.wikipedia.org/wiki/Analog_computer for some background information)? Why have digital computers become much more commonplace than analog computers?

Digital means that all the values are encoded as 0/1, whereas analog uses continuous values – usually voltages. Digital computers are programmable, and process near-continuous values by considering them as binary strings (various representations of numbers, fixed/floating point). Analog computers work by modeling some system as analog circuitry, but this necessarily introduced small errors, and thee mount up. Digital computers are exact (up to the quantization in digitization). But all this will need explained appropriately for the level of the class…
Digital computers are popular because (i) they can be mass produced without the need for precision components, because we have become extraordinarily good at producing digital circuitry, because we have become good at writing programs which can be used on many machines, …

2. What might the binary string 01100110 01101011 represent?

Lots of things. Hex 6667, letters AB (in ASCII), some sort of instruction, a number 26219 as a 16 bit integer, the address 262129 in memory, something inside a program…

3. Discuss the advantages and disadvantages of using flash memory (sometimes known as solid state drives) rather than hard disc memory for secondary storage. Which sorts of devices use which type of storage, and why?

Flash memory: non-volatile (indefinitely?), directly addressable (word by word), no moving parts,
HD memory: non-volatile (indefinitely?), addressable cylinder by cylinder, with delays caused by rotation etc., has moving parts and therefore more likely to be damaged by physical shock.
Note the use of caching in HD systems to improve performance (but that could be done in flash memory systems too).
Costs? Currently Flash is more expensive, Gbyte by Gbyte, but this may not remain the case.
Can flash memory we rewritten indefinitely? Seems to be the case. But not in reality -
Note that reading Flash memory is faster than writing it. (Why? Deeply involved quantum physics relating to tunneling… see http://en.wikipedia.org/wiki/Flash_memory for a reasonable start). Often got around by providing considerable RAM buffering on the device, and by using multiple Flash chips.

4. How many bits are needed to store a variable which represents something that might have 250000 different values? How many bits would be likely to be used if the code was programmed in high level language? Why are these different from each other?

   Round_upwards(\log_{2} (250000)) – i.e. the index of the lowest power of 2 above 250000, which is to say 262144 = 2^18. But one would end up using 32 or 64 in a high level language program, because that's what supported (a) by the CPU and (b) by the language.

5. A CPU can operate at 2,000,000,000 instructions/second. Each instruction is 4 bytes long. On average, 25% of the instructions need to read data (again, 4 bytes) from the memory, and, on average 10% of instructions need to write data to memory (again 4 bytes). Calculate the amount of data that needs to be transferred from the memory to the computer, and from the computer to the memory to keep the processor running at full speed.
   *4 * 2* 10^9 + (1/4) * 4 * 2 * 10^9 : reads for instructions and data*
   *(1/10) * 4 * 2 * 10^9 : writes. All in bytes.*

6. In general, it is not possible to achieve this data rate using a large (say 16 GByte) main memory. It is however, possible to achieve it using a smaller cache memory. Why does cache memory help?
   Cache catches data read once, and enables it to be read again very quickly. For instructions, almost all have been used before recently, so that cache really helps reduce traffic to/from the main memory. Similarly for writes – though that's a whole lot more complex, because there may be a read to recently written data that might get the old value.

7. Processors are often described as running at a certain number of Gigahertz (GHz). What does this mean? If I have a 2.8 GHz processor, will it actually run this many instructions/second? What factors affect the number of instructions/second that it can run?
   Processor speed is the number of the 9simplest, single cycle) instructions that it can operate on per second. Many instructions actually take a number of cycles. And keeping the processor supplied continuously with enough data also makes this hard to achieve in practice. There's other factors too, but ignore them (!) (Pipelining, for example)

8. The machine I am writing this on is a dual processor quad core machine. What does that mean?
   2 processors, each with a quad core. So there are (up to) 8 concurrent streams of instructions processing at any one time. Of course, it is generally difficult to achieve this: almost always, there's only a few actually executing.

