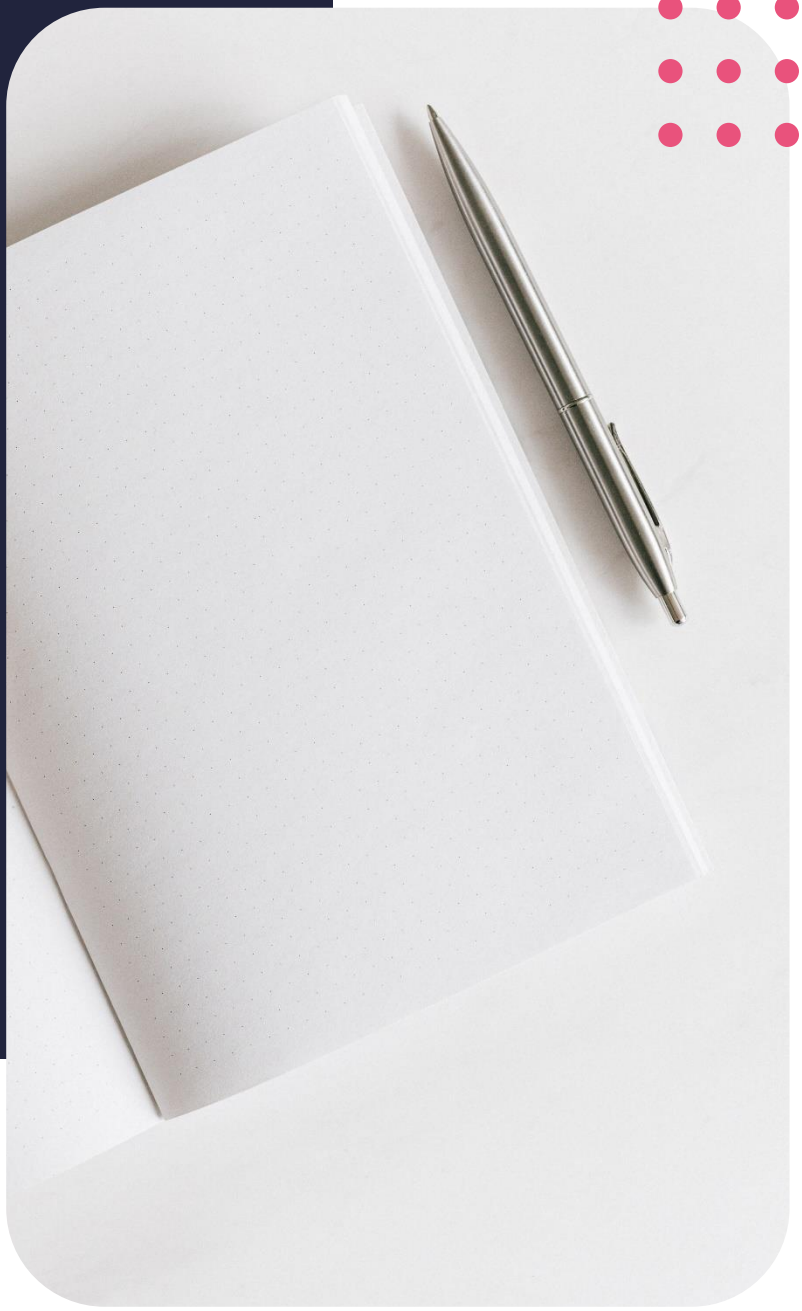
Diploma in Computer Science

Measuring computing power

Summary Notes

Contents



Lesson outcomes

By the end of this lesson, you should be able to:

- Know more about computing systems, including their architecture and the terms CISC and RISC
- Understand the different units in a binary system
- Explore the weaknesses of computers

Introduction

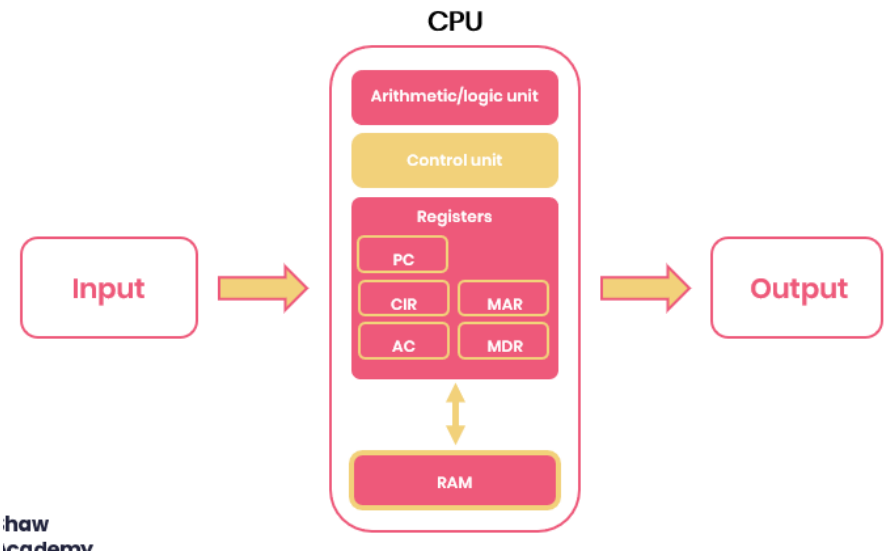
In this lesson, we will be taking a tour of the hardware that powers our entire digital lives. We will also explore the weaknesses of computers, because they are not invincible and all-knowing!

Computing systems

Structure of computers

Just like a house, a computer is built using a plan that shows where everything is placed. For a computer, there are variations in which computer systems can be built. But all computer systems have basic components such as input devices, storage devices, output devices and the CPU.

The **CPU** is not actually that big box that sits on your desk. It is a small chip that is resident inside that box. The CPU itself has some important components inside it: the control unit, arithmetic logic unit, registers, cache, buses, and clock.



All these components provide a spectacular level of system control. Let's look at each of these components:

- The **arithmetic and logic unit (ALU)** performs all the math and logic (decision) operations in the CPU.
- The **cache** is a small high-speed RAM built directly into the processor. It is used to hold data and instructions that the processor is likely to use. You may have seen this written on processor specifications as 3MB cache.

- A **register** is a small section of high-speed memory that is found inside the CPU. There are general purpose registers and special purpose registers.
- **Buses** are an internal high-speed connection. Just as they do on roads, buses carry memory addresses from the processor to memory, input, storage, or output devices. Data buses carry data while control buses carry control signals
- The **clock** keeps time. All the processes in the CPU must be timed precisely, to the tune of billions of pulses per second. This is measured in hertz, which are cycles per second. A cycle is when a single instruction has been processed. For example, if someone refers to a processor as a 2.5gigahertz processor, this means that the processor does 2.5 billion processing cycles per second!

Computer architecture

Computers are built according to the Von Neumann architecture. This architecture is based on the premise of storing program instructions in memory along with the instructions that operate on the data. This means that each computer can be built to perform tasks that are literally limited by the programmer's imagination. This design was proposed in 1945 by John Von Neumann to improve the special purpose computers of the day. In those days, you had to hard-wire a computer for a specific purpose. This was a tedious, error prone and time-consuming task.

The Von Neumann architecture uses five special registers:

- The program counter (PC) holds the next instruction to be fetched.
- The current instruction register (CIR) holds the address of the instruction that is currently being decoded and executed.
- The memory address register (MAR) holds the address of the current instruction that is to be fetched from memory, or the memory address to which data is to be transferred.
- The memory data register (MDR) holds the contents found at the address held in the MAR or data which is to be transferred to main memory.
- The accumulator (ACC) holds the data being processed and the results of processing.

With the coming of the Von Neumann architecture came programmable computers. A computer is able to process data because it has what is called an instruction set. An instruction set is the set of all possible commands that can be issued to the processor. These instructions enhance the capabilities of the processor in defined contexts. Instruction sets are commands that allow a program to tell the processor to switch relevant transistors on or off to perform an operation. These evolved and ended up as two categories: CISC and RISC. CISC stands for Complex Instruction Set Computing and RISC stands for Reduced Instruction Set Computing.

Complex Instruction Set Computing (CISC)

The complex instruction set computer has a primary goal of completing a task in as few lines of assembly code as possible. A CISC computer typically has microcode that allows it to do this. This microcode is a group of low-level instructions stored in fast memory and is also updateable. This means that the low-level instructions issued to the processor are shorter, but the processor still knows what to do. For example, for an addition operation, the computer uses the microcode to determine that it needs to move the contents of one register and put them into another register and store the result. This cannot be executed in one cycle, because it involves several steps. So, it is spread out over a number of cycles. This has the overall effect of improving system performance, but the hardware itself is very complex.

Reduced Instruction Set Computing (RISC)

The reduced instruction set computer aims at simple instructions that can be completed in 1 clock cycle. The programmer needs to code each individual step in order to perform the same operation we looked at for CISC above. This may sound inefficient, but RISC systems need less transistors to carry out the same task, which leaves more room to add registers. The instructions are executed in a uniform amount of time, so they can be staggered using a process called pipelining.

CISC vs RISC

The easiest way to compare CISC and RISC is to compare a child and an adult. When you give an adult an instruction, such as 'Do the laundry', an adult will immediately know that they have to sort the clothes, put them in the washing machine, add detergent, turn the machine on etc. For a child, this would be an overwhelming instruction, which would have to be given step by step. In this case, CISC would be the adult, and the child would be RISC.

The RISC does have several advantages. It has simple, standardised instructions, which means the programmer has less hassles when programming RISC systems, as the compiler does most of the hard work.

RISC does have drawbacks too, such as needing more ram, which can cause bottlenecks if the ram is limited.

CISC on the other hand, uses less ram and has the ability to add more instruction sets, which makes it more flexible. Also, the microcode can be extended to add more features to the instruction set.

The binary system

Defining a 'bit' and a 'byte' (and 'nibble' and 'word')

Digital computing at a very basic level is just a bunch of switches that are either on or off depending on the machine state. The "ON" state is represented by a 1 and the "OFF" state is represented by a 0. This forms the basis of all modern computing. All your pictures, music, drawings, robot movements, and everything else you can think of is, at the very basic level, just a whole horde of ones and zeros! Each one of those states is called a **'bit'**, and that system of numbering is called the binary system.

The binary system is a system of two digits; no matter how many you add or multiply, the sum will never go above 2. From 0, add 1 and it becomes 1, but if we add another 1, we have a problem - we can't write the number 2!

Remember from first grade Maths, when you ran out of numbers, you added another column and started counting from zero, like what we do when we count from 8, 9, then add another column and write 10.

So, we now write 1 0 (pronounced one zero). From 1 0 we add another 1, and it becomes 1 1, add another 1 and it becomes 1 0 0. The sequence ultimately becomes 0,1,10,11,100,101,111,1000 etc.

The problem is that there isn't much information that we can represent using 2 bits. There was a lot of chaos back in the day, because computers were using varied numbers of bits to represent characters. Somewhere along the line, powers of 2 slowly became the standard, so computers started using 8 bits to represent data. This became a standard unit, known as a **byte**.

By today's standards, a byte is a very small unit of data. It can represent only one dot on an image. In comparison, a 4k TV has 8 294 400 pixels! We need a way to represent this in a readable manner. This is where we use larger units of measurement. Unlike other units of data that we are used to, such as kilograms, 1 kilobyte is not 1000 kilobytes.

A byte is an addressable area, and these come in powers of 2. There are also units such as a **nibble**, which is 4 bits and a **word**, which is 16 bits.

Storing data

When you are going to store something in memory, you need to tell the storage device what memory location to put it in. That is done by providing a memory address (or a set of addresses). Since addresses are also provided in binary, every time you increase the number of address bits, you double the number of possible storage locations. So generally, new storage devices must be at least 2x as large as the previous largest one. If they aren't, there will be potential memory locations that are not actually addressable.

For example, let's say your old storage device had 4 address lines. This means that it can store data in 16 different locations (24 = 16). If you add one address line, it can now store data in 32 different locations. Making a storage device that could only hold 25 different pieces of data would mean that there were 7 addresses that were not usable. If nothing else, that means some complexity must be added to the controller so that if you try to store something in one of those 7 locations an error gets sent back. Otherwise, one of the other locations will be incorrectly overwritten and/or data will be lost.

Since this is the natural progression of things in binary addressing, people rarely have made storage devices that didn't work on powers of 2. It's just asking for trouble, and it has become a de-facto standard now. DVDs are one of the exceptions of this rule, as the information that is written to the disk may not be a power of 2. This is compensated for by the fact that the drive controllers are fairly complex and can decode the data. For most other controllers, however, things are confined to powers of 2.

Weaknesses of computers

As much artificial intelligence has advanced, humans are still better than computes at certain tasks.

Emotions

The one problem that is still puzzling computer scientists is getting computers to show emotion. This is a branch of artificial intelligence known as ‘affective computing’. Experts in this field are busy trying to get computers to analyse people’s expressions and understand them. Humans, from a very early age, can distinguish and interpret facial expressions intuitively. Currently, even the fastest computers have a hard time making out what a person’s expressions mean. Humans can also make decisions based on emotion. It is very hard for a computer to, for example, give a child a less complicated game because he looks nervous.

Creativity

Computers also have a hard time creating something completely new. This is a new research field, though, and there are some bizarre things happening, for example, Google created AI who can ‘dream’. The machine was given a blank slate of white noise and it generated some zany images such as a pig-snail and camel-bird! The process of creating images from nowhere is called ‘inceptionism’.

Growth

Computers have a hard time improving themselves. Typically, programmers have to keep coming up with better software for computers. Humans grow, from the few things we knew at birth in our tiny bodies to the vast amount of knowledge and strength we gather as we grow. The only way for a computer to improve is through the hard work of engineers and scientists. There is progress, however, in artificial intelligence, where a computer can be presented with simple tasks and it can learn from them to be able to perform more complex tasks.

Decision making

Computers are only as good as the information they’re given. All their processing is based on what the programmer said the computer should do. When it comes to independent thought and decision making, computers instantly appear dumb. They rely on some form of input from somewhere to make a decision.

Human thought

Human thought , which is taken for granted in our world, is near impossible in the world of computers. Phones that come with voice assistants, such as Alexa, Bixby, Cortana, Google assistant and Siri, sound very intelligent when they tell you who sang a particular song or remind you to turn the lights off, but they cannot hold a deep conversation.

Conventional communication traits such as continuing from the previous sentence, changing the topic, offering opinions and using figurative language are very challenging for computers.

Judgement

Some decisions rely on human judgement and reasoning, for example, when you decide to paint your house yellow or green or purple. Computers are particularly challenged by situations that have too many random variables and require a level of judgement, like driving.

Processing

In natural language processing, a single word can be pronounced differently by several people. This presents a challenge for the computer. Fuzzy logic is a branch of Computer Science that attempts to solve this problem by allowing the computer to reason without fitting into exact categories.

Computers use a cycle known as the fetch-decode-execute cycle, which does exactly what it says. Instructions are fetched from RAM, then the CPU "makes sense" of these instructions by decoding them. the CPU then carries out the instruction in the execute stage. A complete computer cycle includes all three stages. Remember the clock that we discussed earlier? The cycles are perfectly timed according to the clock. A computer's speed is measured by the number of cycles that can be completed in a second. Modern computers can complete billions of these cycles in a second!

Conclusion

In this lesson, we have looked at computing systems more closely, particularly the binary system. We also examined the various weaknesses of computers compared to humans. However, clearly there are huge advantages to the rising power of computers and significant advances in artificial intelligence.

References

- 2 Von Neumann Architecture 2.1 INTRODUCTION. (2012). [online] Available at: http://www2.cs.siu.edu/~cs401/Textbook/ch2.pdf.
- Teach Computer Science. (2020). RISC & CISC Processors | What, Characteristics & Advantages. [online] Available at: https://teachcomputerscience.com/risc-and-cisc-processors/
- Dinesh Thakur (2014). What are the Basic Computer Components? - Computer Notes. [online] Computer Notes. Available at: https://ecomputernotes.com/fundamental/introduction-to-computer/computer-components
- Computer Science GCSE GURU. (2019). Von Neumann Architecture - Computer Science GCSE GURU. [online] Available at: https://www.computerscience.gcse.guru/theory/von-neumann-architecture
- Wolf, M. (2017). Instruction Sets. Computers as Components, [online] pp.55–98. Available at: https://www.sciencedirect.com/science/article/pii/B9780128053874000029
- BBC Bitesize. (2020). Factors affecting CPU performance - Computer systems - AQA - GCSE Computer Science Revision - AQA - BBC Bitesize. [online] Available at: https://www.bbc.co.uk/bitesize/guides/z7qqmsg/revision/5
- Stanford.edu. (2020). RISC vs. CISC. [online] Available at: https://cs.stanford.edu/people/eroberts/courses/soco/projects/risc/risccisc/
- Thornton, S. (2020). RISC vs. CISC Architectures: Which one is better? [online] Microcontrollertips.com. Available at: https://www.microcontrollertips.com/risc-vs-cisc-architectures-one-better/
- What is a Byte? - Definition & Measurements Video (2020). What is a Byte? - Definition & Measurements | Study.com. [online] Study.com. Available at: https://study.com/academy/lesson/what-is-a-byte-definition-measurements.html
- Google.com. (2019). Inceptionism: Going deeper into Neural Networks. [online] Available at: https://photos.google.com/share/AF1QipPX0SCl7OzWilt9LnuQliattX4OUCj_8EP65_cTVnBmS1jnYgsGQAieQUc1VQWdgQ?key=aVBxWjhwSzg2RjJWLWRuVFBBZEN1d205bUdEMnhB
- Complex Operational Decision Making in Networked Systems of Humans and Machines. (2014). [online] Washington, D.C.: National Academies Press. Available at: https://www.nap.edu/read/18844/chapter/4#12
- Andrew, E. (2015). Google’s AI Can Dream, and Here’s What it Looks Like. [online] IFLScience. Available at: https://www.iflscience.com/technology/artificial-intelligence-dreams/