Topic: 1.4.1 CPU architecture

The Von Neumann Architecture

The earliest computing machines had fixed programs. For example, a desk calculator (in principle) is a fixed program computer. It can do basic mathematics, but it cannot be used as a word processor or a gaming console. Changing the program of a fixed-program machine requires re-wiring, re-structuring, or re-designing the machine. The earliest computers were not so much "programmed" as they were "designed". "Reprogramming", when it was possible at all, was a laborious process, starting with flowcharts and paper notes, followed by detailed engineering designs, and then the often-arduous process of physically re-wiring and re-building the machine. It could take up to 3 weeks to set up a program on ENIAC (a computer of 1940s) and get it working.

The phrase Von Neumann architecture derives from a paper written by computer scientist John von Neumann in1945. This describes a design architecture for an electronic digital computer with subdivisions of a central arithmetic part, a central control part, a memory to store both data and instructions, external storage, and input and output mechanisms. The meaning of the phrase has evolved to mean "**A stored-program computer**". A stored-program digital computer is one that keeps its programmed instructions, as well as its data, in read-write, random-access memory (RAM). So John Von Neumann introduced the idea of the stored program. Previously data and programs were stored in separate memories. Von Neumann realized that data and programs are somewhat of the same type and can, therefore, use the same memory. On a large scale, the ability to treat instructions as data is what makes assemblers, compilers and other automated programming tools possible. One can "write programs which write programs". This led to the introduction of compilers which accepted high level language source code as input and produced binary code as output.

Topic: 1.4.1 CPU architecture

The Von Neumann architecture uses a single processor which follows a linear sequence of **fetch-decode-execute**. In order to do this, the processor has to use some special registers, which are discrete memory locations with special purposes attached. These are:

Register	Meaning
PC	Program Counter
CIR	Current Instruction Register
MAR	Memory Address Register
MDR	Memory Data Register
Status	A General Purpose Register
IX	Index Register
Accumulator	Holds results

- **Program counter (PC):** Keeps track of where to find the next instruction so that a copy of the instruction can be placed in the current instruction register. Sometimes the program counter is called the Sequence Control Register (SCR) as it controls the sequence in which instructions are executed.
- **Current instruction register (CIR):** Holds the instruction that is to be executed.
- **Memory address register (MAR):** Used to hold the memory address that contains either the next piece of data or an instruction that is to be used.
- **Memory data register (MDR):** Acts like a buffer and holds anything that is copied from the memory ready for the processor to use it.
- **Index register (IR):** A microprocessor register used for modifying operand addresses during the run of a program, typically for doing vector/array operations. Index registers are used for a special kind of indirect addressing where an immediate constant (i.e. which is part of the instruction itself) is added to the contents of the index register to form the address to the actual operand or data.

Topic: 1.4.1 CPU architecture

- **Status Register (General purpose register):** It sits inside ALU and has following three purposes/uses during a programs' instructions execution.
  It holds:
  1) Results of comparisons to decide later for action.
  2) Interim results of arithmetic performed
  3) Any errors occurred during the arithmetic operations, like overflow etc.

The central processor contains the **arithmetic-logic unit** (also known as the arithmetic unit) and the **control unit**. The arithmetic-logic unit (ALU) is where data is processed. This involves arithmetic and logical operations. Arithmetic operations are those that add and subtract numbers, and so on. Logical operations involve comparing binary patterns and making decisions.
The control unit fetches instructions from memory, decodes them and synchronizes the operations before sending signals to other parts of the computer.

The accumulator is in the arithmetic unit, the Program counter (PC) and the Instruction registers (CIR) are in the **control unit** and the memory data register (MDR) and memory address register (MAR) are in the **processor**.

Topic: 1.4.1 CPU architecture

A typical layout is shown in Fig. a.1 which also shows the data paths.

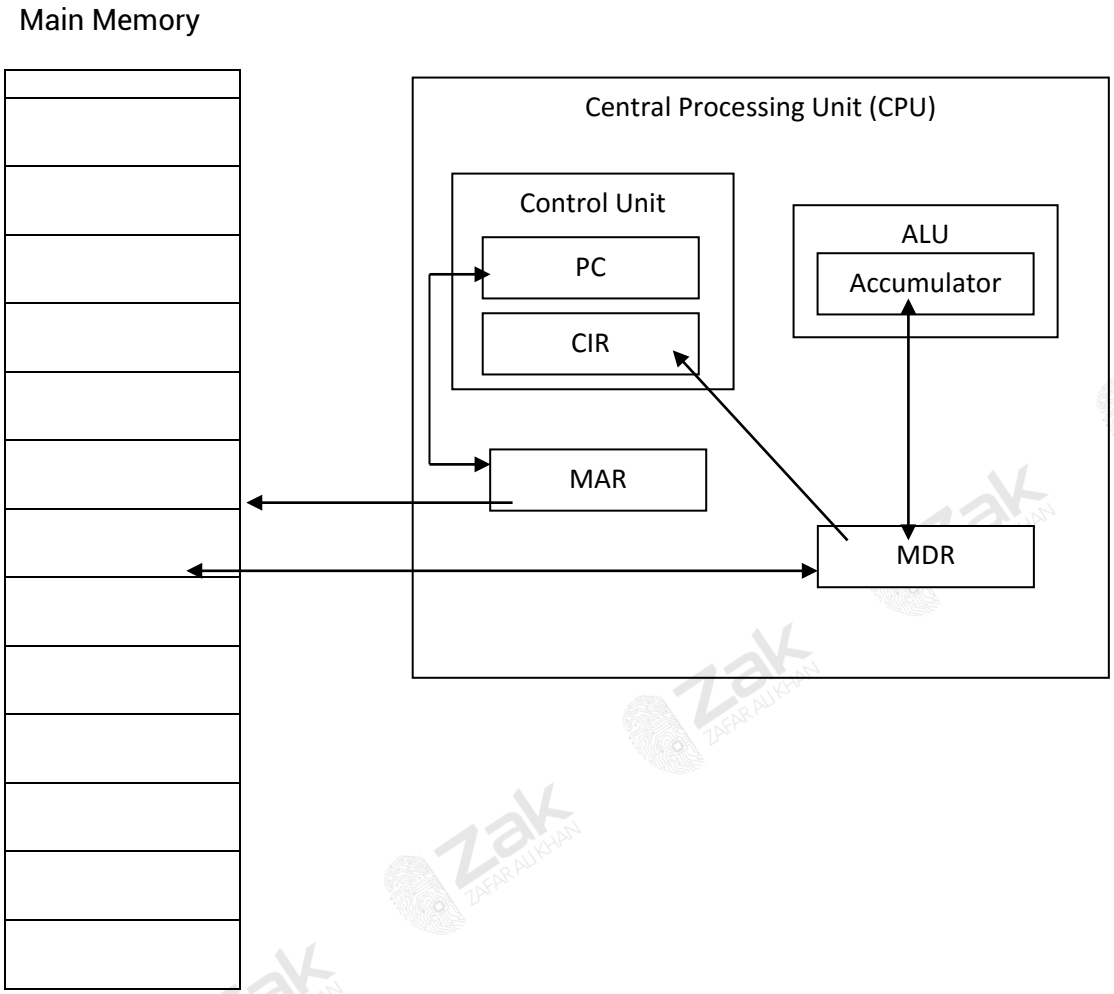


Fig a.1

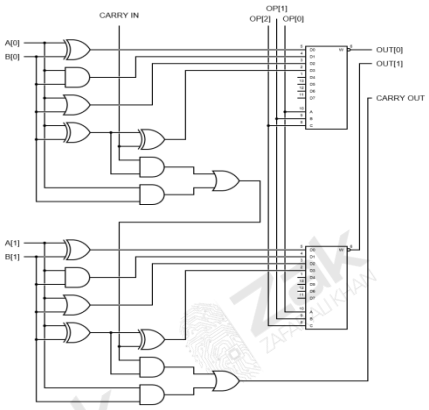
Topic: 1.4.1 CPU architecture

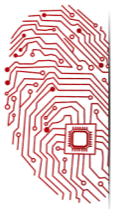
Arithmetic logic unit

A simple example of an arithmetic logic unit (2-bit ALU) that does AND, OR, XOR, and addition

The Arithmetic Logic Unit or the ALU is a digital circuit that performs arithmetic and logical operations. Where arithmetic operations include things such as ADDITION and SUBTRACTION and the logical operations include things such as AND, OR, NOT.

The ALU is a fundamental building block in the central processing unit (CPU) of a computer and without it the computer wouldn't be able to calculate anything!





Topic: 1.4.1 CPU architecture

Some examples of assembly code instructions that would use the ALU are as follows (not all processors will have all these instructions):

Instruction		Explanation
Op Code	Operand	
LDM	#n	Immediate addressing. Load the number n to ACC
LDD	<address>	Direct addressing. Load the contents of the given address to ACC
LDI	<address>	Indirect addressing. The address to be used is at the given address. Load the contents of this second address to ACC
LDX	<address>	Indexed addressing. Form the address from <address> + the contents of the index register. Copy the contents of this calculated address to ACC
LDR	#n	Immediate addressing. Load the number n to IX
STO	<address>	Store the contents of ACC at the given address
ADD	<address>	Add the contents of the given address to the ACC
INC	<register>	Add 1 to the contents of the register (ACC or IX)
DEC	<register>	Subtract 1 from the contents of the register (ACC or IX)
JMP	<address>	Jump to the given address
CMP	<address>	Compare the contents of ACC with the contents of <address>
CMP	#n	Compare the contents of ACC with number n
JPE	<address>	Following a compare instruction, jump to <address> if the compare was True
JPN	<address>	Following a compare instruction, jump to <address> if the compare was False
IN		Key in a character and store its ASCII value in ACC
OUT		Output to the screen the character whose ASCII value is stored in ACC
END		Return control to the operating system

Topic: 1.4.1 CPU architecture

Control unit - sits inside the CPU and coordinates the input and output devices of a computer system. It coordinates the fetching of program code from main memory to the CPU and directs the operation of the other processor components by providing timing and control signals.

Processor clock-is a timing device connected to the processor that synchronizes when the fetch, decode execute cycle runs

Your computer might contain several clocks that regulate different things. The clock we are going to look at here will keep the processor in line. It will send the processor a signal at regular times telling it to start the fetch decode execute routine.

Lights flash at *frequencies of* 0.5 Hz, 1.0 Hz and 2.0 Hz, where x Hz means x flashes per second. (Hz= Hertz)

Clock speed - The number of cycles that are performed by the CPU per second

Clock speed is measured in Hertz, which means 'per second'. You have probably heard of clock speeds such as 1 MHz, this means 1,000,000 cycles per second and potentially a million calculations. A computer of speed 3.4 GHz means it might be capable of processing 3,400,000,000 instructions per second! However it isn't as simple at that, some processors can perform more than one calculation on each clock cycle, and processors from different manufacturers and using different architecture are often difficult to compare. Also with the increase in multi-core processors such as the PS3 (7 cores) and the Xbox 360 (3 cores) there might be times where the clock might be ticking but there is nothing for the processor to calculate, the processor will then sit idle.

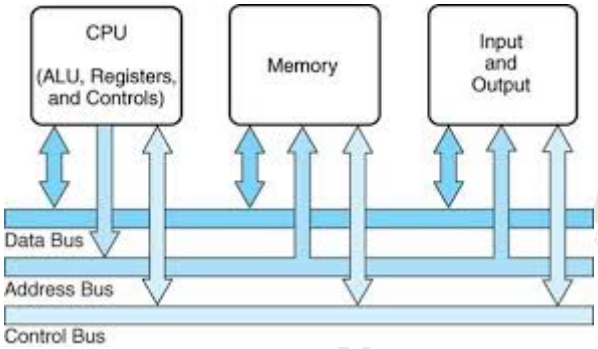
Topic: 1.4.1 CPU architecture

Data, Address and Control Buses

A **bus** is a set of parallel wires connecting two or more components of the computer.

The CPU is connected to main memory by three separate buses. When the CPU wishes to access a particular memory location, it sends this address to memory on the address bus. The data in that location is then returned to the CPU on the data bus. Control signals are sent along the control bus.

In Figure below, you can see that data, address and control buses connect the processor, memory and I/O controllers. These are all system buses. Each bus is a shared transmission medium, so that only one device can transmit along a bus at any one time.



Busses: These are mediums (wires) connecting the microprocessor with the main memory and other parts of the system. There are three types of the busses, i.e. Control bus, address bus, and data bus.

Control bus: This is used by the control unit to communicate and transfer signals to and from the internal and external devices of the system. It is used to minimise the communication lines required for the communication. *This is a bi-directional bus that is comprised of interrupt line, read/write signals & status line*. It makes sure that data when transferred is on one channel, from one device and in one direction only to avoid collision and loss of data.

Address Bus: This bus is used to carry the address from where data needs to be fetched or placed in main memory. This is a one directional bus whose width (number of wires) defines the range of addresses a microprocessor can reach to. For example an address bus of 12 bits can reach to 2^{12}=4096 (4k) addresses whereas 32 bit bus can reach 2^{32}=4GB addresses.

Data Bus: A bi-directional bus to access data to and from memory address mentioned in MAR.

Topic: 1.4.1 CPU architecture

Address bus: When the processor wishes to read a word (say 8, 16, or 32 bits) of data from memory, it first puts the address of the desired word on the address bus. The width of the address bus determines the maximum possible memory capacity of the system. For example, if the address bus consisted of only 8 lines, then the maximum address it could transmit would be (in binary) 11111111 or 255 – giving a maximum memory capacity of 256 (including address 0). A more realistic minimum bus width would be 20 lines, giving a memory capacity of 2^{20}, i.e. 1Mb.

Increasing performance

If we want to increase the performance of our computer, we can try several things

- Increasing the clock speed
- Adjusting word length
- Increasing bus widths

For each different method, we are going to look at these old games consoles to see how performance increase was achieved:

System	Year	Speed	Word size	Notes
NES	1983	1.79 MHz	8 bit	
SNES	1990	3.58 MHz	16 bit	
Nintendo 64	1996	93.75 MHz	64 bit	
GameCube	2001	486 MHz	128 bit	cooling fan introduced

Topic: 1.4.1 CPU architecture

Clock speed - The number of cycles that are performed by the CPU per second

The most obvious way to increase the speed of a computer would be to increase the speed of the computer clock. With a faster clock speed the processor would be forced to perform more instructions per second.

But what is to stop us increasing the clock speed as much as we want? If you study Physics you might already know this, but the problem with increased clock speed is that an increased current will have to flow through the circuits. The more current that flows through a circuit, the more heat will be generated. You might notice that a laptop will get hot or even your mobile phone when you are doing a heavy task like playing a game. The faster the clock speed, the hotter the processor runs.

To counter this issue, computer scientists have come up with more advance chip designs and introduced heat sinks, fans, and even liquid cooling into computers. If a processor runs too hot it can burn out!

Peripherals

Input/output devices are used by the system to get information in and out, as they are not internal but are connected to the CPU, we refer to them as **peripherals** (your hands are peripheral to your torso). We cover the specific ones you need to learn in chapter 1.3.1, but for the moment you need to know the fundamental difference:

- Input Devices - used to feed information to the system. E.g. Keyboard
- Output Devices - An output device is used to display the results of data processing on the input data. E.g. Visual Display Unit (VDU)

If you look at the Von Neumann Architecture notice that it doesn't mention Keyboard or display, this was a very well thought out action, as you don't want to force every computer to have a keyboard (think about a games console) or a VDU (some devices such as MP3 players don't have a screen like the iPod Shuffle). However, some computer architecture does include specific I/O controllers:

I/O controllers- an electronic circuit that connects to a system bus and an I/O device; it provides the correct voltages and currents for the system bus and the I/O device. Examples would include:

- keyboard controller, attached to a keyboard
- disk controller for a Hard Disk (Hard Disks are not main memory!)
- video display controller, attaching a video display unit (monitor)

Topic: 1.4.1 CPU architecture

I/O port - a complementary method of performing input/output between the CPU and peripheral devices in a computer. This allows I/O devices to be connected to the CPU without having to have specialized hardware for each one. Think about the USB port on your computer, you can connect Keyboards, Mice, Game pads, Cameras, Phones, etc. and they all connect using the same type of port!