But how does a Computer work? The CPU

Jaden Furtado

Abstract

Everyone knows how a CPU works, or do they? Since I've seen numerous students understand how DL and AI models work, but can't answer how a basic interrupt works, I feel the need to put this literature out there. In this paper we explore how a CPU works at a very high level along with the components used. This is a compilation of work from various sources.

1. The Von Neuman architecture

The von Neumann architecture, also known as the von Neumann model or Princeton architecture, is a computer architecture based on a 1945 description by John von Neumann, and by others, in the First Draft of a Report on the EDVAC.[1] The document describes a design architecture for an electronic digital computer with these components:

- A processing unit with both an arithmetic logic unit and processor registers
- A control unit that includes an instruction register and a program counter
- Memory that stores data and instructions
- External mass storage
- Input and output mechanisms

2. But what is a CPU?

"The Central Processing Unit (CPU) is the primary component of a computer that acts as its "control center." The CPU, also referred to as the "central" or "main" processor, is a complex set of electronic circuitry that runs the machine's operating system and apps. The CPU interprets, processes and executes instructions, most often from the hardware and software programs running on the device.

The CPU performs arithmetic, logic, and other operations to transform data input into more usable information output. While the CPU must contain at least one processing core, many contain multiple cores. A server with two hexa-core CPUs, for example, will have a total of 12 processors."[1]

[1] https://www.arm.com/glossary/cpu

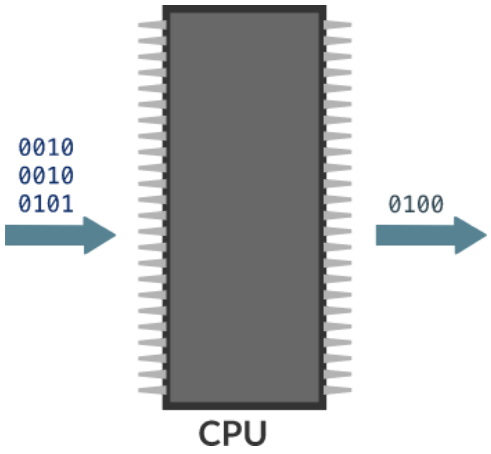


Fig 1 The is a simple architecture of a CPU

1.1 The main functions that a CPU needs to complete

Data processing: Data may take a wide variety of forms, and the range of processing requirements is broad. However, we shall see that there are only a few fundamental methods or types of data processing.

Data storage: Even if the computer is processing data on the fly (i.e., data come in and get processed, and the results go out immediately), the computer must temporarily store at least those pieces of data that are being worked on at any given moment. Thus, there is at least a short-term data storage function. Equally important, the computer performs a long-term data storage function. Files of data are stored on the computer for subsequent retrieval and update.

Data movement: The computer's operating environment consists of devices that serve as either sources or destinations of data. When data are received from or delivered to a device that is directly connected to the computer, the process is known as input–output (I/O), and the device is referred to as a peripheral. When data are moved over longer distances, to or from a remote device, the process is known as data communications.

Control: Within the computer, a control unit manages the computer's resources and orchestrates the performance of its functional parts in response to instructions.

2. The Components of CPU

We could divide a CPU into two main components, i.e. the control unit and the processing unit.

2.1 Control Unit—CU

Control unit CU is the part of CPU that helps orchestrate the execution of instructions. It tells what to do. According

to the instruction, it helps activate the wires connecting CPU to different other parts of computer including the ALU. Control unit is the first component of CPU to receive the instruction for processing.

There are two types of control unit:

- hardwired control units.
- micro programmable (microprogrammed) control units.

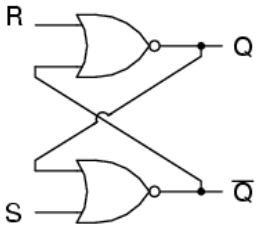
Hardwired control units are the hardware and needs the change in hardware to add modify it's working where as [2]microprogrammable control unit can be programmed to change its behavior. Hardwired CU are faster in processing instruction whereas microprogrammable as more flexible.

2.2 Arithmetic and logical unit—ALU

Arithmetic and logical unit ALU as name suggest does all the arithmetic and logical computations. ALU performs the operations like addition, subtraction. ALU consists of logic circuitry or logic gates which performs these operations.

Most logic gates take in two input and produces one output

Registers



S	R	Q	$\overline{Q}$
0	0	latch	latch
0	1	0	1
1	0	1	0
1	1	0	0

Fig 2. An SR latch and its truth table

CPU has registers to store the data of output. Sending to main memory(RAM) would be slow as it is the intermediate data. This data is send to other register which is connected by a **BUS**. A register can store instruction, output data, storage address or any kind of data.

2.2 Memory(RAM)

Ram is a collection of register arranged and compact together in an optimized way so that it can store a higher number of data. RAM(Random Access Memory) are volatile and it's data get's lost when we turn off the power. As RAM is a collection of register to read/write data a RAM takes input of 8bit address, data input for the actual data to be stored and finally read and write enabler which works as it is for the latches.

2.3 Instructions

Instruction is the granular level computation a computer can perform. There are various types of instruction a CPU can process.

Instructions include:

- Arithmetic such as **add** and **subtract**
- Logic instructions such as **and**, **or**, and **not**
- Data instructions such as **move**, **input**, **output**, **load**, and **store**
- Control Flow instructions such as **goto**, **if … goto**, **call**, and **return**
- Notify CPU that the program has ended **Halt**

Instruction are provided to computer using assembly language or are generated by compiler or are interpreted in some high level languages.

These instructions are hardwired inside CPU. ALU contains the arithmetic and logical where as the control flow are managed by CU.

In one **clock cycle** computers can perform one instruction but modern computers can perform more than one.

A group of instructions a computer can perform is called an **instruction set**.

2.4 CPU clock

2.4.1 Clock cycle

The speed of a computer is determined by its clock cycle. It is the number of **clock periods** per second a computer works on. A single clock cycles are very small like around 250 * 10 *-12 sec. Higher the clock cycle faster the processor is.

CPU clock cycle is measure in gHz(**Gigahertz**). 1gHz is equal to 10^9 Hz(**hertz**). A hertz means a second. So 1Gigahertz means 10^9 cycles per second.

The faster the clock cycle, the more instructions the **CPU** can execute.Clock cycle = 1/clock rateCPU Time = number of clock cycle / clock rate

This means to improve CPU time we can increase clock rate or decrease number of clock cycle by optimizing the instruction we provide to CPU. Some processor provide the ability to increase the clock cycle but since it is physical changes there might be over heating and even smokes/fires.

2.5 How does an instruction get executed

[2] https://www.freecodecamp.org/news/how-does-a-cpu-work/

Instructions are stored on the **RAM** in a sequential order. For a hypothetical CPU Instruction consists of **OP** code(operational code) and **memory or register address**.

There are two registers inside a Control Unit **Instruction register(IR)** which loads the OP code of the instruction and **Instruction address register** which loads the address of the current executing instruction. There are other registers inside a CPU which stores the value stored in the address of the last 4 bits of a instruction. Let's take an example of a set of instruction which adds two number. The following are the instructions along with there description:

STEP 1—LOAD_A 8:

The instruction is saved in RAM initially as let's say <1100 1000>. The first 4 bit is the op code. This determines the instruction. This instruction is **fetched** into the **IR** of the control unit. The instruction is **decode** to be load_A which means it needs to load the data in the address 1000 which is the last 4 bit of the instruction to register A.

STEP 2—LOAD_B 2

Similar to above this loads the the data in memory address 2 (0010) to CPU register B.

STEP 3—ADD B A

Now the next instruction is to add these two numbers. Here the CU tells ALU to perform the add operation and save the result back to register A.

STEP 4—STORE_A 23

This is a very simple set of instruction that helps add two numbers.

2.6 BUS

All the data between CPU, register, memory and IO devise are transferred via bus. To load the data to memory that it has just added, the CPU puts the memory address to address bus and the result of the sum to data bus and enables the right signal in control bus. In this way the data is loaded to memory with the help of the bus.

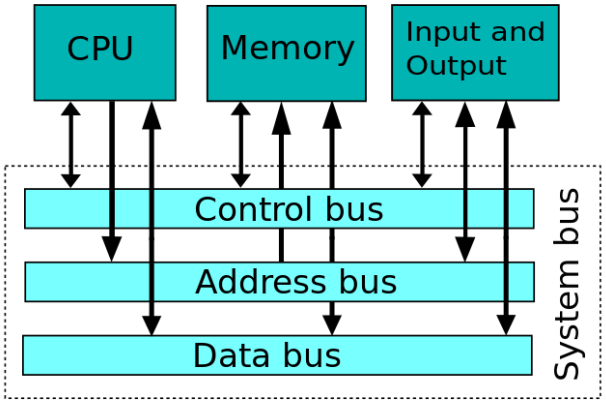


Fig 3. System Bus

2.7 Cache

CPU also has a mechanism to prefetch the instruction to its cached. As we know there are millions of instruction a processor can complete within a second. This means that there will be more time spent in fetching the instruction from RAM than executing them. So the CPU cache prefetches some of the instruction and also data so that the execution gets fast. If the data in cache and operating memory is different the data is marked as a **dirty bit**.

3. Instruction pipelining

Modern CPU uses **Instruction pipelining** for parallelization in instruction execution. Fetch, Decode, Execute. When one instruction is in decode phase the CPU can process another instruction for fetch phase.

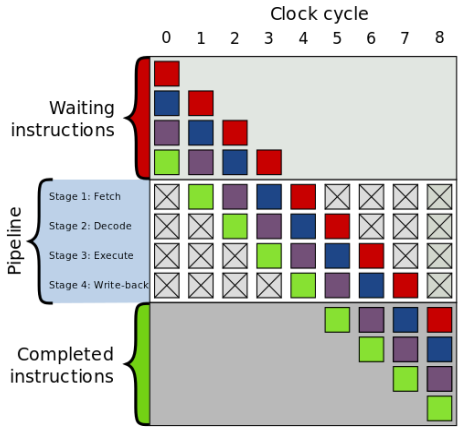


Fig 4. Instruction Pipelining

This has one problem when one instruction is dependent on another. So processors execute the instruction that are not dependent and in different order.

4. Future work

Thus far, we have explored the various components of a CPU along with a cursory view of how they work. However, while this is good from a mile high view perspective, we need to take a deep dive into the components, how they work, how they are built, how they are programmed, etc

References

[1] von Neumann, John (1945), *First Draft of a Report on the EDVAC* (PDF), archived from the original (PDF) on March 14, 2013, retrieved August 24, 2011.

[2] How does a CPU work?