previous sub-module | module | course | next sub-module

3.2. Internal Hardware

Processor (and more)

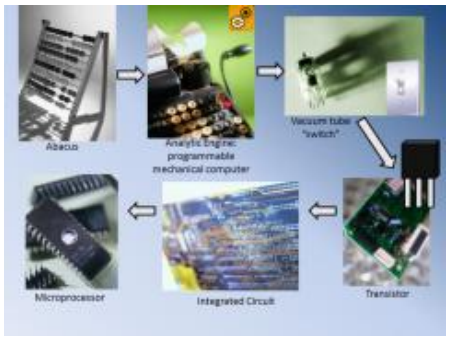
History of Computing (Part 1)

I have indicated several times that a computer is a very simple machine at its lowest level. More precisely, it is made up of simple units. (That's like saying a precision mechanical watch is made up of a bunch of gears, but it *can* help in understanding the device.) Since most won't be interested, we traverse the history (development) of computers in a paragraph or three. Those that are more interested can follow the links provided, such as the history of computer hardware.

Mechanical computers have been around for a long time. Programmable mechanical computers (general purpose computers) were first described in the 1800's by Charles Babbage (see analytic engine). In the early 1900's vacuum tubes replaced relays as a means of creating logic gates, leading to the earliest electronic computers (right around World War 2, in the 40's). The first silicon transistor, a smaller and more dependable switch, was developed in 1954.

The transistor is the basis for much of our electronics today. (A key development for me was the hand-held device of my youth, the transistor radio.) In 1958, Jack Kilby (a Nobel prize winner) demonstrated the first working integrated circuit — what we typically call a "card". This was followed by successively smaller and smaller designs for integrated circuits. A microprocessor is an integrated circuit on a chip.

Curta hand-held mechanical calculator, c.1948
By Larry McElhiney (Created this image in Indianapolis, IN) [CC-BY-SA-2.5], via Wikimedia Commons



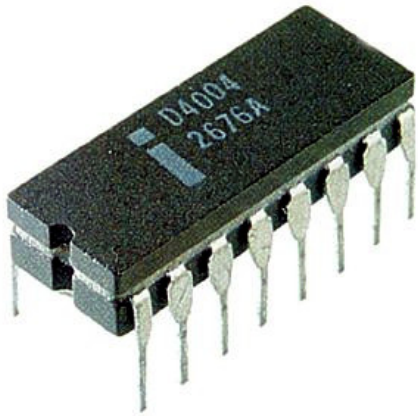
Computer history at a glance
By Paul Mullins: constructed

Microprocessor [src]

A microprocessor incorporates the functions of a computer's central processing unit (CPU) on a single integrated circuit (IC, or microchip). It is a multipurpose, programmable, clock-driven, register based electronic device that accepts binary data as input, processes it according to instructions stored in its memory, and provides results as output.

That concise description includes only one term we haven't covered and one fairly important concept. A *register* is local, temporary storage. When the processor needs to add a couple of numbers, the numbers need to be stored locally, as does the answer. They are stored in registers.

The *clock* acts a metronome that synchronizes (almost) all of the activity on a computer. The faster the clock ticks, the faster things occur. This is a key specification when buying (or upgrading) a system. Clock speed is measured in



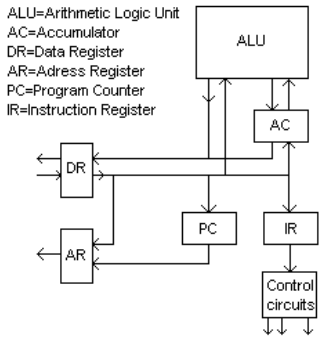
Intel 4004, the first general-purpose, commercial microprocessor
By LucaDetomi at it.wikipedia (Transfered from it.wikipedia) [GFDL], from Wikimedia Commons

Hertz (cycles per second), so a bigger number is better. Current PC clocks run in the GHz (gigahertz or billions of cycles per second) range, while hand-held devices tend to be in the MHz (megahertz or millions of cycles per second) range.

Mechanical metronome
By Vincent Quach (Invincible) (Own work) [CC-BY-SA-3.0], via Wikimedia Commons

Central Processing Unit (CPU)

In the previous module we talked about how the computer processes data. The processing occurs in the CPU (Central Processing Unit). To simplify discussion of the CPU, we break all of its functionality down into two basic groups: the control unit and the arithmetic & logic unit. In the diagram to the right, all of the other elements shown are **registers**.



CPU architecture
public domain image

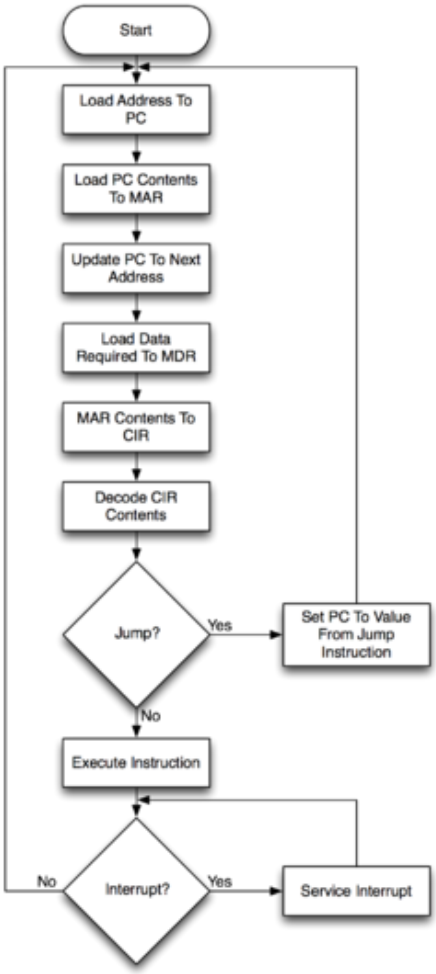
Control Unit [src]

The control unit (often called a control system or central controller) manages the computer's various components; it reads and interprets (decodes) the program instructions, transforming them into a series of control signals which activate other parts of the computer.

A key component common to all CPUs is the program counter, a special memory cell (register) that keeps track of which location in memory the next instruction is to be read from.

The control system's function is as follows – note that this is a simplified description, and some of these steps may be performed concurrently or in a different order depending on the type of CPU:

1. Read the code for the next instruction from the cell indicated by the program counter.
2. Decode the numerical code for the instruction into a set of commands or signals for each of the other systems.
3. Increment the program counter so it points to the next instruction.



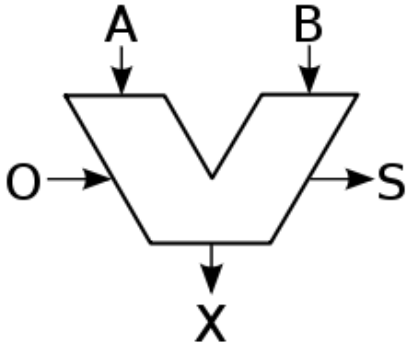
Fetch Execute Cycle
By Ratbum (Own work) [CC-BY-3.0], via Wikimedia Commons

4. Read whatever data the instruction requires from cells in memory (or perhaps from an input device). The location of this required data is typically stored within the instruction code.
5. Provide the necessary data to an ALU (see below) or register.
6. If the instruction requires an ALU or specialized hardware to complete, instruct the hardware to perform the requested operation.
7. Write the result from the ALU back to a memory location or to a register or perhaps an output device.
8. Jump back to step (1).

This description includes the fetch-decode-execute instruction cycle that is basic to the operation of the computer.

ALU (Arithmetic & Logic Unit) [src]

In computing, an arithmetic logic unit (ALU) is a digital circuit that performs arithmetic and logical operations. The ALU is a fundamental building block of the central processing unit (CPU) of a computer, and even the simplest microprocessors contain one for purposes such as maintaining timers. The processors found inside modern CPUs and graphics processing units (GPUs) accommodate very powerful and very complex ALUs; a single component may contain a number of ALUs.



Schematic representation of ALU
By derivative work: Eadthem (talk)ALU_symbol.svg: en:User:Cburnett (ALU_symbol.svg) [GFDL or CC-BY-SA-3.0], via Wikimedia Commons

For the operation: $x = A + B$

- the registers **A** and **B** would supply the values to be added.
- **O**, the operand (addition in this case), tells the CPU what operation is to be performed.
- The result is supplied as **X**,
- along with a "status" signal, **S**.

For logic operations, the ALU is simply comparing two values (equal, less than, or greater than). Each operation performed in an instruction cycle is similarly small. The computer seems to be doing much more by doing a lot of these operations every second. (It can also utilize time better than this implies, for example, while adding the two numbers it can simultaneously be fetching the next instruction.)

Speed

The clock speed is the main determinant for the overall speed of the computer, specified in gigahertz (GHz) for desktop units or MHz for hand-held devices. This refers to how many times a set of bits (**words**) are presented to the processor in a second. The processor receives information in waves. Each of the 32 (or 64 according to the word size) receptacles receive either a zero or a one (electric charge or no charge) in one cycle. Then the next cycle comes with its own set of bits. The rate at which these waves enter into the processor is measured in cycles per second (Hertz or Hz), for example, 2.3 GHz.[src]

Word size

Earlier, we talked about a **byte** as the important grouping of (8) bits in a computer. The size of a **word** on a computer is a changing thing. As indicated above, a word is the number of bits that the CPU can read or write all at once, in one instruction cycle. Word size for PCs (workstations, mainframes and supercomputers vary) has changed from 16 bits, to 32 bits, and is now changing to 64 bits.

The more bits that can be processed at once, the more that can be done in one cycle. One number often uses 32 or 64 bits. (Recall that 8 bits only has 256 possible values!) A smaller word size would mean we have to read multiple times to get a single number, one of the values to be added. Hence, a larger word size means faster operations.

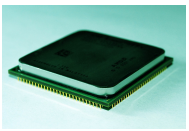
Highway 401 with 9 lanes in one direction. More lanes, means more cars (bits, if you see the analogy) per unit time.

By Robert Jack ?? Will (http://www.flickr.com/photos/bob406/3860422159/) [CC-BY-SA-2.0], via Wikimedia Commons

Multicore [src]

A multi-core processor is a single computing component with two or more independent actual processors (called "cores"), which are the units that read and execute program instructions. The data in the instruction tells the processor what to do. The instructions are very basic things like reading data from memory or sending data to the user display, but they are processed so rapidly that we experience the results as the smooth operation of a program. Manufacturers typically integrate the cores onto a single integrated circuit.

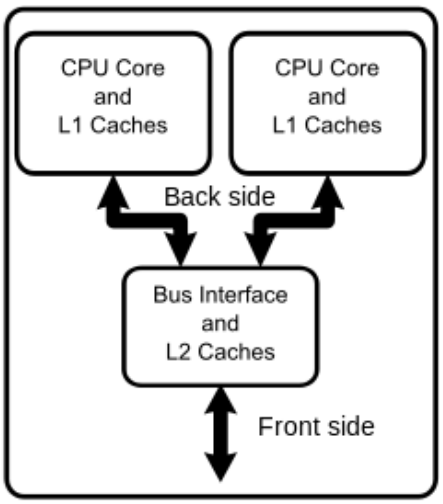
Intel(R) Core(TM)2 Duo – *public domain image*



AMD Athlon X2 Dual-Core Processor 6400+ in AM2 package

By Babylonfive David W. Smith (Own work) [CC-BY-3.0], via Wikimedia Commons

Current PCs are being sold with dual– or even quad–core processors. This means that there are two (or four) processors sharing the workload. (You may recognize this idea as the basis for supercomputers.) Under certain specific conditions, a dual core machine may reach speeds twice as high as a single core machine, but generally two cores does not mean twice as fast. This is because some programs are not easily split among multiple cores and, as you can see in the diagram, the multiple cores *share* some components.



Generic dual core model

public domain image

The big advantage to you is the ability to shift some things, like playing music or displaying TV, to one core, while you work almost uninterrupted on another core.

#25 Multicore demystified (alternate)

#49 Core 2 demystified (alternate)

#132 Centrino: what is it? (alternate)

Cache memory [src]

Like RAM (to be discussed in more detail), cache memory is temporary, or volatile, memory. If the processor is a doctor, and RAM is the big waiting room with everyone in it, while cache is like the examining room that you still have to wait in before the doctor arrives.

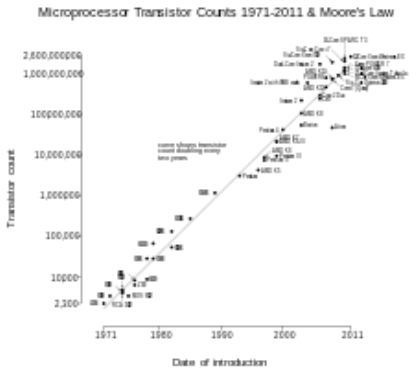
Cache briefly stores data which the processor will use soon. This is done so that while working the computer does not have to always fetch instructions directly from RAM, speeding the process up. There are three types of caches. They include *level 1*, *level 2* and *level 3*. The first level cache is called the internal cache which is part of the microprocessor chip and contains the least kilobytes of the three. It operates faster even though it has lower capacity. It was the first developed (thus the number), but didn't satisfy our need for the volume of data that we needed to process quickly. So Level 2 cache was developed. It is external cache (separate from the microprocessor), but has more memory (from 64 KB to 2MB). The level 3 cache is also separate from the microprocessor but only found in high end computers. Cache memory is searched in order for the data being sought.

More cache memory implies more speed (and more cost).

Moore's Law [src]

Moore's law describes a long-term trend in the history of computing hardware. The number of transistors that can be placed inexpensively on an integrated circuit doubles approximately every two years. This trend has continued for more than half a century and is expected to continue until 2015 or later.

Memory capacity and processor speed are directly related to the number of transistors and how far apart they are. The key point you can get from this graph is the likely speed up you will get by replacing (or upgrading) the CPU after a few years. Since most people don't upgrade the CPU, this adds to the argument for replacing a system that is several years old.



Moore's Law
By Wgsimon (Own work) [CC-BY-SA-3.0 or GFDL], via Wikimedia Commons

previous sub-module | module | course | next sub-module