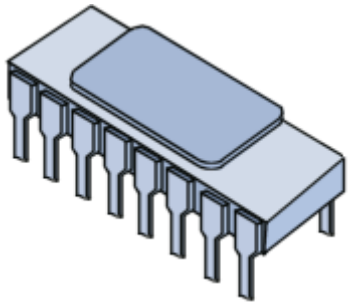
Processor



Introduction

The **processor** (**CPU**, for *Central Processing Unit*) is the computer's brain. It allows the processing of numeric data, meaning information entered in binaryform, and the execution of instructions stored in memory.

The first **microprocessor** (Intel 4004) was invented in 1971. It was a 4-bit calculation device with a speed of 108 kHz. Since then, microprocessor power has grown exponentially. So what exactly are these little pieces of silicone that run our computers?

Operation

The **processor** (called **CPU**, for *Central Processing Unit*) is an electronic circuit that operates at the speed of an internal clock thanks to a quartz crystal that, when subjected to an electrical currant, send pulses, called "**peaks**". The **clock speed** (also called **cycle**), corresponds to the number of pulses per second, written in Hertz (Hz). Thus, a 200 MHz computer has a clock that sends 200,000,000 pulses per second. Clock frequency is generally a multiple of the system frequency (*FSB*, *Front-Side Bus*), meaning a multiple of the motherboardfrequency.

With each clock peak, the processor performs an action that corresponds to an instruction or a part thereof. A measure called **CPI** (*Cycles Per Instruction*) gives a representation of the average number of clock cycles required for a microprocessor to execute an instruction. A microprocessorâ€(TM)s power can thus be characterized by the number of instructions per second that it is capable of processing. **MIPS** (millions of instructions per second) is the unit used and corresponds to the processor frequency divided by the *CPI*.

Instructions

An **instruction** is an elementary operation that the processor can accomplish. Instructions are stored in the main memory, waiting to be processed by the processor. An instruction has two fields:

- the **operation code**, which represents the action that the processor must execute;
- the **operand code**, which defines the parameters of the action. The operand code depends on the operation. It can be data or a memory address.

Operation Code	Operand Field

The number of bits in an instruction varies according to the type of data (between 1 and 4 8-bit bytes).

Instructions can be grouped by category, of which the main ones are:

- **Memory Access**: accessing the memory or transferring data between registers.
- **Arithmetic Operations**: operations such as addition, subtraction, division or multiplication.
- **Logic Operations**: operations such as AND, OR, NOT, EXCLUSIVE NOT, etc.
- **Control**: sequence controls, conditional connections, etc.

Registers

When the processor executes instructions, data is temporarily stored in small, local memory locations of 8, 16, 32 or 64 bits called **registers**. Depending on the type of processor, the overall number of registers can vary from about ten to many hundreds.

The main registers are:

- the **accumulator register** (*ACC*), which stores the results of arithmetic and logical operations;
- the **status register** (*PSW*, *Processor Status Word*), which holds system status indicators (carry digits, overflow, etc.);
- the **instruction register** (*RI*), which contains the current instruction being processed;
- the **ordinal counter** (*OC* or *PC* for *Program Counter*), which contains the address of the next instruction to process;

- the **buffer register**, which temporarily stores data from the memory.

Cache Memory

Cache memory (also called *buffer memory*) is local memory that reduces waiting times for information stored in the RAM (Random Access Memory). In effect, the computer's main memory is slower than that of the processor. There are, however, types of memory that are much faster, but which have a greatly increased cost. The solution is therefore to include this type of local memory close to the processor and to temporarily store the primary data to be processed in it. Recent model computers have many different levels of cache memory:

- **Level one cache memory** (called **L1 Cache**, for **Level 1 Cache**) is directly integrated into the processor. It is subdivided into two parts:
  - the first part is the instruction cache, which contains instructions from the RAM that have been decoded as they came across the pipelines.
  - the second part is the data cache, which contains data from the RAM and data recently used during processor operations.

  Level 1 caches can be accessed very rapidly. Access waiting time approaches that of internal processor registers.
- **Level two cache memory** (called **L2 Cache**, for **Level 2 Cache**) is located in the case along with the processor (in the chip). The level two cache is an intermediary between the processor, with its internal cache, and the RAM. It can be accessed more rapidly than the RAM, but less rapidly than the level one cache.
- **Level three cache memory** (called **L3 Cache**, for **Level 3 Cache**) is located on the motherboard.

All these levels of cache reduce the latency time of various memory types when processing or transferring information. While the processor works, the level one cache controller can interface with the level two controller to transfer information without impeding the processor. As well, the level two cache interfaces with theRAM (level three cache) to allow transfers without impeding normal processor operation.

Control Signals

Control signals are electronic signals that orchestrate the various processor units participating in the execution of an instruction. Control signals are sent using an

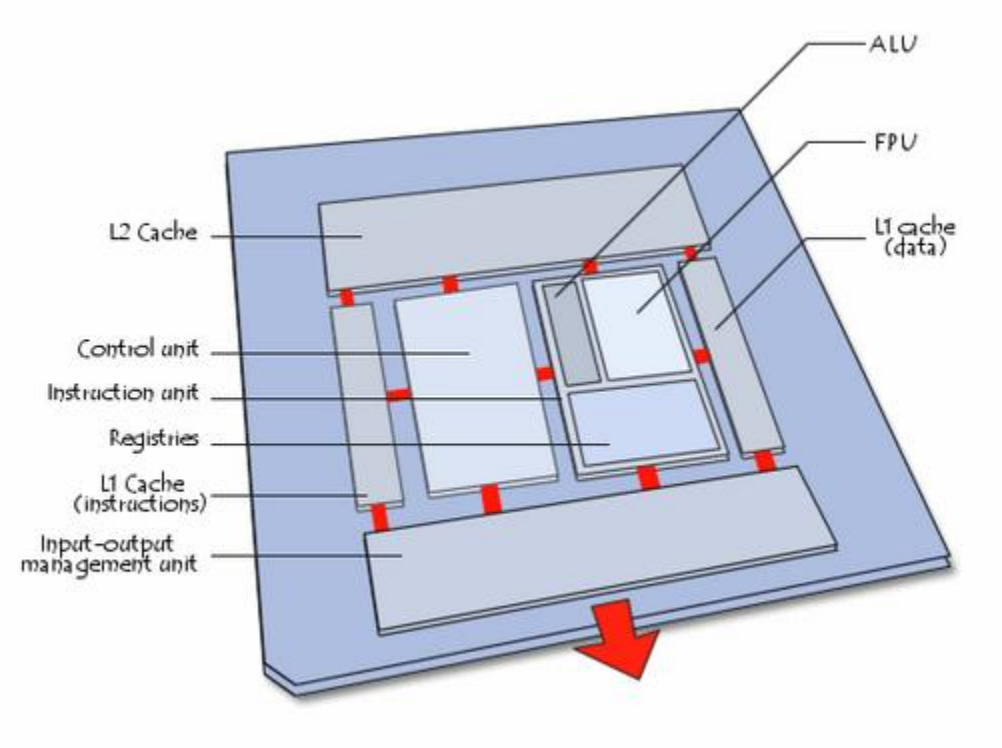
element called a *sequencer*. For example, the *Read / Write* signal allows the memory to be told that the processor wants to read or write information.

Functional Units

The processor is made up of a group of interrelated units (or control units). Microprocessor architecture varies considerably from one design to another, but the main elements of a microprocessor are as follows:

- A **control unit** that links the incoming data, decodes it, and sends it to the execution unit:The control unit is made up of the following elements:
  - **sequencer** (or *monitor and logic unit*) that synchronizes instruction execution with the clock speed. It also sends control signals;
  - **ordinal counter** that contains the address of the instruction currently being executed;
  - **instruction register** that contains the following instruction.
- An **execution unit** (or *processing unit*) that accomplishes tasks assigned to it by the instruction unit. The execution unit is made of the following elements:
  - The **arithmetical and logic unit** (written **ALU**). The ALU performs basic arithmetical calculations and logic functions (AND, OR, EXCLUSIVE OR, etc.);
  - The **floating point unit** (written **FPU**) that performs partial complex calculations which cannot be done by the arithmetical and logic unit.
  - The **status register**;
  - The **accumulator register**.
- A **bus management unit** (or *input-output unit*) that manages the flow of incoming and outgoing information and that interfaces with system RAM;

The diagram below gives a simplified representation of the elements that make up the processor (the physical layout of the elements is different than their actual layout):

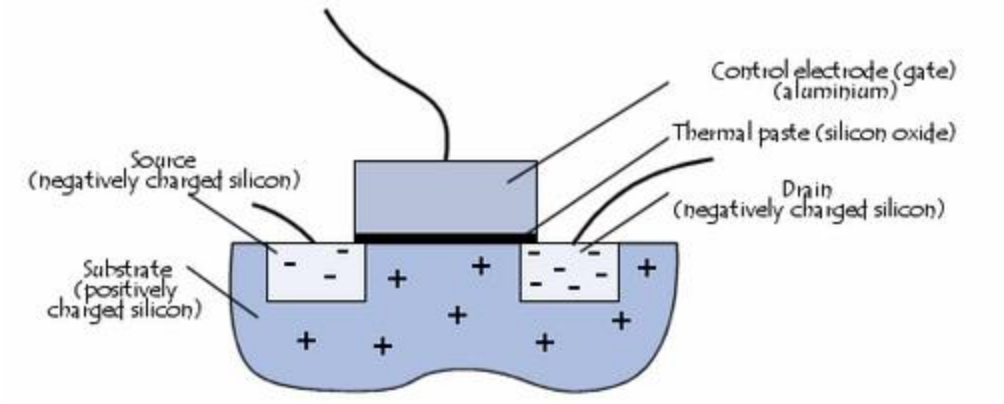


Transistor

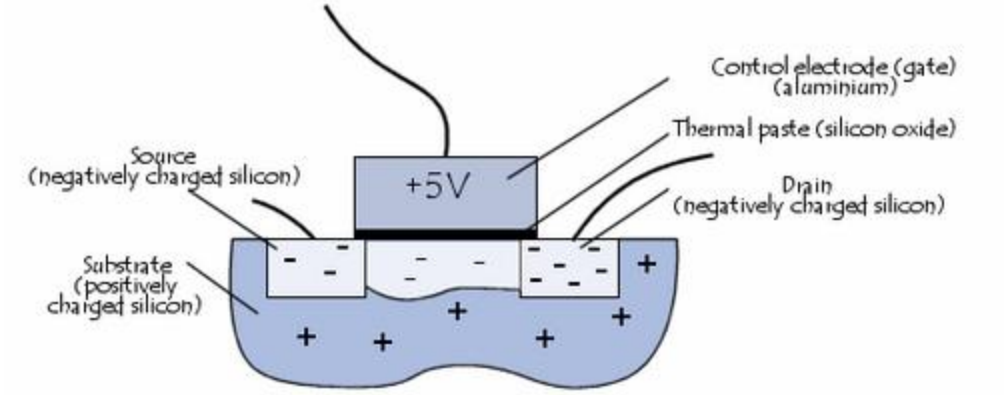
To process information, the microprocessor has a group of instructions, called the "**instruction set**", made possible by electronic circuits. More precisely, the instruction set is made with the help of semiconductors, little "circuit switches" that use the **transistor effect**, discovered in 1947 by *John Barden*, *Walter H. Brattain* and *William Shockley* who received a Nobel Prize in 1956 for it.

A **transistor** (the contraction of *transfer resistor*) is an electronic semi-conductor component that has three electrodes and is capable of modifying current passing through it using one of its electrodes (called control electrode). These are referred to as "active components", in contrast to "passive components", such as resistance or capacitors which only have two electrodes (referred to as being "bipolar").

A MOS (*metal, oxide, silicone*) transistor is the most common type of transistor used to design integrated circuits. MOS transistors have two negatively charged areas, respectively called **source** (which has an almost zero charge) and **drain**(which has a 5V charge), separated by a positively charged region, called a**substrate**). The substrate has a control electrode overlaid, called a **gate**, that allows a charge to be applied to the substrate.



When there is no charge on the control electrode, the positively charged substrate acts as a barrier and prevents electron movement from the source to the drain. However, when a charge is applied to the gate, the positive charges of the substrate are repelled and a negatively charged communication channel is opened between the source and the drain.



The transistor therefore acts as a programmable switch, thanks to the control electrode. When a charge is applied to the control electrode, it acts as a closed interrupter and, when there is no charge, it acts as an open interrupter.

Integrated Circuits

Once combined, transistors can make logic circuits, that, when combined, form processors. The first integrated circuit dates back to 1958 and was built by *Texas Instruments*.

MOS transistors are therefore made of slices of silicone (called *wafers*) obtained after multiple processes. These slices of silicone are cut into rectangular elements to form a "**circuit**". Circuits are then placed in cases with input-output connectors and the sum of these parts makes an "**integrated circuit**". The minuteness of the engraving, written in microns (micrometers, written *Âµm*) defines the number of transistors per surface unit. There can be millions of transistors on one single processor.

Moore's Law, penned in 1965 by Gordon E. Moore, cofounder of Intel, predicted that processor performance (by extension of the number of transistors integrated in the silicone) would double every twelve months. This law was revised in 1975, bringing the number of months to 18. Mooreâ€(TM)s Law is still being proven today.

Because the rectangular case contains input-output pins that resemble legs, the term "**electronic flea**" is used in French to refer to integrated circuits.

Families

Each type of processor has its own instruction set. Processors are grouped into the following families, according to their unique instruction sets:

- 80x86: the "x" represents the family. Mention is therefore made to 386, 486, 586, 686, etc.
- ARM
- IA-64
- MIPS
- Motorola 6800
- PowerPC
- SPARC
- ...

This explains why a program produced for a certain type of processor can only work directly on a system with another type of processor if there is instruction translation, called **emulation**. The term "**emulator**" is used to refer to the program performing this translation.

Instruction Set

An **instruction set** is the sum of basic operations that a processor can accomplish. A processorâ€(TM)s instruction set is a determining factor in its architecture, even though the same architecture can lead to different implementations by different manufacturers.

The processor works efficiently thanks to a limited number of instructions, hardwired to the electronic circuits. Most operations can be performed using basic functions. Some architecture does, however, include advanced processor functions.

CISC Architecture

CISC (*Complex Instruction Set Computer*) architecture means hardwiring the processor with complex instructions that are difficult to create using basic instructions.

CISC is especially popular in 80x86 type processors. This type of architecture has an elevated cost because of advanced functions printed on the silicone.

Instructions are of variable length and may sometimes require more than one clock cycle. Because CISC-based processors can only process one instruction at a time, the processing time is a function of the size of the instruction.

RISC Architecture

Processors with **RISC** (*Reduced Instruction Set Computer*) technology do not have hardwired, advanced functions.

Programs must therefore be translated into simple instructions which complicates development and/or requires a more powerful processor. Such architecture has a reduced production cost compared to CISC processors. In addition, instructions, simple in nature, are executed in just one clock cycle, which speeds up program execution when compared to CISC processors. Finally, these processors can handle multiple instructions simultaneously by processing them in parallel.

Technological Improvements

Throughout time, microprocessor manufacturers (called *founders*) have developed a certain number of improvements that optimize processor performance.

Parallel Processing

Parallel processing consists of simultaneously executing instructions from the same program on different processors. This involves dividing a program into multiple processes handled in parallel in order to reduce execution time.

This type of technology, however, requires synchronization and communication between the various processes, like the division of tasks in a business: work is divided into small discrete processes which are then handled by different departments. The operation of an enterprise may be greatly affected when communication between the services does not work correctly.

Pipelining

Pipelining is technology that improves instruction execution speed by putting the steps into parallel.

To understand the pipelineâ€(TM)s mechanism, it is first necessary to understand the execution phases of an instruction. Execution phases of an instruction for a processor with a 5-step "classic" pipeline are as follows:

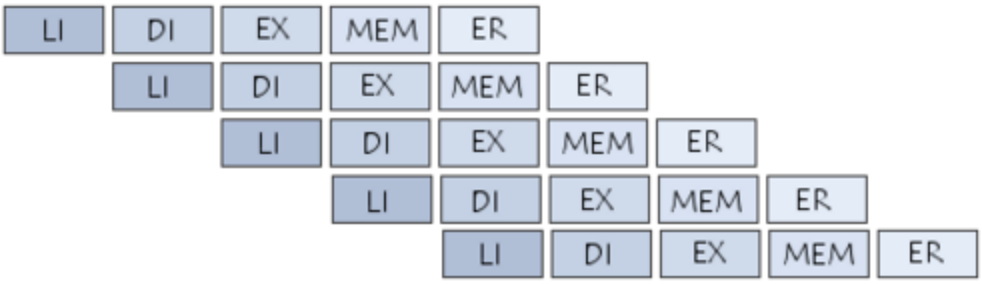
- **FETCH**: (*retrieves* the instruction from the cache;
- **DECODE**: *decodes the instruction* and looks for operands (register or immediate values);
- **EXECUTE**: *performs the instruction* (for example, if it is an ADD instruction, addition is performed, if it is a SUB instruction, subtraction is performed, etc.);
- **MEMORY**: *accesses the memory*, and writes data or retrieves data from it;
- **WRITE BACK (retire)**: *records* the calculated value in a register.

Instructions are organized into lines in the memory and are loaded one after the other.

Thanks to the pipeline, instruction processing requires no more than the five preceding steps. Because the order of the steps is invariable (FETCH, DECODE, EXECUTE, MEMORY, WRITE BACK), it is possible to create specialized circuits in the processor for each one.

The goal of the pipeline is to perform each step in parallel with the preceding and following steps, meaning reading an instruction (FETCH) while the previous step is being read (DECODE), while the step before that is being executed (EXECUTE), while the

step before that is being written to the memory (MEMORY), and while the first step in the series is being recorded in a register (WRITE BACK).



In general, 1 to 2 clock cycles (rarely more) for each pipeline step or a maximum of 10 clock cycles per instruction should be planned for. For two instructions, a maximum of 12 clock cycles are necessary (10+2=12 instead of 10*2=20) because the preceding instruction was already in the pipeline. Both instructions are therefore being simultaneously processed, but with a delay of 1 or 2 clock cycles. For 3 instructions, 14 clock cycles are required, etc.

The principle of a pipeline may be compared to a car assembly line. The car moves from one workstation to another by following the assembly line and is completely finished by the time it leaves the factory. To completely understand the principle, the assembly line must be looked at as a whole, and not vehicle by vehicle. Three hours are required to produce each vehicle, but one is produced every minute!

It must be noted that there are many different types of pipelines, varying from 2 to 40 steps, but the principle remains the same.

Superscaling

Superscaling consists of placing multiple processing units in parallel in order to process multiple instructions per cycle.

HyperThreading

HyperThreading (written *HT*) technology consists of placing two logic processors with a physical processor. Thus, the system recognizes two physical processors and behaves like a multitasking system by sending two simultaneous threads, referred to as **SMT** (*Simultaneous Multi Threading*). This "deception" allows processor resources to be better employed by guaranteeing the bulk shipment of data to the processor.

Source: http://en.kioskea.net/contents/408-processor