Processor

Definition

The processor (or CPU, Central Processing Unit) is a complex integrated circuit characterized by very high integration and equipped with the ability to interpret and execute the instructions of a program.

Definition

- It is responsible for organizing the tasks specified by the program and ensuring their execution.

- It must also take into account information external to the system and ensure its processing.

Definition

- A microprocessor brings together ever more complex functionalities over a few square millimeters.

- Their power continues to increase and their size decreases regularly, still respecting, for the moment, the famous Moore's law.

Definition

- The processor executes each instruction very quickly, in a few clock cycles.

- All computer activity is timed by a single clock, so that all electronic circuits all work together in synchronization.

Definition

- The frequency of this clock is expressed in MHz (millions of cycles per second) or GHz (billions of cycles per second).

- The processor executes program instructions using a set of instructions.

Definition

- **Example :**

- A processor clocked at 2GHz will perform roughly 2 billion beats per second.

- For human, to count up to 2 billion, at a single number per second

It takes approximately 63 years

Processor Characteristics

A processor is defined by:

- The number and width of its internal registers
- The clock speed expressed in MHz or GHz;
- The number of computing cores;
- Instruction Set Architecture;
- The number of internal cache memories;
- Manufacturing Process (Node Size).
- Thermal Design Power
- Power Efficiency

Instruction Set

- The first step in designing a microprocessor is defining its instruction set.

- The set of instructions describes all of the elementary operations that the microprocessor can execute

Instruction Set

- The instructions found in each microprocessor are classified into 4 groups:

- Data transfer
- Arithmetic operations
- Logical operations
- Sequence control

Instruction Set

Arithmetic Instructions:

- Operations: Addition, subtraction, multiplication, division, and other arithmetic operations.
- Example Instruction: ADD, SUB, MUL, DIV

Logic Instructions:

- Operations: Bitwise operations, logical AND, OR, NOT, etc.
- Example Instruction: AND, OR, XOR, NOT

Instruction Set

Data Transfer Instructions:

- Operations: Move data between registers, memory, and I/O devices.
- Example Instruction: MOV, LOAD, STORE

Control Transfer Instructions:

- Operations: Change the sequence of program execution, alter program flow.
- Example Instruction: JUMP, BRANCH, CALL, RET

Instruction Set

Comparison Instructions:

- Operations: Perform comparisons between values.
- Example Instruction: CMP, TEST

Shift and Rotate Instructions:

- Operations: Shift and rotate bits within a data word.
- Example Instruction: SHL, SHR, ROL, ROR

Instruction Set

Stack Instructions:

- Operations: Manipulate the stack (push, pop).
- Example Instruction: PUSH, POP

Floating-Point Instructions:

- Operations: Perform arithmetic and logic operations on floating-point numbers.
- Example Instruction: FADD, FSUB, FMUL, FDIV

Instruction Set

String Instructions:

- Operations: Process strings of characters or bytes.
- Example Instruction: MOVS, LODS, STOS

SIMD Instructions (Single Instruction, Multiple Data):

- Operations: Perform the same operation on multiple data elements simultaneously.
- Example Instruction: MMX, SSE, AVX (used in multimedia and parallel processing)

Instruction Set

I/O Instructions:

- Operations: Input and output operations.
- Example Instruction: IN, OUT

Privileged Instructions:

- Operations: Special instructions that can only be executed in privileged modes (e.g., supervisor or kernel mode).
- Example Instruction: HLT (halt), CLI (clear interrupt flag)

Instruction Set

```
for (int variable = startvalue; variable <= endvalue; ++variable)
{
    statement
}
```

```
MOV CX, startvalue
For1:
   CMP CX, endvalue
   JA  Endfor1
   statement
   INC CX
   JMP  For1
Endfor1:
```

Instruction Set

- Processors use tiny transistors to do basic operations

- The processor actually works using a very limited number of functions (logical AND, logical OR, addition, etc.).

- It is impossible to put all the instructions on a processor, because it is limited by the Node Size.

Instruction Set

- To put more instructions, we need a processor with a very large surface area.

- Unfortunately, the processor is made of silicon (platinum or gold bus) and silicon is expensive and it gets very hot.

- The processor therefore processes complicated information using simple instructions.

Processor family

CISC (Complex Instruction Set Computing):

- **Characteristics:**
  - Large and complex instruction set.
  - Instructions can perform multiple low-level operations.
  - Variable-length instructions.
  - Emphasizes hardware-based complexity.
- **Examples:**
  - x86 architecture (Intel and AMD processors).
  - Motorola 68k.

Processor family

RISC (Reduced Instruction Set Computing):

- **Characteristics:**
  - Simplified instruction set with a focus on a small, highly optimized set of instructions.
  - Single-cycle execution for most instructions.
  - Fixed-length instructions.
  - Emphasizes compiler-based optimization.
- **Examples:**
  - ARM architecture.
  - MIPS architecture.
  - PowerPC architecture.

Processor family

EPIC (Explicitly Parallel Instruction Computing):

- **Characteristics:**
  - Developed by Intel and Hewlett-Packard.
  - Similar to RISC but emphasizes parallel processing.
  - Uses very long instruction word (VLIW) architecture.
- **Examples:**
  - Intel Itanium (IA-64).

Processor family

VLIW (Very Long Instruction Word):

- **Characteristics:**
  - Similar to EPIC architecture.
  - Instructions specify multiple operations that can be executed in parallel.
- **Examples:**
  - Elbrus architecture.

Processor family

SIMD (Single Instruction, Multiple Data):

- **Characteristics:**
  - Executes the same operation on multiple data elements simultaneously.
- **Examples:**
  - Intel MMX, SSE, AVX instructions.

Processor family

DSP (Digital Signal Processor) Architecture:

- **Characteristics:**
  - Optimized for processing digital signals.
  - Often uses SIMD (Single Instruction, Multiple Data) instructions for parallelism.
- **Examples:**
  - Texas Instruments C6000 series.

Processor family

Hybrid Architectures:

- Some modern processors incorporate features from both CISC and RISC architectures, blurring the lines between the traditional distinctions.

➢ This explains why a program created for one type of processor cannot run directly on a system with another type of processor, unless the instructions are translated, called emulation.

➢ The term “emulator” is used to designate the program carrying out this translation.