Processor Basis.

- CA

③

Fundamental Set of CPU:

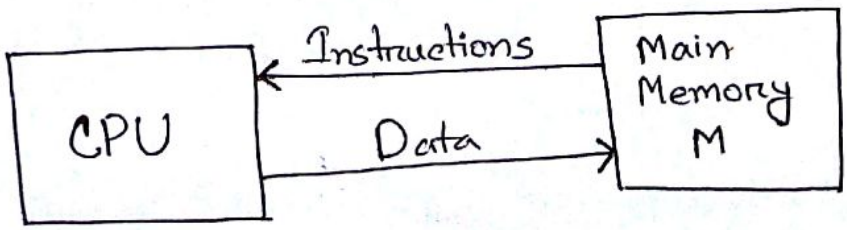
The primary function of the CPU and other instruction-set processors is to execute sequence of instructions, that is, programs, which are stored in an external main memory. Program execution is therefore carried out as follows:

1. The CPU transfers instructions and, when necessary, their input data (operands) from main memory to registers in the CPU.

2. The CPU executes the instructions in their stored sequence except when execution sequence is explicitly altered by a branch instruction.

3. When necessary, the CPU transfers output data (results) from the CPU registers to main memory.

Processor-main memory communication without cache:



as in figure, no cache memory is present, the CPU communicates directly with the main memory M, which is typically a high-capacity multichip random-access memory (RAM). The CPU is significantly faster than M; that is, it can read from or write to the CPU's registers perhaps 5 to 10 times faster than it can read from or write to.

Processor - Main memory communication using cache:

Many computers have a cache memory CM positioned between the CPU and main memory. The cache CM is smaller and faster than main memory and may reside, wholly or in part, on the same chip as the CPU. It typically permits the CPU to perform a memory load or store operation in a single clock-cycle, whereas a memory access that bypasses the cache and is handled by main memory takes many clock cycle.

CPU-IO communication :

The CPU communicates with IO devices in much the same way as it communicates with external memory. The IO devices are associated with addressable registers called IO ports to which the CPU can store a word (an output operation) or from which it can load a word (an input operation). In some computers there are IO instruction per se; all IO data transfers are implemented by memory-referencing instructions, an approach called "memory-mapped IO". This approach requires that memory locations and IO ports share the same set of addresses, so an address bit pattern that is assigned to memory cannot also be assigned to an IO port, and vice versa. Other computers employ IO instructions that are distinct from memory-referencing instructions. These instructions produce control signals to which IO ports, but not memory locations, respond. This second approach is sometimes called "IO mapped IO".

Interrupt:

To design a CPU so that it can receive requests for supervisor services directly from secondary memory units and other IO devices. Such a request is called an "interrupt".

CPU operation:

The sequence of operations performed by the CPU in processing an instruction constitutes an instruction cycle.

All instructions require two major steps: a "fetch step" during which a new instruction is read from the external memory M and an "execute step" during which the operations specified by the instruction are executed. A check for pending interrupt requests is also usually included in the instruction cycle.

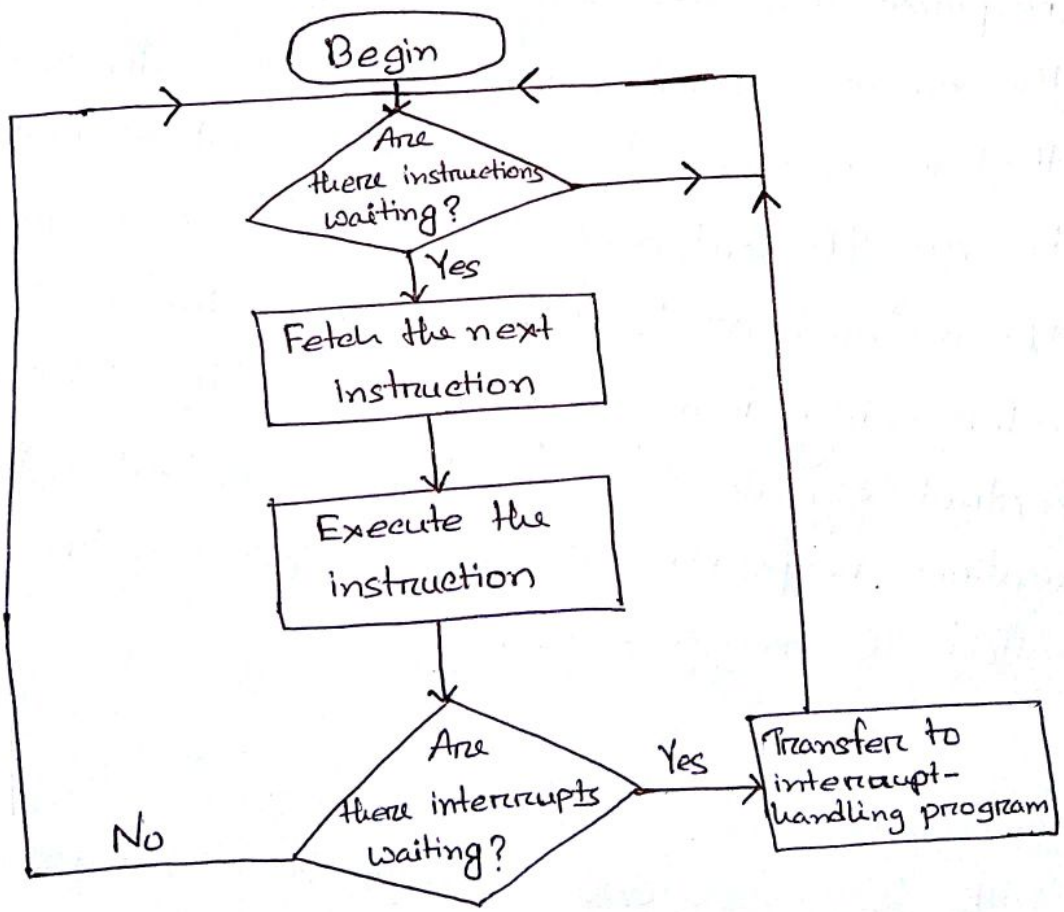


Fig: Overview of CPU behavior.

Accumulator-based CPU:

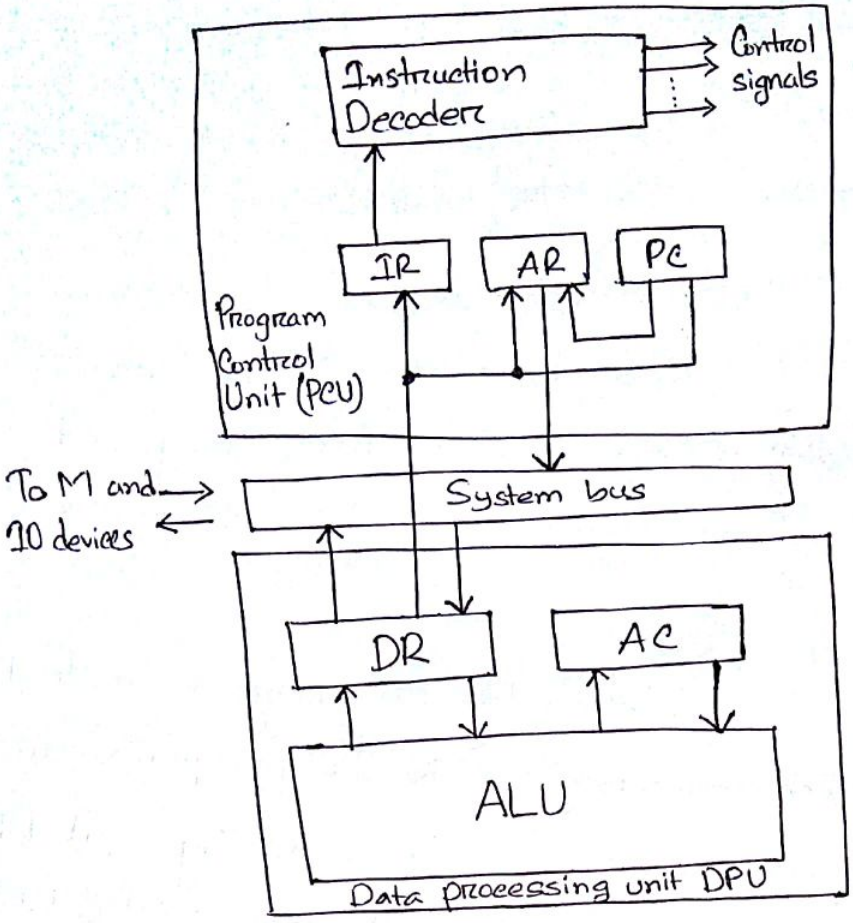
The CPU organization proposed by von Neumann and his colleagues for the IAS computer is the basis for most subsequent designs.

In many early designs, one of the CPU registers, the "accumulator", played a central role, being used to store an input or output operand (result) in the execution for many instructions.

Instructions are fetched by the program control unit PCU, whose main register is the program counter PC. They are executed in the data processing unit DPU, which contains an n-bit arithmetic-logic unit (ALU) and two data registers AC and DR. Most instructions perform operations of the form:

$$X1 := f_i(X1, X2)$$

where X1 and X2 denote a CPU register (AC, DR or PC) or an external memory location M (adr). The operations f_i performed by the ALU are limited of fixed-point (integer) addition and subtraction, shifting and logical (word-gate) operations.



AR : Address Register
IR : Instruction "
PC : Program counter

AC : Accumulator register
DR : Data register.

fig: A small accumulator-based CPU.

Programming Considerations:

Data-processing operations normally require up to three operands. For example, the addition

$$Z := X + Y$$

has three distinct operands X, Y and Z. The accumulator-based CPU of upper figure supports only single-address instructions, that is, instructions with one explicit memory address. However, AC and DR can serve as implicit operand locations so that multioperand operations can be implemented by executing several instructions in sequence. For example, a program to implement, assuming that X, Y and Z all refer to data words in M, can take the following form:

HDL format	Narrative format (comment)
AC := M(X)	Load X from M into accumulator AC
DR := AC	Move contents of AC to DR
AC := M(Y)	Load Y into accumulator AC
AC := AC + DR	Add DR to AC
M(Z) := AC	Store contents of AC in M

Architecture Extensions:

There are many ways in which the basic design of the figure of accumulator based CPU can be improved. Most recent CPUs contain the following extensions, which significantly improve their performance and ease of programming.

(i) Multipurpose register set for storing data and addresses.

(ii) Additional data, instruction and address types.

(iii) Register to indicate computation status.

(iv) Program control stack.

① These replace the accumulator AC and the auxilary registers DR and AR of our basic CPU.

(ii) Most CPUs have instructions to handle data and addre with several different word sizes and formats.

(iii) A status register (also called condition code or flag register) indicates infrequent or exceptional conditions resulting from the instruction execution.

(iv) Various special registers and instructions facilitate the transfer of control among programs due to procedure calling or external interrupts.

Pipelining :

Modern CPUs employ a variety of speedup techniques, including cache memories, and several forms of instruction level parallelism. Such parallelism may be present in the internal organization of the DPU or in the overlapping of the operations carried out by the DPU and PCU

The ARM6 Microprocessor:

The ARM has its origins in the Acorn RISC machine, a microprocessor developed in the UK to serve as the CPU of a personal computer.

The ARM family is primarily aimed at low-cost, low-power applications such as portable computer and games. The ARM6 is a 32-bit processor in that both its data words and its address words are 32 bits (4 bytes) long.

The maximum memory size of an ARM6 computer is 2^{32} bytes, also referred to as 4 GB.

Internal organization:

It has a 32-bit ALU and a file of 32-bit general-purpose registers. To permit direct interaction between data and control register.

In user mode the register file appears to contain sixteen 32-bit registers designated, R0:R15, where R15 is also the program counter.

The ALU is designed to perform basic arithmetic operations on 32-bit integers. It employs combinational logic for addition and subtraction and a sequential shift-and-add method. A combinational shift circuit is attached to the ALU support multiplication and other operations. A separate address-incrementer circuit implements address-manipulation operations such as

$PC := PC + 1$. independentaly of the ALU.

CISC Machine :

It is designed to handle 32-bit words (termed long words in 680X0 literature) efficiently, but instructions are also provided to handle operands of 1, 8, 16, and 64 bits

The data-processing unit has a register file containing sixteen 32-bit registers, half of which are data registers designed D0:D7 and half are address registers designed A0:A7. The ALU can execute a large set of fixed point.

Basic Data Representation Format:

In view of the importance of numerical computation, computer designs have paid a great deal of attention to the representation of numbers.

Two main number formats have evolved:

(i) Fixed-point
(ii) Floating-point.

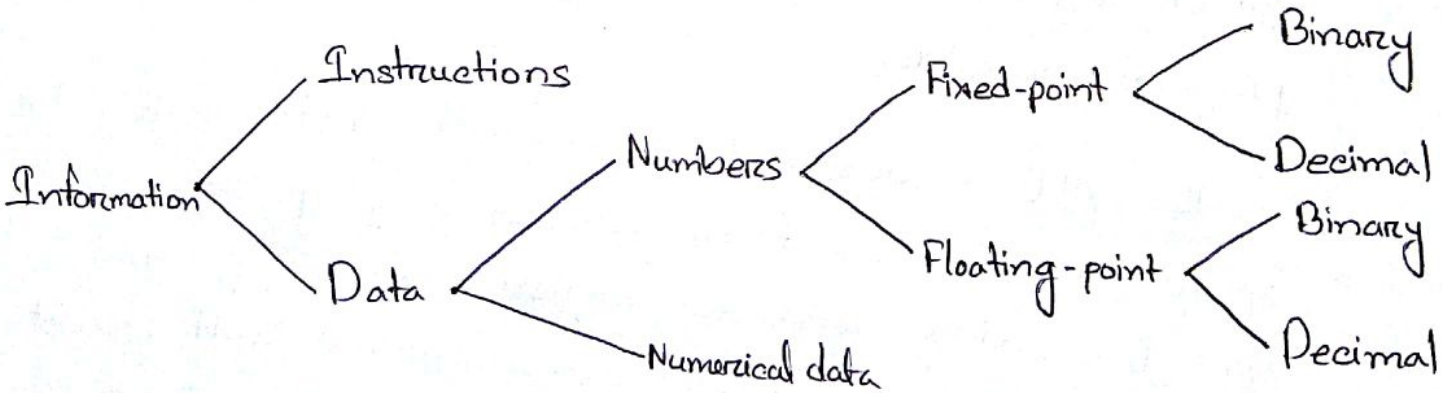


fig: The basic information types.

Word Length:

Information is represented in a decimal computer by means of binary words, where a word is a unit of information of some fixed ~~length~~. length n. An n=4, we can encode the 10 decimal digits as follows:

0=0000, 1=0001, 2=0010, 3=0011, 4=0100
5=0101, 6=0110, 7=0111, 8=1000, 9=1001

To encode alphanumeric symbols or characters, 8-bit words called bytes are comonly used.

Word size is typically a multiple of 8, common CPU word sizes being 8, 16, 32, 64 bits.

Fixed-Point Numbers:

In selecting a number representation to be used in a computer, the following factors should be taken into account:

- ⇒ The number types to be represented; for example, integers or real numbers.
- ⇒ The range of values (number magnitudes) likely to be encountered.
- ⇒ The precision of the numbers, which refers to the maximum accuracy of the representation.
- ⇒ The cost of the hardware required to store and process the numbers.

The range of numbers that can be represented by a fixed-

P

Floating-Point Numbers:

The range of numbers that can be represented by a fixed-point number code is insufficient for many operations or applications, particularly scientific computations where very large are very small numbers are encounted. Scientific notation permits us to represent such numbers using relatively few digits. For example, it is easier to write a quintillion as

$$1.0 \times 10^{18}$$

Basic Formats:

Three numbers are associated with a floating-point number:

mantissa, M
exponent, E
base, B.

The mantissa M is also referred to as the significant or fraction in the literature. These three components together represent the real number: $M \times B^E$.

Completeness: A function $f(x)$ is said to be computable if it can be evaluated in a finite number of steps by a Turing machine. While real computers differ from Turing machines in having only a finite ~~ammount~~ amount of memory, they can, in practice, evaluate any computable function to a reasonable degree of approximation.

Instructions:

Instructions are conveniently divided into the following 5 types.

1. "Data-transfer instructions", which copy information from one location to another either in the processor's internal register set or in the external main memory.

2. "Arithmetic instructions", which perform operations on numerical data.

3. "Logical instructions", which include Boolean and other nonnumerical operations.

4. "Program-Control instructions", such as branch instructions, which change the sequence in which programs are executed.

5. "Input-Output Instructions", which cause information to be transferred between the processor or its main memory and external IO devices.