**Von Neumann architecture**

The von Neumann architecture, is a [computer architecture](https://en.wikipedia.org/wiki/Computer_architecture) based on a design architecture for an electronic [digital computer](https://en.wikipedia.org/wiki/Digital_computer) with parts consisting of a unit containing an [arithmetic logic unit](https://en.wikipedia.org/wiki/Arithmetic_logic_unit) and [processor registers](https://en.wikipedia.org/wiki/Processor_register); a [control unit](https://en.wikipedia.org/wiki/Control_unit) containing an [instruction register](https://en.wikipedia.org/wiki/Instruction_register) and [program counter](https://en.wikipedia.org/wiki/Program_counter); a [memory](https://en.wikipedia.org/wiki/Computer_memory) to store both [data](https://en.wikipedia.org/wiki/Data_%28computing%29) and [instructions](https://en.wikipedia.org/wiki/Instruction_set); external [mass storage](https://en.wikipedia.org/wiki/Mass_storage); and [input and output](https://en.wikipedia.org/wiki/Input_and_output) mechanisms with a shared common system [bus](https://en.wikipedia.org/wiki/Bus_%28computing%29)



Figure 1 shows Von Neumann architecture scheme.

# System bus

A system bus is a single [computer bus](https://en.wikipedia.org/wiki/Bus_%28computing%29) that connects the major components of a computer system, a central processing unit (CPU) or processor, a memory unit, and input/output (I/O) devices.

The term bus is used to represent a group of electrical signals or the wires that carry these signals. the three major components of the system bus are the address bus, data bus, and control bus. Simply, the functions of a [data bus](https://en.wikipedia.org/wiki/Memory_bus) to carry information, an [address bus](https://en.wikipedia.org/wiki/Address_bus) to determine where it should be sent, and a [control bus](https://en.wikipedia.org/wiki/Control_bus) to determine its operation.

The width of address bus determines the memory addressing capacity of the processor. The width of data bus indicates the size of the data transferred between the processor and memory or I/O device. For example, the 8086 processor had a 20-bit address bus and a 16-bit data bus. The amount of physical memory that this processor can address is 220 bytes, or 1 MB, and each data transfer involves 16 bits. The Pentium processor, for example, has 32 address lines and 64 data lines. Thus, it can address up to 232 bytes, or a 4 GB memory. Furthermore, each data transfer an move 64 bits. The control bus consists of a set of control signals. Typical control signals include memory read, memory write, I/O read, I/O write, interrupt, interrupt acknowledge, bus request, and bus grant. These control signals indicate the type of action taking place on the system bus. For example, when the processor is writing data into the memory, the memory write signal is asserted. Similarly, when the processor is reading from an I/O device, the I/O read signal is asserted.

**Good to know :**

--> The data bus consist of 8, 16, or 32 parallel lines.

--> The data bus is bidirectional bus, means the data can be transferred from CPU to main memory and vice versa.

--> The number of data lines used in the data bus is equal to the size of data word being written or read.

--> The data bus also connects the I/O ports and CPU. So, the CPU can write data to or read it from the memory or I/O ports.

# The Memory unit

The memory of a computer system consists of tiny electronic switches, with each switch set in one of two states: open or closed. It is, however, more convenient to think of these states as 0 and 1 rather than open and closed. A single such switch can be used to represent two (i.e., binary) numbers. The memory unit consists of millions of switches, each switch can represent one bit. In order to make memory more manageable, bits are organized into groups of eight bits called bytes. Memory can then be viewed as consisting of an ordered sequence of bytes. Each byte in this memory can be identified by its sequence number starting with 0, This is referred to as the memory address of the byte. Such memory is called byte addressable memory. The amount of memory that a processor can address depends on the address bus width. Typically, 32-bit processors support 32-bit addresses. Thus, these processors can address up to 4 GB (2^^ bytes) of main memory as shown in Figure 3.1. This number is referred to as the memory address space. The actual memory in a system, however, is always less than or equal to the memory address space.



**Basic Memory Operations**

The memory unit supports two fundamental operations: read and write. The read operation reads a previously stored data and the write operation stores a value in memory. Both of these operations require an address in memory from which to read a value or to which to write a value. In addition, the write operation requires specification of the data to be written.

Two metrics are used to characterize memory. Access time refers to the amount of time required by the memory to retrieve the data at the addressed location. The other metric is the memory cycle time, which refers to the minimum time between successive memory operations. Memory transfer rates can be measured by the bandwidth metric, which specifies the number of bytes transferred per second.

Steps in a typical read cycle

1. Place the address of the location to be read on the address bus;

2. Activate the memory read control signal on the control bus;

3. Wait for the memory to retrieve the data from the addressed memory location and place it on the data bus;

4. Read the data from the data bus;

5. Drop the memory read control signal to terminate the read cycle.

For example, a simple Pentium read cycle takes three clock cycles. During the first clock cycle, steps 1 and 2 are performed. The processor waits until the end of the second clock and reads the data and drops the read control signal. If the memory is slower (and therefore cannot supply data within the specified time), the memory unit indicates its inability to the processor and the processor waits longer for the memory to supply data by inserting wait cycles. Note that each wait cycle introduces a waiting period equal to one system clock period and thus slows down the system operation.

**You have to know**

A clock cycle is defined as the time between two successive rising edges as shown in Figure 2.26. You can also treat the period between successive falling edges as a clock cycle. Clock rate or frequency is measured in number of cycles per second. This number is referred to as Hertz (Hz). The clock period is defined as the time represented by one clock cycle. All three clock signals in Figure 2.26 have the same clock period.

Clock period = 1/ Clock frequency

For example, a clock frequency of 1 GHz yields a clock period of 1/ 1 X 109 = 1ns.

Note that one nanosecond (ns) is equal to 10"^ second.



**Steps in a typical write cycle**

1. Place the address of the location to be written on the address bus;

2. Place the data to be written on the data bus;

3. Activate the memory write control signal on the control bus;

4. Wait for the memory to store the data at the addressed location;

5. Drop the memory write signal to terminate the write cycle.

**Types of Memory**

The memory unit can be implemented using a variety of memory chips—different speeds, different manufacturing technologies, and different sizes. The two basic types of memory are the read-only memory and read/write memory. A volatile memory requires power to retain its contents. A nonvolatile memory can retain its values even in the absence of power.

**Read-Only Memories**

Read-only memory (ROM) allows only read operations to be performed. As the name suggests, we cannot write into this memory. The main advantage of ROM is that it is nonvolatile. Most ROM is factory programmed and cannot be altered. The term programming in this context refers to writing values into a ROM.

The program that controls the standard input and output functions (called BIOS), for instance, is kept in ROM. Current systems use the flash memory rather than a ROM (see our discussion later).

Other types include programmable ROM (PROM) and erasable PROM (EPROM). PROM is

useful in situations where the contents of ROM are not yet fixed EPROM offers further flexibility during system prototyping. Contents of EPROM can be erased by exposing them to ultraviolet light for a few minutes. Once erased, EPROM can be reprogrammed again.

Electrically erasable PROMs (EEPROMs) allow further flexibility EEPROMs, on the other hand, allow the user to selectively erase contents. Furthermore, erasing can be done in place; there is no need to place it in a special ultraviolet chamber.

Flash memory is a special kind of EEPROM. Flash memory, however, is slower than the RAMs we discuss next. For example, flash memory cycle time is about 80 ns whereas the corresponding value for RAMs is about 10 ns.

**Read/Write Memory**

 Read/write memory is commonly referred to as random access memory (RAM), even though ROM is also a random access memory. This terminology is so entrenched in the literature that we follow it here with a cautionary note that RAM actually refers to RWM. Read/write memory can be divided into static and dynamic categories. Static random access memory (SRAM) retains the data, once written, without further manipulation so long as the source of power holds its value. SRAM is typically used for implementing the processor registers and cache memories.

The bulk of main memory in a typical computer system, however, consists of dynamic random access memory (DRAM). DRAM is a complex memory device that uses a tiny capacitor to store a bit. A charged capacitor represents 1 bit. Since capacitors slowly lose their charge due to leakage, they must be periodically refreshed to replace the charges representing 1 bit. A typical refresh period is about 64 ms. Reading from DRAM involves testing to see if the corresponding bit cells are charged. Unfortunately, this test destroys the charges on the bit cells. Thus, DRAM is a destructive read memory.

For proper operation, a read cycle is followed by a restore cycle. As a result, the DRAM cycle time, the actual time necessary between accesses, is typically about twice the read access time, which is the time necessary to retrieve a datum from the memory.

Next : *you should have some understanding of how the processor is organized and the system is put together.Ch4*

*Ref : Guide to assembly language in Linux*