

College of Computer Technology

Memory Addressing

The memory of a computer system consists of tiny electronic switches, with each switch 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. Thus, each switch can represent a bit. The memory unit consists of millions of such bits. In order to make memory more manageable, eight bits are grouped into a byte. Memory can then be viewed as consisting of an ordered sequence of bytes. Each byte in this memory is identified by its sequence number starting with 0, as shown in Figure 1. This is referred to as the *memory address* of the byte. Such memory is called *byte addressable* memory because each byte has a unique address.

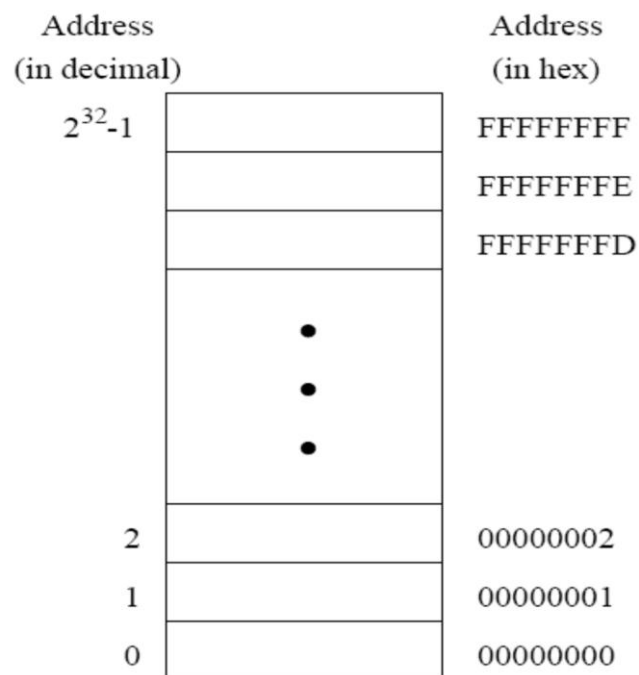


Figure 1 Logical view of the system memory.

College of Computer Technology

The Pentium can address up to 4 GB (2^{32} bytes) of main memory (see Figure 1). This magic number comes from the fact that the address bus of the Pentium has 32 address lines. This number is referred to as the *memory address space*. The memory address space of a system is determined by the address bus width of the processor used in the system. The actual memory in a system, however, is always less than or equal to the memory address space. The amount of memory in a system is determined by how much of this memory address space is *populated* with memory chips.

Memory chips:

It is possible to visualize a typical internal main memory structure as consisting of rows and columns of basic cells. Each cell is capable of storing one bit of information. Figure 2 provides a conceptual internal organization of a memory chip. In this figure, cells belonging to a given row can be assumed to form the bits of a given memory word. Address lines $A_0 A_1 \dots A_{n-2} A_{n-1}$ are used as inputs to the address decoder in order to generate the word select lines $W_0 W_1 \dots W_{2^n-1}$. A given word select line is common to all memory cells in the same row. At any given time, the address decoder activates only one word select line while deactivating the remaining lines. A word select line is used to enable all cells in a row for read or write. Data (bit) lines are used to input or output the contents of cells. Each memory cell is connected to two data lines. A given data line is common to all cells in a given column.

College of Computer Technology

The address lines $A_0 \dots A_{n-1}$ in the memory chip shown in Figure 2 contain an address, which is decoded from an n -bit address into one of 2^n locations (The number of locations may be obtained from the address width of the chip) within the chip, each of which has a w -bit word associated with it. The storage capacity of a memory chip is the product of the number of locations and the word width. The chip thus contains $2^n \times w$ bits.

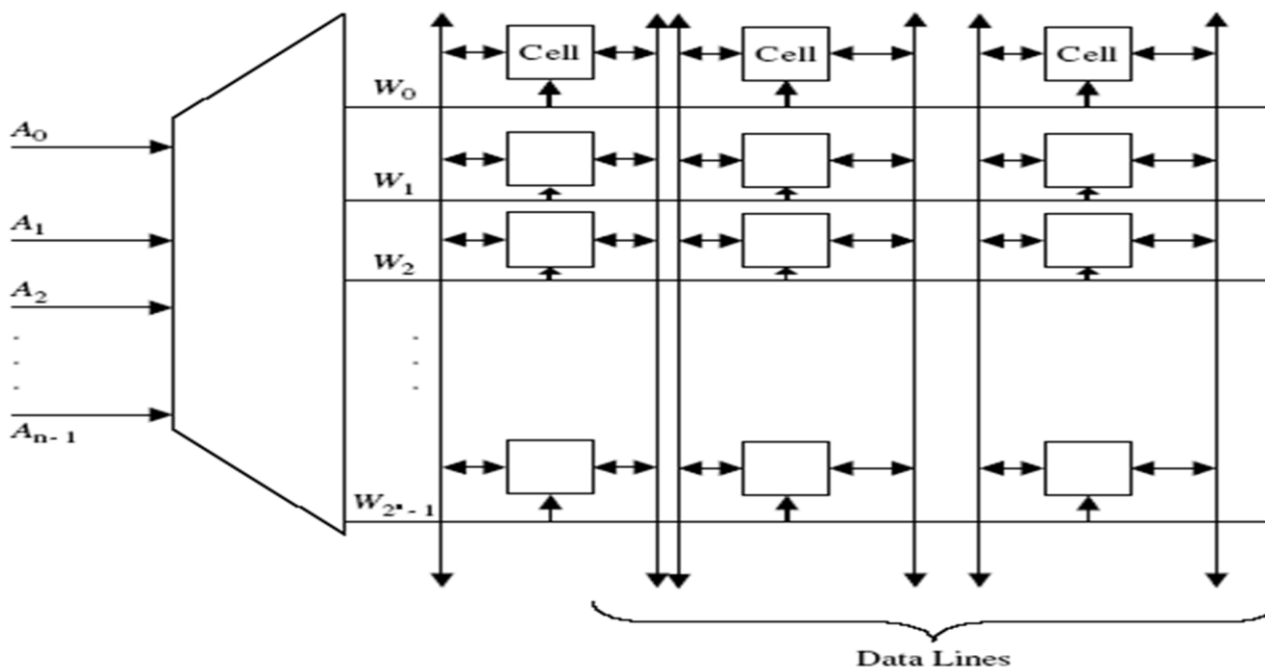


Figure 2: A conceptual internal organization of a memory chip

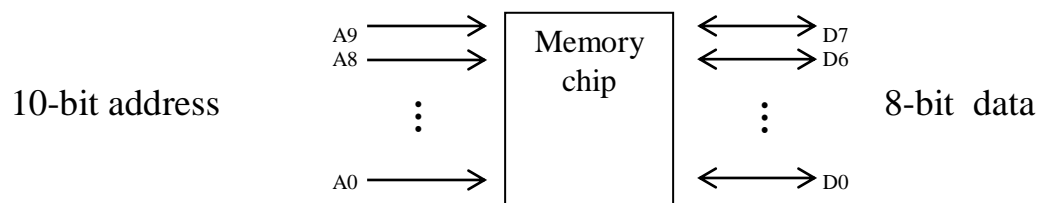
The computer's word size can be the size of the data bus which carries data between the CPU and memory and the CPU and I/O devices.

To access the memory, to store or retrieve a single word of information, it is necessary to have a unique address. The word address is the number that identifies the location of a word in a memory.

College of Computer Technology

For example, a chip with 10 address lines has $2^{10}=1024$ locations. Given an 8 bit data width, a 10-bit address chip has a memory size of :

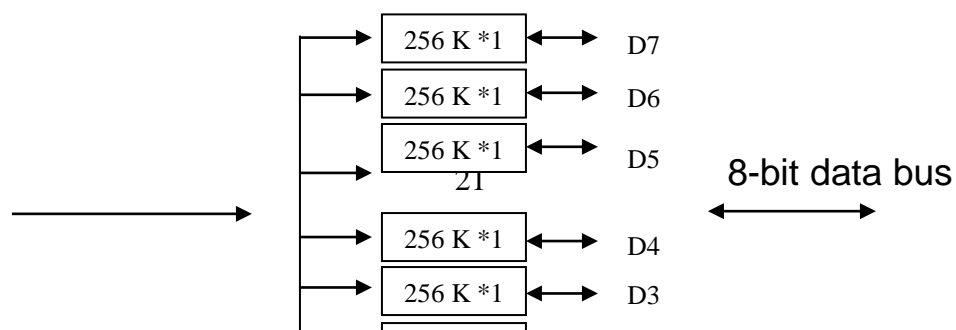
$$2^{10} * 8 = 1024 * 8 = 1K * 1\text{byte} = 1\text{Kbyte}$$



The second property of memory chips is access time, access time is the speed with which a location within the memory chip may be made available to the data bus. It is defined as the time interval between the instant that an address is sent to the memory chip and the instant that the data stored in the locations appears on the data bus. Access time is given in nanoseconds (ns) and varies from 25 ns to the relatively slow 200 ns.

In general a computer with a larger word size can execute programs of instructions at a faster rate because more data and more instructions are stuffed into words. The larger word sizes, however, mean more lines making up the data bus, and therefore more interconnections between the CPU and memory and I/O devices.

A single chip is usually insufficient to provide the memory requirements of a computer. A number of chips are therefore connected in parallel to form what is known as a **memory bank**. Figure below shows a memory bank consisting of eight 1-bit chips. Each has 18 address lines ($A_0 \dots A_{17}$). The total storage capacity of one chip = $2^{18} * 1 = 256 \text{ K}$ bits. Total size of the memory bank = $256 \text{ K bits} \times 8 = 256 \text{ KB}$



address bus A0 ...A17

Ex: A certain memory chip is specified as 2KX8

- 1- How many words can be stored on this chip?**
- 2- What is the word size :**
- 3- How many total bits can this chip store?**

Solution:

- 1- $2K=2*1024=2048$ words
- 2- The word size is 8-bits(1-byte)
- 3- Capacity = $2048 * 8=16383\text{bits}=16\text{Kbit}$

16- Byte Ordering

Storing data often requires more than a byte. For example, we need four bytes of memory to store an integer variable that can take a value between 0 and $2^{32}-1$. Let us assume that the value to be stored is the one in Figure 16a.

Suppose that we want to store these 4-byte data in memory at locations 100 through 103. How do we store them? Figure 16 shows two possibilities: least significant byte (Figure 16b) or most significant byte (Figure 16c) is stored at location 100. These two byte ordering schemes are referred to as the *little endian* and *big endian*. In either case, we always refer to such multibyte data by specifying the lowest memory address (100 in this example).

Is one byte ordering scheme better than the other? Not really! It is largely a matter of choice for the designers. For example, Pentium processors use the little-endian byte ordering. However, most processors leave it up to the system designer to configure the processor. For

College of Computer Technology

example, the MIPS and PowerPC processors use the big-endian byte ordering by default, but these processors can be configured to use the little-endian scheme.

The particular byte ordering scheme used does not pose any problems as long as you are working with machines that use the same byte ordering scheme. However, difficulties arise when you want to transfer data between two machines that use different schemes. In this case, conversion from one scheme to the other is required. For example, the Pentium provides two instructions to facilitate such conversion: one to perform 16-bit data conversions and the other for 32-bit data.

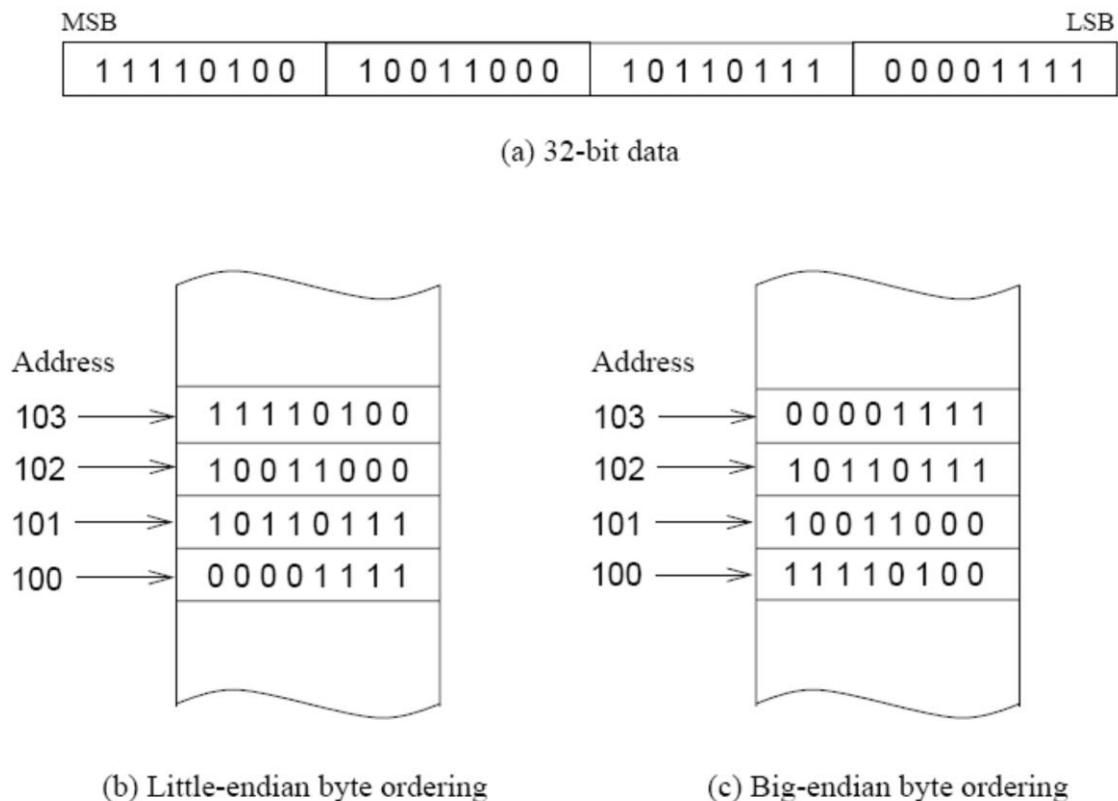


Figure 10 Two byte ordering schemes commonly used by computer systems.