Memory Organization

- CA

Memory types:

The information-storage components of a computer can be placed-in four groups:

(i) CPU register.
(ii) Main (primary) memory.
(iii) Secondary memory.
(iv) Cache.

(i) CPU register:

These high-speed registers in the CPU serve as the working memory for temporary storage of instructions and data. They usually form a general-purpose "register file" for storing data as it is processed. A capacity of 32 data words is typical of a register file, and each register can be accessed, that is, read from or written into, within a single clock cycle (a few nanoseconds).

(ii) Main memory:

This large, fairly fast external memory stores programs and data that are in active use. Storage locations in main memory are addressed directly by the CPU's load and store instructions.

(iii) Secondary memory: This memory type is much larger in capacity but also much slower than memory. Secondary memory stores system programs, large data files, and the

like that are not continually required by the CPU. It also acts as an overflow memory when the capacity of the main memory is exceeded.

⑳ Cache:

Most computers now ~~to~~ have another level of IC memory- ~~someth~~ sometimes several such level - called cache memory, which is positioned logically between the CPU registers and main memory. A cache's storage capacity is less than that of main memory, but with an access time of one to three cycles, the cache is much faster than main memory because some or all of it can reside on the same IC as the CPU.

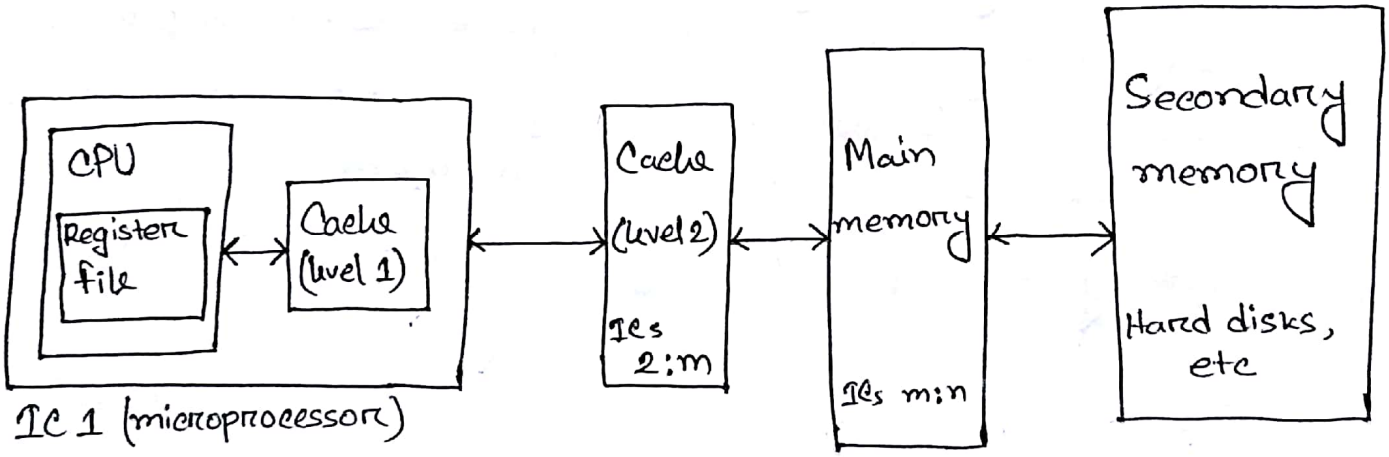

fig: Conceptual organizational of a multilevel memory system in a computer.

Performance and Cost:

The most meaningful measure of the cost of a memory device is the purchase price to the user of a complete unit. The price should include not only the cost of the information storage medium itself but also the cost of the peripheral equipment (access circuitry) needed to operate the memory.

Let C be the price in dollars of a complete memory system with S bits of storage capacity. We define the cost c of the memory as follow:

$$c = \frac{C}{S} \text{ dollars/bit}$$

The performance of an individual memory device is primarily determined by the rate at which information can be read from or written into the memory. A basic performance measure is the average time to read a fixed amount of information for instance, one word, from the memory. This parameter is called the "read access time" or simply the access time of the memory and is denoted by t_A.

The write access time is defined similarly; it is often, but not always, equal to the read access time.

The access time depends on the physical nature of the storage medium and on the access mechanisms used.

Destructive Readout :

In some memories the method of reading the memory destroys the stored information; this phenomenon is called "destructive readout" (DRO).

Memories in which reading does not affect the stored data have "nondestructive readout" (NDRO).

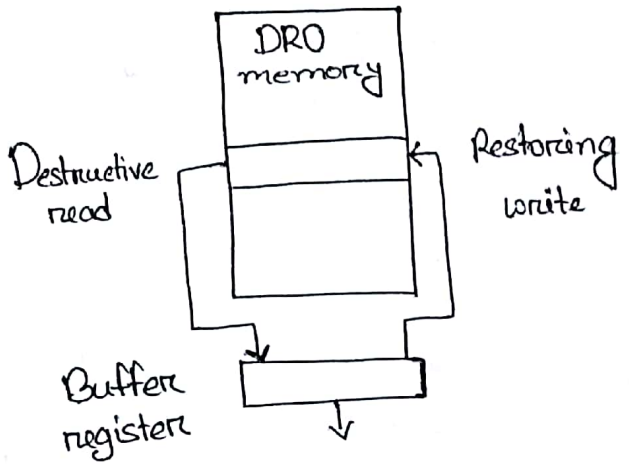

fig: Memory restoration in a destructive readout (DRO) memory.

Dynamic Memory :

Memory that requires periodic refreshing are called dynamic memory.

SRAM, DRAM :

Main memories are usually built from dynamic ICs reffered to as dynamic RAMs (DRAMs).

ICs can also implement static memories reffered to as static RAM (SRAMs).

N.B:

SRAMs tend to be faster, that is have lower access time, than DRAMs, but the cost per bit of SRAMs is higher.

SRAMs are often used to build caches.

Organization : RAM

The RAM operates as follows: First the address of the target location to be accessed in transferred via the address bus to the RAM's address buffer. The address is then processed by the address decoder, which selects the required location in the storage cell unit. A control line indicates the types of access to be performed. If a read operation (load) is requested, the contents of the addressed location are transferred from the storage cell unit to the data buffer and from there to the data bus. If a write (store) is requested, the word to be stored is transferred from the data bus to the selected location in the storage unit. Since it is not usually necessary or desirable to permit simultaneous reading and writing, the input and output data buses are often combined into a single, bidirectional data bus.

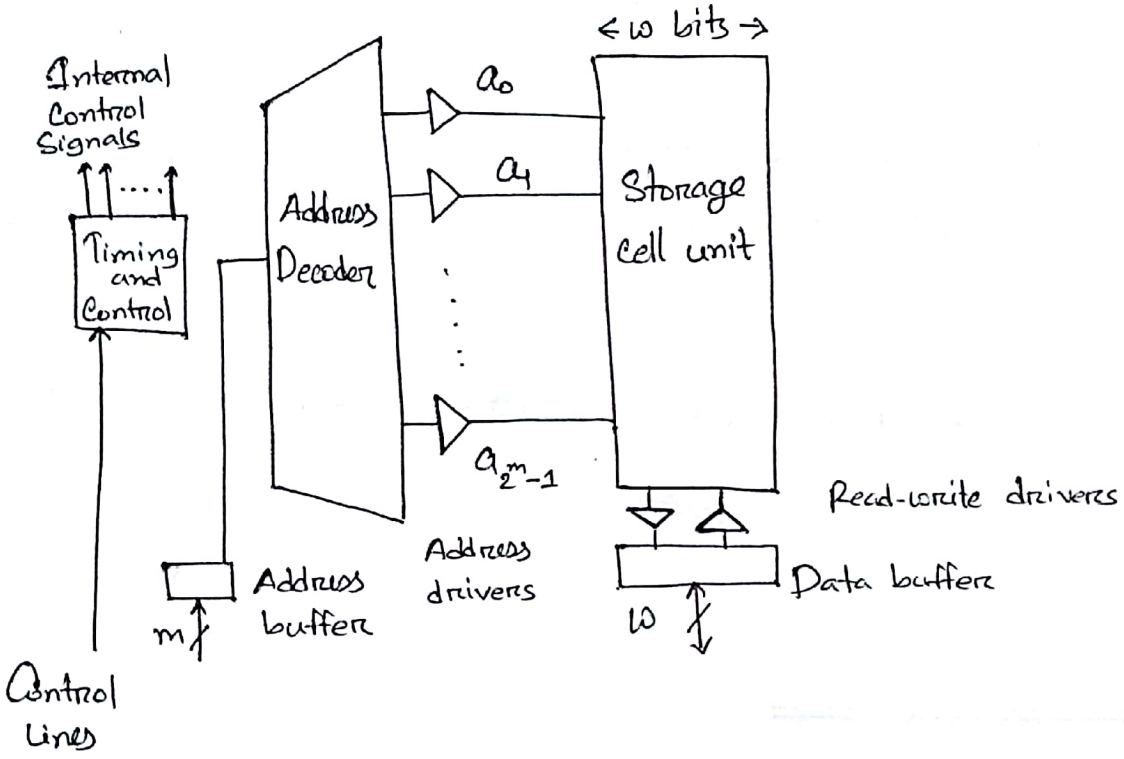

fig: 1D RAM unit.

RAM Design:

WE is the write-enable line, a memory write (read) operation takes place if WE = 1 (0). A second control line, the chip-select line CS, triggers a memory operation. A word is accessed for either reading or writing only when CS is activated. The line signals that the data bus has a word ready to be written into the RAM or, in the case of a read operation, that the data bus is ready to receive a data word.

The RAM has a bidirectional data bus D, which is directly wired to all addressable storage locations, and so it requires a third control line, output enable OE. In write (input) operations this line is deactivated (OE=0), allowing D to act as an input

bus to all storage locations. Of course, only the addressed location actually stores the word received on D. In read (output) operations, OE must be activated (OE = 1) so that only the addressed memory location transfers its data to D.

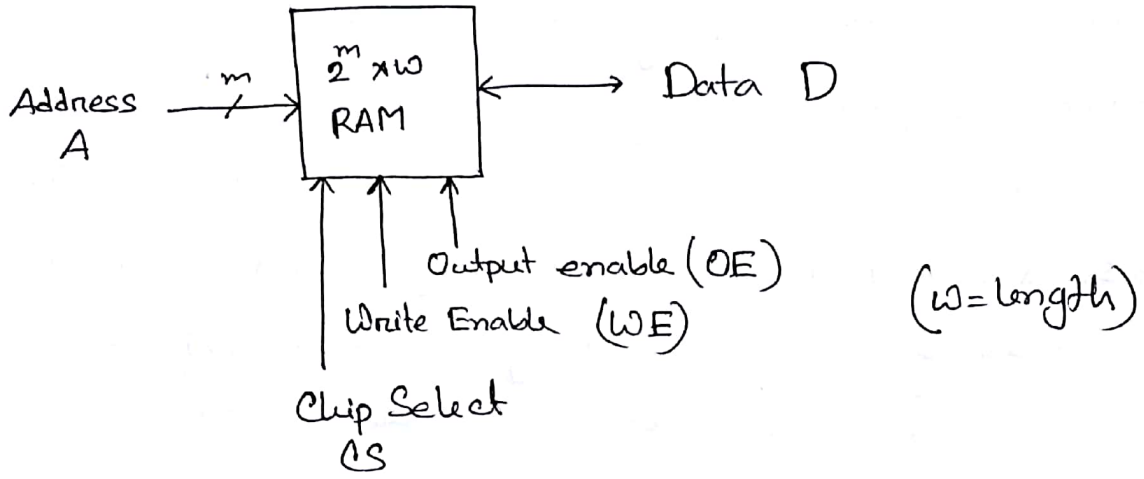

fig: A RAM IC showing its major external connections.

A commercial 64 Mb DRAM chip:

The Micron Technology MT4LC8M8E1, which we will call the 8E1 for short, is a commercial DRAM chip introduced in 1996. It stores 64 Mb, that is 2^{26} bits of data, in single-transistor storage cells of the kind. The stored information is organized as 2^{23} 8-bit bytes. So the 8E1 is also referred to as an 8 M × 8-bit DRAM.

The memory address size $m=23$, and the data word size $w=8$.

~~The internal structure of the 8E1 appears~~

Two-dimensional addressing is employed, with the 23-bit address broken into two parts: a 13-bit row address and a 10-bit column address. Only 13 external address lines are used, allowing the 8E1 to be housed in a small, 32-pin package, which implies that row and column addresses must be multiplexed over the address bus, a common tactic in large RAM chips. This multiplexing is controlled by two lines: RAS (row address select) and CAS (column address select), which replace the generic CS control line.

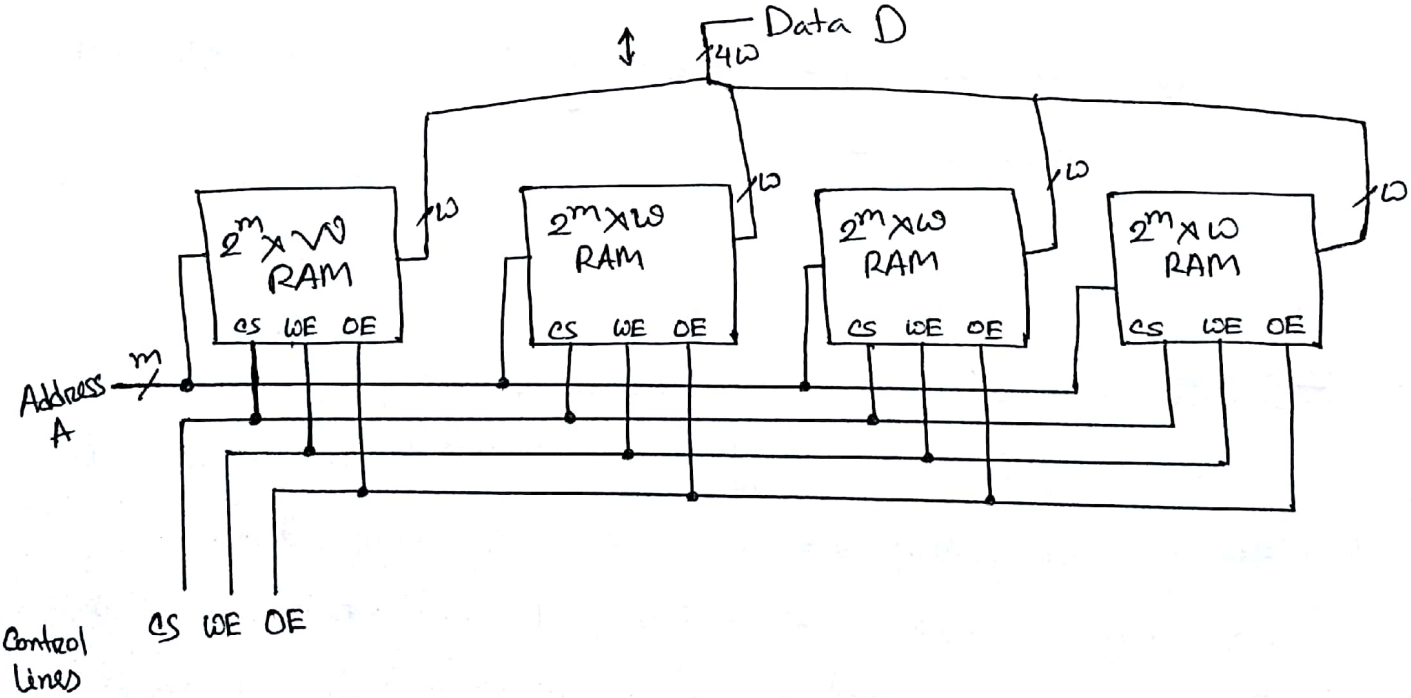

fig: Increasing the word size of a RAM by a factor of four.

Fast RAM interfaces:

A particular RAM technology must supply a faster external processor with individually addressable n-bit words.

There are two basic ways we can increase the data-transfer rate across its external interface by factor of S:

① Use a bigger memory word. We can design the RAM with an internal memory word size of $W = Sn$ bits. The size permits Sn bits to be accessed as a unit in one memory cycle time T_M. We then need fast circuits inside the RAM that, in the case of read operation, can access as Sn-bit word, break it into S parts, and output them to the processor, all within the period T_M. During write operations, these circuits must accept upto Sn-bit words from the processor, assemble them into an nS-bit word, and store the result, again within the period T_M.

② Access more than one word at a time. We can partition the RAM into S separate banks $M_0, M_1, \ldots, M_{S-1}$, each covering part of the memory address space

and each provided with its own addressing circuitry. Then it is possible to carry out S indendent accesses simultaneously in one memory clock period T_M. Once more, we need fast circuits inside the RAM unit to assemble and disassemble the words being accessed.

Multilevel Memories : General characteristics :

Consider a general n-level system of n memory types $(M_1, M_2, \ldots, M_n)$. Typical technologies used in these hierarchies are semiconductor DRAMs for main memory, and magnetic-disk units for secondary memory.

The following relations normally hold between adjacent memory levels M_i and M_{i+1} in a memory hierarchy.

Cost per bit $C_i > C_{i+1}$

Access time $t_{A_i} < t_{A_{i+1}}$

Storage capacity $S_i < S_{i+1}$.

Virtual memory:

The term virtually is applied when the main and secondary memories appear to a user program like a single, large and directly addressable memory.

Traditionally there are three reasons for using virtual memory:

(i) To free user programs from the need to carry out storage allocation and to permit efficient sharing of the available memory space among different users.

(ii) To make programs independent of the configuration and capacity of the physical memory present for their execution; for example, to allow seamless overflow into secondary memory when the capacity of main memory is exceeded.

(iii) To achieve the very low access time and cost per bit that are possible with a memory hierarchy.

Cost and Performance :

The overall goal in memory-hierarchy design is to achieve a performance close to that of the fastest device M_1 and a cost per bit close to that of the cheapest device M_n. The performance of a memory system depends on various related factors, the more important of which are the following :

⇒ The address-reference statistics, that is, the order and frequency of the logical addresses generated by programs that use the memory hierarchy.

⇒ The access time t_{A_i} of each level M_i relative to the CPU.

⇒ The storage capacity S_i of each level.

⇒ The size Sp_i of the blocks (pages) transferred between adjacent levels.

⇒ The allocation algorithm used to determine the regions of memory to which blocks are transferred by the block-swapping process.

Address Translation:

Address translation can be viewed abstractly as a function $f: V \rightarrow R$. This function is not easily characterized, since address assignment and translation is carried out at various stages in the life of a program, specifically:

1. By the programmer while writing the program.
2. By the compiler during program compilation.
3. By the loader at initial program-load time.
4. By run-time memory management hardware and/or software.

Translation look-aside buffer:

The figure shows how various parts of a ~~manual~~ multilevel memory management typically realize the address-translation ideas just discussed. The input address A_v is a virtual address consisting of a (virtual) base address B_v concatenated with a displacement D. A_v contains an effective address computed in accordance with some program-defined addressing mode (direct, indirect, indexed, and so on) for the memory item being accessed. It also can contain system-specific control information—a segment address.

To speed up the mapping process, part (or occationally all) of the memory map is placed in a small high-speed

memory in the CPU called a "translation look-aside buffer (TLB)".

If the virtual address B_v is not currently assigned to the TLB, then the part of the memory map that contains B_v is first transferred from the external memory into the TLB. Hence the TLB itself forms a cachelike level within a multilevel address-storage system for memory maps. For this reason, the TLB is sometimes referred to as an address cache.

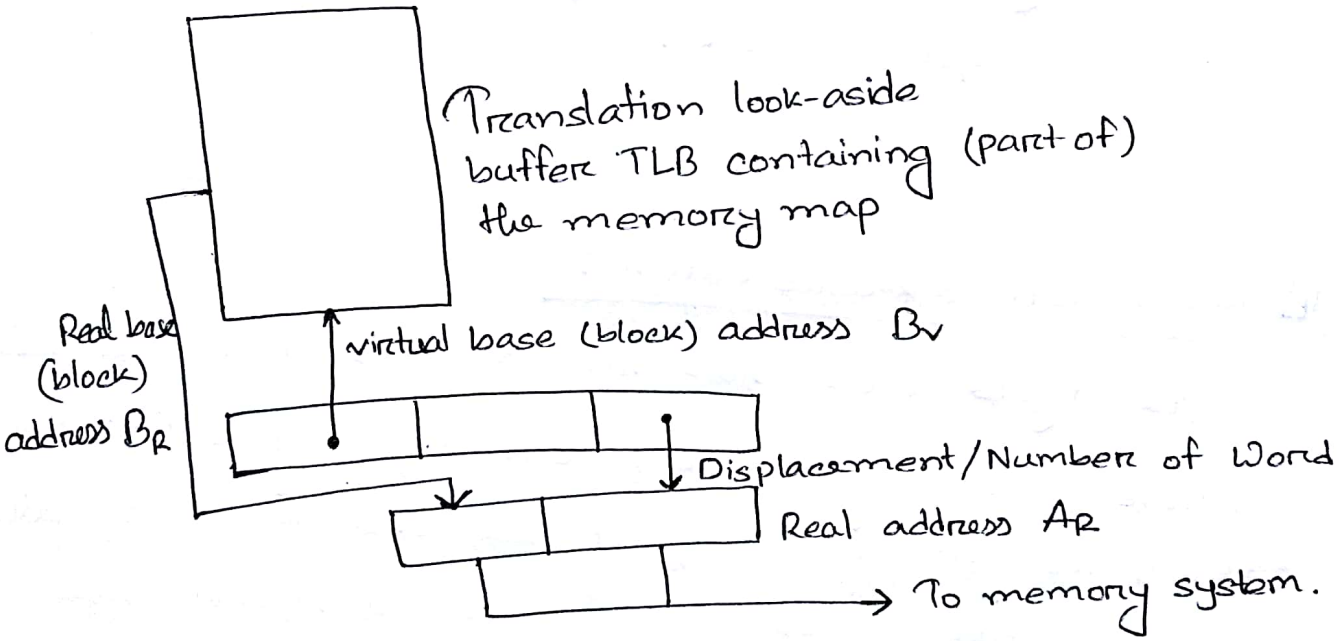

fig: Structure of a dynamic address-translation system.