Chapter -6

Memory Organization

Memory unit is an essential component of computer as it is used for storing programs and data. The memory unit that communicates directly with the CPU is called main memory. A memory device that provides backup storage is called auxiliary memory. E.g. magnetic disks and tapes. Auxiliary memory stores large data files and other information. Programs and data that are currently needed by processor reside in main memory.

Memory hierarchy

The total memory capacity of a computer can be visualized as being hierarchy of components. Memory hierarchy consists of all storage device in a computer system from slow but high capacity auxiliary memory to fast main memory, to a smaller and faster cache memory.

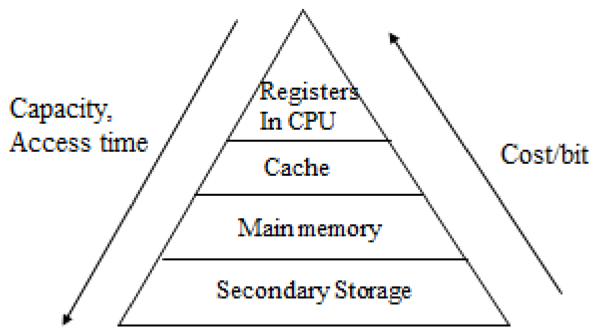


Fig. the memory hierarchy

Main memory

Main memory is the central storage unit in a system. It is a large and fast memory. Main memory is based on semiconductor integrated circuits. RAM and ROM chips are semiconductor ICs. RAM is available in two types:

1. Static RAM
2. Dynamic RAM

Static RAM consists of flip-flop and stores binary information. The information remains valid as long as power is applied to the unit.

Dynamic RAM stores information in the form of electric charge applied for capacitors. The stored charge on capacitors discharges with time thus needs to be charged by refreshing dynamic memory. DRAM offers reduced power consumption and larger storage capacity in a single memory chip.

ROM is used for storing programs that permanently resides in computer. ROM of main memory is needed for storing an initial program called bootstrap loader. Bootstrap loader is the program whose function is to start computer software operating when power is turned on. Contents of ROM remain unchanged when power is turned off. Types of ROM:

1. PROM
2. EPROM
3. EEPROM
4. Flash EEPROM

RAM and ROM chips

A RAM chip is suited for communication with CPU if it has one or more control inputs that selects the chip only when needed. Another feature is bidirectional bus i.e. data transfer takes place either form memory to CPU during read operation and from CPU to memory during write operation. Bidirectional bus is constructed with three state buffer: logic 1 and logic 0 signals or a high impedance state (behaves like open circuit i.e. output does not carry signal with no logic).

Capacity of memory is 128 words of eight bits(one byte) per word. This requires a 7 bit address an 8 bidirectional data bus. Read and write specify memory operation. Chip select are for enabling chip only when it is selected by processor. When chip is selected the two binary states in the line specify two operations of read or write.

CS1	CS2	RD	WR	Memory fxn	State of data bus
0	0	X	X	Inhibit	High Impedance
0	1	X	X	Inhibit	High Impedance
1	0	X	X	Inhibit	High Impedance
1	0	0	1	Write	Input data to RAM
1	0	1	X	Read	Output data from RAM
1	1	X	X	inhibit	High Impedance

Fig: Function Table

ROM chip is organized in a similar manner. Since a ROM can only read the data bus is in output mode. For same chip more bits of ROM is possible than of RAM because internal binary cells in ROM occupy less space than in RAM.

The nine address lines I the ROM chip specify one of 512 bytes stored in it. CS=1 and (CS2)'=0 selects unit to operate. Otherwise it is in high impedance state. No need for read or write control as it can only read.

Auxiliary Memory

Devices provides backup storage are called auxiliary memory. Most common auxiliary memory used in systems are magnetic disks and magnetic tapes. The important characteristics of auxiliary memory devices are access mode, access time, transfer rate, capacity and cost.

The average time required to reach a storage location in memory and obtain its time is called access time. In electromechanical devices with moving parts, the access time consists of seek time required to position read-write head to a location and transfer time required to transfer data to or from the device. Seek time is usually larger than transfer time. Thus data is organized in records or blocks. Reading or writing is done in entire block. The transfer rate is the number of characters or words that the device can transfer per second, after it has been positioned at the beginning of record.

Magnetic Disks

A magnetic disk is a circular plate constructed of both metal or plastic coated with magnetic material. Both sides of disks are used and many disks can be stacked on one spindle with read/write heads available on each surface. Bits are stored in tracks. Tracks are divided into sections called sectors. The minimum quantity of information that can be transferred is sector.

Some units may use single read/write head for each disk surface. In this type track address bits are used by a mechanical assembly to move head into specified track position before reading or writing.

Permanent timing tracks are used in disks to synchronize the bits and recognize the sectors. Here address bits specify disk number, disk surface, the sector number and track within sector. Read/write head must be positioned on specified track for information transfer.

Track nearer to center is smaller than track nearer to circumference. Thus track recording may vary. To make all records in a sector of equal length, disk use variable recording density with higher density on tracks near center than on circumference.

Magnetic Tapes

A magnetic tape transport consists of the electrical, mechanical and electronic components to provide the parts the control mechanism for a magnetic tape. It is a strip of plastic coated with magnetic recording. Bits are recorded as magnetic spots on the tape along tracks. Seven or nine bits data recorded simultaneously with a parity bit. Magnetic tape can be started, stopped can be move forward or reverse or can be rewound. For this purpose information is stored in blocks. Gaps of unrecorded tape are inserted between records where tape can be stopped. Each record has identification bit at the beginning and at the end. Records may be fixed or variable length.

Associative Memory

It is a memory unit accessed in parallel by the content of the data itself rather than by an address. Hence, it is also called C*ontent Addressable Memory (CAM).*When a word is written in an associative memory, no address is given but the memory is capable of finding an empty unused location to store the word. When the word is to be read from an associative memory, the content of the word, or part of word is specified. The memory locates all words which match the specified content and marks them for reading. It is suited for parallel searching and is more expensive than sequential memory. Associative memory is used in applications where the search time is very critical and must be very fast.

Hardware Organization

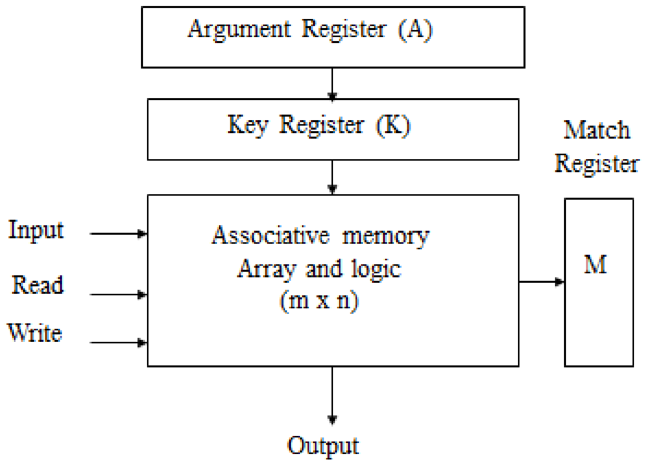


Fig. Associative memory

Associative memory consists of a memory array and logic for m words with n bits per word . the argument register A and key register K each have n bits, one for each bit of a word. The match register M has m bits, one for each memory word. Each word in memory is compared in parallel with the content of argument register. The words that match bits of argument register set a corresponding bit in twords whose corresponding bits in the match register have been set.

The key register provides a mask for choosing a particular field or key in argument word. The argument is compared with each memory word if key registers contains all 1’s. Otherwise only those bits that have 1’s in their corresponding position of key register are compared. Thus key provides a mask or identifying piece of information.

Example of operation (from the figure below):

Suppose –

Data Register = 101 111100

Mask Register = 111 000000

Word 1 = 100 111100 => no match

Word 2 = 101 000001 => match

The relation between memory array and external registers in an associative memory is shown in figure. The cells are marked by C with two subscripts. First gives the word number and second specify the bit position in the word. Thus cell C_{ij} is the cell for bit j in word i. A_j in the argument register is compared with all bits in column j of array provided that $K_j=1$. This is done for all column. If a match occurs between argument and bits in words, the corresponding M_i is set to 1. If one or more unmasked bits of argument and the word do not match, M_i is cleared to 0.

Cache Memory

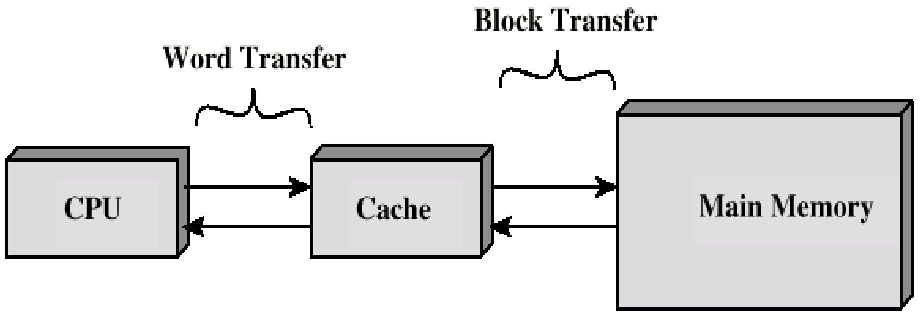
Cache memory is the fast small memory where active portion of programs and data are stored, so that the average access time is reduced. Thus it reduces the total execution time of program. Frequently used data are stored in cache memory. the cache memory access is 5 to 10 times faster than main memory. Average access time of memory can be approached to access time of cache by placing frequently used instruction into cache.

The basic operation of cache is as follows. When CPU needs to access memory, the cache is examined. If the word is found in cache, it is read from there. If the word is not found in cache, main memory is accessed to read the word. A block of word is then transferred to cache memory from main memory.

Performance of cache memory is measured in terms of a quantity called hit ratio. When the CPU refers to memory and finds it in cache, it is said to produce a hit. If the word is not found in cache, it is said to be a miss. The ratio of number of hits divided by total references to memory is the hit ratio. If the hit ratio is high, so that the most of CPU access is in cache than main memory, average access time is closer to access time of cache.

Cache – Main Memory interface

Assume an access to main memory causes a block of K words to be transferred to the cache. The block transferred from main memory is stored in the cache as a single unit called *a slot, line, or page*. Once copied to the cache, individual words within a line can be accessed by the CPU. Because of the high speeds involved with the cache, management of the data transfer and storage in the cache is done in hardware – the O/S does not know about the cache. If there are 2^n words of memory, then there will be $M= 2^n/K$ blocks in the memory.



M will be much greater than the number of lines, C, in the cache. Every line of data in the cache must be tagged in some way to identify what main memory block it is. The line of data and its tag are stored in the cache.

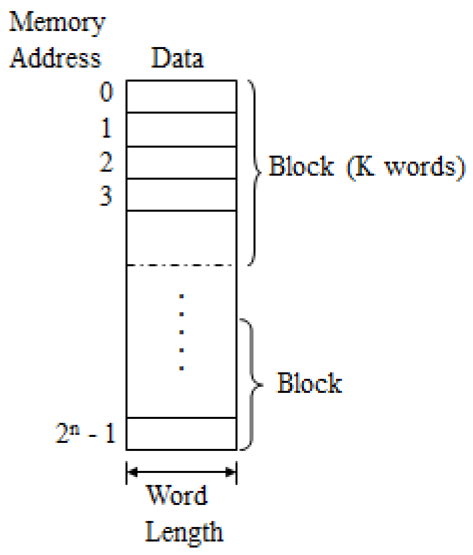


Fig. Main Memory Structure

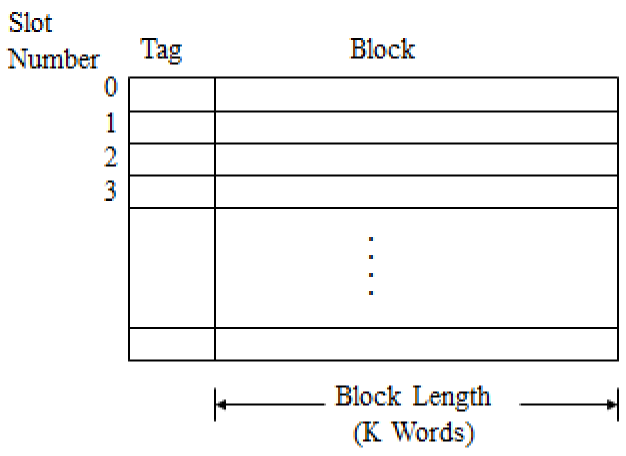


Fig. Cache Memory Structure

The transformation of data from main memory to cache memory is referred as mapping process. Three types of mapping procedure:

1. Direct Mapping
2. Associative Mapping
3. Set-Associative Mapping

Direct Mapping

Each main memory block is assigned to a specific line in the cache. Mapping is expressed as

i = j modulo C

Where i is the cache line number assigned to main memory block j.

If M = 64, C= 4

line 0 can hold blocks 0, 4, 8, 12,………..

line 1 can hold blocks 1, 5, 9, 13,………..

line 2 can hold blocks 2, 6, 10, 14, ………

line 3 can hold blocks 3, 7, 11, 15,………

Direct Mapping cache treats a main memory address as 3 distinct fields –

Tag identifier- The tag is stored in the cache along with the data words of the line.

Line number identifier- Line identifier specifies the physical line in cache that will hold the referenced address.

Word identifier (offset)-Word identifier specifies the specific word (or addressable unit) in a cache line that is to be read.

For every memory reference that the CPU makes, the specific line that would hold the reference (if it has already been copied into the cache) is determined. The tag held in that line is checked to see if the correct block is in the cache.

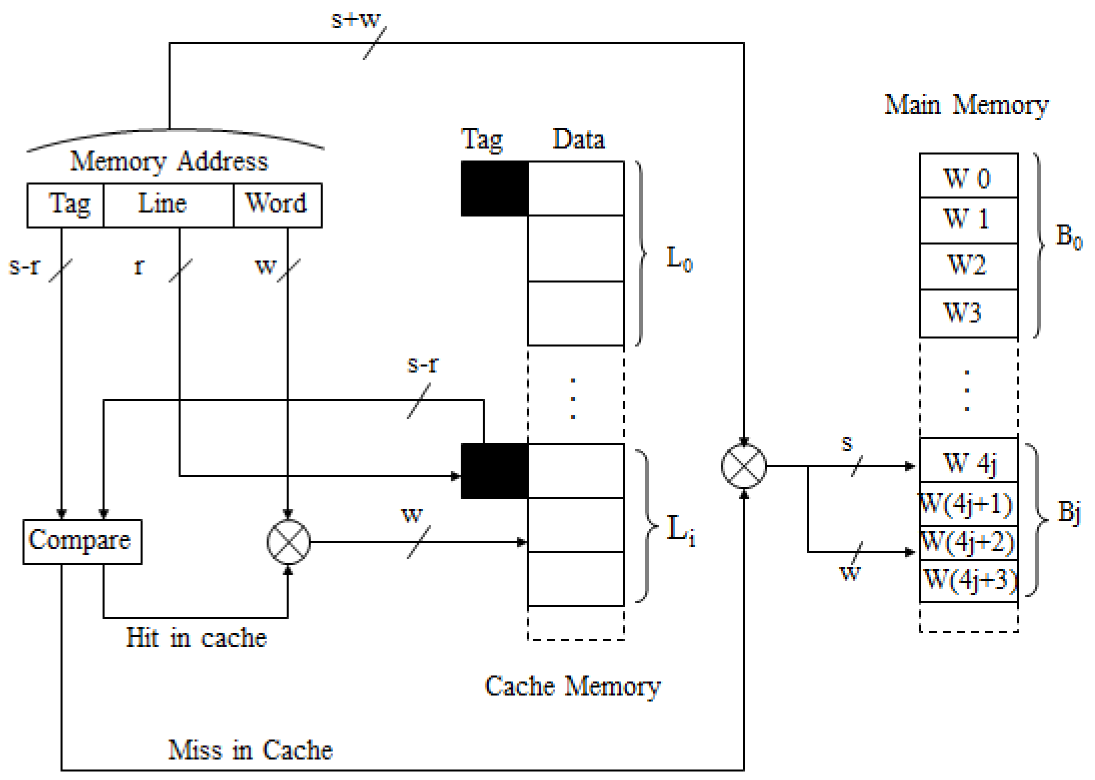


Fig: Direct Mapping Cache Organization

- **Advantages:**
  - — Easy to implement.
  - — Relatively inexpensive to implement.
  - — Easy to determine where a main memory reference can be found in cache.

- **Disadvantages:**
  - — Each main memory block is mapped to a specific cache line.
  - — Trough locality of reference, it is possible to repeatedly reference to blocks that map to the same line number.
  - — These blocks will be constantly swapped in and out of cache, causing the hit ratio to be low.

Associative Mapping

Associative mapping removes the drawback of direct mapping by providing main memory block to be loaded into any line of cache. Memory address logic interprets as a tag and a word field. Tag field identifies the block of main memory. to find whether the block is in cache, the cache control logic examines every line's tag for a match.

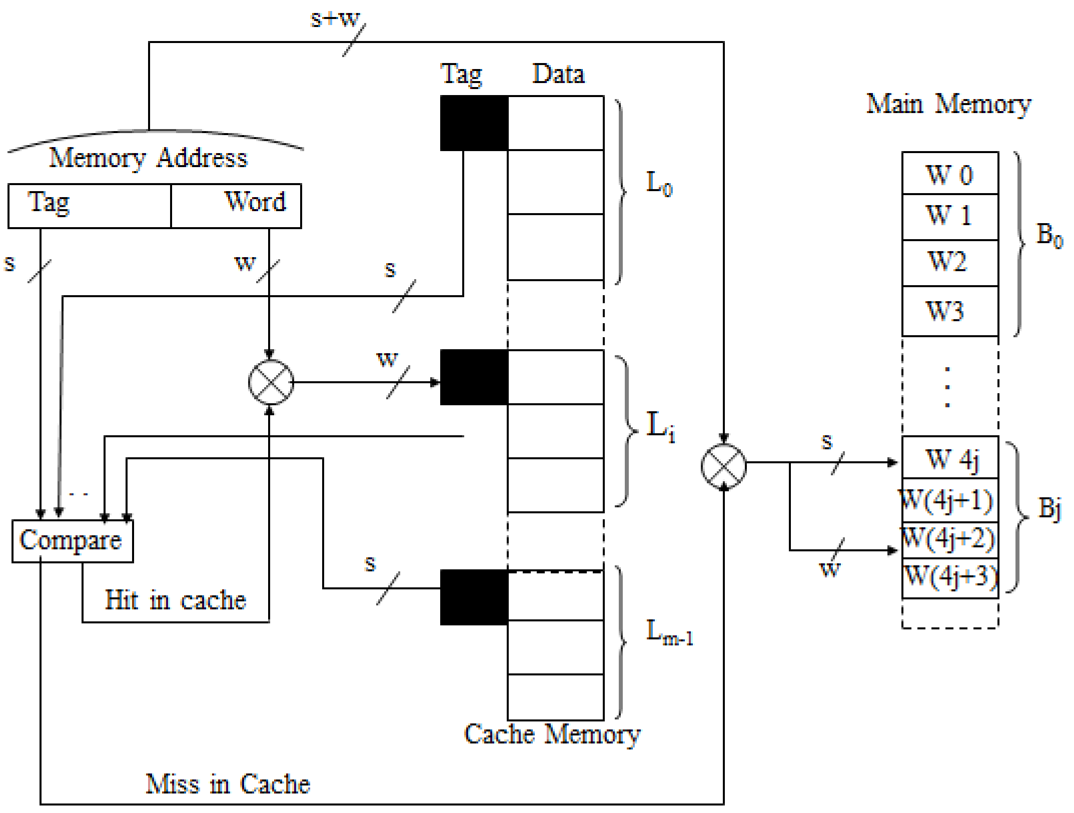


Fig: Associative Mapping

Advantage

Fast due to associative memory. Flexible to replace block when new block is read into cache.

Disadvantage

Need of complex circuit to examine the tags of all cache lines in parallel.

Set-Associative Mapping

Compromise between direct and associative mappings that builds on the strengths of both. Divide cache into a number of sets (v), each set holding a number of lines (k). A main memory block can be stored in any one of the k lines in a set such that

set number = j modulo v

Where, j is the assigned main memory block.

If a set can hold X lines, the cache is referred to as an *X-way set associative cache.* Most cache system today that use set associative mapping are 2- or 4- way set associative.

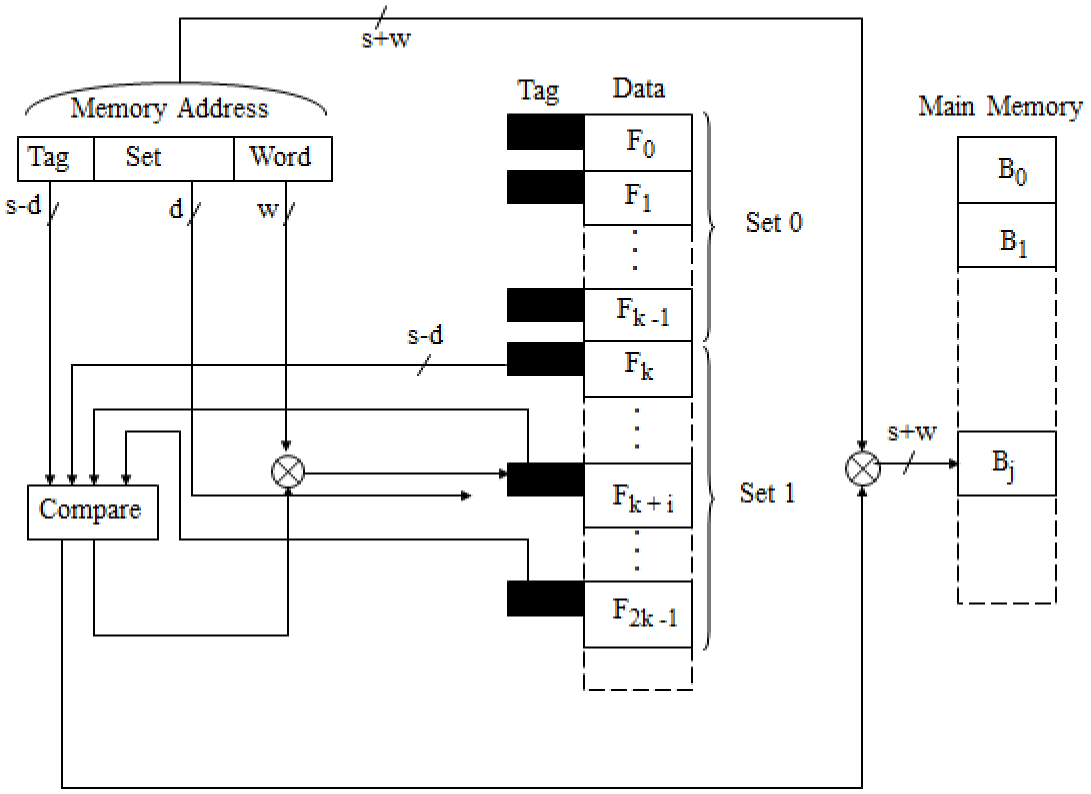


Fig: Set-Associative Cache organization

Replacement Algorithm

When a new block is brought into cache, one of the existing block must be replaced. There are four commonly used algorithms:

1. Least recently used (LRU)
2. Least frequently used (LFU)
3. First-in –First out (FIFO)
4. Random

With LRU, cache replaces the block in the set that has been in the cache for longest with no reference to it. With least frequently used that block is replaced in the set that has fewest references. It can be implemented by associating the counter with each line. FIFO replaces the block that has been in the cache for longest. it can be implemented by round robin or circular buffer technique. Random picks a line randomly and replaces that block in the set.

Write Policies:

Before replacement of block from cache it is necessary to examine if it has been changed in cache. if no, the block can be overwritten. If yes, then there is write operation in cache and it should be updated in main memory. Write policies are:

1. Write through
2. Write back

With write through, write operations are made to main memory as well as to the cache insuring that main memory is valid. Cache memory module can monitor traffic to main memory to maintain its consistency. Main disadvantage is it creates memory traffic and bottleneck.

An alternative is write back policies. In this technique update is made only in the cache. When an update occurs UPDATE bit associated with the line is set. Then when a block is replaced it is written back to main memory if and only if the update bit is set. Disadvantage of write back policy is portion of main memory is invalid. Thus access from I/O module is allowed only through cache.

Virtual Memory

In memory hierarchy, programmed data are first stored in auxiliary memory. Portion of data or program are brought into main memory as they are needed by CPU. Virtual memory is the concept used in large computers that permits user to construct a program as through a large memory space is available equal to total of auxiliary memory. Virtual memory gives programmer a illusion of having large memory. Each address referenced by the CPU goes through an address mapping from virtual address to a physical address in main memory. The address mapping is handled by mapping table.

Address Space and Memory space

An address used by programmer is called virtual address and set of such address is called address space. An address in the main memory is called location or physical address and set of such address is called memory space. An address space is allowed to be larger than memory space in computers with virtual memory,

E.g.

Main memory capacity= 32 k words (k=1024)

Physical address=2^{15}

Auxiliary Memory=2^{20} (1024k words)

Thus auxiliary memory has a capacity of storing information equivalent TO 32 main memories.

Address space, N=1024 k

Memory space, M= 32 k

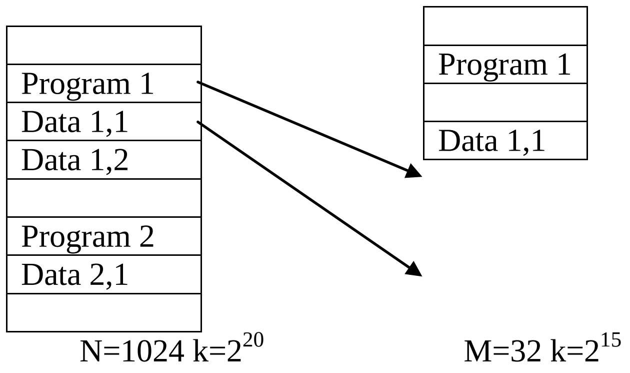


Fig: Relation of address and memory space in a virtual memory system

In virtual memory system programmers are said that they have total address space. The address field of instruction code has a sufficient number of bits to specify all virtual address. Here in the example CPU references data with 20 bit address but physical memory gives access to 15 bit address. Thus a table is needed to map 20 bit virtual address to 15 bit physical memory. Mapping is dynamic operation, which means that every address is translated immediately as a word is referenced by CPU.

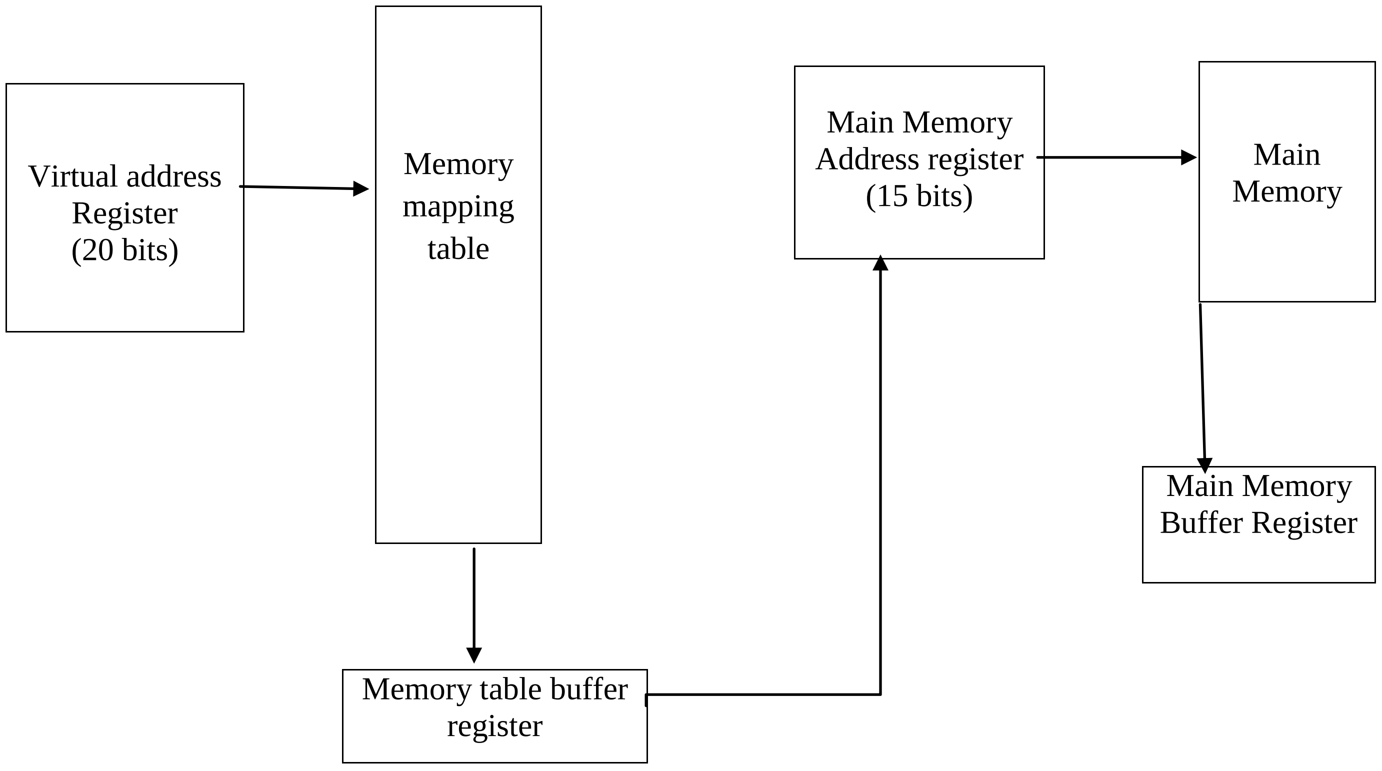

Virtual address Register (20 bits)
Memory mapping table
Main Memory Address register (15 bits)
Main Memory
Main Memory Buffer Register
Memory table buffer register

Fig: Memory table for mapping virtual address

Address mapping using pages: The table implementation is easy if information in address space and memory space is divided into fixed size. Physical memory is divided into groups of equal size is called blocks. Groups of address space of same size are called pages.

e.g.

For 1 k words page or block is divided into 1024 pages and 32 blocks.

Address space = 8k

Memory space= 4k

If we split them into 1 k words we obtain 8 pages and 4 blocks. At any given time up to four pages of address space can reside in main memory in any one of four blocks.

Page 0
Page 1
Page 2
Page 3
Page 4
Page 5
Page 6
Page 7
N= 8 k =2^{13}
Block 0
Block 1
Block 2
Block 3
M=4 k=2^{12}

Fig: Address space and memory space split into 1 k words

For mapping address space to memory space, virtual address is represented by two numbers: a page number address and a line with in page. For 2^p words, p bits gives page number. In above example each page is 2^{10} = 1024 words and virtual address has 13 bits. Thus 10 bits give line and higher 3 bits. Thus 10 bits give line and higher 3 bits give page number.

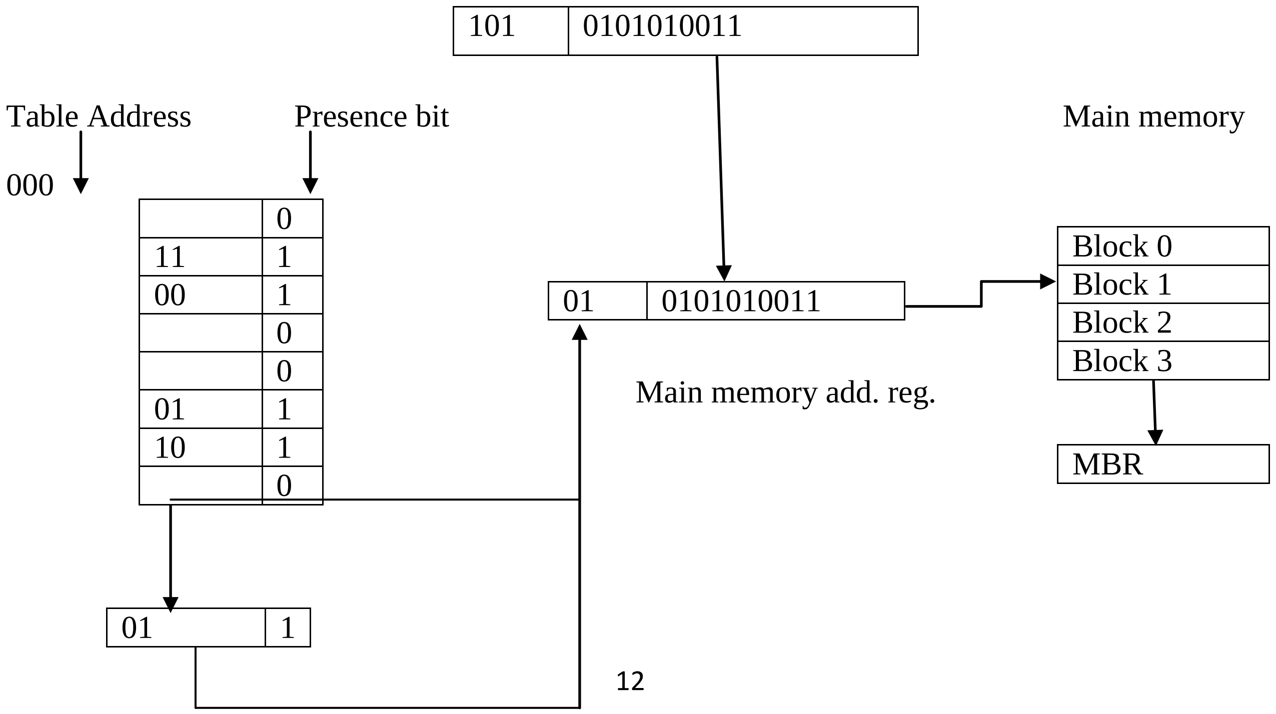


Fig: Memory table in a paged system

Note: Table address ranges from 000 to 111

Memory page table consists of 8 words, one for each page. The address in page table gives page no. and content of words gives block number where that page is stored. Table shows page 1,2,5,6 are available in main memory in blocks 3,0,1,2 respectively. Presence bit gives the auxiliary memory pages are transferred to main memory for 0 in presence bit indicates the page is not present in the main memory. CPU references words in 13 bits. The three high order bits specify page no. and also address of memory page table. For presence bit 1, block no. is transferred to memory table buffer register. The block no. is transferred to two higher bits of main memory address register. The line no. from virtual address is transferred to 10 lower bits. A record signal to main memory transfers the content of word to main memory buffer register ready to be used by CPU. If there is 0 , it means that the main memory does not have the word . Thus a call is made to fetch the required page from auxiliary memory to main memory.

Associative memory page table:

In above mapping technique memory accommodation depends upon the blocks allocation in main memory. In this case some pages remain unused. Thus efficient way to organize page table is to construct it with a number of words equals to the number of blocks in main memory. In this way the size of memory is reduced and each location is fully utilized. This method can be implemented by means of an associative memory.

Each word in main memory contains page number with its corresponding block number. Page field in each word compared with page no. in virtual address. If a match occurs the word is read from memory and its corresponding block number is extracted.

101	Line number

111 00

001	11
010	00
101	01
110	10

Fig: Associative memory page table

Associative memory consists of two fields. The first three bits specify page number and last two bits specify block number. Virtual address is in argument register. Page no. bits are compared with all pages nos. in page field of associative memory. If page no. id found the 6-bit word is read out from memory. The corresponding block number is transferred to main memory address register. If no match occurs, a call is generated to bring required page from auxiliary memory.