What are the differences among Sequential access, direct access & random access?

→ Sequential Access:

Memory is organized into units of data called records. Access must be made in a specific linear sequence.

Direct Access:

Individual blocks & records have a unique address based on physical location. Access is accomplished by direct access to reach a general vicinity plus sequential searching, counting or waiting to reach the final location. Again, access time is variable.

Random Access:

Each addressable location in memory has a unique, physically wired-in addressing mechanism. The time to access a given location is independent of the sequence of prior accesses & is constant

What are the differences among direct & associative mapping?

→ Direct mapping:

Each block from main memory has only one possible place in the cache organization in this technique.

$i = j \bmod m$

where j = main memory block number

i = cache block number

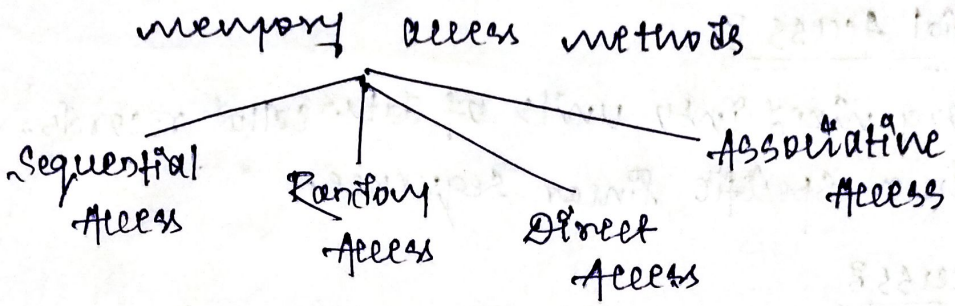
m = number of blocks in the cache

Associative mapping:

Here the mapping of the main memory block can be done with any of the cache block. The memory address has only 2 fields here: word & tag. This technique is called as fully associative cache mapping.

Q3. Explain the methods of accessing the units of data in memory system?

→



Sequential Access:

In this method, the memory is accessed in a specific linear sequential manner, like accessing like a in a single linked list The access time depends on the location of the data.

Random Access:

In this method, any location of the memory can be accessed rand like accessing in Array. Physical locations are independent in access methods

Direct Access:

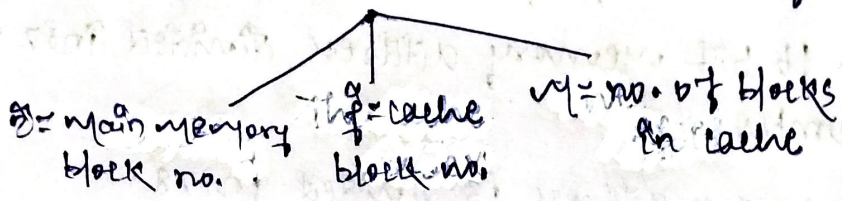
In this method, individual blocks/records have a unique addres based on physical location. The access time depends on both the memory organization & characteristics of storage technology The access is semi-random or direct.

Associative Access:

In this memory, a word is accessed rather than its address. This access method is a special type of random access method. Application of thus associative memory access is cache memory.

For a direct-mapped cache, a main memory address is viewed as consisting of three fields. List & define the three fields.

→ Three fields direct-mapped cache: $s = r + w$ module.



Consider a direct mapped cache of size 32KB & block size 32 byte. The CPU generates 32 bit addresses. The number of bits needed for cache indexing & the number of tag bits are:

→ Size of cache = 32KB = 32000 byte = $2^{s+w} = 2^{15}$ byte

Block size = line size = 2^w = 32 byte = 2^5 byte

Total address length = $(s+w)$ = 32 bits

So, Total no. of blocks = $\frac{\text{cache size}}{\text{Block size}} = \frac{2^{15}}{2^5} = 2^{10}$

Tag Bits:

Total address bits = 32 Bits

Index bits = 10 bits

Block offset bits = log2 (Block size) = log2 (32) = 5 bits

Tag Bits = (Total address bits) − (Index Bits) − (Block offset bits)

= (32 − 10 − 5) Bits

= 17 bits

Index bits:

Index bits = log2 (No. of Blocks)

= log2 (2^{10}) = 10 bits

8.6 Consider a machine with a byte addressable main memo of 2^{16} bytes & block size of 8 bytes. Assume that a direct mapped cache consisting of 32 lines is used with this mach

(a) How is a 16 bit memory address divided into tag, line n & byte number?

→ The 16-bit memory address is divided into:

- Tag: 6 bits (bits 10-15)
- Line number: 5 bits (bits 5-9)
- Byte number: 3 bits (bits 0-2)

(b) In what line would bytes with each of the following addresses be stored?

→ 0001 0001 0001 1011

* Stored in line 11 (binary 01011)

1100 0011 0011 0100

* Stored in line 4 (binary 00100)

1101 0000 0001 1101

* Stored in line 13 (binary 01101)

1010 1010 1010 1010

* Stored in line 10 (binary 01010)

(c) Suppose the byte with address 0001 1010 0001 1010 is stored in the cache. What are the addresses of the other bytes stored along with it?

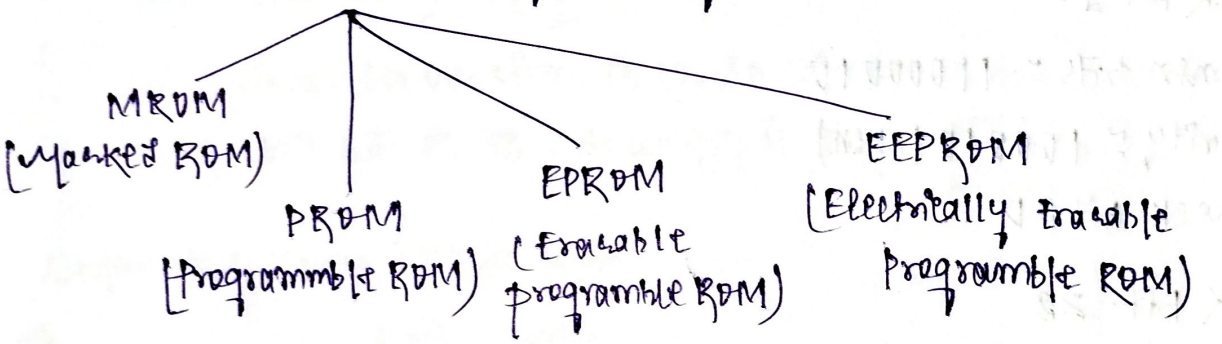
→ 0001 1010 0001 1011
0001 1010 0001 1100
0001 1010 0001 1101
0001 1010 0001 1110
0001 1010 0001 1111
0001 1010 0010 0000
0001 1010 0010 0001

(d) How many total byte of memory can be stored in the cache?
→ The cache has 32 lines, & each line can store a block of 8 byte. Therefore the total number of byte that can be stored in the cache is $32 \times 8 = 256$ byte.

(e) Why is the tag also stored in the cache?
→ To check if the requested memory address is present in the cache or not.

What is the difference between DRAM & SRAM in terms of application?
→ SRAM is used in specialized applications such as cache memory of computer systems. On the other hand, DRAM is used as the main memory in most personal computers.

Explain different types of ROM.
→ ROM (Read-Only-Memory)



How is the Syndrome for the Hamming code interpreted?
→ If the syndrome contains all 0s, no error has been detected. If the syndrome contains one & only one bit set to 1, then an error has occurred in one of the 4-check bits, & no correction is necessary.

Explain Semi-conductor main-memory organization?
→ In a semi-conductor memory chip, each bit of binary data is stored in a tiny circuit called a memory cell consisting of one to several transistors. The memory cells are laid out in rectangular arrays on the surface of the chips.

Q11. If there are m input lines, n output lines for a decoder that used to uniquely address a byte addressable 1KB RAM, then minimum value of m+n is ____.

→ We need 2^{10} outputs to map 1KB RAM.

For this we need 10×2^{10} decoder.

Here $m = 10$ & $n = 2^{10}$

∴ $m+n = 10 + 2^{10} = \boxed{1034}$

Q12. Suppose an 8-bit data word stored in memory is 11000010. Using Hamming algorithm, determine what check bits would be stored in memory with the data word. Show how you got your answer.

→ Data bits = 1 1 0 0 0 0 1 0

Bit index = 8 7 6 5 4 3 2 1

check bit 1:

Data bits = 1100001 0

Parity = 100010 (even)

check bit 2 0

check bit 2:

Data bits = 1100001 0

parity = 1100 (even)

check bit 8 1

check bit 4:

Data bits = 1100001 0

Parity = 10010 (even)

check bit 8 1

∴ Hamming code for 11000010 is Data bits = 11000010

check bits = 0 11

Hamming code = 1100 11011

For the 8-bit word 00111001, the check bits stored with it would be 0111. Suppose when the word is read from memory, the check bits are calculated to be 1101. What is the 8-bit word that was read from memory?

→ Given 8-bit word 00111001.

Encoded in d8=0, d7=0, d6=1, d5=1, d4=1, d3=0, d2=0, d1=1

Parity is calculated as 0111 i.e, p8p4p2p1.

p1=1, p2=1, p4=1, p8=0

Encode the string as d8d7d6d5p8d4d3d2p4d1p2p1.

i.e, 00110001111.

Check bit at other end is 1101.

c1=1, c2=0, c4=1, c8=1

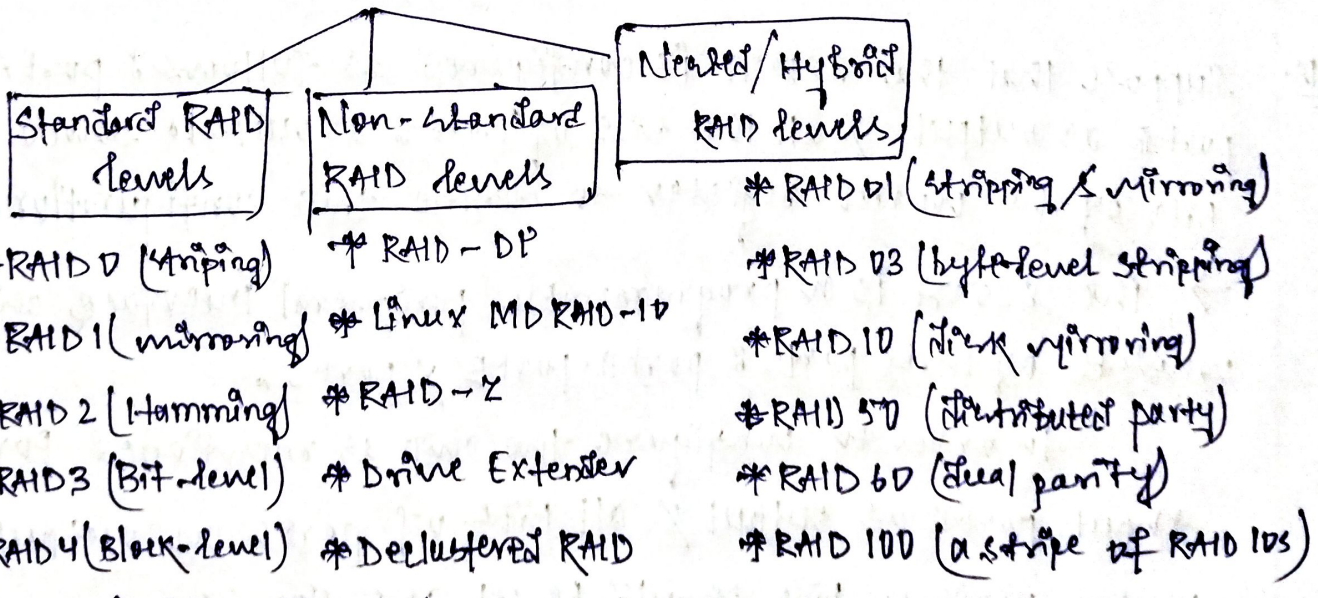
xor (1101, 0111) = 1010

Its decimal conversion is 10. So 10th bit is changed.

Here 10th bit is d6, so we flip it & got 00011001.

Define the seven RAID levels

→



Q15. RAID configuration of disk are used to provide

→ (a) Faunt tolerance

(b) High speed

(c) High data density

(d) None of above

Q16. What are the major functions of an I/O module?

→ The major functions include:

* processor communication
* Device communication
* Control & timing
* Data Buffering
* Error Detection

Q17. List three broad classification of external/peripheral devices?

→ Three classification of peripheral devices are:

* Human Readable
* Machine Readable
* Communication

Q18. Suppose that the 8255A is configured as follows: port A as input port B as output & all the bits of port C as output. Show the bits of the control register to define this configuration.

→ The 8255A is a programmable peripheral interface chip that consists of three ports: port A, port B & port C.

In order to configure the chip as mentioned port A as inputs port B as output & all bits of port C as outputs the control register bits should be set as follows:

Control Register Bit configuration :

Bit 7 : Mode set bit group selection

Bit 6 : port C upper/lower group selection

Bit 5 : port C mode selection

Bit 4 : port B mode selection

Bit 3 : port A mode selection

Bit 2 : port C bit 3

Bit 1 : port C bit 2

Bit 0 : port C bit 1

To configure the 8255A as described, the control register bits should be set as follows:

Bit 7 : 0 (mode 0)

Bit 6 : 0 (lower group selection)

Bit 5 : 1 (mode 1 for port C)

Bit 4 : 1 (mode 1 for port B)

Bit 3 : 1 (mode 1 for port A)

Bit 2 : 1 (output for port C bit 3)

Bit 1 : 1 (output for port C bit 2)

Bit 0 : 1 (output for port C bit 1)

The 8255 programmable peripheral interface is used as described below :

(i) An A/D converter is interface to a microprocessor through an 8255.

(ii) Two computers exchange data using a pair of 8255s.