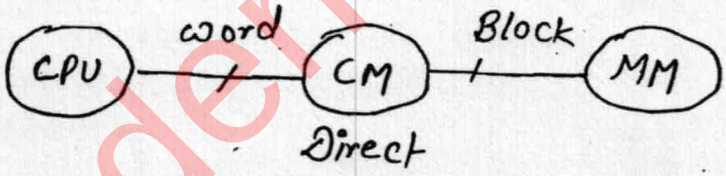
"updating Techniques" :-

* According to Heirarchy design, CPU always accessing the Data from the cache memory.

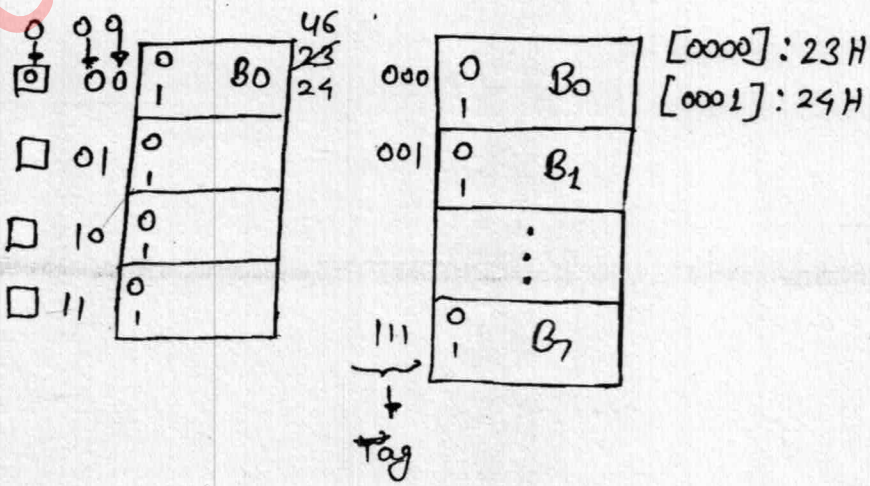
:- In this regard CPU check the, availability of a datablock in the cache memory.

When It is in the cache, then operation become Read Hit or write Hit otherwise operation become Read Miss or write Miss.

* When Miss occurs then the Respective Block will be transferred from main-memory to cache called Read Allocate or write Allocate. After the Allocation CPU performs read and write operation only on a cache memory.



Direct



$K \bmod N = i$

$\{0 \bmod 4 = 0\}$

Prog:-

I_1: Mov r_0, [0000] ; B_0 – 0^{th} Byte → r_0

I_2: Add r_0, #23 ; $r_0 + 23 \rightarrow r_0$

I_3: Mov [0000] r_0 ; $r_0 \rightarrow B_0 - 0^{th}$ Byte

Execution:-

I_1: Read Hit ; $r_0 = 23H$

I_2: $r_0 + 23 \rightarrow r_0$; $r_0 = 46H$

I_3: write Hit :- CM

[0000] : 46H

MM

[0000] : 23

CM

[0000] : 46

{ Cache Coherence }

* In the above code during the execution of I_3 instruction, CPU updates only the cache m/y datablock. the corresponding updation is not performed in the main-m/y so same address contain different values at different places. This kind of data inconsistency problem in the m/y system is called as 'cache coherence'

- Coherence causes the data loss described below:

eg:-

I_4 : Mov r_0, [1000]; B_4 - 0th Byte → r_0
B_4

I_5 :

I_6 :

I_7 :

I_8 : Mov r_0, [0000]; B_0 - 0th Byte → r_0
B_0

Example:-

I_4 : Read Miss : Read-Allocate

{ k mod N = i
4 mod 4 = 0 }

CM: B_0 B_4
↳ updated
Data is lost

I_8 : Read Miss : Read-Allocate

{ k mod N = i
0 mod 4 = 0 }

CM: B_4 B_0

{ B_0 Contain old data }

Data Loss

★ To Handle the above data-Loss Problem, writting policies are used in the cache design.

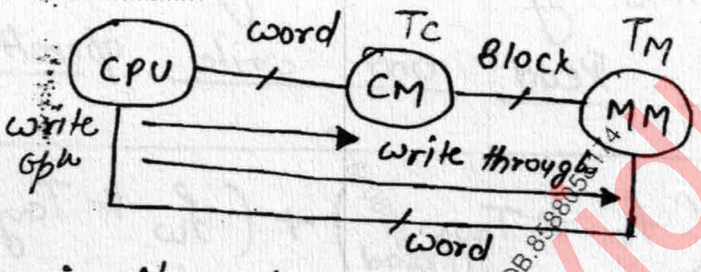
These are two types:

1) write through
2) write Back

:- Write Through :-

* In this protocol, CPU perform the simultaneous write operation in both cache memory and main-memory so 'No coherence' inclusion is always success.

* Inclusion means lower level memory data always become the subset of a Higher level memory data.



:- No-coherence

:- Inclusion success

* T_w [Simultaneous write op^w] = max [word updation time in C.M. , word updation time in M.M.]

* Amount of the Time required to read one word data from the memory is called as Read cycle time i.e.

$$T_{avg\ read} = H_r T_c + (1-H_r)(T_m + T_c)$$

- H_r : Read Hit
- T_c : Read data
- $(1-H_r)$: Read Miss
- T_m : Read allocate
- T_c : Read data.

Amount of the time required to write one word data in the memory is called as write cycle time i.e.

$$T_{avg\,write} = H_w T_w + (1-H_w)(T_m + T_w)$$

- H_w → write Hit
- T_w → Simultaneous write
- $(1-H_w)$ → write miss
- T_m → write allocate
- T_w → Simultaneous write

* Average Access time of the memory when considering both Read and write operation is

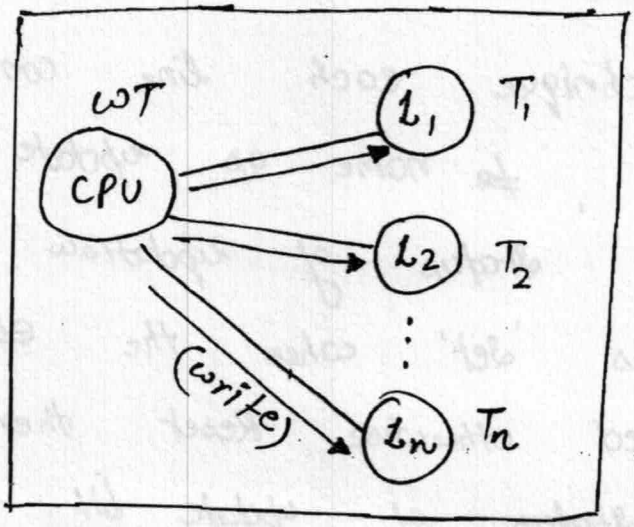
$$T_{avg\,WT} = (f_r * T_{avg\,read}) + (f_w * T_{avg\,write})$$

- $T_{avg\,WT}$ → write through
- f_r → # read op^n
- f_w → # write op^n

$$\eta_{WT} = \frac{1}{T_{avg\,WT}} \text{ words/sec.}$$

* Note :-

when the write-Through Technique is implimented in a simultaneous access memory organization then Hit ratio for write operation always become '1' because Data Block is need not be required in level -i m/y in this organization



$$\left\{ \begin{array}{l} H_w = 1 \\ T_{avg\,write} = T_n \end{array} \right\}$$

Note :-

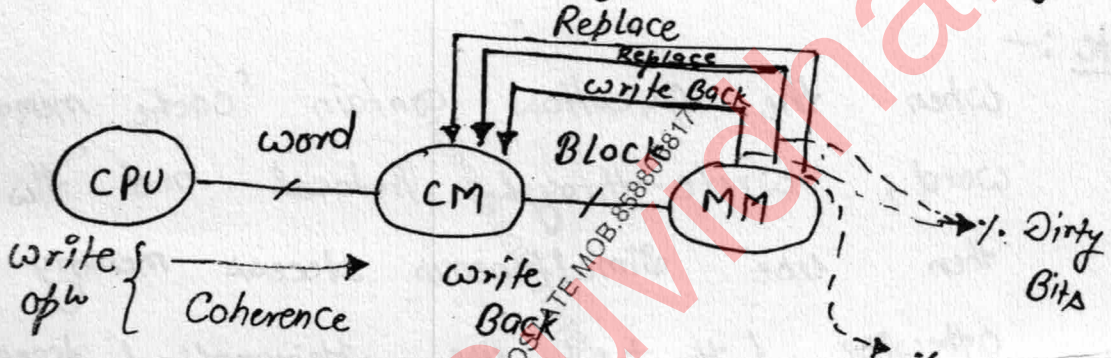
when the question contain 'cache memory' word, write-through Protocol and "$H_w = 1$" then use simultaneous access memory org^n otherwise { $H_w \neq 1$ } use Heirarchical access m/y org^n.

:- If H_w not given, then consider it as 1' and take simultaneous m/y org^n. eg. $H_r = 70\%$.
$H_r = 0.8$
Remaining write op^n

:- write-Back :-

In this Protocol, CPU perform the write operation only in the C.M. So coherence present but coherence doesn't cause the data loss because update bit is present to indicate the dat status of Block.

* In this technique each line contain a extra bit . to name as update bit to indicate the status of updation that is this bit is 'set' when the ~~the~~ Cache Block is updated otherwise Reset therefore Based on the status of update bit CPU write-Back the Data-Block into a main-mly before Replacement. therefore Data is safe.



update bit

0 (Clean) — % clean bits (read op^w)

1 (Dirty) — % dirty bits (write op^n)

$$T_{avg_{read}} = H_r T_c + (1-H_r)\left[\%\ dirty\ Bits\ (T_m + T_m + T_c) + \text{Read Data}\right]$$

$$\%\ clean\ Bits\ (T_m + T_c)$$

Read Hit — Read Data — Read Miss — write Back — ~~write~~ Read allocate — Read allocate — Read Data

:- write cycle time is :

$$T_{avg_{write}} = H_w T_c + (1-H_w)\left[\%\ dirty\ Bits\ (T_m + T_m + T_c) + \text{write Data}\right]$$

$$\%\ clean\ Bits\ (T_m + T_c)$$

write Hit — write Data — write miss — write Back — write Allocate — write allocate — write data.

:- Average Accessing time, when considering Both Read & write operation is :

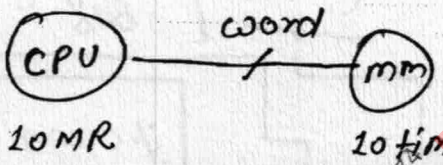
$$T_{avg_{write\ Back\ (WB)}} = (f_r * T_{avg_{read}}) + (f_w * T_{avg_{write}})$$

"Multi-level cache organization" :-

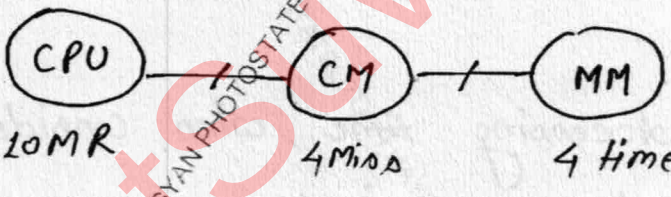
* Multi-level cache orgn is used in the computer system to minimize the miss penality.

:- 'Miss Penality' means time required to transfer the data from Higher level to lower level due to a miss operation.

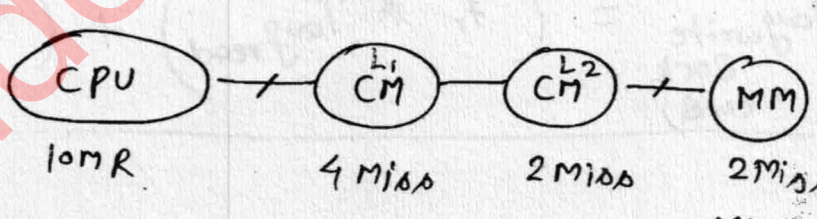
System old :



System with cache :



System with multi-level cache :



* In the multi-level cache design, capacity is in a increasing order and associativity is in a decreasing order from the Level-1 cache to Level-2 cache.

* Two kinds of miss-rate are calculated in the multi-level cache design.

1) GMR (Global miss rate)
2) LMR (Local miss rate)

$$\begin{cases} GMR = \dfrac{\#\text{ misses in cache}}{\text{Total \# CPU generated references.}} \\ LMR = \dfrac{\#\text{ misses in cache}}{\text{Total \# Access to cache.}} \end{cases}$$

$$\begin{cases} GMR_{L_2} = \dfrac{2}{10} = 0.2 \\ LMR_{L_2} = \dfrac{2}{4} = 0.5 \end{cases} \rightarrow (\text{or}) \dfrac{\#\text{ miss in } L_2}{\#\text{ miss in } L_1}$$

$$\begin{cases} GMR_{L_1} = \dfrac{4}{10} = 0.4 \\ LMR_{L_1} = \dfrac{4}{10} = 0.4 \end{cases}$$

* Average Access time of the m/y is calculated in ~~time~~ terms of Hit time, miss rate and miss-Penality is as follows:

:- **

$$\boxed{T_{avg} = \text{Hit time}_{L_1} + (\text{Miss Rate}_{L_1} * \text{Miss Penality}_{L_1})}$$

$$\boxed{\text{Miss Penality}_{L_1} = \text{Hit time}_{L_2} + (\text{Miss Rate}_{L_2} * \text{Miss Penality}_{L_2})}$$

$$\boxed{\text{Miss Penality } L_2 = \text{M.M. Access time}}$$

(number of) memory stalls created per instruction during the program execution is calculated as:

** # memory stalls/Instn = (# misses$_{L_1}$/Instn * Hit time$_{L_2}$) + (# misses$_{L_2}$/Instn * miss Penalty$_{L_2}$)

↓

M.M. Access time.

Dated 22/08/2017

"Types of cache misses":-

1.) "Compulsory miss" [Cold start miss] or [first reference miss].

:-

* This miss will occur when the very first referenced blocked is miss in cache. These misses can be minimized by increasing the block-size.

2.) "capacity miss":-

This miss will occur when the cache is full. These misses can be minimized by increase the cache-size.

3.) "conflict miss" [Collision miss or Inference miss]:-

This miss will occur when too many Blocks are mapped into same cache line or same cache set. these cache-misses can be minimized by double the Associativity.

Que:- Consider 2-Level m/y organization, Level-1 m/y is cache-memory, Its access time is 40ns. main-memory is a Level-2 m/y and Its access time is 200ns. write through protocal is used in the cache design to Handle the cache Coherence. In the Application program 70% of requests are used for read operation and remaining are used for write op^n. Hit ratio for read op^n is 80%. what is the average access time op^n of m/y when considering both read & write operation.

Soln:- Given:-

L_1 : CM – T_c : 40ns

L_2 : MM – T_m : 200ns

f_r : 70% H_r = 80% = 0.8

f_w : 30% H_w = 1 {Simultaneous Access m/y organization}

write-through:-

$T_{avg_{read}} = H_r T_c + (1-H_r) T_m$

$= 0.8 \times 40 + (1-0.8) * 200$

$= 72ns$

$T_{avg_{write}} = H_w T_m$ ← ∵ T_m is max

$= 1 * 200$

$= 200ns$

$T_{avg_{WT}} = (f_r * T_{avg_{read}}) + (f_w * T_{avg_{write}})$

$= (70\% * 72ns) + (30 * 200ns) = 110.4ns$

Que :- Consider a write-Back cache with the Hit ratios of read & write operation is 70% and 80% respectively. Cache memory is 8 times faster than main-m/y. mm access time is 600 ns. In the System 60% of requests are used for Read operation and remaining are used for write operation what is the avg access time of the memory when considering both read & write operations.

Soln :- write-Back [Hierarchical]

$H_r = 70\%$

$H_w = 80\%$

$T_m = 600\,ns$ $\quad T_c = \frac{600\,ns}{8} = 75\,ns$

$f_r = 60\%$ [clean bits %]

$f_w = 40\%$ [% dirty bits].

$$T_{avg\,read} = H_r T_c + (1-H_r)\left[\%\,\text{dirty Bits}\,(T_m + T_m + T_c) + \%\,\text{clean Bits}\,(T_m + T_c)\right]$$

$$= (0.7)*75 + (0.3)\left[0.4(600+600+75) + 0.6(600+75)\right]$$

$$= 327\,ns$$

$$T_{avg\,write} = H_w T_c + (1-H_w)\left[\%\,\text{dirty Bits}\,(T_m + T_m + T_c) + \%\,\text{clean Bits}\,(T_m + T_c)\right]$$

$$= 243\,ns$$

$= (0.8 * 75) + (1-0.8)[(0.4)(600+600+75) + (0.6)(600+75)]$

$= 243 \text{ ns}$

$T_{avg\,WB} = (fr * T_{avg\,read}) + [fw * T_{avg\,write}]$

$= (0.6 * 327) + [0.4 * 243]$

$= 293.4 \text{ ns}$

Que :- Consider the following m/y specifications :

memory	Access time (ns)	Hit ratio
I-cache	10 ns	0.8
D-cache	15 ns	0.7
MM	150 ns	0.9
SM	500 ns	1

In the application program 60% ~~instruction~~ of requests are used for instruction accessing and 40% requests are used for data accessing what is the time required to access the program information.

Soln :- Heirarchical System:

60% – I-cache (Instn)

40% – D-cache (Data)

$$T_{avg_{Instn}} = H_c T_c + (1-H_c) H_m (T_m + T_c) + (1-H_c)(1-H_m) H_s (T_s + T_m + T_c)$$

$$= (0.8 * 10) + (0.2) * 0.9 (150 + 10) + (1-0.8)(1-0.9)(1)(150 + 500 + 10)$$

$$= 50np$$

Time required for Instn access $= \left[f_{instn\ req.} * T_{avg_{Instn}} \right]$

$$= [0.6 * 50np] = 30ns$$

$$T_{avg_{Data}} = H_c T_c + (1-H_c) H_m (T_m + T_c) + (1-H_c)(1-H_m) H_s (T_s + T_m + T_c)$$

$$= (0.7 * 15) + (1-0.7) 0.9 (150 + 15) + (1-0.7)(1-0.9)(1)(500 + 150 + 15)$$

$$= 75np.$$

Time required for Data access $= \left[f_{Data\ req.} * T_{avg_{Data}} \right]$

$$= [0.4 * 75ns]$$

$$= 30np$$

Total T_{avg} $= 30 + 30$

$= 60np.$

Que :- Consider a memory design where CPU generates 100 m/y references in which 40% miss operation occured in level-1 cache and 10 miss operation occured in level-2 cache. Hit time of Level-1 cache is 3 cycles Hit time of level-2 cache is 15 cycles. miss penality from main m/y to level-2 cache is 150 cycles. In the Program each Instruction uses 1.75 m/y references:

a) what is the Avg. Access time of the m/y

b.) what are m/y stall cycles per Instn.

Soln :-

a) T_{avg} = Hit time$_{L_1}$ + (Miss Rate$_{L_1}$ * miss Penality$_{L_1}$)

miss Penality L_1 = Hit time$_{L_2}$ + (Miss Rate$_2$ * miss penality$_{L_2}$)

L_1 miss = 40%

L_2 miss = 10

Hit time of L_1 = 3 cycles

Hit time of L_2 = 15 cycles)

= 3 cycles + (0.4 *

miss Penality$_{L_1}$ = 15 + $\left(\frac{10}{40}\right)$ * 150 → misses of L_1

= 52.5 cycles

T_{avg} = 3 + (0.4 * 52.5)

= 24 cycles.

b) Avg m/y stalls/ Instn = (# misses L_1 / Instn * Hit time L_2) + (# misses L_2 / Instn * miss penalty L_2)

1.75 MR — 1 Instn

100 MR — ? # Instn

$\Rightarrow \frac{100}{1.75}$

$= 57.14$

$= \underline{58}$ Instn for 100 m/y references.

Level-1 m/y 58 instn experienced 40 miss.

1 instn — # ? miss

$\Rightarrow \frac{40}{58} = L_1$

Level-2 m/y

58 instn — 10 misses

1 instn — # ? misses

$\Rightarrow \frac{10}{58} = L_2$

Putting the value :-

$\Rightarrow \left(\frac{40}{58} * 15\right) + \left(\frac{10}{58} * 150\right)$

$= 36.197$ cycle

"I/O organization" :-

★ Any Device which is connected to a detechable ports of a CPU is called as peripheral device, these are two kinds :

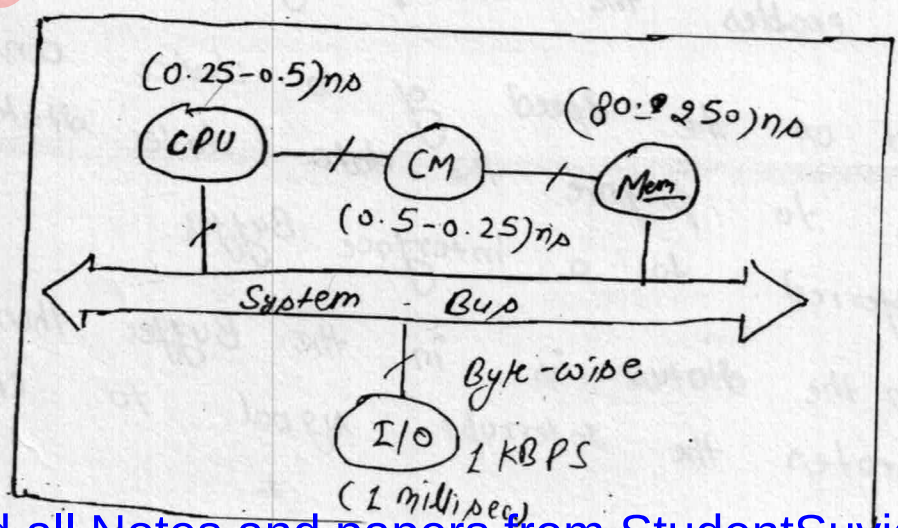
1) Input devices
2) output devices.

★ I/o devices are electro-magnetic components and CPU is a electronic components. So there is a difference existed in terms of operating modes, Data Transfer Rate and word formats

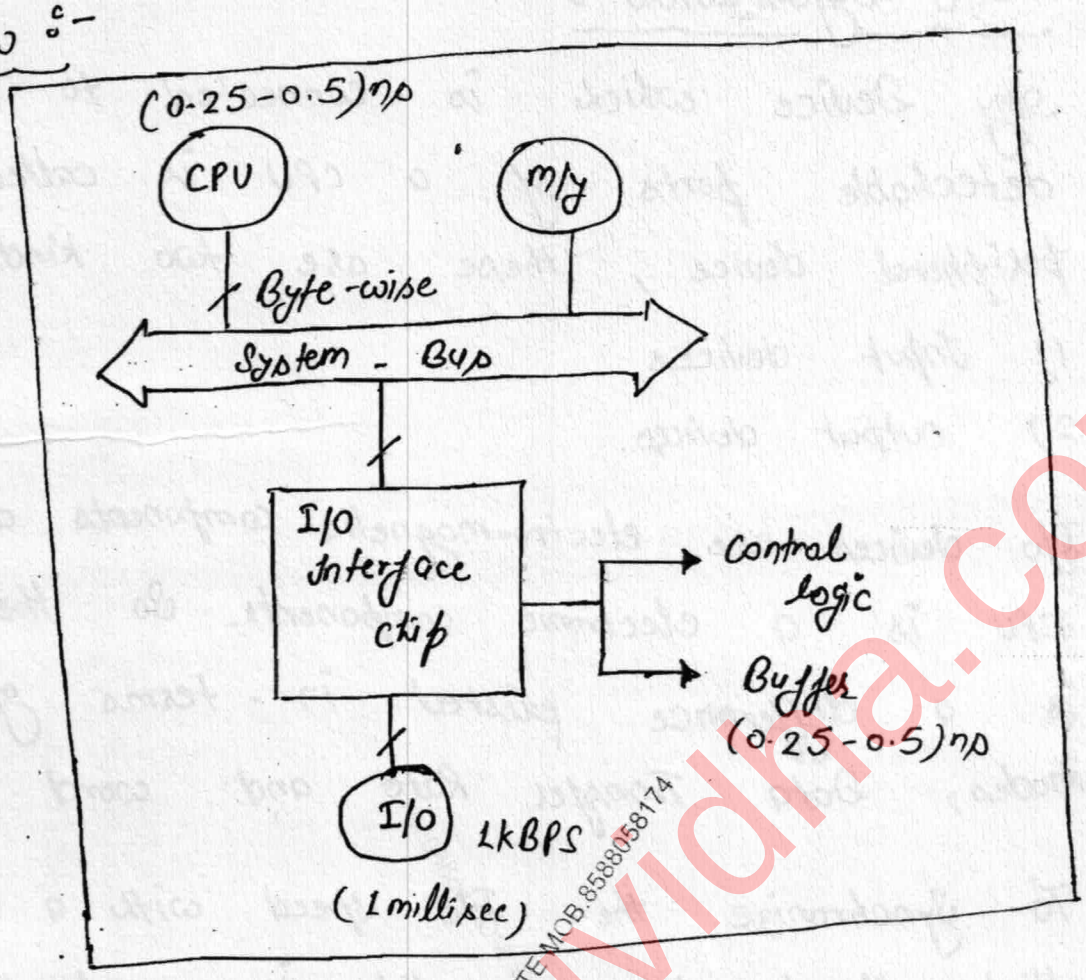
★ To Synchronize the I/o speed with a CPU, High Speed Interface chip is used name as I/o interface or I/o module.

★ I/o Interface chip is responsible for I/o operations therefore in the Computer design all the I/o devices are connected to a CPU via I/o interface chip.

System old :



System New :-



Accessing sequence :-

* CPU initializes the I/O interface along with a I/O command, later busy in other useful task.
* I/O Interface control logic interprets the I/O command and enables the corresponding port for the operation.
* Based on the speed of a device consume the time to prepare the data later status will be transferred to a interface Buffer.
* When the status is in the Buffer then Interface generates the interrupt signal to CPU and

waiting for acknowledgment signal.

★ After Recieving the ack. signal Buffer content will be transferred to CPU with High speed therefore,

★ Speedgap is synchronized.

★ Different I/O interface chips used in the computer design:

- 8255 PPI [Programmable peripheral Interface]
- 8251 USART [universal synchronous Asynchronous Reciever - Transmitter]
- 8259 A INT. Controller
- 8237 / 8257 DMA.

"I/O Modes" :—

Three different I/O transfer modes used in the Computer design to transfer the data from I/O to other Components of computer (CPU & m/y).

1) Programmed I/O
2.) Interrupt v IO Driven
3.) DMA.