classmate
Date
Page

⇒ What is CO ?

→ CO is the conselt with the way the hardware components Operate and the way they are connected together to form compu. system.

→ The various components are assumed to be in the place and task is to investigate

The computer part operates with intended

⇒ Computer Architecture -

→ It is conselt with the structure and behaviour of the computer as seen by user.

→ It includes the information formats the instruction set and techniques for addressing the memory.

→ The Architectural design of computer system is consult with specification of various functional modules such as processors, memories and structuring them together in comp. system.

→ Exa. of computer Architecture –

(1) Von-Neumann Architecture
(2) Harverd "

⇒ Computer Design –

→ Computer design is consult with hardward design of computer.

→ It covers the determination of what hardware should be used and how path is connected.

→ It is sometimes reffered to as comp. Implementation

⇒ Von-Neumann Architecture –

→ It describes a general framework or structure that a computer's harware, programming and data should follow.

→ Von-neumann envisioned the structure of computer system with following components.

(1) Central Arithmatic Unit.
(2) Memory
(3) Control Unit.

(4) Man-Machine Interface

→ The most note-worthy concept of Von-nueman Arch. was store programme principle.

⇒ store programme principle -

→ It holds the data as well as instructions used to menipulate the data.

→ The data and instructions should be stored together in the same memory area of the comp. and instructions are carried out sequencial manner, One instruction at a time.

⇒ Harvard Architecture -

→ It uses physically seperate storage and signal pathways for their instructions and data. The term Originated from harvard mark-I relay base comp.

→ It store instructions on punch tape (with 24-bits) and data on relay latches. (24 digits wide)

→ In such architecture, CPU can read both data and instruction from memory at same time, leading to in increase in memory band-width.

→ Ex. of harvard architecture -
Micro-~~processor~~ controller based comp.

⇒ Digital signal processing based comp. system -

→ The data representation consists of four key elements.
(1) Data type.
(2) Representation of data.
(3) conversion ~~of~~ into info.

→ The term data refers to factual info. used for analysis and resoning. It has no meaning in itself. On the other hand on information is a collection of data that needs to be communicated.

→ Ex. Binary information is stored in memory on processor registers.

classmate

Date
Page

⇒ Registers -

→ It contains data on control information. Control info. is a bit or a group of bits used to specify the sequence of command of signals, needed for menilpulation of data in Other registers.

⇒ Data -

→ It can be defined as the nos. and binary coded information, that are operated onto achieve required computation results.

⇒ Assignment -

→ Diff. b/w von-neumann and harverd architecture.

⇒ Data types—

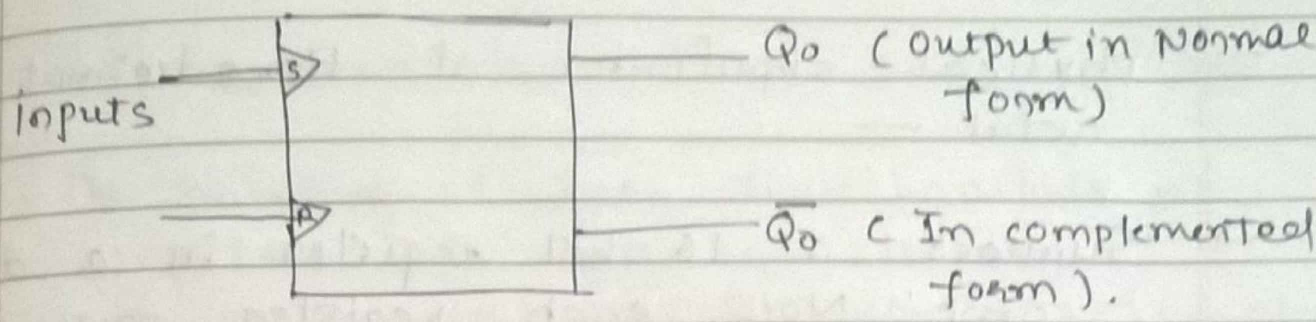
→ The data types found in the registers of digital computers mainly classified as (1) Numbers used in numeric computation
(2) latters of the alphabet used in data processing
(3) Other descreate symbols used for specific purposes

→ All types data accepts binary nos are represented in computer registers in binary coded form. This is because resisters are made up of flip-flops (flip-flops are two state devices that can store only 0's and 1's.
The storage element is employed in clock scquencial circuits are called flip-floap. A flip-floap stores 1-bit of information.)

⇒ Clock sequencial circuits-

→ In such circuits, the output is determined by periodic clock pulses. A flip-flop has two outputs,

→ One for ~~(1)~~ Normal value and other for the complemented value of bits stored



S-R flip-floap.

⇒ Data representation format —

⇒ Number System –

→ A Number system of base or redix represented by $(N)_r$ is a system that uses distinct symbols for r digits. Numbers are represented by a string of digit symbols.

(1) Decimal Number system - It employes the redix 10 system. The symbols are from 0 to 9. To determine the quantity a no. represent, It is necessary to determine all digit by an integer power of redix and then form of weighted digits.

(2) Binary Number system - It uses redix 2 (0 & 1)

(3) Octal - It has redix 8 (0 - 7)

(4) Hexadecimal - It has redix 16 (0 to 9, A to F)

Physical significance of Hexadecimal and octal —

Suppose, a 16-bit register in a digital comp. Now, each register may be composed of 16 binary storage cells. With each cell capable of Holding a 0 and 1.

Thus, A 16-bit register can hold or store any binary no. from 0 to $2^{16}-1$

Binary coded Octal.	Binary coded Hexade
e.g $(10)_8 \Rightarrow 001\ 000$	e.g $(14)_{16} \Rightarrow 0001\ 0100$

Binary code —

It is a group of n bits that assume upto 2^n of distinct combinations 0s and 1s. with each combination representing 1 element of the set that is being coded.

$(\text{Number})^{\text{bits}}$ = No. of elements.

⇒ Alpha Numeric –

→ for handing of data that consists not only no. but also of letters of alphabet and certain special characters.

→ It consists of a set of elements. That includes 10 decimal 26 alphabet letters and different special characters.

⇒ Assignment –

(1) What are char and their ASCII values?

(2) What is the difference among binary, octal, deci, hexa, BCO, BCD, BCH.

⇒ complements – complements are used in Digital comp. to simply find subtraction operation and some logical Manipulation

→ There are 2 types of complements, for any radix (R)
(1) R's comp.
(2) (R-1)'s comp.

⇒ $(r-1)$'s comp. — Given a no N and radix r Then has n digits

$(r-1)$'s comp. $= (r^n - 1) - N$

e.g. $r = 10 \Rightarrow (10^n - 1) - N$

e.g. $n = 4 \Rightarrow (r-1)$'s comp.

$(r-1)$'s comp. $= (10^4 - 1) - N$

e.g. 9's comp. - 546700 , OP

(1) Difference between Von-nuemann and Harvard Architecture —

Von Nuemann	Harvard.
→ The data and program are stored in the same memory.	→ The data and program memories are Seperate
→ The code is executed serially and takes more clock cycles.	→ The code is executed in parallel.
→ The programs can be optimized in lesser size.	→ The program tend to grow big in size.
→ Used in conventional processors found in PCs and Servers.	→ Used in embedded systems and Mobile communication Systems.

(2) ASCII values of characters —

000	NUL	007	BEL
001	SOH	008	BS
002	STX	009	HT
003	ETX	010	LF
004	EOT	011	VT
005	ENQ	012	FF
006	ACK	013	CR

classmate
Date
Page

014	SO
015	SI
016	DLE
017	DC1
018	DC2
019	DC3
020	DC4
021	NAK
022	SYN
023	ETB
024	CAN
025	EM
026	SUB
027	ESC
028	FS
029	GS
030	RS
031	US
032	blank
033	!
034	"
035	#
036	$
037	%
038	&
039	'
040	(
041	)
042	*
043	+
044	,
045	-
046	.
047	/
048	0
049	1
050	2
051	3
052	4
053	5
054	6
055	7
056	8
057	9
058	:
059	;
060	<
061	=
062	>
063	?
064	@
065	A
066	B
067	C
068	D
069	E
070	F
071	G
072	H
073	I
074	J
075	K

classmate

Date
Page

076	L	101	e
077	M	102	f
078	N	103	g
079	O	104	h
080	P	105	i
081	Q	106	j
082	R	107	k
083	S	108	l
084	T	109	m
085	U	110	n
086	V	111	o
087	W	112	p
088	X	113	q
089	Y	114	r
090	Z	115	s
091	[	116	t
092	\	117	u
093	]	118	v
094	↑	119	w
095	-	120	x
096	←	121	y
097	a	122	z
098	b	123	{
099	c	124	|
100	d	125	}
		126	~
		127	DEL

⇒ 1's comp. —

→ r = 2 then (r−1)'s comp. = $(2^n - 1) - N$

e.g. $n = 4$

↓

$2^4 = (10000)_2$

$2^4 - 1 = (1111)_2$

⇒ **Note:-** (r−1)'s comp of octal and hexadecimal are obtain by subtracting each digit from 7 and F respectively

⇒ r's complement —

→ r's comp. of N = $r^n - N$, $N \neq 0$
and r's comp. of N = 0, $N = 0$.

⇒ 10's complement —

(i) 2 (ii) 3 (iii) 8 (iv) 9.

10
01 (1's comp.)
1
---
10 (2's comp.)

11
00
1
---
01 (2's co.

99
02

9
2

7

classmate
Date
Page

⇒ 10's comp —

(i) 2389

$$\begin{array}{r} 9999 \\ -\ 2389 \\ \hline 7610 \\ +\ 1 \\ \hline 7611 \end{array}$$ (10's comp.)

$= 2^n - N$

$= 10^4 - 2389$

$$\begin{array}{r} 10000 \\ -\ 2389 \\ \hline 07611 \end{array}$$

$= 7611$

⇒ 2's comp. − (101100)

$$\begin{array}{r} 010011 \\ +\ 1 \\ \hline 010100 \end{array}$$

010011 (1's comp.)

010100 (2's comp.)

$= 2^n - N$

$= 2^6 - 101100$

$= 1000000 - 101100$

$= 010100.$

→ for 10^n it will be 1 followed n 0's.

→ fon $10^n - N$ it will be 10's comp. of (N).

→ $2^n - N$ is 2's comp. of N.

⇒ 9's comp. (246700) = 353299.

10's comp. = 353399.

⇒ find the 1's comp. $(1101100)_2$

$$\cancel{\begin{array}{r} = 0010011 \\ +\ 1 \\ \hline 0010100 \end{array}}$$

= 0010011

classmate

Date

Page

⇒ Note: The comp. of the comp. of a number is the number itself.

⇒ R's comp. of R's comp. of ~~N~~

R's comp. of ~~N~~ = ~~(2^n)~~ N $R^n - N$

R's comp. of R's comp. $= R^n - (R^n - N)$
$= N$.

⇒ Binary Point — It is needed to represent fractions, integers and Mixed integer fractions.

→ In registers, the complication are by the fact that binary points are characterised by position in registers.

→ There are two ways of specifying the position of binary point in register
(1) fixed point representation -
(2) floating point "

(1) It assumes that binary point is always fixed point. Two Most widely used positions are (i) Binary point in the extreme ~~left~~ right at the register to store no. as integer

(ii) Binary point in the extreme left at the register to store no as ~~float~~ fractional
~~decimal~~ ~~fractional~~

(1) ~~fixed~~ Integer representation -

(A) signed numbers -

→ In ~~th~~ signed numbers,

(i) Positive numbers - Sign = 0
magnitude = Binary numbers

(ii) Negative numbers - Sign = 1
and rest is represented in
One of three possible ways.
(1) signed magnitude
(2) signed 1's comp.
(3) signed 2's comp.

→ +14 = 0000 1110.
-14 = $\xrightarrow{SM}$ $(10001110)_2$.
$\xrightarrow{S\ 1's\ c}$ 1111 0001
$\xrightarrow{S\ 2's}$ 1111 0010.

⇒ Signed Magnitude - It is used in Ordinary Architecture but not employed in computer Arithmetic

classmate

Date
Page

⇒ Difficulty with 1's comp. –

→ In case of 1's comp., there are 2 0's (+0, -0)

(1) Arithmetic Addition –

⇒ Subtraction – 2 n digits unsigned nos.
~~N ≠ 0~~ M − N, N ≠ 0
(i) Add Minuend M. to 2 comp. ~~(ii)~~ of Subtrahend = $M + (2^n - N)$

(ii) If $M \geq N$, Sum will produce, ~~and~~ end carry (2^n) which is discarded and left out M − N.

(iii) If $M < N$, Sum does not produce, end carry, and carry $2^n - (N-M)$.

→ exa. 72532 − 13250

10's comp. of subtrahend.

99999 13250 86749 1 86750	72532 86750 (1)59282 − 100000 0 59282

Ans. = $\boxed{59282}$

5 12 4 13
72532
13250
59282

classmate
Date
Page

→ exa : 13250 − 72532 (M < N)

$= r^n - (N - M)$

$= 10^5 - (72532 - 13250)$

$= 100000 - \cancel{0282}\ 59282$

=

→ exa. $X = (1010100)_2$

$Y = (1000011)_2$.

$X - Y$

$= 1010100 - 1000011$

2's comp. of $Y = 2^n - N$

$= 2^7 - 1000011$

$= 10000000 - 1000011$

$=$ 0 1 1 1 1 1 1 10

 10000000

$-$ 1000011

$\overline{00111101}$

⇒ Addition Algorithm –

→ signs of A & B are identical
Add two magnitude and attach sign of A to the result

(1) Signs of A and B are different
(A) compare the magnitudes and subtract the smaller no. from larger no. Choose the sign of result to be same as A if $A > B$ or complement of sign of A if $A < B$.

(B) If Magnitudes are equal Subtract ~~B~~ from A and make the sign of result positive.

→ flip-floaps are used to store "<u>Sign bits</u>"

⇒ Hardware components –

⇒ Parallel Adder – $A + B$ (Micro-operation)

⇒ comparator – $A >= B$, $A <= B$, $A = B$.

⇒ Parallel Subtractor – $(A - B)$, $(B - A)$

⇒ ~~Sign~~ flip-floaps (store sign bits).

classmate

Date ____
Page ____

Note: Subtraction is achieved by additions and complement.

⇒ E-Resister - It is used to check to determine the relative magnitudes of two numbers.

⇒ AVF (Add overflow flip-flop).

→ IF M = 0 (Mode control) then output = B and Input = 0 and Adder output = A + B

→ IF M = 1 then output will be 1's comp of B and input = 1 and Adder output = $A + \overline{B} + 1$

⇒ X = 1010100
Y = 1000011

$$
\begin{array}{rl}
X+Y = & 1010100 \\
 & +\underline{1000011} \\
 & (1)\,0010111
\end{array}
$$

X − Y = 1010100 + (2^n − Y)

= 1010100 + (2^7 − 1000011)

= 1010100 + (10000000 − 1000011)

= 1010100 + 0111101

= 1 0010001

⇒ 2's comp. Arithmatic -

(a) Addition: Add two nos including their sign bits and discard any carry out of the sign bit position

e.g.

+6	00000110
+13	00001101
+19	00010011

→ If no carry is generated, no is negative.

+6	00000110
−13	11110011
−7	11111001

2's comp

00000110
+ 1
00000111

13 = 00001101

11110010
+ 1
11110011

Sign bit represented by magnitude

(c)

−6	11111010
−13	11110011
−19	$\boxed{1}$ 11101101

Note: Negative numbers must initially be in 2's comp. and that if sum obtain after addition is negative, It is in 2's comp. form.

(c) Subtraction: Take the 2s comp. of subtrahend (including the sign bit) and add to the minuend (including the sign bit). A carry out of the sign bit position is discarded

Note: Binary nos in sign 2's comp. system are added and subtracted using same basic addition and subtraction rules as unsigned nos. Therefore computer need only one common hardware circuits to handle both types of arithmatic operation.

e.g.

$$\begin{array}{r} (-6) \\ -(-13) \\ \hline +7 \end{array} \qquad \begin{array}{r} \boxed{1} \\ 11111010 \\ +\ 00001101 \\ \hline 00000111 \end{array}$$

⇒ Overflow –

When two nos of n digits are added and sum occupies of $n+1$ digits is called as "Overflow."

→ An overflow can not occur after an addition if one no. is positive and other is negative.

→ An overflow may occur if two nos added are both positive or both negative.

carry out. carry in

e.g.

$$\begin{array}{r} +70 \\ +80 \\ \hline +150 \end{array} \qquad \begin{array}{r} 0\,\boxed{1} \quad \\ 01000110 \\ 01010000 \\ \hline 10010110 \end{array}$$

$$\begin{array}{r} -70 \\ -80 \\ \hline -150 \end{array} \qquad \begin{array}{r} 1\,0 \quad \\ 10111010 \\ 10110000 \\ \hline 01101010 \end{array}$$

⇒ Overflow Detection —

→ ~~Observing~~ Observing the carry into sign bit position and carry out of the sign bit position. If these two carries are not equal then their is a Overflow

$$C_{in} \oplus C_{out} = \begin{cases} 1, & \text{overflow} \\ 0, & \text{no Overflow.} \end{cases}$$

⇒ Algo for signed 2's comp

Subtract

↓

| Minuend in AC
| Subtrahend in Br

↓

| AC = AC + Br' + 1
| V ← overflow

↓

End.

Addition

↓

| Minuend in AC
| Sub. in Br

↓

| AC = AC + Br
| V ← overflow

↓

End.

⇒ Floating point representation -

(1) Mentissa - It is a signed fixed point number it may be a fraction or an integer.

(2) Exponent - It designate the position of decimal or binary point.

e.g. 1001.11 convert into fraction and exponent.

~~0.100111 × 2^4~~

sign bit

0 1001110 | 000100

Mentissa exponent.

→ convert in Fixed point represent.

$(.100111)_2 * 2^4$.

Note: A floating point no. is said to be normalised if MSB of mentissa is non-zero

e.g. $(00011010000)_2$ ← Normalize it.

→ zero ~~1101000~~ can't be normalized.

$(.1101000)_2 * 2^{-3}$.

⇒ Two main standard forms of floating point numbers are ANSI and IEEE

~~Eg~~ ~~143~~

Ansi floating point representation -

S MMMMMMMM I MMMMMMMM 8 M's 8 M

$\downarrow$ Sign bit. of no. — $\downarrow$ Expo. — $\downarrow$ Implied bit — $\downarrow$ ~~Expo~~ Mentissa

e.g. $(1101)_2 = (0.1101) * 2^4$

$\downarrow$ Ansi

00000100 11010000 00000000 000000

e.g. $(-17)_{10} = (\bar{1}0001)_2$

$= (-0.10001)_2 * 2^5$

$\downarrow$ ANSI

10000101 10001000 00000000 0000000

<u>Note:</u> Mentissa is represented in normalised sign magnitude fraction. The exponent is in biased form. If bias field has k bits then $BIAS = 2^{k-1}$.

S	E (k bits)	M

classmate

Date
Page

- The value of a no. is given by,

$$V = (-1)^S (.M)_B * B^{E-BIAS} \quad (B = \text{Base})$$

- Implicit normalized form increases the accuracy and value of no.

$$V = (-1)^S (1.M)_B * B^{E-BIAS}$$

⇒ IEEE - it provides the standard form floating point no. is stored in either in single precision (32 bits) ~~and~~ or double precision (64 bits)

→ The floating point no. can be denoted with (a) Implicit normalization
(b) Fractional form.

32 bit   Base: 2.

S(1)	E(8)	M(23)	Value
0/1	00...0	000...0	±0
0/1	111...1	00...0	±∞

→ If $E \neq 0$, $E \neq 255 \Rightarrow V = (-1)^S (1.M)_2 * 2^{E-127}$

By Default

→ If $E = 0$, $M \neq 0 \Rightarrow V = (-1)^S (.M)_2 * 2^{-126}$

By default

→ Other combination ⇒ NAN (Not a Number)

⇒ Floating point Arithmatic —

$$X = X_S\ (.XM)_2 * 2^{Ex - BIAS}$$
$$Y = Y_S\ (.YM)_2 * 2^{Ey - BIAS}$$

(i) $Z = X * Y = Z_S\ (.ZM)_2 * 2^{Ez - BIAS}$

where $Z_S = X_S \oplus Y_S$.
$Z_M = X_M * Y_M$.
$E_Z = E_X + E_Y$.

(ii) $Z = X/Y = Z_S\ (.ZM)_2 * 2^{Ez - BIAS}$

where, $Z_S = X_S \oplus Y_S$.
$Z_M = X_M / Y_M$.
$E_Z = E_X - E_Y$.

(iii) Addition and Subtraction

(1) Check for zeros.
(2) Align the Mentissa.
(3) Add or subtract the Mentissa.
(4) Normalize the results.

Multiplication. —

(1) Check for zeros.
(2) Add the exponents.
(3) Multiply the Mentissas.
(4) Normalize the products.

Division

(1) Check for zeros.
(2) Initialize registers and
(3) Align the dividend.
(4) Subtract the exponents.
(5) Divide the Mantissas.

⇒ Booth Algorithm – It gives a procedure for multiplying binary integers in signed 2's comp. representation.

→ If there is a string of 0's, there is no addition but only shifting operations. if there is a string of 1s, from weight 2^k 2^m can be treated as $2^{k+1} - 2^m$

$$\begin{array}{ccccc} 4 & 3 & 2 & 1 & 0 \\ 0 & 1 & 1 & 1 & 0 \end{array}$$

eg. $(+14)_{10} = 01110.$

$j = 3$ $k = 1$

k will be 1st "1" Occur in binary from LSB to MSB

$2^{J+1} - 2^{K}$

$= 2^4 - 2^1 = 14$

Rule: (1) Multiplicant is subtracted from partial product upon encountering the first least significant 1 in string of 1s in the Multiplyer

(2) The Multiplicant is added to the partial product upon encountering the first zero (provided there was a previous one) in a string of 0s in the Multiplier.

(3) The Partial product doesn't change when the Multiplier bit is identical to previous Multiplier bit.

e.g. $(-9) \times (-13)$ SC = No. of bits of BR. BR = Multiplicand, QR = Multiplier.

$BR = 10111$
$\overline{BR} + 1 = 01001$

Q_n	Q_{n+1}		AC	QR (Q_n)	Q_{n+1}	SC
		Initial	0	10011	0	101
1	0	Sub br / Add $\overline{BR}+1$	00000 + 01001 = 01001			
		ashr (shifting)	00100	11001	1	100
1	1	ashr	00010	01100	1	011
0	1	Add Br	00010 + 10111 = 11001			
		ashr	11100	10110	0	010
0	0	ashr	11110	01011	0	001
1	0	sub br	11110 + 01001 = $\boxed{1}$ 00111	01011		

$\begin{array}{r} 11101 \\ 10011 \end{array}$

Ch - 3.10.

ashr 00011 10101 1 000

So, MULT = AC BR
= 00011 10101

$\begin{array}{cccccccc} 7 & 6 & 5 & 4 & 3 & 2 & 1 & 0 \\ 0 & 1 & 1 & 1 & 1 & 1 & 1 & 0 \end{array}$

e.g. 0 1 1 1 1 1 1 0
k

conventional : $2^1 + 2^2 + 2^3 + 2^4 + 2^5 + 2^6$
6 Add operation, 21 shift

Booth : $+2^{J+1} - 2^K$
$= +2^7 - 2^1$

$= + A - B = A + (-B)$

1 Add, 1 sub, 8 shift.

classmate
Date
Page

⇒ Sign Extension -

→ Placing a smaller value in larger register is called sign extension.

→ The sign extension doesn't change the value and sign of the number

→ In 1s, 2s comp. notation the sign bit is copied in all the extended places.

→ In sign magnitude notation, the sign bit is moved to ~~most~~ MSB of extended register and remaining bits are filled with zero.

e.g. (-5) = 10101 (5-bit register)

1	0	1	0	1

1

sign

= 11101 (copied Answer)

$(3)_{10} \Rightarrow (10011)_2$ 10001 (-) → (101

Memory - Organization.

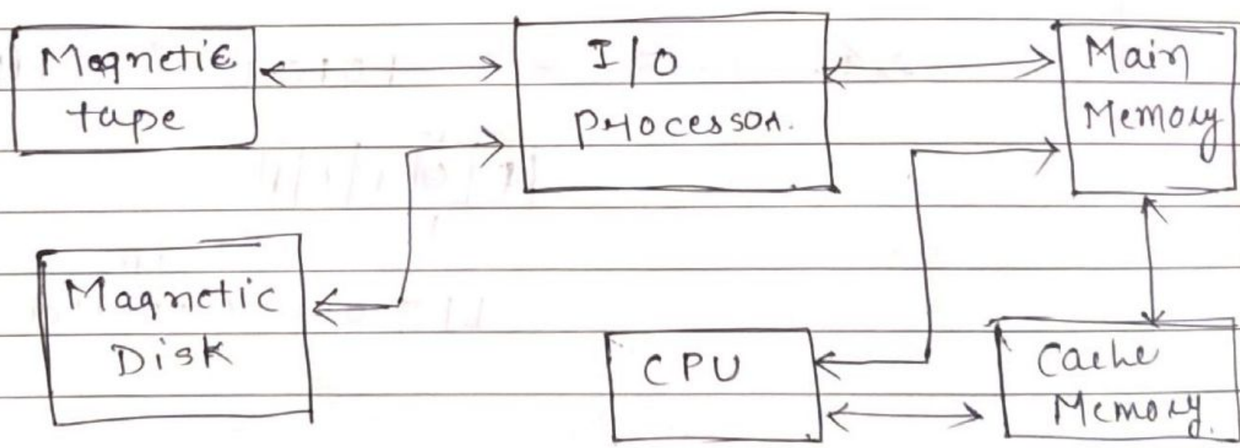
⇒ Memory hierarchy —

⇒ (1) Main Memory — The memory unit that communicates directly with CPU.

(2) Auxiliary Memory — Devices or Memory which provides backup storage are called auxiliary memory.

→ They are used for storing, System programmes, large data files and other backup information.

Auxiliary Memory (M/W)



→ Only Programs and data currently needed ~~by~~ by processor resides in main memory.

⇒ Cache Memory — It compansates for speed diff. between main memory and auxiliary memory. It makes current programs and data available to CPU at a rapid rate.

⇒ Goal of Memory hierarchy – To increase the performance rate of computer and obtain highest possible average access speed with minimization of the cost of entire system.

⇒ Explaination about diagram –

(1) The cache holds those parts of programs and data that heavily used while auxiliary memory holds those parts of programs and data not presently used by CPU.

(2) CPU has direct access to both cache and main memory but not to auxiliary memory.

(3) The transfer from auxiliary to main memory is usually done by means of direct memory access of large blocks of data

(4) The typical access time ratio b/w cache and main memory is about 1 to 7.

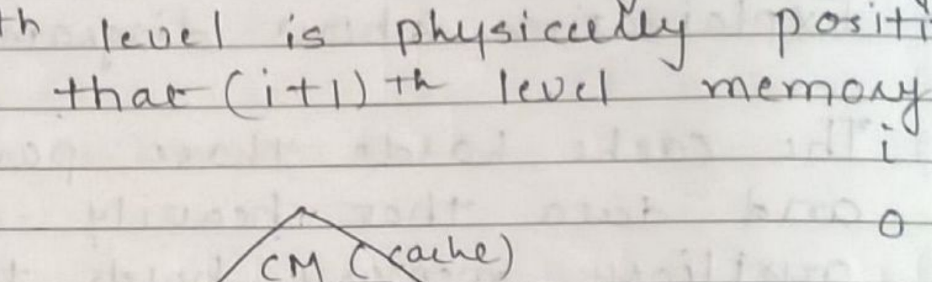
(5) Auxiliary memory access is usually thousands times that of main memory.

classmate
Date
Page

→ The Memory hirarchy is based on four parameters (1) T (2) S (c) C (D) F.

T - time
S - Size
C - cost
F - Frequency

→ In the structure of Memory hirarchy, the i^{th} level is physically positioned rather than $(i+1)^{th}$ level memory



$T_i < T_{i+1}$
$S_i < S_{i+1}$
$C_i > C_{i+1}$
$f_i > f_{i+1}$
$I_i \subset I_{i+1}$ (information)

→ The performance of Memory hirarchy is given by <u>hit - ratio</u>.

→ It is availability of refferred info at refferred level.
The hit ratio of last level is always one.

classmate
Date
Page

→ The hit ratio is inversely proportional to average access time and directly proportional to size.

→ Larger main memory and larger cache size is an ideal choice.

⇒ The side effects of the Memory hierarchy is data inconsistency (same information is available differently at diff. levels)

→ The proper updation technique resolves the problem of data inconsistency.

⇒ Mathematical expression of Memory Hierarchy.

Let T_1, S_1, C_1, H_1 are the specifications of level 1 memory in a two level system. Similarly the specification of Level 2 memory are T_2, S_2, C_2, H_2

→ Total size of Memory $S_T = S_1 + S_2$.

→ Average cost $C_{avg} = \dfrac{C_1 S_1 + C_2 S_2}{S_1 + S_2}$.

→ Average access time,

$T_{avg} = \cancel{\dfrac{H_1 T_1 + H_2 T_2}{H_1 + H_2}}$ $\quad H_1 T_1 + (1 - H_1) T_2$

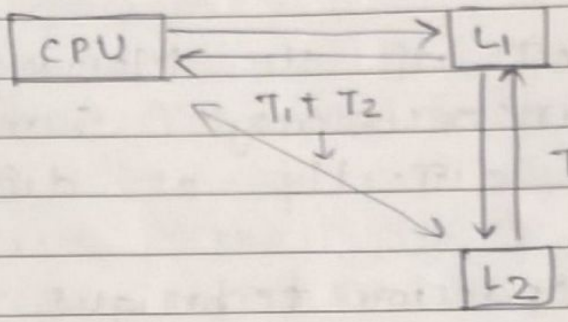
when $T_1 << T_2$.

$T_{avg} = H_2 T_2 + (1 - H_2) T_1$

when $T_1 >> T_2$.

→ If the process is present in L_1, then $H_1 = 1$. If the process is present in L_2 Then $H_2 = 1$ and $H_1 = 0$.

⇒ strict hierarchy –



$$T_{avg} \text{ for CPU} = H_1 \cdot T_1 + (1-H_1)(T_1 + T_2).$$

→ Inspite of moving a single word, He can move a clock of words from L_2 to L_1. So that next time when any thing asked, it will be present in L_1 itself where the time to move from L_2 to L_1 is given by $T_B = N * T_2$. (N = size of word-blocks).

So, Now the avg access time will be,

$$= H_1 T_1 + (1-H_1)(T_B + T_1).$$

$$= H_1 T_1 + (1-H_1)(\overset{T_B}{\cancel{N T_2}} + T_1)$$

$$= H_1 T_1 + (1-H_1) T_B + (1-H_1) T_1$$

$$= T_1 (H_1 + 1 - H_1) + (1-H_1) T_B$$

$$= T_1 + (1-H_1) T_B.$$

classmate
Date
Page

→ when the word is not available in L_1 then T_B is the extra time to move from L_2 to L_1 and N = size of blocks

→ For Multiple levels Average access time
$T_{avg} = T_1 + (1-H_1) TB_1 + (1-H_1)(1-H_2) TB_2 + \dots$

→ If the word is available in L_1, then T_1 is access time. Not in L_1, then $H_1 = 0$ and $H_2 = 1$.

(1) $T_{avg} = T_1$, $H_1 = 1$
(2) $T_{avg} = T_1 + TB_1$, $H_1 = 0$
(3) $T_{avg} = T_1 + TB_1 + TB_2$, $H_1 = H_2 = 0$.

Que-1 consider a two level memory system. The cache access time is 100ns and it is 12 times faster than Main memory what will be the minimum heat ratio than ensures average access time should not exceed 120ns.

for two level memory system,

$T_{avg} = H_1 T_1 + (1-H_1) T_2$

here, $T_1 = T_c$ and T_m. $T_m = 12 \times T_c$
$= 100$ ns $= 1200$ ns.

so, $T_{avg} = H_1 (100) + (1-H_1)(1200)$ — ①

~~120 H~~

$H \propto \frac{1}{T_{avg}}$

Average access time is max. ⇒ H will be min.

classmate
Date
Page

$\therefore (120)_{max} = H_{min}(100) + (1 - H_{min}) \times 1200$

$\therefore$ ~~Hmin =~~ $120 = 100H + 1200 - 1200H$

$\therefore 120 = 1200 - 1100H$

$\therefore 1100H = 1200 - 120$

$\therefore H = 98\%$

(b) If the heat ratio is 100% what will be ~~avg.~~ access time of Cache

$H \propto \frac{1}{T(avg)}$

→ Heat ratio does not effect the individual access time.

So, T_c won't change. It will remain Same = 100 ns.

Que-2 Consider a three level memory system with heat ratios 0.8, 0.9 and 1 Access times are 10ns, 20ns and 1000ns If the referred word is not available in level 1 or ~~two~~ 2 word block is ~~to~~ from level 2 to level 1 and handed over to the processor. If it is not present in level 2 then associated 8-word block is first transferred

classmate

Date
Page

from level-3 to level-2 and ~~consett~~ concerned block is transferred from level-2 to level 1. What will be the avg access time

$T_{avg} = T_1 + (1-H_1)T_{B1} + (1-H_1)(1-H_2)T_{B2}$

here $T_{B1} = n * T_2$ (From level 2 to Level 1)

$\therefore T_{B1} = 2T_2 = 40\,ns$.

and $T_{B2} = 8T_3$
$= 8 \times 1000$
$= 8000\,ns$

$T_1 = 10ns$, $H_1 = 0.8$, $H_2 = 0.9$.

$T_{avg} = 10 + (1-0.8)(40) + (1-0.8)(1-0.9)(8000)$

$= 18 + 160$

$\therefore T_{avg} = 178\,ns$

Que-3 consider a cache which is 5 times faster than memory with heat ratio 80%. The avg. access time of cache is increased by 20% from 50ns.
(a) What is the cache access time.
(b) what is the main memory access time.
(c) What is the new head ratio

$T_m = 5 \times T_c$
$H_1 = 0.8$.

classmate
Date
Page

99 9v

(a) $T_{avg} = H_1 T_c + (1 - H_1)\, T_m$

$\therefore\ 50 = 0.8\,(T_c) + (1 - 0.8)\,(5T_c)$

$\therefore\ 50 = 0.8\,T_c + T_c$

$\therefore\ T_c = 27.77\ ns$, (b) $T_m = 5 \times 27.77 = 138.85\ ns$

→ Avg. access time is increased by 20%

$(T_{avg})_{new} = \dfrac{50 \times 20}{100} = \dfrac{1000}{100} = 10\,ns + 50$

$\therefore\ (T_{avg})_{new} = 60\ ns$

$(H)_{new} = 0.64 \qquad \left(\because H \propto \dfrac{1}{T_{avg}}\right)$

$T_{avg} = H T_c + (1 - H)\, T_m$

$\therefore\ 60 = (0.64)\,T_c + (1 - 0.64)\,(5T_c)$

$\therefore\ 60 = 0.64\,T_c + 1.8\,T_c$

$\therefore\ \cancel{(H)_{new} =}$

$\therefore\ 60 = H_{new}(27.77) + (1 - H_{new})(138.85)$

$\therefore\ 60 = 27.77\,H_{new} + 138.85 - 138.85\,H_{new}$

$\therefore\ 111.08\,H_{new} = 78.85$

(c) $\boxed{\therefore\ H_{new} = 0.7098}$

classmate

Date

Page

Cache Memory –

→ It is the smallest and fastest memory component in hierarchy and maintain the locality of reference.

→ The cache and main memory is divided into fixed sized partitions (Blocks). The address mapping decides which block of the main memory is to be placed in which position.

→ There are three types of address mapping
 (1) Direct mapping.
 (2) Associative mapping.
 (3) Set-associative mapping.

→ Address mapping converts physical add. to cache address

→ Locality of reference is of two types.
 (1) ~~Special~~ Spatial (space/position)
 (2) Temporal (In time).

(1) Direct Mapping –

→ It is the simplest one and requires less no. of Tag bits.

(2) Associative mapping –

→ It is the fastest one but it requires more hardware.

classmate
Date
Page

(3) Set-associative mapping —

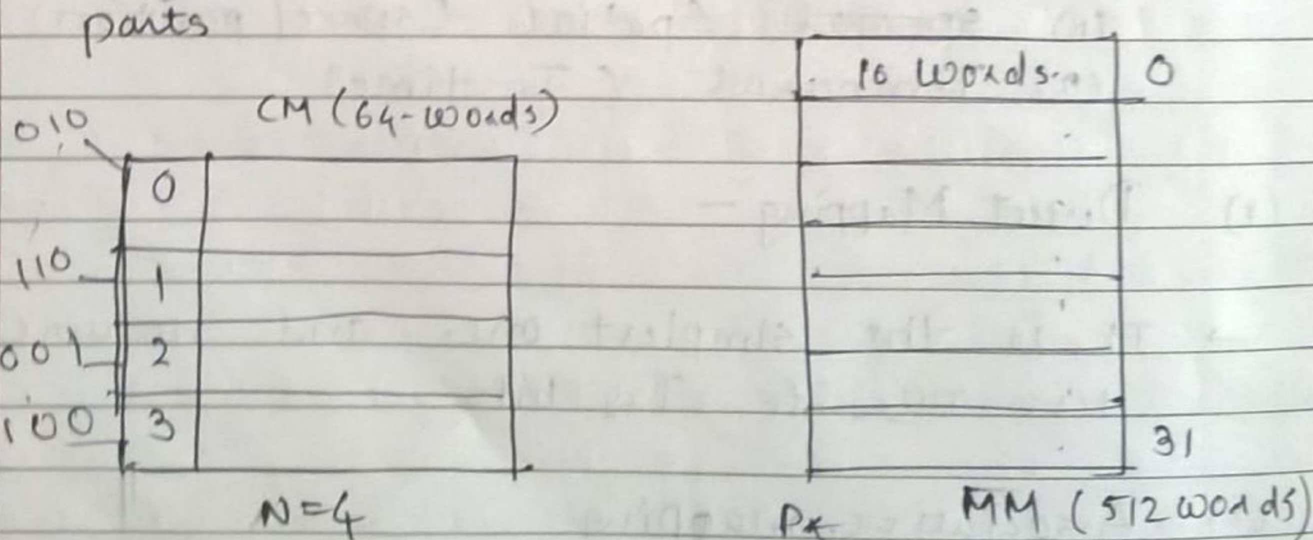
→ It gives the advantage of direct and associative mapping.

(1) Direct mapping –

→ k^{th} block of MM (main memory) has to be placed in $(k \bmod N)^{th}$ cache position. Many main memory block complete for same cache position. Only one among them is to be placed at a time in cache.

→ Tag bits denote which main memory block is currently placed in cache. Therefore, every cache block is attached with tag field.

→ Physical Address is divided into three parts



Tag	cache block offset	word offset

→ In the above exa. word offset = 4 bits
(Position of word within the block)
Cache offset = 2 bits. (since 4 blocks are in cache)
Tag = 3 bits.

Total no. of bits = 9 bits (since 512 words in main memory). $2^N = 2^9 = 512$

⇒ MM Blocks –

$$\left.\begin{array}{l} [0, 4, 8, 12, 16, 20, 24, 28] \rightarrow 0 \\ [1, 5, 9, 13, 17, 21, 25, 29] \rightarrow 1 \\ [2, 6, 10, 14, 18, 22, 26, 30] \rightarrow 2 \\ [3, 7, 11, 15, 19, 23, 27, 31] \rightarrow 3 \end{array}\right\} \text{cache}$$

→ Tag bit will tell that which block of MM is placed in which block of CM.
It is of 3 bits

Tag = {000, 001, 010, 011, 100, 101, 110, 111}.

le Check wether 200 words of MM is available ~~or~~ in cache or not.

$$[\underset{\uparrow \atop \text{Tag.}}{\underline{011}}\ \underset{\uparrow \atop \text{cache}}{\underline{00}}\underset{\uparrow \atop \text{word offset.}}{\underline{1000}}]$$

200 words of MM is not available in cache because Tag bits is $\underline{\underline{011}}$.

le check wether 121 words of MM is available in cache or not.

$$[\underset{\text{Tag}}{\underline{001}}\ \underset{\text{cache}}{\underline{11}}\ \underset{\text{word offset.}}{\underline{1001}}]$$

121 words of MM is not available because 001 Tag is available but 11 (=7) is not related to 001 2 is related to $\underline{001}$.

le 92 words.

[001 $\underline{\underline{01}}$ 1100]

"$\underline{No}$".

Example 96 words

[001 10 0000].

" Yes ".

⇒ Limitations of Direct-Mapping –

→ Suppose, a cache has four blocks with CPU ref. as given : 0 4 0 8 4 0 0 8.

→ For placing first block in cache it is placed in 0^{th} place of cache. Now, Again references of blocks are placed in 0^{th} block of cache by removing the previous once

(1) The No. of page replacements are more
(2) The time is slower
(3)

⇒ Associative Mapping –

→ It is a solution for limitations of direct mapping.

→ Any block of main memory can be placed any-where in a cache

→ Since, It involves less block movement it is the fastest mapping technique.

classmate
Date
Page

→ Associative mapping requires more hardware therefore costiest technique.

→ The no. of Tag comparators is equal to No. of blocks in the cache.

Tag	Word offset
5 bits	4 bits

Physical Add. (PA)

→ The no. of Tag bits are more in associative mapping.

$Tag_A = Tag_D +$ cache offset

Actual Size of Cache = Data M/W (Memory) + Tag M/W
↓
N * Tag

→ N * Tag = No. of blocks in cache no. of Tag bits.

R: 0 4 0 8 4 0 2 2

0	0
1	4
2	8
3	2

classmate
Date
Page

Limitations -

(1) It is a costliest technique.

Set associative Mapping -

In this mapping, k^{th} block of main memory to be placed in $k \pmod s$ set, where s = Total no. of sets in cache.
$s = \frac{N}{P}$, P = refferred as p-way set-association.

~~withing +~~ within the set, it can be placed anywhere in set.
The no. of Tag comparator is equal to the size of set P.

The p-way set association divides the Phy-add. into 3 components.

$\log_2 s$ bits.

Tag	Set offset	Word offset

PA

If P=1, it is direct mapping.
If P=N, associative mapping.

classmate
Date ____
Page ____

(Mb - Mega bits)
Byte

Consider 1 MB of cache memory and 1 GB of Main memory. Both are divided into 32 - word blocks. Each word contains 32 bits.

(1) What will be the no. of bits required to address main memory.

(2) What will be the no. of tag bits for following mapping technique Direct, associative, Set-associative. (8 - way)

(3) How many comparator and of what size are required for each case.

(4) What is the actual size of cache memory require for above mapping.

(2) Consider a cache memory which is applied with direct mapping. The no. of tag bits is equal to the ~~block of~~ no. of block in cache. Each blocks contain N words. N = total cache blocks. How many words are there in main memory.

~~6~~

classmate
Date
Page

Que The k^{th} block of main-memory has to be placed in which cache set if 2-way set association is used for the entire 2c cache blocks.

Ans-3 $s = \frac{N}{p} = \frac{2c}{2} = c.$

so, k^{th} block is to be place in cache memo

$= k \bmod c.$

Ans-2

N	$\log_2 N$	$\log_2 N$
↑	↑	↑
Tag	cache offset	word offset

Here Tag bits = The no. of blocks in cache.

$\therefore$ Tag bits = N bits

If 4 ~~blocks in~~ blocks in cache $\Rightarrow$ 2 bits
N blocks $\Rightarrow \log_2 N$

$PA = N + \log_2 N + \log_2 N$

$= N + \log_2 N^2$

$\therefore 2^K = 2^{N + \log_2 N^2}$

$10^9 = 2^{28}$ kitads

Ans. 1 $S_{CM} = 1\,MB$

$S_{MM} = 1\,GB$

Blocksize = 32 words

No. of bits ~~at~~ in each word = 32 bits.

→ No. of words = 1000 KB

$= \underline{1000 \times 1000}$ Bytes

$= 10^6$ bytes

$= \frac{10^6}{4}$

$= \frac{2^{20}}{2^2}$

$= 2^{18}$ words in Cache Memory.

= 256K words

→ No. of blocks $= \frac{2^{18}}{2^5}$

$= 2^{13}$ blocks

$= 2^{10} \cdot 2^3$

= 8K blocks.

⇒ For Main Memory,

No. of words $= \frac{10^9 \text{ bytes}}{2^2 \text{ bytes}}$

$= \frac{2^{30}}{2^2}$

$= 2^{28}$ ~~bytes~~ words

$= 2^{10} \cdot 2^{18}$ words

= ~~2^{25} K words~~ $= 2^{18}$ K words. = 256M words

No. of blocks $= \dfrac{2^{28}}{2^{5}}$

$= 2^{23}$ blocks

$= 2^{10} \cdot 2^{13}$

$= 2^{13}$ k blocks

$=$ 8M blocks

(a) No. of words $= 2^{28}$ words

$\log_2 2^{28} = 2^{28}$ bits

(b) (1) Direct Mapping:

Tag	Cache offset	Word offset
(remaining) 10 bits	13 bits	5 bits

$\longleftrightarrow$ 28 bits.

(2) Associative: Tag bits $= 10 + 13 = 23$ bits
Word off $= 5$ bits

(3) Set-associative mapping:

$\underline{P = 8}$

$s = N/p = \dfrac{2^{13}}{2^{3}} = 2^{10}$

set offset $= 10$ bits

(Remaining). 13 bits	10 bits	5 bits

Note: Higher association, Higher will be the tag bits.

(c) In Direct mapping:

1 Tag comparator having 10 bits (because Tag bits + [illegible]

In Associative mapping:

8K Tag comp. = ~~No. of~~ blocks in cache. ~~8K~~ has 23 bits

In SAM:

8 Tag comp. have 13 bits

(d) Actual size = Data M/w + Tag M/w

$$= 1MB + \left(\frac{\text{No. of blocks} \times 10 \text{ bits}}{8} \right) \text{bytes}$$

8 ← to convert bit in bytes

(i) · D = $1MB + \frac{2^{13} * 10}{8}$ bytes

(ii) A = $1MB + \frac{2^{13} * 23}{8}$ bytes

(iii) S = $1MB + \frac{2^{13} * 13}{8}$ bytes

⇒ Updation Techniques - They are used to deal with cache coherence problem

1) → The ~~Write through~~ updation simultaneously update the cache and Main
→ The Time for Updation = Max (T_c, T_M)

2) → It gives better performance for less no of updation. In write back updation the main memory updation is done only when consider updated block chosen for replacement
→ If the block chosen for replacement is not modified then Incoming block can be simply over-written in the case.
→ The dirty bit is used to indicate the status of the block in cache
→ The time for updation is given by,

$$T_{update} = \underbrace{T_B}_{\text{New block}} + \underbrace{T_c}_{\text{current Updation}}$$

This ~~block~~ formula is used to clean block

for dirty block, $T_{update} = \underbrace{2T_B}_{T_{B(old)} + T_{B(new)}} + T_c$

→ The current updation is always done in cache Memory.

Que Consider a hypothetical processor which issues 25% References for updation. It uses 2 level memory hierarchy with $T_c = 10ns$

$T_m = 100ns$
$T_c = 10ns$
$H_R = 0.8$ (Read heat)
$H_w = 0.9$ (write head)

If the ret word is not available than a four word block is to be moved from main memory to cache (either for read or write) what is the performance of this memory with write through strategy.

$$\text{performance} = \frac{1}{T_{avg}} \text{ words/sec}$$

→ $T_{avg} = 25\%\ T_{avg}(w) + 75\%\ T_{avg}(\text{Read})$ —— ①

$T_{avg}(w) = H_w \times T_{updation} + (1-H_w)(T_{updation} + T_B)$

$= 0.9 \times Max(T_c, T_m) + (0.1)(Max(T_c, T_m) +$

$T_{avg}(w) = 140ns$

$T_{avg}(R) = H_R \times T_c + (1-H_R)(T_c + T_B)$
$= 90ns$

$BR = -13 = 11101$
$= 10011$

$\overline{BR}+1 = 01101$, $QR = 0111$

$-13 * 7$

Put Multiplier in "2's compl. ~~Magnitude~~ form"

Q_n	Q_{n+1}	BR	AC	Q_R	Q_{n+1}	SC

putting values in eq. (1).

$T_{avg} = 25\%(140) + 75\%(90)$

$= 102.5$ ns.

Performance $= \frac{1}{T_{avg}} = 9.75 \times 10^{-3}$ words/ns

$= 9.75 \times 10^{6}$ words/sec.

Que Perform the same previous que. with write back policy. At any point of time 20% cache blocks are modified and cache is full

~~Tavg=~~ performance $= \frac{1}{T_{avg}}$ words/sec.

$T_{avg}(R) = H_R * T_C + (1-H_R)\left[\underbrace{(T_B+T_C)*80\%}_{clean} + 20\% * \underbrace{(T_{B(old)} + T_{B(new)} + T_C)}_{Dirty}\right]$

$= 0.8 \times 10 + (0.2)\left[\frac{410 \times 80}{100} + \frac{20}{100}(400+400+10)\right]$

$(4 \times T_m)$

$= 106$ ns./words

$T_{avg}(W) = H_W * T_C$ ~~updation~~ $+ (1-H_W)\left[(T_B+T_C)*80\% + 20\% * (T_{B(old)} + T_{B(new)} + T_C)\right]$

$= 58$ ns/words

$\therefore T_{avg} = 25\% \, T_{avg}(W) + 75\% \, T_{avg}(R) = 94$ ns/words.

$\therefore$ Performance $= 10.6 \times 10^{6}$ words/sec.

⇒ Block replacement techniques –

→ This techniques are aimed to choose such a cache block for replacement that may result in less or no penalty.

→ There are 3 techniques for block replacement.

(1) FIFO – The block that spent longer time in cache is choosen for replacement on the basis of arrival time. This strategy is ~~the~~ implemented using queue.

(2) LRU (least recently used) – The block that is not recently used is chosen for replacement. It can be implemented with a queue with slight adjacement on hit.

Note (1) If there is no hit, FIFO and LRU are same.

(2) Heat is the Recently ~~refreshed~~ reference.

(3) Direct Mapping – $(K \bmod n)^{th}$ block is chosen for replacement, where K = Recently referred Main memory block.

Q-1 No. of hits.
Q-2 Which blocks are present?
Q-3 " " " not present?

⇒ LRU is of two types.

LRU
- Associative (single Queue)
- Set Associative (Multiple Queues)

Que Consider a four block cache. The following main memory blocks are referred. Which replacement strategy use best performance? (No. of miss)

Referred blocks: 4, 5, 7, 12, 4, 5, 13, 4, 5, 7, 12, 13

(a) FIFO

4*	5*	7*	12*	4 (Hit)	5 (Hit)
4	4	4	4	4	4
	5	5	5	5	5
		7	7	7	7
			12	12	12

13*	4*	5*	7*	12*	13*
13	13	13	13	12	12
5	4	4	4	4	13
7	7	5	5	5	5
12	12	12	7	7	7

* - shows that first time it is coming

No. of miss = 10.
No. of Hit = 2.
Last replacement = ~~4~~.
Blocks present =

12
13
5
7

(2) LRU –

4*	5*	7*	12*	4	5	13*	4*	5	7*	12*	13*
4	4	4	4	4	4	4	4	4	4	4	13
	5	5	5	5	5	5	5	5	5	5	5
		7	7	7	7	13	13	13	13	12	12
			12	12	12	12	12	12	7	7	7

No. of Miss = 8
Hit = 4

(3) Direct Mapping – (k (mod 4))

	4*	5*	7*	12*	4*	5	13*	4	5*	7	12*	13*
0	4	4	4	12	4	4	4	4	4	4	12	12
1		5	5	5	5	5	13	13	5	5	5	13
2												
3			7	7	7	7	7	7	7	7	7	7

No. of miss = 9.
Hit = 3.
last replacement = 5.
Performance: FIFO < DM < LRU.

$S = \frac{16}{P} = \frac{16}{4} = 4$

classmate
Date
Page

Que Consider a 4 way set associative cache which was empty initially. Cache contain 16-blocks & main memory contains 256. Use LRU replacement policy with following References.

Ref: 0, 255, 1, 4, 3, 8, 133, 159, 216, 129, 63, 8, 48, 32, 73, 92, 155

formula $k(\bmod s)$, $s = \frac{N}{P} = \frac{16}{4\text{ way}} = 4$

$= k(\bmod 4)$

⇒ Memory hierachy - The Objective of Memory hierache is to prove the matching of the data transfer rate of faster process to that of lowest level memory of Resonable cost.

⇒ Memory Unit: It is an essential unit in a digital computer. since, it is needed for storing for program that are executed by CPU

⇒ Main Memory: The Memory Unit that communicates directly with CPU is called Main memory. e.x. Ram & ROM. It is also called "Primary Memory."

→ The Principle tech. used for main memory is based on semi conductor ICs.

→ The static RAM consists essentially of internal flip flops that stores binary info.

→ The stored information remains valid as long as power is apply to the Unit.

→ The dynamic RAM stores the binary info. in the form of electric charges that are apply to capacitor.

→ The stored charge tend to and the capacitors must be periodically recharged by refreshing the dynamic memory.

→ Refreshing is done by cycling through the words every few milliseconds to restore the decaying charge.

→ The dynamic RAM offers reduced power consumption and larger storage capacity in a single chip.

→ The static RAM is easier to use and has shorter read and write cycles

⇒ Cache Memory - A special very high speed m/w called cache is used to increase the speed of processing by making current programs and data available to CPU at Rapid Rate.

→ The cache Memory is employed in comp system to compensate for the speed diff b/w main memory access time and processor logic.

→ The speed of Main memory is very low in comparision with speed of Morden processors.

→ For good performance, the processor can't spent much of its time waiting to access instructions and data in Main memory. An efficient solution is to used a fast cache m/w, which essentially makes the main memory appears to the processor to be faster than it Really is.

→ The effect of the cache mechanism is based on a property of computer program called locality of program

→ Many instru. in localised area of Prog. are executed Repeatedly during some time period and the remainder of program is accessed relatively infrequ this is ref. as locality of ~~program~~ References.

→ It Manifests in two ways.
(1) Temporal (2) Special

(1) The temporal aspect of locality of Ref. suggests that whenever an Information (Instr./data) is first needed this item ~~This~~ should be broad into the cache where it will ~~be~~ hopefully remain until is needed again.

(2) The special aspect suggests that instead of fetching one item from MM to the cache. It is useful to fetch several items that reside at adjacent addresses as well. We will use the term block to ref. to a set of contiguous address locations of some size. Another term that is often used to ref. to a cache block is cache line.

→ The correspondence b/w the main memory blocks and those in cache is specified by mapping function. When the cache is full and memory word that is not in cache is ref. the cache control hardware must decide which block should be removed to create space for the new block that contain the ref. words. The collection of rule for making this decision constitutes the Replacement algorithm.

→ The Processor doesn't need to know ex about the existance of It simply issues Read and write

locations in the memory. In a read operation MM is not involved for a write operation the system proceed in two ways

(1) Right-trough Mechanism -

→ The cache location and MM location are updated simultaneous.

→ The second tech. is to update only the cache location and is to mark it as updated with an associative flag bit often called dirty or Modified bit

→ The MM location of the word is updated later when the block containing this marked word is to be removed from the cache to make room for a new block. This technique is known as write back / copy back

→ When the addressed word in a Read operation is not in a cache a Read miss occurs

→ The block of words that contains the requested word is copied from the main memory into cache. After entire block is loaded into the cache

classmate
Date
Page

the perticullar word requested is forwarded to the processor.

→ Alternatiocly The word is may be sent to the processor as soon as it is send from the MM. The later approach called load through/ early restart reduces the processors waiting period

⇒ Cache coherence Problem –
(1)→ will have multiple copies of same data in diff caches.
(2)→ The data in caches modified localy.
(3)→ Cache become inconsistant b/w caches and b/w cache and MM. from which cache data was copied
→ The transformation of data from MM to CM is ref. as a mapping process.

⇒ Address Mapping –

→ when a tag add. is presented to the cache it must be quickly compare to the stored tags to determine wether the matching tag is currently assign to the cache.

→ The fastest tech. for implementing tag comparisons in associative or content addressing which permits the i/p tag to be compared simultaneously to all tags in the cache tag RM.

(A) Associative Mapping: More flexible mapping Method in which a main Memory block can be replaced into any cache block position. The tag bits of an address received from the processor are compared to the tag bits of each block of the cache to see if the desired block is present or not.

→ An associative cache employes a tag that is a block add. as the key. At the start of Memory access the incoming tag is compared simultaneously to all the tag stored in the cache tag memory.

→ If a match occurs, a match indicating signal trigger the cache to service the requested M. access

→ A no match signal identifier a catch miss and the memory access requested to be forwarded to Main Memory for service. A cache block containing the target add is then sent ~~to~~ from M to C. and at the same time a data word is sent to the CPU or transfered from CPU to Cache In response to the original access request.

(b) Direct Mapping -

→ An Alternative and simpler address mapping tech for cache

→ The lower order s bit of each block identifies the Unique cache set that can store a block. the remaining high order bit constitute a tag and only these bits need to be stored in the caches ~~tag~~ memory. Then the lower order d bits form the displacement address of the word with its block.

Note: This is easy to implement but it is not very flexible.

→ The main drawback of direct mapping is that the caches hit ratio drop sharply if two or more
on to the same

(c) Set associative mapping The combinations of direct and associative tech.

→ The blocks of the cache are group into sets and the mapping allows a block of main memory to reside in any block of a specified set at the same time the hardware cost is reduced by decreasing the size of associative search the tag field of the add. must then be associatively compared to the tags of the two blocks of the set to check if the desired block is present or not. This 2-way associative search is simple to implement.

→ The no. of blocks per set is a parameter that can be selected to suit the requi of a particular comp.

⇒ Virtual Memory – It is a concept used in large computer systems that permits the users to construct his programs as though he had a large memory space to the totality of auxiliary memory. A virtual M. is used to give the programmer the illution that he has very large memory at his disposal even though the computer actually has a relatively small main memory. An address used by the programmer is called a "virtual add" and set of such add is called "add space".

→ M. Mapping table is used to transfer virtual add. to physical address.

→ The virtual memory is devided into pages and the main memory is devided into blocks. Generally the size of page must be equal to the size of block. Random access memory page table tech. uses the no. of Memory location require to store memory mapping table equal to the no. of pages available in V.M.

→ In associative M. page table tech. the no of M. locations required to store memory mapping table equal to the blocks available in M.M.

→ The wastage of Memory is minimum in the case of associative mem. page table tech.

→ The most commonly used page replacement algorithm in the VM are
(1) FIFO
(2) LRU.

classmate
Date
Page

I/o data transfer

→ ~~Interphasing~~

⇒ Interfacing — writing s/w inst. and designing hardwar ckt to enable the CPU to communicate with peripheral devices is called Interfacing and the hardware ckt is called Interfacing ckt.

⇒ Need for Interfacing – Most of the peripherals are electromechanical dvc. and their manner of op. is diff from the op of CPU and Memory which are electronic devices.

→ The data transfer rate of peripheral is ~~weak~~ must slower than the transfer rate in central comp.

→ The op. of ~~pla~~peripherals must be synchronised with op. of CPU and M. Unit.

→ Data formats in peripheral differ from the word format in central processor

→ The op of each peripheral must be controlled so as not to disturb the op. ~~of~~ of the controlled comp. and other peripherals conn to the system

⇒ I/O Processor: It is sometimes called
" A Data-channel controller"

→ There are 2-ways to connect p.p. devices
to CPU (1) Memory Mapped i/o
(2) Isolated i/o or i/o mapped i/o

(1) Programmed I/O — CPU has direct control
over i/o for
- Sending status
- Read-write commands
- transferring data

→ CPU waits for i/o module to complete operation
→ limitation: wastage of time

⇒ steps - how program i/o work —

(1) CPU request i/o op.
(2) i/o module performs op
(3) i/o module sets status bits
(4) CPU checks status bits periodically
(5) i/o module doesn't interrupt CPU
(6) CPU may wait or come back later

→ ~~CPU~~

⇒ I/O mapping —

(1) M.M. i/O — devices and memory share
an address space
→ i/o looks just like memory read/write
op.

- No special commands are given for i/o.
- Large selection of Memory access commands are available.

(2) Isolated i/o -

- Seperate add space for device and M.
- Needs i/o or memory select lines
- There are special commands for i/o.
- Limited set of Memory access commands are available.

⇒ Interrupt driven i/o -

- It overcomes CPU waiting.
- No repeated CPU checking of device
- I/O module interrupts when ready.

⇒ Basic Op. of intrupt driven i/o.

(1) CPU read command
(2) i/o module gets data from peripheral while CPU does other work.
(3) i/o module interrupts CPU
(4) CPU requests data
(5) i/o module transfers data

- In Memory mapped i/o technique the i/o .
under that assumption they will be given address. Same control lines are used to activate memory location or

i/o devices.

→ In i/o or isolated i/o tech, the i/o devices are given Seperate add. space and control signals to activate M. locations and i/o devices.

→ Data transfer b/w CPU and peripheral is handled in one of three possible modes
(1) Data transfer under program controlled
(2) Interrupt initiated data transfer
(3) Direct Memory access transfer.

→ Program controlled op. are the result of i/o instructions written in comp. programmes
→ each data item transfer is initiated by an instruction in the program.

⇒ Disadvantage of program controlled data transfer:
(1) The processor stays in a program mood until the i/o unit indicates that it is ready.

→ This is a time consuming Process since it keeps the processor busy need lessly.

→ In interrupt initiated data transfer, when the P.P is ready for data transfer it generates an interrupt request to the processor. when the processor stops momentarily the task it is doing, branches to a Service routine to Process

12.1, 12.2, 12.3, 12.5, 12.6
11.2, 11.3, 11.4, 11.6,

the data transfer and then returns to the task it was performing.

→ In ~~dire~~ direct Memory access the interface transfers data into and out of Memory unit through the memory bus generally to transfer bulk amount of data from memory to peripheral or from peripheral to CPU. There are two formats of data transfer
- Parallel
- Serial

→ In parallel mode data bits (usually a byte) are transferred parallely over the communication line (known as buses). Thus all the bits of a byte are transferred simultaneously within the time frame allotted for the transmission.

→ In serial data transfer each data bit is sent sequencialy over a single data line. In order to implement serial data transfer, the sender and receiver must devide the time frame alloted for the transmission of a byte into some intervals during which each bit is send and receive

→ In serial transmission, the information is transferred in the form of frames. The frame consist of three parts
- Start bit
- character code
- stop bit

Numericals - Write back, write through

⇒ RAID

Ch.- 3, 10, 12, 11, basics of RAM, ROM

~~Associative Memory, I/o processor.~~

→ start bit is always at logic 0
stop bit " " 1

→ The data transfer in both the parallel and serial mode of op. can be either synchronous or Asyn.

→ In synchronous mode, both source and destination unit works in synchronous with same control signal.

→ In A synchronous mode, the source unit and destination unit have their own independent control signals.

→ Asynchronous parallel data transfer can be of 3 types
- strobe control
- two wire ~~and~~ hand shaking method
- three wire hand shaking method

→ Strobe control is of two types (1) Source initiated (2) Destination initiated.

→ Similarly two wire handshaking method is also of two types (1) Source " (2) Destination "

→ Exchange of control signals b/w source and destination during the data transfer called as hand shaking.

→ The dis Ad. of hand shaking is that it is not possible to conn. ~~two~~ more than one destination

Unit to a single source Unit. transmission is used to interface serial i/o device to a Used structure

→ In DMA transfer the CPU initializes the DMA by sending a M. Address and no. of words to be transferred.

→ DMA is preferred data transfer for high speed peripheral devices such as Magnetic disks.

→ Three modes of DMA Operation.
  - continuous DMA/ burst mode
  - cycle stealing
  - inter~~ne~~leaved

→ In burst mode a sequence of arbitrary level data is transferred in a single continuous burst during which the DMA controller is the master of the system.

→ In cycle stealing tech, the transfer is done by first checking if the Memory Unit is not Used by CPU. and then the DMA steals one memory cycle to access a word in Memory.

→ In interleaved DMA, the processor and DMA access of Memory can be interleaved on alternate half cycle of the clock.
The M/w during first half is access by DMA

classmate
Date ______
Page ______

while CPU accesses during the 2^{nd} half.

→ Interrupts - It is an exceptional event which causes the CPU to temporarily suspend the current program being executed. The control is subsequently transferred to some other program referred to as interrupt service routine which specifies the actions to be taken if the exceptional event occurs.

while CPU accesses during the 2^{nd} half.

⇒ Interrupts - It is an exceptional event which causes the CPU to temporarily suspend the current program being executed. The control is subsequently transferred to some other program referred to as interrupt service routine which specifies the actions to be taken if the exceptional event occurs.