10/1/19

Date
Page

Number Representation

→ Signed Numbers.

- The MSB of a binary number is used to represent the sign bit. If the sign bit is equal to zero, the signed binary number is positive otherwise it is negative.

- Three Ways to Represent Signed Numbers –

(i) Signed Magnitude
(ii) One's Complement
(iii) Two's Complement

→ Signed Mangitude

- In this method, 1 denotes a negative number & 0 denotes a positive number.

+0	0	000	−0	1	000
+1	0	001	−1	1	001
+2	0	010	−2	1	010
+3	0	011	−3	1	011
+4	0	100	−4	1	100
+5	0	101	−5	1	101
+6	0	110	−6	1	110
+7	0	111	−7	1	111

- Drawbacks –

(i) Has the problem of double representing zeroes (+0 & -0)

(ii) Complications in arthemetic operations.

- Examples –

$$\begin{array}{r} 1010 \rightarrow 10 \\ \underline{0110} \rightarrow 6 \\ \underline{1\,0000} \rightarrow 16 \end{array} \qquad \begin{array}{r} 1010 \rightarrow -2 \\ \underline{0110} \rightarrow +6 \\ \underline{10000} \end{array}$$

→ <u>One's Complement.</u>

- It behaves like the negative of the original number in some arthematic operations or inverse of negative number into positive.

+0	0	000	-7	1	000
+1	0	001	-6	1	001
+2	0	010	-5	1	010
+3	0	011	-4	1	011
+4	0	100	-3	1	100
+5	0	101	-2	1	101
+6	0	110	-1	1	110
+7	0	111	-0	1	111

- <u>Drawbacks</u> :–

(i) Has the problem of double representing zeroes (+0 & -0)

(ii) Overflow can occur after addition of 1.

- **Examples:-**

0001) is complement
1110
  +1
1111

0011 → +3
1011 → -4

0101 → +5
1100 → -3
(1)0001
  +1
0010 → 2

1011
0111
(1)0010
  +1
0011 → +3

→ Two's Complement

- In this method, 1 is added to the 1's Complement of binary number.

+0	0 000		-8	1 000
+1	0 001		-7	1 001
+2	0 010		-6	1 010
+3	0 011		-5	1 011
+4	0 100		-4	1 100
+5	0 101		-3	1 101
+6	0 110		-2	1 110
+7	0 111		-1	1 111

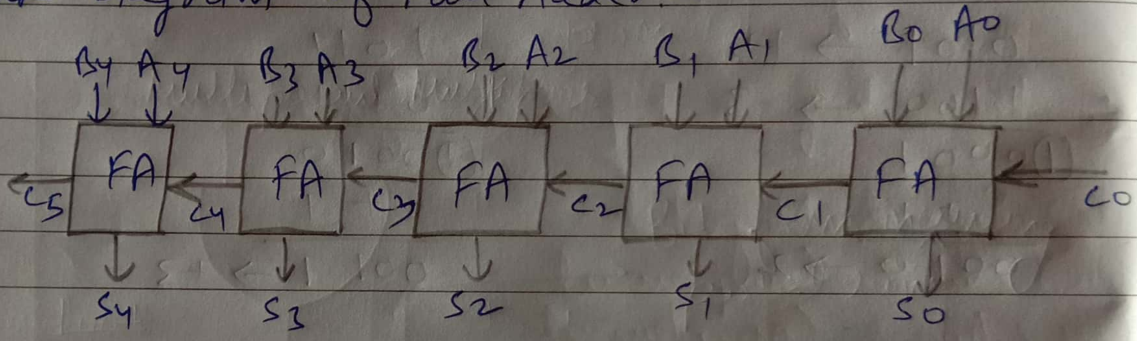
- There is no drawback in this method.

• Examples:-

0001 } is complement
1110
$+1$
1111

0011 → +3
1011 → -5
1110 → -2

• Block Diagram of Full Adder:-



(5 bits)

$$\text{Sum} = 2n$$
$$= 2 \cdot 5 = 10$$

Delay of 10 clk cycles in Full ~~Add~~ Adder of 5 bits.

$$\text{Carry} = 2(n-1)+1$$
$$= 2n-1$$

→ Adder

• Difference between half adder & full adder-

Half Adder	Full Adder
(i) No. of inputs: 2 No. of outputs: 2	(i) No. of inputs: 3 No. of outputs: 2
(ii) Previous Carry is not used	(ii) Previous Carry is used
(iii) Exp. - $S = A \oplus B$ $C = A \cdot B$	(iii) Exp. - $S = A \oplus B \oplus C$ $C = AB + BC + AC$

Scanned by CamScanner

→ Carry Look Ahead Adder → (CLA)

(i) Generate function → $x_i \cdot y_i$
(ii) Propagate function → $x_i + y_i$

$$C_{i+1} = G_i + P_i C_i$$
$$C_3 = G_2 + P_2 C_2$$
$$C_4 = G_3 + P_3 C_3$$
$$= G_3 + P_3 (G_2 + P_2 G_1 + P_2 P_1 G_0 + P_2 P_1 P_0 C_0)$$

$$C_{i+1} = G_i + P_i C_i$$
$$C_{i+1} = x_i y_i + (x_i \oplus y_i) C_i$$

$$C_{i+1} = x_i y_i + (x_i \overline{y_i} + \overline{x_i} y_i) C_i$$
$$= x_i y_i + (x_i \overline{y_i} c_i + \overline{x_i} y_i c_i)$$
$$= x_i y_i + (y_i + \overline{y_i} c_i) + \overline{x_i} y_i c_i$$
$$= x_i (y_i + c_i) + \overline{x_i} y_i c_i$$
$$= x_i y_i + x_i c_i + \overline{x_i} y_i c_i$$
$$= y_i (x_i + \overline{x_i} c_i) + x_i c_i$$
$$= y_i (x_i + c_i) + x_i c_i$$
$$= y_i x_i + y_i c_i + x_i c_i$$

• Drawback

(i) We can use only 4 bit adder.

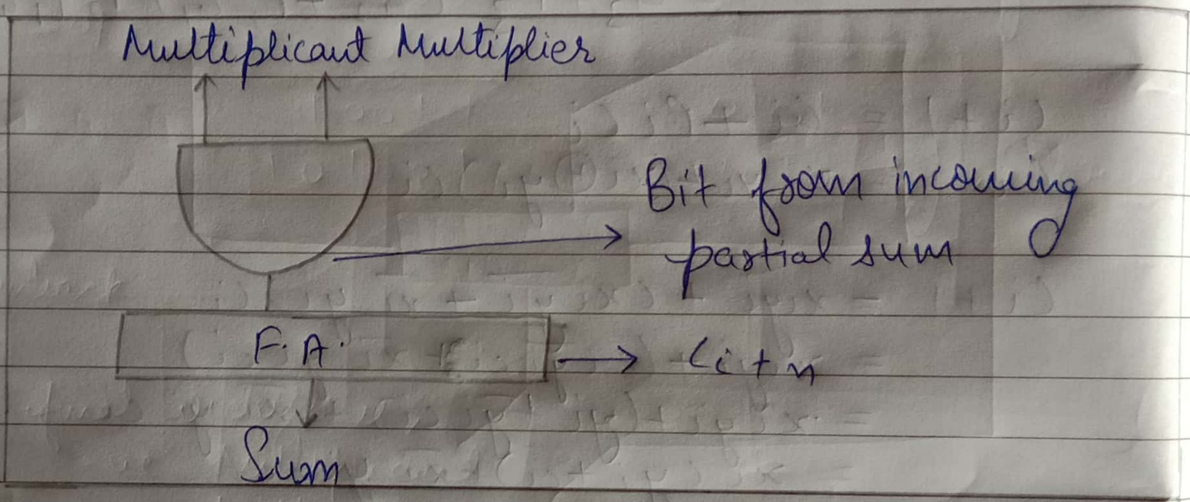
Scanned by CamScanner

17/1/19

Multipliers

(i) Array Multiplier
(ii) Sequential Multiplier
(iii) Booth's Algorithm for Multiplication

I Array Multiplier.



• Block Diagram

• Examples :-

(i)

```
  11 → 3
  11 → 3
1001 → 9
```

(ii)

```
   111 → 7
   111 → 7
110001 → 49
```

(iii)

```
   1001 → 9
   0101 → 5
   1001
  0000
 1001
0000
0101101 → 45
```

0000

0101101

- Drawbacks

(i) Complexity increases.

(ii) Cost increases

(iii) Cannot increase effiency of a system.

II Sequential Multiplier.

- It is also called add & shift multiplier.

$\boxed{1011}$ → Multiplicant

Carry ← $\boxed{0}$ $\boxed{0000}$ $\boxed{1110}$ → Multiplier

Carry ← $\boxed{0}$	$\boxed{0000}$	$\boxed{1110}$ → Multiplier
0	0000	1110
0	0000	0111 → Shift.
0	1011	0111
	1011 →	Add
0	0101	1011
	1011	
	10000	1011 → Add
0	1000	0101 → Shift
	10111	
	10011	0101
	101001	1010 → final result

- Drawback

(i) If final result is,
-ve Multiplicand
+ve Multiplier
OR
-ve Multiplicand
+ve Multiplier

III Booth's Algorithm for Multiplication

- Conversion Chart and example:

(i) 0011010 1 $\boxed{0}$ → Imaginary 0
0 +1 0 -1 +1 -1 +1 -1

(ii) 1100100 $\boxed{0}$ → Imaginary 0
0 -1 0 +1 -1 0 0

00	→ 0
01	→ -1
10	→ +1
11	→ 0

classmate

Date ______

Page ______

· Multiply (Examples)

(i) 6-Bit

```
        010011        → Multiplicand
        001100 [0]    → Multiplier.
        0-10-100 0
0000000000000
000000000000X
ϕϕϕϕϕ0ϕϕ0ϕXX
        011XXX
```

(ii) 4-Bit

```
~~1001~~
~~1101 [0]~~
~~0-1+1-1~~
~~Dϕϕ1ϕ~~
```

```
    1001
    1101 [0]
   0-1+1-1
 00000111
 111001X
 000111XX
 0000XXX
(1)0000101
```

Note:-

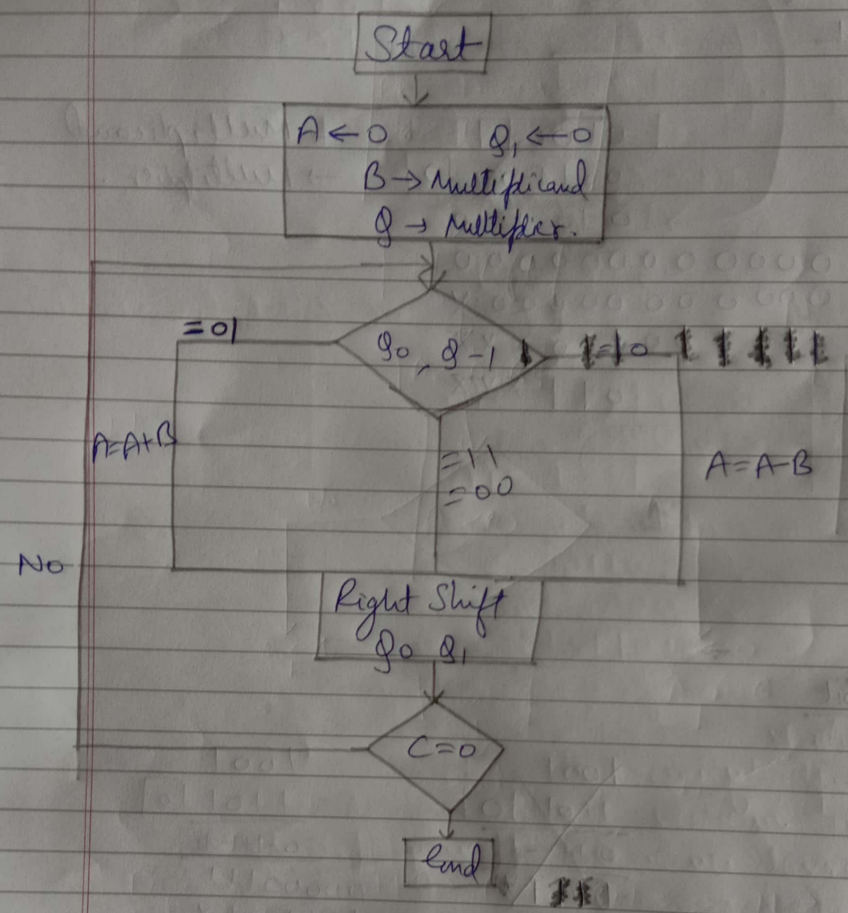
If MSB=0

No. of positive.

If MSB=1

No. is negative

• Flow Chart :-



• Operation Right Shift :

Multipli Mul.

-5×4

1011×0100

$B = 0101$ 2's comp.

Count = 4
= 3
= 2
= 1
= 0

	A	Q_1	Q_{-1}
Initial	0000	0100	0
Step 1	0000	0010	0
Step 2	0000	0001	0
Step 3	0101		
	0101 } A = A − B	0001	0
	0010 } A = A + B	1000	1
	1011		
	1101	1000	1
	1 1110	1100	0

★ Division

$12 \overline{)169}$

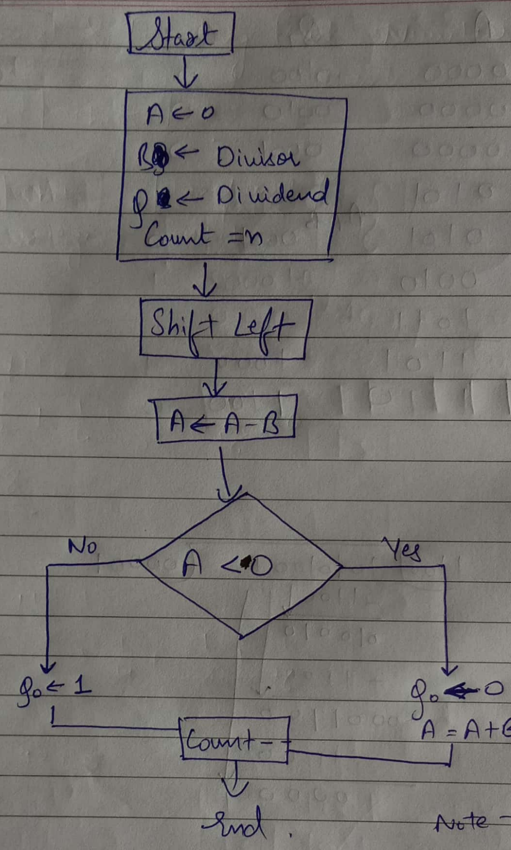
```
1100 ) 10101001 ( 00001
      -1100
       010010
      - 1100
       0001100
          1100
          0000 1
```

$13 \overline{)178}$

```
1101 ) 10110010 ( 00001101
       -1101
        010010
          1101
        0001011
        - 1101
        0001010
         -1101
          01001
```

Restoring Division Algorithm.

★ Flow Diagram of Shift Subtract Division.



~~Flow~~

After A = A − B.

A = 0.

Then $Q_0 \to 1$

End.

A = 1

$Q_0 \leftarrow 0$

A = A + B

End.

Note − If equal to zero then 1

equal to 1 zero

Divisor − 0011 → B ⇒ B+1 = 00011 → 11101

Dividend − 1010 → Q

acc. to the − dividing

Operation		A	Q
Initialize		00000	1010
LS	A	00001	010$\boxed{\ }$
A=A−B	B	11101	0100
		$\boxed{(1)1110}$	
A = A + B		00011	
		(1)00001	0100

Count → 4

Count → 3

Operation	A	Q
	1 00001	0100.
LS	00010	100 $\boxed{\ }$
A=A-B	11101	
	1 11111	1000
A=A+B	00011	
	1 00010	1000
LS	00101	000 $\boxed{\ }$
A=A-B	11101	
	1 00010	0001
LS	00100	001 $\boxed{\ }$
	11101	
	1 00001	0011

B → Divisor – 0100 → B ⇒ B + 1 ⇒ 00100 ⇒ 11100

Q → Dividend – 1111

Operation	A	Q	
Initialize	00000	1111	Count = 4
LS	00001	111 $\boxed{\ }$	Count = 3
A=A-B	11100		
	11101		
A=A+B	00100		
	1 00001	1111	
	1 00001	1111	
	00011	111	
A=A-B	11100		
	11111		
A=A+B			

Scanned by CamScanner

classmate

Date ____
Page ____

⇒ Fixed point & Floating Point Number Representation

| 4 bits | 4 bits |

| 0/1 | 3 bits | . | Fractional parts |

Sign bit Integer 4 bits.

→ 0110 . 1110

$6. \quad \frac{1}{2} + \frac{1}{4} + \frac{1}{8} + \frac{1}{16} \times 0$

6.8

→ 0111 . 1111

$7. \quad \frac{1}{2} + \frac{1}{4} + \frac{1}{8} + \frac{1}{16}$

7.9375

If 32 bit

| 1 bit | 15 bits | . | 16 bits. |

Floating Point Formula $-$ $(-1)^s \times (1.M) \times 2^{e-bias}$

Bias $- 2^{n-1}-1$

$(-1)^0 = 1$ $(-1)^1 = -1 \times$ something whole eq. ~~rotate~~ will become -ve.

$Bias = 2^{n-1}-1$

★ Floating point

0	010	0001
Sign bit	exponent bit	Mantissa

→ Convention by IEEE 754

Formula $- (-1)^s \times (1.M) \times 2^{e-bias}$

$(-1)^0 \times (1.0001) \times 2^{2-bias \,(2-3)}$

$\boxed{(1.0001) \times 2^{-1}}$ → Positive Number.

Bias $= 2^{3-1}-1$

$= 2^2-1$

$= 4-1$

$= 3$

$= 1 + \frac{1}{16} + \frac{1}{2}$

$=$

1	110	1101
Sign bit	exponent bit	mantissa.

$\Rightarrow (-1)^s \times (1.M) \times 2^{e-bias}$

$\Rightarrow (-1)^1 \times (1.1101) \times 2^{6-bias}$

$\Rightarrow -1 \times (1.1101) \times 2^{6-3}$

$\Rightarrow$

$\Rightarrow -\left(1 + \frac{1}{2} + \frac{1}{4} + \frac{1}{16}\right) \times 8$

$\Rightarrow -\left(1 +\right.$

Bias $= 2^{3-1}-1$

$= 2^2-1$

$= 4-1$

$= 3$

1 bit	8 bits.	23 bits
Sign bit	Exponent	Mantissa

$$Bias = 2^{8-1} - 1$$
$$= 2^7 - 1$$
$$= 127.$$

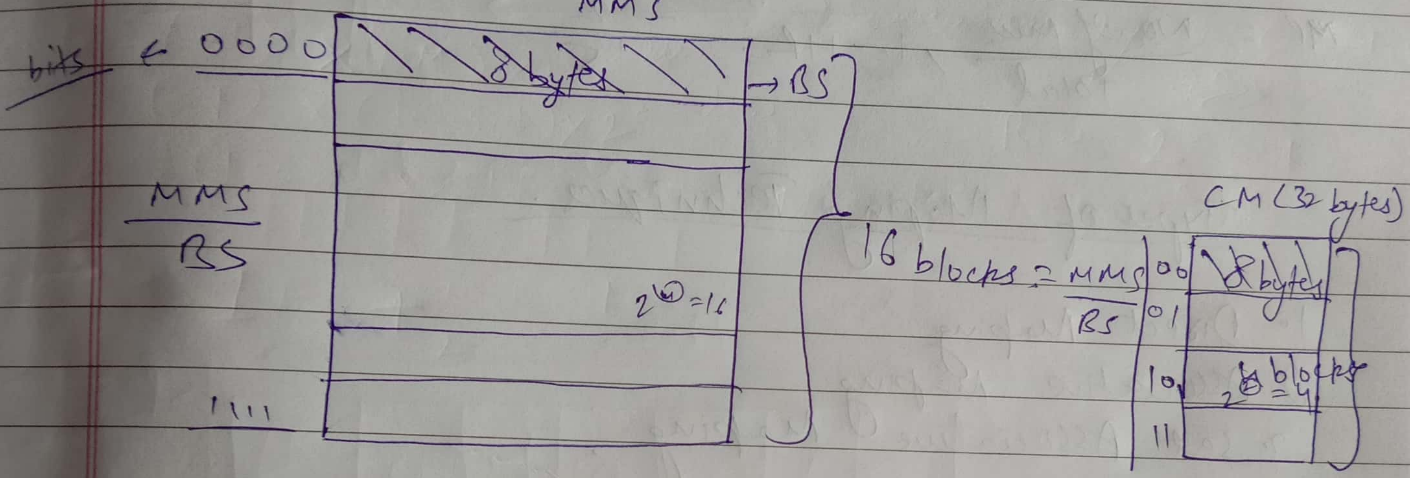
1 bit	11 bits	52 bits

28/1/19

Unit 5

Cache Memory

Main Memory (128 bytes)
MMS

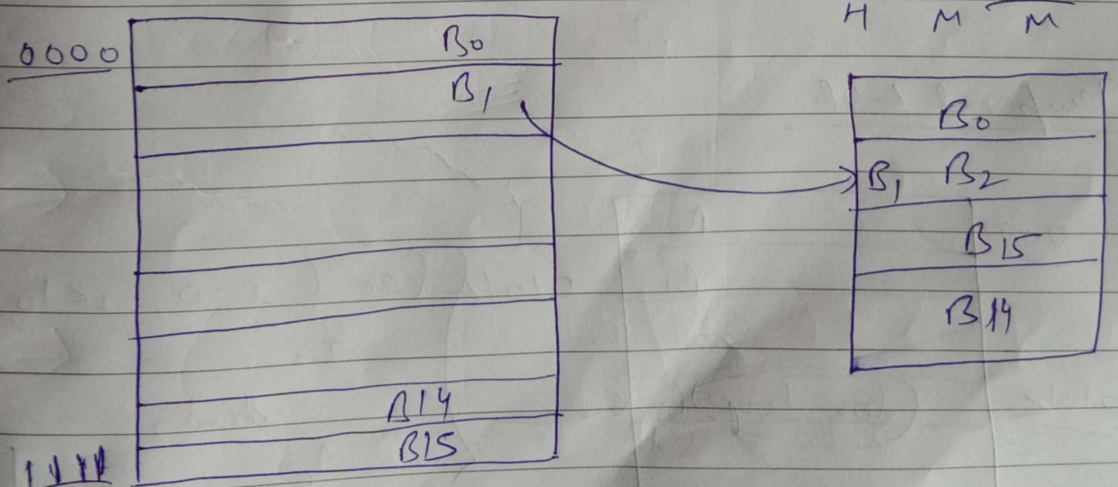


Locality of References
- Spatial
- Temporal

Locality of References
- Spatial
- Temporal.

Cache hit & miss

$\frac{B_0}{H}$ $\frac{B_1}{M}$ $\frac{B_{14}}{M}$ $\frac{B_{15}}{H}$ $\frac{B_6}{H}$



Hit Ratio & Miss Ratio.

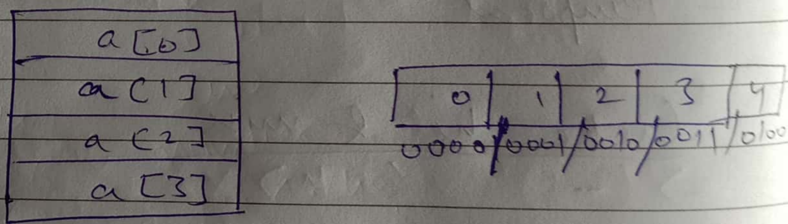
$HR = \frac{\text{No. of Hits}}{\text{Total}}, 1 - MR \quad = \frac{3}{5}$

$MR = \frac{\text{No. of Miss}}{\text{Total}}, 1 - HR. \quad = \frac{2}{5}$

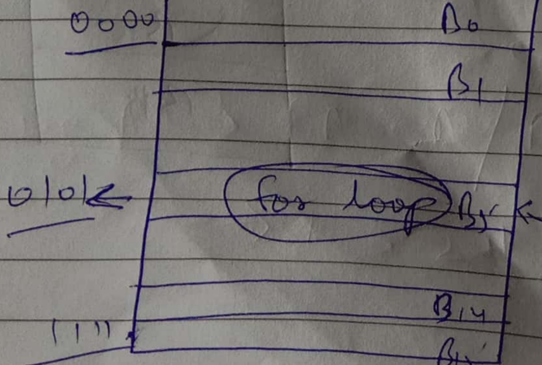
Types of Maping Techniques.

1. Direct Maping
2. Associative Maping
3. set Associative Maping.

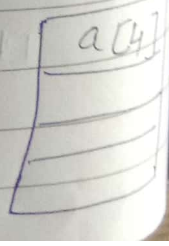
→ ~~Sep~~ Spatial LOR



→ Temporal LOR



0101 for (i=0; i<10; i++)
{
Stmts;
}.



31/1/19.

Direct Mapping

at a time only one block

M.M (128 bytes)

B_0	⇒	00 $\boxed{00}$
B_1	⇒	00 $\boxed{01}$
B_2	⇒	00 $\boxed{10}$
B_3	⇒	00 $\boxed{11}$
B_4	⇒	$\boxed{0100}$
B_5	⇒	0101
B_6	⇒	0110
B_7	⇒	0111
B_8	⇒	1000
B_9	⇒	1001
B_{10}	⇒	1010
B_{11}	⇒	1011
B_{12}	⇒	1100
B_{13}	⇒	1101
B_{14}	⇒	1110
B_{15}	⇒	1111

5th line requirement

C.M (32 Bytes)

Compare, line →	00 ⇐	B_0, B_4, B_8, B_{12}
line →	01 ⇐	B_1, B_5, B_9, B_{13}
line →	10 ⇐	B_2, B_6, B_{10}, B_{14}
line →	11 ⇐	B_3, B_7, B_{11}, B_{15}

Block Size = 8 bytes

No. of blocks $= \frac{\text{MM Size}}{BS} = \frac{128}{8} = 16$

LSB = To store block

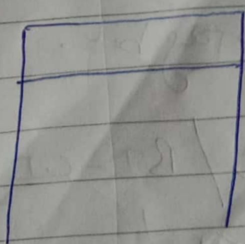
MSB = To store blocks in Cache Memory.

Block Index $= 2^4 = 16 \Rightarrow 4$ bits → BI

No. of lines $= \frac{CS}{LS/BS} = \frac{32}{8} = 4$

Cache Index $= 2^2 = 4 = 2$ bits → CI.

Tag Directory –

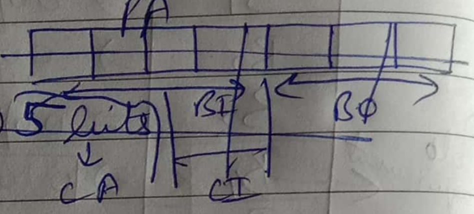


Blocksize

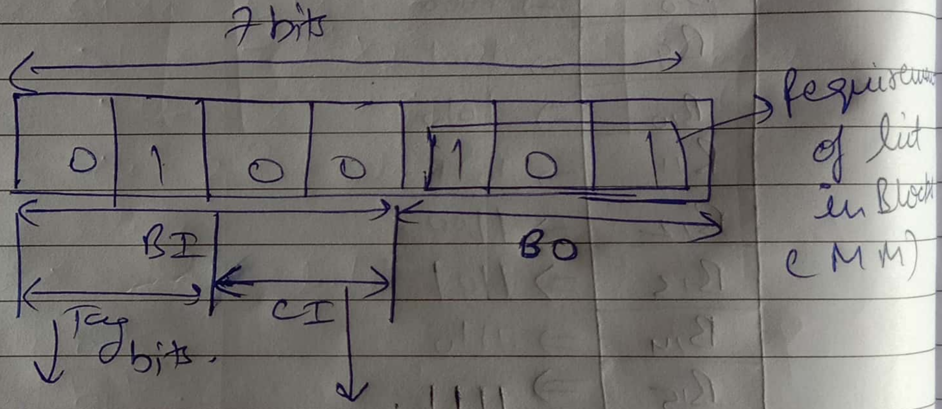
Block offset → 8 bytes = 2^{3} ⇒ 3 bits

Physical Address → 128 bytes ⇒ 2^{7} = 7 bits → memory size

~~Cache Address → 32 = 2^{5} ⇒ 5 bits~~



Cache Address → 32 = 2^{5} ⇒ 5 bits



It is present in cache already if there is 01 in Tag bit.

It is line number for cache memory.

Tag Bit → PA-CA / BI-CI

→ 7-5=2 / 4-2=2.

2/2/19

In Case of Associative Mapping
Tag Bits = Block Index

M.M Size = 1 GB = 2^{30} bytes = 1024.
C.M. Size = 256 Bytes = 2^8
B S = 64 bytes = 2^6.

No. of blocks = $\frac{1G}{64} = \frac{2^{30}}{2^6} = 2^{24}$

No. of lines = $\frac{2^8}{2^6} = 2^2 = 4$

Block Index = 2^{24}

Cache Index = 2^{2}

Physical Address = 30 ~~bytes~~ bits

Cache Address = 8 ~~bytes~~ bits

Tag = 22 bits.

To know the actual bytes
BO = 6 bits → Power Size of Block Size.

$2^{10} = 1KB$ $2^{40} = 1TB$

$2^{20} = 1MB$

$2^{30} = 1GB$.

e.g.-2 M.M no. of Block ~~size~~ = → 64.

C.M. Size = 128 bytes.

BS = 32.

M.M. Size = 64×32 = 2048.

e.g.3 No. of lines → 64

~~Block size~~/ line size − 8 bytes

No. of blocks − 512 = 2^{9}

$CS = 64 \times 8 = 512 = 2^{9}$

$MM = 512 \times 8 = 2^9 \times 2^3 = 2^{12} = 4096$ bytes.

$BI = 512 = 2^{9} = 9$ bits

$CI = 64 = 2^{6} = 6$ bits

Tag = 3 bits

$BO = 8 = 2^{3} = 3$ bits

PA = 12 bits

CA = 9 bits.

e.g 4

→ M.M Size = $1TB = 2^{40}$

C. Size → 2046

Block Size → 64 bytes.

~~M.M. Size =~~

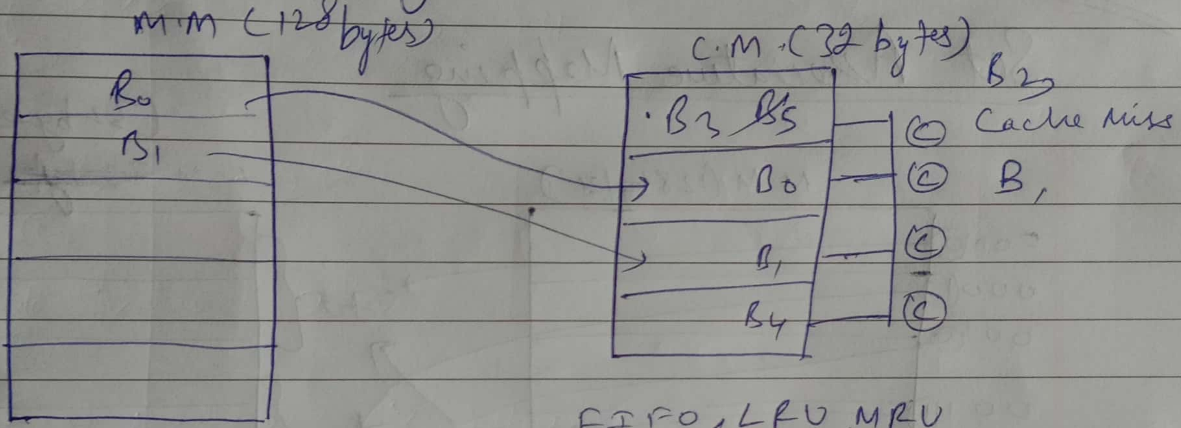
No. of blocks = $\frac{2^{40}}{2^6} = 2^{34}$

BI = 34 bits

No. of lines = $\frac{2^{11}}{2^6} = 2^{5}$

CI = 5
PA = 40
CA = 11
TB = 29
BO = 6

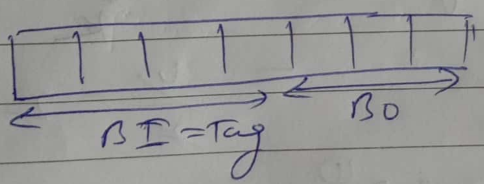
Associative Mapping



FIFO, LRU, MRU

BS = 8 bytes.   Tag Bytes = Block Index

No. of blocks = 7.



e.g.-1 Cache Size :- 512 bytes
no. of lines :- 16
MM Size = 64KB.

~~Tag Bits~~ Tag Bits.
DM - 7 bits
AM - 11 bits

64

$512 \times 16 = 3072$

$\frac{512}{16} = \frac{2^9}{2^4} = 2^5$

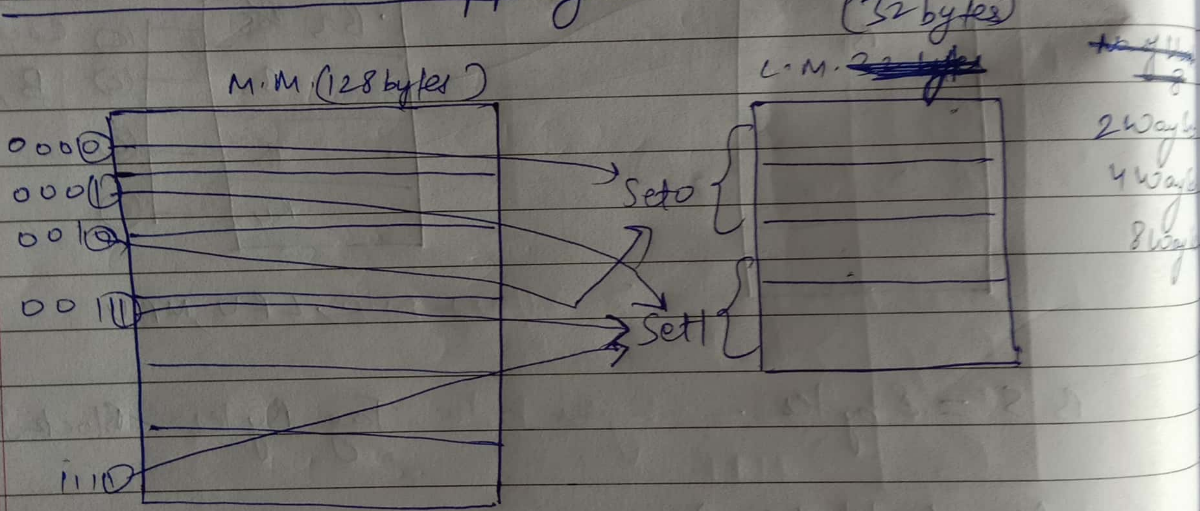
MM Size $= 64\,KB$
$= 2^6 \cdot 2^{10} = 2^{16}$

Cache line size =

64

4/2/19

Set Associative Mapping



~~Set Index~~ No. of sets $= \frac{\text{no. of lines}}{K}$

$= \frac{4}{2} = 2$

Set Index / No. $-2 = 2^1 = 1$ bit $\rightarrow$ Set Index.

Tag Bit = Block Index − Set Index / PA − CA

$= 4 - 1$

$= 3$ bits

Block Index = 4.

No. of lines $= \frac{CS}{BS}$

Physical Address = 40 bits.
Cache Size = ~~5~~ 512 KB.
It is 8 way Set Associative.
~~2^{40}~~
BS = 32 bytes.
~~SI~~ SI →
TB →

No. of lines = $\frac{2^{40}}{2^{9}}$

Physical Address = 40 bits.
Cache Size = 512 KB
It is a 8 way Set Associate
BS = 32 bytes.

SI →
TB →
Main Memory Size = 2^{40}.
No. of lines → ~~2^{40}~~

~~Block Index~~
No. of blocks = $\frac{2^{40}}{2^{5}}$
$= 2^{35}$

~~No. of sets = $\frac{2^{40}}{8}$~~
=

~~No. of sets =~~ No. of lines = $\frac{2^{9} + 2^{10}}{8}$
$= \frac{2^{19}}{2^{5}} = 2^{14}$

$1KB = 2^{10}$
$1MB = 2^{20}$
$1GB = 2^{30}$
~~...~~ $1TB = 2^{40}$

SI, TB

Q→ No. of lines = 128
4 way set associative
line size = 64 bytes.
PA = 20 bits.

$\frac{\text{no of lines}}{K}$ = ~~Set Index~~ No. of sets

$\frac{128}{4}$ = ~~Set Index~~ No. of sets

= 32 = No. of Sets
= 2^{5} = 5 = Set Index.

$BI = \frac{2^{20}}{2^{6}} = 2^{14}$

TB = BI - SI
= 14 - 5.

Q→ Cache Size = 512 KB
M.M. = 1GB.
4 Way Set Associative
BS = 64 bytes.

No.

$BI = 2^{6}$

$\frac{MM}{BS} = \frac{2^{30}}{2^{6}} = 2^{24}$ = No. of blocks

No. of lines = $\frac{2^{9}+2^{10}}{2^{6}} = \frac{2^{19}}{2^{6}} = 2^{13}$

Set Index = $\frac{2^{13}}{2^{2}} = 2^{11}$

Scanned by CamScanner

BO = power size of BS.

classmate
Date
Page

$TB = BI - SI$
$= 24 - 11$
$= 13$

Q→ Cache Size = 1024 bytes
M.M. = 1TB
BS = 64 bytes
It is 16 way Set Associative K=16.

SI ⇒
BO ⇒.

No. of blocks = $\frac{2^{40}}{2^6} = 2^{34}$.

Block Index = 34.
Block Size = 64
2^{6}
BO ⇒ 6

No. of lines = $\frac{1024}{64} = \frac{2^{10}}{2^6} = 2^{4}$

Set Index = $\frac{2^{4}}{2^{4}} = 1$

Tag Bits = 34 - 1
= 33.

1001
0101 X
0000
+ 1001
1001
0000X
1001
1001XX
101101
0000XXX
101101

1001 X 101
1001
0000 X
1001 XX
101101