1. What is response time?

- Response time is also known as execution time, refers to the total total time it takes for a computer system to complete any task or program from start to finish.

$$\text{Execution time} = \frac{\text{Total Cycle}}{\text{Clock Rate}}$$

2. What is throughput?

- Throughput refers to the number of tasks or instruction that a computer can complete in a given amount of time.

$$\text{Throughput} = \frac{\text{total instruction executed}}{\text{total execution time}}$$

3. Elapsed time = CPU time + Wait time

Elapsed time : Counts everything (disk and memory access, waiting for input/output for other program) from start to finish.

CPU time : doesn't count waiting for I/O or time spent running other program.
Can be devided into two part user cpu + system cpu.

Wait time : time spent executing the lines of code that are in our program.

Definition of performance :

$$\text{performance } X = \frac{1}{\text{execution time } X}$$

performance of X > Performance of Y

1/execution time X > 1/execution time Y

execution time Y > execution time X

Example of Relative performance -

Afia's computer execution time = 10S

Tahmid's computer execution time = 15S

Afia's computer performance > Tahmid's computer performance

$$\text{execution time} = \frac{\text{Execution time of tahmid}}{\text{Execution time of Afia}}$$

$$= \frac{15}{10} = 1.5\,s.$$

Afia's computer runs 1.5 s faster than Tahmid's computer.

$$\text{Clock Cycle} = \frac{\text{seconds}}{\text{program}} = \frac{\text{Cycles}}{\text{program}} \times \frac{\text{seconds}}{\text{Cycle}}$$

CPU execution time = Clock cycle x Clock cycle time

$$= \frac{\text{Clock cycle}}{\text{Clock rate}}$$

Performance equation : 1

Execution time of computer A = 10 s

Clock rate of A = 400 MHz

Clock cycle of A = execution time of computer A x Clock rate of A

$= 10 \times 400 = 4000$ MHZ

Clock cycle of B = 1.2 x Clock cycle of A

$= 1.2 \times 4000$

$= 4800$ MHZ

Clock rate of B = $\frac{4800}{6} = 800$ MHz

CPU execution time = Instruction count x CPI x Clock cycle time

Performance equation : 2

Both machine A and Machine B has same ISA.

machine A clock cycle = $\overline{10\,ns} = 10 \times 10^{-9}$ s

CPI of A = $\overline{2.0}$

execution time of A = $IC \times 2.0 \times 10 \times 10^{-9}$

$= IC \times 20 \times 10^{-9}$

machine B clock cycle = $\overline{20\,ns} = 20 \times 10^{-9}$ s

CPI of B = 1.2

execution time of B = $IC \times 1.2 \times 20 \times 10^{-9}$

$= IC \times 24 \times 10^{-9}$

$\therefore$ execution time = $\frac{IC \times 24 \times 10^{-9}}{IC \times 20 \times 10^{-9}} = 1.2$

So, machine A is 1.2 times faster than machine B.

CPI Example 2 :

CPI

A	B	C
1	2	3

Code Sequ.	A	B	C
1	2	1	2
2	4	1	1

CPU1 clock cycle = CPI x Instruction

$= (1 \times 2) + (2 \times 1) + (3 \times 2)$

$= 2 + 2 + 6 = 10$ cycles

CPU2 clock cycle = $(1 \times 4) + (2 \times 1) + (3 \times 1)$

$= 9$ cycles

Code sequence 2 fasten.

$CPI\,1 = \frac{10}{5} = 2$

$CPI\,2 = \frac{9}{6} = 1.5$

Code	instruction count		
Sequence	A	B	C
1	5 M	1M	1M
2	10M	1M	1M

Clock rate = 100 MHZ

$= 100 \times 10^6$ HZ

Clock cycle,

$C1 = (1 \times 5 \times 10^6) + (2 \times 10^6) + (3 \times 10^6)$

$= 1 \times 10^7$

$C2 = (1 \times 10 \times 10^6) + (2 \times 10^6) + (3 \times 10^6)$

$= 1.5 \times 10^7$

Execution time for, $1 = \frac{1 \times 10^7}{100 \times 10^6}$

$= 0.1S$

Execution time for $2 = \frac{1.5 \times 10^7}{100 \times 10^6}$

$= 0.15\,S$

$MIPS = \frac{\text{Instruction}}{\text{Execution time} \times 10^6}$

For 1, $(MIPS) = \frac{7 \times 10^6}{0.1 \times 10^6}$

$= 70$

For 2, $(MIPS) = \frac{12 \times 10^6}{0.15 \times 10^6}$

$= 80$

ans.

Lecture-1

1. What is computer architecture?

- Computer architecture refers to the design and organization of a computer system. It defines how different components of a computer system interact and work together to execute instruction.

Computer architecture includes three main aspects:

1. Instruction set Architecture: Defines the set of instruction the cpu can execute such as arithmatic operation, memory access, control flow instruction etc.

2. Microarchitecture: Describes how a processor is internally designed to implement the ISA. including register, memory cache etc.

3. System architecture: Covers the overall system design, including the cpu, memory hierarchy, input/output devices, interconnection like buses and network interface.

2. Why we learn architecture?

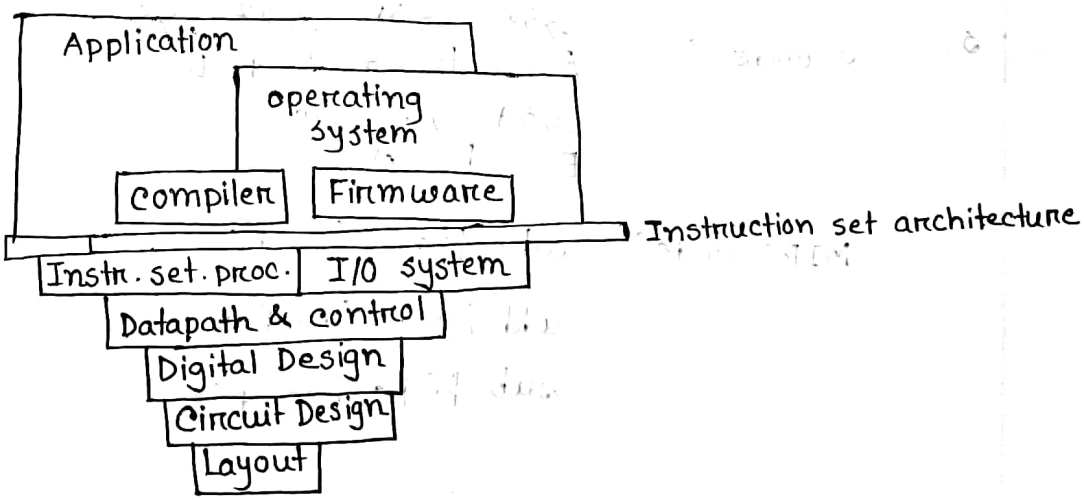
1. Understanding how computer works: It helps us to understand the internal works of a computer including data process, stored and transfer.

2. Optimizing software performance: knowledge of computer architecture allows software developers to write efficient program by leveraging hardware capabilities including caching, processing.

3. Understanding how hardware executes instructions helps designing better software.

4. Troubleshooting and debuging: A strong grasp of architecture helps in diagnosing system performance issues, debugging hardware failures and optimizing resource usage.

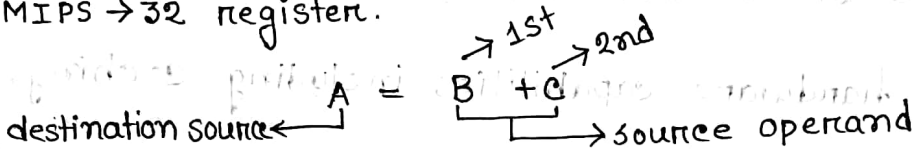
3. Level of abstraction



Lecture-3 (Language of the machine)

MIPS arithmatic

* All MIPS arithmatic instruction have 3 operand.
* operand order is fixed.
* operation code: an instruction that tells the processor what operation to perform. (Add, Sum, MUL, DIV, MOV → (move data from one register to another), Load, store, JMP)
* MIPS → 32 register.



* Converting High level to low level language:

1. C Code: $A^{S0} = B^{S1} + C^{S2}$ (Value কে register এ assign করব)

   MIPS code: add $S0, $S1, $S2

2. C Code: $A^{S0} = B^{S1} + C^{S2} + D^{S3}$

   MIPS Code: add $t0, $S1, $S2
   add $S0, $t0, $S3

3. C code: $A^{S0} = B^{S1} + C^{S2} + D^{S3}$
   $E^{S4} = F^{S5} - A^{S0}$

   MIPS code: add $t0, $S1, $S2
   add $S0, $t0, $S3
   sub $S4, $S5, $S0

R-type arithmatic instruction

Instruction Format:

6 bit	5 bit	5 bit	5 bit	5 bit	6 bit
OP	rs	rt	rd	shamt	funct
	1st	2nd	dest.		

Example: 1

C code: $A = B + C$ ($A \to S_0$, $B \to S_1$, $C \to S_2$)

MIPS code: add $S_0, $S_1, $S_2

$A = B + C$ → 16 = 17 + 18

16 → 1 0 0 0 0

17 → 1 0 0 0 1

18 → 1 0 0 1 0

000000	10001	10010	10000	00000	100000
OP	rs	rt	rd	shamt	func

Example: 2

C code: $A = B + C + D$ ($A \to S_0$, $B \to S_1$, $C \to S_2$, $D \to S_3$)

$A = B + C + D$ → 10 = 20 + 30

add = 32

sub = 35

~~MIPS~~: $F = C - A$ ($F \to S_4$, $C \to S_2$, $A \to S_0$)

$F = C - A$ → 15 = 30 − 10

-12

MIPS: add $t_0, $S_1, $S_2

add, $S_0, $t_0, $S_3

sub $S_4, $S_2, $S_0

t_0 → 0 1 1 0 0

$S_1 \to B$ → 1 0 1 0 0

1 1 1 1 0

0 1 0 1 0

$ → 1 0 0 0 0

1 0 0 0 1 1

0 1 1 1 1

000000	10100	11110	01100	00000	100000
OP	rs (S_1)	rt (S_2)	rd (t_0)	shamt	func (add)

000000	01100	10000	01010	00000	100000
op	rs (t_0)	rt (S_3)	rd (S_0)	shamt	func (add)

000000	11110	01010	01111	00000	100011
op	rs (S_2)	rt (S_0)	rd (S_4)	shamt	func (sub)

I-type instruction

1. $A[10] = B + C[8] \rightarrow t_0$ ($A \rightarrow S_0$, $B \rightarrow S_1$, $C \rightarrow S_2$)

```
=> lw  $t0 , 32($S2)
   add $S0 ,$S1, $S2
   sw  $t0, 40 ($S0)
```

2. $A[16] = B + A[0] \rightarrow t_0$ ($A \rightarrow S_0$, $B \rightarrow S_1$, $A \rightarrow S_2$)

```
lw  $t0, 0($S2)
add $t1 , $S1 , $t0
sw  $t1, 64($S0)
```

I-type instruction format:

6 bit	5 bit	5 bit	16 bit
op	rs	rt	offset

* Write down MIPS assembly code and machine code

$X[2] = B - C + Y[4] \rightarrow t_0$ ($X \rightarrow S_0$, $B \rightarrow S_1$, $C \rightarrow S_2$, $Y \rightarrow S_3$)

$t_0 \rightarrow 16$ $S_1 \rightarrow 17$ $S_2 \rightarrow 18$

$S_0 \rightarrow 19$, $S_3 \rightarrow 20$

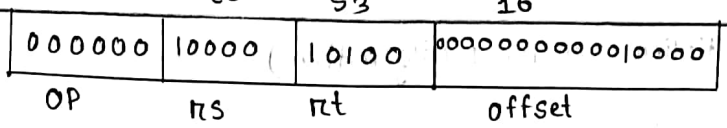
$t_1 \rightarrow 25$ $t_2 = 30$

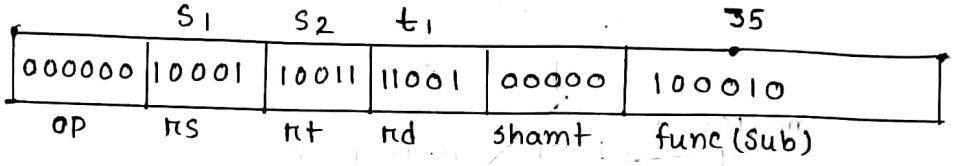
~~sub $t1, $S1, $S2~~

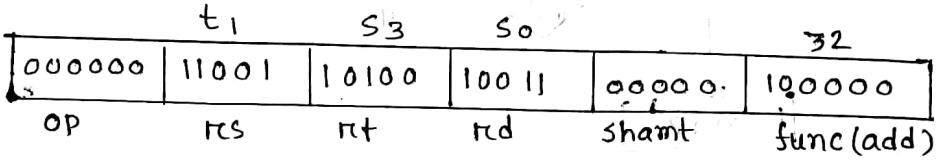
```
lw  $t0 , 16($S3)
sub $t1, $S1, $S2
~~sw~~
add $S0, $t1, $S3
sw  $t2, 8($S0)
```

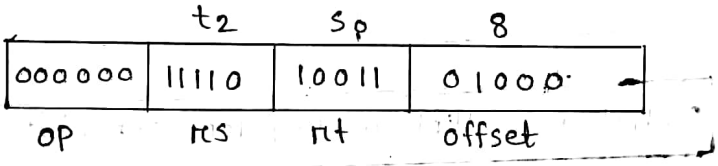
32	16	8	4	2	1
	1	0	0	0	0
	1	0	1	0	0
1	0	0	0	0	0
1	0	0	0	1	0
	0	1	0	0	0
	1	1	0	0	1
	1	0	0	1	1
	1	1	1	1	0

lw









Constants

~~Immediate operands~~ Lecture → 3

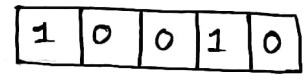
Arithmatic Right Shift

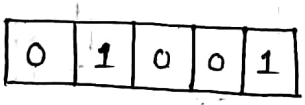
Example: 4 bit 8 bit

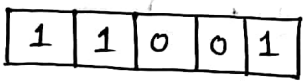
0 0 1 0 0000 0010

MSB ← (1) 0 1 0 1111 1010

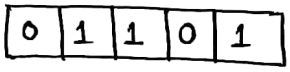
Example: 1

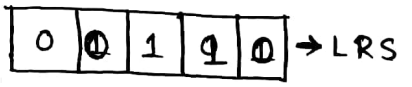


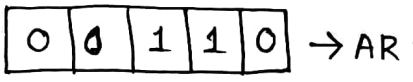




Example: 2

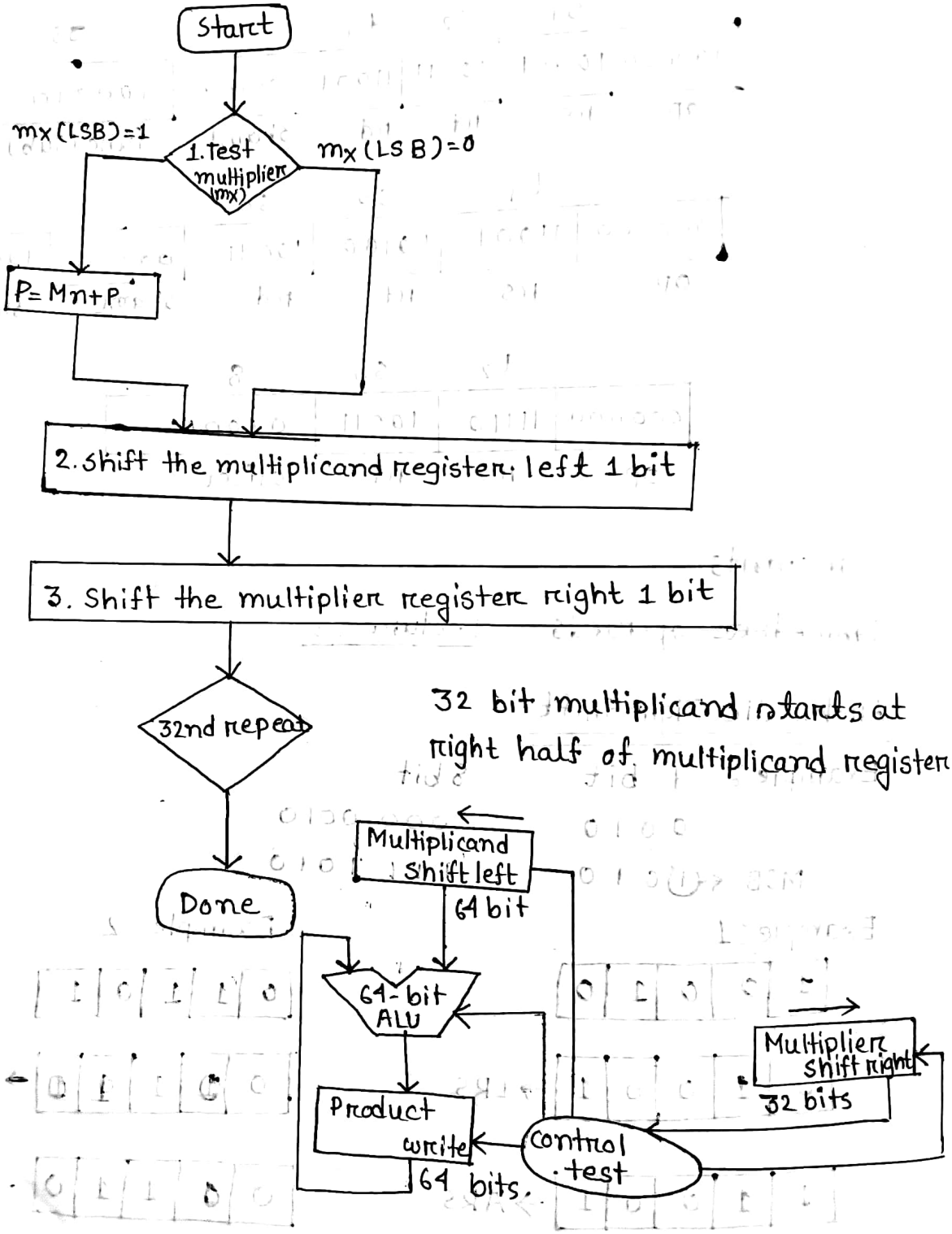






Lecture -4

Shift-add Multiplier Version 1 (algorithm)



Example : 1 (Version 1)

Mn = 3

Mx = 5

P = 3 x 5 = 15

using 4 bit multiplier :

P = 0000 0000
Mn = 0 0000 011
P = 0 0000 011
0000 1111
0000 1111

0 0 0 → 0
0 0 1 − 1
0 1 1 0 − 2
0 1 1 − 3
1 1 0 − 1
1 0 1 − 5

Sol: Mx = 4 bit , Mn = 8 bit, P = 8 bit.

Itr.	Steps	Multiplicand(Mn)	Multiplier(Mx)	Product
0	intn	0000 0011	0101 → LSB	0000 0000
1	1. Mx(LSB)=1 1a. P= Mn+P 2. Mn (1 bit left shf.) 3. Mx (1 bit right shf.)	0000 0110	0010 → LSB	0000 0011
2.	1. Mx(LSB)=0 2a. NOP 2. Mn (1 bit left Shift) 3. Mx (1 bit right Shift)	0000 1100	0001 → LSB	0000 0011
3.	1. Mx(LSB)=1 1a. p+Mn=P 2. Mn (1bit left) 3. Mx (1 bit right)	0001 1000	0000 → LSB	0000 1111
4.	1. Mx(LSB)=0 1a. NOP 2. Mn(1 bit left) 3. Mx(1 bit right)	0011 0000	0000	0000 1111

Example :2

Mn = 2
Mx = 4
} P = 2 × 4 = 8

using 4 bit multiplier

Mn = 8 bit

P = 8 bit.

Itr	Steps	Multiplicand	Multiplier (Mx)	Product
0	int n	00000010	010◎ LSB	0000 0000
1	1. Mx = 0 1a. NOP 2. Mn ← (1 bit left shift) 3. Mx → (1 bit right shift)	00000100	00 1◎ → LSB	0000 0000
2.	1. Mx (LSB) = 0 1a. NOP 2. Mn (left shift) 3. Mx (right shift)	0000 1000	0001	0000 0000
3.	1. Mx (LSB) = 1 1a. Mn + P = P 2. Mn (left shift) 3. Mx (right shift)	00010000	000◎ → LSB	0000 1000
4.	1. Mx (LSB) = 0 1a. NOP 2. Mn (Left shift) 3. Mx (Right shift)	00100000	000	0000 1000