NOTRE DAME UNIVERSITY BANGLADESH

Computer Architecture
Assignment Mid

Submitted to: Mondira Chakraborty (LecturerCSE)

Submitted by: Shazidul Alam

Student ID: 0692220005101009

Batch: CSE-19

Lecture → 1 (Introduction)

(25-08-24)

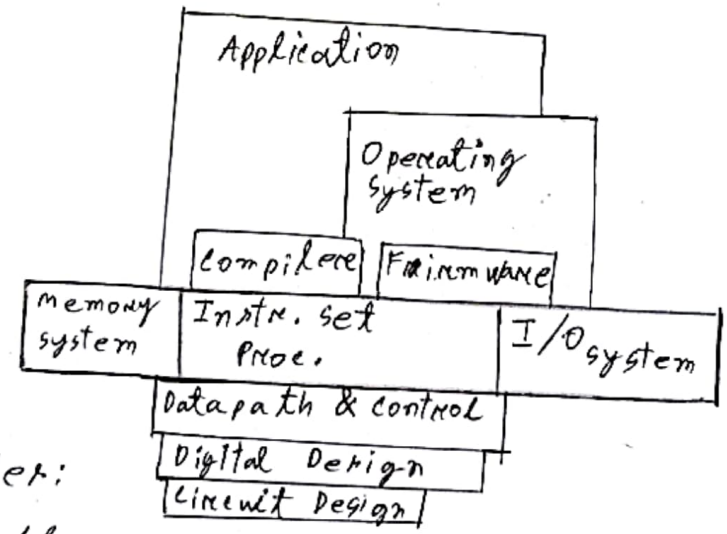
1) Definition of computer Architecture?

=> In computer engineering, computer architecture is a set of rules and methods that describe the functional organization, and implementation of computer systems. Some definitions of architecture define it as describing the capabilities and programming model of a computer but not a particular implementation. In other descriptions computer architecture ~~work~~ involves instruction set architecture design, micro architecture design, logic design and implementation.

* Computer architecture = Instruction set architecture + Machine organization + Hardware

2) Instruction set Architecture (ISA)?

=> ISA is the part of the processor that is visible to the programmer or compiler writer. The ISA serves as the boundary between software and hardware.



(ISA)

Example: MIPS

Desired properties:

a) Convenience (from software side)

b) Efficiency (from hardware side)

3) Machine Organization?

=) High-level aspects of a computer's design

- Principal components: memory, CPU, I/O, ...
- How components are interconnected
- How information flows between components.
- e.g. AMD Opteron 64 and Intel Pentium 4: Same ISA but different organizations.

4) Hardware ?

=) Detailed logic design and packaging technology of a computer

- E.g. Pentium 4 and Mobile pentium 4: nearly identical organizations but different hardware details.

5) Types of computers and their applications ?

=) a) Desktop → Run third-party software
→ Office to home applications

b) Servers
→ Modern version of what used to be called mainframes, minicomputers and supercomputers
→ Large workloads
→ Built using the same technology in desktops but higher capacity: i) Expandable
ii) Scalable
iii) Reliable

c) Large Spectrum: from low-end (file storage, small businesses) to supercomputers (high end scientific and engineering applications)

Examples: file servers, web servers, database servers.

d) Embedded?

=> i) Microprocessors everywhere! (washing machines, cell phones, automobiles, video games).

ii) Run one or a few applications.

iii) Specialized hardware integrated with the application.

iv) Usually stringent limitations (battery power)

v) High tolerance for failure.

vi) Becoming ubiquitous.

vii) Engineered using processor cores:

→ The core allows the engineer to integrate other functions into the process for fabrication on the same chip.

→ Using hardware description languages: Verilog, VHDL

---

* Computer organization/Anatomy:

Input → Memory → Datapath → Processor → Memory → Output

1) Input: Keyboard, Mouse

2) Memory: Where programs, data live when running.

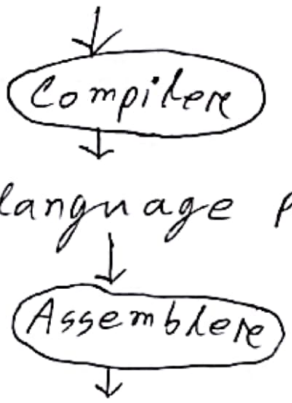
3) Control: (Brain) Guides the operation of other components based on the user instructions.

4) Datapath: (Brawn) Performs arithmetic operation

5) Output: Display, printer.

(28-08-2024)

(6) High-level language in C



---

(7) Benifits of High level Programming language?

=> 1) Allow the programmere to think in a more natural language using English words and algebraic notation. which looks much more like text.

2) Allow languages to be designed according to their intended use. Fortran used for scientific computation, Cobol for business data processing, Lisp for symbol manipulation and so on.

3) Another advantage of programming language is improved programmere poductivity.

4) Widespread agreement is that it takes less time to develop programs when they are written in languages that require fewer lines to express an idea.

5) Compilers and assemblers can translate high-level language programs to the binary instructions of any computer.

There advantages are so strong that today very little programming done in assembly language.

(Chapter-2)

(28-08-2024) (Performance)

① Response Time / Execution time

→ task

② Throughput: Total amount of work done in a give line. Example:

1 task — 5 sec

15 sec — 3 task (throughput)

③ Performance: To maximize performance, response time or execution time should be minimized.

Example: For computer X, $\text{Performance}_x = \frac{1}{\text{Execution time}_x}$

$\text{Performance}_x > \text{Performance}_y$

$\frac{1}{\text{Execution time}_x} > \frac{1}{\text{Execution time}_y}$

$\text{Execution time}_y > \text{Execution time}_x$

$\frac{\text{Performance}_x}{\text{Performance}_y} = \frac{\text{Execution time}_y}{\text{Execution time}_x} = n$

* ④ If computer A runs a program in 10 seconds and computer B runs the same program in 15 seconds, how much faster is A than B?

=> Here, Execution time of A = 10 sec
Execution time of B = 15 sec

$\therefore$ Performance of $A = \frac{1}{10} = 0.1$

$\therefore$ " " $B = \frac{1}{15} = 0.067$

$\therefore \frac{\text{performance } A}{\text{Performance } B} = \frac{0.1}{0.067} = 1.49$

$\therefore$ A is 1.49 times faster than B

Substitute: $\frac{\text{Performance } A}{\text{performance } B} = \frac{15}{10}$

$\Rightarrow \frac{P_A}{P_B} = 1.5$

$\therefore P_A = 1.5 P_B$

(01-09-2024)

(5) Measuring Performance?

Ans: a) The computer that performs the same amount of work in the least time is the fastest.

b) Program execution time is measured in seconds per program.

c) Time is also called, wall clock time, response time, execution time or elapsed time.

d) These means total time to complete a task. including disk accesses, memory accesses, input/output (I/O) activities, operating system overhead everything.

e) Processors may work work on several program simultaneously.

f) Cpu execution time or cpu time which recognizer the distinction, is the time the cpu spends computing for this task and does not include time spent waiting for I/O or running other programs.

---

(6) Cpu time - a) User CPU time
b) System CPU time

(7) User CPU time: CPU time spent in the program is called User CPU time.

(8) System CPU time: CPU time spent in the operating system performing tasks requested by the program is called System CPU time.

The term system performance is used to refer to elapsed time on an unloaded system while CPU performance refers to user CPU time on an unloaded system.

1) Atto second $\longrightarrow 10^{-18}$
2) Femto " $\longrightarrow 10^{-15}$
3) Pico " $\longrightarrow 10^{-12}$
4) Nano " $\longrightarrow 10^{-9}$
5) Micro " $\longrightarrow 10^{-6}$
6) Mili " $\longrightarrow 10^{-3}$
7) Centi " $\longrightarrow 10^{-2}$
8) Deci " $\longrightarrow 10^{-1}$
9) **Second** $\longrightarrow 1$
10) Deca Hz $\longrightarrow 10^{1}$
11) Hecto Hz $\longrightarrow 10^{2}$
12) Kilo Hz $\longrightarrow 10^{3}$
13) Mega Hz $\longrightarrow 10^{6}$
14) Giga Hz $\longrightarrow 10^{9}$
15) Tera Hz $\longrightarrow 10^{12}$

(9) Clock Period: Time for a complete clock cycle. (T)

(10) Clock rate: Inverse of clock period.

(11) Math: Clock period = 250 ps

$$\therefore \text{Clock rate} = \frac{1}{\text{clock period}}$$

$$= \frac{1}{250\ ps}$$

$$= \frac{1}{250 \times 10^{-12}\ s}$$

$$= 4 \times 10^{-3} \times 10^{12}\ Hz$$

$$= 4 \times 10^{9}\ Hz$$

$$= 4\ GHz$$

(04-09-2024)

12) Features of a clock?

Ans: **Period** is the time it takes for a clock signal to repeat. It's unit of measurement is seconds. The symbol for the period is T.

Frequency is 1/T. It's unit is s^{-1}, which is inverse seconds. This also called Hz. Sometimes "cycles per second". They all mean the same. "Speed" of a CPU also means the clock frequency.

Positive edge: The signal transitions from 0 to 1.

Negative edge: The signal transitions from 1 to 0.

---

13) Why clock used for?

Ans: We use a clock to synchronize the events of a CPU. Devices in a CPU can be categorized as sequential circuits or combinational circuits. Sequential logic devices use a clock. Sequential logic devices can only change output once a period, usually during a positive or a negative edge.

---

14) Can't we make CPU faster?

Ans: It takes time for signal to move around in a circuit. It can be reduced in several ways. The main way is to make the circuit smaller. When the circuit is smaller, signals have less distance to travel and therefore everything can happen more quickly.

15) CPU Performance and It's Factors ?

* CPU Execution time for a program = CPU clock cycles for a program × clock cycle time

* CPU Execution time for a program = $\frac{\text{CPU clock cycles for a program}}{\text{clock rate}}$

---

16) Our favourite program runs in 10 seconds on computer A, which has 2 GHz clock. We need to build computer B which will run this program in 6 seconds. So clock rate must be increased. This will cause computer B to require 1.2 times as many clock cycles as computer A for this program. What clock rate should we tell the designer to target?

Ans: Here, CPU Execution time, A = 10s
CPU clock rate, A = 2 GHz
CPU Execution time, B = 6s

Let, computer A clock cycle = x
computer B clock cycle = 1.2 × x

Computer A CPU Execution time = $\frac{\text{Computer A clock cycle}}{\text{clock rate computer A}}$

$\Rightarrow 10s = \frac{x}{2\,GHz} \Rightarrow 10s = \frac{x}{2 \times 10^9\,Hz} \Rightarrow x = 20 \times 10^9$ sec

∴ Computer A clock cycle $x = 20 \times 10^9$ cycles

CPU Execution time, B = $\frac{\text{clock cycle B}}{\text{clock rate B}}$

$\Rightarrow$ clock rate, B = $\frac{\text{clock cycle B}}{\text{CPU Execution time B}}$

$\Rightarrow$ clock rate, B = $\frac{1.2 \times 20 \times 10^9}{6s} = 4 \times 10^9 s^{-1} = 4\,GHz$

(08-09-2024)

17) Suppose we have two implementations of the same instruction set architecture. Computer A has a clock cycle time of 250 ps and a CPI of 2.0 for some program. and computer AB has a clock cycle time of 500 ps and a CPI of 1.2 for the same program.

Which computer is faster for this program and by how much?

Ans: Here, clock cycle time, A = 250 ps

clock cycle time, B = 500 ps

$CPI_A = 2$

$CPI_B = 1.2$

clock cycle time

CPU clock cycle, A = I × ~~CPI~~ = I × ~~250~~ 250 ps

" " ", B = I × ~~CPI~~ = I × 500 ps

CPU Execution time, A = CPU clock cycle × ~~clock cycle time~~ CPI

= ~~200~~ 250 I × 2

" " ", B = CPU clock cycle × ~~cloc cycle time~~ CPI

= 500 I × 1.2

$$\frac{\text{performance A}}{\text{performance B}} = \frac{2.2\text{I} \times 272}{4.8\text{I} \times 125} \rightarrow \frac{\text{CPU Execution time B}}{\text{CPU Execution time A}}$$

$$= \frac{500\text{I} \times 1.2}{250\text{I} \times 2}$$

$$= 1.2$$

18) A compiler designer is trying to decide between two code sequences to a particular computer. The hardware designers have supplied the following facts:

CPI for each instruction class			
	A	B	C
CPI	1	2	3

For a particular high-level language statement, the compiler writers is considering two code sequences that require the following instruction ~~facts~~ counts:

	Instruction counts for each instruction class		
	A	B	C
1	2	1	2
2	4	1	1

Which code sequence executes the most instructions? Which will be faster? What is the CPI for each sequence?

Ans: Sequence 1 executes = 2+1+2 = 5 instruction

" 2 " = 4+1+1 = 6 "

∴ CPU clock cycles$_1$ = (2×1)+(1×2)+(2×3) = 10 cycles

∴ " " " $_2$ = (4×1)+(1×2)+(1×3) = 9 cycles

$$\therefore CPI_1 = \frac{\text{CPU clock cycles}_1}{\text{Instruction count}_1} = \frac{10}{5} \; 2.0$$

$$\therefore CPI_2 = \frac{\text{CPU clock cycles}_2}{\text{Instruction count}_2} = \frac{9}{6} = 1.5$$

(11-09-2024)

~~Ins~~ Lecture-3 (Instruction set)

Instruction set: The words of a computer's language are called instructions and its vocabulary is called and instruction set. The chosen instruction set comes from MIPS technologies. MIPS is a reduced instruction set computer (RISC) instruction set architecture (CISA) develop by MIPS Technologies.

MIPS → (Microprocessor without Interlocked Pipeline Stages).

---

1) a = b + c + d + e (Translate c code into assembly code)

=> add a, b, c
add a, a, d
add a, a, e

or,

add a, b, c
add t, d, e
add a, a, t

---

2) t = (g+h) - (i+j)

=> add t0, g, h
~~add~~
add t1, i, j
sub t, t0, t1

or,

add t, g, h
sub t, t, i
sub t, t, j

---

3) Register: It is a small, fast storage location within a CPU used to hold data temporarily for processing.

(15-09-2024)

Example-3 $f = (g+h) - (i+j)$

```
add  $t0, $S1, $S2
add  $t1, $S3, $S4
sub  $S0, $t0, $t1
```

$t_0 = g+h$

$t_1 = i+j$

$f = t_0 - t_1$

or,

$f = (g+h) - i - j$

```
add  $t0, $S1, $2
sub  $t1, $t0, $S3
sub  $S0, $t1, $S4
```

$t_0 = g+h$

$t_1 = t_0 - i$

$f = t_1 - j$

$a = (b+d) - (-j+K) - c$

$\Rightarrow a = (b+d) + j - K - c$

```
add $t0, $S1, $S3
add $t1, $t0, $S4
sub $t2, $t1, $S5
sub $S0, $t2, $S2
```

$t_0 = b+d$

$t_1 = t_0 + j$

$t_2 = t_1 - k$

$a = t_2 - c$

(22-09-2024)

(i) $P = (-q-r)-t-(m-n)$

$= -(q+r) - t + m + n$

$= m+n-(q+r)-t$

Let,
$\$s_0 = p$, $\$s_1 = m$
$\$s_2 = n$, $\$s_3 = q$
$\$s_4 = r$, $\$s_5 = t$

=>

add $\$t_0$, $\$s_1$, $\$s_2$	$t_0 = m+n$
add $\$t_1$, $\$s_3$, $\$s_4$	$t_1 = q+r$
sub $\$t_2$, $\$t_0$, $\$t_1$	$t_2 = t_0 - t_1$
sub $\$s_0$, $\$t_2$, $\$s_5$	$p = t_2 - s_5$

(ii) $a = (b+c)-(c-d)-(e+f)$

where, $a = \$s_0$, $b = \$s_1$

$c = \$s_2$, $d = \$s_3$

$e = \$s_4$, $f = \$s_5$

=>

add $\$t_0$, $\$s_1$, $\$s_2$	$t_0 = b+c$
sub $\$t_1$, $\$s_2$, $\$s_3$	$t_1 = c-d$
sub $\$t_2$, $\$t_0$, $\$t_1$	$t_2 = t_0 - t_1$
add $\$t_3$, $\$s_4$, $\$s_5$	$t_3 = e+f$
sub $\$s_0$, $\$t_2$, $\$t_3$	$a = t_2 + t_3$

(iii) $g = h + A[8]$

Assembly:

```
lw $t0, 8($s3)
add $s1, $s2, $t0
```

(25-09-2024)

Assume variable h is associated with register $s2 and the base address of the array A is in $s3. What is the MIPS assembly code for the C statement below?

```
A[12] = h + A[8];
↓          ↓      ↓
Array name  Variable  Array name
```

Ans:

```
lw $t0, 32($s3)
add $t0, $s2, $t0
sw $t0, 48($s3)
```

Arr[124] = Arr[22] - Arr[25] + h + (p+q)

Ans:

```
lw $t0, 88($s1)
lw $t1, 100($s1)
sub $t1, $t0, $t1
add $t2, $t1, $s2
add $t2, $t2, $s3
add $t2, $t2, $s4
sw $t2, 996($s1)
```

t_0 = Arr[22]

t_1 = Arr[25]

$t_1 = t_0 + t_1$

$t_2 = t_1 + s_2$

$t_2 = t_3 + s_3$

$t_2 = t_2 + s_4$

(29-09-2024)

1/ A[22] = A[20] + B[12] + C[5] → [A base = $s1, B base = $s4, C base = $s3]

=>

lw $t0, 80($s1)	t0 = Arr[20]
lw $t1, 48($s4)	t1 = Arr[12]
lw $t2, 20($s3)	t2 = Arr[5]
add $t3, $t0, $t1	t3 = t0 + t1
add $t3, $t3, $t2	t3 = t3 + t2
sw $t3, 88($s1)	t3 = Arr[22]

2/ g = h + arr[255]

p = g + h + s

arr[253] = arr[23] + p + g

[$s1 = arr[253], $s2 = g, $s3 = h, $s4 = s, $s5 = p]

=>

~~add $t0, $t0, $s2~~

lw $t0, 1020($s1)	t0 = arr[255]
add $s2, $t0, $s3	g = t0 + h
add $t1, $s2, $s3	t1 = g + h
add $s5, $t1, $s4	p = t1 + s
add $t2, $s5, $s2	t2 = p + g
lw $t3, 92($s1)	t3 = arr[23]
add $t4, $t3, $t2	t3 = t3 + t2
sw $t4, 1012($s4)	t4 = t3 + t2

NOTRE DAME UNIVERSITY BANGLADESH

Computer Architecture
Assignment Final

Submitted to: Mondira Chakraborty (LecturerCSE)

Submitted by: Shazidul Alam

Student ID: 0692220005101009

Batch: CSE-19

(20-10-2024)

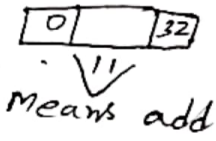
Lecture-4 | Representing Instructions in the Computers

1) **Instruction**: Instructions are kept in the computer as a series of high and low electronic signals and may be represented as numbers.

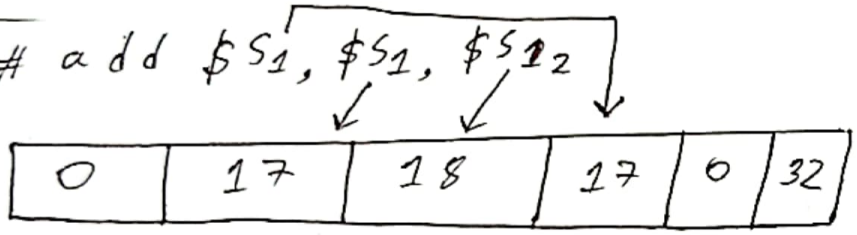
$t0 = 8	$s0 = 16
$t1 = 9	$s1 = 17
$t2 = 10	$s2 = 18
$t3 = 11	$s3 = 19
$t4 = 12	$s4 = 20
$t5 = 13	$s5 = 21
$t6 = 14	$s6 = 22
$t7 = 15	$s7 = 23

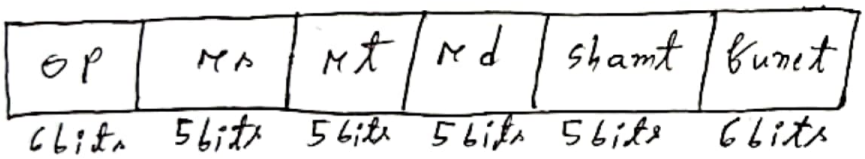
add $s1 + $s2 = $t0

$s1 ↓ first operand; $s2 ↓ second operand



add $s1, $s1, $s2





rs : register source 1st operand

rt : register source 2nd operand

rd : register destination

shamt : shift amount.

function : function code.

lw $t0, 32($s3)

op	19	8	address

Explain

op	rs	rt	constant or address
6 bits	5 bits	5 bits	16 bits

(23-10-2024)

add, sub, addi, lw, sw

add, sub → R type

addi, lw, sw → I type

---

a = b + c

d = e + f

p = a + d

$a = \$s_0 \rightarrow 16$

$b = \$s_1 \rightarrow 17$

$c = \$s_2 \rightarrow 18$

$d = \$s_3 \rightarrow 19$

$e = \$s_4 \rightarrow 20$

$f = \$s_5 \rightarrow 21$

$p = \$s_6 \rightarrow 22$

add $s0, $s1, $s2 — $s0 = a + b

add $s3, $s4, $s5 — $s3 = e + f

add $s6, $s0, $s3 — $s6 = a + d

2nd operand, 1st operand, destination

op	rs	rt	rd	shamt	funct
0	17	18	16	0	32
0	20	21	19	0	32
0	16	19	22	0	32

---

lw $t0, 32($s3)

35	19	8	128
op	rs	rt	address

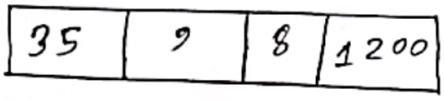
sw $t0, 32($s3)

43	19	8	128

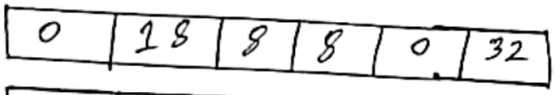
A[300] = h + A[300]

$t_0 = 8$, $t_1 = 9$, $S_2 = 18$

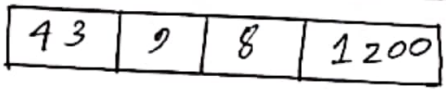
lw $t0, 1200($t1)



add $t0, $S2, $t0



sw $t0, 1200($t1)



(03-11-2024)

A base = $S0, B base = $S1

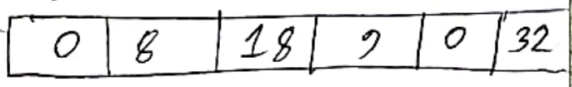
A[500] = A[412] + a + b + c

B[413] = A[500] + d

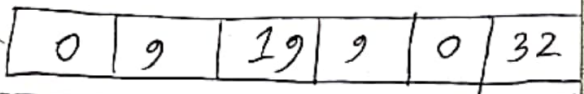
Let, a = $S2, b = $S3, c = $S4, d = $S5

=> lw $t0, 1698($S0)

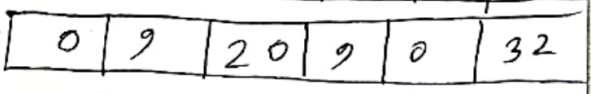
add $t1, $t0, $S2



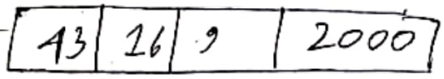
add $t1, $t1, $S3



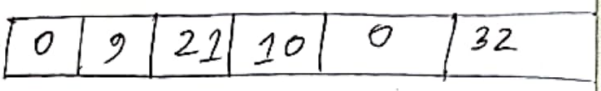
add $t1, $t1, $S4



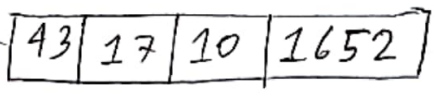
sw $t1, 2000($S0)



add $t2, $t1, $S5



sw $t2, 1652($S1)



(03-11-2024) [Lecture-5 (Logical Operations)

Logical operations	'C operator	Java Operator	MIPS I Instruction
Shift Left	<<	<<	sll
Shift Right	>>	>>>	srl
Bit-by-bit AND	&	&	and, andi
Bit-by-bit OR	|	|	or, ori
Bit-by-bit NOT	~	~	nor

i) << (Left shift)

00100 ⟶ 4

⇒01000 ⟶ 8

ii) >> (right shift)

00100 → 4

⇒00010 → 2

iii) NOT → NOR

a = 4 → 00100

$\bar{a}$ ⟶ 11011

iv) AND

a = 8 ⟶ 01000

b = 4 ⟶ 00100

AND B ⟶ 00000

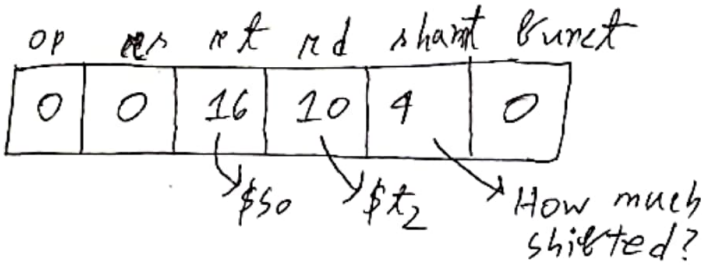
$1001_{two} \longrightarrow 9_{ten}$

two → base 2 binary

ten → base 10 decimal

(Shift left logic)

sll $t2, $s0, 4



(06-11-2024)

① A → [300] + A[400] + h = B[212] [Here, A = $s1, M = $s3, B = $s2]

② Let, a = 12
b = 13

Ⓘ a and b ; Ⅱ b or a ; Ⅲ shift left by 4 on the variable a

~~Solution:~~ ③ P = h + a + b + c
↳ M[53] = M[32] + p + a Let, s_3 = h

Solution: ① A[300] + A[400] + h = B[212]

type		
I	=> lw $t0, 1200($s1)	t_0 = A[300]
I	=> lw $t1, 1600($s1)	t_1 = A[400]
R	=> add $t0, $t0, $t1	t_0 = t_0 + t_1
R	=> add $t0, $t0, $s3	t_0 = t_0 + h
I	=> sw $t0, 848($s2)	t_0 = B[212]

35	17	8	1200		
35	17	9	16		
0	8	9	8	0	32
0	8	19	8	0	32
43	18	8	848		

② Ⓘ Here,

a = 12 → 0000 0000 0000 0000 0000 0000 0000 1100 two
b = 13 → 0000 0000 0000 0000 0000 0000 0000 1101 two
a AND b → 0000 0000 0000 0000 0000 0000 0000 1100 two

Ⅱ b OR a → 0000 0000 0000 0000 0000 0000 0000 1101 two

Ⅲ sll $t4, $t1, 4 [shift left by 4 on variable a]

0000 0000 0000 0000 0000 0000 1101 0000 two

③ Let, a = $s0, b = $s1, c = $s2, h = $s4, P = $s5, M[53] = $s3

P = h + a + b + c

type		
R	=> add $t0, $s4, $s0	t_0 = h + a
R	=> add $t0, $t0, $s1	t_0 = t_0 + b
R	=> add $s5, $t0, $s2	$s5$ = t_0 + c
~~=> add $~~		
~~=> lw $t0~~		
~~=> lw $s5~~		

op	rs	rt	rd	shamt	funct
0	20	16	8	0	32
0	8	17	8	0	32
0	8	18	21	0	32

```
lw  $t0, 128($s3)
add $t0, $t0, $s5
add $t0, $t0, $s0
sw  $t0, 212($s3)
```

$t_0 = M[32]$

$t_0 = t_0 + P$

$t_0 = t_0 + a$

$M[53] = t_0$

35	19	8	128		
0	8	21	8	0	32
0	8	16	8	0	32
43	19	8	212		

(10-11-2024)

(Instructions for making decisions)

if ($p \neq q$)

{ $A = p + q$;

$B = p - q$ }

else { $A = p + q$

$B = A + q$ }

Let,

$A = \$s_0$, $p = \$s_2$

$B = \$s_1$, $q = \$s_3$

=>

```
beq $s2, $s3, Else
add $s0, $s2, $s3
sub $s1, $s2, $s3
j Exit
Else: add $s0, $s3
add $s1, $s0, $s3
Exit:
```

go to if $p \neq q$

$A = p + q$

$B = p - q$

go to exit

$A = p + q$

$B = A + q$

Label : Exit

```
bne $s2, s3, L1
add $s0, $s2, $s3
add $s1, $s0, $s3
j Exit
L1: add $s0, $s2, $s3
    sub $s1, $s2, $s3
Exit:
```

goto else

$A = p + q$

$B = A + q$

Jump to exit

$A = p + q$

$B = p - q$

Label : L1

(17-11-2024)

Lecture-6 | What is the decimal value of this 32 bit two's complement number?

1101 1101 0011 1111 1011 0111 0000 1001

Ans:

$\Rightarrow -2^{31} + (2^{30} + 2^{28} + 2^{27} + 2^{26} + 2^{24} + 2^{21} + 2^{20} + 2^{19} + 2^{18} + 2^{17} + 2^{16} + 2^{15} + 2^{13} + 2^{12} + 2^{10} + 2^{9} + 2^{8} + 2^{3} + 2^{0})$

$= -7172,4963$

or,

$\Rightarrow -2^{31} + (\text{If all value is 1})$

$= -2,147,483,648 + (2,147,483,647 - 2^{1} - 2^{2} - 2^{4} - 2^{5} - 2^{6} - 2^{7} - 2^{11} - 2^{14} - 2^{22} - 2^{23} - 2^{25} - 2^{29})$

$= -1 - 2^{1} - 2^{2} - 2^{4} - 2^{5} - 2^{6} - 2^{7} - 2^{11} - 2^{14} + 2^{22} - 2^{23} - 2^{25} - 2^{29}$

$= -7172,4963$

Example: If first value is 0,

0111 1111 1111 ....

$= 2,147,483,647 - 2^{0} - 2^{1} \ldots$

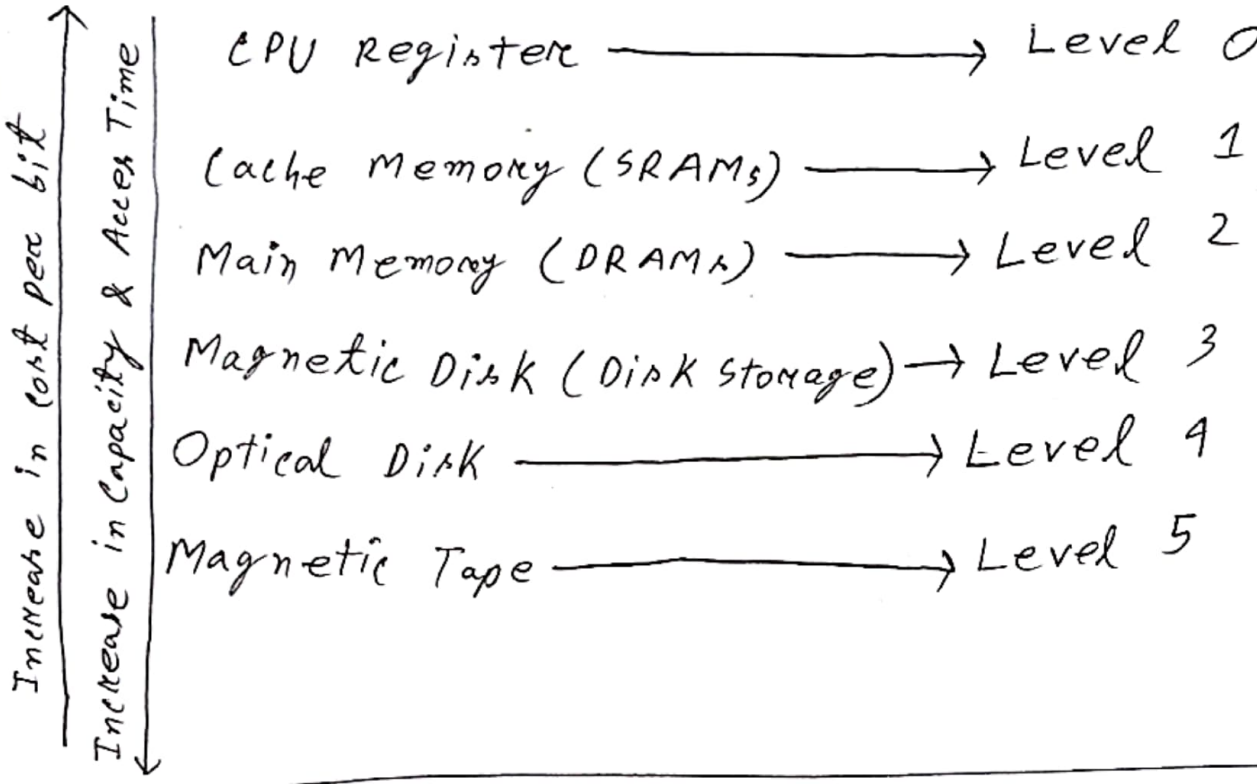
Lecture-9 | 1) Memory Hierarchy

A memory hierarchy consists of multiple levels of memory with different speeds and sizes. The faster memories are more expensive per bit than the slower memories and thus smaller.

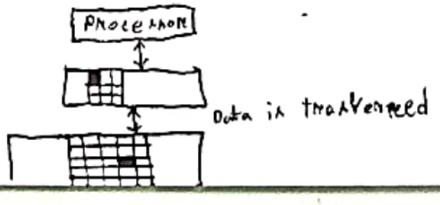
Three techs in memory hierarchies:

(1) DRAM (Main memory) (2) SRAM (Cache) (3) Magnetic disks (Hard disk).

2) Memory Hierarchy Design:



3) Data Hierarchy: The data hierarchy is a layered structure where data is organized from the fastest and smallest storage level (registers) to the slowest and largest (hard disks). Data is copied between adjacent level in blocks to optimize performance. Higher levels are faster but smaller and more expensive. While lower levels are slower but larger and cheaper.



(20-11-2024)

1) Hit: When the requested data is found in the upper memory level. Example: Cache.

2) Miss: When the requested data is not found, requiring access to a lower memory level.

3) Hit Rate: The percentage of memory accesses found in the upper level.

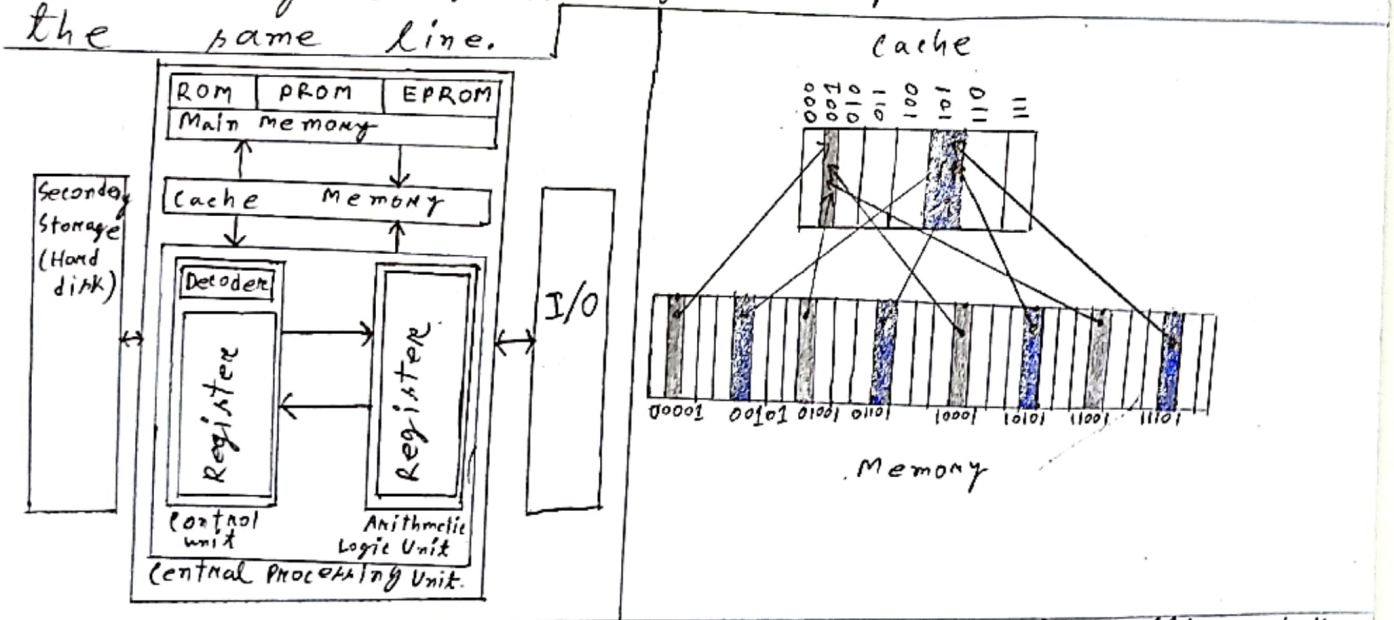
4) Miss Rate: The percentage of memory accesses not found, calculated as (1 - Hit Rate).

5) Hit Time: Time taken to access the upper memory level, including checking for a hit or miss.

6) Miss Penalty: Time required to fetch data from the lower memory level, replace a block in the upper level and deliver it to the processor.

7) Cache: Is fast, small memory near the CPU that stores frequently used data to reduce access time and improve performance. It has levels (L1, L2, L3) for efficiency.

8) Direct Mapped Cache: It maps each memory block to a specific cache line, offering simplicity but risking conflicts if multiple blocks map to the same line.



9) Tag: Is a part of a memory address in caching that identifies whether a specific block of data is stored in the cache. It helps match cache entries with requested memory addresses during a lookup.

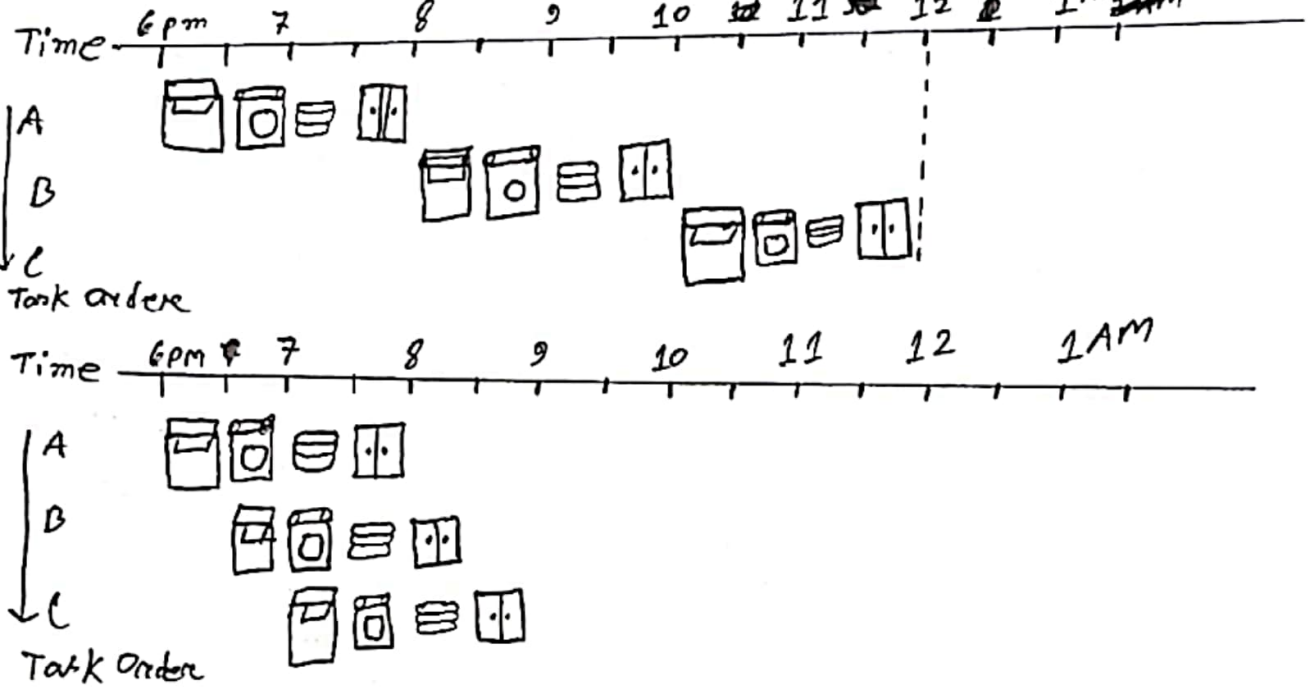
(27-11-2024)

Lecture-10 | Pipelining And Hazards

Pipelining: Is an implementation technique in which multiple instructions are overlapped in execution. It is used to make the processor fast.

The laundry analogy for pipelining:

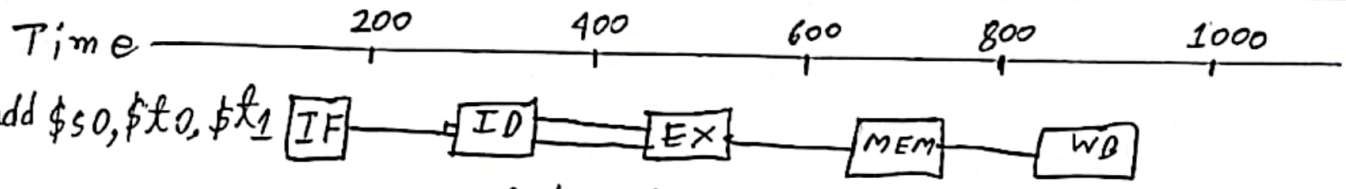
Sequential 60 hours, while pipelined laundr takes 3 hours



MIPS Pipeline: Classically take five steps:

1) Fetch instruction from memory.
2) Read register while decoding the instruction. The regular format of MIPS instructions allows reading and decoding to occur simultaneously.
3) Execute the operation or calculate and address.
4) Access an operand in data memory.
5) Write the result into a register.

Graphical representation of the instruction pipeline:



IF: Instruction Fetch stage, with the block representing Instruction Memory.

ID: Instruction decode/register file read stage, with the drawing showing the register file being read.

EX: Execution stage; with the drawing representing ALU.

MEM: Memory access stage, with the box representing data memory.

WB: Write-Back stage, with the drawing the register file being written.

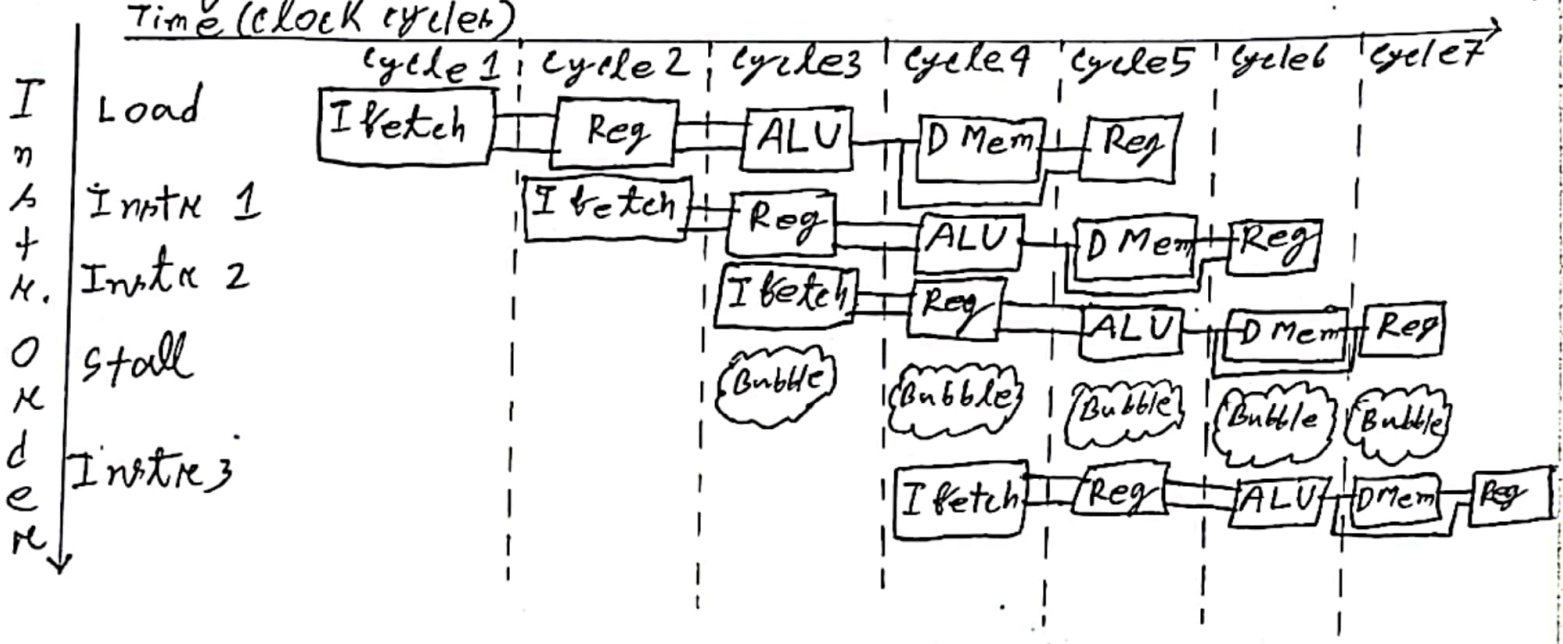
Hazards: There are situations in pipelining when the next instruction can't execute in the following clock cycle. These events are called hazards.

Three types of hazards:

(1) Structural Hazards (2) Data Hazards (3) Control Hazards

1) Structural Hazards: A structural hazards occurs when a part of the processor's hardware is needed by two or more instructions at the same time. ~~A~~ The can be resolved by separating the component into orthogonal units (such as separate cache) or bubbling the pipeline.

Resolving Structural Hazard: To resolve this, we stall the pipeline for one clock cycle when a data-memory access occurs. The effect of the stall is actually to occupy the resources for that instruction slot.



Data Hazards: Data hazards occur when the pipeline must be stalled because one step must wait for another to complete.

		1	2	3	4	5	6	7	8	9	
ADD	R1, R2, R3	IF	ID	EX	MEM	WB					
SUB	R4, R5, R1		IF	ID_{sub}	EX	MEM	WB				
AND	R6, R1, R7			IF	ID_{and}	EX	MEM	WB			
OR	R8, R1, R9				IF	ID_{or}	EX	MEM	WB		
XOR	R10, R1, R11					IF	ID_{XOR}	EX	MEM	WB	