$\boxed{1GHz = 10^9 Hz}$

* Indexing and Hashing | * Transactions

—

CO lecture

21/08/23
Monday

for every instruction co
is different, so we co
the avg.

* Processor performance eq
→ Execution time $= N \times CPI \times t$

$t = \frac{1}{F}$ → Freq

N = total no. of instruction

CPI = cycles per instruction

Performace factors
① Information count (IC) or (N)
② Cycles per instruction (CPI).
③ memory reference per instruction.
access
④ Memory latency per instruction.
⑤ time per unit cycle.

Systemattributes:
* Instruction set architecture [ISA]. → affects ①, ②
Processor Implementation Control → affects ②
cache $\xrightarrow{\text{affects}}$ ④, ⑤
compiler technology. → affects ①

throughput → ~~Most~~ no. of transactions done per unit time.
in
eg. banking

Responsetime, Relative performance.

$$\text{performance} = \frac{1}{\text{Execution time}} \quad - \left[\text{Performance} \propto \frac{1}{\text{Execution time}}\right]$$

→ so if performance of x < performance of y

then Execution time of x > execution time of x.

Q.1] $\boxed{\begin{array}{l}\text{Machine A} \to 10\text{ sec} \\ \text{Machine B} \to 15\text{ sec}\end{array}}$ (execution time) For same instruction set

→ Relative performance of A wrt B $= \frac{15}{10} = 1.5$ so machine A is 1.5 times faster that

Relative performance of ~~B wrt A~~ $= \frac{\cancel{10}\,2}{\cancel{15}\,3} =$

Q.2] Computer A → 2GHz clock → 10 sec CPU time.
Design computer B → aim 6 sec CPU time
and it can do faster clock but causes 1.2 times clock cycles. How much faster computer B's clock should be?

$$\text{CPU time} = \frac{\text{clock cycles}}{\text{clock rate}} \quad - ①$$

$$10 = \frac{\text{clock cycles}}{2 \times 10^9 \text{ Hz}} \quad \therefore \text{clock cycles of A} = 2 \times 10^{10} \quad - ②$$

$$\text{cpu time of B} = \frac{1.2 \times 2 \times 10^{10}}{\text{clock rate}}$$

$$\therefore \text{clock rate} = \frac{\overset{0.2}{\cancel{1.2}} \times 2 \times 10^{10}}{\cancel{6}_{1}} = 4 \times 10^{9}$$

$$\therefore \text{clock rate of B} = 4\text{GHz}$$

* Calculation of avg. CPI

→ $$\text{clock cycles} = \sum_{i=1}^{n} \left(\text{CPI of } i^{th} \text{ instruction} \times \text{IC} \right)$$

$$\text{CPI} = \frac{\text{clock cycles}}{\text{Instruction count}} = \sum_{i=1}^{n} \left(\text{CPI of } i \text{ instruction} \times \frac{\text{IC of } i}{\text{Instructuction count}} \right)$$

Q.1].

	A	B	C
CPI	1	2	3
IC in seq1	2	1	2
IC in seq2	4	1	1

Calculate avg CPI for both the seque

→ clock/cycles =

avg CPI for seq. 1 $= \dfrac{(1\times2 + 2\times1 + 3\times2)}{2+1+2}$

$= \dfrac{10}{5} = 2.$

avg. CPI for seq. 2 $= \dfrac{(1\times4 + 2\times1 + 3\times1)}{4+1+1}$

$= \dfrac{\cancel{9}\,3}{\cancel{6}\,2} = 1.5.$

Q.7 A 400 MHz processor was used for execute a program with following IC mix.

Instruction ~~time~~ count	clock cycle count
4,50,000	1
3,20,000	2
50,000	2
80,000	2

$$CPI = \frac{(4,50,000 + 6,40,000 + 3,00,000 + 1,60,000)}{4,50,000 + 3,29000 + 1,50,000 + 80,000}$$

* MIPS rate = $\dfrac{IC}{t \times 10^6} \cancel{\times 10^6} = \dfrac{f}{CPI \times 10^6} \times$ [illegible]

$$= \frac{f \times IC}{C \times 10^6}$$

Total no. of clock cycles required to executed the instruction

The MIPS rate for the problem on pg. ⑤ is [~~downside~~] (located ~~below~~ at bottom) $= \dfrac{400 \times \cancel{10^6} \times \text{[illegible]}}{1.55 \times \cancel{10^6}} = 258.06.$

Execution time $= (\cancel{10 \times 10^5})$

$$\frac{1.55 \times 10 \times 10^5}{400 \times 10^6}.$$

Q.1]. Consider the execution with 2×10^6 instructions of an code on 400MHz CPU. Program has 4 types of instruction.

Instruction type	CPI	No. of
Arithmetic	1	60%
Load and Store	2	18%
Branch instruction	4	12%
Memory reference	8	10%

Calculate avg. CPI, calculate also MIPS.

—

21/08/23
Monday

Robotics Lecture

* Robot → Reprogrammable Multifunction manipulator which can perform ~~the specified task~~, variety of tasks
rough programmed motions.

*

$$\begin{tikzcd} & \text{Robots} \arrow[dl] \arrow[dr] & \\ \text{Fixed} & & \text{Moving} \end{tikzcd}$$

Sugar

29|09|23 DBMS Lecture

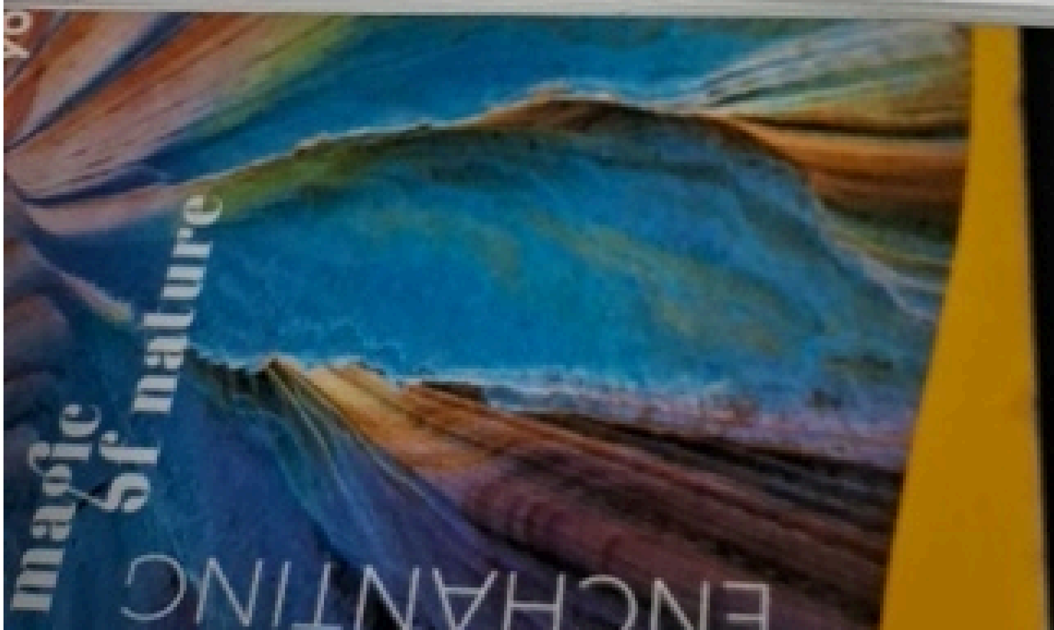
* XML: Extensible Markup Language

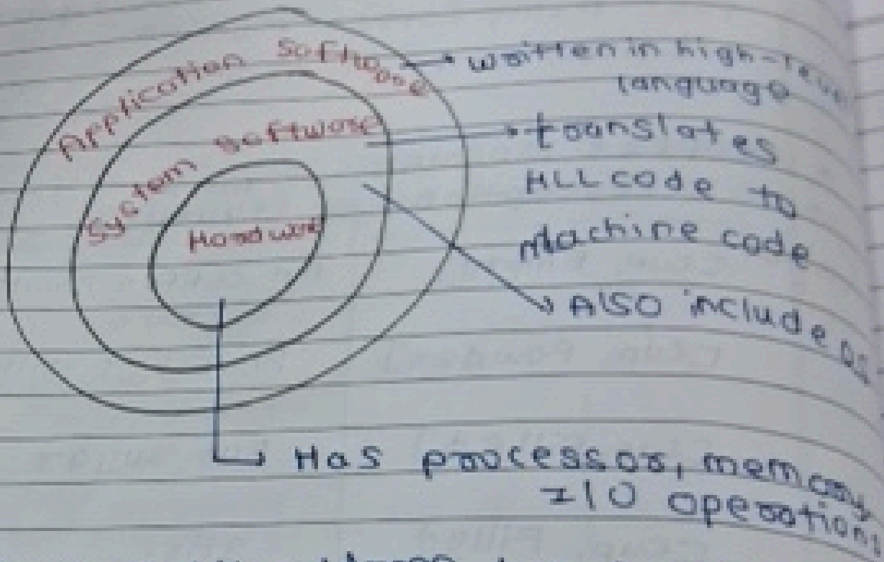
9|09|23
uesday

CO Lecture

7. SISD → single Instruction single Datastream.

parallelism exist in software as well as in hardwa




Application Software
System Software
Hardware
→ written in high-level language
→ translates HLL code to machine code
→ Also include OS
→ Has processor, memory, I/O operations

X86-64 → means 64-bit address based on 8086 instruction set.

4 CPU/cores → 8 $\overset{\text{hardware}}{\text{threads}}$

– [1 core has two hardware threads]

Cache → L1 [Size in KB] → private
L2 [Size in KB but more than L1]
L3 [Size in MB] → Shared.

→

registers → Limited in no.
↳ holds limited data

Speed wise:

Register > Cache
memory   memory

—

20/02/23
wednesday

CN Lecture

* OSI model

OSI model	TCP/IP
Application	Application
Presentation	Transport layer
Session	Network layer
Transport	Data link layer
Network	Physical layer
Data Link	
Physical	

* CAT6 wire → 100 m distance coverag
Multimode Optic → 400 m —II——
wire

Single mode optic ——→ >400m —II—
wire

Q.5] Events are independent.

—

01/09/23

CO Lecture

* Intruction set → 64-bit
  └ Types: x86, ARM, MIPS, RISC-V ← open architecture
  x86 ↓ in our laptop, computer
  MIPS └ in embedded systems

RISC-V → Has a 32×64-bit register file.
[32 bit data is called word].
[64 bit data is called double word]

* RISC-V registers
  X_0 - X_{17} register.

$\boxed{\text{Stored program concept}}$

Little Endian	Big Endian
↓	most ↓
Least significant bit at least address	significant bit at least address.

rage classes → auto, static, extern, register.

X86 → variable Instruction size
RISC-V → Fixed instruction size
32 bit instruction

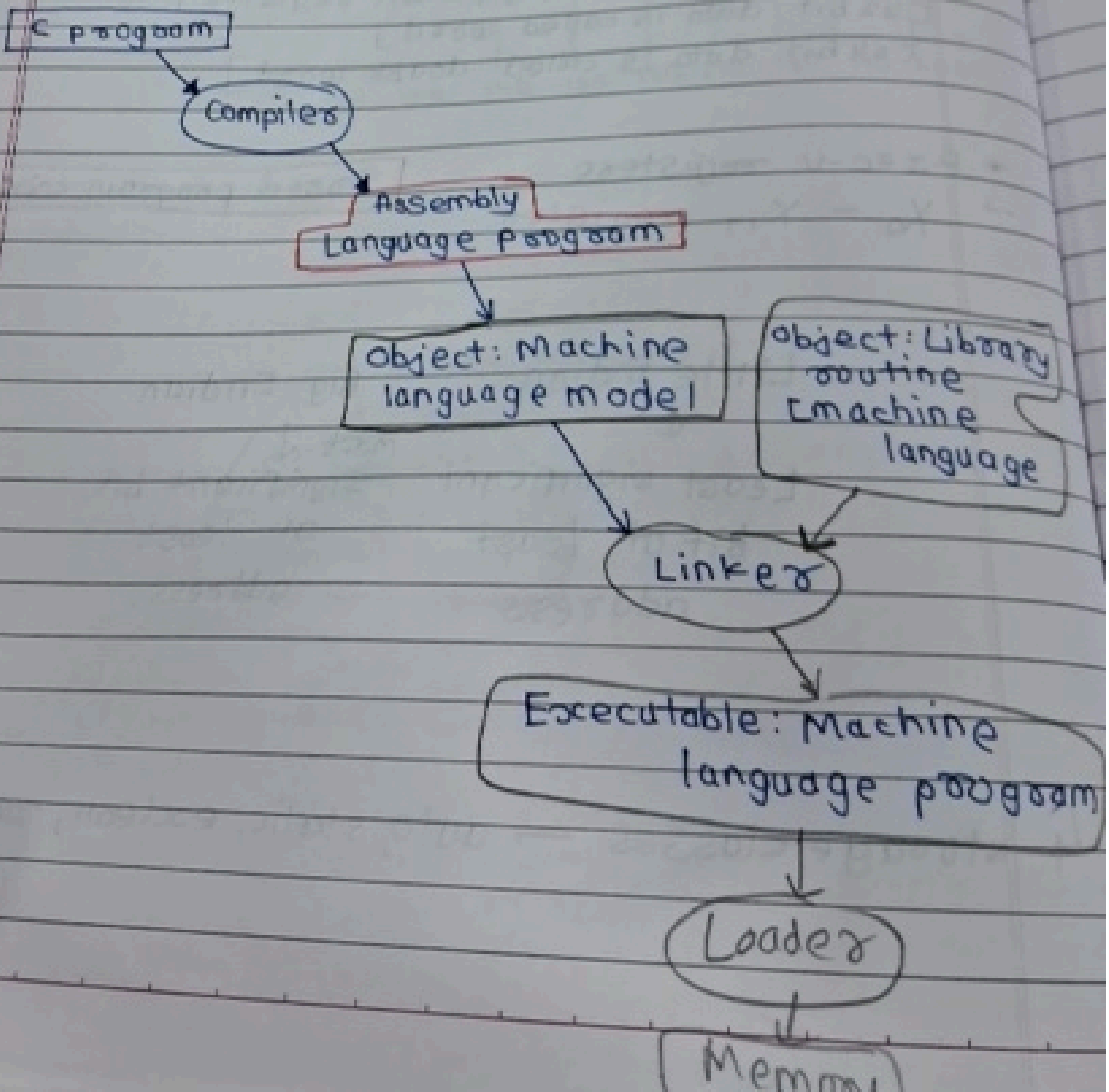
* RISC-V Instruction Format
→

funct 7	rs2	rs1	Funct3	rd	opcode
7bits	5bits	5bits	3bits	5bits	7bits

opcode: operation code
rd: destination register no.

⊛ Binary compatability

* Translation and setup
→



C program
Compiler
Assembly Language Program
Object: Machine language model
object: Library routine [machine language
Linker
Executable: Machine language program
Loader
Memory

R1.....R17 → registers

e.g. Z = K + B * C

let K → stored at L1
B → stored at L2
C → stored at L3
Z → stored at L4

→ How to write | execute it in 3-instruction set and 2-instruction set.

→ a]. 3-instruction set
→ MOV Z, MUL (B, C)
ADD Z, K.

b]. 2-instruction set
→ MOV Z, C
MUL Z, B
ADD Z, K.

c]. 1- instruction $\overset{set}{\rightarrow}$ wrt accumulator
→ Load C
M~~A~~UL B
ADD K
STORE Z.

d]. 0-instrunction → wrt stack
→ postfix

Z = K BC * +

PUSH K
PUSH B
PUSH C
POP B, C
PUSH B * C
POP B * C, K

Foreign key → Refrencing and Refrenced tables
~~constraints~~

IMP → For 1 to many relation the many table's pk is the pk of the relation.

—

04/09/23
Monday

CO Lecture

$$\begin{tikzcd} & \text{Primary memory} \\ \text{* Memory unit} \arrow[ur] \arrow[dr] \arrow[d] & \\ \text{Store progams and data} & \text{Secondary memory} \end{tikzcd}$$

5 stage pipeline:

$$\begin{tikzcd} & & & \text{Memory} & \\ \text{IF} \arrow[d] & \text{ID} \arrow[d] & \text{IE} \arrow[dr] & \text{M} \arrow[u] & \text{WB} \arrow[r] & \text{write/read block} \\ \text{Instruction fetch} & \text{Instruction decode} & & \text{Instruction execute} & \end{tikzcd}$$

* Control unit

↳ control all computer operations

It generates certain timing signals for carrying the operations.

William
~~Sta~~ Stalling

operations of computer:
① Accept info in form of programs and required data through I/O transfer.
[Incomplete].

* 32bit word

b31	b30	....	b1	b0

* Little Endian Artitecture	Big Endian Architecture
↓	↓
lower byte address/point to MSB	Lower byted addsenss/ point to LSB.

—

04/09/23 * PSE Lecture *

Q.1] A box contains two coins: one is Fair and other has heads on both side. one coin is choosen at random and Flip twice; both times it shows head. what is the probablity than the Fair coin was choosen?

→

vent A : Getting heads twice in a row.
vent B : Chosing fair coin
vent C : Choosing unFair coin.

→

11/09/23

CO Lecture

* William Stalling 10th Edition → 20th chapter
  MS Zakir → 5th chapter. + 14th chapter

* Instruction Fetch | Instruction Execute

* Micro-operations
  ↳ functional, or atomic, operations of a processor.

  ↓
  It refers that each step is very simple and accomplishes very little.

* Datapath in a processor.
  1. Instruction Fetch

$$\begin{tikzcd}
\text{Instruction Fetch} \arrow[d] \\
\text{Instruction Decode} \arrow[d] \\
\text{Instruction execute} \arrow[d] \\
\text{Memory} \arrow[d] \\
\text{Write back.}
\end{tikzcd}$$

This consits of the datapath.

* Inorder and out-of-order
  - Inorder ↓ execute the program in the way instructions are written
  - out-of-order ↓ will execute subpart of program and then combine the result.

ADD R7, R3, R5
↓
Here the contents of all three registers is added and stored in R7.

ADD R7, R5, X(R3)
↓
Here contents of R7, R5 and memory location pointed by (R3) are added and stored in R7.

ADD R1, 10, 20
↳ Here the ~~contents~~ of 10+20=30 + contents of R
is stored in R1.

* Fetch cycle
→ consist of 4 main registers:
a) Memory address register
   ↳ connected to address bus.
   ↳ specifies address for read or write operation.

b) Memory Buffer register
   ↳ connected to data bus
   ↳ Holds data to write or last read data.

[2 more are their]

* the contents of PC and IR does not change untill the execution of instruction is complete.

ADD R7, R3, R5
↓
Here the contents of all three registers is added and stored in R7.

ADD R7, R5, X(R3)
↓
Here contents of R7, R5 and memory location pointed by (R3) are added and stored in R7.

ADD R1, 10, 20
↳ Here the ~~contents~~ 10+20=30 + contents of R of is stored in R1.

* Fetch cycle
→ consist of 4 main registers:

a] Memory address Register
↳ connected to address bus.
↳ specifies address for read or write operation.

b] Memory Buffer Register
↳ connected to data bus
↳ Holds data to write or last read data.

[2 more are their]

* the contents of PC and IR does not change until the execution of instruction is complete.

)
R7,
ocation
are
sed in

ER

* Rules for Micro-operations grouping
→ 1. Proper sequence must be followed.
MAR = (PC) must procede MBR = (memory).
2. conflicts must be avoided.

* Indirect cycle * Interrupt cycle * Execute cycle
* Flow chart for Instruction cycle

* Control unit Functional Requirements
TWO tasks
↙ Sequencing ↘ Execution.

three step process to lead
↳ 1. Define basic

* one Imp. table found on page 719 in the textbook

—

11/09/23
Monday

COI - Lecture

* Rule of Law → connected to the preamble
"[illegible]trine"

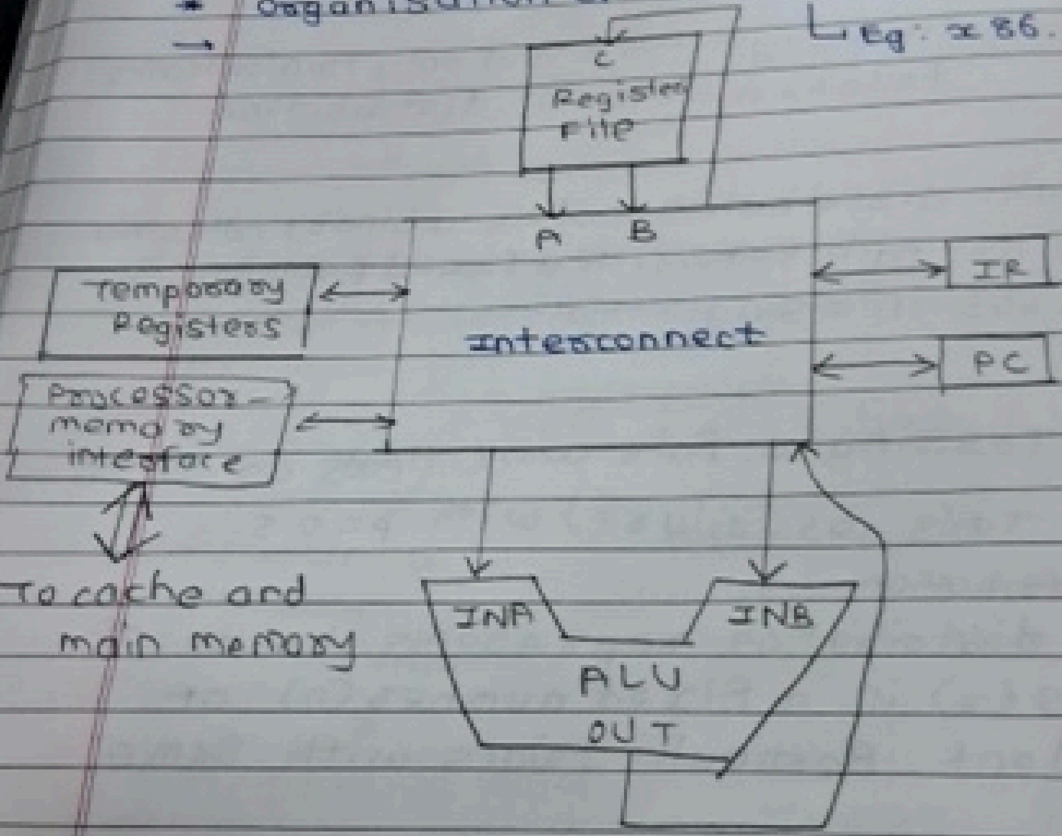
Human Rights v/s Fundamental Rights

15/09/23
Friday

CO - Lecture

RISC v/s CISC

* Organisation of CISC-style processor
→ └ Eg: x86.



* RISC v/s CISC → complex and difficult implementation of instru

comparatively simple instruction implementation

3 style instruction Format

2 style instruction Format.

Eg: ADD R7, R5, R9

Eg: AND X(R5), R2

↓

Here only we can do operations using registers

↓

Here we can d operations usin memory as well registers

ISC]

sor

Hardware control ↑

Micro-program control ↑

* RISC block diagram v/s CISC block diagram

* Execution of instruction in CISC
- Eg: AND X(R7), R9

$\downarrow$

means memory location whose address stored in R7.

Step 1 Memory address $\leftarrow$ [PC], Read memory, wait for memory function to complete [MFC]

IR $\leftarrow$ Memory data.

PC $\leftarrow$ [PC] + 4.

Step 2 Decode Instruction.

Step 3 Memory address $\leftarrow$ [PC], Read memory, wait for MFC

Temp1 $\leftarrow$ Memory data.

PC $\leftarrow$ [PC] + 4.

Step 4 Temp 2 $\leftarrow$ [Temp1] + [R7]

Step 5 Memory address $\leftarrow$ [Temp2], Read memory wait for MFC.

Temp1 $\leftarrow$ Memory data.

Step 6 Temp1 ← [Temp1] AND [R2]

Step 7 Memory address ← [Temp 2],
Memory data ← [Temp1]
write memory
wait for MFC.

* Microprogrammed control
→ we know for each step execution we need control signals.

Inhardwired control [RISC]	Micro-program control
↓	↓
these signals are generated by some type of circuits.	these signals are generated by software.

* Micro-instruction → Horizontal micro-instruction
→ Vertical micro-instruction

L1, L2 cache → private
L3 cache → shared cache

* Perf Assignment

→ L1, L2, L3 cache Misc cycles, word size	block size for our laptop

No. of cycles
Execution time }
instructions

—

13/09/22
Friday

AI - Lect

* Crypthârithmethic prob

→

$$\begin{array}{r} \text{TWO} \\ +\ \text{TWO} \\ \hline \text{FOUR} \end{array}$$