Computer Organisation

Prof Ram Sarkar

01/08/2019

Books :- Computer Organisation & Architecture → Wilson & Stallings

William-Stalling - Computer Org.

Architecture v/s organisation

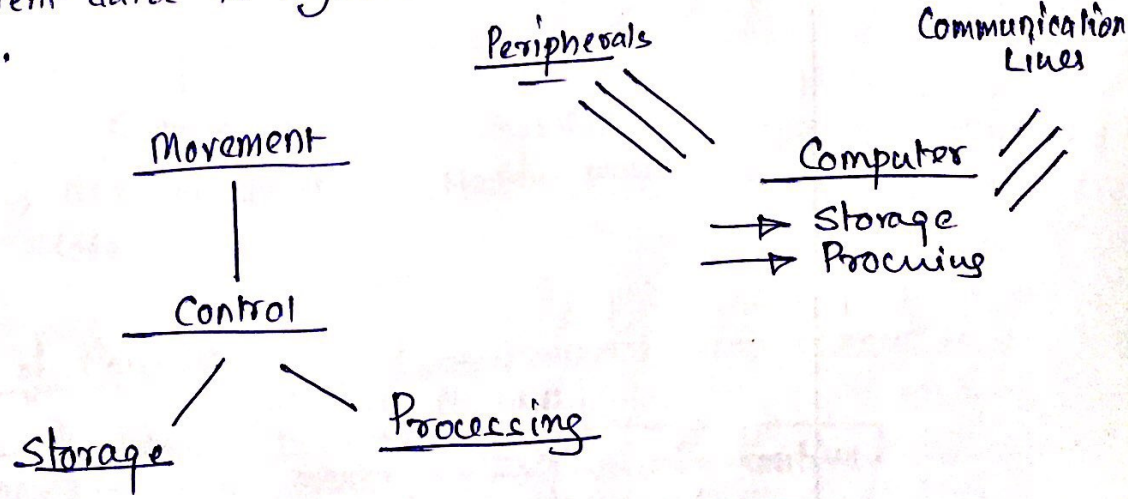
- Architecture → Directly available to the programmer.
- organisation → Inter-connection b/w various units

- **IBM - 370 Architecture**
- **Hierarchy** → Memory → Secondary Memory, Primary Memory, Cache, etc.

  for eg. C → Procedural programming
  1. Seperated into blocks.
  2. Reuse.
  3. Error detection becomes easy.

- **Structure & Function**

  Consistent data is highest priority.



02/08/2019 Computer Organisation

Prof. Ram Sarkar

Compiler

→ Acts as a translator transforms HLL into Machine oriented language.

→ Helps the programmer ignore machine dependent details.

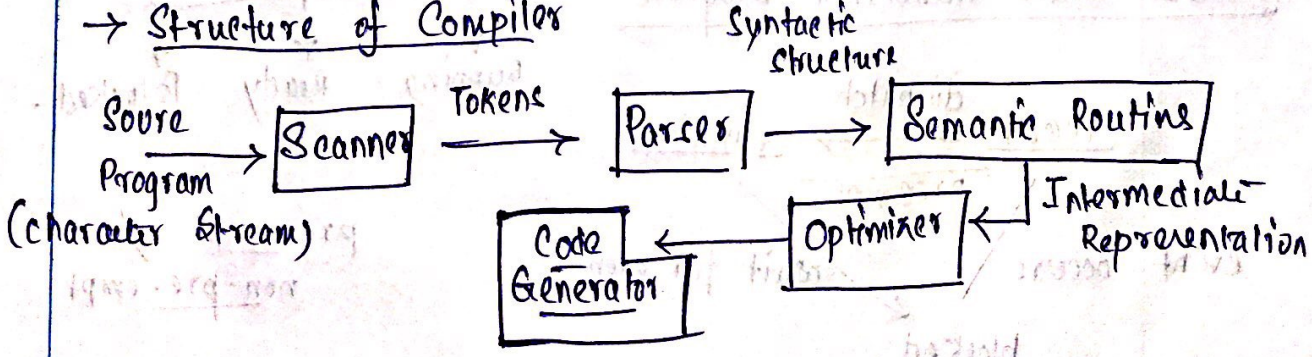
Source → Compiler → Machine Language (Target).

Compiler →
- Pure Machine Code → OS routine
- Augmented Machine Code (C/C++) (Library).
- V.M. code (Python, Java).

Compiler →
- Assembly Level Language
- Relocatable binary (Relative Address).
- Absolute binary. (Absolute → Fixed address).

Compiler
- Analysis → Analyses source code
- Synthesis → Machine code.

→ Structure of Compiler



→ Compiler generats a symbol table.

(Refer to the example).

Operating System

Main Goals
- → Resource management
- → Resource Abstraction
- → virtualization.

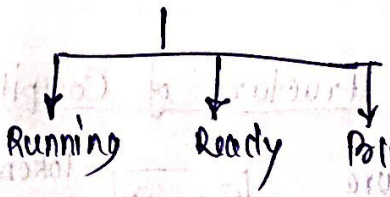
Execution Mode
- → user Mode
- → Kernel (or supervisor) Mode

Process ⟶ Instance of a progam in execution
- → Process image
- → Static & State instr.
- → Process switch.

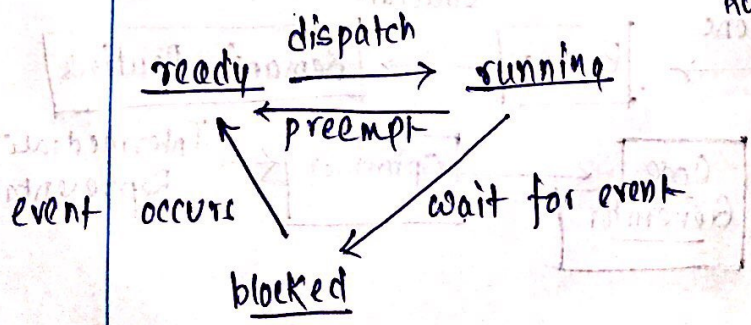
↓ encompasses static concept of program & dynamic aspect of execution

Threads
- → Division of proces.

Process Execution



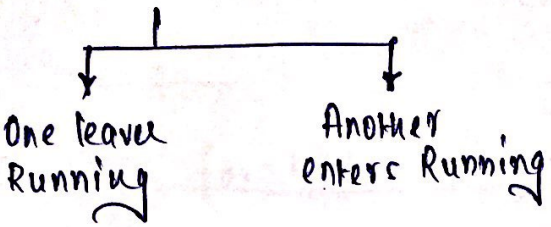
Process State Transition Diagram



pre-empt v/s non-pre-empt

Other state ⟶ New → Exit

Context switch



Concurrent Process

Uniprocess → Time sharing using Multiprogramming
Multiprocess → True Parallel execution

Multiprogramming → Round Robin Scheduling

Scheduling

→ Process (Thread) v/s Preemptive

→ FCFS & RRS.

Goals

→ Avoid starvation (be "fair").
→ Optimize throughput

Deadlock

→ When two dependent process are in Blocked state.

Deadlock v/s Starvation

8/8/2019 Computer organisation Prof. Ram Sarkar

①

→ Von Neumann Architecture

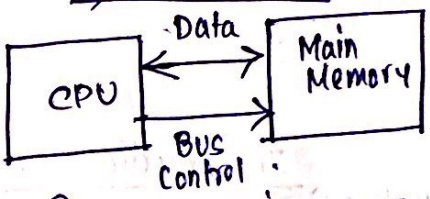
→ Instruction Cycle
  → Basic
  → Exception

→ Instruction Architecture
  → Software design
  → Hardware circuit

→ Digital Design

Must follow an instruction architecture
→ NOP no. operation
→ TSTST test & set

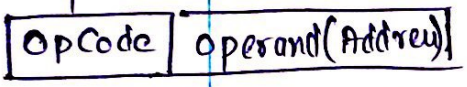
① Data & instruction are both stored in main memory.
② content is addressable by location



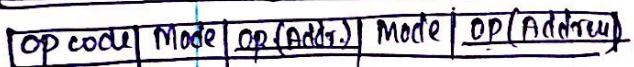
③ Instructions are executed sequentially under order is explicitly modified.

④

eg. ADD

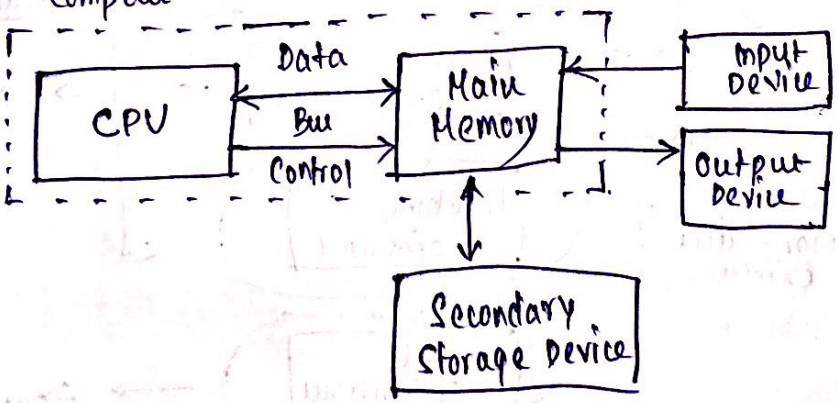


→ Different addressing mode.



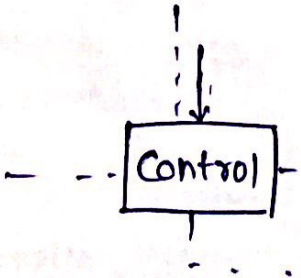
Basic Architecture

Computer System



⑤ component

1. CPU
2. Main Memory
3. I/O Devices
4. Mass Storage
5. Interconnection Network

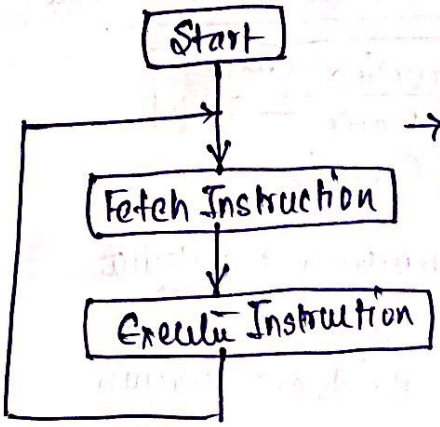


Control Unit
connects every other component of the computer.

Instruction Cycle

Basic

All instructions are stored in memory



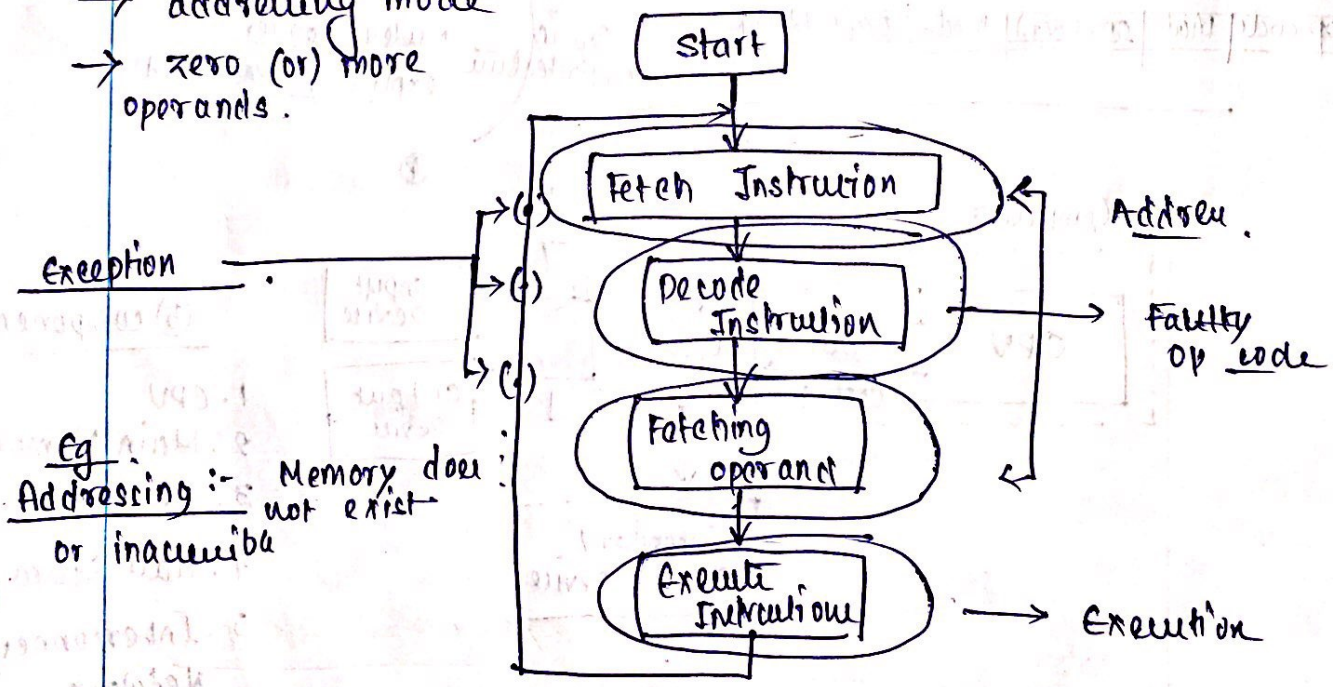
~~All instructions~~

→ All instructions are stored in memory

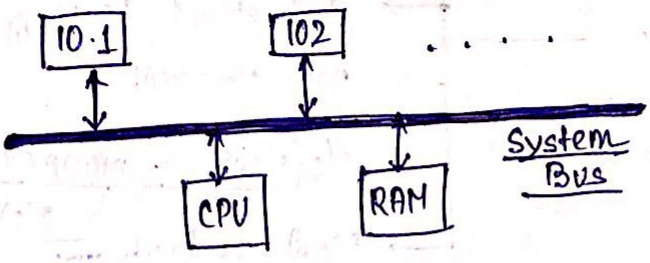
Intermediate view

Complete instruction

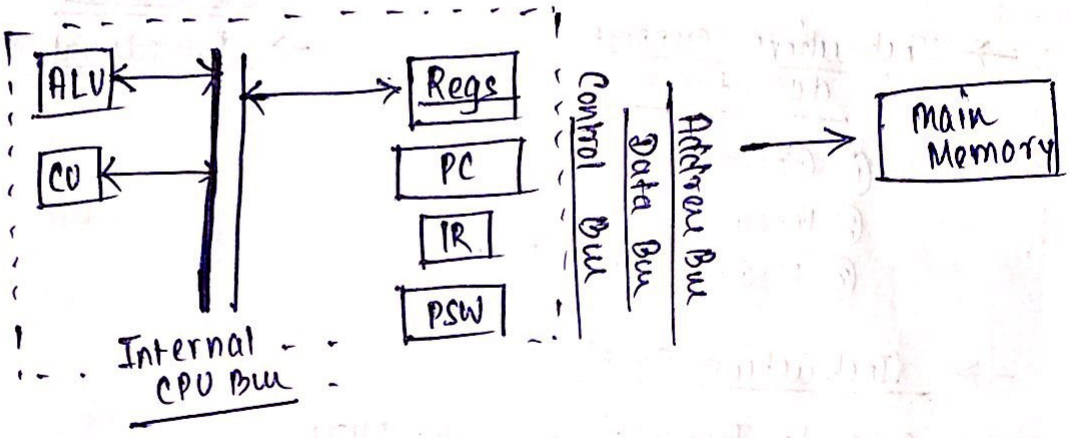
→ operation code
→ addressing mode
→ zero (or) more operands.



Eg:
Addressing :- Memory does not exist or inaccessible



PC → Program Counter
PSW → Program Status Word
IR → Instruction Register



At the hardware level, they are implemented using XNOR, OR, XOR, NOT, ... etc.

Data → signals → Combination of gates → Integrated Circuits (ICs).

→ LSI, VLSI, ULSI.

(Design complexity)

→ Control Unit

① Manages the data stored in register & accumulator stages, must decode instructions, communication with RAM addresses.

② Each operation requires seperate circuitry

③ Necessary to have knowledge of various data representations.

ALU

→ CISC (Complex instruction set)
→ RISC (Reduced instruction set)

Instruction Set Design

- Instruction Type
  - Arithmetic & Logic (4)
  - Data Movement
  - I/O (Memory-mapped, isolated I/O).
  - Flow control
    - Branches
    - Procedural calls
- No. of addresses
- Addressing modes.

- Instructions support data-types
  1. Characters
  2. Integers
  3. Floating Point type

- Instruction Overload
- Seperate Instruction eg. MIPS

Number of Addresses

- ~~3~~ 3-address machine
- 2 address machine
- 1 address machine
- 0 address machine.

eg. $Y = \frac{A - B}{C + (D \times E)}$.

(eg.) 3-address

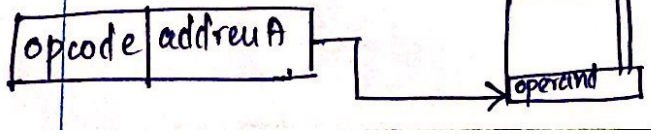
```
SUB Y, A, B.
MUL T, D, E
ADD T, T, C
DIV Y, Y, T
```

eg. ① IA32 (2-address)
② IA32's MUL (EAX) (1-address)
③ IA32's FPU, HP3000

Addressing Modes

- Operands placed in
  - Registers
  - Part of instruction
  - Memory
- Common addressing modes.
- Implied addressing
- Immediate addressing

- Direct addressing



09/8/2019 Computer Organisation

Prof. Ram Sarkar

Addressing Modes

→ Place where operand is stored.

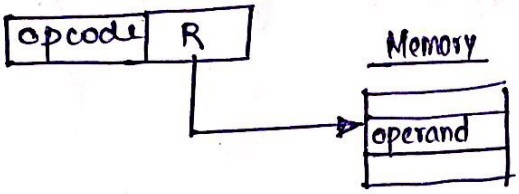
CLC; ← implied addressing

ADD 5; ← immediate addressing

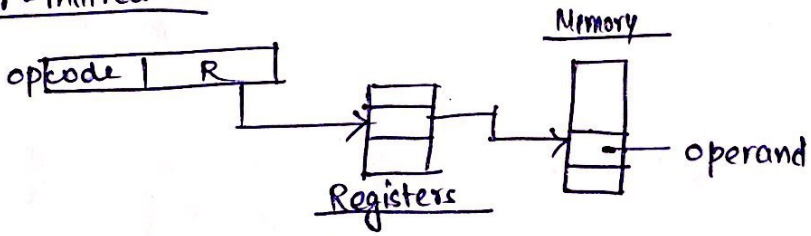
→ direct addressing, indirect addressing.

→ register addressing

→ Register - direct



→ Register - indirect



→ Displacement addressing

→ Stack addressing

implicit →

First In Last Out Order / Last In First out

EA → effective address

William Stallings → Chapter - 2

→ IAS computer. (Von Neumann Architecture)

→ Structure of IAS computer

→ IAS instruction set

Registers

→ MBR (Memory Buffer Register)

→ IBR (Instruction Buffer Register)

→ MAR (Memory address register).

→ Program Counter (PC).

→ Instruction Register (IR) ← Contains the opcode

→ Accumulator & Multiplier Quotient (MQ)

M(X) → memory contents at X.

Conditional branch → Left instruction
→ Right instruction

IAS Operations

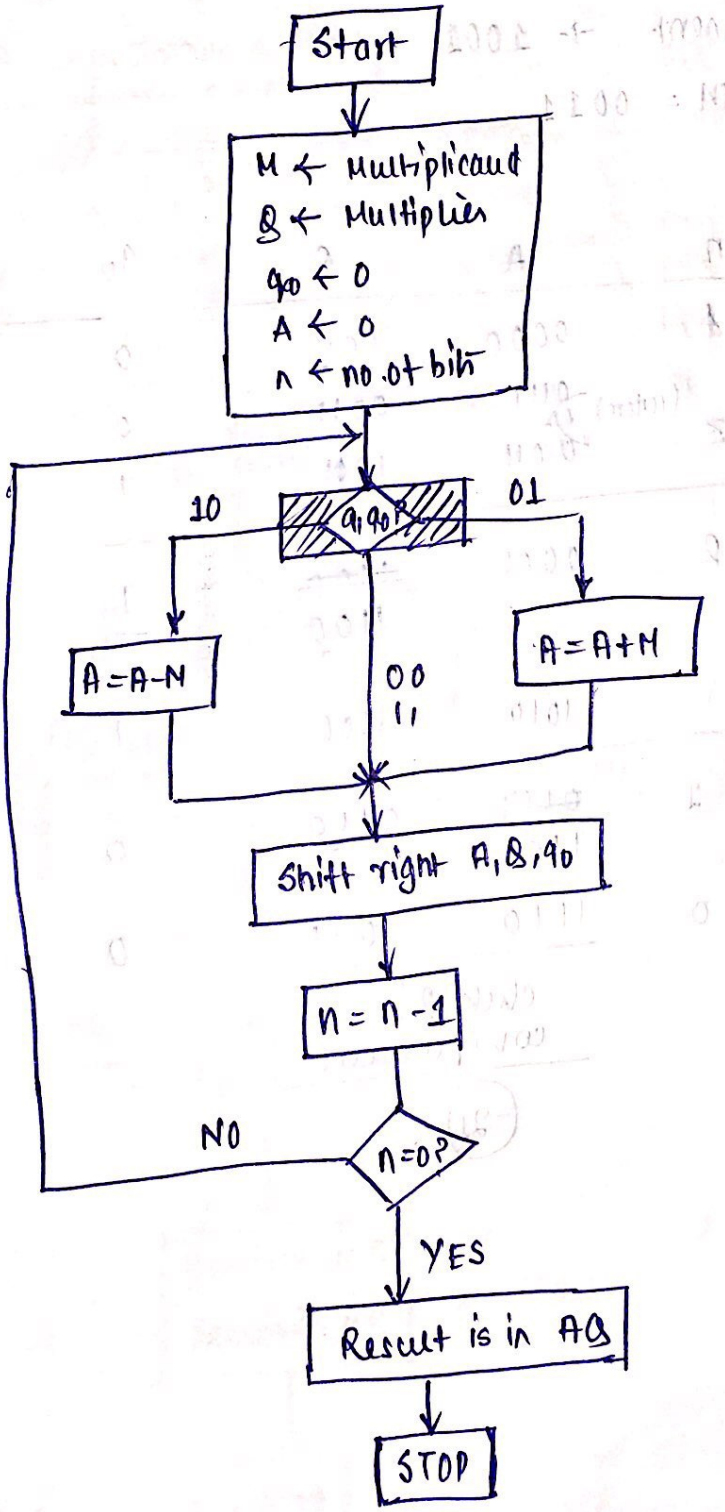
→ Problems with Clock Speed & Login Density.
- → Power
- → RC Delay
- → Memory Latency

27/8/2019

Computer Organisation

Post. Ram Sarkar

Booth's Algorithm



Example

$(-7) \times (3)$

M Q

2s complement → 1001

M = 0011.

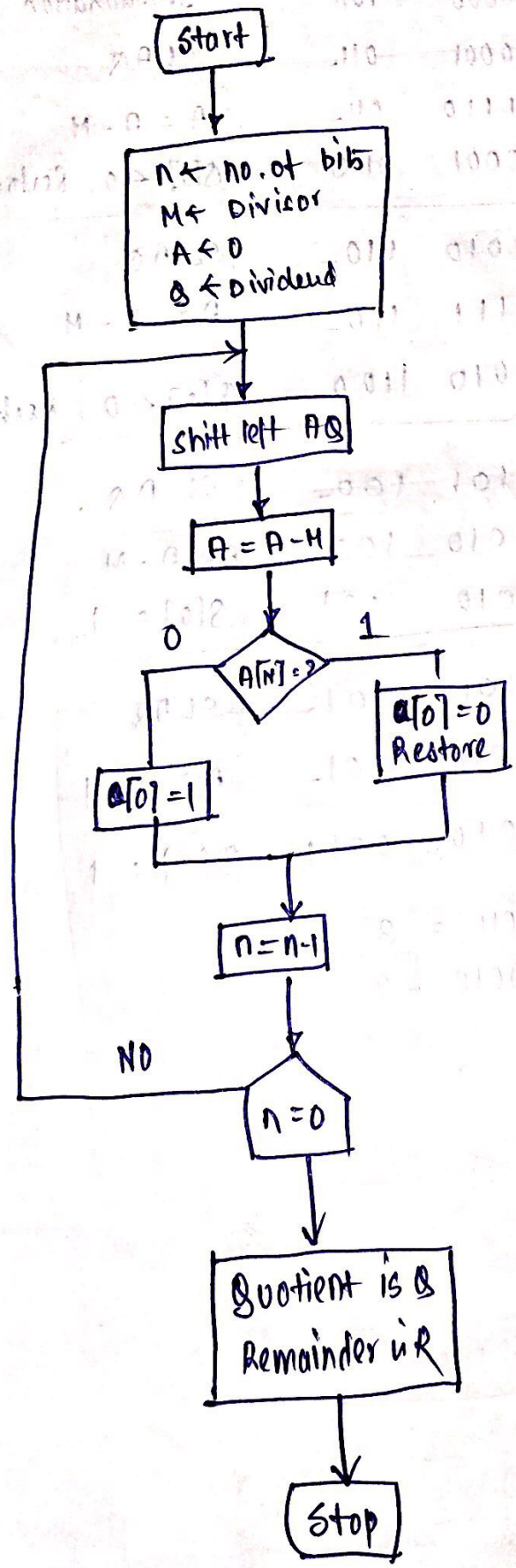
n	A	Q	q_0	Action
4	0000	0011	0	Initialization
	0111	0011	0	A = A − M
3	(restore) 0011	1001	1	Right shift AQq_0
2	0001	~~1100~~ 1100	1	Right shift AQq_0
	1010	1100	1	A = A + M
1	~~0101~~ 1101	0110	0	RS AQq_0.
0	1110	1011	0	RS AQq_0

1001
1
1010

check 2's complement

(-21).

Restoring Division Algorithm.



(4)

11/3
= 3(Q),
2(R)

11 → Dividend
3 → Divisor

~~M = 1101~~
~~-M = 11101~~

3(Q), 2(R).

n	M	A	Q	Action
4	00011	00000	1011	Initialization
		00001	011_	SLAQ
		11110	011_	A = A − M
3		00001	0110	Q[0] ← 0, Restore
		00010	110_	SL AQ
		11111	110_	A = A − M
2		00010	1100	Q[0] ← 0, Restore
		00101	100_	SL AQ .
		00010	100_	A = A − M
1		00010	1001	Q[0] = 1 .
		00101	001_	SLAQ
		00010	001_	A = A − M
0		00010	0011	Q[0] = 1

00001
~~10000~~
~~10001~~
~~11111~~

2 − 3 = (−1)

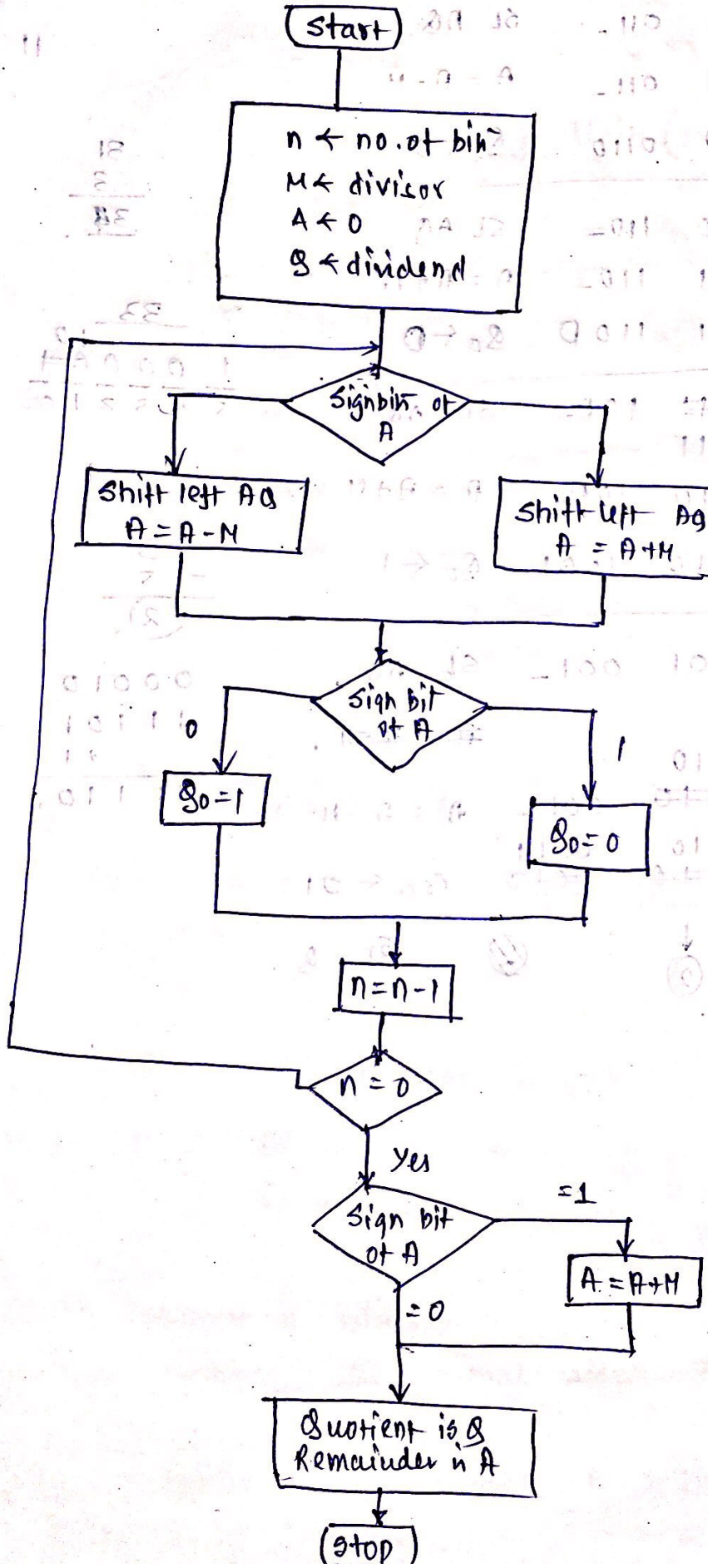
5
3
(2)

5
−3
2

Q = 0011 = 3
R = 00010 = 2

Scanned with CamScanner

Non-restoring Division Algorithm



n	M	A	Q	Action
4	00011	00000	1011	Initialization
		00001	011_	SL AQ
		11110	011_	A = A - M
3		11110	0110	$Q_0 \leftarrow 0$
		11100	110_	SL AQ
		11111	110_	A = A + M
2		11111	1100	$Q_0 \leftarrow 0$
		11111	100_	SL AQ
		00010	100_	A = A + M
1		00010	1001	$Q_0 \leftarrow 1$
		00101	001_	SL AQ.
		00010	001_	A = A - M
0		00010	0011	$Q_0 \leftarrow 1$

$R = (2)$ (2) $(3) = Q$.

3
11
31
3
34.

33
10
1 0 0 0 0 1
5 4 3 2 1 0

3
− 5
(+2).

00010
11101
+1
11110.

29/8/2019

Computer Organization

Post
Ram
Sarkar

Control Design

[JB Hayes]

→ Digital Processing Unit (DPU)
→ Control Unit (CU)
→ DPU is reconfigured by the CU to perform a set of (micro) operations.

CU

→ Instruction Sequencing → methods by which instructions are selected for execution
→ Instruction Interpretation → methods used for activating the control signals.

Instruction Sequencing

→ Memory ~~is~~ becomes a problem. The instruction length becomes increased, which in turn increase the cost of memory it is stored.

Soln. PC (Program Counter).

PC contains A of l_1.
$PC \leftarrow PC + K$. (K is the word length of l_1)

→ Increment is automatic
→ however, the actual address must be there in PC.
$PC \leftarrow X$.
Before PC, you need a buffer.

② Conditional branch test the value of condition C

if C is present
PC ← X

else
PC is incremented to point to the next instruction

Program Control Transfer

→ Often it is required to temporal tranfer of control from P1 to P2 due to subroutine call / interrupt.

→ In case of interrupt, call instruction is replaced by interrupt signal.

Control Stack

→ Stack stores return address.

→ Also used to store pass variables from P1 to P2, and variable local to P2.

LIFO ⟶ Last In First Out

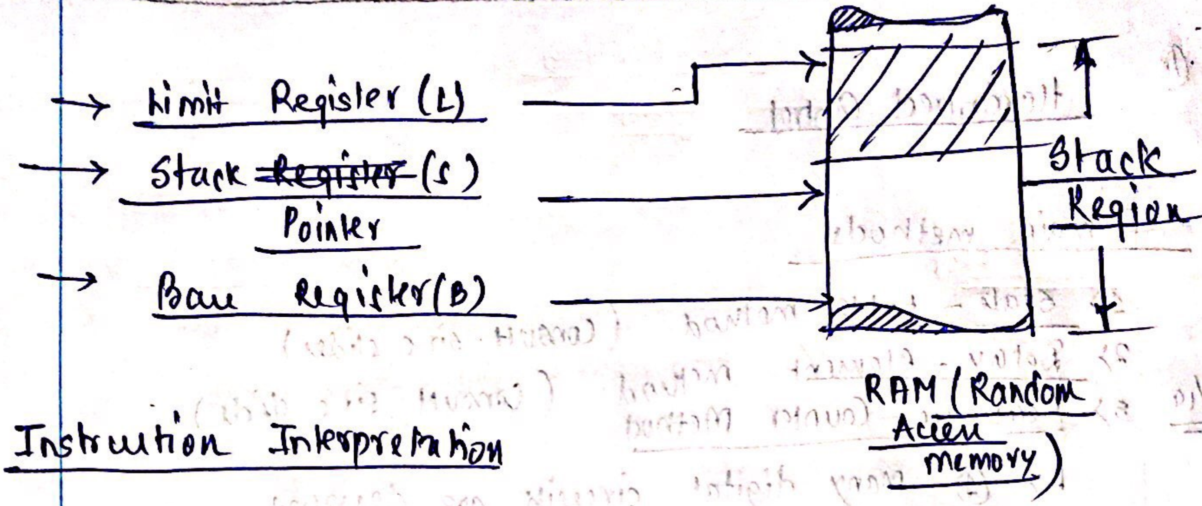
→ Pop out using return.

Stack Implementation

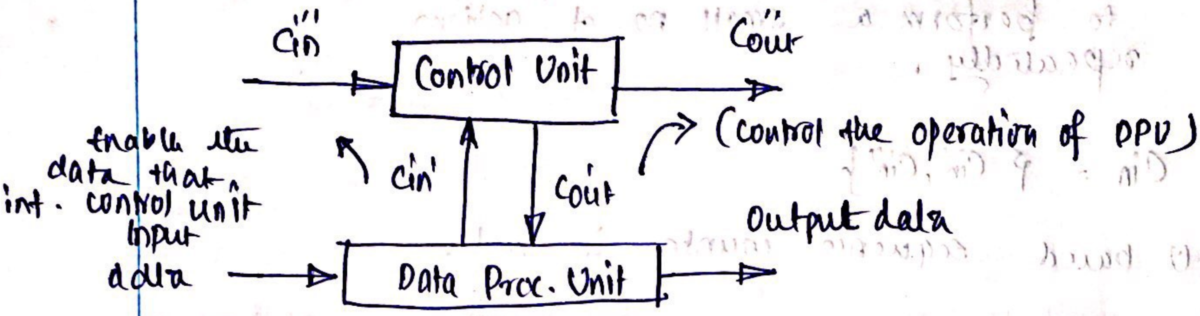
→ Pop & Push

→ Smaller stack: Shift Register
Larger stack: RAM

→ Stack pointer ← contains ~~on~~ the pointer acting as top of the stack.



Instruction Interpretation



Behaviour of CU

→ z₅+n identified by C_{ij}.

Implementation methods

→ Hardwired CU (HCU)

Goal

→ minimize the no. of components
→ maximize the speed of operation.
→ lack of structure makes HCU costly to design & debug.

Microprogramming

→ A (micro)instruction stored in specially addressable memory called a ~~Cont~~ Control Memory (CM).

→ flexible & systematic.

Emulation & Limitation

Microprogram

→ Sequence of microinstruction needed to execute a particular operation.

(A)

Hardwired Control

3 main methods

1) State - table method (Consult sir's slides)
2) Delay - Element method (Consult sir's slide)

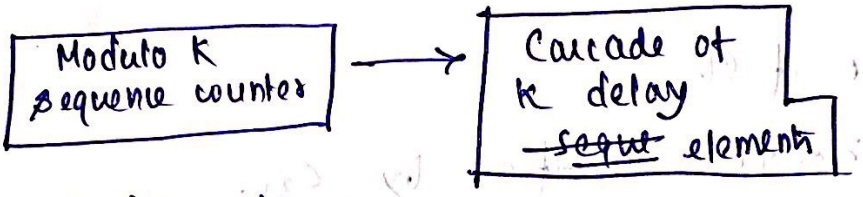
30/8/19

3) Sequence - Counter Method

↳ (1) Many digital circuits are designed to perform a small no. of actions repeatedly.

$C_{in} = \{ C_{in}', C_{in}'' \}$

→ CU based sequence counter is used.



→ Cascade of Delay Elements can be made to behave like a sequence counter

Micro-programmed control

→ Every instruction is implemented using no of operations.
→ Information is stored in RAM/ROM, called control memory (CM).
→ Each microinstruction explicitly / implicitly specifies the next microinstruction to be used.
→ Set of related micro-instructions → micro program

~~With~~ ~~With~~ Wilke's Design

→ Control field
→ Address field.

05/5/19

Computer Organisation

Prof. Ram Sarkar

Area of concern

- Word length of micro-instruction
- Way control info. is represented
- ~~Way next micro~~

Class Test.
upto Control unit
19th sept, 2019
2:30 – 3:30
(upto 6/9/19).

Control Memory

- ROM (diode)
- RAM (ferrite core)

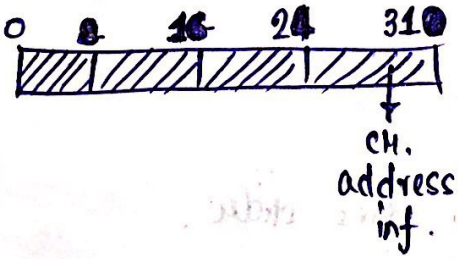
ROS → Read Only Store

Wilkes Idea :- Writable CM (WCM)

1. Dynamically microprogrammable.
2. Computer with WCM :- no instruction set in real sense.

~~3.~~

Parallelism in Microinstructions



Micro-instruction format
IBM sys. 370

Unencoded format :- Control signals may be derived directly from micro-instruction.

MI is loaded into CM data register (called μIR) [microinstruction register]

② Horizontal Microinstructions

(i) Long format ⟶ Parallelism.

~~(ii) ...~~

Vertical Microinstructions

(i) Short format ⟶ No parallelism.

Micro-instruction Addressing

→ Microprogram counter (μPC) for branching

→ Analogous to PC at instruction level

μPC ⟶ used as CM address register

Conditional Branching ⟶ flag generated by DPU.

~~if several such conditions exist,~~

$R \leftarrow f(R_1, R_2)$

Register Transfer Operation

4-phases ⟶ Refer to sir's slide.

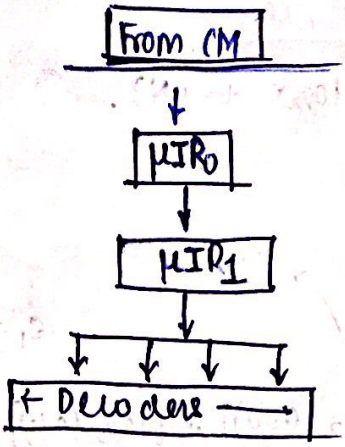
Phase $\phi 1$
Phase $\phi 2$
Phase $\phi 3$
Phase $\phi 4$

Micro-operation Timing

→ Fetch & execute steps can be overlapped.

→ Replace μIR by a pair of registers forming a two segment pipeline.

μIR_1 & μIR_0.



Residual Control

MI is used to allocate a resource eg. connection b/w 2 units may be established by MI and maintained via Residual Control.

Control Unit Organization

(Please refer to the diagram)

Prof. Rana Sarkar

04/9/19

Nano-programmed Computer

→ Typical microprograms are encoded in vertical format.

→ They have highly parallel horizontal format.

Advantages

→ Reduce size of chip.

→ Greater design flexibility resulting from the loosening of the bonds b/w instruction & hardware.

Disadvantage

→ Reduction in speed due to extra memory & more complex CU organisation.

(For the Math, refer to sir's slides)

④

$S_2 = H_M (2\log_2 H_M + \log_2 r + rN)$.

$H_M \rightarrow 650$
$r \rightarrow 0.4$
$N \rightarrow 70$
$H_N = 260$

$S_1 = H_H (N + \lceil \log_2 H_H \rceil)$

$\Rightarrow S_2 = 650 \left(\lceil 2\log_2 (650) \rceil + \lceil \log_2 (0.4) \rceil + 0.4 \times 70 \right)$

$S_1 = 650 \left(70 + \lceil \log_2 650 \rceil \right)$

$\Rightarrow S_1 = 80 \times 650 = 52000$

$S_2 = 29,900$

$S_2 < S_1$

$20 - 2 + 28$
$= 20 + 26$
$= 46$

Key characteristics of Memory Systems

[Refer to William Stroline]

Method of Accessing Units of Data

Linked List → Sequential Accen

Array → Random Accen

Accening a variable → Direct Accen.
& more structured

~~Acc~~ Key → Associative

capacity & Performance

16/08/2019 **Computer Organisation**

Arithmetic

Prof. Ram Sarka

→ Addition/Subtraction of Signed Numbers.

→ Propagation of carry.

→ n-bit adder.

→ k n-bit adder

→ n-bit subtractor ⟶ $x - y = x + \bar{y} + 1$.

concept of overflow & underflow

$$S_i = x_i \oplus y_i \oplus c_i$$

$$C_{i+1} = x_i y_i + y_i c_i + c_i x_i$$

'Control-Line' in adder/subtractor'

⟶ Overflow occurs when sign of the result & sign of the operands is different.

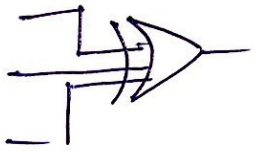
$$\text{Overflow} = x_{n-1} y_{n-1} \bar{s}_{n-1} + \bar{x}_{n-1} \bar{y}_{n-1} s_{n-1}$$

$$\text{Overflow} = c_n \oplus c_{n+1}$$

Time ⟶

For - Sum: 1 Gate Delay

For Carry: 2 Gate Delays.



For a n-bit adder

S_{n-1} avaibable after $2n-1$

C_n is available after $2n$ Gate Delays

$$S_i = x_i \oplus y_i \oplus c_i$$
$$c_{i+1} = x_i y_i + (x_i + y_i) c_i$$

$$c_{i+1} = G_i + P_i c_i$$
$$G_i = x_i y_i \qquad P_i = x_i + y_i$$

$G_i \rightarrow$ generate fn.
$P_i \rightarrow$ propagate fn.

→ Carry LookAhead adder

$$c_{i+1} = G_i + P_i G_{i-1} + P_i P_{i-1} G_{i-2} + \ldots + P_i P_{i-1} \ldots P_0 c_0,$$

All carries are obtained by 3 gate delays

→ Multiplication of unsigned number.

↳ AC, MQ (Von Neumann Arch.)

AC → accumulator
MQ → Multiplier Quotient.

Unsigned Binary Multipl.

12/9/19

Computer Organisation

William Stollings

Prof. Ram Sarkar

Memory

- Volatile memory
- Non-volatile memory
- Magnetic surface memory
- Semi-conductor

Refer to sir's slides.

Memory-hierarchy

→ All about

Inbound Memory
↓
Outbound Memory
↓
off-line Storage

~~CPU~~ ⟹

Cache & Main Memory

CPU → L1 → L2 → L3 → Main Memory

Cache Read Operation

Typical Cache Organisation

Elements of Cache Design

Cache Address

~~Main~~ Virtual Memory

Logical & Physical Cache

Direct Mapping

Logical cache and virtual cache

Disadvantage

Mapping th

Set Associative

Replacement Algorithm

4 most common replacement algorithm

→ LRU

→ FIFO

→ LFU.

Write Policy.

→ Write Through & Write Back

13/9/19

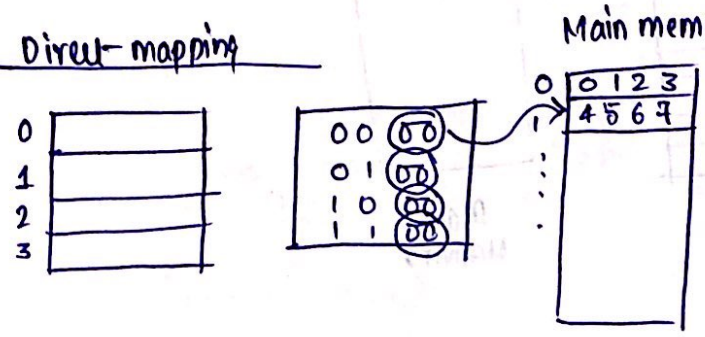
Computer Organisation

Prof. Ram Sarkar

Memory Mapping

i = cache line no
j = block in main memory
m = no. of lines in cache

1) Direct mapping



block no | block offset

0001 | 01 (=1) = 5.

cache memory

2	2	2

Tag line Block no. offset

Block size
4 words

No. of blocks in main memory $= \frac{64}{4} = 16$

Physical address = 6 bits.

00	0
11	1
10	2
11	3

Words

(0, 1, 2, 3)
(52, 53, 54, 55)
(40, 41, 42, 43).
(60, 61, 62, 63)

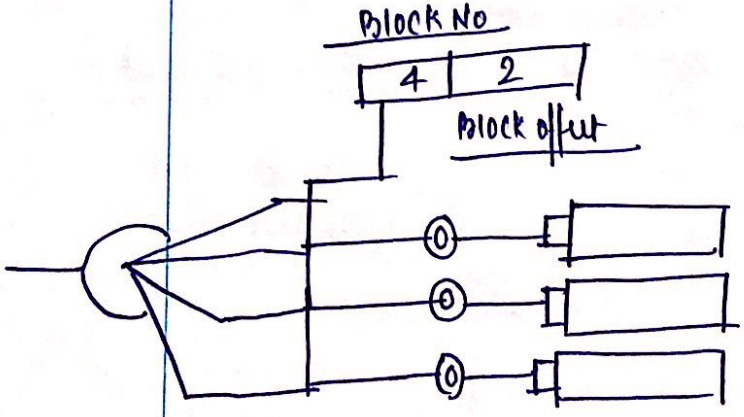
00	00	00

01
10
11

11	01	00

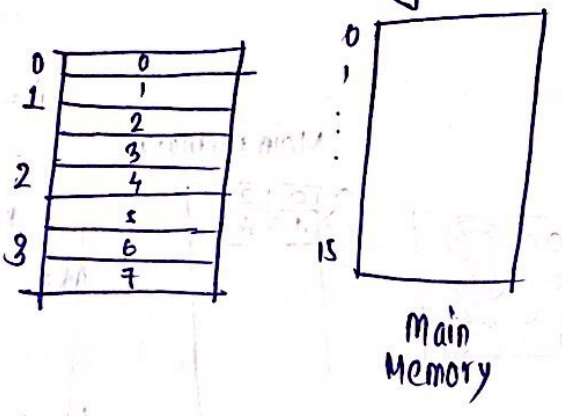
01
10
11

2) Associative Mapping



Main Memory = 32 GB (35 bits)
Cache memory = 32 KB (15 bits)
Tag = (35 - 15) = 20 bits
Propagation delay for comparator = 10s
" (20 x 10) = 200 ns
" OR gate = 10s
Hit latency = 210 ns

Set associative mapping



20/9/19

Computer Organisation

Prof. Ram Sarkar

Multi-Level Cache

Hit Ratio

→ Write Policy, Complicated design issues
Optimal Ratio b/w L1 & L2.

Unified cache vs Split cache

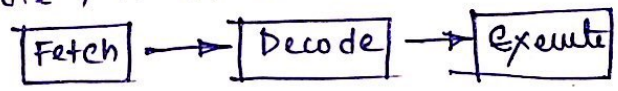
Read this from Siri slides

- One dedicated to instruction
- One dedicated to data

- Both exist at some level, typically L1 cache
- Eliminate cache contention.

Pipeline Architecture

When the fetch unit is idle, it can be used another instruction.



Split cache

RAM

Memory Cell Operation

Semi-conductor memory

Terminal

Select terminal, Control terminal, (terminal)

Both volatile,

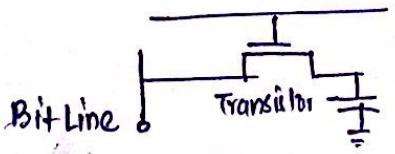
SRAM v/s DRAM

Dynamic RAM & Static RAM

Refer to the circuit diagram from Sir' slide.

No ~~refer~~ refreshing is required.

Read the mechanism used for cache memory.



DRAM

↳ We need refresh counter as capacitors are used

→ Typical 16Mb DRAM (4M x 4)
(Refer to sir's slides)

→ 4 bits can be read/written at a time.

→ Data In/ Sense.

$2^{11} = 2048$ ⟶ Row Decoder
↓
address lines

→ Buffer is required

Output is passed to the sense amplifier and presented to the data lines.

22 address lines

- 11 column address lines
- 11 row address lines.

Inter-leaved Memory

Collection is called memory bank.

→ Not good for localism
→ Not sequential.

Hard failure
& Soft Error.

Error Correcting Code Function

Advanced DRAM Organisation

~~SD-RAM~~ S-DRAM → synchronous with clock

burst mode
↳ useful when all bits accessed are in sequence & in the same row of the array as the initial access.

MR → specifies bus length.
helps to adjust latency.

DDR-DRAM

~~RT~~ R-DRAM

C-DRAM

Rambus

Data rate is doubled

Cache DRAM

27/9/19 — Computer Organisation — Prof Ram Sarkar

(Courtesy :- Titir Adhikary)

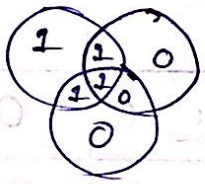
Parity - Bit

→ only flip the bit incase of inconsistancy.

M : 4. No. of comparisons :- 4

Data :- Inner component

Parity :- Outer component.



Design of Error Correction Function

η = Syndrome word → Function (X-OR)

8 bits (M)

4 bits (K)

12 bits (M+K).

K bit syndrome word $(2^K - 1)$.

$$(2^K - 1) \geq (M + K).$$

Check bit :- Bit posn whose position members are powers of 2.

- If K has all zeroes, there is no error
- If K has only 1, error has occured in K, no correction is reqd.
- If more than one 1s in K, posn indicate error, bits need to be inverted.

(Please refer to Mita Maam's notes).

D_3 avoided, D_5 avoided

$$C_1 = D_1 \oplus D_2 \oplus D_4 \oplus D_5 \oplus D_7.$$

$$C_2 = D_1 \oplus D_3 \oplus D_4 \oplus D_6 \oplus D_7$$

(D_2 avoided, D_5 avoided)

$$C_4 = D_2 \oplus D_3 \oplus D_4 \oplus D_8.$$

$$C_8 = D_5 \oplus D_6 \oplus D_7 \oplus D_8$$

Bit posn	12	11	10	9	8	7	6	5	4	3	2	1
Posn no	1100	1011	1010	1001	1000	0111	0110	0101	0100	0011	0010	0001
Data bit	D_8	D_7	D_6	D_5		D_4	D_3	D_2		D_1		
Check bit					C_8				C_4		C_2	C_1
Word stored as	0	0	1	1	0	1	0	0	1	1	1	1
Word fetched as	0	0	1	1	0	1	1	0	1	1	1	1

0111
+ 000
0110

6th posn

External Memory

Magnetic Disk

- Aluminium & Aluminium alloy.
- Magnetic film surface is uniform
- Ability to support lower fly height
- Better stiffness to withstand shock & damage.

Magnetic Read & Write Mechanisms

- 2 heads.
  Read & Write.
  +ve pole, -ve pole

Write Mechanism :- electricity flowing through a coil produces a magnetic field.

Inductive Write / Magnetoresistive Read Head

Disk Data Layout

Disk Layout Methods Diagram

- Constant angular velocity layout is not preferred.
  Multiple zoned is preferred.

Winchester Disk Format
Seagate ST506

Physical Characteristics of Disk Systems

Characteristics

- Fixed Head Disk
- Movable Head disk
- Non-removable disk.

(2)

Characteristics

(1) Tracks & Cylinders

Disk Classification

Winchester Heads

Typical Hard Disk Parameters

Timing of Disk I/O Transfer

Disk Performance Parameters

Non movable head system < delay Movable head system.

→ Access time
→ Transfer time.

RAID

Redundant Array of Independent Disks

No hierarchy implied

RAID 0 (Non-Redundant)
RAID 1 (Mirrored)
RAID 2 (Redundancy through Hamming code)
⋮
RAID 6

Advantages & Disadvantages.

18/10/2019

Computer Organisation

Post. Ram Sarkar.

RAID.

3 levels

→ OS should view the entire thing as a single drive.

→ Striping

→ ~~Redun~~ Concept of redundancy.

Raid Levels (RAID 2)

For Redundant via Hamming Code N+m drives are required.

m ∝ log N.

RAID 6 → N + 2.

RAID 4, 5, 6 → Independent Access.

RAID 2, 3 → Parallel Acccss.

Parallel accss → ~~Multiple Users~~ Single User

Independent accss → ~~Independent User~~. Multiple User.

RAID Comparison

RAID Level 0

RAID Level 1

→ No write penalty.

RAID Level 2

→ Hamming code is used

RAID Level 3

RAID Level 4

RAID Level 5 → Round Robin Scheme is used.

RAID Level 6 → Prevent single point failure.

→ ~~this~~ Most advantageous.

Morris-Mano

Pipelining → Improves throughput

→ Makes the execution of programs faster.

→ Arrange the hardware so that more operation can be performed at the same time.

→ The no. of operations can be increased.

Parallelism is there.

Traditional Pipeline Concept

Laundry eg.

→ It does not help latency, rather it helps throughput of entire workload.

Refer to the slides.

Use the Idea of Pipelining in a Computer

FDEW

+ F → Fetch
+ D → Decode
+ E → Execution
+ W → Write.

Role of Cache Memory

→ Expected to complete in one complete clock cycle.

→ The clock cycle should be long enough to let the slowest pipeline stage to complete.

→ Pipeline becomes useless, when each instruction has to be extracted from main memory.

Pipeline Performance

→ Refer to sir's slides

→ Any condition that causes a pipeline to stall is called a **Hazard**.

→ **Data Hazard**

→ **Instruction (or control) Hazard**

→ **Structural Hazard**

[Throughput is measured by the rate at which instruction execution is completed.]

Data Hazards

eg.
```
A ← 3+A
B ← 4×A
```

Sequential
```
MUL   R2, R3, R4
Add   R5, R4, R6
```

Operand Forwarding

Handling Data Hazards

~~Se~~ Let the compiler detect & handle the hazard.

Side Effects

eg. Autodecrement

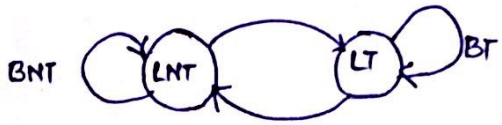
(Morris Mano)

Pipelining

Instruction Queue | Delayed Branch | Branch Prediction

↳ Compiler should extra-bit, when pipelining is used. However, it takes more time.

Instruction history can be used.
(LT → Likely to be taken,
LNT → Likely not to be taken).



✓ 2-state algorithm

✓ 4-state algorithm

Superscalar operation

→ Max throughput :- 1 inst./clock cycle.
→ Procesor capable of achieving an instruction execution throughput of more than one inst./cycle. → superscalar processors.

Superscalar

→ Order of execution may change.
(May cause inconsistency).

Out of order Execution

→ Inconsistent state
→ Imprecise Exception
→ Precise Exception
→ Commitment step

Execution Completion

→ Commitment unit → solves the problem of out of order execution
→ Reorder Buffer

Retired

→ Concept of a deadlock situation