19BCE2074
KULVIR SINGH

Date
Page

① Restoring Division Method.

117 / -9

Dividend = 117 = 01110101

-M = 111110111

n	M	A	Q	Operation
8	000001001	000000000	01110101	initialise.
		000000000	1110101-	L.S (AQ)
		111110111	1110101-	A = A - M
		000000000	11101010	Q[0] = 0, Restore
7		000000001	1101010-	L.S (AQ)
		111111000	1101010-	A = A - M
		000000001	11010100	Q[0] = 0, R.
6		000000011	1010100-	LS (AQ)
		111111010	1010100-	A = A - M
		000000011	10101000	Q[0] = 0, R.
5.		000000111	0101000-	LS (AQ)
		111111110	0101000-	A = A - M
		000000111	01010000	Q[0] = 0, R
4		000001110	1010000-	LS(AQ)
		000000101	1010000-	A = A - M
		000000101	10100001	Q[0] = 1
3		000001011	0100001-	LS (AQ)
		000000010	0100001-	A = A - M
		000000010	01000011	Q[0] = 1
2		000000100	1000011-	LS (AQ)
		111111011	1000011-	A = A - M
		000000100	10000110	Q[0] = 0, R

n.	M	A	Q	Operation.
•				
1	000001001	000001001	0000110–	LS (AQ)
		000000000	0000110–	A = A – M
		000000000	00001101	Q [0] = 1

$\underbrace{000000000}_{\text{Remainder}}$ $\underbrace{00001101}_{\text{Quotient}}$

Remainder: 0 | Quotient: $\underline{\underline{(+13)}}$

13 → Magnitude of Q

However the output should be negative.

∴ 2's comp. of 13.

$= (11110011)_2 = -13$

Remainder = $(000000000)_2$
Quotient = $(11110011)_2$
→ Answer

Non - Restoring Method.

n	M	A	Q	operation
8	000001001	000000000	01110101	Initialise
		000000000	1110101 –	LS (AQ)
		111110111	1110101 –	A = A – M
		111110111	11101010	Q[0] = 0
7		111101111	1101010 –	LS(AQ)
		111111000	1101010 –	A = A+M
		111111000	11010100	Q[0]=0
6		111110001	1010100 –	LS(AQ)
		111111010	1010100 –	A = A+M
		111111010	10101000	Q[0]=0
5.		111110101	0101000 –	LS(AQ)
		111111110	0101000 –	A = A+M
		111111110	01010000	Q[0]=0
4		111111100	1010000 –	LS(AQ)
		000000101	1010000–	A = A+M
		000000101	10100001	Q[0] = 1
3		000001011	0100001 –	LS(AQ)
		000000010	0100001 –	A = A–M
		000000010	01000011	Q[0] = 1
2		000000100	1000011 –	LS(AQ)
		111111011	1000011 –	A = A–M
		111111011	10000110	Q[0]=0
1		111110111	0000110 –	LS(AQ)
		000000000	0000110 –	A = A + M
0		000000000	00001101	Q[0] =1

Remainder = $(000000000)_2$

Quotient = $(000001101)_2$ = +13

However we considering signed integer.

$\therefore$ 2's Comp. of $\underline{\underline{13}}$

Quotient = $(111110011)_2$

Remainder = $(000000000)_2$ → Answer

2 a) 10100011 by 1011 | M = 00001011
-M = 11110101

Restoring Division.

n	A	Q	Operation.
8	00000000	10100011	initialise
	00000001	0100011 –	LS(AQ)
	11110110	0100011 –	A = A – M
	00000001	01000110	Q[0] = 0, R
7	00000010	1000110 –	LS(AQ)
	11110111	1000110 –	A = A – M
	00000010	10001100	Q[0] = 0, R
6	00000101	0001100 –	LS(AQ)
	11111010	0001100 –	A = A – M
	00000101	00011000	Q[0] = 0, R
5	00001010	0011000 –	LS(AQ)
	11111111	0011000 –	A = A – M
	00001010	00110000	Q[0] = 0, R
4	00010100	0110000 –	LS(AQ)
	00001001	0110000 –	A = A – M
	00001001	01100001	Q[0] = 1
3	00010010	1100001 –	LS(AQ)
	00000111	1100001 –	A = A – M
	00000111	11000011	Q[0] = 1
2	00001111	1000011 –	LS(AQ)
	00000100	1000011 –	A = A – M
	00000100	10000111	Q[0] = 1
1	00001001	0000111 –	LS(AQ)
	11111110	0000111 –	A = A – M
	00001001	00001110	Q[0] = 0, R

Remainder = 9 Quotient = 14

2b) 00001111 by 0011 | M = 00000011
Restoring Method | −M = 11111101

n	A	Q	Operation
8	00000000	00001111	Initialise
	00000000	0001111 –	LS(AQ)
	11111101	0001111 –	A = A − M
	00000000	00011110	Q[0] = 0, R
7	00000000	0011110 –	LS(AQ)
	11111101	0011110 –	A = A − M
	00000000	00111100	Q[0] = 0, R
6	00000000	0111100 –	LS(AQ)
	11111101	0111100 –	A = A − M
	00000000	01111000	Q[0] = 0, R
5	00000000	1111000 –	LS(AQ)
	11111101	1111000 –	A = A − M
	00000000	11110000	Q[0] = 0, R.
4	00000001	1110000 –	LS(AQ)
	11111110	1110000 –	A = A − M
	00000001	11100000	Q[0] = 0, R
3	00000011	1100000 –	LS(AQ)
	00000000	1100000 –	A = A − M
	00000000	11000001	Q[0] = 1
2	00000001	1000001 –	LS(AQ)
	11111110	1000001 –	A = A − M
	00000001	10000010	Q[0] = 0, R
1	00000011	0000010 –	LS(AQ)
	00000000	0000010 –	A = A − M
	00000000	00000101	Q[0] = 1
	Remainder = 0	Quotient = 5	

(3)

Flynn's Classification of Computers

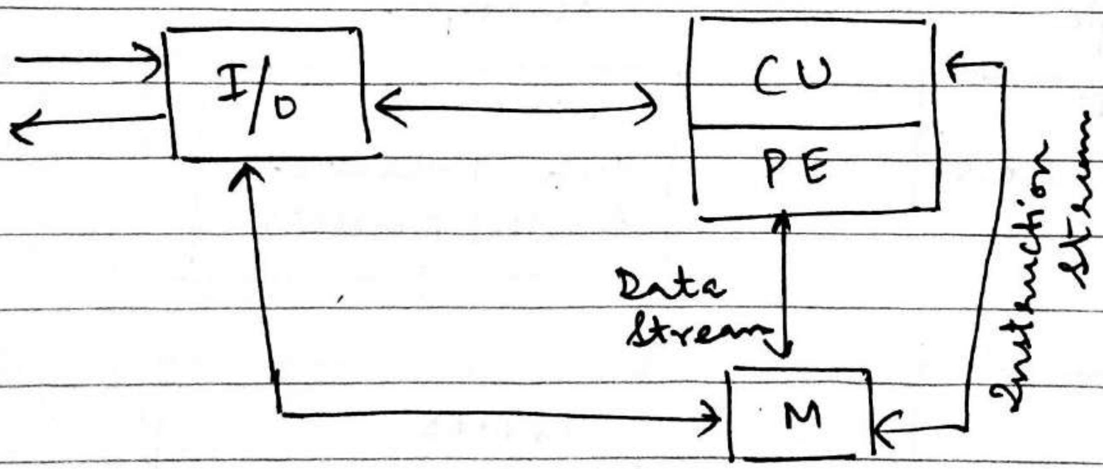
MJ Flynn proposed a classification for the organisation of a computer system by the number of instructions and data items that are manipulated simultaneously. The sequences of instruction read from memory constitutes an INSTRUCTION STREAM. The operations performed on the data in the processor constitute a DATA STREAM. Parallel processing may occur in the instruction stream, data stream or both.

DATA STREAM

INSTRUCTION STREAM	Single	Multiple
Single	SSID Uniprocessors	SMID Vector Processors Parallel Processing
Multiple	MISD May be pipelined computers	MIMD Multi-Computers Multi-Processors

SSID

SSID stands for Single Instruction and Single Data Stream. It represents the organisation of a single computer containing a control unit, a processor unit and a memory unit. Instructions are executed sequentially, and the system may or may not have internal parallel processing capabilities. Most conventional computers have SSID architecture like the traditional Von-Neumann computer.
Parallel Processing in this case, may be achieved by means of multiple functional units or by pipeline processing.



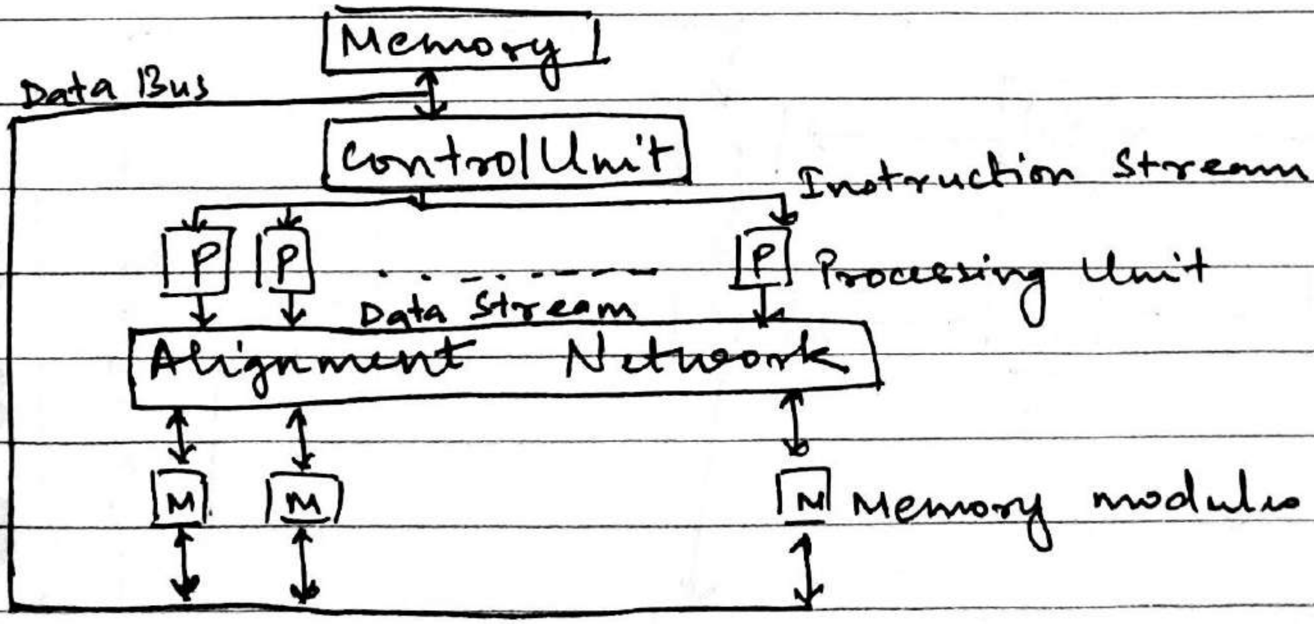
CU = control unit, PE = Processing Element, M = Memory.

Instructions are decoded by the CU, then sends them to the P.E for execution. Data stream between processors & memory is bi directional

Eg:
Older generation computers, minicomputers etc.

SIMD

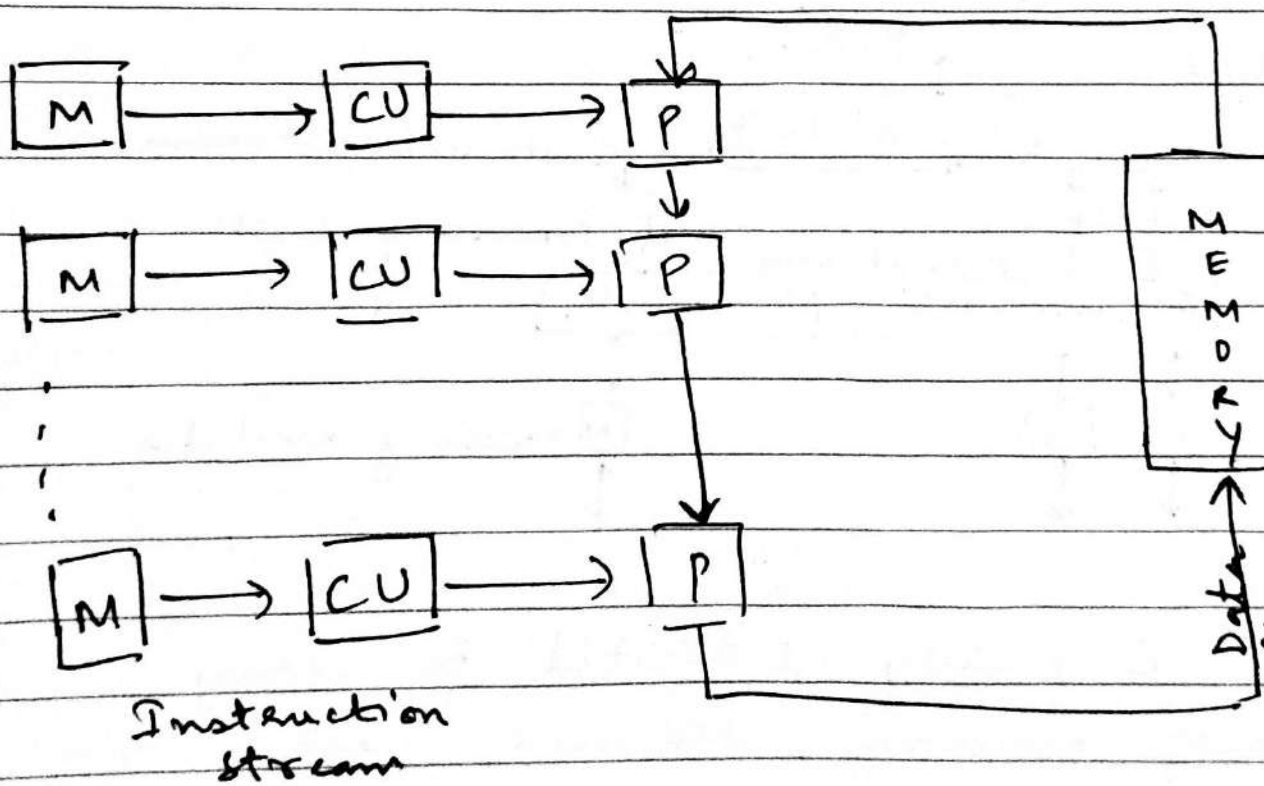
Single instruction and Multiple Data Stream.
It represents an organisation that includes many processing units under the supervision of a common control unit. All processors recieve the same information from control unit but operate on different items of data. The shared memory unit must contain multiple modules so that it can communicate with all processors simultaneously.



SMID is mainly dedicated to array processing machines. However, vector processors can also be seen as a part of this group.

MISD :

MISD stands for Multiple Instruction and Single Data Stream. MISD structure is only of theoritical interest since no practical system has been constructed using this organisation. In MISD, multiple processing units operate on one single-data stream. Each processing unit operates on the data independently via separate instruction stream.

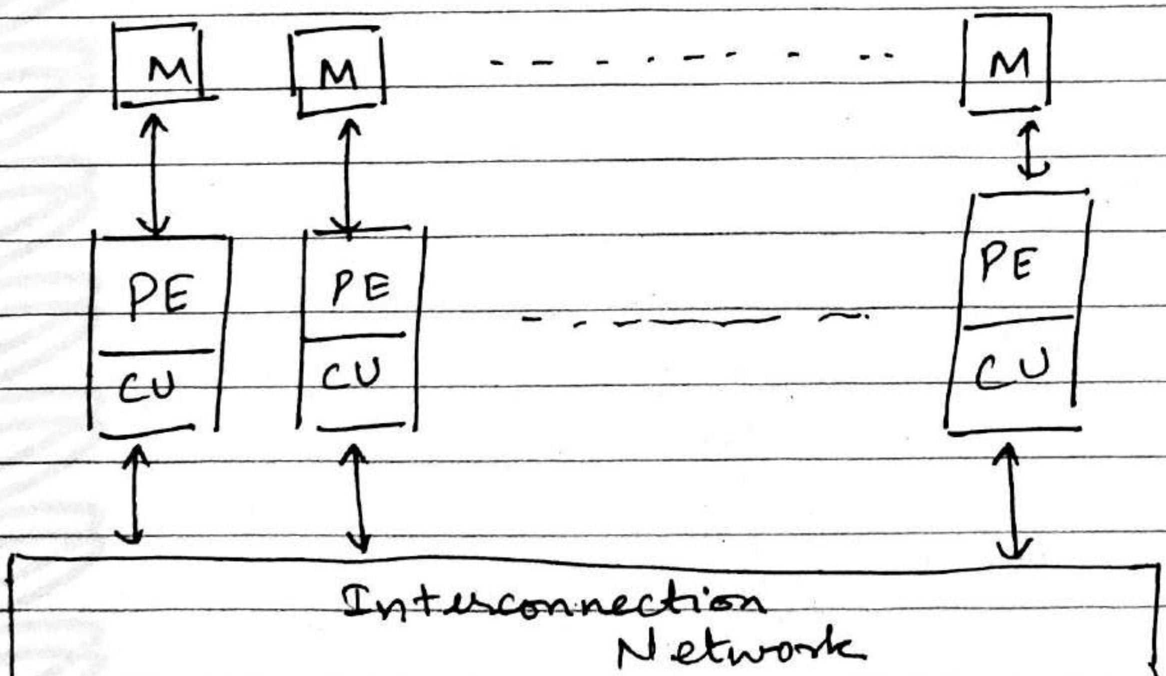


M = Memory Modules, CU = Control Unit,
P = Processing Units.

Eg :
The experimental Carnegie-Mellon C.mmp computer (1971)

MIMD

MIMD stands for Multiple Instruction and Multiple Data Stream. In this organisation, all processors in a parallel computer can execute different instructions and operate on various data at the same time. In MIMD, each processor has a separate program and an instruction stream is generated from each program.



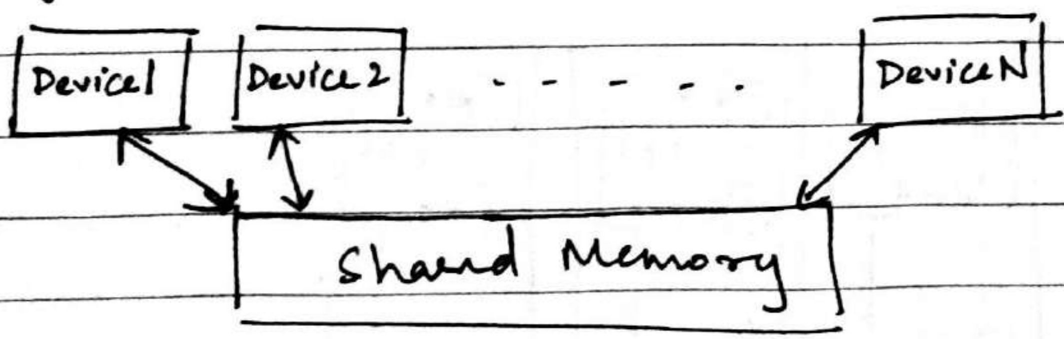
M : Memory Module, PE : Processing Element, CU : Control Unit.

Eg:
Cray T90, Cray T3E, IBM-SP2

4

Shared Memory Architecture

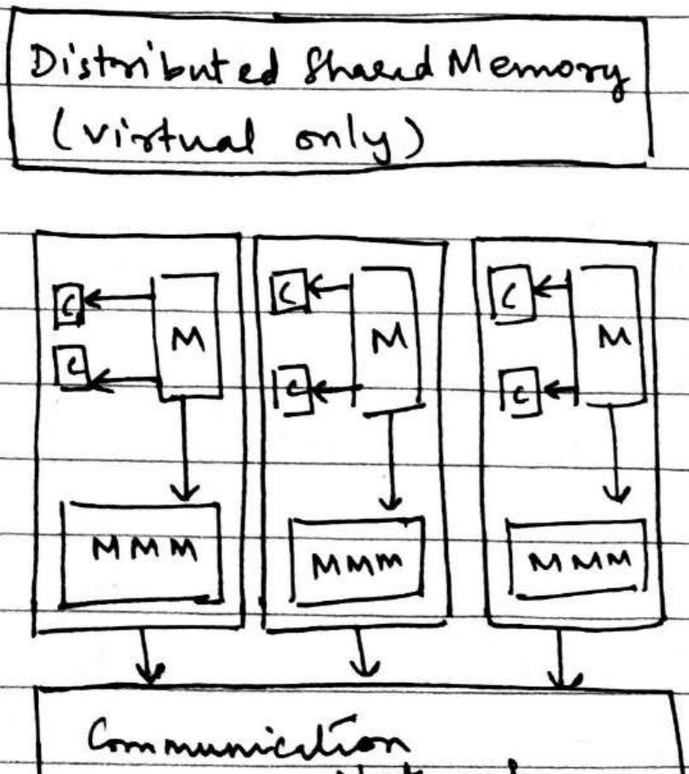
Devices exchange information by writing to and reading from a pool of shared memory as shown in the figure below. Unlike a shared bus architecture, in a shared memory architecture, there are only point-to-point connection between the device and the shared memory, somewhat easing the board diagram and layout. Also each interface can run at full bandwidth all the time so the overall bandwidth is much higher than in shared bus architecture



In practice, the device may require an external arbiter in order to gain fair access to the memory with no collisions.

Architecture of Distributed Shared Memory (DSM)

DSM implements the distributed systems shared memory model in a distributed system, that hasn't any physically shared memory. shared model provides a virtual address area shared between any or all nodes. To beat the high forged of communication in distributed system. DSM memo, model provides a virtual address area shared between all nodes.



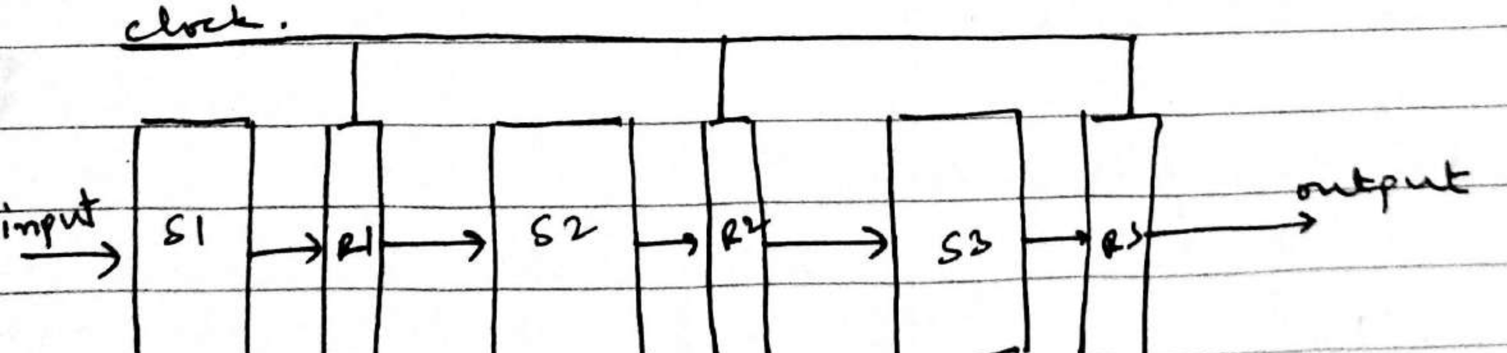
C ⇒ CPU [1 to n]

M ⇒ Memory

MMM ⇒ Memory Mapping Manager

(5)

Pipelining is the process of accumulating instruction from the processor through a pipeline. It allows storing and executing instructions in an orderly process. It is also known as pipeline processing. Pipelining is a technique where multiple instructions are overlapped during execution. Pipeline is divided into four stages and these stages are connected with one another to form a pipe like structure. Instructions enter from one end and exit from another end.



Pipeline system →

Pipeline Hazards are those when a functional unit is not fully pipelined. The use of the functional unit requires more than one clock cycle. If an instruction follows an instruction that is using it, and the second instruction also requires the resource it must stall.

Data Hazards occur when several instruction are in partial execution, and if they reference the same data then the problem arises. We must ensure that next instruction does not attempt to access data before the current instruction, because this will lead to incorrect results.

Advantages / Importance :-

(i) cycle time of processor is reduced

~~(iii)~~

(ii) Increases the throughput of system

(iii) Makes the system reliable.