LECTURE 7

Example 1: Timing for a simple vector operation

Line	*Instruction*	*Description*
1	A1 53	Set Register A1 to 53
2	VL A1	Set the length of the vector to 53
3	V4 V2+V3	Perform the addition

Setup Time _____ 3 cycles
(Vector integer adder is a 3 stage pipeline)

Time required to compute first result _____ 3 cycles.

First result emerges after _____ 3+3= 6 cycles

Time required to computer remaining 52 results _____ 52 cycles
Shutdown Time _____ 3 cycles
Destination register V4 will be available after 6+52+3 = 61 cycles
The functional unit, i.e. adder can be reused after VL+4= 57 cycles
The vector source register V2 and V3 are reserved for VL+3 = 56 cycles

---

Example 2(a): No source reservation conflict

Line	Instruction	Description
1	A1 10	Set Register A1 to 10
2	VL A1	Set vector length to 10
3	V4 V3+FV2	Set V4 to floating sum of V3 and V2
4	V6 V5*FV7	Set V6 to floating product of V5 and V7

Instr.	1	2	3	4	5	6	7	8	9	0	1 10	2	3	4	5	6	7	8	9	0 20	1	2	3	4	5	6
1	I	E																								
2		I	E																							
3			I	E	E	E	E	E	E	E	E	E	E	E	E	E	E	E	E	E	E	E	E	E		
4				I	E	E	E	E	E	E	E	E	E	E	E	E	E	E	E	E	E	E	E	E	E	E

Example 2(b): Source reservation results in a conflict

Line	Instruction	Description
1	A1 10	Set A1 to 10
2	VL A1	Set the vector length to 10
3	V4 V3+FV2	Set V4 to floating sum of V3 and V2
4	V6 V3*FV7	Set V6 to floating product of V3 and V7

Instr	1	2	3	4	5	6	7	8	9	0	10	2	3	4	5	6	7	8	9	20	1	2	3	4	5	6	7	8	9	30	1	2	3	4	5	6	7	8
1	I	E																																				
2		I	E																																			
3			I	E	E	E	E	E	E	E	E	E	E	E	E	E	E	E	E	E	E	E	E	E	E													
4																I	E	E	E	E	E	E	E	E	E	E	E	E	E	E	E	E	E	E	E	E	E	E

V3 is reserved by instruction 3 for VL+3 cycles, i.e. 13 cycles (4-16)

VECTOR CHAINING

Line	Instruction
1	AL 10
2	VL A1
3	V4 V3+FV2
4	V5 V4*FV7

In chaining, the results produced by one operation can be used as input to a succeeding operation can be used as input to a succeeding operation before the first instruction has completed. That is, the component results from the first instruction are used by the second instruction as they are produced.

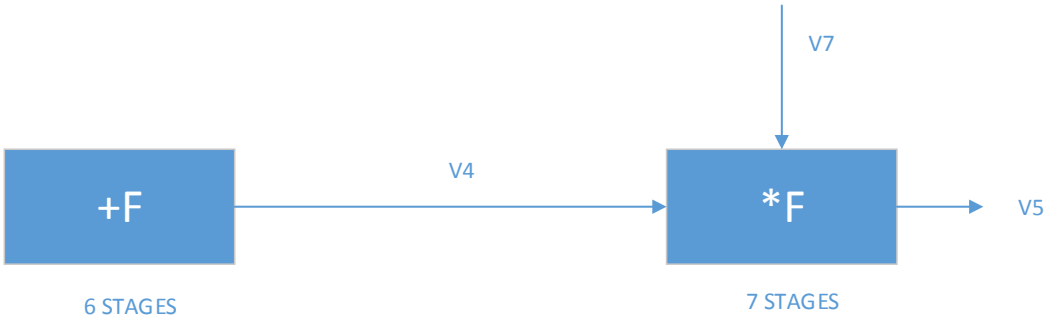
In chaining, a value which emerges from the pipeline may be used by a waiting operation 2 cycles after it is produced. This is contrast to unchained operations where the destination register cannot be accessed until 3 cycles after the last component computation is complete.

Ins	1	2	3	4	5	6	7	8	9	0	10	2	3	4	5	6	7	8	9	20	1	2	3	4	5	6	7	8	9	30	1	2	3
1	I	E																														
2		I	E																													
3			I	S	S	S	E	E	E	E	E	E	E	E	E	E	E	E	E	E	E	E	E	E								
4				I	S	S	S	H	H	H	H	H	H	H	E	E	E	E	E	E	E	E	E	E	E	E	E	E	E	E	E	E	E

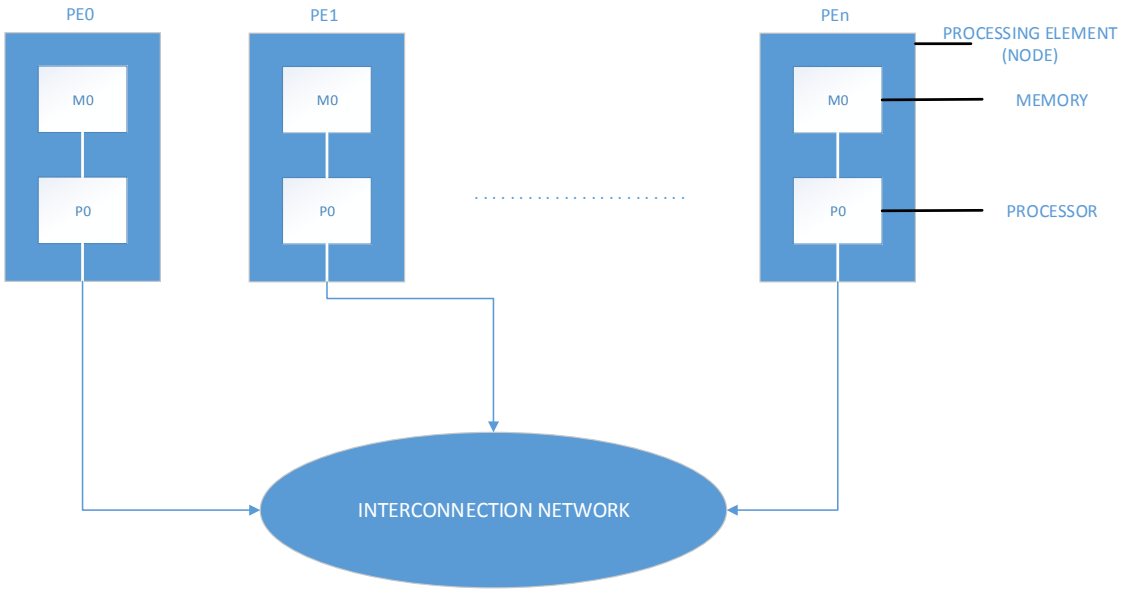
First result of line 3

Line 4 begins chained execution

First result of line 4



MIMD ARCHITECTURES



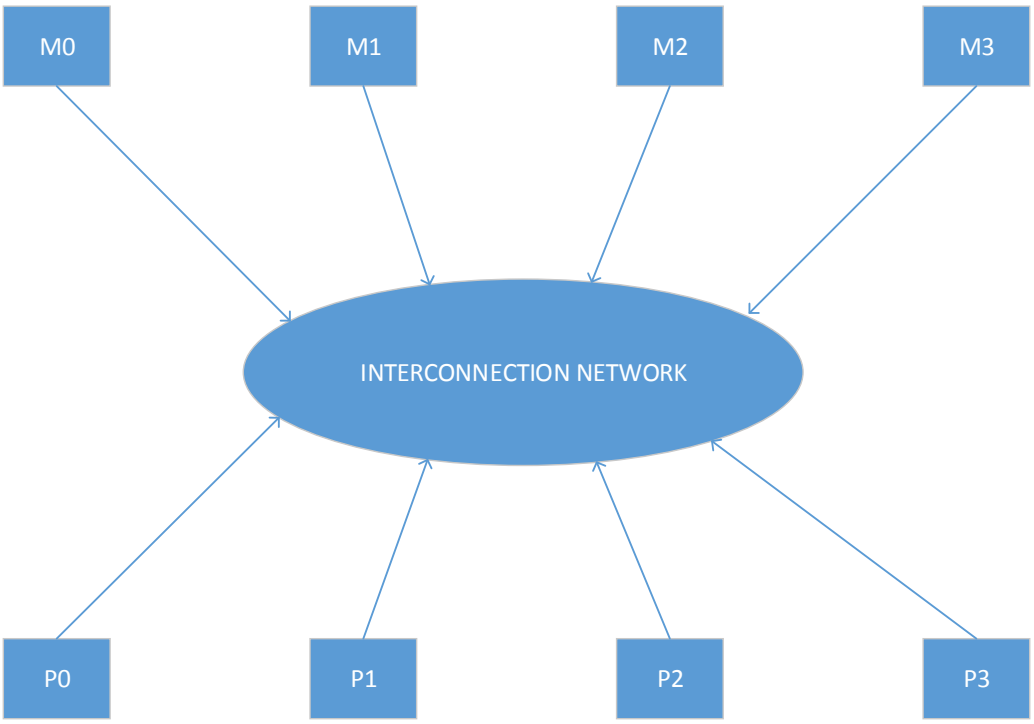
(FIGURE 2 - Structure of Distributed Memory MIMD Architectures)

(1) Distributed Memory (or message passing):

Whenever interaction among PEs is necessary, they send messages to each other. None of the PEs can ever access the memory module of another PE directly.

(2) Shared Memory:

Any processor can directly access any memory via an interconnection network. The set of memory modules defines a global address space which is shared among the processors.



(FIGURE 3- Structure of Shared memory MIMD Architectures)

Interconnection networks

Distributed memory MIMD architectures are often simply called multicomputers while shared memory MIMD architectures are referred to as multiprocessors

According to their topology interconnection networks can be classified as static or dynamic.

Static networks: Connection of switching units is fixed and typically realized as direct or point-to-point connections. These networks are also called direct networks.

Dynamic networks: Communication links can be reconfigured by setting the active switching units of the system.

Multicomputers are typically based on static networks while dynamic networks are mainly employed in multiprocessors

Distributed Memory Systems – ADVANTAGES:

- Since processors work with their attached local memory module most of the time, the contention problem is not as severe as in shared memory systems. As a result, distributed memory multicomputers are highly scalable and good architectural candidates for building massively parallel computers
- Processes cannot communicate through shared data structures and hence sophisticated synchronization techniques like monitors are not needed. Message passing solves not only communication, but synchronization as well.

Distributed Memory Systems – DISADVANTAGES:

- In order to achieve high performance in multicomputers, special attention should be paid to load balancing.
- Message passing based communication and synchronization can lead to deadlock situations
- Although there is no architectural bottleneck in multicomputers, message passing requires physical copying of data structures between processes. Intensive data copying can result in significant performance degradation.