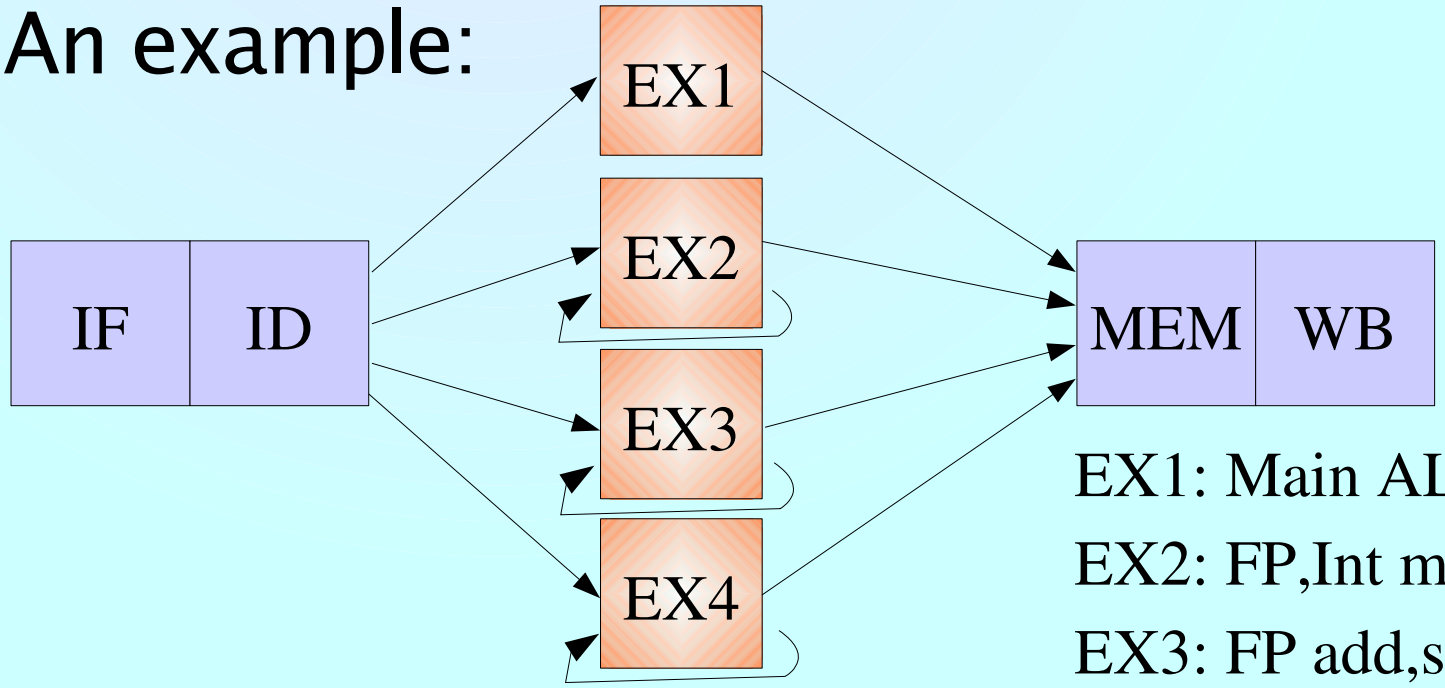
LECTURE - 09

Pipelining Multi-cycle Opns.

- Some operations take > 1 cycle (e.g. FP)
- Handling multi-cycle opns. in the pipeline:
  - Multiple EX stages
  - Multiple functional units
- An example:



EX1: Main ALU

EX2: FP,Int mult.

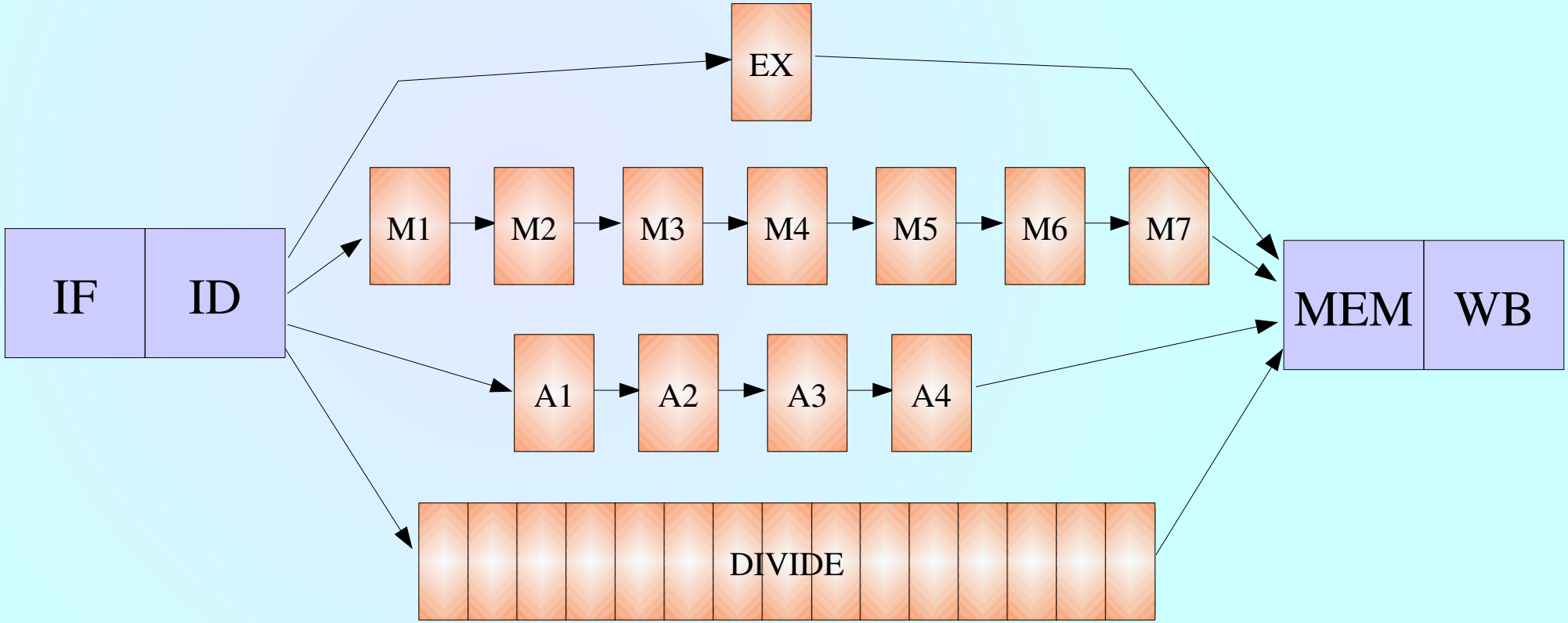
EX3: FP add,sub

EX4: FP,Int div.

Pipelining Multi-cycle Opns. (continued)

- Two things to consider:
  - Different units may take different # cycles
  - Some units may not be pipelined
- Corresponding definitions:
  - *Latency:* # cycles between an instn. & another which can use its result
  - *Initiation/repeat interval:* # cycles between issue of two operations of the same type

The Multi-cycle Pipeline



Functional Unit	Latency	Initiation interval
Main ALU	0	1
Data memory	1	1
FP add, sub	3	1
FP, Int mul	6	1
FP div	14	15

Pipeline Timing: An Example

	CC1	CC2	CC3	CC4	CC5	CC6	CC7	CC8	CC9	CC10	CC11
MULTD	IF	ID	M1	M2	M3	M4	M5	M6	M7	MEM	WB
ADDD		IF	ID	A1	A2	A3	A4	MEM	WB		
LD			IF	ID	EX	MEM	WB				

- Additional details:
  - We require more latches
  - ID/EX register must be expanded

More Hazards!

- Structural hazards:
  - Divide unit is not pipelined
  - Multiple writes possible in the same cycle
- Data hazards:
  - RAW is more frequent
  - WAW is possible
- Control hazards:
  - Out-of-order completion ==> difficulty in handling exceptions

Multiple Writes/Cycle: An Example

	CC1	CC2	CC3	CC4	CC5	CC6	CC7	CC8	CC9	CC10	CC11
MULTD F0, F4, F6	IF	ID	M1	M2	M3	M4	M5	M6	M7	MEM	WB
...		IF	ID	EX	MEM	WB					
...			IF	ID	EX	MEM	WB				
ADDD F2, F4, F6				IF	ID	A1	A2	A3	A4	MEM	WB
...					IF	ID	EX	MEM	WB		

Multiple Writes/Cycle: Solution

- Provide multiple write-ports
- Or, detect and stall; Two possibilities:
  - Detect in ID stage
    - Instruction reserve the write port using a reservation register
    - Reservation register is shifted one bit each clock
  - Detect in MEM stage
    - Easier to check
    - Can also give priority to longer cycle operation
    - But, stall can now be in two places
    - Stall may trickle back