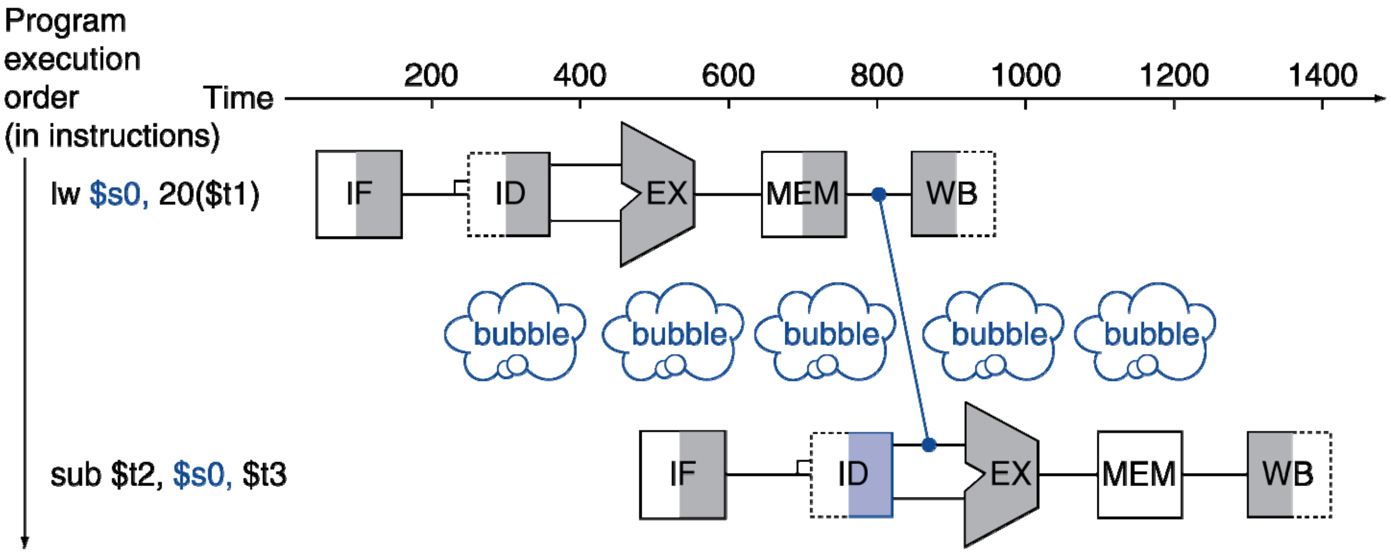
Chapter 4

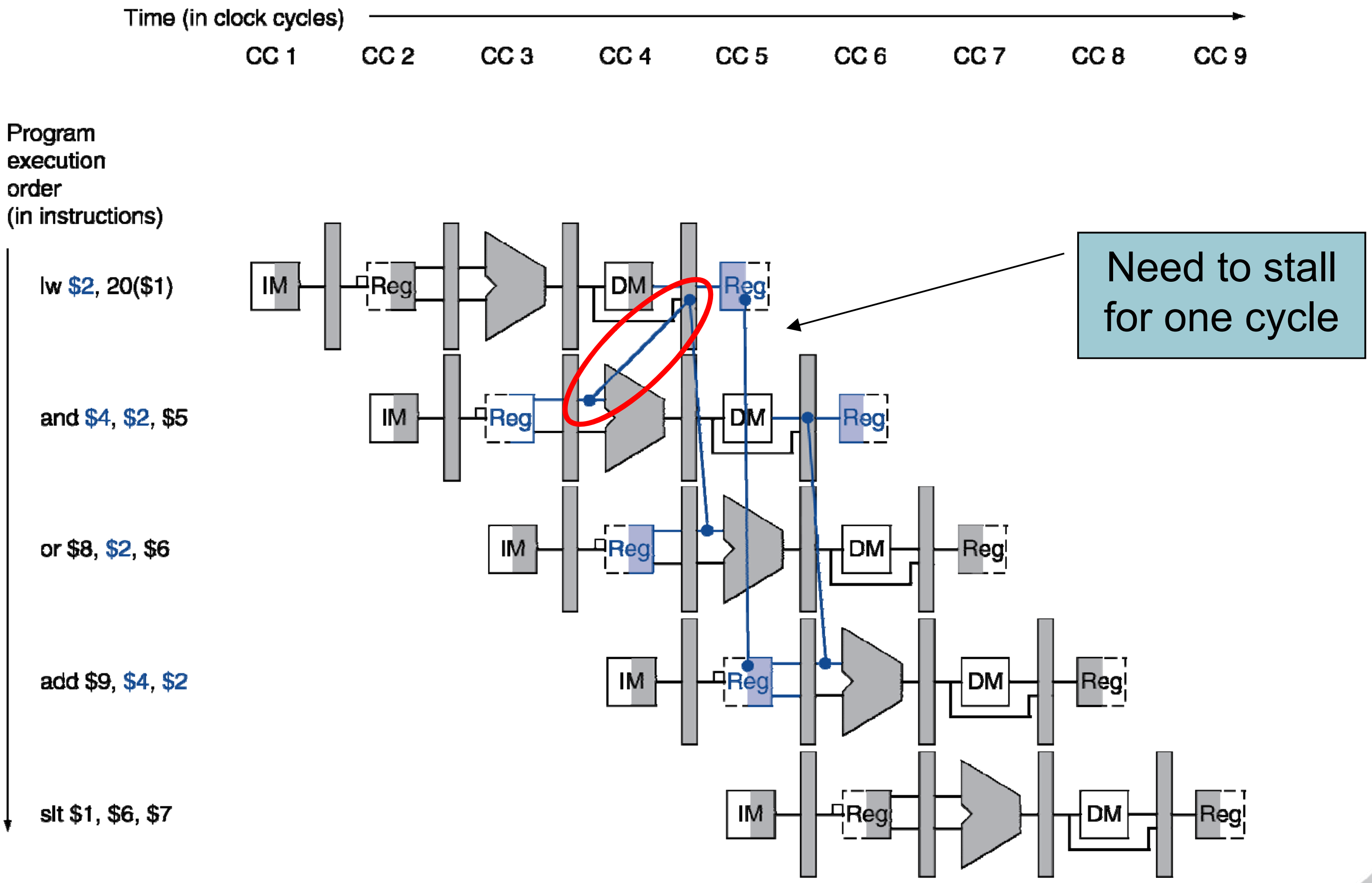
The Processor

Load-Use Data Hazard

- Can’t always avoid stalls by forwarding
  - If value not computed when needed
  - Can’t forward backward in time!



Load-Use Data Hazard



Need to stall for one cycle

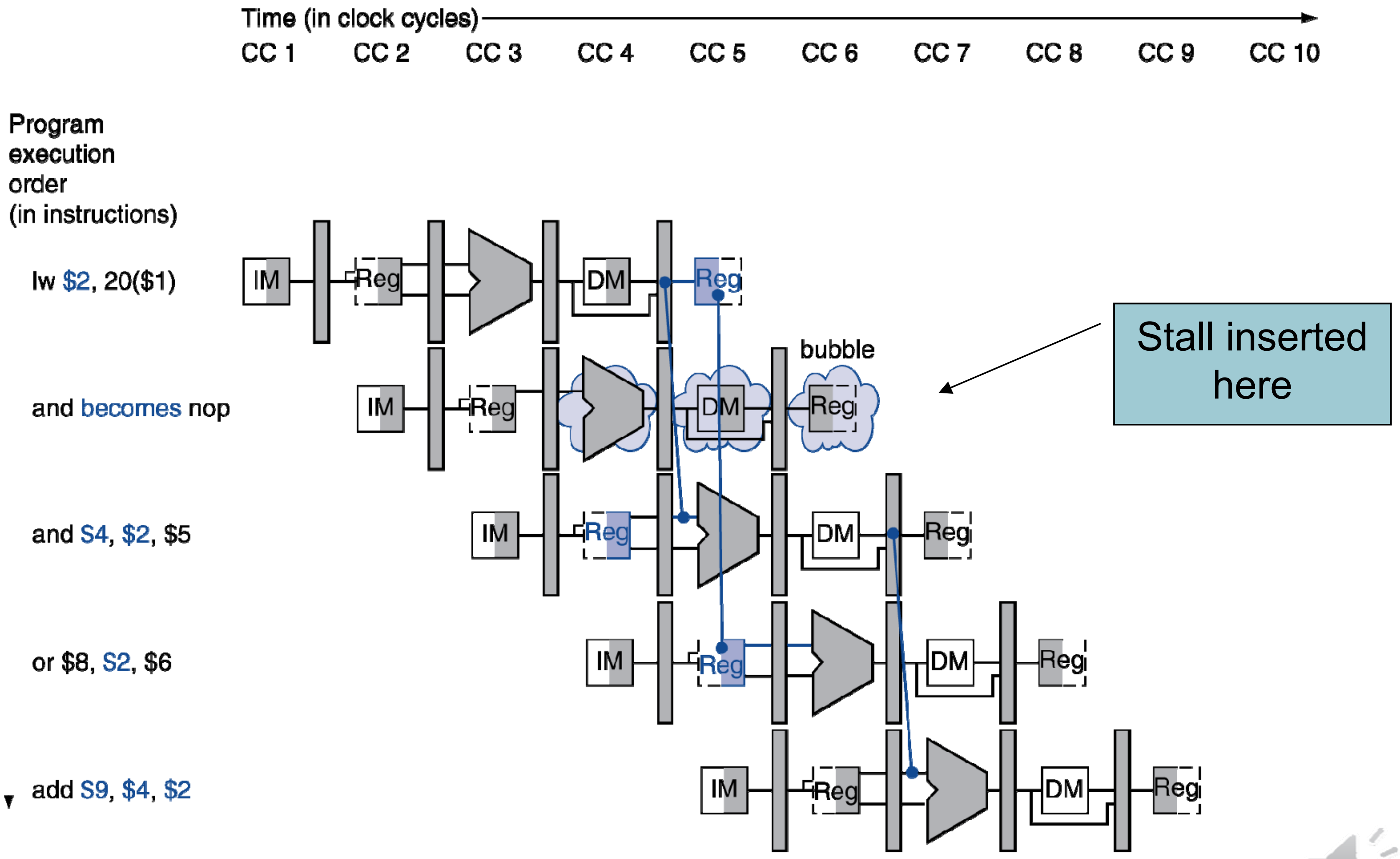
Load-Use Hazard Detection

- Check when using instruction is decoded in ID stage
- ALU operand register numbers in ID stage are given by
  - IF/ID.RegisterRs, IF/ID.RegisterRt
- Load-use hazard when
  - ID/EX.MemRead and
    ((ID/EX.RegisterRt = IF/ID.RegisterRs) or
    (ID/EX.RegisterRt = IF/ID.RegisterRt))
- If detected, stall and insert bubble

How to Stall the Pipeline

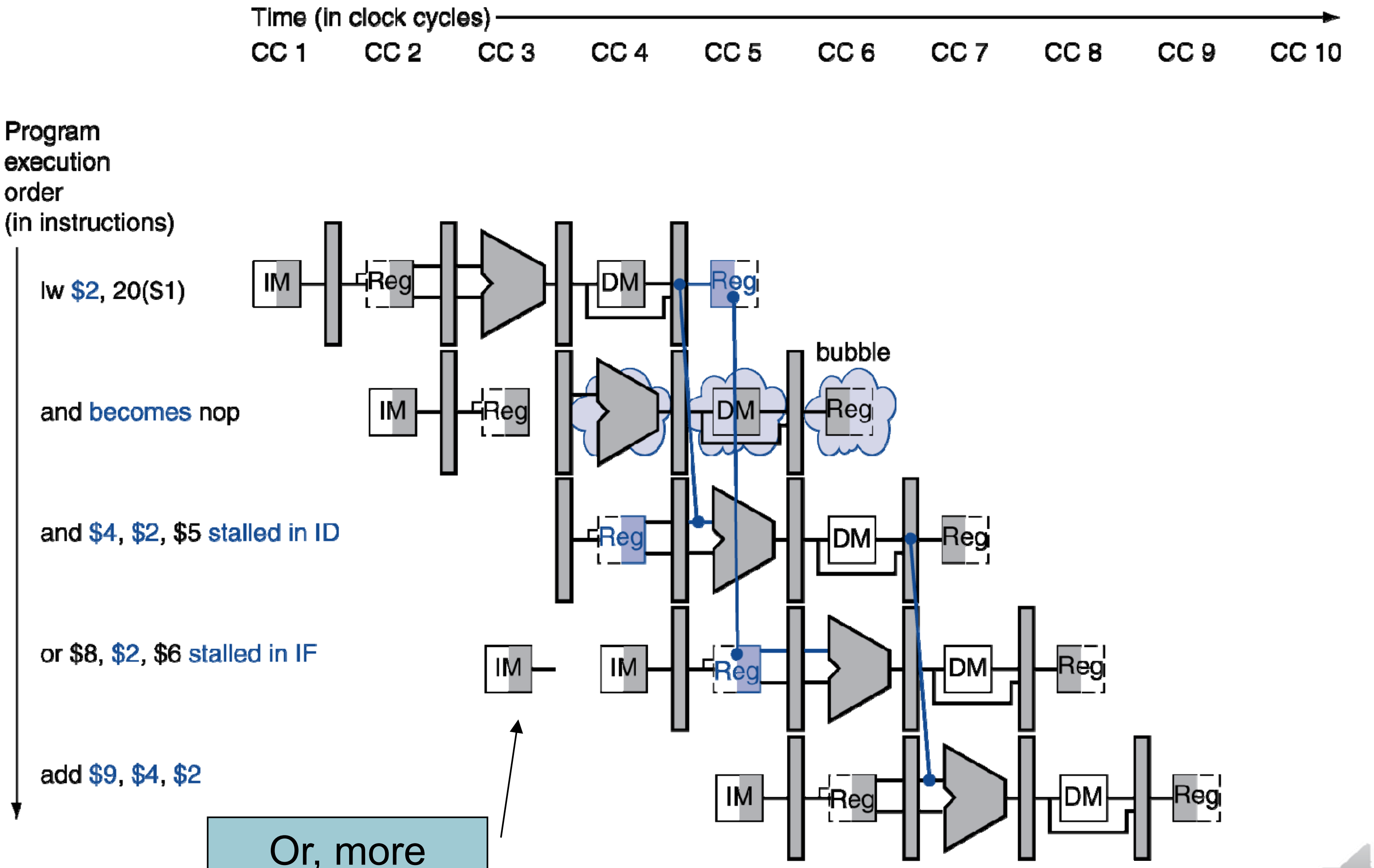
- Force control values in ID/EX register to 0
  - EX, MEM and WB do `nop` (no-operation)
- Prevent update of PC and IF/ID register
  - Using instruction is decoded again
  - Following instruction is fetched again
  - 1-cycle stall allows MEM to read data for `lw`
    - Can subsequently forward to EX stage

Stall/Bubble in the Pipeline



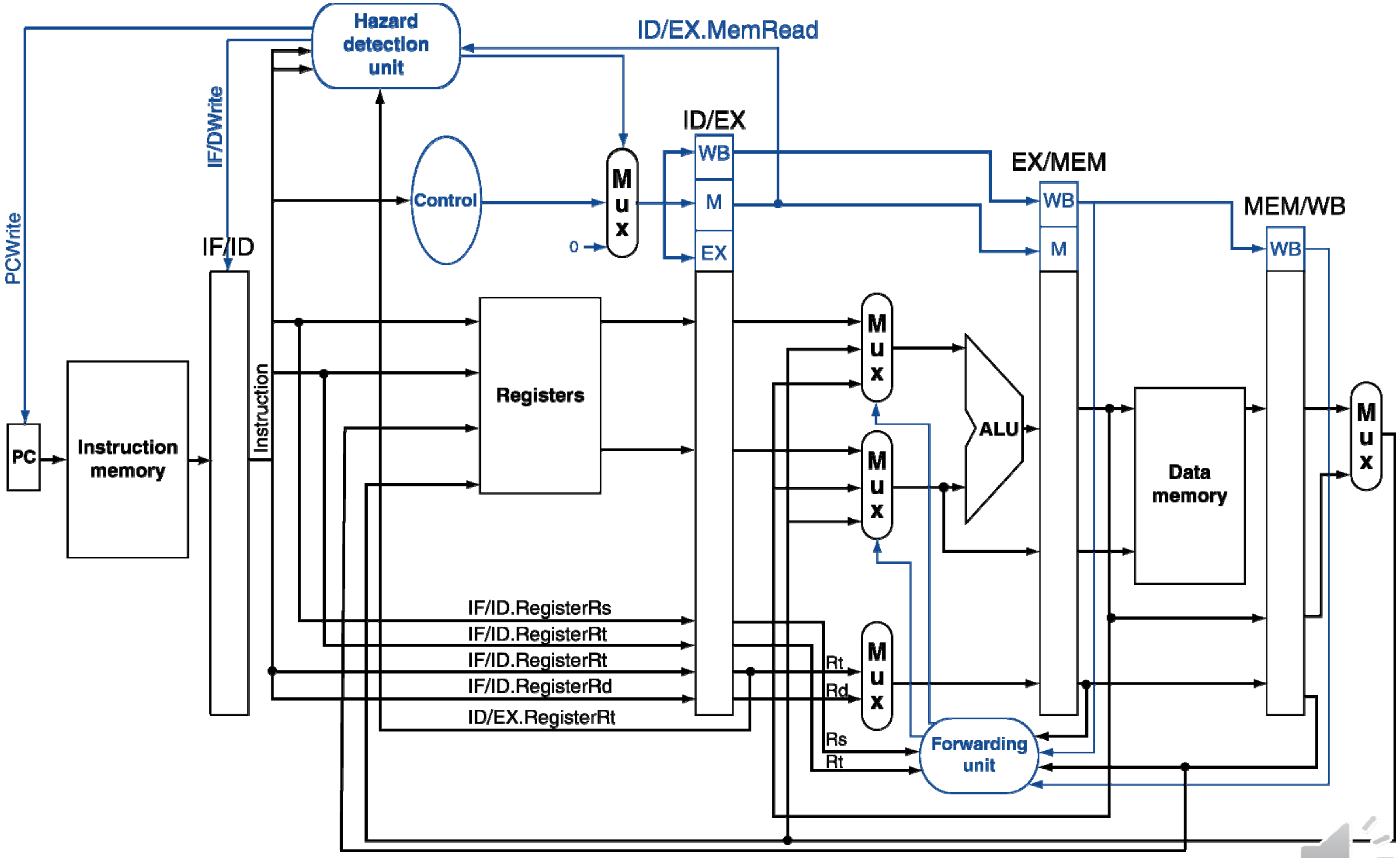
Stall inserted here

Stall/Bubble in the Pipeline



Or, more accurately…

Datapath with Hazard Detection


Hazard detection unit
ID/EX.MemRead
IF/DWrite
PCWrite
ID/EX
EX/MEM
MEM/WB
IF/ID
Control
M u x
0
WB
M
EX
WB
M
WB
PC
Instruction memory
Instruction
Registers
M u x
M u x
ALU
Data memory
M u x
IF/ID.RegisterRs
IF/ID.RegisterRt
IF/ID.RegisterRt
IF/ID.RegisterRd
ID/EX.RegisterRt
Rt
Rd
M u x
Rs
Rt
Forwarding unit