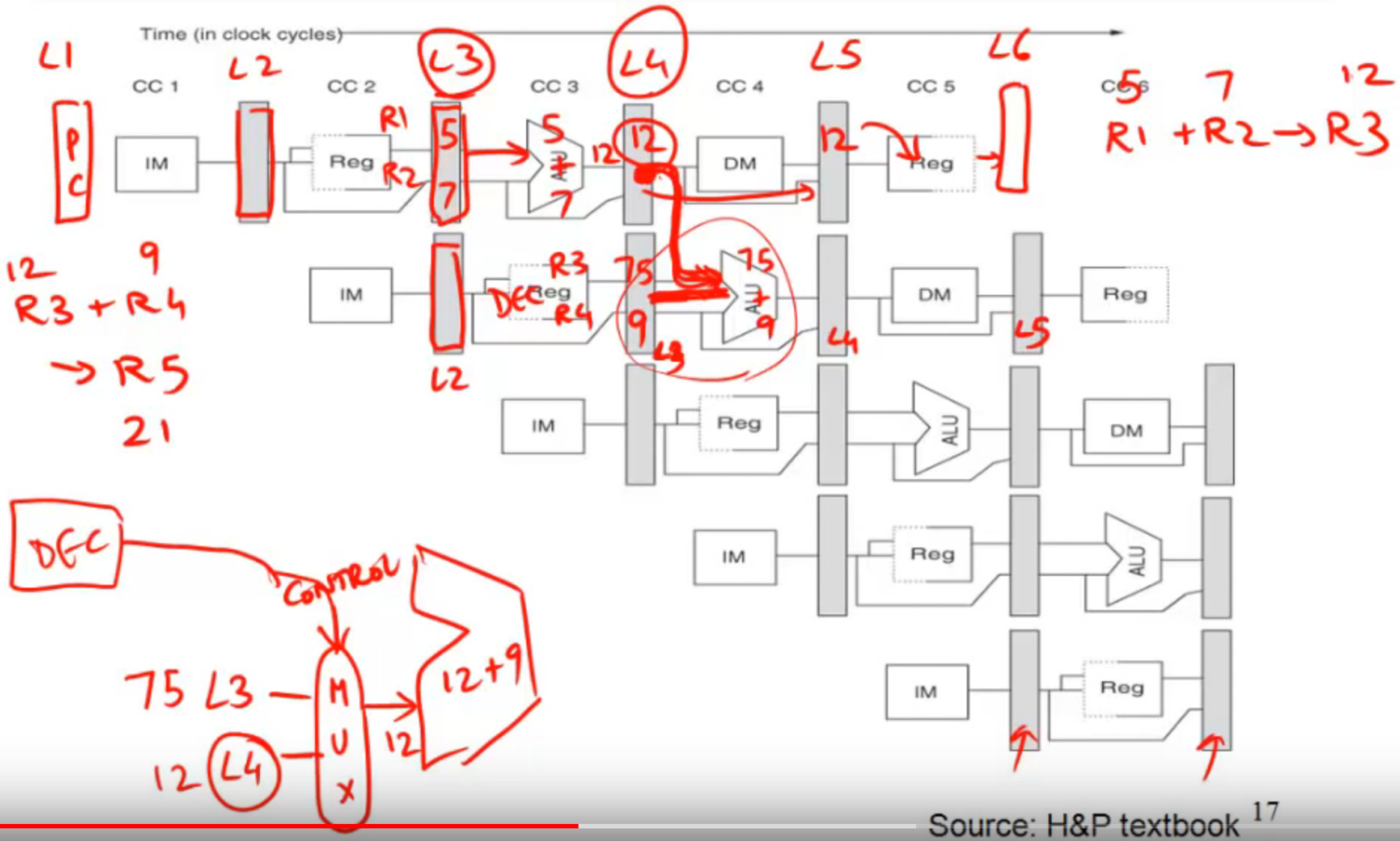
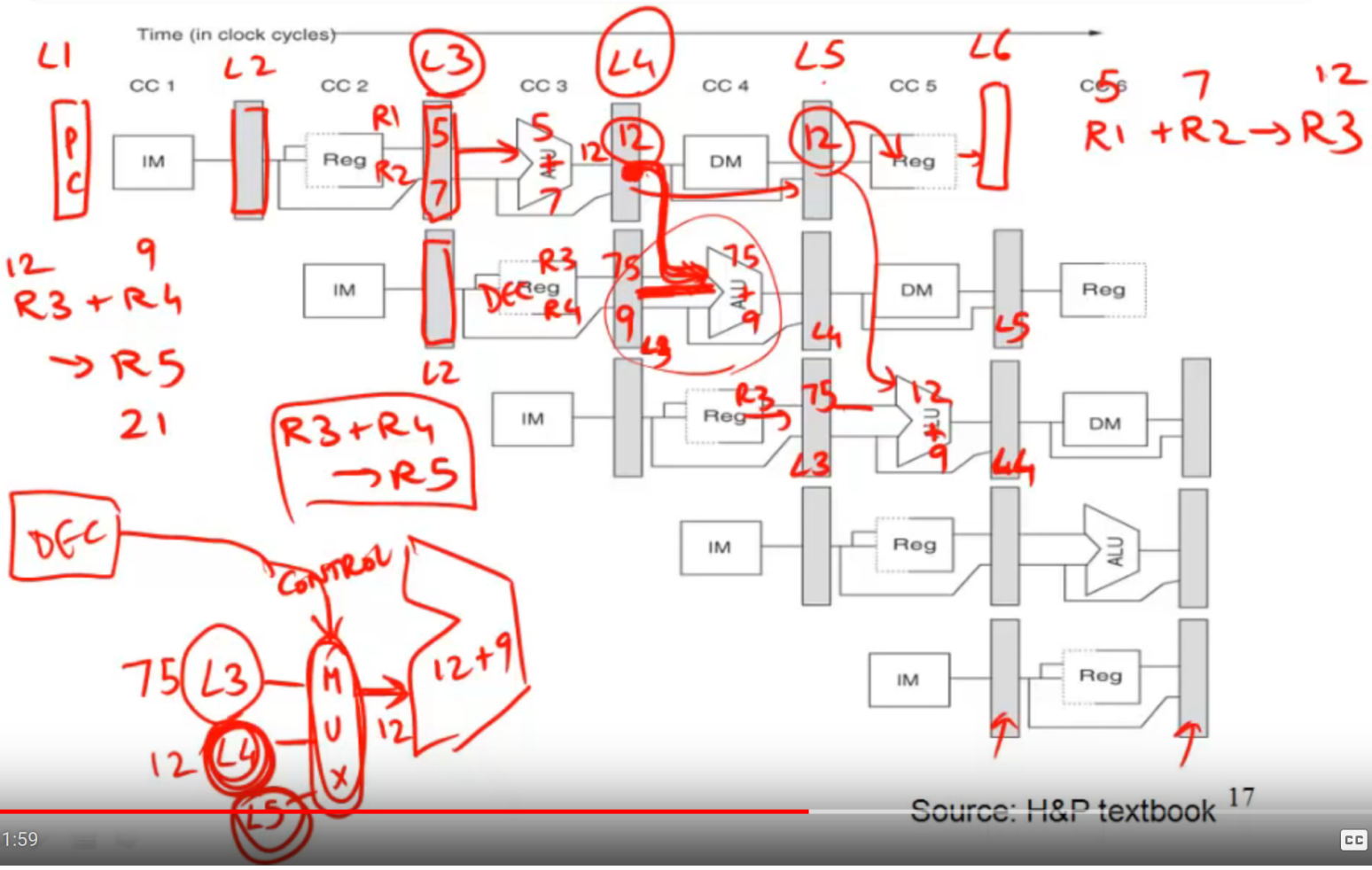
A 5-Stage Pipeline



Source: H&P textbook

A 5-Stage Pipeline



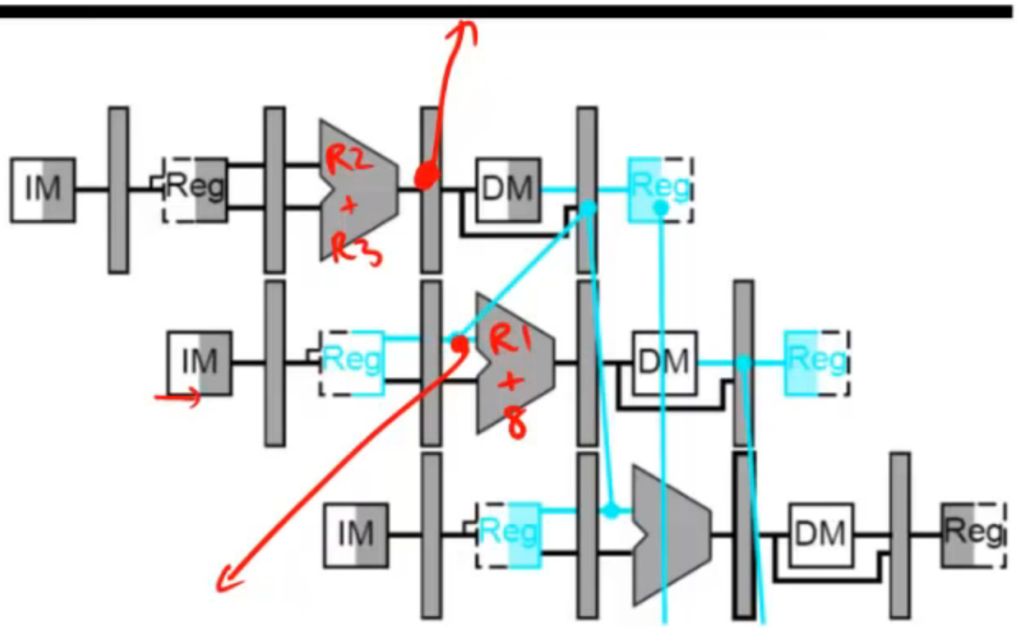
Source: H&P textbook

Example

Point of Production

add R1, R2, R3

lw R4, 8(R1)



Point of Consumption

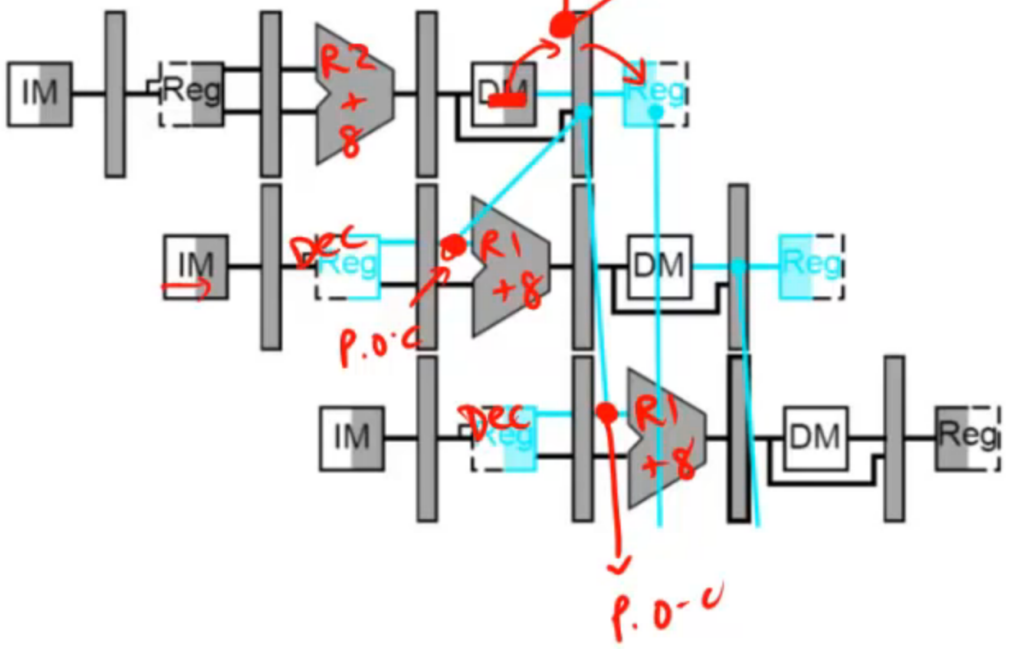
Source: H&P textbook

Example

lw R1, 8(R2)

1 stall

lw R4, 8(R1)



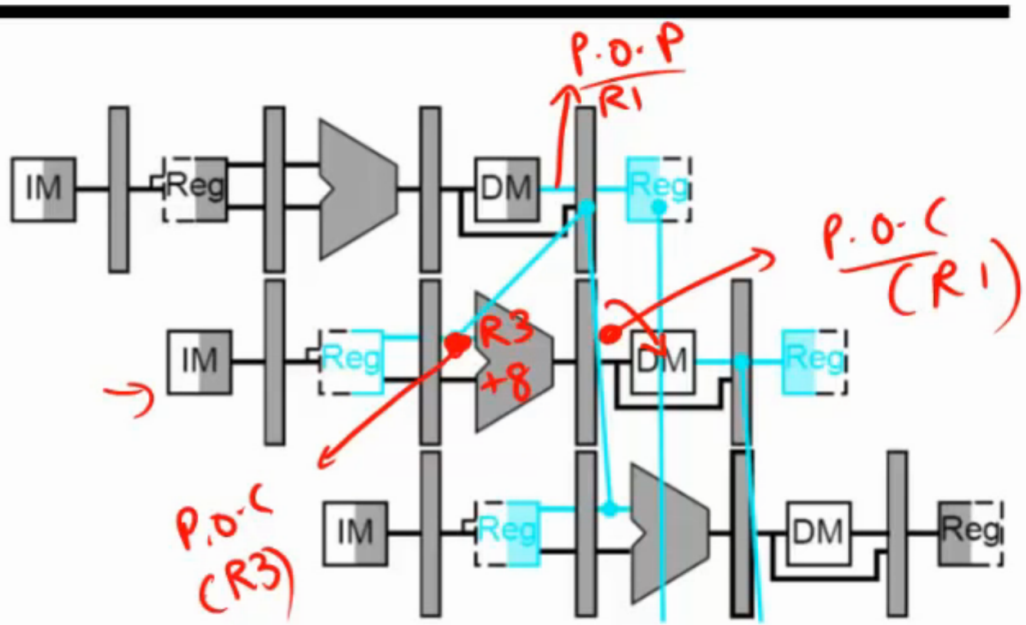
Source: H&P textbook

Example

lw R1, 8(R2)

0 stalls

sw R1, 8(R3)



Source: H&P textbook

Summary

- For the 5-stage pipeline, bypassing can eliminate delays between the following example pairs of instructions:

  add/sub R1, R2, R3 → P.O.P → 3rd stage — 0 stalls
  add/sub/lw/sw R4, R1, R5

  lw R1, 8(R2) → P.O.P → 4th stage
  sw R1, 4(R3) → P.O.C → 4th stage — 0 stalls

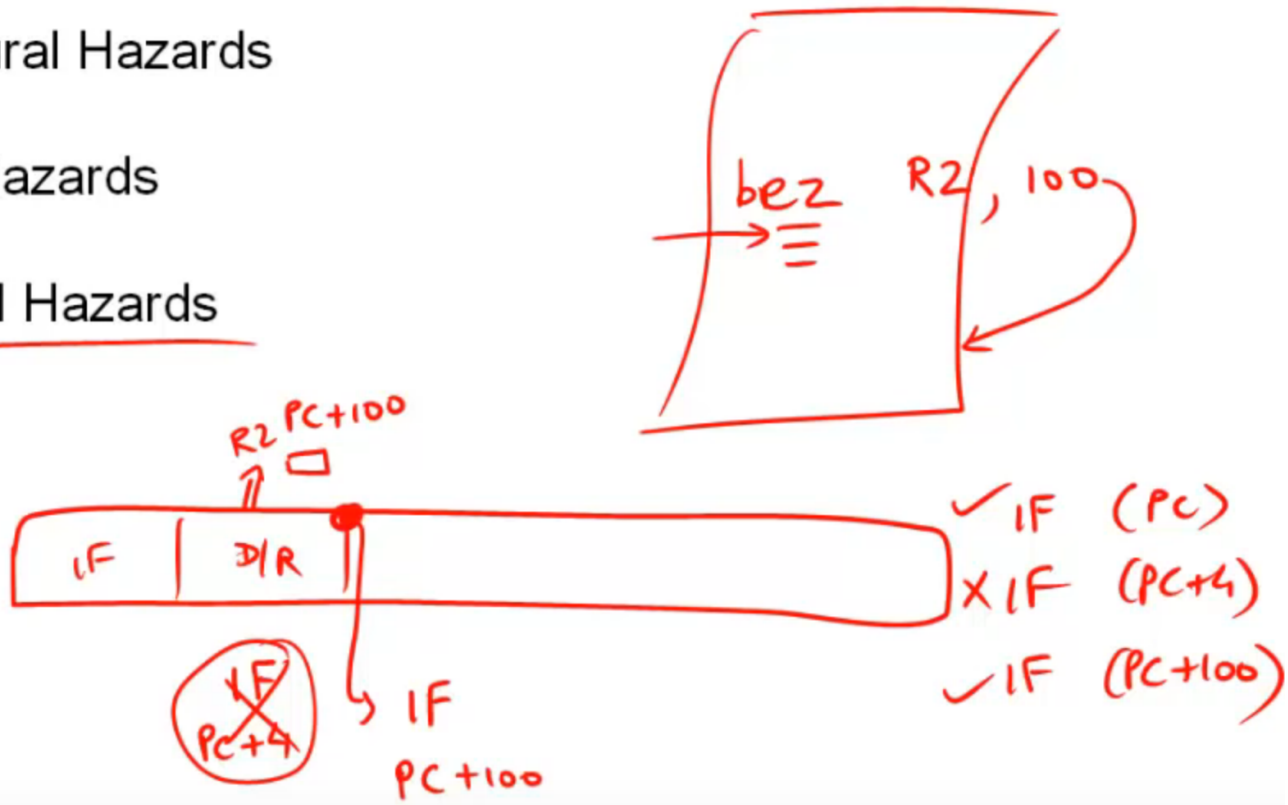
- The following pairs of instructions will have intermediate stalls:

  lw R1, 8(R2) → P.O.P — 4th st
  add/sub/lw R3, R1, R4 or sw R3, 8(R1) → P.O.C — 3rd st — 1 stall

  fmul F1, F2, F3
  fadd F5, F1, F4

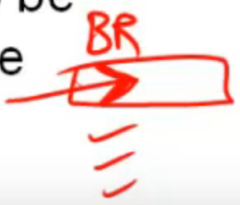
Hazards

- Structural Hazards
- Data Hazards
- Control Hazards



Control Hazards

- Simple techniques to handle control hazard stalls:
  - for every branch, introduce a stall cycle (note: every 6th instruction is a branch on average!)
  - assume the branch is not taken and start fetching the next instruction – if the branch is taken, need hardware to cancel the effect of the wrong-path instructions
  - predict the next PC and fetch that instr – if the prediction is wrong, cancel the effect of the wrong-path instructions
  - fetch the next instruction (branch delay slot) and execute it anyway – if the instruction turns out to be on the correct path, useful work was done – if the instruction turns out to be on the wrong path, hopefully program state is not lost



Branch Delay Slots