Project 2

CS 3410

New Instructions (Table B)

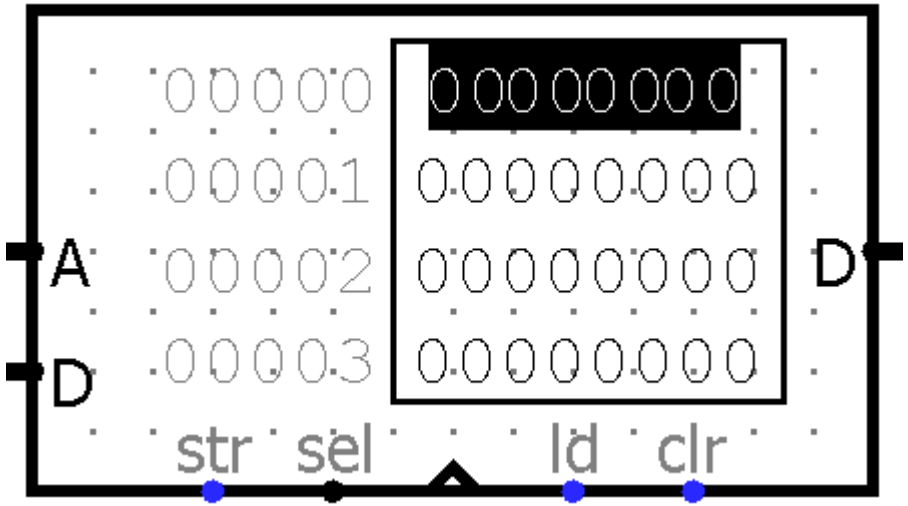
Jumps - J, JR, JAL, JALR

Branches - BEQ, BNE, BLEZ, BGTZ, BLTZ, BGEZ

Memory - LW, LB, LBU, SW, SB

However, Project 2 **is** cumulative.

Logisim RAM


00000
00000000
00001
00000000
A
00002
00000000
D
D
00003
00000000
str
sel
ld
clr

Load-Use Hazard

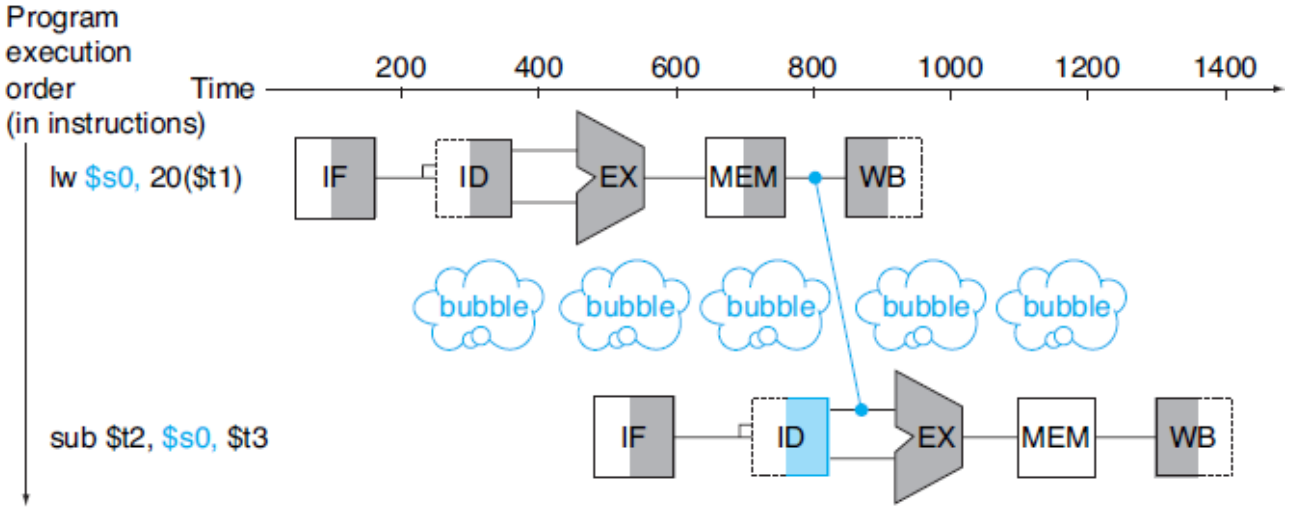


FIGURE 4.30 We need a stall even with forwarding when an R-format instruction following a load tries to use the data. Without the stall, the path from memory access stage output to execution stage input would be going backward in time, which is impossible. This figure is actually a simplification, since we cannot know until after the subtract instruction is fetched and decoded whether or not a stall will be necessary. Section 4.7 shows the details of what really happens in the case of a hazard.

Load-Use Hazard

Load-use Stall = ID/Ex.MemRead &&
(IF/ID.Ra == ID/Ex.Rd || IF/ID.Rb == ID/Ex.Rd)

Delay Slot

Branch and Jump instructions have 1 delay slot.

Control Hazards

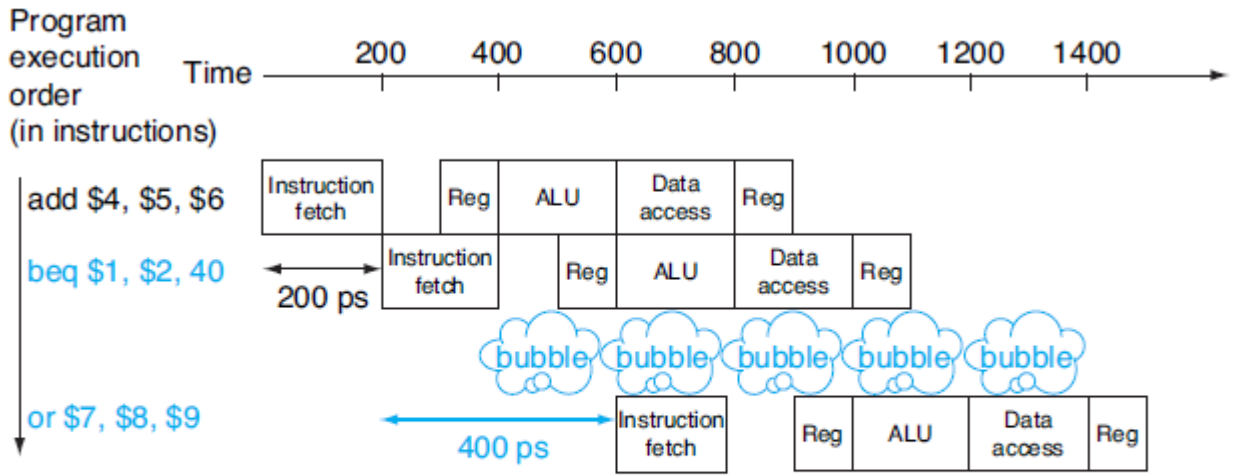


FIGURE 4.31 Pipeline showing stalling on every conditional branch as solution to control hazards. This example assumes the conditional branch is taken, and the instruction at the destination of the branch is the OR instruction. There is a one-stage pipeline stall, or bubble, after the branch. In reality, the process of creating a stall is slightly more complicated

Control Hazards

“Easy” Solution:

Stall

More Complex Solution (Not required):

Branch Prediction + Zap/Flush

Data Hazards

Data hazards can occur on branches and jumps!

Relative vs. Absolute PC

Branches use relative addressing.

Ex: Array indices, A[0], A[1].....A[N].

Jumps use absolute addresses.

Ex: Goto statements

Hailstone Programs

Three hailstone MIPS programs required.

An exercise of converting from C to MIPS.

In these you must follow calling conventions.