Chapter 4

The Processor

Adapted and Supplemented by,
Dr. R. Shathanaa

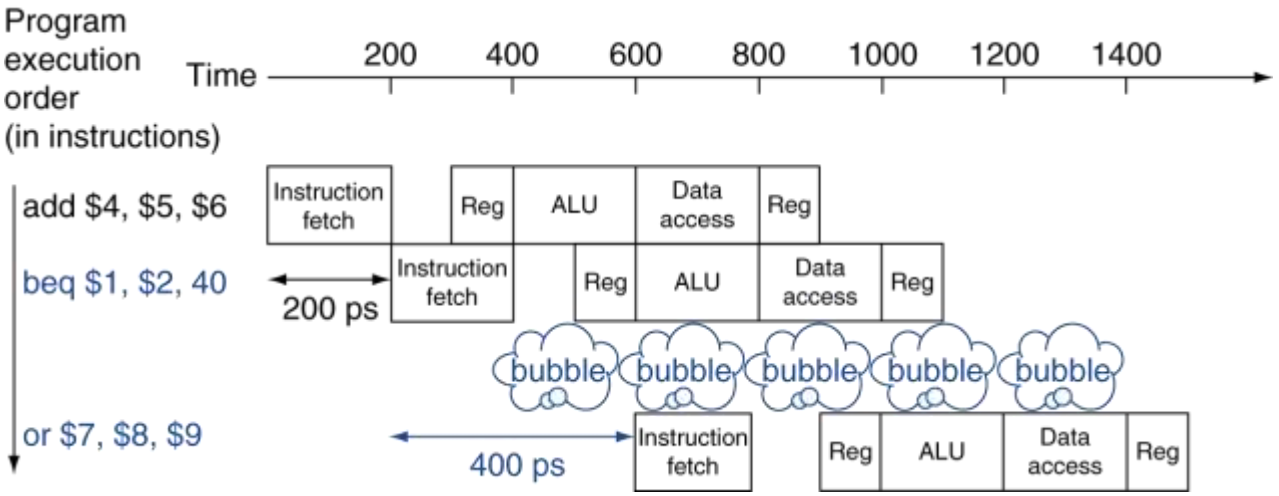
Control Hazards

- Branch determines flow of control
  - Fetching next instruction depends on branch outcome
  - Pipeline can’t always fetch correct instruction
    - Still working on ID stage of branch
- In MIPS pipeline
  - Need to compare registers and compute target early in the pipeline
  - Add hardware to do it in ID stage

Stall on Branch

- Wait until branch outcome determined (during Decode stage) before fetching next instruction

```
            add $4,$5,$6
            beq $1,$2,40
            lw $3,300($0)
40:         or $7,$8,$9
```



Performance of "Stall on Branch"

- Estimate the impact on the *clock cycles per instruction* (CPI) of stalling on branches. Assume all other instructions have a CPI of 1 and branches occur 17% of the time.
- Solution
  - Since the other instructions run have a CPI of 1 and branches occur 17% of time, they will incur latency only for that 17% of time
  - new CPI = old CPI + Branch penalty

    = 1 + (0.17 x 1 cycle)

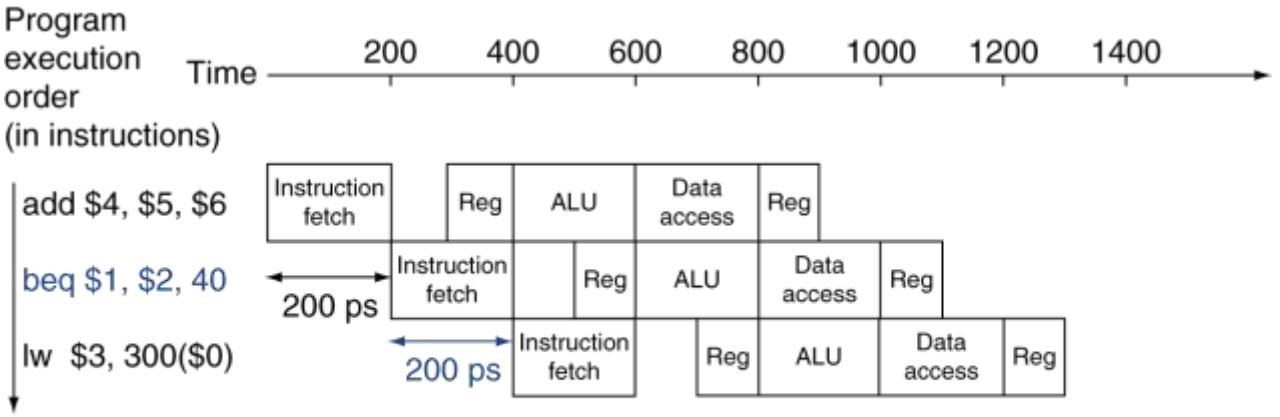
    = 1.17

Branch Prediction

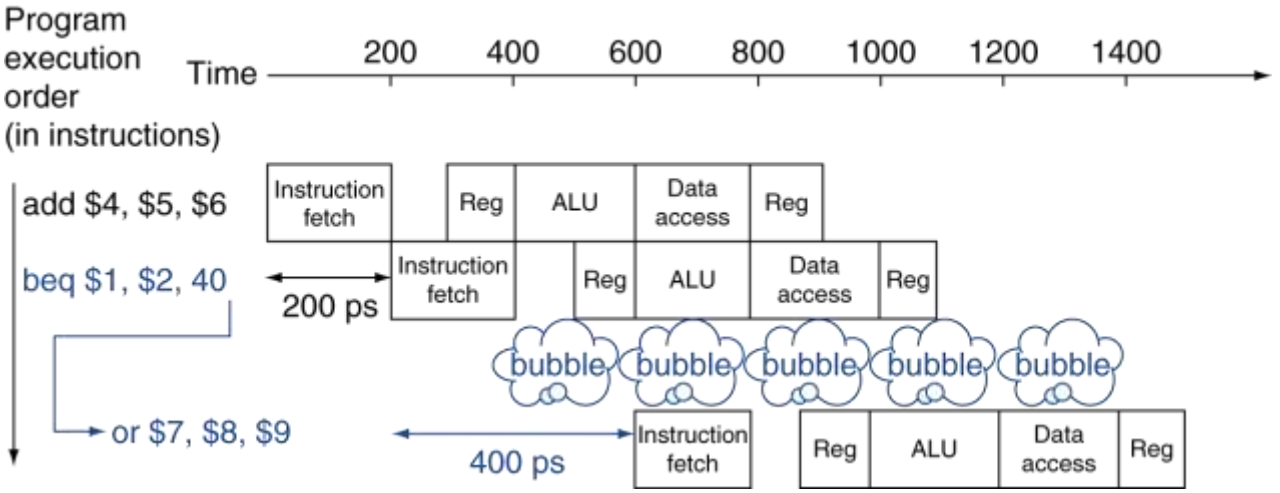
- Longer pipelines can't readily determine branch outcome early
  - Stall penalty becomes unacceptable
- Predict outcome of branch
  - Only stall if prediction is wrong
- In MIPS pipeline
  - Can predict branches not taken
  - Fetch instruction after branch, with no delay

MIPS with Predict Not Taken

Prediction correct



Prediction incorrect



More-Realistic Branch Prediction

- Static branch prediction
  - Based on typical branch behavior
  - Example: loop and if-statement branches
    - Predict backward branches taken
    - Predict forward branches not taken
- Dynamic branch prediction
  - Hardware measures actual branch behavior
    - e.g., **record recent history** of each branch
  - Assume future behavior will continue the trend
    - When wrong, stall while re-fetching, and update history

Pipeline Summary

The BIG Picture

- Pipelining improves performance by increasing instruction throughput
  - Executes multiple instructions in parallel
  - Each instruction has the same latency
- Subject to hazards
  - Structure, data, control
- Instruction set design affects complexity of pipeline implementation