# Comp4611 Tutorial 4

## Pipelining and Control Hazards

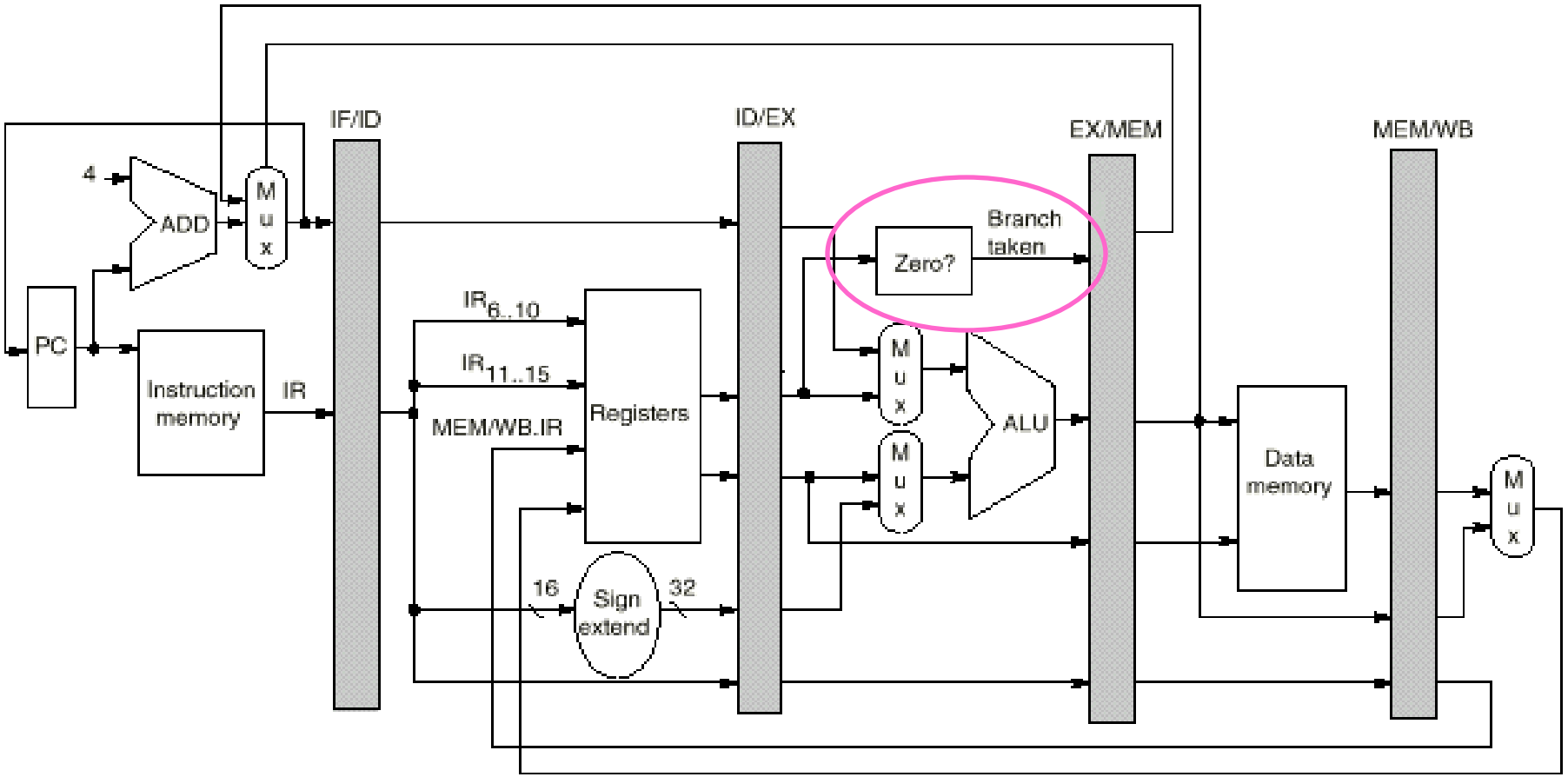
Oct. 8, 10, 12 2012

# Outline

1. Branch Hazard
   - Modified Pipelined MIPS Datapath
   - 4 Reducing Branch Penalty Alternatives
   - Dynamic Branch Prediction

2. Exercise on Pipeline Time Chart

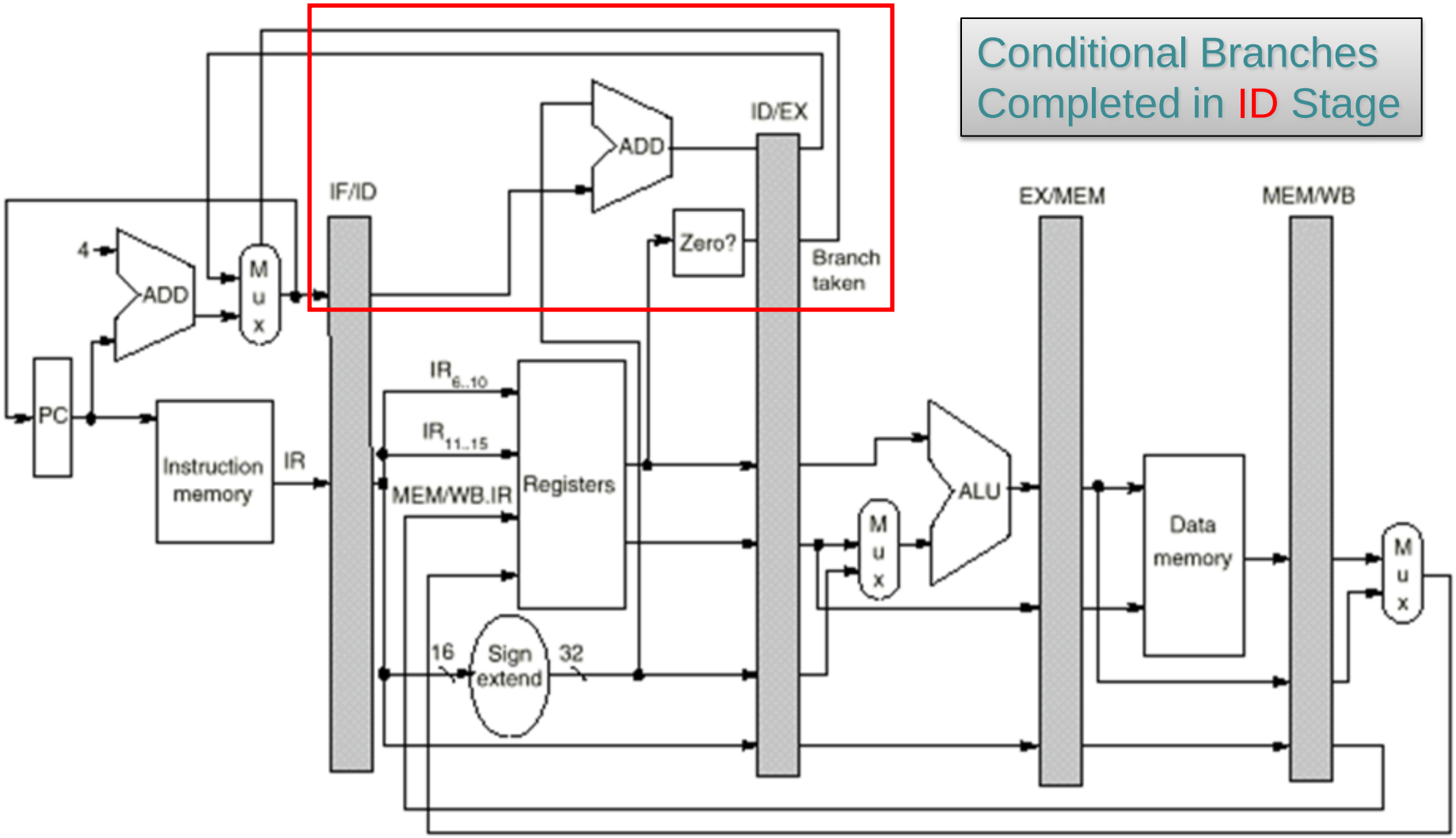
# Pipelined MIPS Datapath




**Conditional Branches:**

**beq R1,R2,100** **if (R1 == R2) go to PC+4+100**

# Modified Pipelined MIPS Datapath with Early Branch Determination



# Reducing Branch Penalty

1. Stall until branch direction is clear
   - Move Zero test to ID stage
   - Adder to calculate new PC in ID stage
   - Less wasted clock cycles

     ➔ Modified pipelined MIPS still need one stall

| Branch | IF | **ID** | EX | MEM | WB | | |
|---|---|---|---|---|---|---|---|
| Next | | **IF(stall)** | IF | ID | EX | MEM | WB |

# Reducing Branch Penalty

## 2. Predict Branch Not Taken

- Immediately execute the next instruction
- Execute successor instructions in sequence
- "Squash" instructions in pipeline if branch actually taken
- Advantage of late pipeline state update
- 47% MIPS branches not taken on average
- PC+4 already calculated, so use it to get next instruction (each instruction take 4 bytes)
- 0 clock cycle wasted when branch not taken
- 1 clock cycle wasted when branch taken

# Reducing Branch Penalty

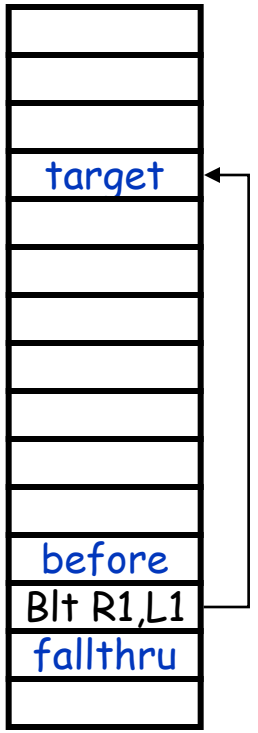
3. Predict Branch Taken
   - 53% MIPS branches taken on average
   - But haven't calculated branch target address in MIPS

     ➔ MIPS still incurs 1 cycle branch penalty
   - Predict-taken scheme makes sense only in some machines where the target address of branch is known before the branch outcome

# Reducing Branch Penalty

## 4. Delayed Branch

- The first instruction following the branch is ALWAYS executed
- Compiler can figure out what to put there
- Where to get instructions to fill branch delay slot?
  - Before branch instruction
  - From the target address: only valuable when branch taken
  - From fall through: only valuable when branch not taken

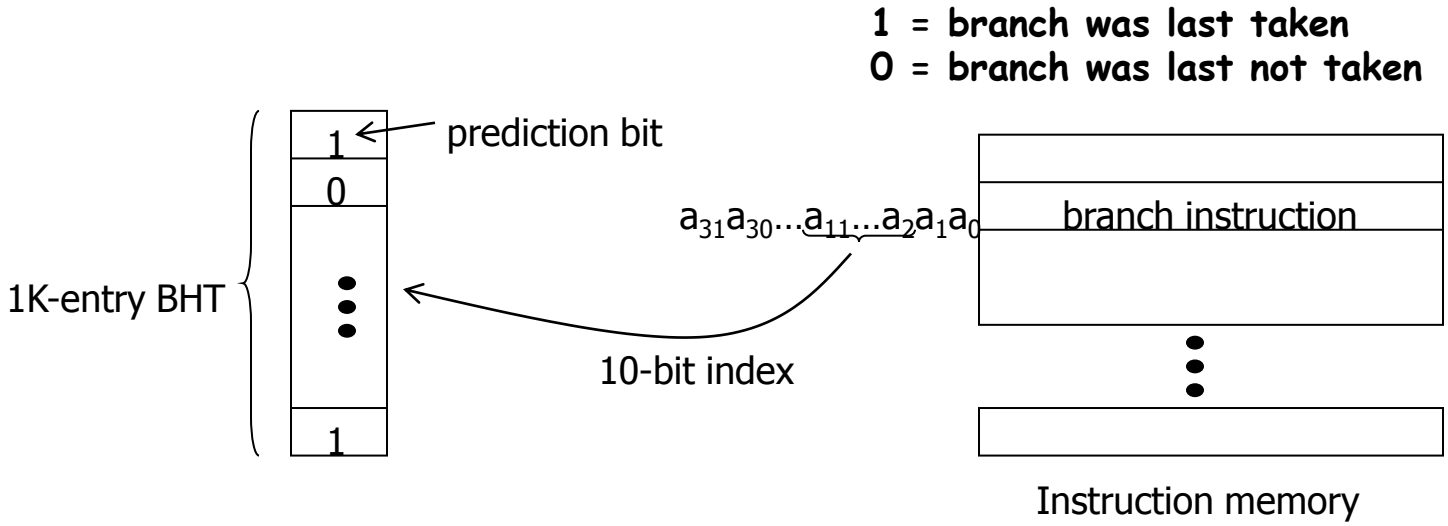


# Dynamic Branch Prediction

- Builds on the premise that history matters
  - Observe the behavior of branches in previous instances and try to predict future branch behavior
  - Try to predict the outcome of a branch early on in order to avoid stalls
  - Branch prediction is critical for multiple issue processors
    - In an n-issue processor, branches will come n times faster than a single issue processor

# 1-Bit Branch Prediction

- Branch History Table (BHT): Lower bits of PC address index table of 1-bit values
  - Says whether or not the branch was taken last time
  - No address check (saves HW, but may not be the right branch)
  - If prediction is wrong, invert prediction bit




*Hypothesis: branch will do the same again.*

# 1-Bit Branch Prediction

**Example:**

Consider a loop branch that is taken 9 times in a row and then not taken once. What is the prediction accuracy of the 1-bit predictor for this branch assuming only this branch ever changes its corresponding prediction bit?

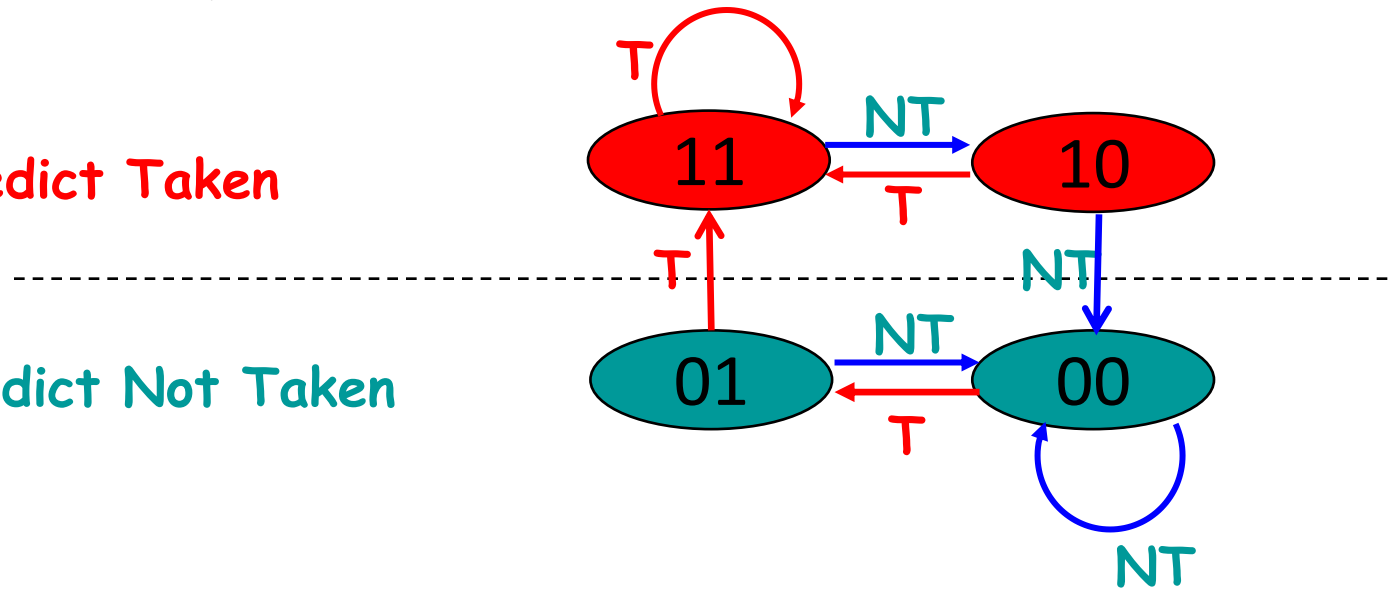
**Answer:**

*80%.* Because there are two mis-predictions;

one on the first iteration and one on the last iteration.

# 2-bit Predictors

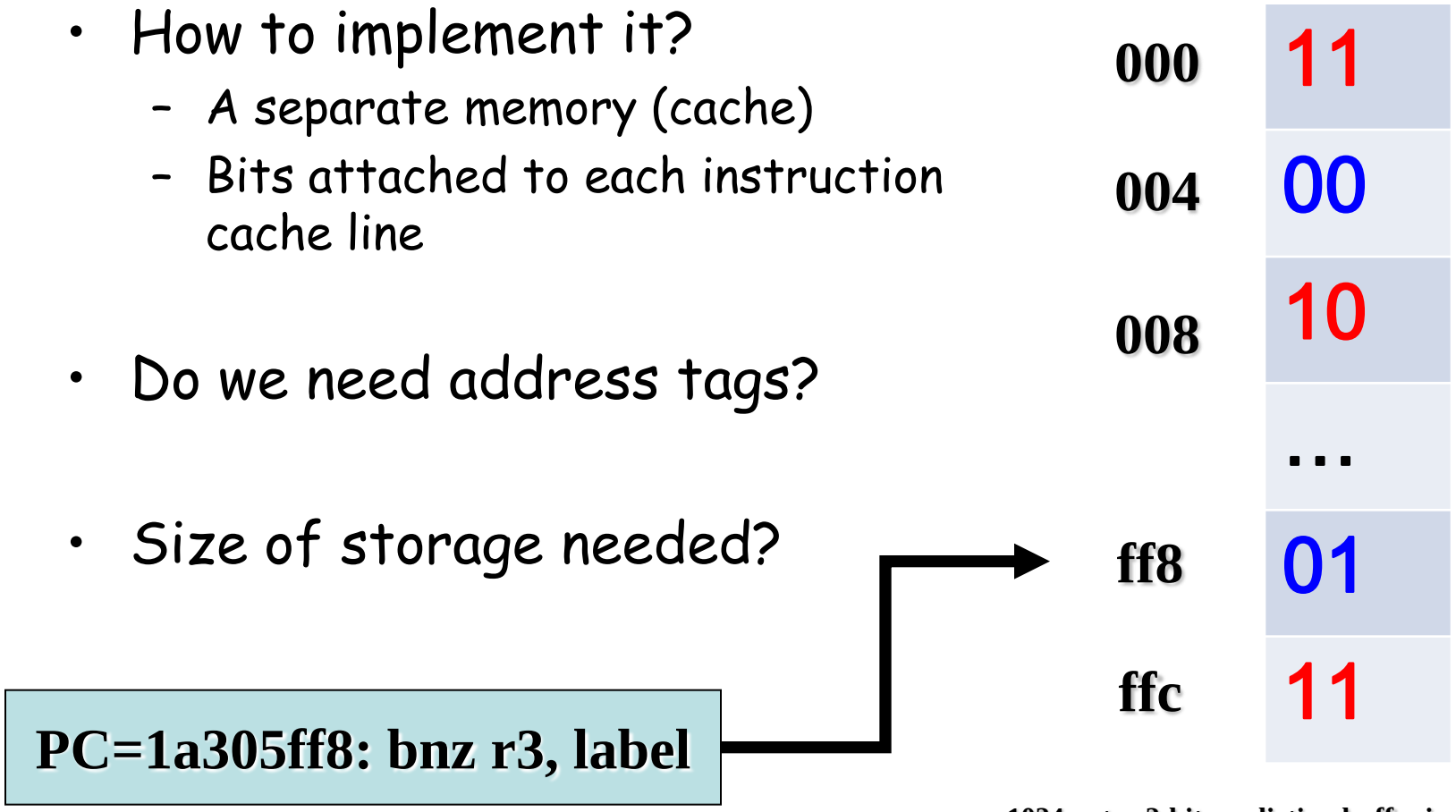
- 2-bit scheme ➔ 2 mis-predictions change the prediction




- Greatly improve the prediction performance

# 2-bit Prediction Buffer

- How to implement it?
  - A separate memory (cache)
  - Bits attached to each instruction cache line
- Do we need address tags?
- Size of storage needed?




1024-entry 2-bit prediction buffer indexed by 10 bits ($12^{th}$ — $3^{rd}$ least significant bits)

# Pipeline Time Chart Exercise

```
Loop:   LD    R1, 0(R2)
        DADDI R1, R1, #1
        SD    0(R2), R1
        DADDI R2, R2, #4   ; R2 = R2 + 4
        DSUB  R4, R3, R2   ; R4 = R3 - R2
        BNEZ  R4, Loop
```

Given: 1. Initially R3 = R2 + 396, all stages take 1 cycle each.
2. Register can perform read/write in same cycle.
3. With Modified MIPS Datapath

(a) Draw pipeline timing chart without forwarding. Branch handled by stalling pipeline. How many cycles does the loop take?

(b) Draw pipeline timing chart with forwarding. Branch handled by predicted not taken. How many cycles does the loop take?

# Answer

- For Modified MIPS Pipeline, Conditional Branch will be resolved in ID stage. (S - Stall, r - branch result ready)

- Total No. of Loops = 396/4 = 99
- (a) Total cycles = 15 x 99 = 1485

| | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 | 19 | 20 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| LD R1, 0(R2) | F | D | E | M | W | | | | | | | | | | | | | | | |
| DADDI R1, R1, #1 | | F | D | S | S | E | M | W | | | | | | | | | | | | |
| SD 0(R2), R1 | | | F | S | S | D | S | S | E | M | W | | | | | | | | | |
| DADDI R2, R2, #4 | | | | | | F | S | S | D | E | M | W | | | | | | | | |
| DSUB R4, R3, R2 | | | | | | | | | F | D | S | S | E | M | W | | | | | |
| BNEZ R4, Loop | | | | | | | | | | F | S | S | D | S | r | | | | | |
| LD R1, 0(R2) | | | | | | | | | | | | | F | S | S | F | D | E | M | W |

# Answer (cont.)

- (b) Total cycles = 9 x 98 + 10 = 892

| | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| LD R1, 0(R2) | F | D | E | M | W | | | | | | | | | |
| DADDI R1, R1, #1 | | F | D | S | E | M | W | | | | | | | |
| SD 0(R2), R1 | | | F | S | D | E | M | W | | | | | | |
| DADDI R2, R2, #4 | | | | | F | D | E | M | W | | | | | |
| DSUB R4, R3, R2 | | | | | | F | D | E | M | W | | | | |
| BNEZ R4, Loop | | | | | | | F | D | r | | | | | |
| LD R1, 0(R2) | | | | | | | | F | D | E | M | W | | |
| | | | | | | | | F | D | F | D | E | M | W |

Prediction Correct

Prediction incorrect