# DSD Final Project Presentation

GROUP 8

MEMBER: 陳柏帆 洪鈺萌 林志皓

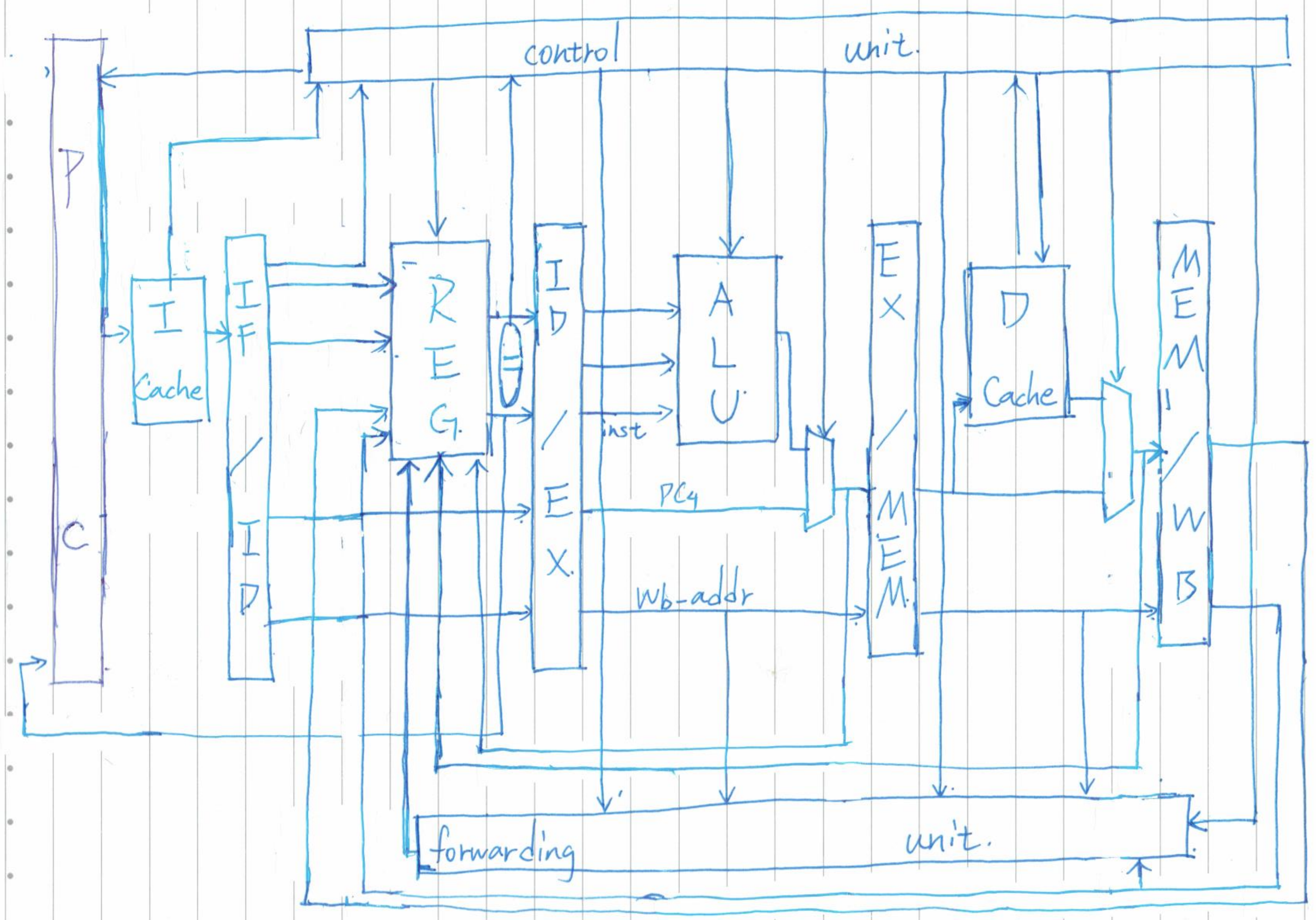
# Contents

- MIPS pipeline (baseline)
- Cache design
- Extention1 -- Branch Prediction
- Extention2 -- Tow Level Cache
- Extention3 -- Multiplication & Division

# 1.

# MIPS Pipeline

# structure

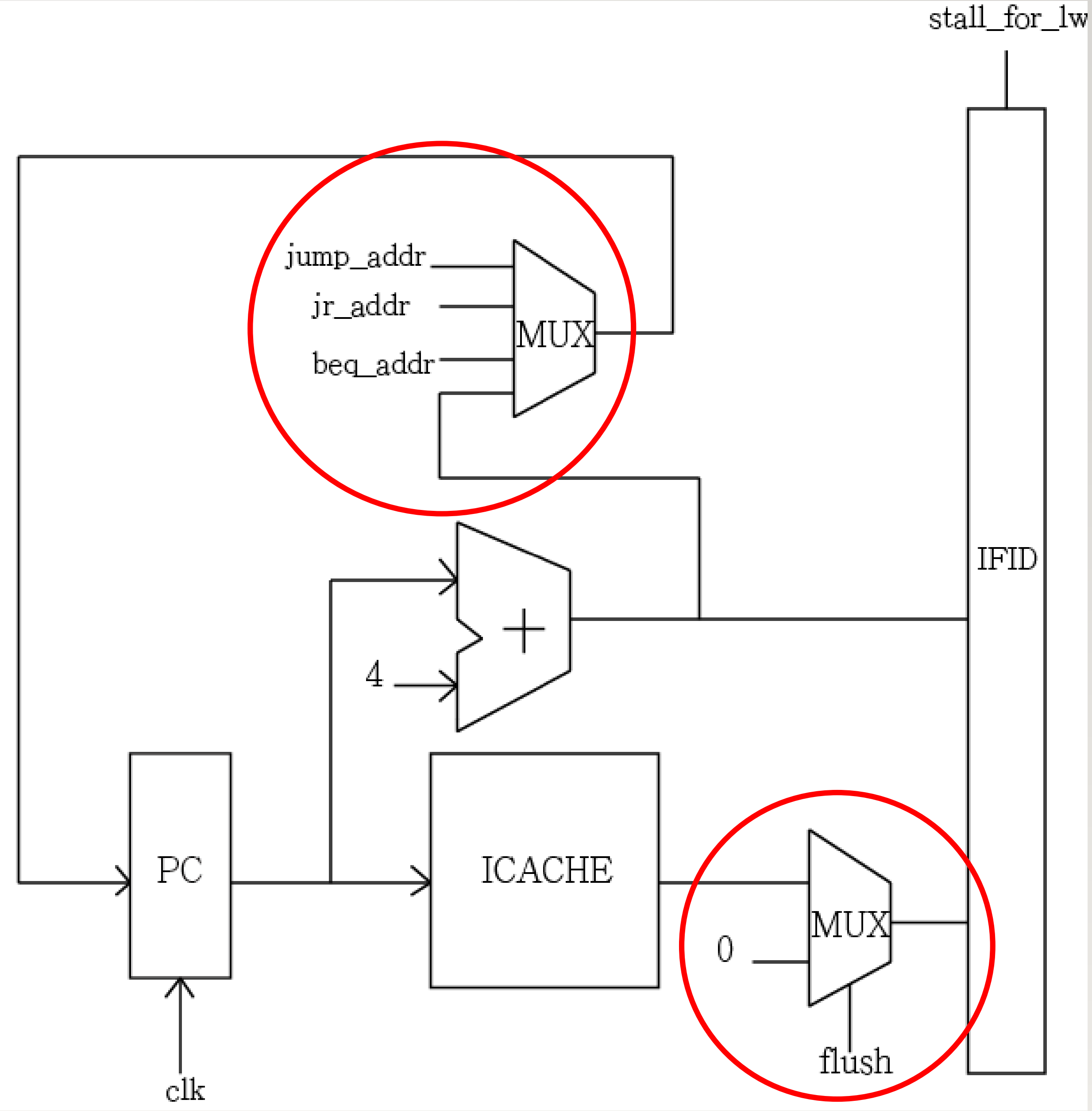


Hand-drawn by
柏帆

# Instruction Fetch

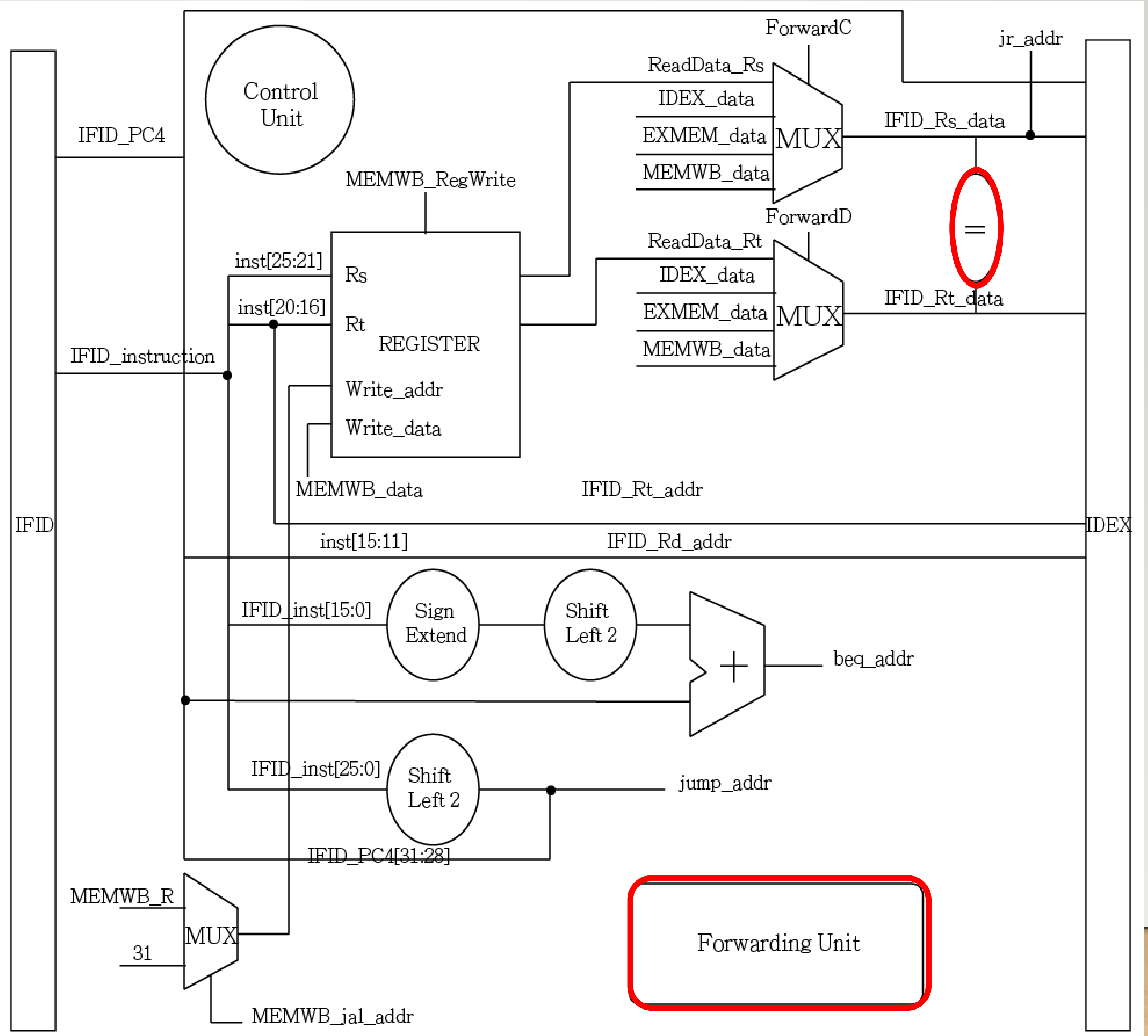
- Select next Instruction address
- (jump/ branch/ PC+4)
- When next address is not PC+4 , we set a "FLUSH"

(32'b0 for instruction)



# Instruction Decode

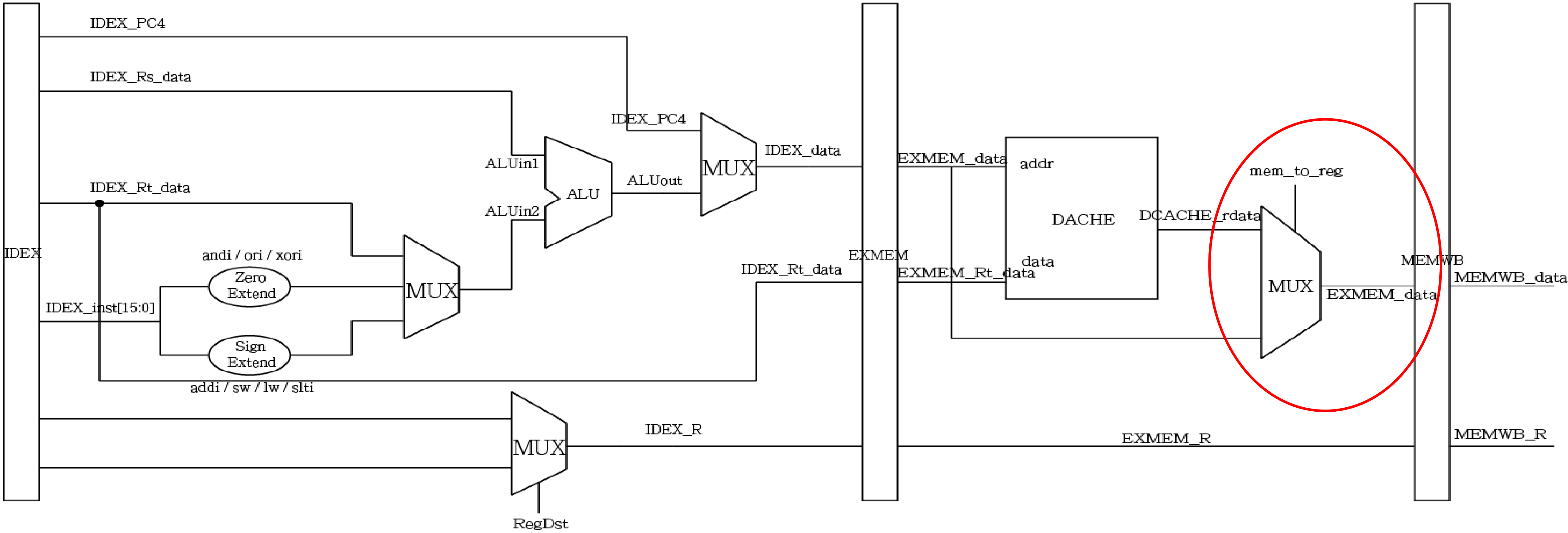
- Control unit
- Forwarding Unit
- Determine whether Branch or not



# EX, MEM, WB

- Select the data to be forward in MEM stage (from ALU or memory)

→ Simplified forwarding unit



# 2.

# Cache

# Structure

- Direct-map( 4 words * 8 block)
- Write-back
- Valid bit (no dirty bit)

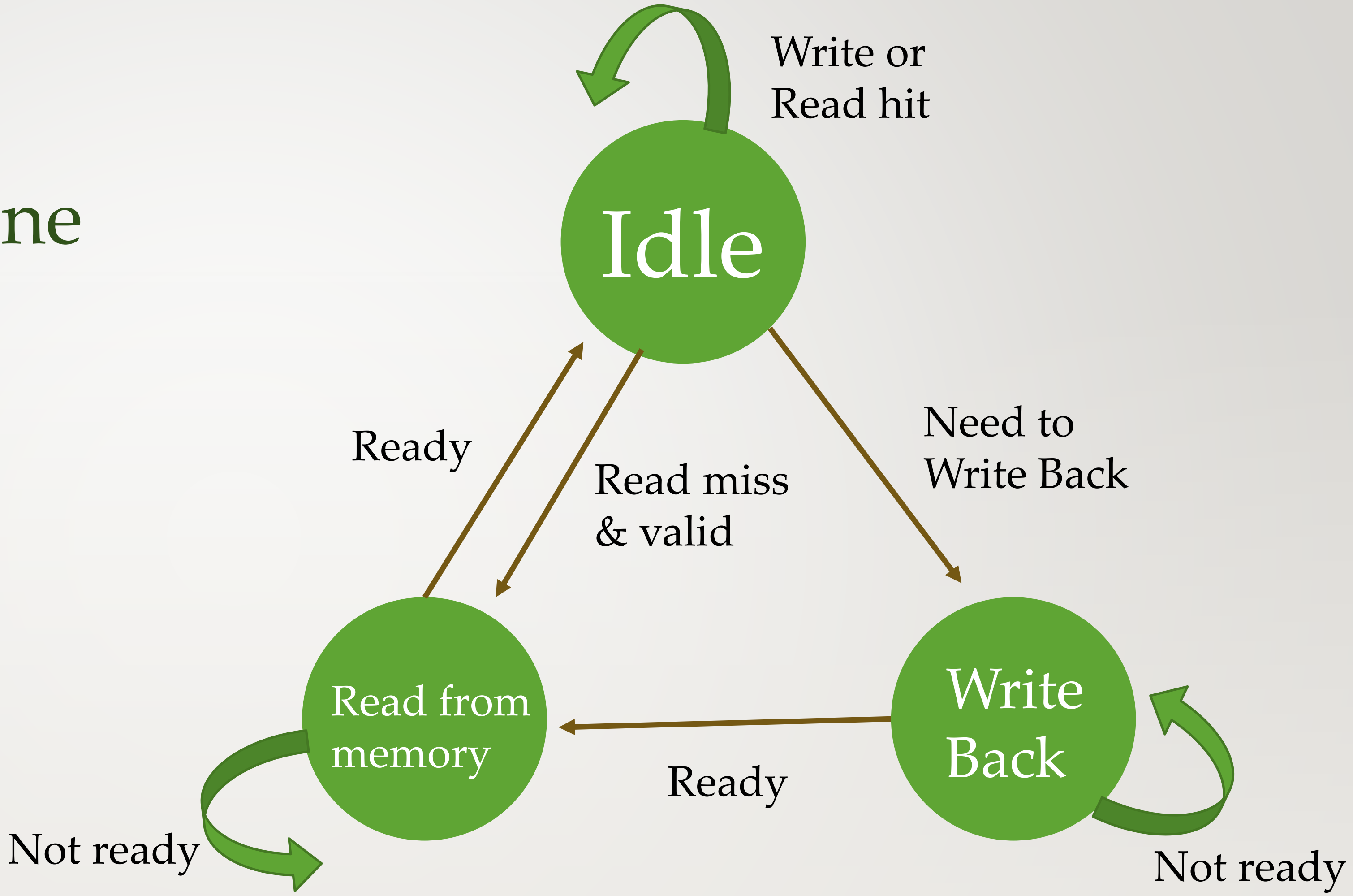
| Valid bit | Tag | DATA 1 | DATA 2 | DATA 3 | DATA 4 |
|---|---|---|---|---|---|
| 1 bit | 25 bits | 32 bits | 32 bits | 32 bits | 32 bits |

A block

# Finite State Machine

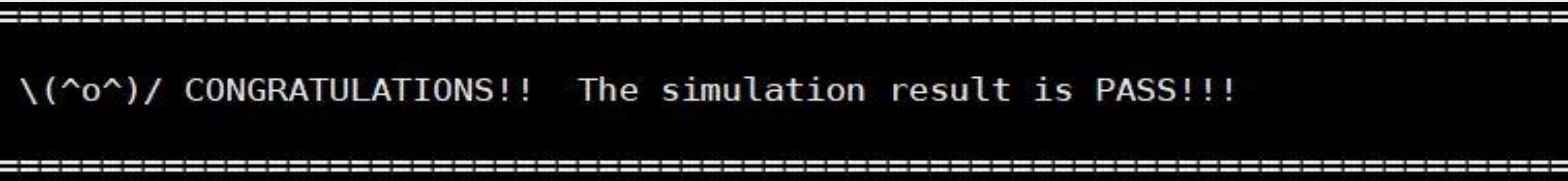
- Only 3 states !
- If hit, processor can get data from cache in the cycle of request

→ **Reduce total cycle**



# Baseline Performance

- Cell Area : 284158
- Execution Time : 9960.75 (ns)
- AT rate : **2830426799**



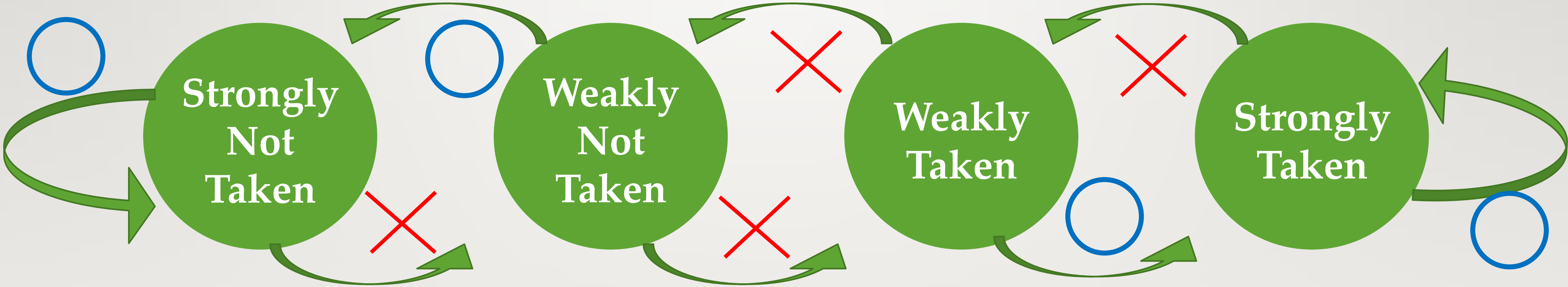
```
========================================================================================

\(^o^)/ CONGRATULATIONS!!  The simulation result is PASS!!!

========================================================================================
```

# 3.

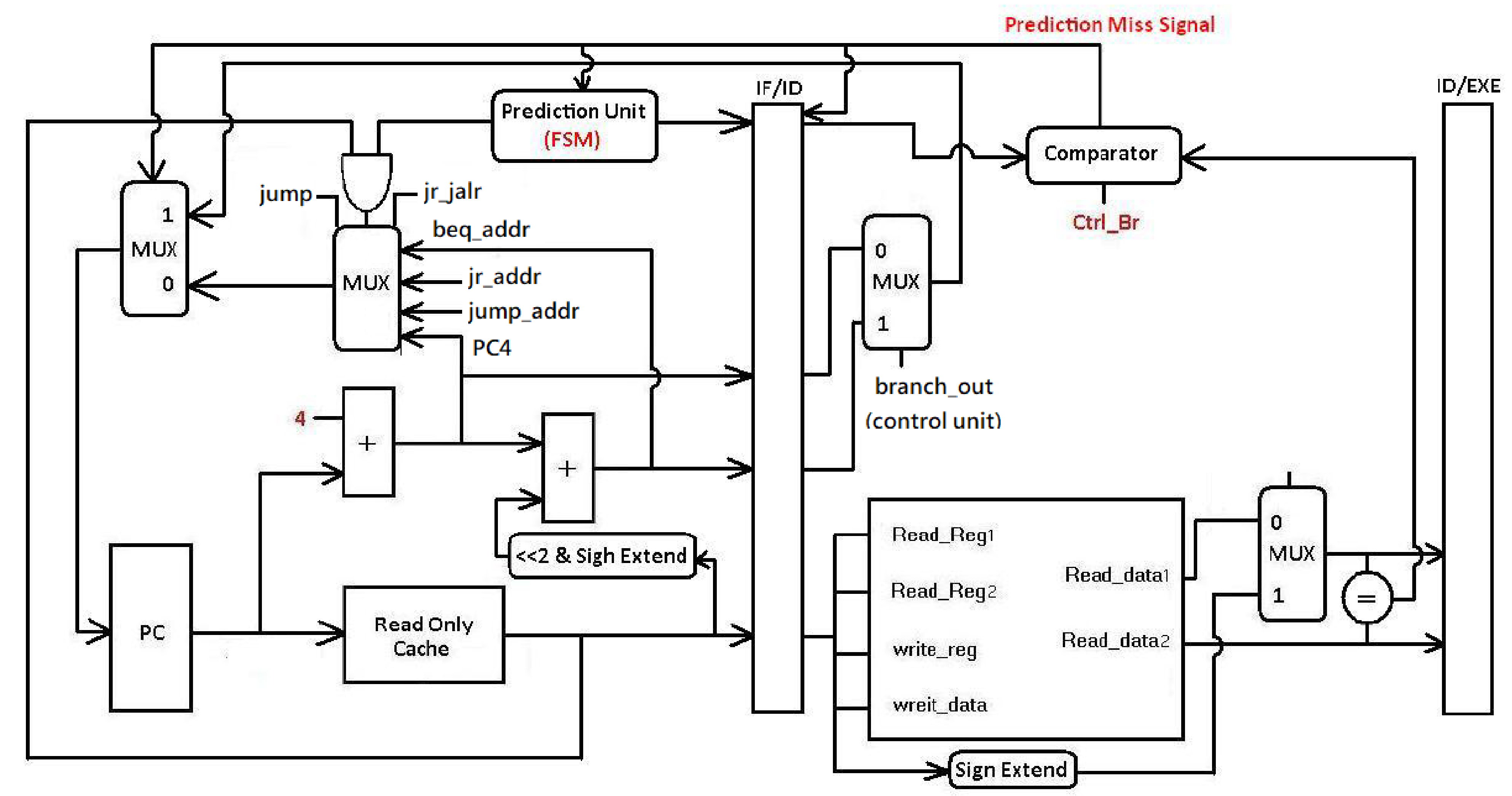
# Extention – Branch Prediction

# Finite State Machine

- Initially, the IDLE state is Strongly Not Taken
- If prediction wrongly two times in a row,  the state will go to Taken

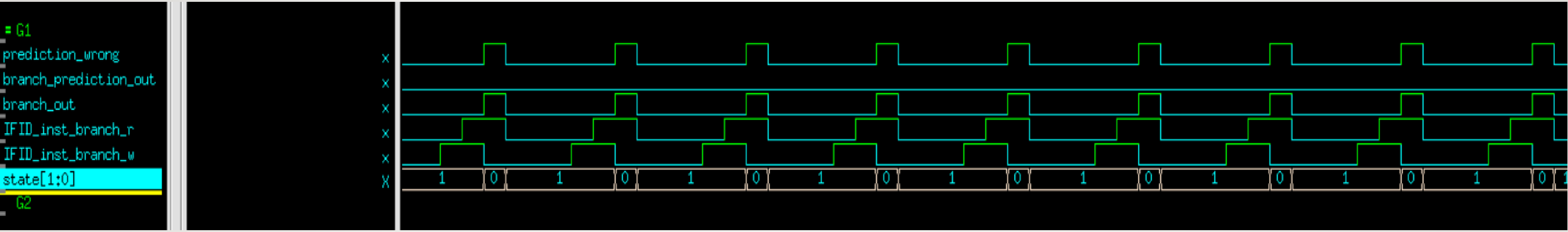


# Overall Circuit



# Result

- Cannot predict two successive branch instructions effectively



- 1234568

# 4.

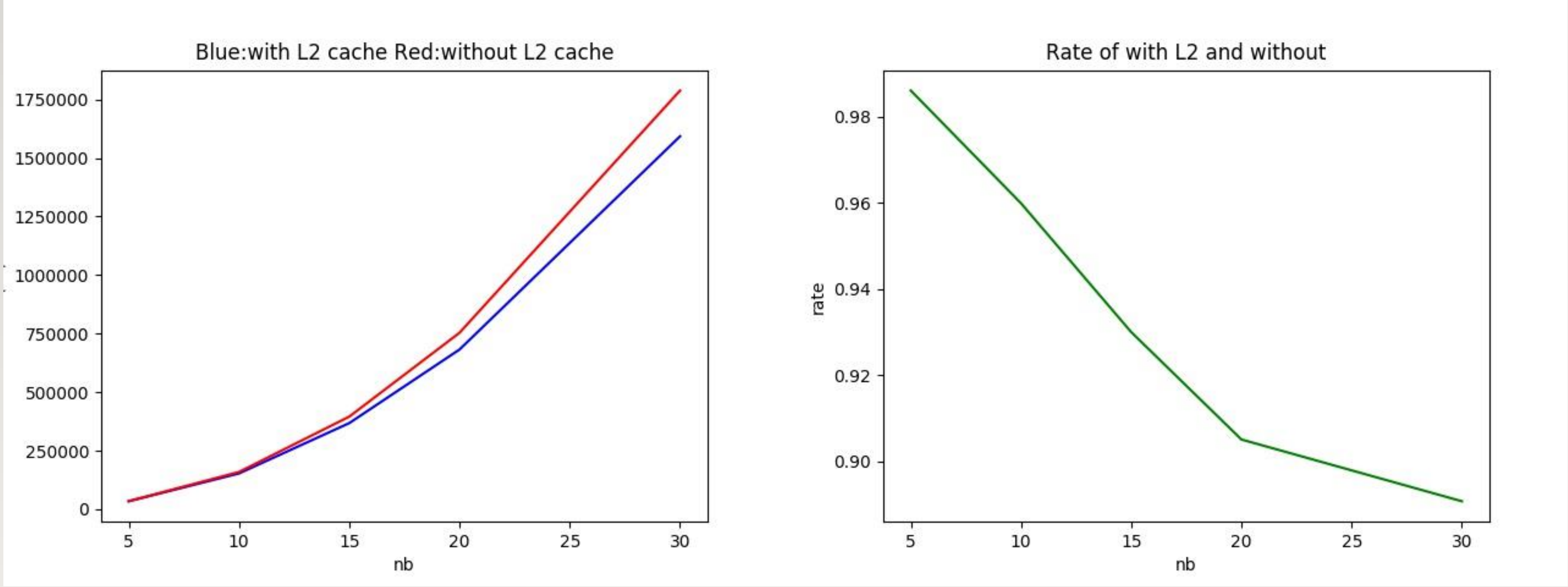
# Extention –
# Two Level Cache

# Implementation Details

- Separate 2 L2 caches for I cache & D cache
- Direct Map
- 4 words/block, total 64 blocks(entries)
- FSM: 3 state

| Valid | Tag | Data |
|---|---|---|
| 1 bit | 22 bits | 128 bits |

# Result


Blue:with L2 cache Red:without L2 cache
1750000
1500000
1250000
1000000
750000
500000
250000
0
5
10
15
20
25
30
nb
Rate of with L2 and without
rate
0.98
0.96
0.94
0.92
0.90
5
10
15
20
25
30
nb

# 5.

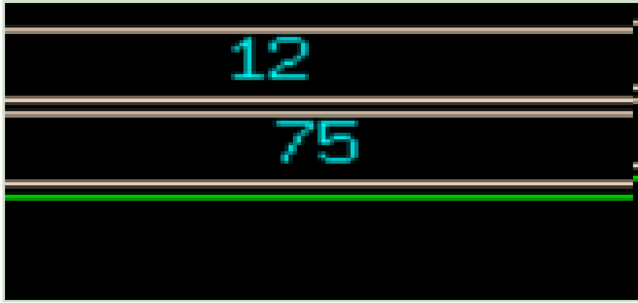
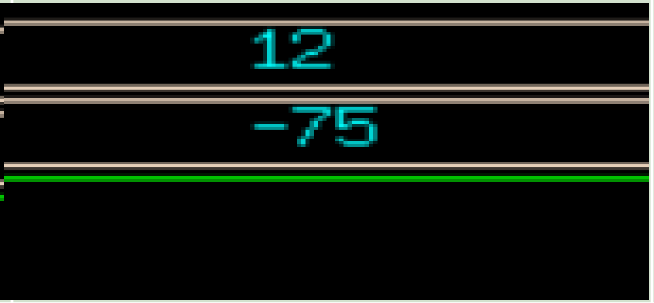
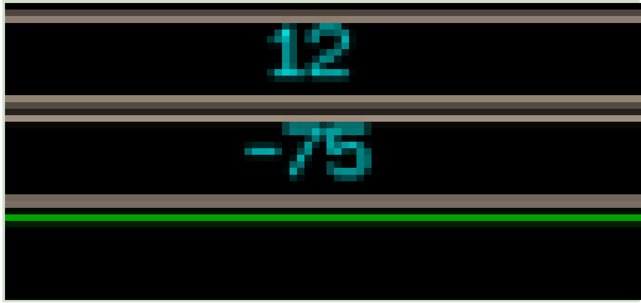
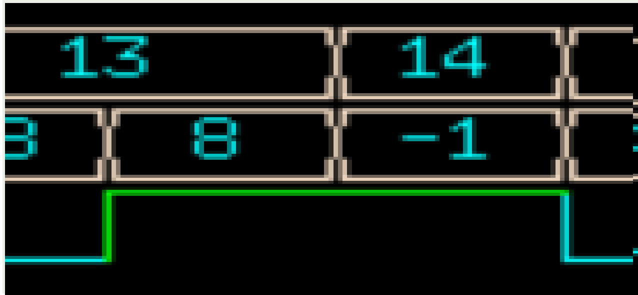
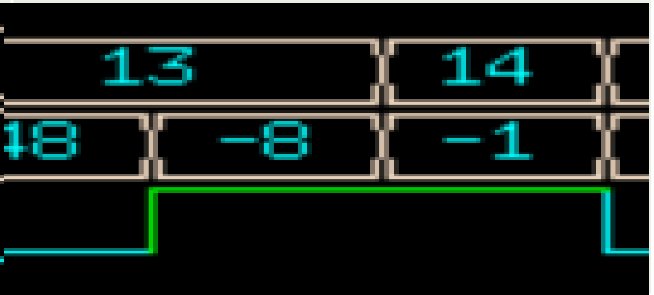
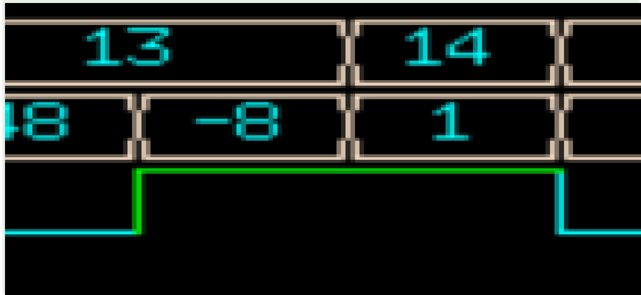
# Extention – Multiplication & Division

# Implementation Details

- **Iterative**, stall 32 cycles for each Multiplication/Division Instruction
- Record the position of Add/Subtract with **Counter**,
- Thus determine when to stop "stalling"

- **Correctness Verification** :
- Check data in D cache with monitoring nWave
- Add some negative number for testing

# Result

| Numbers to be Multiplication/Division | -25<br>-3 | -25<br>3 | 25<br>-3 |
|---|---|---|---|
| Multiplication |  12<br>75 |  12<br>-75 |  12<br>-75 |
| Division |  13<br>14<br>3<br>8<br>-1 |  13<br>14<br>48<br>-8<br>-1 |  13<br>14<br>48<br>-8<br>1 |