DESIGN SPECIFICATION – MIPS 32-BIT PIPELINE CPU

1. Feature List

- Kiến trúc MIPS 32-bit, hỗ trợ định dạng R, I, J (các tập lệnh cơ bản).
- Pipeline 5-stage: IF, ID, EX, MEM, WB, in-order.
- Data Hazard Handling: Forwarding (EX/MEM → EX, MEM/WB → EX) và Hazard Detection Unit (stall 1 chu kỳ cho load-use).
- Control Hazard Handling: Branch Prediction Unit gồm BHT 2-bit (1024 entries) + Branch Target Buffer (1024 entries); flush pipeline khi dự đoán sai.
- Caches: I-Cache 4KB và D-Cache 2KB, direct-mapped, line 32B; hit 1 chu kỳ; miss → stall & refill (write-through cho D-Cache).
- Harvard Architecture: tách bus lệnh và dữ liệu để tránh structural hazard.
- Register File 32×32-bit, 2 read + 1 write ports; $zero bất biến.
- Hardwired Control Unit: giải mã opcode/funct trực tiếp, không dùng microcode.

2. Functional Description

1. Instruction Fetch (IF)

- Thành phần: PC (32-bit), Instruction Memory/Cache, Adder (PC+4).
- Hoạt động:
    - Mỗi chu kỳ, nạp lệnh tại địa chỉ PC.
    - Tính trước PC+4 cho luồng tuần tự.
    - Nếu Branch Predictor báo Taken → PC được gán bằng địa chỉ trong BTB.
- Xử lý đặc biệt:
    - Stall: giữ nguyên PC và IF/ID khi hazard.
    - Flush: nếu dự đoán nhánh sai, PC được ghi đè địa chỉ đúng và IF/ID bị xóa.

2. Instruction Decode (ID)

- Thành phần: Instruction Decoder, Register File (32×32-bit), Sign-extend, Hazard Detection Unit.
- Hoạt động:
    - Giải mã opcode, funct, rs, rt, rd, immediate.
    - Tạo tín hiệu điều khiển (ALUOp, MemRead, MemWrite, RegWrite,…).
    - Đọc 2 thanh ghi nguồn qua cổng đọc.

    - Sign-extend immediate (16→32 bit).
- Hazard detection:
    - Phát hiện load-use hazard: nếu lệnh EX là lw và thanh ghi đích trùng với rs/rt ở ID.
    - Khi xảy ra: stall PC và IF/ID, chèn NOP vào ID/EX.

3. Execute (EX)

- Thành phần: ALU 32-bit, Multiplexors, Forwarding Unit.
- Hoạt động:
    - ALU tính toán theo ALUControl.
    - Forwarding Unit:
        - EX/MEM → EX (Forward=10): lấy dữ liệu từ kết quả ở EX/MEM.
        - MEM/WB → EX (Forward=01): lấy dữ liệu từ MEM/WB.
    - Xử lý hầu hết RAW hazard mà không cần stall.
    - Trường hợp đặc biệt: load-use vẫn phải stall.
- Nhánh:
    - ALU so sánh (BEQ/BNE), tính địa chỉ target = PC+4 + offset<<2.

4. Memory Access (MEM)

- Thành phần: Data Cache L1 (2KB, direct-mapped, 64 block × 32B), Cache Controller, Branch Resolution Logic.
- Hoạt động:
    - Load (LW): đọc cache → nếu miss, dừng pipeline chờ nạp từ main memory (64KB).
    - Store (SW): ghi dữ liệu vào cache/bộ nhớ (giả sử write-through).
- Nhánh:
    - Kiểm tra dự đoán nhánh:
        - Đúng → pipeline tiếp tục.
        - Sai → gửi flush đến IF/ID/EX, cập nhật PC = target đúng.

5. Write Back (WB)

- Thành phần: Mux MemToReg, Register File write port.
- Hoạt động:
    - Chọn dữ liệu ghi (ALU result hoặc Data Memory).
    - Ghi vào thanh ghi đích ở nửa sau chu kỳ.
    - Nếu không cần ghi (branch, store) → RegWrite=0.

6. Hazard Control Units

Hazard Detection Unit (ID stage)

- Logic:

```
if (ID/EX.MemRead == 1) and
   ((ID/EX.RegisterRt == IF/ID.RegisterRs) or
    (ID/EX.RegisterRt == IF/ID.RegisterRt))
then stall
```

- Khi xảy ra:
    - Giữ PC, IF/ID.
    - Chèn NOP vào EX.

Forwarding Unit

- So sánh Register Rd ở EX/MEM, MEM/WB với Register Rs/Rt ở ID/EX.
- Quy tắc:
    - Nếu EX/MEM.RegWrite=1 và EX/MEM.Rd = ID/EX.Rs → ForwardA=10.
    - Nếu MEM/WB.RegWrite=1 và MEM/WB.Rd = ID/EX.Rs → ForwardA=01.
    - Tương tự cho Rt → ForwardB.
- 00: lấy từ Register File.

7. Branch Prediction Unit

- Cấu trúc:
    - Branch History Table (BHT): 1024 entry, mỗi entry 2-bit saturating counter.

    - Branch Target Buffer (BTB): lưu (PC, target). Nếu hit và dự đoán Taken → cung cấp target ngay cho IF.
- Cơ chế:
    - Khi nhánh được giải quyết ở EX/MEM → so sánh kết quả thật.
    - Nếu sai: flush pipeline, sửa PC, cập nhật BHT/BTB.
- Ưu điểm:
    - Độ chính xác ~90%.
    - Giảm penalty nhánh: chỉ mất chu kỳ khi dự đoán sai.

8. Cache & Memory System

- Instruction Cache: 4KB, direct-mapped.
- Data Cache: 2KB direct-mapped, 1 cycle hit.
- Miss: pipeline bị stall đến khi nạp block từ main memory.
- Đơn giản hóa: không xét L2/L3 và coherence.

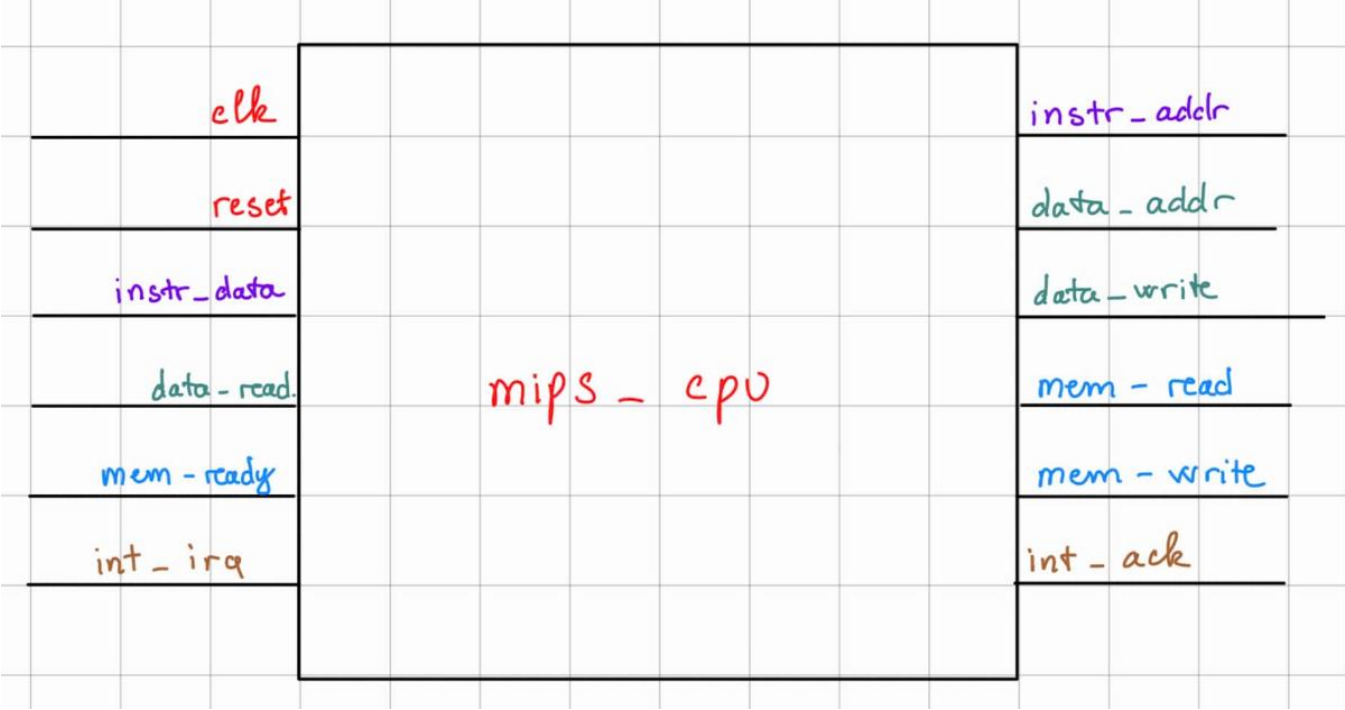
9. Hazard Handling Summary

- RAW hazard: giải quyết bằng Forwarding.
- Load-use hazard: cần 1 cycle stall.
- Control hazard: xử lý bằng Branch Prediction (BHT + BTB).
- Cache miss: pipeline tạm dừng cho đến khi nạp dữ liệu.

3. I/O Ports

1. mips_cpu

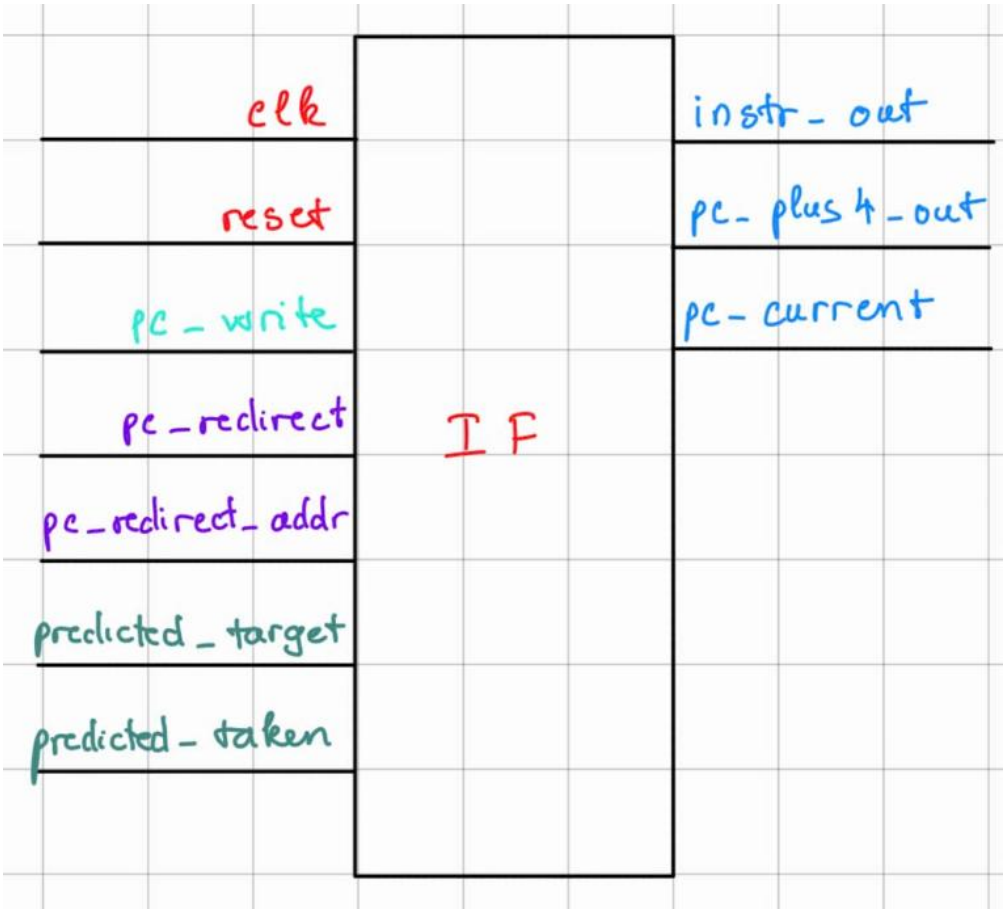
STT	Tên tín hiệu	I/O	Bit	Mô tả
1	clk	Input	1	Xung nhịp CPU
2	reset	Input	1	Tín hiệu reset CPU
3	instr_addr	Output	32	Địa chỉ lệnh hiện tại đến I-CACHE, là giá trị PC dùng để fetch lệnh
4	instr_data	Input	32	Dữ liệu lệnh đọc từ I-CACHE, là lệnh được fetch đưa vào pipeline IF/ID
5	data_addr	Output	32	Địa chỉ đến D-CACHE
6	data_read	Input	32	Dữ liệu đọc từ D-CACHE
7	data_write	Output	32	Dữ liệu cần ghi vào D-CACHE
8	mem_read	Output	1	Tín hiệu yêu cầu đọc đến D-CACHE
9	mem_write	Output	1	TÍn hiệu yêu cầu ghi vào D-CACHE
10	mem_ready	Input	1	Tín hiệu báo hiệu CACHE đã sẵn sàng. Nếu 0 (cache miss -> stall)
11	int_irq	Input	1	Tín hiệu ngắt từ bên ngoài
12	int_ack	Output	1	Tín hiệu xác nhận đã ngắt



2. Instruction Fetch

STT	Tên tín hiệu	I/O	Bit	Mô tả
1	clk	Input	1	Xung nhịp CPU
2	reset	Input	1	Tín hiệu reset CPU
3	pc_write	Input	1	Tín hiệu cho phép cập nhật PC. Nếu 0 -> PC giữ nguyên (stall), nếu 1 -> PC cập nhật
4	pc_redirect	Input	1	Tín hiệu yêu cầu chuyển hướng (prediction sai). Nếu 0 -> giữ nguyên, nếu 1 -> bỏ dự đoán (flush), cập nhật bằng địa chỉ mới.
5	pc_redirect_addr	Input	32	Địa chỉ đích để cập nhật PC khi xảy ra pc_redirect (=1)
6	predicted_target	Input	32	Địa chỉ dự đoán tiếp theo cho PC do "branch prediction unit" cung cấp.
7	predicted_taken	Input	1	Tín hiệu cờ của "branch prediction unit". Nếu 1 -> PC = predicted_target, nếu 0 -> PC = PC + 4

8	instr_out	Output	32	Lệnh fetch từ I-CACHE -> lưu vào thanh ghi pipeline IF/ID
9	pc_plus4_out	Output	32	Giá trị PC + 4 -> lưu vào thanh ghi IF/ID phục vụ branch, jal address
10	pc_current	Output	32	Giá trị PC hiện tại -> gửi đến "branch prediction unit" để tính toán

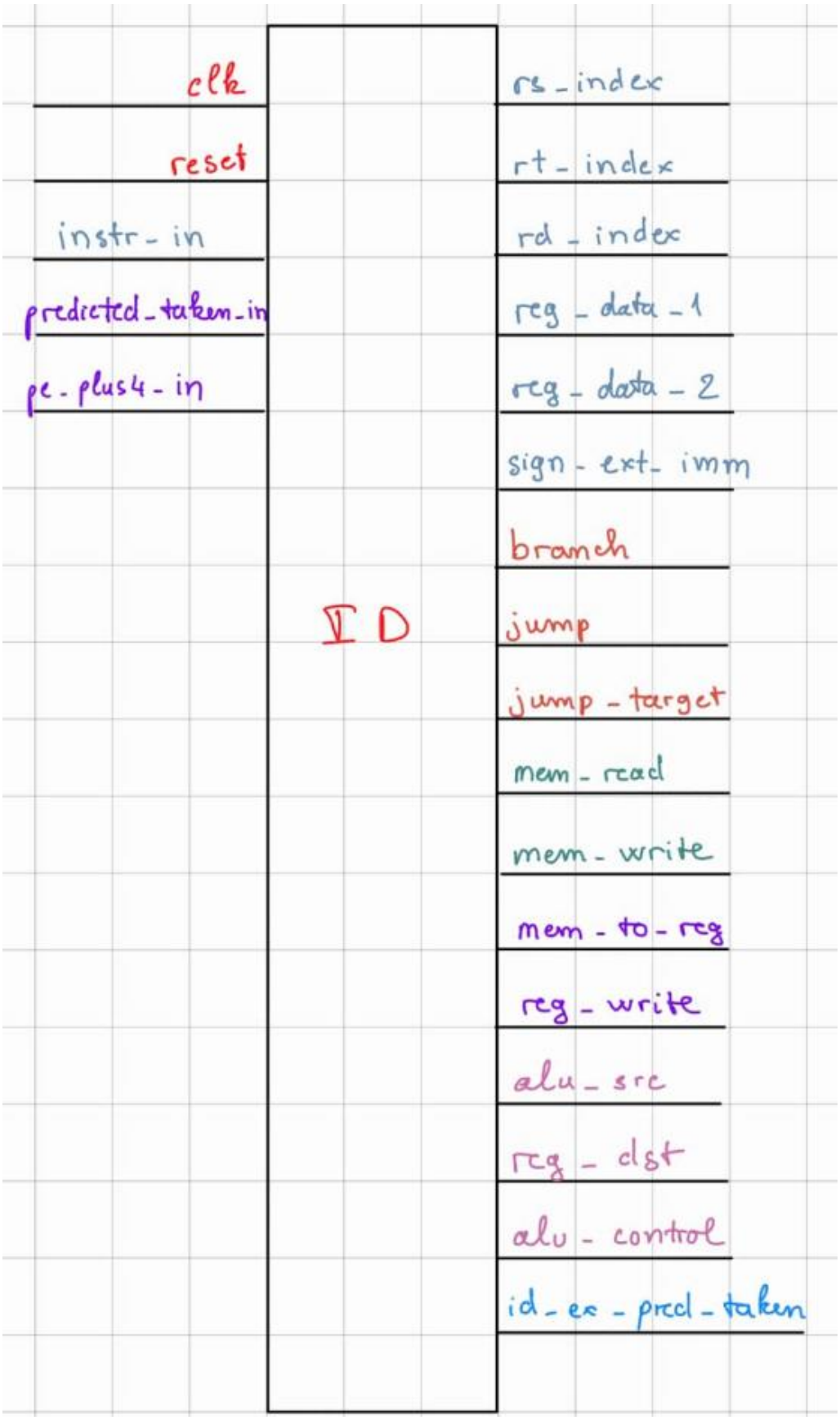


3. Instruction Decode

STT	Tên tín hiệu	I/O	Bit	Mô tả
1	clk	Input	1	Xung nhịp CPU
2	reset	Input	1	Tín hiệu reset CPU
3	instr_in	Input	32	Lệnh cần decode từ pipeline IF/ID

4	pc_plus4_in	Input	32	Giá trị PC + 4 từ pipeline IF/ID
5	predicted_taken_in	Input	32	Cờ dự đoán nhánh do IF chuyển sang (để EX so sánh sau này)
6	reg_data1	Output	32	Thanh ghi Rs được đọc từ Register File
7	reg_data2	Output	32	Thanh ghi Rt được đọc từ Register File
8	rs_index	Output	5	Chỉ số thanh ghi Rs gửi sang Hazard Unit (so sánh sự phụ thuộc)
9	rt_index	Output	5	Chỉ số thanh ghi Rt gửi sang Hazard Unit (so sánh sự phụ thuộc)
10	rd_index	Output	5	Chỉ số thanh ghi Rd
11	sign_ext_imm	Output	32	Giá trị immediate 16 bit được mở rộng dấu lên 32 (dùng cho lệnh I)
12	branch	Output	1	Tín hiệu điều khiển nhánh. Nếu 1 -> lệnh hiện tại là branch (EX stage sẽ đổi sang pc_branch), nếu 0 -> giữ nguyên
13	jump	Output	1	Tín hiệu điều khiển nhảy. Nếu 1 -> lệnh hiện tại là jump (xử lý sớm ở ID), nếu 0 -> giữ nguyên
14	jump_target	Output	32	Địa chỉ đích của jump
15	mem_read	Output	1	Tín hiệu điều khiển đọc dữ liệu từ D-CACHE
16	mem_write	Output	1	Tín hiệu điều khiển ghi dữ liệu vào D-CACHE
17	mem_to_reg	Output	1	Tín hiệu điều khiển. Nếu 1 -> dữ liệu ghi vào Register File từ D-CACHE (lệnh load), nếu 0 -> dữ liệu ghi vào từ ALU
18	reg_write	Output	1	Tín hiệu điều khiển Register File. Nếu 1 -> ghi vào Register File, nếu 0 -> không ghi
19	alu_src	Output	1	Tín hiệu chọn nguồn đầu vào thứ 2 cho ALU. Nếu 0 -> dùng reg_data2, nếu 1 -> dùng sign_ext_imm
20	reg_dst	Output	1	Tín hiệu chọn đích ghi thanh ghi. Nếu 0 -> thanh ghi đích là Rt, nếu 1 -> thanh ghi đích là Rd
21	alu_control	Output	4	Mã điều khiển ALU -> dùng để xác định phép toán

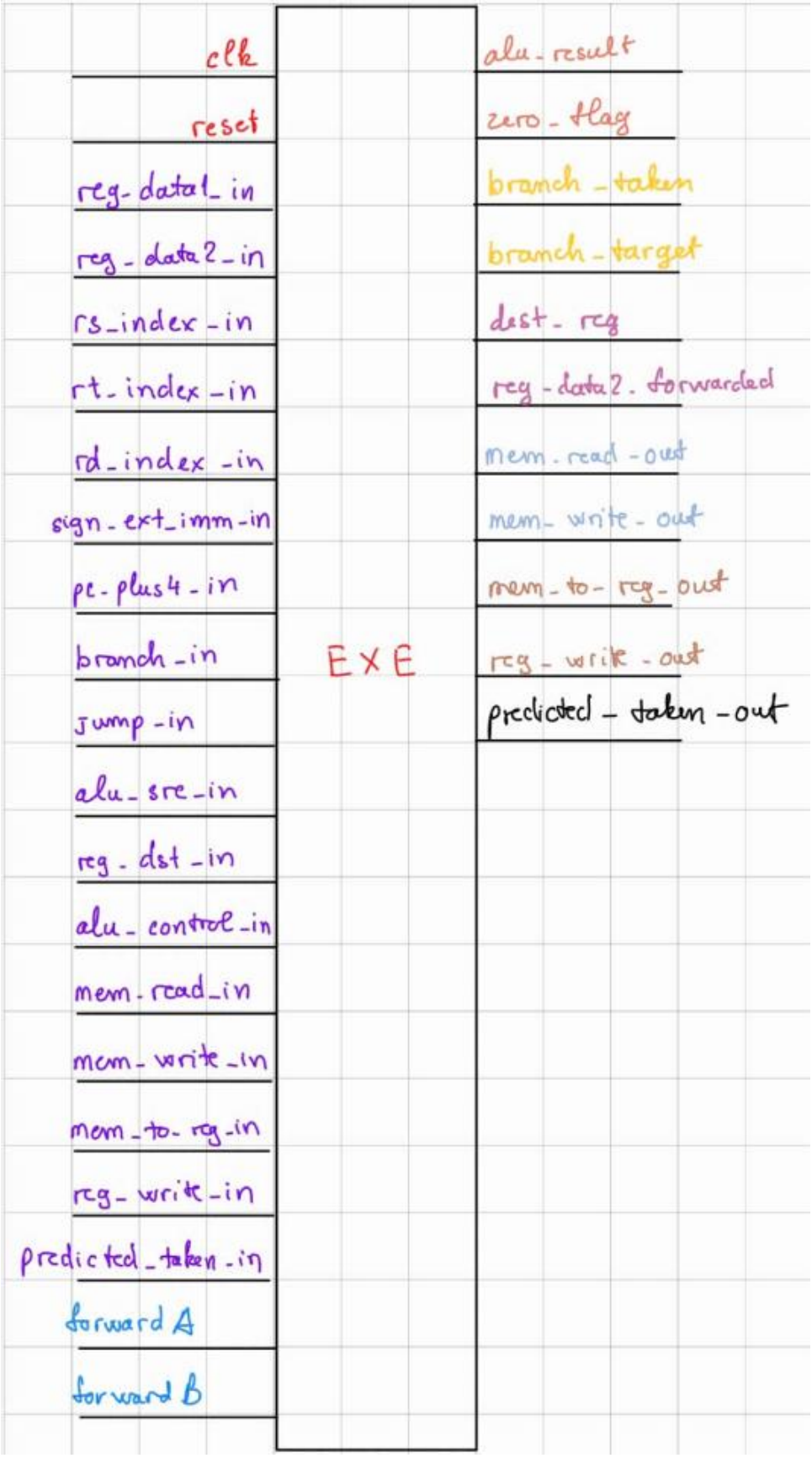
22	id_ex_pred_taken	Output	1	Chuyển tiếp cờ predicted_take_in sang EX stage để so sánh với kết quả thực tế (xác định flush)


clk
reset
instr_in
predicted_taken_in
pc_plus4_in
ID
rs_index
rt_index
rd_index
reg_data_1
reg_data_2
sign_ext_imm
branch
jump
jump_target
mem_read
mem_write
mem_to_reg
reg_write
alu_src
reg_dst
alu_control
id_ex_pred_taken

4. Execute

STT	Tên tín hiệu	I/O	Bit	Mô tả
1	clk	Input	1	Xung nhịp CPU
2	reset	Input	1	Tín hiệu reset CPU
3	reg_data1_in	Input	32	Giá trị đọc từ Register File
4	reg_data2_in	Input	32	Giá trị đọc từ Register File
5	rs_index_in	Input	5	Số hiệu thanh ghi Rs
6	rt_index_in	Input	5	Số hiệu thanh ghi Rt
7	rd_index_in	Input	5	Số hiệu thanh ghi Rd
8	sign_ext_imm_in	Input	32	Immediate đã được mở rộng dấu
9	pc_plus4_in	Input	32	Dùng để tính địa chỉ nhánh
10	branch_in	Input	1	Tín hiệu cho biết lệnh là branch
11	jump_in	Input	1	Tín hiệu cho biết lệnh là jump (đồng bộ là chủ yếu, vì jump đã được xử lý ở ID)
12	alu_src_in	Input	1	Tín hiệu chọn ALU thứ 2, 0 = reg_data2_in, 1 = sign_ext_imm_in
13	reg_dst_in	Input	1	Tín hiệu chọn thanh ghi đích, 0 = rt_index_in (I-type), 1 = rd_index_in (R-type)
14	alu_control_in	Input	4	Mã điều khiển ALU, chọn phép tính
15	mem_read_in	Input	1	Tín hiệu nếu 1 -> đọc vào D-CACHE
16	mem_write_in	Input	1	Tín hiệu nếu 1 -> ghi vào D-CACHE
17	mem_to_reg_in	Input	1	Tín hiệu chọn dữ liệu về WB (ALU / Mem)
18	reg_write_in	Input	1	Tín hiệu cho phép ghi ở WB
19	predicted_taken_in	Input	1	Cờ dự đoán nhánh từ IF/ID, giữ để so sánh với branch_taken (thực tế)
20	forwardA	Input	2	Mux chọn toán hạng A cho ALU: 00 -> reg_data1_in

				01 -> forward từ EX/MEM
21	forwardB	Input	2	Tương tự
22	alu_result	Output	32	Đầu ra ALU
23	zero_flag	Output	1	= 1 nếu alu_result = 0 -> dùng để xác định điều kiện nhánh
24	branch_taken	Output	1	Quyết định thực tế có branch hay không
25	branch_target	Output	32	Địa chỉ nhánh (PC+4 + offset)
26	dest_reg	Output	5	Số hiệu thanh ghi đích
27	reg_data2_forwarded	Output	32	Giá trị Rt sau khi qua mux ForwardB
28	mem_read_out	Output	1	Pass từ ID sang
29	mem_write_out	Output	1	Pass từ ID sang
30	mem_to_reg_out	Output	1	Pass từ ID sang
31	reg_write_out	Output	1	Pass từ ID sang
32	predicted_taken_out	Output	1	Cờ báo hiệu đã lấy địa chỉ dự đoán


clk
reset
reg_data1_in
reg_data2_in
rs_index_in
rt_index_in
rd_index_in
sign_ext_imm_in
pc_plus4_in
branch_in
jump_in
alu_src_in
reg_dst_in
alu_control_in
mem_read_in
mem_write_in
mem_to_reg_in
reg_write_in
predicted_taken_in
forward A
forward B
EXE
alu_result
zero_flag
branch_taken
branch_target
dest_reg
reg_data2_forwarded
mem_read_out
mem_write_out
mem_to_reg_out
reg_write_out
predicted_taken_out

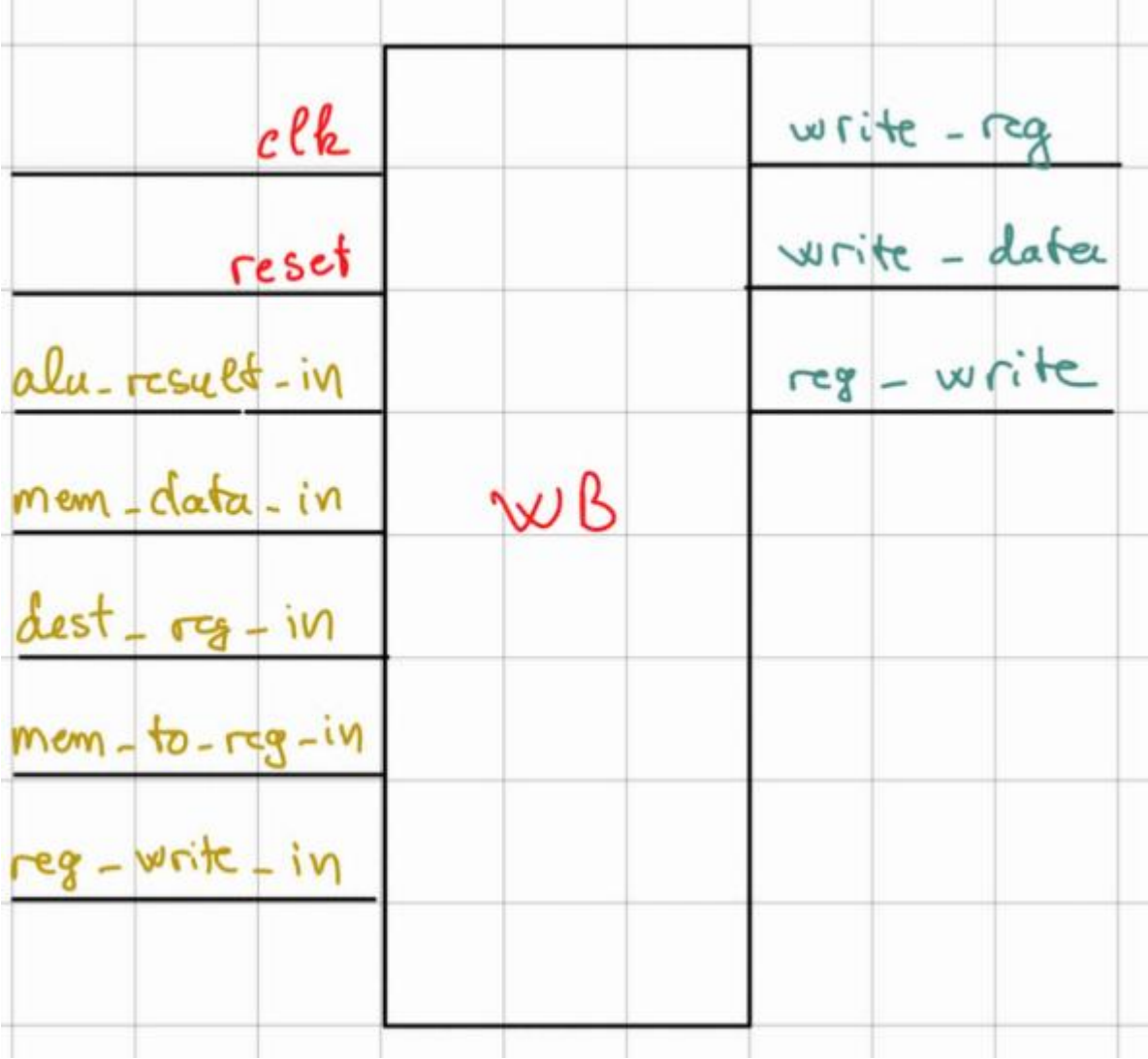
5. MEM

STT	Tên tín hiệu	I/O	Bit	Mô tả
1	clk	Input	1	Xung nhịp CPU
2	reset	Input	1	Tín hiệu reset CPU
3	alu_result_in	Input	32	Kết quả hoặc địa chỉ đã được tính ở EXE
4	write_data_in	Input	32	Dữ liệu cần ghi vào D-CACHE, lấy từ reg_data2
5	dest_reg_in	Input	5	Chỉ số thanh ghi đích từ EX/MEM
6	branch_taken_in	Input	1	Cờ thực tế có rẽ nhánh không
7	branch_target_in	Input	32	Địa chỉ rẽ nhánh được tính ở EXE
8	mem_read_in	Input	1	Tín hiệu báo hiệu truy cập (đọc) vào D-CACHE
9	mem_write_in	Input	1	Tín hiệu bái hiệu truy cập (viết) vào D-CACHE
10	mem_to_reg_in	Input	1	Tín hiệu xác định giá trị nào được ghi ở WB (data đọc từ D-CACHE hay ALU-Result)
11	reg_write_in	Input	1	Tín hiệu cho phép ghi ở WB
12	data_addr	Output	32	Địa chỉ bộ nhớ cần truy cập
13	data_write_data	Output	32	Dữ liệu cần ghi vào bộ nhớ
14	data_read_data	Input	32	Dữ liệu đọc được từ bộ nhớ
15	data_req	Output	1	Yêu cầu truy cập bộ nhớ (khi mem_write_in, mem_read_in = 1)
16	data_ready	Input	1	Báo bộ nhớ đã sẵn sàng
17	mem_data_out	Output	32	Dữ liệu đọc từ D-CACHE sang WB
18	alu_result_out	Output	32	Giá trị ALU từ EXE
19	dest_reg_out	Output	5	Thanh ghi đích từ EX/MEM (pass)
20	mem_to_reg_out	Output	1	Pass
21	reg_write_out	Output	1	Pass


clk
reset
alu_result_in
dest_reg_in
branch_taken_in
branch_target_in
mem_read_in
mem_write_in
mem_to_reg_in
reg_write_in
write_data_in
data_read
data_ready
MEM
data_reg
mem_data_out
alu_result_out
dest_reg_out
mem_to_reg_out
reg_write_out
data_addr
data_write

6. Write Back

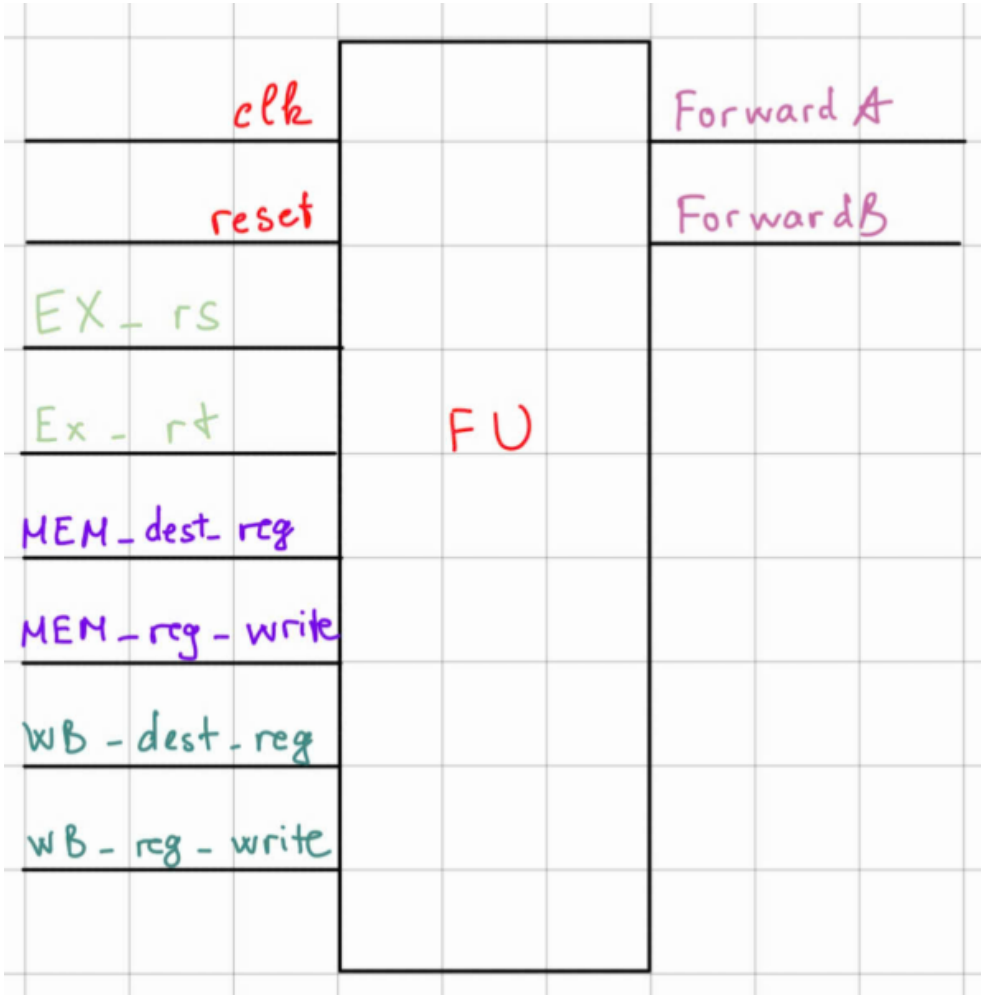
STT	Tên tín hiệu	I/O	Bit	Mô tả
1	clk	Input	1	Xung nhịp CPU
2	reset	Input	1	Tín hiệu reset CPU
3	alu_result_in	Input	32	Kết quả ALU
4	mem_data_in	Input	32	Dữ liệu từ D-CACHE
5	dest_reg_in	Input	5	Chỉ số thanh ghi đã chọn ở EXE
6	mem_to_reg_in	Input	1	Tín hiệu chọn nguồn ghi vào Register File (mem hoặc alu)
7	reg_write_in	Input	1	Tín hiệu cho phép ghi
8	write_reg	Output	5	Địa chỉ thanh ghi để ghi vào
9	write_data	Output	32	Data đã được chọn để ghi
10	reg_write	Output	1	Tín hiệu cho phép ghi (pass), chủ yếu để chặn ghi zero



7. Forwarding Unit

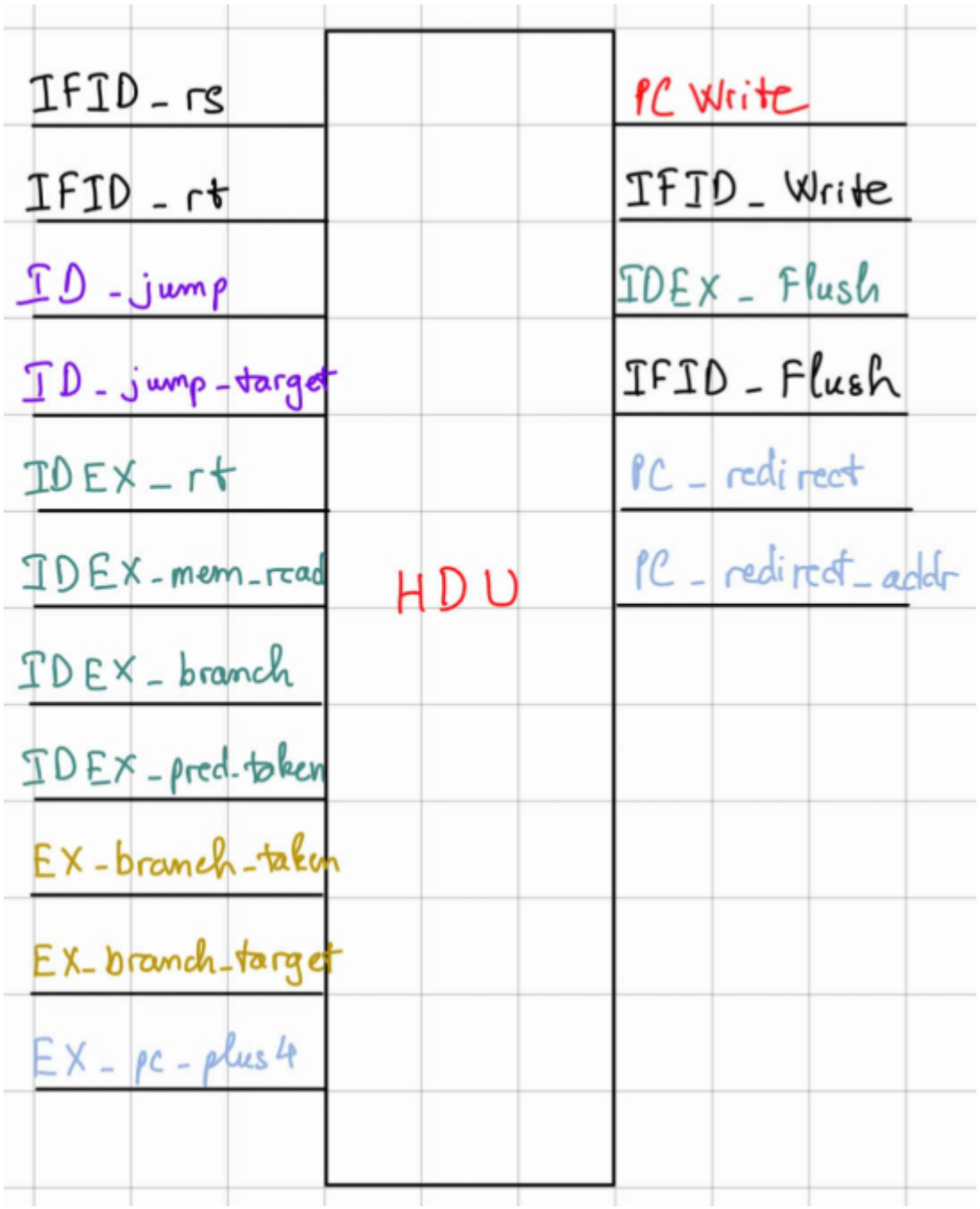
STT	Tên tín hiệu	I/O	Bit	Mô tả
1	clk	Input	1	Xung nhịp CPU
2	reset	Input	1	Tín hiệu reset CPU
3	EX_rs	Input	5	Chỉ số thanh ghi nguồn Rs
4	EX_rt	Input	5	Chỉ số thanh ghi nguồn Rt
5	MEM_dest_reg	Input	5	Thanh ghi đích của lệnh ở MEM
6	MEM_reg_write	Input	1	Cờ có ghi kết quả không

7	WB_dest_reg	Input	5	Thanh ghi đích của lệnh ở WB
8	WB_reg_write	Input	1	Cờ có ghi kết quả không
9	ForwardA	Output	2	00: Không hazard → dùng ID/EX.reg_data1 gốc. 10: Forward từ EX/MEM.alu_result. 01: Forward từ MEM/WB.write_data. 11: Không dung.
10	ForwardB	Output	2	00: Không hazard → dùng ID/EX.reg_data2 gốc. 10: Forward từ EX/MEM.alu_result. 01: Forward từ MEM/WB.write_data. 11: Không dùng.



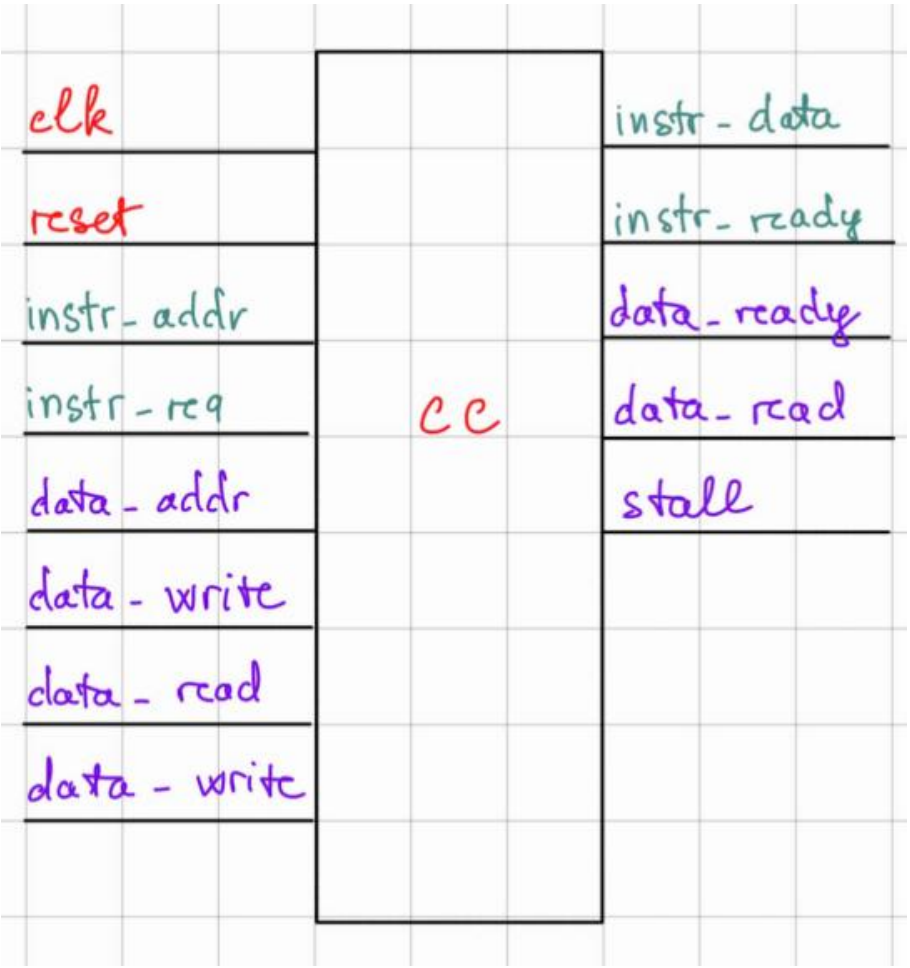
8. Hazard Detection Unit

STT	Tên tín hiệu	I/O	Bit	Mô tả
1	IFID_rs	Input	5	Số hiệu của thanh ghi Rs ở ID
2	IFID_rt	Input	5	Số hiệu của thanh ghi Rt ở ID
3	ID_jump	Input	1	Cờ cho biết lệnh ở ID có phải Jump
4	ID_jump_target	Input	32	Địa chỉ đích của Jump
5	IDEX_rt	Input	5	Thanh ghi đích Rd của lệnh ở EXE
6	IDEX_mem_read	Input	1	Nếu lệnh EXE là load + dest_reg == Rs/Rt của lệnh ở ID -> stall
7	IDEX_branch	Input	1	Cờ cho biết lệnh ở EXE có phải Branch
8	IDEX_pred_taken	Input	1	Giá trị dự đoán nhánh cho lệnh ở EXE
9	EX_branch_taken	Input	1	Kết quả nhánh thực tế từ ALU ở EXE
10	EX_branch_target	Input	32	Địa chỉ nhánh thực tế
11	EX_pc_plus4	Input	32	Địa chỉ tuần tự sau nhánh (dùng để xác định dự đoán sai -> redirect)
12	PCWrite	Output	1	Cho phép PC cập nhật (0: stall, 1: bình thường)
13	IFID_Write	Output	1	Cho phép IF/ID nạp lệnh mới (0: stall, 1: bình thường)
14	IDEX_Flush	Output	1	Chèn NOP vào EXE (load-use hazard)
15	IFID_Flush	Output	1	Xoá lệnh hiện tại ở IF/ID (branch mispredict, jump sai)
16	PC_redirect	Output	1	Yêu cầu thay đổi PC
17	PC_redirect_addr	Output	32	Giá trị PC mới khi redirect


IFID_rs
IFID_rt
ID_jump
ID_jump_target
IDEX_rt
IDEX_mem_read
IDEX_branch
IDEX_pred_taken
EX_branch_taken
EX_branch_target
EX_pc_plus4
HDU
PC Write
IFID_Write
IDEX_Flush
IFID_Flush
PC_redirect
PC_redirect_addr

9. Cache Controller

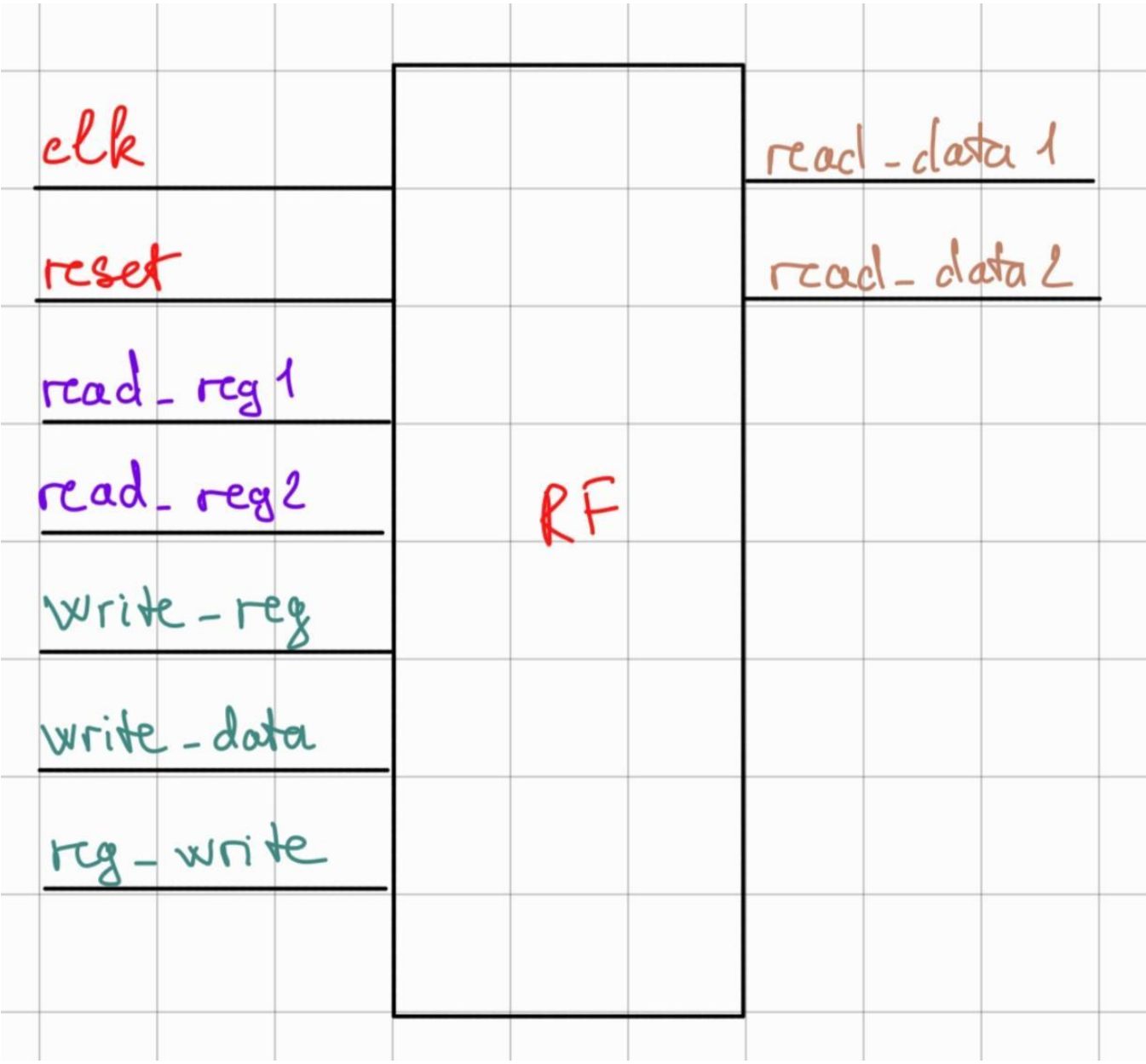
STT	Tên tín hiệu	I/O	Bit	Mô tả
1	clk	Input	1	Xung nhịp CPU
2	reset	Input	1	Tín hiệu reset CPU
3	instr_addr	Input	32	Địa chỉ lệnh cần fetch
4	instr_req	Input	1	Yêu cầu lấy lệnh I-CACHE
5	instr_data	Output	32	Lệnh từ I-CACHE (hit) hoặc memory (miss)
6	instr_ready	Output	1	Cờ báo hiệu sẵn sàng (1: đã xong, 0:chưa xong)
7	data_addr	Input	32	Địa chỉ dữ liệu cần truy cập
8	data_write_data	Input	32	Dữ liệu cần ghi vào D-CACHE
9	data_read	Input	1	Yêu cầu đọc (mem_read)
10	data_write	Input	1	Yêu cầu ghi (mem_write)
11	data_ready	Output	1	Cờ báo hiệu sẵn sàng (1: đã xong, 0: chưa xong)
12	data_read_data	Output	32	Dữ liệu đọc từ D-CACHE
13	stall	Output	1	báo CPU phải dừng vì cache miss



10. Register File

STT	Tên tín hiệu	I/O	Bit	Mô tả
1	clk	Input	1	Xung nhịp CPU
2	reset	Input	1	Tín hiệu reset CPU
3	read_reg1	Input	5	Địa chỉ thanh ghi nguồn Rs
4	read_reg2	Input	5	Địa chỉ thanh ghi nguồn Rt
5	write_reg	Input	5	Địa chỉ thanh ghi đích
6	write_data	Input	32	Dữ liệu ghi
7	reg_write	Input	1	Tín hiệu cho phép ghi

8	read_data1	Output	32	Dữ liệu đọc từ thanh ghi ở read_reg1
9	read_data2	Output	32	Dữ liệu đọc từ thanh ghi ở read_reg2



11. Bus IF/ID

STT	Tên tín hiệu	I/O	Bit	Mô tả
1	if_id_instr	Output	32	Lệnh từ IF stage chuyển sang ID (instr_out). Nếu flush → NOP.
2	if_id_pc_plus4	Output	32	Địa chỉ PC+4 đi kèm lệnh, dùng để tính toán nhánh/jump.
3	if_id_pred_taken	Output	1	Cờ dự đoán nhánh cho lệnh này (1 = predicted taken).

12. Bus ID/EX

STT	Tên tín hiệu	I/O	Bit	Mô tả
1	id_ex_instr	Output	32	Toàn bộ mã lệnh (tùy chọn, hỗ trợ debug hoặc ALU control).
2	id_ex_pc_plus4	Output	32	Địa chỉ PC+4 cho lệnh đang ở EX.
3	id_ex_reg_data1	Output	32	Toán hạng nguồn Rs.
4	id_ex_reg_data2	Output	32	Toán hạng nguồn Rt.
5	id_ex_rs	Output	5	Chỉ số thanh ghi Rs.
6	id_ex_rt	Output	5	Chỉ số thanh ghi Rt.
7	id_ex_rd	Output	5	Chỉ số thanh ghi Rd (R-type).
8	id_ex_sign_ext_imm	Output	32	Immediate mở rộng dấu từ ID.
9	id_ex_branch	Output	1	Cờ lệnh nhánh.
10	id_ex_jump	Output	1	Cờ lệnh jump.
11	id_ex_mem_read	Output	1	Tín hiệu MemRead cho lệnh load.
12	id_ex_mem_write	Output	1	Tín hiệu MemWrite cho lệnh store.
13	id_ex_mem_to_reg	Output	1	Chọn dữ liệu từ MEM để ghi về WB.
14	id_ex_reg_write	Output	1	Cho phép ghi Register File ở WB.
15	id_ex_alu_src	Output	1	Chọn toán hạng ALU thứ 2 (reg_data2 hoặc imm).
16	id_ex_reg_dst	Output	1	Chọn thanh ghi đích (Rt hoặc Rd).
17	id_ex_alu_control	Output	4	Mã điều khiển ALU.
18	id_ex_pred_taken	Output	1	Cờ dự đoán nhánh kèm lệnh sang EX.

13. Bus EX/MEM

STT	Tên tín hiệu	I/O	Bit	Mô tả
1	ex_mem_alu_result	Output	32	Kết quả ALU hoặc địa chỉ hiệu dụng.
2	ex_mem_write_data	Output	32	Dữ liệu cần ghi bộ nhớ (store).
3	ex_mem_dest_reg	Output	5	Thanh ghi đích.
4	ex_mem_zero_flag	Output	1	Cờ Zero từ ALU.
5	ex_mem_branch_taken	Output	1	Nhánh thực tế được thực hiện hay không.
6	ex_mem_branch_target	Output	32	Địa chỉ nhánh được tính.
7	ex_mem_mem_read	Output	1	Tín hiệu MemRead (load).
8	ex_mem_mem_write	Output	1	Tín hiệu MemWrite (store).
9	ex_mem_mem_to_reg	Output	1	MemtoReg sang WB.
10	ex_mem_reg_write	Output	1	RegWrite sang WB.
11	ex_mem_pred_taken	Output	1	Cờ dự đoán nhánh ban đầu (tham khảo).

14. Bus MEM/WB

STT	Tên tín hiệu	I/O	Bit	Mô tả
1	mem_wb_alu_result	Output	32	Kết quả ALU từ MEM sang WB (pass-through).
2	mem_wb_mem_data	Output	32	Dữ liệu đọc từ bộ nhớ (load).
3	mem_wb_dest_reg	Output	5	Thanh ghi đích.
4	mem_wb_mem_to_reg	Output	1	Chọn dữ liệu từ memory thay vì ALU ở WB.
5	mem_wb_reg_write	Output	1	Cho phép ghi Register File.