BỘ GIÁO DỤC VÀ ĐÀO TẠO

TRƯỜNG ĐẠI HỌC SƯ PHẠM KỸ THUẬT TP. HỒ CHÍ MINH

KHOA ĐIỆN – ĐIỆN TỬ

BỘ MÔN KỸ THUẬT MÁY TÍNH – VIỄN THÔNG

REPORT

THIẾT KẾ VÀ ĐÁNH GIÁ BỘ VI XỬ LÝ RISC-V 32 BITS

GVHD: TS. Đỗ Duy Tân

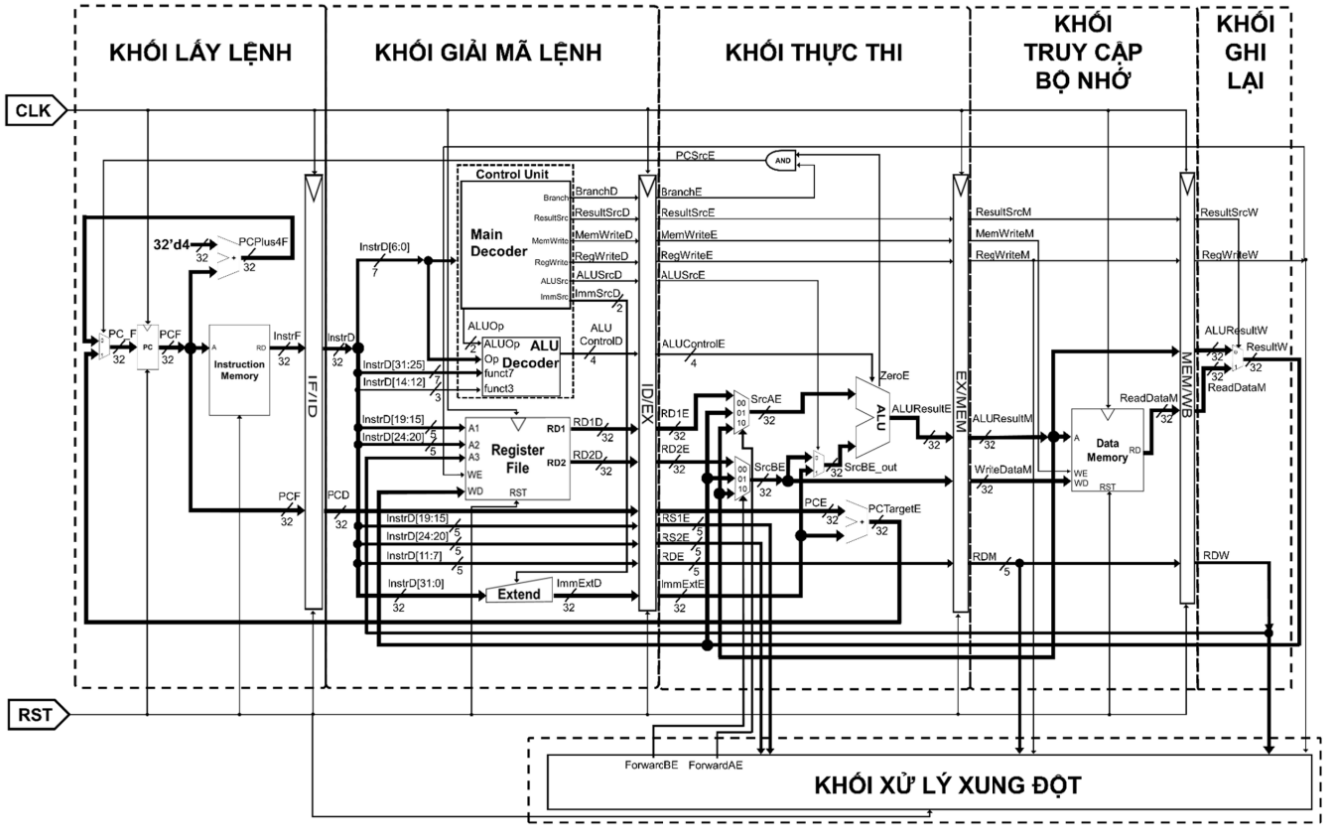
SVTH: Phạm Quang Hợp

MSSV: 22119178

TP. Hồ Chí Minh, tháng 5 năm 2025

I. SƠ ĐỒ KHỐI VÀ CHỨC NĂNG MỖI KHỐI

1. Top Module



Hình 1: Sơ đồ chi tiết hệ thống

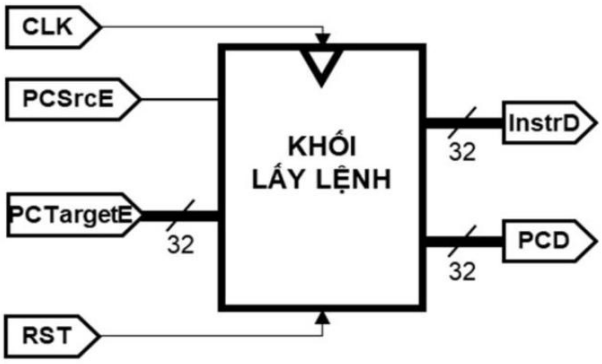
Hệ thống CPU 32 bit RISC-V sử dụng kỹ thuật đường ống 5 tầng để tối ưu hóa hiệu suất và tốc độ xử lý bằng cách chia quá trình thực hiện lệnh thành năm giai đoạn riêng biệt: lấy lệnh, giải mã lệnh, thực thi, truy cập bộ nhớ, và ghi lại.

Trong giai đoạn lấy lệnh, bộ nhớ lệnh cung cấp các lệnh từ địa chỉ do PC chỉ định, và PC được cập nhật để trỏ đến lệnh tiếp theo. Giai đoạn giải mã lệnh giải mã lệnh để xác định các thanh ghi nguồn, thanh ghi đích, và loại thao tác, đồng thời đọc giá trị từ các thanh ghi nguồn và tạo ra các tín hiệu điều khiển cần thiết. Ở giai đoạn thực thi, ALU thực hiện các thao tác tính toán hoặc luận lý dựa trên các giá trị đầu vào và tín hiệu điều khiển, và kết quả được lưu vào thanh ghi trung gian EX/MEM. Giai đoạn truy cập bộ nhớ thực hiện việc đọc hoặc ghi dữ liệu từ hoặc đến bộ nhớ dữ

liệu nếu lệnh yêu cầu, và lưu kết quả vào thanh ghi MEM/WB. Cuối cùng, giai đoạn ghi lại ghi kết quả cuối cùng của lệnh vào tệp thanh ghi, cập nhật giá trị của thanh ghi đích.

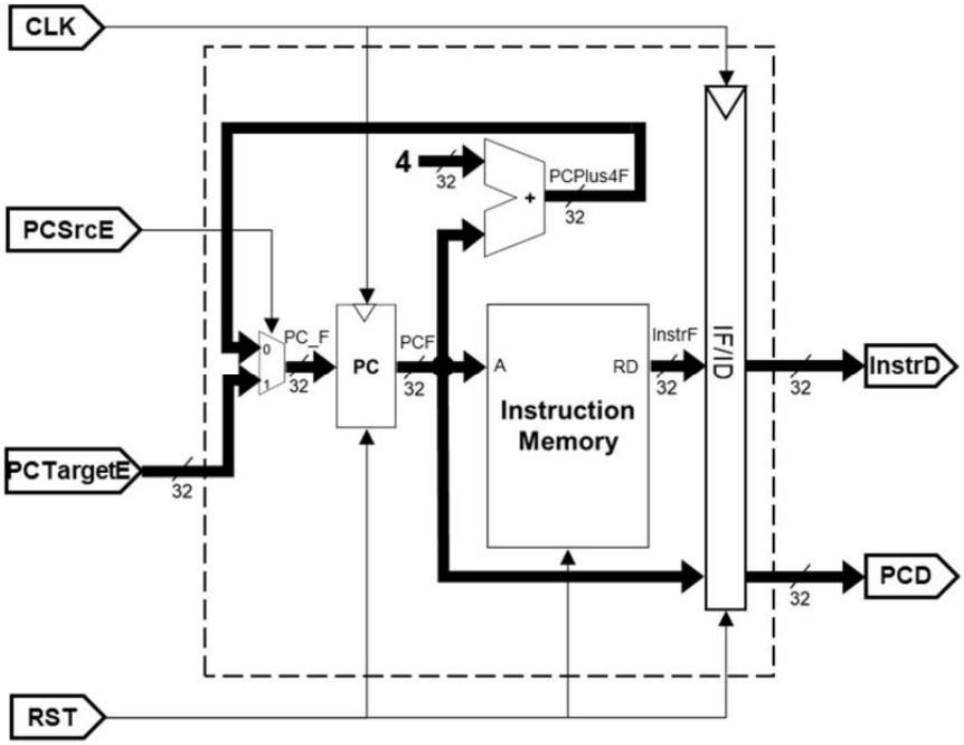
2. Khối lấy lệnh

Sơ đồ khối: gồm 4 ngõ vào và 2 ngõ ra.



Hình 2: Sơ đồ Khối lấy lệnh

Sơ đồ chi tiết gồm các mô đun: bộ ghép kênh chọn đầu vào của chương trình, bộ cộng, bộ đếm chương trình, bộ nhớ lệnh và thanh ghi IF/ID.



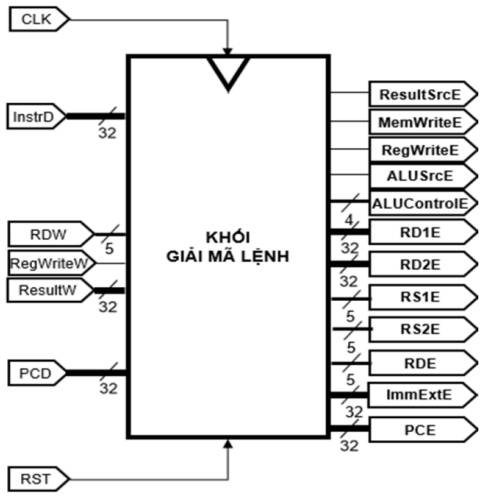
Hình 3: Sơ đồ chi tiết Khối lấy lệnh

Khối lấy lệnh trong Hình 3 thực hiện nhiệm vụ lấy lệnh từ bộ nhớ lệnh và cập nhật bộ đếm chương trình. Dưới đây là các bước hoạt động chi tiết của khối lấy lệnh:

- Tín hiệu PCSrcE và PCTargetE:
  - PCSrcE: Tín hiệu điều khiển, quyết định nguồn giá trị mới của PC.
  - PCTargetE: Địa chỉ đích, được sử dụng khi PCSrcE = 1.
  - PCPlus4F: Giá trị của PC được cộng thêm 4, sử dụng để trỏ đến lệnh kế tiếp trong bộ nhớ.
- Bộ ghép kênh chọn đầu vào cho PC.
  - Nếu PCSrcE = 0, chọn PC + 4 (tăng PC lên 4 để trỏ đến lệnh kế tiếp).
  - Nếu PCSrcE = 1, chọn giá trị từ PCTargetE.
- Bộ đếm chương trình lưu trữ giá trị hiện tại của bộ đếm chương trình. Giá trị này được cập nhật mỗi chu kỳ xung đồng hồ.
- Bộ nhớ lệnh, lưu trữ các lệnh của chương trình. Địa chỉ A là giá trị của PC. Đầu ra RD (InstrF) là lệnh tại địa chỉ tương ứng.
- Thanh ghi IF/ID là thanh ghi đường ống, lưu trữ lệnh (InstrF) và giá trị PC (PCF) để chuyển sang giai đoạn giải mã lệnh (InstrD và PCD).

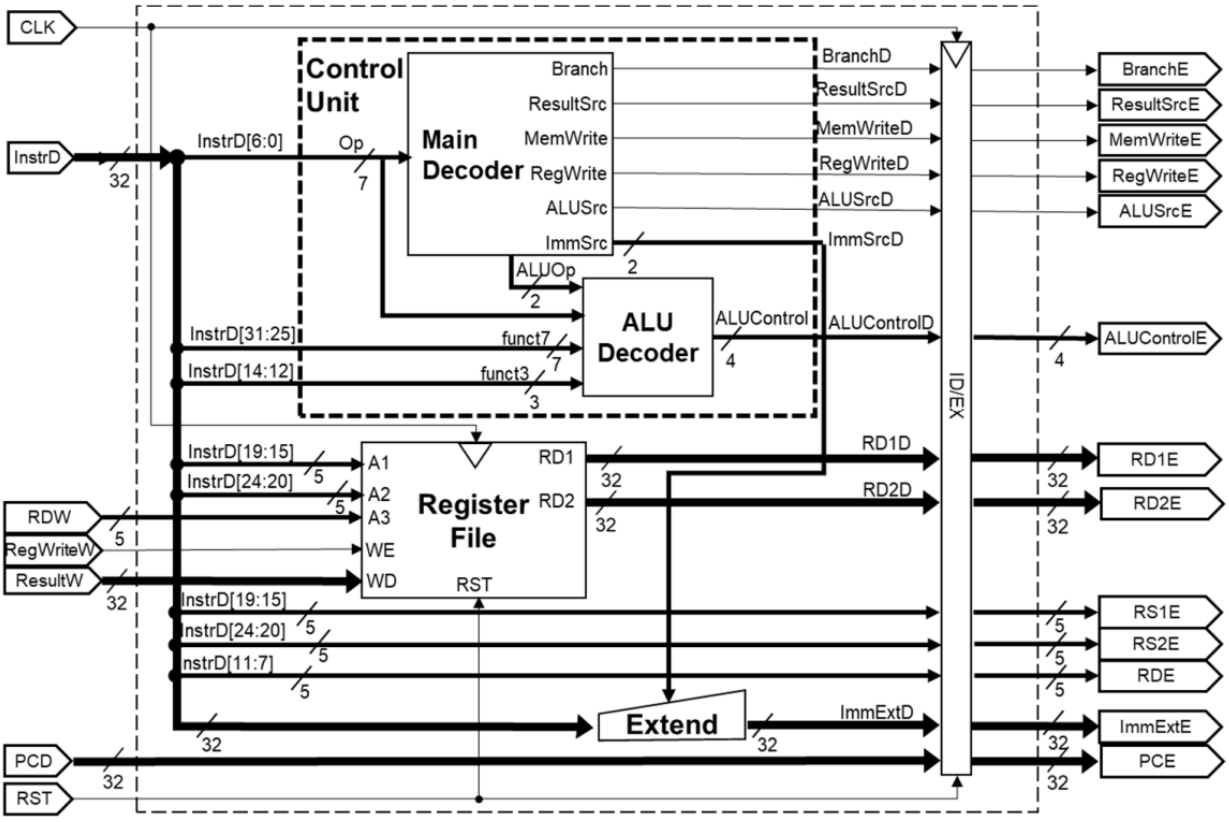
3. Khối giải mã lệnh

Sơ đồ khối: gồm 7 ngõ vào và 13 ngõ ra.



Hình 4: Sơ đồ Khối giải mã lệnh

Sơ đồ chi tiết gồm các mô đun: Đơn vị điều khiển có bộ giải mã chính và bộ giải mã ALU, tệp thanh ghi, khối mở rộng và thanh ghi ID/EX.



Hình 5: Sơ đồ chi tiết Khối giải mã lệnh

Khối giải mã lệnh ở Hình 5 có nhiệm vụ phân tích lệnh và tạo ra các tín hiệu điều khiển cần thiết để thực hiện lệnh. Quá trình giải mã bao gồm việc xác định loại lệnh, các tham số của lệnh, và điều khiển các đơn vị chức năng khác trong CPU. Dưới đây là mô tả chi tiết các bước hoạt động của khối giải mã lệnh:

- Bộ giải mã chính gồm tín hiệu đầu vào là Op nhận 7 bit opcode từ lệnh (InstrD[6:0]) để xác định loại lệnh và tạo ra các tín hiệu điều khiển như mô tả tại Bảng 1:

• Branch: Tín hiệu nhánh.

• ResultSrc: Chọn nguồn kết quả (ALU hoặc bộ nhớ).

• MemWrite: Tín hiệu ghi bộ nhớ.

• RegWrite: Tín hiệu ghi thanh ghi.

• ALUSrc: Chọn đầu vào thứ hai của ALU (thanh ghi hoặc hằng số mở rộng). ImmSrc: Chọn loại hằng số mở rộng.

- Bộ giải mã ALU có tín hiệu đầu vào funct7, funct3 và ALUOp nhận các bit chức năng từ lệnh (InstrD[31:25], InstrD[14:12]) để tạo ra tín hiệu điều khiển ALU(ALUControlD) như mô tả Bảng 2.
- Bộ thanh ghi:

  • A1, A2, A3: Địa chỉ các thanh ghi nguồn và đích từ lệnh (InstrD[19:15], InstrD[24:20], InstrD[11:7]).

  • RD1, RD2: Giá trị đọc từ các thanh ghi nguồn.

  • WD: Giá trị ghi vào thanh ghi đích khi thực hiện lệnh ghi.

  • WE: Tín hiệu cho phép ghi thanh ghi.
- Khối mở rộng hằng số từ lệnh, được tạo ra từ các bit tương ứng của lệnh như Bảng 3
- Thanh ghi ID/EX lưu trữ các tín hiệu điều khiển và giá trị cần thiết để chuyển sang giai đoạn thực thi lệnh.

Bảng 1: Bảng trạng thái của tín hiệu điều khiển với từng loại lệnh

Instruction	**Op**	**RegWrite**	**ImmSrc**	**ALUSrc**	**MemWrite**	**ResultSrc**	**Branch**	**ALUOp**
I-Type (loads)	0000011	1	00	1	0	1	0	00
S-Type (Stores)	0100011	0	01	1	1	x	0	00
B-Type (Branch)	1100011	0	10	0	0	x	1	01
I-Type (Arithmetic)	0010011	1	00	1	0	0	0	00
R-Type	0110011	1	xx	0	0	0	0	10

Bảng 2: Bảng trạng thái của tín hiệu điều khiển ALU

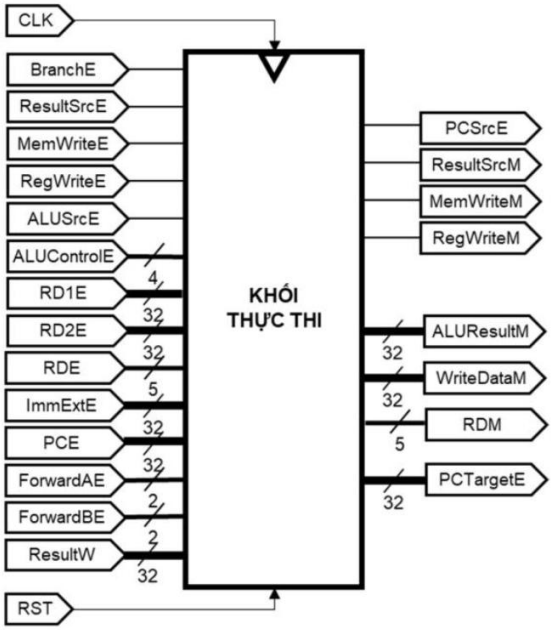
ALUOP	funct3	{op[5],funct7[5],funct7[0]}	ALUControl	Instruction
00	x	x	0000	lw,sw
01	x	x	0001	beq
10	000	000,001,010,011,100,111	0000	add
		110	0001	sub
		101	0010	mul
	001	x	0011	sll
	010	x	0100	slt
	100	xx0	0101	xor
		xx1	0110	div
	101	x0x	0111	srl
		x1x	1000	sra
	110	xx0	1001	or
		xx1	1010	rem
	111	x	1011	and

Bảng 3: Mã hóa ImmSrc

ImmSrc	ImmExt	Loại	Mô tả
00	{{20{Instr[31]}}, Instr[31:20]}	I	Giá trị tức thời 12 bit có dấu
01	{{20{Instr[31]}},Instr[31:20],Instr[11:7]}	S	Giá trị tức thời 12 bit có dấu
10	{{20{Instr[31]}},Instr[7],Instr[30:25],Instr[11:8],1'b0}	B	Giá trị tức thời 13 bit có dấu

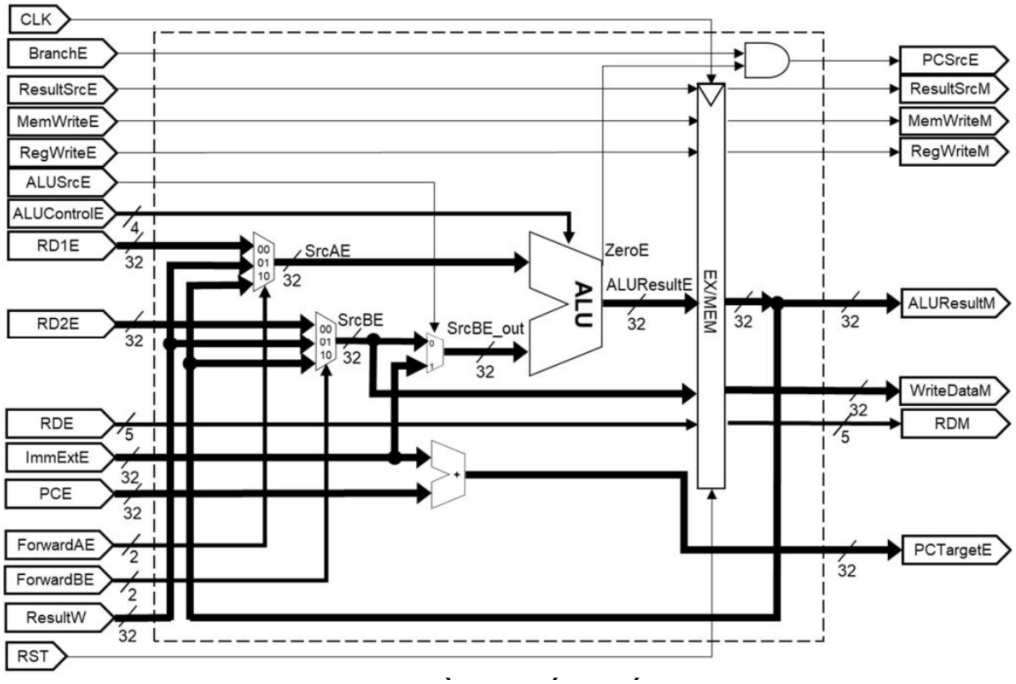
4. Khối thực thi

Sơ đồ khối: gồm 16 ngõ vào và 8 ngõ ra.



Hình 6: Sơ đồ Khối thực thi

Sơ đồ chi tiết gồm: Đơn vị luận lý số học, bộ cộng, bộ ghép kênh, thanh ghi EX/MEM.



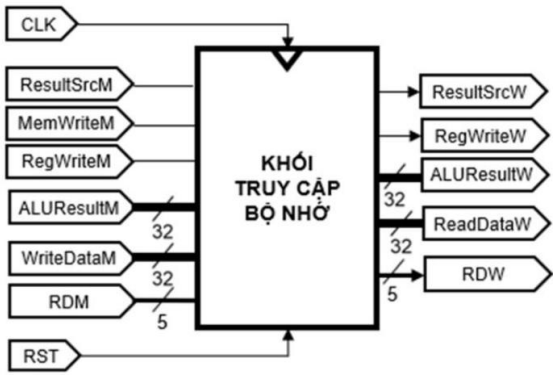
Hình 7: Sơ đồ chi tiết Khối thực thi

Khối thực thi ở Hình 7 có chức năng chính là thực hiện các phép tính và hoạt động được xác định bởi lệnh, như được giải mã từ khối giải mã lệnh. Chức năng cụ thể của khối thực thi bao gồm:

- ALU thực hiện các phép toán số học và luận lý dựa trên tín hiệu điều khiển ALUControlE. Các giá trị đầu vào cho ALU được chọn từ thanh ghi, giá trị tức thời hoặc các giá trị chuyển tiếp. Tín hiệu ZeroE chỉ ra kết quả ALU bằng không, dùng cho các quyết định nhánh.
- Xử lý xung đột: Các tín hiệu ForwardAE và ForwardBE xử lý xung đột dữ liệu bằng cách chuyển tiếp dữ liệu từ các giai đoạn đường ống sau về giai đoạn thực thi.
- Các bộ chọn lựa chọn giữa dữ liệu chuyển tiếp, dữ liệu thanh ghi hoặc các giá trị tức thời cho các phép tính của ALU.
- Tín hiệu điều khiển: BranchE, ResultSrcE, MemWriteE, RegWriteE, ALUSrcE, ALUControlE điều khiển các hành động của khối thực thi. Tín hiệu ResultW điều khiển kết quả cuối cùng.
- Tín hiệu đầu ra ALUResultM chuyển kết quả từ ALU được chuyển đến giai đoạn bộ nhớ. Tín hiệu WriteDataM ghi dữ liệu vào bộ nhớ. Tín hiệu PCTargetE địa chỉ mục tiêu cho các lệnh nhánh.

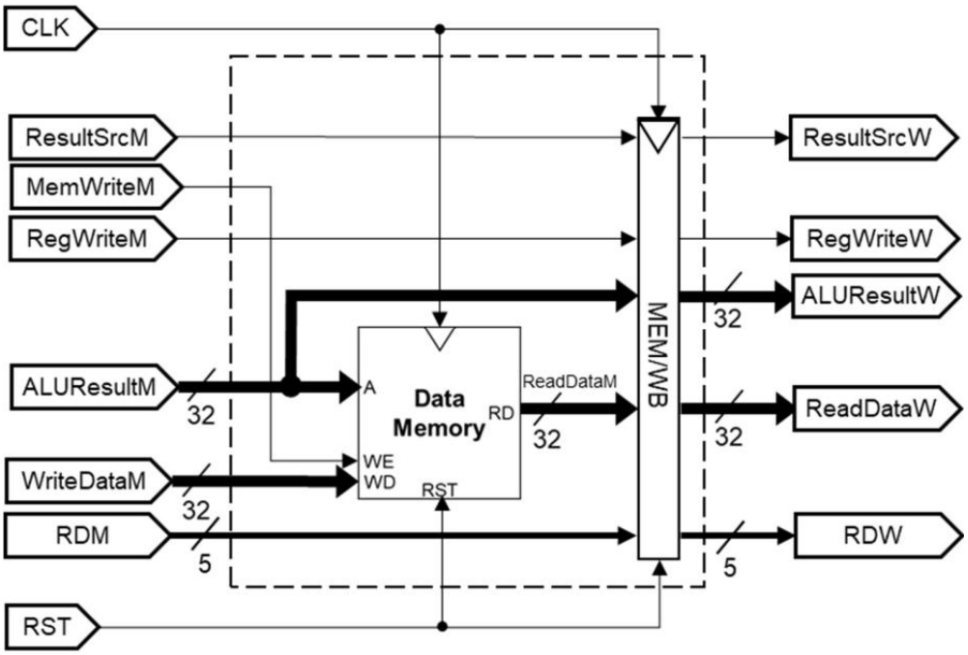
5. Khối truy cập bộ nhớ

Sơ đồ khối bao gồm có 8 ngõ vào và 5 ngõ ra:



Hình 8: Sơ đồ Khối truy cập bộ nhớ

Sơ đồ chi tiết gồm bộ nhớ dữ liệu và thanh ghi MEM/WB:



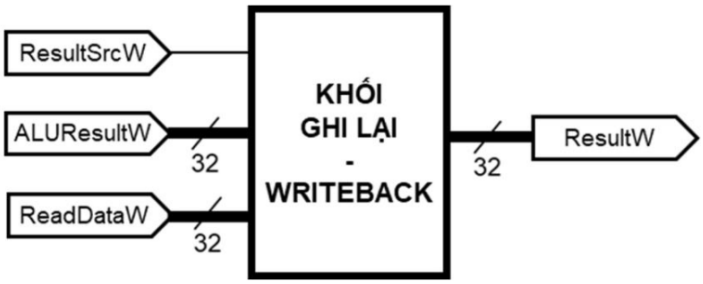
Hình 9: Sơ đồ chi tiết Khối truy cập bộ nhớ

Khối truy cập bộ nhớ đọc và ghi dữ liệu vào bộ nhớ chính. Khối truy cập bộ nhớ có các chức năng quan trọng sau:

- Tín hiệu MemWriteM điều khiển cho phép ghi dữ liệu vào bộ nhớ.
- Tín hiệu ALUResultM là địa chỉ từ đơn vị tính toán (ALU) để ghi vào bộ nhớ.
- Tín hiệu WriteDataM là dữ liệu để ghi vào bộ nhớ.
- Thanh ghi MEM/WB lưu trữ các tín hiệu điều khiển và giá trị cần thiết để chuyển sang giai đoạn thực thi lệnh.

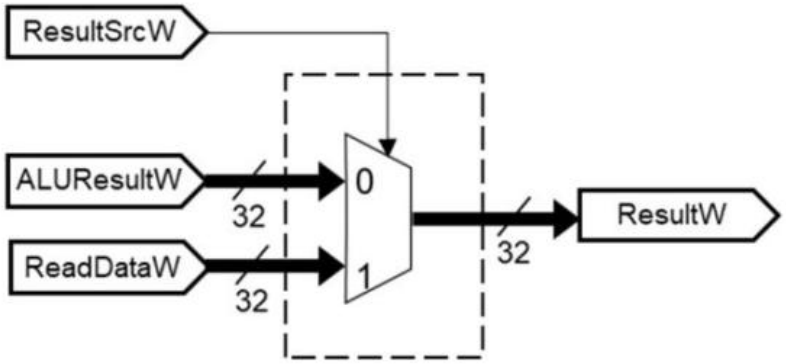
6. Khối ghi lại

Sơ đồ khối: gồm 3 ngõ vào và 1 ngõ ra.



Hình 10: Sơ đồ Khối ghi lại

Sơ đồ chi tiết chỉ có một bộ ghép kênh 2 ngõ vào.



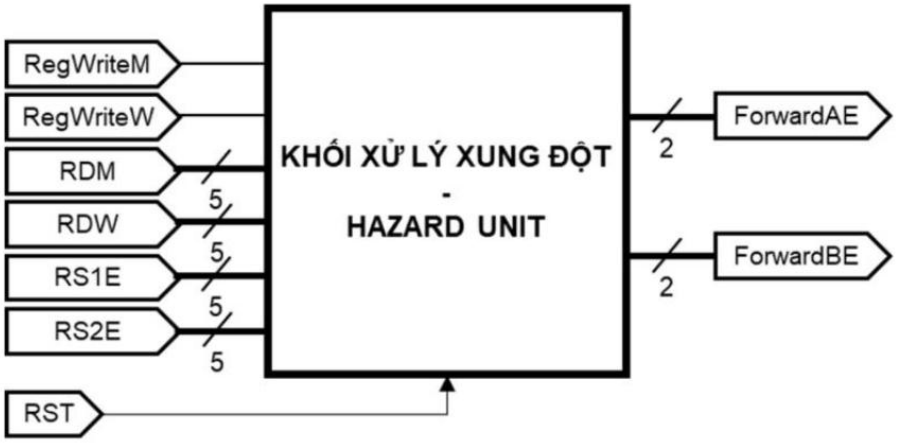
Hình 11: Sơ đồ chi tiết Khối ghi lại

Khối ghi lại trong sơ đồ này có chức năng chọn kết quả cuối cùng để ghi vào các thanh ghi trong hệ thống. Hoạt động của khối này có thể được giải thích như sau:

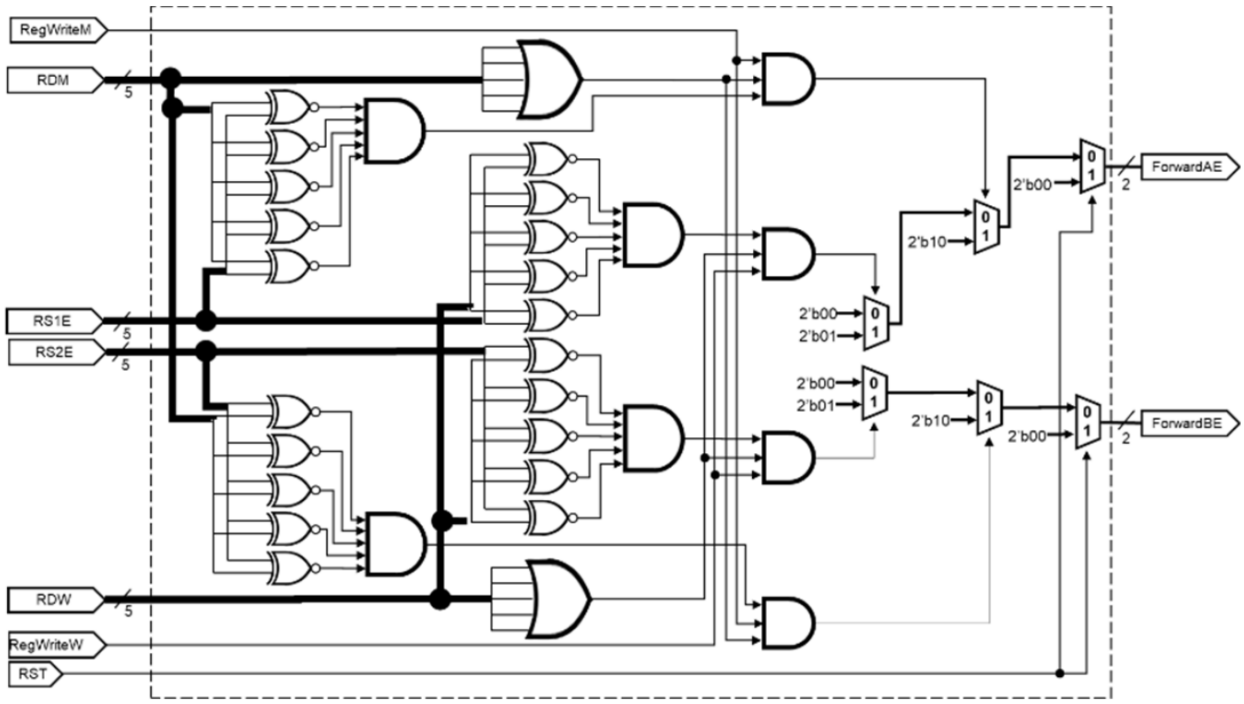
- Tín hiệu ResultSrcW điều khiển để chọn nguồn dữ liệu kết quả. Tín hiệu ALUResultW là dữ liệu kết quả từ đơn vị tính toán (ALU). Tín hiệu ReadDataW là dữ liệu được đọc từ bộ nhớ.
- Bộ chọn quyết định dữ liệu nào sẽ được truyền ra ngoài thông qua tín hiệu điều khiển ResultSrcW.

7. Khối xử lý xung đột

Khối xử lý xung đột trong sơ đồ này có nhiệm vụ phát hiện và giải quyết các xung đột xảy ra trong quá trình thực hiện lệnh.



Hình 12: Sơ đồ Khối xử lý xung đột



Hình 13: Sơ đồ chi tiết Khối xử lý xung đột

ForwardAE, ForwardBE được sử dụng để chọn nguồn dữ liệu phù hợp để tránh xung đột:

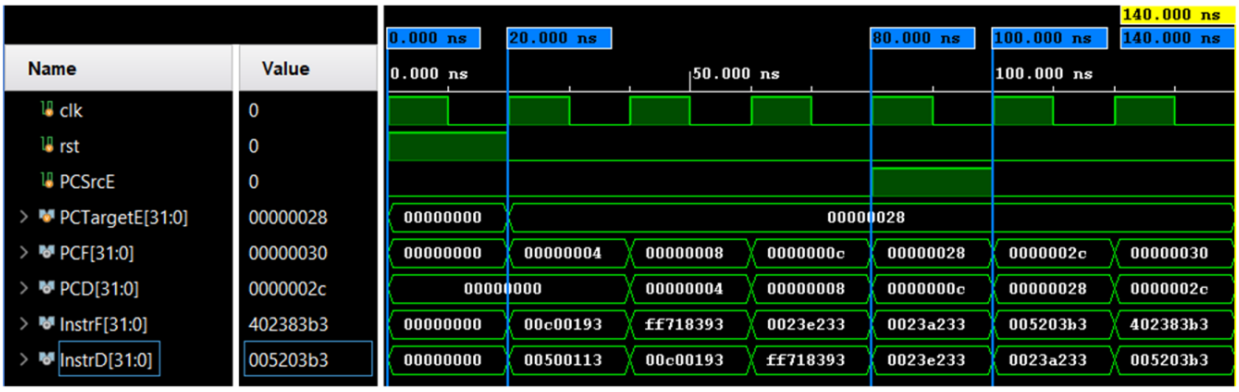
- Nếu có xung đột, khối sẽ điều khiển để chuyển tiếp dữ liệu từ giai đoạn trước đến giai đoạn hiện tại , đảm bảo rằng lệnh hiện tại có được dữ liệu chính xác mà không cần chờ đợi.
- Nếu không có xung đột, các tín hiệu này sẽ cho phép dữ liệu được sử dụng trực tiếp từ các thanh ghi nguồn.

Bảng 4: Các giá trị điều khiển cho các bộ ghép kênh chuyển tiếp

Điều khiển MUX	Nguồn	Giải thích
ForwardA = 00	ID/EX	Toán hạng ALU đầu tiên xuất phát từ Tệp thanh ghi.
ForwardA = 10	EX/MEM	Toán hạng ALU đầu tiên xuất phát từ kết quả ALU trước đó.
ForwardA = 01	MEM/WB	Toán hạng ALU đầu tiên xuất phát từ bộ nhớ dữ liệu.
ForwardB = 00	ID/EX	Toán hạng ALU thứ hai xuất phát từ Tệp thanh ghi.
ForwardB = 10	EX/MEM	Toán hạng ALU thứ hai xuất phát từ kết quả ALU trước đó.
ForwardB = 01	MEM/WB	Toán hạng ALU thứ hai xuất phát từ bộ nhớ dữ liệu.

II. KẾT QUẢ MÔ PHỎNG DẠNG SÓNG CÁC KHỐI

1. Khối lấy lệnh



Hình 14: Dạng sóng của Khối lấy lệnh

Trong hình 14 mô tả chức năng của khối lấy lệnh được chia làm 4 trường hợp:

- **Trường hợp 1: Từ 0 – 20ns kiểm tra chức năng đặt lại hệ thống.**

  Tín hiệu rst ở mức 1, do đó các ngõ ra của khối được đặt về mức 0.

- **Trường hợp 2: Từ 20 – 80ns kiểm tra chức năng tăng địa chỉ PC.**

  Tín hiệu rst ở mức 0, tín hiệu PCSrcE ở mức 0, cho phép giá trị PC tăng tuần tự lần lượt là PCF = 32'h00000004, 32'h00000008, 32'h0000000c, tương ứng với các giá trị PC đó là lệnh InstrF = 32'h00c00193, 32'hff718393, 32'h0023e233. Vì tín hiệu PCD và InstrD là ngõ ra của thanh ghi với ngõ vào là PCF và InstrF nên các giá trị của ngõ ra sẽ chậm hơn so với ngõ vào 1 chu kỳ xung đồng hồ.

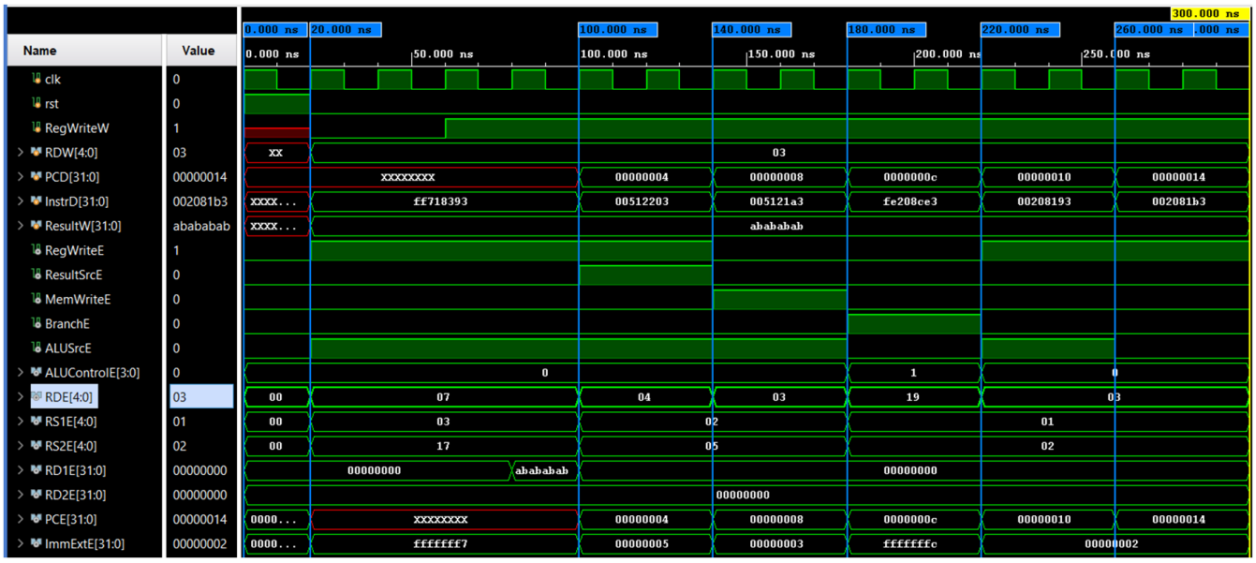
- **Trường hợp 3: Từ 80 – 100ns kiểm tra chức năng nhảy địa chỉ.**

  Tín hiệu rst ở mức 0, tín hiệu PCSrcE ở mức 1, cho phép giá trị PC nhận giá trị đích. Giá trị đích PCTargetE = 32'h00000028 nên giá trị PC = 32'h00000028, tương ứng giá trị PC là lệnh InstrF = 32'h23a233. Vì tín hiệu PCD và InstrD là ngõ ra của thanh ghi với ngõ vào là PCF và InstrF nên các giá trị của ngõ ra sẽ chậm hơn so với ngõ vào 1 chu kỳ xung đồng hồ.

- **Trường hợp 4: Từ 100 – 140ns kiểm tra chức năng tăng địa chỉ PC sau khi kết thúc lệnh nhảy.**

  Tín hiệu rst ở mức 0, tín hiệu PCSrcE ở mức 0, cho phép giá trị PC tăng tuần tự lần lượt là PCF = 32'h0000002c, 32'h00000030. Lệnh tương ứng là InstrF = 32'h005203b3, 32'h402393b3. Vì tín hiệu PCD và InstrD là ngõ ra của thanh ghi với ngõ vào là PCF và InstrF nên các giá trị của ngõ ra sẽ chậm hơn so với ngõ vào 1 chu kỳ xung đồng hồ.

2. Khối giải mã lệnh



Hình 15: Dạng sóng của Khối giải mã lệnh

Hình 15 mô tả chức năng của Khối giải mã lệnh được chia làm 8 trường hợp:

- **Trường hợp 1: Từ 0 – 20ns kiểm tra chức năng đặt lại hệ thống.**

  Tín hiệu rst ở mức 1, do đó các ngõ ra của khối được đặt về mức 0.

- **Trường hợp 2: Từ 20 – 100ns kiểm tra chức năng ghi vào tệp thanh ghi.**

  - Tín hiệu rst ở mức 0.
  - Lệnh ngõ vào InstrD = 32'hff718393 (addi x7, x3, -9) tương ứng cộng giá trị -9 và giá trị tại thanh ghi x3 lưu vào thanh ghi x7.
  - Giá trị và địa chỉ ghi
    - ResultW = 32'hababababab là giá trị cần ghi vào thanh ghi

- RDW = 5’h3 là địa chỉ thanh ghi đích x3.

- Từ 20 – 60ns, tín hiệu RegWriteW = 0 không cho phép ghi vào tệp thanh ghi nên ngõ ra RD1D = 0 không có giá trị nào được ghi vào thanh ghi x3.
- Từ 60 – 100ns, tín hiệu RegWriteW =1 cho phép ghi vào tệp thanh ghi nên ngõ ra RD1D = 32’hababababab, giá trị được ghi vào thanh ghi x3.

➢ **Trường hợp 3: Từ 100 – 140ns kiểm tra tín hiệu điều khiển lệnh loại tải.**

- Tín hiệu rst ở mức thấp.
- Lệnh ngõ vào InstrD = 32’h00512203 (lw x4 5(x2)) tương ứng lấy giá trị từ địa chỉ bộ nhớ tính bằng giá trị trong thanh ghi x2 cộng với giá trị tức thời 5 và nạp giá trị này vào thanh ghi x4.
- Các tín hiệu điều khiển:
  - RegWriteE = 1 cho phép ghi vào thanh ghi.
  - ResultSrcE = 1 lấy kết quả từ bộ nhớ.
  - MemWriteE = 0 không ghi vào bộ nhớ.
  - BranchE = 0 không nhảy.
  - ALUSrcE = 1 sử dụng giá trị tức thời.
  - ALUControlE = 4’b0000 tương ứng phép cộng.
  - ImmExtE = 32’h00000005 tương ứng mở rộng theo kiểu I
- Địa chỉ thanh ghi:
  - RS1E = 5’h2 tương ứng thanh ghi nguồn x2.
  - RDE = 5’h4 tương ứng thanh ghi đích x4
- Giá trị RD1E = 0 vì không có giá trị nào được ghi vào thanh ghi x2.

➢ **Trường hợp 4: Từ 140 – 180ns kiểm tra tín hiệu điều khiển lệnh loại S.**

- Tín hiệu rst ở mức thấp.
- Lệnh ngõ vào InstrD = 32’h005121a3 (sw x5 3(x2)) lấy giá trị từ thanh ghi x5 và lưu vào địa chỉ bộ nhớ tính bằng giá trị trong thanh ghi x2 cộng với giá trị tức thời 3.

- Các tín hiệu điều khiển:
  - RegWriteE = 0 không cho phép ghi vào thanh ghi.
  - ResultSrcE = 0 không lấy kết quả từ bộ nhớ.
  - MemWriteE = 1 cho phép ghi dữ liệu vào bộ nhớ.
  - BranchE = 0 không nhảy.
  - ALUSrcE = 1 sử dụng giá trị tức thời.
  - ALUControlE = 4'b0000 tương ứng phép cộng.
  - ImmExtE = 32'h00000003 tương ứng mở rộng theo kiểu S
- Địa chỉ thanh ghi:
  - RS1E = 5'h2 tương ứng thanh ghi chứa địa chỉ cơ sở x2.
  - RS2E = 5'h5 tương ứng thanh ghi chứa dữ liệu cần lưu trữ x5
- Giá trị RD1E và RD2E đều bằng 0 vì không có giá trị nào được ghi vào thanh ghi x2 và x5.

➢ **Trường hợp 5: Từ 180 – 220ns kiểm tra tín hiệu điều khiển lệnh loại B.**

- Tín hiệu rst ở mức thấp.
- Lệnh ngõ vào InstrD= 32'hfe208ce3 (beq x1, x2, -8) tương ứng sẽ nhảy
- đến giá trị PC đích nếu hai thanh ghi x1 và x2 bằng nhau.
- Các tín hiệu điều khiển:
  - RegWriteE = 0 không cho phép ghi vào thanh ghi.
  - ResultSrcE = 0 không lấy kết quả từ bộ nhớ.
  - MemWriteE = 0 không cho phép ghi dữ liệu vào bộ nhớ.
  - BranchE = 1 thực hiện nhảy nếu điều kiện so sánh đúng.
  - ALUSrcE = 0 sử dụng giá trị từ thanh ghi
  - ALUControlE = 4'b0001 thực hiện phép trừ để kiểm tra bằng (x1 - x2).
  - ImmExtE = 32'hfffffffc tương ứng -8 mở rộng theo kiểu B
- Địa chỉ thanh ghi:
  - RS1E=5'h1 tương ứng thanh ghi chứa giá trị cần so sánh x1.

- RS2E=5’h2 tương ứng thanh ghi chứa giá trị cần so sánh x2

- Giá trị RD1E và RD2E đều bằng 0 vì không có giá trị nào được ghi vào thanh ghi x1 và x2.

➢ **Trường hợp 6: Từ 220 – 260ns kiểm tra tín hiệu điều khiển lệnh loại I.**

- Tín hiệu rst ở mức thấp.
- Lệnh ngõ vào InstrD = 32’h00208193 (addi x3, x1, 2) tương ứng cộng giá trị tại thanh ghi x1 với giá trị tức thời 2 lưu vào địa chỉ thanh ghi x3.
- Các tín hiệu điều khiển:
  - RegWriteE = 1cho phép ghi vào thanh ghi.
  - ResultSrcE = 0 không lấy kết quả từ bộ nhớ.
  - MemWriteE = 0 không cho phép ghi dữ liệu vào bộ nhớ.
  - BranchE = 0 không nhảy.
  - ALUSrcE = 1 sử dụng giá trị tức thời.
  - ALUControlE = 4’b0000 thực hiện phép cộng.
  - ImmExtE = 32’h00000002 tương ứng mở rộng theo kiểu I
- Địa chỉ thanh ghi:
  - RS1E = 5’h1 tương ứng thanh ghi nguồn x1.
  - RDE = 5’h3 tương ứng thanh ghi đích x3.
- Giá trị RD1E = 0 vì không có giá trị nào được ghi vào thanh ghi x1

➢ **Trường hợp 7: Từ 260 – 300ns kiểm tra tín hiệu điều khiển lệnh loại R.**

- Tín hiệu rst ở mức thấp.
- Lệnh ngõ vào InstrD= 32’h002081b3 (addi x3, x1, x2) tương ứng cộng giá trị tại thanh ghi x1 và x2 sau đó lưu vào địa chỉ tại thanh ghi x3.
- Các tín hiệu điều khiển:
  - RegWriteE = 1cho phép ghi vào thanh ghi.
  - ResultSrcE = 0 không lấy kết quả từ bộ nhớ.
  - MemWriteE = 0 không cho phép ghi dữ liệu vào bộ nhớ.

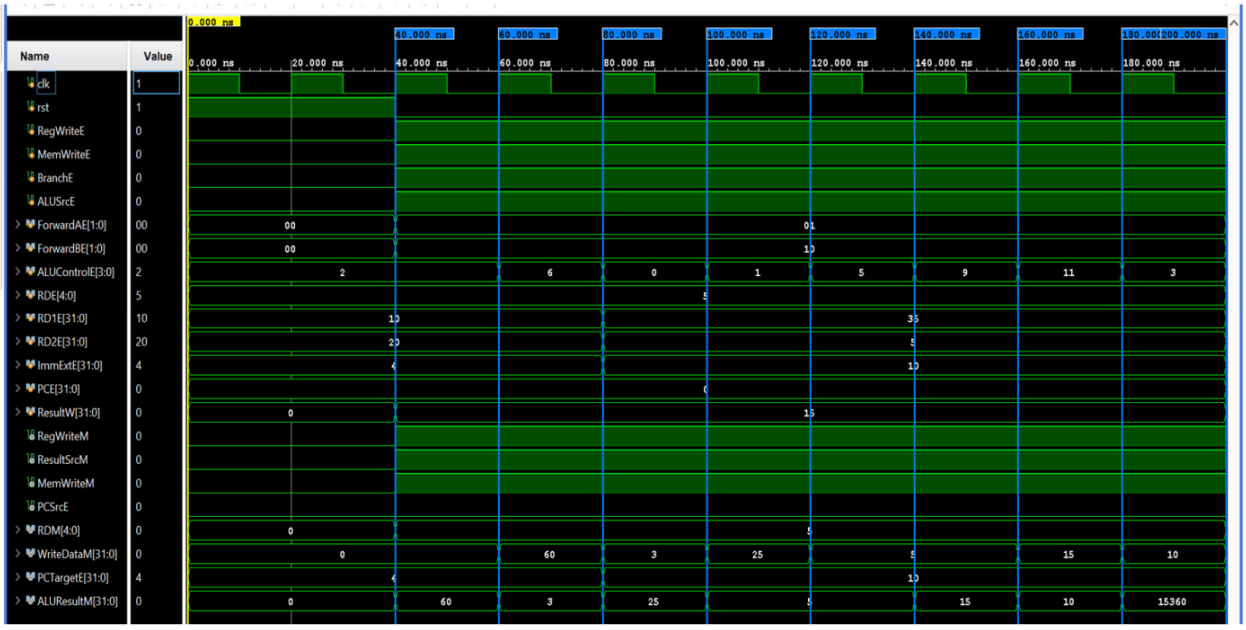
- BranchE = 0 không nhảy.
- ALUSrcE = 1 sử dụng giá trị tức thời.
- ALUControlE = 4’b0000 thực hiện phép cộng.
- ImmExtE không sử dụng.

- Địa chỉ thanh ghi:
  - RS1E = 5’h1 tương ứng thanh ghi nguồn x1.
  - RS2E = 5’h2 tương ứng thanh ghi nguồn x2.
  - RDE = 5’h3 tương ứng thanh ghi đích x3.
- Giá trị RD1E = RD2E = 0 vì không có giá trị nào được ghi vào thanh ghi x1 và x2.

➢ **Trường hợp 8: Từ 100 – 300ns kiểm tra chức năng ghi thanh ghi ID/EX.**

- Tín hiệu PCD là ngõ vào thanh ghi lần lượt được xuất ra thanh ghi bằng ngõ ra PCE tương ứng:
  - Từ 100 – 140ns, PCD = 32’h00000004 = PCE.
  - Từ 140 – 180ns, PCD = 32’h00000008 = PCE.
  - Từ 180 – 220ns, PCD = 32’h0000000c = PCE.
  - Từ 220 – 260ns, PCD = 32’h00000010 = PCE.
  - Từ 260 – 300ns, PCD = 32’h00000014 = PCE.

3. Khối thực thi



Hình 16: Dạng sóng của Khối thực thi

Kết quả mô phỏng của khối thực thi kiểm tra các chức năng tính toán của bộ vi xử lý 32 bit được thể hiện qua Hình 16 như sau:

- **Trường hợp 1: Từ 0 – 40ns kiểm tra chức năng đặt lại.**

  Tín hiệu rst ở mức 1, do đó các ngõ ra của khối được đặt về mức 0

- **Trường hợp 2: Từ 40 – 200ns kiểm tra chức năng tính toán của ALU.**

  Tín hiệu rst ở mức 0, RegWriteE = 1, ResultSrcE = 1, MemWriteE = 1, BranchE = 1, ALUSrcE = 1, ForwardAE = 2'b01, ForwardBE = 2'b10, ResultW = 15, WriteDataM = 25.

  - **Từ 40ns – 60ns: ALUControlE = 4’b0010 (Phép nhân)**
    - Tính toán trong ALU:
      - SrcAE = ResultW = 15 (ForwardAE = 2'b01)
      - SrcBE = ALUResultW = 25 (ForwardBE = 2'b10)
      - SrcBE_out = ImmExtE = 4 (ALUSrcE = 1)
      - ALU thực hiện phép nhân: SrcAE * SrcBE_out = 15 * 4 = 60
      - ALUResultE = 60

- ZeroE = 0 (vì ALUResultE không phải là 0)

▪ Các giá trị đầu ra:

- RegWriteM = 1
- ResultSrcM = 1
- MemWriteM = 1
- ALUResultM = 60
- WriteDataM = SrcBE = 25
- RDM = 5
- PCSrcE = 0 (vì ZeroE = 0 và BranchE = 1)

- **Từ 60ns – 80ns: Thay đổi ALUCOntrolE = 4’b0110 (Phép chia)**

▪ Tính toán trong ALU:

- SrcAE = ResultW = 15 (ForwardAE = 2'b01)
- SrcBE = ALUResultW = 25 (ForwardBE = 2'b10)
- SrcBE_out = ImmExtE = 4 (ALUSrcE = 1)
- ALU thực hiện phép chia: SrcAE / SrcBE_out = 15 / 4 = 3
- ALUResultE = 3
- ZeroE = 0 (vì ALUResultE không phải là 0)

▪ Các giá trị đầu ra:

- RegWriteM = 1
- ResultSrcM = 1
- MemWriteM = 1
- ALUResultM = 3
- WriteDataM = SrcBE = 25
- RDM = 5
- PCSrcE = 0 (vì ZeroE = 0 và BranchE = 1)

- **Từ 80ns – 100ns: Thay đổi ALUControlE = 4’b0000 (Phép cộng) và các giá trị đầu vào RDE1 = 35, RDE2 = 5, ImmExtE = 10.**

- Tính toán trong ALU:
  - SrcAE = ResultW = 15 (ForwardAE = 2'b01)
  - SrcBE = RD2E = 5 (ForwardBE = 2'b10)
  - SrcBE_out = ImmExtE = 10 (ALUSrcE = 1)
  - ALU thực hiện phép cộng: SrcAE + SrcBE_out = 15 + 10 = 25
  - ALUResultE = 25
  - ZeroE = 0 (vì ALUResultE không phải là 0)
- Các giá trị đầu ra:
  - RegWriteM = 1
  - ResultSrcM = 1
  - MemWriteM = 1
  - ALUResultM = 25
  - WriteDataM = SrcBE = 5
  - RDM = 5
  - PCSrcE = 0 (vì ZeroE = 0 và BranchE = 1)

- **Từ 100ns – 120ns: Thay đổi ALUControlE = 4'b0001 (Phép trừ) và các giá trị đầu vào RDE1 = 35, RDE2 = 5, ImmExtE = 10.**
  - Tính toán trong ALU:
    - SrcAE = ResultW = 15 (ForwardAE = 2'b01)
    - SrcBE = RD2E = 5 (ForwardBE = 2'b10)
    - SrcBE_out = ImmExtE = 10 (ALUSrcE = 1)
    - ALU thực hiện phép trừ: SrcAE - SrcBE_out = 15 - 10 = 5
    - ALUResultE = 5
    - ZeroE = 0 (vì ALUResultE không phải là 0)
  - Các giá trị đầu ra:
    - RegWriteM = 1
    - ResultSrcM = 1

- MemWriteM = 1
- ALUResultM = 25
- WriteDataM = SrcBE = 25
- RDM = 5
- PCSrcE = 0 (vì ZeroE = 0 và BranchE = 1)

- **Từ 120ns – 140ns: Thay đổi ALUControlE = 4’b0101 (Phép XOR) và các giá trị đầu vào RDE1 = 35, RDE2 = 5, ImmExtE = 10.**
  - Tính toán trong ALU:
    - SrcAE = ResultW = 15 (ForwardAE = 2'b01)
    - SrcBE = RD2E = 5 (ForwardBE = 2'b10)
    - SrcBE_out = ImmExtE = 10 (ALUSrcE = 1)
    - ALU thực hiện phép XOR: SrcAE XOR SrcBE_out = 15 XOR 10 = 5
    - ALUResultE = 5
    - ZeroE = 0 (vì ALUResultE không phải là 0)
  - Các giá trị đầu ra:
    - RegWriteM = 1
    - ResultSrcM = 1
    - MemWriteM = 1
    - ALUResultM = 5
    - WriteDataM = SrcBE = 5
    - RDM = 5
    - PCSrcE = 0 (vì ZeroE = 0 và BranchE = 1)
- **Từ 140ns – 160ns: Thay đổi ALUControlE = 4’b1001 (Phép OR) và các giá trị đầu vào RDE1 = 35, RDE2 = 5, ImmExtE = 10.**
  - Tính toán trong ALU:
    - SrcAE = ResultW = 15 (ForwardAE = 2'b01)
    - SrcBE = RD2E = 5 (ForwardBE = 2'b10)

- SrcBE_out = ImmExtE = 10 (ALUSrcE = 1)
- ALU thực hiện phép OR: SrcAE OR SrcBE_out = 15 OR 10 = 15
- ALUResultE = 15
- ZeroE = 0 (vì ALUResultE không phải là 0)

▪ Các giá trị đầu ra:

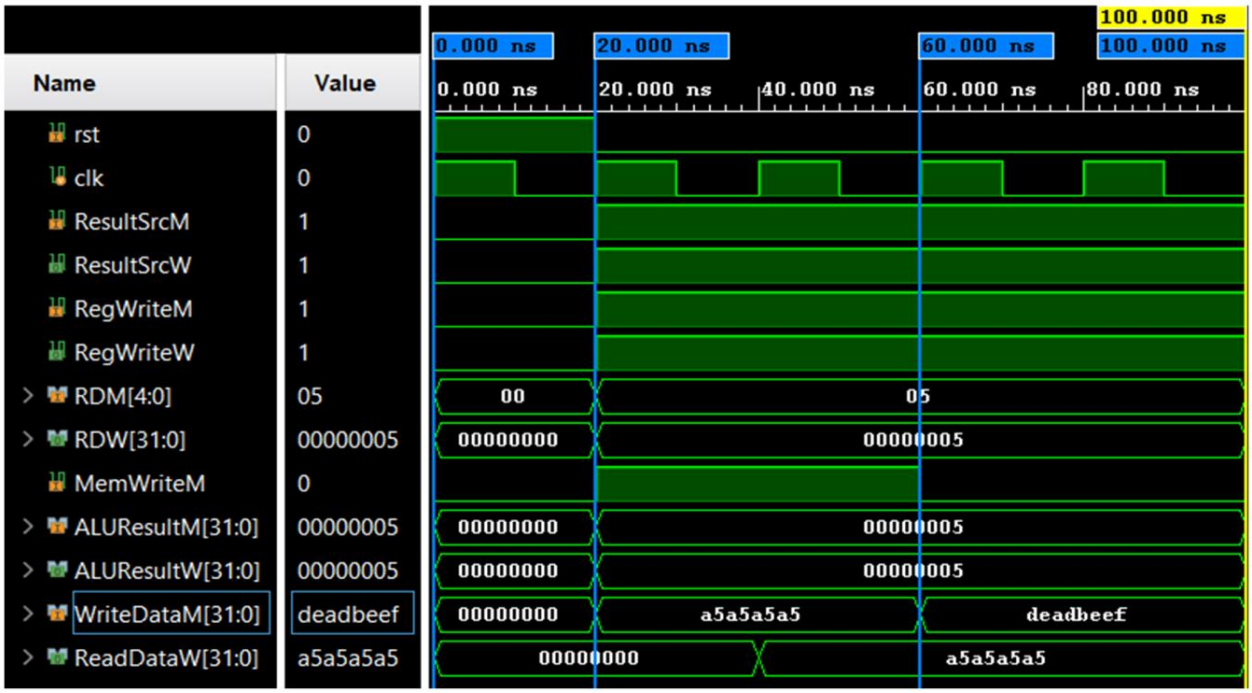
- RegWriteM = 1
- ResultSrcM = 1
- MemWriteM = 1
- ALUResultM = 15
- WriteDataM = SrcBE = 5
- RDM = 5
- PCSrcE = 0 (vì ZeroE = 0 và BranchE = 1)

- **Từ 160ns – 180ns: Thay đổi ALUControlE = 4’b1011 (Phép AND) và các giá trị đầu vào RDE1 = 35, RDE2 = 5, ImmExtE = 10.**
  - Tính toán trong ALU:
    - SrcAE = ResultW = 15 (ForwardAE = 2'b01)
    - SrcBE = RD2E = 5 (ForwardBE = 2'b10)
    - SrcBE_out = ImmExtE = 10 (ALUSrcE = 1)
    - ALU thực hiện phép AND: SrcAE AND SrcBE_out = 15 AND 10 = 10
    - ALUResultE = 10
    - ZeroE = 0 (vì ALUResultE không phải là 0)
  - Các giá trị đầu ra:
    - RegWriteM = 1
    - ResultSrcM = 1
    - MemWriteM = 1
    - ALUResultM = 10
    - WriteDataM = SrcBE = 15

- RDM = 5
- PCSrcE = 0 (vì ZeroE = 0 và BranchE = 1)

- **Từ 180ns – 200ns: Thay đổi ALUControlE = 4’b0011 (Phép dịch) và các giá trị đầu vào RDE1 = 35, RDE2 = 5, ImmExtE = 10.**
  - Tính toán trong ALU:
    - SrcAE = ResultW = 15 (ForwardAE = 2'b01)
    - SrcBE = RD2E = 5 (ForwardBE = 2'b10)
    - SrcBE_out = ImmExtE = 10 (ALUSrcE = 1)
    - ALU thực hiện phép dịch: SrcAE dịch SrcBE_out = 15 dịch 10 = 15360
    - ALUResultE = 15360
    - ZeroE = 0 (vì ALUResultE không phải là 0)
  - Các giá trị đầu ra:
    - RegWriteM = 1
    - ResultSrcM = 1
    - MemWriteM = 1
    - ALUResultM = 15360
    - WriteDataM = SrcBE = 10
    - RDM = 5
    - PCSrcE = 0 (vì ZeroE = 0 và BranchE = 1)

4. Khối truy cập bộ nhớ



Hình 17: Dạng sóng của Khối truy cập bộ nhớ

Kết quả mô phỏng của Khối truy cập bộ nhớ truy xuất dữ liệu từ bộ nhớ hoặc lưu dữ liệu vào bộ nhớ được thể hiện qua Hình 17 như sau:

- **Trường hợp 1: Từ 0 – 20ns kiểm tra chức năng đặt lại.**

  Tín hiệu rst ở mức 1, do đó các ngõ ra của khối được đặt về mức 0.

- **Trường hợp 2: Từ 20 – 60ns kiểm tra chức năng đọc ghi dữ liệu bộ nhớ.**

- Tín hiệu rst = 0.
- Các tín hiệu điều khiển và dữ liệu:
  - MemWriteM = 1 cho phép ghi dữ liệu vào bộ nhớ.
  - ALUResultM = 32'd5 là địa chỉ bộ nhớ để ghi.
  - WriteDataM = 32'hA5A5A5A5 là dữ liệu cần ghi vào bộ nhớ.
- Các giá trị:
  - ReadDataW = 32’ha5a5a5a5 cho thấy địa chỉ được lưu vào bộ nhớ và trích xuất ra chậm 1 xung đồng hồ vì bộ nhớ cần 1 chu kỳ xung để hoàn thành quá trình đọc dữ liệu xuất ra.

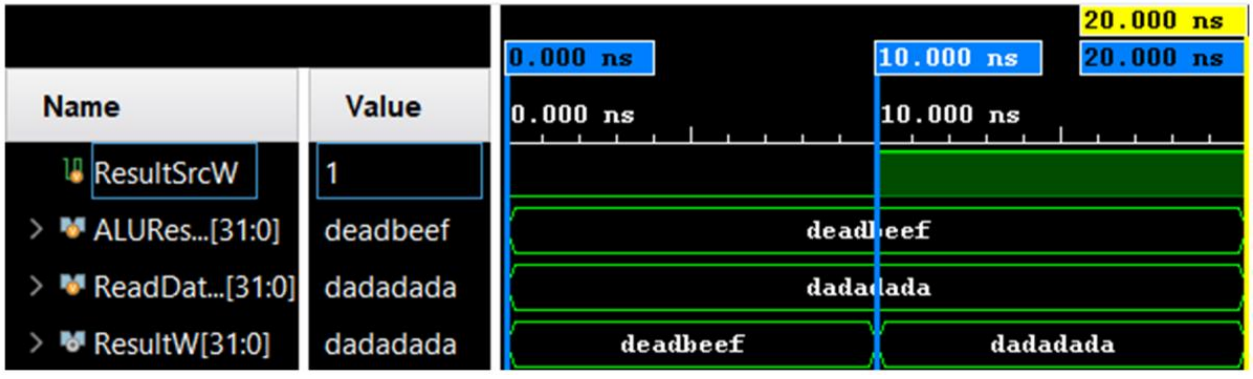
➢ **Trường hợp 3: Từ 60 – 100ns kiểm tra chức năng cho phép ghi dữ liệu vào bộ nhớ.**

- Tín hiệu rst = 0.
- Các tín hiệu điều khiển và dữ liệu:
  - MemWriteM = 0 không cho phép ghi dữ liệu vào bộ nhớ.
  - ALUResultM = 32'd5 là địa chỉ bộ nhớ để ghi.
  - WriteDataM = 32'hDEADBEEF là dữ liệu cần ghi vào bộ nhớ.
- Các giá trị:
  - ReadDataW = 32’ha5a5a5a5 cho thấy dữ liệu mới không được lưu vào bộ nhớ do tín hiệu MemWriteM = 0 không cho phép ghi dữ liệu vào bộ nhớ.

➢ **Trường hợp 4: Từ 20 – 80ns kiểm tra chức năng cho thanh ghi.**

- Các tín hiệu ResultSrcM, RegWriteM, RDM đều được xuất ra ngõ ra qua thanh ghi MEM/WB tương ứng các giá trị:
  - ResultSrcM = ResultSrcW = 1.
  - RegWriteM = RegWriteW =1.
  - RDM = RDW = 5’h05.

5. Khối ghi lại

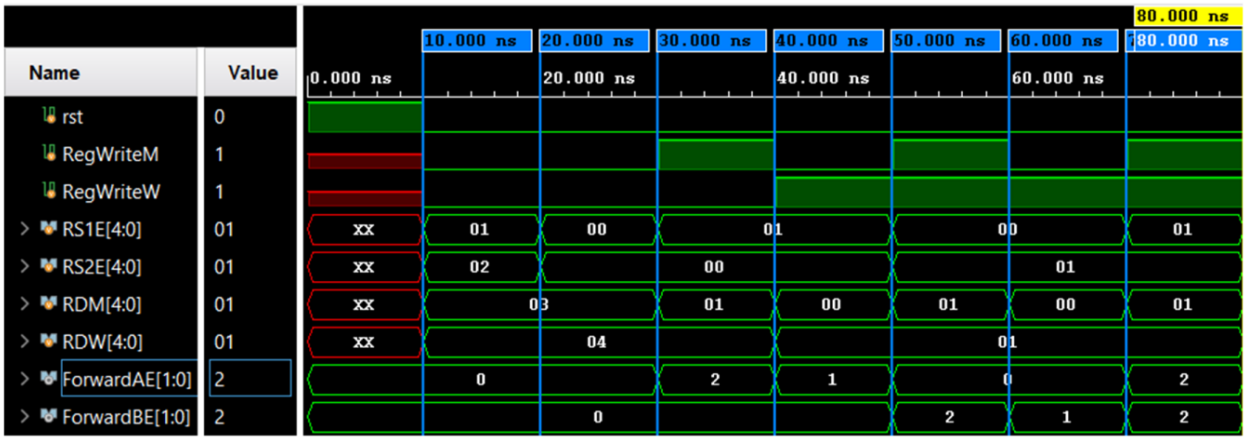


Hình 18: Dạng sóng của Khối ghi lại

- Từ 0 – 10ns: Tín hiệu ResultSrcW = 0 chọn giá trị ALU làm đầu ra, ResultW = ALUResultW = 32’hdeadbeef.

- Từ 10 – 20ns: Tín hiệu ResultSrcW = 1 chọn giá trị bộ nhớ làm đầu ra, ResultW = ReadDataW= 32’hdadadada.

6. Khối xử lý xung đột



Hình 19: Dạng sóng của Khối xử lý xung đột

- **Trường hợp 1: Từ 0ns – 10ns kiểm tra tín hiệu reset của hệ thống**

  Khi reset (rst = 1) ở mức cao, tất cả các tín hiệu đều đặt về 2'b00, không có chuyển tiếp nào xảy ra

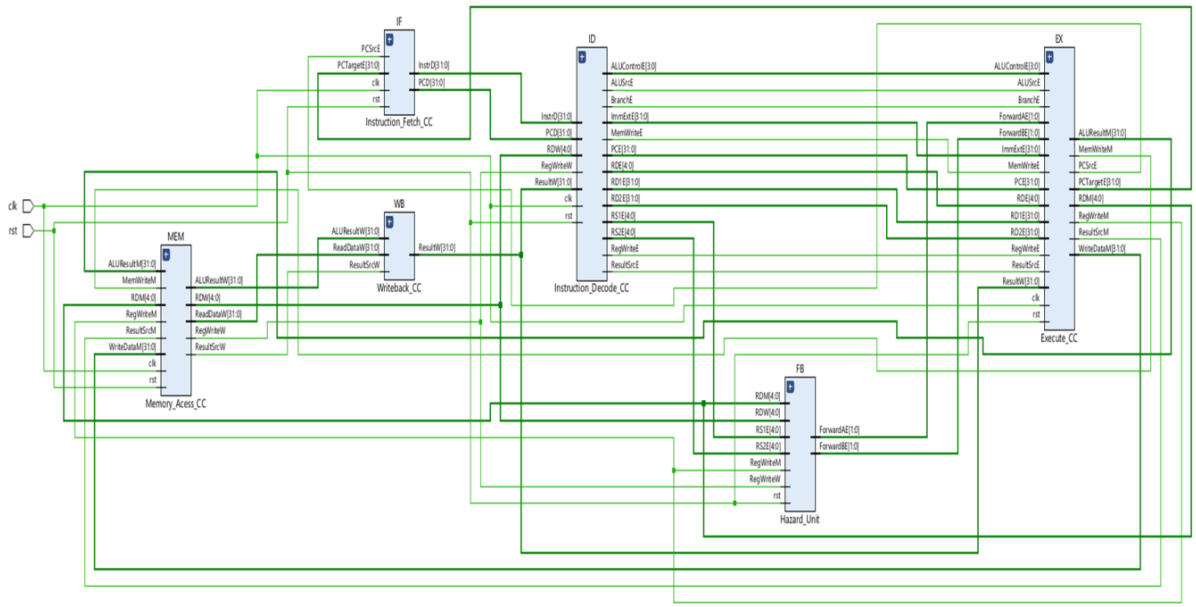
- **Trường hợp 2: Từ 10ns – 30ns không có tín hiệu nào được chuyển tiếp**
  - Từ 10 – 20ns, tín hiệu đầu vào là: rst = 0, RegWriteM = RegWriteW = 0, không có ghi vào thanh ghi ở giai đoạn MEM hoặc WB, nên không có chuyển tiếp.
  - Từ 20 – 30ns, tín hiệu đầu vào là: rst = 0, RDW = RDM = 0 không có chuyển tiếp vì cả hai thanh ghi đều là thanh ghi x0.
- **Trường hợp 3: Từ 30ns – 50ns Chuyển tiếp với toán tử thứ nhất.**
  - Từ 30 – 40ns, tín hiệu đầu vào là: rst = 0, RegWriteM = 1, RDM = RS1E = 5’h1 và khác không nên ForwardAE =2’b10, chuyển tiếp từ giai đoạn truy cập bộ nhớ.
  - Từ 40 – 50ns, tín hiệu đầu vào là: rst = 0, RegWriteW = 1, RDW = RS1E = 5’h1 và khác không nên ForwardAE =2’b01, chuyển tiếp từ giai đoạn ghi lại.

- **Trường hợp 4: Từ 50ns – 70ns Chuyển tiếp với toán tử thứ hai.**
  - Từ 50 – 60ns, tín hiệu đầu vào là: rst = 0, RegWriteM = 1, RDM = RS2E = 5’h1 và khác không nên ForwardBE =2’b10, chuyển tiếp từ giai đoạn truy cập bộ nhớ.
  - Từ 60 – 70ns, tín hiệu đầu vào là: rst = 0, RegWriteW = 1, RDW = RS2E = 5’h1 và khác không nên ForwardBE =2’b01, chuyển tiếp từ giai đoạn ghi lại.
- **Trường hợp 5: Từ 70ns – 80ns có tín hiệu được chuyển tiếp từ cả 2 giai đoạn (Giai đoạn truy cập bộ nhớ có mức độ ưu tiên hơn).**

  Tín hiệu đầu vào là: rst = 0, RegWriteM = RegWriteW = 1, RS1E = RS2E = 5'h1, RDM = RDW = 5'h1.

  Khi cả RegWriteM và RegWriteW đều bằng 1 và cả RDM và RDW đều bằng RS1E và RS2E , tín hiệu ForwardAE và ForwardBE được đặt là 2'b10 để chuyển tiếp từ giai đoạn truy cập bộ nhớ vì giai đoạn truy cập bộ nhớ có độ ưu tiên cao hơn.

III. Kết quả mô phỏng toàn hệ thống

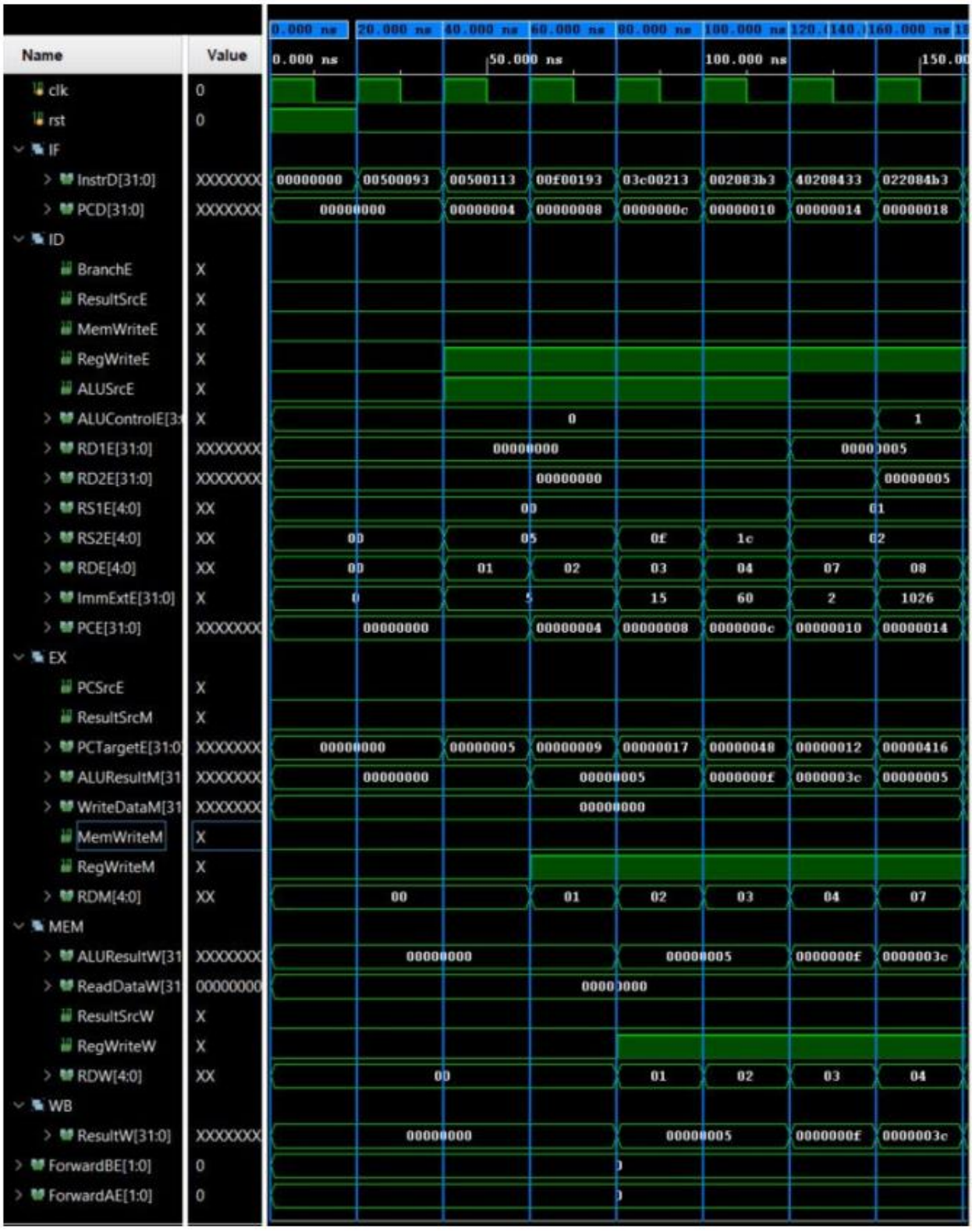


Hình 20: Sơ đồ RTL của Vi xử lý 32 bit

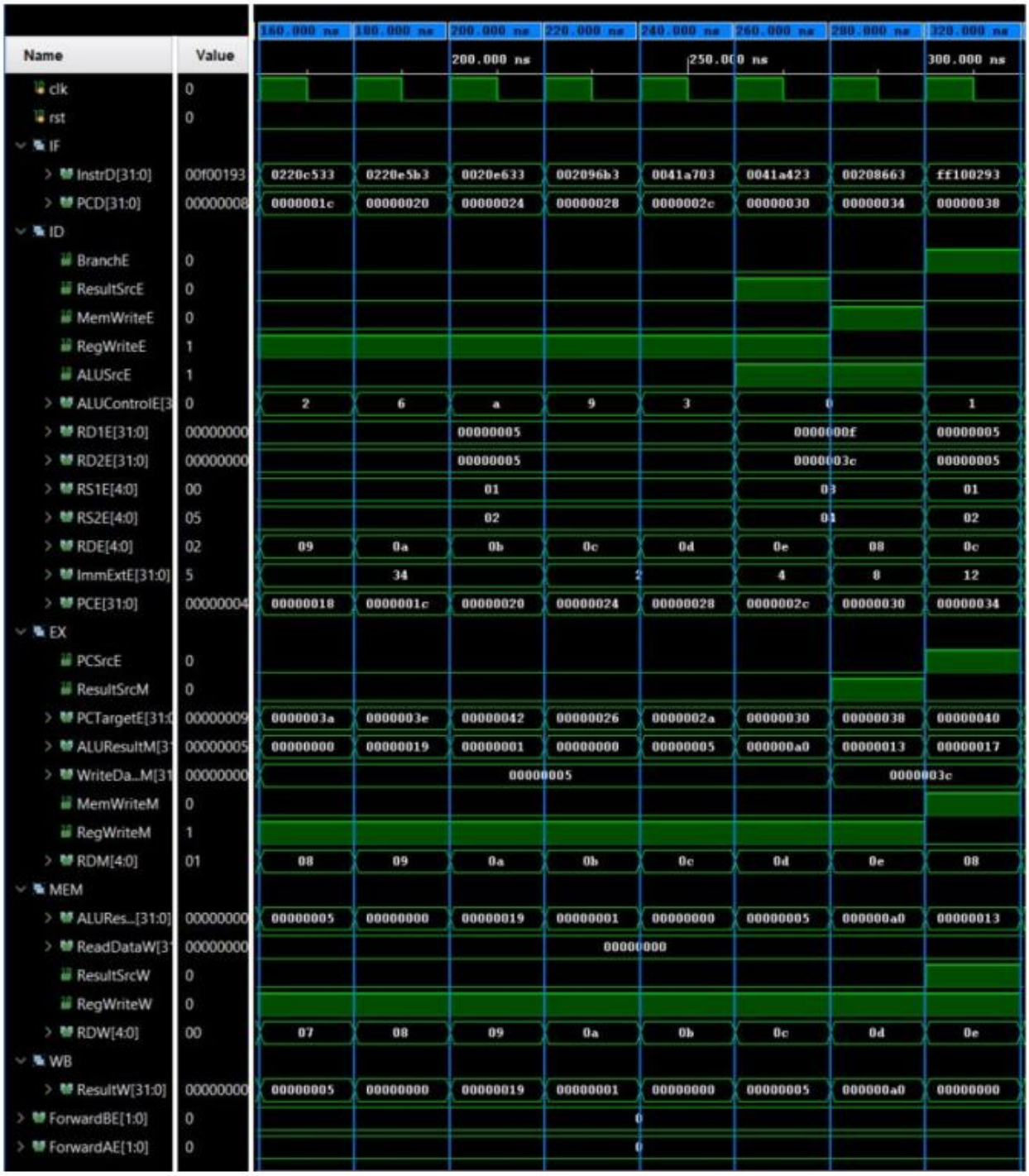
Bảng 5: Mã Assembly dùng mô phỏng chương trình

Nhãn	Mã Assembly	Mã HEX
	addi x1, x0, 5	0x00500093
	addi x2, x0, 5	0x00500113
	addi x3, x0, 15	0x00F00193
	addi x4, x0, 60	0x03C00213
label:	add x7, x1, x2	0x002083B3
	sub x8, x1, x2	0x40208433
	mul x9, x1, x2	0x022084B3
	div x10, x1, x2	0x0220C533
	rem x11, x1, x2	0x0220E5B3
	or x12, x1, x2	0x0020E633
	sll x13, x1, x2	0x002096B3

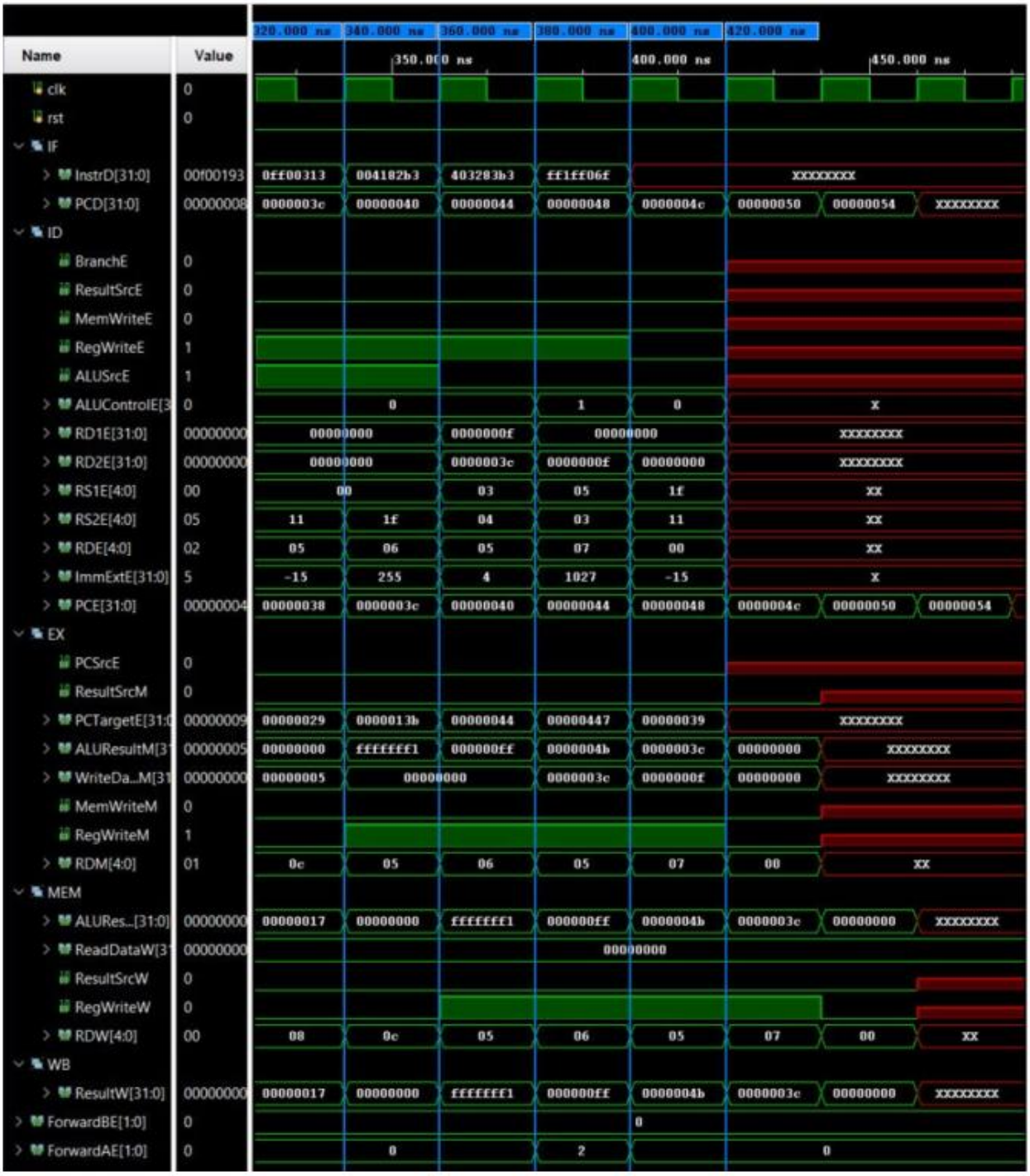
Nhãn	Mã Assembly	Mã HEX
	lw x14, 4(x3)	0x0041A703
	sw x4, 8(x3)	0x0041A423
	beq x1, x2, tag	0x00208663
end:	addi x5, x0, -15	0xFF100293
	addi x6, x0, 255	0x0FF00313
tag:	add x5, x3, x4	0x004182B3
	sub x7, x5, x3	0x403283B3
	j end	0xFF1FF06F



Hình 21: Dạng sóng của bộ vi xử lý 32 bit (từ 0 – 160ns)



Hình 22: Dạng sóng của bộ vi xử lý 32 bit (từ 160 – 320ns)



Hình 23: Dạng sóng của bộ vi xử lý 32 bit (từ 320 – 480ns)

Kết quả mô phỏng được trình bày trong các Hình 21, Hình 22 và Hình 23 dựa vào Bảng 5.

- **Trường hợp 1: kiểm tra chức năng đặt lại.**

  Từ 0 – 20ns, tín hiệu rst ở mức 1 nên tất cả ngõ ra được đặt về mức 0.

- **Trường hợp 2: kiểm tra các lệnh loại I khởi tạo giá trị.**

  Từ 20 – 120ns thực hiện lệnh loại I tính toán gồm 5 chu kỳ xung:

- Chu kỳ xung thứ nhất, Khối lấy lệnh lấy lệnh InstrD = 32'h00500093 là lệnh cộng, tương ứng địa chỉ PCD = 32'h00000000.
- Chu kỳ xung thứ hai, Khối giải mã giải mã lệnh ra thành các tín hiệu điều khiển: BranchE = 0, ResultSrcE = 0, MemWriteE = 0, RegWriteE = 1, ALUSrcE = 1, ALUControlE = 4'h0; các địa chỉ thanh ghi RS1E = 5'0, RDE = 5'h1; giá trị RD1E = 32'd0, ImmExtE =32d5, PCE = 32'h0000000.
- Chu kỳ xung thứ ba, Khối thực thi thực hiện tính toán phép cộng. Vì ALUSrcE = 1 nên chọn ngõ vào thứ hai ALU lấy từ ImmExtE = 32'd5. Lệnh không bị xung đột dữ liệu (Forward AE = 0) nên ngõ vào thứ nhất ALU lấy từ thanh ghi RD1E = 32'd0. Ngõ ra ALUResultM = RD1E + ImmExtE = 32'd5.
- Chu kỳ xung thứ tư, Khối truy cập bộ nhớ chuyển kết quả ALUResultM tới Khối ghi lại thông qua thanh ghi MEM/WB. Lệnh tính toán loại I không cần ghi vào bộ nhớ.
- Chu kỳ xung thứ năm, Khối ghi lại ghi giá trị vào Tệp thanh ghi. Vì ResultSrcW = 0 nên ngõ ra ResultW chọn ngõ vào là ALUResultW = 32'd5 ghi vào thanh ghi x1.
- Khối xử lý xung đột không phát hiện xung đột nên ForwardAE = ForwardBE = 0.
- Tương tự với các lệnh có địa chỉ PCD = 0x00000004, 0x00000008, 0x0000000c thực hiện lần lượt phép cộng thanh ghi x2 với 32'd5, thanh ghi

x3 với 32’d15, thanh ghi x4 với 32’d60 trong khoảng thời gian từ 40 – 140ns , 60 – 160ns, 80 – 180ns.

- **Trường hợp 3: kiểm tra các lệnh loại R tính toán phép tính số học và luận lý**

Từ 100 – 200ns thực hiện lệnh cộng loại R tính toán gồm 5 chu kỳ xung:

- Chu kỳ xung thứ nhất, Khối lấy lệnh lấy lệnh InstrD = 32’h002083b3 là lệnh cộng, tương ứng địa chỉ PCD = 32’h00000010.
- Chu kỳ xung thứ hai, Khối giải mã giải mã lệnh ra thành các tín hiệu điều khiển: BranchE = 0, ResultSrcE = 0, MemWriteE = 0, RegWriteE = 1, ALUSrcE = 0, ALUControlE = 4’h0; các địa chỉ thanh ghi RS1E = 5’1, RS2E =5’h2, RDE = 5’h7; giá trị RD1E = 32’d5, RD2E = 32’d0, PCE = 32’h0000010.
- Chu kỳ xung thứ ba, Khối thực thi thực hiện tính toán phép cộng. Lệnh không bị xung đột dữ liệu (Forward AE = 0, ForwardBE = 0) và ALUSrcE = 0 nên cả hai ngõ vào ALU lấy từ thanh ghi RD1E = 32’d5 và RD2E = 32’d0. Ngõ ra ALUResultM = RD1E + RD2E = 32’d5.
- Chu kỳ xung thứ tư, Khối truy cập bộ nhớ chuyển kết quả ALUResultM tới Khối ghi lại thông qua thanh ghi MEM/WB. Lệnh tính toán loại R không cần ghi vào bộ nhớ.
- Chu kỳ xung thứ năm, Khối ghi lại ghi giá trị vào Tệp thanh ghi. Vì ResultSrcW = 0 nên ngõ ra ResultW chọn ngõ vào là ALUResultW = 32’d5 ghi vào thanh ghi x7.
- Khối xử lý xung đột không phát hiện xung đột nên ForwardAE = ForwardBE = 0.
- Phép tính trừ , nhân, chia lấy nguyên, chia lấy dư, or, dịch được thực hiện tương tự trong khoảng thời gian lần lượt là:

- Từ 120 – 220ns thực hiện phép trừ x8 = x1 – x2 = 32’d5 - 32’d5 = 32’d0.
- Từ 140 – 240ns thực hiện phép nhân
  x9 = x1 * x2 = 32’d5 * 32’d5 = 32’d25.
- Từ 160 – 260ns thực hiện phép chia lấy phần nguyên
  x10 = x1 / x2 = 32’d5 / 32’d5 = 32’d1.
- Từ 180 – 280ns thực hiện phép chia lấy phần dư
  x11 = x1 % x2 = 32’d5 % 32’d5 = 32’d0.
- Từ 200 – 300ns thực hiện phép or x12 = x1 | x2 = 32’d5 | 32’d5 =32’d5.
- Từ 220 – 320ns thực hiện phép dịch trái
  x13 = x1 << x2 = 32’d5 << 32’d5 = 32’d160.

➢ **Trường hợp 4: kiểm tra các lệnh tải.**

Từ 240 – 340ns thực hiện lệnh tải tính toán gồm 5 chu kỳ xung:

- Chu kỳ xung thứ nhất, Khối lấy lệnh lấy lệnh InstrD = 32’h0041a703 là lệnh tải, tương ứng địa chỉ PCD = 32’h0000002c.
- Chu kỳ xung thứ hai, Khối giải mã giải mã lệnh ra thành các tín hiệu điều khiển: BranchE = 0, ResultSrcE = 1, MemWriteE = 0, RegWriteE = 1, ALUSrcE = 1, ALUControlE = 4’h0; các địa chỉ thanh ghi RS1E = 5’h3, giá trị RD1E = 32’d15, ImmExt = 32’d4.
- Chu kỳ xung thứ ba, Khối thực thi thực hiện tính toán địa chỉ bộ nhớ cần truy cập. Lệnh không bị xung đột dữ liệu (Forward AE = 0, ForwardBE = 0) và ALUSrcE = 1 nên lấy ngõ vào ALU thứ nhất từ thanh ghi RD1E = 32’d15 và ImmExtE = 32’d4. Ngõ ra ALUResultM = RD1E + ImmExtE = 32’d19.
- Chu kỳ xung thứ tư, Khối truy cập bộ nhớ chuyển kết quả ALUResultM tới Khối ghi lại thông qua thanh ghi MEM/WB. Lệnh tính toán tải không cần ghi vào bộ nhớ.

- Chu kỳ xung thứ năm, Khối ghi lại ghi giá trị vào Tệp thanh ghi. Vì ResultSrcW = 1 nên ngõ ra ResultW chọn ngõ vào là ReadDataW = 32’d0 tương ứng địa chỉ 32’d19 và ghi vào thanh ghi x14.
- Khối xử lý xung đột không phát hiện xung đột nên ForwardAE = ForwardBE = 0.

➢ **Trường hợp 5: kiểm tra các lệnh lưu trữ.**

Từ 260 – 360ns thực hiện lệnh tải tính toán gồm 5 chu kỳ xung:

- Chu kỳ xung thứ nhất, Khối lấy lệnh lấy lệnh InstrD = 32’h041a423 là lệnh lưu trữ, tương ứng địa chỉ PCD = 32’h00000030.
- Chu kỳ xung thứ hai, Khối giải mã giải mã lệnh ra thành các tín hiệu điều khiển: BranchE = 0, ResultSrcE = 0, MemWriteE = 1, RegWriteE = 0, ALUSrcE = 1, ALUControlE = 4’h0; các địa chỉ thanh ghi RS1E = 5’h3, RS2E = 5’h4, giá trị RDE = 32’d8, RD1E = 32’d15, RD2E =32’d60, ImmExtE = 32’d8.
- Chu kỳ xung thứ ba, Khối thực thi thực hiện tính toán địa chỉ bộ nhớ cần truy cập. Lệnh không bị xung đột dữ liệu (Forward AE = 0, ForwardBE = 0) và ALUSrcE = 1 nên lấy ngõ vào ALU thứ nhất từ thanh ghi RD1E = 32’d15 và ImmExt = 32’d4. Ngõ ra ALUResultM = RD1E + ImmExtE = 32’d23.
- Chu kỳ xung thứ tư, Khối truy cập bộ nhớ truy cập vào địa chỉ 32’d23 tại bộ nhớ và dữ liệu từ thanh ghi x4 là RD2E = 32’60 được lưu vào bộ nhớ tại địa chỉ 32’d23
- Chu kỳ xung thứ năm, Khối ghi lại không dùng vì không có hoạt động ghi vào thanh ghi trong giai đoạn này.
- Khối xử lý xung đột không phát hiện xung đột nên ForwardAE = ForwardBE = 0.

- **Trường hợp 6: kiểm tra các lệnh loại B.**

  Từ 280 – 380ns thực hiện lệnh tải tính toán gồm 5 chu kỳ xung:

- Chu kỳ xung thứ nhất, Khối lấy lệnh lấy lệnh InstrD = 32’h00208663 là lệnh nhảy, tương ứng địa chỉ PCD = 32’h00000034.
- Chu kỳ xung thứ hai, Khối giải mã giải mã lệnh ra thành các tín hiệu điều khiển: BranchE = 1, ResultSrcE = 0, MemWriteE = 0, RegWriteE = 0, ALUSrcE = 0, ALUControlE = 4’h1; các địa chỉ thanh ghi RS1E = 5’h1, RS2E = 5’h2, giá trị RDE = 32’d12, RD1E = 32’d5, RD2E =32’d5, ImmExtE = 32’d12.
- Chu kỳ xung thứ ba, Khối thực thi thực hiện so sánh giá trị của x1 và x2 bằng cách lấy x1 - x2 = 32’d5 – 32’d5 = 32’d0. x1 – x2 = 0 tương ứng x1 = x2 nên tính toán địa chỉ đích của nhánh PctargetE = PCE + ImmExtE = 32’h00000034 + 32’h00000012 (32’d12) = 32’h00000040.
- Chu kỳ xung thứ tư, vì là lệnh nhảy nên không thực hiện truy cập bộ nhớ.
- Chu kỳ xung thứ năm, vì là lệnh nhảy nên không thực hiện ghi vào thanh ghi.
- Khối xử lý xung đột không phát hiện xung đột nên ForwardAE = ForwardBE = 0.
- Sau 1 chu kỳ, giá trị PCD = PCTargetE = 32’h00000040.

⇨ Khoảng thời gian tiếp theo hệ thống tiếp tục lấy lệnh như mã assemble PCD = 32’h000000044, PCD = 32’h00000048.