ĐẠI HỌC QUỐC GIA TP.HỒ CHÍ MINH
TRƯỜNG ĐẠI HỌC BÁCH KHOA KHOA ĐIỆN – ĐIỆN TỬ
BỘ MÔN ĐIỆN TỬ

o0o

***

BÁO CÁO

BÀI TẬP LỚN CẤU TRÚC MÁY TÍNH

LAB2:
ALU and Branch Comparison

GVHD:Thầy Trần Hoàng Linh

SVTT:Dương Nhật Huy – 1810162

1. Thiết kế ALU

a. Phân tích

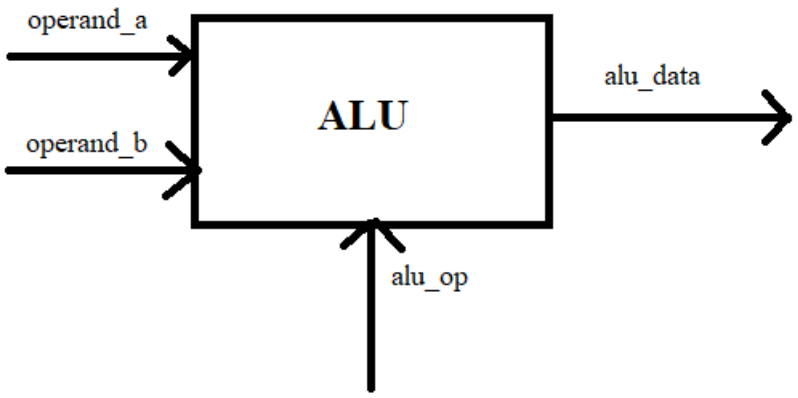
Trong Risc – V , ALU có vai trò thực hiện các phép toán số học và phép toán thao tác bit trên dãy số nhị phân.

Bảng dưới đây là các phép toán mà ALU cần phải thực hiện

alu_op	funct7	funct3	Description (R type)	Description (I type)
ADD	0x00	0x0	rd = rs1 + rs2	rd = rs1 + imm
SUB	0x20	0x0	rd = rs1 – rs2	n/a
SLT	0x00	0x2	rd = (rs1 < rs2)? 1 : 0	rd = (rs1 < imm)? 1 : 0
SLTU	0x00	0x3	rd = (rs1 < rs2)? 1 : 0	rd = (rs1 < imm)? 1 : 0
XOR	0x00	0x4	rd = rs1 ^ rs2	rd = rs1 ^ imm
OR	0x00	0x6	rd = rs1 | rs2	rd = rs1 | imm
AND	0x00	0x7	rd = rs1 & rs2	rd = rs1 & imm
SLL	0x00	0x1	rd = rs1 << rs2[4:0]	rd = rs1 << imm[4:0]
SRL	0x00	0x5	rd = rs1 >> rs2[4:0]	rd = rs1 >> imm[4:0]
SRA	0x20	0x5	rd = rs1 >>> rs2[4:0]	rd = rs1 >>> imm[4:0]

b. Thiết kế

i. Sơ đồ khối



ii. Mô tả thiết kế

*) Input :

- operand_a , operand_b : 2 toán hạng 32 bit
- alu_op : Tín hiệu 4 bit ,chọn phép toán thực hiện trong ALU
  + 0000: ADD
  + 0001: SUB
  + 0010: SLL
  + 0011: SLT
  + 0100: SLTU
  + 0101: XOR
  + 0110: SRL

+ 0111: SRA
+ 1000: OR
+ 1001: AND
+ 1010: LUI

*) Output:

- alu_data: Kết quả của phép toán

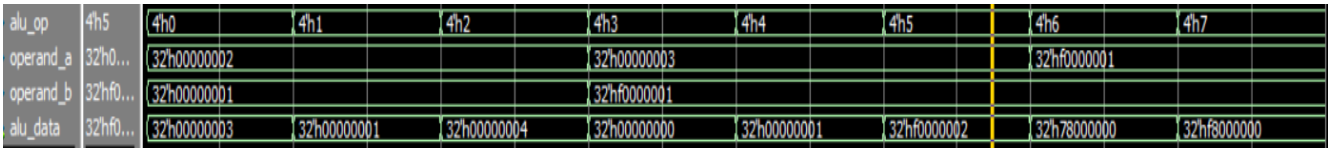
*) Giải thuật : Sử dụng các phép toán có sẵn trong Symstem Verilog ngoại trừ các phép toán '- ' , ' < ', '>' theo yêu cầu của Lab2. Ta sử dụng các giải thuật thay thế sau

- SUB : Phép trừ của a và b chính là phép cộng của a và số bù 2 của b nên a – b được thay thế là a+ ~b + 1;
- SLT : Phép so sánh có dấu giữa a và b . Ta lấy a + ~b + 1(a – b) . Nếu kết quả là số âm(bit 31 của kết quả là 1) thì ngõ ra trả về là 1 . Ngược lại là 0.
- SLTU: Phép so sánh không dấu giữa a và b. Ta kiểm tra bit 31 của a và b . Nếu 2 bit này khác nhau (a[31] ^ b[31] = 1) thì ngõ ra trả về chính là ~a[31] .Nếu 2 bit này giống nhau, ta đưa trở về phép so sánh có dấu giữa a[30:0] và b[30:0]. Lấy a[30:0] – b[30:0] , Nếu kết quả là số âm(bit 31 của kết quả là 1) thì ngõ ra trả về là 1 . Ngược lại là 0.

c. Kiểm tra thiết kế :

Lần lượt thay đổi ngõ vào operand_a , operand_b , alu_op . Kiểm tra ngõ ra alu_data của thiết kế đã chính xác chưa.

d. Kết quả mô phỏng:



e. Kết luận

Thiết kế chạy đúng với yêu cầu

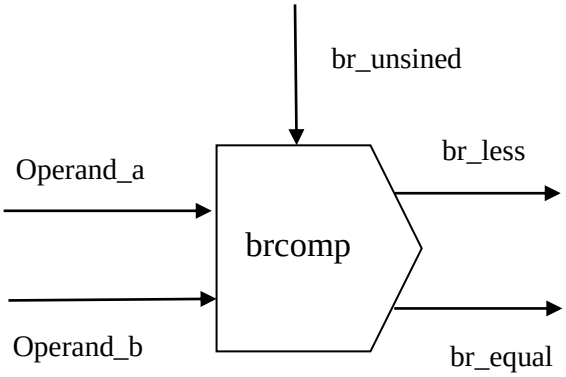
2. Thiết kế Branch Comparison

a. Phân tích

Trong Risc – V , Branch Comparison có vai trò thực hiện các phép so sánh giữa 2 ngõ vào để thực hiện các lệnh nhảy có điều kiện

b. Thiết kế

i. Sơ đồ khối



ii. Mô tả thiết kế

*) Input :

- operand_a , operand_b : 2 toán hạng 32 bit
- br_unsined : 1 nếu 2 toán hạng là số không dấu, 0 nếu là số có dấu

*) Output:

- br_equal : 1 nếu operand_a = operand_b
- br_less: 1 nếu operand_a < operand_b

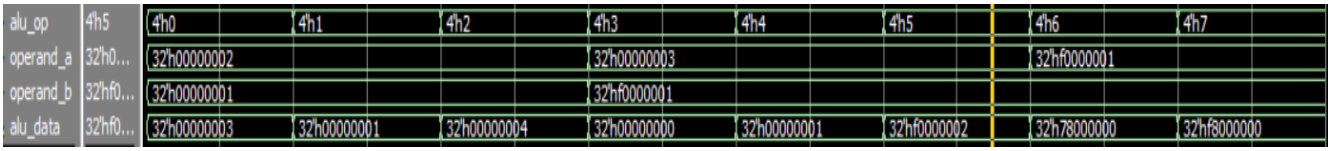
*) Giải thuật : Sử dụng các phép toán có sẵn trong Symstem Verilog ngoại trừ các phép toán '- ' , ' < ', '>' theo yêu cầu của Lab2. Ta sử dụng các giải thuật thay thế sau

- br_unsined = 0 : Phép so sánh có dấu giữa a và b . Ta lấy a + ~b + 1(a – b) . Nếu kết quả là số âm(bit 31 của kết quả là 1) thì ngõ ra trả về là 1 . Ngược lại là 0.
- br_unsined =1 : Phép so sánh không dấu giữa a và b. Ta kiểm tra bit 31 của a và b . Nếu 2 bit này khác nhau (a[31] ^ b[31] = 1) thì ngõ ra trả về chính là ~a[31] .Nếu 2 bit này giống nhau, ta đưa trở về phép so sánh có dấu giữa a[30:0] và b[30:0]. Lấy a[30:0] – b[30:0] , Nếu kết quả là số âm(bit 31 của kết quả là 1) thì ngõ ra trả về là 1 . Ngược lại là 0.

c. Kiểm tra thiết kế :

Lần lượt thay đổi ngõ vào operand_a , operand_b , br_unsigned . Kiểm tra ngõ ra của thiết kế đã chính xác chưa.

d. Kết quả mô phỏng:



e. Kết luận

Thiết kế chạy đúng với yêu cầu