***ALU***

Functional description

Selects imm\_data\_sgnext as operand 2 when Sel\_mux\_oprnd2 = 1

Add op1 and op2

|  |  |
| --- | --- |
| Functionality | Select signals |
| Add | Sel\_mux\_wb[0:1] = “11” Sel\_mux\_neg = ‘0’ |
| Sub | Sel\_mux\_wb[0:1] = “11” Sel\_mux\_neg = ‘1’ |
| And | Sel\_mux\_wb[0:1] = “10” Sel\_mux\_neg = ‘0’ Sel\_mux\_andor =’0’ |
| Or | Sel\_mux\_wb[0:1] = “10” Sel\_mux\_neg = ‘0’ Sel\_mux\_andor =’1’ |
| Nor | Sel\_mux\_wb[0:1] = “10” Sel\_mux\_neg = ‘1’ Sel\_mux\_andor =’1’ |
| SHR | Sel\_mux\_wb[0:1] = “01” |
| SHL | Sel\_mux\_wb[0:1] = “00” |

Branching signal out\_sel\_brnch2pc asserted

with op1 < op2 when BLT

with op1 = op2 when BEQ

with op1 != op2 when BNE

***PC:***

Pc <= pc +1 as DEFAULT

Pc = pc + imm\_add when BRANCH

Pc = pc+1 (31:28 ) & imm\_add &”00” when JUMP

Pc = pc when HALT

Test cases for ALU

-- insert stimulus here

in\_data\_oprnd1 <= x"00000002";

in\_data\_oprnd\_Rt <= x"00000003";

in\_data\_imm\_sgnext <= x"00000001";

wait for 20 ns;

in\_sel\_mux\_wb <= "11"; --right shift

wait for 20 ns;

in\_sel\_mux\_wb <= "10"; --add

wait for 20 ns;

in\_sel\_mux\_wb <= "01"; --and

wait for 20 ns;

in\_sel\_mux\_wb <= "11"; --add

wait for 20 ns;

in\_sel\_mux\_neg <= '1'; -- sub rt

wait for 20 ns;

in\_sel\_mux\_oprnd2 <= '1'; --sub imm

wait for 20 ns;

in\_sel\_mux\_andor <= '1'; --nor

in\_sel\_mux\_wb <= "10";

wait for 20 ns;

in\_sel\_mux\_neg <= '0';

wait for 20 ns;

in\_sel\_mux\_oprnd2 <= '0';

wait for 20 ns;

in\_sel\_mux\_andor <= '0';

wait for 20 ns;

in\_sel\_mux\_wb <= "00"; --left shift

***Test case for PC:***

-- insert stimulus here

clr <= '1';

wait for clk\_period;

clr <= '0';

wait for 10\*clk\_period; -- default case pc increement

in\_data\_imm\_sgnext <= x"00000026";

in\_address\_imm <= "00000000000000000000111111";

in\_sel\_mux\_brnch <= '1'; -- braching

wait for clk\_period;

in\_sel\_mux\_brnch <= '0';

wait for 10\*clk\_period; -- normal incremant

in\_sel\_mux\_jmp <= '1'; -- jumping

wait for clk\_period;

in\_sel\_mux\_jmp <= '0';

wait for 10\*clk\_period;-- normal incremant

in\_sel\_mux\_halt<='1';

wait for 10\*clk\_period; -- halting

in\_sel\_mux\_halt<='0'; -- resuming after halt

***Decoder signal description:***

|  |  |  |  |
| --- | --- | --- | --- |
| Sl no | Signal | Purpose | Logical description |
| 1 | Sel\_mux\_oprnd2 | Selecting 2nd operand to ALU | “1” when immediate data “0” when register “Rt” provides 2nd operand |
| 2 | Sel\_mux\_neg | Selecting between “sub”-“add,’ “or”-“nor” operations | “1” when operation is “SUB” or “NOR” “0” when “ADD” or “OR”, else don’t care |
| 3 | Sel\_mux\_andor | Selecting between “and” –“or/nor” operations | “1” when operation is “OR/NOR” “0” when its “AND” else don’t care |
| 4 | Sel\_mux\_wb[0:1] | Selecting which operations output as o/p data of ALU | “11” – when “ADD/SUB” “10” when “AND/OR/NOR”  “01” – when SHR “00” when SHL, else don’t care |
| 5 | Sel\_mux\_brnch[0:1] | Selecting conditional branch flag | “11” operation is BLT “10” when BEQ “01” when BNE, else “00” for other operations |
| 6 | in\_sel\_mux\_jmp | On jump | When jump instruction in\_sel\_mux\_jmp = '1' |
| 7 | in\_sel\_mux\_halt | On halt | When HALT instruction in\_sel\_mux\_halt = '1' |
|  |  |  |  |