MT3

Matt Gambill

ECE3561

11/8/2019

**addSub**

-- File: alu.vhdl

-- Engineer: Matt Gambill

-- Date: 11/8/2019

-- Synopsis: File Defines a 16 bit add/subtractor with overflow detection

-- operation =0 is addition, operation=1 is subtraction

-- if the unit overflows the operation is invalid

LIBRARY ieee;

USE ieee.std\_logic\_1164.ALL;

USE ieee.std\_logic\_arith.ALL;

ENTITY addSub IS

PORT

(

operation : IN std\_logic;

A, B : IN std\_logic\_vector(15 DOWNTO 0);

sum : OUT std\_logic\_vector(15 DOWNTO 0);

overflow,Cout : OUT std\_logic);

END addSub;

ARCHITECTURE rtl OF addSub IS

SIGNAL Cvec : std\_logic\_vector(16 DOWNTO 0);

SIGNAL Aop : std\_logic\_vector(15 DOWNTO 0);

BEGIN

Cvec(0) <= operation; -- operation could also be named Cin

Aop <= NOT A WHEN operation = '1' ELSE A; -- Take 2's Complement if subtraction operation is asserted

sum <= Aop XOR B XOR Cvec(15 DOWNTO 0);

Cvec(16 DOWNTO 1) <= (Aop AND B) OR (Aop AND Cvec(15 DOWNTO 0)) OR (B AND Cvec(15 DOWNTO 0));

overflow <= Cvec(16) XOR Cvec(15);

Cout <= Cvec(16);

END rtl;

**Test Bench**

-- File: tbalu.vhdl

-- Engineer: Matt Gambill

-- Date: 11/8/2019

-- Synopsis: File Defines a 16 bit add/subtractor with overflow detection

-- operation =0 is addition, operation=1 is subtraction

-- if the unit overflows the operation is invalid

LIBRARY ieee;

USE ieee.std\_logic\_1164.ALL;

USE ieee.std\_logic\_arith.ALL;

entity tbaddSub is

end entity;

ARCHITECTURE one of tbaddSub is

SIGNAL A,B,sum,ans,Aop : std\_logic\_vector(15 DOWNTO 0);

SIGNAL operation, overflow, check,Cout : std\_logic := '0';

SIGNAL Cvec : std\_logic\_vector(16 DOWNTO 0);

BEGIN

dut : entity work.addSub

port map(operation => operation,A=>A,B=>B,Cvec=>Cvec,sum=>sum,Aop=>Aop,overflow=>overflow,Cout=>Cout);

simulation : process

begin

operation <='0';

wait for 100 ns;

A <= "0000000000001010";

B <= "0000000000001011";

ans <= "0000000000010101";

if (sum = ans) then

check <= '1';

else

check <='0';

end if;

wait for 100 ns;

operation <='0';

A <= "0000000000011011";

B <= "0000000000001011";

ans <= "0000000000100110";

if (sum = ans) then

check <= '1';

else

check <='0';

end if;

wait for 100 ns;

operation <='0';

A <= "1000000000001010";

B <= "1000000000001011";

ans <= "0000000000000001";

if (ans(0) = overflow) then

check <= '1';

else

check <='0';

end if;

-- Subtraction test

wait for 100 ns;

operation <='1';

A <= "0000000000001010";

B <= "0000000000001111";

ans <= "0000000000000101";

if (sum = ans) then

check <= '1';

else

check <='0';

end if;

wait for 100 ns;

operation <='1';

B <= "0000000000011011";

A <= "0000000000001011";

ans <= "0000000000010000";

if (sum = ans) then

check <= '1';

else

check <='0';

end if;

wait for 100 ns;

operation <='1';

B <= "0000000000001011";

A <= "0000000000000000";

ans <= "0000000000000001";

if (ans(0) = overflow) then

check <= '1';

else

check <='0';

end if;

wait for 200 ns;

wait;

end process;

end ARCHITECTURE one;



