library IEEE;

use IEEE.STD\_LOGIC\_1164.ALL;

use IEEE.STD\_LOGIC\_ARITH.ALL;

USE IEEE.STD\_LOGIC\_UNSIGNED.ALL;

entity alu is

Port ( a,b : in STD\_LOGIC\_VECTOR (3 downto 0);

sel : in STD\_LOGIC\_VECTOR (2 downto 0);

y : out STD\_LOGIC\_VECTOR (3 downto 0));

end alu;

architecture Behavioral of alu is

begin

process(a,b,sel)

begin

case sel is

when "000" => y <= a + b;

when "001" => y <= a - b;

when "010" => y <= a - 1;

when "011" => y <= a + 1;

when "100" => y <= a and b;

when "101" => y <= a or b;

when "110" => y <= not a;

when "111" => y <= a xor b;

when others => Null;

end case;

end process;

end Behavioral;

IBRARY ieee;

USE ieee.std\_logic\_1164.ALL;

ENTITY alu\_tech IS

END alu\_tech;

ARCHITECTURE behavior OF alu\_tech IS

COMPONENT alu

PORT(

a : IN std\_logic\_vector(3 downto 0);

b : IN std\_logic\_vector(3 downto 0);

sel : IN std\_logic\_vector(2 downto 0);

y : OUT std\_logic\_vector(3 downto 0)

);

END COMPONENT

--Inputs

signal a : std\_logic\_vector(3 downto 0) := (others => '0');

signal b : std\_logic\_vector(3 downto 0) := (others => '0');

signal sel : std\_logic\_vector(2 downto 0) := (others => '0');

--Outputs

signal y : std\_logic\_vector(3 downto 0);

BEGIN

uut: alu PORT MAP (

a => a,

b => b,

sel => sel,

y => y

);

stim\_proc: process

begin

a <= "1001";

b <= "0100";

sel <= "000"; wait for 100 ns;

sel <= "001"; wait for 100 ns;

sel <= "010"; wait for 100 ns;

sel <= "011"; wait for 100 ns;

sel <= "100"; wait for 100 ns;

sel <= "101"; wait for 100 ns;

sel <= "110"; wait for 100 ns;

sel <= "111"; wait for 100 ns;

wait;

end process;

END;

