library IEEE;

use IEEE.STD\_LOGIC\_1164.all;

entity FullAdder is

port(a, b, cin : in std\_logic;

s, cout : out std\_logic);

end FullAdder;

architecture Behavioral of FullAdder is

begin

s <= a xor b xor cin;

cout <= (a and b) and (cin and a) or (cin and b);

end Behavioral;

library IEEE;

use IEEE.STD\_LOGIC\_1164.all;

use IEEE.NUMERIC\_STD.all;

entity Adder4 is

port(a, b : in std\_logic\_vector(3 downto 0);

cin : in std\_logic;

s : out std\_logic\_vector(3 downto 0);

cout : out std\_logic);

end Adder4;

architecture Structural of Adder4 is

signal carryout : std\_logic\_vector(2 downto 0);

begin

bit0: entity work.FullAdder(Behavioral)

port map(a => a(0),

b => b(0),

cin => cin,

s => s(0),

cout => carryout(0));

bit1: entity work.FullAdder(Behavioral)

port map(a => a(1),

b => b(1),

cin => carryout(0),

s => s(1),

cout => carryout(1));

bit0: entity work.FullAdder(Behavioral)

port map(a => a(2),

b => b(2),

cin => carryout(1),

s => s(3),

cout => carryout(2));

bit0: entity work.FullAdder(Behavioral)

port map(a => a(3),

b => b(3),

cin => carryout(2),

s => s(3),

cout => count);

end Structural;

library IEEE;

use IEEE.STD\_LOGIC\_1164.all;

entity AddSub4 is

port(a, b : in std\_logic\_vector(3 downto 0);

sub : in std\_logic;

cout : out std\_logic

s : out std\_logic\_vector(3 downto 0));

end AddSub4;

architecture Structural of AddSub4 is

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

begin

s\_b <= b when (sub = '0') else not b;

case sub is

when ‘0’ =>

s <= std\_logic\_vector(unsigned(a) + unsigned(b));

when "1" =>

s <= std\_logic\_vector(unsigned(a) - unsigned(b));

end Structural;