library IEEE;

use IEEE.STD\_LOGIC\_1164.ALL; use IEEE.NUMERIC\_STD.ALL;

use IEEE.STD\_LOGIC\_UNSIGNED.ALL;

entity alu is

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

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

carryout : out STD\_LOGIC

);

end entity alu;

architecture Behavioral of alu is

signal result : std\_logic\_vector(3 downto 0); signal cryt : std\_logic\_vector (4 downto 0); begin

process(a,b,op)is begin

case op is

when "0000" => result <= a + b; when "0001" => result <= a - b; when "0010" => result <= a and b; when "0011" => result <= a or b; when "0100" => result <= a xor b; when "0101" => result <= not a;

when "0110" => result <= std\_logic\_vector(to\_unsigned((to\_integer(unsigned(a)) \* to\_integer(unsigned(b))),4));

when "0111" => result <= std\_logic\_vector(to\_unsigned((to\_integer(unsigned(a)) / to\_integer(unsigned(b))),4));

when "1000" => result <= std\_logic\_vector( a srl 1); when "1001" => result <= std\_logic\_vector( b sll 1); when "1010" =>

if a < b then result <= "0001"; else

result <= "0000"; end if;

when "1011" => if a <= b then result <= "0001"; else

result <= "0000"; end if;

when "1100" =>

if a = b then result <= "0001"; else

result <= "0000"; end if;

when "1101" => if a >= b then result <= "0001"; else

result <= "0000"; end if;

if a > b then result <= "0001"; else

result <= "0000"; end if;

when others => result <= "0000"; end case;

end process;

c <= result;

cryt <= ('0' & a) + ('0' & b); carryout <= cryt(4);

end Behavioral;