library ieee;

use ieee.std\_logic\_1164.all;

use ieee.std\_logic\_unsigned.all;

entity bcdaddsub is

port

(

switches : in std\_logic\_vector(7 downto 0);

KEY0, KEY1, SW9 : std\_logic;

HEX0, HEX1, HEX2, HEX3, HEX4, HEX5 : out std\_logic\_vector(0 to 6);

LEDR3 : out std\_logic

);

end bcdaddsub;

architecture n of bcdaddsub is

signal a, b : std\_logic\_vector(7 downto 0);

signal b\_temp0, b\_temp1, sum\_temp0, sum\_temp1 : std\_logic\_vector(4 downto 0);

signal sum\_temp0\_adj, sum\_temp1\_adj : std\_logic\_vector(4 downto 0);

signal carry, carryin, overflow : std\_logic;

signal sum0, sum1 : std\_logic\_vector(3 downto 0);

signal d1, d2, d3, d4, d5, d6 : std\_logic\_vector(6 downto 0);

begin

process (switches, KEY0, KEY1) is

begin

if(KEY1 = '0') then --addition code

a <= (switches);

case (a(7 downto 4)) is

when "0000" => d1 <= "0000001";

when "0001" => d1 <= "1001111";

when "0010" => d1 <= "0010010";

when "0011" => d1 <= "0000110";

when "0100" => d1 <= "1001100";

when "0101" => d1 <= "0100100";

when "0110" => d1 <= "0100000";

when "0111" => d1 <= "0001111";

when "1000" => d1 <= "0000000";

when "1001" => d1 <= "0000100";

when others => d1 <= "1111111";

end case;

case (a(3 downto 0)) is

when "0000" => d2 <= "0000001";

when "0001" => d2 <= "1001111";

when "0010" => d2 <= "0010010";

when "0011" => d2 <= "0000110";

when "0100" => d2 <= "1001100";

when "0101" => d2 <= "0100100";

when "0110" => d2 <= "0100000";

when "0111" => d2 <= "0001111";

when "1000" => d2 <= "0000000";

when "1001" => d2 <= "0000100";

when others => d2 <= "1111111";

end case;

elsif (KEY0 = '0') then

b <= (switches);

case (b(7 downto 4)) is

when "0000" => d3 <= "0000001";

when "0001" => d3 <= "1001111";

when "0010" => d3 <= "0010010";

when "0011" => d3 <= "0000110";

when "0100" => d3 <= "1001100";

when "0101" => d3 <= "0100100";

when "0110" => d3 <= "0100000";

when "0111" => d3 <= "0001111";

when "1000" => d3 <= "0000000";

when "1001" => d3 <= "0000100";

when others => d3 <= "1111111";

end case;

case (b(3 downto 0)) is

when "0000" => d4 <= "0000001";

when "0001" => d4 <= "1001111";

when "0010" => d4 <= "0010010";

when "0011" => d4 <= "0000110";

when "0100" => d4 <= "1001100";

when "0101" => d4 <= "0100100";

when "0110" => d4 <= "0100000";

when "0111" => d4 <= "0001111";

when "1000" => d4 <= "0000000";

when "1001" => d4 <= "0000100";

when others => d4 <= "1111111";

end case;

end if;

end process;

process (b, SW9) is

begin

if SW9 = '1' then --subtraction code

carryin <= '1';

case (b(3 downto 0)) is

when "0000" => b\_temp0 <= "01001";

when "0001" => b\_temp0 <= "01000";

when "0010" => b\_temp0 <= "00111";

when "0011" => b\_temp0 <= "00110";

when "0100" => b\_temp0 <= "00101";

when "0101" => b\_temp0 <= "00100";

when "0110" => b\_temp0 <= "00011";

when "0111" => b\_temp0 <= "00010";

when "1000" => b\_temp0 <= "00001";

when "1001" => b\_temp0 <= "00000";

when others => b\_temp0 <= "00000";

end case;

case (b(7 downto 4)) is

when "0000" => b\_temp1 <= "01001";

when "0001" => b\_temp1 <= "01000";

when "0010" => b\_temp1 <= "00111";

when "0011" => b\_temp1 <= "00110";

when "0100" => b\_temp1 <= "00101";

when "0101" => b\_temp1 <= "00100";

when "0110" => b\_temp1 <= "00011";

when "0111" => b\_temp1 <= "00010";

when "1000" => b\_temp1 <= "00001";

when "1001" => b\_temp1 <= "00000";

when others => b\_temp1 <= "00000";

end case;

else

carryin <= '0';

b\_temp0 <= ('0' & b(3 downto 0));

b\_temp1 <= ('0' & b(7 downto 4));

end if;

end process;

process (a, b\_temp0, b\_temp1, sum\_temp0\_adj, sum\_temp1\_adj)

begin

sum\_temp0 <= ('0' & a(3 downto 0)) + b\_temp0 + ("0000" & carryin);

sum\_temp0\_adj <= sum\_temp0 + "00110";

if (sum\_temp0 > 9) then

sum0 <= sum\_temp0\_adj(3 downto 0);

carry <= '1';

else

carry <= '0';

sum0 <= sum\_temp0(3 downto 0);

end if;

case sum0 is

when "0000" => d6 <= "0000001";

when "0001" => d6 <= "1001111";

when "0010" => d6 <= "0010010";

when "0011" => d6 <= "0000110";

when "0100" => d6 <= "1001100";

when "0101" => d6 <= "0100100";

when "0110" => d6 <= "0100000";

when "0111" => d6 <= "0001111";

when "1000" => d6 <= "0000000";

when "1001" => d6 <= "0000100";

when others => d6 <= "1111111";

end case;

sum\_temp1 <= ('0' & a(7 downto 4)) + b\_temp1 + ("0000" & carry);

sum\_temp1\_adj <= sum\_temp1 + "00110";

if (sum\_temp1 > 9) then

overflow <= '1';

sum1 <= sum\_temp1\_adj(3 downto 0);

else

overflow <= '0';

sum1 <= sum\_temp1(3 downto 0);

end if;

case sum1 is

when "0000" => d5 <= "0000001";

when "0001" => d5 <= "1001111";

when "0010" => d5 <= "0010010";

when "0011" => d5 <= "0000110";

when "0100" => d5 <= "1001100";

when "0101" => d5 <= "0100100";

when "0110" => d5 <= "0100000";

when "0111" => d5 <= "0001111";

when "1000" => d5 <= "0000000";

when "1001" => d5 <= "0000100";

when others => d5 <= "1111111";

end case;

end process;

process(d1, d2, d3, d4, d5, d6)

begin

HEX5 <= d1;

HEX4 <= d2;

HEX3 <= d3;

HEX2 <= d4;

HEX1 <= d5;

HEX0 <= d6;

LEDR3 <= overflow;

end process;

end n;