```
LIBRARY ieee;
USE ieee.std_logic_1164.all;
USE ieee.std_logic_unsigned.all ;

entity soru1 is
      generic (bitsayisi:INTEGER:=4);
      port
      (
            A,B:IN STD_LOGIC_VECTOR(bitsayisi-1 downto 0);
            secme:BUFFER STD_LOGIC_VECTOR(2 downto 0);
            Clock:IN STD_LOGIC;
            F:OUT STD_LOGIC_VECTOR(bitsayisi-1 downto 0)
      );
end soru1;

ARCHITECTURE Behavior OF soru1 IS
BEGIN
      PROCESS(Clock)
      variable temp:STD_LOGIC_VECTOR(bitsayisi downto 0);
      BEGIN
            if (Clock'EVENT and Clock='1') then

                  temp:="00000";
                  case secme is
                        when "000" =>
                              F<="0000";
                        when "001" =>
                              temp:=B-A;
                              F<=temp;
                        when "010" =>
                              temp:=A-B;
                              F<=temp;
                        when "011" =>
                              temp:=A+B;
                              F<=temp;
                        when "100" =>
                              F<=A xor B;
                        when "101" =>
                              F<=A or B;
                        when "110" =>
                              F<=A and B;
                        when OTHERS =>
                              F<="1111";
                  end case;

            if secme<"111" then
                  secme<=secme+1;
            end if;

            end if;

      end process;

end behavior;
```

```
LIBRARY ieee;
USE ieee. std_logic_1164.all;
USE ieee.std_logic_unsigned.all ;

entity soru2 is
      port
      (
            w:BUFFER STD_LOGIC_VECTOR(3 downto 0);
            Clock:IN STD_LOGIC;
            abcdefg:OUT STD_LOGIC_VECTOR(6 downto 0)
      );
end soru2;

ARCHITECTURE Behavior OF soru2 IS
BEGIN
      PROCESS(Clock)
      BEGIN
            if (Clock'EVENT and Clock='1') then
                  case w is
                        when "0000" =>
                              abcdefg<="1111110";
                        when "0001" =>
                              abcdefg<="0110000";
                        when "0010" =>
                              abcdefg<="1101101";
                        when "0011" =>
                              abcdefg<="1111001";
                        when "0100" =>
                              abcdefg<="0110011";
                        when "0101" =>
                              abcdefg<="1011011";
                        when "0110" =>
                              abcdefg<="1011111";
                        when "0111" =>
                              abcdefg<="1110000";
                        when "1000" =>
                              abcdefg<="1111111";
                        when "1001" =>
                              abcdefg<="1111011";
                        when others =>
                              abcdefg<="0000000";
                  end case;

                  if w<"1010" then
                        w<=w+1;
                  end if;
            end if;
      end process;
end behavior;
```

```
LIBRARY ieee;
USE ieee. std_logic_1164.all;

entity D_ff is
      port
      (
            D: IN STD_LOGIC;
            Clock:IN STD_LOGIC;
            Enable:IN STD_LOGIC;
            Clear:IN STD_LOGIC;
            Q: OUT STD_LOGIC
      );
end D_ff;

architecture behv of D_ff is
begin
      process(D,Clock,Enable,Clear)
      begin
            if(Clock='1' and Clock'event) then
                  Q<=D;
            end if;
      end process;
end behv;
-------------------------------------------------------
entity reg is
      port
      (
            D0: IN STD_LOGIC;
            D1: IN STD_LOGIC;
            Clk: IN STD_LOGIC;
            Rst: IN STD_LOGIC;
            Load: IN STD_LOGIC;
            Q0: OUT STD_LOGIC;
            Q1: OUT STD_LOGIC
      );
end reg;

architecture behv of reg is
begin
      blok1:D_ff port map (D0,Clk,Load,Rst,Q0);
      blok2:D_ff port map (D1,Clk,Load,Rst,Q1);
end behv;









entity soru3 is
      port
      (
            w2:IN STD_LOGIC_VECTOR(2 downto 0);
            Clock2:IN STD_LOGIC;
            E2:IN STD_LOGIC;
```

```
            y2:OUT STD_LOGIC_VECTOR(7 downto 0)
      );
end soru3;

ARCHITECTURE Behavior OF soru3 IS

component decoder is
      port
      (
            w:IN STD_LOGIC_VECTOR(1 downto 0);
            Clock:IN STD_LOGIC;
            E:IN STD_LOGIC;
            y:OUT STD_LOGIC_VECTOR(3 downto 0)
      );
end component;

signal e2_1:STD_LOGIC;
signal e2_2:STD_LOGIC;

BEGIN
      PROCESS(Clock2)
      BEGIN
            e2_1<=(not(w2(2)) and E2);
            e2_2<=(w2(2) and E2);
      end process;

            decoder1:decoder port map (w2(1 downto 0),Clock2,e2_1,y2(3
downto 0));
            decoder2:decoder port map (w2(1 downto 0),Clock2,e2_2,y2(7
downto 4));
end behavior;
```

```
LIBRARY ieee;
USE ieee. std_logic_1164.all;

entity muxdevresi is
      port
      (
            s: in std_logic;
            w: in std_logic_vector(1 downto 0);
            f: out std_logic
      );
end muxdevresi;

architecture behavior of muxdevresi is
begin
process(s,w)
begin

      if s='0' then
            f<=w(0);
      else
            f<=w(1);
      end if;

end process;
end behavior;

LIBRARY ieee;
USE ieee. std_logic_1164.all;

entity soru4 is
      port
      (
            s2: in std_logic_vector(1 downto 0);
            w2: in std_logic_vector(3 downto 0);
            f2:out std_logic
      );
end soru4;

architecture behavior of soru4 is
component muxdevresi is
      port
      (
            s: in std_logic;
            w: in std_logic_vector(1 downto 0);
            f: out std_logic
      );
end component;

signal fa:std_logic_vector(1 downto 0);

begin

mux1:muxdevresi port map(s2(0),w2(1 downto 0),fa(0));
mux2:muxdevresi port map(s2(0),w2(3 downto 2),fa(1));
mux3:muxdevresi port map(s2(1),fa,f2);

end behavior;
```

```
LIBRARY ieee;
USE ieee.std_logic_1164.all;

entity soru5 is
      port
      (
            A: in std_logic_vector(3 downto 0);
            B: in std_logic_vector(3 downto 0);
            Clock: in std_logic;
            Cin:in std_logic;
            Cout:out std_logic;
            addsub:in std_logic;
            Sonuc:out std_logic_vector(3 downto 0)
      );
end soru5;

architecture behavior of soru5 is

component fulladder is
      port
      (
            x: in std_logic;
            y: in std_logic;
            z: in std_logic;
            s: out std_logic;
            c: out std_logic
      );
end component;

signal c1,c2,c3:std_logic;
signal op:std_logic_vector(3 downto 0);

begin

      op(3)<=B(3) xor addsub;
      op(2)<=B(2) xor addsub;
      op(1)<=B(1) xor addsub;
      op(0)<=B(0) xor addsub;

      fa1:fulladder port map(A(0),op(0),Cin,Sonuc(0),c1);
      fa2:fulladder port map(A(1),op(1),c1,Sonuc(1),c2);
      fa3:fulladder port map(A(2),op(2),c2,Sonuc(2),c3);
      fa4:fulladder port map(A(3),op(3),c3,Sonuc(3),Cout);

end behavior;



LIBRARY ieee;
USE ieee.std_logic_1164.all;

entity fulladder is
      port
      (
            x: in std_logic;
            y: in std_logic;
            z: in std_logic;
            s: out std_logic;
            c: out std_logic
      );
```

```
end fulladder;

architecture behavior of fulladder is
begin
      s<=x xor y xor z;
      c<=(y and z) or (x and (y xor z));
end behavior;
```

```
LIBRARY ieee;
USE ieee.std_logic_1164.all;

entity soru6 is
      port
      (
            x:in std_logic;
            Clock:in std_logic;
            z:out std_logic
      );
end soru6;

architecture behavior of soru6 is

      component flipflop IS
            PORT
            (
                  D: IN STD_LOGIC ;
                  Q: OUT STD_LOGIC ;
                  Qtumleyen: out std_logic;
                  C: IN std_logic
            );
      END component;

      signal q1,q2,q1t,q2t,d1,d2:std_logic;

begin

      d1<=q2 or q1t;
      d2<=x and q2t;

      dff1:flipflop port map(d1,q1,q1t,Clock);
      dff2:flipflop port map(d2,q2,q2t,Clock);

      z<=q2t or q1 ;
end behavior;


LIBRARY ieee;
USE ieee.std_logic_1164.all;

ENTITY flipflop IS
      PORT
      (
            D: IN STD_LOGIC ;
            Q: OUT STD_LOGIC ;
            Qtumleyen: out std_logic;
            C: IN std_logic
      );
END flipflop ;

ARCHITECTURE Behavior OF flipflop IS
BEGIN
      PROCESS(C)
      BEGIN
            IF C'EVENT AND C = '1' THEN
                  Q <= D ;
                  Qtumleyen<=not(D);
            END IF ;
      END PROCESS ;
```

```
END Behavior ;
```

```
LIBRARY ieee;
USE ieee.std_logic_1164.all;

entity soru7 is
      port
      (
            Clock:in std_logic;
            Q:out std_logic_vector(3 downto 0)
      );
end soru7;

architecture behavior of soru7 is
      component tflipflop IS
      PORT
      (
            T: IN STD_LOGIC ;
            Q: OUT STD_LOGIC ;
            QT: OUT STD_LOGIC ;
            C: IN std_logic
      );
      END component ;

      signal t1,t2,t3,t4,q1,q2,q3,q4,q1t,q2t,q3t,q4t:std_logic;
begin
      t1<='1';
      t2<=q1;
      t3<=q2 and q1;
      t4<=q3 and q2 and q1;

      tff1:tflipflop port map(t1,q1,q1t,Clock);
      tff2:tflipflop port map(t2,q2,q2t,Clock);
      tff3:tflipflop port map(t3,q3,q3t,Clock);
      tff4:tflipflop port map(t4,q4,q4t,Clock);

      Q(3)<=q4;
      Q(2)<=q3;
      Q(1)<=q2;
      Q(0)<=q1;

end behavior;



LIBRARY ieee;
USE ieee.std_logic_1164.all;

ENTITY tflipflop IS
      PORT
      (
            T: IN STD_LOGIC ;
            Q: OUT STD_LOGIC ;
            QT: OUT STD_LOGIC ;
            C: IN std_logic
      );
END tflipflop ;

ARCHITECTURE Behavior OF tflipflop IS
BEGIN
      PROCESS(C)
            variable deger:std_logic;
```

```
      BEGIN
            IF C'EVENT AND C = '1' THEN
                  if T='1' then
                        deger:=not(deger);
                  end if;
            END IF ;
            Q<=deger;
            QT<=not(deger);
      END PROCESS ;
END Behavior ;
```

```
LIBRARY ieee;
USE ieee.std_logic_1164.all;

entity soru8 is
      port
      (
            Clock:in std_logic;
            enable:in std_logic;
            cikis:buffer std_logic_vector(2 downto 0)
      );
end soru8;

architecture behavior of soru8 is
      component sayici is
            port
            (
                  clock: in std_logic;
                  d: in std_logic_vector(2 downto 0);
                  load: in std_logic;
                  enable:in std_logic;
                  q: out std_logic_vector(2 downto 0)
            );
      end component;

      signal sifir:std_logic_vector(2 downto 0);
      signal sonuc:std_logic_vector(2 downto 0);
      signal load:std_logic;

begin
      sifir<="000";
      load<=sonuc(2) and sonuc(0);

      sayan:sayici port map(Clock,sifir,load,enable,sonuc);
      cikis<=sonuc;
end behavior;


LIBRARY ieee;
USE ieee.std_logic_1164.all;
USE ieee.std_logic_unsigned.all ;

entity sayici is
      port
      (
            clock: in std_logic;
            d: in std_logic_vector(2 downto 0);
            load: in std_logic;
            enable:in std_logic;
            q: out std_logic_vector(2 downto 0)
      );
end sayici;

architecture behavior of sayici is
begin

      process(clock)
            variable temp:std_logic_vector(2 downto 0);
      begin
            if clock'event and clock='1' then
                  if load='1' then
```

```
                        temp:=d;
                  else
                        if enable='1' then
                              if temp="111" then
                                    temp:="000";
                              else
                                    temp:=temp+1;
                              end if;
                        end if;
                  end if;
                  q<=temp;
            end if;

      end process;
end behavior;
```

```
LIBRARY ieee;
USE ieee. std_logic_1164.all;
USE ieee.std_logic_unsigned.all ;

entity soru9 is
      generic(bitsayisi:INTEGER:=8);
      port
      (
            Clock:IN std_logic;
            soldan:in std_logic;
            sagdan:in std_logic;
            s:in std_logic_vector(1 downto 0);
            q:in std_logic_vector(bitsayisi-1 downto 0);
            o:inout std_logic_vector(bitsayisi-1 downto 0)
      );
end soru9;

architecture behavior of soru9 is
begin

process(Clock)
variable i:INTEGER;
variable temp:std_logic_vector(bitsayisi-1 downto 0);
begin

if Clock'EVENT and Clock='1' then
      if s="00" then
            o<=o;
      elsif s="01" then
            for i in 1 to bitsayisi-1 loop
                  temp(i-1):=temp(i);
            end loop;
            temp(bitsayisi-1):=soldan;
            o<=temp;

      elsif s="10" then
            for i in bitsayisi-2 downto 0 loop
                  temp(i+1):=temp(i);
            end loop;
            temp(0):=sagdan;
            o<=temp;

      elsif s="11" then
            o<=q;

      end if;
end if;

end process;
end behavior;
```

```
LIBRARY ieee;
USE ieee.std_logic_1164.all;

entity soru10 is
      port
      (
            clock:in std_logic;
            sil:in std_logic;
            byegelen:in std_logic;
            sayia:in std_logic_vector(7 downto 0);
            sayib:in std_logic_vector(7 downto 0);
            cikis:out std_logic
      );
end soru10;

architecture behavior of soru10 is

component shifter is
      port
      (
            giris:in std_logic_vector(7 downto 0);
            yukle:in std_logic;
            si: in std_logic;
            so: out std_logic;
            shiftr: in std_logic;
            clk: in std_logic
      );
end component;

component flipflop IS
      PORT
      (
            D: IN STD_LOGIC ;
            Q: OUT STD_LOGIC ;
            C: IN std_logic;
            sil: in std_logic
      );
END component ;

component fulladder is
      port
      (
            x: in std_logic;
            y: in std_logic;
            z: in std_logic;
            s: out std_logic;
            c: out std_logic
      );
end component;

signal fa_s,fa_x,fa_y,fa_z,fa_c:std_logic;
signal ayukleizin,byukleizin,otele:std_logic;

type durum is (T1,T2,T3);
signal Y:durum;

begin
      process(clock)
            variable sayac:integer:=0;
      begin
```

```
            if clock'event and clock='1' then
                  Y<=T1;

                  case Y is
                        when T1=>
                              ayukleizin<='1';
                              byukleizin<='1';
                              otele<='0';
                              Y<=T2;
                        when T2=>
                              ayukleizin<='0';
                              byukleizin<='0';
                              otele<='1';
                              sayac:=sayac+1;

                              if sayac=10 then
                                    cikis<='1';
                                    Y<=T3;
                              else
                                    Y<=T2;
                              end if;
                        when T3=>
                              otele<='0';
                  end case;
            end if;
      end process;

      regA:shifter port map(sayia,ayukleizin,fa_s,fa_x,otele,clock);
      regB:shifter port map(sayib,byukleizin,byegelen,fa_y,otele,clock);
      toplayici:fulladder port map(fa_x,fa_y,fa_z,fa_s,fa_c);
      dflipflop:flipflop port map(fa_c,fa_z,clock,sil);

end behavior;


LIBRARY ieee;
USE ieee.std_logic_1164.all;

entity shifter is
      port
      (
            giris:in std_logic_vector(7 downto 0);
            yukle:in std_logic;
            si: in std_logic;
            so: out std_logic;
            shiftr: in std_logic;
            clk: in std_logic
      );
end shifter;

architecture behavior of shifter is
begin
      process(clk)
            variable temp:std_logic_vector(7 downto 0);
            variable i:integer;
      begin

            if clk'event and clk='1' then
                  if shiftr='1' then
                        so<=temp(0);
```

```
                        for i in 1 to 7 loop
                              temp(i-1):=temp(i);
                        end loop;
                        temp(7):=si;
                  elsif yukle='1' then
                        temp:=giris;
                  end if;
            end if;

      end process;
end behavior;

LIBRARY ieee;
USE ieee.std_logic_1164.all;

ENTITY flipflop IS
      PORT
      (
            D: IN STD_LOGIC ;
            Q: OUT STD_LOGIC ;
            C: IN std_logic;
            sil: in std_logic
      );
END flipflop ;

ARCHITECTURE Behavior OF flipflop IS
BEGIN
      PROCESS(sil,C)
      BEGIN
            IF sil = '0' THEN
                  Q <= '0' ;
            ELSIF C'EVENT AND C = '1' THEN
                  Q <= D ;
            END IF ;
      END PROCESS ;
END Behavior ;

LIBRARY ieee;
USE ieee.std_logic_1164.all;

entity fulladder is
      port
      (
            x: in std_logic;
            y: in std_logic;
            z: in std_logic;
            s: out std_logic;
            c: out std_logic
      );
end fulladder;

architecture behavior of fulladder is
begin
      s<=x xor y xor z;
      c<=(y and z) or (x and (y xor z));
end behavior;
```

```
LIBRARY ieee;
USE ieee. std_logic_1164.all;

entity mydecoder is
port(w0,w1,en: in std_logic;
           y:out std_logic_vector(3 downto 0)
           );
           end mydecoder;

architecture behaviour of mydecoder is
begin
process(w0,w1,en)
begin

if en='1' then
if w1='0' and w0='0' then
y<="0001";
elsif w1='0' and w0='1' then
y<="0010";
elsif w1='1' and w0='0' then
y<="0100";
elsif w1='1' and w0='1' then
y<="1000";
end if;
end if;
end process;
end behaviour;

LIBRARY ieee;
USE ieee. std_logic_1164.all;

entity bigdecoder is
port(w0,w1,w2,en: in std_logic;
           y:out std_logic_vector(7 downto 0)
           );
           end bigdecoder;

architecture structural of bigdecoder is
component mydecoder is
port(w0,w1,en: in std_logic;
           y:out std_logic_vector(3 downto 0)
           );
           end component;

           signal s1,s2:std_logic;
begin
s1<=not(w2) and en;
s2<=w2 and en;
dec1: mydecoder port map(w0,w1,s1,y(3 downto 0));
dec2: mydecoder port map(w0,w1,s2,y(7 downto 4));

end structural;
```