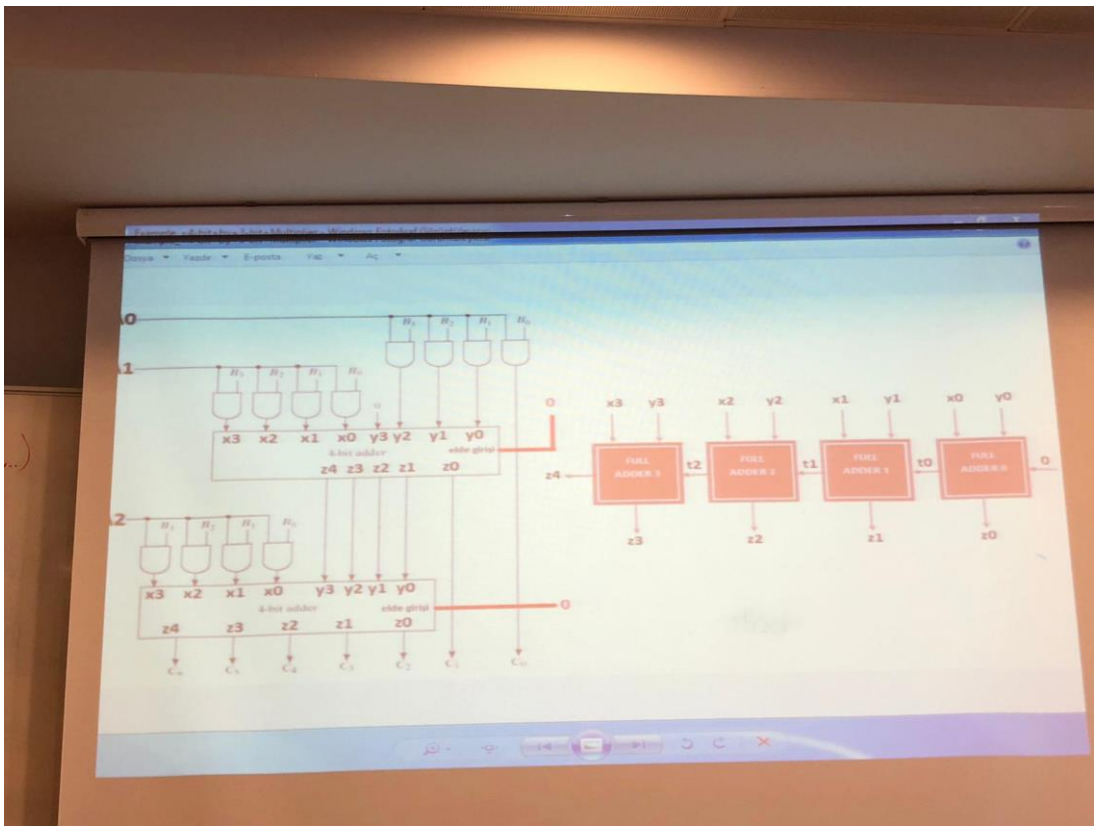


VHDL KODU

```
----------------------------------------------------------------------------------
-- Create Date:
-- Module Name:

----------------------------------------------------------------------------------

library IEEE;

use IEEE.STD_LOGIC_1164.ALL;

entity VE_KAPISI is

   port(   ve_g1: in STD_LOGIC;

           ve_g2: in STD_LOGIC;

           ve_cikis: out STD_LOGIC);

end VE_KAPISI;

architecture veri_akisi of VE_KAPISI is

begin

   process(ve_g1,ve_g2)

   begin

      ve_cikis <= ve_g1 and ve_g2;
```

```
   end process;
end veri_akisi;
----------------------------------------------------
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
entity VEYA_KAPISI is
   port(   veya_g1: in STD_LOGIC;
           veya_g2: in STD_LOGIC;
           veya_cikis: out STD_LOGIC);
end VEYA_KAPISI;
architecture veri_akisi of VEYA_KAPISI is
begin
   process(veya_g1, veya_g2)
   begin
      veya_cikis <= veya_g1 or veya_g2;
   end process;
end veri_akisi;
----------------------------------------------------
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
entity XOR_KAPISI is
   port(   xor_g1: in STD_LOGIC;
           xor_g2: in STD_LOGIC;
           xor_cikis: out STD_LOGIC );
end XOR_KAPISI;
architecture veri_akisi of XOR_KAPISI is
begin
   process(xor_g1, xor_g2)
   begin
      xor_cikis <= xor_g1 xor xor_g2;
   end process;
end veri_akisi;
```

```
----------------------------------------------------
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
entity YT is
   port(   yt_g1: in STD_LOGIC;
           yt_g2: in STD_LOGIC;
           yt_toplam: out STD_LOGIC;
           yt_eldecikisi: out STD_LOGIC );
end YT;
architecture yapisal of YT is
   component VE_KAPISI is
      port(   ve_g1: in STD_LOGIC;
              ve_g2: in STD_LOGIC;
          ve_cikis: out STD_LOGIC );
   end component;
   component XOR_KAPISI is
      port(   xor_g1: in STD_LOGIC;
              xor_g2: in STD_LOGIC;
              xor_cikis: out STD_LOGIC );
   end component;
begin
   blok1: XOR_KAPISI port map(xor_g1 => yt_g1 , xor_g2 => yt_g2 , xor_cikis => yt_toplam);
   blok2: VE_KAPISI port map(ve_g1 => yt_g1 , ve_g2 => yt_g2 , ve_cikis => yt_eldecikisi);
end yapisal;
--------------------------------------------------------------------------------
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
entity TT is
      port(tt_g1: in STD_LOGIC;
           tt_g2: in STD_LOGIC;
           tt_eldegirisi: in STD_LOGIC;
```

```
            tt_toplam: out STD_LOGIC;
            tt_eldecikisi: out STD_LOGIC );
end TT;
architecture yapisal of TT is
   component YT is
      port(   yt_g1: in STD_LOGIC;
              yt_g2: in STD_LOGIC;
              yt_toplam: out STD_LOGIC;
              yt_eldecikisi: out STD_LOGIC );
   end component;
   component VEYA_KAPISI is
      port(   veya_g1: in STD_LOGIC;
           veya_g2: in STD_LOGIC;
           veya_cikis: out STD_LOGIC);
   end component;
   signal arakablo1: STD_LOGIC;
   signal arakablo2: STD_LOGIC;
   signal arakablo3: STD_LOGIC;
begin
   blok1: YT port map(yt_g1 => tt_g1 , yt_g2 => tt_g2 , yt_eldecikisi => arakablo2 ,
   yt_toplam => arakablo1);
   blok2: YT port map(yt_g1 => arakablo1 , yt_g2 => tt_eldegirisi ,
   yt_eldecikisi => arakablo3 , yt_toplam => tt_toplam);
   blok3: VEYA_KAPISI port map(veya_g1 => arakablo2 , veya_g2 => arakablo3 ,
   veya_cikis => tt_eldecikisi);
end yapisal;
--------------------------------------------------------------------------------
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
entity TOPLAYICI is
   port(   d_a0: in STD_LOGIC;
           d_a1: in STD_LOGIC;
```

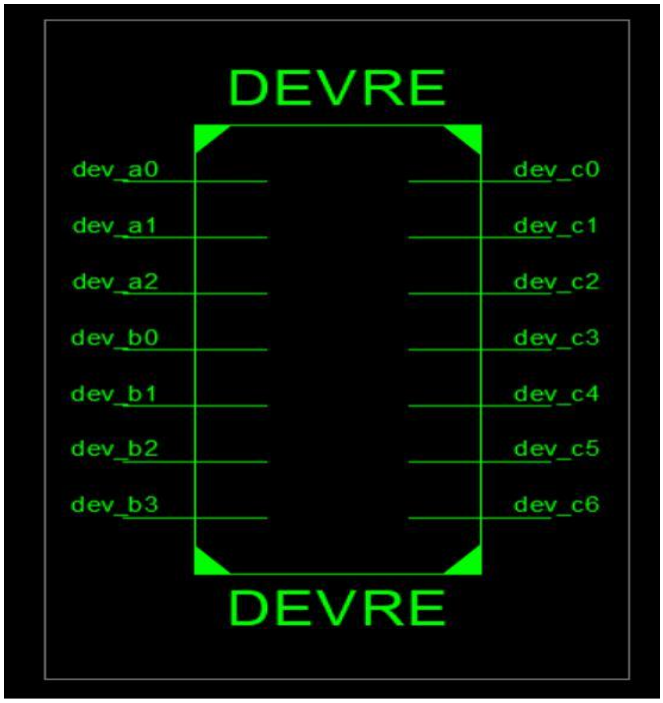
```
            d_a2: in STD_LOGIC;
            d_a3: in STD_LOGIC;
            d_b0: in STD_LOGIC;
            d_b1: in STD_LOGIC;
            d_b2: in STD_LOGIC;
            d_b3: in STD_LOGIC;
            d_s0: out STD_LOGIC;
            d_s1: out STD_LOGIC;
            d_s2: out STD_LOGIC;
            d_s3: out STD_LOGIC;
            d_s4: out STD_LOGIC;
            x: in STD_LOGIC);
        end TOPLAYICI;
architecture yapisal of TOPLAYICI is
  component TT is
    port(  tt_g1: in STD_LOGIC;
           tt_g2: in STD_LOGIC;
           tt_eldegirisi: in STD_LOGIC;
           tt_toplam: out STD_LOGIC;
           tt_eldecikisi: out STD_LOGIC   );
  end component;
  component XOR_KAPISI is
    port(xor_g1: in STD_LOGIC;
         xor_g2: in STD_LOGIC;
         xor_cikis: out STD_LOGIC);
  end component;
  signal arakablo1: STD_LOGIC;
  signal arakablo2: STD_LOGIC;
  signal arakablo3: STD_LOGIC;
  signal arakablo4: STD_LOGIC;
  signal arakablo5: STD_LOGIC;
```

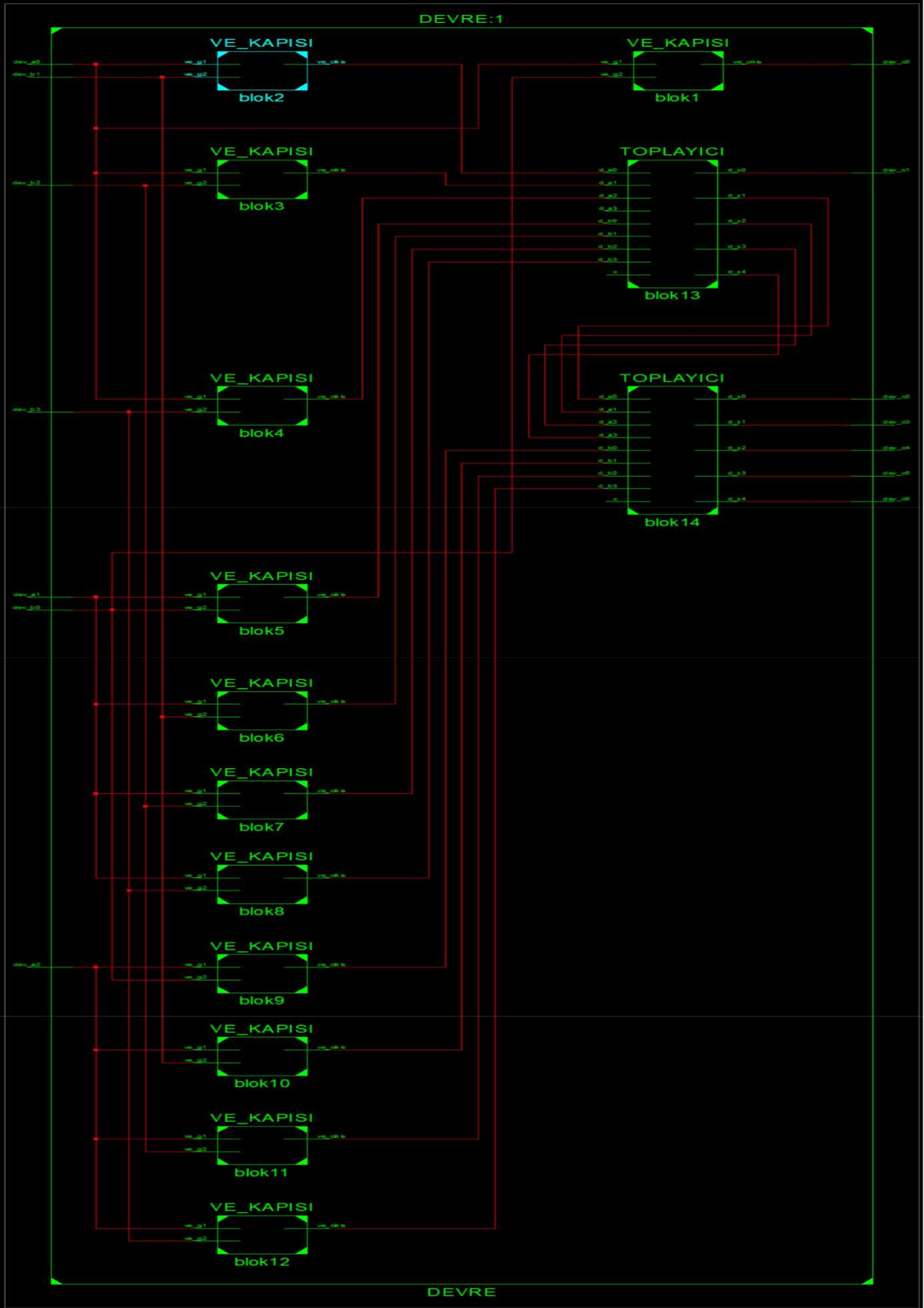
```
    signal arakablo6: STD_LOGIC;
    signal arakablo7: STD_LOGIC;


begin
    blok1: XOR_KAPISI port map( xor_g1 => d_b0 , xor_g2 => x,
    xor_cikis => arakablo1 );
    blok2: XOR_KAPISI port map( xor_g1 => d_b1 , xor_g2 => x,
    xor_cikis => arakablo2 );
    blok3: XOR_KAPISI port map( xor_g1 => d_b2 , xor_g2 => x ,
    xor_cikis => arakablo3 );
    blok4: XOR_KAPISI port map( xor_g1 => d_b3 , xor_g2 => x ,
    xor_cikis => arakablo4 );
    blok5: TT port map(tt_g1 => d_a0, tt_g2 => arakablo1, tt_eldecikisi => arakablo5, tt_eldegirisi => x,
tt_toplam => d_s0);
    blok6: TT port map(tt_g1 => d_a1, tt_g2 => arakablo2, tt_eldecikisi => arakablo6 ,
tt_eldegirisi => arakablo5 , tt_toplam => d_s1);
    blok7: TT port map(tt_g1 => d_a2, tt_g2 => arakablo3, tt_eldecikisi => arakablo7 ,
tt_eldegirisi => arakablo6, tt_toplam => d_s2);
    blok8: TT port map(tt_g1 => d_a3, tt_g2 => arakablo4, tt_eldecikisi => d_s4 ,
tt_eldegirisi => arakablo7, tt_toplam => d_s3);
end yapisal;
--------------------------------------------------------------------------------
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
entity DEVRE is
    port( dev_a0: in STD_LOGIC;
          dev_a1: in STD_LOGIC;
          dev_a2: in STD_LOGIC;
          dev_b0: in STD_LOGIC;
          dev_b1: in STD_LOGIC;
          dev_b2: in STD_LOGIC;
          dev_b3: in STD_LOGIC;
          dev_c0: out STD_LOGIC;
```

```
        dev_c1: out STD_LOGIC;
        dev_c2: out STD_LOGIC;
        dev_c3: out STD_LOGIC;
        dev_c4: out STD_LOGIC;
        dev_c5: out STD_LOGIC;
        dev_c6: out STD_LOGIC);
end DEVRE;
architecture yapisal of DEVRE is
   component TOPLAYICI is
        port(d_a0: in STD_LOGIC;
             d_a1: in STD_LOGIC;
             d_a2: in STD_LOGIC;
          d_a3: in STD_LOGIC;
          d_b0: in STD_LOGIC;
          d_b1: in STD_LOGIC;
          d_b2: in STD_LOGIC;
          d_b3: in STD_LOGIC;
          d_s0: out STD_LOGIC;
          d_s1: out STD_LOGIC;
          d_s2: out STD_LOGIC;
          d_s3: out STD_LOGIC;
          d_s4: out STD_LOGIC;
          x: in STD_LOGIC);
   end component;
    component VE_KAPISI is
      port(   ve_g1: in STD_LOGIC;
              ve_g2: in STD_LOGIC;
              ve_cikis: out STD_LOGIC);
   end component;
   signal arakabloX0: STD_LOGIC;
   signal arakabloX1: STD_LOGIC;
   signal arakabloX2: STD_LOGIC;
```

```
    signal arakabloX3: STD_LOGIC;
    signal arakabloY0: STD_LOGIC;
    signal arakabloY1: STD_LOGIC;
    signal arakabloY2: STD_LOGIC;
    signal arakabloY3: STD_LOGIC;
    signal arakabloZ1: STD_LOGIC;
    signal arakabloZ2: STD_LOGIC;
    signal arakabloZ3: STD_LOGIC;
    signal arakabloZ4: STD_LOGIC;
    signal arakabloxx0: STD_LOGIC;
    signal arakabloxx1: STD_LOGIC;
    signal arakabloxx2: STD_LOGIC;
    signal arakabloxx3: STD_LOGIC;
begin
    blok1: VE_KAPISI port map(ve_g1 => dev_a0, ve_g2 => dev_b0, ve_cikis => dev_c0 );
    blok2: VE_KAPISI port map(ve_g1 => dev_a0, ve_g2 => dev_b1, ve_cikis => arakabloY0 );
    blok3: VE_KAPISI port map(ve_g1 => dev_a0, ve_g2 => dev_b2, ve_cikis => arakabloY1 );
    blok4: VE_KAPISI port map(ve_g1 => dev_a0, ve_g2 => dev_b3, ve_cikis => arakabloY2 );
    blok5: VE_KAPISI port map(ve_g1 => dev_a1, ve_g2 => dev_b0, ve_cikis => arakabloX0 );
    blok6: VE_KAPISI port map(ve_g1 => dev_a1, ve_g2 => dev_b1, ve_cikis => arakabloX1 );
    blok7: VE_KAPISI port map(ve_g1 => dev_a1, ve_g2 => dev_b2, ve_cikis => arakabloX2 );
    blok8: VE_KAPISI port map(ve_g1 => dev_a1, ve_g2 => dev_b3, ve_cikis => arakabloX3 );
    blok9: VE_KAPISI port map(ve_g1 => dev_a2, ve_g2 => dev_b0, ve_cikis => arakabloxx0 );
    blok10: VE_KAPISI port map(ve_g1 => dev_a2, ve_g2 => dev_b1, ve_cikis => arakabloxx1 );
    blok11: VE_KAPISI port map(ve_g1 => dev_a2, ve_g2 => dev_b2, ve_cikis => arakabloxx2 );
    blok12: VE_KAPISI port map(ve_g1 => dev_a2, ve_g2 => dev_b3, ve_cikis => arakabloxx3 );
    blok13: TOPLAYICI port map(d_a0 => arakabloY0, d_a1 => arakabloY1, d_a2 => arakabloY2, d_a3
=> '0', d_b0 => arakabloX0, d_b1 => arakabloX1, d_b2 => arakabloX2, d_b3 => arakabloX3, x => '0',
d_s0 => dev_c1, d_s1 => arakabloZ1, d_s2 => arakabloZ2, d_s3 => arakabloZ3, d_s4 => arakabloZ4 );
    blok14: TOPLAYICI port map(d_a0 => arakabloZ1, d_a1 => arakabloZ2, d_a2 => arakabloZ3, d_a3 =>
arakabloZ4, d_b0 => arakabloxx0, d_b1 => arakabloxx1, d_b2 => arakabloxx2, d_b3 => arakabloxx3, x
=> '0', d_s0 => dev_c2, d_s1 => dev_c3, d_s2 => dev_c4, d_s3 => dev_c5, d_s4 => dev_c6);
end yapisal;
```

RTL ŞEMASI


DEVRE
dev_a0
dev_a1
dev_a2
dev_b0
dev_b1
dev_b2
dev_b3
dev_c0
dev_c1
dev_c2
dev_c3
dev_c4
dev_c5
dev_c6
DEVRE


DEVRE:1
VE_KAPISI
blok2
VE_KAPISI
blok1
VE_KAPISI
blok3
TOPLAYICI
blok13
VE_KAPISI
blok4
TOPLAYICI
blok14
VE_KAPISI
blok5
VE_KAPISI
blok6
VE_KAPISI
blok7
VE_KAPISI
blok8
VE_KAPISI
blok9
VE_KAPISI
blok10
VE_KAPISI
blok11
VE_KAPISI
blok12
DEVRE

SİMÜLASYON DALGA FORMU

✓ Sırasıyla 0001x001, 0000x001, 0010x010, 0101x010 ve 0011x100 işlemleri yapılmıştır.