Elektrik-Elektronik Mühendisliği

Sayısal Lojik

Deney Raporu-6

Yakup Demiryürek

180711049

(Bahar 2023)

Amaç

3-bit toplayıcı devre tasarımı amaçlanmıştır.

Ekipmanlar

- Xilinx yüklü bilgisayar

Deney Çalışması

DÇ1

Xilinx programında **Şekil 1**'de gösterildiği gibi tam-toplayıcı devrenin Vhdl kodu yazılmıştır.

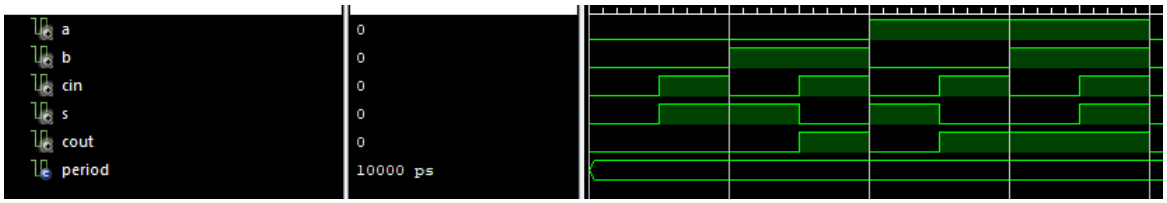
```
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
use IEEE.NUMERIC_STD.ALL;
library UNISIM;
use UNISIM.VComponents.all;

entity deney6adder is
    Port ( A : in  STD_LOGIC;
           B : in  STD_LOGIC;
           Cin : in  STD_LOGIC;
           S : out  STD_LOGIC;
           Cout : out  STD_LOGIC);
end deney6adder;

architecture Behavioral of deney6adder is
signal AxrB: std_logic;
begin
AxrB <= A xor B;
S <= AxrB xor Cin;
Cout <= (AxrB and Cin) or (A and B);
end Behavioral;
```

Şekil 1.Tam-toplayıcı Vhdl

Simülasyonun için gerekli kod **Şekil 2**'de simülasyon **Şekil 3**'de gösterilmiştir.

```
   A_process :process
   begin
      A <= '0';
      wait for period*4;
      A <= '1';
      wait for period*4;
   end process;
   B_process :process
   begin
      B <= '0';
      wait for period*2;
      B <= '1';
      wait for period*2;
   end process;
   Cin_process :process
   begin
      Cin <= '0';
      wait for period;
      Cin <= '1';
      wait for period;
   end process;

END;
```

Şekil 2.Simülasyon kodu



Şekil 3.Tam-toplayıcı simülasyon

DÇ2

Şekil 1'de yazılan Tam-toplayıcı VHDL kodu ile 3-bitlik A ($A_2A_1A_0$) ve B ($B_2B_1B_0$) sayılarını toplayıp 4-bitlik S ($S_3S_2S_1S_0$) sonucunu veren devrenin kodu **Şekil 4**'deki gibi yazılmıştır.

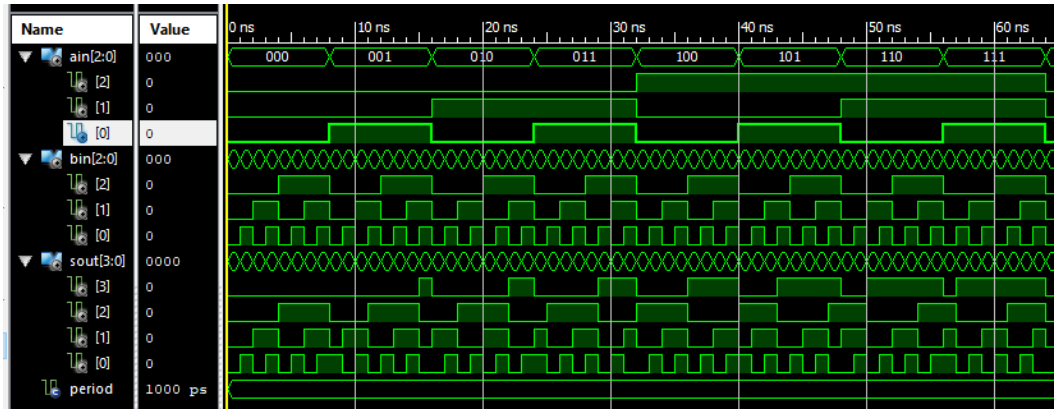
```
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
use IEEE.NUMERIC_STD.ALL;
library UNISIM;
use UNISIM.VComponents.all;

entity threebitadder is
    Port ( Ain : in  STD_LOGIC_VECTOR (2 downto 0);
           Bin : in  STD_LOGIC_VECTOR (2 downto 0);
           Sout : out  STD_LOGIC_VECTOR (3 downto 0));
end threebitadder;

architecture Behavioral of threebitadder is

component deney6adder
    Port ( A : in  STD_LOGIC;
           B : in  STD_LOGIC;
           Cin : in  STD_LOGIC;
           S : out  STD_LOGIC;
           Cout : out  STD_LOGIC);
end component;

signal c:std_logic_vector(2 downto 0);

begin

FA0: deney6adder port map(A=>Ain(0),B=>Bin(0),Cin=>'0',S=>Sout(0),Cout=>c(0));
FA1: deney6adder port map(A=>Ain(1),B=>Bin(1),Cin=>c(0),S=>Sout(1),Cout=>c(1));
FA2: deney6adder port map(A=>Ain(2),B=>Bin(2),Cin=>c(1),S=>Sout(2),Cout=>c(2));
Sout(3)<= c(2);
end Behavioral;
```

Şekil 4. 3-bit tam-toplayıcı Vhdl

Simülasyonun için gerekli kod **Şekil 5**'de simülasyon **Şekil 6**'de gösterilmiştir.

```
Ain2_process :process
begin
    Ain(2) <= '0';
    wait for period*64;
    Ain(2) <= '1';
    wait for period*64;
end process;
Ain1_process :process
begin
    Ain(1) <= '0';
    wait for period*32;
    Ain(1) <= '1';
    wait for period*32;
end process;
Ain0_process :process
begin
    Ain(0) <= '0';
    wait for period*16;
    Ain(0) <= '1';
    wait for period*16;
end process;
Bin2_process :process
begin
    Bin(2) <= '0';
    wait for period*4;
    Bin(2) <= '1';
    wait for period*4;
end process;
Bin1_process :process
begin
    Bin(1) <= '0';
    wait for period*2;
    Bin(1) <= '1';
    wait for period*2;
end process;
Bin0_process :process
begin
    Bin(0) <= '0';
    wait for period;
    Bin(0) <= '1';
    wait for period;
end process;
```

Şekil 5.Simülasyon kodu



Şekil 6. 3-bit tam-toplayıcı simülasyon

DÇ3

4-bit girişli 8-bit çıkışlı 7-parçalı göstergenin kodu **Şekil 7**'deki gibi yazılmıştır.

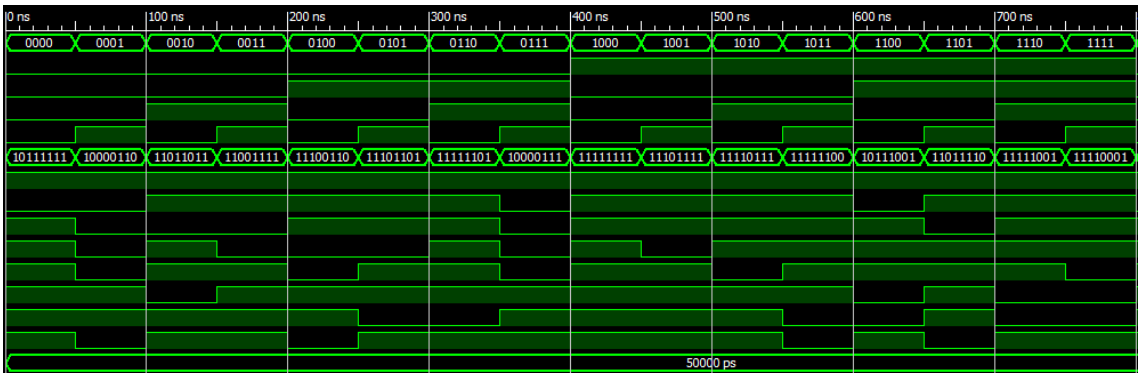
```
with A select S <=
"01000000" when "0000",
"01111001" when "0001",
"00100100" when "0010",
"00110000" when "0011",
"00011001" when "0100",
"00010010" when "0101",
"00000010" when "0110",
"01111000" when "0111",
"00000000" when "1000",
"00010000" when "1001",
"00001000" when "1010",
"00000011" when "1011",
"01000110" when "1100",
"00100001" when "1101",
"00000110" when "1110",
"00001110" when "1111",
"01111111" when others;
```

Şekil 7.Sev-seg Vhdl

Simülasyonun için gerekli kod **Şekil 8**'de simülasyon **Şekil 9**'de gösterilmiştir.

```
A3_process :process
begin
   A(3)<= '0';
   wait for period*8;
   A(3)<= '1';
   wait for period*8;
end process;
A2_process :process
begin
   A(2)<= '0';
   wait for period*4;
   A(2)<= '1';
   wait for period*4;
end process;
A1_process :process
begin
   A(1)<= '0';
   wait for period*2;
   A(1)<= '1';
   wait for period*2;
end process;
A0_process :process
begin
   A(0)<= '0';
   wait for period;
   A(0)<= '1';
   wait for period;
end process;
JD;
```

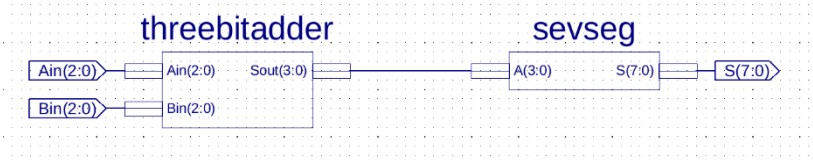
Şekil 8.Simülasyon Kodu



Şekil 9. Sev-seg simülasyon

DÇ4

DÇ2'deki 3-bit toplayıcı devresi ile **DÇ3**'deki 7-parçalı gösterge devreleri şema haline getirilip **Şekil 10**'daki gibi birleştirilmiştir.



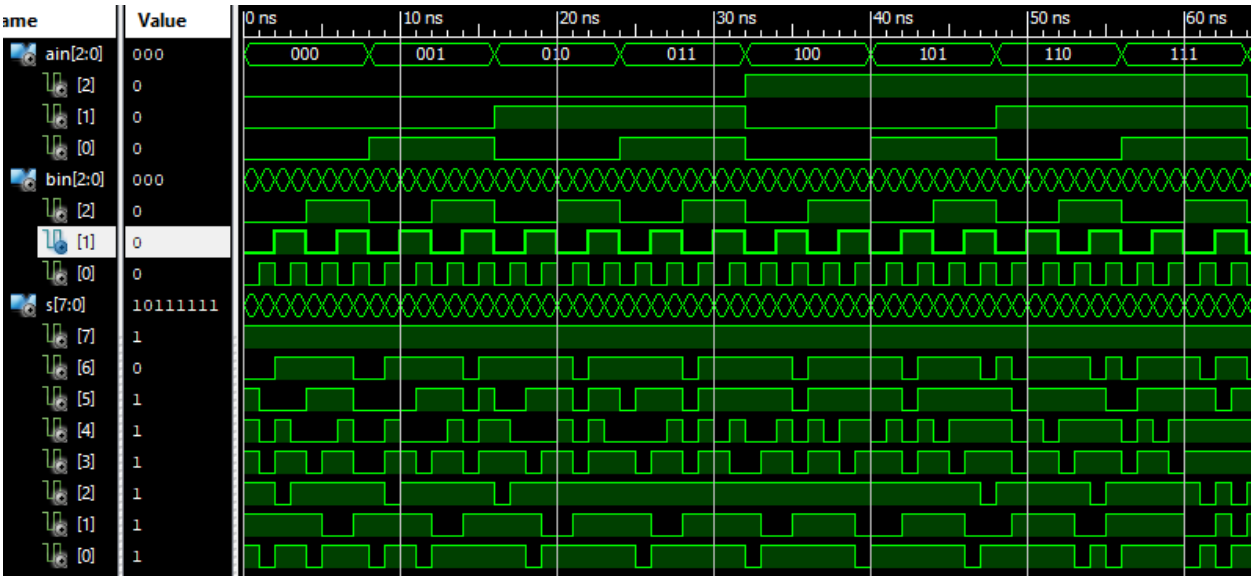
Şekil 10. 3-bit tam toplayıcı devre ile 7-parçalı gösterge devrelerinin birleşimi

DÇ5

DÇ4'teki devrenin simülasyonu için gerekli kod **Şekil 11**'de simülasyon **Şekil 12**'de gösterilmiştir.

```
Ain2process : process
begin
Ain(2)<='0';wait for period*32;
Ain(2)<='1';wait for period*32;
end process;
Ain1process : process
begin
Ain(1)<='0';wait for period*16;
Ain(1)<='1';wait for period*16;
end process;
Ain0process : process
begin
Ain(0)<='0';wait for period*8;
Ain(0)<='1';wait for period*8;
end process;
Bin2process : process
begin
Bin(2)<='0';wait for period*4;
Bin(2)<='1';wait for period*4;
end process;
Bin1process : process
begin
Bin(1)<='0';wait for period*2;
Bin(1)<='1';wait for period*2;
end process;
Bin0process : process
begin
Bin(0)<='0';wait for period;
Bin(0)<='1';wait for period;
end process;
```

Şekil 11.Simülasyon Kodu



Şekil 12. 3-bit tam toplayıcı devre ile 7-parçalı gösterge devrelerinin simülasyonu

Sonuç

S2

F çıkışı elde edilebilmesi için 4-bit tam-toplayıcı devresine ihtiyaç duyulur.

S1

A_2	A_1	A_0	B_2	B_1	B_0	S_0	S_1	S_2	S_3	S_4	S_5	S_6	S_7	
0	0	0	0	0	0	1	1	1	1	1	1	0	1	0
0	0	0	0	0	1	0	1	1	0	0	0	0	1	1
0	0	0	0	1	0	1	1	0	1	1	0	1	1	2
0	0	0	0	1	1	1	1	1	1	0	0	1	1	3
0	0	0	1	0	0	0	1	1	0	0	1	1	1	4
0	0	0	1	0	1	1	0	1	1	0	1	1	1	5
0	0	0	1	1	0	1	0	1	1	1	1	1	1	6
0	0	0	1	1	1	1	1	1	0	0	0	0	1	7
0	0	1	0	0	0	0	1	1	0	0	0	0	1	1
0	0	1	0	0	1	1	1	0	1	1	0	1	1	2
0	0	1	0	1	0	1	1	1	1	0	0	1	1	3
0	0	1	0	1	1	0	1	1	0	0	1	1	1	4
0	0	1	1	0	0	1	0	1	1	0	1	1	1	5
0	0	1	1	0	1	1	0	1	1	1	1	1	1	6
0	0	1	1	1	0	1	1	1	0	0	0	0	1	7
0	0	1	1	1	1	1	1	1	1	1	1	1	1	8
0	1	0	0	0	0	1	1	0	1	1	0	1	1	2
0	1	0	0	0	1	1	1	1	1	0	0	1	1	3
0	1	0	0	1	0	0	1	1	0	0	1	1	1	4
0	1	0	0	1	1	1	0	1	1	0	1	1	1	5
0	1	0	1	0	0	1	0	1	1	1	1	1	1	6
0	1	0	1	0	1	1	1	1	0	0	0	0	1	7
0	1	0	1	1	0	1	1	1	1	1	1	1	1	8
0	1	0	1	1	1	1	1	1	1	0	1	1	1	9
0	1	1	0	0	0	1	1	1	1	0	0	1	1	3
0	1	1	0	0	1	0	1	1	0	0	1	1	1	4
0	1	1	0	1	0	1	0	1	1	0	1	1	1	5

0	**1**	**1**	**0**	**1**	**1**	**1**	**0**	**1**	**1**	**1**	**1**	**1**	**1**	**6**
0	**1**	**1**	**1**	**0**	**0**	**1**	**1**	**1**	**0**	**0**	**0**	**0**	**1**	**7**
0	**1**	**1**	**1**	**0**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**8**
0	**1**	**1**	**1**	**1**	**0**	**1**	**1**	**1**	**1**	**0**	**1**	**1**	**1**	**9**
0	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**0**	**1**	**1**	**1**	**1**	**A**
1	**0**	**0**	**0**	**0**	**0**	**0**	**1**	**1**	**0**	**0**	**1**	**1**	**1**	**4**
1	**0**	**0**	**0**	**0**	**1**	**1**	**0**	**1**	**1**	**0**	**1**	**1**	**1**	**5**

1	**0**	**0**	**0**	**1**	**0**	**1**	**0**	**1**	**1**	**1**	**1**	**1**	**1**	**6**
1	**0**	**0**	**0**	**1**	**1**	**1**	**1**	**1**	**0**	**0**	**0**	**0**	**1**	**7**
1	**0**	**0**	**1**	**0**	**0**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**8**
1	**0**	**0**	**1**	**0**	**1**	**1**	**1**	**1**	**1**	**0**	**1**	**1**	**1**	**9**
1	**0**	**0**	**1**	**1**	**0**	**1**	**1**	**1**	**0**	**1**	**1**	**1**	**1**	**A**
1	**0**	**0**	**1**	**1**	**1**	**0**	**0**	**1**	**1**	**1**	**1**	**1**	**1**	**B**
1	**0**	**1**	**0**	**0**	**0**	**0**	**1**	**1**	**0**	**1**	**1**	**0**	**1**	**5**
1	**0**	**1**	**0**	**0**	**1**	**0**	**1**	**1**	**1**	**1**	**1**	**0**	**1**	**6**
1	**0**	**1**	**0**	**1**	**0**	**1**	**1**	**0**	**0**	**0**	**0**	**1**	**1**	**7**
1	**0**	**1**	**0**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**8**
1	**0**	**1**	**1**	**0**	**0**	**1**	**1**	**1**	**0**	**1**	**1**	**1**	**1**	**9**
1	**0**	**1**	**1**	**0**	**1**	**1**	**1**	**0**	**1**	**1**	**1**	**1**	**1**	**A**
1	**0**	**1**	**1**	**1**	**0**	**0**	**1**	**1**	**1**	**1**	**1**	**0**	**1**	**B**
1	**0**	**1**	**1**	**1**	**1**	**0**	**0**	**1**	**1**	**1**	**0**	**0**	**1**	**C**
1	**1**	**0**	**0**	**0**	**0**	**0**	**1**	**1**	**1**	**1**	**1**	**0**	**1**	**6**
1	**1**	**0**	**0**	**0**	**1**	**1**	**1**	**0**	**0**	**0**	**0**	**1**	**1**	**7**
1	**1**	**0**	**0**	**1**	**0**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**8**
1	**1**	**0**	**0**	**1**	**1**	**1**	**1**	**1**	**0**	**1**	**1**	**1**	**1**	**9**
1	**1**	**0**	**1**	**0**	**0**	**1**	**1**	**0**	**1**	**1**	**1**	**1**	**1**	**A**
1	**1**	**0**	**1**	**0**	**1**	**0**	**1**	**1**	**1**	**1**	**1**	**0**	**1**	**B**
1	**1**	**0**	**1**	**1**	**0**	**0**	**0**	**1**	**1**	**1**	**0**	**0**	**1**	**C**
1	**1**	**0**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**0**	**1**	**1**	**1**	**D**
1	**1**	**1**	**0**	**0**	**0**	**1**	**1**	**1**	**0**	**0**	**0**	**0**	**1**	**7**
1	**1**	**1**	**0**	**0**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**8**
1	**1**	**1**	**0**	**1**	**0**	**1**	**1**	**1**	**1**	**0**	**1**	**1**	**1**	**9**
1	**1**	**1**	**0**	**1**	**1**	**1**	**1**	**1**	**0**	**1**	**1**	**1**	**1**	**A**
1	**1**	**1**	**1**	**0**	**0**	**0**	**0**	**1**	**1**	**1**	**1**	**1**	**1**	**B**
1	**1**	**1**	**1**	**0**	**1**	**1**	**0**	**0**	**1**	**1**	**1**	**0**	**1**	**C**
1	**1**	**1**	**1**	**1**	**0**	**0**	**1**	**1**	**1**	**1**	**0**	**1**	**1**	**D**
1	**1**	**1**	**1**	**1**	**1**	**1**	**0**	**0**	**1**	**1**	**1**	**1**	**1**	**E**