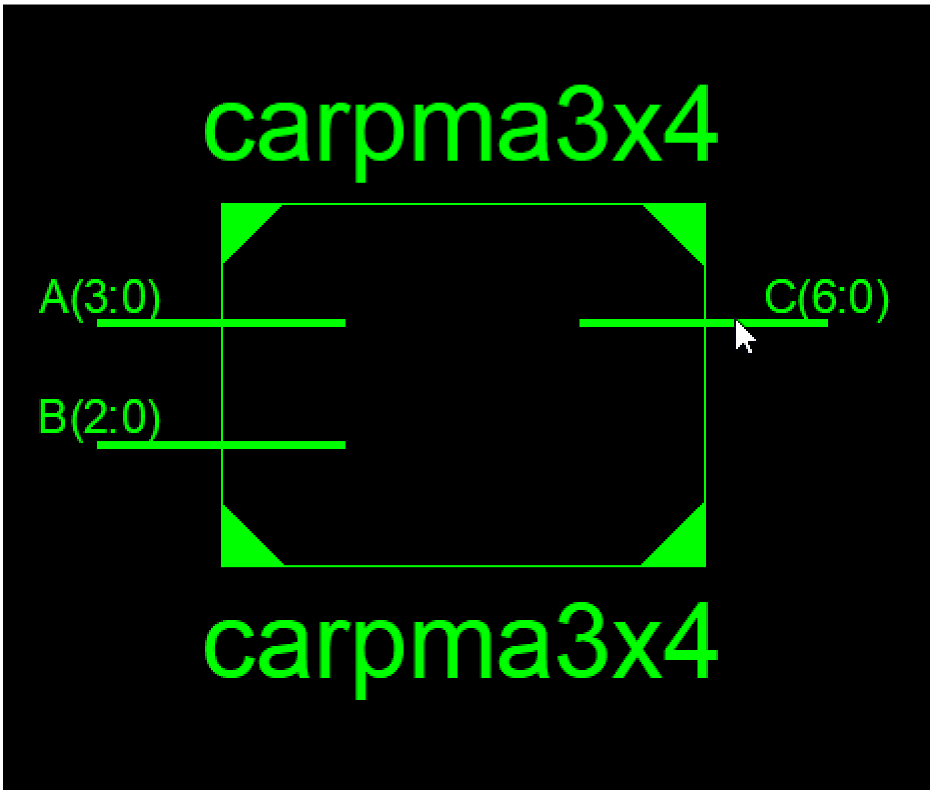
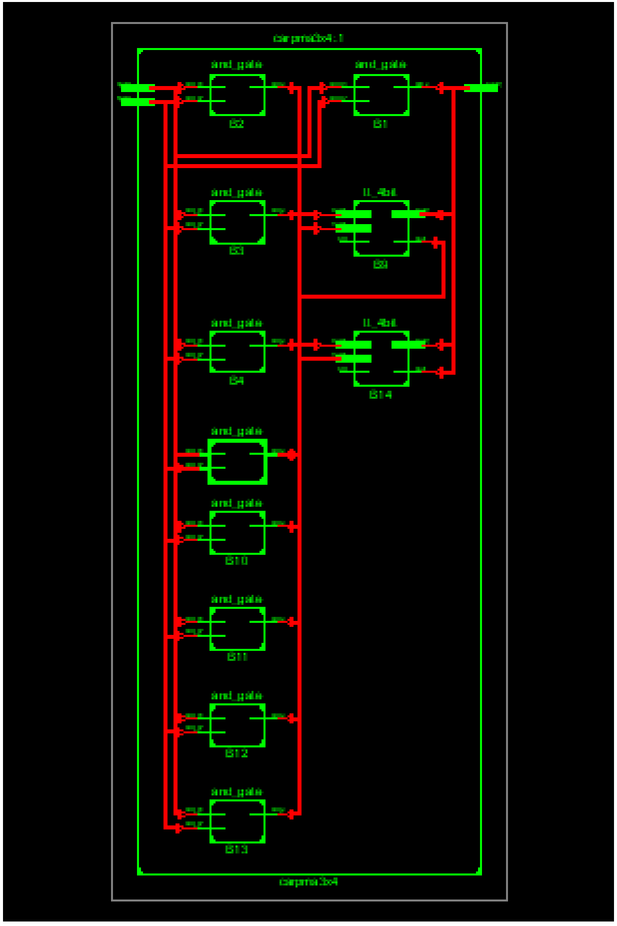
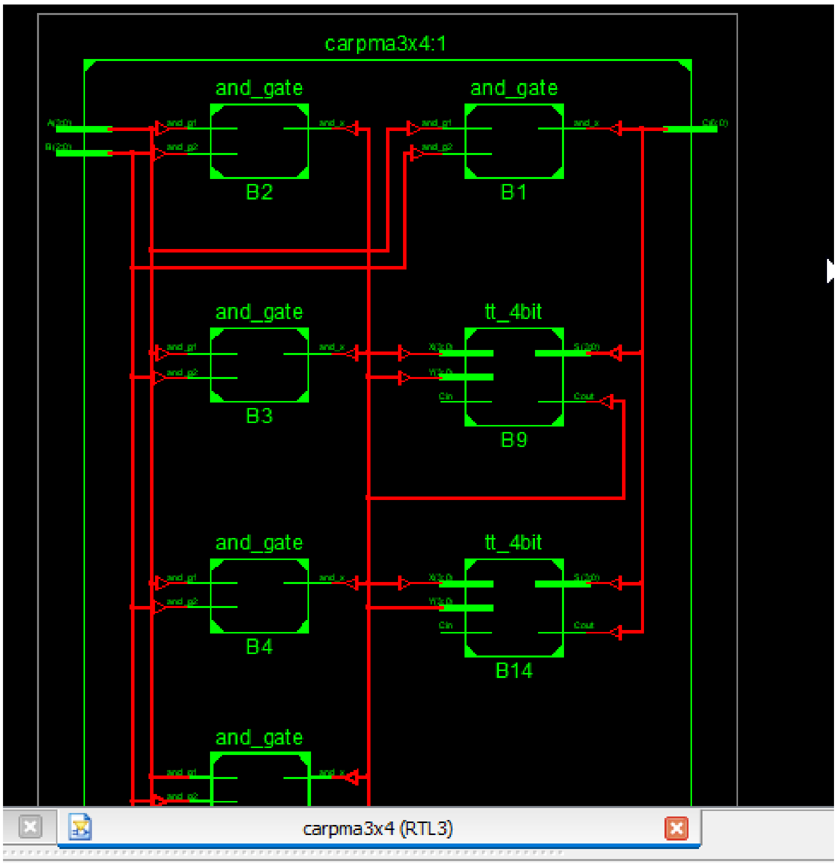
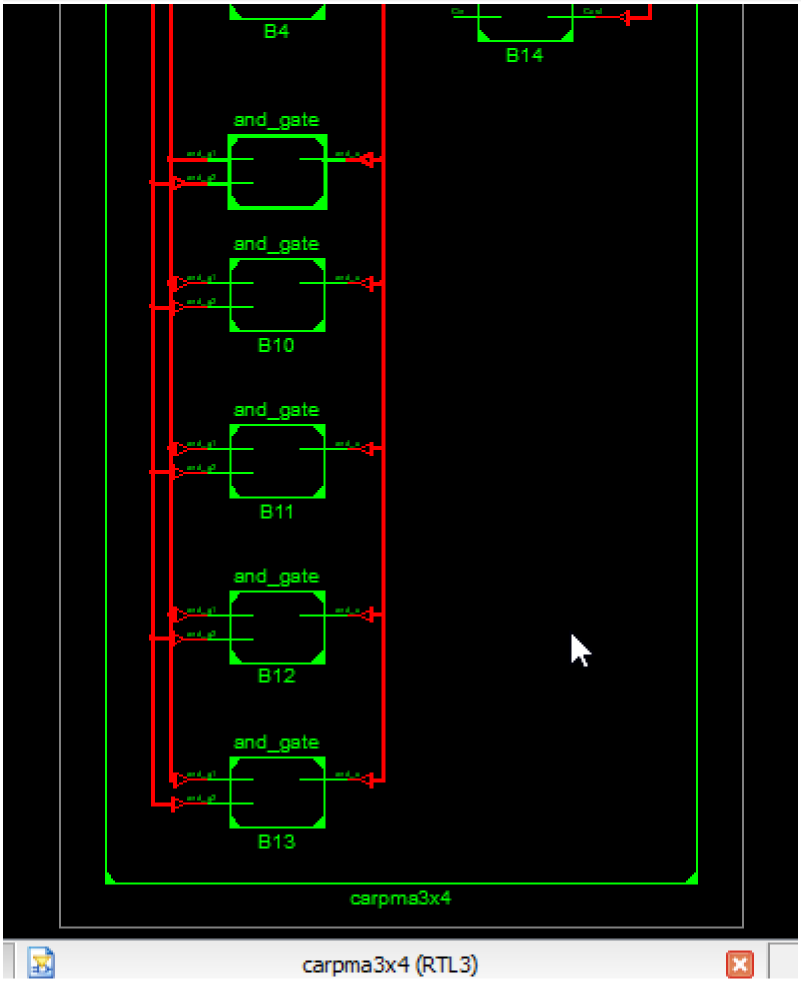

carpma3x4
A(3:0)
B(2:0)
C(6:0)
carpma3x4


carpma3x4:1
and_gate
B2
and_gate
B1
and_gate
B3
ll_4bit
B8
and_gate
B4
ll_4bit
B14
and_gate
and_gate
B10
and_gate
B11
and_gate
B12
and_gate
B13
carpma3x4


carpma3x4:1
and_gate
B2
and_gate
B1
and_g1
and_g2
and_x
and_gate
B3
tt_4bit
B9
X(3:0)
Y(3:0)
Cin
S(3:0)
Cout
and_gate
B4
tt_4bit
B14
and_gate
carpma3x4 (RTL3)


B4
B14
and_gate
and_gate
B10
and_gate
B11
and_gate
B12
and_gate
B13
carpma3x4
carpma3x4 (RTL3)

****

VHDL Kod:

```
--------------------------------------------------------------------------------
-- Company: İstanbul Üniversitesi - Cerrahpaşa
-- Engineer:     Buse Dağıdır
--
-- Create Date:    20:37:50 11/05/2020
-- Design Name:
-- Module Name:    devre - Behavioral
-- Project Name:
-- Target Devices:
-- Tool versions:
-- Description:
--
-- Dependencies:
--
-- Revision:
-- Revision 0.01 - File Created
-- Additional Comments:
--
--------------------------------------------------------------------------------
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;

entity and_gate is
        port( and_g1: in STD_LOGIC;
        and_g2: in STD_LOGIC;
        and_x: out STD_LOGIC);
end and_gate;
```

```
architecture Behavioral of and_gate is
begin
	and_x <= and_g1 and and_g2;
end Behavioral;

------------------------------------------------------------------------
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;

entity or_gate is
	port( or_g1, or_g2: in STD_LOGIC;
	or_x: out STD_LOGIC);
end or_gate;
architecture Behavioral of or_gate is
begin
	or_x <= or_g1 or or_g2;
end Behavioral;
------------------------------------------------------------------------
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;

entity xor_gate is
	port( xor_g1, xor_g2: in STD_LOGIC;
	xor_x: out STD_LOGIC);
end xor_gate;
architecture Behavioral of xor_gate is
begin
	xor_x <= xor_g1 xor xor_g2;
end Behavioral;
```

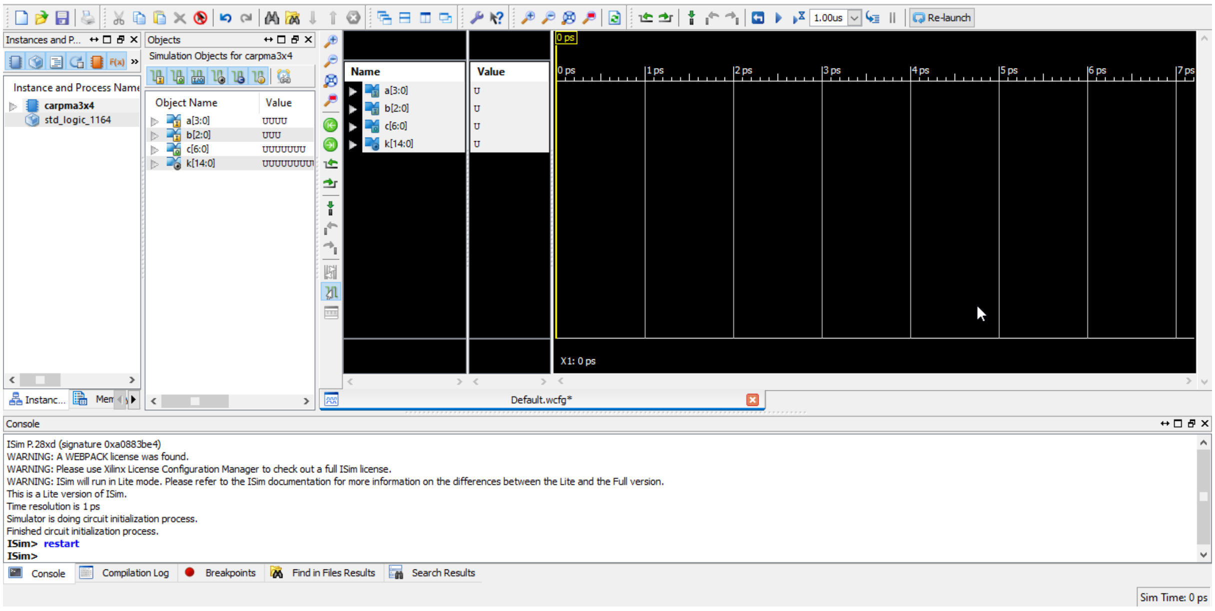
```
--------------------------------------------------------------------------------
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;

entity yari_toplayici is
    port( yt_g1: in STD_LOGIC;
    yt_g2: in STD_LOGIC;
    yt_sum: out STD_LOGIC;
    yt_carryout: out STD_LOGIC);
end yari_toplayici;
architecture Behavioral of yari_toplayici is
    component xor_gate is
        port( xor_g1: in STD_LOGIC;
        xor_g2: in STD_LOGIC;
        xor_x: out STD_LOGIC);
    end component;
    component and_gate is
        port( and_g1: in STD_LOGIC;
        and_g2: in STD_LOGIC;
        and_x: out STD_LOGIC);
    end component;
begin
    -- yt_sum <= yt_g1 xor yt_g2;
    -- yt_carryout <= yt_g1 and yt_g2;
Blok1: xor_gate port map( xor_g1 => yt_g1, xor_g2 => yt_g2, xor_x => yt_sum);
Blok2: and_gate port map( and_g1 => yt_g1, and_g2 => yt_g2, and_x => yt_carryout);
end Behavioral;
--------------------------------------------------------------------------------
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
```

```
entity tam_toplayici is
        port( tt_g1: in STD_LOGIC;
        tt_g2: in STD_LOGIC;
        tt_eldegiris: in STD_LOGIC;
        tt_sum: out STD_LOGIC;
        tt_carryout: out STD_LOGIC);
end tam_toplayici;
architecture Behavioral of tam_toplayici is
        component yari_toplayici is
                port(yt_g1: in STD_LOGIC;
                yt_g2: in STD_LOGIC;
                yt_sum: out STD_LOGIC;
                yt_carryout: out STD_LOGIC);
        end component;
        component or_gate is
                port( or_g1: in STD_LOGIC;
                or_g2: in STD_LOGIC;
                or_x: out STD_LOGIC);
        end component;
                signal K1: STD_LOGIC;
                signal K2: STD_LOGIC;
                signal K3: STD_LOGIC;
begin
        B1: yari_toplayici port map( yt_g1 => tt_g1, yt_g2 => tt_g2, yt_sum => K1,
                                                        yt_carryout => K3 );
        B2: yari_toplayici port map( yt_g1 => tt_eldegiris, yt_g2 => K1, yt_sum => tt_sum,
                                                        yt_carryout => K2);
        B3: or_gate port map( or_g1 => K2, or_g2 => K3, or_x => tt_carryout);
end Behavioral;
--------------------------------------------------------------------------------
```

```
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;


entity tt_4bit is
        port(X: in STD_LOGIC_VECTOR(3 downto 0); --3 2 1 0--
        Y: in STD_LOGIC_VECTOR(3 downto 0);
        Cin: in STD_LOGIC;
        S: out STD_LOGIC_VECTOR(3 downto 0);
        Cout: out STD_LOGIC);
end tt_4bit;
architecture Behavioral of tt_4bit is
        component tam_toplayici is
        port( tt_g1: in STD_LOGIC;
        tt_g2: in STD_LOGIC;
        tt_eldegiris: in STD_LOGIC;
        tt_sum: out STD_LOGIC;
        tt_carryout: out STD_LOGIC);
        end component;
        signal arakablo:STD_LOGIC_VECTOR(2 downto 0);
begin
        B1: tam_toplayici port map(X(0), Y(0), Cin, S(0), arakablo(0)); -- => yapmana gerek yok eğer sadece aktarıldığı yeri yazarsan --
        B2: tam_toplayici port map(X(1), Y(1), arakablo(0), S(1), arakablo(1));
        B3: tam_toplayici port map(X(2), Y(2), arakablo(1), S(2), arakablo(2));
        B4: tam_toplayici port map(X(3), Y(3), arakablo(2), S(3), Cout);
end Behavioral;


--------------------------------------------------------------------------------
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
```

```
entity tt_nbit is
        generic(bit_sayisi: integer:=16);
        port(A: in STD_LOGIC_VECTOR(bit_sayisi-1 downto 0); --3 2 1 0--
        B: in STD_LOGIC_VECTOR(bit_sayisi-1 downto 0);
        Cin: in STD_LOGIC;
        S: out STD_LOGIC_VECTOR(bit_sayisi-1 downto 0);
        Cout: out STD_LOGIC);
end tt_nbit;
architecture Behavioral of tt_nbit is
        component tam_toplayici is
        port( tt_g1: in STD_LOGIC;
                tt_g2: in STD_LOGIC;
                tt_eldegiris: in STD_LOGIC;
                tt_sum: out STD_LOGIC;
                tt_carryout: out STD_LOGIC);
        end component;
        signal arakablo:STD_LOGIC_VECTOR(bit_sayisi-2 downto 0);
begin
        ILKTOPLAYICI: tam_toplayici port map(A(0), B(0), Cin, S(0), arakablo(0));
        ARABLOK:
                for I in 1 to bit_sayisi-2
                generate
                        ORTABLOKLAR: tam_toplayici port map(A(I), B(I), arakablo(I-1),
                                                            S(I), arakablo(I));
                end generate
        ARABLOK;
        SONBLOK: tam_toplayici port map(A(bit_sayisi-1), B(bit_sayisi-1),
                                arakablo(bit_sayisi-2), S(bit_sayisi-1), Cout);
end Behavioral;
```

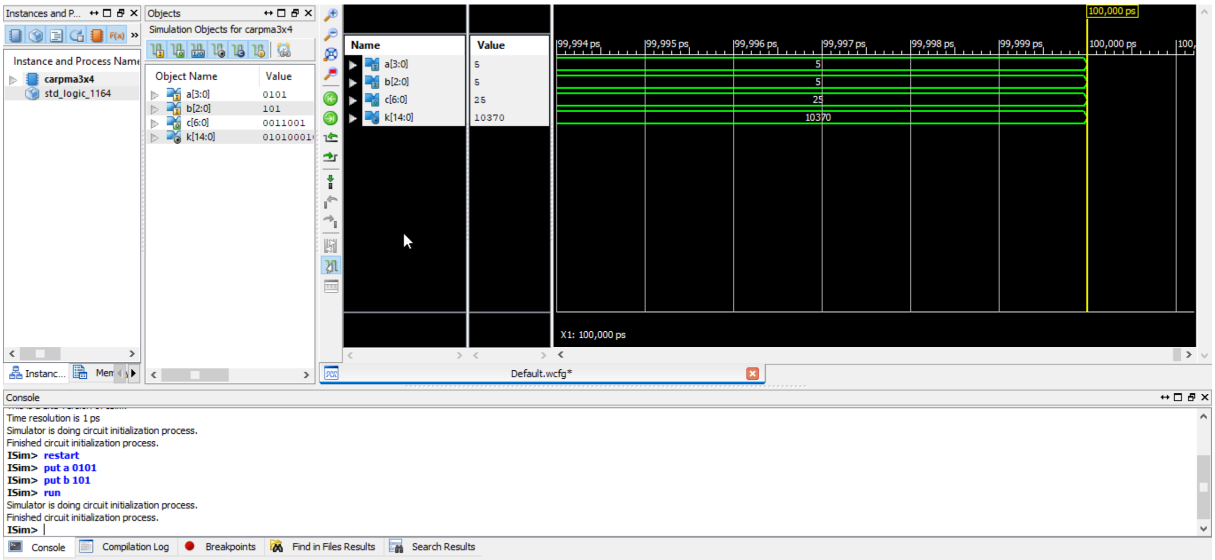
```
--------------------------------------------------------------------------------
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;

entity carpma3x4 is
        port(A: in STD_LOGIC_VECTOR(3 downto 0);
             B: in STD_LOGIC_VECTOR(2 downto 0);
             C: out STD_LOGIC_VECTOR(6 downto 0));
end carpma3x4;

architecture Behavioral of carpma3x4 is
        component tt_4bit is
                port(X: in STD_LOGIC_VECTOR(3 downto 0); --3 2 1 0--
                     Y: in STD_LOGIC_VECTOR(3 downto 0);
                     Cin: in STD_LOGIC;
                     S: out STD_LOGIC_VECTOR(3 downto 0);
                     Cout: out STD_LOGIC);
        end component;
        component and_gate is
        port( and_g1: in STD_LOGIC;
              and_g2: in STD_LOGIC;
              and_x: out STD_LOGIC);
        end component;

        signal K: STD_LOGIC_VECTOR(14 downto 0);

begin
        B1: and_gate port map(A(0), B(0), C(0));
        B2: and_gate port map(A(1), B(0), K(0));
        B3: and_gate port map(A(2), B(0), K(1));
        B4: and_gate port map(A(3), B(0), K(2));
        B5: and_gate port map(A(0), B(1), K(3));
        B6: and_gate port map(A(1), B(1), K(4));
        B7: and_gate port map(A(2), B(1), K(5));
        B8: and_gate port map(A(3), B(1), K(6));
        B9: tt_4bit port map(X(0)=>K(0), X(1)=>K(1), X(2)=>K(2), X(3)=>'0',
              Y(0)=>K(3), Y(1)=>K(4), Y(2)=>K(5), Y(3)=>K(6), Cin=>'0',S(0)=>C(1),
              S(1)=>K(7), S(2)=>K(8), S(3)=>K(9), Cout=>K(10));
        B10: and_gate port map(A(0), B(2), K(11));
        B11: and_gate port map(A(1), B(2), K(12));
        B12: and_gate port map(A(2), B(2), K(13));
        B13: and_gate port map(A(3), B(2), K(14));
        B14: tt_4bit port map(X(0)=>K(7), X(1)=>K(8), X(2)=>K(9), X(3)=>K(10),
              Y(0)=>K(11), Y(1)=>K(12), Y(2)=>K(13), Y(3)=>K(14), Cin=>'0',
              S(0)=>C(2), S(1)=>C(3), S(2)=>C(4), S(3)=>C(5), Cout=>C(6));

end Behavioral;
```



Object Name	Value
a[3:0]	UUUU
b[2:0]	UUU
c[6:0]	UUUUUUU
k[14:0]	UUUUUUUUU

Name	Value
a[3:0]	U
b[2:0]	U
c[6:0]	U
k[14:0]	U

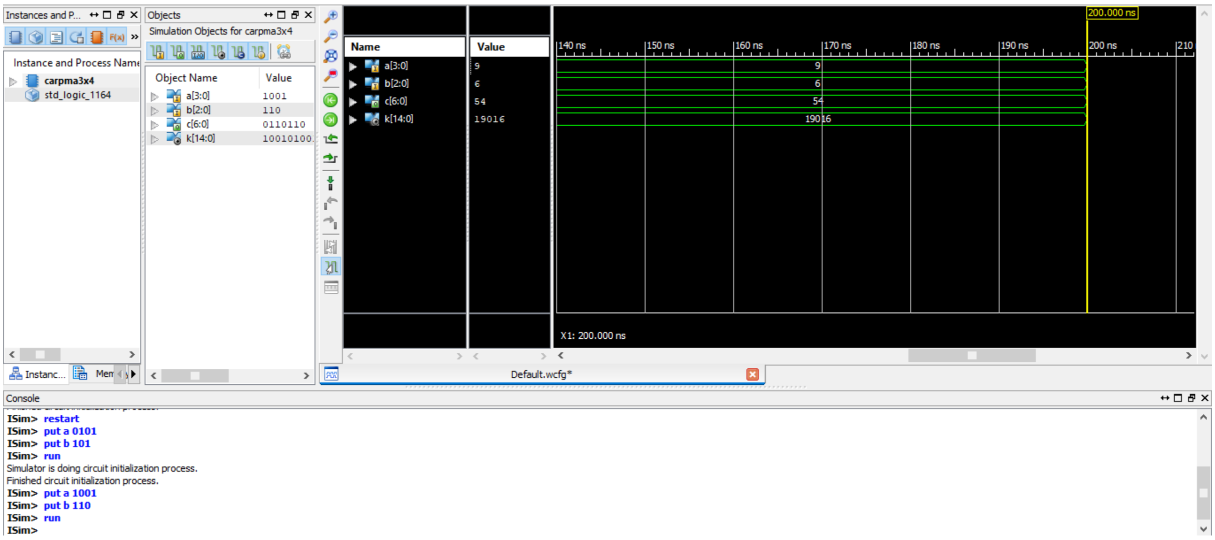
```
ISim P.28xd (signature 0xa0883be4)
WARNING: A WEBPACK license was found.
WARNING: Please use Xilinx License Configuration Manager to check out a full ISim license.
WARNING: ISim will run in Lite mode. Please refer to the ISim documentation for more information on the differences between the Lite and the Full version.
This is a Lite version of ISim.
Time resolution is 1 ps
Simulator is doing circuit initialization process.
Finished circuit initialization process.
ISim> restart
ISim>
```



Object Name	Value
a[3:0]	0101
b[2:0]	101
c[6:0]	0011001
k[14:0]	01010001

Name	Value
a[3:0]	5
b[2:0]	5
c[6:0]	25
k[14:0]	10370

```
Time resolution is 1 ps
Simulator is doing circuit initialization process.
Finished circuit initialization process.
ISim> restart
ISim> put a 0101
ISim> put b 101
ISim> run
Simulator is doing circuit initialization process.
Finished circuit initialization process.
ISim>
```



Object Name	Value
a[3:0]	1001
b[2:0]	110
c[6:0]	0110110
k[14:0]	10010100

Name	Value
a[3:0]	9
b[2:0]	6
c[6:0]	54
k[14:0]	19016

```
ISim> restart
ISim> put a 0101
ISim> put b 101
ISim> run
Simulator is doing circuit initialization process.
Finished circuit initialization process.
ISim> put a 1001
ISim> put b 110
ISim> run
ISim>
```