**4 Bitlik Full Adder**

1306160048 Barış Yarar

Kod:

----------1 bitlik full adder--------

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 Behavioral of xor\_kapisi is

begin

process(xor\_g1, xor\_g2)

begin

xor\_cikis <= xor\_g1 xor xor\_g2;

end process;

end Behavioral;

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

library IEEE;

use IEEE.STD\_LOGIC\_1164.ALL;

entity and\_kapisi is

port(and\_g1: in STD\_LOGIC;

and\_g2: in STD\_LOGIC;

and\_cikis: out STD\_LOGIC);

end and\_kapisi;

architecture Behavioral of and\_kapisi is

begin

process(and\_g1, and\_g2)

begin

and\_cikis <= and\_g1 and and\_g2;

end process;

end Behavioral;

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

library IEEE;

use IEEE.STD\_LOGIC\_1164.ALL;

entity or\_kapisi is

port(or\_g1: in STD\_LOGIC;

or\_g2: in STD\_LOGIC;

or\_cikis: out STD\_LOGIC);

end or\_kapisi;

architecture Behavioral of or\_kapisi is

begin

process(or\_g1, or\_g2)

begin

or\_cikis <= or\_g1 or or\_g2;

end process;

end Behavioral;

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

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 Behavioral of FullAdder is

component xor\_kapisi is

port(xor\_g1: in STD\_LOGIC;

xor\_g2: in STD\_LOGIC;

xor\_cikis: out STD\_LOGIC);

end component;

component and\_kapisi is

port(and\_g1: in STD\_LOGIC;

and\_g2: in STD\_LOGIC;

and\_cikis: out STD\_LOGIC);

end component;

component or\_kapisi is

port(or\_g1: in STD\_LOGIC;

or\_g2: in STD\_LOGIC;

or\_cikis: out STD\_LOGIC);

end component;

signal ara1: STD\_LOGIC;

signal ara2: STD\_LOGIC;

signal ara3: STD\_LOGIC;

begin

blok1: xor\_kapisi port map(xor\_g1=>x, xor\_g2=>y, xor\_cikis=>ara1);

blok2: xor\_kapisi port map(xor\_g1 =>ara1, xor\_g2 =>z, xor\_cikis => S);

blok3: and\_kapisi port map(and\_g1 =>ara1, and\_g2 =>z, and\_cikis => ara2);

blok4: and\_kapisi port map(and\_g1 =>x, and\_g2 =>y, and\_cikis => ara3);

blok5: or\_kapisi port map(or\_g1 =>ara2, or\_g2 =>ara3, or\_cikis => C);

end Behavioral;

--------------4 bitlik full adder----------------

library IEEE;

use IEEE.STD\_LOGIC\_1164.ALL;

entity fourbitFA is

port(A: in STD\_LOGIC\_VECTOR (3 downto 0);

B: in STD\_LOGIC\_VECTOR (3 downto 0);

S: out STD\_LOGIC\_VECTOR (3 downto 0);

C4: out STD\_LOGIC);

end fourbitFA;

architecture Behavioral of fourbitFA 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;

begin

FA1: FullAdder port map(A(0),B(0),'0',S(0),C1);

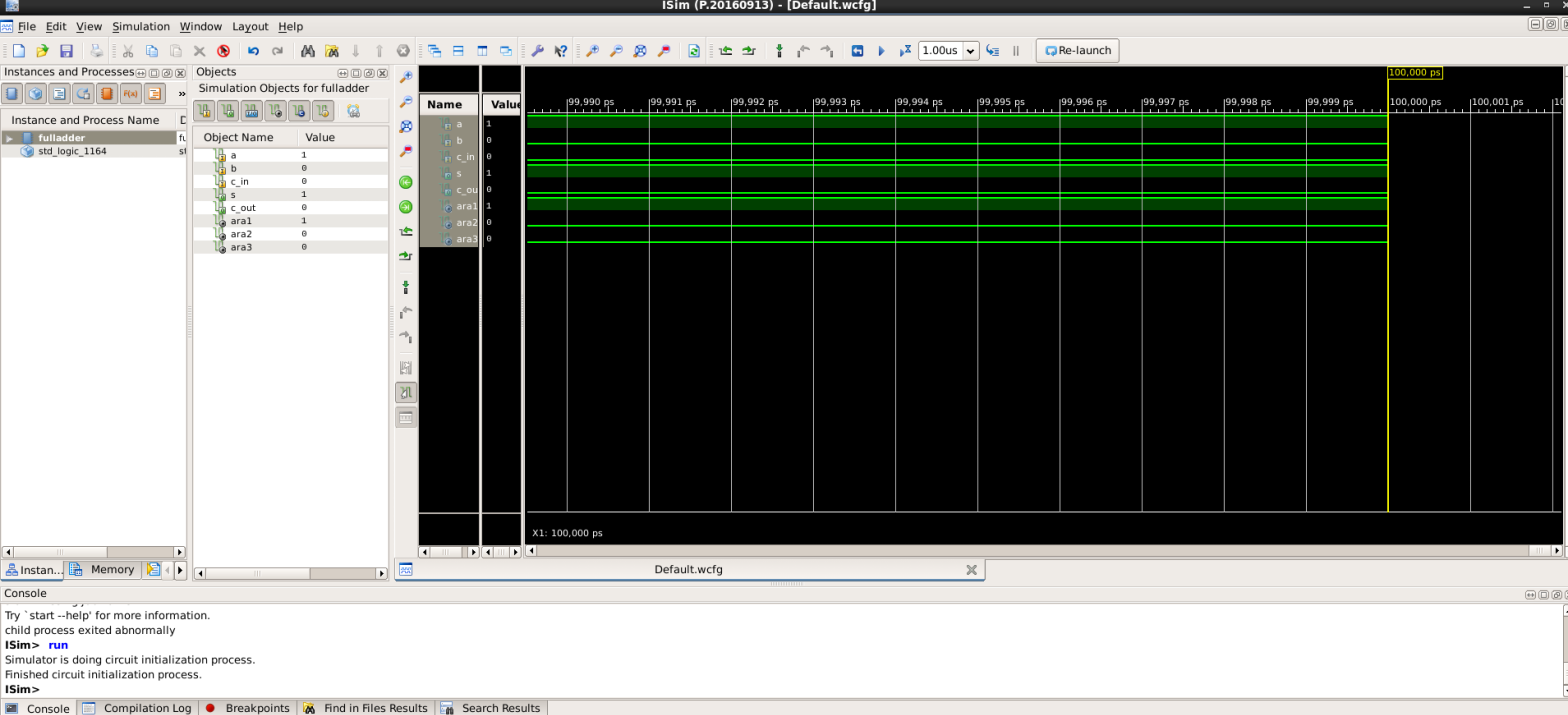
FA2: FullAdder port map(A(1),B(1),C1,S(1),C2);

FA3: FullAdder port map(A(2),B(2),C2,S(2),C3);

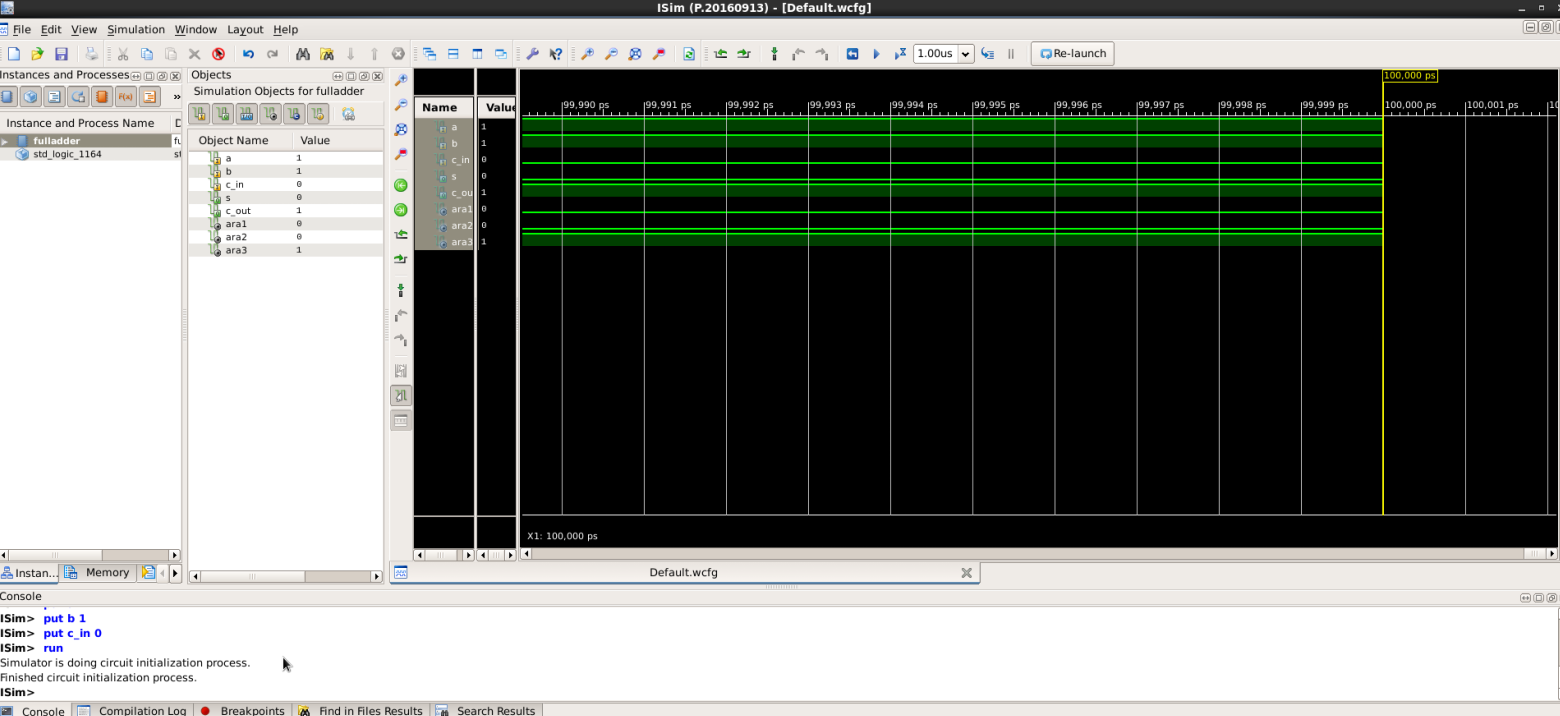
FA4: FullAdder port map(A(3),B(3),C3,S(3),C4);

end Behavioral;

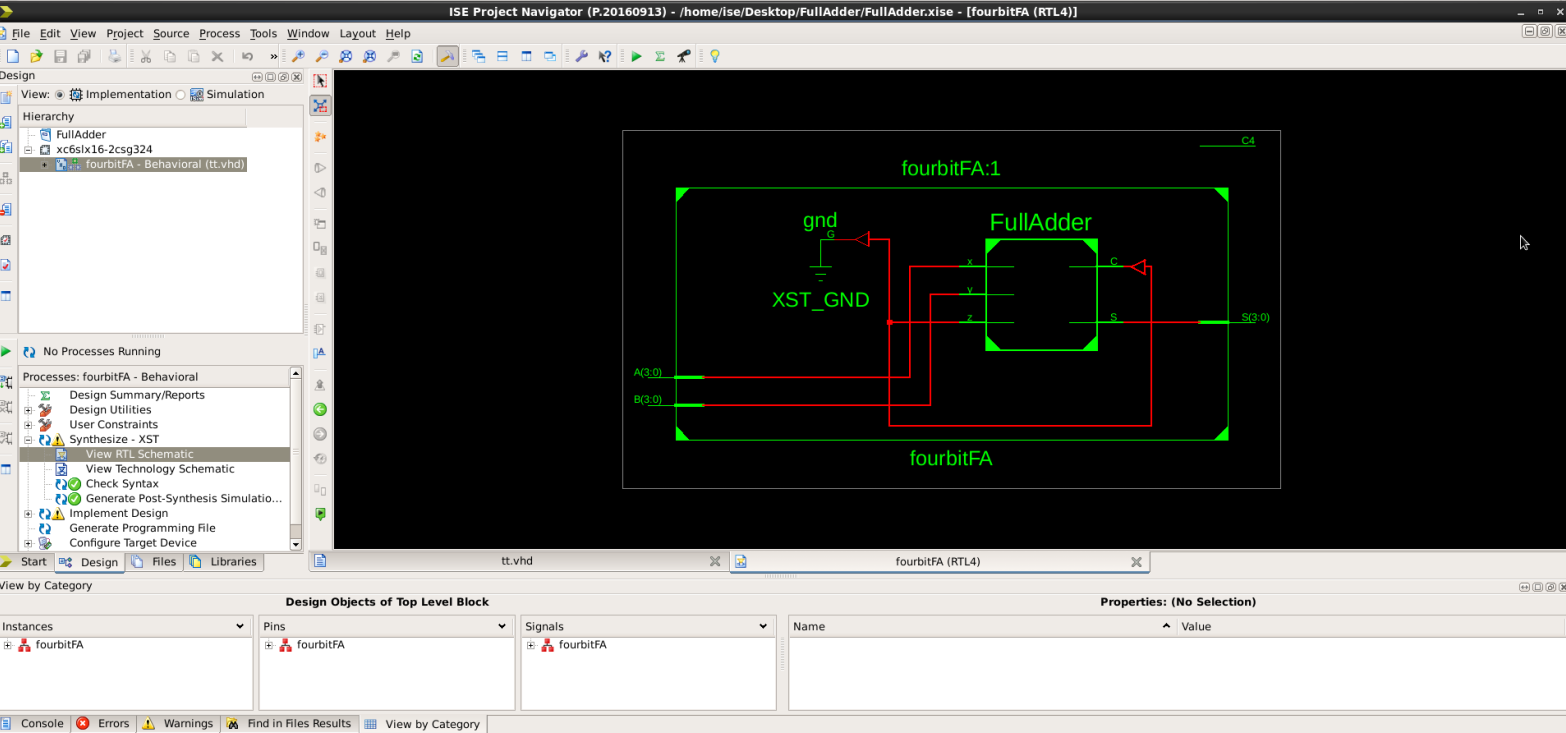
Full Adder’ın çalışıp çalışmadığını test etmek için örnek 1



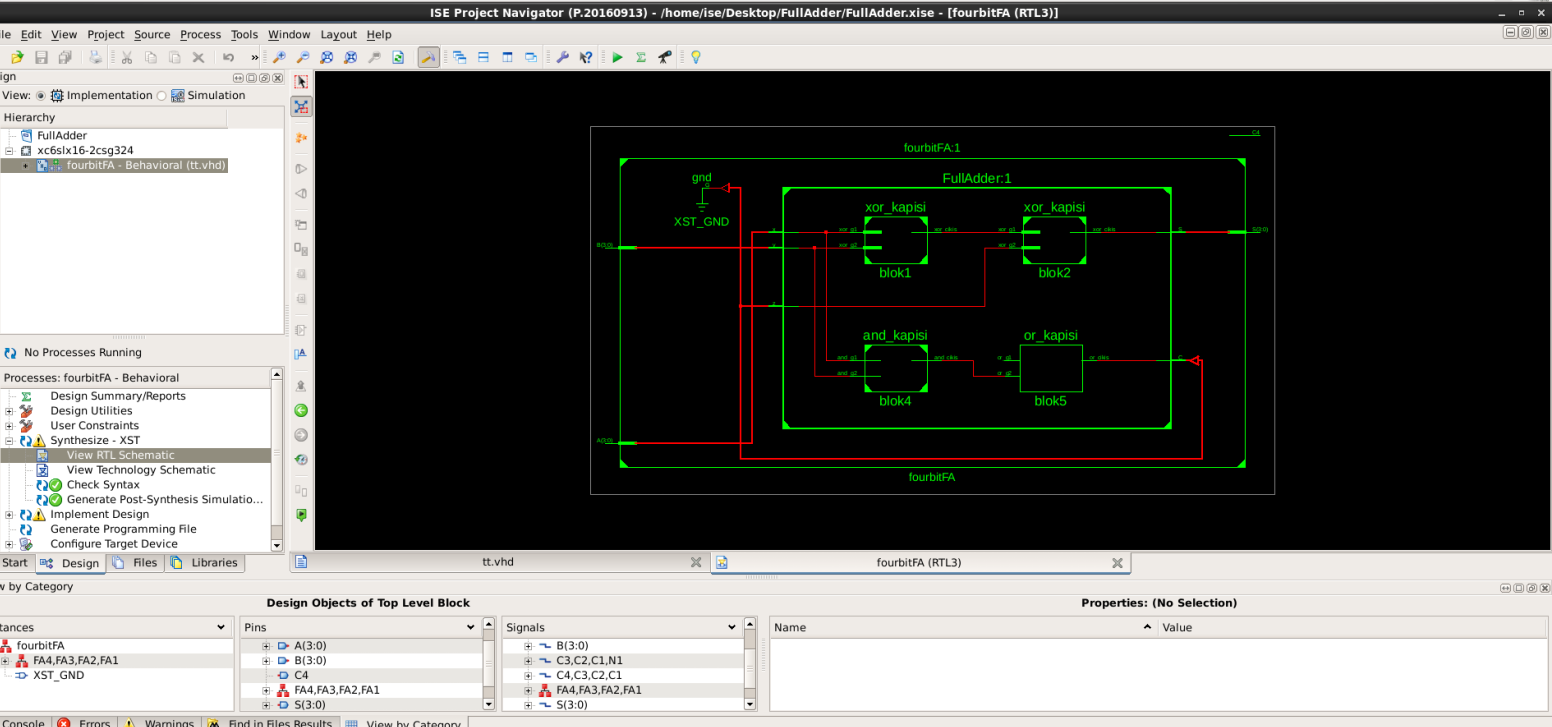
Örnek 2:



4 bitlik Full Adder Şeması

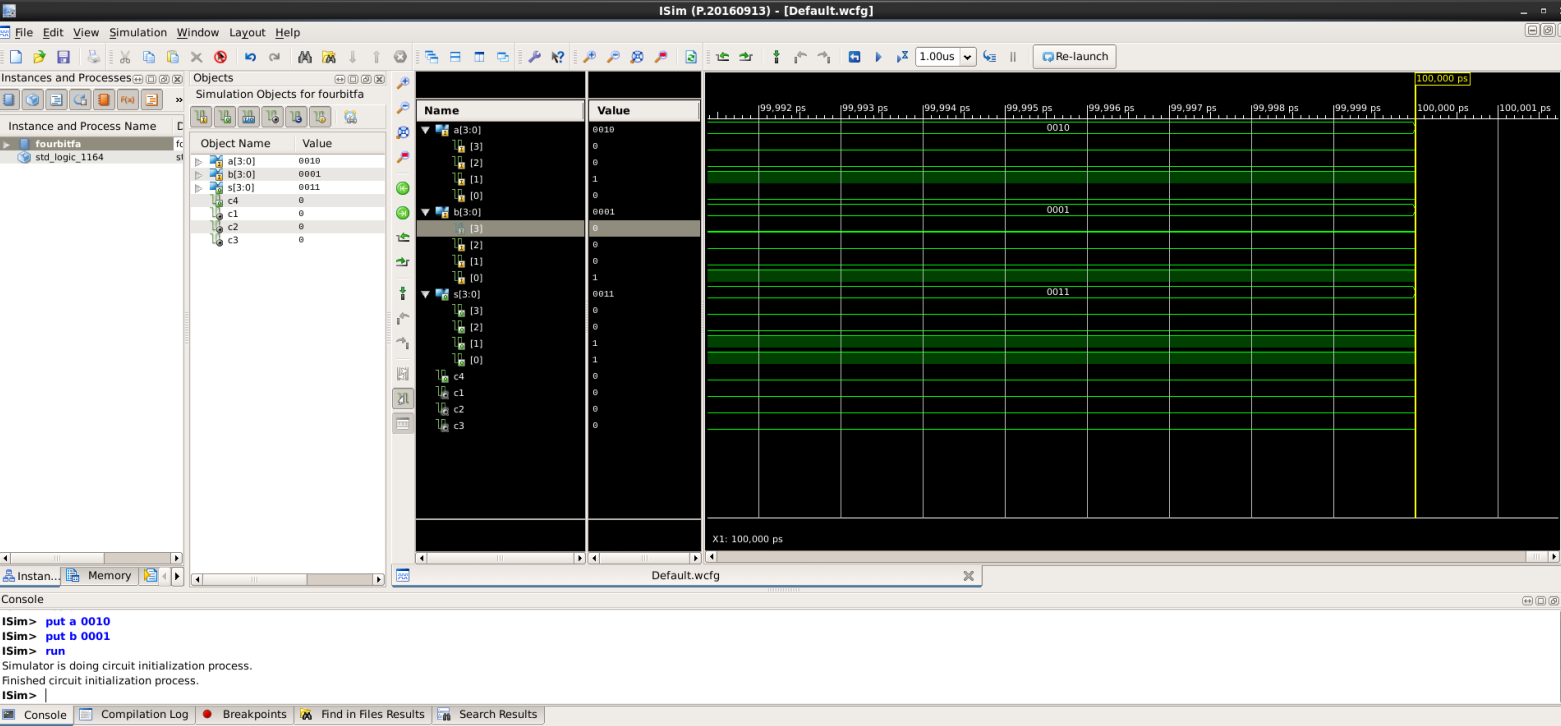


Açık versiyon:



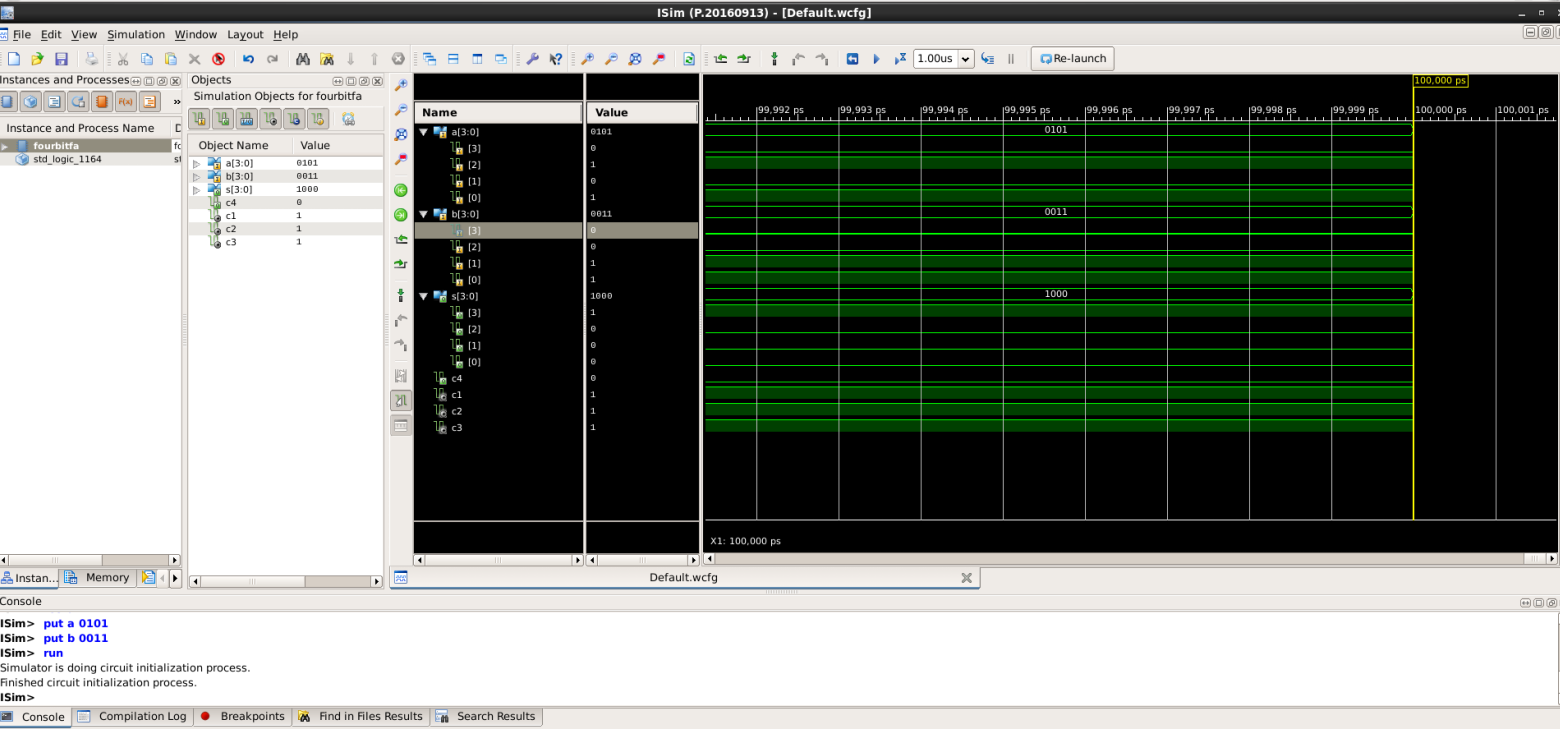
4 bitlik Full Adder örnekleri

Örnek 1:



A: 0010 B: 0001 S: 0011

Örnek 2:



A: 0101 B: 0011 S:1000