INSTITUTE OF ENGINEERING AND TECHNOLOGY, INDORE

CIRCUIT DESIGN USING HDL(EIR7C4)

Lab Assignment

Session (2020-2021)

TOPIC: TWO BIT SUBTRACTOR USING STRUCTURAL FLOW

Submitted To
Dr. Vaibhav Neema

Submitted By
Akshita Ranawat
Roll No.: 17E7007
EI(7th Sem)
Enroll no. DE17199

Structural Modeling Architecture in VHDL

- The term structural modeling is the terminology that VHDL uses for modular design.
- If you are designing a complex project, you should split into two or more simple designs in order to easily handle the complexity. Modular designs allow you to pack low-level functionality into modules.
- This approach allows design reuse without the need to reinvent and re-test the wheel every time.
- We use signals to interconnect components and eventually create large systems using small sub-systems. Structural design deals with interconnection of gates, how gates are connected to each other.
- Entity declaration(no need to mention Intermediate signals only input and output)
- In Structural modeling, the architecture body is composed of two parts − the declarative part (before the keyword ‘begin’) and the statement part (after the keyword ‘begin’).
- Component Declaration, (we define the name, Inputs/outputs, and type of Inputs/outputs of the component).
- Component Instantiation

  ```
  component label: component-name port map (
  association-list) ;
  ```

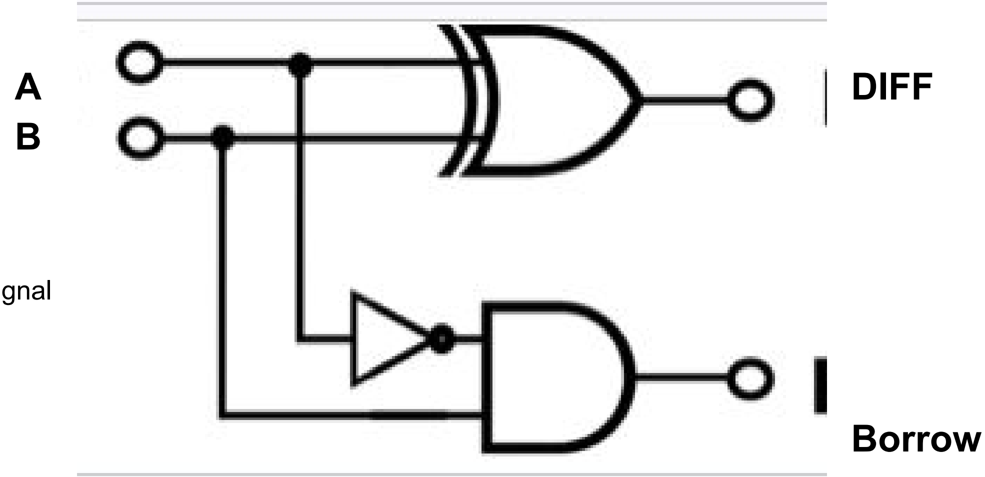
Two Bit Subtractor

Two bit subtractor also known as half subtractor is a combinational circuit used to perform subtraction of two bits. If it has two inputs, the minuend X and subtrahend Y and two outputs the Difference D and Borrow out B. The borrow out signal is set when the subtractor needs to borrow from the next digit. That is Borrow=1, A<B, B=1 if A=0 B=1.The logic of half subtractor looks like this,

$$\mathrm{DIFF} = A \oplus B$$

$$\mathrm{Borrow} = \overline{A}.B$$

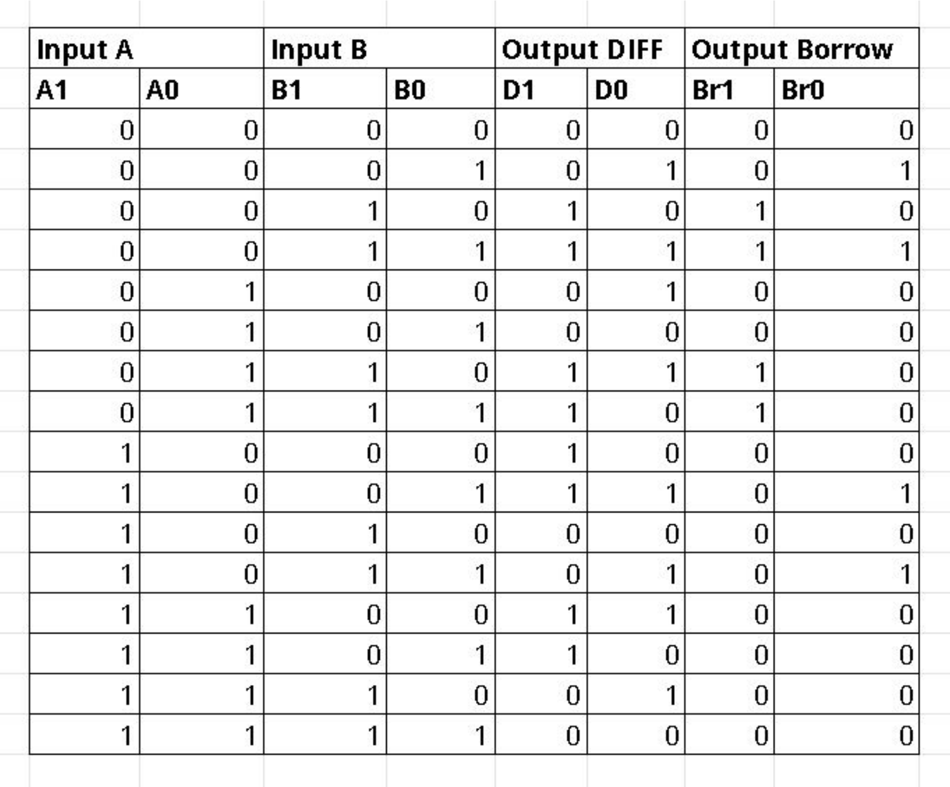
Logic Diagram for Two Bit Subtractor



Intermediate signal

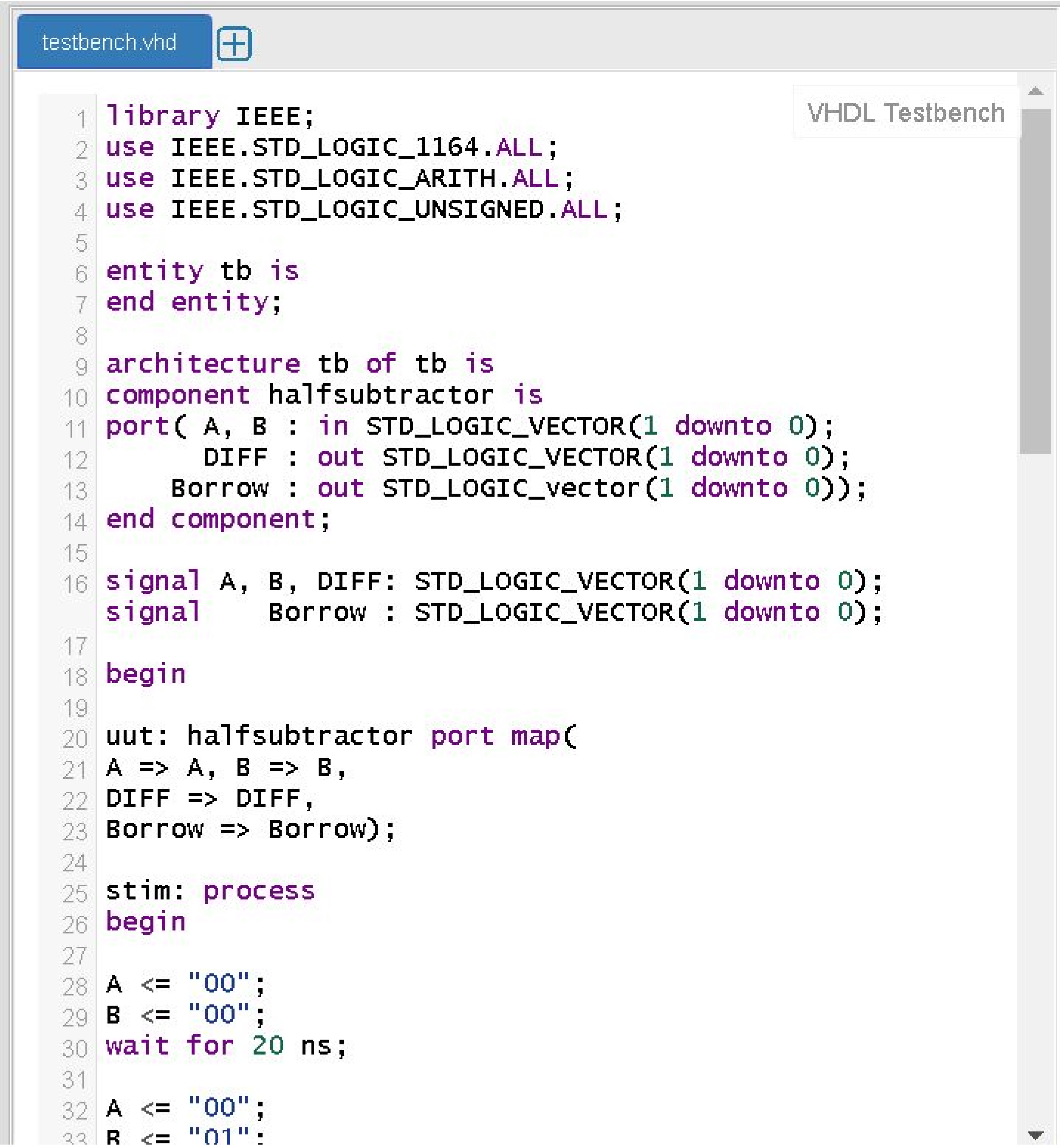
is **s**

Truth Table for Two Bit Subtractor

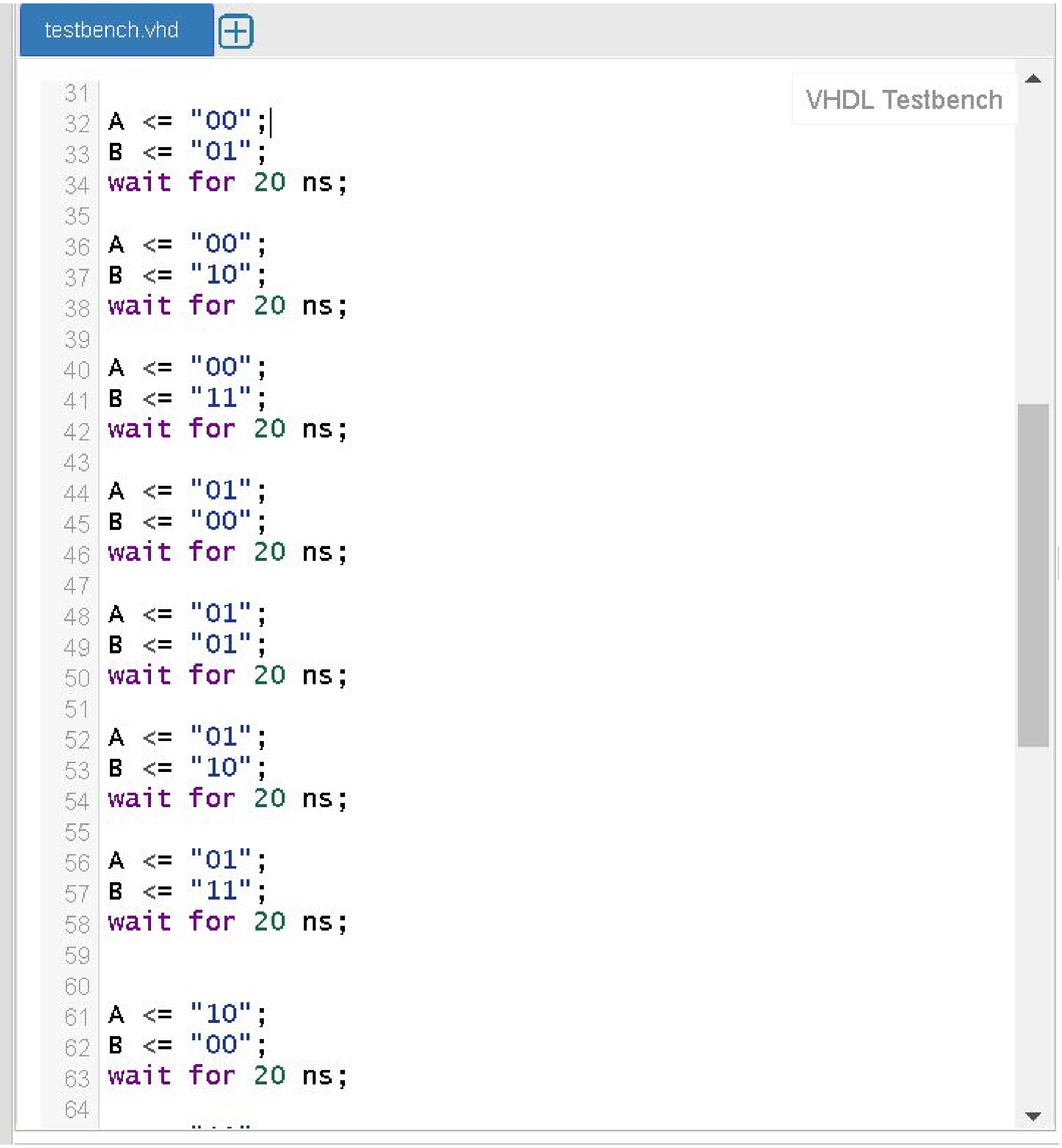


Input A		Input B		Output DIFF		Output Borrow	
A1	A0	B1	B0	D1	D0	Br1	Br0
0	0	0	0	0	0	0	0
0	0	0	1	0	1	0	1
0	0	1	0	1	0	1	0
0	0	1	1	1	1	1	1
0	1	0	0	0	1	0	0
0	1	0	1	0	0	0	0
0	1	1	0	1	1	1	0
0	1	1	1	1	0	1	0
1	0	0	0	1	0	0	0
1	0	0	1	1	1	0	1
1	0	1	0	0	0	0	0
1	0	1	1	0	1	0	1
1	1	0	0	1	1	0	0
1	1	0	1	1	0	0	0
1	1	1	0	0	1	0	0
1	1	1	1	0	0	0	0

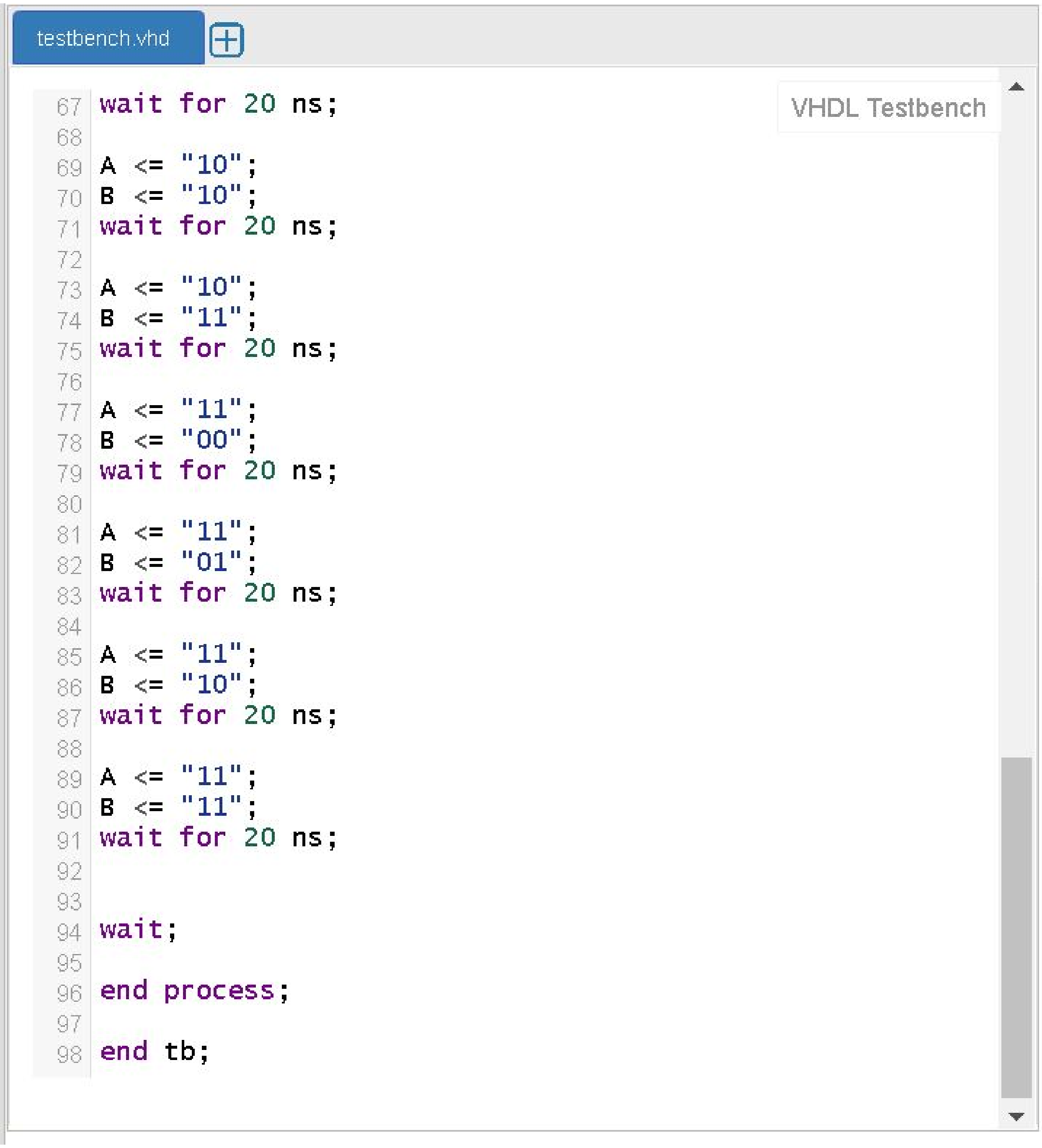
Test Bench for Two Bit Subtractor



```vhdl
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
use IEEE.STD_LOGIC_ARITH.ALL;
use IEEE.STD_LOGIC_UNSIGNED.ALL;

entity tb is
end entity;

architecture tb of tb is
component halfsubtractor is
port( A, B : in STD_LOGIC_VECTOR(1 downto 0);
      DIFF : out STD_LOGIC_VECTOR(1 downto 0);
    Borrow : out STD_LOGIC_vector(1 downto 0));
end component;

signal A, B, DIFF: STD_LOGIC_VECTOR(1 downto 0);
signal    Borrow : STD_LOGIC_VECTOR(1 downto 0);

begin

uut: halfsubtractor port map(
A => A, B => B,
DIFF => DIFF,
Borrow => Borrow);

stim: process
begin

A <= "00";
B <= "00";
wait for 20 ns;

A <= "00";
B <= "01";
```

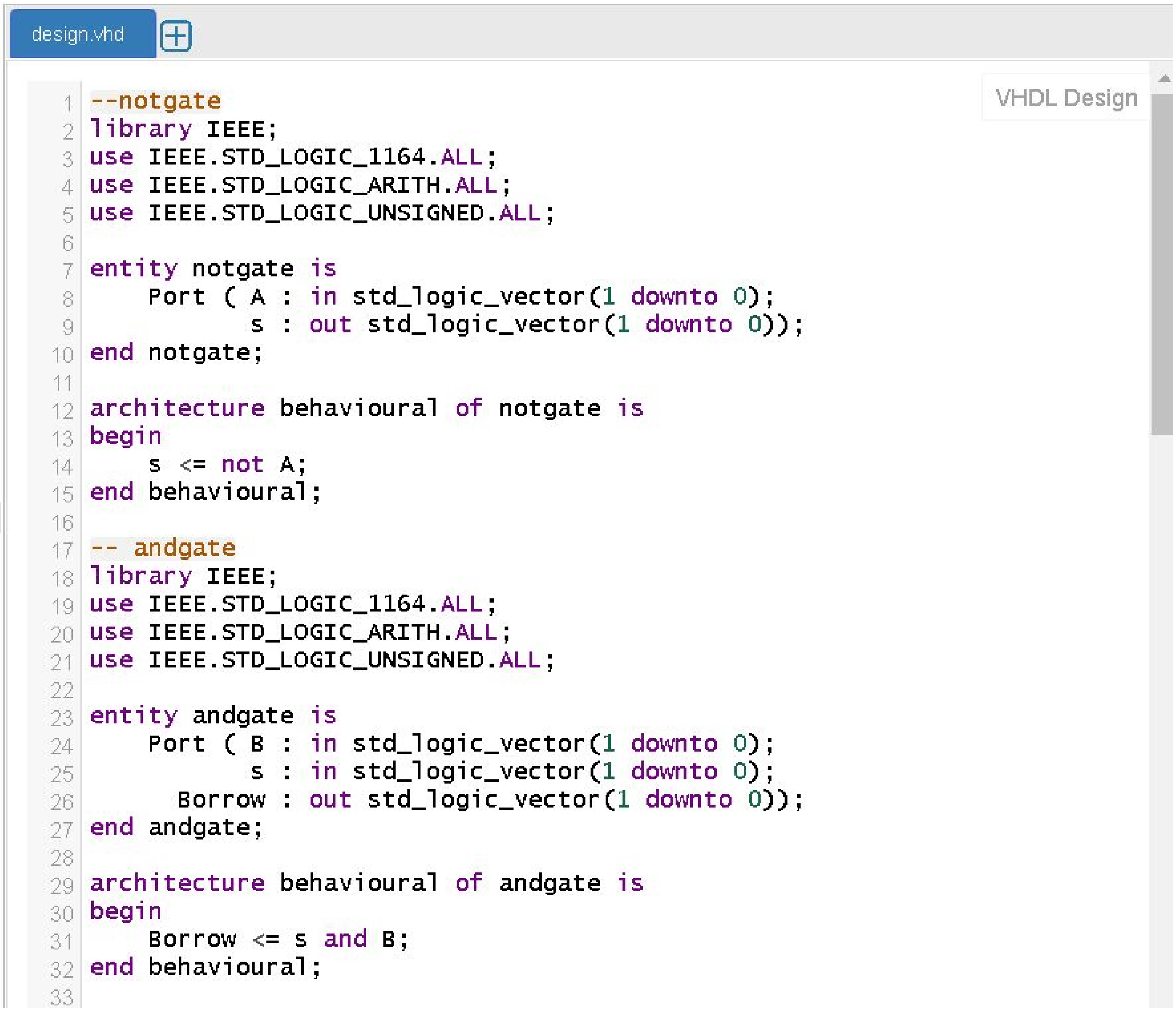


```

A <= "00";
B <= "01";
wait for 20 ns;

A <= "00";
B <= "10";
wait for 20 ns;

A <= "00";
B <= "11";
wait for 20 ns;

A <= "01";
B <= "00";
wait for 20 ns;

A <= "01";
B <= "01";
wait for 20 ns;

A <= "01";
B <= "10";
wait for 20 ns;

A <= "01";
B <= "11";
wait for 20 ns;


A <= "10";
B <= "00";
wait for 20 ns;

```

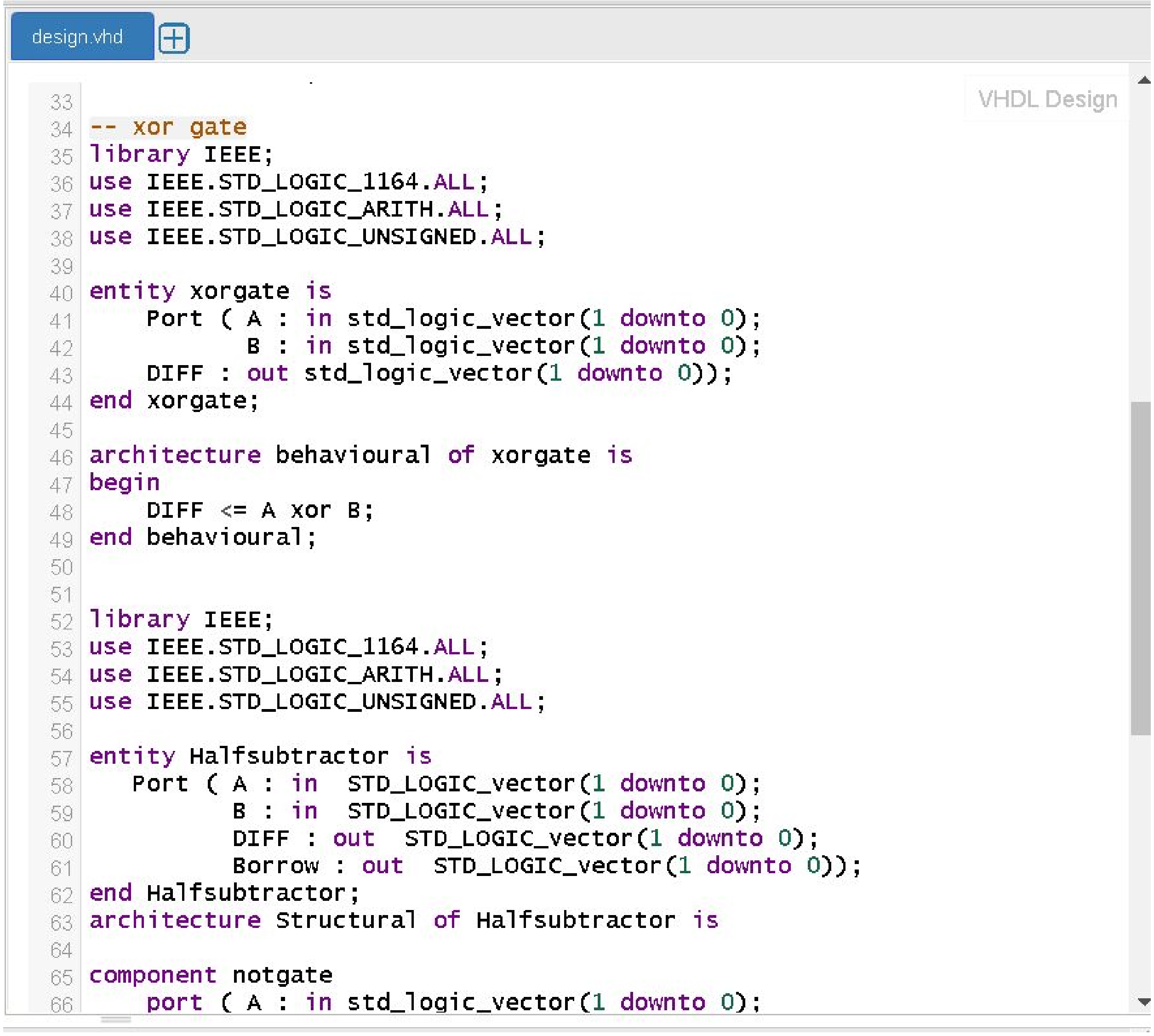


```
wait for 20 ns;

A <= "10";
B <= "10";
wait for 20 ns;

A <= "10";
B <= "11";
wait for 20 ns;

A <= "11";
B <= "00";
wait for 20 ns;

A <= "11";
B <= "01";
wait for 20 ns;

A <= "11";
B <= "10";
wait for 20 ns;

A <= "11";
B <= "11";
wait for 20 ns;


wait;

end process;

end tb;
```

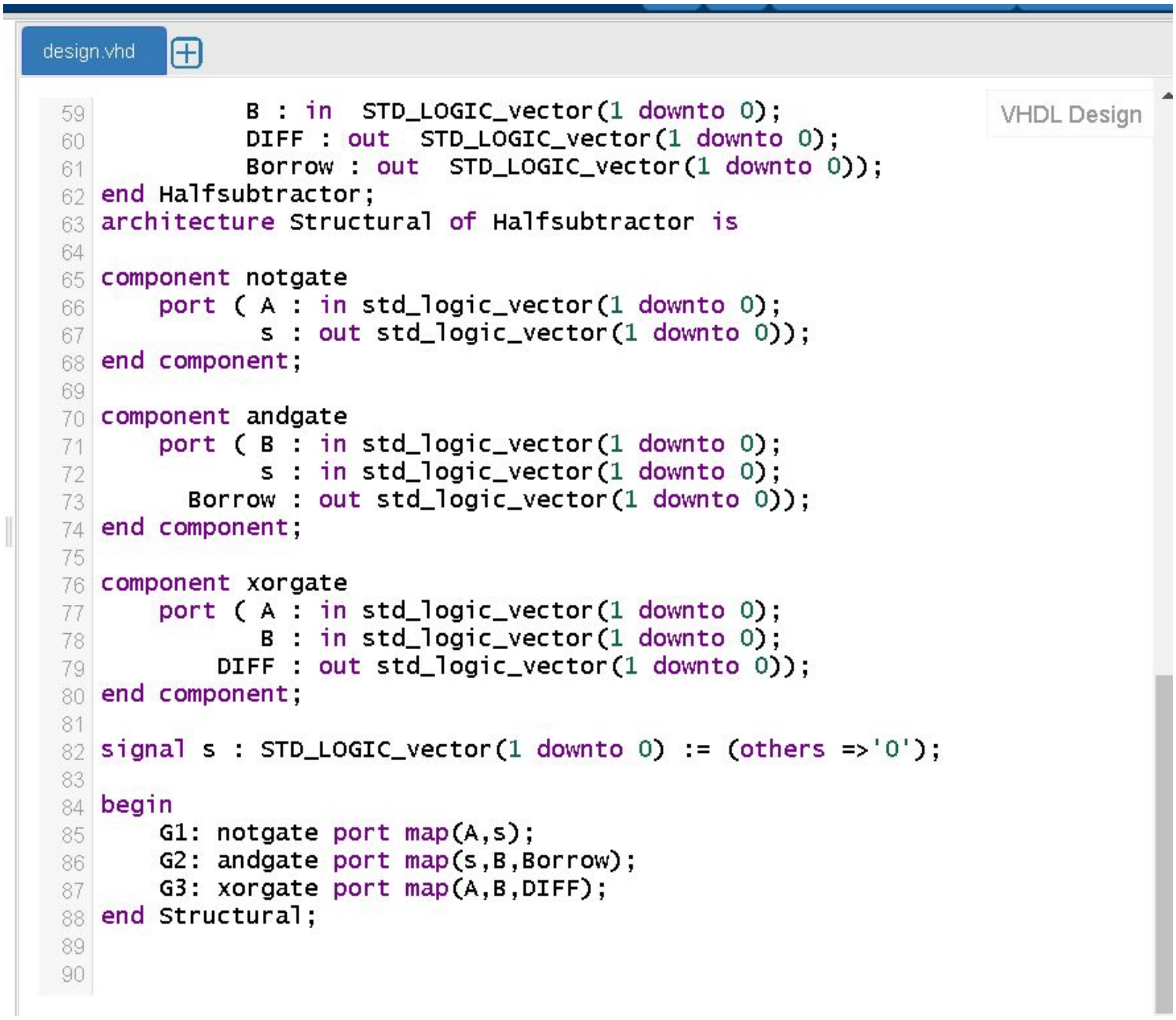
VHDL Design Code for Two Bit Subtractor



```
--notgate
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
use IEEE.STD_LOGIC_ARITH.ALL;
use IEEE.STD_LOGIC_UNSIGNED.ALL;

entity notgate is
    Port ( A : in std_logic_vector(1 downto 0);
           s : out std_logic_vector(1 downto 0));
end notgate;

architecture behavioural of notgate is
begin
    s <= not A;
end behavioural;

-- andgate
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
use IEEE.STD_LOGIC_ARITH.ALL;
use IEEE.STD_LOGIC_UNSIGNED.ALL;

entity andgate is
    Port ( B : in std_logic_vector(1 downto 0);
           s : in std_logic_vector(1 downto 0);
      Borrow : out std_logic_vector(1 downto 0));
end andgate;

architecture behavioural of andgate is
begin
    Borrow <= s and B;
end behavioural;

```

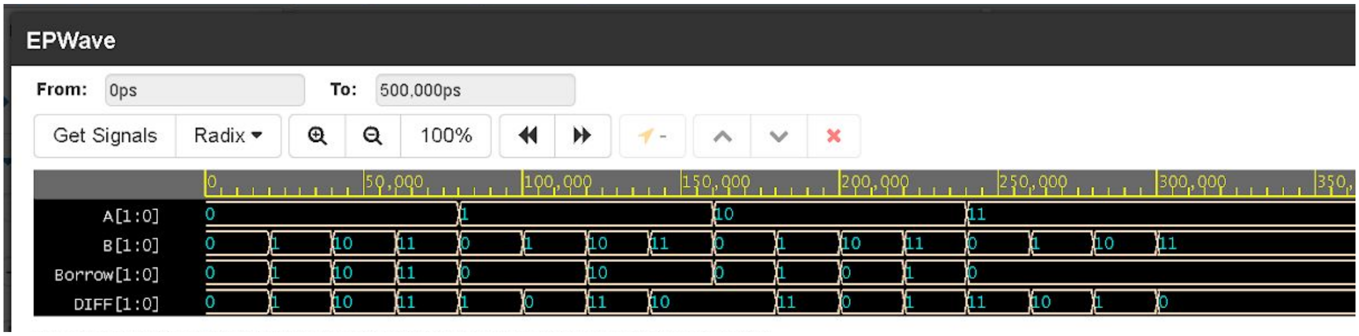


```

-- xor gate
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
use IEEE.STD_LOGIC_ARITH.ALL;
use IEEE.STD_LOGIC_UNSIGNED.ALL;

entity xorgate is
    Port ( A : in std_logic_vector(1 downto 0);
           B : in std_logic_vector(1 downto 0);
    DIFF : out std_logic_vector(1 downto 0));
end xorgate;

architecture behavioural of xorgate is
begin
    DIFF <= A xor B;
end behavioural;


library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
use IEEE.STD_LOGIC_ARITH.ALL;
use IEEE.STD_LOGIC_UNSIGNED.ALL;

entity Halfsubtractor is
   Port ( A : in  STD_LOGIC_vector(1 downto 0);
          B : in  STD_LOGIC_vector(1 downto 0);
          DIFF : out  STD_LOGIC_vector(1 downto 0);
          Borrow : out  STD_LOGIC_vector(1 downto 0));
end Halfsubtractor;
architecture Structural of Halfsubtractor is

component notgate
    port ( A : in std_logic_vector(1 downto 0);
```



```
          B : in  STD_LOGIC_vector(1 downto 0);
          DIFF : out  STD_LOGIC_vector(1 downto 0);
          Borrow : out  STD_LOGIC_vector(1 downto 0));
end Halfsubtractor;
architecture Structural of Halfsubtractor is

component notgate
    port ( A : in std_logic_vector(1 downto 0);
           s : out std_logic_vector(1 downto 0));
end component;

component andgate
    port ( B : in std_logic_vector(1 downto 0);
           s : in std_logic_vector(1 downto 0);
      Borrow : out std_logic_vector(1 downto 0));
end component;

component xorgate
    port ( A : in std_logic_vector(1 downto 0);
           B : in std_logic_vector(1 downto 0);
        DIFF : out std_logic_vector(1 downto 0));
end component;

signal s : STD_LOGIC_vector(1 downto 0) := (others =>'0');

begin
    G1: notgate port map(A,s);
    G2: andgate port map(s,B,Borrow);
    G3: xorgate port map(A,B,DIFF);
end Structural;


```

Wave Form for Two Bit Subtractor


EPWave
Get Signals
Radix
100%
0
50,000
100,000
150,000
200,000
250,000
300,000
350,
A[1:0] 0 1 10 11
B[1:0] 0 1 10 11 0 1 10 11 0 1 10 11 0 1 10 11
Borrow[1:0] 0 1 10 11 0 10 0 1 0 1 0
DIFF[1:0] 0 1 10 11 1 0 11 10 11 0 1 11 10 1 0

From: 0ps **To:** 500,000ps

____________XXXXX____________