**Computer Hardware Experiments**

Lab\_02: Using Port Map

**Name : 李泽宇 Student\_ID: 1709853W-I011-0036**

Part 1: Design a one bit half adder.

Xor gate’s code:

—————————————————————————

library IEEE;

use IEEE.STD\_LOGIC\_1164.ALL;

-- Uncomment the following library declaration if using

-- arithmetic functions with Signed or Unsigned values

--use IEEE.NUMERIC\_STD.ALL;

-- Uncomment the following library declaration if instantiating

-- any Xilinx primitives in this code.

--library UNISIM;

--use UNISIM.VComponents.all;

entity myxor is

Port ( A : in STD\_LOGIC;

B : in STD\_LOGIC;

S : out STD\_LOGIC);

end myxor;

architecture Behavioral of myxor is

begin

S <= A xor B;

end Behavioral;

AND gate’s code:

library IEEE;

use IEEE.STD\_LOGIC\_1164.ALL;

-- Uncomment the following library declaration if using

-- arithmetic functions with Signed or Unsigned values

--use IEEE.NUMERIC\_STD.ALL;

-- Uncomment the following library declaration if instantiating

-- any Xilinx primitives in this code.

--library UNISIM;

--use UNISIM.VComponents.all;

entity myand is

Port ( A : in STD\_LOGIC;

B : in STD\_LOGIC;

C : out STD\_LOGIC);

end myand;

architecture Behavioral of myand is

begin

C <= A and B;

end Behavioral;

Halfadder’s code:

library IEEE;

use IEEE.STD\_LOGIC\_1164.ALL;

-- Uncomment the following library declaration if using

-- arithmetic functions with Signed or Unsigned values

--use IEEE.NUMERIC\_STD.ALL;

-- Uncomment the following library declaration if instantiating

-- any Xilinx primitives in this code.

--library UNISIM;

--use UNISIM.VComponents.all;

entity halfadder is

Port ( A : in STD\_LOGIC;

B : in STD\_LOGIC;

S : out STD\_LOGIC;

C : out STD\_LOGIC);

end halfadder;

architecture Behavioral of halfadder is

component myxor

Port ( A : in STD\_LOGIC;

B : in STD\_LOGIC;

S : out STD\_LOGIC

);

end component;

component myand

Port ( A : in STD\_LOGIC;

B : in STD\_LOGIC;

C : out STD\_LOGIC

);

end component;

begin

myxor1 : myxor port map (A,B,S);

myand1 : myand port map (A,B,C);

end Behavioral;

Than this is the test code:(halfadder code)

LIBRARY ieee;

USE ieee.std\_logic\_1164.ALL;

-- Uncomment the following library declaration if using

-- arithmetic functions with Signed or Unsigned values

--USE ieee.numeric\_std.ALL;

ENTITY halfaddertest IS

END halfaddertest;

ARCHITECTURE behavior OF halfaddertest IS

-- Component Declaration for the Unit Under Test (UUT)

COMPONENT halfadder

PORT(

A : IN std\_logic;

B : IN std\_logic;

S : OUT std\_logic;

C : OUT std\_logic

);

END COMPONENT;

--Inputs

signal A : std\_logic := '0';

signal B : std\_logic := '0';

--Outputs

signal S : std\_logic;

signal C : std\_logic;

-- No clocks detected in port list. Replace <clock> below with

-- appropriate port name

-- constant <clock>\_period : time := 10 ns;

BEGIN

-- Instantiate the Unit Under Test (UUT)

uut: halfadder PORT MAP (

A,

B,

S,

C

);

-- Clock process definitions

-- <clock>\_process :process

-- begin

-- <clock> <= '0';

-- wait for <clock>\_period/2;

-- <clock> <= '1';

-- wait for <clock>\_period/2;

-- end process;

-- Stimulus process

stim\_proc: process

begin

-- hold reset state for 100 ns.

A <= '1';

B <= '0';

wait for 10 ns;

A <= '0';

B <= '1';

wait for 10 ns;

A <= '1';

B <= '1';

wait for 10 ns;

A <= '0';

B <= '0';

wait for 10 ns;

-- insert stimulus here

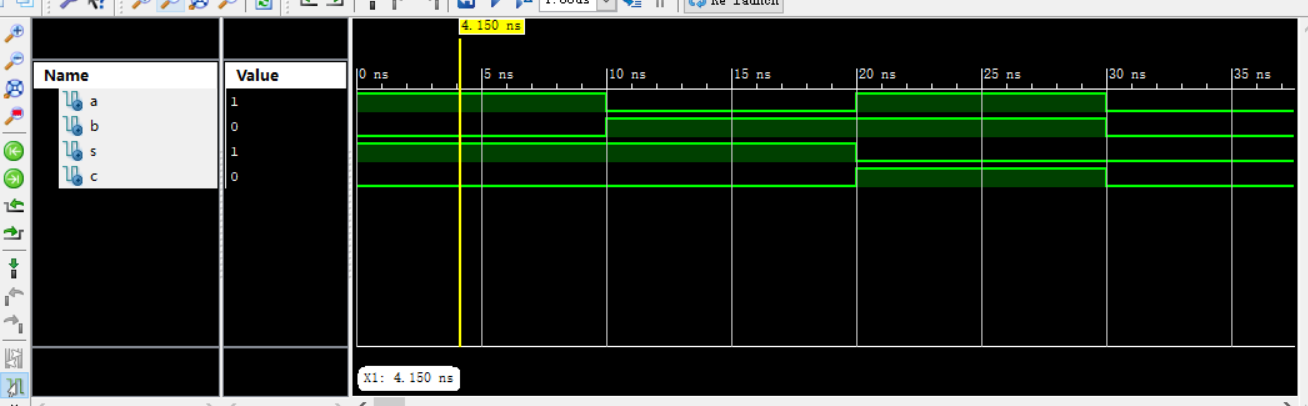
wait;

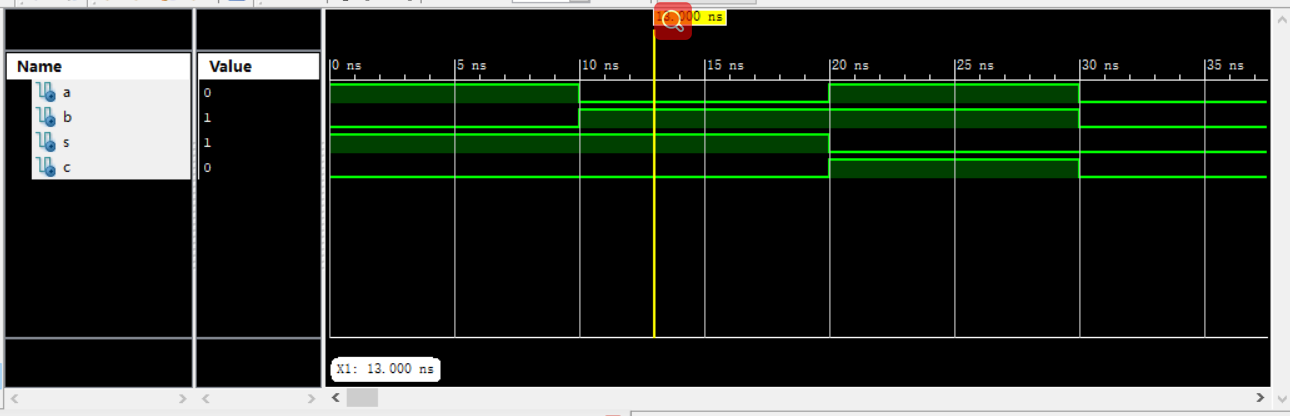
end process;

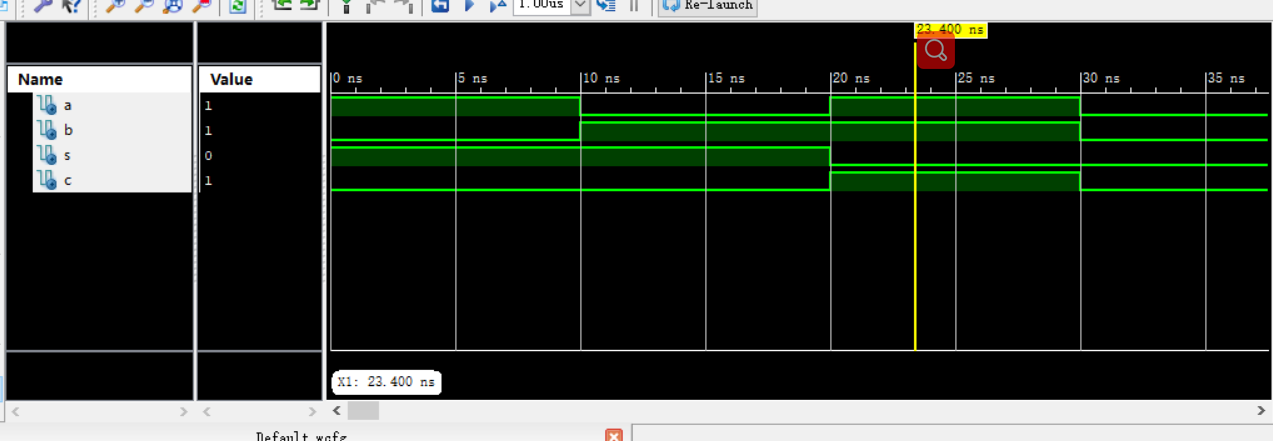
END;

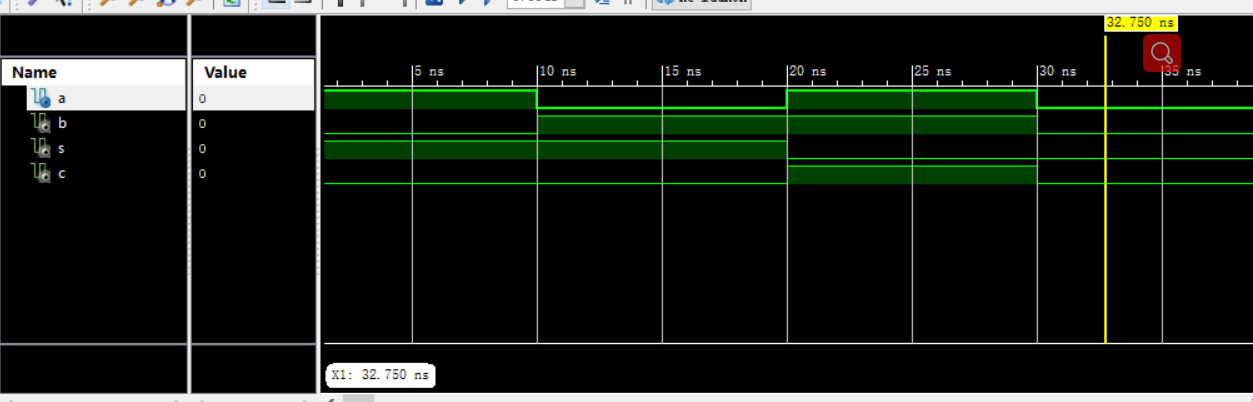
———————————————————————————————————————

And the The simulation waveform is here:









Part 2: Design a one bit full adder using port map and two half adders.

Fulladder code:

library IEEE;

use IEEE.STD\_LOGIC\_1164.ALL;

-- Uncomment the following library declaration if using

-- arithmetic functions with Signed or Unsigned values

--use IEEE.NUMERIC\_STD.ALL;

-- Uncomment the following library declaration if instantiating

-- any Xilinx primitives in this code.

--library UNISIM;

--use UNISIM.VComponents.all;

entity fulladder is

Port ( A : in STD\_LOGIC;

B : in STD\_LOGIC;

Cin : in STD\_LOGIC;

S : out STD\_LOGIC;

Cout : out STD\_LOGIC);

end fulladder;

--architecture Behavioral of fulladder is

architecture structure\_view of fulladder is

component halfadder

Port ( A : in STD\_LOGIC;

B : in STD\_LOGIC;

S : out STD\_LOGIC;

C : out STD\_LOGIC);

end component;

component myxor

Port ( A : in STD\_LOGIC;

B : in STD\_LOGIC;

S : out STD\_LOGIC);

end component;

signal Y0,Z0,Z1 : STD\_LOGIC;

begin

--S <= A xor B xor Cin;

--Cout <= (A and B) or (B and Cin) or (A and Cin);

HA0: halfadder port map(A,B,Y0,Z0);

HA1: halfadder port map(Cin,Y0,S,Z1);

OG: myxor port map(Z0,Z1,Cout);

--end Behavioral;

end structure\_view;

Than this is the test code:(fulladdertest code)

LIBRARY ieee;

USE ieee.std\_logic\_1164.ALL;

-- Uncomment the following library declaration if using

-- arithmetic functions with Signed or Unsigned values

--USE ieee.numeric\_std.ALL;

ENTITY fulladdertest IS

END fulladdertest;

ARCHITECTURE behavior OF fulladdertest IS

-- Component Declaration for the Unit Under Test (UUT)

COMPONENT fulladder

PORT(

A : IN std\_logic;

B : IN std\_logic;

Cin : IN std\_logic;

S : OUT std\_logic;

Cout : OUT std\_logic

);

END COMPONENT;

--Inputs

signal A : std\_logic := '0';

signal B : std\_logic := '0';

signal Cin : std\_logic := '0';

--Outputs

signal S : std\_logic;

signal Cout : std\_logic;

-- No clocks detected in port list. Replace <clock> below with

-- appropriate port name

-- constant <clock>\_period : time := 10 ns;

BEGIN

-- Instantiate the Unit Under Test (UUT)

uut: fulladder PORT MAP (

A => A,

B => B,

Cin => Cin,

S => S,

Cout => Cout

);

-- Clock process definitions

-- <clock>\_process :process

-- begin

-- <clock> <= '0';

-- wait for <clock>\_period/2;

-- <clock> <= '1';

-- wait for <clock>\_period/2;

-- end process;

-- Stimulus process

stim\_proc: process

begin

-- hold reset state for 100 ns.

wait for 10ns;

A <= '0';

B <= '1';

Cin <= '1';

wait for 10 ns;

A <= '1';

B <= '1';

Cin <= '0';

wait for 10 ns;

A <= '0';

B <= '1';

Cin <= '0';

wait for 10 ns;

A <= '1';

B <= '1';

Cin <= '1';

--wait for <clock>\_period\*10;

-- insert stimulus here

wait;

end process;

END;

The simulation waveform：

