Experiment _____ Name: _____ Page No. Date

Experiment-01

Objective

To design and analyze the behaviour of basic logic gates (AND, OR, NOT, XOR and XNOR) using VHDL and verify their truth tables.

Apparatus / Software Required

1.) Model Sim (for simulation) - Intel FPGA edition 2020.1
2.) Computer System

Theory

Logic gates are the building blocks of digital circuits. They operate on binary input (0 and 1) to produce a defined logical output

→ AND Gate : Output is 1 only if all inputs are 1.
→ OR Gate : Output is 1 if any input is 1
→ NOT Gate : Produces the complement (inverts input)
→ NAND Gate : Output is the complement of AND.
→ NOR Gate : Output is the complement of OR.
→ XOR Gate : Output is 1 if inputs are different.
→ XNOR Gate : Output is 1 if inputs are the same.

Teacher's Signature : _____

Experiment _____ Name: _____ Page No. Date

Truth Tables

AND Gates

A	B	Y
0	0	0
0	1	0
1	0	0
1	1	1

OR Gate

A	B	Y
0	0	0
0	1	1
1	0	1
1	1	1

NOT Gate

A	Y
0	1
1	0

XOR ~~NAND~~ Gate

A	B	Y
0	0	0
0	1	1
1	0	1
1	1	0

Teacher's Signature : _____

Experiment _____ Name: _____

Page No.
Date

XNOR Gate

A	B	Y
0	0	1
0	1	0
1	0	0
1	1	1

VHDL Codes

(i) AND Gate

```
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
entity AndGate is
    Port (A, B : in STD_LOGIC;
          Y : out STD_LOGIC;
end AndGate;

architecture Behavioral of AndGate is
begin
    Y <= A AND B
end Behavioral;
```

(ii) OR Gate

```
library IEEE;
```


Teacher's Signature : _____

Experiment _____ Name: _____ Page No. Date

```
use IEEE.STD_LOGIC_1164.ALL;
entity OrGate is
    Port (A, B : in STD_LOGIC;
          Y :  out STD_LOGIC);
end OrGate;

architectural Behavioral of OrGate is
begin
    Y <= A OR B;
end Behavioral;
```

(iii) NOT Gate

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

entity NOTGate is
  Port (A : in STD_LOGIC;
        Y : out STD_LOGIC);
end NotGate;

architecture Behavioral of NotGate is
begin
    Y <= NOT A;
end Behavioral;
```

Teacher's Signature : _____

Experiment _____ Name: _____ Page No. Date

(iv) XOR Gate

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

entity XorGate is
    Port (A, B : in STD_LOGIC;
          Y : out STD_LOGIC);
end XorGate;

architecture Behavioral of XorGate is
begin
    Y <= A XOR B;
end Behavioral;
```

(v) XNOR Gate

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

entity XnorGate is
    Port (A, B : in STD_LOGIC;
          Y : out STD_LOGIC);
end XnorGate;
```

Teacher's Signature : _____

Experiment _____ Name: _____ Page No. Date

```
architecture Behavioral of XnorGate is
begin
    Y <= A XNOR B;
end Behavioral;
```

Procedure

1) Create new project in Modelsim/Quartus.
2) Add a new VHDL file one by one for each gate seperately.
3) Write the code in editor and save it.
4) Compile the file.
5) Write a testbench to apply input combinations.
6) Simulate (vsim <entity_name>_tb).
7) Add signals to waveforms (add wave)
8) Run simulation (run at 100ns)
9) Verify the output with truth table.

Result

The basis gates were successfully designed using VHDL and simulated in Modelsim

Precautions

1) Ensure that the entity name in VHDL code

Teacher's Signature : _____

Experiment _____ Name: _____ Page No. Date

exactly matches the file name in ModelSim.

2) Do not leave the VHDL file blank - always save code before compiling.

Teacher's Signature : _____

Experiment _____ Name: _____ Page No. Date

Experiment-02

Aim

To design and implement 2-input NAND and 2-input NOR gates using CMOS logic and verify their truth tables.

Apparatus / Software Required

1) ModeSim Software - Intel FPGA edition 2020.1.
2) Computer system

Theory

1) NAND Gate

- The NAND gate is the complement of the AND gate.

  Boolean expression:

$$Y = \overline{A \cdot B}$$

- CMOS implementation

  Pull-up n/w : parallel PMOS transistors

  Pull-down n/w : series NMOS transistors

2) NOR Gate

Teacher's Signature : _____

Experiment _____ Name: _____ Page No. Date

- The NOR gate is the complement of the OR gate

Boolean expression: $Y = \overline{A + B}$

CMOS implementation

- Pull-up n/w : series PMOS transistors
- Pull-down n/w : parallel NMOS transistors

Truth Tables

NAND Gate

A	B	Y
0	0	1
0	1	1
1	0	1
1	1	0

NOR Gate

A	B	Y
0	0	1
0	1	0
1	0	0
1	1	0

Teacher's Signature : _____

Experiment _____ Name: _____ Page No. Date

VHDL Codes

NAND Gate

```
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
entity NAND_Gate is
    Port (
        A : in STD_LOGIC;
        B : in STD_LOGIC;
        Y : out STD_LOGIC
    );
end NAND Gate;

architecture Behavioral of NAND_Gate is
begin
    Y <= not (A and B);
end Behavioral;
```

NOR Gate

```
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
entity NOR_Gate is
  Port (
      A : in STD_LOGIC;
      B : in STD_LOGIC;
```


Teacher's Signature : _____

Experiment _____ Name: _____ Page No. Date

```
        Y : out STD_LOGIC
    );
end NOR_Gate;
architecture Behavioral of NOR_Gate is
begin
    Y <= not (A or B);
end Behavioral;
```

Procedure

1) Create new file and write code in editor.
2) Save file and compile it.
3) Now Simulate the file and add ~~wav~~ signals to waveforms.
4) Verify the truth table.

Result

B. The ~~implementation~~ implementation of NAND, & NOR gate is done. and truth table is verified.

Precaution

1) Verify that the entity name matches the file name in VHDL.
2) Save file properly before compiling.

Teacher's Signature : _____

Experiment _____ Name: _____ Page No. Date

Experiment - 03

Aim

To design and implement a Half Adder circuit using CMOS logic and verify its truth table.

Software Req. ModelSim (Intel FPGA edition 2020.1)

Theory

A Half Adder is a combinational circuit that performs the addition of two 1-bit binary numbers.

It has two inputs (A, B) and two outputs (SUM, CARRY).

- SUM = $A \oplus B$ (XOR)
- CARRY = $A \cdot B$ (AND)

In CMOS logic

- XOR is implemented using combinations of PMOS and NMOS transistors.
- AND gate is realized using CMOS NAND + inverter.

Truth Table (Half Adder)

A	B	SUM ($A \oplus B$)	CARRY ($A \cdot B$)
0	0	0	0
0	1	1	0
1	0	1	0
1	1	0	1

Teacher's Signature : _____

Experiment _____ Name: _____

Page No.
Date

VHDL Code

```
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
entity HalfAdder is
  Port (
        A : in STD_LOGIC;
        B : in STD_LOGIC;
      SUM : out STD_LOGIC;
    CARRY : out STD_LOGIC;
        );
end HalfAdder;

architecture Behavioral of HalfAdder is
begin
    SUM <= A XOR B;
  CARRY <= A AND B;
end Behavioral;
```

Observation

The simulated waveform matches the theoretical truth table.

Conclusion

The Half Adder was successfully

Teacher's Signature : _____

Experiment _____ Name: _____

Page No.
Date

designed and implemented using CMOS logic and verified with VHDL simulation

Precautions

1) Save the code before compiling it.
2) Ensure that the entity name in VHDL code exactly matches the file name is ModelSim.

Teacher's Signature : _____

Experiment _____ Name: _____ Page No. Date

Experiment - 04

Aim

To design and implement a Full Adder circuit using CMOS logic and verify its truth table.

Theory

A Full Adder adds three binary inputs (A, B, Cin) and produces two outputs (SUM, Cout).

- SUM = $A \oplus B \oplus Cin$
- Cout = $(A \cdot B) + (B \cdot Cin) + (A \cdot Cin)$

CMOS logic requires XOR (for SUM) and combination of AND/OR gate (for Cout)

Truth Table

A	B	Cin	SUM	Cout
0	0	0	0	0
0	0	1	1	0
0	1	0	1	0
0	1	1	0	1
1	0	0	1	0
1	0	1	0	1
1	1	0	0	1
1	1	1	1	1

Teacher's Signature : _____

Experiment _____ Name: _____ Page No. Date

VHDL Code (Full Adder.vhd)

```
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
entity FullAdder is
    Port (
        A : in STD_LOGIC;
        B : in STD_LOGIC;
        Cin : in STD_LOGIC;
        SUM : out STD_LOGIC;
        Cout : out STD_LOGIC;
    );
end FullAdder;

architecture Behavioral of FullAdder is
begin
    SUM <= A XOR B XOR Cin;
    Cout <= (A AND B) OR (B AND Cin) OR (A AND Cin);
end Behavioral.
```

~~Observation~~ ~~The simulation~~

Procedure

1) Create new project or add new file to the existing project.

Teacher's Signature : _____

Experiment _____ Name: _____ Page No. Date

2) Write the code in the editor, ~~mode~~. and save.
3) Compile the code and then simulate it.
4) Add signals to waveforms.
5) Verify the output with truth table.

Observation

The simulation results match the expected truth table of a Full Adder.

Conclusion

The Full Adder was successfully designed and implemented using CMOS logic and verified using VHDL simulation.

Precaution

1) Save the code before compiling.
2) Ensure that entity name in VHDL code exactly matches the file name is Modelsim.

Teacher's Signature : _____

Experiment _____ Name: _____ Page No. Date

Experiment - 05

Objective

To design, implement and simulate an operational amplifier comparator using VHDL.

Apparatus / Software Req.

1.) ModelSim - Intel FPGA edition 2020.1
2.) Computer system

Theory

A comparator is an application of an operational amplifier (Op-Amp) where the op-amp. compares two voltages.

- If $V_{in+} > V_{in-}$ → output = HIGH (logic 1)
- If $V_{in+} < V_{in-}$ → output = LOW (logic 0)

Truth Table

~~Input ($V_+ > V_-$)~~	V_{in+}	V_{in-}	output
~~0 (False)~~	0	0	0
~~1 (True)~~	1	0	1
	0	1	0
	1	1	0

Teacher's Signature : _____

Experiment _____ Name: _____ Page No. Date

VHDL Code

```
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
entity comparator is
  Port (
        A : in STD_LOGIC;          -- Non-inverting i/p
        B : in STD_LOGIC;          -- inverting i/p
        Y : out STD_LOGIC          output
  );
end comparator;
architecture Behavioral of Comparator is
 begin
      process (A, B)
      begin
          if (A = '1' and B = '0') then
              Y <= '1';            -- A>B
          else
              Y <= '0';            -- A<=B
          end if;
      end process;
end Behavioral;
```

Teacher's Signature : _____

Experiment _____ Name: _____ Page No. Date

Procedure

1) Create new file and write code in editor mode.
2) Compile the code and save it
3) Simulate the file.
4) Add s/g to waveforms and RUN.
5) Verify truth table by inputs.

Result

The op-amp comparator was successfully designed and simulated in ModelSim. The output matched truth table.

Precaution

1) Save code before compiling.
2) Ensure that entity name is same as VHDL code exactly matches the file name.

Teacher's Signature : _____