DAILY ASSESSMENTFORMAT

Date:	**06-06-2020**	**Name:**	**Bindu.N.R**
Course:	**DIGITAL DESIGN USING HDL**	**USN:**	**4AL17EC101**
Topic:	• **FPGA Basics: Architecture, Applications and Uses** • **Verilog HDL Basics by Intel** • **Verilog Test bench code to** • **Verify the design under the test (DUT)**	**Semester & Section:**	**6-B**
Github Repository:	**Bindu-N-R**		

FORENOON SESSION DETAILS
Report – FPGA Basics: Architecture, Applications and Uses: • A basic FPGA architecture (Figure 1) consists of thousands of fundamental elements called configurable logic blocks (CLBs) surrounded by a system of programmable interconnects, called a fabric, that routes signals between CLBs. Input/output (I/O) blocks interface between the FPGA and external devices. • Depending on the manufacturer, the CLB may also be referred to as a logic block (LB), a logic element (LE) or a logic cell (LC). Application: • Many applications rely on the parallel execution of identical operations; the ability to configure the FPGA's CLBs into hundreds or thousands of identical processing blocks has applications in image processing, artificial intelligence (AI), data center hardware accelerators, enterprise networking and automotive advanced driver assistance systems (ADAS). • Many of these application areas are changing very quickly as requirements evolve and new protocols and standards are adopted. FPGAs enable manufacturers to implement systems that can be updated when necessary. • A good example of FPGA use is high-speed search: Microsoft is using FPGAs in its data

centers to run Bing search algorithms. The FPGA can change to support new algorithms as they are created. If needs change, the design can be repurposed to run simulation or modeling routines in an HPC application. This flexibility is difficult or impossible to achieve with an ASIC.

• Other FPGA uses include aerospace and defense, medical electronics, digital television, consumer electronics, industrial motor control, scientific instruments, cybersecurity systems and wireless communications.

Verilog HDL Basics by Intel:

- Verilog is a HARDWARE DESCRIPTION LANGUAGE (HDL). It is a language used for describing a digital system like a network switch or a microprocessor or a memory or a flip−flop.
- It means, by using a HDL we can describe any digital hardware at any level. Designs, which are described in HDL are independent of technology, very easy for designing and debugging, and are normally more useful than schematics, particularly for large circuits.
- Behavioral level
- Register-transfer level
- Gate level
- Lexical Tokens
- Numbers
- Identifiers
- Operators
- Data Types
- Operators
- Operands
- Modules

Verilog Test bench code to verify the design under test (DUT):

TASK:
Implement a 4:1 MUX and write the test bench code to verify the module:

Multiplexer (4:1)

Verilog design:

```
module mux41(
input i0,i1,i2,i3,sel0,sel1, output reg y); always @(*) begin

case ({sel0,sel1}) 2'b00 : y = i0; 2'b01 : y = i1; 2'b10 : y = i2; 2'b11 : y = i3;
end case

end


endmodule
```

Test Bench:

```
module tb_mux41;

reg I0,I1,I2,I3,SEL0,SEL1; wire Y;

mux41 MUX (.i0(I0),.i1(I1),.i2(I2),.i3(I3),.sel0(SEL0),.sel1(SEL1),.y(Y));

initial begin I0 =1'b0; I1= 1'b0; I2 =1'b0;


I3 =1'b0; SEL0 =1'b0; SEL1 =1'b0; #45 $finish;

end

always #2 I0 = ~I0; always #4 I1 =~I1; always #6 I2 =~I1; always #8 I3 =~I1; always #3 SEL0 = ~SEL0;
always #3 SEL1 = ~SEL1;

always @(Y)

$display( "time =%0t INPUT VALUES: \t I0=%b I1 =%b I2 =%b I3 =%b SEL0 =%b
```

```
SEL1 =%b \t output value Y =%b ",$time,I0,I1,I2,I3,SEL0,SEL1,Y);

endmodule
```

Output:

time=0 INPUT VALUES: output value Y=0 time=2
INPUT VALUES: output value Y=1

time=3 INPUT VALUES: output value Y=0 time=6
INPUT VALUES: output value Y=1

time=8 INPUT VALUES: output value Y=0 time=14 INPUT
VALUES: output value Y=1

time=15 INPUT VALUES: output value Y=0

I0=0 I1=0 I2=0 I3=0 SEL0=0 SEL1=0 I0=1 I1=0 I2=0 I3=0 SEL0=0 SEL1=0
I0=1 I1=0 I2=0 I3=0 SEL0=1 SEL1=1 I0=1 I1=1 I2=0 I3=0 SEL0=0 SEL1=0
I0=0 I1=0 I2=0 I3=0 SEL0=0 SEL1=0

I0=1 I1 =1 I2 =1 I3 =0 SEL0 =0 SEL1 =0 I0=1 I1 =1 I2 =1 I3 =0 SEL0 =1 SEL1 =1