Table of Contents

Note : Remove “Table of Content” before including in CP Book

17ECL58 : Hardware Description Language Lab

A. LABORATORY INFORMATION

1. Lab Overview

Degree:	BE	*Program:*	EC
Year / Semester :	3/5	*Academic Year:*	2019-20
Course Title:	Hardware Description Language Lab	*Course Code:*	15ECL58
Credit / L-T-P:	4 / 0-1-2	*SEE Duration:*	180 Minutes
Total Contact Hours:	42 Hrs	*SEE Marks:*	100 Marks
CIA Marks:	40	*Assignment*	
Course Plan Author:	Narasimha Murthy R	*Sign*	Dt : 10/08/2019
Checked By:		*Sign*	Dt : 10/08/2019

2. Lab Content

Unit	Title of the Experiments	Lab Hours	Concept	Blooms Level
1	Write Verilog code to realize all the logic gates	3	Basic gates functionality	L4 Analyze
2	Write a Verilog program for the following combinational designs a. 2 to 4 decoder b. 8 to 3 (encoder without priority & with priority) c. 8 to 1 multiplexer. d. 4 bit binary to gray converter e. Multiplexer, de-multiplexer, comparator.	6	Combinational logic circuits functionality	L4
3	Write a Verilog code to model 32 bit ALU	3	ALU functionality	L4

4	Develop the Verilog code for the following flip-flops: SR, D, JK and T.	3	Flipflops functionality	L4
5	Design a 4 bit binary, BCD counters (Synchronous reset and Asynchronous reset) and “any sequence” counters, using Verilog code.	3	Counters functionality	L4
6	Write HDL code to display messages on an alpha numeric LCD display	3	LCD Display implementation	L4
7	Write HDL code to interface Hex key pad and display the key code on seven segment display.	3	Hex Keypad implementation	L4
8	Write HDL code to control speed, direction of DC and Stepper motor.	3	Motor implementation	L4
9	Write HDL code to accept Analog signal, Temperature sensor and display the data on LCD or Seven segment display.	3	Temperature sensor	L4
10	Write HDL code to generate different waveforms (Sine, Square, Triangle, Ramp etc.,) using DAC – change the frequency.	3	ADC implementation	L4
11	Write HDL code to simulate Elevator operation.	3	Elevator operation.	L4

3. Lab Material

Unit	Details	Available
1	Text books	
		In Lib
2	Reference books	
	Lab manual prepared by Department of E & C Engg, SKIT.	In dept
	HDL Programming (VHDL and Verilog)- Nazeih M.Botros- Dreamtech Press (Available through John Wiley – India and Thomson Learning), 2006 Edition	
3	Others (Web, Video, Simulation, Notes etc.)	

EC

Prepared by Checked by

Approved

		Not Available

4. Lab Prerequisites:

-	-	Base Course:		-	-
SNo	Course Code	Course Name	Topic / Description	Sem	Remarks
1	17ECL38	Digital Electronics	Knowledge on basic gates, Combinational and Sequential logic circuits	3	

Note: If prerequisites are not taught earlier, GAP in curriculum needs to be addressed. Include in Remarks and implement in B.5.

5. General Instructions

SNo	Instructions	Remarks
1	Observation book and Lab record are compulsory.	
2	Students should report to the concerned lab as per the time table.	
3	After completion of the program, certification of the concerned staff in-charge in the observation book is necessary.	
4	Student should bring a notebook of 100 pages and should enter the readings /observations into the notebook while performing the experiment.	
5	The record of observations along with the detailed experimental procedure of the experiment in the Immediate last session should be submitted and certified staff member in-charge.	
6	Should attempt all problems / assignments given in the list session wise.	
7	It is responsibility to create a separate directory to store all the programs, so that nobody else can read or copy.	
8	When the experiment is completed, should disconnect the setup made by them, and should return all the components/instruments taken for the purpose.	
9	Any damage of the equipment or burn-out components will be viewed seriously either by putting penalty or by dismissing the total group of students from the lab for the semester/year	
10	Completed lab assignments should be submitted in the form of a Lab Record in which you have to write the algorithm, program code along with comments and output for various inputs given	

6. Lab Specific Instructions

SNo	Specific Instructions	Remarks
1	Turn on the computer.	

EC

Prepared by Checked by

Approved

2	Double click on Xilinx ISE 12.1 Project Navigator icon.	
3	Select new project in file menu.	
4	Enter the project name and location as shown below and hit Next.	
5	Select the Family, Device, Package and speed as per the requirements and hit Next.	
6	Create a new source by using new source icon or right click on the device/project folder to create new source.	
7	Select the verilog module and enter the file name in New Source Wizard window and hit Next.	
8	Enter the module name – dataflow/behavioral/structural, port name and select the direction. This will create .v source file. Hit Next and finish the initial project creation.	
9	Write complete VHDL/Verilog code for implementation and save.	
10	Click on implementation and check for syntax using “Check syntax” option under synthesize tab. If any error, edit and correct VHDL/Verilog code and repeat check syntax until zero errors.	
11	Double click on ISIM simulator by selecting simulation mode to complete the functional simulation of your design.	
12	Click on user constraints and select pre synthesis/post synthesis for assigning the ports, select the ports and save. It will generate .ucf file to source file.	
13	Click on Implement design for checking Place, Route and Map.	
14	Click generate programming file to generate the .bit file for loading into FPGA kit.	
15	Select the COM port and load the bit file to FPGA kit and check the results. Note down the results in observation book.	

B. OBE PARAMETERS

1. Lab / Course Outcomes

#	COs	Teach. Hours	Concept	Instr Method	Assessment Method	Blooms' Level
17ECL58.1	Create and verify functionality of various gates at the different level of abstractions.	3	Basic gates functionality	Tutorial / Demonstratipon/ Practical	CIA	L2,L3, L4,L5
17ECL58.2	Design, verify and implement the functionality of various Combinational logic circuits.	6	Combinational logic circuits functionality	Tutorial / Demonstratipon/ Practical	CIA	L2,L3, L4,L5
17ECL58.3	Design and Analyze the functionality of	3	ALU	Tutorial /	CIA	L2,L3,

	32 bit ALU.		functionality	Demonstratipon/ Practical		L4,L5
17ECL58.4	Design, verify and implement the functionality of Flipflops.	3	Flipflops functionality	Tutorial / Demonstratipon/ Practical	CIA	L2,L3, L4,L5
17ECL58.5	Design, verify and implement the functionality of counters.	3	Counters functionality	Tutorial / Demonstratipon/ Practical	CIA	L2,L3, L4,L5
17ECL58.6	Design the digital system for Interfacing FPGA to alpha numeric LCD display.	3	LCD Display implementation	Tutorial / Demonstratipon/ Practical	CIA	L2,L3, L4,L5
17ECL58.7	Design the digital system for Interfacing FPGA with Hex Keypad.	3	Keypad implementation	Tutorial / Demonstratipon/ Practical	CIA	L2,L3, L4,L5
17ECL58.8	Design the digital system for Controlling the speed and direction of stepper motor using HDL	3	Motor implementation	Tutorial / Demonstratipon/ Practical	CIA	L2,L3, L4,L5
17ECL58.9	Design the digital system for Interfacing FPGA with temperature sensor	3	Temperature sensor	Tutorial / Demonstratipon/ Practical	CIA	L2,L3, L4,L5
17ECL58.10	Study of DAC and generation of different signals using HDL .	3	ADC implementation	Tutorial / Demonstratipon/ Practical	CIA	L2,L3, L4,L5
17ECL58.11	Design Elevator system.	3	Elevator operation.	Tutorial / Demonstratipon/ Practical	CIA	L2,L3, L4,L5
-	**Total**	**42**	-	-	-	-

Note: Identify a max of 2 Concepts per unit. Write 1 CO per concept.

2. Lab Applications

SNo	Application Area	CO	Level
1	Design of adder and multiplier circuits.		L3
2	Design of PLDs, PLAs, PALs.		L3

EC

Prepared by Checked by

Approved

3	Design of 32 bit Microprocessor and Microcontrollers.		L3
4	Design of counters, shift registers.		L3
5	Design of memory elements		L3
6	Design of digital systems.		L3

Note: Write 1 or 2 applications per CO.

3. Articulation Matrix

(CO – PO MAPPING)

-	Course Outcomes	Program Outcomes												
#	COs	PO 1	PO 2	PO 3	PO 4	PO 5	PO 6	PO 7	PO 8	PO 9	PO 10	PO 11	PO 12	Level
17ECL58.1	Create and verify functionality of various gates at the different level of abstractions.	2	2	3		3				3			3	L4
17ECL58.2	Design, verify and implement the functionality of various Combinational logic circuits.	2	2	3		3				3			3	L4
17ECL58.3	Design and Analyze the functionality of 32 bit ALU.	2	2	3		3				3			3	L4
17ECL58.4	Design, verify and implement the functionality of Flipflops.	2	2	3		3				3			3	L4
17ECL58.5	Design, verify and implement the functionality of counters.	2	2	3		3				3			3	L4
17ECL58.6	Design the digital system for Interfacing FPGA to alpha numeric LCD display.	2	2	3		3				3			3	L4
17ECL58.7	Design the digital system for Interfacing FPGA with Hex Keypad.	2	2	3		3				3			3	L4
17ECL58.8	Design the digital system for Controlling the speed and direction of stepper motor using HDL	2	2	3		3				3			3	L4
17ECL58.9	Design the digital system for Interfacing FPGA with temperature sensor	2	2	3		3				3			3	L4
17ECL58.10	Study of DAC and generation of different signals using HDL .	2	2	3		3				3			3	L4
17ECL58.11	Design Elevator system.	2	2	3		3				3			3	L4

15ECL58.	Average													

Note: Mention the mapping strength as 1, 2, or 3

4. Mapping Justification

Mapping		Mapping Level	Justification
CO	**PO**	**-**	**-**
CO1	PO1	L4	The basic engineering knowledge is applied for the basic digital integrated circuit coding.
CO1	PO2	L4	Performing experiment allows the easy analysis of problems.
CO1	PO3	L4	Designing a digital system to meet the specific needs within the realistic constraints can be done.
CO1	PO5	L4	Modern tools are used for designing and analysis of systems.
CO1	PO9	L4	Experiments are done in teams to develop team work.
CO1	PO12	L4	Practical knowledge inculcates inquisitiveness towards continuous learning.
CO2	PO1	L4	The basic engineering knowledge is applied for the basic digital integrated circuit coding.
CO2	PO2	L4	Performing experiment allows the easy analysis of problems.
CO2	PO3	L4	Designing a digital system to meet the specific needs within the realistic constraints can be done.
CO2	PO5	L4	Modern tools are used for designing and analysis of systems.
CO2	PO9	L4	Experiments are done in teams to develop team work.
CO2	PO12	L4	Practical knowledge inculcates inquisitiveness towards continuous learning.
CO3	PO1	L4	The basic engineering knowledge is applied for the basic digital integrated circuit coding.
CO3	PO2	L4	Performing experiment allows the easy analysis of problems.
CO3	PO3	L4	Designing a digital system to meet the specific needs within the realistic constraints can be done.
CO3	PO5	L4	Modern tools are used for designing and analysis of systems.
CO3	PO9	L4	Experiments are done in teams to develop team work.
CO3	PO12	L4	Practical knowledge inculcates inquisitiveness towards continuous learning.
CO4	PO1	L4	The basic engineering knowledge is applied for the basic digital integrated circuit coding.
CO4	PO2	L4	Performing experiment allows the easy analysis of problems.
CO4	PO3	L4	Designing a digital system to meet the specific needs within the realistic constraints can be done.
CO4	PO5	L4	Modern tools are used for designing and analysis of systems.

CO4	PO9	L4	Experiments are done in teams to develop team work.
CO4	PO12	L4	Practical knowledge inculcates inquisitiveness towards continuous learning.
CO5	PO1	L4	The basic engineering knowledge is applied for the basic digital integrated circuit coding.
CO5	PO2	L4	Performing experiment allows the easy analysis of problems.
CO5	PO3	L4	Designing a digital system to meet the specific needs within the realistic constraints can be done.
CO5	PO5	L4	Modern tools are used for designing and analysis of systems.
CO5	PO9	L4	Experiments are done in teams to develop team work.
CO5	PO12	L4	Practical knowledge inculcates inquisitiveness towards continuous learning.
CO6	PO1	L4	The basic engineering knowledge is applied for the basic digital integrated circuit coding.
CO6	PO2	L4	Performing experiment allows the easy analysis of problems.
CO6	PO3	L4	Designing a digital system to meet the specific needs within the realistic constraints can be done.
CO6	PO5	L4	Modern tools are used for designing and analysis of systems.
CO6	PO9	L4	Experiments are done in teams to develop team work.
CO6	PO12	L4	Practical knowledge inculcates inquisitiveness towards continuous learning.
CO7	PO1	L4	The basic engineering knowledge is applied for the basic digital integrated circuit coding.
CO7	PO2	L4	Performing experiment allows the easy analysis of problems.
CO7	PO3	L4	Designing a digital system to meet the specific needs within the realistic constraints can be done.
CO7	PO5	L4	Modern tools are used for designing and analysis of systems.
CO7	PO9	L4	Experiments are done in teams to develop team work.
CO7	PO12	L4	Practical knowledge inculcates inquisitiveness towards continuous learning.
CO8	PO1	L4	The basic engineering knowledge is applied for the basic digital integrated circuit coding.
CO8	PO2	L4	Performing experiment allows the easy analysis of problems.
CO8	PO3	L4	Designing a digital system to meet the specific needs within the realistic constraints can be done.
CO8	PO5	L4	Modern tools are used for designing and analysis of systems.
CO8	PO9	L4	Experiments are done in teams to develop team work.
CO8	PO12	L4	Practical knowledge inculcates inquisitiveness towards continuous learning.
CO9	PO1	L4	The basic engineering knowledge is applied for the basic digital integrated circuit coding.

CO9	PO2	L4	Performing experiment allows the easy analysis of problems.
CO9	PO3	L4	Designing a digital system to meet the specific needs within the realistic constraints can be done.
CO9	PO5	L4	Modern tools are used for designing and analysis of systems.
CO9	PO9	L4	Experiments are done in teams to develop team work.
CO9	PO12	L4	Practical knowledge inculcates inquisitiveness towards continuous learning.
CO10	PO1	L4	The basic engineering knowledge is applied for the basic digital integrated circuit coding.
CO10	PO2	L4	Performing experiment allows the easy analysis of problems.
CO10	PO3	L4	Designing a digital system to meet the specific needs within the realistic constraints can be done.
CO10	PO5	L4	Modern tools are used for designing and analysis of systems.
CO10	PO9	L4	Experiments are done in teams to develop team work.
CO10	PO12	L4	Practical knowledge inculcates inquisitiveness towards continuous learning.
CO11	PO1	L4	The basic engineering knowledge is applied for the basic digital integrated circuit coding.
CO11	PO2	L4	Performing experiment allows the easy analysis of problems.
CO11	PO3	L4	Designing a digital system to meet the specific needs within the realistic constraints can be done.
CO11	PO5	L4	Modern tools are used for designing and analysis of systems.
CO11	PO9	L4	Experiments are done in teams to develop team work.
CO11	PO12	L4	Practical knowledge inculcates inquisitiveness towards continuous learning.

Note: Write justification for each CO-PO mapping.

5. Curricular Gap and Content

SNo	Gap Topic	Actions Planned	Schedule Planned	Resources Person	PO Mapping
1					
2					
3					
4					
5					

Note: Write Gap topics from A.4 and add others also.

6. Content Beyond Syllabus

SNo	Gap Topic	Actions Planned	Schedule Planned	Resources Person	PO Mapping

EC

Prepared by Checked by

Approved

1					
2					
3					
4					
5					
6					
7					
8					
9					
10					
11					
12					
13					
14					
15					

Note: Anything not covered above is included here.

C. COURSE ASSESSMENT

1. Course Coverage

Unit	Title	Teaching Hours	No. of question in Exam						CO	Levels
			CIA-1	CIA-2	CIA-3	Asg-1	Asg-2	Asg-3	SEE		
1	Write Verilog code to realize all the logic gates	03	1	–	–	–	–	–	1	CO1	L2
2	Write a Verilog program for the following combinational designs a. 2 to 4 decoder b. 8 to 3 (encoder without priority & with priority) c. 8 to 1 multiplexer. d. 4 bit binary to gray converter e. Multiplexer, de-multiplexer, comparator.	03	1	–	–	–	–	–	1	CO2	L3
3	Write a Verilog code to model 32 bit ALU	03	1	–	–	–	–	–	1	CO3	L3
4	Develop the Verilog code for the following flip-flops: SR, D, JK and T.	03	1	–	–	–	–	–	1	CO4	L3
5	Design a 4 bit binary, BCD	03	1	–	–	–	–	–	1	CO5	L4

	counters (Synchronous reset and Asynchronous reset) and "any sequence" counters, using Verilog code.										
6	Write HDL code to display messages on an alpha numeric LCD display	03	1	-	-	-	-	-	1	CO6	L4
7	Write HDL code to interface Hex key pad and display the key code on seven segment display.	03	1	-	-	-	-	-	1	CO7	L4
8	Write HDL code to control speed, direction of DC and Stepper motor.	03	-	1	-	-	-	-	1	CO8	L4
9	Write HDL code to accept Analog signal, Temperature sensor and display the data on LCD or Seven segment display.	03	-	1	-	-	-	-	1	CO9	L4
10	Write HDL code to generate different waveforms (Sine, Square, Triangle, Ramp etc.,) using DAC - change the frequency.	03	-	1	-	-	-	-	1	CO10	L4
11	Write HDL code to simulate Elevator operation.	03	-	1	-	-	-	-	1	CO11	L4
-	**Total**	**42**	**7**	**8**	**5**	**5**	**5**	**5**	**20**	**-**	**-**

Note: Write CO based on the theory course.

2. Continuous Internal Assessment (CIA)

Evaluation	Weightage in Marks	CO	Levels
CIA Exam – 1	30	CO1, CO2, CO3, CO4	L23, L3
CIA Exam – 2	30	CO5, CO6, CO7,CO8	L1, L2, L3 . .
CIA Exam – 3	30	CO9,CO10,CO11	L1, L2, L3 . .
Assignment – 1	05	CO1, CO2, CO3, CO4	L2, L3, L4 ...
Assignment – 2	05	CO5, CO6, CO7,CO8	L1, L2, L3 ...
Assignment – 3	05	CO9,CO10,CO11	L1, L2, L3 ...
Seminar – 1	05	CO1, CO2, CO3, CO4	L2, L3, L4 . ..
Seminar – 2	05	CO5, CO6, CO7,CO8	L2, L3, L4 . ..
Seminar – 3	05	CO9,CO10,CO11	L2, L3, L4 . ..

Other Activities – define – Slip test		CO1 to CO11	L2, L3, L4 . ..
Final CIA Marks	**40**	**-**	**-**

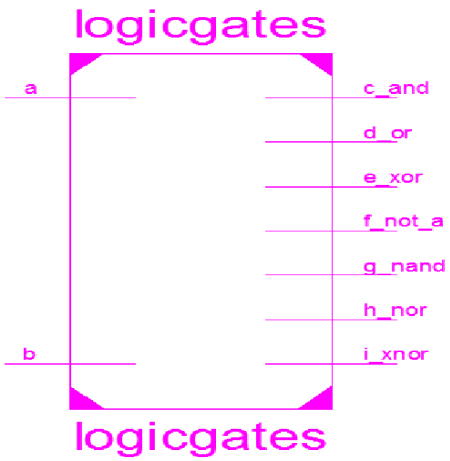
–

SNo	Description	Marks
1	Observation and Weekly Laboratory Activities	05 Marks
2	Record Writing	10 Marks for each Expt
3	Internal Exam Assessment	20 Marks
4	Internal Assessment	5 Marks
5	SEE	600 Marks
-	**Total**	**100 Marks**

D. EXPERIMENTS

Experiment 01 : Write Verilog code to realize all the logic gates

–	Experiment No.:	1	Marks		Date Planned		Date Conducted	
1	Title	Write verilog code to realize all the logic gates						
2	Course Outcomes	Create and verify functionality of various gates at the different level of abstractions.						
3	Aim	Write Verilog code to realize all the logic gates in behavioural, dataflow and gate level modeling.						
4	Material / Equipment Required	Lab Manual						
5	Theory, Formula, Principle, Concept	Basic structure of programming in verilog, Logical expression and Truth table for all the logic gates.						



6	Procedure, Program, Activity, Algorithm, Pseudo Code	• step 1: start • step 2: write programming • step 3: save the program • step 4: check syntax • step 5:if error then correct the errors • step 6:simulate the design • step 7: FPGA implementation • step 8: stop

7	Block, Circuit, Model Diagram, Reaction Equation, Expected Graph	**Verilog code**
		module logicgates(input a,b,
		outputc_and,d_or,e_xor,f_not_a,g_nand, h_nor, i_xnor);
		assign c_and= a & b; assign d_or= a|b;
		assign e_xor= a^b; assign f_not_a= ~a;
		assign g_nand= ~(a&b);
		assign h_nor= ~(a|b);

8	Observation Table, Look-up Table, Output	**Truth Table** • Truth table verification
9	Sample Calculations	• –

Input		Output						
a	b	c_and	d_or	e_xor	f_not_a	g_nand	h_nor	i_xnor
0	0	0	0	0	1	1	1	1
0	1	0	1	1	1	1	0	0
1	0	0	1	1	0	1	0	0
1	1	1	1	0	0	0	0	1

10	Graphs, Outputs	
11	Results & Analysis	• The logic Gates have been realized and simulated using HDL codes.
12	Application Areas	• To write the verilog program
13	Remarks	
14	Faculty Signature with Date	

Experiment 02 : Write a Verilog program for the combinational designs

-	**Experiment No.:**	2	**Marks**		**Date Planned**		**Date Conducted**	
1	Title	Write a Verilog program for the combinational designs						
2	Course Outcomes	Design, verify and implement the functionality of various Combinational logic circuits.						
3	Aim	Write a Verilog program for the following combinational designs a. 2 to 4 decoder						
4	Material / Equipment Required	Lab Manual						
5	Theory, Formula, Principle, Concept	**Boolean Expression** $y0 = \bar{a}.\overline{b}.en$ $y1 = \bar{a}.b.en$ $y2 = a.\overline{b}.en$ $y3 = a.b.en$						
6	Procedure, Program, Activity, Algorithm, Pseudo Code	• step 1: start • step 2: write programming • step 3: save the program • step 4: check syntax • step 5:if error then correct the errors						

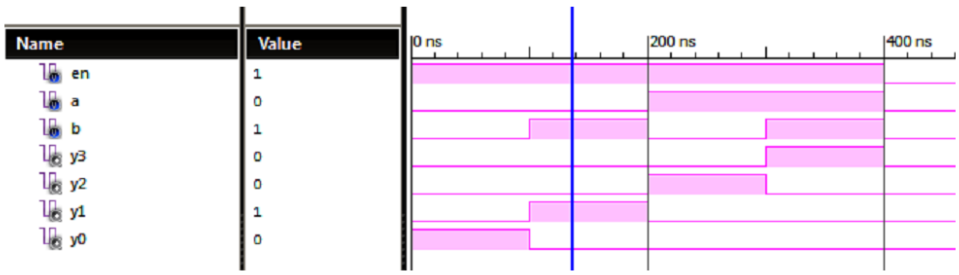
EC

Prepared by Checked by

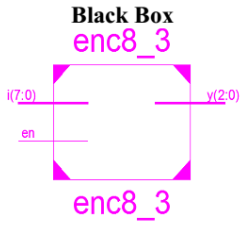
Approved

		• step 6:simulate the design

7	Block, Circuit, Model Diagram, Reaction Equation, Expected Graph	**VERILOG CODE**
		module dec2_4 (a,b,en,y0,y1,y2,y3)
		input a, b, en;
		output y0,y1,y2,y3;
		assign y0= (~a) & (~b) & en;
		assign y1= (~a) & b & en;
		assign y2= a & (~ b) & en;
		assign y3= a & b & en;
		endmodule

8	Observation Table, Look-up Table, Output	**Truth Table** Input: en, a, b; Output: y3, y2, y1, y0 1 0 0 0 0 0 1 1 0 1 0 0 1 0 1 1 0 0 1 0 0 1 1 1 1 0 0 0 0 X X 0 0 0 0 • Truth table verification
9	Sample Calculations	
10	Graphs, Outputs	**Simulation Waveforms** 

11	Results & Analysis	
12	Application Areas	Design of top level digital circuits.
13	Remarks	
14	Faculty Signature with Date	

-	**Experiment No.:**	2	**Marks**		**Date Planned**		**Date Conducted**	
1	Title	Write a Verilog program for the combinational designs						
2	Course Outcomes	Design, verify and implement the functionality of various Combinational logic circuits.						
3	Aim	Write a Verilog program for the following combinational designs b. 8 to 3 (encoder without priority)						
4	Material / Equipment Required	Lab Manual						
5	Theory, Formula, Principle, Concept							
6	Procedure, Program, Activity, Algorithm, Pseudo Code	• step 1: start • step 2: write programming • step 3: save the program • step 4: check syntax • step 5:if error then correct the errors • step 6:simulate the design						
7	Block, Circuit,							

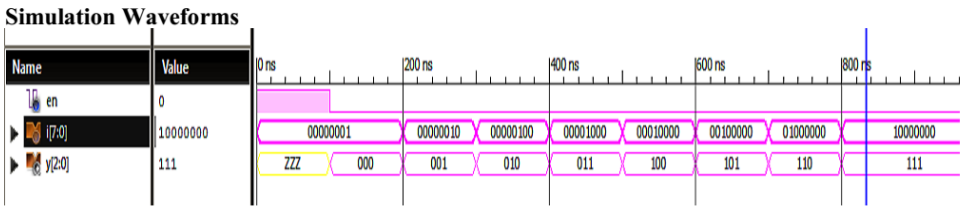
VERILOG CODE

```
module enc8_3(i,en,y);
input [7:0]i;
input en;
output reg [2:0]y;
always @(i or en)
begin
if(en==1)
y<=3'bzzz;
else
case(i)
8'b00000001:y<=3'b000;
8'b00000010:y<=3'b001;
8'b00000100:y<=3'b010;
8'b00001000:y<=3'b011;
8'b00010000:y<=3'b100;
8'b00100000:y<=3'b101;
8'b01000000:y<=3'b110;
8'b10000000:y<=3'b111;
default:y<=3'bxxx;
endcase
end
endmodule
```

	Model Diagram, Reaction Equation, Expected Graph	
8	Observation Table, Look-up Table, Output	• Truth table verification

Truth Table

Input								Output		
en	i[7]	i[6]	i[5]	i[4]	i[3]	i[2]	i[1]	i[0]	y[2]	y[1]	y[0]
1	X	X	X	X	X	X	X	X	Z	Z	Z
0	0	0	0	0	0	0	0	1	0	0	0
0	0	0	0	0	0	0	1	0	0	0	1
0	0	0	0	0	0	1	0	0	0	1	0
0	0	0	0	0	1	0	0	0	0	1	1
0	0	0	0	1	0	0	0	0	1	0	0
0	0	0	1	0	0	0	0	0	1	0	1
0	0	1	0	0	0	0	0	0	1	1	0
0	1	0	0	0	0	0	0	0	1	1	1

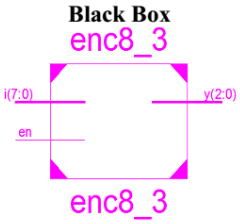
9	Sample Calculations	
10	Graphs, Outputs	
11	Results & Analysis	
12	Application Areas	Design of top level digital circuits.
13	Remarks	
14	Faculty Signature with Date	

-	**Experiment No.:**	2	**Marks**		**Date Planned**		**Date Conducted**	
1	Title	Write a Verilog program for the combinational designs						

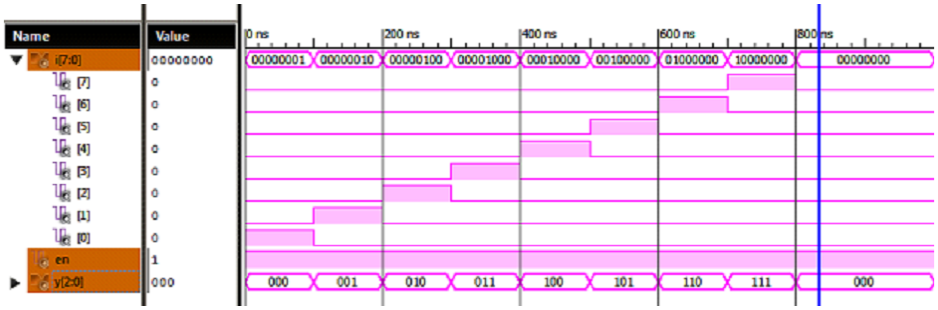
EC

Prepared by Checked by

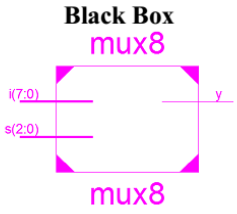
Approved

2	Course Outcomes	Design, verify and implement the functionality of various Combinational logic circuits.
3	Aim	Write a Verilog program for the following combinational designs b. 8 to 3 (encoder without priority)
4	Material / Equipment Required	Lab Manual
5	Theory, Formula, Principle, Concept	
6	Procedure, Program, Activity, Algorithm, Pseudo Code	• step 1: start • step 2: write programming • step 3: save the program • step 4: check syntax • step 5:if error then correct the errors • step 6:simulate the design • step 7: FPGA implementation • step 8: stop

7	Block, Circuit, Model Diagram, Reaction Equation, Expected Graph	**VERILOG CODE**
		module prienc(
		input [7:0] i,
		input en,
		output reg [2:0] y);
		always @ (en,i)
		begin
		if(en==1)
		begin
		if (i[7]==1) y=3'b111;
		else if (i[6]==1) y=3'b110;
		else if (i[5]==1) y=3'b101;
		else if (i[4]==1) y=3'b100;
		else if (i[3]==1) y=3'b011;
		else if (i[2]==1) y=3'b010;
		else if (i[1]==1) y=3'b001;
		else if (i[0]==1) y=3'b000;
		else y=3'b000;
		end
		else
		y=3'b000;
		end
		endmodule

8	Observation Table, Look-up Table, Output	• Truth table verification **Truth Table** **Input** / **Output** en i_7 i_6 i_5 i_4 i_3 i_2 i_1 i_0 y_2 y_1 y_0 0 X X X X X X X X 0 0 0 1 0 0 0 0 0 0 0 1 0 0 0 1 0 0 0 0 0 0 1 X 0 0 1 1 0 0 0 0 0 1 X X 0 1 0 1 0 0 0 0 1 X X X 0 1 1 1 0 0 0 1 X X X X 1 0 0 1 0 0 1 X X X X X 1 0 1 1 0 1 X X X X X X 1 1 0 1 1 X X X X X X X 1 1 1
9	Sample Calculations	
10	Graphs, Outputs	**Simulation Waveforms** 
11	Results & Analysis	
12	Application Areas	Design of top level digital circuits.
13	Remarks	
14	Faculty Signature with Date	

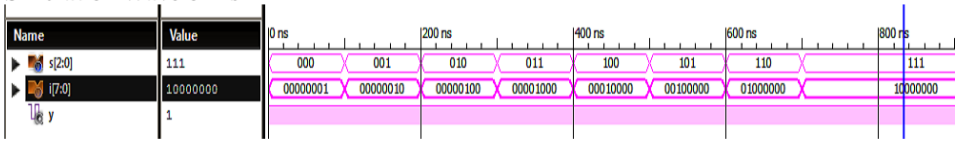
-	**Experiment No.:**	2	**Marks**		**Date Planned**		**Date Conducted**	
1	Title	Write a Verilog program for the combinational designs						
2	Course Outcomes	Design, verify and implement the functionality of various Combinational						

		logic circuits.
3	Aim	Write a Verilog program for the following combinational designs c. 8 to 1 multiplexer.
4	Material / Equipment Required	La
5	Theory, Formula, Principle, Concept	 **Boolean Expression** $y = \overline{s[2].s[1].}\ \overline{s[0].}\ i[0] + \overline{s[1].}\ \overline{s[1].}\ s[0].i[1] +$ $\overline{s[2].}s[1].\ \overline{s[0].}\ i[2] + \overline{s[2].}s[1].s[0].i[3] +$ $s[2].\overline{s[1].}\ \overline{s[0].}\ i[4] + s[2].\overline{s[1].}\ s[0].i[5] +$ $s[2].s[1].\ \overline{s[0].}\ i[6] + s[2].s[1].s[0].i[7]$

6	Procedure, Program, Activity, Algorithm, Pseudo Code	• step 1: start
		• step 2: write programming
		• step 3: save the program
		• step 4: check syntax
		• step 5:if error then correct the errors
		gn
		ation
		`module mux8(i,s,y);`
		`input [7:0]i;`
		`input [2:0]s;`
		`output reg y;`
		`always @(i,s)`
		`begin`
		`case(s)`
		`3'b000:y=i[0];`
		`3'b001:y=i[1];`
		`3'b010:y=i[2];`
		`3'b011:y=i[3];`
		`3'b100:y=i[4];`
		`3'b101:y=i[5];`
		`3'b110:y=i[6];`
		`3'b111:y=i[7];`
		`default:y=3'b000;`
		`endcase`
		`end`
		`endmodule`

7	Block, Circuit, Model Diagram, Reaction Equation, Expected Graph	
8	Observation Table, Look-up Table, Output	• Truth table verification **Truth Table**

Input			Output
sel[2]	sel[1]	sel[0])	Y
0	0	0	i[0]
0	0	1	i[1]
0	1	0	i[2]
0	1	1	i[3]
1	0	0	i[4]
1	0	1	i[5]
1	1	0	i[6]
1	1	1	i[7]

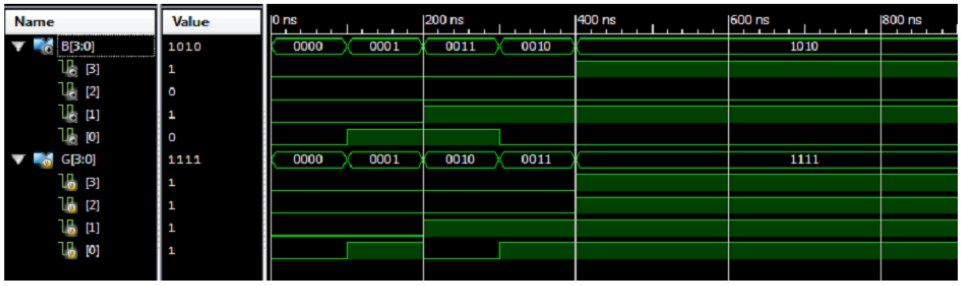
9	Sample Calculations	
10	Graphs, Outputs	**Simulation Waveforms** 
11	Results & Analysis	
12	Application Areas	Design of top level digital circuits.
13	Remarks	
14	Faculty Signature with Date	

-	**Experiment No.:**	2	**Marks**		**Date Planned**		**Date Conducted**	
1	Title	Write a Verilog program for the combinational designs						
2	Course Outcomes	Design, verify and implement the functionality of various Combinational logic circuits.						
3	Aim	d. 4 bit binary to gray converter						
4	Material / Equipment Required	Lab Manual						
5	Theory, Formula, Principle, Concept	**BOOLEAN EXPRESSIONS** G3= B3; G2= B3⊕B2; G1=B2⊕B1; G0=B1⊕B0;						
6	Procedure, Program, Activity, Algorithm, Pseudo Code	• step 1: start • step 2: write programming • step 3: save the program • step 4: check syntax • step 5:if error then correct the errors • step 6:simulate the design • step 7: FPGA implementation						

7	Block, Circuit,	**VERILOG CODE**
		module binarytogray(B,G);
		input [3:0]B;
		output [3:0]G;
		assign G[3]=B[3];
		assign G[2]=B[3]^B[2];
		assign G[1]=B[2]^B[1];
		assign G[0]=B[1]^B[0];
		endmodule						

	Model Diagram, Reaction Equation, Expected Graph	
8	Observation Table, Look-up Table, Output	• Truth table verification **TRUTH TABLE**

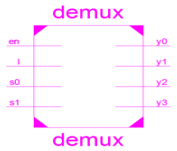
INPUT				OUTPUT			
B3	B2	B1	B0	G3	G2	G1	G0
0	0	0	0	0	0	0	0
0	0	0	1	0	0	0	1
0	0	1	0	0	0	1	1
0	0	1	1	0	0	1	0
0	1	0	0	0	1	1	0
0	1	0	1	0	1	1	1
0	1	1	0	0	1	0	1
0	1	1	1	0	1	0	0
1	0	0	0	1	1	0	0
1	0	0	1	1	1	0	1
1	0	1	0	1	1	1	1
1	0	1	1	1	1	1	0
1	1	0	0	1	0	1	0
1	1	0	1	1	0	1	1
1	1	1	0	1	0	0	1
1	1	1	1	1	0	0	0

9	Sample Calculations	
10	Graphs, Outputs	**Simulation Waveforms** 
11	Results & Analysis	
12	Application Areas	Design of top level digital circuits.
13	Remarks	
14	Faculty Signature with Date	

EC

Prepared by Checked by

Approved

-	**Experiment No.:**	2	**Marks**		**Date Planned**		**Date Conducted**	
1	Title	Write a Verilog program for the combinational designs						
2	Course Outcomes	Design, verify and implement the functionality of various Combinational logic circuits.						
3	Aim	e. 1:4 DEMUX						
4	Material / Equipment Required	Lab Manual						
5	Theory, Formula, Principle, Concept	**Black Box**  **Boolean Expression** $y0 = \overline{s1}.\overline{s0}.I.\overline{en}$ $y1 = \overline{s1}.s0.I.\overline{en}$ $y2 = s1.\overline{s0}.I.\overline{en}$ $y3 = s1.s0.I.\overline{en}$						
6	Procedure, Program, Activity, Algorithm, Pseudo Code	• step 1: start • step 2: write programming • step 3: save the program • step 4: check syntax • step 5:if error then correct the errors • step 6:simulate the design • step 7: FPGA implementation • step 8: stop						

7	Block, Circuit, Model Diagram, Reaction Equation, Expected Graph	module demux(
		input s1,s0,I,en,
		output y3,y2,y1,y0
		);
		assign y0=(~s1)&(~s0)& I& ~en;
		assign y1=(~s1)& s0& I& ~en;
		assign y2=s1&(~s0)& I & ~en;
		assign y3=s1& s0 & I & ~en;
		endmodule						

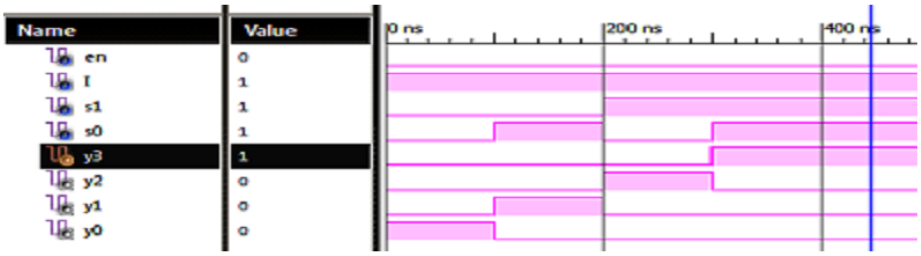
8	Observation Table, Look-up	**Truth Table** INPUT: I, en, s1, s0 — OUTPUT: y3, y2, y1, y0 1 0 0 0 | 0 0 0 1 1 0 0 1 | 0 0 1 0 1 0 1 0 | 0 1 0 0 1 0 1 1 | 1 0 0 0 0 1 X X | 0 0 0 0					• Truth table verification	

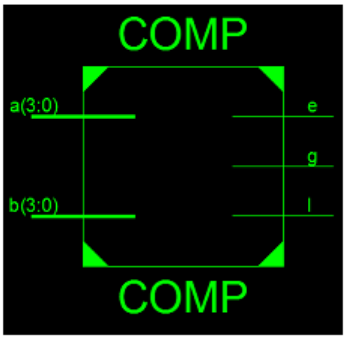
EC

Prepared by

Checked by

Approved

	Table, Output	
9	Sample Calculations	
10	Graphs, Outputs	
11	Results & Analysis	
12	Application Areas	Design of top level digital circuits.
13	Remarks	
14	Faculty Signature with Date	

-	**Experiment No.:**	2	**Marks**		**Date Planned**		**Date Conducted**	
1	Title	Write a Verilog program for the combinational designs						
2	Course Outcomes	Design, verify and implement the functionality of various Combinational logic circuits.						
3	Aim	e. 1:4 DEMUX						
4	Material / Equipment Required	Lab Manual						
5	Theory, Formula, Principle, Concept	**4 bit comparator** **BLACK BOX** 						

EC

Prepared by

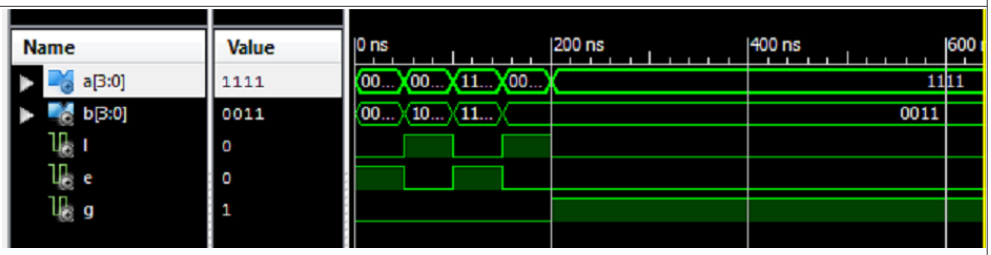
Checked by

Approved

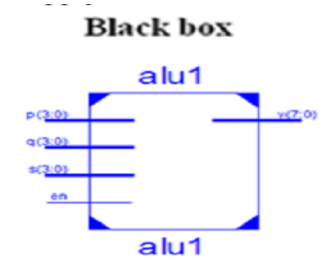
6	Procedure, Program, Activity, Algorithm, Pseudo Code	• step 1: start • step 2: write programming • step 3: save the program • step 4: check syntax • step 5:if error then correct the errors • step 6:simulate the design ntation

7	Block, Circuit, Model Diagram, Reaction Equation, Expected Graph	module COMP(
		input [3:0] a,b,
		output l,e,g
		);
		reg l,e,g;
		always @(a or b)
		begin
		l=0; e=0; g=0;
		if(a>b)
		begin
		l=0; e=0; g=1;
		end
		else if (a<b)
		begin
		l=1; e=0; g=0;
		end
		else
		begin
		l=0; e=1; g=0;
		end
		end
		endmodule

8	Observation Table, Look-up	• Truth table verification

	Table, Output	**TRUTH TABLE** INPUT: a3 a2 a1 a0 b3 b2 b1 b0 — OUTPUT: l e g 0 0 0 0 0 0 0 0 0 1 0 1 0 0 0 0 1 1 1 0 0 1 1 0 0 0 1 1 1 1 1 0 0 1 0 0 1 1 1 1 1 1 0 0 1 1 1 1 1 1 1 1 0 1 0
9	Sample Calculations	
10	Graphs, Outputs	
11	Results & Analysis	
12	Application Areas	Design of top level digital circuits.
13	Remarks	
14	Faculty Signature with Date	

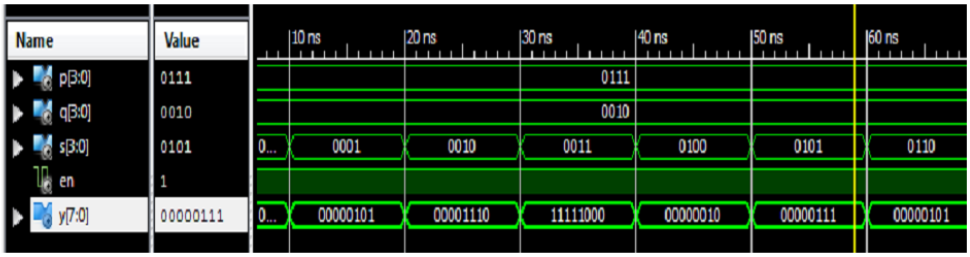
Experiment 03 : Write a Verilog code to model 32 bit ALU

-	**Experiment No.:**	3	**Marks**		**Date Planned**		**Date Conducted**	
1	Title	Write a Verilog code to model 32 bit ALU						
2	Course Outcomes	Design and Analyze the functionality of 32 bit ALU.						
3	Aim	Write a Verilog code to model 32 bit ALU						
4	Material / Equipment Required	Lab Manual						
5	Theory, Formula, Principle, Concept							

6	Procedure, Program, Activity, Algorithm, Pseudo Code	• step 1: start • step 2: write the verilog program • step 3: save the program • step 4: check syntax • step 5:if error then correct the errors • step 6:simulate the design
7	Block, Circuit, Model Diagram, Reaction Equation, Expected Graph	
8	Observation Table, Look-up Table, Output	• Truth table verification

```
module alu(
  input [3:0] a,b,s,
  input en,
  output reg[7:0] y );
  always@( a, b, s, en, y )
 begin
    if(en==1)
    begin
      case (s)
       4'd0: y=a+b;
       4'd1: y=a-b;
       4'd2: y=a*b;
       4'd3: y={4'd0, ~a};
       4'd4: y={4'd0, (a & b)};
       4'd5: y={4'd0, (a | b)};
       4'd6: y={4'd0, (a ^ b)};
       4'd7: y={4'd0, ~(a & b)};
       4'd8: y={4'd0, ~(a | b)};
       4'd9: y={4'd0, ~(a ^ b)};
       default: y=8'b00000000;
       endcase
       end
```

INPUT					OUTPUT
Operation	**en**	**S**	**A**	**B**	**Y**
A+B	1	0000	0111	0010	00001001
A-B	1	0001	0111	0010	00000101
A*B	1	0010	0111	0010	00001110
Not A	1	0011	0111	0010	11111000
A and B	1	0100	0111	0010	00000010
A or B	1	0101	0111	0010	00000111
A xor B	1	0110	0111	0010	00000101
A nand B	1	0111	0111	0010	11111101
A nor B	1	1000	0111	0010	11111000
A xnor B	1	1001	0111	0010	11111010
Any operation	0	S	1001	0111	00000000

9	Sample Calculations	
10	Graphs, Outputs	
11	Results & Analysis	
12	Application Areas	Design of top level digital circuits.
13	Remarks	
14	Faculty Signature with Date	

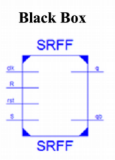
Experiment 04 : Develop the Verilog code for the following flip-flops: SR, D, JK and T.

-	**Experiment No.:**	4	**Marks**		**Date Planned**		**Date Conducted**	
1	Title	Develop the Verilog code for the SR flip-flop. SR						
2	Course Outcomes	Design, verify and implement the functionality of Flipflops.						
3	Aim	Develop the Verilog code for the following flip-flops: SR, D, JK and T.						
4	Material / Equipment	Lab Manual						

EC

Prepared by

Approved

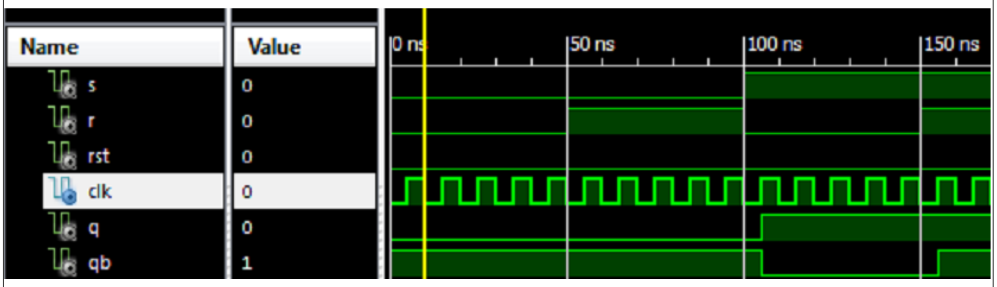


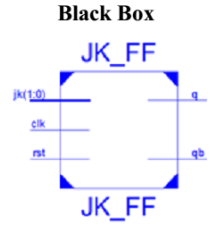
Checked by

	Required	
5	Theory, Formula, Principle, Concept	
6	Procedure, Program, Activity, Algorithm, Pseudo Code	• step 1: start • step 2: write the program • step 3: save the program • step 4: check syntax • step 5:if error then correct the errors module srff(ı input clk,rst, ion input [1:0]SR, output reg q, qb);

7	Block, Circuit, Model Diagram, Reaction Equation, Expected Graph	always@(posedge clk or posedge rst)
		begin
		if(rst==1)
		begin
		q=1'b0; qb=~q; end
		else
		begin
		case (SR)
		2'b01:begin
		q = 1'b0;qb = 1'b1; end
		2'b10:begin
		q = 1'b1;qb = 1'b0; end
		2'b11:begin
		q = 1'b1;qb = 1'b1; end
		default: begin end
		endcase
		end
		end
		endmodule

8	Observation Table, Look-up Table, Output	• Truth table verification **Truth Table** rst | S R | Q | Qb 1 | X X | 0 | 1 0 | 0 0 | Q | Qb 0 | 0 1 | 0 | 1 0 | 1 0 | 1 | 0 0 | 1 1 | 1 | 1

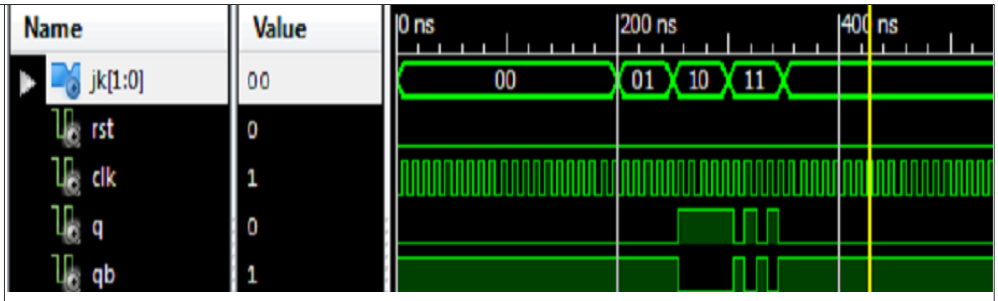
9	Sample Calculations	
10	Graphs, Outputs	
11	Results & Analysis	
12	Application Areas	Design of top level digital circuits.
13	Remarks	
14	Faculty Signature with Date	

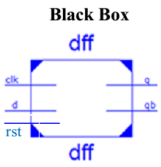
-	**Experiment No.:**	4	**Marks**		**Date Planned**		**Date Conducted**	
1	Title	Develop the Verilog code for the following flip-flop: JK FF						
2	Course Outcomes	Design, verify and implement the functionality of Flipflops.						
3	Aim	Develop the Verilog code for the following flip-flops: SR, D, JK and T.						
4	Material / Equipment Required	Lab Manual						
5	Theory, Formula, Principle, Concept							

6	Procedure, Program, Activity, Algorithm, Pseudo Code	• step 1: start • step 2: write the program • step 3: save the program • step 4: check syntax • step 5:if error then correct the errors • step 6:simulate the design • step 7: FPGA implementation • step 8: stop

7	Block, Circuit, Model Diagram, Reaction Equation, Expected Graph	`module jk_ff(`
		`input [1:0] jk,`
		`input rst,clk,`
		`output reg q,qb);`
		`always@(posedge clk,posedge rst)`
		`begin`
		`if(rst==1)`
		`begin q=1'b0; qb=~q; end`
		`else`
		`begin`
		`case (jk)`
		`2'b01:q=1'b0;`
		`2'd10:q=1'b1;`
		`2'b11:q=~q;`
		`default: begin end`
		`endcase`
		`qb = ~q;`
		`end`
		`end`
		`endmodule`

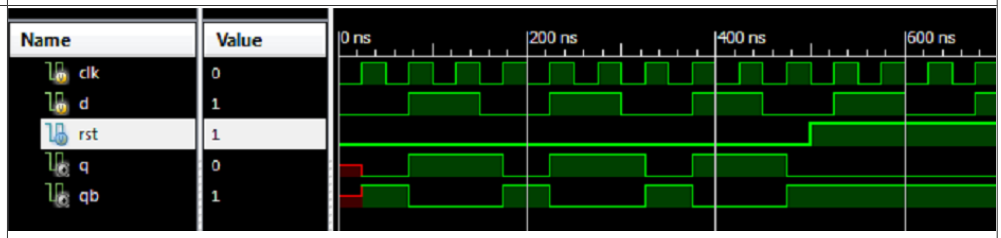
8	Observation Table, Look-up Table, Output	• Truth table verification **Truth Table** rst | J K | Q | Qb 1 | X X | 0 | 1 0 | 0 0 | Q | Qb 0 | 0 1 | 0 | 1 0 | 1 0 | 1 | 0 0 | 1 1 | Toggle
9	Sample Calculations	

10	Graphs, Outputs	
11	Results & Analysis	
12	Application Areas	Design of top level digital circuits.
13	Remarks	
14	Faculty Signature with Date	

–	**Experiment No.:**	4	**Marks**		**Date Planned**		**Date Conducted**	
1	Title	Develop the Verilog code for the following flip-flops: SR, D, JK and T.						
2	Course Outcomes	Design, verify and implement the functionality of Flipflops.						
3	Aim	Develop the Verilog code for the following flip-flop: D FF						
4	Material / Equipment Required	Lab Manual						
5	Theory, Formula, Principle, Concept							
6	Procedure, Program, Activity, Algorithm, Pseudo Code	• step 1: start • step 2: write the program • step 3: save the program • step 4: check syntax • step 5:if error then correct the errors • step 6:simulate the design • step 7: FPGA implementation • step 8: stop						
7	Block, Circuit, Model Diagram,	• – • –						

	Reaction Equation, Expected Graph	• –
8	Observation Table, Look-up Table, Output	• Truth table verification

9	Sample Calculations	module dff(
		input d,clk,rst,
		output q,qb);
		reg q,qb;
		always@(posedge clk,posedge rst)
		begin
		if (rst==1)
		begin q=0; qb=~q; end
		else
		begin q=d; qb=~q; end
		end
		endmodule

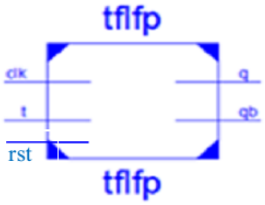
10	Graphs, Outputs	
11	Results & Analysis	
12	Application Areas	Design of top level digital circuits.
13	Remarks	
14	Faculty Signature with Date	

–	**Experiment No.:**	4	**Marks**		**Date Planned**		**Date Conducted**	
1	Title	Develop the Verilog code for the following flip-flops: T FF.						
2	Course Outcomes	Design, verify and implement the functionality of Flipflops.						
3	Aim	Develop the Verilog code for the following flip-flops: SR, D, JK and T.						
4	Material /	Lab Manual						

EC

Prepared by Checked by

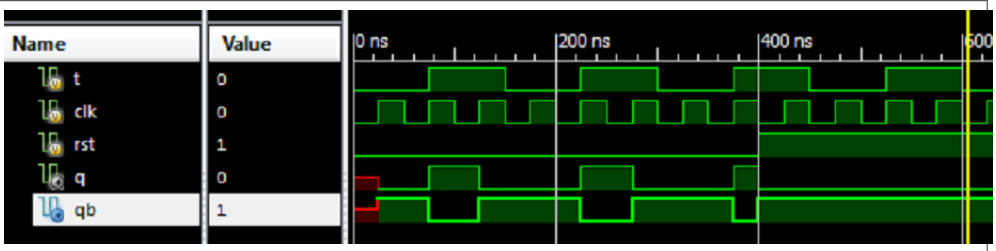
Approved

	Equipment Required	
5	Theory, Formula, Principle, Concept	**Black Box** 
6	Procedure, Program, Activity, Algorithm, Pseudo Code	• step 1: start • step 2: write the program • step 3: save the program • step 4: check syntax • step 5:if error then correct the errors esign entation

7	Block, Circuit, Model Diagram, Reaction Equation, Expected Graph	module tff(t,clk,rst,q,qb);
		input t,clk,rst;
		output q,qb;
		reg q,qb;
		always@(posedge clk,posedge rst)
		begin
		if (rst==1)
		begin q=0;qb=~q; end
		else if(t==1) begin
		q=~ q; qb = ~q; end
		end
		endmodule

8	Observation Table, Look-up Table, Output	**Truth Table** • Truth table verification

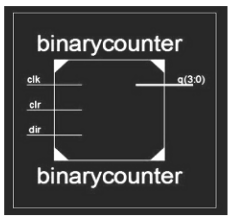
rst	T	Q	Qb
1	X	0	1
0	0	Q	Qb
0	1	Toggle	

9	Sample Calculations	
10	Graphs, Outputs	
11	Results & Analysis	
12	Application Areas	Design of top level digital circuits.
13	Remarks	
14	Faculty Signature with Date	

Experiment 05 : Design a 4 bit binary, BCD counters (Synchronous reset and Asynchronous reset) and “any sequence” counters, using Verilog code.

-	**Experiment No.:**	5	**Marks**		**Date Planned**		**Date Conducted**	
1	Title	Design a 4 bit binary counters (Synchronous reset and Asynchronous reset) and “any sequence” counters, using Verilog code.						
2	Course Outcomes	Design, verify and implement the functionality of counters.						
3	Aim	Design a 4 bit binary, BCD counters (Synchronous reset and Asynchronous reset) and “any sequence” counters, using Verilog code.						
4	Material / Equipment Required	Lab Manual						
5	Theory, Formula, Principle, Concept							
6	Procedure, Program, Activity, Algorithm, Pseudo Code	• step 1: start • step 2: write the program • step 3: save the program • step 4: check syntax • step 5:if error then correct the errors • step 6:simulate the design • step 7: FPGA implementation • step 8: stop						
7	Block, Circuit, Model Diagram,	• - • -						

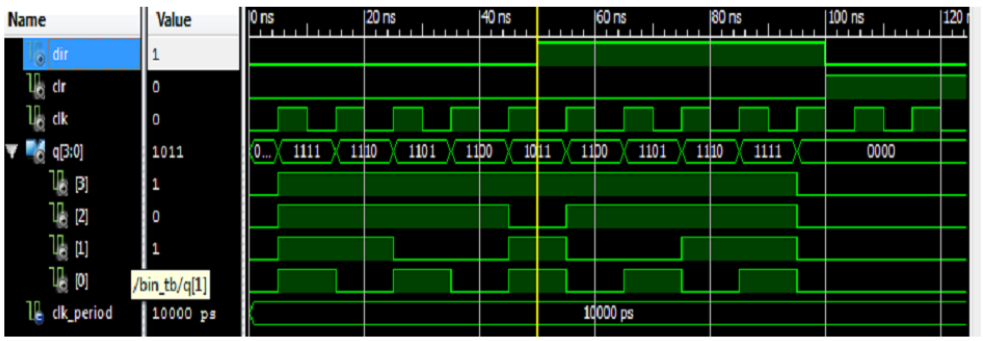
	Reaction Equation, Expected Graph	• -
8	Observation Table, Look-up Table, Output	• Truth table verification
9	Sample Calculations	
10	Graphs, Outputs	
11	Results & Analysis	
12	Application Areas	Design of top level digital circuits.
13	Remarks	
14	Faculty Signature with Date	

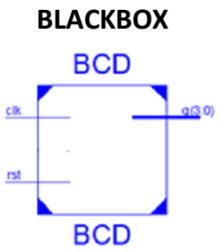
-	**Experiment No.:**	5	**Marks**		**Date Planned**		**Date Conducted**	
1	Title	Design a 4 bit binary, BCD counters (Synchronous reset and Asynchronous reset) and “any sequence” counters, using Verilog code.						
2	Course Outcomes	Design, verify and implement the functionality of counters.						
3	Aim	Design a 4 bit binary, BCD counters (Synchronous reset and Asynchronous reset) and “any sequence” counters, using Verilog code.						
4	Material / Equipment Required	Lab Manual						
5	Theory, Formula, Principle, Concept							
6	Procedure, Program, Activity, Algorithm, Pseudo Code	• step 1: start • step 2: write the program • step 3: save the program • step 4: check syntax • step 5:if error then correct the errors • step 6:simulate the design • step 7: FPGA implementation • step 8: stop						

7	Block, Circuit, Model Diagram, Reaction Equation, Expected Graph	
8	Observation Table, Look-up Table, Output	• Truth table verification
9	Sample Calculations	

```
module binarycounter(input clk, clr,dir,
output[3:0] q);
reg [3:0] count;
always @(posedge clk or posedge clr)
begin
if(clr)
 count <= 4'h0;
else if (dir)
 count <= count - 4'd1;
else
 count <= count + 4'd1;
end
assign q = count;
endmodule
```

clk	dir=0	dir=1
	q	q
1	0000	1111
1	0001	1110
1	0010	1101
1	0011	1100
1	0100	1011
1	0101	1010
1	0110	1001
1	0111	1000
1	1000	0111
1	1001	0110
1	1010	0101
1	1011	0100
1	1100	0011
1	1101	0010
1	1110	0001
1	1111	0000
0	0000	1111

10	Graphs, Outputs	
11	Results & Analysis	
12	Application Areas	Design of top level digital circuits.
13	Remarks	
14	Faculty Signature with Date	

–	**Experiment No.:**	5	**Marks**		**Date Planned**		**Date Conducted**	
1	Title	Design a 4 bit BCD counters (Synchronous reset and Asynchronous reset) and “any sequence” counters, using Verilog code.						
2	Course Outcomes	Design, verify and implement the functionality of counters.						
3	Aim	Design a 4 bit BCD counters (Synchronous reset and Asynchronous reset) and “any sequence” counters, using Verilog code.						
4	Material / Equipment Required	Lab Manual						
5	Theory, Formula, Principle, Concept							
6	Procedure, Program, Activity, Algorithm, Pseudo Code	• step 1: start • step 2: write the program • step 3: save the program • step 4: check syntax • step 5:if error then correct the errors						

- step 6:simulate the design

```
module bcd(rst,clk,q);
   input rst,clk;
   output reg[3:0] q;
         always@(posedge clk or posedge rst)
         begin
            if(rst)
              q=4'd0;
            else
               begin
                 q=q+1'b1;
                  if (q==4'd10)
                       q=4'd0;
               end
         end
endmodule
```

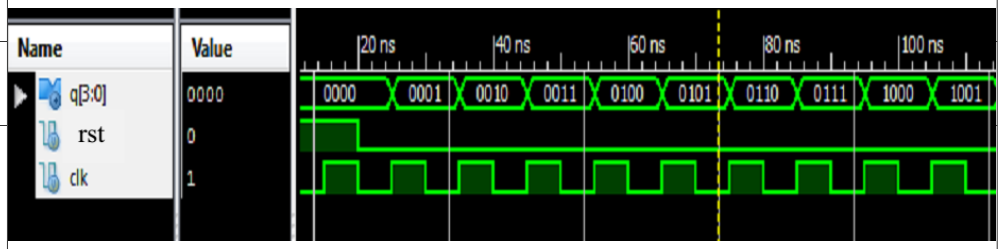
INPUT		OUTPUT
Rst	Clk	Q
0	1	0000
0	1	0001
0	1	0010
0	1	0011
0	1	0100
0	1	0101
0	1	0110
0	1	0111
0	1	1000
0	1	1001
1	X	0000

8 Observation Table, Look-up Table, Output

- Truth table verification

9 Sample Calculations

10 Graphs, Outputs



11 Results & Analysis

12	Application Areas	Design of top level digital circuits.
13	Remarks	
14	Faculty Signature with Date	

Experiment 06 : Write HDL code to display messages on an alpha numeric LCD display

-	**Experiment No.:**	1	**Marks**		**Date Planned**		**Date Conducted**	
1	Title	Write HDL code to display messages on an alpha numeric LCD display						
2	Course Outcomes	Design the digital system for Interfacing FPGA with Hex Keypad.						
3	Aim	Write HDL code to display messages on an alpha numeric LCD display						
4	Material / Equipment Required	Lab Manual						
5	Theory, Formula, Principle, Concept							
6	Procedure, Program, Activity, Algorithm, Pseudo Code	• step 1: start • step 2: write programming • step 3: save the program • step 4: check syntax • step 5:if error then correct the errors • step 6:simulate the design • step 7: FPGA implementation • step 8: stop						
7	Block, Circuit, Model Diagram, Reaction Equation, Expected Graph	• - • - • -						
8	Observation Table, Look-up Table, Output	• Truth table verification						
9	Sample Calculations							
10	Graphs, Outputs							
11	Results & Analysis							
12	Application Areas	Design of top level digital circuits.						
13	Remarks							
14	Faculty Signature with Date							

Experiment 07 : Write HDL code to control speed, direction of DC and Stepper motor.

-	**Experiment No.:**	8	**Marks**		**Date Planned**		**Date Conducted**	
1	Title	Write HDL code to control speed, direction of DC and Stepper motor.						
2	Course Outcomes	Design the digital system for Controlling the speed and direction of stepper motor using HDL						
3	Aim	Write HDL code to control speed, direction of DC and Stepper motor.						
4	Material / Equipment Required	Lab Manual						
5	Theory, Formula, Principle, Concept							
6	Procedure, Program, Activity, Algorithm, Pseudo Code	• step 1: start • step 2: write programming • step 3: save the program • step 4: check syntax • step 5:if error then correct the errors • step 6:simulate the design • step 7: FPGA implementation • step 8: stop						
7	Block, Circuit, Model Diagram, Reaction Equation, Expected Graph	• - • - • -						
8	Observation Table, Look-up Table, Output	• Truth table verification						
9	Sample Calculations							
10	Graphs, Outputs							
11	Results & Analysis							
12	Application Areas	Design of top level digital circuits.						
13	Remarks							
14	Faculty Signature with Date							

Experiment 08 : Write HDL code to accept Analog signal, Temperature sensor and display the data on LCD or Seven segment display.

-	**Experiment No.:**	9	**Marks**		**Date Planned**		**Date Conducted**	
1	Title	Write HDL code to accept Analog signal, Temperature sensor and display the data on LCD or Seven segment display.						
2	Course Outcomes	Design the digital system for Interfacing FPGA with temperature sensor						
3	Aim	Write HDL code to accept Analog signal, Temperature sensor and display the data on LCD or Seven segment display.						
4	Material / Equipment Required	Lab Manual						
5	Theory, Formula, Principle, Concept							
6	Procedure, Program, Activity, Algorithm, Pseudo Code	• step 1: start • step 2: write programming • step 3: save the program • step 4: check syntax • step 5:if error then correct the errors • step 6:simulate the design • step 7: FPGA implementation • step 8: stop						
7	Block, Circuit, Model Diagram, Reaction Equation, Expected Graph	• - • - • -						
8	Observation Table, Look-up Table, Output	• Truth table verification						
9	Sample Calculations							
10	Graphs, Outputs							
11	Results & Analysis							
12	Application Areas	Design of top level digital circuits.						
13	Remarks							
14	Faculty Signature with Date							

Experiment 10 : Write HDL code to generate different waveforms (Sine, Square, Triangle, Ramp etc.,) using DAC – change the frequency.

–	**Experiment No.:**	10	**Marks**		**Date Planned**		**Date Conducted**	
1	Title	Write HDL code to generate different waveforms (Sine, Square, Triangle, Ramp etc.,) using DAC – change the frequency.						
2	Course Outcomes	Study of DAC and generation of different signals using HDL .						
3	Aim	Write HDL code to generate different waveforms (Sine, Square, Triangle, Ramp etc.,) using DAC – change the frequency.						
4	Material / Equipment Required	Lab Manual						
5	Theory, Formula, Principle, Concept							
6	Procedure, Program, Activity, Algorithm, Pseudo Code	• step 1: start • step 2: write programming • step 3: save the program • step 4: check syntax • step 5:if error then correct the errors • step 6:simulate the design • step 7: FPGA implementation • step 8: stop						
7	Block, Circuit, Model Diagram, Reaction Equation, Expected Graph							
8	Observation Table, Look-up Table, Output	• Truth table verification						
9	Sample Calculations							
10	Graphs, Outputs							
11	Results & Analysis							
12	Application Areas	Design of top level digital circuits.						
13	Remarks							
14	Faculty Signature with Date							

-	**Experiment No.:**	10	**Marks**		**Date Planned**		**Date Conducted**	
1	Title	Write HDL code to generate different waveforms (Sine, Square, Triangle, Ramp etc.,) using DAC – change the frequency.						
2	Course Outcomes	Study of DAC and generation of different signals using HDL .						
3	Aim	Write HDL code to generate different waveforms (Sine, Square, Triangle, Ramp etc.,) using DAC – change the frequency.						
4	Material / Equipment Required	Lab Manual						
5	Theory, Formula, Principle, Concept							
6	Procedure, Program, Activity, Algorithm, Pseudo Code	• step 1: start • step 2: write programming • step 3: save the program • step 4: check syntax • step 5:if error then correct the errors • step 6:simulate the design • step 7: FPGA implementation • step 8: stop						
7	Block, Circuit, Model Diagram, Reaction Equation, Expected Graph	• – • – • –						
8	Observation Table, Look-up Table, Output	• Truth table verification						
9	Sample Calculations							
10	Graphs, Outputs							
11	Results & Analysis							
12	Application Areas	Design of top level digital circuits.						
13	Remarks							
14	Faculty Signature with Date							

-	**Experiment No.:**	10	**Marks**		**Date Planned**		**Date Conducte**	

							d	
1	Title	Write HDL code to generate different waveforms (Sine, Square, Triangle, Ramp etc.,) using DAC – change the frequency.						
2	Course Outcomes	Study of DAC and generation of different signals using HDL .						
3	Aim	Write HDL code to generate different waveforms (Sine, Square, Triangle, Ramp etc.,) using DAC – change the frequency.						
4	Material / Equipment Required	Lab Manual						
5	Theory, Formula, Principle, Concept							
6	Procedure, Program, Activity, Algorithm, Pseudo Code	• step 1: start • step 2: write programming • step 3: save the program • step 4: check syntax • step 5:if error then correct the errors • step 6:simulate the design • step 7: FPGA implementation • step 8: stop						
7	Block, Circuit, Model Diagram, Reaction Equation, Expected Graph	• – • – • –						
8	Observation Table, Look-up Table, Output	• Truth table verification						
9	Sample Calculations							
10	Graphs, Outputs							
11	Results & Analysis							
12	Application Areas	Design of top level digital circuits.						
13	Remarks							
14	Faculty Signature with Date							

–	**Experiment No.:**	10	**Marks**		**Date Planned**		**Date Conducted**	
1	Title	Write HDL code to generate different waveforms (Sine, Square, Triangle, Ramp etc.,) using DAC – change the frequency.						

2	Course Outcomes	Study of DAC and generation of different signals using HDL .
3	Aim	Write HDL code to generate different waveforms (Sine, Square, Triangle, Ramp etc.,) using DAC – change the frequency.
4	Material / Equipment Required	Lab Manual
5	Theory, Formula, Principle, Concept	
6	Procedure, Program, Activity, Algorithm, Pseudo Code	• step 1: start • step 2: write programming • step 3: save the program • step 4: check syntax • step 5:if error then correct the errors • step 6:simulate the design • step 7: FPGA implementation • step 8: stop
7	Block, Circuit, Model Diagram, Reaction Equation, Expected Graph	• – • – • –
8	Observation Table, Look-up Table, Output	• Truth table verification
9	Sample Calculations	
10	Graphs, Outputs	
11	Results & Analysis	
12	Application Areas	Design of top level digital circuits.
13	Remarks	
14	Faculty Signature with Date	

Experiment 11 : Write HDL code to simulate Elevator operation.

–	**Experiment No.:**	1	**Marks**		**Date Planned**		**Date Conducted**	
1	Title	Write HDL code to simulate Elevator operation.						
2	Course Outcomes	Design an Elevator						
3	Aim	Write HDL code to simulate Elevator operation.						
4	Material / Equipment Required	Lab Manual						

EC

Prepared by Checked by

Approved

5	Theory, Formula, Principle, Concept	
6	Procedure, Program, Activity, Algorithm, Pseudo Code	• step 1: start • step 2: write programming • step 3: save the program • step 4: check syntax • step 5:if error then correct the errors • step 6:simulate the design • step 7: FPGA implementation • step 8: stop
7	Block, Circuit, Model Diagram, Reaction Equation, Expected Graph	• - • - • -
8	Observation Table, Look-up Table, Output	• Truth table verification
9	Sample Calculations	
10	Graphs, Outputs	
11	Results & Analysis	
12	Application Areas	Design of top level digital circuits.
13	Remarks	
14	Faculty Signature with Date	

APPENDIX-1
DETAILED PROCEDURE FOR PROGRAM EXECUTION

Procedure To Work With Xilinx Tool :

To Create A Project

1. Double Click On XILINX ISE Icon
2. In ise project navigator window go to file and click on new project
3. Give any project name and note down the path, click next
4. Select spartan6 in family field and click next

To write HDL code

1. go to hierarchy window
2. right click on XC6SLX4-3TQC144
3. click on new source
4. select vhdl or verilog module
5. give file name without space and note down the path, click next
6. give name to input and output varible and click next
7. click on finish
8. type the program in editor window and save it
9. select the file hierarchy window
10. go to process window and click on synthesize- xst

11. double click on check syntax

12. to check syntax error click on console if any error is present click on particular error and fix the error

To Check Simulation Result

1. Select Simulation
2. Select Program In Hierarchy Window
3. Open Isim Simulator In Process Window
4. Double Click On Simulate Behavioral Model
5. Simulation Window Will Open
6. Right Click On Input(At A Time One Input)
7. Click On Force Constant
8. Enter The Input Value In ''force To Value'' Field
9. Apply And Then Ok
10. Repeat Steps 6 To 9 For All Inputs
11. In Time Period Field Make It To 1second
12. Click On Run For The Time Specified On Toolbar
13. Verify The Simulation Results With Truth Table

Procedure To Download Onto Fpga:

Procedure For Creating A Bit File

1. Only One Program Should Be Available In Hierarchy Window
2. Select File (program)
3. Go To User Constraints In Process Window
4. Double Click On I/O Pin Planning (plan Ahead) Pre-Sysnthesis
5. Click On Yes
6. In I/O Ports Window Open Scalar Ports
7. I/O Port Properties Window Should Be Opened
8. Click On Site And
9. Enter The Pin No (provided By Vendor) Example: P124- And Click On Apply

(assigning Pin Will Be Done For All Inputs And Outputs)

10. Click On Save

11. Ucf File Will Be Created In Hierarchy Window

12. In Process Window Double Click On Design: Generate Programming File

13. In Process Window Double Click On Configure Target Device->Ok

14. Bit File Will Be Generated

Procedure For Dumping Bit File On To The FPGA Kit

1. Double Click On Vsf-6-2.0(Present On Desktop)

2. Check Com Port Number (right Click My Computer-->Manage-->Device

Manager-->Open Ports(Com&Lpt))

3. Select Particular Com Port Number

4. In VSF-6-2.0 Window Select File And Click On Load File

5. Select The Particular Bit File And Click On Open

6. Click On Configure File

''FPGA Programmed Successfully'' Will Be Displayed

7. Verify The Results With Truth Table