

ATRIA INSTITUTE OF TECHNOLOGY (Affiliated To Visvesvaraya Technological University, Belgaum) Anandanagar, Bangalore-24

# DEPARTMENT OF ELECTRONICS AND COMMUNICATION

# HDL LAB MANUAL

5th SEMESTER ELECTRONICS AND COMMUNICATION

**SUBJECT CODE: 18ECL58** 

2020-21



## (Affiliated To Visvesvaraya Technological University, Belgaum) Anandanagar, Bangalore-24

## DEPARTMENT OF ELECTRONICS AND COMMUNICATION HDL LAB MANUAL

The HDL Laboratory Manual pertaining V semester ECE has been prepared as per VTU syllabus and all the experiments are designed, tested and verified according to the experiment list.

This manual typically contains practical/lab sessions related to Verilog HDL and interfacing various hardware devices with CPLD XC9572 which provides a better understanding of the subject. Students are advised to thoroughly go through this manual as it provides them practical insights.

# SYLLABUS

Laboratory Code	18ECL58	CIE Marks	40		
Number of Lecture Hours/Week	02Hr Tutorial (Instructions)+ 02 Hours Laboratory	SEE Marks	60		
RBT Level	L1, L2, L3	Exam Hours	03		
	CREDITS – 02				
Course Learning Objectives: T Familiarize with the CA Understand simulation a Program FPGAs/CPLDs Interface hardware to pr Choose either Verilog or	This course will enable students to: D tool to write HDL programs. nd synthesis of digital design. s to synthesize the digital designs. ogrammable ICs through I/O ports.				
<b>Note:</b> Programming can be do performance testing may be do by simulation with tools such as	one using any compiler. Download the prog one using 32 channel pattern generator and log Altera/Modelsim or equivalent. Laboratory Experiments	grams on a FPGA/CPLD gic analyzer apart from	board and verification		
	PART A : Programming				
<ol> <li>Write Verilog program for the following combinational design along with test bench to verify the design:         <ol> <li>2 to 4 decoder realization using NAND gates only (structural model)</li> <li>8 to 3 encoder with priority and without priority (behavioural model)</li> <li>8 to 1 multiplexer using case statement and if statements</li> <li>4-bit binary to gray converter using 1-bit gray to binary converter 1-bit adder and subtractor</li> </ol> </li> <li>Model in Verilog for a full adder and add functionality to perform logical operations of XOR, XNOR, AND and OR gates. Write test bench with appropriate input patterns to verify the modeled behaviour</li> </ol>					
<ul> <li>3. Verilog 32-bit ALU shown in figure below and verify the functionality of ALU by selecting appropriate test patterns. The functionality of the ALU is presented in Table 1.</li> <li>a. Write test bench to verify the functionality of the ALU considering all possible input patterns b. The enable signal will set the output to required functions if enabled, if disabled all the outputs are set to tri-state</li> <li>c. The acknowledge signal is set high after every operation is completed</li> </ul>					

		A(31:0) B(31:0)	
	• Opcode(2:0)	→ 32-bit ALU	
	Enable		Result[32:0]
	Figure	e 1 ALU top level block diagram	
Opcode(2:0)	ALU Operation	Rei	marks
000	A + B	Addition of two numbers	Both A and B are in two's
001	4 D	Subtraction of two works	complement format
001	A - B	Subtraction of two numbers	A is in two's complement
010	A + 1	Decrement accumulator by 1	format
100	Δ		Inputs can be in any format
100	A Complement	Complement	
101			
110			_
111		Table 1 ALU Eurotions	
. Write Verilog code	e for SR, D and JK and	verify the flip flop.	
. Write Verilog code . Write Verilog code	e for SR, D and JK and e for 4-bit BCD synchro	verify the flip flop.	
. Write Verilog code . Write Verilog code . Write Verilog code performing division	e for SR, D and JK and e for 4-bit BCD synchro e for counter with given of clock by 2, 4, 8 and	verify the flip flop. onous counter. a input clock and check whether it v 16. Verify the functionality of the c	works as clock divider code.
Write Verilog code Write Verilog code Write Verilog code performing division ART-B : Interfaci quivalent tool can b	e for SR, D and JK and e for 4-bit BCD synchro e for counter with given of clock by 2, 4, 8 and <b>ng and Debugging</b> (EI e used)	verify the flip flop. onous counter. a input clock and check whether it v 16. Verify the functionality of the o DWinXP, PSpice, MultiSim, Proteu	works as clock divider code. is, CircuitLab or any other
Write Verilog code Write Verilog code Write Verilog code performing division ART-B : Interfacion quivalent tool can b Write a Verilog given input clo	e for SR, D and JK and e for 4-bit BCD synchro e for counter with given of clock by 2, 4, 8 and <b>ng and Debugging</b> (EI e used) code to design a clock ock. Port the design to 1	verify the flip flop. onous counter. n input clock and check whether it w 16. Verify the functionality of the o DWinXP, PSpice, MultiSim, Proteu divider circuit that generates 1/2, 1 FPGA and validate the functionality	works as clock divider code. us, CircuitLab or any other /3 <sup>rd</sup> and 1/4 <sup>th</sup> clock from a y through oscilloscope.
<ul> <li>Write Verilog code</li> <li>Write Verilog code</li> <li>Write Verilog code</li> <li>performing division</li> <li>PART-B : Interfacing</li> <li>quivalent tool can be</li> <li>Write a Verilog given input close</li> <li>Interface a DC restance</li> </ul>	e for SR, D and JK and e for 4-bit BCD synchro e for counter with given of clock by 2, 4, 8 and <b>ng and Debugging</b> (EI e used) code to design a clock ock. Port the design to 1 motor to FPGA and wri	verify the flip flop. onous counter. a input clock and check whether it v 16. Verify the functionality of the o DWinXP, PSpice, MultiSim, Proteu divider circuit that generates 1/2, 1 FPGA and validate the functionality ite Verilog code to change its speed	works as clock divider code. Is, CircuitLab or any other /3 <sup>rd</sup> and 1/4 <sup>th</sup> clock from a y through oscilloscope.
<ul> <li>Write Verilog code</li> <li>Write Verilog code</li> <li>Write Verilog code</li> <li>Write Verilog code</li> <li>performing division</li> <li>PART-B : Interfacing</li> <li>xquivalent tool can be</li> <li>Write a Verilog given input cle</li> <li>Interface a DC re</li> <li>Interface a Step in turn may construct of the second stepper motor Dip switch is</li> </ul>	e for SR, D and JK and e for 4-bit BCD synchro e for counter with given of clock by 2, 4, 8 and <b>ng and Debugging</b> (EI e used) code to design a clock ock. Port the design to 1 motor to FPGA and wri oper motor to FPGA a control a Robotic Arm : (i) +N steps if Switch closed (iii) –N steps if	verify the flip flop. onous counter. n input clock and check whether it v 16. Verify the functionality of the o DWinXP, PSpice, MultiSim, Proteu divider circuit that generates 1/2, 1 FPGA and validate the functionality ite Verilog code to change its speed nd write Verilog code to control b. External switches to be used for h no.1 of a Dip switch is closed (if Switch no. 3 of a Dip switch is closed)	works as clock divider code. Is, CircuitLab or any other /3 <sup>rd</sup> and 1/4 <sup>th</sup> clock from a y through oscilloscope. I and direction. the Stepper motor rotation which or different controls like rotate the ii) +N/2 steps if Switch no. 2 of a losed etc.
<ul> <li>Write Verilog code</li> <li>Write Verilog code</li> <li>Write Verilog code</li> <li>performing division</li> <li>PART-B : Interfaci</li> <li>quivalent tool can b</li> <li>Write a Verilog given input clo</li> <li>Interface a DC r</li> <li>Interface a Step in turn may c Stepper motor Dip switch is</li> <li>Interface a DA KHz) frequen Down sampled</li> </ul>	e for SR, D and JK and e for 4-bit BCD synchro e for counter with given of clock by 2, 4, 8 and <b>ng and Debugging</b> (EI e used) code to design a clock ock. Port the design to 1 motor to FPGA and write control a Robotic Arm : (i) +N steps if Switch closed (iii) –N steps if C to FPGA and write cy. Modify the code to d signals by connecting	verify the flip flop. Donous counter. A input clock and check whether it w 16. Verify the functionality of the of DWinXP, PSpice, MultiSim, Proteu divider circuit that generates 1/2, 1 FPGA and validate the functionality ite Verilog code to change its speed and write Verilog code to control b. External switches to be used for h no.1 of a Dip switch is closed (if Switch no. 3 of a Dip switch is closed verilog code to generate Sine w o down sample the frequency to b g them to an oscilloscope.	works as clock divider code. Is, CircuitLab or any other /3 <sup>rd</sup> and 1/4 <sup>th</sup> clock from a y through oscilloscope. I and direction. the Stepper motor rotation which or different controls like rotate the ii) +N/2 steps if Switch no. 2 of a losed etc. ////////////////////////////////////

6. Write Verilog code to convert an analog input of a sensor to digital form and to display the same on a suitable display like set of simple LEDs, 7-segment display digits or LCD display.

**Course Outcomes:** At the end of this course, students should be able to:

- Write the Verilog/VHDL programs to simulate Combinational circuits in Dataflow, Behavioral and Gate level Abstractions.
- Describe sequential circuits like flip flops and counters in Behavioral description and obtain simulation waveforms.
- Synthesize Combinational and Sequential circuits on programmable ICs and test the hardware.
- Interface the hardware to the programmable chips and obtain the required output

#### **Conduct of Practical Examination:**

- All laboratory experiments are to be included for practical examination.
- Students are allowed to pick one experiment from the lot.
- Strictly follow the instructions as printed on the cover page of answer script for breakup of marks.
- Change of experiment is allowed only once and Marks allotted to the procedure part to be made zero.

## **Introduction to HDL**

Hardware description language (HDL) is a computer aided design (CAD) tool for the modern design and synthesis of digital systems. The recent, steady advances in semiconductor technology continue to increase the power and complexity of digital systems. Due to their complexity, such systems cannot be realized using discrete integrated circuits. They are usually realized using high density, programmable chips, such as application specific Integrated circuits (ASICs) and Field programmable gate arrays (FPGAs) and require sophisticated CAD tools. HDL is an integral part of such tools. HDL offers the designer a very efficient tool for implementing and synthesizing designs on chips.

The two widely used hardware description languages are VHDL and Verilog. These languages provide support for modeling the system hierarchically and also supports top down and bottom up design methodologies. The system and its subsystems can be described at any level of abstraction ranging from the architecture level to the gate level.

The complex constructs and features of these languages are enough to be able to model designs with high degrees of complexity.

Software Required: Xilinx ISE14.7 Hardware Used: XC9572 CPLD

S.NO	NAME OF THE EXPERIMENT	PAGE NO			
Part A					
<b>1.a</b>	2:4 Decoder	23			
<b>1.b</b>	8:3 Encoder with and without Priority	27			
<b>1.c</b>	8x1 MUX using case and IF statments	35			
2	Realization of Full adder with basic gates	40			
3	32 Bit ALU	42			
<b>4.(i)</b>	SRFF	47			
(ii)	JKFF	50			
(iii)	DFF	53			
5	4 Bit synchronous BCD counter	56			
6	Frequency Divider	59			
	Part B				
1	DC Motor Interface	62			
2	Stepper Motor Interface	64			
3	Elevator operation	67			
4	Hardware Clock Divider	71			
		•			
	Question Bank	75			

# LIST OF EXPERIMENTS

# PROCEDURE

#### **Procedure to work with Xilinx ISE 14.7 software:**

#### 1. To Create a Project:

A project in ISE is a collection of all files necessary to create and download a design to the selected device.

Open XilinxISE Window.



# 2. Select File-New Project

EIF Project Nevgetor (F.2	IN IN INC.			Contraction of the local division of the loc	- 3- <b>-</b> 1
Edit View Project	Source Process 1	ols Window Layest Help			
New Project.	n eix (i q	· //88/31	3 3 7 7 PR	≥ = <b>1</b> 9	
Open Project-	+08×				
Open (sumple	lesign Suite				
Copy Preject.					
Close Rispect	a				
Nev Col+N					
9 Open Chil+O	E.				
Close					
3 See . (2663	t beau to apen				
See AL.					
@liceA	1				
Post Preview-					
Ret. CdvP	6				
Recent Files I					
Recent Projects	K I				
for .	-				
	-				
Canadie					+0 <i>8</i>
1 Anna 1 Anna 1	· Warmen 1 P	w They Dan He			
Create a new parter	The second state of	COME NOW			
					10
					4 1 · · · · · · · · · · · · · · · · · ·

Start +D # X	TTERES NO FLO V	
Welcome to the ISER Design Suite		
Project comments		
Open Project	and the second se	
New Project	💝 New Project Without	
Recent projects	Create New Project	
Double did: on a project in the latibelow to open		
and5k +	Sectory or operations and remeat the the research	
DFF C		
rpush +	Aine dres	100
10070	Lighten Ut area	
Addition@ivesaurces	Working Developmy, UI (Barro)	
Tuturals on the Web Design Resolution	L'estipato;	
Assumetress		
	Select the type of top-level assume for the propert	
	Too level assars how:	
Consile	HQ.	+ 0
	Laure I	David 1
🖹 Consais 🥥 Errors 🦺 Harnings 🙀 Find in File	s Results	
		15 M 4 4 4 45 M

3. Give the name of the project, browse the location for the project and select 'Next'

4. Enter the CPLD device details and choose 'Next'.

Welcome to the ISE® Design Suite Project commands				
Open Project Project Browner	The subscription of the			
New Project	New Provent Wigard			
Securit projects	Project Settings			
Double clob on a project in the last below to spon	Rect/ drive and provid properties			
andik +	Select the device and design flow for the p	(960)		
1010	Property Name	Value	10	
mush *	Evaluation Development Roard	None Specified		
	Freduct Category	A.	R	
ddavut recauces	Family	XC9500-CPLDs	- 20	
and the second se	Device	xc9572	- A-	
Laticidad pol tres como	Factors	PC84	- A	
Bapérutina Vérdra	fired	-15		
	Tup-Level Source Type	NOT ANOTH ANALYSI		
	Considered	Zari (Millar) (Medical		
	Enderset Languages	There is a second	-8	
	Encost, Confiction in Brief Cir.	Chara all utility		
	Manual Crystacke Online	Store of the store		
	VHDL Source Analysis Standard	WHOLES		
		Magazine	····	
	(CONTRACT)		Contract of the second	
	La real and an	- New T		
M. Commission				

5. Verify the previous steps in the summary and choose 'Finish'.



Project window will be opened.



### **II.** To create a Verilog source(program) under the project.

Select the device 'xc9572-15PC84' ,right click and select 'New Source'.



Select 'Verilog Module' from source type window and give the file name without any extension.

Verify the check mark in 'Add to Project' and choose 'Next'.

The enders recondence of the process of the	CANADELET HILE	<ul> <li>Commit Address of State</li> </ul>	
File 245 Year Project Source Per	on has make been the		
CHARLE KOOKL	0 - + + + + + + + + = = = = = = = = = = =	STATE PRINTER	
Design → Write: ● ② beplementation ○  be ﷺ Herarchy = □ m2572.139034	e		
In the second terms of terms of the second terms of	Select Source Type Descreation type, By near and to location. Schematic Unit Document Webby Rest Future WHD, Module WHD, Module WHD, Module WHD, Eachge WHD, Test Bench	Flemene: Loodon Di Vale	
Charl Mr. Darlan - Charles - D. Bar	Rosalath	Weight a project.	
Mar Not To to To the			1000 P. 1000
4 11 Conside Conside Conside Office new location to the project	🙀 Faul in Film Results		**************************************
🥱 é 📋 o	🔞 🖪 🖻 🖉 🚺		Star Pr. 4 % R 4 434 PM

ISE Project Navigator (P.20131013) - Dride	emoldemolaise	inp at or o mp		p		000
fire first laws from Monte Por	ner tek Winne Look ing					
C S E S I D I X D D X	10 04 × 1 7 2 2 3	AGET AL PR	1225	Q.		
1927 ++    Vesi & 🛱 Inplementation 🗇 🛃 Se    Hierarcha	G New Source Historie					
e Constant and States	Define Module Seenty ports for module. Module neral: amt2					
toring the Depict network of the using the Depict network and by using the Depict network of Libraries parvels.	Post Narra a,8 Y	Direction input • input • input •	Bin MSB	LSH -		
No Processes Running		• NIPS	10 	-		
No single design module is selected.		bupi     bupi	2 5 5 5 5 6			
	Mare 31fa		liest	Cancel		
Start 💐 Design 🖸 Files 🔯 🕅	996		-			
rook						+D1
and the second						
Conseil O Thron 1 Werings	A Proin Res Tesuits					
🧿 (ĉ 📋 💿	🔞 😕 🔄 /2	0 >			1	

Write the port signal names ,select input or output from pull down menu and choose 'Next'.

Verify the source details in the summary and choose 'Finish'.

SE Project Nan gator (P.20131013) - Dry	erro/denixia	
The Lat New Print Source Pr	uns bale Rindon Laurat Pag	
COBRE SOCK	NO - PARAL TARA TARA PROVIDE 9	
Cesign +4	m	Mark Land
West # 13 Indementation () 20 Se		
Heanthy = 0 +6972-159084	G here Source Warnet.	
= Enuty View	summary	
The size surrently contrains no f	Project live/gator will create a new skelemet approx with the following specifications.	
tou can add flee to the proved.     using the toother as left, coment     that the Project remu, and the     using the Coarsy, Hies, and     Literates paralle.	Add to Angent Peri Source Transform Children Source Transform Hellow Source Transform Hellow Module owner and Tr Part Definitioner auß Peri	
P () To Process Surving	v An Argot	
112 No single design module is solicited 12 Design Utilities 13 14 15 15 15 15 15 15 15 15 15 15		
	Hare Strik	Const
Start At Desgr C Res C		
000001e		
+ 1		
I Console O Erran L Harmings	🙊 Find in File: Faluits	
Add a new source to the project		
🔊 E 🛄 🛛	🔌 🖪 📾 🛤 🦁 🍃	420 PM

Verilog source file'and2.v' will be opened with skeleton. Edit the program and save.



EE Project Navigator (P.20131013) - Dridemolde	idema.nize-[and2x]	
File Edit View Project Source Process	es Tools Window Layout Help	- 513
Start         ••□ ∉ ×           Start         ••□ ∉ ×           Werx         ● ∰ indemension •         ● Starts           Hearthy         ••□ ∉ ×         ●           ● Ø anno         ••○ ∉ ×         ●           ● Ø ann	X = 1 cimercele lns / lps = E 2 //////////////////////////////////	
No Processes Running	0 16 // Serision: 17 // Revision 0.01 - File Created	
Processes and2	18 // Additional Comments:	
E     Design Utilities       B     Design Utilities	18 // 29 ////////////////////////////////////	
Start = Desgr C Files C Ubranes	is and 2 v 🔯 🛣 Design Summary 🛄	
ansole		+D8>
/ INFO:BDLCompiler:1845 - Analyzin / INFO:ProjectMgmt - Parsing desig	ing Verilog file "D:/demo/and2.9" into library work ign hierarchy completed successfolly.	

Select the 'and2.v' in project window and select Implementation Design---Synthesize XST ----- Check syntax.(Double click).



the Column Date

Check for any errors in the console window. If any error is shown ,edit the program and do the corrections until 'process' check syntax' completed successfully.

Bit Project Numpeter 9(.20131015) - Ortebranderpasse - [and 2.0]	0.0
😭 File Lafé Ween Project Source Process Tools Window Layout Halp	1,161*
Orgen     +0.00     +0.00     +0.00       Wext + 10 reference     +0.00     +0.00     +0.00       Henrothy     -0.00     -0.00     +0.00       Henrothy     -0.00     -0.00<	
Image: Second	
Fart at page () for () the () there () and () and ()	
Consts Frocess *Check System* completed successfully + Const Const 1 Martnar 1 Plan Results	+03× - - -
	Ln 25 Col II Vertion

To Create a test bench Program and Perform the simulation:

3	File Edit Ways Project	t Source Process Tools Wind	ow Lejout Help	(atera)
ΞĒ.	1000 - X	00×(00) + 22	2012年1日 B B B B B B B B B B B B B B B B B B B	
1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1	Venc # [2] Statementate     Hexardsy     @ dems     # 25572159634     # 26572159634		Uneersite Ine / Ipe // Company //	
1		Add Source	Taropet Hansi Taropet Devices	
ON IN		Open 🔐 Remove	Description: Descodencient	
*	() The Processes Running	Marvad Compile Order	Revision: Antinion 0.01 - File Created	
75	Processes and	🕨 Implement Top Module	Saditions1 Commants)	
1	11-2 Design Utilities	e Frie/Path Criptoy +	Anternational anternation and anternation and anternation and anternation and anternation and anternation and a	
	Construction     C	E Expand All Collapse All	imput s,b, output y	
	Checks	s Mi Frei	nip ye sib;	
	Ca Fit Generata Pr	<ul> <li>Design Properties</li> <li>Source Properties</li> </ul>	3000119	
	Stive at Desgr	Ris Clubrates	andZ v 🖸	
Den	ander .			******
	Launiching Design S: Started : "Launichin	unmary/Report Viewer ng ISE Text Editor to min	t and2.v*;	* (11) *
1	County O Dawn 1	Barroos & God or Flar Data dr.		
ñ		County I wanted and		Ln JS Col 3 Verting
1	🖣 (é 📋	0 🔕 🖪		A THE P - 4 TH B 41 244 PM

Select the design program from Project window, right click , select 'New source'.

EX Project Namples (P.2015101.5) - Order	mostermane - (estita)		-	
D. TRUE VOID NO		3333 88 83 8 9		Celler.
	a a contract of the second sec	and has	0	
Vec. # [2] Indementan () [2] Set     Vec. # [2] Indementan () [2] Set     Vec. # [2] Indementan	New Source Woard     Select Source Type     Select source     Select     Select source     Select		annhannn	
The Processes Ramme     Processes and     Proceses and     Proceses and     Processes     Processes and     Process	Weiter (Weiter) Weiter, Lawry Weiter, Dankage Weiter, Lawry Weiter, Markage Weiter, Lawry	Plat name: Lacabani (Drojeve)	manananan	
Vers FTL Schemens Warn Rechestory Set Cleack System Cleack System Fit Ganavatic Programming Constraint Programming Constraint Score	- Hare Strike	If we wanter	ļ	
a real of pools (c) use ( f) as	100			
Launching Design Summary/Rep Started : "Launching ISE Tex	cet Viewer n Editor to mdit andl.y*,			****
Lonoole O Errere S Harrings	👸 Find in File: Faluita			Ln 25 Col 1 Vertice
🥱 倽 🛄 🐧	🔕 🔼 🔯 🕼	🧿 🔤 🍃	<b>4</b> :	245 PM

Select source type as 'Verilog Test Fixture' and give file name for the test bench program.

Choose Next---Next --- and Finish.

Change the view above Project window from 'Implementation' to 'Simulation' mode.

File Edit Ways Project Source Process	Tools	Window Leyout Help				Liteta
Della Annxies		//20/2038	TO PR PIL	9		
Definition     +-C (f) (f) (k)       Verson:     20 trajementation (k) (k)       Extension     (k)       Extension	EE = = = = = = = = = = = = = = = = = =	25 ////////////////////////////////////	Duit Hider Test (UTT)	ana	nanan	
> () To Propert fumme	9	58 +¥ (¥)				
No single design module is selected.		<pre>44 initial begin 45 // Initialize Trg 45 a = 0; 46 b = 0; 46 // Heim 100 ne f; 47 4200; </pre>	nuta ny glamal sevet to fini	***		
		es () // ADD stimulus ) ()	with w			
	104	0.0.				
Fibri M Doop C Res C Utraves	B)	and2.e	0.0	and parts	0	
Contole						*DSX
L'INFO: RDLCompiler:1045 - Analyzing & INFO: RDLCompiler:1045 - Analyzing	Veril	og file "Di/demo/and2.v" in og file "Di/demi/and test.v	ete likeary work e* into likeary work			
()INFOrProjectNumb - Parsing design	biere	uchy completed automatally	Ť.			
+ Las Manager						
📗 Corosia 🔕 Errara 🔝 Marriage 🖓 Sa	et in Files I	Radia Atri				La 1 Col 1 Veries
🔊 lê 📋 🛛 🍯		8 🙆 🖾 🥱	181 🗩			A THE P 14 A R 11 247PM

In the skeleton of test bench program, do the changes in the input test vectors inside the initial block.(Or the necessary changes according to the program requirement).

ER Project Nampster (P.20151011) -Underendermanne	[and (mint)]	A COMPANY OF			000
File Edit View Project Source Process Tools.	Weidow Leyout Help				aleis.
CORP. SOCK PART	##80/2 3 B	5 TO 24 10 10	6 8		
Cenign + C (F × 4) Venc: C (2) Statementation (8) (2) Statistics (8) Statement	11 ///////////////////////////////////	mmmmm	ananananana		-
Heardy ****	26         // Imputus           28         // Imputus           28         prog au           30         tread but           31         // Imputus           32         winne gr           33         // Imputus           34         // Imputus           35         wind work (           36         -#100.	e Unine Test (1977)			
P () Tas Processes Fuerting	18 +219) 20 II				
No single design module is selected.	40 41 initial begin 42 // Initiality Tupit 43 & * 0) 44 h * 0; 45 & 471 article 0; 46 & 421 article 0; 47 & 401 article 0; 48 & 401 article 0; 48 & 401 article 0; 49 & 401 article 0; 49 & 401 article 0; 40 &				*
	49 804				
	28 million and a second				
🖌 Stort 24 Desgr 💽 Res 🖸 Litzaries 😭	and2.+	60 68	and part of	0	
Contable				160	** <b>D</b> <i>S</i> ×
A THFO:HDC/mplie:1445 - Analyzing Yers A THFO:FrojeotHom - Parsing design bler Conste Strum (1) Harmon (1) Forberfam	log file "Di/bend/and_test brohy completed successfall Panda	v" into library sock Y:			
and the second second		1			Lin 48 Col 54 Vertica
🔗 😂 🔛 🗕 📦 I	A 🖻 🖪 🧿			*	Te ∰ ™ at % @ 41 251 PM

Perform the 'Behavioral check syntax' for any errors. If no errors, then choose 'Simulate Behavioral Model'.

52 Project Nampelor (P.20131011) - Octoberrand	ernalates - 1	and texts?	and the second second			CHO
The bds New Project Source Proces	e Tools	Window Layout Help				.tele
Della ZOOXIA:		* # 8 8 * 8 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1	170 P8 201	6 8		
Cesage + C # ×	8	14 ////////////////////////////////////				
History His	12 2 2 2 (MININ	56 17 // Constant 18 /P cop Br 19 // Outputs 13 // Outputs 13 // Outputs 14 // Distantiates the Unit 15 // Auto 1/ 16 // Auto 1/ 16 // Auto 1/ 16 // Auto 1/ 16 // Auto 1/ 17 // Distantiates the Unit 16 // Auto 1/ 17 // Distantiates the Unit 18 // Distantiates the Unit 19 // Distantiates th	t Dames Teac (TVT)			
Ore threeses funning      Processes and yes      Sen Senator      Defaultion Check System      Defaultion Check System      Tendom Titles and Model	00	00 (919) 10 () 11 () 12 () 13 () 14 () 15 () 15 () 16 () 16 () 17 () 16 () 16 () 17 () 16 () 17 () 16 () 16 () 17 () 16 () 16 () 16 () 17 () 16 () 17 () 16 ()				
Start at texp O fin   C throws	6	and2.v	0 13	and test a	0	
Consider			1.000			+0.6×
1 ENFO:HELC-mpiler:1845 - Analytia 1 ENFO:ProjectHyst - Parsing Hesig 1 T 1 T 1 Const O Fins <u>1</u> Harmon N F	ng Verila n Sleven	og fils "D:/demo/and_test. nnhy oospleted successfall	τ* into isbrary wor≷ γ,			
	-		10 CO 10 CO		_	Ln 48 Col 34 Winlog
1 C 🖺 🛛 🍯			四 🥜			253 PM

Isim window will be opened and open "default.wfcg" tab to view simulation result.



#### Select 'Zoom to full view'.

Eim (P 20030003	- (Defeationity)			and the second second	_	COLO HILL
fite fit ihre	w Simulation Window	Legrat Hole				aleta
00000	X D D X O I	GILMM	SEDE 28 220	■● ● ● ● ● ● ●	1 II ≥ 1 1,000 € € II	Gitsland
Interes += II # >	C. JR			Zoom to Full View	200,000	
CIMINIS .	10	101.5	The local and the local states in	treatment - The Same	- I'm lest an	
Instance and Proces	B	Note -				the second the second the second the
= 🔛 and test	12					
the line	0					
1 D pts	0					
	100					
	- the					
	1					
	3					
	3.					
	EL					
	(20)					
	100					
			11: 260, 200 pm			
Advant 2	1 N	6.4	1.5			(j. 7
if beier. Rep		Default with	0 8	wd_tex/v	(C)	
Canade						+- D Ø K
WARFERING: A Rul Time L	Jame proof to the backed su	f due to the locure listed above. Plea	ee use Klinx License Cardigutation Manage	r to for Niese Jacans in order to check ou	ut a ful Sim Resse.	
This is a Life version of	Sin.	The Caulor and the second of the second	Auntificial die 1940 processen die n	Re and the rule renames.		
Time resolution is 2 ps. Similar is stored on all	Contraction processo					
Financed or sub initiation	itice process.	Prove Bi				8
15em>	( 125. 8,08550301 EBD)	LIGE III				
Coresia 1	Completiol Log 🔹 Sect	spente 🔥 finitie Results	a Seath Readly			
Set view such that the	te entire contents la visible					Serr Terre: 360,000 ps
A A	1000	A B 6	10 6 10	100 I 100		256 PM
			1 100	COLUMN TO A DECIMAL OF THE OWNER		11/11/2001

Eim (P 20030003	0 - (Defeat write)			THE OWNER OF TAXABLE				IC IC IN IN
fie fit Do	w Simulation Wind	ion Legal Help			1100 110	2		-102m
00000	XOOX®	OT DR M NO	BRIDE AR	アクロア 岡 山	コート・コート	\$ 1.00 × 4 II GA	Related	
Destarce ++ Cl # 3	x p							250.000 mi
(I) IM IPS (2)	1	Webs	1000	70	1170 m	UNDER .	000 m	1700.000
Instance and Proce	8	- Distance						
m 🔛 and test	1							
La ballat 41	0							
II 🔛 pits	•							
	100							
	1							
	1							
	100							
	120							
	000							
			11: 250, 000 m					
April 2 3		1.1	1.5	1.H.	-10			¥ +
in betere . HE + 1		Defealt with		eid,	181/v	01		
Canade	1122				Do'derroy and Just a			+-□ <i>0</i> ×
WARSTONG: A Ad High I WARSTONG: Dist off to:	Upmue monot he chacked number radio. Nesse refe	d out due to the locure listed above. Plea in the flam documentation for more of	ee use Kinx License Configuratio Ionation on the differences betw	in Manager to Sx Newe Jakamin weni Rwitche and New Pull version	order to check out a full film live	CDP.		-
The is a Life iversion of	/Se.		AUTHOR DECEMPTICAL SERVICE					
Simulatur is storig circu	and a state of the							
Stopped at tive 250	re i file D./Research Ins	8AC000 11						
15ems	Canadalitati una 🗸 👼 👘	manualty M. Ford & Fire Republy	a lead leads					
	Contraction of the second	Contract Management						Sen Time: 260,000 ps
A 6	19462	A 8 5	10 00 10	A STATE		-		257 FM
							and the second second	17/23/2020

Place the marker at different inputs and verify the output.

## III. To Create .UCF file for pin assignment.

Change the view from 'Simulation' to 'Implementation' (above the project window). Select the design program by just one click and choose User constraints----Floor Planning-IO Pre synthesis.

EX Project Nampater (220151013) -Undersonder	The Indian sector is a sector of the sector	
file Edit New Project Source Process	Tools Window Layout Help	leiela
Della Lanaxiee	· / / 20/2 3 5 5 7 / 20 > 2 / 2	
Vence     Image: The second seco	1 Clamencal Tar (10 (20)) 1 Clamencal Tar (10) 2 // Company: 4 // Englined: 4 // Englined: 5 // Company: 4 // Englined: 5 // Company: 4 // Englined: 5 // Company: 4 // Englined: 5 // Company: 5 // Company: 6 // Englined: 6 // Englined: 7 // Englined: 8 // Englined: 8 // Englined: 9 /	
An Processes Survey      Processes and     Processes and     Desgn USBlee      Desgn USBlee      Desgn USB Contracts     Contrats     Contracts     Contrats     Contracts     Contracts     Cont	<pre>15 // Revision 0.01 - File Crested 15 // Revision 0.01 - File Crested 16 // Revision 0.01 - File Crested 16 // Revision 0.01 - File Crested 16 // Revision 0.01 - File Crested 17 // Revision 0.01 - File Crested 18 // Revision 0.01 - File Crested 19 // Revision 0.01 - File Crested 19 // Revision 0.01 - File Crested 19 // Revision 0.01 - File Crested 10 // Revis</pre>	
Stat at Desta C tits C reverse	0 end2+ O	
Controlle		**************************************
Launching Design Summary/Seport Vi Started : "Launching ISE Text Edit Couck O Snew & Harnoge & Sne Couck Start	seb2, oz to mdit shd2.v*,	
		En JI Col 1 Verlog
🤏 (C) 🔛 🛛 🕹		A THE P. 20 IN 8 11 13/13/201

Choose 'yes'.

EX Project New palor (P-2015101.) - Or demonstrations are - Jan		
2 744 Lik Har Provid Lines Franks, Back W	die Land Mitt	lateta
Della Kaaxiea + 2	28828 BBBB 28 28 28 9	
Design	Cuerencial in f / fpp	
a first and meres and first an annual and	and a second sec	
Conside		******
Started + "Flocoplans IO - Fre-Synthesis". Frepering flocoplaness launch script. 4		in 31 Cell Westing
🐨 é 🐃 o 💩 🖪		P

In the Xilinx PACE window, do the pin assignment according to the requirement and save, select ok and close the file. By doing this, and 2.ucf file would have created and saved under the design file.





With 'Edit constraints' option, we can open and edit the .UCF file.

Select the design program in project window and double click on'Inplementation design' and wait till we get the message as Process "Generate Programming file" completed successfully.



- 152 Project Nampater @ 201 stoth -Uralere	ratemaas - 10 aarmaa	no. Joon Atuantionet	tare i	And in case of the local division of the loc				10 M	Court Area
File Edit Wey Project Source Pro	casa Toola Window	Leyout Help							la Hela
Delevis Socxie	0 × 778	20 1 2 2 3	BBBBB	PER V					
Cesign + C Ø	×.							-	
Vent # 🛐 Statementature 🔿 🛃 Smilete	per .								
Hierarchy									
and a set of									
									÷
2 10 Process forms									
Processes and?	E								
Design Dummary/Reports	1								
Coaste Terring Constraints								unrtion Block Inputs Used	
= tan Implement Design	1000							2/144 (2%)	
Wen RTL Schematic Wen RTL Schematic Wen Rthiology Sche	MOL M				PIN BE	SOCIECES			
Check Syntax	-		Siraal Tyne	Required	Manned	Pin Type	Thed	Total	+
🖌 Start 📲 Design 💽 Ries 🖸 Litrar	66 (E)	andQ/v	0	widdlast.	U 🔪 a	Werner word 2_http://///icecsketter/Criter	0		
Contaile									HDSX
Copyright (c) 1998-2010 Kiline	e. Inc. All sight	a reserved.							+
Erocess "Generate Programming	file" completes a	Socessfully							1.00
41 10 11									
I Consis O Error 1 Harrings &	First in Files Raturds								
Done									
A 199 N	A 10 1	100	20 I I W I					and the second	A10 PM
			- HE -	<i></i>				17	(73/2008

Doule click on 'Configure Target device' for dumping the code on CPLD chip and choose 'ok'.

ER Project Nampster (P.201 \$101.0) -Understrate	-	a (Duamatana) (militrarpla	vet.tem]	States and address of	-			Const Long
File Edit Wew Project Source Process	To	ols. Window Layout Help						i la i tel a
G BB - KOOXIGS	10	· / / 8 2 / 2 🖂	BBRR P	P PIE P				
Cesign → CI @ X	0							
Vec: # [2] Sedementator () [2] Sedementator Heavely Construction of the sedementator Construction of the sedementation of the sedementatio	N N WE CO							
P      Processes Survey     Processes and							saction Block Isputs Used 2/144 (2%)	
Optional Implementation		The cost of the cost			PIN RESOURCES		<i></i>	
· Start Mt Deser Co Bas Ch Literate	E	anD /	Sireal Type	Received	Manned Pin Type	Thed	Total	17.
Controls								100 TH # 1
Copyraght (s) 1985-2013 Kilins, Frocess "Generate Frogramming fi 4 Conste C fine K Manage & 9	Inc. 1e*	All rights reserved. completed stocensfully NerBaude						
Done							_	
🔏 (é 📋 🛛 🍯			9 B	*		4.5	******	152 PM

Xilinx ISE IMPACT window will be opened, select 'Boundary scan'. Right click on the workspace and select 'Initialise the chain'.



Browse the file name with .jed extension and assign to the IC icon.

Chat of		
Hiat	Your Rightawark References Manange Review West Octon	. W
C Sto	THE TWO ROOMS THE TAKE OF BEER TO BE THE TO BE AND A AND CHARDE AN	alline Autored A Alline -
7. 110	THE ANYACT (F 2013)0130 - (Ricontary Scar)	California States
Carter	He File Ealt View Operations Subjust Debug Window Help	LINE COMP
5		
-	APPACT These ++ C & X apple disk denier to other spectrees	
	Boundary Seam     Soundary Seam     Soundary Seam     Soundary Seam     Total PROMATING (PROMATING Format)     Total     Soundary Seam     Soundary Sea	
	APACT Recomment In The R	
a contractor	Bandar Gon	
- C		337 FM

Select the icon and right click, choose' Program'.

Canal I	B - C - A C - C - C - C - C - C - C - C -			in isan	×
(C) (	The two we have the two the two two of the two two of the two two of the two two have the two two have the two	dinceD	A	Miring *	~
Tarts J ton	an Exercise 2013/01/2014/2014	-	Change - Styles *	Se Sent	
Conte	le file fat Vew Operators Disput Debug Window Help	11	1	Classing	-
-	098 0 2 2 4 2 4 2 4 2 4 2 4 2 4 2 4 2 4 2 4				1
	With Trave     Image: Second and Second				
	MPACT Processes → □ S ×				
	Available Operations are				
Page 11 of 11	Bundley Kan		8   100% (C)		120
🐴 (ê		3.2	* * *	2 1 357	750



After getting 'Programming succeeded", feed the inputs and check the ouput in LEDs in hardware domain.

# PART-A

## E.No:1.a:

**<u>AIM</u>**: Write Verilog program for 2 to 4 decoder realization using NAND gates only (structural model) along with test bench to verify the design

## **Block Diagram:**



## Truth Table:

En	INF	PUT	OUTPUT						
	A1	A0	D3	D2	D1	D0			
1	Х	Х	1	1	1	1			
0	0	0	1	1	1	0			
0	0	1	1	1	0	1			
0	1	0	1	0	1	1			
0	1	1	0	1	1	1			

# <u>Circuit diagram:</u>



#### **Program**

```
module decoder_24(
input E,
input [1:0] A,
output [3:0] D
);
```

wire E\_bar, A0\_bar, A1\_bar;

not not\_01(E\_bar, E); not not\_02(A0\_bar, A[0]); not not\_03(A1\_bar, A[1]);

nand nand\_01(D[0], A0\_bar, A1\_bar, E\_bar); nand nand\_02(D[1], A[0], A1\_bar, E\_bar); nand nand\_03(D[2], A0\_bar, A[1], E\_bar); nand nand\_04(D[3], A[0], A[1], E\_bar);

endmodule

#### Test bench:

module TB\_decoder\_24;

```
// Inputs
reg E;
reg [1:0] A;
// Outputs
wire [3:0] D;
// Instantiate the Unit Under Test (UUT)
decoder_24 uut (
       .E(E),
       .A(A),
       .D(D)
);
initial begin
       // Initialize Inputs
       E = 0;
       A = 0; // A = 00 in binary
       // Wait 100 ns for global reset to finish
       #100;
```

A = 1; // A= 01 in binary #100; A = 2; // A= 10 in binary #100; A = 3; // A= 11 in binary #100; E = 1; A = 2; // A= 11 in binary #100;

\$finish;

end

endmodule

## **Output waveform**

			112.000 ns				
Name	Value	0 ns	100 ns	200 ns	300 ns	400 ns	500 ns
▼ <mark>&gt;</mark> D[3:0] 1 (a) 1 (a) 1 (a) 1 (a) 1 (a) 1 (b) 1 (a) 1 (b) 1 (c) 1 (c	1101 1 0 1 0				X 0111	X 1111	
V N(1:0]	01 0 1	( <u>00</u>			<u>× 11</u>		

## Exp.No:1.b:

**<u>AIM</u>**: Write Verilog program for 8 to 3 encoder with priority and without priority (behavioral model) along with test bench to verify the design.

## **Block Diagram:**



## **Encoder Without Priority:**

## Truth Table:

INPUT										OUTPUT		
EN	D7	D6	D5	D4	D3	D2	D1	D0	Y2	Y1	Y0	
0	Х	Х	Х	Х	Х	Х	Х	Х	Z	Z	Z	
1	0	0	0	0	0	0	0	1	0	0	0	
1	0	0	0	0	0	0	1	0	0	0	1	
1	0	0	0	0	0	1	0	0	0	1	0	
1	0	0	0	0	1	0	0	0	0	1	1	
1	0	0	0	1	0	0	0	0	1	0	0	
1	0	0	1	0	0	0	0	0	1	0	1	
1	0	1	0	0	0	0	0	0	1	1	0	
1	1	0	0	0	0	0	0	0	1	1	1	

1	0	0	0	0	0	0	0	0	Х	Х	Х

**Program:** 

```
module encoder_83(
  input en,
  input [7:0] D,
  output reg [2:0] y
  );
always@(en,D)
       begin
              if(en)
                     case(D)
                     8'd1
                            : y=3'b000;
                     8'd2
                            : y=3'b001;
                            : y=3'b010;
                     8'd4
                            : y=3'b011;
                     8'd8
                     8'd16 : y=3'b100;
                     8'd32 : y=3'b101;
                     8'd64 : y=3'b110;
                     8'd128: y=3'b111;
                     default: y=3bxxx;
                     endcase
              else
                     y=3'bzzz;
```

end	//end	always
-----	-------	--------

end module

## **Test Bench:**

module encoderWOP\_83\_TB;

en = 0; D = 0; #50;
en = 1; D = 0; #50;
D = 1; #50;
D = 2; #50;
D = 4; #50;
D = 8; #50;
D =16; #50;
D = 32; #50;
D = 64; #50;
D = 128; #50;
D = 50; #50;
D = 75; #50;

end endmodule

\$finish;

## **Result**

Sim (P.201310)	13) - [Default	t.wcfg]								الر کار ک	dista :		
ile Edit Vie	w Simulat	ion Wir	ndow	Layout	Help								
28	× 🖻 🖨 ×		CH A		10	i 🖻 E		PA	? 🍺	PB.	- 3	生土	110
		0.000 ns											6
Name	Value	0 ns	Tanad	100 ns	mana	200 ns		300 ns	พมษะที่	400 ns	nované	500 ns	6
▼ 🛃 y[2:0]	ZZZ	ZZZ	XXX	000	001	010	011	100	101	110	( 111		xxx
10, [2]	z	-										-	
UL [1]	z							L					
16 (0)	z									<u> </u>			
🐚 en	0												_
▼ 📢 D[7:0]	00000000	0000	0000	0000	0000	0000	0000	0001	0010	0100	(1000)	0011	0100
[7]	0												
16]	0												
14 [5]	0		_	-	_					<u> </u>	_		
[4]	0	_		-			_						
131	0			-								<u> </u>	
16 [2]	0						L						
11	0					<u> </u>							_
[0]	0					<u> </u>				<u> </u>		<u> </u>	

# Encoder With Priority: Truth Table:

	INPUT										OUTPUT		
EN	D7	D6	D5	D4	D3	D2	D1	D0	Y2	Y1	Y0		
0	Х	Х	Х	Х	Х	Х	Х	Х	Z	Z	Z		
1	0	0	0	0	0	0	0	1	0	0	0		
1	0	0	0	0	0	0	1	Х	0	0	1		
1	0	0	0	0	0	1	Х	Х	0	1	0		
1	0	0	0	0	1	Х	Х	Х	0	1	1		
1	0	0	0	1	Х	Х	Х	Х	1	0	0		
1	0	0	1	Х	Х	Х	Х	Х	1	0	1		
1	0	1	Х	Х	х	Х	Х	х	1	1	0		
1	1	Х	Х	Х	Х	Х	Х	Х	1	1	1		
1	0	0	0	0	0	0	0	0	Х	Х	Х		

Program:

```
module encoder83_WP(
    input en,
    input [7:0] D,
    output reg [2:0] y
    );
always@(en,D)
```

begin

if(en)

casex(D)11111111

```
8'b0000_0001: y=3'b000;
8'b0000_001x: y=3'b001;
8'b0000_01xx: y=3'b010;
8'b0000_1xxx: y=3'b011;
8'b0001_xxxx: y=3'b100;
8'b001x_xxxx: y=3'b101;
8'b01xx_xxxx: y=3'b110;
8'b1xxx_xxxx: y=3'b111;
default : y=3'bxxx;
endcase
```

else

y=3'bZZZ;

end

end module

### Test Bench:

module TB\_encoder83\_WP;

// Inputs
reg en;
reg [7:0] D;

// Outputs
wire [2:0] y;

```
// Instantiate the Unit Under Test (UUT)
encoder83_WP uut (
                .en(en),
                .D(D),
               .y(y)
);
```
initial b	begin		
	en = 0;		
	D = 0:		
	#50.	// Add stimulus	here
	11.50,	// Tuu stimulus	nere
	an = 1		
	$\mathcal{C} = 1,$		
	D = 0;		
	#50;		
	-		
	D = 1;		
	#50;		
	D = 2;		
	#50;		
	,		
	D = 4:		
	#50·		
	1150,		
	$D = 8 \cdot$		
	D = 0,		
	#30,		
	D -16		
	D = 10,		
	#50;		
	D 00		
	D = 32;		
	#50;		
	D = 64;		
	#50;		
	D = 128;		
	#50:		
	D = 50		
	B = 50,		
	11.50,		
	D - 75.		
	D = 73;		
	#50;		
	<b>•</b> • • • •		
	\$finish;		
end			
endmodule	•		

# Result Waveform:

			550 rrs	X	
			500 ns	101	01001100
			450 ns		
		Re-launch	50 US	110	00000010
		1.00us 🖌 🍋	50 m2	101	
ļ		M + 0	0 1 1	8	
		14 - 14 - 14 - 14 - 14 - 14 - 14 - 14 -	€ sug	110	
		2 4 8	0 ns	010	0010000
1000		1 2 2 1	0 ns	8	
ALC: NO		б Н Ш Ю	0 ns   15	8	
	syout Help	0118	ns 10	×	
cfo]	Window La	M 2 4	50	222	000000
3) - [Default.w	w Simulation	N A A A X	Value	110 1 1 1	
im (P.2013101	le Edit Vier	S 10 10	Name	10 21 10 10 10 10 10 10 10 10 10 10 10 10 10	10 en 10 en 10 en 11 en 11 en 12 en 13 en 14 en 14 en 15 en 16 en 1
N IS	E		10 m 10		· 1 本 4 ~ 日間回日

#### HDL Lab Manual

Exp.No:1.c:

**<u>AIM</u>**: Write Verilog program for 8 to 1 multiplexer using case statement and if statements along with test bench to verify the design

#### **Block Diagram:**



Truth Table:

Е		Y		
2	S2			
0	X	X	X	Z
1	0	0	0	10
1	0	0	1	I1
1	0	1	0	I2
1	0	1	1	I3
1	1	0	0	I4
1	1	0	1	15
1	1	1	0	<b>I</b> 6
1	1	1	1	I7



```
Program using case statement.
module mux81_case(
  input en,
  input [7:0] i,
  input [2:0] sel,
  output reg y
  );
always@(en,i,sel)
       begin
       if(en)
              case(sel)
              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=1'bx;
               endcase
       else
              y=1'bz;
       end
```

endmodule

#### Test bench:

module TB\_mux81\_case;

```
// Inputs
    reg en;
    reg [7:0] i;
    reg [2:0] sel;
    // Outputs
    wire y;
    // Instantiate the Unit Under Test (UUT)
    mux81_case uut (
            .en(en),
            .i(i),
            .sel(sel),
            .y(y)
    );
    initial begin
            // Initialize Inputs
            en = 0;
            i = 0;
            sel = 0;
            // Wait 50 ns for global reset to finish
            #50;
            en = 1;
            i = 8'b10101010;
            sel = 3'b000;
            #50 \text{ sel} = 1;
            #50 \text{ sel} = 2;
            #50 \text{ sel} = 3;
            #50 \text{ sel} = 4;
            #50; sel = 5;
            #50; sel = 6;
            #50; sel = 7;
            #50; sel = 8;
            #50;
            $finish;
            end
endmodule
```

#### Output waveform



**Exp.No:2.** Model in Verilog for a full adder and add functionality to perform logical operations of XOR, XNOR, AND and OR gates. Write test bench with appropriate input patterns to verify the modeled behaviour.

Aim: To model the full adder circuit along with and,or,xor and xnor functions. Logic Symbol: sum



#### **Program:**

module FA\_Gates(
 input a,b,c,
 output sum,cout,
 output xor\_op,xnor\_op,andop,orop
 );
assign sum= xor\_op^c;
assign cout=(andop)|(b&c)|(c&a);
assign xor\_op=a^b;
assign xnor\_op=-xor\_op;
assign andop=a&b;
assign orop=a|b;
endmodule

#### **Test Bench Program:**

'timescale 1ns/1ps module FA\_test;

// Inputs
reg a;
reg b;
reg c;
// Outputs
wire sum;
wire cout;
wire xor\_op;
wire andop;
wire orop;

// Instantiate the Unit Under Test (UUT)

```
FA_Gates uut (.a(a),.b(b), .c(c), .sum(sum), .cout(cout), .xor_op(xor_op), .xnor_op(xnor_op), .andop(andop), .orop(orop));
```

initial begin // Initialize Inputs a = 0; b = 0; c = 0; #20 a=1;b=0;c=0; #20 a=0;b=1;c=0; #20 a=0;b=1;c=0; #20 a=1;b=1;c=0; #20 a=1;b=0;c=1; #20 a=1;b=1;c=0; #20 a=1;b=1;c=1; #20 a=1;b=1;c=1; #200 \$finish; end endmodule



#### Exp.No:3:

**<u>AIM</u>**: Write a verilog program for 32-bit ALU shown in figure below and verify the functionality of ALU by selecting appropriate test patterns. The functionality of the ALU is presented in Table 1.

- a Write test bench to verify the functionality of the ALU considering all possible input patterns
- b. The enable signal will set the output to required functions if enabled, if disabled all the outputs are set to tri-state
- c. The acknowledge signal is set high after every operation is completed

#### **Block Diagram:**



#### Figure 1 ALU top level block diagram Table 1 ALU Functions

#### res

#### Logic Table:

Opcode(2:0)	ALU Operation	
000	A+B	Addition of two numbers
001	A-B	Subtraction of two numbers
010	A+1	Increment Accumulator by 1
011	A-1	Decrement accumulator by 1
100	A	True
101	A Complement	Complement
110	A OR B	Logic OR
111	A AND B	Logic AND

#### Program:

```
module ALU32(
input en,
input [0:2] op ,
input [31:0] A,
input [31:0] B,
output reg ack,
output reg [32:0] res
);
```

always@(en,op,A,B) begin

if(en)

case(op)

3'b000: res= A+B; 3'b001: res= A-B; 3'b010: res= A+1; 3'b011: res= A-1; 3'b100: res= A; 3'b101: res= A; 3'b110: res= A|B; 3'b111: res= A&B; default: res=33'hxxxxxx; endcase

else

res=33'hzzzzzz;

```
ack = 1'b1;
#5 ack =1'b0;
```

end endmodule

#### **Test Bench:**

module TB\_ALU32;

// Inputs reg en; reg [2:0] op; reg [31:0] A; reg [31:0] B; // Outputs wire [32:0] res; wire ack; // Instantiate the Unit Under Test (UUT) ALU32 uut ( .en(en), .op(op), .A(A), .B(B), .ack(ack), .res(res) );

initial begin // Initialize Inputs en = 0; op = 02; A = 20;B = 10;

// Wait 50 ns for global reset to finish
#50;

// Add stimulus here

```
en = 1;
A = 32'hfff_fff;
B = 32'h2;
op = 0;
#50;
       op = 1;
       op = 2;
#50;
#50;
      op = 3;
#50;
      op = 4;
       op = 5;
#50;
#50;
       op = 6;
#50;
       op = 7;
       op = 3'b1xx;
#50;
#50;
```

\$finish; end endmodule

#### **Result Waveform**



Exp.No:4: i). Write the verilog code for SR-Flip Flop and verify

#### AIM:

To model SR Flip Flop using verilog behavioral description and verify its function using ISE simulator as well as on the CPLD XC9572.

#### **TOOLS REQUIRED:**

i). Software : Xilinx ISE14.7

ii).Hardware: XC9572 based CPLD Kit,JTAG,Power Adapter,Flying Leads.

#### LOGIC SYMBOL:



Truth Table:

CLK	RST	S	R	<b>q</b> <sub>prev</sub>	q	qь
	1	Х	Х	Х	0	1
~						
^	0	0	0	0	0	1
~	0	0	0	1	1	0
~	0	0	1	Х	0	1
<	0	1	0	Х	1	0
~	0	1	1	Х	Indeter	minate
					sta	ate

#### **Behavioral Description:**

```
`timescale 1ns / 1ps

module srff( input clk, input rst, input s, input r, output reg q, output reg

qb );

reg[1:0] sr;

always@(posedge clk)

begin

if(rst==1'b1)

q=1'b0;

else

begin

sr=\{s,r\};
```

```
case (sr)
      2'b00: q=q;
      2'b01: q=1'b0;
      2'b10: q=1'b1;
      2'b11: q=1'bZ;
      default: q=1'bx;
      endcase
  end
 qb=~q;
end
endmodule
Test Bench Program:
`timescale 1ns / 1ps
module srff_test;
reg clk;
reg rst;
reg s;
reg r;
wire q;
wire qb;
srff uut (.clk(clk), .rst(rst), .s(s), .r(r), .q(q), .qb(qb));
initial
begin
               clk = 0;
               rst = 1;
               s = 0;
               r = 0;
               #30 s=0 ;r=0;rst=0;
               #30 s=0;r=1;
               #30 s=1;r=0;
               #30 s=1;r=1;
               #200 $finish;
 end
 always
#10 clk=~clk;
endmodule
User Constraint File:
#PINLOCK_BEGIN
NET "clk"
                 LOC = "S:PIN1";
NET "rst"
                 LOC = "S:PIN2";
NET "s "
               LOC = "S:PIN3";
```

NET "r "	LOC = "S:PIN4";
NET "q"	LOC = "S:PIN5";
NET "qb"	LOC = "S:PIN6";
#PINLOCK_	END
Simulation <b>F</b>	<u>Result:</u>

Name	Value	0 ns	50 ns	100 ns	150 ns	200
le q	1					
dp 🕤	0					
🐻 cik	1					
🌆 rst	0					
Tip z	1					
13 г	0					

#### Exp.No:4.ii). Write the verilog code for JK-Flip Flop and verify

#### AIM:

To model JK Flip Flop using verilog behavioral description and verify its function using ISE simulator as well as on the CPLD XC9572.

#### **TOOLS REQUIRED:**

i). Software : Xilinx ISE14.7

ii).Hardware: XC9572 based CPLD Kit,JTAG,Power Adapter,Flying Leads.

#### LOGIC SYMBOL:



#### **Truth Table:**

CLK	RST	J	K	q <sub>prev</sub>	q	qь
~	1	X	Х	Х	0	1
~	0	0	0	0	0	1
^	0	0	0	1	1	0
^	0	0	1	Х	0	1
^	0	1	0	Х	1	0
^	0	1	1	0	1	0
~	0	1	1	1	0	1

#### **Behavioral Description:**

`timescale 1ns / 1ps module jkff( input clk, input rst, input j, input k, output reg q, output reg qb ); reg[1:0] jk; always@(posedge clk)

```
begin
if(rst==1'b1)
   q=1'b0;
else
  begin
      jk={j,k};
      case (jk)
      2'b00: q=q;
      2'b01: q=1'b0;
      2'b10: q=1'b1;
      2'b11: q=~q;
      default: q=1'bx;
      endcase
  end
qb=~q;
end
endmodule
Test Bench Program:
`timescale 1ns / 1ps
module jkff_test;
reg clk;
reg rst;
reg j;
reg k;
wire q;
wire qb;
jkff uut (.clk(clk), .rst(rst), .j(j), .k(k), .q(q), .qb(qb));
initial
begin
               clk = 0;
               rst = 1;
               j = 0;
               k = 0;
               #30 j=0 ;k=0;rst=0;
               #30j=0;k=1;
               #30 j=1;k=0;
               #30 j=1;k=1;
               #200 $finish;
 end
 always
#10 clk=~clk;
```

### endmodule

<u>User Constr</u>	<u>aint File:</u>								
#PINLOCK	_BEGIN								
NET "clk"	LOC = "S:PIN1";								
NET "rst"	LOC = "S:PIN2";								
NET "j "	LOC = "S:PIN3";								
NET "k "	LOC = "S:PIN4";								
NET "q"	LOC = "S:PIN5";								
NET "qb"	LOC = "S:PIN6";								
<b>#PINLOCK</b>	_END								
Simulation Result:									

							130.000 ns	<mark>8</mark>			
Name	Value	0 ns		50 ns		100 ns		150 ns		200 ns	3
lle q	0										
🗓 qb	1							i			
🕼 clk	1					ШГ			ш		
16 rst	0										
1 🔓 ј	1										
16 к	1										

#### Exp.No.4.iii). Write the verilog code for D-Flip Flop and verify

#### AIM:

To model D\_Flip Flop using verilog behavioral description and verify its function using ISE simulator as well as on the CPLD XC9572.

#### **TOOLS REQUIRED:**

i). Software : Xilinx ISE14.7ii).Hardware: XC9572 based CPLD Kit,JTAG,Power Adapter,Flying Leads.

#### LOGIC SYMBOL:



CLK	RST	D	qprev	q	qь
	1	Х	Х	0	1
~					
~	0	0	X	0	1
~	0	1	X	1	0

`timescale 1ns / 1ps
module dff(clk,rst,d, q,qb);
input clk,rst,d;
output q,qb;
reg q,qb;
always@(posedge clk)
begin
if(rst==1'b1)
q=1'b0;
else

q=d; qb=~q; end endmodule **Test Bench Program:** `timescale 1ns / 1ps module dFF\_test; reg clk; reg rst; reg d; wire q; wire qb; dFF uut (.clk(clk),.rst(rst),.d(d),.q(q),.qb(qb)); initial begin **clk** = **0**; rst = 1; **d** = 0; #30 d=1; #30 rst=0; #30 d=0; #30 d=1; **#100 \$finish;** end always #10 clk=~clk; endmodule **User Constraint File: #PINLOCK\_BEGIN** NET "clk" LOC = "S:PIN1"; NET "rst" LOC = "S:PIN2";NET "d " LOC = "S:PIN3";NET "q" LOC = "S:PIN4";LOC = "S:PIN5";

NET "qb" LC #PINLOCK\_END

#### Simulation Result:

					130.000 n	s
Name Value	0 ns	.	50 ns	100 ns		150 ns
16 q 1						
Ve qb o						
1 1						
🐻 rst 🛛 0						
1 d						

## Exp.No:5. Write a Verilog code for 4-bit BCD synchronous counter and verify.

#### AIM:

To model 4 bit Synchronous BCD counter using verilog and verify its function using ISE simulator as well as on the CPLD XC9572.

#### **TOOLS REQUIRED:**

i). Software : Xilinx ISE14.7ii).Hardware: XC9572 based CPLD Kit,JTAG,Power Adapter,Flying Leads.

#### LOGIC SYMBOL:



BCD Counter

4 bits)

#### **Truth Table:**

CLK	CLR	Q
	1	0000
~		
^	0	0001
^	0	0010
^	0	0011
^	0	0100
~	0	0101
~	0	0110
~	0	0111
^	0	1000
~	0	1001

#### **Behavioral Description:**

`timescale 1ns / 1ps module cntr( input clk, input clr, output reg[3:0] q ); initial q=4'd0; always@(posedge clk) begin if(clr==1) q=4'd0;else if(q==4'b1001)q=4'd0;

q=1 do; else q=q+1;

end endmodule

#### **Test Bench Program:**

```
`timescale 1ns / 1ps
module cntr_test;
reg clk;
reg clr;
wire [3:0] q;
cntr uut ( .clk(clk), .clr(clr), .q(q));
initial
begin
      clk = 0;
      clr = 1;
      #25 clr=0;
       #250 clr=1;
       #25 clr=0;
      #450 $finish;
end
always
      #10 clk=~clk;
endmodule
```

#### **User Constraint File:**

NET "clk" LOC = "P1"; NET "clr" LOC = "P2"; NET "q<0>" LOC = "P3"; NET "q<1>" LOC = "P4"; NET "q<2>" LOC = "P4"; NET "q<3>" LOC = "P5";

#### Simulation Result:

Name	Value	0 ns	50 ns	100 ns	150 ns	200 ns	250 ns	300 ns
🕨 🖓 q[3:0]	0110	0000 0001	0010 0011 01	00 0101 0110	0111 1000 1	001 0000 0001	0010 0011	000 🗙 0001 🕽
🔚 cik	0							
16 dr	0							

# Exp.No:6. Write a Verilog code for counter with given input clock and check whether it works as clock divider performing division of clock by 2, 4, 8 and 16. Verify the functionality of the code.

**AIM:** To model a frequency divider (frequency division by 2,4,8, and 16) using verilog and verify the same using ISE simulator as well as on the CPLD XC9572.

**TOOLS REQUIRED:** i). Software : Xilinx ISE14.7 ii).Hardware: XC9572 based CPLD Kit,JTAG,Power Adapter,Flying Leads.



**CONCEPT:** On every rising edge of the clock, temp variable is incremented by 1.

clk	Temp[3]	Temp[2]	Temp[1]	Temp[0]
	0	0	0	0
<b>≜</b>	0	0	0	1
<b></b>	0	0	1	0
 ▲	0	0	1	1
•	0	1	0	0
<b></b>	0	1	0	1
•	0	1	1	0
•	0	1	1	1
i A	1	0	0	0
	1	0	0	1
<b>≜</b>	1	0	1	0
<b>↑</b>	1	0	1	1
↑	1	1	0	0
↑	1	1	0	1
♠	1	1	1	0
<b>↑</b>	1	1	1	1

clk2=temp[0] Clk4=temp[1] Clk8=temp[2] Clk16=temp[3] **Behavioral Description:** 

```
`timescale 1ns / 1ps
module clkdiv( input clk,output reg clk2, output reg clk4, output reg clk8,
               output reg clk16 );
reg[3:0] temp=4'd0;
always@(posedge clk)
begin
temp=temp+1;
clk2=temp[0];
clk4=temp[1];
clk8=temp[2];
clk16=temp[3];
end
endmodule
Test Bench Program:
`timescale 1ns / 1ps
module clkdiv_test;
reg clk;
wire clk2;
wire clk4:
wire clk8;
wire clk16;
clkdiv uut (.clk(clk), .clk2(clk2), .clk4(clk4), .clk8(clk8), .clk16(clk16));
initial
begin
clk = 0;
#300 $finish;
end
always
#5 clk=~clk;
endmodule
User Constraint File:
NET "clk" LOC = "P1";
NET "clk2" LOC = "P2";
NET "clk4" LOC = "P3" ;
NET ''clk8'' LOC = ''P4'' ;
NET "clk16" LOC = "P5" ;
```

#### Simulation Result:

Name	Value	0 ns	50 ns	100 ns	150 ns	200 ns
le cik2	0					
To cik4	1					
Ug cik8	1					
🔓 cik16	1					
🚡 clk	1					

#### <u>Part –B</u>

# Q1. Interface a DC motor to FPGA and write Verilog code to change its speed and direction.

**AIM:** To Interface a DC motor to FPGA/CPLD and write Verilog code to change its speed and direction.

#### **TOOLS REQUIRED:**

i). Software : Xilinx ISE14.7

**ii).Hardware:** XC9572 based CPLD Kit,JTAG,Power Adapter,Flying Leads,dc motor interfacing board with motor.

#### LOGIC SYMBOL:



#### timescale 1ns / 1ps

module dco( input clk, input [1:0] speed, output reg dco); reg[11:0] clkdiv=12'd0;

```
always@(posedge clk)
begin
clkdiv=clkdiv+1;
if(clkdiv==12'd3000)
clkdiv=12'd0;
end
```

```
always@(speed)
begin
if(clkdiv==12'd0)
dco=1'b1;
else if(speed==2'd0 && clkdiv==12'd700)
dco=1'b0;
else if (speed==2'd1 && clkdiv==12'd1400)
dco=1'b0;
else if (speed==2'd2 && clkdiv==12'd2100)
dco=1'b0;
else if(speed==2'd3 && clkdiv==12'd2800)
dco=1'b0;
end
endmodule
```

User Constraint File: NET "clk" LOC = "P1" ; NET "dco" LOC = "P4" ; NET "speed<0>" LOC = "P2" ; NET "speed<1>" LOC = "P3" ;

#### **EXPECTED RESULT:**

- **1.** By varying the speed signal from 00 to 11,the speed of rotation can be changed.
- 2. The dco ouput can be given to in1 and in2 connection points for changing the direction of rotation.

#### Q2.

**AIM: To** Interface a Stepper motor to FPGA and write Verilog code to control the Stepper motor rotation which in turn may control a Robotic Arm. External switches to be used for different controls like rotate the Stepper motor (i) +N steps if Switch no.1 of a Dip switch is closed (ii) +N/2 steps if Switch no. 2 of a Dip switch is closed (iii) –N steps if Switch no. 3 of a Dip switch is closed etc.

#### **TOOLS REQUIRED:**

i). Software : Xilinx ISE14.7

**ii).Hardware:** XC9572 based CPLD Kit,JTAG,Power Adapter,Flying Leads,stepper motor interfacing board with motor.

#### LOGIC SYMBOL:





#### **ALGORITHM:**

1. Declare the external ports and internal objects.

2. Perform frequency division to control the speed of rotation

3.check for the reset status. If reset is high, reset all the step control variables to zero else go to step 4.

4. Read the status of switch1 .If it is pressed, send the data to stepper motor coil in such a way that it will be moving in forward direction for N steps else go to step 5.

5. Read the status of switch2 .If it is pressed, send the data to stepper motor coil in such a way that it will be moving in reverse direction for N steps else go to step 6.

6. Read the status of switch3. If it is pressed, send the data to the stepper motor coil in such a way that it will be moving in forward direction for (N/2) steps else stop.

[Note: To energize the coils of the stepper motor, send a 4 bit data 0001,0010,0100 and 1000 in this sequence for a forward direction and send these data in a reverse order for anticlockwise direction rotation].

#### **PROGRAM:**

```
`timescale 1ns / 1ps
module stepper( input clk,rst, inout [7:0] kb, output reg[7:0] smdata );
reg tclk;
reg[15:0] clkdiv=16'd0;
reg[1:0]sts=2'd0;
reg[3:0]coil=4'b00001;
reg[7:0] i1=8'd0;
reg[7:0] i2=8'd0;
reg[7:0] i3=8'd0;
```

```
reg[3:0] N=4'd15;
reg[2:0] stdir;
assign kb[7:3]=5'b00001;
always@(posedge clk)
begin
clkdiv=clkdiv+1;
tclk=clkdiv[15];
end
always@(posedge tclk)
begin
if(rst==1)
begin
      i1=0;
      i2=0;
      i3=0;
end
case (kb[2:0])
3'b110:
      if(i1!=N)
      begin
      sts=sts+1;
      i1=i1+1;
      end
3'b101:
      if(i2!=N)
      begin
      sts=sts-1;
      i2=i2+1;
      end
3'b011:
      if(i3!=(N/2))
      begin
      sts=sts+1;
      i3=i3+1;
      end
endcase
end
always@(sts)
begin
```

```
case(sts)

2'b00: coil=4'b0001;

2'b01: coil=4'b0010;

2'b10: coil=4'b0100;

2'b11: coil=4'b1000;

default: coil=4'b0001;

endcase

smdata={4'b0000,coil};

end

endmodule
```

#### **User Constraint File:**

NET "clk" LOC = "P53" ; NET "kb<0>" LOC = "P1" NET "kb<1>" LOC = "P2" NET "kb<2>" LOC = "P3" NET "kb<3>" LOC = "P4" NET "kb<4>" LOC = "P5" NET "kb<5>" LOC = "P6" NET "kb<6>" LOC = "P7" NET "kb<7>" LOC = "P9" ; NET "rst" LOC = "P54"; NET "smdata<0>" LOC = "P10" ; NET "smdata<1>" LOC = "P11" NET "smdata<2>" LOC = "P12" NET "smdata<3>" LOC = "P13" NET "smdata<4>" LOC = "P14" NET "smdata<5>" LOC = "P15" NET "smdata<6>" LOC = "P17" NET "smdata<7>" LOC = "P18" ;

#### **EXPECTED RESULT:**

- 1. Set the number of required steps (N) in the program.
- **2.** By pressing sw1 of 4x4 hexa keypad, stepper motor will rotate in clockwise direction for N steps.
- **3.** By pressing sw2 of 4x4 hexakeypad, stepper motor will rotate in anticlockwise direction for N steps.
- **4.** By pressing sw3 of 4x4 hexakeypad, stepper motor will rotate in clockwise direction for (N/2) steps.

Q3.

AIM: To write Verilog code using FSM to simulate elevator operation. TOOLS REQUIRED:

i). Software : Xilinx ISE14.7

ii).Hardware: XC9572 based CPLD Kit(with Hexa keypad and 7 segment

display), JTAG, Power Adapter, Flying Leads, FRCs.

#### LOGIC SYMBOL:



```
endcase
end
always@(keyhit)
begin
if(keyhit==1'b1)
if(kyrow==4'b1110 && kycol==4'b1110)
kvflr=4'd0:
else if(kyrow==4'b1110 && kycol==4'b1101)
kyflr=4'd1;
else if(kyrow==4'b1110 && kycol==4'b1011)
kyflr=4'd2;
else if(kyrow==4'b1110 && kycol==4'b0111)
kyflr=4'd3;
else if(kyrow==4'b1101 && kycol==4'b1110)
kyflr=4'd4;
else if(kyrow==4'b1101 && kycol==4'b1101)
kyflr=4'd5;
else if(kyrow==4'b1101 && kycol==4'b1011)
kyflr=4'd6;
else if(kyrow==4'b1101 && kycol==4'b0111)
kyflr=4'd7;
else if(kyrow==4'b1011 && kycol==4'b1110)
kyflr=4'd8;
else if(kyrow==4'b1011 && kycol==4'b1101)
kyflr=4'd9;
else if(kyrow==4'b1011 & kycol==4'b1011)
kvflr=4'd10:
else if(kyrow==4'b1011 && kycol==4'b0111)
kyflr=4'd11;
else if(kyrow==4'b0111 && kycol==4'b1110)
kyflr=4'd12;
else if(kyrow==4'b0111 && kycol==4'b1101)
kyflr=4'd13;
else if(kyrow==4'b0111 && kycol==4'b1011)
kyflr=4'd14;
else if(kyrow==4'b0111 && kycol==4'b0111)
kyflr=4'd15;
else
kyflr=kyflr;
```
HDL Lab Manual

end

always@(posedge flrclk) begin if(kyflr>curflr) curflr=curflr+1; else if (kyflr<curflr) curflr=curflr-1; else curflr=curflr; end always@(posedge flrclk) begin dismux=4'b1110; case (curflr) 4'd0: disseg=8'd63; 4'd1: disseg=8'd6;4'd2: disseg=8'd91; 4'd3: disseg=8'd79; 4'd4: disseg=8'd102; 4'd5: disseg=8'd109; 4'd6: disseg=8'd125; 4'd7: disseg=8'd7; 4'd8: disseg=8'd127; 4'd9: disseg=8'd111; 4'd10: disseg=8'd191; 4'd11: disseg=8'd124; 4'd12: disseg=8'd88; 4'd13: disseg=8'd94; 4'd14: disseg=8'd121; 4'd15: disseg=8'd113; default: disseg=8'd63; endcase end

endmodule

#### **User Constraint File:**

NET "clk" LOC = "P55";

NET "dismux<0>" LOC = "P14" ; NET "dismux<1>" LOC = "P15" ; NET "dismux<2>" LOC = "P17" ; NET "dismux<3>" LOC = "P18" ;

- NET "disseg<0>" LOC = "P31" ; NET "disseg<1>" LOC = "P32" ; NET "disseg<2>" LOC = "P33" ; NET "disseg<3>" LOC = "P34" ; NET "disseg<4>" LOC = "P35" ; NET "disseg<5>" LOC = "P36" ; NET "disseg<6>" LOC = "P37" ; NET "disseg<7>" LOC = "P39" ;
- NET "kycol<0>" LOC = "P1" ; NET "kycol<1>" LOC = "P2" ; NET "kycol<2>" LOC = "P3" ; NET "kycol<3>" LOC = "P4" ;

NET "kyrow<0>" LOC = "P5" ; NET "kyrow<1>" LOC = "P6" ; NET "kyrow<2>" LOC = "P7" ; NET "kyrow<3>" LOC = "P9" ;

#### **EXPECTED RESULT:**

- 1. Hexkey pad and 7 segment display are used to simulate the elevator experiment.
- **2.** To specify the floor number to which we would like to go, that number should be pressed in the hexkey pad.
- **3.** The transition from current floor to next floor (in steps) should be displayed in the 7 segment display.

HDL Lab Manual

Q4.

Aim: To write a Verilog code to realize a clock divider circuit that generates 1/2, 1/3rd and 1/4<sup>th</sup> clock from a given input clock using FPGA/CPLD and validate the functionality through oscilloscope.

# **BLOCK DIAGRAM**



## **WAVEFORM**



Program:

```
module CLK_DIVIDER(
  input CLK_IN,
  output reg CLK_OUT = 0,
  input [1:0] DIVISOR
  );
reg [1:0] counter=0;
always@(CLK_IN)
begin
      if(counter != DIVISOR)
             counter = counter + 1;
      else
             begin
             CLK_OUT = ~CLK_OUT;
        counter = 0;
             end
      end
endmodule
```

### **OUTPUT RESULTS:**



Output on Oscilloscope for DIVISOR=00, (input clock divide by 1)



Output on Oscilloscope for DIVISOR=01, (input clock divide by 2)



Output on Oscilloscope for DIVISOR=10, (input clock divide by 3)



Output on Oscilloscope for DIVISOR=11, (input clock divide by 4)



#### **EXAMINATION QUESTION COMBINATIONS**

Class: V-sem (EC-A,B,C) Sub Code : 18ECL58 Hrs/Week: 03

Exam Hours: 03 Hrs Exam Marks: 100 Marks

a).Write a gate level Verilog description to realize 2:4 decoder using NAND logic.
 b). Write Verilog code to simulate Elevator Operation.

2. a). Write a behavioral description to realize the 8:3 encoder without priority. b).Interface a DC motor to CPLD and write Verilog code to change its speed and direction.

3. a).Write a behavioral description to realize 8:3 encoder with priority.
b). Write a Verilog code to design a clock divider circuit that generates 1/2, 1/3<sup>rd</sup> and 1/4<sup>th</sup> clock from a given input clock. Port the design to CPLD and validate the functionality through oscilloscope.

4. a). Write a behavioral description to realize 8x1 MUX using IF statement. b). Interface a Stepper motor to CPLD and write Verilog code to control the Stepper motor rotation which in turn may control a Robotic Arm. External switches to be used for different controls like rotate the Stepper motor (i) +N steps if Switch no.1 of a Dip switch is closed (ii) +N/2 steps if Switch no. 2 of a Dip switch is closed (iii) –N steps if Switch no. 3 of a Dip switch is closed etc

5. a).Write a behavioral description to realize 8X1 MUX using CASE construct.
b). Write a Verilog code to design a clock divider circuit that generates 1/2, 1/3<sup>rd</sup> and 1/4<sup>th</sup> clock from a given input clock. Port the design to CPLD and validate the functionality through oscilloscope.

6. a).Write a behavioral description to model a 32 bit ALU .Simulate and verify the functionality.

b). Interface a DC motor to CPLD and write Verilog code to change its speed and direction.

- 7. a).Write a Verilog program to model FULL ADDER and AND,OR,XOR and XNOR gates and verify .
- b). Write Verilog code using FSM to simulate elevator operation.

8. a).Write a Verilog code to SR Flip flop and verify.

b). Interface a Stepper motor to CPLD and write Verilog code to control the Stepper motor rotation which in turn may control a Robotic Arm. External switches to be used for different controls like rotate the Stepper motor (i) +N steps if Switch no.1 of a Dip switch is closed (ii) +N/2 steps if Switch no. 2 of a Dip switch is closed (iii) –N steps if Switch no. 3 of a Dip switch is closed etc

9. a).Write a Verilog code to JK flip flop and verify.

b). Write a Verilog code to design a clock divider circuit that generates 1/2,  $1/3^{rd}$  and  $1/4^{th}$ clock from a given input clock. Port the design to CPLD and validate the functionality through oscilloscope.

10. a).Write a Verilog code to D flip flop and verify.

b). Write Verilog code using FSM to simulate elevator operation.

11. a).Write a verilog program to model 4 bit BCD synchronous counter and verify.b). Interface a DC motor to CPLD and write Verilog code to change its speed and direction.

12. a). Write Verilog code for counter with given input clock and check whether it works as clock divider performing division of clock by 2, 4, 8 and 16. Verify the functionality of the code.

b). Interface a Stepper motor to CPLD and write Verilog code to control the Stepper motor rotation which in turn may control a Robotic Arm. External switches to be used for different controls like rotate the Stepper motor (i) +N steps if Switch no.1 of a Dip switch is closed (ii) +N/2 steps if Switch no. 2 of a Dip switch is closed (iii) -N steps if Switch no. 3 of a Dip switch is closed etc

## **Department of Electronics and Communication Engineering**

#### <u>Vision</u>

"Imparting quality technical education through interdisciplinary research & innovation towards moulding the young talent with professional competence and ethical values for developing inclusive and sustainable technology in the area of Electronics and Communication Engineering".

### <u>Mission</u>

Create conducive environment for the holistic development of students and staff members.

- Provide quality technical education to produce industry ready engineers with an entrepreneurial and research outlook.
- Establish centers of excellence in collaboration with industries/universities for exposing the students to latest technologies.
- Nurture the students to actively participate in solving the societal problems and uphold ethics and morality.
- To train the students to meet global challenges in interdisciplinary fields by inculcating a quest for modern technologies in the emerging areas.