

Kandula Srinivasa Reddy Memorial College of Engineering (Autonomous)

Kadapa-516003. AP

(Approved by AICTE, Affiliated to JNTUA, Ananthapuramu, Accredited by NAAC)

(An ISO 9001-2008 Certified Institution)

Department of Electronics and Communication Engineering



Certification Course

On

“Digital System Design using Verilog HDL”

Resource Person : Sri. R V Sreehari

Course Coordinators: Sri. Y.V. Raju

Duration : 23-03-2022 to 08-04-2022



K.S.R.M. COLLEGE OF ENGINEERING

(UGC - AUTONOMOUS)

Kadapa, Andhra Pradesh, India - 516003

Approved by AICTE, New Delhi & Affiliated to JNTUA, Ananthapuramu.

An ISO 14001:2004 & 9001: 2015 Certified Institution



Lr./KSRMCE/ (Department of ECE)/2021-22

Date: 14-03-2022

To
The Principal,
KSRM College of Engineering (A),
Kadapa, AP.

Sub: KSRMCE - (Department of ECE) – Permission to conduct certification course on DSD using Verilog HDL–Request– reg.

---***---

Respected Sir,

With reference to the cited, the Department of ECE is planning to conduct a certification course on “DSD using Verilog HDL” for B.Tech VI SEM students from 23-03-2022 to 08-04-2022 in online/offline mode. In this regard, we kindly request you to grant permission to conduct a certification course. This is submitted for your kind perusal.

Thanking you sir,

Yours Faithfully,

Coordinators

Sri R.V. Sreehari

Sri Y.V. Raju

Cc:

To All Deans/HODs

Permitted

V. S. S. Mulu
PRINCIPAL
K.S.R.M. COLLEGE OF ENGINEERING
KADAPA-516005, (A.P.)



/ksrmce.ac.in

Follow Us:



/ksrmceofficial



K.S.R.M. COLLEGE OF ENGINEERING

(UGC - AUTONOMOUS)

Kadapa, Andhra Pradesh, India - 516003

Approved by AICTE, New Delhi & Affiliated to JNTUA, Ananthapuramu.

An ISO 14001:2004 & 9001: 2015 Certified Institution



Date: 16-03-2022

Circular

All the B.Tech VI SEM students are hereby informed that the department of ECE is going to conduct a 30 hours a certification course on “DSD using Verilog HDL” from 23-03-2022 to 08-04-2022. Interested students may register their names with the following link on or before 23-03-2022.

Registration Link:

https://docs.google.com/forms/d/e/1FAIpQLSfcPN1i6URP8mICiRV8QvS75dbJ_Eq96hGUXGrfKf3RHw5CMA/viewform?usp=sf_link

For any queries contact,

Coordinator

Sri R.V. Sreehari, Associate Professor, ECE Dept.,

Sri Y.V. Raju , Assistant Professor, ECE Dept.,

Cc to:

The Management /Director / All Deans / All HODS/Staff / Students for information

The IQAC Cell for Documentation

HOD

Professor & H.O.D.
Department of E.C.E.
K.S.R.M. College of Engineering
KADAPA - 516 003



/ksrmce.ac.in

Follow Us:



/ksrmceofficial

Registration For Certificate Course on "Digital System Design using Verilog HDL" from 23/03/2022 to 08/04/2022

Registration For Certificate Course on "Digital System Design using Verilog HDL" from
23/03/2022 to 08/04/2022

* Required

1. Name of the Student(As per the SSC) *

2. Roll number *

3. Email ID *

4. Phone number(Whats App) *

5. Branch & Sem

This content is neither created nor endorsed by Google.

Google Forms



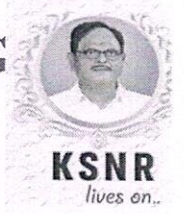
K.S.R.M. COLLEGE OF ENGINEERING

(UGC - AUTONOMOUS)

Kadapa, Andhra Pradesh, India - 516003

Approved by AICTE, New Delhi & Affiliated to JNTUA,
Ananthapuramu.

An ISO 14001:2004 & 9001: 2015 Certified Institution



Department of Electronics & Communication Engineering

Certificate Course on DSD using Verilog HDL

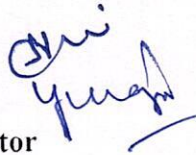
Registered Students List

S.No.	Roll Number	Name of the student	Email Address
1	199Y1A0403	ANGAJALA KAVYA SREE	199y1a0403@ksrmce.ac.in
2	199Y1A0407	AYYALURI VENKATA PAVAN KUMAR REDDY	199y1a0407@ksrmce.ac.in
3	199Y1A0412	BARIVENKULA SREENATH	199y1a0412@ksrmce.ac.in
4	199Y1A0413	BINGIMALLA VENKATA THARUN KUMAR	199y1a0413@ksrmce.ac.in
5	199y1a0415	BOOSI VENKATA SAINATH REDDY	199y1a0415@ksrmce.ac.in
6	199Y1A0418	C.TEJESH KUMAR REDDY	199y1a0418@ksrmce.ac.in
7	199Y1A0422	CHEEPATI VARAPRASAD REDDY	199y1a0422@ksrmce.ac.in
8	199y1a0425	CHERUVU SAI PRAKASH REDDY	199y1a0425@ksrmce.ac.in
9	199y1a0426	CHILUMURU GEETHA PRIYA	199y1a0426@ksrmce.ac.in
10	199y1a0429	Chinnireddy Neetha	199y1a0429@ksrmce.ac.in
11	199Y1A0432	DANDE MOUNIKA	199y1a0432@ksrmce.ac.in

12	199Y1A0434	DESURI VARSHINI	199y1a0434@ksrmce.ac.in
13	199Y1A0441	Anushka Gadde	199y1a0441@ksrmce.ac.in
14	199Y1A0442	Gajjala Navya Tejasree	199y1a0442@ksrmce.ac.in
15	199Y1A0443	GAMPA SIVA	199y1a0443@ksrmce.ac.in
16	199Y1A0447	GODI KAVEETHA	199y1a0447@ksrmce.ac.in
17	199Y1A0451	GOURIPEDDI K S SREEDHANYA	199y1a0451@ksrmce.ac.in
18	199Y1A0452	GULYAM SHARATH	199y1a0452@ksrmce.ac.in
19	199y1a0455	Jampala Anjali	199y1a0455@ksrmce.ac.in
20	199y1a0457	K. ANUSHA	199y1a0457@ksrmce.ac.in
21	199Y1A0458	K S Gowthami	199y1a0458@ksrmce.ac.in
22	199Y1A0459	SOWMYA PRIYA. K	199y1a0459@ksrmce.ac.in
23	199y1a0472	K.prathyusha	199y1a0472@ksrmce.ac.in
24	199Y1A0474	KONDURU SUMITHRA	199y1a0474@ksrmce.ac.in
25	199y1a04a5	M. Pranitha	199y1a04a5@ksrmce.ac.in
26	199Y1A04A6	MURARI SRAVANTHI	199y1a04a6@ksrmce.ac.in
27	199Y1A04A7	M.V.SUBRAMANYAM	199y1a04a7@ksrmce.ac.in
28	199Y1A04A9	NALLABOTHULA POORNIMA	199y1a04a9@ksrmce.ac.in
29	199Y1A04B0	NAMALA JYOTHIKA GNANA CHANDRIKA	199y1a04b0@ksrmce.ac.in
30	199Y1A04B1	NANDIPATI MANEESHA	199y1a04b1@ksrmce.ac.in
31	199Y1A04C0	P. LINGESWARI	199y1a04c0@ksrmce.ac.in

32	199Y104c4	P.S.KOUSHIK	199y1a04c4@ksrmce.ac.in
33	199Y1A04D1	SADIVE LAKSHMI PRASANNA	199y1a04d1@ksrmce.ac.in
34	199Y1A04F9	TADIMARRI MOUNIKA	199y1a04f9@ksrmce.ac.in
35	199y1a04h2	Vanam vishnu naga vardhan reddy	199y1a04h2@ksrmce.ac.in

Coordinator



HOD



Professor & H.O.D.
Department of E.C.E.
K.S.R.M. College of Engineering
KADAPA - 516 003

Certification Course on “DSD using Verilog HDL” Syllabus

30 Hrs

Course Objectives:

- Learn the basic Verilog programming.
- Understand Design Methodologies of Combinational circuits.
- Gain knowledge to design Verilog model for sequential circuits and test pattern generation.
- Understand the application development using verilog HDL.

Course Outcomes:

The students will be able to:

- Learn the Verilog programming language and HDL tools.
- Construct the combinational circuits, using discrete gates and verilog its modules.
- Describe Verilog model for sequential circuits and test pattern generation.
- Develop applications like traffic light controller, ROM, FIFO etc., using Verilog HDL.

Module 1: Overview of Digital Design With Verilog HDL (8 Hours)

Typical Design Flow, Importance of HDLs, Popularity of Verilog HDL, Trends In HDLs, Hierarchical Modeling Concepts : Design Methodologies, 4-Bit Ripple Carry Counter, Modules Instances, Components of A Simulation, Example - Design Block - Stimulus Block.

Module 2: Basic Concepts, Modules And Ports (4 Hours)

Lexical Conventions, Data Types, System Tasks And Compiler Directives, Ports, Hierarchical Names.

Module 3: Gate-Level Modeling and Dataflow Modeling (6 Hours)

Gate Types, Gate Delays, Continuous Assignments, Delays, Expressions, Operators, and Operands, Operator Types, Examples.

Module 4: Behavioral Modeling , Tasks & Functions (6 Hours)

Structured Procedures, Procedural Assignments, Timing Controls, Conditional Statements, Multi Way Branching, Loops, Sequential and Parallel Blocks, Generate Blocks, Differences Between Tasks And Functions, Tasks, Functions.

Module 5: Case Studies (6 Hours)

Traffic Signal Controller, Paper Vending Machine, Dual Port RAM, FIFO.

Text book:

1. Verilog HDL: A Guide to Digital Design and Synthesis ,Second Edition By Samir Palnitkar.

Reference:

1. Charles Roth, Lizy K. John, Byeong Kil Lee, "Digital Systems Design Using Verilog", Cengage, 1st Edition.
2. Michael D. Ciletti, "Advanced Digital Design with the Verilog HDL" Pearson (Prentice Hall), Second edition.
3. A Verilog HDL Primer by J Bhaskar, Star Galaxy Press (1 March 1997).



K.S.R.M. COLLEGE OF ENGINEERING

(UGC - AUTONOMOUS)

Kadapa, Andhra Pradesh, India - 516003

Approved by AICTE, New Delhi & Affiliated to JNTUA,

Ananthapuramu.

An ISO 14001:2004 & 9001: 2015 Certified Institution



Department of Electronics & Communication Engineering

Certificate Course on DSD using Verilog HDL

Schedule

S.No	Date	Time	Faculty	Topic
1	23/03/2022	4 PM to 5PM	Dr. G.Hemalatha Sri M. Madan Gopal, Sri R.V. Sreehari Sri Y.V. Raju	Inauguration
2	24/03/2022	3 PM to 5PM	Sri M. Madan Gopal, Sri R.V. Sreehari Sri Y.V. Raju	Typical Design Flow, Importance of HDLs, Popularity of Verilog HDL, Trends In HDLs
3	25/03/2022	3 PM to 5PM	Sri M. Madan Gopal, Sri R.V. Sreehari Sri Y.V. Raju	Hierarchical Modeling Concepts: Design Methodologies, 4-Bit Ripple Carry Counter,
4	26/03/2022	3 PM to 5PM	Sri M. Madan Gopal, Sri R.V. Sreehari Sri Y.V. Raju	Modules Instances, Components of a Simulation, Example - Design Block - Stimulus Block.
5	28/03/2022	3 PM to 5PM	Sri M. Madan Gopal, Sri R.V. Sreehari Sri Y.V. Raju	Lexical Conventions, Data Types, System Tasks
6	29/03/2022	3 PM to 5PM	Sri M. Madan Gopal, Sri R.V. Sreehari Sri Y.V. Raju	Compiler Directives, Ports, Hierarchical Names.
7	30/03/2022	3PM to 5PM	Sri M. Madan Gopal, Sri R.V. Sreehari Sri Y.V. Raju	Gate Types, Gate Delays, Continuous Assignments
8	31/03/2022	3 PM to 5PM	Sri M. Madan Gopal, Sri R.V. Sreehari Sri Y.V. Raju	Delays, Expressions, Operators, and Operands, Operator Types, Examples.

9	01/04/202022	4 PM to 6PM	Sri M. Madan Gopal, Sri R.V. Sreehari Sri Y.V. Raju	Structured Procedures, Procedural Assignments,
10	02/04/202022	4 PM to 6PM	Sri M. Madan Gopal, Sri R.V. Sreehari Sri Y.V. Raju	Timing Controls, Conditional Statements,
11	04/04/202022	4 PM to 6PM	Sri M. Madan Gopal, Sri R.V. Sreehari Sri Y.V. Raju	Multi Way Branching, Loops, Sequential and Parallel Blocks.
12	05/04/202022	4 PM to 6PM	Sri M. Madan Gopal, Sri R.V. Sreehari Sri Y.V. Raju	Generate Blocks, Differences Between Tasks And Functions, Tasks, Functions.
13	06/04/202022	4PM to 6PM	Sri M. Madan Gopal, Sri R.V. Sreehari Sri Y.V. Raju	Traffic Signal Controller
14	07/04/202022	4 PM to 6PM	Sri M. Madan Gopal, Sri R.V. Sreehari Sri Y.V. Raju	Traffic Signal Controller
15	08/04/202022	4PM to 6PM	Sri M. Madan Gopal, Sri R.V. Sreehari Sri Y.V. Raju	Paper Vending Machine

Coordinator

*Sri
Y.V. Raju*

HOD

G. H. H.
Professor & H.O.D.
Department of E.C.E.
K.S.R.M. College of Engineering,
KADAPA - 516 003.



K.S.R.M. COLLEGE OF ENGINEERING

(UGC - AUTONOMOUS)

Kadapa, Andhra Pradesh, India - 516003

Approved by AICTE, New Delhi & Affiliated to JNTUA, Ananthapuramu.

An ISO 14001:2004 & 9001: 2015 Certified Institution



Department of Electronics & Communication Engineering

Certificate Course on *Digital System Design using Verilog HDL*

Attendance Sheet

S.No	Roll No.	Name of the Student	2	2	2	2	2	2	3	3	0	0	0	0	0	0	0
			3	4	5	6	8	9	0	1	1	2	4	5	6	7	8
			/	/	/	/	/	/	/	/	/	/	/	/	/	/	
			0	0	0	0	0	0	0	0	0	0	0	0	0	0	
			3	3	3	3	3	3	3	4	4	4	4	4	4	4	
			/	/	/	/	/	/	/	/	/	/	/	/	/	/	
			2	2	2	2	2	2	2	2	2	2	2	2	2	2	
			0	0	0	0	0	0	0	0	0	0	0	0	0	0	
			2	2	2	2	2	2	2	2	2	2	2	2	2	2	
			2	2	2	2	2	2	2	2	2	2	2	2	2	2	
1	199Y1A0403	ANGAJALA KAVYA SREE	p	p	p	p	p	p	p	p	p	p	p	p	p	p	
2	199Y1A0407	AYYALURI VENKATA PAVAN KUMAR REDDY	p	p	p	p	p	p	p	p	p	p	p	p	p	p	
3	199Y1A0412	BARIVENKULA SREENATH	p	p	p	p	p	p	a	p	p	p	p	p	p	a	
4	199Y1A0413	BINGIMALLA VENKATA THARUN KUMAR	p	p	p	a	a	p	p	p	p	p	p	p	p	p	
5	199y1a0415	BOOSI VENKATA SAINATH REDDY	p	p	a	p	p	p	p	p	p	p	p	p	p	p	
6	199Y1A0418	C.TEJESH KUMAR REDDY	p	p	p	p	p	p	p	p	p	p	p	p	p	p	
7	199Y1A0422	CHEEPATI VARAPRASAD REDDY	p	a	p	p	p	p	p	a	p	a	p	p	a	p	

8	199y1a0425	CHERUVU SAI PRAKASH REDDY	p	p	p	p	a	a	p	p	a	p	p	p	p	p	p
9	199y1a0426	CHILUMURU GEETHA PRIYA	p	p	p	p	p	p	p	p	p	p	a	p	p	p	p
10	199y1a0429	Chinnireddy Neetha	p	p	p	a	p	p	a	p	p	p	p	p	p	a	p
11	199Y1A0432	DANDE MOUNIKA	p	p	p	p	p	p	p	p	p	p	p	p	p	p	p
12	199Y1A0434	DESURI VARSHINI	p	p	p	p	p	a	p	p	p	p	a	p	p	p	p
13	199Y1A0441	Anushka Gadde	p	p	p	p	p	a	p	p	p	p	p	p	p	p	p
14	199Y1A0442	Gajjala Navya Tejasree	p	p	p	p	p	p	p	p	p	p	p	p	p	p	p
15	199Y1A0443	GAMPA SIVA	p	p	p	p	p	p	p	p	p	p	p	p	p	p	p
16	199Y1A0447	GODI KAVEETHA	a	p	p	p	p	p	p	p	p	p	p	p	p	p	p
17	199Y1A0451	GOURIPEDDI K S SREEDHANYA	p	p	p	p	p	p	a	p	p	p	p	a	p	p	p
18	199Y1A0452	GULYAM SHARATH	p	p	a	p	p	p	p	p	p	p	p	p	p	p	a
19	199y1a0455	Jampala Anjali	p	p	p	p	p	p	p	p	p	a	p	p	p	a	p
20	199y1a0457	K. ANUSHA	p	p	p	p	p	p	p	a	p	p	p	p	p	a	p
21	199Y1A0458	K S Gowthami	p	a	p	a	p	p	p	p	p	p	p	p	p	p	p
22	199Y1A0459	SOWMYA PRIYA. K	p	p	p	p	p	p	p	p	p	p	p	p	a	p	p
23	199y1a0472	K.prathyusha	p	p	p	p	p	a	p	p	p	p	p	p	p	p	p
24	199Y1A0474	KONDURU SUMITHRA	p	p	p	p	p	a	p	p	a	p	p	p	p	a	p
25	199y1a04a5	M. Pranitha	p	p	a	p	a	p	p	p	p	p	p	p	p	p	p
26	199Y1A04A6	MURARI SRAVANTHI	a	p	p	p	p	p	p	p	p	a	p	p	p	p	p
27	199Y1A04A7	M.V.SUBRAMANYAM	p	a	p	a	p	p	p	p	p	p	p	a	p	p	p
28	199Y1A04A9	NALLABOTHULA POORNIMA	p	p	p	p	p	p	p	p	p	p	p	p	p	a	p
29	199Y1A04B0	NAMALA JYOTHIKA GNANA CHANDRIKA	p	p	p	p	p	a	p	p	p	p	a	p	p	p	p
30	199Y1A04B1	NANDIPATI MANEESHA	p	a	p	p	p	p	p	p	a	p	p	p	p	p	p
31	199Y1A04C0	P. LINGESWARI	p	p	a	p	p	p	p	p	p	p	a	p	p	p	p
32	199Y104c4	P.S.KOUSHIK	p	p	p	a	a	p	p	p	p	p	p	p	p	p	p
33	199Y1A04D1	SADIVE LAKSHMI PRASANNA	a	p	p	p	p	p	p	a	p	p	p	p	a	p	p
34	199Y1A04F9	TADIMARRI MOUNIKA	p	p	p	p	p	a	p	p	a	a	p	p	p	p	p
35	199y1a04h2	Vanam vishnu naga vardhan reddy	p	p	p	p	p	a	p	p	p	p	p	p	p	p	p

An yuvam
Coordinator

G. Y. H.
HOD
Professor & H.O.D.
Department of E.C.E.
K.S.R.M. College of Engineering
KADAPA - 516 003



K.S.R.M. COLLEGE OF ENGINEERING

(UGC - Autonomous)

Kadapa, Andhra Pradesh, India- 516 003

Approved by AICTE, New Delhi & Affiliated to JNTUA, Ananthapuramu.



KSNR
lives on.

DEPARTMENT OF ECE

A Certification Course on

"Digital System Design using Verilog HDL"



23-03-2022 to
08-04-2022



Offline/Online
(both)

Coordinators

Sri R.V. Sreehari,
Associate Professor,

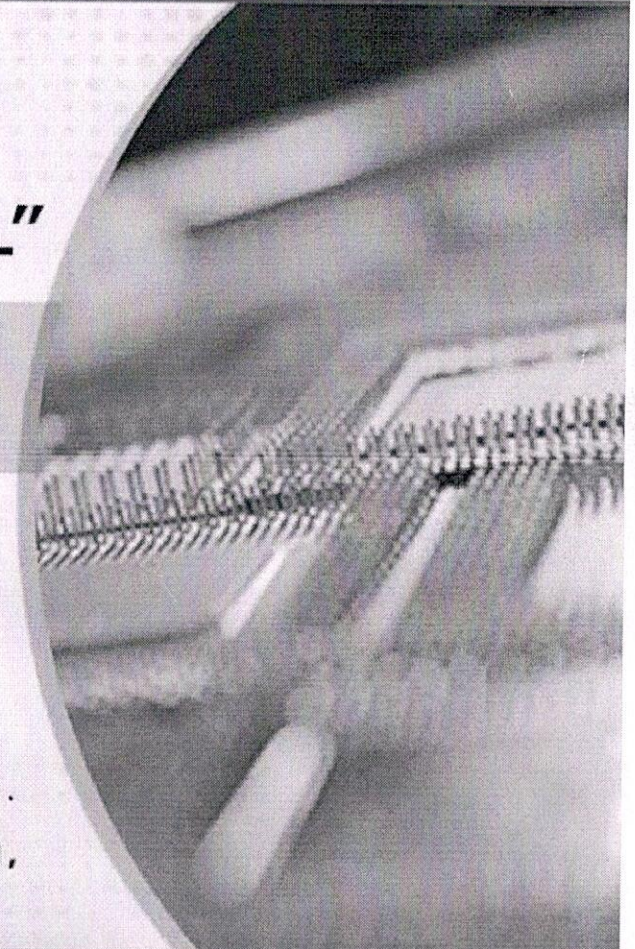
Sri Y. Venkateswara Raju,
Assistant Professor,

Resource Persons

Sri Madan Gopal. Mekala,
Corporate Trainer, Aark IC Technologies Pvt. Ltd., Bangalore.

Sri R.V. Sreehari,
Associate Professor, ECE

Sri Y. Venkateswara Raju,
Assistant Professor, ECE



Dr. G. Hemalatha
(HOD)

Dr. V.S.S. Murthy
(Principal)

Dr. Kandula Chandra Obul Reddy
(Managing Director)

Smt. K.Rajeswari
(Correspondent Secretary, Treasurer)

Sri K. Madan Mohan Reddy
(Vice - Chairman)

Sri K. Raja Mohan Reddy
(Chairman)

f @ t v ▶ **ksrmceofficial**



www.ksrmce.ac.in



8143731980, 8575697569



K.S.R.M. COLLEGE OF ENGINEERING

(UGC-AUTONOMOUS)

Kadapa, Andhra Pradesh, India- 516 003

Approved by AICTE, New Delhi & Affiliated to JNTUA, Ananthapuramu.

An ISO 14001:2004 & 9001: 2015 Certified Institution



ACTIVITY REPORT

Certification Course

On

“Digital System Design using Verilog HDL”

23rd March, 2022 to 08th April, 2022

Target Group	:	VI Sem ECE Students
Details of Participants	:	35 Students
Co-ordinator	:	Sri R.V. Sreehari, Associate Professor, Dept. of ECE, Sri Y.V. Raju, Assistant Professor, Dept. of ECE
Organizing Department	:	Department of Electronics & Communication Engineering
Venue	:	Online mode (WebEx), Offline - DSP Lab

<https://meet195.webex.com/wbxmjs/joinservice/sites/meet195/meeting/download/925ec9af13a64ab9a36d1ff07146d30?protocolUID=401037907c7d51e775f94b9ae068f74f&MTID=ma2251f3c3ab875e3a48d5a6db52c1c2a#noRefreshs>

Description: The certification course on "Digital System Design using Verilog HDL" was organized by the Dept. of ECE from March 23rd, 2022 to April 8th, 2022 in an online/offline mode. Mr. M. Madan Gopal, Corporate Trainer, AarkIC Technologies Pvt. Ltd., Bangalore, acted as Resource Person. Mr. R.V. Sreehari, Associate Professor, and Mr. Y.V. Raju, Assistant Professor, Department of ECE, acted as course instructors and coordinators. The main aim of the course is to design a digital system using Verilog HDL. This Thirty-Hour course was completed and participation certificates were provided to the participants.



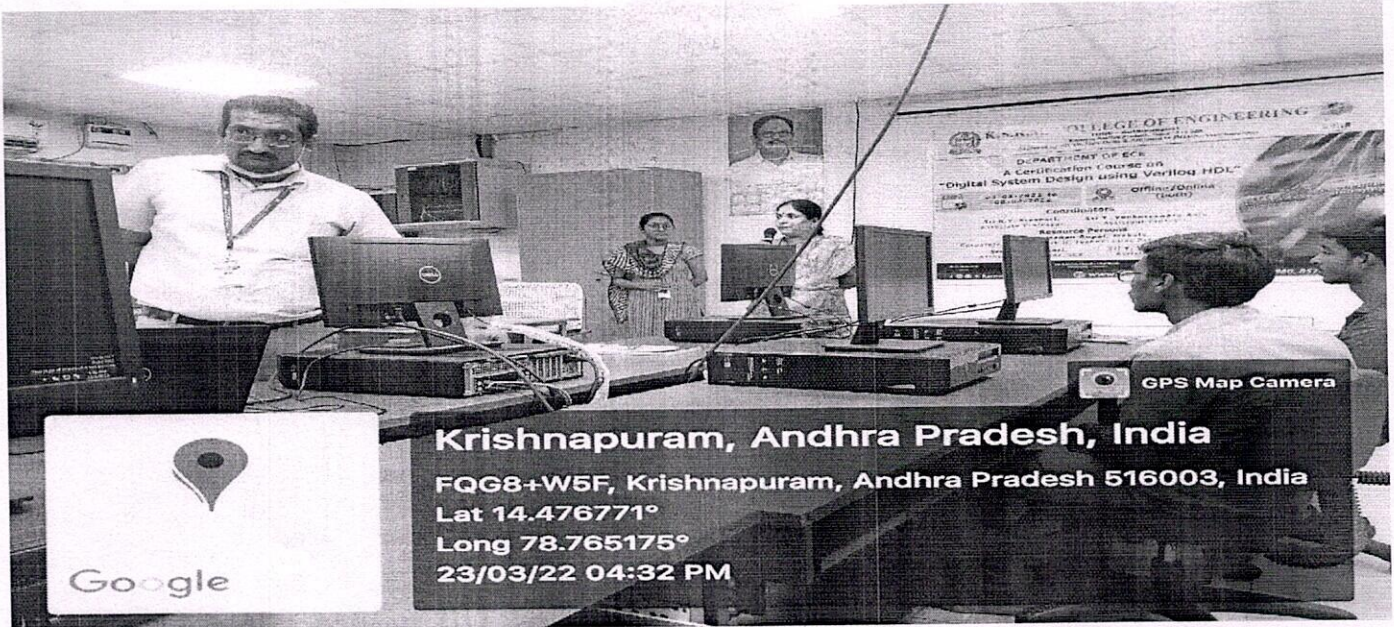
/ksrmce.ac.in

Follow Us:

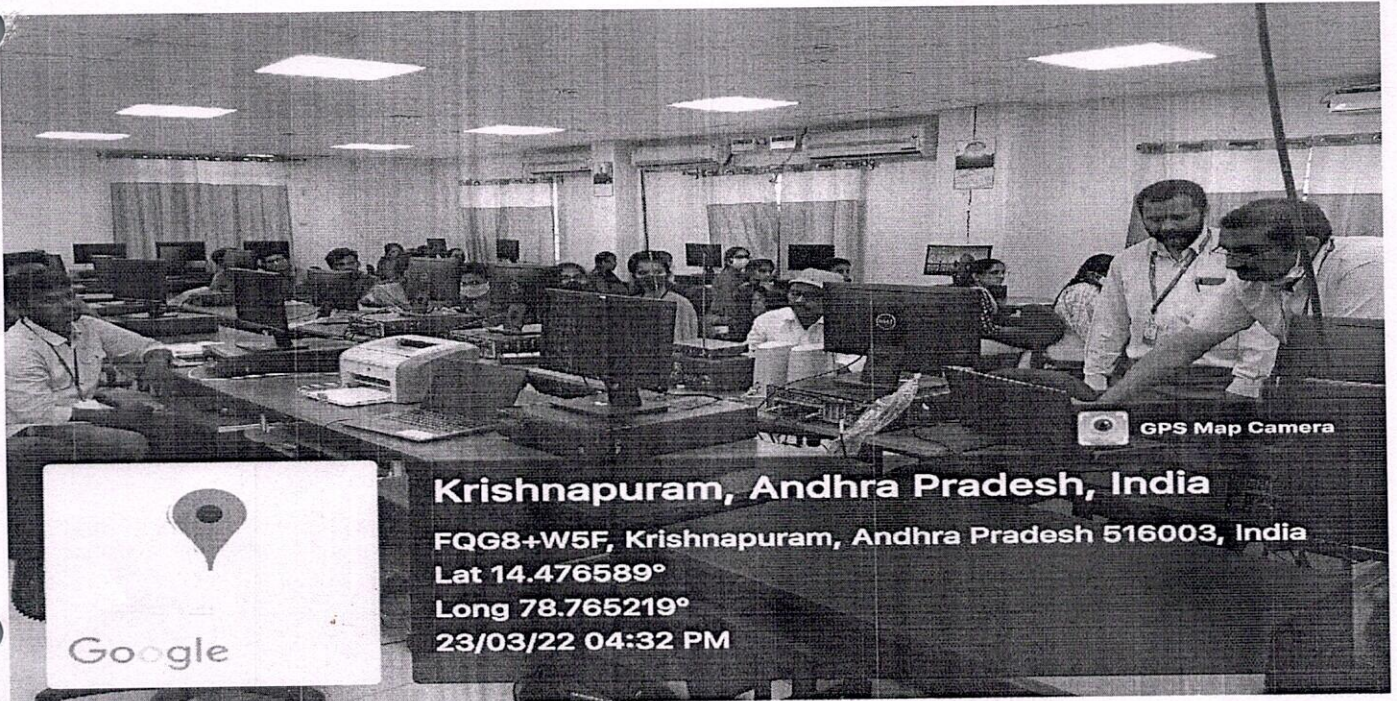


/ksrmceofficial

Photo



Krishnapuram, Andhra Pradesh, India
 FQG8+W5F, Krishnapuram, Andhra Pradesh 516003, India
 Lat 14.476771°
 Long 78.765175°
 23/03/22 04:32 PM



Krishnapuram, Andhra Pradesh, India
 FQG8+W5F, Krishnapuram, Andhra Pradesh 516003, India
 Lat 14.476589°
 Long 78.765219°
 23/03/22 04:32 PM

Coordinators

Sri R.V. Sreehari

Sri Sreehari
Y.V. Raju

Sri Y.V. Raju

G. H. H.
 HoD

Professor & H.O.D.
Department of E.C.E.
K.S.R.M. College of Engineering
KADAPA - 516 003



K.S.R.M. COLLEGE OF ENGINEERING

(UGC - Autonomous)

Kadapa, Andhra Pradesh, India- 516 003

Approved by AICTE, New Delhi & Affiliated to JNTUA, Ananthapuramu.



KSNR

lives on..

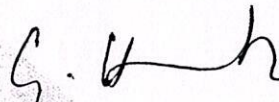
A CERTIFICATION COURSE ON "DIGITAL SYSTEM DESIGN USING VERILOG HDL"

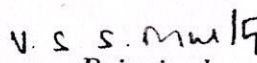
CERTIFICATE OF PARTICIPATION

This is to certify that Mr./Miss V. Vishnu Naga Vardhan Reddy (19941A04H2)
Department of ECE, K.S.R.M. College of Engineering, Kadapa has attended
A Certification Course on "Digital System Design using Verilog HDL"
during 23-03-2022 to 08-04-2022, Organized by the Department of ECE,
K.S.R.M. College of Engineering, Kadapa, in association with Aark IC Technol-
ogies Pvt. Ltd., Bangalore.


Co-Coordinator


Coordinator


HOD ECE


Principal



K.S.R.M. COLLEGE OF ENGINEERING

(UGC - Autonomous)

Kadapa, Andhra Pradesh, India- 516 003

Approved by AICTE, New Delhi & Affiliated to JNTUA, Ananthapuramu.



KSNR
lives on..

A CERTIFICATION COURSE ON "DIGITAL SYSTEM DESIGN USING VERILOG HDL"

CERTIFICATE OF PARTICIPATION

This is to certify that *Mr./Miss* *K. Prathyusha (19941A0472)*
Department of ECE, K.S.R.M.College of Engineering, Kadapa has attended
A Certification Course on "Digital System Design using Verilog HDL"
during 23-03-2022 to 08-04-2022, Organized by the Department of ECE,
K.S.R.M. College of Engineering, Kadapa, in association with Aark IC Technol-
ogies Pvt. Ltd., Bangalore.

[Signature]
Co-Coordinator

[Signature]
Coordinator

[Signature]
HOD ECE

[Signature]
Principal



K.S.R.M. COLLEGE OF ENGINEERING

(UGC - Autonomous)

Kadapa, Andhra Pradesh, India- 516 003

Approved by AICTE, New Delhi & Affiliated to JNTUA, Ananthapuramu.

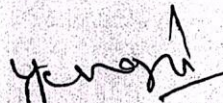


KSNR
lives on..

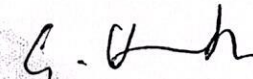
A CERTIFICATION COURSE ON DIGITAL SYSTEM DESIGN USING VERILOG HDL

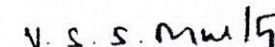
CERTIFICATE OF PARTICIPATION

This is to certify that *Mr./Miss B. Venkata Tharun Kumar (19941A0413)*
Department of ECE, K.S.R.M. College of Engineering, Kadapa has attended
A Certification Course on "Digital System Design using Verilog HDL"
during 23-03-2022 to 08-04-2022, Organized by the Department of ECE,
K.S.R.M. College of Engineering, Kadapa, in association with Aark IC Technol-
ogies Pvt. Ltd., Bangalore.


Co-Coordinator


Coordinator


HOD ECE


Principal

Feedback form on Certificate Course

DSD Using VerilogHDL(23/03/2022 to 08/04/2022)

* Required

1. Roll Number *

2. Name of the Student *

3. B.Tech Semester *

Mark only one oval.

I SEM

II SEM

III SEM

IV SEM

V SEM

VI SEM

VII SEM

VIII SEM

4. Branch *

Mark only one oval.

Civil Engineering

EEE

ME

ECE

CSE

AI&ML

5. Email ID *

6. Is the course content meet your expectation. *

Mark only one oval.

Yes

No

7. Is the lecture sequence well planned. *

Mark only one oval.

Strongly disagree

Disagree

Neutral

Agree

Strongly agree

8. The contents of the course is explained with examples. *

Mark only one oval.

Strongly disagree

Disagree

Neutral

Agree

Strongly Agree

9. Is the level of course high. *

Mark only one oval.

Strongly disagree

Disagree

Neutral

Agree

Strongly Agree

10. Is the course exposed you to the new knowledge and practice. *

Mark only one oval.

Strongly disagree

Disagree

Neutral

Agree

Strongly Agree

11. Is the lecture clear and easy to understand. *

Mark only one oval.

- Strongly disagree
- Disagree
- Neutral
- Agree
- Strongly agree

12. Rate the value of the course increasing your skills. *

Mark only one oval.

- Strongly disagree
- Disagree
- Neutral
- Agree
- Strongly Agree

13. Any suggestions

This content is neither created nor endorsed by Google.

Google Forms



(UGC - AUTONOMOUS)

K.S.R.M. COLLEGE OF ENGINEERING

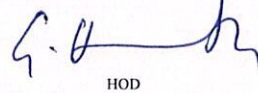
Kadapa, Andhra Pradesh, India - 516003
Approved by AICTE, New Delhi & Affiliated to JNTUA, Ananthapuramu.
Department of Electronics and Communication Engineering
Feedback Form



S.No.	Email address	Name of the student	Year & Semester	Branch	Roll No.	Is the course content met your expectation	Is the lecture sequence well planned	The contents of the course is explained with examples	Is the level of course high	Is the course exposed you to the new knowledge and practices	Is the lecturer clear and easy to understand	Rate the value of course in increasing your skills 1 to 5	Any issues
1	199y1a0403@ksrmce.ac.in	ANGAJALA KAVYA SREE AYYALURI VENKATA PAVAN KUMAR REDDY	B.Tech VI Sem	ECE	199Y1A0403	Yes	Yes	Agree	Agree	Strongly agree	Yes	5	Nothing
2	199y1a0407@ksrmce.ac.in	ANGAJALA KAVYA SREE AYYALURI VENKATA PAVAN KUMAR REDDY	B.Tech VI Sem	ECE	199Y1A0407	Yes	Yes	Agree	Agree	Strongly agree	Yes	5	Nothing
3	199y1a0412@ksrmce.ac.in	BARIVENKULA SREENATH BINGIMALLA VENKATA THARUN KUMAR	B.Tech VI Sem	ECE	199Y1A0412	Yes	Yes	Agree	Agree	Strongly agree	Yes	5	Good
4	199y1a0413@ksrmce.ac.in	BARIVENKULA SREENATH BINGIMALLA VENKATA THARUN KUMAR	B.Tech VI Sem	ECE	199Y1A0413	Yes	Yes	Agree	Agree	Strongly agree	Yes	5	nothing
5	199y1a0415@ksrmce.ac.in	BOOSI VENKATA SAINATH REDDY	B.Tech VI Sem	ECE	199Y1A0415	Yes	Yes	Agree	Agree	Strongly agree	Yes	5	Good
6	199y1a0418@ksrmce.ac.in	C.TEJESH KUMAR REDDY	B.Tech VI Sem	ECE	199Y1A0418	Yes	Yes	Agree	Agree	Strongly agree	Yes	5	very good
7	199y1a0422@ksrmce.ac.in	CHEEPATI VARAPRASAD REDDY	B.Tech VI Sem	ECE	199Y1A0422	Yes	Yes	Strongly agree	Agree	Strongly agree	Yes	3	Nothing
8	199y1a0425@ksrmce.ac.in	CHERUVU SAI PRAKASH REDDY	B.Tech VI Sem	ECE	199Y1A0425	Yes	Yes	agree	Agree	Strongly agree	Yes	4	no
9	199y1a0426@ksrmce.ac.in	CHILUMURU GEETHA PRIYA	B.Tech VI Sem	ECE	199Y1A0426	Yes	Yes	Strongly agree	Agree	Strongly agree	Yes	5	Nothing
10	199y1a0429@ksrmce.ac.in	Chinnireddy Neetha	B.Tech VI Sem	ECE	199Y1A0429	Yes	Yes	Strongly agree	Agree	Strongly agree	Yes	5	Good
11	199y1a0432@ksrmce.ac.in	DANDE MOUNIKA	B.Tech VI Sem	ECE	199Y1A0432	Yes	Yes	Agree	Agree	Strongly agree	Yes	4	Good
12	199y1a0434@ksrmce.ac.in	DESURI VARSHINI	B.Tech VI Sem	ECE	199Y1A0434	Yes	Yes	agree	Agree	Strongly agree	Yes	5	Good
13	199y1a0441@ksrmce.ac.in	Anushka Gadde	B.Tech VI Sem	ECE	199Y1A0441	Yes	Yes	agree	Agree	Strongly agree	Yes	5	Good
14	199y1a0442@ksrmce.ac.in	Gajjala Navya Tejasree	B.Tech VI Sem	ECE	199Y1A0442	Yes	Yes	agree	Agree	Strongly agree	Yes	4	very good
15	199y1a0443@ksrmce.ac.in	GAMPA SIVA	B.Tech VI Sem	ECE	199Y1A0443	Yes	Yes	agree	Agree	Strongly agree	Yes	4	very good
16	199y1a0447@ksrmce.ac.in	GODI KAVEETHA	B.Tech VI Sem	ECE	199Y1A0447	Yes	Yes	agree	Agree	Strongly agree	Yes	4	very good
17	199y1a0451@ksrmce.ac.in	GOURIPEDDI K S SREEDHANYA	B.Tech VI Sem	ECE	199Y1A0451	Yes	Yes	agree	Agree	Strongly agree	Yes	5	no
18	199y1a0452@ksrmce.ac.in	GULYAM SHARATH	B.Tech VI Sem	ECE	199Y1A0452	Yes	Yes	agree	Agree	Strongly agree	Yes	5	nothing
19	199y1a0455@ksrmce.ac.in	Jampala Anjali	B.Tech VI Sem	ECE	199Y1A0455	Yes	Yes	Strongly agree	Agree	Strongly agree	Yes	5	Good
20	199y1a0457@ksrmce.ac.in	K. ANUSHA	B.Tech VI Sem	ECE	199Y1A0457	Yes	Yes	Strongly agree	Agree	Strongly agree	Yes	4	Good

21	199y1a0458@ksrmce.ac.in	K S Gowthami	B.Tech VI Sem	ECE	199Y1A0458	Yes	Yes	Strongly agree	Agree	Strongly agree	Yes	3	Good
22	199y1a0459@ksrmce.ac.in	SOWMYA PRIYA. K	B.Tech VI Sem	ECE	199Y1A0459	Yes	Yes	agree	Agree	Strongly agree	Yes	4	Good
23	199y1a0472@ksrmce.ac.in	K.prathyusha	B.Tech VI Sem	ECE	199Y1A0472	Yes	Yes	agree	Agree	Strongly agree	Yes	4	Good
24	199y1a0474@ksrmce.ac.in	KONDURU SUMITHRA	B.Tech VI Sem	ECE	199Y1A0474	Yes	Yes	Strongly agree	Agree	Strongly agree	Yes	4	Good
25	199y1a04a5@ksrmce.ac.in	M. Pranitha	B.Tech VI Sem	ECE	199y1A04A5	Yes	Yes	agree	Agree	Strongly agree	Yes	5	Good
26	199y1a04a6@ksrmce.ac.in	MURARI SRAVANTHI	B.Tech VI Sem	ECE	199Y1A04A6	Yes	Yes	agree	Agree	Strongly agree	Yes	5	Nothing
27	199y1a04a7@ksrmce.ac.in	M.V.SUBRAMANYAM	B.Tech VI Sem	ECE	199Y1A04A7	Yes	Yes	agree	Agree	Strongly agree	Yes	5	no
28	199y1a04a9@ksrmce.ac.in	NALLABOTHULA POORNIMA NAMALA JYOTHIKA GNANA CHANDRIKA	B.Tech VI Sem	ECE	199Y1A04A9	Yes	Yes	agree	Agree	Strongly agree	Yes	4	no
29	199y1a04b0@ksrmce.ac.in	NANDIPATI MANEESHA	B.Tech VI Sem	ECE	199Y1A04B0	Yes	Yes	Strongly agree	Agree	Strongly agree	Yes	4	no
30	199y1a04b1@ksrmce.ac.in	P. LINGESWARI	B.Tech VI Sem	ECE	199Y1A04C0	Yes	Yes	Strongly agree	Agree	Strongly agree	Yes	4	nothing
31	199y1a04c4@ksrmce.ac.in	P.S.KOUSHIK	B.Tech VI Sem	ECE	199Y1A04C4	Yes	Yes	agree	Agree	Strongly agree	Yes	5	Nothing
33	199y1a04d1@ksrmce.ac.in	SADIVE LAKSHMI PRASANNA	B.Tech VI Sem	ECE	199Y1A04D1	Yes	Yes	agree	Agree	Strongly agree	Yes	4	no
34	199y1a04f9@ksrmce.ac.in	TADIMARRI MOUNIKA	B.Tech VI Sem	ECE	199Y1A04F9	Yes	Yes	agree	Agree	Strongly agree	Yes	4	Nothing
35	199y1a04h2@ksrmce.ac.in	Vanam vishnu naga vardhan reddy	B.Tech VI Sem	ECE	199Y1A04H2	Yes	Yes	agree	Agree	Strongly agree	Yes	4	Good


Coordinator


HOD

Professor & H.O.D.
Department of E.C.E.
K.S.R.M. College of Engineering
KADAPA - 516 003

K.S.R.M. COLLEGE OF ENGINEERING (AUTONOMOUS), KADAPA-516003
DEPARTMENT OF ELECTRONICS AND COMMUNICATION ENGINEERING
VALUE ADDED/CERTIFICATE COURSE ON
DIGITAL SYSTEM DESIGN USING VERILOG HDL FROM 23/03/2022 TO 08/04/2022

AWARD LIST

S.No	Roll Number	Name of the Student	Marks Obtained
1.	199Y1A0403	ANGAJALA KAVYA SREE	14
2.	199Y1A0407	AYYALURI VENKATA PAVAN KUMAR REDDY	13
3.	199Y1A0412	BARIVENKULA SREENATH	15
4.	199Y1A0413	BINGIMALLA VENKATA THARUN KUMAR	16
5.	199y1a0415	BOOSI VENKATA SAINATH REDDY	14
6.	199Y1A0418	C.TEJESH KUMAR REDDY	15
7.	199Y1A0422	CHEEPATI VARAPRASAD REDDY	15
8.	199y1a0425	CHERUVU SAI PRAKASH REDDY	16
9.	199y1a0426	CHILUMURU GEETHA PRIYA	11
10.	199y1a0429	Chinnireddy Neetha	05
11.	199Y1A0432	DANDE MOUNIKA	12
12.	199Y1A0434	DESURI VARSHINI	13
13.	199Y1A0441	Anushka Gadde	06
14.	199Y1A0442	Gajjala Navya Tejasree	09
15.	199Y1A0443	GAMPA SIVA	18
16.	199Y1A0447	GODI KAVEETHA	15
17.	199Y1A0451	GOURIPEDDI K S SREEDHANYA	14
18.	199Y1A0452	GULYAM SHARATH	13
19.	199y1a0455	Jampala Anjali	12
20.	199y1a0457	K. ANUSHA	12

21.	199Y1A0458	K S Gowthami	13
22.	199Y1A0459	SOWMYA PRIYA. K	12
23.	199Y1A0472	K.prathyusha	14
24.	199Y1A0474	KONDURU SUMITHRA	14
25.	199Y1A04A5	M. Pranitha	15
26.	199Y1A04A6	MURARI SRAVANTHI	09
27.	199Y1A04A7	M.V.SUBRAMANYAM	14
28.	199Y1A04A9	NALLABOTHULA POORNIMA	13
29.	199Y1A04B0	NAMALA JYOTHIKA GNANA CHANDRIKA	11
30.	199Y1A04B1	NANDIPATI MANEESHA	13
31	199Y1A04C0	P. LINGESWARI	16
32	199Y104C4	P.S.KOUSHIK	17
33	199Y1A04D1	SADIVE LAKSHMI PRASANNA	11
34	199Y1A04F9	TADIMARRI MOUNIKA	13
35	199Y1A04H2	Vanam vishnu naga vardhan reddy	12

Coordinator

HoD

Professor & H.O.D.
Department of E.C.E.
K.S.R.M. College of Engineering
KADAPA - 518 023

K.S.R.M. COLLEGE OF ENGINEERING (AUTONOMOUS), KADAPA-516003
DEPARTMENT OF ELETRONICS AND COMMUNICATIONS OF ENGINEERING
VALUE ADDED /CERTIFICATE COURSE ON
DIGITAL SYSTEM DESIGN USING VERILOG HDL FROM 23/03/2022 TO 08/04/2022
ASSESSMENT TEST

Roll Number: _____ Name of the Student: _____

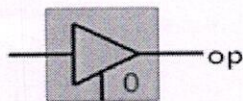
Time: 20 Min (Objective Questions) **Max.Marks: 20**

Note: Answer the following Questions and each question carries one mark.

1 Which among the following is a process of transforming RTL to a gate-level netlist? []

A. Simulation B. Optimization C. Synthesis D. Verification

2. []



The output of the following logic =?

A. Logic 0 B. Logic 1 C. X D. Z

3. #40 \$finish indicates []

A. End of simulation time B. End of simulation at 40-time units
C. Suspend simulation at 40-time units D. None

4. The characteristic equation of 'T' Flip flop is given by []

A. $Q(n+1) = T Q + T'Q'$ B. $Q(n+1) = T + Q'T'$ C. T D. None

5. The hexadecimal number 'A0' has the decimal value equivalent to: []

A. 80 B. 256 C. 100 D. 160

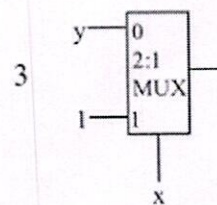
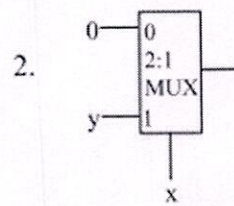
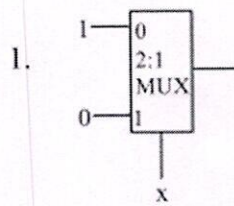
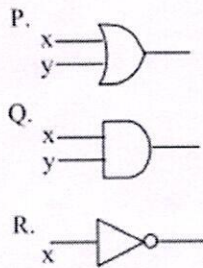
12. A 2-input XOR gate can be worked as an Inverter (NOT Gate) if a=? & b=? []

- A. $A = a, b = 0$ B. $A = a, b = 1$ C. $A = b, b = a$ D. None

13. Which model uses transistors as their basic components? []

- A. Switch level B. Gate Level C. Circuit Level D. Layout Level

14. []



Match the following:

- A. P-3 Q-2 R-1 B. P-3 Q-1 R-2 C. P-2 Q-3 R-1 D. None

15. What is the default value of the reg data type? []

- A. 0 B. 1 C. Z D. X

16. The Gray code for decimal number 6 is equivalent to: []

- A. 1100 B. 1001 C. 0101 D. 0110

17. The Boolean expression $(A'B+AB'+AB)$ is equivalent to: []

- A. $A+B$ B. $A'+B$ C. $(A+B)'$ D. $A.B$

18. What is the time period of clock #20 clock = ~clock?

[]

A. 10

B. 20

C. 40

D. 50

19. Verilog is case sensitive.

[]

A. True

B. False

20. The FPGA is abbreviated as:

[]

A. First programmable Gate Array

B. Field Programmable Gate Array

C. Field Post Gate Array

D. Field Program Gate Array

16/20

K.S.R.M. COLLEGE OF ENGINEERING (AUTONOMOUS), KADAPA-516003
DEPARTMENT OF ELECTRONICS AND COMMUNICATIONS OF ENGINEERING
VALUE ADDED /CERTIFICATE COURSE ON
DIGITAL SYSTEM DESIGN USING VERILOG HDL FROM 23/03/2022 TO 08/04/2022

ASSESSMENT TEST

Roll Number: 199Y1AD425 Name of the Student: C. Sai Prakash Reddy

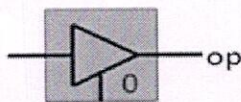
Time: 20 Min (Objective Questions) **Max.Marks: 20**

Note: Answer the following Questions and each question carries one mark.

1 Which among the following is a process of transforming RTL to a gate-level netlist? [C]

- A. Simulation B. Optimization C. Synthesis D. Verification

2.



The output of the following logic =?

- A. Logic 0 B. Logic 1 C. X D. Z

3. #40 \$finish indicates

- A. End of simulation time B. End of simulation at 40-time units
C. Suspend simulation at 40-time units D. None

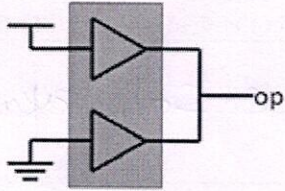
4. The characteristic equation of 'T' Flip flop is given by

- A. $Q(n+1) = TQ + T'Q'$ B. $Q(n+1) = T + Q'T'$ C. T D. None

5. The hexadecimal number 'A0' has the decimal value equivalent to:

- A. 80 B. 256 C. 100 D. 160

6.



[C]



The output of the following logic =?

- A. Logic 0 B. Logic 1 C. X D. Z

7. $(734)_8 = ()_{16}$

- A. CID B. DC1 C. 1CD D. IDC

8. The full form of VLSI is _____.

- A. Very Long Single Integration B. Very Least Scale Integration
C. Very Large Scale Integration D. Very Long Scale Integration

9. VHDL stands for

- A. Very High-Speed Integrated Circuit Hardware Description Language
B. Very High Description Language
C. Verilog Hardware Description Language
D. None

10. The digital logic family which has minimum power dissipation is:

- A. TTL B. RTL C. DTL D. CMOS

11. ASIC stands for:

- A. Application speedy integrated circuit B. Application-specific integrated circuit
C. Advanced speed integrated circuit D. Advanced standard integrated circuit

[B]

[D]

[C]

[C]

[B]

12. A 2-input XOR gate can be worked as an Inverter (NOT Gate) if a=? & b=?

[A]

- A. $A = a, b = 0$ B. $A = a, b = 1$ C. $A = b, b = a$ D. None

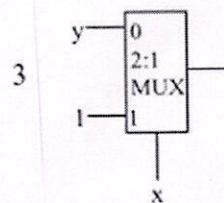
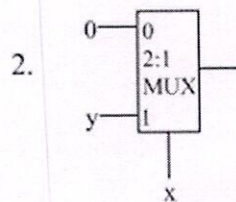
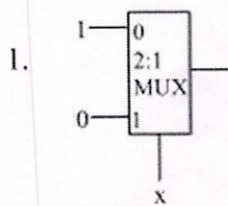
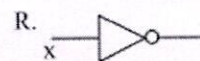
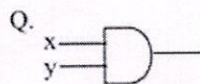
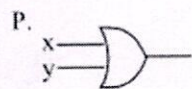
13. Which model uses transistors as their basic components?

[A]

- A. Switch level B. Gate Level C. Circuit Level D. Layout Level

14.

[B]



Match the following:

- A. P-3 Q-2 R-1 B. P-3 Q-1 R-2 C. P-2 Q-3 R-1 D. None

15. What is the default value of the reg data type?

[C]

- A. 0 B. 1 C. Z D. X

16. The Gray code for decimal number 6 is equivalent to:

[A]

- A. 1100 B. 1001 C. 0101 D. 0110

17. The Boolean expression $(A'B + AB' + AB)$ is equivalent to:

[C]

- A. $A+B$ B. $A'+B$ C. $(A+B)'$ D. $A.B$

18. What is the time period of clock #20 clock = ~clock?

A. 10

B. 20

C. 40

D. 50

[C]

19. Verilog is case sensitive.

A. True

B. False

[A]

20. The FPGA is abbreviated as:

A. First programmable Gate Array

B. Field Programmable Gate Array

C. Field Post Gate Array

D. Field Program Gate Array

[B]

18/20

K.S.R.M. COLLEGE OF ENGINEERING (AUTONOMOUS), KADAPA-516003
DEPARTMENT OF ELETRONICS AND COMMUNICATIONS OF ENGINEERING
VALUE ADDED /CERTIFICATE COURSE ON
DIGITAL SYSTEM DESIGN USING VERILOG HDL FROM 23/03/2022 TO 08/04/2022
ASSESSMENT TEST

Roll Number: 19941AD443 Name of the Student: G. Siva

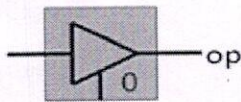
Time: 20 Min (Objective Questions) **Max.Marks: 20**

Note: Answer the following Questions and each question carries **one** mark.

1 Which among the following is a process of transforming RTL to a gate-level netlist? [C]

- A. Simulation B. Optimization C. Synthesis D. Verification

2. [D]



The output of the following logic =?

- A. Logic 0 B. Logic 1 C. X D. Z

3. #40 \$finish indicates [A]

- A. End of simulation time B. End of simulation at 40-time units
C. Suspend simulation at 40-time units D. None

4. The characteristic equation of 'T' Flip flop is given by [D]

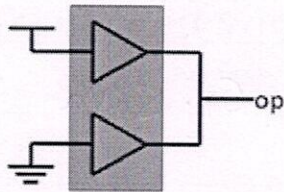
- A. $Q(n+1) = T Q + T'Q'$ B. $Q(n+1) = T + Q'T'$ C. T D. None

5. The hexadecimal number 'A0' has the decimal value equivalent to: [D]

- A. 80 B. 256 C. 100 D. 160

6.

[C] ✓



The output of the following logic =?

- A. Logic 0 B. Logic 1 C. X D. Z

7. $(734)_8 = ()_{16}$

[C] ✓

- A. C1D B. DC1 C. 1CD D. IDC

8. The full form of VLSI is _____.

[A] ✓

- A. Very Long Single Integration B. Very Least Scale Integration
C. Very Large Scale Integration D. Very Long Scale Integration

9. VHDL stands for

[D] ✓

- A. Very High-Speed Integrated Circuit Hardware Description Language
B. Very High Description Language
C. Verilog Hardware Description Language
D. None

10. The digital logic family which has minimum power dissipation is:

[B] ✓

- A. TTL B. RTL C. DTL D. CMOS

11. ASIC stands for:

[B] ✓

- A. Application speedy integrated circuit B. Application-specific integrated circuit
C. Advanced speed integrated circuit D. Advanced standard integrated circuit

12. A 2-input XOR gate can be worked as an Inverter (NOT Gate) if a=? & b=?

[A]

- A. $A = a, b = 0$ B. $A = a, b = 1$ C. $A = b, b = a$ D. None

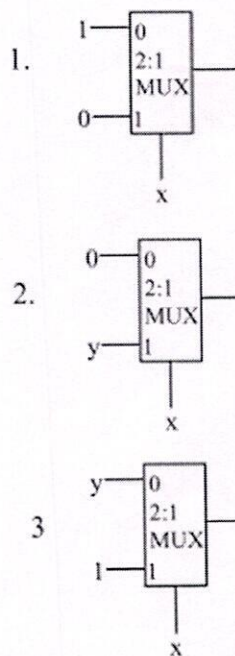
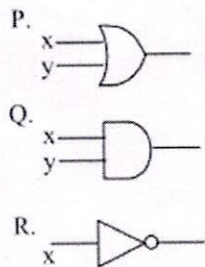
13. Which model uses transistors as their basic components?

[A]

- A. Switch level B. Gate Level C. Circuit Level D. Layout Level

14.

[D]



Match the following:

- A. P-3 Q-2 R-1 B. P-3 Q-1 R-2 C. P-2 Q-3 R-1 D. None

15. What is the default value of the reg data type?

[C]

- A. 0 B. 1 C. Z D. X

16. The Gray code for decimal number 6 is equivalent to:

[A]

- A. 1100 B. 1001 C. 0101 D. 0110

17. The Boolean expression $(A'B + AB' + AB)$ is equivalent to:

[B]

- A. $A+B$ B. $A'+B$ C. $(A+B)'$ D. $A.B$

18. What is the time period of clock #20 clock = ~clock? [C]

- A. 10 B. 20 C. 40 D. 50

19. Verilog is case sensitive. [A]

- A. True B. False

20. The FPGA is abbreviated as: [B]

- A. First programmable Gate Array B. Field Programmable Gate Array
C. Field Post Gate Array D. Field Program Gate Array

Introduction

- Thirty years ago, the primary tools of a digital designer included a logic-drawing template, a ruler, and a pencil.
- In the 1990s, HDL (Hardware Description Language) usage by digital system designers accelerated as PLD, CPLD, and FPGAs became inexpensive and commonplace.
- At the same time (1990s), as ASIC densities continued to increase, it became increasingly more difficult to describe large circuits using schematics alone, and many ASIC designers turned to HDLs as a means to design individual modules within a system-on-a-chip (SoC).

2

Certificate course-DSD using Verilog HDL

HDL-Based Digital Design

HDL Tool Suites

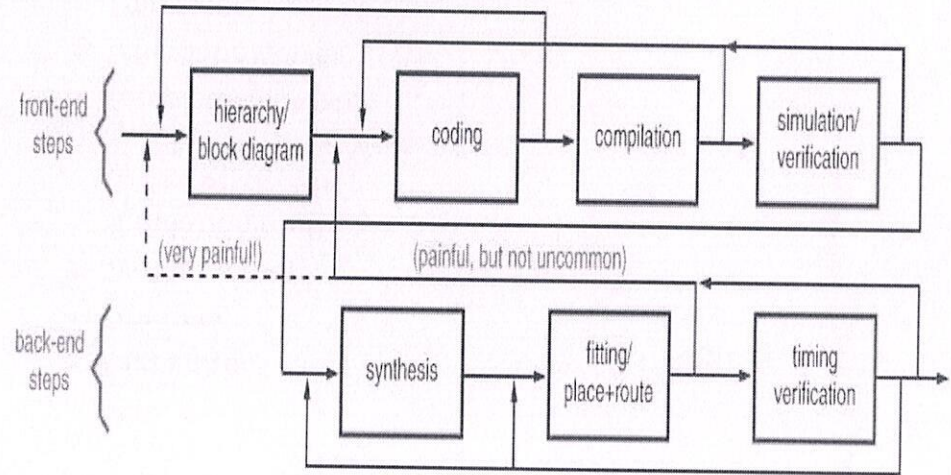
- **Text editor**
- **Compiler**
- **Synthesis tool**
- **Simulation tool**
 - Test bench
 - Waveform editor/viewer
- **Template generator**
- **Schematic viewer**
- **Translator**
- **Timing analyzer**
- **Back annotator**

4

Major HDL Tool Suites

- Xilinx ISE
- Altera Quartus II

HDL-Based Design Flow



Hardware Description Languages (HDLs)

- **Benefits**
 - Complete, unambiguous specification of design
 - Inputs, outputs, behavior, timing
 - Simulation for design debug
 - Synthesis for physical realization of design
- **Several widely used**
 - ABEL – primarily for PLDs
 - VHDL (VHSIC Hardware Description Language)
 - Verilog (1995, 2001)
- **Numerous tools and vendors**
 - Capture, edit, simulate, synthesize...

The Verilog HDL

Introduction

- *Verilog HDL*, or simply *Verilog*, was introduced by Gateway Design Automation in 1984 as a proprietary hardware description and simulation language.
- The introduction of Verilog-based synthesis tools in 1988 by then-fledgling Synopsys and the 1989 acquisition of Gateway by Cadence Design Systems were important events that led to widespread use of the language.
- IEEE standard
 - *Verilog-1995*: Verilog became IEEE standard in 1995.
 - *Verilog-2001*: IEEE updated Verilog standard in 2001.

9

Introduction (cont'd)

- Many books treat Verilog as a programming language, which is not the best way of viewing it.
- A common error among beginners is to write a program without thinking about the hardware that is implied.
- If you do not know what hardware you are implying, you are almost certain to get something that you did not want.
- Two general styles of description:
 - Behavioral
 - structural

11

Introduction (cont'd)

- *Verilog synthesis tools* can create logic-circuit structures directly from Verilog behavioral descriptions, and target them to a selected technology for realization.
- Using Verilog, you can design, simulate, and synthesize anything from a simple combinational circuit to a complete microprocessor system on a chip.
- Like VHDL, Verilog started out as a documentation and modeling language, allowing the behavior of digital-system designs to be precisely specified and simulated. But as with VHDL, synthesis tools are what led to widespread use of Verilog.

10

Features of Verilog

- Designs may be decomposed hierarchically.
- Each design element has both a well-defined interface (for connecting it to other elements) and a precise functional specification (for simulating it).
- Functional specifications can use either a behavioral algorithm or an actual hardware structure to define an element's operation. For example, an element can be defined initially by an algorithm, to allow design verification of higher-level elements that use it; later, the algorithmic definition can be replaced by a preferred hardware structure.
- Concurrency, timing, and clocking can all be modeled. Verilog handles asynchronous as well as synchronous sequential-circuit structures.
- The logical operation and timing behavior of a design can be simulated.

12

Verilog vs. VHDL

- “Both languages are easy to learn and hard to master. Once you have learned one of these languages, you will have no trouble transitioning to the other.”

– David Pellerin and Douglas Taylor, *VHDL Made Easy!* (Prentice Hall, 1997)

- “It is hard to go back and forth between the two on a daily or even weekly basis. So my advice is to learn one well and, only if necessary, tackle the other later”

– authors of *Digital Design*, 4th edition (Prentice Hall, 2007)

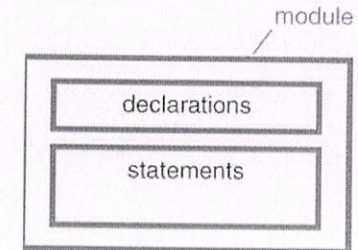
13

Verilog Module

- The basic unit of design and programming in Verilog is a **module**.
- Verilog statements can specify a module’s operation **behaviorally** and **structurally**.
- One module per file, please.

Different types of module descriptions

1. Structural – actual gates
2. Behavioral – behavior, no structure
3. Dataflow – simple output F (inputs)
4. Combination of above



14

Verilog Module

- Verilog modules can use a mix of behavioral and structural specifications, and may do so **hierarchically**.
- Just as procedures and functions in a high-level software programming language can “call” others, Verilog modules can “instantiate” other modules.
- A higher-level module may use a lower-level module multiple times, and multiple top-level modules may use the same lower-level one.
- In Verilog, the scope of signal, constant, and other definitions remains local to each module; values can be passed between modules only by using declared input and output signals.

15

Verilog Modules

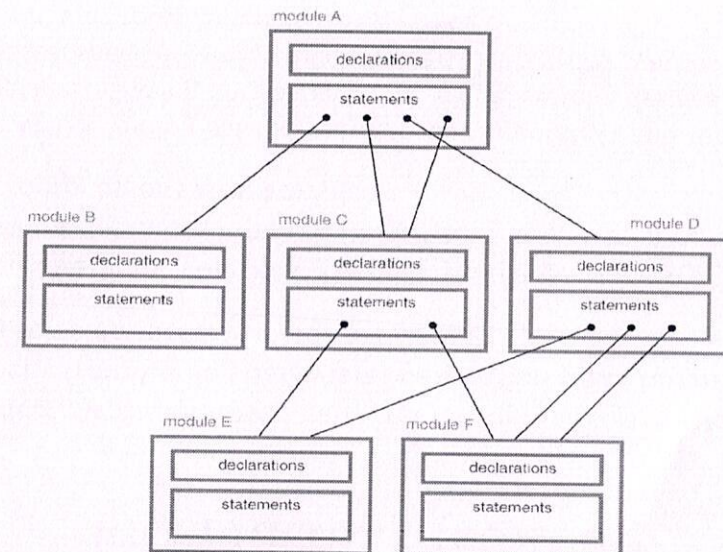


Fig : Modules instantiating other modules hierarchically

16

Verilog Syntax

- Case sensitive
 - "module", "Module", "MODULE" not same
 - Reserved words are lower case (e.g. "module")
 - User-defined identifiers (e.g. variables, modules, ports) no requirements
 - Useful convention: use mixed case, leading capitals
 - "BufferFull" not "bufferfull" or "BUFFERFULL"
- White space (e.g. spaces, line breaks) generally ignored
 - Use it to make your modules readable
- Comments
 - // single line comments
 - /* multiple line comments */

This is dataflow type of description



```

module VrInhibit( X, Y, Z ); // also known as 'BUT-NOT'
  input X, Y;                // as in 'X but not Y'
  output Z;                  // (see [Klir, 1972])
  assign Z = X & ~Y;
endmodule
    
```

17

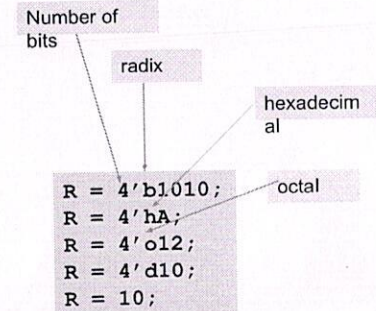
Comments

- Verilog uses C-style comments
- Single-line comments (//)
- Multi-line comments (/* and */)

19

Verilog Syntax

- Identifiers
 - Begin with letter or underscore
 - Contain letters, digits, underscores, \$
- 4 Valued Logic
 - 0, 1, X, Z
- Literals (in language sense)
 - n'Bddd...d
 - n = (decimal) number of bits
 - B = radix
 - b = binary
 - o = octal
 - h = hexadecimal
 - d = decimal
 - ddd...d = digits in designated radix

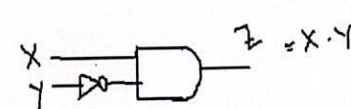


18

A Simple Verilog Program

```

module VrInhibit( X, Y, Z ); // also known as 'BUT-NOT'
  input X, Y;                // as in 'X but not Y'
  output Z;                  // (see [Klir, 1972])
  assign Z = X & ~Y;
endmodule
    
```



20

A Simple Verilog Program

- User-defined *identifiers* begin with a letter or underscore and can contain letters, digits, underscores (`_`), and dollar signs (`$`).
- Identifiers that start with a dollar sign refer to built-in system functions.
- Identifiers in the example: `VrInhibit`, `X`, `Y`, and `Z`.
- Verilog is case sensitive.
 - However, it is best not to rely on case alone to distinguish two different identifiers.

21

Syntax of Module Declaration

```
module module-name (port-name, port-name, ..., port-name);  
    input declarations  
    output declarations  
    inout declarations  
    net declarations  
    variable declarations  
    parameter declarations  
    function declarations  
    task declarations  
  
    concurrent statements  
endmodule
```

22

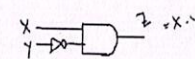
Syntax of Input/Output Declarations

```
input identifier, identifier, ..., identifier;  
output identifier, identifier, ..., identifier;  
inout identifier, identifier, ..., identifier;  
  
input [msb:lsb] identifier, identifier, ..., identifier;  
output [msb:lsb] identifier, identifier, ..., identifier;  
inout [msb:lsb] identifier, identifier, ..., identifier;
```

23

A Simple Verilog Program

```
module VrInhibit( X, Y, Z ); // also known as 'BUT-NOT'  
    input X, Y; // as in 'X but not Y'  
    output Z; // (see [Klir, 1972])  
  
    assign Z = X & ~Y;  
endmodule
```



- **X, Y, and Z** are the **ports** of the module `VrInhibit`.
- **input**: The signal is an input to the module.
- **output**: The signal is an output of the module.
- **inout**: The signal can be used as an input or an output.

24

Operators in Verilog

- Operators

- Bit-wise Boolean
- Arithmetic and shift
- Logical
- Misc: concatenation, replication

Operator	Operation
&&	logical AND
	logical OR
!	logical NOT
==	logical equality
!=	logical inequality
>	greater than
>=	greater than or equal
<	less than
<=	less than or equal

Operator	Operation
&	AND
	OR
^	Exclusive OR (XOR)
~^, ^~	Exclusive NOR (XNOR)
~	NOT

Operator	Operation
+	Addition
-	Subtraction
*	Multiplication
/	Division
%	Modulus (remainder)
<<	Shift left
>>	Shift right

```

logical-expression ? true-expression : false-expression
X ? Y : Z
(A>B) ? A : B;
(sel==1) ? op1 : (
  (sel==2) ? op2 : (
    (sel==3) ? op3 : (
      (sel==4) ? op4 : 8'bx )))
  
```

reg

- A **reg** variable is a single bit or a vector of bits.
- The main use of **reg** variables is to store values in Verilog procedural code (e.g., **always** block).

Verilog Variable Types

- The most commonly used Verilog variable types are **reg** and **integer**.
- The difference between Verilog's nets and variables is subtle. A variable's value can be changed only within procedural code within a module; it cannot be changed from outside the module.
- Thus, input and inout ports cannot have a variable type; they must have a net type such as **wire**.
- Output ports, on the other hand, can have either a net or a **reg** type, and can drive the input and inout ports of other modules.
- Another important difference is that procedural code can assign values only to variables.

Syntax of Vector Signals

- "vector" signals
 - syntax: [*msb*:*lsb*]
 - e.g., [7:0], [0:7], and [13:20] are all valid 8-bit ranges.

Vectors in Verilog

- Vectors

- Signals or ports with width

- Bit select

- Zbus[5]

- Part select

- word1[15:8]

```

input  identifier, identifier, ..., identifier;
output identifier, identifier, ..., identifier;
inout  identifier, identifier, ..., identifier;

input  [msb:lsb] identifier, identifier, ..., identifier;
output [msb:lsb] identifier, identifier, ..., identifier;
inout  [msb:lsb] identifier, identifier, ..., identifier;
    
```

```

reg [7:0] byte1, byte2, byte3;
reg [15:0] word1, word2;
reg [1:16] zbus;
    
```

Logic Values & Bitwise Boolean Operators

- 0 : Logical 0, or false
- 1 : Logical 1, or true
- x : An unknown logical value
- z : High impedance, as in three-state logic

Operator	Operation
&	AND
	OR
^	Exclusive OR (XOR)
^^, ^^	Exclusive NOR (XNOR)
~	NOT

Constants (Numeric Literals)

- *n' Bdd...d*

- *n* : A decimal number indicating the number of bits represented, **not the number of digits dd...d**.

- *B* : A single letter specifying the base

- b: binary
 - o: octal
 - h: hexadecimal
 - d: decimal and the default

- *dd...d* : A string of one or more digits in the specified base.

Constants – Examples

Number	# Bits	Base	Decimal Equivalent	Stored
3'b101	3	Binary	5	101
'b11	unsized	Binary	3	000000..00011
8'b11	8	Binary	3	00000011
8'b1010_1011	8	Binary	171	10101011
3'd6	3	Decimal	6	110
6'o42	6	Octal	34	100010
8'hAB	8	Hexadecimal	171	10101011
42	unsized	Decimal	42	0000...00101010

Constants

- It is good practice to specify the length of the numbers in bits, even though this is not strictly necessary.
- Underscores in numbers are ignored and can be helpful in breaking long numbers into more readable chunks.
- If the base is omitted, the number is assumed to be decimal.

33

Delays

All delays in a Verilog HDL model are specified in terms of time units. Here is an example of a continuous assignment with a delay.

```
assign #2 Sum = A ^ B;
```

The #2 refers to 2 time units.

The association of a time unit with physical time is made using the ``timescale` compiler directive. Such a directive is specified before a module declaration. An example of such a directive is:

```
`timescale 1ns / 100ps
```

which says that one time unit is to be treated as 1ns and that the time precision is to be 100ps (the time precision says that all delays must be rounded to 0.1ns). If this compiler directive is present in the module containing the above continuous assignment, the #2 refers to 2ns.

5.4.6 Compiler Directives

A Verilog compiler provides several directives for controlling compilation; we'll introduce two of them here. All compiler directives begin with an accent grave (`). First is the ``include` compiler directive, with the syntax below:

```
`include filename
```

The named file is read immediately and processed as if its contents were part of the current file. This facility is typically used to read in definitions that are common to multiple modules in a project. Nesting is allowed; that is, an `include'd` file can contain ``include` directives of its own.

Next is the ``define` compiler directive, with the syntax below:

```
`define identifier text
```

Notice that there is no ending semicolon. The compiler textually replaces each appearance of `identifier` in subsequent code with `text`. Remember, this is a *textual* substitution; no expression evaluation or other processing takes place. Also, it is important to know that the definition is in effect not only in the current file, but in subsequent files that are processed during a given compiler run (for example, in files that are `include'd` by this one).

34

Verilog data flow design model

Syntax for continuous assignment statement

Table 5-74

Syntax of Verilog
continuous-assignment
statements.

```
assign net-name = expression;  
assign net-name[bit-index] = expression;  
assign net-name[msb:lsb] = expression;  
assign net-concatenation = expression;
```

35

Here is a simple example of a module that models the half-adder circuit shown in Figure 2-1.

```

module HalfAdder (A, B, Sum, Carry);
  input A, B;
  output Sum, Carry;

  assign #2 Sum = A ^ B;
  assign #5 Carry = A & B;
endmodule

```

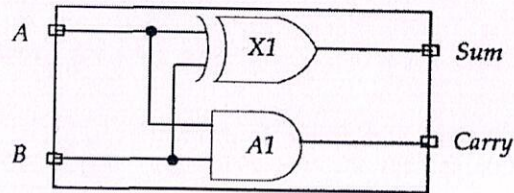


Figure 2-1 A half-adder circuit.

32-bit Adder

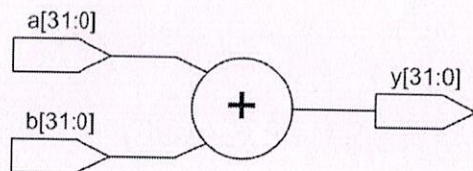
- A 32-bit adder is a good example of combinational logic.

```

module adder( input [31:0] a,
               input [31:0] b,
               output [31:0] y );

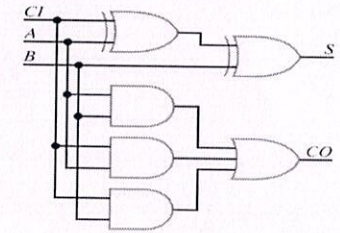
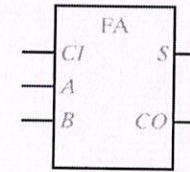
  assign y = a + b;
endmodule

```



Verilog Dataflow Example: Full Adder

Ci	A	B	Co	S
0	0	0	0	0
0	0	1	0	1
0	1	0	0	1
0	1	1	1	0
1	0	0	0	1
1	0	1	1	0
1	1	0	1	0
1	1	1	1	1



$$S = A \oplus B \oplus CI$$

```

module FullAdder(S, CO, A, B, CI);
  input A,B, CI;
  output S, CO;

  assign S = A ^ B ^ CI;
  assign CO = A & B | A & CI | B & CI;
endmodule

```

$$CO = A \cdot B + A \cdot CI + B \cdot CI$$

Bitwise Operator - Inverters

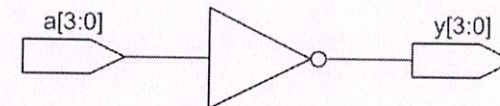
- The following module describes four inverters.

```

module inv( input [3:0] a,
             output [3:0] y );

  assign y = ~a;
endmodule

```



Bitwise Operator – Logic Functions

```
module gates( input  [3:0] a, b,
              output [3:0] y1, y2, y3, y4, y5 );

  /* Five different two-input logic
     gates acting on 4 bit busses */
  assign y1 = a & b;      // AND
  assign y2 = a | b;      // OR
  assign y3 = a ^ b;      // XOR
  assign y4 = ~(a & b);  // NAND
  assign y5 = ~(a | b);  // NOR
endmodule
```

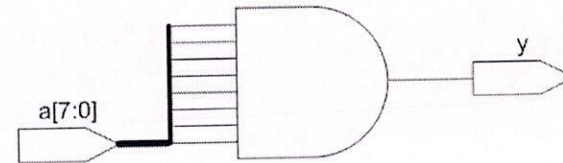
41

Reduction Operator – AND Gate

- The following module describes an 8-input AND gate.

```
module and8( input  [7:0] a,
            output   y );

  assign y = &a;
endmodule
```

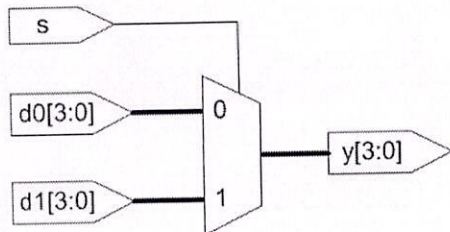


42

Multiplexer – An Example

```
module mux2( input  [3:0] d0, d1,
            input   s,
            output [3:0] y );

  assign y = s ? d1 : d0;
endmodule
```



43

Multiplexer – Another Example

```
module Vrchap1mux(A, B, S, Z);
  input A, B, S;
  output Z;

  assign Z = (S==0) ? A : B;
endmodule
```

44

Operator Precedence

Table A.1 Operator Precedence

Symbol	Meaning	Precedence
~	NOT	Highest
*, /, %	MUL, DIV, MODULO	
+, -	PLUS, MINUS	
<<, >>	Logical Left/Right Shift	
<<<, >>>	Arithmetic Left/Right Shift	
<, <=, >, >=	Relative Comparison	
==, !=	Equality Comparison	
&, ~&	AND, NAND	
^, ~^	XOR, XNOR	
, ~	OR, NOR	
?:	Conditional	Lowest

45

Verilog Built-in Gates

and	xor	bufif0
nand	xnor	bufif1
or	buf	notif0
nor	not	notif1

46

Verilog has several *built-in gate types*, shown in Table 5-68. The names of these gates are reserved words. The `and`, `or`, and `xor` gates and their complements may have any number of inputs. A `buf` gate is a 1-input noninverting buffer, and a `not` gate is an inverter.

The remaining four gates are 1-input buffers and inverters with three-state outputs. They drive the output with the data input (or its complement) if the enable input is 0 or 1, as in the gate's name; else the output is `z`. For example, `bufif0` drives its output with its data input if the control input is 0.

A typical design environment has libraries that provide many other predefined components such as AND-OR-INVERT gates, flip-flops, and higher-

47

Wire

- Internal Signals

48

Table 5-59
Verilog net types.

wire	trior	trireg	supply0
tri	tri0	wand	supply1
triand	tri1	wor	

```
wire identifier, identifier, ..., identifier;
wire [msb:lsb] identifier, identifier, ..., identifier;

tri identifier, identifier, ..., identifier;
tri [msb:lsb] identifier, identifier, ..., identifier;

reg identifier, identifier, ..., identifier;
reg [msb:lsb] identifier, identifier, ..., identifier;

integer identifier, identifier, ..., identifier;
```

Verilog Instantiation and parameter lists(structural model)

- Instantiating a module
 - component name
 - instance identifier
- Two forms of parameter lists
 - By order of declaration
 - By explicit port name

```
component-name instance-identifier ( expr, expr, ..., expr );

component-name instance-identifier ( .port-name(expr),
                                       .port-name(expr),
                                       ...,
                                       .port-name(expr) );
```

Verilog Structural Example: Full Adder

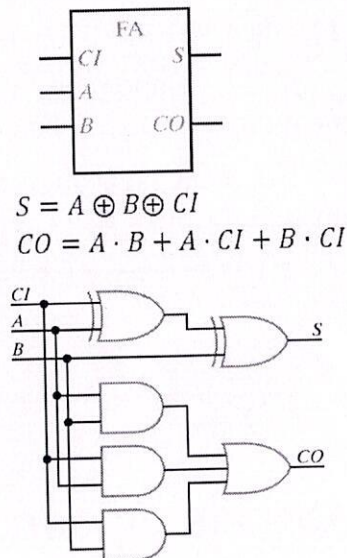
```
module FullAdder(S, CO, A, B, CI);
input A,B, CI;
output S, CO;

wire X1;
wire P1,P2,P3;

xor
  U1a(X1,A,CI),
  U1b(S,X1,B);

and
  U2a(P1,A,B),
  U2b(P2,A,CI),
  U2c(P3,B,CI);

or
  U3a(CO,P1,P2,P3);
endmodule
```



Verilog Structural Example: Full Adder

$$S = A \oplus B \oplus CI \quad CO = A \cdot B + A \cdot CI + B \cdot CI$$

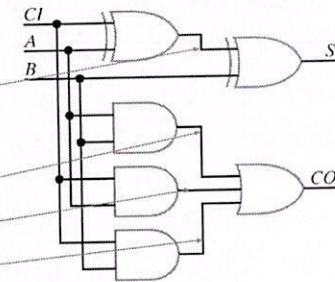
```
module FullAdder(S, CO, A, B, CI);
input A,B, CI;
output S, CO;

wire X1;
wire P1,P2,P3;

xor
  U1a(X1,A,CI),
  U1b(S,X1,B);

and
  U2a(P1,A,B),
  U2b(P2,A,CI),
  U2c(P3,B,CI);

or
  U3a(CO,P1,P2,P3);
endmodule
```



Behavioral Model(procedural code)

Syntax for always block

```
always @ ( signal-name or signal-name or ... or signal-name )  
    procedural-statement
```

```
always procedural-statement
```

- Procedural statements in an **always** block execute sequentially, as in software programming languages.
- However, the **always** block itself executes concurrently with other concurrent statements in the same module.
- If one of the procedural statements changes the value of a net or variable that is used in another concurrent statement, it may cause that statement to be re-executed.
- Sensitivity list

53

always block – Sensitivity List

- A Verilog concurrent statement such as an **always** block is always either executing or suspended.
- A concurrent statement initially is suspended; when any signal in its sensitivity list changes value, it (the **always** block) resumes execution, starting with its first procedural statement and continuing until the end.
- If any signal in the sensitivity list changes value as a result of executing the concurrent statement, it executes again. This continues until the statement executes without any of these signals changing value at the current time.

54

```
begin  
    procedural-statement  
    ...  
    procedural-statement  
end
```

```
begin : block-name  
    variable declarations  
    parameter declarations
```

```
    procedural-statement  
    ...  
    procedural-statement  
end
```

Table 5-81

Syntax of Verilog
begin-end blocks.

Table 5-86

Syntax of a Verilog
case statement.

```
case ( selection-expression )  
    choice , ... , choice : procedural-statement  
    ...  
    choice , ... , choice : procedural-statement  
    default : procedural-statement  
endcase
```

Table 5-90

Syntax of a Verilog
for statement.

```
for ( loop-index = first-expr ; logical-expression ; loop-index = next-expr )  
    procedural-statement
```

```
for ( loop-index = first; loop-index <= last; loop-index = loop-index + 1; )  
    procedural-statement
```

```
if ( condition ) procedural-statement
```

```
if ( condition ) procedural-statement  
else procedural-statement
```

Table 5-83

Syntax of Verilog
if statements.

The other Verilog looping statements are `repeat`, `while`, and `forever`, with the syntax shown in Table 5-92. A *repeat statement* repeats a procedural statement a number of times determined by *integer-expression*. A *while statement* repeats a procedural statement until *logical-expression* is false. A *forever statement* repeats a procedural statement "forever."

As in a `for` statement, the procedural statement is usually a begin-end block containing a series of other procedural statements. Any of these looping statements can be temporarily suspended by certain timing control statements that can appear in such a begin-end block.

```
repeat ( integer-expression )
  procedural-statement

while ( logical-expression )
  procedural-statement

forever
  procedural-statement
```

Table 5-92
Syntax of Verilog
`repeat`, `while`, and
`forever` statements.

57

A TestBench

- Testbench provides stimulus to module being tested
- Testbench must obey interface and protocol
 - Interface: signal direction and width (e.g. bus)
 - Protocol: timing and edge relationships
- Testbench facilitates response from module
 - TestBench can be written to independently verify response
 - Human can examine waveforms produced in simulation
 - TestBench can be written to format response (e.g. table) for later examination or processing
- Testbench is a Verilog module!

59

Time Dimension

Delays

All delays in a Verilog HDL model are specified in terms of time units. Here is an example of a continuous assignment with a delay.

```
assign #2 Sum = A ^ B;
```

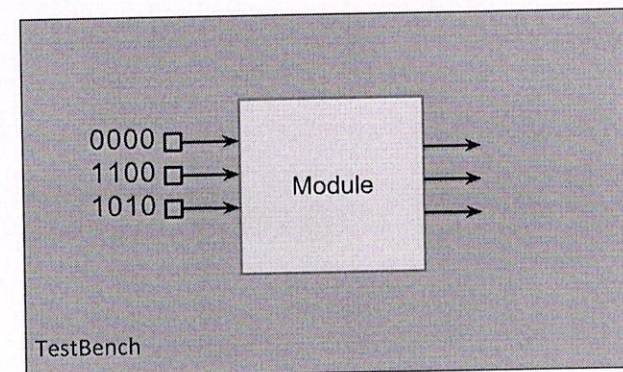
The #2 refers to 2 time units.

The association of a time unit with physical time is made using the ``timescale` compiler directive. Such a directive is specified before a module declaration. An example of such a directive is:

```
`timescale 1ns / 100ps
```

which says that one time unit is to be treated as 1ns and that the time precision is to be 100ps (the time precision says that all delays must be rounded to 0.1ns). If this compiler directive is present in the module containing the above continuous assignment, the #2 refers to 2ns.

A TestBench



- TestBench instantiates module and provides necessary inputs, called test vectors
- Declares registers ("reg") which will be connected to the module input ports
- Declares wires which will be connected to the module output ports

60

A while ago, we promised to introduce one more kind of Verilog concurrent statement, which is typically used in test benches. An *initial* block has the syntax shown in Table 5-98. Like an *always* block, it contains one or more procedural statements, but it does not have a sensitivity list. An *initial* block executes just once, beginning at simulated time zero. As in an *always* block, the *begin-end* block can be named and can have its own variable and parameter declarations.

Table 5-98
Syntax of Verilog
initial blocks.

```

initial
  procedural-statement
initial begin
  procedural-statement
  ...
  procedural-statement
end
  
```

61

Testbench Example for previously presented Full Adder

```

module test();
  reg A,B,CI;
  wire S,CO;

  FullAdder FA0(S,CO,A,B,CI);

  initial
  begin
    A=0; B=0; CI=0;
    #10 A=0; B=0; CI=1;
    #10 A=0; B=1; CI=0;
    #10 A=0; B=1; CI=1;
    #10 A=1; B=0; CI=0;
    #10 A=1; B=0; CI=1;
    #10 A=1; B=1; CI=0;
    #10 A=1; B=1; CI=1;

    #10
    $finish();
  end
endmodule
  
```



```

module FullAdder(S, CO, A, B, CI);
  input A,B, CI;
  output S, CO;

  wire X1;
  wire P1,P2,P3;

  xor
    U1a(X1,A,CI),
    U1b(S,X1,B);

  and
    U2a(P1,A,B),
    U2b(P2,A,CI),
    U2c(P3,B,CI);

  or
    U3a(CO,P1,P2,P3);
endmodule
  
```

62

Blocking and Nonblocking Assignment

- Comparisons of “**assign**”, “**=**”, and “**<=**” statements
- Verilog supports two types of assignments inside an **always** block – **blocking** assignments and **nonblocking** assignments.
- **Blocking** assignments use the “**=**” statement.
 - Can be used in the **always** block.
- **Nonblocking** assignment use the “**<=**” statement.
- Do not confuse either type with the “**assign**” statement, which cannot appear inside **always** blocks at all.

63

Blocking and Nonblocking Assignment (cont’d)

- A group of **blocking** assignments inside a **begin/end** block are evaluated **sequentially**, just as you would expect in a standard programming language.
 - Can be used in the **always** block.
- A group of **nonblocking** assignments are evaluated in **parallel**; all of the statements are evaluated before any of the left sides are updated. This is what you would expect in hardware because real logic gates all operate independently rather than waiting for the completion of other gates.

64

Blocking and Nonblocking Assignment (cont'd)

- Always use **blocking** assignments (=) in **always** blocks intended to create **combinational logic**.
- Always use **nonblocking** assignments (<=) in **always** blocks intended to create **sequential logic**.
- Do not mix blocking and nonblocking assignments in the same **always** block.
- Do not make assignments to the same variable in two different **always** block.

Positive-Edge-Triggered D Flip-Flop Behavioral-Style Verilog Module

```

module VrposDff(CLK, D, Q);
  input CLK, D;
  output Q;
  reg Q;

  always @ (posedge CLK)
    Q <= D;
endmodule

```

Table 7-53

Behavioral Verilog for a positive-edge-triggered D flip-flop.

65

66

Positive-Edge-Triggered D Flip-Flop with Preset and Clear Behavioral-Style Verilog Module

```

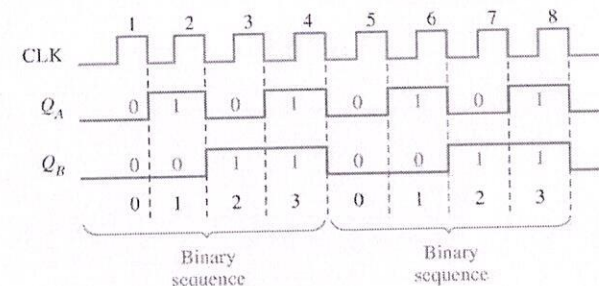
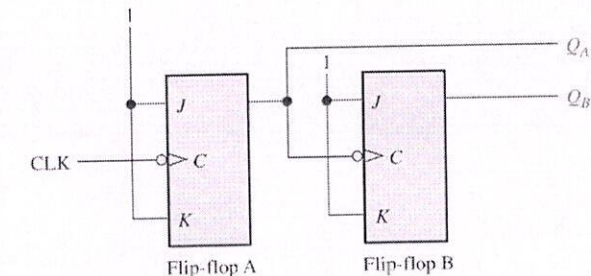
module Vrposdffpc(CLK, PR_L, CLR_L, D, Q);
  input CLK, PR_L, CLR_L, D;
  output Q;
  reg Q;

  always @ (posedge CLK or negedge PR_L or negedge CLR_L)
    if (PR_L==0) Q = 1;
    else if (CLR_L==0) Q <= 0;
    else Q <= D;
endmodule

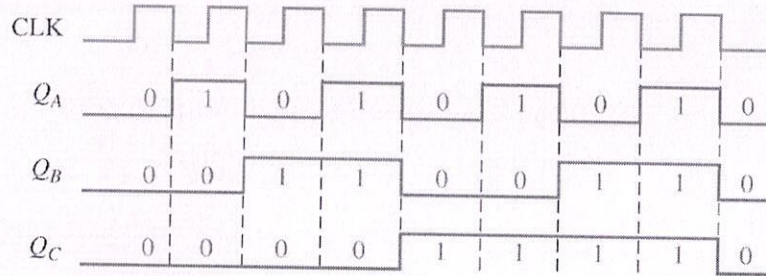
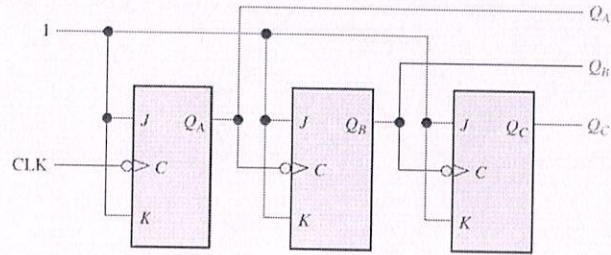
```

67

Counting



Counting



Verilog code for D -latch

Table 8-9 Verilog behavioral module for a D latch.

```

module VrDlatch( C, D, Q, QN );
input C, D;
output Q, QN;
reg Q, QN;

always @ (C or D or Q) begin
    if (C==1) Q <= D; else Q <= Q;
    QN <= !Q;
end
endmodule
    
```

Positive-Edge-Triggered D Flip-Flop Behavioral-Style Verilog Module

```

module VrposDff(CLK, D, Q);
input CLK, D;
output Q;
reg Q;

always @ (posedge CLK)
    Q <= D;
endmodule
    
```

Table 7-53

Behavioral Verilog for a positive-edge-triggered D flip-flop.

Positive-Edge-Triggered D Flip-Flop with Preset and Clear Behavioral-Style Verilog Module

```

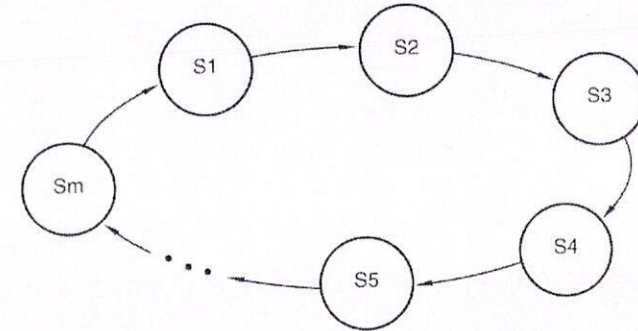
module Vrposdffpc(CLK, PR_L, CLR_L, D, Q);
input CLK, PR_L, CLR_L, D;
output Q;
reg Q;

always @ (posedge CLK or negedge PR_L or negedge CLR_L)
    if (PR_L==0) Q = 1;
    else if (CLR_L==0) Q <= 0;
    else Q <= D;
endmodule
    
```

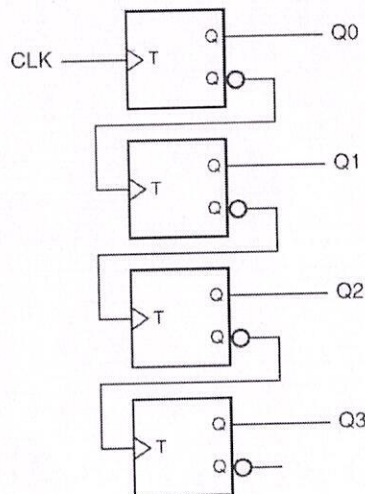
Counter

- The name **counter** is generally used for any clocked sequential circuit whose state diagram contains a single cycle, as shown in the next slide (Figure 8-23).
- The **modulus** of a counter is the number of states in the cycle.
- A counter with m states is called a **modulo- m counter**, or **divide-by- m counter**.
- A counter with a non-power-of-2 modulus has extra states that are not used in normal operation.

General Structure of a Counter State Diagram – A Single Cycle



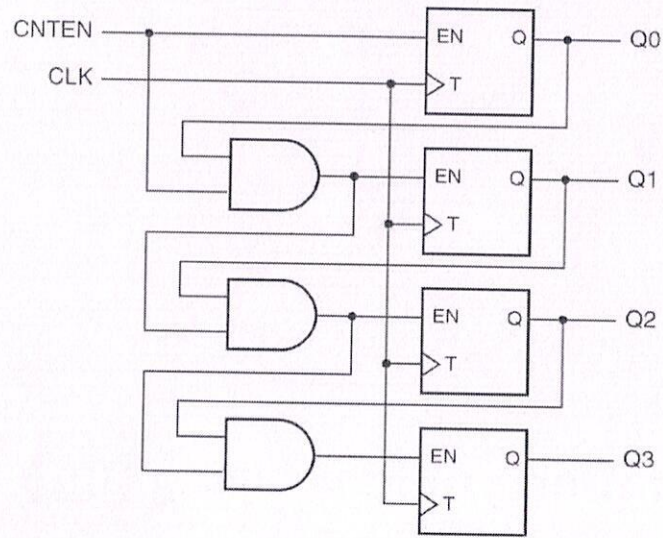
A 4-Bit Binary Ripple Counter



Ripple Counter

- A T flip-flop changes state (toggles) on every rising edge of its clock input.
- Thus, each bit of the counter toggles if and only if the immediately preceding bit changes from 1 to 0, it generates a carry to the next most significant bit.
- Although a ripple counter requires fewer components than any other type of binary counter, it does so at a price – it is slower than any other type of binary counter.
- In the worst case, when the most significant bit must change, the output is not valid until time $n \cdot t_{TQ}$ after the rising edge of CLK, where t_{TQ} is the propagation delay from input to output of a T flip-flop.

Synchronous Serial Counter



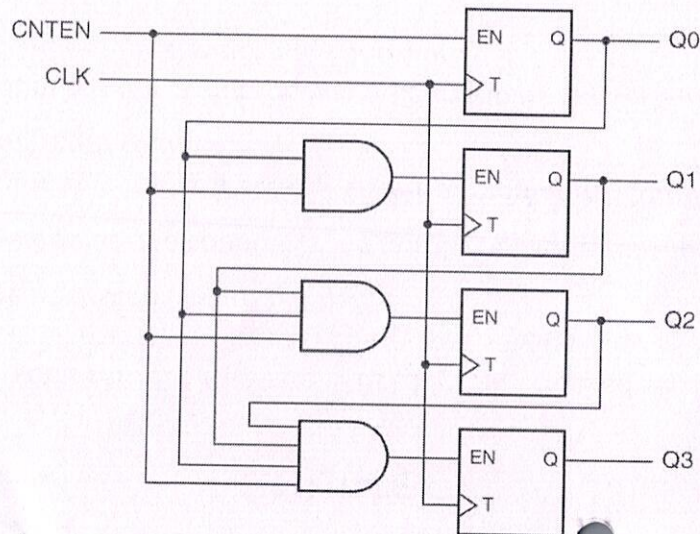
77

Synchronous Serial Counter

- CNTEN: Master count-enable signal
- Each T flip-flop toggles if and only if CNTEN is asserted and all of the lower-order counter bits are 1.

78

Synchronous Parallel Counter



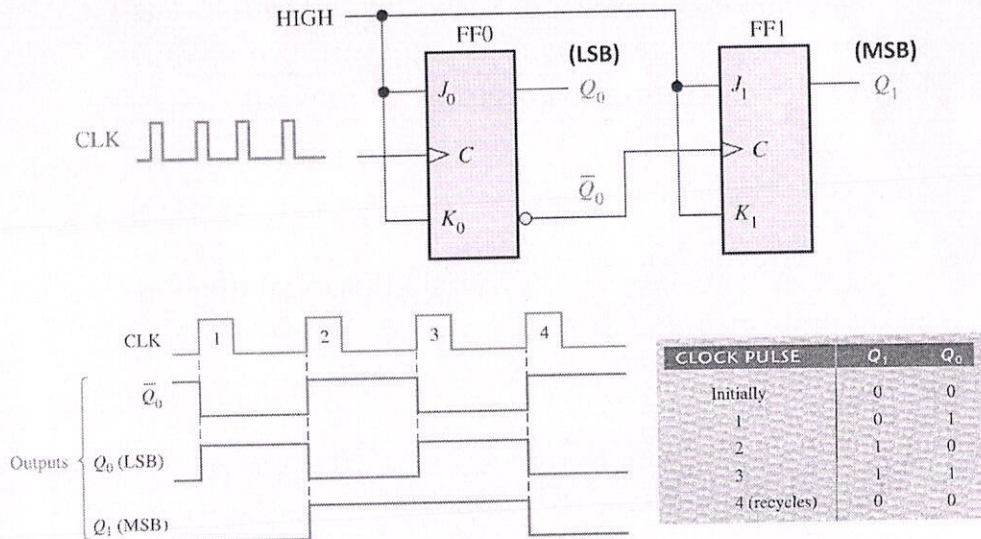
79

Synchronous Parallel Counter

- Synchronous parallel counter is the fastest binary counter structure.

80

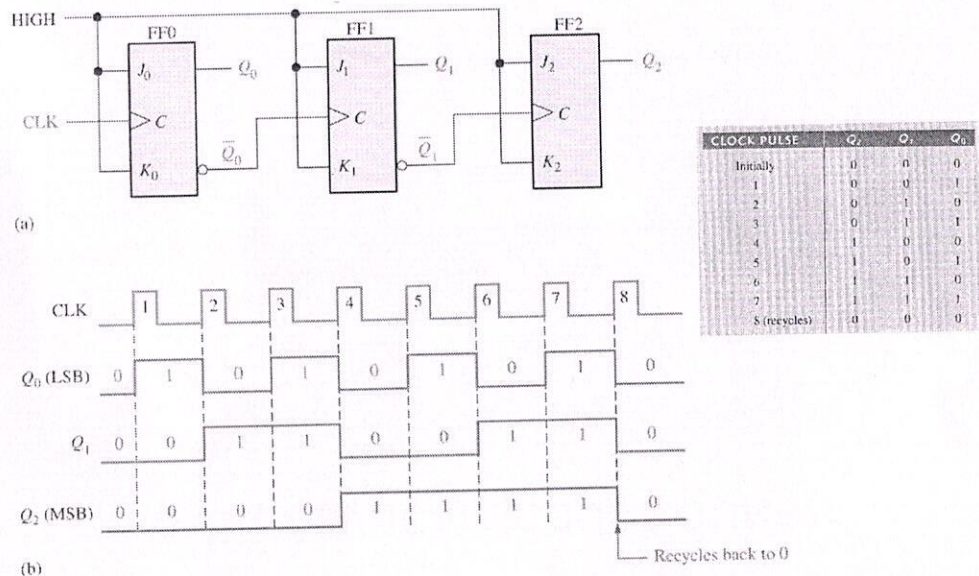
2-Bit Asynchronous Counter



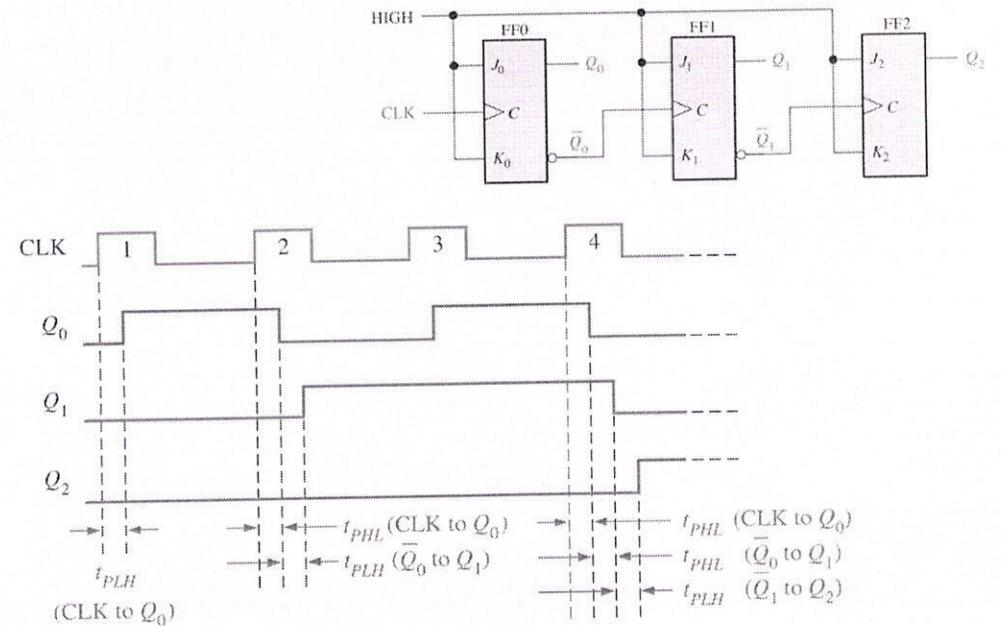
Asynchronous Counters

- The clock input of an asynchronous counter is always connected only to the LSB flip-flop.
- Asynchronous counters are also known as *ripple counters*.

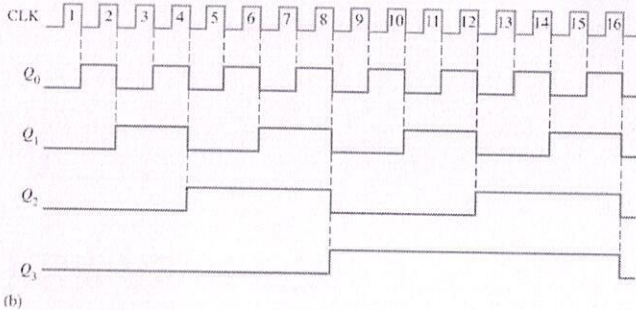
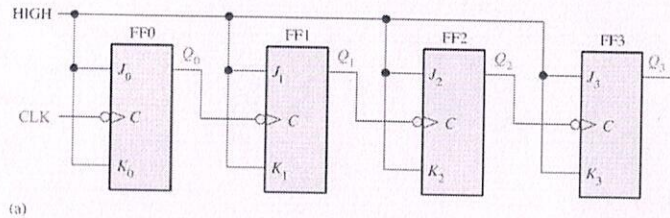
3-Bit Asynchronous Counter



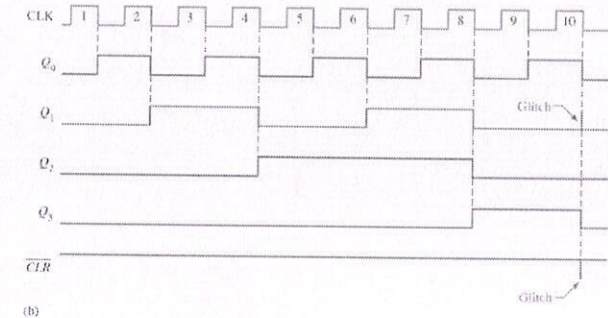
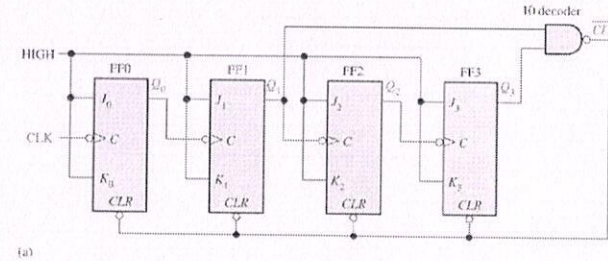
Propagation Delay in 3-Bit Counter



4-Bit Asynchronous Counter



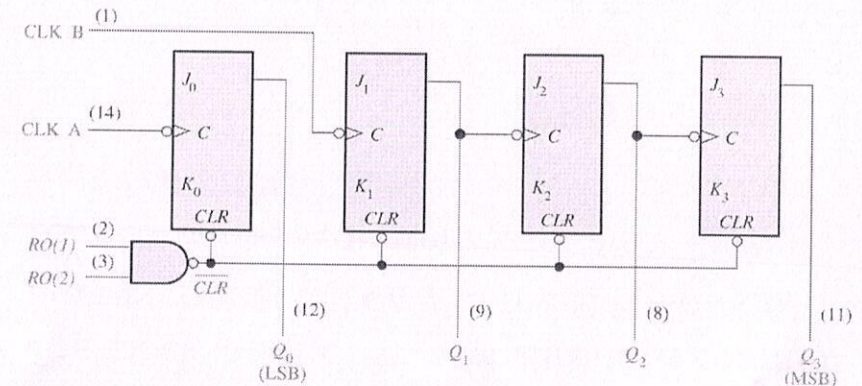
Asynchronous Decade Counter



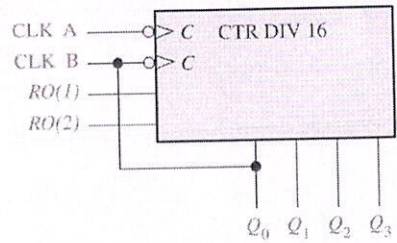
Terms

- Recycle
 - the transition of the counter from its final state back to its original state.
- Modulus
 - the number of states
 - the maximum possible number of states (maximum modulus) of a counter is 2^n , where n is the number of flip-flops in the counter.

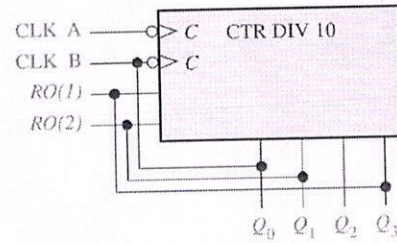
74LS93 (4-Bit Asyn. Counter)



74LS93 (4-Bit Asyn. Counter)

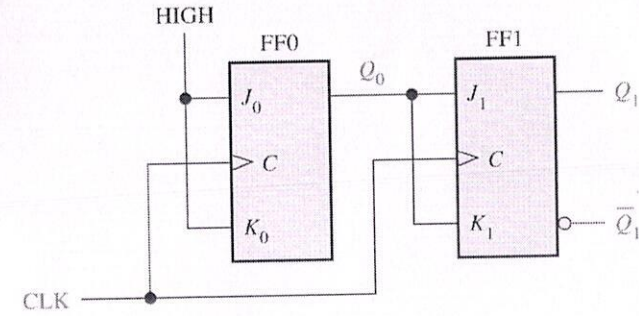


(a) 74LS93 connected as a modulus-16 counter

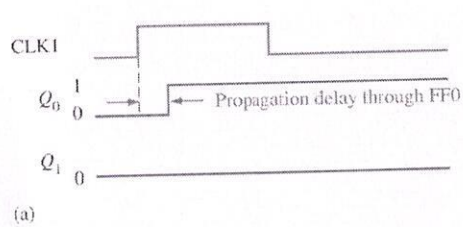


(b) 74LS93 connected as a decade counter

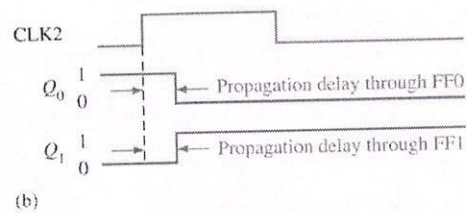
2-Bit Synchronous Counter



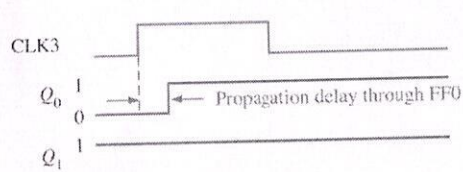
2-Bit Synchronous Counter



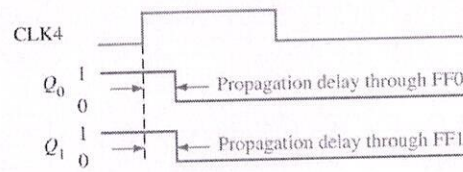
(a)



(b)

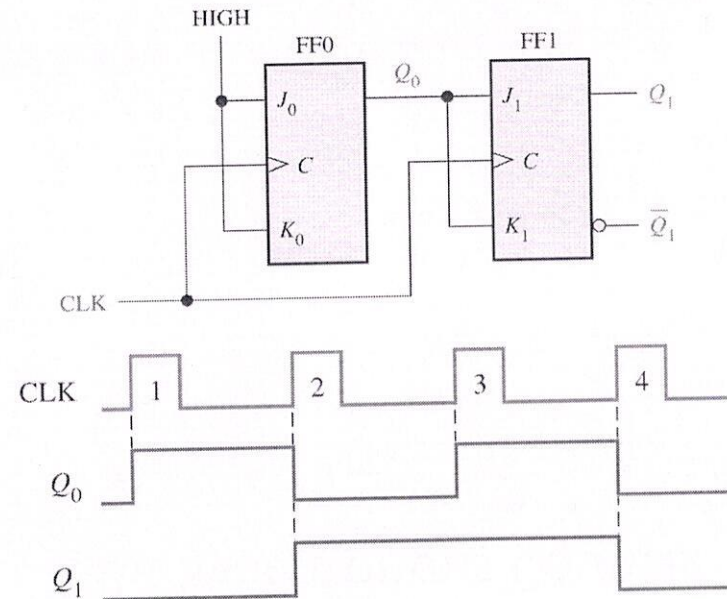


(c)

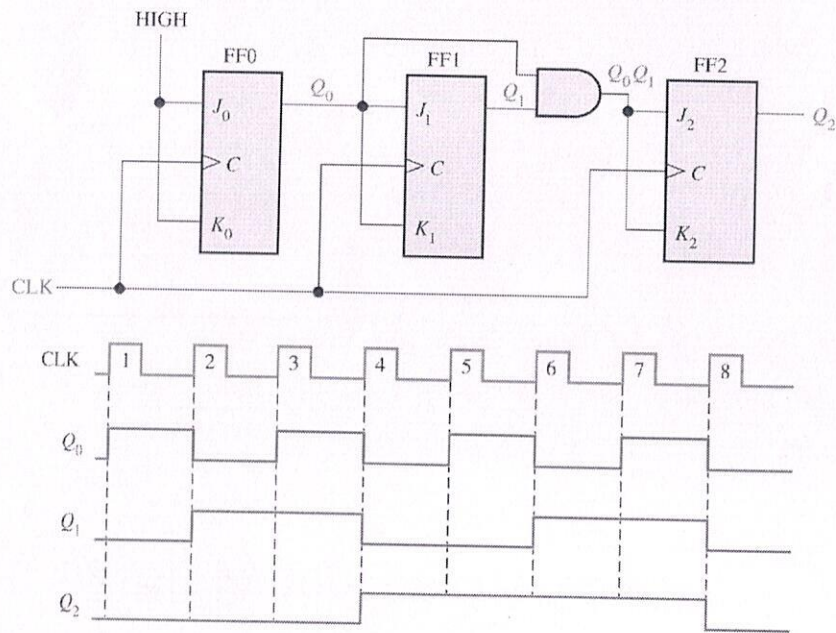


(d)

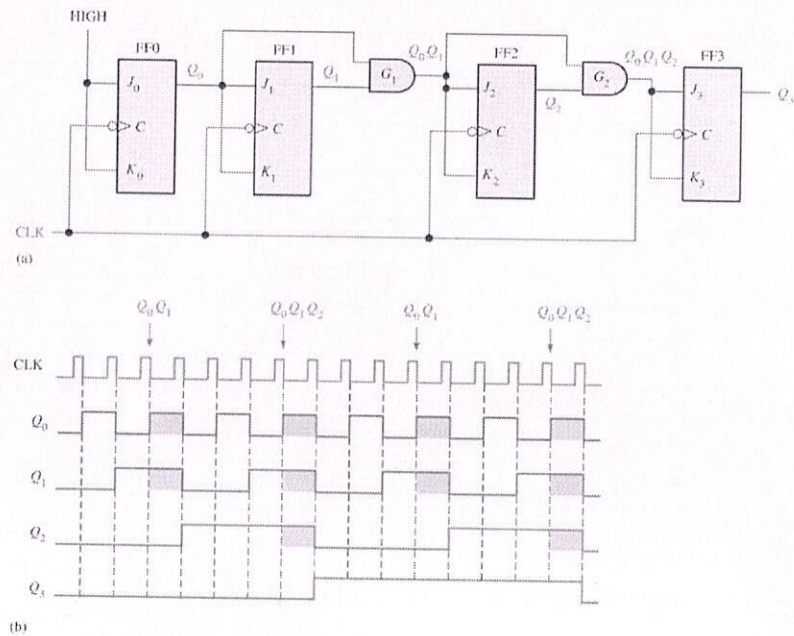
2-Bit Synchronous Counter



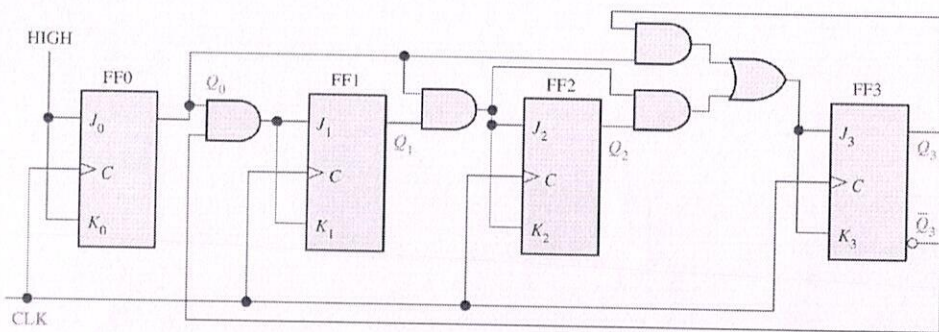
3-Bit Synchronous Counter



4-Bit Synchronous Counter

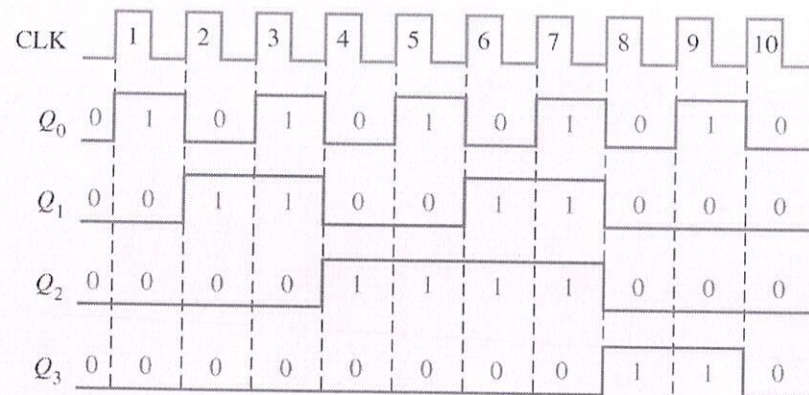


4-Bit Synchronous Decade Counter



- $J_0 = K_0 = 1$
- $J_1 = K_1 = \overline{Q_0} \overline{Q_3}$
- $J_2 = K_2 = Q_0 Q_1$
- $J_3 = K_3 = Q_0 Q_1 Q_2 + Q_0 Q_3$

4-Bit Synchronous Decade Counter



- $J_0 = K_0 = 1$
- $J_1 = K_1 = \overline{Q_0} \overline{Q_3}$
- $J_2 = K_2 = Q_0 Q_1$
- $J_3 = K_3 = Q_0 Q_1 Q_2 + Q_0 Q_3$

Verilog Module for 16 bit Register

```

module Vrreg16( CLK, CLKEN, OE_L, CLR_L, D, Q );
input CLK, CLKEN, OE_L, CLR_L;
input [1:16] D;
output [1:16] Q;
reg [1:16] IQ;

always @ (posedge CLK or negedge CLR_L)
if (CLR_L==0) IQ <= 16'b0;
else if (CLKEN==1) IQ <= D;
else IQ <= IQ;

assign Q = (OE_L==0) ? IQ : 16'bz;

endmodule

```

97

Verilog module for Extended function 8 bit Shift register

```

module Vrshftreg ( CLK, CLR, RIN, LIN, S, D, Q );
input CLK, CLR, RIN, LIN;
input [2:0] S;
input [7:0] D;
output [7:0] Q;
reg [7:0] Q;

always @ (posedge CLK or posedge CLR)
if (CLR == 1) Q <= 0;
else case (S)
0: Q <= Q; // Hold
1: Q <= D; // Load
2: Q <= {RIN, Q[7:1]}; // Shift right
3: Q <= {Q[6:0], LIN}; // Shift left

```

98

Verilog module for 8 bit Shift register

```

4: Q <= {Q[0], Q[7:1]}; // Shift circular right
5: Q <= {Q[6:0], Q[7]}; // Shift circular left
6: Q <= {Q[7], Q[7:1]}; // Shift arithmetic right
7: Q <= {Q[6:0], 1'b0}; // Shift arithmetic left
default Q <= 8'bx; // should not occur
endcase
endmodule

```

4-bit binary counter

4-bit binary counter

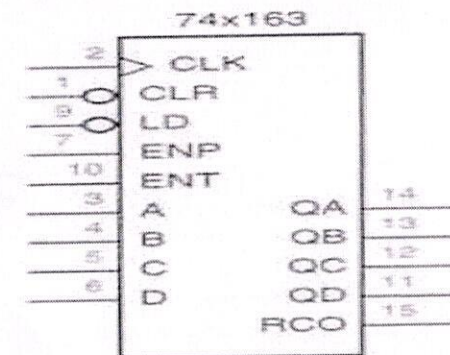


Figure 8-27
Traditional logic
symbol for
the 74x163.

99

100

Table 8-13 State table for a 74x163 4-bit binary counter.

Inputs				Current State				Next State			
CLR_L	LD_L	ENT	ENP	QD	QC	QB	QA	QD ⁿ	QC ⁿ	QB ⁿ	QA ⁿ
0	x	x	x	x	x	x	x	0	0	0	0
1	0	x	x	x	x	x	x	D	C	B	A
1	1	0	x	x	x	x	x	QD	QC	QB	QA
1	1	x	0	x	x	x	x	QD	QC	QB	QA
1	1	1	1	0	0	0	0	0	0	0	1
1	1	1	1	0	0	0	1	0	0	1	0
1	1	1	1	0	0	1	0	0	0	1	1
1	1	1	1	0	1	0	0	0	1	0	0
1	1	1	1	0	1	0	1	0	1	1	0
1	1	1	1	0	1	1	0	0	1	1	1
1	1	1	1	1	0	0	0	1	0	0	0
1	1	1	1	1	0	0	1	1	0	1	0
1	1	1	1	1	0	1	0	1	0	1	1
1	1	1	1	1	1	0	1	1	1	0	0
1	1	1	1	1	1	1	0	1	1	0	1
1	1	1	1	1	1	1	1	1	1	1	0
1	1	1	1	1	1	1	1	0	1	1	1
1	1	1	1	1	1	1	1	0	0	0	0

Verilog module for 4 bit binary counter

```

module Vr74x163 ( CLK, CLR_L, LD_L, ENP, ENT, D, Q, RCO );
input CLK, CLR_L, LD_L, ENP, ENT;
input [3:0] D;
output [3:0] Q;
output RCO;
reg [3:0] Q;
reg RCO;

always @ (posedge CLK) // Create the counter f-f behavior
if (!CLR_L)      Q <= 4'b0;
else if (!LD_L)  Q <= D;
else if (ENT && ENP)  Q <= Q + 1;
else            Q <= Q;

always @ (Q or ENT) // Create RCO combinational output
if (ENT && (Q == 4'd15)) RCO = 1;
else                    RCO = 0;
endmodule
    
```

Verilog code for 74x162 like 4 bit Decimal Counter

```

module Vr74x162 ( CLK, CLR_L, LD_L, ENP, ENT, D, Q, RCO );
input CLK, CLR_L, LD_L, ENP, ENT;
input [3:0] D;
output [3:0] Q;
output RCO;
reg [3:0] Q;
reg RCO;

always @ (posedge CLK) // Create the counter f-f behavior
if (!CLR_L)      Q <= 4'b0;
else if (!LD_L)  Q <= D;
else if (ENT && ENP && (Q == 4'd9)) Q <= 4'b0;
else if (ENT && ENP)  Q <= Q + 1;
else            Q <= Q;

always @ (Q or ENT) // Create RCO combinational output
if (ENT && (Q == 4'd9)) RCO = 1;
else                    RCO = 0;
endmodule
    
```