COL215L: Digital Logic & System Design

Lecture 2: CMOS Circuits

M. Balakrishnan
CSE@IITD

Vireshwar Kumar
CSE@IITD

August 11, 2021

Platforms

- Moodle
  - Useful Links, Quizzes, Tutorials, Slides, Notes
- Teams
  - Live Lectures
- Impartus
  - Recorded Lectures with Slides
- Gradescope (https://www.gradescope.com/)
  - Minor and Major Exams
- Piazza (http://piazza.com/iit_delhi/fall2021/col215) Access Code: col215
  - Discussion

Books

- Book-1
  - Stephen Brown and Zvonko Vranesic,“Fundamental of Digital Logic with VHDL Design,” 3rd edition, McGraw Hill, 2013.

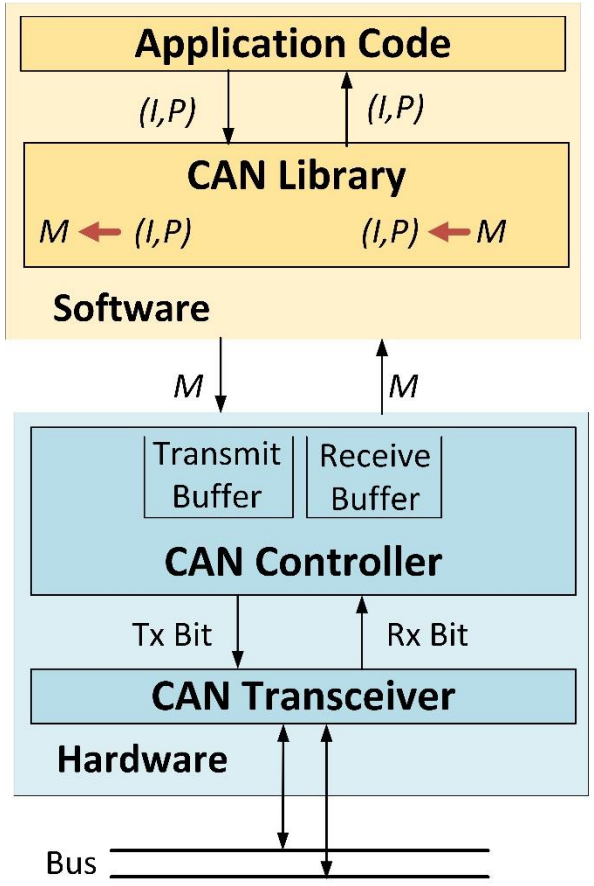
- Book-2
  - M. Morris Mano, “Digital Logic and Computer Design,” Pearson, 2017.

- Book-3
  - M. Rafiquzzaman “Fundamentals Of Digital Logic And Microcomputer Design,” 5th edition, Wiley, 2005.

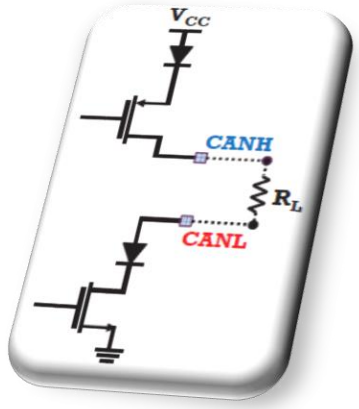
Agenda

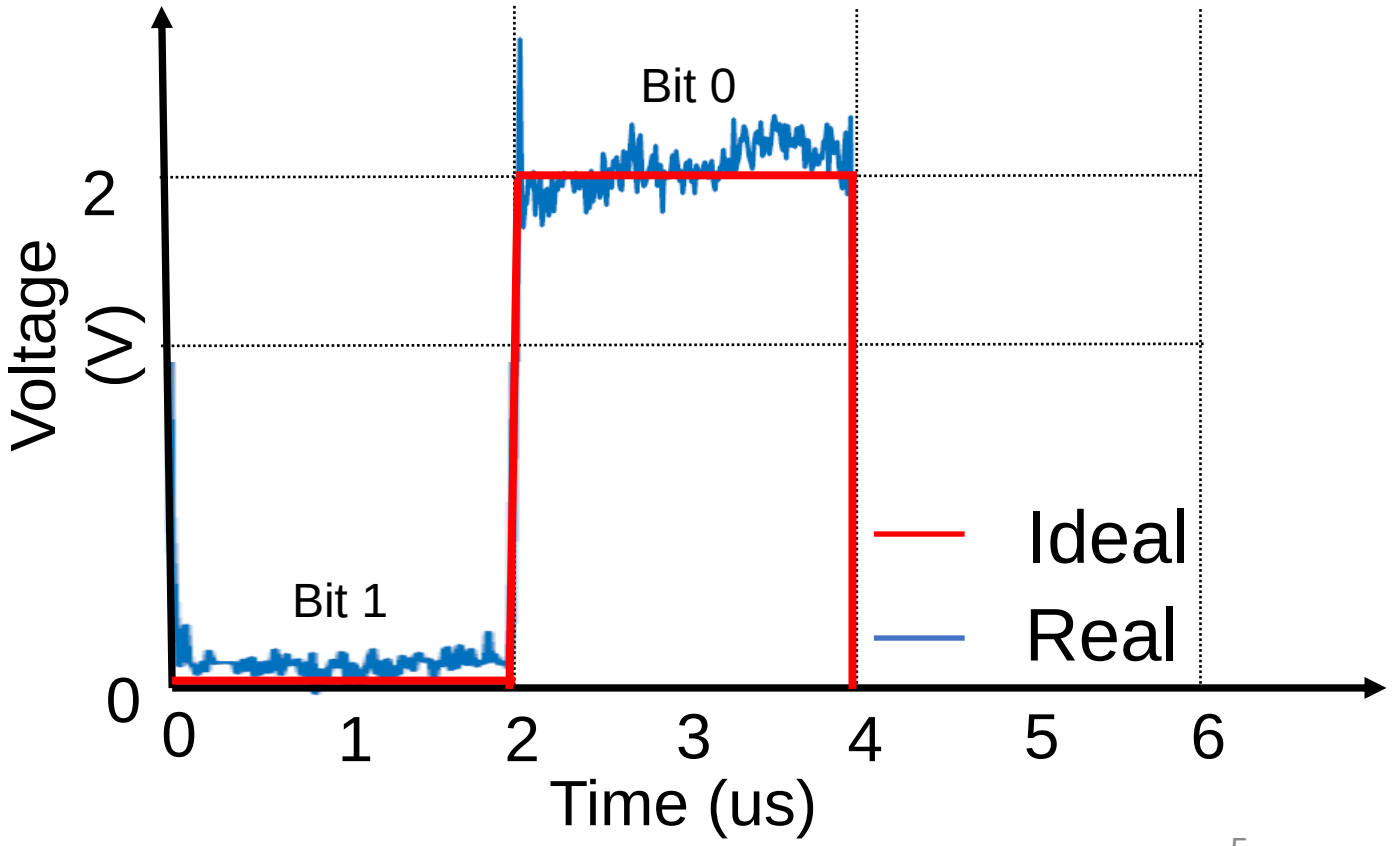
- Transistors NMOS
  - Chapter 3, Book-1 [Brown & Vranesic, 2013]

Automotive Controller Area Network (CAN)

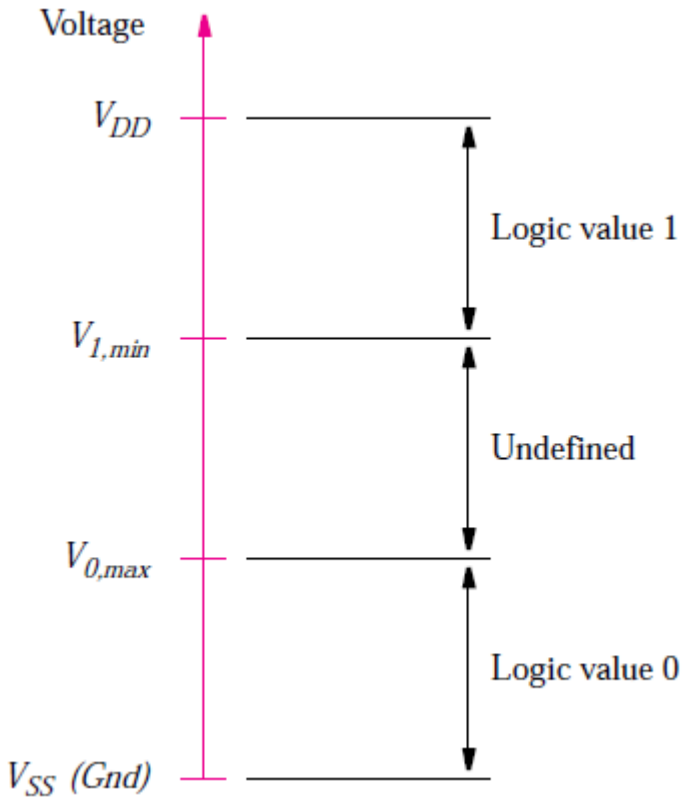


I – Message Identifier
P – Message Payload
M – CAN Message





Digital Logic and Voltage Values



Mapping of digital logic and voltage values [Brown & Vranesic, 2013].

COL215L: Digital Logic & System Design

Lecture 3: CMOS Circuits (Cont.)

M. Balakrishnan
CSE@IITD

Vireshwar Kumar
CSE@IITD

August 13, 2021

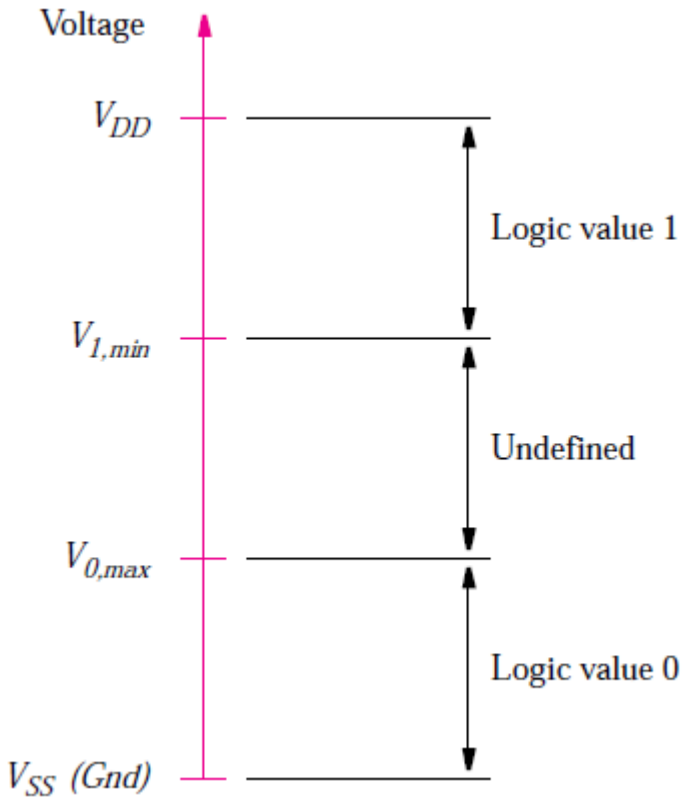
Digital Logic

- Any system
  - Storage
  - Computation
  - Communication

- Implementation of any system
  - Digital logic (0-1)

- Realization of digital logic
  - Transistors for controlling voltages that we interpret as 0 or 1

Digital Logic and Voltage Values



Mapping of digital logic and voltage values [Brown & Vranesic].

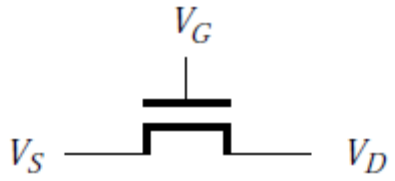
Realizing the Mapping

- Metal Oxide Semiconductor Field-effect Transistor (MOSFET)
  - n-channel MOSFET (NMOS)
  - p-channel MOSFET (PMOS)

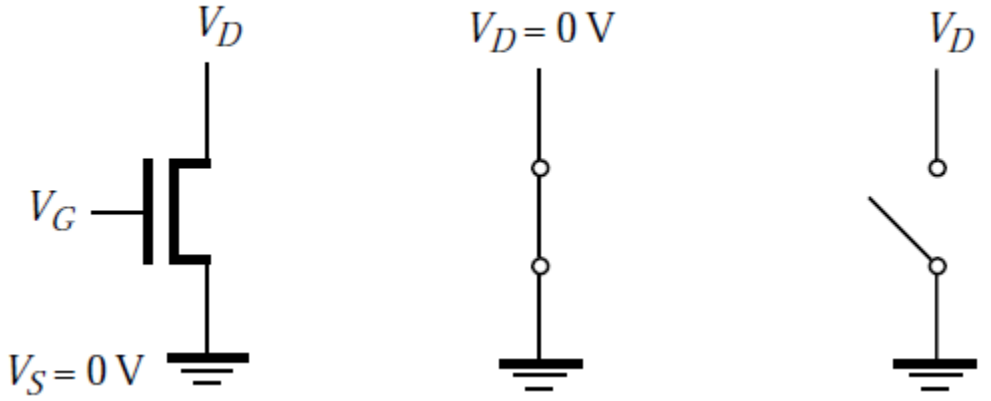
- Complementary MOS (CMOS)
  - Circuits using both NMOS and PMOS

NMOS

- Drain
- Source
- Gate

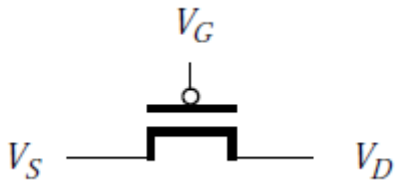


NMOS [Brown & Vranesic].

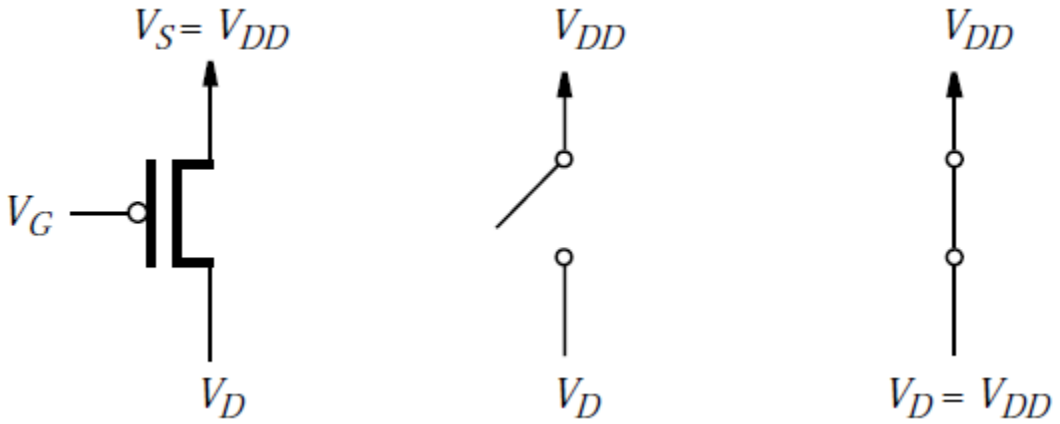


NMOS logic circuit [Brown & Vranesic].

PMOS

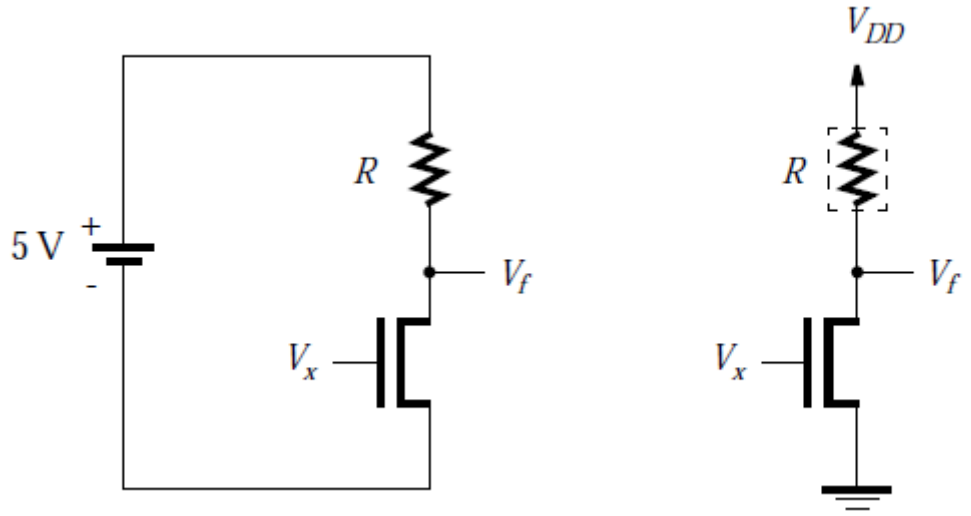


PMOS [Brown & Vranesic].

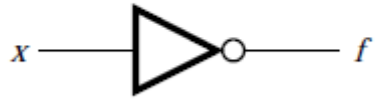


PMOS logic circuit [Brown & Vranesic].

Realizing NOT using NMOS

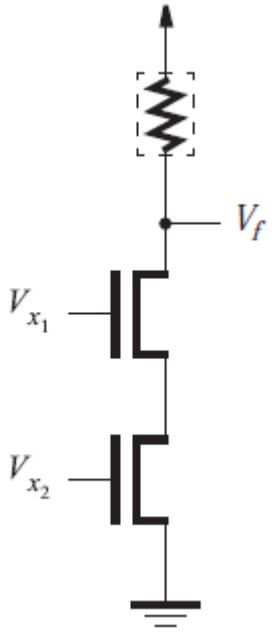


NOT Circuit [Brown & Vranesic].



Representation.

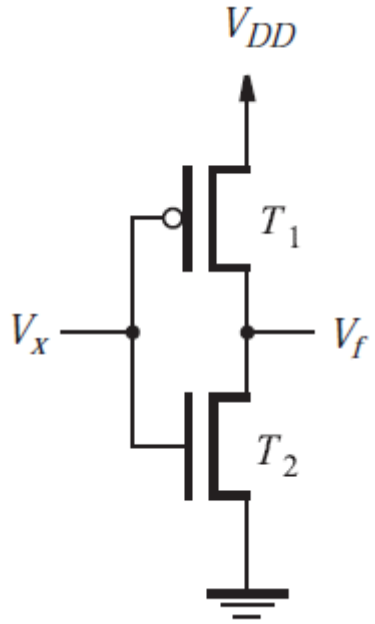
Realizing NAND using NMOS



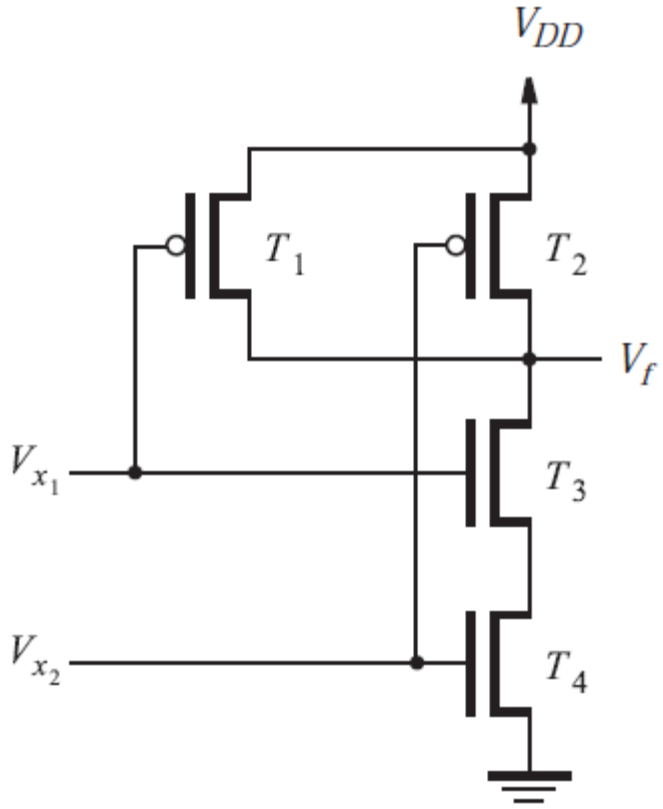
NAND Circuit [Brown & Vranesic].

Representation

Realizing logic gates using CMOS

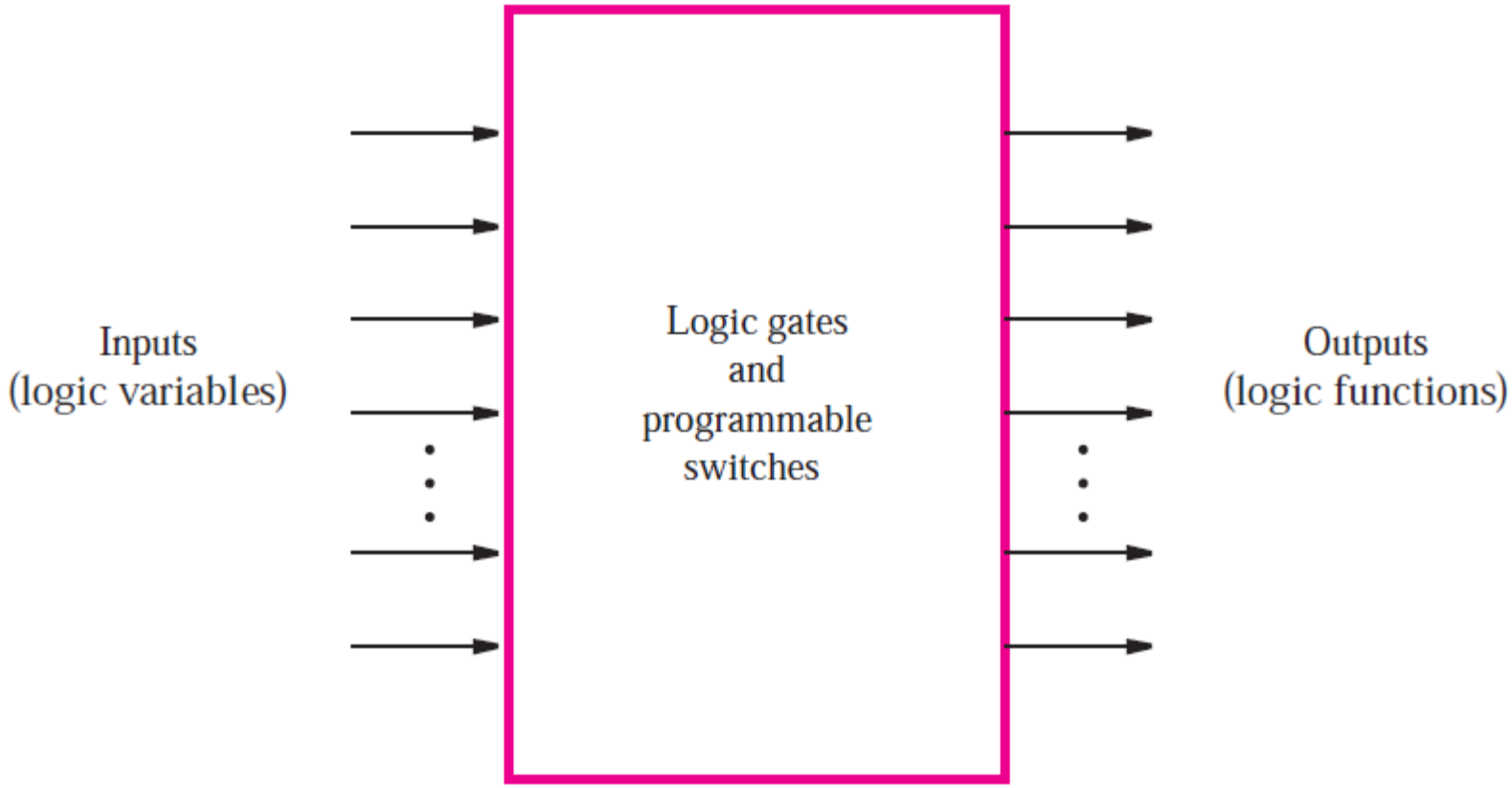


NOT circuit [Brown & Vranesic].



NAND circuit [Brown & Vranesic].

Programmable Logic Array (PLA)



PLA [Brown & Vranesic].

COL215L: Digital Logic & System Design

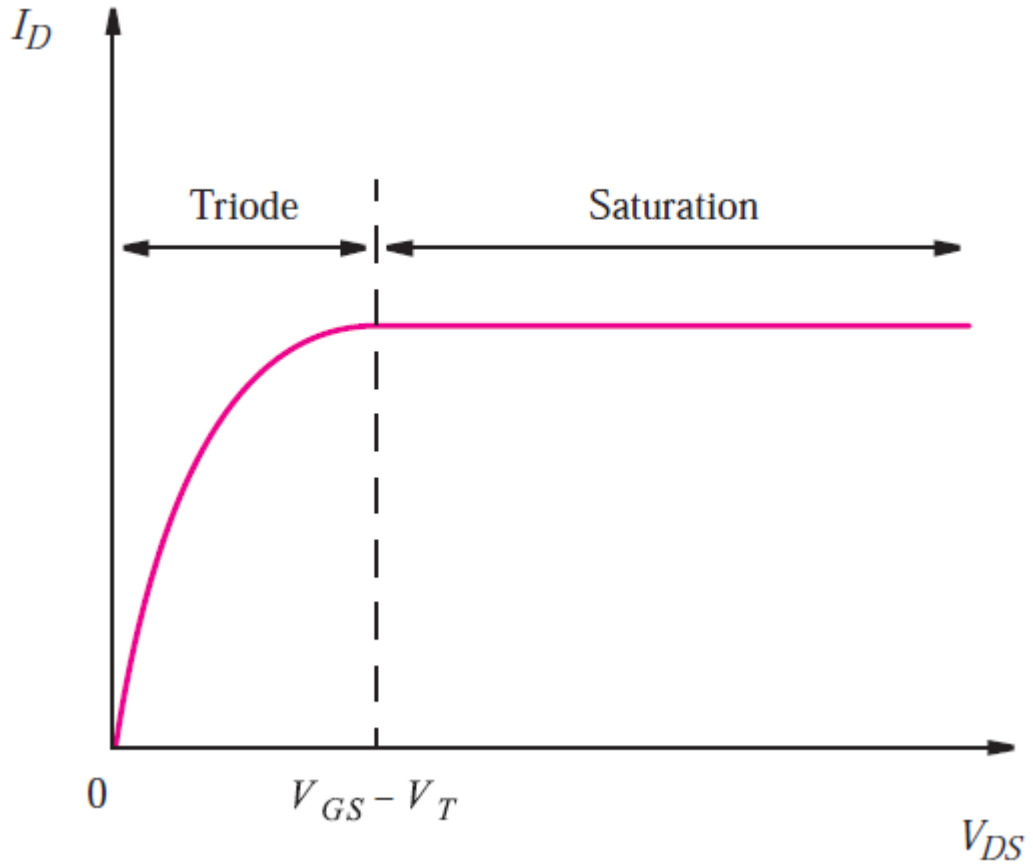
Lecture 4: CMOS Circuits (Cont.)

M. Balakrishnan
CSE@IITD

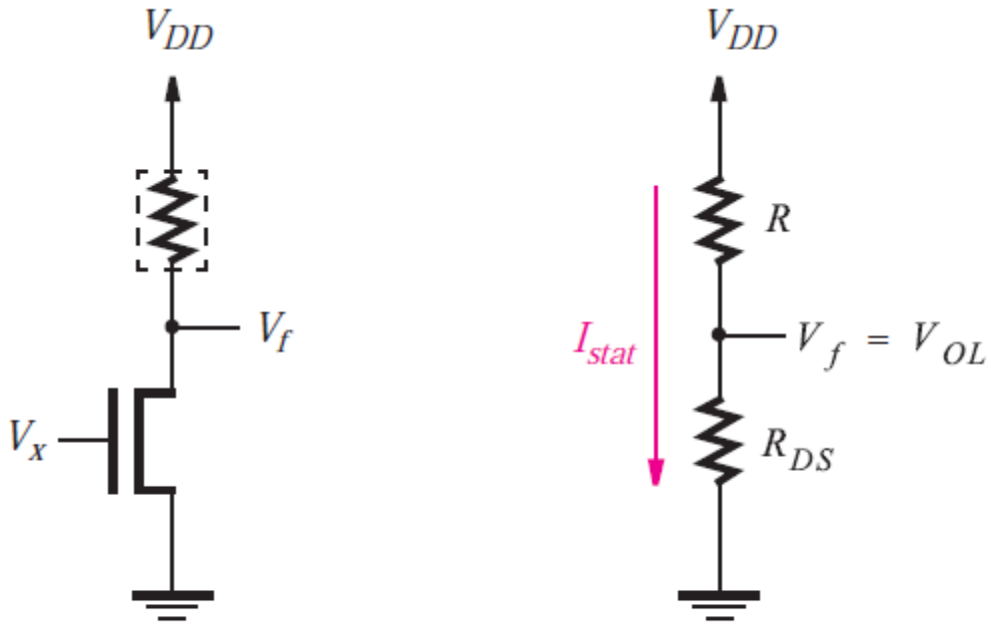
August 17, 2021

Vireshwar Kumar
CSE@IITD

Current-Voltage Relationship in NMOS



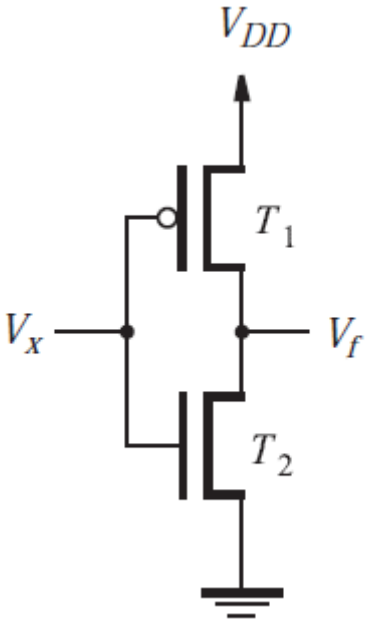
On-resistance in NMOS

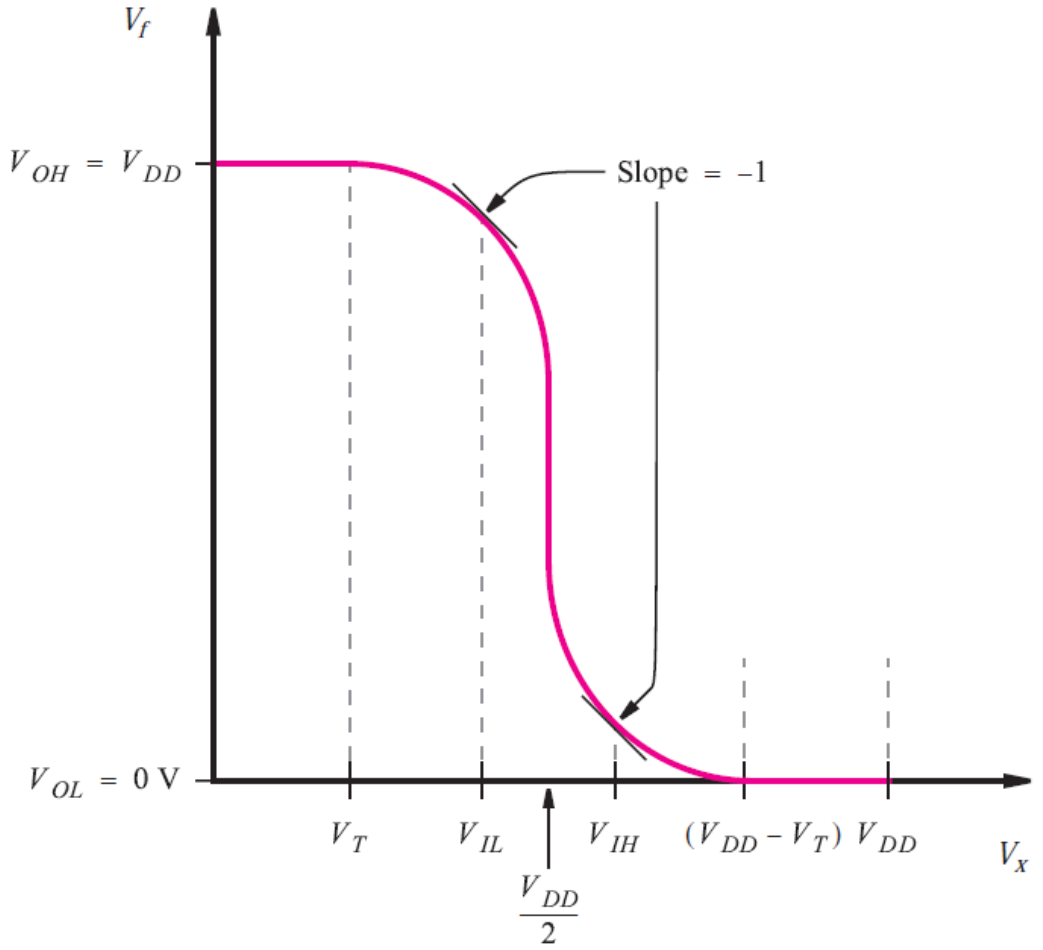


(a) NMOS NOT gate

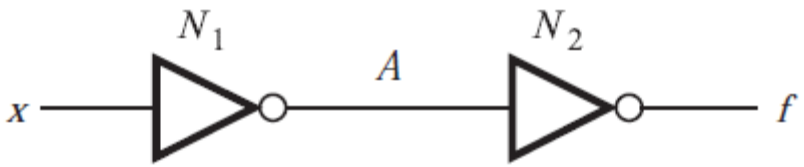
(b) $V_x = 5$ V

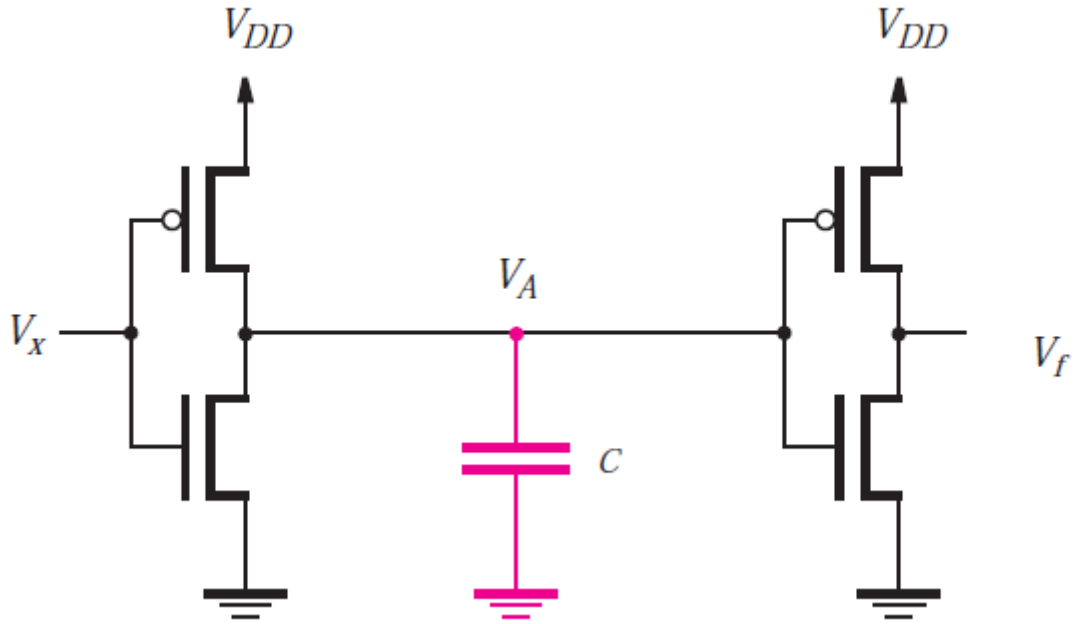
Voltage Characteristics in CMOS Inverter



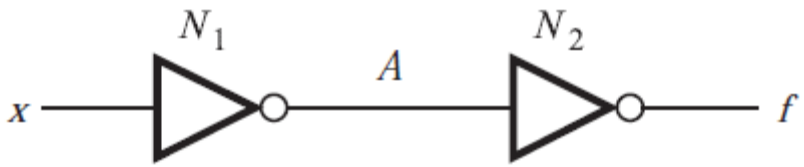


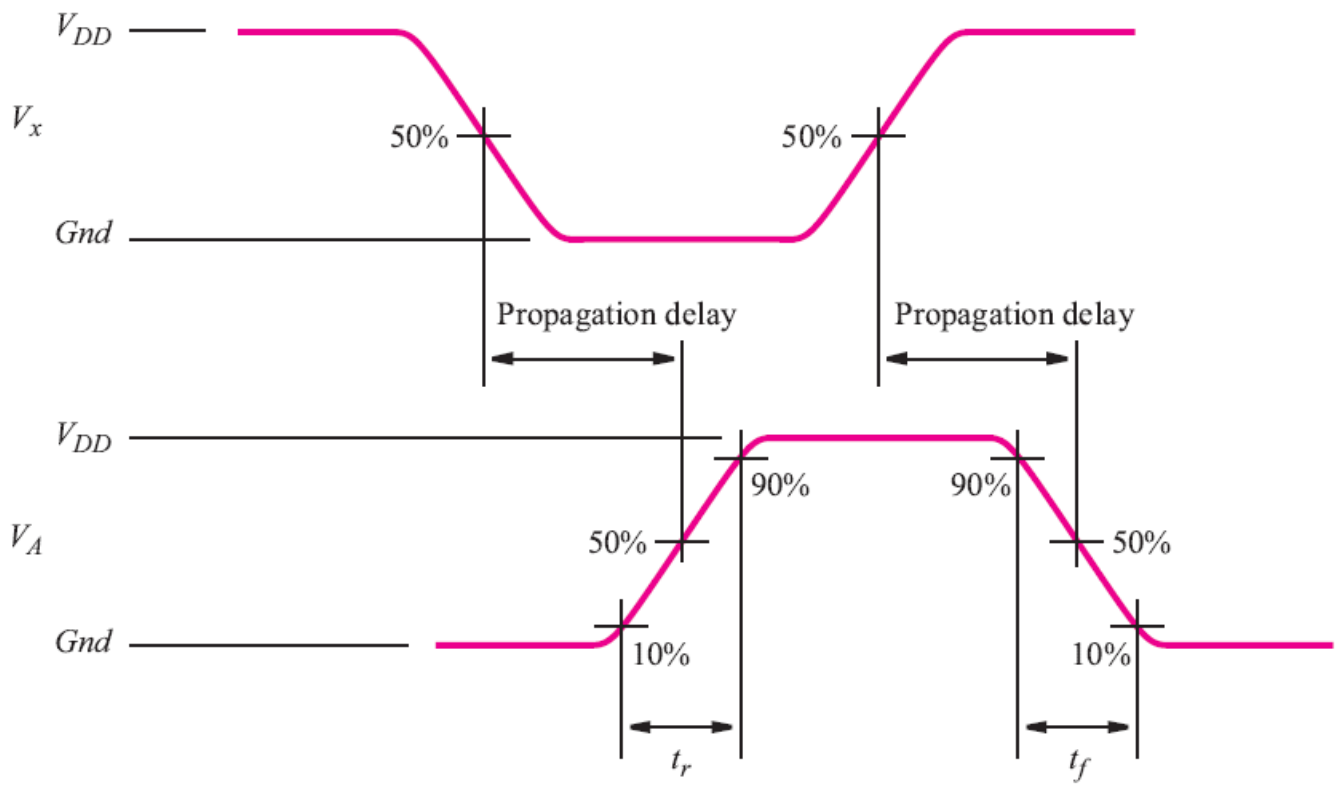
Dynamic Behavior





Propagation Delay





COL215L: Digital Logic & System Design

Lecture 5: CMOS Circuits (Cont.)

M. Balakrishnan
CSE@IITD

Vireshwar Kumar
CSE@IITD

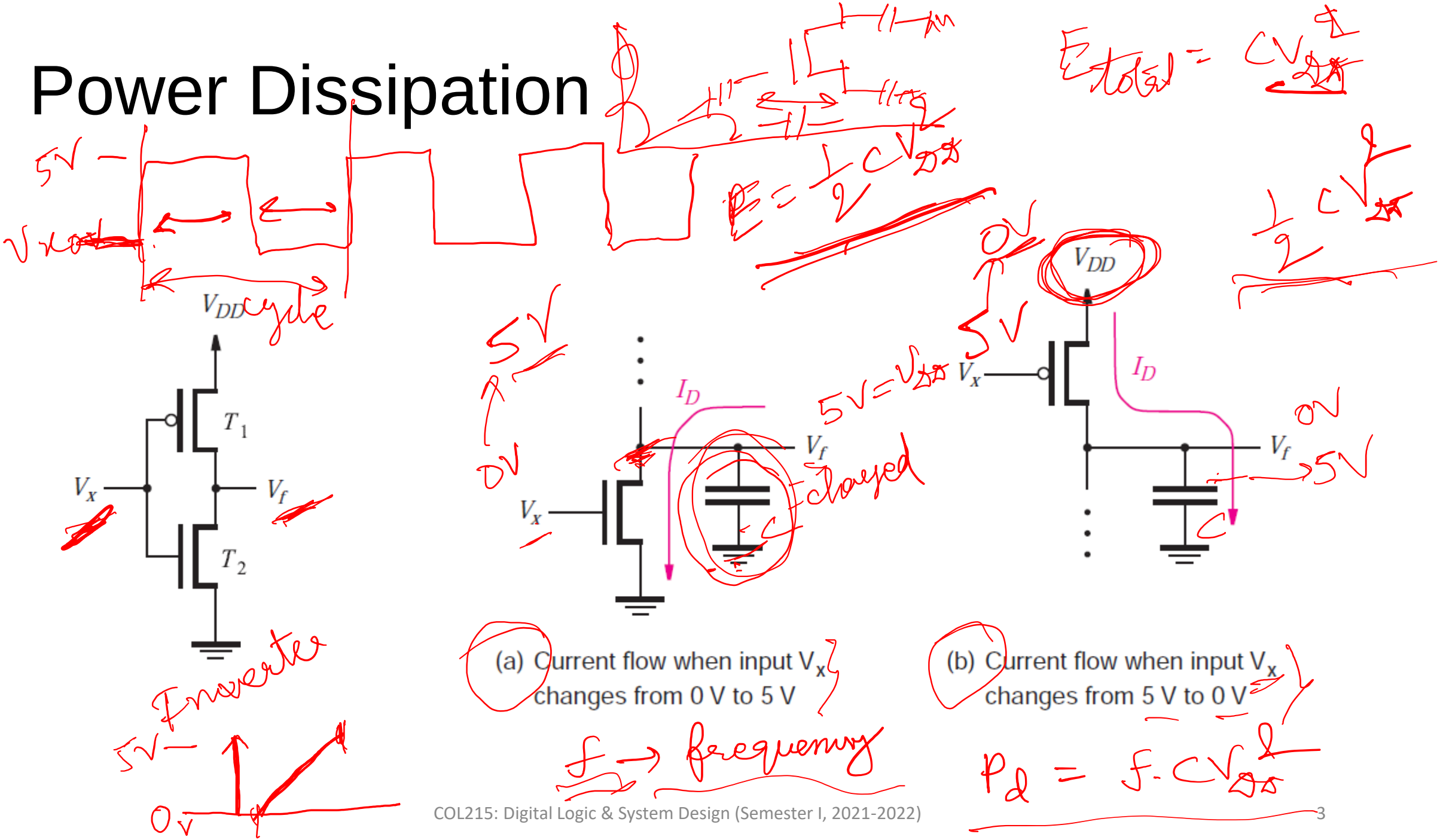
August 18, 2021

Agenda

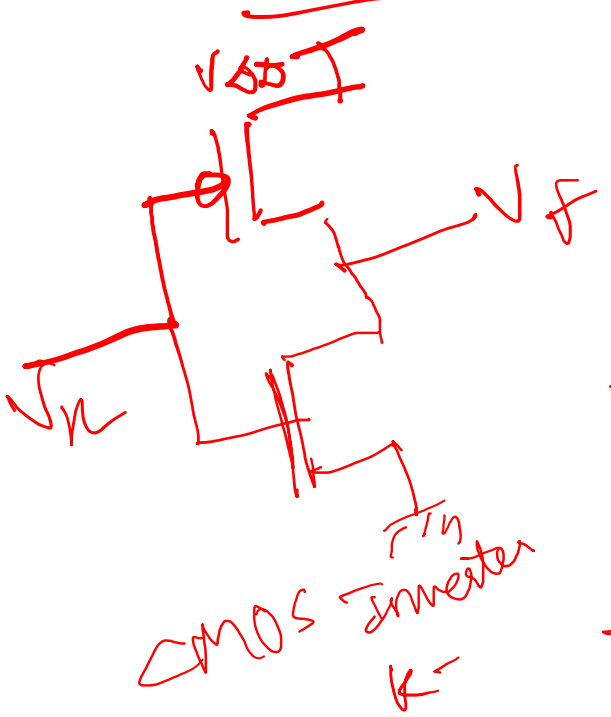
- Previously
  - Dynamic behavior
  - Propagation delay

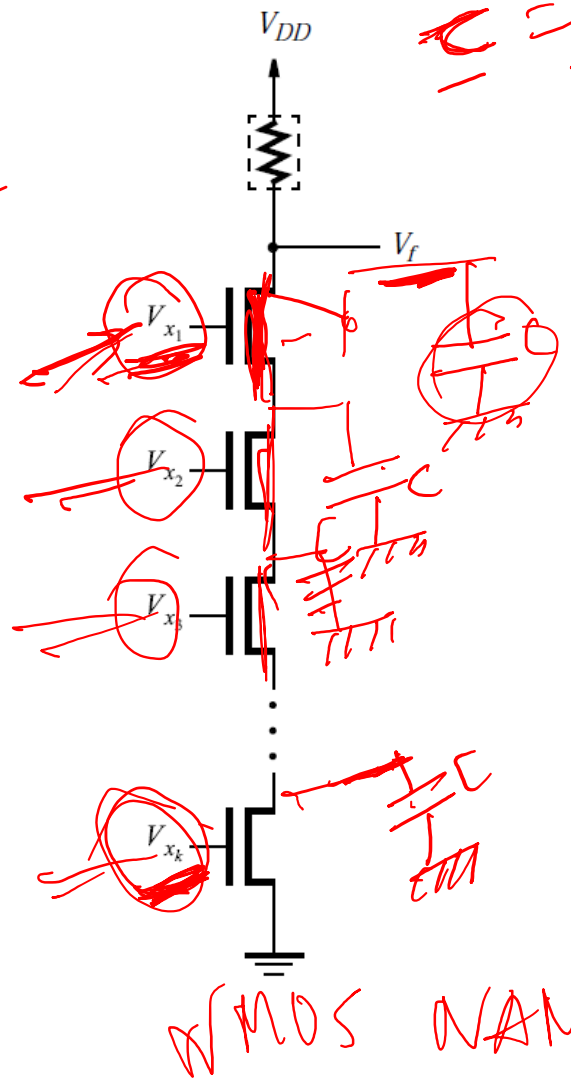
- Today
  - Power dissipation
  - Fan-in and fan-out limitations

Power Dissipation

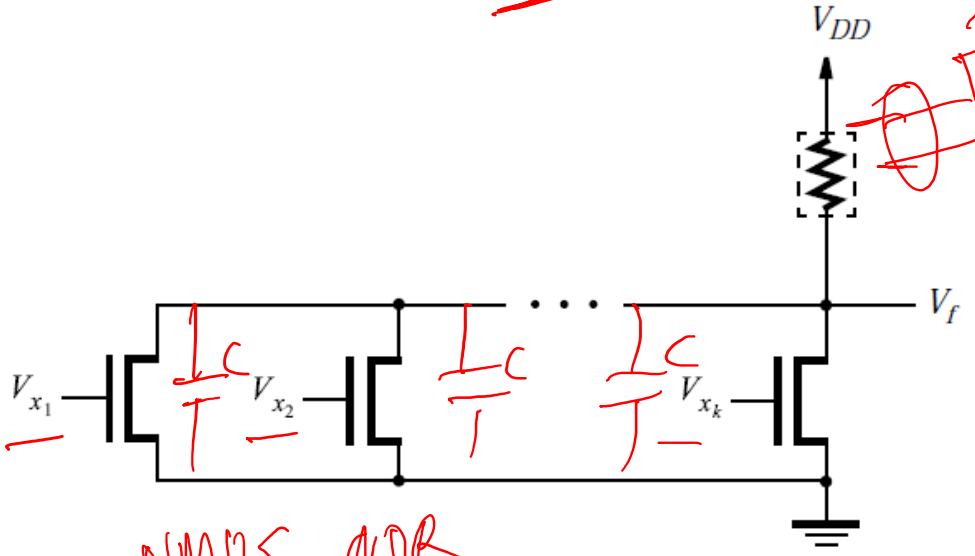


Impact of High Fan-in



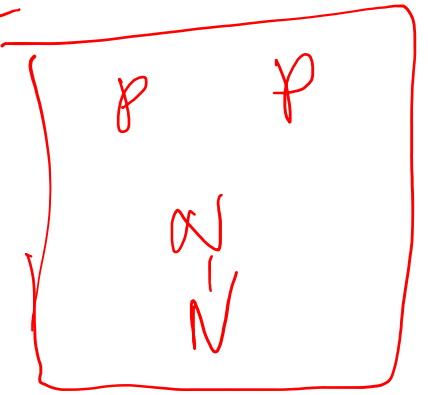


NMOS NAND

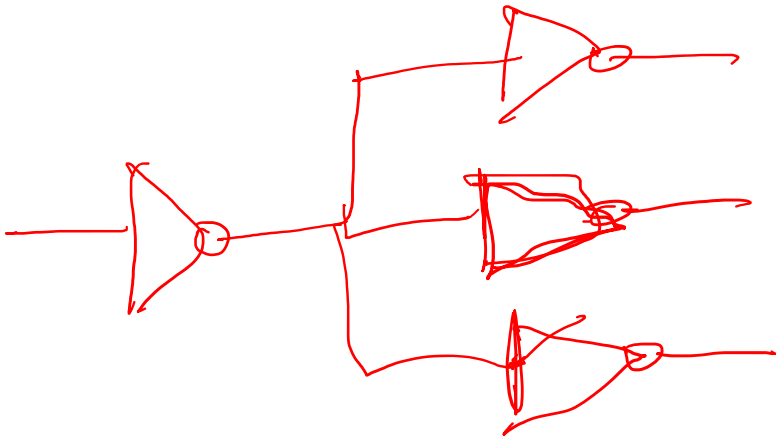


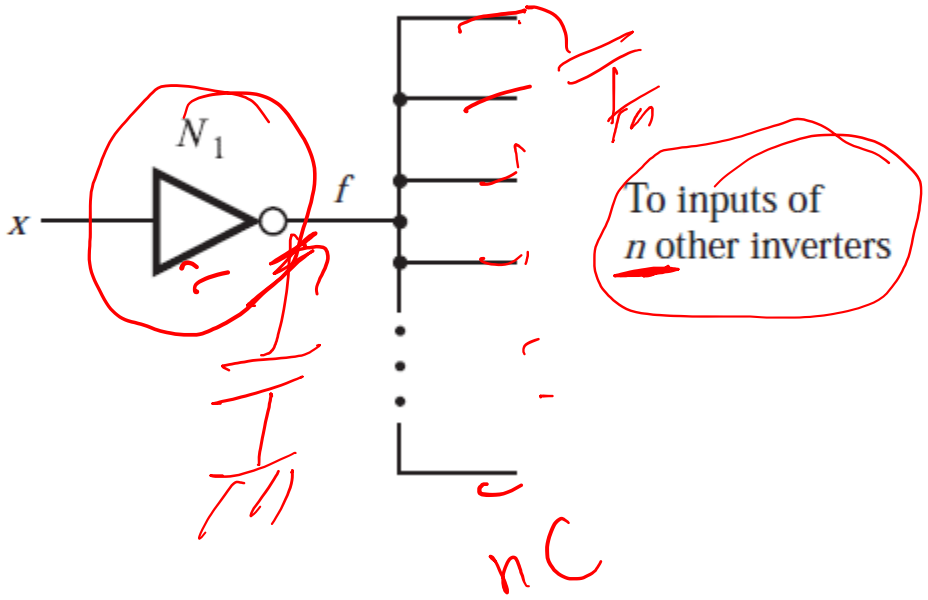
NMOS NOR

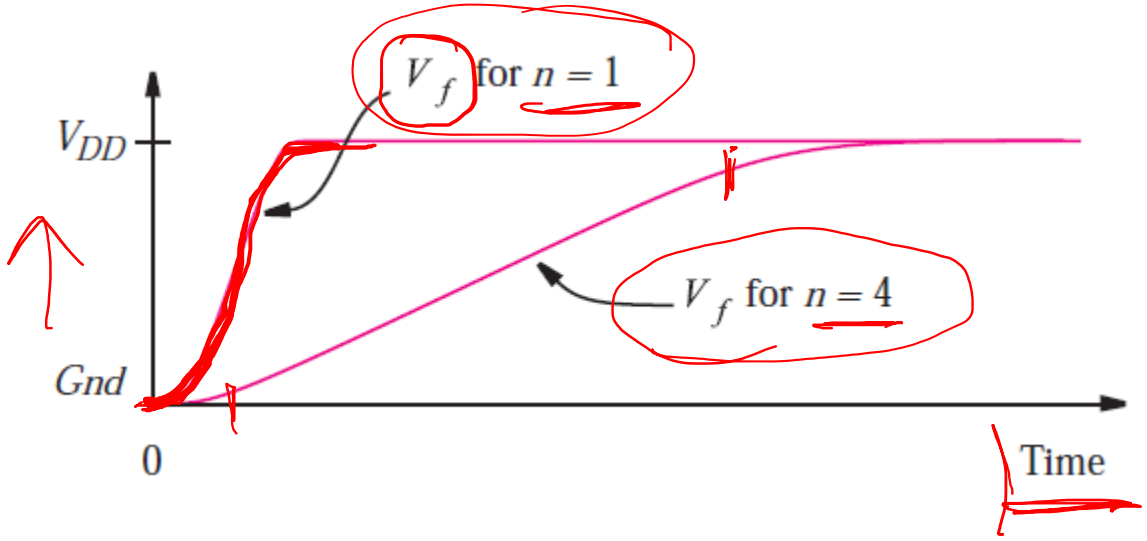
CMOS



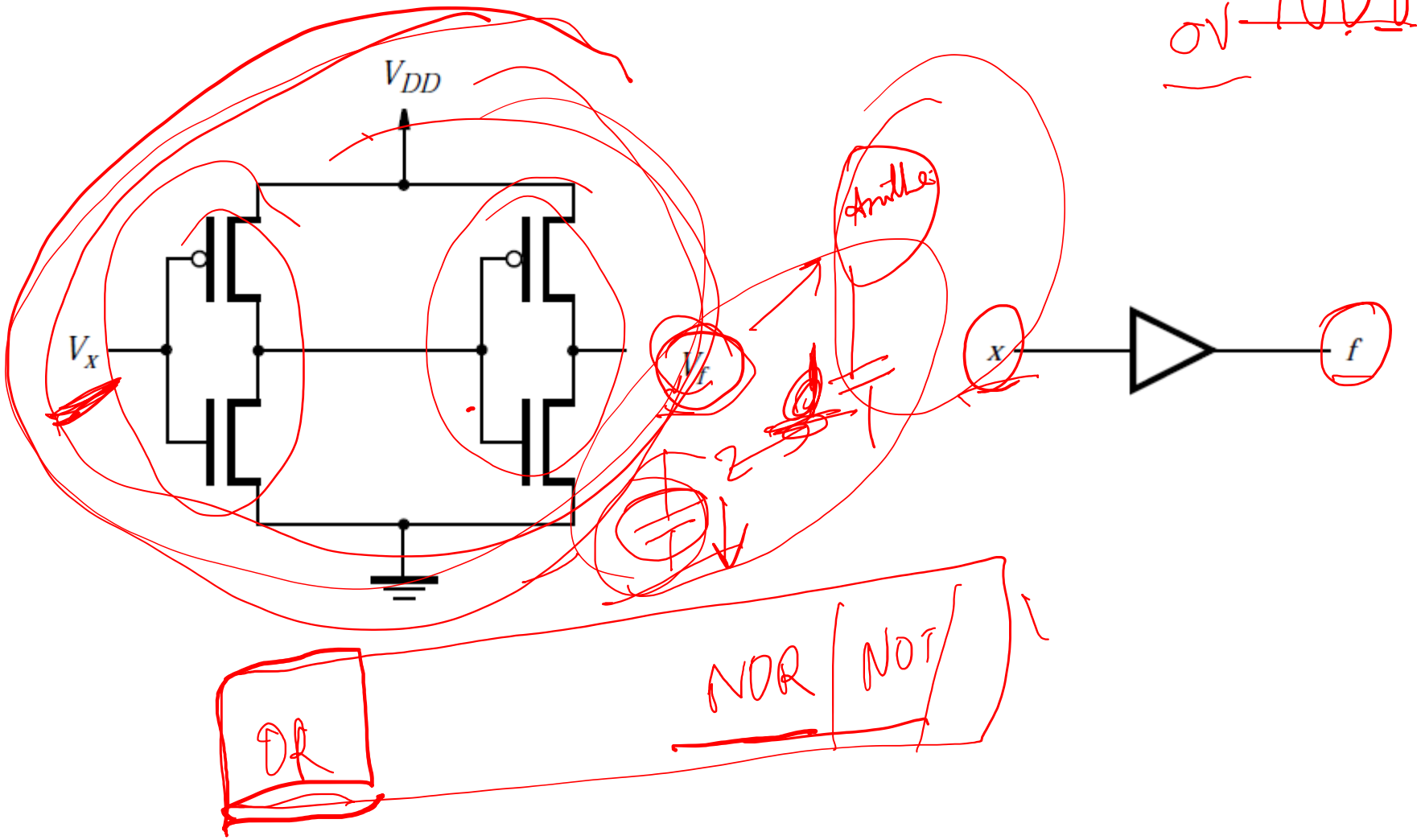
Impact of High Fan-out



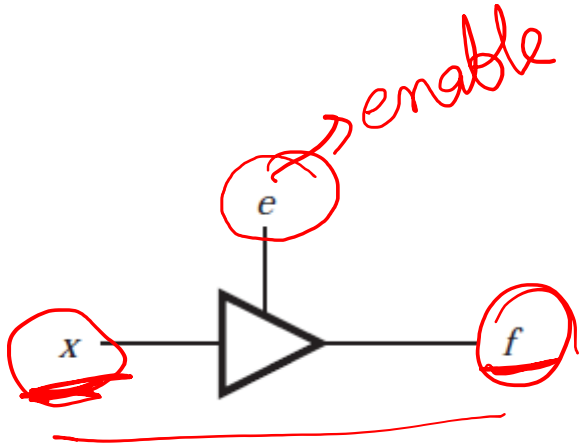




Buffers

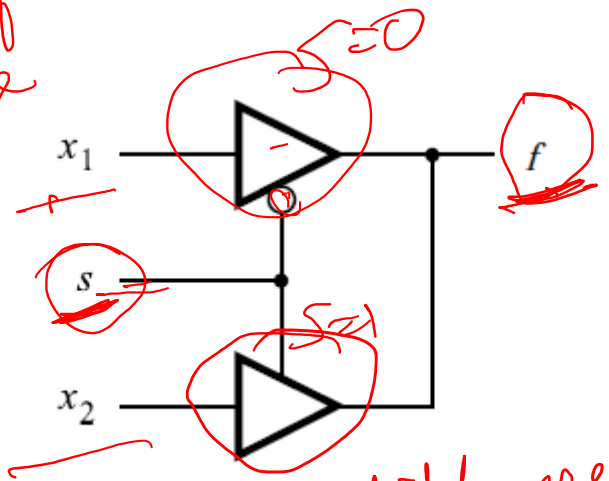


Tri-state Buffer



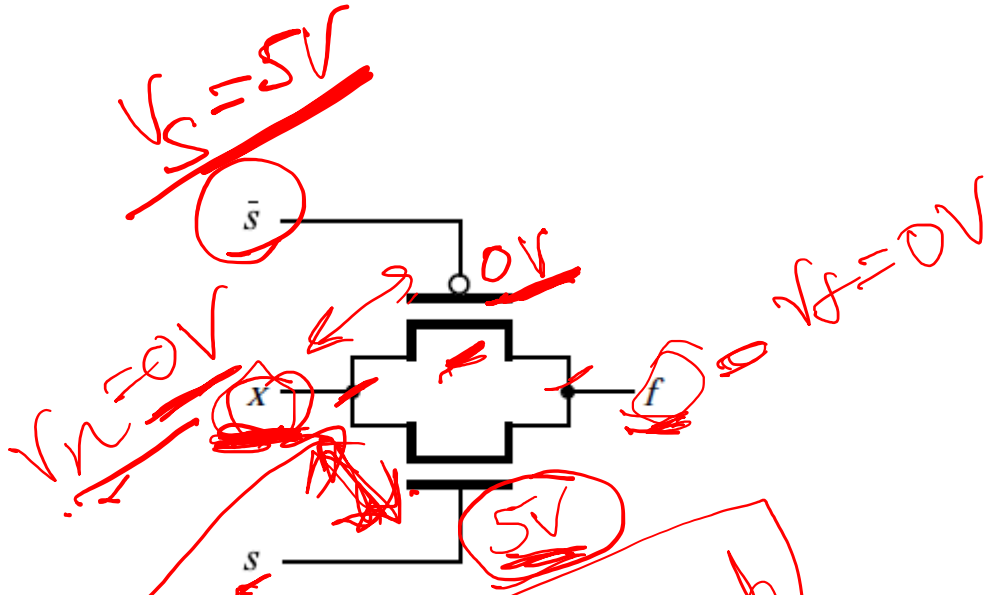
e	x	f
0	0	Z
0	1	Z
1	0	0
1	1	1

floating state

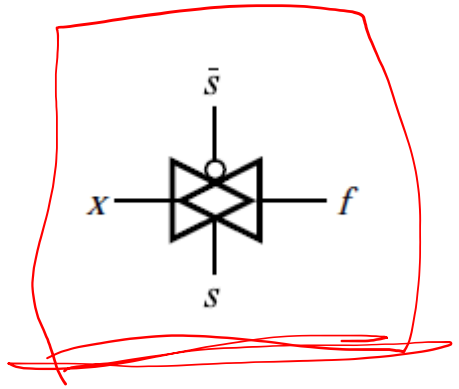


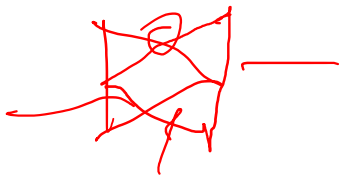
multiplexer

Transmission Gate

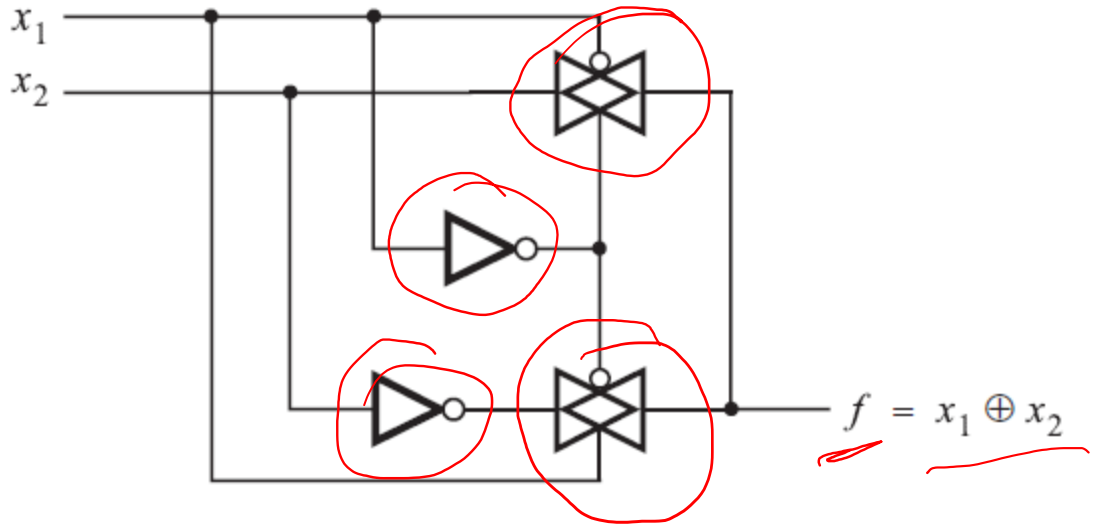


s	f
0	Z
1	x





XOR Gate



COL215L: Digital Logic & System Design

Lecture 6: Combinational Circuits

M. Balakrishnan
CSE@IITD

Vireshwar Kumar
CSE@IITD

August 21, 2021

Notation

AND — $\cdot$

OR — $+$

NOT — $'$

$x \rightarrow$ Input

$y \rightarrow$ Input

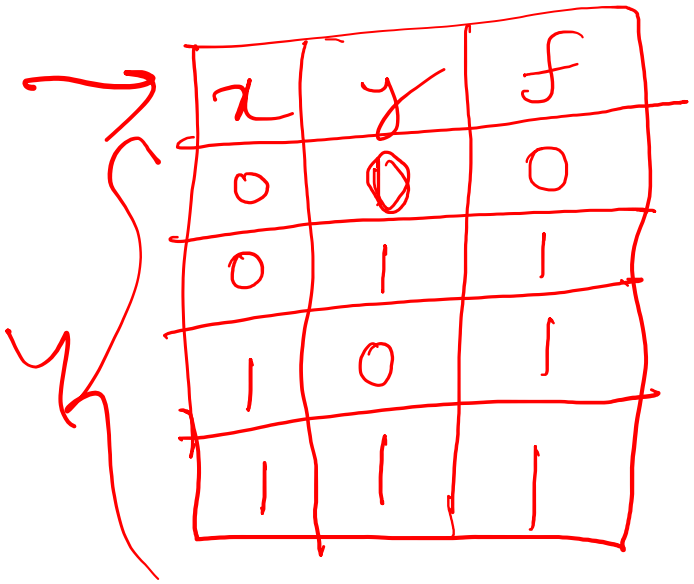
$f \rightarrow$ output

$f = x \cdot y$

$f = x + y$

x' $\quad$ $\bar{x}$

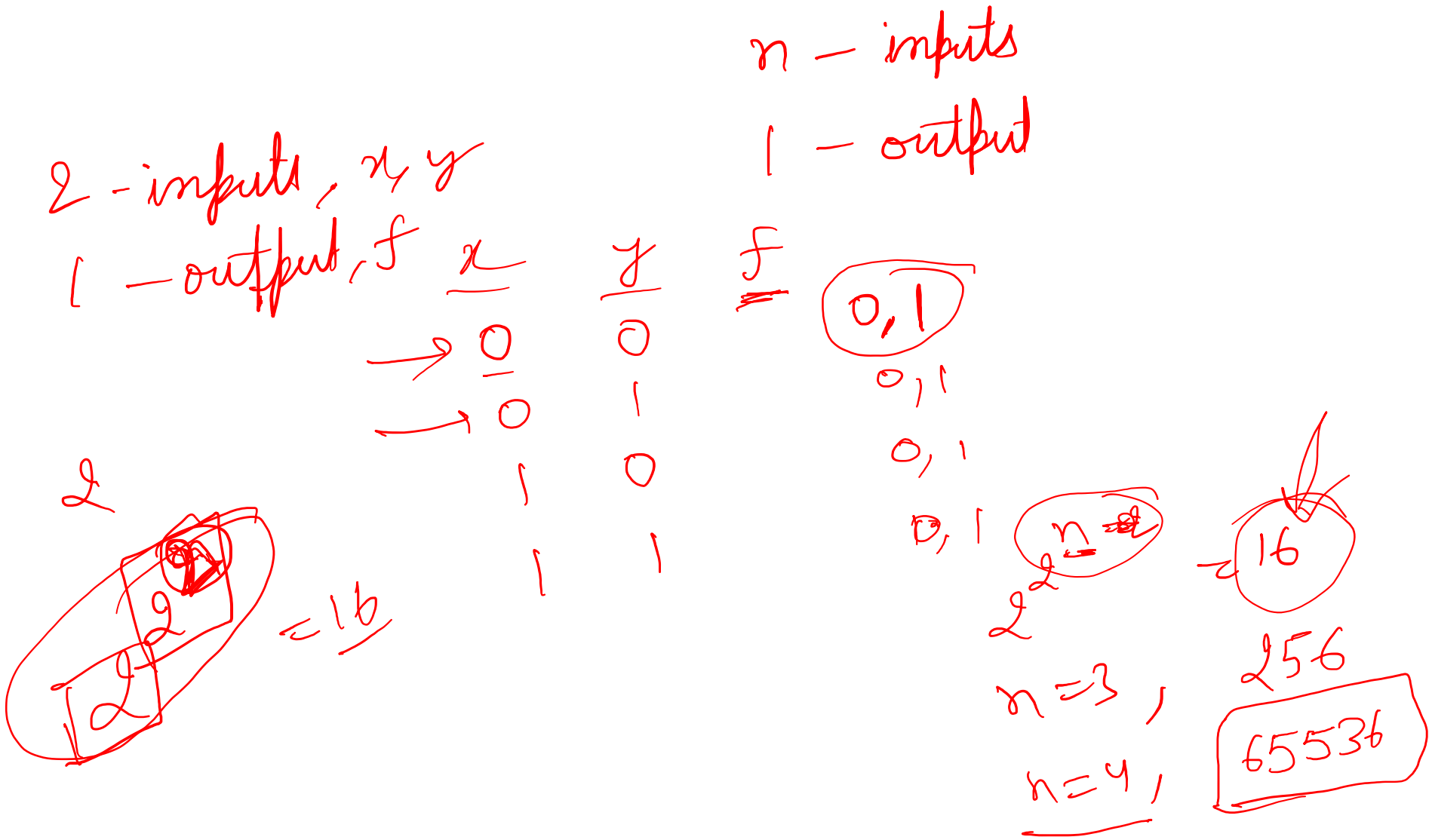
Truth Table



x	y	f
0	0	0
0	1	1
1	0	1
1	1	1

$f = x + y$

Number of Distinct Functions



Universal Gate Set

{AND, OR, NOT} →

→ {NAND}

→ {NOR}

x, y, z

f = x · y + z

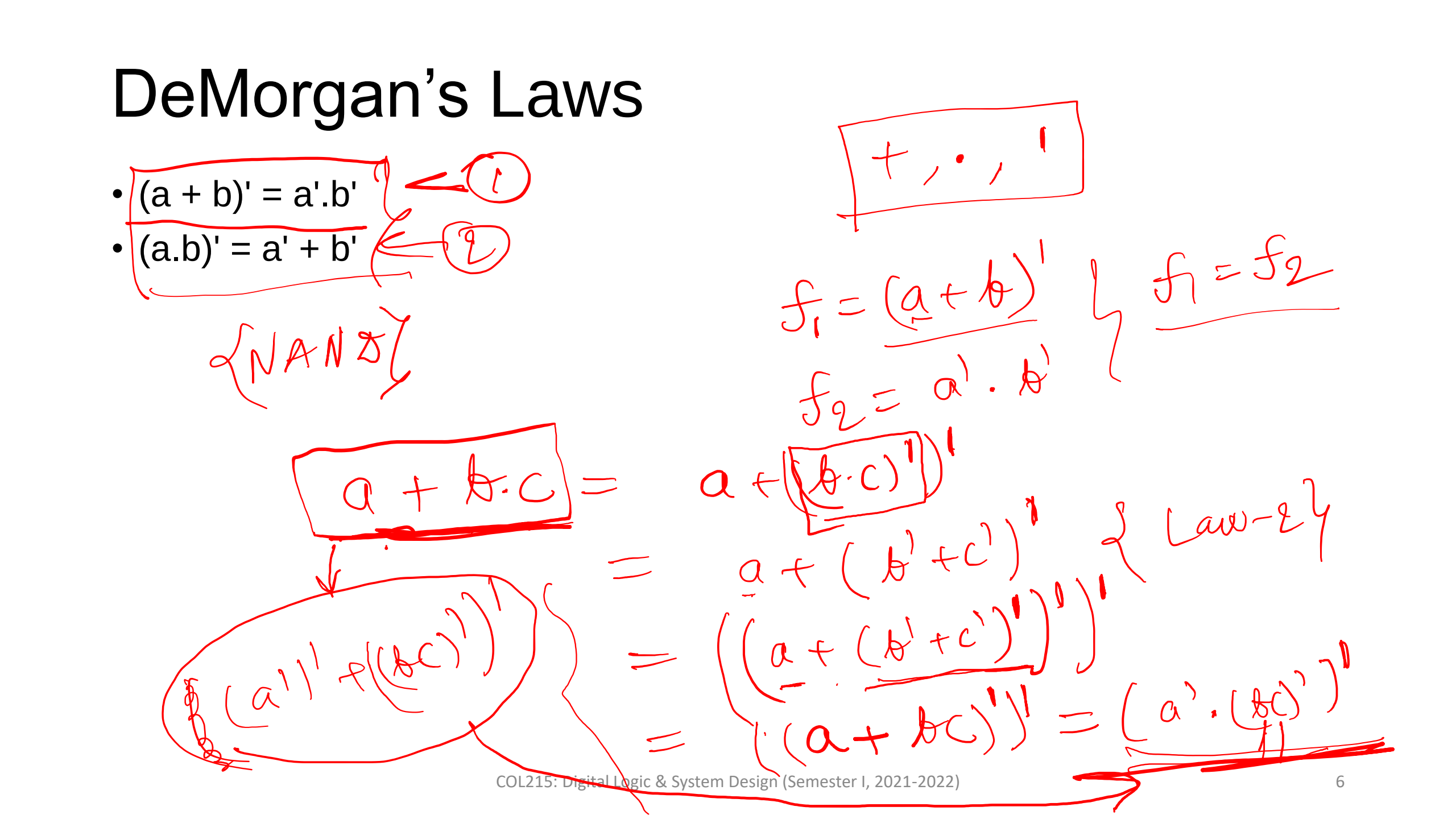
x	y	z	F
0	0	0	1
0	0	1	0
0	1	0	1

f = x' · y' · z'

+ x · y' · z

DeMorgan's Laws

- (a + b)' = a'.b'
- (a.b)' = a' + b'



Duality

- Replace AND with OR
- Replace OR with AND

$$(a+b+c')\cdot d = a\cdot d + b\cdot d + c'\cdot d$$

$$a\cdot b\cdot c' + d = (a+d)\cdot(b+d)\cdot(c'+d)$$

COL215L: Digital Logic & System Design

Lecture 7: Combinational Circuits (Cont.)

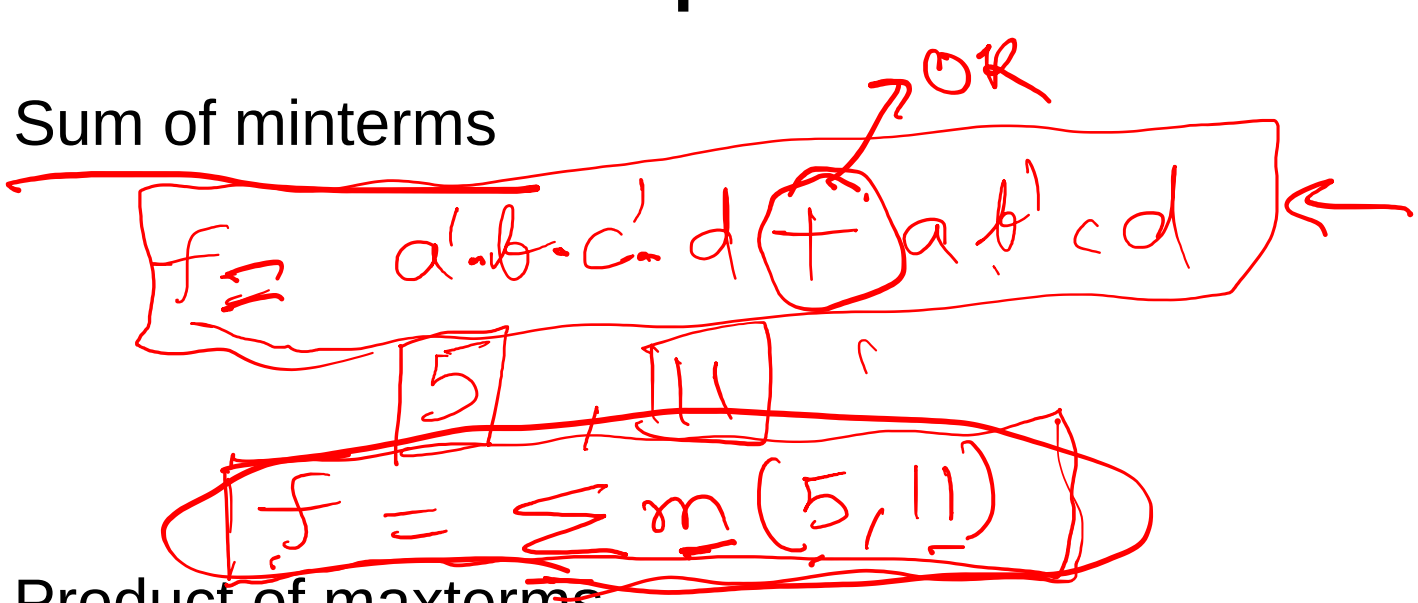
M. Balakrishnan
CSE@IITD

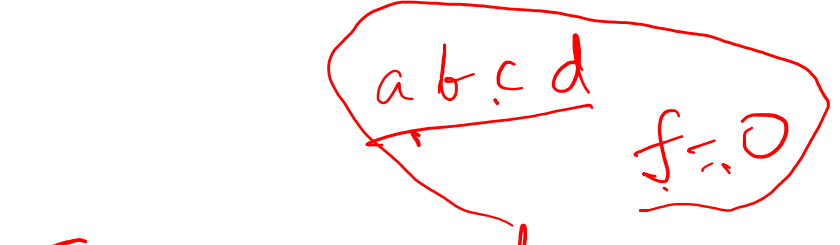
August 24, 2021

Vireshwar Kumar
CSE@IITD

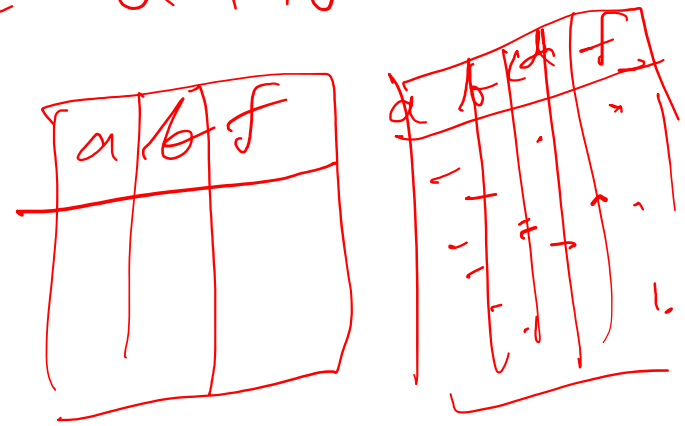
Canonical Representation

- Sum of minterms
- Product of maxterms

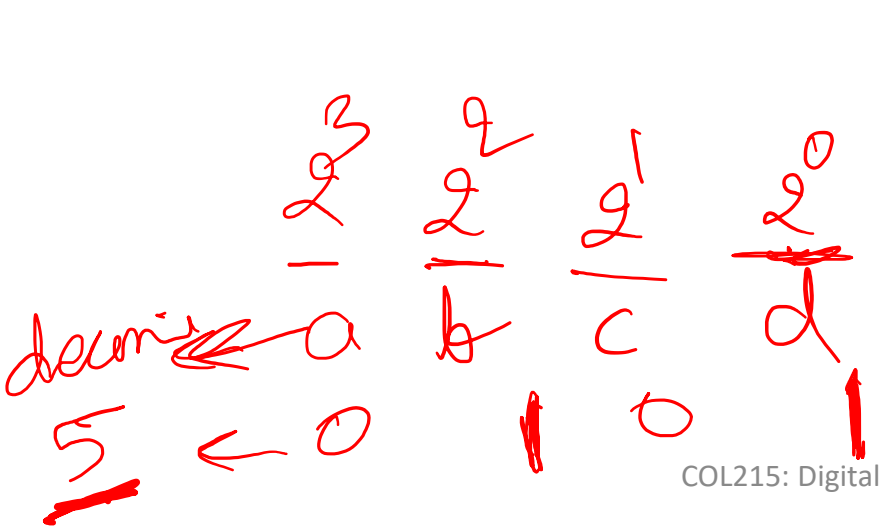


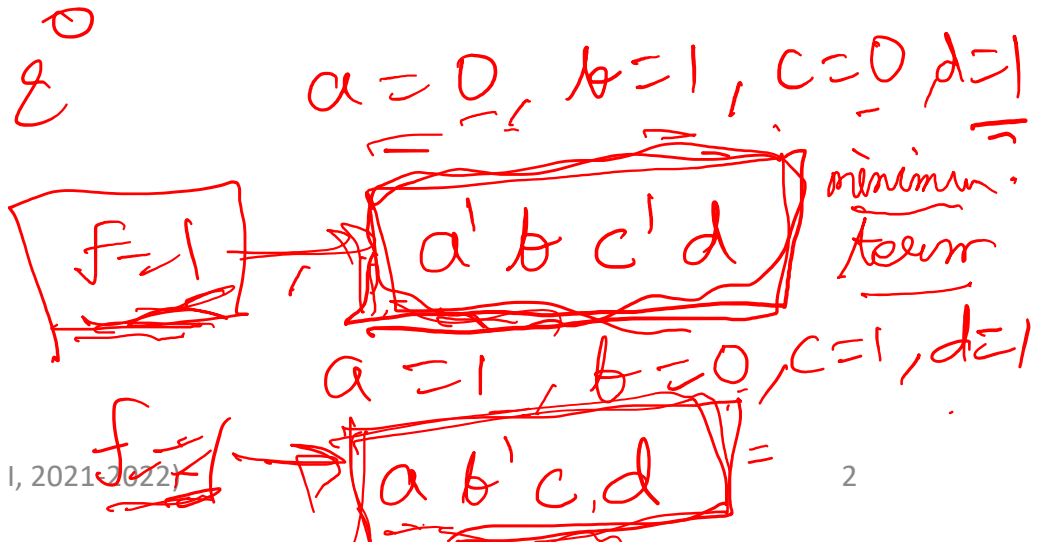


f = a + b



a, b, c, d





product of max terms

$$f = (a^1 + b^1 + c^0 + d^0)\,(a^0 + b^0 + c^1 + d^1)$$

$f=0$, $a=1, b=1, c=0, d=0$

$(a^0 + b^0 + c^1 + d^1)$

$f=0$, $a=0, b=0, c=1, d=1$

$(a^0 + b^0 + c^1 + d^1)$

$2^3 \; 2^2 \; 2^1 \; 2^0$

$a \; b \; c \; d$

$2^3 + 2 + 0 + 0 = 12$

$0 + 0 + 2^1 + 2^0 = 3$

$$f = \Pi M(3, 12)$$

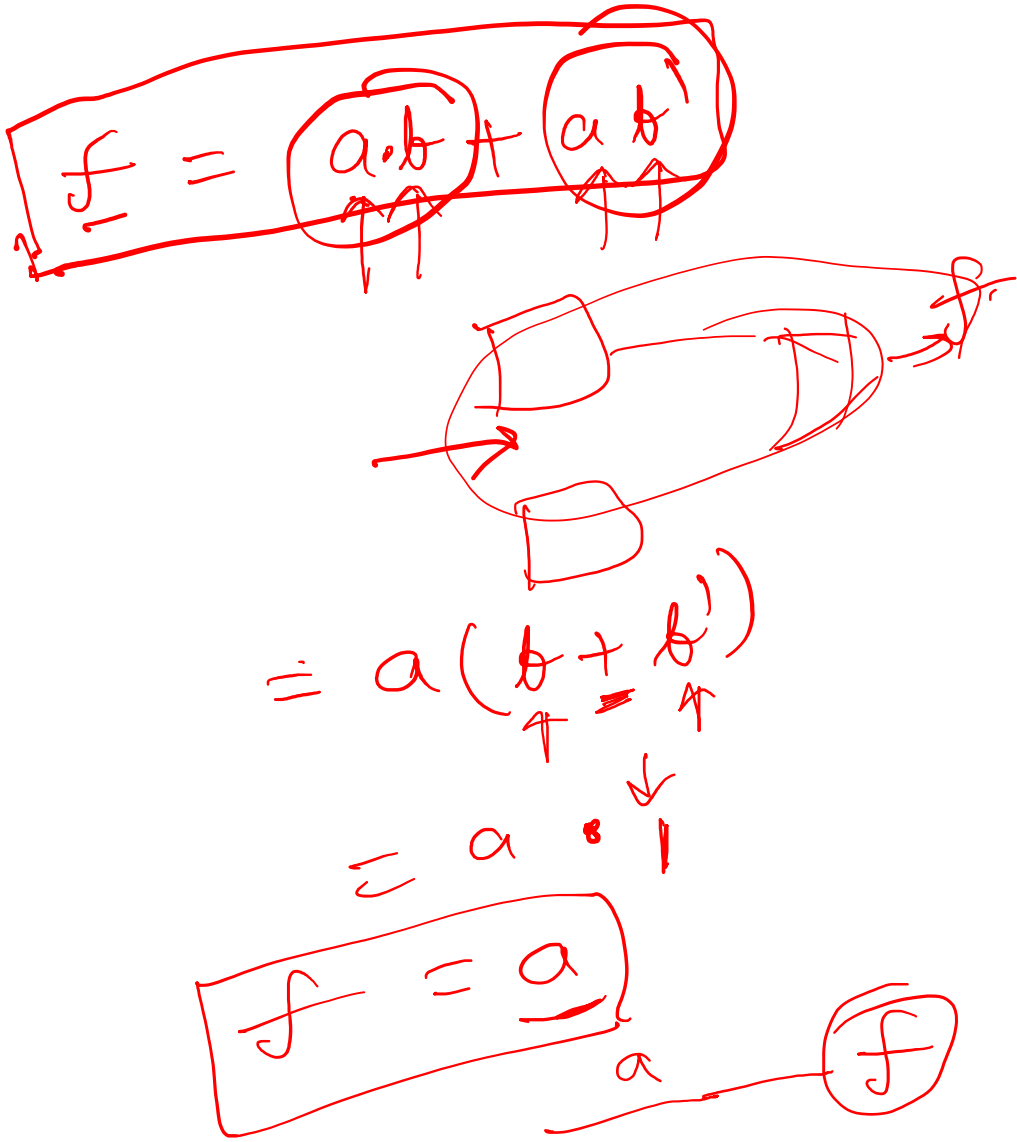
for Minterms

of $f(a, b, c, d)$

Quality of f

Minimization Objectives

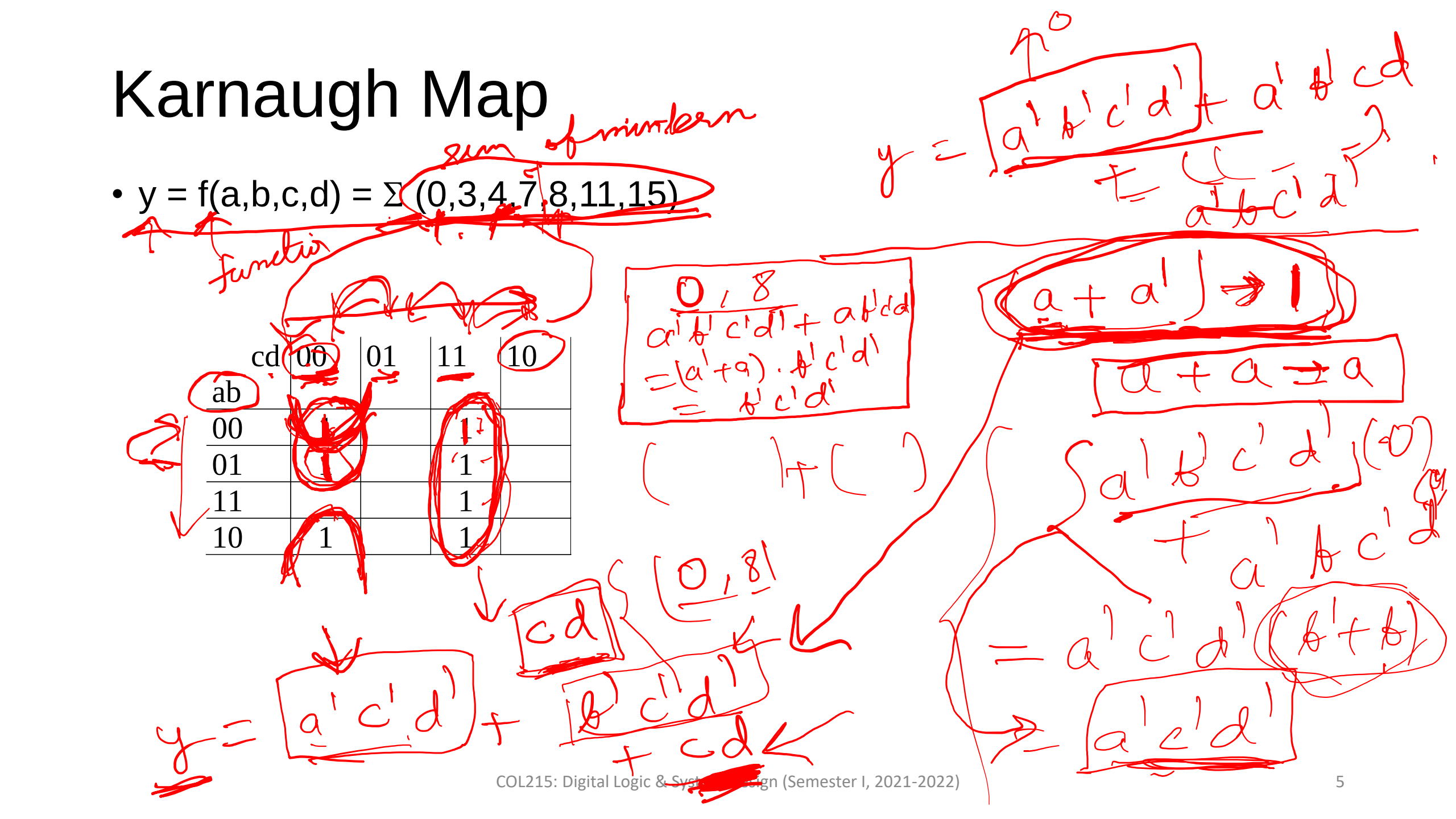
- Cost/area
- Time



Karnaugh Map

- y = f(a,b,c,d) = Σ (0,3,4,7,8,11,15)

ab \ cd	00	01	11	10
00	1		1	
01	1		1	
11			1	
10	1		1	



Karnaugh Map (Cont.)

- $y = f(a,b,c,d) = \Pi\ (1,5,9,13,15)$

ab \ cd	00	01	11	10
00		0		
01		0		
11		0	0	
10		0		

$y = (a'+b'+d') \cdot (c+d')$

Challenges with K-Map

- K-Map
  - A graphical method and thus not suitable for large no. of variables
  - Not suitable for programming

- QM (Quine-Mcluskey) method
  - Does not suffer from these disadvantages

COL215L: Digital Logic & System Design

Lecture 8: Combinational Circuits (Cont.)

M. Balakrishnan
CSE@IITD

August 25, 2021

Vireshwar Kumar
CSE@IITD

Challenges with K-Map

- K-Map
  - A graphical method and thus not suitable for large no. of variables
  - Not suitable for programming

- Quine-Mcluskey (QM) method
  - Does not suffer from these disadvantages

QM Method

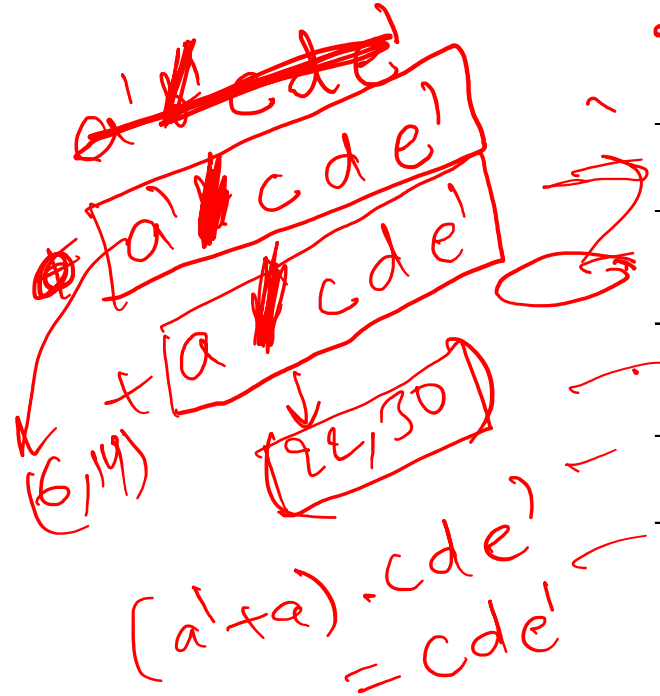
- $y = f(a,b,c,d,e) = \Sigma\, m(0,6,7,8,14,20,22,24,26,30,31)$
- Step-1: Order by number of 1's in the binary representation

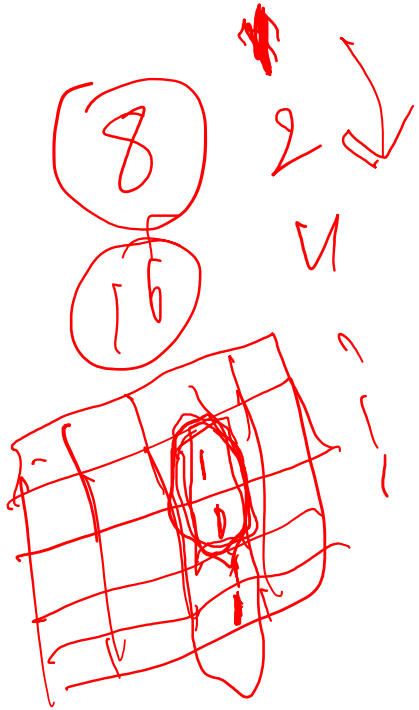
No. of 1's	Minterms
0	0
1	8
2	6, 20, 24
3	7, 14, 22, 26
4	30
5	31

QM Method (Cont.)

- Step-2: Combine minterms considering following conditions
  - The integer values of the two minterms differ by 2^k for some $k \geq 0$.
  - The minterm with a larger integer value has one more 1's than the minterm with a smaller integer value.
- The combination of the minterms is called an implicant.

0	0	(0,8)	
1	8	(8,24)	
2	6, 20, 24	(6,7), (6,14), (6,22), (20,22), (24,26)	(6,14,22,30), (6,22,14,30)
3	7, 14, 22, 26	(14,30), (22,30), (26,30)	
4	30	(30,31)	
5	31		

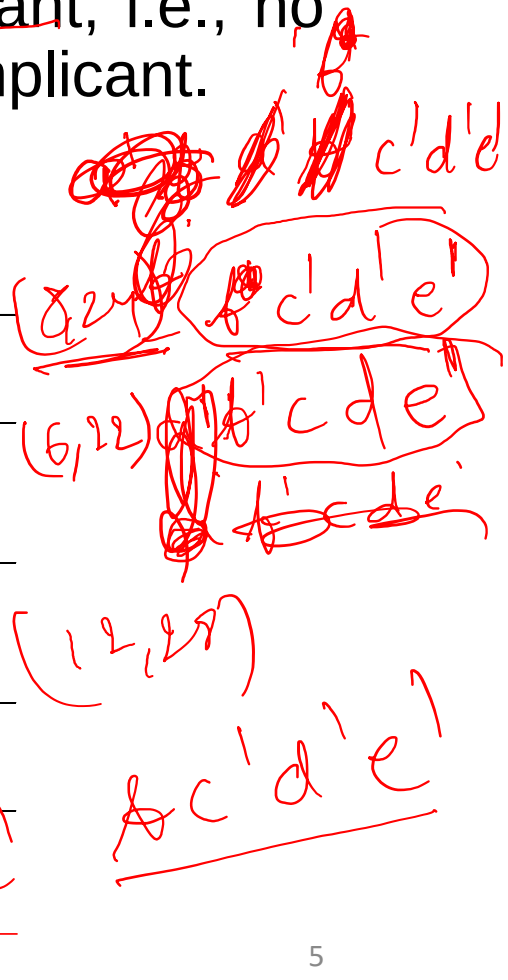




QM Method (Cont.)

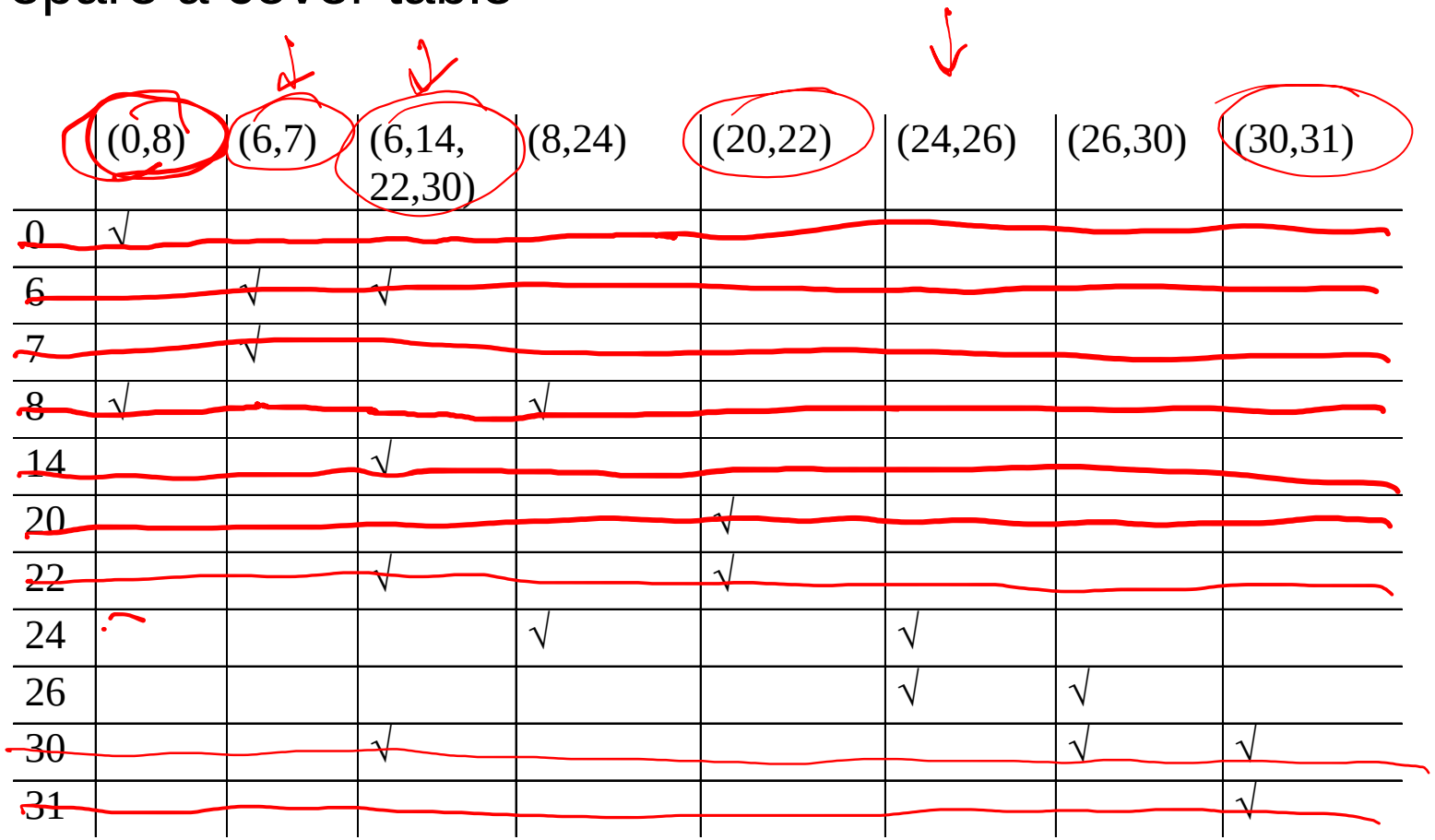
- Step-3: Identify prime implicants
  - A prime implicant is not fully contained in any other implicant, i.e., no other implicant contains all the minterms contained in this implicant.

0	0	(0,8) √	
1	8	(8,24) √	
2	6, 20, 24	(6,7), √ (6,14), (6,22), (20,22), √ (24,26) √	(6,14,22,30), √ (6,22,14,30)
3	7, 14, 22, 26	(14,30), (22,30), (26,30) √	
4	30	(30,31) √	
5	31		



QM Method (Cont.)

- Step-4: Prepare a cover table



	(0,8)	(6,7)	(6,14, 22,30)	(8,24)	(20,22)	(24,26)	(26,30)	(30,31)
0	√							
6		√	√					
7		√						
8	√			√				
14			√					
20					√			
22			√		√			
24				√		√		
26						√	√	
30			√				√	√
31								√

$y = a'c'd'e' + (\quad)$

Essential Prime Implicants

- Step-5: Identify essential prime implicants
  - Essential prime implicants are those which cover minterm(s) not covered by any other prime implicant.
- Any minimal function have to include the essential prime implicants
- There can be some non-essential prime implicants

QM Method: Steps

1. Order minterms by number of 1’s
2. Generate implicants by combination of minterms/implicants
3. Identify prime implicants
4. Prepare a cover table
5. Identify essential/other prime implicants covering all minterms
6. Express the function as a sum of identified prime implicants

COL215L: Digital Logic & System Design

Lecture 9: Combinational Circuits (Cont.)

M. Balakrishnan
CSE@IITD

August 27, 2021

Vireshwar Kumar
CSE@IITD

Incompletely Specified Functions

- Don’t-care conditions
  - Outputs actually do not matter for certain subset of inputs
  - A subset of inputs do not (or cannot) occur

- K-Map
  - Include *don’t care* minterms

- QM Method
  - Use *don’t care* minterms for generation of the implicants
  - Ignore *don’t care* minterms during the covering process

Incompletely Specified Functions (Cont.)

ab \ cd	00	01	11	10
00			1	1
01	1	1		1
11	X	X	X	X
10	1	1	X	X

Multiplexer

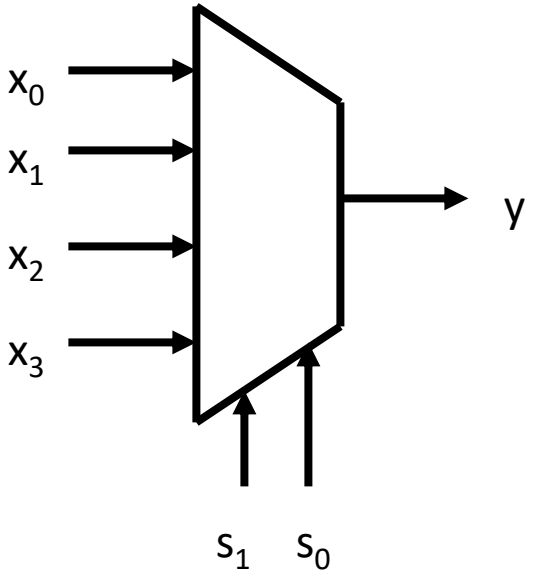
- 2-to-1 Multiplext

Multiplexer (Cont.)

Multiplexer (Cont.)

- 2^n-to-1 Multiplexer (n select lines)

- 4-to-1 Multiplexer (2 select lines)
  - Exercise
    - Complete truth table
    - Realize using logic gates



De-multiplexer

- 1-to-2 Demultiplexer (1 select line)

- 1-to-4 Demultiplexer (2 select lines)
  - Exercise
    - Complete truth table
    - Realize using logic gates

Decoder

- n-2^n
- 2-to-4 decoder with enable

Read-Only Memory (ROM)

COL215L: Digital Logic & System Design

Lecture 10: Binary Arithmetic

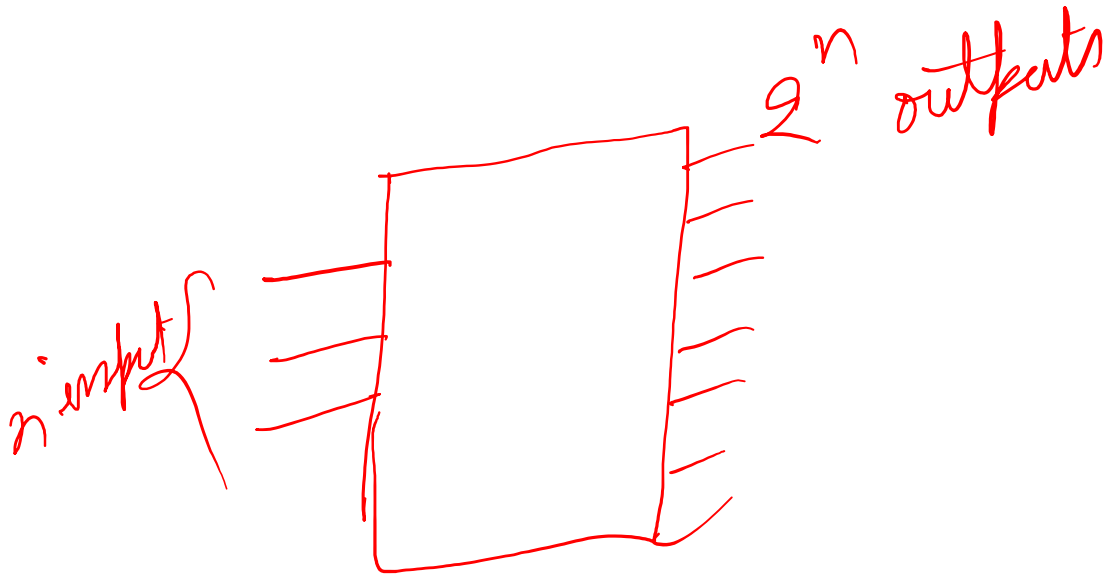
M. Balakrishnan
CSE@IITD

Vireshwar Kumar
CSE@IITD

August 31, 2021

Decoder

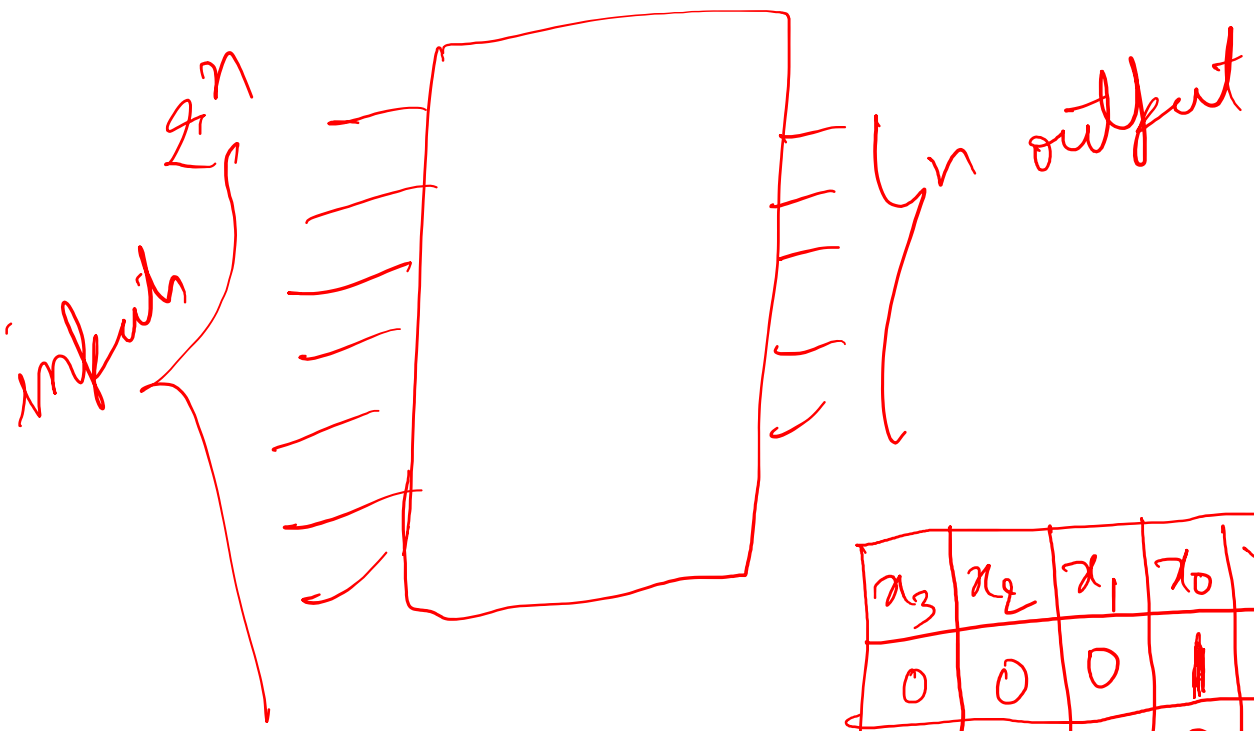
- n-2^n



Encoder

- 2^n-n

- 4-to-2 Encoder



x_3	x_2	x_1	x_0	y_1	y_0
0	0	0	1	0	0
0	0	1	0	0	1
0	1	0	0	1	0
1	0	0	0	1	1

$y_1 =$ ( )

$y_0=$ ( )

Priority Encoder

- 2^n-n

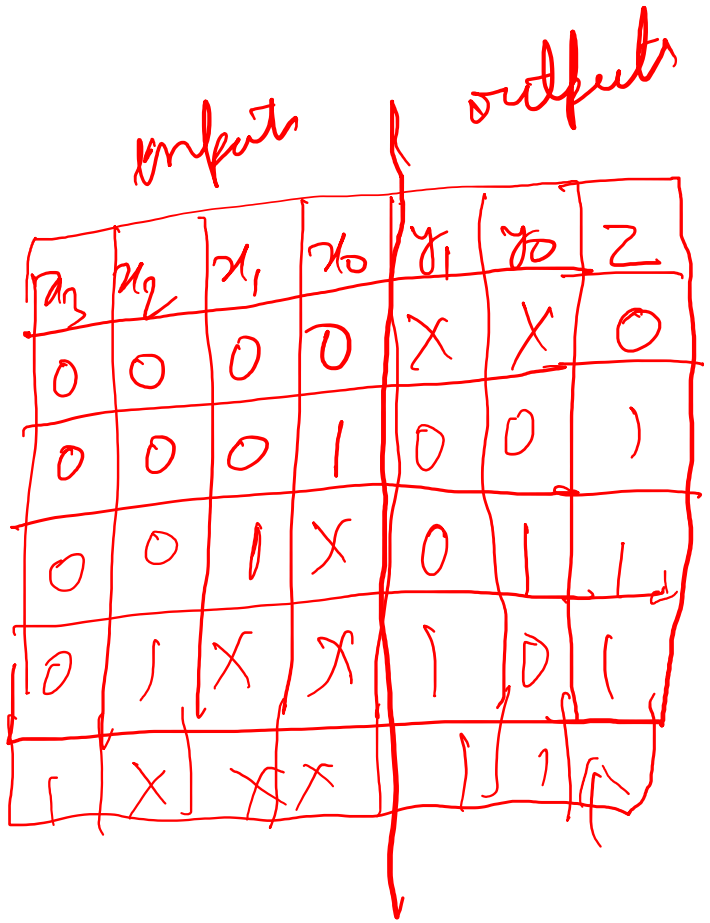
- Inputs have priorities

4 inputs $x_3 > x_2 > x_1 > x_0$

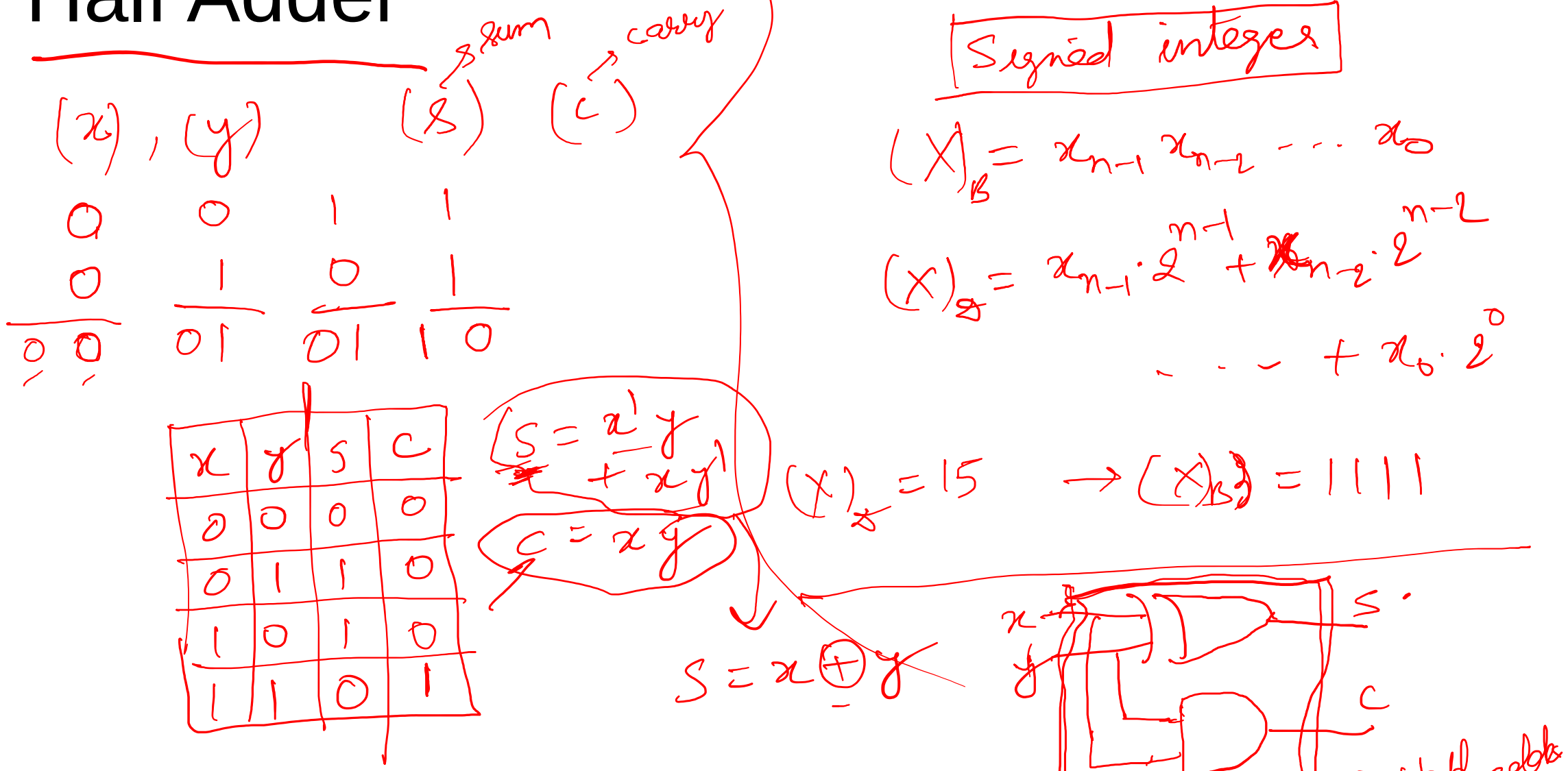
$z = (x_3 + x_2 + x_1 + x_0)$

$y_1 =$ ( )

$y_0 =$ ( )



Half Adder


sum
carry
(x), (y) (s) (c)
0 0 1 1
0 1 0 1
0 1 0 1 1 0
x y s c
0 0 0 0
0 1 1 0
1 0 1 0
1 1 0 1
s = x'y + xy'
c = xy
s = x ⊕ y
Unsigned integer
Signed integer
(X)_B = x_{n-1} x_{n-2} ... x_0
(X)_S = x_{n-1}·2^{n-1} + x_{n-2}·2^{n-2} ... + x_0·2^0
(X)_S = 15 → (X)_B = 1111
x
y
s
c
half adder

Full Adder

c_i	x_i	y_i	c_{i+1}	s_i
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

$$c_{i+1} = x_i y_i + x_i c_i + y_i c_i$$

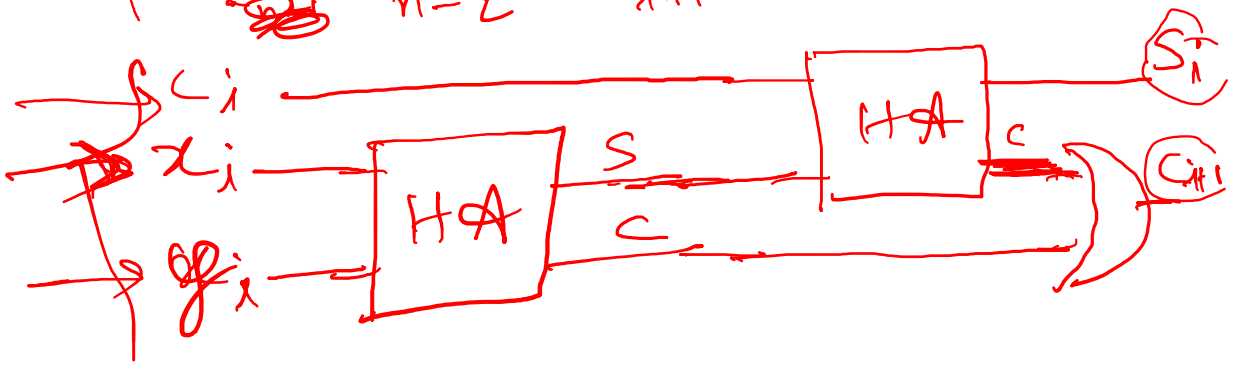
$$s_i = x_i \oplus y_i \oplus c_i$$

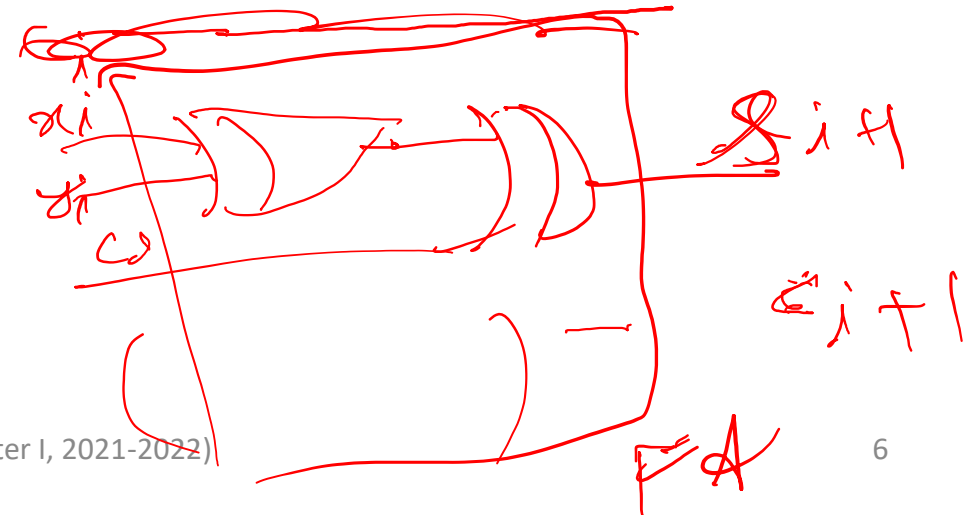
$x_{n-1} x_{n-2} \cdots x_{i+1} x_i \cdots x_0$

$y_{n-1} y_{n-2} \cdots y_{i+1} y_i \cdots y_0$

$s_{n-1} s_{n-2} \cdots s_{i+1} s_i \cdots s_0$

$c_{n-2} \cdots c_{i+1} c_i \cdots c_0$





COL215L: Digital Logic & System Design

Lecture 11: Binary Arithmetic (Cont.)

M. Balakrishnan
CSE@IITD

Vireshwar Kumar
CSE@IITD

September 1, 2021

Signed Numbers

- Sign-and-Magnitude
  - MSB bit – sign
- 1' Complement
  - Neg - complementing bits
  - N = 1111 - P
- 2' Complement
  - Complement and add one
  - N = 10000 – P

$(N)_B = (2^n - 1) - (P)_B$

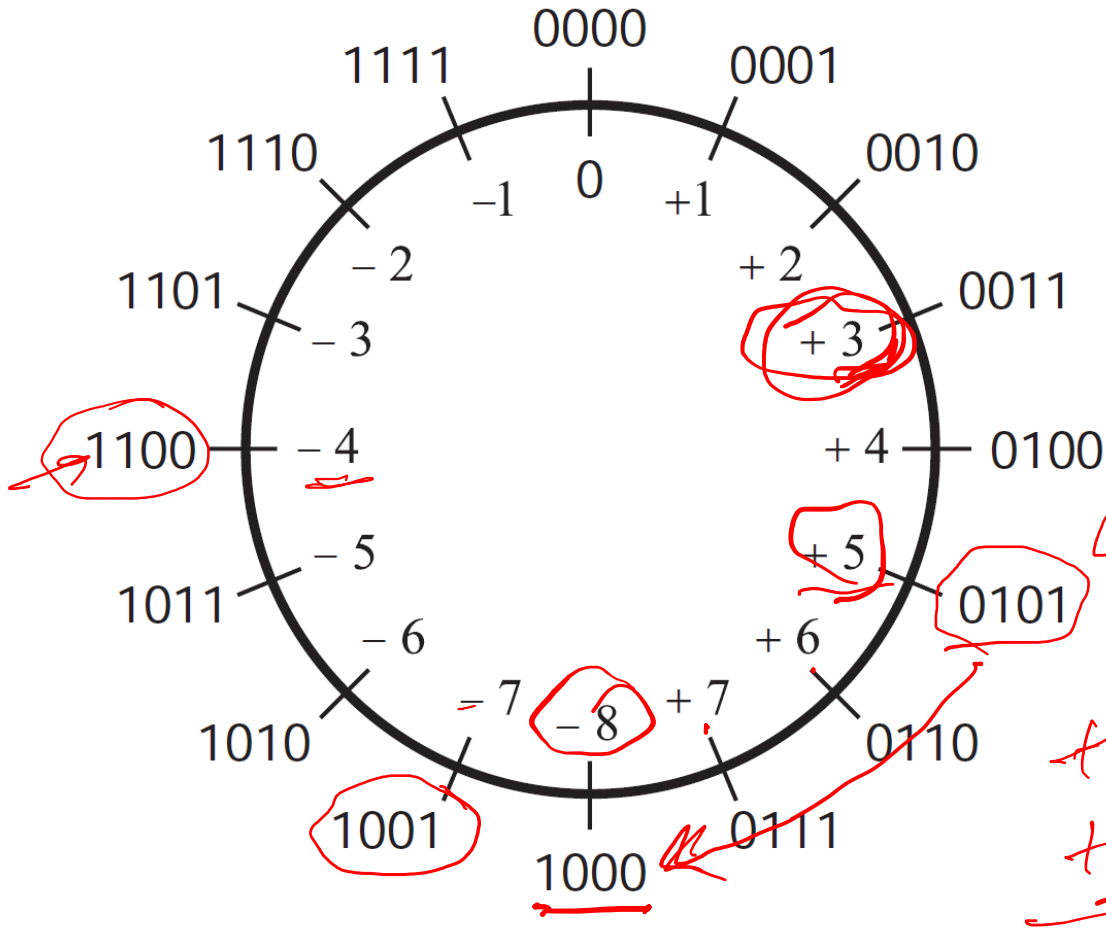
$(N)_B = 2^n - (P)_B$

$b_3b_2b_1b_0$	Sign and magnitude	1's complement	2's complement
0111	+7	+7	+7
0110	+6	+6	+6
0101	+5	+5	+5
0100	+4	+4	+4
0011	+3	+3	+3
0010	+2	+2	+2
0001	+1	+1	+1
0000	+0	+0	+0
1000	−0	−7	−8
1001	−1	−6	−7
1010	−2	−5	−6
1011	−3	−4	−5
1100	−4	−3	−4
1101	−5	−2	−3
1110	−6	−1	−2
1111	−7	−0	−1

1010
0001
1011

2's Complement Addition/Subtraction

- Circular
- Overflow



COL215L: Digital Logic & System Design

Lecture 12: Binary Arithmetic (Cont.)

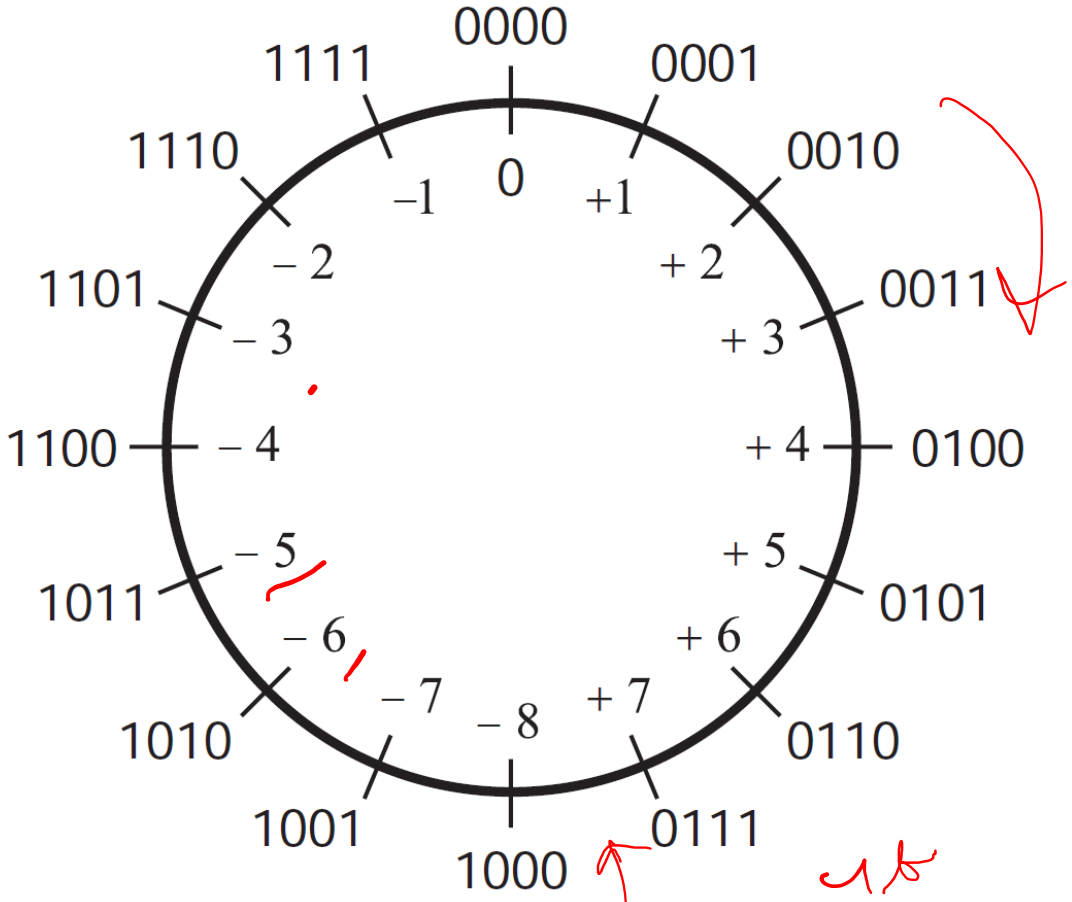
M. Balakrishnan
CSE@IITD

Vireshwar Kumar
CSE@IITD

September 3, 2021

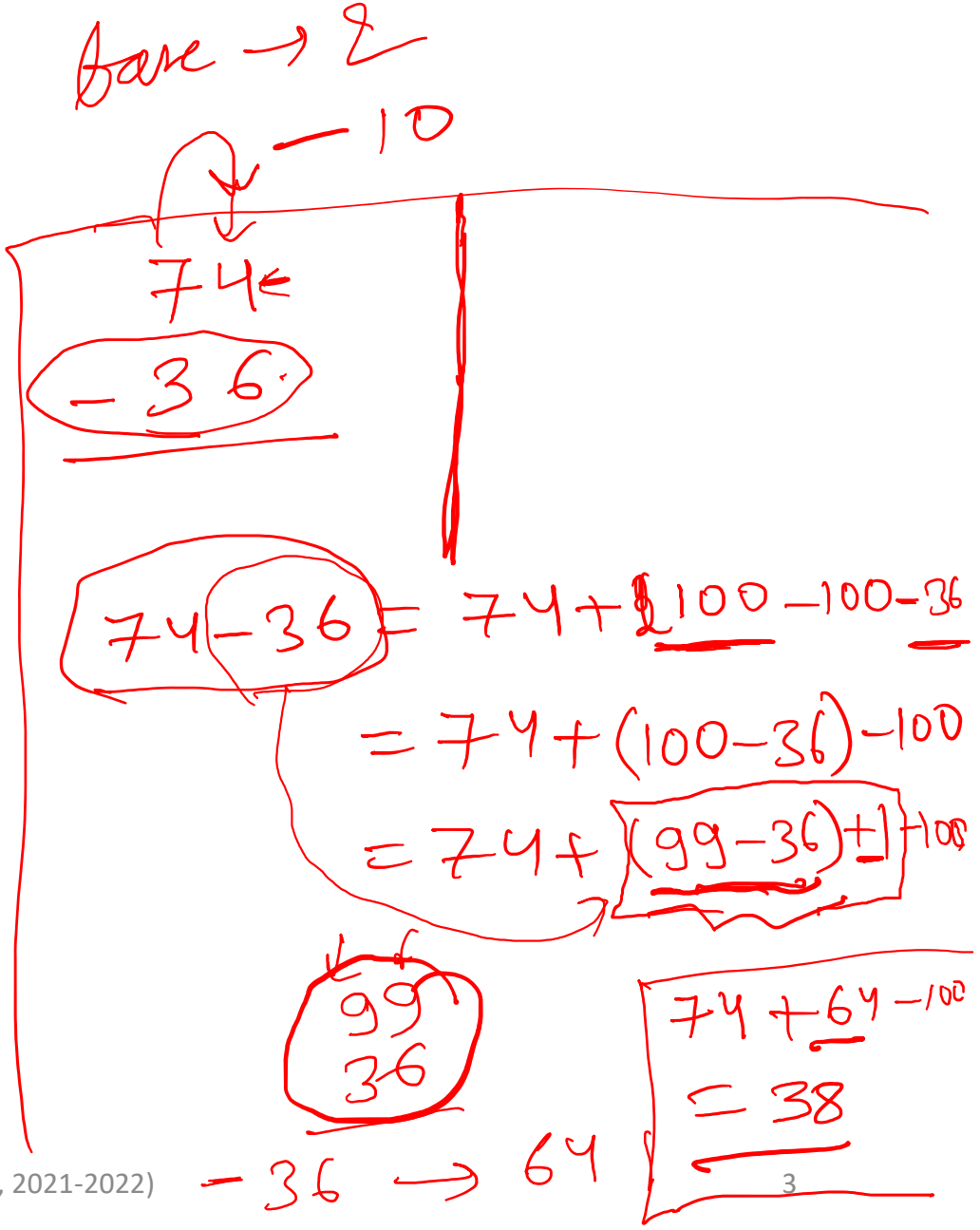
2's Complement Addition/Subtraction

- Circular
- Overflow



Radix Representation

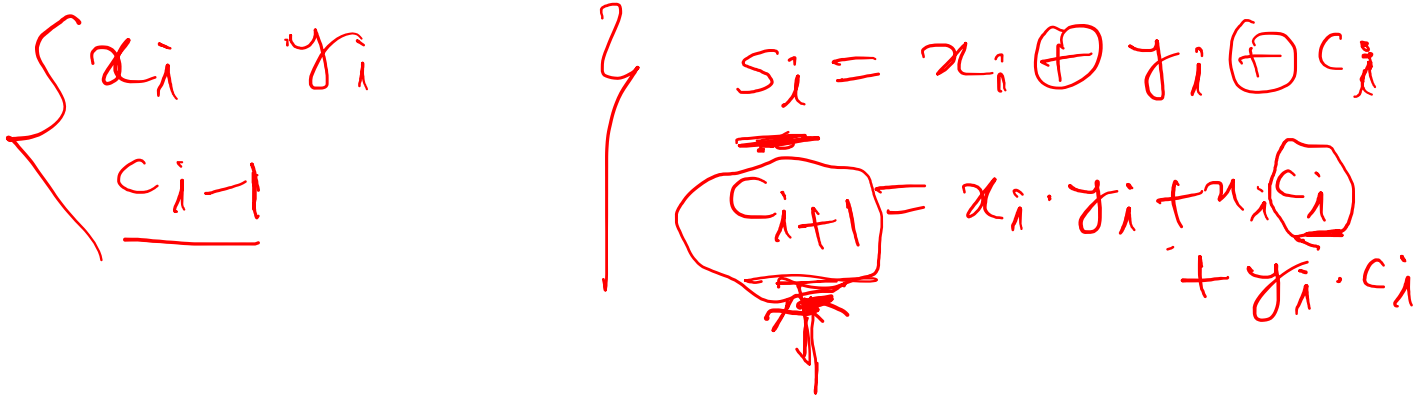
- Radix/base – R
- (R-1)' complement
  - $N = R^n - 1 - P$
- R' complement
  - $N = R^n - P$



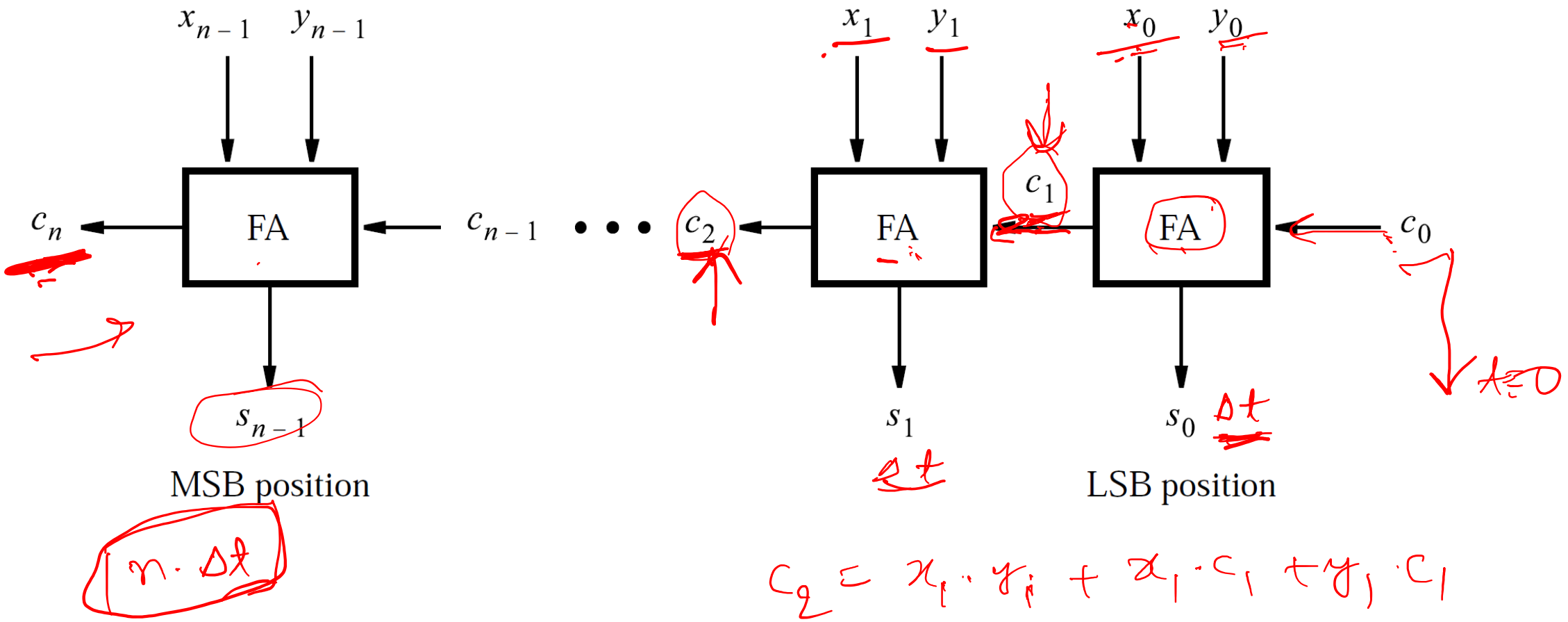
Binary Arithmetic

- Adding 1-bit numbers
  - Half adder
  - Full adder

- Adding N-bit numbers
  - Combination of full adders



Ripple Carry Adder



Carry-Lookahead Adder

- Generate function

$$g_i = x_i \cdot y_i$$

- Propagate function

$$p_i = x_i + y_i$$

$$c_{i+1} = x_i \cdot y_i + x_i c_i + y_i c_i$$

$$= x_i \cdot y_i + (x_i + y_i) \cdot c_i$$

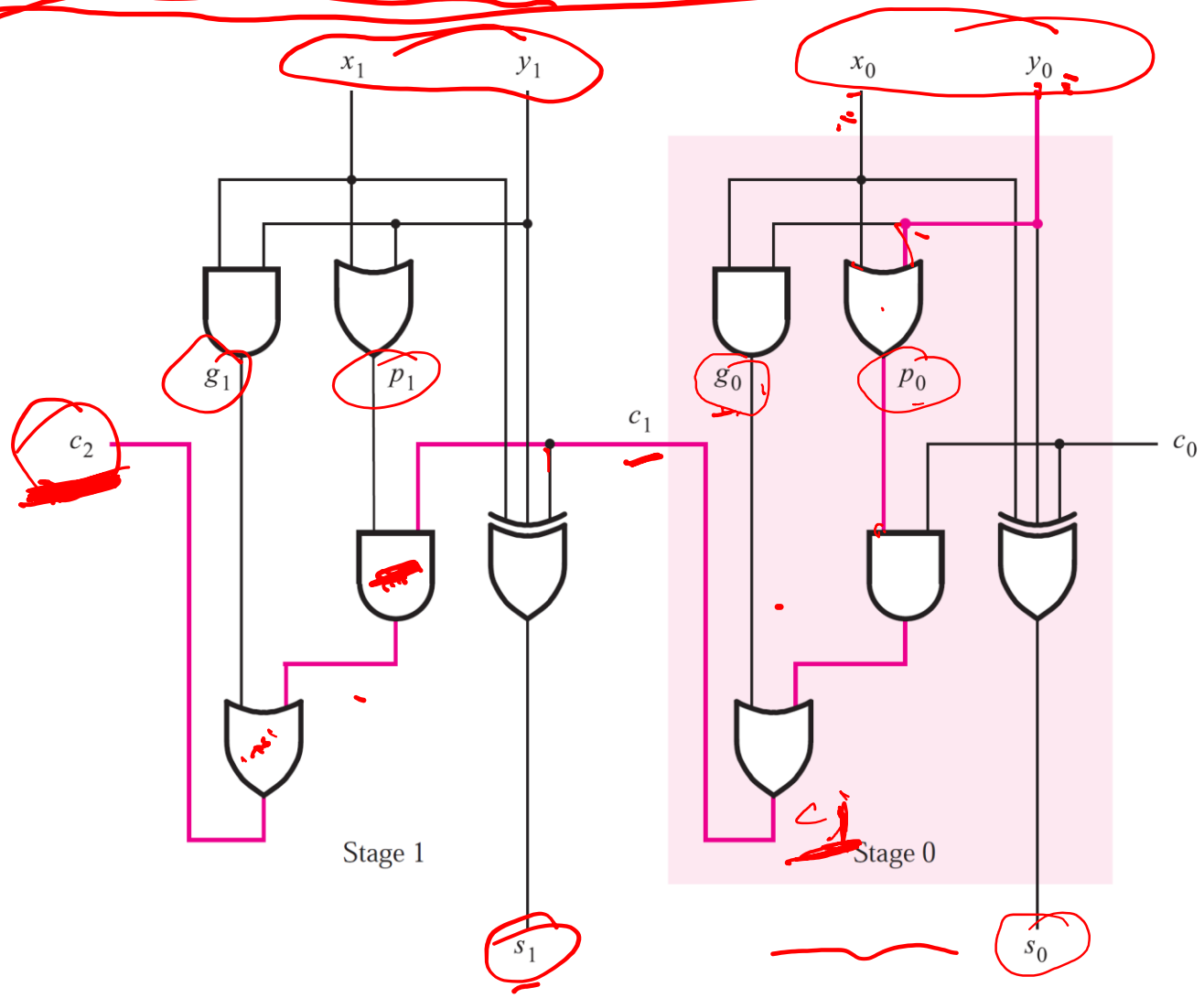
$$\Rightarrow c_{i+1} = g_i + p_i \cdot c_i$$

$$= g_i + p_i (g_{i-1} + p_{i-1} \cdot c_{i-1})$$

$$\Rightarrow c_{i+1} = g_i + p_i \cdot g_{i-1} + p_i \cdot p_{i-1} \cdot g_{i-2}$$

$$- - - - p_1 \cdot p_0 \cdot c_0$$

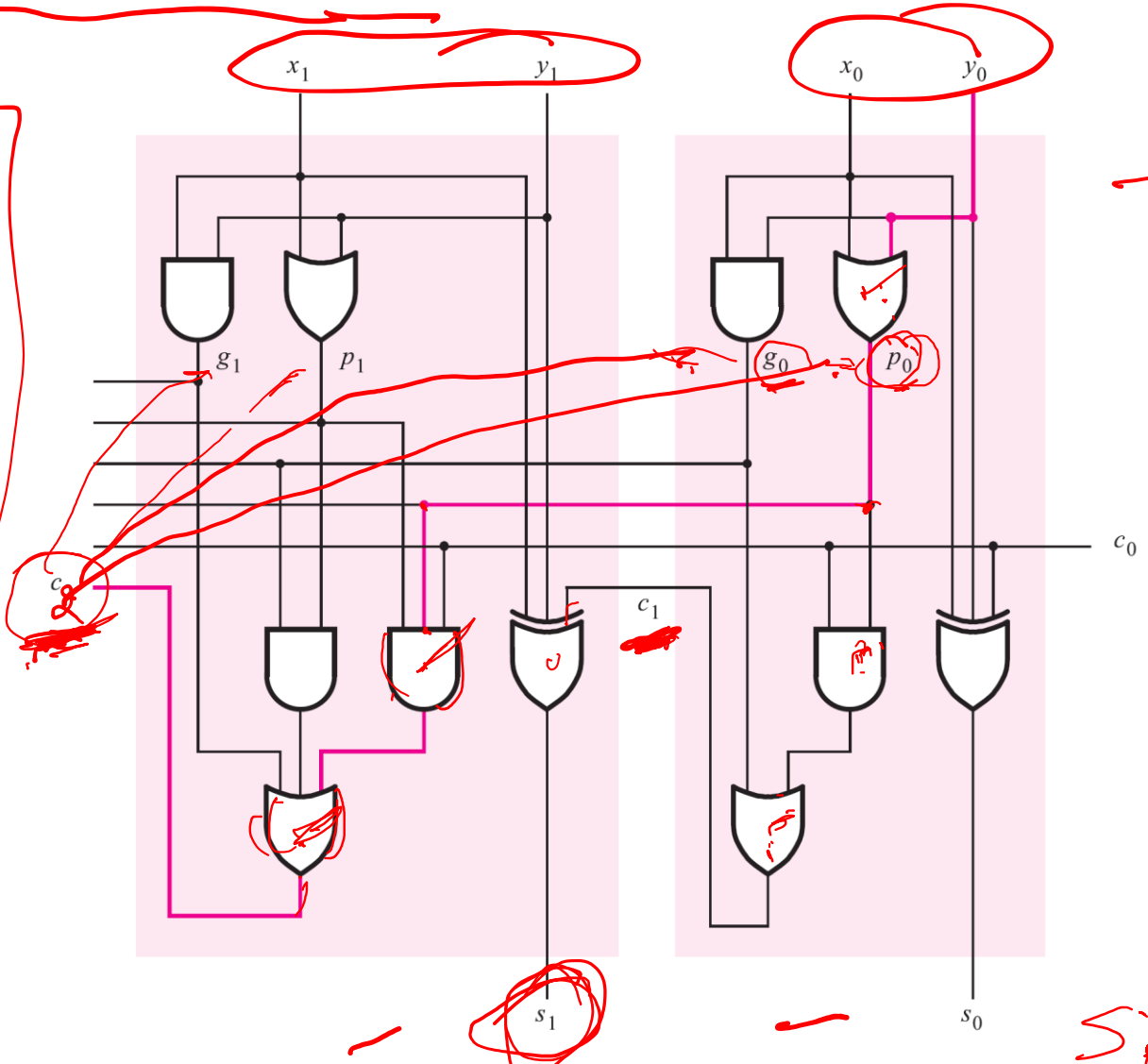
Ripple Carry Adder Circuit



n - bit

$2n+1$

Carry-Lookahead Adder Circuit



COL215L: Digital Logic & System Design

Lecture 13: Binary Arithmetic (Cont.)

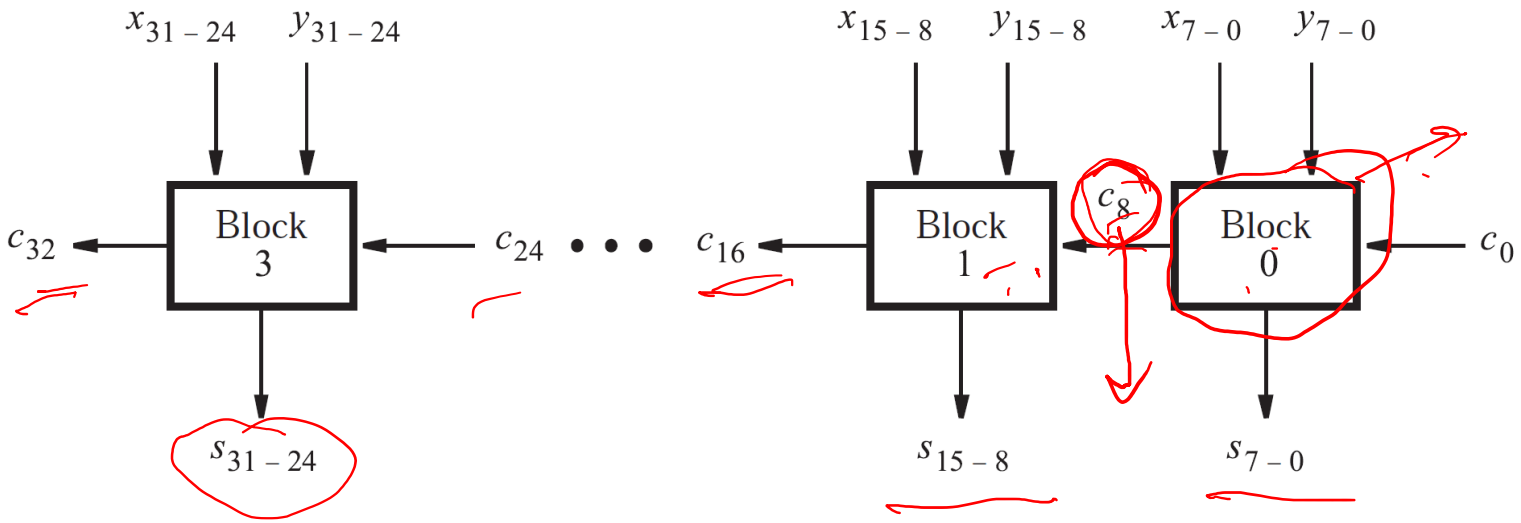
M. Balakrishnan
CSE@IITD

September 7, 2021

Vireshwar Kumar
CSE@IITD

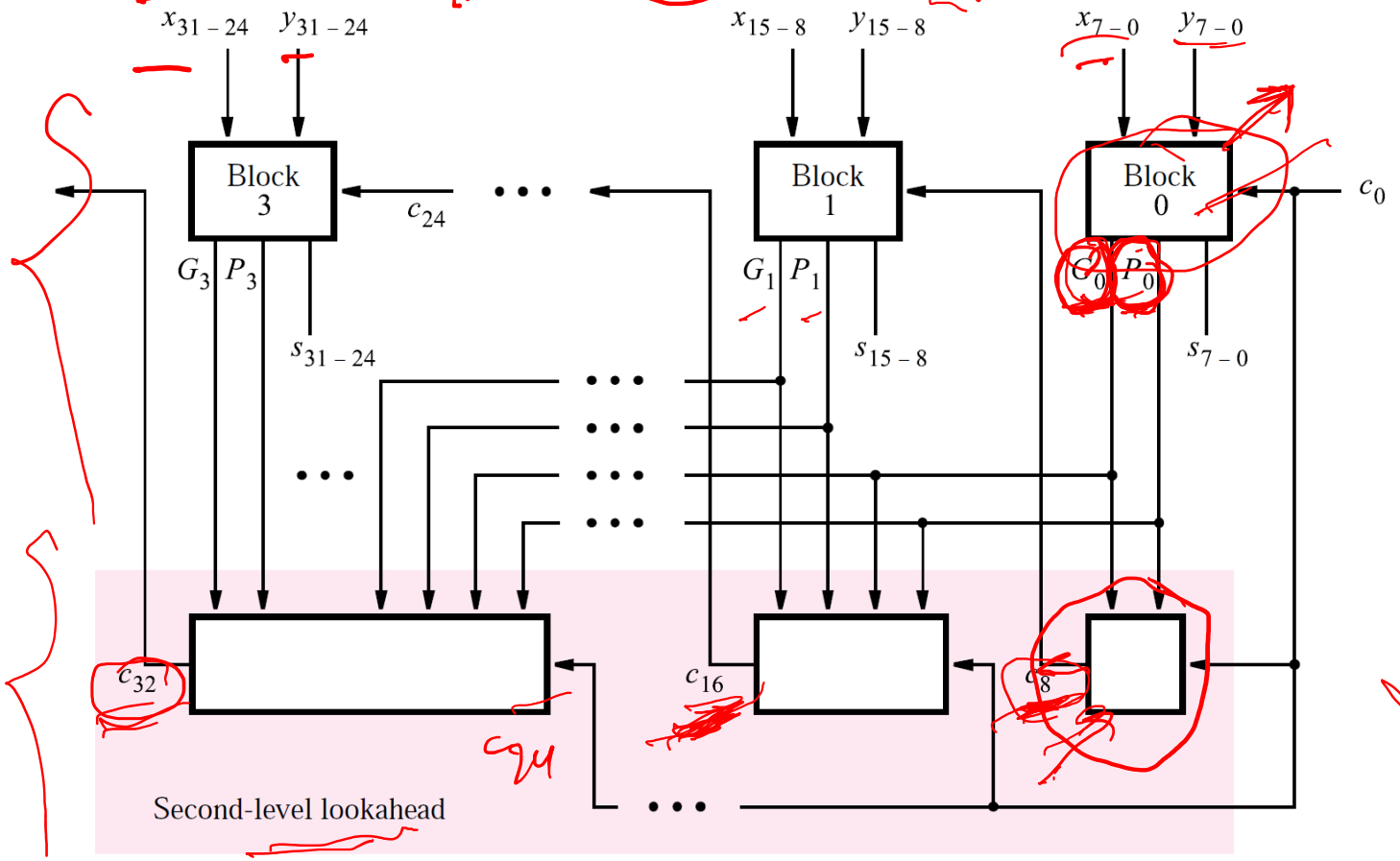
Hierarchical Design

- 32-bit adder
  - 4 blocks of 8-bit adders with ripple-carry between blocks



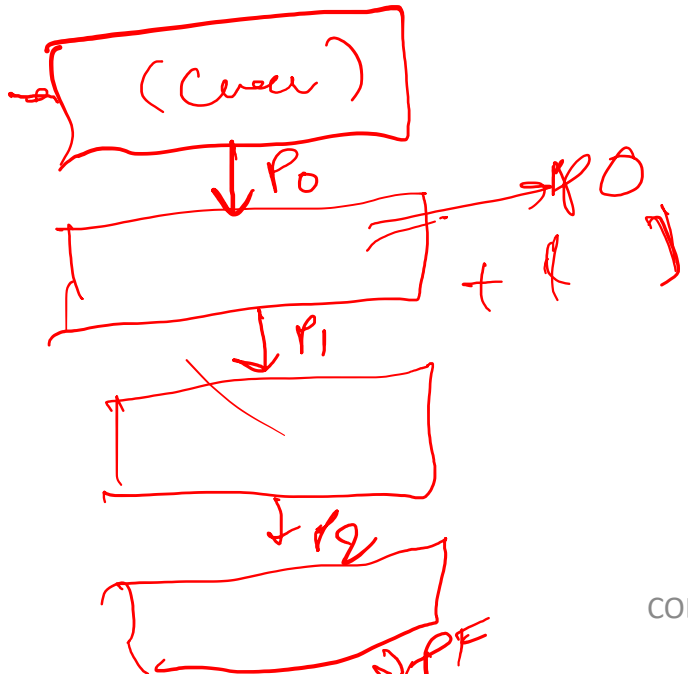
Hierarchical Design (Cont.)

- 32-bit adder
  - 4 blocks of 8-bit adders with carry-lookahead between blocks



Array Multiplication of Unsigned Numbers

- Example: Multiply a number B by 2
- Example: Multiply 11 and 14



Multiplicand M	(11)	1 1 1 0
Multiplier Q	(14)	1 0 1 1
Partial product 0		1 1 1 0
		+ 1 1 1 0
Partial product 1		1 0 1 0 1
		+ 0 0 0 0
Partial product 2		0 1 0 1 0
		+ 1 1 1 0
Product PF	(154)	1 0 0 1 1 0 1 0

LSB

COL215L: Digital Logic & System Design

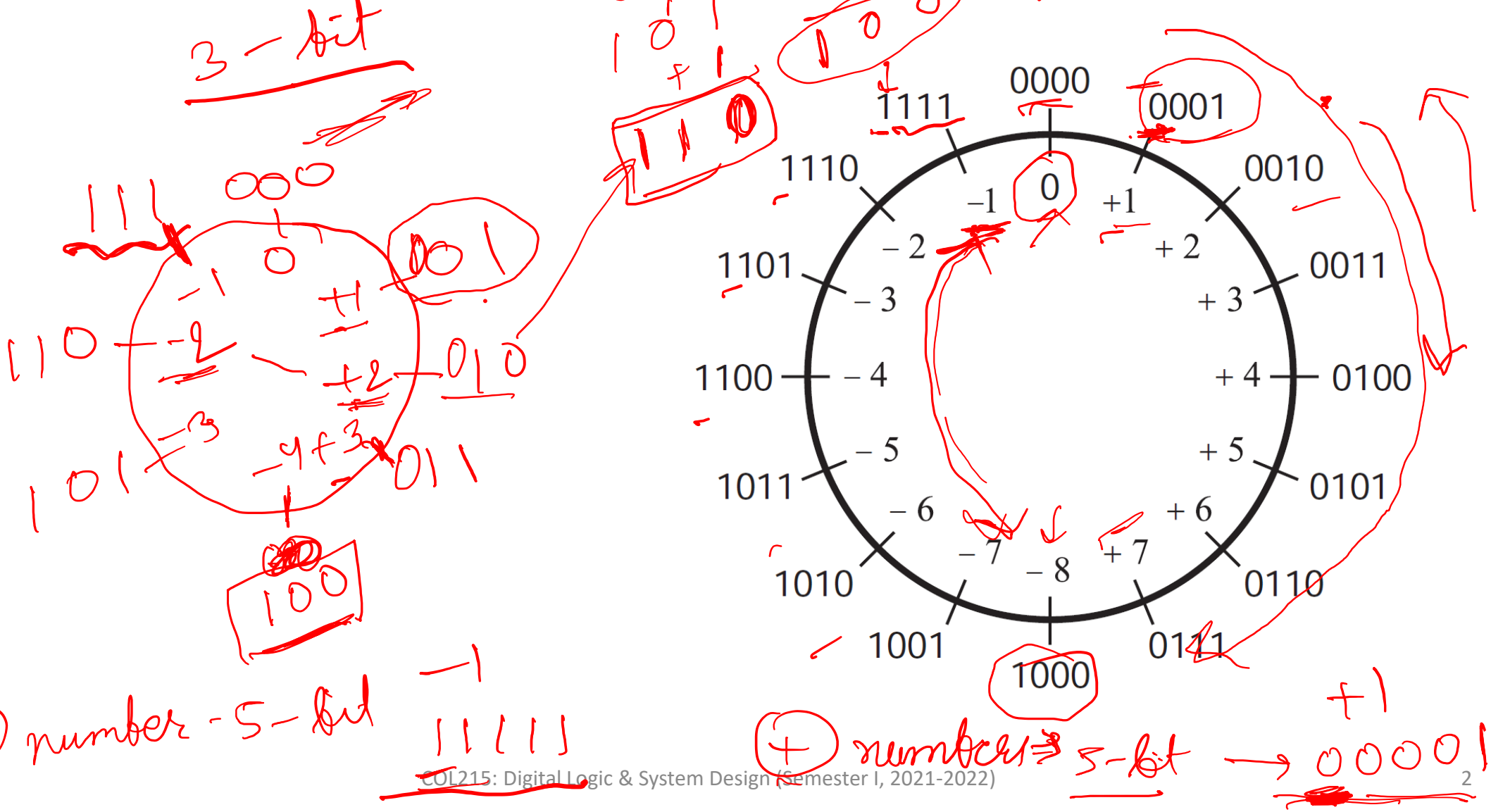
Lecture 14: Binary Arithmetic (Cont.)

M. Balakrishnan
CSE@IITD

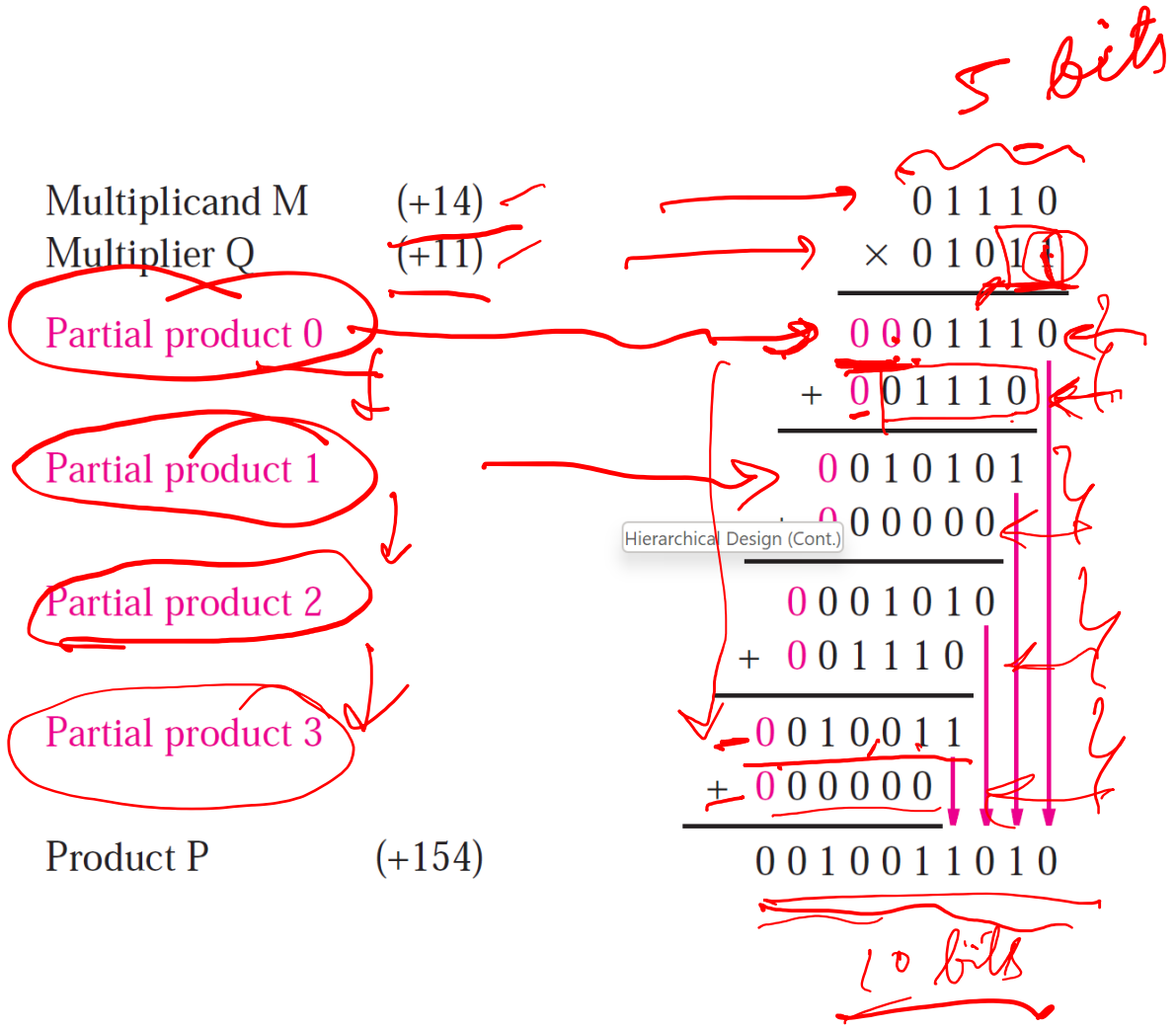
Vireshwar Kumar
CSE@IITD

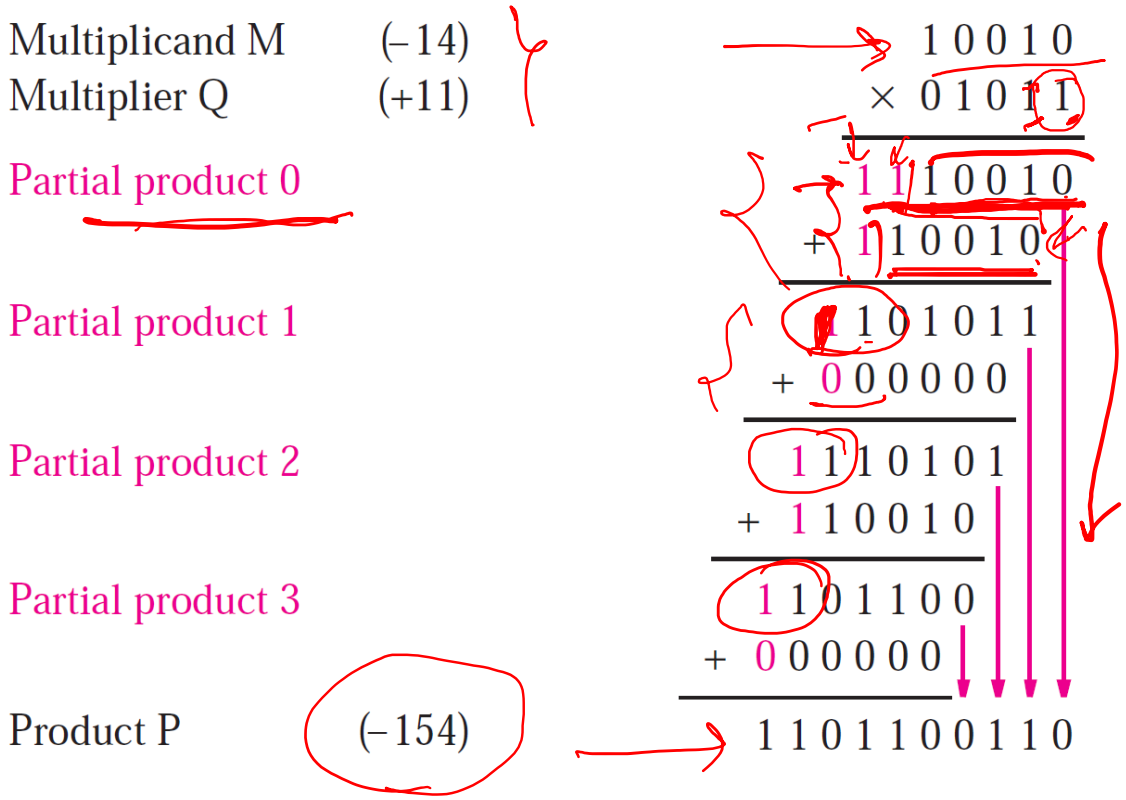
September 8, 2021

Sign Extension

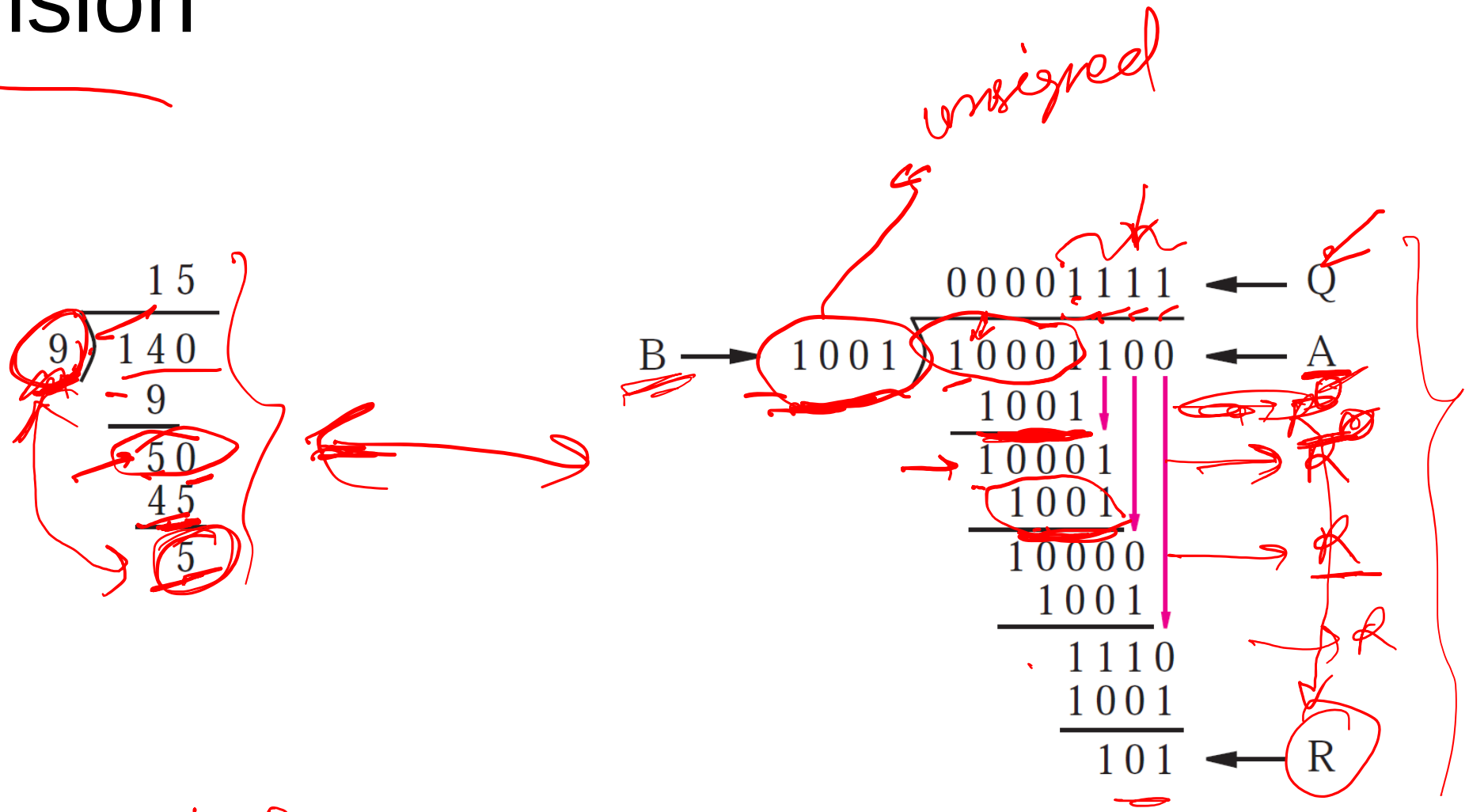


Multiplication of Signed Integers



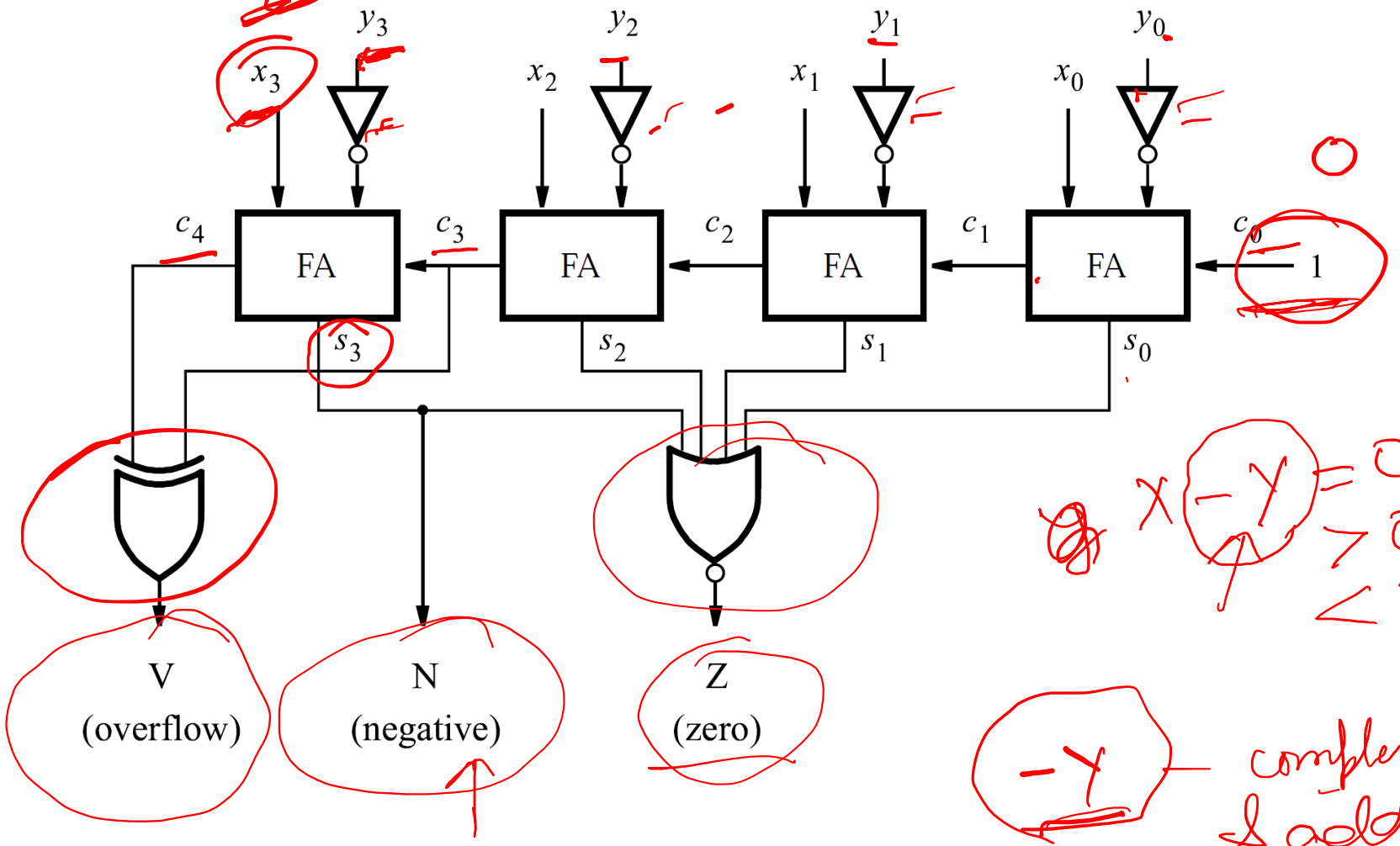


Division



100
-4 ÷ 2 = 110
-2

Comparator


y_3
y_2
y_1
y_0
x_3
x_2
x_1
x_0
c_4
c_3
c_2
c_1
c_0
FA
FA
FA
FA
1
s_3
s_2
s_1
s_0
V
(overflow)
N
(negative)
Z
(zero)

COL215L: Digital Logic & System Design

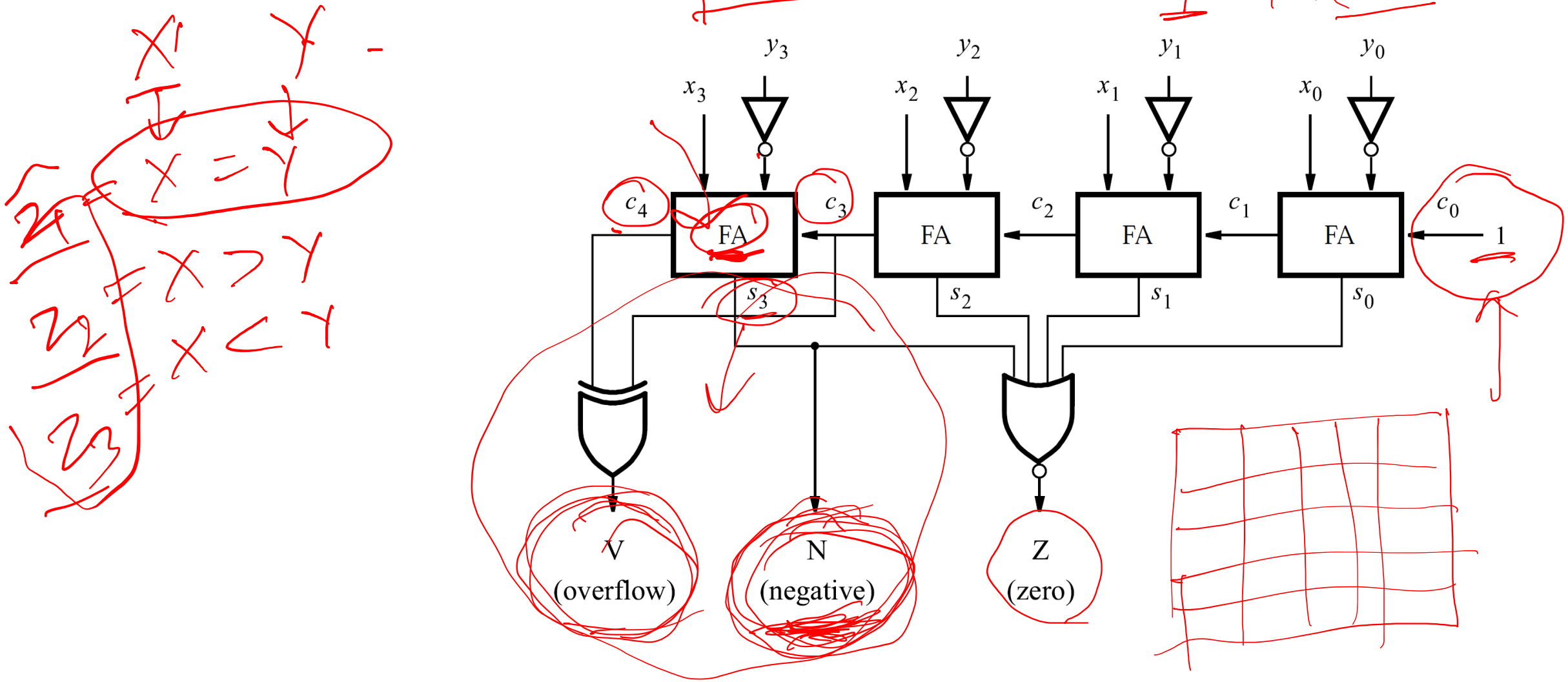
Lecture 15: Binary Arithmetic (Cont.)

M. Balakrishnan
CSE@IITD

Vireshwar Kumar
CSE@IITD

September 10, 2021

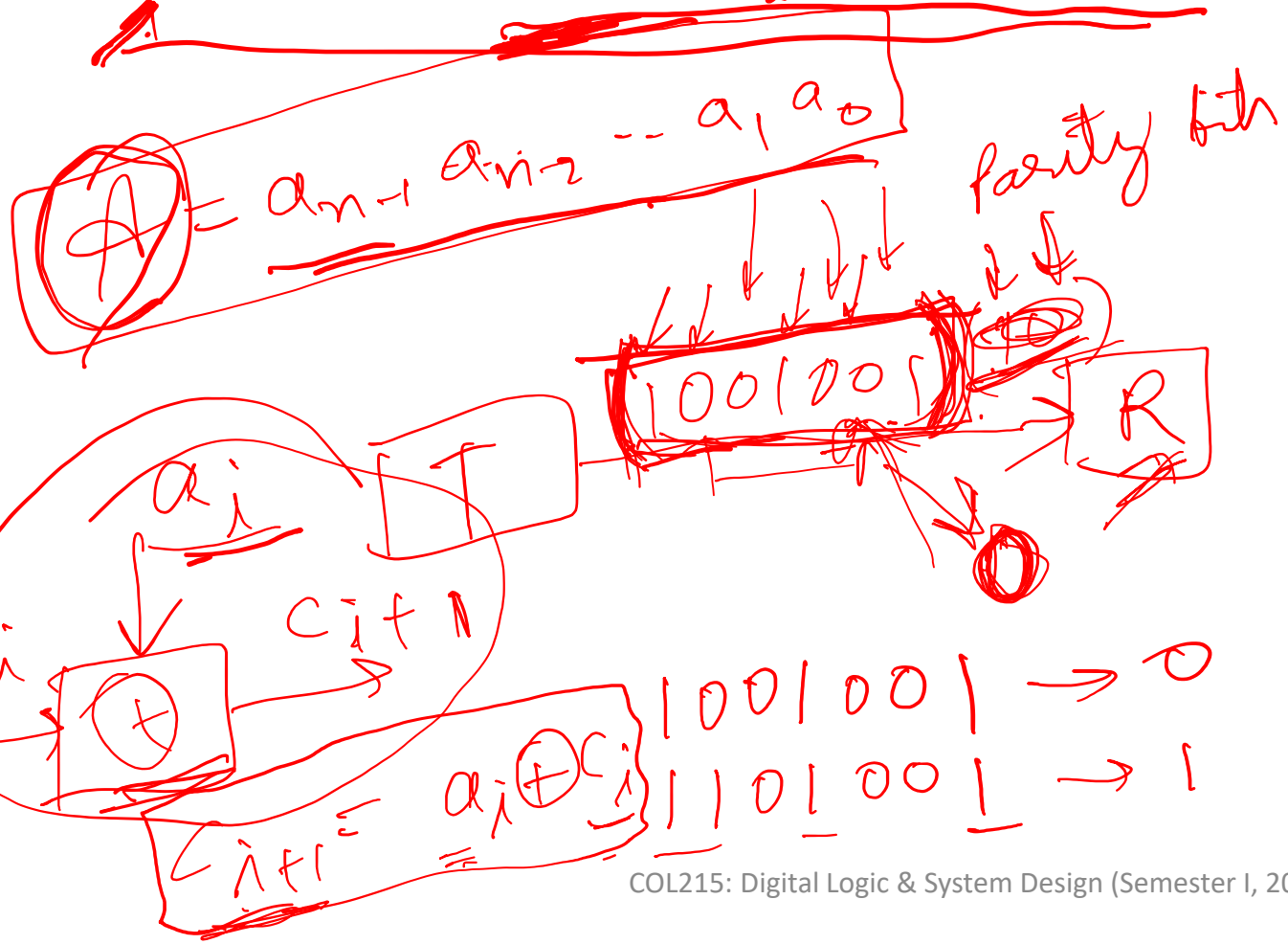
Comparator


y_3
y_2
y_1
y_0
x_3
x_2
x_1
x_0
c_4
c_3
c_2
c_1
c_0
FA
FA
FA
FA
1
s_3
s_2
s_1
s_0
V
(overflow)
N
(negative)
Z
(zero)

Parity Generation

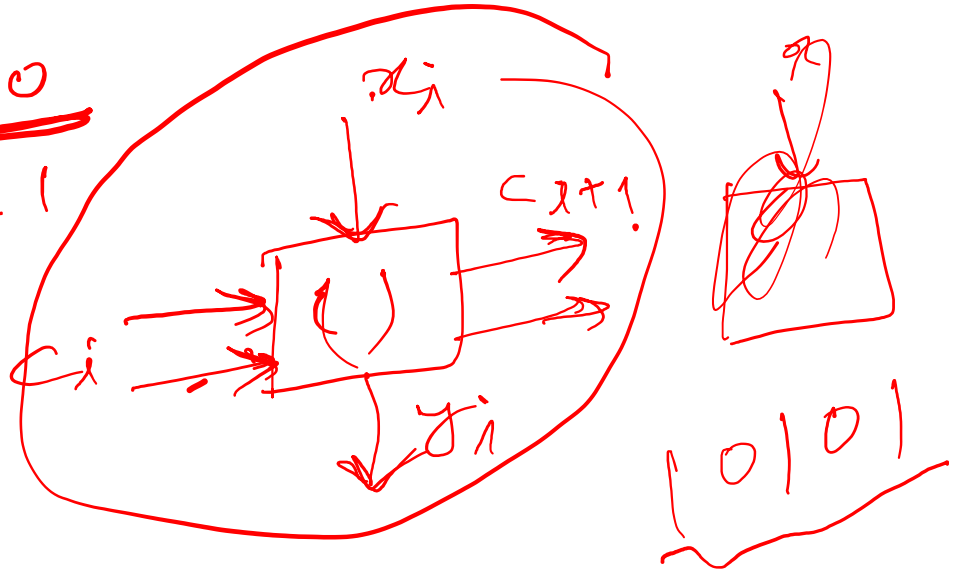
- Even parity bit
  - 1 if number of 1's in the data bits is even

c_i	a_i	c_{i+1}
0	0	0
0	1	1
1	0	1
1	1	0



Pattern Recognition

- Example
  - Detect 1011
- Basic Method
  - Input: 101101101010
  - Output: 00010010000
- Another Method (Define c_i)
  - 00 (if no bit match till now)
  - 01 (if one-bit match till now)
  - 10 (if two-bit match till now)
  - 11 (if three-bit match till now)



c_i	x_i	c_{i+1}	y_i
00	0	00	0
00	1	01	0
01	0	10	0
01	1	01	0
10	0	00	0
10	1	11	0
11	0	10	0
11	1	01	1

COL215L: Digital Logic & System Design

Lecture 16: Sequential Circuits

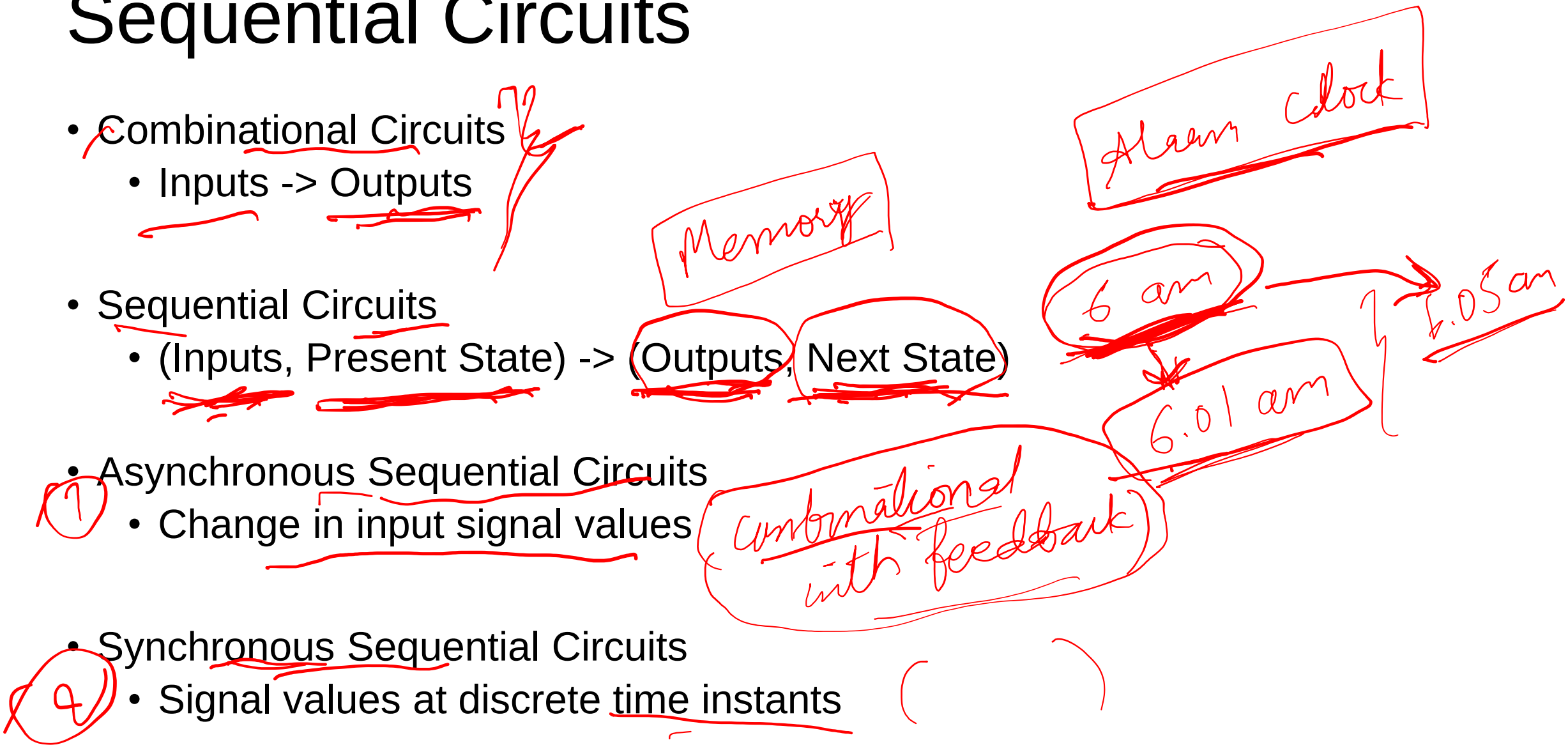
M. Balakrishnan
CSE@IITD

Vireshwar Kumar
CSE@IITD

September 14, 2021

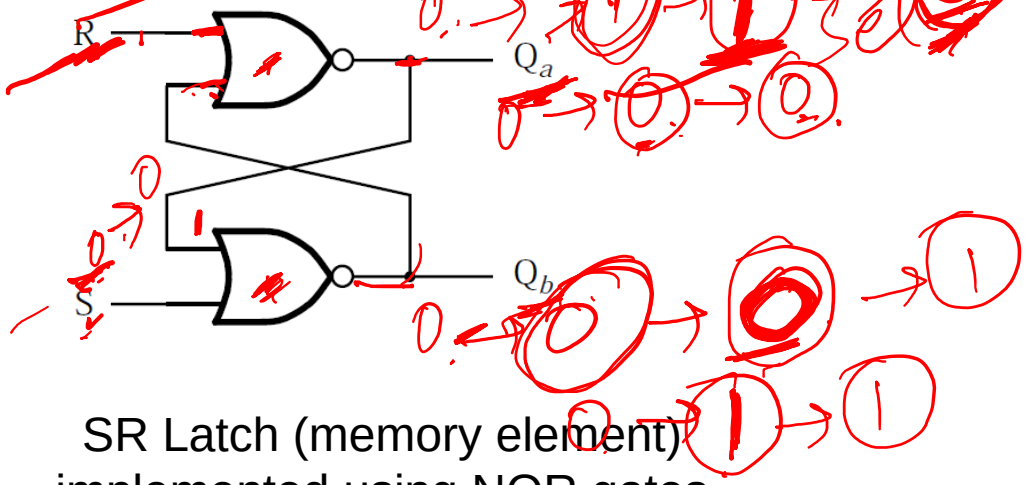
Sequential Circuits

- Combinational Circuits
  - Inputs -> Outputs
- Sequential Circuits
  - (Inputs, Present State) -> (Outputs, Next State)
- Asynchronous Sequential Circuits
  - Change in input signal values
- Synchronous Sequential Circuits
  - Signal values at discrete time instants

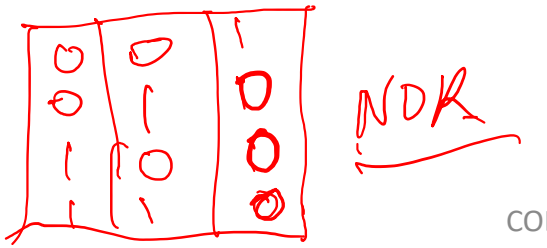


Storing State – SR Latch

- Example
  - Alarm clock



SR Latch (memory element) implemented using NOR gates

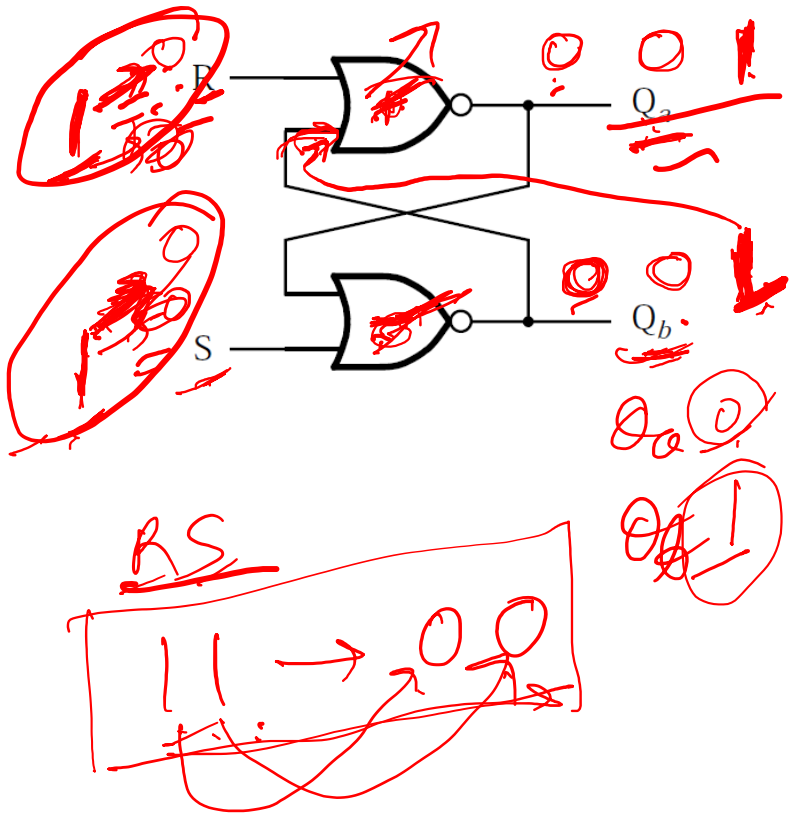


S	R	Q_a	Q_b	
0	0	0/1	1/0	(no change)
0	1	0	1	
1	0	1	0	
1	1	0	0	

Characteristic table

Limitation of a Basic SR Latch

- Oscillation



S	R	Q_a	Q_b	
0	0	0/1	1/0	(no change)
0	1	0	1	
1	0	1	0	
1	1	0	0	

COL215L: Digital Logic & System Design

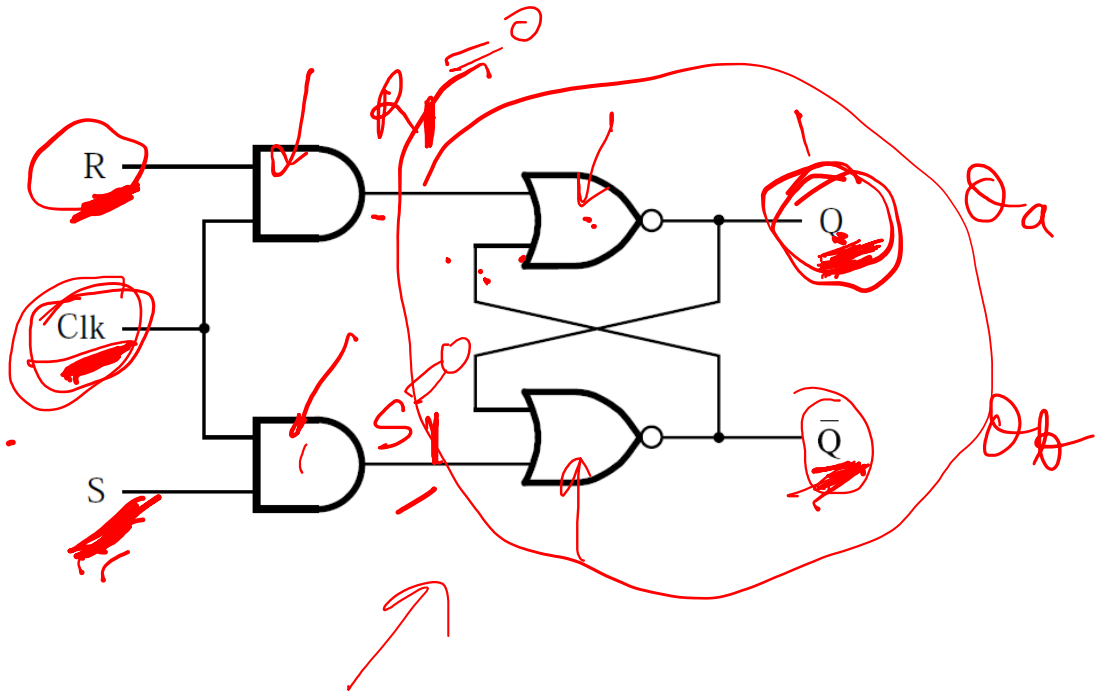
Lecture 17: Sequential Circuits (Cont.)

M. Balakrishnan
CSE@IITD

Vireshwar Kumar
CSE@IITD

September 15, 2021

Gated SR Latch



Clk	S	R	Q(t + 1)
0	x	x	Q(t) (no change)
1	0	0	Q(t) (no change)
1	0	1	0
1	1	0	1
1	1	1	x

- Set (Q = 1) and reset (Q = 0)

$\bar{Q}$ complement

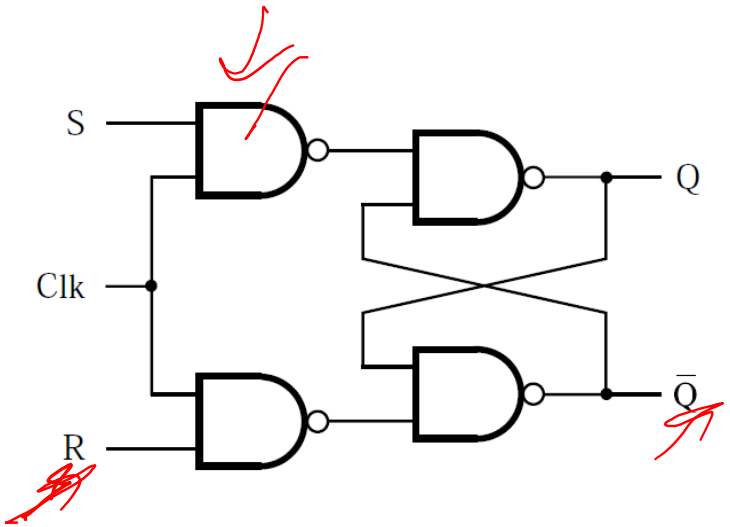
$$Q^{t+1} = \overline{(Clk \cdot R) + \bar{Q}^{t}}$$

$$= \overline{Clk \cdot R} \cdot Q^{t}$$

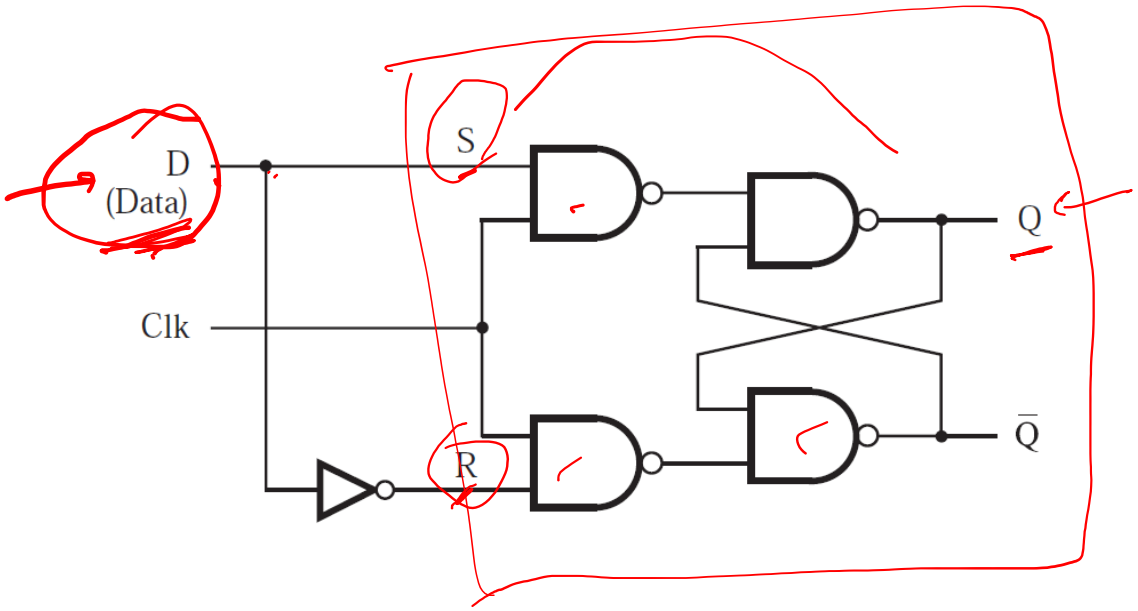
$$\Rightarrow \bar{Q}^{t+1} = \overline{(Clk \cdot R)} \cdot Q^{t}$$

Gated SR Latch with NAND

- Exercise
  - Write the characteristic table

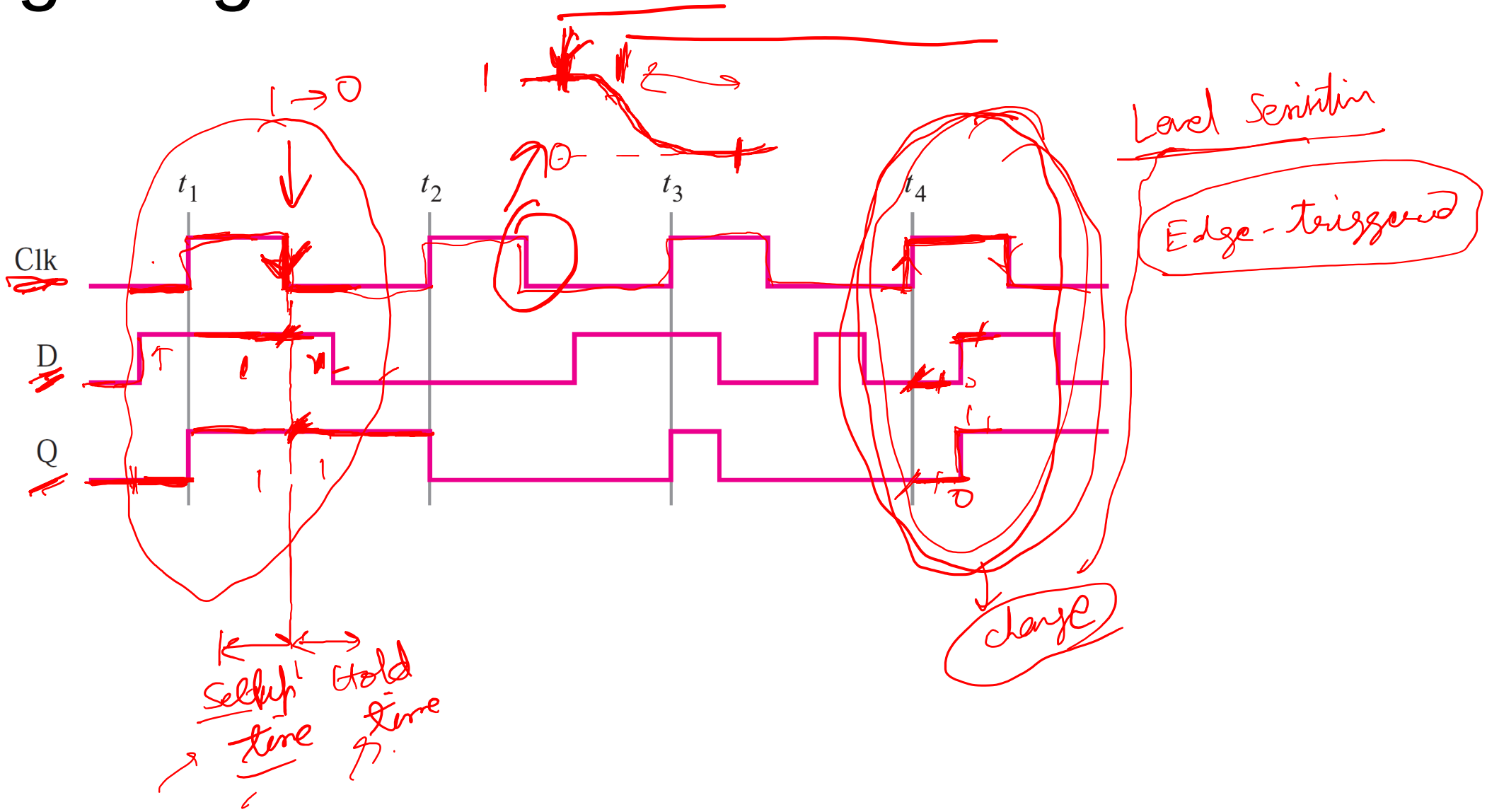


Gated D Latch

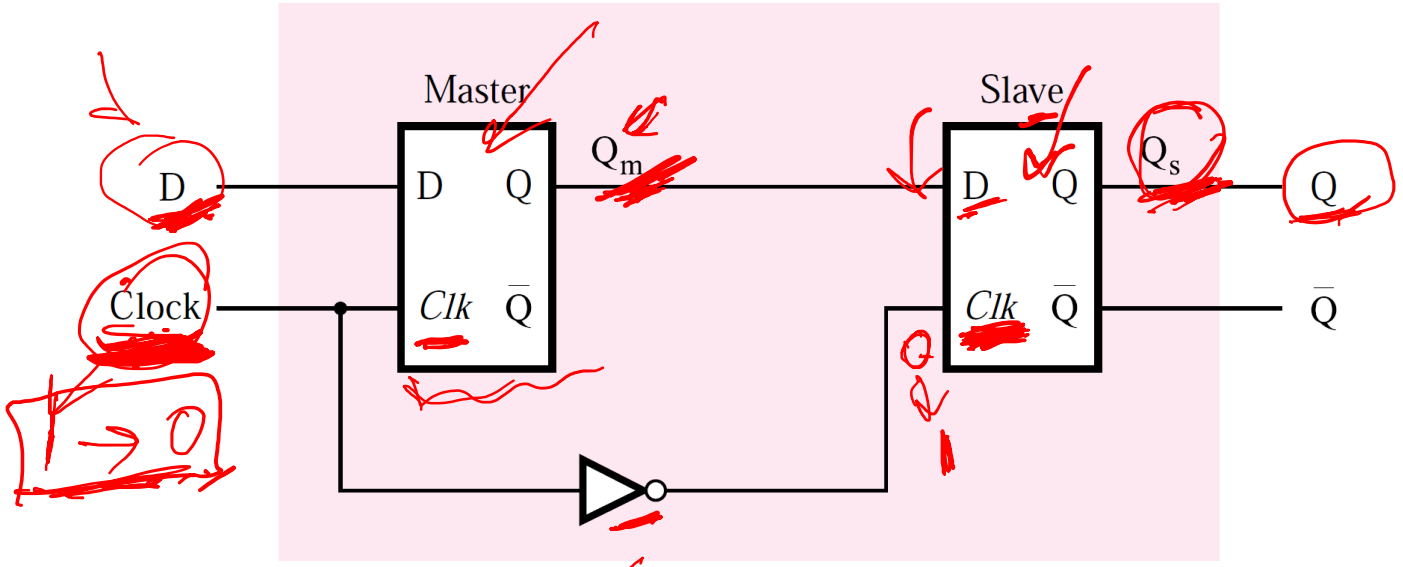


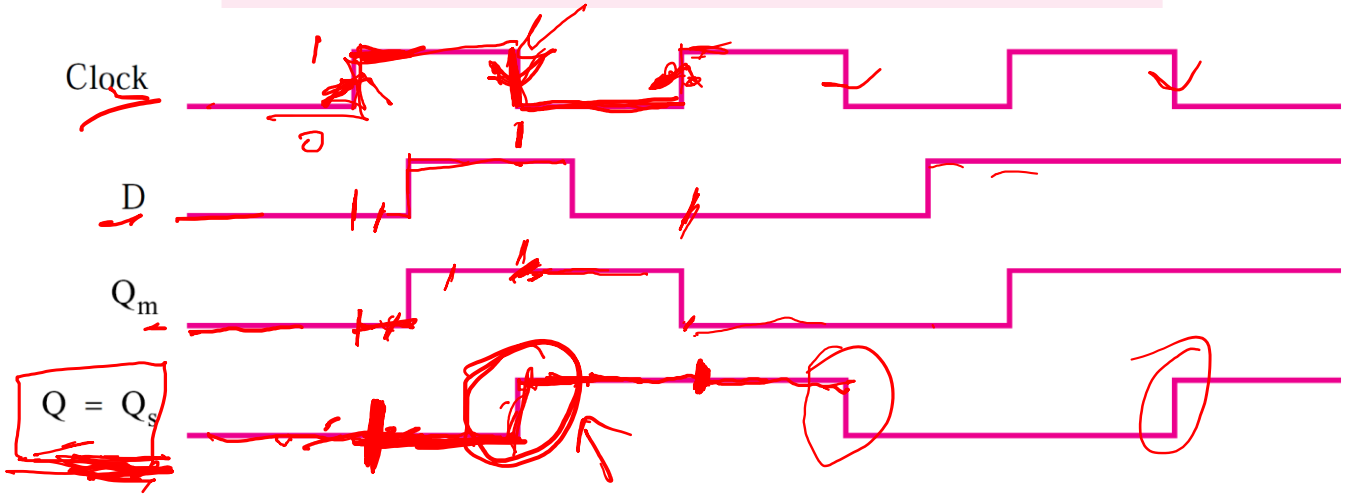
Clk	D	$Q(t+1)$
0	x	$Q(t)$
1	0	0
1	1	1

Timing Diagram of Gated D Latch



Master-Slave D Flip-Flop





edge-trigger

COL215L: Digital Logic & System Design

Lecture 18: Sequential Circuits (Cont.)

M. Balakrishnan
CSE@IITD

Vireshwar Kumar
CSE@IITD

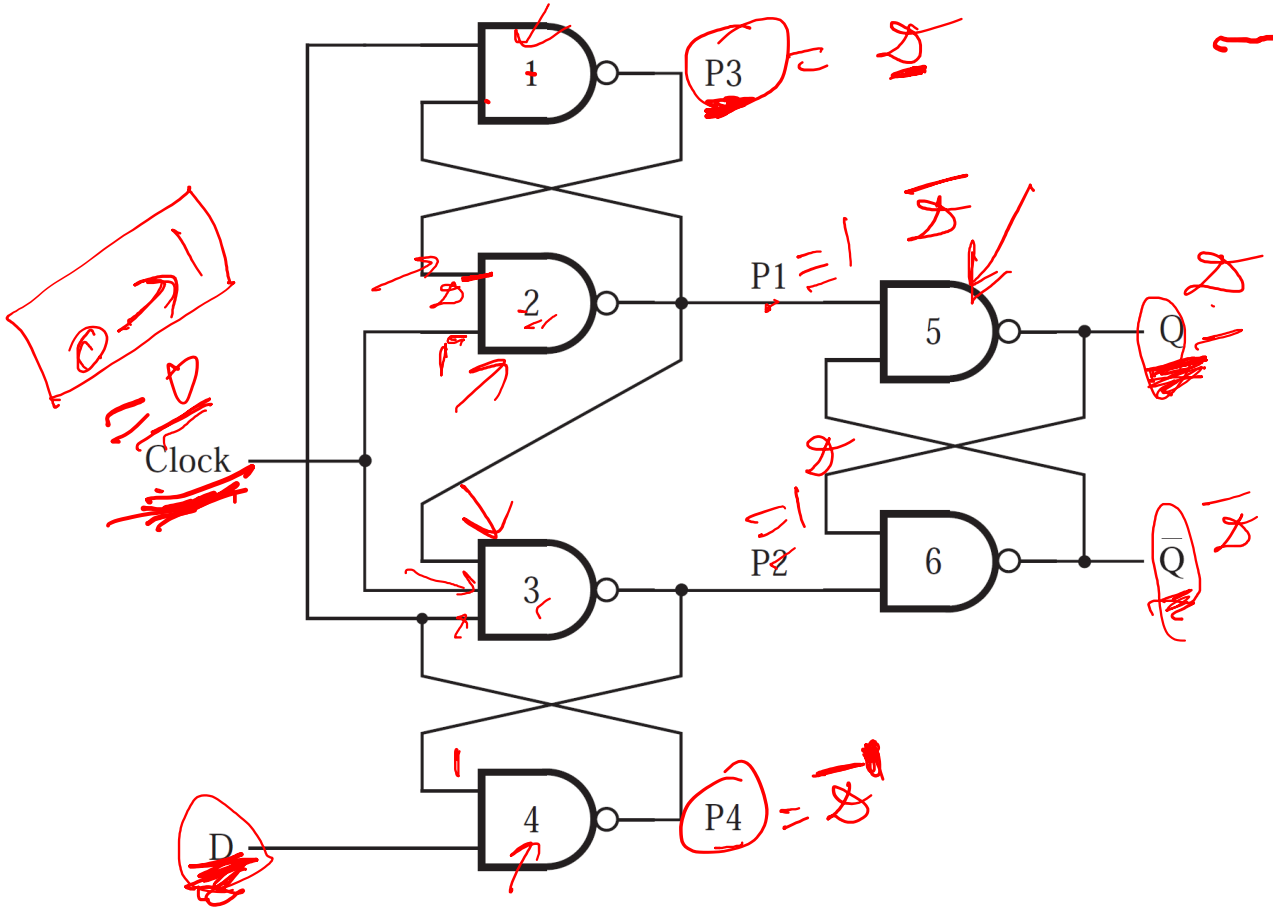
September 17, 2021

Sequential Circuits (Revision)

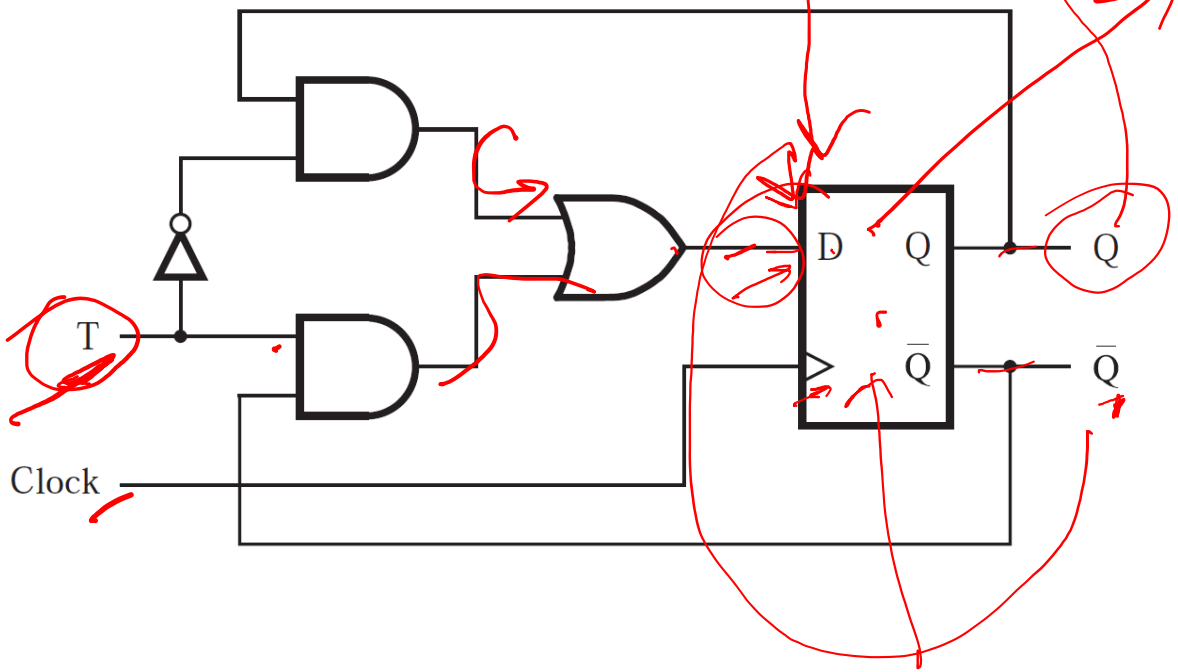
- Level Sensitive
  - SR Latch
  - Gated SR Latch
  - Gated D Flip-Flop

- Edge-Triggered
  - Master-Slave D Flip-Flop
    - Clock
    - Clock’

Edge-Triggered D Flip-Flop



T Flip-Flop

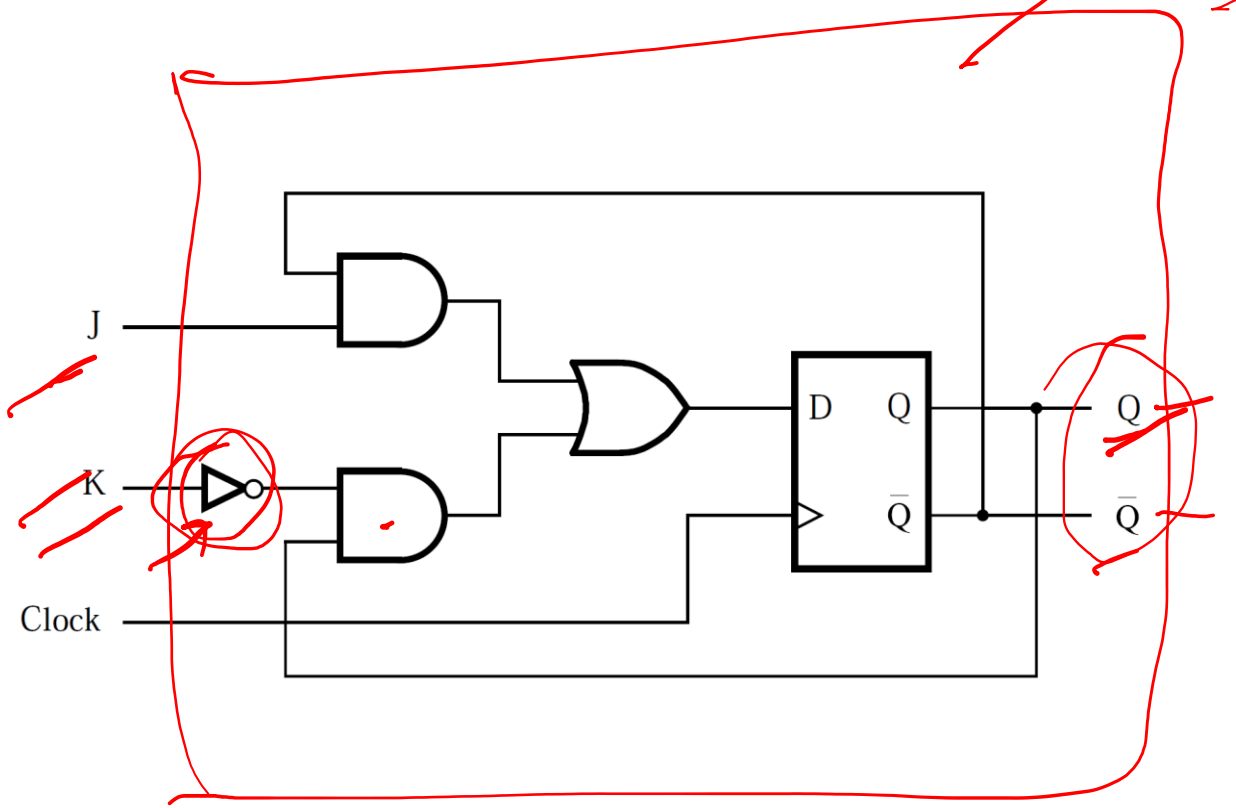


T	$Q(t+1)$
0	$Q(t)$
1	$\bar{Q}(t)$

$$Q(t+1) = T \cdot \bar{Q}(t) + \bar{T} \cdot Q(t)$$

$Q(t) = 0$ -(1)

JK Flip-Flop

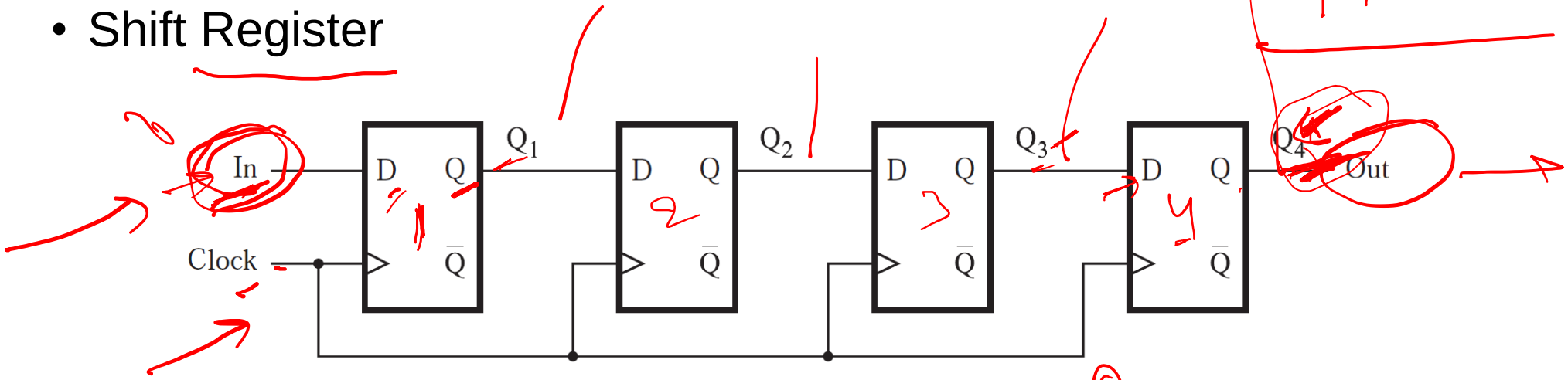


J	K	Q(t + 1)
0	0	Q(t)
0	1	0
1	0	1
1	1	$\bar{Q}(t)$

$$Q_{t+1} = J\bar{Q}_t + \bar{K}Q_t$$

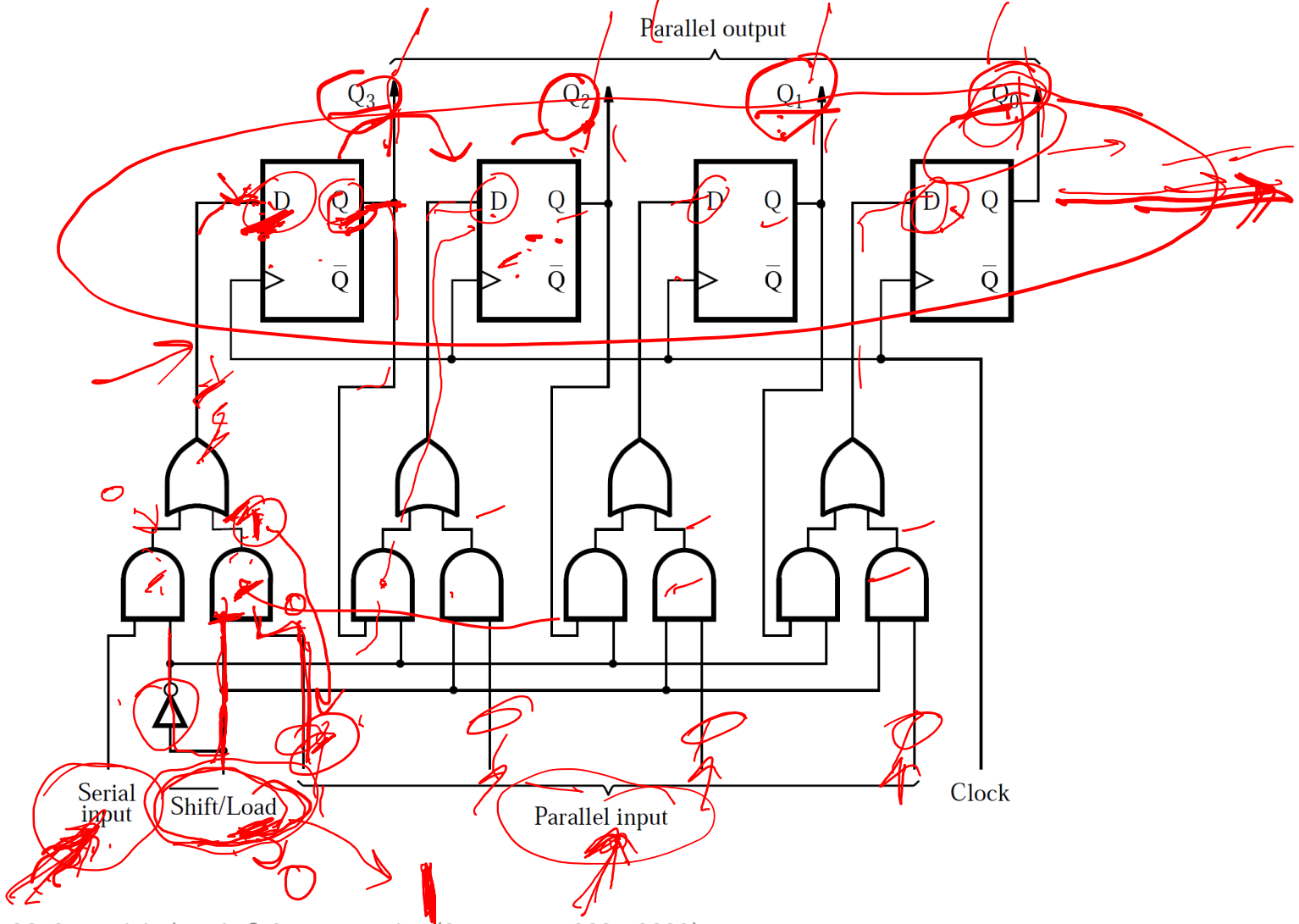
Register

- Shift Register



Parallel-Access Shift Register

- Serial-Serial
- Parallel-Parallel
- Serial-Parallel
- Parallel-Serial



COL215L: Digital Logic & System Design

Lecture 20: Finite State Machines

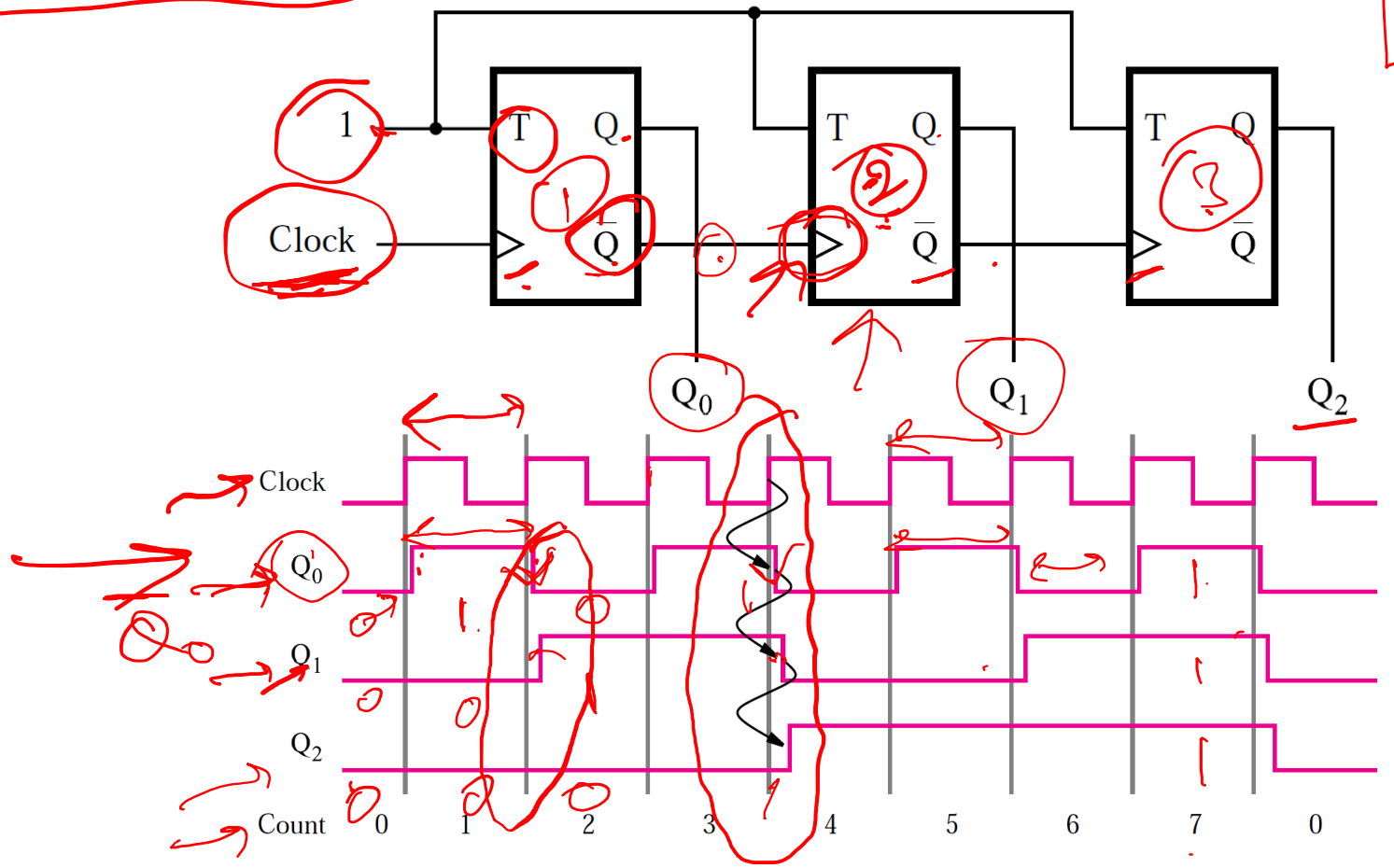
M. Balakrishnan
CSE@IITD

Vireshwar Kumar
CSE@IITD

September 28, 2021

Asynchronous Counter

- Up-Counter with T Flip-Flops



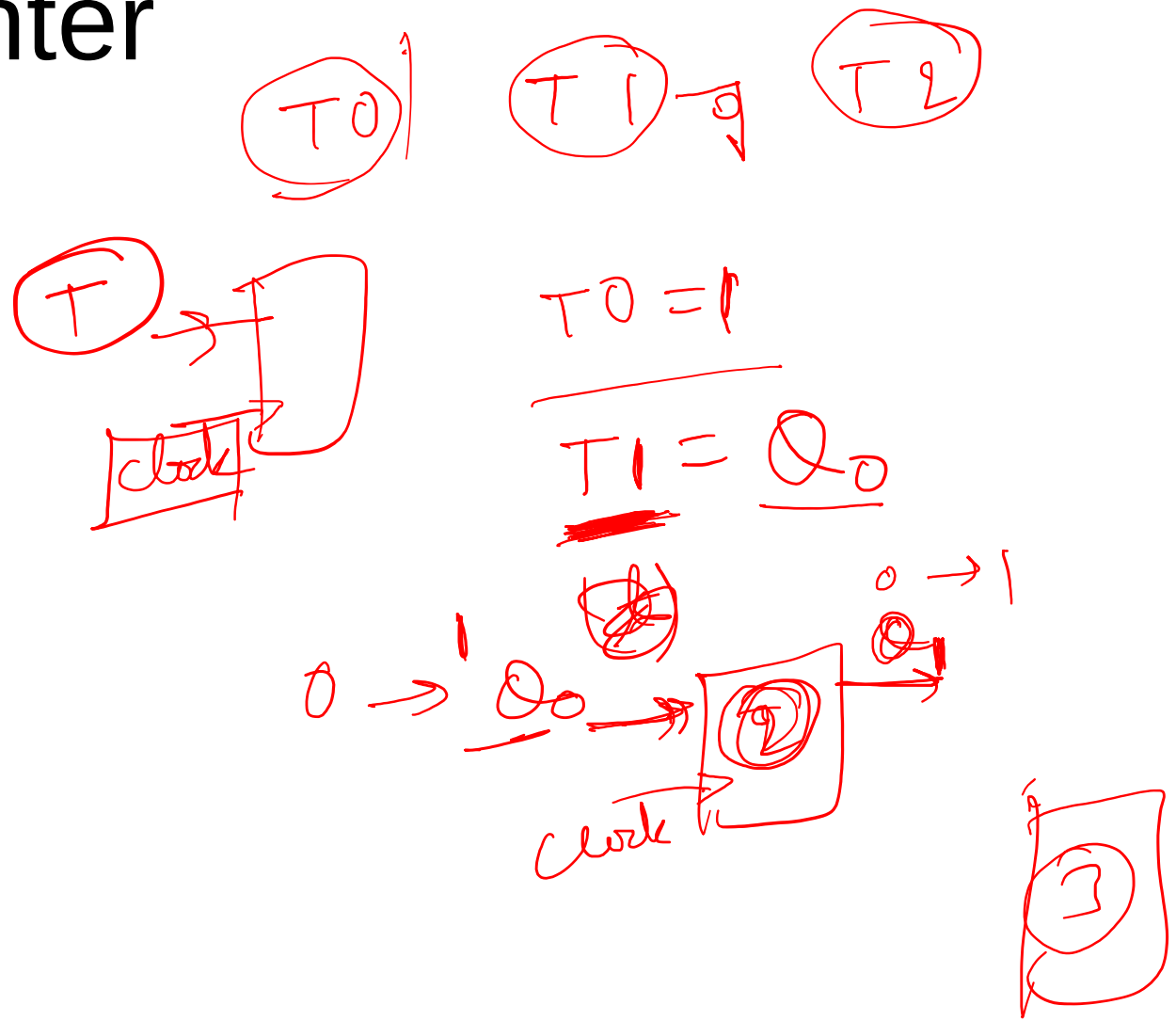
T	Q(t + 1)
0	Q(t)
1	$\overline{Q}(t)$

Synchronous Counter

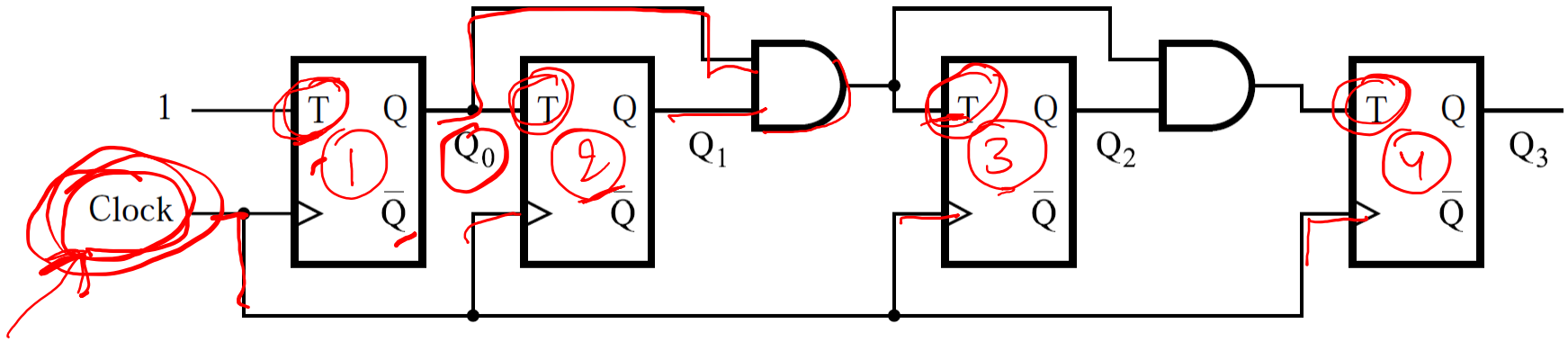
Clock cycle	Q_2 Q_1 Q_0
0	0 0 0
1	0 0 1
2	0 1 0
3	0 1 1
4	1 0 0
5	1 0 1
6	1 1 0
7	1 1 1
8	0 0 0

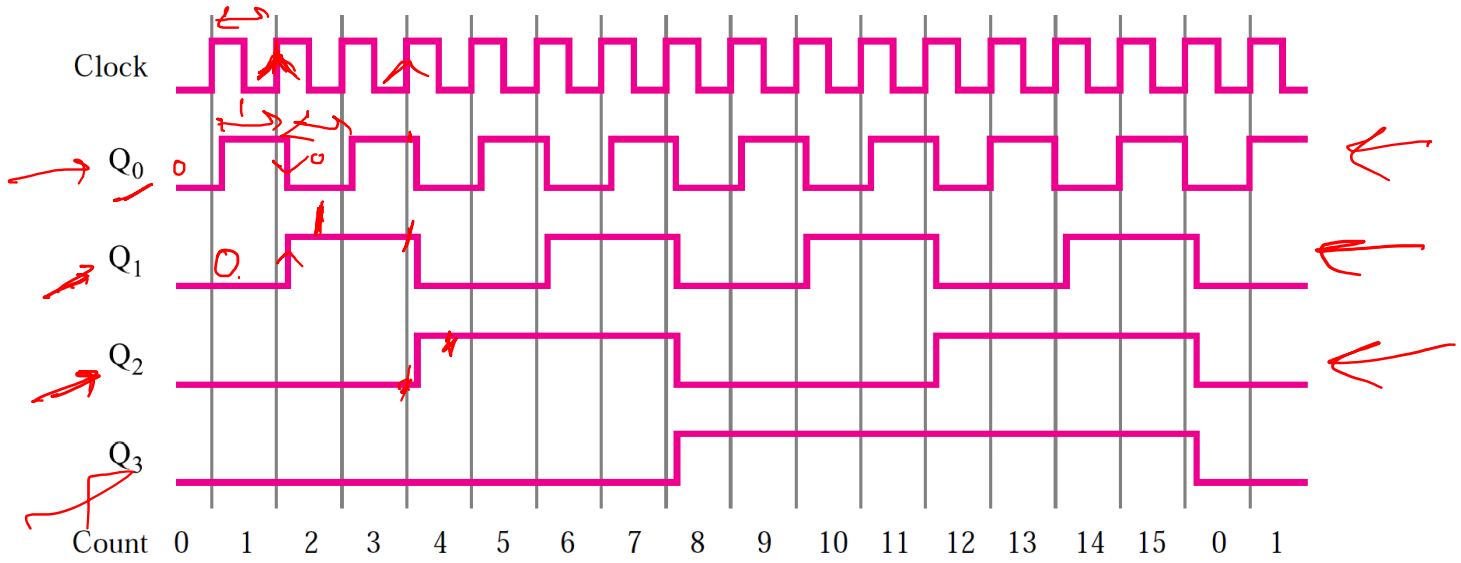
Q_1 changes

Q_2 changes

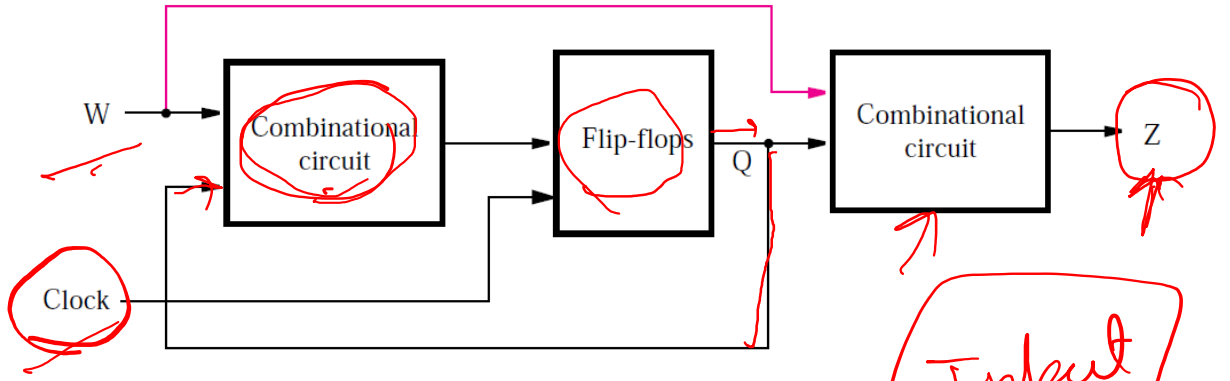


Synchronous Counter (Cont.)

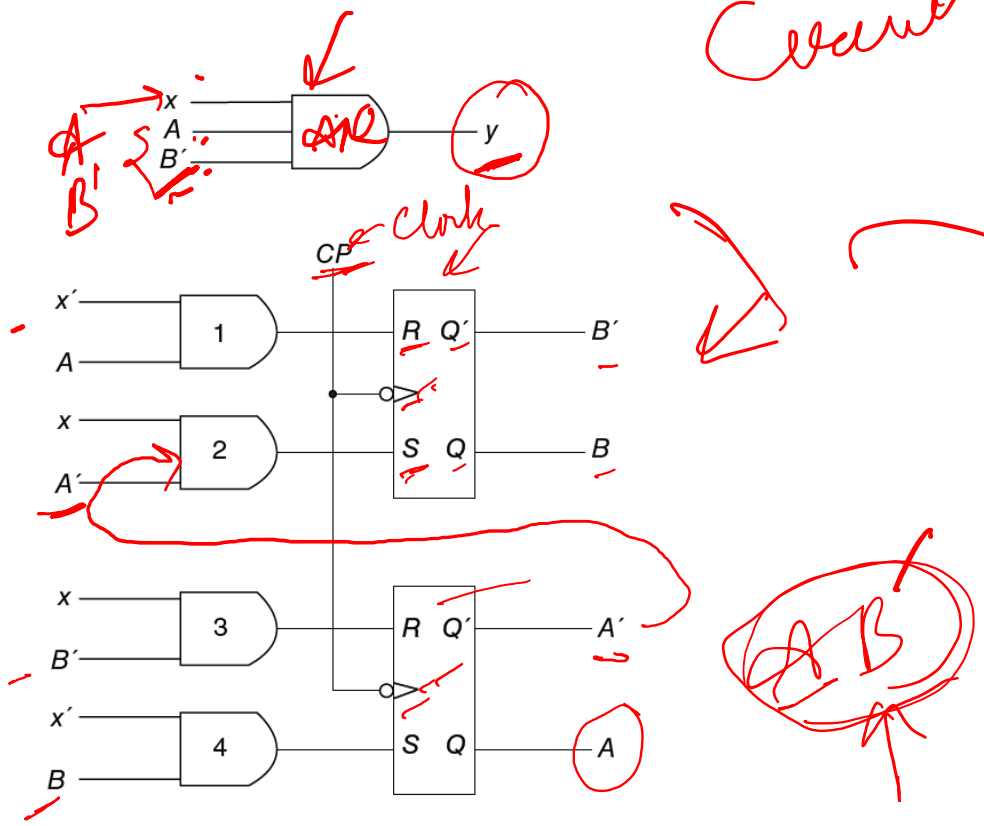




Sequential Circuits (Revisited)



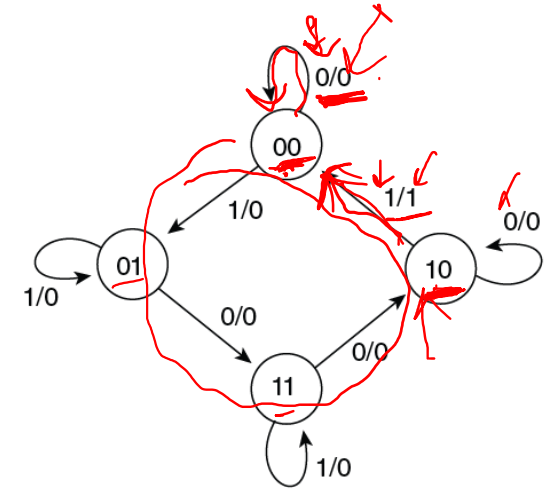
Example



Present State	Next state		Output	
	$x = 0$	$x = 1$	$x = 0$	$x = 1$
AB	AB	AB	y	y
00	00	01	0	0
01	11	01	0	0
10	10	00	0	1
11	10	11	0	0

State Table and Diagram

Present State	Next state		Output	
	$x = 0$	$x = 1$	$x = 0$	$x = 1$
AB	AB	AB	y	y
00	00	01	0	0
01	11	01	0	0
10	10	00	0	1
11	10	11	0	0



Table

Diagram

$y = x \cdot A \cdot B'$

COL215L: Digital Logic & System Design

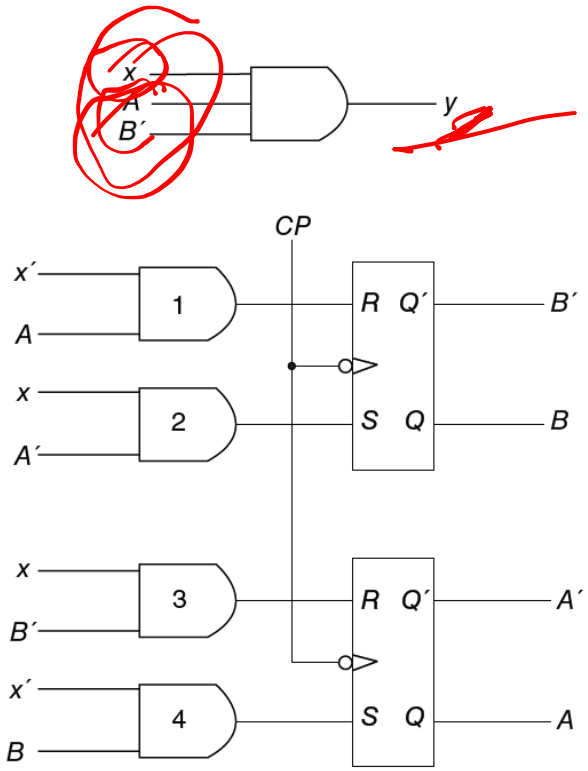
Lecture 21: Finite State Machines (Cont.)

M. Balakrishnan
CSE@IITD

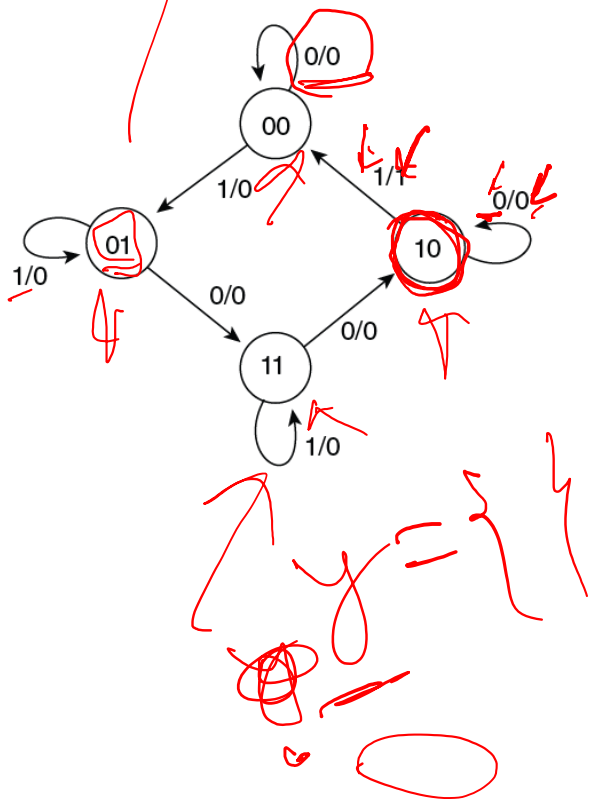
Vireshwar Kumar
CSE@IITD

September 29, 2021

Circuit, State Table and State Diagram

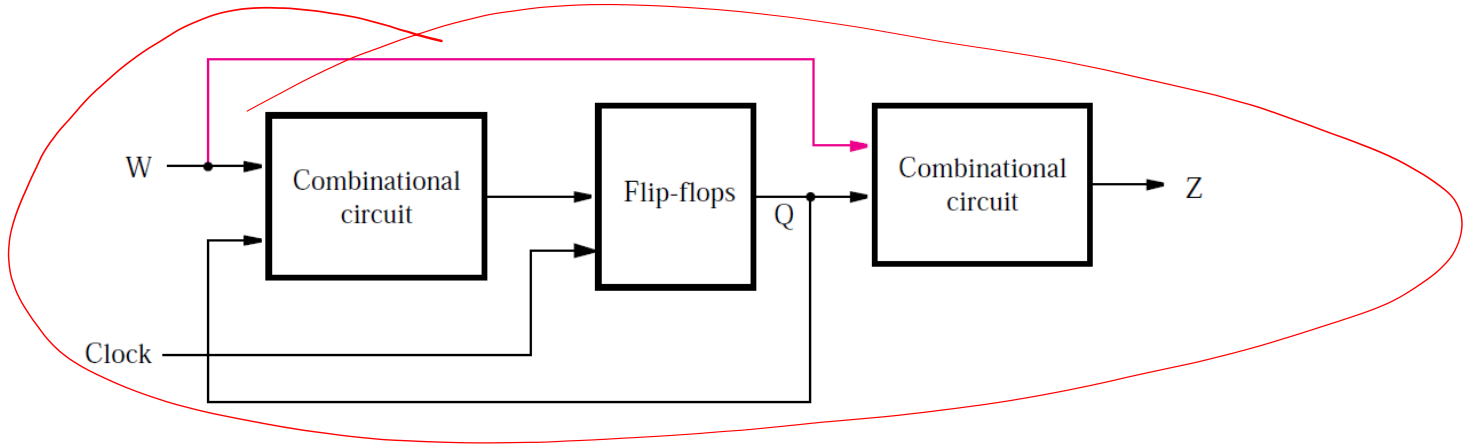


Present State	Next state		Output	
	$x = 0$	$x = 1$	$x = 0$	$x = 1$
AB	AB	AB	y	y
00	00	01	0	0
01	11	01	0	0
10	10	00	0	1
11	10	11	0	0



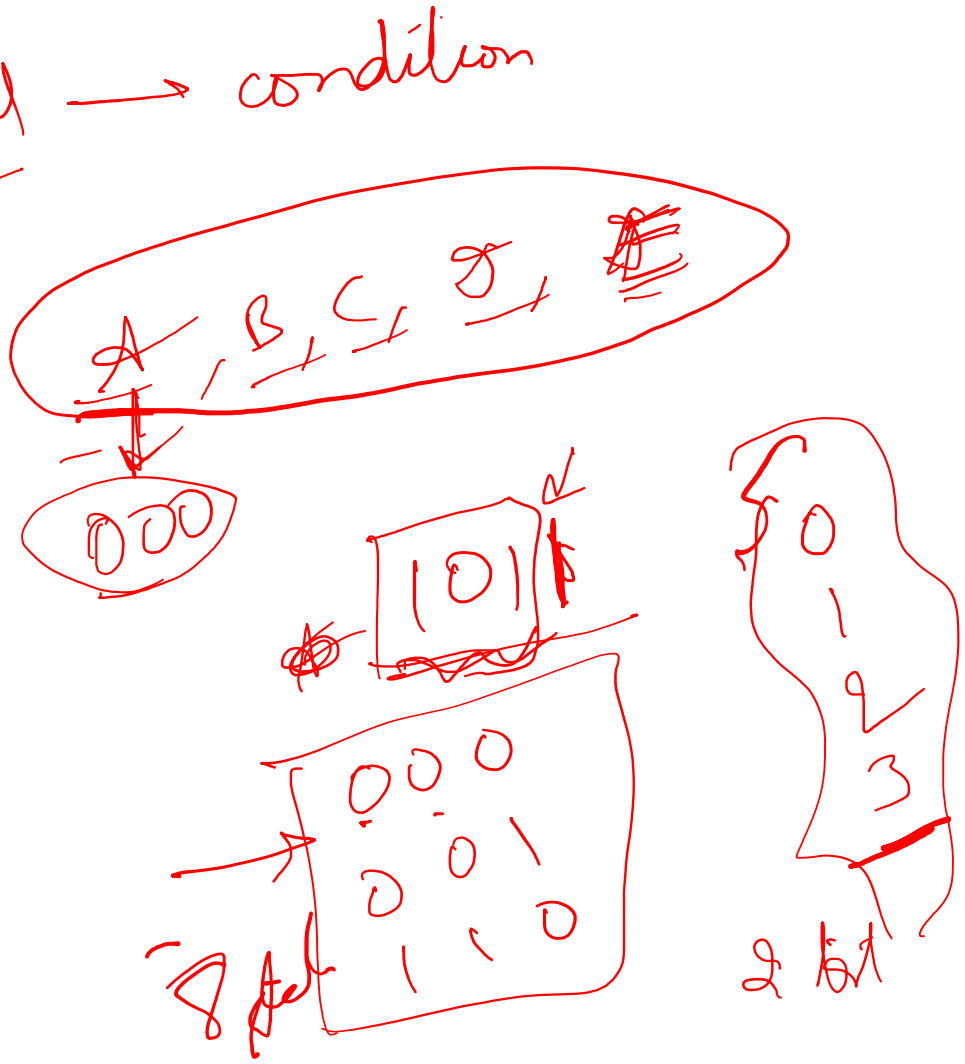
Finite State Machine

- Circuits that can be represented with finite number of states
  - Output – Present State with/without Input
- Output
  - Present State - Moore type
  - Present State & Inputs – Mealy type



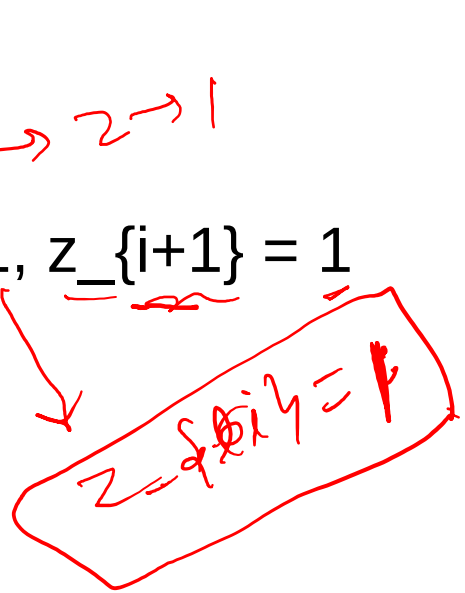
Design Steps

1. Specification
2. State Diagram
3. State Table
4. State Assignment (Minimize States)
5. Select Flip-Flop and Implement Circuit

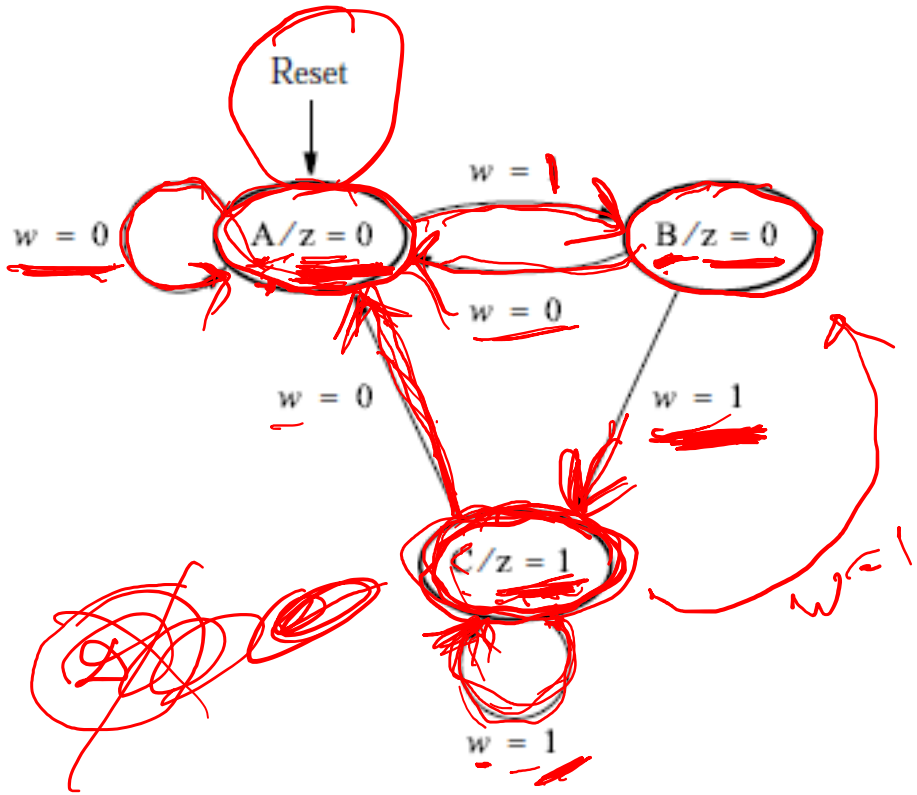


Step-1: Specification

- one input, w
- one output, z
- Recognize the pattern 11
  - If w_{i-1} = 1 and w_{i} = 1, z_{i+1} = 1
  - Otherwise, z_{i+1} = 0

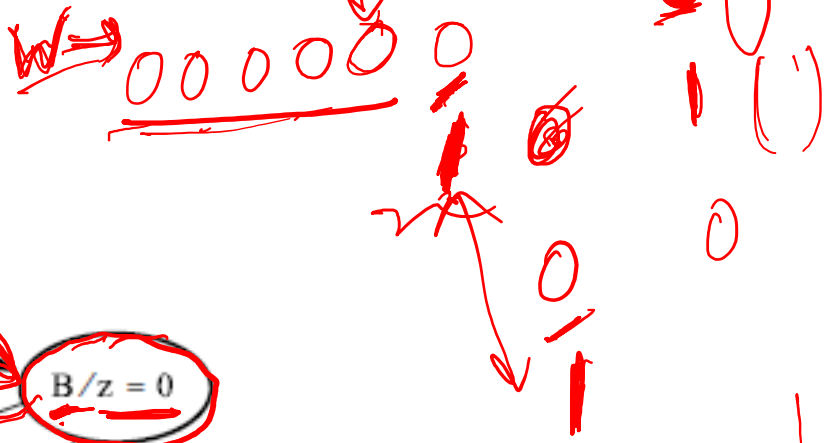


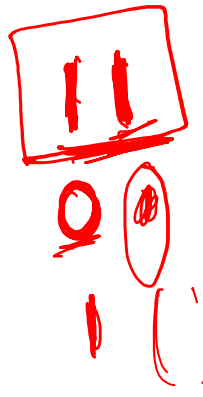
Step-2: State Diagram



Moore

Reset





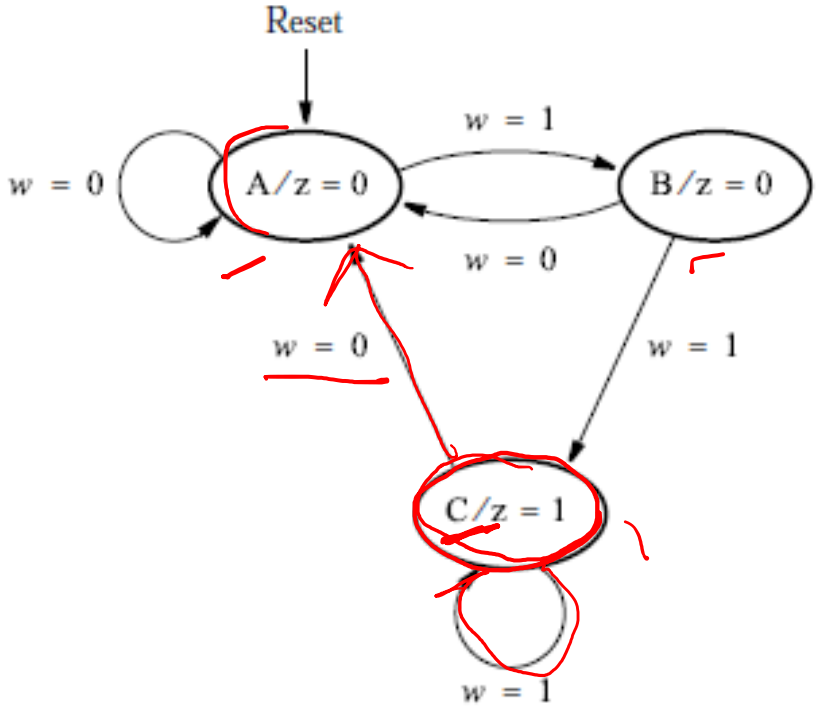
$w_{i-1} = 1$

$w_i = 1$

$z_{i+1} = 1$

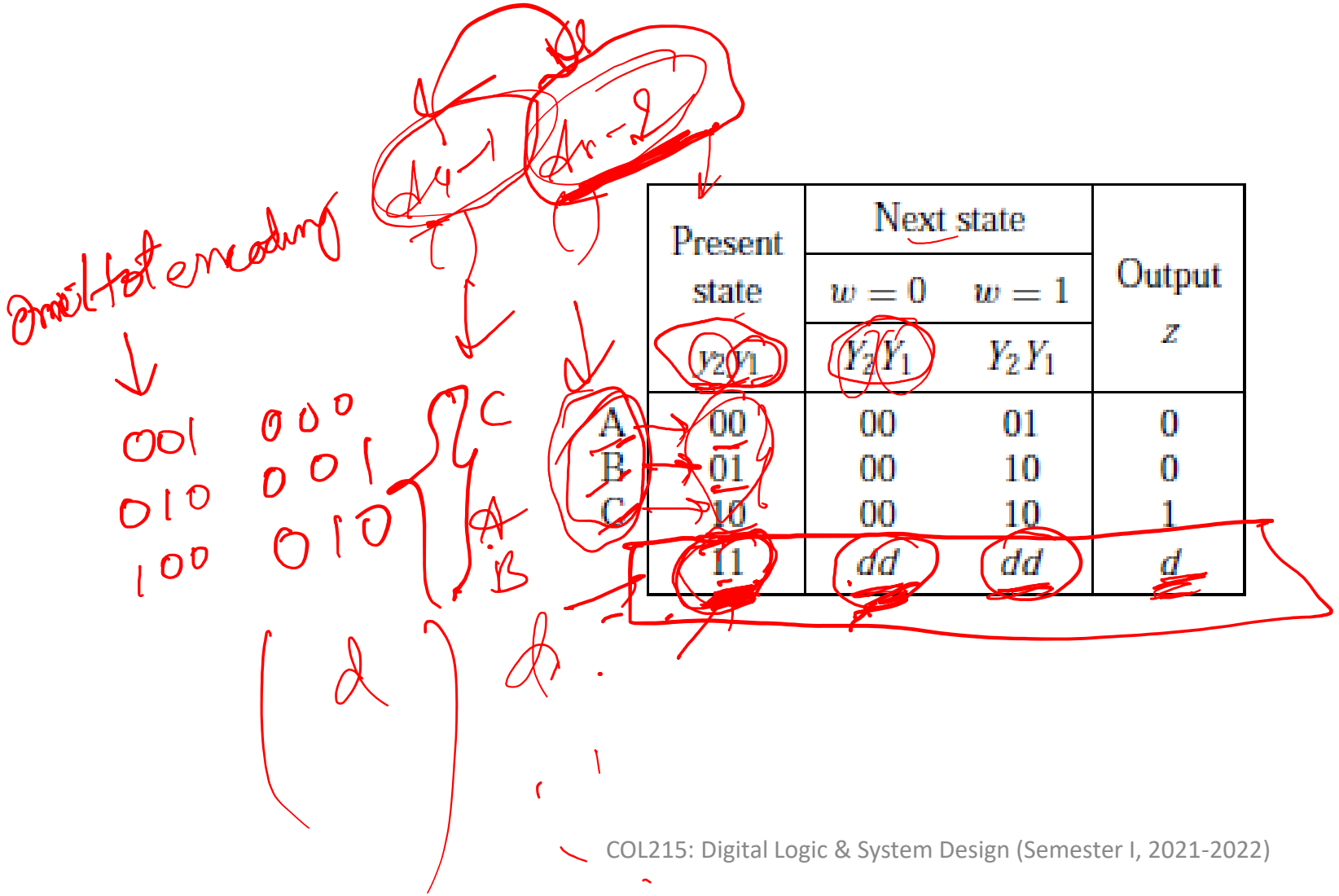
Output → Present state

Step-3: State Table

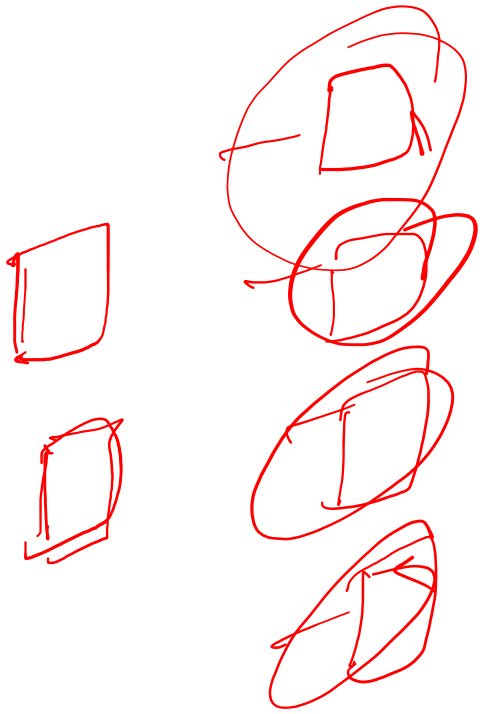


Present state	Next state		Output
	$w = 0$	$w = 1$	z
A	A	B	0
B	A	C	0
C	A	C	1

Step-4: State Assignment



Present state	Next state $w=0$	Next state $w=1$	Output z
y_2y_1	Y_2Y_1	Y_2Y_1	
A	00	01	0
B	01	10	0
C	10	00	1
11	dd	dd	d



Step-5: Circuits

Flip

$Y_2 = w(y_1 + y_2)$

$Y_1 = w\,y_2\,y_1$

$z = y_2$

$y_2\,y_1\,w$

	Present state	Next state		Output
		$w = 0$	$w = 1$	
	y_2y_1	Y_2Y_1	Y_2Y_1	z
A	00	00	01	0
B	01	00	10	0
C	10	00	10	1
	11	*dd*	*dd*	*d*

Y_1

w \ y_2y_1	00	01	11	10
0	0	0	d	0
1	1	0	d	0

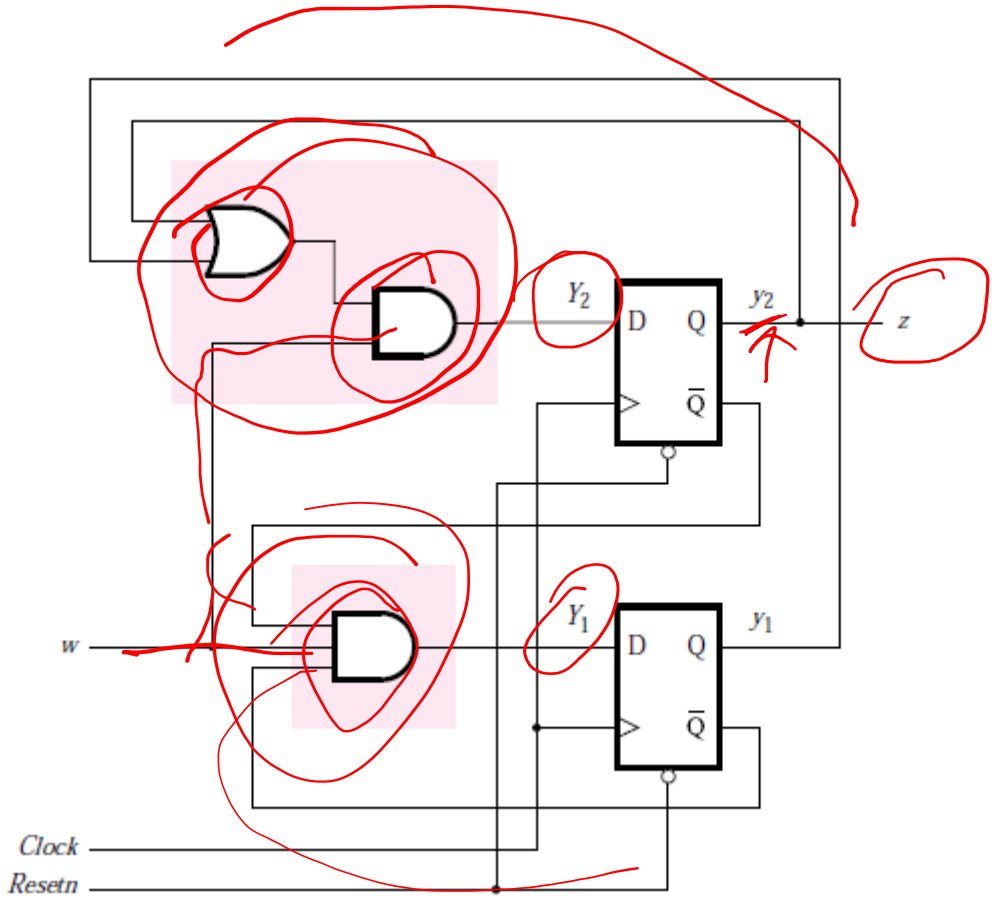
Y_2

w \ y_2y_1	00	01	11	10
0	0	0	d	0
1	0	1	d	1

z

y_2 \ y_1	0	1
0	0	0
1	1	d

Step-5: Circuits (Cont.)



COL215L: Digital Logic & System Design

Lecture 22: Finite State Machines (Cont.)

M. Balakrishnan
CSE@IITD

Vireshwar Kumar
CSE@IITD

October 1, 2021

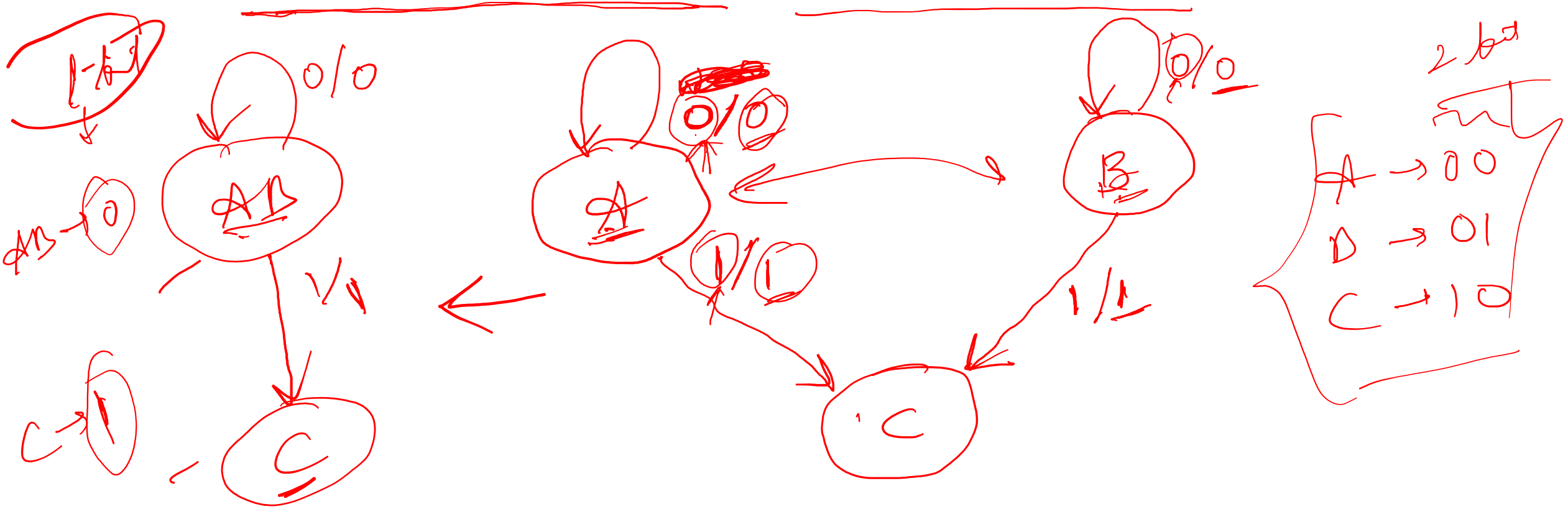
Handling Suspicious Cases in Minor

- There is a direct correlation between those who are suspicious and those who used only one device
  - Penalty - TBD

- Suspicious cases
  - Viva related to all topics discussed before Minor
  - Disproportionate performance in the Viva and the Minor Exam
    - Penalty – TBD

State Minimization

- State A – equivalent to – State B
  - Every possible input sequence – the same output sequence



Minimization Procedure - Example

- P1 = (ABCDEFG)
- P2 = (ABD)(CEFG)
- P3 = (ABD)(CEG)(F)
- P4 = (AD)(B)(CEG)(F)
- P5 = (AD)(B)(CEG)(F)

Present state	Next state $w=0$	Next state $w=1$	Output z
A	B	C	1
B	D	F	1
C	F	E	0
D	B	G	1
E	F	C	0
F	E	D	0
G	F	G	0

Minimized State

Present state	Next state $w=0$	Next state $w=1$	Output z
A	B	C	1
B	D	F	1
C	F	E	0
D	B	G	1
E	F	C	0
F	E	D	0
G	F	G	0

Present state	Next state $w=0$	Next state $w=1$	Output z
A	B	C	1
B	A	F	1
C	F	C	0
F	C	A	0

Incompletely Specified FSM

- z=0
  - P1 = (ABCDEFG)
  - P2 = (ABDG)(CEF)
  - P3 = (AB)(D)(G)(CE)(F)
  - P4 = (A)(B)(D)(G)(CE)(F)
  - P5 = (A)(B)(D)(G)(CE)(F)

- z=1
  - P1 = (ABCDEFG)
  - P2 = (AD)(BCEFG)
  - P3 = (AD)(B)(CEFG)
  - P4 = (AD)(B)(CEG)(F)
  - P5 = (AD)(B)(CEG)(F)

Present state	Next state		Output z	
	$w = 0$	$w = 1$	$w = 0$	$w = 1$
A	B	C	0	0
B	D	—	0	—
C	F	E	0	1
D	B	G	0	0
E	F	C	0	1
F	E	D	0	1
G	F	—	0	—