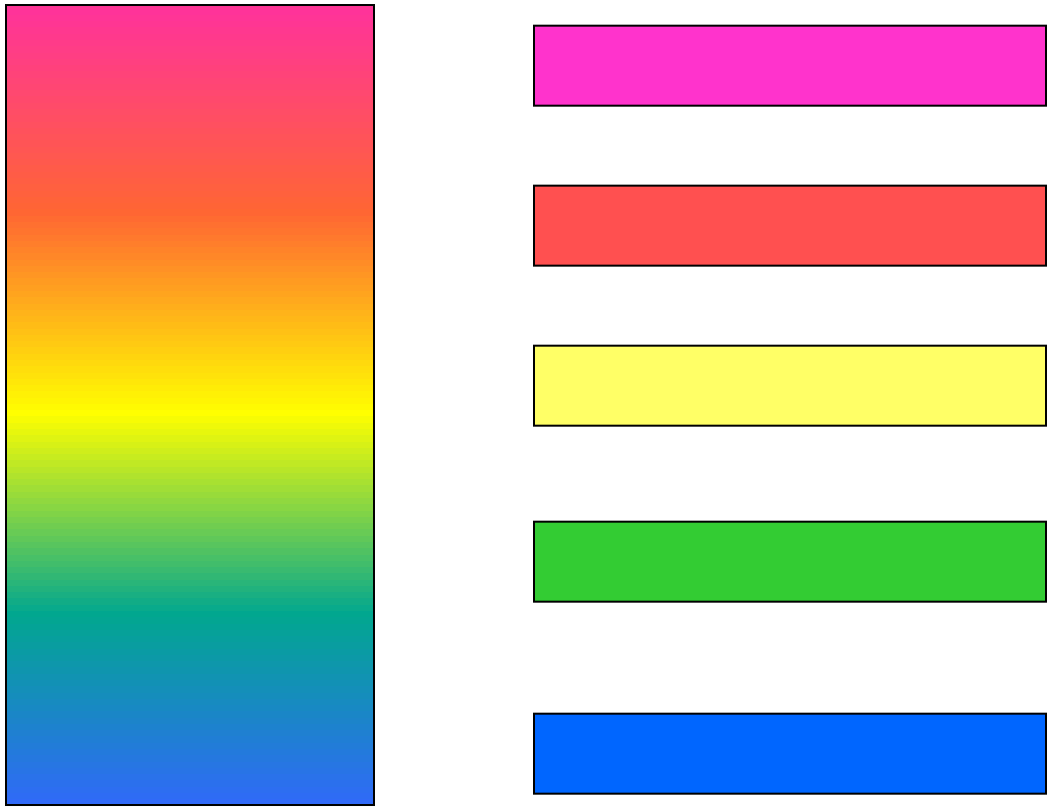
Department of Computer and IT Engineering
University of Kurdistan

Computer Architecture
(Review of Digital Design)

By: Dr. Alireza Abdollahpouri

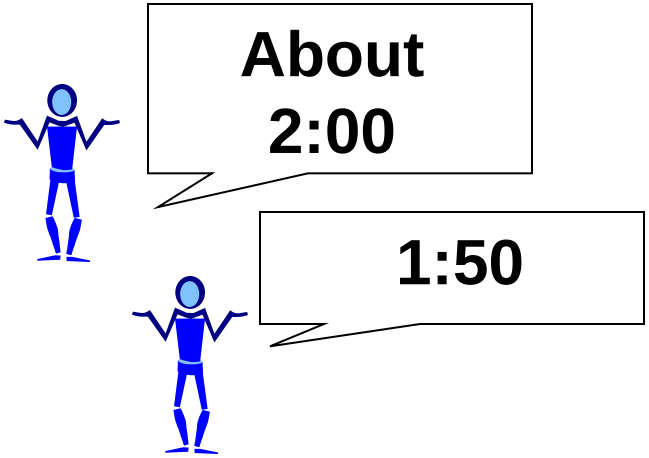
Analog offers Continuous Spectrum
Digital offer distinct Steps



Analog vs. Digital

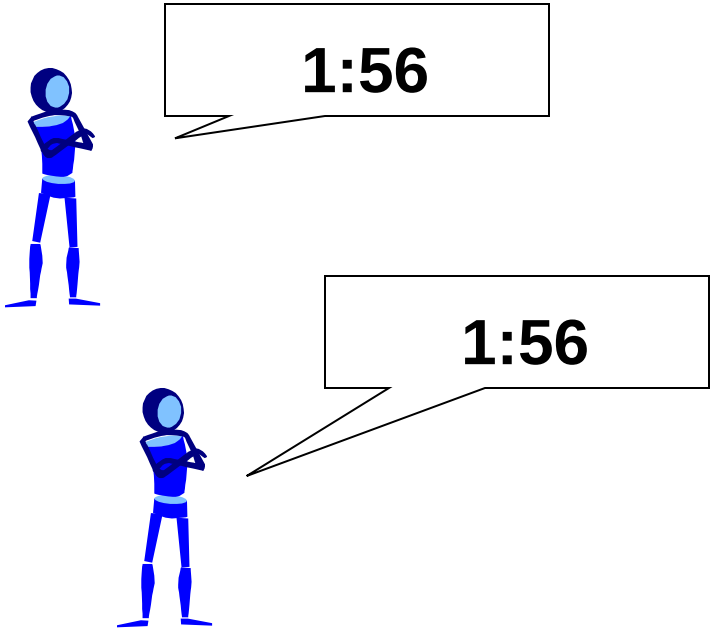
Analog has Ambiguity
Digital has only one interpretation

Analog Clock

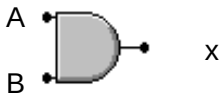
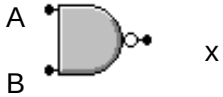
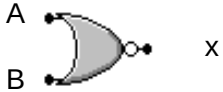
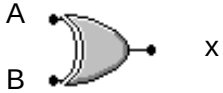
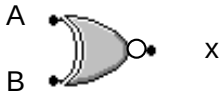


Digital Clock

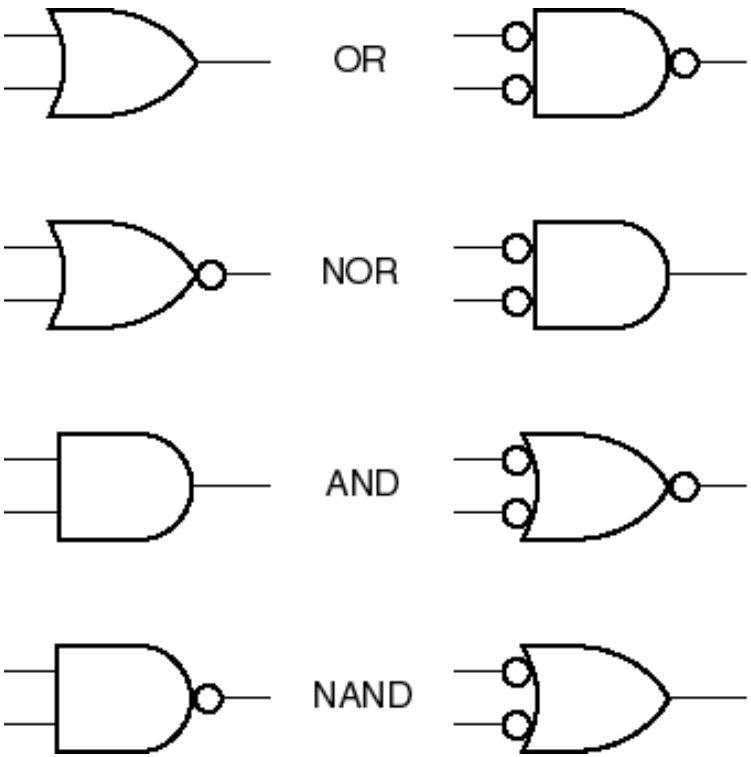
1:56 pm

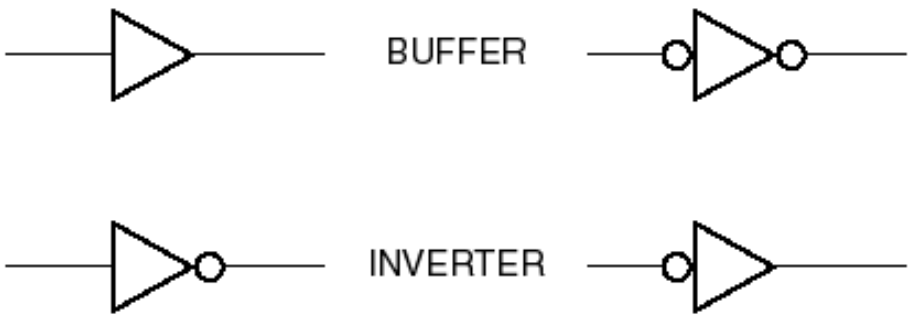


LOGIC GATES

Name	Graphic Symbol	Algebraic Function	Truth Table
AND		x = A . B or x = A B	**A B x** 0 0 0 0 1 0 1 0 0 1 1 1
OR		x = A + B	**A B x** 0 0 0 0 1 1 1 0 1 1 1 1
Inverter		x = A'	**A x** 0 1 1 0
NAND		x = (A B)'	**A B x** 0 0 1 0 1 1 1 0 1 1 1 0
NOR		x = (A + B)'	**A B x** 0 0 1 0 1 0 1 0 0 1 1 0
Exclusive-OR (XOR)		x = A ⊕ B	**A B x** 0 0 0 0 1 1 1 0 1 1 1 0
Exclusive-NOR or equivalence		x = (A ⊕ B)'	**A B x** 0 0 1 0 1 0 1 0 0 1 1 1

DeMorgan equivalent symbols





Which symbol to use?

Answer depends on signal names and active levels.

BOOLEAN ALGEBRA

Boolean Algebra is an algebra that deals with binary variables and logic operations.

The three basic logic operations are AND, OR and complement (invert).

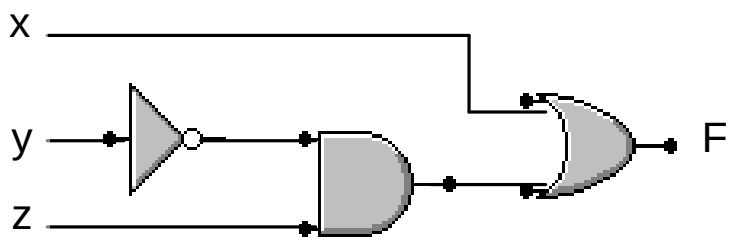
Example: $F = x + y' z$

BOOLEAN ALGEBRA

$$F = x + y' z$$

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

Truth Table



Logic Diagram for

Minterms and Maxterms

			Minterms		Maxterms	
X	**Y**	**Z**	**Product Term**	**Symbol**	**Sum Term**	**Symbol**
0	0	0	X' Y' Z'	m_0	X + Y + Z	M_0
0	0	1	X' Y' Z	m_1	X + Y + Z'	M_1
0	1	0	X' Y Z'	m_2	X + Y' + Z	M_2
0	1	1	X' Y Z	m_3	X + Y' + Z'	M_3
1	0	0	X Y' Z'	m_4	X' + Y + Z	M_4
1	0	1	X Y' Z	m_5	X' + Y + Z'	M_5
1	1	0	X Y Z'	m_6	X' + Y' + Z	M_6
1	1	1	X Y Z	m_7	X' + Y' + Z'	M_7

Deriving Boolean Expression from Truth Table

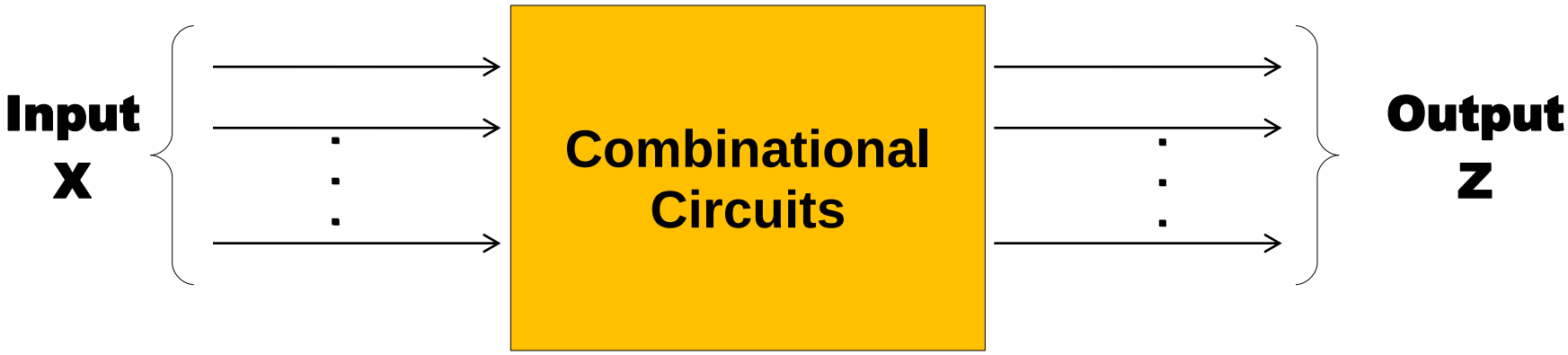
Input			Output	Minterm	
A	B	C	F	term	designation
0	0	0	1	A'B'C'	m0
0	0	1	0	A'B'C	m1
0	1	0	0	A'BC'	m2
0	1	1	1	A'BC	m3
1	0	0	0	AB'C'	m4
1	0	1	0	AB'C	m5
1	1	0	0	ABC'	m6
1	1	1	0	ABC	m7

=> F = A'B'C' + A'BC

=> F = m0 + m3

F = Σ m (0, 3)

Combinational Circuits



$$Z = F(X)$$

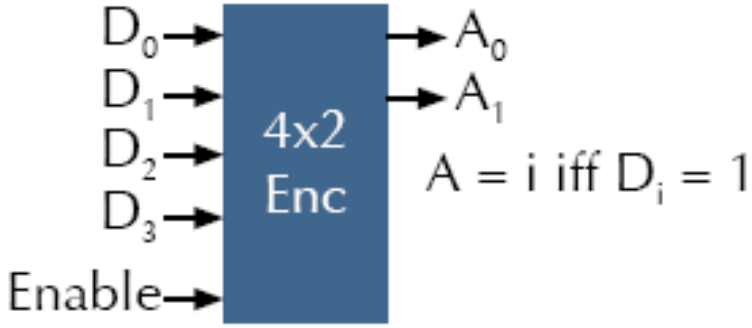
In combinational circuits, the output at any time is a direct function of the applied external inputs

Decoder & Encoder

Decoder

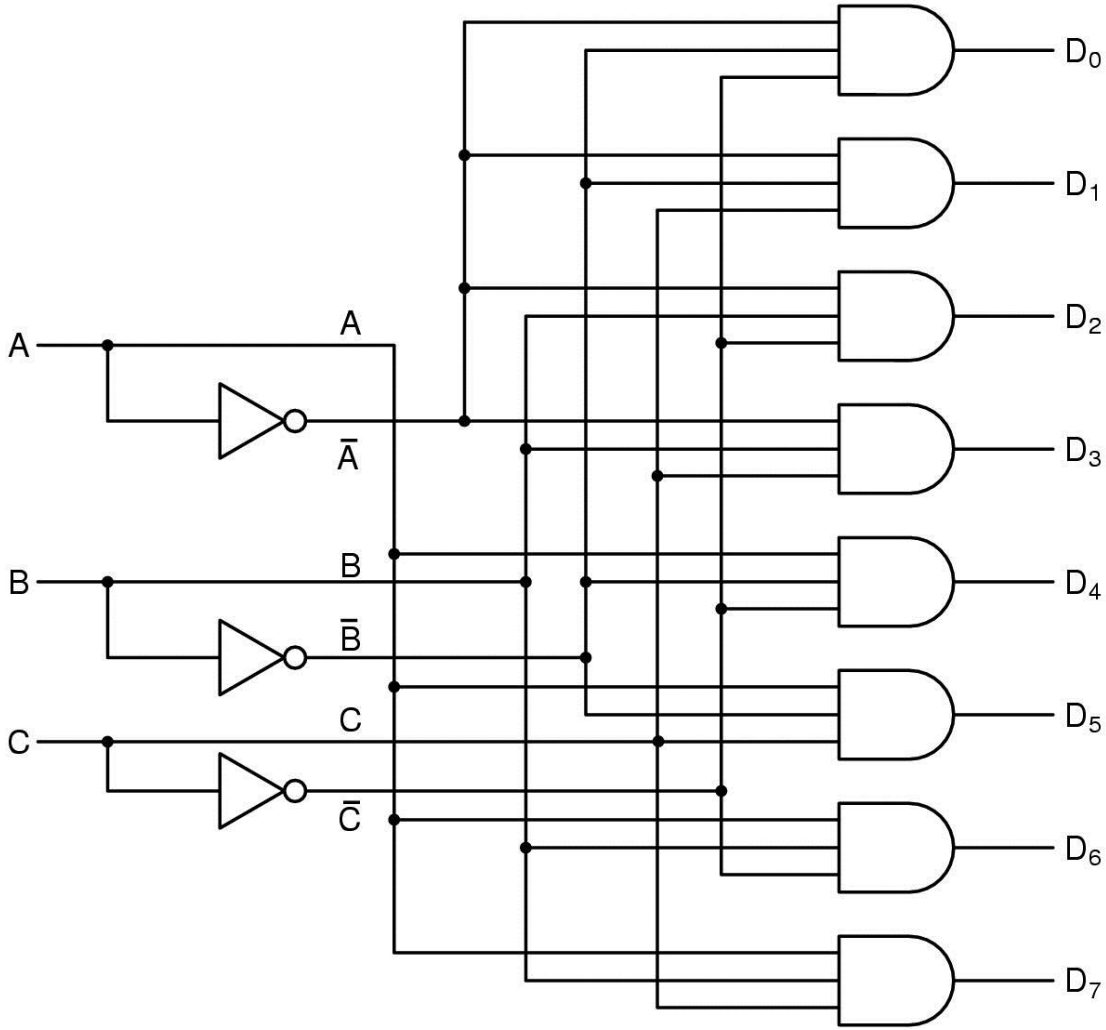
E	A_1	A_0	D_0	D_1	D_2	D_3
1	0	0	1	0	0	0
1	0	1	0	1	0	0
1	1	0	0	0	1	0
1	1	1	0	0	0	1
0	x	x	0	0	0	0

Encoder

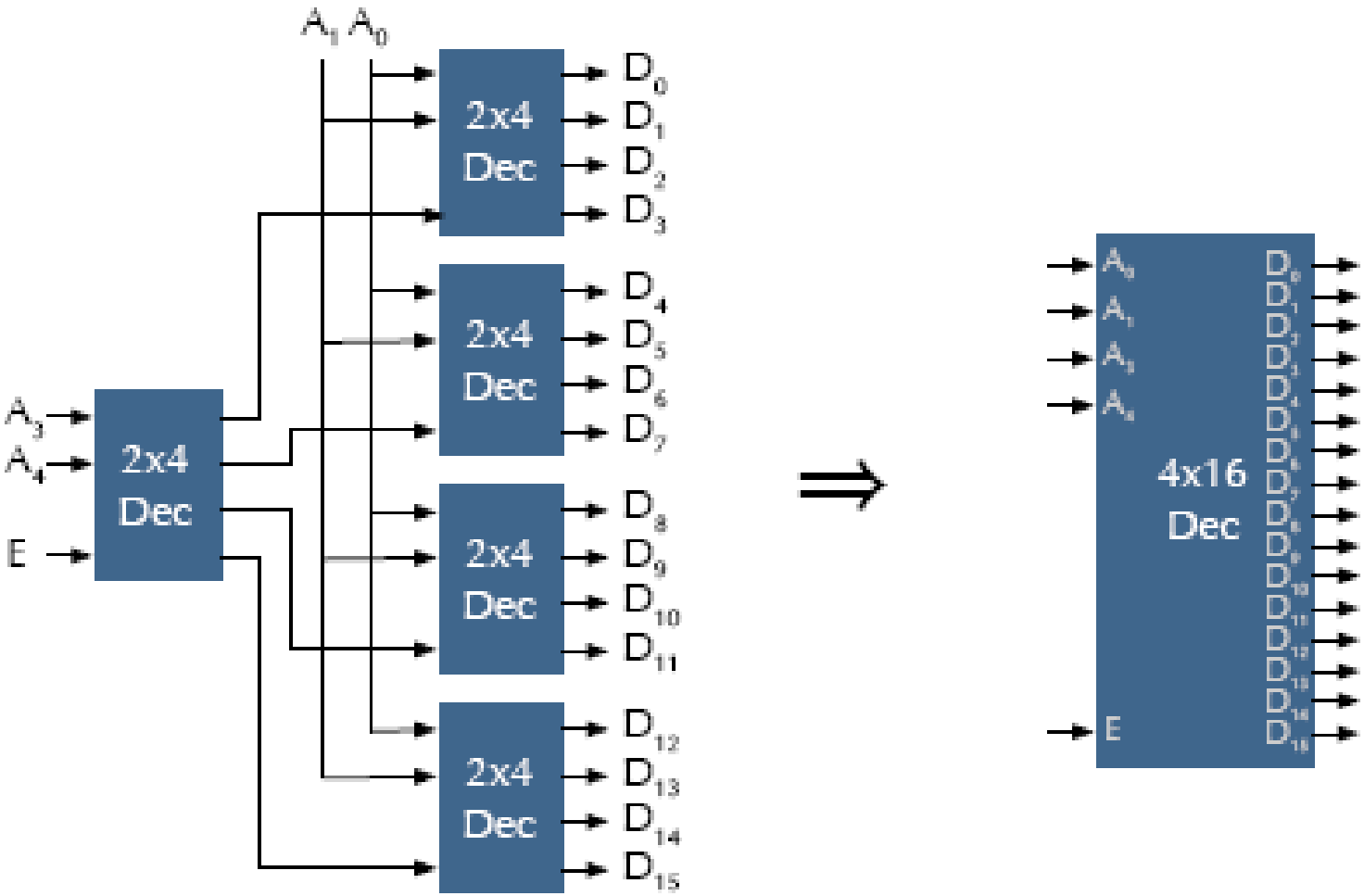


E	D_0	D_1	D_2	D_3	A_1	A_0
1	1	0	0	0	0	0
1	0	1	0	0	0	1
1	0	0	1	0	1	0
1	0	0	0	1	1	1
0	0	0	0	0	x	x

3-8 Decoder



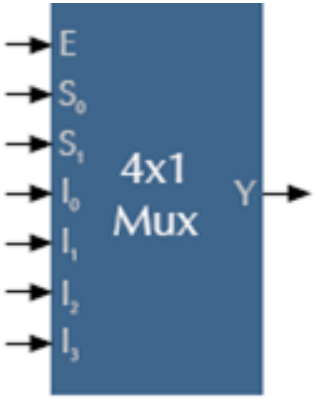
Combining small decoders to build a bigger one



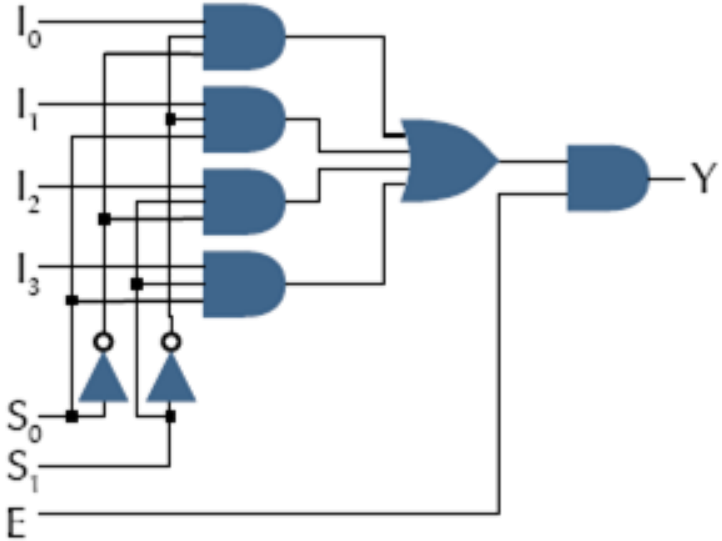
Multiplexer

select one input to send to output

- 2^n *data* inputs plus n *select* inputs
  - data inputs are labelled with unique n-bit number
- one output
  - has value of data input with label matching select

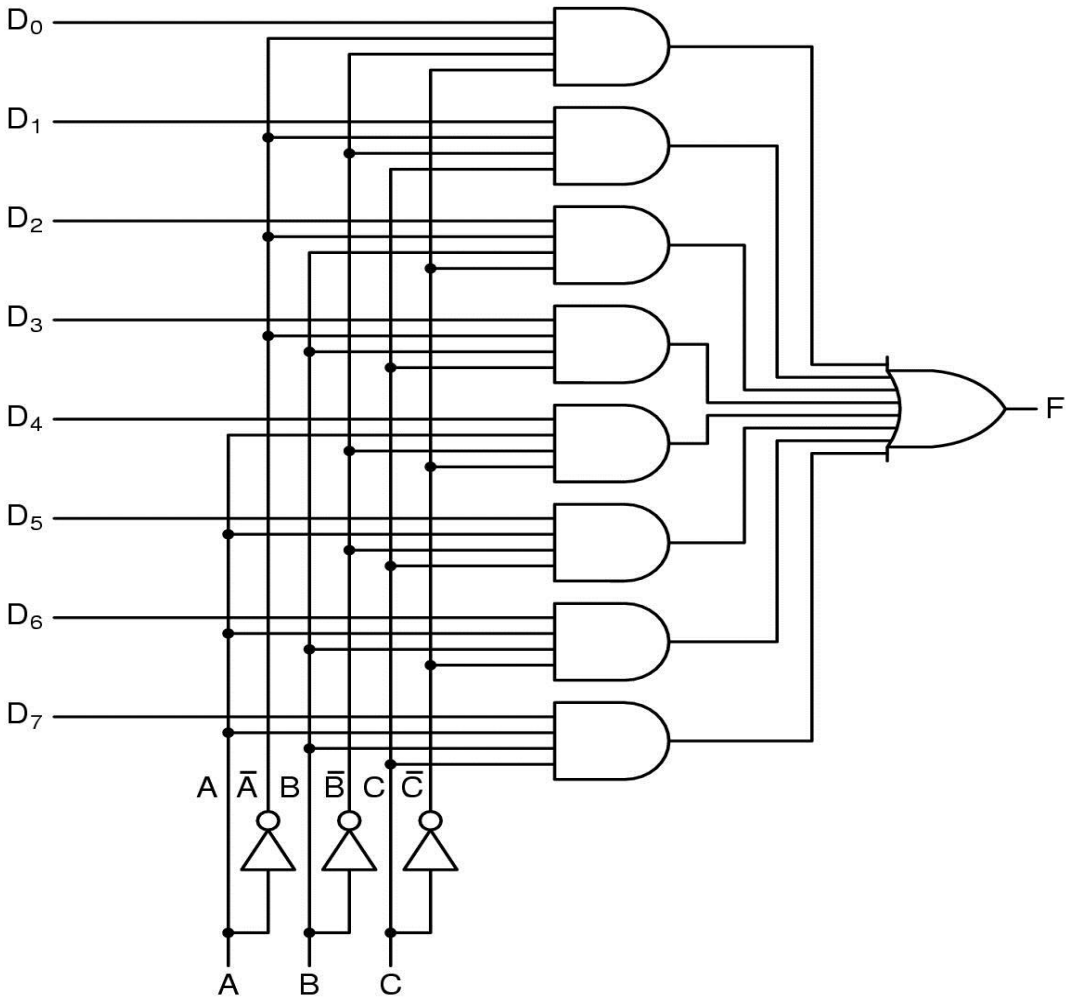


implementation

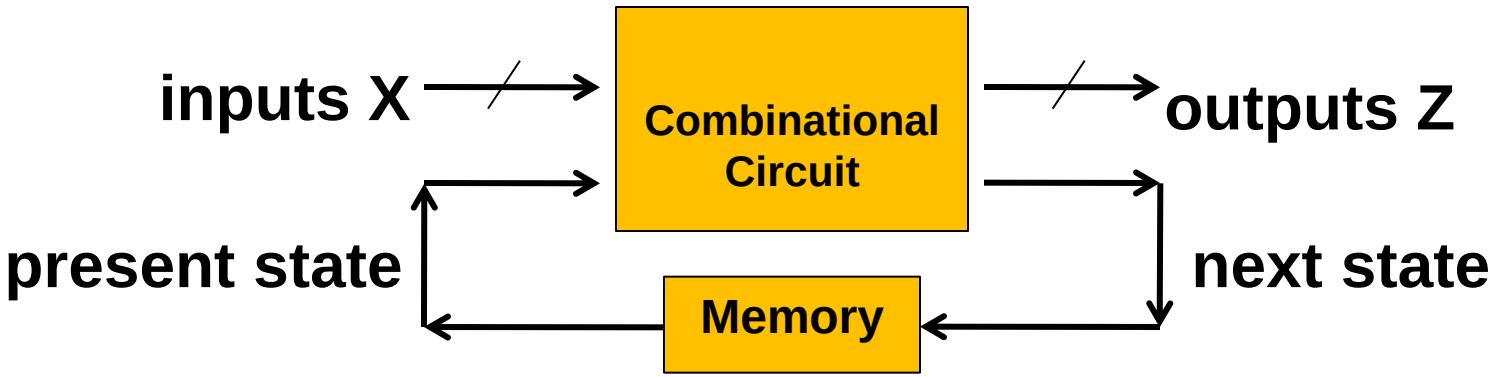


E	S_1	S_0	Y
1	0	0	I_0
1	0	1	I_1
1	1	0	I_2
1	1	1	I_3
0	x	x	0

Multiplexer

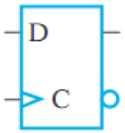
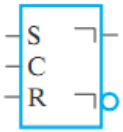
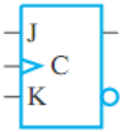
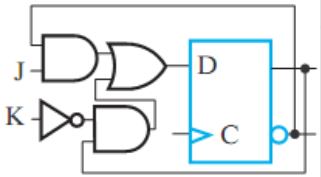
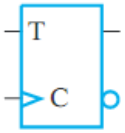
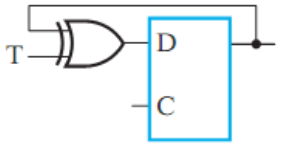


Sequential Circuits

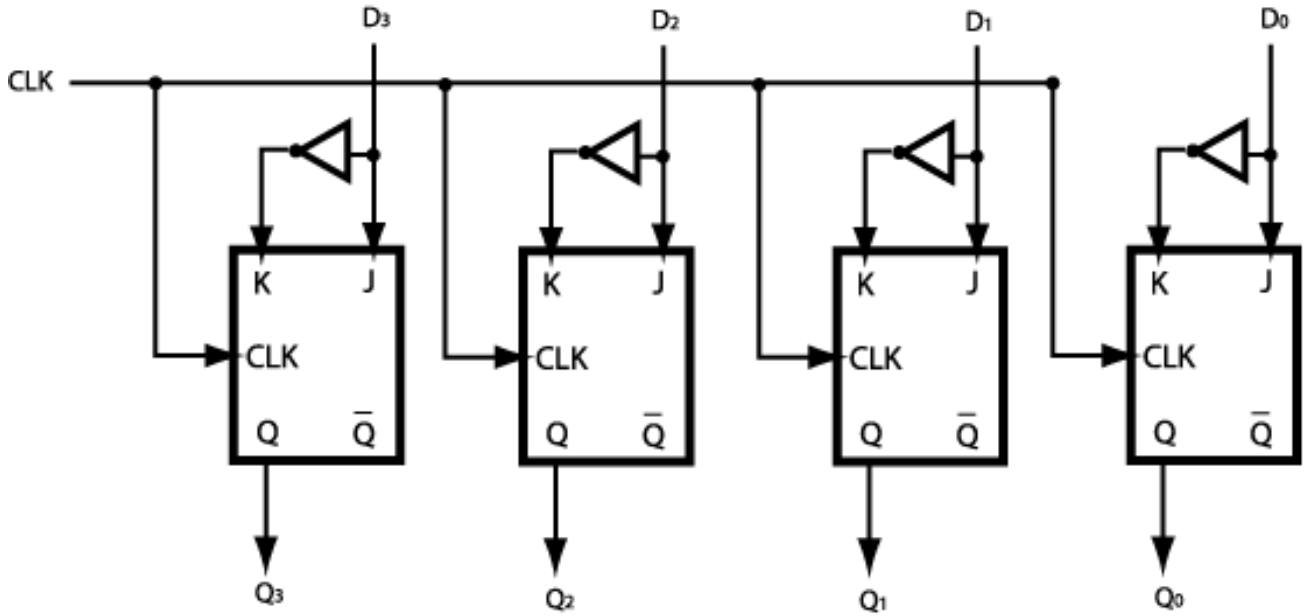


- A **sequential** circuit:
- outputs depends on inputs and previous inputs
  - Previous inputs are stored as binary information into memory
  - The stored information at any time defines a **state**
- next state depends on inputs and present state

Flip Flops Sheet (Mano's Textbook)

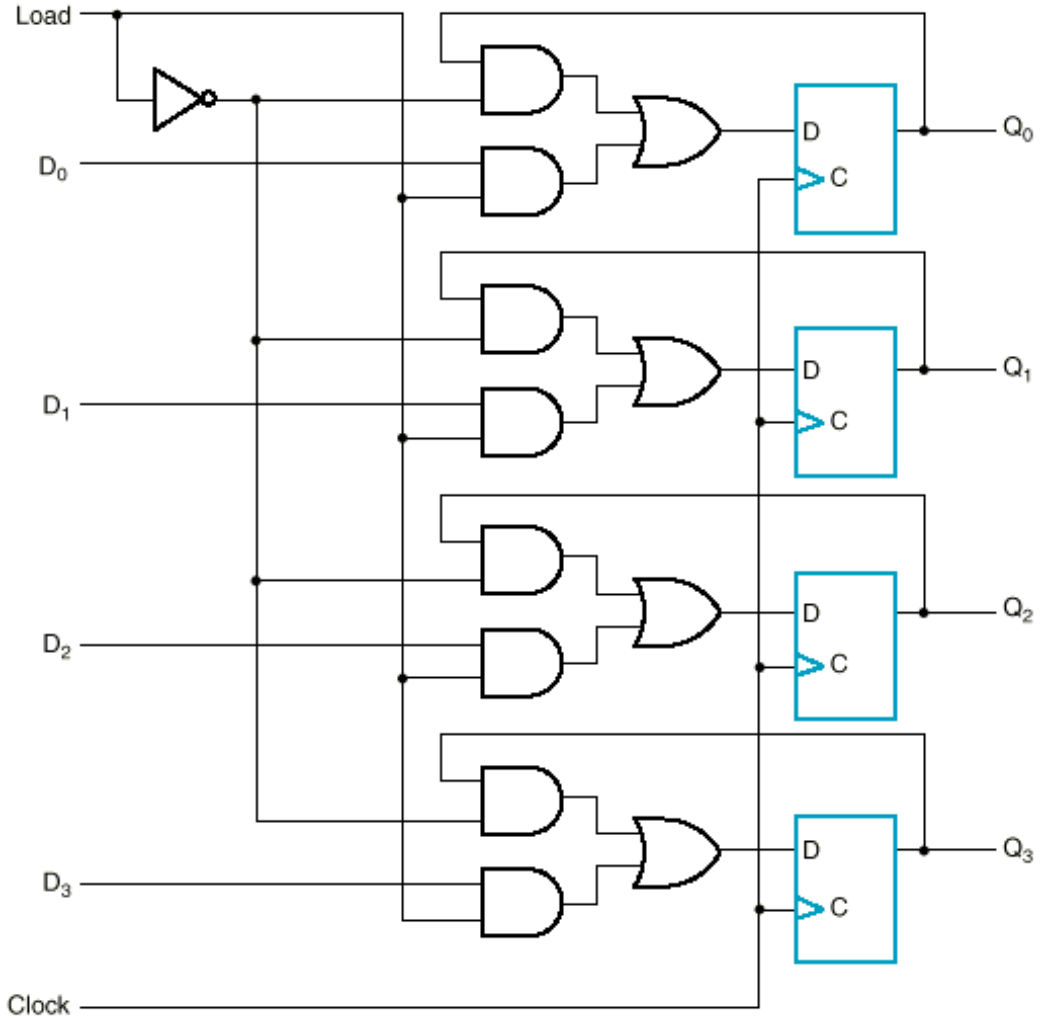
Type	Symbol	Logic Diagrams	Characteristic Table				Characteristic Equation	Excitation Table				
			D		Q(t+1)	Operation		Q(t+1)		D		Operation
D		See Figure 5-12	0		0	Reset	$Q(t+1) = D(t)$	0		0		Reset
			1		1	Set		1		1		Set
			S	R	Q(t+1)	Operation		Q(t)	Q(t+1)	S	R	Operation
SR		See Figure 5-9	0	0	$Q(t)$	No change	$Q(t+1) = S(t) + \overline{R}(t)\,Q(t)$	0	0	0	X	No change
			0	1	0	Reset		0	1	1	0	Set
			1	0	1	Set		1	0	0	1	Reset
			1	1	?	Undefined		1	1	X	0	No change
			J	K	Q(t+1)	Operation		Q(t)	Q(t+1)	J	K	Operation
JK			0	0	$Q(t)$	No change	$Q(t+1) = J(t)\,\overline{Q}(t) + \overline{K}(t)\,Q(t)$	0	0	0	X	No change
			0	1	0	Reset		0	1	1	X	Set
			1	0	1	Set		1	0	X	1	Reset
			1	1	$\overline{Q}(t)$	Complement		1	1	X	0	No Change
				T	Q(t+1)	Operation		Q(t+1)		T		Operation
T				0	$Q(t)$	No change	$Q(t+1) = T(t) \oplus Q(t)$	$Q(t)$		0		No change
				1	$\overline{Q}(t)$	Complement		$\overline{Q}(t)$		1		Complement

Register

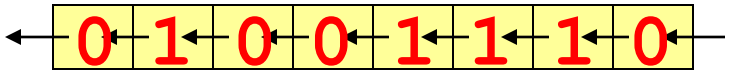


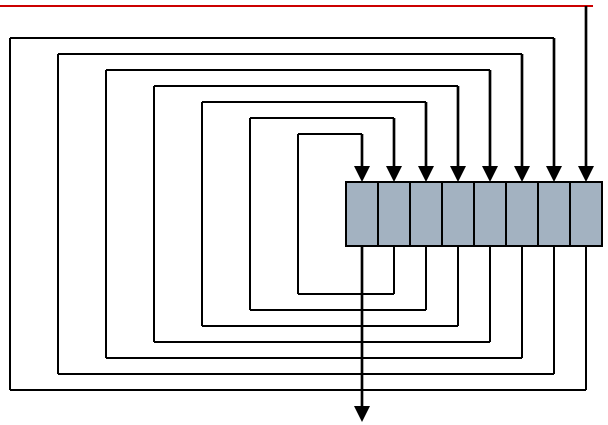
- **A register is a group of flip-flops.**
- **An n-bit register is made of n flip-flips and can store n bits**
- **A register may have additional combinational gates to perform certain operations**

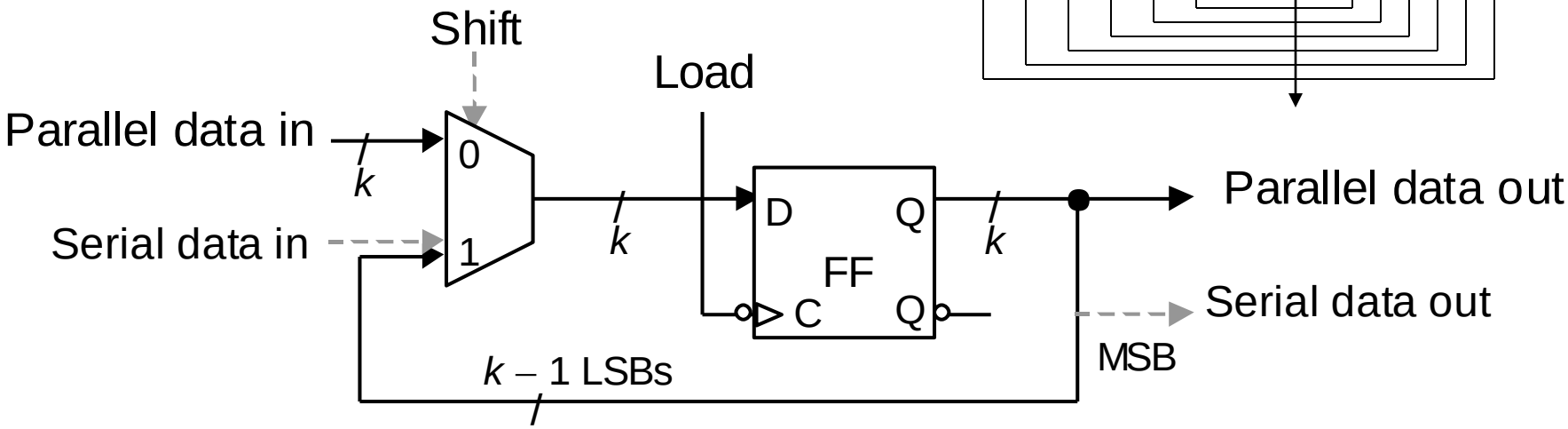
Register



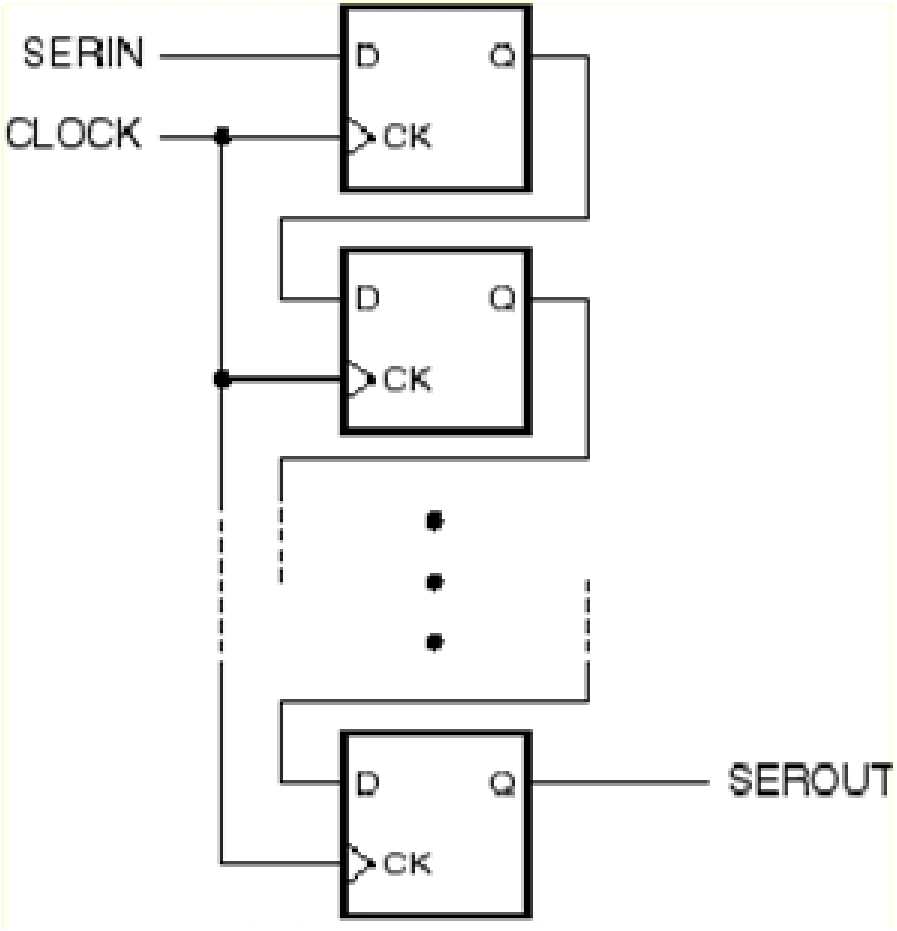
Shift Register



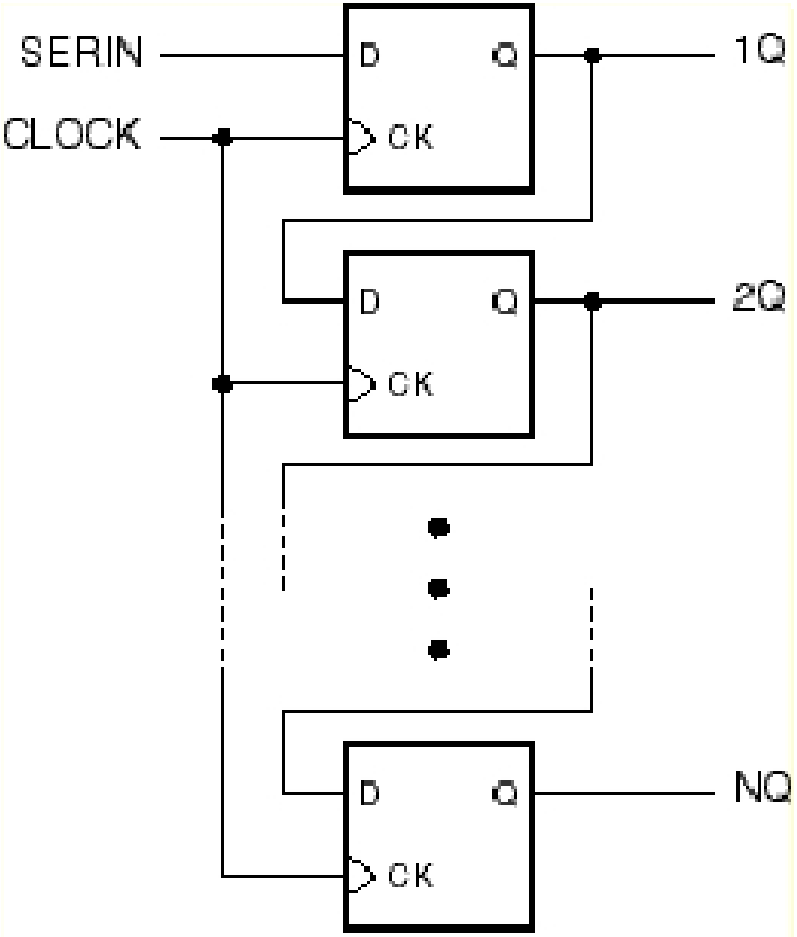




shift register

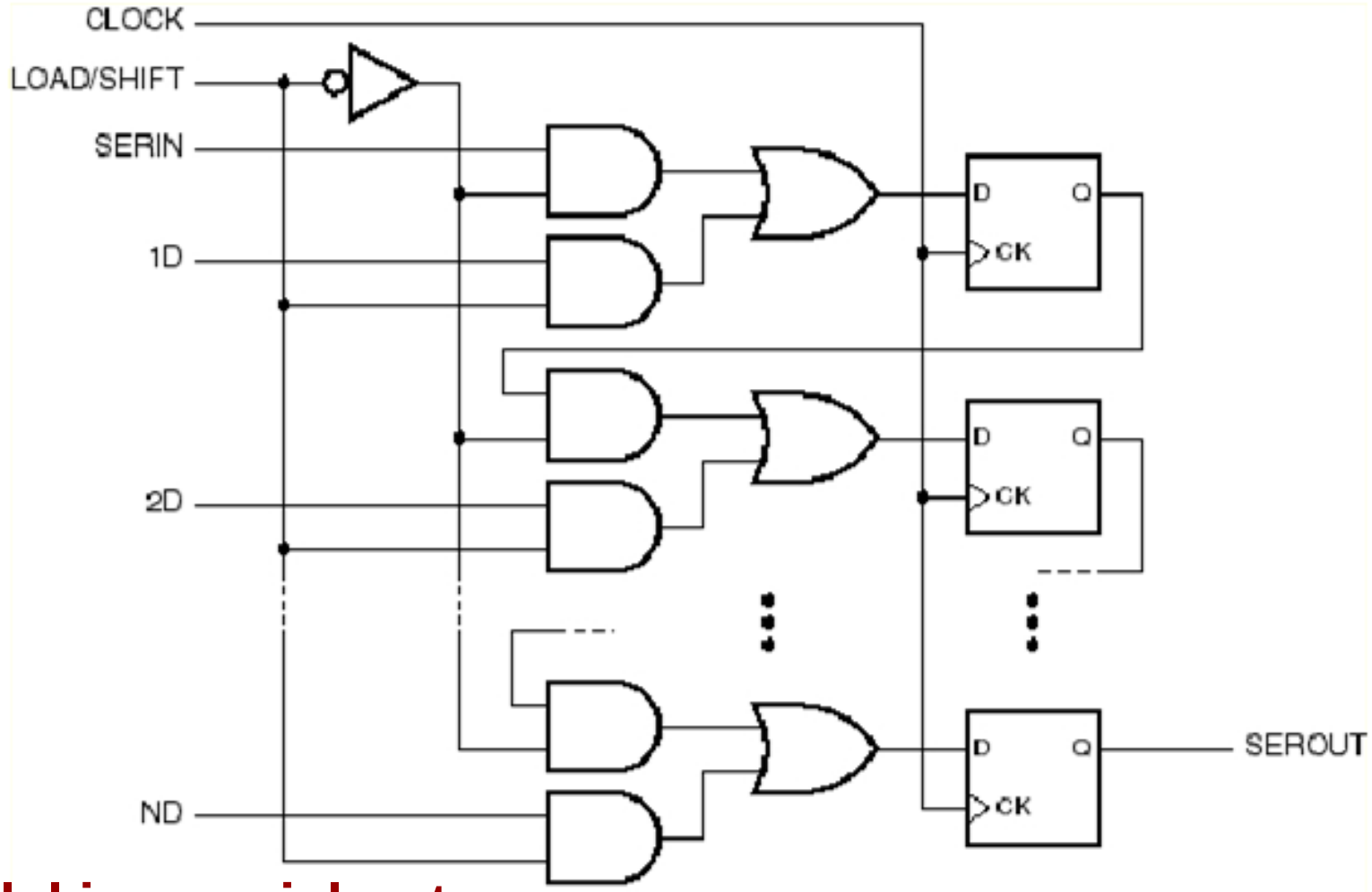


Serial in – serial out



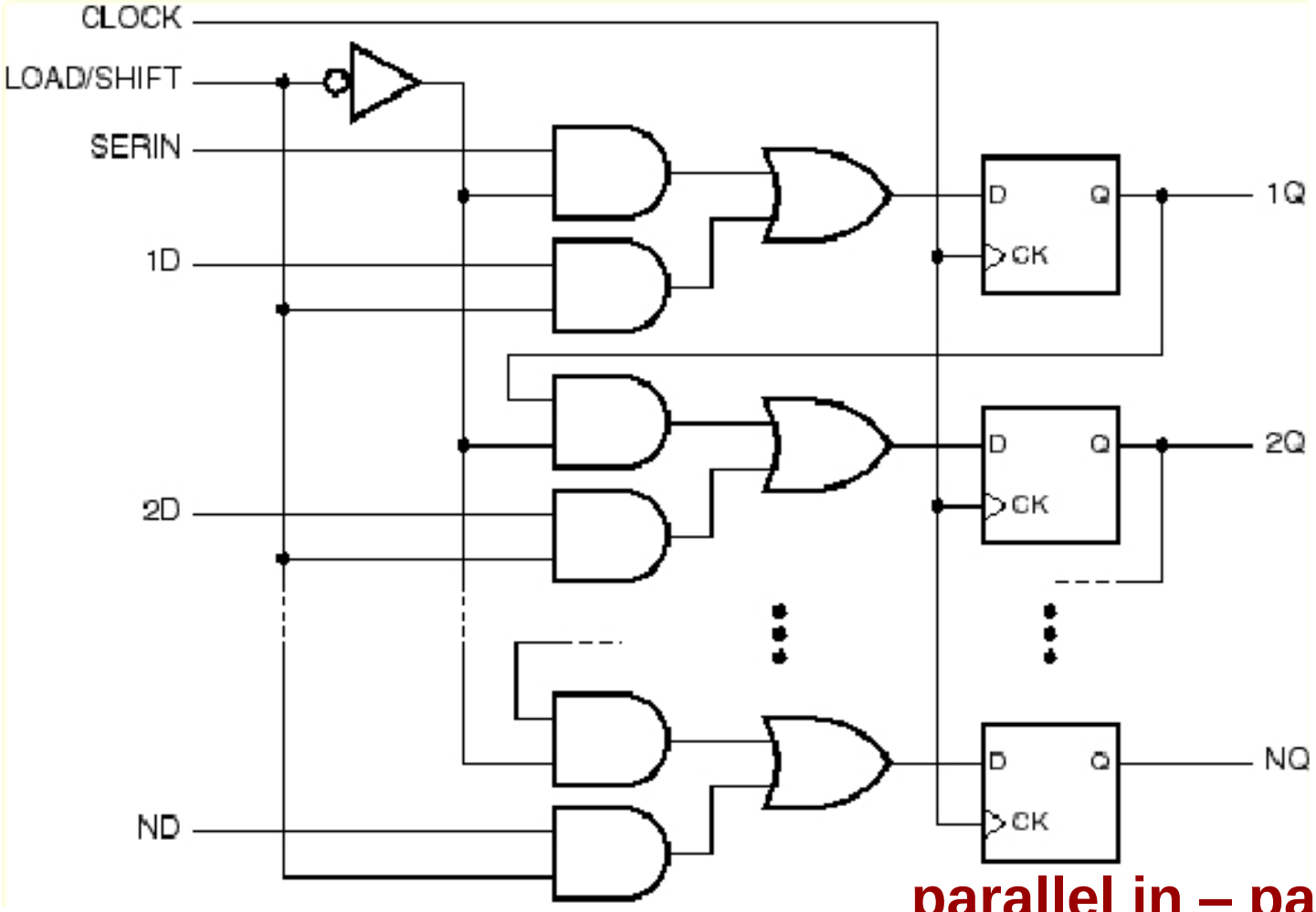
Serial in – parallel out

shift register



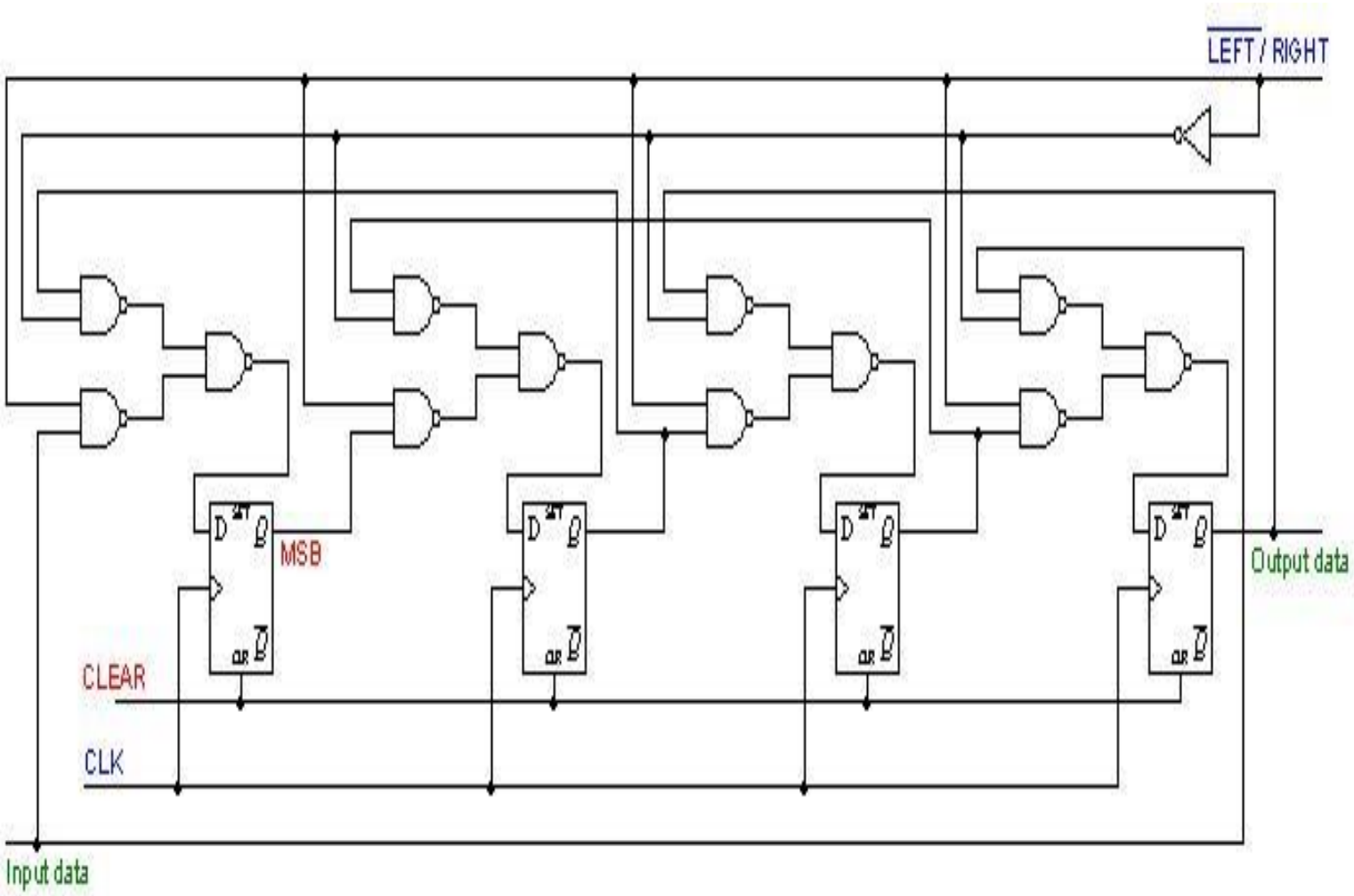
parallel in – serial out

shift register

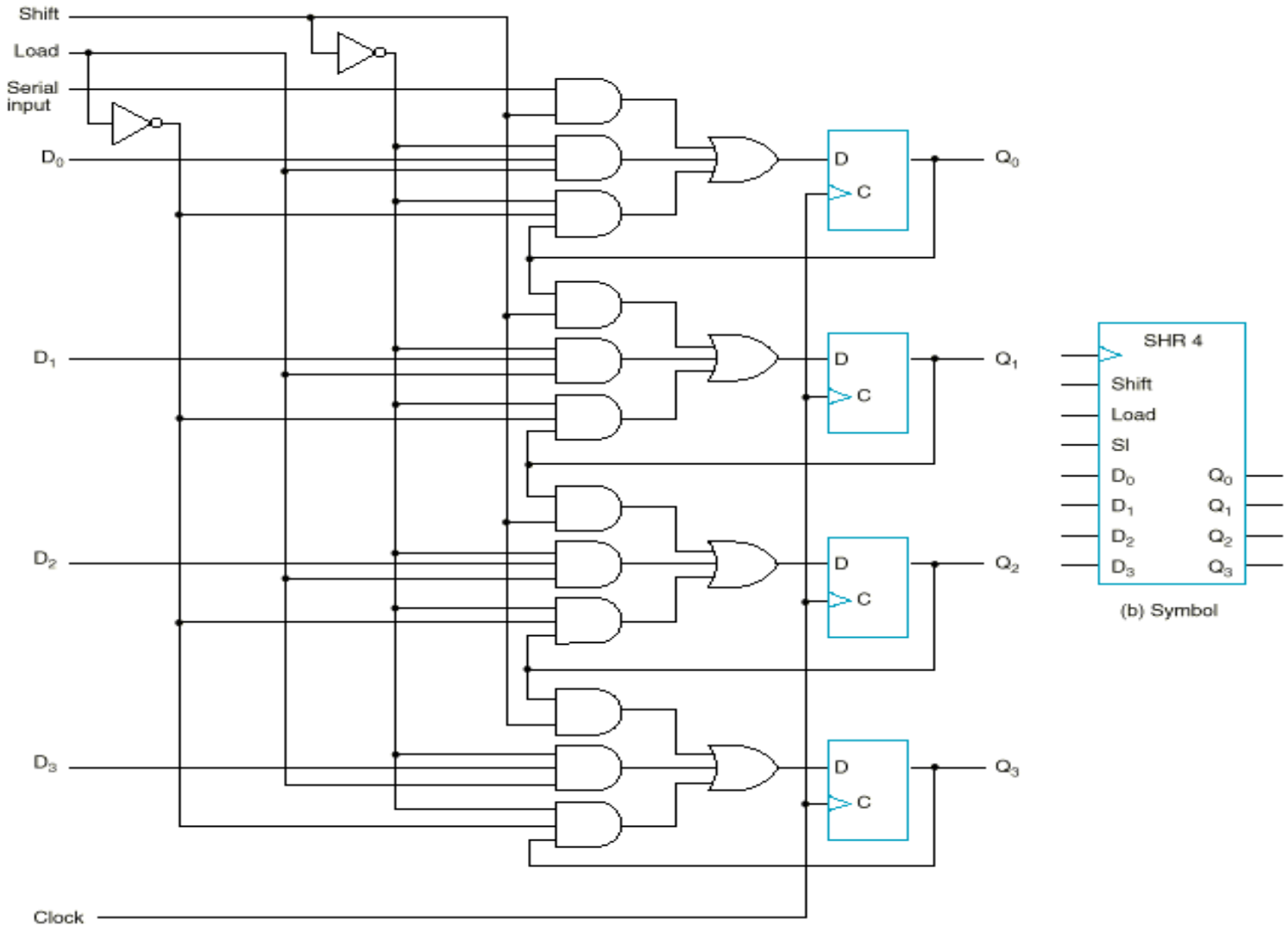


parallel in – parallel out

Bidirectional shift register

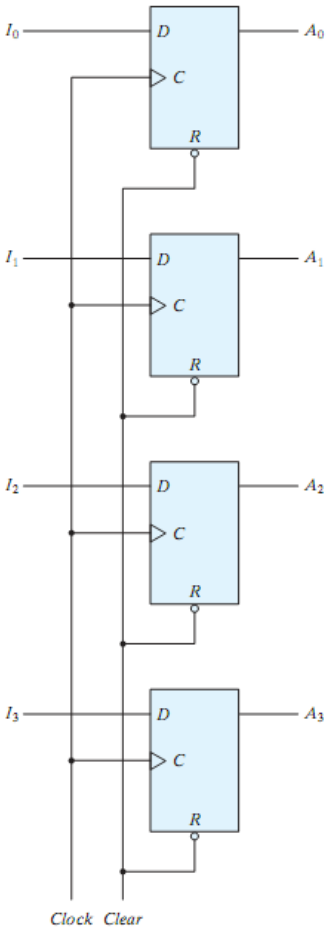


shift register with load



4-Bit Register

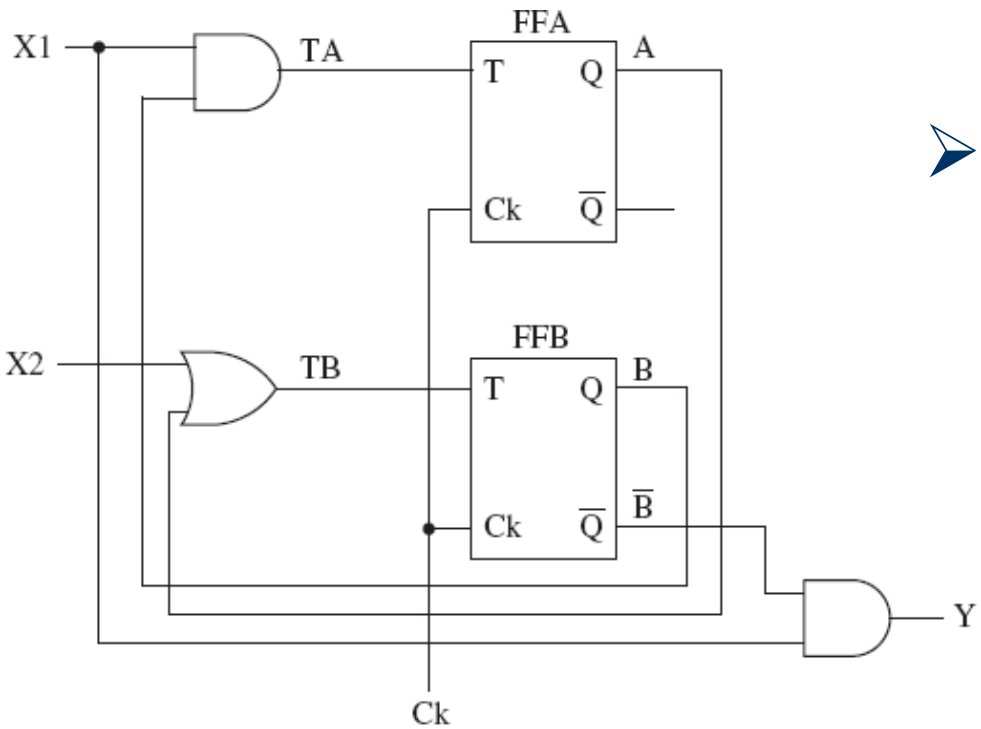
- A simple 4-bit register can be made with 4 D-FF
- Common Clock
  - At each positive-edge, 4 bits are loaded in parallel
  - Previous data is overwritten
- Common Clear
  - Asynchronous clear
  - When Clear = 0, all FFs are cleared; i.e. 0 is stored.



Sequential analysis

- Step 1: List all possible combinations of current state and current input in an analysis table
- Step 2: For each combination, compute the output and the current inputs to the state registers
- Step 3: From the characteristic table, determine the next state and construct the state transition table and diagram

Example problem



- State registers: FFA & FFB (T flip flops)
- Combinational circuit
  - inputs:
    - X1 AND B (TA)
    - X2 OR A (TB)
    - TA & TB are inputs to FFA & FFB
  - output:
    - B’ AND X1 (Y)

Example problem

- 2 flip flops, so 4 possible states:

A	B
0	0
0	1
1	0
1	1

- **2 inputs, so 4 possible input combinations:**

X1	X2
0	0
0	1
1	0
1	1

Analysis table for sample problem circuit

A(t)	B(t)	X1(t)	X2(t)	Y(t)	TA(t)	TB(t)	A(t + 1)	B(t + 1)
0	0	0	0	0	0	0	0	0
0	0	0	1	0	0	1	0	1
0	0	1	0	1	0	0	0	0
0	0	1	1	1	0	1	0	1
0	1	0	0	0	0	0	0	1
0	1	0	1	0	0	1	0	0
0	1	1	0	0	1	0	1	1
0	1	1	1	0	1	1	1	0
1	0	0	0	0	0	1	1	1
1	0	0	1	0	0	1	1	1
1	0	1	0	1	0	1	1	1
1	0	1	1	1	0	1	1	1
1	1	0	0	0	0	1	1	0
1	1	0	1	0	0	1	1	0
1	1	1	0	0	1	1	0	0
1	1	1	1	0	1	1	0	0

- 1st 4 columns list possible combinations of initial state & initial input
- By the logic diagram, we know:
  - Y(t)=X1(t) AND B'(t)
  - TA(t)=X1(t) AND B(t)
  - TB(t)=X2(t) OR A(t)
- Compute next 3 columns given above
- Compute last 2 from:
  - characteristic table for T flip flop
  - initial state of flip flop
  - flip flop's initial input

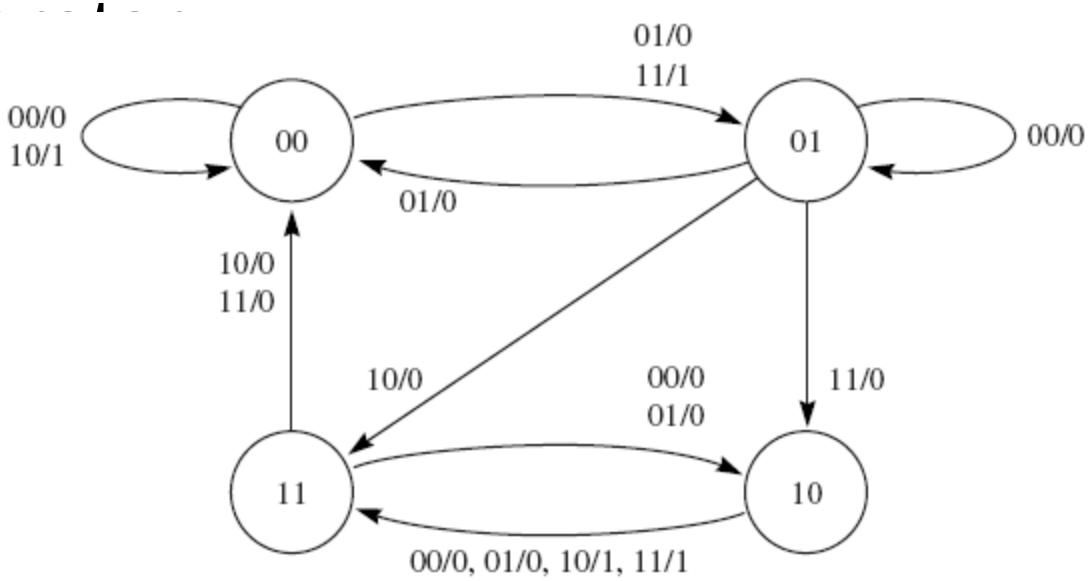
State transition table

- Table shows simple rearrangement of selected columns from table on previous slide
- For given initial state A(t)B(t) and input X1(t)X2(t), lists next state (A+1)(t)(B+1)(t) and initial output Y(t)
- States listed as ordered pairs – next state followed by initial output

	X1(*t*) X2(*t*)			
A(*t*) B(*t*)	00	01	10	11
00	00, 0	01, 0	00, 1	01, 1
01	01, 0	00, 0	11, 0	10, 0
10	11, 0	11, 0	11, 1	11, 1
11	10, 0	10, 0	00, 0	00, 0
	A(*t* + 1) B(*t* + 1), Y(*t*)			

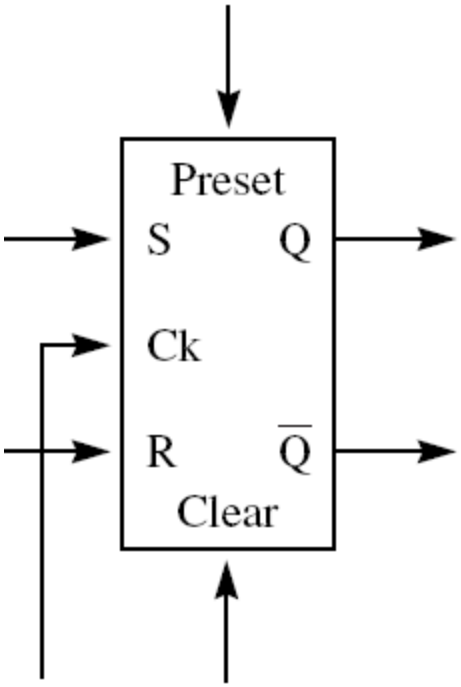
State transition diagram

- Easier to visualize circuit behavior
- Transitions listed as ordered pairs of input followed by initial output, with slash separ



Asynchronous inputs

- An *asynchronous input* changes state of a flip-flop immediately without regard to CP
  - Preset sets Q to 1
  - Clear clears Q to 0
- Used to initialize the state of a machine
- Normal operation: both lines 0



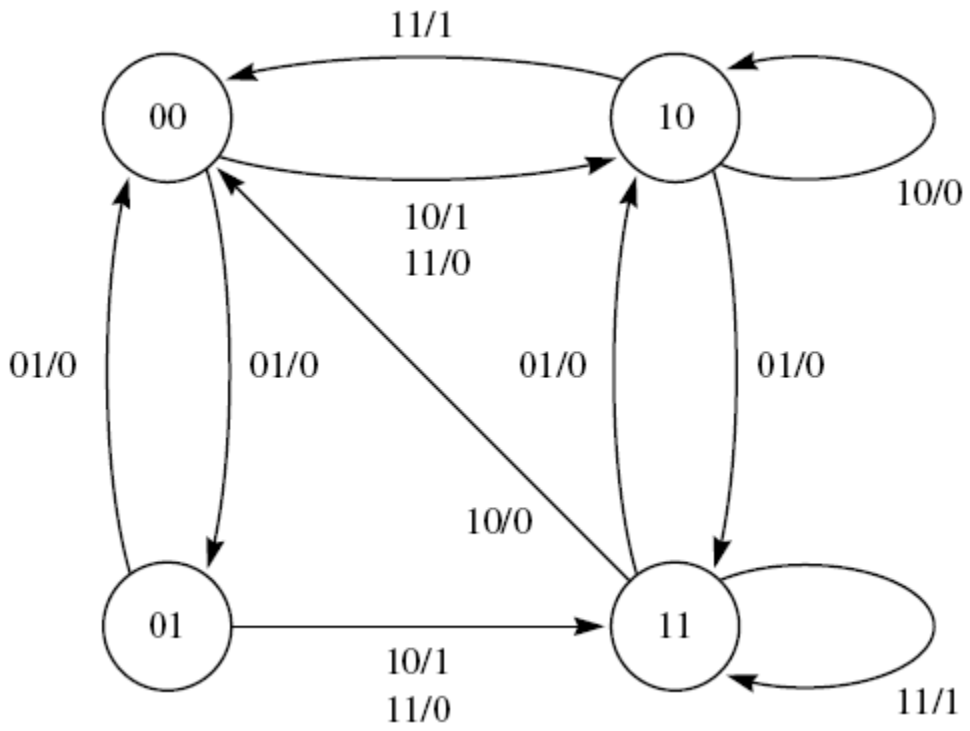
Sequential design

- Given the state transition diagram, the output, and the type of flip-flop to be used, design the combinational circuit
- Any unused input combinations or unused states are don’t care conditions
- 2^n *states are possible with n flip-flops*

Design steps

- Step 1: In a design table, list the initial state, input, and output, and from the transition diagram list the next state
- Step 2: Use the excitation table for the given type of flip-flop to determine the input required for the state registers
- Step 3: Use Karnaugh maps to design a minimized two-level circuit for each flip-flop input

Sample problem



Design table for sample problem

Initial state		Initial input		Initial output	Next state		Flip-flop input conditions			
							FFA		FFB	
$A(t)$	$B(t)$	$X1(t)$	$X2(t)$	$Y(t)$	$A(t+1)$	$B(t+1)$	$SA(t)$	$RA(t)$	$SB(t)$	$RB(t)$
0	0	0	1	0	0	1	0	×	1	0
0	1	0	1	0	0	0	0	×	0	1
1	1	0	1	0	1	0	×	0	0	1
1	0	0	1	0	1	1	×	0	1	0
0	0	1	1	0	1	0	1	0	0	×
0	1	1	1	0	1	1	1	0	×	0
1	1	1	1	1	1	1	×	0	×	0
1	0	1	1	1	0	0	0	1	0	×
0	0	1	0	1	1	0	1	0	0	×
0	1	1	0	1	1	1	1	0	×	0
1	1	1	0	0	0	0	0	1	0	1
1	0	1	0	0	1	0	×	0	0	×

Sequential design & K-maps

- Each flip flop in the problem can be considered a function of four variables:
  - initial state (AB)
  - input (X1X2)

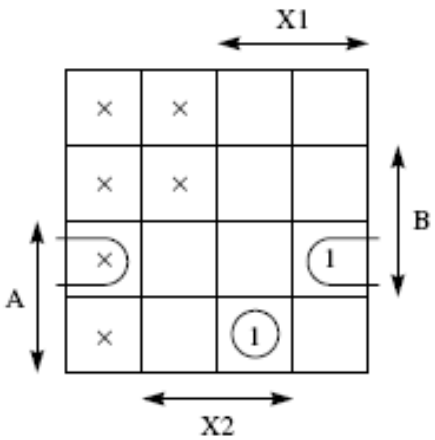
- To design the combinational circuit we need a 4-variable K-map for each flip flop input

K-maps for sample problem

- Figures a and b below show K-maps for S & R inputs to FFA
  - Row values are AB, columns are X1X2
  - X1X2 = 00 is a don’t care condition for both inputs, so first column of both tables is X

AB \ X1 X2	00	01	11	10
00	×		1	1
01	×		1	1
11	×	×	×	
10	×	×		×

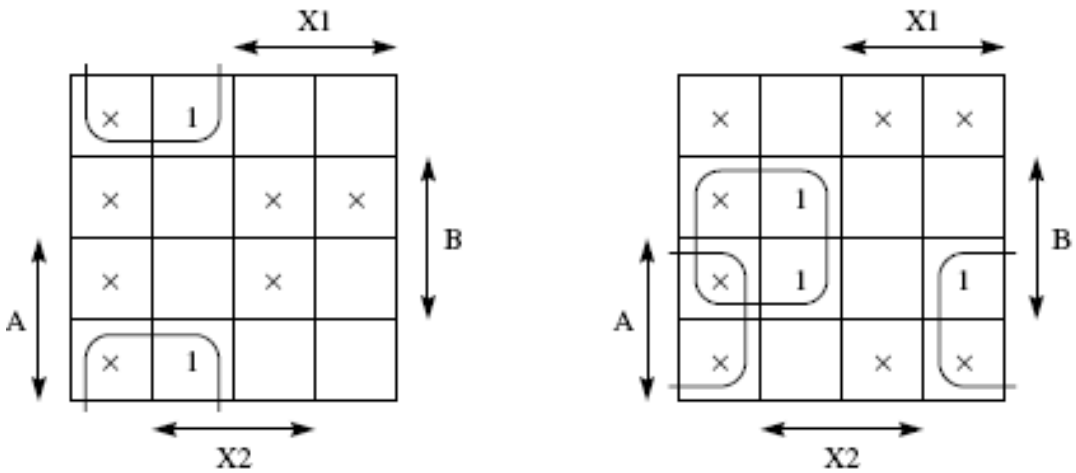
(a) SA = $\overline{A}$ X1



(b) RA = A B $\overline{X2}$ + A $\overline{B}$ X1 X2

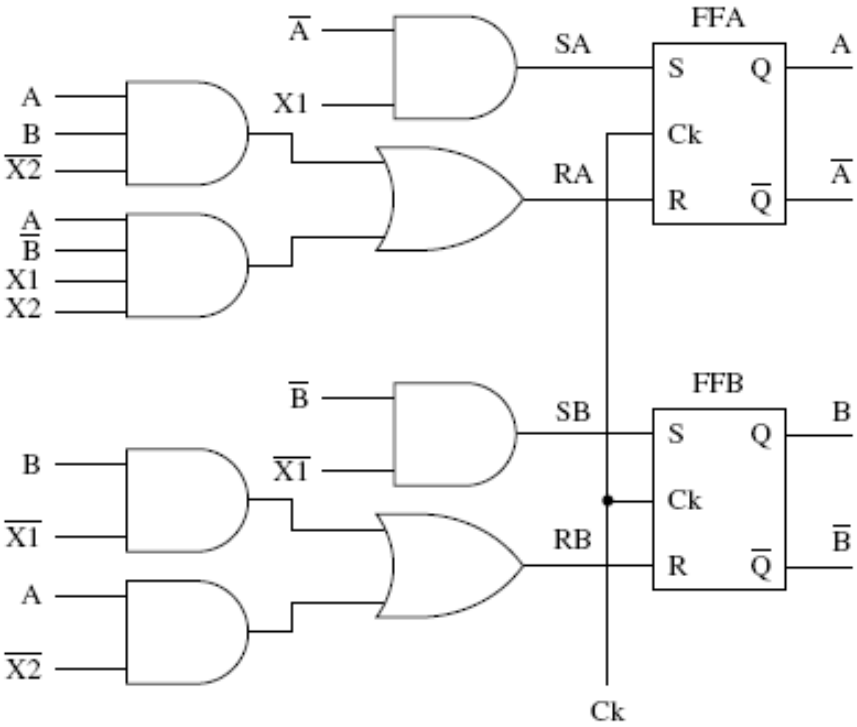
K-maps for sample problem

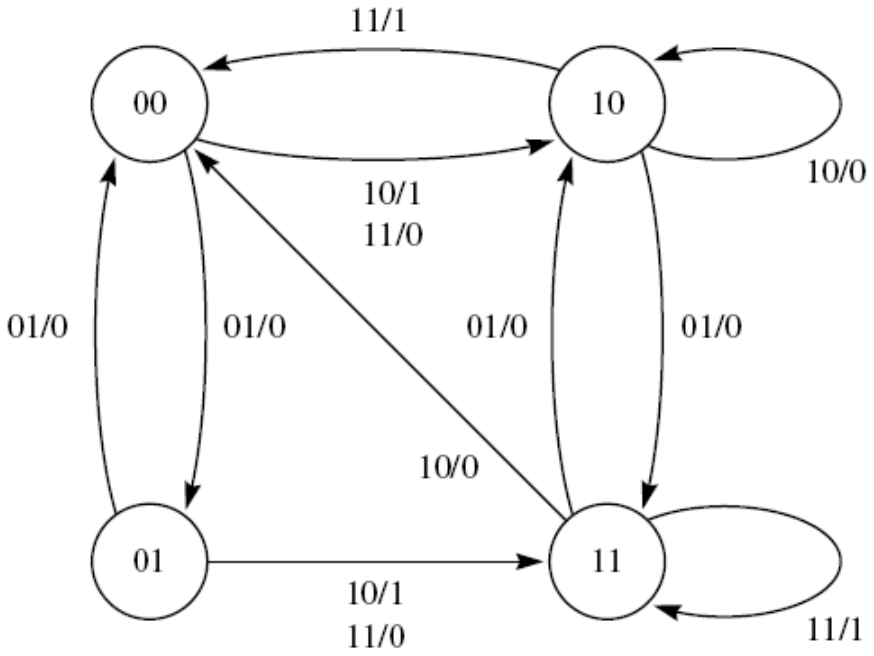
- Figures c and d show inputs to FFB
- Note that we can take advantage of don't care conditions to minimize circuit



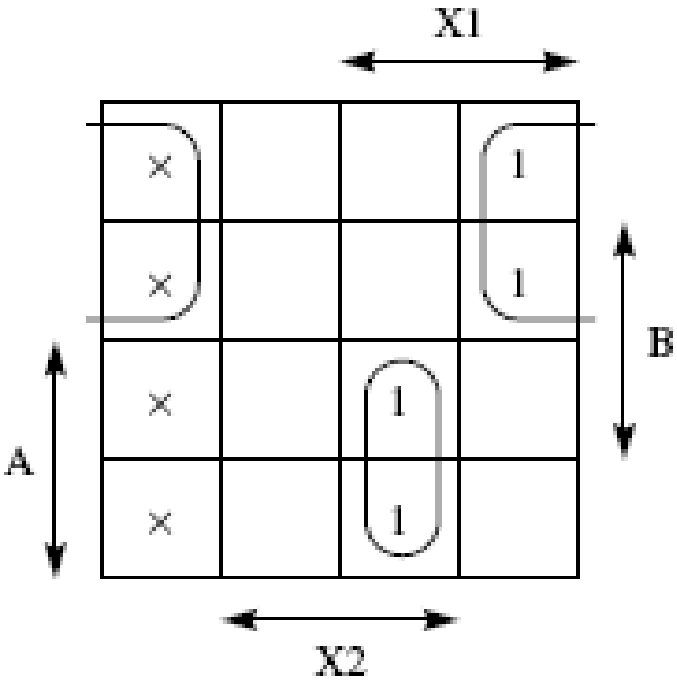
(c) SB = $\overline{B}\,\overline{X1}$ (d) RB = $B\,\overline{X1} + A\,\overline{X2}$

Resulting circuit with original spec

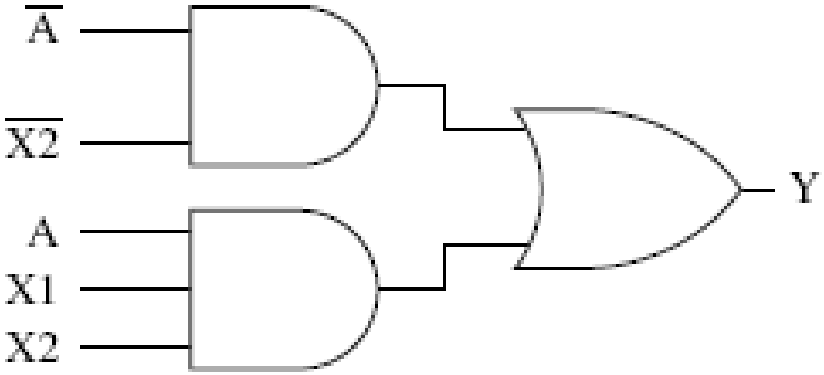




K-map & circuit for output Y



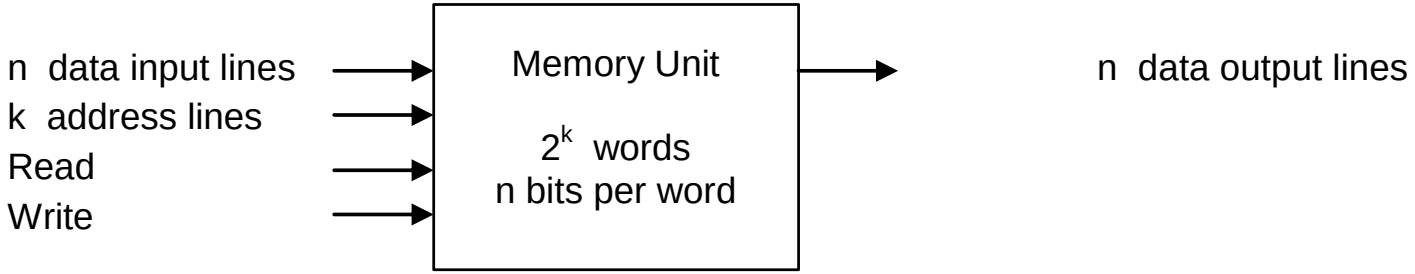
(e) $Y = \overline{A}\,\overline{X2} + A\,X1\,X2$



Memories

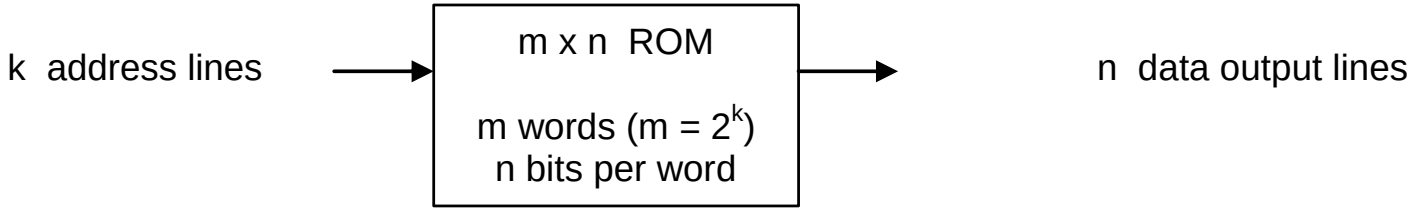
RAM: Random Access Memory
ROM: Read Only Memory

RAM (block Diagram):



2^k x n memory unit (i.e. number of words x word length)

ROM: Asynchronuously, ROM data ouyput lines automatically provide the "n" bits of the word selected by address lines.



ROM types: PROM (Programmable ROM), EPROM (Erasable PROM), EEPROM (Electrically Erasable PROM).


Questions