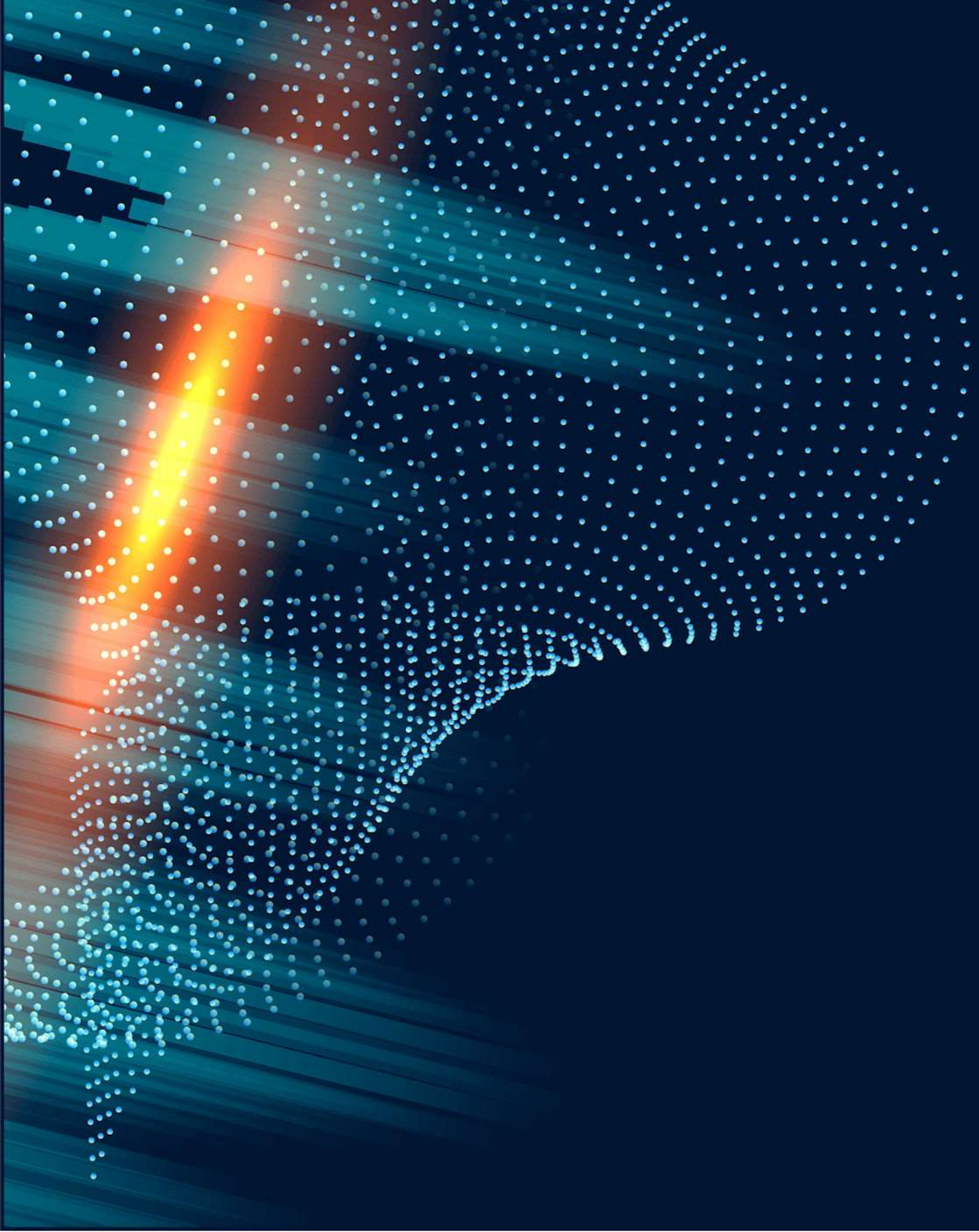
Digital Design

Digital electronics continuation….



Combinational Circuits

Combinational Circuits

- A combinational circuit is a type of digital circuit where the output at any time depends only on the current inputs, not on past inputs or outputs (no memory).
- No Feedback: There is no feedback path in combinational circuits.
- No Clock Signal: These circuits are time-independent and don’t require a clock.
- Output = Function of Input: Outputs change immediately based on input changes.
- Combinational circuits are built using basic logic gates: AND, OR, NOT, NAND, NOR, XOR, and XNOR.
- Combinational circuits are analyzed and designed using Boolean algebra.
- Karnaugh Maps (K-Maps) and Quine-McCluskey methods simplify Boolean expressions to minimize circuits.

Combinational Circuits

Combinational Logic

- Outputs, "at any time", are determined by the input combination
- When input change, output change immediately
  - Real circuits is imperfect and have "propagation delay"
- A combinational circuit
  - Performs logic operations that can be specified by a set of Boolean expressions
  - Can be built hierarchically

Design Procedure

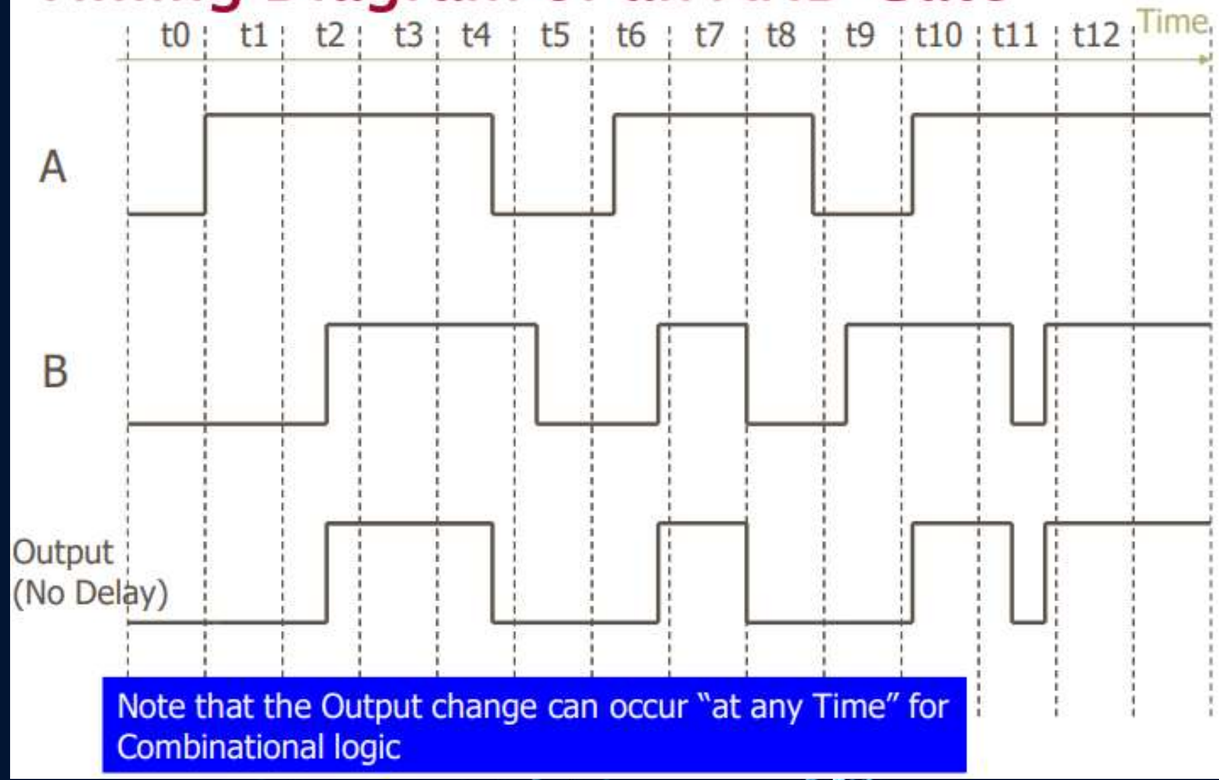
- The design of a combinational circuit involves the following steps:
  - Specification: How the circuit operates is clearly expressed
  - Formulation: Derivation of the truth table or the Boolean equations that define the relationship between inputs and outputs
  - Optimization: Algebraic or K-map optimization of the truth table and drawing the corresponding logic diagram
  - Technology Mapping: Tranform the logic diagram to a new diagram using the available implementation technology
  - Verification: Verify the correctness of the final design

Combinational Circuits

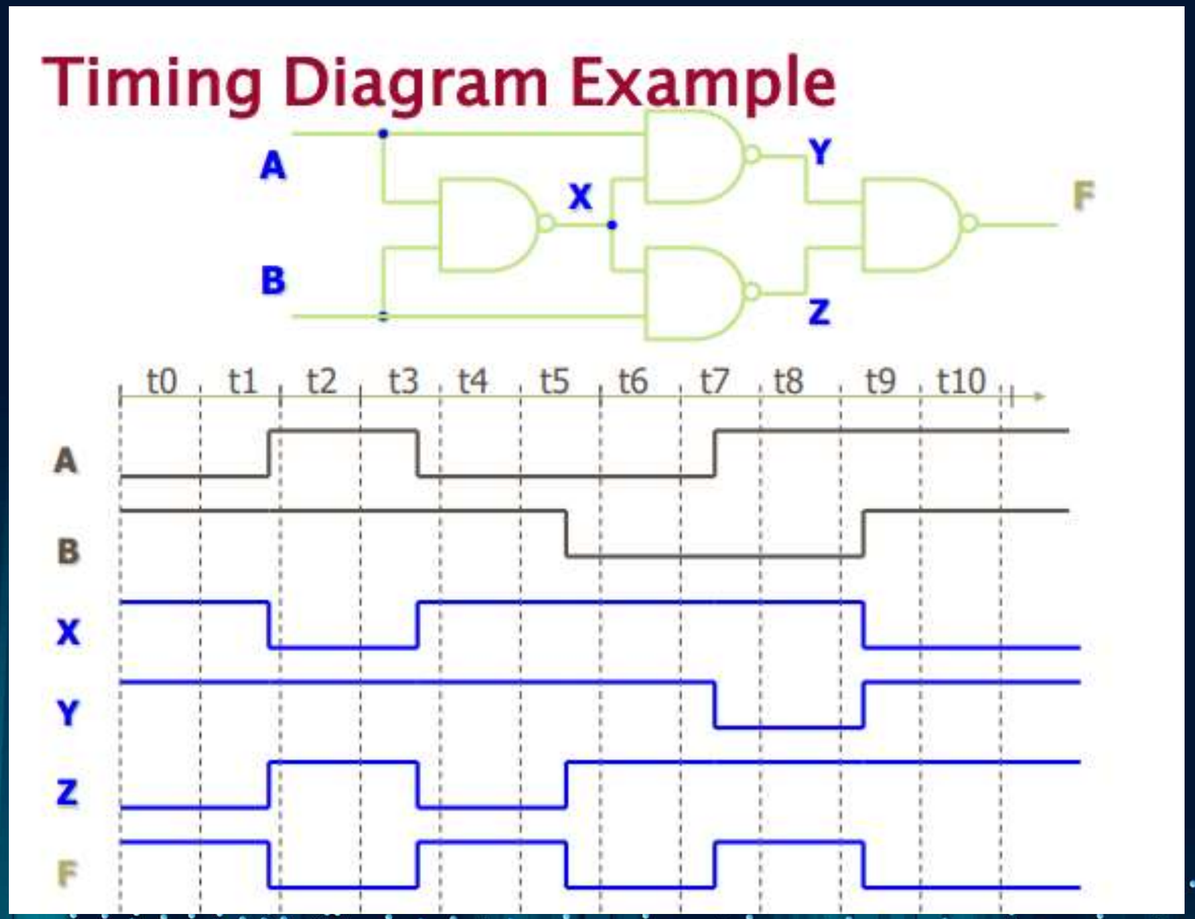
Timing Diagram

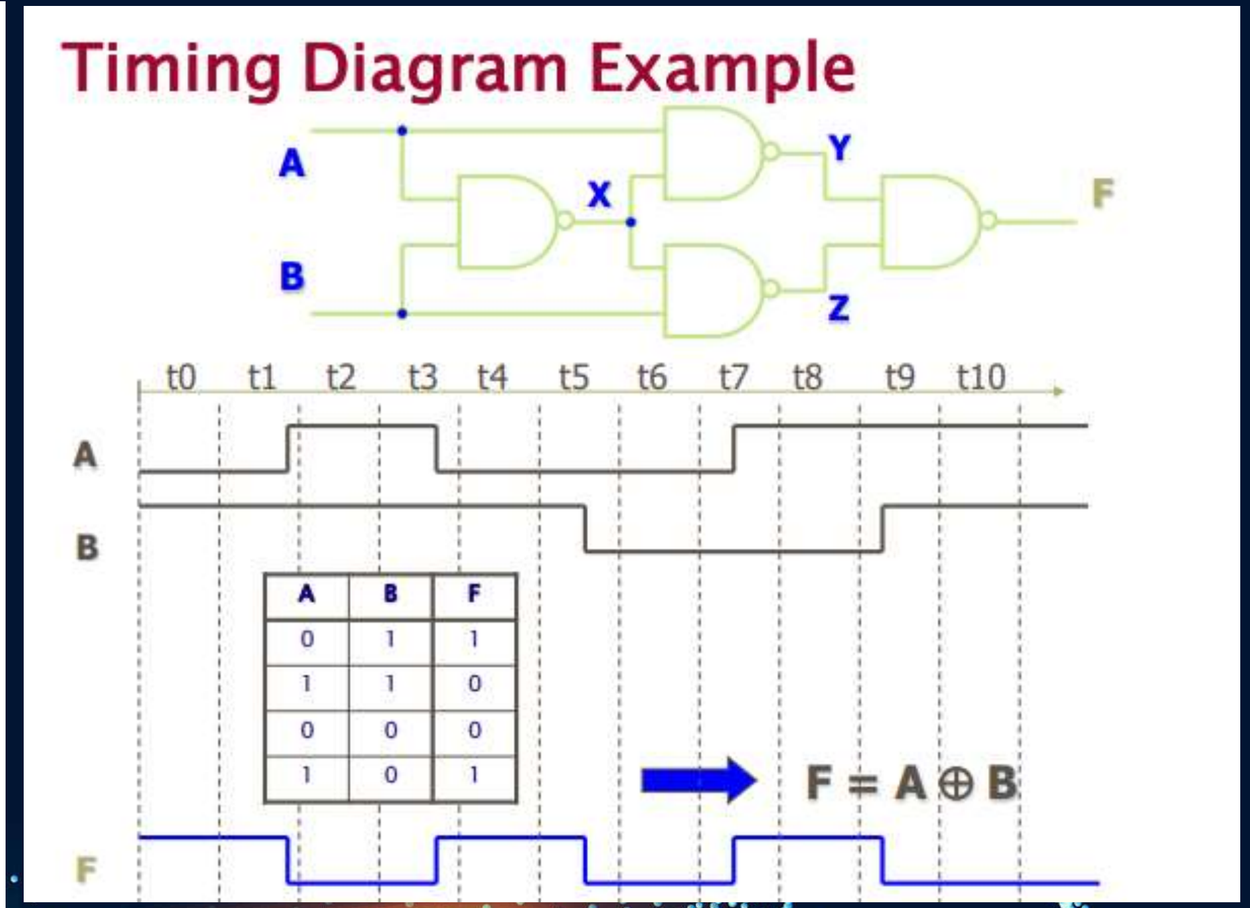
- Describe the functionality of a logic circuit across time
- Represented by a waveform
- For combinational logic, Output is a function of inputs

Timing Diagram of an AND Gate



Combinational Circuits





A	B	F
0	1	1
1	1	0
0	0	0
1	0	1

Combinational Circuits

Combinational Logic

- Outputs, "at any time", are determined by the input combination
- We will discuss
  - Decoders / Encoders
  - Multiplexers / De-Multiplexers
  - Comparators
  - Parity Checkers / Generators
  - Binary Adders / Subtractors
  - Multipliers

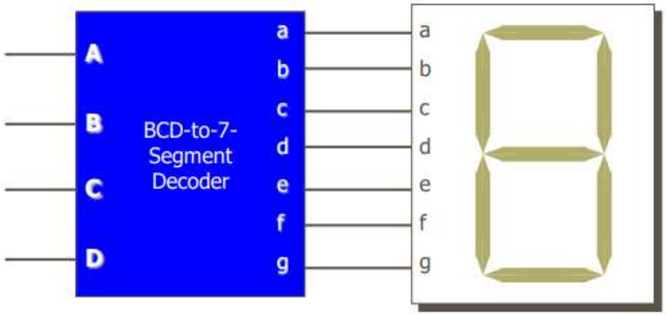
Code Converters

- Convert from one code to another. Examples include
  - BCD to Seven_Segment_Decoder
  - Binary to Excess 3 Code Converter
- Design process:
  - Every bit of the output code is represented as a function of the input code bits, all such functions are reduced and implemented

Combinational Circuits

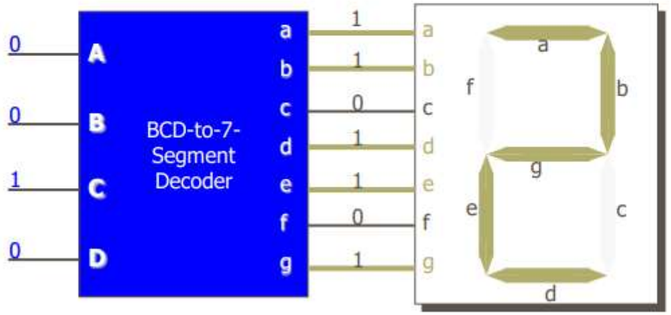
BCD-to-7-Segment Converter

- A kind of decoder



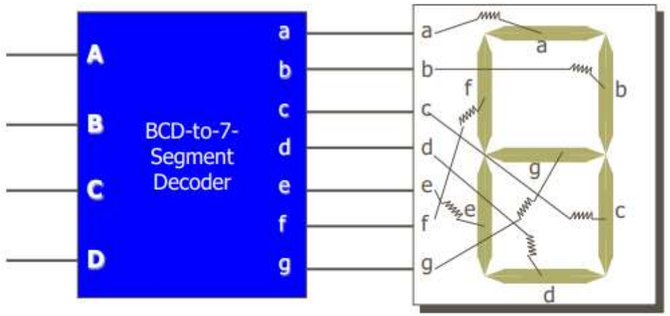
BCD-to-7-Segment Converter

- Decode "2" and show 2



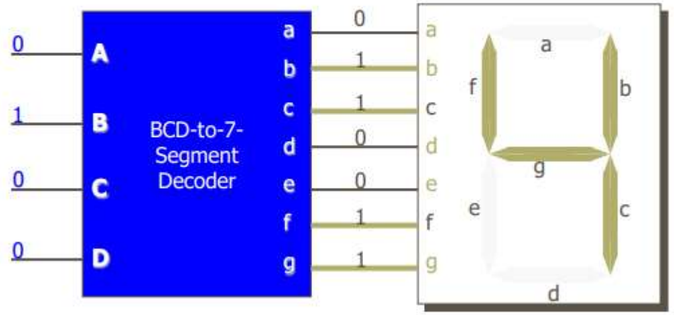
BCD-to-7-Segment Converter

- A kind of decoder



BCD-to-7-Segment Converter

- Decode "4" and show 4



Combinational Circuits

BCD-to-7-Seg. Converter Truth Table

	A	B	C	D	a	b	c	d	e	f	g	
0	0	0	0	0	1	1	1	1	1	1	0	0
1	0	0	0	1	0	1	1	0	0	0	0	1
2	0	0	1	0	1	1	0	1	1	0	1	2
3	0	0	1	1	1	1	1	1	0	0	1	3
4	0	1	0	0	0	1	1	0	0	1	1	4
5	0	1	0	1	1	0	1	1	0	1	1	5
6	0	1	1	0	0	0	1	1	1	1	1	6
7	0	1	1	1	1	1	1	0	0	0	0	7
8	1	0	0	0	1	1	1	1	1	1	1	8
9	1	0	0	1	1	1	1	0	0	1	1	9
>10	All other inputs				0	0	0	0	0	0	0	

Combinational Circuits

Design Each Output Individually —"a"

	A	B	C	D	a
0	0	0	0	0	1
1	0	0	0	1	0
2	0	0	1	0	1
3	0	0	1	1	1
4	0	1	0	0	0
5	0	1	0	1	1
6	0	1	1	0	0
7	0	1	1	1	1
8	1	0	0	0	1
9	1	0	0	1	1
>10	All other inputs				0

AB \ CD	00	01	11	10
00	1	0	1	1
01	0	1	1	0
11	0	0	0	0
10	1	1	0	0

$$a = \overline{A}\,\overline{B}\,\overline{D} + \overline{A}CD + \overline{A}BD + A\overline{B}\,\overline{C}$$

Design Each Output Individually —"b"

	A	B	C	D	b
0	0	0	0	0	1
1	0	0	0	1	1
2	0	0	1	0	1
3	0	0	1	1	1
4	0	1	0	0	1
5	0	1	0	1	0
6	0	1	1	0	0
7	0	1	1	1	1
8	1	0	0	0	1
9	1	0	0	1	1
>10	All other inputs				0

AB \ CD	00	01	11	10
00	1	1	1	1
01	1	0	1	0
11	0	0	0	0
10	1	1	0	0

$$b = \overline{B}\,\overline{C} + \overline{A}\,\overline{B} + \overline{A}\,\overline{C}\,\overline{D} + \overline{A}CD$$

Combinational Circuits

Binary Decoder: A combinational circuit that takes an n-bit binary input and activates exactly one of its 2^n outputs based on the input combination. It is used for address decoding, demultiplexing, and display systems.

Binary Encoder: A combinational circuit that converts one active input line among 2^n inputs into an n-bit binary code. It simplifies large input systems into compact binary outputs, often used in data compression.

Priority Encoder: A special type of encoder that encodes the highest-priority active input into an n-bit binary code. If multiple inputs are active, the one with the highest priority is selected, commonly used in interrupt systems and resource allocation.

Combinational Circuits

Decoders

- n inputs-to-2^n outputs
  - –Each state corresponds to a unique input combination

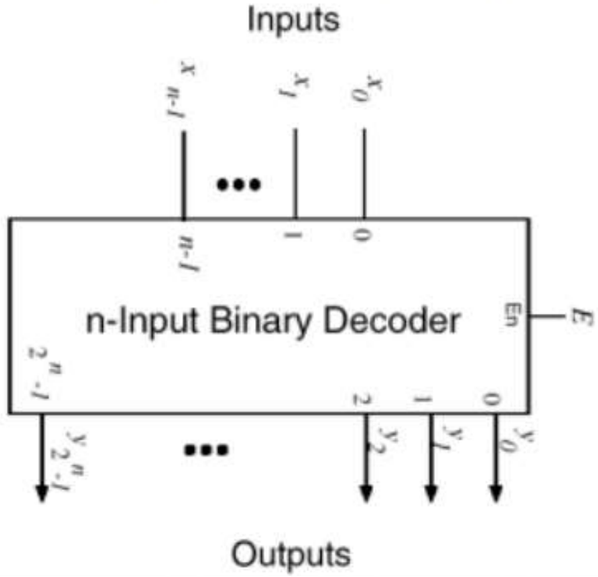
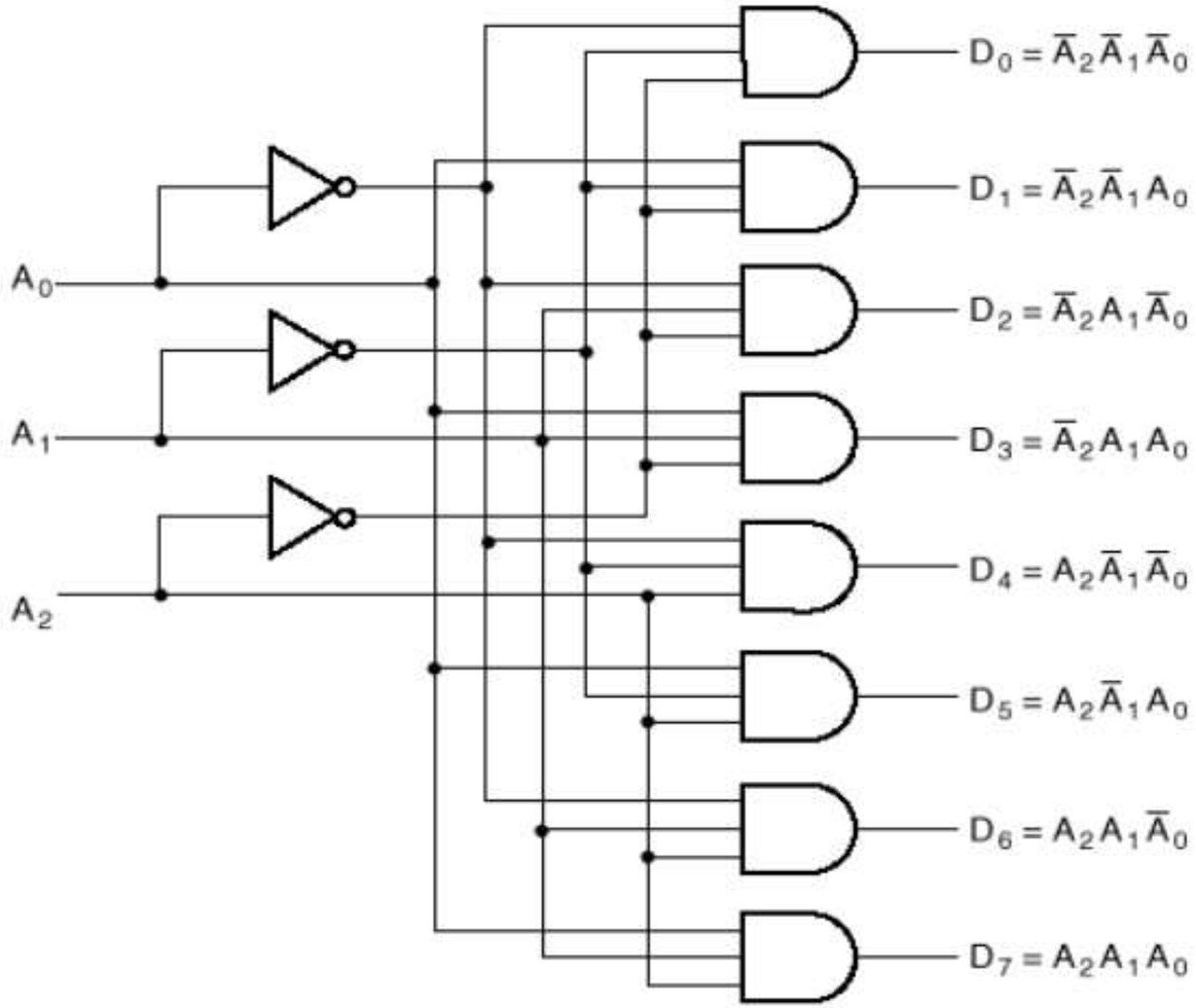


TABLE 3-4
Truth Table for 3–to–8-Line Decoder

Inputs			Outputs							
A_2	A_1	A_0	D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0
0	0	0	0	0	0	0	0	0	0	1
0	0	1	0	0	0	0	0	0	1	0
0	1	0	0	0	0	0	0	1	0	0
0	1	1	0	0	0	0	1	0	0	0
1	0	0	0	0	0	1	0	0	0	0
1	0	1	0	0	1	0	0	0	0	0
1	1	0	0	1	0	0	0	0	0	0
1	1	1	1	0	0	0	0	0	0	0

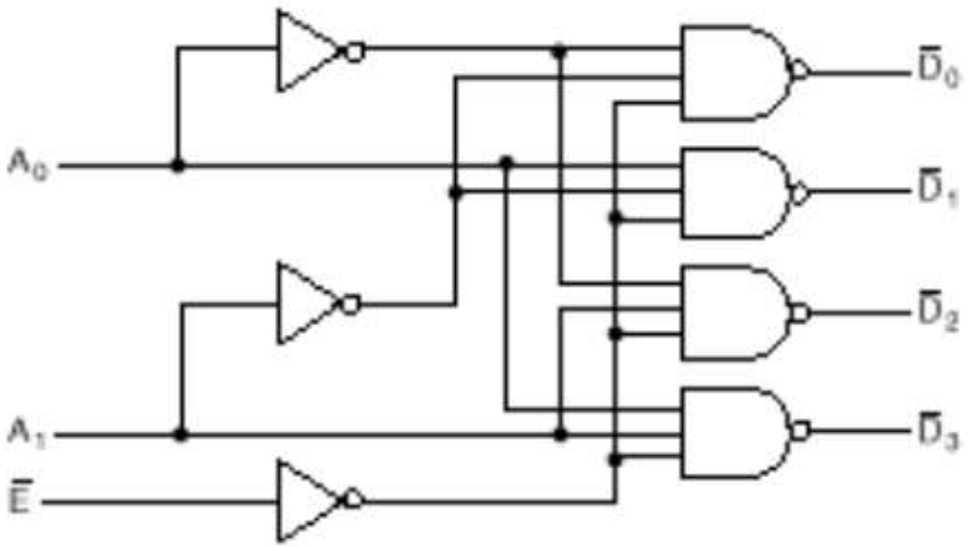
Combinational Circuits

3-to-8 Line Decoder



Combinational Circuits

A 2-to-4-Line Decoder



(a) Logic diagram

$\overline{E}$	A_1	A_0	$\overline{D}_0$	$\overline{D}_1$	$\overline{D}_2$	$\overline{D}_3$
0	0	0	0	1	1	1
0	0	1	1	0	1	1
0	1	0	1	1	0	1
0	1	1	1	1	1	0
1	X	X	1	1	1	1

(b) Truth table

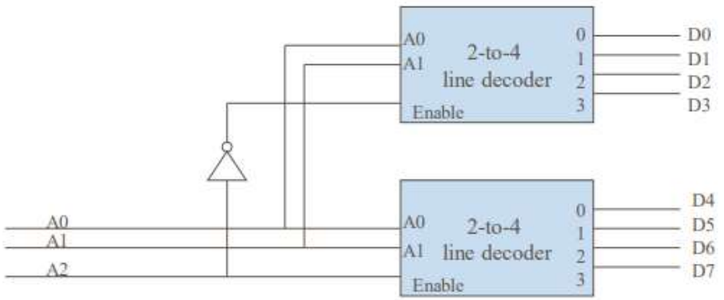
$$\overline{D}_0 = \overline{E\ \overline{A}_1\ \overline{A}_0}$$
$$\overline{D}_1 = \overline{E\ \overline{A}_1\ A_0}$$
$$\overline{D}_2 = \overline{E\ A_1\ \overline{A}_0}$$
$$\overline{D}_3 = \overline{E\ A_1\ A_0}$$

(c) Logic Equations

Combinational Circuits

Large Sized Decoders

- It is possible to connect smaller sized decoders to form a larger decoder



4-input Tree Decoder

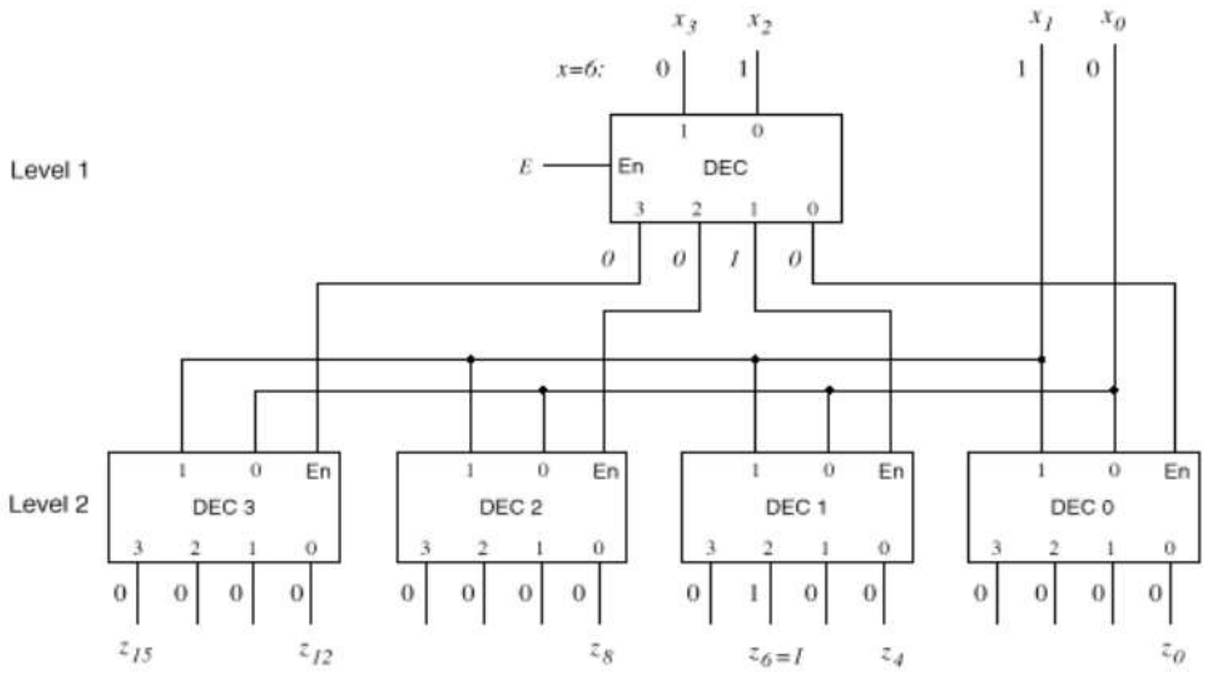
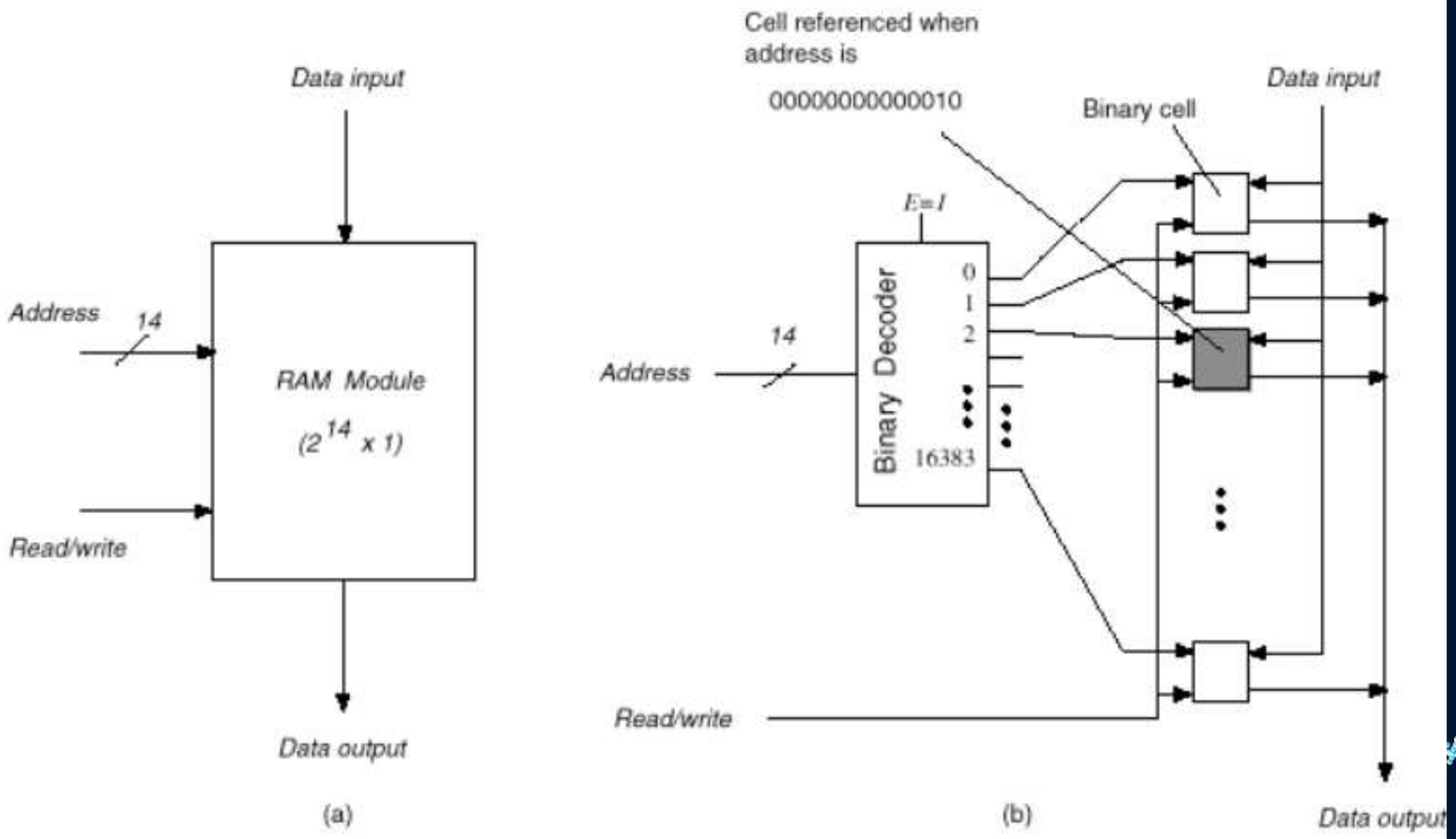


Figure 9.8. 4-input tree decoder

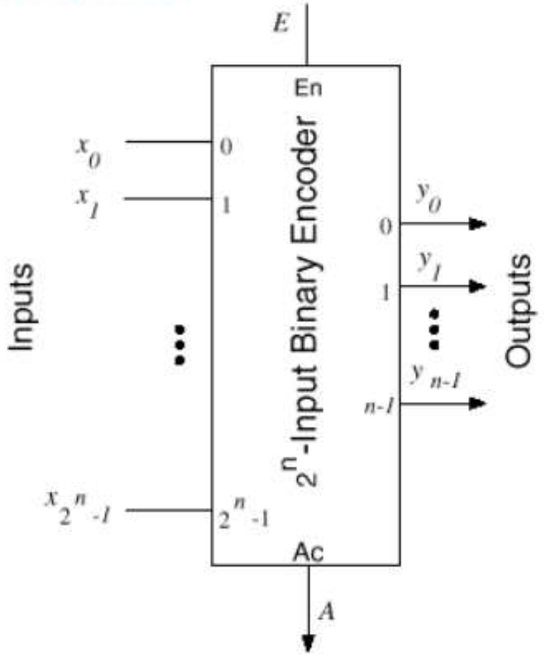
Combinational Circuits

Decoder uses



Combinational Circuits

Binary Encoders



Encoders

- Binary: 2^n inputs-to-n outputs
  - It is assumed that only one input is TRUE or ONE at one time: Most of the input combinations are Don't Care
  - Implementation is using OR gates only; one gate is needed for each output

A0 = D1 + D3

A1 = D2 + D3

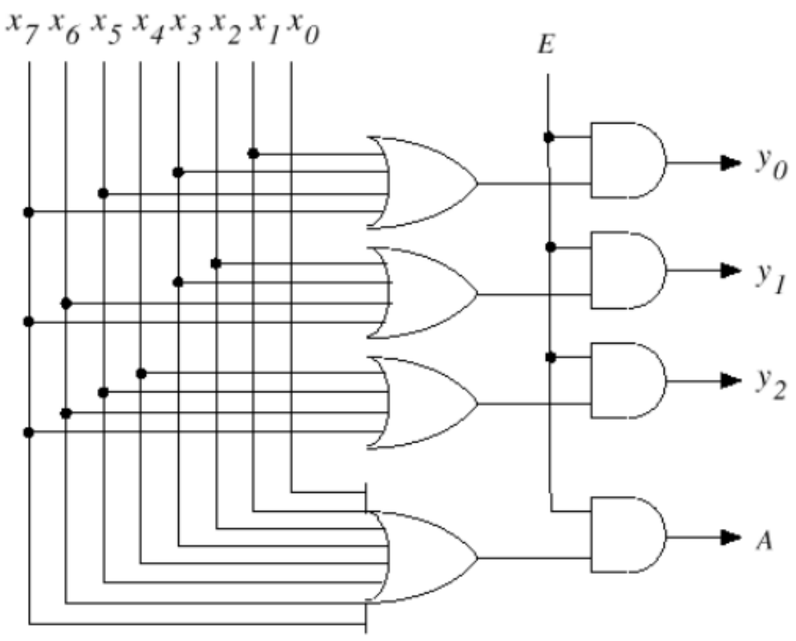
INPUTS				OUTPUTS	
D3	D2	D1	D0	A1	A0
0	0	0	1	0	0
0	0	1	0	0	1
0	1	0	0	1	0
1	0	0	0	1	1

Combinational Circuits

TABLE 3-5
Truth Table for Octal–to–Binary Encoder

Inputs								Outputs		
D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0	A_2	A_1	A_0
0	0	0	0	0	0	0	1	0	0	0
0	0	0	0	0	0	1	0	0	0	1
0	0	0	0	0	1	0	0	0	1	0
0	0	0	0	1	0	0	0	0	1	1
0	0	0	1	0	0	0	0	1	0	0
0	0	1	0	0	0	0	0	1	0	1
0	1	0	0	0	0	0	0	1	1	0
1	0	0	0	0	0	0	0	1	1	1

$$A_0 = D_1 + D_3 + D_5 + D_7$$
$$A_1 = D_2 + D_3 + D_6 + D_7$$
$$A_2 = D_4 + D_5 + D_6 + D_7$$



Combinational Circuits

8-to-3 Encoder

D7	D6	D5	D4	D3	D2	D1	D0	A2	A1	A0
0	0	0	0	0	0	0	1	0	0	0
0	0	0	0	0	0	1	0	0	0	1
0	0	0	0	0	1	0	0	0	1	0
0	0	0	0	1	0	0	0	0	1	1
0	0	0	1	0	0	0	0	1	0	0
0	0	1	0	0	0	0	0	1	0	1
0	1	0	0	0	0	0	0	1	1	0
1	0	0	0	0	0	0	0	1	1	1

Since Dx=1 only in one column at a time

A0 = D1 + D3 + D5 + D7
A1 = D2 + D3 + D6 + D7
A2 = D4 + D5 + D6 + D7

8-to-3 Priority Encoder

D7	D6	D5	D4	D3	D2	D1	D0	A2	A1	A0	Active
0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	1	0	0	0	1
0	0	0	0	0	0	1	X	0	0	1	1
0	0	0	0	0	1	X	X	0	1	0	1
0	0	0	0	1	X	X	X	0	1	1	1
0	0	0	1	X	X	X	X	1	0	0	1
0	0	1	X	X	X	X	X	1	0	1	1
0	1	X	X	X	X	X	X	1	1	0	1
1	X	X	X	X	X	X	X	1	1	1	1

Combinational Circuits

4-to-2 Priority Encoder

D3	D2	D1	D0	A1	A0	Active
0	0	0	0	0	0	0
0	0	0	1	0	0	1
0	0	1	X	0	1	1
0	1	X	X	1	0	1
1	X	X	X	1	1	1

For A1

D3 D2 \ D1 D0	00	01	11	10
00	0	0	0	0
01	1	1	1	1
11	1	1	1	1
10	1	1	1	1

$$A1 = D2 + D3$$

Or using simplification property

$$A1 = D3 + \overline{D3}D2 = D3 + D2$$

4-to-2 Priority Encoder

D3	D2	D1	D0	A1	A0	Active
0	0	0	0	0	0	0
0	0	0	1	0	0	1
0	0	1	X	0	1	1
0	1	X	X	1	0	1
1	X	X	X	1	1	1

For A0

D3 D2 \ D1 D0	00	01	11	10
00	0	0	1	1
01	0	0	0	0
11	1	1	1	1
10	1	1	1	1

$$A0 = D3 + \overline{D2}D1$$

Or using simplification property

$$A0 = D3 + \overline{D3}\,\overline{D2}D1 = D3 + \overline{D2}D1$$

Combinational Circuits

4-to-2 Priority Encoder

D3	D2	D1	D0	A1	A0	Active
0	0	0	0	0	0	0
0	0	0	1	0	0	1
0	0	1	X	0	1	1
0	1	X	X	1	0	1
1	X	X	X	1	1	1

For Active

D3 D2 \ D1 D0	00	01	11	10
00	0	1	1	1
01	1	1	1	1
11	1	1	1	1
10	1	1	1	1

$$\text{Active} = D3 + D2 + D1 + D0$$

8-to-3 Priority Encoder (A2)

D7	D6	D5	D4	D3	D2	D1	D0	A2	A1	A0	Active
0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	1	0	0	0	1
0	0	0	0	0	0	1	X	0	0	1	1
0	0	0	0	0	1	X	X	0	1	0	1
0	0	0	0	1	X	X	X	0	1	1	1
0	0	0	1	X	X	X	X	1	0	0	1
0	0	1	X	X	X	X	X	1	0	1	1
0	1	X	X	X	X	X	X	1	1	0	1
1	X	X	X	X	X	X	X	1	1	1	1

$$A2 = \overline{D7}\,\overline{D6}\,\overline{D5}D4 + \overline{D7}\,\overline{D6}D5 + \overline{D7}D6 + D7$$

$$= \overline{D6}\,\overline{D5}D4 + \overline{D6}D5 + D6 + D7$$

$$= \overline{D5}D4 + D5 + D6 + D7$$

$$= D4 + D5 + D6 + D7$$

Combinational Circuits

4-to-2 Priority Encoder

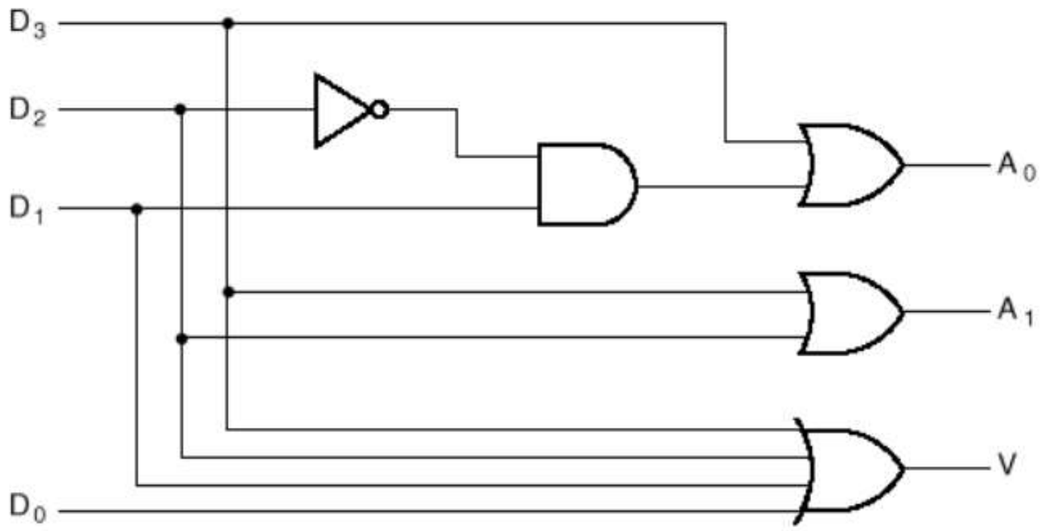
D3	D2	D1	D0	A1	A0	Active
0	0	0	0	0	0	0
0	0	0	1	0	0	1
0	0	1	X	0	1	1
0	1	X	X	1	0	1
1	X	X	X	1	1	1

For Active

D3 D2 \ D1 D0	00	01	11	10
00	0	1	1	1
01	1	1	1	1
11	1	1	1	1
10	1	1	1	1

$$\text{Active} = D3 + D2 + D1 + D0$$

Logic Diagram of a 4-Input Priority Encoder



Combinational Circuits

8-to-3 Priority Encoder (A0)

D7	D6	D5	D4	D3	D2	D1	D0	A2	A1	A0	Active
0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	1	0	0	0	1
0	0	0	0	0	0	1	X	0	0	1	1
0	0	0	0	0	1	X	X	0	1	0	1
0	0	0	0	1	X	X	X	0	1	1	1
0	0	0	1	X	X	X	X	1	0	0	1
0	0	1	X	X	X	X	X	1	0	1	1
0	1	X	X	X	X	X	X	1	1	0	1
1	X	X	X	X	X	X	X	1	1	1	1

$$A0 = \overline{D7}\,\overline{D6}\,\overline{D5}\,\overline{D4}\,\overline{D3}\,\overline{D2}D1 + \overline{D7}\,\overline{D6}\,\overline{D5}\,\overline{D4}D3 + \overline{D7}\,\overline{D6}D5 + D7$$

$$= \overline{D6}\,\overline{D5}\,\overline{D4}\,\overline{D3}\,\overline{D2}D1 + \overline{D6}\,\overline{D5}\,\overline{D4}D3 + \overline{D6}D5 + D7$$

$$= \overline{D6}(\overline{D5}\,\overline{D4}\,\overline{D3}\,\overline{D2}D1 + \overline{D5}\,\overline{D4}D3 + D5) + D7 = \overline{D6}(\overline{D4}\,\overline{D3}\,\overline{D2}D1 + \overline{D4}D3 + D5) + D7$$

$$= \overline{D6}(\overline{D4}(\overline{D3}\,\overline{D2}D1 + D3) + D5) + D7 = \overline{D6}(\overline{D4}(\overline{D2}D1 + D3) + D5) + D7$$

$$= \overline{D6}\,\overline{D4}\,\overline{D2}D1 + \overline{D6}\,\overline{D4}D3 + \overline{D6}D5 + D7$$

8-to-3 Priority Encoder (All)

D7	D6	D5	D4	D3	D2	D1	D0	A2	A1	A0	Active
0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	1	0	0	0	1
0	0	0	0	0	0	1	X	0	0	1	1
0	0	0	0	0	1	X	X	0	1	0	1
0	0	0	0	1	X	X	X	0	1	1	1
0	0	0	1	X	X	X	X	1	0	0	1
0	0	1	X	X	X	X	X	1	0	1	1
0	1	X	X	X	X	X	X	1	1	0	1
1	X	X	X	X	X	X	X	1	1	1	1

$$A2 = D4 + D5 + D6 + D7$$

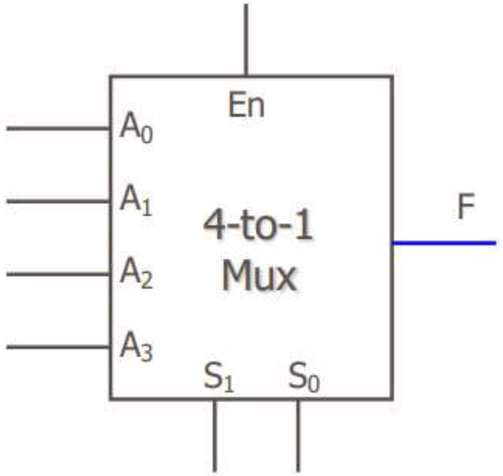
$$A1 = \overline{D5}\,\overline{D4}D2 + \overline{D5}\,\overline{D4}D3 + D7$$

$$A0 = \overline{D6}\,\overline{D4}\,\overline{D2}D1 + \overline{D6}\,\overline{D4}D3 + \overline{D6}D5 + D7$$

$$Active = D1 + D2 + D3 + D4 + D5 + D6 + D7$$

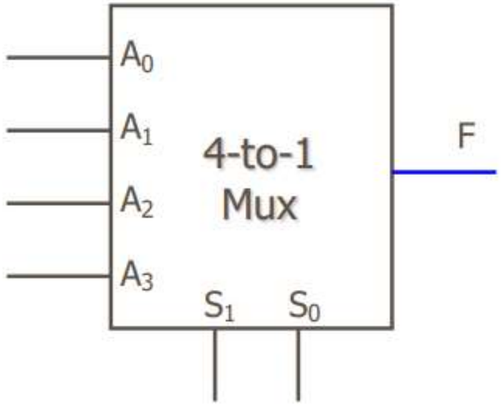
Combinational Circuits

Multiplexers (Mux)



- Functionality: Selection of a particular input
- Route 1 of N inputs (A) to the output F
- En(able) bit can disable the route and set F to 0

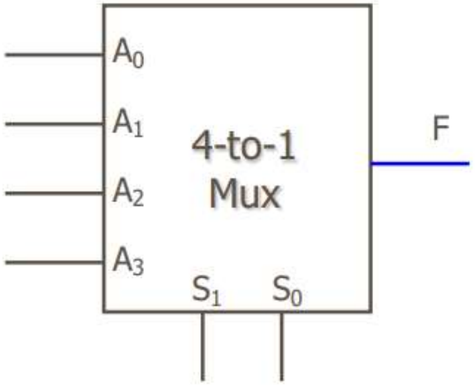
Multiplexers (Mux) w/out Enable



S1	S0	A3	A2	A1	A0	F
0	0	X	X	X	0	0
0	1	X	X	0	X	0
1	0	X	0	X	X	0
1	1	0	X	X	X	0
0	0	X	X	X	1	1
0	1	X	X	1	X	1
1	0	X	1	X	X	1
1	1	1	X	X	X	1

Combinational Circuits

Multiplexers (Mux) w/out Enable

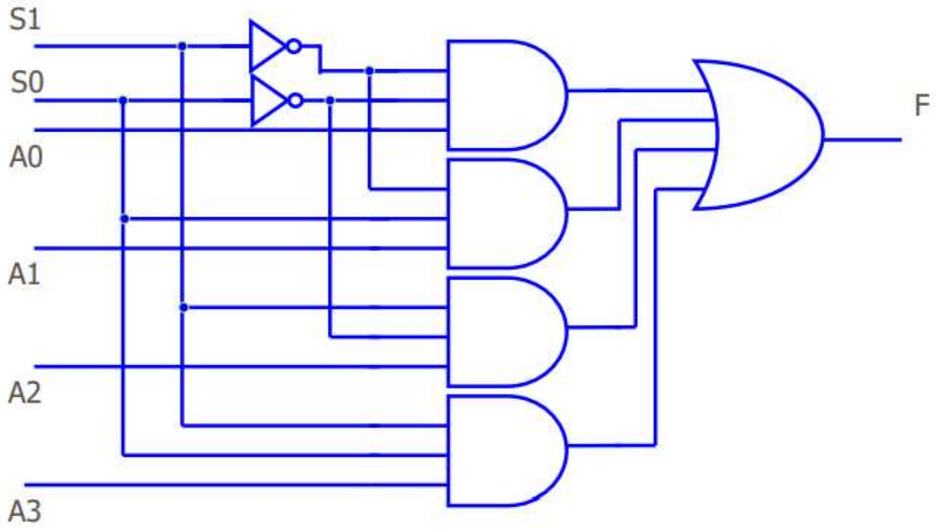


S1	S0	F
0	0	A0
0	1	A1
1	0	A2
1	1	A3

$$F = \overline{S_1}\,\overline{S_0}A_0 + \overline{S_1}S_0A_1 + S_1\overline{S0}A_2 + S_1S_0A_3$$

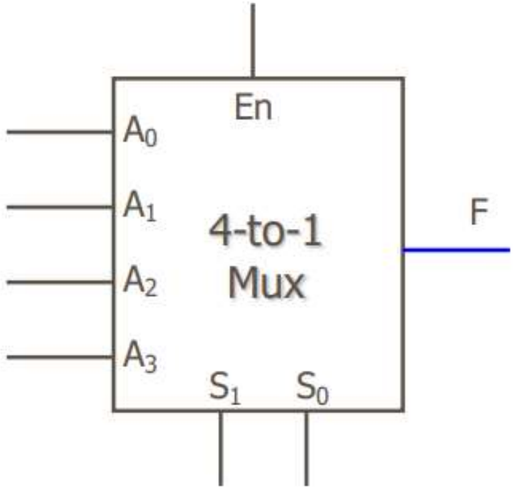
Logic Diagram of a 4-to-1 Mux

$$F = \overline{S_1}\,\overline{S_0}A_0 + \overline{S_1}S_0A_1 + S_1\overline{S0}A_2 + S_1S_0A_3$$



Combinational Circuits

Multiplexers (Mux) w/ Enable



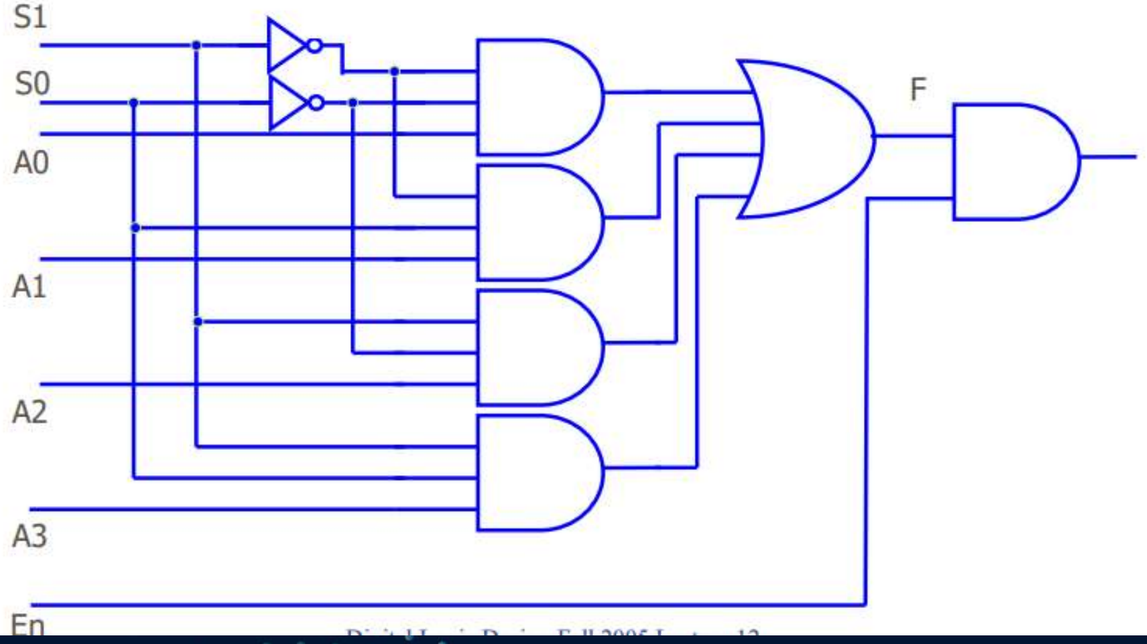
En	S1	S0	F
0	X	X	0
1	0	0	A0
1	0	1	A1
1	1	0	A2
1	1	1	A3

$$F = En \cdot (\overline{S_1}\,\overline{S_0}A_0 + \overline{S_1}S_0A_1 + S_1\overline{S0}A_2 + S_1S_0A_3)$$

$$= En\overline{S_1}\,\overline{S_0}A_0 + En\overline{S_1}S_0A_1 + EnS_1\overline{S0}A_2 + EnS_1S_0A_3$$

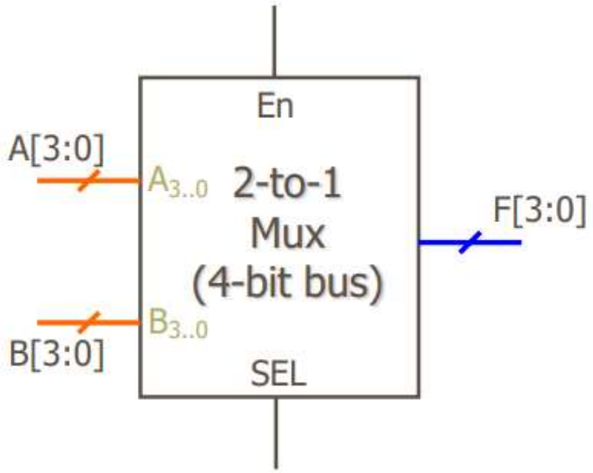
4-to-1 Mux w/ Enable Logic

$$F = En \cdot (\overline{S_1}\,\overline{S_0}A_0 + \overline{S_1}S_0A_1 + S_1\overline{S0}A_2 + S_1S_0A_3)$$



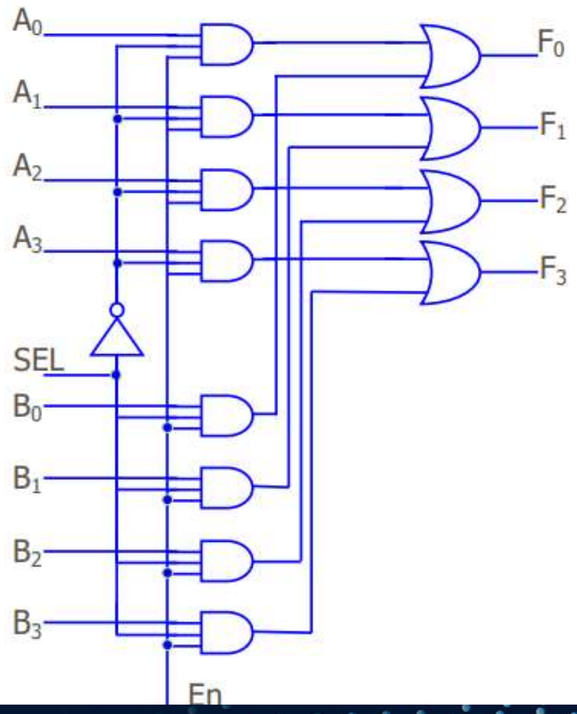
Combinational Circuits

Quadruple 2-to-1 Line Mux



En	SEL	F[3:0]
0	X	0000
1	0	A[3:0]
1	1	B[3:0]

Quadruple 2-to-1 Line Mux

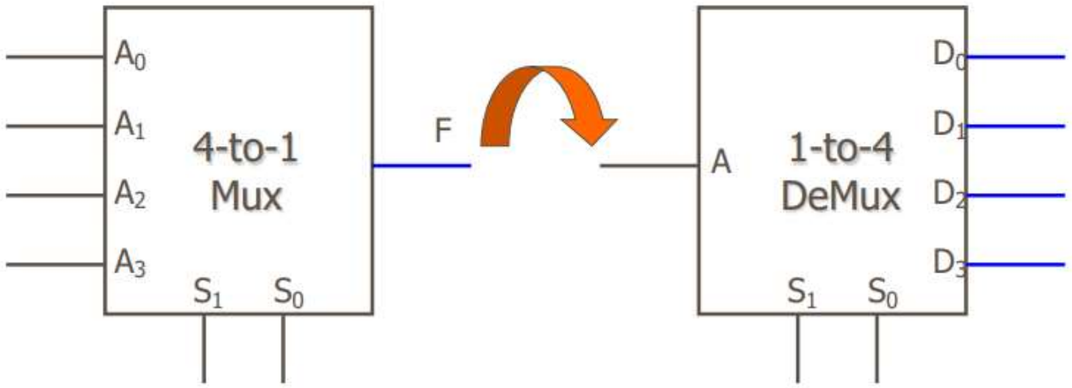


$F_x = A_x \cdot En \cdot \overline{SEL} + B_x \cdot En \cdot SEL$

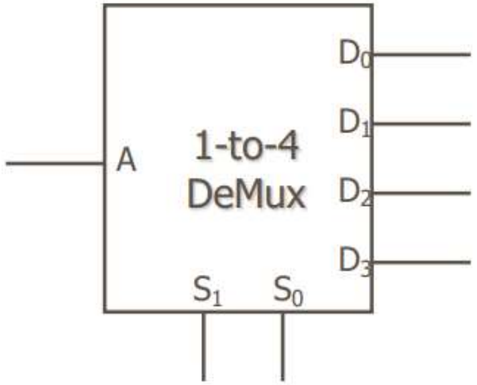
En	SEL	F[3:0]
0	X	0000
1	0	A[3:0]
1	1	B[3:0]

Combinational Circuits

Demultiplexers (DeMux)



DeMux Operations



S1	S0	D3	D2	D1	D0
0	0	0	0	0	A
0	1	0	0	A	0
1	0	0	A	0	0
1	1	A	0	0	0

$$D_0 = \overline{S_1}\,\overline{S_0}A$$

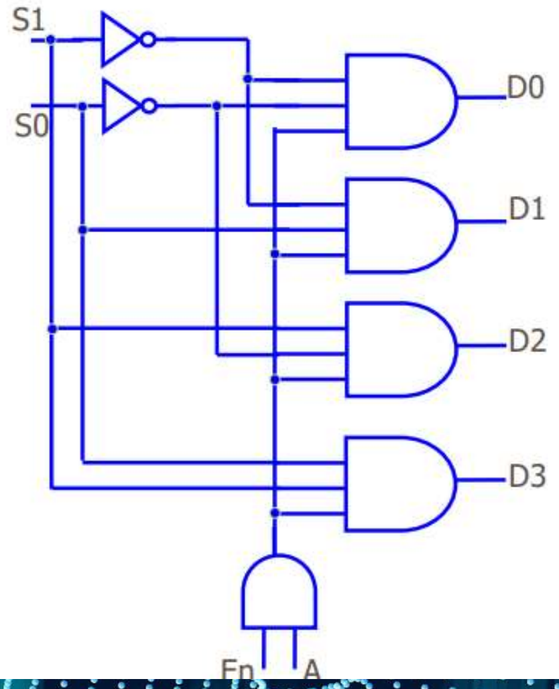
$$D_1 = \overline{S_1}S_0A$$

$$D_2 = S_1\overline{S_0}A$$

$$D_3 = S_1S_0A$$

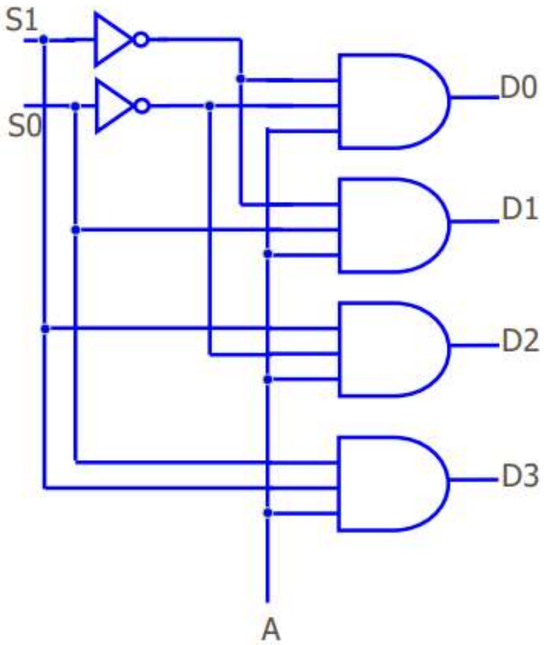
Combinational Circuits

DeMux Operations w/ Enable



En	S1	S0	D3	D2	D1	D0
0	X	X	0	0	0	0
1	0	0	0	0	0	A
1	0	1	0	0	A	0
1	1	0	0	A	0	0
1	1	1	A	0	0	0

DeMux Operations



S1	S0	D3	D2	D1	D0
0	0	0	0	0	A
0	1	0	0	A	0
1	0	0	A	0	0
1	1	A	0	0	0

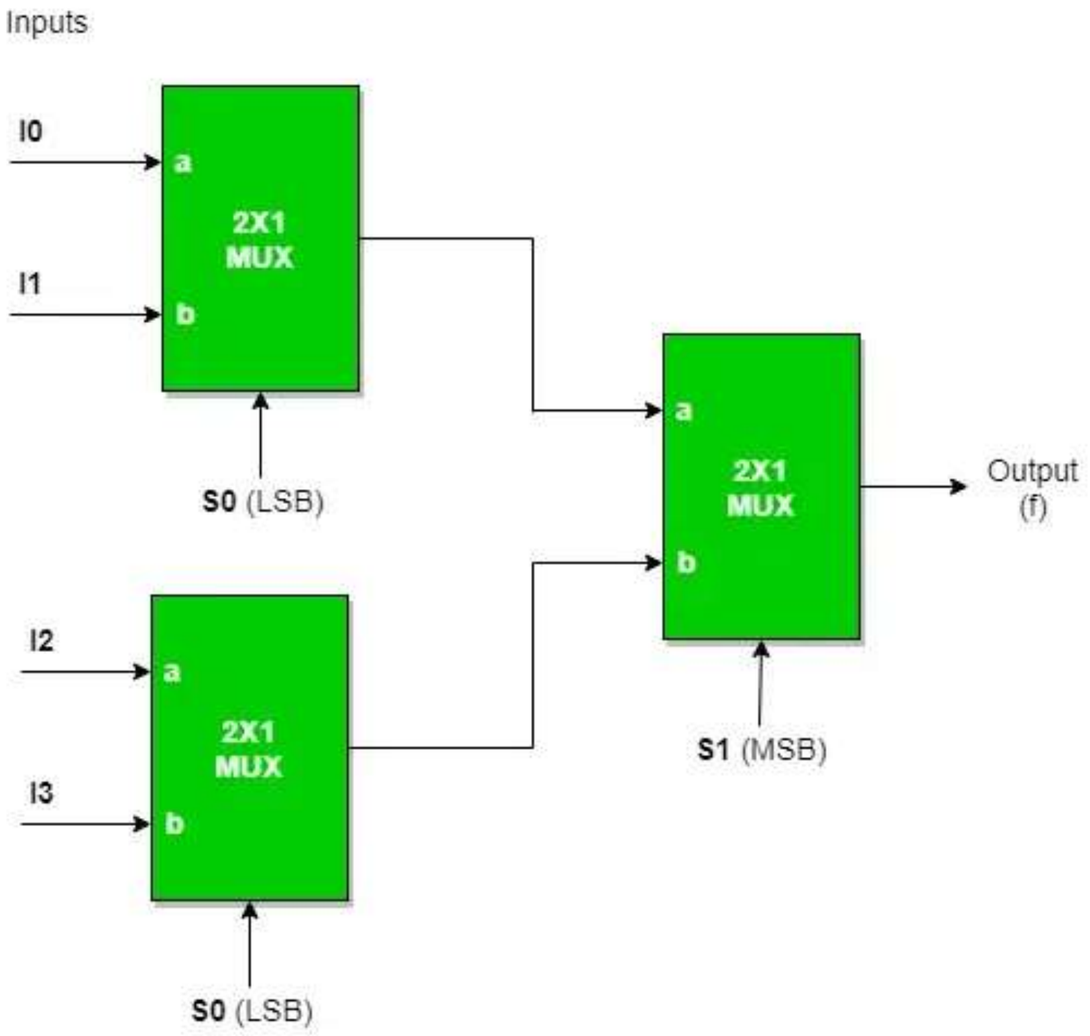
$$D_0 = \overline{S_1}\,\overline{S_0}A$$

$$D_1 = \overline{S_1}S_0A$$

$$D_2 = S_1\overline{S_0}A$$

$$D_3 = S_1S_0A$$

Combinational Circuits

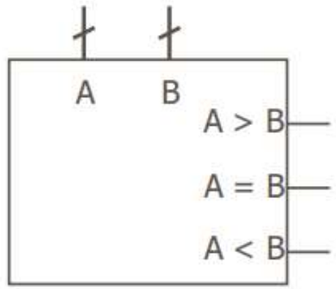


Truth Table

S1	S0	f
0	0	I0
0	1	I1
1	0	I2
1	1	I3

Combinational Circuits

1-bit Magnitude Comparator



A	B	A?B
0	0	A=B
0	1	A<B
1	0	A>B
1	1	A=B

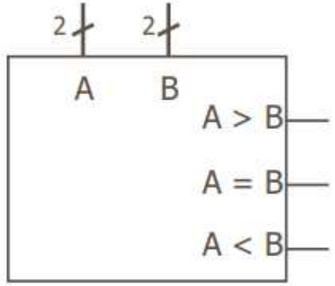
Single bit comparison

$$A > B \rightarrow A\overline{B}$$

$$A < B \rightarrow \overline{A}B$$

$$A = B \rightarrow \overline{A \oplus B}$$

2-bit Magnitude Comparator (unsigned)



$$\begin{array}{c} A_1A_0 \\ B_1B_0 \\ \hline \end{array}$$

Two-bit comparison

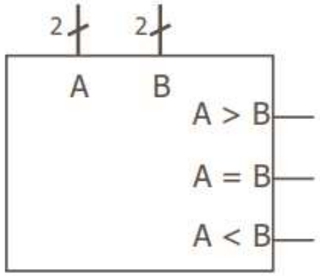
$$(A > B) \rightarrow A_1\overline{B_1} + (\overline{A_1 \oplus B_1})A_0\overline{B_0}$$

$$(A < B) \rightarrow \overline{A_1}B_1 + (\overline{A_1 \oplus B_1})\overline{A_0}B_0$$

$$(A = B) \rightarrow (\overline{A_1 \oplus B_1})(\overline{A_0 \oplus B_0})$$

Combinational Circuits

2-bit Magnitude Comparator (unsigned)



A_1A_0
B_1B_0

Two-bit comparison

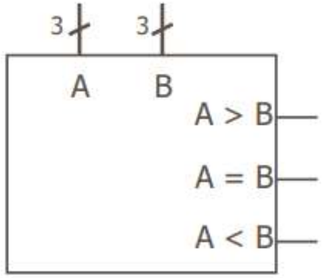
Assign $X_n = \overline{A_n \oplus B_n}$

$(A > B) \rightarrow A_1\overline{B_1} + X_1A_0\overline{B_0}$

$(A < B) \rightarrow \overline{A_1}B_1 + X_1\overline{A_0}B_0$

$(A = B) \rightarrow X_1X_0$

3-bit Magnitude Comparator (unsigned)



$A_2A_1A_0$
$B_2B_1B_0$

Three-bit comparison

Assign $X_n = \overline{A_n \oplus B_n}$

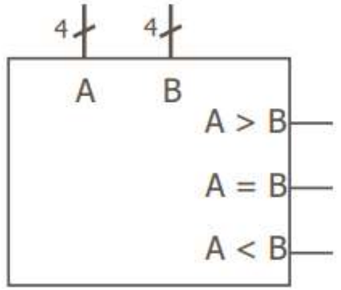
$(A > B) \rightarrow A_2\overline{B_2} + X_2A_1\overline{B_1} + X_2X_1A_0\overline{B_0}$

$(A < B) \rightarrow \overline{A_2}B_2 + X_2\overline{A_1}B_1 + X_2X_1\overline{A_0}B_0$

$(A = B) \rightarrow X_2X_1X_0$

Combinational Circuits

4-bit Magnitude Comparator (unsigned)



$A_3A_2A_1A_0$
$B_3B_2B_1B_0$

Four-bit comparison

$$\text{Assign} \quad X_n = \overline{A_n \oplus B_n}$$

$$(A > B) \rightarrow A_3\overline{B_3} + X_3A_2\overline{B_2} + X_3X_2A_1\overline{B_1} + X_3X_2X_1A_0\overline{B_0}$$

$$(A < B) \rightarrow \overline{A_3}B_3 + X_3\overline{A_2}B_2 + X_3X_2\overline{A_1}B_1 + X_3X_2X_1\overline{A_0}B_0$$

$$(A = B) \rightarrow X_3X_2X_1X_0$$

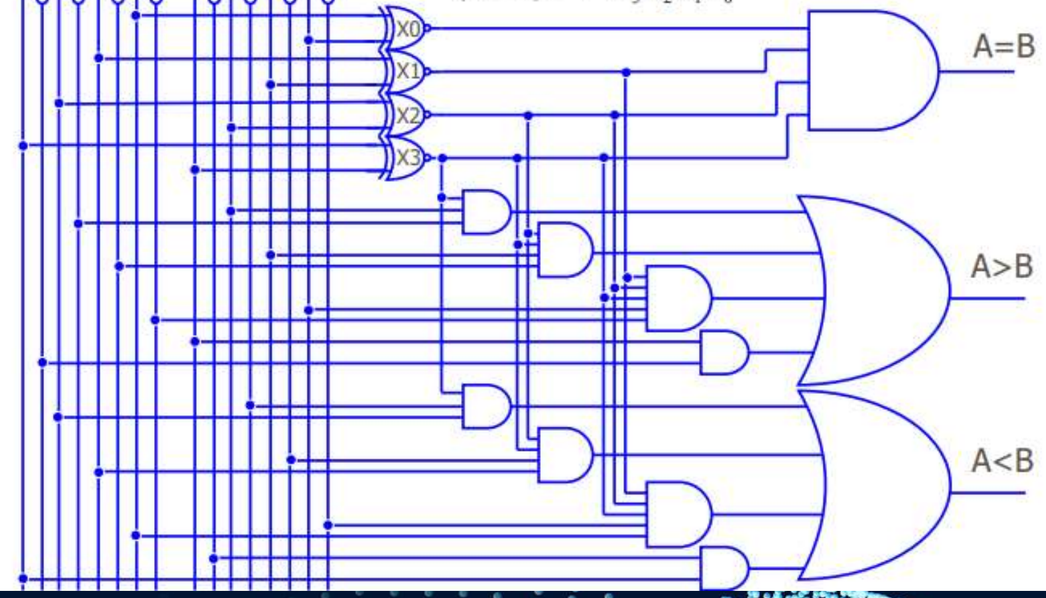
4-bit Magnitude Comparator

$$\text{Assign} \quad X_n = \overline{A_n \oplus B_n}$$

$$(A > B) \rightarrow A_3\overline{B_3} + X_3A_2\overline{B_2} + X_3X_2A_1\overline{B_1} + X_3X_2X_1A_0\overline{B_0}$$

$$(A < B) \rightarrow \overline{A_3}B_3 + X_3\overline{A_2}B_2 + X_3X_2\overline{A_1}B_1 + X_3X_2X_1\overline{A_0}B_0$$

$$(A = B) \rightarrow X_3X_2X_1X_0$$



Combinational Circuits

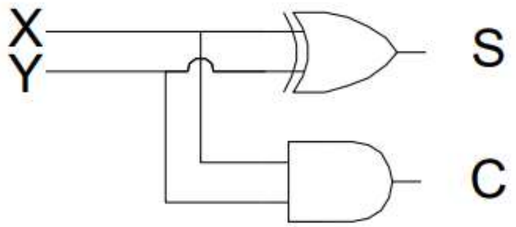
Arithmetic Circuits: Adders

- Half Adder

$S = X'Y + Y'X$

$C = XY$

Inputs		Outputs	
X	Y	C	S
0	0	0	0
0	1	0	1
1	0	0	1
1	1	1	0



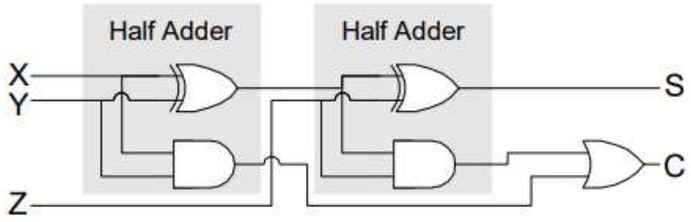
Adders: Full Adder

- Adder

INPUTS			OUTPUTS	
X	Y	Z	C	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

$S = \Sigma m(1,2,4,7)$

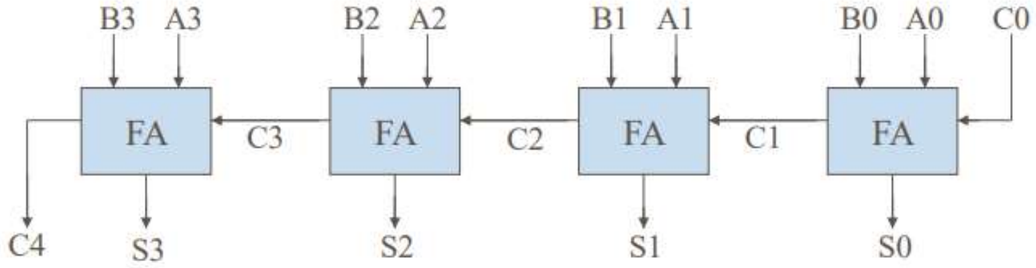
$C = \Sigma m(3,5,6,7)$



Combinational Circuits

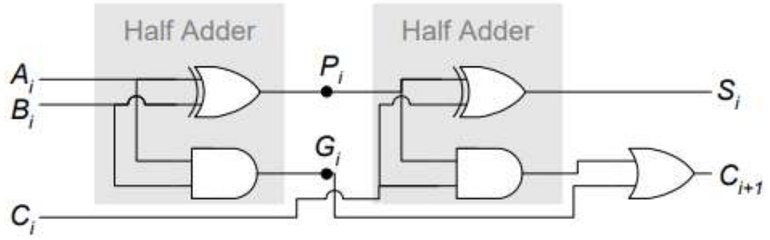
Multi-bit Adders

- Challenge: No of inputs for a 16 bit adder = 32+1
  - Truth Table is not practical; Reduction will take forever
- Solution: Use Single-bit adder and cascade
  - Ripple Carry Adder:



Carry Path Delays and Enhancements

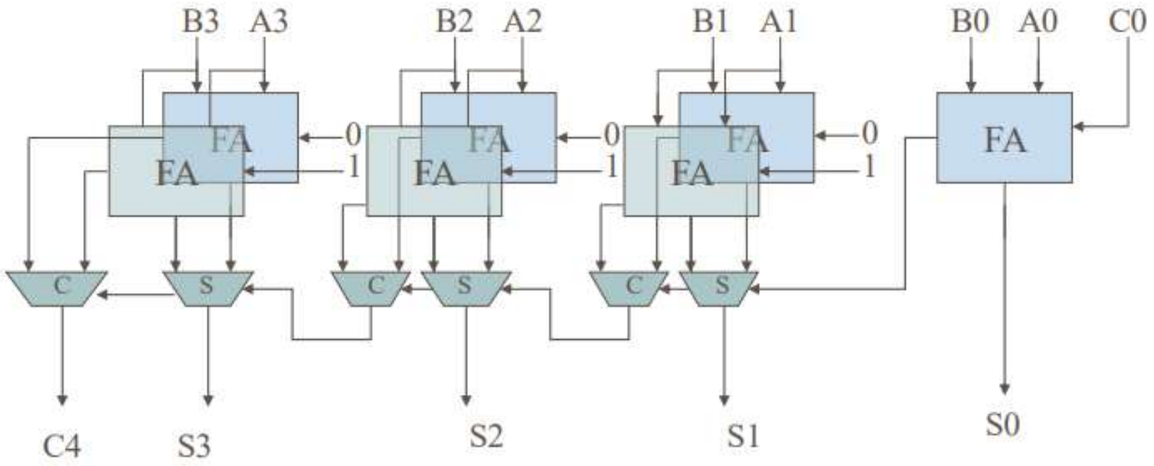
- Carry-path is long: high-speed Adder circuits can not be implemented using ripple carry method
  - No. of Carry Logic Levels for n-bit Addition = $2n + 2$
- Delay optimization can be achieved by Logic Manipulation
  - Define $P_i = A_i \oplus B_i$ & $G_i = A_i.B_i$
  - Then $S_i = P_i \oplus C_i$ & $C_{i+1} = G_i + P_i.C_i$



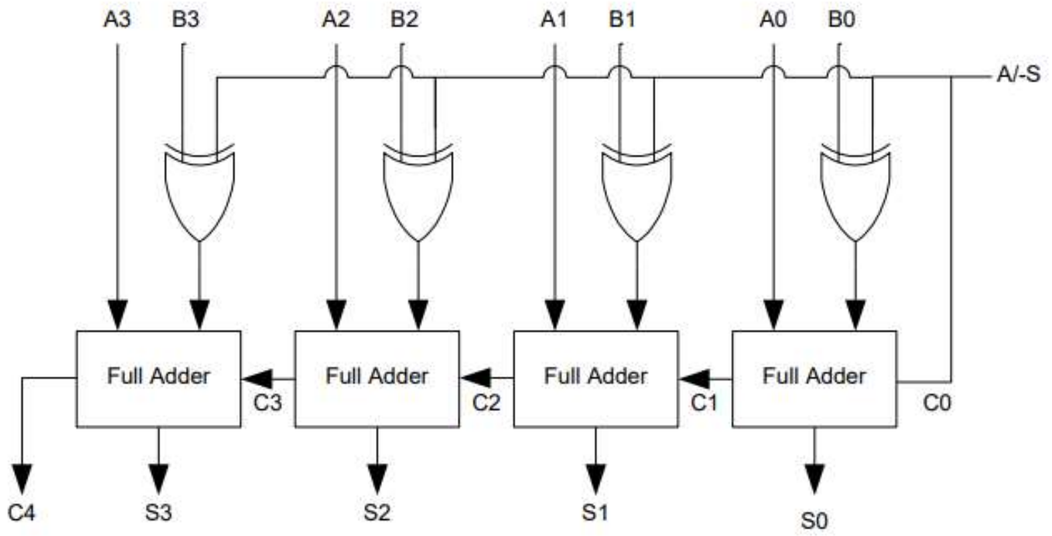
Combinational Circuits

Carry Select Adder

- Evaluate all carry's in parallel and then select the correct one



Binary Adder/Subtractor



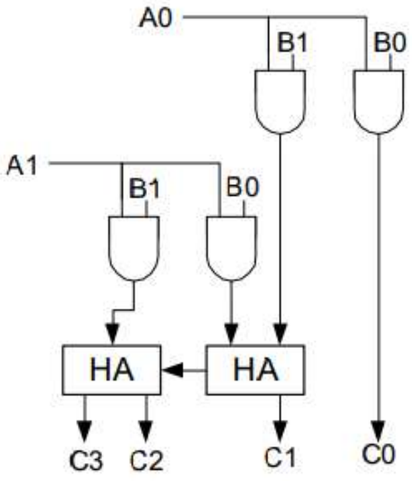
Combinational Circuits

Multipliers

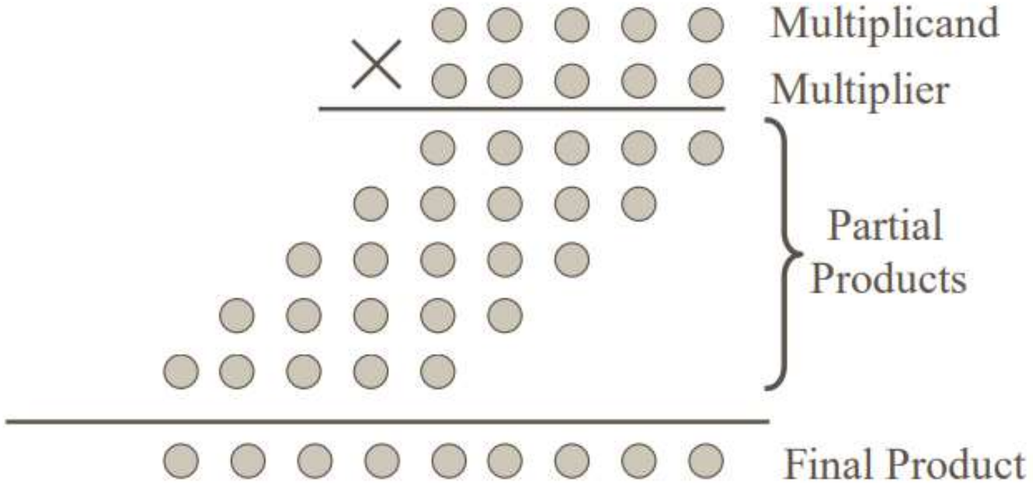
Binary Multiplication

In1	In0	product
0	0	0
0	1	0
1	0	0
1	1	1

		B1	B0
		A1	A0
		A0B1	A0B0
	A1B1	A1B0	
C3	C2	C1	C0

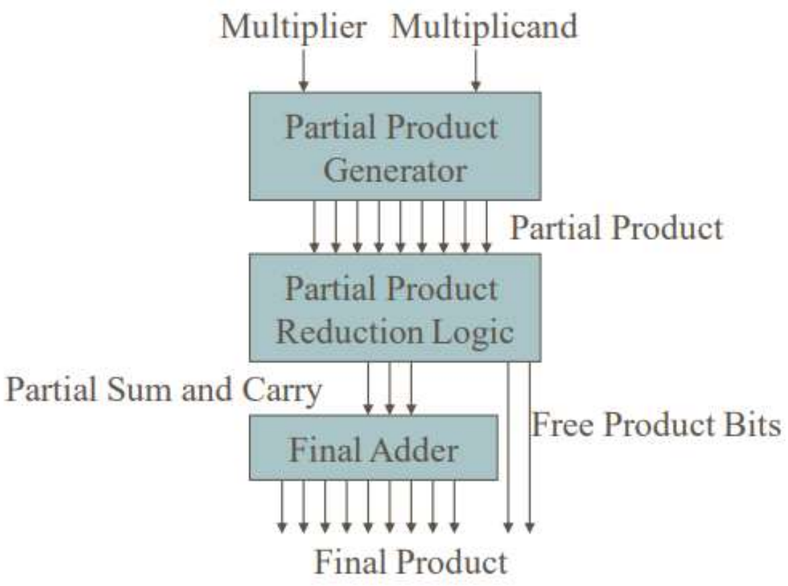


Multi-bit Multiplication



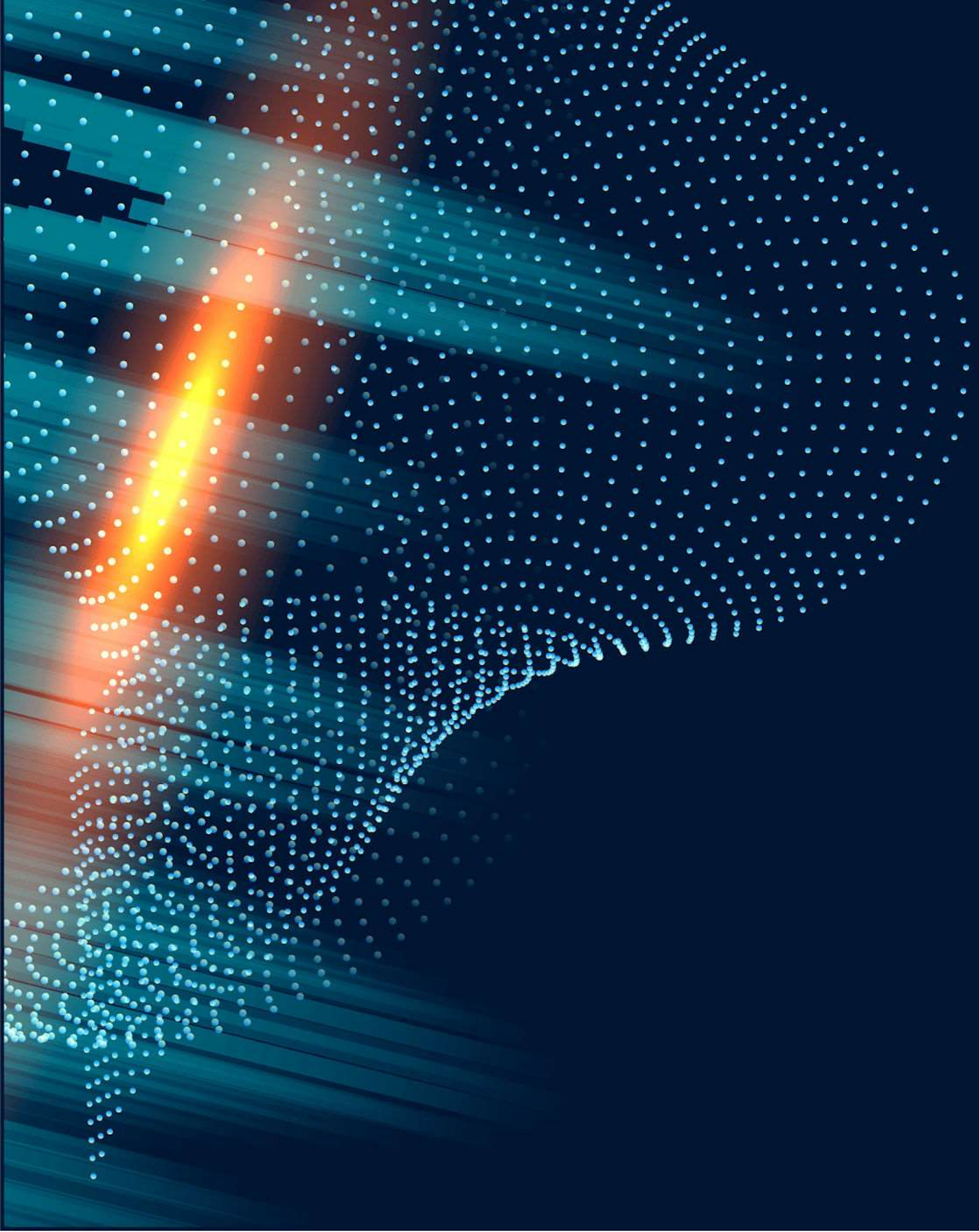
Combinational Circuits

Multi-bit Parallel Multiplier



Parallel Multipliers

- Consist of Three main components
  - Partial product generator:
    - Implemented with AND Gates
  - Partial Product Reduction Logic
    - Many implementation like
      - Carry Save, Dual Carry Save
      - Wallac Tree, Dadda Tree
      - Vuillemin Reduction
  - Final Adder
    - Ripple Carry, Carry Look-ahead or Carry Save



Sequential Circuits

Sequential Circuits

Sequential Logic

- Memory added to Combinational Logic
- Memory elements: Output is held in One of Two Stable States
  - Latches
  - Flip Flops
- Identified by the presence of Feedback
- Output depends on Inputs & State of the Circuit

Sequential Logic Circuits

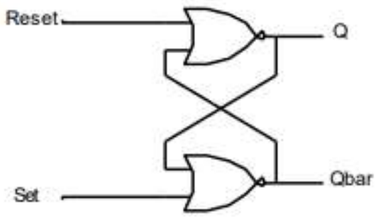
- Classification
  - Synchronous: External timing signal determines output updates
  - Asynchronous: Outputs are updated following changes in state or inputs after propagation delays
- Structural Classification:
  - Moore: Output only derived from Flip Flops
  - Mealy: Output is combinational function of Flip Flops and inputs

Sequential Circuits

Latches

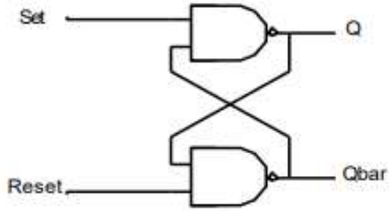
- Binary Storage Elements
  - Asynchronous in nature: No Clock
  - Types include SR and D
  - NOR and NAND implementations possible

Set	Reset	Q	Qbar	
1	0	1	0	Latch in Set State
0	0	1	0	
0	1	0	1	Reset State of the Latch
0	0	0	1	
1	1	0	0	Next State is Undefined



NAND Implementation of Latch

Set	Reset	Q	Qbar	
0	1	1	0	Latch in Set State
1	1	1	0	
1	0	0	1	Reset State of the Latch
1	1	0	1	
0	0	1	1	Next State is Undefined

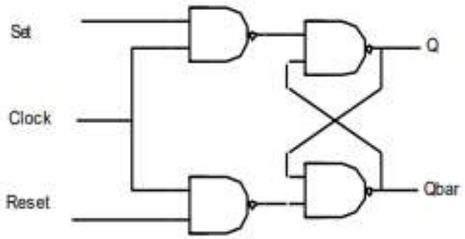


Sequential Circuits

The *Clocked* Latch

- A control input (called Clock) is added
  - Output changes only when this input is HIGH
  - Allows more control over the time when state changes are required

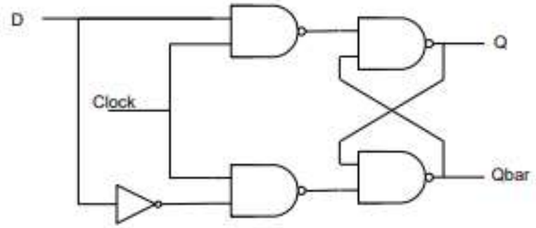
Clock	Set	Reset	Q	Qbar	
0	X	X	1	0	No change: Previous state held
1	0	0	1	0	
1	0	1	0	1	Reset State
1	1	0	1	0	Set State
1	1	1	1	1	Undefined



The *D* Latch

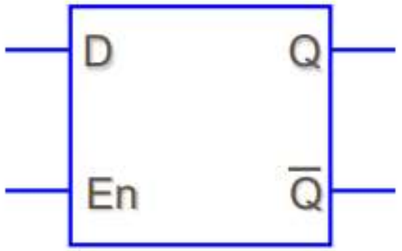
- Avoids ambiguous inputs
- Used as Data (*D*) storage device enabled by the clock
- Also called a Transparent Latch

Clock	D	Q	
0	X	0	No change:
1	0	0	Reset State
1	1	1	Set State



Sequential Circuits

D Latch Symbol

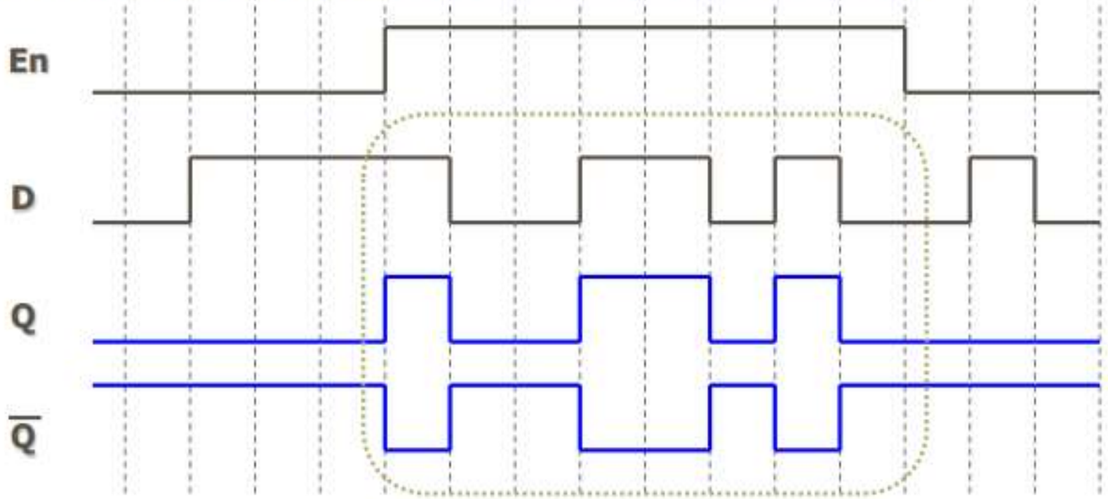


En	D	Q	$\overline{Q}$
0	X	NC	NC
1	0	0	1
1	1	1	0

NC: No Change

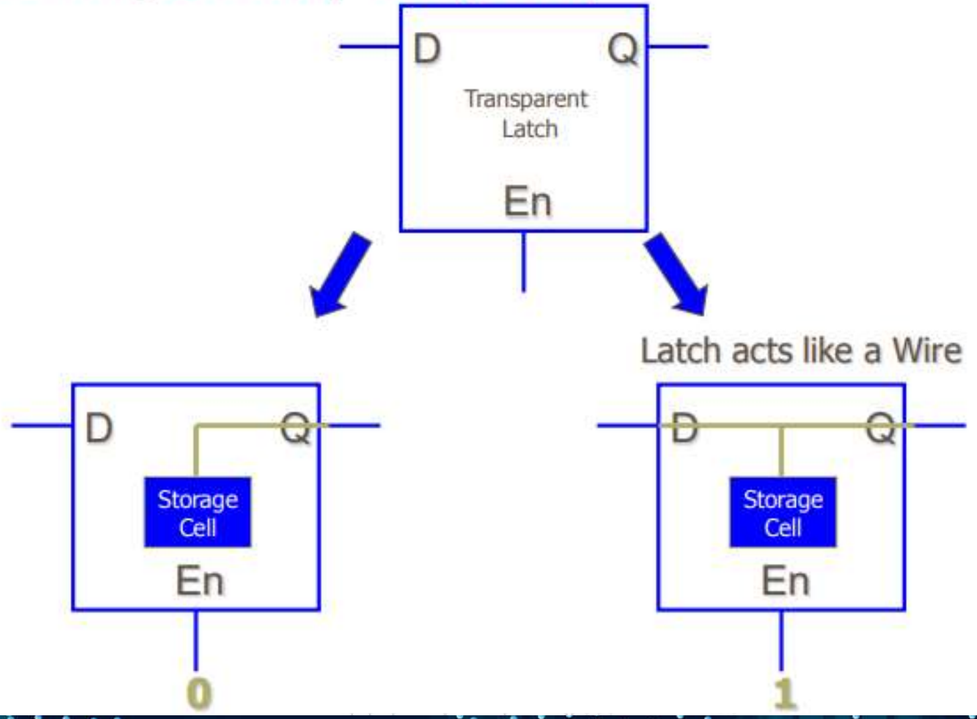
Latch is Transparent

- D Latch is called “transparent” or “level sensitive”
- Output follows input instantaneously

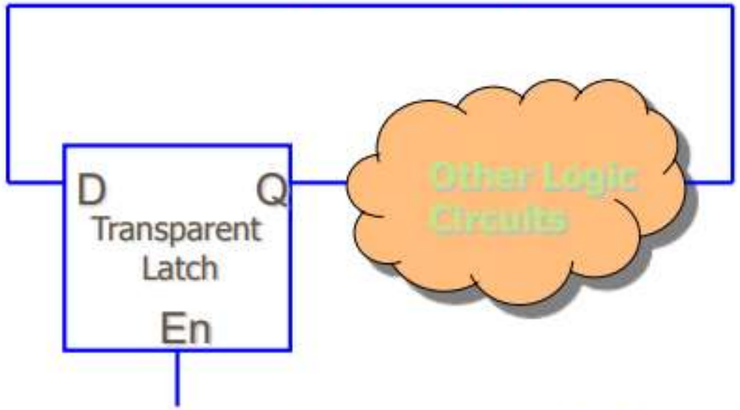


Sequential Circuits

Transparency Property



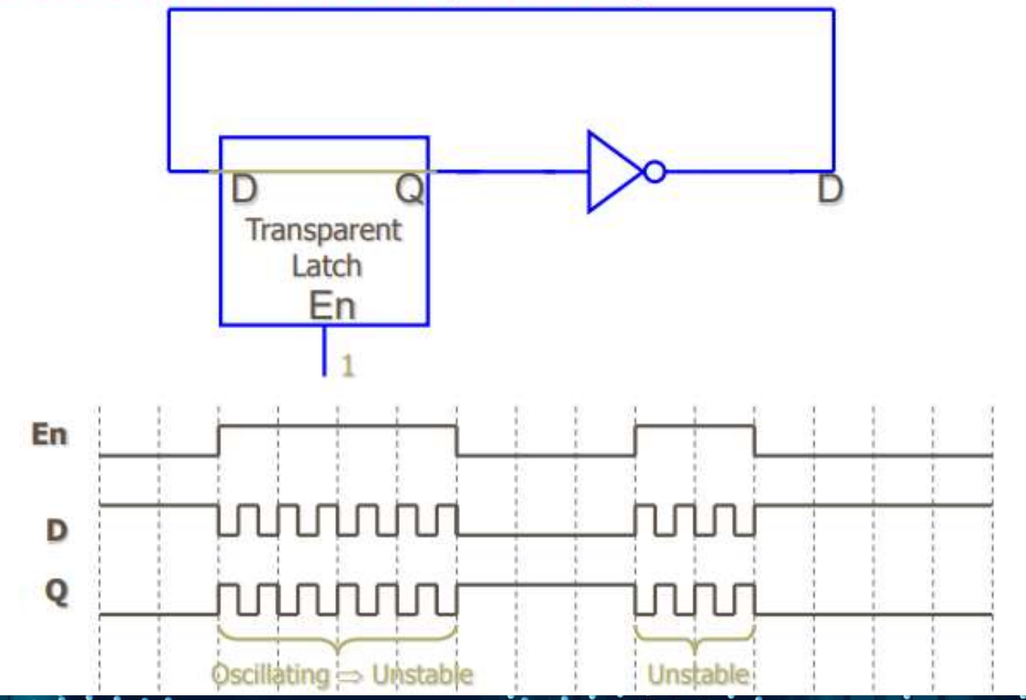
Problem of Transparency



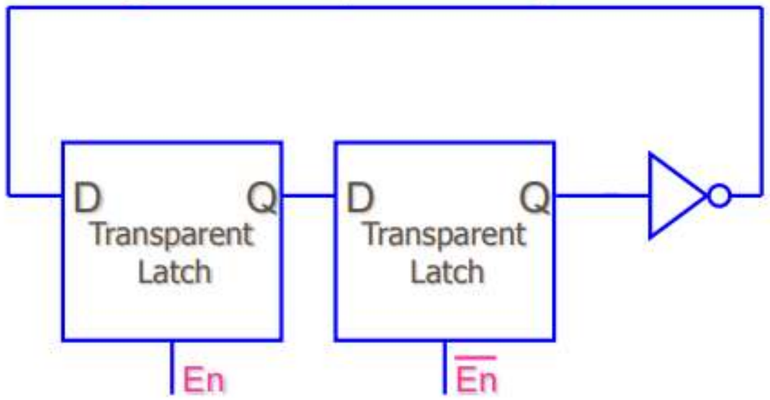
- A momentary input change tunnels through the latch and the entire circuitry
- What problem this can cause?

Sequential Circuits

Problem of Transparency



Eliminating Transparency



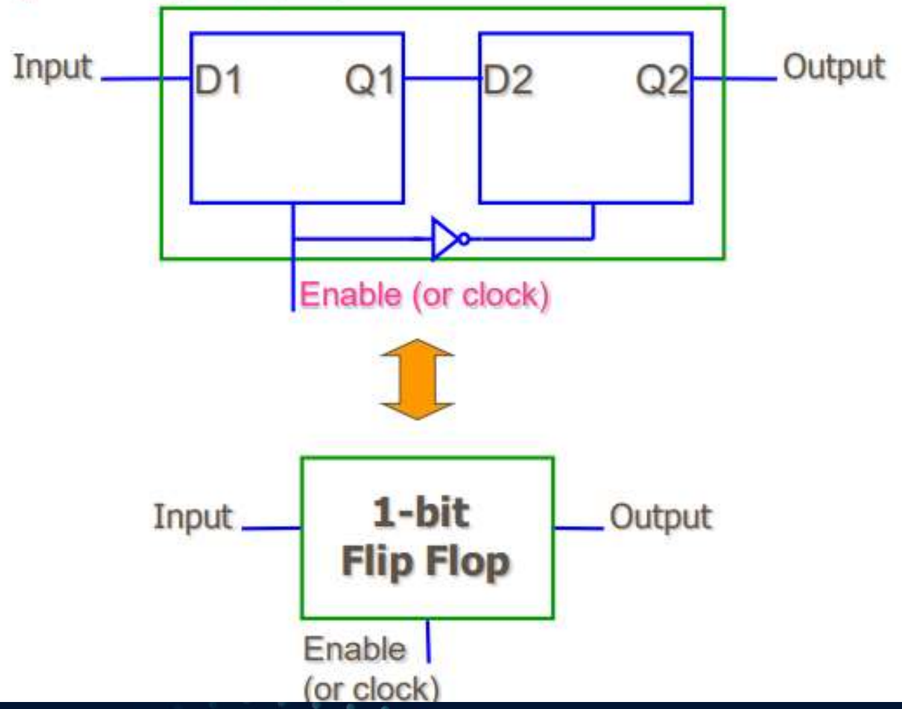
- Separating the input and output, so they are independently controlled
- Only open one gate at a time to avoid tunneling

Sequential Circuits

The Flip Flop

- Two ways to design F/F
- Construction Methods
  - Master Slave
  - Edge-Triggered
- Master Slave F/F
  - Two Cascaded Latches and an Inverter
    - First Latch is enabled during High time of the Clock
    - Second Latch is enabled during Low time of Clock
- Edge Triggered Latch
    - *Outputs can change only at the Transition of Clock*
    - *Hi-to-Low (falling) and Low-to-High (rising) Edge*
- Designed for Synchronous Logic
  - Time Window for Data to Pass to the Output is Extremely Short

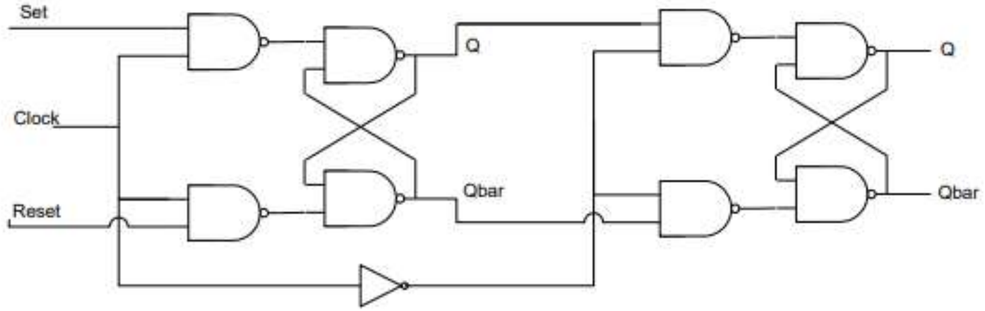
Flip-Flop (F/F)



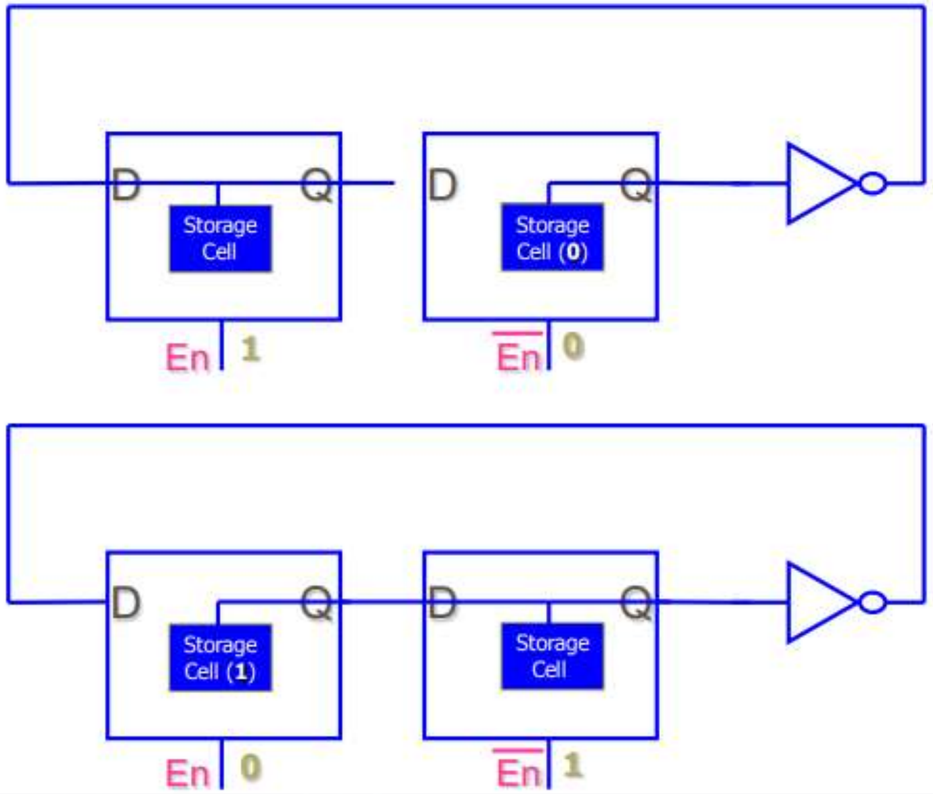
Sequential Circuits

The Master-Slave SR Flip Flop

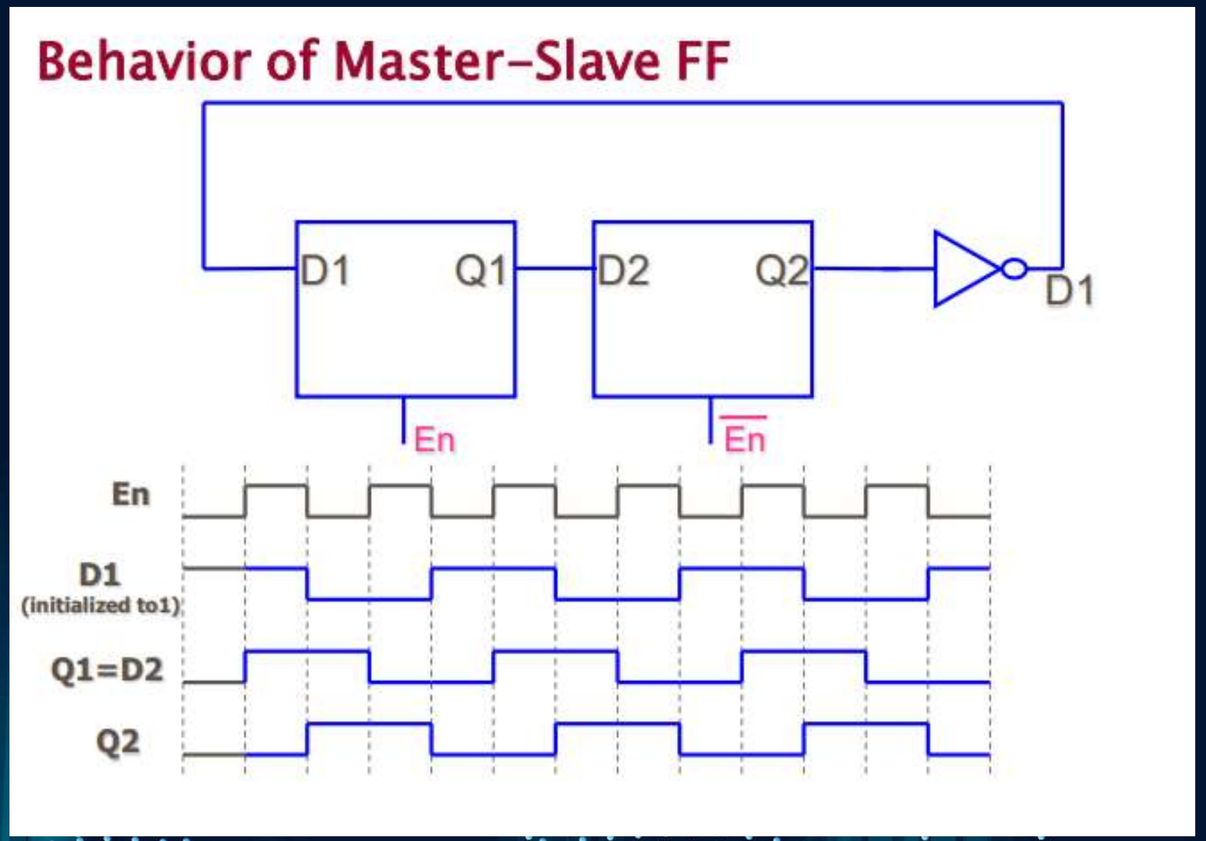
- Two Cascaded Latches and an Inverter
  - First Latch is enabled during High time of the Clock
  - Second Latch is enabled during Low time of Clock

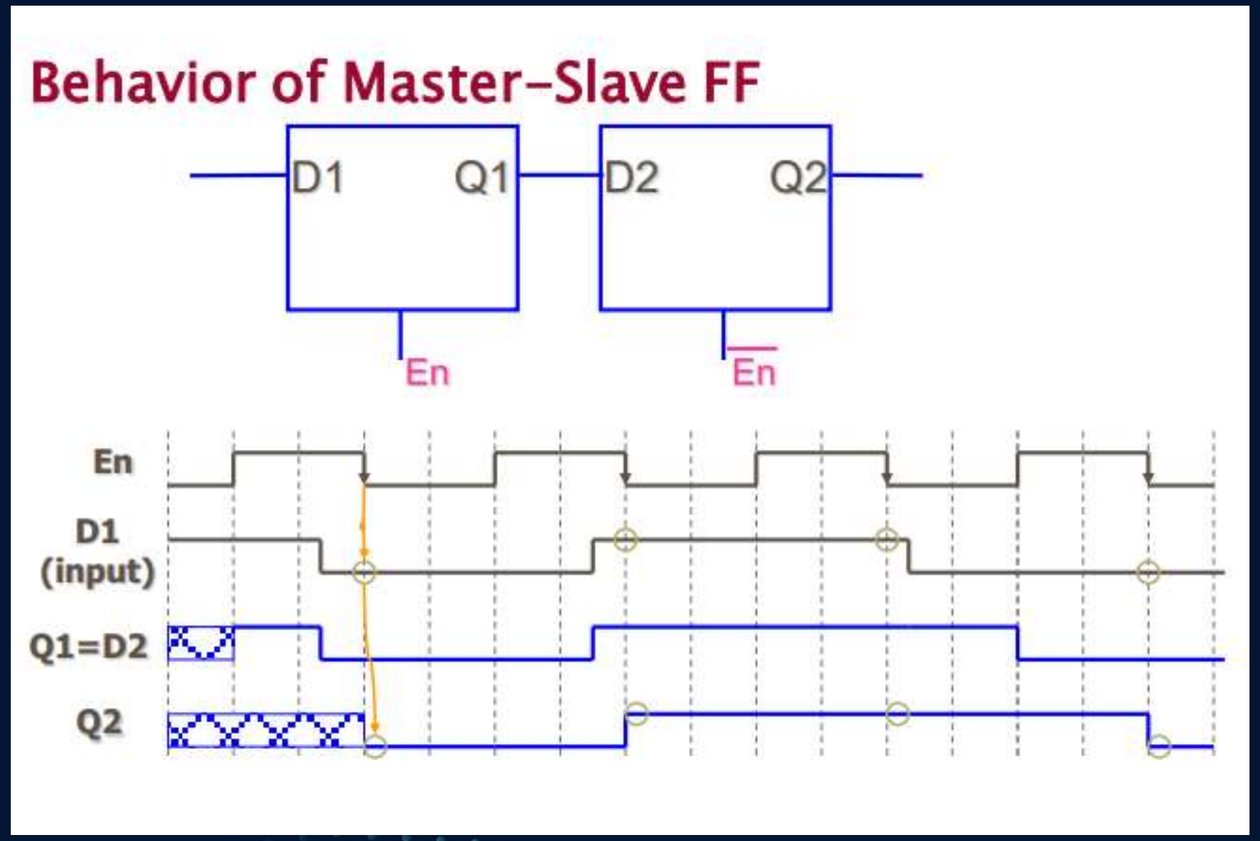


Behavior of Master-Slave FF



Sequential Circuits





Sequential Circuits

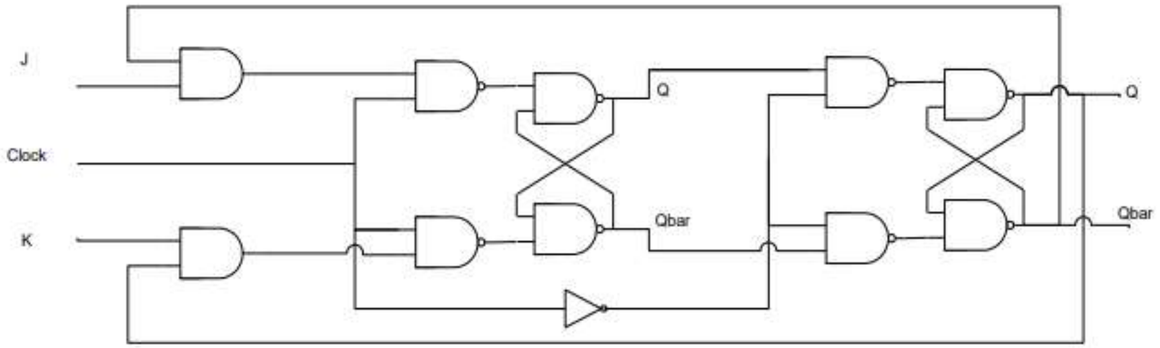
The JK Flip Flop

- SR Flip Flop or Latch has *One Prohibited* input State
- JK overcomes the SR limitation of both inputs being asserted simultaneously
- The Behavior is described by the following Table
  - Both J and K being TRUE cause a TOGGLE of the State

J	K	Next State
0	0	Q
0	1	0
1	0	1
1	1	Q′

The JK Flip Flop

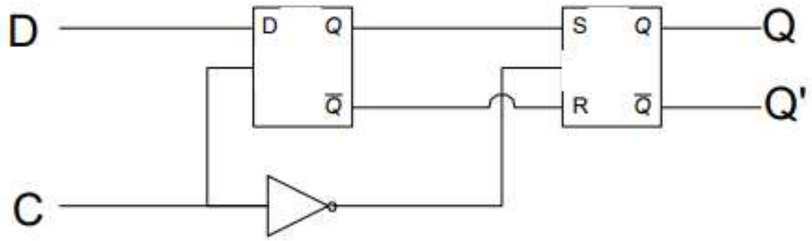
- Two AND gates added to the SR FF
  - The current state of the FF allows the RIGHT input to pass through



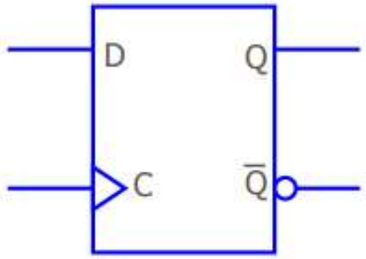
Sequential Circuits

D–type ↓ Edge Triggered Flip Flop

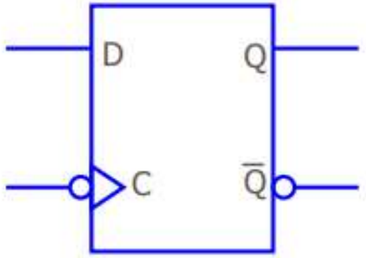
- Cascade of a D and an SR Latch



Flip Flops Symbols



Positive Edge Triggered D Flip Flop



Negative Edge Triggered D Flip Flop

Sequential Circuits

Important FF Parameters

- Setup Time: Minimum time that the input e.g. *D* must be present and stable before the clock edge
- Hold Time: Minimum time that the input e.g. *D* must be held stable after the clock edge
  - A violation of Setup or Hold Time will result in unpredictable outputs
- Propagation Delay or Clock to Q delay:
  - The time after the clock edge to the output being stable in the new state

The Characteristic Table

- Defines the Logical Properties of the Flip Flop
- Similar to the Truth Table for Combinational Circuits

S	R	Next State Q(t+1)	
0	0	Q(t)	Hold
0	1	0	Reset
1	0	1	Set
1	1	?	Undefined

J	K	Next State Q(t+1)	
0	0	Q(t)	Hold
0	1	0	Reset
1	0	1	Set
1	1	Q'(t)	Toggle

D	Next State Q(t+1)	
0	0	Reset
1	1	Set

T	Next State Q(t+1)	
0	Q(t)	Maintain state
1	Q'(t)	Toggle

Sequential Circuits

Sequential Circuit Analysis

- Label All Flip Flop Outputs (arbitrarily)
- Write Equations describing combinational logic leading to all Flip Flop Inputs
- Determine the FF Outputs from Characteristic Table
- Complete the **State Table**
  - Present State: Inputs: Next State: Outputs
- Circuit with m FF and n inputs need 2^{m+n} entries in the **State Table**

State Table

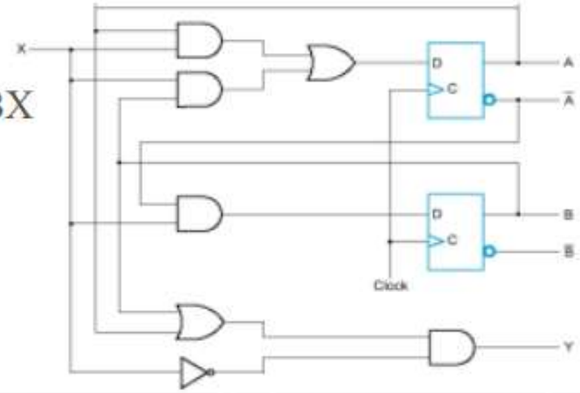
- The table shows the present states, inputs, next states and outputs
- One Dimensional (present state and inputs combined) Moore Model
- Two Dimensional (present state left column and inputs tabulated across the top) Mealy Model

Sequential Circuits

Present State		Input	Next State		Output
A	B	X	A	B	Y
0	0	0	0	0	0
0	0	1	0	1	0
0	1	0	0	0	1
0	1	1	1	1	0
1	0	0	0	0	1
1	0	1	1	0	0
1	1	0	0	0	1
1	1	1	1	0	0

$A(t+1) = D_A = AX + BX$

$B(t+1) = D_B = \bar{A}X$



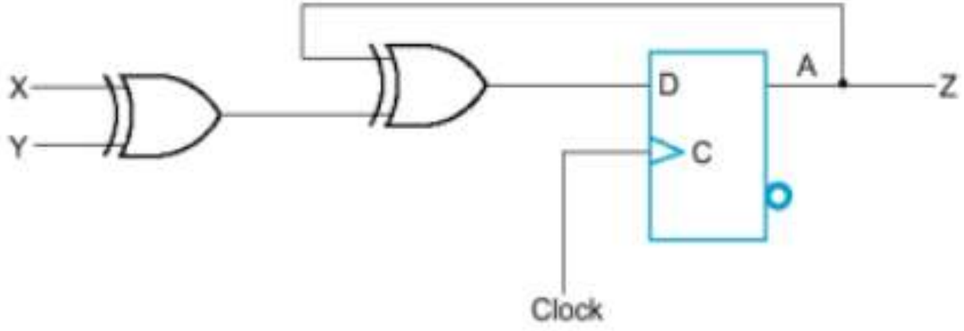
$Y = A\bar{X} + B\bar{X}$

2-D State Table

Present state		Next state				Output	
		X = 0		X = 1		X = 0	X = 1
A	B	A	B	A	B	Y	Y
0	0	0	0	0	1	0	0
0	1	0	0	1	1	1	0
1	0	0	0	1	0	1	0
1	1	0	0	1	0	1	0

Sequential Circuits

Logic Diagram and State Table for $D_A = A \oplus X \oplus Y$



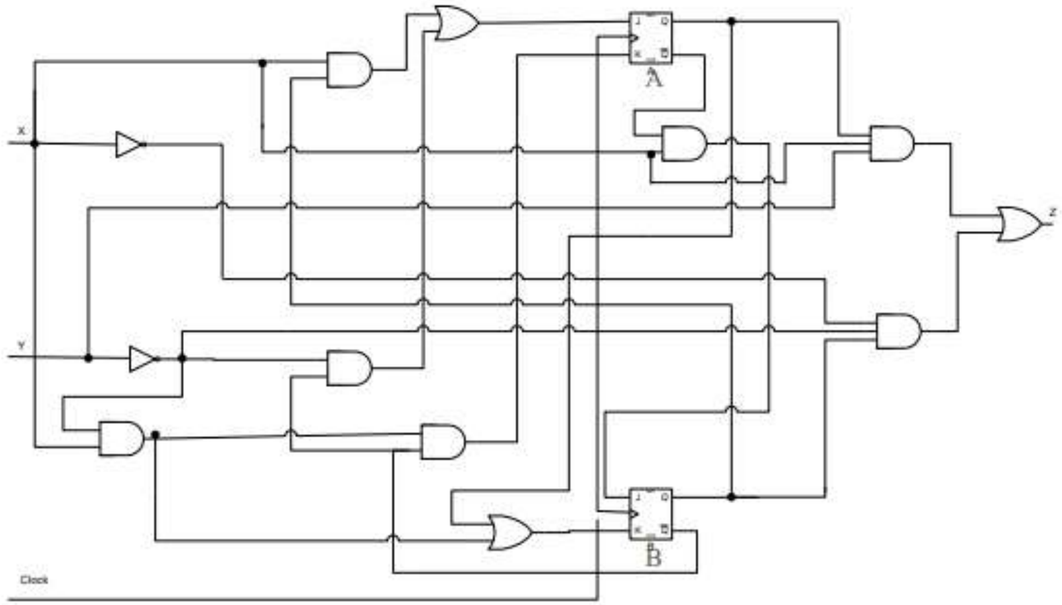
(a)

Present state	Inputs		Next state	Output
A	X	Y	A	Z
0	0	0	0	0
0	0	1	1	0
0	1	0	1	0
0	1	1	0	0
1	0	0	1	1
1	0	1	0	1
1	1	0	0	1
1	1	1	1	1

(b) State table

Sequential Circuits

Sequential Circuit Analysis: Example



Sequential Circuit Analysis: Example

1. Find the Flip Flop Input Equations

$$J_A = BX + \overline{B}\,\overline{Y} \qquad K_A = \overline{B}X\overline{Y}$$

$$J_B = \overline{A}X \qquad K_B = A + X\overline{Y}$$

2. Find the Output Equations

$$Z = AXY + B\overline{X}\,\overline{Y}$$

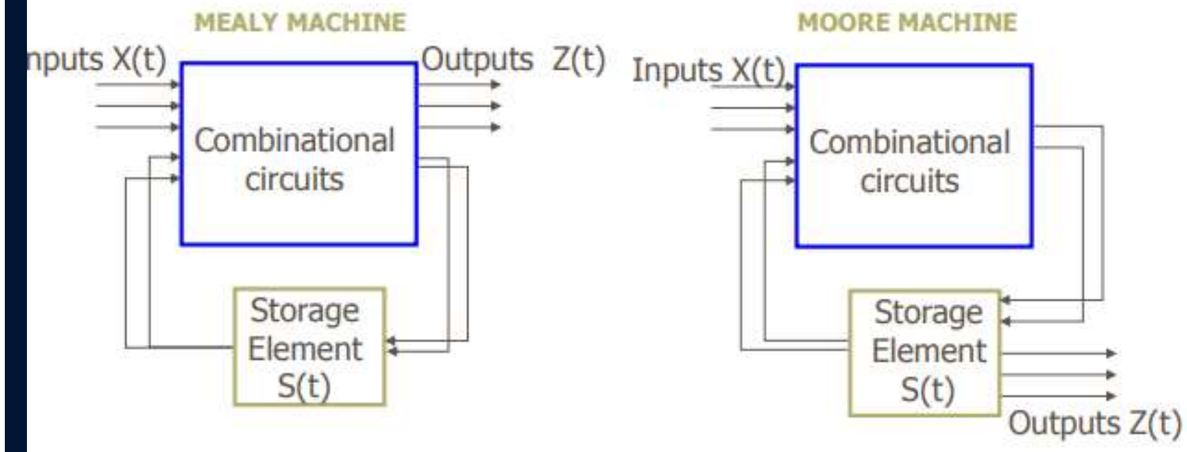
3. Determine the Next State of the Flip Flops from Characteristic Tables
4. Fill the State Table
5. Draw the State Diagram

Sequential Circuits

State and State Diagram

- A **state** represents the machine snapshot at a given clock period
- A clock is typically used to **synchronize** the state transition
- A graph consists of a set of
  - Circles:
    - Each represents a **state**
    - Use double circle to represent the **initial state**
  - Directed arc: each represents a state transition
  - Inputs/outputs
- Mealy machine:
  - Label **input/output** along each arc
- Moore machine:
  - Label **input** along each arc
  - Label **output** inside the circle (i.e. state)

Mealy and Moore Machines



$Z(t) = \lambda\{S(t), X(t)\}$

Output is dependent on state and input

$Z(t) = \lambda\{S(t)\}$

Output is dependent on state only

Sequential Circuits

Definition: A Moore state machine is a finite state machine where the outputs depend **only on the current state** and not on the inputs.

Outputs: The output remains stable and changes only when the state changes.

State Transition: Transitions between states depend on the input conditions.

Applications: Used in synchronous systems where output stability is critical, like counters and traffic light controllers.

Definition: A Mealy state machine is a finite state machine where the outputs depend on **both the current state and the current inputs**.

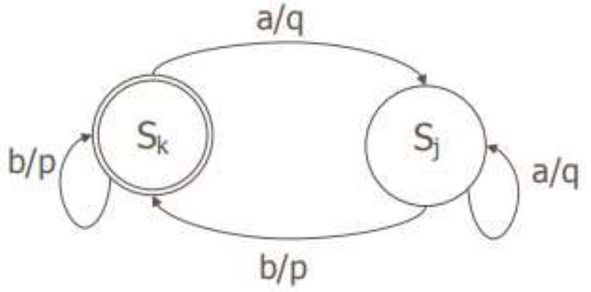
Outputs: The output can change immediately when inputs change, even without a state change.

State Transition: Transitions and outputs are determined simultaneously based on input and state.

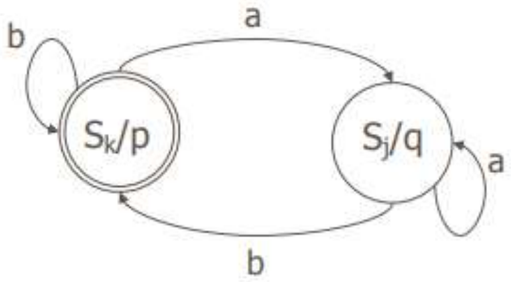
Applications: Used in systems requiring faster response to input changes, such as edge detection and sequence recognition.

Sequential Circuits

State Diagrams



A Mealy machine example



A Moore machine example

Example:

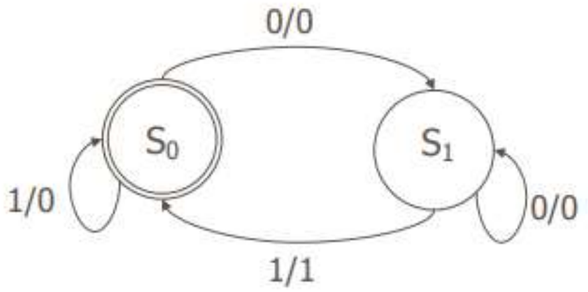
State: $S(t) \in \{S_k, S_j\}$

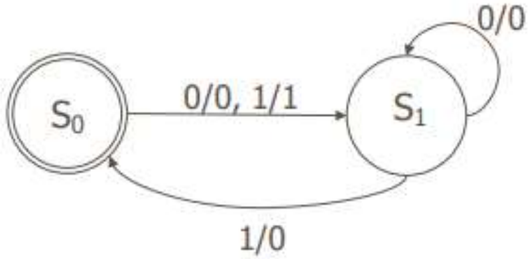
Inputs: $X(t) \in \{a, b\}$

Outputs: $Z(t) \in \{p, q\}$

Initial state: $S(0) = S_k$

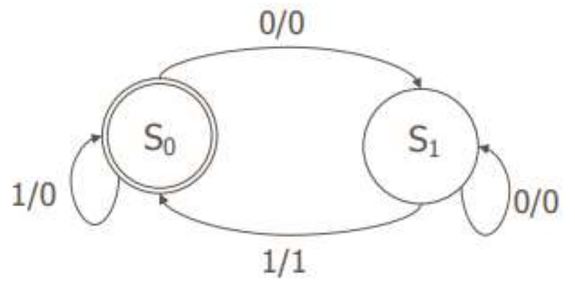
State Diagram Examples (Mealy)

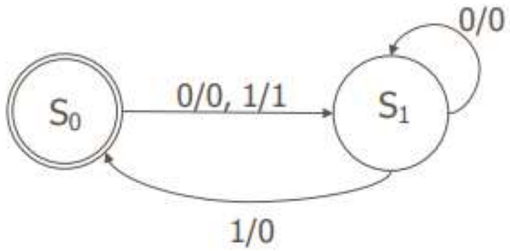




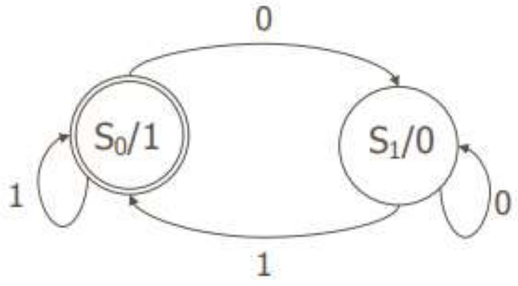
Sequential Circuits

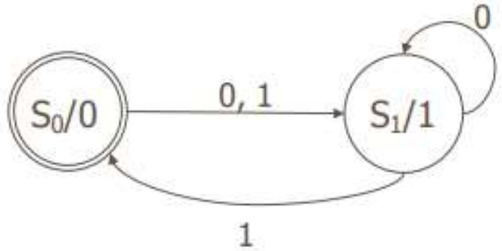
State Diagram Examples (Mealy)



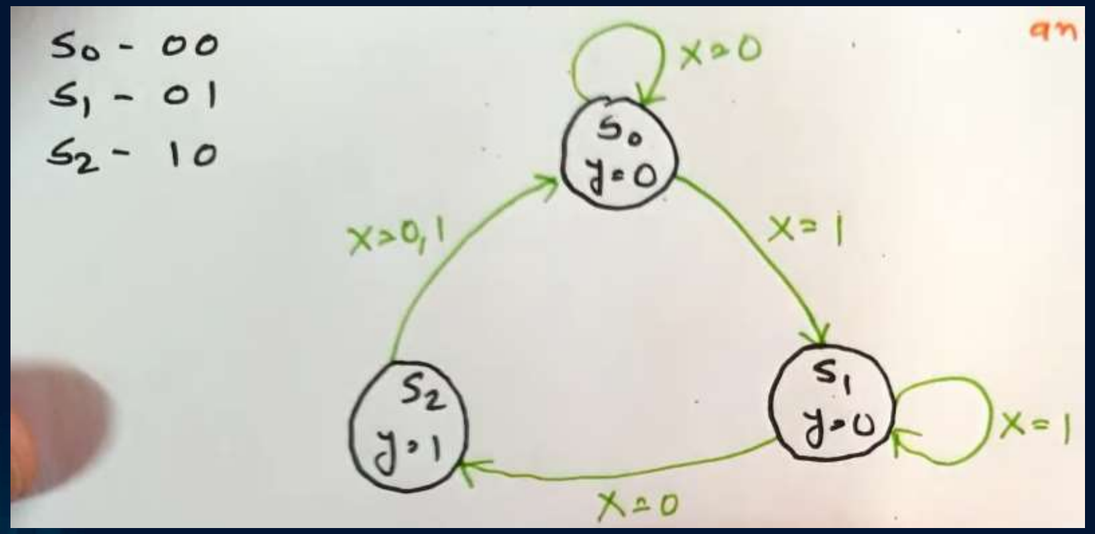


State Diagram Examples (Moore)



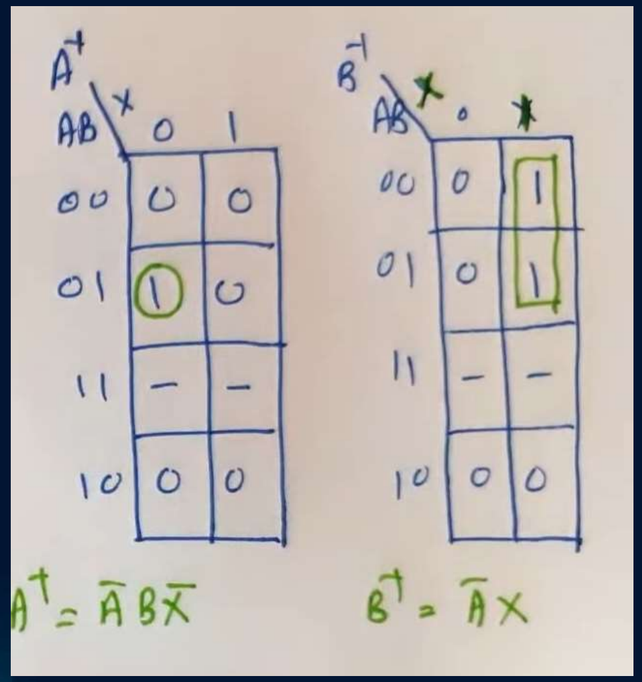


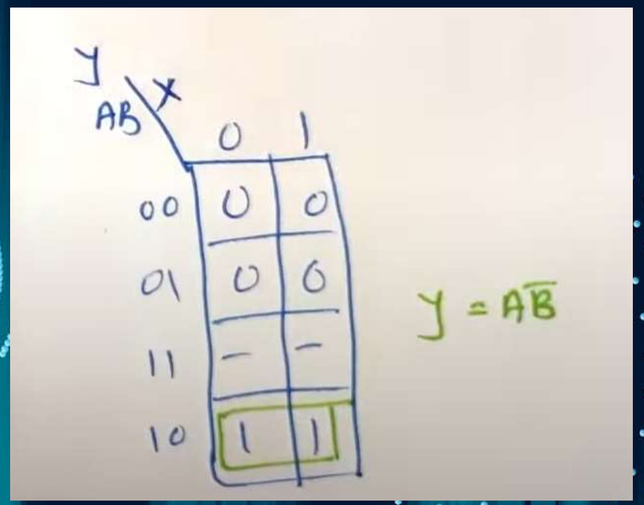
Sequential Circuits

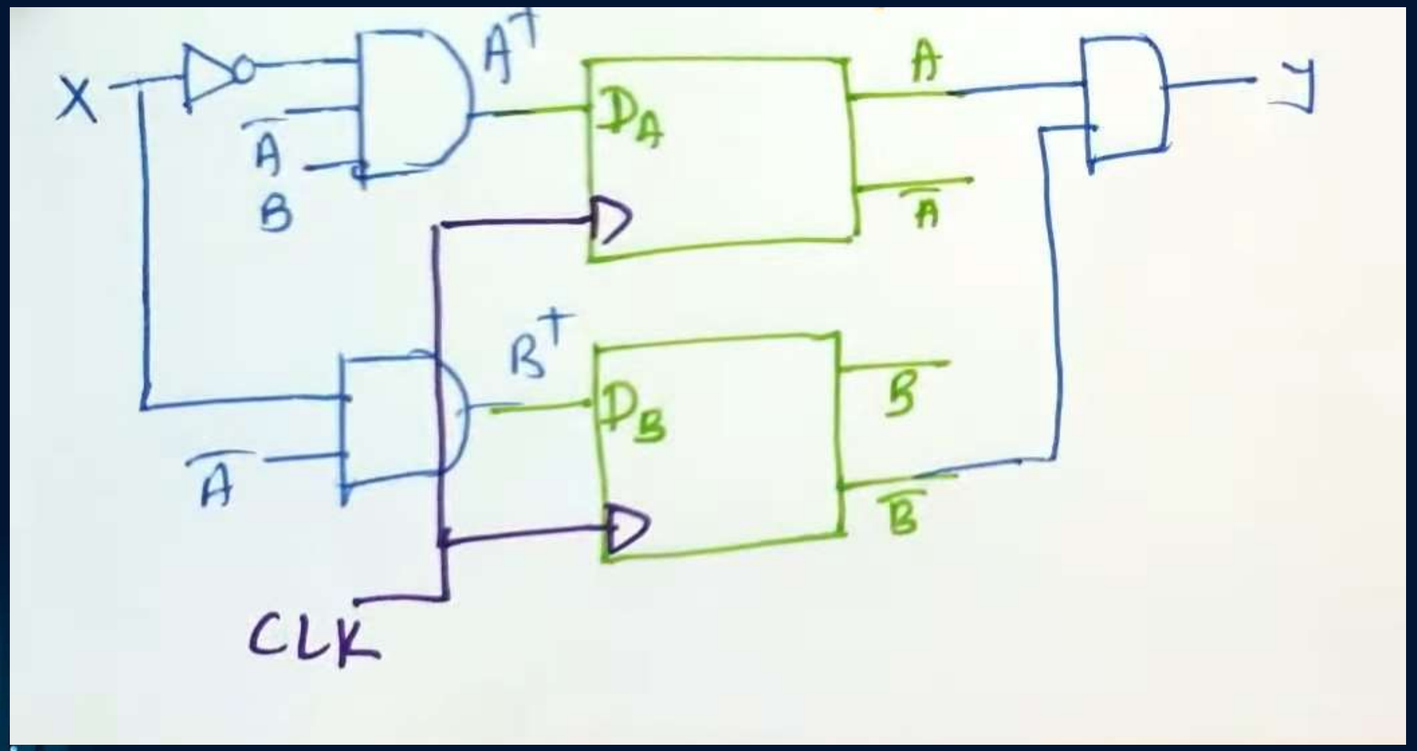


Present state A	Present state B	Input x	Next state $A^+(D_A)$	Next state $B^+(D_B)$	output y
0	0	0	0	0	0
0	0	1	0	1	0
0	1	0	1	0	0
0	1	1	0	1	0
1	0	0	0	0	1
1	0	1	0	0	1
1	1	0	—	—	—
1	1	1	—	—	—

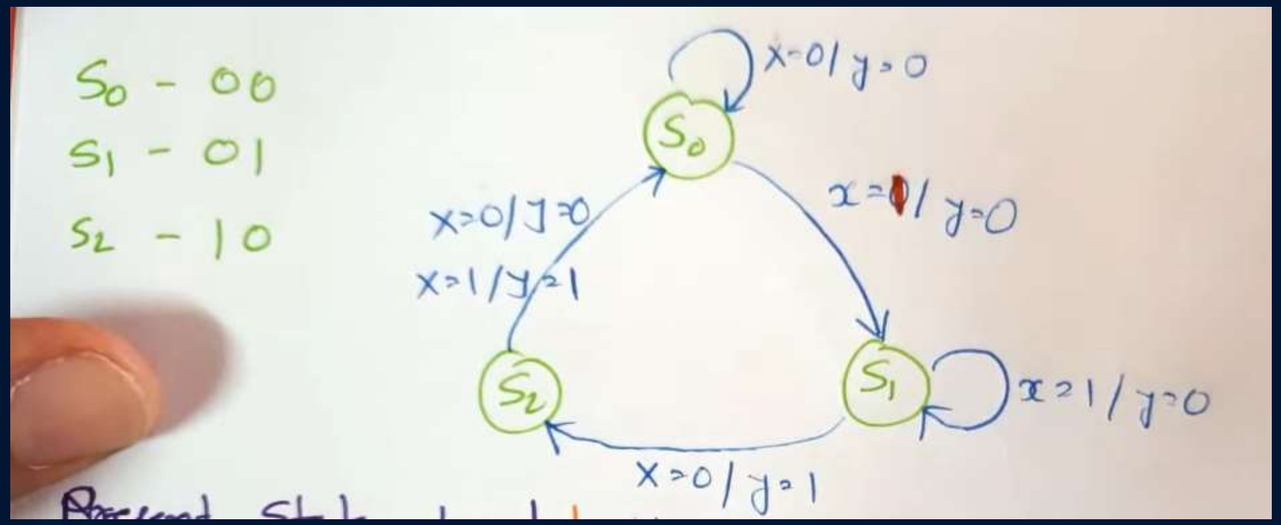
Sequential Circuits

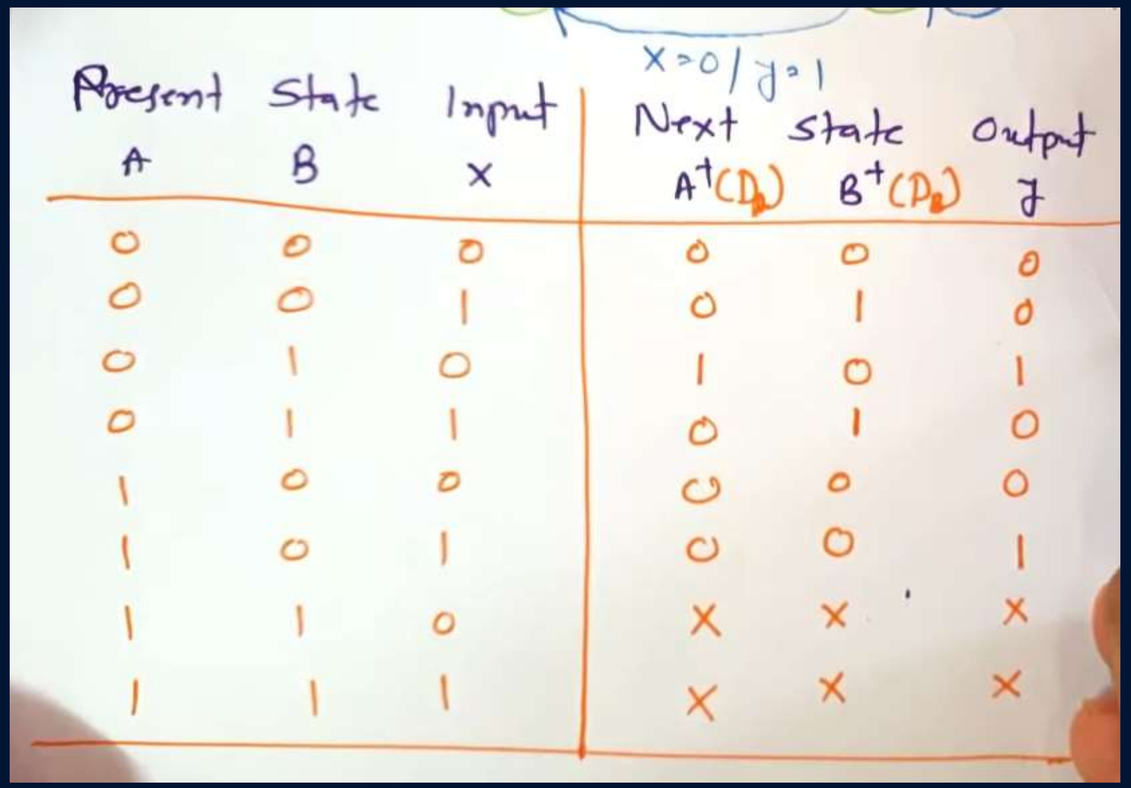






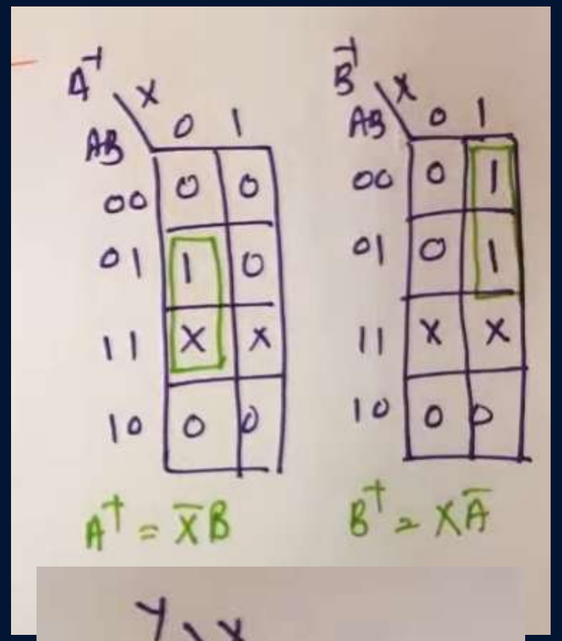
Sequential Circuits

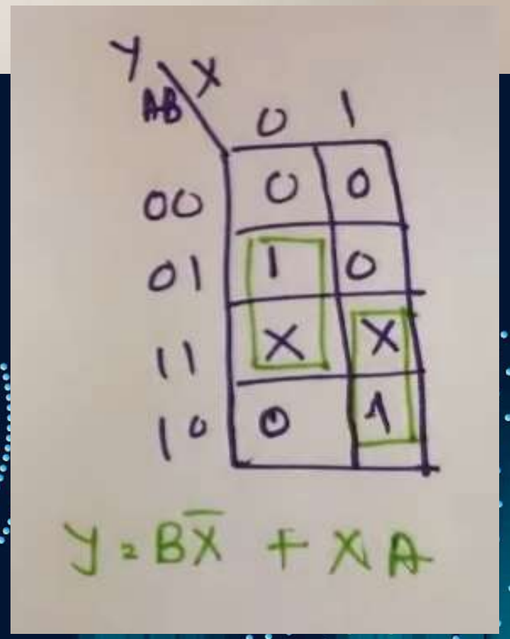


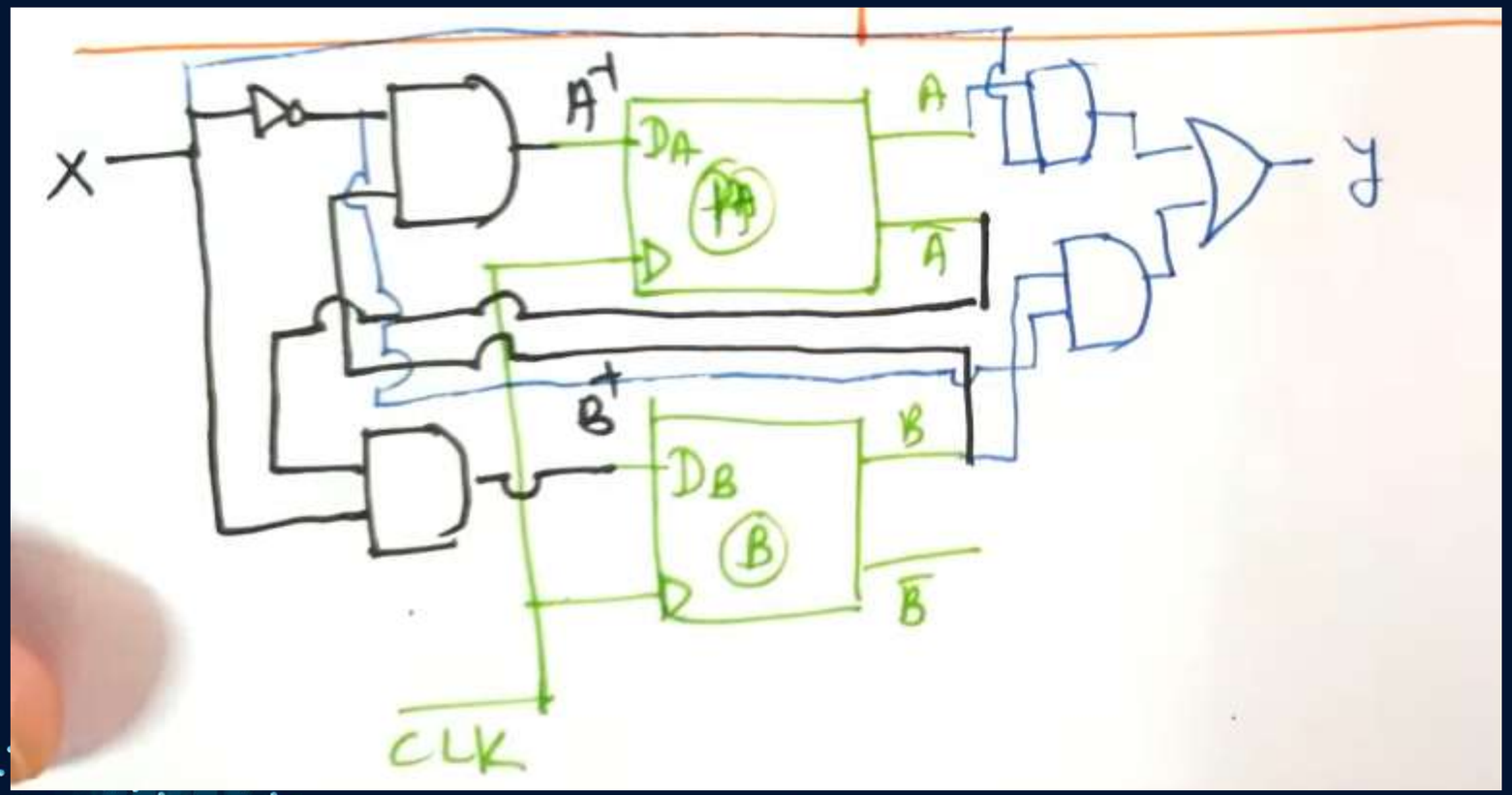


Present State A	Present State B	Input x	Next state $A^+(D_A)$	Next state $B^+(D_B)$	Output z
0	0	0	0	0	0
0	0	1	0	1	0
0	1	0	1	0	1
0	1	1	0	1	0
1	0	0	0	0	0
1	0	1	0	0	1
1	1	0	x	x	x
1	1	1	x	x	x

Sequential Circuits







Thanks!

Instagram

Facebook

Linkedin