Department of Electrical & Computer Engineering

# COEN 212:
# DIGITAL SYSTEMS DESIGN I
## Lecture 8: Common Combinational Logic Circuits
### Decoders, Encoders, Multiplexers

**Instructor:** Dr. Reza Soleymani, Office: EV-5.125,
Telephone: 848-2424 ext.: 4103.

# Lecture 8:
# Objectives of this lecture

Université Concordia University
*Department of Electrical & Computer Engineering*

- In this lecture, we talk about:
  - Decoders.
  - Encoders.
  - Multiplexers.

# Lecture 8:
# Reading for this lecture

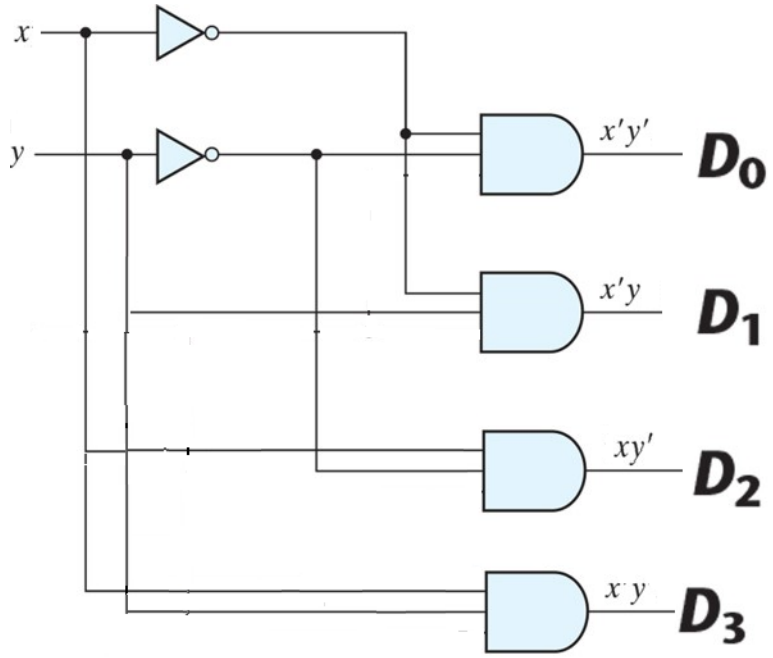


- Digital Design by M. Morris R. Mano and Michael D. Ciletti, 6th Edition, Pearson, 2018:
  - Chapter 4 (4.9 to 4.11)

# Lecture 8: Decoders:

Department of Electrical & Computer Engineering

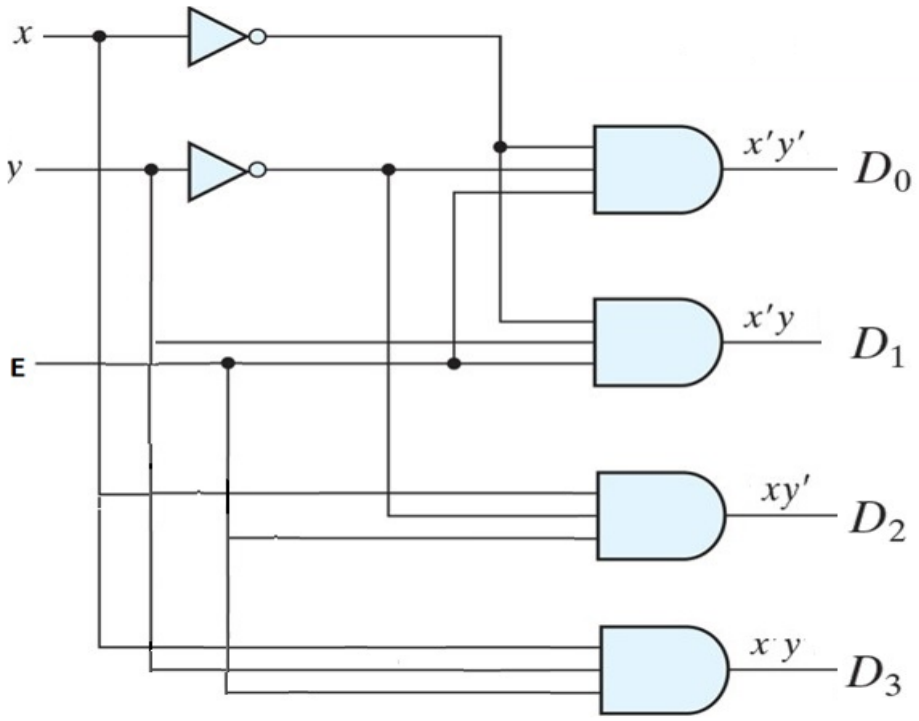
- A 2-bit (2x4) Decoder:



# Lecture 8:
# Decoders, Encoders, Multiplexers:

Department of Electrical & Computer Engineering

- A 2-bit Decoder with ENABLE input:



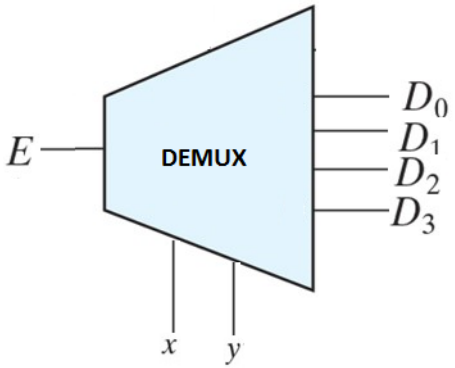
# Lecture 8:
# Decoders, Encoders, Multiplexers:

Concordia UNIVERSITÉ UNIVERSITY
*Department of Electrical & Computer Engineering*

- Truth Table of Decoder:

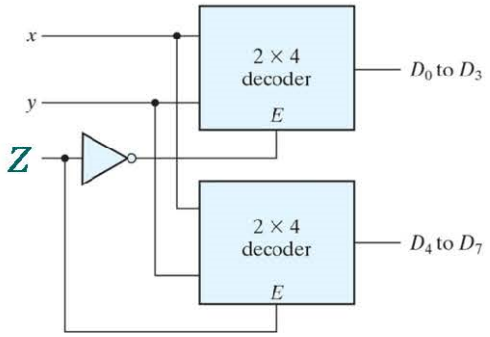
| $E$ | $x$ | $y$ | $D_0$ | $D_1$ | $D_2$ | $D_3$ |
|---|---|---|---|---|---|---|
| **0** | **X** | **X** | 0 | 0 | 0 | 0 |
| **1** | **0** | **0** | 1 | 0 | 0 | 0 |
| **1** | **0** | **1** | 0 | 1 | 0 | 0 |
| **1** | **1** | **0** | 0 | 0 | 1 | 0 |
| **1** | **1** | **1** | 0 | 0 | 0 | 1 |

- $x$ and $y$ cause the enable to appear on pins $D_0$, $D_1$, $D_2$, $D_3$ This is what is termed a **demultiplexer**.



# Lecture 8:
# Decoders, Encoders, Multiplexers:

- **Expanding the decoders:** An $n$-bit decoder with can be implemented using two $n$ – 1-bit decoders.
- **Example:** Use two 2-input decoders to implement a 3-input decoder.



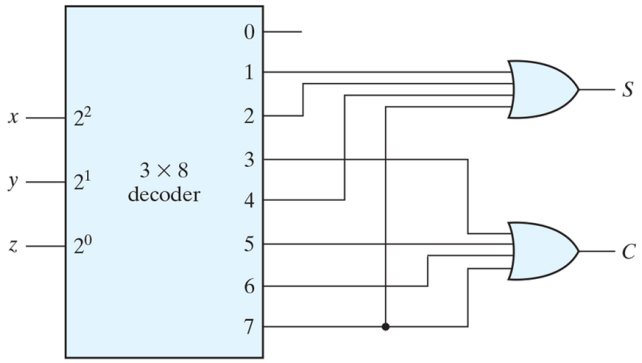
# Lecture 8:
# Combinational logic implementation using decoders:

- **Combination logic implementation using decoders:**
  - Outputs of a decoder represent the minterms of the input.
  - ORing these outputs we can implement any function.
- **Example:** Design a Full Adder.

| $x$ | $y$ | $z$ | $S$ | $C$ |
|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 1 | 1 | 0 |
| 0 | 1 | 0 | 1 | 0 |
| 0 | 1 | 1 | 0 | 1 |
| 1 | 0 | 0 | 1 | 0 |
| 1 | 0 | 1 | 0 | 1 |
| 1 | 1 | 0 | 0 | 1 |
| 1 | 1 | 1 | 1 | 1 |

$S=\sum(1,2,4,7)$

$C=\sum(3,5,6,7)$



Concordia UNIVERSITÉ UNIVERSITY
Department of Electrical & Computer Engineering

- **Truth Table for an Octal Encoder:**

| $D_0$ | $D_1$ | $D_2$ | $D_3$ | $D_4$ | $D_5$ | $D_6$ | $D_7$ | $x$ | $y$ | $z$ |
|---|---|---|---|---|---|---|---|---|---|---|
| 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 |
| 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 |
| 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 1 | 1 |
| 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 1 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 1 | 0 | 1 |
| 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 1 | 1 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 1 |

- **Implementation:**

$$z = D_1 + D_3 + D_5 + D_7$$
$$y = D_2 + D_3 + D_6 + D_7$$
$$x = D_4 + D_5 + D_6 + D_7$$

# Lecture 8: Encoders:



- Everything is fine as long as one and only one input is high.
  - When the output is 000: is $D_1$ zero or one?
  - When the output is 011: is $D_3$ high or both $D_3$ and $D_1$?
  - Press 3 and 6 Simultaneously. Then $D_3 = D_6 = 1$ and $z = y = x = 1$. But, 111 represents $D_7$ !

- Solution: Including extra logic
  - indicating whether any of the inputs is on or not.
  - A priority logic selecting one of the inputs when more than one input is high.

# Lecture 8:
# Encoder with priority Logic

Université Concordia University
Department of Electrical & Computer Engineering

- **Example: a 4-input encoder:**

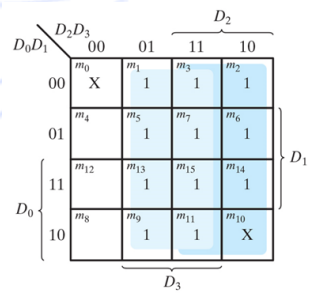
| $D_0$ | $D_1$ | $D_2$ | $D_3$ | $x$ | $y$ | $V$ |
|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | X | X | 0 |
| 1 | 0 | 0 | 0 | 0 | 0 | 1 |
| X | 1 | 0 | 0 | 0 | 1 | 1 |
| X | X | 1 | 0 | 1 | 0 | 1 |
| X | X | X | 1 | 1 | 1 | 1 |

- **Extra output** $V = D_0 + D_1 + D_2 + D_3$.
- $V = 0$ : **no input.** $V = 1$: **input shown by** $(x, y)$.
- When two inputs are high, the one with larger index is selected.

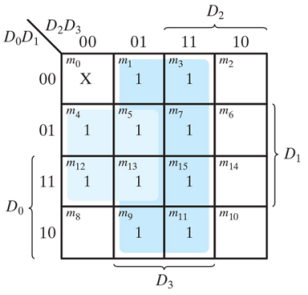
# Lecture 8:
# Encoder with priority Logic

Department of Electrical & Computer Engineering

- **K-map for $x$ :**

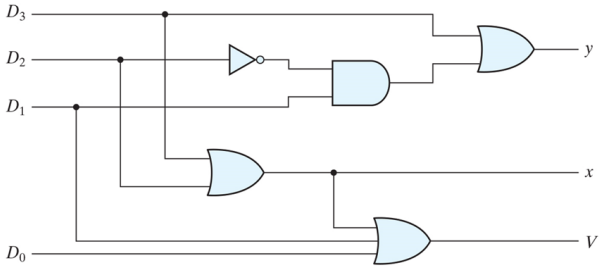



**K-map for $y$ :**




- **So:** $x = D_2 + D_3$ and $y = D_3 + D_1 D_2'$.

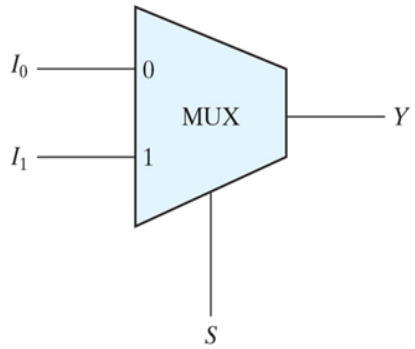
- **Implementation:**



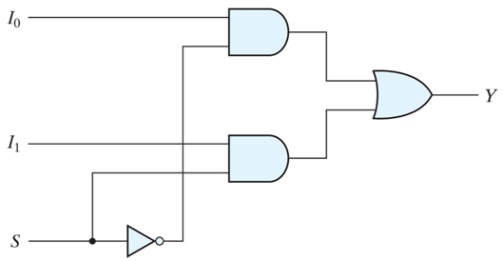
# Lecture 8: Multiplexers



- Example: 2-bit MUX



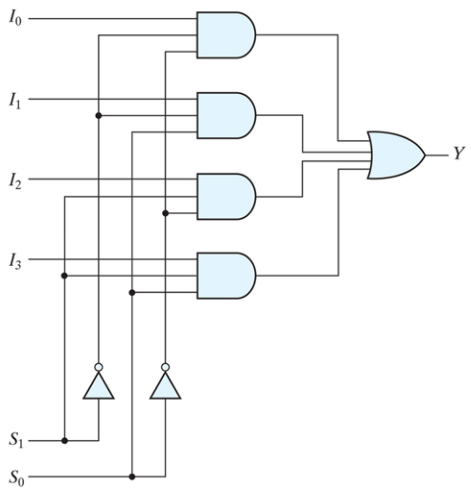

- Implementation:



# Lecture 8: Multiplexers



- **Example: 4-bit MUX**
- **Need $\log_2^4 = 2$ select inputs:**
- **Implementation:**



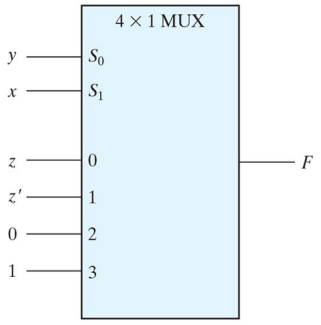
# Lecture 8:
# Boolean Function Design using MUX

Concordia University
Department of Electrical & Computer Engineering

- A Boolean function with n variables can be implemented using a multiplexer with $2^{n-1}$ inputs.
- It has $n-1$ select lines.
- Connect the first $n-1$ variables to the $n-1$ select inputs.
- The remaining variables, say $z$, will be used for data inputs. Depending on the function, the inputs will receive $z, z'$, 1, or 0.
- Example: $F(x, y, z) = \sum(1,2,6,7)$

| $x$ | $y$ | $z$ | $F$ | |
|---|---|---|---|---|
| 0 | 0 | 0 | 0 | $F = z$ |
| 0 | 0 | 1 | 1 | |
| 0 | 1 | 0 | 1 | $F = z'$ |
| 0 | 1 | 1 | 0 | |
| 1 | 0 | 0 | 0 | $F = 0$ |
| 1 | 0 | 1 | 0 | |
| 1 | 1 | 0 | 1 | $F = 1$ |
| 1 | 1 | 1 | 1 | |

Truth Table




Implementation

# Lecture 8:
# Boolean Function Design using MUX

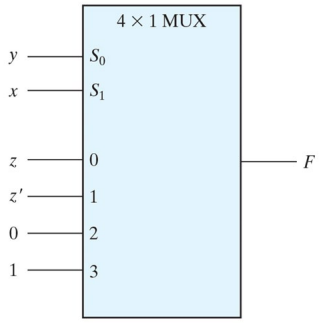
Department of Electrical & Computer Engineering

- Example: Design $(F(A,B,C,D) = \sum(1,3,4,11,12,13,14,15)$
- $n = 4$ so $2^{n-1}$=$2^{4-1} = 8$. We need an 8-to-1 MUX.

- Truth Table:

| $x$ | $y$ | $z$ | $F$ | |
|---|---|---|---|---|
| 0 | 0 | 0 | 0 | $F = z$ |
| 0 | 0 | 1 | 1 | |
| 0 | 1 | 0 | 1 | $F = z'$ |
| 0 | 1 | 1 | 0 | |
| 1 | 0 | 0 | 0 | $F = 0$ |
| 1 | 0 | 1 | 0 | |
| 1 | 1 | 0 | 1 | $F = 1$ |
| 1 | 1 | 1 | 1 | |

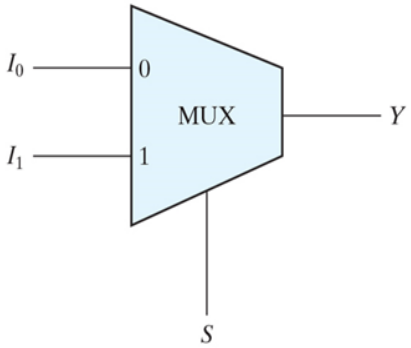
- Implementation:



# Lecture 8: Knowledge Check

Concordia University
Department of Electrical & Computer Engineering

- **Question 1:** In the circuit shown let $I_0 = 1$, $I_0 = 0$, and $S = 1$:




- The output will be?
- a) 0, b) 1,

- **Question 2:** To design a function with inputs A, B, C, D, E, F, we need a multiplexer with:
- a) 5 inputs, b) 8 inputs, c)16 inputs, d) 4 inputs