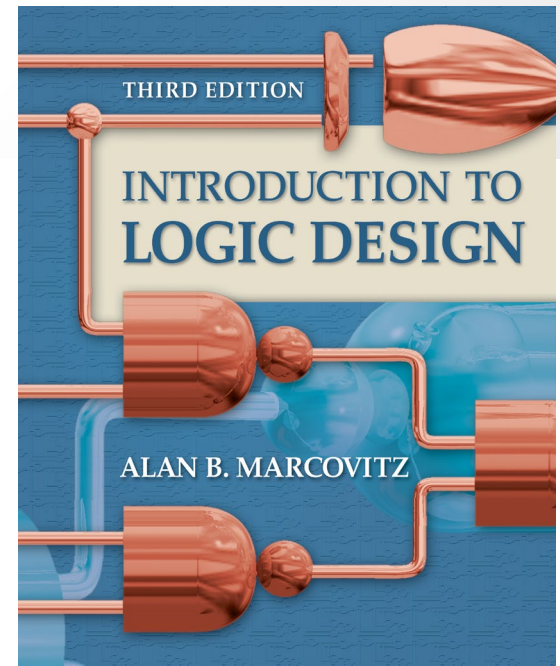


EE2000: Logic Circuit Design

Chapter 5

Acknowledgements: Dr. Wang Hong, Dr. Ray Cheung, Dr. Ricky Lau
Dr. Chang Chip Hong, Dr. Goh Wang Ling



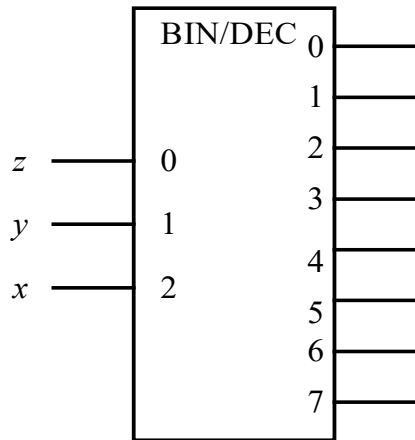
Outline

- 4.1 Logical functional blocks
 - Decoder
 - Encoder
 - Multiplexer
 - Demultiplexer
- 4.2 Programmable logic devices
 - Covered in Topic 6

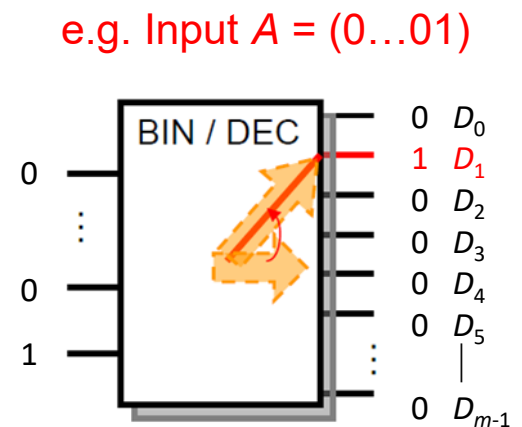
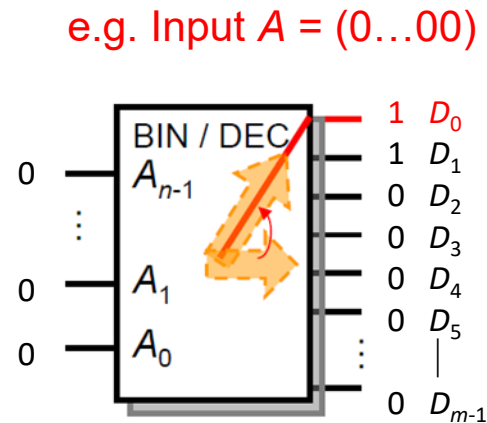
4.1 Logical functional Modules

4.1.1 Decoder

- A decoder is a combinational circuit with n -input and m -output ($0 < n \leq m \leq 2^n$, but usually $m = 2^n$)
- A very important functional blocks as it can be incorporated into many of the other functions



Decoder symbol



1-to-2-Line decoder

Specification:

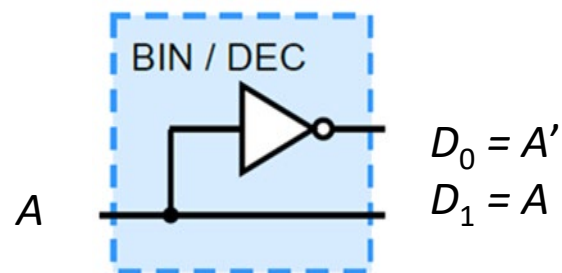
Input: 1-bit (A_0)

Outputs: 2-bit (D_0, D_1)

<i>Input</i>	<i>Outputs</i>	
A	D_0	D_1
0	1	0
1	0	1

$$D_0 = m_0 = A'$$

$$D_1 = m_1 = A$$



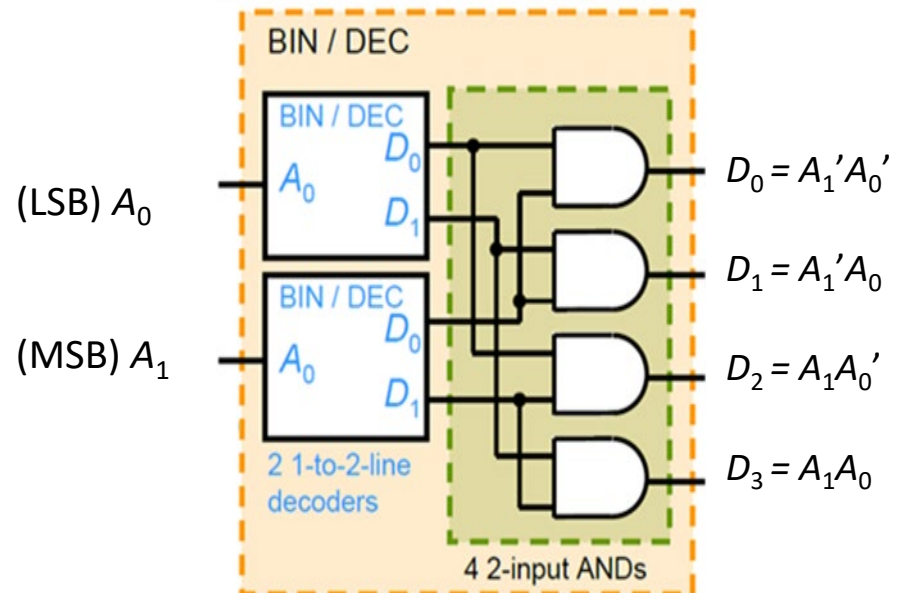
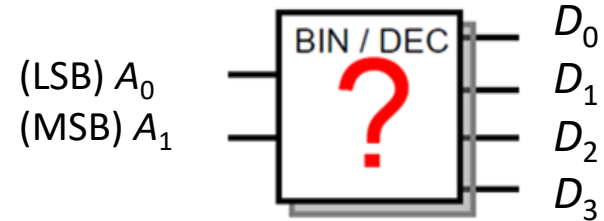
2-to-4-Line Decoder

Specification:

Inputs: 2-bit (A_1, A_0)

Outputs: 4-bit (D_0, D_1, D_2, D_3)

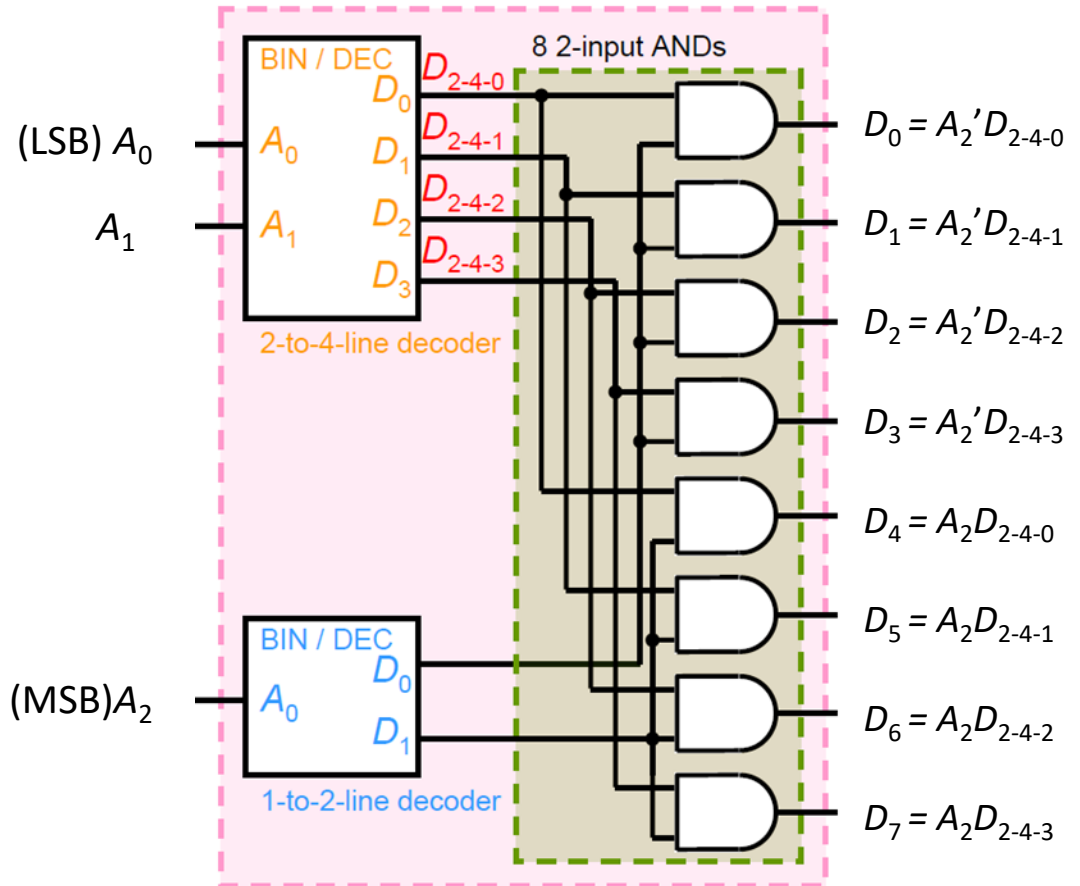
<i>Inputs</i>		<i>Outputs</i>			
A_1	A_0	D_0	D_1	D_2	D_3
0	0	1	0	0	0
0	1	0	1	0	0
1	0	0	0	1	0
1	1	0	0	0	1



3-to-8-Line Decoder

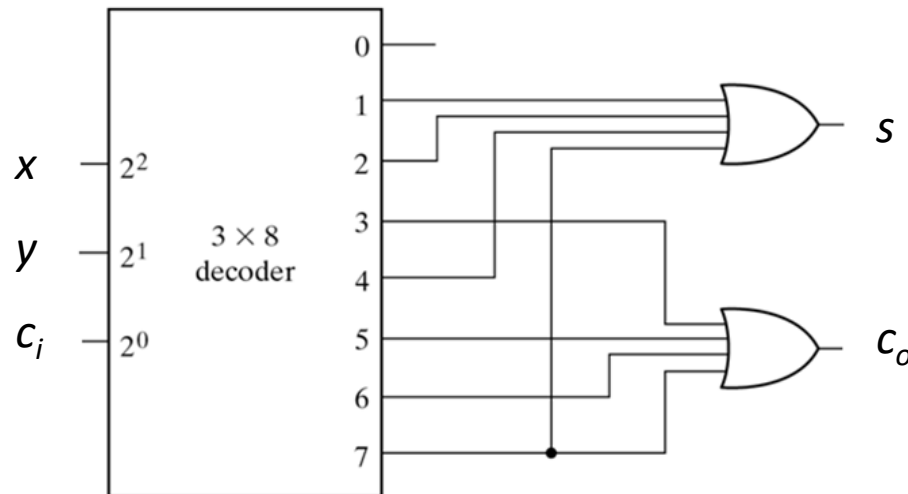
Inputs			Outputs							
A_2	A_1	A_0	D_0	D_1	D_2	D_3	D_4	D_5	D_6	D_7
0	0	0	1	0	0	0	0	0	0	0
0	0	1	0	1	0	0	0	0	0	0
0	1	0	0	0	1	0	0	0	0	0
0	1	1	0	0	0	1	0	0	0	0
1	0	0	0	0	0	0	1	0	0	0
1	0	1	0	0	0	0	0	1	0	0
1	1	0	0	0	0	0	0	0	1	0
1	1	1	0	0	0	0	0	0	0	1

- Build the simplest block:
 - 1-to-2-line decoder
 - 2-to-4-line decoder
 - 3-to-8-line decoder
 - n -to- 2^n -line decoder



Full-adder using decoder

- For a full-adder operation, we have:
- $s(x, y, c_i) = \sum m(1, 2, 4, 7)$ and $c_o = \sum m(3, 5, 6, 7)$
- Since there are 3 inputs, we can use a 3-to-8-line decoder to implement the full-adder circuit.



NOTE: You can only produce more than one output with a single decoder !!

Decoder with Enable input

Specification:

A 2-to-4-line decoder with enable input

<i>Inputs</i>			<i>Outputs</i>			
<i>EN</i>	<i>A₁</i>	<i>A₀</i>	<i>D₀</i>	<i>D₁</i>	<i>D₂</i>	<i>D₃</i>
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

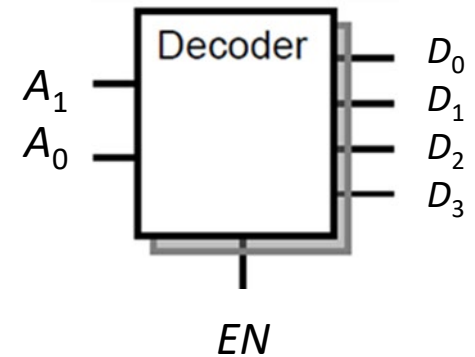
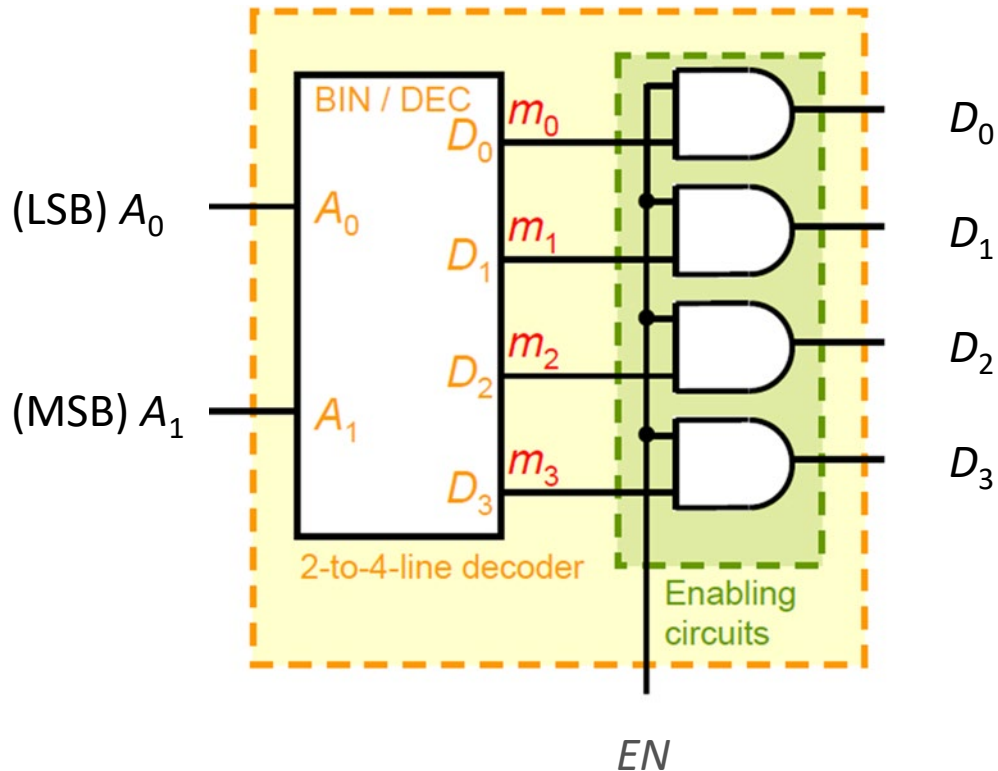
If *EN* to 0 (**disabled**), all outputs are 0

If *EN* to 1 (**enabled**), the output are same as normal decoder (i.e. only the corresponding $D_i = 1$, and all others are 0)

Note: the Xs in the above table are don't care inputs

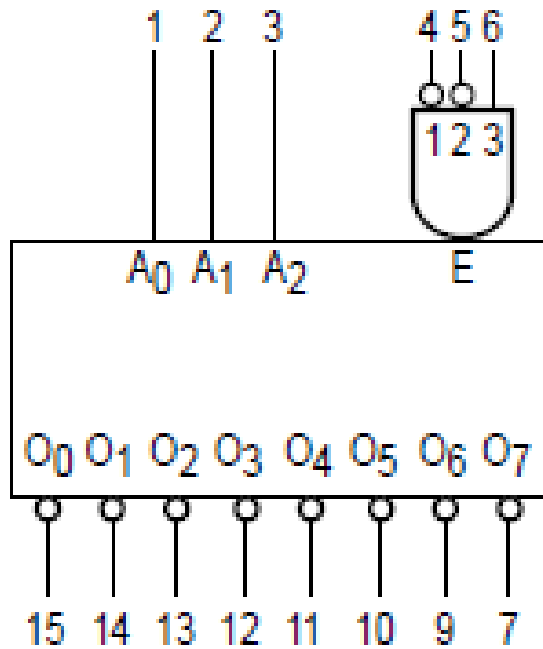
Decoder with Enable input

$$D_i = EN \cdot m_i$$

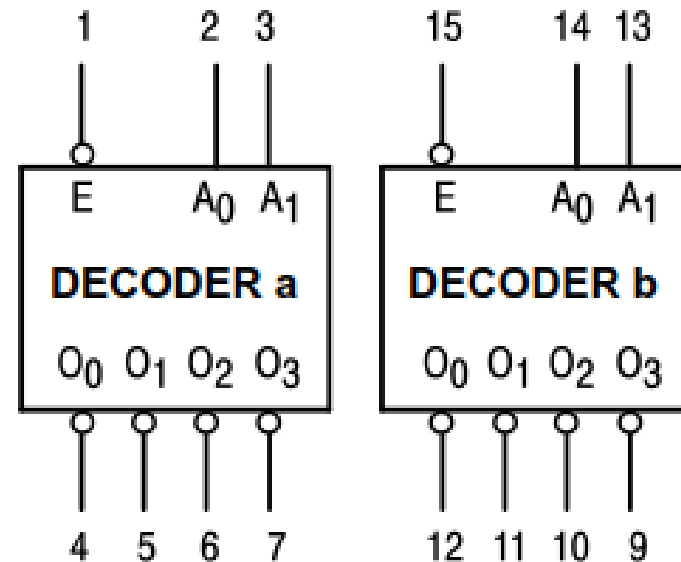


The corresponding symbol of 2-to-4-line decoder with enable input

Decoder IC 74LS138/139

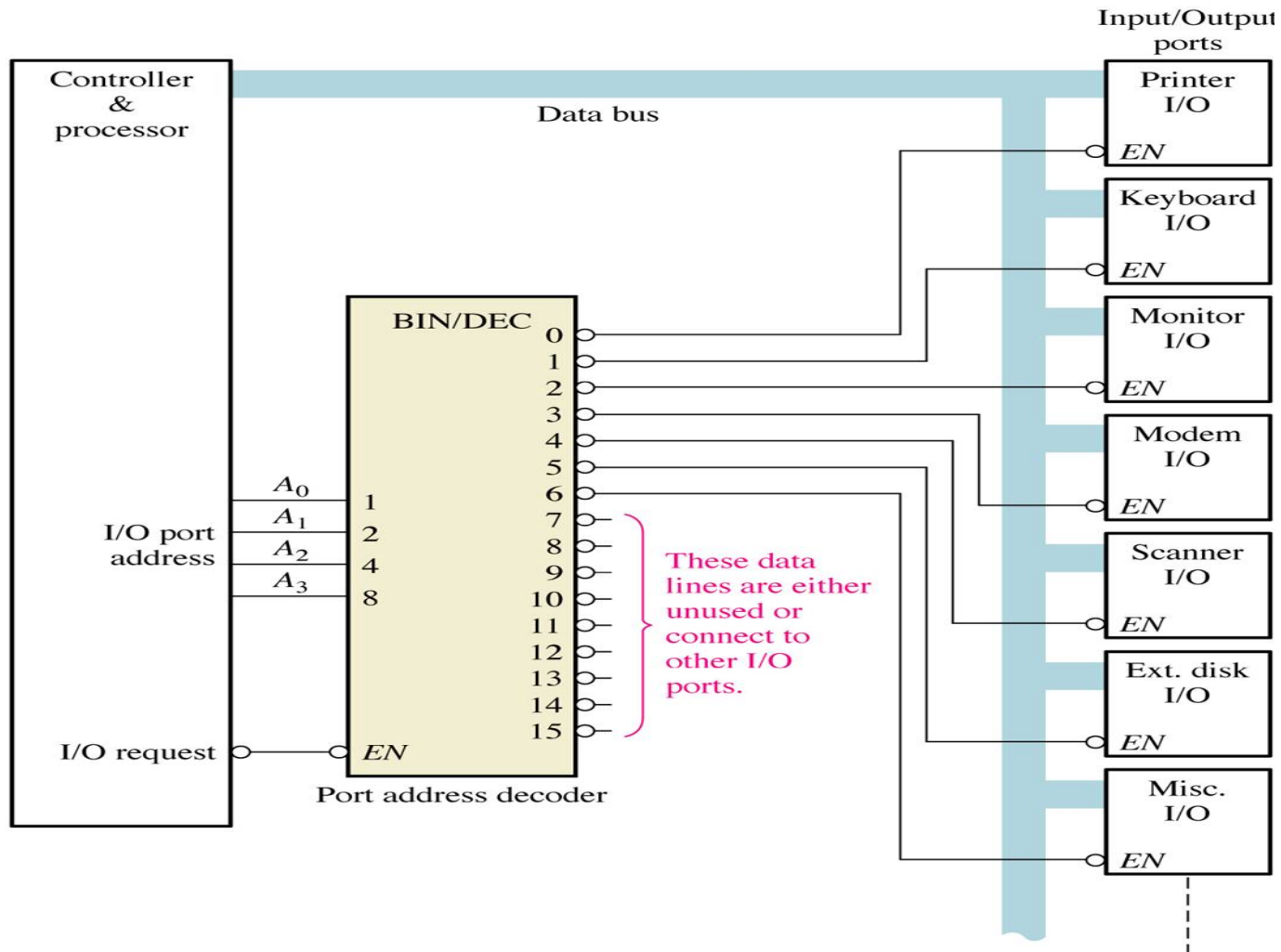


74LS138



74LS139

Application of decoder

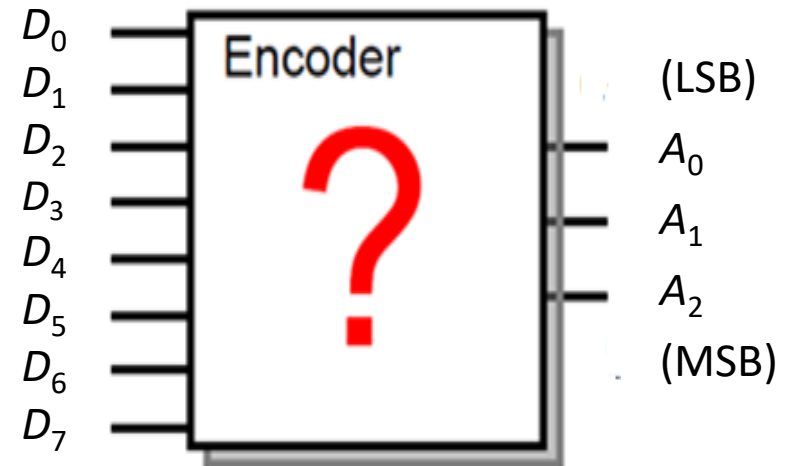


4.1.2 Encoder

- An encoder is a functional block that performs the inverse operation of a decoder
- m inputs and n outputs
- $0 < n \leq m \leq 2^n$, but usually $m = 2^n$

Example of Octal-to-Binary Encoder

<i>Inputs</i>								<i>Outputs</i>		
D_0	D_1	D_2	D_3	D_4	D_5	D_6	D_7	A_2	A_1	A_0
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
All other invalid inputs								X	X	X



Octal-to-binary encoder

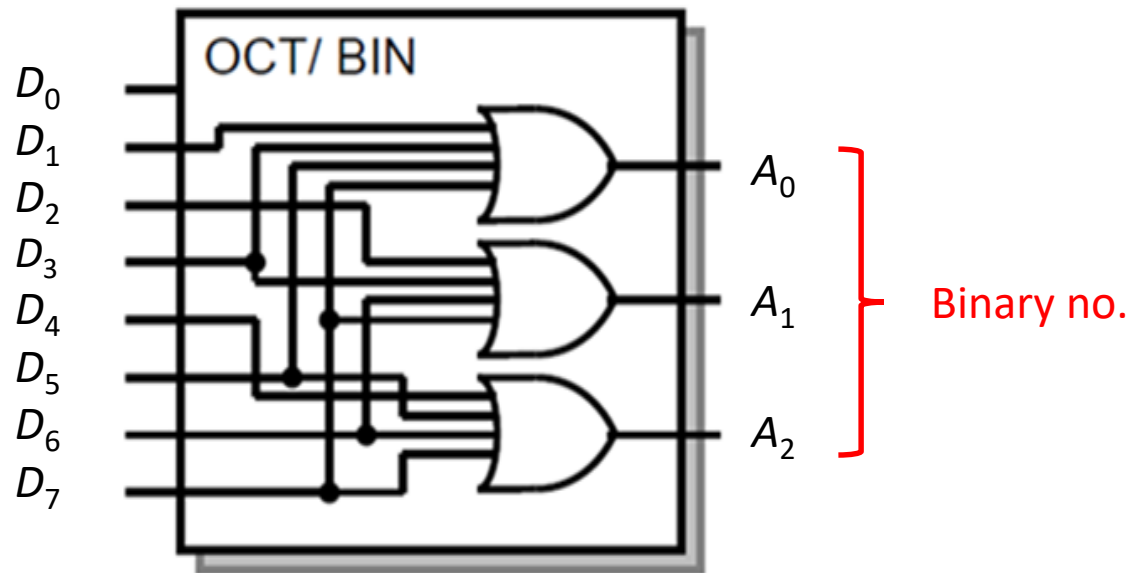
- $m = 8, n = 3$
- i.e. inputs:8, outputs: 3

Octal-to-binary Encoder

$$A_0(D_0, D_1, D_2, D_3, D_4, D_5, D_6, D_7) = D_1 + D_3 + D_5 + D_7$$

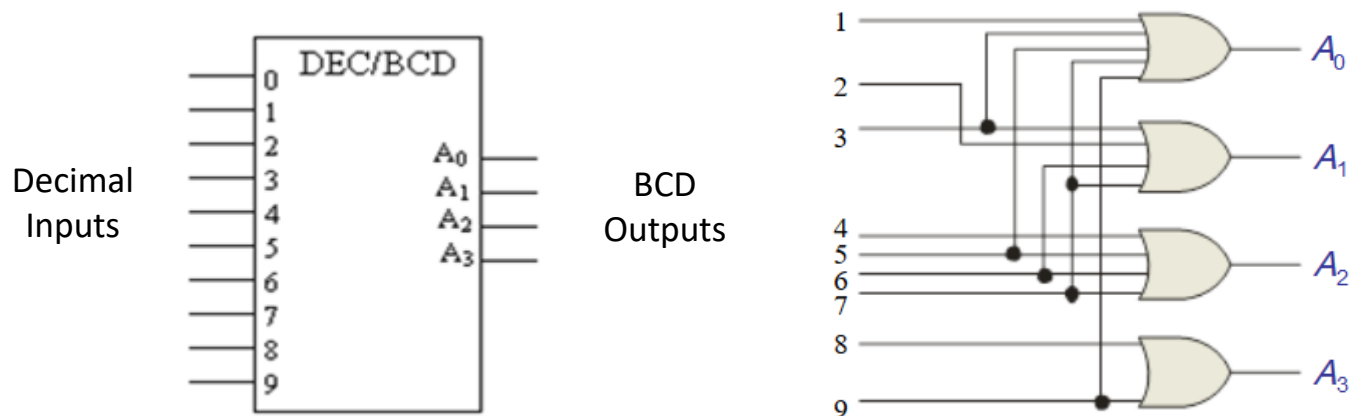
$$A_1(D_0, D_1, D_2, D_3, D_4, D_5, D_6, D_7) = D_2 + D_3 + D_6 + D_7$$

$$A_2(D_0, D_1, D_2, D_3, D_4, D_5, D_6, D_7) = D_4 + D_5 + D_6 + D_7$$



Decimal-to-BCD encoder

Inputs										Outputs			
0	1	2	3	4	5	6	7	8	9	A_3	A_2	A_1	A_0
1	0	0	0	0	0	0	0	0	0	0	0	0	0
0	1	0	0	0	0	0	0	0	0	0	0	0	1
0	0	1	0	0	0	0	0	0	0	0	0	1	0
0	0	0	1	0	0	0	0	0	0	0	0	1	1
0	0	0	0	1	0	0	0	0	0	0	1	0	0
0	0	0	0	0	1	0	0	0	0	0	1	0	1
0	0	0	0	0	0	1	0	0	0	0	1	1	0
0	0	0	0	0	0	0	1	0	0	0	1	1	1
0	0	0	0	0	0	0	0	1	0	1	0	0	0
0	0	0	0	0	0	0	0	0	1	1	0	0	1



Limitation of Encoders

- Only one input can be active (i.e. 1)
- If two or more inputs active simultaneously, the output produces an incorrect combination
- **Solution:**
 - To resolve this ambiguity, introduce an **input priority**
 - Each input pin has different priority
 - If two or more inputs are 1 at the same time, only consider input that has higher priority
 - This kind of encoder is called **priority encoder**

4.1.3 Priority Encoders

- Design a 4-input priority encoder
- Inputs with higher subscript numbers has higher priority

<i>Inputs</i>				<i>Outputs</i>		
D_3	D_2	D_1	D_0	A_1	A_0	V
0	0	0	0	X	X	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

Introduce one more output V (valid output).

If all inputs are 0 (i.e invalid Input), V is 0.

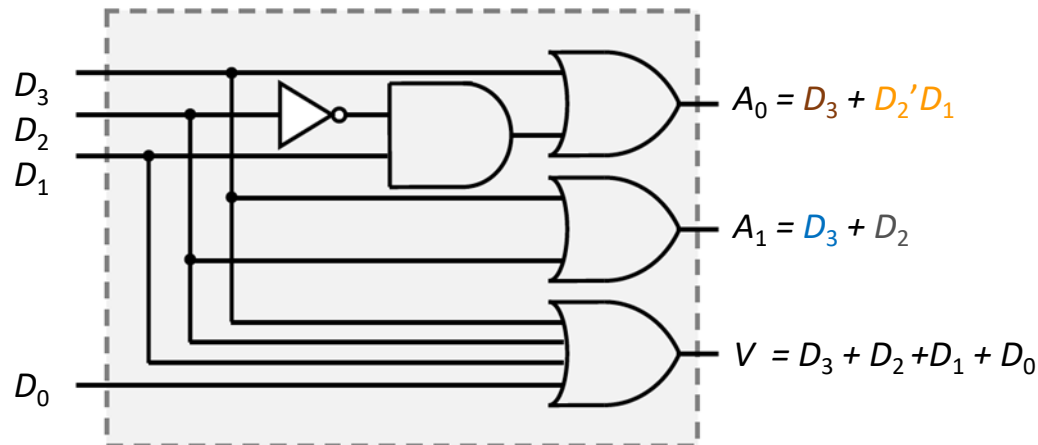
Otherwise V is 1

If D_3 and D_2 are both 1, ignore D_2 (as if D_2 is 0)

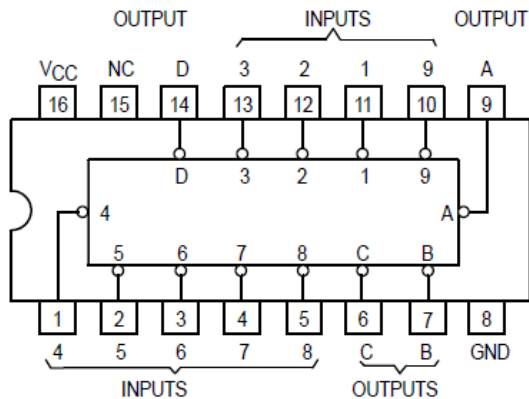
4-Input Priority Encoder

		A_1				A_0				V			
D_1D_0	D_3D_2	00	01	11	10	00	01	11	10	00	01	11	10
	00	x	1	1	1	x		1	1	0	1	1	1
01			1	1	1			1	1	1	1	1	1
11			1	1	1	1		1	1	1	1	1	1
10			1	1	1	1		1	1	1	1	1	1

$A_1(D_3, D_2, D_1, D_0)$		$A_0(D_3, D_2, D_1, D_0)$		$V' = D_3'D_2'D_1'D_0'$	
$= D_3 + D_2$		$= D_3 + D_2'D_1$		$V = D_3 + D_2 + D_1 + D_0$	



Priority Encoder IC 74LS147



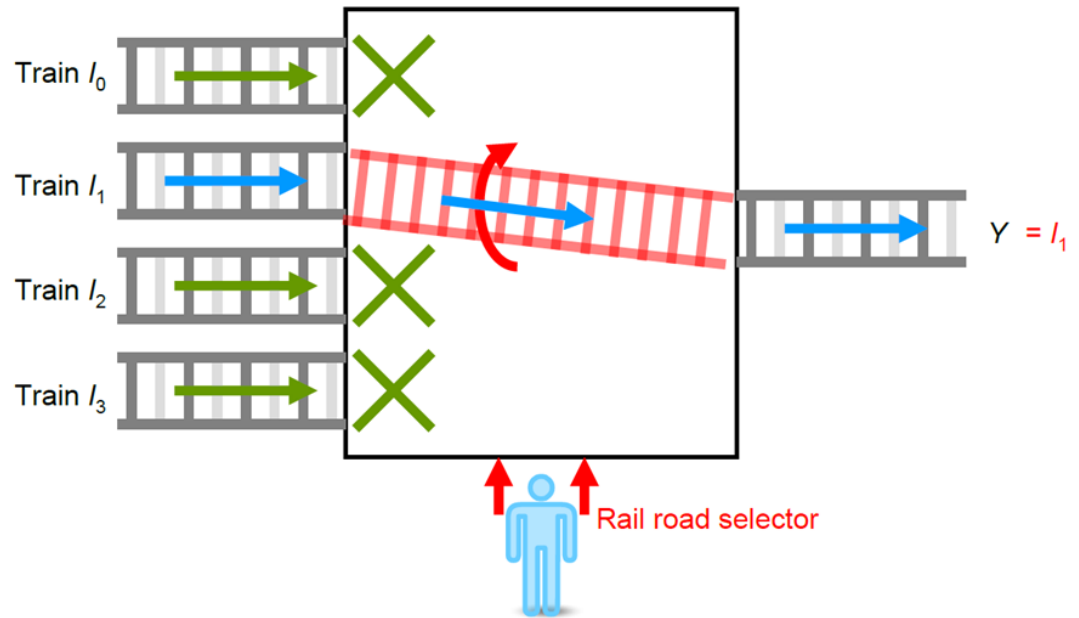
FUNCTION TABLE - '147, 'LS147

INPUTS									OUTPUTS			
1	2	3	4	5	6	7	8	9	D	C	B	A
H	H	H	H	H	H	H	H	H	H	H	H	H
X	X	X	X	X	X	X	X	L	L	H	H	L
X	X	X	X	X	X	X	L	H	L	H	H	H
X	X	X	X	X	X	L	H	H	H	L	L	L
X	X	X	X	X	L	H	H	H	H	L	L	H
X	X	X	X	L	H	H	H	H	H	L	H	L
X	X	X	L	H	H	H	H	H	H	L	H	H
X	X	L	H	H	H	H	H	H	H	H	L	L
X	L	H	H	H	H	H	H	H	H	H	L	H
L	H	H	H	H	H	H	H	H	H	H	H	L

H = high logic level, L = low logic level, X = irrelevant

4.1.4 Logical functional blocks (Multiplexer -- MUX)

- Combinational circuit performs selection
- To appear I_s on a **single output** (Y) from **many inputs** (I_i)



- A set of **selection input variables** $(S)_{10} = (S_{n-1} \dots S_1 S_0)_2$

2-to-1-line MUX

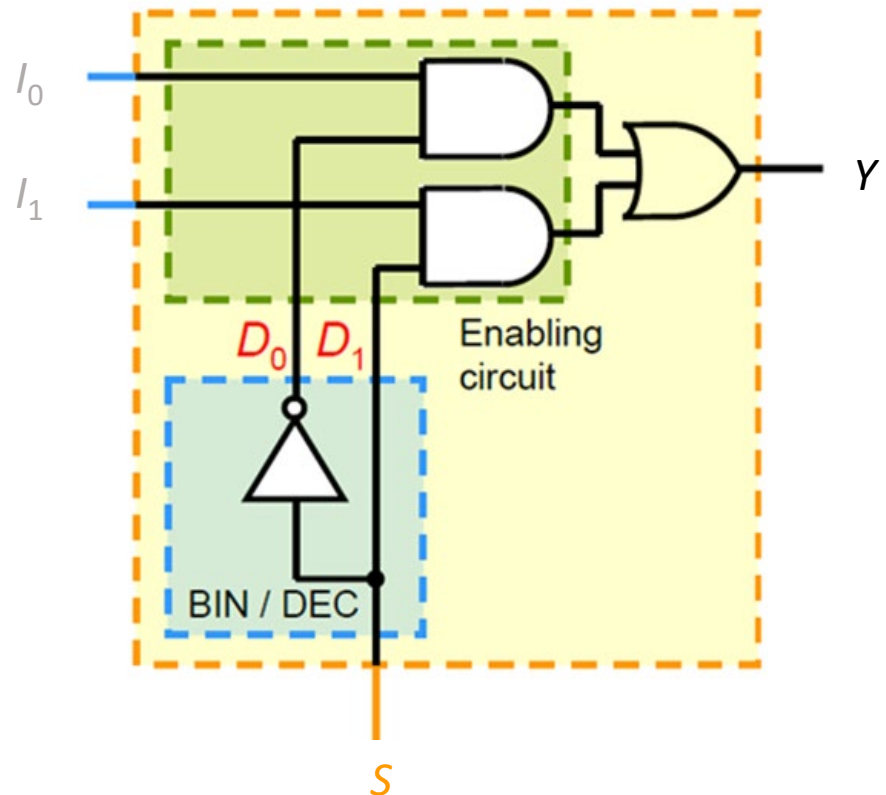
Specification:

$$m = 2$$

$$n = \log_2 m = 1$$

Inputs			Output
I_1	I_0	S_0	Y
X	X	0	I_0
X	X	1	I_1

$$Y(I_0, I_1, S_0) = S_0' I_0 + S_0 I_1$$



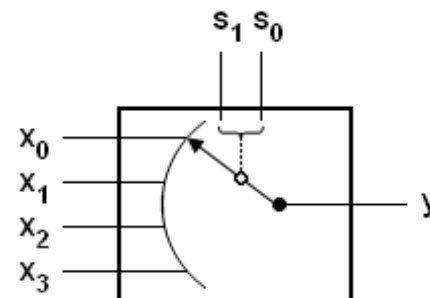
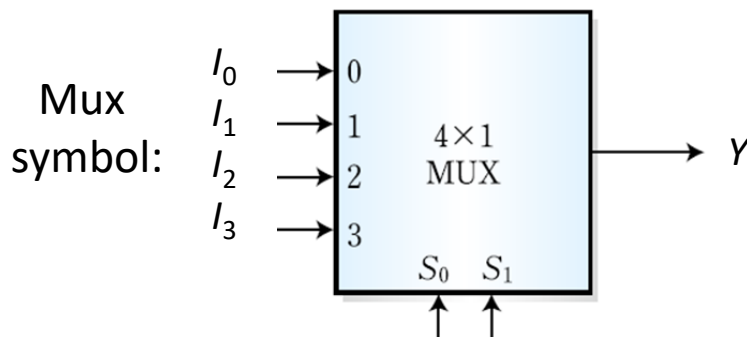
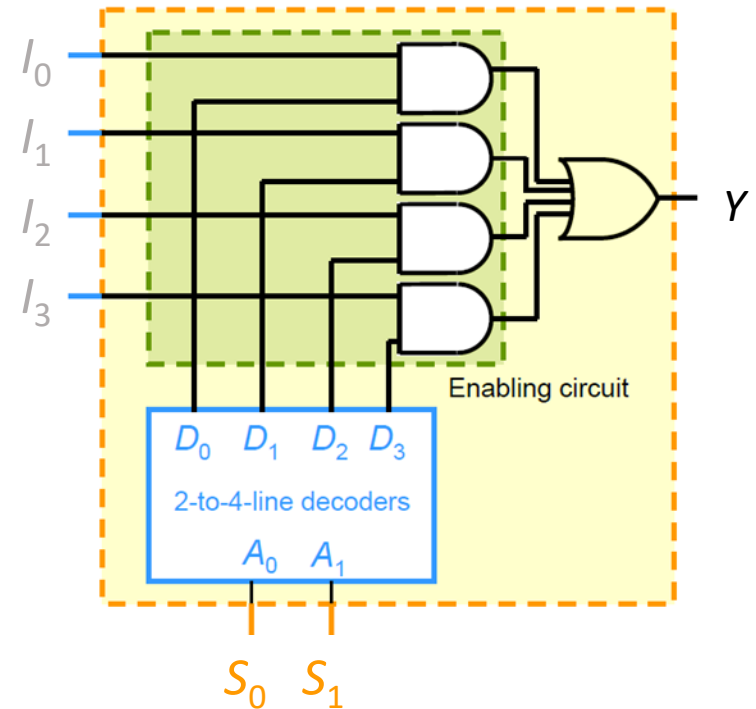
4-to-1-line MUX

$$m = 4$$

$$n = \log_2 m = 2$$

<i>Inputs</i>						<i>Output</i>
I_3	I_2	I_1	I_0	S_1	S_0	Y
X	X	X	X	0	0	I_0
X	X	X	X	0	1	I_1
X	X	X	X	1	0	I_2
X	X	X	X	1	1	I_3

$$Y(I_0, I_1, I_2, I_3, S_1, S_0) \\ = S_1' S_0' I_0 + S_1' S_0 I_1 + S_1 S_0' I_2 + S_1 S_0 I_3$$



Example

Use a 4-input MUX to realize the following function:

$$F(w,x,y,z) = \sum m(1,2,5,7,9,11,13)$$

Step 1: Plot the K-map
(no need to group)

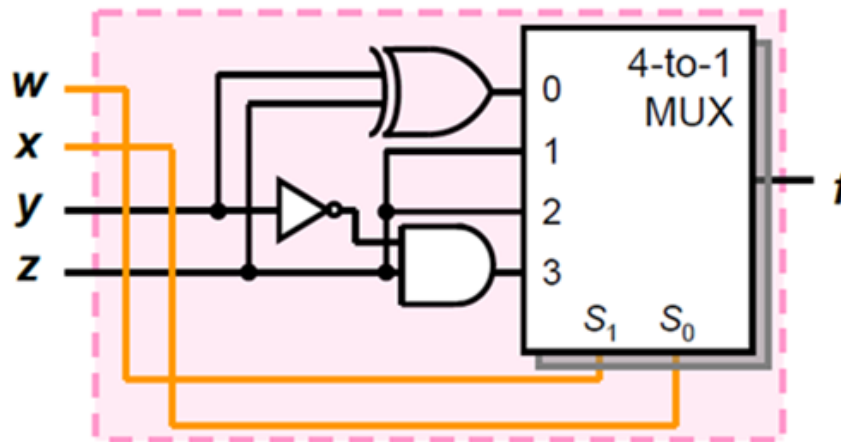
Step 2: Since $n = 2$
Pick w, x as selection variables
(i.e. w as S_1 , x as S_0)
Remaining variables: y, z

$wx \backslash yz$	00	01	11	10
00		1		1
01		1	1	
11		1		
10		1	1	

Step 3:

		yz				
		00	01	11	10	
wx	00		1		1	} $wx = 00, f = y'z + yz' = y \oplus z$
	01		1	1		
	11		1			
	10		1	1		

Step 4:



Exercise: instead of using 4-to1-line MUX, try to use 8-to-1-line MUX and 2-to-1-line MUX

NOTE: You can only produce a single output with a single MUX!!

With

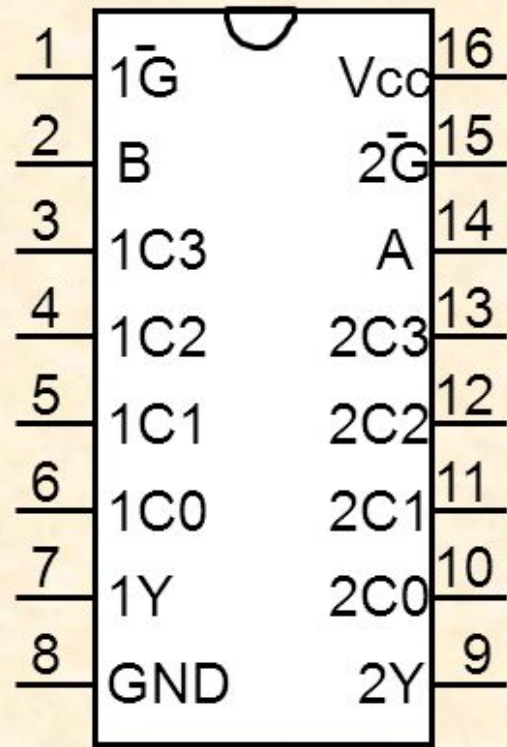
2-to-1 Mux

4-to-1 Mux

8-to-1 Mux

wxyz	F	$S_0=w$	$S_1=w, S_0=x$	$S_2=w, S_1=x, S_0=y$
0000	0	$I_0 = xz + y'z + x'yz'$	$I_0 = y \oplus z$	$I_0 = z$
0001	1			$I_1 = z'$
0010	1			
0011	0		$I_1 = z$	$I_2 = z$
0100	0			$I_3 = z$
0101	1			
0110	0			$I_4 = z$
0111	1			
1000	0	$I_1 = x'z + y'z$	$I_2 = z$	$I_5 = z$
1001	1			
1010	0			$I_6 = z$
1011	1			
1100	0		$I_3 = y'z$	$I_7 = 0$
1101	1			
1110	0			
1111	0			

Dual 4-to-1 line Mux IC - 74LS153

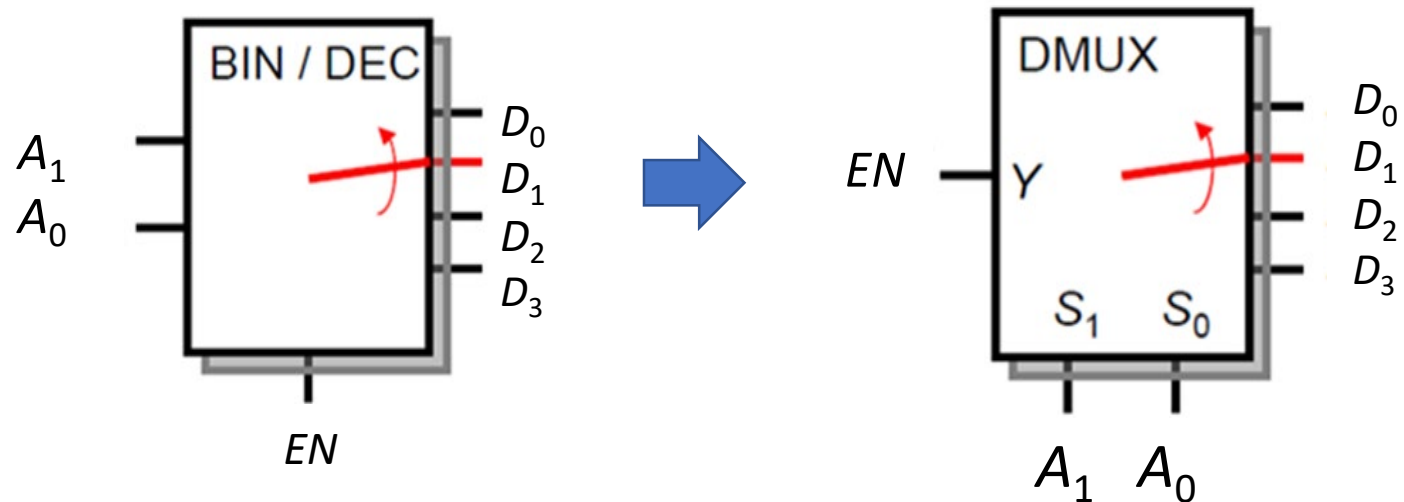


74LS153

B	A	C0	C1	C2	C3	G	Y
X	X	X	X	X	X	1	0
0	0	0	X	X	X	0	0
0	0	1	X	X	X	0	1
0	1	X	0	X	X	0	0
0	1	X	1	X	X	0	1
1	0	X	X	0	X	0	0
1	0	X	X	1	X	0	1
1	1	X	X	X	0	0	0
1	1	X	X	X	1	0	1

4.1.5 Demultiplexer (DMUX)

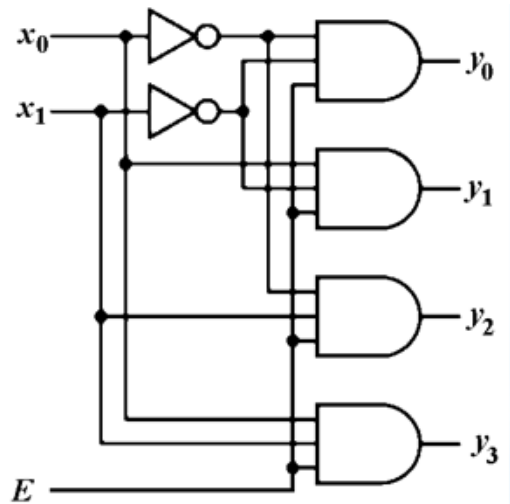
Remember the decoder with enable?



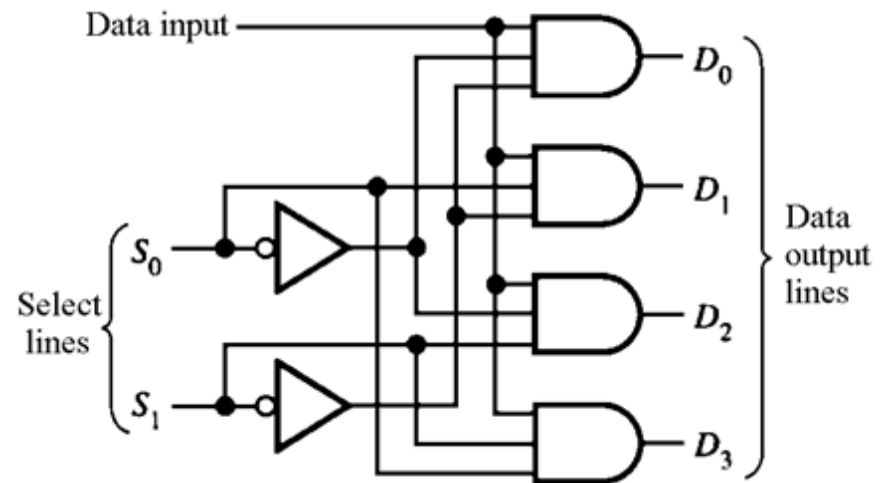
The decoder can perform demultiplexer if we take EN as the input line, A_i (input lines of decoder) as the selection inputs

Demultiplexer (DMUX)

A demultiplexer (DMUX) basically reverses the multiplexing function. A DMUX is a circuit that receives information on a single line and transmits this information on one of 2^n possible output lines. The selection of a specific output line is controlled by the bit pattern of n select lines. For this reason, the demultiplexer is also known as a data distributor.



Decoder with enable



Demultiplexer

Decoder can function as demultiplexer if the E line is taken as a data input line and input lines taken as the selection lines.

Typical MUX and DMUX application

- A MUX is a device that allows digital information from several sources to be routed onto a single line for transmission over that line to a common destination and a DMUX basically reverses the multiplexing function.

