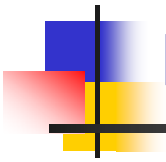
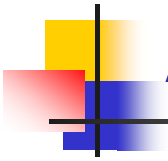


Modular Combinational Logic

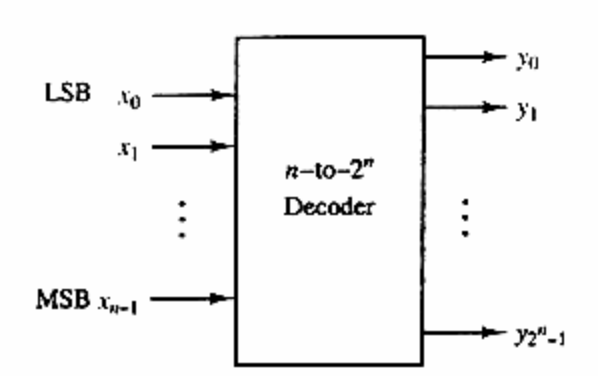


Agenda

- Decoder
- Encoder
- Multiplexers / Data Selectors
- Demultipluxers / Data Distributors

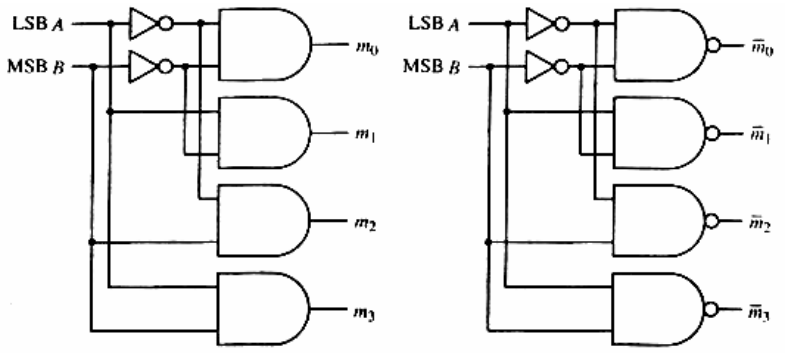
Decoder

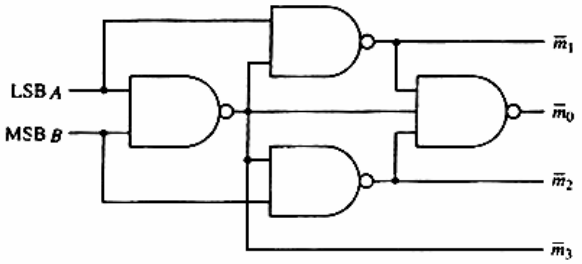
- Decoder is a multiple-output combinational logic network with *n* input line and 2^N output line.



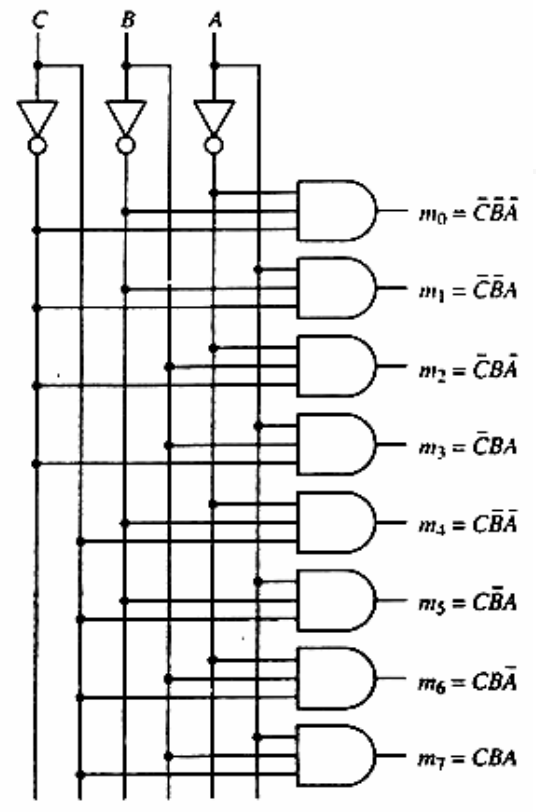
Decoder

$$m_0 = \bar{B}\bar{A}$$
$$m_1 = \bar{B}A$$
$$m_2 = B\bar{A}$$
$$m_3 = BA$$





Decoder



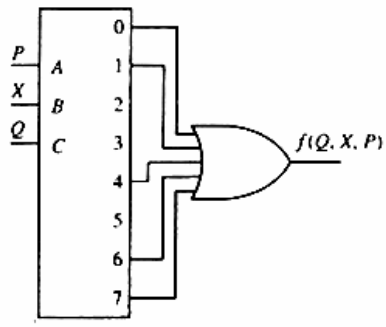
Decoder

Let us implement the following logic functions using decoders and logic gates.

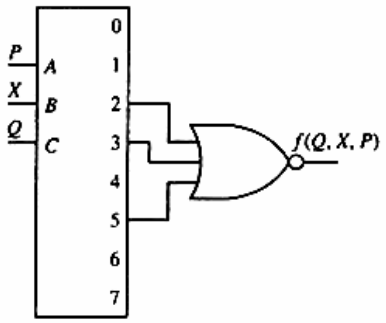
$$f(Q,X,P) = \sum m(0,1,4,6,7)$$
$$= \prod M(2,3,5)$$

We may implement the function in several ways:

1. Use a decoder (with active-high outputs) with an OR gate:
$$f(Q, X, P) = m_0 + m_1 + m_4 + m_6 + m_7$$
2. Use a decoder (with active-low outputs) with a NAND gate:
$$f(Q, X, P) = \overline{\bar{m}_0 \cdot \bar{m}_1 \cdot \bar{m}_4 \cdot \bar{m}_6 \cdot \bar{m}_7}$$
3. Use a decoder (with active-high outputs) with a NOR gate:
$$f(Q, X, P) = \overline{m_2 + m_3 + m_5}$$
4. Use a decoder (with active-low outputs) with an AND gate:
$$f(Q, X, P) = \bar{m}_2 \cdot \bar{m}_3 \cdot \bar{m}_5$$

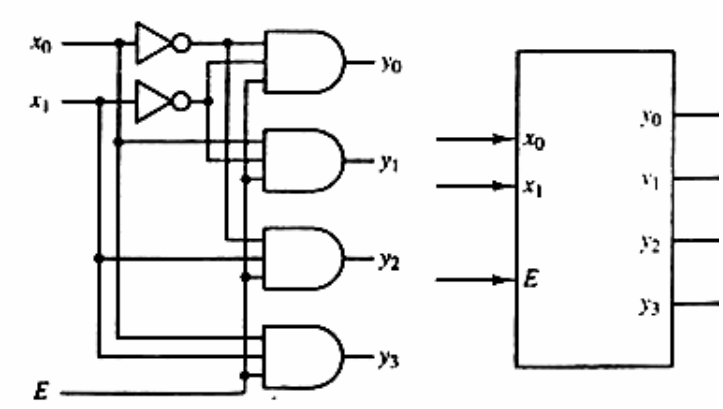


(a)

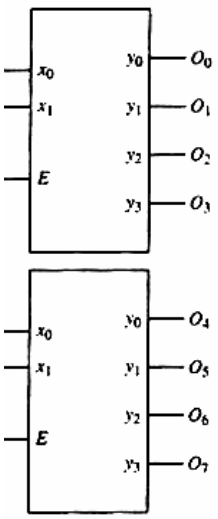


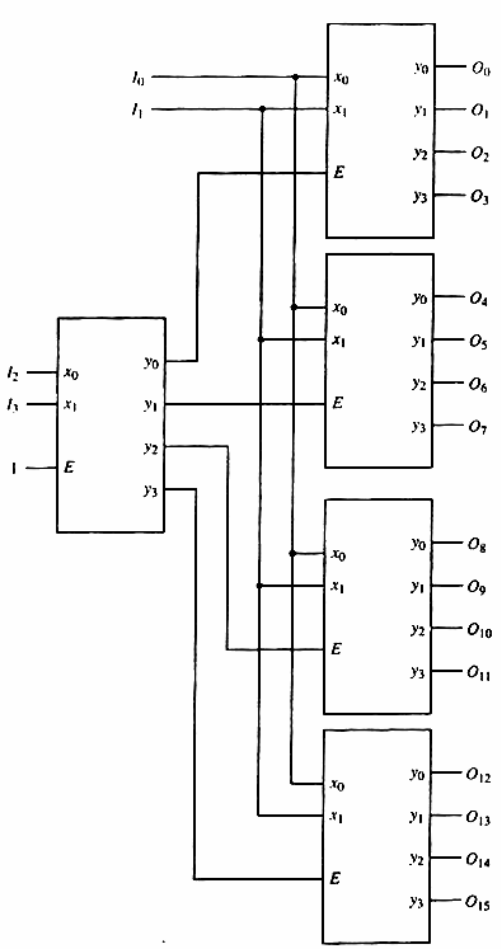
Decoder

- Enable Control Input



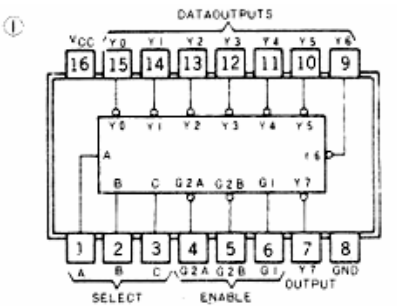
Decoder





Decoder

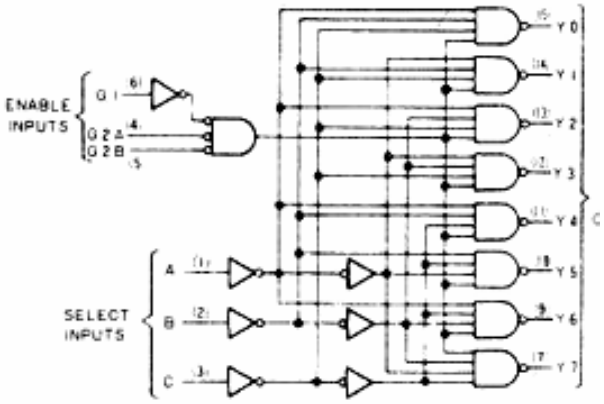
- Standard MSI Decoder



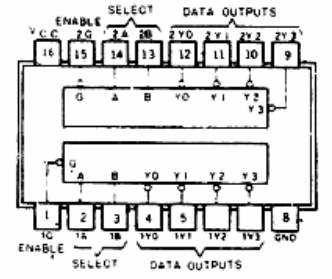
'S138 'LS138

INPUTS					OUTPUTS							
ENABLE		SELECT										
G1	G2*	C	B	A	Y0	Y1	Y2	Y3	Y4	Y5	Y6	Y7
X	H	X	X	X	H	H	H	H	H	H	H	H
L	X	X	X	X	H	H	H	H	H	H	H	H
H	L	L	L	L	L	H	H	H	H	H	H	H
H	L	L	L	H	H	L	H	H	H	H	H	H
H	L	L	H	L	H	H	L	H	H	H	H	H
H	L	L	H	H	H	H	H	L	H	H	H	H
H	L	H	L	L	H	H	H	H	L	H	H	H
H	L	H	L	H	H	H	H	H	H	L	H	H
H	L	H	H	L	H	H	H	H	H	H	L	H
H	L	H	H	H	H	H	H	H	H	H	H	L

• G2 = G2A + G2B



Decoder

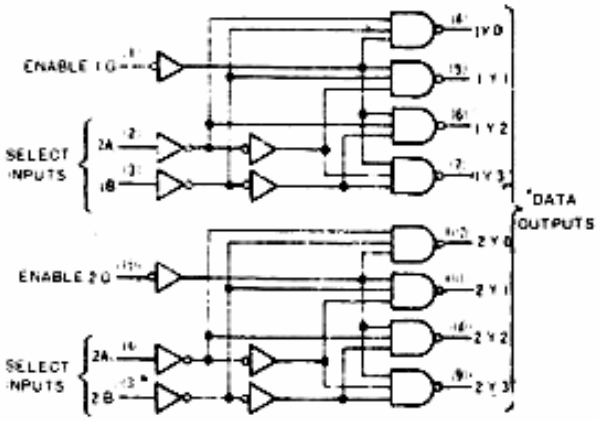


'S139 'LS139

(EACH DECODER/DEMULTIPLEXER)

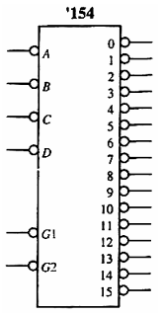
INPUTS			OUTPUTS			
ENABLE	SELECT					
G	B	A	Y0	Y1	Y2	Y3
H	X	X	H	H	H	H
L	L	L	L	H	H	H
L	L	H	H	L	H	H
L	H	L	H	H	L	H
L	H	H	H	H	H	L

Functional Block Diagram

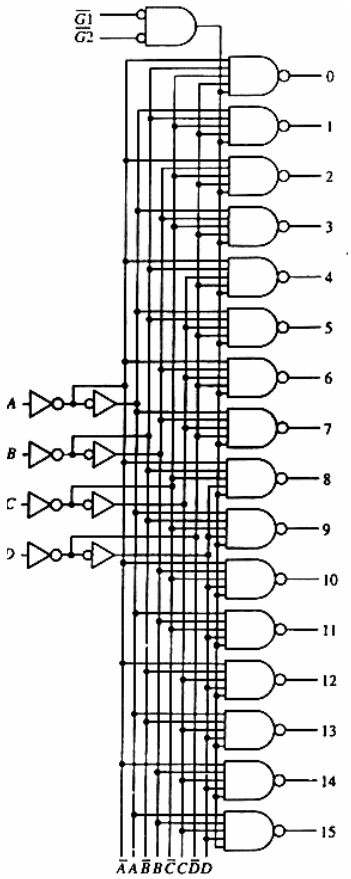


Decoder

- Standard MSI Decoder

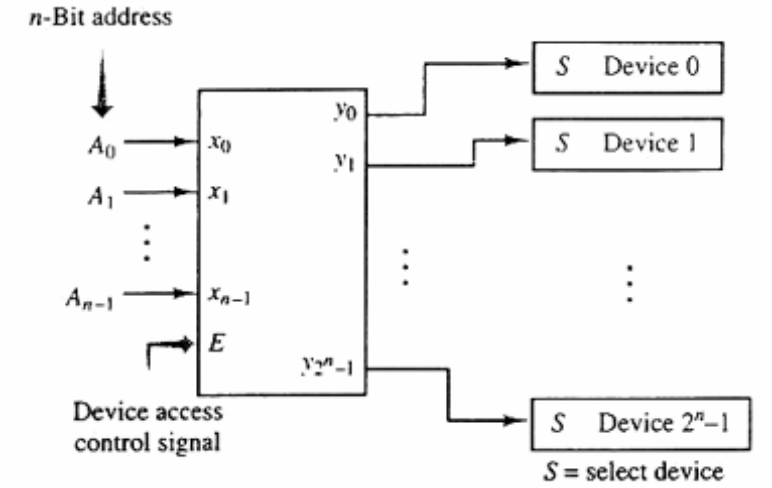


Inputs						Outputs													
$\overline{G1}$	$\overline{G2}$	D	C	B	A	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
L	L	L	L	L	L	L	H	H	H	H	H	H	H	H	H	H	H	H	H	H	H
L	L	L	L	L	H	H	L	H	H	H	H	H	H	H	H	H	H	H	H	H	H
L	L	L	L	H	L	H	H	L	H	H	H	H	H	H	H	H	H	H	H	H	H
L	L	L	L	H	H	H	H	H	L	H	H	H	H	H	H	H	H	H	H	H	H
L	L	L	H	L	L	H	H	H	H	L	H	H	H	H	H	H	H	H	H	H	H
L	L	L	H	L	H	H	H	H	H	H	L	H	H	H	H	H	H	H	H	H	H
L	L	L	H	H	L	H	H	H	H	H	H	L	H	H	H	H	H	H	H	H	H
L	L	L	H	H	H	H	H	H	H	H	H	H	L	H	H	H	H	H	H	H	H
L	L	H	L	L	L	H	H	H	H	H	H	H	H	L	H	H	H	H	H	H	H
L	L	H	L	L	H	H	H	H	H	H	H	H	H	H	L	H	H	H	H	H	H
L	L	H	L	H	L	H	H	H	H	H	H	H	H	H	H	L	H	H	H	H	H
L	L	H	L	H	H	H	H	H	H	H	H	H	H	H	H	H	L	H	H	H	H
L	L	H	H	L	L	H	H	H	H	H	H	H	H	H	H	H	H	L	H	H	H
L	L	H	H	L	H	H	H	H	H	H	H	H	H	H	H	H	H	H	L	H	H
L	L	H	H	H	L	H	H	H	H	H	H	H	H	H	H	H	H	H	H	L	H
L	L	H	H	H	H	H	H	H	H	H	H	H	H	H	H	H	H	H	H	H	L
H	L	×	×	×	×	H	H	H	H	H	H	H	H	H	H	H	H	H	H	H	H
L	H	×	×	×	×	H	H	H	H	H	H	H	H	H	H	H	H	H	H	H	H
H	H	×	×	×	×	H	H	H	H	H	H	H	H	H	H	H	H	H	H	H	H



Decoder

- Decoder Application
  - Address Decoding



Decoder

- Minterm Generation

Realize the following functions using a 74154 and logic gates:

$$f_1(W,X,Y,Z) = \sum m(1,9,12,15)$$

$$f_2(W,X,Y,Z) = \sum m(0,1,2,3,4,5,7,8,10,11,12,13,14,15)$$

Using implementations 2 and 3 from Example 4.1:

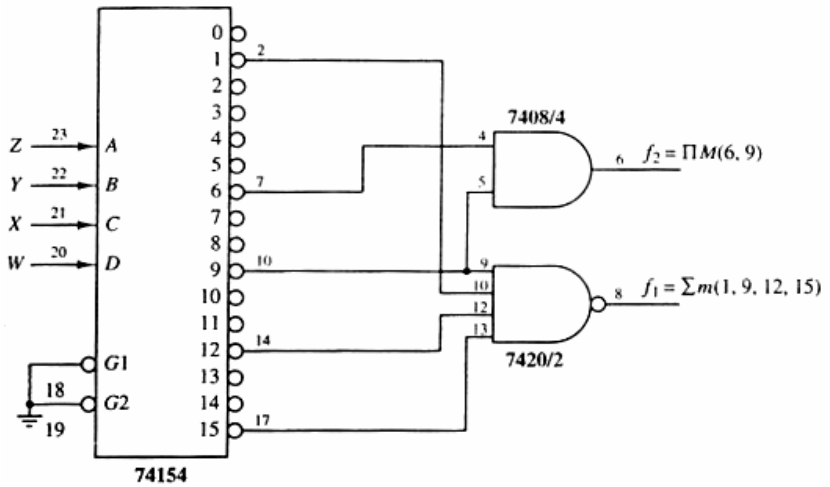
$$f_1(W, X, Y, Z) = \overline{\bar{m}_1 \bar{m}_9 \bar{m}_{12} \bar{m}_{15}}$$

and

$$f_2(W, X, Y, Z) = \bar{m}_6 \cdot \bar{m}_9$$

Decoder

- Minterm Generation



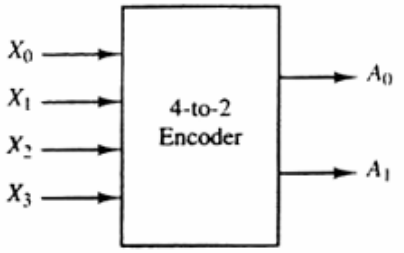
Encoder

- Encoder is a combinational logic module that assigns a unique output code.
- Each input signal applied to the device.

Design an encoder for four input lines if one and only one is active at any moment in time. See Fig. 4.17a.

Let us define the code:

	A_1	A_0
$X_0 \rightarrow$	0	0
$X_1 \rightarrow$	0	1
$X_2 \rightarrow$	1	0
$X_3 \rightarrow$	1	1



Encoder

X_3	X_2	X_1	X_0	A_1	A_0
0	0	0	0	*d*	*d*
0	0	0	1	0	0
0	0	1	0	0	1
0	0	1	1	*d*	*d*
0	1	0	0	1	0
0	1	0	1	*d*	*d*
0	1	1	0	*d*	*d*
0	1	1	1	*d*	*d*
1	0	0	0	1	1
1	0	0	1	*d*	*d*
1	0	1	0	*d*	*d*
1	0	1	1	*d*	*d*
1	1	0	0	*d*	*d*
1	1	0	1	*d*	*d*
1	1	1	0	*d*	*d*
1	1	1	1	*d*	*d*

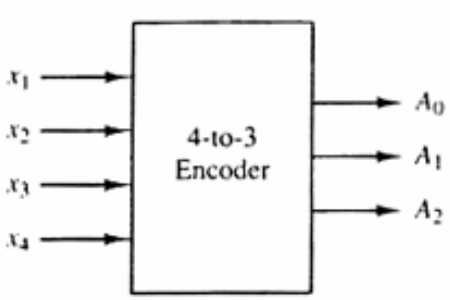
$$A_1 = X_3 + X_2$$
$$A_0 = X_3 + X_1$$

Encoder

Design a four-line encoder that outputs a zero code unless one and only one input line is active.

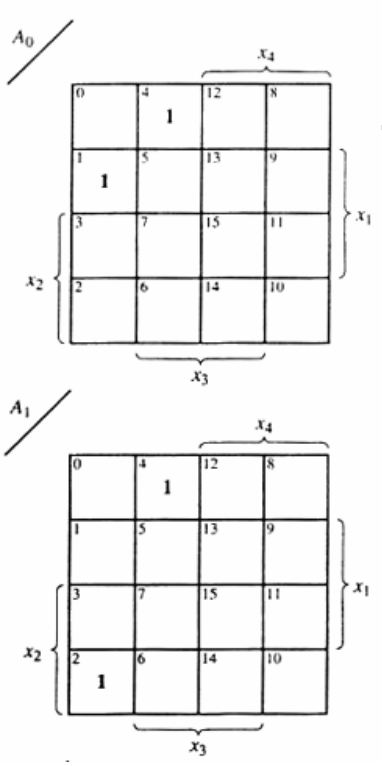
Let us define the code:

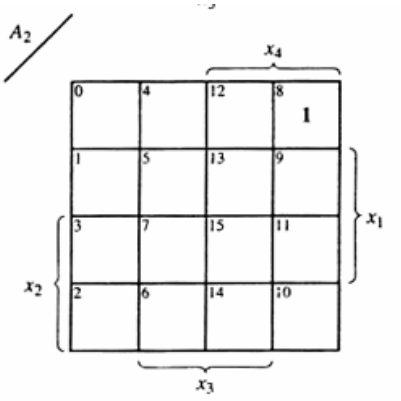
	A_2	A_1	A_0
$X_1 \rightarrow$	0	0	1
$X_2 \rightarrow$	0	1	0
$X_3 \rightarrow$	0	1	1
$X_4 \rightarrow$	1	0	0
All others $\rightarrow$	0	0	0



Encoder

X_4	X_3	X_2	X_1	A_2	A_1	A_0
0	0	0	0	0	0	0
0	0	0	1	0	0	1
0	0	1	0	0	1	0
0	0	1	1	0	0	0
0	1	0	0	0	1	1
0	1	0	1	0	0	0
0	1	1	0	0	0	0
0	1	1	1	0	0	0
1	0	0	0	1	0	0
1	0	0	1	0	0	0
1	0	1	0	0	0	0
1	0	1	1	0	0	0
1	1	0	0	0	0	0
1	1	0	1	0	0	0
1	1	1	0	0	0	0
1	1	1	1	0	0	0



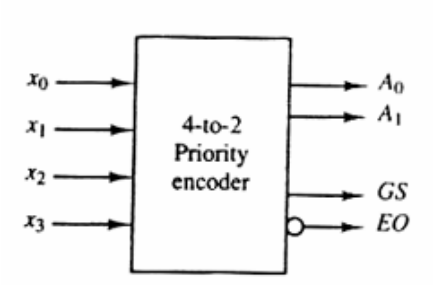


$$A_2 = X_4\bar{X}_3\bar{X}_2\bar{X}_1$$
$$A_1 = \bar{X}_4\bar{X}_3X_2\bar{X}_1 + \bar{X}_4X_3\bar{X}_2\bar{X}_1$$
$$A_0 = \bar{X}_4\bar{X}_3\bar{X}_2X_1 + \bar{X}_4X_3\bar{X}_2\bar{X}_1$$

Encoder

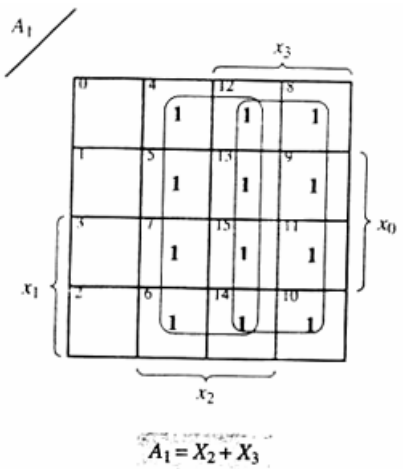
- Priority Encoder

	A_1	A_0
$X_0 \rightarrow$	0	0
$X_1 \rightarrow$	0	1
$X_2 \rightarrow$	1	0
$X_3 \rightarrow$	1	1

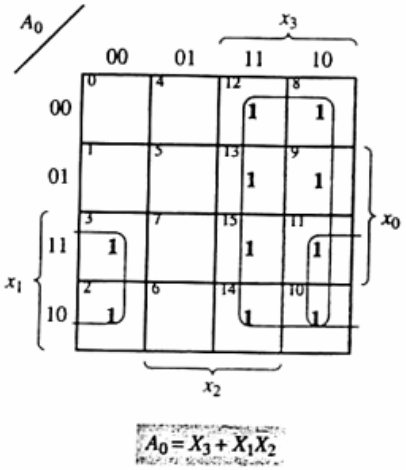


Inputs				Outputs			
X_3	X_2	X_1	X_0	A_1	A_0	GS	EO
0	0	0	0	0	0	0	1
0	0	0	1	0	0	1	0
0	0	1	0	0	1	1	0
0	0	1	1	0	1	1	0
0	1	0	0	1	0	1	0
0	1	0	1	1	0	1	0
0	1	1	0	1	0	1	0
0	1	1	1	1	0	1	0
1	0	0	0	1	1	1	0
1	0	0	1	1	1	1	0
1	0	1	0	1	1	1	0
1	0	1	1	1	1	1	0
1	1	0	0	1	1	1	0
1	1	0	1	1	1	1	0
1	1	1	0	1	1	1	0
1	1	1	1	1	1	1	0

Encoder



$A_1 = X_2 + X_3$



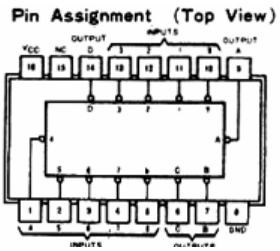
$A_0 = X_3 + X_1\bar{X}_2$

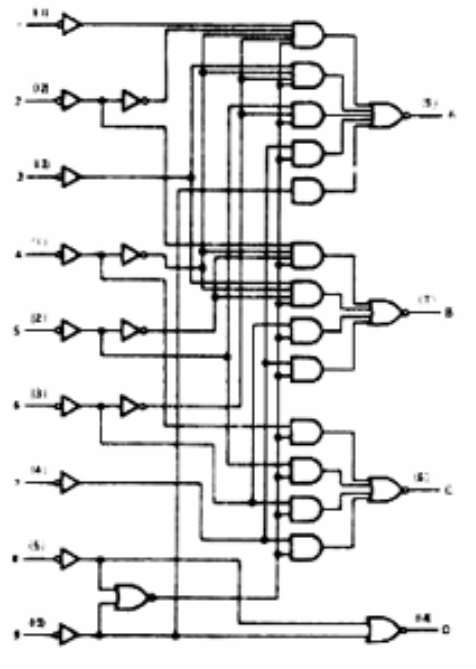
$$A_1 = X_2 + X_3$$
$$A_0 = X_3 + X_1\bar{X}_2$$

$$EO = \overline{GS} = \overline{X_3 + X_2 + X_1 + X_0}$$

Encoder

- Standard MSI Encoder

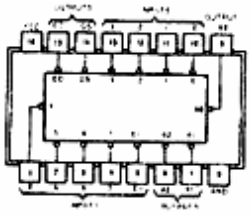




'147

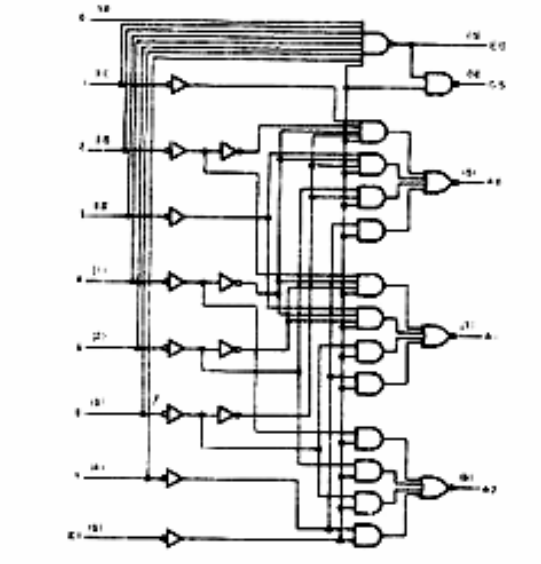
INPUTS									OUTPUTS			
1	2	3	4	5	6	7	8	9	D	C	B	A
H	H	H	H	H	H	H	H	H	H	H	H	H
X	X	X	X	X	X	X	X	L	L	H	H	L
X	X	X	X	X	X	X	L	H	L	H	H	H
X	X	X	X	X	X	L	H	H	H	L	L	L
X	X	X	X	X	L	H	H	H	H	L	L	H
X	X	X	X	L	H	H	H	H	H	L	H	L
X	X	X	L	H	H	H	H	H	H	L	H	H
X	X	L	H	H	H	H	H	H	H	H	L	L
X	L	H	H	H	H	H	H	H	H	H	L	H
L	H	H	H	H	H	H	H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

Encoder



'148

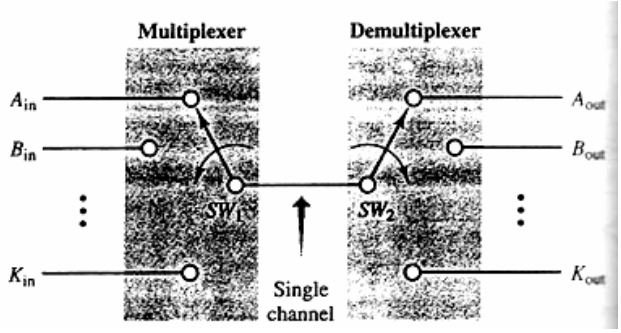
INPUTS								OUTPUTS				
EI	0	1	2	3	4	5	6	7	A2	A1	A0	GS	EO
H	X	X	X	X	X	X	X	X	H	H	H	H	H
L	H	H	H	H	H	H	H	H	H	H	H	H	L
L	X	X	X	X	X	X	X	L	L	L	L	L	H
L	X	X	X	X	X	X	L	H	L	L	H	L	H
L	X	X	X	X	X	L	H	H	L	H	L	L	H
L	X	X	X	X	L	H	H	H	L	H	H	L	H
L	X	X	X	L	H	H	H	H	H	L	L	L	H
L	X	X	L	H	H	H	H	H	H	L	H	L	H
L	X	L	H	H	H	H	H	H	H	H	L	L	H
L	L	H	H	H	H	H	H	H	H	H	H	L	H

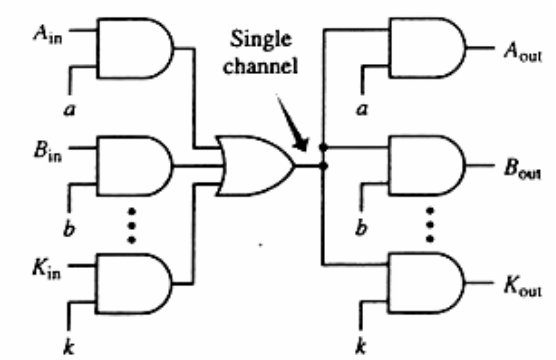


148 8-LINE-TO-3LINE PRIORITY ENCODER

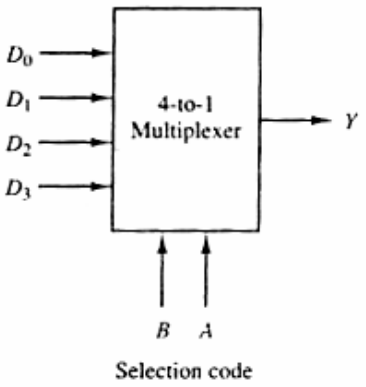
Multiplexers / Data Selectors

- Multiplexer is a modular device that selects one of many input lines to appear on a single output line.





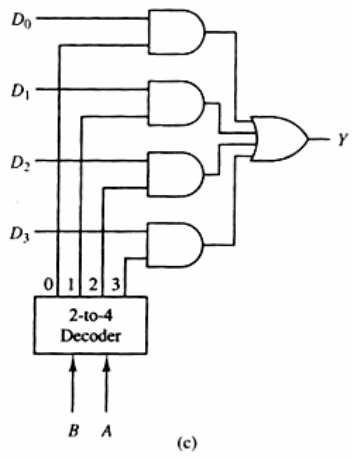
Multiplexers / Data Selectors

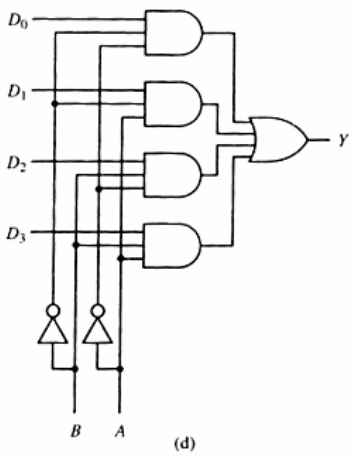


B	A	Y
0	0	D_0
0	1	D_1
1	0	D_2
1	1	D_3

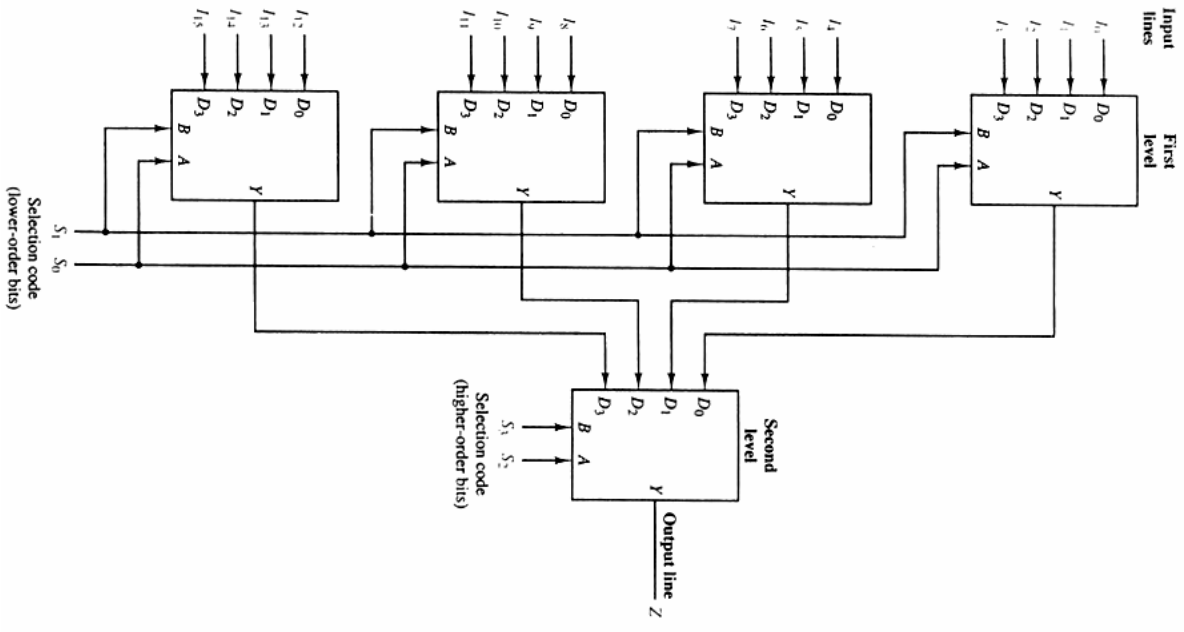
$$Y = (\bar{B}\bar{A})D_0 + (\bar{B}A)D_1 + (B\bar{A})D_2 + (BA)D_3$$

$$Y = \sum_{i=0}^{3} m_i D_i$$



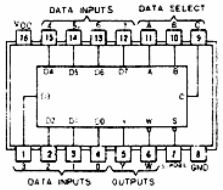


Multiplexers / Data Selectors



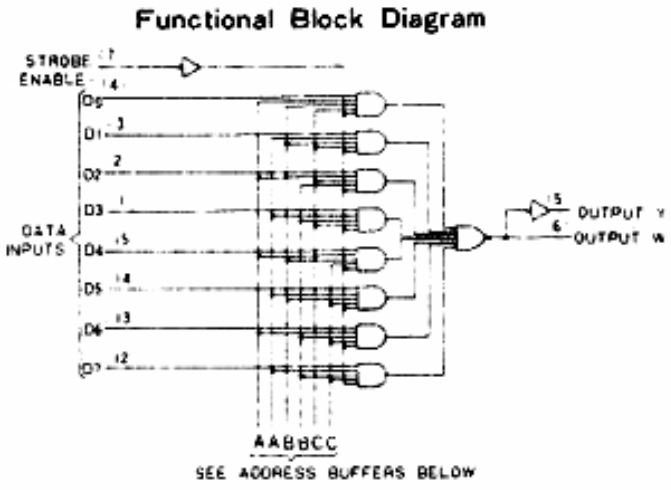
Multiplexers / Data Selectors

- Standard MSI Multiplexers



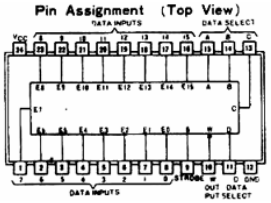
'151A, 'LS151, 'S151

INPUTS				OUTPUTS	
SELECT			STROBE		
C	B	A	S	Y	W
X	X	X	H	L	H
L	L	L	L	D0	$\overline{D0}$
L	L	H	L	D1	$\overline{D1}$
L	H	L	L	D2	$\overline{D2}$
L	H	H	L	D3	$\overline{D3}$
H	L	L	L	D4	$\overline{D4}$
H	L	H	L	D5	$\overline{D5}$
H	H	L	L	D6	$\overline{D6}$
H	H	H	L	D7	$\overline{D7}$



Multiplexers / Data Selectors

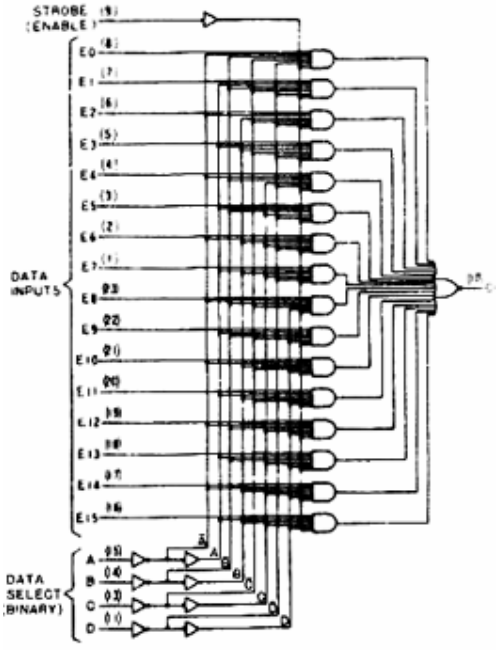
Pin Assignment (Top View)



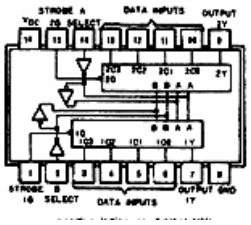
'150

INPUTS					OUTPUT
SELECT				STROBE	
D	C	B	A	S	W
X	X	X	X	H	H
L	L	L	L	L	E0
L	L	L	H	L	E1
L	L	H	L	L	E2
L	L	H	H	L	E3
L	H	L	L	L	E4
L	H	L	H	L	E5
L	H	H	L	L	E6
L	H	H	H	L	E7
H	L	L	L	L	E8
H	L	L	H	L	E9
H	L	H	L	L	E10
H	L	H	H	L	E11
H	H	L	L	L	E12
H	H	L	H	L	E13
H	H	H	L	L	E14
H	H	H	H	L	E15

Functional Block Diagram



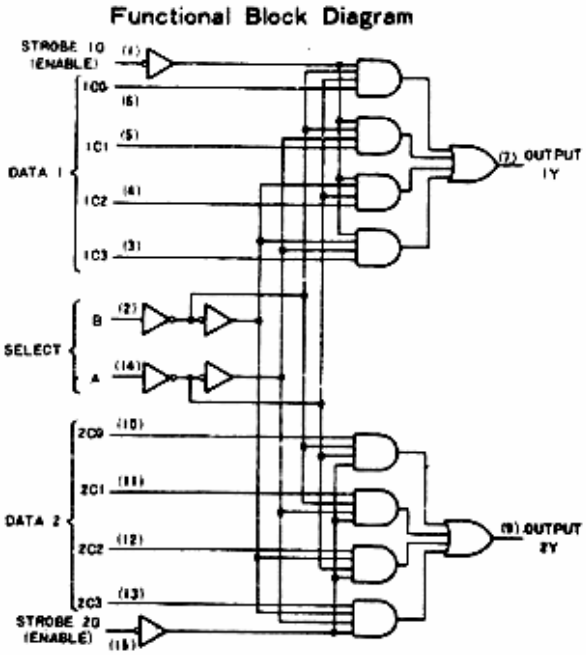
Multiplexers / Data Selectors



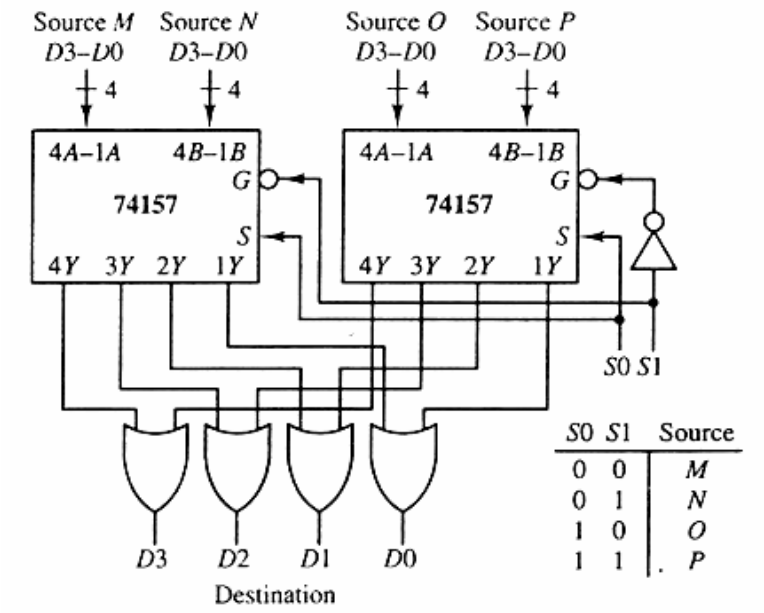
'153, 'S153, 'LS153, 'L153

SELECT INPUTS		DATA INPUTS				STROBE	OUTPUT
B	A	C0	C1	C2	C3	G	Y
X	X	X	X	X	X	H	L
L	L	L	X	X	X	L	L
L	L	H	X	X	X	L	H
L	H	X	L	X	X	L	L
L	H	X	H	X	X	L	H
H	L	X	X	L	X	L	L
H	L	X	X	H	X	L	H
H	H	X	X	X	L	L	L
H	H	X	X	X	H	L	H

Functional Block Diagram



Multiplexers / Data Selectors



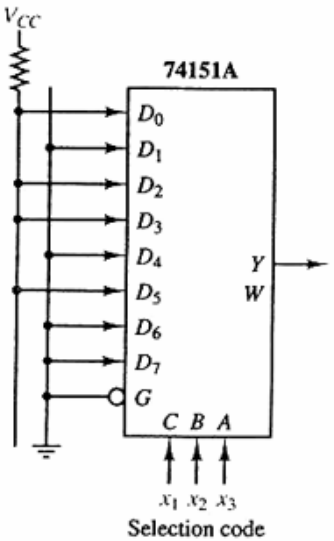
Multiplexers / Data Selectors

- Application of Multiplexers

Use a 74151A to implement
$f(x_1,x_2,x_3) = \sum m(0,2,3,5)$.

i	C	B	A	Y	
	x_1	x_2	x_3	f	
0	0	0	0	1	$D_0 = 1$
1	0	0	1	0	$D_1 = 0$
2	0	1	0	1	$D_2 = 1$
3	0	1	1	1	$D_3 = 1$
4	1	0	0	0	$D_4 = 0$
5	1	0	1	1	$D_5 = 1$
6	1	1	0	0	$D_6 = 0$
7	1	1	1	0	$D_7 = 0$

(a)



Multiplexers / Data Selectors

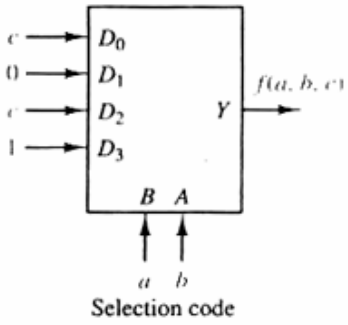
Implement $f(a,b,c) = ab + \bar{b}c$ using the 4-to-1 multiplexer of Fig. 4.23.

$$f(a,b,c) = ab + \bar{b}c$$
$$= ab\bar{c} + abc + \bar{a}\bar{b}c + a\bar{b}c$$

$$f(a,b,c) = \bar{a}\bar{b}(c) + a\bar{b}(c) + ab(\bar{c} + c)$$

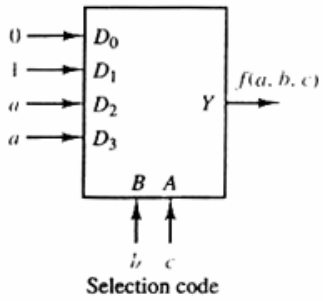
a	b	$f(a,b,c)$	MUX Inputs
0	0	c	$D_0 = c$
0	1	0	$D_1 = 0$
1	0	c	$D_2 = c$
1	1	1	$D_3 = 1$

(a)



b	c	$f(a,b,c)$	MUX Inputs
0	0	0	$D_0 = 0$
0	1	1	$D_1 = 1$
1	0	a	$D_2 = a$
1	1	a	$D_3 = a$

(c)



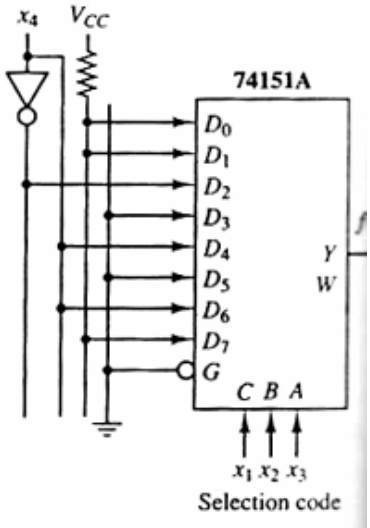
Multiplexers / Data Selectors

Implement
$f(X_1, X_2, X_3, X_4) = \sum m(0, 1, 2, 3, 4, 9, 13, 14, 15)$
using a 74151A.

0, 0 } logic 0 — 0, 1 } variable X_4

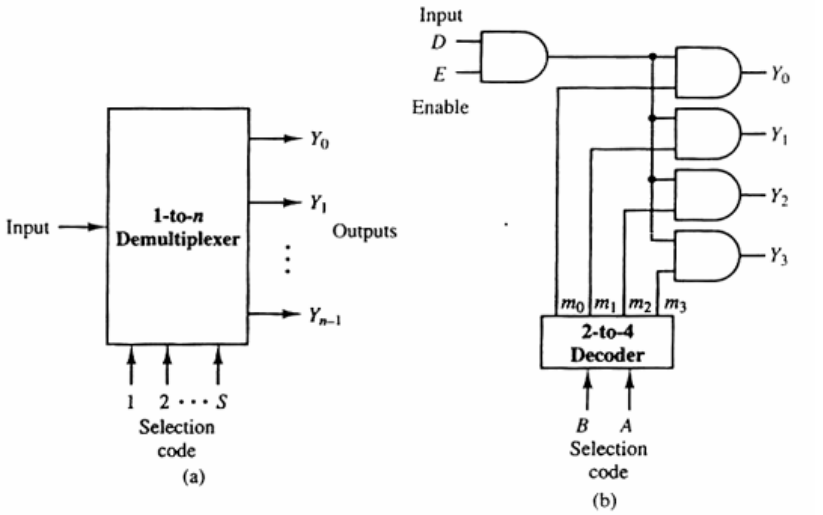
1, 1 } logic 1 — 1, 0 } variable $\bar{X}_4$

	C	B	A				Y
i	X_1	X_2	X_3	X_4	f	f	
0	0	0	0	0	1		
	0	0	0	1	1	1	$D_0 = 1$
1	0	0	1	0	1		
	0	0	1	1	1	1	$D_1 = 1$
2	0	1	0	0	1		
	0	1	0	1	0	$\bar{X}_4$	$D_2 = \bar{X}_4$
3	0	1	1	0	0		
	0	1	1	1	0	0	$D_3 = 0$
4	1	0	0	0	0		
	1	0	0	1	1	X_4	$D_4 = X_4$
5	1	0	1	0	0		
	1	0	1	1	0	0	$D_5 = 0$
6	1	1	0	0	0		
	1	1	0	1	1	X_4	$D_6 = X_4$
7	1	1	1	0	1		
	1	1	1	1	1	1	$D_7 = 1$

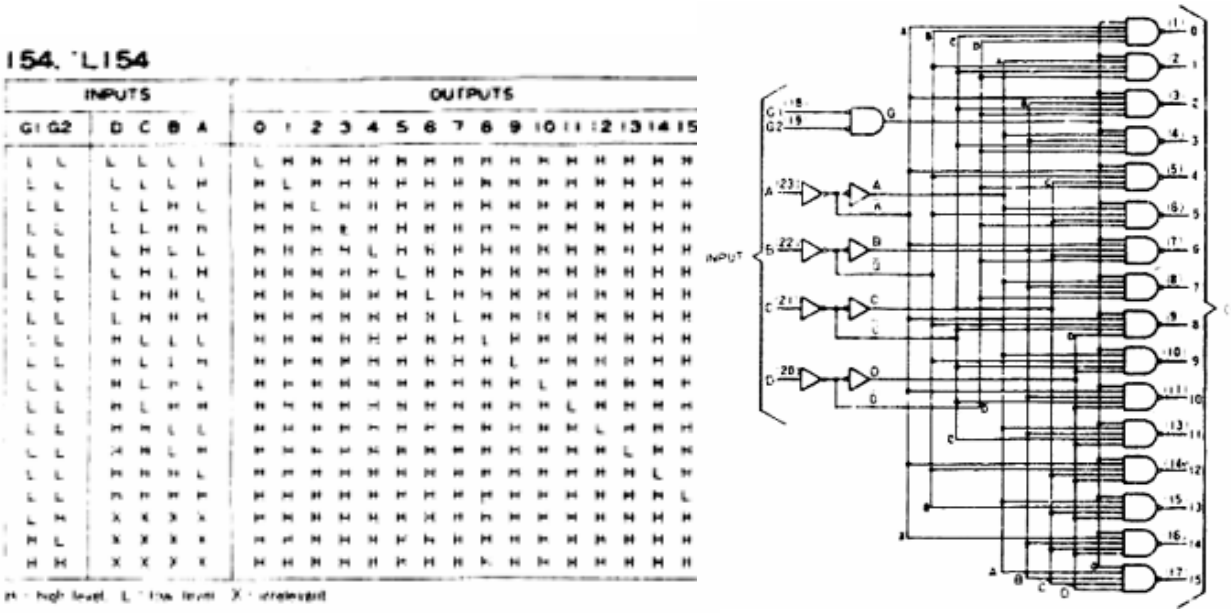


Demultiplexers / Data Distributors

- Demultiplexers using a selection code to specify which input line to connect to output line.



Demultiplexers / Data Distributors



Demultiplexers / Data Distributors