Digital Logic Design ENEE 244-010x

Lecture 15

Announcements

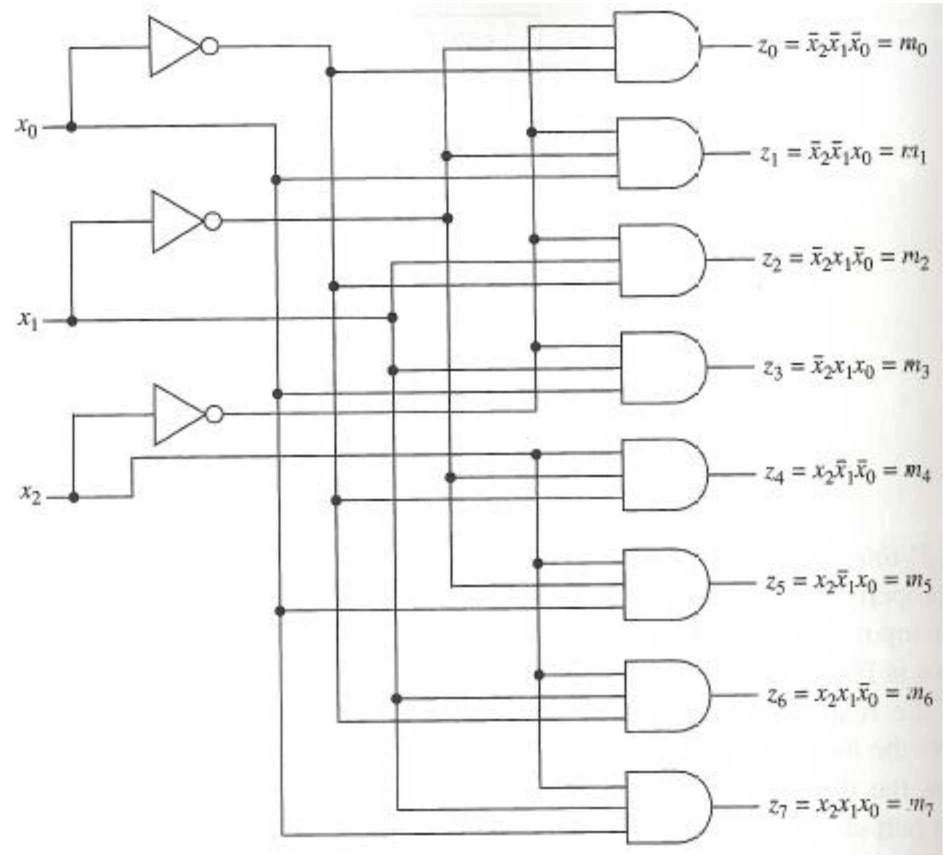
- Homework 7 due Wednesday 11/4
- Coming up: Midterm on 11/11
  - Topics for Midterm posted online
  - Review problems will be posted by Tuesday night

Agenda

- Last time:
  - Decimal Adders (5.2)
  - Comparators (5.3)
  - Decoders (5.4)
  - Encoders (5.5)
  - Multiplexers (5.6)

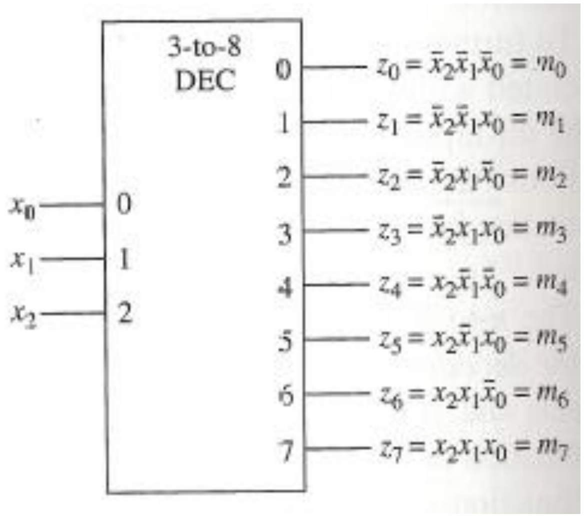
- This time:
  - Logic Design with Decoders and Multiplexers (5.4, 5.6)
  - Start Programmable Logic Devices (PLD) (5.7)

Logic Design Using Decoders



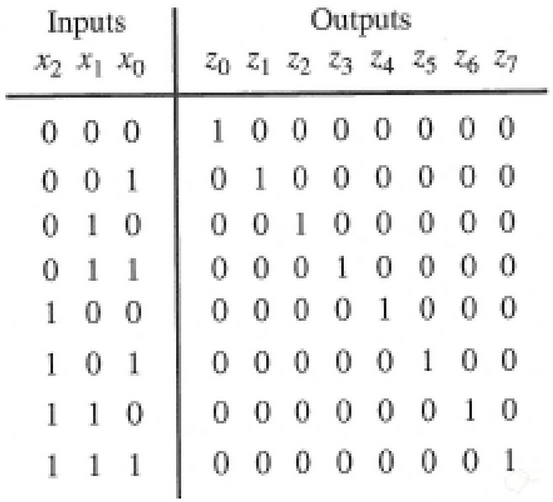
x_0
x_1
x_2
$z_0 = \bar{x}_2\bar{x}_1\bar{x}_0 = m_0$
$z_1 = \bar{x}_2\bar{x}_1x_0 = m_1$
$z_2 = \bar{x}_2x_1\bar{x}_0 = m_2$
$z_3 = \bar{x}_2x_1x_0 = m_3$
$z_4 = x_2\bar{x}_1\bar{x}_0 = m_4$
$z_5 = x_2\bar{x}_1x_0 = m_5$
$z_6 = x_2x_1\bar{x}_0 = m_6$
$z_7 = x_2x_1x_0 = m_7$

Logic Diagram



3-to-8
DEC
x_0 0
x_1 1
x_2 2
0 $z_0 = \bar{x}_2\bar{x}_1\bar{x}_0 = m_0$
1 $z_1 = \bar{x}_2\bar{x}_1x_0 = m_1$
2 $z_2 = \bar{x}_2x_1\bar{x}_0 = m_2$
3 $z_3 = \bar{x}_2x_1x_0 = m_3$
4 $z_4 = x_2\bar{x}_1\bar{x}_0 = m_4$
5 $z_5 = x_2\bar{x}_1x_0 = m_5$
6 $z_6 = x_2x_1\bar{x}_0 = m_6$
7 $z_7 = x_2x_1x_0 = m_7$

Symbol



Inputs			Outputs							
x_2	x_1	x_0	z_0	z_1	z_2	z_3	z_4	z_5	z_6	z_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

Truth Table

Logic Design Using Decoders

- An n-to-2^n line decoder is a minterm generator.
- By using or-gates in conjunction with an n-to-2^n line decoder, realizations of Boolean functions are possible.
- Do not correspond to minimal sum-of-products.
- Are simple to produce. Particularly convenient when several functions of the same variable have to be realized.

Minterms using OR Gates

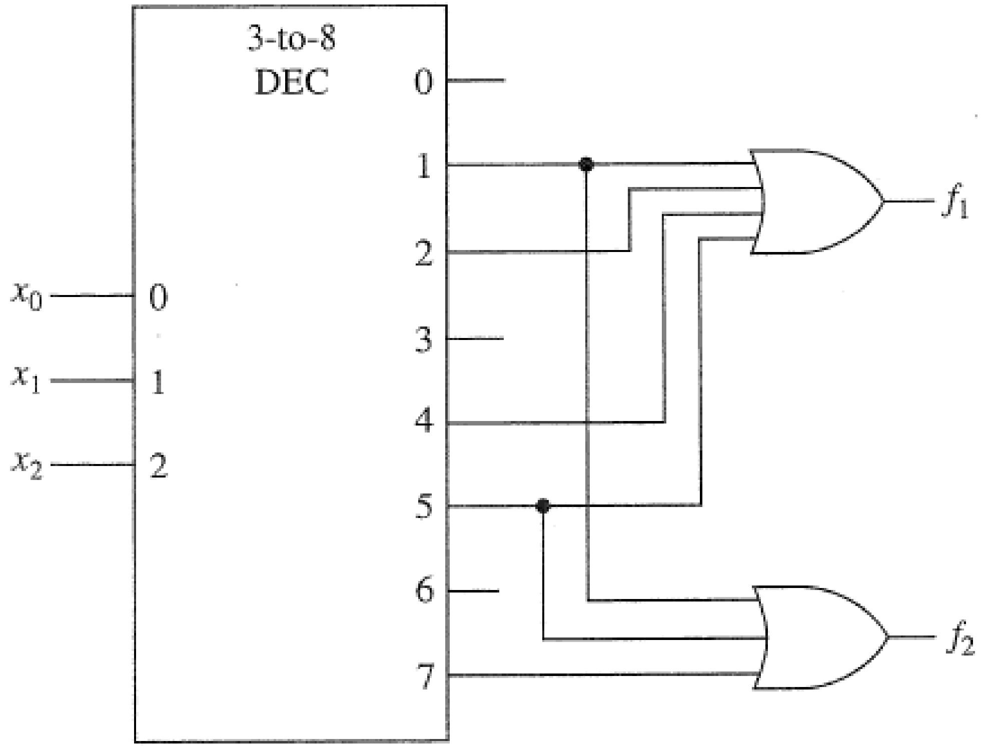


Figure 5.19 Realization of the Boolean expressions $f_1(x_2,x_1,x_0) = \Sigma m(1,2,4,5)$ and $f_2(x_2,x_1,x_0) = \Sigma m(1,5,7)$ with a 3-to-8-

Minterms using NOR Gates

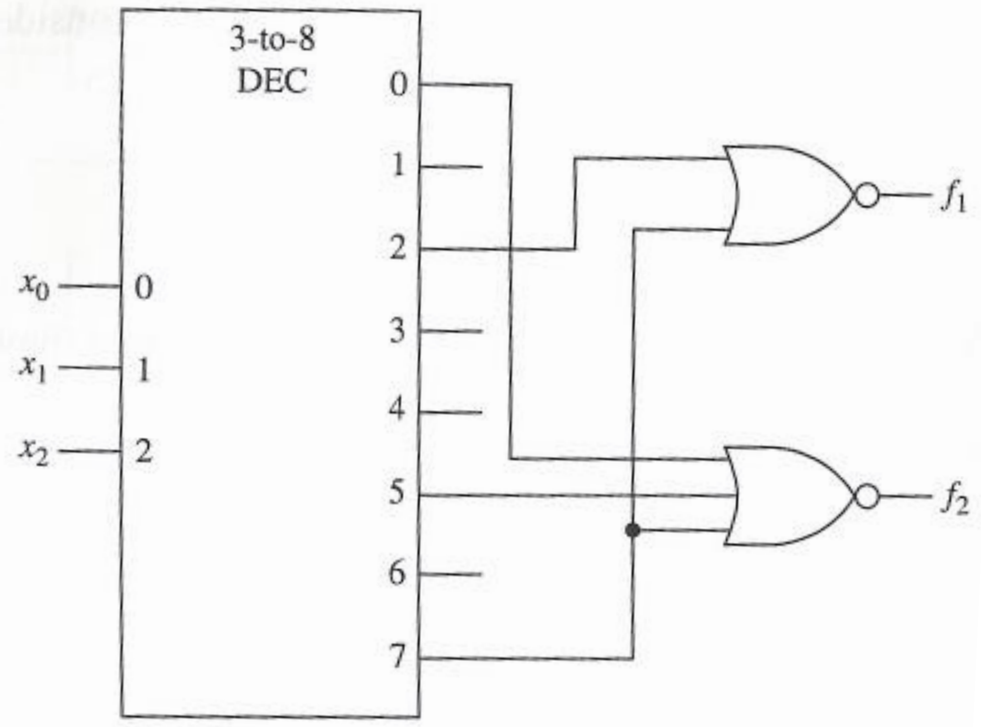
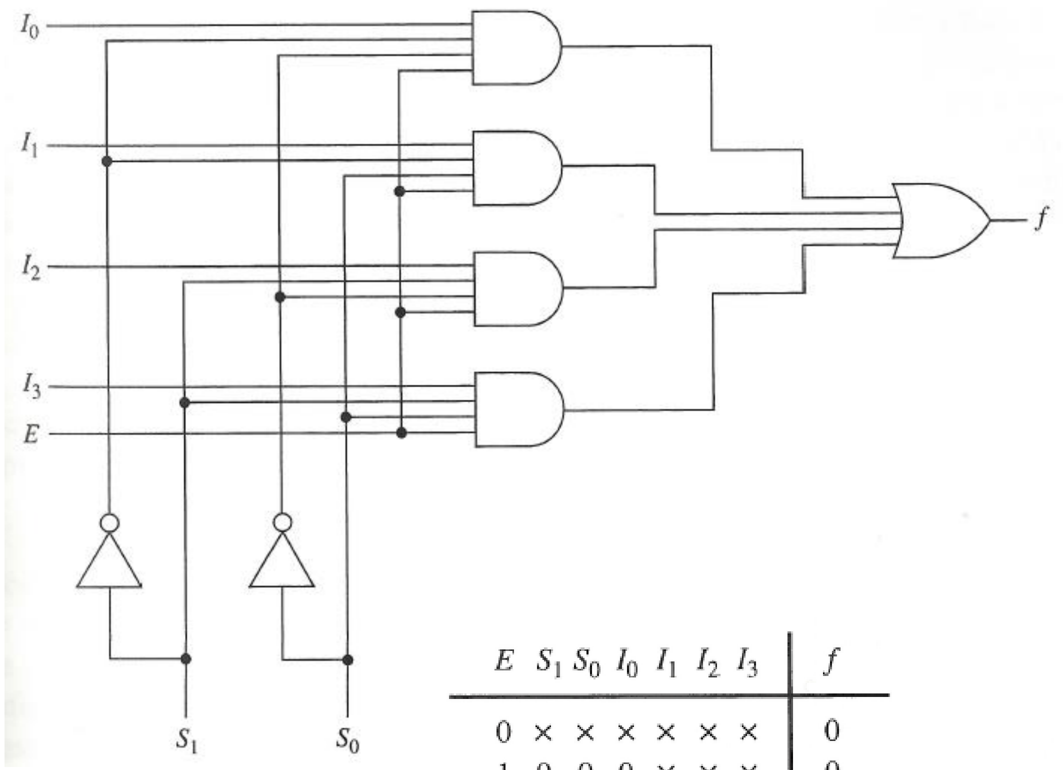
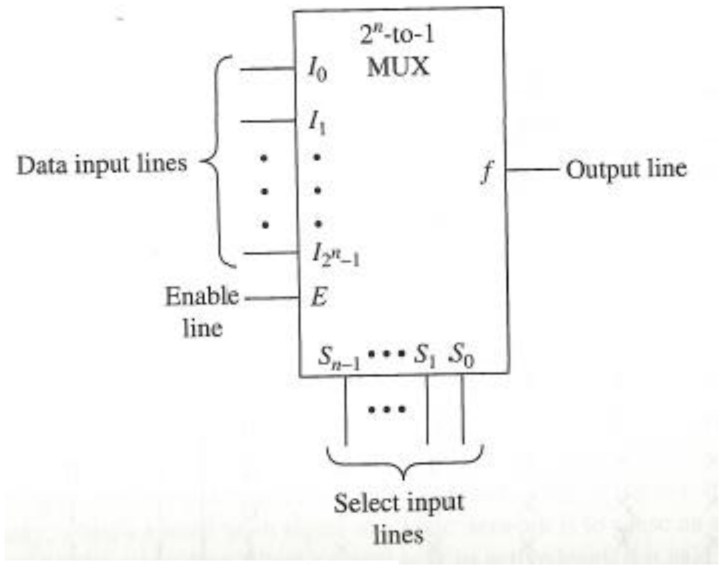


Figure 5.20 Realization of the Boolean expressions $f_1(x_2,x_1,x_0) = \Sigma m(0,1,3,4,5,6) = \overline{\Sigma m(2,7)}$ and $f_2(x_2,x_1,x_0) = \Sigma m(1,2,3,4,6) = \overline{\Sigma m(0,5,7)}$ with a 3-to-8-line decoder anc two nor-gates.

Logic Design with Multiplexers

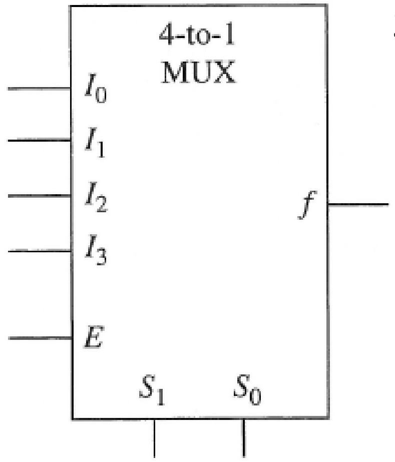


Logic Diagram



E	S_1	S_0	I_0	I_1	I_2	I_3	f
0	×	×	×	×	×	×	0
1	0	0	0	×	×	×	0
1	0	0	1	×	×	×	1
1	0	1	×	0	×	×	0
1	0	1	×	1	×	×	1
1	1	0	×	×	0	×	0
1	1	0	×	×	1	×	1
1	1	1	×	×	×	0	0
1	1	1	×	×	×	1	1

Truth Table



Symbol

Logic Design with Multiplexers

x	y	z	f
0	0	0	f_0
0	0	1	f_1
0	1	0	f_2
0	1	1	f_3
1	0	0	f_4
1	0	1	f_5
1	1	0	f_6
1	1	1	f_7

The Boolean expression corresponding to this truth table can be written as:

$$\begin{aligned} f(x, y, z) \\ &= f_0 \cdot \overline{x}\,\overline{y}\,\overline{z} + f_1 \cdot \overline{x}\,\overline{y}\,z + f_2 \cdot \overline{x}y\overline{z} + f_3 \cdot \overline{x}yz + f_4 \cdot x\overline{y}\,\overline{z} \\ &+ f_5 \cdot x\overline{y}z + f_6 xy\overline{z} + f_7 \cdot xyz. \end{aligned}$$

Logic Design with Multiplexers

- The Boolean expression corresponding to this truth table can be written as:

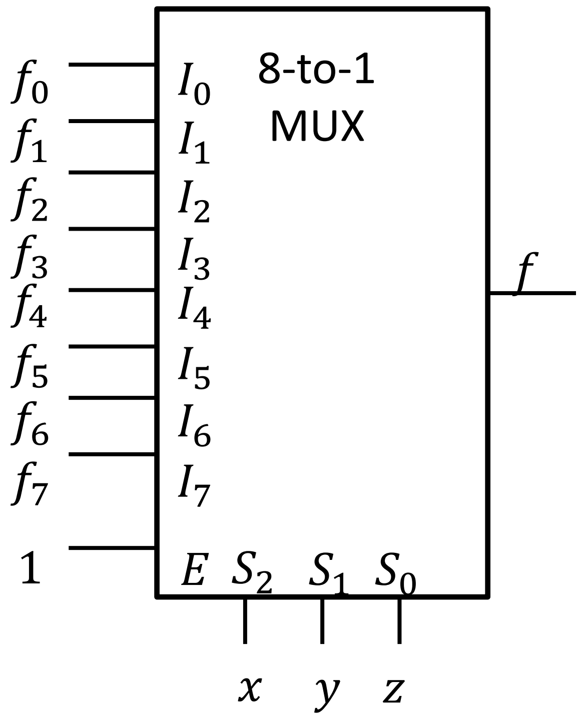
$$f(x,y,z) = f_0 \cdot \overline{x}\,\overline{y}\,\overline{z} + f_1 \cdot \overline{x}\,\overline{y}\,z + f_2 \cdot \overline{x}y\overline{z} + f_3 \cdot \overline{x}yz + f_4 \cdot x\overline{y}\,\overline{z} + f_5 \cdot x\overline{y}z + f_6 xy\overline{z} + f_7 \cdot xyz.$$

- The Boolean expression for an 8-to-1-line multiplexer is:

$$f = \big(I_0\overline{S}_2\overline{S}_1\overline{S}_0 + I_1\overline{S}_2\overline{S}_1S_0 + I_2\overline{S}_2S_1\overline{S}_0 + I_3\overline{S}_2S_1S_0 + I_4S_2\overline{S}_1\overline{S}_0 + I_5S_2\overline{S}_1S_0 + I_6S_2S_1\overline{S}_0 + I_7S_2S_1S_0\big).$$

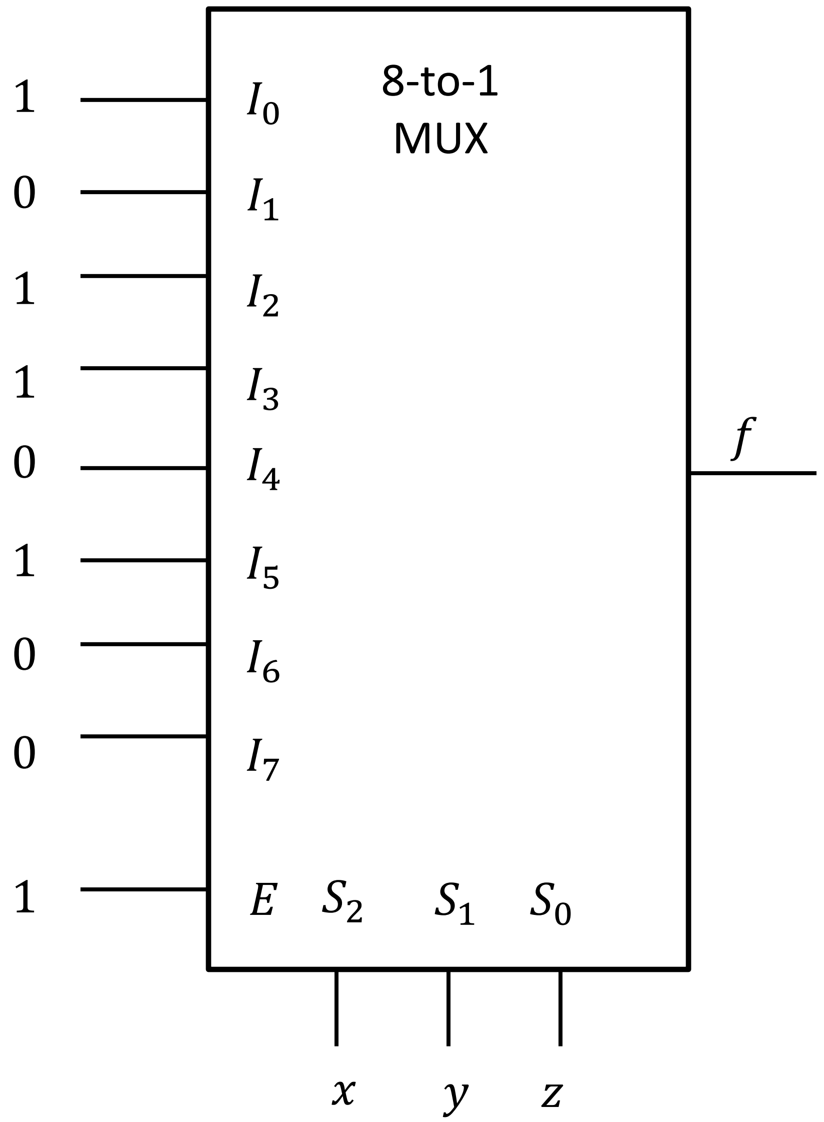
Logic Design with Multiplexers

- If E is logic-1 then the latter is transformed into the former by replacing I_i with f_i, S_2 with x, S_1 with y, and S_0 with z.
- Placing x, y, z on the select lines S_2, S_1, S_0, respectively and placing the functional values f_i on data input lines I_i.



Example:

x	y	z	f
0	0	0	1
0	0	1	0
0	1	0	1
0	1	1	1
1	0	0	0
1	0	1	1
1	1	0	0
1	1	1	0



Logic Design with Multiplexers

- If at least one input variable of a Boolean function is available in both its complemented and uncomplemented form, any n-variable function is realizable with a 2^{n-1}-to-1-line multiplexer.
- For the case of a 3-variable function, only a 4-to-1 multiplexer is needed.
- $f(x,y,z) = f_0 \cdot \overline{x}\,\overline{y}\,\overline{z} + f_1 \cdot \overline{x}\,\overline{y}\,z + f_2 \cdot \overline{x}y\overline{z} + f_3 \cdot \overline{x}yz + f_4 \cdot x\overline{y}\,\overline{z} + f_5 \cdot x\overline{y}z + f_6 xy\overline{z} + f_7 \cdot xyz$

$$= (f_0 \cdot \overline{z} + f_1 \cdot z)\overline{x}\,\overline{y} + (f_2 \cdot \overline{z} + f_3 \cdot z)\overline{x}y + (f_4 \cdot \overline{z} + f_5 \cdot z)x\overline{y} + (f_6 \cdot \overline{z} + f_7 \cdot z)xy$$

- When E = 1, 4-to-1 Multiplexer has the form

$$I_0\overline{S}_1\overline{S}_0 + I_1\overline{S}_1S_0 + I_2S_1\overline{S}_0 + I_3S_1S_2$$

Logic Design with Multiplexers

$$
\begin{aligned}
f(x, y, z) & \\
&= (f_0 \cdot \overline{z} + f_1 \cdot z)\overline{x}\,\overline{y} + (f_2 \cdot \overline{z} + f_3 \cdot z)\overline{x}y \\
&+ (f_4 \cdot \overline{z} + f_5 \cdot z)x\overline{y} + (f_6 \cdot \overline{z} + f_7 \cdot z)xy
\end{aligned}
$$

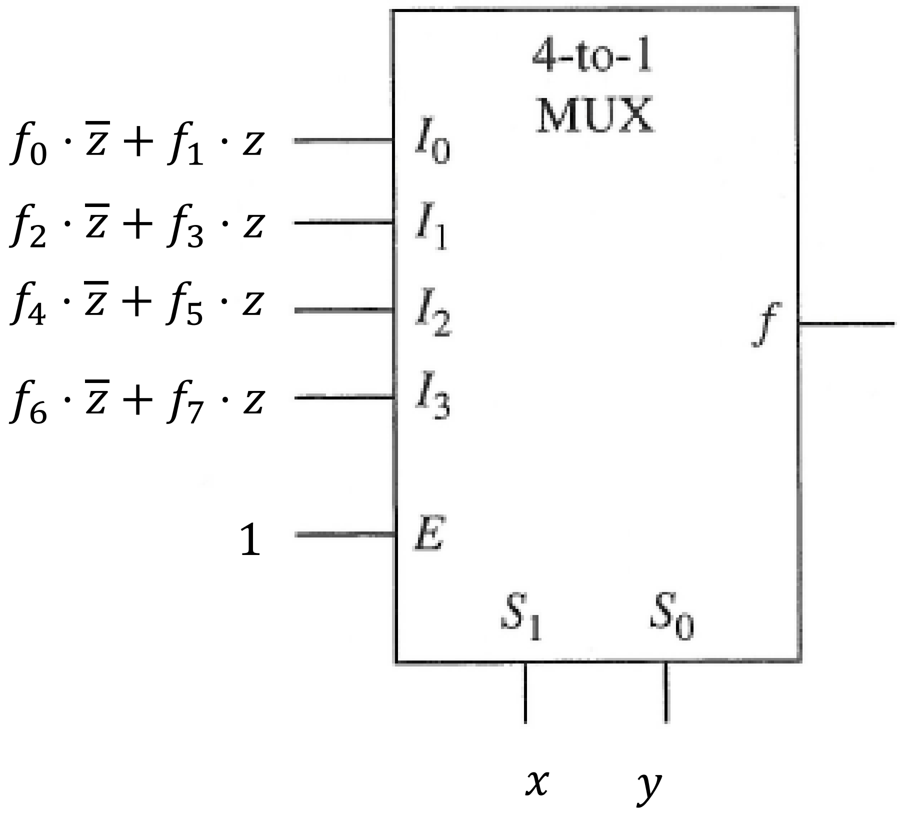
4-to-1 Multiplexer has the form

$$f = I_0\overline{S}_1\overline{S}_0 + I_1\overline{S}_1S_0 + I_2S_1\overline{S}_0 + I_3S_1S_2$$

- Realization of $f(x, y, z)$ is obtained by placing the x and y variables on the S_1, S_0 select lines, the single variable functions $f_i \cdot \overline{z} + f_j \cdot z$ on the data input lines and let E = 1.
- Note: $f_i \cdot \overline{z} + f_j \cdot z$ reduce to 0,1,z or $\overline{z}$.

Example

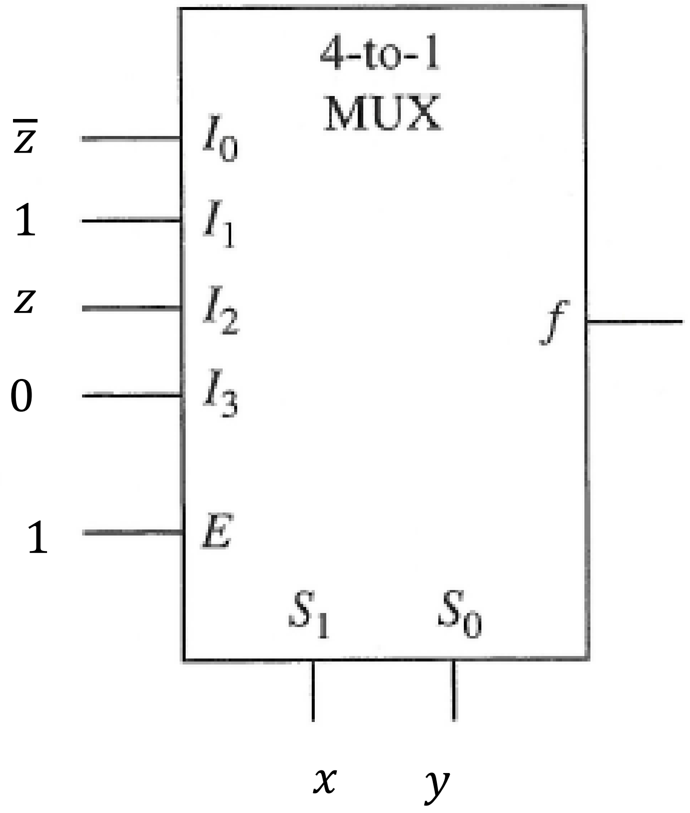
x	y	z	f
0	0	0	1
0	0	1	0
0	1	0	1
0	1	1	1
1	0	0	0
1	0	1	1
1	1	0	0
1	1	1	0



$f_0 \cdot \overline{z} + f_1 \cdot z$
$f_2 \cdot \overline{z} + f_3 \cdot z$
$f_4 \cdot \overline{z} + f_5 \cdot z$
$f_6 \cdot \overline{z} + f_7 \cdot z$
1
4-to-1 MUX
I_0
I_1
I_2
I_3
E
f
S_1
S_0
x
y

Example

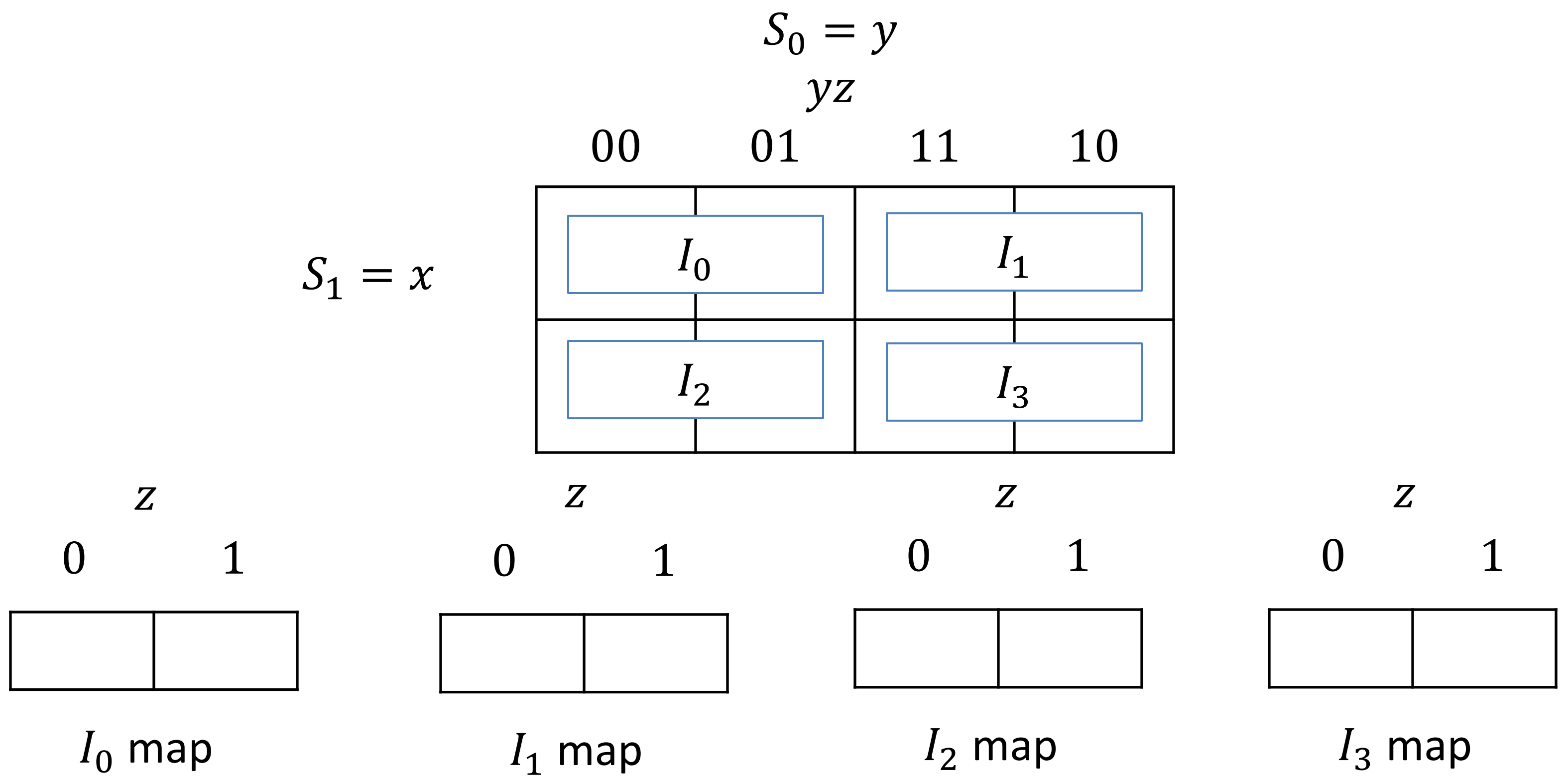
x	y	z	f
0	0	0	1
0	0	1	0
0	1	0	1
0	1	1	1
1	0	0	0
1	0	1	1
1	1	0	0
1	1	1	0



Logic Design with Multiplexers and K-maps

- Consider 3-variable Karnaugh map. Assume x is placed on the S_1 line and y is placed on the S_0 line.
- We get that the output is: $I_0\overline{x}\,\overline{y} + I_1\overline{x}y + I_2x\overline{y} + I_3xy$
- $I_0\overline{x}\,\overline{y}$ corresponds to those cells in which $x = 0, y = 0$
- $I_1\overline{x}y$ corresponds to those cells in which $x = 0, y = 1$
- $I_2x\overline{y}$ corresponds to those cells in which $x = 1, y = 0$
- I_3xy corresponds to those cells in which $x = 1, y = 1$

K-map representation



Example

x	y	z	f
0	0	0	1
0	0	1	0
0	1	0	1
0	1	1	1
1	0	0	0
1	0	1	1
1	1	0	0
1	1	1	0

$S_0 = y$

$S_1 = x$

yz	00	01	11	10
	1	0	1	1
	0	1	0	0

z: 0	1
1	0

I_0 map

z: 0	1
1	1

I_1 map

z: 0	1
0	1

I_2 map

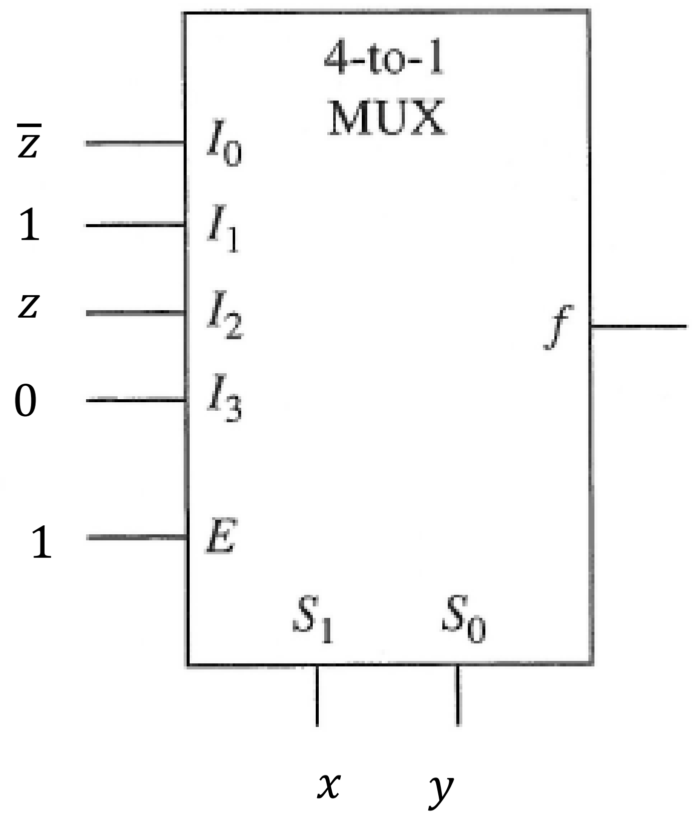
z: 0	1
0	0

I_3 map

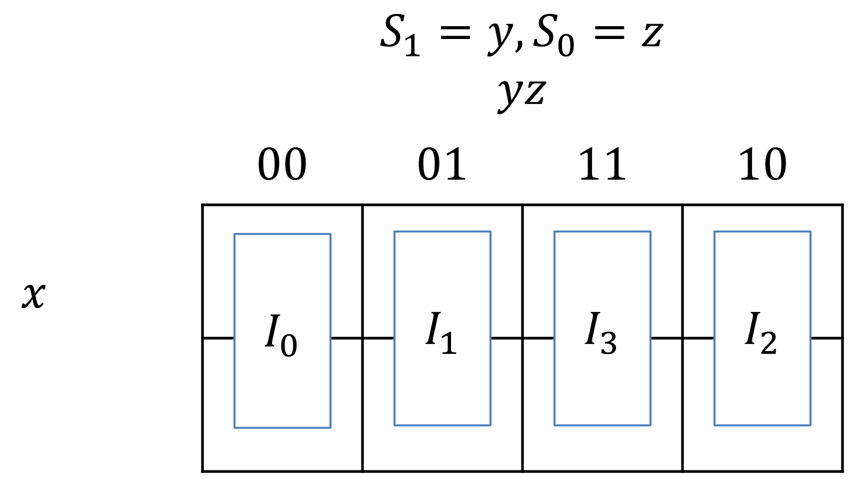
$I_0 = \overline{z}$ $\quad I_1 = 1$ $\quad I_2 = z$ $\quad I_3 = 0$

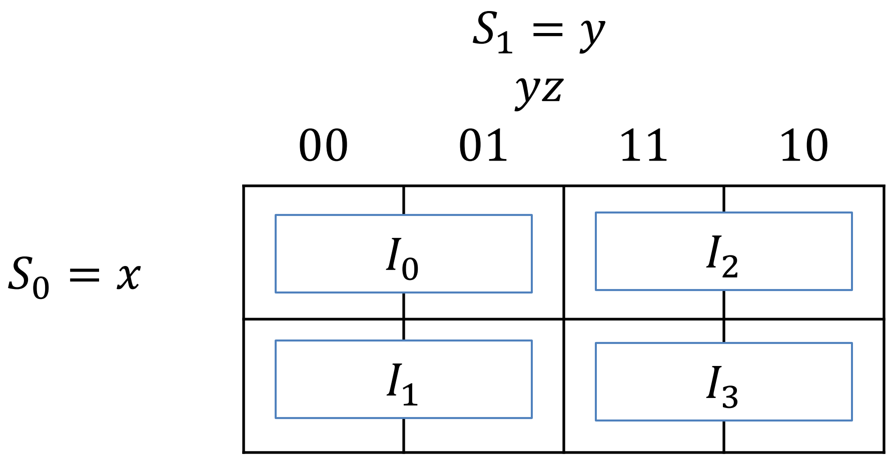
Realization

x	y	z	f
0	0	0	1
0	0	1	0
0	1	0	1
0	1	1	1
1	0	0	0
1	0	1	1
1	1	0	0
1	1	1	0



Alternative Structures





Note that order of variables on input lines matters!

8-to-1-line multiplexers and 4-variable Boolean functions

- Can do the same thing, three variables are placed on select lines, inputs to the data lines are single-variable functions.
- Example:

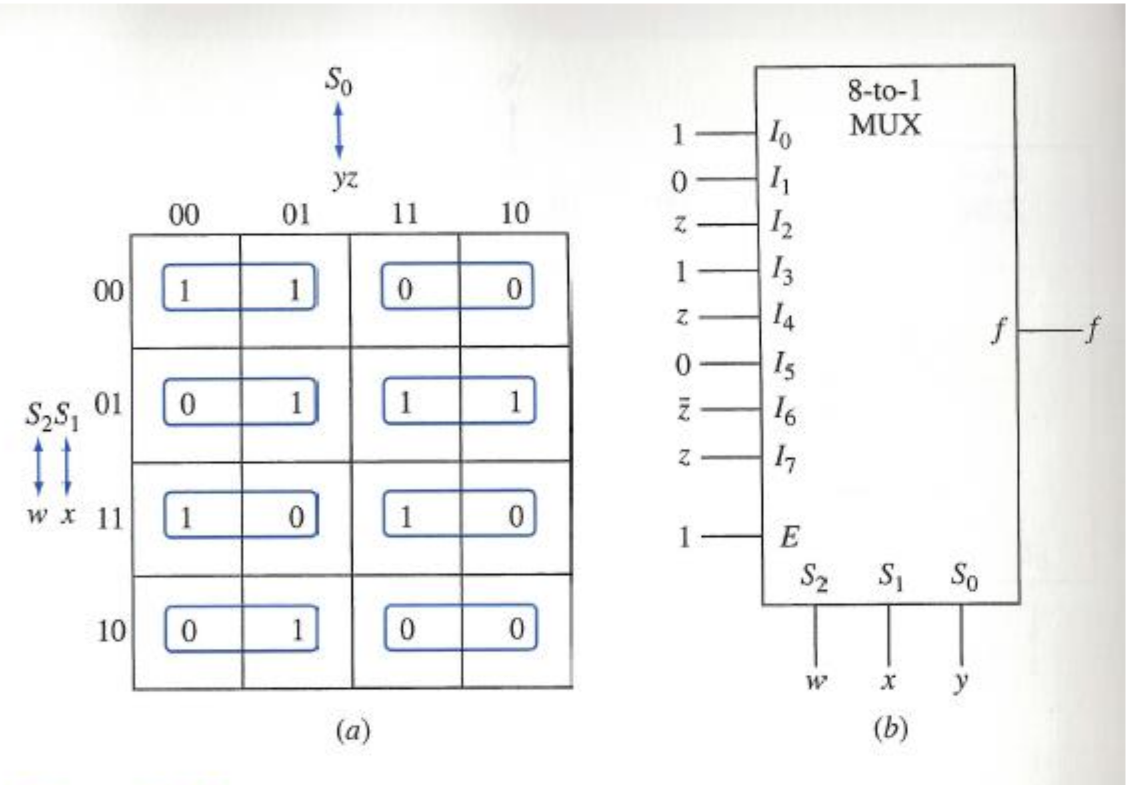


Figure 5.45 Realization of $f(w,x,y,z) = \Sigma m(0,1,5,6,7,9,12,15)$. (a) Karnaugh map. (b) Multiplexer realization.

Can we do better?

- By allowing realizations of m-variable functions as inputs to the data input lines, 2^n-to-1-line multiplexers can be used in the realization of $(n + m)$-variable functions.
- E.g.: input variables w and x are applied to the S_1, S_0 select inputs. Functions of the y and z variables appear at the data input lines.

K-map Structure

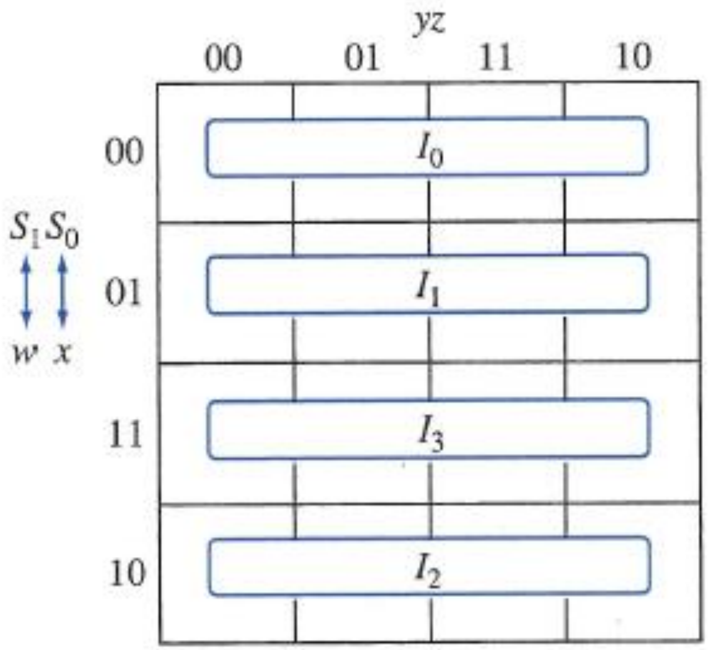
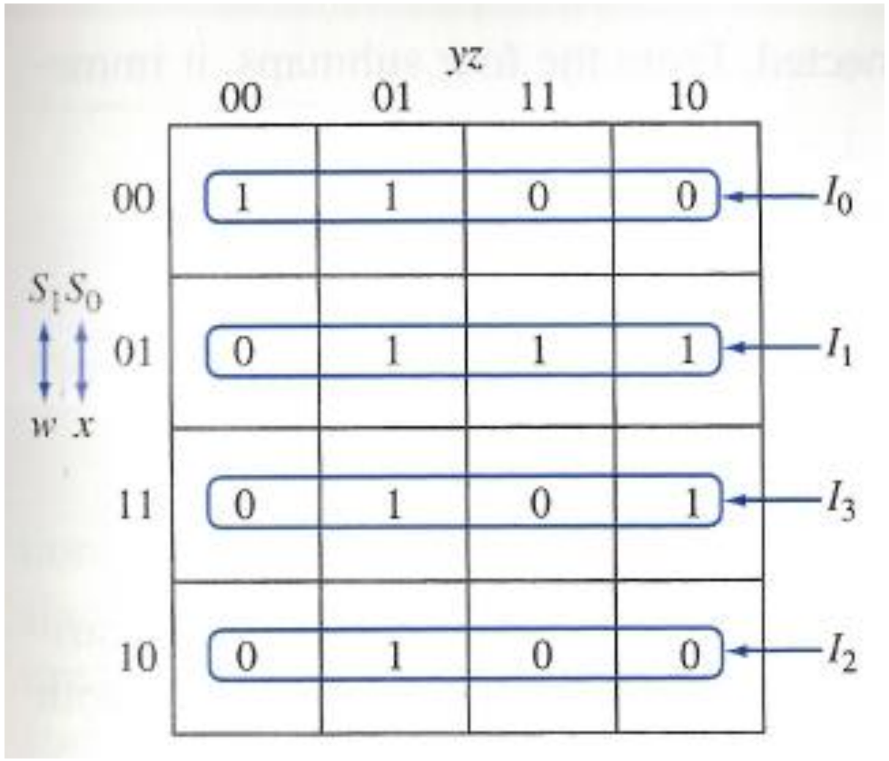
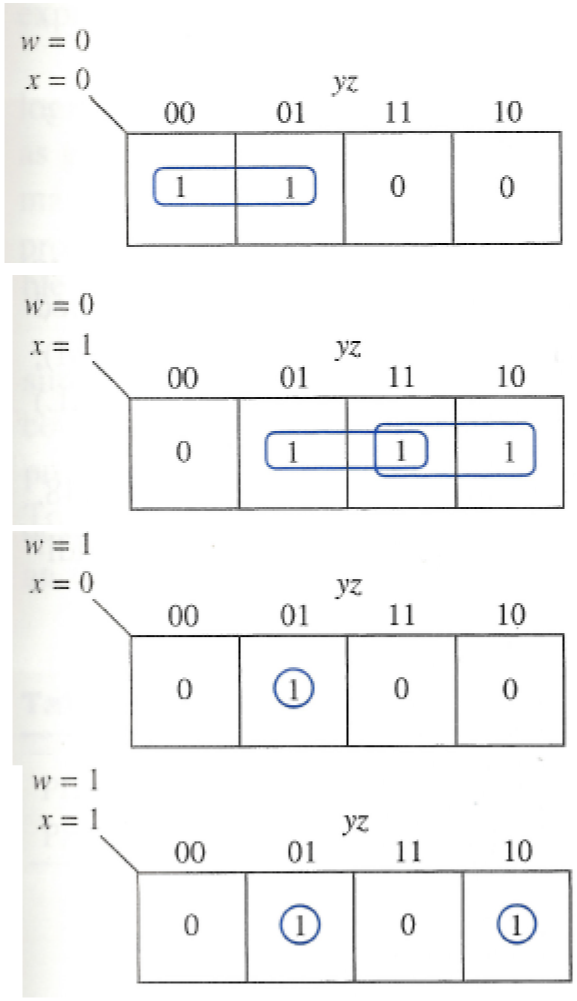


Figure 5.46 Using a four-variable Karnaugh map to obtain a Boolean function realization with a 4-to-1-line multiplexer.

Example:

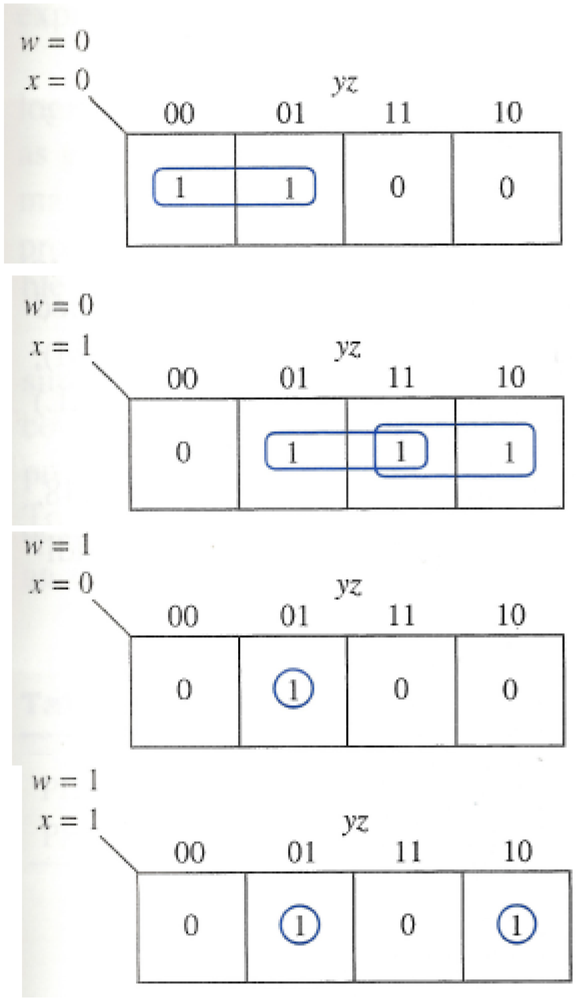
$$f(x,y,z) = \sum m(0,1,5,6,7,9,13,14)$$





$w = 0$, $x = 0$: 1 1 0 0
$w = 0$, $x = 1$: 0 1 1 1
$w = 1$, $x = 0$: 0 1 0 0
$w = 1$, $x = 1$: 0 1 0 1
yz
00
01
11
10

Example



$w = 0$, $x = 0$

yz	00	01	11	10
	1	1	0	0

$w = 0$, $x = 1$

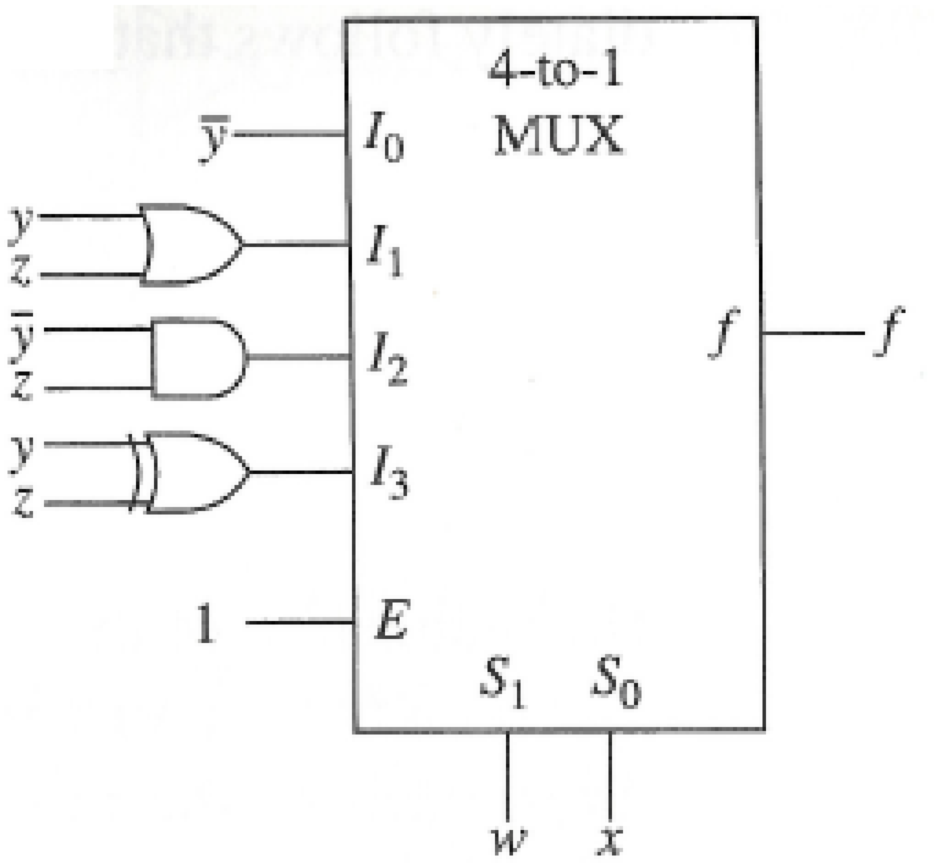
yz	00	01	11	10
	0	1	1	1

$w = 1$, $x = 0$

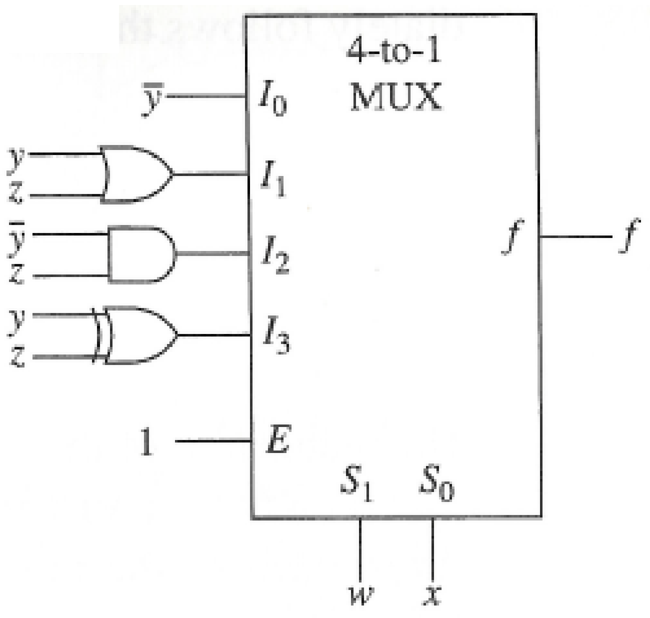
yz	00	01	11	10
	0	1	0	0

$w = 1$, $x = 1$

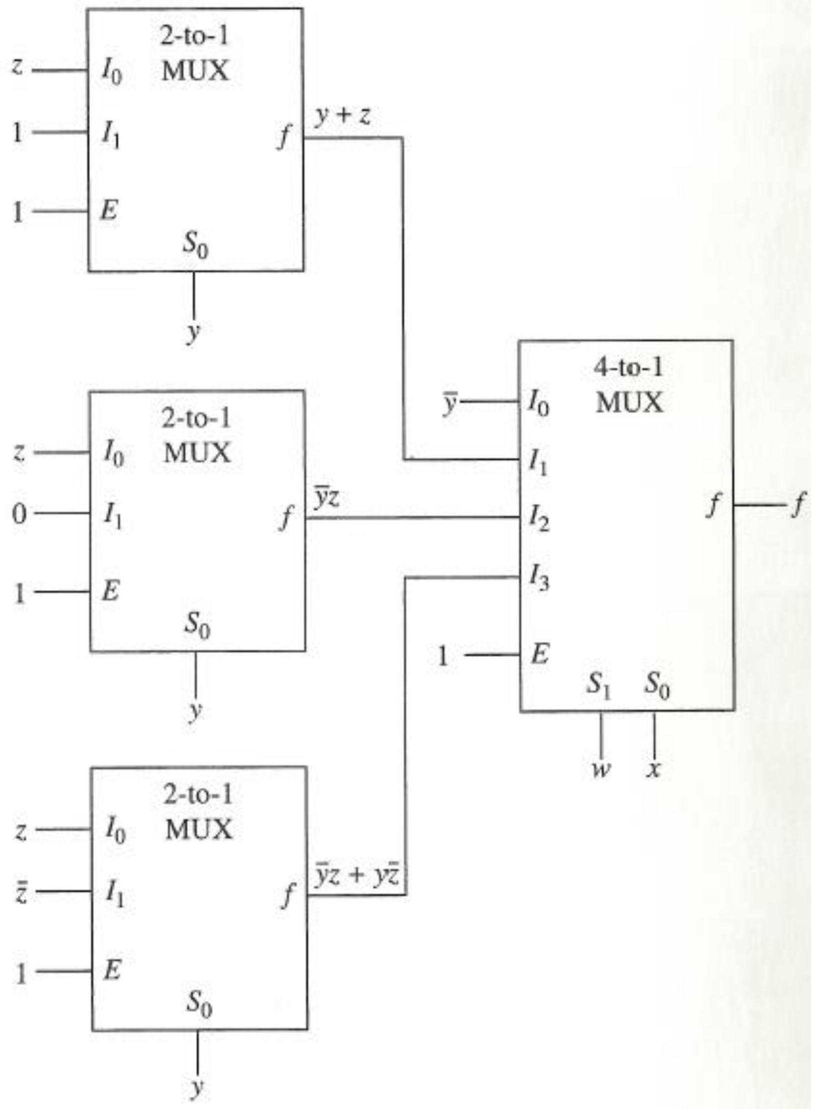
yz	00	01	11	10
	0	1	0	1



Example



Multiplexer Tree


2-to-1
MUX
z
I_0
1
I_1
1
E
S_0
f
$y + z$
y
2-to-1
MUX
z
I_0
0
I_1
1
E
S_0
f
$\bar{y}z$
y
2-to-1
MUX
z
I_0
$\bar{z}$
I_1
1
E
S_0
f
$\bar{y}z + y\bar{z}$
y
4-to-1
MUX
$\bar{y}$
I_0
I_1
I_2
I_3
1
E
S_1
S_0
f
f
w
x