DESIGN OF GATE NETWORKS

- Design of two-level networks:
  AND-OR and OR-AND networks

- Minimal two-level networks
  Karnaugh maps
  Minimization procedure and tools
  Limitations of two-level networks

- Design of two-level NAND-NAND and NOR-NOR networks

- Programmable logic: PLAs and PALs.

Design of two-level networks

Implementation:

Level 1 (optional) NOT gates

Level 2 AND gates

Level 3 OR gates

Literals
  (uncomplemented and complemented variables)
  NOT gates (if needed)

Products: AND gates

Sum: OR gate

Multioutput networks: one OR gate is used for each output

Product of sums networks - similar

Modulo-64 incrementer

Input: $0 \leq x \leq 63$
Output: $0 \leq z \leq 63$
Function: $z = (x + 1) \bmod 64$

x	010101
z	010110

x	001111
z	010000

- Radix-2 representation

$$z_i = \begin{cases} 1 \textbf{ if } (x_i = 1 \textbf{ and } \textit{there exists } j < i \textit{ such that } x_j = 0) \\ \quad \textbf{or } (x_i = 0 \textbf{ and } x_j = 1 \textit{ for all } j < i) \\ 0 \textbf{ otherwise} \end{cases}$$

$$\begin{aligned}
z_5 &= x_5(x_4' + x_3' + x_2' + x_1' + x_0') + x_5' x_4 x_3 x_2 x_1 x_0 \\
&= x_5 x_4' + x_5 x_3' + x_5 x_2' + x_5 x_1' + x_5 x_0' + x_5' x_4 x_3 x_2 x_1 x_0 \\
z_4 &= x_4 x_3' + x_4 x_2' + x_4 x_1' + x_4 x_0' + x_4' x_3 x_2 x_1 x_0 \\
z_3 &= x_3 x_2' + x_3 x_1' + x_3 x_0' + x_3' x_2 x_1 x_0 \\
z_2 &= x_2 x_1' + x_2 x_0' + x_2' x_1 x_0 \\
z_1 &= x_1 x_0' + x_1' x_0 \\
z_0 &= x_0'
\end{aligned}$$

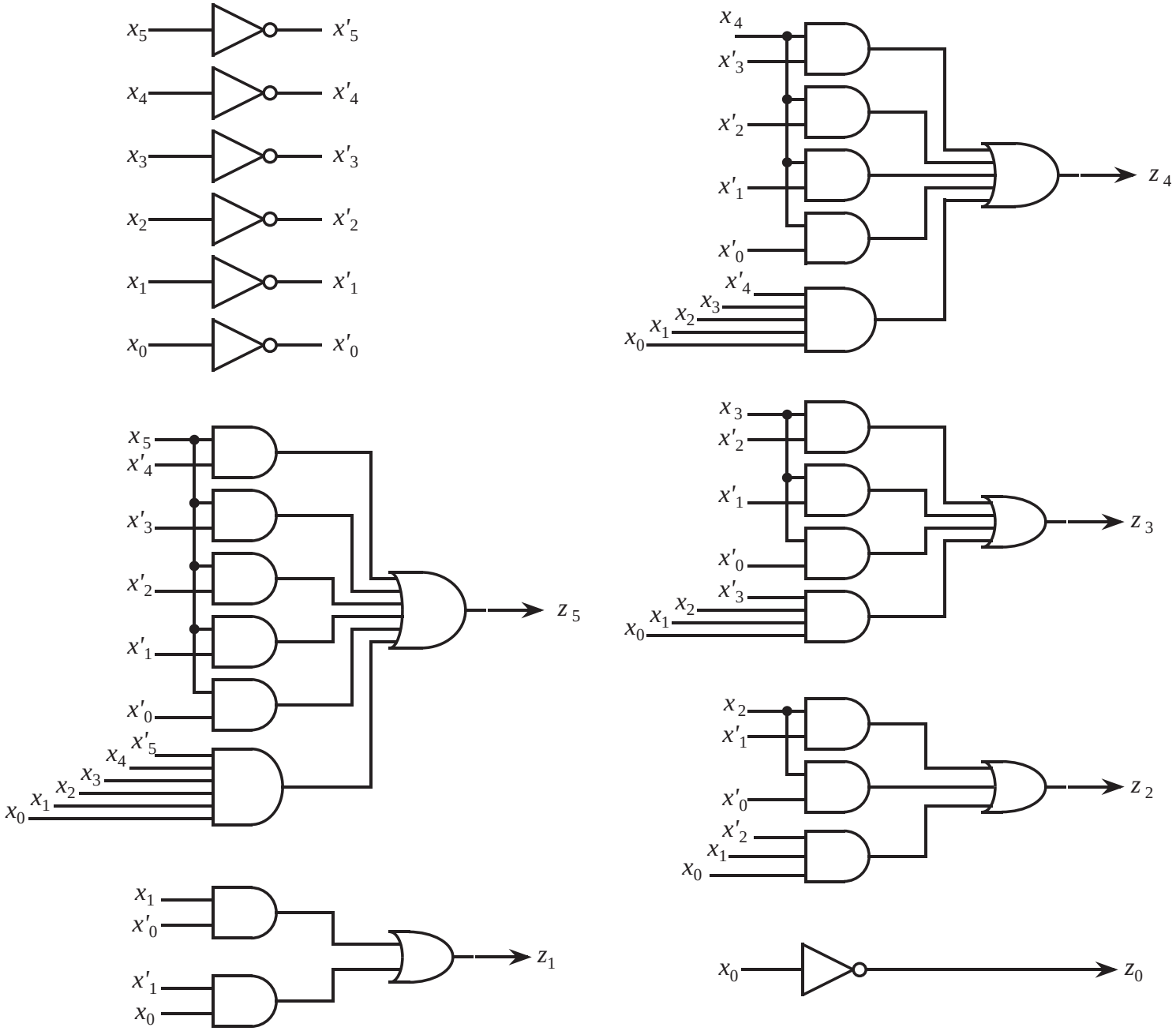


Figure 5.1: NOT-AND-OR modulo-64 incrementer network.

Uncomplemented and complemented inputs available

- Two types of two-level networks:

AND-OR **network** $\Leftrightarrow$ Sum of products (NAND-NAND network)

OR-AND **network** $\Leftrightarrow$ Product of sums (NOR-NOR network)

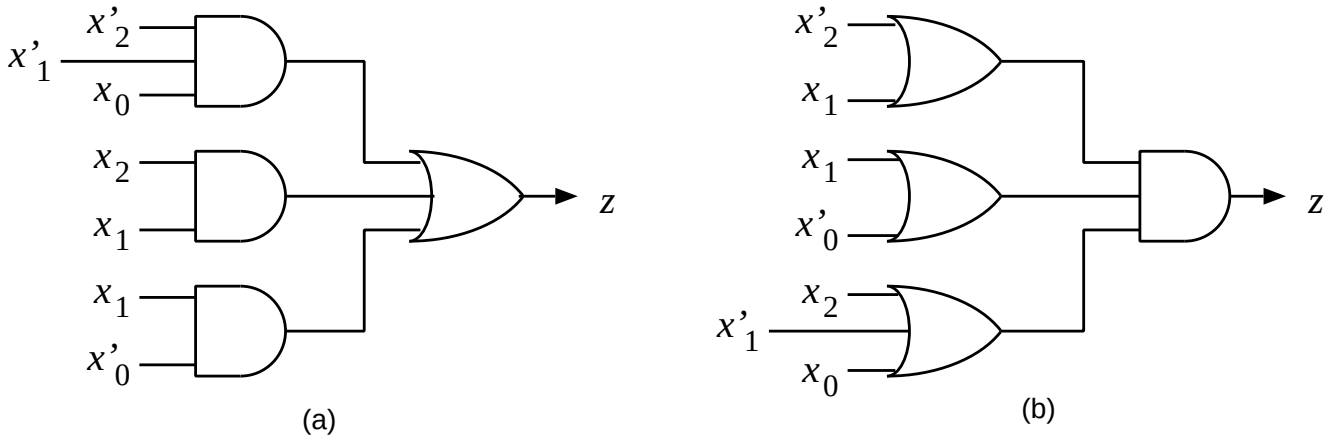


Figure 5.2: AND-OR and OR-AND networks.

$$E(x_2, x_1, x_0) = x_2' x_1' x_0 + x_2 x_1 + x_1 x_0'$$

$$E(x_2, x_1, x_0) = (x_2' + x_1)(x_1 + x_0')(x_2 + x_1' + x_0)$$

Minimal two-level networks

1. Inputs: uncomplemented and complemented

2. Fanin unlimited

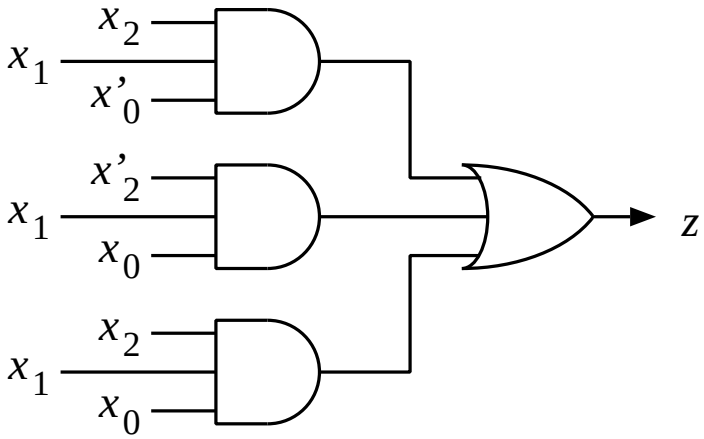
3. Single-output networks

4. Minimal network:

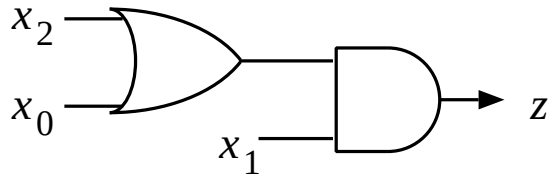
   minimum number of gates with minimum number of inputs

   (minimal expression: min. number of terms with min. number of literals)

Networks with different cost



Network A



Network B

Figure 5.3: Networks with different cost to implement $f(x_2, x_1, x_0) = one\text{-}set(3,6,7)$.

Minimal expressions

- Equivalent but different cost

$$E_1(x_2, x_1, x_0) = x_2' x_1 x_0' + x_1' x_0 + x_2 x_0$$
$$E_2(x_2, x_1, x_0) = x_2 x_1 x_0 + x_2' x_1 x_0' + x_2' x_1' x_0 + x_2 x_1' x_0$$

- Both minimal SP and PS must be obtained and compared

- Basis:

$$ab + ab' = a \quad \text{(for sum of products)}$$
$$(a + b)(a + b') = a \quad \text{(for product of sums)}$$

Graphical representation of switching functions: Karnaugh maps

- 2-dimensional array of cells
- n variables $\longrightarrow$ 2^n cells
- cell i $\longleftrightarrow$ assignment i

  adjacency condition

  any set of 2^r adjacent rows (columns):

  assignments differ in r variables

- representing switching functions
- representing switching expressions
- graphical aid in simplifying expressions

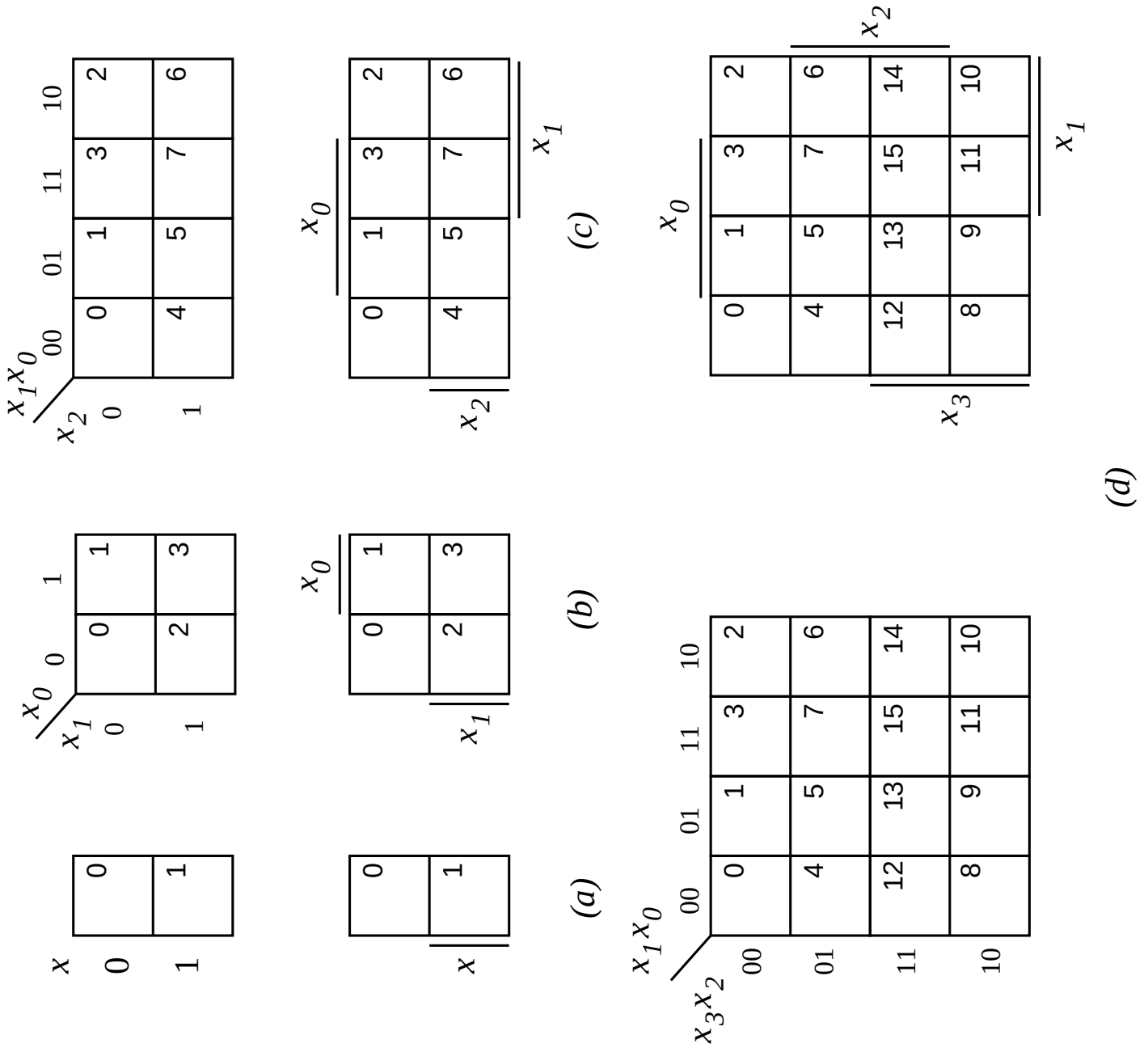


Figure 5.4: K-Maps

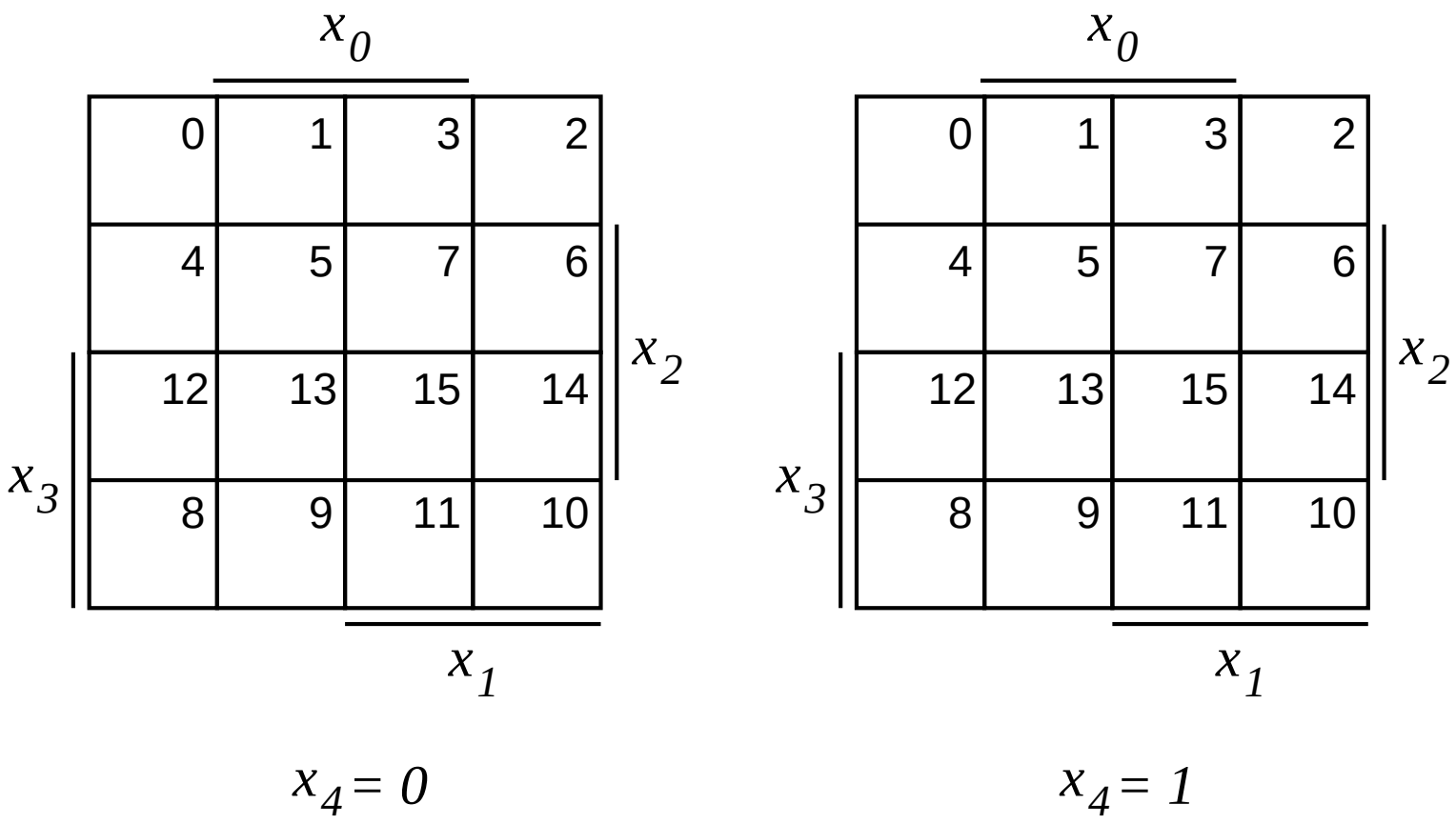
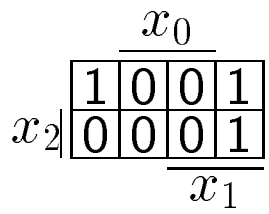


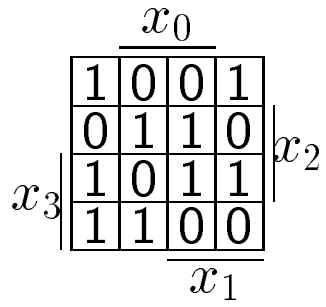
Figure 5.5: K-map for five variables

Representation of switching functions

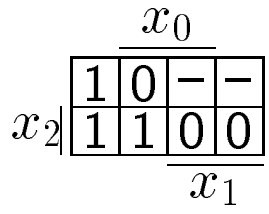
$f(x_2, x_1, x_0) = \mathit{one\text{-}set}(0,2,6)$



$f(x_3, x_2, x_1, x_0) = \mathit{zero\text{-}set}(1,3,4,6,10,11,13)$



$f(x_2, x_1, x_0) = [\mathit{one\text{-}set}(0,4,5), \mathit{dc\text{-}set}(2,3)]$



Rectangles of 1-cells and sum of products

1. Minterm m_j corresponds to 1-cell with label j.
2. Product term of $n-1$ literals $\longleftrightarrow$ rectangle of two adjacent 1-cells.

$$\begin{aligned} x_3x_1'x_0 &= x_3x_1'x_0(x_2 + x_2') \\ &= x_3x_2x_1'x_0 + x_3x_2'x_1'x_0 \\ &= m_{13} + m_9 \end{aligned}$$

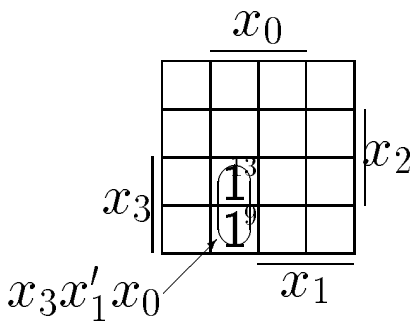


Figure 5.6

Rectangles of 1-cells and sum of products (cont.)

3. Product term of $n-2$ literals $\longleftrightarrow$ rectangle of four adjacent 1-cells.

$$\begin{aligned} x_3x_0 &= x_3x_0(x_1 + x_1')(x_2 + x_2') \\ &= x_3x_2'x_1'x_0 + x_3x_2'x_1x_0 + x_3x_2x_1'x_0 + x_3x_2x_1x_0 \\ &= m_9 + m_{11} + m_{13} + m_{15} \end{aligned}$$

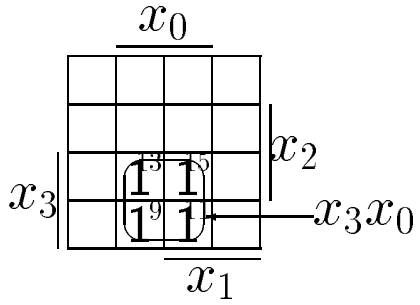


Figure 5.6

4. Product term of $n-s$ literals $\longleftrightarrow$ rectangle of 2^s adjacent 1-cells.

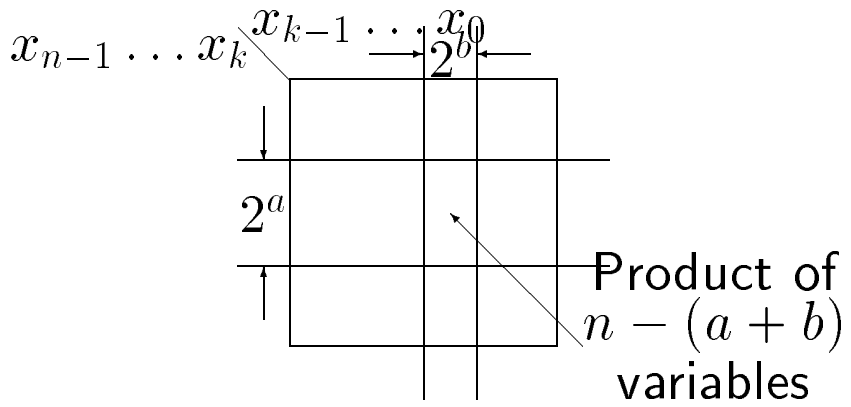


Figure 5.7: Representation of product of $n - (a + b)$ variables.

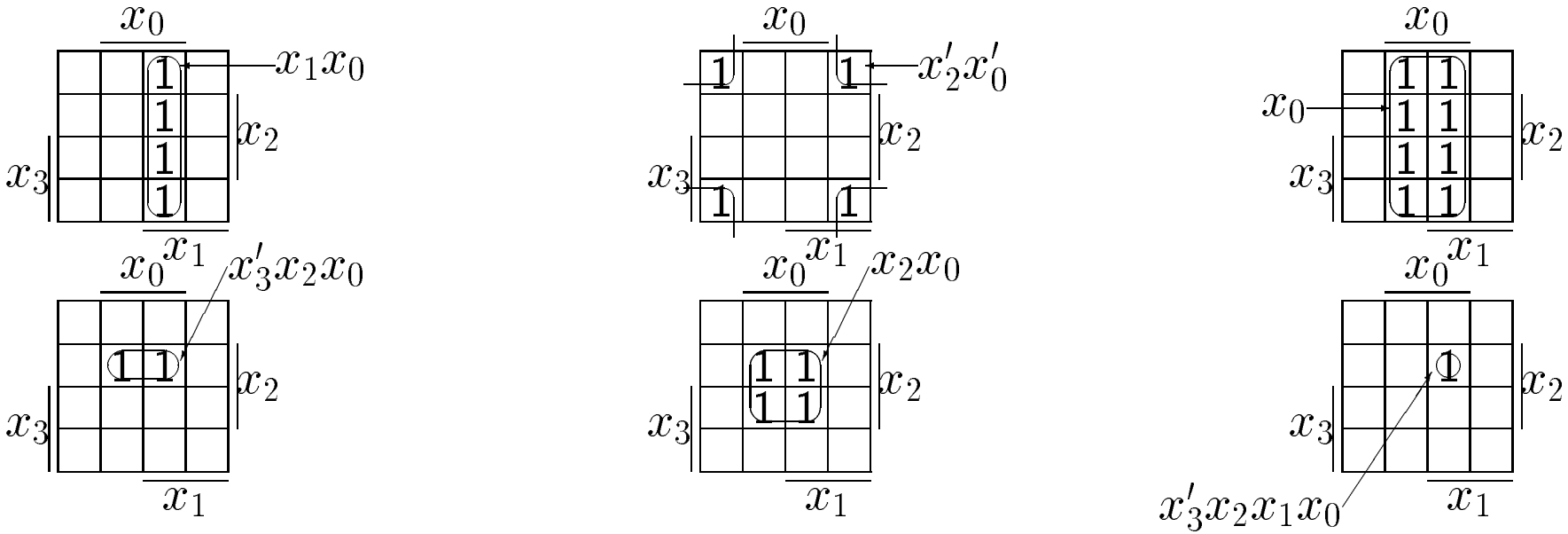
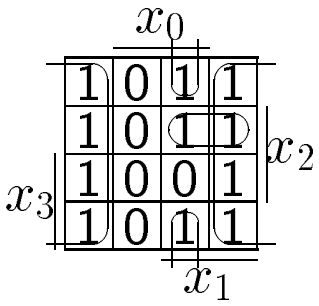


Figure 5.8: Product terms and rectangles of 1-cells.

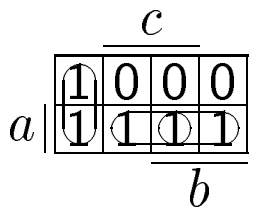
Sum of products

represented in a K-map by the union of rectangles

$$E(x_3, x_2, x_1, x_0) = x_3' x_2 x_1 + x_2' x_1 x_0 + x_0'$$



$$E(a, b, c) = ab + ac + b'c'$$



Rectangles of 0-cells and product of sums

0-cell 13 corresponds to the maxterm

$$M_{13} = x_3' + x_2' + x_1 + x_0'$$

Rectangle of $2^a \times 2^b$ 0-cells $\longleftrightarrow$ sum term of $n - (a + b)$ literals

Minimization of sums of products

Implicant: product term for which **f=1**

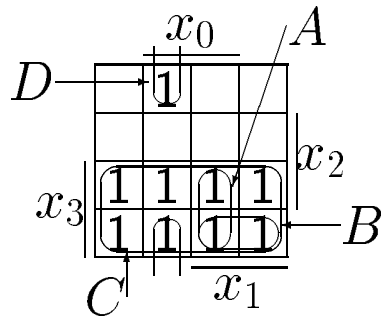


Figure 5.9: Implicant representation.

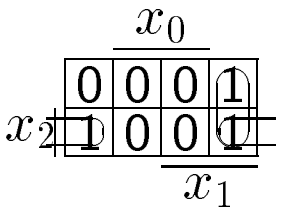
Implicants: $x_3'x_2'x_1'x_0$, all product terms with x_3.

Prime Implicant: Implicant not covered by another implicant.

Prime implicants: $x_2'x_1'x_0$, x_3

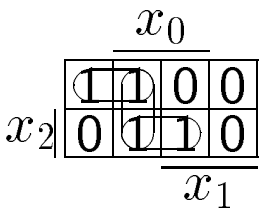
Find all PIs

a) $f(x_2, x_1, x_0) = \mathit{one\text{-}set}(2,4,6)$



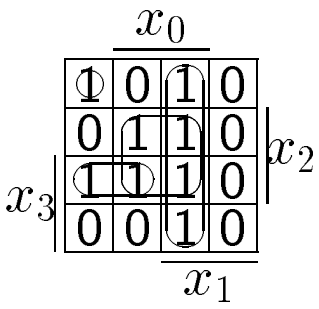
PIs: x_2x_0' and x_1x_0'

b) $f(x_2, x_1, x_0) = \mathit{one\text{-}set}(0,1,5,7)$



PIs: $x_2'x_1'$, x_2x_0, and $x_1'x_0$

c) $f(x_3, x_2, x_1, x_0) = \mathit{one\text{-}set}$(0,3,5,7,11,12,13,15)



PIs: x_2x_0, x_1x_0, $x_3x_2x_1'$, and $x_3'x_2'x_1'x_0'$

Minimal sum of products consists of prime implicants

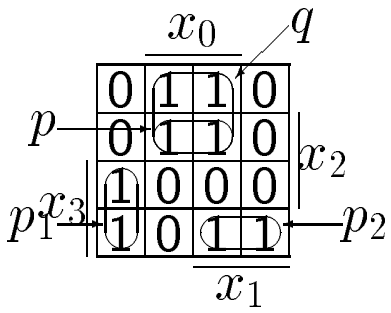
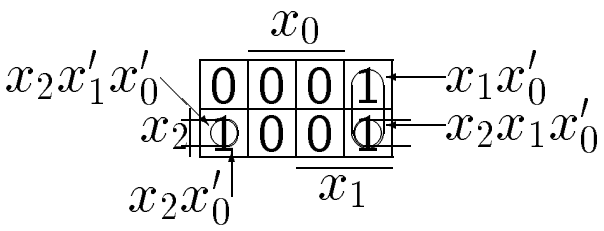


Figure 5.10: Minimal sum of products and prime implicants.

Example 5.9

$$E(x_2, x_1, x_0) = x_2x_1'x_0' + x_2x_1x_0' + x_1x_0'$$



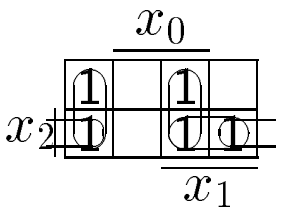
not PIs: $x_2x_1'x_0'$ and $x_2x_1x_0'$

PI: x_2x_0', x_1x_0'

reduced SP: $E(x_2, x_1, x_0) = x_2x_0' + x_1x_0'$

Essential Prime Implicants (EPI)

$p_e(\underline{a}) = 1$ and $p(\underline{a}) = 0$ for any other PI p



EPIs: $x_1'x_0'$ and x_1x_0

non-essential: x_2x_1, x_2x_0'.

- All EPIs are included in a minimal SP

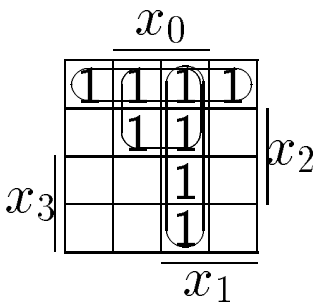
Procedure for finding min SP

1. Determine all PIs
2. Obtain the EPIs
3. If not all 1-cells covered, choose a cover from the remaining PIs

Example 5.10

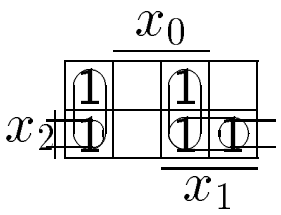
Find a minimal SP:

a) $E(x_3, x_2, x_1, x_0) = x_3'x_2' + x_3'x_2x_0 + x_1x_0$



- PIs: $x_3'x_2'$, $x_3'x_0$, and x_1x_0
- all EPIs
- unique min SP: $x_3'x_2' + x_3'x_0 + x_1x_0$

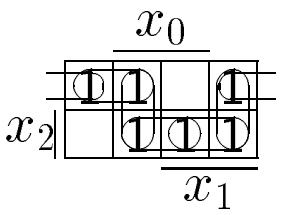
b) $E(x_2, x_1, x_0) = \Sigma\, m(0, 3, 4, 6, 7)$



- PIs: $x_1'x_0'$, x_1x_0, x_2x_0', and x_2x_1
- EPIs: $x_1'x_0'$ and x_1x_0
- extra cover: x_2x_0' or x_2x_1
- Two min SPs:

$$x_1'x_0' + x_1x_0 + x_2x_0' \quad \text{and} \quad x_1'x_0' + x_1x_0 + x_2x_1$$

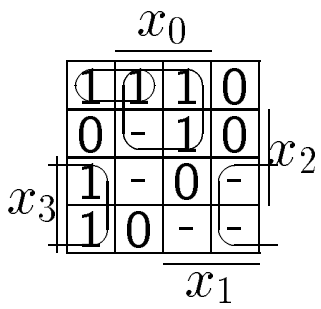
c) $E(x_2, x_1, x_0) = \Sigma\, m(0, 1, 2, 5, 6, 7)$



- PIs: $x_2'x_1'$, $x_2'x_0'$, x_2x_0, x_2x_1, $x_1'x_0$, and x_1x_0'
- No EPIs
- Two min SPs

$$x_2'x_1' + x_2x_0 + x_1x_0' \quad \text{and} \quad x_2'x_0' + x_1'x_0 + x_2x_1$$

Minimal SPs for incompletely specified functions



A minimal SP

$$E(x_3, x_2, x_1, x_0) = x_3 x_0' + x_3' x_0 + x_3' x_2' x_1'$$

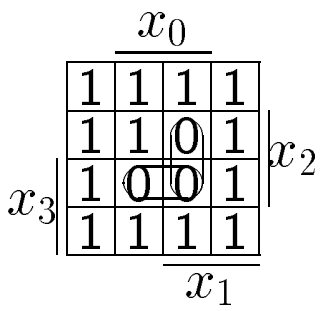
Minimization of products of sums

Implicate: sum term for which $f = 0$.

Prime Implicate: implicate not covered by another implicate

Essential prime implicate: at least one "cell" not included in other implicate

$f(x_3, x_2, x_1, x_0) = zero\text{-}set(7,13,15)$

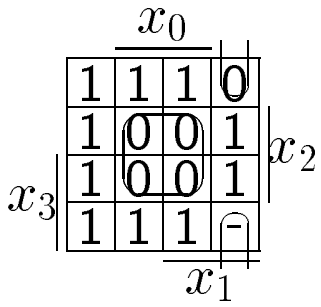


The prime implicates: $(x_3' + x_2' + x_0')$ and $(x_2' + x_1' + x_0')$

Both essential.

Procedure for finding min PS

1. Determine all prime implicates
2. Determine the essential prime implicates
3. From set of nonessential prime implicates, select cover of remaining 0-cells



- The prime implicates: $(x_0' + x_2')$ and $(x_0 + x_2 + x_1')$
- Both essential, the minimal PS is $(x_0' + x_2')(x_0 + x_2 + x_1')$

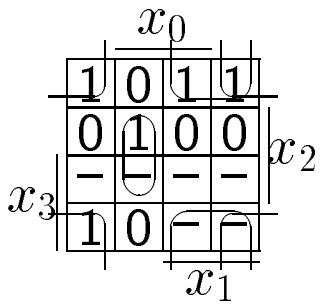
Minimal two-level gate network design: Example 5.14

Input: $x \in \{0, 1, 2, ..., 9\}$, coded in BCD as
$\underline{x} = (x_3, x_2, x_1, x_0),\ x_i \in \{0, 1\}$

Output: $z \in \{0, 1\}$

Function: $z = \begin{cases} 1 \ \textbf{if} \ \ x \in \{0, 2, 3, 5, 8\} \\ 0 \ \textbf{otherwise} \end{cases}$

The values {10,11,12,13,14,15} are "don't cares"



min SP: $z = x_2' x_1 + x_2' x_0' + x_2 x_1' x_0$

min PS: $z = (x_2' + x_1')(x_2' + x_0)(x_2 + x_1 + x_0')$

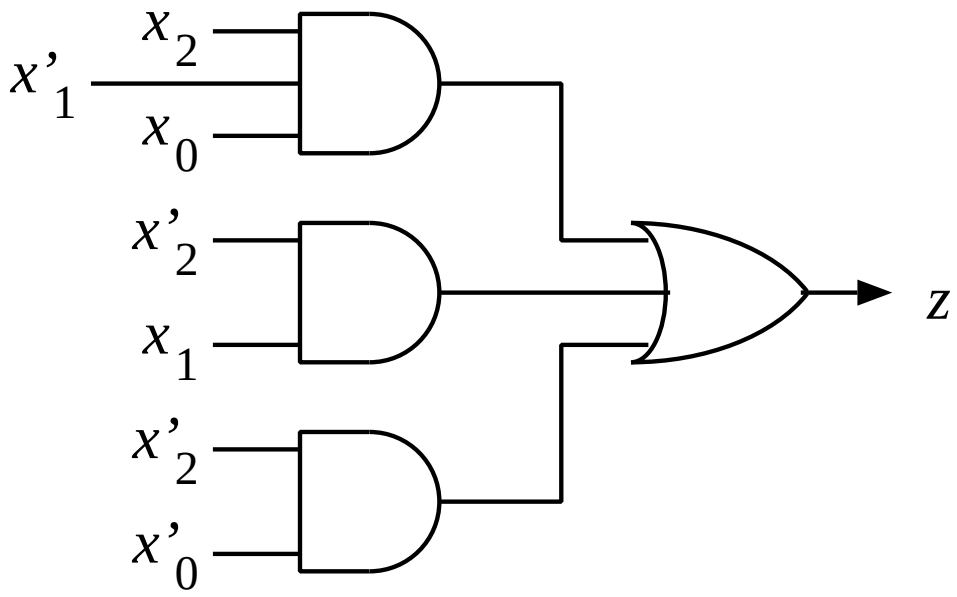


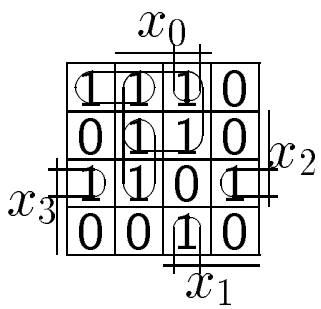
Figure 5.11: Minimal AND-OR network

Example 5.15

Input: $x \in \{0, 1, 2, ..., 15\}$
represented in binary code by $\underline{x} = (x_3, x_2, x_1, x_0)$
Output: $z \in \{0, 1\}$

Function: $z = \begin{cases} 1 \ \textbf{if} \ \ x \in \{0, 1, 3, 5, 7, 11, 12, 13, 14\} \\ 0 \ \textbf{otherwise} \end{cases}$

The K-map:



min SP: $z = x_3'x_0 + x_3'x_2'x_1' + x_2x_1'x_0 + x_3x_2x_0' + x_2'x_1x_0$
min PS: $z = (x_3' + x_2 + x_1)(x_3 + x_2' + x_0)(x_2 + x_1' + x_0)(x_3' + x_2' + x_1' + x_0')$
Cost(PS) < Cost(SP)

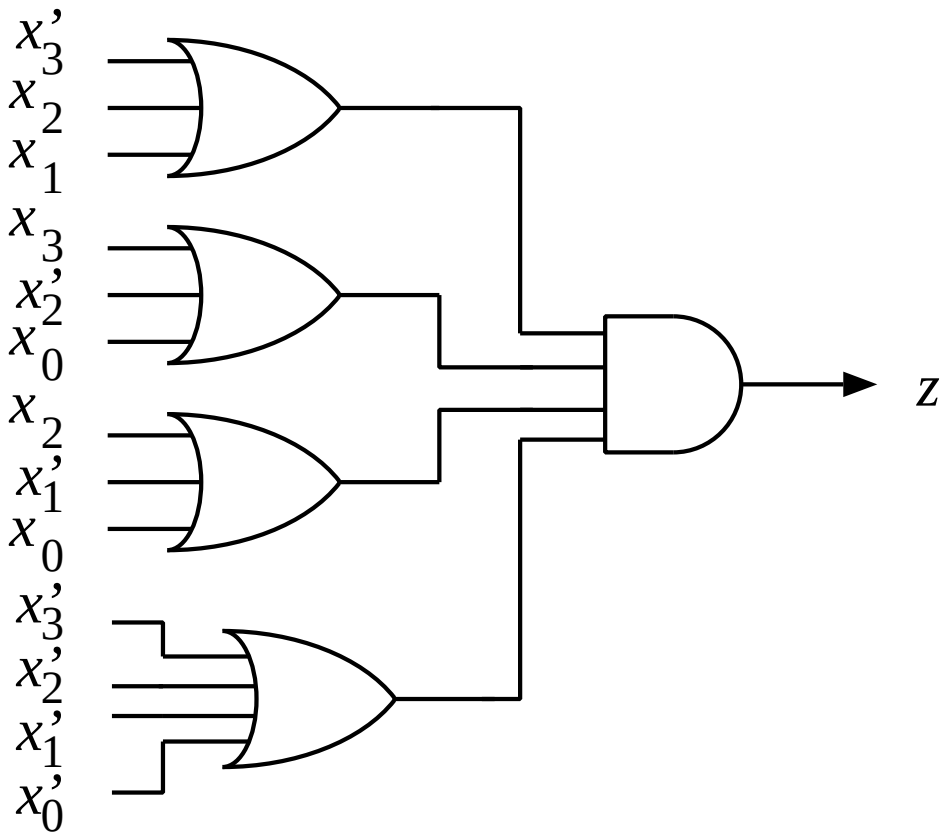


Figure 5.12: Minimal OR-AND network

Design of multiple-output two-level gate networks

- Separate network for each output: no sharing

Example 5.16

Inputs: $(x_2, x_1, x_0), \quad x_i \in \{0, 1\}$
Output: $z \in \{0, 1, 2, 3\}$
Function: $z = \Sigma_{i=0}^{2} x_i$

1. The switching functions in tabular form are

x_2	x_1	x_0	z_1	z_0
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

2. The corresponding K-maps are

z_1

x_2 \ x_0	0	0	1	0
0	0	1	1	1

x_1

z_0

x_2 \ x_0	0	1	0	1
1	0	1	0	1

x_1

3. minimal SPs:

$$z_1 = x_2x_1 + x_2x_0 + x_1x_0$$
$$z_0 = x_2'x_1'x_0 + x_2'x_1x_0' + x_2x_1'x_0' + x_2x_1x_0$$

4. minimal PSs:

$$z_1 = (x_2 + x_0)(x_2 + x_1)(x_1 + x_0)$$
$$z_0 = (x_2 + x_1 + x_0)(x_2 + x_1' + x_0')$$
$$(x_2' + x_1 + x_0')(x_2' + x_1' + x_0)$$

5. SP and PS expressions have the same cost

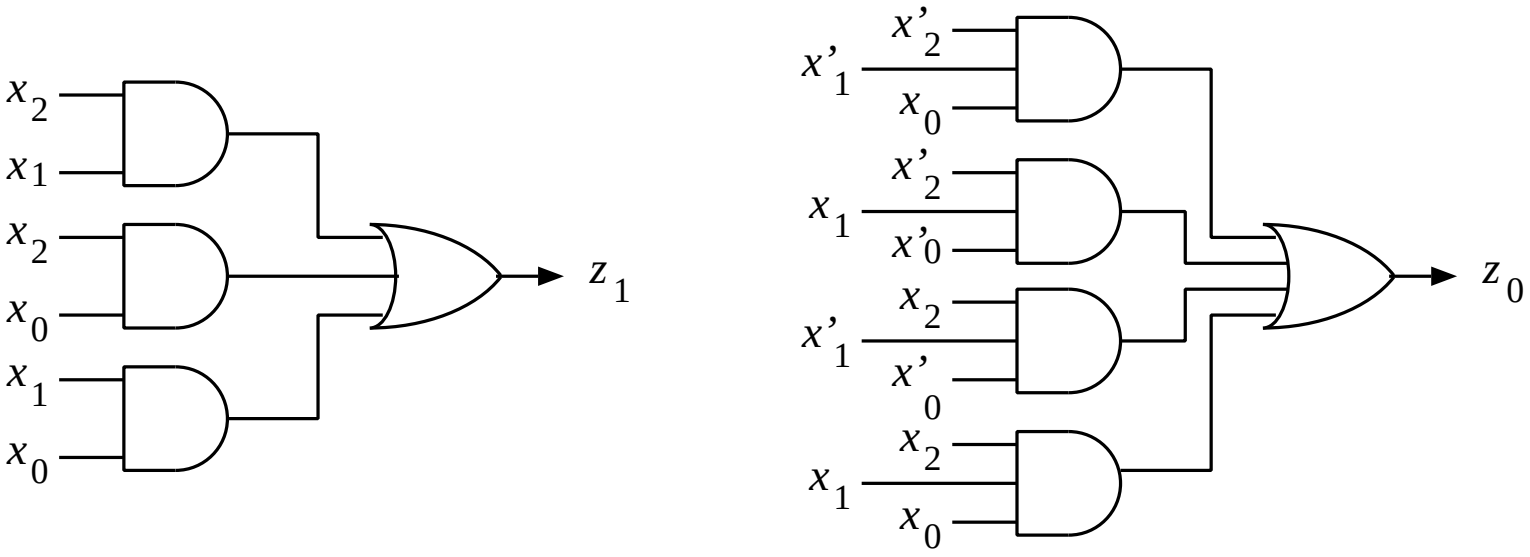


Figure 5.13: Minimal two-output AND-OR network

Two-level NAND-NAND and NOR-NOR networks

$$E = p_1 + p_2 + p_3 + \ldots + p_n$$

$p_1, p_2, \ldots$ are product terms

$$E = (p_1' \cdot p_2' \cdot p_3' \ldots p_n')'$$

or

$$E = \text{NAND}\big(\text{NAND}1, \text{NAND}2, \text{NAND}3, \ldots, \text{NAND}n\big)$$

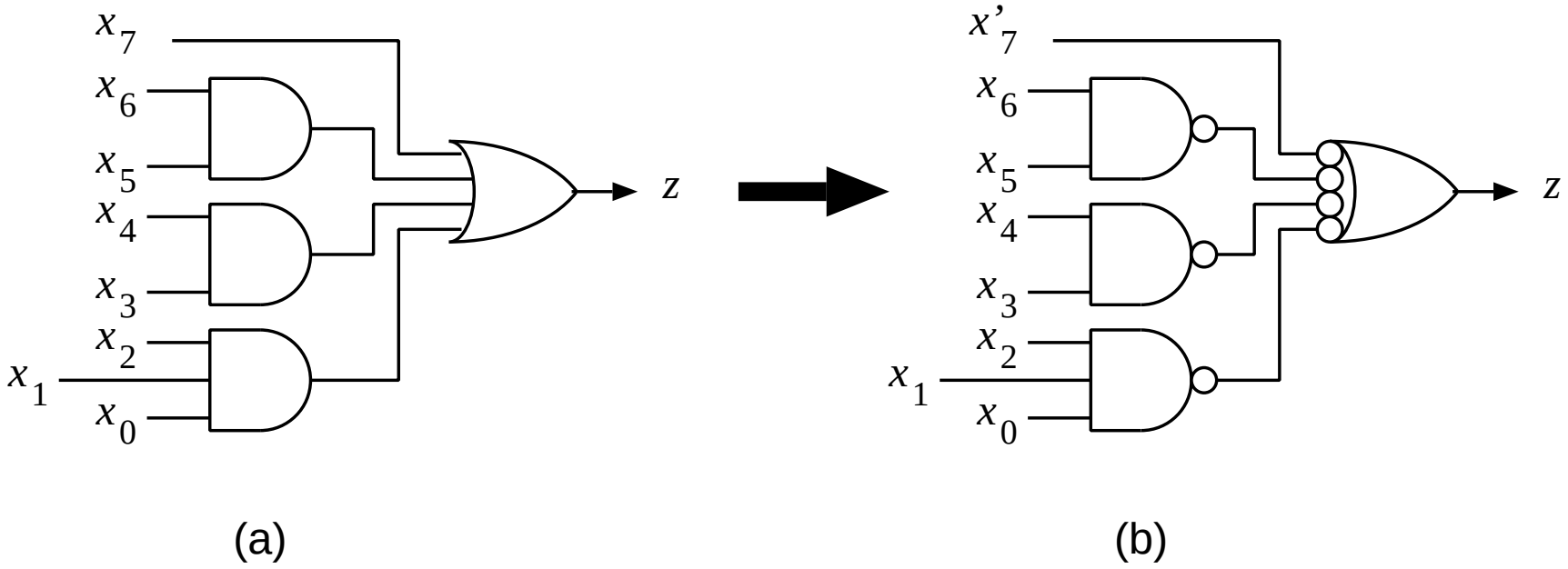


Figure 5.15: Transformation of AND-OR network into NAND network

Example: NOR network

$$z = x_5'(x_4 + x_3')(x_2 + x_1 + x_0)$$

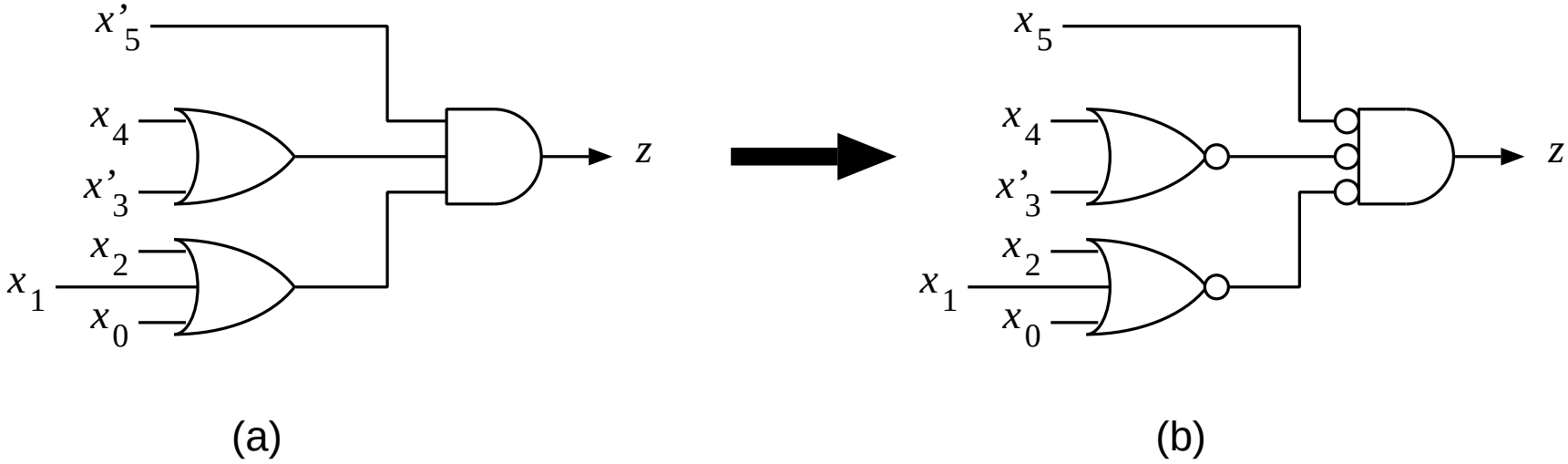


Figure 5.16: Equivalent OR-AND and NOR networks

Limitations of two-level networks

1. The requirement of uncomplemented and complemented inputs
   If not satisfied, an additional level of NOT gates needed

2. A two-level implementation of a function might require a large number of gates and irregular connections

3. Existing technologies have limitations in the fan-in of the gates

4. The procedure essentially limited to the single-output case

5. The cost criterion of minimizing the number of gates is not adequate for many MSI/LSI/VLSI designs

Programmable Modules: PLAs and PALs

- Standard (fixed) structure
- Customized (programmed) for a particular function
  - during the last stage of fabrication
  - when incorporated into a system
- Flexible use
- More expensive and slower than fixed-function modules
- Other types discussed in Chapter 12

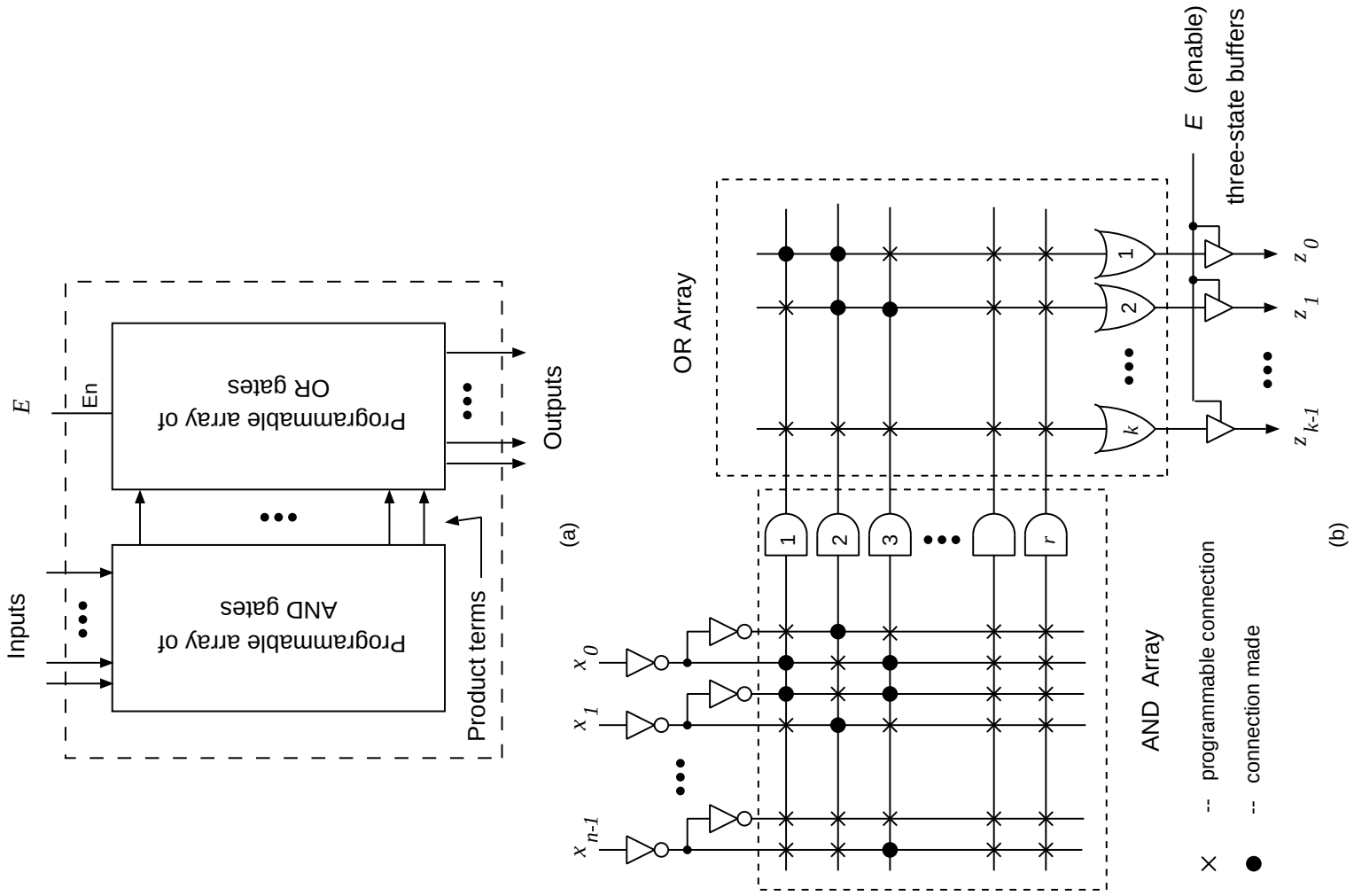


Figure 5.17: Programmable logic array (PLA): a) block diagram; b) logic diagram.

MOS PLA (OR-AND version)

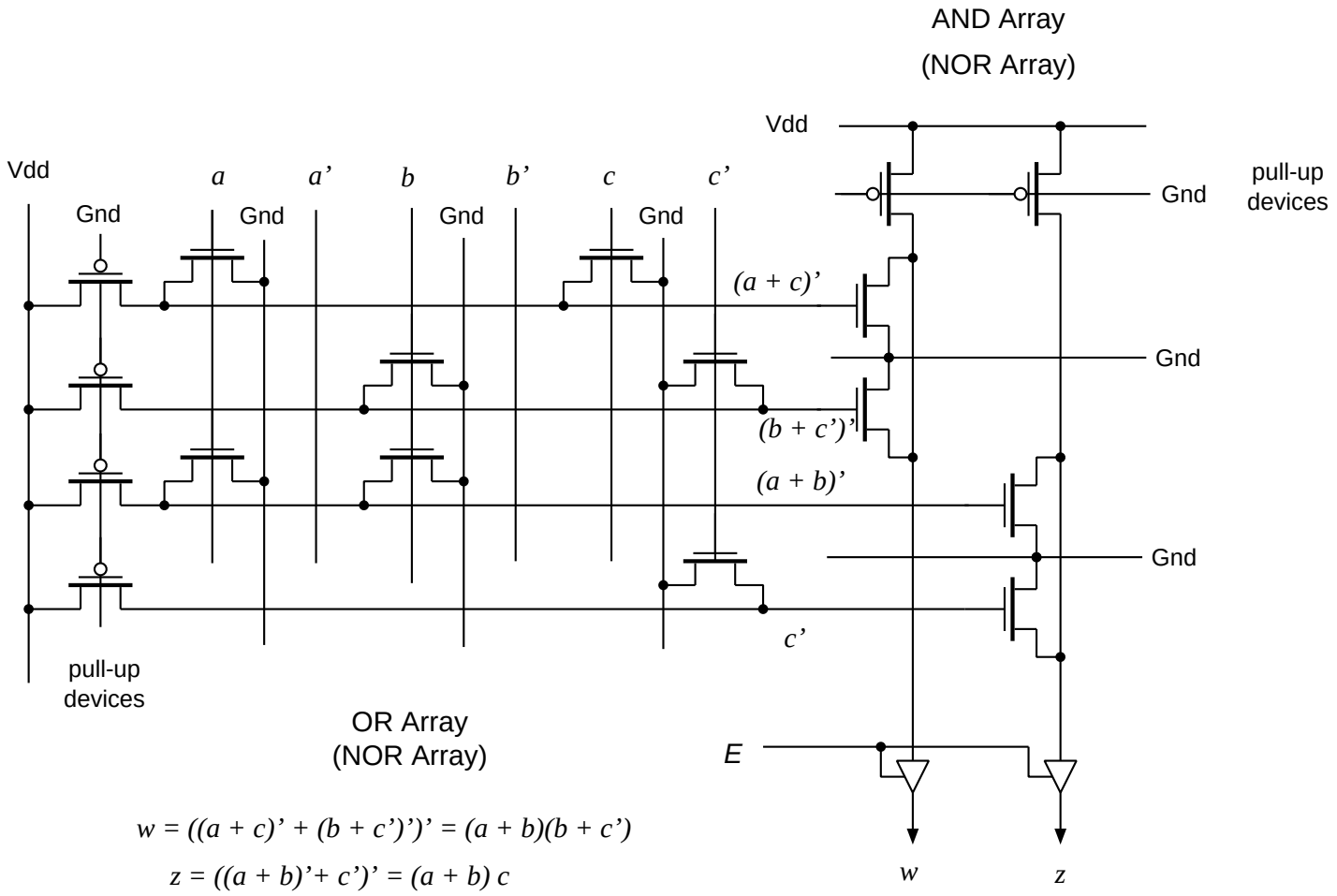


Figure 5.18: Example of PLA implementation at the circuit level: fragment of a MOS PLA.

Implementation of switching functions using PLAs

A BCD-to-Gray converter

Inputs: $\underline{d} = (d_3, d_2, d_1, d_0), \quad d_j \in \{0, 1\}$

Outputs: $\underline{g} = (g_3, g_2, g_1, g_0), \quad g_j \in \{0, 1\}$

Function:

i	$d_3 d_2 d_1 d_0$	$g_3 g_2 g_1 g_0$
0	0000	0000
1	0001	0001
2	0010	0011
3	0011	0010
4	0100	0110
5	0101	0111
6	0110	0101
7	0111	0100
8	1000	1100
9	1001	1101

Expressions:

$$g_3 = d_3$$
$$g_2 = d_3 + d_2$$
$$g_1 = d_2' d_1 + d_2 d_1'$$
$$g_0 = d_1 d_0' + d_1' d_0$$

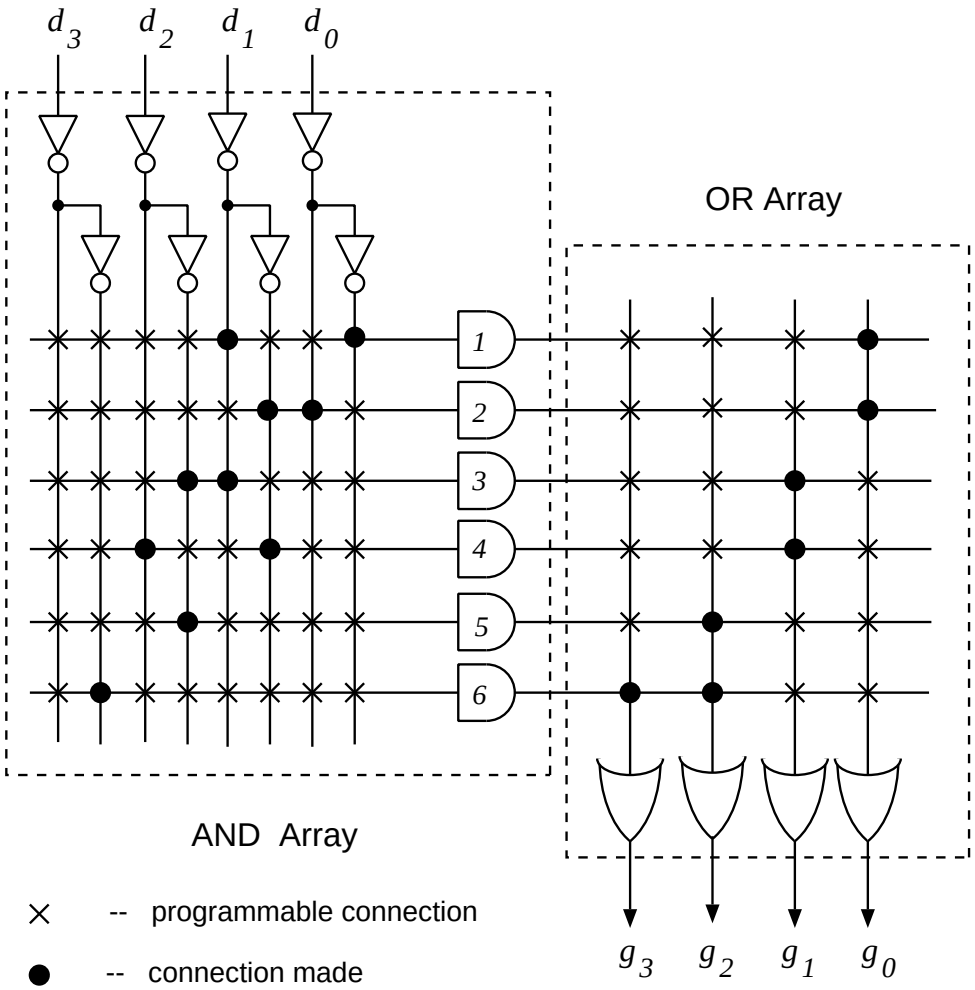


Figure 5.19: PLA implementation of BCD-Gray code converter.

PAL : a programmable module with fixed OR array

- Faster, more inputs and product terms compared to PLAs

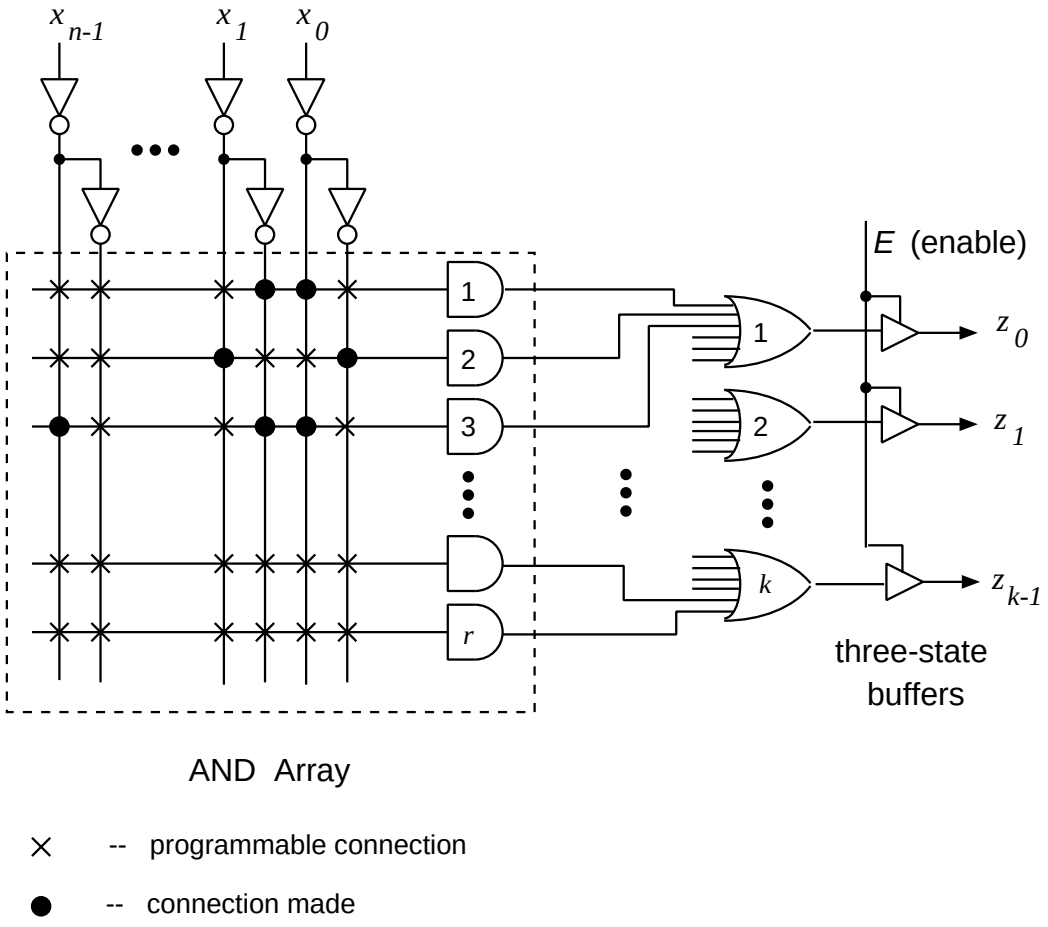


Figure 5.20: Logic diagram of a PAL

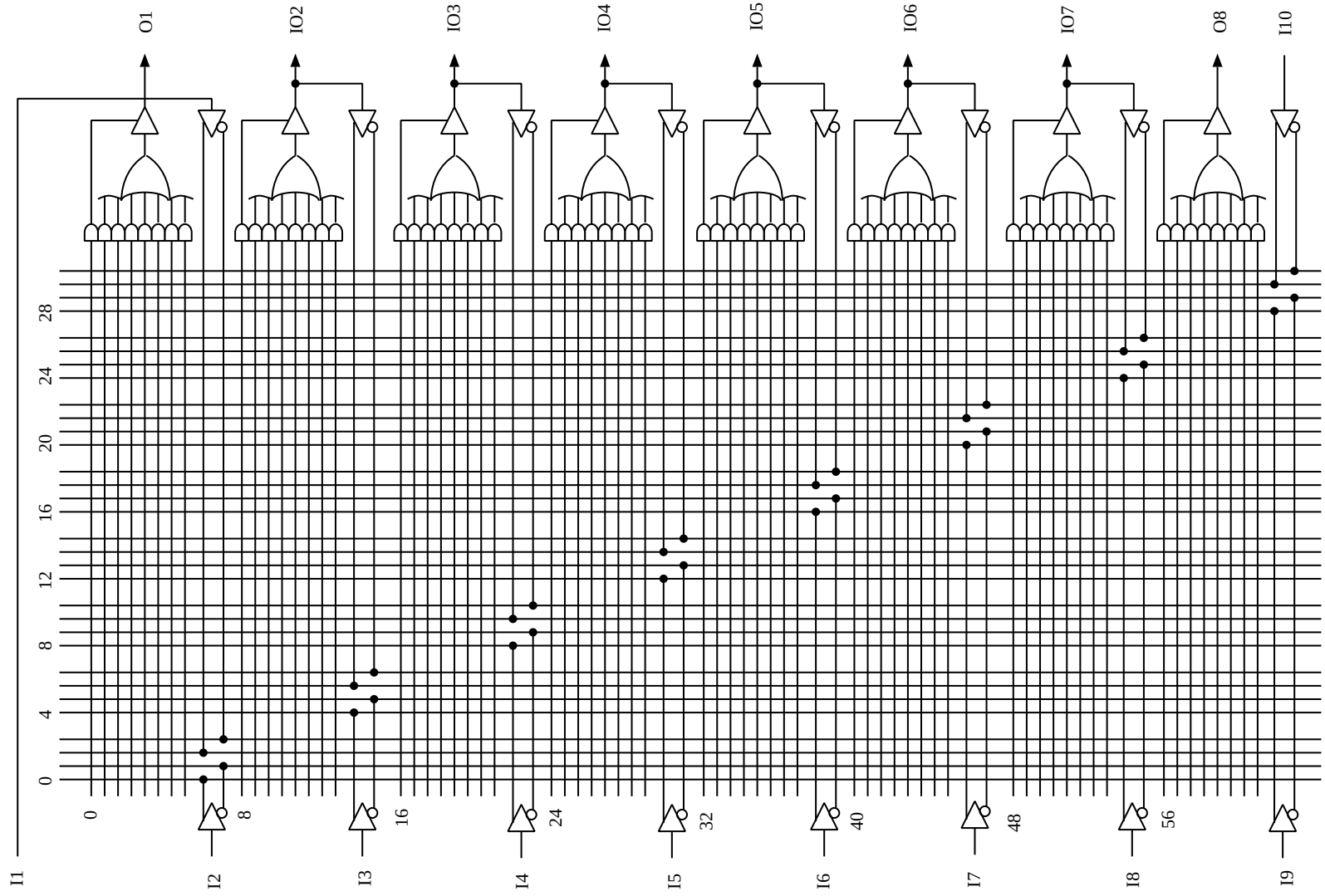


Figure 5.21: 16-input, 8-output PAL(P16H8)