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 incremen ter

**Input:** $0 \leq x \leq 63$

**Output:** $0 \leq z \leq 63$

**Function:** $z = (x + 1) \mod 64$

<table>
<thead>
<tr>
<th>$x$</th>
<th>$010101$</th>
<th>$x$</th>
<th>$001111$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$z$</td>
<td>$010110$</td>
<td>$z$</td>
<td>$010000$</td>
</tr>
</tbody>
</table>

- **Radix-2 representation**

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

$$z_5 = x_5(x'_4 + x'_3 + x'_2 + x'_1 + x'_0) + x'_5x'_4x'_3x'_2x'_1x_0$$
$$z_4 = x_4x'_3 + x_4x'_2 + x'_4x'_1 + x'_4x'_0 + x'_4x'_3x'_2x'_1x_0$$
$$z_3 = x_3x'_2 + x_3x'_1 + x'_3x'_0 + x'_3x'_2x'_1x_0$$
$$z_2 = x_2x'_1 + x'_2x'_0 + x'_2x'_1x_0$$
$$z_1 = x_1x'_0 + x'_1x_0$$
$$z_0 = x'_0$$
Figure 5.1: NOT-AND-OR modulo-64 incrementer network.
Two types of two-level networks:

- **AND-OR network** ⇔ Sum of products (NAND-NAND network)
- **OR-AND network** ⇔ Product of sums (NOR-NOR network)

\[ E(x_2, x_1, x_0) = x'_2 x'_1 x'_0 + x_2 x_1 + x_1 x'_0 \]

\[ E(x_2, x_1, x_0) = (x'_2 + x_1)(x_1 + x'_0)(x_2 + x'_1 + x_0) \]
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 $f(x_2, x_1, x_0) = one-set(3, 6, 7)$. 

Introduction to Digital Systems 5 - Design of Two-Level Gate Networks
Minimal expressions

- Equivalent but different cost

\[ E_1(x_2, x_1, x_0) = x'_2x'_1x'_0 + x'_1x_0 + x_2x_0 \]
\[ E_2(x_2, x_1, x_0) = x_2x_0 + x'_2x'_1x'_0 + x'_2x'_1x_0 + x_2x'_1x_0 \]

- 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 \( \leftrightarrow \) \( 2^n \) cells
- cell \( i \) \( \leftrightarrow \) assignment \( i \)
  
  \[ \textit{adjacency condition} \]
  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 5.1: K-Maps

(a) (b) (c) (d)

Introduction to Digital Systems
Design of Two-Level Gate Networks
Figure 5.5: K-map for five variables

\[ x_4 = 0 \]

\[ x_4 = 1 \]
Representation of switching functions

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

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

\[ f(x_2, x_1, x_0) = [\text{one-set}(0, 4, 5), \quad \text{dc-set}(2, 3)] \]
Rectangles of 1-cells and sum of products

1. Minterm $m_j$ corresponds to 1-cell with label $j$.

2. Product term of $n \leftrightarrow 1$ literals $\longleftrightarrow$ rectangle of two adjacent 1-cells.

\[ x_3 x'_1 x_0 = x_3 x'_1 x_0 (x_2 + x'_2) \]
\[ = x_3 x_2 x'_1 x_0 + x_3 x'_2 x'_1 x_0 \]
\[ = m_{13} + m_9 \]

![Figure 5.6](image-url)
3. Product term of $n \leftrightarrow 2$ literals $\longleftrightarrow$ rectangle of four adjacent 1-cells.

\[ x_3x_0 = x_3x_0(x_1 + x'_1)(x_2 + x'_2) = x_3x'_2x'_1x_0 + x_3x'_2x_1x_0 + x_3x_2x'_1x_0 + x_3x_2x_1x_0 = m_9 + m_{11} + m_{13} + m_{15} \]

Figure 5.6

4. Product term of $n \leftrightarrow s$ literals $\longleftrightarrow$ rectangle of $2^s$ adjacent 1-cells.
Figure 5.7: Representation of product of \( n \equiv (a + b) \) variables.

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

represented in a K-map by the union of rectangles

\[ E(x_3, x_2, x_1, x_0) = x'_3x_2x_1 + x'_2x_1x_0 + x'_0 \]

\[ E(a, b, c) = ab + ac + b'c' \]
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 \(\leftrightarrow\) sum term of \(n \Leftrightarrow (a + b)\) literals
Minimization of sums of products

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

![Figure 5.9: Implicant representation.](image)

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 Pls

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

\[
\begin{array}{c|ccc}
   & x_0 & x_1 & x_2 \\
---&---&---&---
0 & 0 & 0 & 0 \\
0 & 0 & 0 & 1 \\
1 & 0 & 0 & 1 \\
1 & 0 & 0 & 0 \\
\end{array}
\]

Pls: \( x_2x'_0 \) and \( x_1x'_0 \)

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

\[
\begin{array}{c|ccc}
   & x_0 & x_1 & x_2 \\
---&---&---&---
1 & 1 & 0 & 0 \\
1 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 \\
\end{array}
\]

Pls: \( x'_2x'_1 \), \( x_2x_0 \), and \( x'_1x_0 \)
c) $f(x_3, x_2, x_1, x_0) = \text{one-set}(0,3,5,7,11,12,13,15)$

<table>
<thead>
<tr>
<th></th>
<th>$x_0$</th>
<th>$x_1$</th>
<th>$x_2$</th>
<th>$x_3$</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
</tbody>
</table>

Pls: $x_2x_0, \ x_1x_0, \ x_3x_2x_1', \ \text{and} \ x_3'x_2'x_1'x_0'$
Minimal sum of products consists of prime implicants

Figure 5.10: Minimal sum of products and prime implicants.
Example 5.9

\[ E(x_2, x_1, x_0) = x_2 x_1^\prime x_0^\prime + x_2 x_1 x_0^\prime + x_1 x_0^\prime \]

not Pls: \( x_2 x_1^\prime x_0^\prime \) and \( x_2 x_1 x_0^\prime \)

Pl: \( x_2 x_0^\prime, \ x_1 x_0^\prime \)

reduced SP: \( E(x_2, x_1, x_0) = x_2 x_0^\prime + x_1 x_0^\prime \)
Essential Prime Implicants (EPI)

\[ p_e(a) = 1 \text{ and } p(a) = 0 \] for any other PI \( p \)

EPIs: \( x'_1 x'_0 \) and \( x_1 x_0 \)

non-essential: \( x_2 x_1, x_2 x'_0 \).

- 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) \( E(x_3, x_2, x_1, x_0) = x'_3x'_2 + x'_3x_2x_0 + x_1x_0 \)

- PIs: \( x'_3x'_2, \ x'_3x_0, \) and \( x_1x_0 \)
- all EPIs
- unique min SP: \( x'_3x'_2 + x'_3x_0 + x_1x_0 \)
b) \( E(x_2, x_1, x_0) = \Sigma m(0, 3, 4, 6, 7) \)

- Pls: \( x'_1 x'_0, \ x_1 x_0, \ x_2 x'_0, \text{ and } x_2 x_1 \)

- EPls: \( x'_1 x'_0 \text{ and } x_1 x_0 \)

- extra cover: \( x_2 x'_0 \text{ or } x_2 x_1 \)

- Two min SPs:
\[
  x'_1 x'_0 + x_1 x_0 + x_2 x'_0 \text{ and } x'_1 x'_0 + x_1 x_0 + x_2 x_1
\]
c) \( E(x_2, x_1, x_0) = \Sigma m(0, 1, 2, 5, 6, 7) \)

- PIs: \( x'_2x'_1, x'_2x'_0, x_2x_0, x_2x_1, x'_1x_0, \) and \( x_1x'_0 \)
- No EPIs
- Two min SPs

\[
x'_2x'_1 + x_2x_0 + x'_1x_0 \quad \text{and} \quad x'_2x'_0 + x'_1x_0 + x_2x_1
\]
Minimal SPs for incompletely specified functions

A minimal SP

\[ E(x_3, x_2, x_1, x_0) = x_3x'_0 + x'_3x_0 + x'_3x'_2x'_1 \]
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

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

<table>
<thead>
<tr>
<th></th>
<th></th>
<th>( x_0 )</th>
<th>( x_1 )</th>
<th>( x_2 )</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>

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: \((x'_0 \oplus x'_2)\) and \((x_0 \oplus x_2 \oplus x'_1)\)

- Both essential, the minimal PS is \((x'_0 \oplus x'_2)(x_0 \oplus x_2 \oplus x'_1)\)
Minimal two-level gate network design: Example 5.14

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

Output: \( z \in \{0, 1\} \)

Function: \( z = \begin{cases} 1 & \text{if } x \in \{0, 2, 3, 5, 8\} \\ 0 & \text{otherwise} \end{cases} \)

The values \( \{10, 11, 12, 13, 14, 15\} \) are “don’t cares”

\[
\begin{array}{cccc}
0 & 1 & 0 & 1 \\
1 & 0 & 1 & 0 \\
0 & 1 & 0 & 0 \\
1 & 0 & 1 & \text{X} \\
\end{array}
\]

\begin{align*}
\text{min SP: } z &= x'_2 x_1 + x'_2 x'_0 + x_2 x'_1 x_0 \\
\text{min PS: } z &= (x'_2 + x'_1)(x'_2 + x_0)(x_2 + x_1 + x'_0)
\end{align*}
Figure 5.11: Minimal AND-OR network
Example 5.15

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

Output: \( z \in \{0, 1\} \)

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

The K-map:

\[ \begin{array}{c|cccc} & x_0 & x_1 & x_2 & x_3 \\ \hline 00 & \text{1} & \text{1} & \text{0} & \text{0} \\ 01 & \text{0} & \text{1} & \text{1} & \text{0} \\ 10 & \text{0} & \text{1} & \text{0} & \text{1} \\ 11 & \text{0} & \text{0} & \text{1} & \text{1} \end{array} \]

\begin{align*}
\text{min SP: } z &= x_3'x_0 + x_3'x_2x_1' + x_2x_1'x_0 + x_3x_2x_0' + x_2'x_1x_0 \\
\text{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') \\
\text{Cost(PS)} &< \text{Cost(SP)}
\end{align*}
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: \(z = \sum_{i=0}^{2} x_i\)

1. The switching functions in tabular form are

<table>
<thead>
<tr>
<th>(x_2)</th>
<th>(x_1)</th>
<th>(x_0)</th>
<th>(z_1)</th>
<th>(z_0)</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>
Example 5.16 (cont.)

2. The corresponding K-maps are

\[
\begin{array}{c|cccc}
& x_0 & x_1 & x_2 & x_3 \\
\hline
x_0 & 0 & 0 & 1 & 0 \\
x_1 & 0 & 1 & 1 & 1 \\
x_2 & 0 & 1 & 1 & 1 \\
0 & 0 & 1 & 0 & 0 \\
\end{array}
\]

\[
\begin{array}{c|cccc}
& x_0 & x_1 & x_2 & x_3 \\
\hline
x_0 & 0 & 1 & 0 & 1 \\
x_1 & 1 & 0 & 1 & 0 \\
x_2 & 1 & 0 & 1 & 0 \\
\end{array}
\]

3. minimal SPs:

\[
z_1 = x_2x_1 + x_2x_0 + x_1x_0
\]

\[
z_0 = x_2'x_1'x_0 + x_2'x_1x_0' + x_2x_1x_0' + x_2x_1x_0
\]

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

\[ E = \text{NAND}(\text{NAND}_1, \text{NAND}_2, \text{NAND}_3, \ldots, \text{NAND}_n) \]
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: \( d = (d_3, d_2, d_1, d_0), \quad d_j \in \{0, 1\} \)

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

Function:

<table>
<thead>
<tr>
<th>( i )</th>
<th>( d_3d_2d_1d_0 )</th>
<th>( g_3g_2g_1g_0 )</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0000</td>
<td>0000</td>
</tr>
<tr>
<td>1</td>
<td>0001</td>
<td>0001</td>
</tr>
<tr>
<td>2</td>
<td>0010</td>
<td>0011</td>
</tr>
<tr>
<td>3</td>
<td>0011</td>
<td>0010</td>
</tr>
<tr>
<td>4</td>
<td>0100</td>
<td>0110</td>
</tr>
<tr>
<td>5</td>
<td>0101</td>
<td>0111</td>
</tr>
<tr>
<td>6</td>
<td>0110</td>
<td>0101</td>
</tr>
<tr>
<td>7</td>
<td>0111</td>
<td>0100</td>
</tr>
<tr>
<td>8</td>
<td>1000</td>
<td>1100</td>
</tr>
<tr>
<td>9</td>
<td>1001</td>
<td>1101</td>
</tr>
</tbody>
</table>

Expressions:

\[
\begin{align*}
g_3 &= d_3 \\
g_2 &= d_3 \oplus d_2 \\
g_1 &= d_2' d_1 + d_2 d_1' \\
g_0 &= d_1 d_0' + d_1' d_0
\end{align*}
\]
Note: a PLA chip would have more rows and columns than shown here.

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

![Logic diagram of a PAL](image)

Figure 5.20: Logic diagram of a PAL

- AND Array
- × -- programmable connection
- ● -- connection made

**Introduction to Digital Systems**

5 - Design of Two-Level Gate Networks
Figure 5.21: 16-input, 8-output PAL (P16H8)