SPECIFICATION OF COMBINATIONAL SYSTEMS

- High-level and binary-level specifications.
- Representation of data elements (signal values) by binary variables (signals) and standard codes for positive integers and characters.
- Representation by switching functions and switching expressions.
- NOT, AND, OR, NOR, XOR, and XNOR switching functions and their representation. Gate symbols.
- Transformation of switching expressions using the switching algebra.
- Use of various specification methods
- Use of the μVHDL description language.

Combinational System

$$z(t) = F(x(t)) \quad \text{or} \quad z = F(x)$$

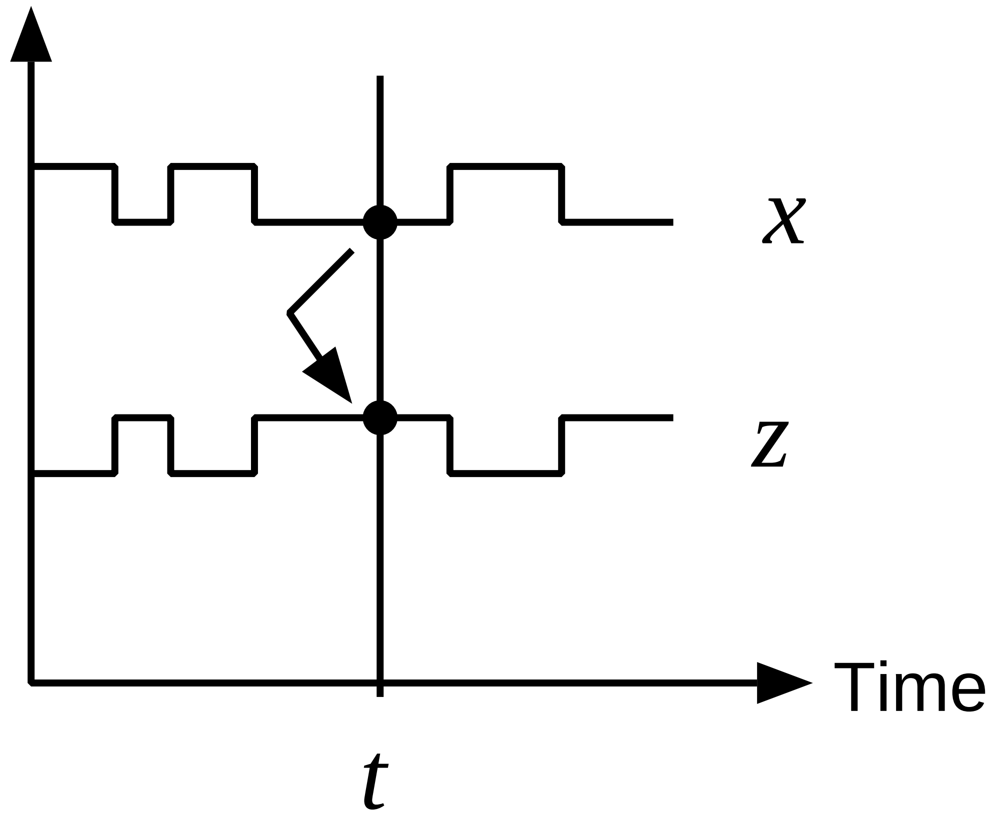


Figure 2.1: Combinational system.

Binary level

$$\underline{z}_b = F_b(\underline{x}_b)$$

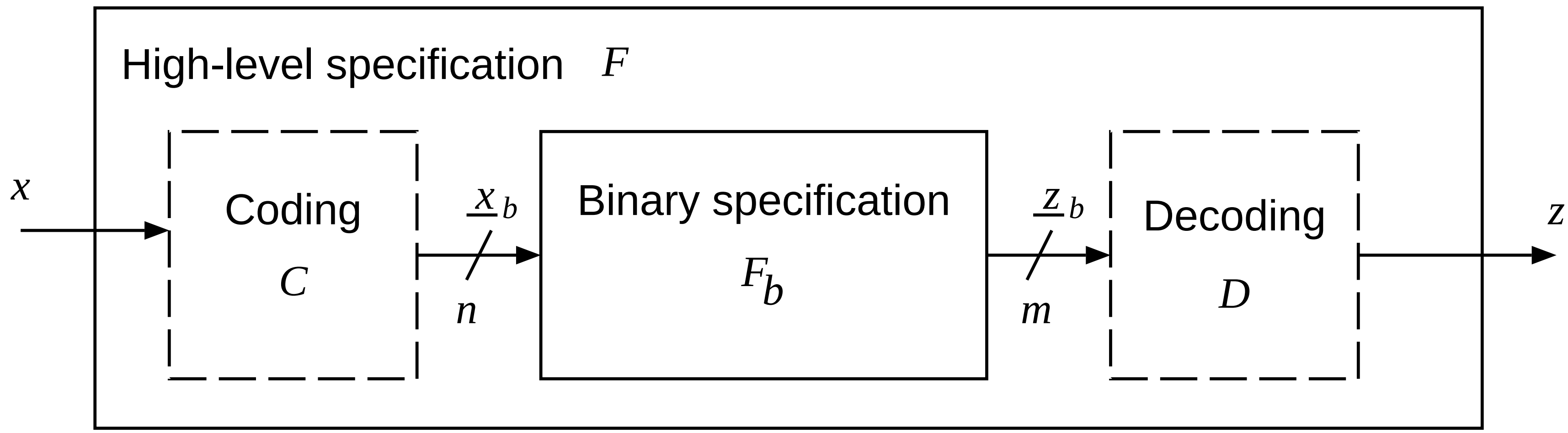


Figure 2.2: High-level and binary-level specification.

Example 2.1:

Input: $x \in \{0,1,2,3,4,5,6,7,8,9\}$
Output: $z \in \{0,1,2\}$

Function: F is described by the following table

x	0	1	2	3	4	5	6	7	8	9
$z = F(x)$	0	1	2	0	1	2	0	1	2	0

or by the arithmetic expression

$$z = x \bmod 3,$$

x	0	1	2	3	4	5	6	7	8	9
$\underline{x}_b = C(x)$	0000	0001	0010	0011	0100	0101	0110	0111	1000	1001

$\underline{z}_b$	00	01	10
$z = D(\underline{z}_b)$	0	1	2

Input: $\underline{x}_b = (x_3, x_2, x_1, x_0),\ x_i \in \{0,1\}$
Output: $\underline{z}_b = (z_1, z_0),\ z_i \in \{0,1\}$

Function: F_b is described by the following table

$\underline{x}_b$	0000	0001	0010	0011	0100	0101	0110	0111	1000	1001
$\underline{z}_b = F_b(\underline{x}_b)$	00	01	10	00	01	10	00	01	10	00

High-level specification

- The set of values for the input, *input set;*
- The set of values for the output, *output set;* and
- The specification of the *input-output function.*

Input and output sets

$$\{\text{UP}, \quad \text{DOWN}, \quad \text{LEFT}, \quad \text{RIGHT}, \quad \text{FIRE}\}$$

$$\{x \mid (5 \leq x \leq 10^4) \text{ and } (x \bmod 3 = 0)\}$$

Examples of vectors

Vector type		Example
Digit	$\underline{x} = (x_{n-1}, x_{n-2}, \ldots, x_0)$ $x_i \in \{0, 1, 2, \ldots, 9\}$	$\underline{x} = (7, 0, 6, 3)$
Character	$\underline{c} = (c_{n-1}, c_{n-2}, \ldots, c_0)$ $c_i \in \{\ , A, B, \ldots, Z\}$	$\underline{c} = (B, O, O, K)$
Set	$\underline{s} = (s_{n-1}, s_{n-2}, \ldots, s_0)$ $s_i \in \{\text{red}, \text{ blue}, \text{ white}\}$	$\underline{s} = (\text{red}, \text{blue}, \text{blue})$
Bit	$\underline{y} = (y_{n-1}, y_{n-2}, \ldots, y_0)$ $y_i \in \{0, 1\}$	$\underline{y} = (1, 1, 0, 1, 0, 0)$ $\underline{y} = 110100$

Input-output function

1. *Table*

x	z
A	65
B	66
C	67
D	68
E	69

2. *Arithmetic expression*

$$z = 3x + 2y - 2$$

3. *Conditional expression*

$$z = \begin{cases} a+b & \textbf{if} \quad c > d \\ a-b & \textbf{if} \quad c = d \\ 0 & \textbf{if} \quad c < d \end{cases}$$

Input-output function (cont.)

4. *Logical expression*

$$z = (\text{SWITCH1} = \text{CLOSED}) \textbf{ and } (\text{SWITCH2} = \text{OPEN}) \textbf{ or } (\text{SWITCH3} = \text{CLOSED})$$

5. *Composition of simpler functions*

GREATER:

$$\text{MAX}(v, w, x, y) = \text{GREATER}(v, \text{GREATER}(w, \text{GREATER}(x, y)))$$

in which

$$\text{GREATER}(a, b) = \begin{cases} a & \textbf{if } \; a > b \\ b & \textbf{otherwise} \end{cases}$$

Example 2.2

Inputs: $\underline{x} = (x_3, x_2, x_1, x_0)$,
$x_i \in \{\mathrm{A}, \mathrm{B}, \ldots, \mathrm{Z}, \mathrm{a}, \mathrm{b}, \ldots, \mathrm{z}\}$
$y \in \{\mathrm{A}, \mathrm{B}, \ldots, \mathrm{Z}, \mathrm{a}, \mathrm{b}, \ldots, \mathrm{z}\}$
$k \in \{0, 1, 2, 3\}$

Outputs: $\underline{z} = (z_3, z_2, z_1, z_0)$,
$z_i \in \{\mathrm{A}, \mathrm{B}, \ldots, \mathrm{Z}, \mathrm{a}, \mathrm{b}, \ldots, \mathrm{z}\}$

Function: $z_j = \begin{cases} x_j & \textbf{if} \quad j \neq k \\ y & \textbf{if} \quad j = k \end{cases}$

Input: $\underline{x} = (\mathrm{C,A,S,E})$, $y = \mathrm{R}$, $k = 1$
Output: $\underline{z} = (\mathrm{C,A,R,E})$

Data representation and coding

{AL, BERT, DAVE, JERRY, LEN}

	Fixed-length Code 1	Fixed-length Code 2	Variable-length Code 3
AL	000	0110	01
BERT	010	0101	001
DAVE	100	0011	0001
JERRY	110	1001	00001
LEN	111	1111	000001

Alphanumeric Codes

Character	Codes	
	ASCII	EBCDIC
A	100 0001	1100 0001
B	100 0010	1100 0010
C	100 0011	1100 0011
⋮	⋮	⋮
Y	101 1001	1110 1000
Z	101 1010	1110 1001
0	011 0000	1111 0000
1	011 0001	1111 0001
2	011 0010	1111 0010
⋮	⋮	⋮
8	011 1000	1111 1000
9	011 1001	1111 1001
blank	010 0000	0100 0000
.	010 1110	0100 1011
(	010 1000	0100 1101
+	010 1011	0100 1110
⋮	⋮	⋮

A	blank	C	A	B	.
01000001	00100000	01000011	01000001	01000010	00101110

Representation of positive integers

Level 1: integer $\Longleftrightarrow$ *digit-vector*

Level 2: digits $\Longleftrightarrow$ *bit-vector*

Level 1: Integer (Digit-vector)	5			6			3			0		
Level 2: Bit-vector	1	0	1	1	1	0	0	1	1	0	0	0

Representation by digit-vector

$$\underline{x} = (x_{n-1}, x_{n-2}, \ldots, x_1, x_0)$$

$$x = \sum_{i=0}^{n-1} x_i r^i$$

digit x_i in $\{0, 1, \ldots, r-1\}$, r – the radix

$$\underline{x} = (1, 0, 0, 1, 0, 1)$$

$\Longleftrightarrow$

$$1 \times 2^5 + 0 \times 2^4 + 0 \times 2^3 + 1 \times 2^2 + 0 \times 2^1 + 1 \times 2^0 = (37)_{10}$$

Set of representable values

$$0 \leq x \leq r^n - 1$$

Example of Binary Codes

Digit Value (Symbol)	Binary	Quaternary	Octal	Hexadecimal
	$k = 1$	$k = 2$	$k = 3$	$k = 4$
	d_0	$d_1 d_0$	$d_2 d_1 d_0$	$d_3 d_2 d_1 d_0$
0	0	00	000	0000
1	1	01	001	0001
2		10	010	0010
3		11	011	0011
4			100	0100
5			101	0101
6			110	0110
7			111	0111
8				1000
9				1001
10 (A)				1010
11 (B)				1011
12 (C)				1100
13 (D)				1101
14 (E)				1110
15 (F)				1111

Binary-code representation of a digit-vector

<table>
<tr><td colspan="3">5</td><td colspan="3">6</td><td colspan="3">3</td><td colspan="3">0</td></tr>
<tr><td>1</td><td>0</td><td>1</td><td>1</td><td>1</td><td>0</td><td>0</td><td>1</td><td>1</td><td>0</td><td>0</td><td>0</td></tr>
</table>

a) Gray code. b) Gray-code bit-vector representation of a digit-vector.

Digit	Gray code
0	000
1	001
2	011
3	010
4	110
5	111
6	101
7	100

(a)

<table>
<tr><td colspan="3">4</td><td colspan="3">5</td><td colspan="3">3</td><td colspan="3">0</td></tr>
<tr><td>1</td><td>1</td><td>0</td><td>1</td><td>1</td><td>1</td><td>0</td><td>1</td><td>0</td><td>0</td><td>0</td><td>0</td></tr>
</table>

(b)

Codes for Decimal Digits

Digit Value	BCD 8421	2421	Excess-3	2-Out-of-5
0	0000	0000	0011	00011
1	0001	0001	0100	11000
2	0010	0010	0101	10100
3	0011	0011	0110	01100
4	0100	0100	0111	10010
5	0101	1011	1000	01010
6	0110	1100	1001	00110
7	0111	1101	1010	10001
8	1000	1110	1011	01001
9	1001	1111	1100	00101

Binary-level specification of C - systems

Switching functions

Figure 2.3: a) Binary combinational system; b) a switching function for $n = 3$

Tabular representation of switching functions

n-tuple notation		Simplified notation	
$x_2x_1x_0$	$f(x_2,x_1,x_0)$	j	$f(j)$
0 0 0	0	0	0
0 0 1	0	1	0
0 1 0	1	2	1
0 1 1	1	3	1
1 0 0	0	4	0
1 0 1	0	5	0
1 1 0	1	6	1
1 1 1	1	7	1

	$x_2x_1x_0$							
x_4x_3	000	001	010	011	100	101	110	111
00	0	0	1	1	0	1	1	1
01	0	1	1	1	1	0	1	1
10	1	1	0	1	1	0	1	1
11	0	1	0	1	1	0	1	0

f

Important switching functions

Table 2.10: Switching functions of one variable

x	f_0 0-CONSTANT (always 0)	f_1 IDENTITY (equal to x)	f_2 COMPLEMENT (NOT)	f_3 1-CONSTANT (always 1)
0	0	0	1	1
1	0	1	0	1

Table 2.11: Switching functions of two variables

Function	x_1x_0 00	01	10	11	
f_0	0	0	0	0	
f_1	0	0	0	1	AND
f_2	0	0	1	0	
f_3	0	0	1	1	
f_4	0	1	0	0	
f_5	0	1	0	1	
f_6	0	1	1	0	EXCLUSIVE-OR (XOR)
f_7	0	1	1	1	OR
f_8	1	0	0	0	NOR
f_9	1	0	0	1	EQUIVALENCE (EQU)
f_{10}	1	0	1	0	
f_{11}	1	0	1	1	
f_{12}	1	1	0	0	
f_{13}	1	1	0	1	
f_{14}	1	1	1	0	NAND
f_{15}	1	1	1	1	

Incomplete switching functions

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

or

$$[\ one\text{-}set(1,4,7),\ zero\text{-}set(0,2,5)\ ]$$
$$[\ one\text{-}set(1,4,7),\ dc\text{-}set(3,6)\ ]$$
$$[\ zero\text{-}set(0,2,5),\ dc\text{-}set(3,6)\ ]$$

Composition of switching functions

$$
\begin{aligned}
\mathrm{AND}(x_3, x_2, x_1, x_0) &= \mathrm{AND}(\mathrm{AND}(x_3, x_2), \mathrm{AND}(x_1, x_0)) \\
\mathrm{XOR}(x_1, x_0) &= \mathrm{OR}(\mathrm{AND}(\mathrm{NOT}(x_0), x_1), \mathrm{AND}(x_0, \mathrm{NOT}(x_1))) \\
\mathrm{MAJ}(x_3, x_2, x_1, x_0) &= \mathrm{OR}(\mathrm{AND}(x_3, x_2, x_1), \mathrm{AND}(x_3, x_2, x_0), \\
&\qquad \mathrm{AND}(x_3, x_1, x_0), \mathrm{AND}(x_2, x_1, x_0))
\end{aligned}
$$

Switching expressions

1. The symbols 0 and 1 are SEs.
2. A symbol representing a binary variable is a SE.
3. If A and B are SEs, then
   - $(A)'$ is a SE. This is referred to as "A complement." Sometimes we use $\overline{A}$ to denote complementation.
   - $(A) + (B)$ is a SE. This is referred as "A OR B"; it is also called "A plus B" or "sum" due to the similarity with the corresponding arithmetic symbol.
   - $(A) \cdot (B)$ is a SE. This is referred to as "A AND B"; it is also called "A times B" or "product" due to the similarity with the corresponding arithmetic symbol.

Precedence rules: $'$ precedes $\cdot$ which precedes $+$

Well-formed switching expressions are:

$$x_0 \qquad x_1 + x_2x_3' \qquad 1 + 0(x + y)$$

whereas $(x_1 +' x_2 +)x_3$ and "This is a switching expression" are not.

Switching algebra and expression evaluation

- *Switching algebra*:

  two elements 0 and 1

  operations $+$, $\cdot$, and $'$

$+$	0	1
0	0	1
1	1	1

$\cdot$	0	1
0	0	0
1	0	1

$'$	
0	1
1	0

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

The value of E for assignment $(1, 0, 1)$ is

$$E(1, 0, 1) = 1 + 1' \cdot 0 + 0 \cdot 1' = 1 + 0 + 0 = 1$$

Representing switching functions by switching expressions

$E(x_2, x_1, x_0) = x_2 + x_2' x_1 + x_1 x_0'$ represents f:

$x_2x_1x_0$	f
000	0
001	0
010	1
011	1
100	1
101	1
110	1
111	1

	2 variables	n variables
AND	x_1x_0	$x_{n-1}x_{n-2}\ldots x_0$
OR	$x_1 + x_0$	$x_{n-1} + x_{n-2} + \ldots + x_0$
XOR	$x_1x_0' + x_1'x_0 = x_1 \oplus x_0$	
EQUIV	$x_1'x_0' + x_1x_0$	
NAND	$(x_1x_0)' = x_1' + x_0'$	$(x_{n-1}x_{n-2}\ldots x_0)' = x_{n-1}' + x_{n-2}' + \ldots + x_0'$
NOR	$(x_1 + x_0)' = x_1'x_0'$	$(x_{n-1} + x_{n-2} + \ldots + x_0)' = x_{n-1}'x_{n-2}'\ldots x_0'$

Switching functions and gates

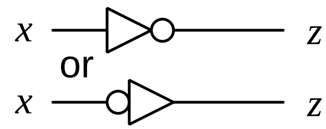
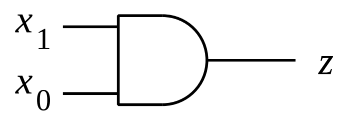
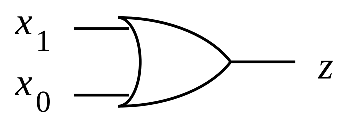
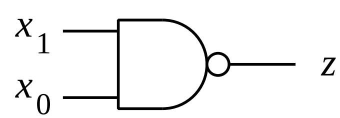
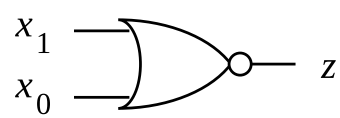
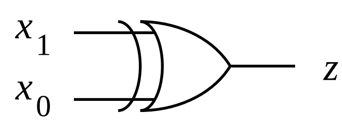
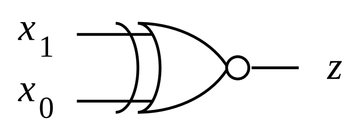
Gate type	Symbol	Switching expression
NOT		$z = x'$
AND		$z = x_1 x_0$
OR		$z = x_1 + x_0$
NAND		$z = (x_1 x_0)'$
NOR		$z = (x_1 + x_0)'$
XOR		$z = x_1 x_0' + x_1' x_0$ $= x_1 \oplus x_0$
XNOR		$z = x_1' x_0' + x_1 x_0$

Figure 2.4: Gate symbols

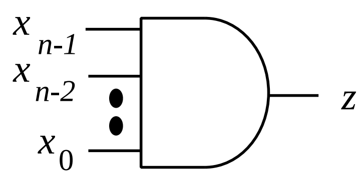
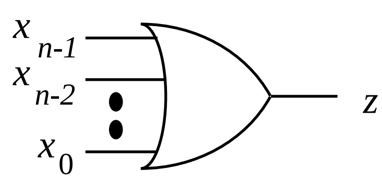
Gate type	Symbol	Switching expression
AND		$z = x_{n-1}\, x_{n-2} \,....\, x_0$
OR		$z = x_{n-1} + x_{n-2} \,....+ x_0$

Figure 2.5: n-input AND and OR gate symbols

Equivalent switching expressions

W and Z switching expressions are equivalent

$$W = x_1 x_0 + x_1'$$
$$Z = x_1' + x_0$$

The corresponding switching functions are

$x_1 x_0$	W	Z
00	1	1
01	1	1
10	0	0
11	1	1

Equivalence Classes

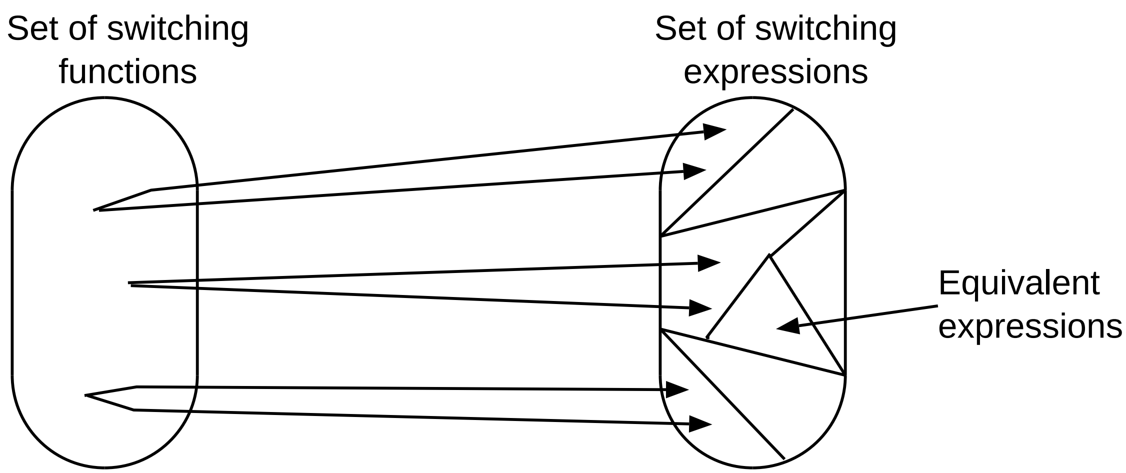


Figure 2.6: Correspondence among switching functions and switching expressions

Algebraic method of obtaining equivalent expressions

- The principal identities of Boolean algebra

1.	$a + b$	$= b + a$	ab	$= ba$	Commutativity
2.	$a + (bc)$	$= (a + b)(a + c)$	$a(b + c)$	$= (ab) + (ac)$	Distributivity
3.	$a + (b + c)$	$= (a + b) + c$	$a(bc)$	$= (ab)c$	Associativity
		$= a + b + c$		$= abc$	
4.	$a + a$	$= a$	aa	$= a$	Idempotency
5.	$a + a'$	$= 1$	aa'	$= 0$	Complement
6.	$1 + a$	$= 1$	$0a$	$= 0$	
7.	$0 + a$	$= a$	$1a$	$= a$	Identity
8.	$(a')'$	$= a$			Involution
9.	$a + ab$	$= a$	$a(a + b)$	$= a$	Absorption
10.	$a + a'b$	$= a + b$	$a(a' + b)$	$= ab$	Simplification
11.	$(a + b)'$	$= a'b'$	$(ab)'$	$= a' + b'$	DeMorgan's Law

Example

Show that E_1 and E_2 are equivalent:

$$\begin{aligned} E_1(x_2, x_1, x_0) &= x_2x_1 + x_2x_1' + x_2x_0 \\ E_2(x_2, x_1, x_0) &= x_2 \end{aligned}$$

$$\begin{aligned} x_2x_1 + x_2x_1' + x_2x_0 &= x_2(x_1 + x_1') + x_2x_0 && \text{using } ab + ac = a(b + c) \\ &= x_2 \cdot 1 + x_2x_0 && \text{using } a + a' = 1 \\ &= x_2(1 + x_0) && \text{using } ab + ac = a(b + c) \\ &= x_2 \cdot 1 && \text{using } 1 + a = 1 \\ &= x_2 && \text{using } a \cdot 1 = a \end{aligned}$$

Sum of products and sum of minterms

Literals	x, y, z', x'
Product terms	x_0, x_2x_1, $x_3x_1x_0'$
Sum of products	$x_2' + x_3x_1' + x_3'x_1'x_0$

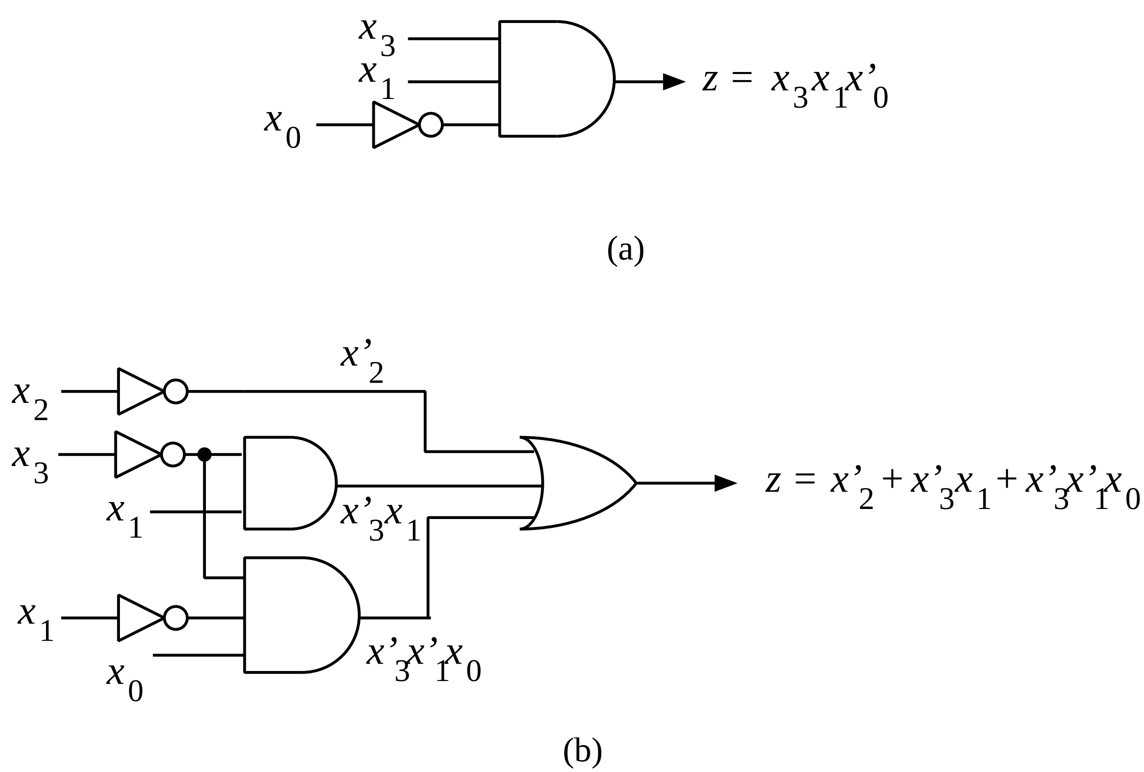


Figure 2.7: Sum of products and AND-OR gate network: a) Product term. b) Sum of products.

Minterm notation

$$x_i \longleftrightarrow 1; \qquad x'_i \longleftrightarrow 0$$

Minterm m_j, j integer

Example: minterm $x_3x'_2x'_1x_0$ denoted m_9
because $1001 = 9$

$$m_j(\underline{a}) = \begin{cases} 1 & \textbf{if} \quad a = j \\ 0 & \textbf{otherwise} \end{cases}$$

$$a = \sum_{i=0}^{n-1} a_i 2^i$$

Example: $m_{11} = x_3x'_2x_1x_0$
– has value 1 only for $\underline{a} = (1, 0, 1, 1)$

Minterm functions

$x_2x_1x_0$	m_0	m_1	m_2	m_3	m_4	m_5	m_6	m_7
	$x_2'x_1'x_0'$	$x_2'x_1'x_0$	$x_2'x_1x_0'$	$x_2'x_1x_0$	$x_2x_1'x_0'$	$x_2x_1'x_0$	$x_2x_1x_0'$	$x_2x_1x_0$
000	1	0	0	0	0	0	0	0
001	0	1	0	0	0	0	0	0
010	0	0	1	0	0	0	0	0
011	0	0	0	1	0	0	0	0
100	0	0	0	0	1	0	0	0
101	0	0	0	0	0	1	0	0
110	0	0	0	0	0	0	1	0
111	0	0	0	0	0	0	0	1

Example

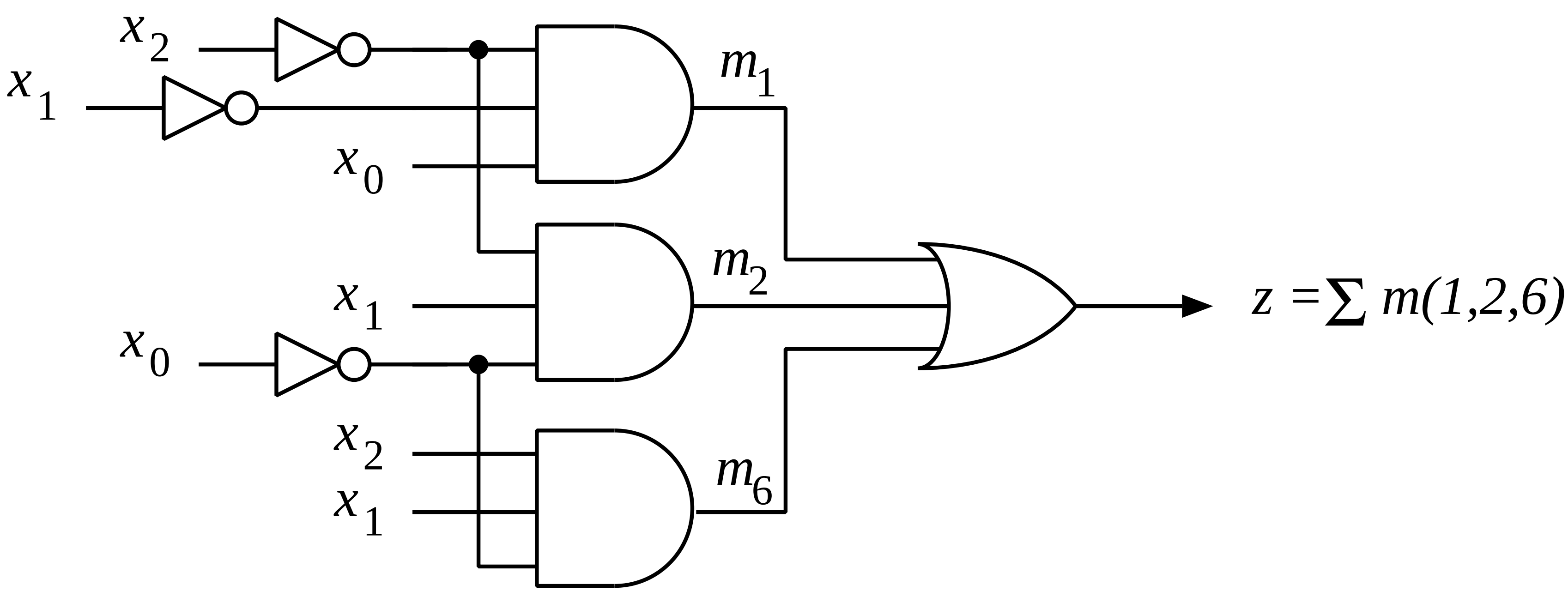


Figure 2.8: Gate network corresponding to $E(x_2, x_1, x_0) = \sum m(1, 2, 6)$.

Example 2.12: Table $\rightarrow$ sum of minterms

j	$x_2\ x_1\ x_0$	f
0	000	0
1	001	0
2	010	1
3	011	1
4	100	0
5	101	1
6	110	0
7	111	0

$$E = \sum m(2,3,5) = x_2'x_1x_0' + x_2'x_1x_0 + x_2x_1'x_0$$

Product of sums and product of maxterms

Sum terms	$x_0,\ x_2 + x_1,\ x_3 + x_1 + x_0'$
Product of sums	$(x_2' + x_3 + x_1')(x_3' + x_1)x_0$

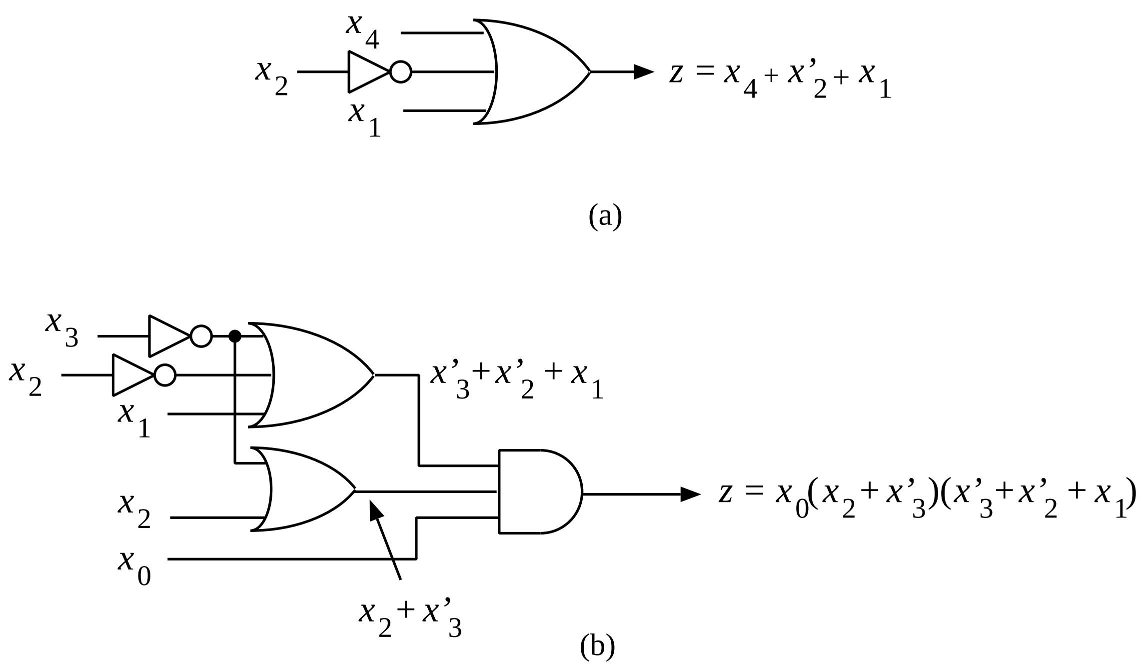


Figure 2.9: Product of sums and OR-AND gate network. a) Sum term. b) Product of sums.

Maxterm Notation

$$x_i \longleftrightarrow 0; \qquad x_i' \longleftrightarrow 1$$

Maxterm M_j , j integer

Example: maxterm $x_3 + x_2' + x_1 + x_0'$ denoted M_5
because $0101 = 5$

$$M_j(\underline{a}) = \begin{cases} 0 & \textbf{if} \quad a = j \\ 1 & \textbf{otherwise} \end{cases}$$

Example: $M_5 = x_3 + x_2' + x_1 + x_0'$
– has value 0 only for assignment 0101

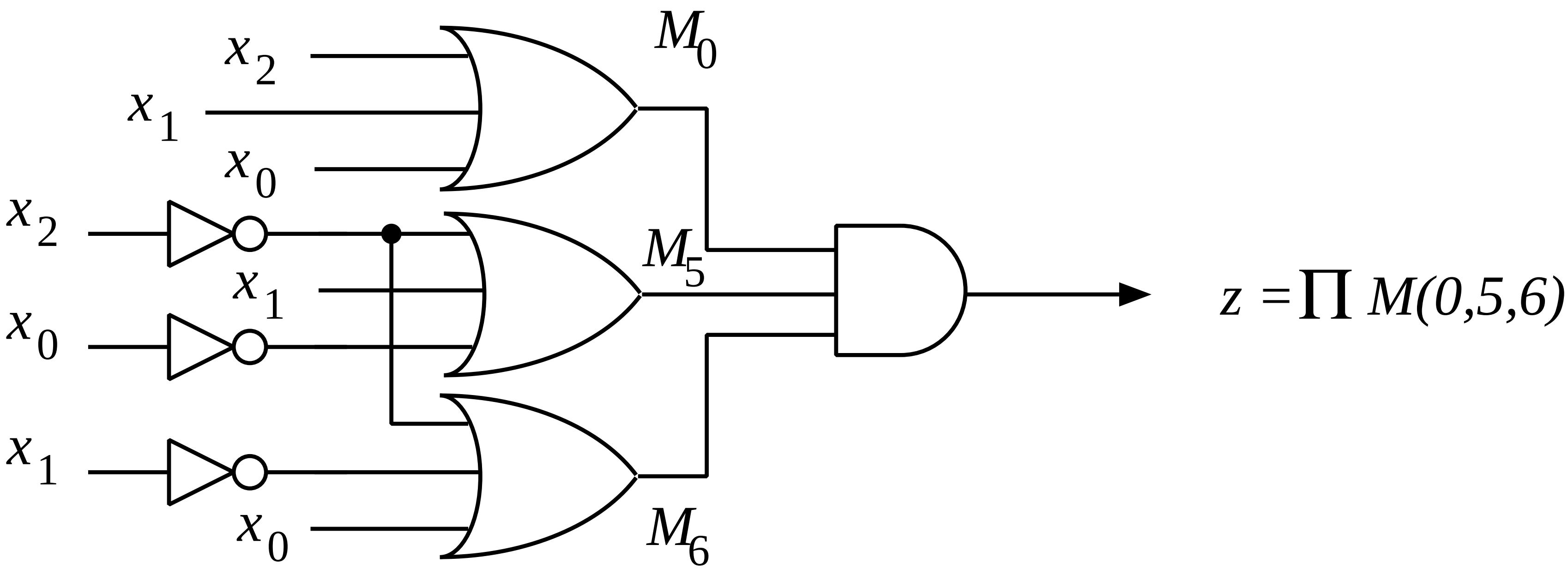


Figure 2.10: Gate network corresponding to $E(x_2, x_1, x_0) = \prod M(0, 5, 6)$.

Example 2.15: Table $\rightarrow$ product of sums

j	$x_2x_1x_0$	f
0	000	0
1	001	1
2	010	1
3	011	0
4	100	0
5	101	0
6	110	1
7	111	0

$$\begin{aligned} E(x_2, x_1, x_0) &= \Pi M(0,3,4,5,7) \\ &= (x_2 + x_1 + x_0)(x_2 + x_1' + x_0')(x_2' + x_1 + x_0) \\ &\qquad (x_2' + x_1 + x_0')(x_2' + x_1' + x_0') \end{aligned}$$

Conversion among canonical forms

Sum of minterms $\longleftrightarrow$ *one-set*
Product of maxterms $\longleftrightarrow$ *zero-set*
$\Rightarrow$ conversion straightforward

$$\sum m(\{j \mid f(j) = 1\}) = \prod M(\{j \mid f(j) = 0\})$$

Example:

m-notation:

$$f(x, y, z) = \sum m(0, 4, 7)$$

M-notation:

$$f(x, y, z) = \prod M(1, 2, 3, 5, 6)$$

Example 2.19: Radix-4 Comparator

Inputs: $x, y \in \{0, 1, 2, 3\}$
Output: $z \in \{\text{G,E,S}\}$

Function: $$z = \begin{cases} \text{G} & \textbf{if} \quad x > y \\ \text{E} & \textbf{if} \quad x = y \\ \text{S} & \textbf{if} \quad x < y \end{cases}$$

		y			
		0	1	2	3
x	0	E	S	S	S
	1	G	E	S	S
	2	G	G	E	S
	3	G	G	G	E

z

Example 2.19 (cont.)

Coding:

$$x = 2x_1 + x_0 \quad \text{and} \quad y = 2y_1 + y_0$$

z	$z_2 z_1 z_0$
G	100
E	010
S	001

Binary specification:

$$z_2 = \begin{cases} 1 \textbf{ if } x_1 > y_1 \textbf{ or } (x_1 = y_1 \textbf{ and } x_0 > y_0) \\ 0 \textbf{ otherwise} \end{cases}$$

$$z_1 = \begin{cases} 1 \textbf{ if } x_1 = y_1 \textbf{ and } x_0 = y_0 \\ 0 \textbf{ otherwise} \end{cases}$$

$$z_0 = \begin{cases} 1 \textbf{ if } x_1 < y_1 \textbf{ or } (x_1 = y_1 \textbf{ and } x_0 < y_0) \\ 0 \textbf{ otherwise} \end{cases}$$

	y_1y_0			
x_1x_0	00	01	10	11
00	010	001	001	001
01	100	010	001	001
10	100	100	010	001
11	100	100	100	010

$z_2z_1z_0$

Example 2.19 (cont)

Switching expressions:

$$
\begin{aligned}
z_2(x_1, x_0, y_1, y_0) &= \sum m(4, 8, 9, 12, 13, 14) \\
z_1(x_1, x_0, y_1, y_0) &= \sum m(0, 5, 10, 15) \\
z_0(x_1, x_0, y_1, y_0) &= \sum m(1, 2, 3, 6, 7, 11)
\end{aligned}
$$