SPECIFICATION OF COMBINATIONAL SYSTEMS - SUMMARY[1]

- HIGH-LEVEL AND BINARY-LEVEL SPECIFICATIONS
- REPRESENTATION OF DATA ELEMENTS BY BINARY VARIABLES; STANDARD CODES FOR POSITIVE INTEGERS AND CHARACTERS
- REPRESENTATION BY SWITCHING FUNCTIONS AND SWITCHING EXPRESSIONS
- BASIC SWITCHING FUNCTIONS: NOT, AND, OR, XOR, XNOR

- BOOLEAN ALGEBRA (Appendix A)
- SWITCHING ALGEBRA
- TRANSFORMATION OF SWITCHING EXPRESSIONS USING SWITCHING ALGEBRA
- USE OF VARIOUS SPECIFICATION METHODS:
  - TABLES
  - COMPOSITION OF FUNCTIONS
  - ARITHMETIC, CONDITIONAL AND SWITCHING EXPRESSIONS
- USE OF THE μVHDL DESCRIPTION LANGUAGE

COMBINATIONAL SYSTEM

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

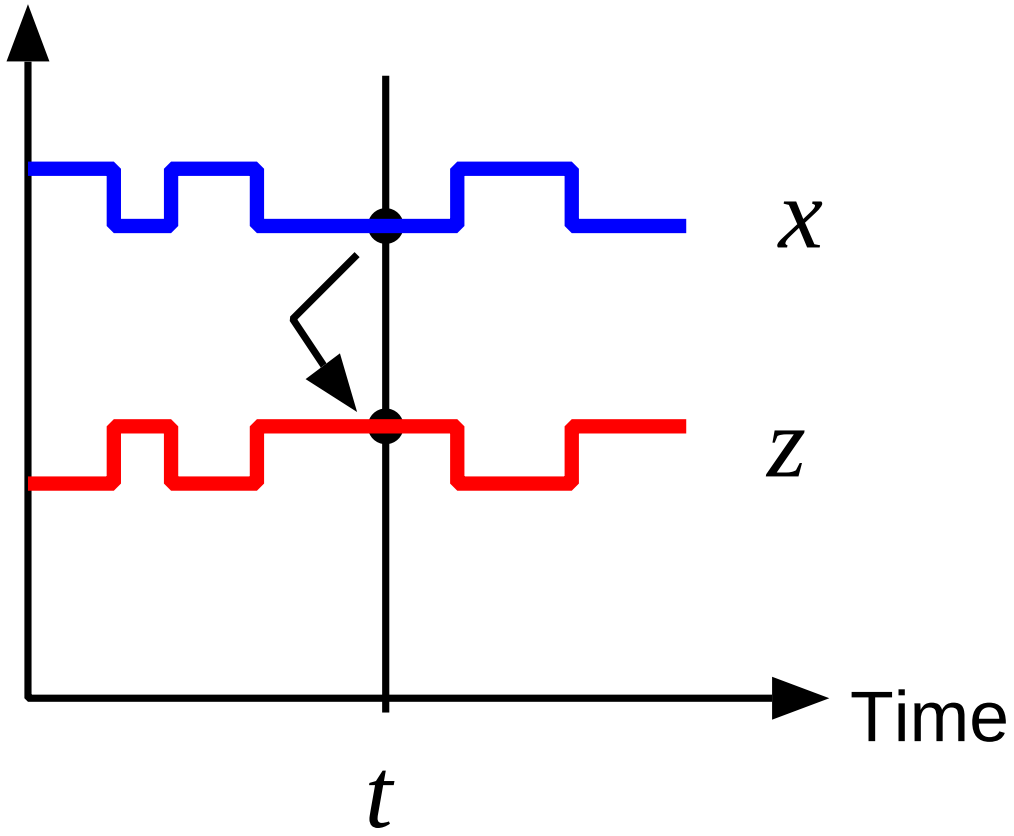


Figure 2.1: Combinational system.

BINARY LEVEL

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

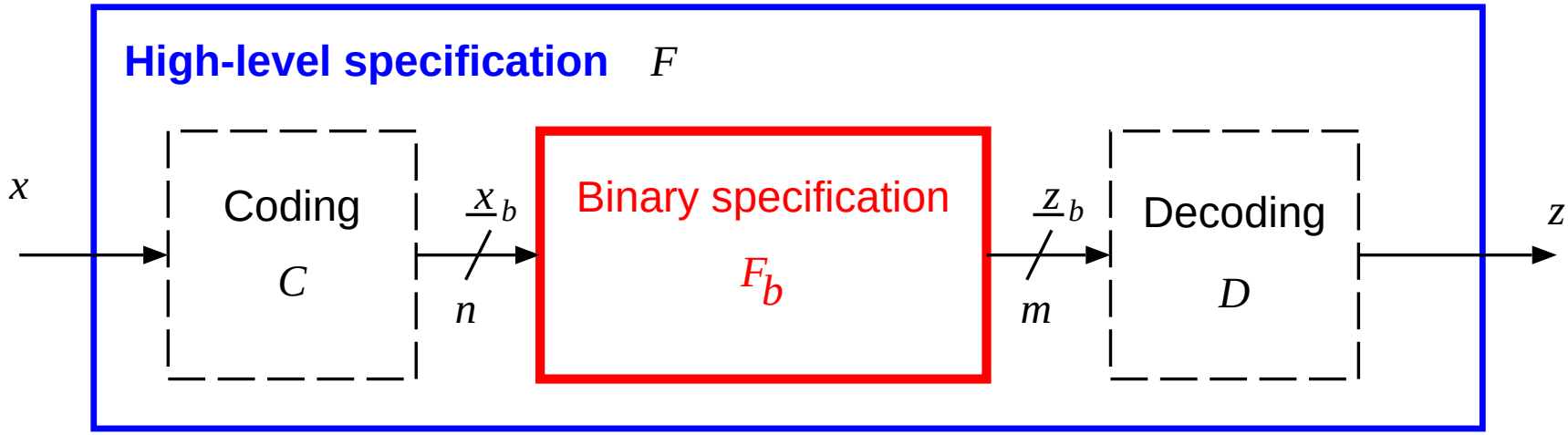


Figure 2.2: High-level and binary-level specification.

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

$$z = x \bmod 3,$$

Example 2.1 (cont.): BINARY ENCODING OF x AND z

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: $\underline{x}_b = (x_3, x_2, x_1, x_0),\ x_i \in \{0, 1\}$

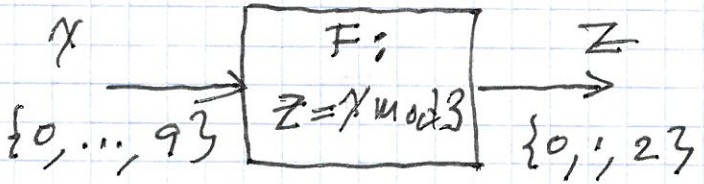
Output: $\underline{z}_b = (z_1, z_0),\ z_i \in \{0, 1\}$

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

CS MSIA EXAMPLE 2.1 BLOCK DIAGRAMS

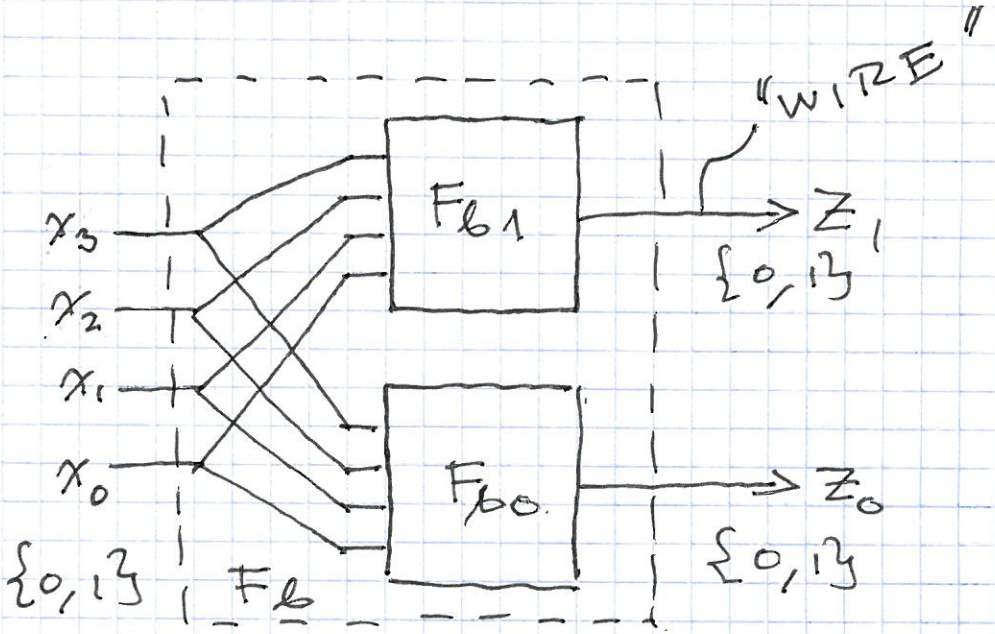
a) HIGH-LEVEL



VARIABLES FROM INPUT & OUTPUT SETS

– TYPICALLY MULTIVALUED

b) BINARY-LEVEL



ALL VARIABLES $\in \{0,1\}$ (BINARY)

HIGH-LEVEL SPECIFICATION

- SET OF VALUES FOR THE INPUT, *input set;*
- SET OF VALUES FOR THE OUTPUT, *output set;* and
- SPECIFICATION OF THE *input-output function.*

INPUT AND OUTPUT SETS

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

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

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*

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

5. *COMPOSITION OF SIMPLER FUNCTIONS*

$$max(v, w, x, y) = GREATER(v, GREATER(w, GREATER(x, y$$

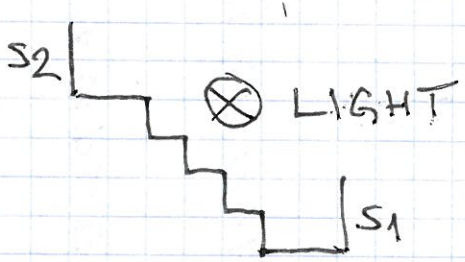
in which

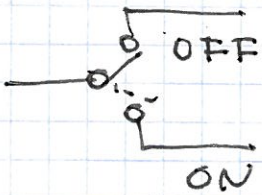
$$GREATER(a, b) = \begin{cases} a \textbf{ if } \ a > b \\ b \textbf{ otherwise} \end{cases}$$

ILLUSTRATIONS OF I/O FUNCTIONS AT HIGH LEVEL

#4 LOGICAL EXPRESSION

PROBLEM: SPECIFY A SYSTEM TO CONTROL A STAIRWELL LIGHT FROM TWO FLOORS USING SWITCHES S2 AND S1



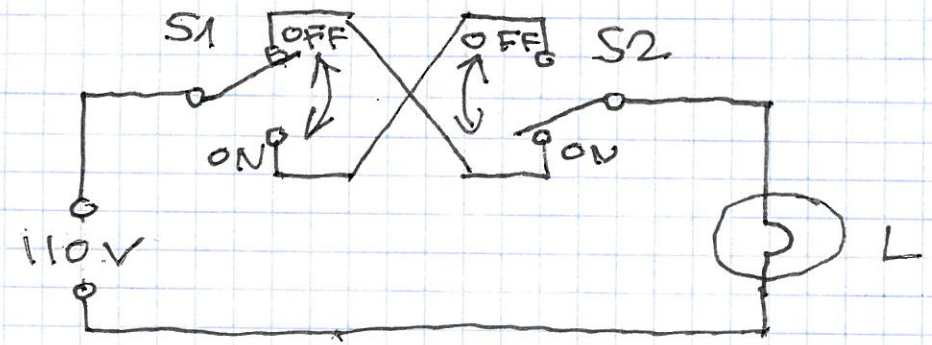


L IS ON IF ((S1 IS OFF) AND (S2 IS ON))
OR ((S1 IS ON) AND (S2 IS OFF))

$\Rightarrow L = f(S1, S2)$

S1	S2	L
OFF	OFF	OFF
OFF	ON	ON
ON	OFF	ON
ON	ON	OFF

CONTROL SYSTEM:



DARK → LIGHT
LIGHT → DARK

FLIP A SWITCH

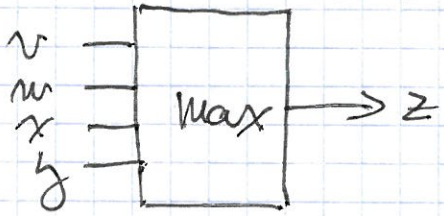
CHALLENGE: EXTEND TO 3 FLOORS!

ILLUSTRATIONS OF I/O FUNCTIONS AT HIGH LEVEL

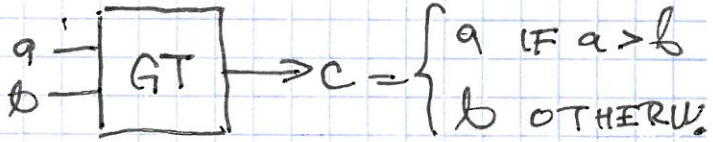
#5 COMPOSITION OF SIMPLER FUNCTIONS

- ONE OF THE MOST IMPORTANT IDEAS IN THE DESIGN OF DIGITAL SYSTEMS
- FOLLOWS A HIERARCHICAL APPROACH (DIVIDE & CONQUER)

PROBLEM: $z = \max(v, w, x, y)$ (+ INTEGERS)

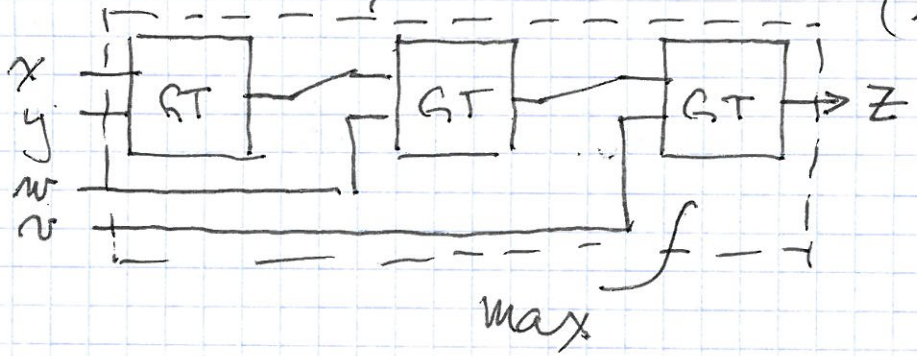


MODULE:

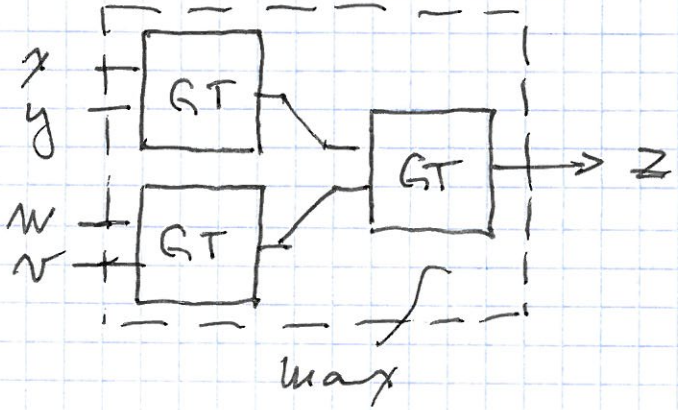


SOLUTIONS:

(a)



(b)



SCHEME:	COST	DELAY	
a	3	3	
b	3	2	←

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

Output: $\underline{z} = (\mathrm{C},\mathrm{A},\mathrm{R},\mathrm{E})$

DATA REPRESENTATION AND CODING

$$\{Al,\ Bert,\ Dave,\ Jerry,\ Len\}$$

	Fixed-length		Variable-length
	Code 1	Code 2	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: DIGIT $\Longleftrightarrow$ *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 OF POSITIVE INTEGERS

M. ERCEGOVAC

1. NUMBER REPRESENTATION SYSTEM (NRS)

SET OF VALUES (FINITE)

$\{x\} \xrightarrow[NRS]{} \{\underline{x}\}$ SET OF DIGIT-VECTORS

DIGIT-VECTOR $\underline{x} = (x_5, x_4, \ldots, x_0)_r$ r - RADIX (BASE)

DIGIT SET $x_i \in \{0, 1, \ldots, r-1\}$

$r = 4$ $\underline{x} = (3, 0, 1, 2)_4 = 3012_4$

VALUE OF $\underline{x}$

$$x = \sum_{i=0}^{n-1} x_i r^i \qquad n - \#\text{ OF DIGITS}$$

$$\underline{x} = 3012_4 \longrightarrow x = 3 \cdot 4^3 + 0 \cdot 4^2 + 1 \cdot 4^1 + 2 \cdot 4^0 = 198_{10}$$

GIVEN r AND n, THE SET OF REPRESENTABLE VALUES IS

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

$r = 2 \quad n = 10 \quad 0 \le x \le 1023$

GIVEN r AND x_{max}

$$n = \lceil \log_r x_{max} \rceil$$

2. REPRESENTATION OF DIGITS

WEIGHTED CODE: DIGIT VALUE d

$$d = \sum_{i=0}^{k-1} d_i w_i$$

FOR $r = 2^k$ (BINARY COMPATIBLE)

$$d = \sum_{i=0}^{k-1} d_i 2^i$$

IN GENERAL:

$$w_0 = 2^0 = 1$$

$$w_{i+1} = w_i \cdot r$$

OR $w_i = r^i$

Ex. $r = 4$ $k = 2$

d	$d_1 d_0$
0	0 0
1	0 1
2	1 0
3	1 1

etc. FOR OTHER k's

FOR $r = 10$ SEVERAL POPULAR CODES

→ SEE TABLE 2.3

EXAMPLE OF BINARY CODES FOR DIGITS ($r = 2^k$)

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

FOR DECIMAL DIGITS:

BCD CODE (BINARY CODED DECIMAL)

WEIGHTS 8, 4, 2, 1 ($n=4$)

$5_{10} \leftrightarrow 0\cdot8 + 1\cdot4 + 0\cdot2 + 1\cdot1 \rightarrow 0101_2$

$9_{10} \leftrightarrow 1\cdot8 + 0\cdot4 + 0\cdot2 + 1\cdot1 \rightarrow 1001_2$

2421 CODE

WEIGHTS: 2, 4, 2, 1

$5_{10} \leftrightarrow$ 1011 OR 0101

COMPLENT WRT 9, $9-d$, OBTAINED BY COMPLEMENTING EACH BIT WRT 1, $1-d_i$.

→ SELF-COMPLEMENTING CODE

USEFUL IN PERFORMING SUBTRACTION

EXCESS-3 CODE: BCD + 3

$5_{10} = 0101_{BCD} = 1000_{EX-3}$

NOT WEIGHTED; SELF-COMPLEMENTING

2-OUT-OF-5 CODE: TWO 1's IN EACH BIT-VECTOR

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

ON 2-4-2-1 CODE (TABLE 2-3) [EXERCISE 2.9]

a) IS IT UNIQUE?

b) IF NOT UNIQUE, HOW MANY DIFFERENT 2-4-2-1 CODES ARE THERE?

	2	4	2	1	
0	0	0	0	0	
1	0	0	0	1	
2	0	0	1	0	}
	1	0	0	0	}
3	0	0	1	1	}
	1	0	0	1	}
4	0	1	0	0	}
	1	0	1	0	}
5	0	1	0	1	}
	1	0	1	1	}
6	0	1	1	0	}
	1	1	0	0	}
7	0	1	1	1	}
	1	1	0	1	}
8	1	1	1	0	
9	1	1	1	1	

$2^6 = 64$ CODES

c) ARE ALL OF THESE CODES SELF-COMPLEMENTING?

IF NO, HOW MANY SELF-COMPLEMENTING CODES ARE THERE?

CS MSIA COMMENTS ON CODES

2421 CODE: 64 DIFFERENT CODES; 32 ARE SELFCOMPLENTIN

CONSIDER

2	0 0 1 0	←
	1 0 0 0	←
7	0 1 1 1	←
	1 1 0 1	←

YES NO → 1/2

2-OUT-OF-5 CODE: SINGLE ERROR CAN BE DETECTED

CHECKER C: INPUT $\underline{x} = (x_4\, x_3\, x_2\, x_1\, x_0)_2$

$S = (S_2, S_1, S_0)$

$C = S_2' S_1 S_0'$

OUTPUT $C = \begin{cases} 1 & \text{IF } S = \sum_{i=0}^{4} x_i = 2 \\ 0 & \text{OTHERWISE} \end{cases}$

($C = 1 \Rightarrow$ NO SINGLE ERROR)

A DESIGN:

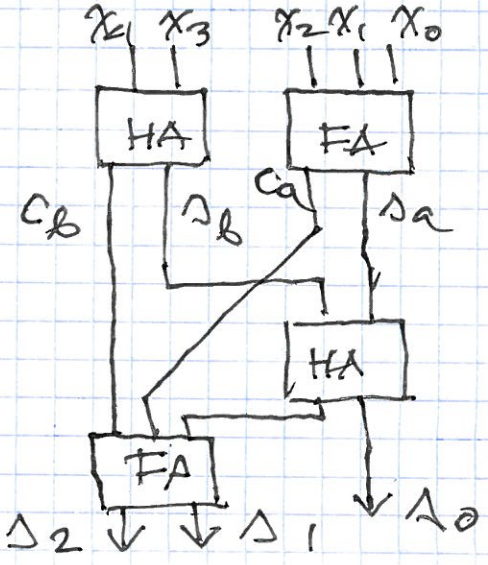
$x_2 + x_1 + x_0 = 2C_a + S_a$ ← FULL ADDER

$x_4 + x_3 = 2C_b + S_b$ ← HALF ADDER

ADD $C_a\ S_a$

$C_b\ S_b$

$S_2\ S_1\ S_0$



COMMENTS ON NUMBER REPRESENTATION CONVERSION

CONSIDER $0 \leq X < 1$

$$X_2 = 0,\ 11101 = \frac{29}{32} = 0,90625_{10}$$

FINITE # DIGITS

$$X_{10} = 0.1 = \frac{1}{10} \Rightarrow 0,0001100\overline{1100}_{REDEATS}$$

INFINITE # BITS

$\Rightarrow$ CONVERSION INTRODUCES AN ERROR

- DECIMAL ARITHMETIC WOULD BE PREFERABLE IN FINANCIAL CALCULATIONS
- EARLY COMPUTERS WERE DECIMAL. BINARY COMPUTERS, BEING MORE EFFICIENT (FASTER & SIMPLER), ARE DOMINANT.

HOWEVER, DECIMAL ARITHMETIC IS MAKING A COMEBACK BECAUSE OF IMPORTANCE OF FINANCIAL COMPUTING AND MORE EFFICIENT HARDWARE. BINARY ARITHMETIC REMAINS MAINSTREAM

EFFICIENCY OF REPRESENTATION

OF BITS NEEDED TO REPRESENT A SET OF VALUES

EXAMPLE: $0 \leq x \leq 10^6 - 1$

IN BCD: 6 DIGITS $\Rightarrow 6 \times 4 = \underline{24}$ BITS

IN BINARY:

$2^{20} - 1 > 10^6$, $2^{19} - 1 < 10^6 - 1$

$\Rightarrow$ NEED $\underline{20}$ BITS

BINARY REP. IS MORE EFFICIENT THAN BCD REPRESENTATION.

WHY IS THAT?

COMPLEMENT $C = 9 - d$

USEFUL IN SUBTRACTION

— HOW TO OBTAIN C?

BCD

	9	1001
	$d = 5$	0101
	$C = 4$	0100

SUBTRACT

2421

	9	1111
	$d = 5$	0101
	$C = 4$	1010

BIT-COMPLEMENT

EX-3

	9	1100
	$d = 5$	1000
	$C = 4$	0111

BIT-COMPLEMENT

2421 & EX-3 ARE SELF-COMPLEMENTING CODES

CONVERSIONS

1. a) DECIMAL → BCD (EX-3, etc)

$6\ \ 1\ \ 0\ \ 3_{10}$

0110 0001 0000 0011 BCD

1001 0100 0011 0110 EX-3

b) BCD → DECIMAL

1000 0010 0101 BCD

8 2 5

SIMILAR FOR OTHER CODES

c) RADIX 2 ⟷ RADIX 2^k

— MAKE k-BIT GROUPS

$X = 10\,|\,00\,|\,01\,|\,11_2$ $\quad 1\cdot2^7 + 1\cdot2^2 + 1\cdot2 + 1\cdot1 = 135_{10}$

$2\ \ 0\ \ 1\ \ 3_4$ $\quad 2\cdot4^3 + 1\cdot4 + 3\cdot1 = 135_{10}$

EXTEND → $010\,|\,000\,|\,111_2$

$2\ \ 0\ \ 7_8$ $\quad 2\cdot8^2 + 7 = 135_{10}$

2. a) DECIMAL TO BINARY (INTEGERS)

- KEEP DIVIDING BY 2 UNTIL QUOTIENT = 1
- TAKE REMAINDERS AS BITS
- LAST QUOTIENT (1) IS THE MOST SIGNIFICANT (MS) BIT

135	→	1 LS
67	→	1
33	→	1
16	→	0
8	→	0
4	→	0
2	→	0
1	→	1 MS

1 0 0 0 0 1 1 1

128 7

135

b) DECIMAL TO BINARY (FRACTIONS)

- KEEP MULTIPLYING BY 2 AS NEEDED OR DESIRED
- TAKE INTEGER PART AS BIT AND REMOVE

0.8	0
1.6	1
1.2	1
0.4	0
0.8	0
1.6	1

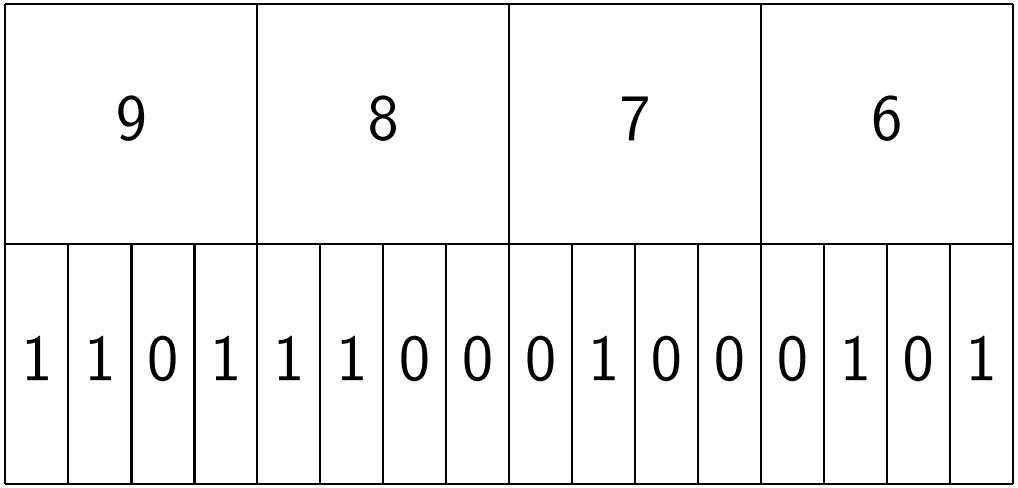
REPEATS!

$\frac{25}{32} \approx 0.78$

ERROR!

Digit	Gray code
0	0000
1	0001
2	0011
3	0010
4	0110
5	0111
6	0101
7	0100
8	1100
9	1101
10	1111
11	1110
12	1010
13	1011
14	1001
15	1000

GRAY CODE BIT-VECTOR REPRESENTATION OF A DIGIT-VECTOR



BINARY-LEVEL SPECIFICATION OF C - SYSTEMS

SWITCHING FUNCTIONS

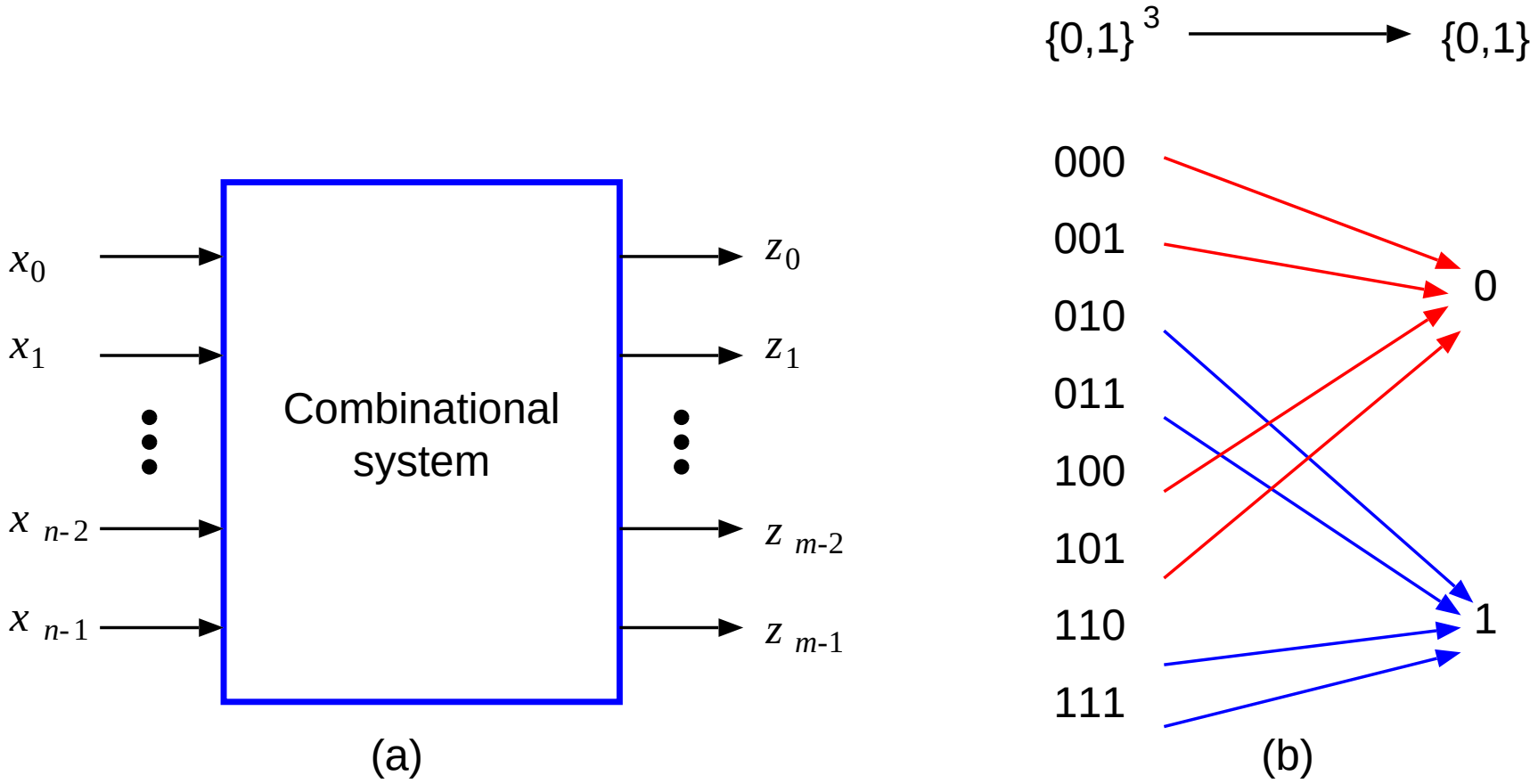


Figure 2.7: a) BINARY COMBINATIONAL SYSTEM; b) A SWITCHING FUNCTION FOR $n = 3$

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

2D TABULAR REPRESENTATION

x_4x_3 \ $x_2x_1x_0$	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

More compact

IMPORTANT SWITCHING FUNCTIONS

Table 2.4: 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 2.5: 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

EXAMPLE OF AN INCOMPLETE SF:

$x \in \{0, \ldots, 9\}$ $\quad \underline{x} = \{0000, 0001, \ldots, 1001\}_{BCD}$

$$z = f(x) = \begin{cases} 1 & \text{if } x \text{ IS ODD} \\ 0 & \text{OTHERWISE} \end{cases}$$

$\{1010, 1011, \ldots, 1111\}$ NEVER OCCUR

j	z
0	0
1	1
2	0
3	1
4	0
5	1
6	0
7	1
8	0
9	1
10	—
11	—
12	—
13	—
14	—
15	—

d.c.-SET $= \{10, 11, 12, \ldots, 15\}$

one-SET $= \{1, 3, 5, 7, 9\}$

zero-SET $= \{0, 2, 4, 6, 8\}$

Can be represented with any of the following

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

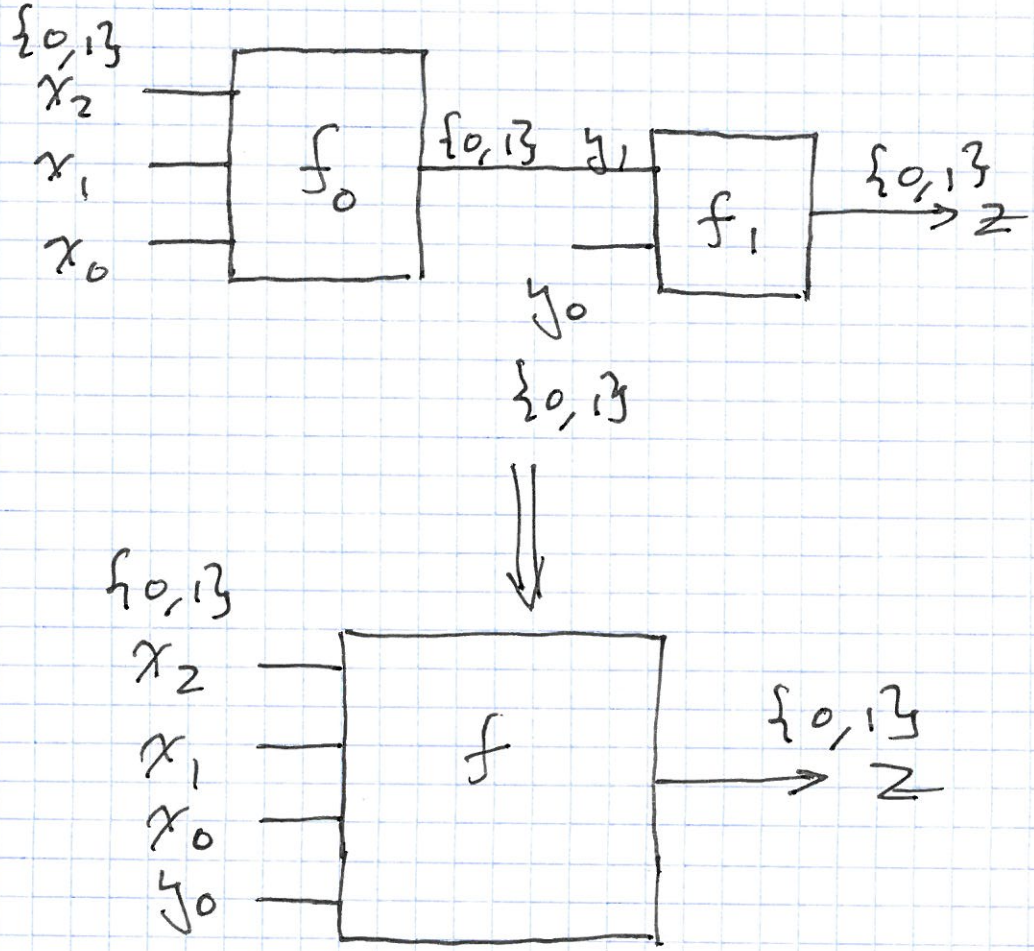
COMPOSITION OF SWITCHING FUNCTIONS

$$\mathrm{AND}(x_3, x_2, x_1, x_0) = \mathrm{AND}(\mathrm{AND}(x_3, x_2), \mathrm{AND}(x_1, x_0))$$

$$\mathrm{XOR}(x_1, x_0) = \mathrm{OR}(\mathrm{AND}(\mathrm{NOT}(x_0), x_1), \mathrm{AND}(x_0, \mathrm{NOT}(x_1)))$$

$$\mathrm{MAJ}(x_3, x_2, x_1, x_0) = \mathrm{OR}(\mathrm{AND}(x_3, x_2, x_1), \mathrm{AND}(x_3, x_2, x_0), \mathrm{AND}(x_3, x_1, x_0), \mathrm{AND}(x_2, x_1, x_0))$$

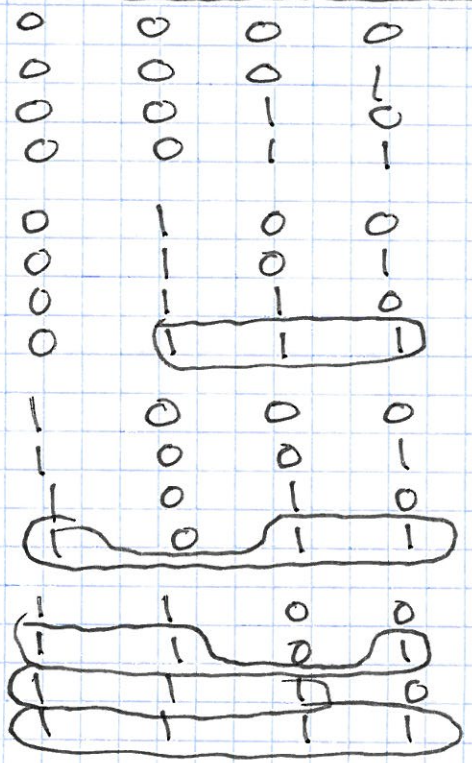
EXAMPLE OF COMPOSITION OF SFs



MAJORITY SF

$$MAJ(x_3, x_2, x_1, x_0) = \begin{cases} 1 & \text{IF 3 OR 4 VARIABLES} = 1 \\ 0 & \text{OTHERWISE} \end{cases}$$

x_3	x_2	x_1	x_0	MAJ($\underline{x}$)
0	0	0	0	0
0	0	0	1	0
0	0	1	0	0
0	0	1	1	0
0	1	0	0	0
0	1	0	1	0
0	1	1	0	0
0	1	1	1	1
1	0	0	0	0
1	0	0	1	0
1	0	1	0	0
1	0	1	1	1
1	1	0	0	0
1	1	0	1	1
1	1	1	0	1
1	1	1	1	1



$$MAJ(\underline{x}) =$$
$$OR($$
$$AND(x_2, x_1, x_0),$$
$$AND(x_3, x_1, x_0),$$
$$AND(x_3, x_2, x_0),$$
$$AND(x_3, x_2, x_1))$$

SWITCHING EXPRESSIONS

1. 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 to 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:

$$x_0 \qquad x_1 + x_2 x_3' \qquad 1 + 0(x + y)$$

$(x_1 + {}'x_2 +)x_3$ AND

"THIS IS A SWITCHING EXPRESSION"

ARE NOT WELL-FORMED SWITCHING EXPRESSIONS.

BOOLEAN ALGEBRAS

- IMPORTANT CLASS OF ALGEBRAS EXTENSIVELY USED FOR MANY PURPOSES
- BASIS OF THE *SWITCHING ALGEBRA* (SA) USED FOR FORMAL TREATMENT OF SWITCHING CIRCUITS:
  - TRANSFORMATION OF SWITCHING EXPRESSIONS
  - IDENTITIES FROM BA ENABLE GRAPHICAL AND TABULAR TECHNIQUES FOR MINIMIZATION OF SWITCHING EXPRESSIONS LEADING TO SIMPLER/FASTER CIRCUITS

DEFINITION OF BOOLEAN ALGEBRA

A BOOLEAN ALGEBRA IS A TUPLE $\{B, +, \cdot\}$:

- B IS A SET OF ELEMENTS;

- $+$ AND $\cdot$ ARE BINARY OPERATIONS APPLIED OVER THE ELEMENTS OF B,

SATISFYING THE FOLLOWING POSTULATES:

P1: If $a, b \in B$, then

$$\text{(i)} \quad a + b = b + a$$
$$\text{(ii)} \quad a \cdot b = b \cdot a$$

That is, $+$ and $\cdot$ are COMMUTATIVE.

P2: If $a, b, c \in B$, then

$$\text{(i)} \quad a + (b \cdot c) = (a + b) \cdot (a + c)$$
$$\text{(ii)} \quad a \cdot (b + c) = (a \cdot b) + (a \cdot c)$$

P3: THE SET B HAS TWO DISTINCT *IDENTITY ELEMENTS*, DENOTED AS 0 AND 1, SUCH THAT FOR EVERY ELEMENT IN B

$$\text{(i)} \quad 0 + a = a + 0 = a$$
$$\text{(ii)} \quad 1 \cdot a = a \cdot 1 = a$$

THE ELEMENTS 0 AND 1 ARE CALLED THE *ADDITIVE IDENTITY ELEMENT* AND THE *MULTIPLICATIVE IDENTITY ELEMENT*, respectively.

P4: FOR EVERY ELEMENT $a \in B$ THERE EXISTS AN ELEMENT a', CALLED THE *COMPLEMENT* OF a, SUCH THAT

$$\text{(i)} \quad a + a' = 1$$
$$\text{(ii)} \quad a \cdot a' = 0$$

COMMENTS

- SYMBOLS + AND · ARE NOT THE ARITHMETIC ADDITION AND MULTIPLICATION SYMBOLS

  For convenience + and · called "plus" and "times"

  Expressions $a + b$ and $a \cdot b$ called "sum" and "product,"

  + and · also called "OR" and "AND"

- THE ELEMENTS OF THE SET B ARE CALLED *CONSTANTS:* 0 AND 1 ARE CONSTANTS
- SYMBOLS REPRESENTING ARBITRARY ELEMENTS OF B ARE *VARIABLES:* a, b and c in the postulates are variables.

- *PRECEDENCE ORDERING* DEFINED ON THE OPERATORS: $\cdot$ HAS PRECEDENCE OVER $+$.

$$a + (b \cdot c) \quad \text{can be written as} \quad a + bc$$

- SYMBOLS $a, b, c, \ldots$ IN THEOREMS AND POSTULATES ARE *GENERIC VARIABLES*: THEY CAN BE SUBSTITUTED BY
  - COMPLEMENTED VARIABLES
  - EXPRESSIONS (FORMULAS)

IMPORTANT THEOREMS IN BOOLEAN ALGEBRA

Theorem 2 *PRINCIPLE OF DUALITY.*
EVERY ALGEBRAIC IDENTITY DEDUCIBLE FROM THE POSTULATES OF A BOOLEAN ALGEBRA REMAINS VALID IF

- *the operations* $+$ *and* $\cdot$ *are interchanged throughout; and*
- *the identity elements 0 and 1 are also interchanged throughout.*
- This theorem is useful because it reduces the number of different theorems that must be proven: every theorem has its dual.

Theorem 6 *IDEMPOTENT LAW.*

For every $a \in B$

$$(1)\ \ a + a = a$$
$$(2)\ \ a \cdot a = a$$

Proof:

		by
(1): $a + a$	$= (a + a) \cdot 1$	P3 (ii)
	$= (a + a) \cdot (a + a')$	P4 (i)
	$= (a + (a \cdot a'))$	P2 (i)
	$= a + 0$	P4 (ii)
	$= a$	P3 (i)
(2):		duality

■

Theorem 8 *ABSORPTION LAW. For every pair of elements* $a, b \in B$,

$$(1)\quad a + a \cdot b = a$$
$$(2)\quad a \cdot (a + b) = a$$

Proof:

			by
(1):	$a + ab$	$= a \cdot 1 + ab$	P3 (ii)
		$= a(1 + b)$	P2 (ii)
		$= a(b + 1)$	P1 (i)
		$= a \cdot 1$	Theorem 4 (1)
		$= a$	P3 (ii)
(2)			duality

■

OTHER EXAMPLES OF BOOLEAN ALGEBRAS

ALGEBRA OF SETS. The elements of B: set of all subsets of a set S

($P(S)$). The operations: set-union ($\cup$) and set-intersection ($\cap$).

$$M = (P(S), \cup, \cap)$$

ALGEBRA OF LOGIC (PROPOSITIONAL CALCULUS).

The elements B: T and F (true and false). The operations: LOGICAL AND and LOGICAL OR.

This algebra is isomorphic with the switching algebra.

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 SFs BY SWITCHING EXPRESSIONS

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

$x_2 x_1 x_0$	f
000	0
001	0
010	1
011	1
100	1
101	1
110	1
111	1

SWITCHING EXPRESSIONS FOR BASIC FUNTIONS

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

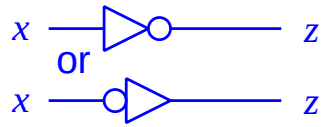
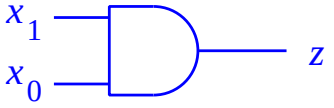
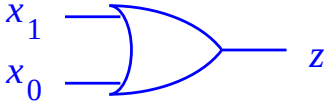
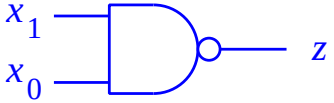
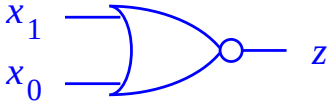
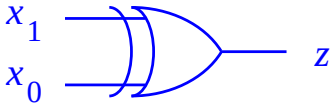
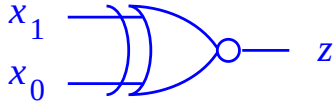
Gate type	Symbol	Switching expression
NOT		$z = x'$
AND		$z = x_1 x_0$
OR		$z = x_1 + x_0$
NAND		$z = (x_1 x_0)'$
NOR		$z = (x_1 + x_0)'$
XOR		$z = x_1 x_0' + x_1' x_0$ $= x_1 \oplus x_0$
XNOR		$z = x_1' x_0' + x_1 x_0$

Figure 2.8: Gate symbols

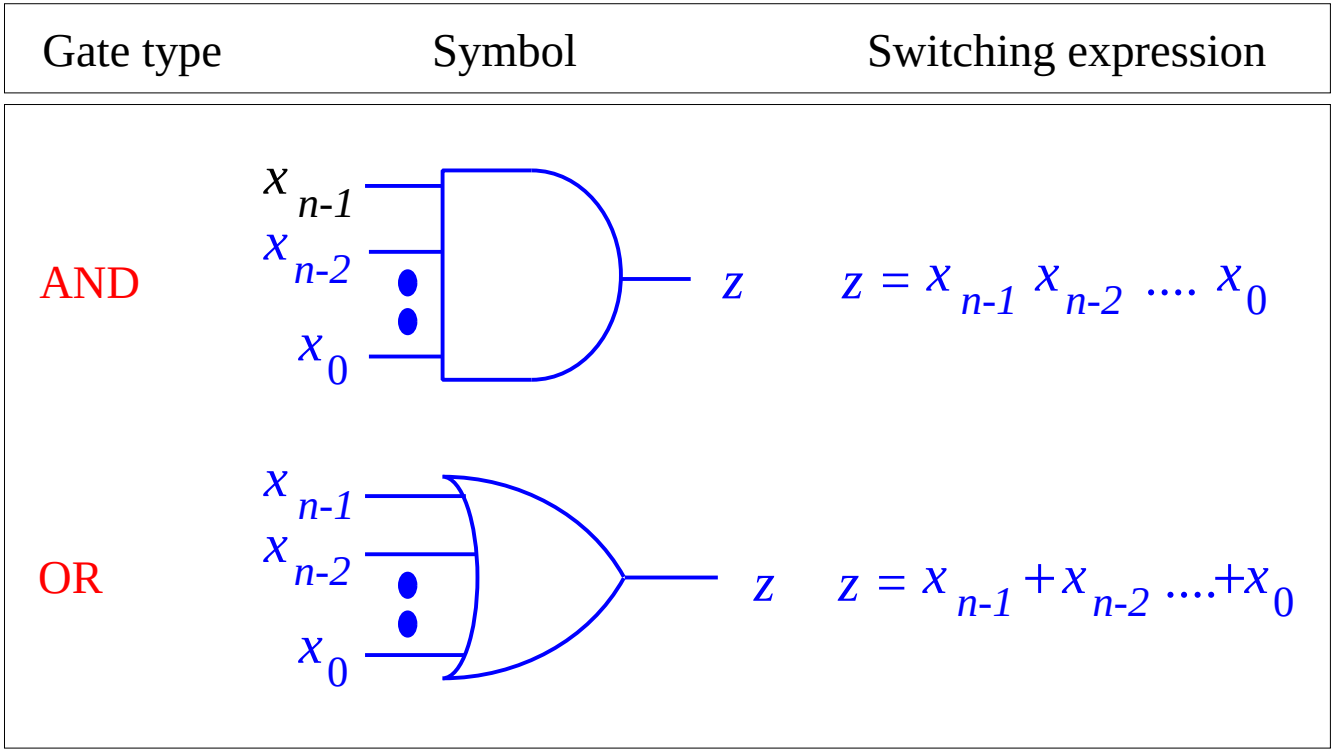


Figure 2.9: n-input AND and OR GATE SYMBOLS

EQUIVALENT SWITCHING EXPRESSIONS

W and Z SWITCHING EXPRESSIONS ARE EQUIVALENT

$$W = x_1x_0 + x_1'$$
$$Z = x_1' + x_0$$

CORRESPONDING SWITCHING FUNCTIONS ARE

x_1x_0	W	Z
00	1	1
01	1	1
10	0	0
11	1	1

EQUIVALENCE CLASSES

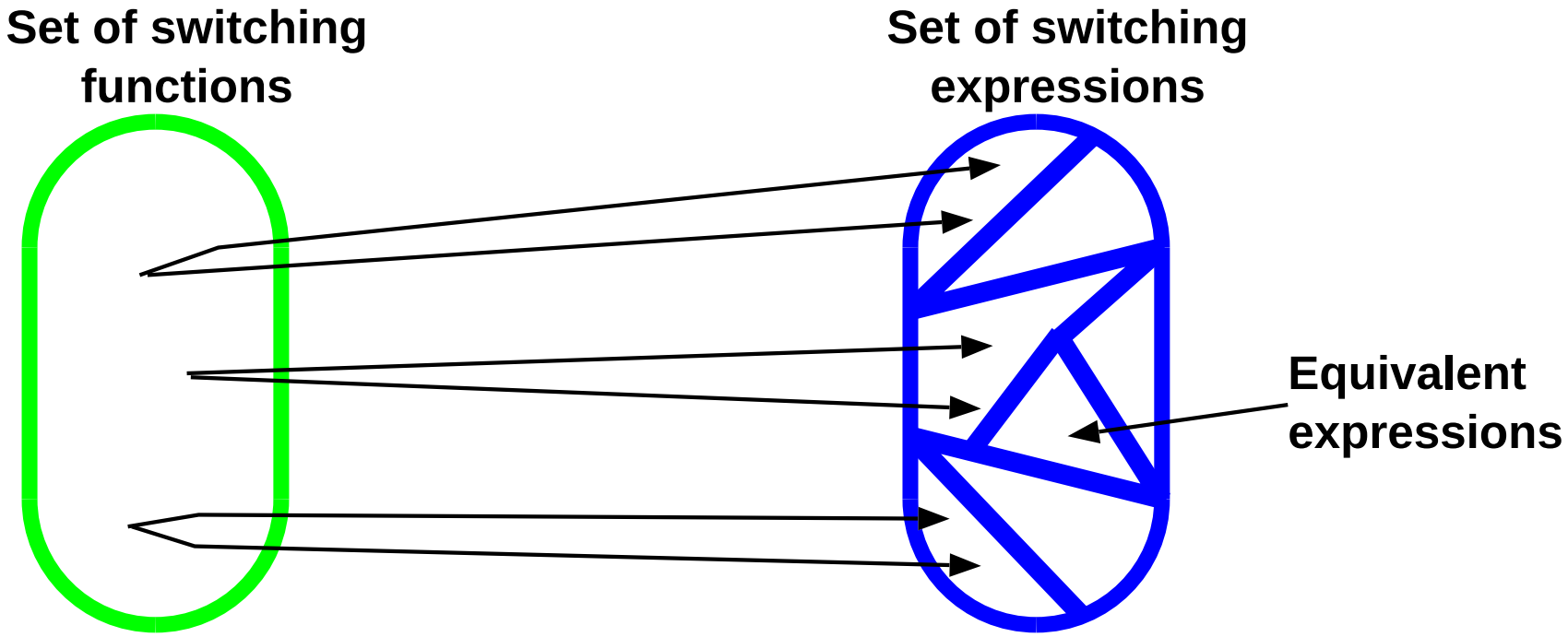


Figure 2.10: CORRESPONDENCE AMONG SFs AND SEs

ALGEBRAIC METHOD OF OBTAINING EQUIVALENT EXPRESSIONS

- MAIN 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

CS MSIA EXAMPLES OF IDENTITIES

— PROOFS

10: (i) $a + a'b = a + b$ SIMPLIFICATION

(ii) $a(a'+b) = ab$

SHOW (i): $a + a'b = (a+a')(a+b)$ — BY DISTRIBUTIVITY P2(i)

$= 1 \cdot (a+b)$ — BY COMPLEMENT P4(i)

$= a + b$ — BY IDENTITY P3(i)

11 (DEMORGAN'S LAW)

(i) $(a+b)' = a'b'$

(ii) $(ab)' = a' + b'$

SHOW (i): FIRST PROVE $(a+b) = (a'b')'$ (SAME AS (i)!)

BY P4: $(a+b) + a'b' = 1$

$(a+b)(a'b') = 0$

$(a+b) + a'b' = [(a+b) + a'][(a+b) + b']$ P2(i)

$= \ldots$

$= (b+1)(a+1) = 1 \cdot 1 = 1$ ✓

$(a+b)(a'b') = aa'b' + ba'b' = 0 + 0 = 0$ ✓

EXAMPLE

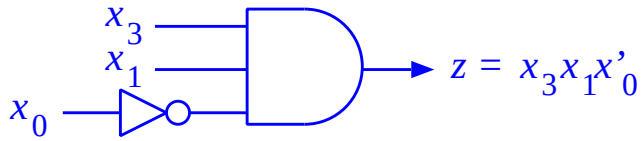
SHOW THAT E_1 AND E_2 ARE EQUIVALENT:

$$E_1(x_2, x_1, x_0) = x_2x_1 + x_2x_1' + x_2x_0$$
$$E_2(x_2, x_1, x_0) = x_2$$

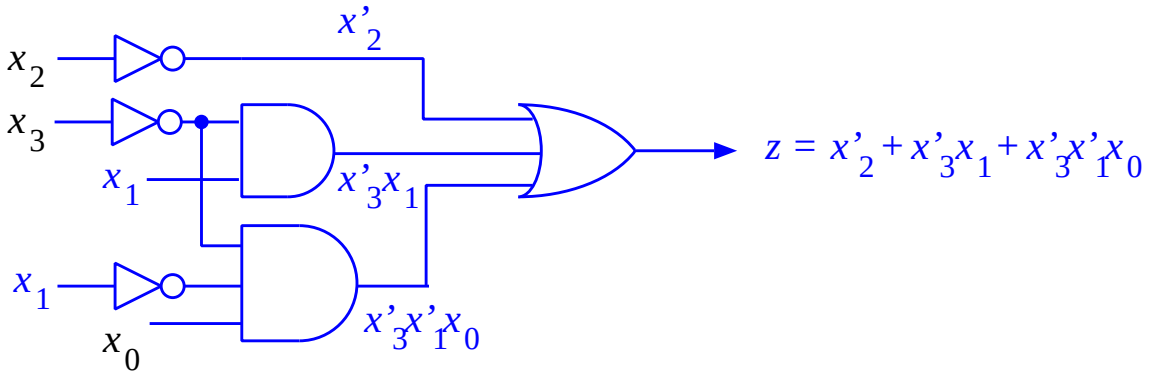
$$\begin{aligned}
x_2x_1 + x_2x_1' + x_2x_0 &= x_2(x_1 + x_1') + x_2x_0 && \text{using } ab + ac = a(b + c)\\
&= x_2 \cdot 1 + x_2x_0 && \text{using } a + a' = 1\\
&= x_2(1 + x_0) && \text{using } ab + ac = a(b + c)\\
&= x_2 \cdot 1 && \text{using } 1 + a = 1\\
&= x_2 && \text{using } a \cdot 1 = a
\end{aligned}$$

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 (SP)	$x_2' + x_3x_1' + x_3'x_1'x_0$



(a)



(b)

Figure 2.11: 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 (product of all n variables), j INTEGER

EXAMPLE: MINTERM $x_3x_2'x_1'x_0$ DENOTED m_9
BECAUSE $1001 = 9$

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

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

EXAMPLE: $m_{11} = x_3x_2'x_1x_0$
– HAS VALUE 1 ONLY FOR $\underline{a} = (1, 0, 1, 1)$

MINTERM FUNCTIONS

$x_2x_1x_0$	m_0 $x_2'x_1'x_0'$	m_1 $x_2'x_1'x_0$	m_2 $x_2'x_1x_0'$	m_3 $x_2'x_1x_0$	m_4 $x_2x_1'x_0'$	m_5 $x_2x_1'x_0$	m_6 $x_2x_1x_0'$	m_7 $x_2x_1x_0$
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

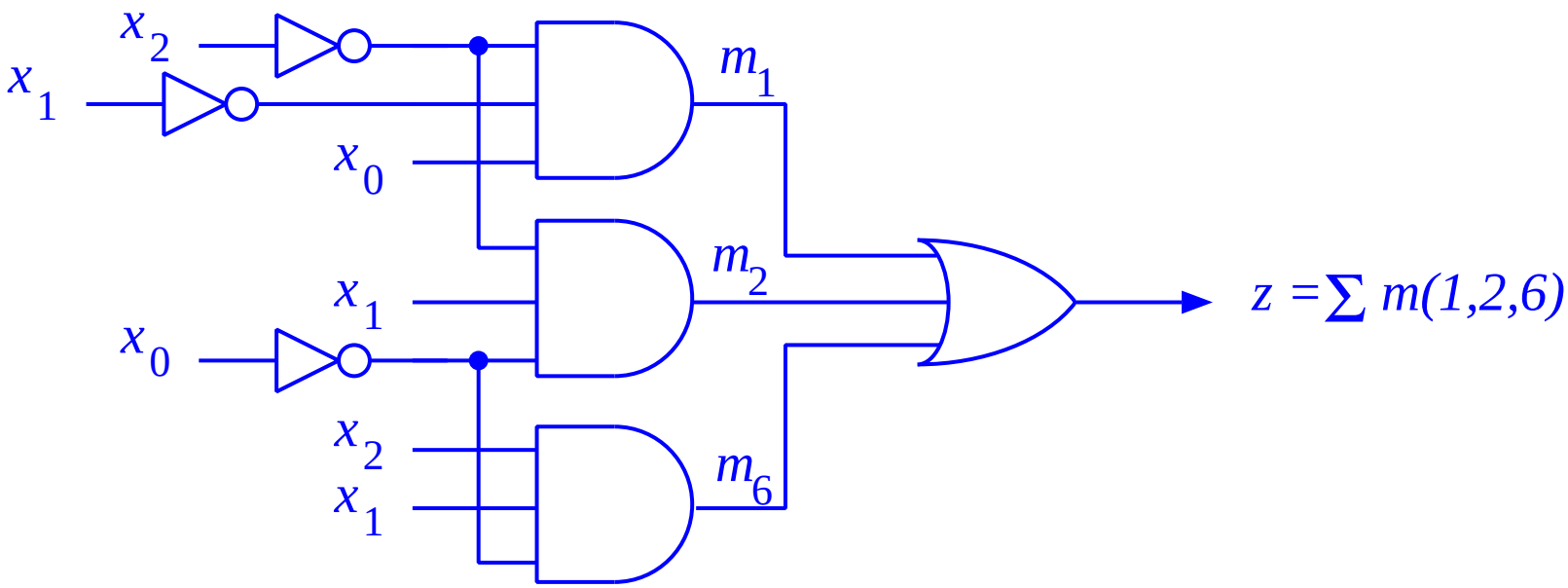


Figure 2.12: GATE NETWORK CORRESPONDING TO $E(x_2, x_1, x_0) = \sum m(1, 2, 6)$.

Example 2.12: TABLE → 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

$$E = \Sigma\, m(2,3,5) = x_2' x_1 x_0' + x_2' x_1 x_0 + x_2 x_1' x_0$$

CONVERSION TO SUM OF MINTERMS

1. CONVERT TO EQUIVALENT SUM OF PRODUCTS

$$\begin{aligned} E(x_2, x_1, x_0) &= (x_2 x_1)' x_0 \\ &= (x_2' + x_1') x_0 \\ &= x_2' x_0 + x_1' x_0 \end{aligned}$$

2. CONVERT PRODUCT TERMS TO MINTERMS

$$\begin{aligned} E(x_2, x_1, x_0) &= x_2' x_0 + x_1' x_0 \\ &= x_2' x_0 (x_1 + x_1') + x_1' x_0 (x_2 + x_2') \\ &= x_2' x_1 x_0 + x_2' x_1' x_0 + x_2 x_1' x_0 + x_2' x_1' x_0 \end{aligned}$$

3. ELIMINATE REPEATED MINTERMS

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

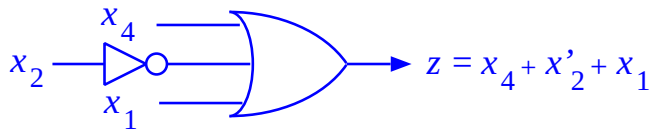
Example 2.13: CONVERSION TO SUM OF MINTERMS

$$
\begin{aligned}
E(x_2, x_1, x_0) &= x_2x_1' + x_2x_0' + x_1x_0' \\
&= x_2x_1'(x_0 + x_0') + x_2x_0'(x_1 + x_1') + x_1x_0'(x_2 + x_2') \\
&= x_2x_1'x_0 + x_2x_1'x_0' + x_2x_0'x_1 + x_2x_0'x_1' + x_1x_0'x_2 + x_1x_0'x_2' \\
&= x_2'x_1x_0' + x_2x_1'x_0' + x_2x_1'x_0 + x_2x_1x_0' \\
&= \Sigma\, m(2, 4, 5, 6)
\end{aligned}
$$

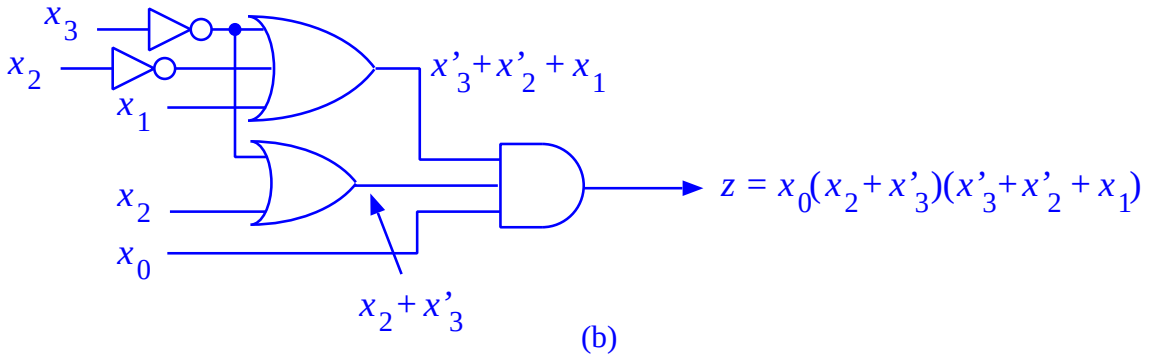
PRODUCT OF SUMS AND PRODUCT OF MAXTERMS

SUM TERMS $x_0,\ x_2 + x_1,\ x_3 + x_1 + x_0'$

PRODUCT OF SUMS $(x_2' + x_3 + x_1')(x_3' + x_1)x_0$



(a)



(b)

Figure 2.13: 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 , (sum of all n variables), j INTEGER

EXAMPLE: MAXTERM $x_3 + x_2' + x_1 + x_0'$ DENOTED M_5
BECAUSE 0101 = 5

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

EXAMPLE: $M_5 = x_3 + x_2' + x_1 + x_0'$
– HAS VALUE 0 ONLY FOR ASSIGNMENT 0101

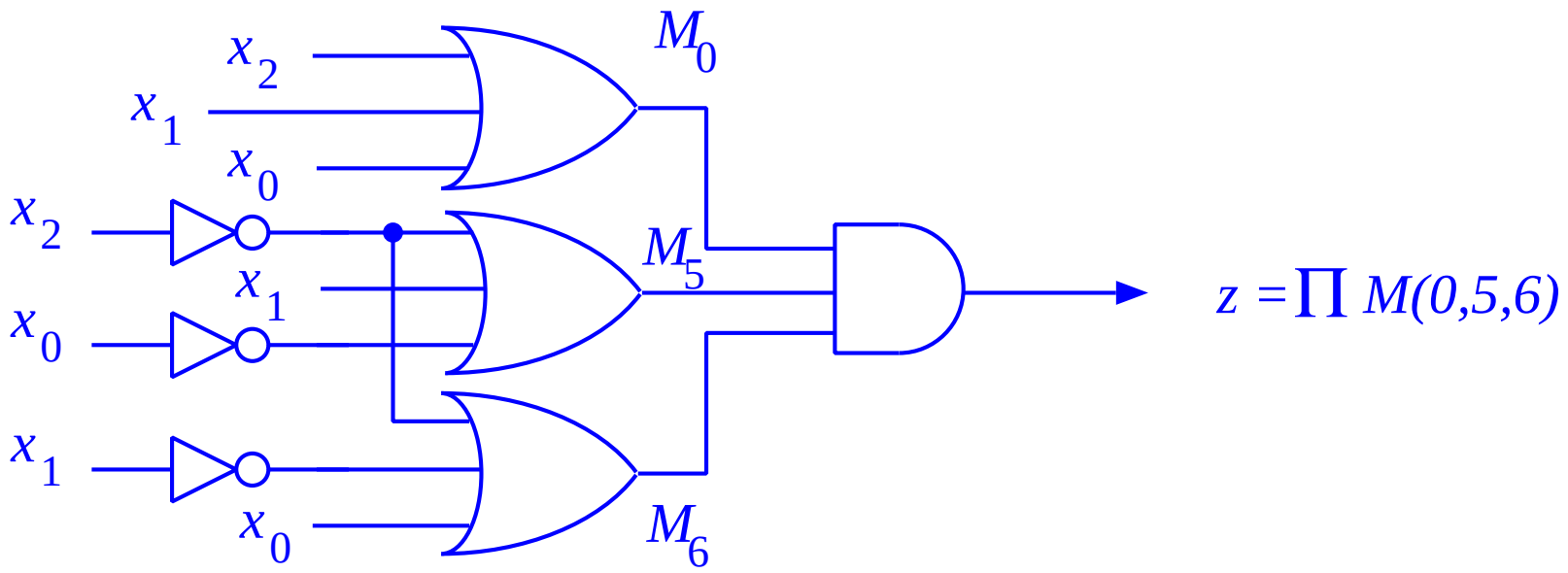


Figure 2.14: GATE NETWORK CORRESPONDING TO $E(x_2, x_1, x_0) = \prod M(0, 5, 6)$.

Example 2.15: TABLE → 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

$$\Sigma\, m(\{j \mid f(j) = 1\}) = \Pi\, M(\{j \mid f(j) = 0\})$$

EXAMPLE:

m-NOTATION:

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

M-NOTATION:

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

CSM51A OBTAINING STANDARD EXPRESSIONS

(SOP, POS, CSP, CPS)

(i) GETTING SOP & POS

$E(a,b,c) = [(a+b')c + a'd]'$ — NOT A STANDARD FORM

TRANSFORM TO SOP USING IDENTITIES

$E(a,b,c) = [(a+b')c]' \cdot [a'd]'$

$= [(a+b')'' + c'](a+d')$

$= (a'b + c')(a+d')$

POS

$= ac' + a'bd' + c'd'$ ← SOP

$E(a,b,c) = (a'+c')(b+c')(a+d')$ ← POS

— USED DISTRIBUTIVITY $xy+z = (x+z)(y+z)$

CS MSIA

(ii) GETTING CANONICAL FORMS

SOP → CSP

$$ac' + a'bd' + c'd' = a(b+b')c'(d+d') + a'b(c+c')d' + (a+a')(b+b')c'd'$$

$$= abc'd + abc'd' + ab'c'd + ab'c'd' + a'bcd' + a'bc'd' + abc'd' + ab'c'd' + a'bc'd' + a'b'c'd'$$

$$= m_{13} + m_{12} + m_9 + m_8 + m_6 + m_4 + \cancel{m_{12}} + \cancel{m_8} + \cancel{m_4} + m_0$$

$$= \sum m(0, 4, 6, 8, 9, 12, 13)$$ ✓✓

CSM 51A

POS $\Rightarrow$ CPS

$(a'+c')(b+c')(a+d') = (a'+bb'+c')(aa'+b+c')(a+bb'+d')$

$= (a'+b+c')(a'+b'+c')$
$(a+b+c')(a'+b+c')$
$(a+b+d')(a+b'+d')$

$= (a'+b+c'+dd')(a'+b'+c'+dd')$
$(a+b+c'+dd')(a+b+cc'+d')$
$(a+b'+cc'+d')$

$= (a'+b+c'+d)(a'+b+c'+d')$
$(a'+b'+c'+d)(a'+b'+c'+d')$
$(a+b+c'+d)(a+b+c'+d')$
$(a+b+c+d')(a+b+c'+d')$
$(a+b'+c+d')(a+b'+c'+d')$

$= M_{10} \cdot M_{11} \cdot M_{14} \cdot M_{15} \cdot M_2 \cdot M_3 \cdot M_1 \cdot M_3 \cdot M_5 \cdot M_7$

$= \prod M(1, 2, 3, 5, 7, 10, 11, 14, 15)$

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:

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

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

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

x_1x_0	y_1y_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:

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

CS MSIA

EXAMPLES OF BINARY SPECIFICATIONS

1. FULL ADDER (FA)

(ADDEND, AUGEND)

INPUTS: $x_i, y_i \in \{0,1\}$ $c_i \in \{0,1\}$ (CARRY-IN)

OUTPUTS: $s_i \in \{0,1\}$ SUM $c_{i+1} \in \{0,1\}$ (CARRY-OUT)

FUNCTION:

$$s_i = (x_i + y_i + c_i) \bmod 2$$

$$c_{i+1} = \begin{cases} 1 & \text{IF } x_i + y_i + c_i \geq 2 \\ 0 & \text{OTHERWISE} \end{cases}$$

} SOLUTIONS OF $x_i + y_i + c_i = 2c_{i+1} + s_i$

SWITCHING FUNCTIONS

$$s_i = f_0(x_i, y_i, c_i) \qquad c_{i+1} = f_1(x_i, y_i, c_i)$$

j	Σ	x_i	y_i	c_i	c_{i+1}	s_i
0	0	0	0	0	0	0
1	1	0	0	1	0	1
2	1	0	1	0	0	1
3	2	0	1	1	1	0
4	1	1	0	0	0	1
5	2	1	0	1	1	0
6	2	1	1	0	1	0
7	3	1	1	1	1	1

$$s_i = \sum m(1,2,4,7)$$

$$c_{i+1} = \sum m(3,5,6,7)$$

CS MSIA EXAMPLES (CONT.)

2, CHARACTER MATCHING

INPUTS $x, y \in \{A, B, \ldots, Z, a, b, \ldots z, 0, 1, \ldots 9, +, !, \ldots\}$

OUTPUT $z \in \{0, 1\}$

FUNCTION $z = \begin{cases} 1 & \text{IF } x = y \\ 0 & \text{IF } x \neq y \end{cases}$

$x \rightarrow \underline{x} = (x_7, x_6, \ldots, x_0)$

$\underline{y} = (y_7, y_6, \ldots, y_0)$

16 VARIABLES $\rightarrow$ TABLE NOT PRACTICAL

OBSERVE: $x = y$ IFF $x_i = y_i, \forall i$

AN SE FOLLOWS DIRECTLY

$$z = (x_7 y_7 + x_7' y_7')(x_6 y_6 + x_6' y_6') \cdots (x_0 y_0 + x_0' y_0')$$

CS M51A EXAMPLES (CONT.)

3. 2-BY-2 MULTIPLIER (2-BIT × 2-BIT)

(i) HL SPEC INPUTS $x, y \in \{0, 1, 2, 3\}$ MULTIPLICAND x, MULTIPLIER y

OUTPUT/FUNCTION $p = x \times y \in \{0, 1, 2, 3, 4, 6, 9\}$ PRODUCT

x \ y	0	1	2	3
0	0	0	0	0
1	0	1	2	3
2	0	2	4	6
3	0	3	6	9

p

(ii) BL SPEC - ASSUME BINARY CODE

$x = (x_1, x_0) = 2x_1 + x_0$ $\quad x_i, y_i \in \{0, 1\}$

$y = (y_1, y_0) = 2y_1 + y_0$

$p = (p_3 p_2 p_1 p_0) = \sum_{i=0}^{3} p_i 2^i$ $\quad p_i \in \{0, 1\}$

$x_1 x_0$ \ $y_1 y_0$	00	01	10	11
00	0000	0000	0000	0000
01	0000	0001	0010	0011
10	0000	0010	0100	0110
11	0000	0011	0110	1001

$p_3 p_2 p_1 p_0$

	00	01	10	11
00	m_0	m_1	m_2	m_3
01	m_4	m_5	m_6	m_7
10	m_8	m_9	m_{10}	m_{11}
11	m_{12}	m_{13}	m_{14}	m_{15}

$m_i (x_1, x_0, y_1, y_0)$

$p_3 = m_{15}$

$p_2 = \sum m(10, 11, 14)$

$p_1 = \sum m(6, 7, 9, 11, 13, 14)$

$p_0 = \sum m(5, 7, 13, 15) = x_0 y_0$

DETERMINE THE COST

NOT # AND # OR