PROGRAMMABLE MODULES

- SPECIFICATION OF PROGRAMMABLE COMBINATIONAL AND SEQUENTIAL MODULES

  1. PSA
  2. ROM
  3. FPGA

- THE WAY THE MODULES ARE PROGRAMMED
- NETWORKS OF PROGRAMMABLE MODULES
- EXAMPLES OF USES

PROGRAMMABLE SEQUENTIAL ARRAYS (PSA)

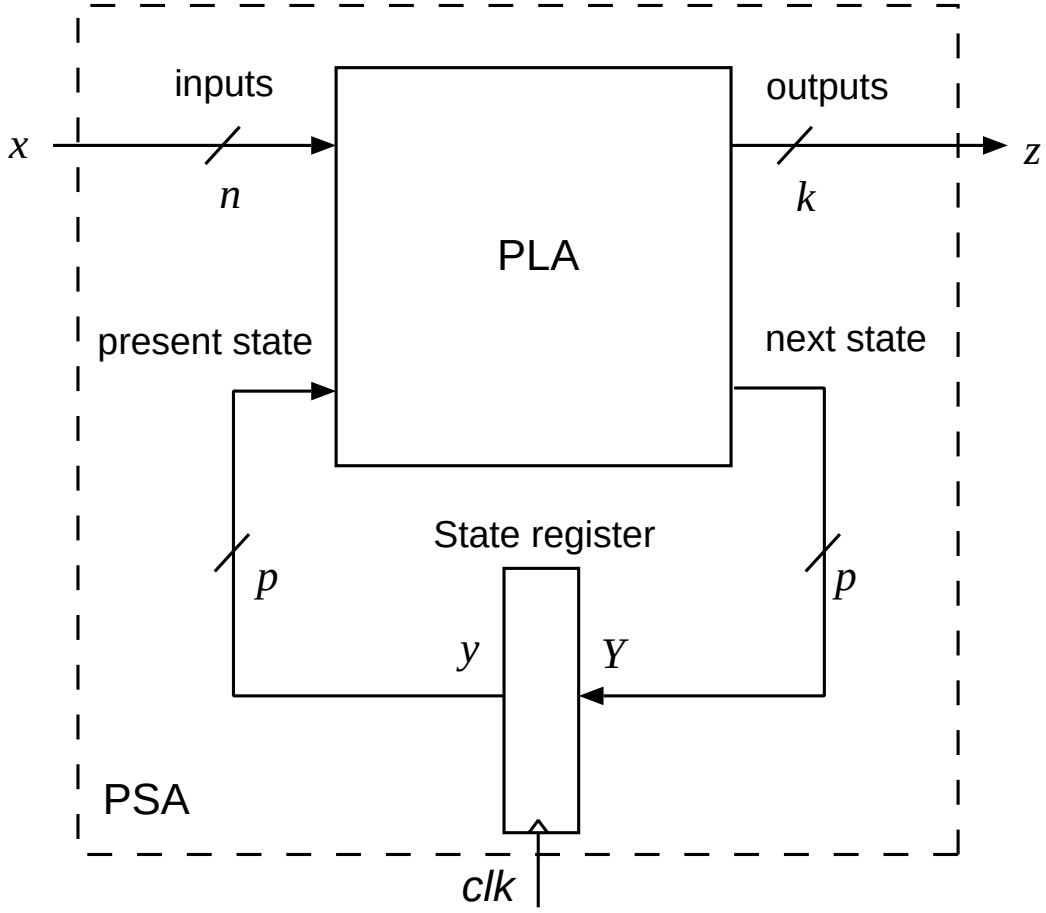


Figure 12.1: PROGRAMMABLE SEQUENTIAL ARRAY (PSA).

Example 12.1: IMPLEMENTATION OF SEQUENTIAL SYSTEMS USING PSAs

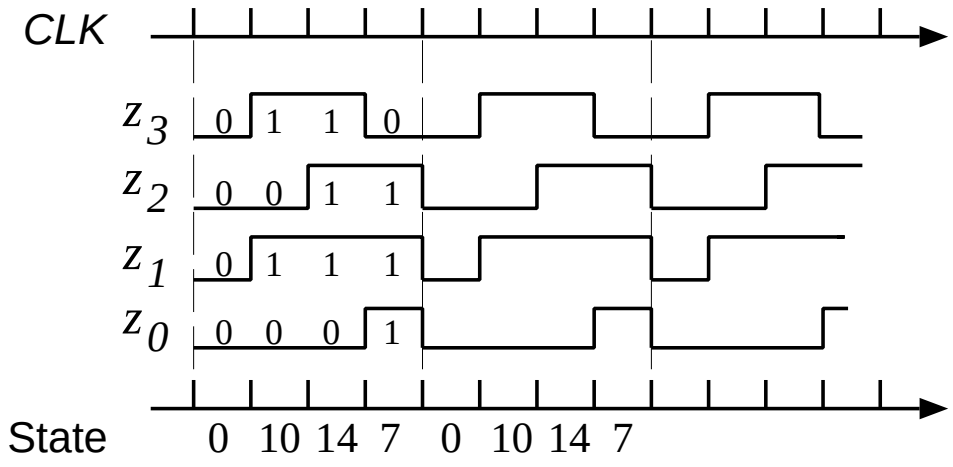
- SEQUENCE GENERATOR

  INPUTS: $x \in \{0, 1\}$
  OUTPUTS: $z \in \{0, 1, 3, 6, 7, 10, 14\}$

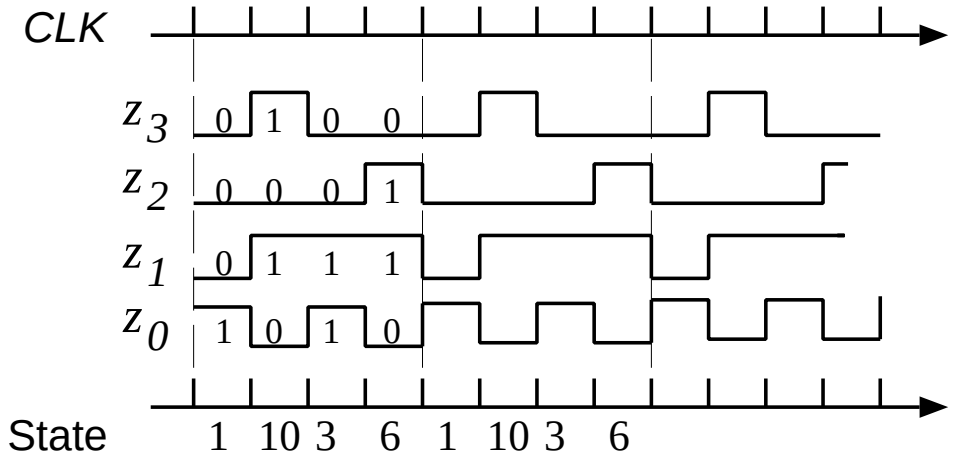
  FUNCTION: The transition and output functions

$$x = 0 \,:\, z = 0 \to 10 \to 14 \to 7 \to 0 \cdots$$
$$x = 1 \,:\, z = 1 \to 10 \to 3 \to 6 \to 1 \cdots$$

$$x = 0 \,:\, z = 0000 \to 1010 \to 1110 \to 0111 \to 0000 \cdots$$
$$x = 1 \,:\, z = 0001 \to 1010 \to 0011 \to 0110 \to 0001 \cdots$$



(a) Case x = 0



(b) Case x = 1

Figure 12.2: TIMING SEQUENCES IN Example 12.1.

Example 12.1 (cont.)

y	$k=0$	$k=1$	
0	10	–	k
1	–	10	k
3	–	6	x
6	0	1	k
7	0	1	k
10	14	3	k
14	7	–	x
	Y		K

$y \in \{2, 4, 5, 8, 9, 11, 12, 13, 15\}$ – don't care states

cont.

$$
\begin{aligned}
K &= xy_3y_2 + xy_3'y_2'y_1 + ky_1' + ky_3'y_2 + ky_3y_2' \\
Y_3 &= y_1' + y_2'k' \\
Y_2 &= y_3'y_2'y_1 + y_3k' \\
Y_1 &= y_2' + y_3 \\
Y_0 &= y_3k + y_2k + y_3y_2
\end{aligned}
$$

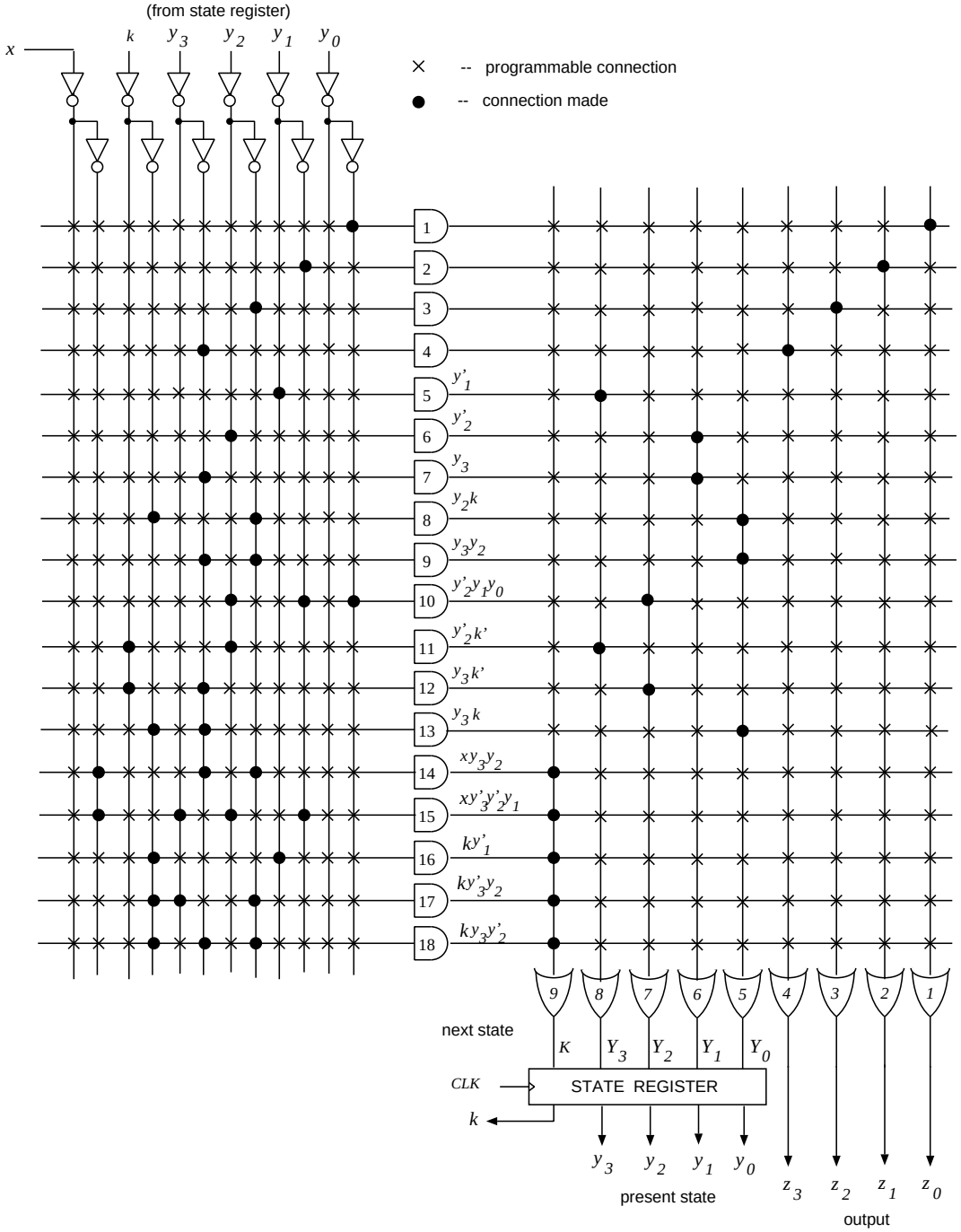


Figure 12.3: PSA IMPLEMENTATION IN Example 12.1.

READ-ONLY MEMORIES (ROM)

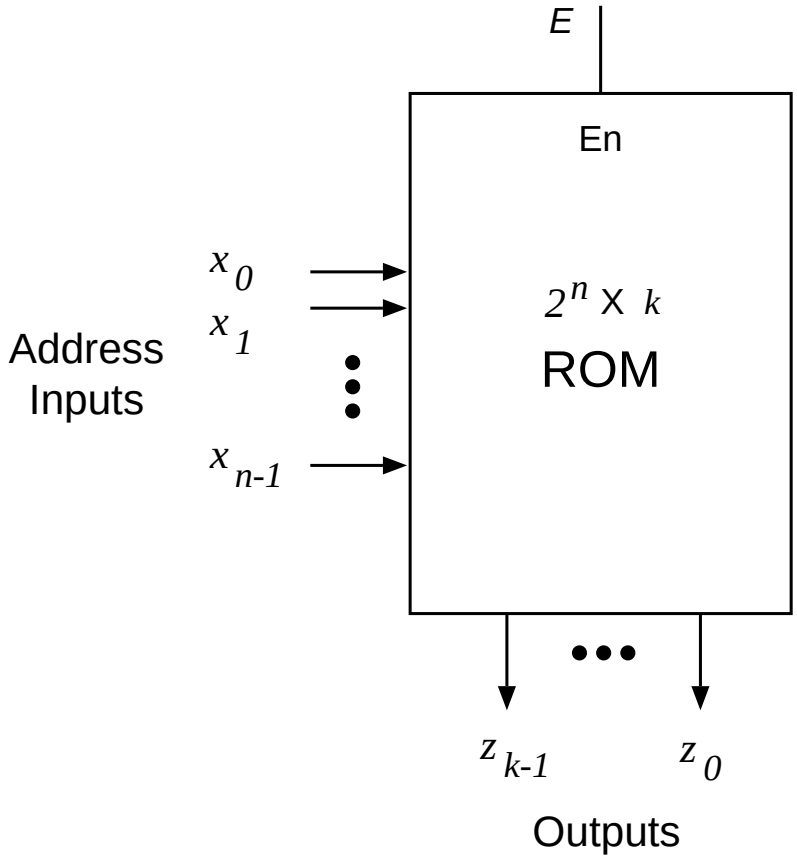


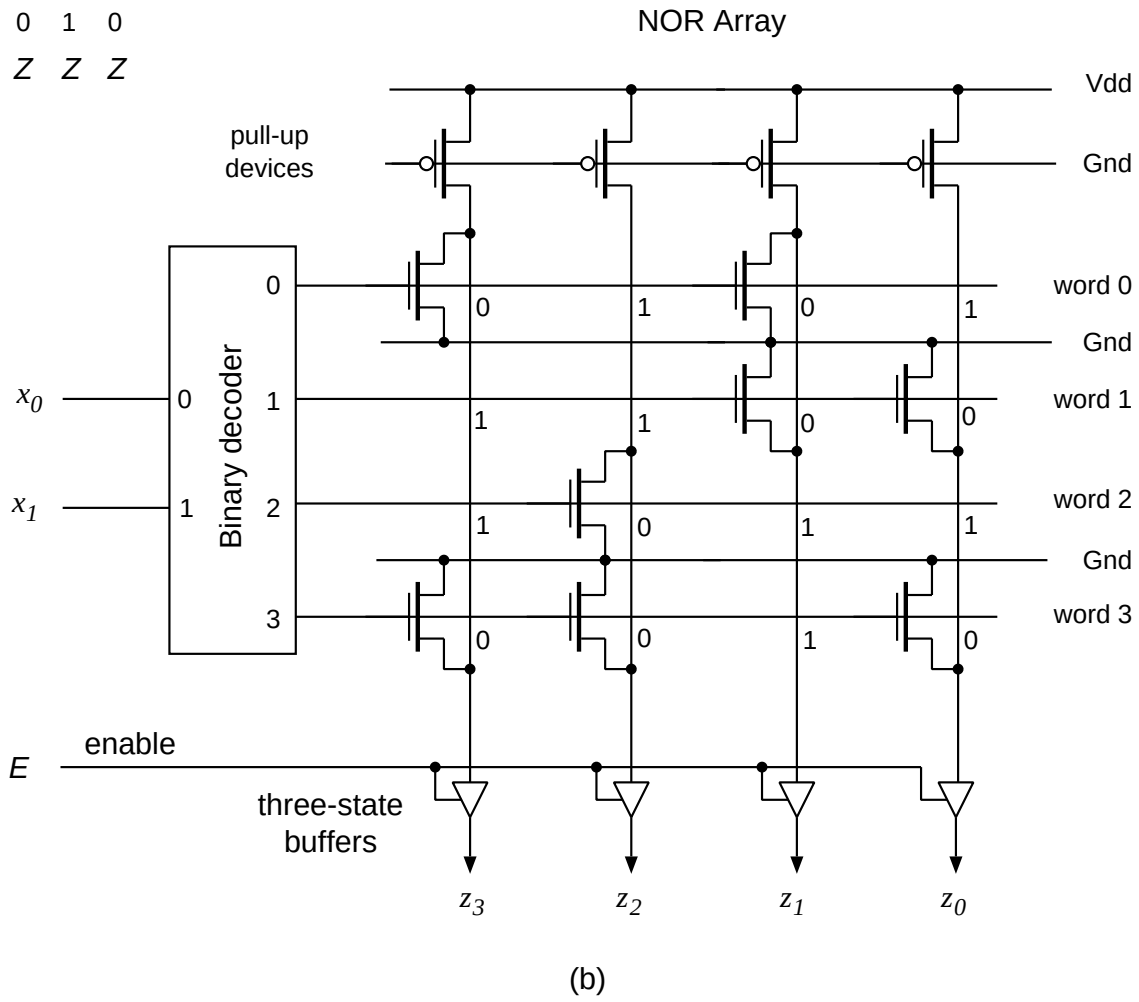
Figure 12.4: READ-ONLY MEMORY (ROM)

EXAMPLE 12.2

Address	Contents
$\underline{x}$	$\underline{z}$
000	1011
001	1101
010	**0111**
011	1000
100	0000
101	1111
110	1111
111	1011

E	x_1	x_0	z_3	z_2	z_1	z_0
1	0	0	0	1	0	1
1	0	1	1	1	0	0
1	1	0	1	0	1	1
1	1	1	0	0	1	0
0	-	-	*Z*	*Z*	*Z*	*Z*

(a)



(b)

Figure 12.5: MOS IMPLEMENTATION OF A 4×4 READ-ONLY MEMORY: a) THE FUNCTION; b) THE CIRCUIT.

IMPLEMENTATION OF SWITCHING FUNCTIONS USING ROMS

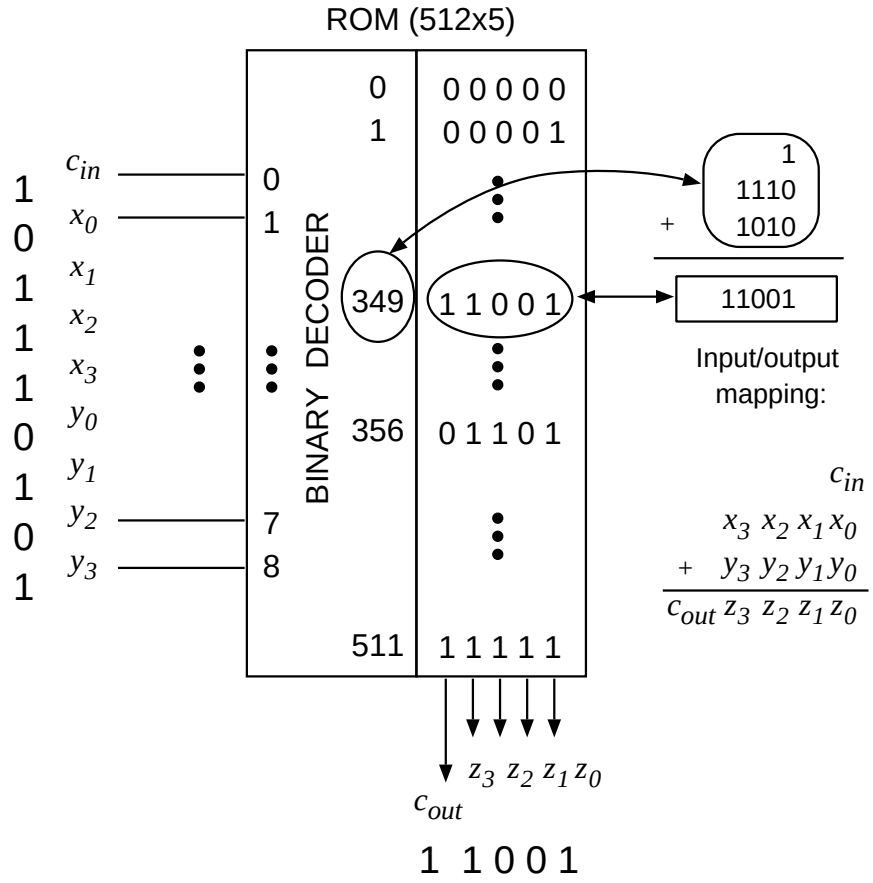


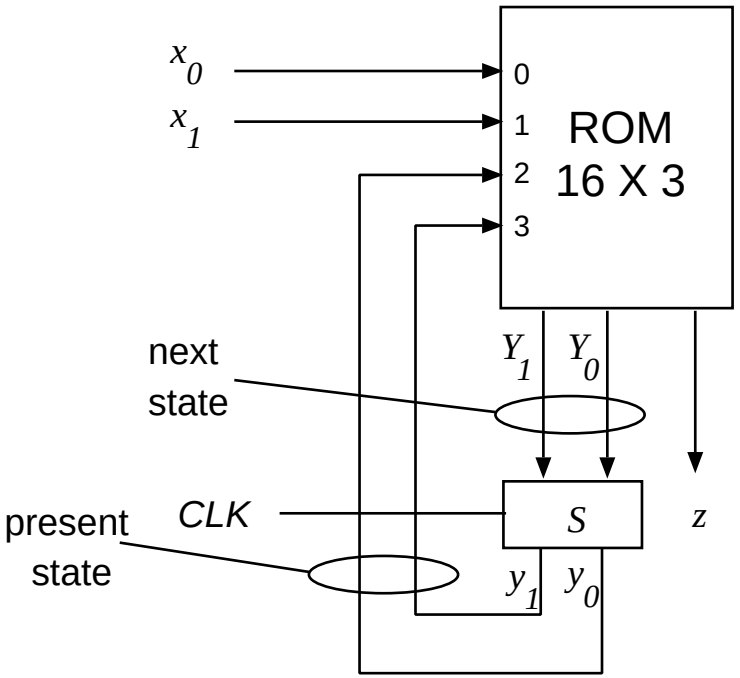
Figure 12.6: ROM-BASED IMPLEMENTATION OF A 4-BIT ADDER.

IMPLEMENTATION OF SEQUENTIAL SYSTEMS USING ROMS

INPUTS: $\underline{x} = (x_1, x_0), \quad x_i \in \{0, 1\}$
OUTPUTS: $z \in \{0, 1, \}$
STATE: $\underline{y} = (y_1, y_0), \quad y_i \in \{0, 1, \}$

FUNCTION: The transition and output function

PS	x_1x_0		
y_1y_0	01	10	11
00	01,0	10,1	10,0
01	00,0	11,1	11,0
10	11,0	10,0	00,1
11	10,0	00,0	11,1
	$Y_1Y_0,\ z$		
	NS, Output		



(a)

ROM address $y_1 y_0 x_1 x_0$	ROM contents $Y_1 Y_0$ z
0000	- - -
0001	0 1 0
0010	1 0 1
0011	1 0 0
0100	- - -
0101	0 0 0
0110	1 1 1
0111	1 1 0
1000	- - -
1001	1 1 0
1010	1 0 0
1011	0 0 1
1100	- - -
1101	1 0 0
1110	0 0 0
1111	1 1 1

next state output

(b)

Figure 12.7: ROM-based implementation of a sequential system: a) network; b) ROM contents.

TYPES OF ROM MODULES

- MASK-PROGRAMMED ROM
- FIELD-PROGRAMMABLE ROM (PROM)
- ERASABLE ROM (EPROM)
- ELECTRICALLY ERASABLE ROM or EEPROM

NETWORKS OF PROGRAMMABLE MODULES

$$f_1(x_4, x_3, x_2, x_1, x_0) = \textit{one-set}(0,3,11,12,16,23,27)$$
$$f_0(x_4, x_3, x_2, x_1, x_0) = \textit{one-set}(5,7,19,21,31)$$

ROM MODULE: 8×2

$$\underline{x} = (\underline{x}^{(0)}, \underline{x}^{(1)})$$
$$\underline{x}^{(0)} = (x_4, x_3)$$
$$\underline{x}^{(1)} = (x_2, x_1, x_0)$$

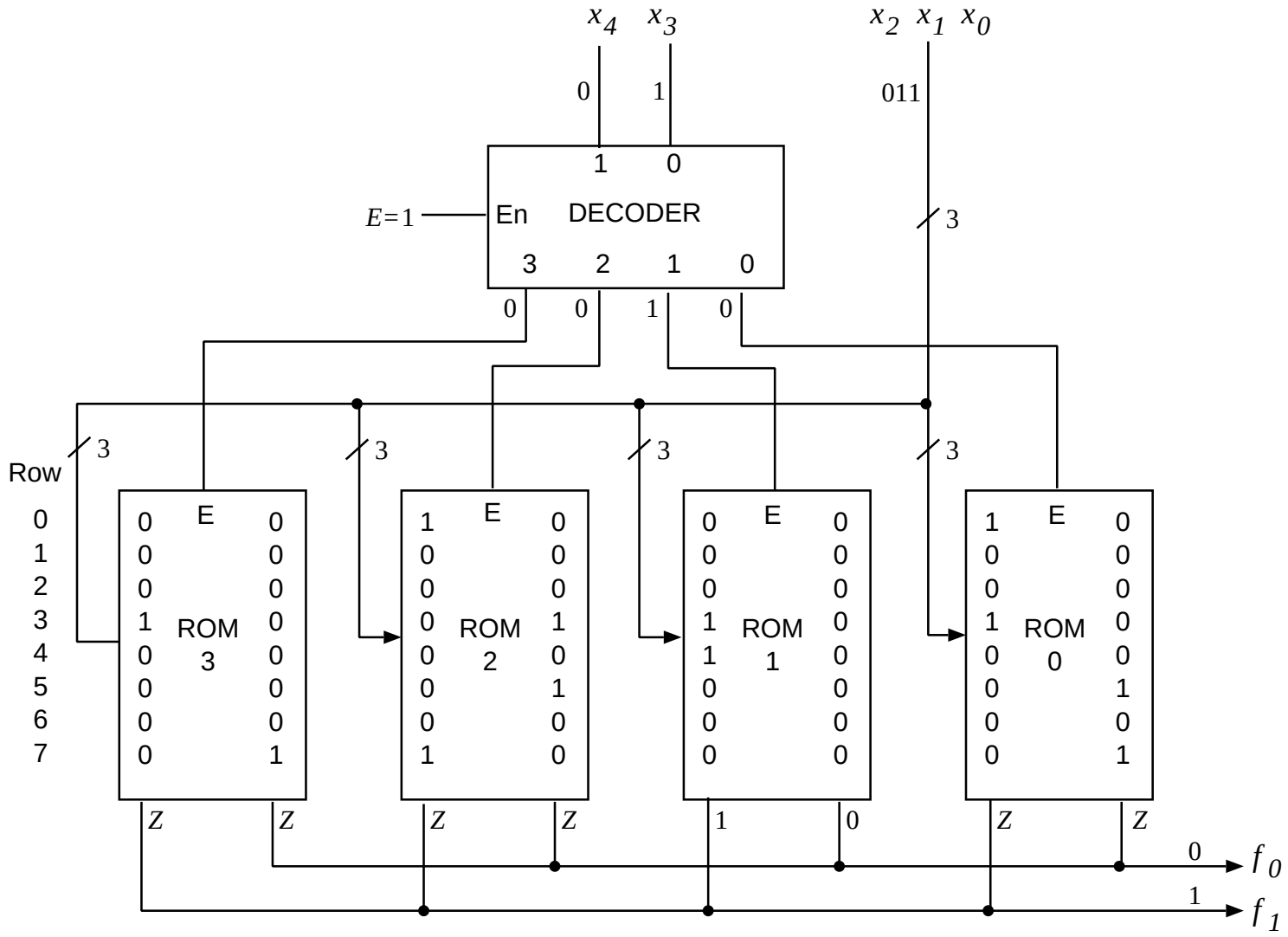


Figure 12.8: ROM-BASED NETWORK FOR THE IMPLEMENTATION OF TWO FUNCTIONS.

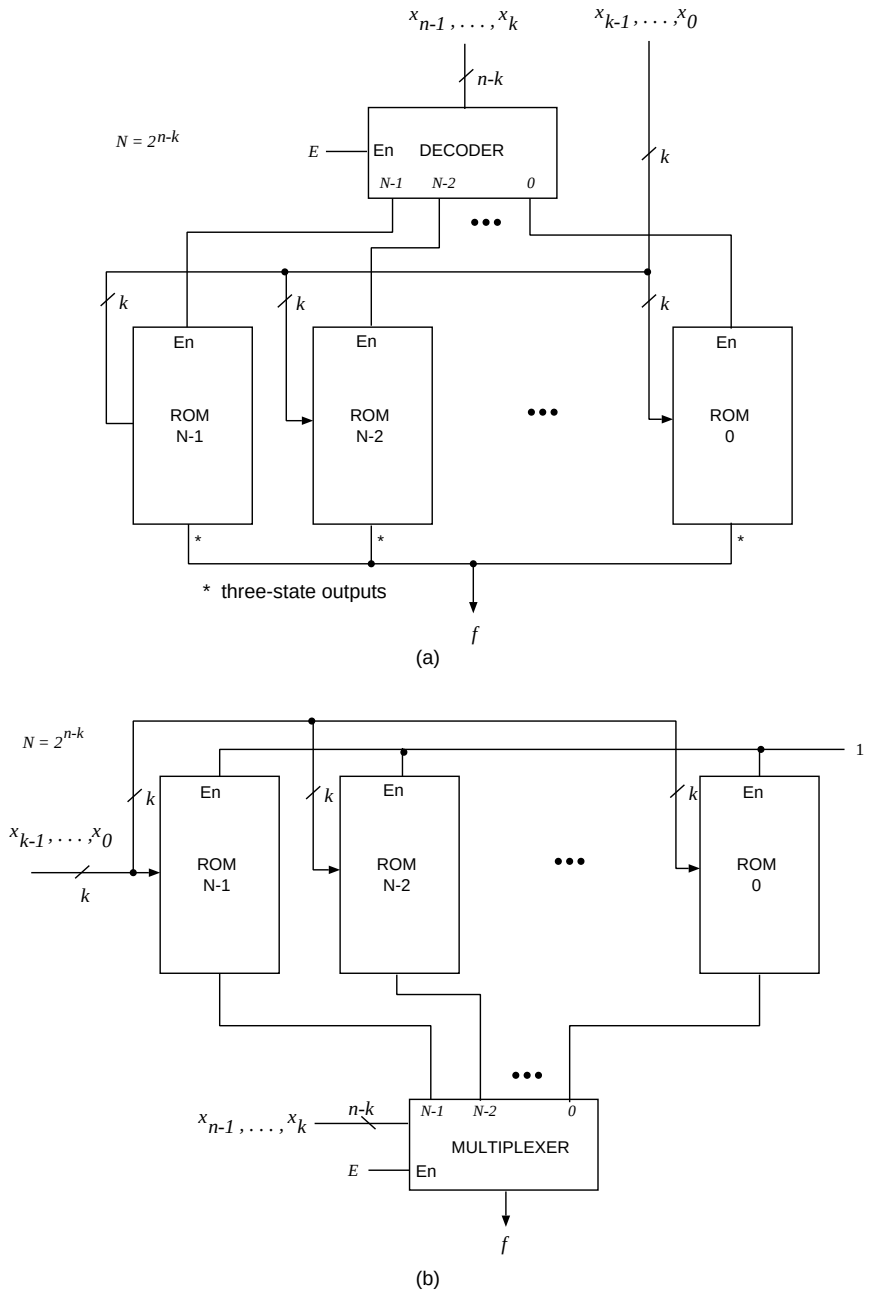


Figure 12.9: IMPLEMENTATIONS OF FUNCTIONS WITH n VARIABLES: a) ROMS AND DECODER; b) ROMS AND MULTIPLEXER

LARGE NUMBER OF SFs

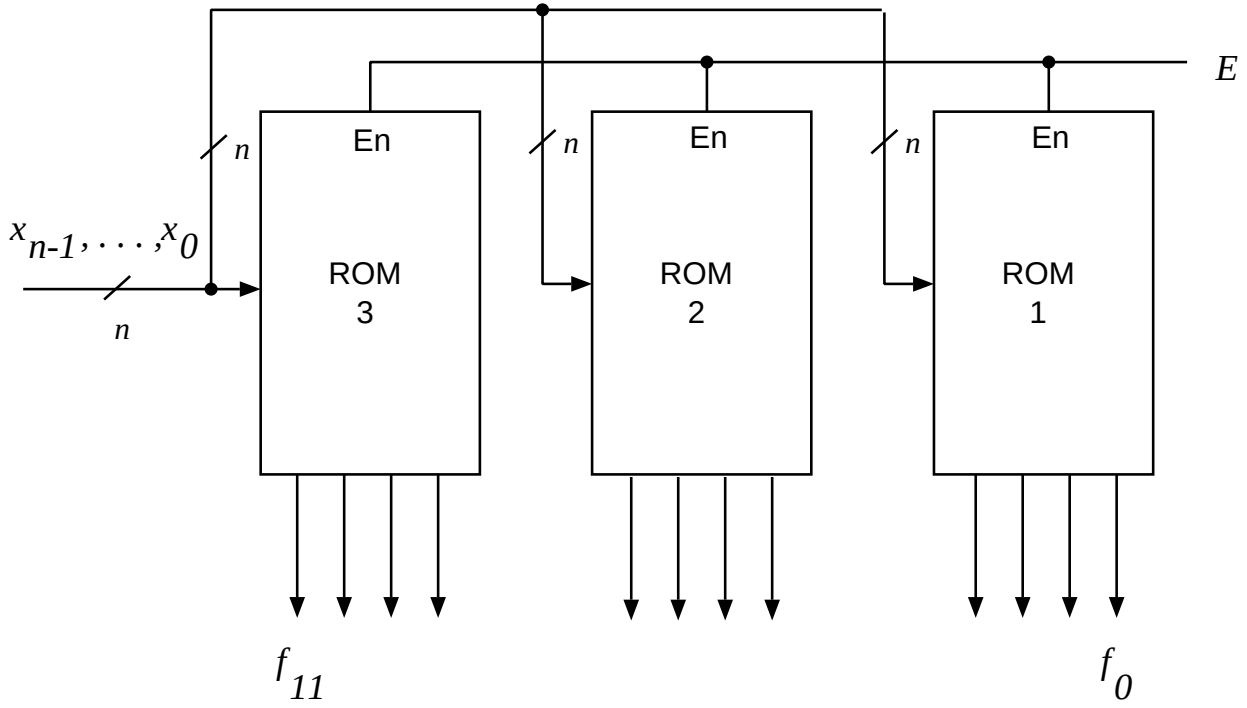


Figure 12.10: ROM-BASED IMPLEMENTATION OF LARGE NUMBER OF SWITCHING FUNCTIONS.

FIELD PROGRAMMABLE GATE ARRAYS (FPGA)

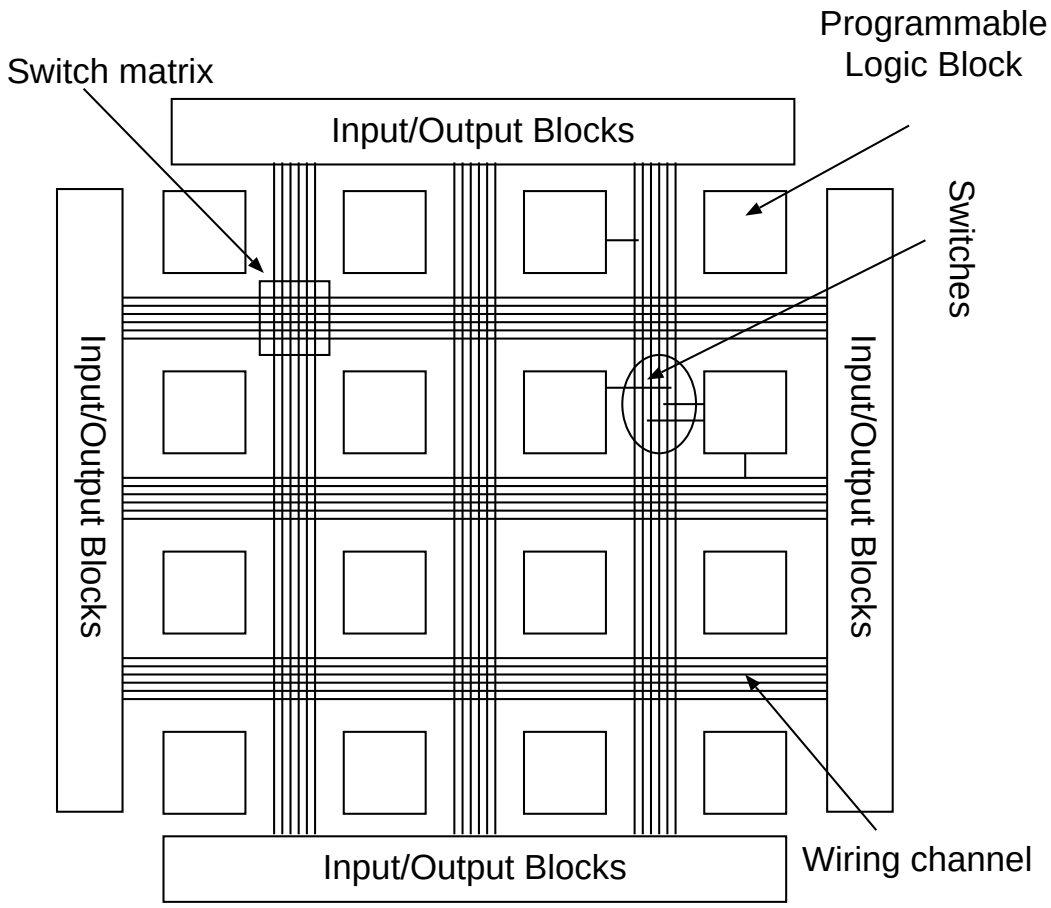


Figure 12.11: ORGANIZATION OF AN FPGA CHIP.

BASIC APPROACHES IN PROGRAMMING OF FPGAs

- ON-CHIP STATIC RAM LOADED WITH CONFIGURATION BIT PATTERNS (SRAM-FPGA). (volatile)

- ANTIFUSE-PROGRAMMED DEVICES PROGRAMMED ELECTRICALLY TO PROVIDE CONNECTIONS THAT DEFINE CHIP CONFIGURATION

- ARRAY-STYLE EPROM and EEPROM PROGRAMMED DEVICES
  USING SEVERAL PLAs AND A SHARED INTERCONNECT MECHANISM

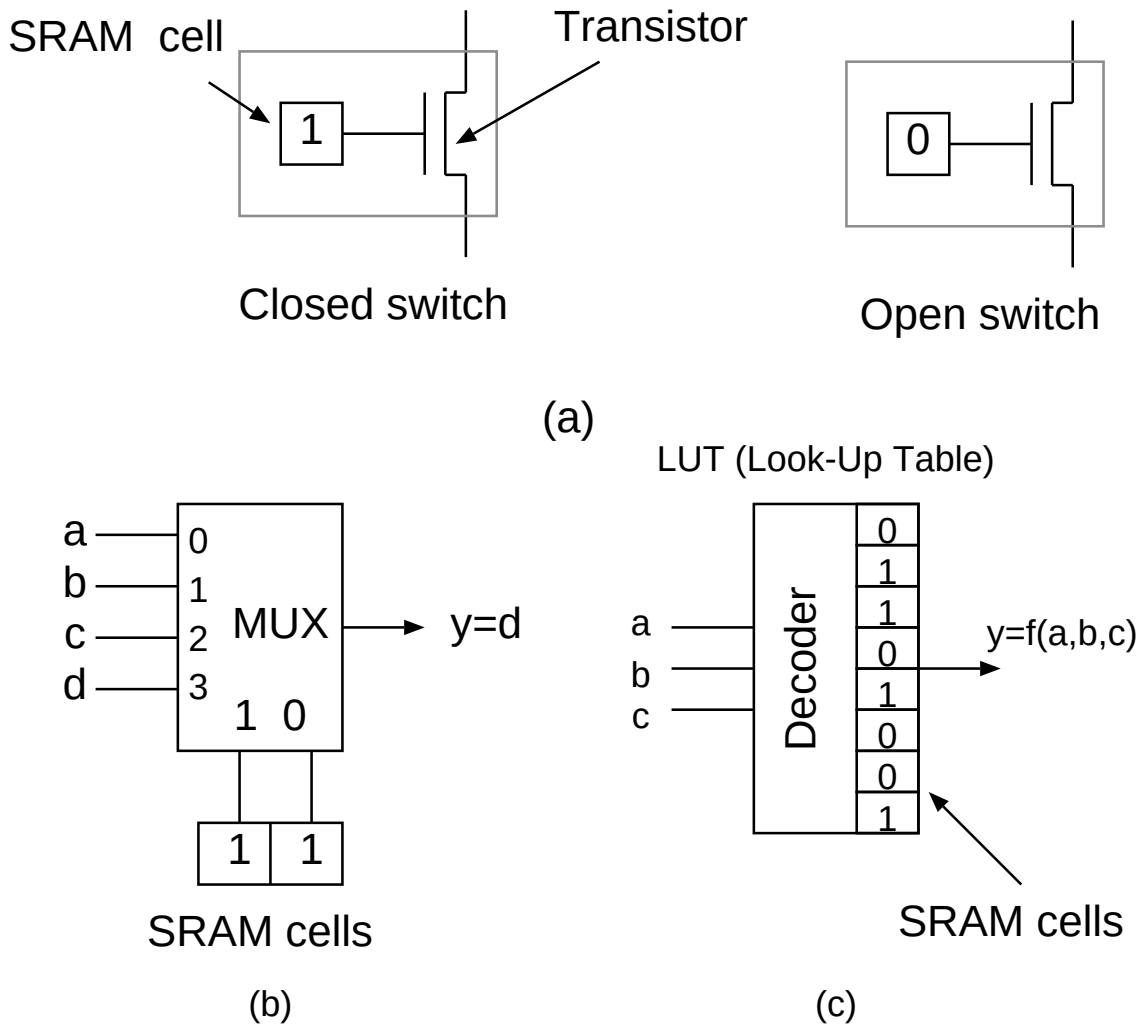


Figure 12.12: SRAM FPGA PROGRAMMABLE COMPONENTS: (a) Switch. (b) 4-input multiplexer. (c) Look-up table (LUT).

Example: XILINX XC2000

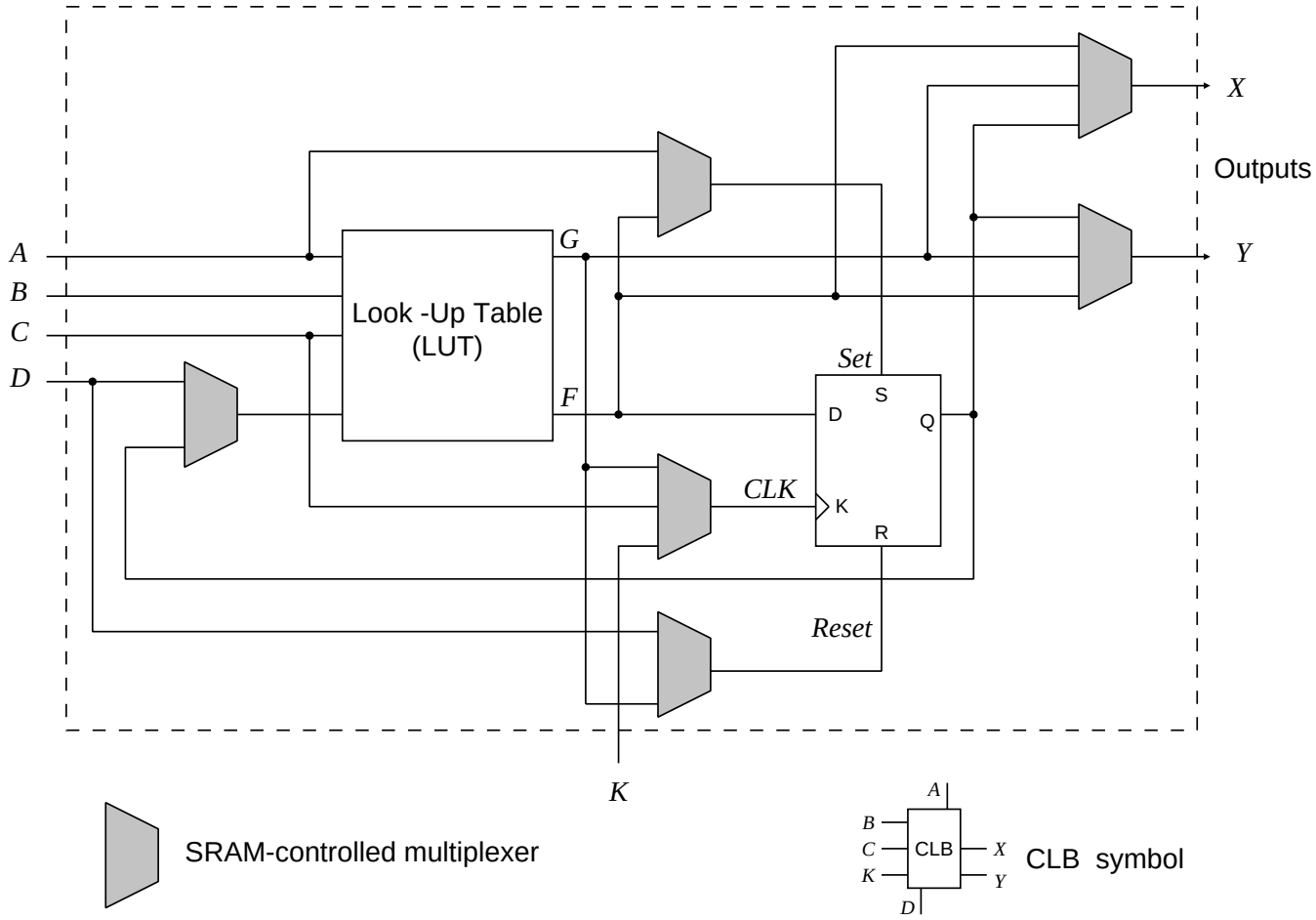


Figure 12.13: A CONFIGURABLE LOGIC BLOCK (CLB) (Courtesy of Xilinx, Inc.)

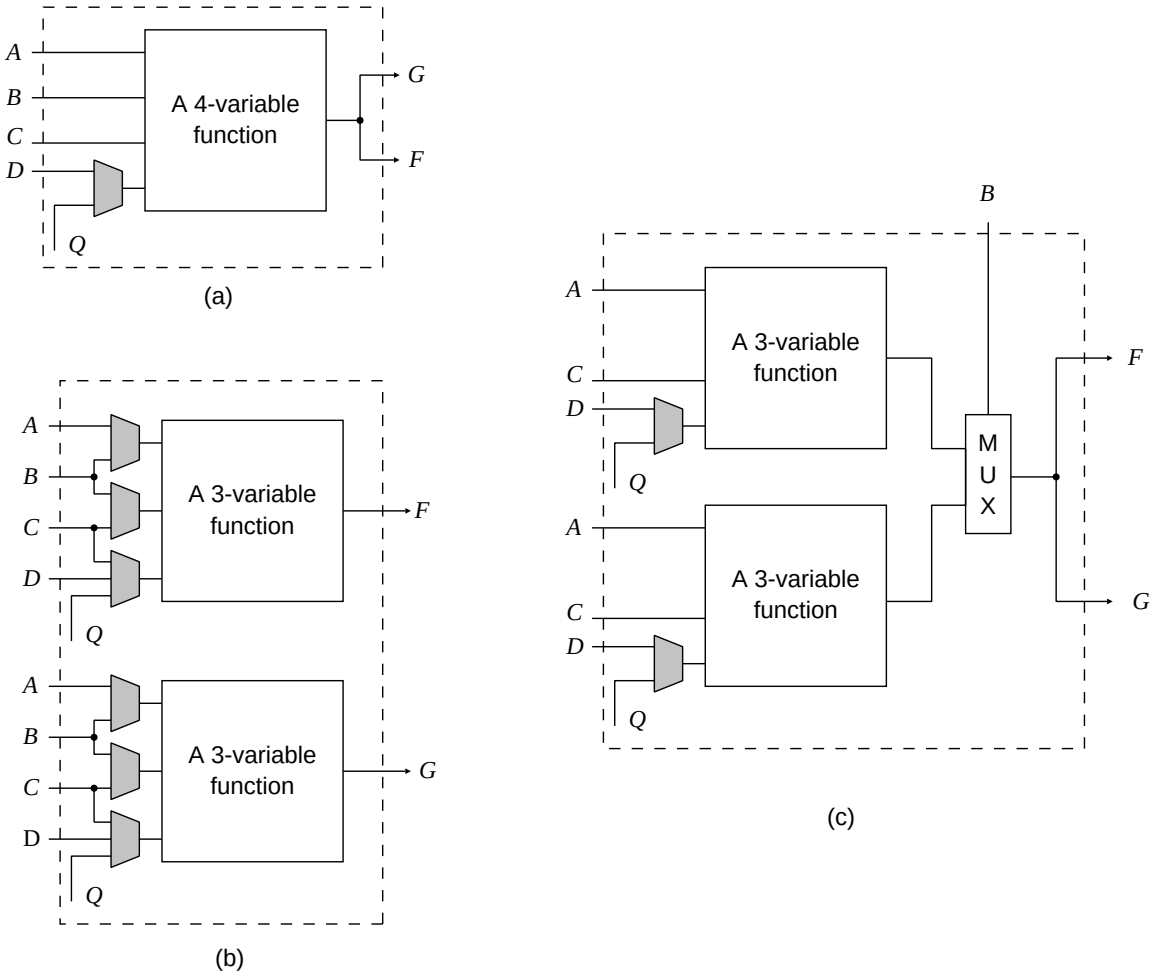


Figure 12.14: SRAM-FPGA options in generating functions: (a) One 4-variable function. (b) Two 3-variable functions. (c) Selection between two functions of 3 variables. (Courtesy of Xilinx, Inc.)

PROGRAMMABLE INTERCONNECT

1. DIRECT INTERCONNECTIONS BETWEEN HORIZONTALLY AND VERTICALLY ADJACENT CLBS – PROVIDE FAST SIGNAL PATHS BETWEEN ADJACENT MODULES

2. GENERAL-PURPOSE INTERCONNECT CONSISTS OF VERTICAL AND HORIZONTAL WIRING SEGMENTS BETWEEN SWITCH MATRICES

3. LONG VERTICAL AND HORIZONTAL LINES SPAN THE WHOLE CLB ARRAY

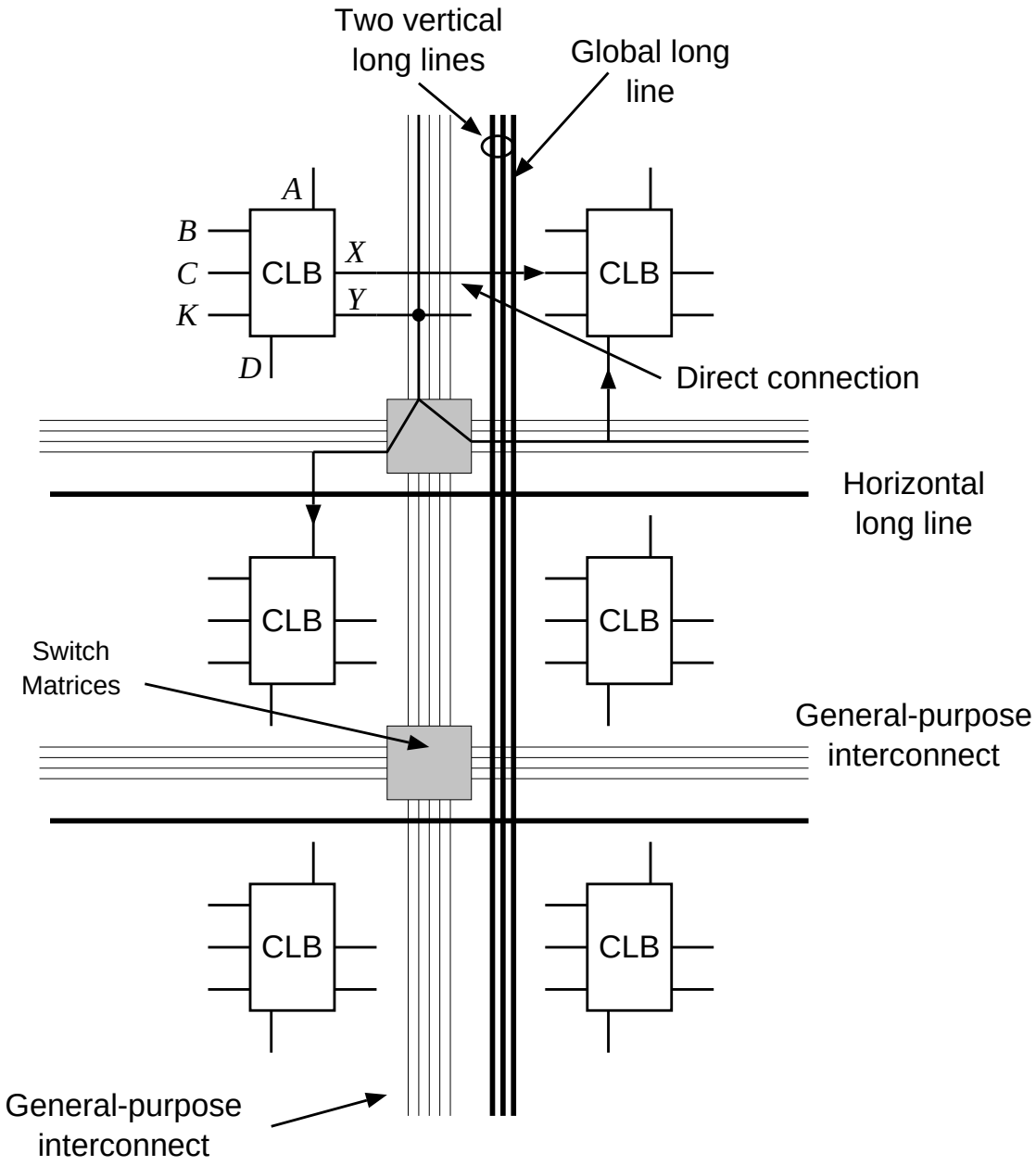


Figure 12.15: PROGRAMMABLE INTERCONNECT. (Courtesy of Xilinx, Inc.)

Example 12.5: BCD ADDER MODULE

- IMPLEMENT A ONE-DIGIT BCD ADDER USING A SRAM-FPGA MODULE OF XC2000 TYPE

INPUTS: $\underline{x} = (x_3, x_2, x_1, x_0), \quad x_j \in \{0, 1\}, \quad x \in \{0, \ldots, 9\}$
$\underline{y} = (y_3, y_2, y_1, y_0), \quad y_j \in \{0, 1\}, \quad y \in \{0, \ldots, 9\}$
$c_{in} \in \{0, 1\}$

OUTPUTS: $\underline{s} = (s_3, s_2, s_1, s_0), \quad s_j \in \{0, 1\}, \quad s \in \{0, \ldots, 9\}$
$c_{out} \in \{0, 1\}$

FUNCTION: $x + y + c_{in} = 10c_{out} + s$

- COMPUTE $16u + v = x + y + c_{in} \in \{0, \ldots, 19\}$ using a 4-bit binary adder

Example 12.5 (cont.)

- THREE CASES:

$$\begin{array}{lll} u = 0 \;\; v \leq 9 & s = v & c_{out} = 0 \\ u = 0 \;\; v > 9 & s = v - 10 = (v + 6) \; mod \; 16 & c_{out} = 1 \\ u = 1 & s = v + 16 - 10 = v + 6 & c_{out} = 1 \end{array}$$

$\Rightarrow$ BCD OUTPUT

$$s = \begin{cases} (v + 6) mod 16 & \textbf{if} \;\; u = 1 \; or \; v \geq 10 \\ v & \textbf{otherwise} \end{cases}$$

$$c_{out} = \begin{cases} 1 & \textbf{if} \;\; u = 1 \; or \; v \geq 10 \\ 0 & \textbf{otherwise} \end{cases}$$

cont.

THE CONDITION $u = 1 \; or \; v \geq 10$ CORRESPONDS TO SWITCHING EXPRESSION

$$t = u + v_3v_2 + v_3v_1$$

Example 12.5 (cont.)

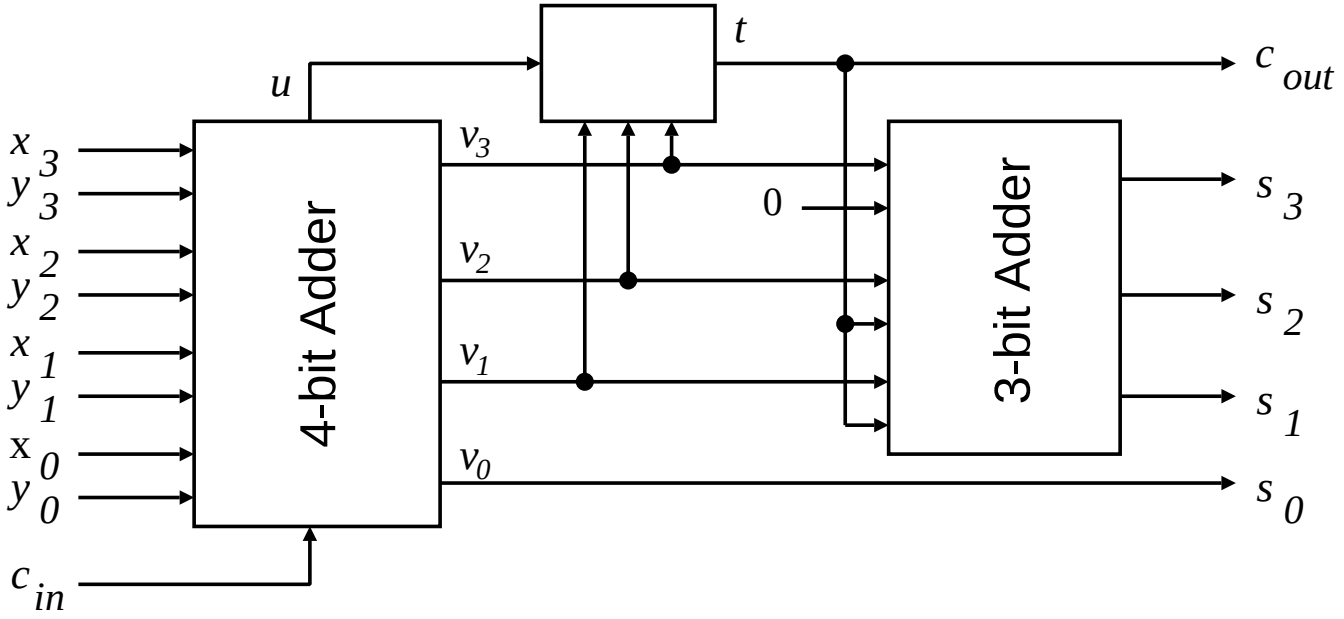


Figure 12.16: IMPLEMENTATION OF BCD ADDER MODULE

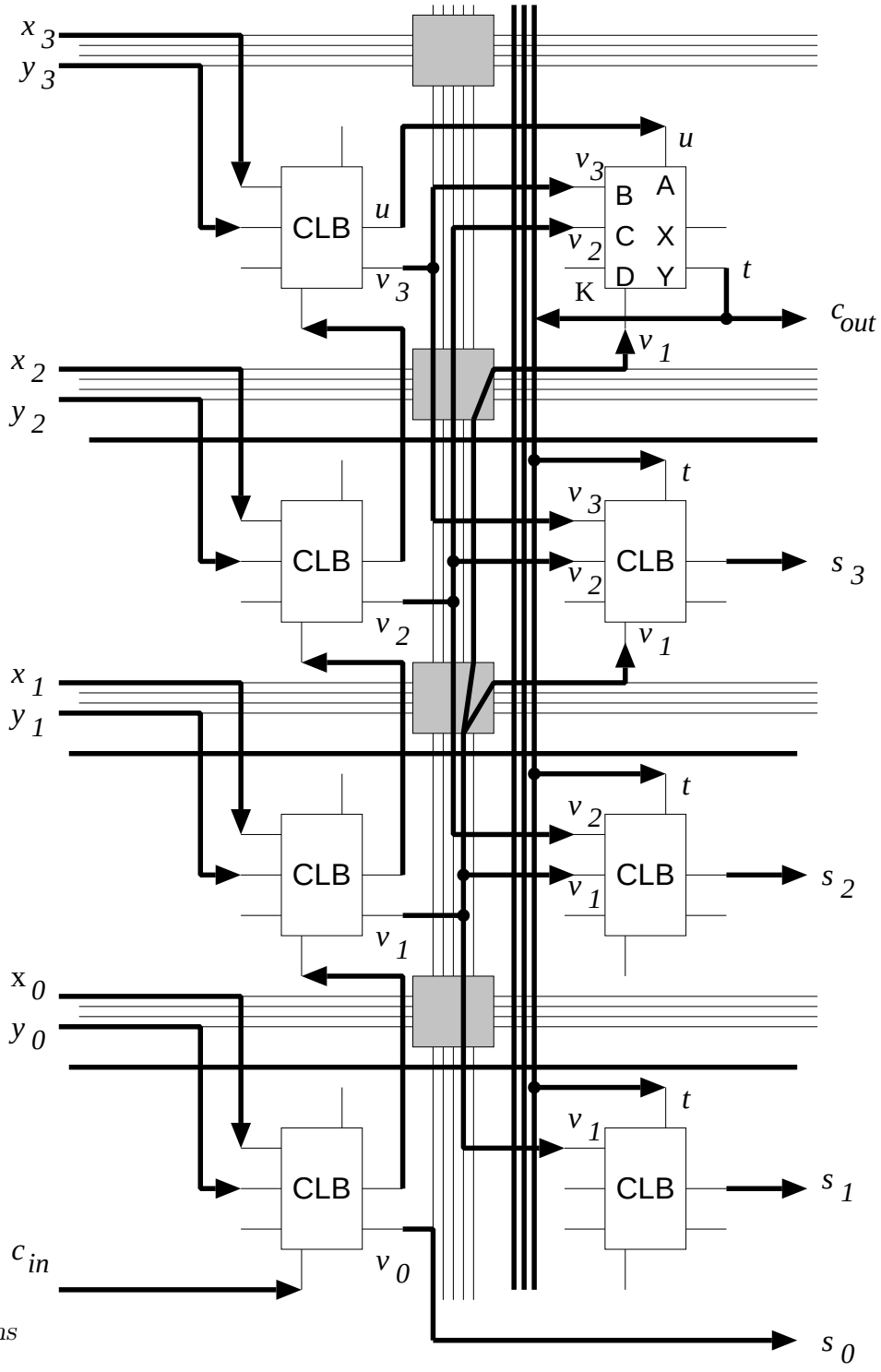
Example 12.5 (cont.)

- SIMPLIFICATION OF THE 3-BIT ADDER

$$
\begin{aligned}
s_3 &= v_3 \oplus t(v_2 + v_1) \\
s_2 &= v_2 \oplus t v_1' \\
s_1 &= v_1 \oplus t
\end{aligned}
$$

MOREOVER,

$$
\begin{aligned}
s_0 &= v_0 \\
c_{out} &= t
\end{aligned}
$$


x 3
y 3
u
v 3
B A
C X
v 2
D Y
K
t
c out
v 1
CLB
u
v 3
x 2
y 2
t
v 3
CLB
CLB
s 3
v 2
v 1
x 1
y 1
t
v 2
CLB
CLB
s 2
v 1
v 1
x 0
y 0
t
v 1
CLB
CLB
s 1
c in
v 0
s 0

DESIGN WITH FPGAs

INVOLVES INTENSIVE USE OF CAD TOOLS AND MODULE LIBRARIES

Design entry : A SCHEMATIC ENTRY OR A BEHAVIORAL DESCRIPTION

Implementation :

- PARTITION OF DESIGN INTO SUBMODULES THAT CAN BE MAPPED ONTO CLBs,
- PLACEMENT OF SUBMODULES ONTO CHIP, AND
- ROUTING OF SIGNALS TO CONNECT THE SUBMODULES

cont.

Design verification :

- IN-CIRCUIT TESTING
- SIMULATION, AND
- TIMING ANALYSIS