

STANDARD SEQUENTIAL MODULES

- REGISTERS
- SHIFT REGISTERS
- SYNCHRONOUS COUNTERS
- FOR EACH MODULE WE SHOW:
 - SPECIFICATION
 - IMPLEMENTATION WITH FFs AND GATES
 - BASIC USES
 - HOW TO IMPLEMENT LARGER MODULES

REGISTERS

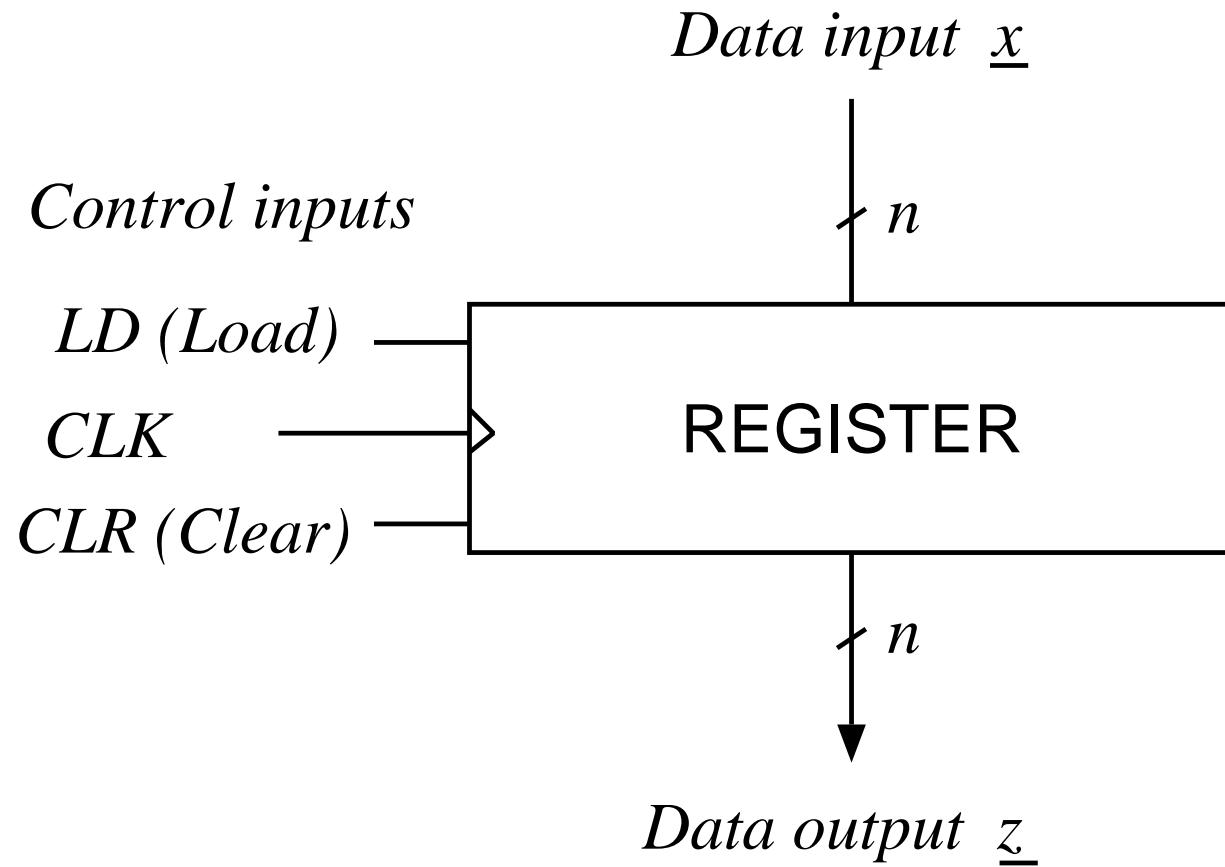


Figure 11.1: n -BIT REGISTER MODULE

REGISTER: HIGH-LEVEL SPECIFICATION

INPUTS: $\underline{x} = (x_{n-1}, \dots, x_0)$, $x_i \in \{0, 1\}$
 $LD, CLR \in \{0, 1\}$

OUTPUTS: $\underline{z} = (z_{n-1}, \dots, z_0)$, $z_i \in \{0, 1\}$

STATE: $\underline{s} = (s_{n-1}, \dots, s_0)$, $s_i \in \{0, 1\}$

FUNCTION: STATE TRANSITION AND OUTPUT FUNCTIONS

$$\underline{s}(t+1) = \begin{cases} \underline{x}(t) & \text{if } LD(t) = 1 \text{ and } CLR(t) = 0 \\ \underline{s}(t) & \text{if } LD(t) = 0 \text{ and } CLR(t) = 0 \\ (0 \dots 0) & \text{if } CLR(t) = 1 \end{cases}$$

$$\underline{z}(t) = \underline{s}(t)$$

IMPLEMENTATION OF 4-BIT REGISTER

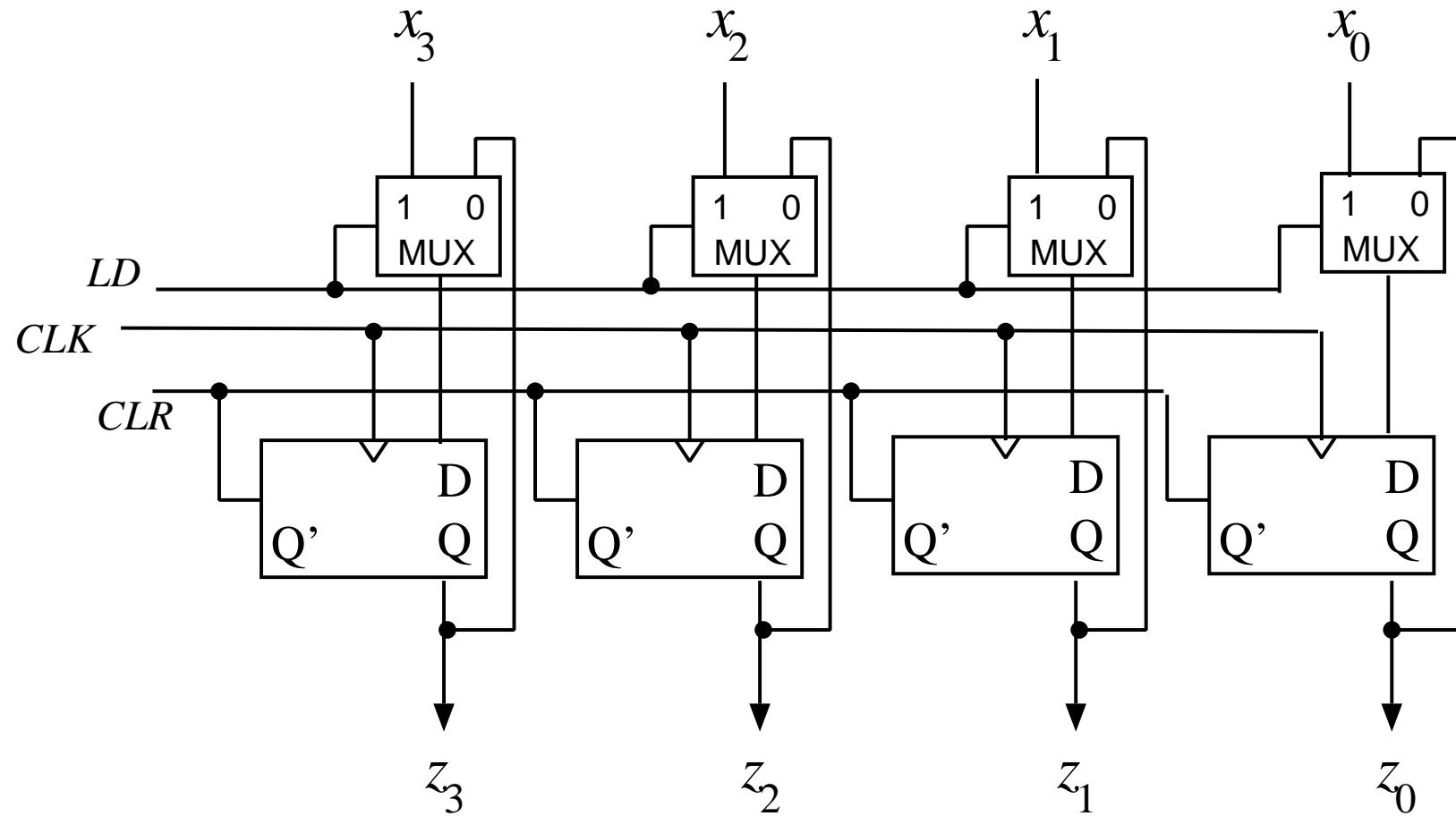


Figure 11.2: IMPLEMENTATION OF 4-BIT REGISTER.

TIME-BEHAVIOR OF REGISTER

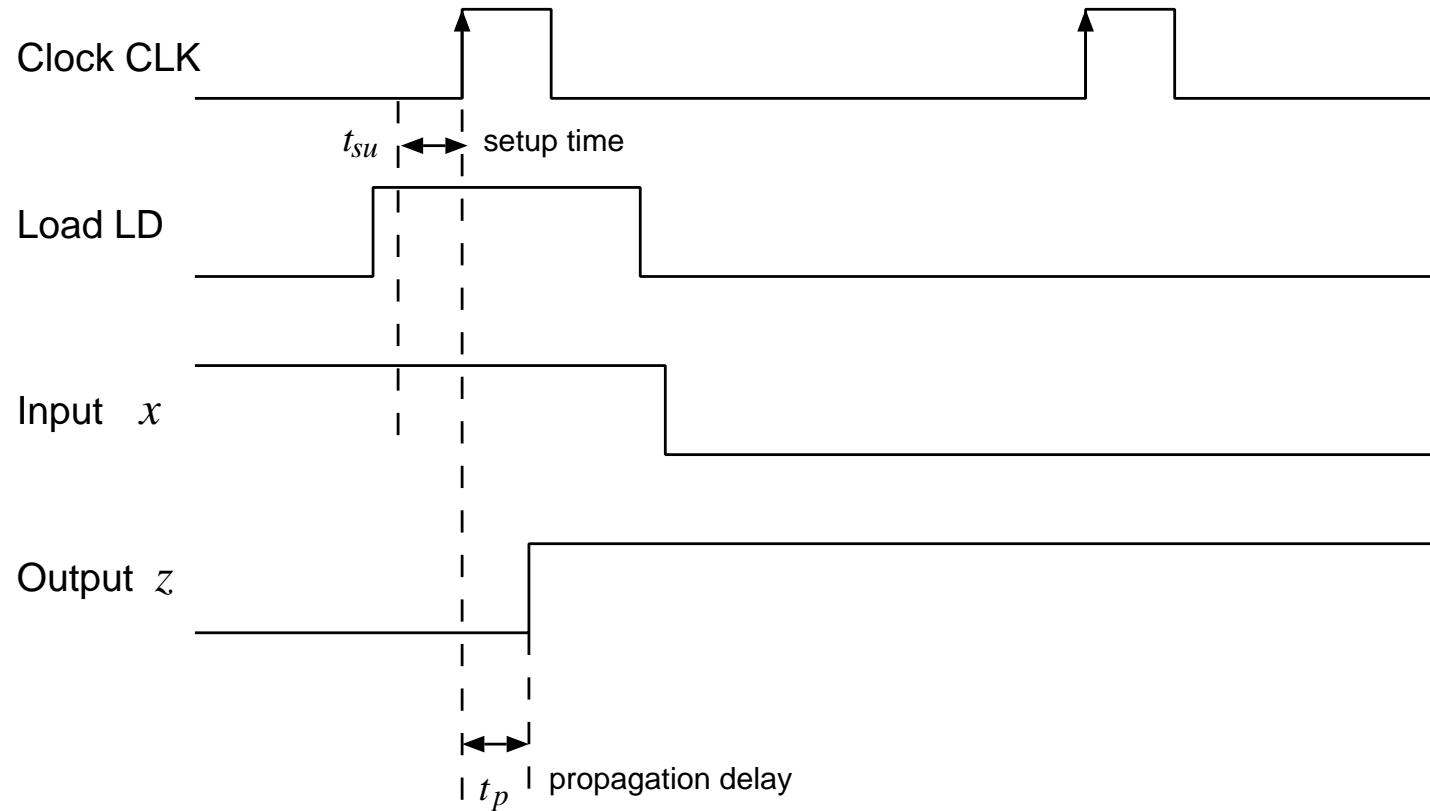
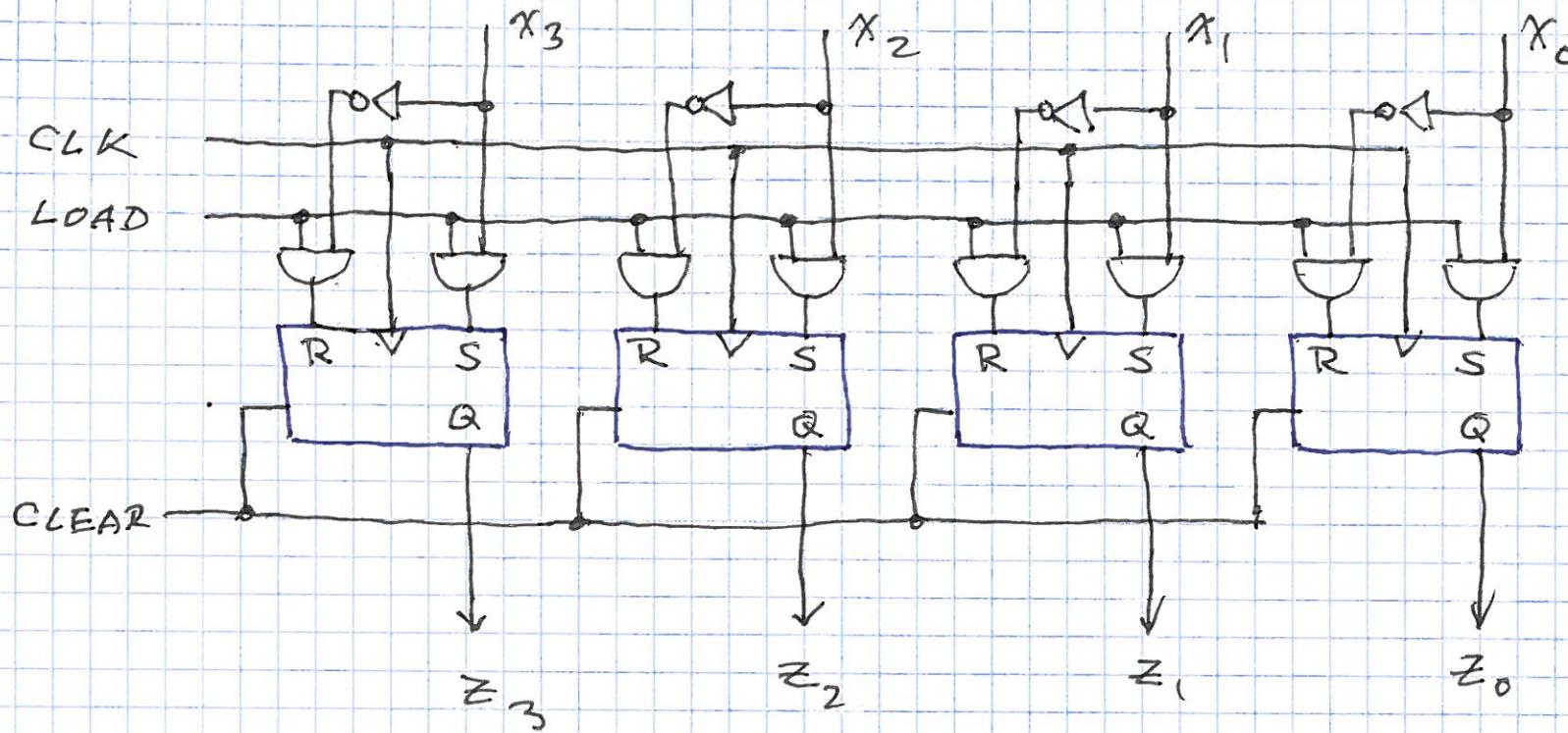


Figure 11.3: TIME-BEHAVIOR OF REGISTER.

CS MSIA

IMPLEMENT 4-BIT REGISTER WITH SR FFS



REPEAT WITH JK AND T. DISCUSS DIFFERENCES.

USES OF REGISTERS: Example 11.1

INPUT: $x \in \{0, 1\}$

OUTPUT: $(z_1, z_0), z_i \in \{0, 1\}$

STATE: $(s_1, s_0), s_i \in \{0, 1\}$

INITIAL STATE: $(s_1, s_0) = (0, 0)$

FUNCTION: STATE TRANSITION AND OUTPUT FUNCTIONS:

PS	Input	
	$x = 0$	$x = 1$
00	00	01
01	01	11
11	11	10
10	10	00
	NS	

$$z(t) = s(t)$$

CANONICAL IMPLEMENTATION

$$Y_1 = y_1x' + y_0x \quad Y_0 = y_0x' + y'_1x$$

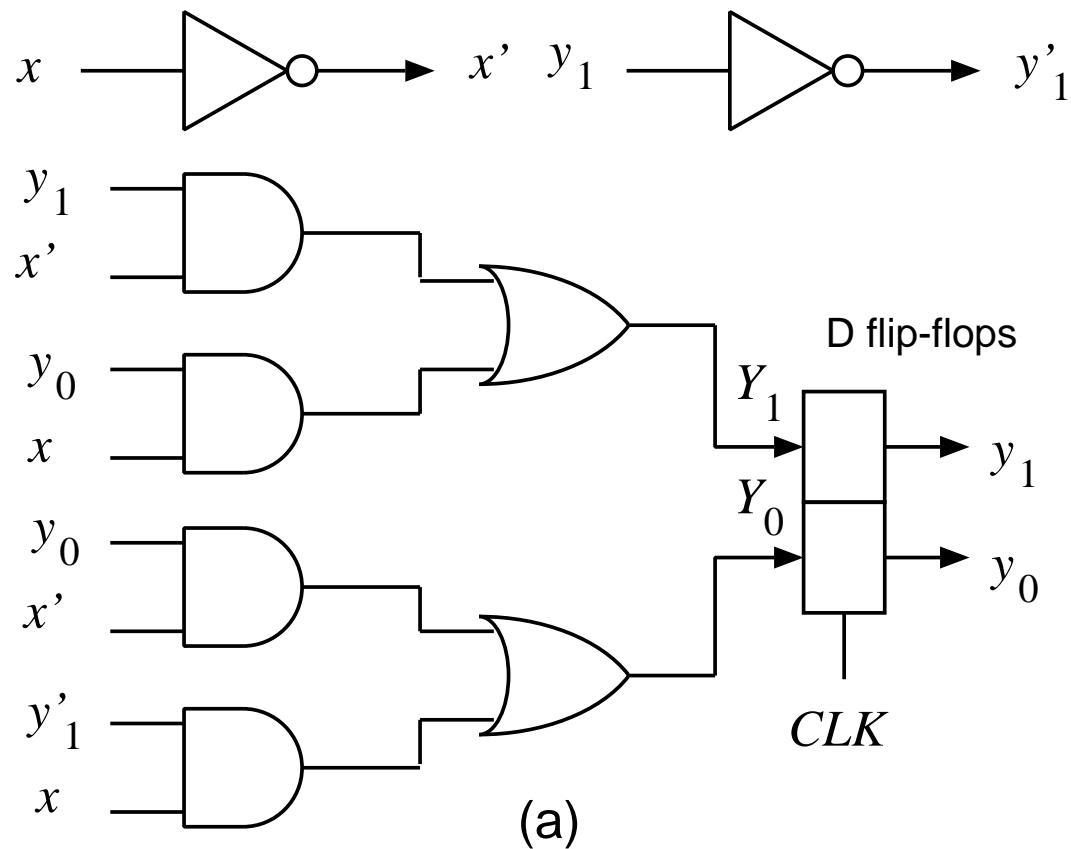


Figure 11.4: NETWORKS FOR Example 11.1: a) NETWORK WITH STATE CELLS;

IMPLEMENTATION WITH REGISTER

$$Y_1 = y_0 \quad Y_0 = y'_1 \quad LD = x$$

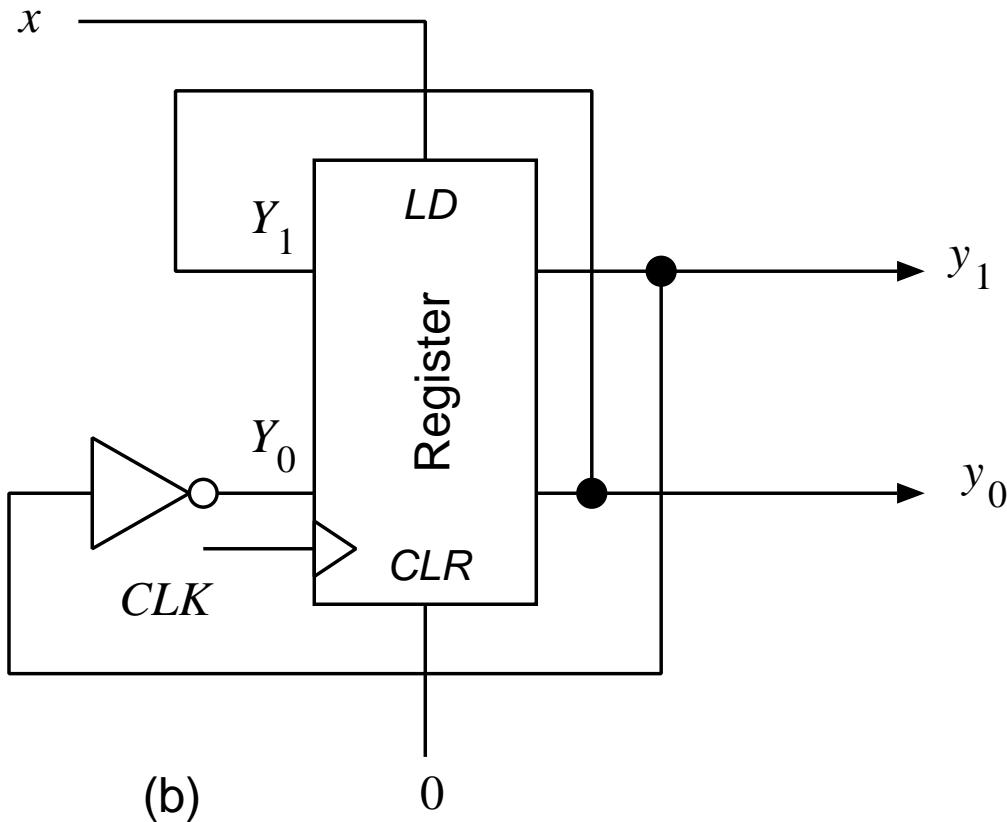


Figure 11.4: NETWORKS for Example 11.1: b) NETWORK WITH STANDARD REGISTER MODULE

CSMSIA F14 MDE

REGISTER FILE: STORES OPERANDS & TEMPORARY RESULTS

FASTER ACCESS THAN TO CACHE

INPUTS:

- READ AND

WRITE ADDRESSES RELATIVELY SMALL SIZE

CLK

• READ AND

WRITE OPS

• DATA IN

OUTPUTS: DATA OUTPUTS
(LEFT, RIGHT)

PARAMETERS: 2^k REGISTERS

k -BIT ADDRESSES

n -BIT PRECISION

EXAMPLE: $k=2$, 4 REGISTERS

$n=16$

2 READS, 1 WRITE

(DoutL, DoutR, DIN)

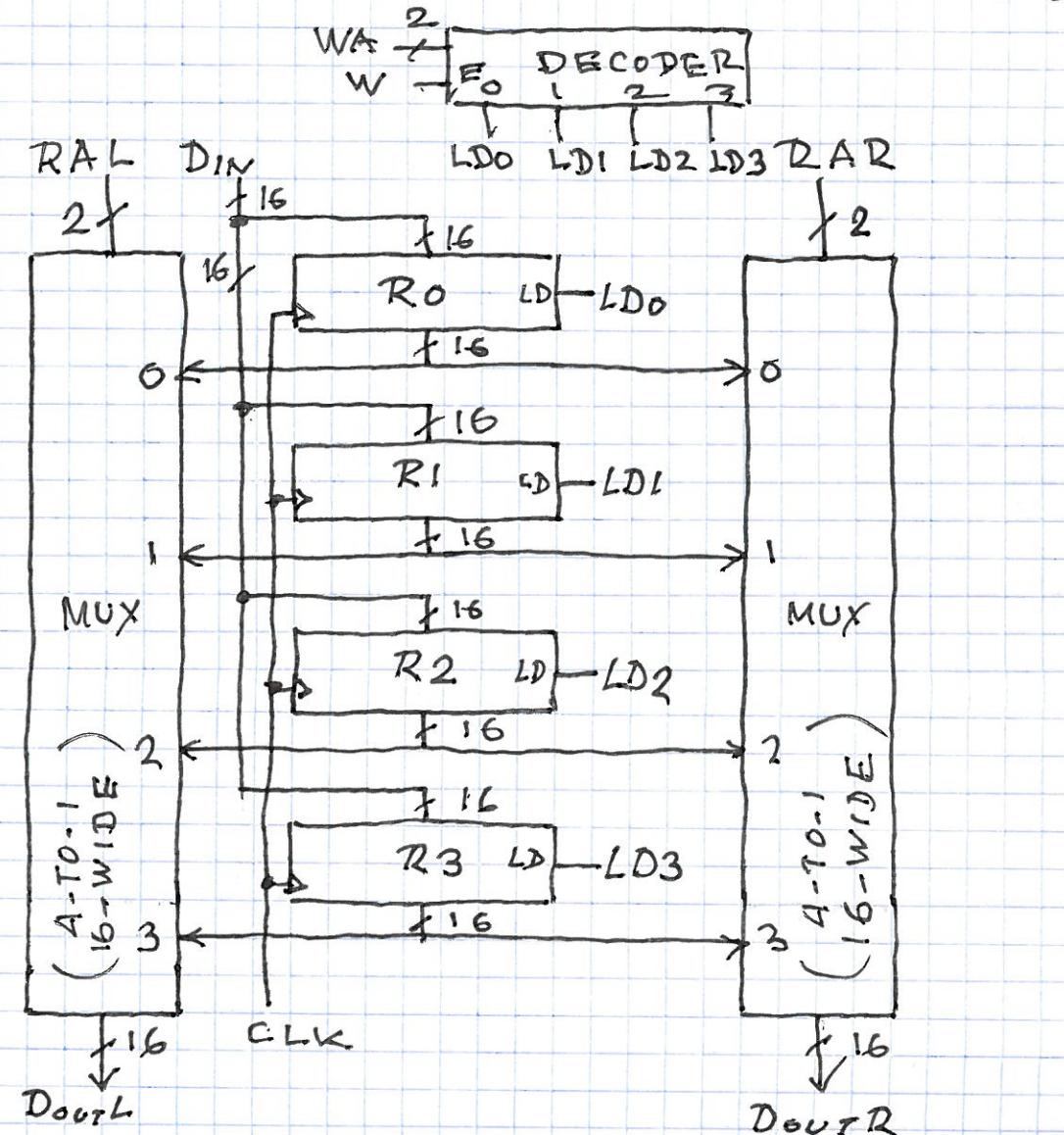
RAL - LEFT READ ADDRESS

ADDS

DAR - RIGHT READ ADDRESS

WA - WRITE ADR,

W - WRITE OP



SHIFT REGISTERS

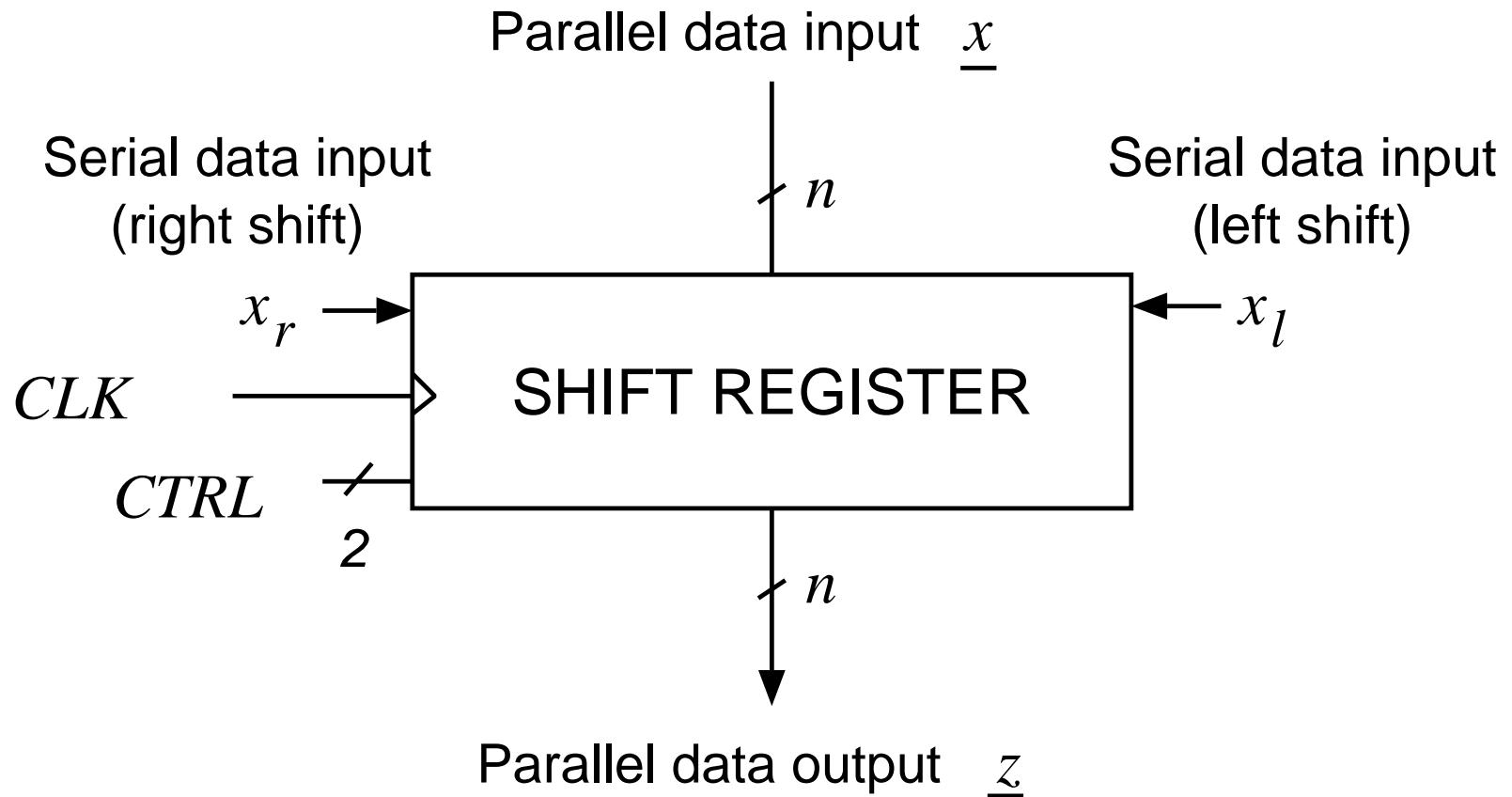


Figure 11.5: SHIFT REGISTER

PARALLEL-IN/PARALLEL-OUT BIDIRECTIONAL SHIFT REGISTER

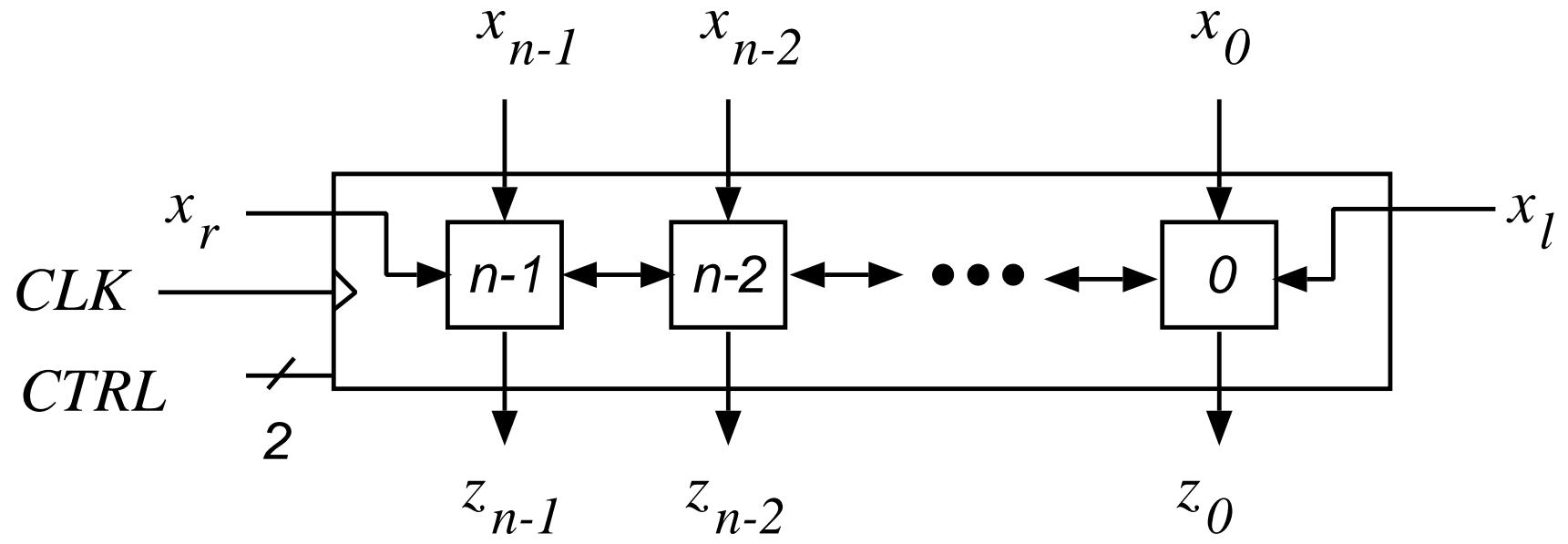


Figure 11.6: PARALLEL-IN/PARALLEL-OUT BIDIRECTIONAL SHIFT REGISTER

HIGH-LEVEL SPECIFICATION

INPUTS: $\underline{x} = (x_{n-1}, \dots, x_0), x_i \in \{0, 1\}$
 $x_l, x_r \in \{0, 1\}$
 $CTRL \in \{LOAD, LEFT, RIGHT, NONE\}$
STATE: $\underline{s} = (s_{n-1}, \dots, s_0), s_i \in \{0, 1\}$
OUTPUT: $\underline{z} = (z_{n-1}, \dots, z_0), z_i \in \{0, 1\}$

FUNCTIONS: STATE TRANSITION AND OUTPUT FUNCTIONS:

$$\underline{s}(t+1) = \begin{cases} \underline{s}(t) & \text{if } CTRL = NONE \\ \underline{x}(t) & \text{if } CTRL = LOAD \\ (s_{n-2}, \dots, s_0, x_l) & \text{if } CTRL = LEFT \\ (x_r, s_{n-1}, \dots, s_1) & \text{if } CTRL = RIGHT \end{cases}$$

$$\underline{z} = \underline{s}$$

SHIFT-REGISTER CONTROL

Control		$s(t + 1) = z(t + 1)$
<i>NONE</i>		0101
<i>LOAD</i>		1110
<i>LEFT</i>	$x_l = 0$	1010
<i>LEFT</i>	$x_l = 1$	1011
<i>RIGHT</i>	$x_r = 0$	0010
<i>RIGHT</i>	$x_r = 1$	1010

<i>CTRL</i>	c_1	c_0
<i>NONE</i>	0	0
<i>LEFT</i>	0	1
<i>RIGHT</i>	1	0
<i>LOAD</i>	1	1

4-BIT BIDIRECTIONAL SHIFT-REGISTER

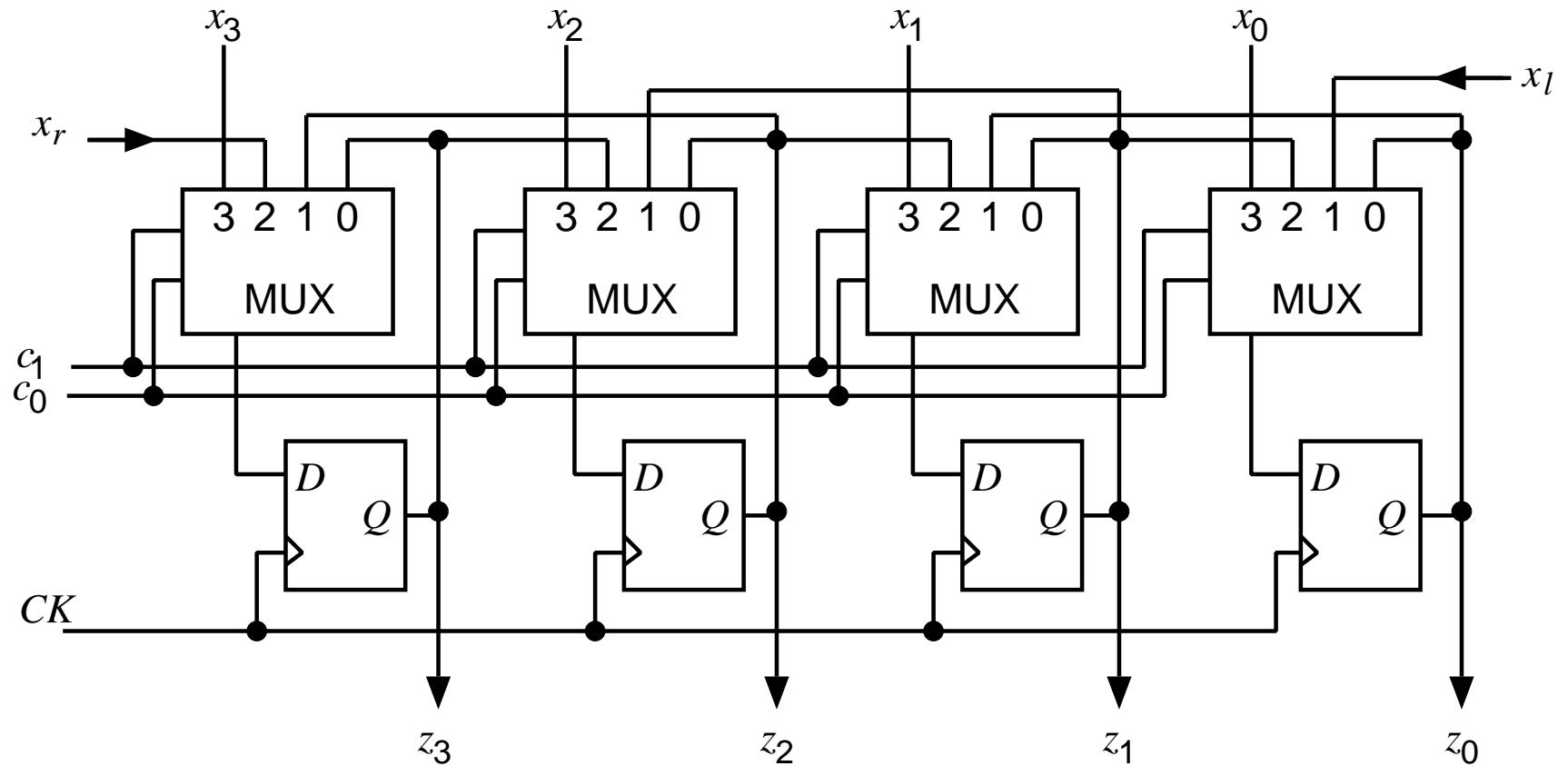
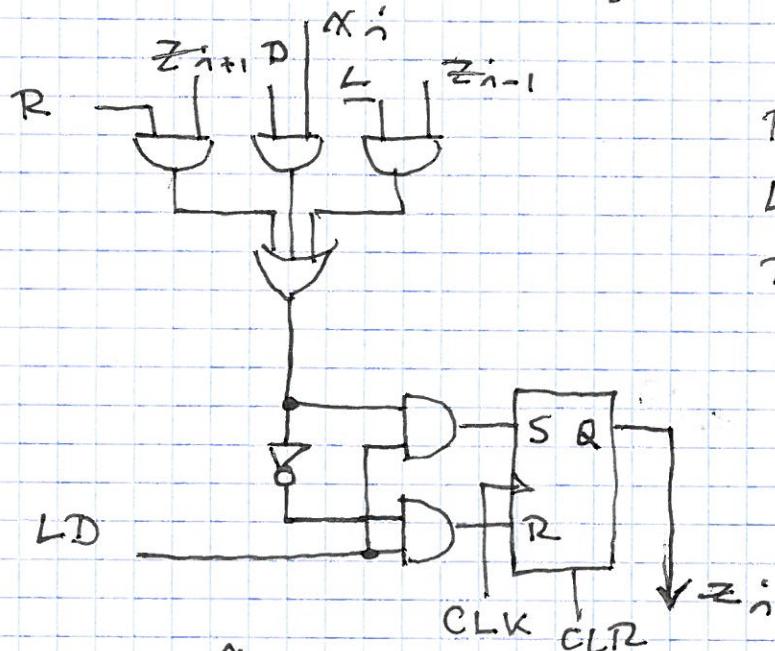


Figure 11.7: IMPLEMENTATION OF A 4-BIT BIDIRECTIONAL SHIFT REGISTER WITH D FLIP-FLOPS.

CS MSIA

IMPLEMENT 4-BIT BIDIRECTIONAL SHIFTER
USING SR FFs AND GATES

SHIFT-REGISTER SLICE: SRS_2

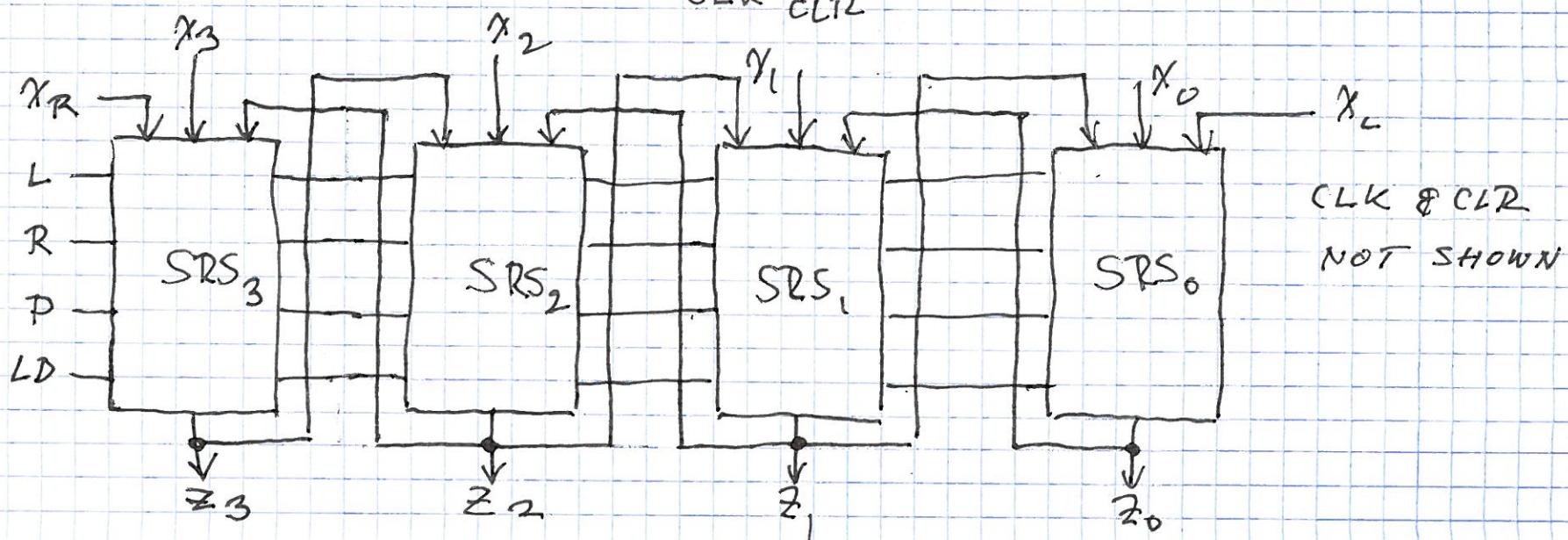


R - SHIFT RIGHT

L - SHIFT LEFT

P - PARALLEL LOAD

LD - LOAD FF



CLK & CLR
NOT SHOWN

SERIAL-IN/SERIAL-OUT UNIDIRECTIONAL SHIFT REGISTER¹⁴

$$z(t) = x(t - n)$$

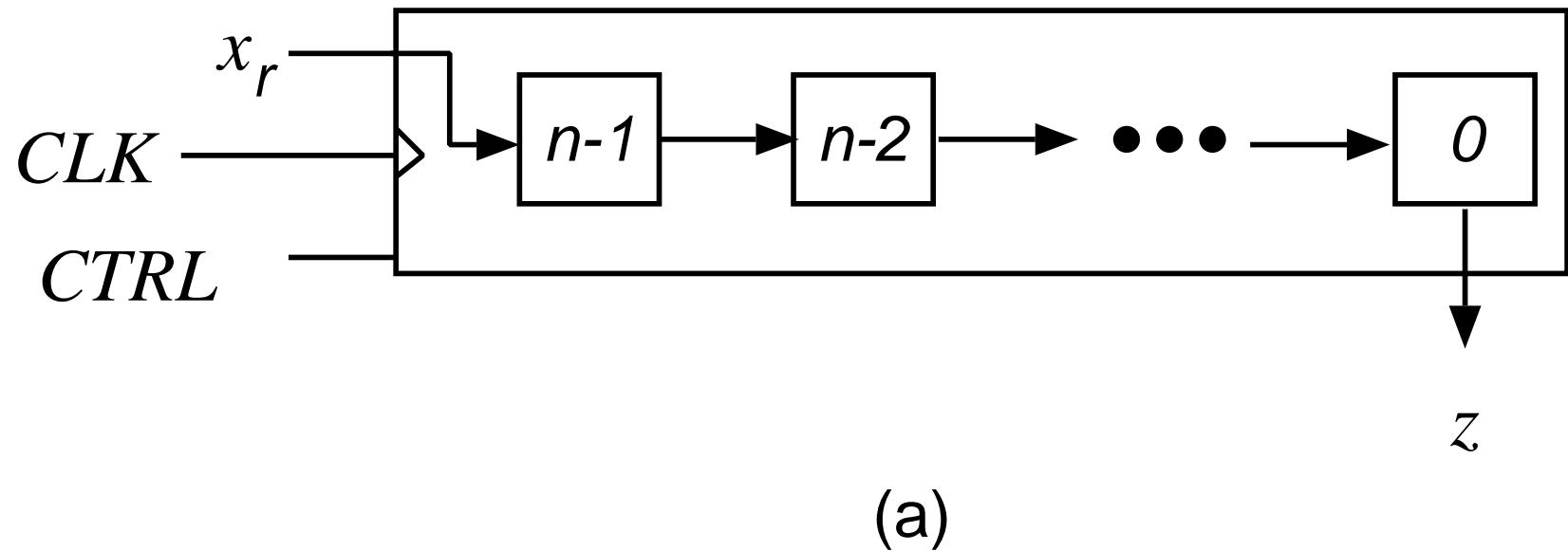


Figure 11.8: COMMON UNIDIRECTIONAL SHIFT REGISTERS: a) SERIAL-IN/SERIAL-OUT

PARALLEL-IN/SERIAL-OUT UNIDIRECTIONAL SHIFT REGISTER

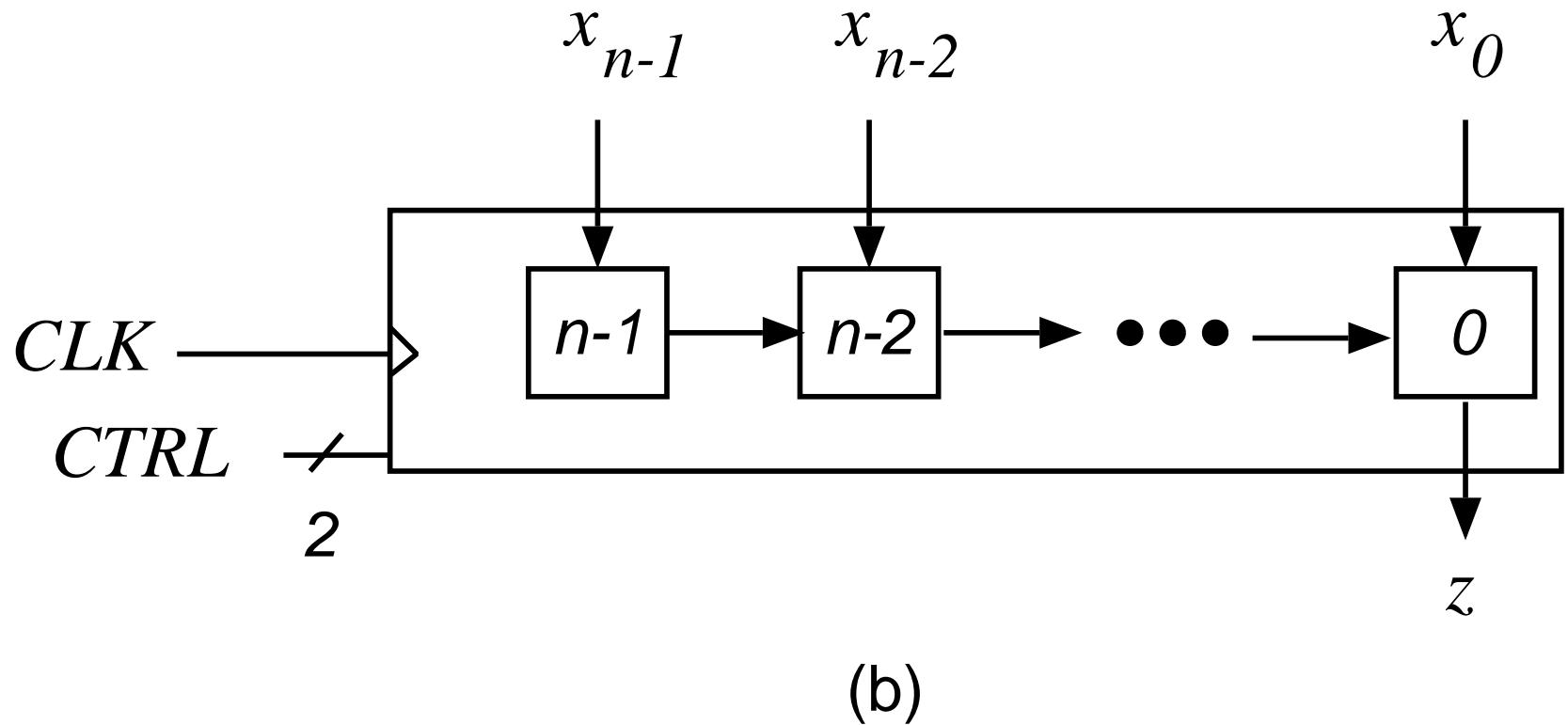


Figure 11.8: COMMON UNIDIRECTIONAL SHIFT REGISTERS: b) PARALLEL-IN/SERIAL-OUT

SERIAL-IN/PARALLEL-OUT UNIDIRECTIONAL SHIFT REGISTER

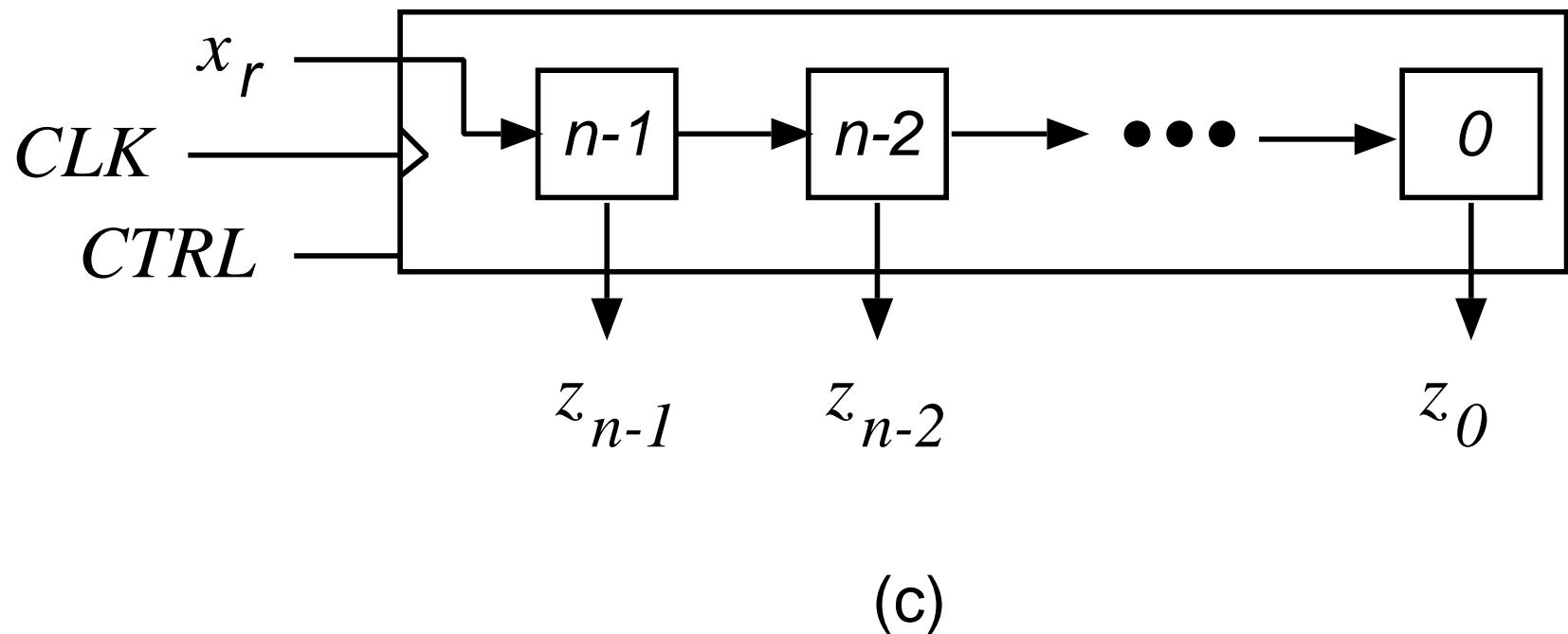


Figure 11.8: COMMON UNIDIRECTIONAL SHIFT REGISTERS: c) SERIAL-IN/PARALLEL-OUT

SUMMARY OF SHIFT-REGISTER TYPES

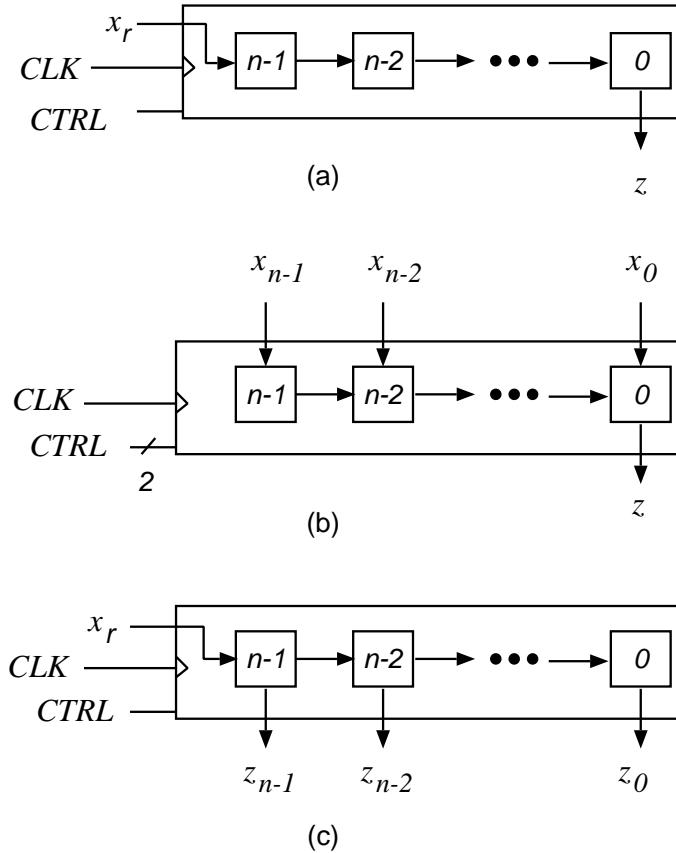


Figure 11.8: COMMON UNIDIRECTIONAL SHIFT REGISTERS: a) SERIAL-IN/SERIAL-OUT; b) PARALLEL-IN/SERIAL-OUT; c) SERIAL-IN/PARALLEL-OUT

USES OF SHIFT REGISTERS

- SERIAL INTERCONNECTION OF SYSTEMS

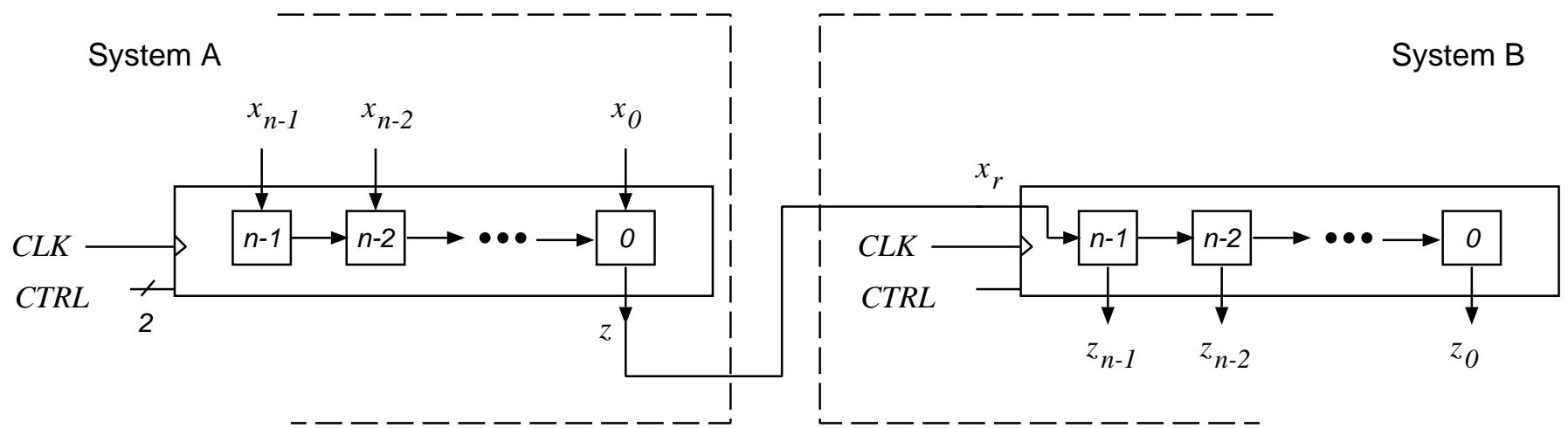


Figure 11.9: SERIAL INTERCONNECTION OF SYSTEMS USING SHIFT REGISTERS

cont.

- BIT-SERIAL OPERATIONS

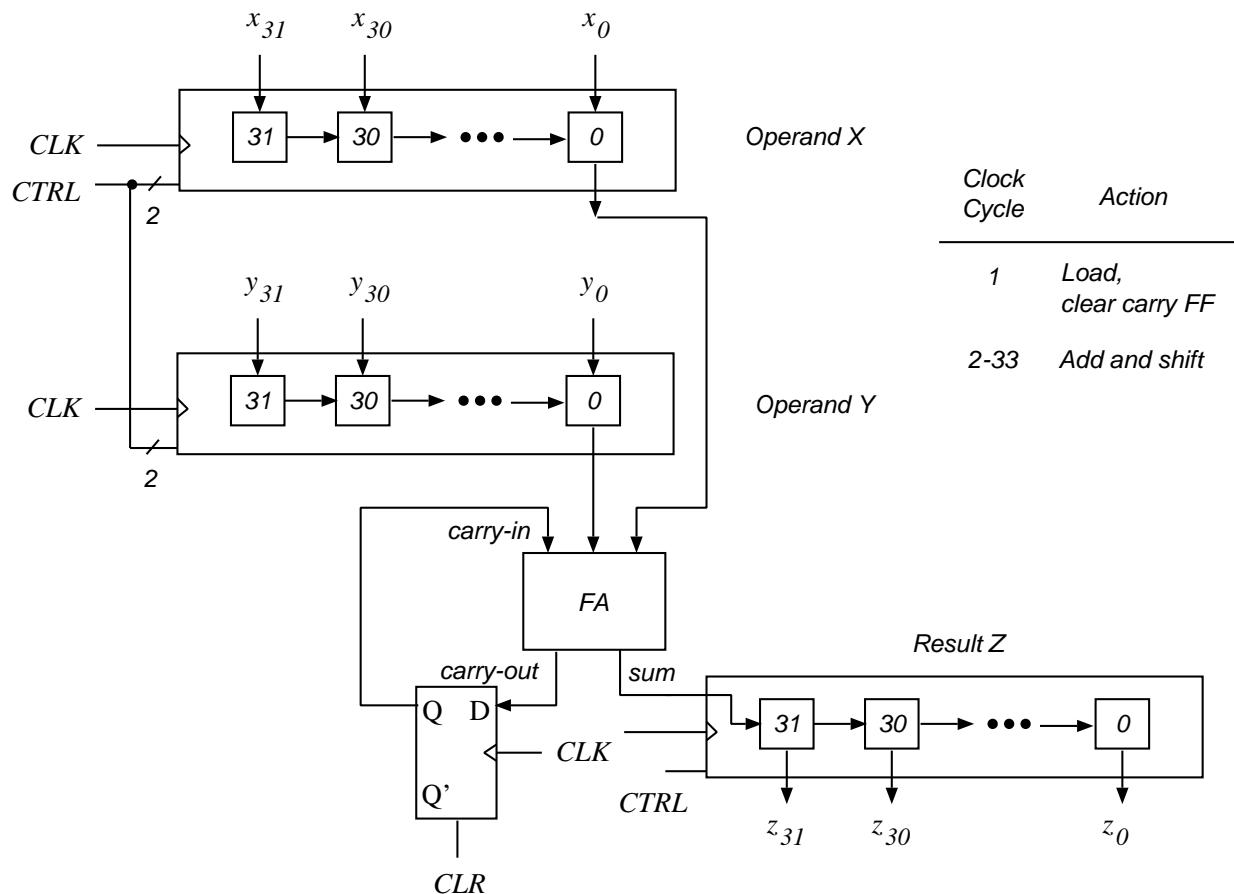


Figure 11.10: BIT-SERIAL ADDER.

INPUT: $X = (x_{15} x_{14} \dots x_1 x_0) = ((\underbrace{x_{15}, x_{14}}_{\in \{0, 1, 2, 3\}}, \underbrace{x_{13}, x_{12}}_{\in \{0, 1, 2, 3\}}, \dots, \underbrace{(x_1, x_0)}_{\in \{0, 1, 2, 3\}})$

$$\text{Cin} \in \{0, 1\}$$

$$Y = (y_{15} y_{14} \dots y_1 y_0) = ((y_{15} y_{14}), \dots, (y_1, y_0))$$

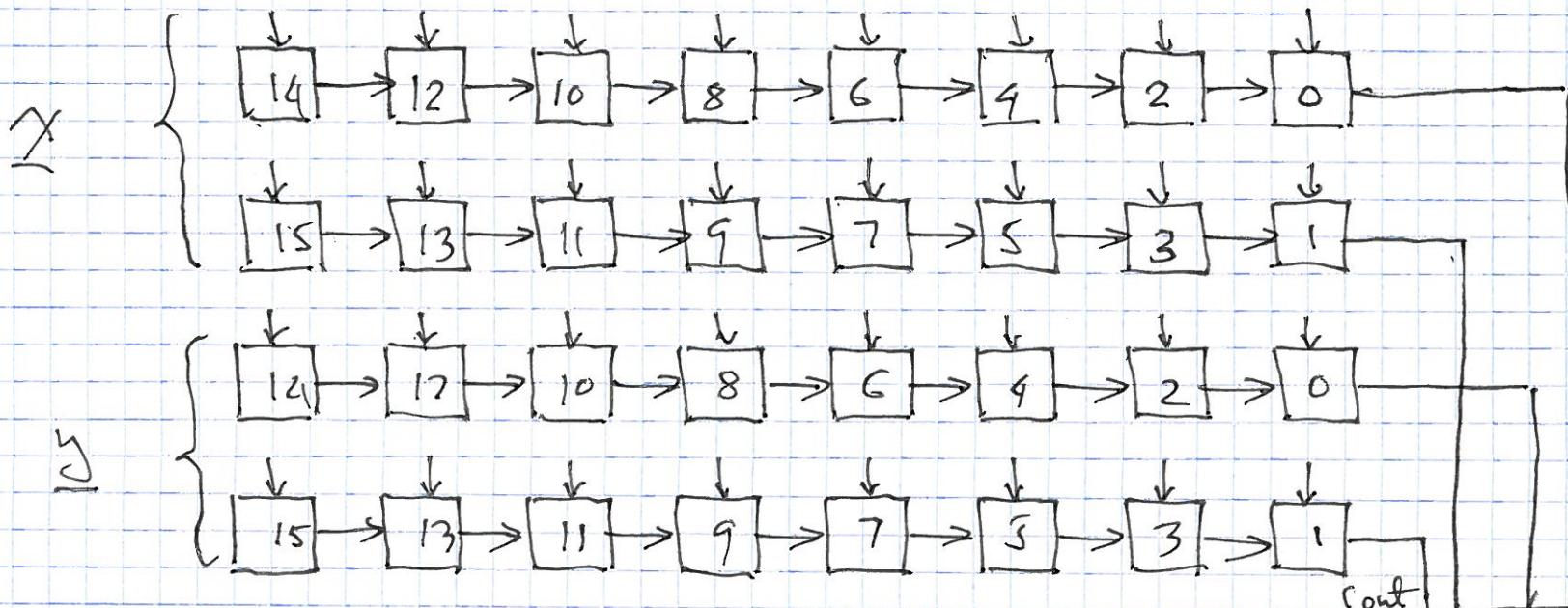
$\in \{0, 1, 2, 3\}$

RADIX-4

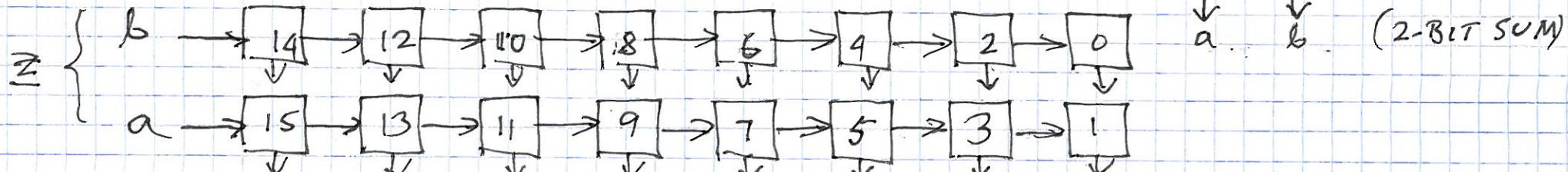
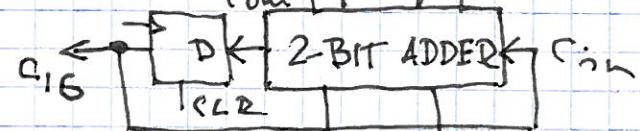
OUTPUT: $Z = (z_{15} z_{14} \dots z_1 z_0)$ SIMILAR

$$\text{Cont} \in \{0, 1\}$$

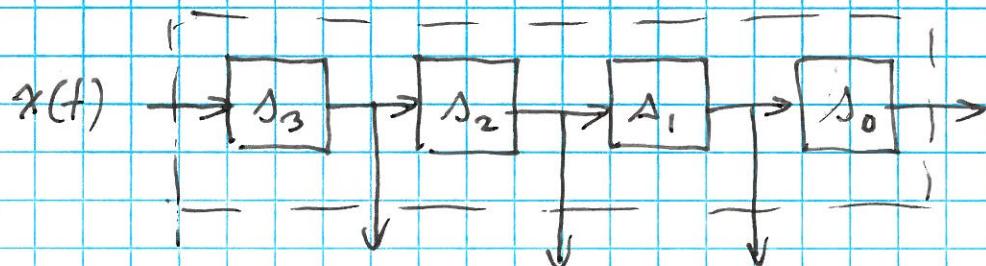
CLK NOT SHOWN



CYCLE 1: LOAD X, Y , CLEAR FF
CYCLES 2 - 9 ADD & SHIFT



SHIFT-REGISTER AS STATE REGISTER



SERIAL-IN, PARALLEL-OUT SHIFT REGISTER

THE STATE REPRESENTED BY A

VECTOR $\underline{S} = (S_{n-1}, S_{n-2}, \dots, S_1, S_0)$

$$S_{n-1}(t+1) = x(t)$$

$$S_i(t+1) = S_{i+1}(t), \quad i = n-2, \dots, 0$$

- FINITE MEMORY SYSTEM

$$z(t) = 1 \text{ IF } x(t) \cdot x(t-8) = 1$$

Example 11.2: SHIFT REGISTER AS STATE REGISTER

$$s_7(t + 1) = x(t)$$

$$s_i(t + 1) = s_{i+1}(t) \text{ for } i = 6, \dots, 0$$

$$z(t) = x(t)s_0(t)$$



Figure 11.11: IMPLEMENTATION OF NETWORK IN Example 11.2

Example 11.3: SHIFT REGISTER AS STATE REGISTER

$$z(t) = \begin{cases} 1 & \text{if } \underline{s}(t) = 01101110 \text{ and } x(t) = 1 \\ 0 & \text{otherwise} \end{cases}$$

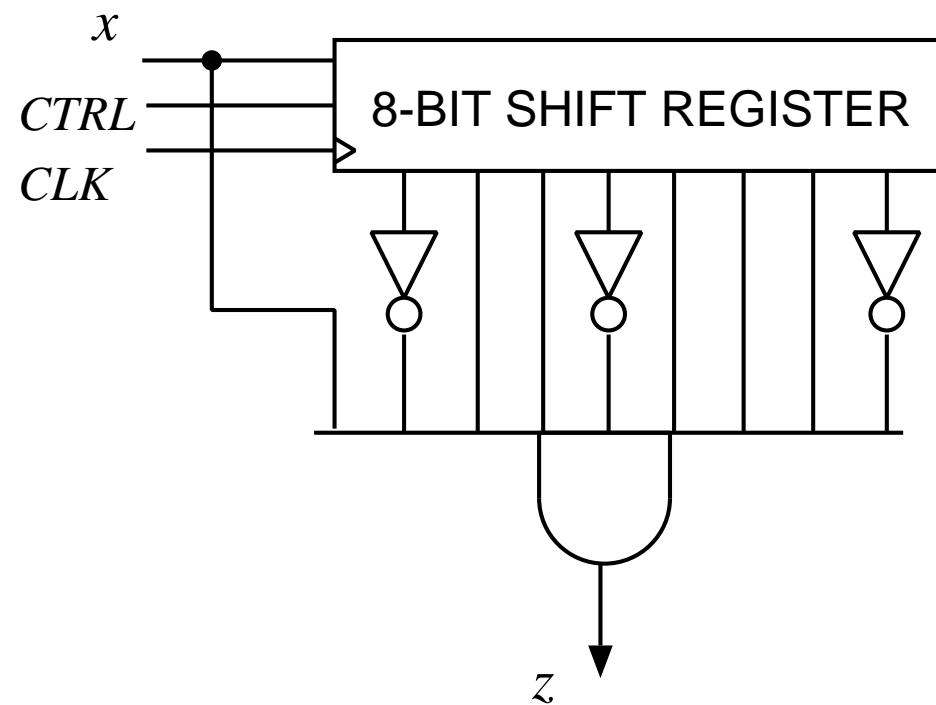


Figure 11.12: IMPLEMENTATION OF NETWORK IN Example 11.3

NETWORKS OF SHIFT REGISTERS

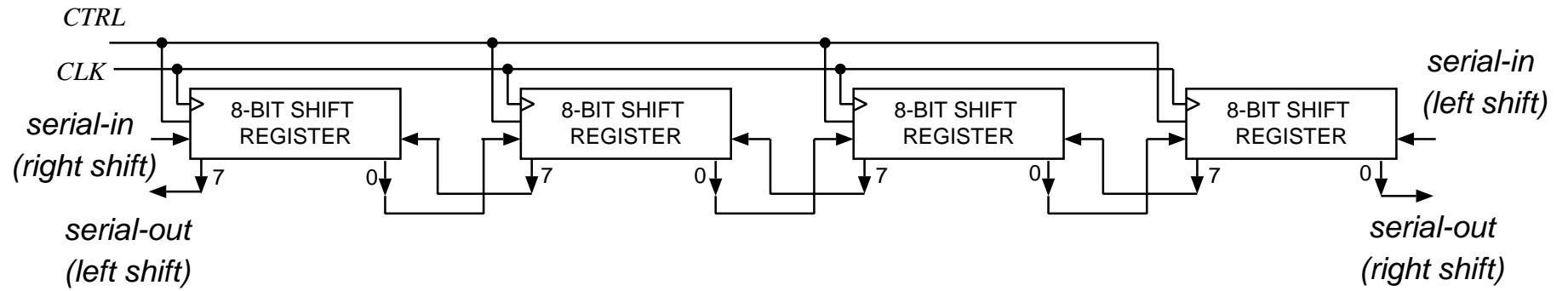


Figure 11.13: NETWORK OF SERIAL-INPUT/SERIAL-OUTPUT SHIFT REGISTER MODULES

COUNTERS

- MODULO- p COUNTER

$$s(t + 1) = (s(t) + x) \bmod p$$

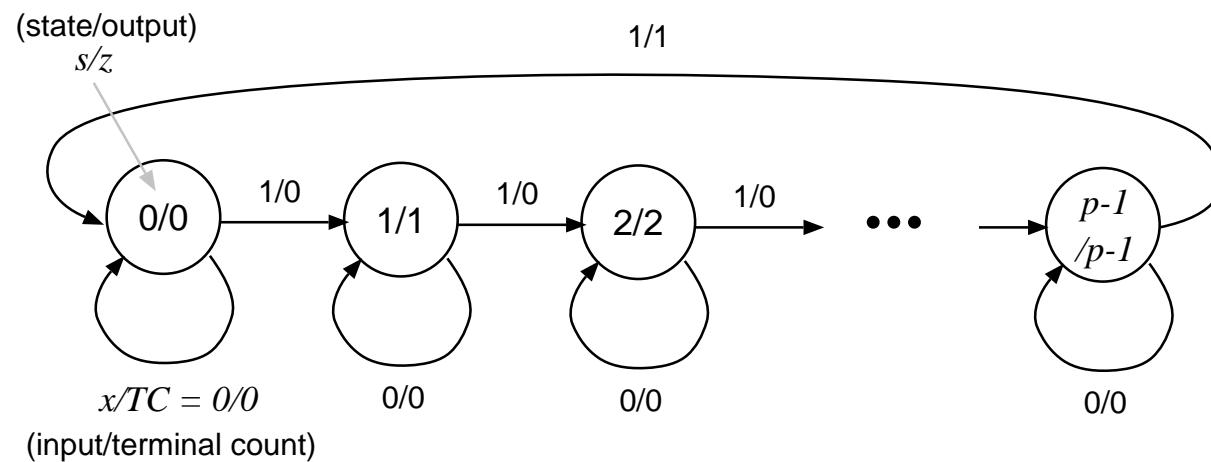


Figure 11.14: STATE DIAGRAM OF A MODULO- p COUNTER

A HIGH-LEVEL DESCRIPTION OF A MODULO- p COUNTER

25

INPUT: $x \in \{0, 1\}$

OUTPUTS: $z \in \{0, 1, \dots, p - 1\}$
 $TC \in \{0, 1\}$

STATE: $s \in \{0, 1, \dots, p - 1\}$

FUNCTION: STATE TRANSITION AND OUTPUT FUNCTIONS

$$s(t + 1) = (s(t) + x) \bmod p$$

$$z(t) = s(t)$$

$$TC(t) = \begin{cases} 1 & \text{if } s(t) = p - 1 \text{ and } x(t) = 1 \\ 0 & \text{otherwise} \end{cases}$$

TYPES OF COUNTERS

- UP or DOWN COUNTERS

State	Binary	BCD	Excess-3	Gray	Ring	Twisted Tail
0	000	0000	0011	000	00000001	0000
1	001	0001	0100	001	00000010	0001
2	010	0010	0101	011	00000100	0011
3	011	0011	0110	010	00001000	0111
4	100	0100	0111	110	00010000	1111
5	101	0101	1000	111	00100000	1110
6	110	0110	1001	101	01000000	1100
7	111	0111	1010	100	10000000	1000
8		1000	1011			
9		1001	1100			

RING and TWISTED-TAIL COUNTERS

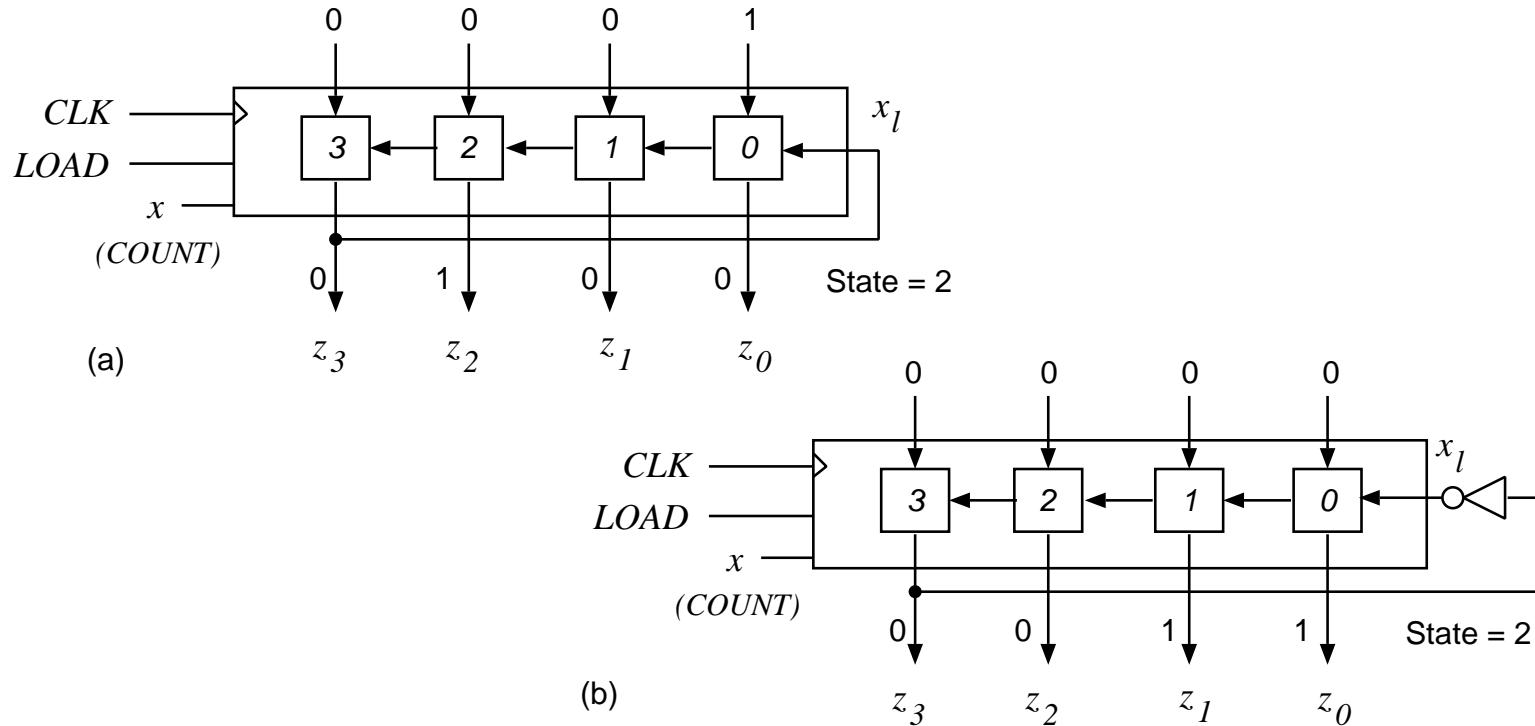


Figure 11.15: a) MODULO-4 RING COUNTER; b) MODULO-8 TWISTED-TAIL COUNTER

BINARY COUNTER WITH PARALLEL INPUT

INPUTS: $\underline{I} = (I_3, \dots, I_0), I_j \in \{0, 1\}, I \in \{0, 1\dots, 15\}$
 $CLR, LD, CNT \in \{0, 1\}$

STATE: $\underline{s} = (s_3, \dots, s_0), s_j \in \{0, 1\}, s \in \{0, 1, \dots, 15\}$

OUTPUT: $\underline{s} = (s_3, \dots, s_0), s_j \in \{0, 1\}, s \in \{0, 1, \dots, 15\}$
 $TC \in \{0, 1\}$

FUNCTION: STATE-TRANSITION AND OUTPUT FUNCTIONS

$$s(t+1) = \begin{cases} 0 & \text{if } CLR = 1 \\ I & \text{if } LD = 1 \\ (s(t) + 1) \bmod 16 & \text{if } CNT = 1 \text{ and } LD = 0 \\ s(t) & \text{otherwise} \end{cases}$$

$$TC = \begin{cases} 1 & \text{if } s(t) = 15 \text{ and } CNT = 1 \\ 0 & \text{otherwise} \end{cases}$$

MODULO-16 COUNTER

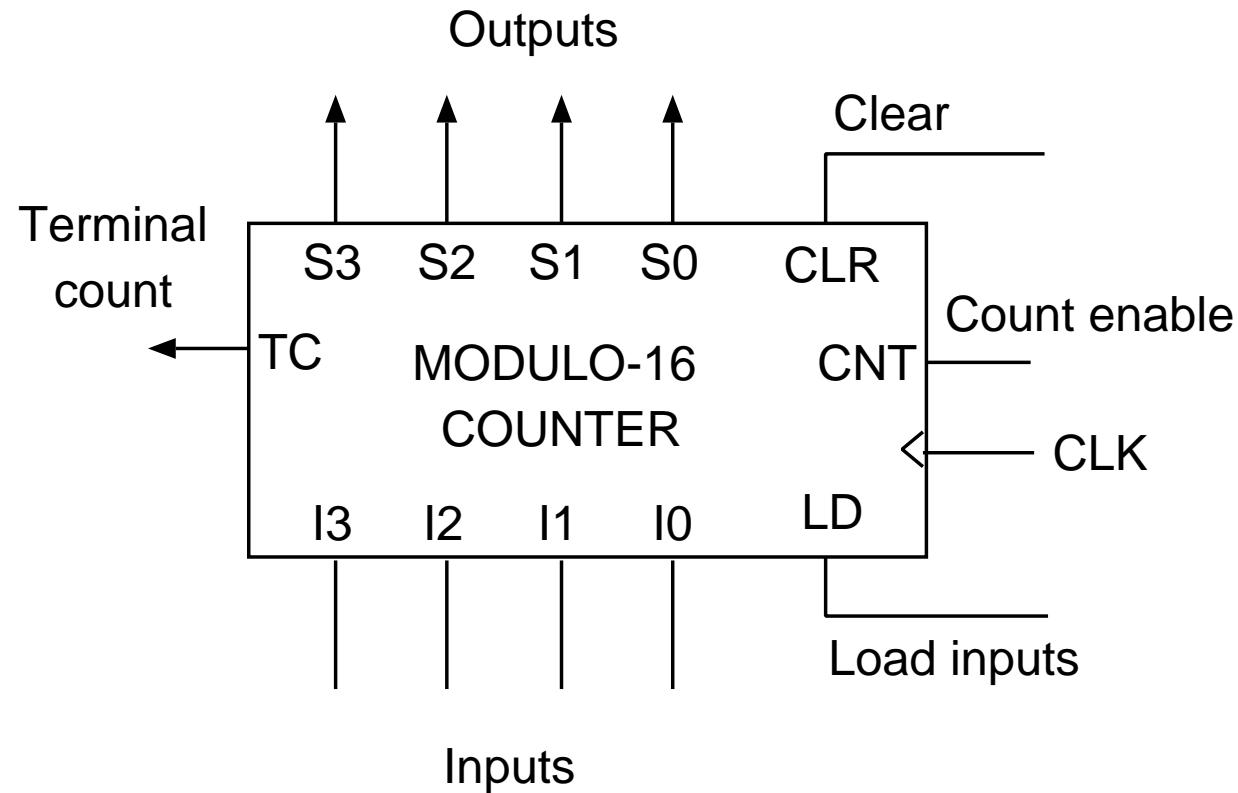


Figure 11.16: A MODULO-16 BINARY COUNTER WITH PARALLEL INPUT

MODULO- k COUNTER ($1 \leq k \leq 16$)(cont.)

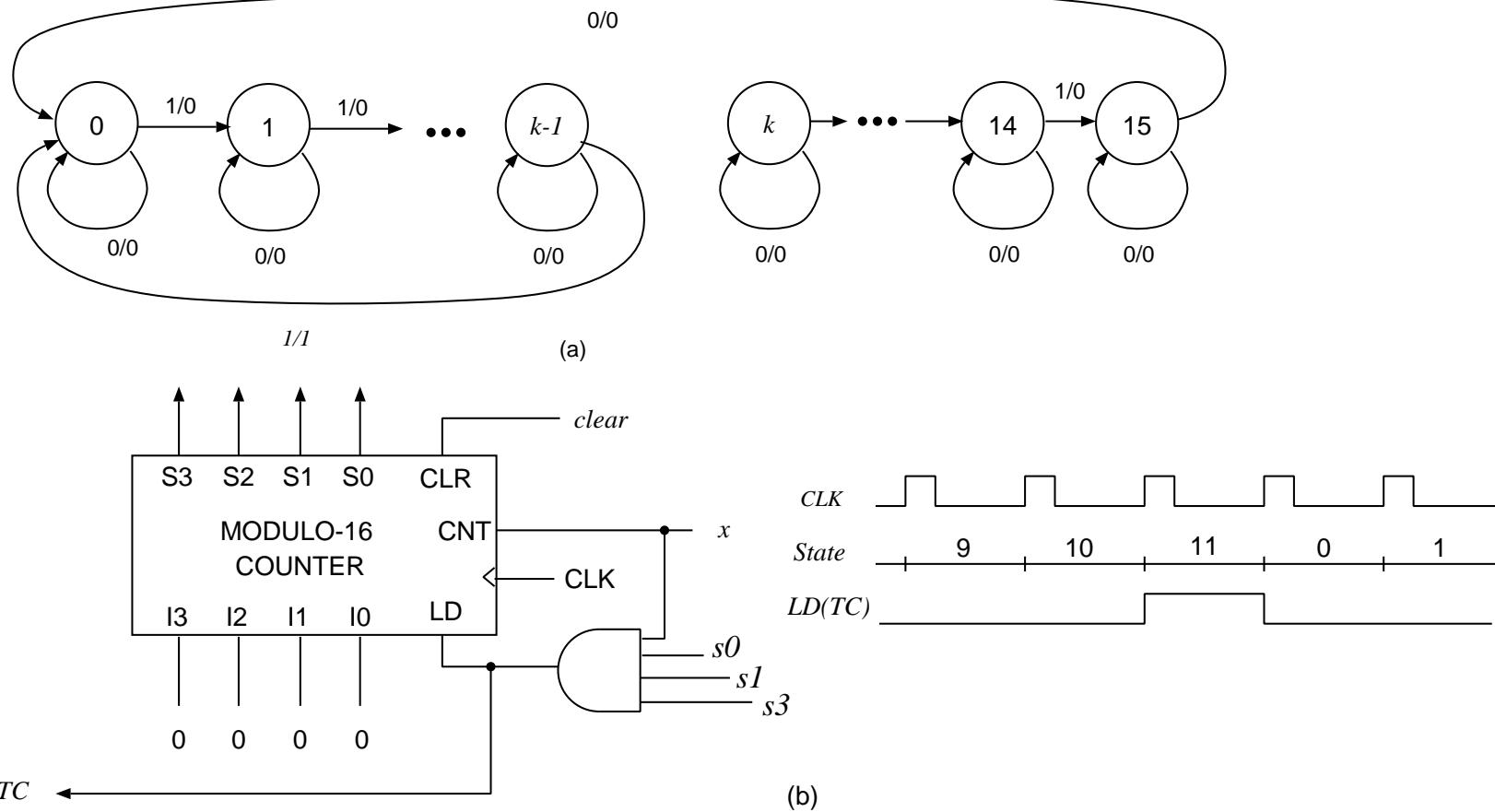


Figure 11.17: a) STATE DIAGRAM OF MODULO- k COUNTER ($1 \leq k \leq 16$); b) MODULO-12 COUNTER AND ITS TIME BEHAVIOR ($x = 1$)

MODULO- k COUNTER ($1 \leq k \leq 16$)

$$CNT = x$$

$$LD = \begin{cases} 1 & \textbf{if } (s = k - 1) \text{ and } (x = 1) \\ 0 & \text{otherwise} \end{cases}$$

$$I = 0$$

$$TC = LD$$

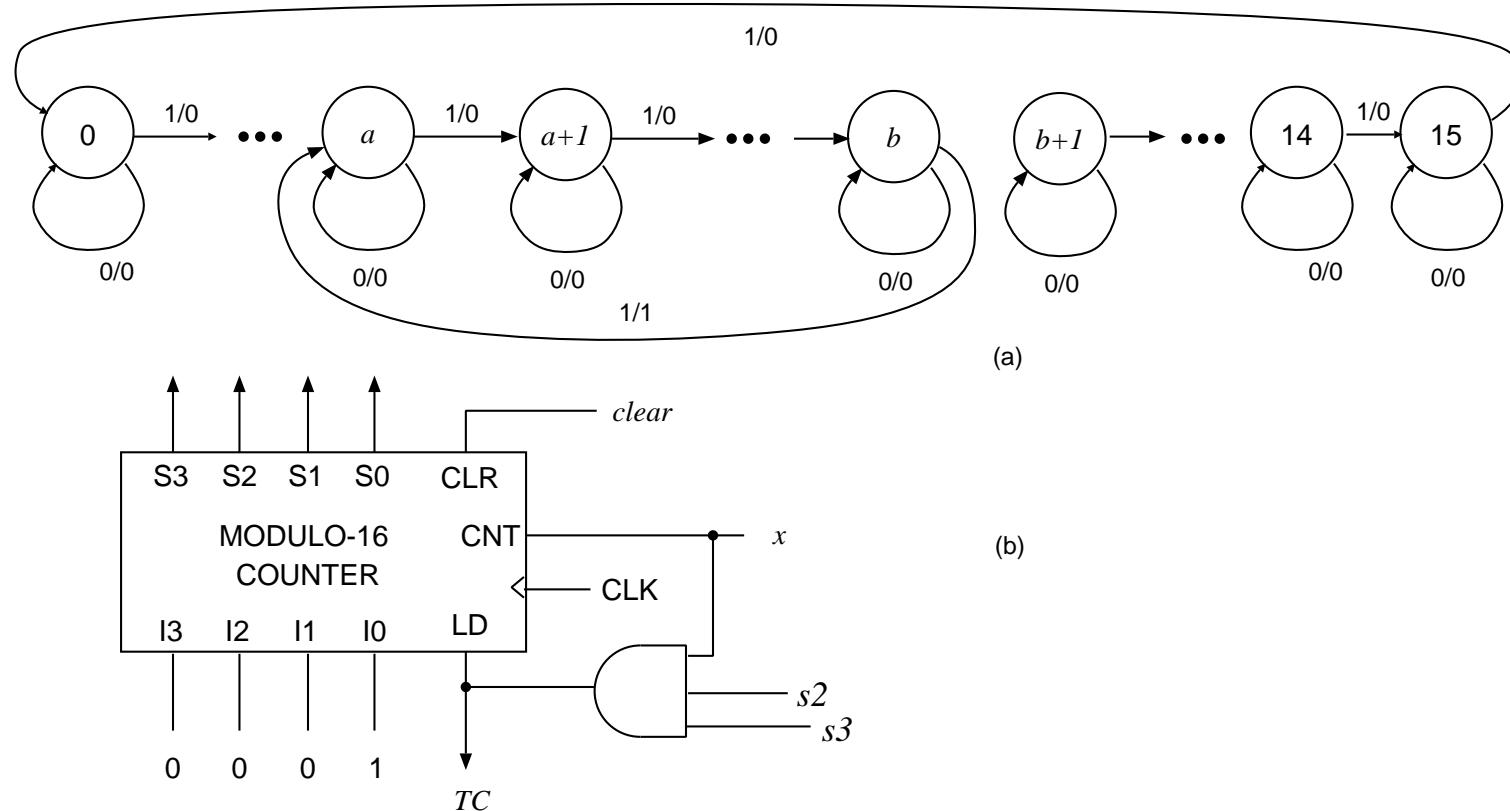


Figure 11.18: a) STATE DIAGRAM OF a -to- b COUNTER; b) A 1-to-12 COUNTER

a -to- b COUNTER ($0 \leq a, b \leq 15$)

$CNT = x$

$$LD = \begin{cases} 1 & \text{if } (s = b) \text{ and } (x = 1) \\ 0 & \text{otherwise} \end{cases}$$

$I = a$

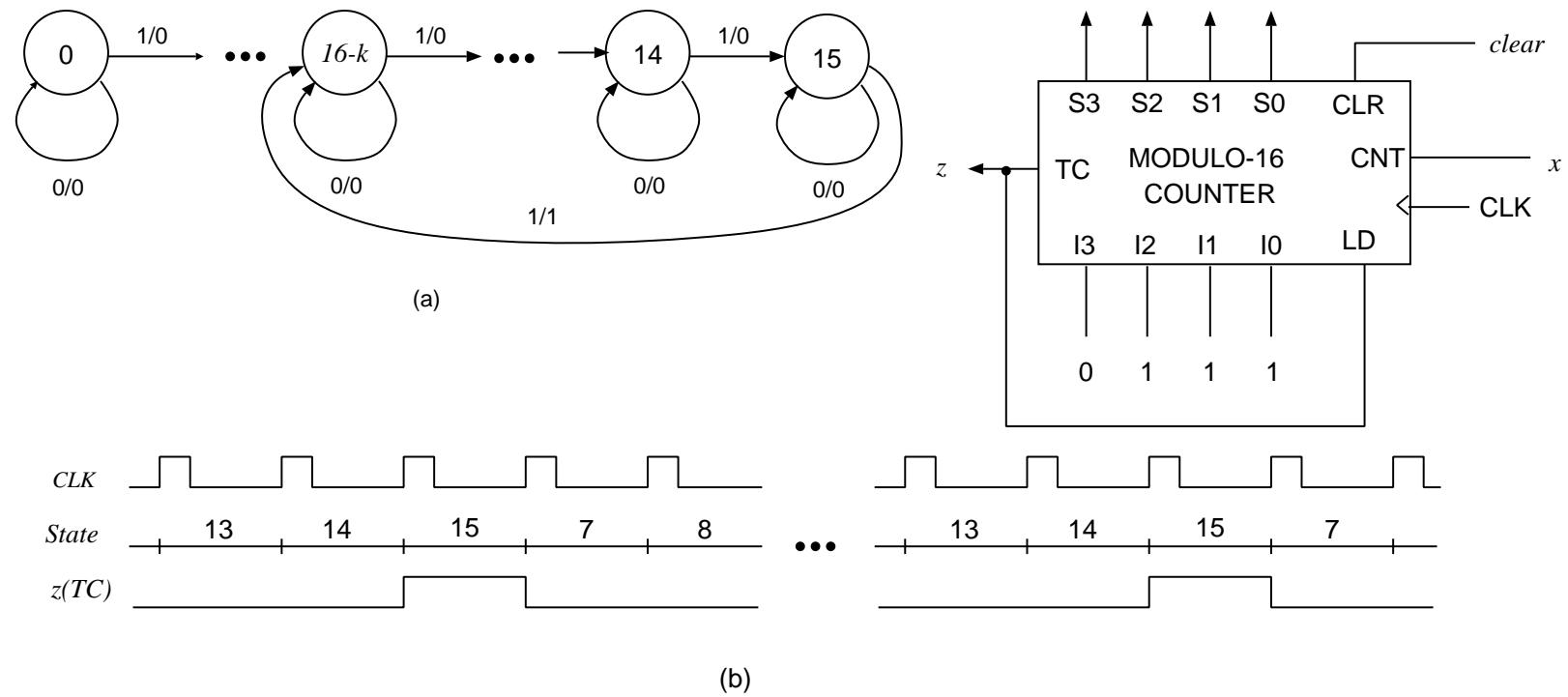


Figure 11.19: a) STATE DIAGRAM OF MODULO- k FREQUENCY DIVIDER; b) MODULO-9 FREQUENCY DIVIDER AND ITS TIME BEHAVIOR ($x = 1$)

MODULO- k FREQUENCY DIVIDER ($1 \leq k \leq 16$)

$$CNT = x$$

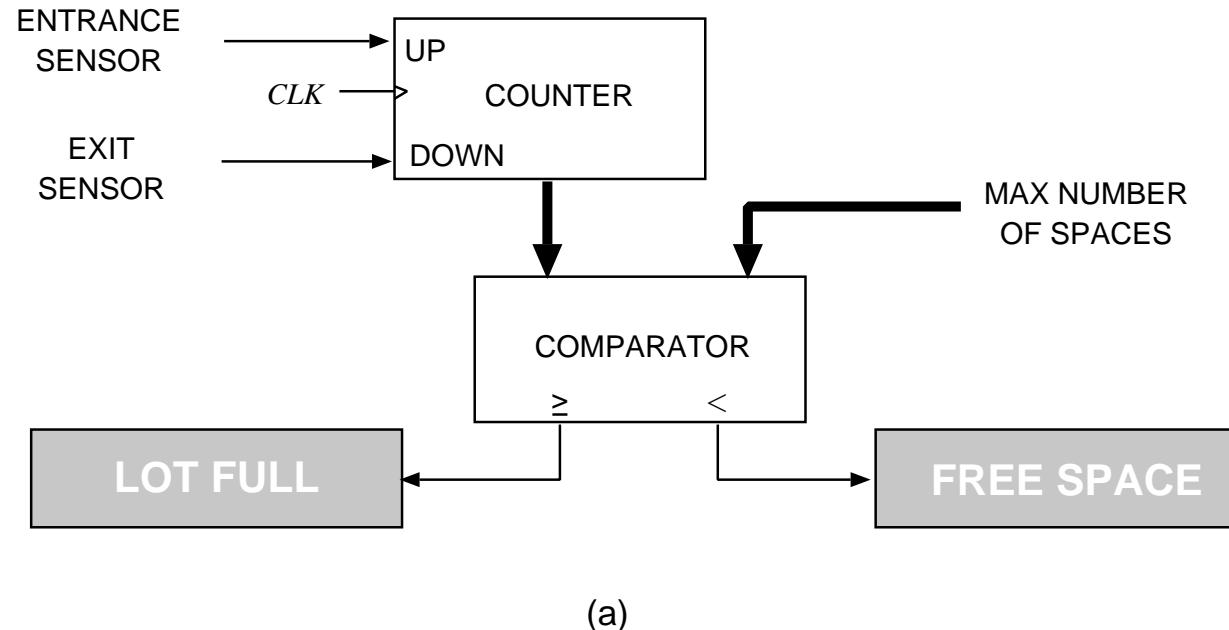
$$LD = \begin{cases} 1 & \textbf{if } TC = 1 \\ 0 & \textbf{otherwise} \end{cases}$$

$$I = 16 - k$$

$$z = TC$$

USES OF COUNTERS

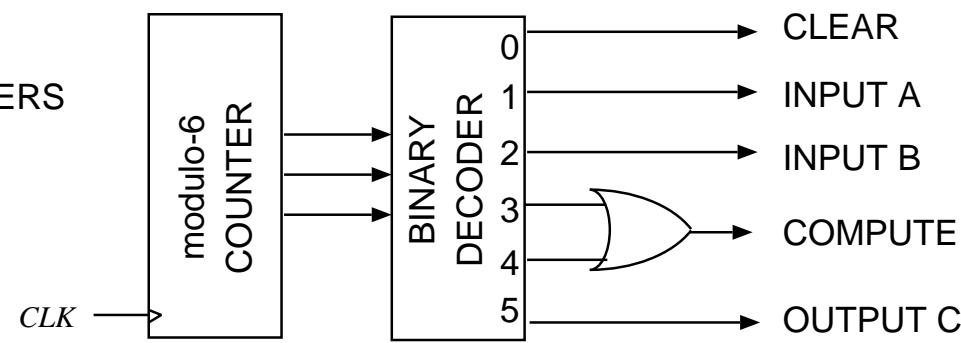
- COUNT THE NUMBER OF TIMES THAT A CERTAIN EVENT TAKES PLACE;
- CONTROL A FIXED SEQUENCE OF ACTIONS
- GENERATE TIMING SIGNALS
- GENERATE CLOCKS OF DIFFERENT FREQUENCIES
- AS STATE REGISTER



(a)

Sequence of actions:

- 0: CLEAR ALL REGISTERS
- 1: INPUT A
- 2: INPUT B
- 3: COMPUTE
- 4: COMPUTE
- 5: OUTPUT C



(b)

Figure 11.20: a) EXAMPLE OF EVENT COUNTER; b) EXAMPLE OF CONTROLLER.

USES OF COUNTERS (cont.)

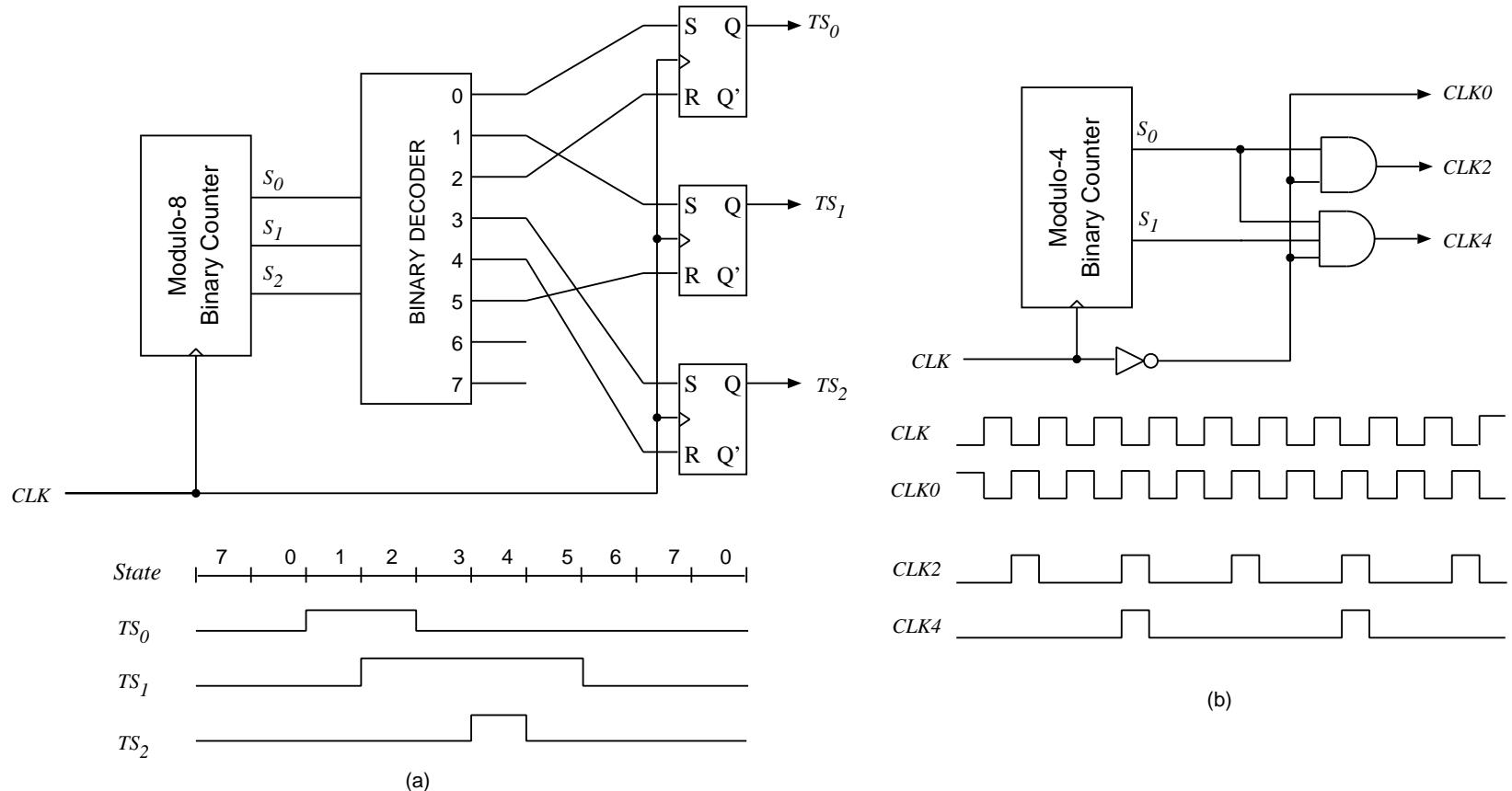
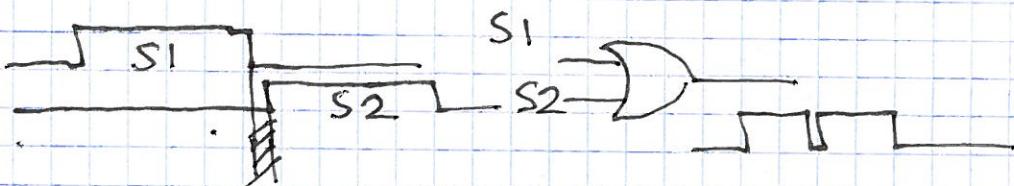
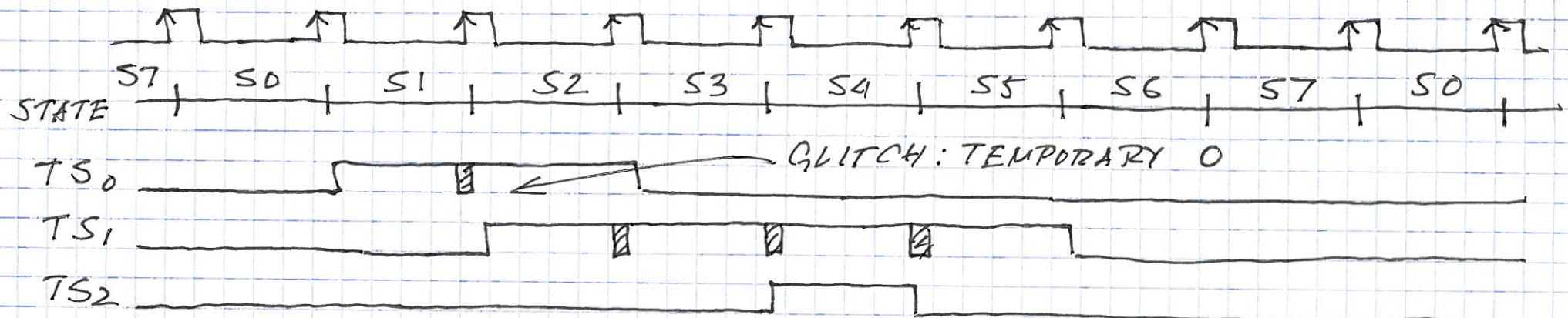
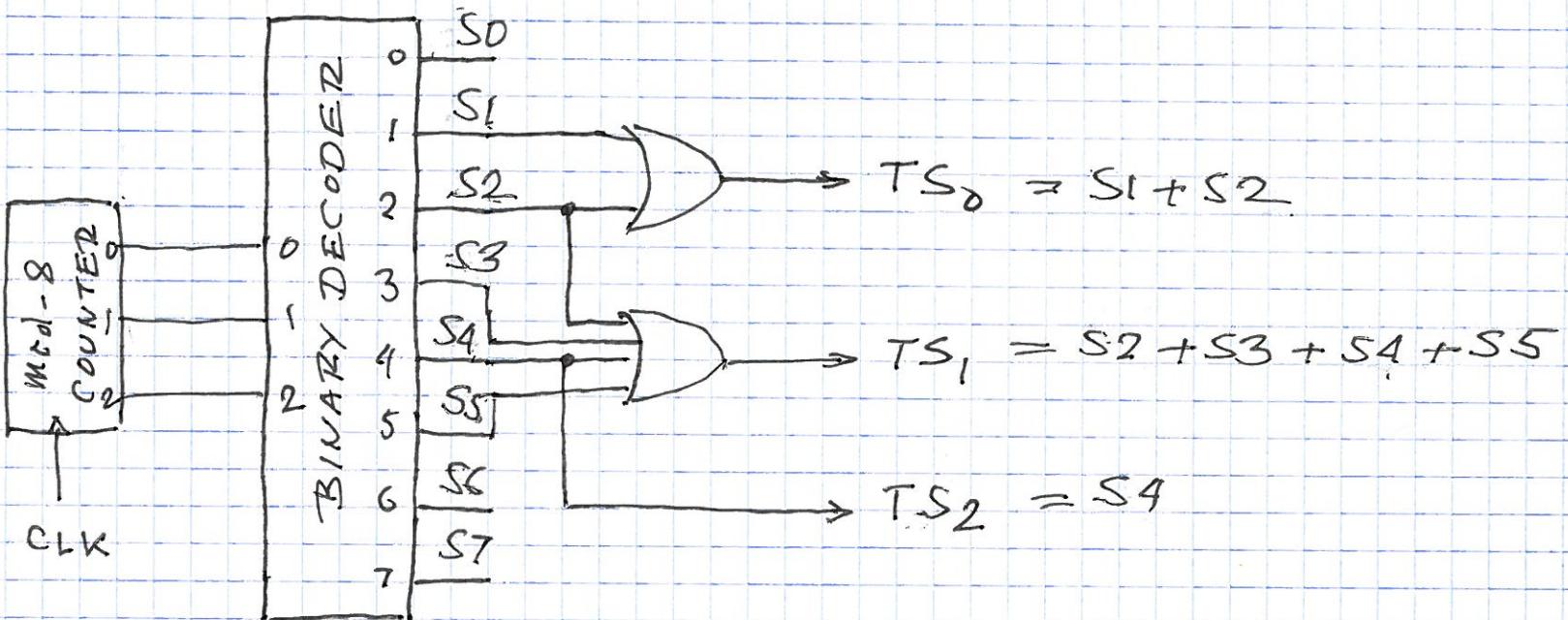
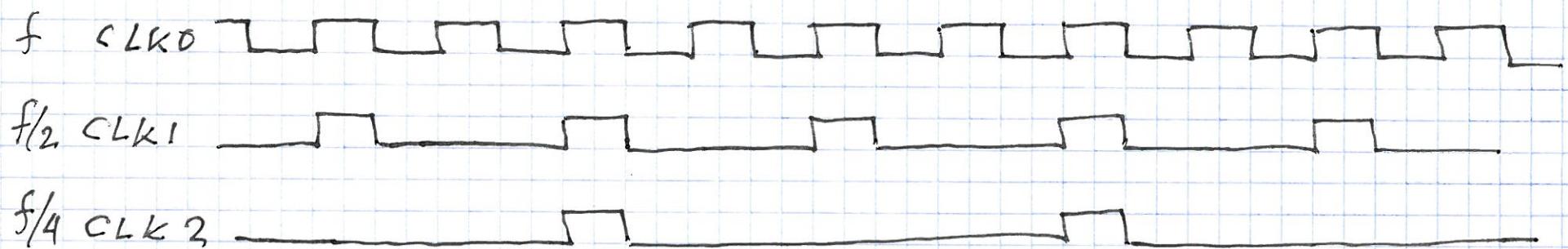
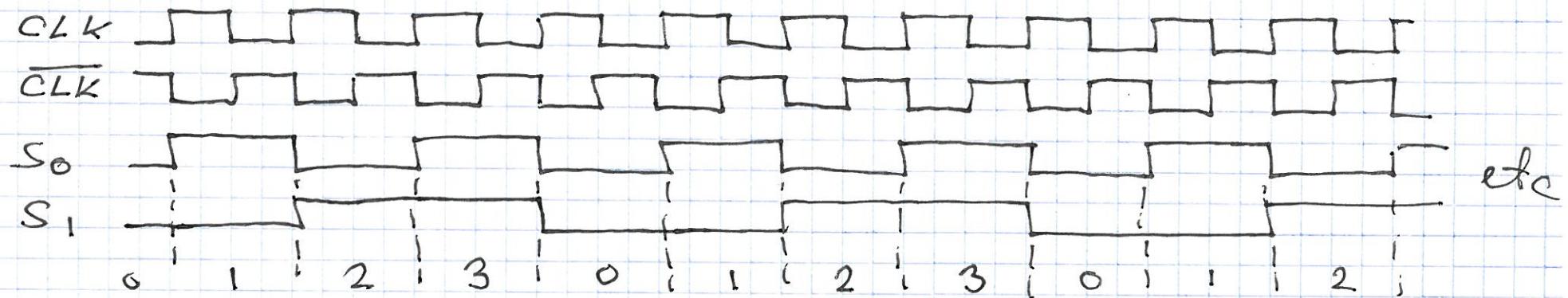


Figure 11.21: EXAMPLES OF NETWORKS FOR GENERATING a) TIMING SIGNALS; b) CLOCKS WITH DIFFERENT FREQUENCIES.

TIMING SIGNAL GENERATOR



CLOCK GENERATOR



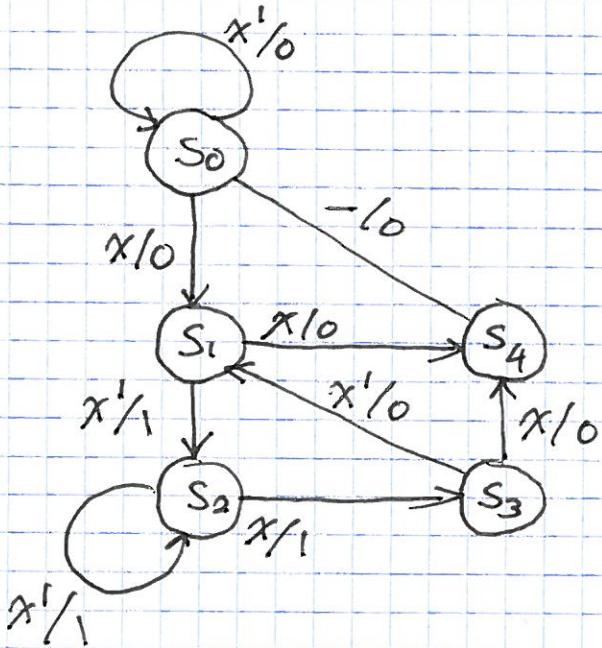
COUNTER AS STATE REGISTER

COUNTING $s(t + 1) = (s(t) + 1) \bmod p$

NO CHANGE $s(t + 1) = s(t)$

ARBITRARY $s(t + 1) \neq (s(t) + 1) \bmod p$ and $s(t + 1) \neq s(t)$

COUNTER AS STATE REGISTER



THREE TRANSITION CASES:

1. COUNTING: $S_0 \xrightarrow{X} S_1 \xrightarrow{X'} S_2 \xrightarrow{X} S_3 \xrightarrow{X'} S_4$
2. NO CHANGE: $S_0 \xrightarrow{X'} S_0; S_2 \xrightarrow{X'} S_2$
3. ARBITRARY: $S_4 \xrightarrow{-1} S_0$

TWO CONTROL SIGNALS

$$\begin{aligned} CNT &= S_0 X + S_1 X' + S_2 X + S_3 X \\ (\text{LOAD}) \quad LD &= CNT' \end{aligned}$$

OR

$$LD = S_4 + S_0 X' + S_2 X'$$

THREE PARALLEL INPUTS

I_2	I_1	I_0	
0	0	0	$(S_0 X' + S_4)$
0	1	0	$S_2 X' \Rightarrow S_2$
0	0	1	$S_3 X' \Rightarrow S_3$
1	0	0	$S_1 X \Rightarrow S_1$

$$\text{OUTPUT } Z = S_1 X' + S_2$$

BECAUSE $LD = 1$ IF $X' = 1$ IN S_2 & S_3
 IF $X = 1$ IN S_1

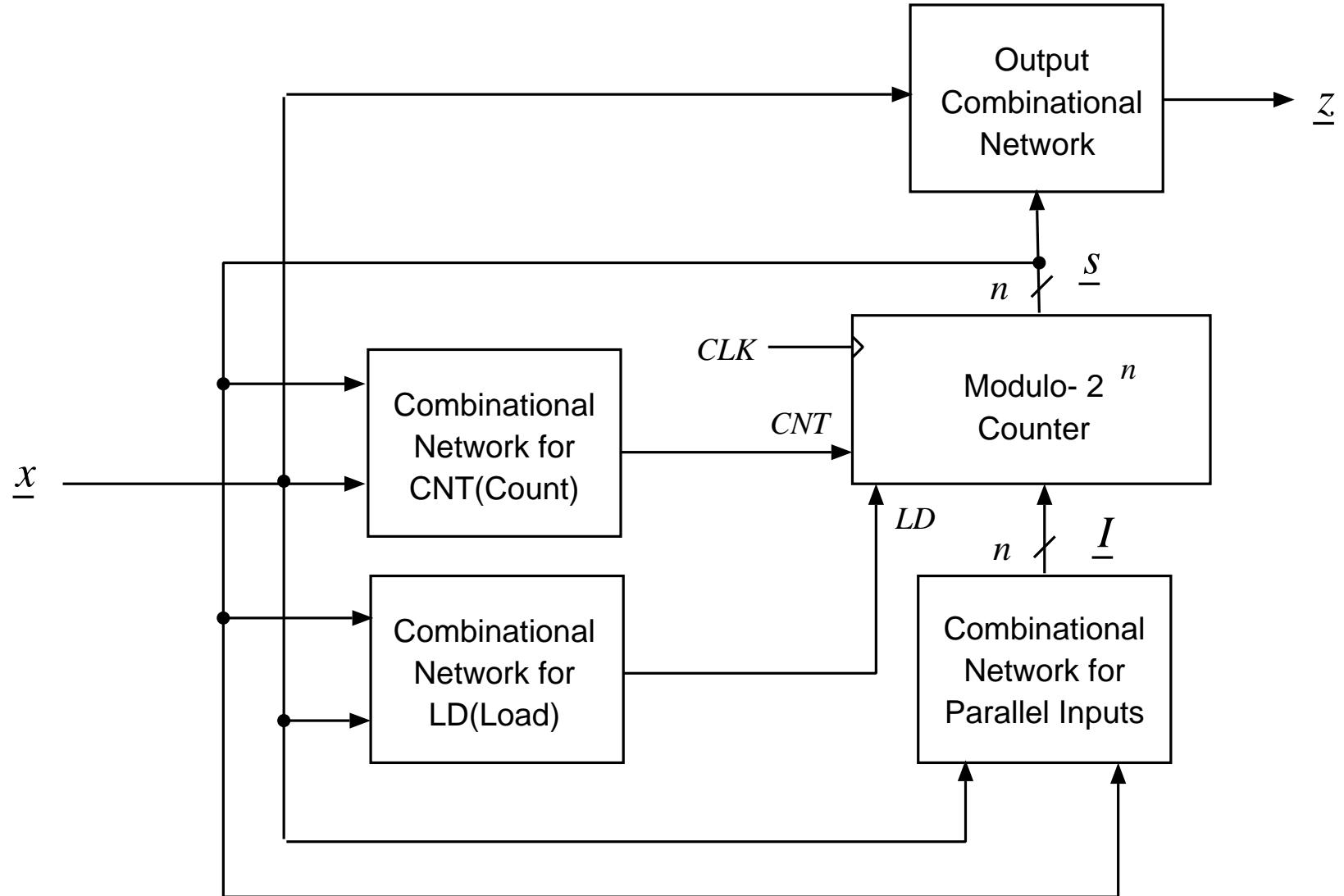


Figure 11.22: IMPLEMENTATION OF SEQUENTIAL SYSTEM WITH COUNTER AND COMBINATIONAL NETWORKS.

COUNTER AS STATE REGISTER (cont.)

$$CNT = \begin{cases} 1 & \text{if } s(t+1) = (s(t) + 1) \bmod p \text{ and } x = 1 \\ 0 & \text{otherwise} \end{cases}$$

$$LD = \begin{cases} 1 & \text{if } s(t+1) \neq s(t) \text{ and} \\ & s(t+1) \neq (s(t) + 1) \bmod p \text{ and } x = 1 \\ 0 & \text{otherwise} \end{cases}$$

$$I = \begin{cases} s(t+1) & \text{if } LD = 1 \\ - & \text{otherwise} \end{cases}$$

S-EXPRESSIONS FOR PARALLEL INPUTS

$$I_2: \begin{array}{c|ccccc} & & & x & \\ \hline & 0 & 0 & | & * & \\ & 0 & 0 & 0 & 0 & \\ & - & - & - & - & \\ & 0 & 0 & | & - & \\ & & & & - & \end{array} \quad \text{BECAUSE } LD=0$$

$Q_1 \quad Q_2 \quad Q_0$

$$I_2 = Q_1' Q_0 = S_1$$

$$I_1: \begin{array}{c|ccccc} & & & x & \\ \hline & 0 & 0 & 0 & 0 & \\ & 1 & * & 0 & 0 & \\ & - & - & - & - & \\ & 0 & 0 & | & - & \\ & & & & - & \end{array}$$

$Q_2 \quad Q_1 \quad Q_0$

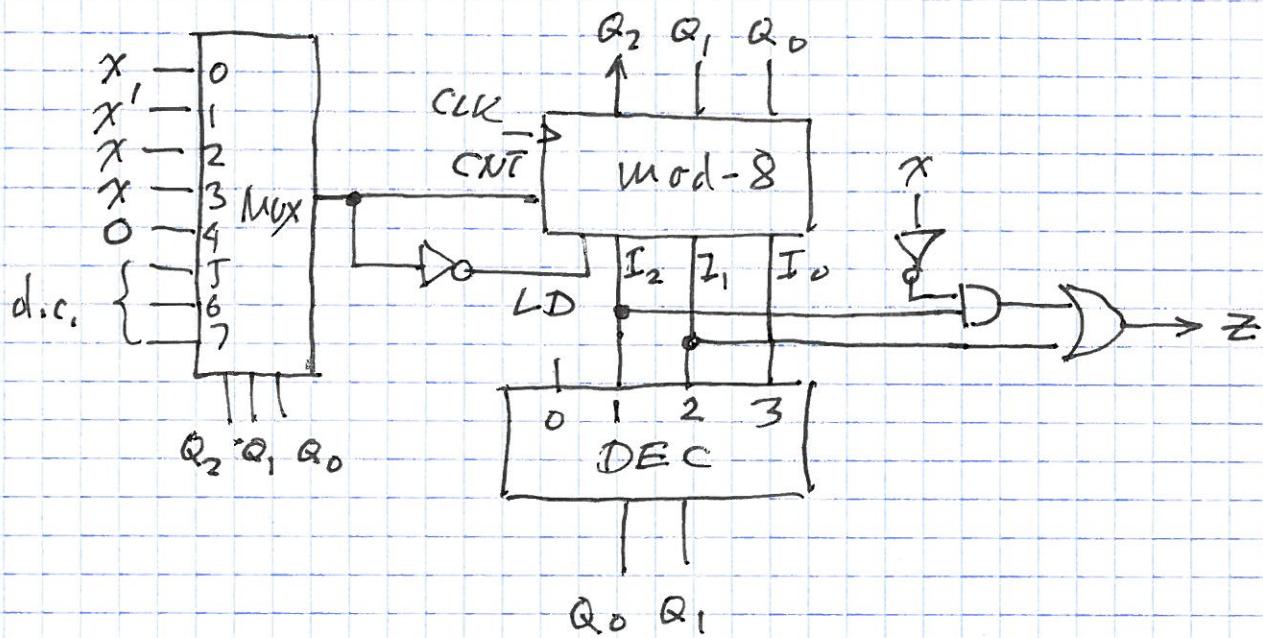
$$I_1 = Q_1 Q_0' = S_2$$

$$I_0: \begin{array}{c|ccccc} & & & x & Q_0 \\ \hline & 0 & 0 & 0 & 0 & \\ & 0 & 0 & * & 1 & \\ & - & - & - & - & \\ & 0 & 0 & | & - & \\ & & & & - & \end{array}$$

$Q_2 \quad Q_1 \quad Q_0$

$$I_0 = Q_1 Q_0 = S_3$$

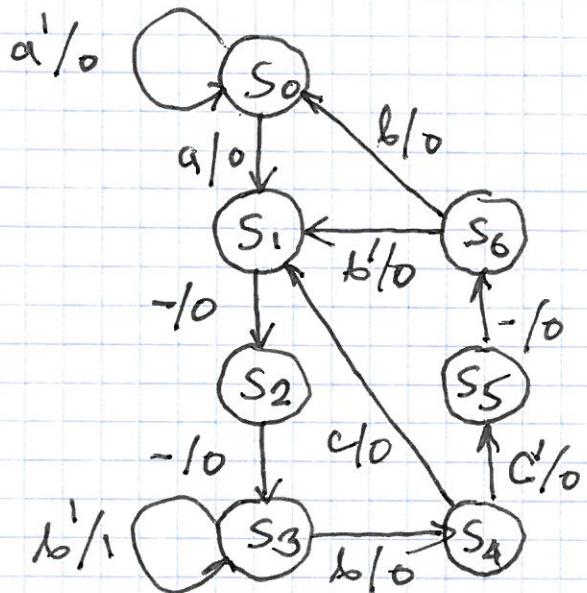
IMPLEMENTATION

 $z:$

				x
				x'
Q_2	Q_1	Q_0	I_2	I_1
0	0	0	1	
1	1	0	0	
-	-	-	-	
0	0	-	A	

$$z = Q_1 Q_0' + Q_1' Q_0 x'$$

EXAMPLE II.4 COUNTER AS STATE REGISTER



COUNTING: $S_0 \xrightarrow{a} S_1 \rightarrow S_2 \rightarrow S_3 \xrightarrow{b} S_4 \xrightarrow{c'} S_5 \rightarrow S_6$

JUMPING: $S_0 \xrightarrow{a'} S_0; S_3 \xrightarrow{b'} S_3$

$S_4 \xrightarrow{c} S_1; S_6 \xrightarrow{b'} S_1; S_6 \xrightarrow{b} S_0$

$$CNT = S_0a + S_1 + S_2 + S_3b + S_4c' + S_5$$

$$LD = \overline{CNT}$$

DETERMINE PARALLEL INPUTS (JUMP STATE)

$$I_3 \ I_2 \ I_1 \ I_0 \quad IF$$

$$\rightarrow S_0 \quad 0 \ 0 \ 0 \ 0 \quad S_0a' + S_6b = S_0 + S_6b \\ (LD \ if \ a')$$

$$\rightarrow S_1 \quad 0 \ 0 \ 0 \ 1 \quad S_4c + S_6b' = S_4 + S_6b'$$

$$\rightarrow S_3 \quad 0 \ 0 \ 1 \ 1 \quad S_3b' = S_3$$

DURING LOAD

S_1, S_2, S_5, S_7 ARE d.c.

STATE MAP:

		Q_2	
		Q_2	
		S_0	S_1
Q_2	S_0	S_0	S_1
	S_1	S_1	S_0
Q_2	S_2	S_2	S_3
	S_3	S_3	S_2
		Q_1	
		Q_1	Q_1

		Q_2	
		Q_0	
		Q_0	Q_1
Q_2	Q_0	0	-
	Q_1	-	-
Q_2	Q_0	0	-
	Q_1	-	-
		Q_1	
		Q_1	Q_1

$$I_1 = Q_0$$

		Q_2	
		Q_0	
		Q_0	Q_1
Q_2	Q_0	0	-
	Q_1	-	-
Q_2	Q_0	1	-
	Q_1	-	b'
		Q_1	
		Q_1	Q_1

$$I_0 = Q_0 + Q_2 Q_1' + Q_2 b'$$

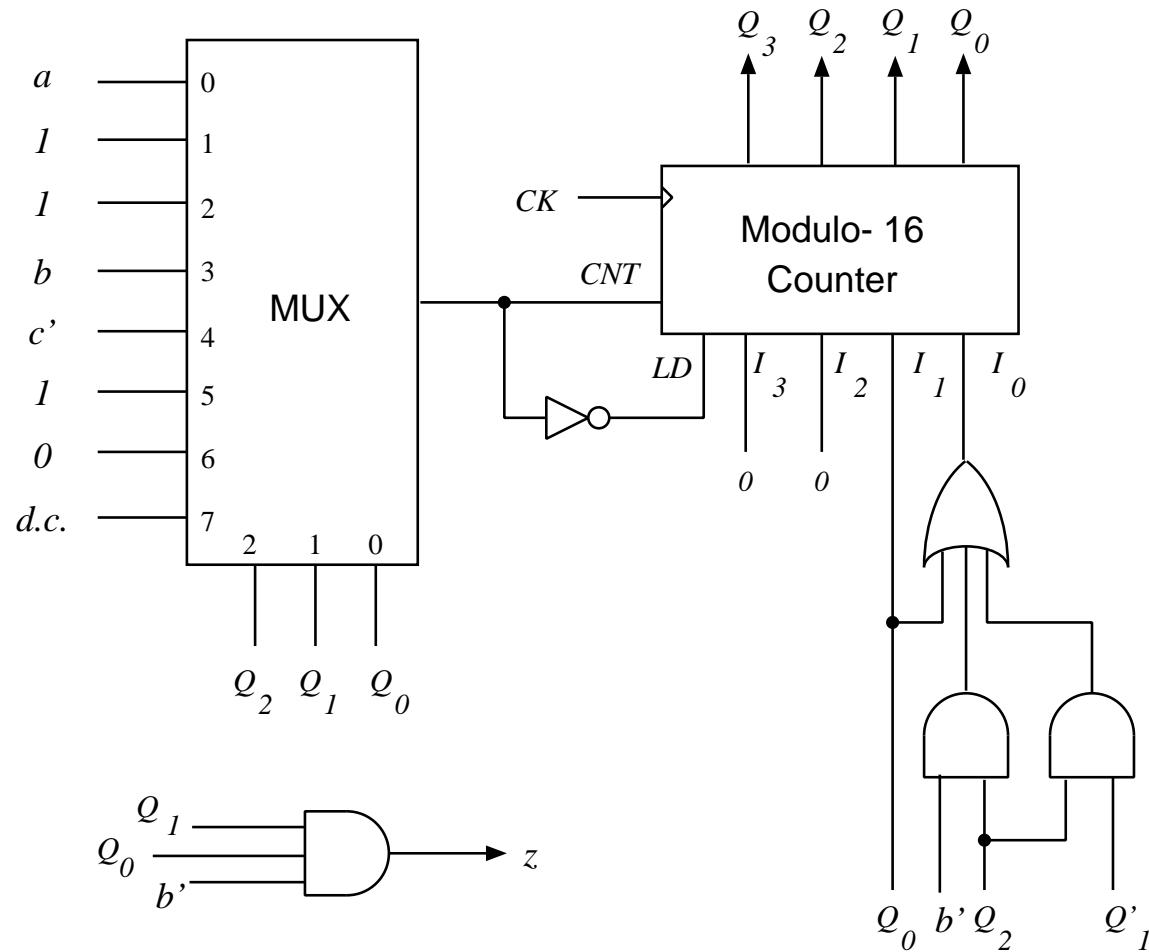


Figure 11.24: SEQUENTIAL NETWORK FOR Example 11.4



- CASCADE COUNTERS

$$TC = \begin{cases} 1 & \text{if } (s = p - 1) \text{ and } (CNT = 1) \\ 0 & \text{otherwise} \end{cases}$$

- FOR THE i -th MODULE

$$CNT^i = \begin{cases} 1 & \text{if } (s^{(j)} = p - 1) \text{ and } (x = 1) \\ & (\text{for all } j < i) \\ 0 & \text{otherwise} \end{cases}$$

where $s^{(j)}$ is the state of counter j .

CASCADE COUNTERS (cont.)

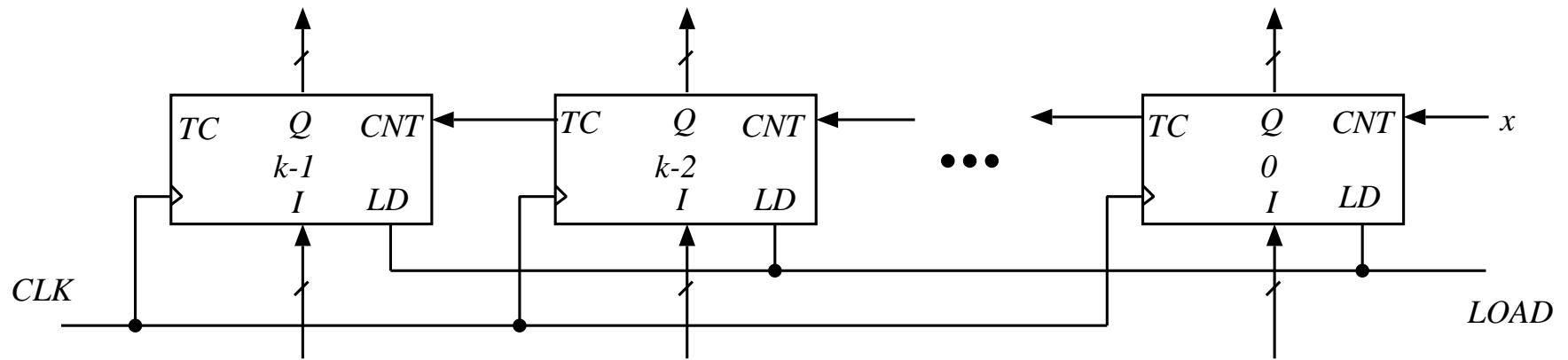


Figure 11.26: CASCADE IMPLEMENTATION OF A MODULO- p^k COUNTER

- THE WORST-CASE DELAY

$$T_{\text{worst-case}} = (k - 1)t_{tc} + t_{su} + t_p$$

- THE MAXIMUM CLOCK FREQUENCY POSSIBLE

$$f_{max} = 1 / [(k - 1)t_{tc} + t_{su} + t_p]$$

CSMSIA

WORST-CASE DELAY OF

CASCADE COUNTER mod p^k

k mod p COUNTERS

WORST CASE: MODULE $k-1$ COUNTS (ASSUME $x=1$)

HAPPENS WHEN MODULES 0 TO $k-2$ ARE

IN STATE $p-1$

ON NEXT CLOCK: GENERATE $TC^{(0)}$

AND PROPAGATE OVER MODULES

1, 2, ... $k-2$

PRODUCE $TC^{(k-1)}$

MODULE $k-1$ READY TO COUNT

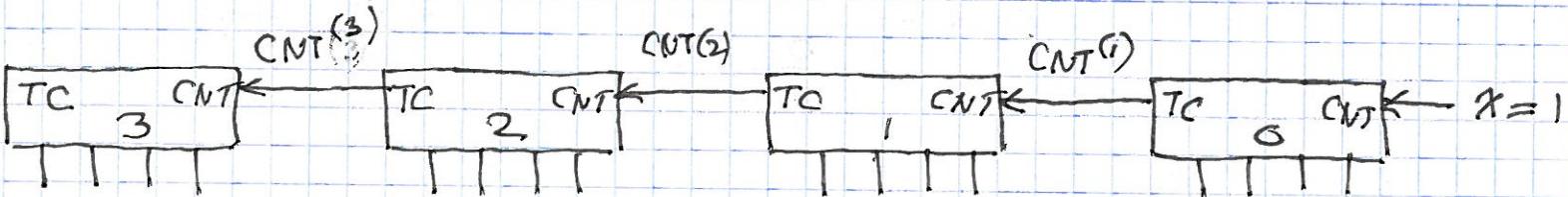
- IN STATE 1

ON NEXT CLOCK: MODULE $k-1$: $1 \rightarrow 1+1$

MODULES 0, ..., $k-1 \rightarrow$ STATE 0

$$k=3$$

$$p=4$$



0 1 0 0

0 1 0 0

0 1 0 0

CNT⁽³⁾

CNT⁽²⁾

CNT⁽¹⁾

CNT⁽³⁾

CNT⁽²⁾

CNT⁽¹⁾

0 0 0 0

0 0 0 0

0 0 0 0

1 1 1 0

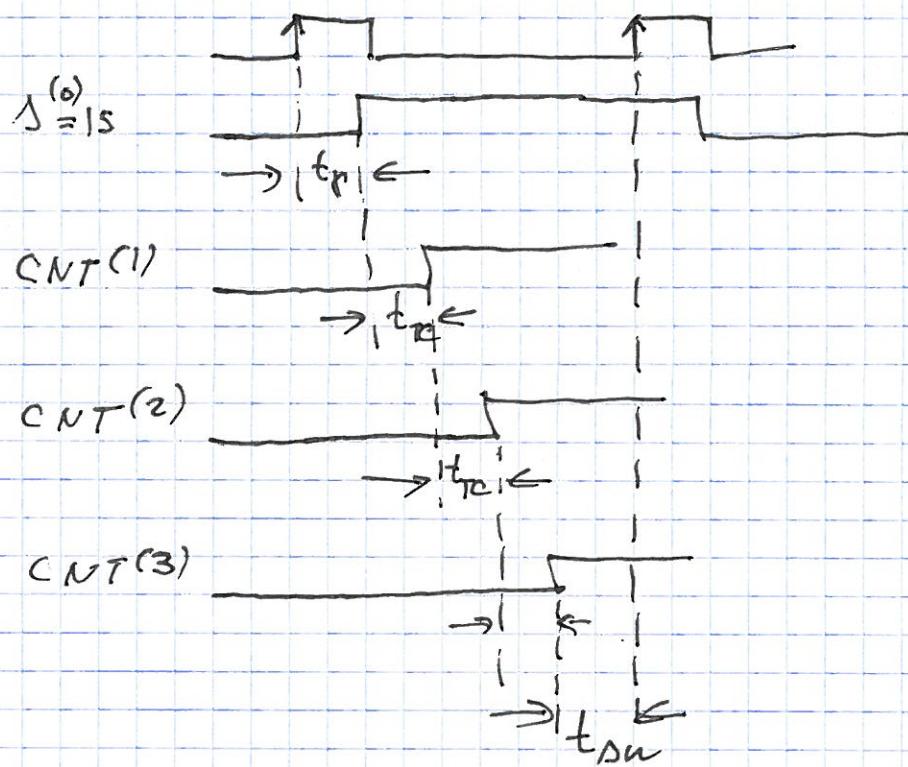
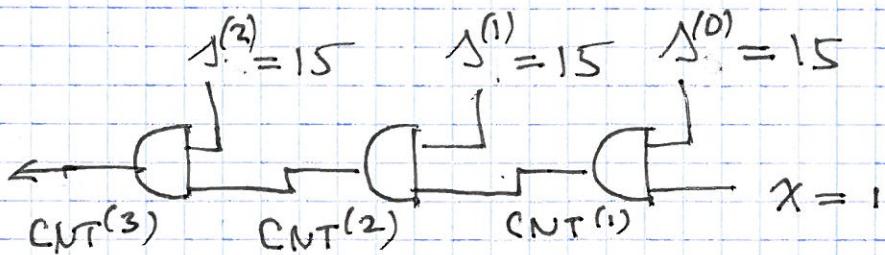
0 0 0 0

CS 651A

TC TRANSMISSION NETWORK

$$TC = \begin{cases} 1 & \text{IF } (j = p-1) \text{ AND } (CNT = 1) \\ 0 & \text{OTHERWISE} \end{cases}$$

FOR $k = 4$



Example 11.5

$t_{su} = 4.5[ns]$ (including the delay of the gates used to produce the inputs to the cells)

$$\begin{aligned}t_p &= 2[ns] \\t_{tc} &= 3.5[ns]\end{aligned}$$

MIN CLOCK PERIOD:

with one module $T = 6.5[ns]$ ($153[MHz]$)

with 8 modules $T = 31[ns]$ ($32[MHz]$)

CASCADE COUNTERS CAN BE MADE FASTER. HOW?

PARALLEL COUNTERS

Modulo-504 counter: $7 \times 8 \times 9$ states

000, 111, 222, 333, 444, 555, 666, 077, 108, 210, 321, 432, ...

PARALLEL MODULO-($7 \times 8 \times 9$) COUNTER

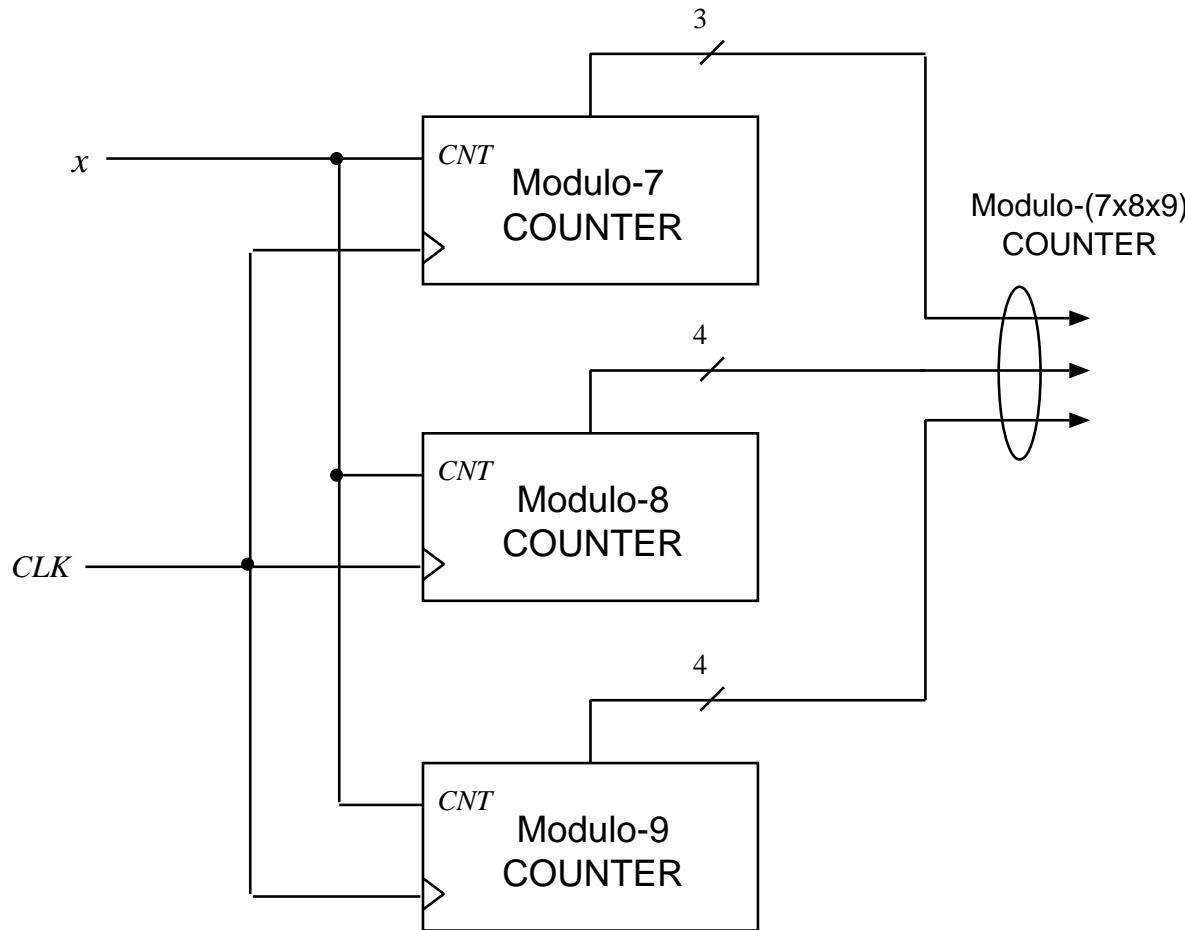


Figure 11.27: PARALLEL IMPLEMENTATION OF MODULO-($7 \times 8 \times 9$) COUNTER.

MULTIMODULE SYSTEMS

- Complex sequential system \Rightarrow several interacting sequential subsystems

Example 11.6: BLOCK PATTERN RECOGNIZER

- INPUT SEQUENCE: blocks of k symbols
- FUNCTION: check for pattern P in each block
- IMPLEMENTATION: counter + recognizer
- OUTPUT OF THE COUNTER:

$$TC(t) = \begin{cases} 1 & \text{if } t \bmod k = k - 1 \text{ and CHECK} = 1 \\ 0 & \text{otherwise} \end{cases}$$

- OUTPUT OF THE SYSTEM: $z(t) = p(t) \cdot TC(t)$

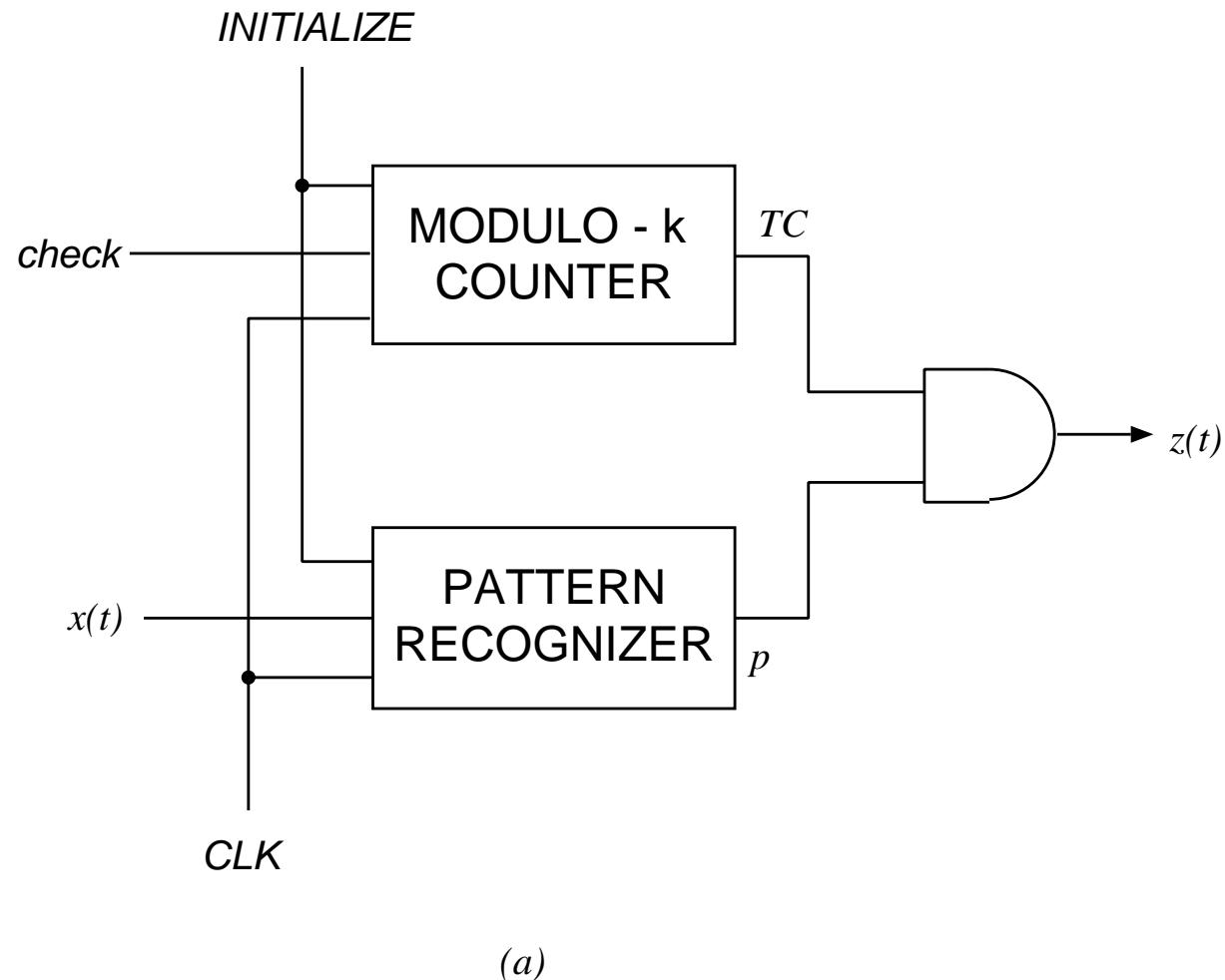


Figure 11.28: BLOCK PATTERN RECOGNIZER

Example 11.6: ILLUSTRATION

t	0	1	2	3	4	5	6	7	8	9	10	11
x	5	3	7	4	1	0	3	7	4	3	7	4
TC	0	0	1	0	0	1	0	0	1	0	0	1
p	0	0	0	1	0	0	0	0	1	0	0	1
z	0	0	0	0	0	0	0	0	1	0	0	1

Example 11.7

- count the number of instances of pattern P in blocks of k symbols

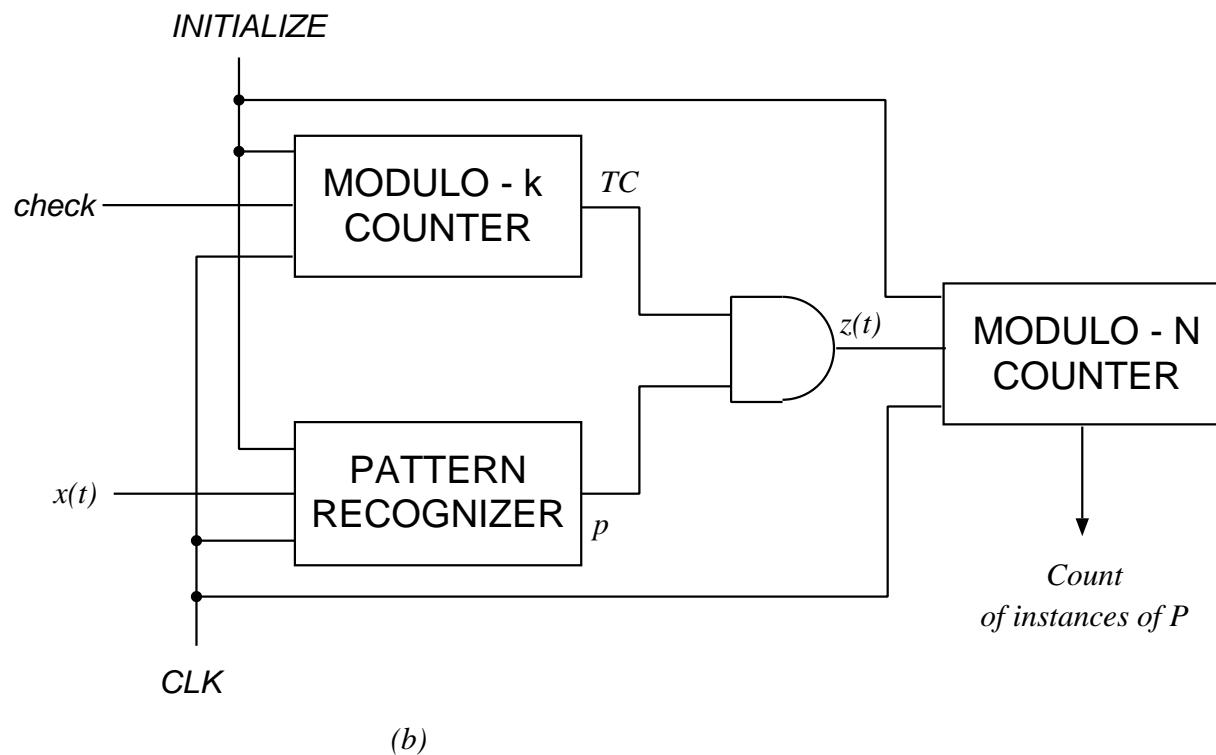


Figure 11.30: BLOCK PATTERN RECOGNIZER.

AN ILLUSTRATIVE DIGITAL DESIGN

PROBLEM: DESIGN A DIGITAL SYSTEM TO ADD TWO INTEGERS IN THE RANGE $[0, 1, \dots, 15]$ WITH CARRY-IN AND CARRY-OUT.

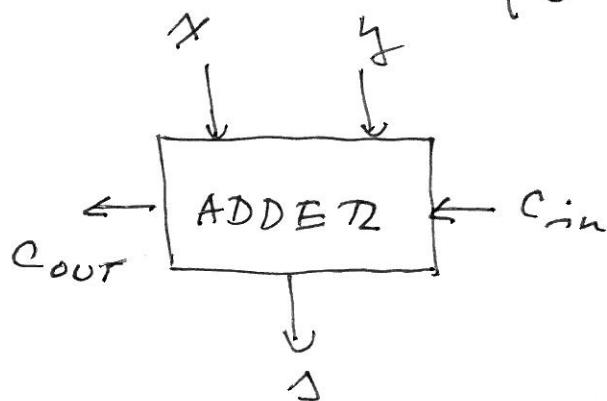
SPECIFICATION

- a) HIGH LEVEL: INPUTS: $x, y \in \{0, \dots, 15\}$
 $c_{in} \in \{0, 1\}$
 OUTPUTS: $s \in \{0, \dots, 15\}$
 $c_{out} \in \{0, 1\}$

FUNCTION:

$$s = \underbrace{(x + y + c_{in})}_{w} \bmod 16$$

$$c_{out} = \begin{cases} 1 & \text{if } w \geq 16 \\ 0 & \text{otherwise} \end{cases}$$



b) BINARY LEVEL:

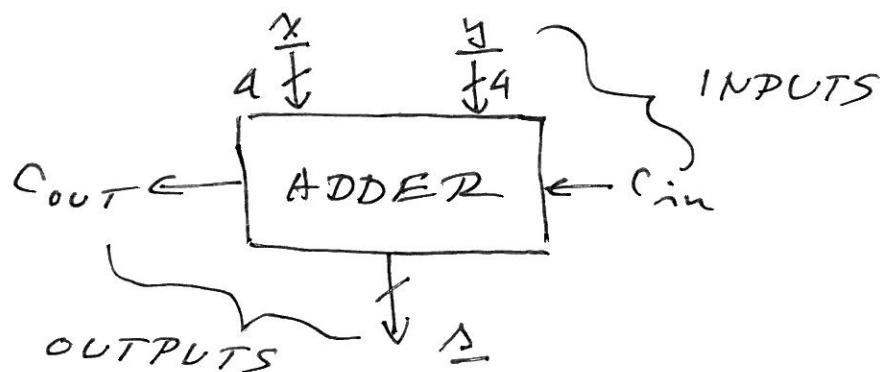
REPRESENT x, y, s AS BIT-VECTORS

$$x \xrightarrow{\text{CODE}} \underline{x} = (x_3, x_2, x_1, x_0) \quad x_i \in \{0, 1\}$$

$$y \rightarrow \underline{y} \quad s \rightarrow \underline{s}$$

SUCH THAT $x = \sum_{i=0}^3 x_i \cdot 2^i$ (BINARY CODE)

E.g. $13_{10} \rightarrow 1101_2$ etc.



FUNCTIONS:

$$s_0 = f_0(x_0, y_0, c_{in})$$

$$s_1 = f_1(x_1, y_1, x_0, y_0, c_{in})$$

$$s_2 = f_2(\dots)$$

$$\begin{matrix} s_3 = f_3(x, y, c_{in}) \\ \downarrow \\ \end{matrix}$$

BINARY
VAR

9 BINARY VARIABLES

$f_3: \text{DOMAIN} \rightarrow \text{RANGE}$

$$\{2^9 \text{ 9-TUPLES}\} \rightarrow \{0, 1\}$$

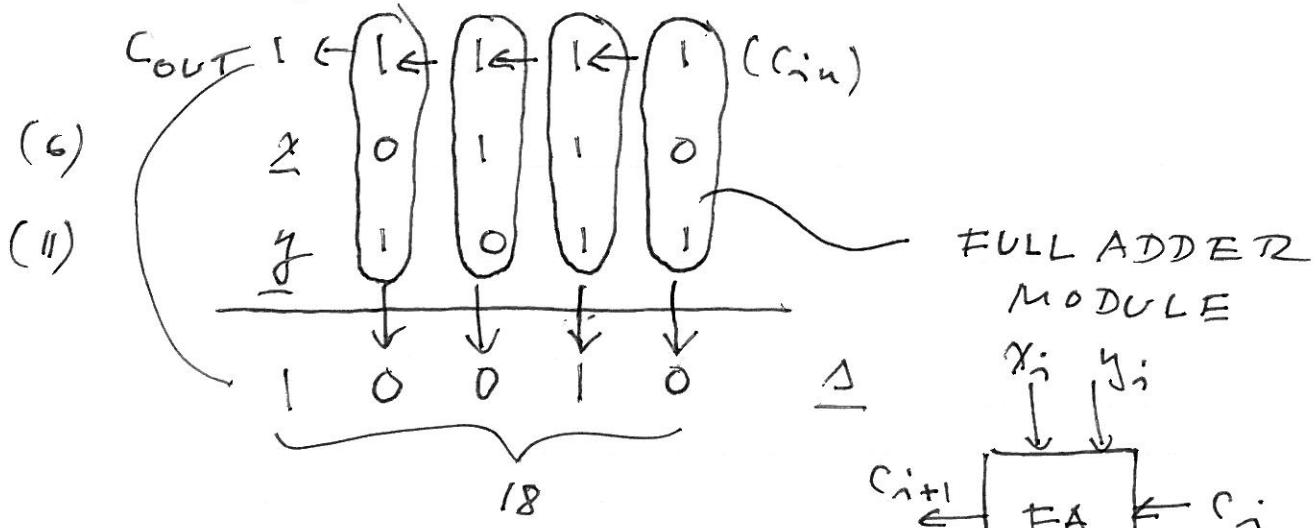
$\Rightarrow f_3$ IS A SWITCHING FUNCTION
A CENTRAL CONCEPT

$$C_{out} = f_{out}(x, y, c_{in})$$

DIFFERENT IMPLEMENTATIONS POSSIBLE,

DIFFER IN COST, DELAY, POWER
(AREA)

CONSIDER A COMMON "PAPER & PENCIL"
METHOD OF ADDITION:



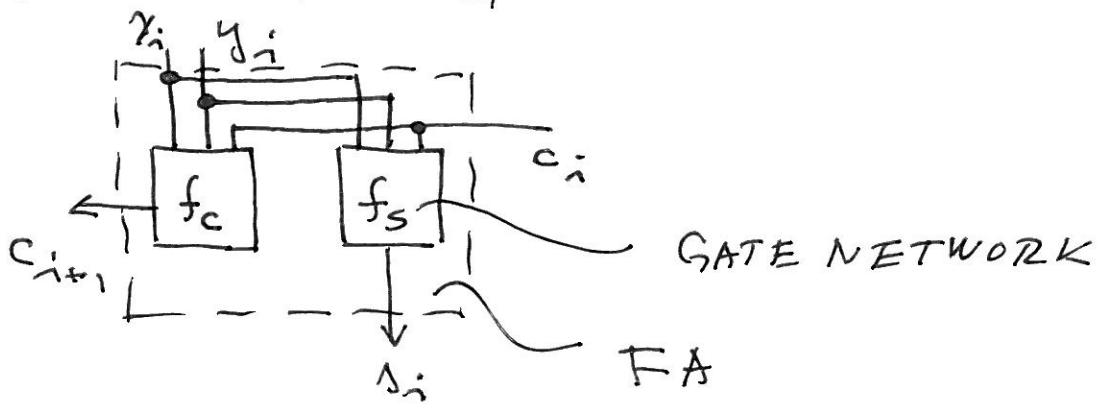
DEFINING ARITH. EXPRESSION:

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

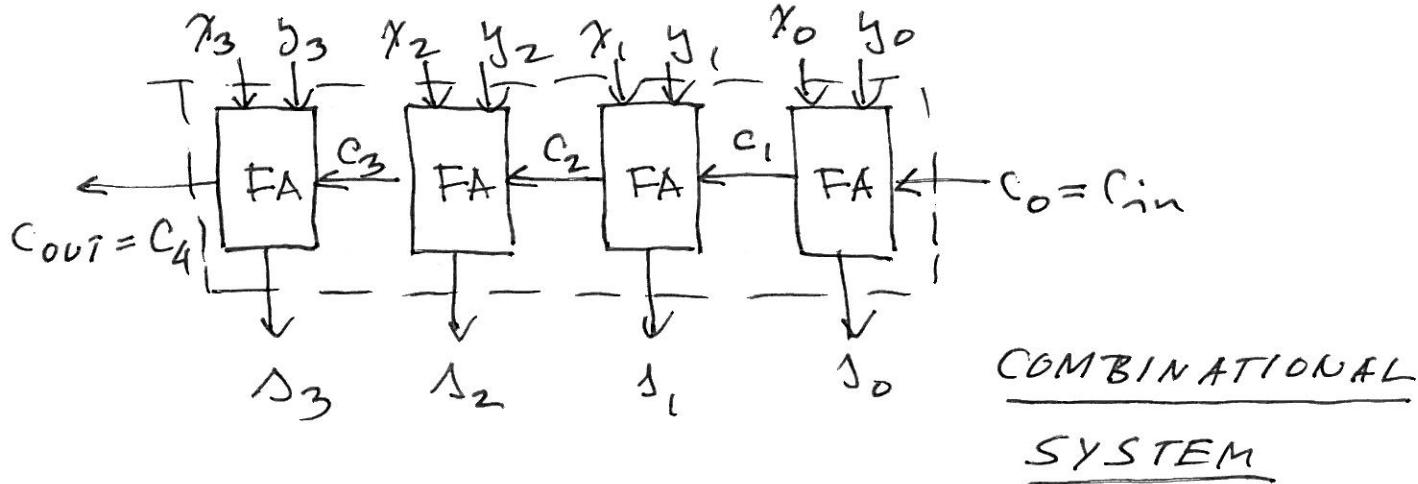
↓ TABLE

Σ	x_i	y_i	c_i	c_{i+1}	s_i	SUM OF INPUTS
0	0	0	0	0	0	\equiv SUM OF OUTPUTS
0	0	1	0	0	1	
0	1	0	0	0	1	
0	1	1	0	1	0	
1	0	0	0	0	1	$s_i = f_s(x_i, y_i, c_i)$
1	0	1	0	1	0	
1	1	0	0	1	0	
1	1	1	1	1	1	$c_{i+1} = f_c(x_i, y_i, c_i)$

→ SIMPLE, FAST



FINAL DESIGN OF 4-BIT ADDER



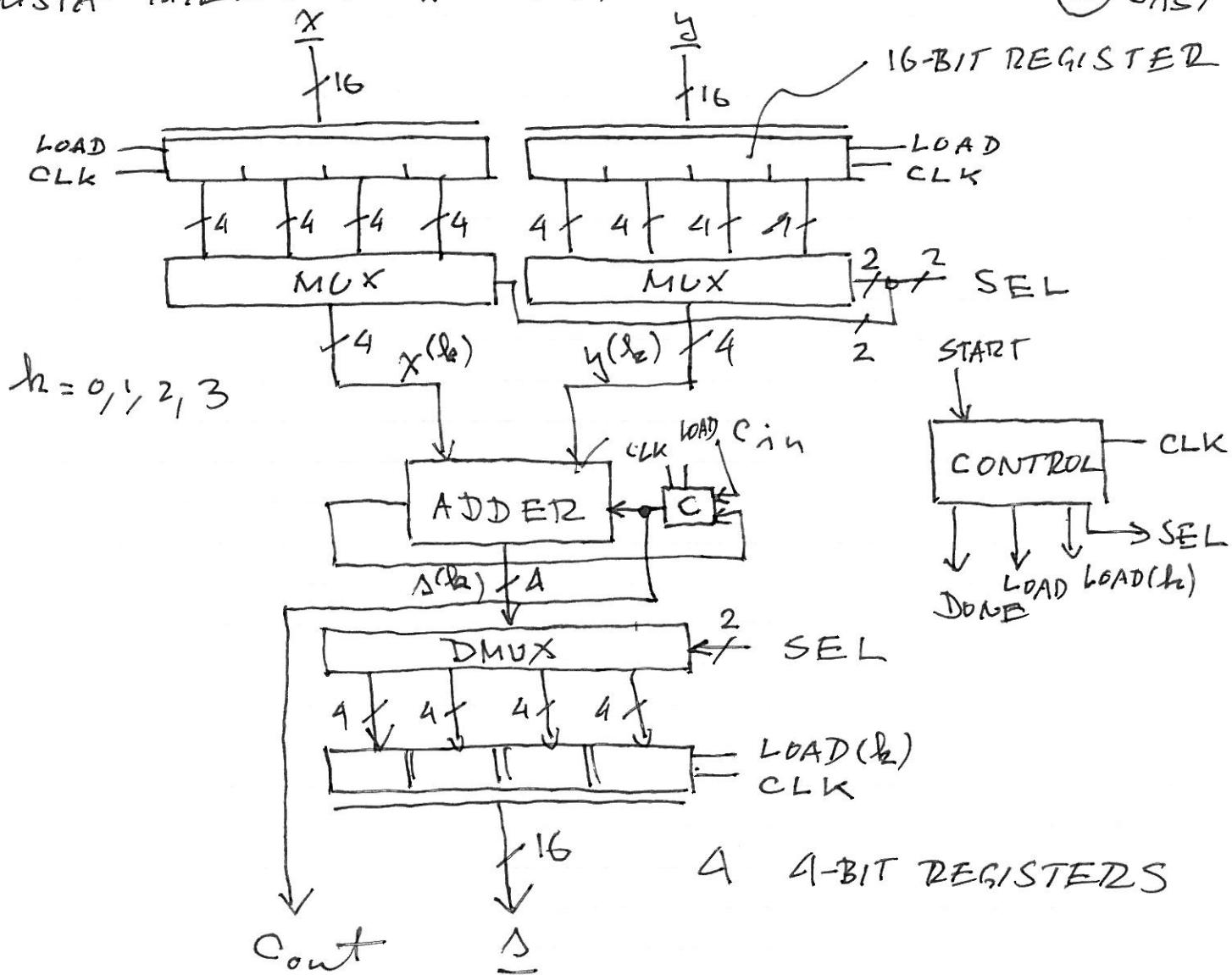
ESTIMATE: COST

DELAY

REPEAT FOR n -BIT ADDER

SUPPOSE YOU NEED TO ADD 16-BIT INTEGERS HAVING ONLY A SINGLE 4-BIT ADDER, REGISTERS, ...

→ DESIGN A SEQUENTIAL SYSTEM



TIME BEHAVIOR:

