Fig. 5-1  Block Diagram of Sequential Circuit
(a) Block diagram

(b) Timing diagram of clock pulses

Fig. 5-2 Synchronous Clocked Sequential Circuit
Fig. 5-3  SR Latch with NOR Gates
(a) Logic diagram  

(b) Function table

**Fig. 5-4** SR Latch with NAND Gates
(a) Logic diagram

(b) Function table

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th>Next state of Q</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>X</td>
<td>X</td>
<td>No change</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>No change</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
<td>$Q = 0$; Reset state</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
<td>$Q = 1$; set state</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>Indeterminate</td>
</tr>
</tbody>
</table>

Fig. 5-5  SR Latch with Control Input
Fig. 5-6  D Latch

(a) Logic diagram

(b) Function table

<table>
<thead>
<tr>
<th>$C$</th>
<th>$D$</th>
<th>Next state of $Q$</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>X</td>
<td>No change</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>$Q = 0$; Reset state</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>$Q = 1$; Set state</td>
</tr>
</tbody>
</table>
Fig. 5-7 Graphic Symbols for Latches
(a) Response to positive level

(b) Positive-edge response

(c) Negative-edge response

Fig. 5-8 Clock Response in Latch and Flip-Flop
Fig. 5-9 Master-Slave $D$ Flip-Flop
Fig. 5-11  Graphic Symbol for Edge-Triggered $D$ Flip-Flop
(a) Circuit diagram

(b) Graphic symbol

Fig. 5-12 JK Flip-Flop

Table 5-1
Flip-Flop Characteristic Tables

<table>
<thead>
<tr>
<th></th>
<th></th>
<th>( Q(t + 1) )</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>( Q(t) )</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>( Q'(t) )</td>
</tr>
</tbody>
</table>

No change
Reset
Set
Complement
Fig. 5-13  T Flip-Flop
Fig. 5-15 Example of Sequential Circuit

\[ A(t + 1) = Ax + Bx \]
\[ B(t + 1) = A'x \]
\[ y = (A + B)x' \]
STATE EQUATIONS OR TRANSITION EQUATIONS

\[ A(t + 1) = Ax + Bx \]
\[ B(t + 1) = A'x \]

OUTPUT BOOLEAN EQUATION

\[ y = (A + B)x' \]

Table 5-2
State Table for the Circuit of Fig. 5-15

<table>
<thead>
<tr>
<th>Present State</th>
<th>Input</th>
<th>Next State</th>
<th>Output</th>
</tr>
</thead>
<tbody>
<tr>
<td>A  B</td>
<td>x</td>
<td>A  B</td>
<td>y</td>
</tr>
<tr>
<td>0  0</td>
<td>0</td>
<td>0  0</td>
<td>0</td>
</tr>
<tr>
<td>0  0</td>
<td>1</td>
<td>0  1</td>
<td>0</td>
</tr>
<tr>
<td>0  1</td>
<td>0</td>
<td>0  0</td>
<td>1</td>
</tr>
<tr>
<td>0  1</td>
<td>1</td>
<td>1  1</td>
<td>0</td>
</tr>
<tr>
<td>1  0</td>
<td>0</td>
<td>0  0</td>
<td>1</td>
</tr>
<tr>
<td>1  0</td>
<td>1</td>
<td>0  0</td>
<td>0</td>
</tr>
<tr>
<td>1  1</td>
<td>0</td>
<td>1  0</td>
<td>0</td>
</tr>
<tr>
<td>1  1</td>
<td>1</td>
<td>1  0</td>
<td>0</td>
</tr>
</tbody>
</table>

Table 5-3
Second Form of the State Table

<table>
<thead>
<tr>
<th>Present State</th>
<th>Next State</th>
<th>Output</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>x = 0   x = 1</td>
<td></td>
</tr>
<tr>
<td>A  B</td>
<td>AB      AB</td>
<td>y</td>
</tr>
<tr>
<td>00</td>
<td>00      01</td>
<td>0</td>
</tr>
<tr>
<td>01</td>
<td>00      11</td>
<td>1</td>
</tr>
<tr>
<td>10</td>
<td>00      10</td>
<td>1</td>
</tr>
<tr>
<td>11</td>
<td>00      10</td>
<td>0</td>
</tr>
</tbody>
</table>

*Note: Table continues with additional rows and columns.*
MEALY FINITE STATE MACHINE (FSM) – OUTPUT IS A FUNCTION OF PRESENT STATE AND INPUT
1. FIND THE STATE TABLE
2. DRAW THE STATE DIAGRAM
(a) Circuit diagram

(b) State table

<table>
<thead>
<tr>
<th>Present state</th>
<th>Inputs</th>
<th>Next state</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>x</td>
<td>y</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>

(c) State diagram

Fig. 5-17 Sequential Circuit with D Flip-Flop
A sequential circuit with two D flip-flops, $A$ and $B$; two inputs, $x$ and $y$; and one output, $z$, is specified by the following next-state and output equations:

$$A(t + 1) = x'y + xA$$
$$B(t + 1) = x'B + xA$$
$$z = B$$

(a) Draw the logic diagram of the circuit.  
(b) List the state table for the sequential circuit.  
(c) Draw the corresponding state diagram.
Fig. 5-18  Sequential Circuit with JK Flip-Flop
flip-flop input equations

\[ J_A = B \quad K_A = Bx' \]
\[ J_B = x' \quad K_B = A'x + Ax' = A \oplus x \]

**Table 5-4**
State Table for Sequential Circuit with JK Flip-Flops

<table>
<thead>
<tr>
<th>Present State</th>
<th>Input</th>
<th>Next State</th>
<th>Flip-Flop Inputs</th>
</tr>
</thead>
<tbody>
<tr>
<td>( A )</td>
<td>( B )</td>
<td>( x )</td>
<td>( A )</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>
Fig. 5-19 State Diagram of the Circuit of Fig. 5-18
MOORE FSM
OUTPUT IS A FUNCTION
OF PRESENT STATE ONLY

(a) Circuit diagram
(b) State diagram

Fig. 5-20 Sequential Circuit with $T$ Flip-Flops
A sequential circuit has two $JK$ flip-flops $A$ and $B$, two inputs $x$ and $y$, and one output $z$. The flip-flop input equations and circuit output equation are

$$J_A = Bx + B'y'$$
$$K_A = B'xy'$$
$$J_B = A'x$$
$$K_B = A + xy'$$
$$z = Ax'y' + Bx'y'$$

(a) Draw the logic diagram of the circuit.  
(b) Tabulate the state table.  
(c) Derive the state equations for $A$ and $B$.  


DESIGN PROCEDURE

1. From the word description and specifications of the desired operation, derive a state diagram for the circuit.
2. Reduce the number of states if necessary.
3. Assign binary values to the states.
4. Obtain the binary-coded state table.
5. Choose the type of flip-flops to be used.
6. Derive the simplified flip-flop input equations and output equations.
7. Draw the logic diagram.
States are application-dependent. The names given here (A, B, C, D, ...) are arbitrary. It is assumed that only the output response to a given sequence of inputs is important.
Table 5-6
State Table

<table>
<thead>
<tr>
<th>Present State</th>
<th>( x = 0 )</th>
<th>( x = 1 )</th>
<th>( x = 0 )</th>
<th>( x = 1 )</th>
</tr>
</thead>
<tbody>
<tr>
<td>( a )</td>
<td>( a )</td>
<td>( b )</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>( b )</td>
<td>( c )</td>
<td>( d )</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>( c )</td>
<td>( a )</td>
<td>( d )</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>( d )</td>
<td>( e )</td>
<td>( f )</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>( e )</td>
<td>( a )</td>
<td>( f )</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>( f )</td>
<td>( g )</td>
<td>( f )</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>( g )</td>
<td></td>
<td>( f )</td>
<td>0</td>
<td>1</td>
</tr>
</tbody>
</table>

An algorithm for the state reduction of a completely specified state table is given here without proof: “Two states are said to be equivalent if, for each member of the set of inputs, they give exactly the same output and send the circuit either to the same state or to an equivalent state.” When two states are equivalent, one of them can be removed without altering the input–output relationships.
### Table 5-6
**State Table**

<table>
<thead>
<tr>
<th>Present State</th>
<th>Next State</th>
<th>Output</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>$x = 0$</td>
<td>$x = 1$</td>
</tr>
<tr>
<td>$a$</td>
<td>$a$</td>
<td>$b$</td>
</tr>
<tr>
<td>$b$</td>
<td>$c$</td>
<td>$d$</td>
</tr>
<tr>
<td>$c$</td>
<td>$a$</td>
<td>$d$</td>
</tr>
<tr>
<td>$d$</td>
<td>$e$</td>
<td>$f$</td>
</tr>
<tr>
<td>$e$</td>
<td>$a$</td>
<td>$f$</td>
</tr>
<tr>
<td>$f$</td>
<td>$g$</td>
<td>$f$</td>
</tr>
<tr>
<td>$g$</td>
<td>$a$</td>
<td>$f$</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Present State</th>
<th>Output</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>$x = 0$</td>
</tr>
<tr>
<td>$a$</td>
<td>$0$</td>
</tr>
<tr>
<td>$b$</td>
<td>$0$</td>
</tr>
<tr>
<td>$c$</td>
<td>$0$</td>
</tr>
<tr>
<td>$d$</td>
<td>$0$</td>
</tr>
<tr>
<td>$e$</td>
<td>$0$</td>
</tr>
<tr>
<td>$f$</td>
<td>$0$</td>
</tr>
</tbody>
</table>

### Table 5-7
**Reducing the State Table**

<table>
<thead>
<tr>
<th>Present State</th>
<th>Next State</th>
<th>Output</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>$x = 0$</td>
<td>$x = 1$</td>
</tr>
<tr>
<td>$a$</td>
<td>$a$</td>
<td>$b$</td>
</tr>
<tr>
<td>$b$</td>
<td>$c$</td>
<td>$d$</td>
</tr>
<tr>
<td>$c$</td>
<td>$a$</td>
<td>$d$</td>
</tr>
<tr>
<td>$d$</td>
<td>$e$</td>
<td>$f$</td>
</tr>
<tr>
<td>$e$</td>
<td>$a$</td>
<td>$f$</td>
</tr>
<tr>
<td>$f$</td>
<td>$e$</td>
<td>$f$</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Present State</th>
<th>Output</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>$x = 0$</td>
</tr>
<tr>
<td>$a$</td>
<td>$0$</td>
</tr>
<tr>
<td>$b$</td>
<td>$0$</td>
</tr>
<tr>
<td>$c$</td>
<td>$0$</td>
</tr>
<tr>
<td>$d$</td>
<td>$0$</td>
</tr>
<tr>
<td>$e$</td>
<td>$0$</td>
</tr>
<tr>
<td>$f$</td>
<td>$0$</td>
</tr>
</tbody>
</table>
### Table 5-8
**Reduced State Table**

<table>
<thead>
<tr>
<th>Present State</th>
<th>Next State</th>
<th>Output</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>$x = 0$</td>
<td>$x = 1$</td>
</tr>
<tr>
<td>$a$</td>
<td>$a$</td>
<td>$b$</td>
</tr>
<tr>
<td>$b$</td>
<td>$c$</td>
<td>$d$</td>
</tr>
<tr>
<td>$c$</td>
<td>$a$</td>
<td>$d$</td>
</tr>
<tr>
<td>$d$</td>
<td>$e$</td>
<td>$d$</td>
</tr>
<tr>
<td>$e$</td>
<td>$a$</td>
<td>$d$</td>
</tr>
</tbody>
</table>

![Reduced State Diagram](image)

*Fig. 5-23  Reduced State Diagram*
# State Assignment

## Table 5-9

*Three Possible Binary State Assignments*

<table>
<thead>
<tr>
<th>State</th>
<th>Assignment 1</th>
<th>Assignment 2</th>
<th>Assignment 3</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Binary</td>
<td>Gray code</td>
<td>One-hot</td>
</tr>
<tr>
<td>a</td>
<td>000</td>
<td>000</td>
<td>000001</td>
</tr>
<tr>
<td>b</td>
<td>001</td>
<td>001</td>
<td>00010</td>
</tr>
<tr>
<td>c</td>
<td>010</td>
<td>011</td>
<td>00100</td>
</tr>
<tr>
<td>d</td>
<td>011</td>
<td>010</td>
<td>01000</td>
</tr>
<tr>
<td>e</td>
<td>100</td>
<td>110</td>
<td>10000</td>
</tr>
</tbody>
</table>

## Table 5-10

*Reduced State Table with Binary Assignment 1*

<table>
<thead>
<tr>
<th>Present State</th>
<th>Next State</th>
<th>Output</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>$x = 0$</td>
<td>$x = 1$</td>
</tr>
<tr>
<td>000</td>
<td>000</td>
<td>001</td>
</tr>
<tr>
<td>001</td>
<td>010</td>
<td>011</td>
</tr>
<tr>
<td>010</td>
<td>000</td>
<td>011</td>
</tr>
<tr>
<td>011</td>
<td>100</td>
<td>011</td>
</tr>
<tr>
<td>100</td>
<td>000</td>
<td>011</td>
</tr>
</tbody>
</table>
SEQUENCE DETECTOR: CIRCUIT THAT DETECTS 3 CONSECUTIVE 1’S IN A STRING OF BITS COMING THROUGH THE INPUT LINE

Fig. 5-24 State Diagram for Sequence Detector
Fig. 5-25  Maps for Sequence Detector

\[ D_A = Ax + Bx \]

\[ D_B = Ax + B'x \]

\[ y = AB \]
Fig. 5-26 Logic Diagram of Sequence Detector
**Table 5-12**
*Flip-Flop Excitation Tables*

<table>
<thead>
<tr>
<th>$Q(t)$</th>
<th>$Q(t+1)$</th>
<th>$J$</th>
<th>$K$</th>
<th>$Q(t)$</th>
<th>$Q(t+1)$</th>
<th>$T$</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>$X$</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
<td>$X$</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>$X$</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>$X$</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
</tbody>
</table>

(a) $JK$

(b) $T$
Table 5-13
State Table and JK Flip-Flop Inputs

<table>
<thead>
<tr>
<th>Present State</th>
<th>Input x</th>
<th>Next State</th>
<th>Flip-Flop Inputs</th>
</tr>
</thead>
<tbody>
<tr>
<td>$A$</td>
<td>$B$</td>
<td></td>
<td>$A$</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
</tbody>
</table>
$J_A = Bx'$

$J_B = x$

$K_A = Bx$

$K_B = (A \oplus x)'$

Fig. 5-27  Maps for J and K Input Equations
STATE ASSIGNMENT GUIDELINES

- Assign neighboring codes if states have the same
  - Next State (G1)
  - Previous State (G2)
  - Outputs (G3)
- Prioritize state combinations for which G1, G2, G3 apply more than once
SEQUENCE DETECTOR FOR 010 OR 1001

\[ s_3 \oplus s_5 \text{ ARE EQUIVALENT} \]

<table>
<thead>
<tr>
<th>present</th>
<th>next ( x=0 )</th>
<th>output ( x=0 )</th>
<th>output ( x=1 )</th>
</tr>
</thead>
<tbody>
<tr>
<td>s_0</td>
<td>s_1 s_4</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>s_1</td>
<td>s_1 s_2</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>s_2</td>
<td>s_3 s_4</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>s_3</td>
<td>s_6 s_2</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>s_4</td>
<td>s_5 s_4</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>s_5</td>
<td>s_6 s_2</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>s_6</td>
<td>s_1 s_2</td>
<td>0</td>
<td>1</td>
</tr>
</tbody>
</table>
SEQUENCE DETECTOR FOR 010 OR 1001 (CONT)

<table>
<thead>
<tr>
<th>present</th>
<th>next</th>
<th>output</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>x=0</td>
<td>x=1</td>
</tr>
<tr>
<td>s0</td>
<td>s1</td>
<td>s4</td>
</tr>
<tr>
<td>s1</td>
<td>s1</td>
<td>s2</td>
</tr>
<tr>
<td>s2</td>
<td>s3</td>
<td>s4</td>
</tr>
<tr>
<td>s3</td>
<td>s6</td>
<td>s2</td>
</tr>
<tr>
<td>s4</td>
<td>s3</td>
<td>s4</td>
</tr>
<tr>
<td>s6</td>
<td>s1</td>
<td>s2</td>
</tr>
</tbody>
</table>

- (s0, s1, s6), (s2, s4), (s0, s2, s4), (s1, s3, s6) G1
- (s1, s2), (s3, s4) G2 x 2
- (s0, s1, s3, s4) G3

<table>
<thead>
<tr>
<th></th>
<th>00</th>
<th>01</th>
<th>11</th>
<th>10</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>s0</td>
<td>s1</td>
<td>s6</td>
<td>X</td>
</tr>
<tr>
<td>1</td>
<td>s4</td>
<td>s2</td>
<td>X</td>
<td>s3</td>
</tr>
</tbody>
</table>

ONE POSSIBILITY
SEQUENCE DETECTOR FOR 010 OR 1001 (CONT)

<table>
<thead>
<tr>
<th>present</th>
<th>Next</th>
<th>output</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>x=0</td>
<td>x=1</td>
</tr>
<tr>
<td>s0</td>
<td>000</td>
<td>001</td>
</tr>
<tr>
<td>s1</td>
<td>001</td>
<td>001</td>
</tr>
<tr>
<td>s2</td>
<td>101</td>
<td>110</td>
</tr>
<tr>
<td>s3</td>
<td>110</td>
<td>011</td>
</tr>
<tr>
<td>s4</td>
<td>100</td>
<td>110</td>
</tr>
<tr>
<td>s6</td>
<td>011</td>
<td>001</td>
</tr>
</tbody>
</table>
5-19 A sequential circuit has three flip-flops $A$, $B$, $C$; one input $x$; and one output $y$. The state diagram is shown in Fig. P5-19. The circuit is to be designed by treating the unused states as don’t-care conditions. Analyze the circuit obtained from the design to determine the effect of the unused states.
(a) Use $D$ flip-flops in the design. 
(b) Use $JK$ flip-flops in the design.
<table>
<thead>
<tr>
<th>Present state</th>
<th>Input $x$</th>
<th>Next state</th>
<th>Output $y$</th>
</tr>
</thead>
<tbody>
<tr>
<td>ABC</td>
<td>ABC</td>
<td></td>
<td></td>
</tr>
<tr>
<td>000</td>
<td>0</td>
<td>011</td>
<td>0</td>
</tr>
<tr>
<td>000</td>
<td>0</td>
<td>100</td>
<td>1</td>
</tr>
<tr>
<td>001</td>
<td>0</td>
<td>001</td>
<td>0</td>
</tr>
<tr>
<td>001</td>
<td>1</td>
<td>100</td>
<td>1</td>
</tr>
<tr>
<td>010</td>
<td>0</td>
<td>010</td>
<td>0</td>
</tr>
<tr>
<td>010</td>
<td>1</td>
<td>100</td>
<td>1</td>
</tr>
<tr>
<td>011</td>
<td>0</td>
<td>010</td>
<td>0</td>
</tr>
<tr>
<td>011</td>
<td>1</td>
<td>100</td>
<td>1</td>
</tr>
<tr>
<td>100</td>
<td>0</td>
<td>010</td>
<td>0</td>
</tr>
<tr>
<td>100</td>
<td>1</td>
<td>101</td>
<td>0</td>
</tr>
</tbody>
</table>

$$d(A, B, C, x) = \Sigma(0, 1, 12, 13, 14, 15)$$

$$DA = A'B'x$$

$$DB = A + C'x + BCx$$

$$DC = Cx' + Ax + A'B'x'$$

$$y = A'x$$
(b) Use JK flip-flops: same state table as in part (a).

Flip-flop inputs

<table>
<thead>
<tr>
<th>JA</th>
<th>KA</th>
<th>JB</th>
<th>KB</th>
<th>JC</th>
<th>KC</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>1</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

\[ JA = B'X \]
\[ JB = A + C'X' \]
\[ JC = AX + AB'X' \]
\[ y = A'X \]

Self-correcting because \( KA = 1 \)

self-correcting
Fig. 5-29 State Diagram of 3-Bit Binary Counter
Fig. 5-30 Maps for 3-Bit Binary Counter

\[ T_{A2} = A_1 A_0 \]

\[ T_{A1} = A_0 \]

\[ T_{A0} = 1 \]
Fig. 5-31 Logic Diagram of 3-Bit Binary Counter
EXERCISE

- DRAW THE STATE DIAGRAM FOR A CIRCUIT THAT DETECTS THE SEQUENCE “0101” (LEFT-TO-RIGHT) USING
  - A MOORE FINITE STATE MACHINE (FSM)
  - A MEALY FSM
EXERCISE

For a clocked synchronous state machine with two inputs, X and Y, and one output, Z, the output should be 1 if the number of 1 inputs on X and Y since reset is a multiple of 4, and 0 otherwise. Draw the state diagram for a

- Moore machine
- Mealy machine
EXERCISE

- Design a circuit to detect the sequence $D_0D_1D_2D_3D_4=01101$, where $D_0$ is the first bit to arrive at input “X”. The output “Y” should be a logic-1 for a full clock cycle following detection of the sequence.

- Draw a state diagram

- Assign binary states

- Write a state table

- Find the combinational circuit’s logic expressions if D, JK and T flip flops will be used for the 1st, 2nd and 3rd state bits, respectively
Fig. P5-8