DIGITAL DESIGN AND COMPUTER ORGANIZATION

State Reduction and Assignment Design

Team DDCO

Department of Computer Science and Engineering

State Reduction(T1- section 5.7)

- The **analysis** of sequential circuits starts from a circuit diagram and culminates in a state table or diagram.

- The **design** (synthesis) of a sequential circuit starts from a set of specifications and culminates in a logic diagram. (section 5.8)

- Two sequential circuits may exhibit the same input–output behavior, but have a different number of internal states in their state diagram.

State Reduction(T1- section 5.7)

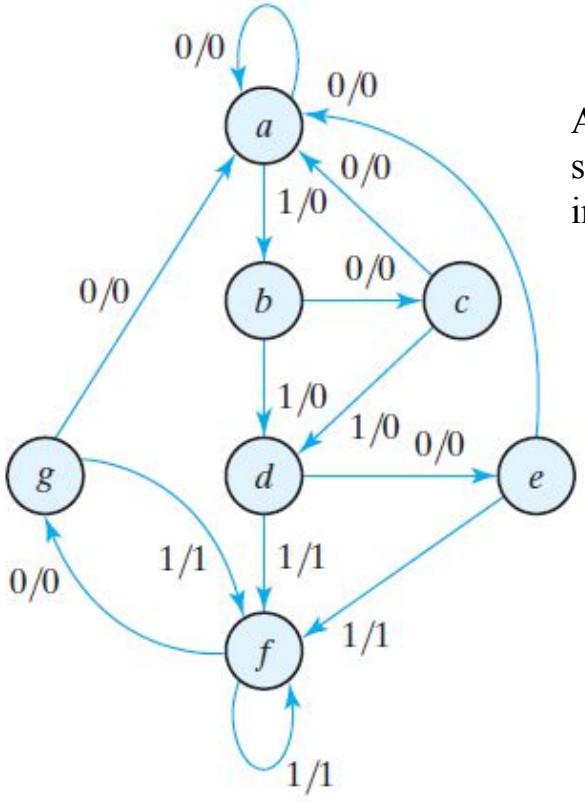
- Properties of sequential circuits that may **simplify a design by reducing the number of gates and flip-flops it uses.** In general, reducing the number of flip flops reduces the **cost** of a circuit.
- **The reduction in the number of flip-flops in a sequential circuit is referred to as the state-reduction problem**

- Since m flip-flops produce 2^m states, a reduction in the number of states may (or may not) result in a reduction in the number of flip-flops.

- Each state can be encoded with binary numbers for 2 flipflops
  - A=00
  - B=01
  - C=10
  - D=11

State Reduction

State diagram



As an example, consider the input sequence 01010110100 starting from the initial state a .

State Reduction

- As an example, consider the input sequence 01010110100 starting from the initial state a .
- Each input of 0 or 1 produces an output of 0 or 1 and causes the circuit to go to the next state.

state	*a*	*a*	*b*	*c*	*d*	*e*	*f*	*f*	*g*	*f*	*g*	*a*
input	0	1	0	1	0	1	1	0	1	0	0	
output	0	0	0	0	0	1	1	0	1	0	0	

State Reduction

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.

State Reduction

Table 5.6
State Table

	Next State		Output	
Present State	$x = 0$	$x = 1$	$x = 0$	$x = 1$
a	a	b	0	0
b	c	d	0	0
c	a	d	0	0
d	e	f	0	1
e	a	f	0	1
f	g	f	0	1
g	a	f	0	1

State Reduction

- States e and g are two such states: They both go to states a and f and have outputs of 0 and 1 for x=0 and x=1, respectively. Therefore, states g and e are equivalent, and one of these states can be removed.
- states f and d are equivalent, and state f can be removed and replaced by d .

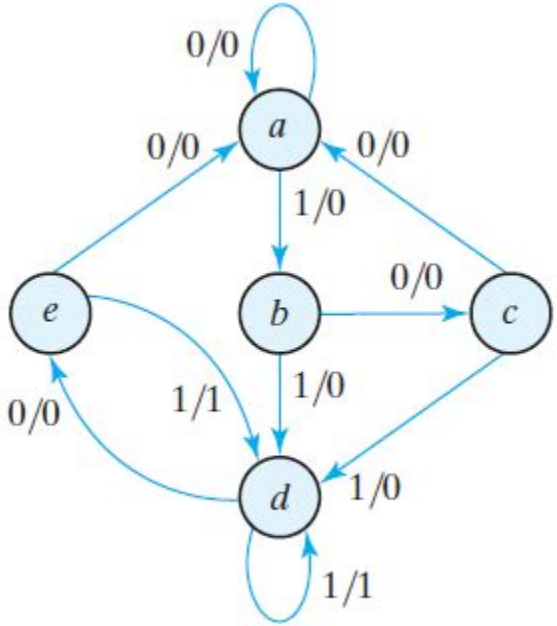
Table 5.7
Reducing the State Table

Present State	Next State		Output	
	$x = 0$	$x = 1$	$x = 0$	$x = 1$
a	a	b	0	0
b	c	d	0	0
c	a	d	0	0
d	e	f	0	1
e	a	f	0	1
f	e	f	0	1

Reduced State Table

Present State	Next State		Output	
	$x = 0$	$x = 1$	$x = 0$	$x = 1$
a	a	b	0	0
b	c	d	0	0
c	a	d	0	0
d	e	d	0	1
e	a	d	0	1

State Reduction



Reduced state diagram

State Reduction

- The sequential circuit of this example was reduced from seven to five states.
- In general, reducing the number of states in a state table may result in a circuit with **less equipment.**
- However, **the fact that a state table has been reduced to fewer states does not guarantee a saving in the number of flip-flops or the number of gates.**
- In actual practice designers may skip this step because target devices are rich in resources.

State Assignment

- For a circuit with m states, the codes must contain n bits, where $2^n >= m$.
- For example, with three bits, it is possible to assign codes to eight states, denoted by binary numbers 000 through 111.
- Example: 7 states are there
  - 000 to 111
  - one state can be unused.
  - In truth table mentioned as don’t care condition

State Assignment

Three Possible Binary State Assignments

State	Assignment 1, Binary	Assignment 2, Gray Code	Assignment 3, One-Hot
a	000	000	00001
b	001	001	00010
c	010	011	00100
d	011	010	01000
e	100	110	10000

- Binary= 2^n=m-The simplest way to code five states is to use the first five integers in binary counting order
- Gray code = adjacent bit differ by one bit
- On hot encoding-t. At any given time, only one bit is equal to 1 while all others are kept at 0. This type of assignment uses **one flipflop per state.**

State Assignment

- One-hot encoding usually leads to simpler decoding logic for the next state and output

- One-hot machines can be **faster** than machines with sequential binary encoding, and the silicon area required by the **extra flip-flops** can be offset by the area saved by using simpler decoding logic.

- This **trade-off** is not guaranteed, so it must be evaluated for a given design.

State Assignment

Reduced State Table with Binary Assignment 1

Present State	Next State $x = 0$	Next State $x = 1$	Output $x = 0$	Output $x = 1$
000	000	001	0	0
001	010	011	0	0
010	000	011	0	0
011	100	011	0	1
100	000	011	0	1

Reduced State Table

Present State	Next State $x = 0$	Next State $x = 1$	Output $x = 0$	Output $x = 1$
a	*a*	*b*	0	0
b	*c*	*d*	0	0
c	*a*	*d*	0	0
d	*e*	*d*	0	1
e	*a*	*d*	0	1

Also referred as transition table

A different assignment will result in a state table with different binary values for the states.

Design Procedure(T1 –section 5.8)

- Design procedures or methodologies **specify hardware** that will implement a desired behavior
- The sequential building block used by synthesis tools is the D flip-flop. Together with additional logic, it can implement the behavior of JK and T flip-flops.
- In fact, designers **generally do not concern themselves with the type of flip-flop; rather, their focus is on correctly describing the sequential functionality that is to be implemented by the synthesis tool**

- In contrast to a combinational circuit, which is fully specified by a truth table, a sequential circuit requires a state table for its specification.

Design Procedure(T1 –section 5.8)

- A synchronous sequential circuit is made up of flip-flops and combinational gates.

- The number of flip-flops is determined from the number of states needed in the circuit and the choice of state assignment codes.

- The combinational circuit is derived from the state table

Design Procedure

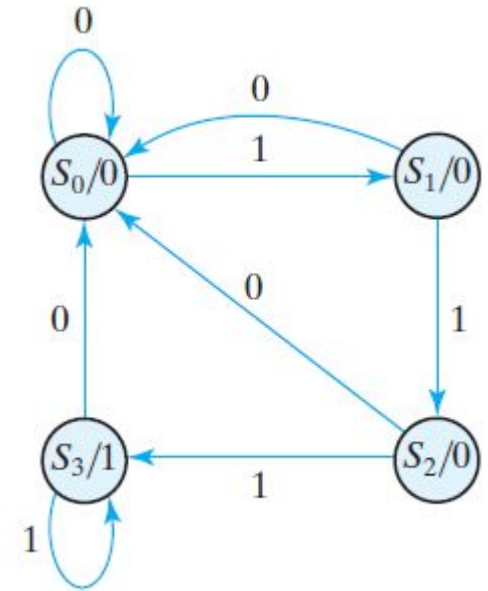
The procedure for designing synchronous sequential circuits can be summarized by a list of recommended steps:

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.

Design Procedure

Suppose we wish to design a circuit that detects a sequence of **three or more consecutive 1's** in a string of bits coming through an input line (i.e., the input is a serial bit stream).

S0= reset state
Below shown is moore model



Design Procedure

If the input is 0, the circuit stays in S0, but if the input is 1, it goes to state S1 to indicate that a 1 was detected. If the next input is 1, the change is to state S2 to indicate the arrival of two consecutive 1's, but if the input is 0, the state goes back to S0. The third consecutive 1 sends the circuit to state S3. If more 1's are detected, the circuit stays in S3. Any 0 input sends the circuit back to S0. In this way, the circuit stays in S3 as long as there are three or more consecutive 1's received.

This is a Moore model sequential circuit, since the output is 1 when the circuit is in state S3 and is 0 otherwise.

Synthesis Using D Flip-Flops

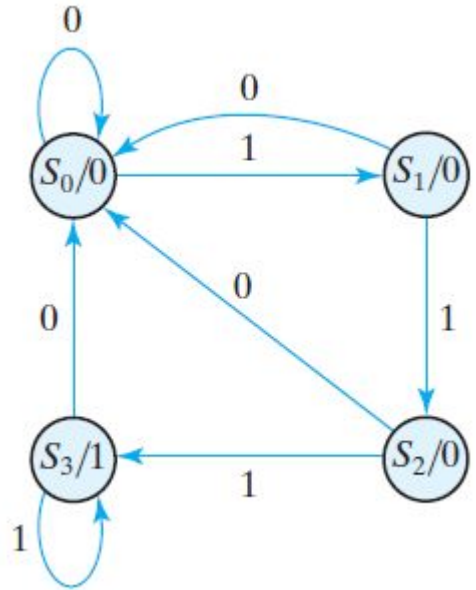
- We choose two D flip-flops to represent the four states, and we label their outputs A and B . There is one input x and one output y.
- The characteristic equation of the D flip-flop is Q(t +1)=DQ, which means that the next-state values in the state table specify the D input condition for the flip-flop.

State Table for Sequence Detector. – BINARY ENCODED STATE TABLE
S0=00, S1=01, S2=10 s3=11

Synthesis Using D Flip-Flops

State Table for Sequence Detector

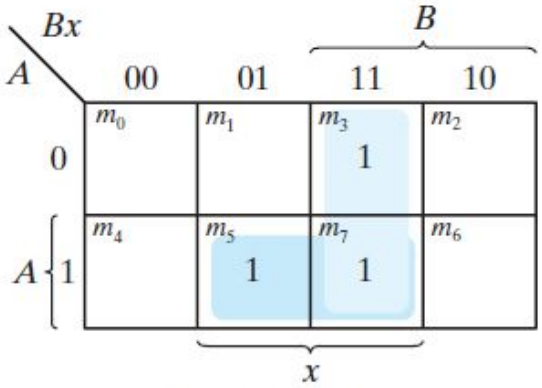
Present State		Input	Next State		Output
A	*B*	*x*	*A*	*B*	*y*
0	0	0	0	0	0
0	0	1	0	1	0
0	1	0	0	0	0
0	1	1	1	0	0
1	0	0	0	0	0
1	0	1	1	1	0
1	1	0	0	0	1
1	1	1	1	1	1



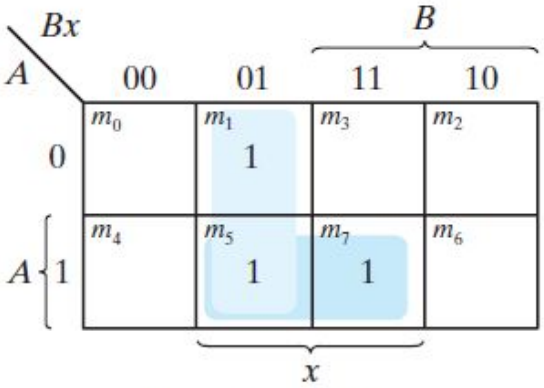
Synthesis Using D Flip-Flops

K-Maps for sequence detector

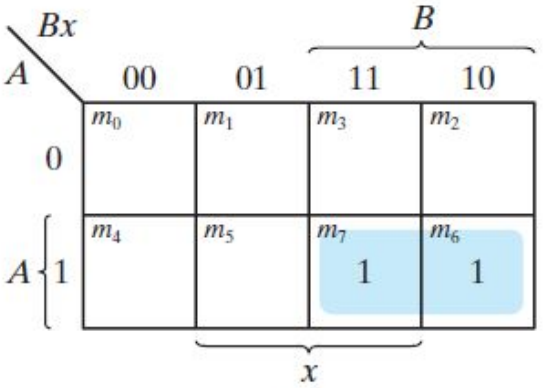
$$A(t+1) = D_A(A,B,x) = \Sigma(3,5,7)$$
$$B(t+1) = D_B(A,B,x) = \Sigma(1,5,7)$$
$$y(A,B,x) = \Sigma(6,7)$$



$D_A = Ax + Bx$



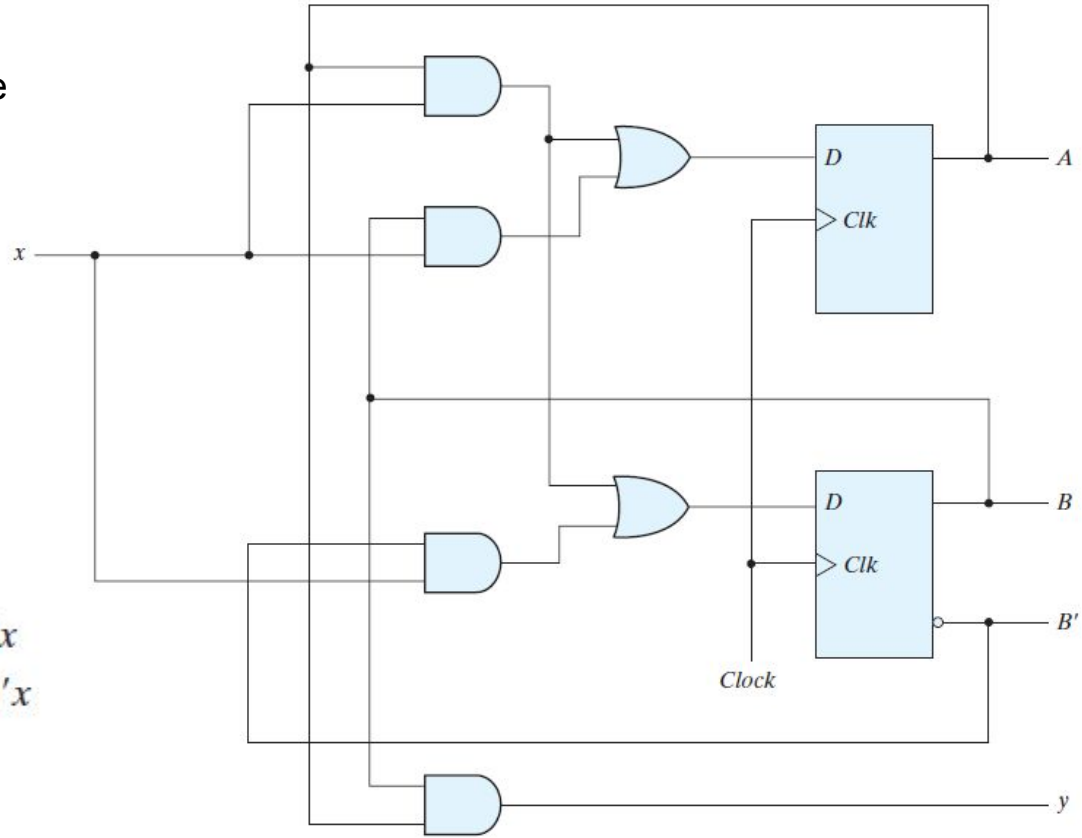
$D_B = Ax + B'x$



$y = AB$

Synthesis Using D Flip-Flops

Logic diagram of a Moore-type sequence detector



$$D_A = Ax + Bx$$
$$D_B = Ax + B'x$$
$$y = AB$$

Excitation Table

- When D -type flip-flops are employed, the input equations are obtained directly from the next state. This is not the case for the JK and T types of flip-flops.
- In order to determine the input equations for these flip-flops, it is necessary to derive a functional relationship between the state table and the input equations.

Flip-Flop Excitation Tables

$Q(t)$	$Q(t+1)$	J	K
0	0	0	X
0	1	1	X
1	0	X	1
1	1	X	0

(a) *JK* Flip-Flop

$Q(t)$	$Q(t+1)$	T
0	0	0
0	1	1
1	0	1
1	1	0

(b) *T* Flip-Flop

Synthesis Using JK Flip-Flops

State Table and JK Flip-Flop Inputs

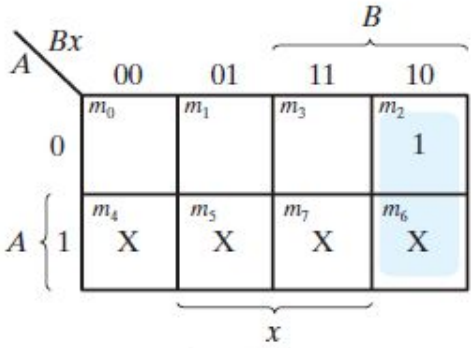
Present State		Input	Next State		Flip-Flop Inputs			
A	B	x	A	B	J_A	K_A	J_B	K_B
0	0	0	0	0	0	X	0	X
0	0	1	0	1	0	X	1	X
0	1	0	1	0	1	X	X	1
0	1	1	0	1	0	X	X	0
1	0	0	1	0	X	0	0	X
1	0	1	1	1	X	0	1	X
1	1	0	1	1	X	0	X	0
1	1	1	0	0	X	1	X	1

Q(t)	Q(t = 1)	J	K
0	0	0	X
0	1	1	X
1	0	X	1
1	1	X	0

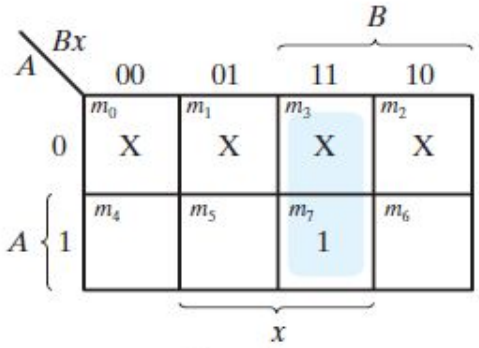
(a) *JK* Flip-Flop

Excitation Table

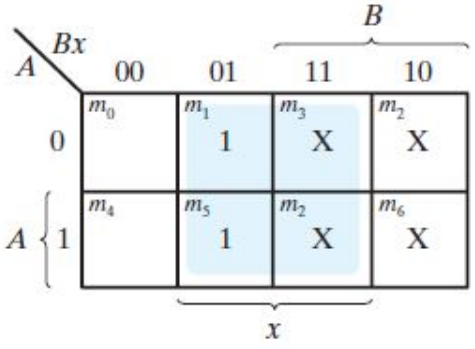
Synthesis Using JK Flip-Flops



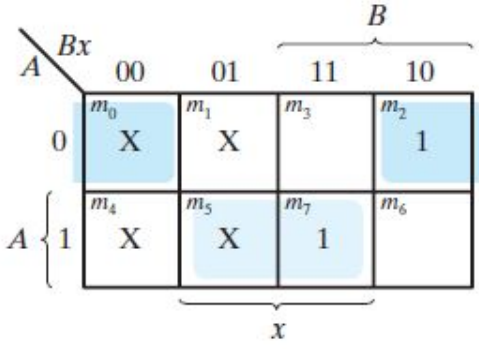
$J_A = Bx'$



$K_A = Bx$

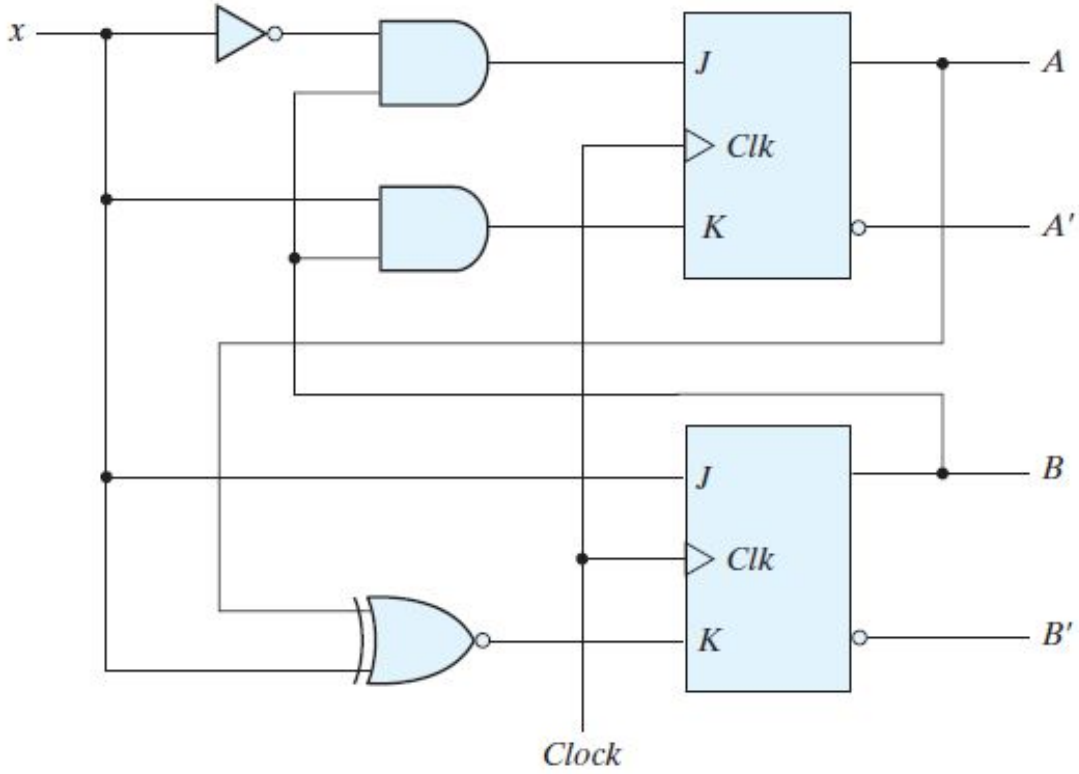


$J_B = x$



$K_B = (A \oplus x)'$

Synthesis Using JK Flip-Flops

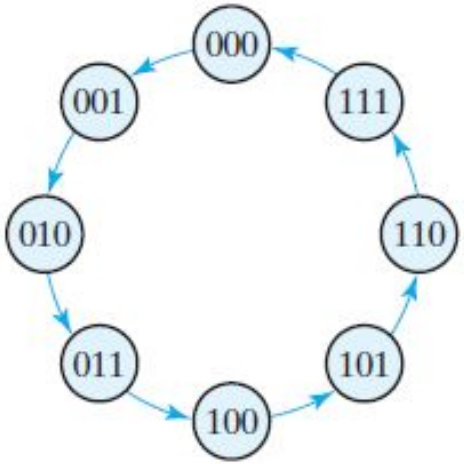


Logic diagram for sequential circuit with *JK* flip-flops

Synthesis Using T Flip-Flops

- The procedure for synthesizing circuits using T flip-flops will be demonstrated by designing a binary counter.
- *An n -bit binary counter consists of n flip-flops* that can count in binary from 0 to 2^n - 1

- the state diagram of a counter does not have to show input and output values along the directed lines.
- The only input to the circuit is the clock, and the outputs are specified by the present state of the flip-flops.
- The next state of a counter depends entirely on its present state, and the state transition occurs every time the clock goes through a transition.



State diagram of three-bit binary counter.

Synthesis Using T Flip-Flops

State Table for Three-Bit Counter

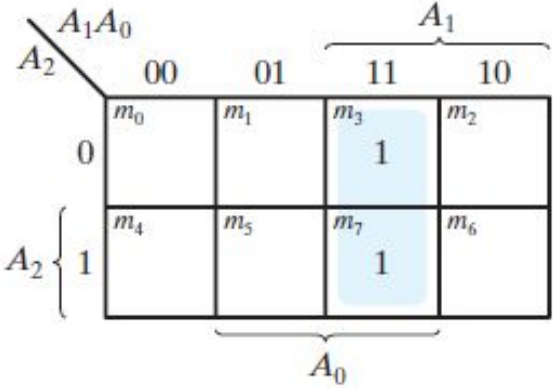
Present State			Next State			Flip-Flop Inputs		
A_2	A_1	A_0	A_2	A_1	A_0	T_{A2}	T_{A1}	T_{A0}
0	0	0	0	0	1	0	0	1
0	0	1	0	1	0	0	1	1
0	1	0	0	1	1	0	0	1
0	1	1	1	0	0	1	1	1
1	0	0	1	0	1	0	0	1
1	0	1	1	1	0	0	1	1
1	1	0	1	1	1	0	0	1
1	1	1	0	0	0	1	1	1

Q(t)	Q(t = 1)	T
0	0	0
0	1	1
1	0	1
1	1	0

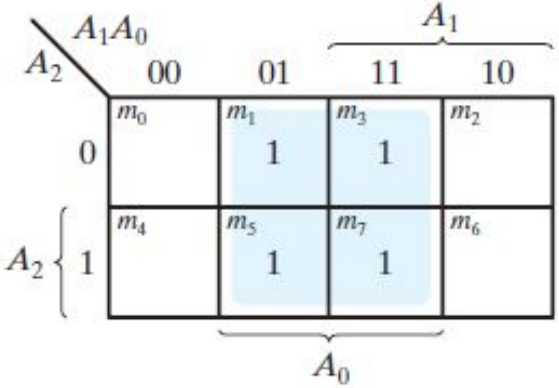
(b) *T* Flip-Flop

Excitation Table

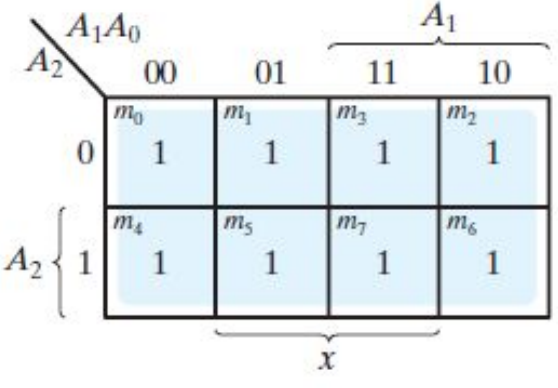
Synthesis Using T Flip-Flops



$T_{A2} = A_1A_0$



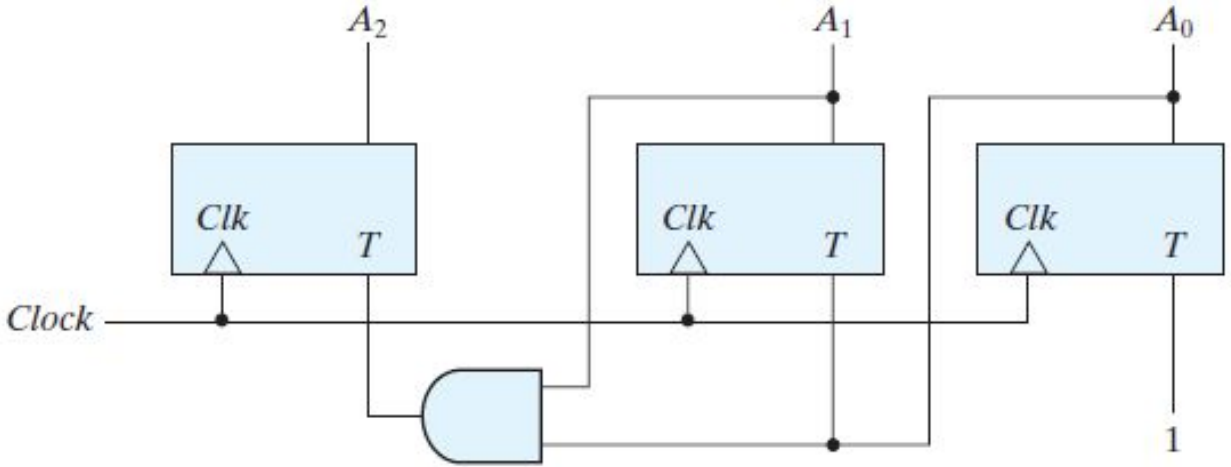
$T_{A1} = A_0$



$T_{A0} = 1$

Maps for three-bit binary counter

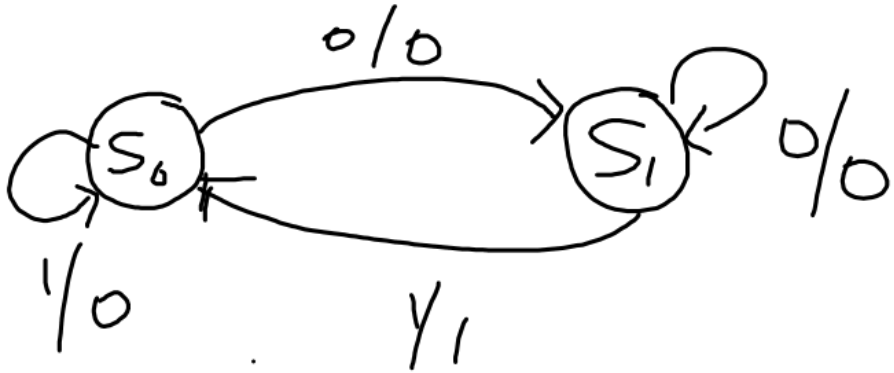
Synthesis Using T Flip-Flops



Logic diagram of three-bit binary counter

Additional questions

Design the circuit diagram for mealy state diagram using D flipflop with binary encoding



Steps: write state table, use binary encodings, build transition table , derive equations for next state and output

Think about it: Design the circuit diagram for mealy state diagram using D flipflop with one hot encoding. One-hot encoding for 8 states → needs 8 flip-flops. Binary encoding for 8 states → needs only 3 flip-flops.

Additional questions

Present state	Input X	Next state	output
S0	0	S1	0
S0	1	S0	0
S1	0	S1	0
S1	1	S0	1

AFTER USING BINARY ENCODING-S_O-0 AND S_1=1

Present state	Input X	Next state	output
0	0	1	0
0	1	0	0
1	0	1	0
1	1	0	1

Additional questions

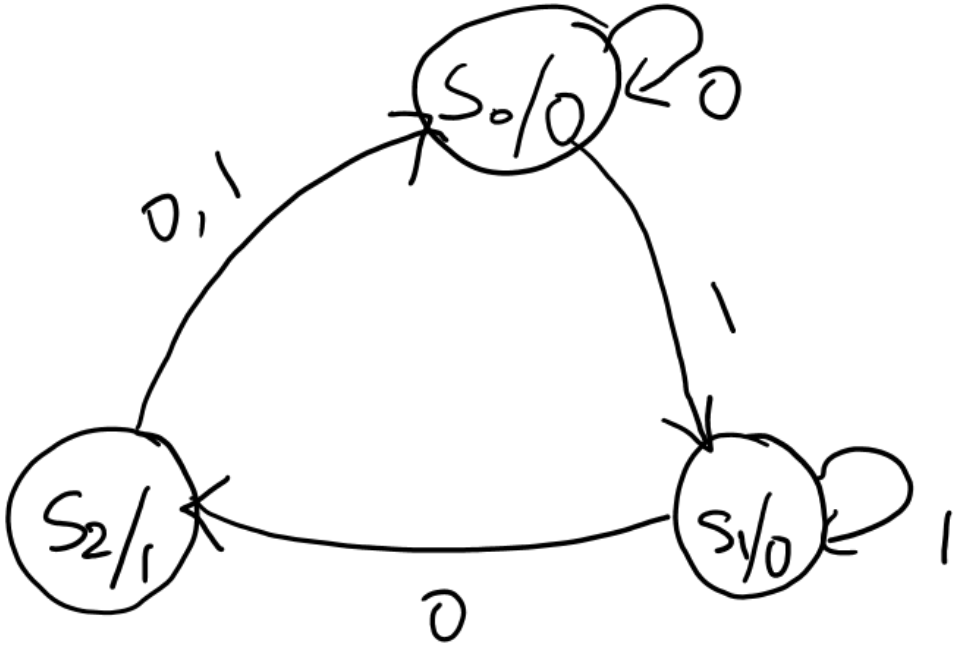
AFTER USING BINARY ENCODING-S_O-0 AND S_1=1

Present state A	Input X	Next state A	Output Y
0	0	1	0
0	1	0	0
1	0	1	0
1	1	0	1

Solving using K map next state equation D_A=x` and y=Ax

Additional questions

Design the circuit diagram for Moore state diagram using D flipflop with Gray code encoding and one hot encoding



Summary

Analysis of circuit:

Circuit diagram → Equations – State table → State diagram

The state table can of 2 format: format 1: output depends on present state(Moore model)
format 2: Output depends on present state and input (Mealey Model)
Both cases Next state always depends on present state and input
For JK and T flipflop to build next state in table use characteristic equation

Design of circuit : From state diagram build Circuit
Steps:(Moore and Mealey model):

State diagram– check for equivalent states– use encoding(binary/gray code/one hot encoding) and build table---select type of flip flop--- get equations for next state and output- build circuit (number of flipflops to be decided on type of encoding)
Fo JK and T flipflop
Use excitation tables inputs

Summary

	Goal	Real-Time Benefit	Example
State Reduction	Minimize number of equivalent states	Lower logic complexity → faster response	Sequence detector in network packet filtering
One-Hot Assignment	One bit per state	Faster decoding logic, suitable for FPGA designs	High-speed motor control
Gray Code	Adjacent states differ by 1 bit	Ensures reliable state transitions (only 1-bit change).	Rotary encoder position tracking-As the shaft rotates, only one bit changes at a time
Binary Assignment	Compact encoding	Since fewer bits are used, fewer **flip-flops** are needed to store the state, reduces hardware cost	Embedded communication controller

MCQ

A finite state machine (FSM) designed using **one-hot encoding** for 5 states will require:

A) 3 flip-flops
B) 4 flip-flops
C) 5 flip-flops
D) 6 flip-flops

A sequential circuit is reduced from 8 states to 5 states after **state reduction**. If implemented with binary state assignment, the change in minimum flip-flops required is:
A) From 4 to 3
B) From 3 to 2
C) From 4 to 2
D) No change

MCQ

A finite state machine (FSM) designed using **one-hot encoding** for 5 states will require:
A) 3 flip-flops
B) 4 flip-flops
C) 5 flip-flops
D) 6 flip-flops
Answer: C
Explanation: In one-hot encoding, **one flip-flop per state** → 5 states → 5 flip-flops.

A sequential circuit is reduced from 8 states to 5 states after **state reduction**. If implemented with binary state assignment, the change in minimum flip-flops required is:
A) From 4 to 3
B) From 3 to 2
C) From 4 to 2
D) No change
Answer: D
Explanation:
$N = 8\lceil \log_2(8) \rceil = \lceil 3 \rceil = 3$ flip-flops
$\lceil \log_2(5) \rceil = \lceil 2.32 \rceil = 3$ flip-flops

THANK YOU

Team DDCO

Department of Computer Science and Engineering