Sequential Circuits

Chapter 4

S. Dandamudi

Outline

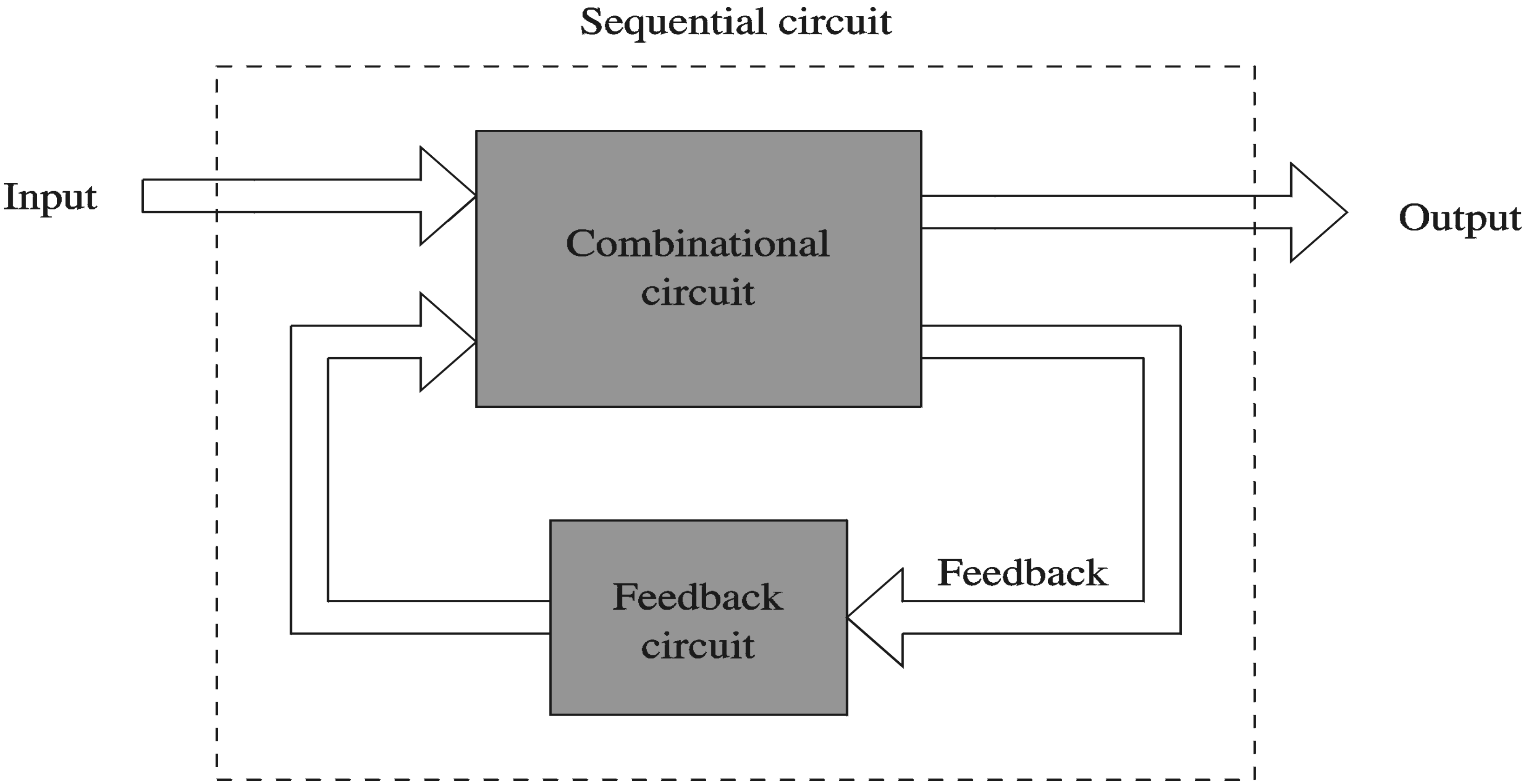
- Introduction
- Clock signal
  - ∗ Propagation delay
- Latches
  - ∗ SR latch
  - ∗ Clocked SR latch
  - ∗ D latch
  - ∗ JK latch
- Flip flops
  - ∗ D flip flop
  - ∗ JK flip flop
- Example chips
- Example sequential circuits
  - ∗ Shift registers
  - ∗ Counters
- Sequential circuit design
  - ∗ Simple design examples
    - » Binary counter
    - » General counter
  - ∗ General design process
    - » Examples
      - – Even-parity checker
      - – Pattern recognition

Introduction

- Output depends on current as well as past inputs
  - ∗ Depends on the history
  - ∗ Have “memory” property
- Sequential circuit consists of
  - » Combinational circuit
  - » Feedback circuit
  - ∗ Past input is encoded into a set of state variables
    - » Uses feedback (to feed the state variables)
      - – Simple feedback
      - – Uses flip flops

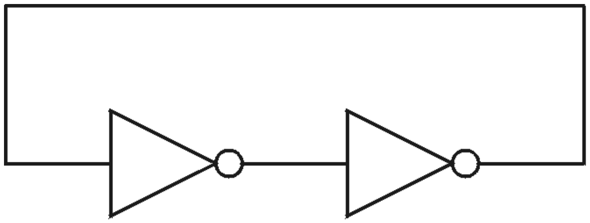
Introduction (cont'd)

Main components of a sequential circuit

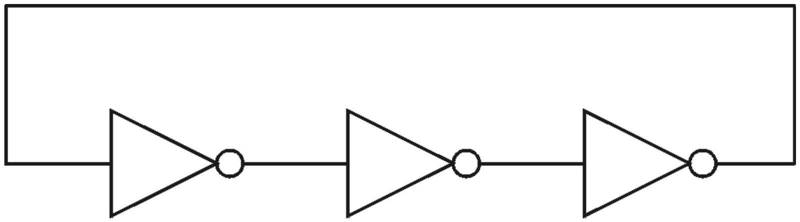


Introduction (cont’d)

- Feedback circuit can be
  - ∗ A simple interconnection some outputs to input, or
  - ∗ A combinational circuit with “memory” property
    - » Uses flip-flops we discuss later
- Feedback can potentially introduce instability



(a) Stable circuit

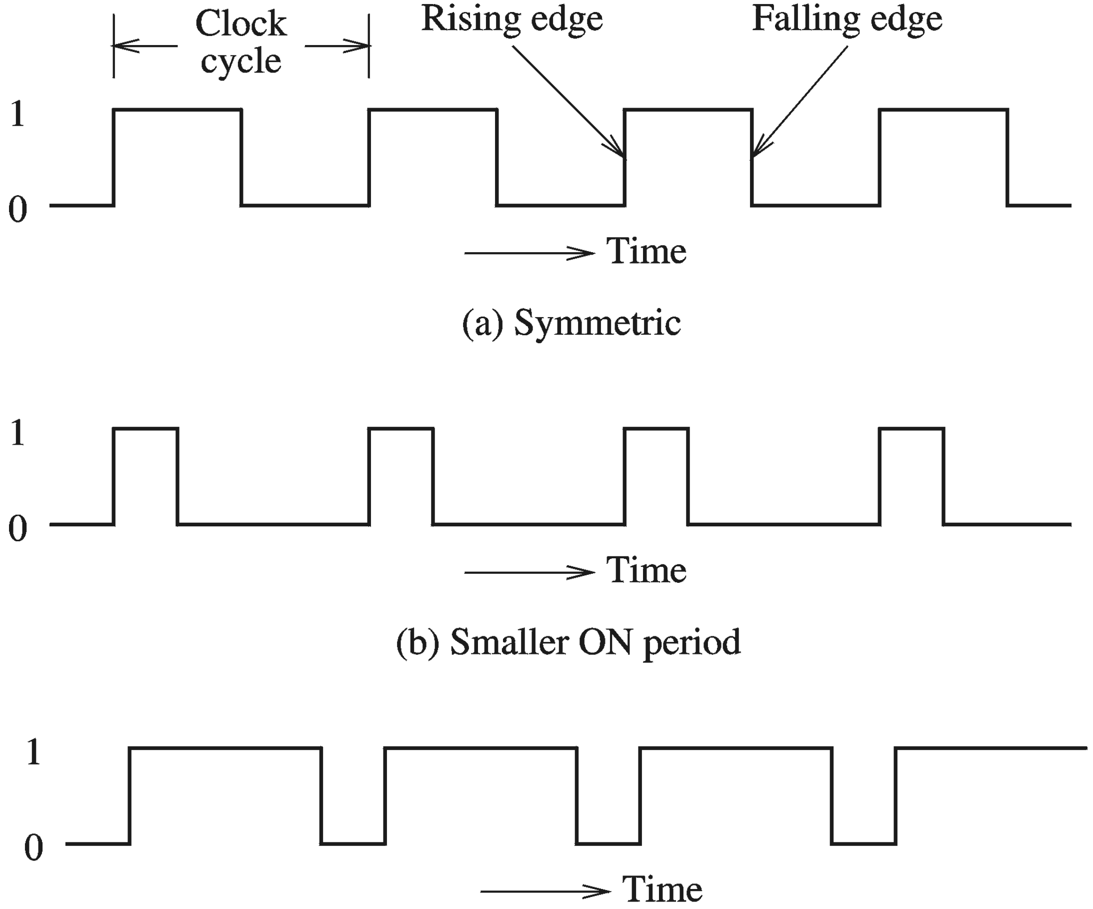


(b) Unstable circuit

Clock Signal

- Digital circuits can be operated in
    - ∗ Asynchronous mode
        - » Circuits operate independently
            - – Several disadvantages
    - ∗ Synchronous mode
        - » Circuits operate in lock-step
        - » A common clock signal drives the circuits
- Clock signal
    - ∗ A sequence of 1s and 0s (ON and OFF periods)
    - ∗ Need not be symmetric

Clock Signal (cont'd)



(a) Symmetric

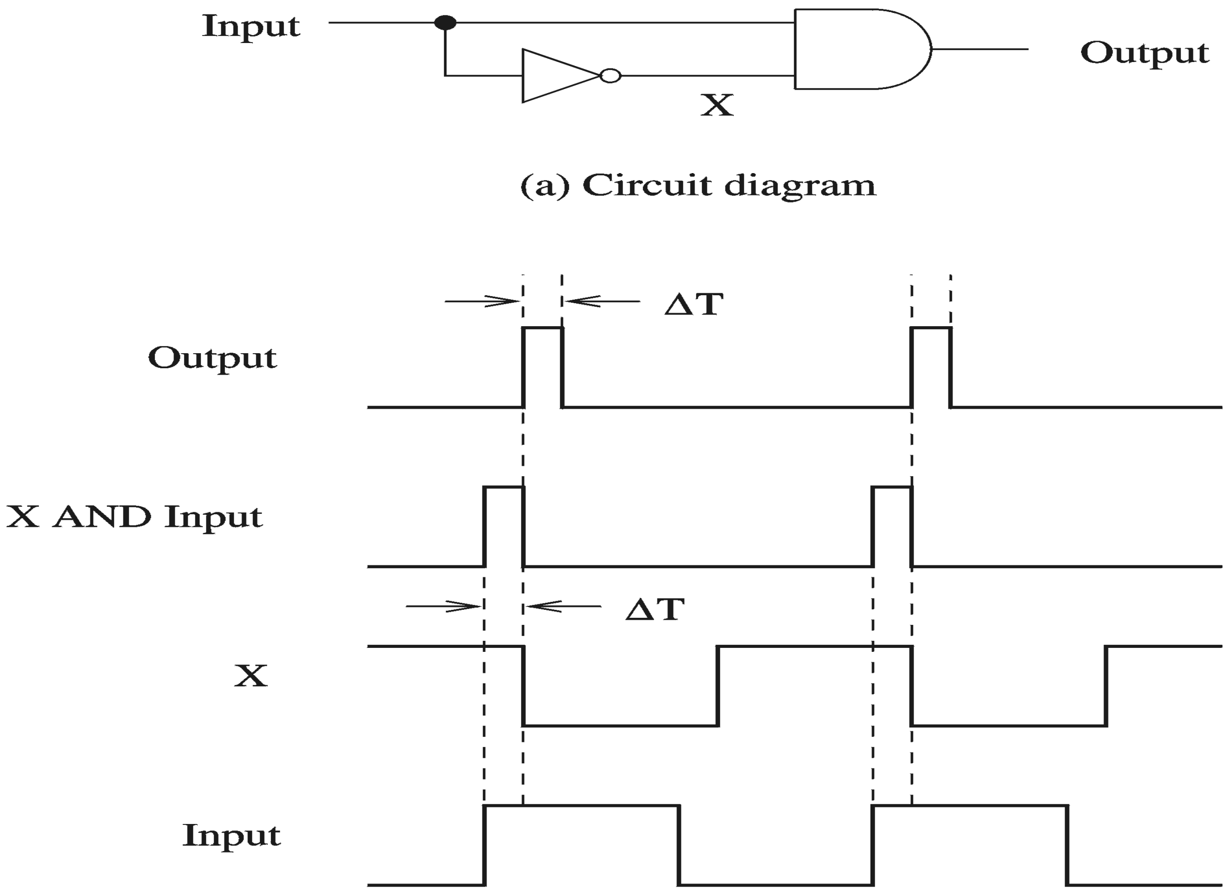
(b) Smaller ON period

(c) Smaller OFF period

Clock Signal (cont'd)

- Clock serves two distinct purposes
  - ∗ Synchronization point
    - » Start of a cycle
    - » End of a cycle
    - » Intermediate point at which the clock signal changes levels
  - ∗ Timing information
    - » Clock period, ON, and OFF periods
- Propagation delay
  - ∗ Time required for the output to react to changes in the inputs

Clock Signal (cont'd)



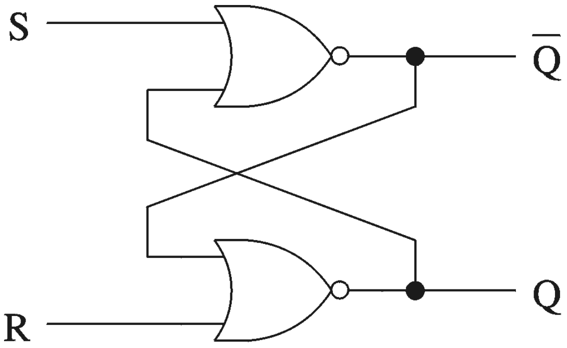
(a) Circuit diagram

(b) Timing diagram

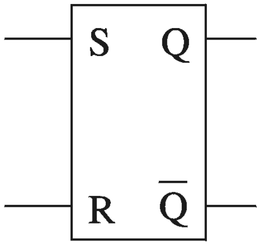
Latches

- Can remember a bit
- Level-sensitive (not edge-sensitive)

A NOR gate implementation of SR latch



(a) Circuit diagram



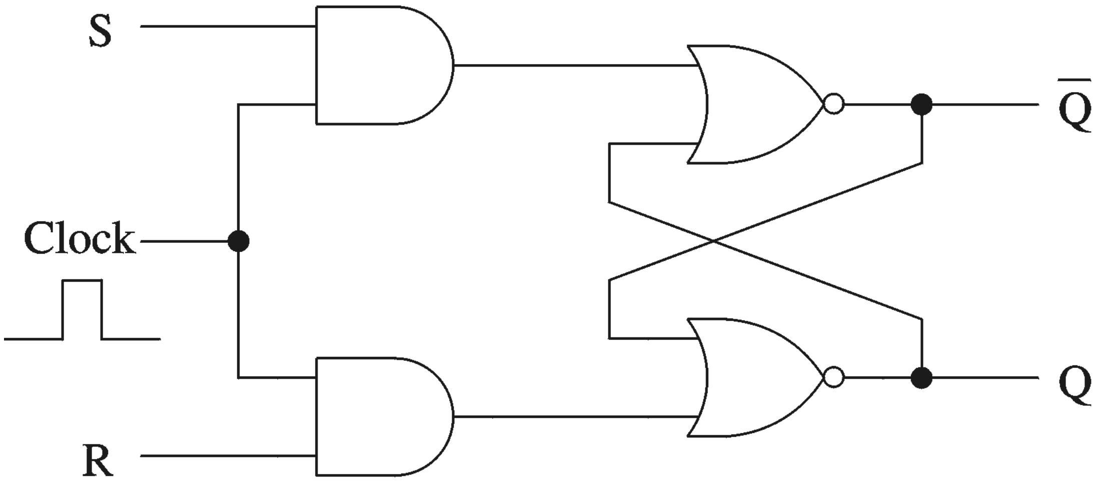
(b) Logic symbol

S	R	Q_{n+1}
0	0	Q_n
0	1	0
1	0	1
~~1~~	~~1~~	~~0~~

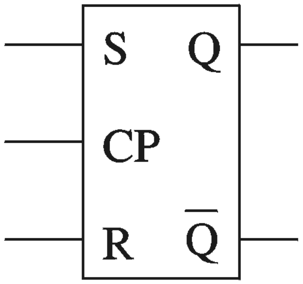
(c) Truth table

Latches (cont'd)

- SR latch outputs follow inputs
- In clocked SR latch, outputs respond at specific instances
  - * Uses a clock signal



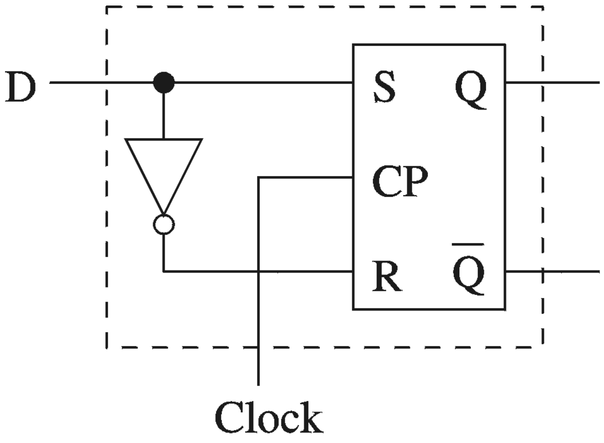
(a) Circuit diagram



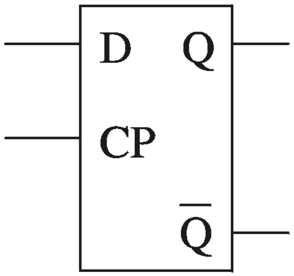
(b) Logic symbol

Latches (cont’d)

- D Latch
  - ∗ Avoids the SR = 11 state



(a) Circuit diagram



(b) Logic symbol

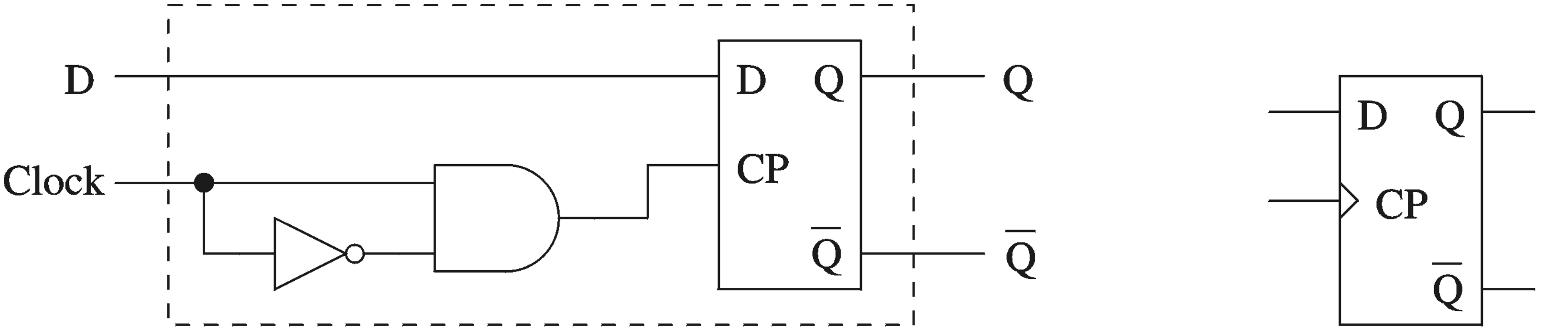
D	Q_{n+1}
0	0
1	1

(c) Truth table

Flip-Flops

- Edge-sensitive devices
  - ∗ Changes occur either at positive or negative edges

Positive edge-triggered D flip-flop

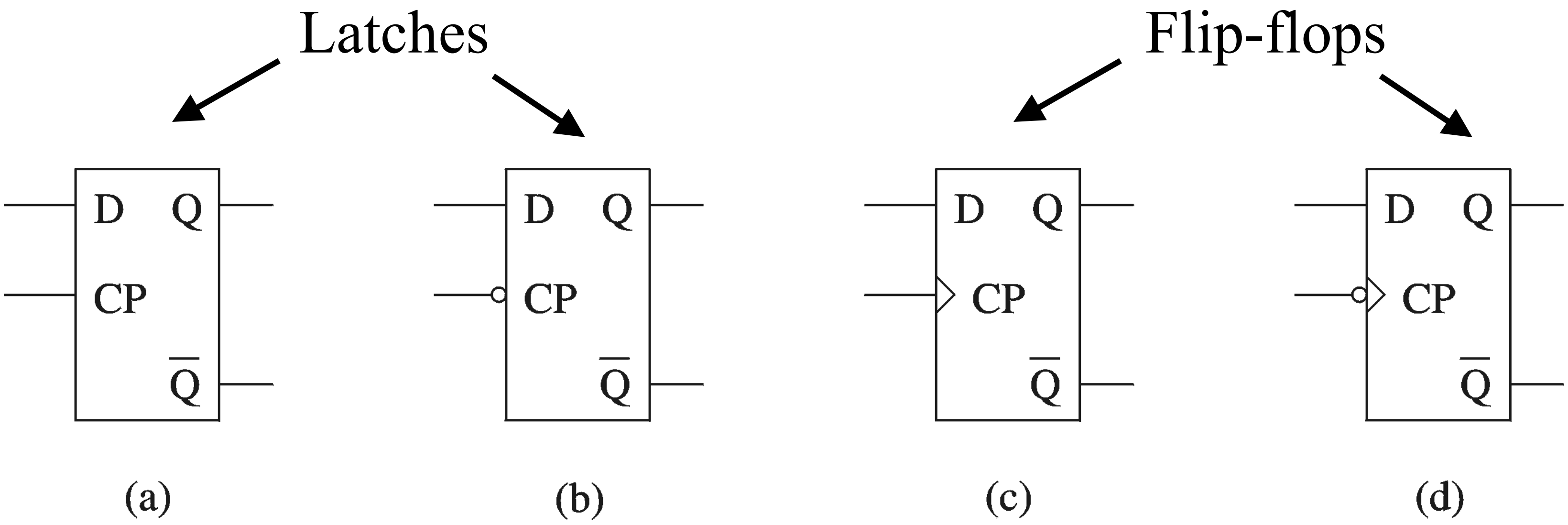


(a) Circuit diagram

(b) Logic symbol

Flip-Flops (cont’d)

- Notation
  - ∗ Not strictly followed in the literature
    - » We follow the following notation for latches and flip-flops



Latches

Flip-flops

(a) Low level

(b) High level

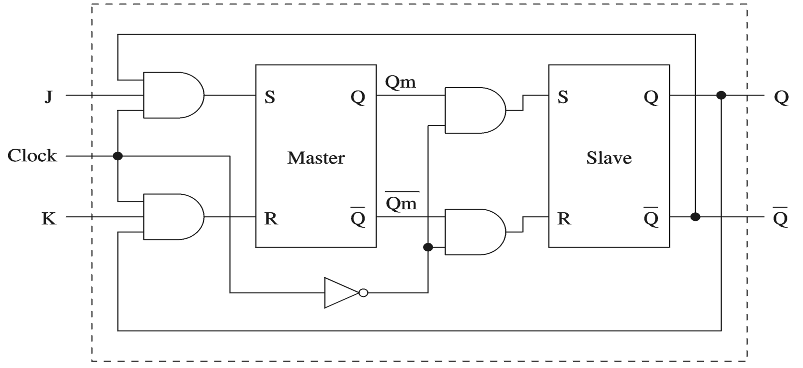
(c) Positive edge

(d) Negative edge

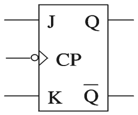
Flip-Flops (cont'd)

JK flip-flop
(master-slave)

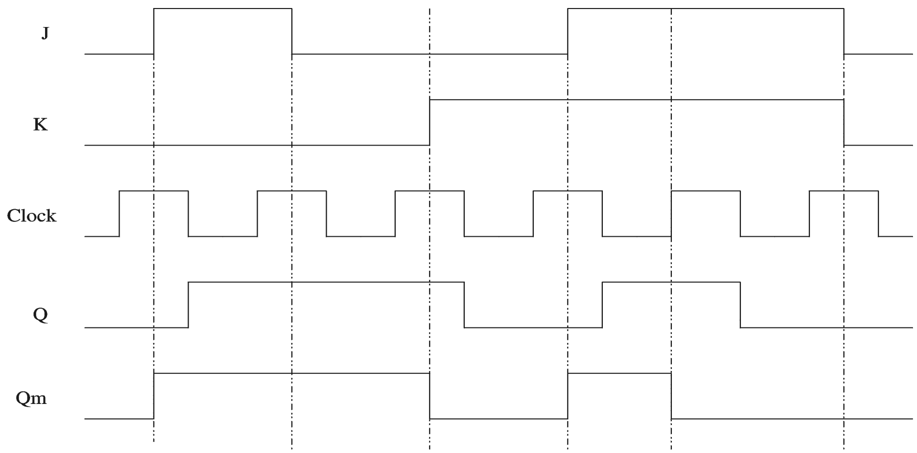
J	K	Q_{n+1}
0	0	Q_n
0	1	0
1	0	1
1	1	$\overline{Q_n}$



(a) Circuit diagram



(b) Logic symbol



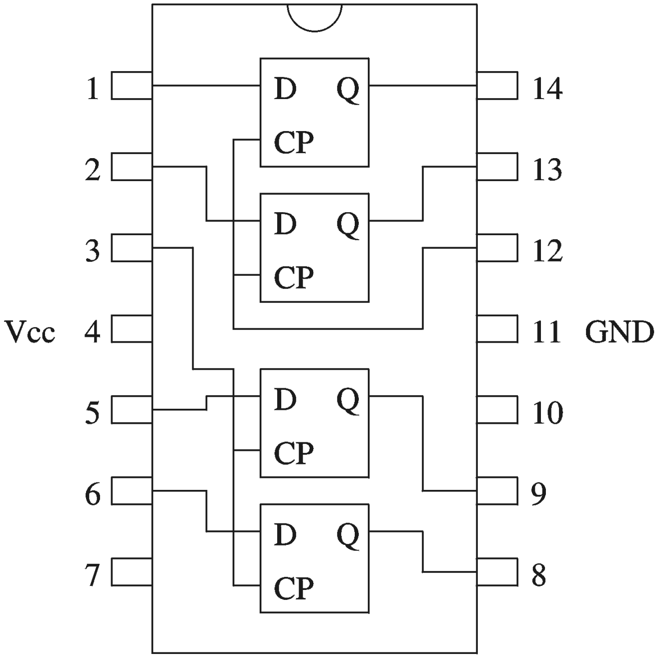
(c) Timing diagram

Flip-Flops (cont'd)

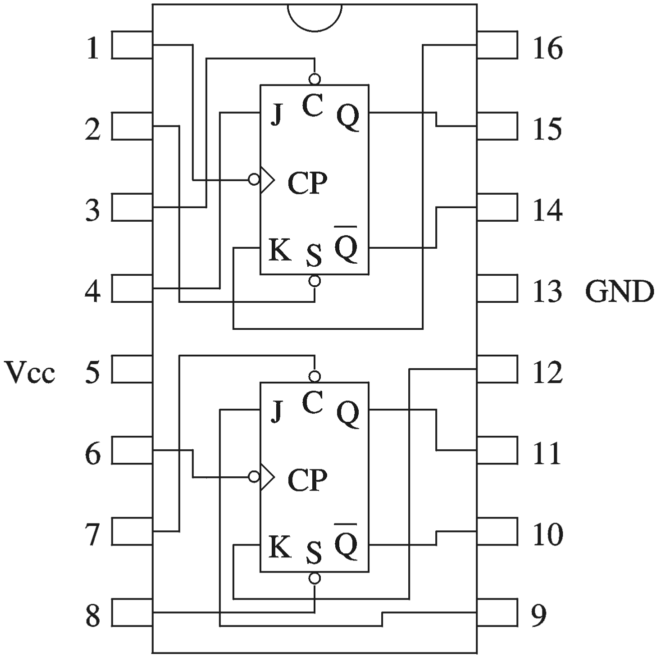
Two example chips

D latches

JK flip-flops



(a) 7477

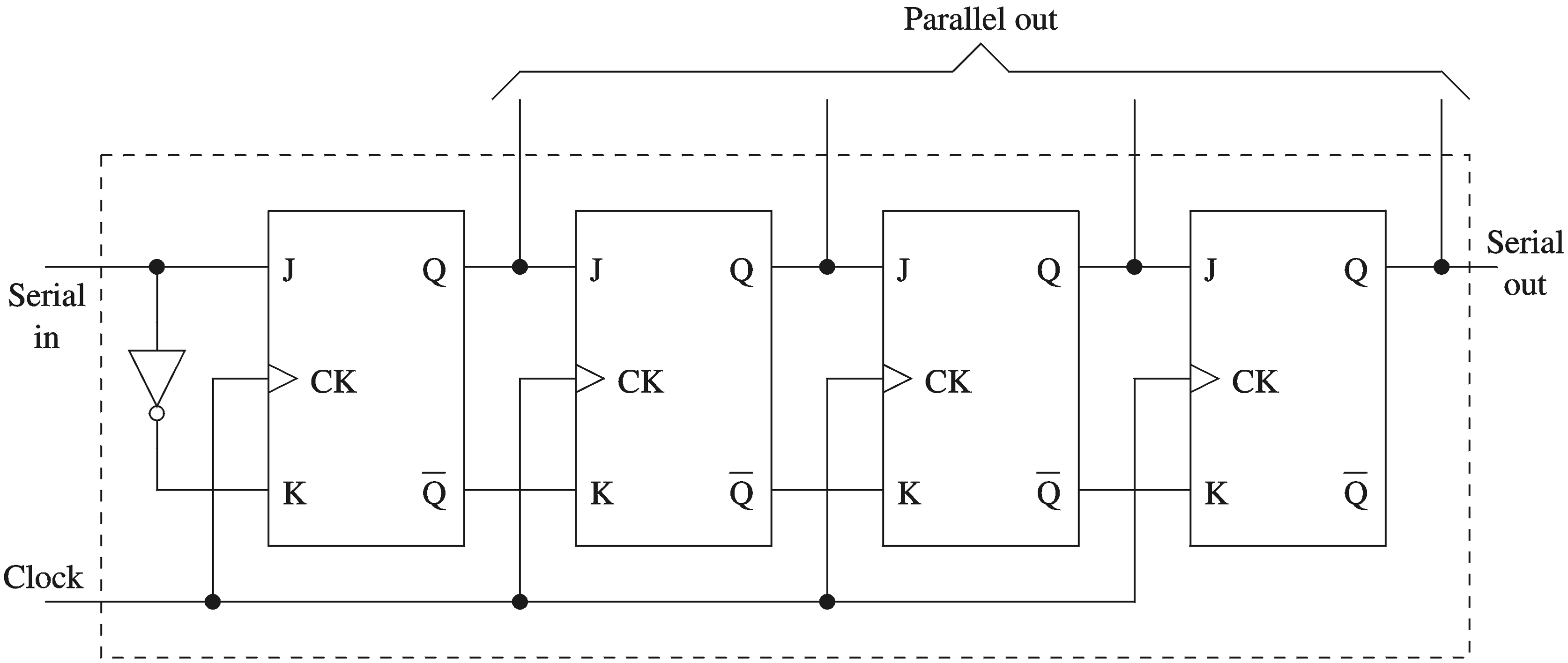


(b) 7476

Example Sequential Circuits

- Shift Registers
  - ∗ Can shift data left or right with each clock pulse

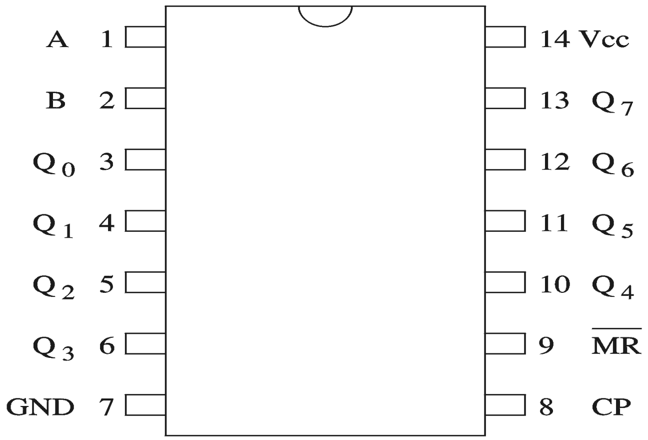
A 4-bit shift register using JK flip-flops



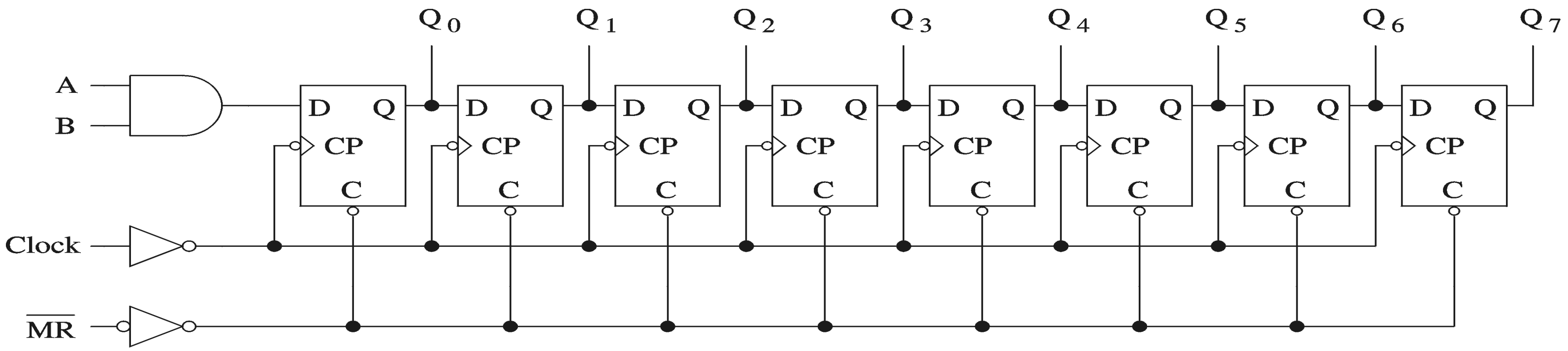
Example Sequential Circuits (cont'd)

74164 shift

Register chip



(a) Connection diagram



(b) Logic diagram

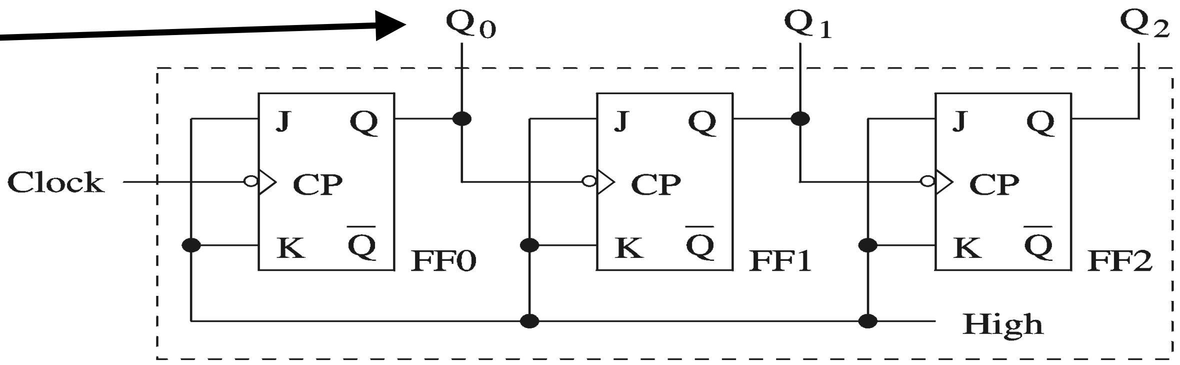
Example Sequential Circuits (cont'd)

- Counters
  - ∗ Easy to build using JK flip-flops
    - » Use the JK = 11 to toggle
  - ∗ Binary counters
    - » Simple design
      - – B bits can count from 0 to 2^B-1
    - » Ripple counter
      - – Increased delay as in ripple-carry adders
      - – Delay proportional to the number of bits
    - » Synchronous counters
      - – Output changes more or less simultaneously
      - – Additional cost/complexity

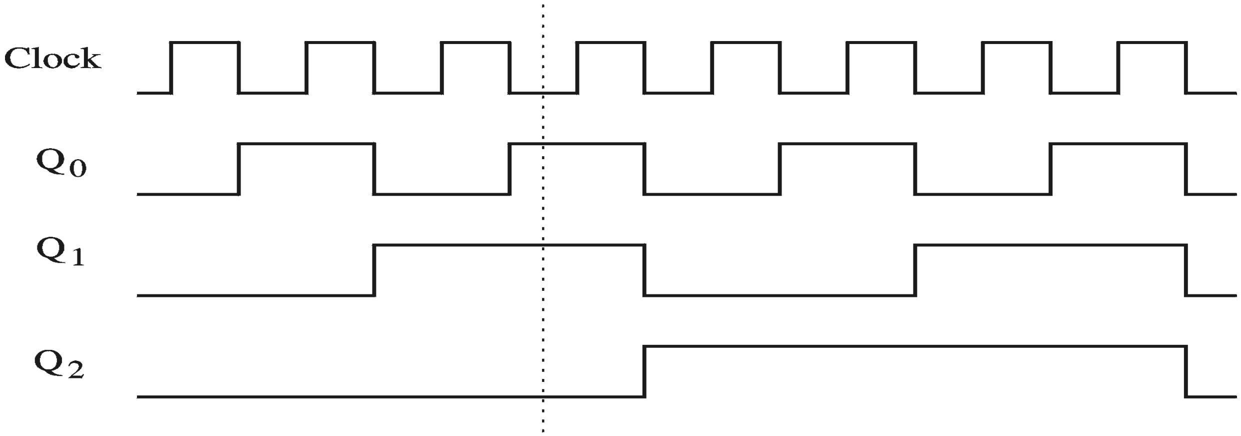
Example Sequential Circuits (cont’d)

LSB

A modulo-8 binary ripple counter



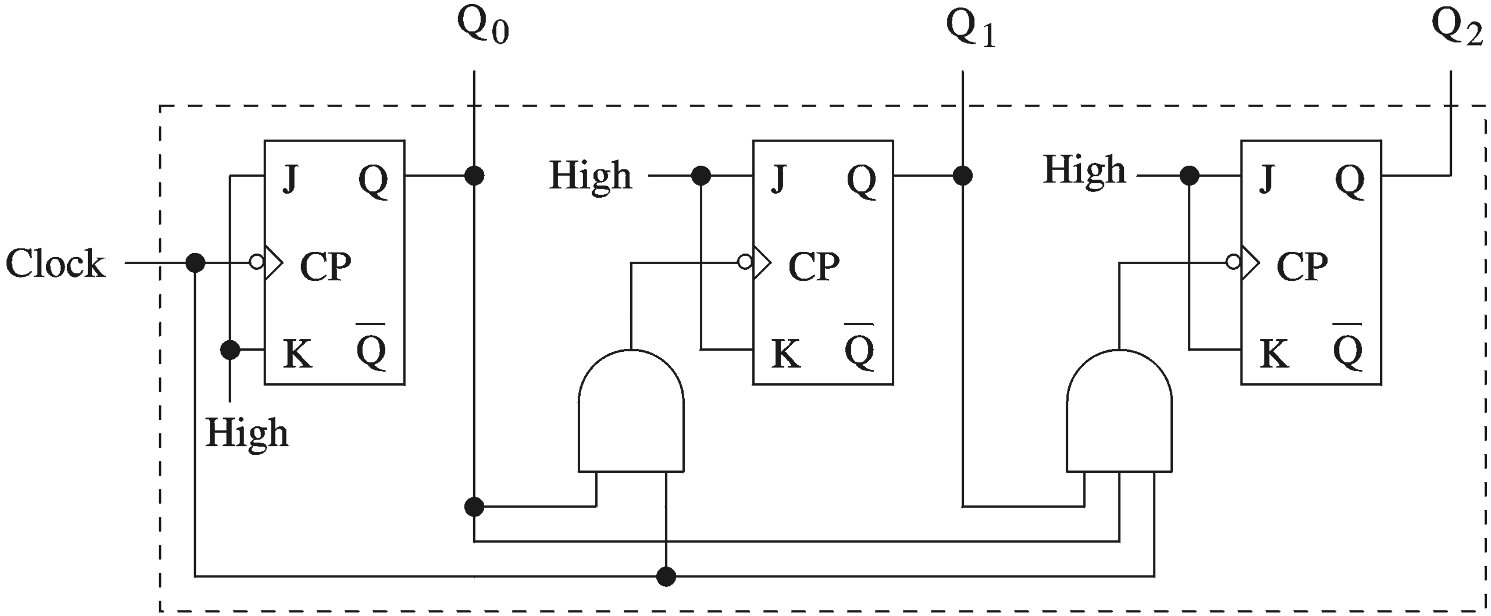
(a) Circuit diagram



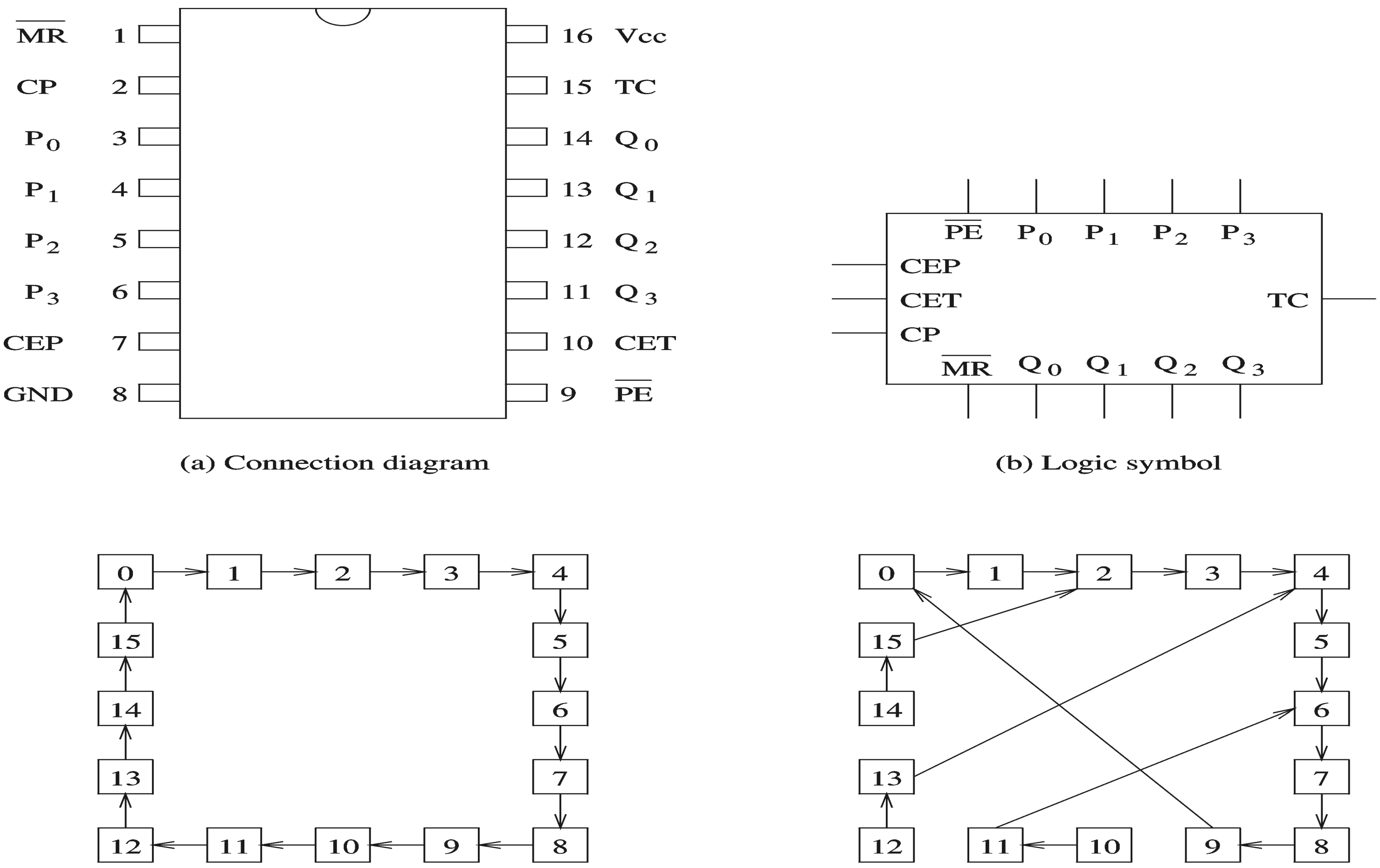
(b) Timing diagram

Example Sequential Circuits (cont'd)

- Synchronous modulo-8 counter
  - ∗ Designed using the following simple rule
    - » Change output if the preceding count bits are 1
      - – Q1 changes whenever Q0 = 1
      - – Q2 changes whenever Q1Q0 = 11



Example Sequential Circuits (cont’d)



(a) Connection diagram

(b) Logic symbol

(c) State diagram of 74161

(d) State diagram of 74160

Example Sequential Circuits (cont’d)

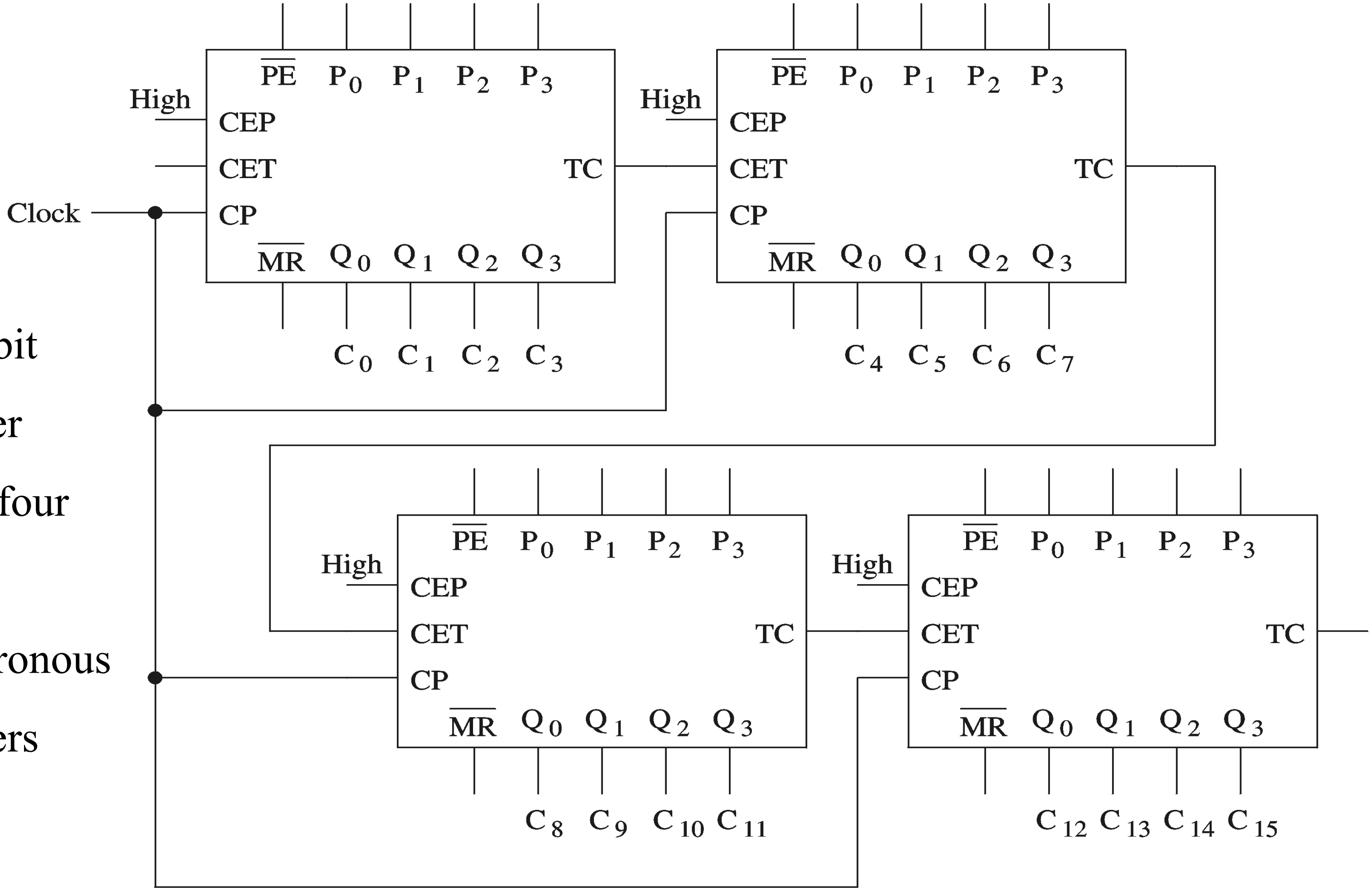
Function table

H = high L = low X = don’t care

$\overline{\text{MR}}$	$\overline{\text{PE}}$	CET	CEP	Action on clock rising edge
L	X	X	X	Clear
H	L	X	X	Parallel load (Pn → Qn)
H	H	H	H	Count (increment)
H	H	L	X	No change (hold); TC is low
H	H	X	L	No change (hold)

Example Sequential Circuits (cont'd)

A 16-bit counter using four 4-bit synchronous counters



Sequential Circuit Design

- Sequential circuit consists of
  - ∗ A combinational circuit that produces output
  - ∗ A feedback circuit
    - » We use JK flip-flops for the feedback circuit
- Simple counter examples using JK flip-flops
  - ∗ Provides alternative counter designs
  - ∗ We know the output
    - » Need to know the input combination that produces this output
    - » Use an excitation table
      - – Built from the truth table

Sequential Circuit Design (cont'd)

(a) JK flip-flop truth table

J	K	Q_n	Q_{n+1}
0	0	0	0
0	0	1	1
0	1	0	0
0	1	1	0
1	0	0	1
1	0	1	1
1	1	0	1
1	1	1	0

(b) Excitation table for JK flip-flops

Q_n	Q_{n+1}	J	K
0	0	0	d
0	1	1	d
1	0	d	1
1	1	d	0

Sequential Circuit Design (cont'd)

- Build a design table that consists of
  - Current state output
  - Next state output
  - JK inputs for each flip-flop
- Binary counter example
  - 3-bit binary counter
  - 3 JK flip-flops are needed
  - Current state and next state outputs are 3 bits each
  - 3 pairs of JK inputs

Sequential Circuit Design (cont’d)

Design table for the binary counter example

Present state			Next state			JK flip-flop inputs					
A	B	C	A	B	C	J_A	K_A	J_B	K_B	J_C	K_C
0	0	0	0	0	1	0	d	0	d	1	d
0	0	1	0	1	0	0	d	1	d	d	1
0	1	0	0	1	1	0	d	d	0	1	d
0	1	1	1	0	0	1	d	d	1	d	1
1	0	0	1	0	1	d	0	0	d	1	d
1	0	1	1	1	0	d	0	1	d	d	1
1	1	0	1	1	1	d	0	d	0	1	d
1	1	1	0	0	0	d	1	d	1	d	1

Sequential Circuit Design (cont'd)

Use K-maps to simplify expressions for JK inputs

A \ BC	00	01	11	10
0	0	0	1	0
1	d	d	d	d

$J_A = B\,C$

A \ BC	00	01	11	10
0	d	d	d	d
1	0	0	1	0

$K_A = B\,C$

A \ BC	00	01	11	10
0	0	1	d	d
1	0	1	d	d

$J_B = C$

A \ BC	00	01	11	10
0	d	d	1	0
1	d	d	1	0

$K_B = C$

A \ BC	00	01	11	10
0	1	d	d	1
1	1	d	d	1

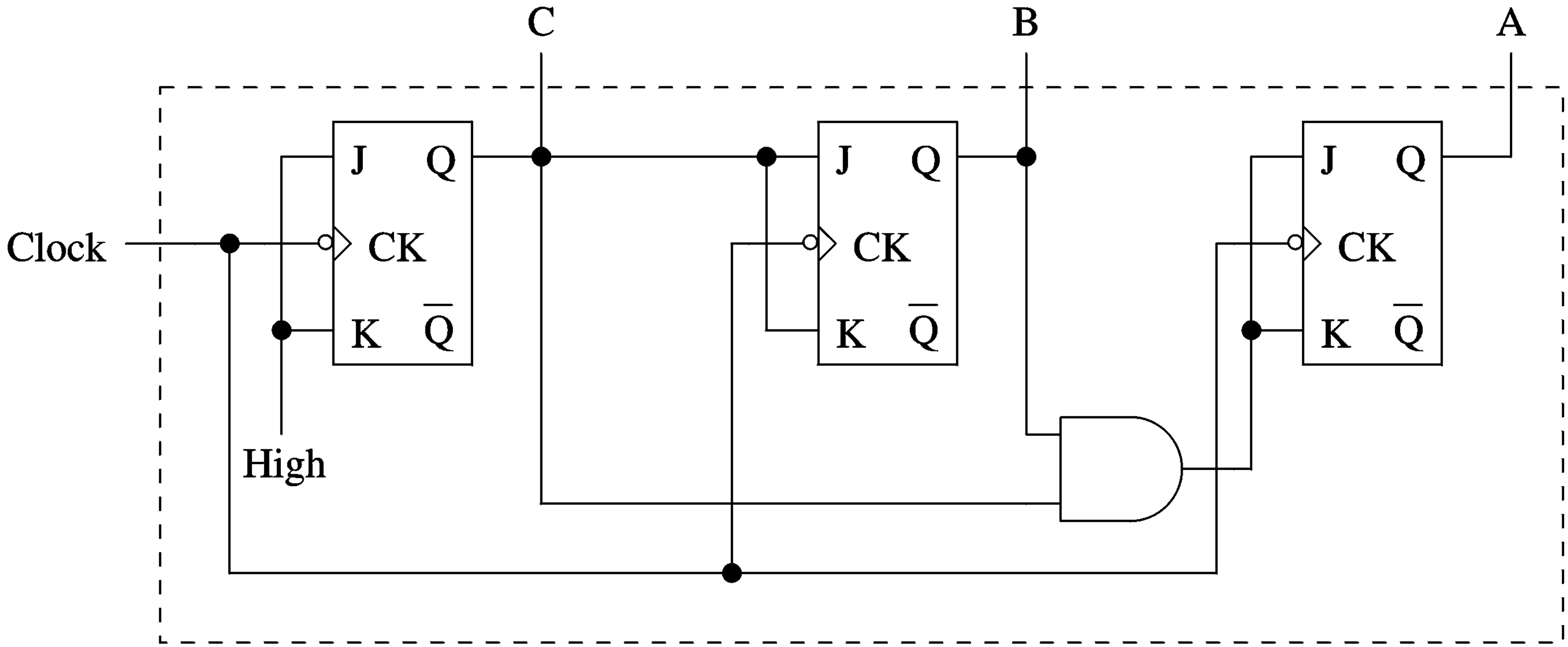
$J_C = 1$

A \ BC	00	01	11	10
0	d	1	1	d
1	d	1	1	d

$K_C = 1$

Sequential Circuit Design (cont'd)

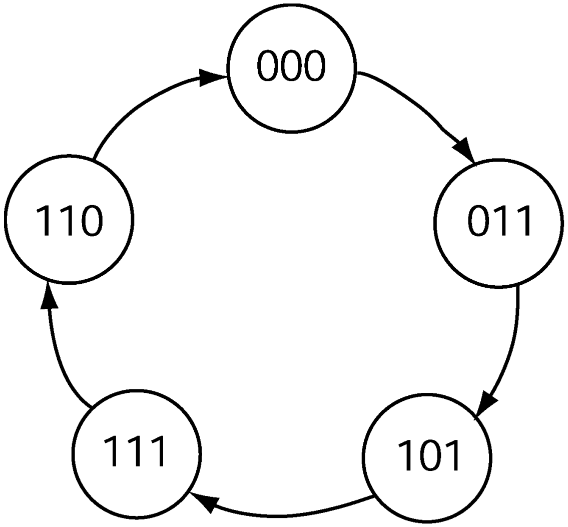
- Final circuit for the binary counter example
  - ∗ Compare this design with the synchronous counter design



Sequential Circuit Design (cont'd)

- A more general counter design
  - ∗ Does not step in sequence

0→3→5→7→6→0



- Same design process
- One significant change
  - ∗ Missing states
    - » 1, 2, and 4
    - » Use don't cares for these states

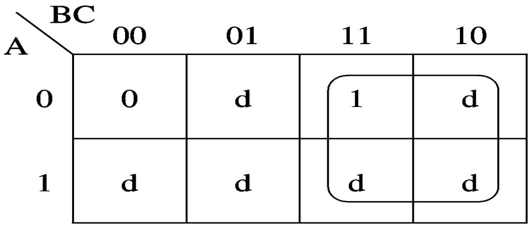
Sequential Circuit Design (cont’d)

Design table for the general counter example

Present state			Next state			JK flip-flop inputs					
A	B	C	A	B	C	J_A	K_A	J_B	K_B	J_C	K_C
0	0	0	0	1	1	0	d	1	d	1	d
0	0	1	–	–	–	d	d	d	d	d	d
0	1	0	–	–	–	d	d	d	d	d	d
0	1	1	1	0	1	1	d	d	1	d	0
1	0	0	–	–	–	d	d	d	d	d	d
1	0	1	1	1	1	d	0	1	d	d	0
1	1	0	0	0	0	d	1	d	1	0	d
1	1	1	1	1	0	d	0	d	0	d	1

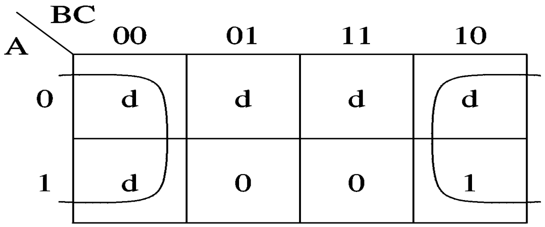
Sequential Circuit Design (cont'd)

K-maps to simplify JK input expressions



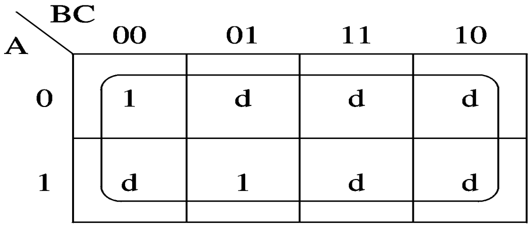
A \ BC	00	01	11	10
0	0	d	1	d
1	d	d	d	d

$J_A = B$



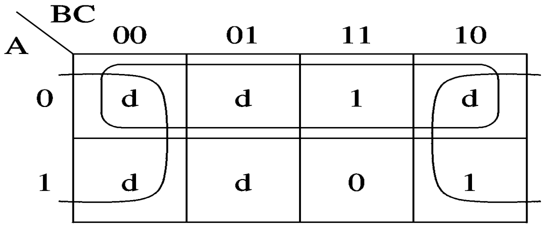
A \ BC	00	01	11	10
0	d	d	d	d
1	d	0	0	1

$K_A = \overline{C}$



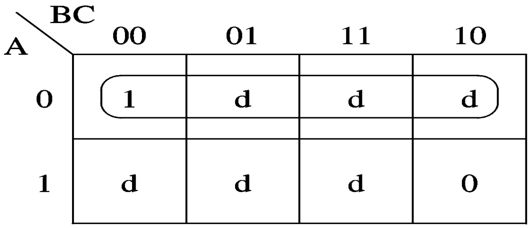
A \ BC	00	01	11	10
0	1	d	d	d
1	d	1	d	d

$J_B = 1$



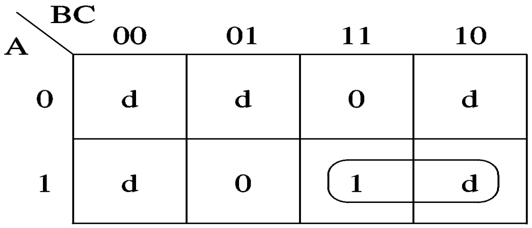
A \ BC	00	01	11	10
0	d	d	1	d
1	d	d	0	1

$K_B = \overline{A} + \overline{C}$



A \ BC	00	01	11	10
0	1	d	d	d
1	d	d	d	0

$J_C = \overline{A}$

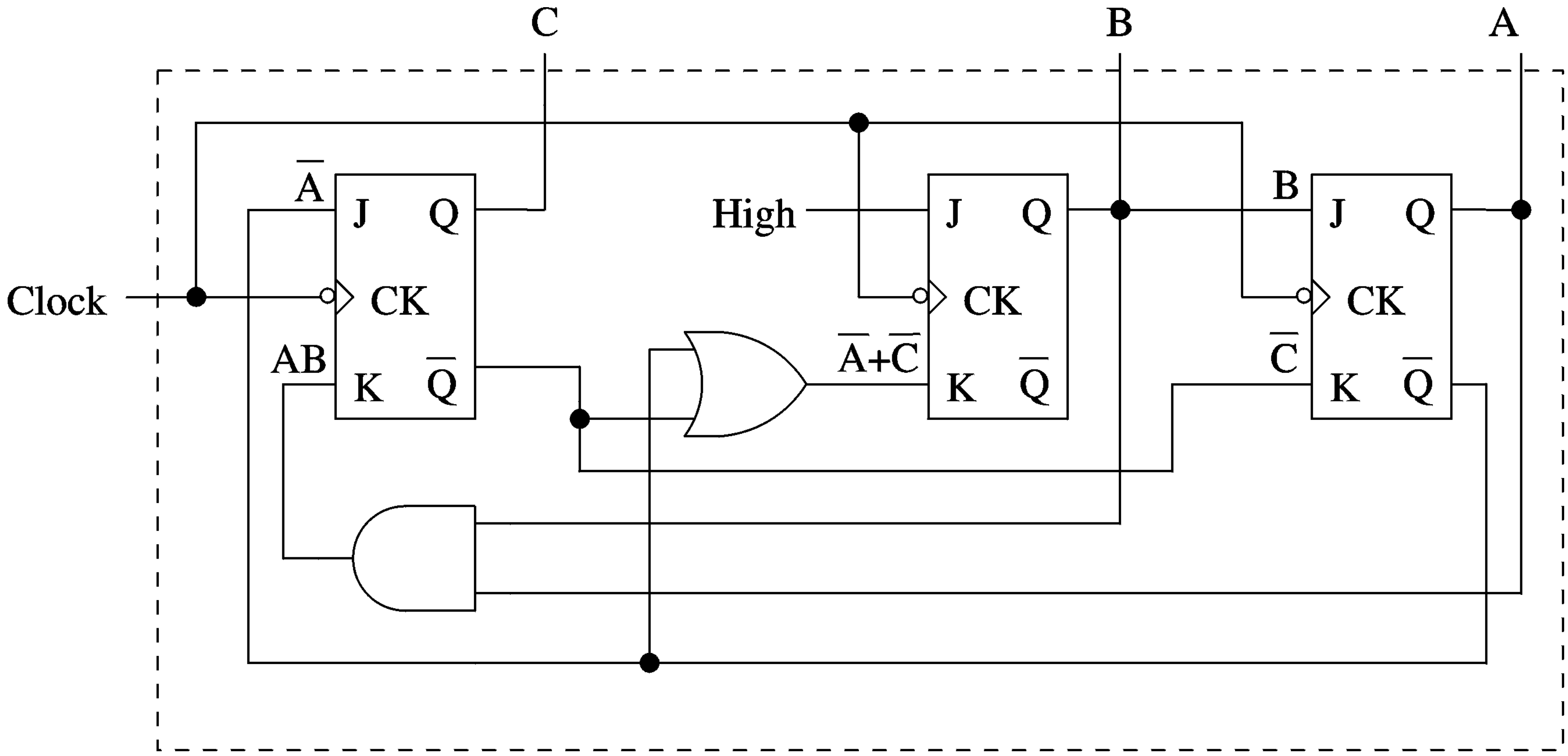


A \ BC	00	01	11	10
0	d	d	0	d
1	d	0	1	d

$K_C = A\,B$

Sequential Circuit Design (cont’d)

Final circuit for the general counter example

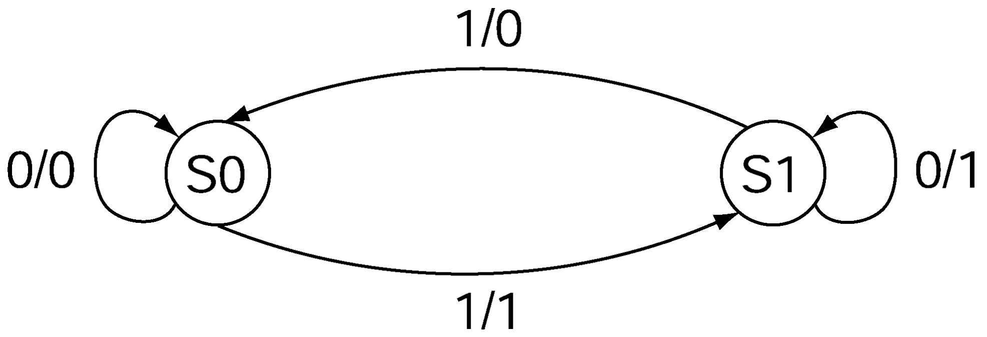


General Design Process

- FSM can be used to express the behavior of a sequential circuit
    - » Counters are a special case
    - ∗ State transitions are indicated by arrows with labels X/Y
        - » X: inputs that cause system state change
        - » Y: output generated while moving to the next state
- Look at two examples
    - ∗ Even-parity checker
    - ∗ Pattern recognition

General Design Process (cont’d)

- Even-parity checker
  - ∗ FSM needs to remember one of two facts
    - » Number of 1’s is odd or even
    - » Need only two states
      - – 0 input does not change the state
      - – 1 input changes state
  - ∗ Simple example
    - » Complete the design as an exercise

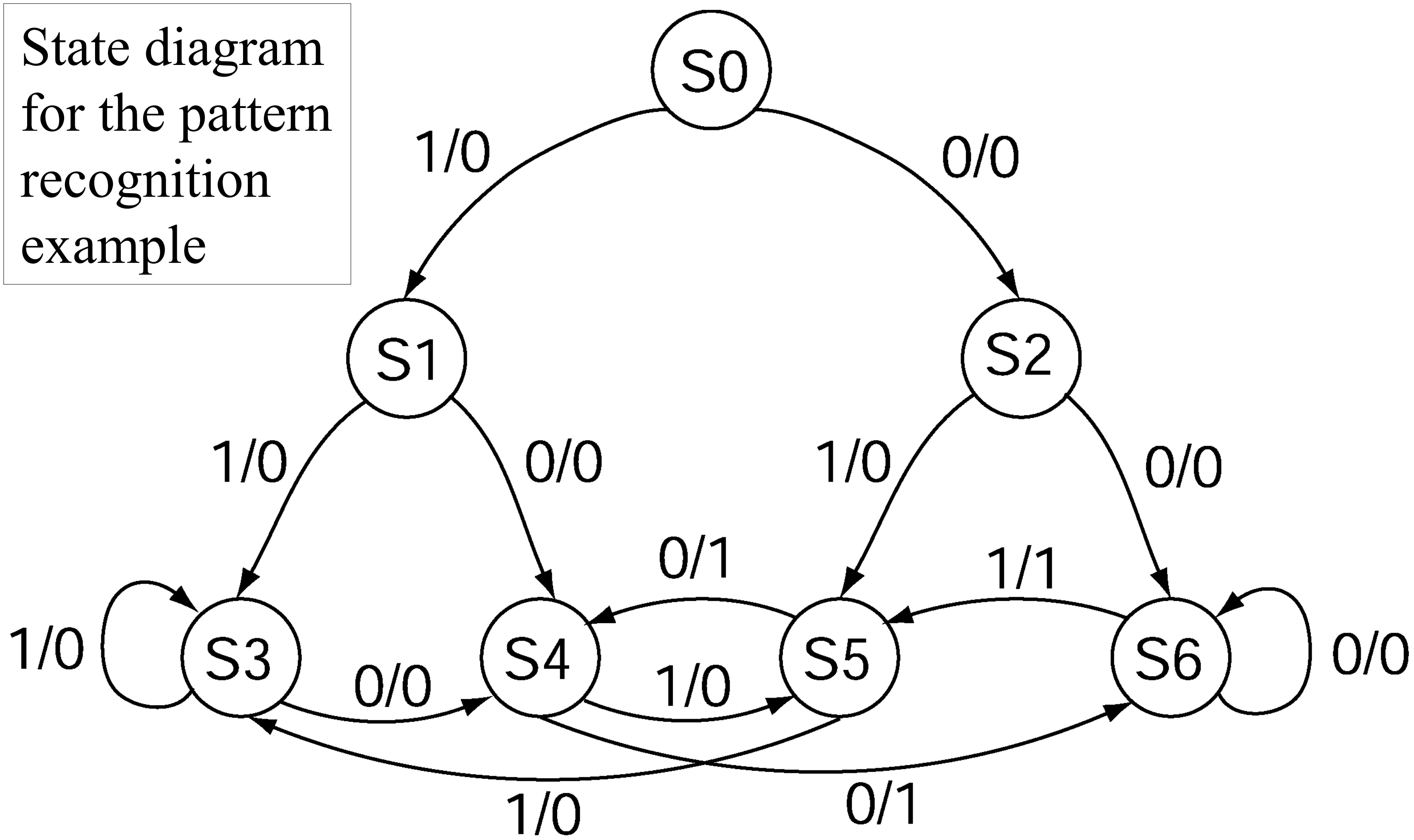


General Design Process (cont'd)

- Pattern recognition example
  - ∗ Outputs 1 whenever the input bit sequence has exactly two 0s in the last three input bits
  - ∗ FSM requires thee special states to during the initial phase
    - » S0 – S2
  - ∗ After that we need four states
    - » S3: last two bits are 11
    - » S4: last two bits are 01
    - » S5: last two bits are 10
    - » S6: last two bits are 00

General Design Process (cont'd)

State diagram for the pattern recognition example



General Design Process (cont'd)

- Steps in the design process
  1. Derive FSM
  2. State assignment
     - ∗ Assign flip-flop states to the FSM states
       - ∗ Necessary to get an efficient design
  3. Design table derivation
     - ∗ Derive a design table corresponding to the assignment in the last step
  4. Logical expression derivation
     - ∗ Use K-maps as in our previous examples
  5. Implementation

General Design Process (cont’d)

- State assignment
  - ∗ Three heuristics
    - » Assign adjacent states for
      - – states that have the same next state
      - – states that are the next states of the same state
      - – States that have the same output for a given input
  - ∗ For our example
    - » Heuristic 1 groupings: (S1, S3, S5)^2 (S2, S4, S6)^2
    - » Heuristic 2 groupings: (S1, S2) (S3, S4)^3 (S5, S6)^3
    - » Heuristic 1 groupings: (S4, S5)

General Design Process (cont’d)

State table for the pattern recognition example

	Next state		Output	
Present state	X = 0	X = 1	X = 0	X = 1
S0	S2	S1	0	0
S1	S4	S3	0	0
S2	S6	S5	0	0
S3	S4	S3	0	0
S4	S6	S5	1	0
S5	S4	S3	1	0
S6	S6	S5	0	1

General Design Process (cont’d)

K-map for state assignment

A \ BC	00	01	11	10
0	S0	S3	S5	S1
1		S4	S6	S2

State assignment

State		A	B	C
S0	=	0	0	0
S1	=	0	1	0
S2	=	1	1	0
S3	=	0	0	1
S4	=	1	0	1
S5	=	0	1	1
S6	=	1	1	1

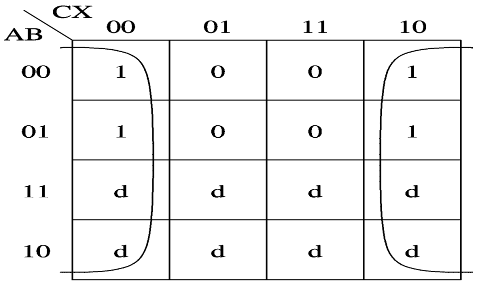
General Design Process (cont’d)

Design table

Present state			Present state	Next state			Present state	JK flip-flop inputs					
A	B	C	X	A	B	C	Y	J_A	K_A	J_B	K_B	J_C	K_C
0	0	0	0	1	1	0	0	1	d	1	d	0	d
0	0	0	1	0	1	0	0	0	d	1	d	0	d
0	0	1	0	1	0	1	0	1	d	0	d	d	0
0	0	1	1	0	0	1	0	0	d	0	d	d	0
0	1	0	0	1	0	1	0	1	d	d	1	1	d
0	1	0	1	0	0	1	0	0	d	d	1	1	d
0	1	1	0	1	0	1	1	1	d	d	1	d	0
0	1	1	1	0	0	1	0	0	d	d	1	d	0
1	0	1	0	1	1	1	1	d	0	1	d	d	0
1	0	1	1	0	1	1	0	d	1	1	d	d	0
1	1	0	0	1	1	1	0	d	0	d	0	1	d
1	1	0	1	0	1	1	0	d	1	d	0	1	d
1	1	1	0	1	1	1	0	d	0	d	0	d	0
1	1	1	1	0	1	1	1	d	1	d	0	d	0

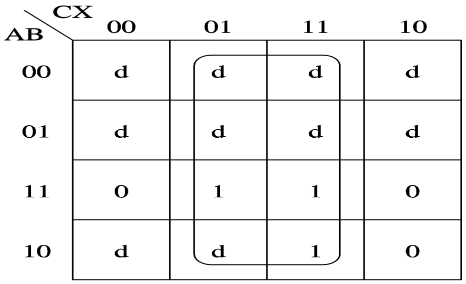
General Design Process (cont'd)

K-maps for JK inputs



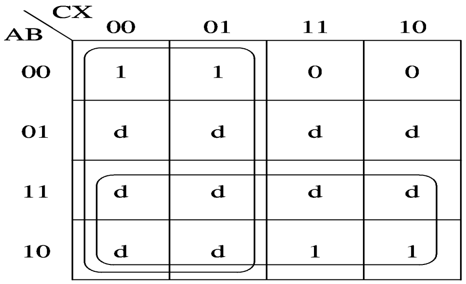
AB \ CX	00	01	11	10
00	1	0	0	1
01	1	0	0	1
11	d	d	d	d
10	d	d	d	d

$J_A = \overline{X}$



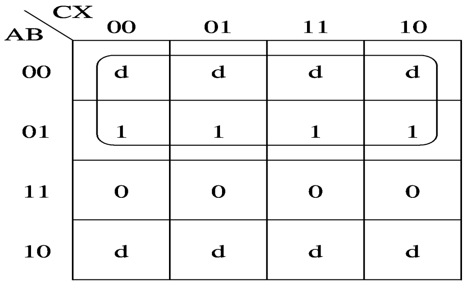
AB \ CX	00	01	11	10
00	d	d	d	d
01	d	d	d	d
11	0	1	1	0
10	d	d	1	0

$K_A = X$



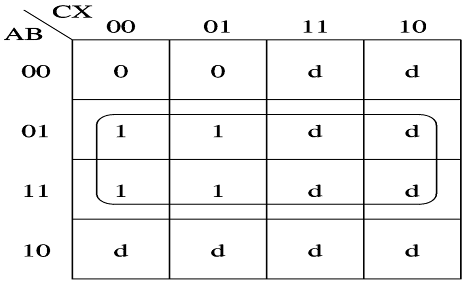
AB \ CX	00	01	11	10
00	1	1	0	0
01	d	d	d	d
11	d	d	d	d
10	d	d	1	1

$J_B = \overline{C} + A$



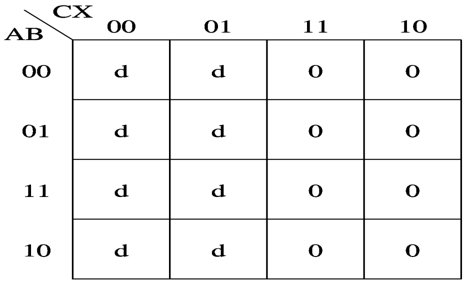
AB \ CX	00	01	11	10
00	d	d	d	d
01	1	1	1	1
11	0	0	0	0
10	d	d	d	d

$K_B = \overline{A}$



AB \ CX	00	01	11	10
00	0	0	d	d
01	1	1	d	d
11	1	1	d	d
10	d	d	d	d

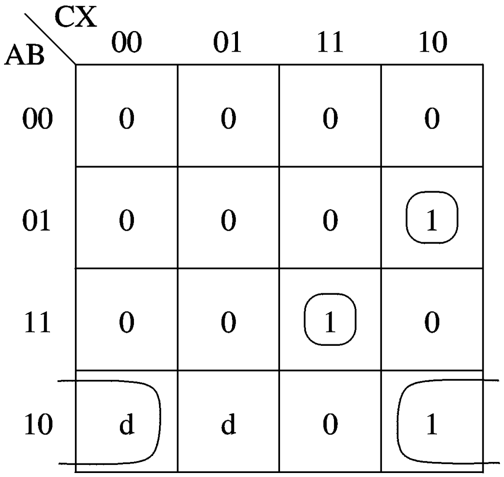
$J_C = B$



AB \ CX	00	01	11	10
00	d	d	0	0
01	d	d	0	0
11	d	d	0	0
10	d	d	0	0

$K_C = 0$

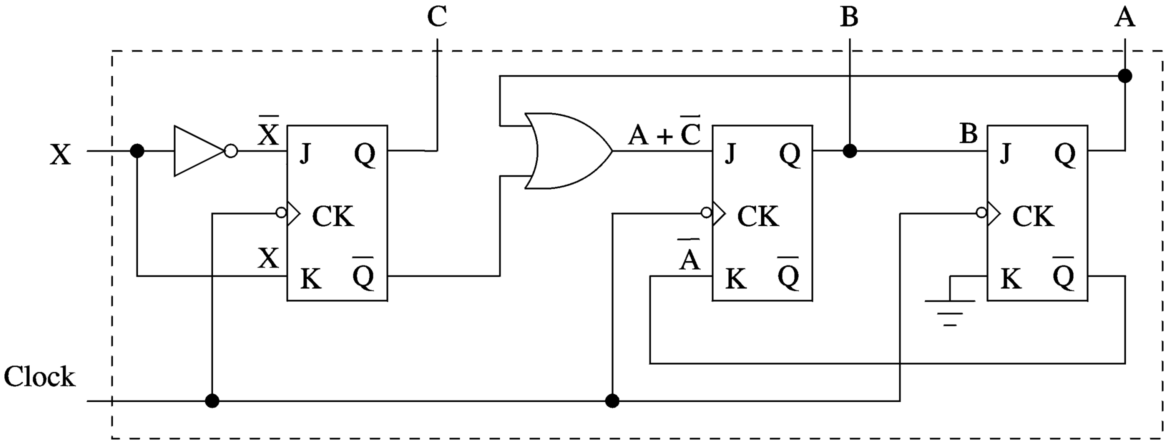
K-map for the output



AB \ CX	00	01	11	10
00	0	0	0	0
01	0	0	0	1
11	0	0	1	0
10	d	d	0	1

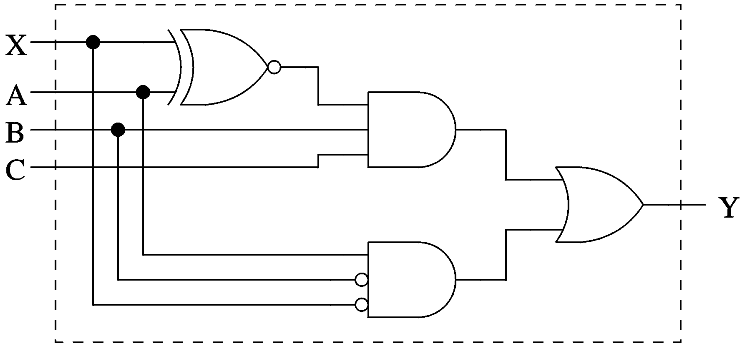
$Y = \overline{A}\,B\,C\,\overline{X} + A\,B\,C\,X + A\,\overline{B}\,\overline{X}$

General Design Process (cont’d)



(a)

Final implementation



(b)

Summary

- Output of a sequential circuit
  - ∗ Depends on the current input, and
  - ∗ Past history
- Typically consists of
  - ∗ A combinational circuit
  - ∗ A feedback circuit
- Provides “memory” property
  - ∗ Can be used to store a single bit of information
- Discussed sequential circuit design

Last slide