Digital Design

"Sequential Logic"

Dr. Cahit Karakuş, February-2018

Objectives of sequential logic

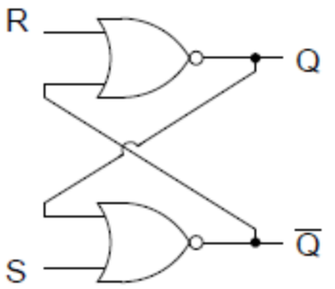
to discuss the difference between **combinational** and **sequential** logic as well as the difference between **asynchronous** and **synchronous** circuits and to show why the operation of synchronous circuits is more predictable, given propagation delays.

- to explain the operation of the common **latches** and **flip-flops**
  - SR or set–reset latch, which may also be called a SR flip-flop
  - D or data flip-flip
  - T or toggle flip-flop
  - JK flip-flop
- to describe clocking and the differences between **positive edge** and **negative edge** triggering and discuss the type of control inputs — active high and active low; asynchronous, jam or direct**.**

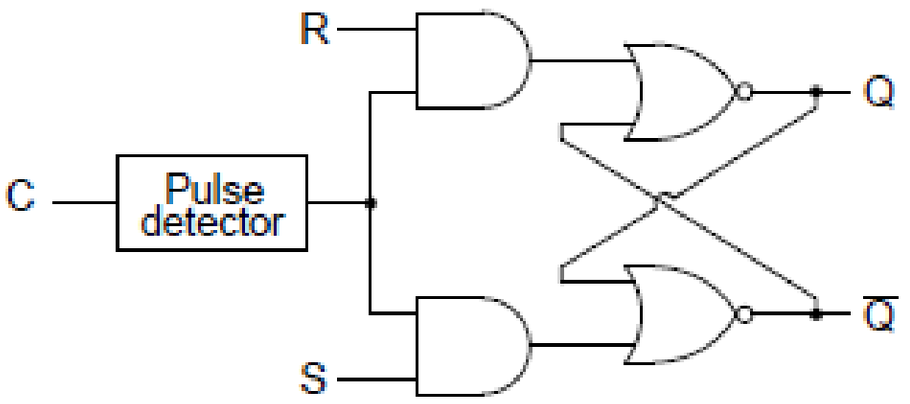
Sequential Logic

- Has **memory;** the circuit stores the result of the previous set of inputs. The current output depends on inputs **in the past** as well as present inputs.
  - The basic element in sequential logic is the **bistable latch** or **flip-flop**, which acts as a memory element for one bit of data.

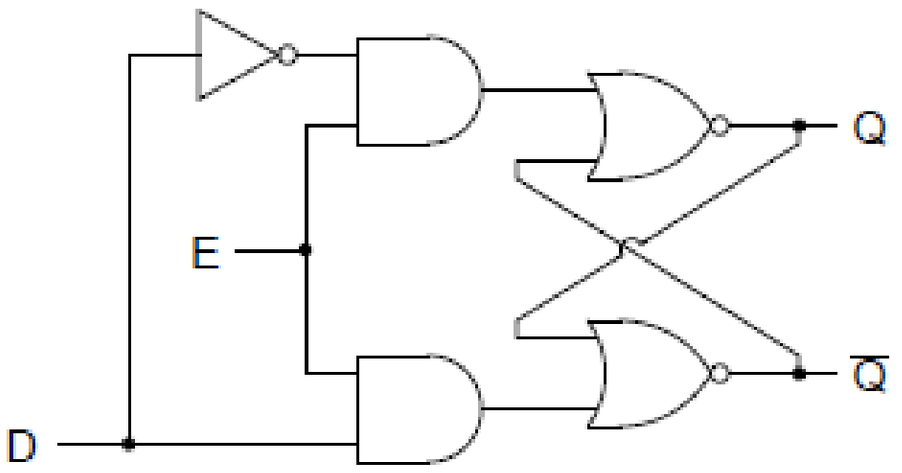
The Flip Flop



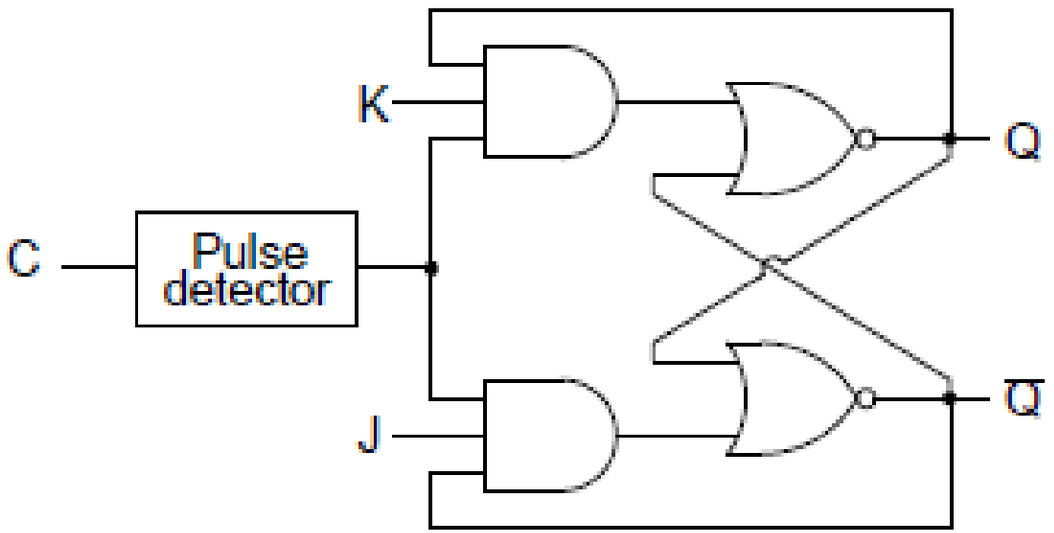
S	R	Q	$\overline{Q}$
0	0	latch	latch
0	1	0	1
1	0	1	0
1	1	0	0



C	S	R	Q	$\overline{Q}$
⌐	0	0	latch	latch
⌐	0	1	0	1
⌐	1	0	1	0
⌐	1	1	0	0
x	0	0	latch	latch
x	0	1	latch	latch
x	1	0	latch	latch
x	1	1	latch	latch



E	D	Q	$\overline{Q}$
0	0	latch	latch
0	1	latch	latch
1	0	0	1
1	1	1	0

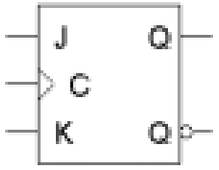


C	J	K	Q	$\overline{Q}$
⌐	0	0	latch	latch
⌐	0	1	0	1
⌐	1	0	1	0
⌐	1	1	toggle	toggle
x	0	0	latch	latch
x	0	1	latch	latch
x	1	0	latch	latch
x	1	1	latch	latch

Flip-flop çeşitleri

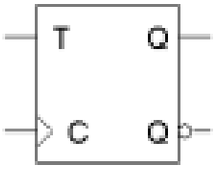
- D flip-flop'u temel alınarak çeşitli flip-floplar tasarlanmıştır.

- JK flip-flop (S ve R a benziyor ama JK=11 flip-flop un şimdiki durumunun tümleyenini almak için kullanılır.)



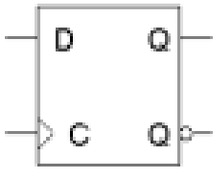
C	J	K	$Q_{gelecek}$
0	x	x	Değişmez
1	0	0	Değişmez
1	0	1	0 (reset)
1	1	0	1 (set)
1	1	1	$Q'_{şimdiki}$

- T flip-flop sadece şimdiki durumu tutar veya tümleyenini alır.

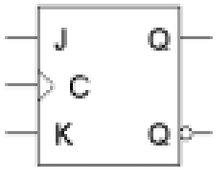


C	T	$Q_{gelecek}$
0	x	Değişmez
1	0	Değişmez
1	1	$Q'_{şimdiki}$

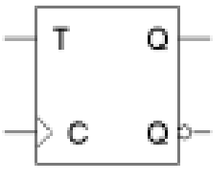
Tüm Flip-Flop(FF)lar için karakteristik tablolar



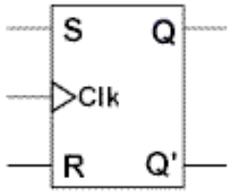
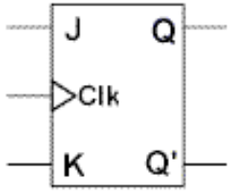
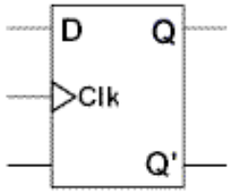
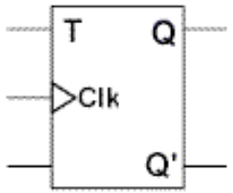
Q(t)	Q(t+1)	D	İşlem
0	0	0	Reset
0	1	1	Set
1	0	0	Reset
1	1	1	Set

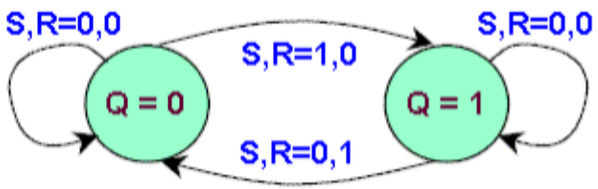
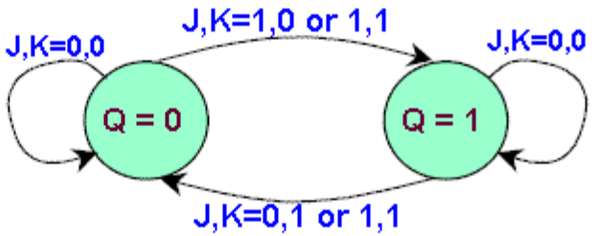
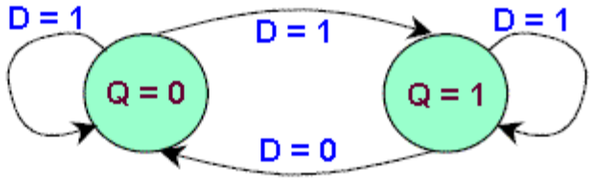
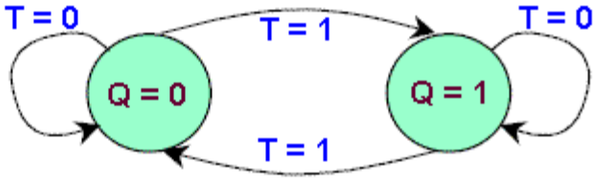


Q(t)	Q(t+1)	J	K	İşlem
0	0	0	x	Değişmez/reset
0	1	1	x	Set/tümleyen
1	0	x	1	Reset/tümleyen
1	1	x	0	Değişmez/set



Q(t)	Q(t+1)	T	Operation
0	0	0	Değişmez
0	1	1	Tümleyen
1	0	1	Tümleyen
1	1	0	Değişmez

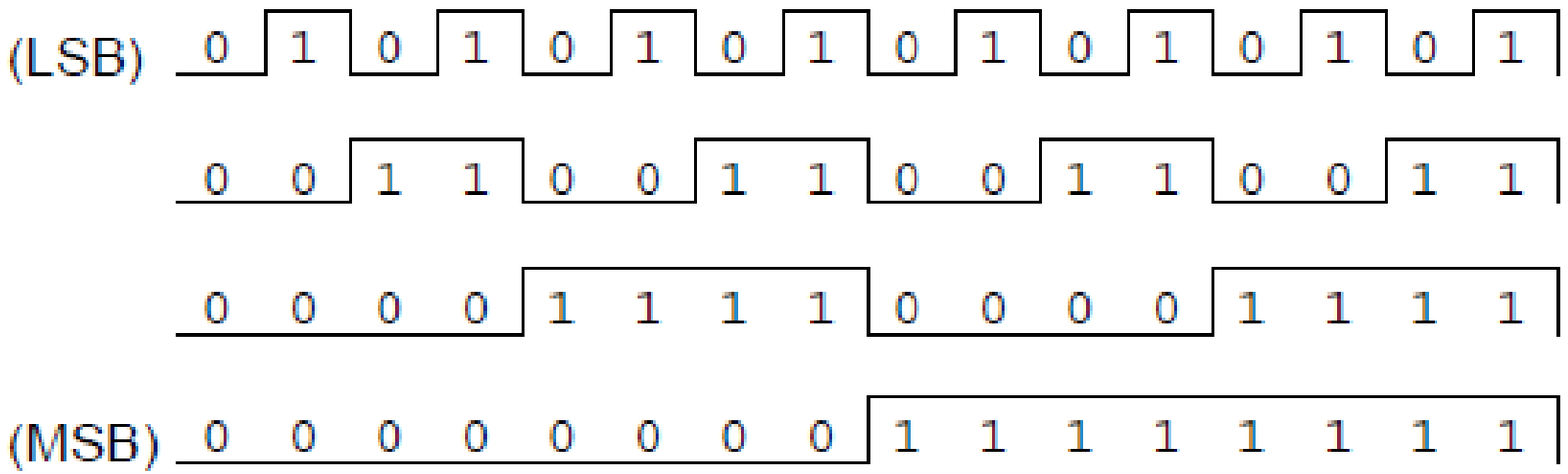
FLIP-FLOP NAME	FLIP-FLOP SYMBOL	CHARACTERISTIC EQUATION	EXCITATION TABLE
SR		$Q_{(next)} = S + R'Q$ $SR = 0$	**Q** | **$Q_{(next)}$** | **S** | **R** 0 | 0 | 0 | X 0 | 1 | 1 | 0 1 | 0 | 0 | 1 1 | 1 | X | 0
JK		$Q_{(next)} = JQ' + K'Q$	**Q** | **$Q_{(next)}$** | **J** | **K** 0 | 0 | 0 | X 0 | 1 | 1 | X 1 | 0 | X | 1 1 | 1 | X | 0
D		$Q_{(next)} = D$	**Q** | **$Q_{(next)}$** | **D** 0 | 0 | 0 0 | 1 | 1 1 | 0 | 0 1 | 1 | 1
T		$Q_{(next)} = TQ' + T'Q$	**Q** | **$Q_{(next)}$** | **T** 0 | 0 | 0 0 | 1 | 1 1 | 0 | 1 1 | 1 | 0

NAME	STATE DIAGRAM
SR	
JK	
D	
T	

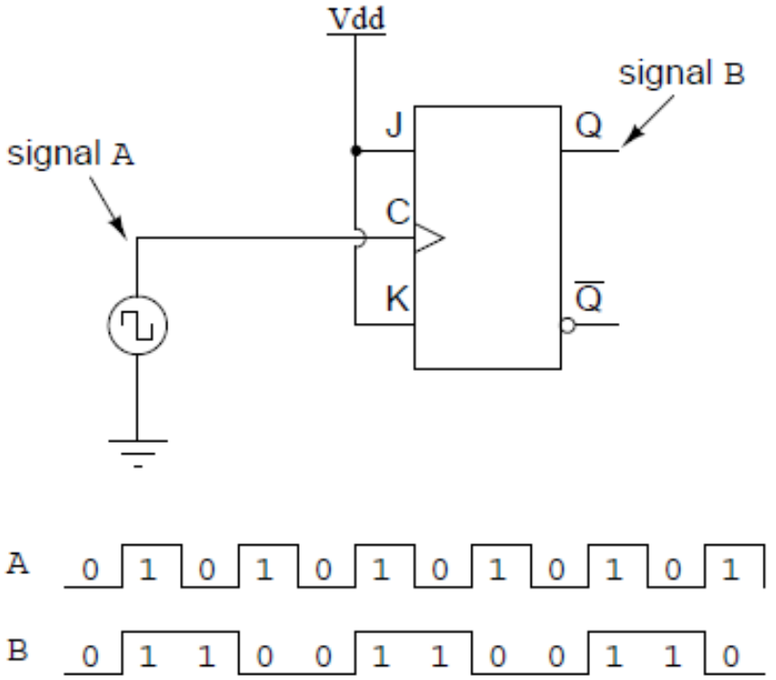
COUNTERS

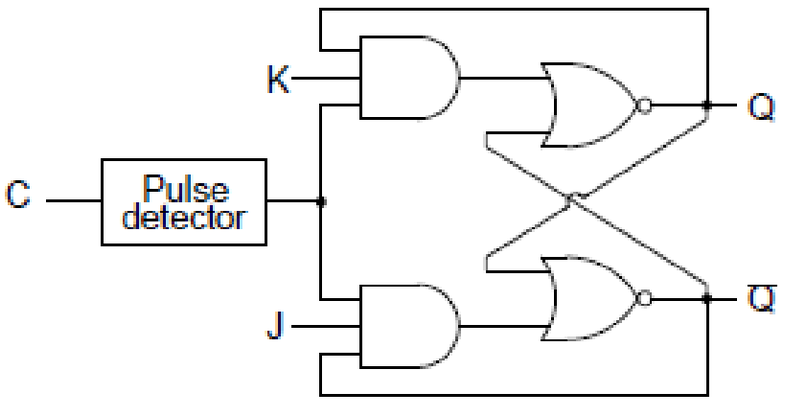
Binary count sequence

- If we examine a four-bit binary count sequence from 0000 to 1111, a definite pattern will be evident in the "oscillations" of the bits between 0 and 1



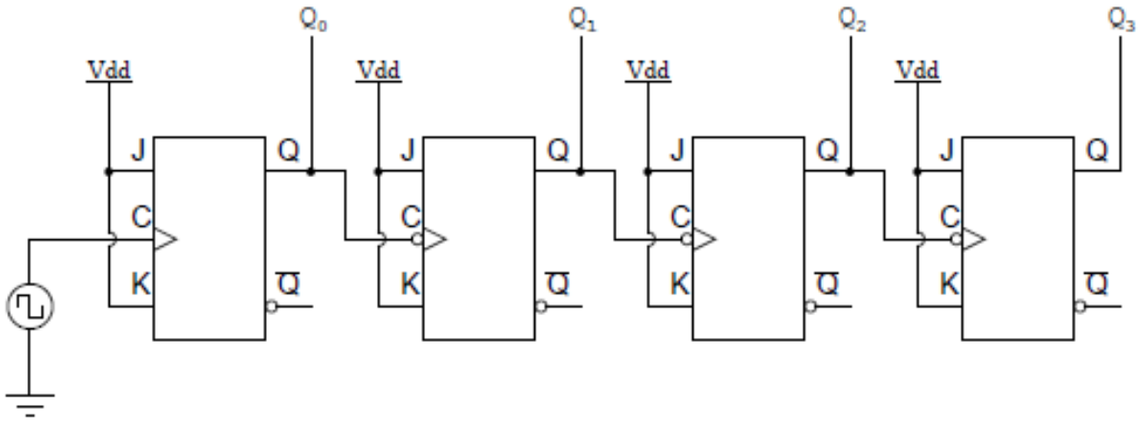
0 0 0 0
0 0 0 1
0 0 1 0
0 0 1 1
0 1 0 0
0 1 0 1
0 1 1 0
0 1 1 1
1 0 0 0
1 0 0 1
1 0 1 0
1 0 1 1
1 1 0 0
1 1 0 1
1 1 1 0
1 1 1 1

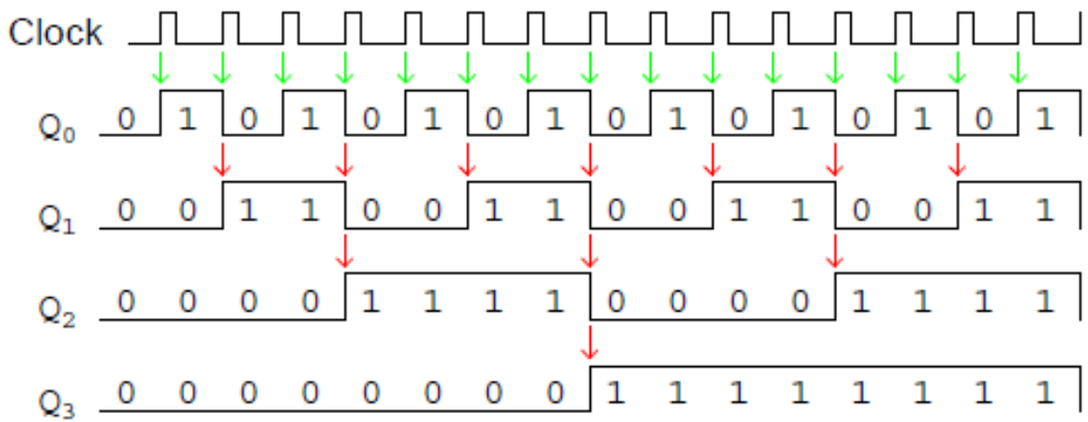




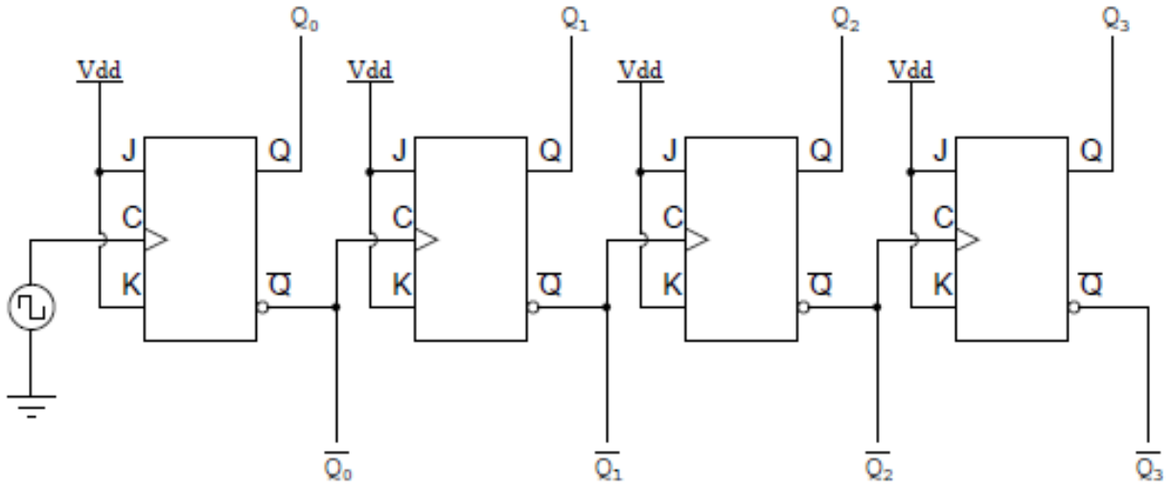
C	J	K	Q	$\overline{Q}$
⌐	0	0	latch	latch
⌐	0	1	0	1
⌐	1	0	1	0
⌐	1	1	toggle	toggle
x	0	0	latch	latch
x	0	1	latch	latch
x	1	0	latch	latch
x	1	1	latch	latch

A four-bit "up" counter

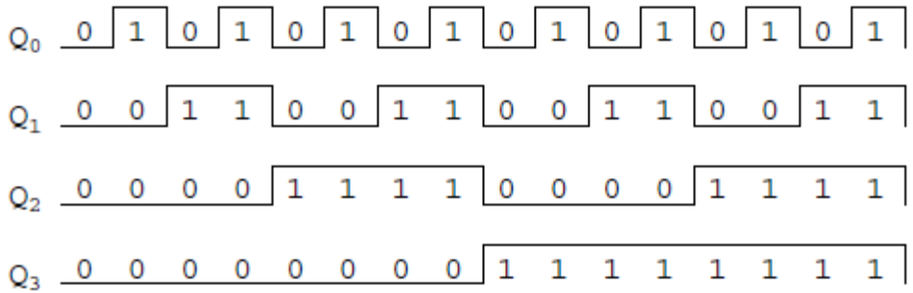




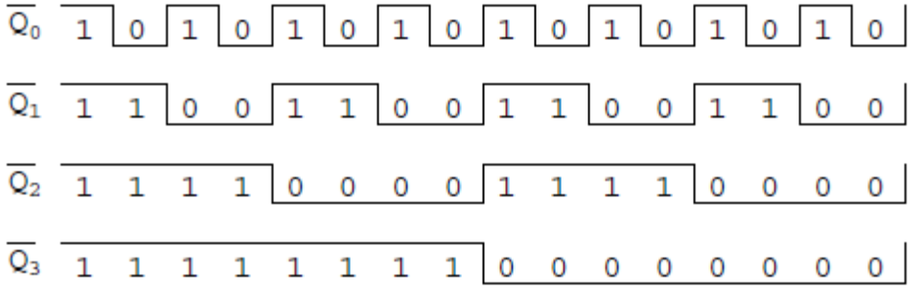
A simultaneous "up" and "down" counter



"Up" count sequence

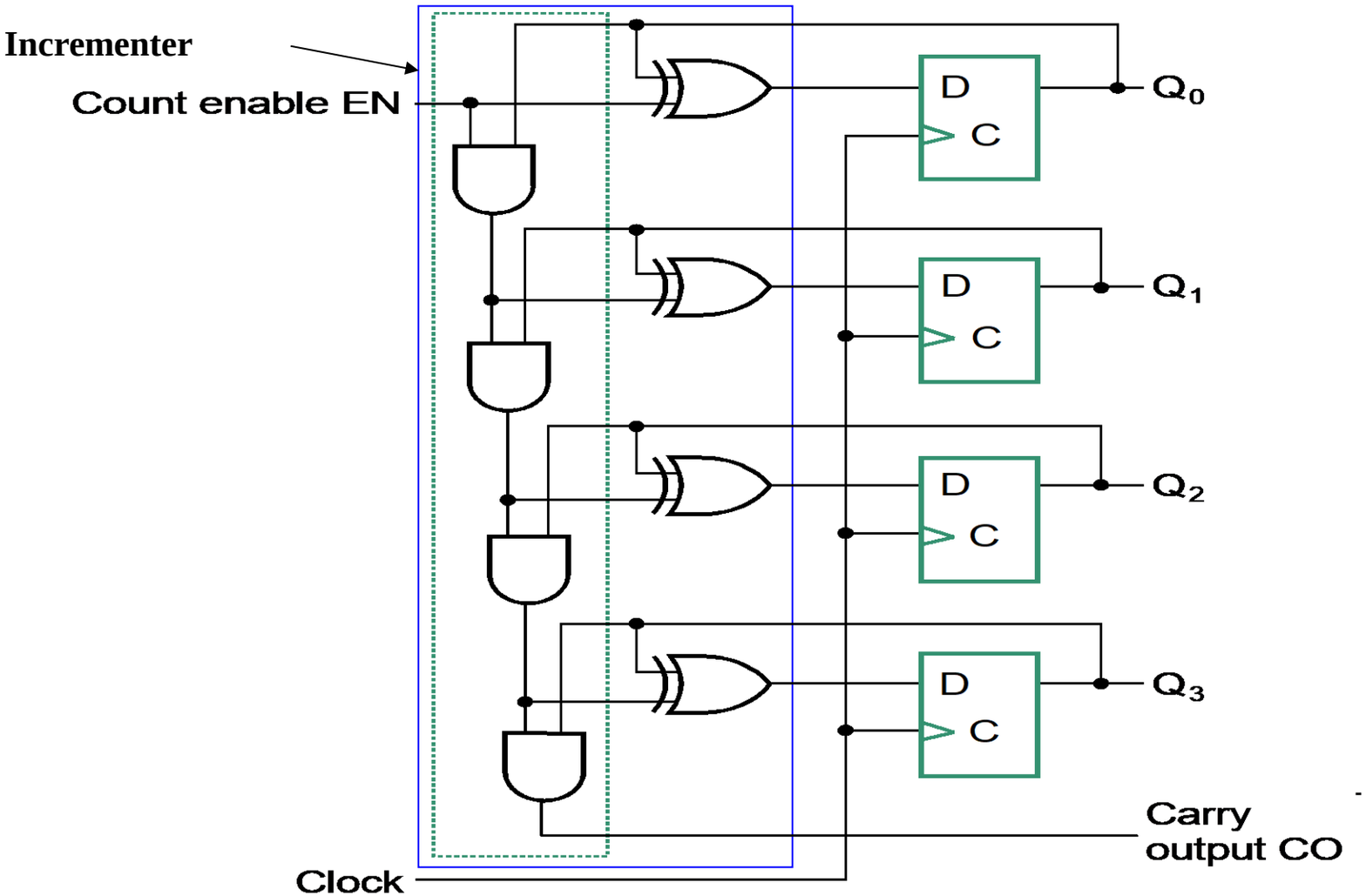


"Down" count sequence



Synchronous Counters (continued)

- Internal details =>
- Internal Logic
  - XOR complements each bit
  - AND chain causes complement of a bit if all bits toward LSB from it equal 1
- Count Enable
  - Forces all outputs of AND chain to 0 to “hold” the state
- Carry Out
  - Added as part of incrementer
  - Connect to Count Enable of additional 4-bit counters to form larger counters

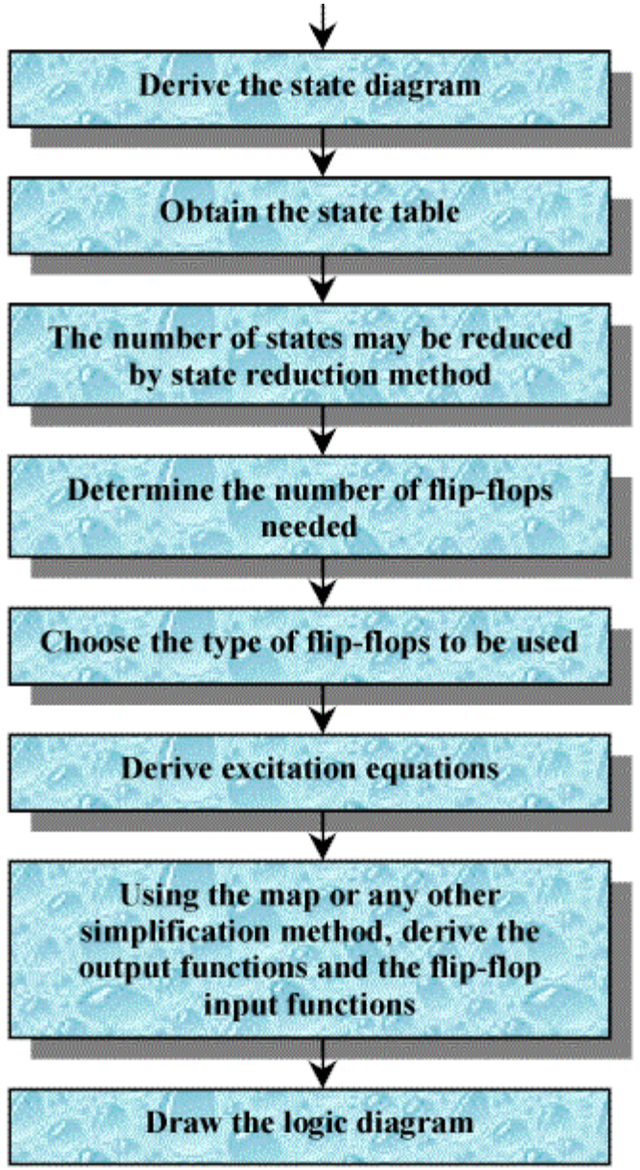


(a) Logic Diagram-Serial Gating

State Table

State Table

- *State table* – a multiple variable table with the following four sections:
  - *Present State* – the values of the state variables for each allowed state.
  - *Input* – the input combinations allowed.
  - *Next-state* – the value of the state at time (t+1) based on the present state and the input.
  - *Output* – the value of the output as a function of the present state and (sometimes) the input.
- From the viewpoint of a truth table:
  - the inputs are Input, Present State
  - and the outputs are Output, Next State


Derive the state diagram
Obtain the state table
The number of states may be reduced by state reduction method
Determine the number of flip-flops needed
Choose the type of flip-flops to be used
Derive excitation equations
Using the map or any other simplification method, derive the output functions and the flip-flop input functions
Draw the logic diagram

State Table

- The state table can be filled in using the next state and output equations:
  - $A(t+1) = A(t)x(t) + B(t)x(t)$
  - $B(t+1) = \overline{A}(t)x(t)$;
  - $y(t) = \overline{x}(t)(B(t) + A(t))$

Present State		Input	Next State		Output
A(t)	B(t)	x(t)	A(t+1)	B(t+1)	y(t)
0	0	0	0	0	0
0	0	1	0	1	0
0	1	0	0	0	1
0	1	1	1	1	0
1	0	0	0	0	1
1	0	1	1	0	0
1	1	0	0	0	1
1	1	1	1	0	0

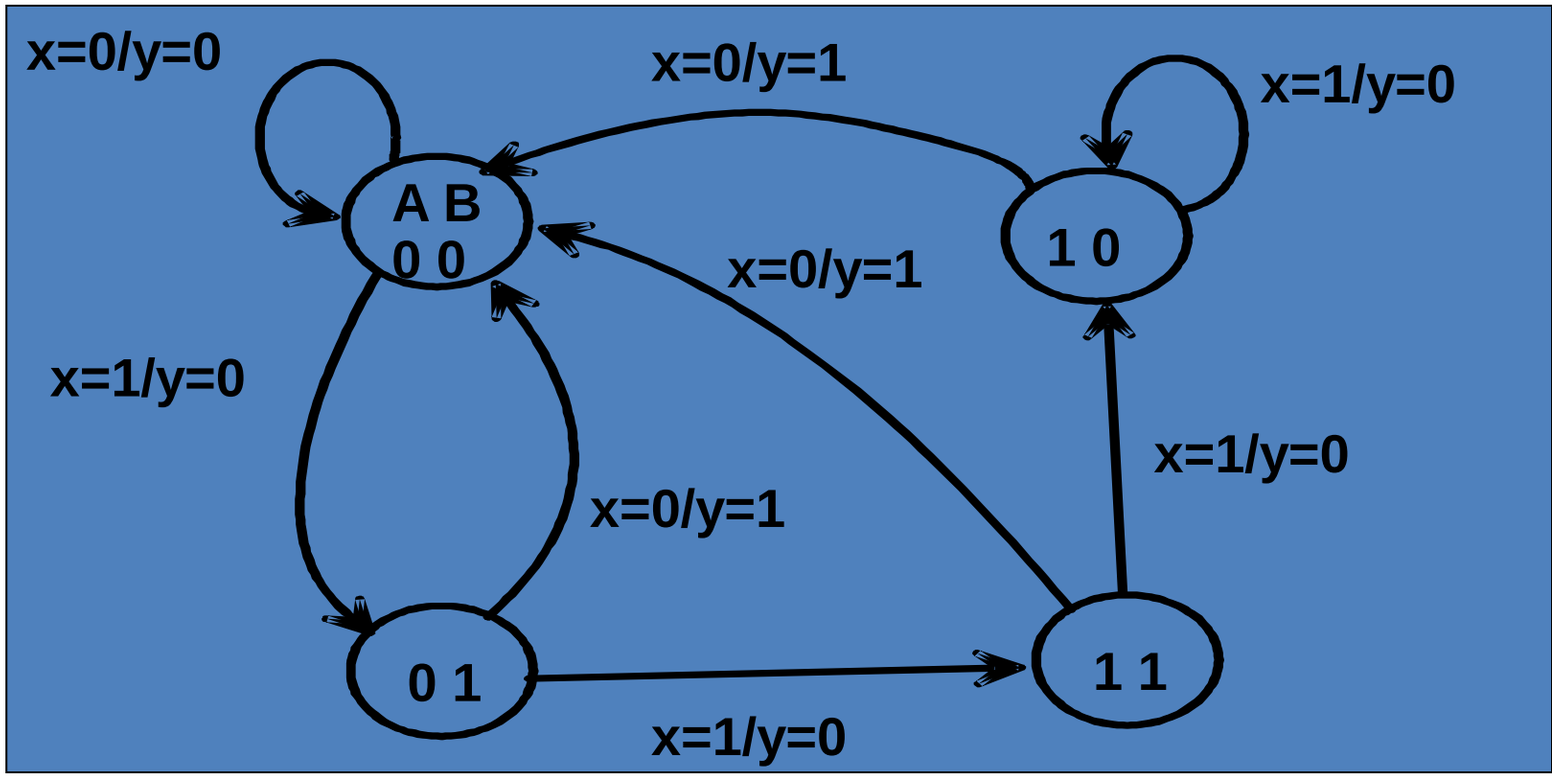
State Diagrams

- The sequential circuit function can be represented in graphical form as a state diagram with the following components:
  - A circle with the state name in it for each state
  - A directed arc from the Present State to the Next State for each state transition
  - A label on each directed arc with the Input values which causes the state transition, and
  - A label:
    - On each circle with the output value produced, or
    - On each directed arc with the output value produced.

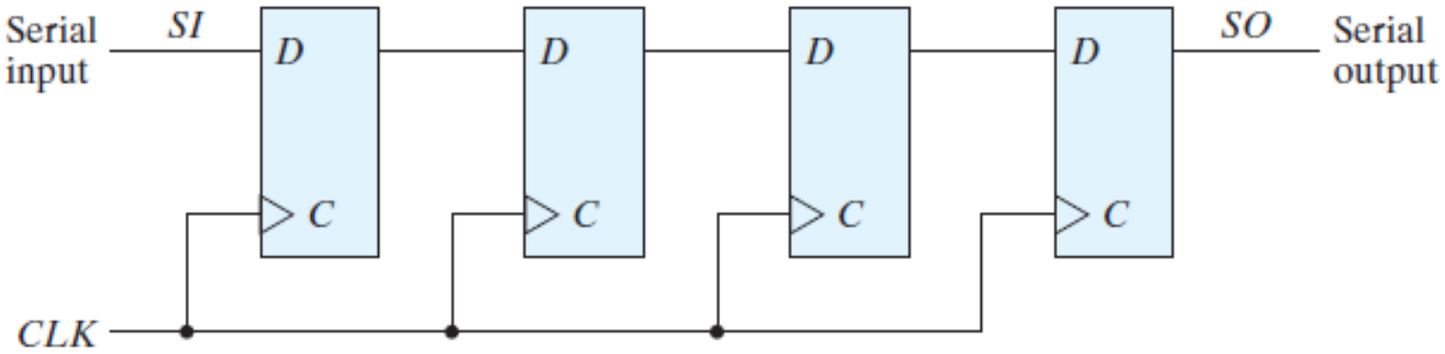
Examples

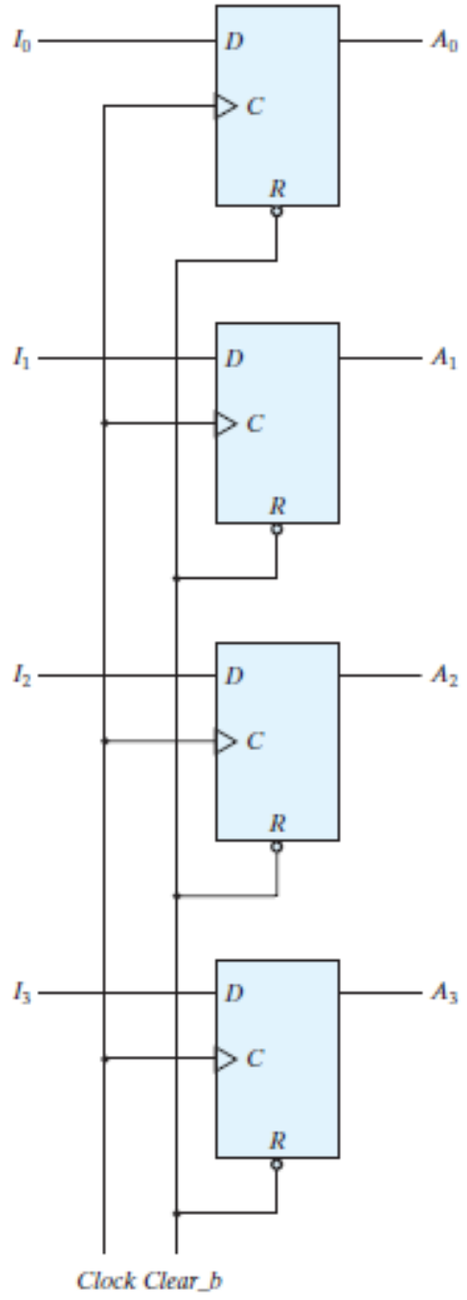
Example : State Diagram

- Which type?
- Diagram gets confusing for large circuits
- For small circuits, usually easier to understand than the state table

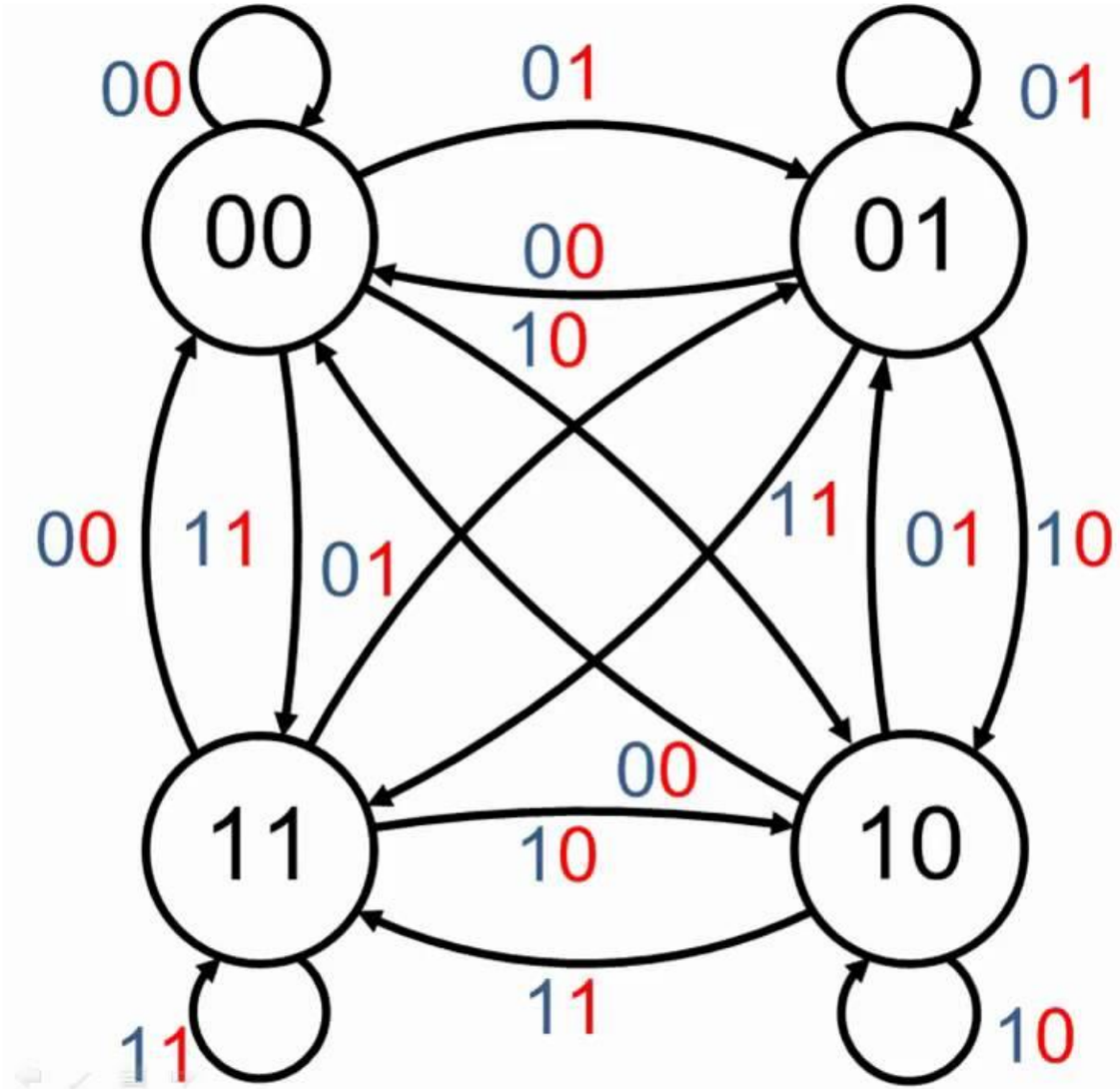


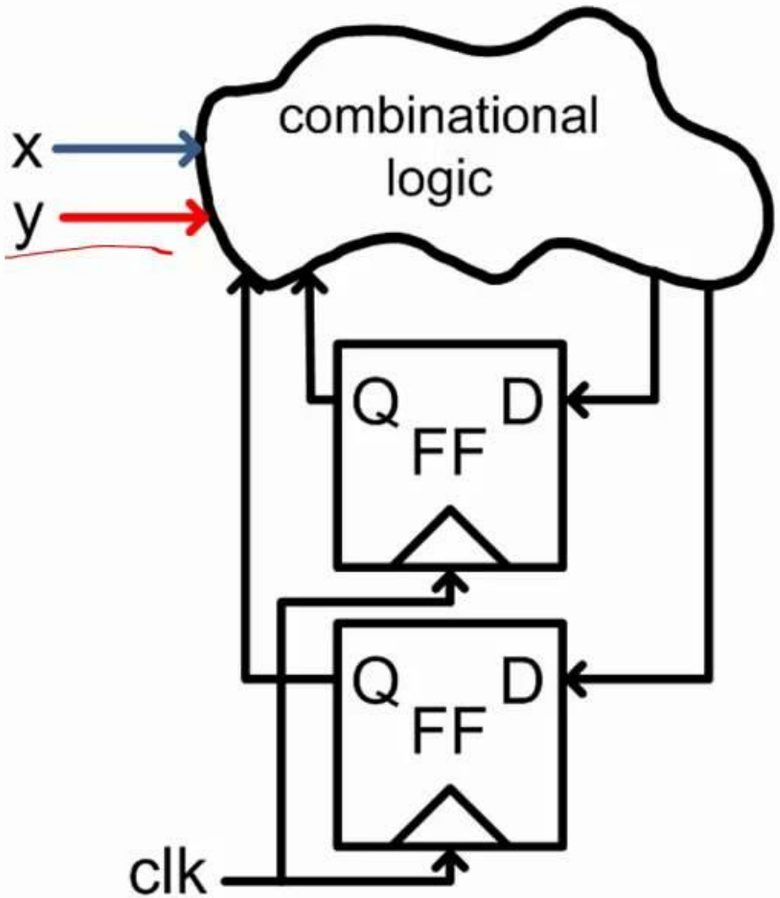
Registers


Serial input
SI
D
C
D
C
D
C
D
C
SO
Serial output
CLK

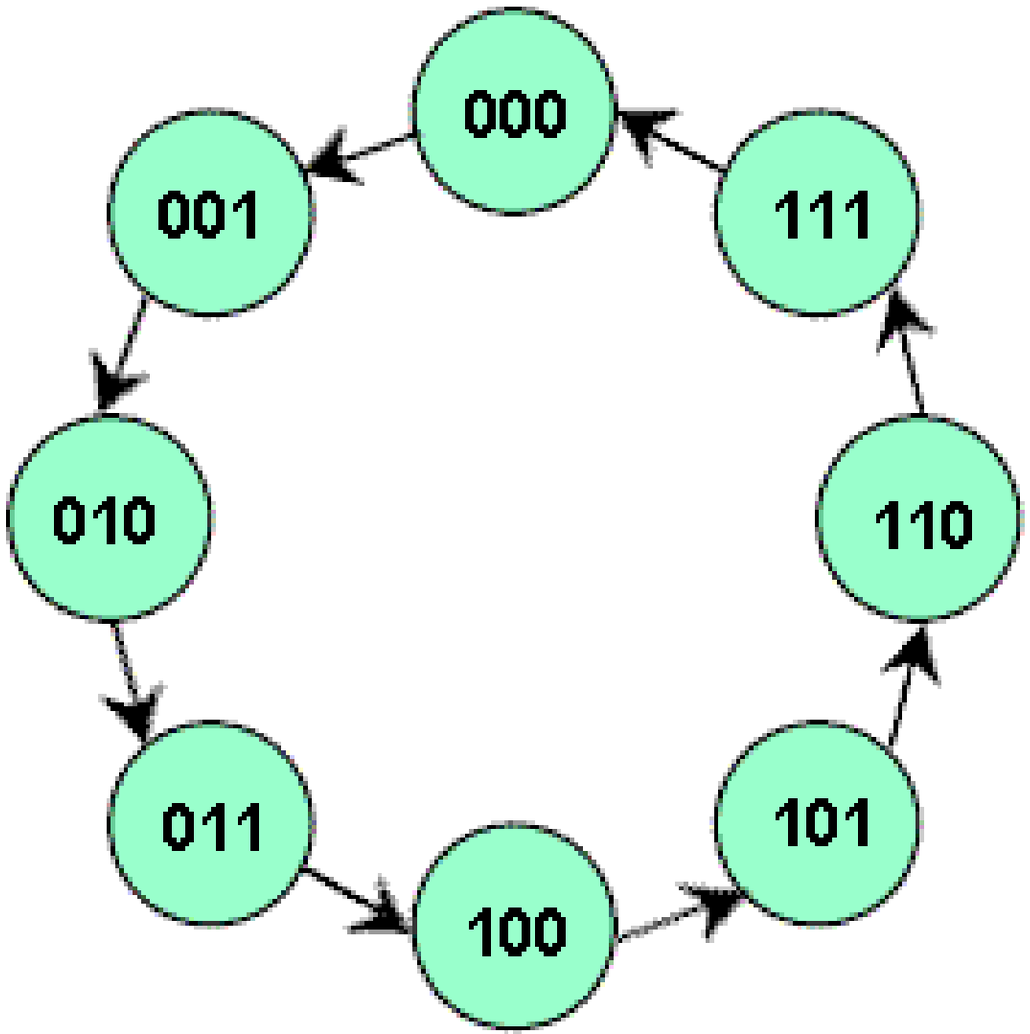

I_0
D
C
R
A_0
I_1
D
C
R
A_1
I_2
D
C
R
A_2
I_3
D
C
R
A_3
Clock Clear_b

Example: Durum Diyagramı





Example: Sayıcı –JK Flip Flop



Tetikleme - Uyarma

Output State Transitions		Flip-flop inputs		
Present State Q2 Q1 Q0	Next State Q2 Q1 Q0	J2 K2	J1 K1	J0 K0
0 0 0	0 0 1	0 X	0 X	1 X
0 0 1	0 1 0	0 X	1 X	X 1
0 1 0	0 1 1	0 X	X 0	1 X
0 1 1	1 0 0	1 X	X 1	X 1
1 0 0	1 0 1	X 0	0 X	1 X
1 0 1	1 1 0	X 0	1 X	X 1
1 1 0	1 1 1	X 0	X 0	1 X
1 1 1	0 0 0	X 1	X 1	X 1

C	J	K	Q	$\overline{Q}$
⌐	0	0	latch	latch
⌐	0	1	0	1
⌐	1	0	1	0
⌐	1	1	toggle	toggle
x	0	0	latch	latch
x	0	1	latch	latch
x	1	0	latch	latch
x	1	1	latch	latch

Q2Q1 \ Q0	0	1
00	0	0
01	0	1
11	X	X
10	X	X

J2 map

Q2Q1 \ Q0	0	1
00	0	1
01	X	X
11	X	X
10	0	1

J1 map

Q2Q1 \ Q0	0	1
00	1	X
01	1	X
11	1	X
10	1	X

J0 map

Q2Q1 \ Q0	0	1
00	X	X
01	X	X
11	0	1
10	0	0

K2 map

Q2Q1 \ Q0	0	1
00	X	X
01	0	1
11	0	1
10	X	X

K1 map

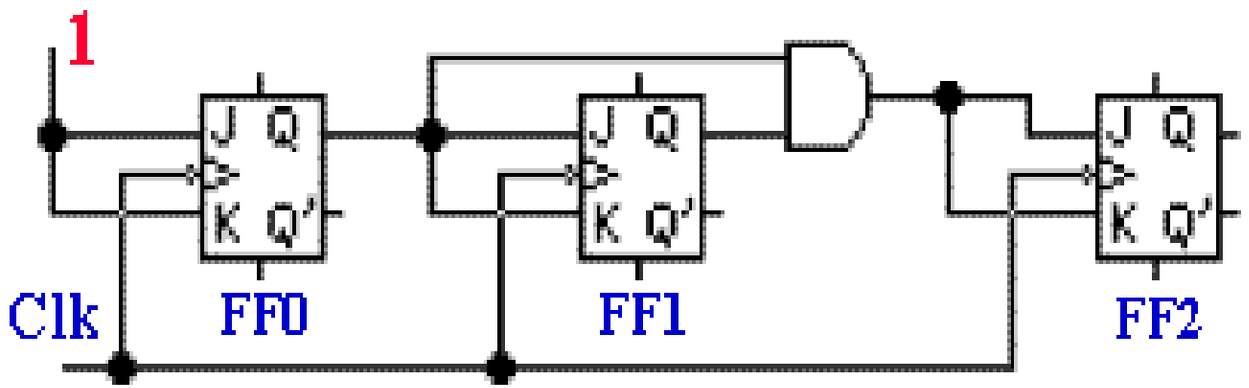
Q2Q1 \ Q0	0	1
00	X	1
01	X	1
11	X	1
10	X	1

K0 map

J0 = K0 = 1

J1 = K1 = Q0

J2 = K2 = Q1*Q0



Output State Transitions		Flip-flop inputs
Present State Q2 Q1 Q0	Next State Q2 Q1 Q0	T2 T1 T0
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

Q2Q1 \ Q0	0	1
00	0	0
01	0	1
11	0	1
10	0	0

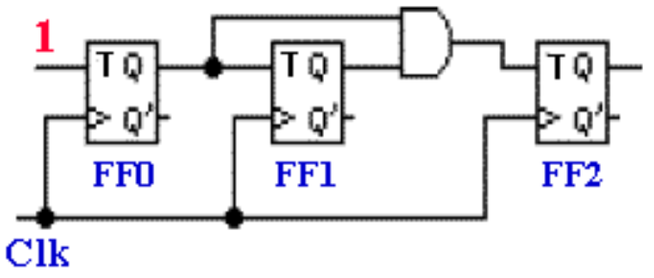
T2 map

Q2Q1 \ Q0	0	1
00	0	1
01	0	1
11	0	1
10	0	1

T1 map

Q2Q1 \ Q0	0	1
00	1	1
01	1	1
11	1	1
10	1	1

T0 map



T0 = 1; T1 = Q0; T2 = Q1*Q0

- JK karakteristik tablosu kullanılarak *her bir* flip-flop'un girişleri bulunur. (şimdiki ve gelecek durumlara bakarak)

Q(t)	Q(t+1)	J	K
0	0	0	x
0	1	1	x
1	0	x	1
1	1	x	0

Şimdiki durum		Giriş	Gelecek durum		Flip flop girişleri				Çıkış
Q_1	Q_0	X	Q_1	Q_0	J_1	K_1	J_0	K_0	Z
0	0	0	0	0	0	x	0	x	0
0	0	1	0	1	0	x	1	x	0
0	1	0	1	0	1	x	x	1	0
0	1	1	0	1	0	x	x	0	0
1	0	0	1	1	x	0	1	x	0
1	0	1	0	1	x	1	1	x	0
1	1	0	0	0	x	1	x	1	0
1	1	1	0	1	x	1	x	0	1

- Karnaugh diyagramı yardımıyla dört flip-flop'un her biri için girişlere ve çıkışa ait denklemler bulunur.
- Bu denklemler şimdiki durum ve girişler cinsindendir.
- JK FF kullanmanın avantajı: birkaç tane don't care durumuna sahip olmalarıdır. Bu sayede daha basit denklemler elde edilir.

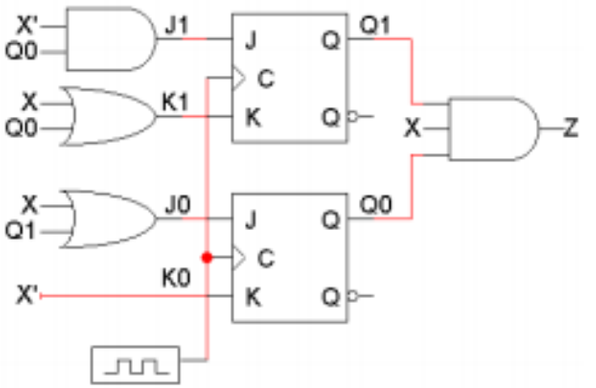
$J_1 = X' Q_0$ $\quad$ $J_0 = X + Q_1$ $\quad$ $Z = Q_1 Q_0 X$

$K_1 = X + Q_0$ $\quad$ $K_0 = X'$

$J_1 = X' Q_0$
$K_1 = X + Q_0$

$J_0 = X + Q_1$
$K_0 = X'$

$Z = Q_1 Q_0 X$



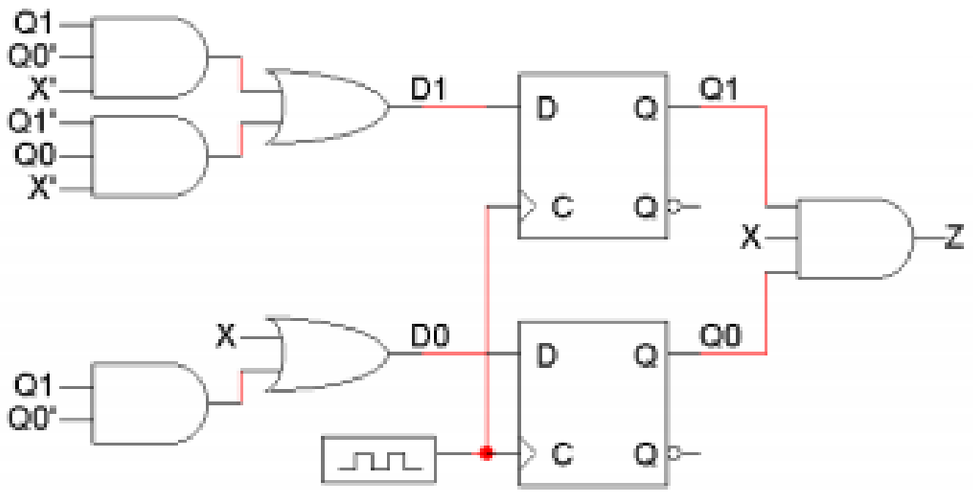
- D FF karakteristik tablosu:

Q(t)	Q(t+1)	D	İşlem
0	0	0	Reset
0	1	1	Set
1	0	0	Reset
1	1	1	Set

Şimdiki durum		Giriş	Gelecek durum		Flip flop girişleri		Çıkış
Q_1	Q_0	X	Q_1	Q_0	D_1	D_0	Z
0	0	0	0	0	0	0	0
0	0	1	0	1	0	1	0
0	1	0	1	0	1	0	0
0	1	1	0	1	0	1	0
1	0	0	1	1	1	1	0
1	0	1	0	1	0	1	0
1	1	0	0	0	0	0	0
1	1	1	0	1	0	1	1

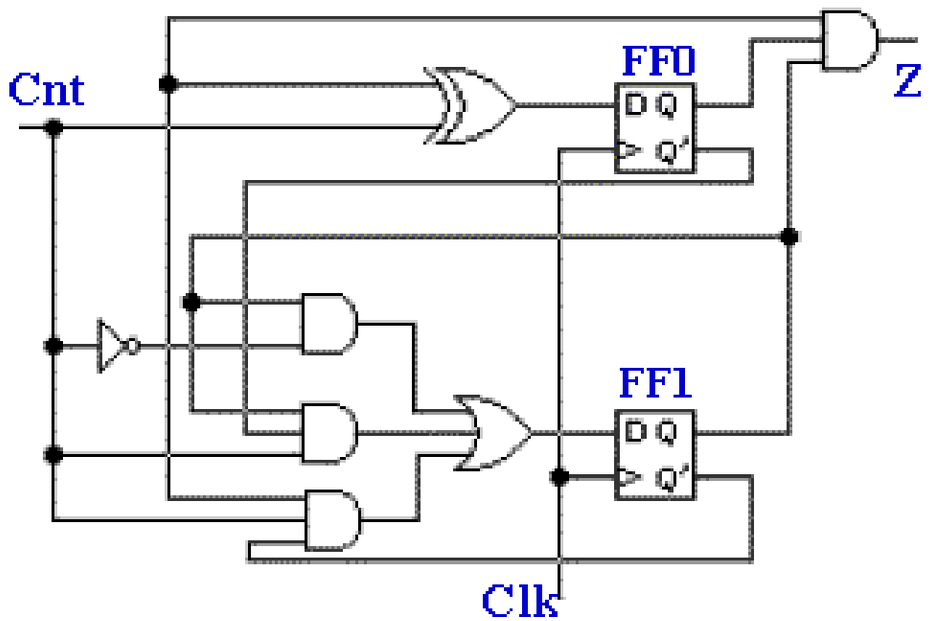
Karnaugh diyagramı yardımıyla aynı denklemleri bulabiliriz:

$$D_1 = Q_1\, Q_0'\, X' + Q_1'\, Q_0\, X'$$
$$D_0 = X + Q_1\, Q_0'$$
$$Z = Q_1\, Q_0\, X$$



Analysis of Sequential Circuits.

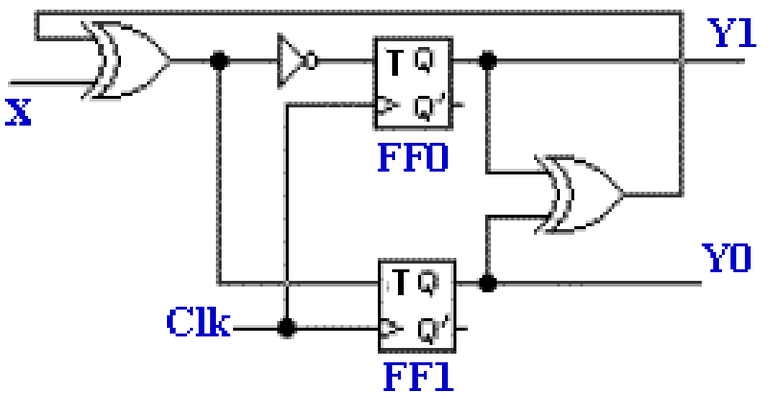
1. Derive a) excitation equations, b) next state equations,



c) a state/output table, and

d) a state diagram

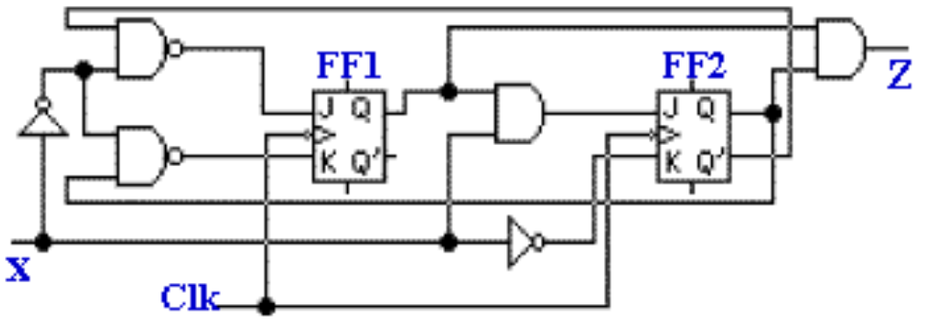
2. Derive a) excitation equations, b) next state equations,



c) a state/output table,

and d) a state diagram

3. Derive a) excitation equations, b) next state equations,



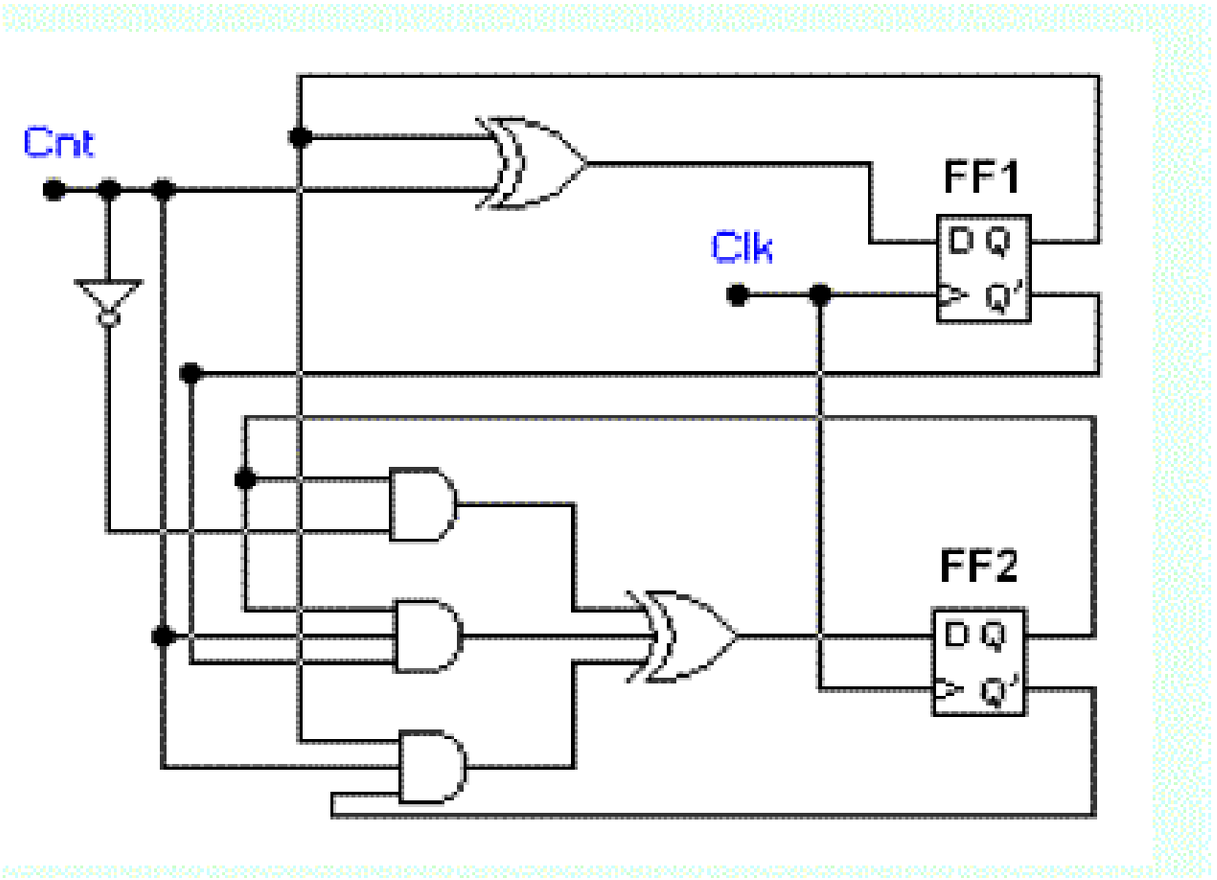
c) a state/output table,

and d) a state diagram

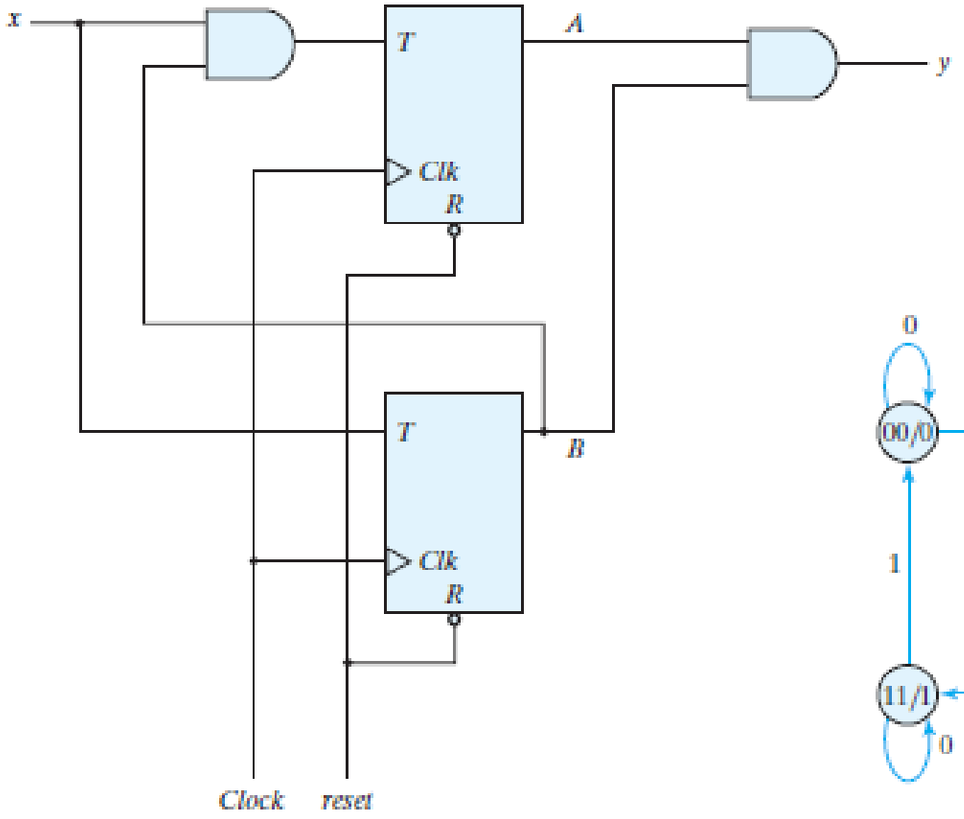
5. A sequential circuit uses two D flip-flops as memory elements.

The behaviour of the circuit is described by the following equations:

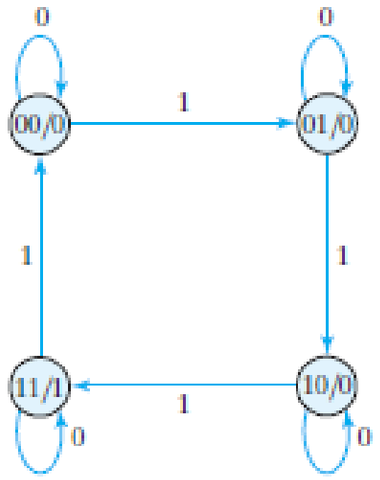
$$D1 = Q1 + x'*Q2$$
$$D2 = x*Q1' + x'*Q2$$
$$Z = x'*Q1*Q2 + x*Q1'*Q2'$$


Cnt
Clk
FF1
D Q
Q'
FF2
D Q
Q'

Example



(a) Circuit diagram

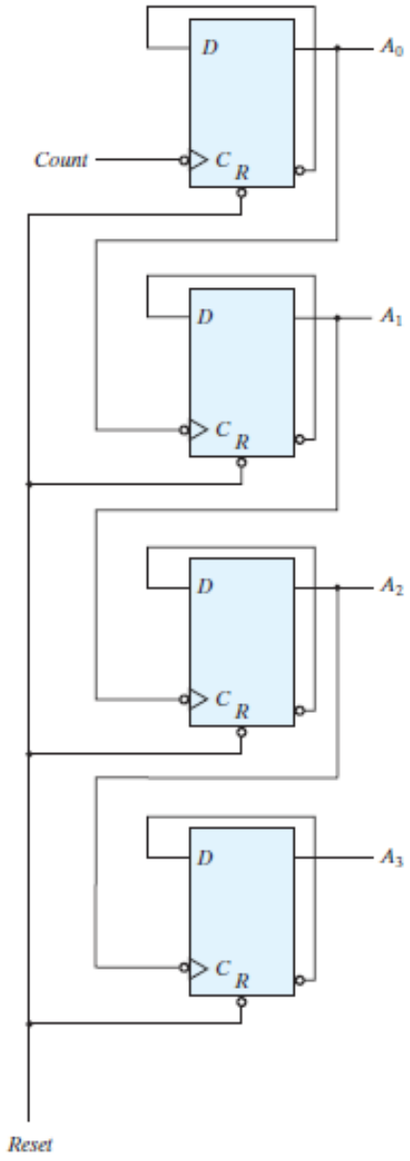


(b) State diagram

State Table for Sequential Circuit with T Flip-Flops

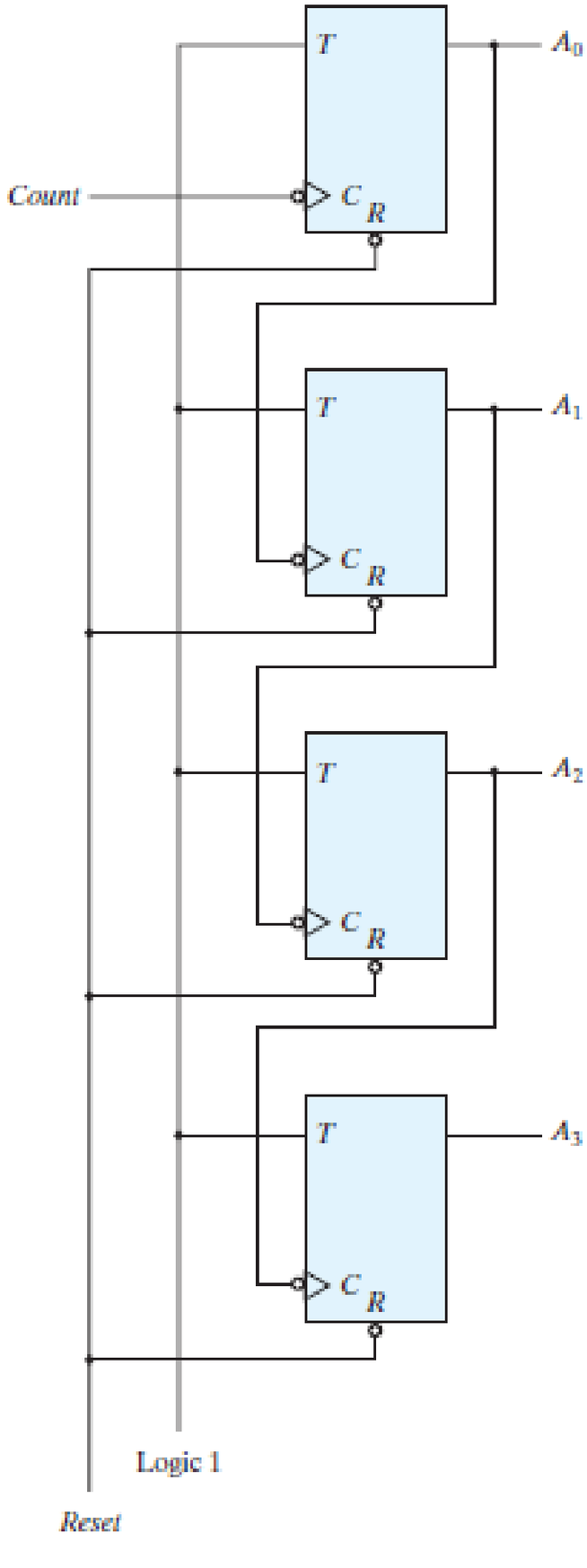
Present State		Input	Next State		Output
A	*B*	*x*	*A*	*B*	*y*
0	0	0	0	0	0
0	0	1	0	1	0
0	1	0	0	1	0
0	1	1	1	0	0
1	0	0	1	0	0
1	0	1	1	1	0
1	1	0	1	1	1
1	1	1	0	0	1

Example



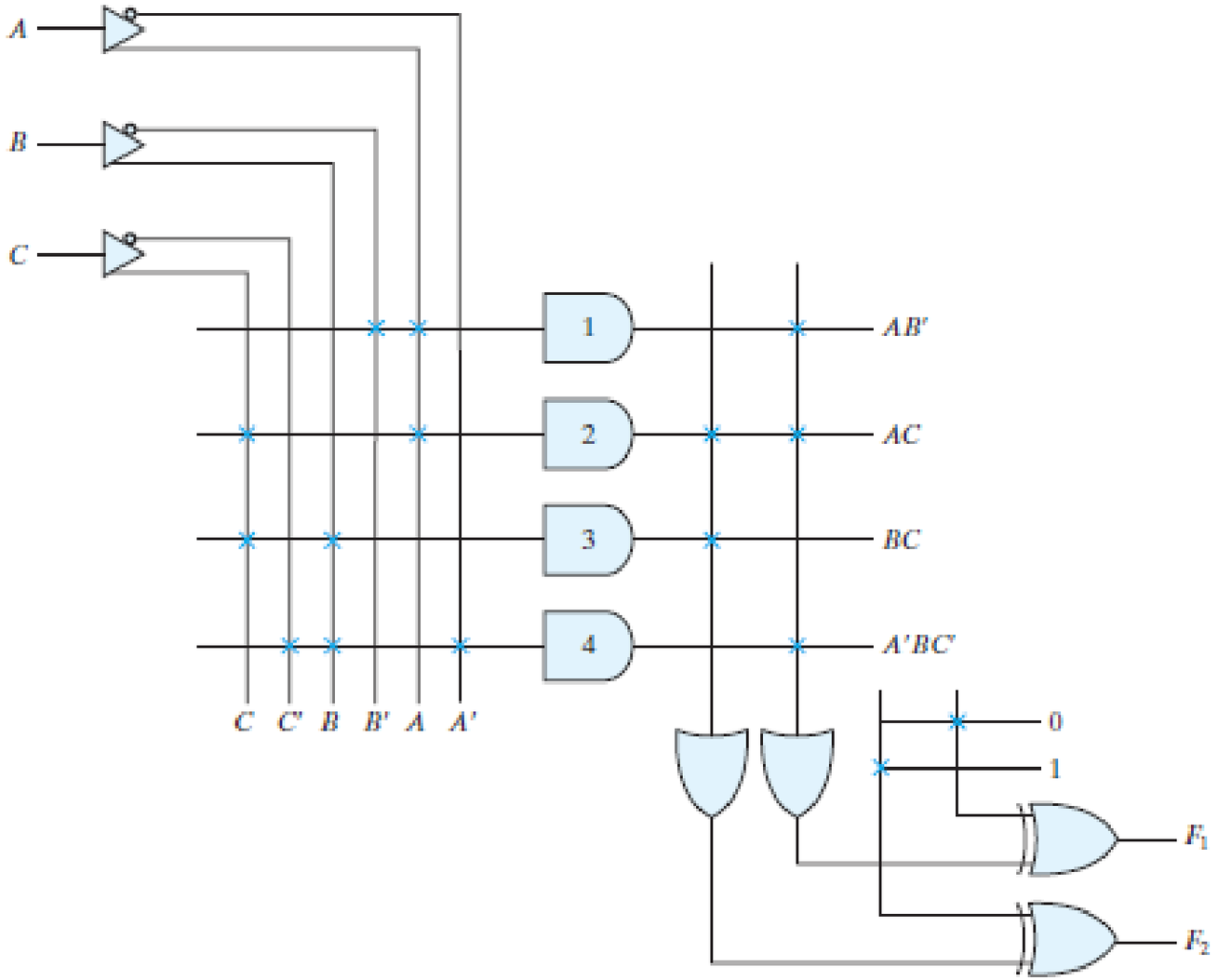
(b) With *D* flip-flops

Example

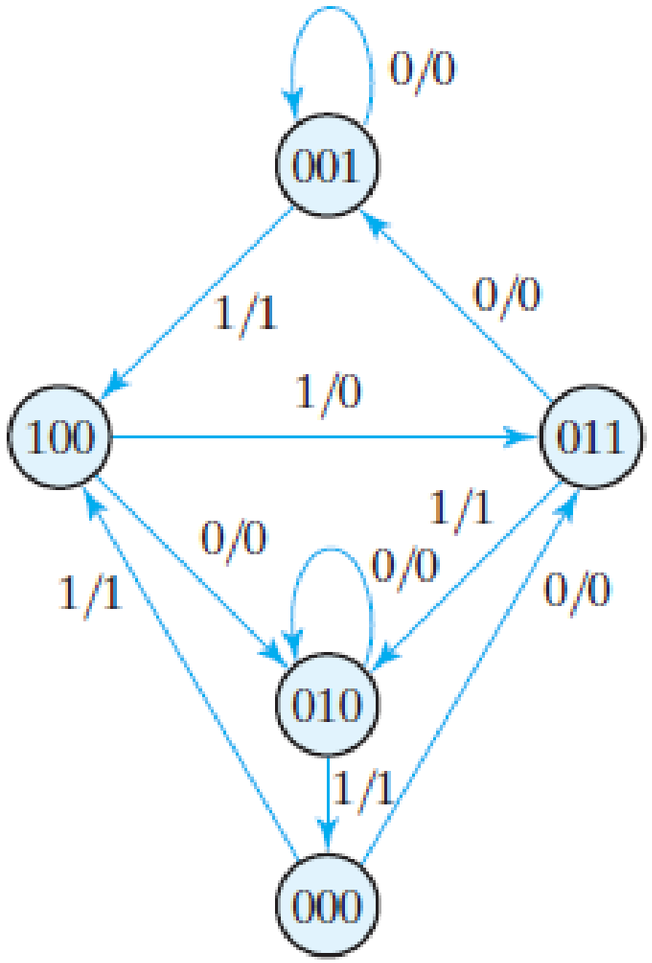


(a) With T flip-flops

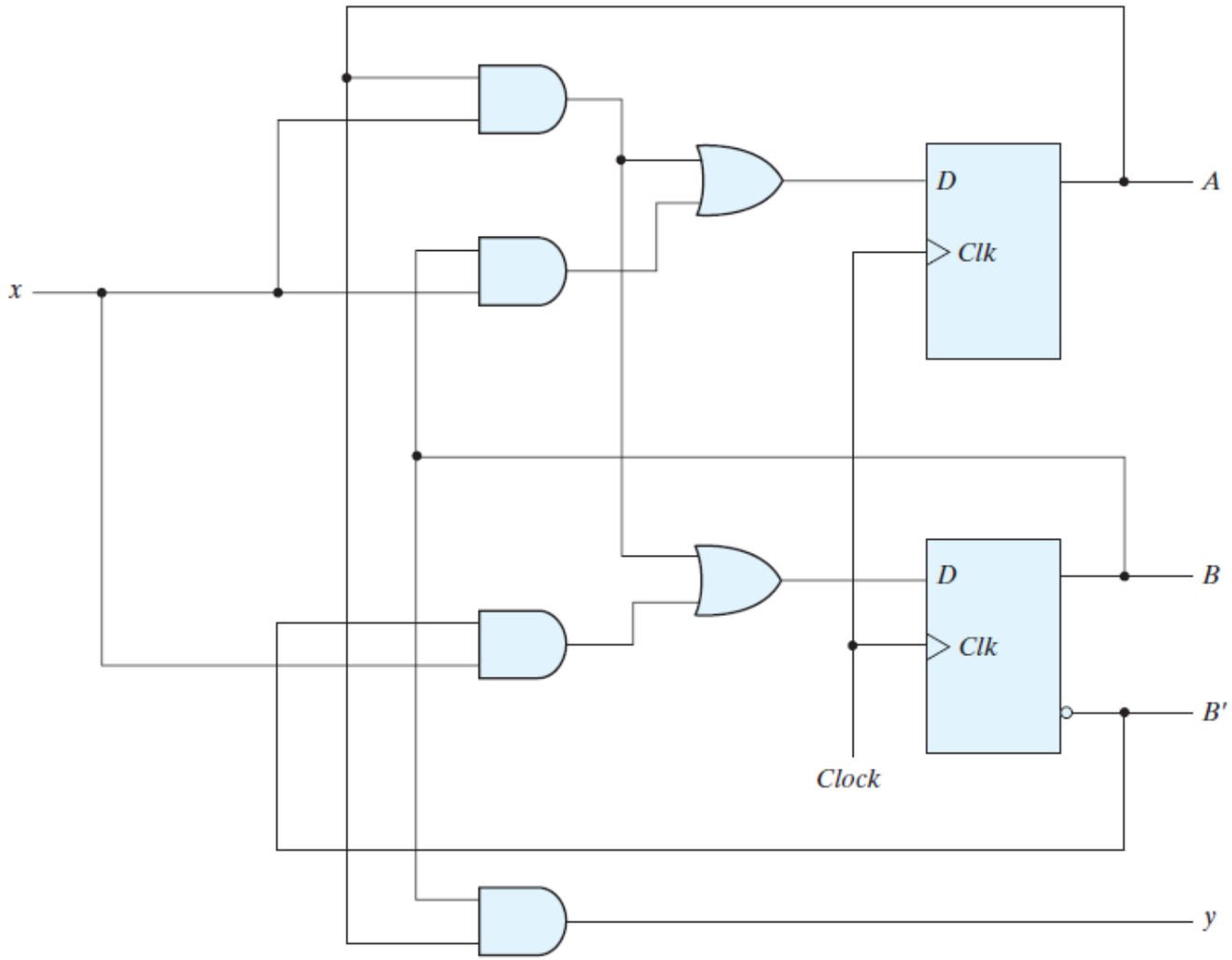
Example



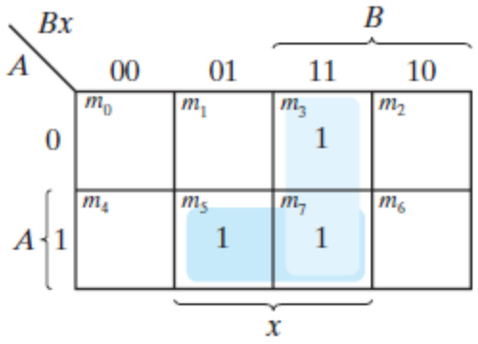
Example



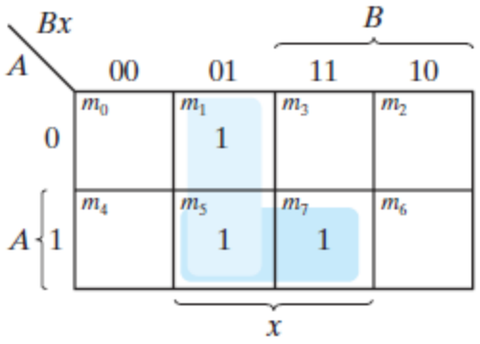
Example



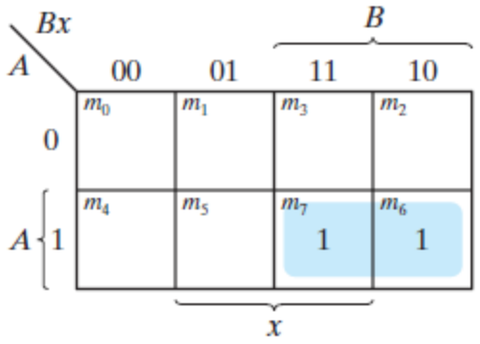
Example



$D_A = Ax + Bx$



$D_B = Ax + B'x$



$y = AB$

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

Kaynakça

- https://ocw.mit.edu/courses/electrical-engineering-and-computer-science/6-111-introductory-digital-systems-laboratory-spring-2006/lecture-notes/
- http://web.ee.nchu.edu.tw/~cpfan/FY92b-digital/Chapter-4.ppt
- http://www.cs.nccu.edu.tw/~whliao/ds2003/ds4.ppt
- http://www.just.edu.jo/~tawalbeh/cpe252/slides/CH1_2.ppt
- Lessons In Electric Circuits, Volume IV { Digital By Tony R. Kuphaldt Fourth Edition, last update July 30, 2004.
- Digital Electronics Part I – Combinational and Sequential Logic Dr. I. J. Wassell.
- Digital Design With an Introduction to the Verilog HDL, M. Morris Mano Emeritus Professor of Computer Engineering California State University, Los Angeles; Michael D. Ciletti Emeritus Professor of Electrical and Computer Engineering University of Colorado at Colorado Springs.
- Digital Logic Design Basics, Combinational Circuits, Sequential Circuits, Pu-Jen Cheng.