Digital design

Ch 5. Synchronous sequential logic

5-1 Sequential circuits

- Outputs are function of inputs and present states
- Present states are supplied by memory elements

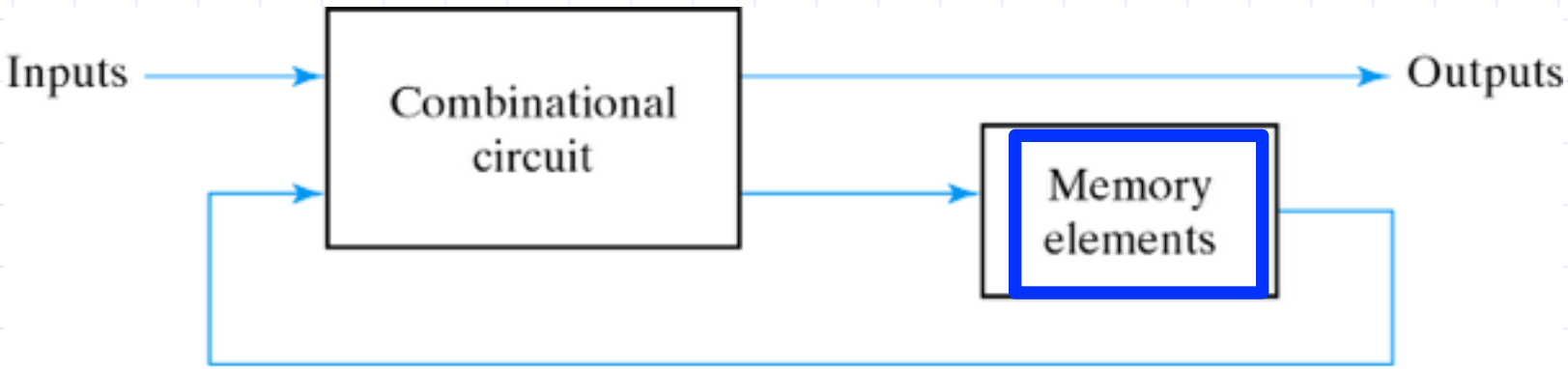
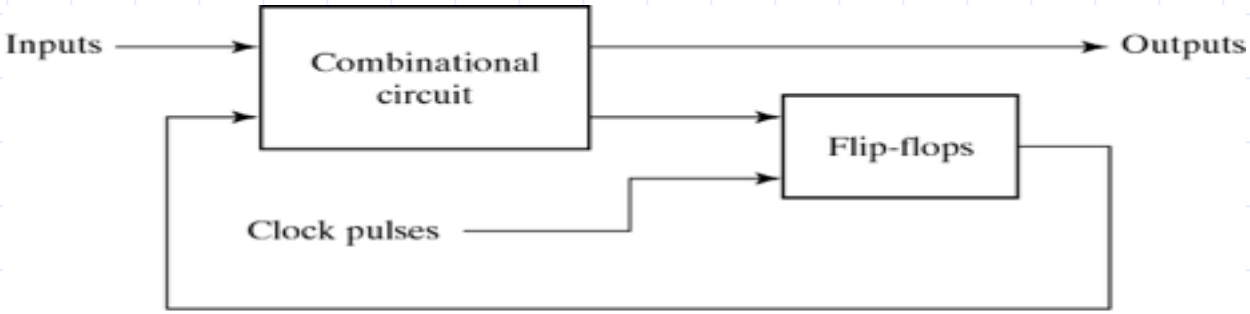


Fig. 5-1 Block Diagram of Sequential Circuit

Sequential circuits

- Two types of sequential circuit
- Synchronous : behavior depends on the signals affecting storage elements at discrete time
- Asynchronous : behavior depends on inputs at any instance of time



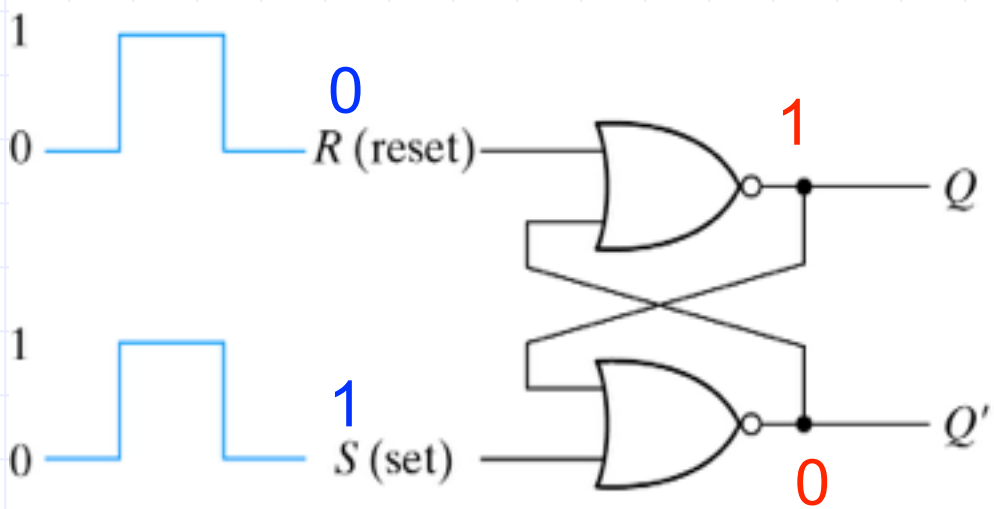
(a) Block diagram

(b) Timing diagram of clock pulses

Fig. 5-2 Synchronous Clocked Sequential Circuit

5-2 Latches

- SR latch : consist of two cross-coupled NOR gates
- S=1,R=0 then Q=1(set)



(a) Logic diagram

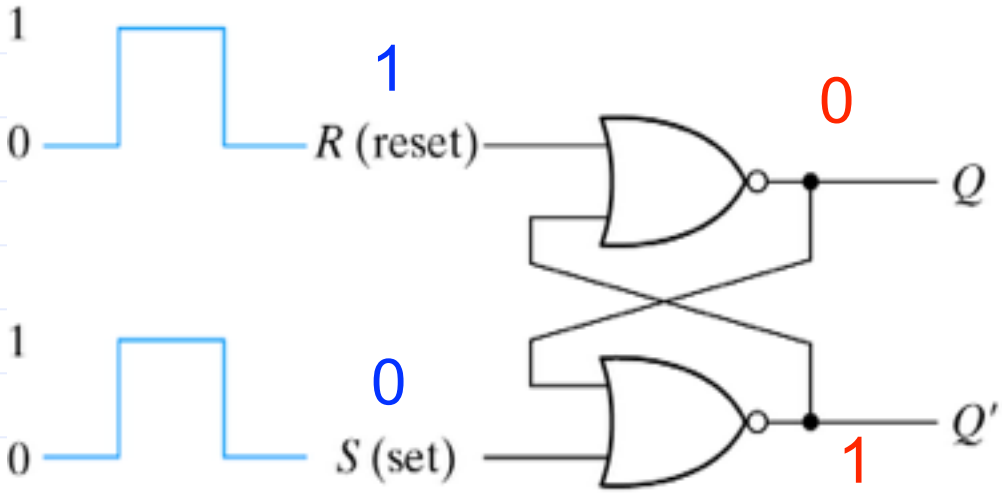
S	R	Q	Q'	
1	0	1	0	
0	0	1	0	(after $S = 1, R = 0$)
0	1	0	1	
0	0	0	1	(after $S = 0, R = 1$)
1	1	0	0	

(b) Function table

Fig. 5-3 *SR* Latch with NOR Gates

5-2 Latches

- SR latch : consist of two cross-coupled NOR gates
- S=0,R=1 then Q=0(reset)



(a) Logic diagram

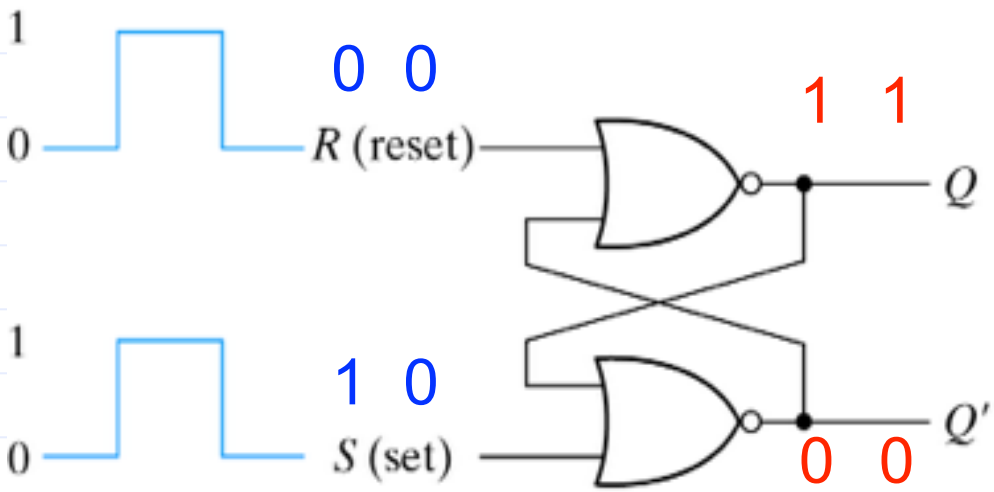
S	R	Q	Q'	
1	0	1	0	
0	0	1	0	(after $S = 1, R = 0$)
0	1	0	1	
0	0	0	1	(after $S = 0, R = 1$)
1	1	0	0	

(b) Function table

Fig. 5-3 *SR* Latch with NOR Gates

5-2 Latches

- SR latch : consist of two cross-coupled NOR gates
- S=0,R=0 then no change(keep condition)



(a) Logic diagram

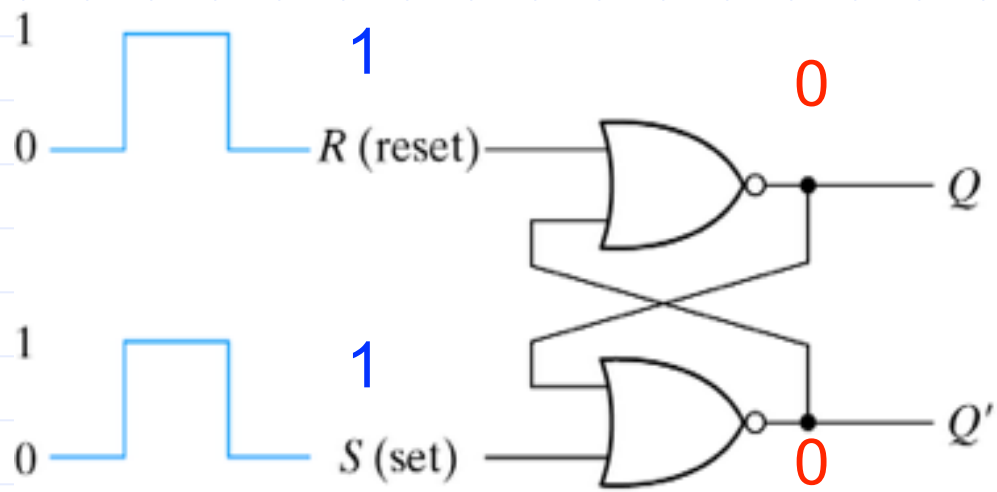
S	R	Q	Q'	
1	0	1	0	
0	0	1	0	(after $S = 1, R = 0$)
0	1	0	1	
0	0	0	1	(after $S = 0, R = 1$)
1	1	0	0	

(b) Function table

Fig. 5-3 *SR* Latch with NOR Gates

5-2 Latches

- SR latch : consist of two cross-coupled NOR gates
- S=1,R=1 Q=Q′=0 (undefined)



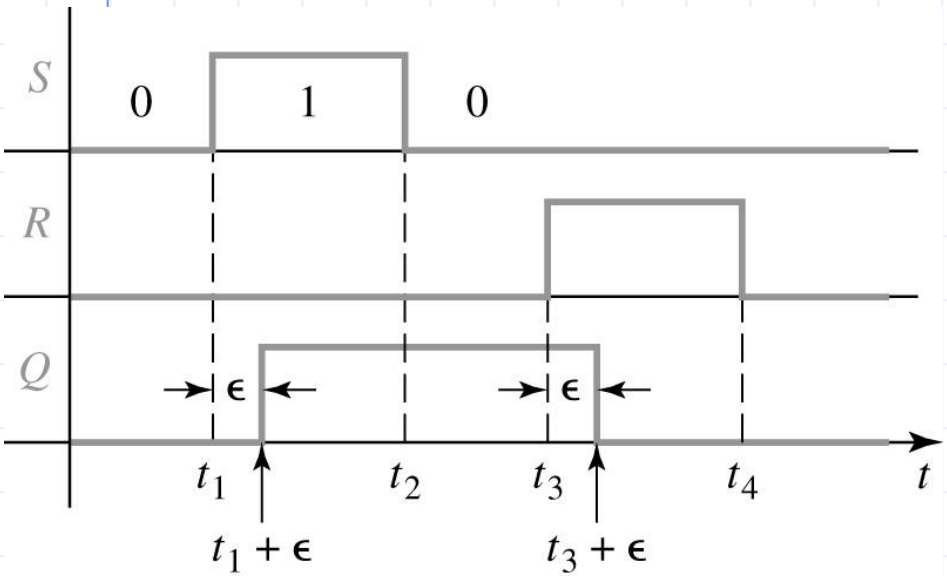
(a) Logic diagram

S	R	Q	Q'	
1	0	1	0	
0	0	1	0	(after $S = 1, R = 0$)
0	1	0	1	
0	0	0	1	(after $S = 0, R = 1$)
1	1	0	0	

(b) Function table

Fig. 5-3 *SR* Latch with NOR Gates

5-2 Latches



SR latch timing diagram

$S(t)$	$R(t)$	$Q(t)$	$Q(t+\varepsilon)$
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	–

SR latch operation

5-2 Latches

$S(t)$	$R(t)$	$Q(t)$	$Q(t+\varepsilon)$
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	–

SR latch operation

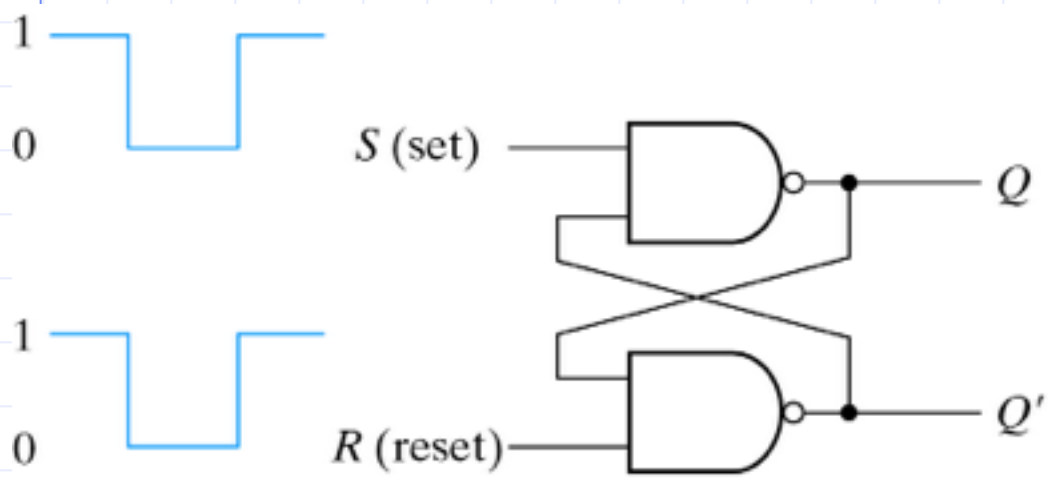
$R(t)\,Q(t)$ \ $S(t)$	00	01
00	0	1
01	1	1
11	0	X
10	0	X

$$Q(t+\epsilon) = S(t) + R'(t)\,Q(t)$$

$$Q^{+} = S + R'Q \qquad (SR = 0)$$

S′R′ latch with NAND gates

- Require the complement value of NOR latch



(a) Logic diagram

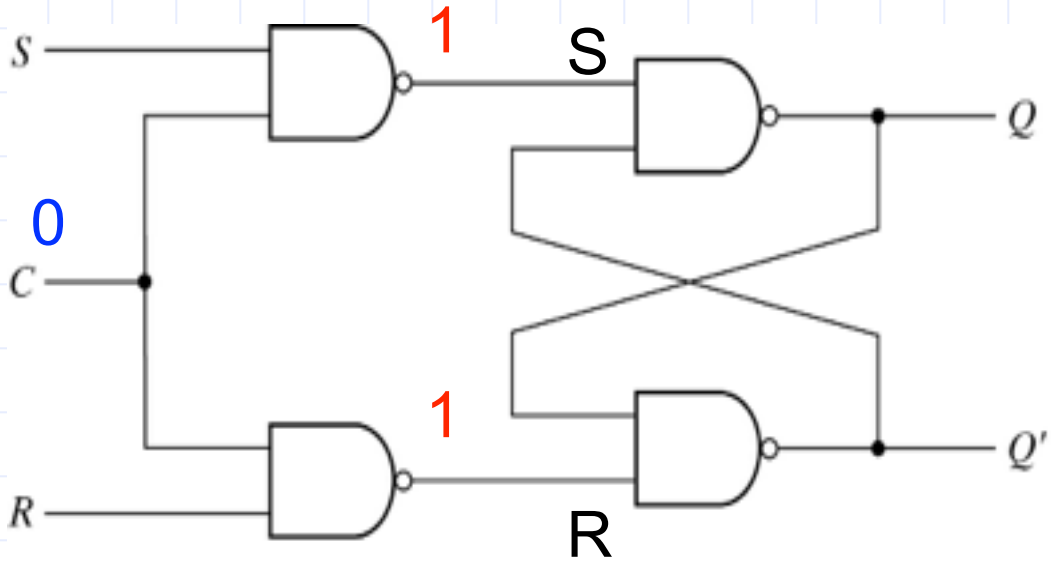
S	R	Q	Q'	
1	0	0	1	
1	1	0	1	(after $S = 1, R = 0$)
0	1	1	0	
1	1	1	0	(after $S = 0, R = 1$)
0	0	1	1	

(b) Function table

Fig. 5-4 *SR* Latch with NAND Gates

SR latch with control input

- Add two NAND gate and control signal
- C=0(no action), C=1(act as SR latch)



(a) Logic diagram

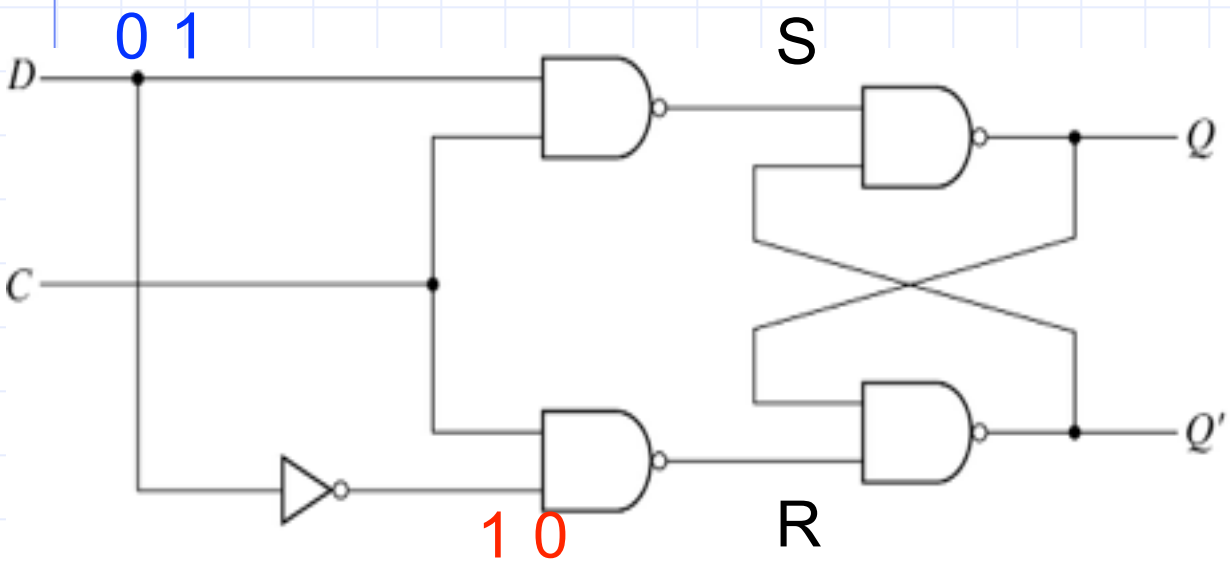
C	S	R	Next state of Q
0	X	X	No change
1	0	0	No change
1	0	1	Q = 0; Reset state
1	1	0	Q = 1; set state
1	1	1	Indeterminate

(b) Function table

Fig. 5-5 SR Latch with Control Input

D latch

- Eliminate indeterminate state in SR latch
- C=1, output value is equal to D



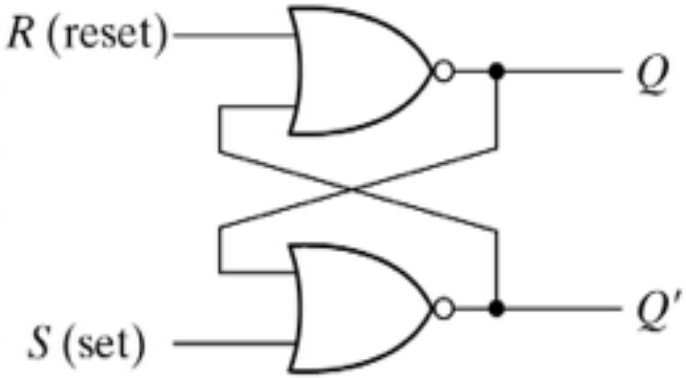
(a) Logic diagram

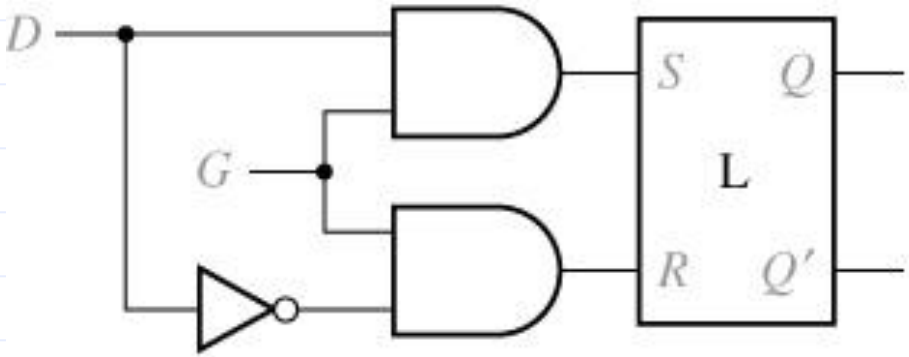
C	D	Next state of Q
0	X	No change
1	0	Q = 0; Reset state
1	1	Q = 1; Set state

(b) Function table

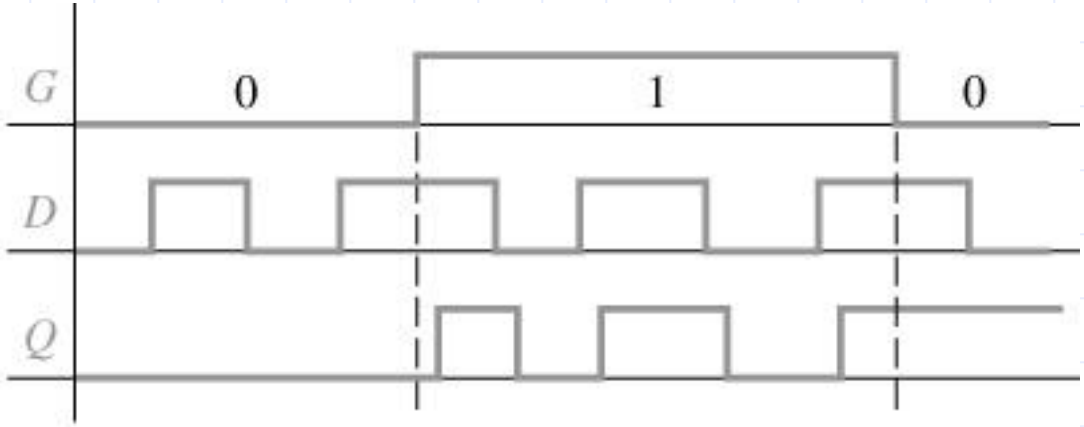
Fig. 5-6 D Latch

D latch



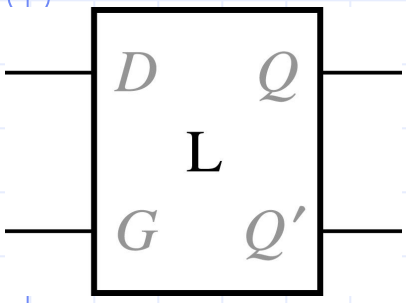


(a)



(b)

D latch



G	D	Q	Q^+
0	0	0	0
0	0	1	1
0	1	0	0
0	1	1	1
1	0	0	0
1	0	1	0
1	1	0	1
1	1	1	1

Q \ GD	00	01	11	10
0	0	0	1	0
1	1	1	1	0

$$Q^+ = G'Q + GD$$

5-3 Flip-flops

- Latch : case (a), output changes as input changes
- Flip-flop : output only changes at clock edge

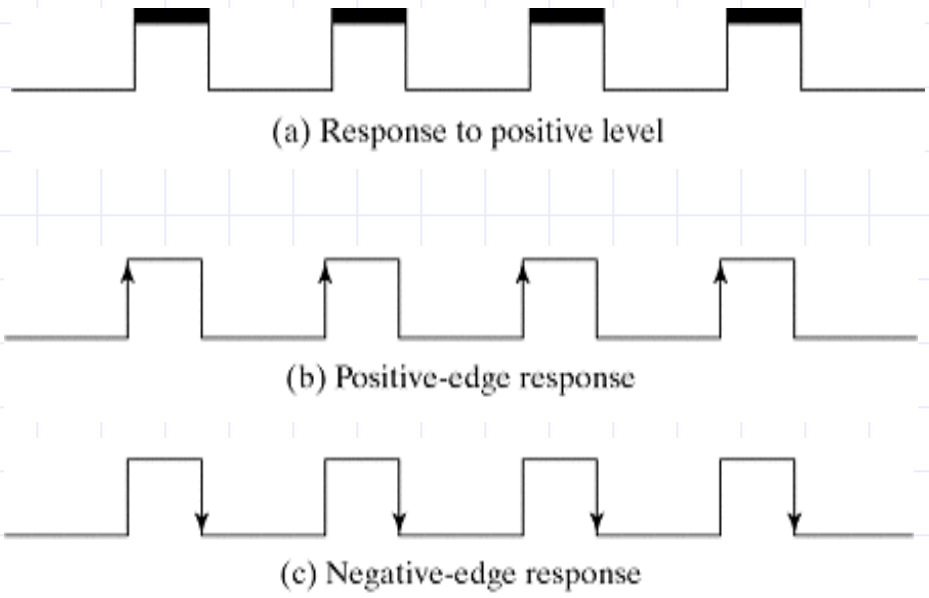


Fig. 5-8 Clock Response in Latch and Flip-Flop

Master-slave D flip-flop

- Negative edge triggered D flip-flop
- CLK=1 : master enable, slave disable

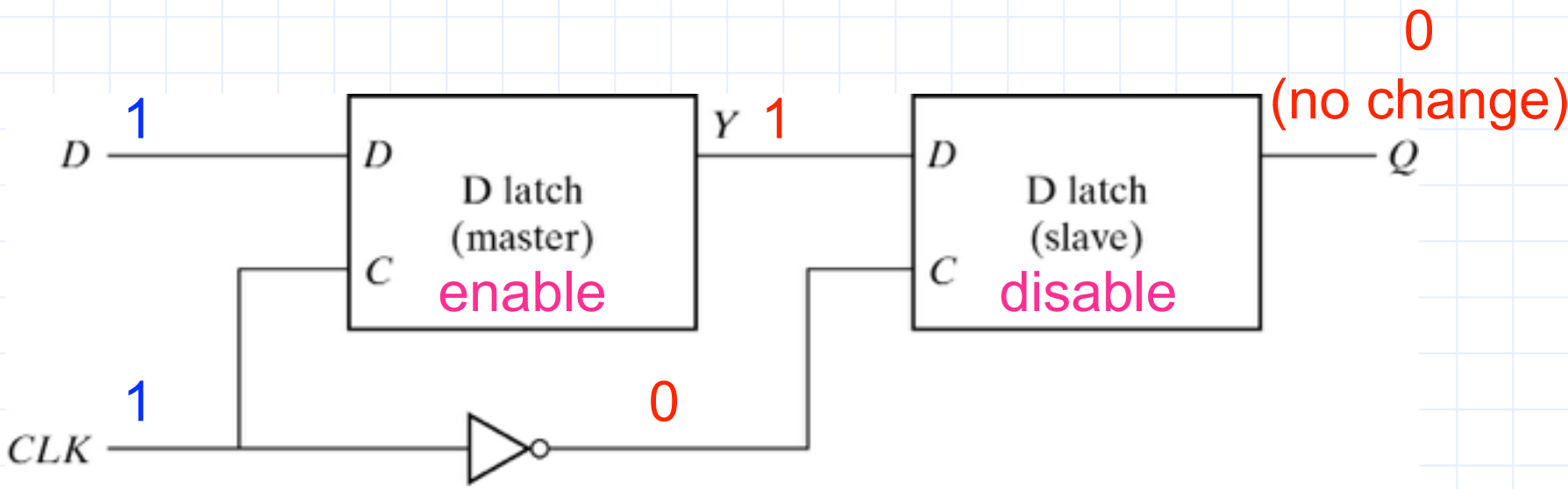


Fig. 5-9 Master-Slave *D* Flip-Flop

Master-slave D flip-flop

- Negative edge triggered D flip-flop
- CLK=0 : master disable, slave enable
  - Output has no relation with input

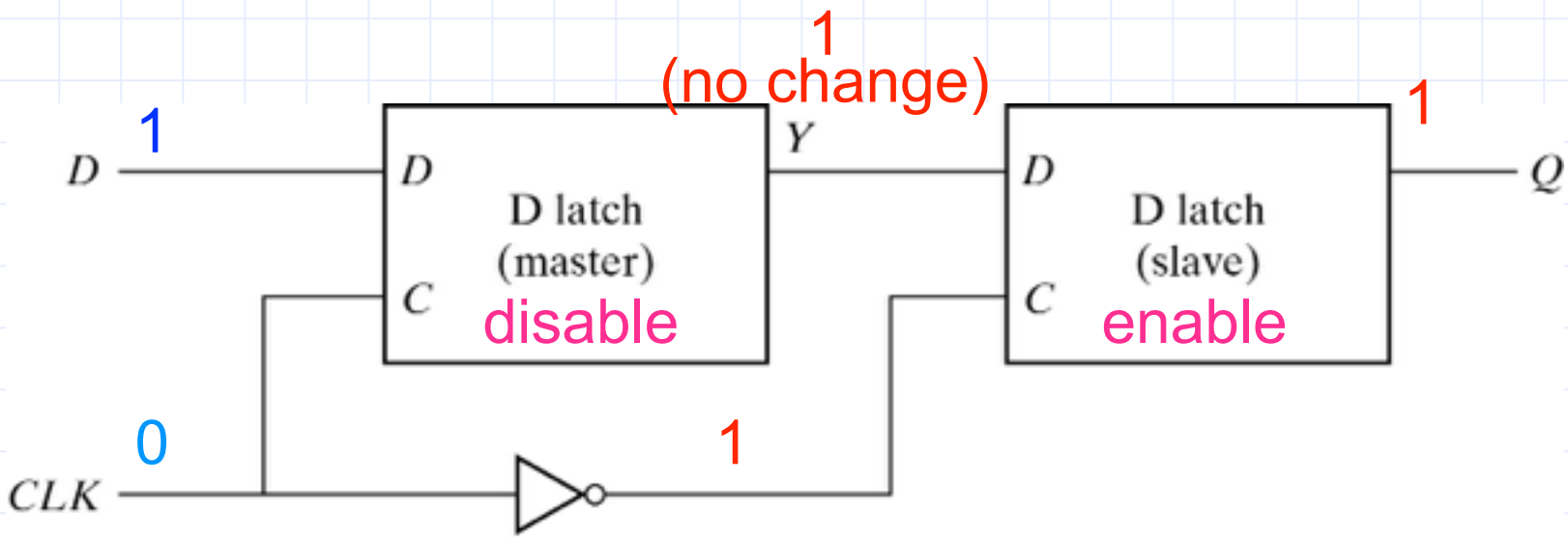
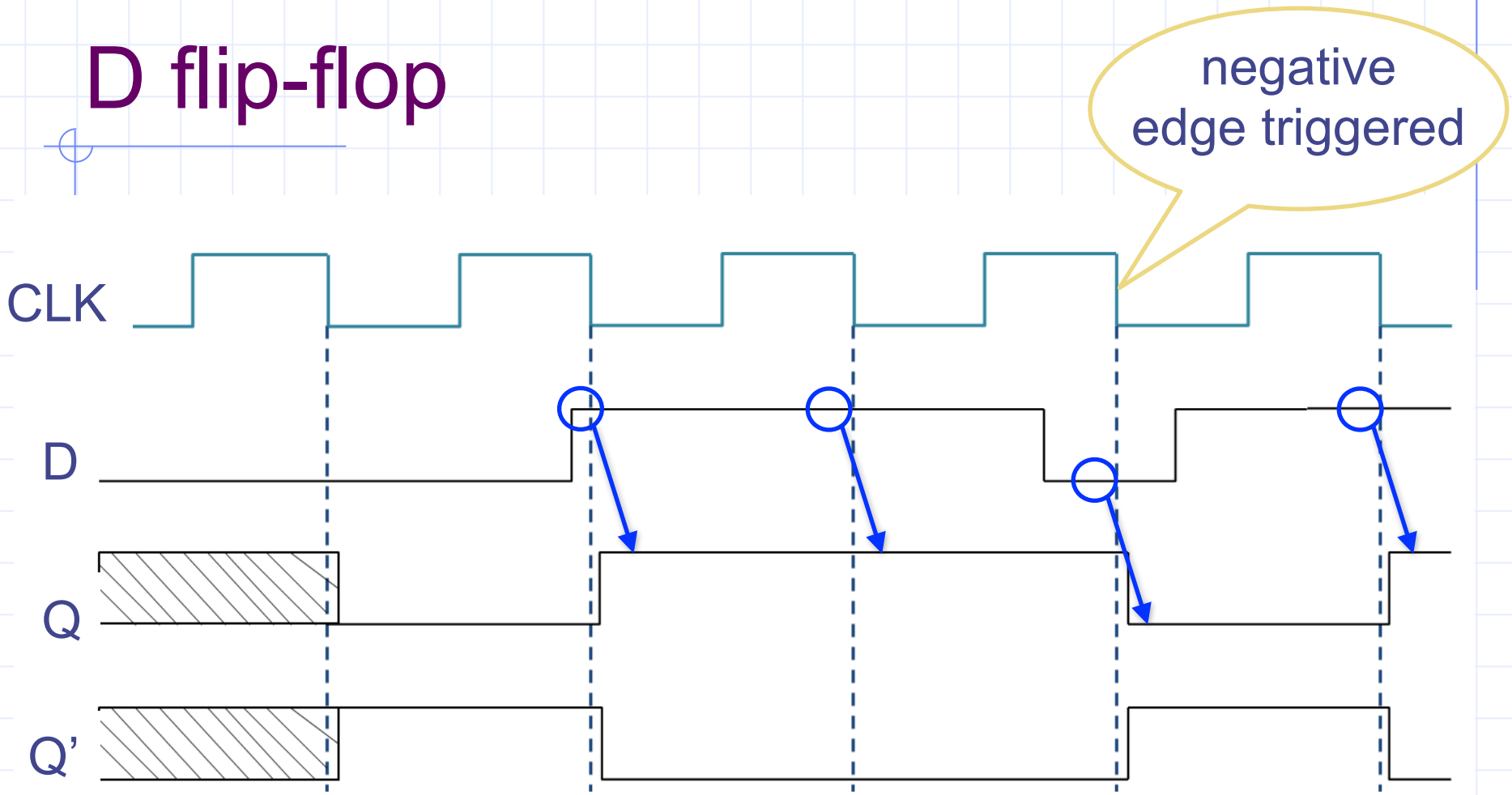
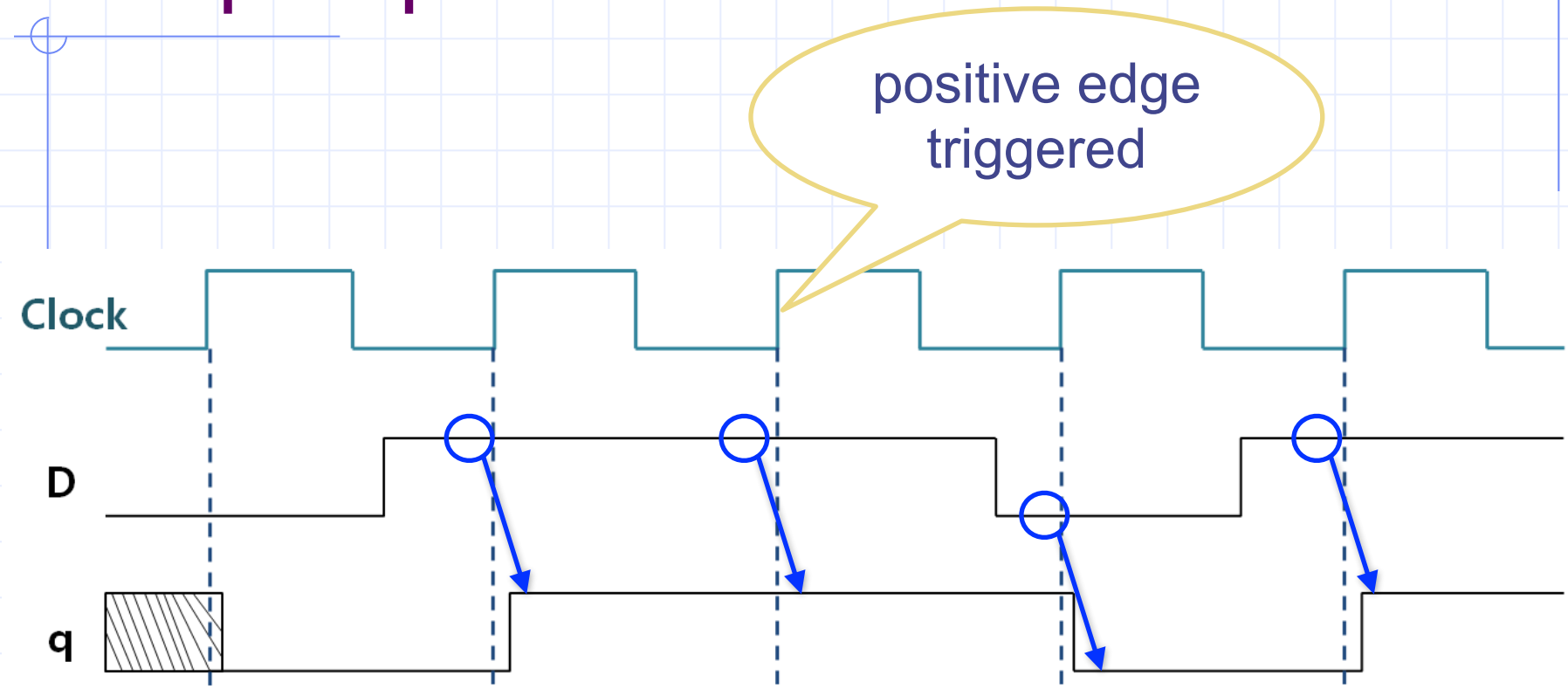


Fig. 5-9 Master-Slave *D* Flip-Flop

D flip-flop



D flip-flop



D-type positive edge triggered flip flop

- Consist of 3 SR-latches
- Q changes only when C becomes 0 to 1

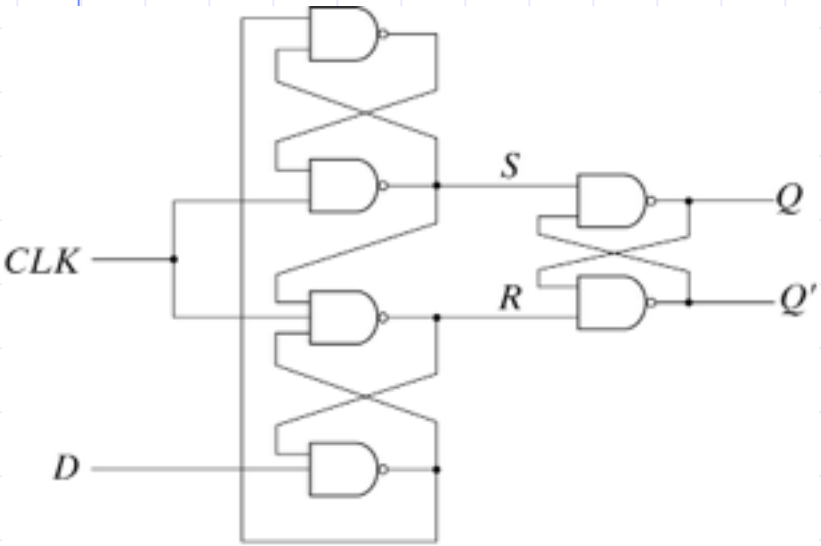


Fig. 5-10 *D*-Type Positive-Edge-Triggered Flip-Flop

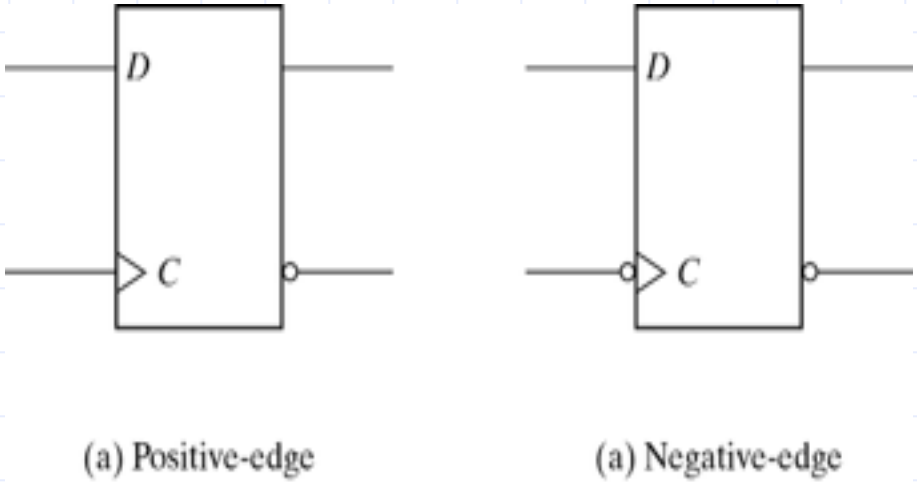
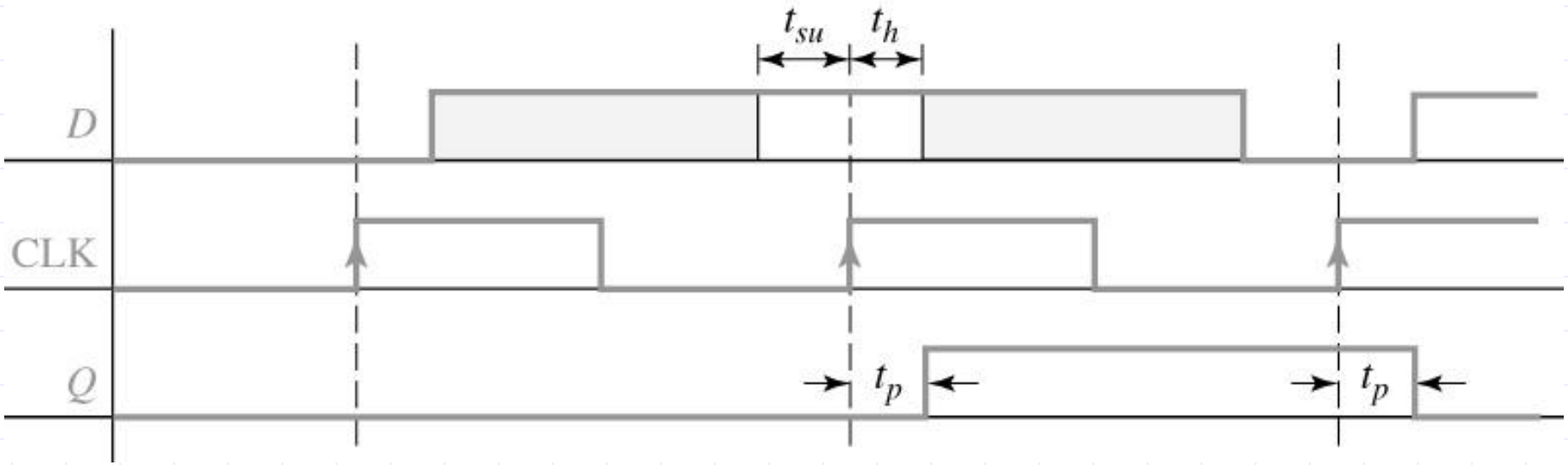


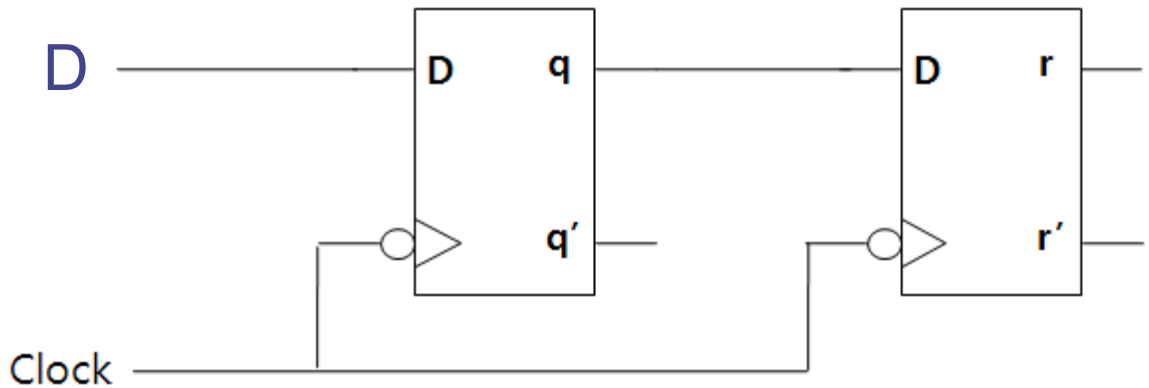
Fig. 5-11 Graphic Symbol for Edge-Triggered *D* Flip-Flop

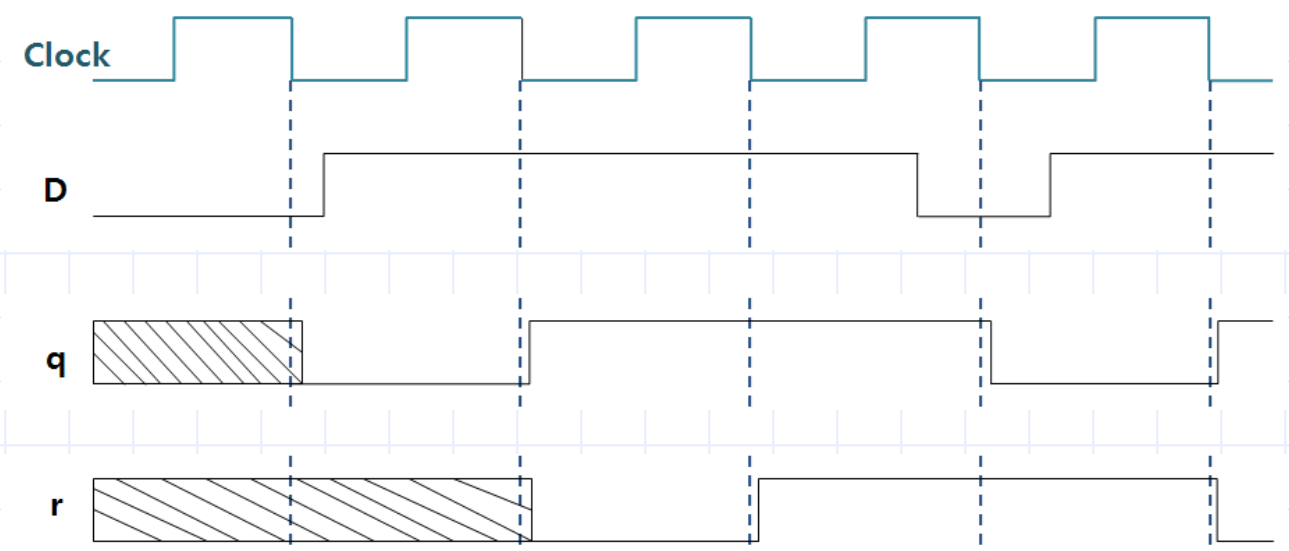
D-type edge triggered flip flop



t_{su}: set-up time
t_h: hold time
t_p: propagation delay

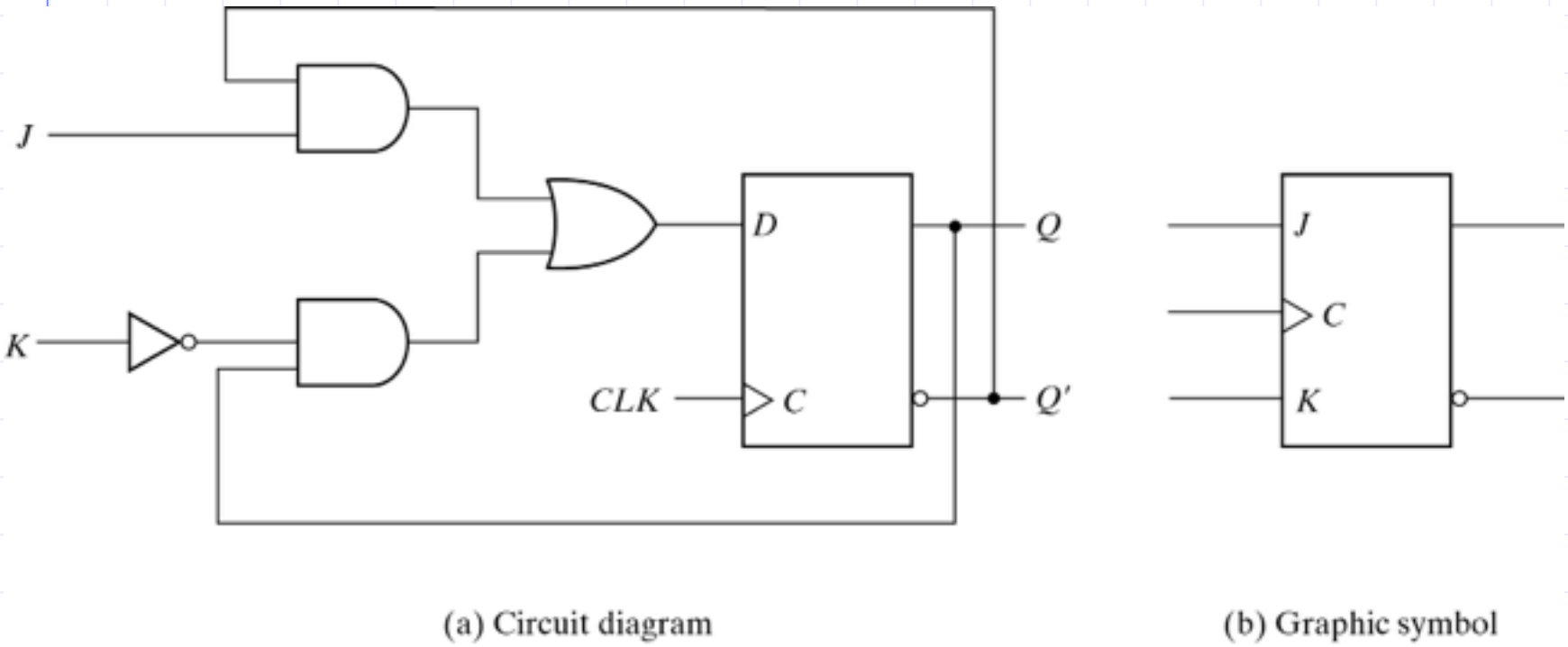
Two D-type flip-flops





JK flip-flop

- Performs three operations
  - Set(J), Reset(K), Complement(J=K=1)
  - D=JQ′+K′Q



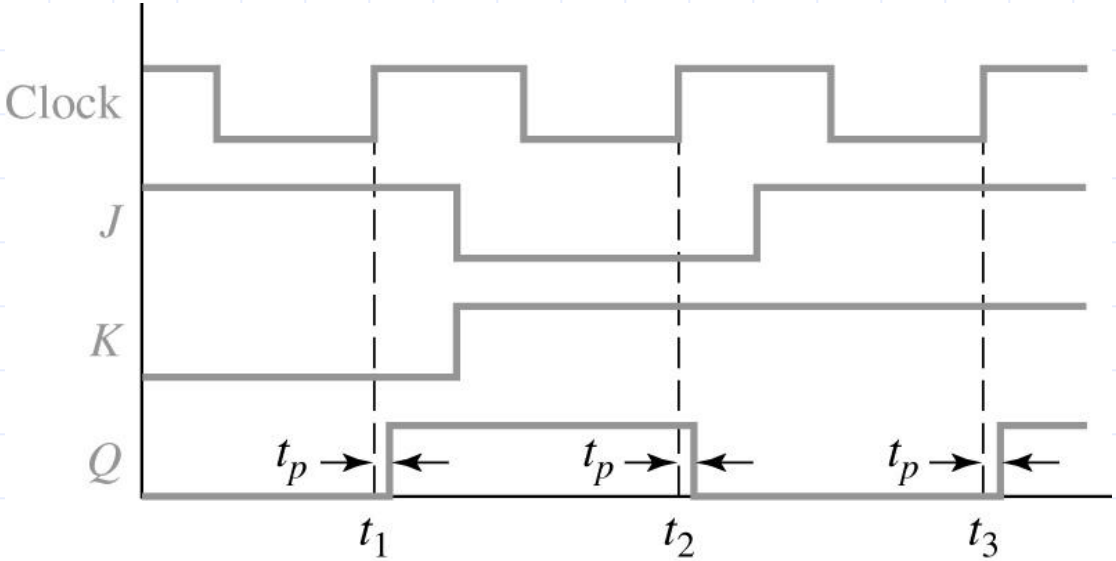
(a) Circuit diagram (b) Graphic symbol

Fig. 5-12 *JK* Flip-Flop

JK flip-flop

- Performs three operations
  - Set(J), Reset(K), Complement(J=K=1)
  - D=JQ′+K′Q

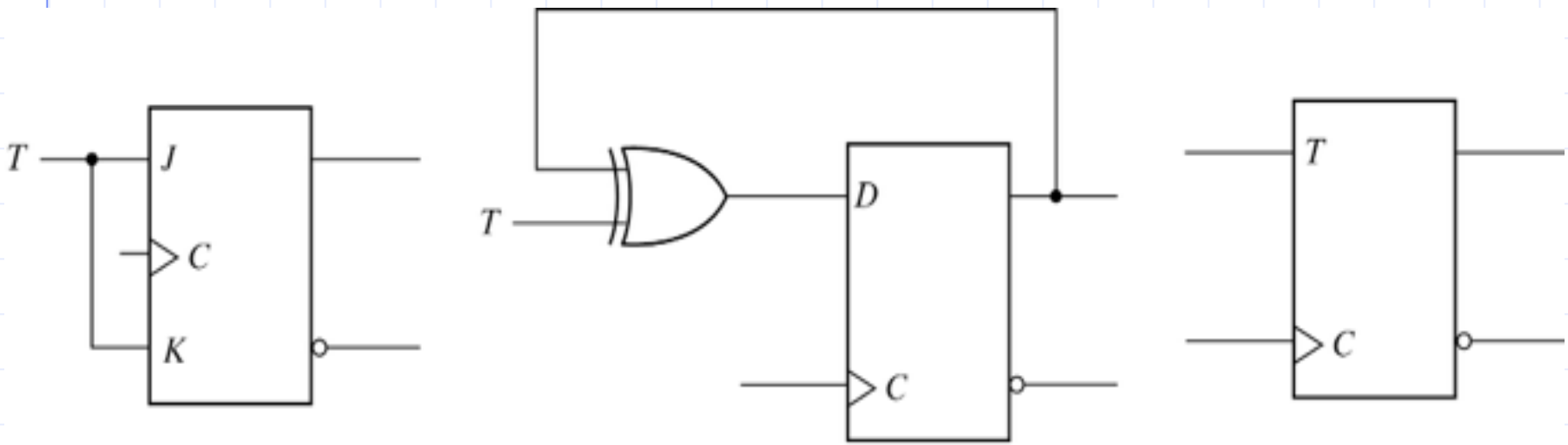
J	K	Q	Q^+
0	0	0	0
0	0	1	1
0	1	0	0
0	1	1	0
1	0	0	1
1	0	1	1
1	1	0	1
1	1	1	0



(c) J-K flip-flop timing

T flip-flop

- Complementing flip-flop
- $D = TQ' + T'Q = T \oplus Q$

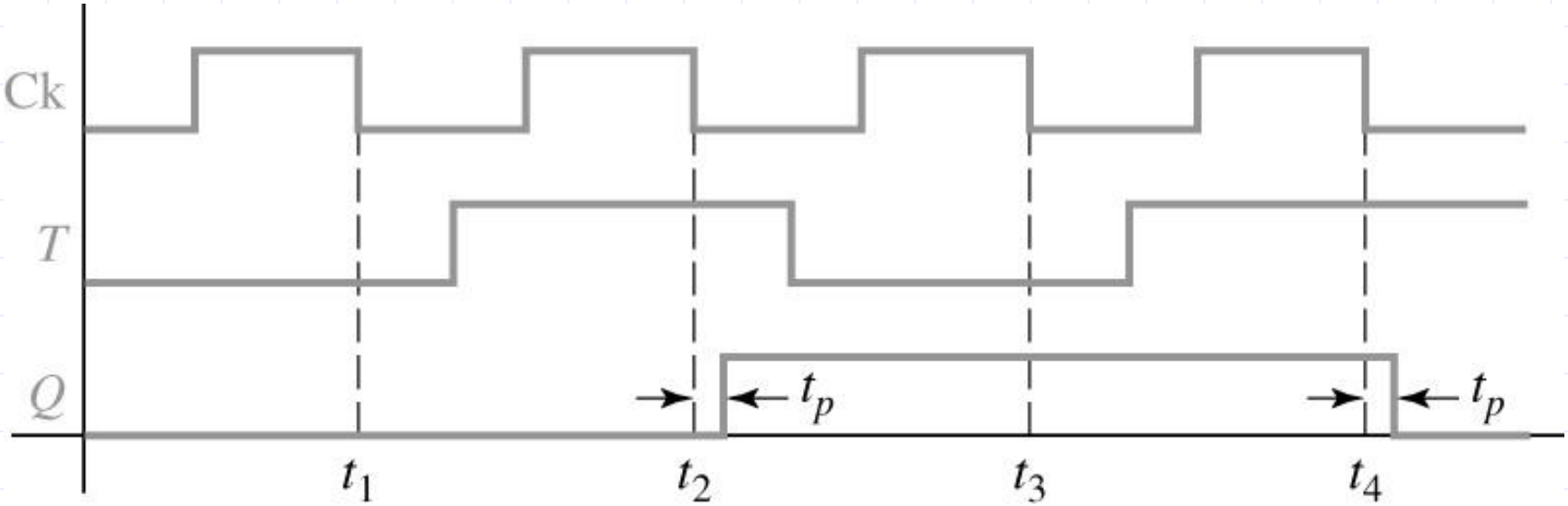


(a) From *JK* flip-flop (b) From *D* flip-flop (c) Graphic symbol

Fig. 5-13 T Flip-Flop

T flip-flop

T	Q	Q^+
0	0	0
0	1	1
1	0	1
1	1	0



Characteristic tables

- Flip-flop characteristic tables

Table 5-1
Flip-Flop Characteristic Tables

***JK* Flip-Flop**

J	K	$Q(t+1)$	
0	0	$Q(t)$	No change
0	1	0	Reset
1	0	1	Set
1	1	$Q'(t)$	Complement

Q(t): present state
Q(t+1): next state
D(t), T(t), J(t), K(t): present input

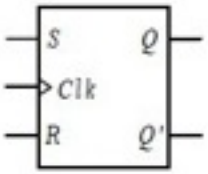
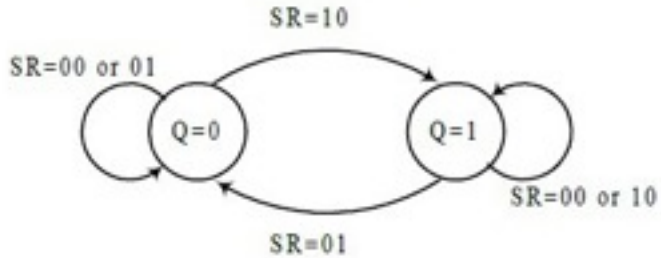
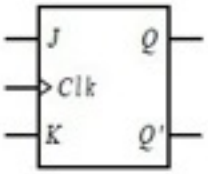
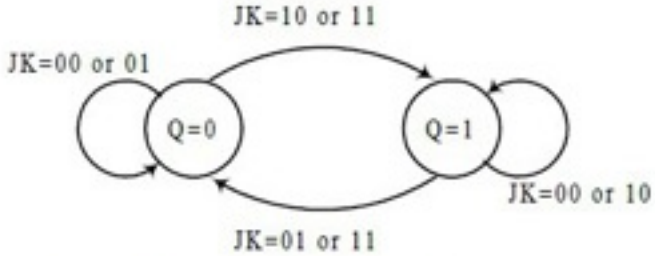
***D* Flip-Flop**

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

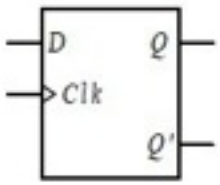
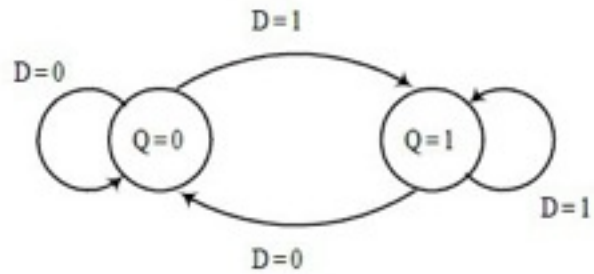
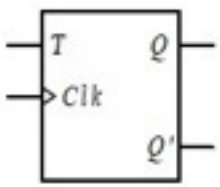
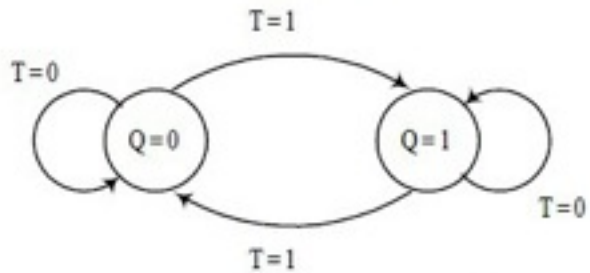
***T* Flip-Flop**

T	$Q(t+1)$	
0	$Q(t)$	No change
1	$Q'(t)$	Complement

Summary

Name / Symbol	Characteristic (Truth) Table	State Diagram / Characteristic Equations	Excitation Table
SR  	$S\ R\ Q\ Q_{next}$ 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 ×	 $Q_{next} = S + R'Q$ $SR = 0$	$Q\ Q_{next}\ S\ R$ 0 0 0 × 0 1 1 0 1 0 0 1 1 1 × 0
JK  	$J\ K\ Q\ Q_{next}$ 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	 $Q_{next} = J'K'Q + JK' + JKQ'$ $= J'K'Q + JK'Q + JK'Q' + JKQ'$ $= K'Q(J'+J) + JQ'(K'+K)$ $= K'Q + JQ'$	$Q\ Q_{next}\ J\ K$ 0 0 0 × 0 1 1 × 1 0 × 1 1 1 × 0

Summary

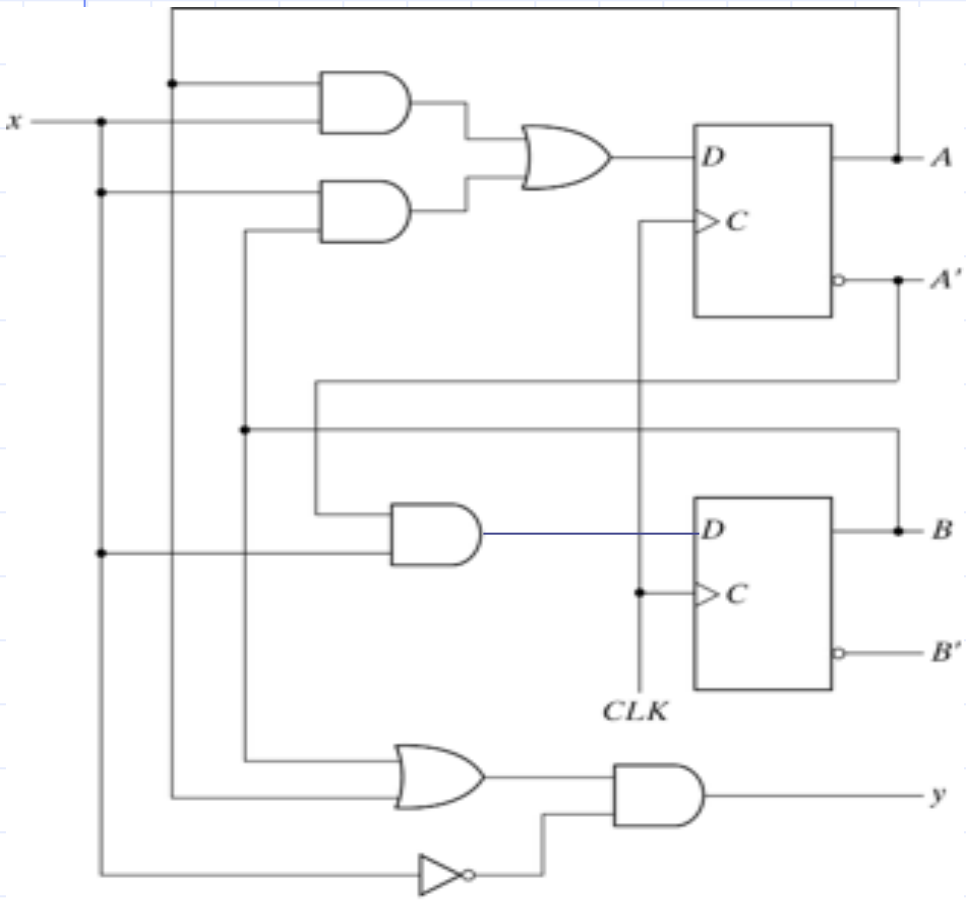
Name / Symbol	Characteristic (Truth) Table	State Diagram / Characteristic Equations	Excitation Table
D  	D Q Q_{next} 0 × 0 1 × 1	 $Q_{next} = D$	Q Q_{next} D 0 0 0 0 1 1 1 0 0 1 1 1
T  	T Q Q_{next} 0 0 0 0 1 1 1 0 1 1 1 0	 $Q_{next} = TQ' + T'Q = T \oplus Q$	Q Q_{next} T 0 0 0 0 1 1 1 0 1 1 1 0

5-4 Analysis of clocked sequential circuits

- Behavior of clocked sequential circuit is determined from input, output and present state
- Output, next state are a function of input and present state

State equations

- Specifies the next state and output as a function of the present state and inputs



- $A(t+1) = D = Ax + Bx$
- $B(t+1) = D = A'x$
- $Y = (A+B)x'$

State table

- Time sequence table of inputs, outputs and flip-flop states
- two types of state table exist

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

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	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

Table 5-3
Second Form of the State Table

Present State	Next State		Output	
	$x = 0$	$x = 1$	$x = 0$	$x = 1$
AB	*AB*	*AB*	*y*	*y*
00	00	01	0	0
01	00	11	1	0
10	00	10	1	0
11	00	10	1	0

State diagram

- A kind of flow diagram
- Can be derived from state table
  - State-circle, transition-line, I/O

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

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	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

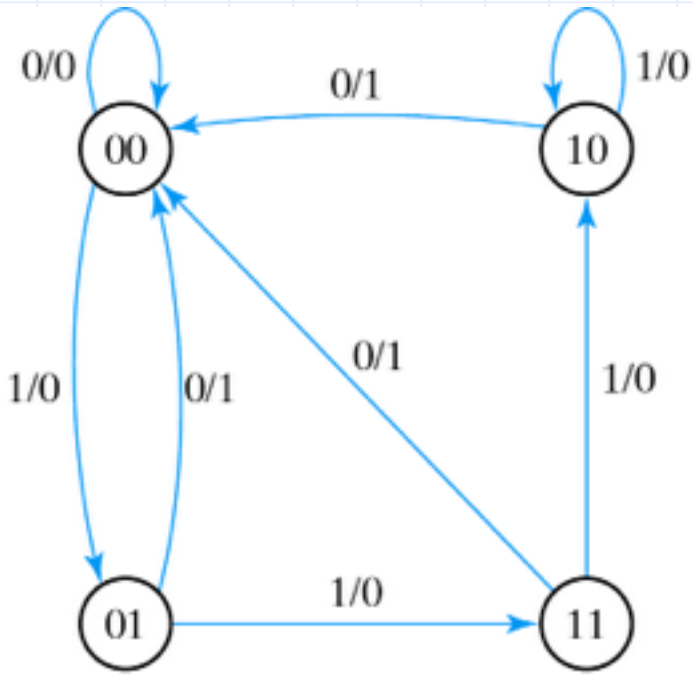
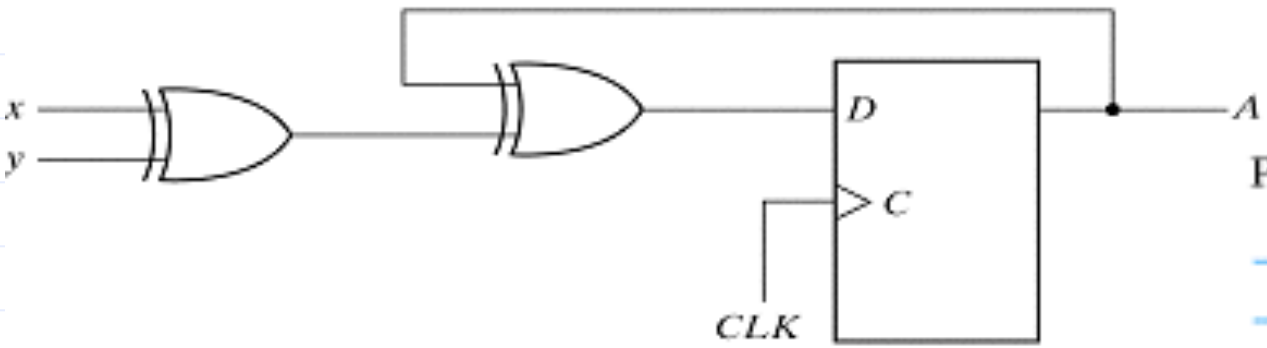


Fig. 5-16 State Diagram of the Circuit of Fig. 5-15

Analysis with D flip-flops

- Input equation : $D_A = A \oplus x \oplus y$
- State equation is equal to input equation



(a) Circuit diagram

Present state	Inputs		Next state
A	x	y	A
0	0	0	0
0	0	1	1
0	1	0	1
0	1	1	0
1	0	0	1
1	0	1	0
1	1	0	0
1	1	1	1

(b) State table

Analysis with D flip-flops

- Input equation : $D_A = A \oplus x \oplus y$
- State equation is equal to input equation

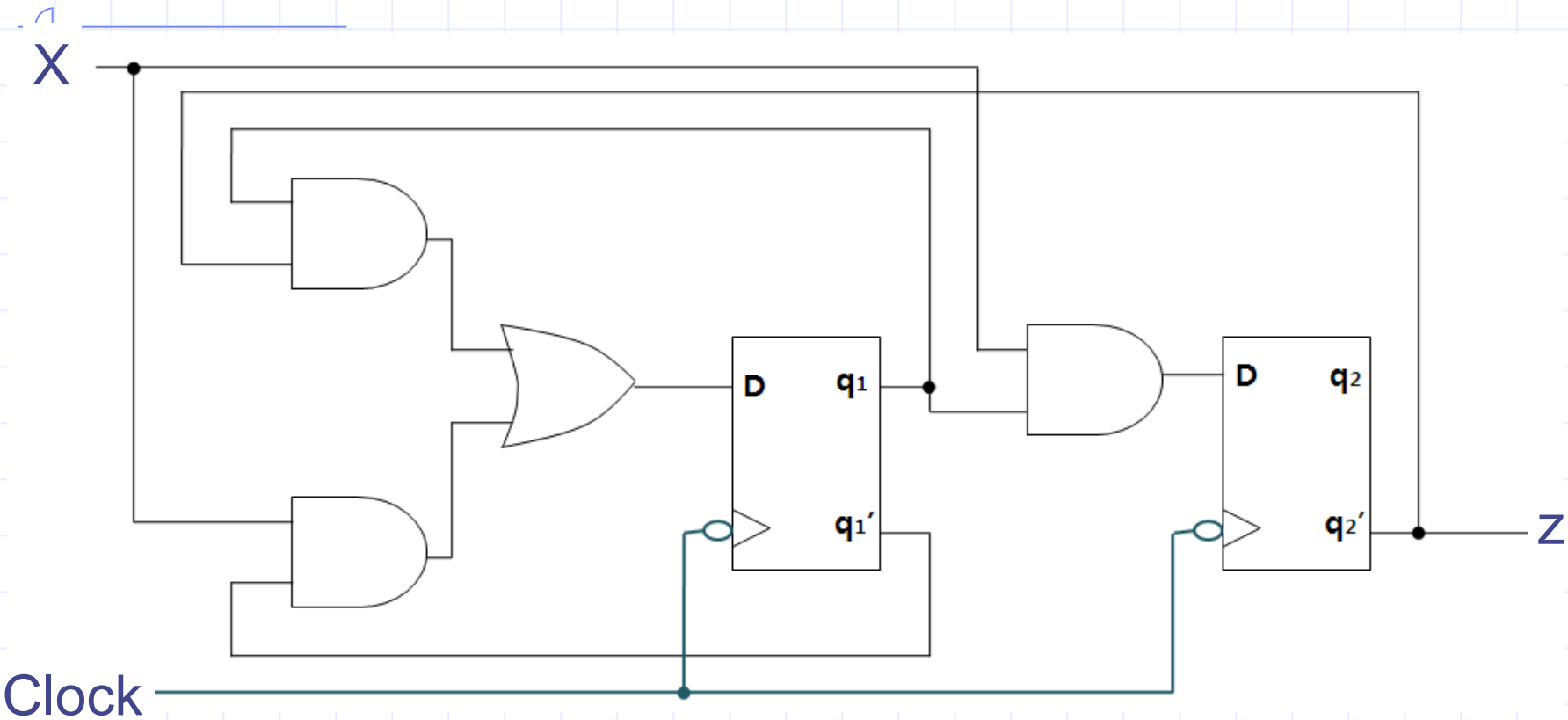
Present state	Inputs		Next state
A	x	y	A
0	0	0	0
0	0	1	1
0	1	0	1
0	1	1	0
1	0	0	1
1	0	1	0
1	1	0	0
1	1	1	1

(b) State table

(c) State diagram

Fig. 5-17 Sequential Circuit with D Flip-Flop

Analysis with D flip-flops



$D_1 = q_1q_2' + xq_1'$

$D_2 = xq_1$

$z = q_2'$

Analysis with D flip-flops

$D_1 = q_1q_2' + xq_1'$
$D_2 = xq_1$
$z = q_2'$

D Flip-Flop

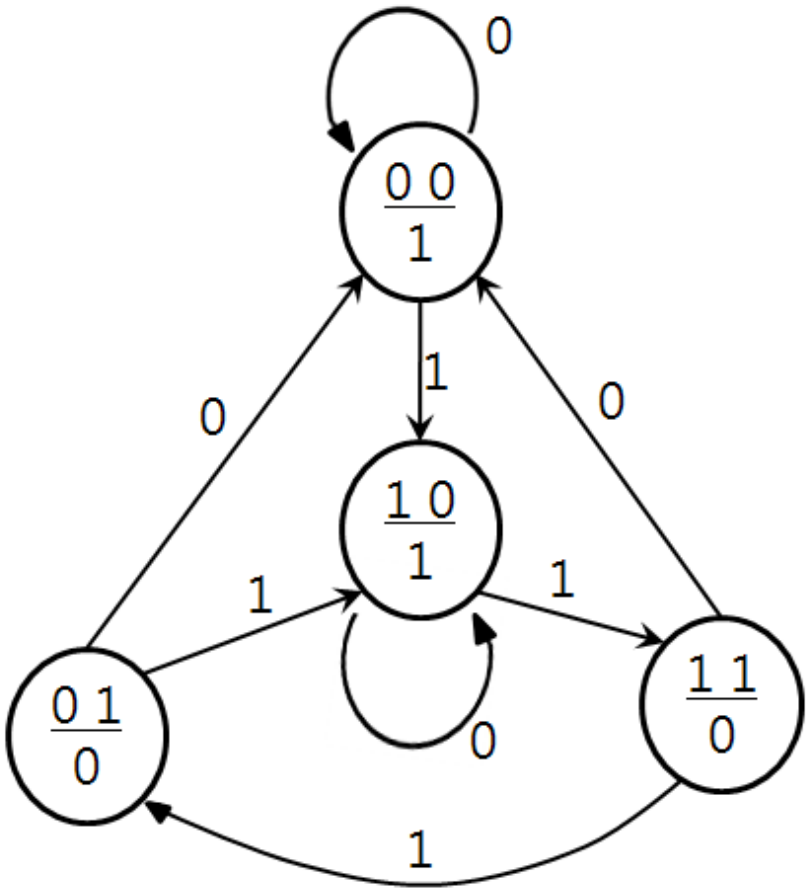
D	$Q(t+1)$	
0	0	Reset
1	1	Set

q	q	D		D		q_1				z
		x=0	x=1	x=0	x=1	x=0		x=1		
0	0	***0***	***0***	***1***	***0***	0	0	1	0	1
0	1	***0***	***0***	***1***	***0***	0	0	1	0	0
1	0	***1***	***0***	***1***	***1***	1	0	1	1	1
1	1	***0***	***0***	***0***	***1***	0	0	1	0	0

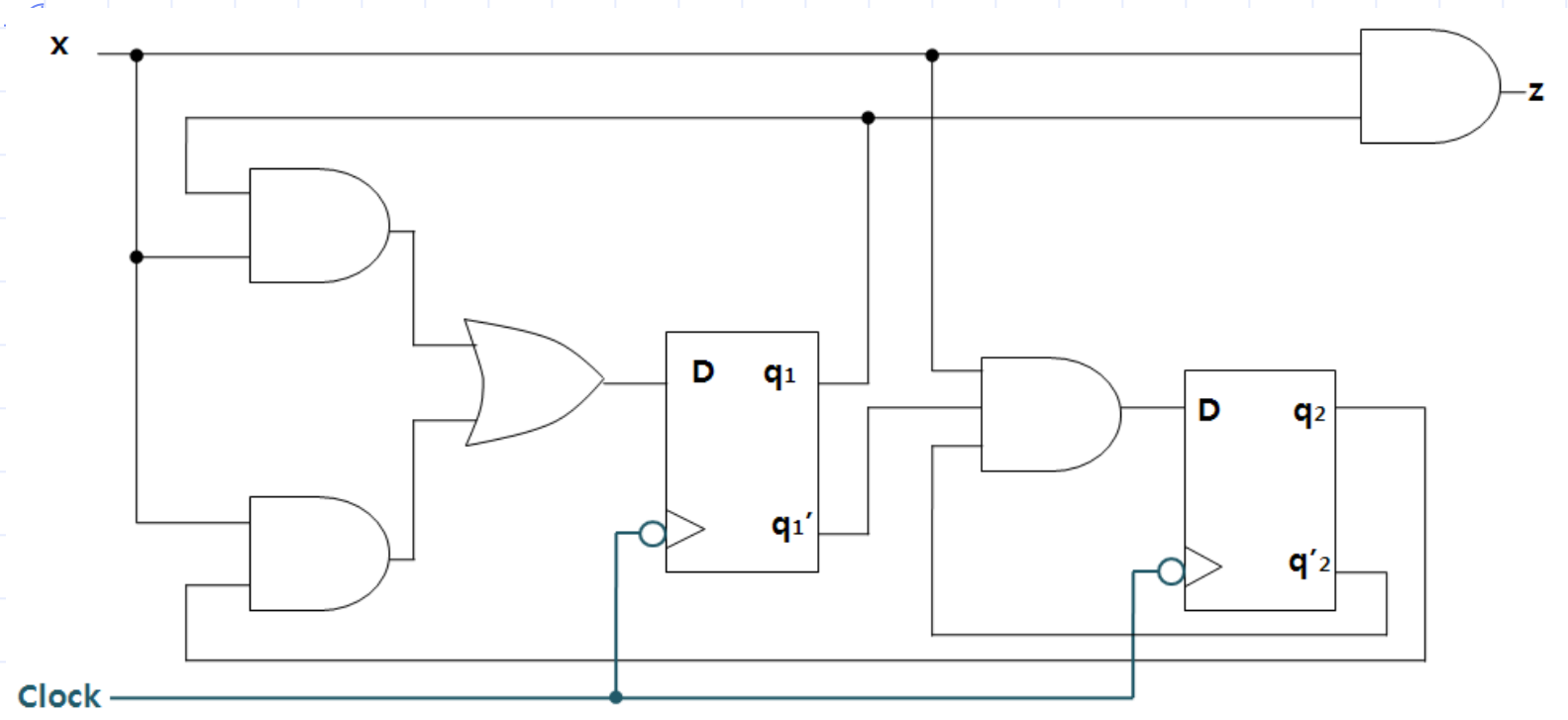
Analysis with D flip-flops

$D_1 = q_1q_2' + xq_1'$
$D_2 = xq_1$
$z = q_2'$

q	q	q_1				z
		x=0		x=1		
0	0	0	0	1	0	1
0	1	0	0	1	0	0
1	0	1	0	1	1	1
1	1	0	0	1	0	0



Analysis with D flip-flops



$D_1 = xq_1 + xq_2$

$D_2 = xq_1'q_2'$

$z = xq_1$

$q_1^* = xq_1 + xq_2$

$q_2^* = xq_1'q_2'$

Analysis with D flip-flops

$D_1 = xq_1 + xq_2$
$D_2 = xq_1'q_2'$
$z = xq_1$
$q_1^* = xq_1 + xq_2$
$q_2^* = xq_1'q_2'$

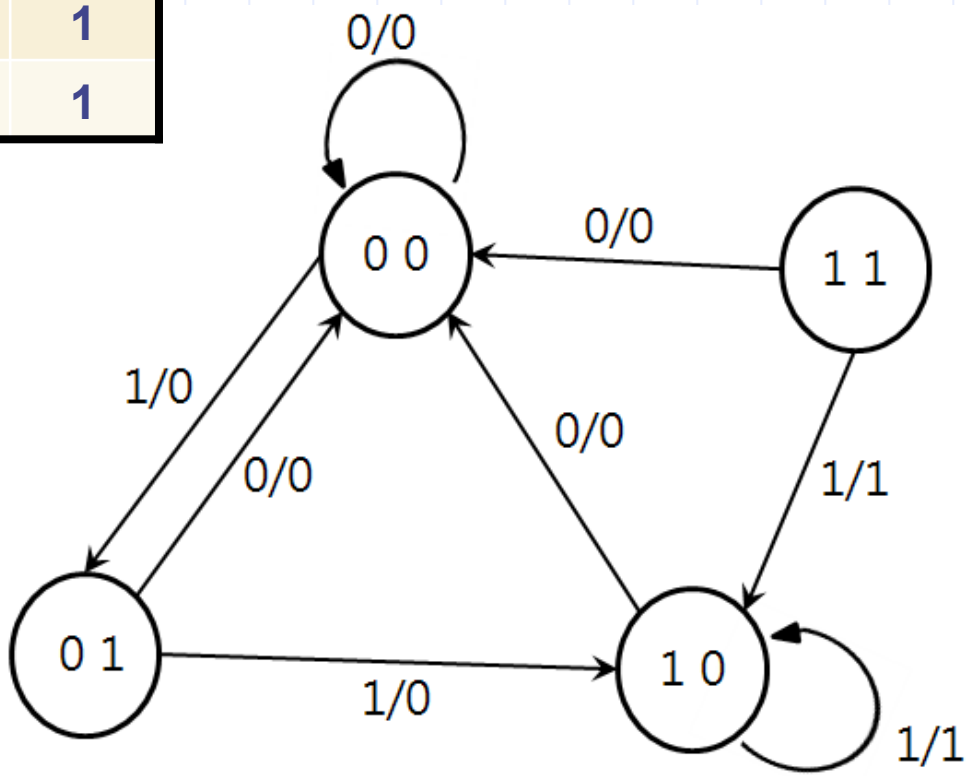
D Flip-Flop

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

q	q	D		D		q_1				z	
		x=0	x=1	x=0	x=1	x=0		x=1		x=0	x=1
0	0	0	0	0	1	0	0	0	1	0	0
0	1	0	1	0	0	0	0	1	0	0	0
1	0	0	1	0	0	0	0	1	0	0	1
1	1	0	1	0	0	0	0	1	0	0	1

Analysis with D flip-flops

q	q	q_1				z	
		x=0		x=1		x=0	x=1
0	0	0	0	0	1	0	0
0	1	0	0	1	0	0	0
1	0	0	0	1	0	0	1
1	1	0	0	1	0	0	1



Analysis with JK flip-flops

- State equation is not the same as the input equation
- Have to refer characteristic table or characteristic equation

Table 5-1
Flip-Flop Characteristic Tables

***JK* Flip-Flop**

J	K	$Q(t+1)$	
0	0	$Q(t)$	No change
0	1	0	Reset
1	0	1	Set
1	1	$Q'(t)$	Complement

Analysis with JK flip-flops

- Input equations

$J_A = B \quad K_A = Bx'$

$J_B = x' \quad K_B = A'x + Ax'$

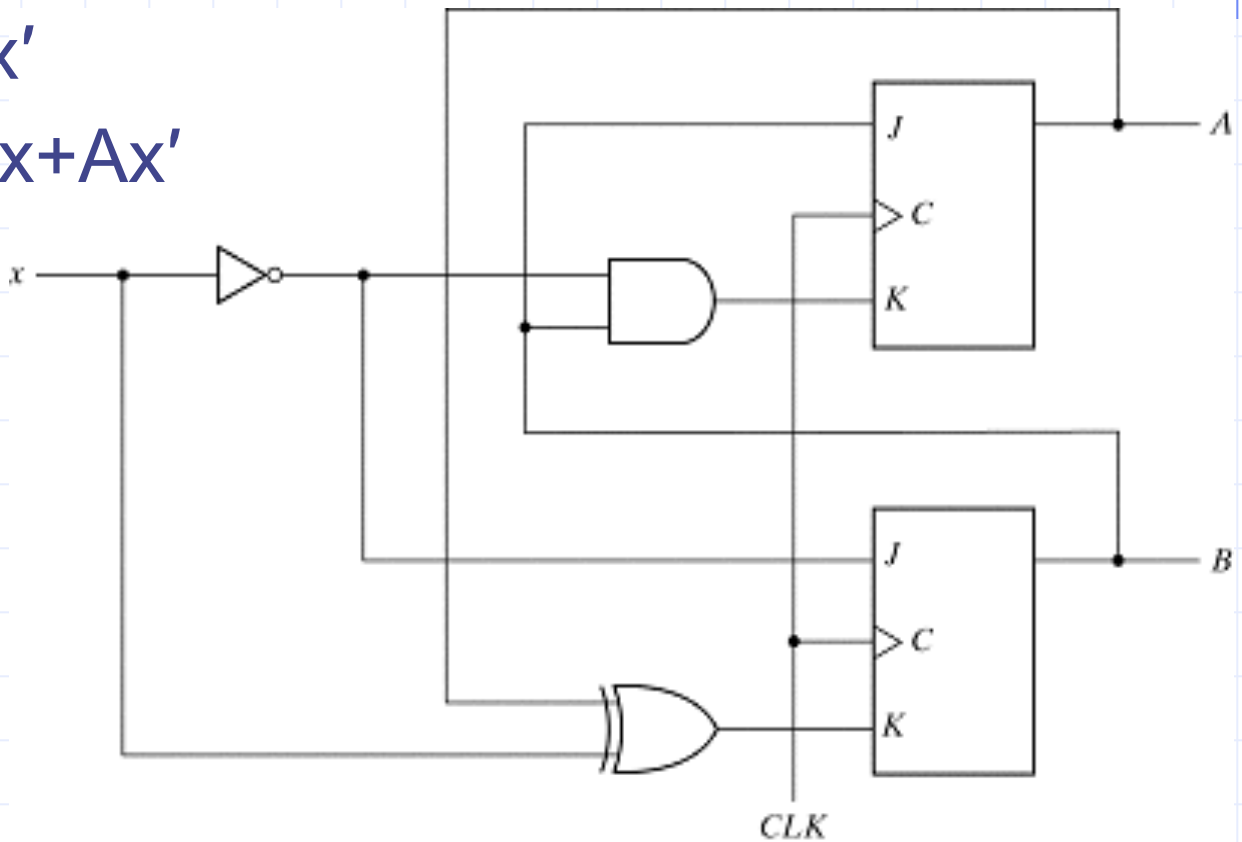


Fig. 5-18 Sequential Circuit with *JK* Flip-Flop

Analysis with JK flip-flops

- State table and state diagram

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

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

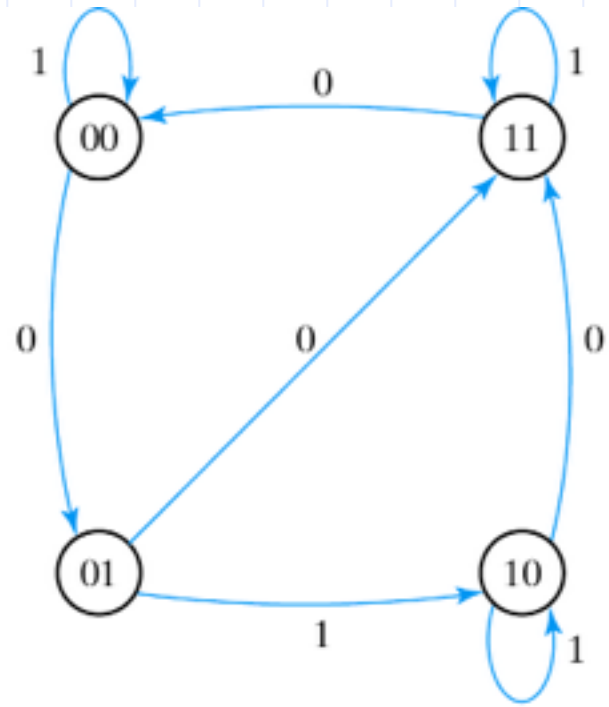
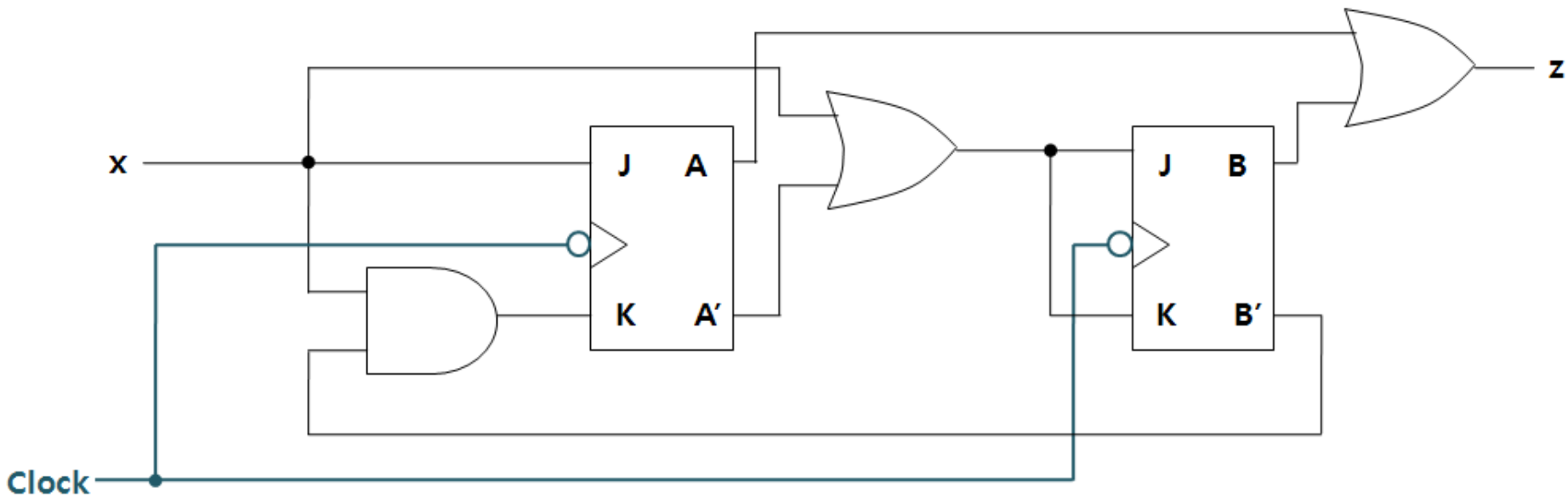


Fig. 5-19 State Diagram of the Circuit of Fig. 5-18

Analysis with JK flip-flops



$J_A = x$

$K_A = xB'$

$J_B = K_B = x + A'$

$z = A + B$

Analysis with JK flip-flops

$J_A = x$
$K_A = xB'$
$J_B = K_B = x + A'$
$z = A + B$

Table 5-1
Flip-Flop Characteristic Tables

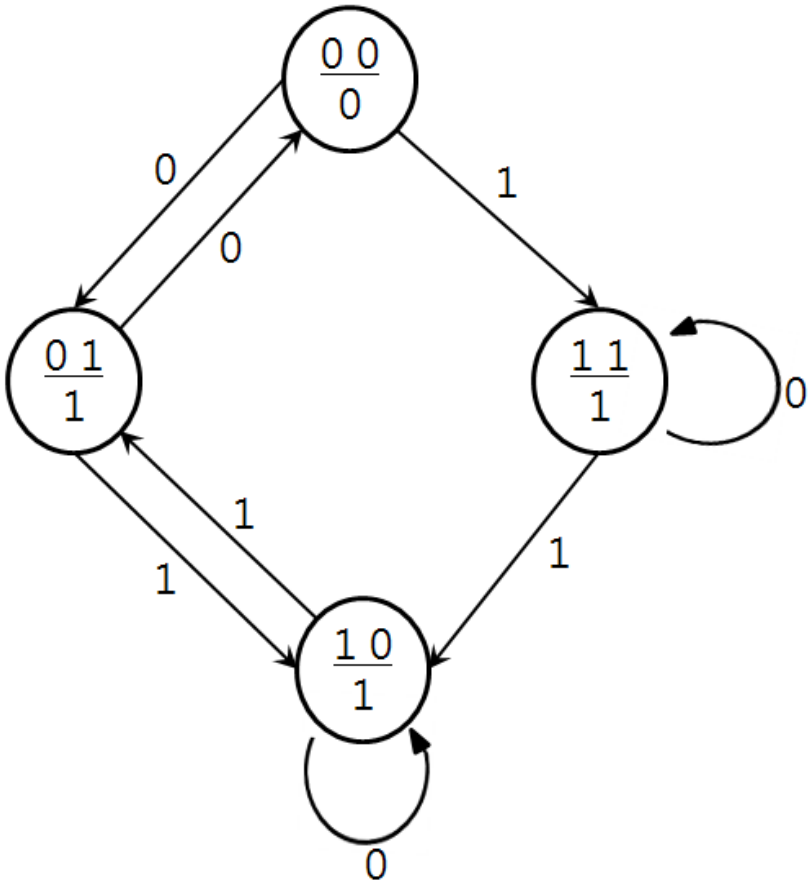
***JK* Flip-Flop**

J	K	$Q(t+1)$	
0	0	$Q(t)$	No change
0	1	0	Reset
1	0	1	Set
1	1	$Q'(t)$	Complement

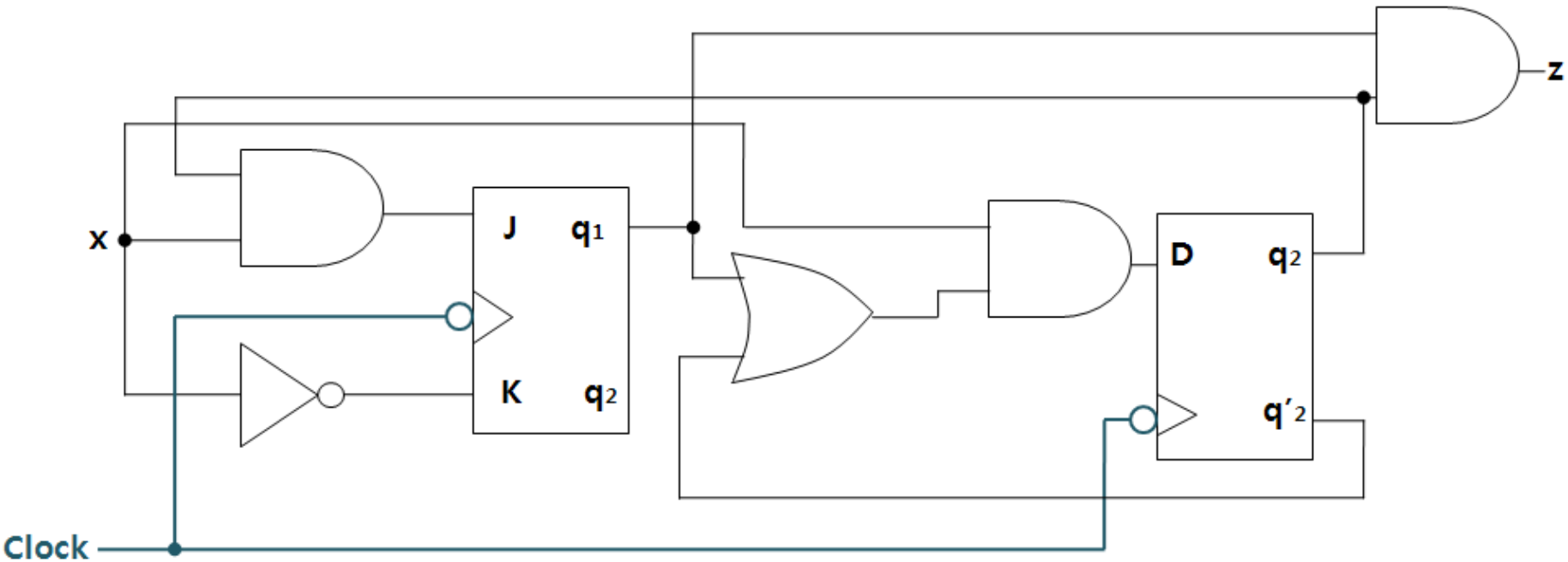
A	B	J	K	J	K	J	K	J	K	A(t+1)		B(t+1)		z
		x=0		x=1		x=0		x=1		x=0		x=1		
0	0	0	0	1	1	1	1	1	1	0	1	1	1	0
0	1	0	0	1	0	1	1	1	1	0	0	1	0	1
1	0	0	0	1	1	0	0	1	1	1	0	0	1	1
1	1	0	0	1	0	0	0	1	1	1	1	1	0	1

Analysis with JK flip-flops

A	B	A(t+1) B(t+1)				z
		x=0		x=1		
0	0	0	1	1	1	0
0	1	0	0	1	0	1
1	0	1	0	0	1	1
1	1	1	1	1	0	1



Analysis with D and JK flip-flops



$J_1 = xq_2$,
$K_1 = x'$
$D_2 = x(q_1+q_2')$
$z = q_1q_2$

Analysis with D and JK flip-flops

$J_1 = xq_2$
$K_1 = x'$
$D_2 = x(q_1+q_2')$
$z = q_1q_2$

Table 5-1
Flip-Flop Characteristic Tables

***JK* Flip-Flop**

J	K	$Q(t+1)$	
0	0	$Q(t)$	No change
0	1	0	Reset
1	0	1	Set
1	1	$Q'(t)$	Complement

***D* Flip-Flop**

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

q	q	J	K	J	K	D	D	q				z
		x=0		x=1		x=0	x=1	x=0		x=1		
0	0	0	1	0	0	0	1	0	0	0	1	0
0	1	0	1	1	0	0	0	0	0	1	0	0
1	0	0	1	0	0	0	1	0	0	1	1	0
1	1	0	1	1	0	0	1	0	0	1	1	1

Analysis with T flip-flops

- Input equations and output equation

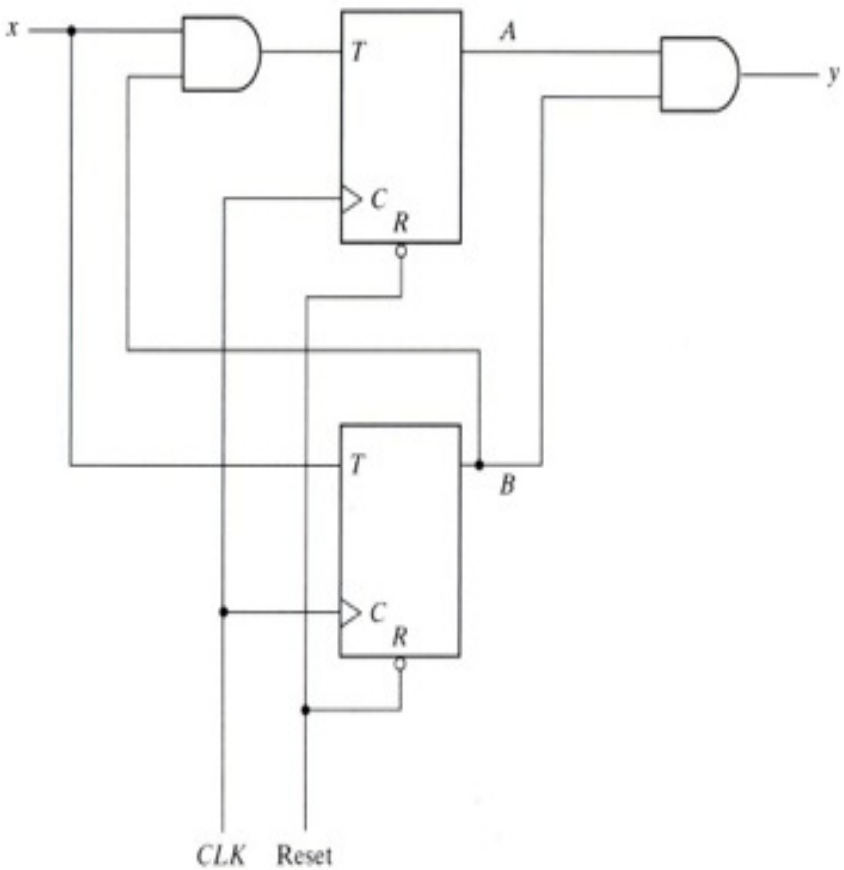
  $T_A = Bx, \quad T_B = x$

  $y = AB$

- State equations are derived from characteristic equation

  $A(t+1) = T_A A' + T_A' A$

  $B(t+1) = T_B B' + T_B' B$

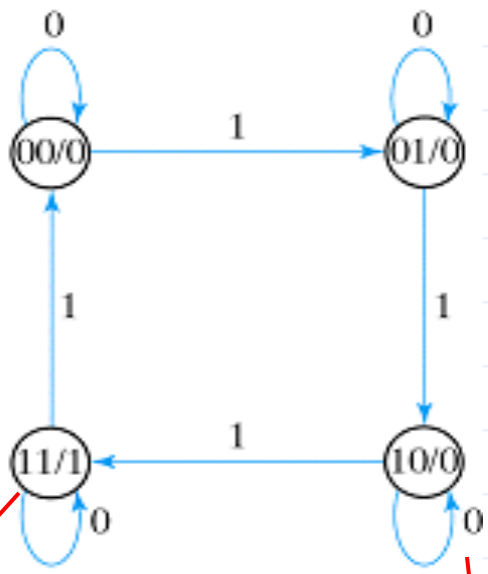


(a) Circuit diagram

Analysis with T flip-flops

Table 5-5
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



(b) State diagram

State/output

Input

Mealy and Moore models

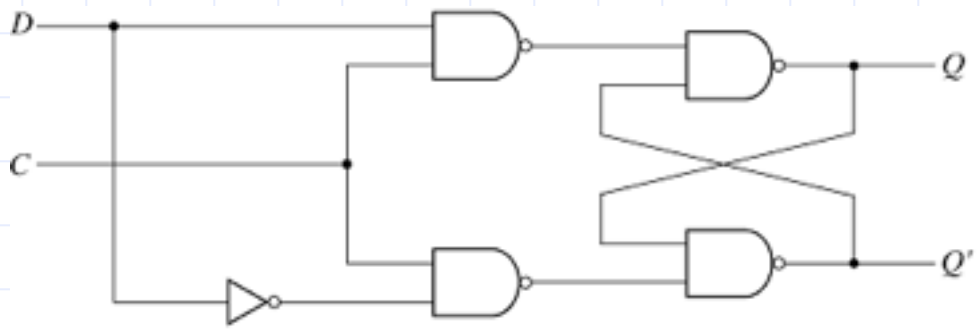
- Mealy model : output is a function of the present state and input
  - Inputs must be synchronized with the clock
  - Outputs must be sampled at the clock edge
- Moore model : output is a function of the present state only
  - Outputs are synchronized with the clock

5-5 HDL for sequential circuits

- Two kinds of behavioral statements
- Initial : executes only once
- Always : executes repeatedly until the simulation terminates

Flip-flops and latches

- D-latch



HDL Example 5-1

```
//Description of D latch (See Fig. 5-6)
module D_latch (Q,D,control);
    output Q;
    input D,control;
    reg Q;
    always @ (control or D)
    if (control) Q = D;      //Same as: if (control == 1)
endmodule
```

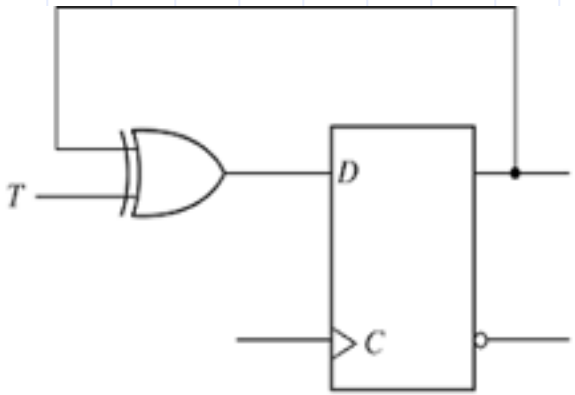
D flip-flop

- D flip-flop with asynchronous reset

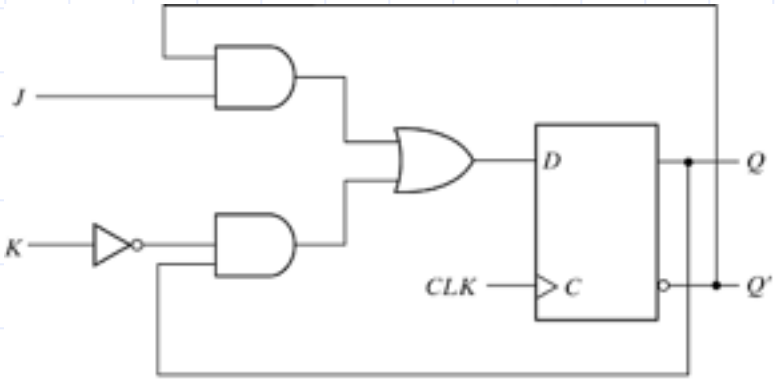
```
//D flip-flop
module D_FF (Q,D,CLK);
    output Q;
    input D,CLK;
    reg Q;
    always @ (posedge CLK)
      Q = D;
endmodule

//D flip-flop with asynchronous reset.
module DFF (Q,D,CLK,RST);
    output Q;
    input D,CLK,RST;
    reg Q;
    always @(posedge CLK or negedge RST)
      if (~RST) Q = 1'b0;     // Same as: if (RST == 0)
      else Q = D;
endmodule
```

T flip-flop from D flip-flop

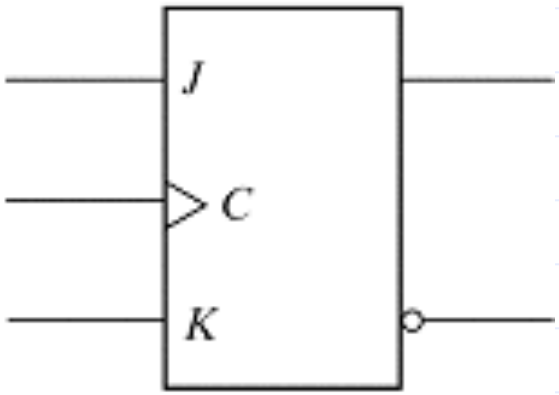
```
//T flip-flop from D flip-flop and gates
module TFF (Q,T,CLK,RST);
    output Q;
    input T,CLK,RST;
    wire DT;
    assign DT = Q ^ T ;
//Instantiate the D flip-flop
    DFF TF1 (Q,DT,CLK,RST);
endmodule
```




```
//JK flip-flop from D flip-flop and gates
module JKFF (Q,J,K,CLK,RST);
    output Q;
    input J,K,CLK,RST;
    wire JK;
    assign JK = (J & ~Q) | (~K & Q);
//Instantiate D flipflop
    DFF JK1 (Q,JK,CLK,RST);
endmodule
```



JK flip-flop



HDL Example 5-4

```
// Functional description of JK flip-flop
module  JK_FF (J,K,CLK,Q,Qnot);
    output Q,Qnot;
    input  J,K,CLK;
    reg  Q;
    assign Qnot = ~ Q ;
    always @ (posedge CLK)
            case ({J,K})
              2'b00: Q = Q;
              2'b01: Q = 1'b0;
              2'b10: Q = 1'b1;
              2'b11: Q = ~ Q;
            endcase
endmodule
```

State diagram

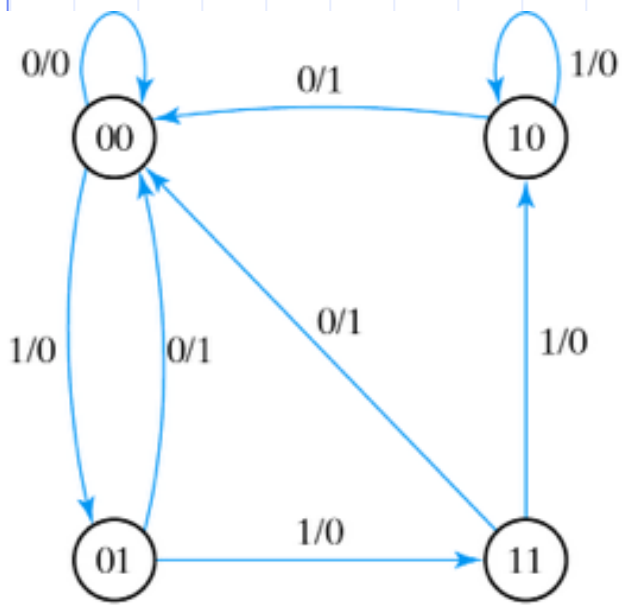


Fig. 5-16 State Diagram of the Circuit of Fig. 5-

(Mealy state diagram)

```
module Mealy_mdl (x,y,CLK,RST);
  input x,CLK,RST;
  output y;
  reg y;
  reg [1:0] Prstate, Nxtstate;
  parameter S0 = 2'b00, S1 = 2'b01, S2 = 2'b10, S3 = 2'b11;
   always @ (posedge CLK or negedge RST)
      if (~RST) Prstate = S0;   //Initialize to state S0
      else Prstate = Nxtstate; //Clock operations
   always @ (Prstate or x)      //Determine next state
         case (Prstate)
            S0: if (x) Nxtstate = S1;
                  else Nxtstate = S0;
            S1: if (x) Nxtstate = S3;
                  else Nxtstate = S0;
            S2: if (~x)Nxtstate = S0;
                  else Nxtstate = S2;
            S3: if (x) Nxtstate = S2;
                  else Nxtstate = S0;
         endcase
   always @ (Prstate or x)      //Evaluate output
         case (Prstate)
            S0: y = 0;
            S1: if (x) y = 1'b0; else y = 1'b1;
            S2: if (x) y = 1'b0; else y = 1'b1;
            S3: if (x) y = 1'b0; else y = 1'b1;
         endcase
  endmodule
```

State diagram

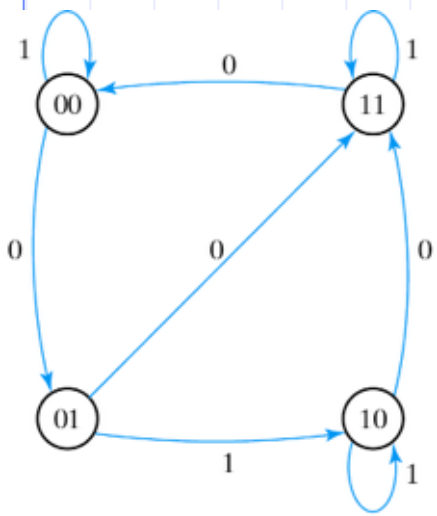
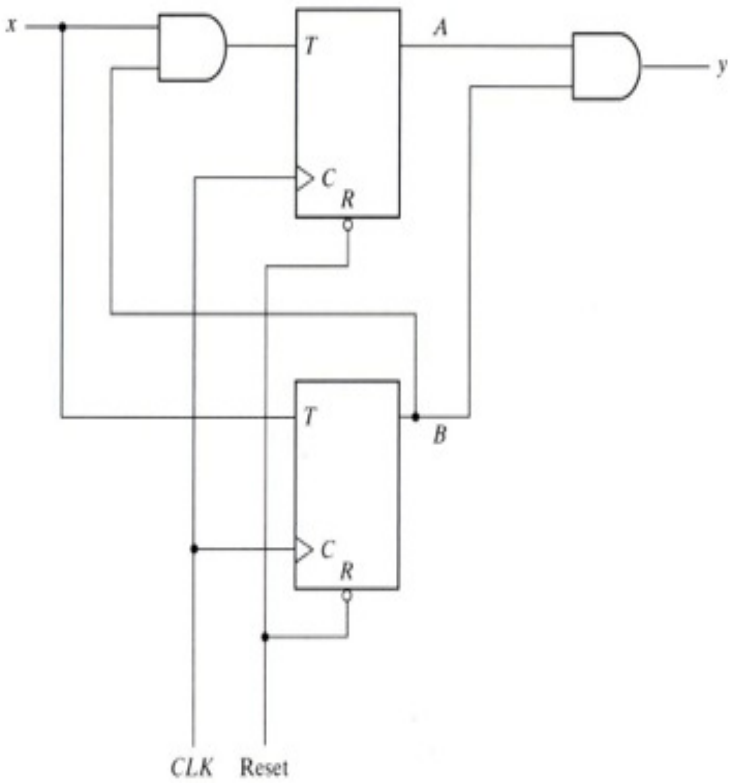


Fig. 5-19 State Diagram of the Circuit of Fig. 5-18

Moore state diagram)

```
//Moore state diagram (Fig. 5-19)
module Moore_mdl (x,AB,CLK,RST);
   input x,CLK,RST;
   output [1:0]AB;
   reg [1:0] state;
   parameter S0 = 2'b00, S1 = 2'b01, S2 = 2'b10, S3 = 2'b11;
      always @ (posedge CLK or negedge RST)
         if (~RST) state = S0;  //Initialize to state S0
         else
         case (state)
           S0: if (~x) state = S1; else state = S0;
           S1: if (x)  state = S2; else state = S3;
           S2: if (~x) state = S3; else state = S2;
           S3: if (~x) state = S0; else state = S3;
         endcase
   assign AB = state;          //Output of flip-flops
endmodule
```

Structural description



(a) Circuit diagram

```
module Tcircuit (x,y,A,B,CLK,RST);
   input x,CLK,RST;
   output y,A,B;
   wire TA,TB;
//Flip-flip input equations
   assign TB = x,
          TA = x & B;
//Output equation
   assign y = A & B;
//Instantiate T flip-flops
   T_FF BF (B,TB,CLK,RST);
   T_FF AF (A,TA,CLK,RST);
endmodule
```

```
module T_FF (Q,T,CLK,RST);
   output Q;
   input T,CLK,RST;
   reg Q;
     always @ (posedge CLK or negedge RST)
       if (~RST) Q = 1'b0;
       else Q = Q ^ T;
endmodule
```

Structural description

(Stimulus and output)

```
module testTcircuit;
  reg x,CLK,RST;  //inputs for circuit
  wire y,A,B;     //output from circuit
  Tcircuit TC (x,y,A,B,CLK,RST);  // instantiate circuit
  initial
     begin
         RST = 0;
         CLK = 0;
      #5 RST = 1;
         repeat (16)
      #5 CLK = ~CLK;
     end
  initial
     begin
          x = 0;
      #15 x = 1;
          repeat (8)
      #10 x = ~ x;
     end
endmodule
```

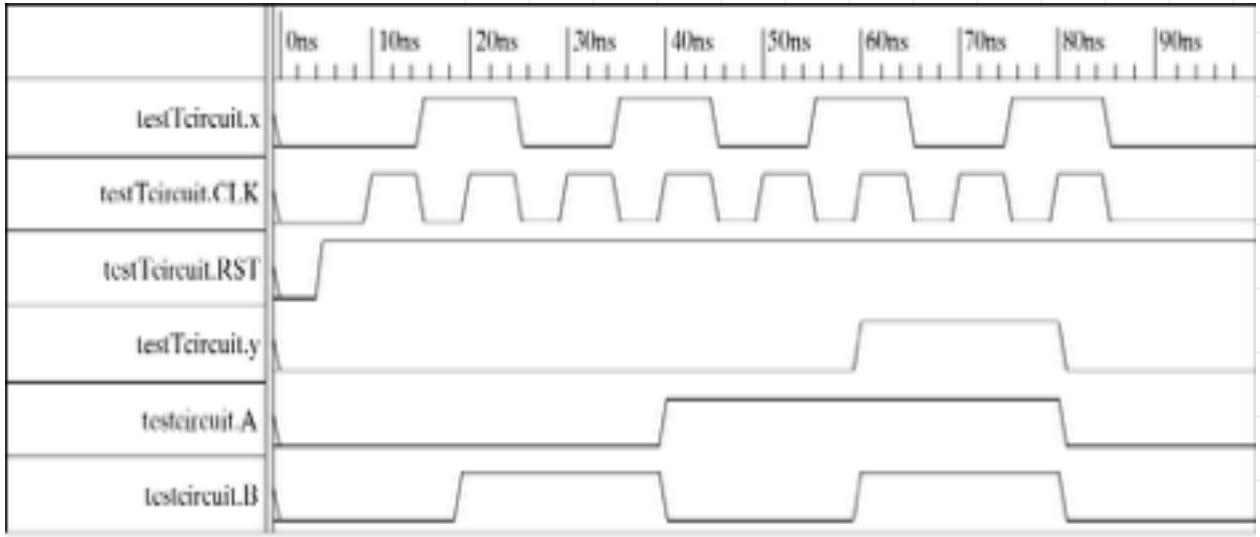


Fig. 5-21 Simulation Output of HDL Example 5-7

5-6 State reduction and assignment

- State reduction is used to reduce the number of flip-flop
- Only input/output sequences are important
- Interested in present states that go to the same next state and have the same output

State reduction

Table 5-6
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	f	0	1
e	a	f	0	1
f	g	f	0	1
g	a	f	0	1

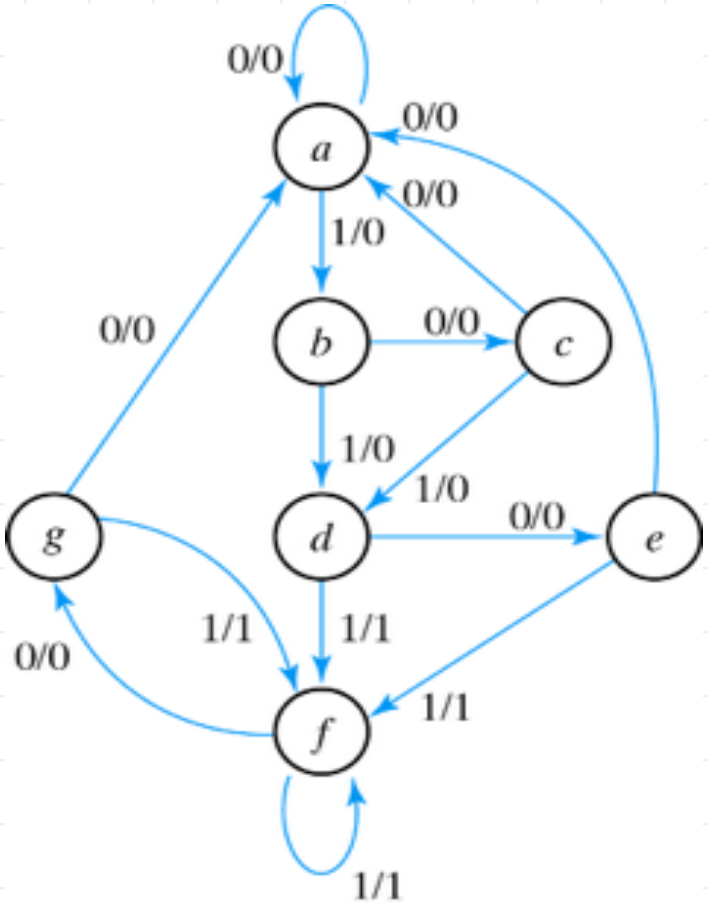


Fig. 5-22 State Diagram

State reduction

Table 5-7
Reducing the 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	f	0	1
e	a	f	0	1
f	e	f	0	1

State reduction

Table 5-8
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

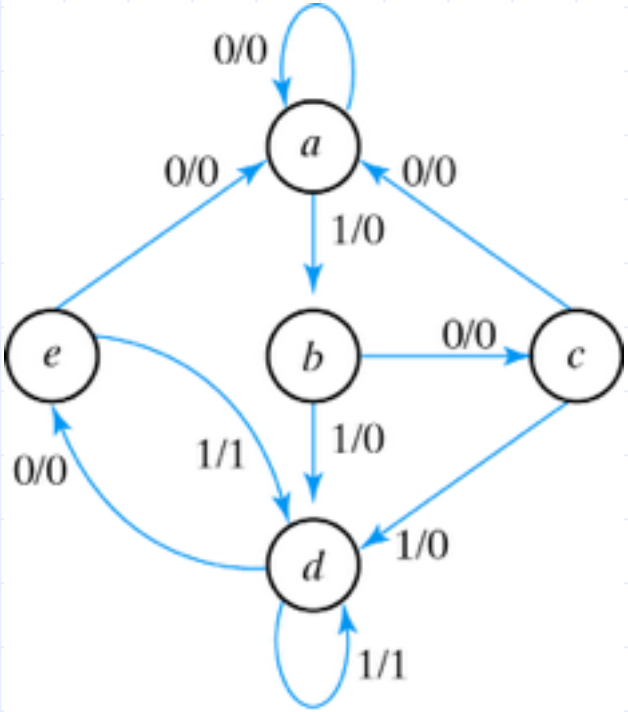


Fig. 5-23 Reduced State Diagram

State assignment

- m states circuit, codes must contain n bits where $2^n \geq m$
- Three possible binary state assignments

Table 5-9
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

5-7 Design procedure

- Sequential circuit design : requires state table
  ⇔ Combinational circuit : truth table
- The number of flip-flop is determined from the number of states in circuit
  - If 2^n states exist, there are n flip-flops

Design procedure

- Design steps
  1) Derive a state diagram or state table
  2) Reduce the number of states if necessary
  3) Assign binary code to the state
  4) Choose the type of flip-flops to be used
  5) Derive the flip-flop input equations and output equations
  6) Draw the logic diagram

Derive a state diagram

- Sequential detector
  - Three or more consecutive 1's in a string of bits coming through an input line

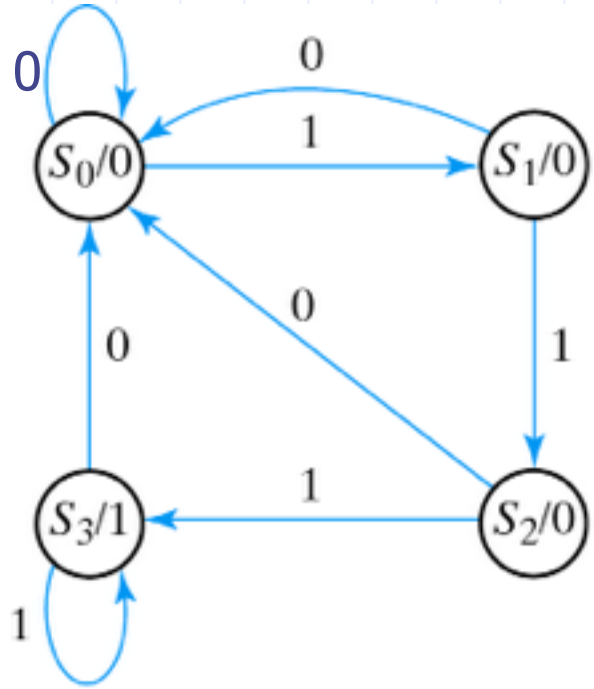


Fig. 5-24 State Diagram for Sequence Detector

S0: 00, S1: 01, S2:10, S3:11

Synthesis using D flip-flops

- Input equations are obtained directly from the next states

Table 5-11
State Table for Sequence Detector

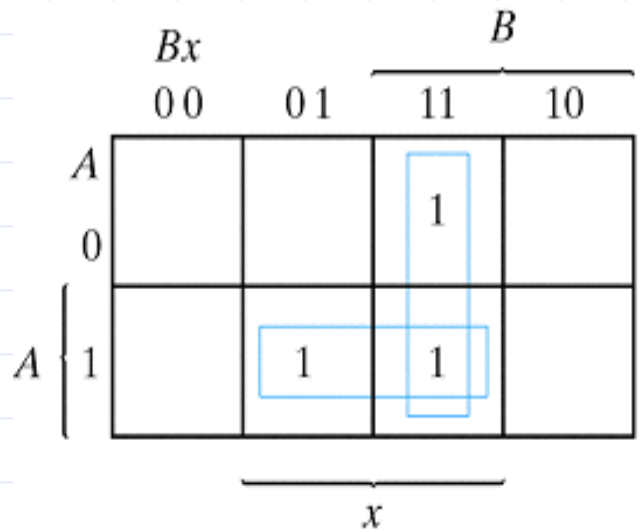
Present State		Input	Next State		Output
A	*B*	*x*	*A*	*B*	*y*
0	0	0	0	0	0
0	0	1	0	1	0
0	1	0	0	0	0
0	1	1	1	0	0
1	0	0	0	0	0
1	0	1	1	1	1
1	1	0	0	0	0
1	1	1	1	1	1

Synthesis using D flip-flops

Table 5-11
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	1
1	1	0	0	0	0
1	1	1	1	1	1

Q	*Qnext*	*D*
0	0	0
0	1	1
1	0	0
1	1	1



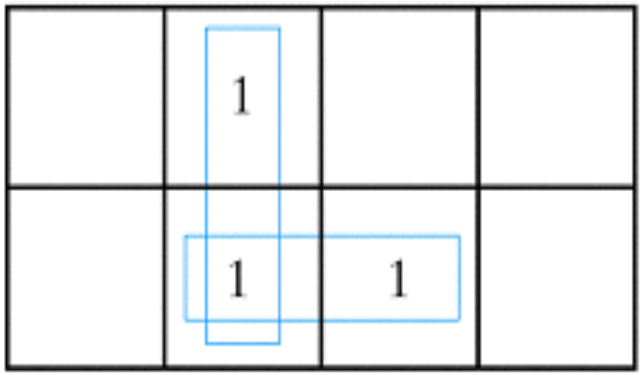
$$D_A = Ax + Bx$$

Synthesis using D flip-flops

Table 5-11
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	1
1	1	0	0	0	0
1	1	1	1	1	1

Q	$Qnext$	D
0	0	0
0	1	1
1	0	0
1	1	1



$$D_B = Ax + B'x$$

Synthesis using D flip-flops

Table 5-11
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	1
1	1	0	0	0	0
1	1	1	1	1	1

	1	1	

$$y = Ax$$

Synthesis using D flip-flops

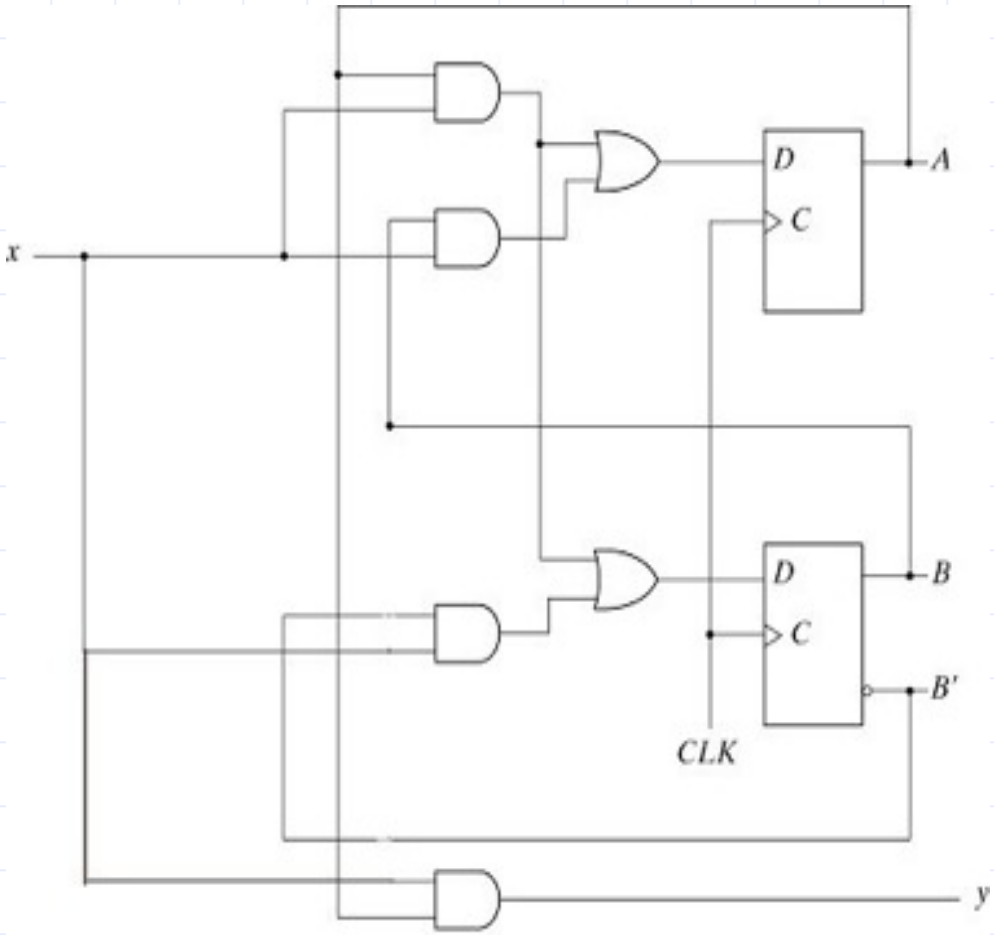


Fig. 5-26 Logic Diagram of Sequence Detector

Synthesis with JK flip-flops

- Input equations evaluated from the present state to next state transition

Table 5-13
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	$Qnext$	J	K
0	0	0	×
0	1	1	×
1	0	×	1
1	1	×	0

Synthesis using JK flip-flops

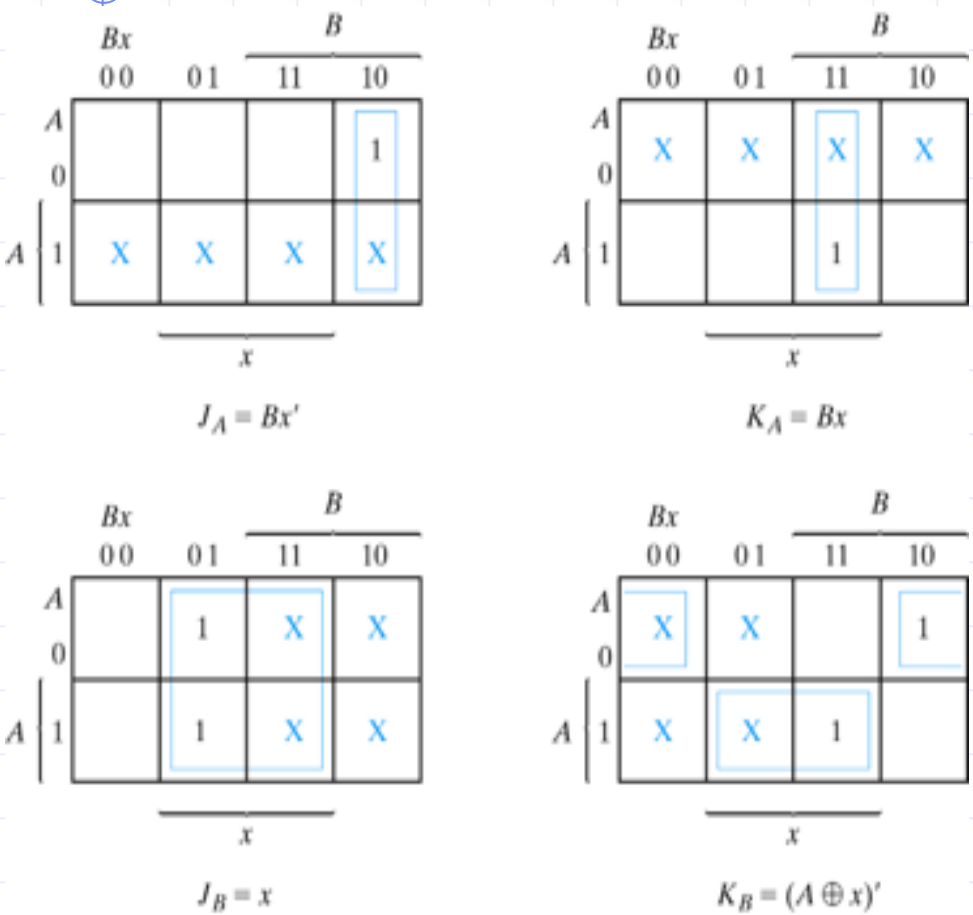


Fig. 5-27 Maps for J and K Input Equations

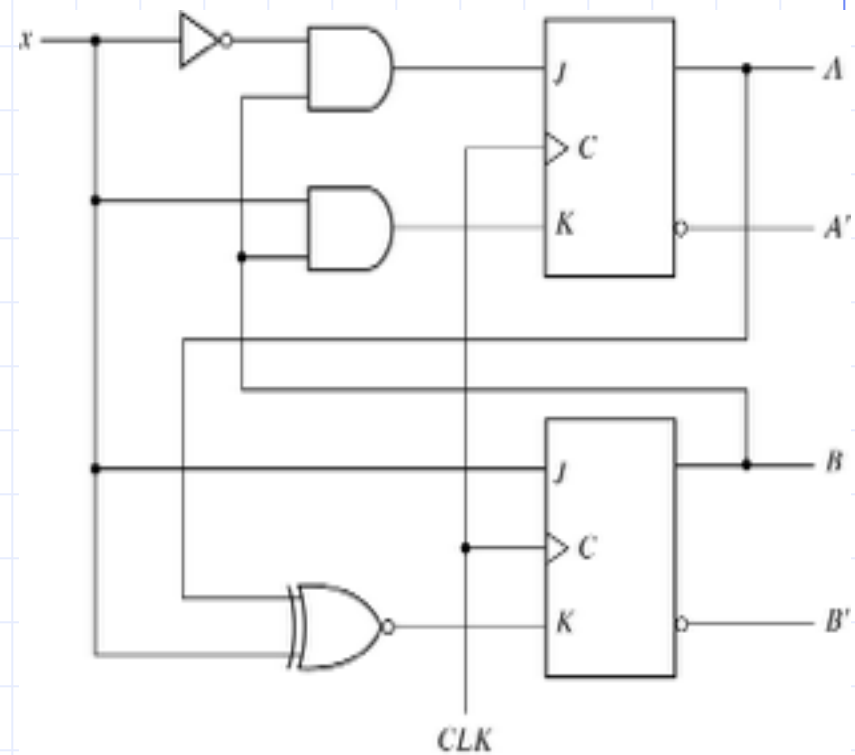


Fig. 5-28 Logic Diagram for Sequential Circuit with JK Flip-Flops

Synthesis using T flip-flops

- 3-bit binary counter
  - 3-bit counter has 3 flip-flops and can count from 0 to 2^n-1(n=3)

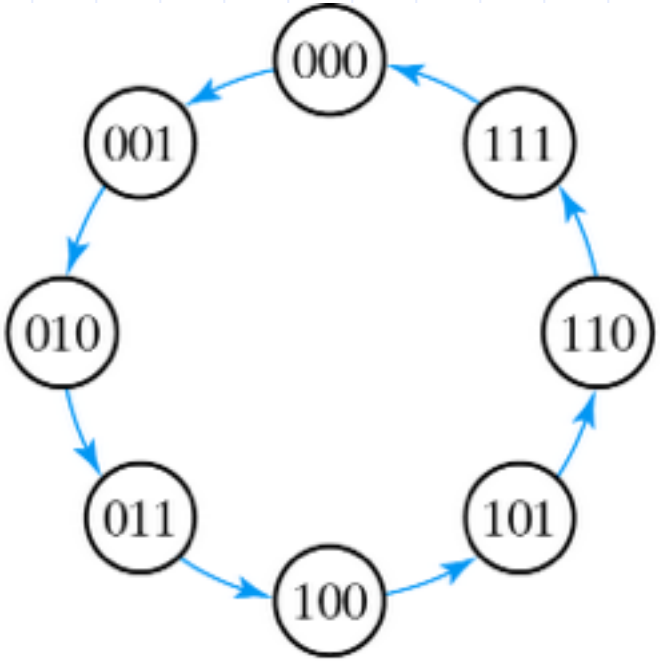


Fig. 5-29 State Diagram of 3-Bit Binary Counter

Synthesis using T flip-flops

Table 5-14
State Table for 3-Bit Counter

Present State			Next State			Flip-Flop Inputs		
A_2	A_1	A_0	A_2	A	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	1	1
1	1	1	0	0	0	1	1	1

Q	$Qnext$	T
0	0	0
0	1	1
1	0	1
1	1	0

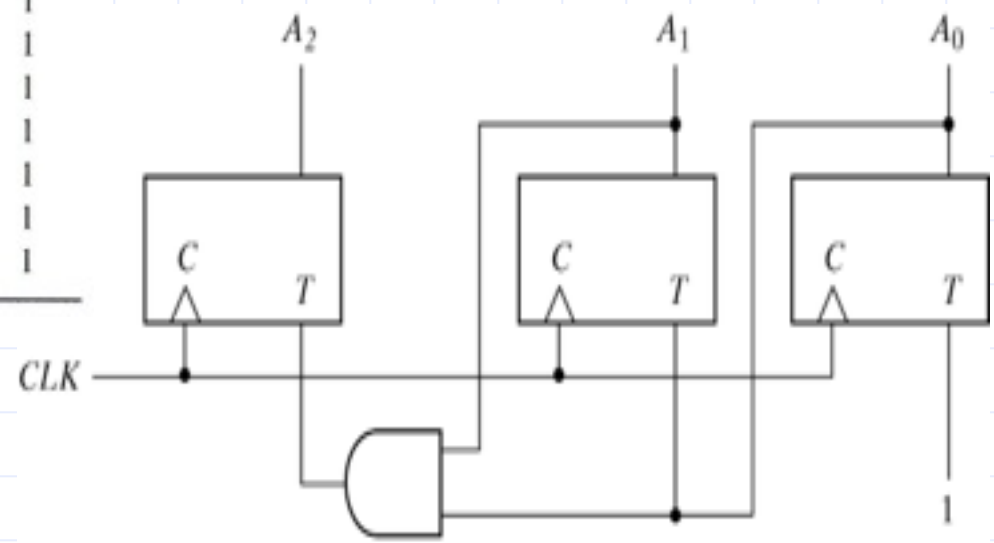


Fig. 5-31 Logic Diagram of 3-Bit Binary Counter

$T_{A2}=A_1A_0$, $T_{A1}=A_0$, $T_{A0}=1$