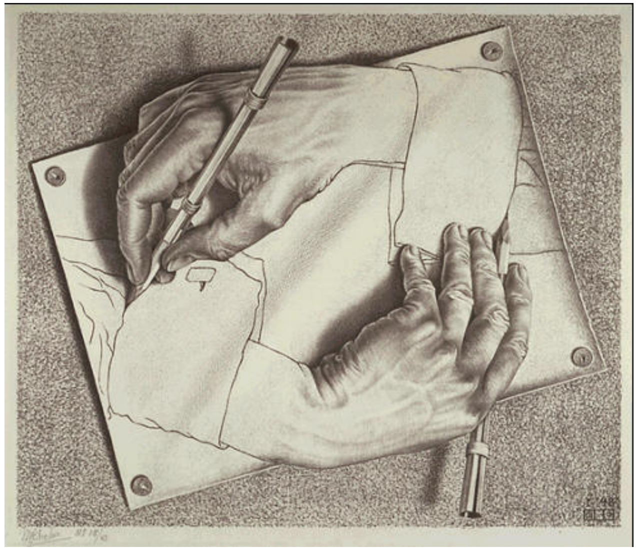
Switching Circuits & Logic Design

Jie-Hong Roland Jiang
江介宏

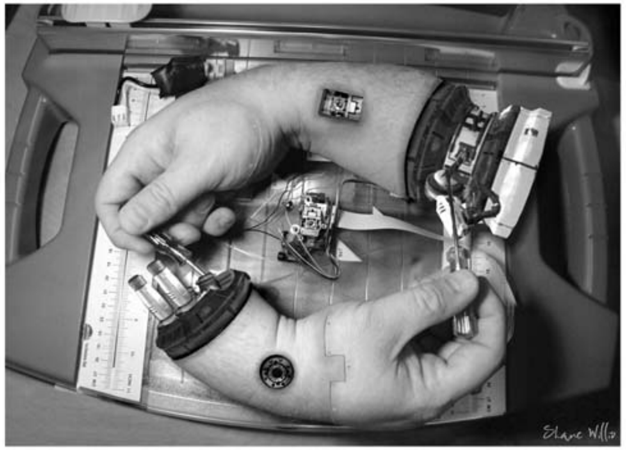
Department of Electrical Engineering
National Taiwan University

Fall 2012

§12 Registers and Counters



Drawing Hands
M.C. Escher, 1948



http://img106.imageshack.us/

Outline

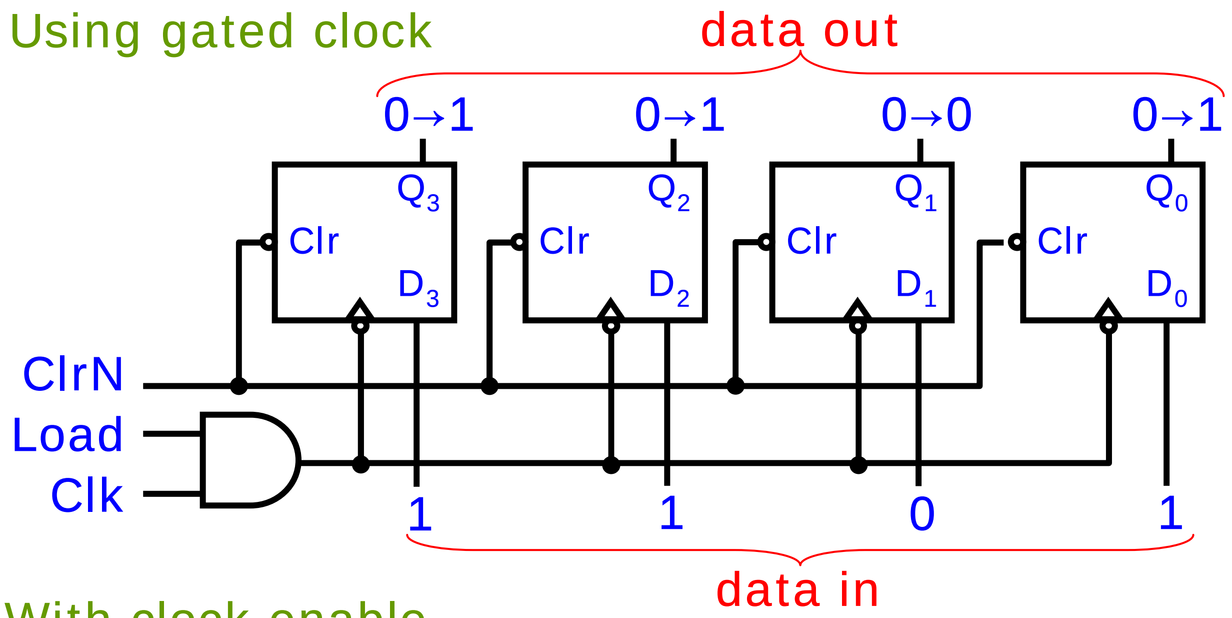
- Registers and register transfers
- Shift registers
- Design of binary counters
- Counters for other sequences
- Counter design using S-R and J-K flip-flops
- Derivation of flip-flop input equations

Registers and Register Transfers

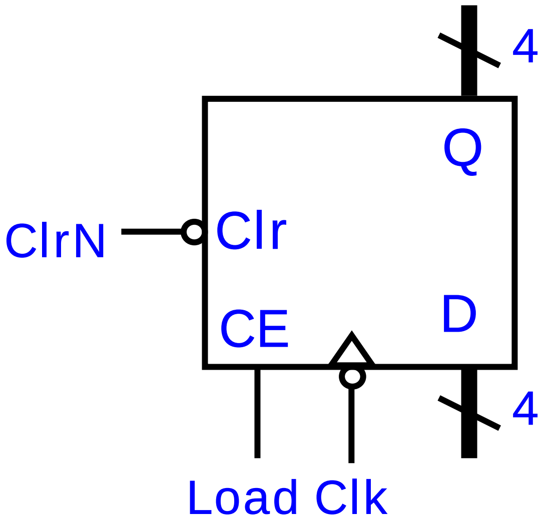
- Several D flip-flops may be grouped together with a common clock to form a **register**

Registers and Register Transfers
4-Bit D Flip-Flop Register

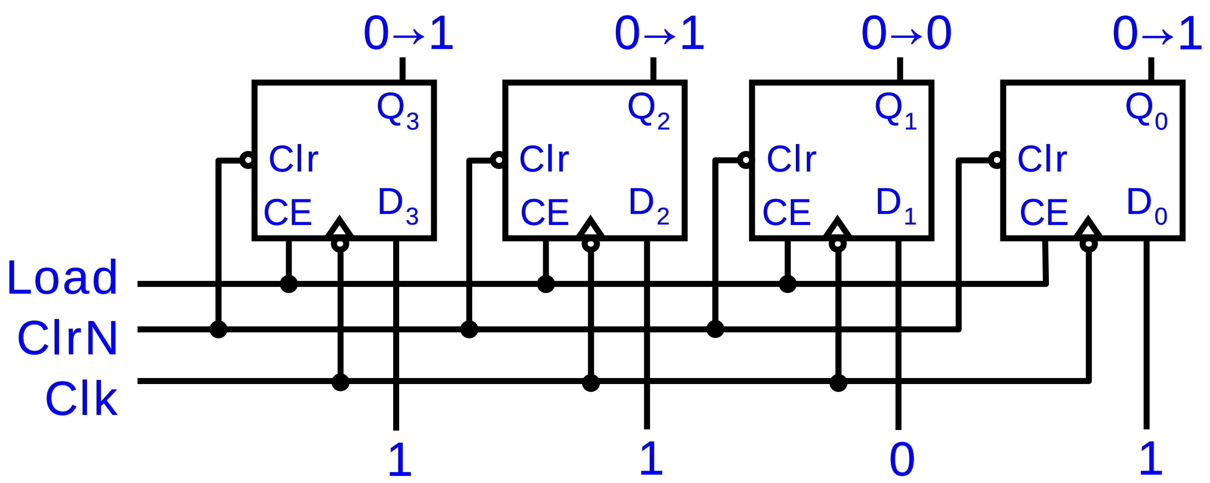
Using gated clock



Symbol



With clock enable



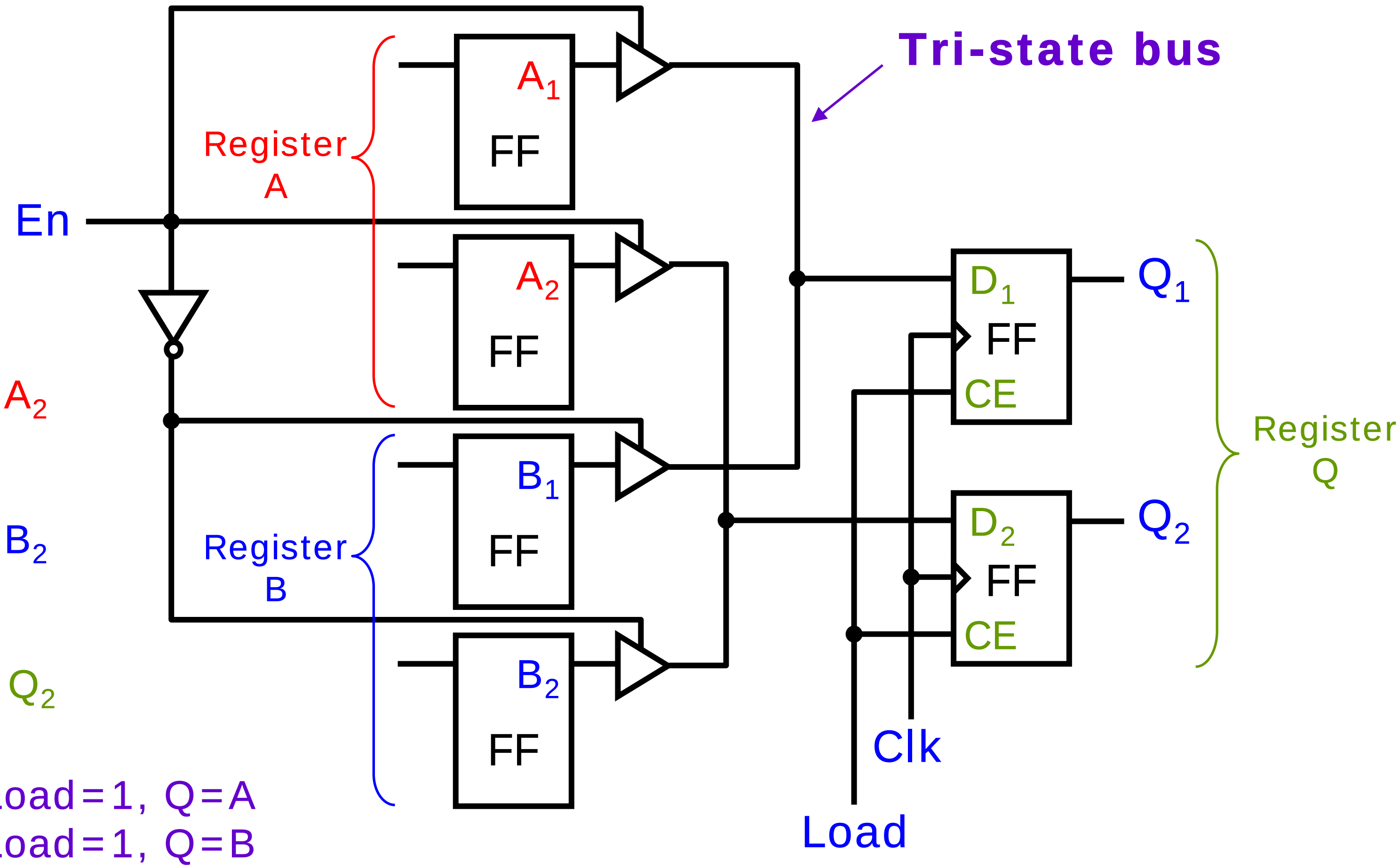
Registers and Register Transfers
Data Transfer between Registers

Register A
  = FFs A_1 and A_2

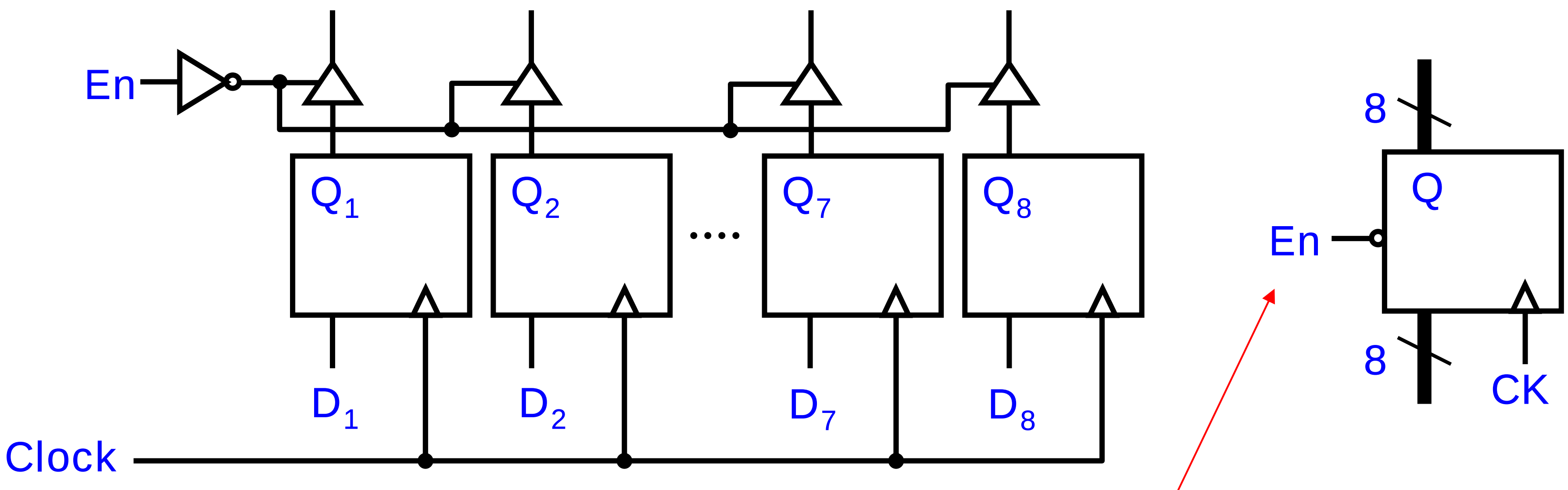
Register B
  = FFs B_1 and B_2

Register Q
  = FFs Q_1 and Q_2

If En=1 and Load=1, Q=A
If En=0 and Load=1, Q=B

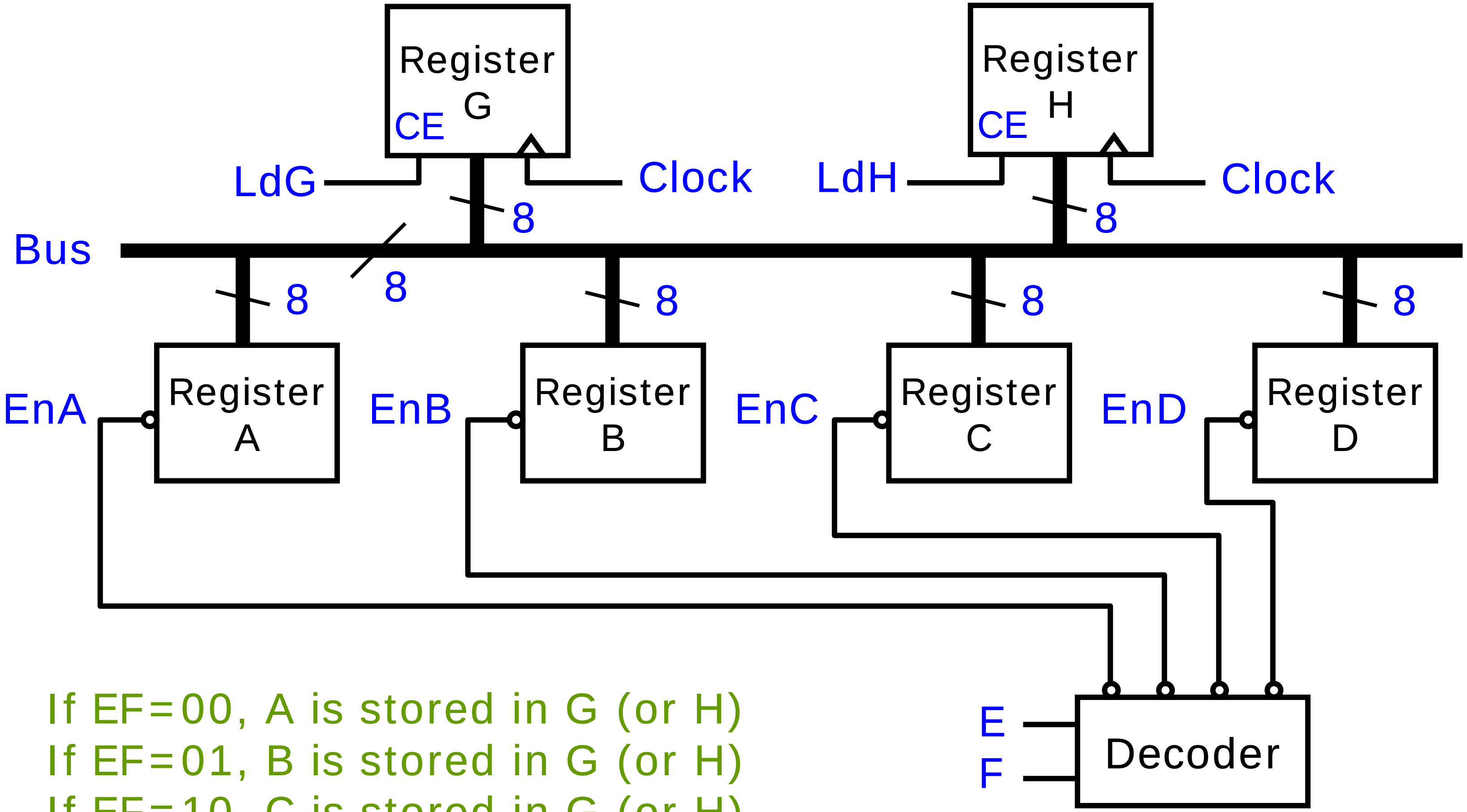


Registers and Register Transfers
8-Bit Register with Tri-State Output



Different from clock enable

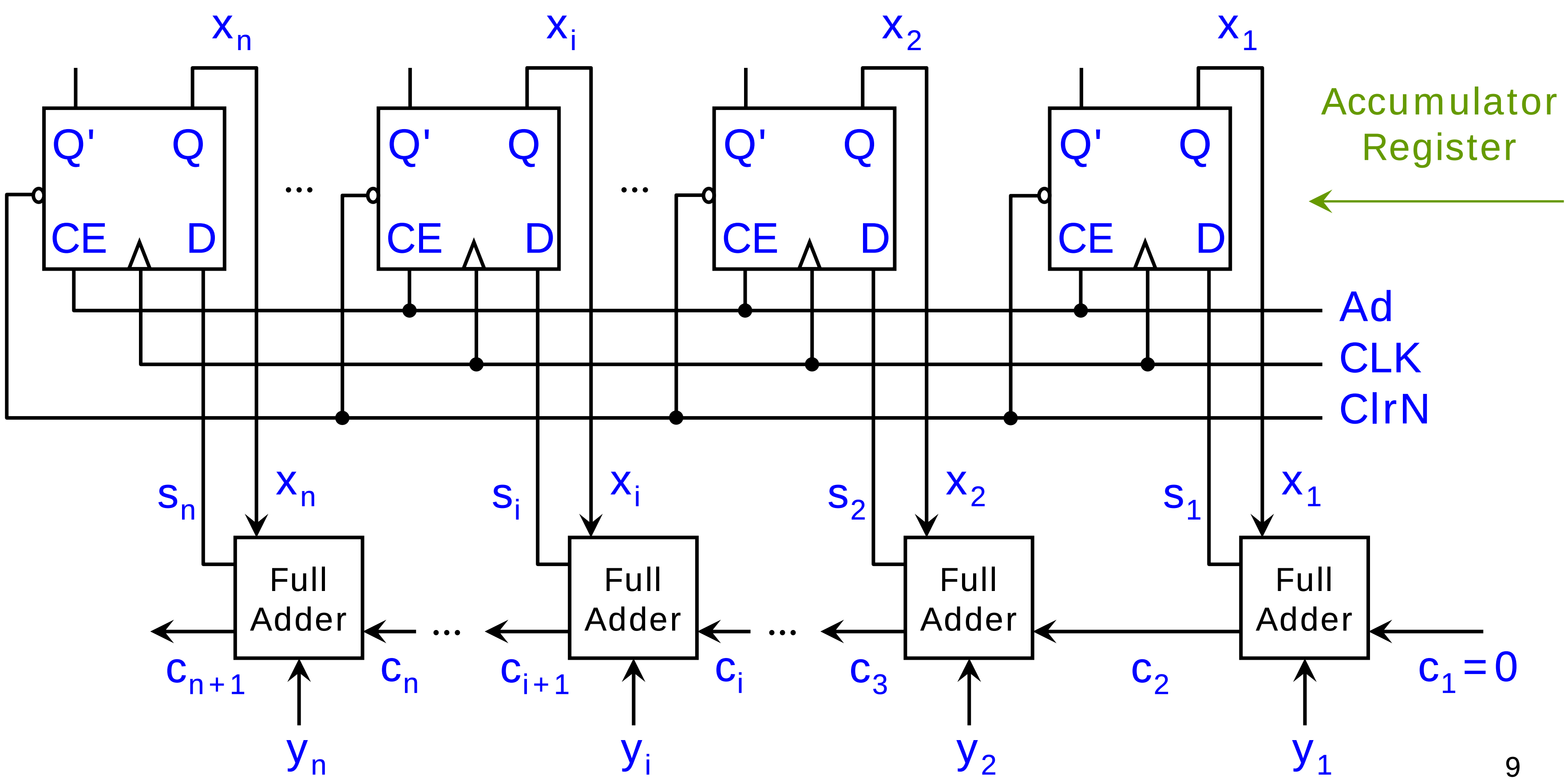
Registers and Register Transfers
Data Transfer Using a Tri-State Bus



If EF=00, A is stored in G (or H)
If EF=01, B is stored in G (or H)
If EF=10, C is stored in G (or H)
If EF=11, D is stored in G (or H)

Registers and Register Transfers
Parallel Adder with Accumulator

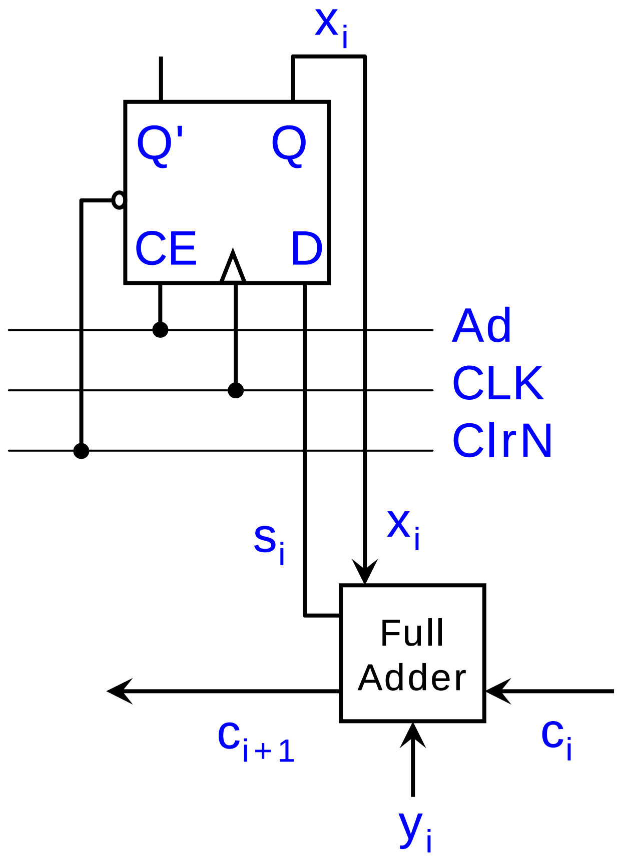
- **Accumulator**:
  A register of FFs that can store one number and add a second number to it, leaving the result stored in it



Registers and Register Transfers
Parallel Adder with Accumulator (cont'd)

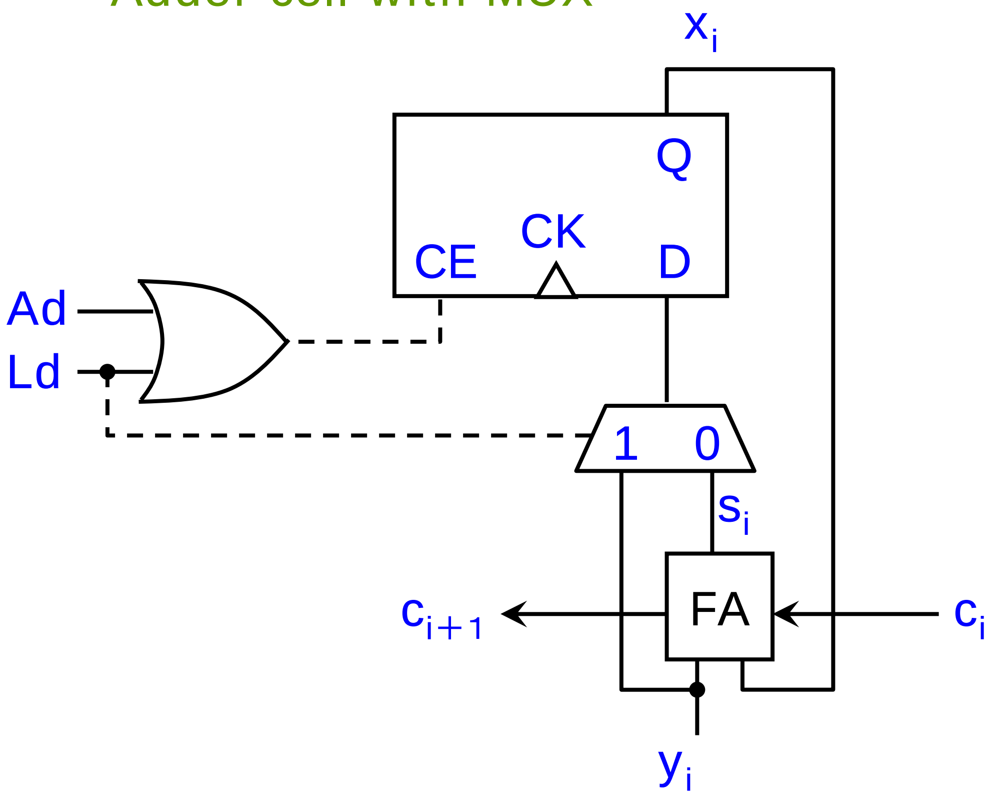
- Implementation of adder cells (module-based design)
  E.g., using Verilog

Adder cell without MUX



Need to clear before loading X to Y

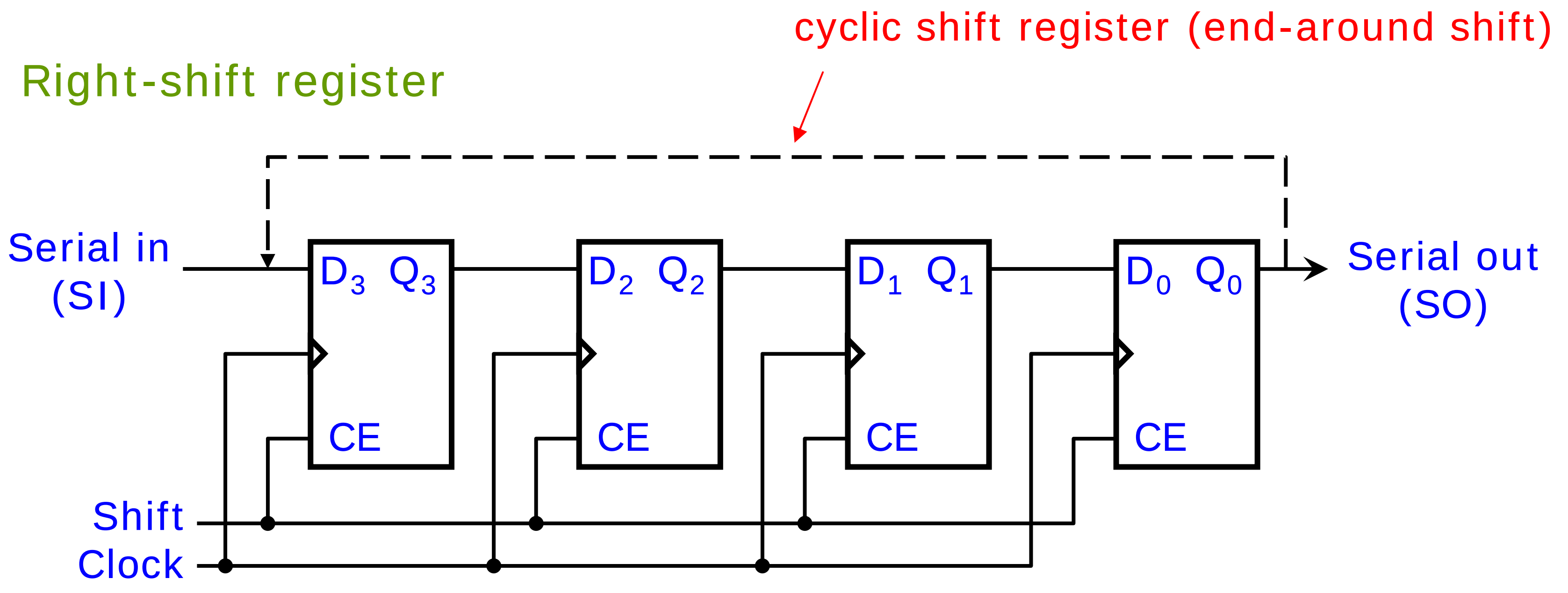
Adder cell with MUX



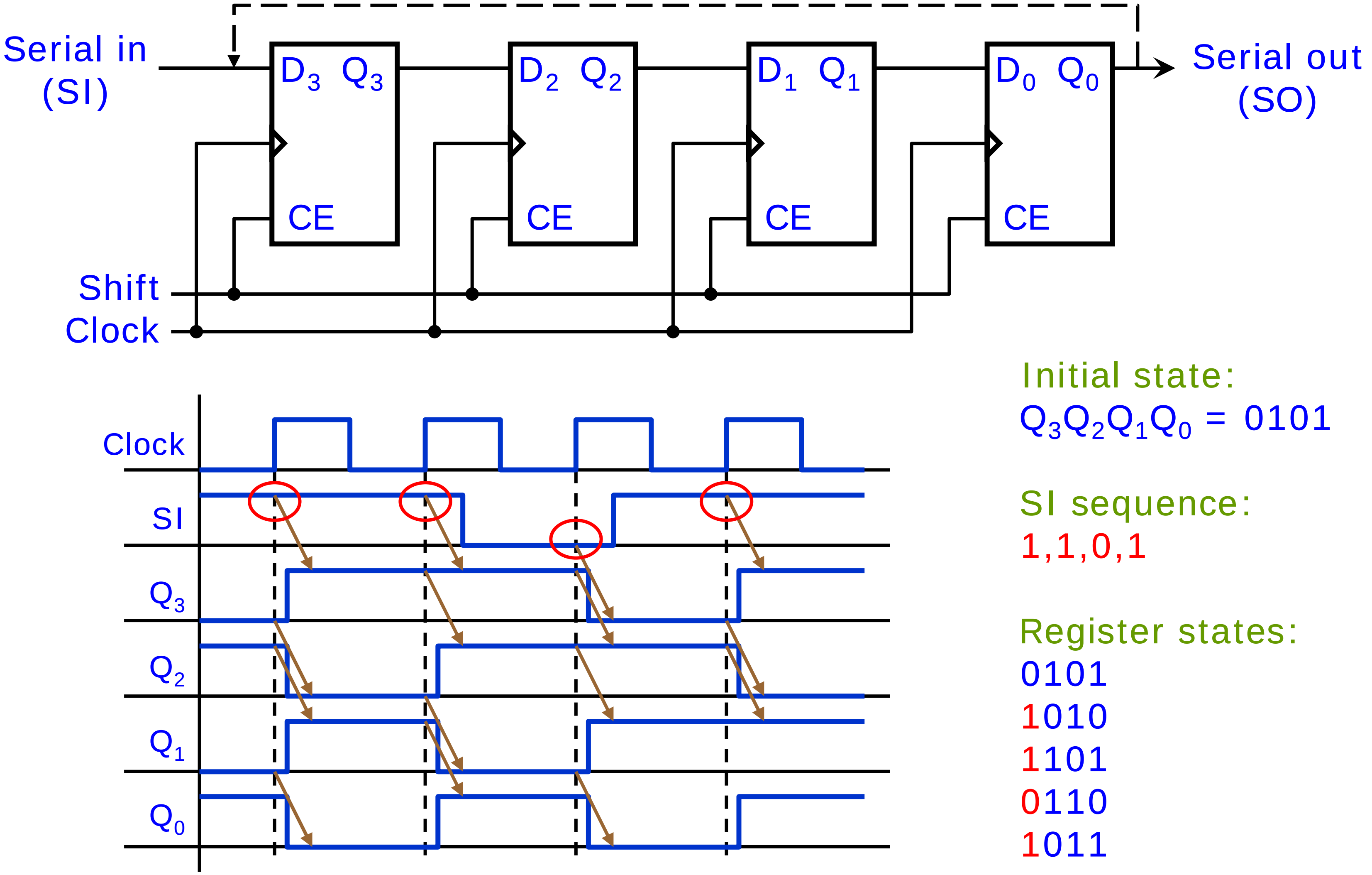
No need to clear before loading X

Shift Registers

- A **shift register** is a register where binary data can be stored and shifted to the left/right when a shift signal is applied



Shift Registers
Right-Shift Register



Initial state:
$Q_3Q_2Q_1Q_0$ = 0101

SI sequence:
1,1,0,1

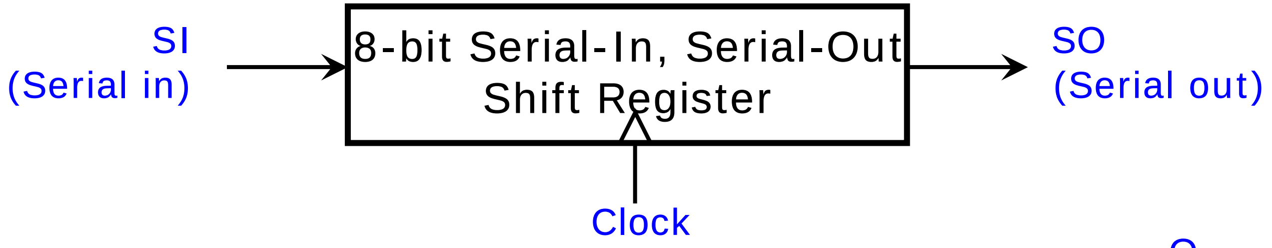
Register states:
0101
1010
1101
0110
1011

Shift Registers
Serial-In, Serial-Out Shift Register

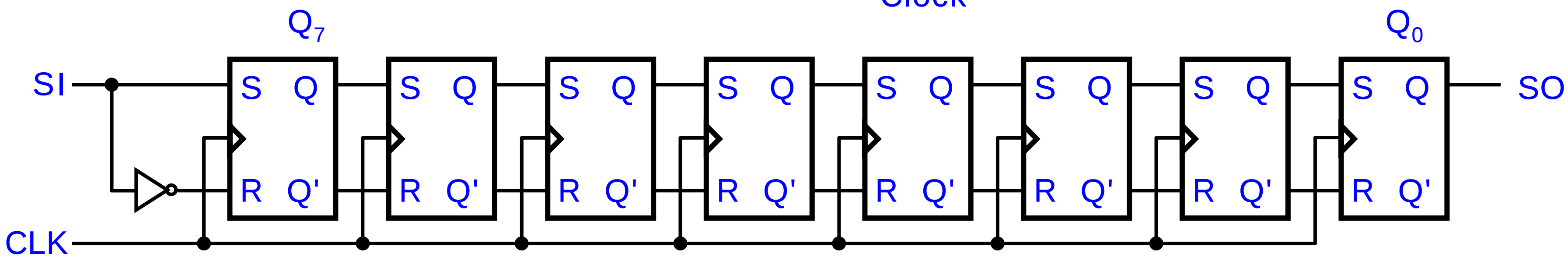
- **Serial in**
  - Data is shifted into the first flip-flop one bit at a time
- **Serial out**
  - Data can only be read out of the last flip-flop one bit at a time

Shift Registers
8-Bit Serial-In, Serial-Out Shift Register

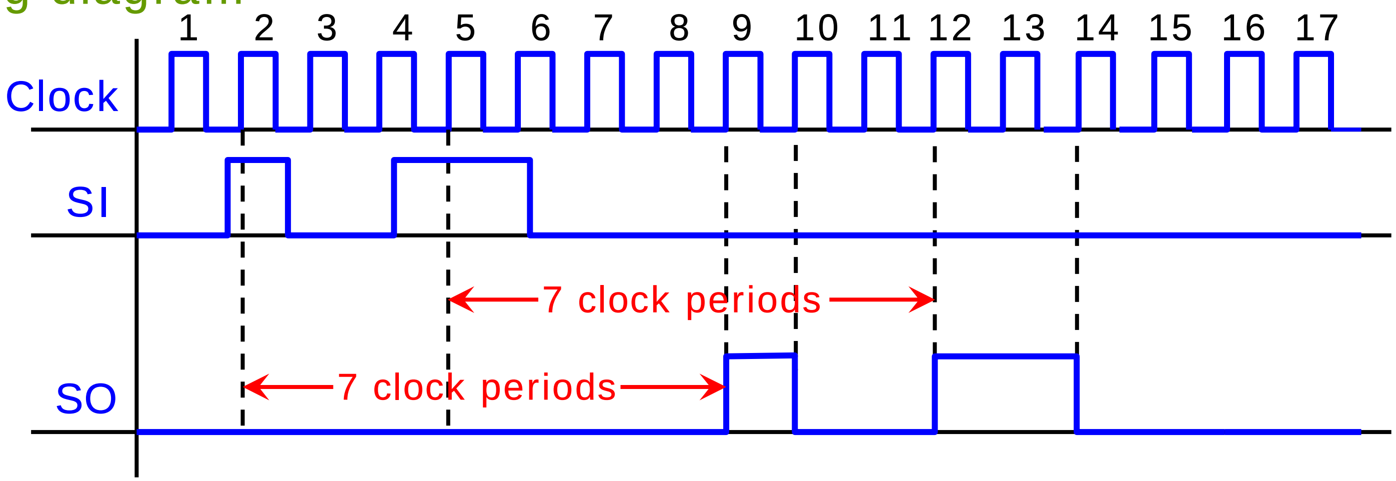
Block diagram



Logic diagram

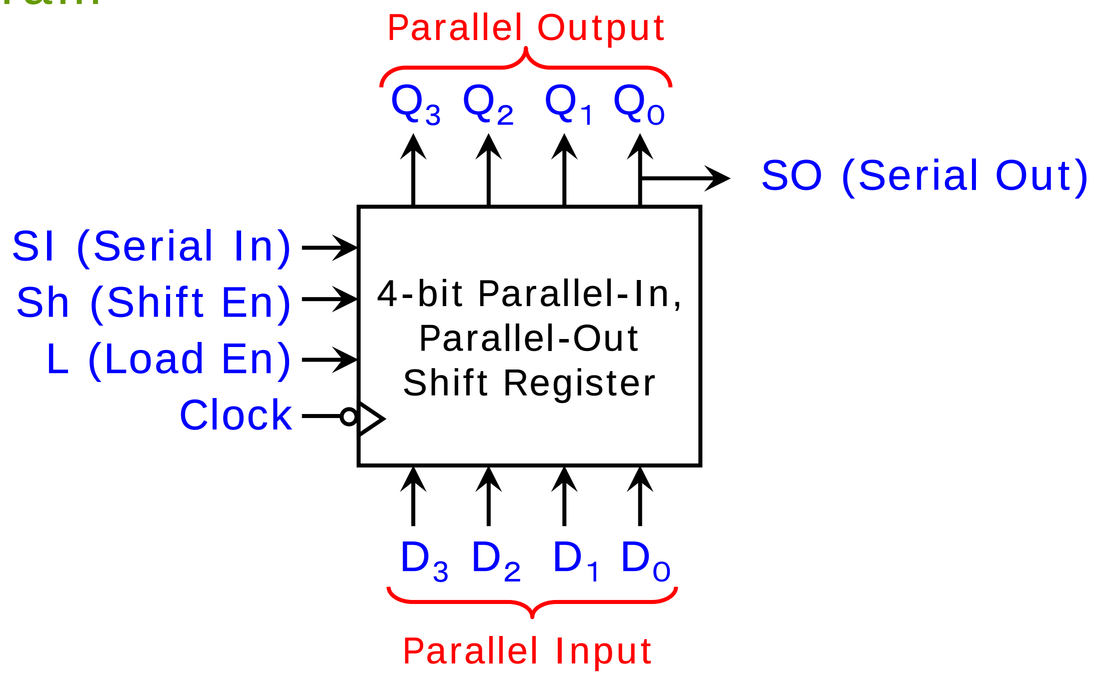


Timing diagram



Shift Registers
Parallel-In, Parallel-Out Shift Register

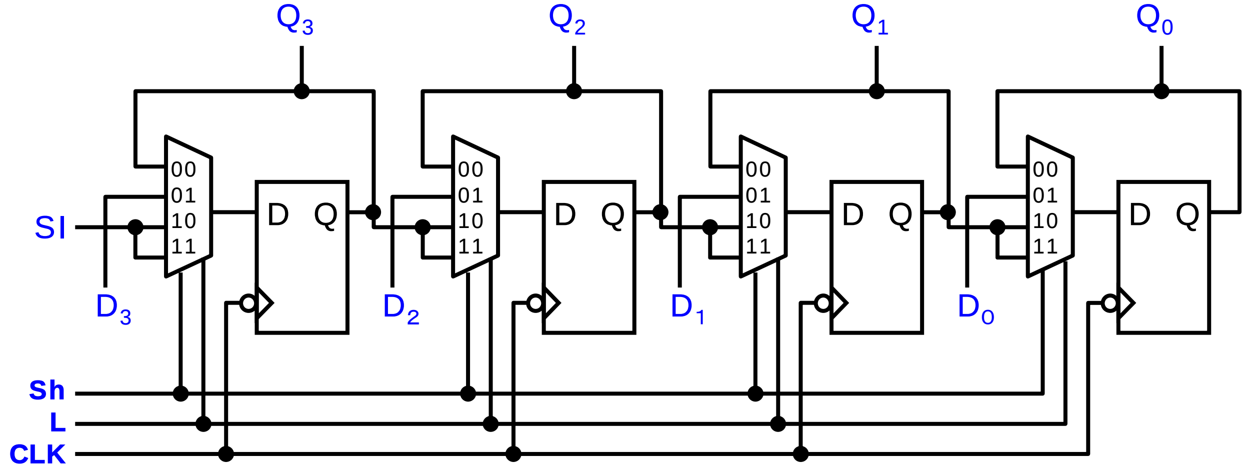
Block diagram



Application: conversion between parallel and serial data

Shift Registers
Parallel-In, Parallel-Out Shift Register

Logic diagram
(implementation using FFs and MUXes)



Shift register operation

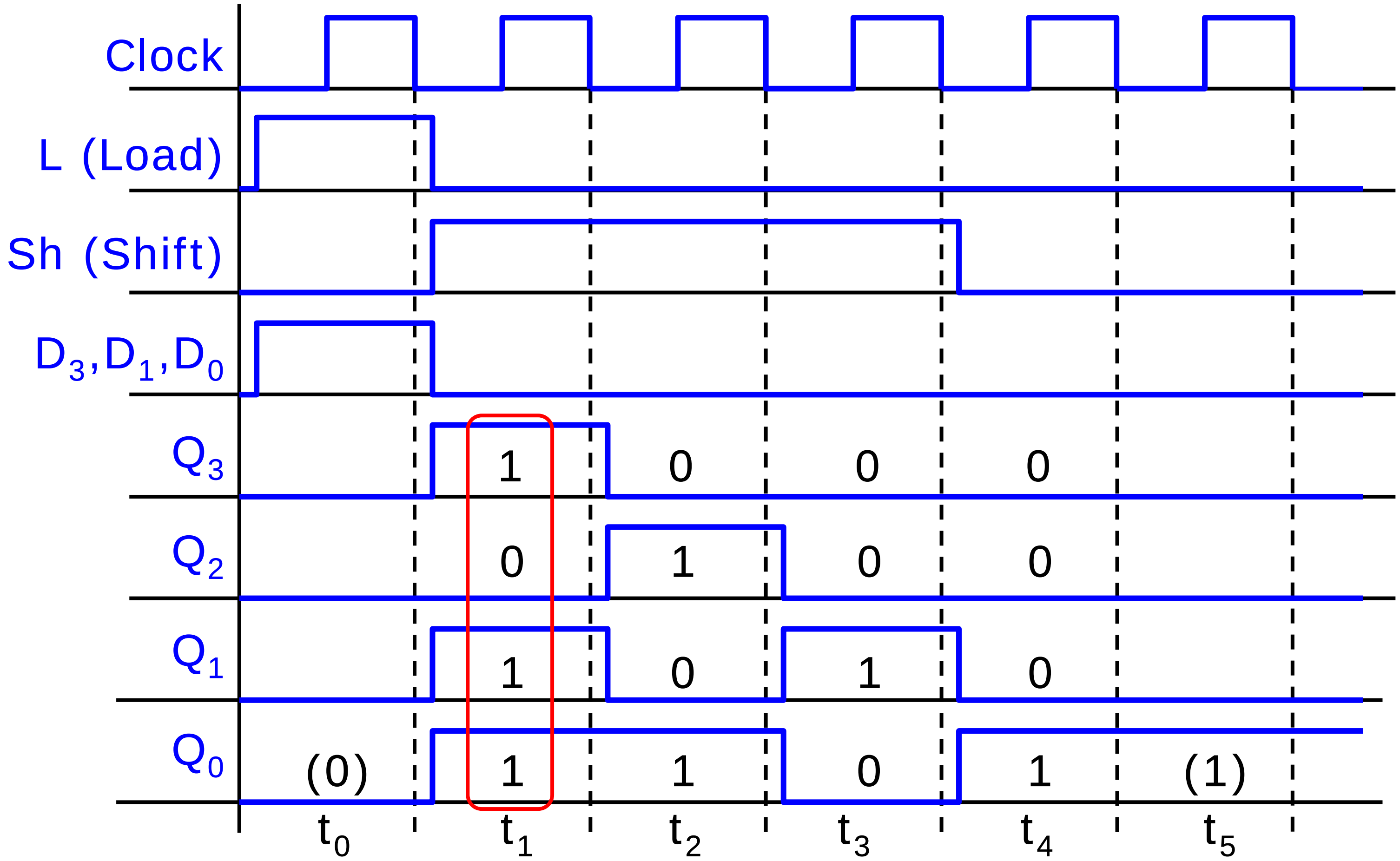
Inputs		Next State	Action
Sh (Shift)	L (Load)	$Q_3^+\ Q_2^+\ Q_1^+\ Q_0^+$	
0	0	$Q_3\ \ Q_2\ \ Q_1\ \ Q_0$	no change
0	1	$D_3\ \ D_2\ \ D_1\ \ D_0$	load
1	X	$SI\ \ Q_3\ \ Q_2\ \ Q_1$	right shift

$$\begin{cases} Q_3^+ = Sh' \cdot L' \cdot Q_3 + Sh' \cdot L \cdot D_3 + Sh \cdot SI \\ Q_2^+ = Sh' \cdot L' \cdot Q_2 + Sh' \cdot L \cdot D_2 + Sh \cdot Q_3 \\ Q_1^+ = Sh' \cdot L' \cdot Q_1 + Sh' \cdot L \cdot D_1 + Sh \cdot Q_2 \\ Q_0^+ = Sh' \cdot L' \cdot Q_0 + Sh' \cdot L \cdot D_0 + Sh \cdot Q_1 \end{cases}$$

Shift Registers
Parallel-In, Parallel-Out Shift Register

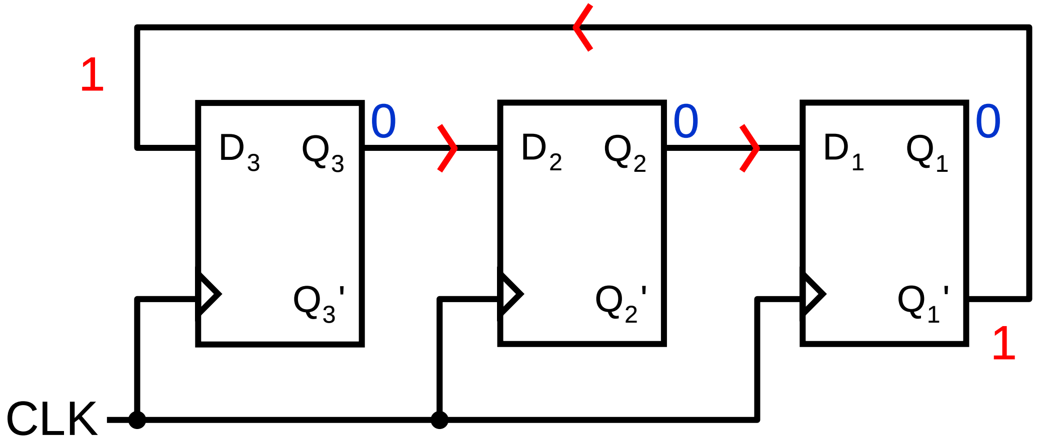
Timing diagram

(assume $D_3D_2D_1D_0$ = 1011 initially and 0000 afterwards)

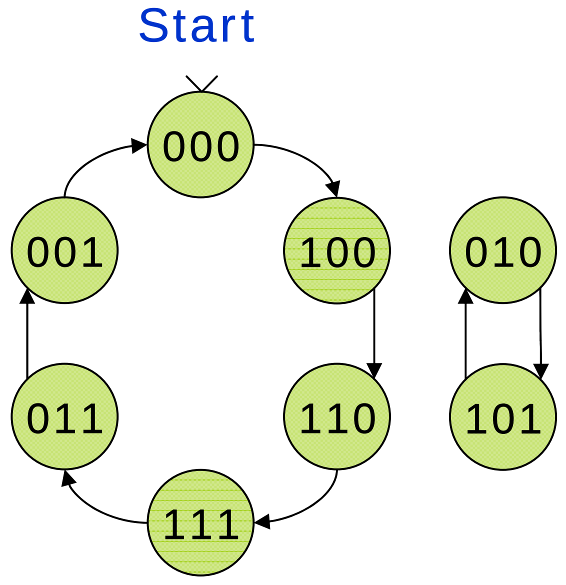


Shift Registers
Shift Register with Inverted Feedback

- A circuit that cycles through a fixed sequence of states is called a **counter**
- A shift register with inverted feedback is often called a **Johnson counter**



Flip-flop connections



State graph

Design of Binary Counters

- We focus on the synchronous counter, where flip-flops are synchronized by a common clock pulse

State table

Present State			Next State		
C	B	A	C^+	B^+	A^+
0	0	0	0	0	1
0	0	1	0	1	0
0	1	0	0	1	1
0	1	1	1	0	0
1	0	0	1	0	1
1	0	1	1	1	0
1	1	0	1	1	1
1	1	1	0	0	0

Design of Binary Counters
D Flip-Flop Implementation

State table

Present State			Next State			Flip-Flop Inputs		
C	B	A	C^+	B^+	A^+	D_C	D_B	D_A
0	0	0	0	0	1	0	0	1
0	0	1	0	1	0	0	1	0
0	1	0	0	1	1	0	1	1
0	1	1	1	0	0	1	0	0
1	0	0	1	0	1	1	0	1
1	0	1	1	1	0	1	1	0
1	1	0	1	1	1	1	1	1
1	1	1	0	0	0	0	0	0

FF inputs are next-state functions in terms of A,B,C

Design of Binary Counters
D Flip-Flop Implementation

- Karnaugh maps for D flip-flops

BA \ C	0	1
00	0	1
01	0	1
11	1	0
10	0	1

$D_C = C \oplus AB$

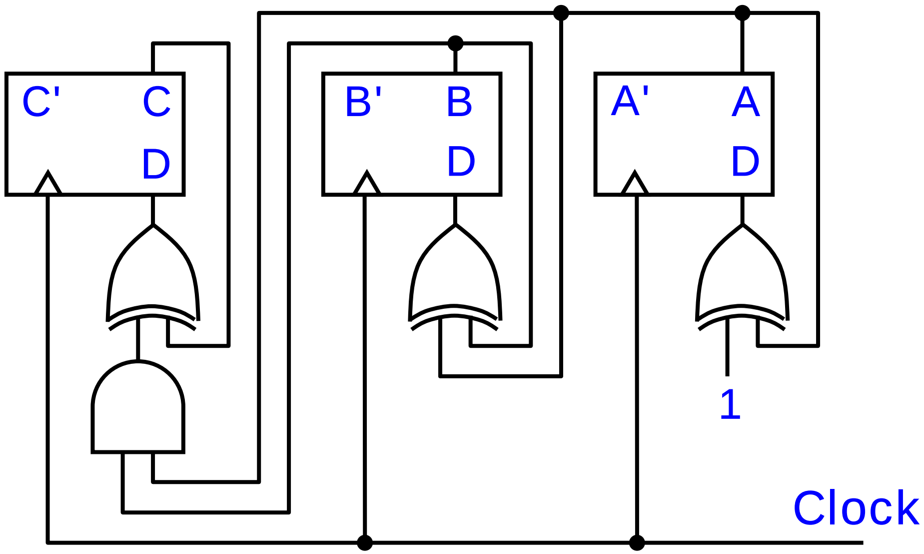
BA \ C	0	1
00	0	0
01	1	1
11	0	0
10	1	1

$D_B = A \oplus B$

BA \ C	0	1
00	1	1
01	0	0
11	0	0
10	1	1

$D_A = A'$

- Binary counter with D flip-flops



Design of Binary Counters
T Flip-Flop Implementation

- State table

Present State			Next State			Flip-Flop Inputs		
C	B	A	C^+	B^+	A^+	T_C	T_B	T_A
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

FF inputs are next-state functions in terms of A,B,C

Design of Binary Counters
T Flip-Flop Implementation

- Karnaugh maps for D flip-flops

BA \ C	0	1
00	0	0
01	0	0
11	1	1
10	0	0

$T_C = AB$

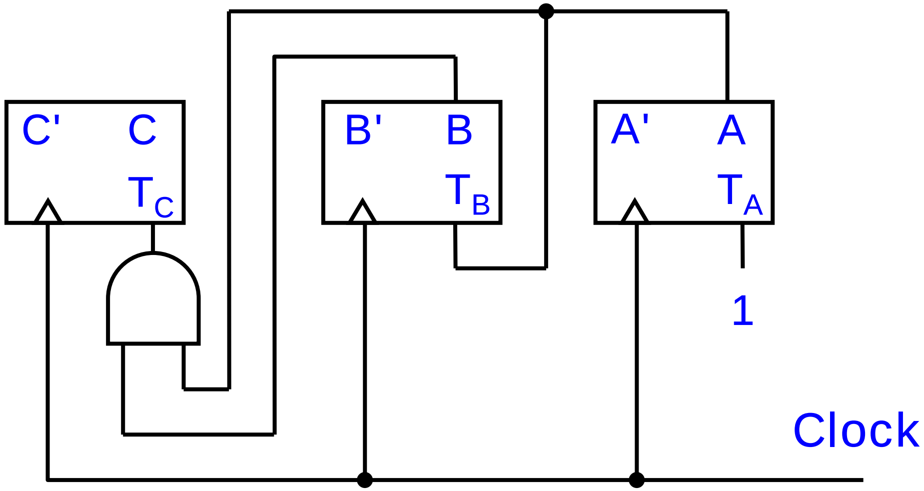
BA \ C	0	1
00	0	0
01	1	1
11	1	1
10	0	0

$T_B = A$

BA \ C	0	1
00	1	1
01	1	1
11	1	1
10	1	1

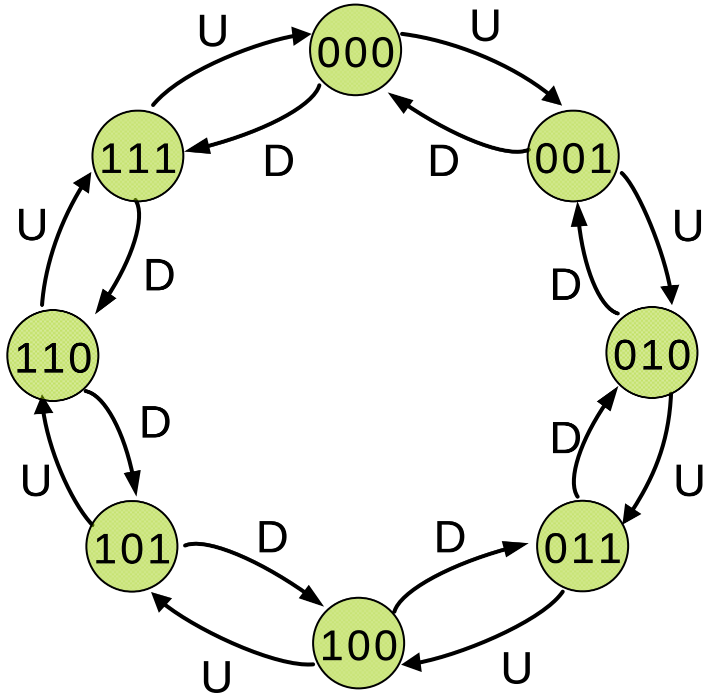
$T_A = 1$

- Binary counter with D flip-flops



Design of Binary Counters
Up-Down Counter

- State graph and table for up-down counter



C B A	C^+ B^+ A^+ U	C^+ B^+ A^+ D
0 0 0	0 0 1	1 1 1
0 0 1	0 1 0	0 0 0
0 1 0	0 1 1	0 0 1
0 1 1	1 0 0	0 1 0
1 0 0	1 0 1	0 1 1
1 0 1	1 1 0	1 0 0
1 1 0	1 1 1	1 0 1
1 1 1	0 0 0	1 1 0

U = 1, D = 0 count up
U = 0, D = 1 count down

Design of Binary Counters
Up-Down Counter

- The up-down counter can be implemented using D FFs and gates through the logic equations

  $D_A = A^+ = A \oplus (U+D)$

  $D_B = B^+ = B \oplus (UA+DA')$

  $D_C = C^+ = C \oplus (UBA+DB'A')$

  - When U=1, D=0, they reduce to binary up counter
  - When U=0, D=1, they reduce to

    $D_A = A^+ = A \oplus 1 = A'$

    $D_B = B^+ = B \oplus A'$

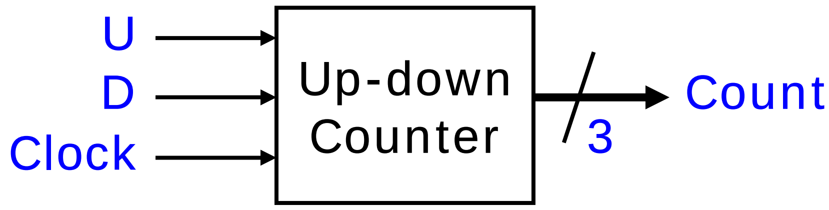
    $D_C = C^+ = C \oplus B'A'$

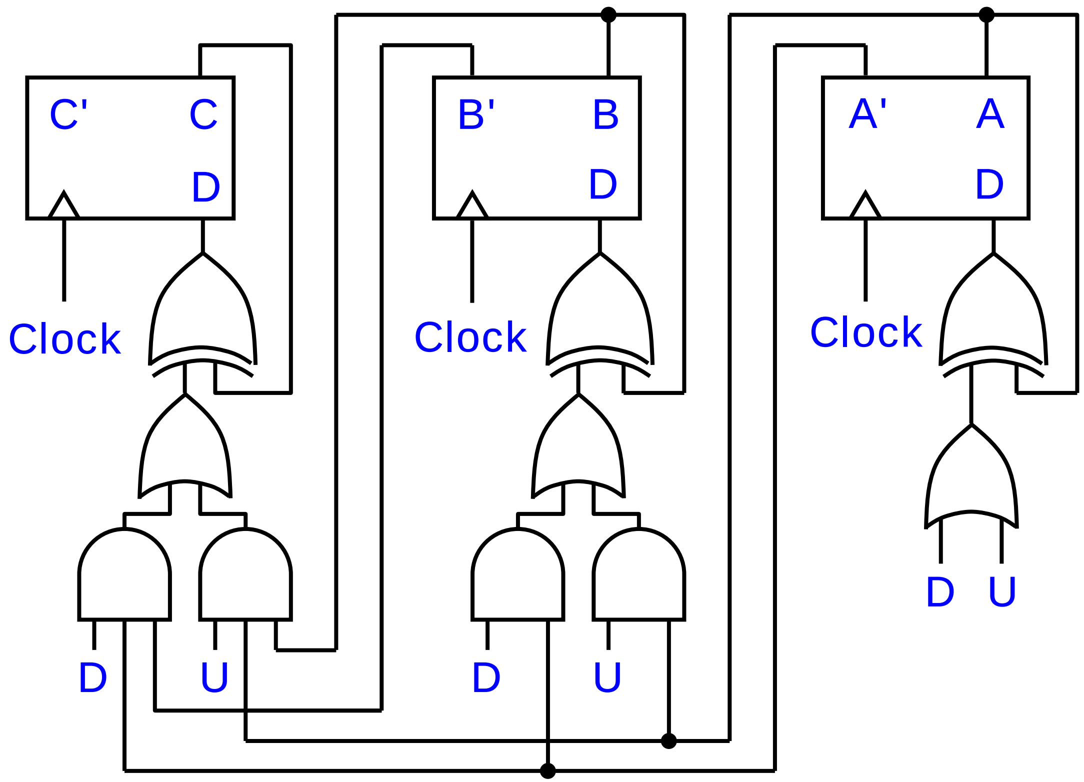
Design of Binary Counters
Up-Down Counter

$D_A = A^+ = A \oplus (U+D)$

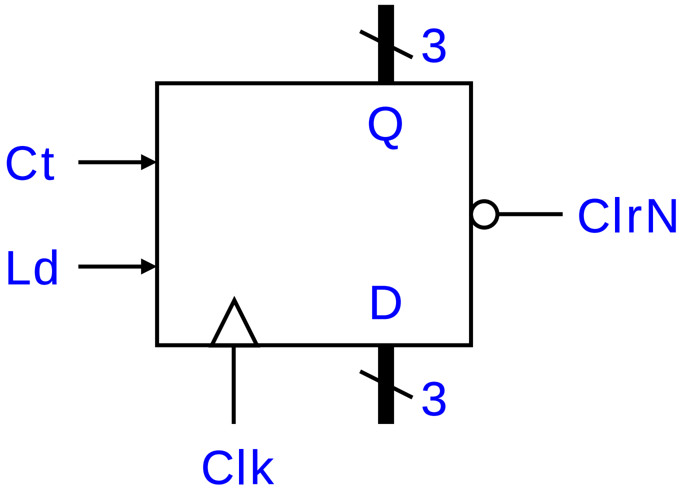
$D_B = B^+ = B \oplus (UA+DA')$

$D_C = C^+ = C \oplus (UBA+DB'A')$





Design of Binary Counters
Loadable Counter with Count Enable



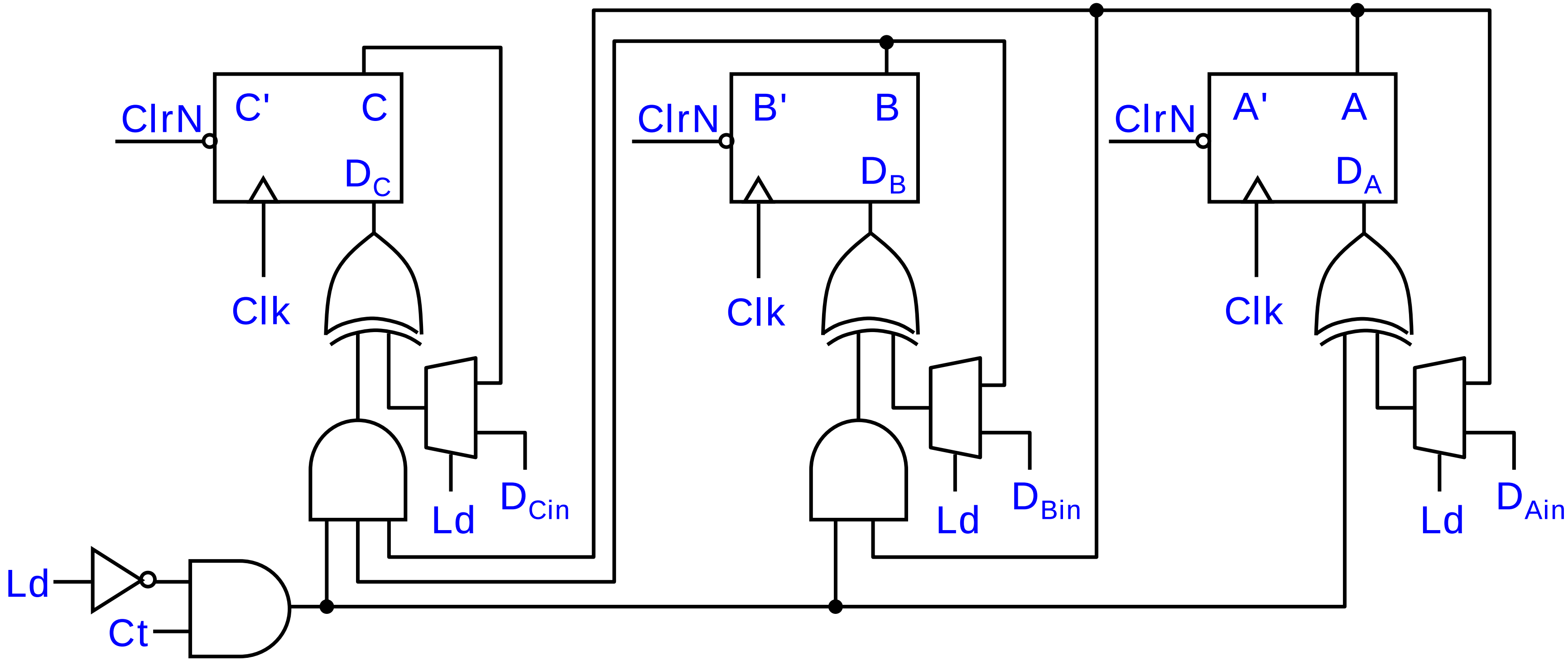
ClrN	Ld	Ct	C^+	B^+	A^+	
0	X	X	0	0	0	
1	1	X	D_C	D_B	D_A	(load)
1	0	0	C	B	A	(no change)
1	0	1	Present state + 1			

$$A^+ = D_A = (Ld' \cdot A + Ld \cdot D_{Ain}) \oplus Ld' \cdot Ct$$
$$B^+ = D_B = (Ld' \cdot B + Ld \cdot D_{Bin}) \oplus Ld' \cdot Ct \cdot A$$
$$C^+ = D_C = (Ld' \cdot C + Ld \cdot D_{Cin}) \oplus Ld' \cdot Ct \cdot B \cdot A$$

Note that ClrN does not appear in these equations.
Why not?

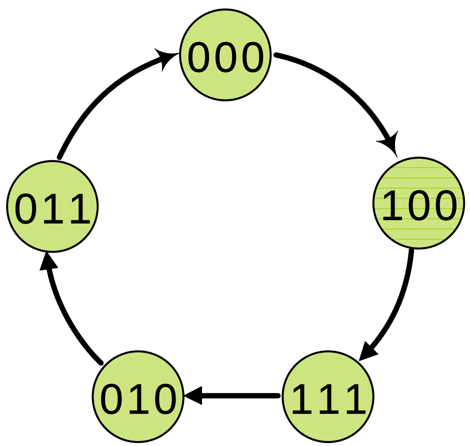
Design of Binary Counters
Loadable Counter with Count Enable

$$A^+ = D_A = (Ld' \cdot A + Ld \cdot D_{Ain}) \oplus Ld' \cdot Ct$$
$$B^+ = D_B = (Ld' \cdot B + Ld \cdot D_{Bin}) \oplus Ld' \cdot Ct \cdot A$$
$$C^+ = D_C = (Ld' \cdot C + Ld \cdot D_{Cin}) \oplus Ld' \cdot Ct \cdot B \cdot A$$



Counters for Other Sequences

State graph



State table

C	B	A	C^+	B^+	A^+
0	0	0	1	0	0
0	0	1	-	-	-
0	1	0	0	1	1
0	1	1	0	0	0
1	0	0	1	1	1
1	0	1	-	-	-
1	1	0	-	-	-
1	1	1	0	1	0

Counters for Other Sequences
Counter Design Using T Flip-Flops

State table / T FF input

C	B	A	C^+	B^+	A^+	T_C	T_B	T_A
0	0	0	1	0	0	1	0	0
0	0	1	-	-	-	-	-	-
0	1	0	0	1	1	0	0	1
0	1	1	0	0	0	0	1	1
1	0	0	1	1	1	0	1	1
1	0	1	-	-	-	-	-	-
1	1	0	-	-	-	-	-	-
1	1	1	0	1	0	1	0	1

QQ^+	T
0 0	0
0 1	1
1 0	1
1 1	0

$$T = Q^+ \oplus Q$$

Counters for Other Sequences
Counter Design Using T Flip-Flops

Next-state maps

C=0 half, C=1 half

BA \ C	0	1
00	1	1
01	X	X
11	0	0
10	0	X

C^+

B=0 half, B=1 half

BA \ C	0	1
00	0	1
01	X	X
11	0	1
10	1	X

B^+

A=0 half, A=1 half

BA \ C	0	1
00	0	1
01	X	X
11	0	0
10	1	X

A^+

QQ^+	T
0 0	0
0 1	1
1 0	1
1 1	0

$$T = Q^+ \oplus Q$$

Derivation of T inputs

C=0 half, C=1 half

BA \ C	0	1
00	1	0
01	X	X
11	0	1
10	0	X

T_C

$T_C = C'B' + CB$

B=0 half, B=1 half

BA \ C	0	1
00	0	1
01	X	X
11	1	0
10	0	X

T_B

$T_B = C'A + CB'$

A=0 half, A=1 half

BA \ C	0	1
00	0	1
01	X	X
11	1	1
10	1	X

T_A

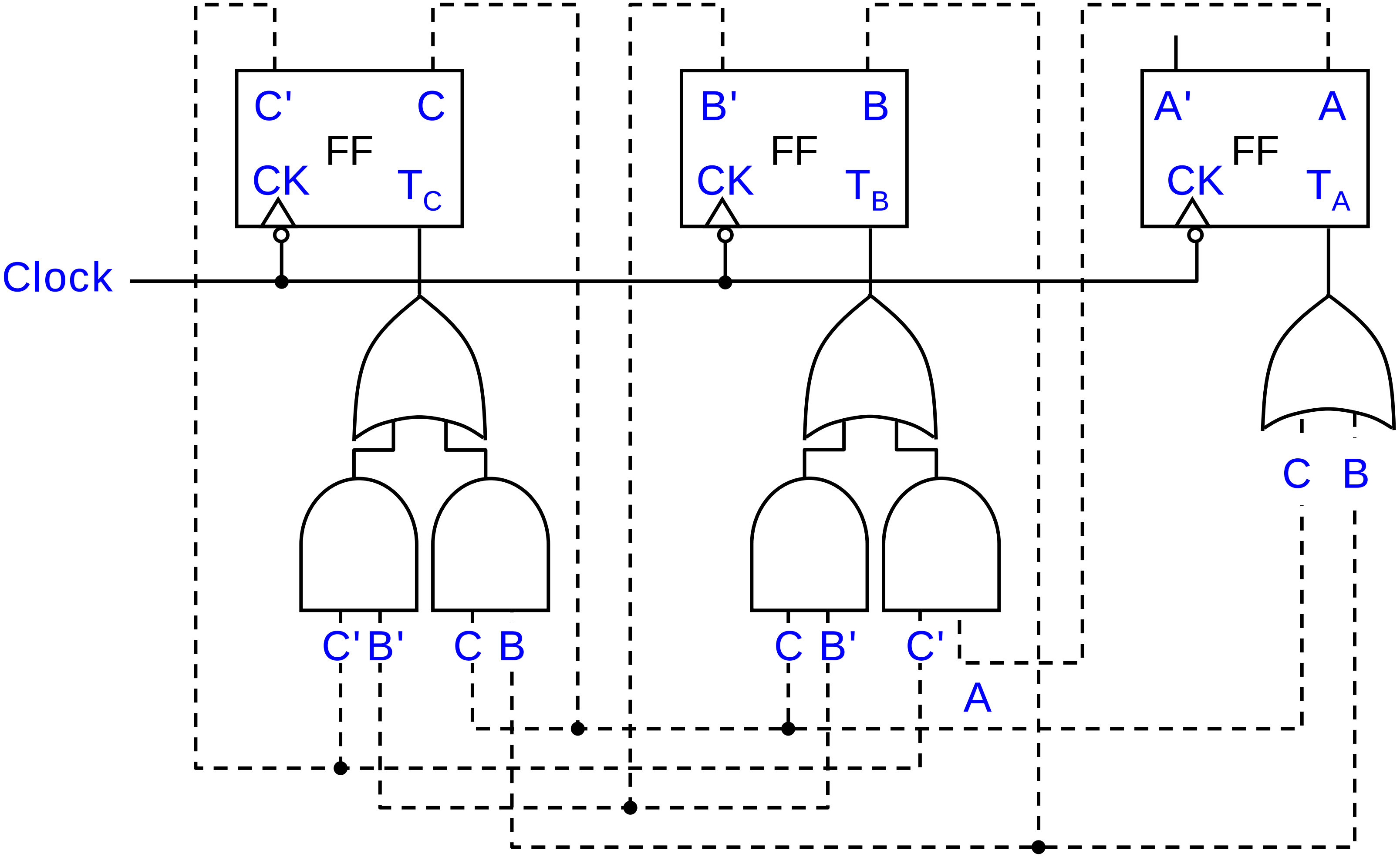
$T_A = C + B$

Counters for Other Sequences
Counter Design Using T Flip-Flops

- For T flip-flop implementation
  - If the Q^+ map has a don't care in some square, the T_Q map will have a don't care in the corresponding square
  - Divide the Q^+ map into two halves corresponding to Q=0 and Q=1, and transform each half of the map
    - Whenever Q=0, $T=Q^+$
      - Copy the half for which Q=0
    - Whenever Q=1, $T=(Q^+)'$
      - Complement the half for which Q=1

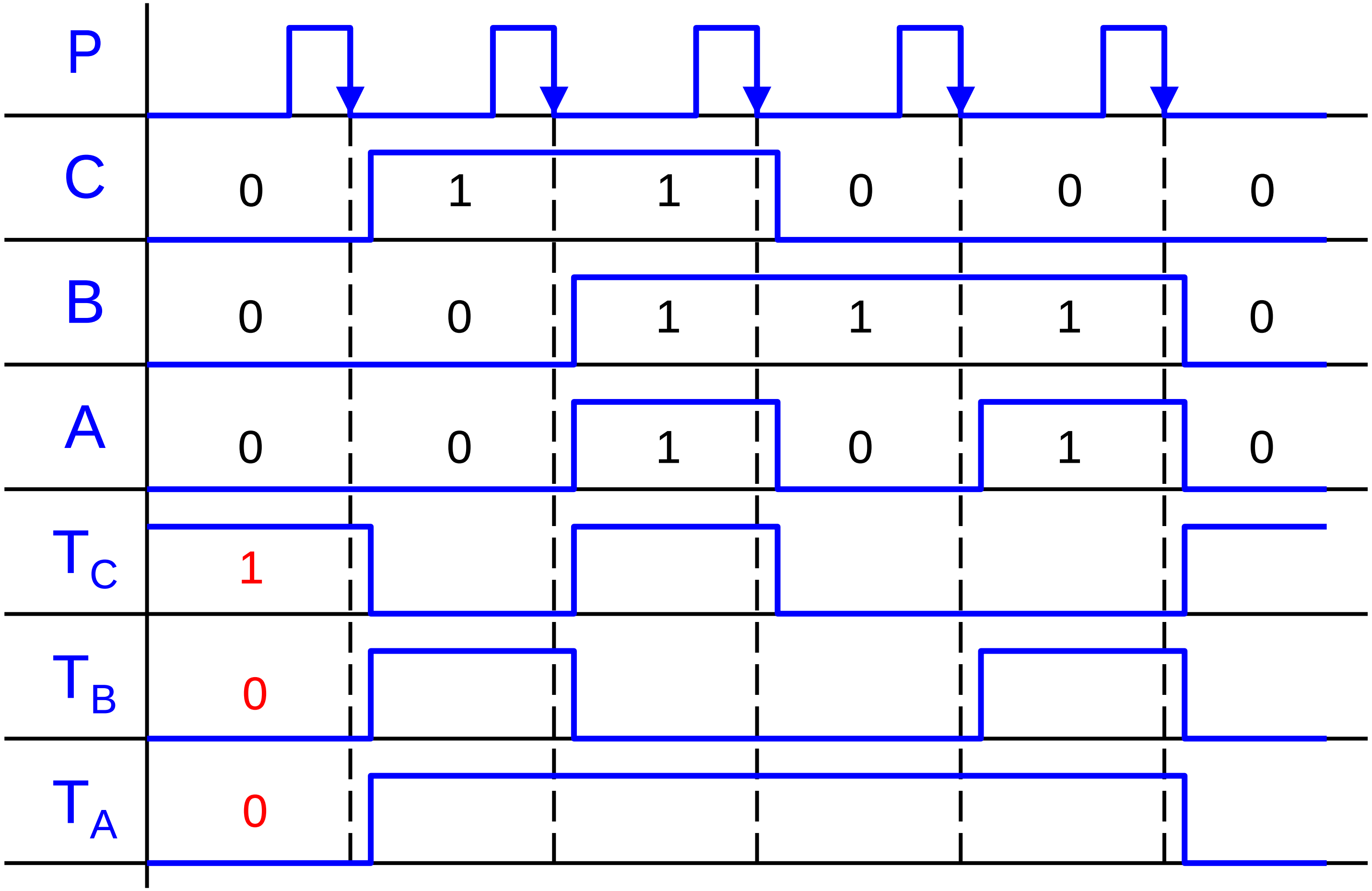
Counters for Other Sequences
Counter Design Using T Flip-Flops

- T flip-flop realization



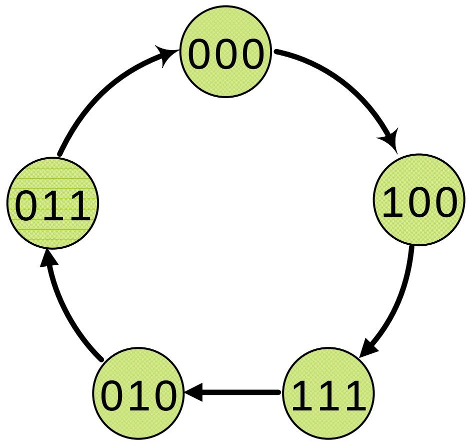
Counters for Other Sequences
Counter Design Using T Flip-Flops

- Timing diagram
  - Negative-edge triggered



Counters for Other Sequences
Counter Design Using T Flip-Flops

- In the process of completing the circuit design, the transitions of the original don’t care sates will specified
  - Don’t care states need to be checked to make sure they eventually lead into the main counting sequence unless a power-up reset is provided
    - When the power in a circuit is first turned on, the initial states of the flip-flops may be unpredictable (can be in an arbitrary state)



What are the transitions for states 001, 101, 110 by the realization in Slide 33?

T_C=C'B'+CB

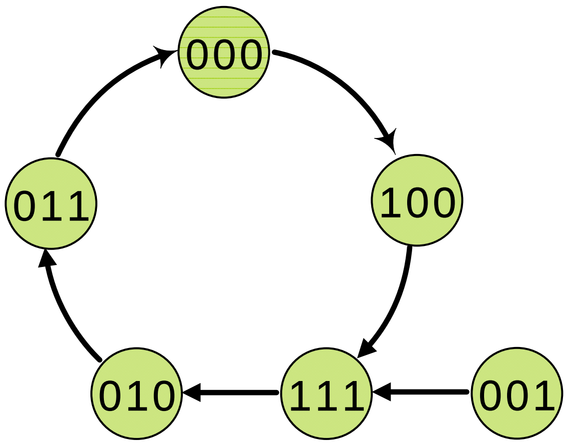
T_B=C'A+CB'

T_A=C+B

Counters for Other Sequences
Counter Design Using T Flip-Flops

- Example (cont'd)

  If FFs are powered up at (CBA)=001, then $T_C=T_B=1$ and $T_A=0$ leads to next state 111



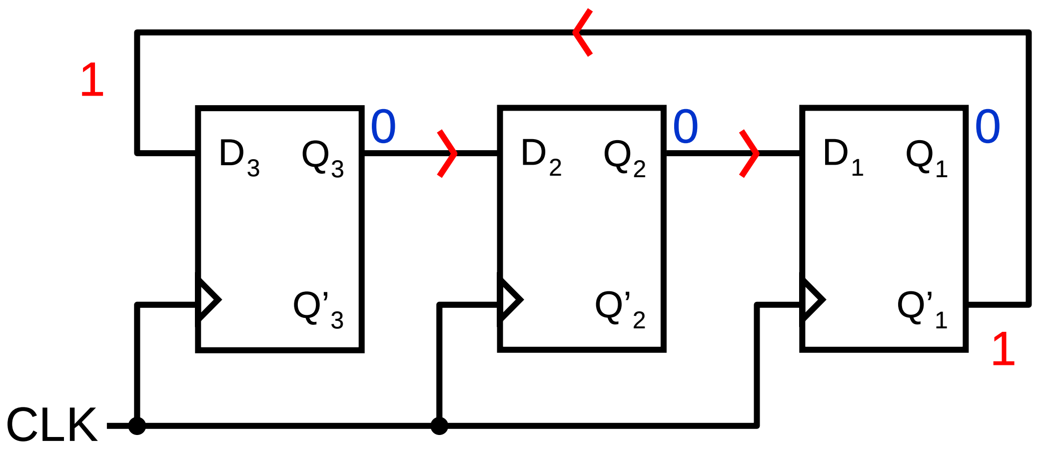
$T_C=C'B'+CB$

$T_B=C'A+CB'$

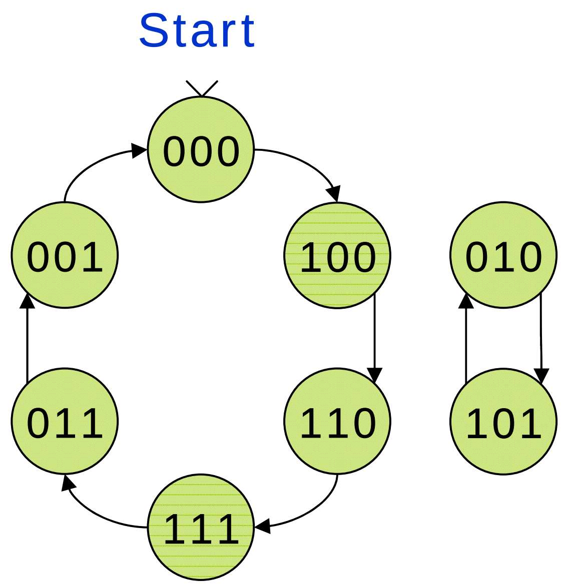
$T_A=C+B$

Counters for Other Sequences
Counter Design Using T Flip-Flops

- Example
  - Without power-up reset, the following Johnson counter may be incorrect when powered up at states 010 and 101



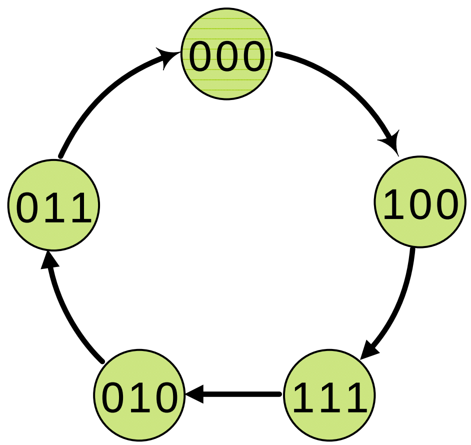
Flip-flop connections



State graph

Counters for Other Sequences
Counter Design Using D Flip-Flops

State graph



State table / D FF input

C	B	A	C^+	B^+	A^+	D_C	D_B	D_A
0	0	0	1	0	0	1	0	0
0	0	1	-	-	-	-	-	-
0	1	0	0	1	1	0	1	1
0	1	1	0	0	0	0	0	0
1	0	0	1	1	1	1	1	1
1	0	1	-	-	-	-	-	-
1	1	0	-	-	-	-	-	-
1	1	1	0	1	0	0	1	0

Not copy this!

D	Q^+
0	0
1	1

$Q^+ = D$

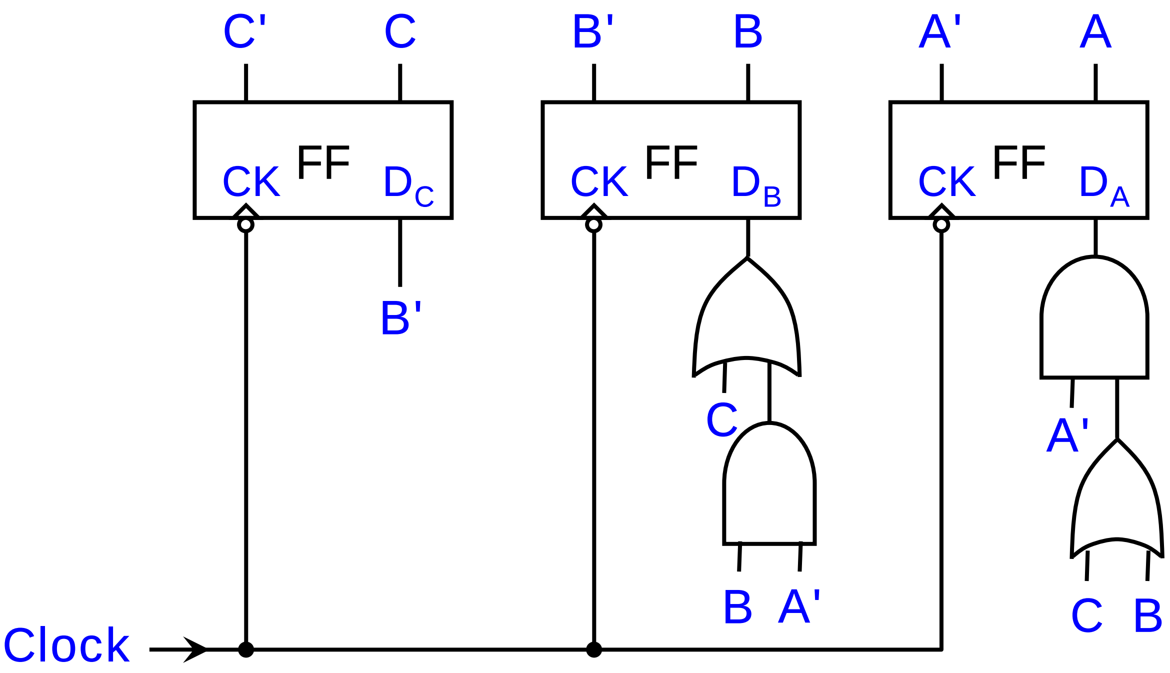
$D_C = C^+ = B'$

$D_B = B^+ = C+BA'$

$D_A = A^+ = CA'+BA' = A'(C+B)$

Counters for Other Sequences
Counter Design Using D Flip-Flops

$D_C = C^+ = B'$
$D_B = B^+ = C + BA'$
$D_A = A^+ = CA' + BA' = A'(C+B)$



Counter Design Using S-R and J-K FFs
Counter Design Using S-R Flip-Flops

- S-R flip-flop inputs

S	R	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	-

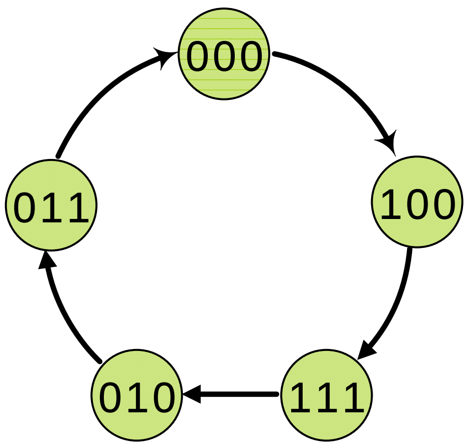
inputs not allowed

Q	Q^+	S	R
0	0	0	0
		0	1
0	1	1	0
1	0	0	1
1	1	0	0
		1	0

Q	Q^+	S	R
0	0	0	X
0	1	1	0
1	0	0	1
1	1	X	0

Counter Design Using S-R and J-K FFs
Counter Design Using S-R Flip-Flops

State graph



C	B	A	C^+	B^+	A^+	S_C	R_C	S_B	R_B	S_A	R_A
						C		B		A	
0	0	0	1	0	0	1	0	0	X	0	X
0	0	1	-	-	-	X	X	X	X	X	X
0	1	0	0	1	1	0	X	X	0	1	0
0	1	1	0	0	0	0	X	0	1	0	1
1	0	0	1	1	1	X	0	1	0	1	0
1	0	1	-	-	-	X	X	X	X	X	X
1	1	0	-	-	-	X	X	X	X	X	X
1	1	1	0	1	0	0	1	X	0	0	1

Q	Q^+	S	R
0	0	0	X
0	1	1	0
1	0	0	1
1	1	X	0

Counter Design Using S-R and J-K FFs

Counter Design Using S-R Flip-Flops

Next-state maps

C^+

BA \ C	0	1
00	1	1
01	X	X
11	0	0
10	0	X

B^+

BA \ C	0	1	
00	0	1	B=0 half
01	X	X	B=0 half
11	0	1	B=1 half
10	1	X	B=1 half

A^+

BA \ C	0	1	
00	0	1	A=0 half
01	X	X	A=1 half
11	0	0	A=1 half
10	1	X	A=0 half

R_C

BA \ C	0	1
00		
01	X	X
11	X	1
10	X	X

S_C

BA \ C	0	1
00	1	X
01	X	X
11		
10		X

$R_C = A$ $\quad S_C = B'$

R_B

BA \ C	0	1
00	X	
01	X	X
11	1	
10		X

S_B

BA \ C	0	1
00		1
01	X	X
11		X
10	X	X

$R_B = C'A$ $\quad S_B = C$

R_A

BA \ C	0	1
00	X	
01	X	X
11	1	1
10		X

S_A

BA \ C	0	1
00		1
01	X	X
11		
10	1	X

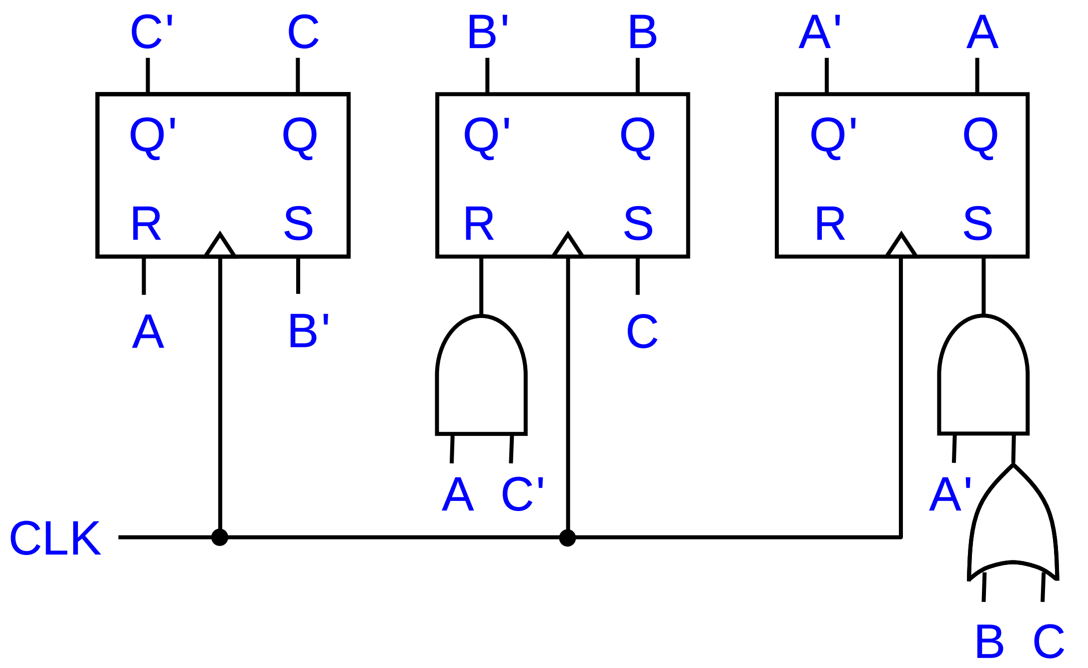
$R_A = A$

$S_A = CA' + BA'$
$= A'(C+B)$

S-R flip-flop equations

Counter Design Using S-R and J-K FFs
Counter Design Using S-R Flip-Flops

- S-R flip-flop realization



(feedback lines omitted)

Counter Design Using S-R and J-K FFs
Counter Design Using J-K Flip-Flops

- J-K flip-flop inputs

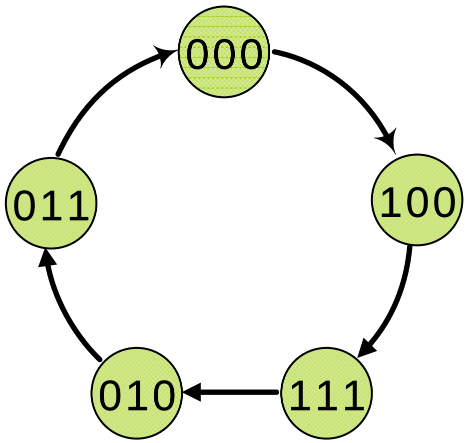
J	K	Q	Q^+
0	0	0	0
0	0	1	1
0	1	0	0
0	1	1	0
1	0	0	1
1	0	1	1
1	1	0	1
1	1	1	0

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

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

Counter Design Using S-R and J-K FFs
Counter Design Using J-K Flip-Flops

State graph



						C		B		A	
C	B	A	C^+	B^+	A^+	J_C	K_C	J_B	K_B	J_A	K_A
0	0	0	1	0	0	1	X	0	X	0	X
0	0	1	-	-	-	X	X	X	X	X	X
0	1	0	0	1	1	0	X	X	0	1	X
0	1	1	0	0	0	0	X	X	1	X	1
1	0	0	1	1	1	X	0	1	X	1	X
1	0	1	-	-	-	X	X	X	X	X	X
1	1	0	-	-	-	X	X	X	X	X	X
1	1	1	0	1	0	X	1	X	0	X	1

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

Counter Design Using S-R and J-K FFs
Counter Design Using J-K Flip-Flops

Next-state maps

BA \ C	0	1
00	1	1
01	X	X
11	0	0
10	0	X

C^+

BA \ C	0	1	
00	0	1	B=0 half
01	X	X	
11	0	1	B=1 half
10	1	X	

B^+

BA \ C	0	1	
00	0	1	A=0 half
01	X	X	A=1 half
11	0	0	
10	1	X	

A^+

BA \ C	0	1
00	1	X
01	X	X
11		X
10		X

J_C

BA \ C	0	1
00	X	
01	X	X
11	X	1
10	X	X

K_C

$J_C = B'$ $K_C = A$

BA \ C	0	1
00		1
01	X	X
11	X	X
10	X	X

J_B

BA \ C	0	1
00	X	X
01	X	X
11	1	
10		X

K_B

$J_B = C$ $K_B = C'A$

BA \ C	0	1
00		1
01	X	X
11	X	X
10	1	X

J_A

BA \ C	0	1
00	X	X
01	X	X
11	1	1
10	X	X

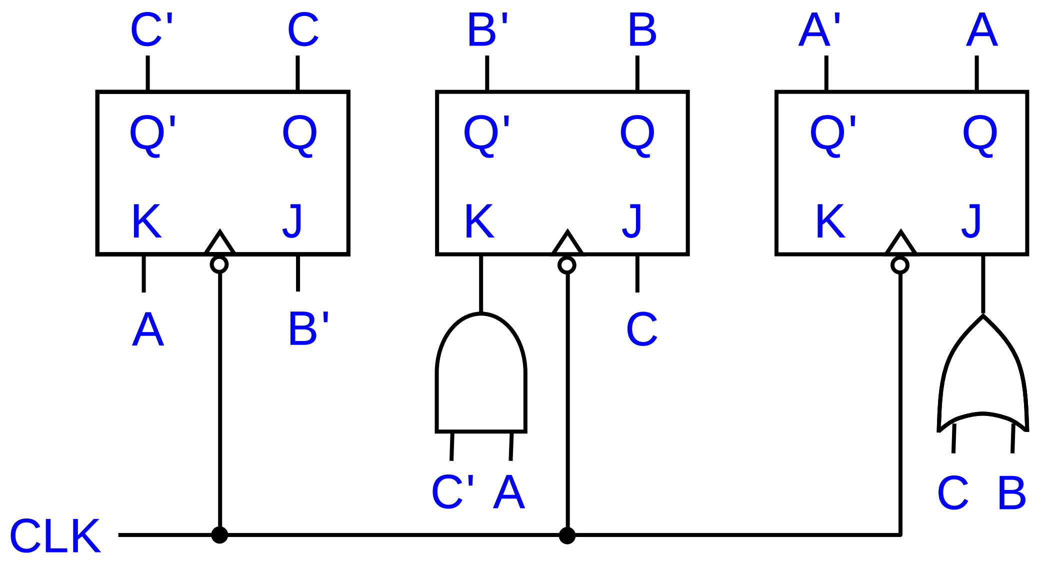
K_A

$J_A = C + B$ $K_A = 1$

J-K flip-flop equations

Counter Design Using S-R and J-K FFs
Counter Design Using J-K Flip-Flops

- J-K flip-flop realization



(feedback lines omitted)

Derivation of Flip-Flop Input Equations

- Determination of flip-flop input equations from next-state equations using Karnaugh maps

Type of Flip-Flop	Input	Q = 0		Q = 1		Rules for Forming Input Map From Next-State Map*	
		$Q^+=0$	$Q^+=1$	$Q^+=0$	$Q^+=1$	Q=0 Half of Map	Q=1 Half of Map
Delay	D	0	1	0	1	No change	No change
Trigger	T	0	1	1	0	No change	Complement
Set-Reset	S	0	1	0	X	No change	Replace 1's with X's**
	R	X	0	1	0	Replace 0's with X's**	Complement
J-K	J	0	1	X	X	No change	Fill in with X's
	K	X	X	1	0	Fill in with X's	Complement

*Always copy X's from the next-state map onto the input maps first
**Fill in the remaining squares with 0's

Derivation of Flip-Flop Input Equations
Flip-Flop Input Tables

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

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

Q	Q^+	S	R
0	0	0	X
0	1	1	0
1	0	0	1
1	1	X	0

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

Realization using J-K flip flops usually yields lower cost implementation

Derivation of Flip-Flop Input Equations Example

BA \ Q	0	1
00	0	1
01	1	0
11	0	0
10	1	X

Q^+

Next-state map

BA \ Q	0	1
00	0	1
01	1	0
11	0	0
10	1	X

D=Q'A'B+QB'+AB'

D input map

BA \ Q	0	1
00	0	0
01	1	1
11	0	1
10	1	X

T=A'B+AB'+QB

T input map

BA \ Q	0	1
00	0	X
01	1	0
11	0	0
10	1	X

S=AB'+Q'A'B

BA \ Q	0	1
00	X	0
01	0	1
11	X	1
10	0	X

R=QB

S-R input maps

BA \ Q	0	1
00	0	X
01	1	X
11	0	X
10	1	X

J=A'B+AB'

BA \ Q	0	1
00	X	0
01	X	1
11	X	1
10	X	X

K=B

J-K input maps

Derivation of Flip-Flop Input Equations Example (1/3)

- Derivation of Q_1 (T flip-flop) input equation using 4-variable maps

BC \ Q_1A	00	01	11	10
00	0	1	0	1
01	X	1	1	0
11	1	X	X	1
10	0	0	0	X

$Q_1=0$ half | $Q_1=1$ half

Q_1^+

BC \ Q_1A	00	01	11	10
00	0	1	1	0
01	X	1	0	1
11	1	X	X	0
10	0	0	1	X

T_1

Derivation of Flip-Flop Input Equations Example (2/3)

- Derivation of Q_2 (S-R flip-flop) input equations using 4-variable maps

$Q_2 = 0$ half

$Q_2 = 1$ half

CQ_2 \ AB	00	01	11	10
00	1	X	1	0
01	0	0	X	1
11	1	0	X	1
10	X	0	0	1

Q_2^+

CQ_2 \ AB	00	01	11	10
00	0	X	0	X
01	1	1	X	0
11	0	1	X	0
10	X	X	X	0

R_2

CQ_2 \ AB	00	01	11	10
00	1	X	1	0
01	0	0	X	X
11	X	0	X	X
10	X	0	0	1

S_2

Derivation of Flip-Flop Input Equations Example (3/3)

- Derivation of Q_3 (J-K flip-flop) input equations using 4-variable maps

	Q_3C \ AB	00	01	11	10
Q_3=0 half	00	0	0	1	X
	01	0	1	X	1
Q_3=1 half	11	X	X	0	0
	10	1	1	1	0

Q_3^+

Q_3C \ AB	00	01	11	10
00	0	0	1	X
01	0	1	X	1
11	X	X	X	X
10	X	X	X	X

$J_3 = A+BC$

Q_3C \ AB	00	01	11	10
00	X	X	X	X
01	X	X	X	X
11	X	X	1	1
10	0	0	0	1

$K_3 = C+AB'$

Derivation of Flip-Flop Input Equations Summary

- To implement a counter of N states, we need at least $\log_2 N$ flip-flops
- Derive input equations depending on the target implementation using D, T, SR, or JK flip-flops
- Pay attention to don't care states for power-up conditions. Sometimes reset may be needed