Sequential Circuit

State Machines and Registers

Ver. 3

Professor: Yang Peng

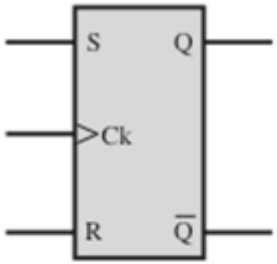
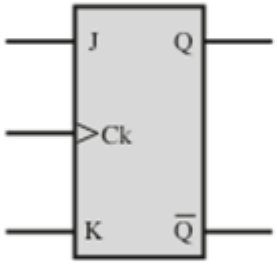
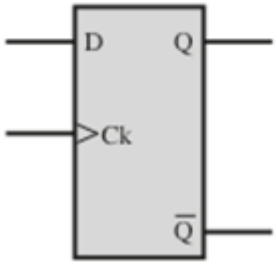
Topics

- Characteristic and Excitation Tables
- Designing circuits to function as Finite State Machines
  - Example 1: Two-bit binary counter
  - Example 2: Three-bit sequential circuit
- Looking at Typical Sequential Circuits
  - Ripple counter
  - Shift register
  - Storage register

Null Ch3.7

Burger Ch5

Characteristic Table for FF

Name	Graphical Symbol	Characteristic Table
S-R		S R | Q_{n+1} 0 0 | Q_n 0 1 | 0 1 0 | 1 1 1 | –
J-K		J K | Q_{n+1} 0 0 | Q_n 0 1 | 0 1 0 | 1 1 1 | $\overline{Q}_n$
D		D | Q_{n+1} 0 | 0 1 | 1

State, Characteristic, and Excitation Tables

- **State Transition Table:** Give the next state information based on the input and current state (like truth table)
- **Characteristic Table:** Give the state transition information based on inputs (simplified state transition table)
- **Excitation Table** (inverted characteristic table)
  - **Input**: current state and next state
  - **Output**: S R or J K or D – the control bit(s) of FF
  - Necessity
    - To understand how to set the control bits to generate a needed state transition
    - If you know the excitation table, you know how the sequential circuit can be built

Excitation Table for Flip-Flops

SR flip-flop

$Q(t)$	$Q(t+1)$	S	R
0	0	0	×
0	1	1	0
1	0	0	1
1	1	×	0

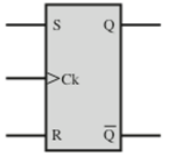
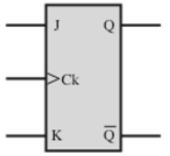
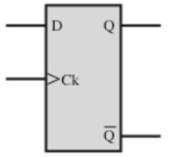
D flip-flop

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

JK flip-flop

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

"X" is the "don't care" state

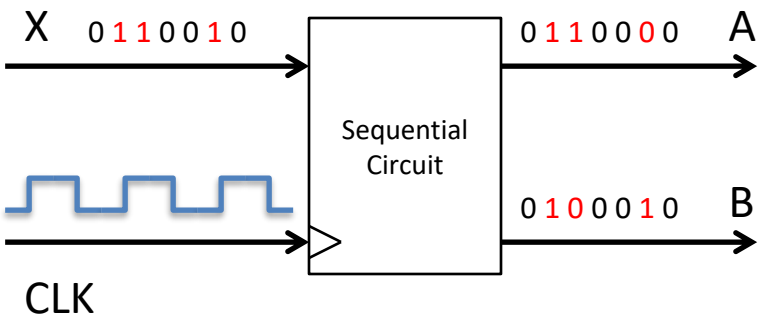
Name	Graphical Symbol	Characteristic Table
S-R		S R Q_{n+1} 0 0 Q_n 0 1 0 1 0 1 1 1 –
J-K		J K Q_{n+1} 0 0 Q_n 0 1 0 1 0 1 1 1 $\overline{Q_n}$
D		D Q_{n+1} 0 0 1 1

Design of Finite State Machines

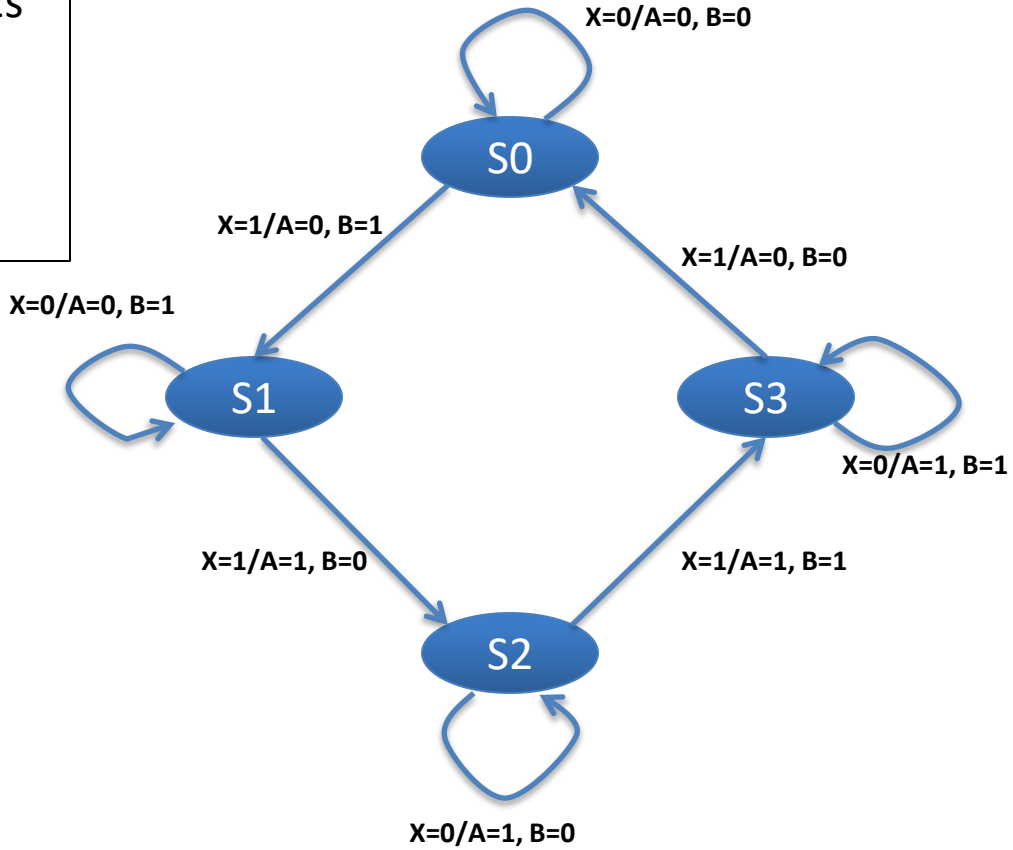
- 5 Steps
  1. Draw a state transition diagram
  2. Draw a state transition table
  3. *Simplify redundant states (not always necessary)*
  4. Allocate flip-flop output signals to the states
  5. Draw K-maps to design combinational logics that generates state transitions and outputs

Example 1: 2-Bit Binary Counter

- The input is a binary stream
- Count the number of encountered 1s in the input signal, from 0 to 3, (i.e., 00 to 11) and thereafter reset to 0, (i.e., 00)



Step 1: State Diagram



Step 2: State Transition Table

	Input X = 0			Input X = 1		
Current State	Next State	Output A	Output B	Next State	Output A	Output B
S0	S0	0	0	S1	0	1
S1	S1	0	1	S2	1	0
S2	S2	1	0	S3	1	1
S3	S3	1	1	S0	0	0

Step 3: Simplification Skipped

No States to Merge in this example

Step 4: State Allocation

State allocation to two JK FFs: A and B

State	AB
S0	00
S1	01
S2	10
S3	11

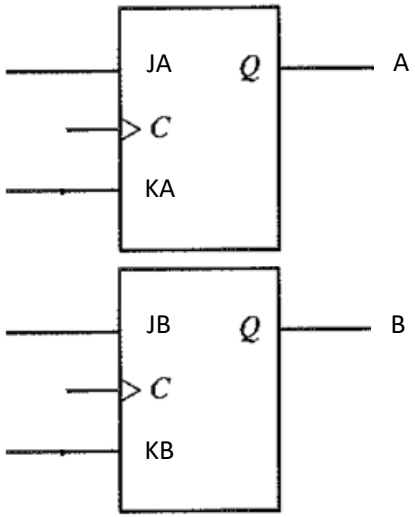
JK FF's Excitation Table

Qt	Qt+1	J	K
0	0	0	*
0	1	1	*
1	0	*	1
1	1	*	0

State Transition Table with JK FFs: A and B

	Input X = 0		Input X = 1		Input X=0		Input X=1		Input X=0		Input X=1	
Current State	Next State	Output	Next State	Output	Inputs to JK FF-A				Inputs to JK FF-B			
A_tB_t	$A_{t+1}B_{t+1}$	$A_{t+1}B_{t+1}$	A_tB_t	$A_{t+1}B_{t+1}$	JA	KA	JA	KA	JB	KB	JB	KB
00	00	00	01	01	0	*	0	*	0	*	1	*
01	01	01	10	10	0	*	1	*	*	0	*	1
10	10	10	11	11	*	0	*	0	0	*	1	*
11	11	11	00	00	*	0	*	1	*	0	*	1

Step 5: Karnaugh Maps



State Transition Table with JK FF A and B

	Input X = 0		Input X = 1		Input X=0		Input X=1		Input X=0		Input X=1	
Current State	Next State	Output	Next State	Output	Inputs to JK FF-A				Inputs to JK FF-B			
A_tB_t	$A_{t+1}B_{t+1}$	$A_{t+1}B_{t+1}$	A_tB_t	$A_{t+1}B_{t+1}$	JA	KA	JA	KA	JB	KB	JB	KB
00	00	00	01	01	0	*	0	*	0	*	1	*
01	01	01	10	10	0	*	1	*	*	0	*	1
10	10	10	11	11	*	0	*	0	0	*	1	*
11	11	11	00	00	*	0	*	1	*	0	*	1

$J_A = BX$

AB / X	00	01	11	10
0	0	0	*	*
1	0	1	*	*

$J_B = X$

AB / X	00	01	11	10
0	0	*	*	0
1	1	*	*	1

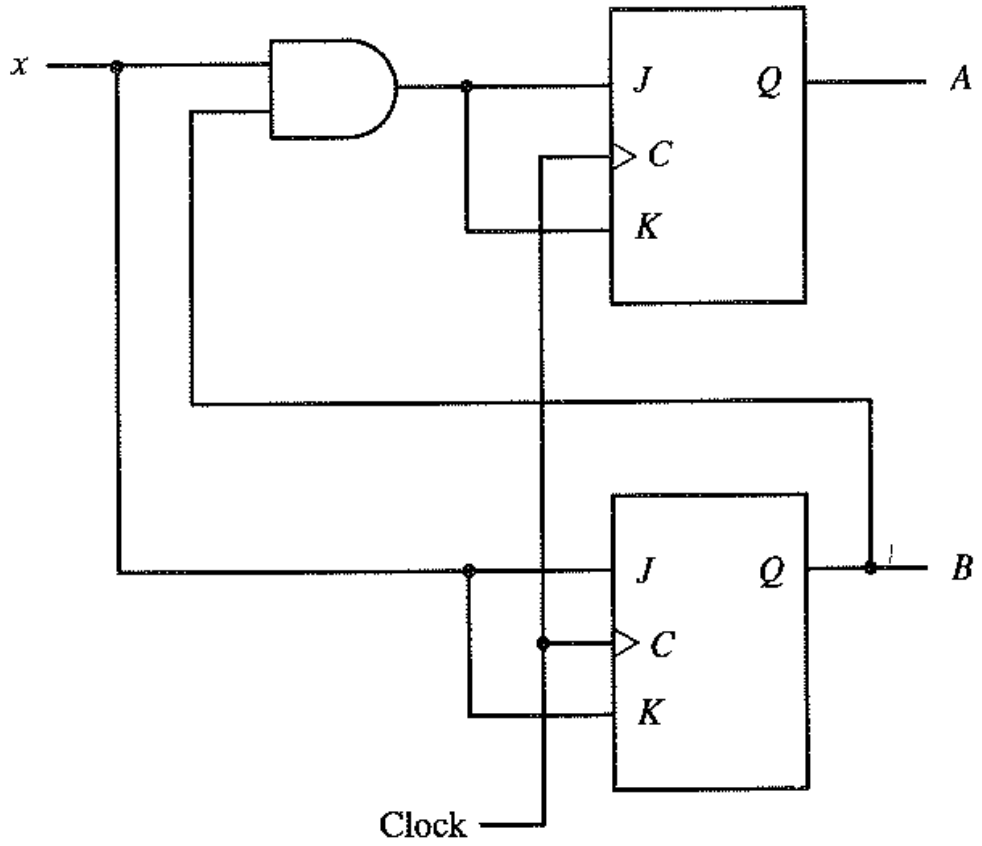
$K_A = BX$

AB / X	00	01	11	10
0	*	*	0	0
1	*	*	1	0

$K_B = X$

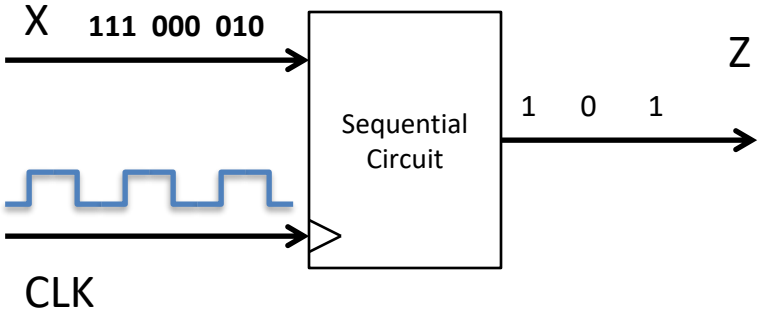
AB / X	00	01	11	10
0	*	0	0	*
1	*	1	1	*

Logic of 2-bit Binary Counter

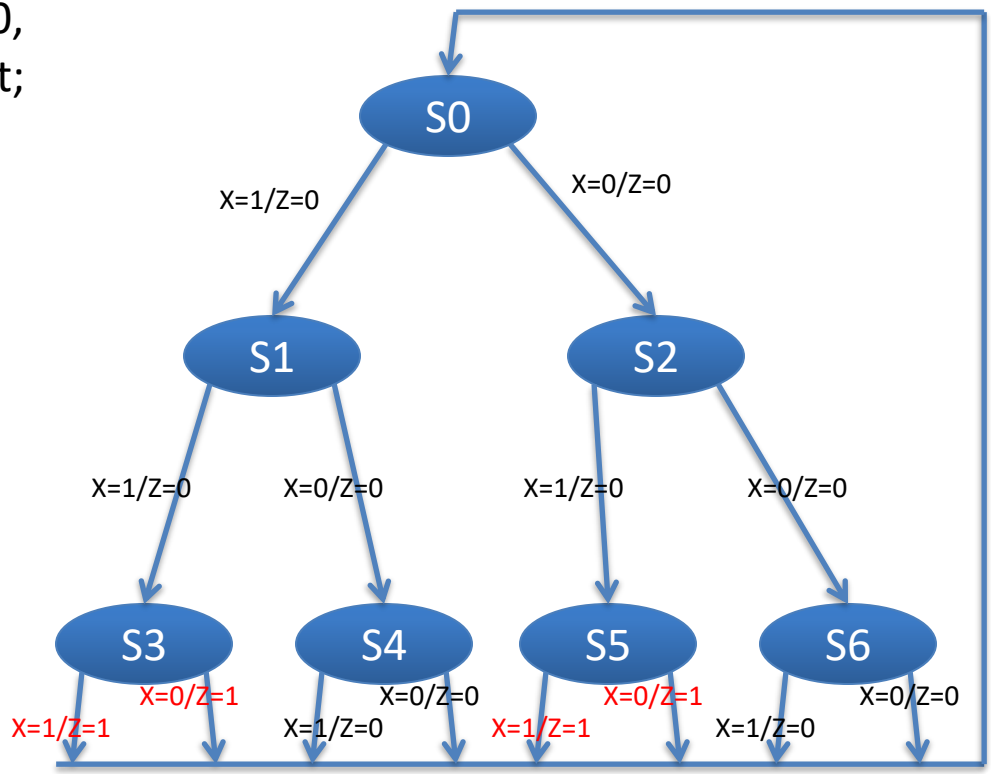


Example 2: 3-bit Sequential Circuit

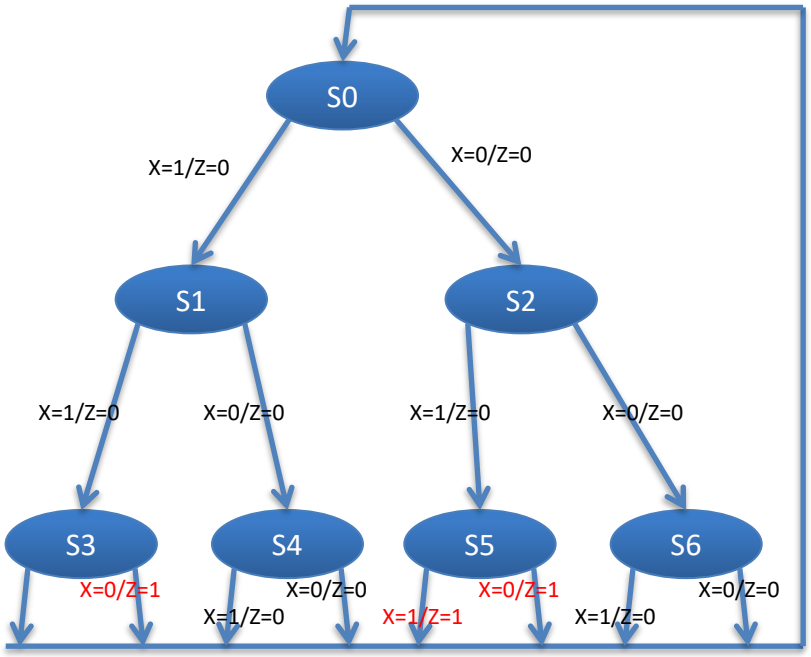
- An input is a sequence of 3-bit binary numbers.
- If it is sequence 111, 110, 011, or 010, the circuit should emit 1 as its output; otherwise, the circuit emits 0.



Step 1: State Diagram



Step 2: State Transition Diagram



	Input X = 0		Input X = 1	
Current State	Next State	Output Z	Next State	Output Z
S0	S2	0	S1	0
S1	S4	0	S3	0
S2	S6	0	S5	0
S3	S0	1	S0	1
S4	S0	0	S0	0
S5	S0	1	S0	1
S6	S0	0	S0	0

Step 3: Simplification

	Input X = 0		Input X = 1	
Current State	Next State	Output Z	Next State	Output Z
S0	S2	0	S1	0
S1	S46	0	S35	0
S2	S46	0	S35	0
S35	S0	1	S0	1
S46	S0	0	S0	0

Coldwell's merging rule:
States that transit to the same state and produces the same output can be merged

Step 3: Simplification

	Input X = 0		Input X = 1	
Current State	Next State	Output Z	Next State	Output Z
S0	S12	0	S12	0
S12	S46	0	S35	0
S35	S0	1	S0	1
S46	S0	0	S0	0

Four states left after merging: S0, S12 (new S1), S35 (new S2), S46 (new S3)

Step 4: State Allocation

State allocation to two JK FFs: A and B

State	AB
S0	**00**
S1 (S12)	**01**
S2 (S35)	**11**
S3 (S46)	**10**

JK FF's Excitation Table

Qt	Qt+1	J	K
0	0	0	*
0	1	1	*
1	0	*	1
1	1	*	0

State Transition Table with JK FFs: A and B

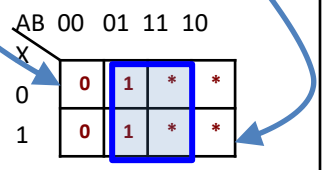
	Input X = 0		Input X = 1		Input X=0		Input X=1		Input X=0		Input X=1	
Current State	Next State	Output	Next State	Output	Inputs to JK FF-A				Inputs to JK FF-B			
A_tB_t	$A_{t+1}B_{t+1}$	Z	$A_{t+1}B_{t+1}$	Z	JA	KA	JA	KA	JB	KB	JB	KB
00	01	0	01	0	0	*	0	*	1	*	1	*
01	10	0	11	0	1	*	1	*	*	1	*	0
11	00	1	00	1	*	1	*	1	*	1	*	1
10	00	0	00	0	*	1	*	1	0	*	0	*

Step 5: Karnaugh Maps

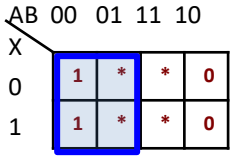
State Transition Table with JK FFs: A and B

	Input X = 0		Input X = 1		Input X=0		Input X=1		Input X=0		Input X=1	
Current State	Next State	Output	Next State	Output	Inputs to JK FF-A				Inputs to JK FF-B			
A_tB_t	$A_{t+1}B_{t+1}$	Z	$A_{t+1}B_{t+1}$	Z	JA	KA	JA	KA	JB	KB	JB	KB
00	01	0	01	0	0	*	0	*	1	*	1	*
01	10	0	11	0	1	*	1	*	*	1	*	0
11	00	1	00	1	*	1	*	1	*	1	*	1
10	00	0	00	0	*	1	*	1	0	*	0	*

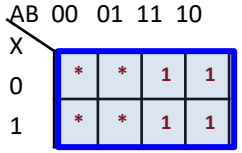
$J_A = B$



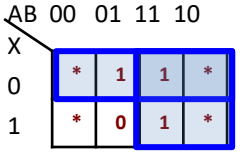
$J_B = \sim A$



$K_A = 1$

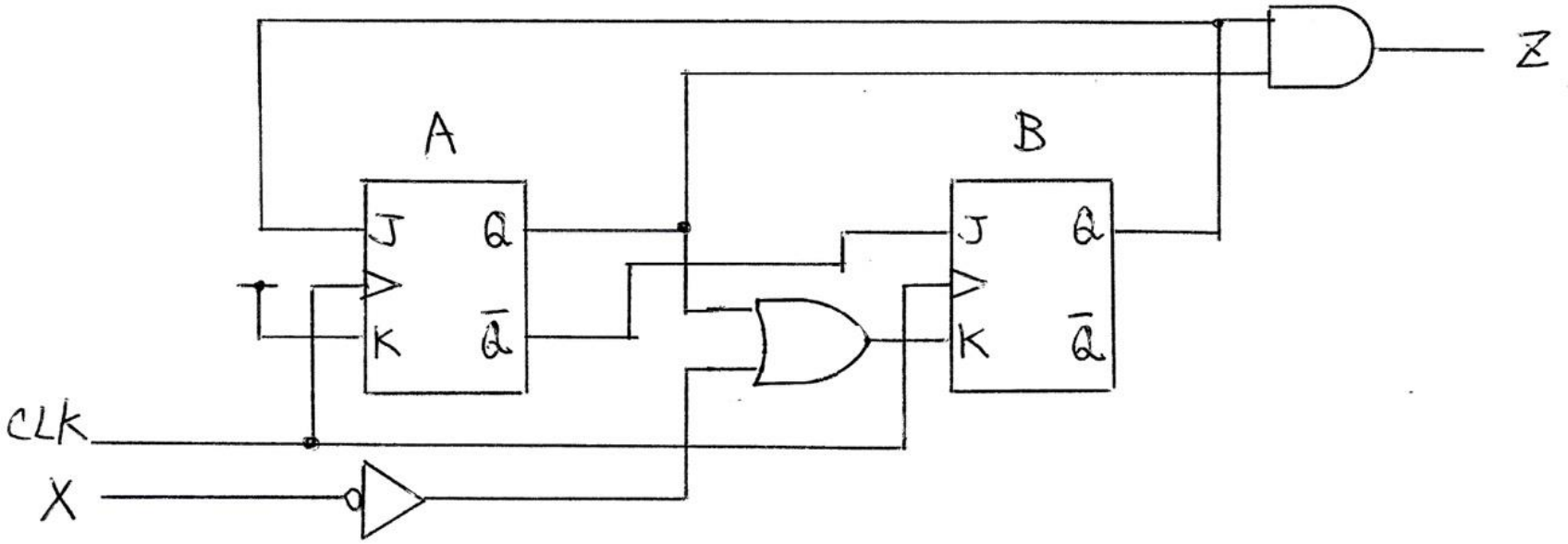


$K_B = \sim X + A$



$Z = AB\sim X + ABX = AB(\sim X + X) = AB$

3-bit Sequential Circuit's Circuit



How about Allocating D FF?

State allocation to two D FFs: A and B

State	AB
S0	**00**
S1 (S12)	**01**
S2 (S35)	**11**
S3 (S46)	**10**

D FF State Transition Table

Qt	Qt+1	D
0	0	0
0	1	1
1	0	0
1	1	1

State Transition Table with D FFs: A and B

	Input X = 0		Input X = 1		Input X = 0	Input X = 1	Input X = 0	Input X = 1
Current State	Next State	Output	Next State	Output	Inputs to D FF-A		Inputs to D FF-B	
A_tB_t	$A_{t+1}B_{t+1}$	Z	$A_{t+1}B_{t+1}$	Z	D_A	D_A	D_B	D_B
00	01	0	01	0	0	0	1	1
01	10	0	11	0	1	1	0	1
11	00	1	00	1	0	0	0	0
10	00	0	00	0	0	0	0	0

Kmap and D FF-based Circuit

State Transition Table with D FFs: A and B

	Input X = 0		Input X = 1		Input X = 0	Input X = 1	Input X = 0	Input X = 1
Current State	Next State	Output	Next State	Output	Inputs to D FF-A		Inputs to D FF-B	
A_tB_t	$A_{t+1}B_{t+1}$	Z	$A_{t+1}B_{t+1}$	Z	D_A	D_A	D_B	D_B
00	01	0	01	0	0	0	1	1
01	10	0	11	0	1	1	0	1
11	00	1	00	1	0	0	0	0
10	00	0	00	0	0	0	0	0

$D_A = \sim AB$

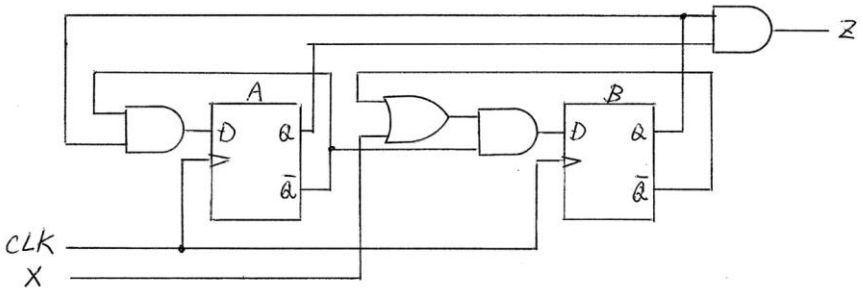
AB / X	00	01	11	10
0	0	1	0	0
1	0	1	0	0

Kmap

$D_B = \sim A\sim B + \sim AX$
$= \sim A(\sim B + X)$

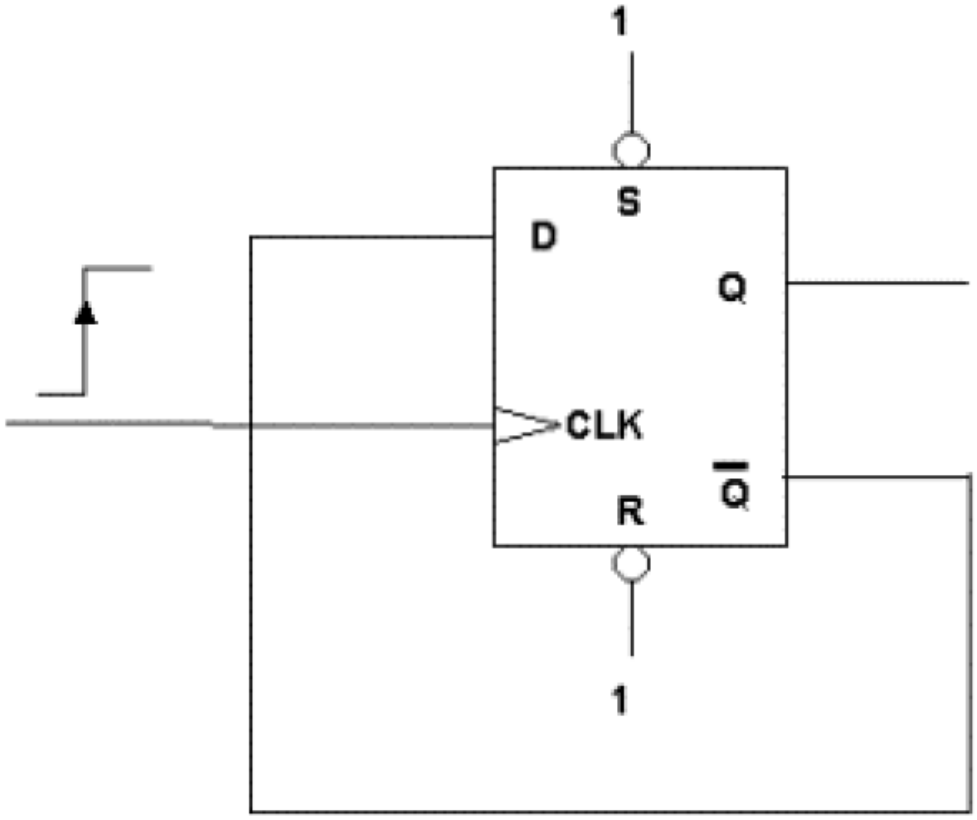
AB / X	00	01	11	10
0	1	0	0	0
1	1	1	0	0

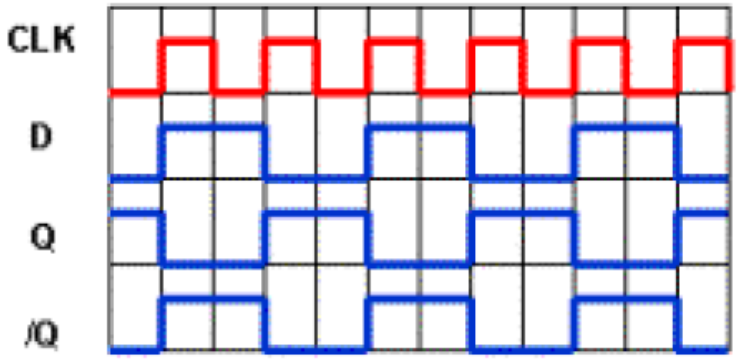
Kmap



$Z = AB(\sim X + X) = AB$

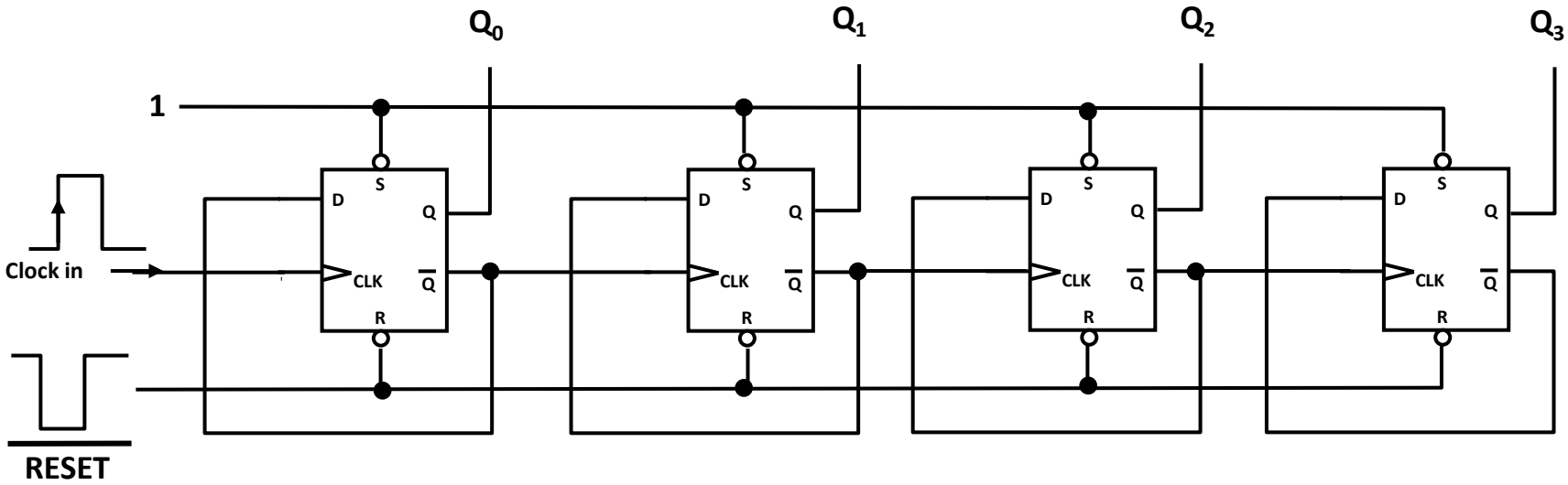
D FF connected in a divide-by-two configuration





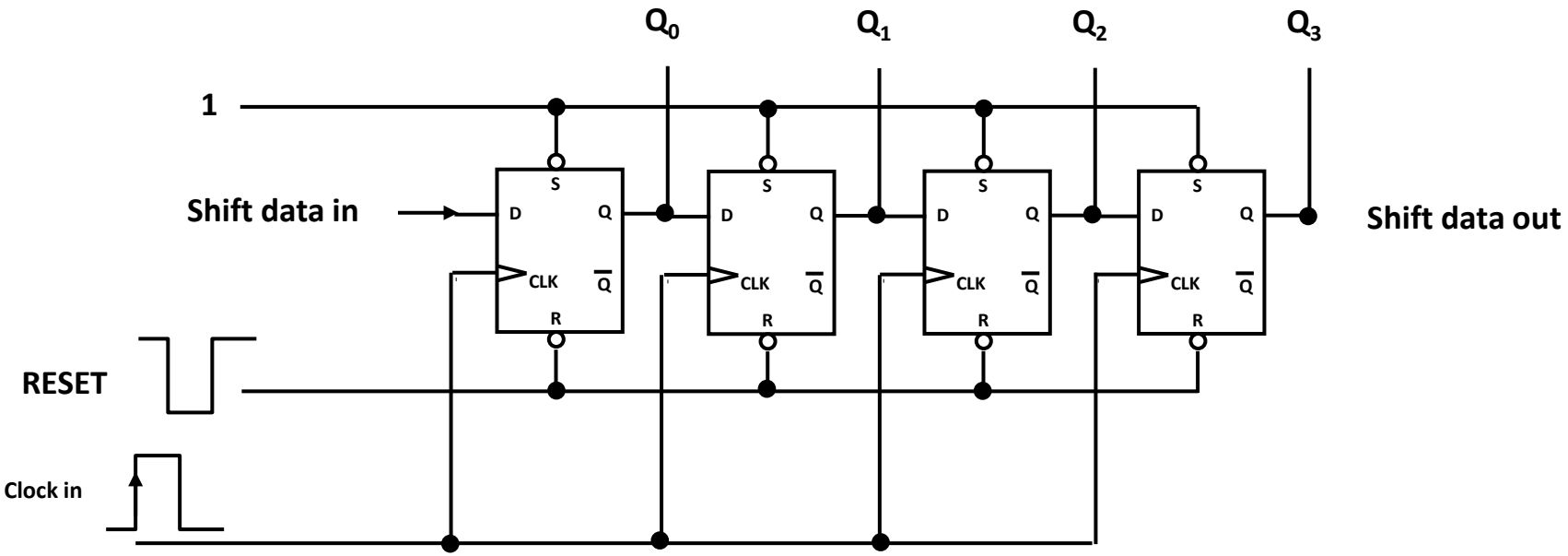
D Flip-Flops as a Ripple Counter

- A 4-bit binary *ripple counter*
  - Given the name because the pulses "ripple" through the circuit



- Each "D" FF divides the incoming clock frequency by 2X
- RESET sets all Q output to 0 without a clock signal (asynchronous)
- Counts as fast as the first stage can toggle, but cannot be read until the count has rippled through to the last stage
- Can build counter/dividers of any length, any binary divisor
  - Clock frequency at output Q_3 equals $f_{Clock\ in} \div 16$

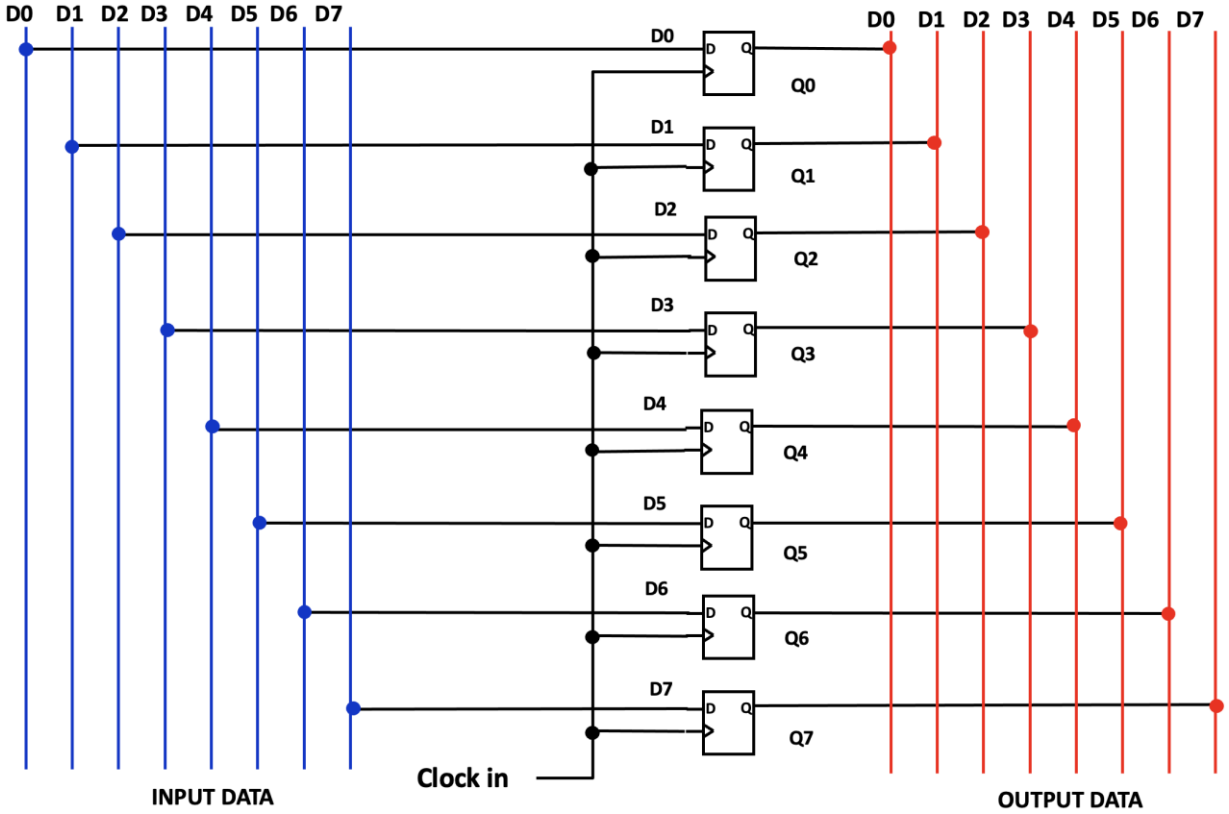
D Flip-Flops as a Shift Register



- **Shift register moves data through successive stages on each clock pulse**
- **Used for serial data communications, multiplication, image processing**
- **Basis for UART (*Universal Asynchronous Receiver/Transmitter*)**
- **Data can be read in serially and then read out in parallel**
- **Serial data communications limits the number of signal wires needed to transmit byte-wide data**

UNIVERSITY of WASHINGTON

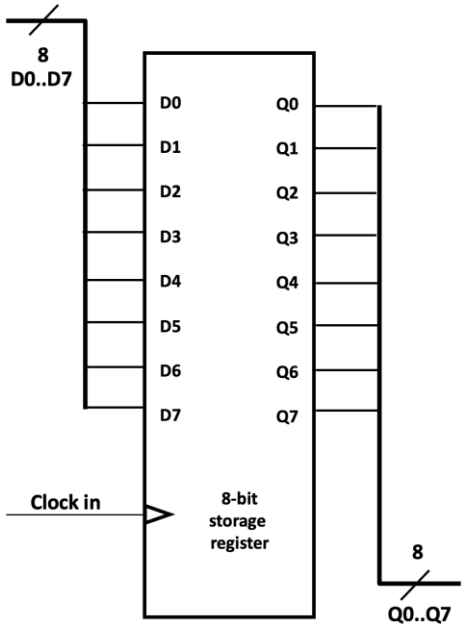
D Flip-Flops as a Storage Register



- ***BUS*: A grouping of "like" signals**

Here we have Data In bus (D0..D7) and Data out bus (Q0..Q7)

- **Busses are usually drawn with** heavier lines to differentiate them from individual signal wires



Summary

- Understand how to design and implement a finite state machine with flip-flops.

- Review typical sequential circuits that are used in computer design.