EE1D1: Digital Systems A

BSc. EE, year 1, 2025-2026, lecture 9

Sequential Logic Timing and Modules

Computer Engineering Lab

Faculty of Electrical Engineering, Mathematics & Computer Science

TU Delft

Delft University of Technology

Some slides were adapted from: S. Brown and Z. Vranesic , Fundamentals of Digital Logic with Verilog Design, B. Mohammad, KU, S. Lysecky, Arizona University

Learning Objectives

As student you should be able to:

- Understand the timing requirements for sequential circuits
- Explain Different types of Flip-Flops
- Interpret counters and registers
- Design some structured counters.

Outline

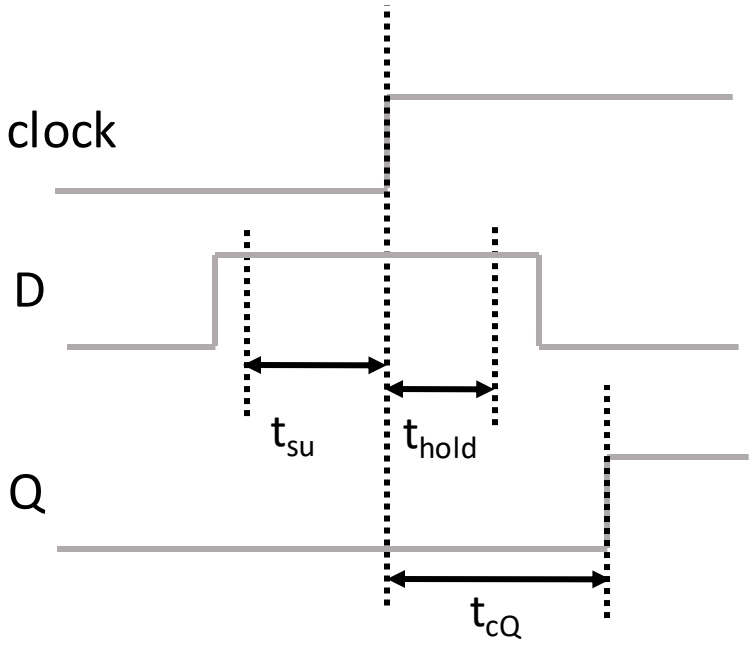
- Timing in sequential networks
- D, T and JK Flip-flop
- Sequential modules
  - Registers
  - Synchronous counters
  - Asynchronous counters
  - Binary Coded Decimal (BCD) counter
  - Ring counter
  - Johnson counter
- FSM using a counter
- Summary

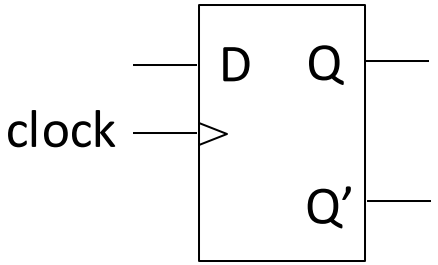
EE1D1: Digital Systems A

Timing in sequential networks

Flip-flop timing

- In practical circuits, propagation delay must be taken into account.
- A flip-flop output will be updated with a delay compared to the active clock edge.
- In addition, for proper operation, the D input signal must be valid and stable (i.e. not changing) around the time the clock signal changes. If this is not observed, the result may be unpredictable.





t_{su} = set-up time

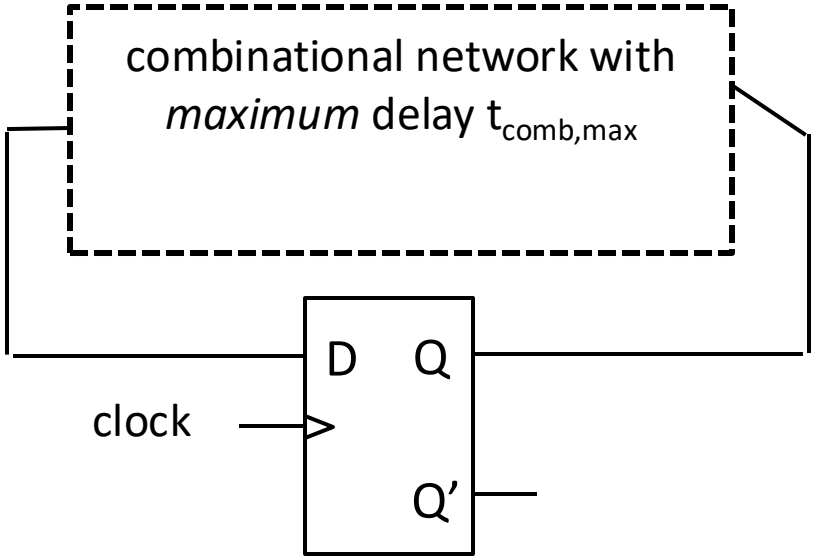
minimal time that D should stay stable *before* active clock edge

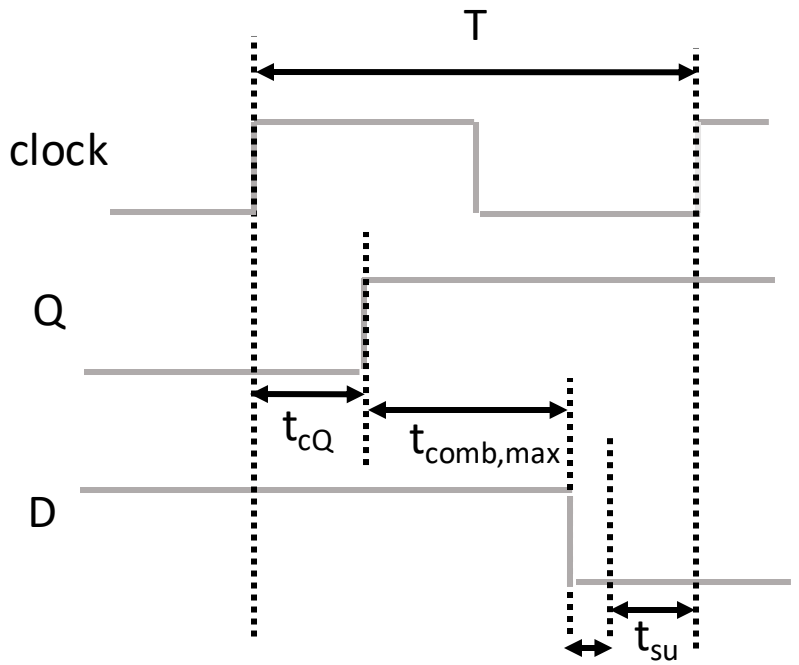
t_{hold} = hold time

minimal time that D should stay stable *after* active clock edge

t_{cQ} = delay time between active clock edge and output Q

Setup time and maximum clock frequency





For clock period T must hold $T > t_{cQ} + t_{comb,max} + t_{su}$

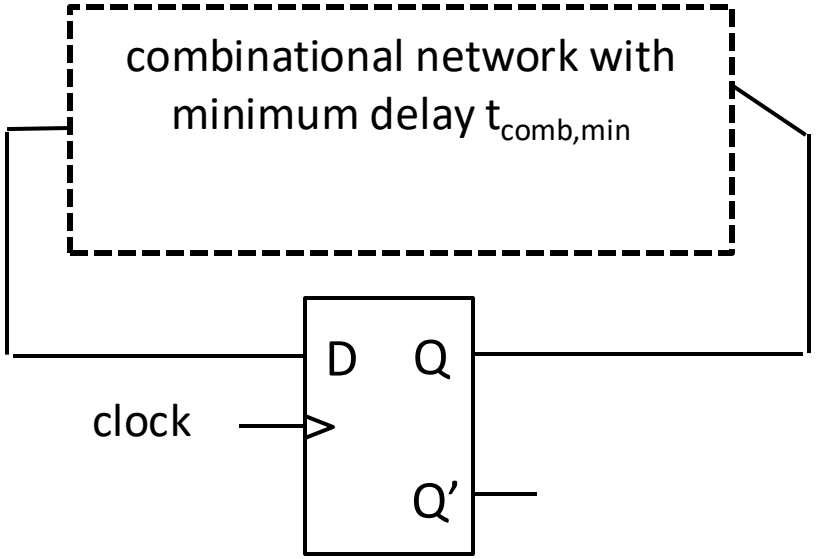
Hence maximal frequency $F_{max} = 1/T_{min} = 1/(t_{cQ} + t_{comb,max} + t_{su})$

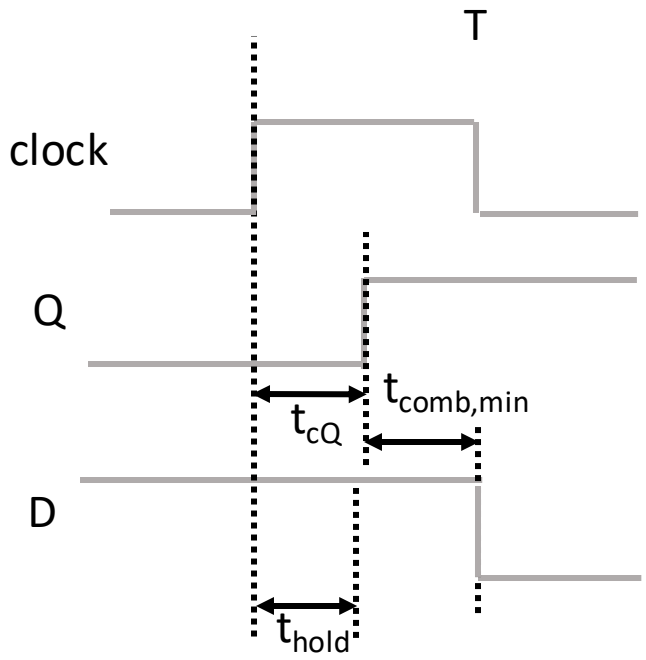
For example: $F_{max} = 1/(1.0 + 5.2 + 0.6\ ns) = 147\ MHz$

Setup slack of a FF input is defined as: $T - (t_{cQ} + t_{comb,max} + t_{su})_{for\ that\ input}$

So, a positive setup slack for each FF input is required

Hold time





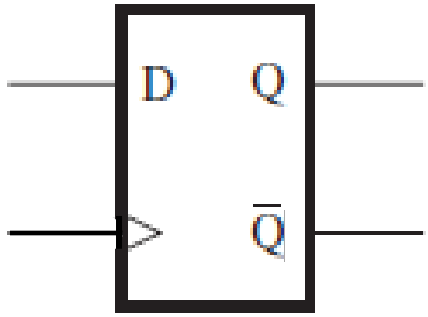
For the hold time, a problem will occur if $t_{cQ} + t_{comb,min} < t_{hold}$
Fortunately, it usually holds that $t_{cQ} > t_{hold}$ so that the problem does not occur (when all flip-flops receive clock edge at same time).

EE1D1: Digital Systems A

Other types of flip-flops

D Flip-Flop

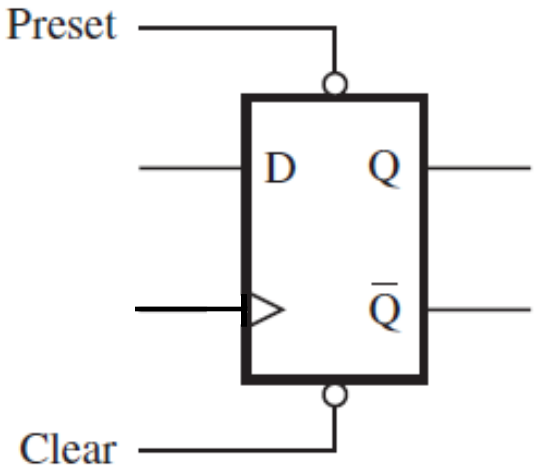
Symbol

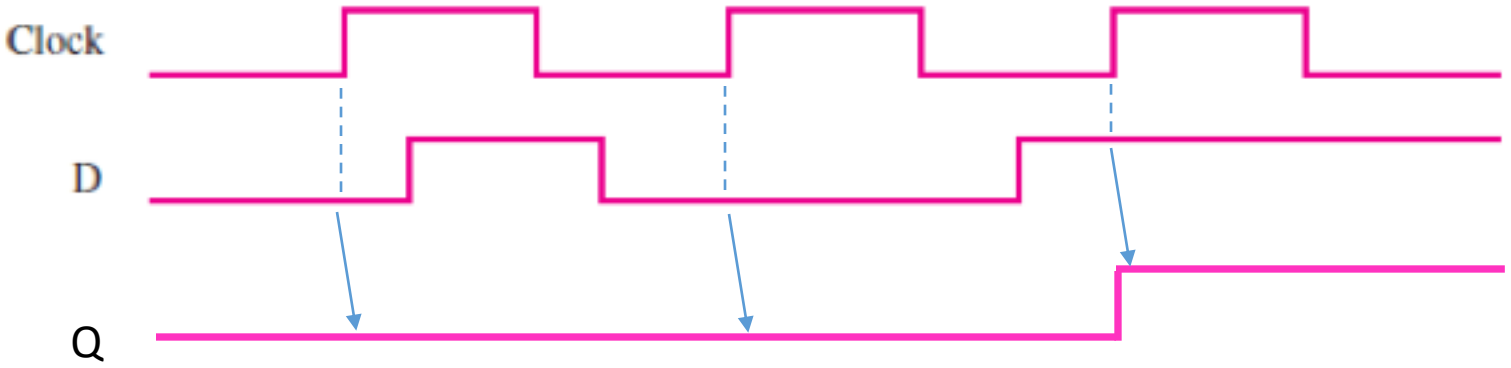


Characteristic table

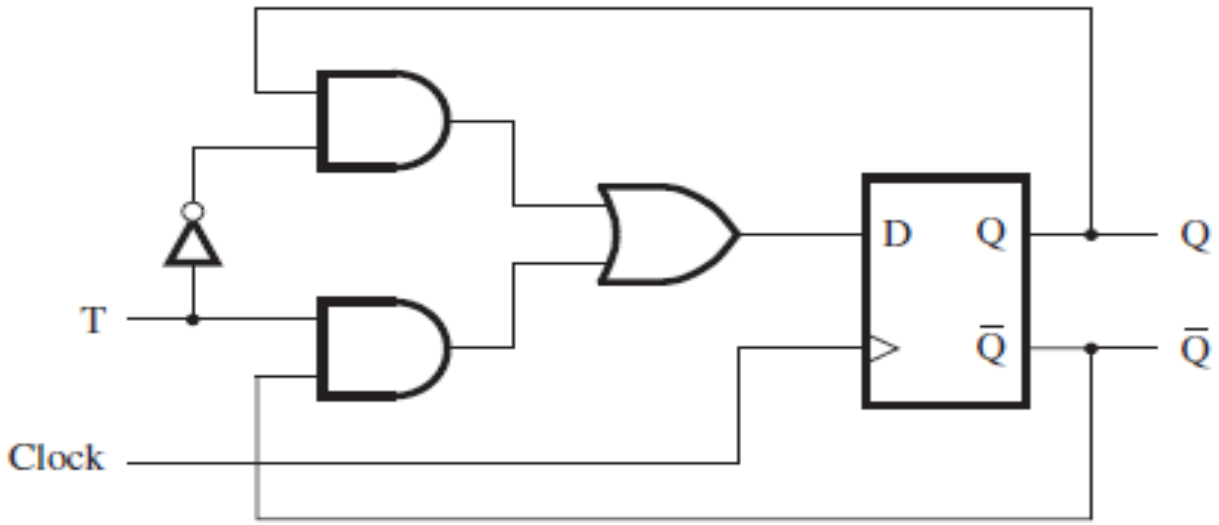
D	Q(t+1)
0	0
1	1

We may add Preset and Clear features to Set or Reset the FF, respectively





T Flip-Flop

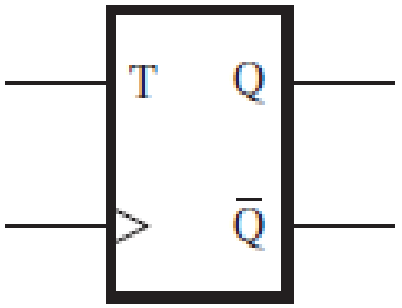


$$D = \overline{T}\,Q + T\,\overline{Q} = T \oplus Q$$

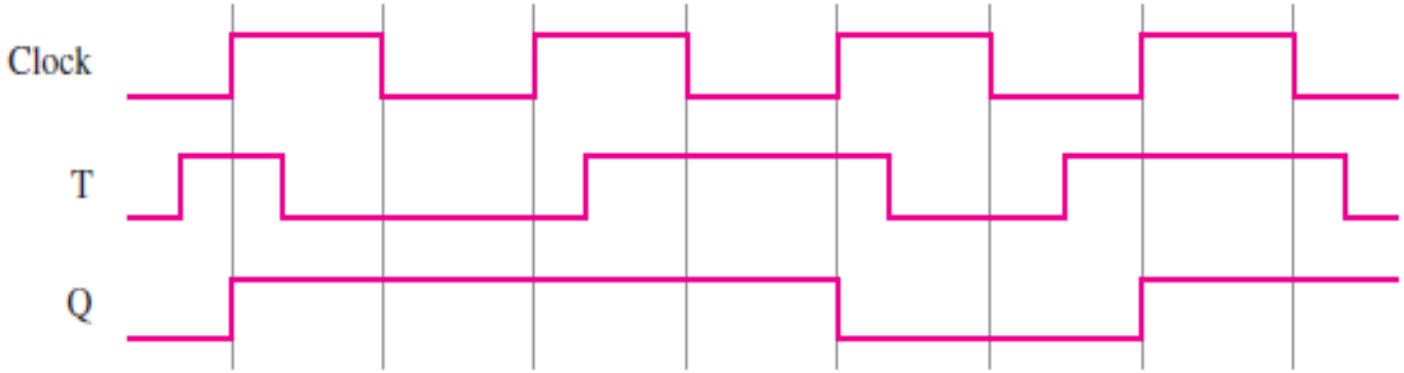
Characteristic table

T	Q(t + 1)
0	Q(t)
1	Q̄(t)

Symbol

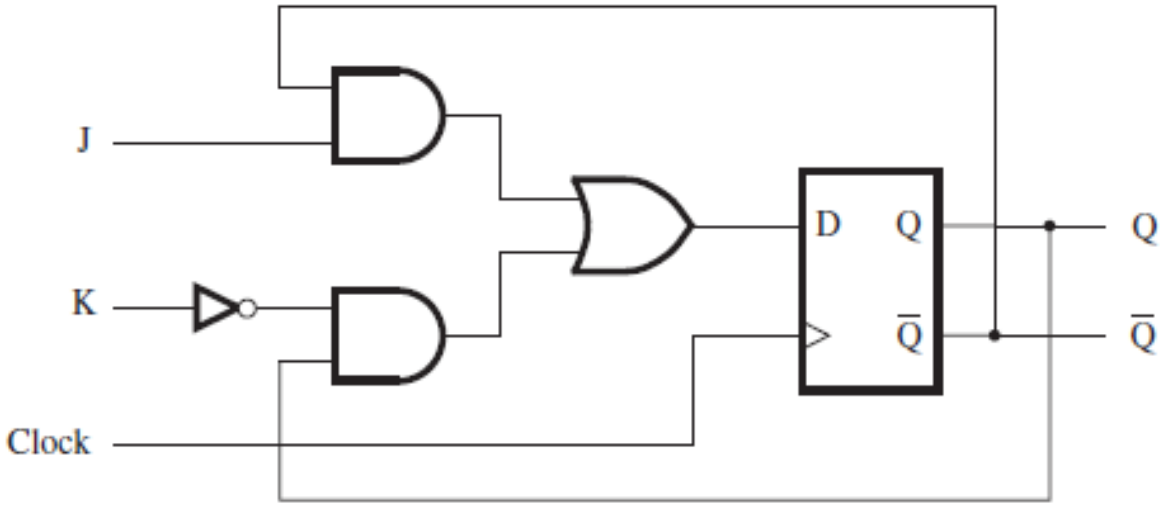


- D equals Q or Q' depending on value of T.
- If T=0, then D=Q and Q(t+1)=Q(t): No change in state.
- If T=1, then D=Q' and Q(t+1)=Q'(t): State will reverse or Toggle.
- T flip-flop useful for designing counter circuits.



JK Flip-Flop

$$D = J\overline{Q} + \overline{K}Q$$

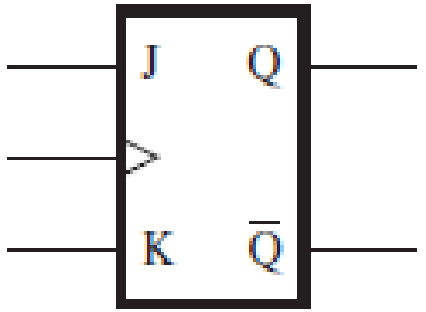


- When J=K=0 the JK flip-flop remembers its state.
- When J=0 and K=1 the JK flip-flop is reset.
- When J=1 and K=0 the JK flip-flop is set.
- When J=K=1 the JK flip-flop toggles its state
- How can JK FF be changed to a T FF?

Characteristic table

J	K	$Q(t+1)$
0	0	$Q(t)$
0	1	0
1	0	1
1	1	$\overline{Q}(t)$

Symbol

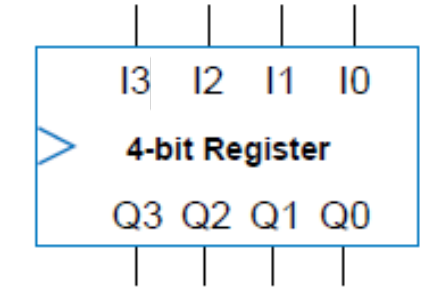


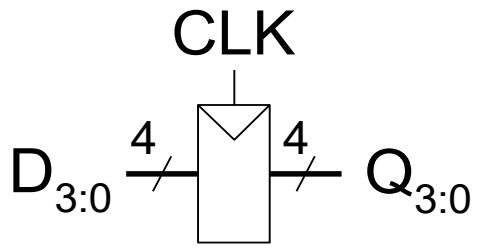
EE1D1: Digital Systems A

Sequential Modules

Basic Register

- Registers
  - Register: a combination of one or more flip-flops (mainly D type) with a common clock input.
  - An n-bit register is a structure that stores n-bits using n flip-flops.
  - A single register can be used to store related or unrelated bits of data or control information.

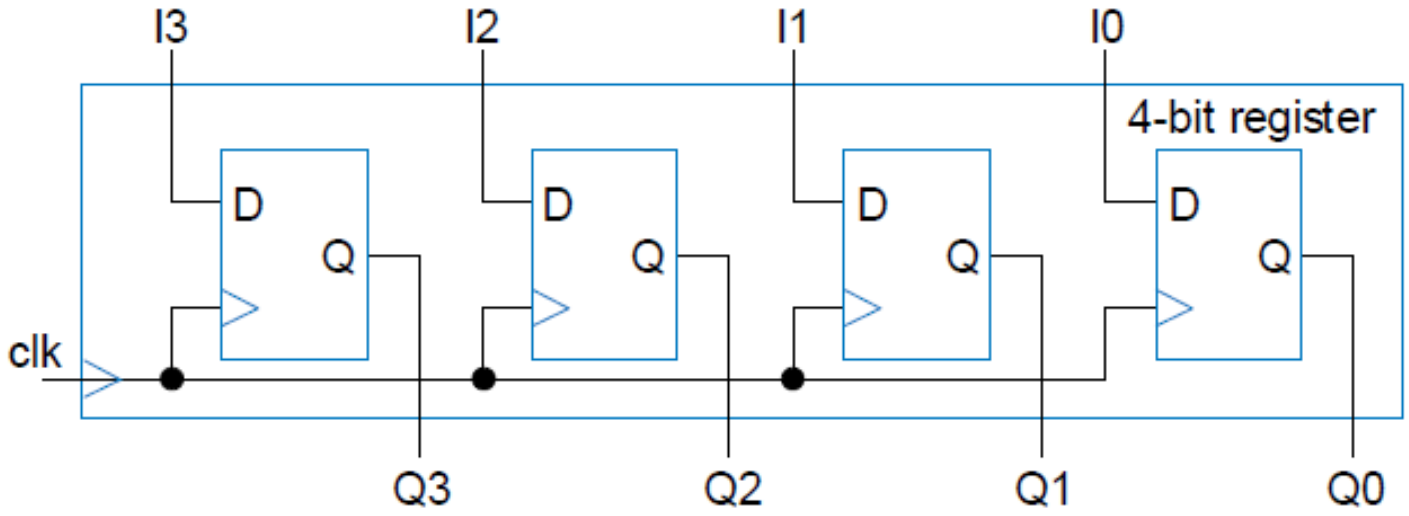




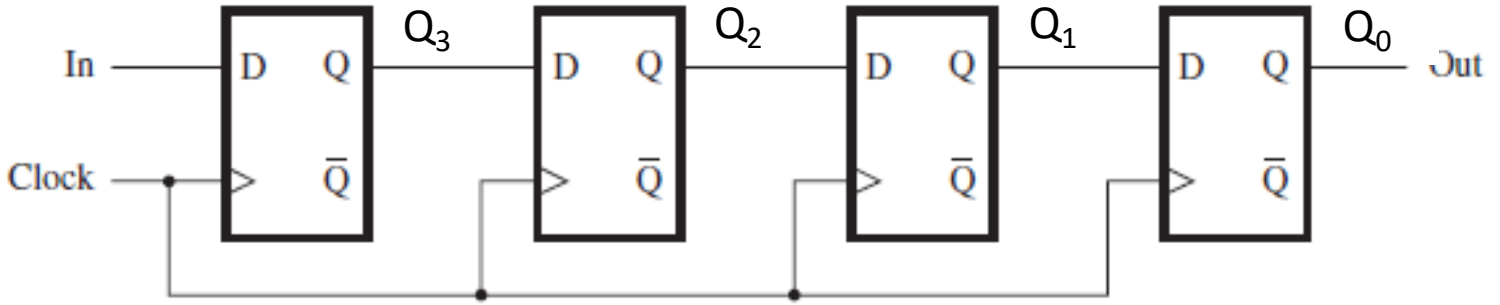
4-bit Register

Example of a typical 4-bit Register

Load bits I3-I0 on every clock cycle.



Shift Register

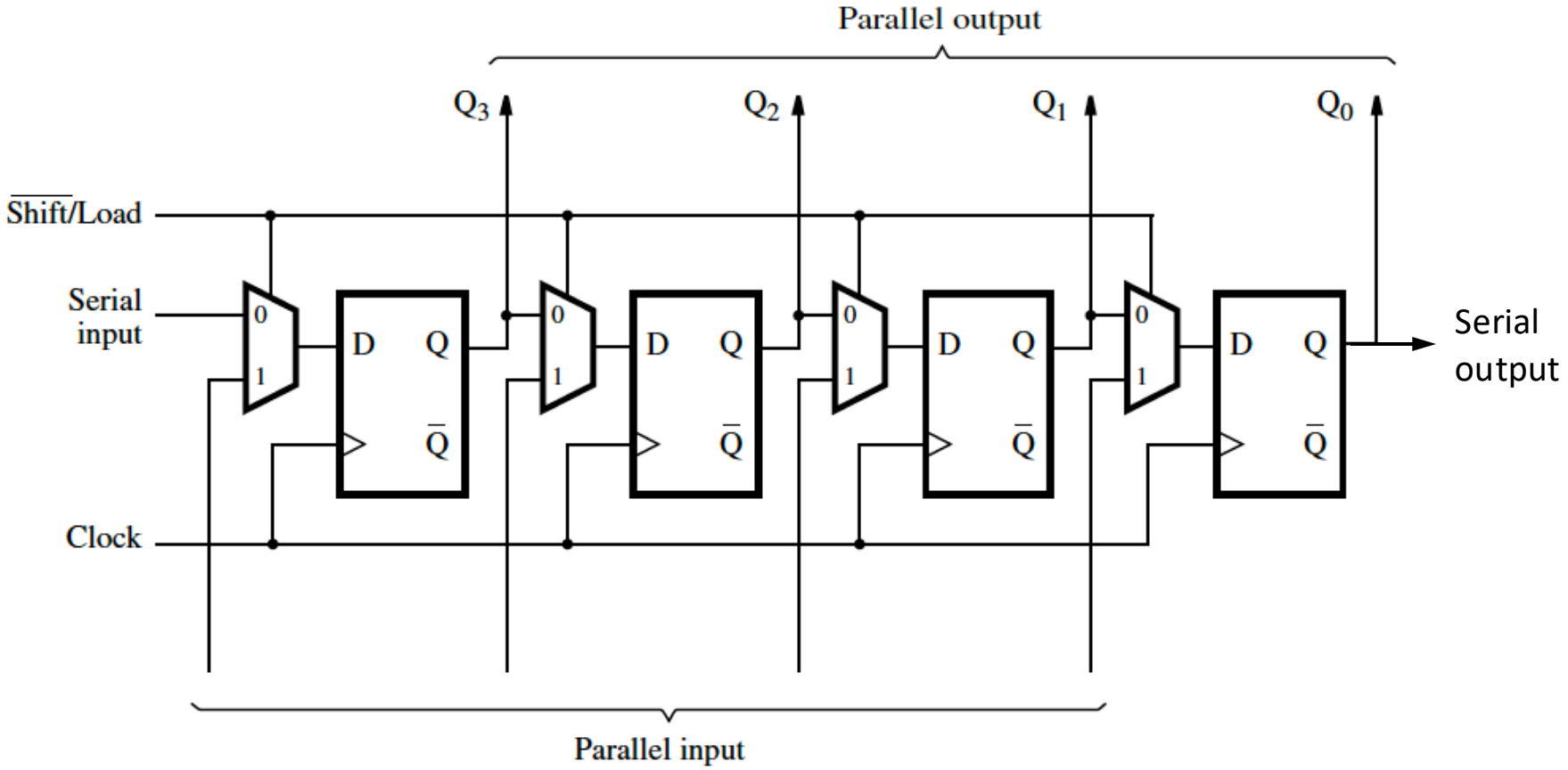


- In is a serial data input.
- Content of each flip-flop is transferred to the next flip-flop at each positive edge of the clock (Shift to the Right).

	In	Q_3	Q_2	Q_1	Q_0 = Out
t_0	1	0	0	0	0
t_1	0	1	0	0	0
t_2	1	0	1	0	0
t_3	1	1	0	1	0
t_4	1	1	1	0	1
t_5	0	1	1	1	0
t_6	0	0	1	1	1
t_7	0	0	0	1	1

Division by 2!

Parallel-Access Shift Register



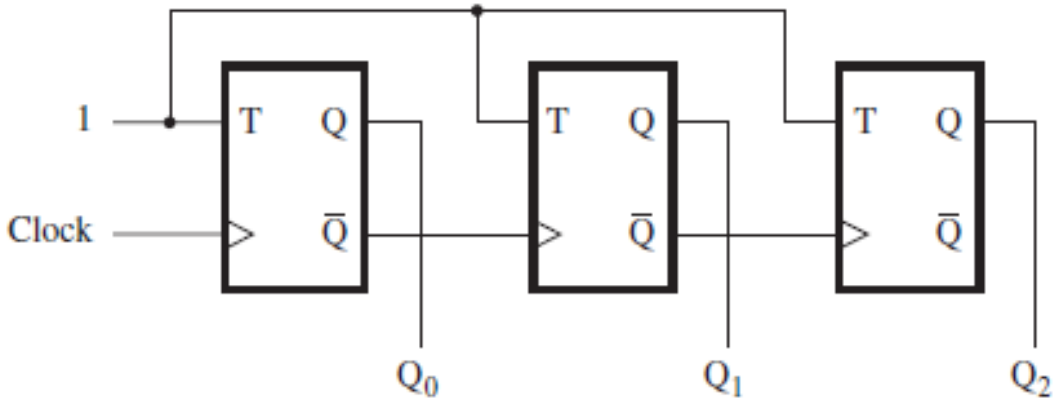
- $\overline{\text{Shift/Load}}$ = 0 circuit operates as a shift register.
- $\overline{\text{Shift/Load}}$= 1 parallel input data gets loaded into the register.
- Register content can be accessed in parallel through FF outputs Q3 – Q0 or serially by observing Q0.
- Circuit performs both series-to-parallel and parallel-to-series conversions

Counters

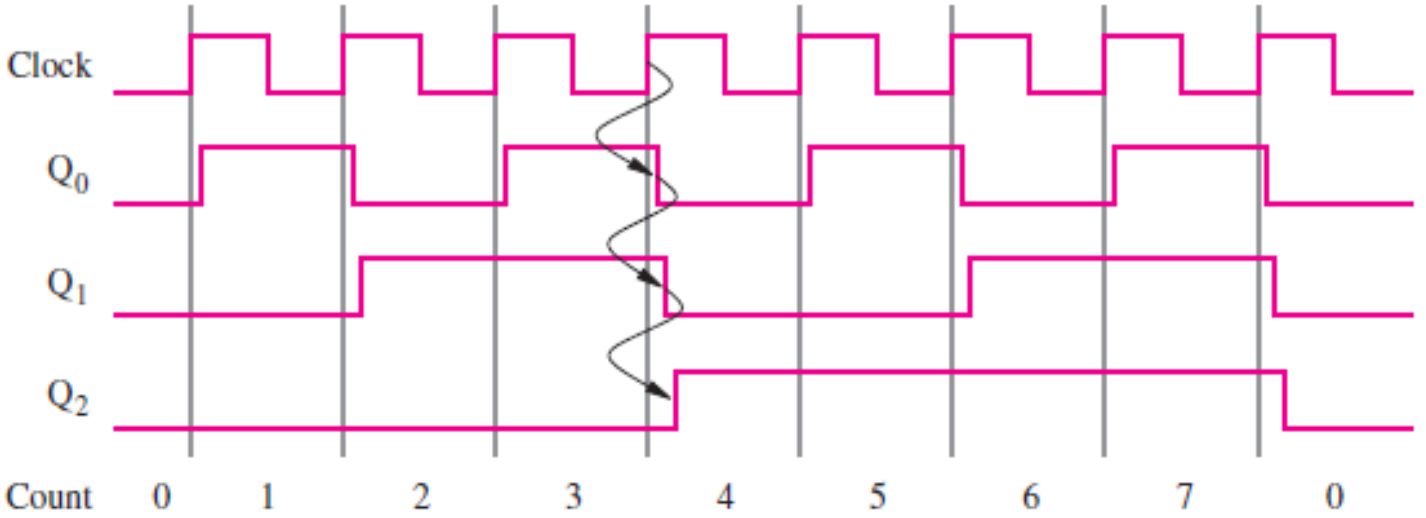
- **Counter circuits** are used in digital systems for many purposes.
  - They may count the number of occurrences of certain events,
  - generate timing intervals for control of various tasks in a system,
  - keep track of time elapsed between specific events, and so on.

- **Counters are broadly divided into two categories**
  - Asynchronous counter: doesn’t use universal clock
  - Synchronous counter: has one global clock so output changes in parallel

Asynchronous Up-Counter with T Flip-Flops



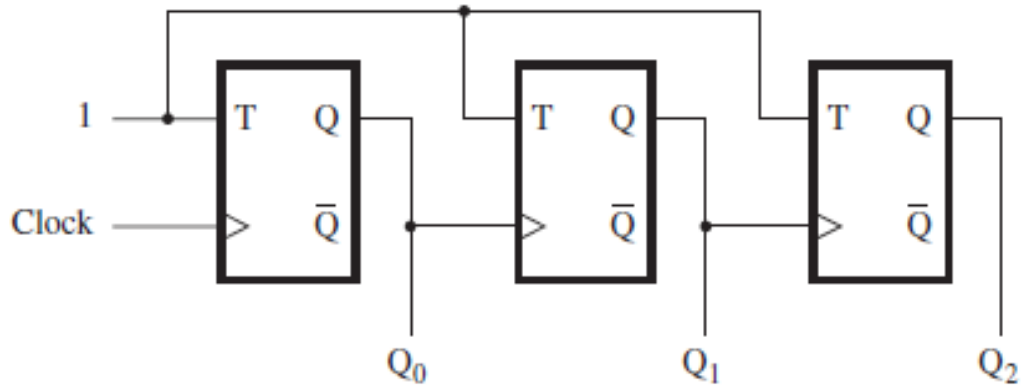
(a) Circuit



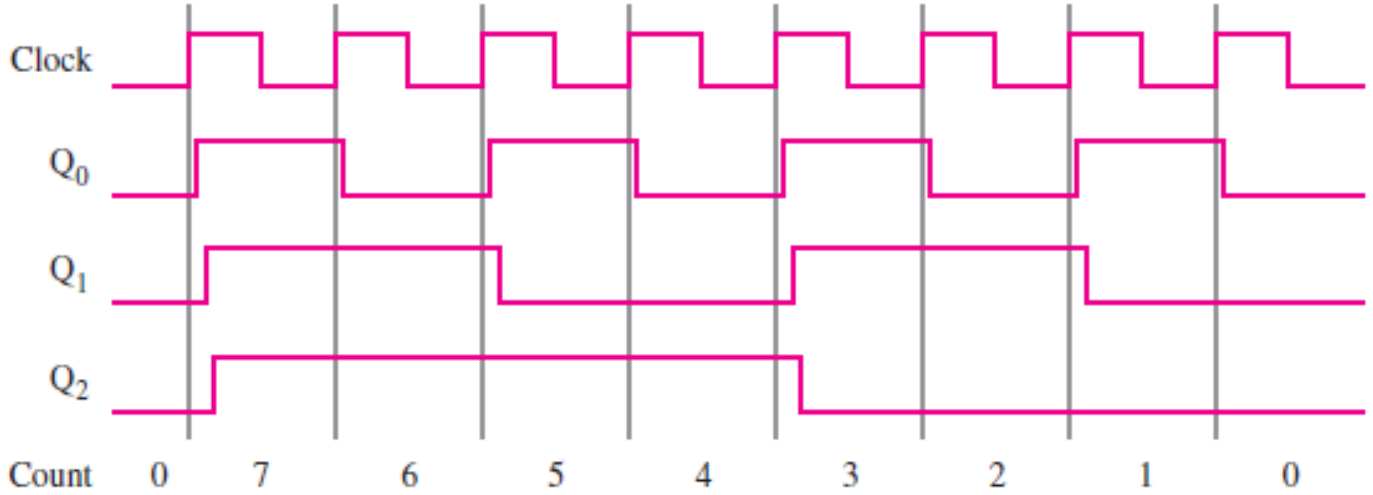
(b) Timing diagram

Asynchronous Down-Counter with T Flip-Flops

What's the drawback of asynchronous counters?



(a) Circuit



(b) Timing diagram

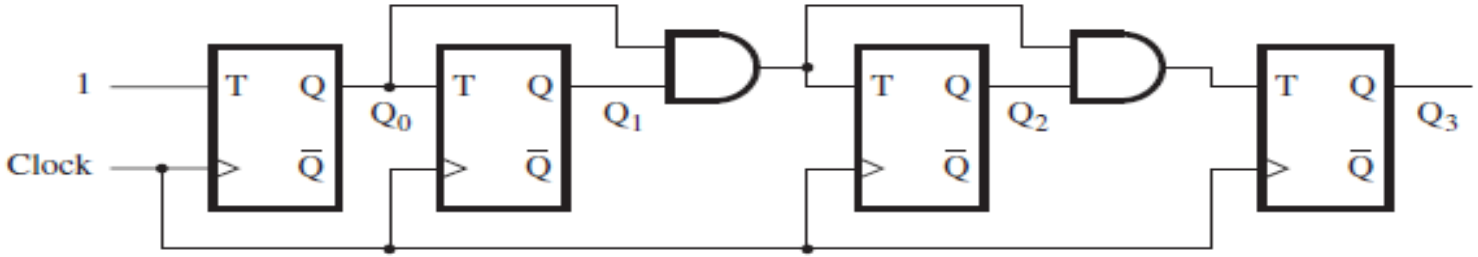
Synchronous Up-Counter

Clock cycle	Q_2 Q_1 Q_0
0	0 0 0
1	0 0 1
2	0 1 0
3	0 1 1
4	1 0 0
5	1 0 1
6	1 1 0
7	1 1 1
8	0 0 0

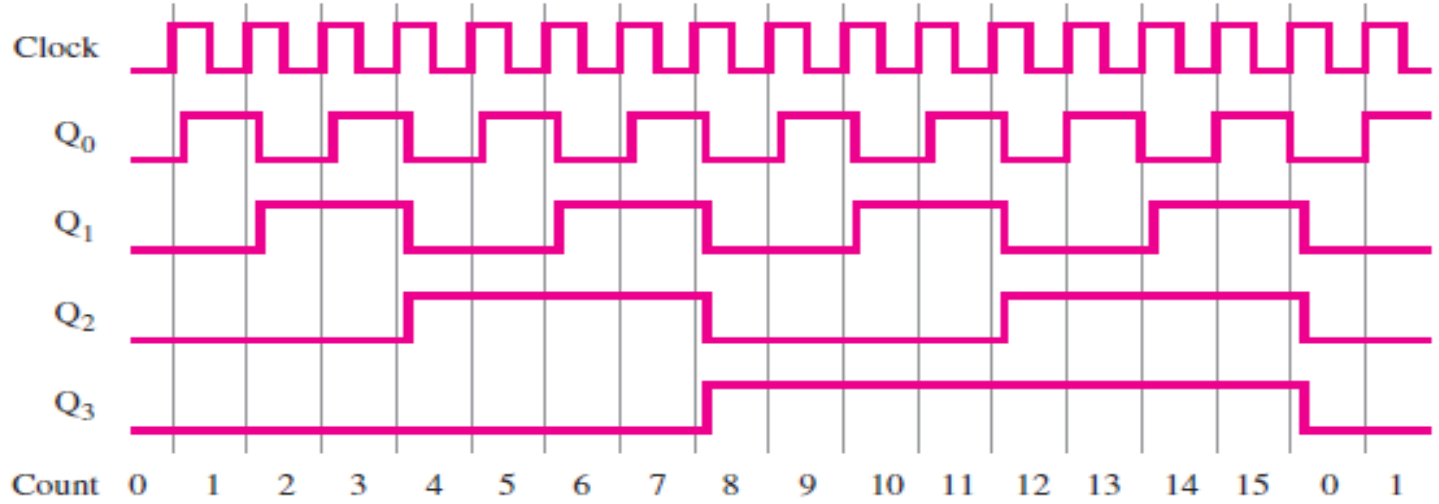
Q_1 changes

Q_2 changes

$$T_0 = 1$$
$$T_1 = Q_0$$
$$T_2 = Q_0 Q_1$$
$$T_3 = Q_0 Q_1 Q_2$$
$$\vdots$$
$$T_n = Q_0 Q_1 \cdots Q_{n-1}$$

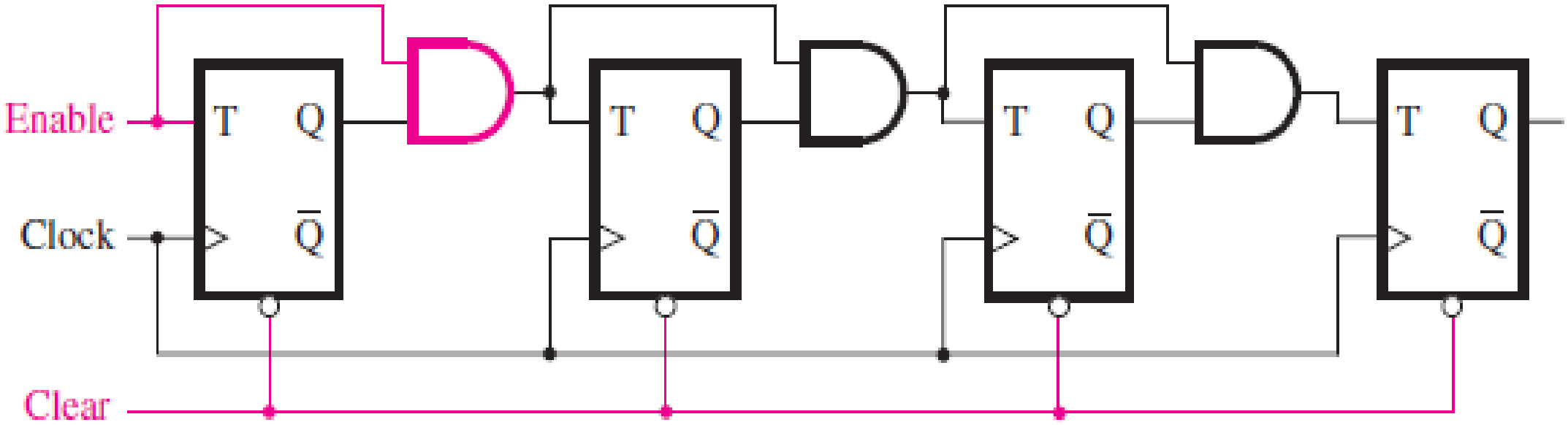


(a) Circuit



(b) Timing diagram

Synchronous Up-Counter: Inclusion of Enable and Clear capability



Synchronous Counter with D Flip-Flops

Assume *Enable* = 1

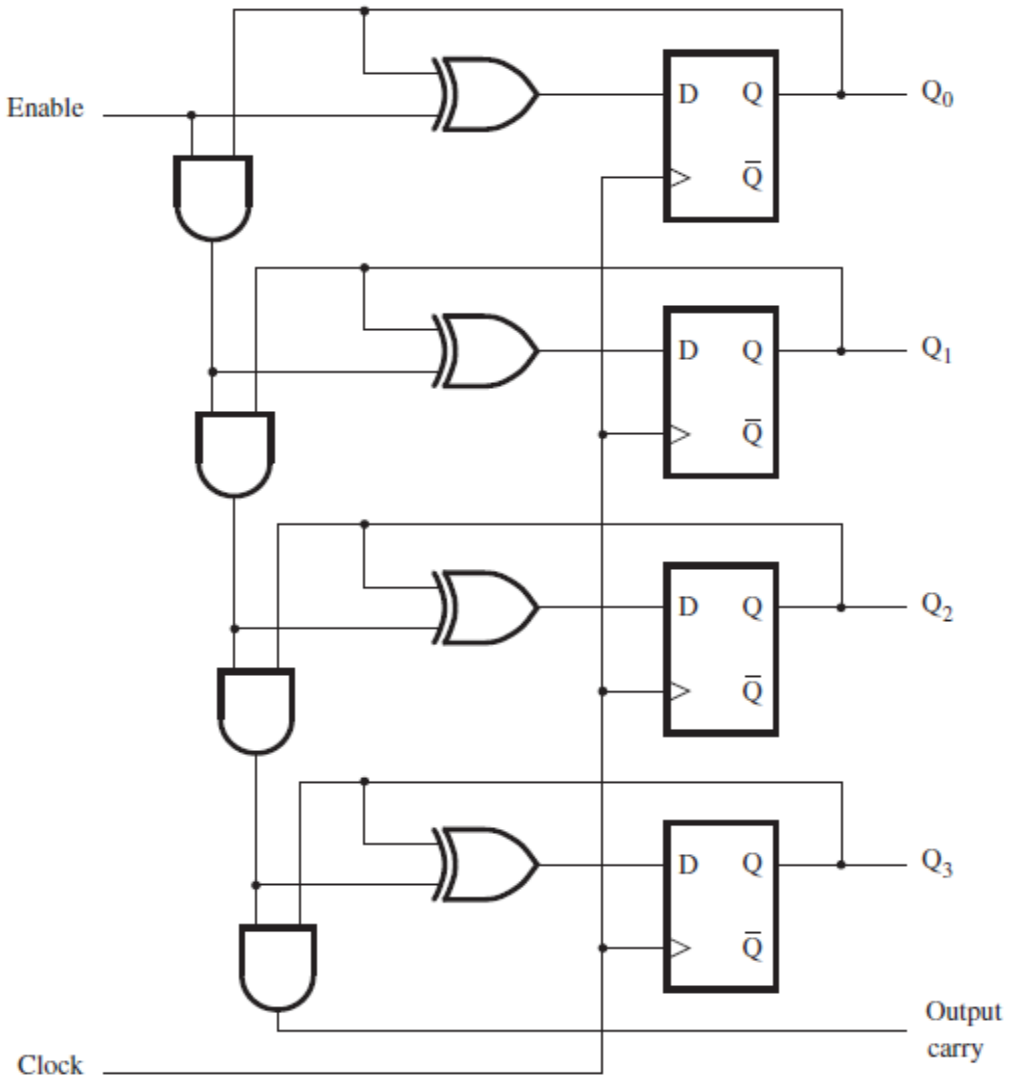
$$D_0 = \overline{Q}_0 = 1 \oplus Q_0$$
$$D_1 = Q_1 \oplus Q_0$$
$$D_2 = Q_2 \oplus Q_1Q_0$$
$$D_3 = Q_3 \oplus Q_2Q_1Q_0$$

For a larger counter, the *i*th stage is defined by

$$D_i = Q_i \oplus Q_{i-1}Q_{i-2} \cdot \cdot \cdot Q_1Q_0$$

Included the *Enable* control signal

$$D_0 = Q_0 \oplus Enable$$
$$D_1 = Q_1 \oplus Q_0 \cdot Enable$$
$$D_2 = Q_2 \oplus Q_1 \cdot Q_0 \cdot Enable$$
$$D_3 = Q_3 \oplus Q_2 \cdot Q_1 \cdot Q_0 \cdot Enable$$

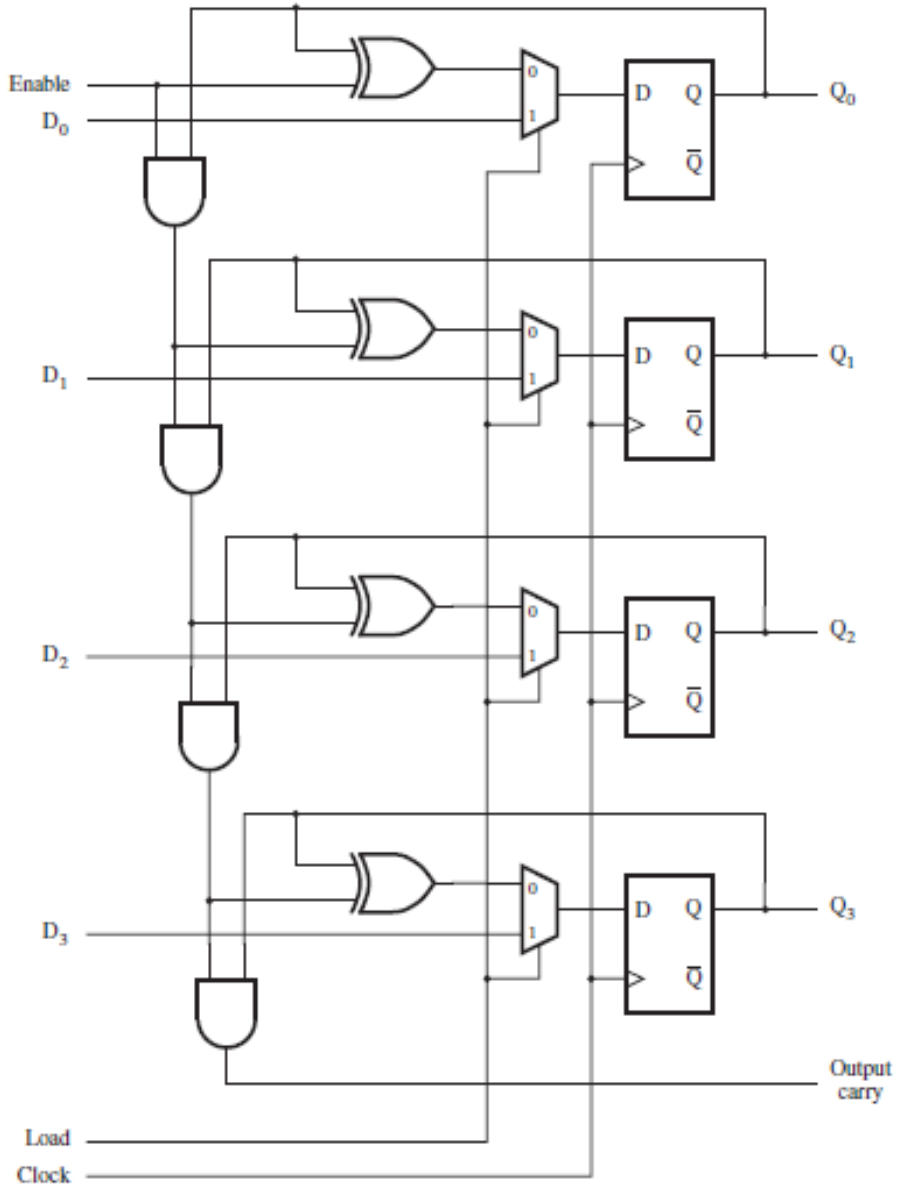


A counter with parallel-load capability

- Sometimes it is desirable to start the counter with a different count. To allow this mode of operation, a counter circuit must have some inputs through which the initial count can be loaded.

- A two-input multiplexer is inserted before each D input. One input to the multiplexer is used to provide the normal counting operation. The other input is a data bit that can be loaded directly into the flip-flop.

- A control input, Load, is used to choose the mode of operation. The circuit counts when Load = 0. A new initial value, $D_3D_2D_1D_0$,, is loaded into the counter when Load = 1.



Binary Coded Decimal (BCD) counter

- With BCD representation each decimal digit is separately represented by a binary number.

E.g.

27 = 0010 0111

95 = 1001 0101

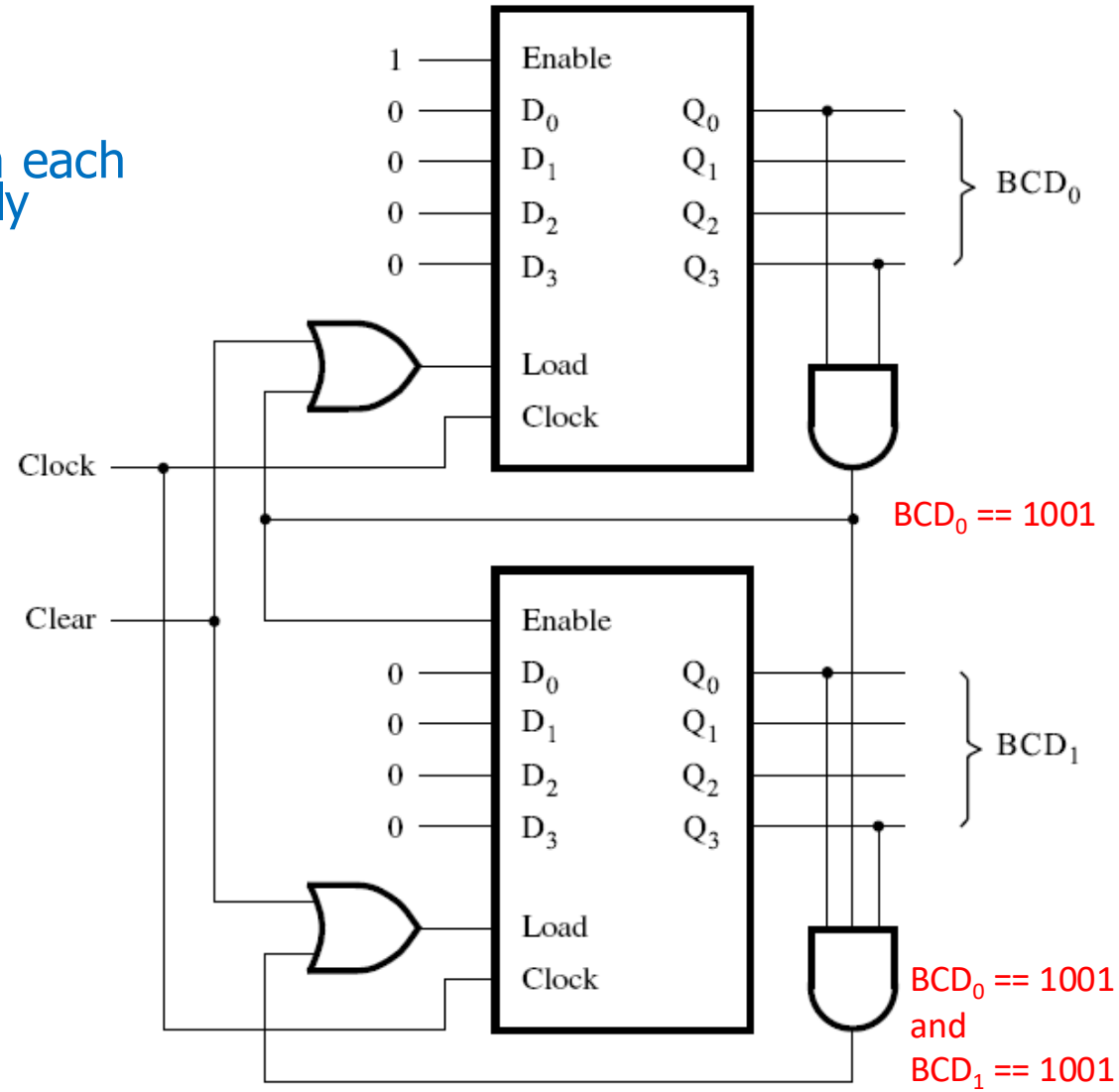
BCD_1 BCD_0

Hence, next state of

49 = 0100 1001

is:

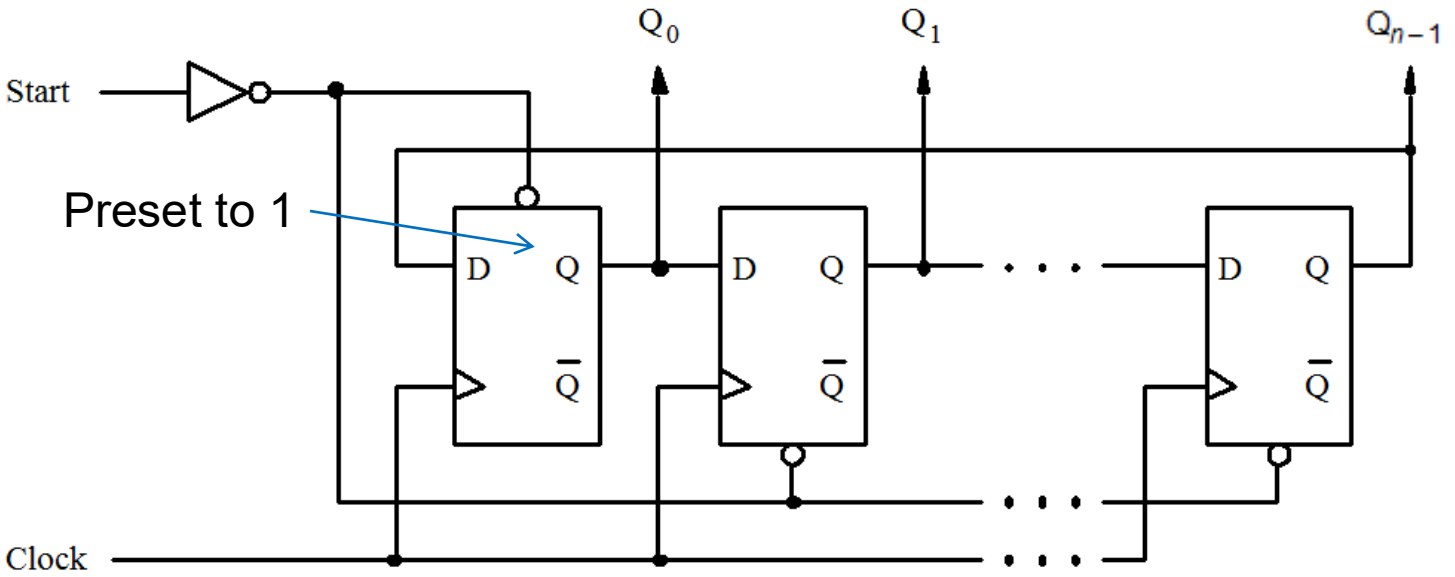
50 = 0101 0000



Ring Counter

- Consists of a shift register that shifts a '1'.
- Uses a start signal to initialize flip-flops.
- Each flip-flop reaches the state Q_i=1 for exactly one count, while for all other flip-flops have Q_i=0.
- An n-bit ring counter generates a sequence of length n.
- Flip-flops content can be thought to represent a code rather than a binary number.

An n-bit ring counter



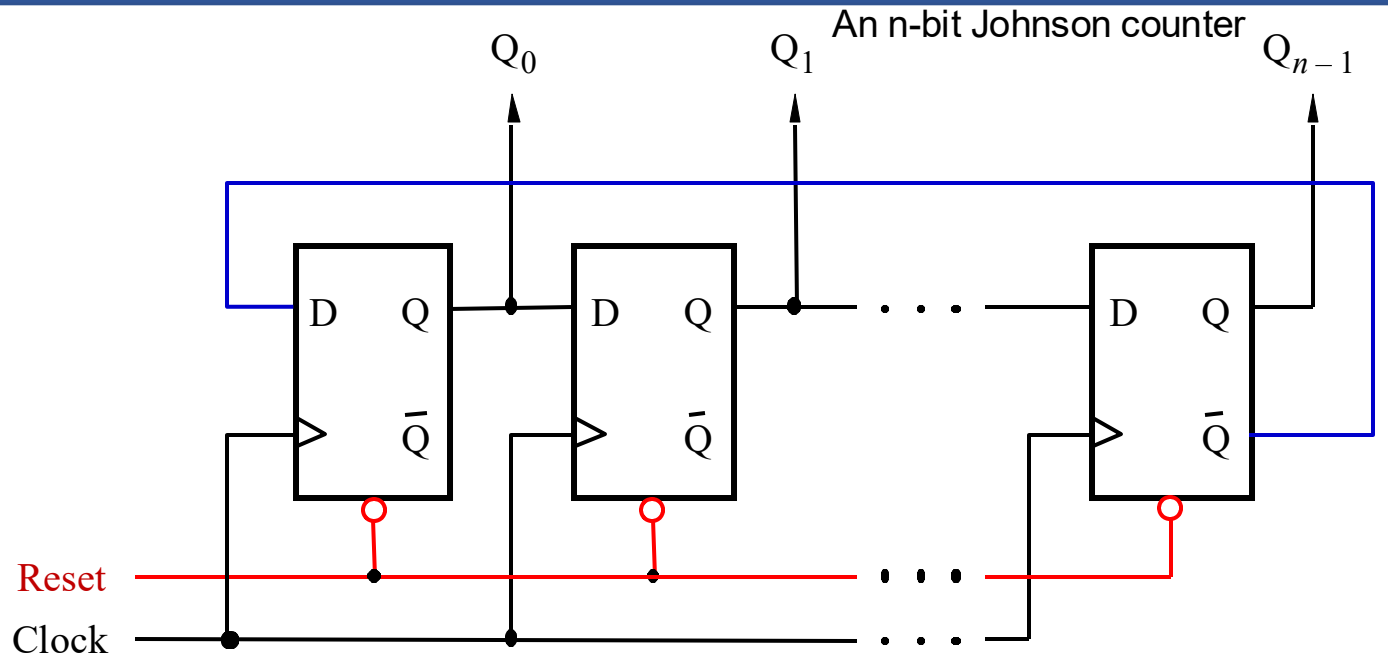
Count sequence for 4-bit Ring Counter

Clock Cycle	Q_0	Q_1	Q_2	Q_3
0	1	0	0	0
1	0	1	0	0
2	0	0	1	0
3	0	0	0	1
4	1	0	0	0

Codes generated by Ring Counter are called **One-Hot-Codes** as each code has a single 1 and the rest of the code variables are 0.

Johnson Counter

- Use a shift register and normal reset for initialization.
- Feedback Q' output of the last stage to input of 1st stage.
- An n-bit Johnson counter generates a sequence of length 2n.
- Compare Johnson counter & Ring counter.



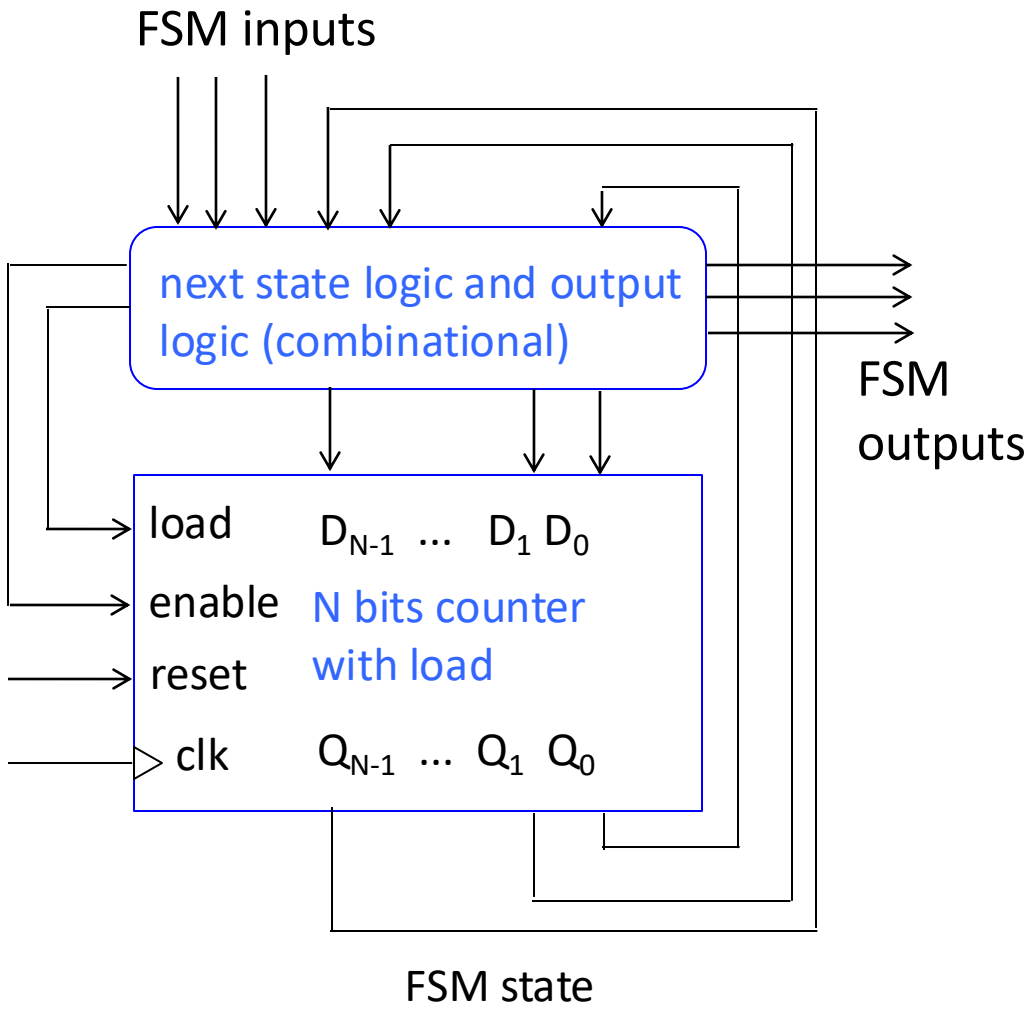
Count sequence for 4-bit Johnson Counter

Clock Cycle	Q_0	Q_1	Q_2	Q_3
0	0	0	0	0
1	1	0	0	0
2	1	1	0	0
3	1	1	1	0
4	1	1	1	1
5	0	1	1	1
6	0	0	1	1
7	0	0	0	1
8	0	0	0	0

FSM implemented using a counter

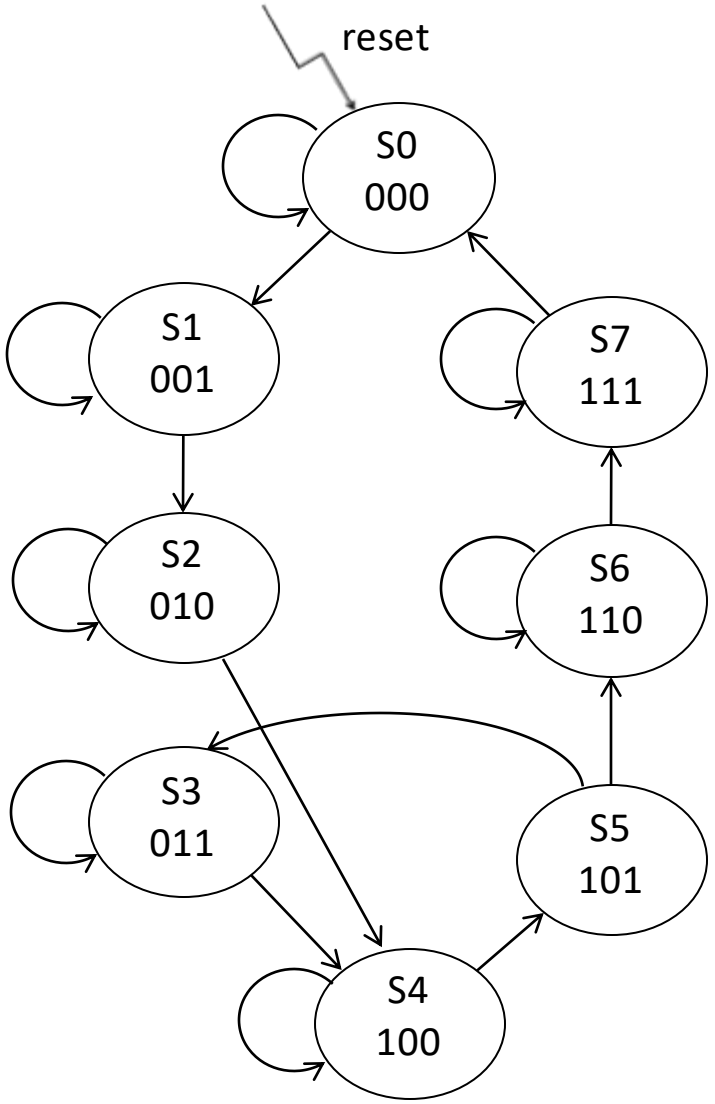
Instead of a set of flip-flops, or a register, a counter may also be used to store the state of an FSM.

- Going to a subsequent state is counting up.
- Staying in the same state is disabling the counter.
- Going to any other state is loading a new value (state) into the counter.



Example FSM implemented using a counter

Example



- Staying in the same state is implemented with load = enable = 0.
- Going to a subsequent state (e.g. from S1 to S2, or from S4 to S5) is implemented with enable = 1.
- Other transitions (e.g. from S2 to S4 or from S5 to S3) are implemented with load = 1.

Question

Given the adjacent FSM, with input A.

It has been implemented with a 3-bit counter.

- If LOAD = ENABLE = 0 the counter maintains its value.
- If LOAD = 0 and ENABLE = 1 the counter counts.
- If LOAD = 1, the counter loads regardless of ENABLE.

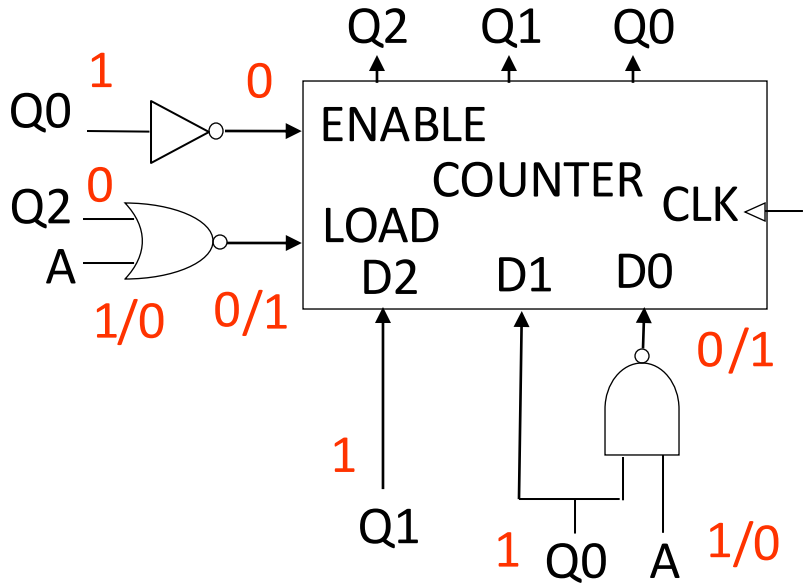
When the FSM is in state S3, what are possible next states?

a. S3 and S4

b. S3 and S7

c. S4 and S7

d. None of the above answers.

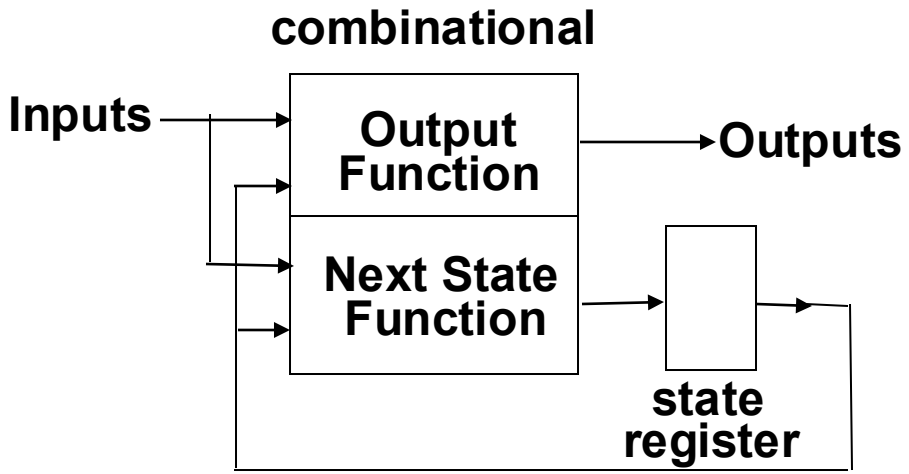


state	Q2,Q1,Q0
S0	000
S1	001
S2	010
S3	011
S4	100
S5	101
S6	110
S7	111

In state S3 we have: Q2,Q1,Q0=011.
If A=1 then LOAD=0 and ENABLE=0, so that the next state is again S3.
If A=0 then LOAD=1 and D2,D1,D0=111, so that the next state is S7.
Hence answer b.

FSM implementation using modules

Finite State Machine model



Implementation possibilities:

- **combinational: Single gates, Multiplexer, Decoder**
- **state register: Single flip-flops, Data register, Counter with parallel load**

Summary

- Flip-flop delay time, combinational network delay time and flip-flop setup time determine maximum clock frequency.
- The name T flip-flop derives from the behavior of the circuit, which “toggles” its state when T = 1.
- JK flip-flop combines the behavior of the T flip-flop, with a set and reset possibility.
- Counters can be built using different types of flip-flops.
- Asynchronous counter doesn’t use a universal clock.
- Synchronous counter has one global clock so outputs change in parallel.
- BCD uses the decimal counting sequence.
- Ring counter generates sequences of codes that do not represent binary numbers.
- Johnson counter is a variation of the ring counter. It is obtained if, instead of the Q output, we take the Q’ output of the last stage and feed it back to the first stage.
- Counters can also be used to create FSMs.

Further reading

“Digital Design”: sections 3.5.1 - 3.5.2, 5.4, exercises 3.9 and 3.10.

Fundamentals of Digital Logic with Verilog Design, Stephen Brown and Zvonko Vranesic, 3rd edition 2014 (pdf can be found on the internet):

- Sections 5.5 and 5.6: T and JK flip-flops
- Sections 5.9 and 5.11: counters