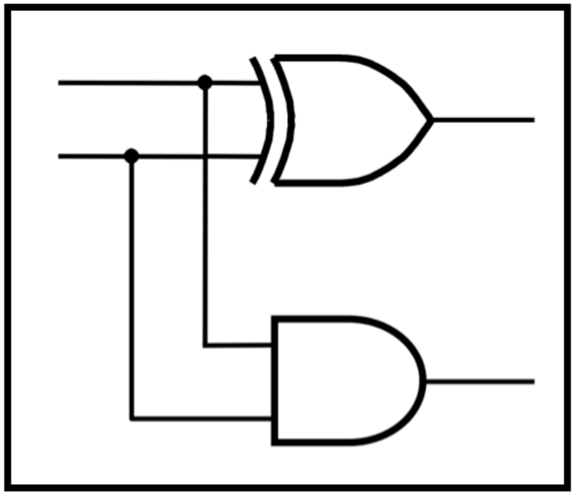


CprE 281: Digital Logic

Instructor: Alexander Stoytchev

http://www.ece.iastate.edu/~alexs/classes/

Registers and Counters

CprE 281: Digital Logic
Iowa State University, Ames, IA

Administrative Stuff

- The second midterm is this Friday.
- Homework 8 is due today.
- Homework 9 is out. It is due on Mon Nov 7.
- No HW due next Monday

Administrative Stuff

- Midterm Exam #2
- When: Friday October 28 @ 4pm.
- Where: This classroom
- What: Chapters 1, 2, 3, 4 and 5.1-5.8
- The exam will be open book and open notes (you can bring up to 3 pages of handwritten notes).

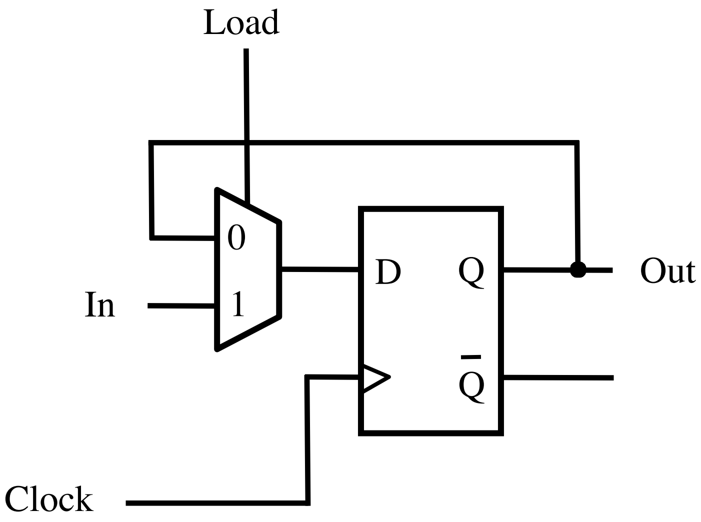
Registers

Register
(Definition)

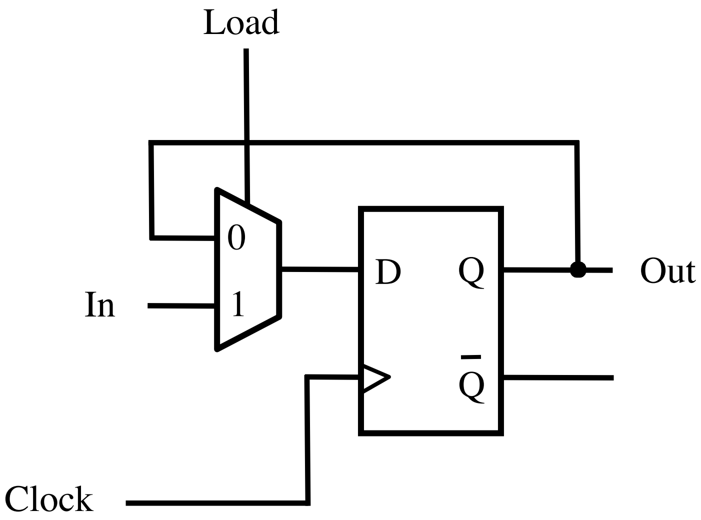
An n-bit structure consisting of flip-flops

Parallel-Access Register

1-Bit Parallel-Access Register



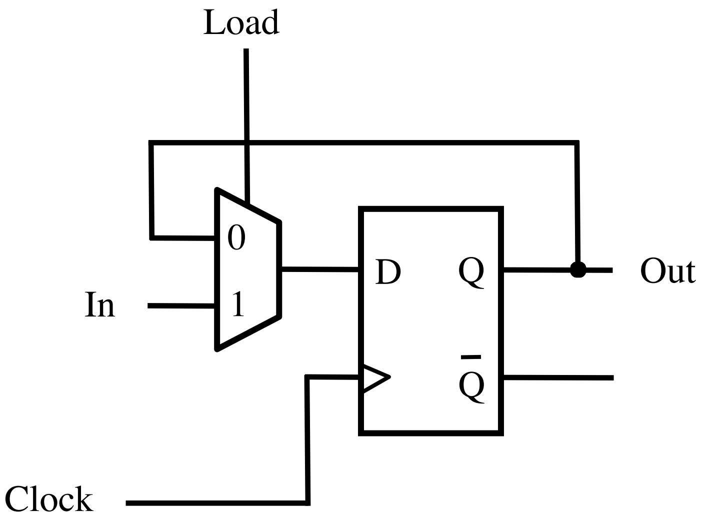
1-Bit Parallel-Access Register



The 2-to-1 multiplexer is used to select whether to load a new value into the D flip-flop or to retain the old value.

The output of this circuit is the Q output of the flip-flop.

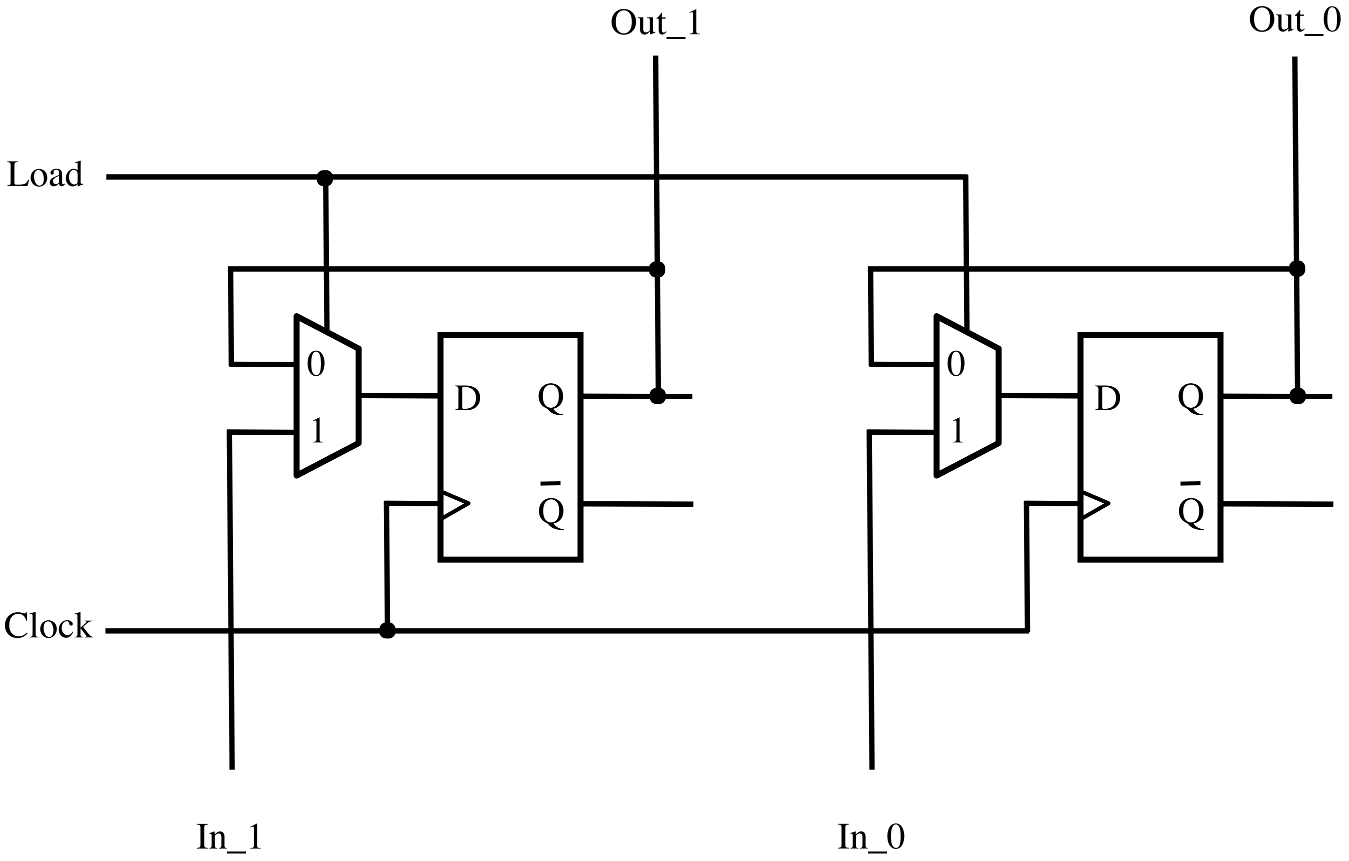
1-Bit Parallel-Access Register



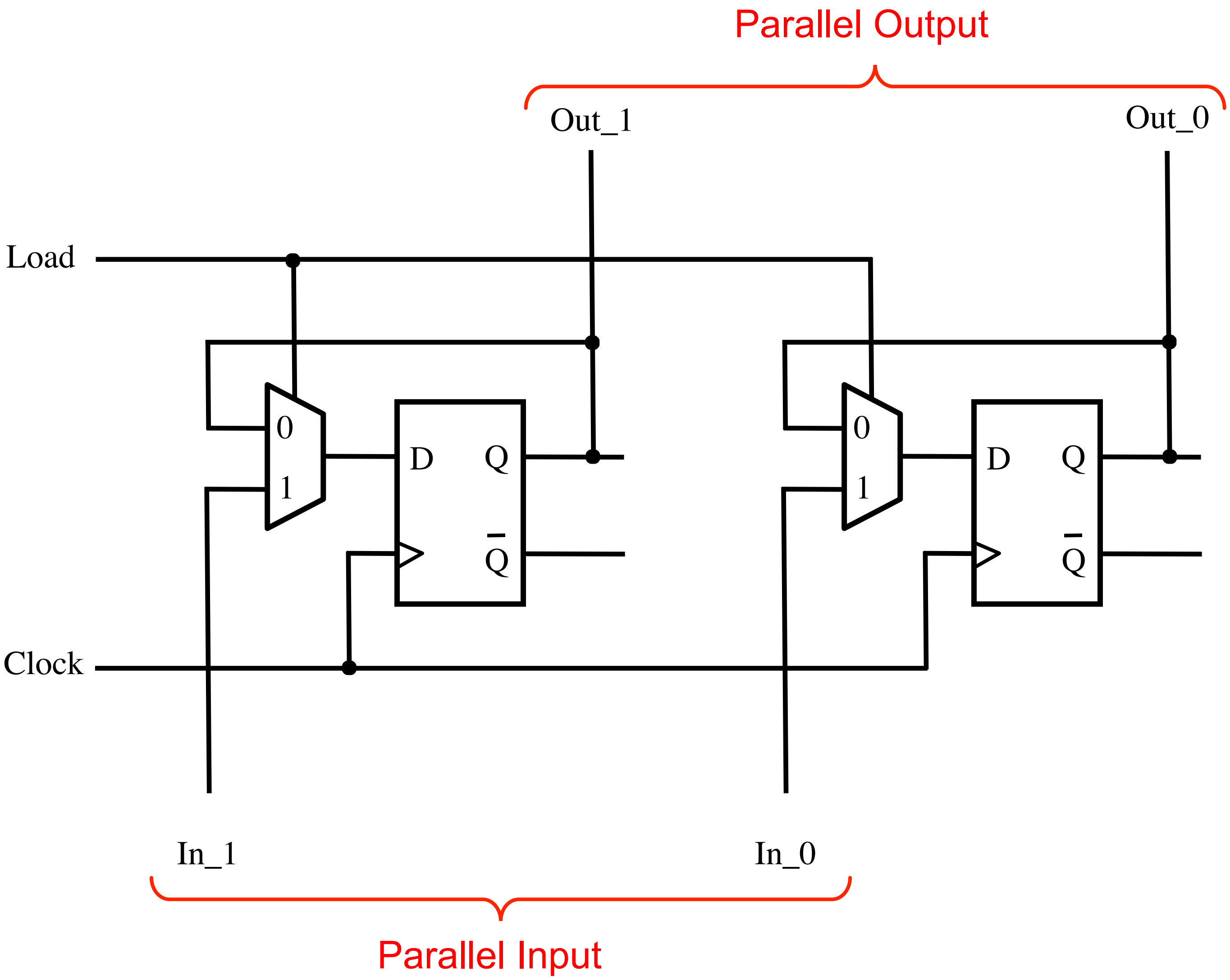
If Load = 0, then retain the old value.

If Load = 1, then load the new value from In.

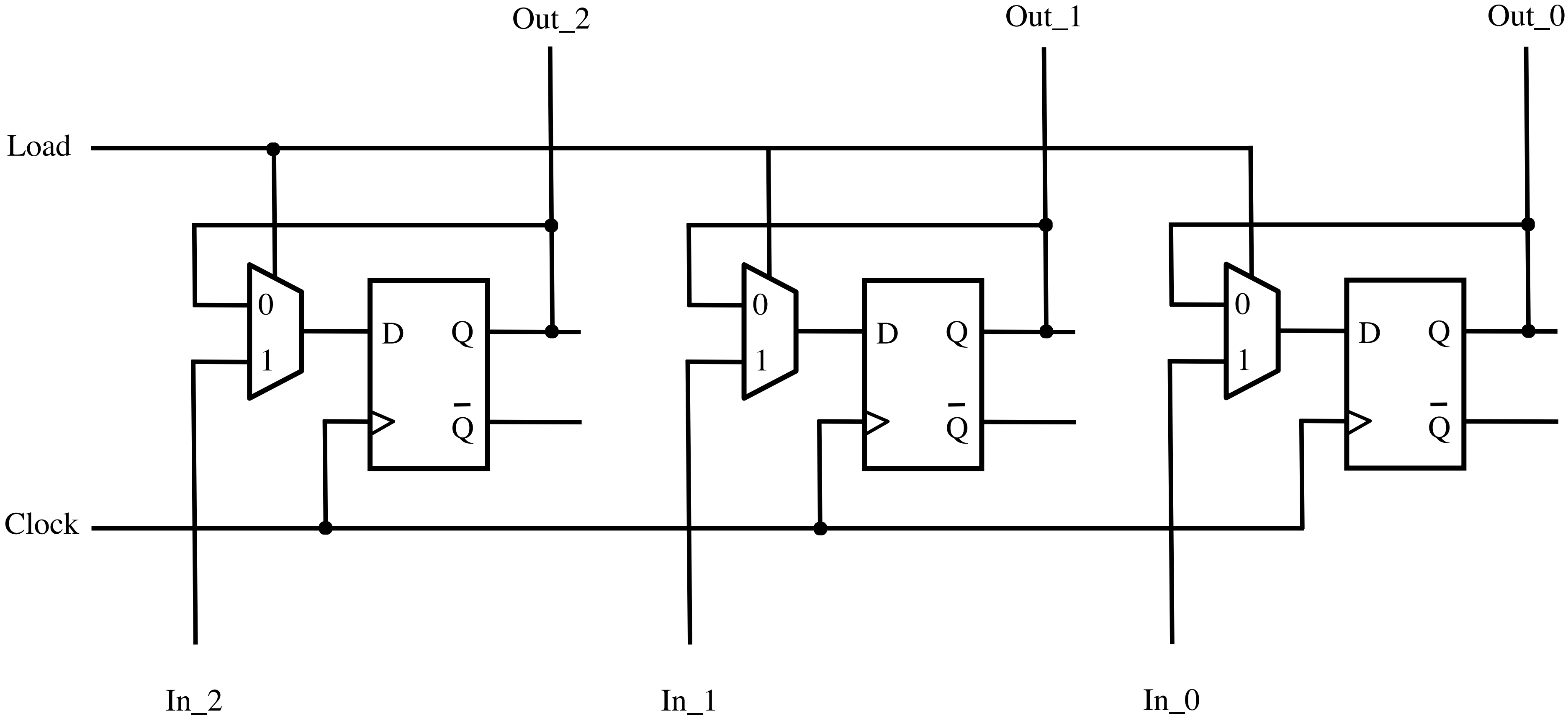
2-Bit Parallel-Access Register



2-Bit Parallel-Access Register

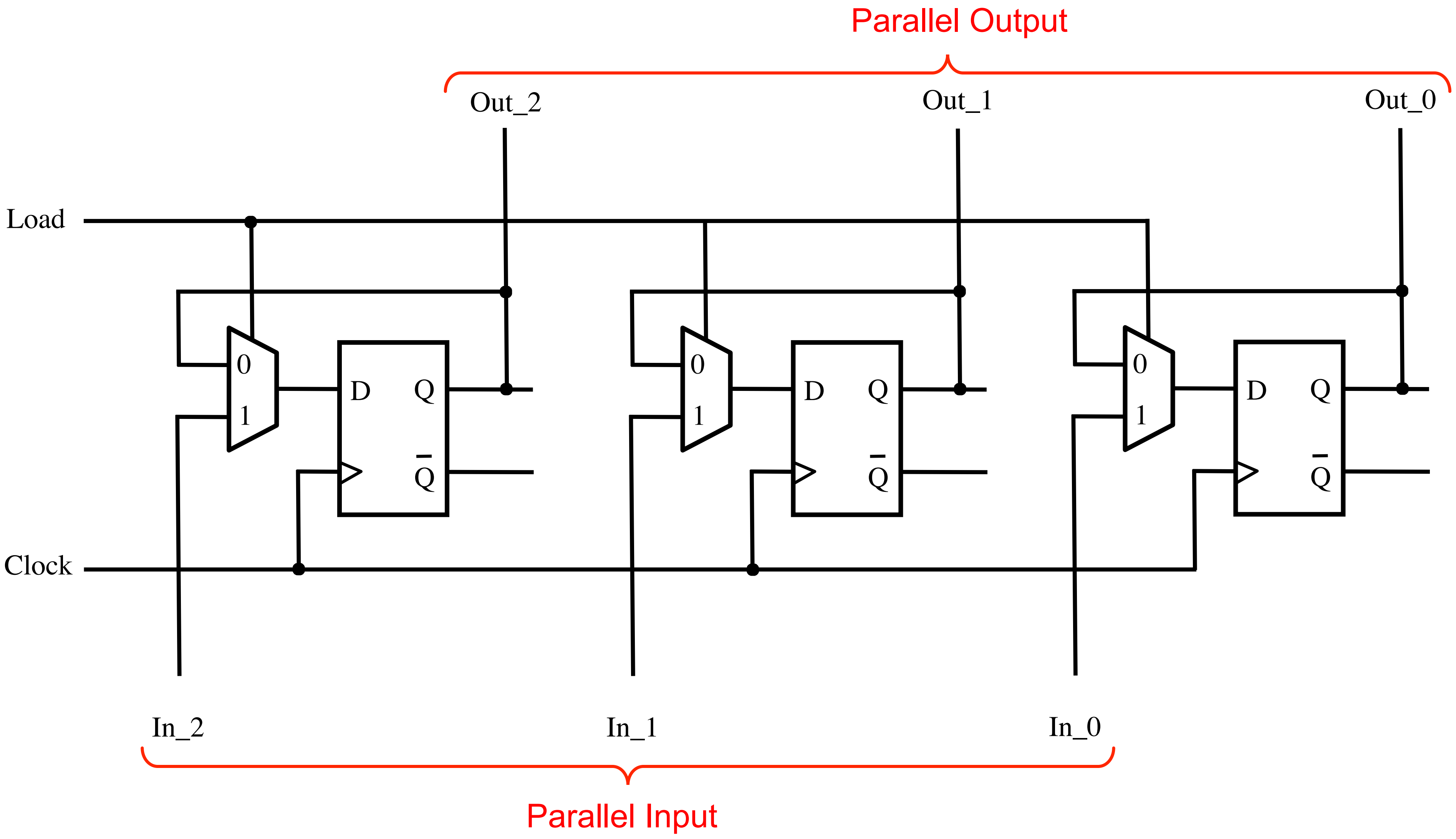


3-Bit Parallel-Access Register

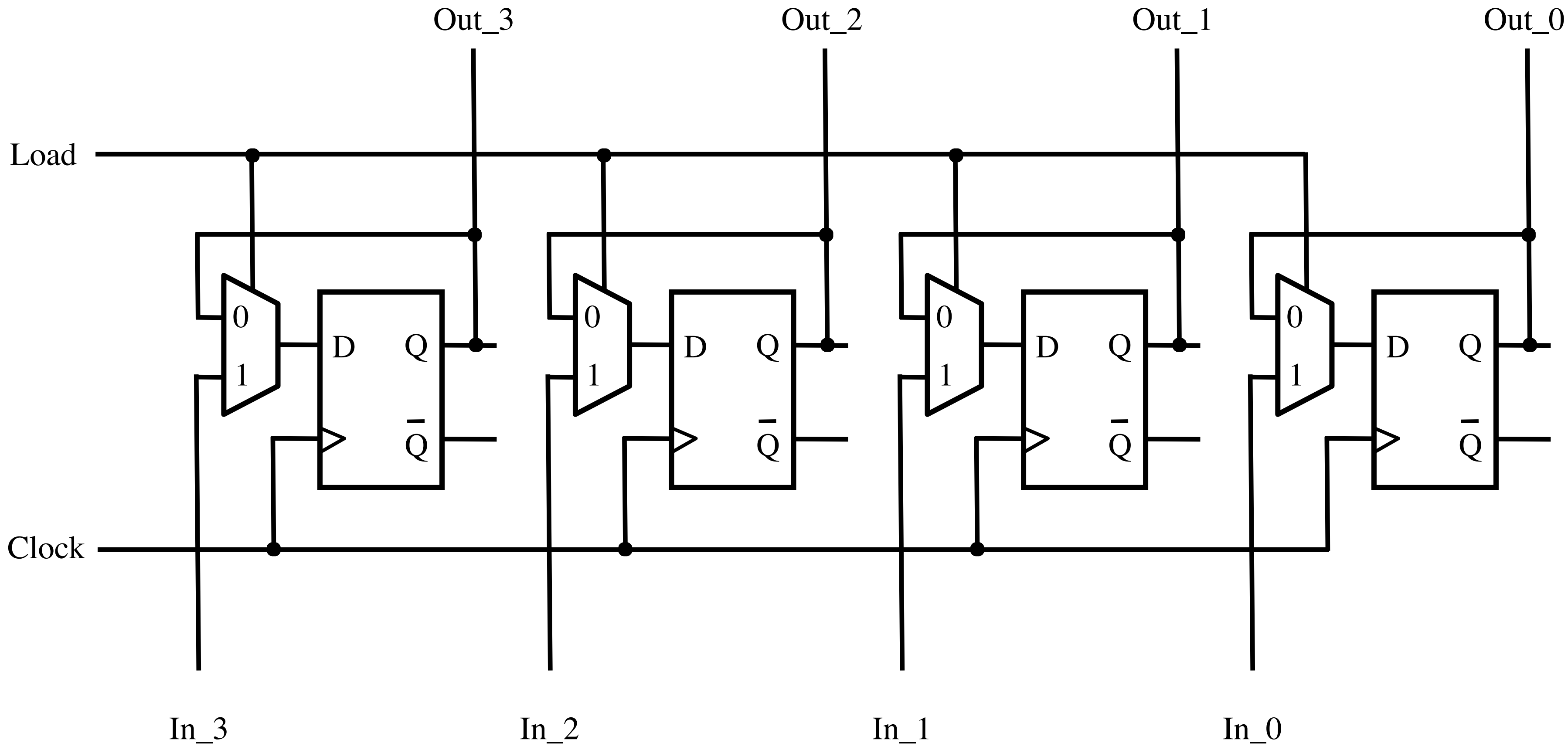


Notice that all flip-flops are on the same clock cycle.

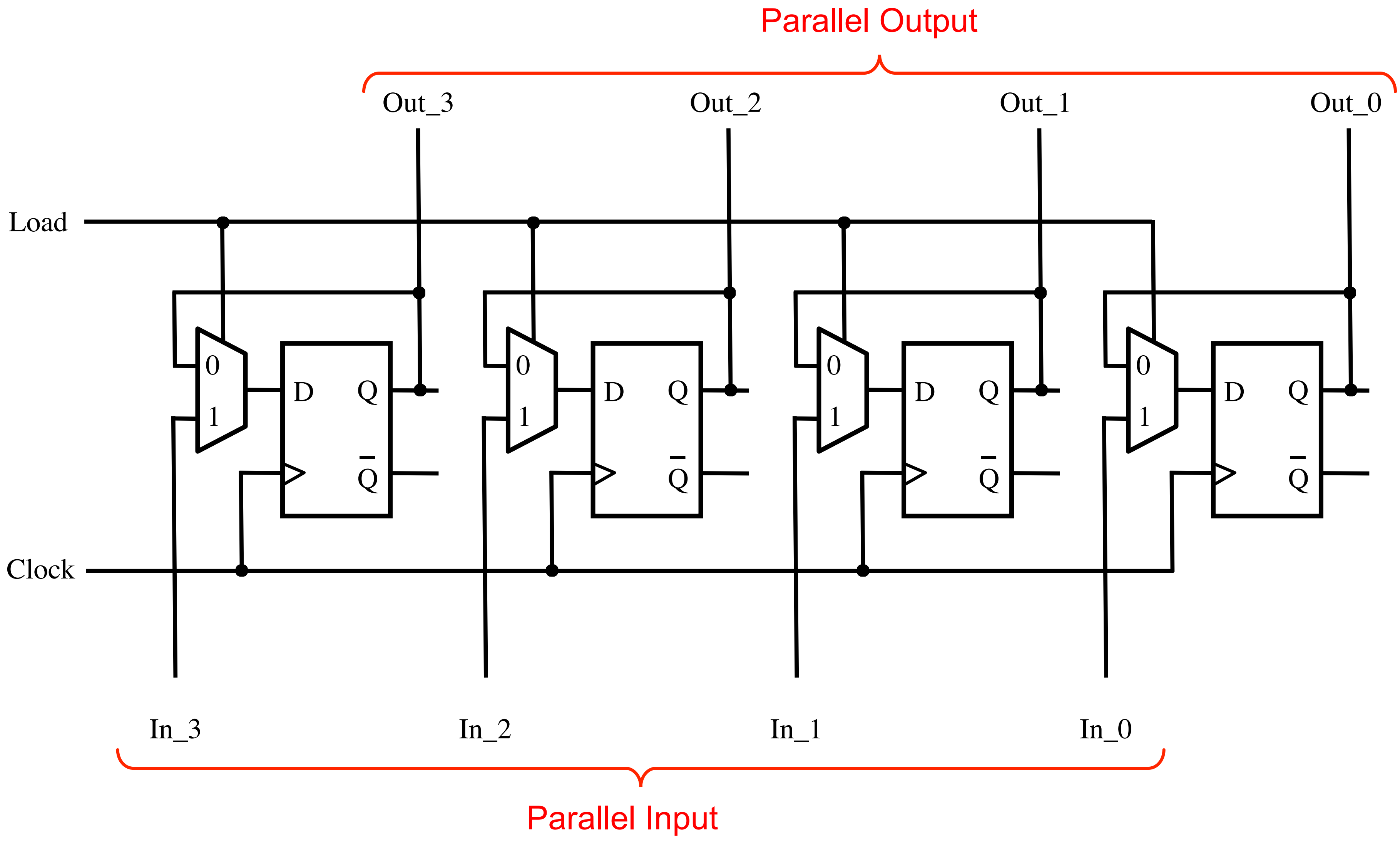
3-Bit Parallel-Access Register



4-Bit Parallel-Access Register



4-Bit Parallel-Access Register

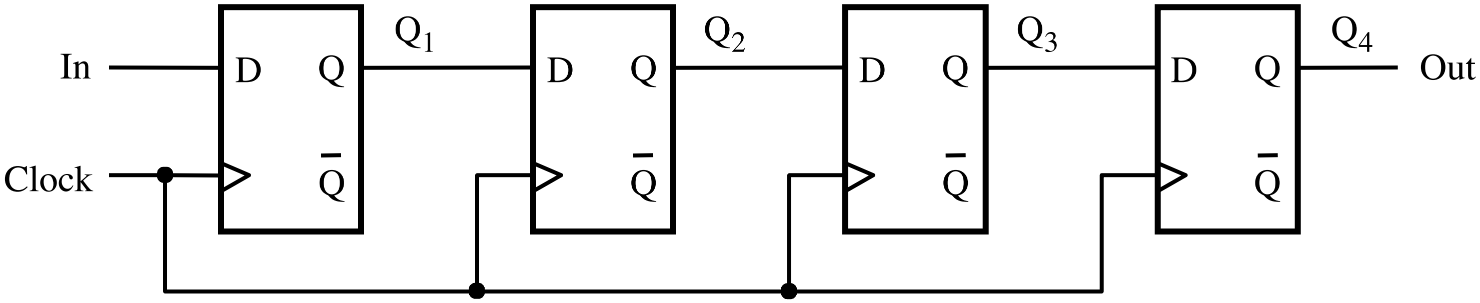


Parallel Output

Parallel Input

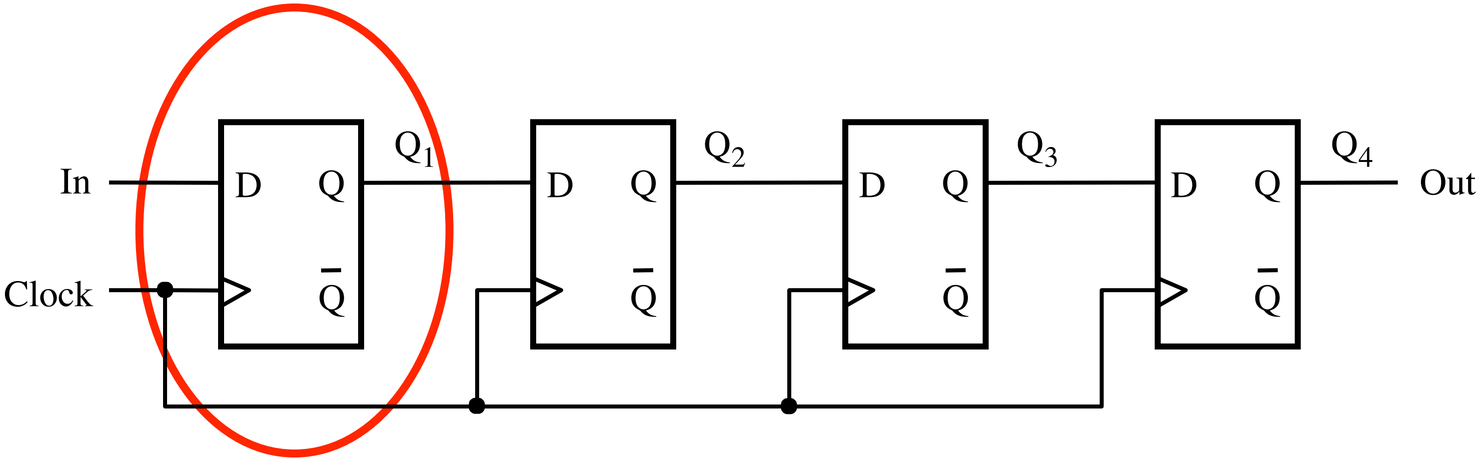
Shift Register

A simple shift register



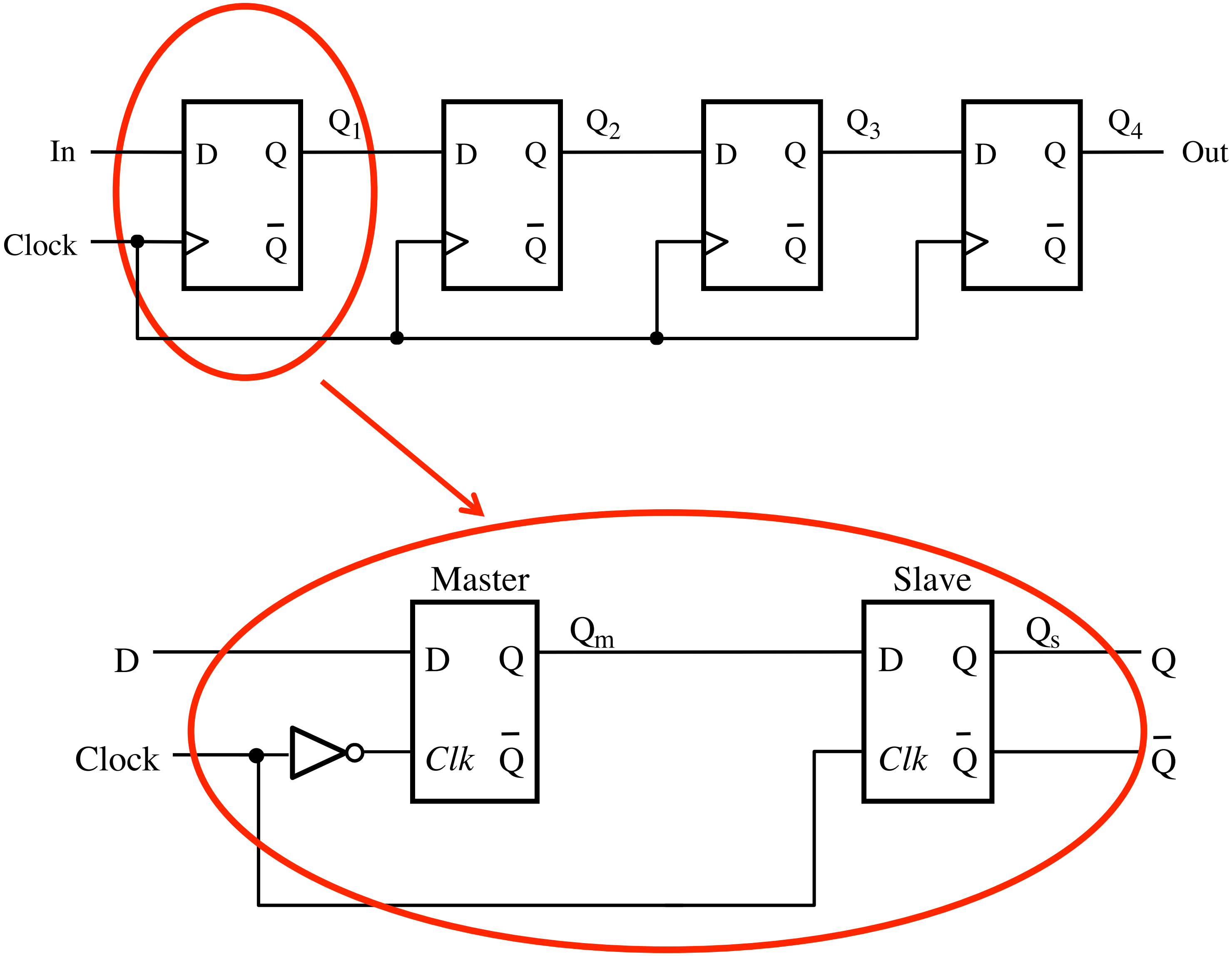
[Figure 5.17a from the textbook]

A simple shift register

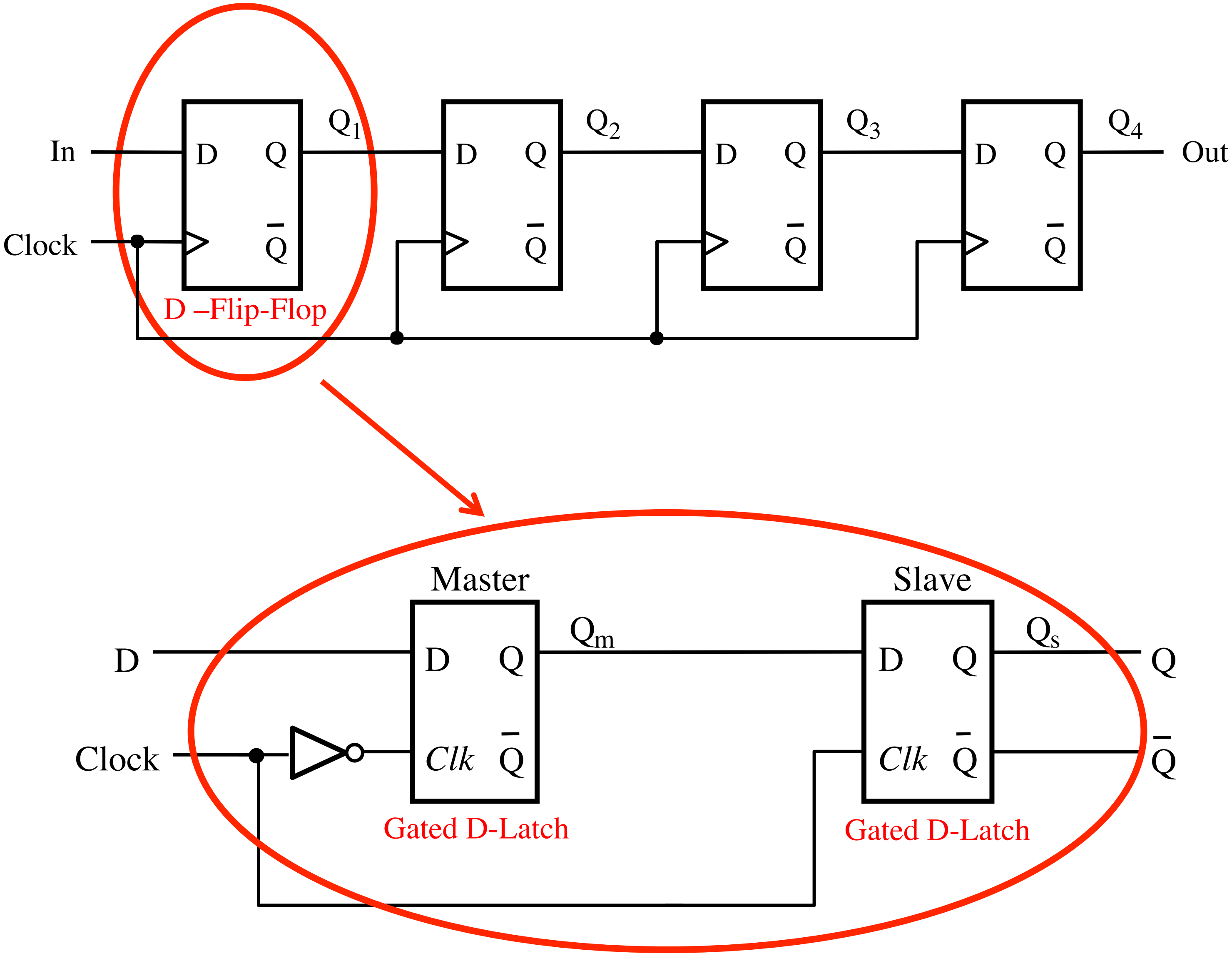


Positive-edge-triggered
D Flip-Flop

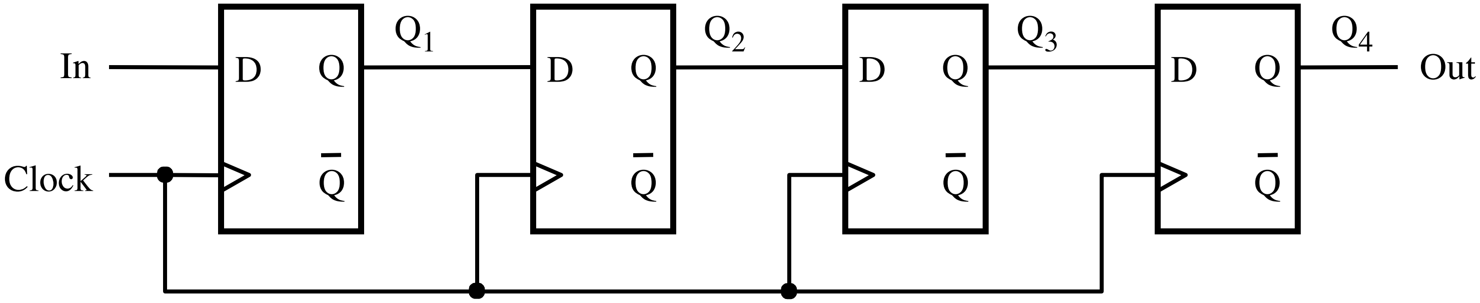
A simple shift register



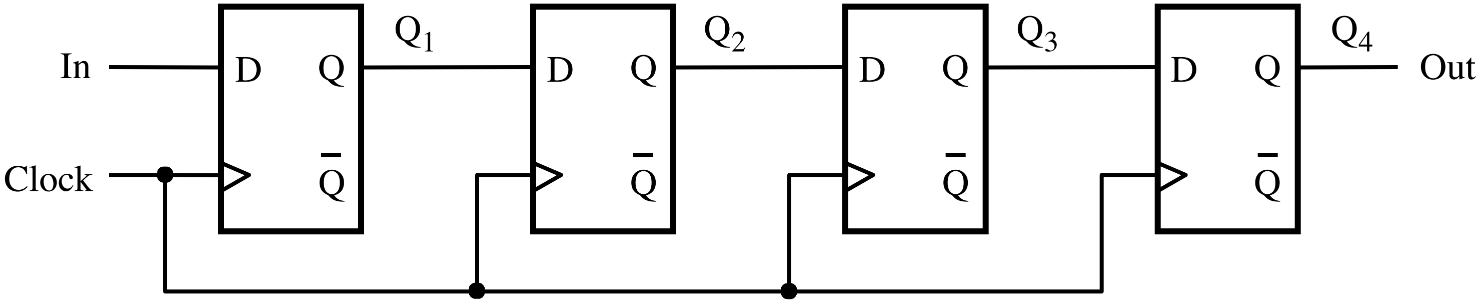
A simple shift register

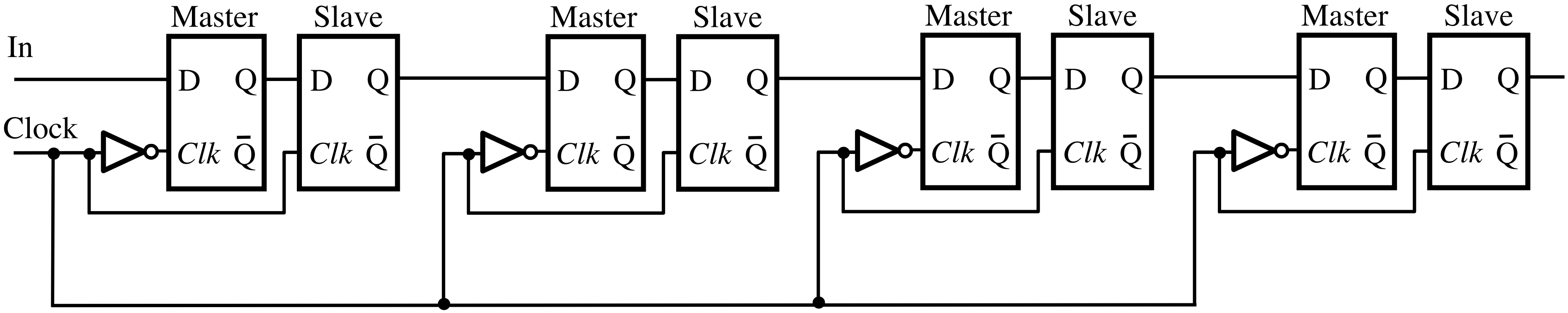


A simple shift register

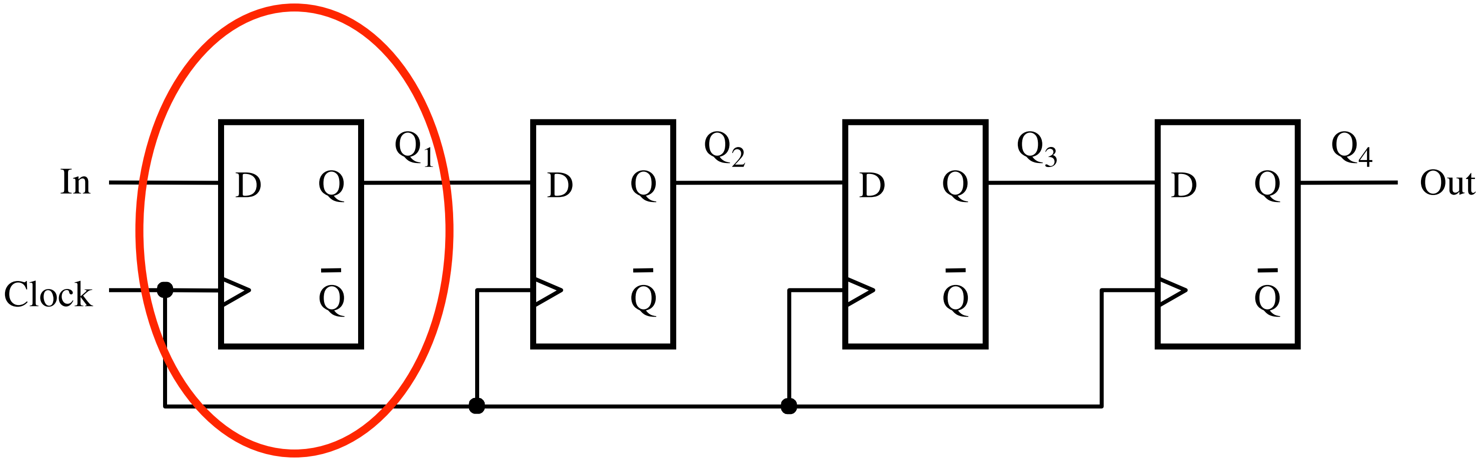


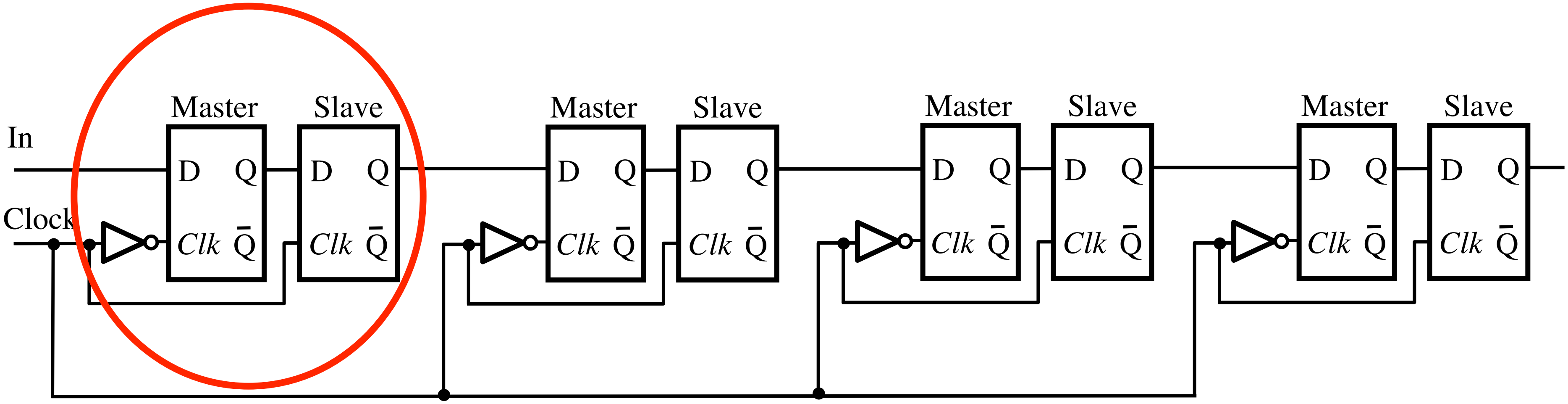
A simple shift register



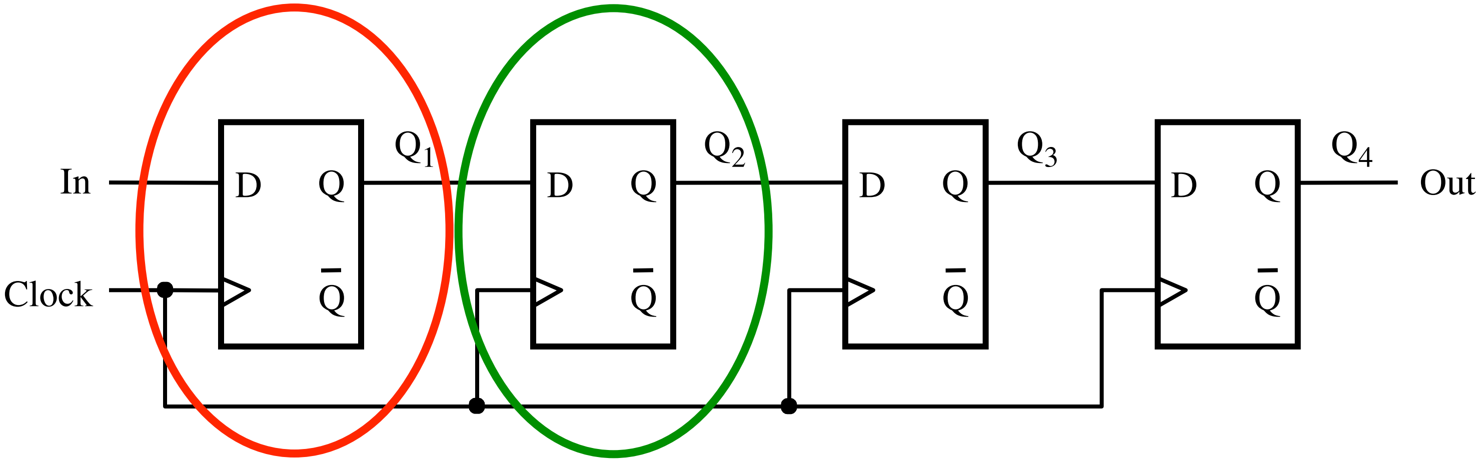


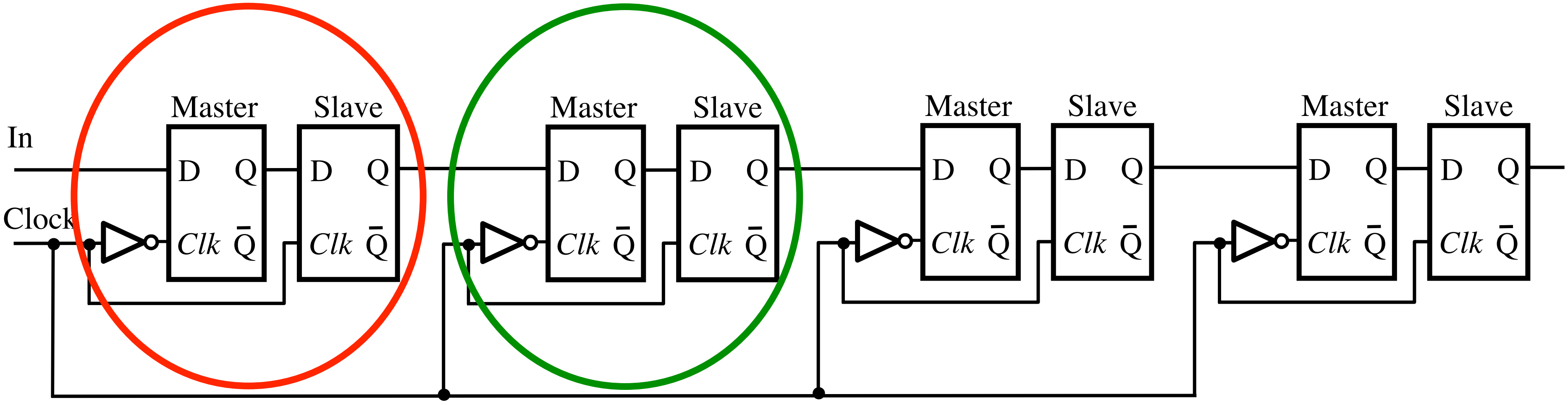
A simple shift register



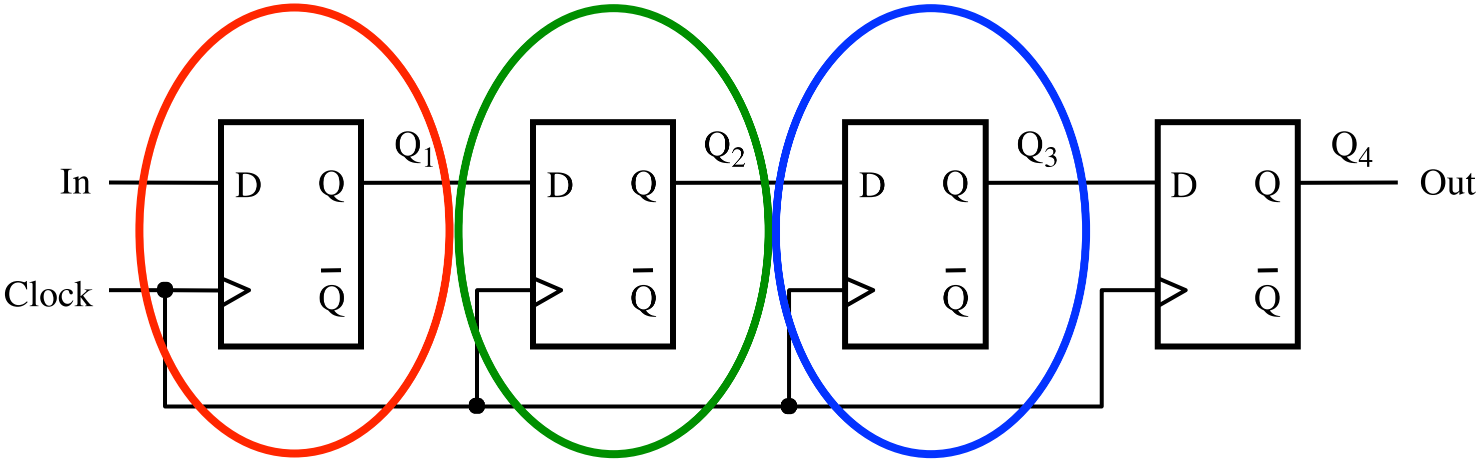


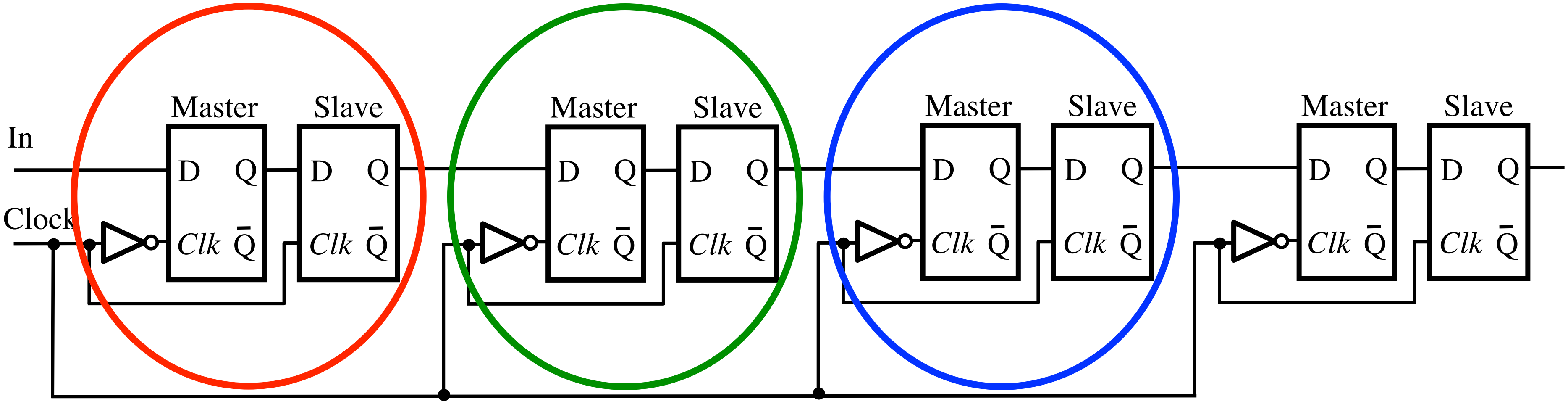
A simple shift register



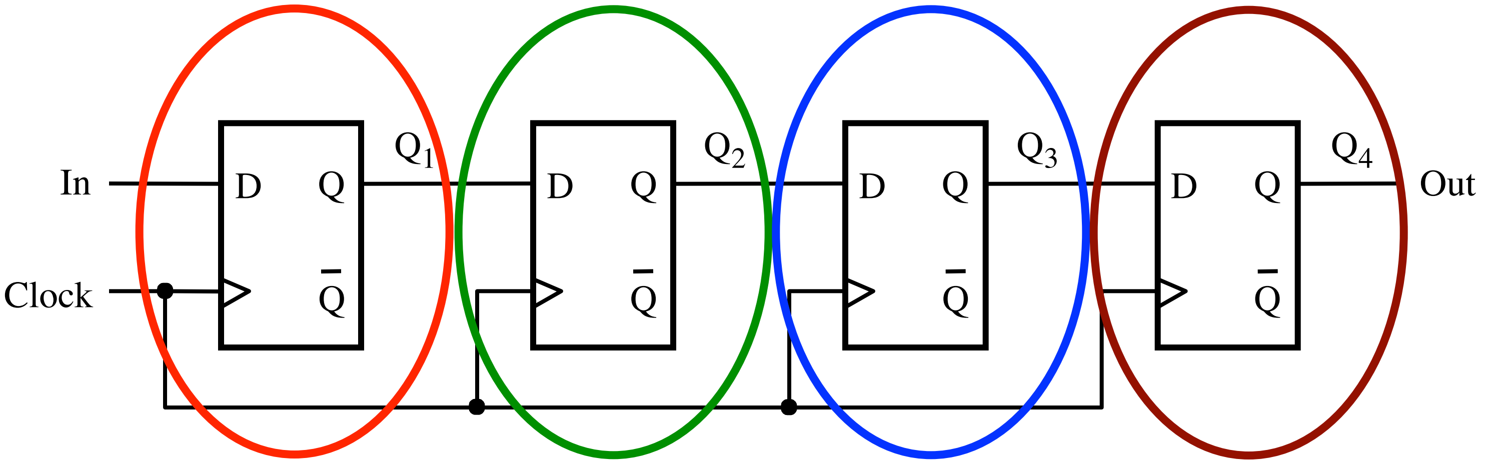


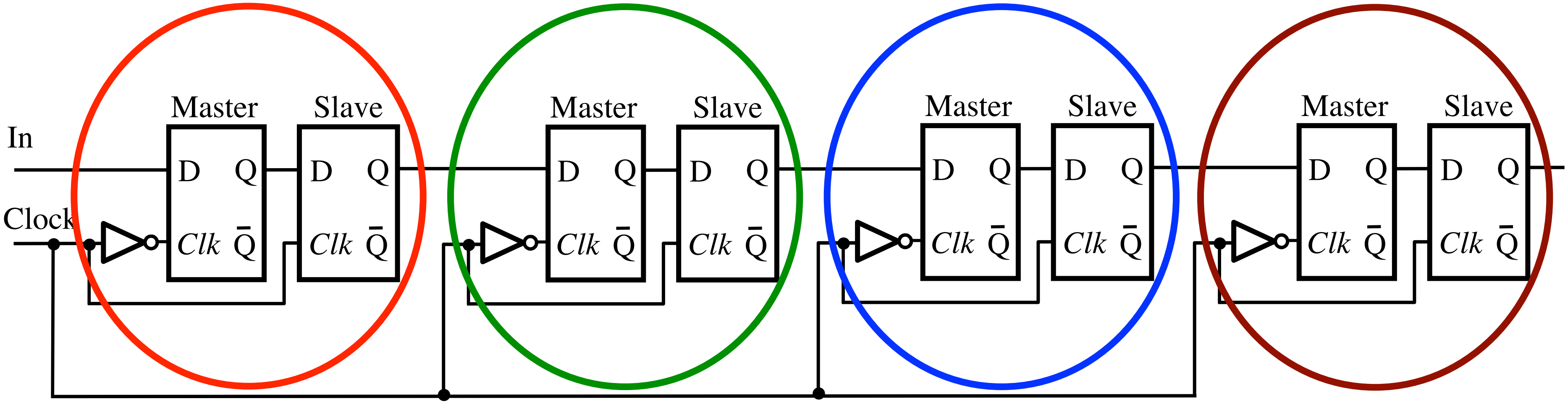
A simple shift register



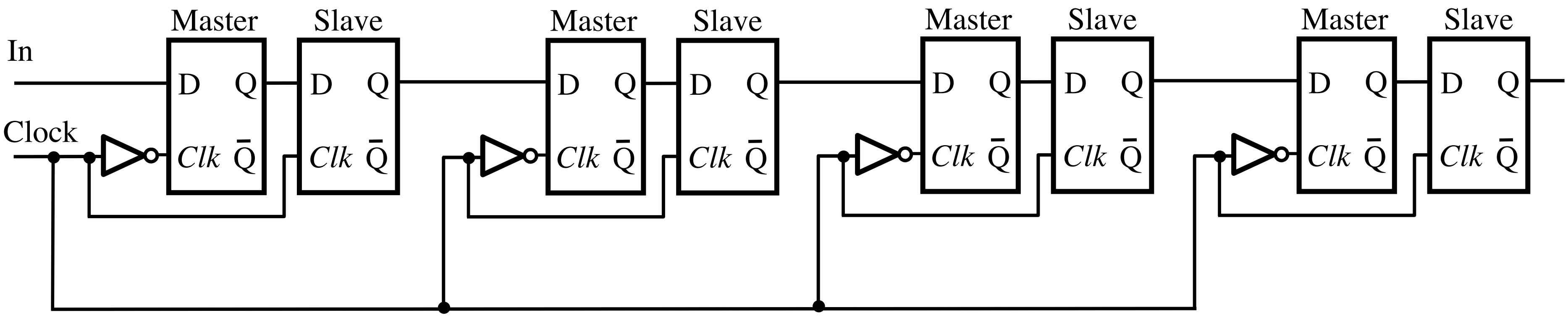


A simple shift register

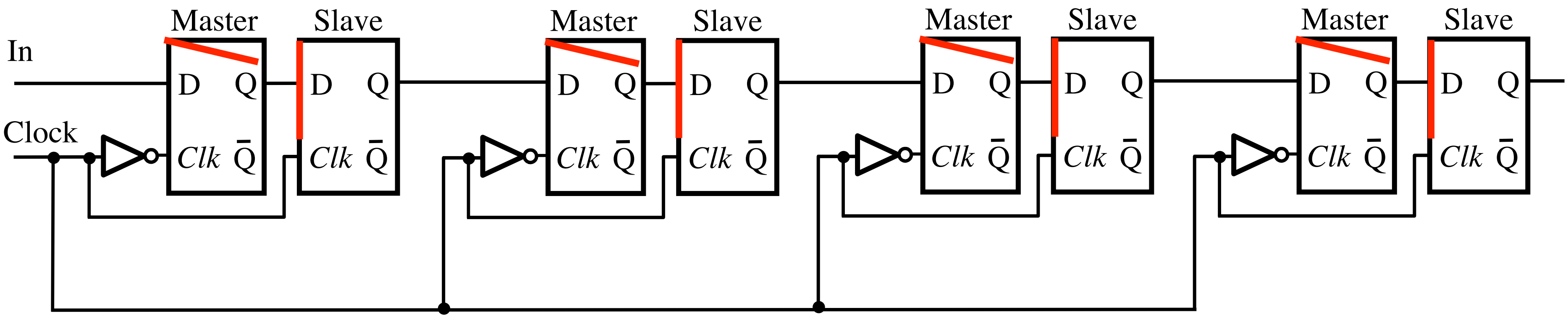


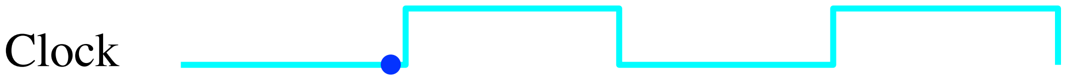


A simple shift register

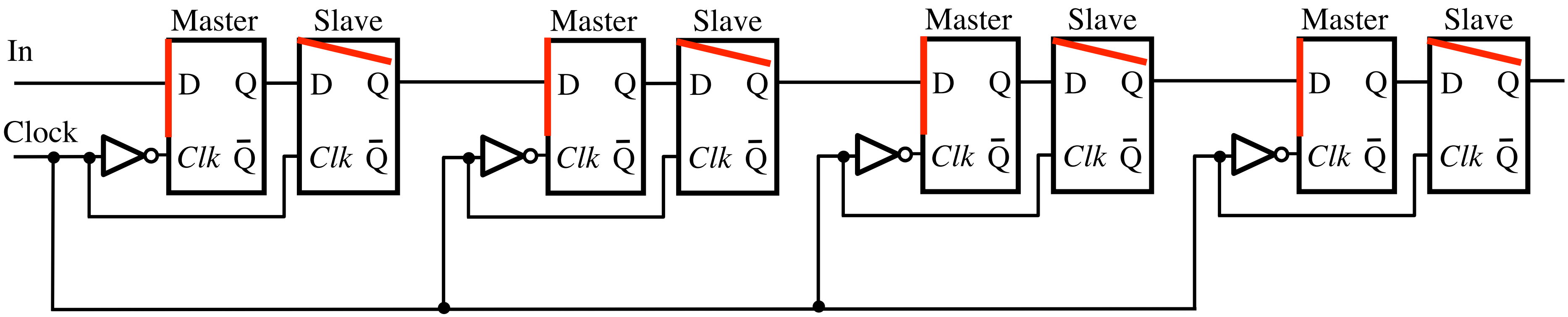


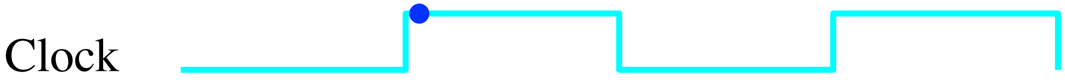
A simple shift register



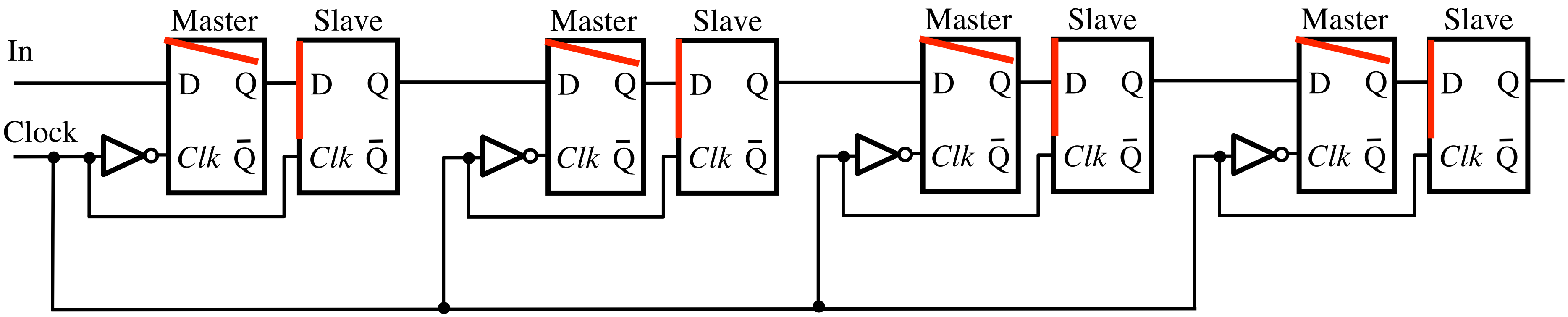


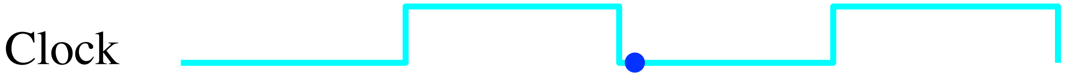
A simple shift register



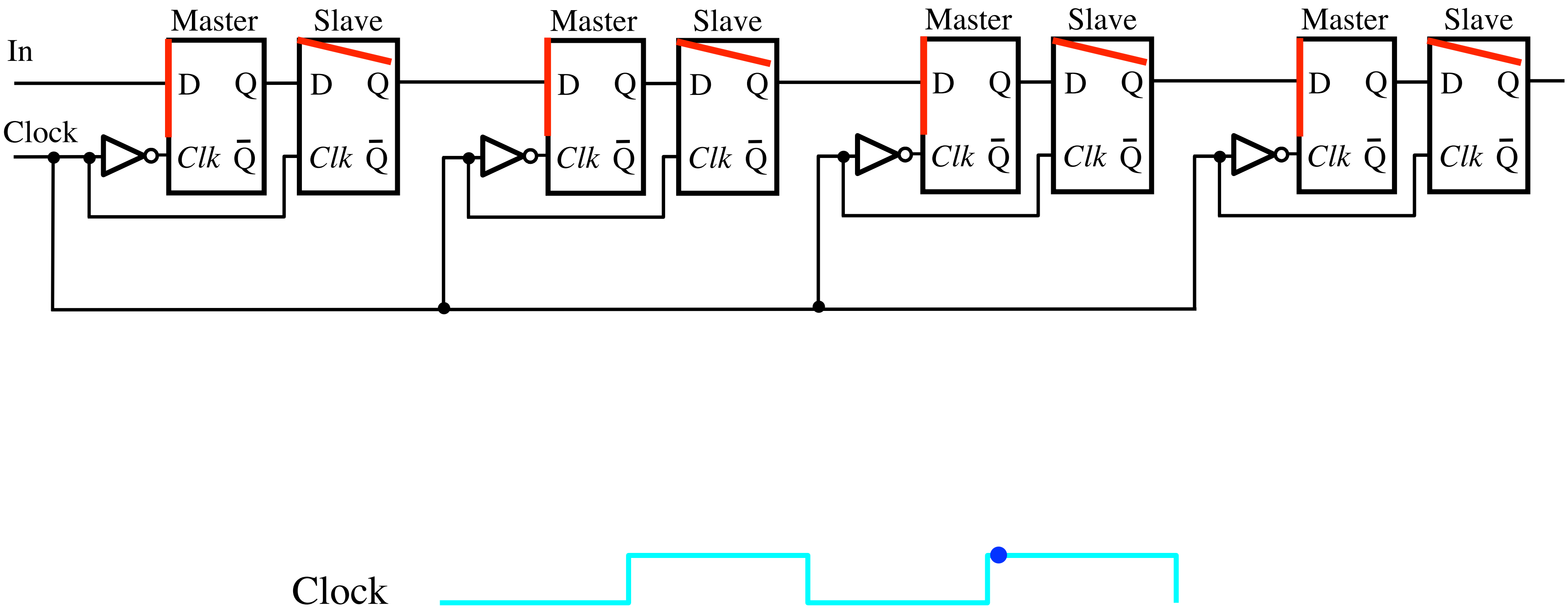


A simple shift register

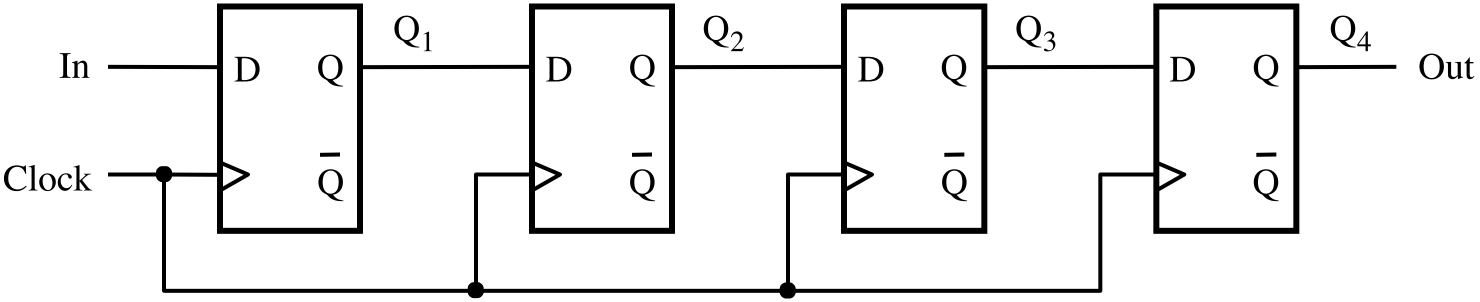




A simple shift register



A simple shift register



(a) Circuit

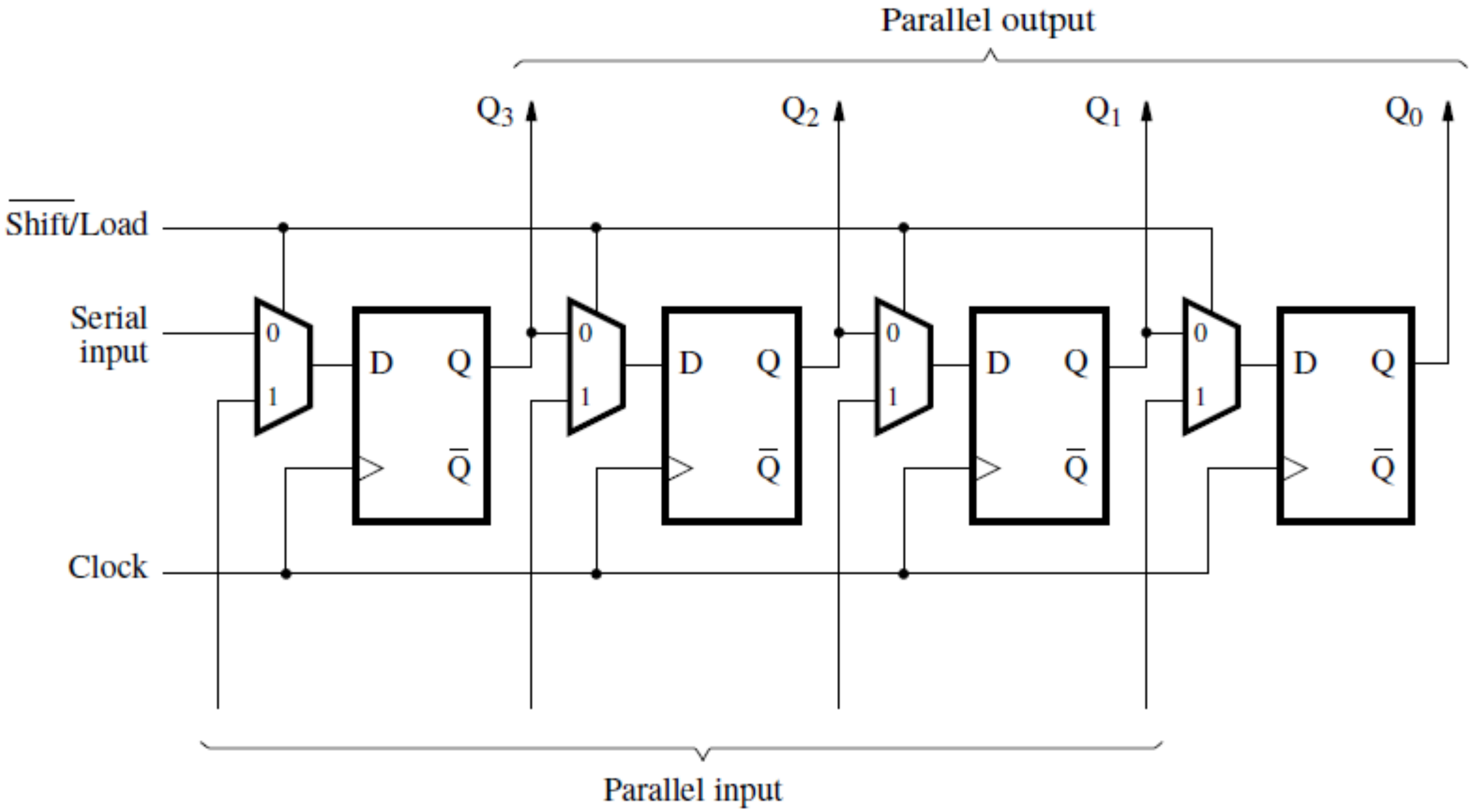
	In	Q_1	Q_2	Q_3	Q_4 = Out
t_0	1	0	0	0	0
t_1	0	1	0	0	0
t_2	1	0	1	0	0
t_3	1	1	0	1	0
t_4	1	1	1	0	1
t_5	0	1	1	1	0
t_6	0	0	1	1	1
t_7	0	0	0	1	1

(b) A sample sequence

[Figure 5.17 from the textbook]

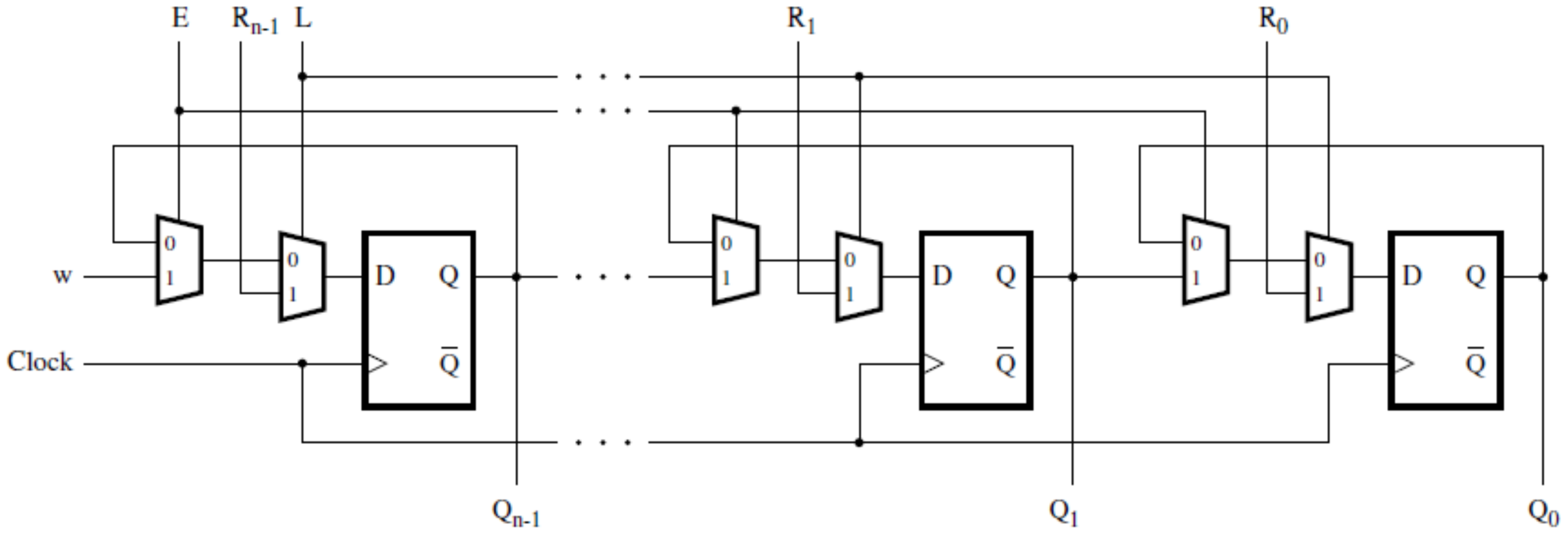
Parallel-Access Shift Register

Parallel-access shift register



[Figure 5.18 from the textbook]

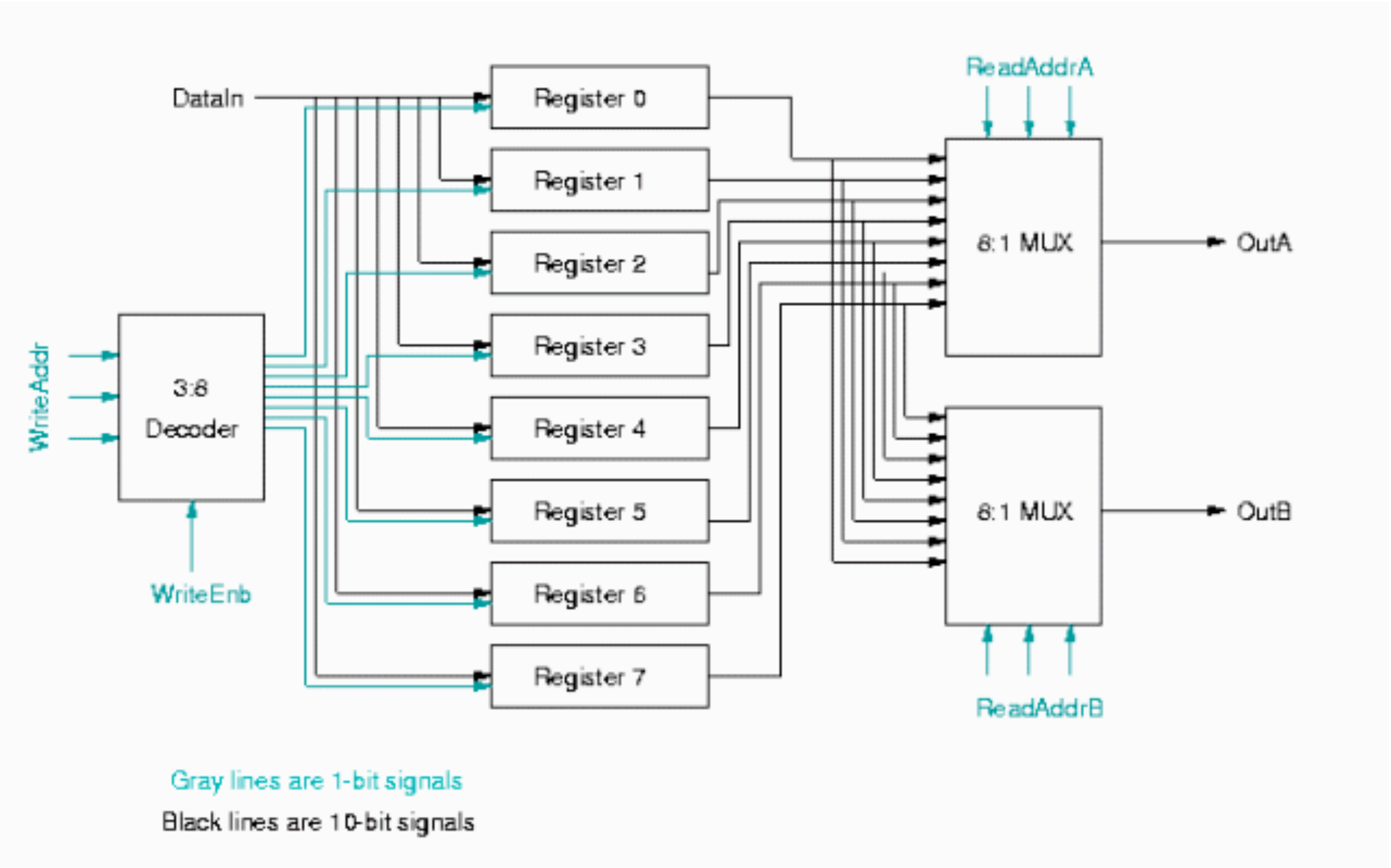
A shift register with parallel load and enable control inputs



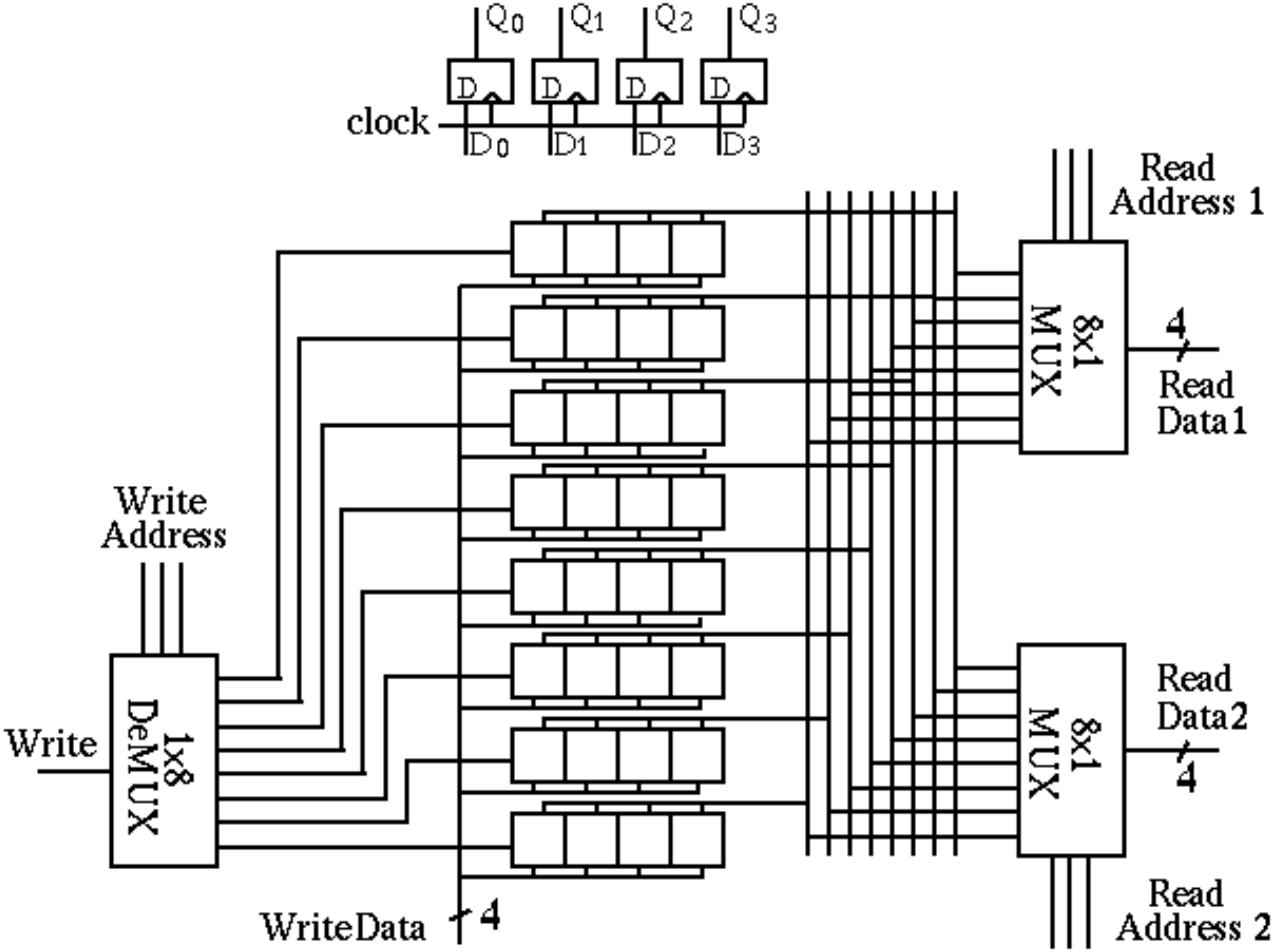
[Figure 5.59 from the textbook]

Register File

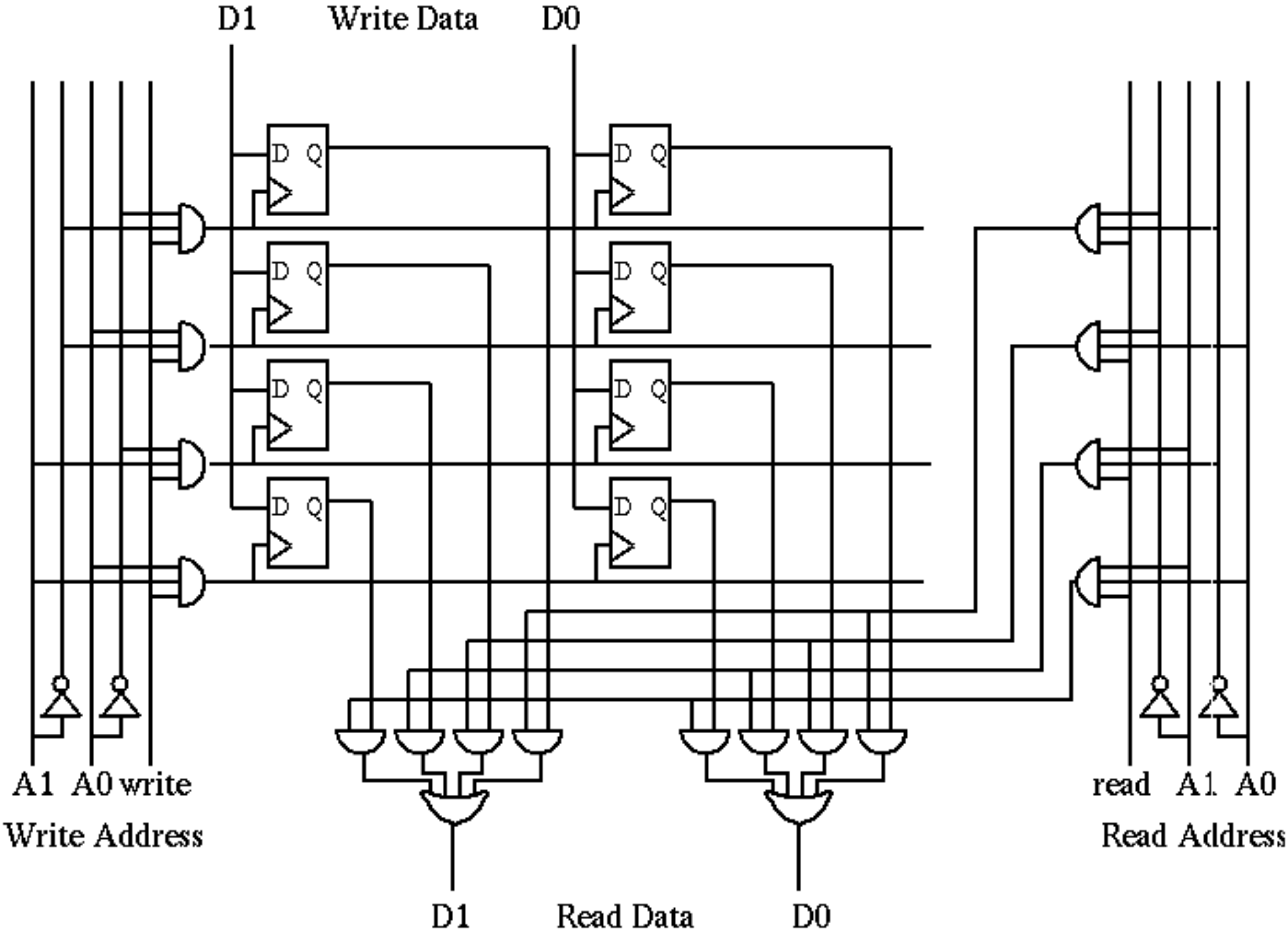
Register File



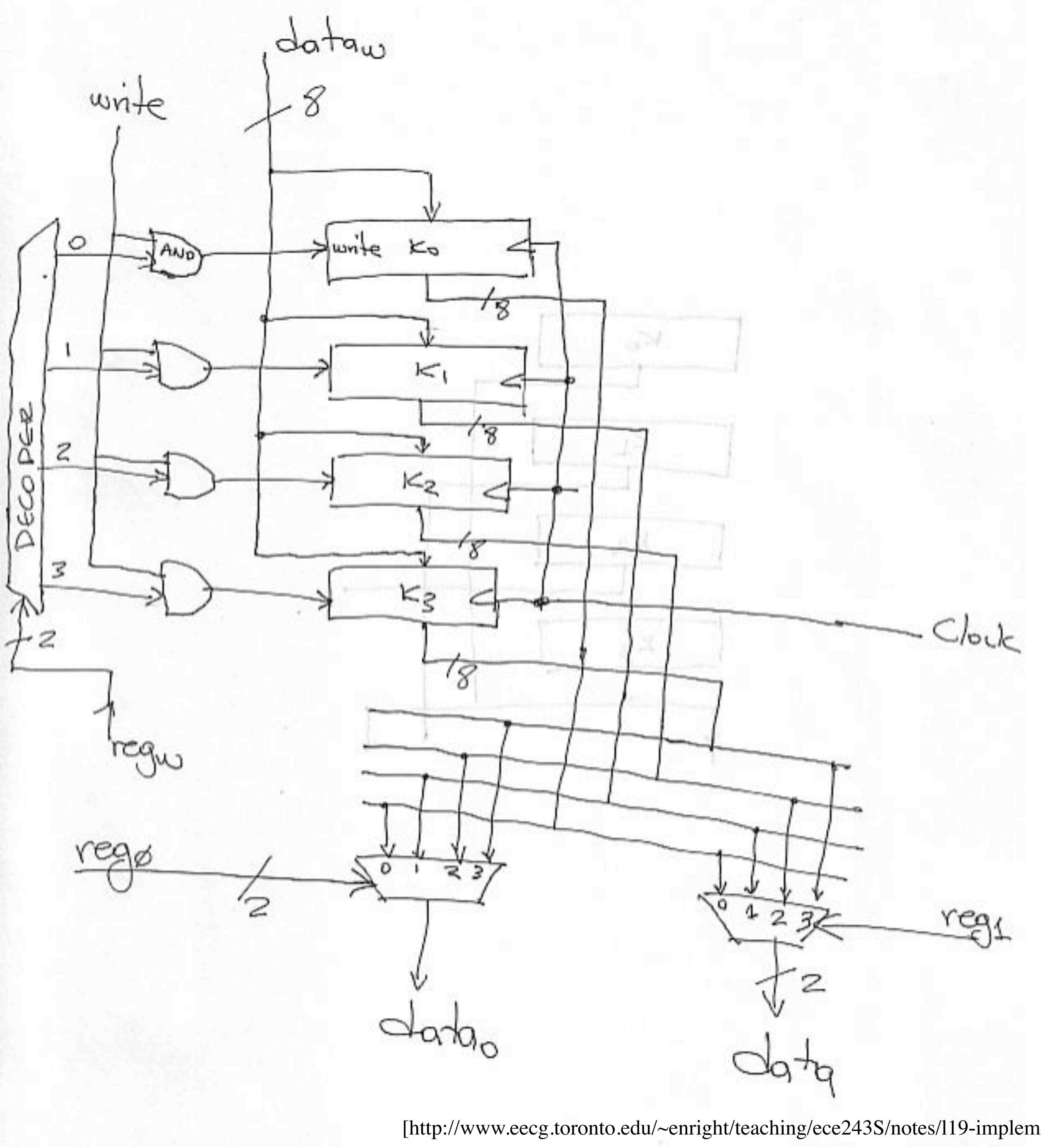
[https://jindongpu.wordpress.com/2012/03/07/register-file/]



[http://fourier.eng.hmc.edu/e85_old/lectures/digital_logic/node19.html]



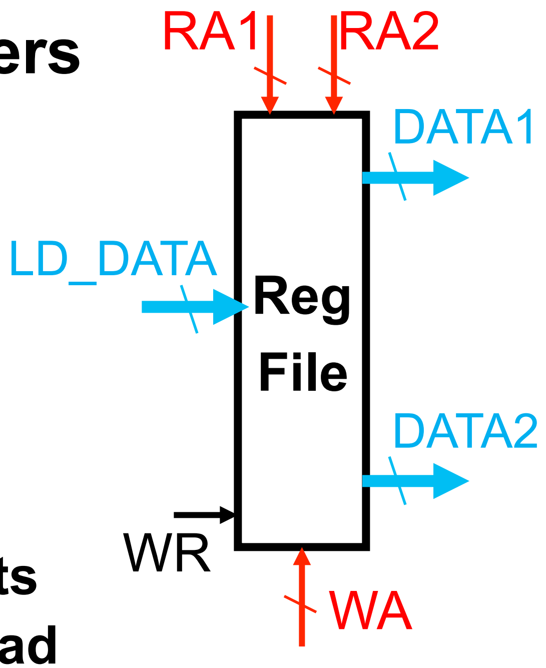
[http://fourier.eng.hmc.edu/e85_old/lectures/digital_logic/node19.html]



[http://www.eecg.toronto.edu/~enright/teaching/ece243S/notes/l19-implemenation-single-cycle.html]

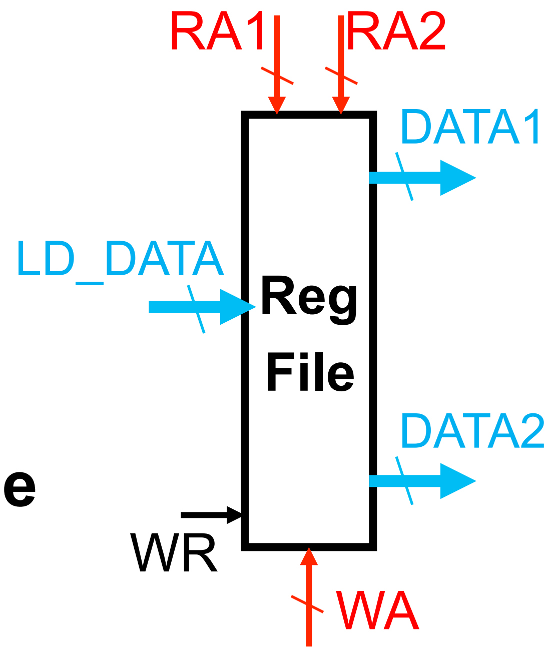
Register File

- Register file is a unit containing r registers
  - r can be 4, 8, 16, 32, etc.
- Each register has n bits
  - n can be 4, 8, 16, 32, etc.
  - n defines the data path width
- Output ports (DATA1 and DATA2) are used for reading the register file
  - Any register can be read from any of the ports
  - Each port needs a $\log_2 r$ bits to specify the read address (RA1 and RA2)
- Input port (LD_DATA) is used for writing data to the register file
  - Write address is also specified by $\log_2 r$ bits (WA)
  - Writing is enabled by a 1-bit signal (WR)



Register File: Exercise

- **Suppose that a register file**
  - **contains 32 registers**
  - **width of data path is 16 bits (i.e., each register has 16 bits)**

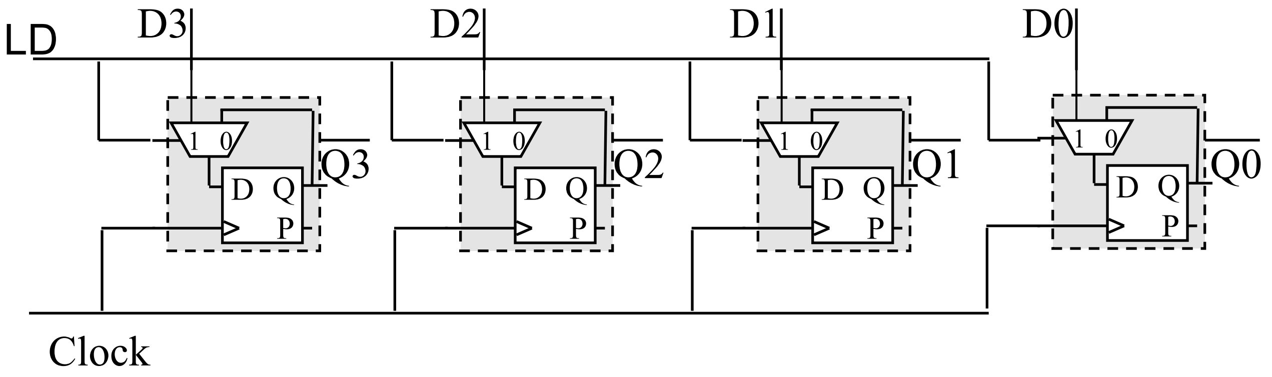


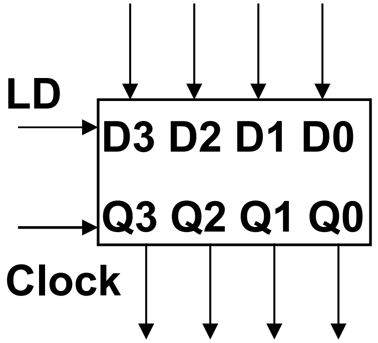
- **How many bits are there for each of the signals?**

RA1	**5**
RA2	**5**
DATA1	**16**
DATA2	**16**
WA	**5**
LD_DATA	**16**
WR	**1**

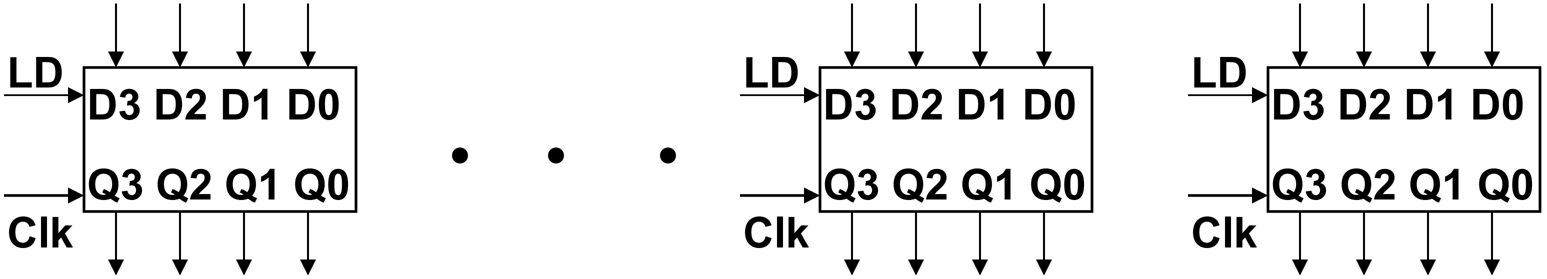
Register file design

- We will design an eight-register file with 4-bit wide registers
- A single 4-bit register and its abstraction are shown below





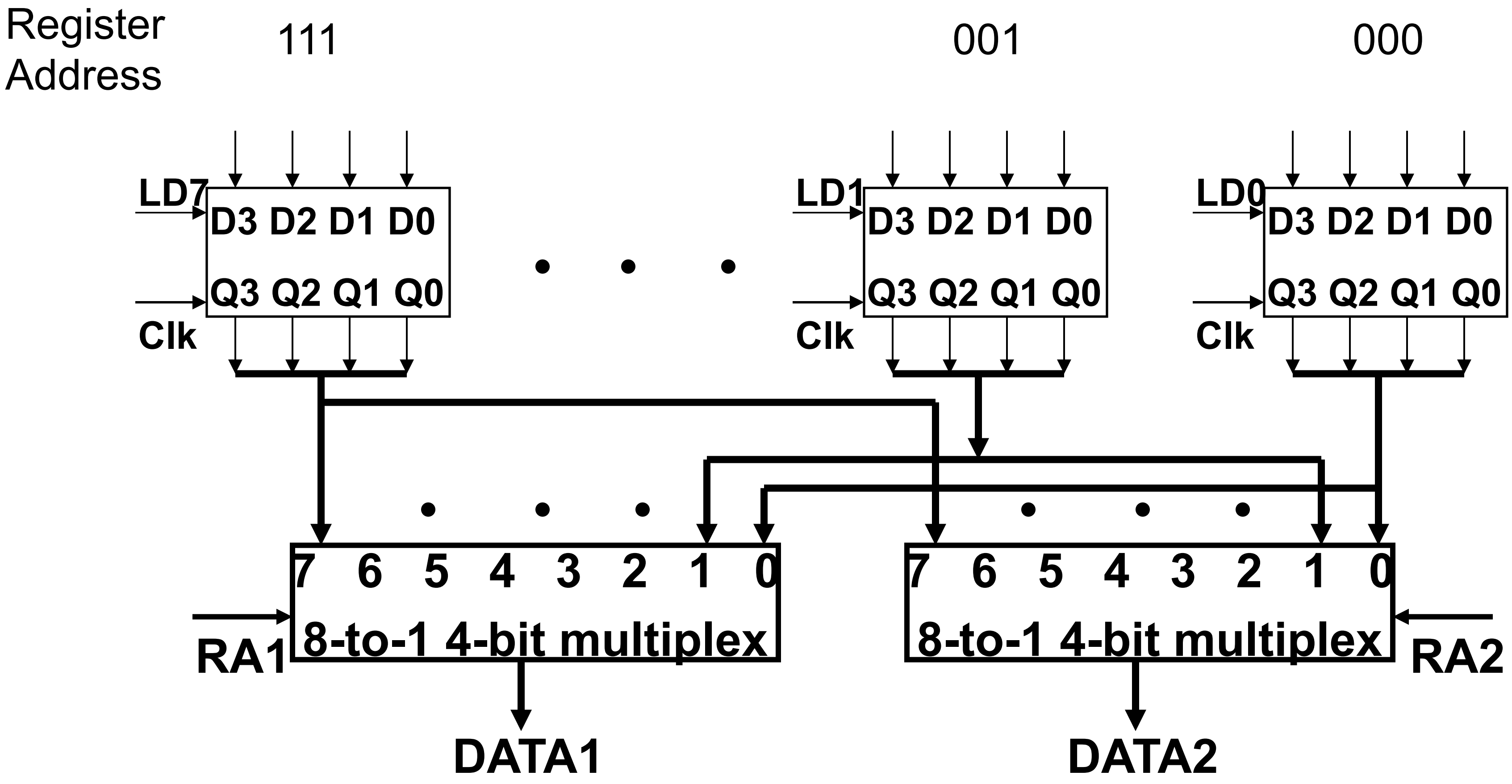
- We have to use eight such registers to make an eight register file



- How many bits are required to specify a register address?

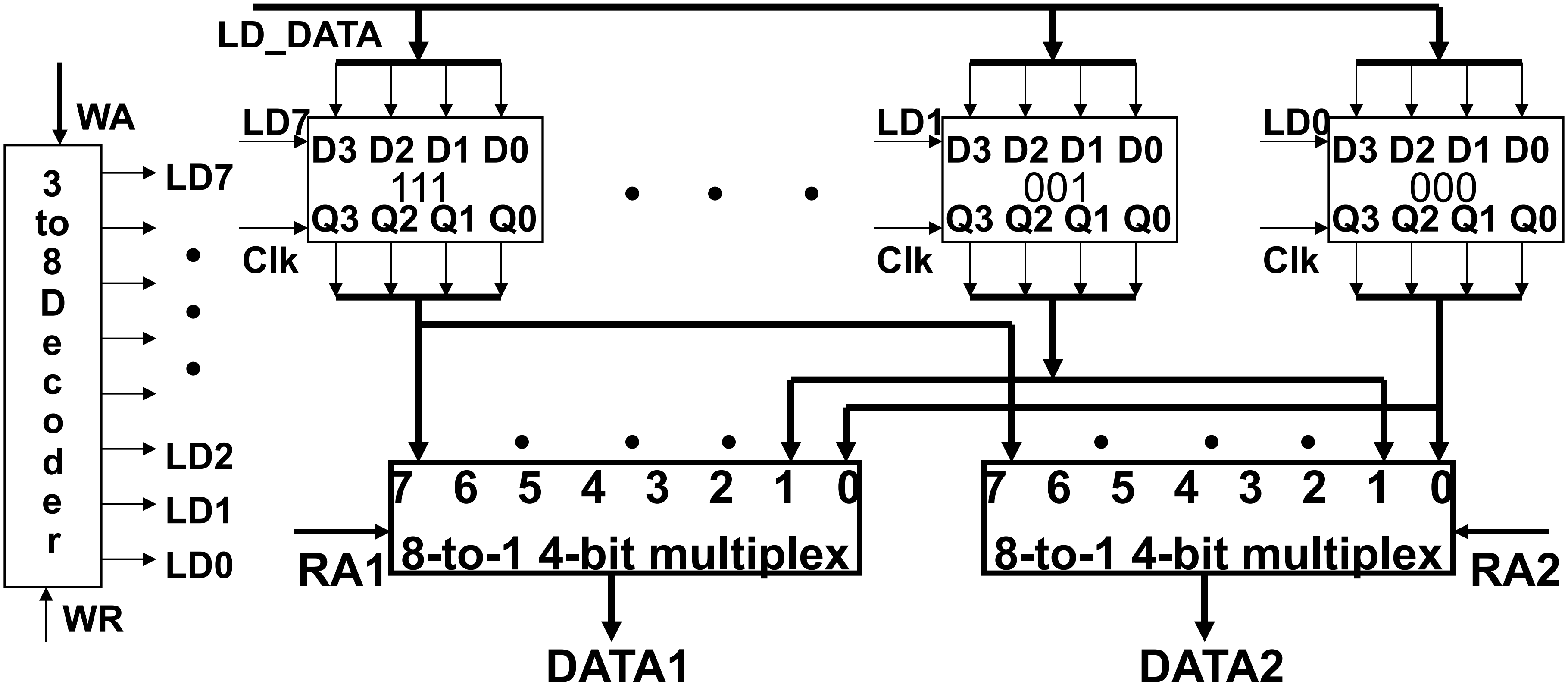
Reading Circuit

- **A 3-bit register address, RA, specifies which register is to be read**
- **For each output port, we need one 8-to-1 4-bit multiplier**



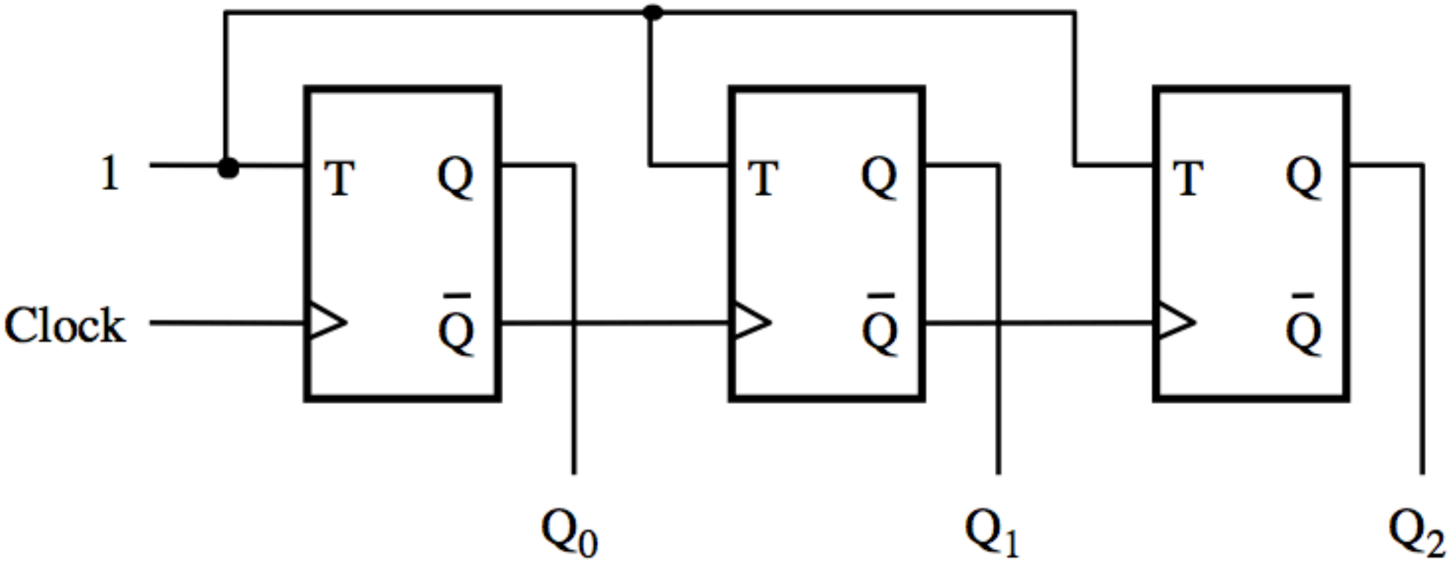
Adding write control to register file

- **To write to any register, we need the register's address (WA) and a write register signal (WR)**
- **A 3-bit write address is decoded if write register signal is present**
- **One of the eight registers gets a LD signal from the decoder**



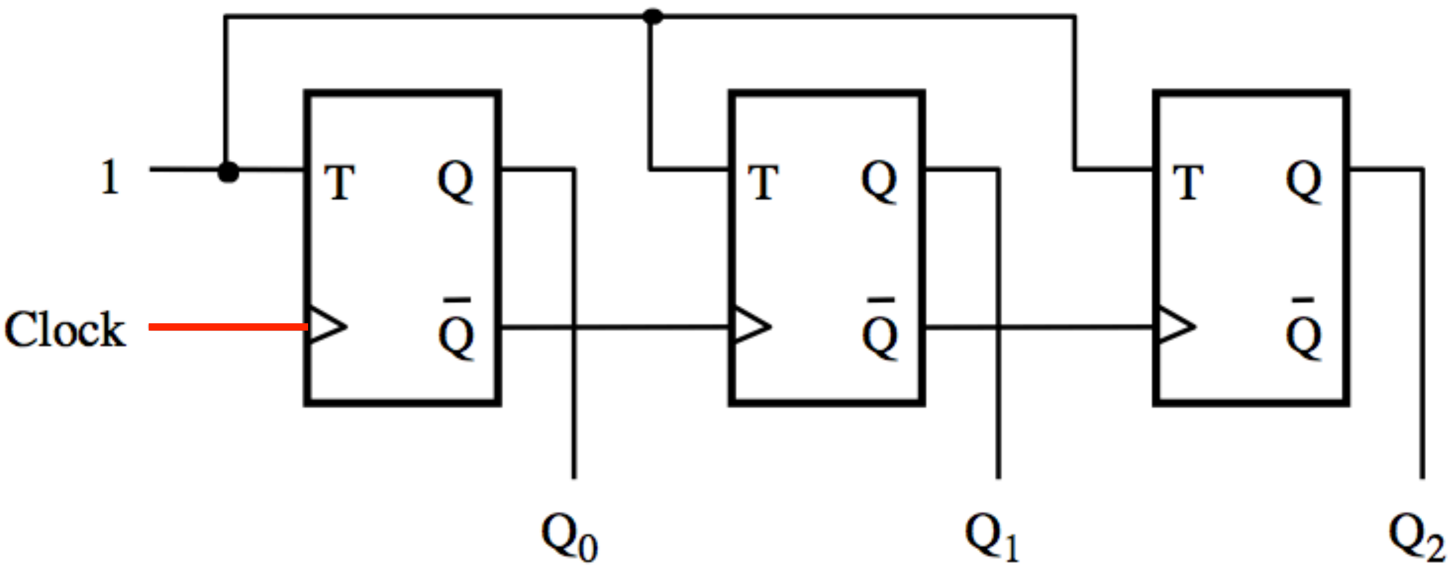
Counters

A three-bit up-counter



[Figure 5.19 from the textbook]

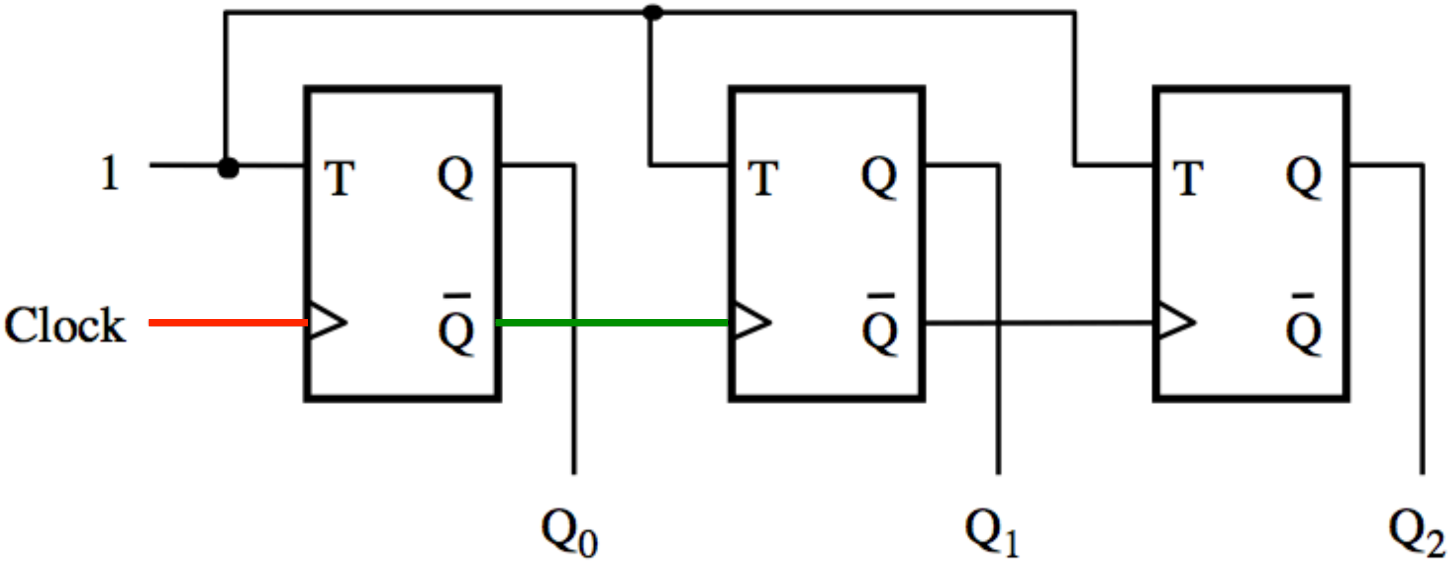
A three-bit up-counter



The first flip-flop changes
on the positive edge of the clock

[Figure 5.19 from the textbook]

A three-bit up-counter

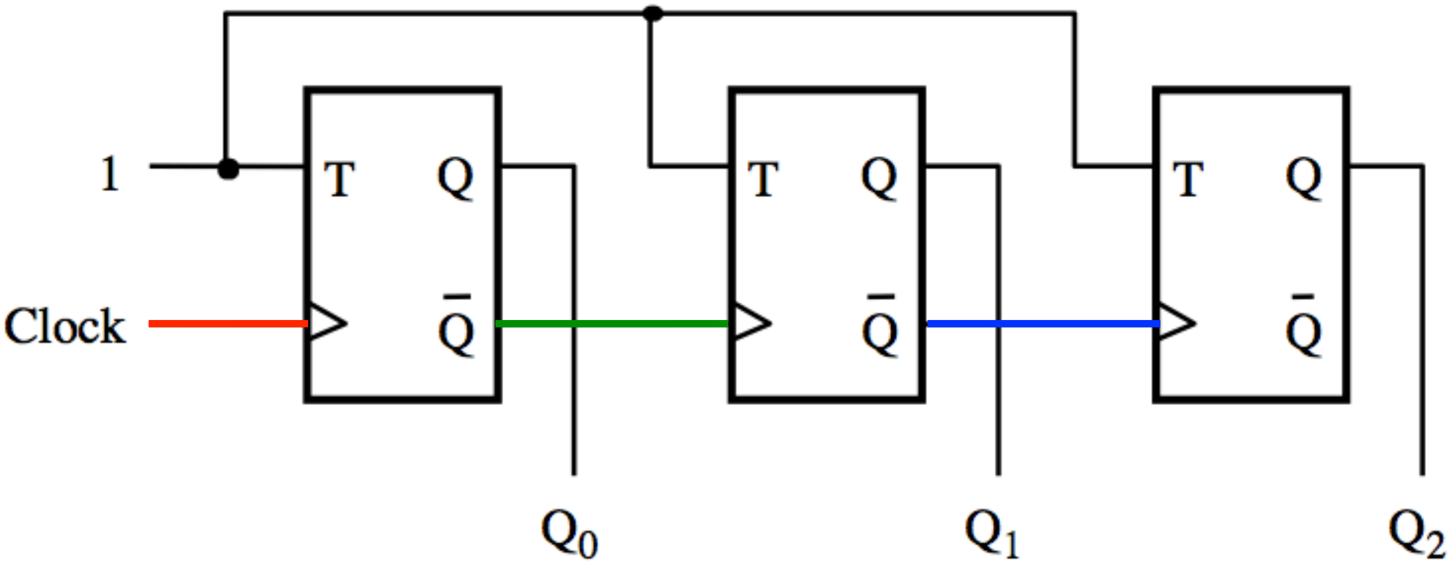


The first flip-flop changes on the positive edge of the clock

The second flip-flop changes on the positive edge of $\overline{Q}_0$

[Figure 5.19 from the textbook]

A three-bit up-counter



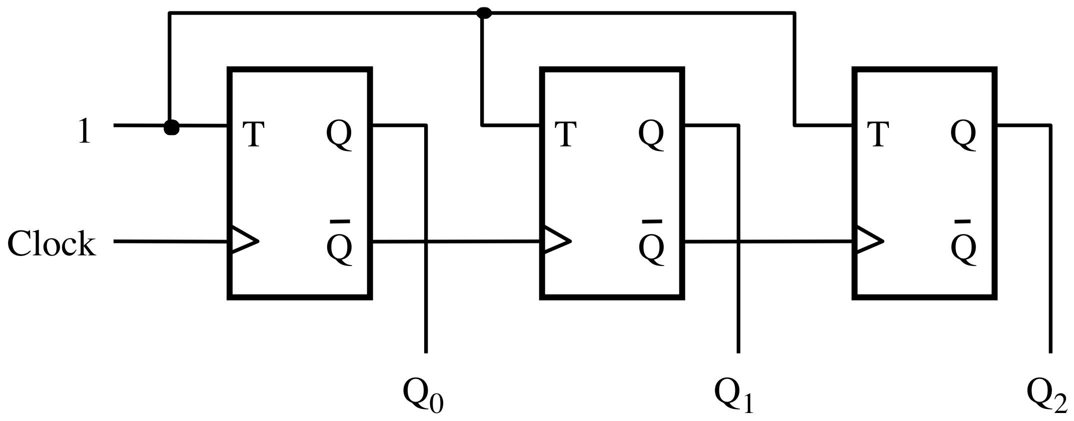
The first flip-flop changes on the positive edge of the clock

The second flip-flop changes on the positive edge of $\overline{Q}_0$

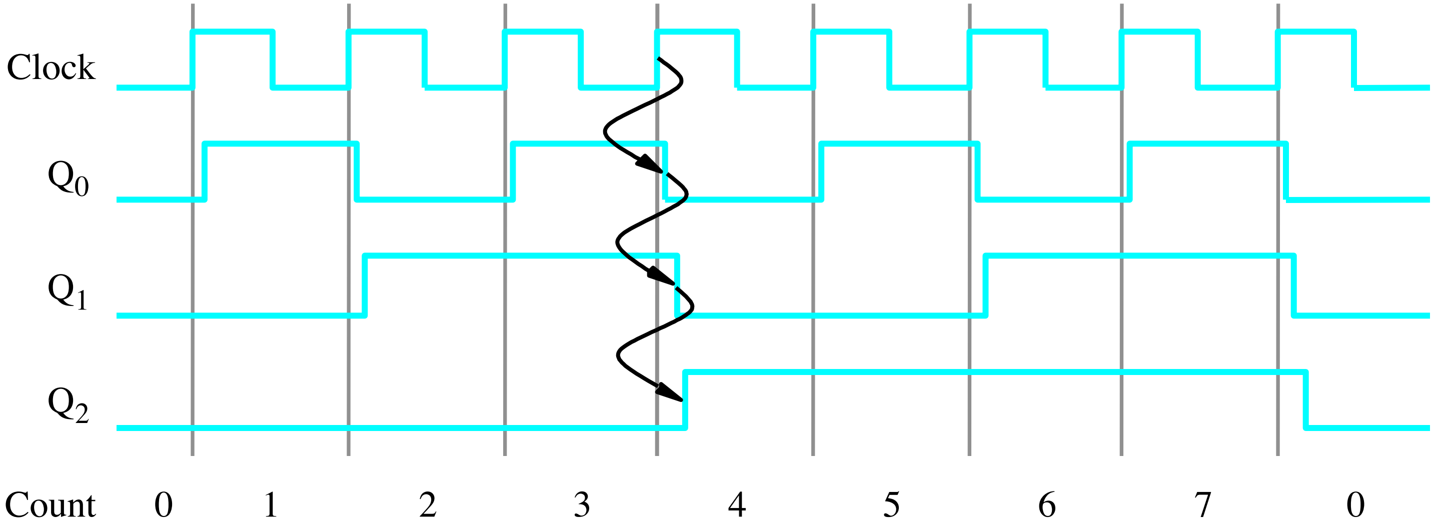
The third flip-flop changes on the positive edge of $\overline{Q}_1$

[Figure 5.19 from the textbook]

A three-bit up-counter



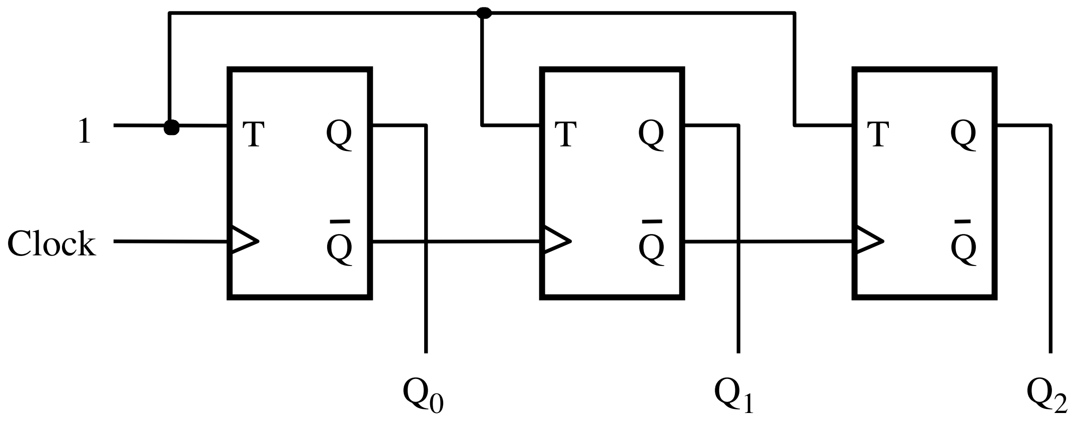
(a) Circuit



(b) Timing diagram

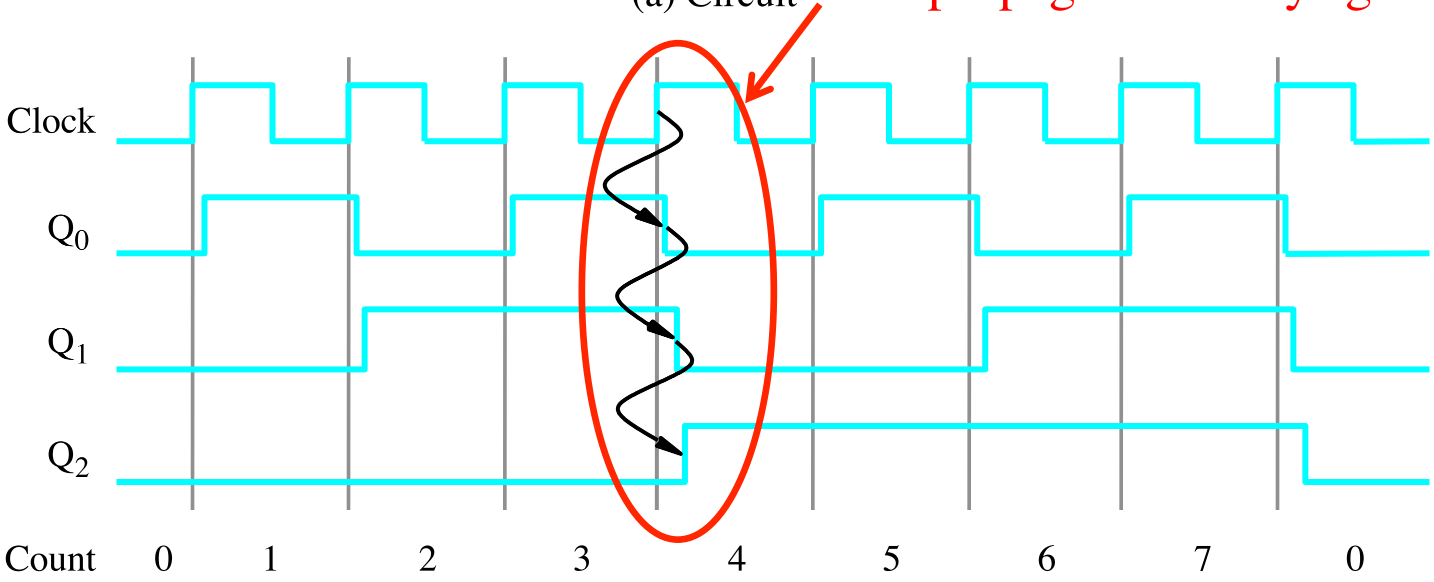
[Figure 5.19 from the textbook]

A three-bit up-counter



(a) Circuit

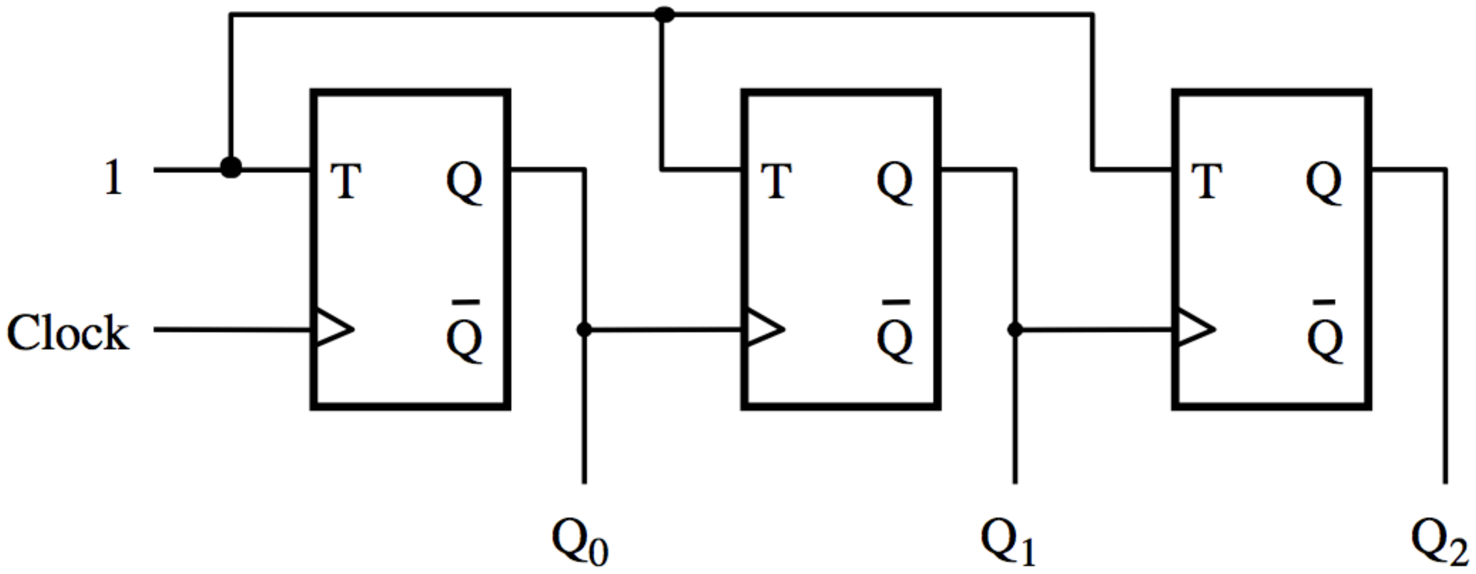
The propagation delays get longer



(b) Timing diagram

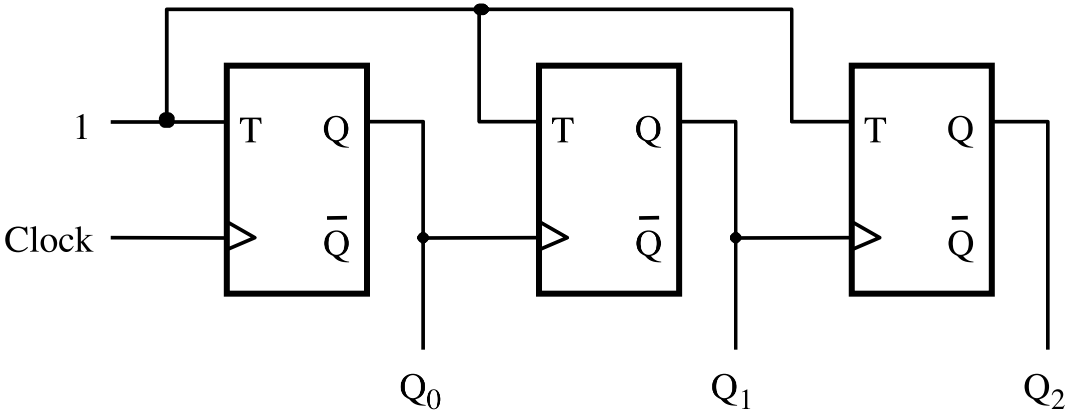
[Figure 5.19 from the textbook]

A three-bit down-counter

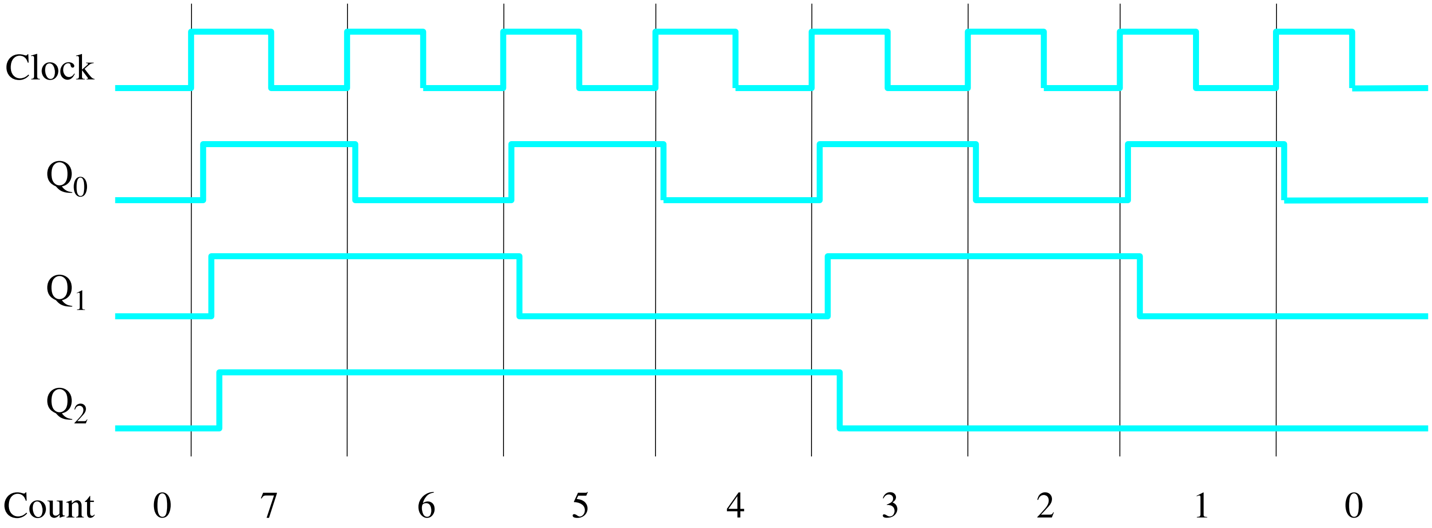


[Figure 5.20 from the textbook]

A three-bit down-counter



(a) Circuit

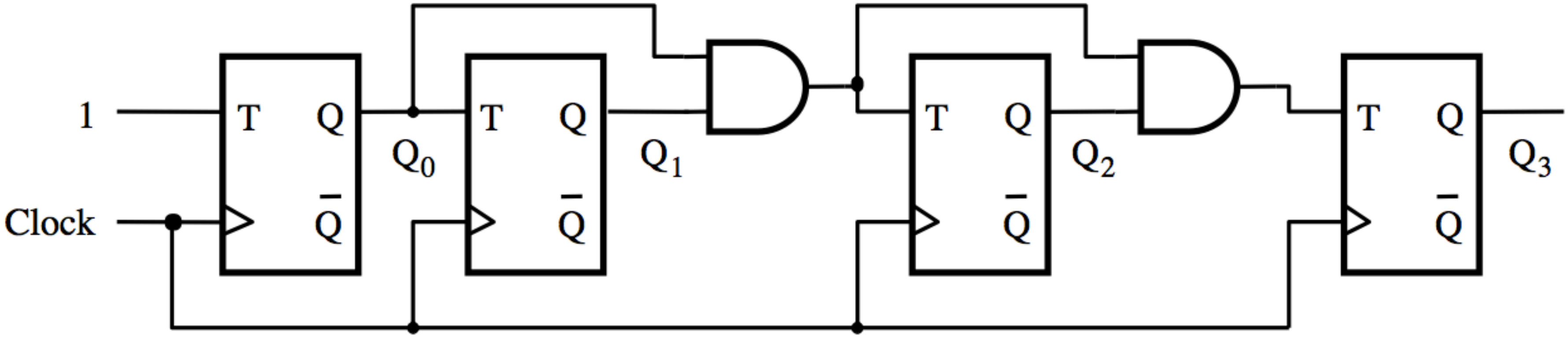


(b) Timing diagram

[Figure 5.20 from the textbook]

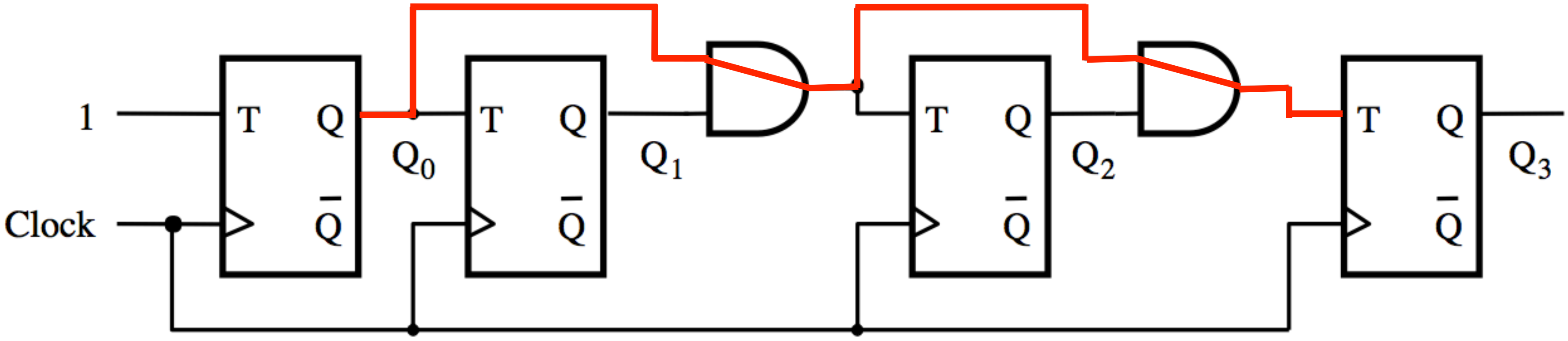
Synchronous Counters

A four-bit synchronous up-counter



[Figure 5.21 from the textbook]

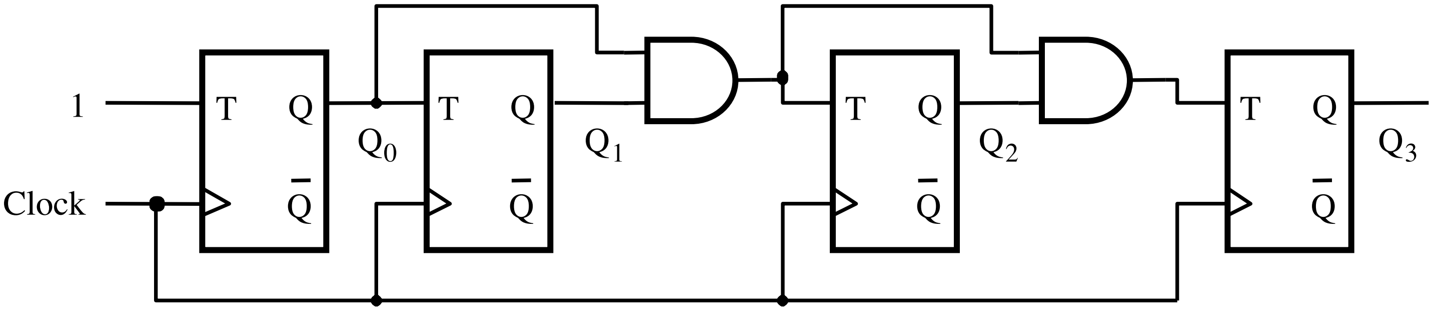
A four-bit synchronous up-counter



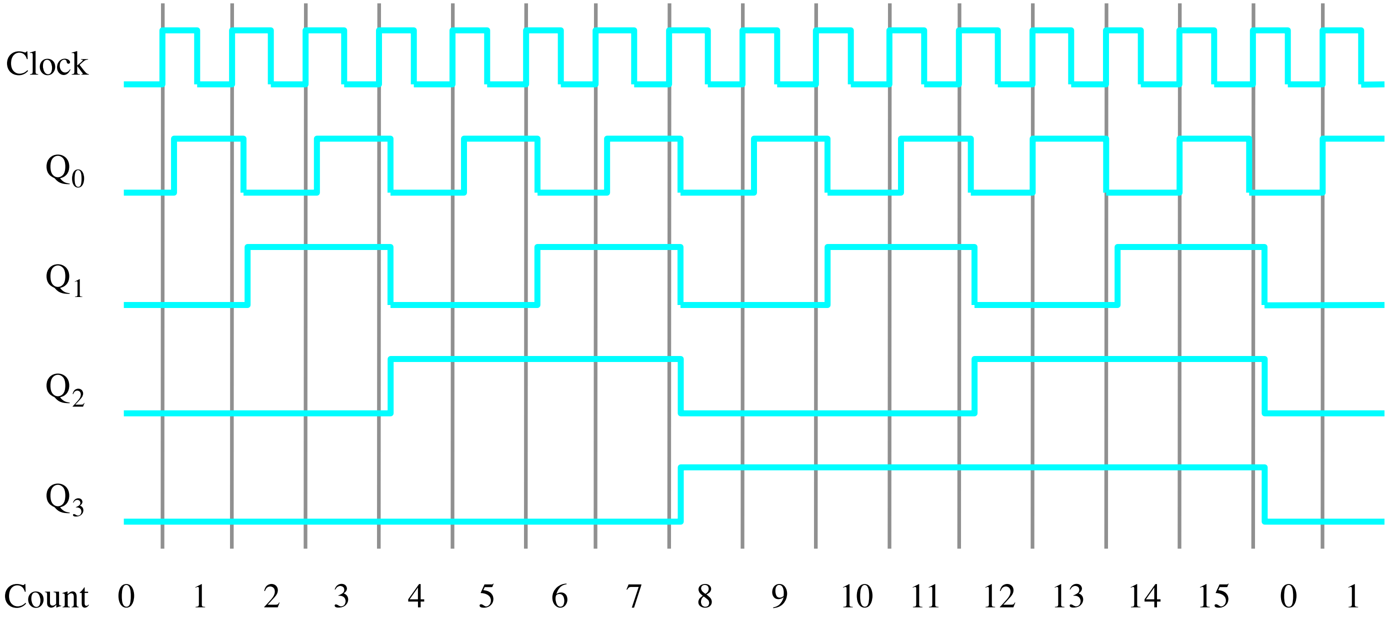
The propagation delay through all AND gates combined must not exceed the clock period minus the setup time for the flip-flops

[Figure 5.21 from the textbook]

A four-bit synchronous up-counter



(a) Circuit

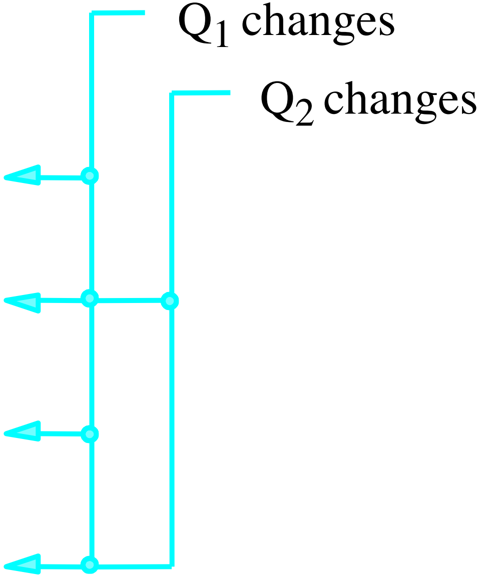


(b) Timing diagram

[Figure 5.21 from the textbook]

Derivation of the 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



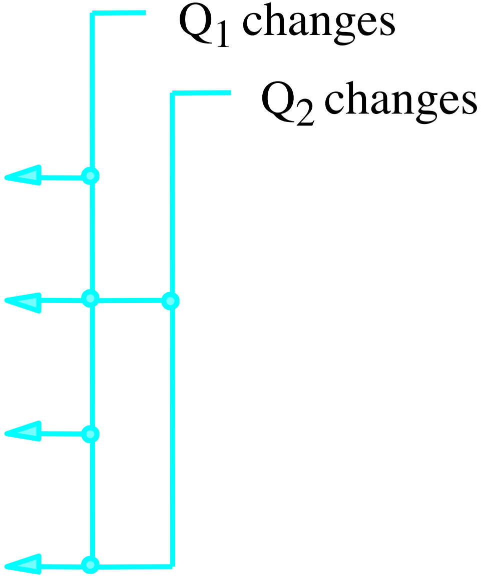
Q_1 changes

Q_2 changes

[Table 5.1 from the textbook]

Derivation of the 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



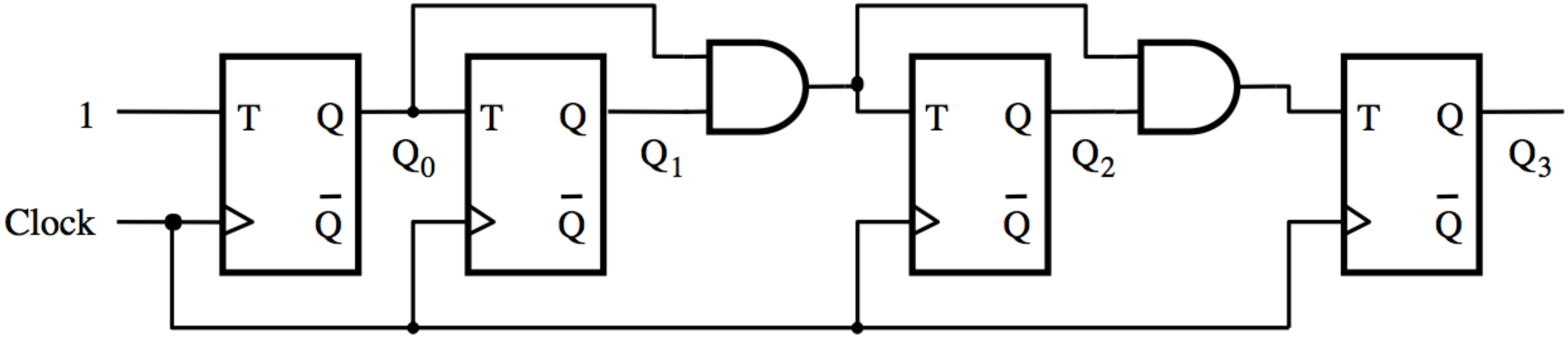
$T_0 = 1$

$T_1 = Q_0$

$T_2 = Q_0 Q_1$

[Table 5.1 from the textbook]

A four-bit synchronous up-counter



$T_0 = 1$
$T_1 = Q_0$
$T_2 = Q_0 Q_1$

[Figure 5.21 from the textbook]

In general we have

$T_0 = 1$

$T_1 = Q_0$

$T_2 = Q_0\ Q_1$

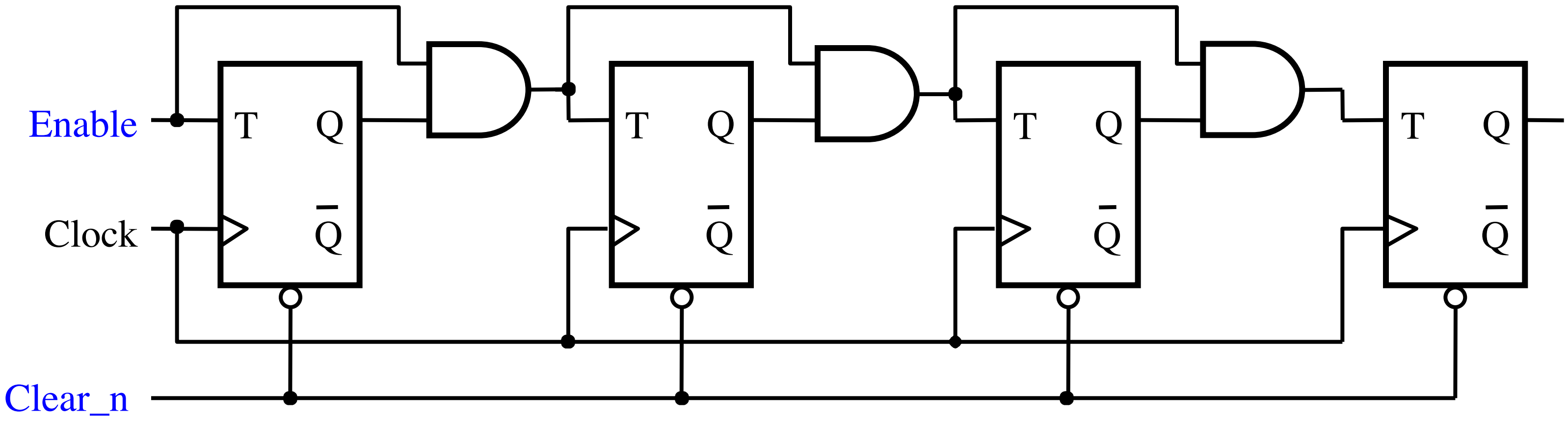
$T_3 = Q_0\ Q_1\ Q_2$

…

$T_n = Q_0\ Q_1\ Q_2\ \ldots Q_{n-1}$

Adding Enable and Clear Capability

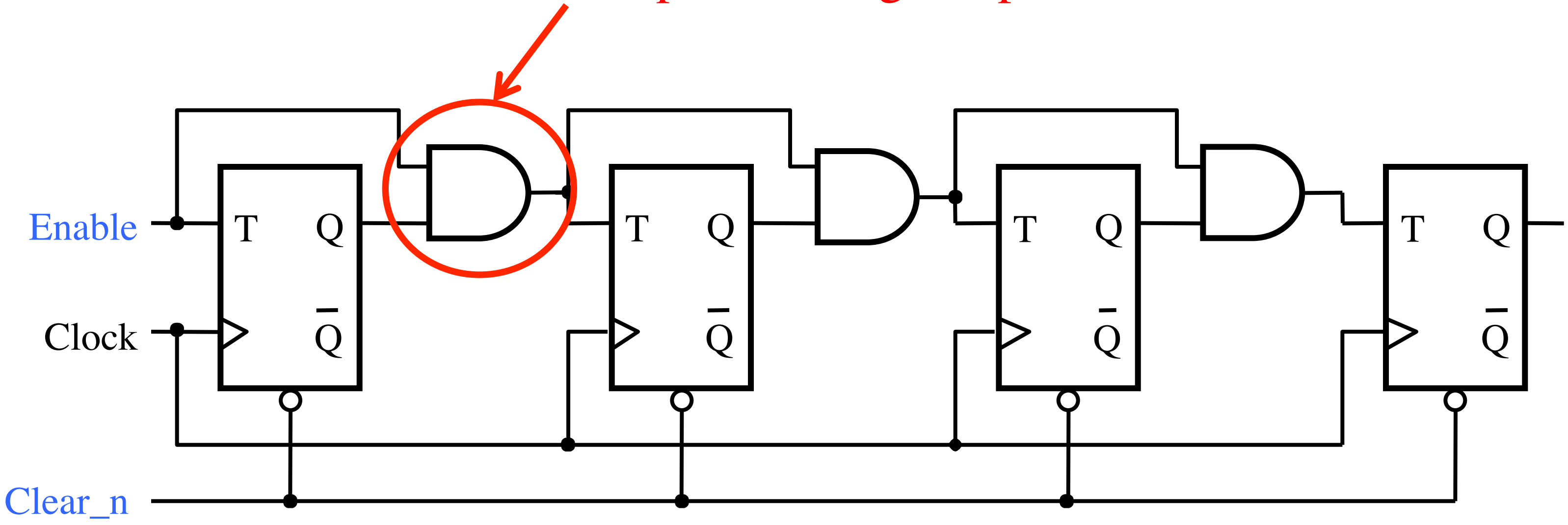
Inclusion of Enable and Clear capability



[Figure 5.22 from the textbook]

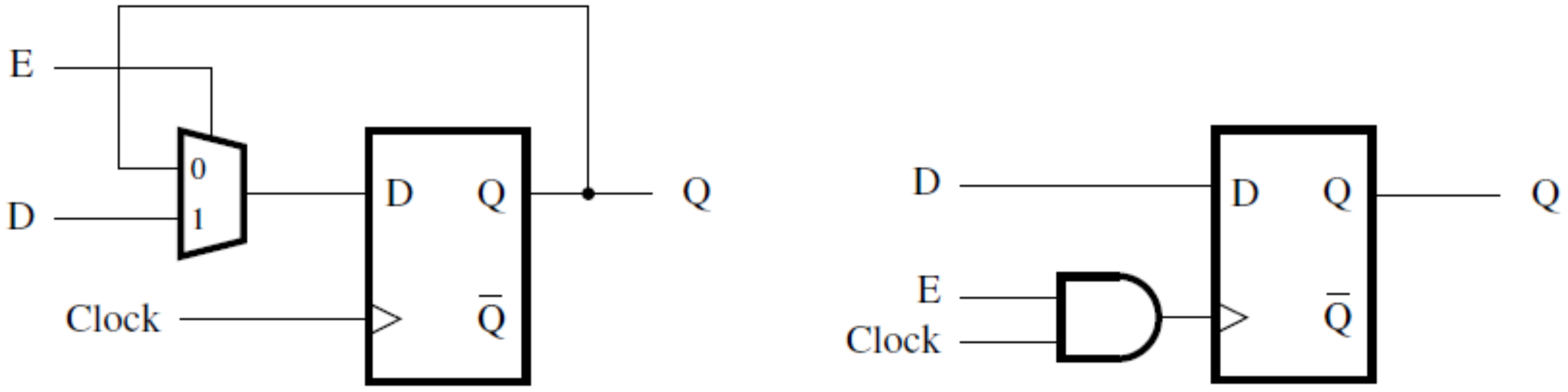
Inclusion of Enable and Clear capability

This is the new thing relative to
the previous figure, plus the clear_n line



[Figure 5.22 from the textbook]

Providing an enable input for a D flip-flop



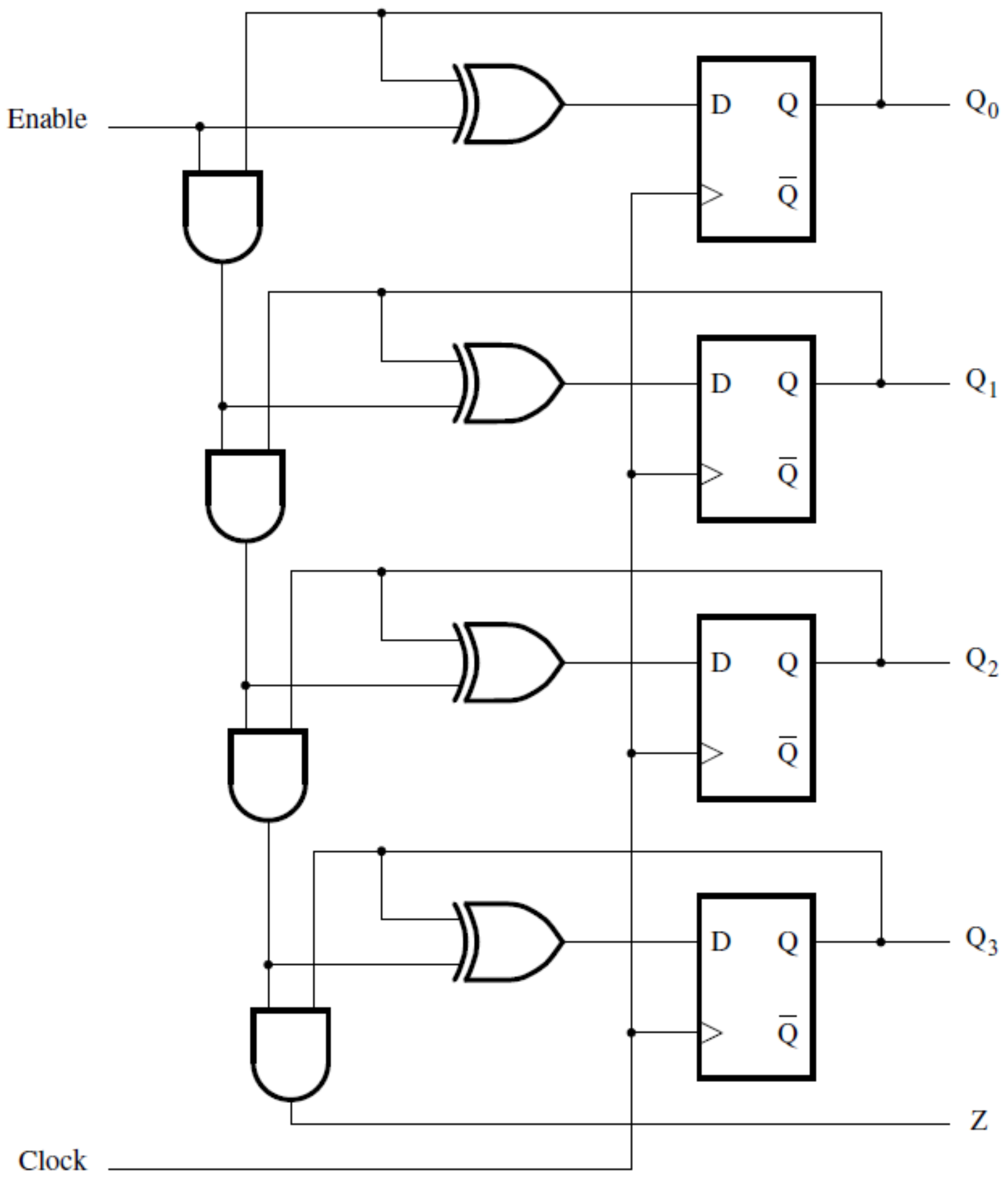
(a) Using a multiplexer

(b) Clock gating

[Figure 5.56 from the textbook]

Synchronous Counter with D Flip-Flops

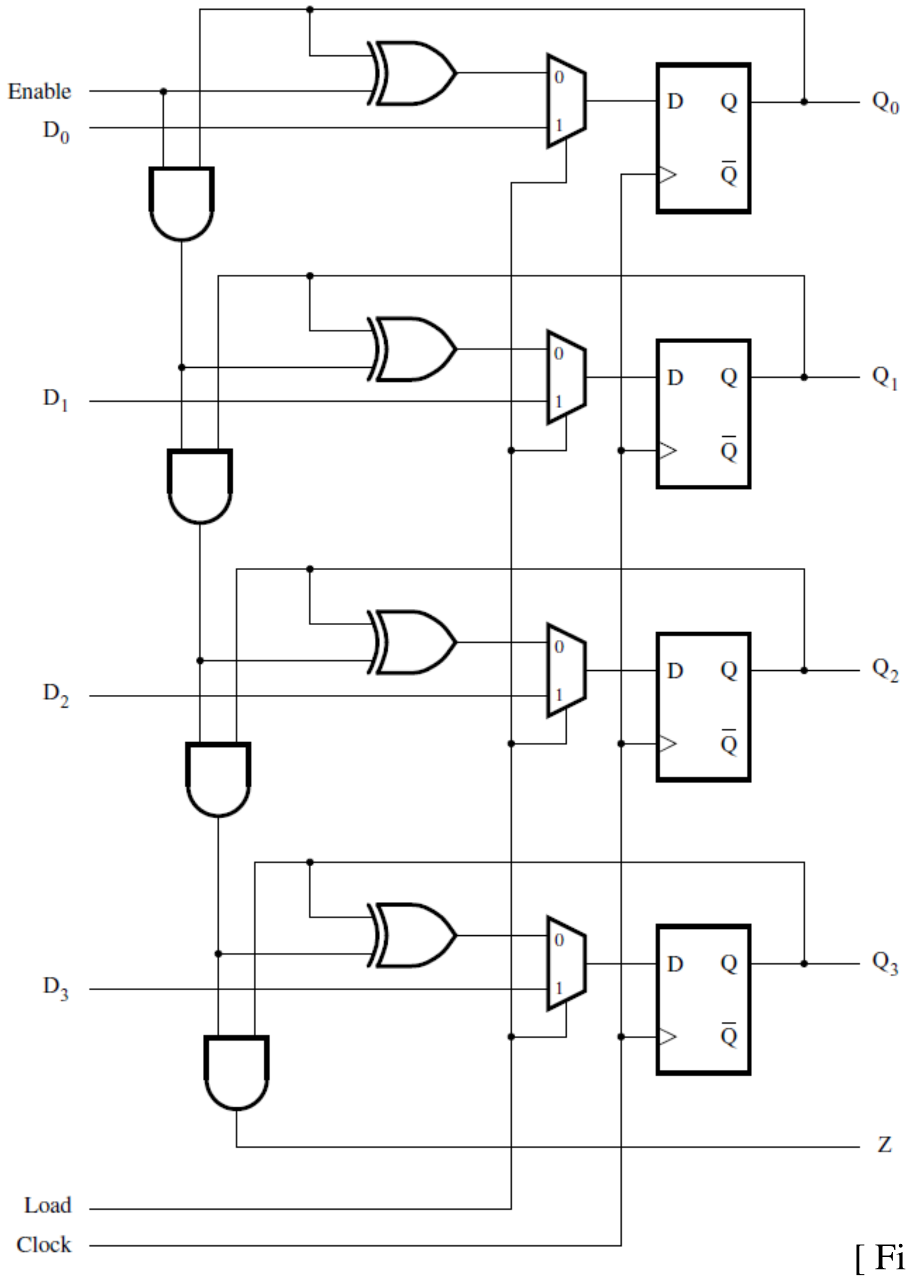
A four-bit counter with D flip-flops



[Figure 5.23 from the textbook]

Counters with Parallel Load

A counter with parallel-load capability



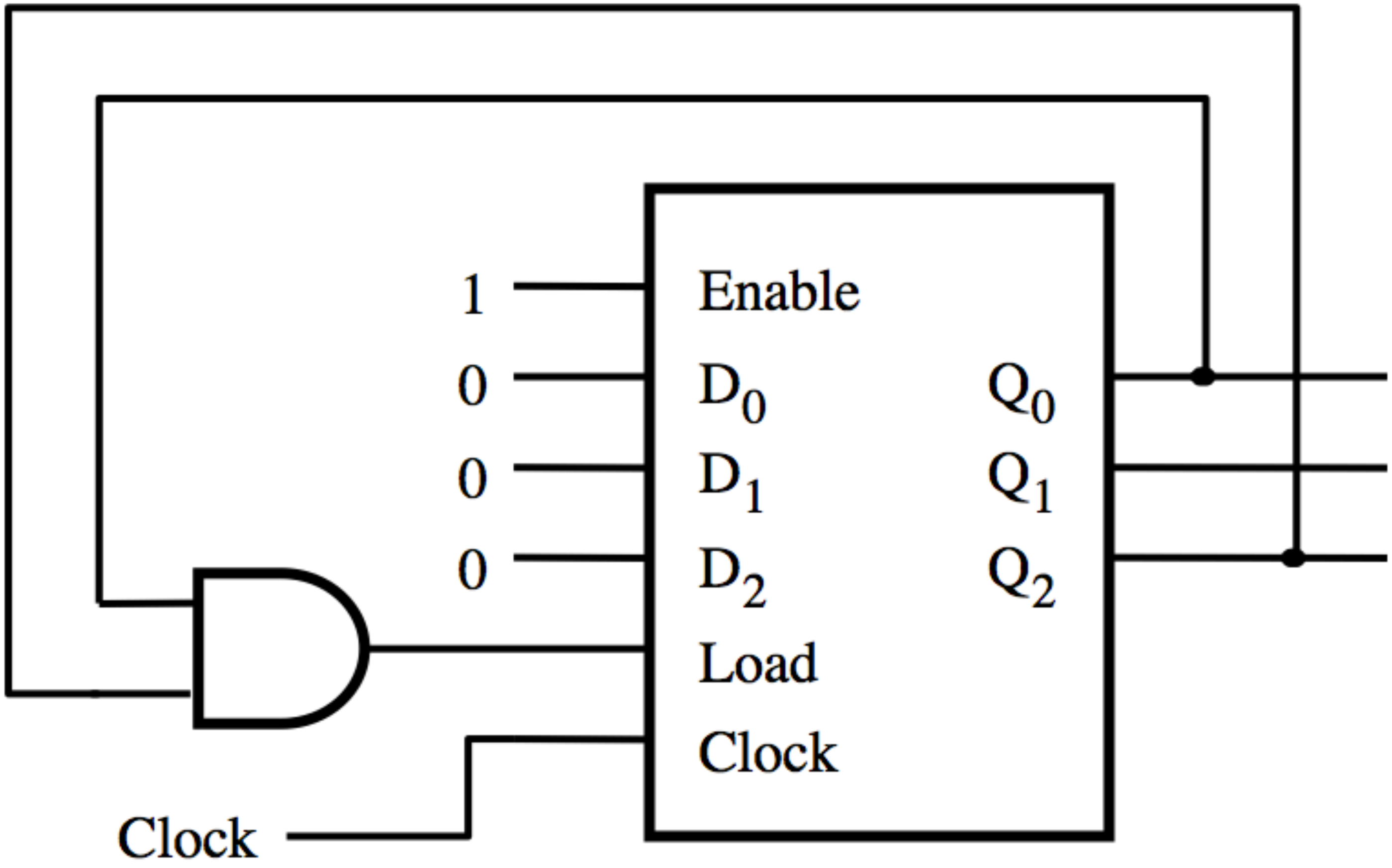
[Figure 5.24 from the textbook]

Reset Synchronization

Motivation

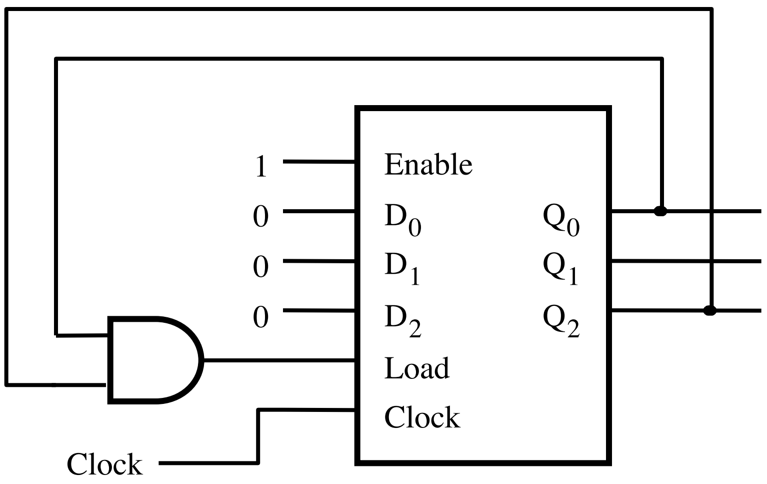
- An n-bit counter counts from 0, 1, …, 2^n-1
- For example a 3-bit counter counts up as follow
  - 0, 1, 2, 3, 4, 5, 6, 7, 0, 1, 2, …
- What if we want it to count like this
  - 0, 1, 2, 3, 4, 5, 0, 1, 2, 3, 4, 5, 0, 1, …
- In other words, what is the cycle is not a power of 2?

What does this circuit do?

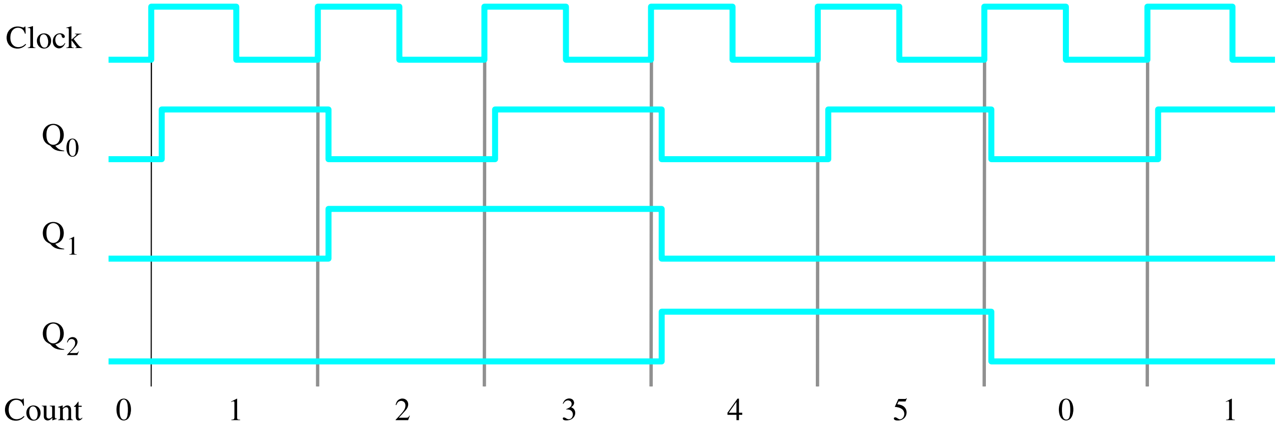


[Figure 5.25a from the textbook]

A modulo-6 counter with synchronous reset



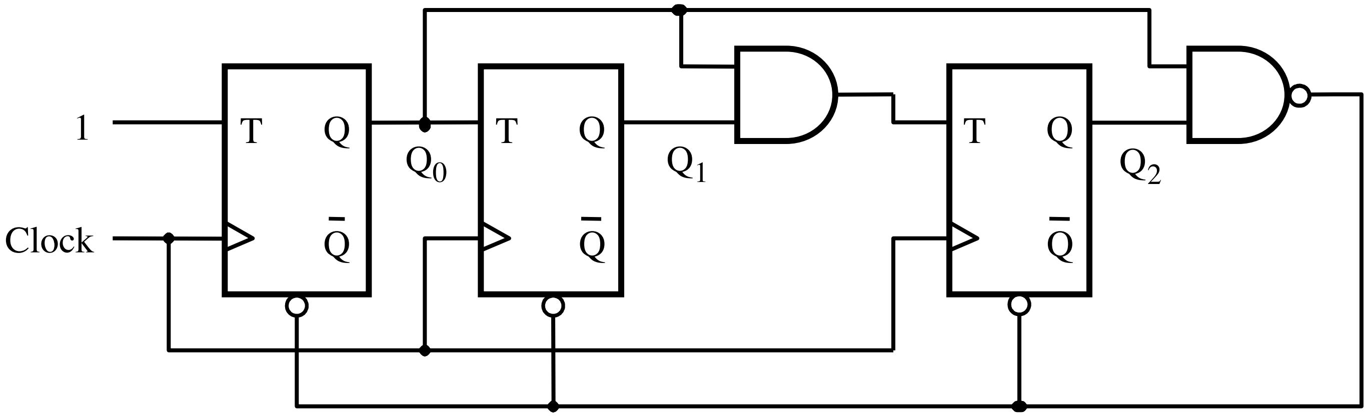
(a) Circuit



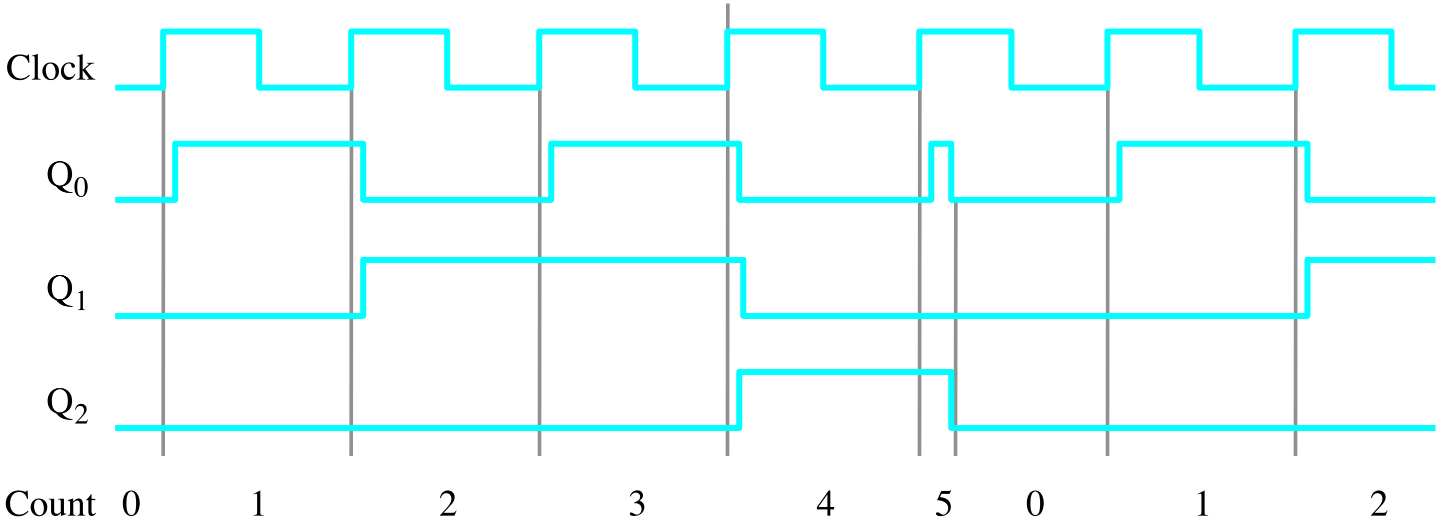
(b) Timing diagram

[Figure 5.25 from the textbook]

A modulo-6 counter with asynchronous reset



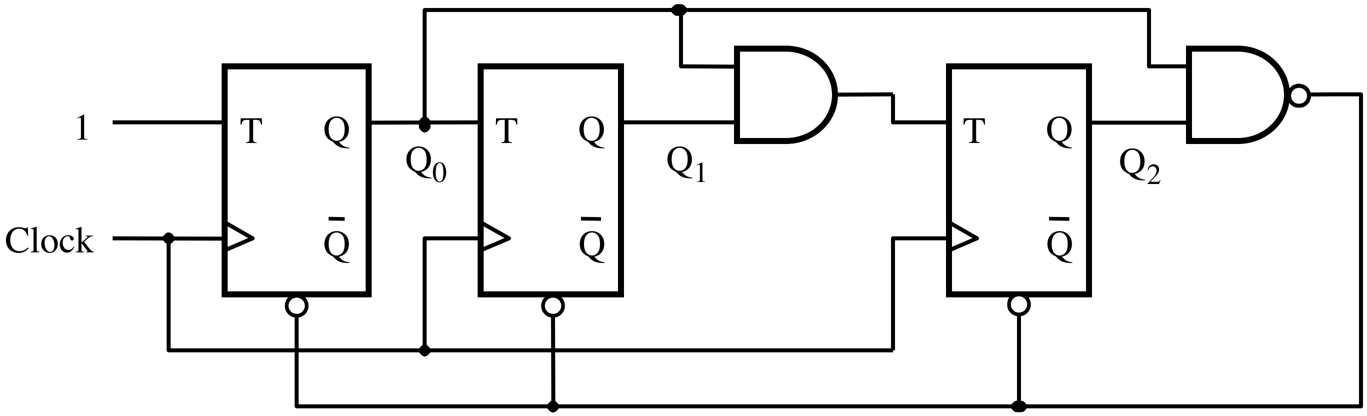
(a) Circuit



(b) Timing diagram

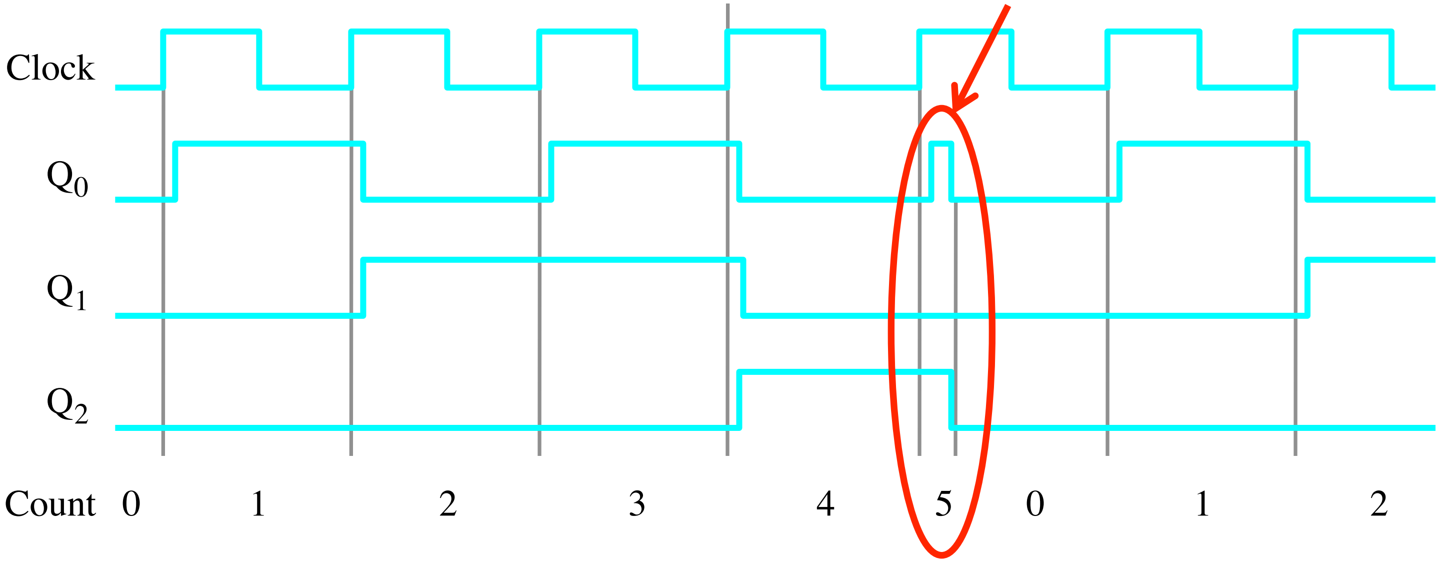
[Figure 5.26 from the textbook]

A modulo-6 counter with asynchronous reset



(a) Circuit

The number 5 is displayed for a very short amount of time



(b) Timing diagram

[Figure 5.26 from the textbook]

Questions?

THE END