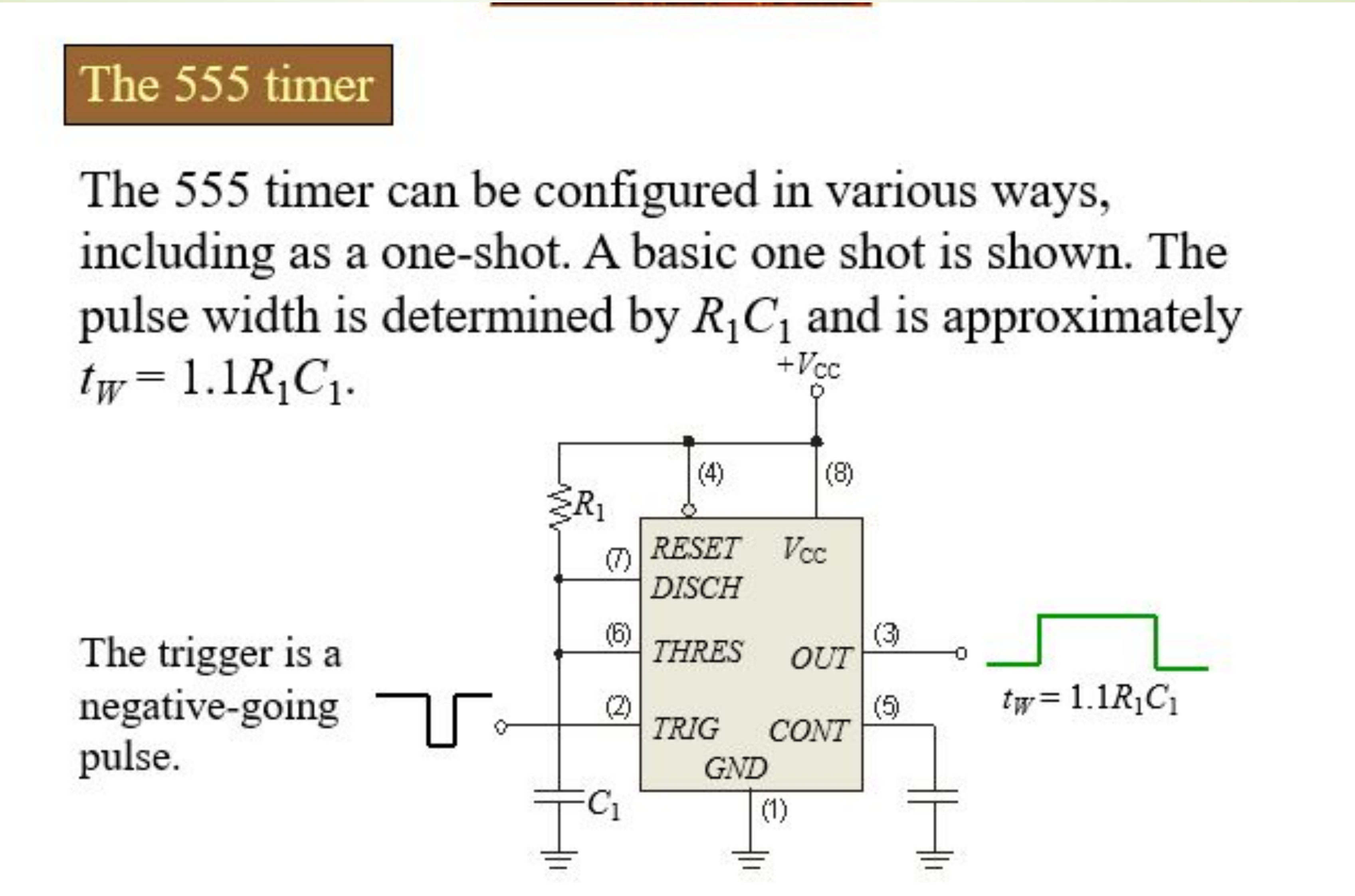
Digital Logic Design EE(1005)

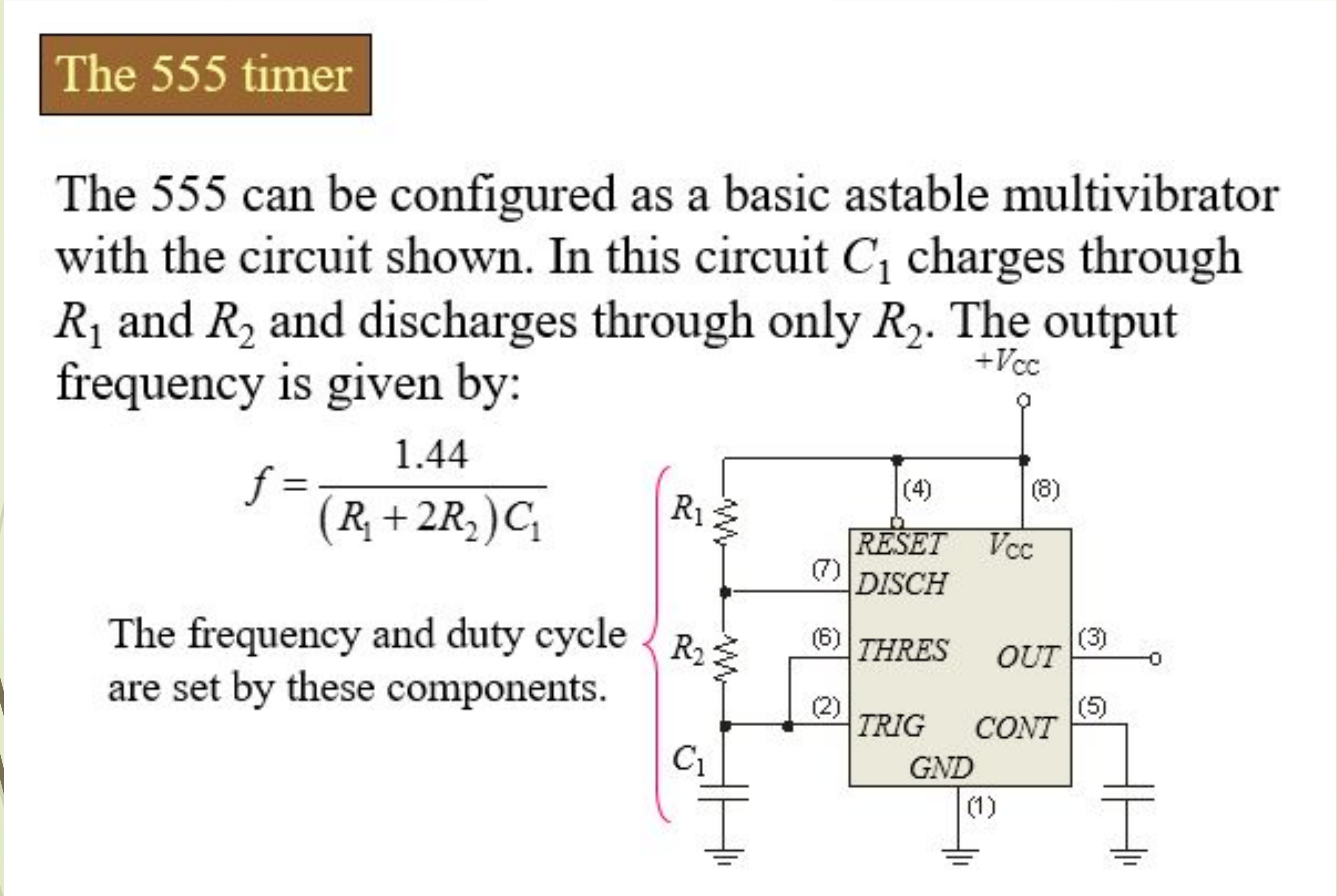
Lecture- 31



The 555 timer

The 555 timer can be configured in various ways, including as a one-shot. A basic one shot is shown. The pulse width is determined by R_1C_1 and is approximately $t_W = 1.1R_1C_1$.

The trigger is a negative-going pulse.



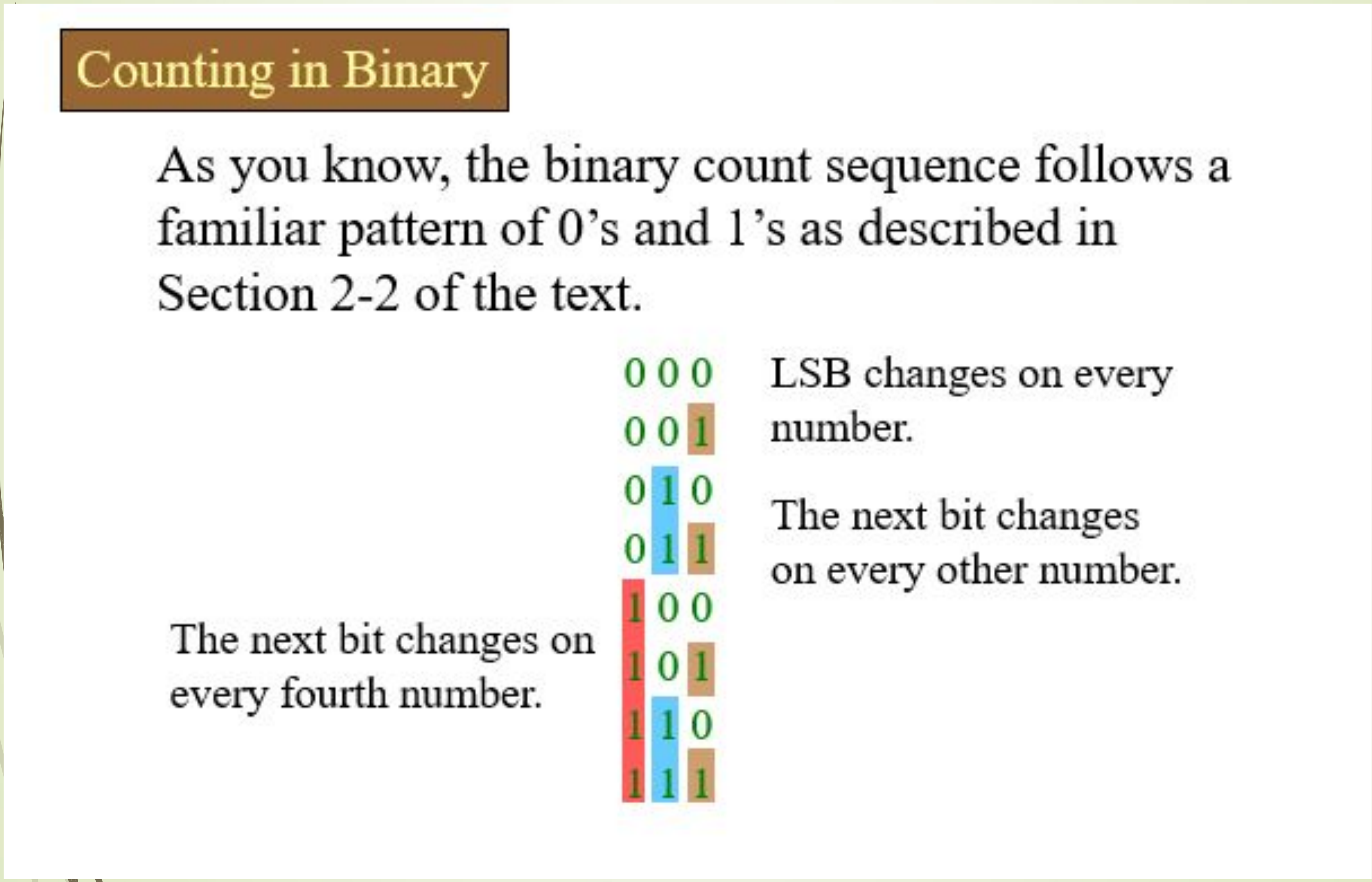
The 555 timer

The 555 can be configured as a basic astable multivibrator with the circuit shown. In this circuit C_1 charges through R_1 and R_2 and discharges through only R_2. The output frequency is given by:

$$f = \frac{1.44}{(R_1 + 2R_2)C_1}$$

The frequency and duty cycle are set by these components.

+V_{CC}
R_1
R_2
C_1
(4)
(8)
RESET
V_{CC}
(7)
DISCH
(6)
THRES
OUT
(3)
(2)
TRIG
CONT
(5)
GND
(1)



Counting in Binary

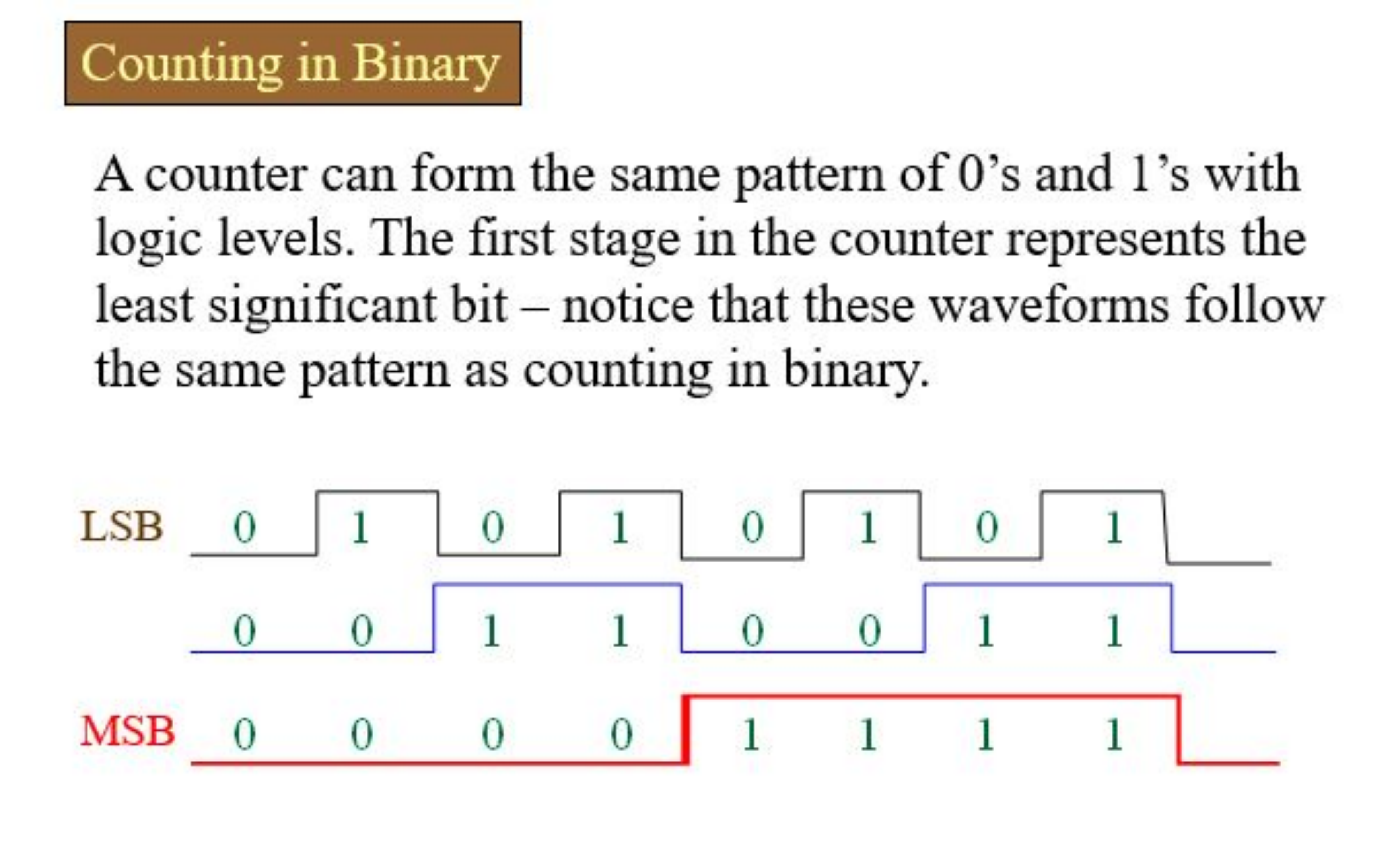
As you know, the binary count sequence follows a familiar pattern of 0’s and 1’s as described in Section 2-2 of the text.

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

LSB changes on every number.

The next bit changes on every other number.

The next bit changes on every fourth number.



Counting in Binary

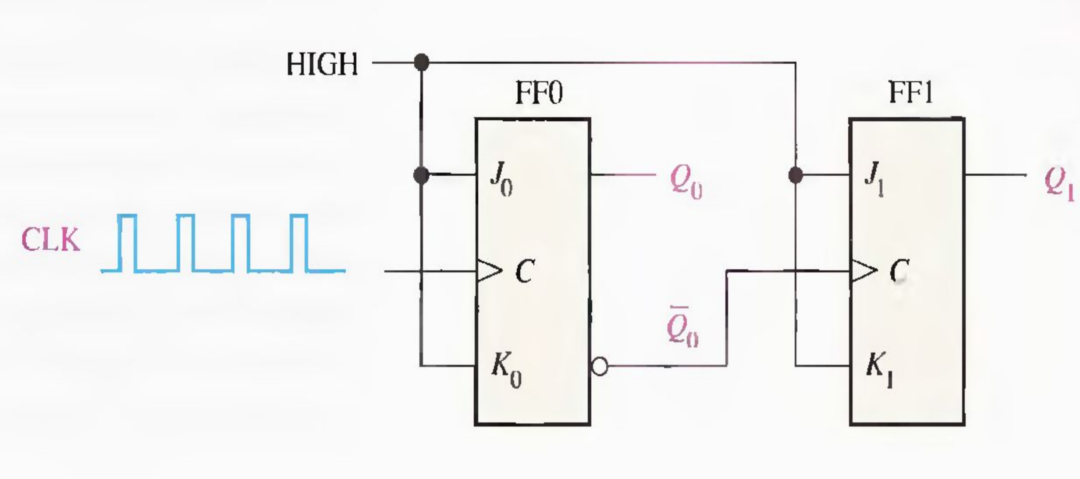
A counter can form the same pattern of 0's and 1's with logic levels. The first stage in the counter represents the least significant bit – notice that these waveforms follow the same pattern as counting in binary.

ASYNCHRONOUS COUNTER OPERATION

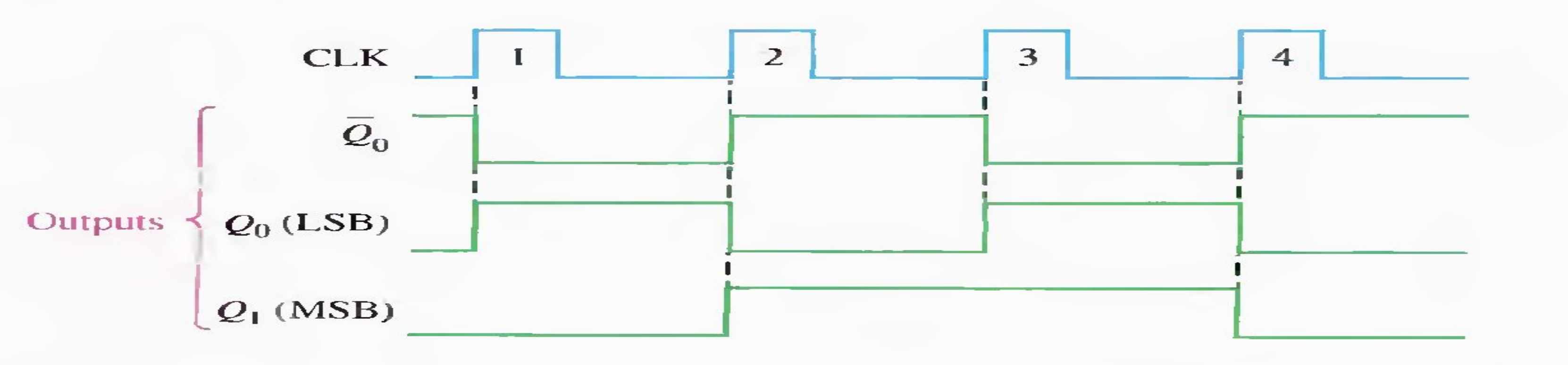
The term **asynchronous** refers to events that do not have a fixed time relationship with each other and, generally, do not occur at the same time. An **asynchronous counter** is one in which the flip-flops (FF) within the counter do not change states at exactly the same time because they do not have a common clock pulse.

A 2-Bit Asynchronous Binary Counter

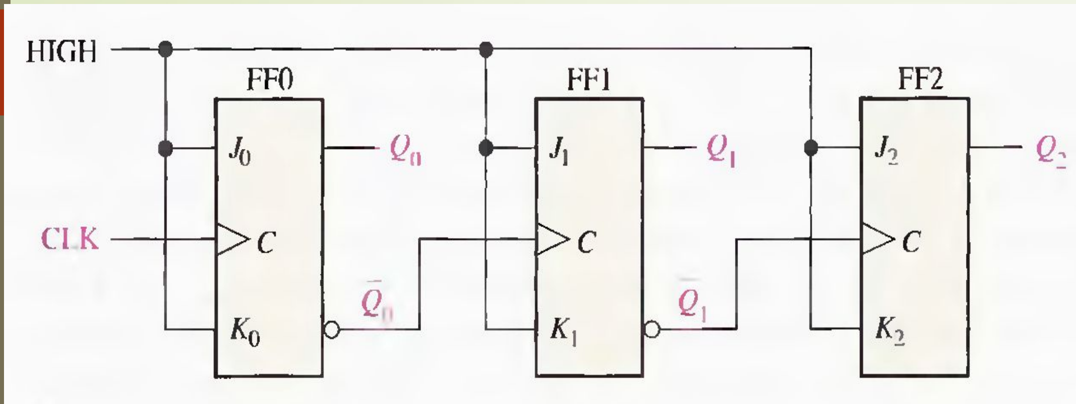
Asynchronous counters are also known as ripple counters.



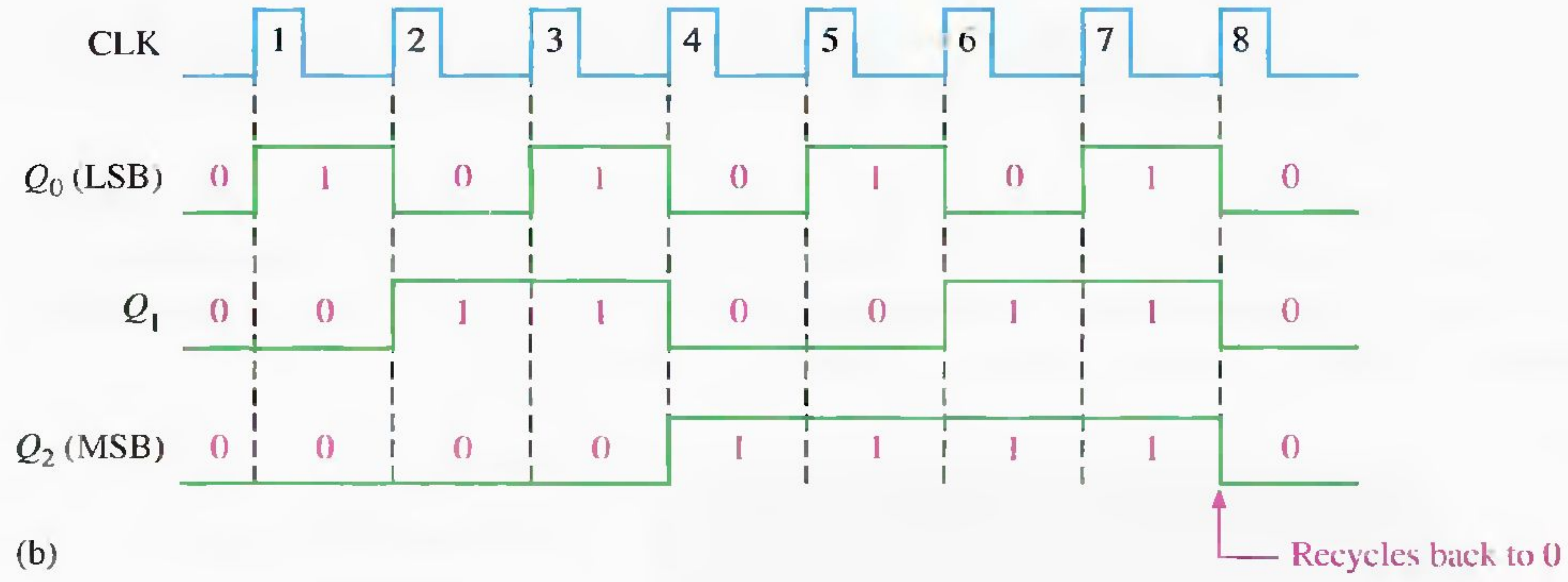
CLOCK PULSE	Q_1	Q_0
Initially	0	0
1	0	1
2	1	0
3	1	1
4 (recycles)	0	0



A 3-Bit Asynchronous Binary Counter



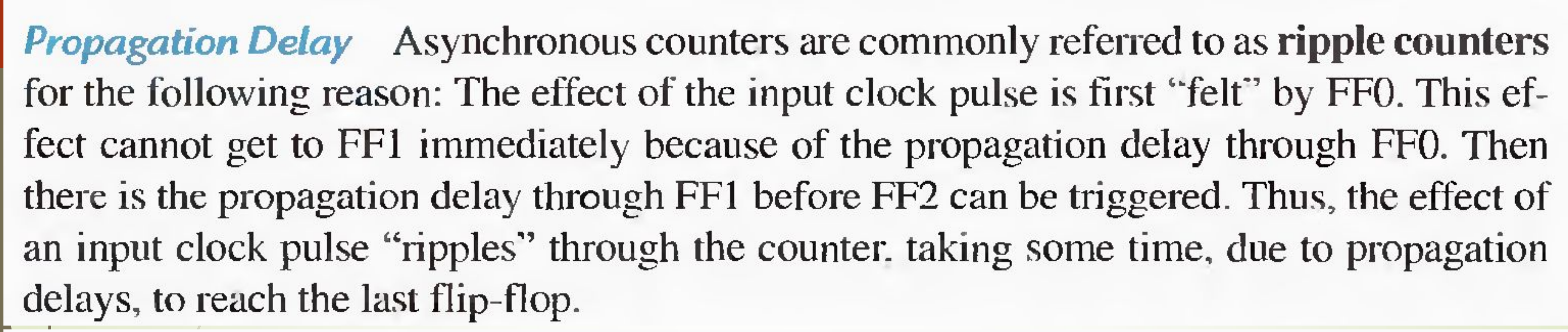
CLOCK PULSE	Q_2	Q_1	Q_0
Initially	0	0	0
1	0	0	1
2	0	1	0
3	0	1	1
4	1	0	0
5	1	0	1
6	1	1	0
7	1	1	1
8 (recycles)	0	0	0



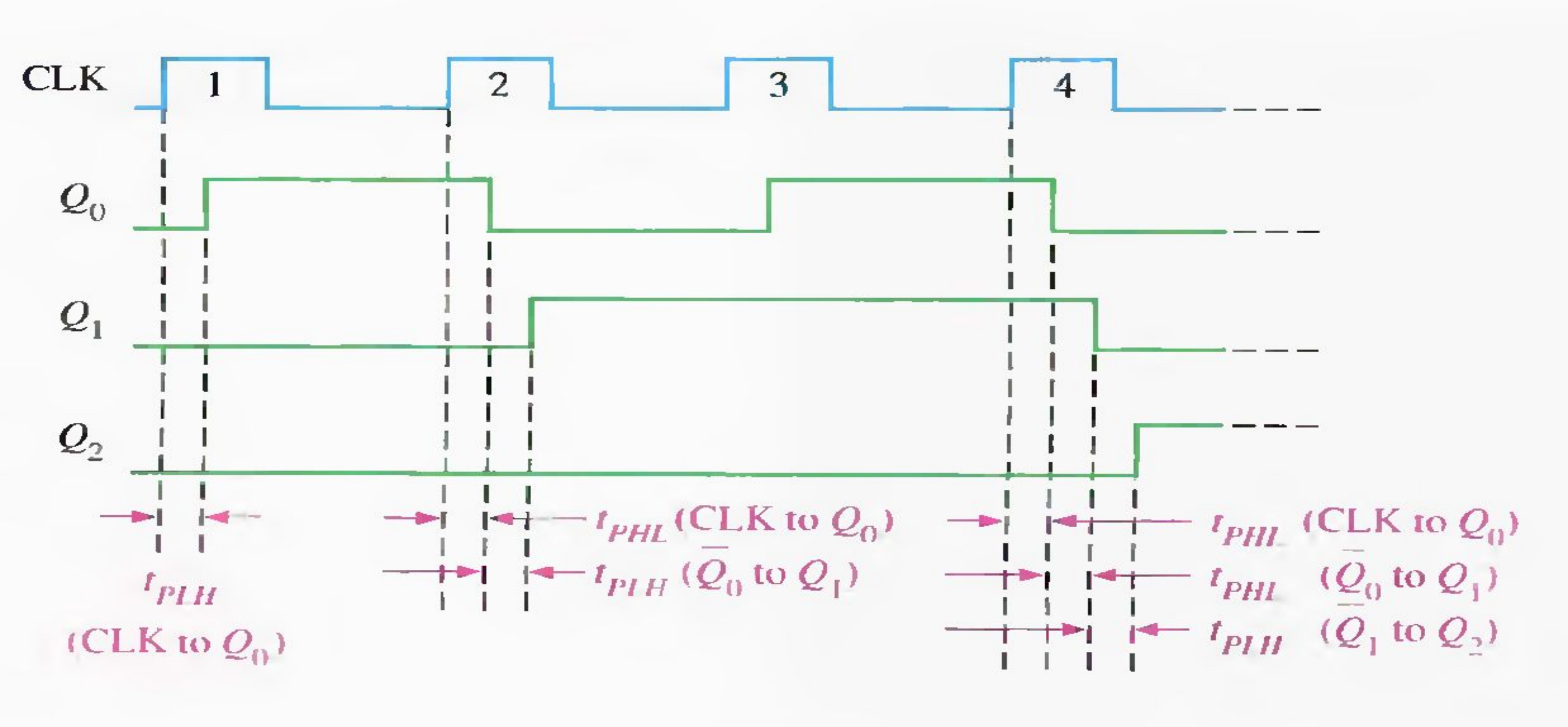
Recycles back to 0

(b)

Propagation Delay

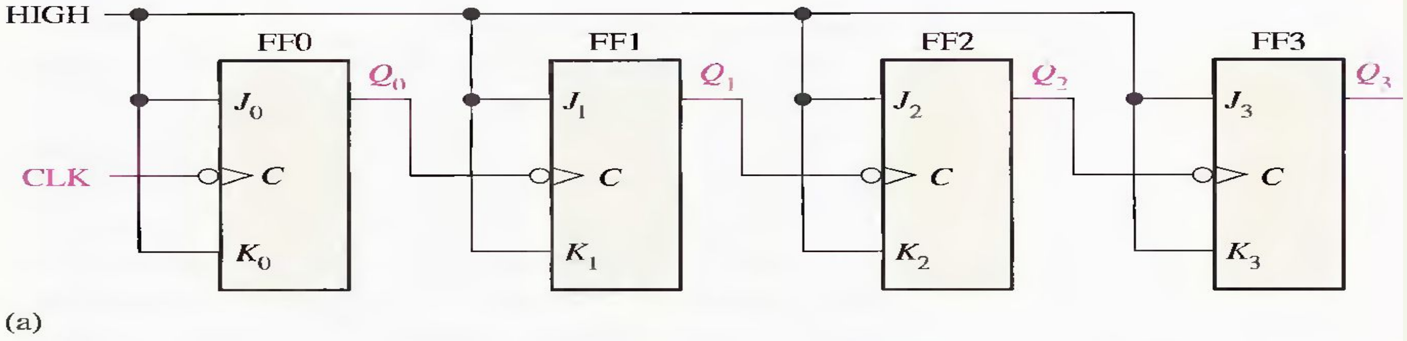


Propagation Delay Asynchronous counters are commonly referred to as **ripple counters** for the following reason: The effect of the input clock pulse is first "felt" by FF0. This effect cannot get to FF1 immediately because of the propagation delay through FF0. Then there is the propagation delay through FF1 before FF2 can be triggered. Thus, the effect of an input clock pulse "ripples" through the counter. taking some time, due to propagation delays, to reach the last flip-flop.

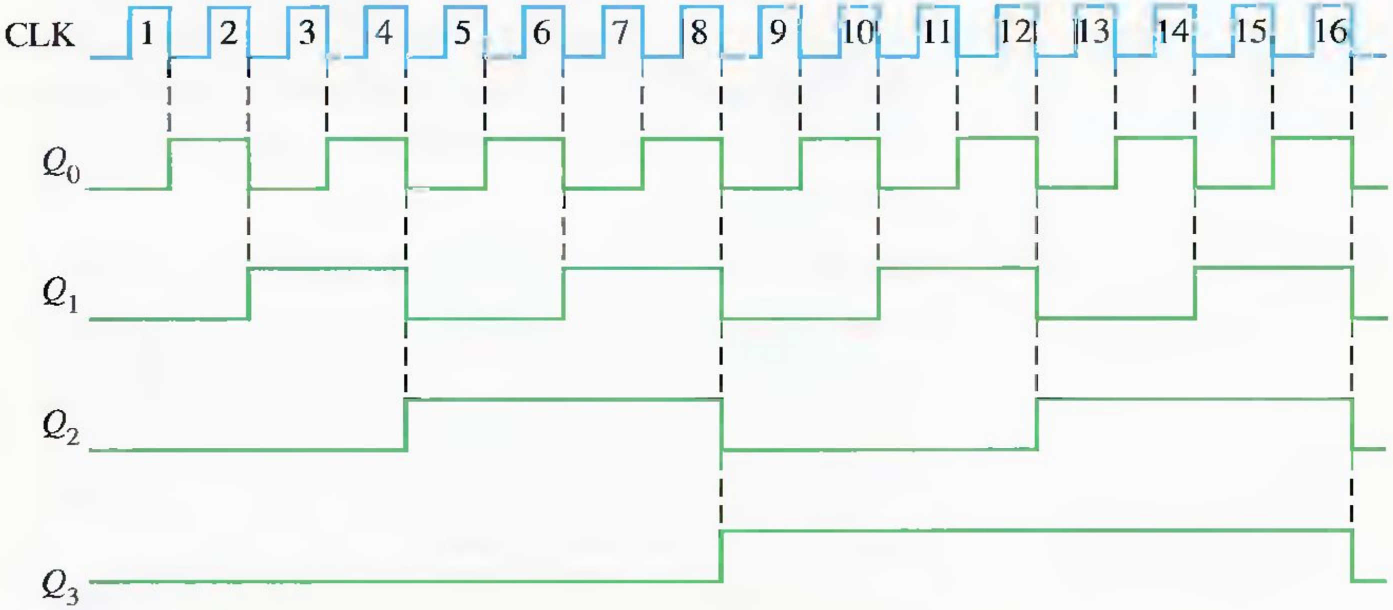


CLK
1
2
3
4
Q_0
Q_1
Q_2
t_{PHL} (CLK to Q_0)
t_{PLH} ($\overline{Q}_0$ to Q_1)
t_{PHL} (CLK to Q_0)
t_{PHL} ($\overline{Q}_0$ to Q_1)
t_{PLH} (Q_1 to Q_2)
t_{PLH}
(CLK to Q_0)

A 4-bit asynchronous binary counter is shown in Figure 8–5(a). Each flip-flop is negative edge-triggered and has a propagation delay for 10 nanoseconds (ns). Develop a timing diagram showing the Q output of each flip-flop, and determine the total propagation delay time from the triggering edge of a clock pulse until a corresponding change can occur in the state of Q_3. Also determine the maximum clock frequency at which the counter can be operated.



(a)



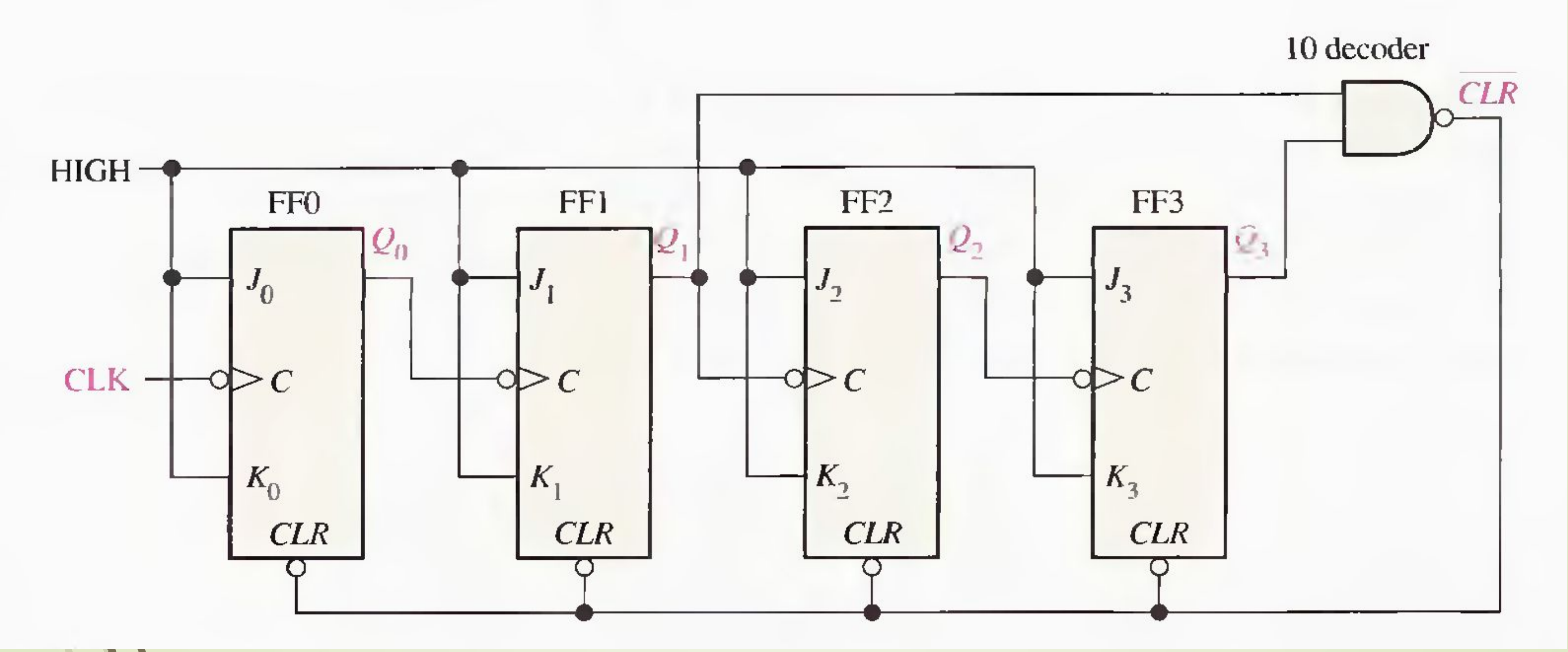
$$t_{p(tot)} = 4 \times 10 \text{ ns} = 40 \text{ ns}$$

$$f_{\max} = \frac{1}{t_{p(tot)}} = \frac{1}{40 \text{ ns}} = 25 \text{ MHz}$$

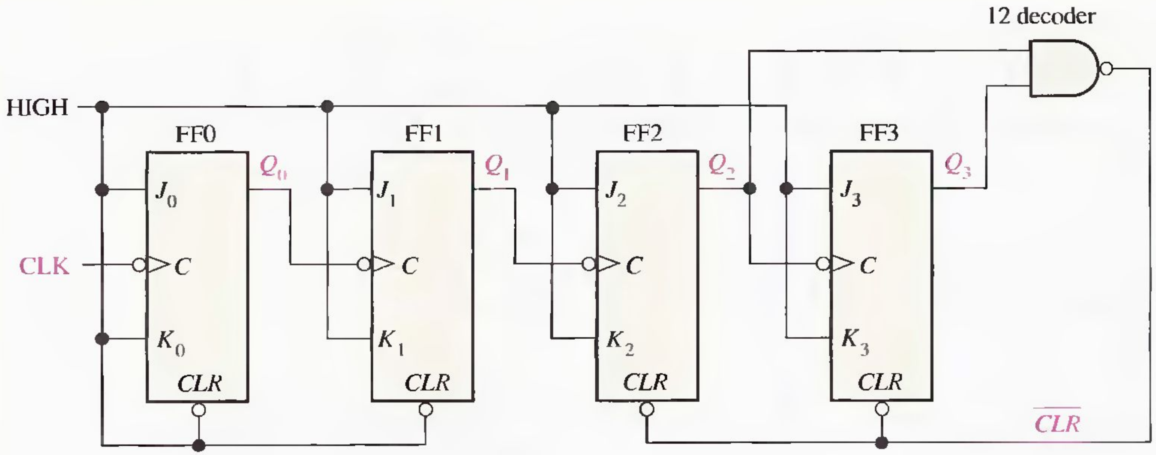
Asynchronous Decade Counters

The **modulus** of a counter is the number of unique states through which the counter will sequence. The maximum possible number of states (maximum modulus) of a counter is 2", where *n* is the number of flip-flops in the counter. Counters can be designed to have a number of states in their sequence that is less than the maximum of 2". This type of sequence is called a *truncated sequence.*

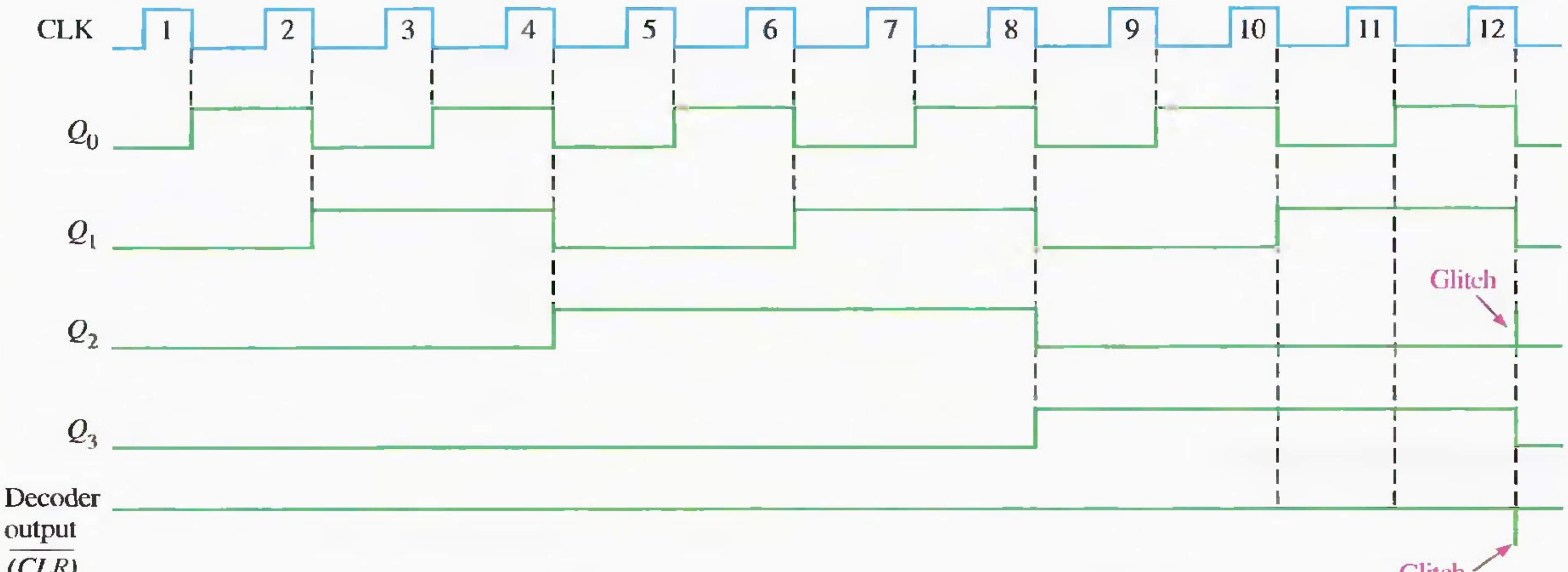
One common modulus for counters with truncated sequences is ten (called MOD10) Counters with ten states in their sequence are called **decade** counters.



Show how an asynchronous counter can be implemented having a modulus of twelve with a straight binary sequence from 0000 through 1011.



Q_3	Q_2	Q_1	Q_0	
0	0	0	0	← Recycles
.	.	.	.	
.	.	.	.	
.	.	.	.	
1	0	1	1	←
1	1	0	0	← Normal next state


CLK
Q_0
Q_1
Q_2
Q_3
Glitch
Decoder output ($\overline{CLR}$)
Glitch