Digital System Design

Electrical Department-Fourth Stage

Lecture Six

By

Assistant Lecturer: Adnan Ali Abdullah

2025-2024

Cascaded Counters

Counters can be connected in cascade to achieve higher-modulus operation. In essence, Cascading means that the last-stage output of one counter drives the input of the next counter. There are two types of connections:

- **Asynchronous Cascading**
- **Synchronous Cascading**

Asynchronous Cascading

- An example of two asynchronous counters connected in cascade is shown in Figure 1 for a 2-bit and a 3-bit ripple counter.

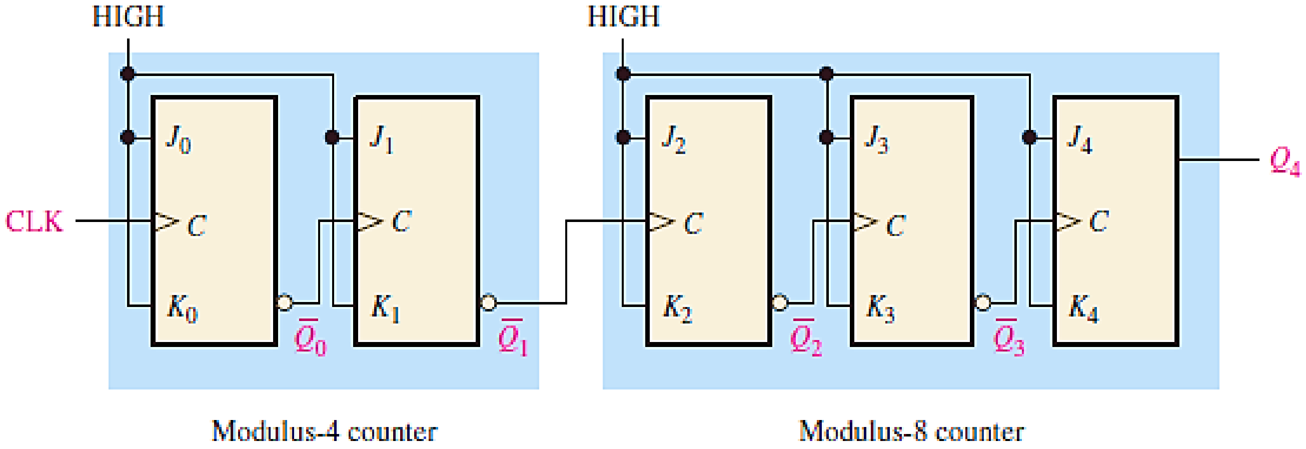

FIGURE 1 Two cascaded asynchronous counters (all J and K inputs are HIGH).

- The timing diagram is shown in Figure 2. Notice that the final output of the modulus-8 counter, Q_4, occurs once for every 32 input clock pulses.
- The overall modulus of the two cascaded counters is 4 * 8 = 32; that is, they act as a divide-by-32 counter.

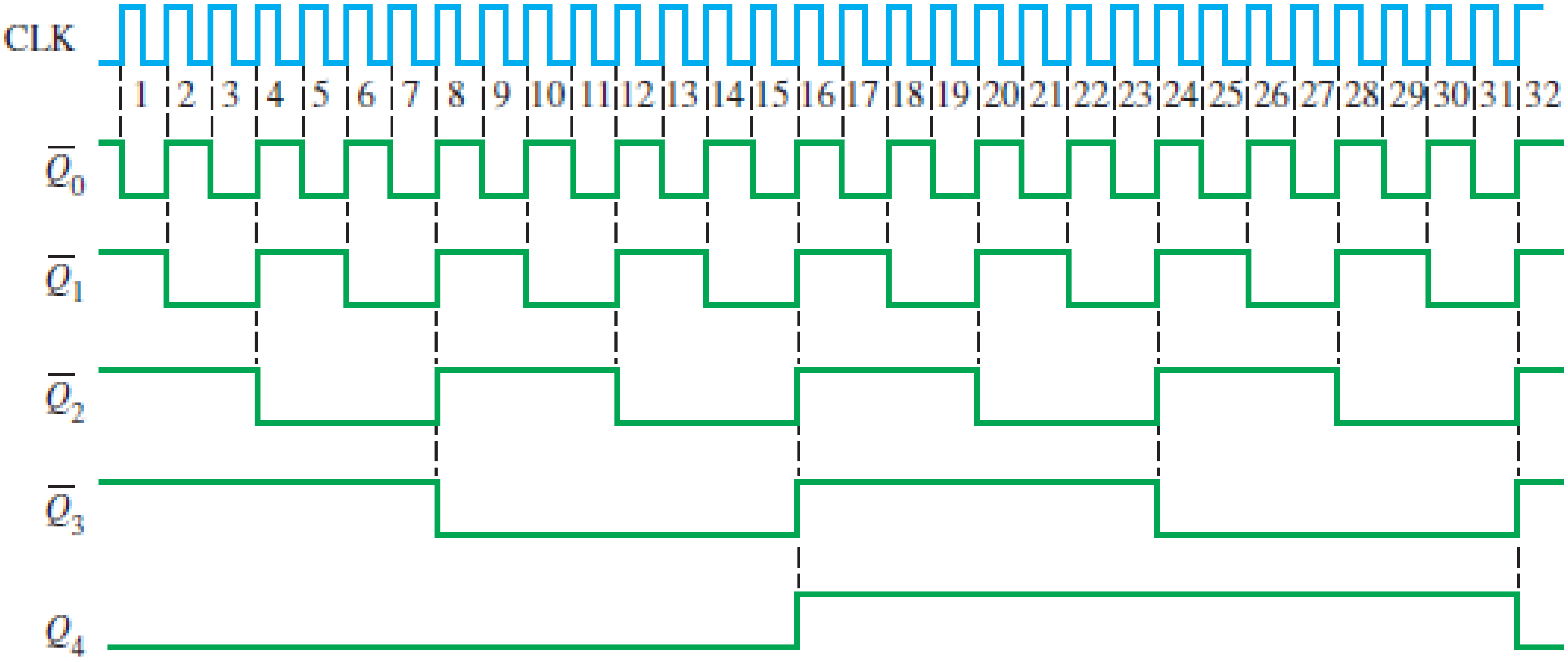

FIGURE 2 Timing diagram for the cascaded counter configuration of Figure 1.

Synchronous Cascading

- When operating synchronous counters in a cascaded configuration, it is necessary to use the count enable and the terminal count functions to achieve higher-modulus operation.

- On some devices the count enable is labeled simply CTEN (or some other designation such as G), and terminal count (TC) is analogous to ripple clock output (RCO) on some IC counters.

- Figure 3 shows two decade counters connected in cascade.

- The terminal count (TC) output of counter 1 is connected to the count enable (CTEN) input of counter 2.

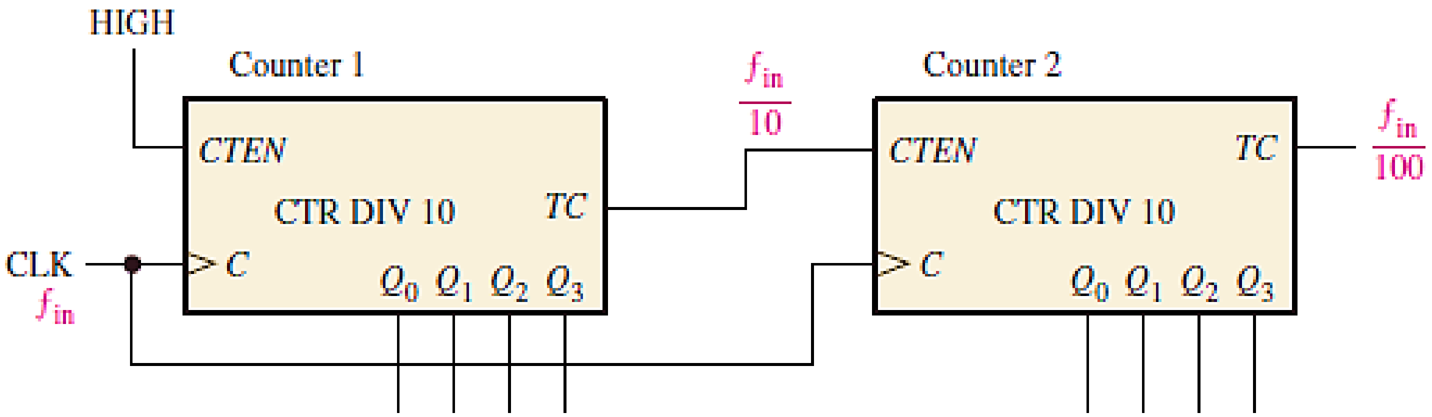

FIGURE 3 A modulus-100 counter using two cascaded decade counters.

- Counter 2 is inhibited by the LOW on its *CTEN* input until counter 1 reaches its last, or terminal, state and its terminal count output goes HIGH.
- This HIGH now enables counter 2, so that when the first clock pulse after counter 1 reaches its terminal count (CLK10), counter 2 goes from its initial state to its second state.
- Upon completion of the entire second cycle of counter 1 (when counter 1 reaches terminal count the second time), counter 2 is again enabled and advances to its next state.

- This sequence continues. Since these are decade counters, counter 1 must go through ten complete cycles before counter 2 completes its first cycle.

- In other words, for every ten cycles of counter 1, counter 2 goes through one cycle. Thus, counter 2 will complete one cycle after one hundred clock pulses.

- The overall modulus of these two cascaded counters is 10 * 10 = 100.

- Cascaded counters are often used to divide a high-frequency clock signal to obtain highly accurate pulse frequencies.

- Cascaded counter configurations used for such purposes are sometimes called *countdown chains.*

- **For example**

suppose that you have a basic clock frequency of 1 MHz and you wish to obtain 100 kHz, 10 kHz, and 1 kHz; a series of cascaded decade counters can be used. If the 1 MHz signal is divided by 10, the output is 100 kHz. Then if the 100 kHz signal is divided by 10, the output is 10 kHz. Another division by 10 produces the 1 kHz frequency.

- General implementation of this countdown chain is shown in Figure 4.

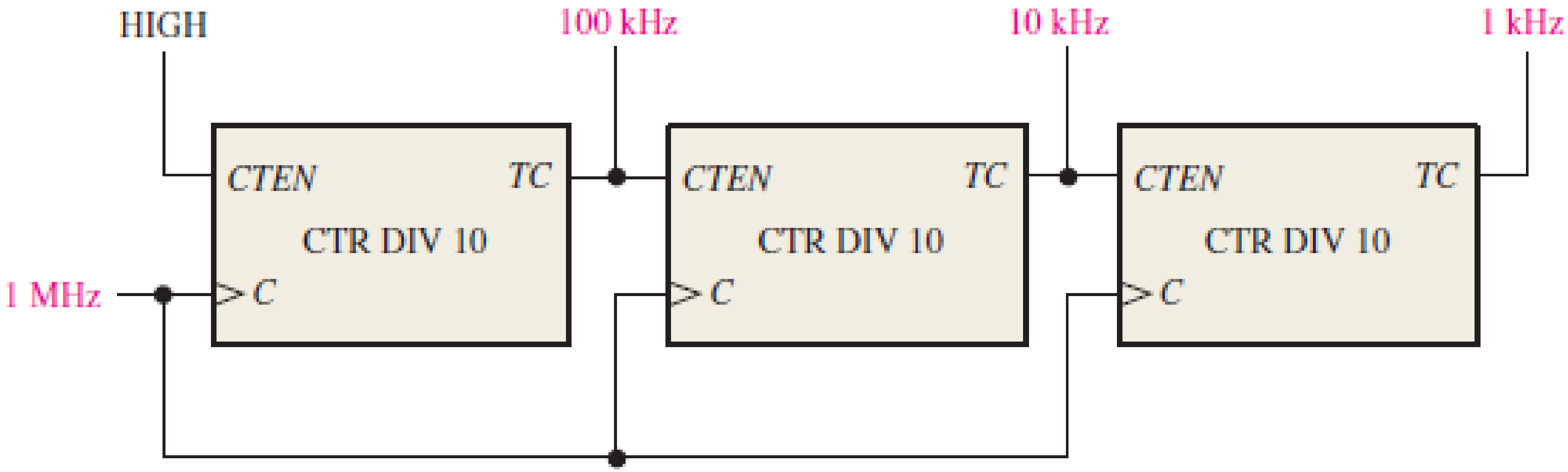

FIGURE 4 Three cascaded decade counters forming a divide-by-1000 frequency divider with intermediate divide-by-10 and divide-by-100 outputs.

Example

Determine the overall modulus of the two cascaded counter configurations in Figure 5.

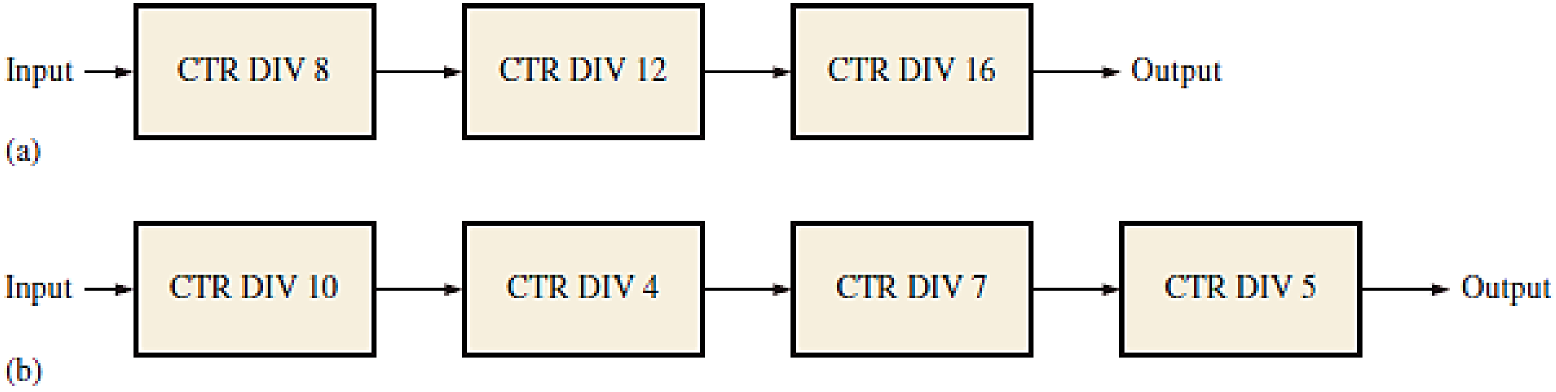

FIGURE 5

Solution

- In Figure 5 (a), the overall modulus for the 3-counter configuration is

8 * 12 * 16 = **1536**

- In Figure 5 (b), the overall modulus for the 4-counter configuration is

10 * 4 * 7 * 5 = **1400**

EXAMPLE

Use 74HC190 up/down decade counters connected in the UP mode to obtain a 10 kHz waveform from a 1 MHz clock. Show the logic diagram.

Solution

To obtain 10 kHz from a 1 MHz clock requires a division factor of 100. Two 74HC190 counters must be cascaded as shown in Figure 6. The left counter produces a terminal count (*MAX*/*MIN*) pulse for every 10 clock pulses. The right counter produces a terminal count (*MAX*/*MIN*) pulse for every 100 clock pulses.

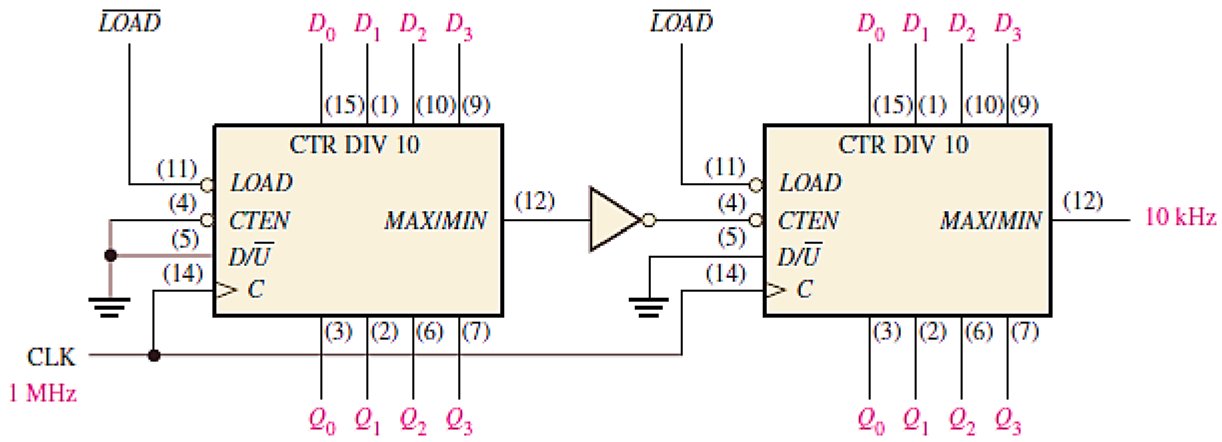

FIGURE 6 A divide-by-100 counter using two 74HC190 up/down decade counters connected for the up sequence

Cascaded Counters with Truncated Sequences

- The preceding discussion has shown how to achieve an overall modulus (divide-by-factor) that is the product of the individual moduli of all the cascaded counters. This can be considered *full-modulus cascading.*
- Often an application requires an overall modulus that is less than that achieved by full modulus cascading. That is, a truncated sequence must be implemented with cascaded counters.
- To illustrate this method, we will use the cascaded counter configuration in Figure 7. This particular circuit uses four 74HC161 4-bit synchronous binary counters.
- If these four counters (sixteen bits total) were cascaded in a full-modulus arrangement, the modulus would be

$$\mathbf{2_{16} = 65,536}$$

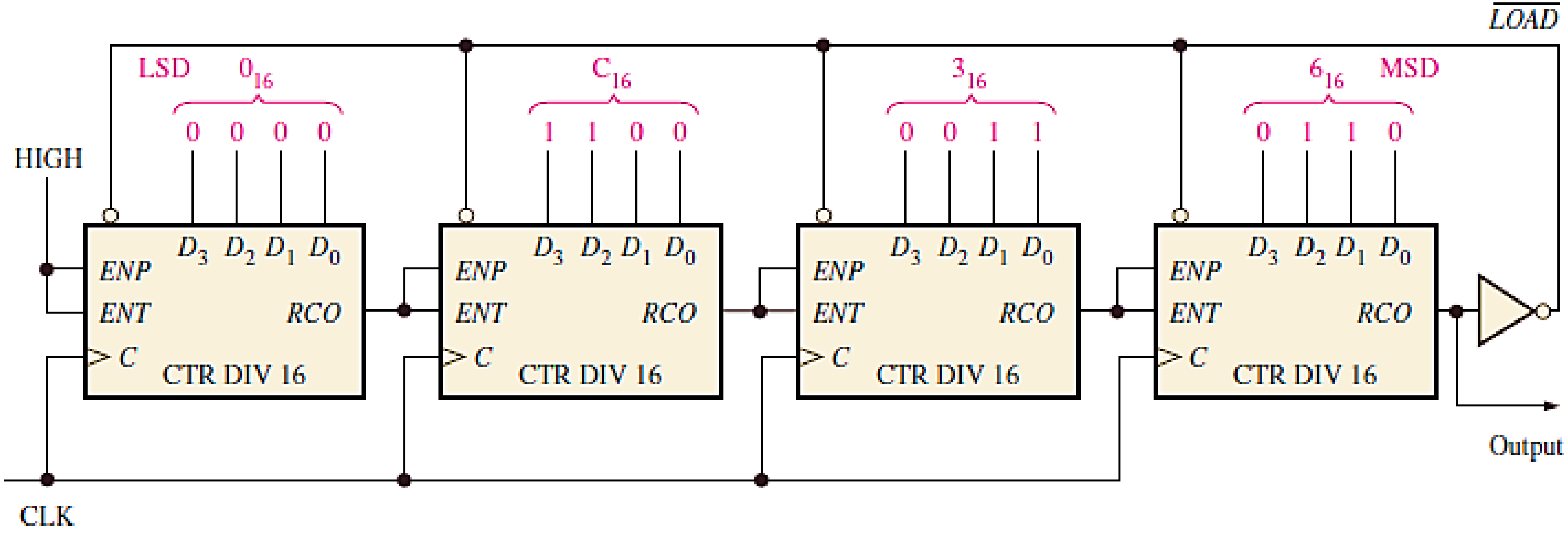

FIGURE 7 A divide-by-40,000 counter using 74HC161 4-bit binary counters. Note that each of the parallel data inputs is shown in binary order (the right-most bit D_0 is the LSB in each counter).

- Let’s assume that a certain application requires a divide-by-40,000 counter (modulus 40,000). The difference between 65,536 and 40,000 is 25,536, which is the number of states that must be *deleted* from the full-modulus sequence. The technique used in the circuit of Figure 7 is to preset the cascaded counter to 25,536 (63C0 in hexadecimal) each time it recycles, so that it will count from 25,536 up to 65,535 on each full cycle. Therefore, each full cycle of the counter consists of 40,000 states.

- Notice in Figure 7 that the RCO output of the right-most counter is inverted and applied to the LOAD input of each 4-bit counter. Each time the count reaches its terminal value of 65,535, which is 11111111111111112, RCO goes HIGH and causes the number on the parallel data inputs (63C016) to be synchronously loaded into the counter with the clock pulse. Thus, there is one RCO pulse from the right-most 4-bit counter for every40,000 clock pulses.

- With this technique any modulus can be achieved by synchronous loading of the counter to the appropriate initial state on each cycle.

Counter Decoding

- In many applications, it is necessary that some or all of the counter states be decoded. The decoding of a counter involves using decoders or logic gates to determine when the counter is in a certain binary state in its sequence.
- For instance, the terminal count function previously discussed is a single decoded state (the last state) in the counter sequence.
- Suppose that you wish to decode binary state 6 (110) of a 3-bit binary counter. When $Q_2 = 1$, $Q_1 = 1$, and $Q_0 = 0$, a HIGH appears on the output of the decoding gate, indicating that the counter is at state 6. This can be done as shown in Figure 8. This is called *active-HIGH decoding.* Replacing the AND gate with a NAND gate provides active-LOW decoding.

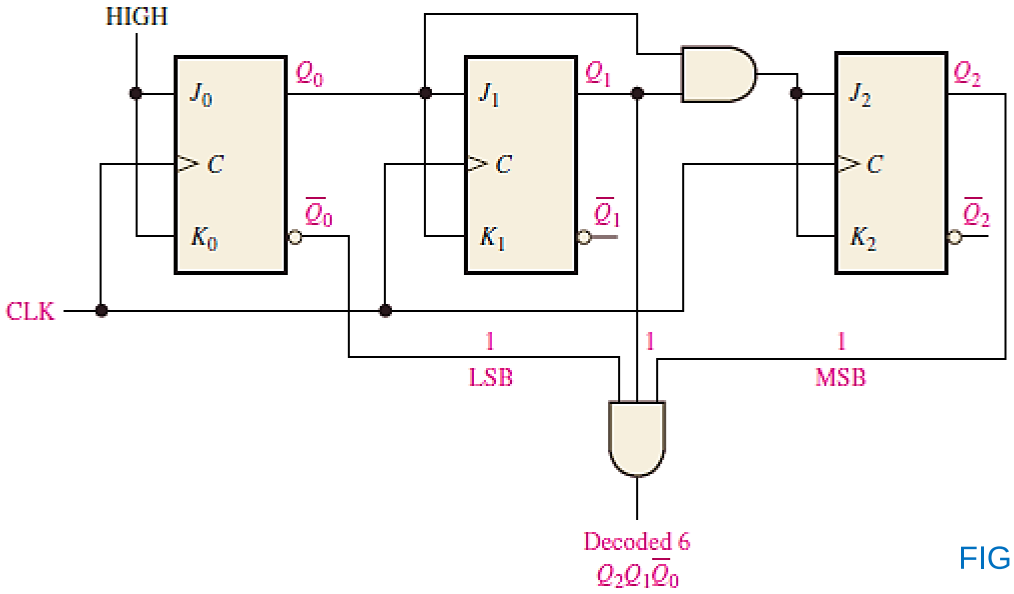

FIGURE 8 Decoding of state 6 (110).

Example

Implement the decoding of binary state 2 and binary state 7 of a 3-bit synchronous counter. Show the entire counter timing diagram and the output waveforms of the decoding gates. Binary 2 = $\overline{Q_2}Q_1\overline{Q_0}$ and binary 7 = $Q_2Q_1Q_0$.

Solution

See Figure 9–43. The 3-bit counter was originally discussed in A 3-Bit Synchronous Binary Counter

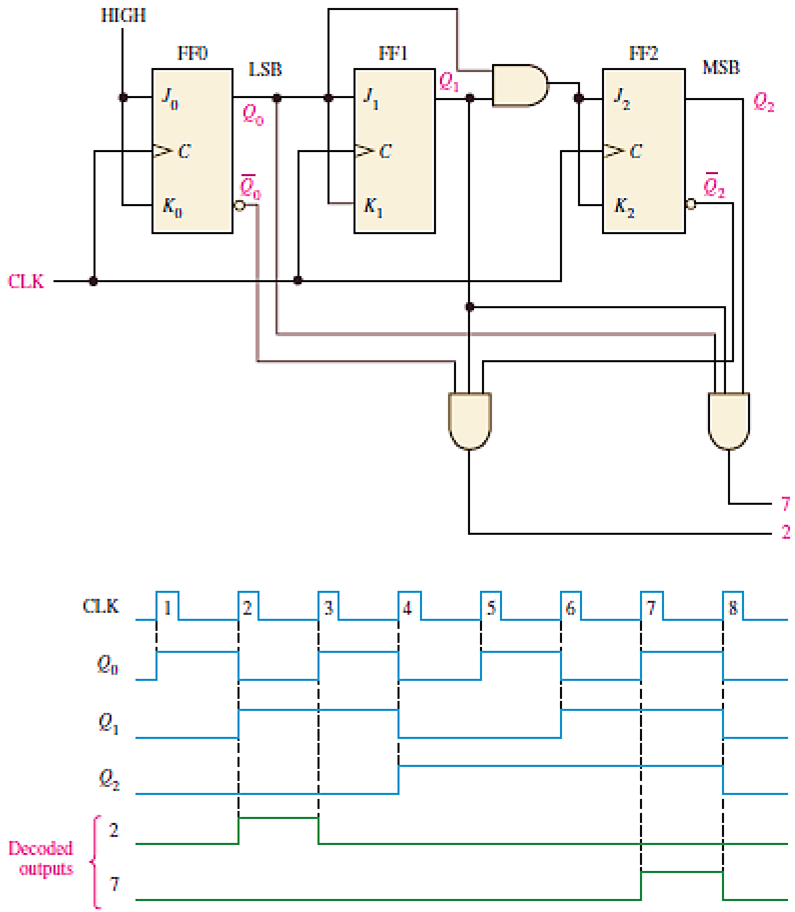

FIGURE 9 A 3-bit counter with active-HIGH decoding of count 2 and count 7.

Counter Applications

The digital counter is a useful and versatile device that is found in many applications. In this section, some representative counter applications are presented.

A Digital Clock

- A common example of a counter application is in timekeeping systems. Figure 10 is a simplified logic diagram of a digital clock that displays seconds, minutes, and hours.
- First, a 60 Hz sinusoidal ac voltage is converted to a 60 Hz pulse waveform and divided down to a 1 Hz pulse waveform by a divide-by-60 counter formed by a divide-by-10 counter followed by a divide-by-6 counter. Both the *seconds* and *minutes* counts are also produced by divide-by-60 counters, the details of which are shown in Figure 11.
- These counters count from 0 to 59 and then recycle to 0; synchronous decade counters are used in this particular implementation. Notice that the divide-by-6 portion is formed with a decade counter with a truncated sequence achieved by using the decoder count 6 to asynchronously clear the counter. The terminal count, 59, is also decoded to enable the next counter in the chain.

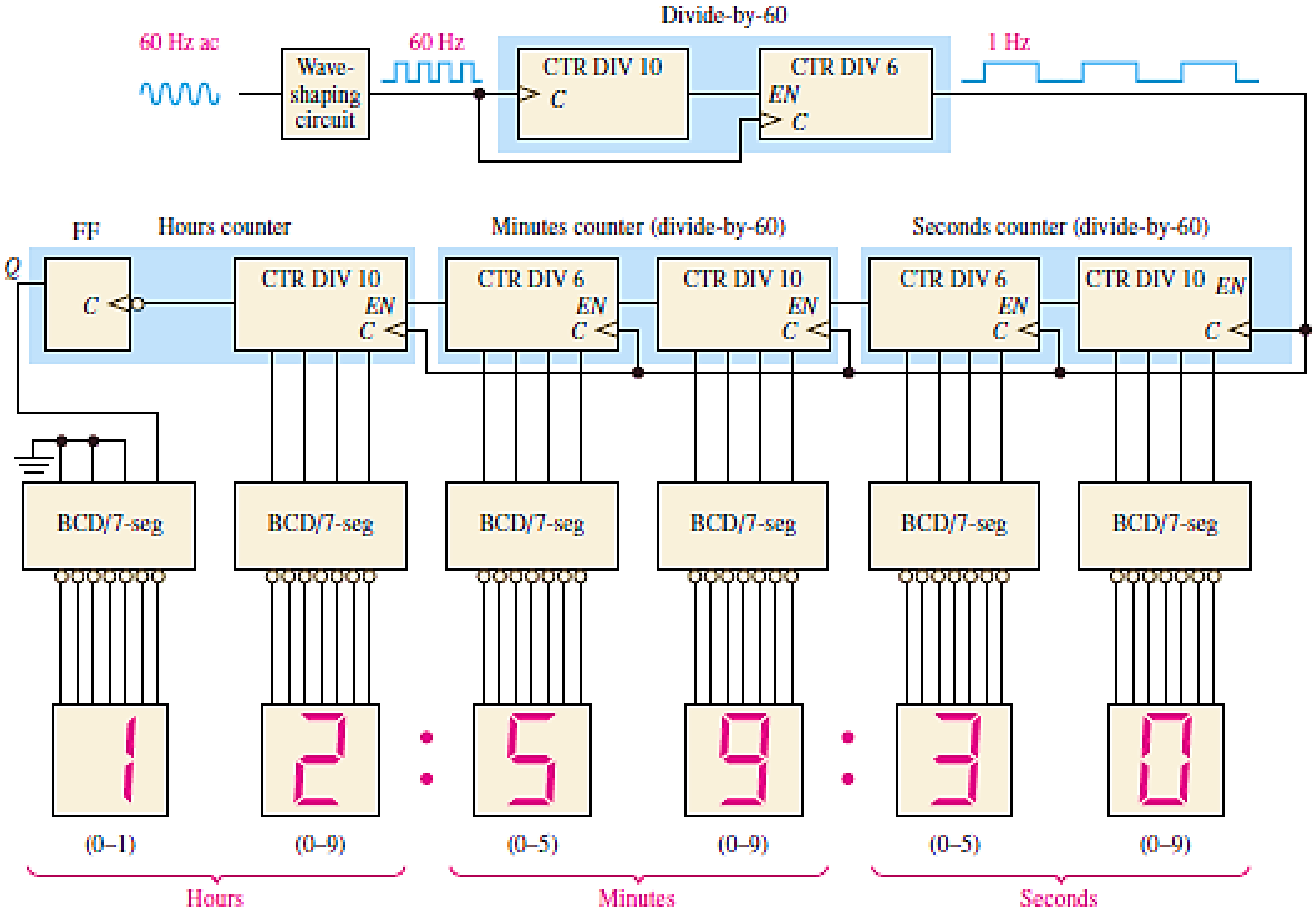

FIGURE 10 Simplified logic diagram for a 12-hour digital clock. Logic details using specific devices are shown in Figures 11 and 12.

The *hours* counter is implemented with a decade counter and a flip-flop as shown in Figure 12. Consider that initially both the decade counter and the flip-flop are RESET, and the decode-12 gate and decode-9 gate outputs are HIGH. The decade counter advances through all of its states from zero to nine, and on the clock pulse that recycles it from nine back to zero, the flip-flop goes to the SET state ($J = 1$, $K = 0$). This illuminates a 1 on the tens-of-hours display. The total count is now ten (the decade counter is in the zero state and the flip-flop is SET).

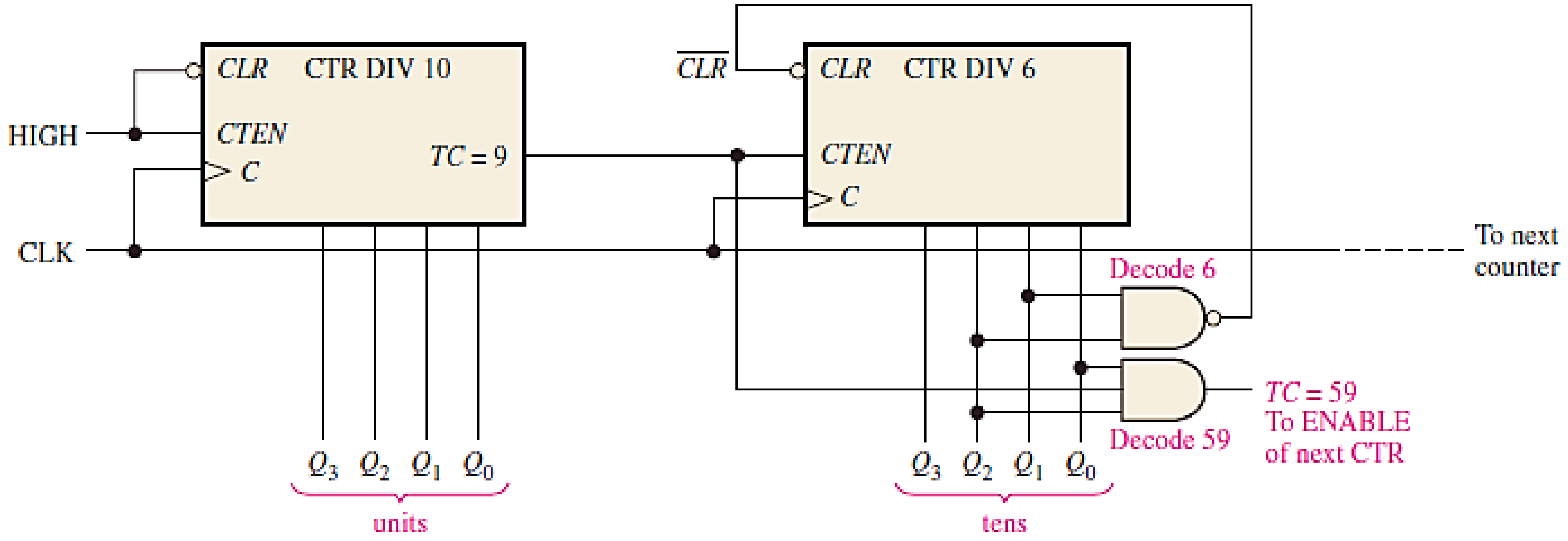

FIGURE11 Logic diagram of typical divide-by-60 counter using synchronous decade counters. Note that the outputs are in binary order (the right-most bit is the LSB).

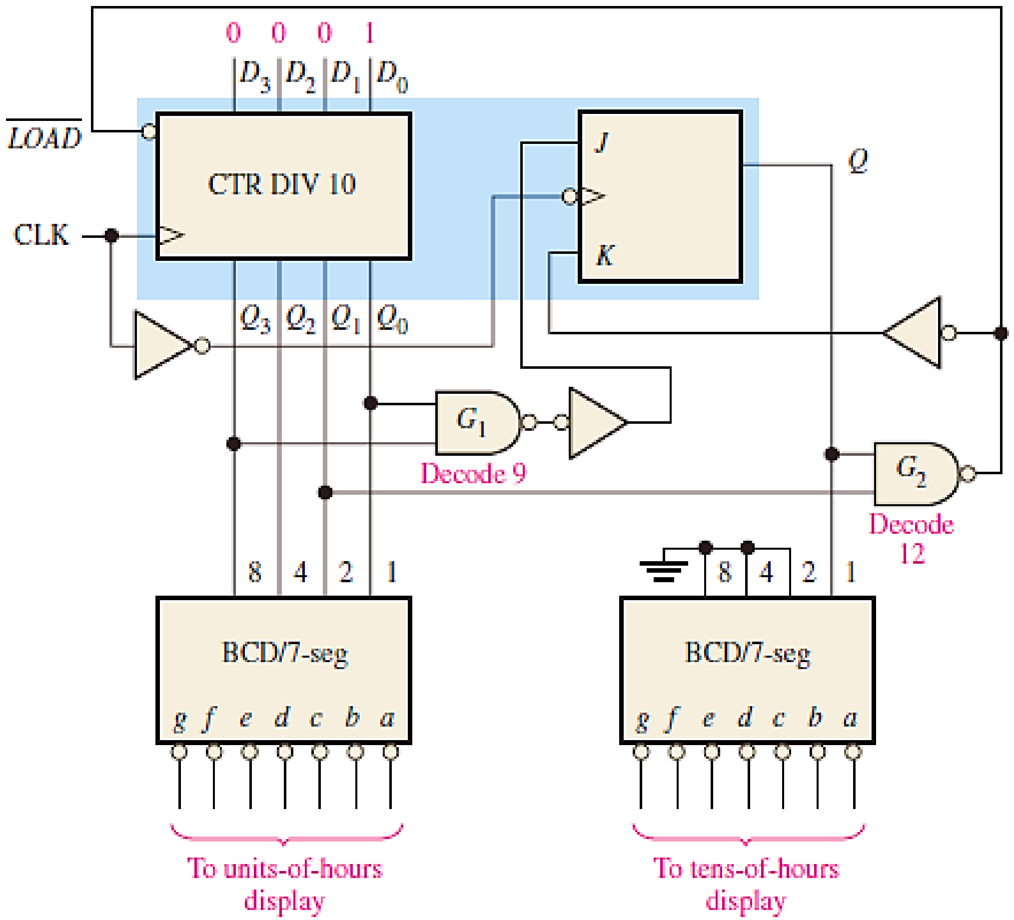

FIGURE 12 Logic diagram for hours counter and decoders. Note that on the counter inputs and outputs, the right-most bit is the LSB.

Next, the total count advances to eleven and then to twelve. In state 12 the Q_2 output of the decade counter is HIGH, the flip-flop is still SET, and thus the decode-12 gate output is LOW. This activates the $\overline{LOAD}$ input of the decade counter. On the next clock pulse, the decade counter is preset to 0001 from the data inputs, and the flip-flop is RESET (J = 0, K = 1). As you can see, this logic always causes the counter to recycle from twelve back to one rather than back to zero.