Montogue

# Quiz EL305
# Registers and Counters

Lucas Monteiro Nogueira

## PROBLEMS

### Problem 1

**Problem 1.1:** Generally, a shift register is constructed with:
**A)** *T* flip-flops.
**B)** *D* flip-flops.
**C)** *JK* flip-flops.
**D)** *SR* flip-flops.
**Problem 1.2:** In which of the following kinds of register is the data entered one bit at a time and unloaded all at once?
**A)** Serial-in, serial-out register
**B)** Serial-in, parallel-out register
**C)** Parallel-in, serial-out register
**D)** Parallel-in, parallel-out register
**Problem 1.3:** In shift registers, shifting binary data to the left by one bit position is equivalent to:
**A)** Addition of 2
**B)** Subtraction of 2
**C)** Multiplication by 2
**D)** Division by 2
**Problem 1.4:** The sequence 1011 is applied to the input of a 4-bit serial right-shift register that is initially cleared. What is the state of the register after three clock pulses?
**A)** 1011
**B)** 1101
**C)** 1100
**D)** 0110
**Problem 1.5:** In order to use a shift register as a counter,
**A)** The serial output of the register is connected back to its serial input.
**B)** The input is applied to the parallel inputs and the output is taken from the serial data output.
**C)** A serial-in, serial-out register must be used.
**D)** The register's serial input is the counter input and the serial output is the counter output.
**Problem 1.6:** A modulus-12 counter must have:
**A)** 3 flip-flops.
**B)** 4 flip-flops.
**C)** 12 flip-flops.
**D)** Synchronous clocking.
**Problem 1.7:** A 4-bit ripple counter consists of flip-flops that each have a propagation delay from clock to *Q* output of 15 nanoseconds. For the counter to recycle from 1111 to 0000, it takes a total of:
**A)** 15 ns
**B)** 30 ns
**C)** 45 ns
**D)** 60 ns
**Problem 1.8:** A decade counter is clocked from an 80-kHz signal. Determine its output frequency.
**A)** 4 kHz
**B)** 8 kHz
**C)** 40 kHz
**D)** 80 kHz

**Problem 1.9:** Assume that a 5-bit binary counter starts in the 00000 state. What will the counter state be after 183 pulses?
**A)** 10110
**B)** 10111
**C)** 11110
**D)** 11111

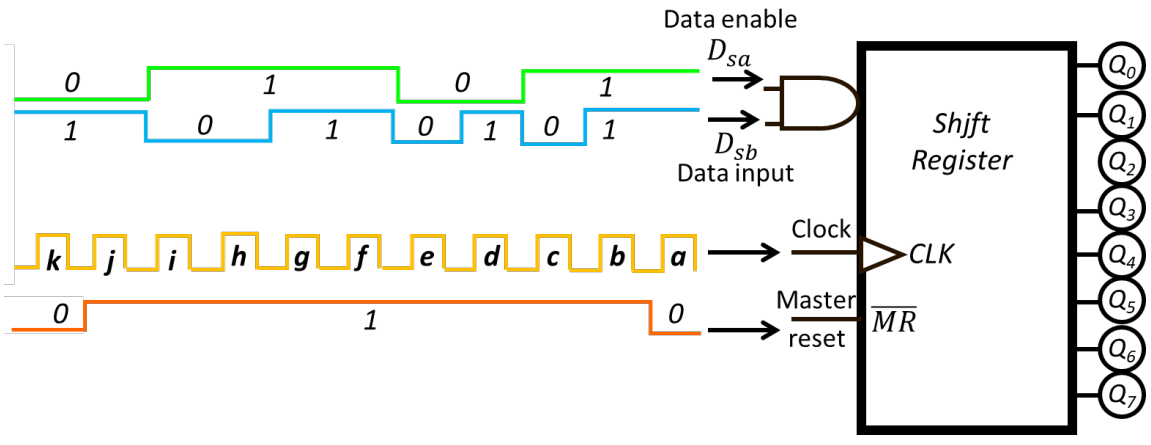
## Problem 2 (Modified from Tokheim, 1994)

The 74HC164 CMOS integrated circuit is an 8-bit serial-in, parallel-out edge-triggered shift register. The device permits only serial data input; specifically, there are two data inputs, $D_{sa}$ and $D_{sb}$; since these are ANDed together, one input can be used as a (active-high) data enable input while serial data is fed into the second data input. If no data enable operation is used, both data inputs are tied together and used as a serial data input. The output is an 8-bit string. Each clock pulse shifts data one position to the right (from $Q_0$ to $Q_7$ in the figure below) in the shift register. The master reset $\overline{MR}$ pin on the 74HC164 IC is an active-low input which resets all eight flip-flops and clears the outputs to 0. The master reset is an asynchronous input that overrides all other inputs. A truth table detailing the operating modes of the 74HC164 IC is shown below.

| Operating modes | Inputs | | | | Outputs | |
|---|---|---|---|---|---|---|
| | $\overline{MR}$ | $CP$ | $D_{sa}$ | $D_{sb}$ | $Q_0$ | $Q_1$-$Q_7$ |
| Reset (clear) | $L$ | × | × | × | $L$ | $L$-$L$ |
| Shift | $H$ | ↑ | $L$ | × | $L$ | $q_0$-$q_6$ |
| | $H$ | ↑ | $H$ | $l$ | $L$ | $q_0$-$q_6$ |
| | $H$ | ↑ | $H$ | $h$ | $H$ | $q_0$-$q_6$ |

*Legend:*
$H$ = high voltage level
$h$ = high voltage level one setup time prior to the $L$-to-$H$ clock transition
$L$ = low voltage level
$l$ = low voltage level one setup time prior to the $L$-to-$H$ clock transition
$q$ = Lowercase letters indicate the state of the referenced input one setup time prior to the $L$-to-$H$ clock transition
× = Don't care
↑ = $L$-to-$H$ clock transition

Using the table above, list the state of the output indicators after each clock pulse (yellow waveform) for the 74HC164 register illustrated below.




## Problem 3 (Modified from Tokheim, 1994)

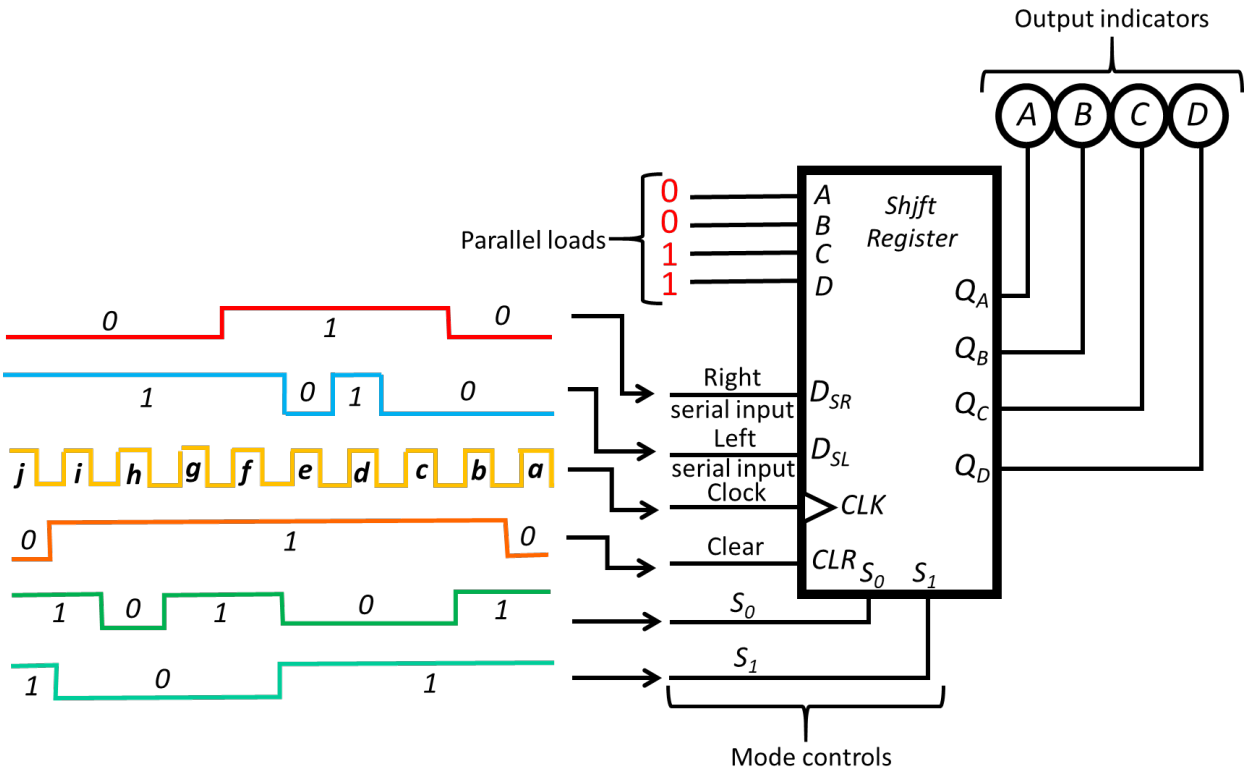
A block logic symbol for the commercial TTL 74194 4-bit universal shift register is illustrated below. The 74194 register has 10 inputs and 4 outputs. The outputs are connected to the normal ($Q$) outputs of each flip-flop inside the integrated circuit. In the 74194 register, the parallel-load inputs $A$, $B$, $C$ and $D$ are the top four inputs. The next two inputs $D_{SR}$ and $D_{SL}$ are for feeding data into the register serially (i.e., one bit at a time). The shift-right serial input $D_{SR}$ feeds bits into position $A$ ($Q_A$) as the register is shifted to the right. The shift-left serial input $D_{SL}$ feeds bits into position $D$ ($Q_D$) as the register is shifted to the left. The clock input (CLK) triggers the four flip-flops on the low-to-high transition of the clock pulse. When activated with a low, the clear (CLR) input resets each flip-flop to 0. The mode controls $S_0$ and $S_1$ combine their levels to instruct the register through a gating network to shift right, shift left, parallel-load, or hold (do nothing). Of course, the 74194, which is a TTL IC, has a +5 V power supply connection. The power-supply connections are not usually shown on the logic symbol.

A mode-select function table for the 74194 shift register is provided below.

| Operating mode | Inputs | | | | | | | Outputs | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| | CLK | CLR | $S_1$ | $S_0$ | $D_{SR}$ | $D_{SL}$ | $D_n$ | $Q_0$ | $Q_1$ | $Q_2$ | $Q_3$ |
| Reset (clear) | × | *L* | × | × | × | × | × | *L* | *L* | *L* | *L* |
| Hold (do nothing) | × | *H* | *l** | *l** | × | × | × | $q_0$ | $q_1$ | $q_2$ | $q_3$ |
| Shift-left | ↑ | *H* | *h* | *l** | × | *l* | × | $q_1$ | $q_2$ | $q_3$ | *L* |
| | ↑ | *H* | *h* | *l** | × | *h* | × | $q_1$ | $q_2$ | $q_3$ | *H* |
| Shift-right | ↑ | *H* | *l** | *h* | *l* | × | × | *L* | $q_0$ | $q_1$ | $q_2$ |
| | ↑ | *H* | *l** | *h* | *h* | × | × | *H* | $q_0$ | $q_1$ | $q_2$ |
| Parallel load | ↑ | *H* | *h* | *h* | × | × | $d_n$ | $d_0$ | $d_1$ | $d_2$ | $d_3$ |

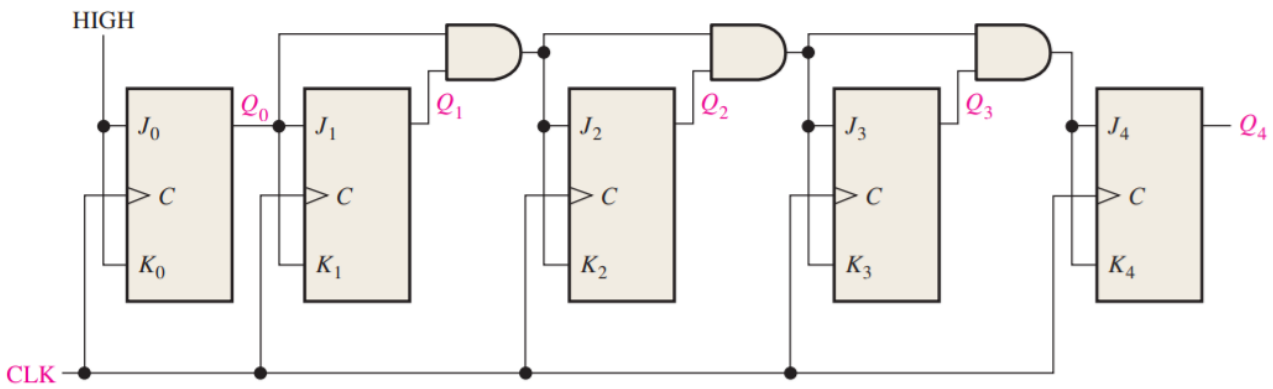
*Legend:*
*H* = high voltage level
*h* = high voltage level one setup time prior to the *L*-to-*H* clock transition
*L* = low voltage level
*l* = low voltage level one setup time prior to the *L*-to-*H* clock transition
$d_n(q_n)$ = (Lowercase letters indicate the state of the referenced input [or output] one setup time prior to the *L*-to-*H* clock transition)
× = Don't care
↑ = *L*-to-*H* clock transition
*The *H*-to-*L* transition of the $S_0$ and $S_1$ inputs on the 74194 should only take place while *CLK* is high for conventional operation.

Using the table above, list the operating mode and the state of the output indicators after each clock pulse (yellow waveform) for the 74194 register illustrated in continuation.



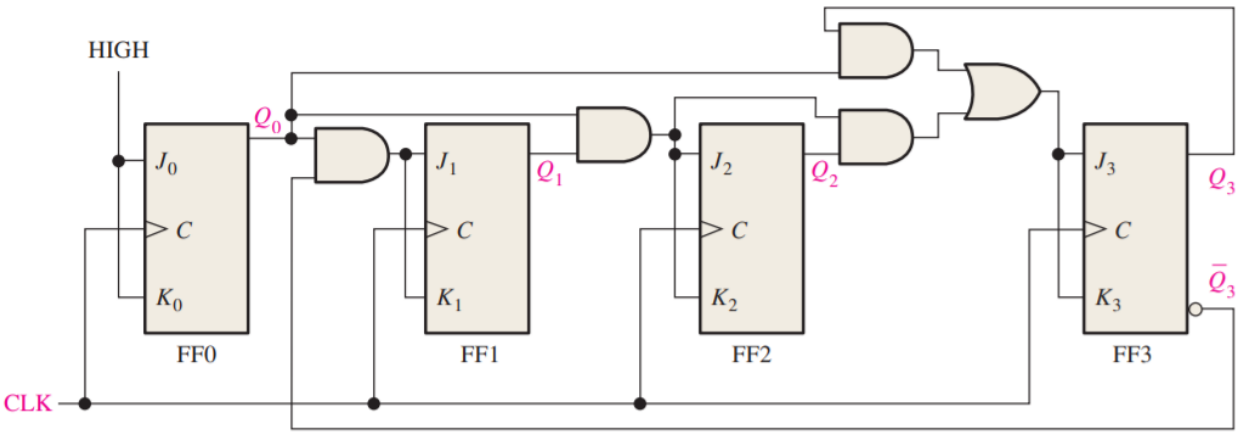

## Problem 4 (Floyd, 2015)

Show the complete timing diagram for the 5-stage synchronous binary counter illustrated below. Verify that the waveforms of the *Q* outputs represent the proper binary number after each clock pulse.



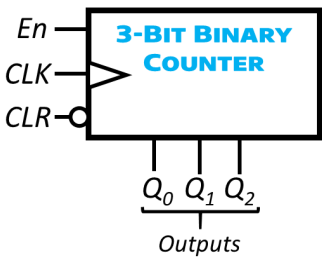
## Problem 5 (Floyd, 2015)

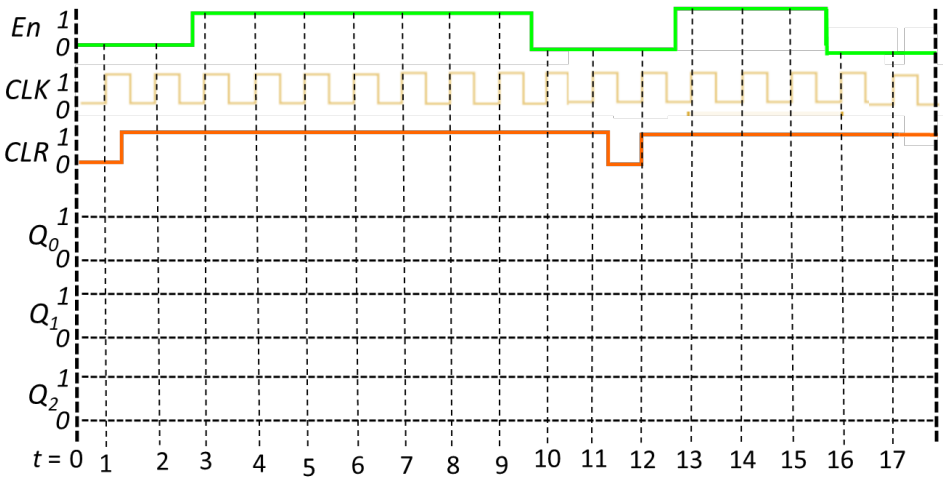
**Problem 5.1**: Use logic equations to show that the decade counter illustrated below progresses through a BCD sequence.
**Problem 5.2**: Show the complete timing diagram for the BCD decade counter illustrated below.




## Problem 6 (Modified from Floyd, 2015)

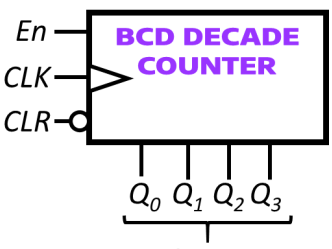
The waveforms illustrated below are applied to the count enable (*En*), clear (*CLR*, which in this case is active-low and synchronous), and clock inputs of a 3-bit rising-edge triggered synchronous binary counter. Show the counter output waveforms in proper relation to these inputs. $Q_0$ and $Q_2$ are the least- and most-significant bits, respectively. The initial output is 000.

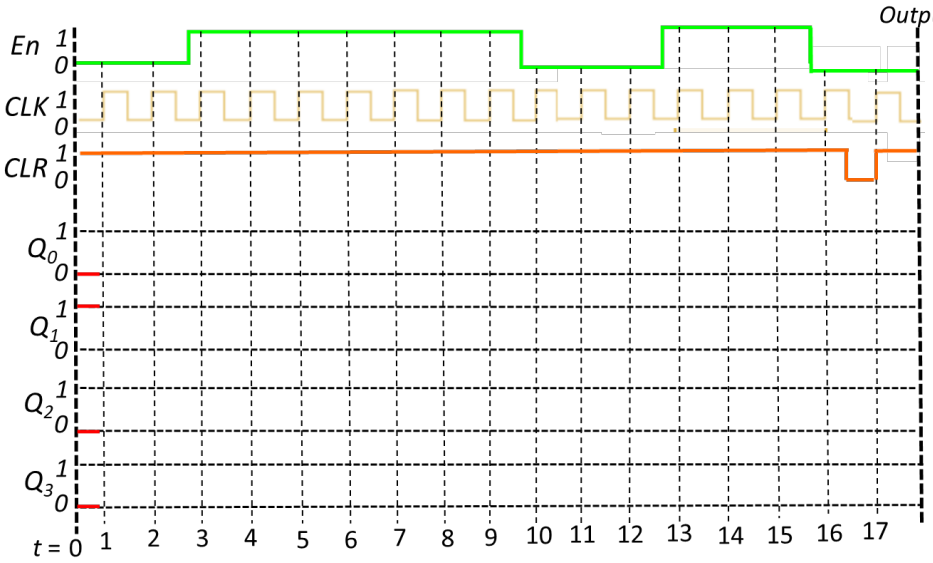



## Problem 7 (Modified from Floyd, 2015)

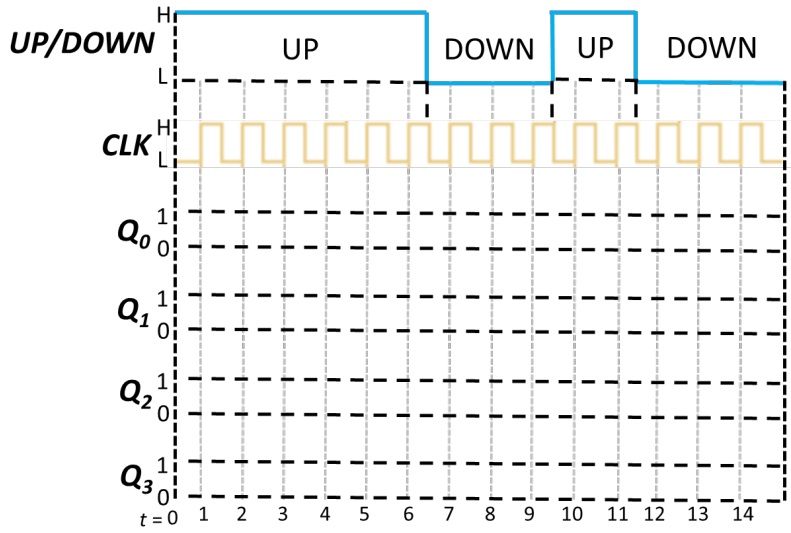
The waveforms illustrated below are applied to the count enable (*En*), clear (*CLR*, which in this case is active-low and synchronous), and clock inputs of a rising-edge triggered BCD decade counter. Determine the waveforms for each of the counter outputs ($Q_0$, $Q_1$, $Q_2$, and $Q_3$). $Q_0$ and $Q_3$ are the least- and most-significant bits, respectively. **The initial output is 0010.**



## Problem 8 (Modified from Floyd, 2015)

Show a timing diagram and determine the sequence of a 4-bit synchronous binary up/down counter if the clock and UP/DOWN inputs have the waveforms illustrated below. The counter starts in the all-0s state and is positive-edge triggered.



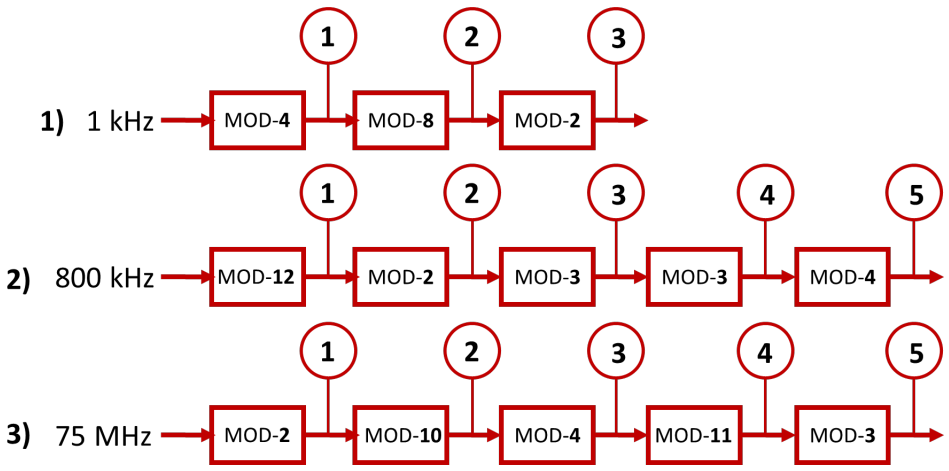

## Problem 9 (Modified from Floyd, 2015)

Show a complete timing diagram for a 3-bit up/down counter that goes through the following sequence. Indicate when the counter is in the UP mode and when it is in the DOWN mode. Assume positive-edge triggering.

**0, 1, 2, 3, 2, 1, 2, 3, 4, 5, 6, 5, 4, 3, 2, 1, 0**

## Problem 10

The following are cascaded counter configurations. For each system, determine the frequency of the waveform at each point indicated by a circled number, and calculate the overall modulus.




# SOLUTIONS

## P.1 ➡ *Solution*

**Problem 1.1:** A shift register is essentially a chain of *D* flip-flops that are each connected to a common clock.

▶ The correct answer is **B**.

**Problem 1.2:** The data of a SIPO register is entered one bit at a time (hence the phrase *serial-in*) and outputted all at once (hence the phrase *parallel-out*).

▶ The correct answer is **B**.

**Problem 1.3:** Shifting a binary number to the left in a shift register is equivalent to multiplying the number by 2.

▶ The correct answer is **C**.

**Problem 1.4:** The serial register begins at 0000 and is to load a 1011 input. After the first clock pulse, the register is updated to state 1000; at the second, it becomes 1100; at the third, the register becomes 0110.

▶ The correct answer is **D**.

**Problem 1.5:** To use a shift register as a counter, the serial output of the register is connected to its serial input.

▶ The correct answer is **A**.

**Problem 1.6:** A mod-12 counter can count up to 11. $N$ flip-flops are used to count to $2^N$. The minimum number of flip-flops required to count to 11 is 4.

▶ The correct answer is **B**.

**Problem 1.7:** Recycling from 1111 to 0000 takes $4 \times 15 = 60$ ns.

▶ The correct answer is **D**.

**Problem 1.8:** A decade counter is essentially a modulus-10 counter, and as such has output frequency of 80/10 = 8 kHz.

▶ The correct answer is **B**.

**Problem 1.9:** A 5-bit binary counter will reset every $2^5 = 32$ pulses. Thus, the counter resets at pulses No. 32, 64, 96, 128, and 160. At pulse No. 183, the counter will be at state No. 183 – 160 = 23, which in binary form equals 10111.

▶ The correct answer is **B**.

## P.2 ➡ *Solution*

First, note that at clock pulse *a* the master reset waveform is low; this asynchronous input overrides other signals and clears the output word back to 0000 0000. At pulse *b*, the master reset pulse train returns to 1 and hence enables the register to shift data normally. The data enable signal $D_{sa}$ is high, which indicates that the register can accept a bit entry from the other input signal, $D_{sb}$; since $D_{sb}$ is high, a 1 is assigned to the leftmost bit of the output and the output becomes 1000 0000. At pulse *c*, $D_{sa}$ is still high and the register can accept another bit from $D_{sb}$; because $D_{sb}$ is low, a 0 is assigned to the leftmost bit of the output, which becomes 0100 0000. At pulse *d*, note that the $D_{sa}$ pulse train goes low; this prevents the register from loading an input from $D_{sb}$, so the register will shift right and by default assign a zero to the leftmost bit; the output changes from 0100 0000 to 0010 0000. At pulse *e*, $D_{sa}$ is still disabled and the register shifts right without reading data from $D_{sb}$; the output changes from 0010 0000 to 0001 0000. At pulse *f*, $D_{sa}$ goes high again, enabling the register to read a bit from $D_{sb}$; since $D_{sb}$ is high, a 1 is assigned to the leftmost bit of the output, which is updated to 1000 1000. At pulse *g*, another 1 is loaded into the output, changing it to 1100 0100. At pulse *h*, input pulse-train $D_{sb}$ goes low, so a 0 is loaded into the output; the output word then becomes 0110 0010. At pulse *i*, another 0 is loaded into the output, changing it to 0011 0001. At pulse *j*, the data enable pulse-train goes low again, preventing the output from reading data sent via $D_{sb}$; the register then shifts right to 0001 1000. At pulse *k*, the master reset pulse train goes low, resetting the register and clearing the output back to 0000 0000. The output changes are summarized below.

| Pulse | Output indicator |
|---|---|
| *a* | 0000 0000 |
| *b* | 1000 0000 |
| *c* | 0100 0000 |
| *d* | 0010 0000 |
| *e* | 0001 0000 |
| *f* | 1000 1000 |
| *g* | 1100 0100 |
| *h* | 0110 0010 |
| *i* | 0011 0001 |
| *j* | 0001 1000 |

## P.3 ➡ *Solution*

Before anything else, check the clear waveform; at pulse *a*, this signal is low, and hence causes the register to reset all outputs to 0. Next, at pulse *b* the clear signal goes high and enables the register to process data. At this point, the mode controls $S_0$ and $S_1$ become crucial. On pulse *b*, both $S_0$ and $S_1$ are high, which indicates that the register is to accept a parallel load – in this case, a 0011 word. At pulse *c*, $S_0$ goes low and $S_1$ remains high, which instructs the register to perform a shift-left operation; a 0 is assigned to the rightmost bit because $D_{SL}$ is low. The output, then, changes from 0011 to 0110. At pulse *d*, $S_0$ remains low and $S_1$ continues to be high, so the register will perform another shift-left operation; since $D_{SL}$ is now high, a 1 is assigned to the rightmost bit, changing the output from 0110 to 1101. At pulse *e*, $S_0$ is still low and $S_1$ is still high, so yet another shift-left operation occurs; the bits change one position to the left and, because $D_{SL}$ is low, a 0 is assigned to the rightmost bit. The output is now 1010. Next, at pulse *f* mode

controls $S_0$ and $S_1$ are respectively set to 1 and 0, which puts the register in shift-right mode. Since $D_{SR}$ is high, a 1 is to be assigned to the leftmost bit; the output word then changes from 1010 to 1101. Shift-right operation continues at pulse *g*; with $D_{SR}$ going low, a 0 is assigned to the leftmost bit and the output changes from 1101 to 0110. Next, at pulse *h* both mode controls are set to 0, which, referring to the table above, puts the register in hold mode; as a result, the output remains unchanged. At pulse *i*, $S_0$ goes high and $S_1$ stays low, placing the register in shift-right mode once again; because $D_{SR}$ is low, a 0 is assigned to the leftmost bit, changing the output from 0110 to 0011. Lastly, at pulse *j* the clear input goes low and overrides the mode controls; as a result, the output is reset to 0000. The following table summarizes the operating modes and output words shown in the indicator at each of the 10 clock pulses analyzed.

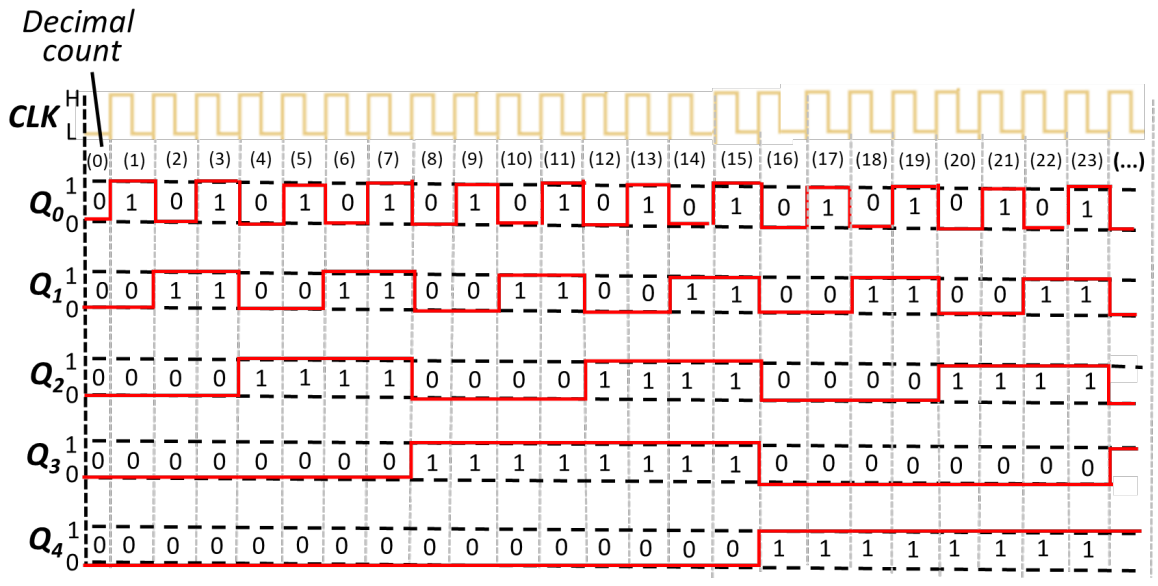
| Pulse | Operating mode | Output indicator |
|---|---|---|
| *a* | Clear | 0 0 0 0 |
| *b* | Parallel load | 0 0 1 1 |
| *c* | Shift-left | 0 1 1 0 |
| *d* | Shift-left | 1 1 0 1 |
| *e* | Shift-left | 1 0 1 0 |
| *f* | Shift-right | 1 1 0 1 |
| *g* | Shift-right | 0 1 1 0 |
| *h* | Hold mode | 0 1 1 0 |
| *i* | Shift-right | 0 0 1 1 |
| *j* | Clear | 0 0 0 0 |

## P.4 ➡ *Solution*

Since the device is a 5-stage binary counter, it counts from $(00000)_2$ to $(11111)_2$. Before sketching the output waveforms for such a large pulse train, it is convenient to draw up a state table.

| Clock | $Q_4$ | $Q_3$ | $Q_2$ | $Q_1$ | $Q_0$ | Decimal count |
|---|---|---|---|---|---|---|
| 1 | 0 | 0 | 0 | 0 | 1 | 1 |
| 2 | 0 | 0 | 0 | 1 | 0 | 2 |
| 3 | 0 | 0 | 0 | 1 | 1 | 3 |
| 4 | 0 | 0 | 1 | 0 | 0 | 4 |
| 5 | 0 | 0 | 1 | 0 | 1 | 5 |
| 6 | 0 | 0 | 1 | 1 | 0 | 6 |
| 7 | 0 | 0 | 1 | 1 | 1 | 7 |
| 8 | 0 | 1 | 0 | 0 | 0 | 8 |
| 9 | 0 | 1 | 0 | 0 | 1 | 9 |
| 10 | 0 | 1 | 0 | 1 | 0 | 10 |
| 11 | 0 | 1 | 0 | 1 | 1 | 11 |
| 12 | 0 | 1 | 1 | 0 | 0 | 12 |
| 13 | 0 | 1 | 1 | 0 | 1 | 13 |
| 14 | 0 | 1 | 1 | 1 | 0 | 14 |
| 15 | 0 | 1 | 1 | 1 | 1 | 15 |
| 16 | 1 | 0 | 0 | 0 | 0 | 16 |
| 17 | 1 | 0 | 0 | 0 | 1 | 17 |
| 18 | 1 | 0 | 0 | 1 | 0 | 18 |
| 19 | 1 | 0 | 0 | 1 | 1 | 19 |
| 20 | 1 | 0 | 1 | 0 | 0 | 20 |
| 21 | 1 | 0 | 1 | 0 | 1 | 21 |
| 22 | 1 | 0 | 1 | 1 | 0 | 22 |
| 23 | 1 | 0 | 1 | 1 | 1 | 23 |
| 24 | 1 | 1 | 0 | 0 | 0 | 24 |
| 25 | 1 | 1 | 0 | 0 | 1 | 25 |
| 26 | 1 | 1 | 0 | 1 | 0 | 26 |
| 27 | 1 | 1 | 0 | 1 | 1 | 27 |
| 28 | 1 | 1 | 1 | 0 | 0 | 28 |
| 29 | 1 | 1 | 1 | 0 | 1 | 29 |
| 30 | 1 | 1 | 1 | 1 | 0 | 30 |
| 31 | 1 | 1 | 1 | 1 | 1 | 31 |

The timing diagram for decimal count 0 to 23 are shown below; sketching the remainder of the diagram is left as an exercise to the reader.




## P.5 ➡ *Solution*

**Problem 5.1:** A BCD decade counter follows the 8421 BCD code, running from 0000 (decimal 0) to 1001 (decimal 9) and then cycling back to zero. For reference, the BCD code table is listed below.

| Decimal number | 8421 BCD |
|---|---|
| 0 | 0000 |
| 1 | 0001 |
| 2 | 0010 |
| 3 | 0011 |
| 4 | 0100 |
| 5 | 0101 |
| 6 | 0110 |
| 7 | 0111 |
| 8 | 1000 |
| 9 | 1001 |

Referring to the circuit provided, the states of a BCD decade counter are tabulated below.

| Clock pulse | $Q_3$ | $Q_2$ | $Q_1$ | $Q_0$ |
|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 |
| 1 | 0 | 0 | 0 | 1 |
| 2 | 0 | 0 | 1 | 0 |
| 3 | 0 | 0 | 1 | 1 |
| 4 | 0 | 1 | 0 | 0 |
| 5 | 0 | 1 | 0 | 1 |
| 6 | 0 | 1 | 1 | 0 |
| 7 | 0 | 1 | 1 | 1 |
| 8 | 1 | 0 | 0 | 0 |
| 9 | 1 | 0 | 0 | 1 |
| 0 Recycles) | 0 | 0 | 0 | 0 |

First, note that output $Q_0$ (flip-flop *FF0*) toggles on each clock pulse, so the logic equation for its $J_0$ and $K_0$ inputs is

$$J_0 = K_0 = 1$$

This equation is implemented by connecting $J_0$ and $K_0$ to a constant high level.

Next, note from the table above that output $Q_1$ (flip-flop *FF1*) toggles on the next pulse each time $Q_0 = 1$ and $Q_3 = 0$, so the logic equation for $J_1$ and $K_1$ inputs is

$$J_1 = K_1 = Q_0\overline{Q}_3$$

Next, note from the table above that output $Q_2$ (flip-flop *FF2*) toggles on the next pulse each time both $Q_0 = 1$ and $Q_1 = 1$. This requires an input logic equation such that

$$J_2 = K_2 = Q_0Q_1$$

This equation is implemented by ANDing $Q_0$ and $Q_1$ and connecting the gate output to the $J_2$ and $K_2$ inputs of *FF2*.

Finally, output $Q_3$ (flip-flop *FF3*) changes to the opposite state on the next clock pulse whenever $Q_0 = 1$, $Q_1 = 1$, and $Q_2 = 1$ (state 7), or when $Q_0 = 1$ and $Q_3 = 1$ (state 9). The equation to represent this behavior is

$$J_3 = K_3 = Q_0 Q_1 Q_2 + Q_0 Q_3$$

This function is implemented with the AND/OR logic connected to the $J_3$ and $K_3$ inputs of *FF3* as shown in the circuit diagram provided in the problem statement. Notice that here we have the main topological differences between a BCD decade counter and an ordinary modulus-16 binary counter; the arrangement of the $Q_0\overline{Q_3}$ AND gate, the $Q_0 Q_3$ AND gate, and the OR gate instructs the counter to cycle back to binary zero after achieving the 1001 state instead of proceeding to $(1010)_2$ like a 4-bit binary counter would.

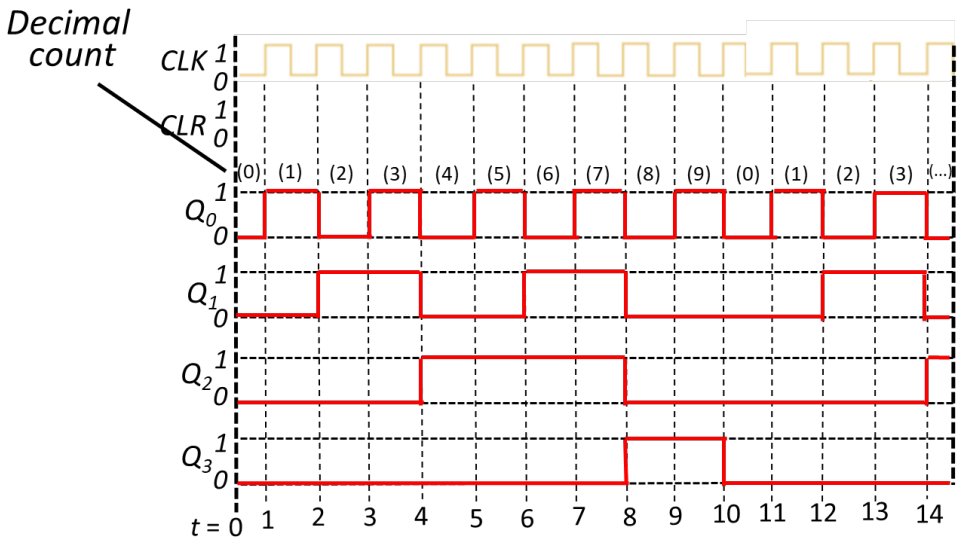
In summary, the BCD decade counter in question can be described by four logic equations:

$$\begin{cases} J_0 = K_0 = 1 \\ J_1 = K_1 = Q_0 \overline{Q}_3 \\ J_2 = K_2 = Q_0 Q_1 \\ J_3 = K_3 = Q_0 Q_1 Q_2 + Q_0 Q_3 \end{cases}$$

Equipped with these equations, we can tabulate the interaction between inputs $J_i$ and $K_i$ and the ensuing outputs $Q_i$; see below.

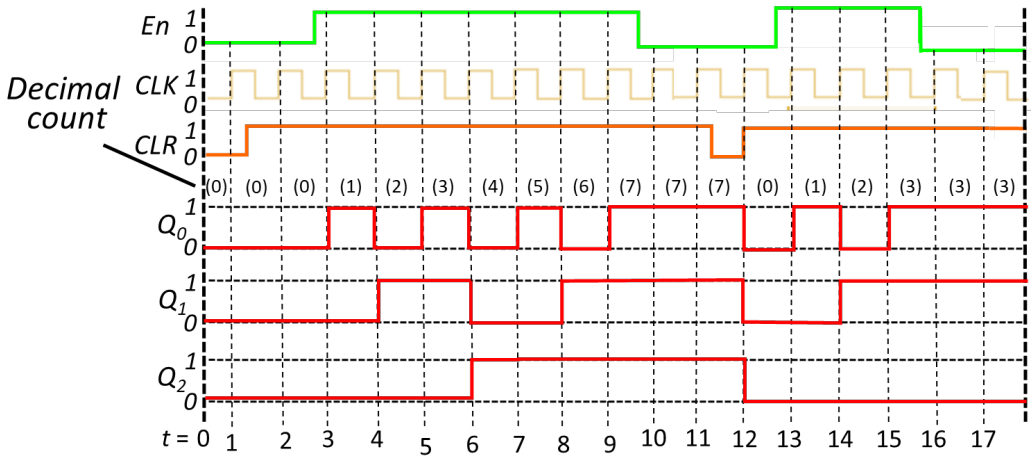
| CLK | $J_0K_0$ | $J_1K_1$ | $J_2K_2$ | $J_3K_3$ | $Q_0$ | $Q_1$ | $Q_2$ | $Q_3$ |
|---|---|---|---|---|---|---|---|---|
| 1 | 1 | 0 | 0 | 0 | 1 | 0 | 0 | 0 |
| 2 | 1 | 1 | 0 | 0 | 0 | 1 | 0 | 0 |
| 3 | 1 | 0 | 0 | 0 | 1 | 1 | 0 | 0 |
| 4 | 1 | 1 | 1 | 0 | 0 | 0 | 1 | 0 |
| 5 | 1 | 0 | 0 | 0 | 1 | 0 | 1 | 0 |
| 6 | 1 | 1 | 0 | 0 | 0 | 1 | 1 | 0 |
| 7 | 1 | 0 | 0 | 0 | 1 | 1 | 1 | 0 |
| 8 | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 1 |
| 9 | 1 | 0 | 0 | 0 | 1 | 0 | 0 | 1 |
| 10 | 1 | 0 | 0 | 1 | 0 | 0 | 0 | 0 |

**Problem 5.2:** Sketching the timing diagram for a BCD decade counter is a simple task, as illustrated below. From $t$ = 0 to $t$ = 9, the device progresses like a regular 4-bit binary counter; however, at $t$ = 10 the timer recycles from $(1001)_2$ (decimal 9) back to $(0000)_2$ (decimal 0) instead of progressing to $(1010)_2$, which represents decimal 10.




## P.6 ➡ *Solution*

The behavior of this binary counter depends on the state of the *enable* asynchronous input. If the enable signal is high, the counter behaves like a typical 3-bit binary counter, increasing its output word by 1 binary unit at the rising edge of each clock pulse. If the enable signal is low, the device stops increasing its count and holds the same output word until *En* goes high again or the clear input resets the count. With these rules in mind, sketching the three *Q* waveforms is quite straightforward; see below.

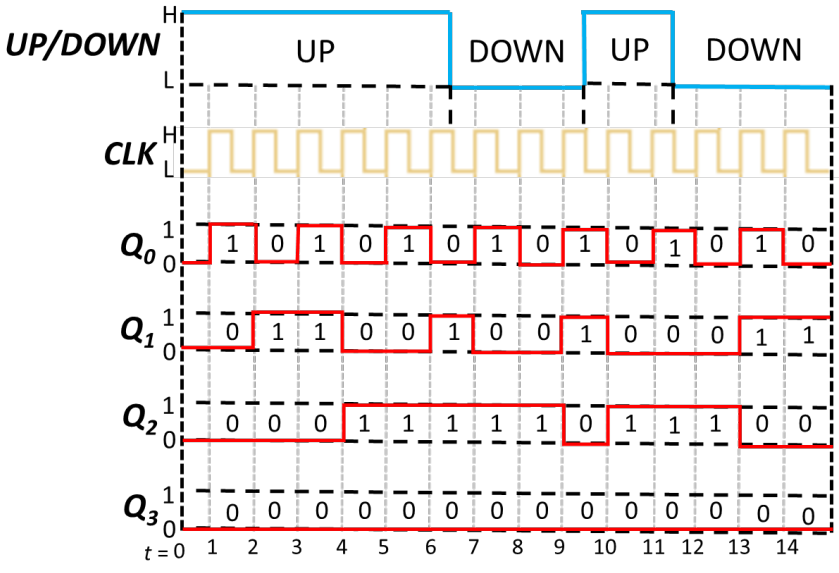



## P.7 ➡ *Solution*

As explained in the solution to Problem 6, a counter equipped with an *enable* input will behave as a typical counter as long as the *En* signal is high; if, however, *En* = 0, the output word will not be updated at the rising edge of the clock pulses. With these rules in mind, we can sketch the output waveforms; see below.


## P.8 ➡ *Solution*

Outlining the output waveform is quite straightforward. At $t = 1$, the UP/DOWN pulse is high and instructs the counter to increase by 1 binary unit, that is, from 0000 to 0001; accordingly, the $Q_0$ output pulse train goes high while the other three outputs remain unchanged. At $t = 2$, the UP/DOWN pulse remains high and the counter rises by another binary unit, i.e., from 0001 to 0010; to represent this change, $Q_0$ changes from 1 to 0 and $Q_1$ changes from 0 to 1, while $Q_2$ and $Q_3$ remain unaltered. Proceed similarly with the remaining clock pulses and you should obtain the following waveforms.



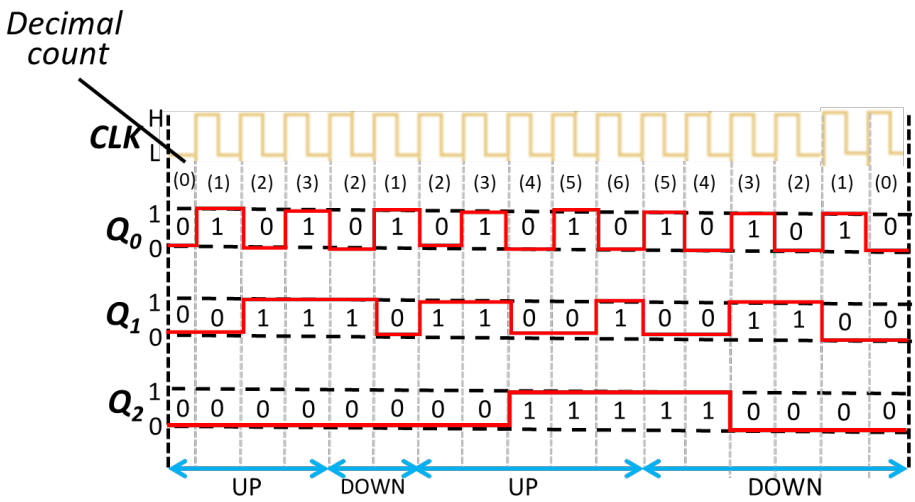

## P.9 ➡ *Solution*

For reference, we need the binary form of numbers 0 to 6:

| Decimal number | Binary form |
|---|---|
| 0 | 000 |

| | |
|---|---|
| 1 | 001 |
| 2 | 010 |
| 3 | 011 |
| 4 | 100 |
| 5 | 101 |
| 6 | 110 |

With these conversions in mind, we can outline the timing diagram. The output bits are shown in red, and the blue arrows indicate whether the counter is in UP or DOWN mode; for convenience, we also show the decimal count.




## P.10 ➡ *Solution*

**Problem 10.1:** The modulus of the cascaded system is given by the product of the individual counter moduli,

$$\overline{\text{MOD}} = 4 \times 8 \times 2 = \boxed{64}$$

Recall that the input frequency at the output of each cascaded stage is divided by a factor equal to the modulus of the individual counter. Thus, at (1), the frequency is

$$f_1 = \frac{1000\,\text{Hz}}{4} = \boxed{250\,\text{Hz}}$$

Likewise, at (2),

$$f_2 = \frac{250}{8} = \boxed{31.3\,\text{Hz}}$$

Then, at (3),

$$f_3 = \frac{31.3}{2} = \boxed{15.7\,\text{Hz}}$$

**Problem 10.2:** The modulus of the cascaded system is

$$\overline{\text{MOD}} = 12 \times 2 \times 3 \times 3 \times 4 = \boxed{864}$$

The frequency at (1) is

$$f_1 = \frac{800{,}000}{12} = \boxed{66.7\,\text{kHz}}$$

The frequency at (2) is

$$f_2 = \frac{66{,}700}{2} = \boxed{33.4\,\text{kHz}}$$

The frequency at (3) is

$$f_3 = \frac{33{,}400}{3} = \boxed{11.1\,\text{kHz}}$$

The frequency at (4) is

$$f_4 = \frac{11{,}100}{3} = \boxed{3.7\,\text{kHz}}$$

The frequency at (5) is

$$f_5 = \frac{3700}{4} = \boxed{925\,\text{Hz}}$$

**Problem 10.3:** The modulus of the cascaded system is

$$\overline{\text{MOD}} = 2 \times 10 \times 4 \times 11 \times 3 = \boxed{2640}$$

The frequency at (1) is

$$f_1 = \frac{75 \times 10^6}{2} = \boxed{37.5\,\text{MHz}}$$

The frequency at (2) is

$$f_2 = \frac{37.5 \times 10^6}{10} = \boxed{3.75\,\text{MHz}}$$

The frequency at (3) is

$$f_3 = \frac{3.75 \times 10^6}{4} = \boxed{938\,\text{kHz}}$$

The frequency at (4) is

$$f_4 = \frac{938{,}000}{11} = \boxed{85.3\,\text{kHz}}$$

The frequency at (5) is

$$f_5 = \frac{85{,}300}{5} = \boxed{17.1\,\text{kHz}}$$

## REFERENCES


- FLOYD, T.L. (2015). *Digital Fundamentals.* 11th edition. Upper Saddle River: Pearson.
- SEDHA, R.S. (2010). *A Textbook of Digital Electronics.* New Delhi: S. Chand and Company.
- TOKHEIM, R.L. (1994). *Schaum's Outline of Digital Principles*. New York: McGraw-Hill.