Counters and Decoders

Purpose

In this experiment, you will design and construct a 4-bit ripple-through decade counter with a decimal read-out display. Such a counter might be used for counting the number of $\alpha$-particles you observed in Experiment 8b.

Reading


Theory

Counters

Binary counters and binary numbers

A counter consists of a cascade of flip-flops connected so that the output of one flip-flop drives the input of the next. The signals to be counted are fed into the first flip-flop of the chain. The output, $Q$, of any given flip-flop represents a binary digit or bit (value 0 or 1). The complete set of outputs ($Q_2$, $Q_1$, $Q_0$) gives the total number of pulses in binary arithmetic, hence the name Binary Counter. See Fig. 10.1 (a).

The counters that we shall consider use JK flip-flops. Recall that the output of a JK changes only at the time of the negative edge of a pulse at the clock input. The new state into which it enters is determined by the values of J and K. In the simplest counter, every flip-flop is connected so that $J = K = 1$. For these values, the truth table for the JK says that:

$$Q_{n+1} = \overline{Q_n}$$

Thus the new state, $Q_{n+1}$, triggered by the arrival of the $n+1$th clock pulse is always the complement i.e., the logical inverse, $\overline{Q_n}$, of the old state, $Q_n$, which was generated by the previous pulse, etc. This mode of operation is called Toggling.

There are two main classes of counters: Asynchronous (also called Ripple-through, the kind you will build) and Synchronous counters. They are distinguished by the precision of timing of the output relative to the clock input. Examples of 3-bit counters of both types are shown in Fig. 10.1.

The Ripple-through counter

This counter is the type that you will build, since it has the fewest components.

The input pulse train drives the clock input, $C_0$, of the first JK. The output, $Q_0$, of the first JK drives the clock input, $C_1$, of the second, causing the second to toggle at each negative going edge of $Q_0$. The process is repeated as you go down the chain. For each input pulse, the flip-flops toggle sequentially. The disadvantage of this type is that the 25 nsec. delays between the input and output of successive flip-flops are cumulative; the outputs get increasingly out of step with the input pulse as you go down the chain. This problem can become serious at high counting rates, say 10 MHz, or
in computers where precise timing is important. There will be no problem in your experiment if you stay below 1 MHz.

Fig. 10.1 Binary Counters. (a) Ripple-through. (b) Synchronous.

In the synchronous counter in Fig. 10.1 (b), the problem is corrected by feeding the clock pulses simultaneously into every flip-flop. Whether or not this causes a change of state is determined by the J and K inputs used as dynamic controls. In the first flip-flop, J = K = 1 as before. In subsequent chips, J = K, but the logical value is determined by an AND gate. Thus, \( J_2 = Q_1 \cdot Q_0 \), \( J_3 = Q_2 \cdot (Q_1 \cdot Q_0) \), and so on. The response of the \( j \)th flip-flop to the \((n+1)\)th clock pulse can be represented by:

\[
(Q_j)_{n+1} = (Q_j)_n \oplus [Q_0 \cdot Q_1 \cdot \ldots \cdot Q_{j-1}]_n
\]

The \( j \)th flip-flop changes at the instant of the next clock pulse (plus a single delay of 25 nsec), if and only if every previous flip-flop in the chain is already in logical state \( Q=1 \). If one or more of the previous flip-flops is in logical 0, the state of the \( j \)th flip-flop remains the same. The overall effect is the same as in the ripple counter, but the delays do not accumulate.

**The input gate and resetting the counter to zero**

During a typical experiment, to measure the rate of radioactive decays for example, one carries out the following sequence of operations:

1. Reset all flip flops to zero.
2. open input gate to start counting.
3. Count for a specified time.
4. Close input gate to stop counting.
5. Read total number of counts.
6. Repeat.

The first flip-flop controls the flow of information into the counter. It can be employed as a gate by placing $J_0$ and $K_0$ under separate control. During normal counting, $J_0 = K_0 = 1$.

To interrupt counting in order to read the total, we can set $J_0 = K_0 = 0$ for which the truth table says: No change with time.

In order to reset the whole counter to zero after reading, we then set CLR = 0 for every flip flop.

**Decimal equivalents of binary numbers**

The total number of pulses is given in binary form by the set of the flip-flop outputs ($...Q_2, Q_1, Q_0$). Each $Q$ represents a single bit with a value of 0 or 1. Looking at the waveforms in Fig. 10.1 (a), you see that $Q_0$ changes for every input pulse, $Q_1$ changes only for every 2nd pulse, and so on.

Thus for an m-bit counter, the decimal number of pulses is given by:

$$N = (Q_0 + 2Q_1 + 4Q_2 + ...) = \sum_{j=0}^{m} (2^j)Q_j$$

For each decimal number, N, there is a Boolean expression which represents the logical operations that you perform on the binary outputs to identify the decimal number. The truth table and Boolean expressions for some 3-bit numbers are shown in the diagram. In practice, you write a bar over the $Q$ when the corresponding bit is 0 and leave out the bar when it is 1. The Boolean expression for a given decimal number gives zero for all combinations of bits that do not represent the number and give 1 for the correct combination. Consider for example, the binary number 1001 representing the decimal number, 9. Replace each binary 0 by a $\overline{Q}$ so that

$$Q_3 \cdot \overline{Q}_2 \cdot \overline{Q}_1 \cdot Q_0 = 1 \cdot \overline{0} \cdot \overline{0} \cdot 1 = 1 \cdot 1 \cdot 1 \cdot 1 = 1.$$ 
You get a ‘true’ result only for the number 1001. The output is zero for every other bit combination.
Decade counters and the general Modulo-n counter

The decade counter resets to zero by every 10th pulse. The output, known as binary coded decimal (BCD), is the same as a 4-bit binary scale up to decimal number 9. Beyond 9, the counter repeats, starting again from 0. The reset at 10 is so fast that you do not see it with a LED. The more general case is a Modulo-n counter, which resets to zero for every n-th pulse. A decade counter is the same as a Modulo-10 counter.

We will illustrate how the conversion is done for a 3-bit Modulo-5 counter. You will work out the details for a 4-bit Decade counter.

Recall that during normal operation, the direct input of every flip-flop is held at CLR = 1. When CLR = 0, all outputs go to zero. The Modulo-5 counter is reset to zero by every 5th pulse using the CLR input. The 5th pulse is identified when the bit pattern satisfies the equation: $Q_2 \cdot Q_1 \cdot Q_0 = 1$. However, the $Q_1$ is redundant, and the 5th pulse is uniquely identified in this counter by the expression $Q_2 \cdot Q_0 = 1$. We set CLR = 0 by using a NAND gate to evaluate the expression.

![Fig. 10.2. Modulo-5 counter. (a) Circuit. (b) Truth table.](image)

Recall that during normal operation, the direct input of every flip-flop is held at CLR = 1. When CLR = 0, all outputs go to zero. The Modulo-5 counter is reset to zero by every 5th pulse using the CLR input. The 5th pulse is identified when the bit pattern satisfies the equation: $Q_2 \cdot Q_1 \cdot Q_0 = 1$. However, the $Q_1$ is redundant, and the 5th pulse is uniquely identified in this counter by the expression $Q_2 \cdot Q_0 = 1$. We set CLR = 0 by using a NAND gate to evaluate the expression.
Decoders

The circuit for the Modulo-5, 3-bit binary-to-decimal decoder is shown in Fig. 10.3. Since we have a 3-bit number, the decoding may be done in principle by a set of 3-input AND gates. Each gate identifies a particular decimal number in the range 0 to 4 according to the Boolean expressions in the truth table for 3-bit numbers. The decoder circuit on the left is shown only to illustrate the logic. It can be greatly simplified in the course of conversion to the required NAND and NOR gates. Here are some guidelines:

Modulo 5 counter with decimal decoder

<table>
<thead>
<tr>
<th>Decimal</th>
<th>Binary</th>
<th>Boolean</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0 0 0</td>
<td>$Q_2\bar{Q}_1\bar{Q}_0 = Q_2\bar{Q}_1\bar{Q}_0$</td>
</tr>
<tr>
<td>1</td>
<td>0 0 1</td>
<td>$Q_2\bar{Q}_1\bar{Q}_0 = Q_1Q_0\bar{Q}_1Q_0 = Q_1\bar{Q}_0$</td>
</tr>
<tr>
<td>2</td>
<td>0 1 0</td>
<td>$Q_2Q_1Q_0 = Q_1Q_0 = Q_1\bar{Q}_0$</td>
</tr>
<tr>
<td>3</td>
<td>0 1 1</td>
<td>$Q_2Q_1Q_0 = Q_1Q_0 = Q_1\bar{Q}_0$</td>
</tr>
<tr>
<td>4</td>
<td>1 0 0</td>
<td>$Q_2\bar{Q}_1Q_0 = Q_2\bar{Q}_1\bar{Q}_0$</td>
</tr>
</tbody>
</table>

notes: $D = \bar{Q}_1Q_0$
the block of $Q_2$'s is redundant

Simplified Circuit using 2-input NOR gates

a) Refer to Experiment 9, page 9.6 for evaluation of 4-element products of the ABCD.

b) Note that both the output, $Q$, and its complement are available from each flip-flop.
c) Look for redundant bits. In the present case, \( \overline{Q}_2 \) is redundant for decimals 1, 2, 3. Each of the combinations \( \overline{Q}_1 Q_0 (=1), Q_1 \overline{Q}_0 (= 2), Q_1 Q_0 (= 3) \) appears only once in the list of numbers that a modulo-5 counter can generate before reset to zero at the 5th input pulse. The two bit combinations therefore identify uniquely the decimals 1, 2, and 3.

d) Look for repeated combinations and evaluate them only once. \( \overline{Q} Q_0 \) is repeated for decimals 0 and 4. We evaluate it once and call it \( D \).

e) The total number of chips on the board is often reduced by avoiding INVERTS. Unused gates on your NAND or NOR chips can be converted to INVERTS as shown in Exp. #9.

Problems

The methods for solving these problems are illustrated in the notes by the design of a 3-digit modulo-5 counter with decimal readout.

1. Draw the circuit for a 4-bit ripple-through binary counter using JK flip-flops. Arrange that the input can be gated by separate control of \( J \) and \( K \) for the first flip-flop; also that the entire counter can be reset to zero. Write down the truth table for the first 12 pulses and draw the waveforms that you would expect for \( C_0, Q_0, Q_1, Q_2, \) and \( Q_3 \). Neglect the small delay introduced by each flip-flop. See Fig. 10.1.

2. Design the modifications you need to convert your binary counter to a decade counter. Write the truth table for the first 12 pulses and graph the expected waveforms. See Fig. 10.2.

3. Design a 4-bit BCD to decimal decoder whose inputs are provided by the outputs of your decade counter. Write down the truth table and the Boolean expression for each decimal output (0 thru 9). Simplify your circuit to use the smallest number of 2-input NOR and NAND gates. See Fig. 10.3 for a modulo-5 decoder.

The Experiment

1. Design the circuits needed for the experiment. See the problems for details. Write down the circuits, truth tables, and predicted waveforms in your lab book before the class. Also a detailed wiring diagram for your circuit board is very helpful. Good organization is especially important with so many chips.

2. Construct a 4-bit ripple-through binary counter using JK flip-flops. Check that it functions correctly when driven by 250 \( \mu \)sec. pulses from your 555 clock at a rate of 1 kHz. Observe the input pulses and the outputs \( Q_0, Q_1, Q_2, \) and \( Q_3 \) with the oscilloscope. A stable display can only be obtained by triggering the scope from \( Q_3 \). Why? Draw the waveforms carefully and verify the truth table from your drawing. Check that the input gate and the zero reset function correctly.

3. Modify your circuit to make a decade counter. Check that the outputs now satisfy the truth table for BCD numbers using the same method as in part 2.
4. Construct a 4-bit BCD to decimal decoder. Connect the inputs of your decoder to the appropriate flip-flop outputs of your counter. Display the ten decoder outputs (0 thru 9) on LED indicators. Verify that the system operates correctly by observing the LEDs when the input pulse rate is slow enough that you can see each one light up in turn.