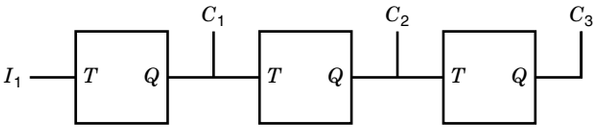


**Figure 1.** Simple binary counter. A three-bit counter will count from 0 to 7.

## COUNTING CIRCUITS

Counting circuits are found in a wide variety of electronic instrumentation and general digital computing systems. These circuits are used to count events, to accumulate sums, to hold pointers to memory instructions and data, and to perform other similar functions which require iterative computations. The term *counter* is most commonly used to refer to circuits that perform counting functions and, more specifically, to binary counters used in digital or hybrid digital/analog systems.

An example of the use of a counting circuit can be seen in a simple frequency meter. The input waveform is sampled over a fixed interval. A counter is used to add up the number of times that the signal increases above a certain threshold voltage. If the sample interval is 1 s and the waveform is sinusoidal, the counter contains the number of cycles per second or the frequency of the signal. In typical simple meters, this value is converted from its binary form stored in the counter circuit to a *binary coded decimal* form used to drive a set of light-emitting diode (LED) displays.

Digital counters are generally based on a series of binary storage elements called *flip-flops*. A flip-flop element stores either a binary 0 or 1. By logically connecting a set of flip-flops, it is possible to represent related binary digits and thus represent numbers.

An elementary binary counter can be constructed by connecting a set of $T$ flip-flops as shown in Fig. 1. Each $T$ flip-flop changes the state of its output $Q$ from 0 to 1 or vice versa every time the $T$ input changes from 1 to 0. The output waveform produced at each stage output $C_1$ through $C_3$ is shown in Fig. 2. Output $C_1$ represents the lowest-order binary digit (called a *bit*), while $C_3$ is the highest-order bit. By examining the waveform, it can be seen that before the first 1 to 0 transition on the input $I_1$ signal, the outputs represent the binary value 000. After the first 1 to 0 transition on input $I_1$, the outputs represent the value 001. The second 1 to 0 transition on $I_1$ causes a 010, or binary 2, to be output by the counter. Counting continues until the value 111 is reached. Once this maximum counter value of 7 is reached, the next 1 to 0 transition on input $I_1$ causes the outputs to change to 000 and the counter begins counting up again.

Each successive stage of the counter in Fig. 1 changes state, or counts, every second time the preceding flip-flop changes state. The frequency of the waveform produced at any output is half the frequency of the stage input. Each stage is said to be a divide-by-two counter, and the overall effect of the three-stage counter is to produce an output waveform that has the frequency of the input waveform divided by eight. An $N$-stage counter divides the frequency of the input waveform by $2^N$. The binary values actually stored in an $N$-bit counter actually range between 0 and $2^{N-1}$.

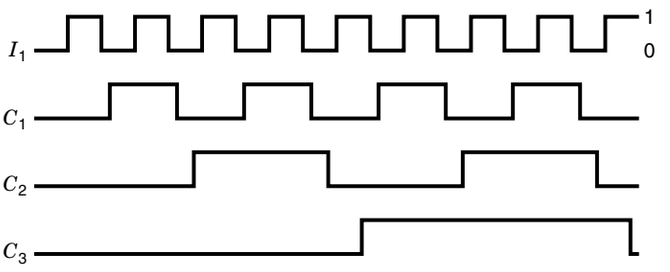


**Figure 2.** Binary counter output waveforms. The inherent delay in each flip-flop stage delays the transition of the next higher-order stage. In the worst case situation, the output of the highest-order stage will be delayed by an amount proportional to the number of stages.

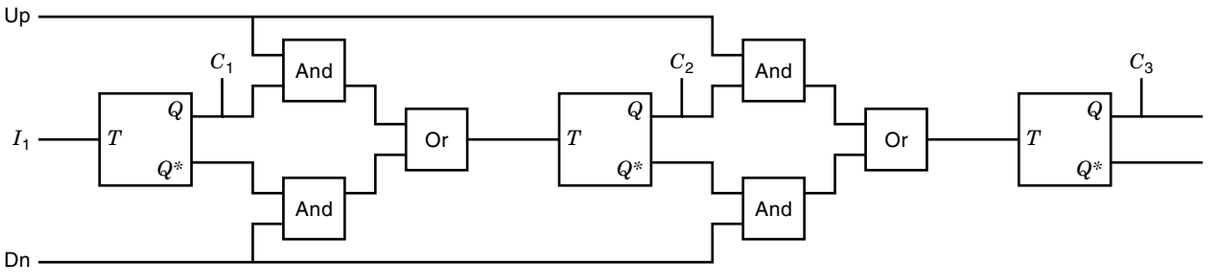


**Figure 3.** Three-bit up/down counter. Signals Up and Dn must remain constant during counting.

The specific interconnection of the feed-forward or feedback signals within a counter determine its counting characteristics. The counter shown in Fig. 3 is a 3-bit up/down counter. Each time the $I_1$ input line transitions from 1 to 0, the counter changes states. When the input UP is a 1, the counter counts up; when DN is a 1, the counter counts down. Only one of the two inputs UP or DN may be at logic 1 any particular time.

The counters shown in Figs. 1 and 3 are both said to be asynchronous counters since each of the flip-flops in the counters operates independent of a central clock. In fact, this type of counter is generally referred to as a *ripple counter,* indicating that the change of state of the overall counter actually transitions through the counter stage by stage rather than having all flip-flops change state simultaneously. This ripple effect can cause difficulties in systems that are meant to be operated synchronously. For example, as the 3-bit ripple counter transitions from state 111 to state 000, there is an inherent delay of signals passing from one stage to the next. The actual output signals transition from 111 to 110 to 100 and finally to 000. Although the intermediate values of 110 and 100 only last for a very short time, their appearance can cause problems. For example, if a logic circuit is monitoring the counter output for the state 110, a false signal will be incorrectly generated for a short duration when the counter transitions from 111 to 000.

For a ripple counter, the amount of delay involved in transitioning from one correct state to the next correct state is dependent upon the delay of the individual flip-flops. Assume that a $T$ flip-flop exhibits a one-unit delay before a change in the input signal effect the output signal. A 3-bit ripple counter presents its worst-case delay behavior when the state of the counter changes from 111 to 000 (or vice versa when counting down). This worst-case delay amounts to a three-unit delay. Although this may seem insignificant, when a more typical counter like a 16- or 32-bit counter is considered, the cumulative delay of 16 or 32 units can become very significant.

A mechanism to relieve this potential error condition is to design all of the counter flip-flops to be simultaneously triggered by the same clock. That is, the outputs of all flip-flops are directed to change at the same time under the control of a single clock signal. This type of counter is called a *synchronous counter.*

An example of a synchronous 3-bit binary counter is shown in Fig. 4. The $T$ flip-flops in this circuit change state if the $T$ input is 1 when the clock input $C$ changes from a 1 to a 0. The circuit counts input CLK pulses whenever the input COUNT is 1. All outputs of this counter change in synchronization with the clock signal CLK. The disadvantage of this type of counter is the additional logic gates that must be included in the circuit. For an $N$-bit counter, each subsequent stage $N$ requires an additional logic gate which must have $N - 1$ inputs.

The timing waveform for the synchronous counter of Fig. 4 is shown in Fig. 5. Notice that this waveform looks very similar to that shown in Fig. 2. The difference is the lack of ripple delay at each of the individual stage transitions.

Other variations on the basic counter scheme are found in a wide variety of applications. Counters that are capable of loading arbitrary starting values are typically used to keep track of the sequence of instructions being executed in a digital computer. The ability to load a new starting value allows the programmer to cause branching within a sequence of instructions. Some counters are configured to have arbitrary reset or terminal count values. This type of counter contains external logic that loads a new starting value once a certain termination count has been reached. Counters of this type are called modulo-$x$ counters, where $x$ is the value representing the terminal count. An example of this type of counter is one which counts from 000 to 101, returns back to 000, and starts counting again. The ability to reset a counter after a fixed count sequence is quite handy when nonprogrammable hardware is being used.

Counters are also frequently used to apply test vectors to a digital circuit. A counter is loaded with an initial count, and

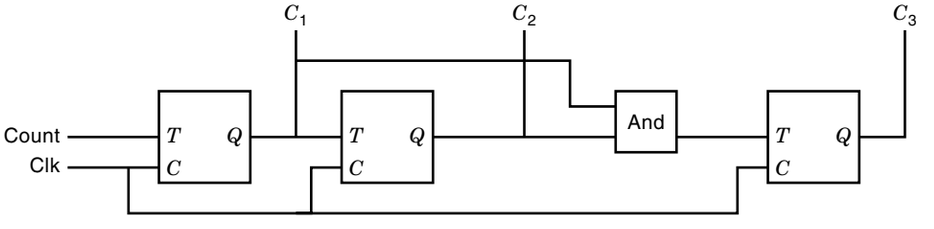


**Figure 4.** Synchronous binary counter. More logic gates are needed than in the ripple counter.

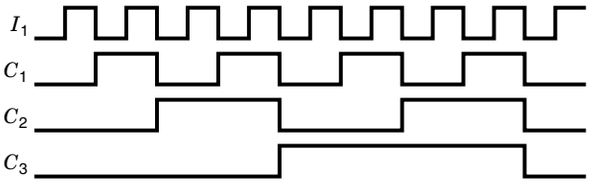


**Figure 5.** Synchronous 3-bit counter output waveforms. All outputs change together after a short flip-flop delay.

a prescribed count sequence applies counter outputs to the inputs to *combinational* circuits. The response of the combinational circuit is monitored to determine if errors exist in the behavior of the circuit. In this situation the counter is not part of the operational circuit but is rather an ancilliary device which only comes into play during the circuit testing procedure. This type of counter must be designed and built to be highly reliable since it becomes the key link in the testing chain of an operational circuit.

Implementation of a counter is no different from the implementation of any other logic circuit. Good very large scale integration (VLSI) design practices dictate that the clocked components (the flip-flops) be physically as close as possible in order to minimize timing anomalies.

## BIBLIOGRAPHY

W. I. Fletcher, *An Engineering Approach to Digital Design,* Englewood Cliffs, NJ: Prentice-Hall, 1980.

R. J. Feugate, Jr. and S. M. McIntyre, *Introduction to VLSI Testing,* Englewood Cliffs, NJ: Prentice-Hall, 1988.

R. L. Geiger, P. E. Allen, and N. R. Strader, *VLSI Design Techniques for Analog and Digital Circuits,* New York: McGraw-Hill, 1990.

F. J. Hill and G. R. Peterson, *Introduction to Switching Theory and Logical Design,* 3rd ed., New York: Wiley, 1981.

E. J. McCluskey, *Logic Design Principles with Emphasis on Testable Semicustom Circuits,* Englewood Cliffs, NJ: Prentice-Hall, 1986.

D. A. Pucknell, *Fundamentals of Digital Logic Design with VLSI Circuit Applications,* Englewood Cliffs, NJ: Prentice-Hall, 1990.

N. Weste and K. Eshraghian, *Principles of CMOS VLSI Design: A Systems Perspective,* Reading, MA: Addison-Wesley, 1985.

JOSEPH G. TRONT
Virginia Polytechnic Institute and
State University

**COUPLERS.** See DIRECTIONAL COUPLERS.

**COUPLING MEASURES.** See SOFTWARE QUALITY.

**COVERAGE, INSURANCE.** See INSURANCE.

**CPM.** See MINIMUM SHIFT KEYING.

**CRITICAL CURRENT, SUPERCONDUCTING.** See SUPERCONDUCTING CRITICAL CURRENT.