Ripple and Arbitrary Counters

In this lesson, you will learn about:

- Ripple Counters
- Counters with arbitrary count sequence

Design of ripple Counters

Two types of counters are identifiable:

- *Synchronous* counters, which have been discussed earlier, and
- *Ripple* counters.

In ripple counters, flip-flop output transitions serve as a source for triggering other flip-flops.

In other words, clock inputs of the flip-flops are triggered by output transitions of other flip-flops, rather than a common clock signal.

Typically, T flip-flops are used to build ripple counters since they are capable of complementing their content (See Figure 1).

The signal with the pulses to be counted, i.e."*Pulse*", is connected to the clock input of the flip-flop that holds the **LSB** (FF # 1).

The output of each FF is connected to the clock input of the next flip-flop in sequence.

The flip-flops are negative edge triggered (bubbled clock inputs).

T=1 for all FFs (J = K= 1). This means that each flip-flop complements its value if C input goes through a negative transition (1 → 0).

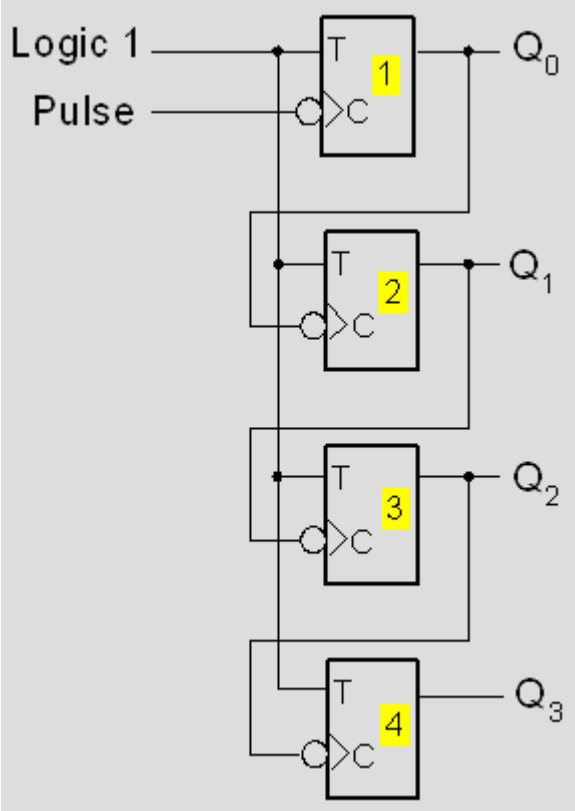

Figure 1: A ripple counter

The previous ripple up-counter can be converted into a down-counter in one of two ways:

- Replace the negative-edge triggered FFs by positive-edge triggered FFs, or
- Instead of connecting C input of FF Q_i to the output of the preceding FF (Q_{i-1}) connect it to the complement output of that FF (Q'_{i-1}).

Advantages of Ripple Counters:

- simple hardware and design.

Disadvantages of Ripple Counters:

- They are asynchronous circuits, and can be unreliable and delay dependent, if more logic is added.
- Large ripple counters are slow circuits due to the length of time required for the ripple to occur.

Counters with Arbitrary Count Sequence:

Design a counter that follows the count sequence: 0, 1, 2, 4, 5, 6. This counter can be designed with any flip-flop, but let's use the JK flip-flop.

Notice that we have two "*unused*" states (3 and 7), which have to be dealt with (see Figure 2). These will be marked by don't cares in the state table (Refer to the design of sequential circuits with unused states discussed earlier). The state diagram of this counter is shown in Figure 2.

In this figure, the unused states can go to any of the valid states, and the circuit can continue to count correctly. One possibility is to take state 7 (111) to 0 (000) and state 3 (011) to 4 (100).

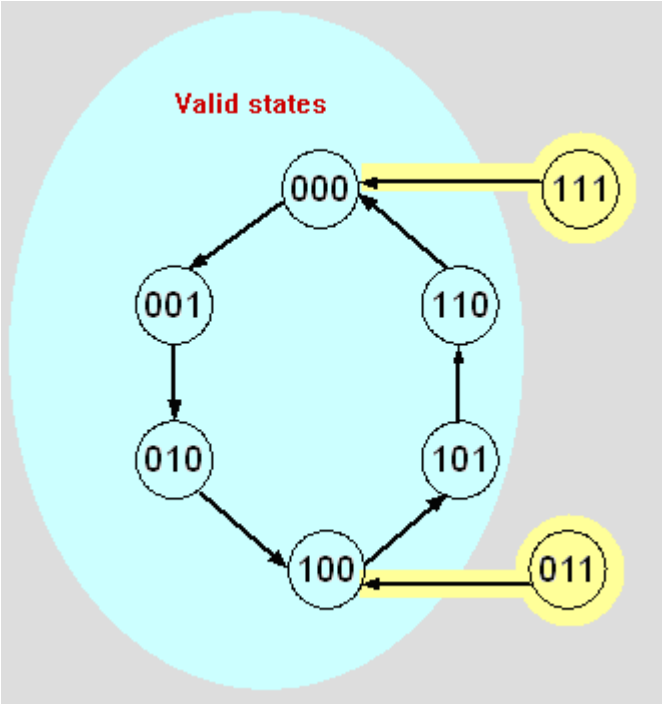

Figure 2: State diagram for arbitrary counting sequence

The design approach is similar to that of synchronous circuits. The state transition table is built as shown in Figure 3 and the equations for all J and K inputs are derived. Notice that we have used don't care for the unused state (although we could have used 100 as the next state for 011, and 000 as the next state of 111).

Unused states

Present State			Next State			Flip-Flop Inputs					
A	B	C	A	B	C	J_A	K_A	J_B	K_B	J_C	K_C
0	0	0	0	0	1	0	X	0	X	1	X
0	0	1	0	1	0	0	X	1	X	X	1
0	1	0	1	0	0	1	X	X	1	0	X
0	1	1	X	X	X	X	X	X	X	X	X
1	0	0	1	0	1	X	0	0	X	1	X
1	0	1	1	1	0	X	0	1	X	X	1
1	1	0	1	1	1	X	0	X	0	1	X
1	1	1	X	X	X	X	X	X	X	X	X

Figure 3: State table for arbitrary counting sequence

The computed J and K input equations are as follows:

$J_A = B$ $K_A = B$

$J_B = C$ $K_B = 1$

$J_C = B'$ $K_C = 1$

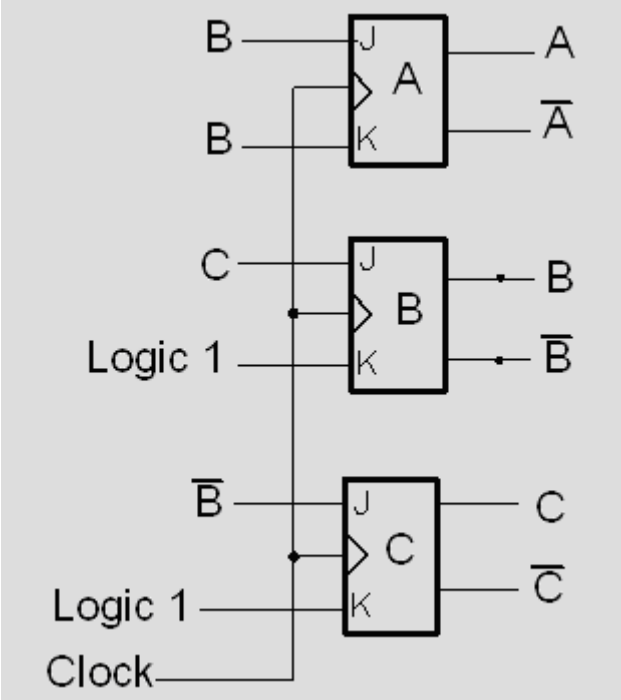

Figure 4: Circuit for arbitrary counting sequence