Counters

In this lesson, the operation and design of Synchronous Binary Counters will be studied.

Synchronous Binary Counters (SBC)

Description and Operation

In its simplest form, a *synchronous binary counter* (**SBC**) receives a train of clock *pulses* as input and outputs the *pulse count* (Q_{n-1} …. Q_2 Q_1 Q_0).

An example is a 3-bit counter that counts from 000 upto 111. Each counter consists of a number of FFs. (Figure 1)

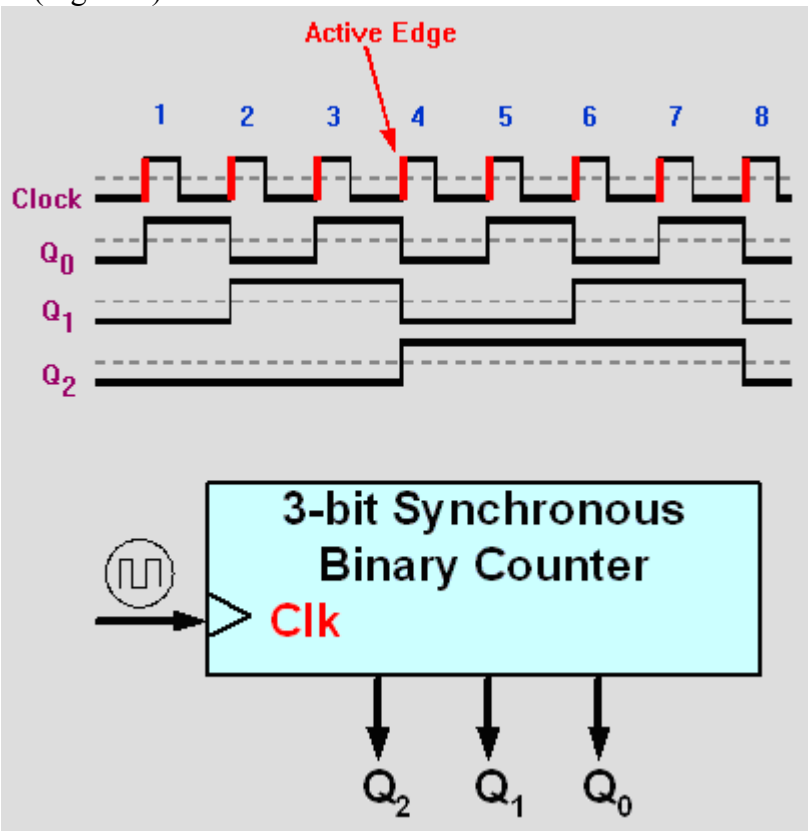

Figure 1: 3-bit SBC

In *synchronous* counters, all FFs are triggered by the same input clock.

An *n*-bit counter has *n*-FFs with 2^n distinct *states*, where each state corresponds to a particular *count*.

Accordingly, the possible *counts* of an *n*-bit counter are 0 to (2^n-1). Moreover an n-bit counter has *n* output bits (Q_{n-1} …. Q_2 Q_1 Q_0).

After reaching the maximum count of (2^n-1), the following clock pulse resets the count back to 0.

Thus, a 3-bit counter counts from 0 to 7 and back to 0. In other words, the output count actually equals (*Total # of input pulses* **Modulo** 2^n).

Accordingly, it is common to identify counters by the modulus 2^n. For example, a 4-bit counter provides a modulo 16 count, a 3-bit counter is a modulo 8 counter, etc.

Referring to the 3-bit counter mentioned earlier, each stage of the counter divides the frequency by 2, where the last stage divides the frequency by 2^n, *n* being the number of bits. (Figure 2)

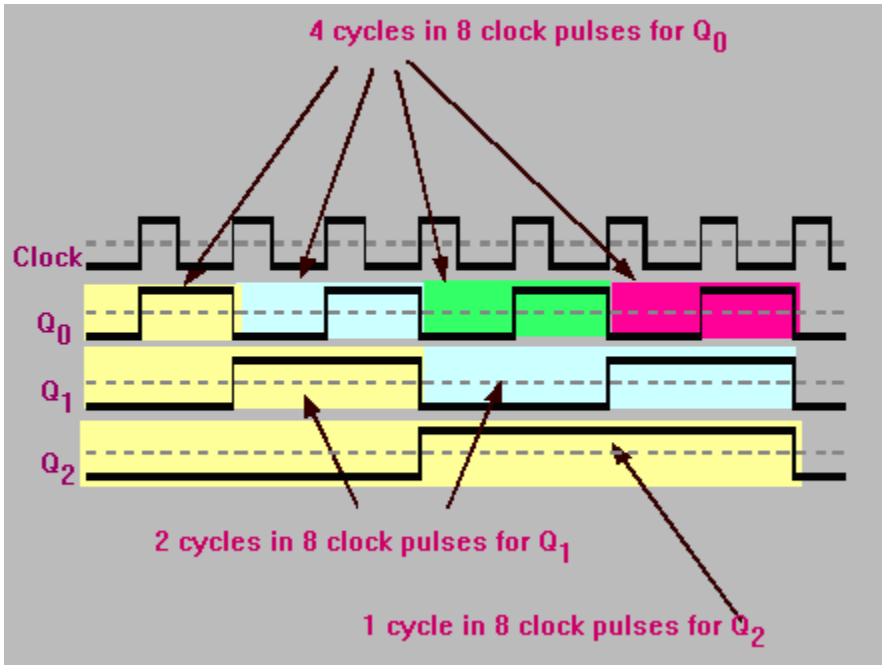

Figure 2: 3-bit SBC

Thus, if the frequency (i.e. no. of cycles/ sec) of clock is F, then the frequency of output waveform of Q_0 is F/2, Q_1 is F/4, and so on. In general, for n-bit counter, we have $F/2^n$.

Design of Binary Counters (SBC)

Design procedure is the same as for other synchronous circuits.

A counter may operate without an external input (except for the clock pulses!)

In this case, the output of the counter is taken from the outputs of the flip-flops without any additional outputs from gates.
Thus, there are no columns for the input and outputs in the state table; we only see the current state and next state…

Example Design a 4-bit SBC using JK flip-flops.

The counter has 4 FFs with a total of 16 states, (0000 to 1111) → 4 state variables Q3 Q2 Q1 Q0 are required.

Present State				Next State				Flip-Flop Inputs							
Q_3	Q_2	Q_1	Q_0	Q_3	Q_2	Q_1	Q_0	J_{Q3}	K_{Q3}	J_{Q2}	K_{Q2}	J_{Q1}	K_{Q1}	J_{Q0}	K_{Q0}
0	0	0	0	0	0	0	1	0	X	0	X	0	X	1	X
0	0	0	1	0	0	1	0	0	X	0	X	1	X	X	1
0	0	1	0	0	0	1	1	0	X	0	X	X	0	1	X
0	0	1	1	0	1	0	0	0	X	1	X	X	1	X	1
0	1	0	0	0	1	0	1	0	X	X	0	0	X	1	X
0	1	0	1	0	1	1	0	0	X	X	0	1	X	X	1
0	1	1	0	0	1	1	1	0	X	X	0	X	0	1	X
0	1	1	1	1	0	0	0	1	X	X	1	X	1	X	1
1	0	0	0	1	0	0	1	X	0	0	X	0	X	1	X
1	0	0	1	1	0	1	0	X	0	0	X	1	X	X	1
1	0	1	0	1	0	1	1	X	0	0	X	X	0	1	X
1	0	1	1	1	1	0	0	X	0	1	X	X	1	X	1
1	1	0	0	1	1	0	1	X	0	X	0	0	X	1	X
1	1	0	1	1	1	1	0	X	0	X	0	1	X	X	1
1	1	1	0	1	1	1	1	X	0	X	0	X	0	1	X
1	1	1	1	0	0	0	0	X	1	X	1	X	1	X	1

Figure 3: State table for the example

Notice that the next state equals the present state plus one.

To design this circuit, we derive the flip-flop input equations from the state transition table. Recall that to find J & K values, we have to use:

- The present state,
- The next state, and
- The JK flip-flop excitation table.

When the *count* reaches 1111, it resets back to 0000, and the count cycle is repeated.

Once the J and K values are obtained, the next step is to find out the simplified input equations by using K-maps, as shown in figure 4.

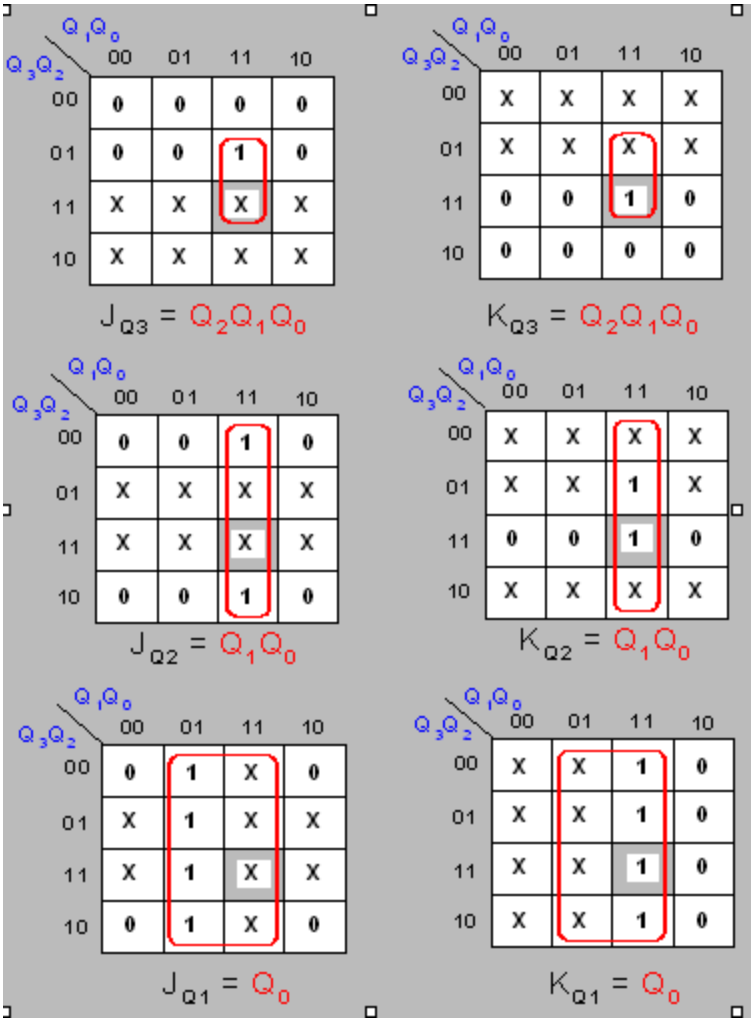

Figure 4: K-maps for the example

Notice that the maps for J_{Q0} and K_{Q0} are not drawn because the values in the table for these two variables either contain 1's or X's. This will result in $J_{Q0} = K_{Q0} = 1$

Note that the Boolean equation for J input is the same as that of the K input for all the FFs $\Rightarrow$ Can use T-FFs instead of JK-FFs.

Count Enable Control

In many applications, controlling the counting operation is necessary $\Rightarrow$ a count-enable (E_n) is required.

If E_n= 1 **then** counting of incoming clock pulses is enabled **Else if** (E_n =0), no incoming clock pulse is counted.

To accommodate the enable control, two approaches are possible.

1. Controlling the clock input of the counter
2. Controlling FF excitation inputs (JK, T, D, etc.).

Clock Control

Here, instead of applying the system clock to the counter directly, the clock is first ANDed with the E_n signal.

Even though this approach is simple, it is not recommended to use particularly with configurable logic, e.g. FPGA's.

FF Input Control (Figure 5)

In this case, the E_n =0 causes the FF inputs to assume the no change value (SR=00, JK=00, T=0, or D_i=Q_i).

To include E_n, analyze the stage when $J_{Q1} = K_{Q1} = Q_0$, and then include E_n. Accordingly, the FF input equations of the previous 4-bit counter example will be modified as follows:

$J_{Q0} = K_{Q0} = 1.\ E_N = E_n$

$J_{Q1} = K_{Q1} = Q_0.\ E_n$

$J_{Q2} = K_{Q2} = Q_1.Q_0.\ E_n$

$J_{Q3} = K_{Q3} = Q_2.Q_1.Q_0.\ E_n$

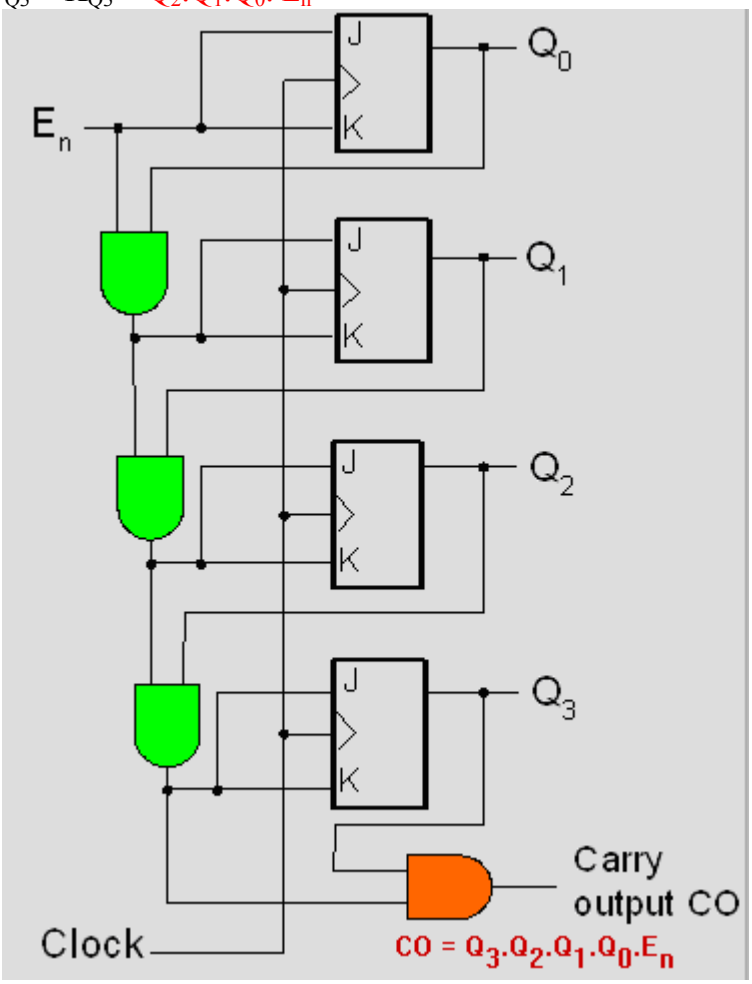

Figure 5: FF input control in counter

Thus, when E_n = 0, all J and K inputs are equal to zero, and the flip flops remain in the same state, even in the presence of clock pulses

When E_n = 1, the input equations are the same as equations of the previous example.

A carry output signal (CO) is generated when the counting cycle is complete, as seen in the timing diagram.

The CO can be used to allow cascading of two counters while using the same clock for both counters. In that case, the CO from the first counter becomes the E_n for the second counter. For example, two modulo-16 counters can be cascaded to form a modulo-256 counter.

Up-Down Binary Counters

In addition to counting up, a SBC can be made to count down as well.

A control input, S is required to control the direction of count.

IF S= 1, the counter counts up, otherwise it counts down.

FF Input Control

Design a Modulo-8 up-down counter with control input S, such that if S= 1, the counter counts up, otherwise it counts down. Show how to provide a count enable input and a carry-out (CO) output. (See figures 6 & 7)

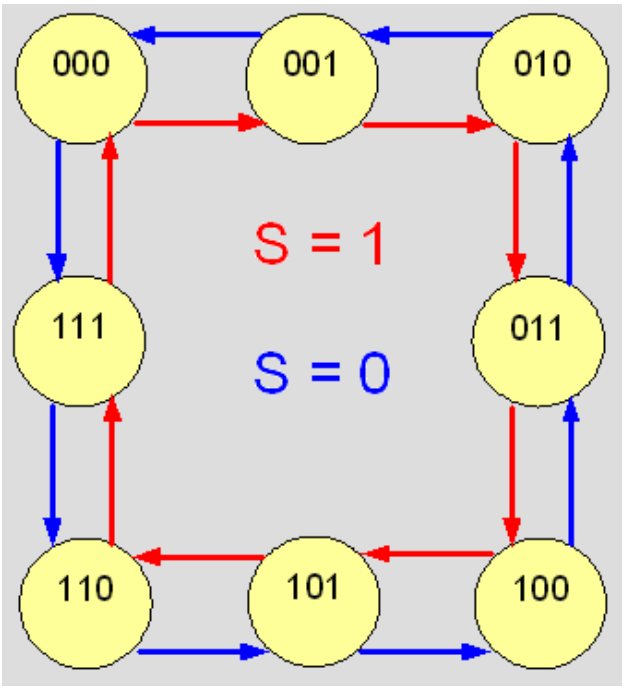

Figure 6: State diagram for FF input control example

Present State				Next State			Flip-Flop Inputs		
Q_2	Q_1	Q_0	S	Q_2	Q_1	Q_0	T_2	T_1	T_0
0	0	0	0	1	1	1	1	1	1
0	0	0	1	0	0	1	0	0	1
0	0	1	0	0	0	0	0	0	1
0	0	1	1	0	1	0	0	1	1
0	1	0	0	0	0	1	0	1	1
0	1	0	1	0	1	1	0	0	1
0	1	1	0	0	1	0	0	0	1
0	1	1	1	1	0	0	1	1	1
1	0	0	0	0	1	1	1	1	1
1	0	0	1	1	0	1	0	0	1
1	0	1	0	1	0	0	0	0	1
1	0	1	1	1	1	0	0	1	1
1	1	0	0	1	0	1	0	1	1
1	1	0	1	1	1	1	0	0	1
1	1	1	0	1	1	0	0	0	1
1	1	1	1	0	0	0	1	1	1

Figure 7: State table for FF input control example

The equations are (see figure 8)

$T_0 = 1$

$T_1 = Q_0 . S + Q'_0 . S'$

$T_2 = Q_1.Q_0 . S + Q'_1 . Q'_0 . S'$

The carry outputs for the next stage are: (see figure 8)

$C_{up} = Q_2.Q_1.Q_0$ for upward counting.

$C_{down} = Q'_2.Q'_1.Q'_0$ for downward counting.

The equations with E_n are (see figure 9)

$T_0 = E_n . 1$

$T_1 = Q_0 . S . E_n + Q'_0 . S' . E_n$

$T_2 = Q_1.Q_0 . S . E_n + Q'_1 . Q'_0 . S' . E_n$

The carry outputs for the next stage, with E_n are (see figure 9):

$C_{up} = Q_2.Q_1.Q_0 . E_n$ for counting up.

$C_{down} = Q'_2.Q'_1.Q'_0 . E_n$ for counting down.

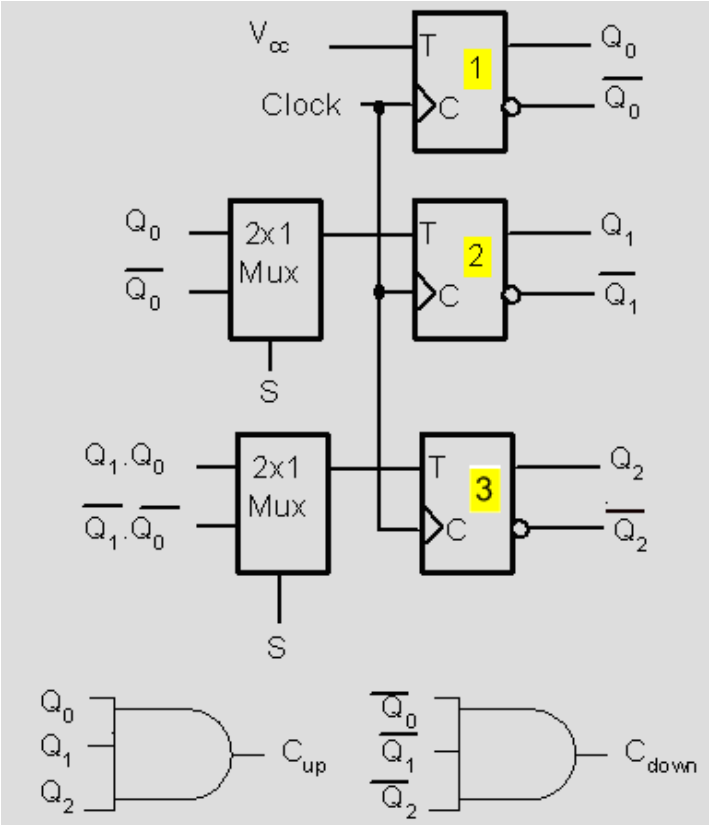

Figure 8: Circuit of up-down counter

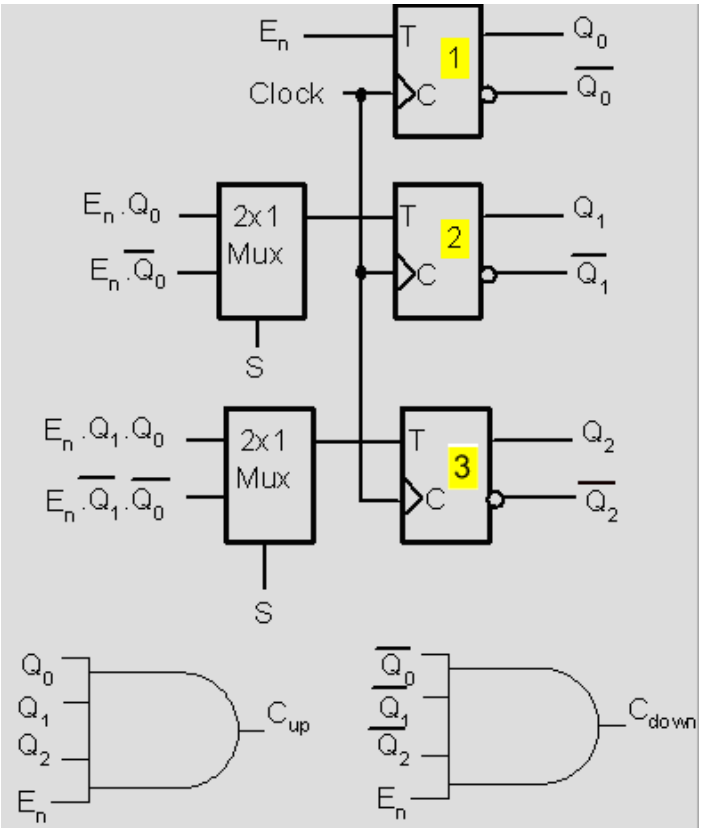

Figure 9: Circuit of up-down counter with En