More on Counters

In this lesson, you will learn

- some important counter control inputs:
  - Parallel Load (Ld)
  - Synchronous Clear
  - Asynchronous Clear
- use of available counters to build counters of different count

Counter Control

We have seen how to include a count-enable control input to enable/disable counting in the counter.

Now we show how to include important counter control inputs; namely:

- Parallel Load (Ld)
- Clear (Synchronous/Asynchronous)

The block diagram of a 4-bit counter with the above capabilities is shown in Figure 1.

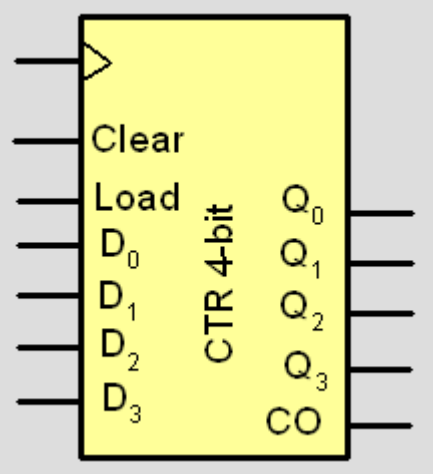

Figure 1: A 4-bit counter

Let us now discuss the design of the counter. We will start with a typical stage of basic counter, and will add the control signals to this stage in a step-wise approach. A positive edge-triggered counter will be assumed.

Figure 2 shows a stage of the basic counter, where we see that the J and K inputs of flip-flop at stage 1 are connected to an AND gate with Q_0 and Count as inputs.

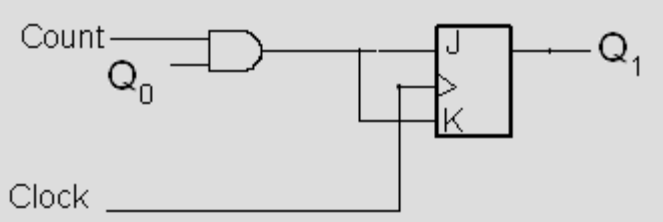

Figure 2: a stage of basic counter

Now let's add the Load control input to this stage. Thus the operation would become as shown in Table 1.

Count	Ld	Clk	Operation	Mode
0	0	x	$Q_i \leftarrow Q_i$ $\forall i = 0, n-1$	Count disable
x	1	↑	$Q_i \leftarrow D_i$ $\forall i = 0, n-1$	Parallel Load
1	0	↑	$Q_i \leftarrow Q_i + 1$ $\forall i = 0, n-1$	Count enable

Table 1: Operation of counter with Load signal added

Based on the above table, the stage will be modified as shown in Figure 3.

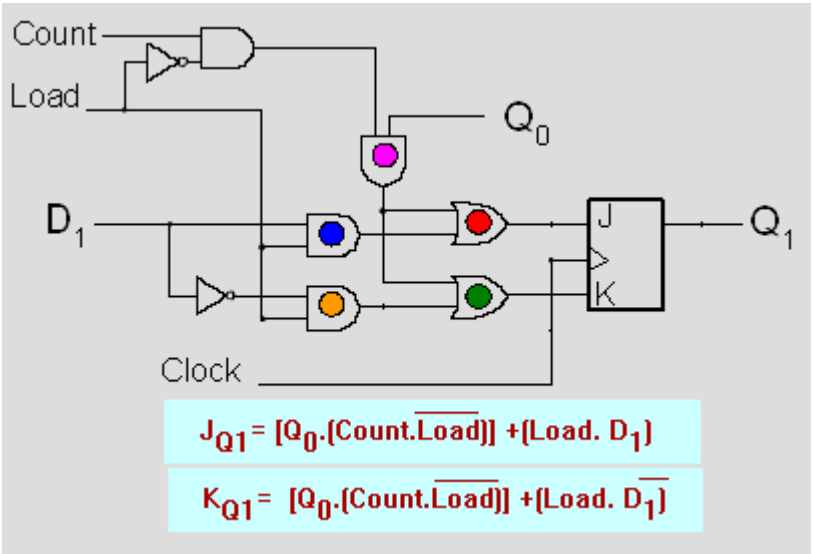

Figure 3: Modified stage of a counter

In this figure, when Count = 0 and Load = 0, J and K inputs in the flip flop will be equal to 0 and 0 ⇒ NO CHANGE state is achieved. Notice that AND gate marked with pink will also be generating an output of 0.

When Count = 0 or 1 (i.e. don't care) and Load = 1, parallel load operation will take place. The blue AND gate will propagate D while orange will propagate D' to J and K inputs respectively.

When Count = 1 and Load = 0, the circuit will operate as counter because JK =11. It is the same behavior with respect to the tradition counter.

Now, the control signal Clear can be added to the stage. We will assume *asynchronous* Clear in the design.

Count	Ld	Clr	Clk	Operation	Mode
x	x	1	x	$Q_i \leftarrow 0$ $\forall i = 0, n\text{-}1$	Asynchronous Clear
x	1	0	↑	$Q_i \leftarrow D_i$ $\forall i = 0, n\text{-}1$	Parallel Load
1	0	0	↑	$Q_i \leftarrow Q_i + 1$ $\forall i = 0, n\text{-}1$	Count enable
0	0	0	x	$Q_i \leftarrow Q_i$ $\forall i = 0, n\text{-}1$	Count disable

Table 2: Counter Operation with Count, Load, and Clear signals

The stage will be modified as given in Figure 4.

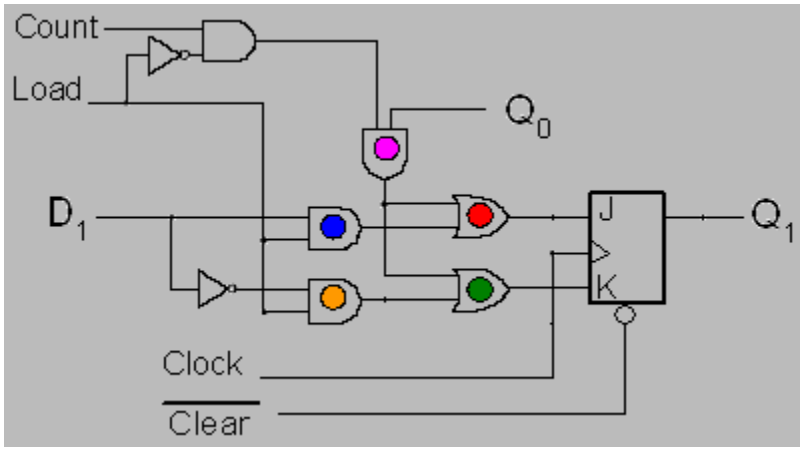

Figure 4: Addition of Clear signal to a counter stage

Designing counters with available counters

Binary counters with parallel load can be used to design different modulo-*n* counters. For example, the 4-bit parallel load counter discussed in this lesson can be used to design any counter of modulo *n* where $2 \leq n \leq 16$.

Design of decade counter

The binary counter with parallel load can be converted into a synchronous DECADE counter (without load input) by connecting an external AND gate to it, as shown in Figure 5.

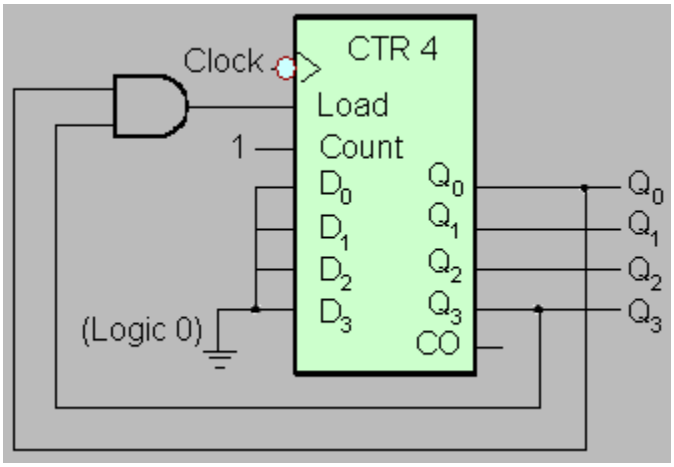

Figure 5: Decade counter

Connect all the D inputs to Ground (Logic 0). Make Count = 1. This will make the circuit always operating in the counter mode.

The 2-input AND gate connected to the Load input of the counter takes Q_0 and Q_3 as its inputs. As long as the Load input (connected to the output of the AND gate) is 0, the counter is incremented by one with each clock pulse.

When the output count reaches 9 (1001), the output of AND gate will equal 1. This puts the counter in the *load mode* (Load =1). Thus, *the next clock pulse* will load the data on the D-inputs (0000) into the counter instead of incrementing the count.

Thus the counter counts from 0000 (decimal 0) to 1001 (decimal 9) then goes back to 0000 and so on. → *Modulo 10 counting*

Design of a counter that counts from 3 to 12.

The counter discussed above can be made to count from 0011 (decimal 3) to 1100 (decimal 12). Only small modifications are required, which are (see Figure 6):

- Connect the D inputs to 0011 (i.e. $D_3D_2D_1D_0 = 0011$). This will make the circuit start counting from 3 whenever 12 has been counted.
- Connect the AND gate to Q_3 and Q_2. Thus, whenever the circuit reaches 12 (1100), $Q_3 = Q_2 = 1$, which will make the output of the AND gate equal to 1, making Load active, *so in the next clock transition* the counter does not count, but is loaded from its four inputs, with a value of 0011.

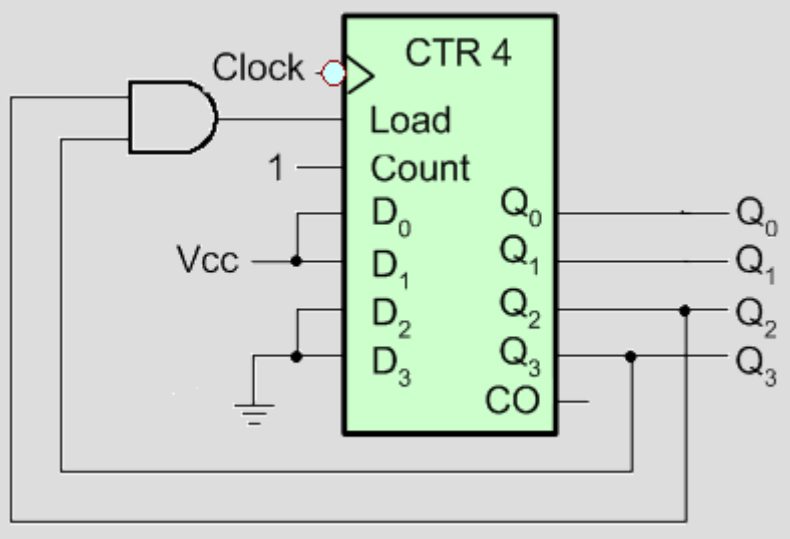

Figure 6: Counter for 3 to 12 counting

Thus the counter counts from 0011 (decimal 3) to 1100 (decimal 12), and back to 0011. This is also a mod-10 counter, since it also counts ten numbers.

Some counters may have the "Clear" control input. With this capability, the counter can be "cleared" at any time. The "Clear" signal can also be classified into two types: Synchronous and Asynchronous.

The Synchronous Clear case

The Synchronous Clear input is activated *in synchronization* with the clock pulse.

To explain this behavior, consider a **MOD-6** counter that counts from 000 (decimal 0) to 101 (decimal 5). The circuit is shown in the figure.

In this circuit, once the count of 5 (101) is detected by the AND gate, the counter is cleared on the next clock pulse. Thus, the counter counts from 0 to 5 back to 0.

Assuming negative edge-triggered FFs, the timing diagram of this counter is shown in Figure 7. Notice the delayed transitions of the counter outputs (Q's) after the negative clock edge due to gate propagation delays.

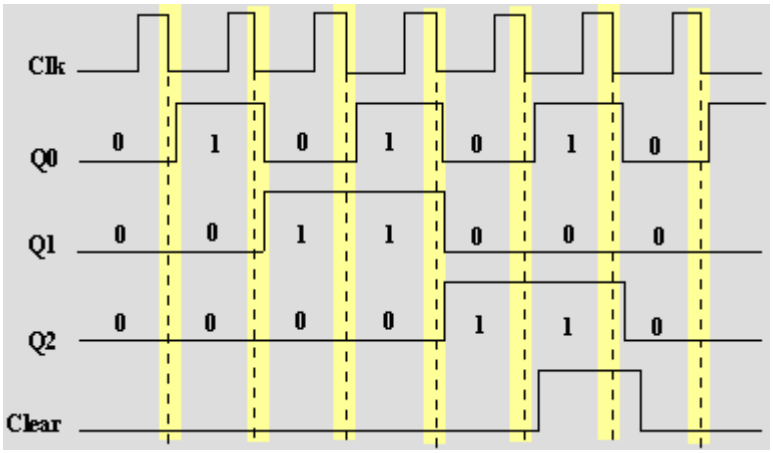

Figure 7: Timing diagram – Synchronous Clear

The timing diagram clarifies the case of $Q_2Q_1Q_0 = 101$ where the Clear input becomes 1 causing the counter to clear on the next negative clock edge. The point to notice here is that the effect of change in "Clear" is not immediately applied, but becomes effective in the following clock pulse, because the "Clear" input is "Synchronous", i.e. *it only takes effect at the next active clock edge*.

The Asynchronous Clear case

If we use asynchronous clear rather than synchronous clear, as soon as the count $Q_2Q_1Q_0$ reaches 101, "Clear"is activated and the FFs are cleared immediately without waiting for the next active clock edge.

This causes the count $Q_2Q_1Q_0 = 101$ to switch to 000 after a small delay. In other words the count 101 does not last for a full clock period as other counts, but rather will appear for a very short duration as a narrow pulse (glitch) as shown in Figure 8. Thus, it would appear that the counter counts from 0 to 4, that is, from 000 to 100.

This happens because "Clear" is "Asynchronous". It does not wait for the clock pulse to come, and does the "clearing" operation immediately.

As a result, the output values become $Q_2Q_1Q_0 = 101$ for a very short duration of time, almost negligible, and then the contents become $Q_2Q_1Q_0 = 000$ within the same clock period.

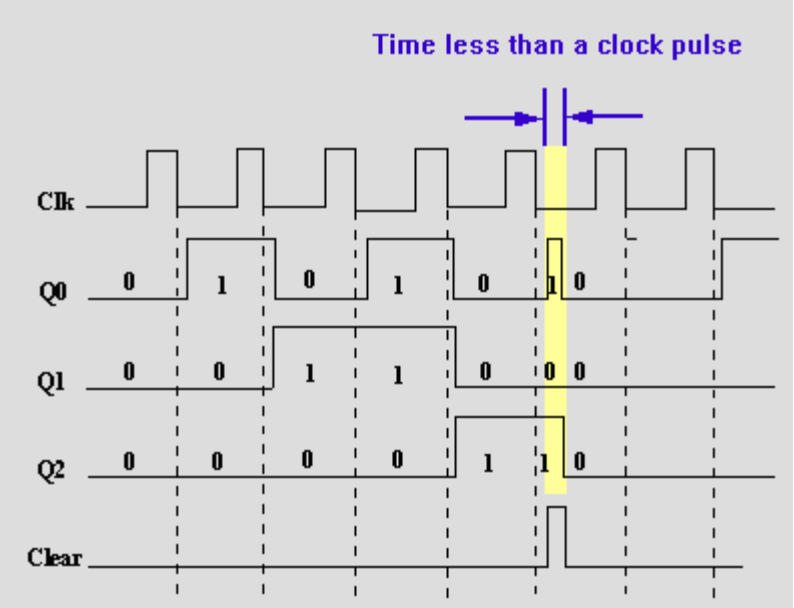

Figure 8: Timing diagram – Asynchronous Clear

To have a MOD6 counter designed using the asynchronous clear, we should detect a count of 6 (instead of 5) and use that to clear the counter asynchronously.
In this case, once the count reaches $Q_2Q_1Q_0 = 110$, "Clear" is activated, you will not be able to observe $Q_2Q_1Q_0 = 110$, because it will be for very short duration. This is shown in Figure 9.

It would seem to you that after $Q_2Q_1Q_0 = 101$, the next state is $Q_2Q_1Q_0 = 000$.

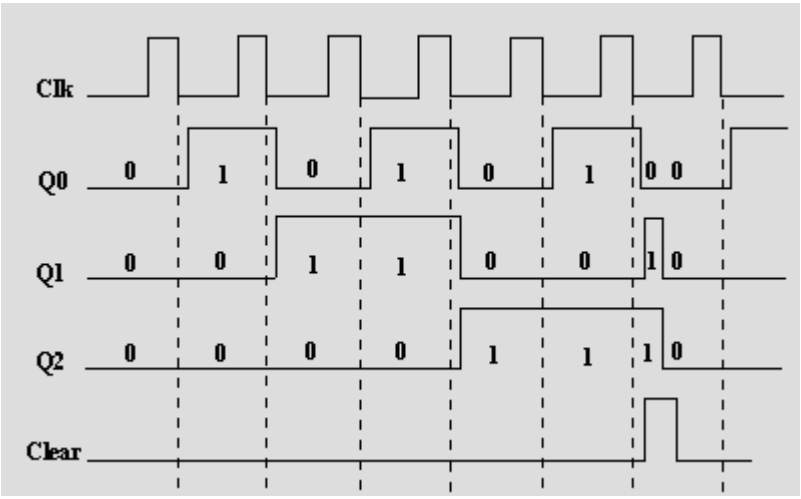

Figure 9: Timing diagram