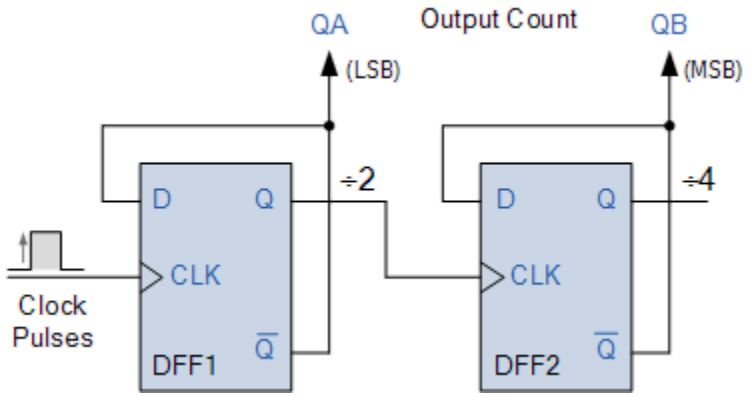


MOD Counters

MOD Counters are cascaded counter circuits which count to a set modulus value before resetting

The job of a counter is to count by advancing the contents of the counter by one count with each clock pulse. Counters which advance their sequence of numbers or states when activated by a clock input are said to operate in a “count-up” mode. Likewise, counters which decrease their sequence of numbers or states when activated by a clock input are said to operate in a “count-down” mode. Counters that operate in both the UP and DOWN modes, are called bidirectional counters.

Counters are sequential logic devices that are activated or triggered by an external timing pulse or clock signal. A counter can be constructed to operate as a synchronous circuit or as an asynchronous circuit. With synchronous counters, all the data bits change synchronously with the application of a clock signal. Whereas an asynchronous counter circuit is independent of the input clock so the data bits change state at different times one after the other.

Then counters are sequential logic devices that follow a predetermined sequence of counting states which are triggered by an external clock (CLK) signal. The number of states or counting sequences through which a particular counter advances before returning once again back to its original first state is called the **modulus** (MOD). In other words, the modulus (or just modulo) is the number of states the counter counts and is the dividing number of the counter.

Modulus Counters, or simply *MOD counters*, are defined based on the number of states that the counter will sequence through before returning back to its original value. For example, a 2-bit counter that counts from 00_2 to 11_2 in binary, that is 0 to 3 in decimal, has a modulus value of 4 (00 → 1 → 10 → 11, and return back to 00) so would therefore be called a modulo-4, or mod-4, counter. Note also that it has taken four clock pulses to get from 00 to 11.

As in this simple example there are only two bits, (n = 2) then the maximum number of possible output states (maximum modulus) for the counter is: $2^n = 2^2$ or 4. However, counters can be designed to count to any number of 2^n states in their sequence by cascading together multiple counting stages to produce a single modulus or MOD-N counter.

Therefore, a "Mod-N" counter will require "N" number of flip-flops connected together to count a single data bit while providing 2^n different output states, (n is the number of bits). Note that N is always a whole integer value.

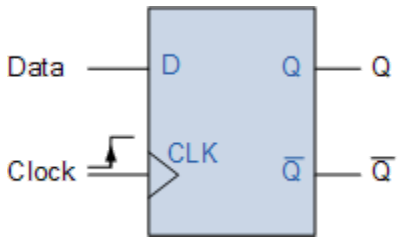
The we can see that MOD counters have a modulus value that is an integral power of 2, that is, 2, 4, 8, 16 and so on to produce an n-bit counter depending on the number of flip-flops used, and how they are connected, determining the type and modulus of the counter.

D-type Flip-flops

MOD counters are made using "flip-flops" and a single flip-flop can produce a count of 0 or 1, giving a maximum count of 2. There are different types of flip-flop designs we could use, the S-R, the J-K, J-K Master-slave, the D-type or even the T-type flip-flop to construct a counter. But to keep things simple, we will use the D-type flip-flop, (DFF) also known as a Data Latch, because a single data input and external clock signal are used, and is also positive edge triggered.

The D-type flip-flop, such as the TTL 74LS74, can be made from either S-R or J-K based edge-triggered flip-flops depending on whether you want it to change state either on the positive or leading edge (0 to 1 transition) or on the negative or trailing edge (1 to 0 transition) of the clock pulse. Here we will assume a positive, leading-edge triggered flip-flop. You can find more information in the following link about **D-type flip-flops**.

D-type Flip-flop and Truth Table



CLK	D	Q	$\overline{Q}$
↑	H	H	L
↑	L	L	H
L	X	no change	

The operation of a D-type flip-flop, (DFF) is very simple as it only has a single data input, called "D", and an additional clock "CLK" input. This allows a single data bit (0 or 1) to be stored under the control of the clock signal thus making the D-type flip-flop a synchronous device because the data on the inputs is transferred to the flip-flops output only on the triggering edge of the clock pulse.

So from the truth table, if there is a logic "1" (HIGH) on the Data input when a positive clock pulse is applied, the flip-flop SET's and stores a logic "1" at "Q", and a complimentary "0" at $\overline{Q}$. Likewise, if there is a LOW on the Data input when another positive clock pulse is applied, the flip-flop RESET's and stores a "0" at "Q", and a resulting "1" at $\overline{Q}$.

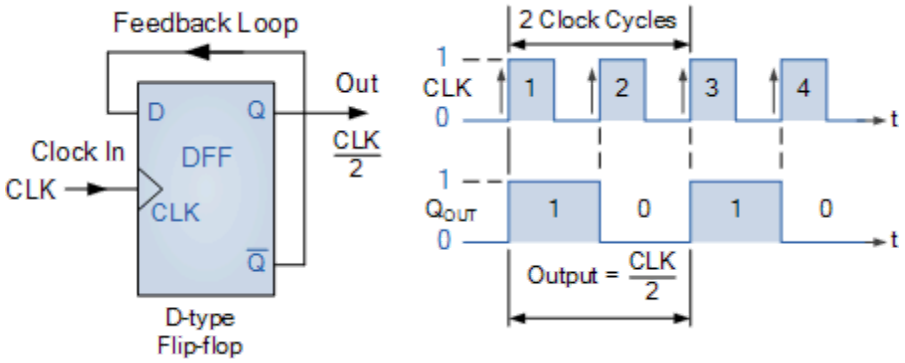
Then the output "Q" of the D-type flip-flop responds to the value of the input "D" when the clock (CLK) input is HIGH. When the clock input is LOW, the condition at "Q", either "1" or "0" is held until the next time the clock signal goes HIGH to logic level "1". Therefore the output at "Q" only changes state when the clock input changes from a "0" (LOW) value to a "1" (HIGH) making it a positive edge triggered D-type flip-flop. Note that negative edge-triggered flip-flops work in exactly the same way except that the falling edge of the clock pulse is the triggering edge.

So now we know how an edge-triggered D-type flip-flop works, lets look at connecting some together to form a MOD counter.

Divide-by-Two Counter

The edge-triggered D-type flip-flop is a useful and versatile building block to construct a MOD counter or any other type of sequential logic circuit. By connecting the $\overline{Q}$ output back to the "D" input as shown, and creating a feedback loop, we can convert it into a binary divide-by-two counter using the clock input only as the $\overline{Q}$ output signal is always the inverse of the Q output signal.

Divide-by-two Counter and Timing Diagram



The timing diagrams show that the "Q" output waveform has a frequency exactly one-half that of the clock input, thus the flip-flop acts as a frequency divider. If we added another D-type flip-flop so that the output at "Q" was the input to the second DFF, then the output signal from this second DFF would be one-quarter of the clock input frequency, and so on. So for an "n" number of flip-flops, the output frequency is divided by 2n, in steps of 2.

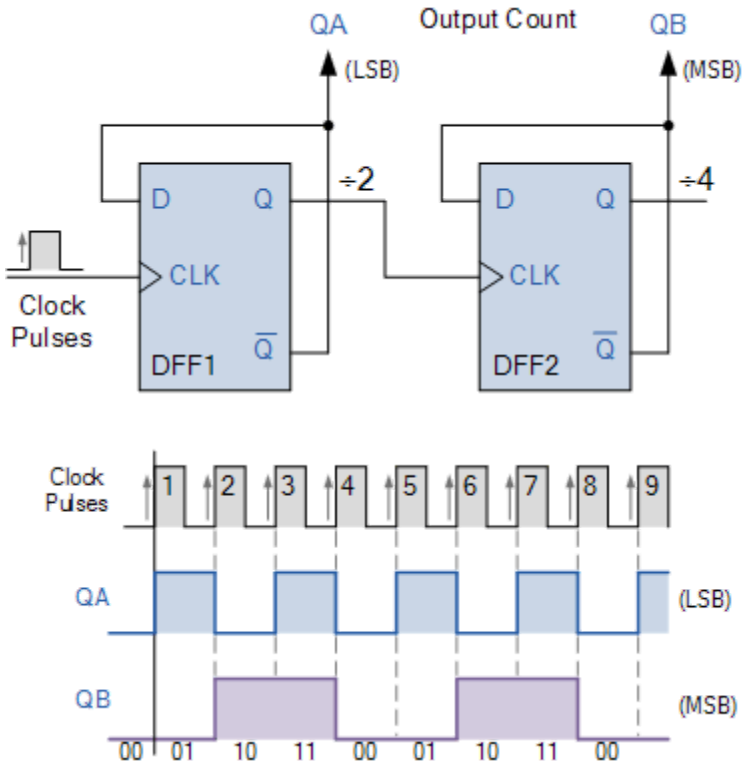
Note that this method of **frequency division** is very handy for use in sequential counting circuits. For example, a 60Hz mains frequency signal could be reduced to a 1Hz timing signal by using a divide-by-60 counter. A divide-by-6 counter would divide the 60Hz down to 10Hz which is then feed to a divide-by-10 counter to divide the 10Hz down to a 1Hz timing signal or pulse, etc.

MOD-4 Counter

Technically as well as being a 1-bit storage device, a single flip-flop on its own could be thought of as a MOD-2 counter, as it has a single output resulting in a count of two, either a 0 or 1, on the application of the clock signal. But a single flip-flop on its own produces a limited counting sequence, so by connecting together more flip-flops to form a chain, we can increase the counting capacity and construct a MOD counter of any value.

If a single flip-flop can be considered as a modulo-2 or MOD-2 counter, then adding a second flip-flop would give us a MOD-4 counter allowing it to count in four discrete steps. The overall effect would be to divide the original clock input signal by four. Then the binary sequence for this 2-bit MOD-4 counter would be: 00, 01, 10, and 11 as shown.

MOD-4 Counter and Timing Diagram



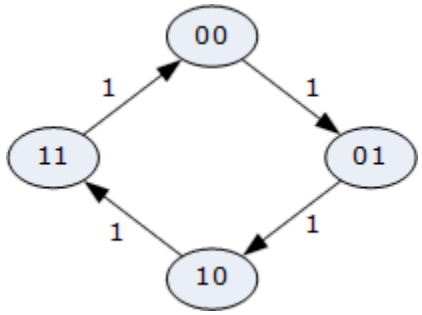
Note that for simplicity, the switching transitions of QA, QB and CLK in the above timing diagram are shown to be simultaneous even though this connection represents an asynchronous counter. In reality there would be a very small switching delay between the application of the positive going clock (CLK) signal, and the outputs at QA and QB.

We can show visually the operation of this 2-bit asynchronous counter using a truth table and state diagram.

MOD-4 Counter State Diagram

Clock Pulse	Present State	Next State	State Diagram

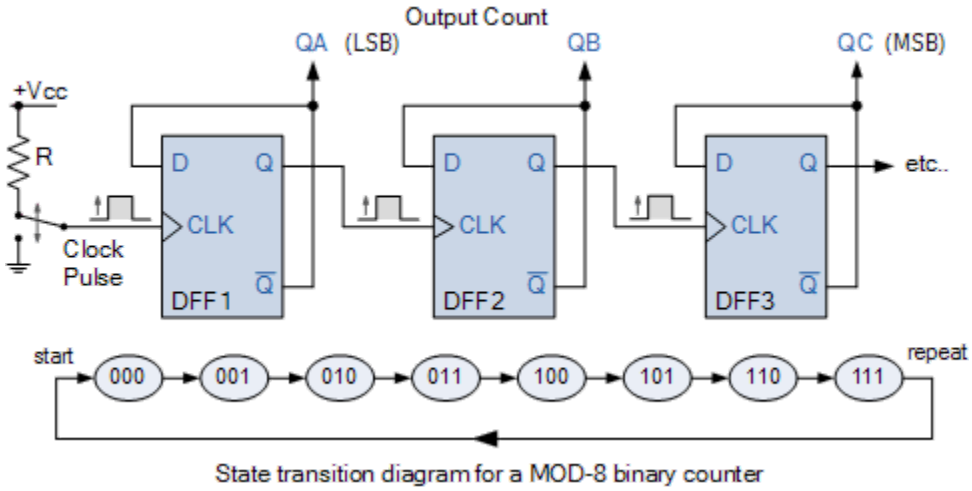
	Q_B	Q_A		Q_B	Q_A
0 (start)	0	0	⇒	0	1
1	0	1	⇒	1	0
2	1	0	⇒	1	1
3	1	1	⇒	0	0
4 (repeat)	0	0	⇒	0	1



We can see from the truth table of the counter, and by reading the values of QA and QB, when QA = 0 and QB = 0, the count is 00. After the application of the clock pulse, the values become QA = 1, QB = 0, giving a count of 01 and after the next clock pulse, the values become QA = 0, QB = 1, giving a count of 10. Finally the values become QA = 1, QB = 1, giving a count of 11. The application of the next clock pulse causes the count to return back to 00, and thereafter it counts continuously up in a binary sequence of: 00, 01, 10, 11, 00, 01 ...etc.

Then we have seen that a MOD-2 counter consists of a single flip-flop and a MOD-4 counter requires two flip-flops,allowing it to count in four discrete steps. We could easily add another flip-flop onto the end of a MOD-4 counter to produce a MOD-8 counter giving us a 2^3 binary sequence of counting from 000 up to 111, before resetting back to 000. A fourth flip-flop would make a MOD-16 counter and so on, in fact we could go on adding extra flip-flops for as long as we wanted.

MOD-8 Counter and State Diagram



State transition diagram for a MOD-8 binary counter

We can therefore construct mod counters to have a natural count of 2^n states giving counters with mod counts of 2, 4, 8, 16, and so on, before repeating itself. But sometimes it is necessary to have a modulus counter that resets its count back to zero during the normal counting process and does not have a modulo that is a power of 2. For example, a counter having a modulus of 3, 5, 6, or 10.

Counters of Modulo “m”

Counters, either synchronous or asynchronous progress one count at a time in a set binary progression and as a result an “n”-bit counter functions naturally as a modulo 2^n counter. But we can construct mod counters to count to any value we want by using one or more external logic gates causing it to skip a few output states and terminate at any count resetting the counter back to zero, that is all flip-flops have Q = 0.

In the case of modulo “m” counters, they do not count to all their possible states, but instead count to the “m” value and then return to zero. Obviously, “m” is a number smaller than 2^n, (**$m < 2^n$**). So how do we get a binary counter to return to zero part way through its count.

Fortunately, as well as counting, up or down, counters can also have additional inputs called *CLEAR* and *PRESET* which makes it possible to clear the count to zero, (all Q = 0) or to preset the counter to some initial value. The TTL 74LS74 has active-low Preset and Clear inputs.

Let’s assume for simplicity that the CLEAR inputs are all connected together and are active-high inputs allowing the flip-flops to operate normally when the Clear input is equal to 0 (LOW). But if the Clear input is at logic level “1” (HIGH), then the next positive edge of the clock signal will reset all the flip-flops into the state Q = 0, regardless of the value of the next clock signal.

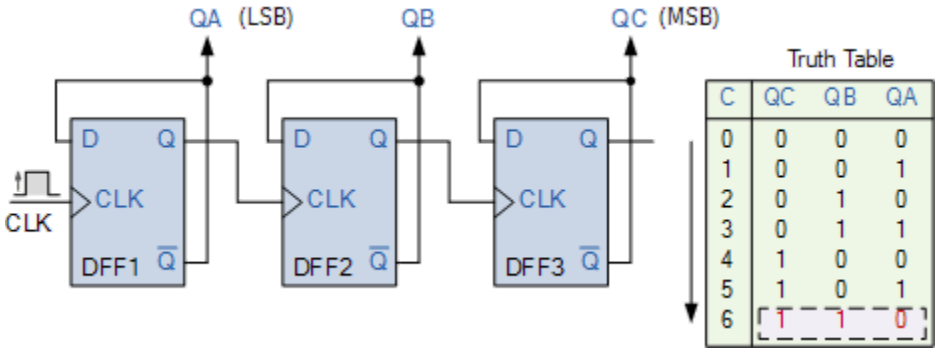
Note also that as all the Clear inputs are connected together, a single pulse can also be used to clear the outputs (Q) of all the flip-flops to zero before counting starts to ensure that the count actually starts from zero. Also some larger bit counters have an additional ENABLE or INHIBIT input pin which allows the counter to stop the count at any point in the counting cycle and hold its present state, before being allowed to continue counting again. This means the counter can be stopped and started at will without resetting the outputs to zero.

A Modulo-5 Counter

Suppose we want to design a MOD-5 counter, how could we do that. First we know that “m = 5”, so 2^n must be greater than 5. As $2^1 = 2$, $2^2 = 4$, $2^3 = 8$, and 8 is greater than 5, then we need a counter with three flip-flops (N = 3) giving us a natural count of 000 to 111 in binary (0 to 7 decimal).

Taking the MOD-8 counter above, the truth table for the natural count is given as:

MOD-8 Counter and Truth Table

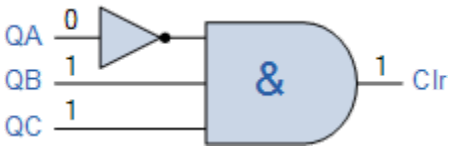


Truth Table

C	QC	QB	QA
0	0	0	0
1	0	0	1
2	0	1	0
3	0	1	1
4	1	0	0
5	1	0	1
6	1	1	0

As we are constructing a MOD-5 counter, we want the counter to reset back to zero after a count of 5. However, we can see from the attached truth table that the count of six gives us the output condition of: QA = 0, QB = 1, and QC = 1.

We can decode this output state of 011 (6) to give us a signal to clear (Clr) the counter back to zero with the help of a 3-input AND gate (TTL 74LS11) and an inverter or NOT gate, (TTL 74LS04).

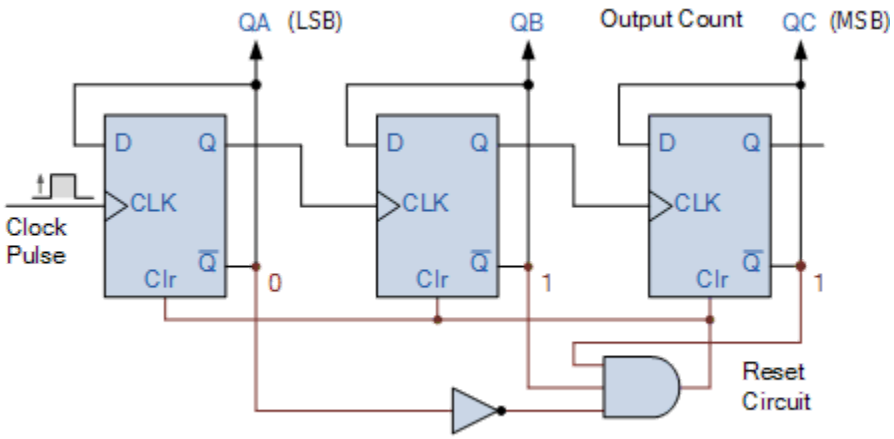


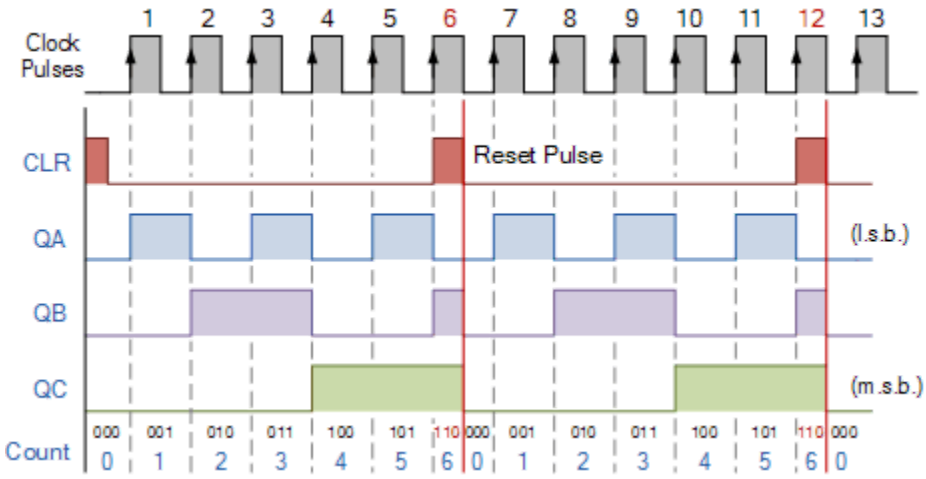
The inputs of the combinational logic circuit of the inverter and AND gate are connected to QA, QB, and QC respectively with the output of the AND gate at logic level “0” (LOW) for any combinations of the input other than the one we want.

In binary code, the output sequence count will look like this: 000, 001, 010, 011, 100, 101. But when it reaches the state of 011 (6), the combinational logic circuit will detect this 011 state and produce an output at logic level “1” (HIGH).

We can then use the resulting HIGH output from the AND gate to reset the counter back to zero after its output of 5 (decimal) count giving us the required MOD-5 counter. When the output from the combinational circuit is LOW it has no effect on the counting sequence.

MOD-5 Counter and Truth Table





Then we can use combinational logic decoding circuits around a basic counter, either synchronous or asynchronous to produce any type of MOD Counter we require as each of the counters unique output states can be decoded to reset the counter at the desired count.

In our simple example above, we have used a 3-input AND gate to decode the 011 state, but the first time that QA and QB are both at logic 1 is is when the count reaches six, so a 2-input AND gate connected to QA and QB could be used without the complication of the third input and the inverter.

However, one of the disadvantages of using asynchronous counters for producing a MOD counter of a desired count is that undesired effects called “glitches” can occur when the counter reaches its reset condition. During this brief time the outputs of the counter may take on an incorrect value, so it is sometimes better to use synchronous counters as modulo-m counters as all the flip-flops are clocked by the same clock signal so change state at the same time.

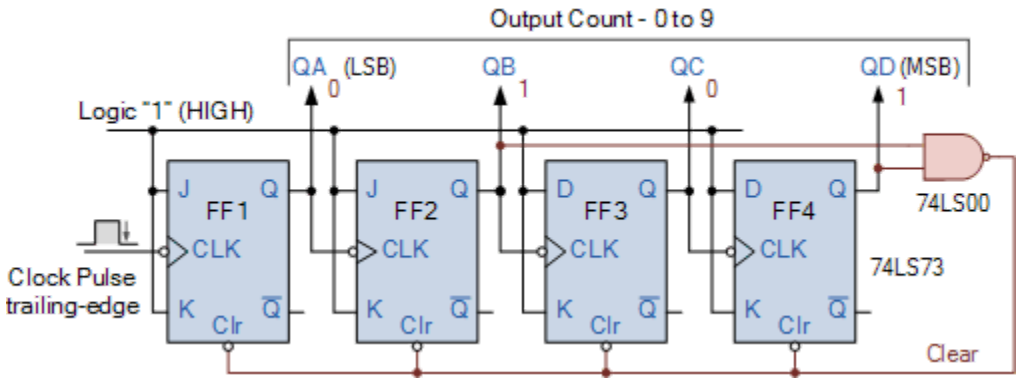
Modulus 10 Counter

A good example of a modulo-m counter circuit which uses external combinational circuits to produce a counter with a modulus of 10 is the Decade Counter. Decade (divide-by-10) counters such as the TTL 74LS90, have 10 states in its counting sequence making it suitable for human interfacing where a digital display is required.

The decade counter has four outputs producing a 4-bit binary number and by using external AND and OR gates we can detect the occurrence of the 9th counting state to reset the counter back to zero. As with other mod counters, it receives an input clock pulse, one by one, and counts up from 0 to 9 repeatedly.

Once it reaches the count 9 (1001 in binary), the counter goes back to 0000 instead of continuing on to 1010. The basic circuit of a decade counter can be made from JK flip-flops (TTL 74LS73) that switch state on the negative trailing-edge of the clock signal as shown.

MOD-10 Decade Counter



MOD Counter Summary

We have seen in this tutorial about **MOD Counters** that binary counters are sequential circuits that generate binary sequences of bits as a result of a clock signal and the state of a binary counter is determined by the specific combination formed by all the counters outputs together.

The number of different output states a counter can produce is called the modulo or modulus of the counter. The Modulus (or MOD-number) of a counter is the total number of unique states it passes through in one complete counting cycle with a mod-n counter being described also as a divide-by-n counter.

The modulus of a counter is given as: 2^n where n = number of flip-flops. So a 3 flip-flop counter will have a maximum count of $2^3 = 8$ counting states and would be called a MOD-8 counter. The maximum binary number that can be counted by the counter is $2^n - 1$ giving a maximum count of $(111)_2 = 2^3 - 1 = 7_{10}$. Then the counter counts from 0 to 7.

Common MOD counters include those with MOD numbers of 2, 4, 8 and 16 and with the use of external combinational circuits can be configured to count to any predetermined value other than one with a maximum 2^n modulus. In general, any arrangement of a "m" number of flip-flops can be used to construct any MOD counter.

A common modulus for counters with truncated sequences is ten (1010), called MOD-10. A counter with ten states in its sequence is known as a decade counter. Decade counters are useful for interfacing to digital displays. Other MOD counters include the MOD-6 or MOD-12 counter which have applications in digital clocks to display the time of day.

34 Comments

Write your comment here

SUBMIT

B Bett titus

Thanks your notes are very well ,one can understand with no difficulties.

Posted on November 07th 2018 | 12:36 pm

Reply

N Niraj Kumar singh

NIC explained

Posted on October 29th 2018 | 10:31 am

Reply

B Bazamewafa shaikh

Thanks, this is a great electronic tutorial

Posted on October 18th 2018 | 8:00 pm

Reply

S

Samuel

Dear Sir , your Electronics notes are very nice. Would you send me practical electronics circuits to upgrade myself?
Thank you ahead.

Samuel

Posted on October 10th 2018 | 7:55 am

Reply

A

Afsana khan

I want to make project for 100 mods using 7490 ic……..can u give me a tips how i can make project…..one by one….plzzz help me as soon as possible….i want truth table …circuit diagram…..and practicle vedio

Posted on September 29th 2018 | 8:35 am

Reply

S

SAI KRISHNA PAKALA

Respected sir, counters, and waveforms are not matching, and check once the mod5 counter also

Posted on September 03rd 2018 | 2:40 pm

Reply

S

Saurabh kumar

Nice thk

Posted on August 31st 2018 | 5:08 am

Reply

M

Manikanta Swami Minam

For 2 nd flipfolp triggering happens only at negedge

Posted on July 23rd 2018 | 2:18 pm

Reply

G

GOVINTHARAJ

sir please arrenged this
MODULO 5 CONTROLLER – EYAS525F001 – 230 V INPUT

Posted on July 21st 2018 | 4:05 am

Reply

R

Rowland Williams

Knowledge sharing is definitely part of God's Grace for humanity. As you share you receive abundantly. In the Name of Jesus Christ, God bless you all.

Posted on April 20th 2018 | 7:26 pm

Reply

View More