**Chapter 6: Registers and Counters**

Table of Contents

[Registers 2](#_Toc44600679)

[Shift Registers 3](#_Toc44600680)

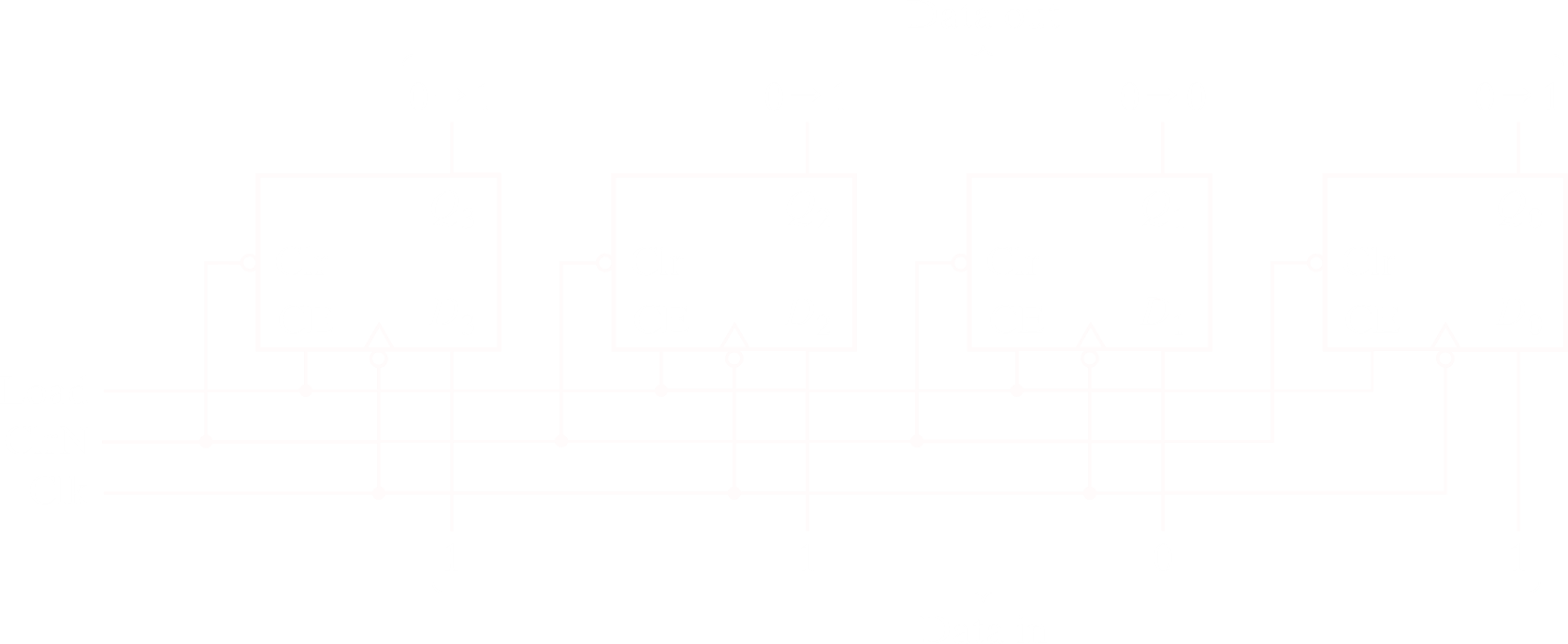
[Counters 4](#_Toc44600681)

## Registers

A register is a group of flip flops, each of which is capable of storing one bit of information. An -bit register is a group of flip flops, and can thus store bits. A register may also contain a combinational circuit that controls the operation of the flip flops.

The simplest register is a 4-bit register consisting of 4 D Flip Flops. Each flip flop is connected to the same clock, and are thus operational at the same time. This means 4-bits of information given to the flip flops are stored simultaneously. Each flip flop is also connected to a single CLEAR input. This allows the entire register to be reset at once before new information is stored.

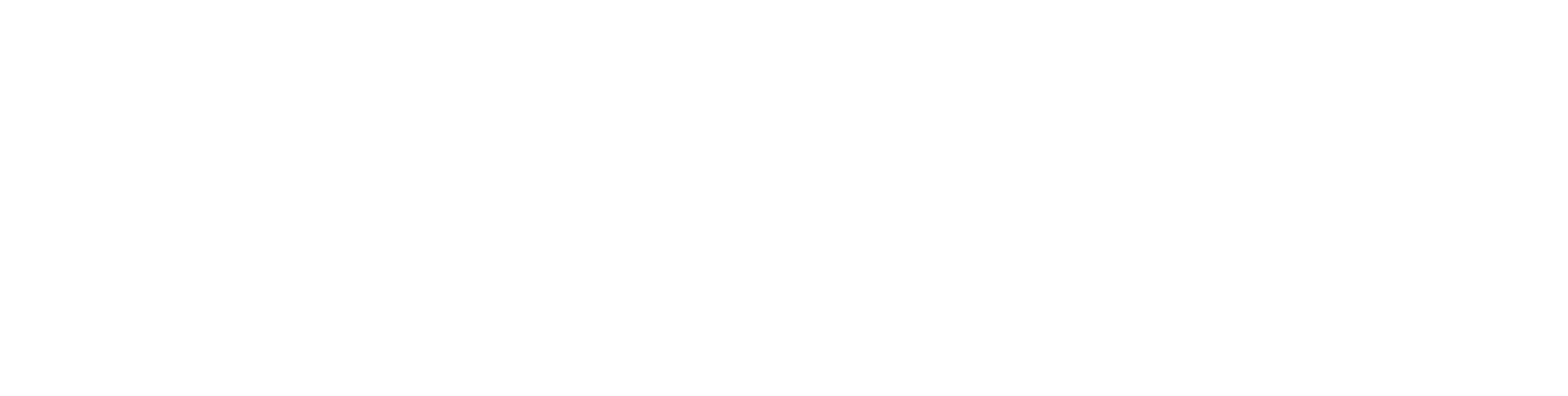
This type of register still poses a small problem. If for example, the clock has a frequency of , the information will only be stored for , and will then be replaced by new information. To prevent this from happening, a single load input is connected to each of the flip flops. This essentially acts as a switch. If the load input is low, regardless of the other inputs, the flip flops will not be active and the data in the register will be stored. A register with a load input is called an -bit register with parallel load.



Notice the symbol for the clock in the diagram. This symbol indicates a negative-edge triggered clock.

## Shift Registers

A shift register is capable of shifting the information held in one cell to its neighbor. The simplest possible shift register is a group of D Flip Flops as shown below.



At each clock pulse, the contents of the shift register are shifted one bit position to the right. The configuration only supports shifting from left to right. Input is given serially to the left-most flip flop and output is received serially from the right-most flip flop. Thus, this is known as a Serial Input, Serial Output (SISO) register. Serial registers take values one by one as opposed to parallel registers, that take inputs all at once.

Other types of shift registers include Serial Input, Parallel Output (SIPO) registers, Parallel Input, Serial Output (PISO) registers and Parallel Input, Parallel Output (PIPO) registers.

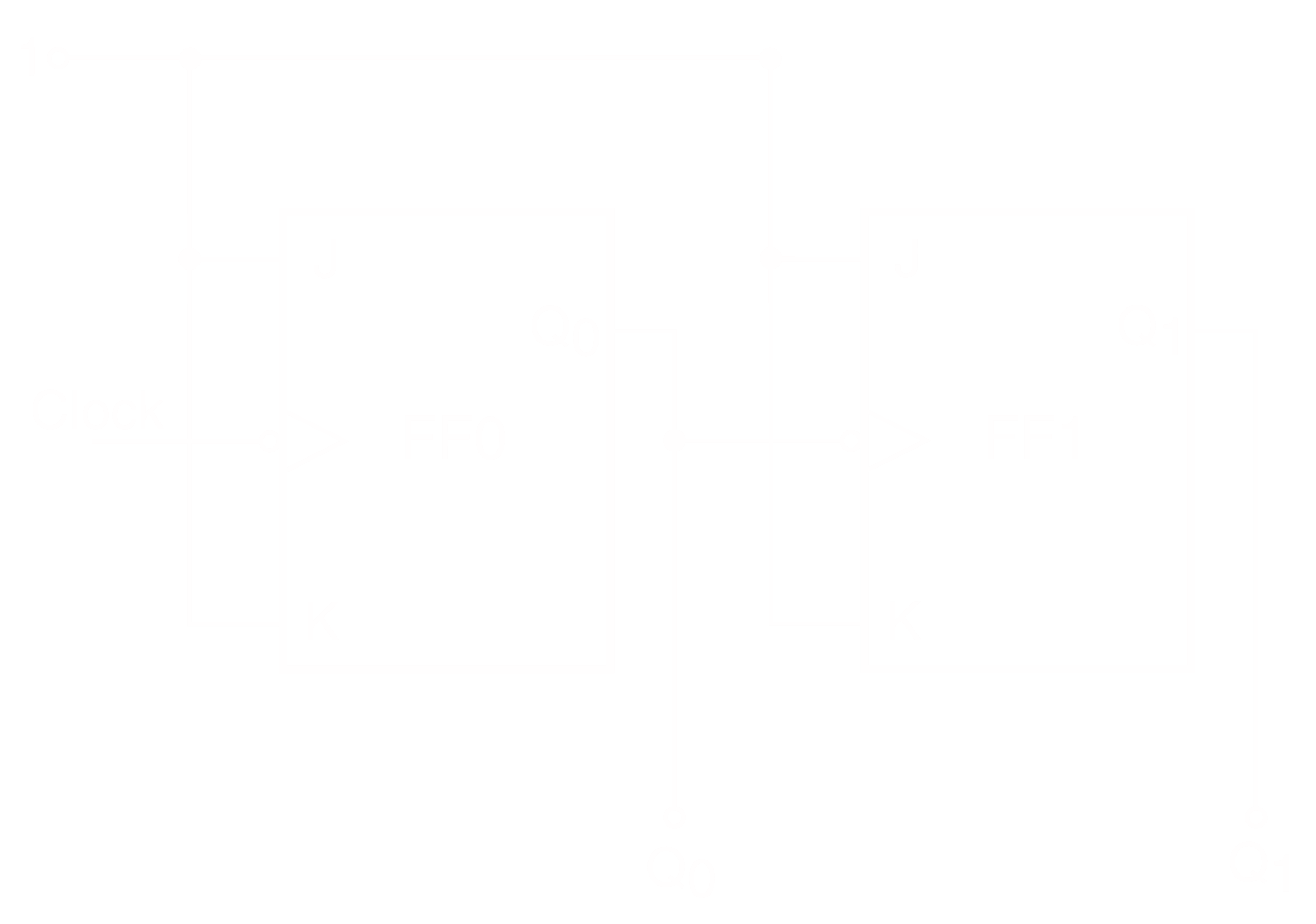
Bidirectional shift registers also exist that allows information to be shifted both from left to right and right to left.

A Universal Shift register is capable of both serial and parallel transfer and is bidirectional. It makes use of multiplexers to control its operation.

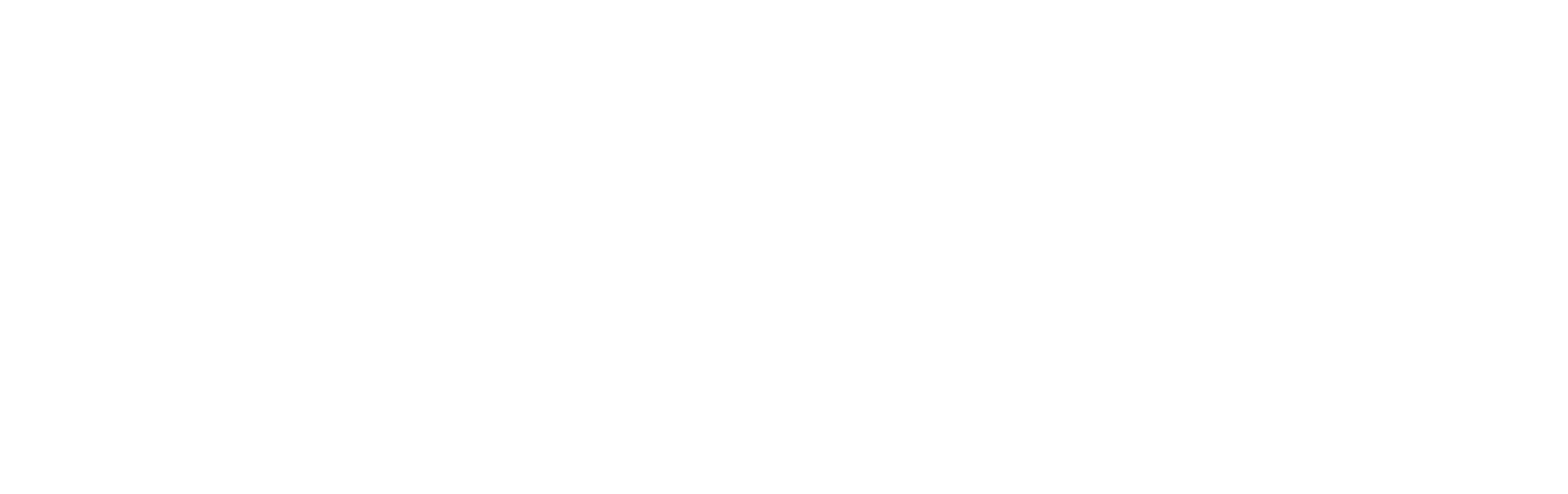
## Counters

A counter is a specific type of register that goes through a predetermined sequence of states upon application of input pulses. There are multiple types of counters such as up-counters, down counters and up-down counters (which can be set to count in either direction). Counters do not take user input.

Consider the following circuit. It is two JK Flip Flops, with every input set to (thus both are acting as T Flip Flops). The output from goes to the clock of the second flip flop.



The signal diagram for the circuit is given below:



Since the circuit is negative edge triggered, every time the CLK input goes from to , the output of changes. This results in giving the same output as the clock, but with double the time period. in turn is the clock for , which again gives double the time period of , or four times the time period of the initial CLK input.

Thus, the following pattern is set up:

|  |  |  |
| --- | --- | --- |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |

Taking the output in binary form as , for each clock pulse, the output increases by in binary. Since this is a 2-bit counter, it goes from to , and then starts again from .

An -bit counter would go from to , with the time period of the last flip flop output being times the time period of the initial CLK input.

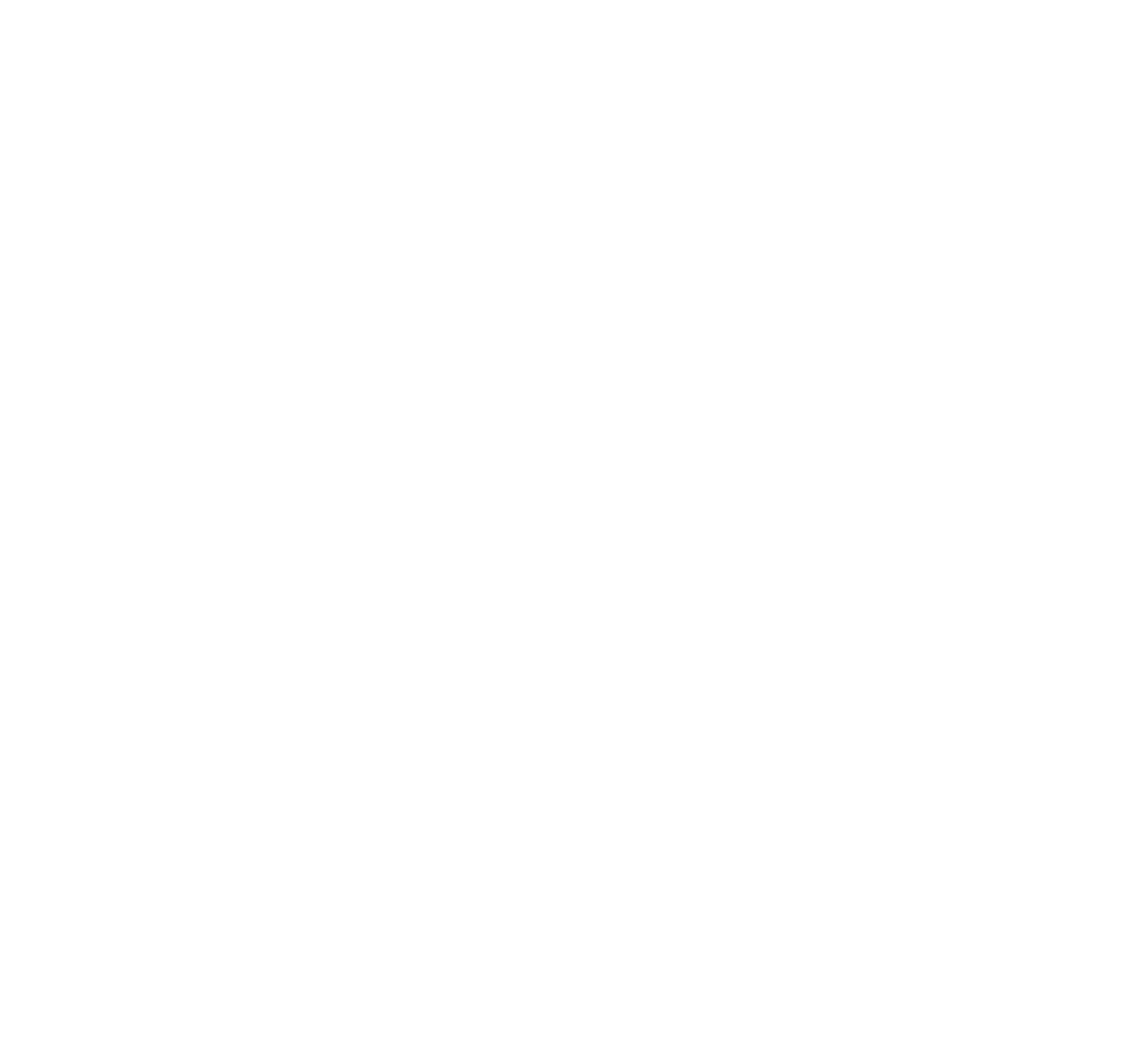
The counter shown above is an asynchronous counter. The clocks of such counters are not synced, with the output of one flip flop acting as the clock for the next flip flop. This results in a simple circuit, but a slow counter. A synchronous counter on the other hand uses the same clock, but results in a more complicated circuit. However, the counter is as fast as an actual clock.

The counter shown above is also called a ripple up counter. For a ripple down counter, the clock inputs for each flip flop would come from the negative result of the previous flip flop, i.e. from instead of . For a ripple up-down counter, we would need to place a small circuit in front of each clock that would check the value of and use or accordingly. This can be done using two AND gates and an OR gate.

For a positive edge triggered circuit, the reverse would occur with the output from being used as the clock of the next flip flop for up counters and the output from being used as the clock of the next flip flop for down counters.

We will now follow the design procedure to create a 2-bit synchronous up counter.

State Diagram:



Note that a counter does not take any input, and the output is just the state that has just been reached.

For a 2-bit synchronous down counter, the arrows would be pointed in the opposite directions.

State Table:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Present State | | Next State | | Output | |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |

Notice that the input column has not been included since there is no input.

Next, we must decide which flip flop we would like to use and how many of them are required. For this example, a JK Flip Flop will be used. Since this is a 2-bit counter, 2 JK Flip Flops will be required.

The excitation table for a single JK Flip Flop (covered in Chapter 5) looks like this:

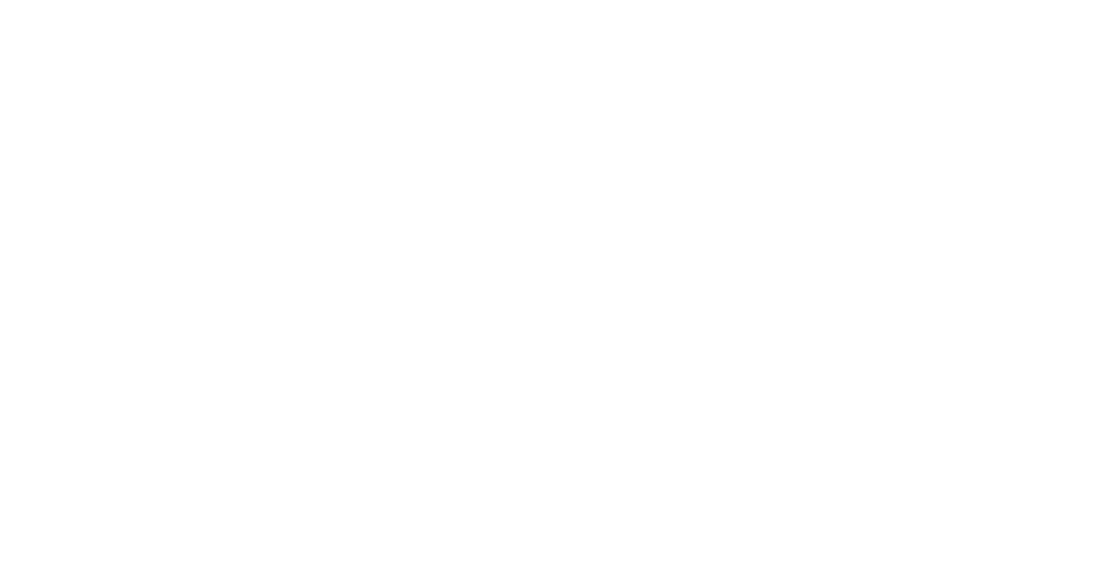
|  |  |  |  |
| --- | --- | --- | --- |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

Thus, the circuit excitation table looks like this:

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Present State | | Next State | | Flip Flop Inputs | | | | Outputs | |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |

Thus,

And thus, the circuit:



Now let us consider a 4-bit BCD counter. This can also be created as a synchronous or asynchronous counter as required. The main difference between a 4-bit BCD counter and a 4-bit binary counter is that BCD codes are only allowed to go up to 9. Thus, when the counter has reached counter has reached 10, i.e. , it must be reset to . This can be done with the help of the preset and clear functions. When is reached, we will set preset to and clear to for each flip flop, thus setting them all to . This is done using a NAND gate (since clear must be set to , not ). Note that preset can constantly be set to , since that will have no effect on the circuit. Also make note of how the preset and clear connections are made.

T

Q

Q

B

T

Q

Q

C

T

Q

Q

D

T

Q

Q

A

CLK

1

S

S

S

S

3

2

1

0

Notice that this same process can be followed to make any given counter count up to a specific value.

The number of states a counter counts is known as its modulus value. For example, the above circuit can be called a MOD-10 counter since it goes over 10 states from 0000 to 1001. A normal 4-bit counter would be a MOD-16 counter. A MOD-11 counter designed using a MOD-16 counter would be a 4-bit counter that counts from 0000 to 1010, over 11 states.