**A**

**SUMMARY REPORT**

**ON**

**FLIP FLOPS, REGISTERS AND COUNTERS**

**FOR COE312**

**BY**

**OBI IFEANYI KINGSLEY**

**17/ENG02/057**

**TO BE SUBMITTED TO**

**THE DEPARTMENT OF ELECTRICAL/ELECTRONICS AND COMPUTER ENGINEERING,ABUAD**

**DATE OF SUBMISSION 10TH MAY.2020**

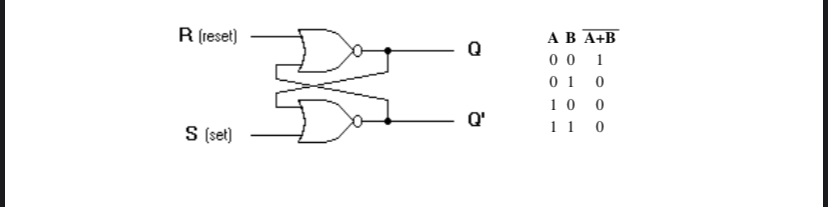
What is a flip flop?

A flip-flop is a specific kind of latch that has two conditions of stability, is enabled for a short time, and can be edge-triggered. In this lesson we will start by looking at a simple circuit that is able to remember it's state (memory) and then we will learn about different latch improvements that allow for better functionality. Binary information can enter a flip-flop in a variety of ways and gives rise to different types of flip-flops.

RS Flip-Flop

RS flip-flop is the simplest possible memory element. It can be constructed from two NAND gates or two NOR gates. Let us understand the operation of the RS flip-flop using NOR gates as shown below using the truth table for ‘A NOR B’ gate. The inputs R and S are referred to as the Reset and Set inputs, respectively. The outputs Q and Q' are complements of each other and are referred to as the normal and complement outputs, respectively. The binary state of the flip-flop is taken to be the value of the normal output. When Q=1 and Q'=0, it is in the set state (or 1-state). When Q=0 and Q'=1, it is in the reset/clear state (or 0-state).

Circuit Diagram:



• S=1 and R=0: The output of the bottom NOR gate is equal to zero, Q'=0. Hence both inputs to the top NOR gate are equal to 0, thus, Q=1. Hence, the input combination S=1 and R=0 leads to the flip-flop being set to Q=1.

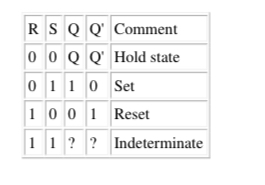
• S=0 and R=1: Similar to the arguments above, the outputs become Q=0 and Q'=1. We say that the flip-flop is reset.

• S=0 and R=0: Assume the flip-flop was previously in set (S=1 and R=0) condition. Now changing S to 0 results Q' still at 0 and Q=1. Similarly, when the flip-flop was previously in a reset state (S=0 and R=1), the outputs do not change. Therefore, with inputs S=0 and R=0, the flip-flop holds its state.

• S=1 and R=1: This condition violates the fact that both outputs are complements of each other since each of them tries to go to 0, which is not a stable configuration. It is impossible to predict which output will go to 1 and which will stay at 0. In normal operation this condition must be avoided by making sure that 1's are not applied to both inputs simultaneously, thus making it one of the main disadvantages of RS flip-flop.

All the above conditions are summarized in the characteristic table below:

Characteristic Table:

Debounce circuit

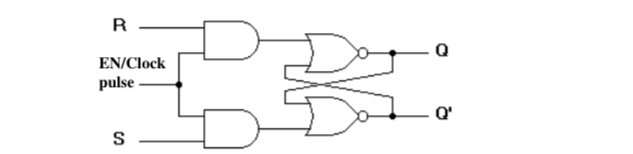
An elementary example using this flip-flop is the debounce circuit. Suppose a piece of electronics is to change state under the action of a mechanical switch. When this switch is moved from position S to R (S=0, R=1), the contacts make and break several times at R before settling to good contact. It is desirable that the electronics should respond to the first contact and then remain stable, rather than switching back and forth as the circuit makes and breaks. This is achieved by RS flip-flop which is reset to Q=0 by the first signal R=1 and remains in a fixed state until the switch is moved back to position S, when the signal S=1 sets the flip-flop to Q=1.

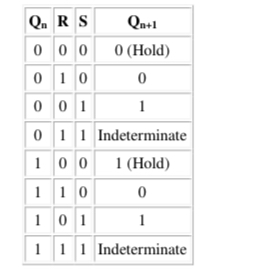
Gated or Clocked RS Flip-Flop

It is sometimes desirable in sequential logic circuits to have a bistable RS flip-flop that only changes state when certain conditions are met regardless of the condition of either the Set or the Reset inputs. By connecting a 2-input AND gate in series with each input terminal of the RS NOR Flip-flop a Gated RS Flip-flop can be created. This extra conditional input is called an "Enable" input and is given the prefix of "EN" as shown below. When the Enable input "EN" = 0, the outputs of the two AND gates are also at logic level 0, (AND Gate principles) regardless of the condition of the two inputs S and R, latching the two outputs Q and Q’ into their last known state. When the enable input "EN" = 1, the circuit responds as a normal RS bistable flip-flop with the two AND gates becoming transparent to the Set and Reset signals. This Enable input can also be connected to a clock timing signal adding clock synchronisation to the flip-flop creating what is sometimes called a "Clocked SR Flip-flop".

So a Gated/Clocked RS Flip-flop operates as a standard bistable latch but the outputs are only activated when a logic "1" is applied to its EN input and deactivated by a logic "0". The property of this flip-flop is summarized in its characteristic table where Qn is the logic state of the previous output and Qn+1 is that of the next output and the clock input being at logic 1 for all the R and S input combinations.

Circuit Diagram:



Characteristic table:

D FLIP-FLOP

An RS flip-flop is rarely used in actual sequential logic because of its undefined outputs for inputs R= S= 1. It can be modified to form a more useful circuit called D flip-flop, where D stands for data. The D flip-flop has only a single data input D as shown in the circuit diagram. That data input is connected to the S input of an RS flip-flop, while the inverse of D is connected to the R input. To allow the flip-flop to be in a holding state, a D-flip flop has a second input called Enable, EN. The Enable- input is AND-ed with the D-input.

• When EN=0, irrespective of D-input, the R = S = 0 and the state is held.

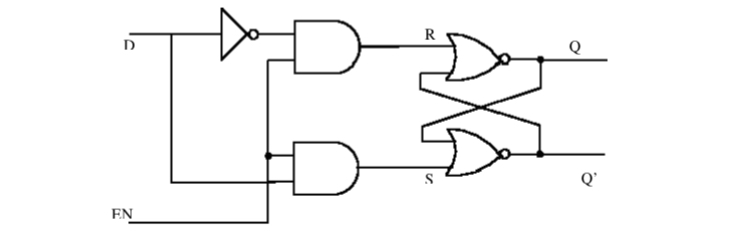
• When EN= 1, the S input of the RS flip-flop equals the D input and R is the

inverse of D. Hence, output Q follows D, when EN= 1.

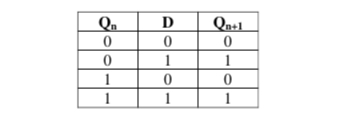
• When EN returns to 0, the most recent input D is ‘remembered'.

The circuit operation is summarized in the characteristic table for EN=1.

Circuit Diagram:



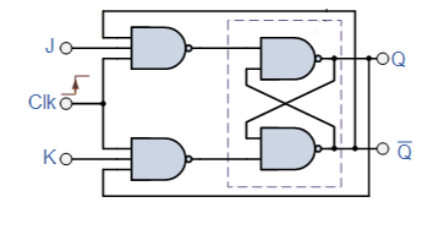
Characterized table:

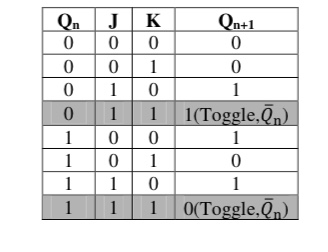


JK FLIP-FLOP:

The JK flip flop (JK means Jack Kilby, a Texas instrument engineer, who invented it) is the most versatile flip-flop, and the most commonly used flip flop. Like the RS flip-flop, it has two data inputs, J and K, and an EN/clock pulse input (CP). Note that in the following circuit diagram NAND gates are used instead of NOR gates. It has no undefined states, however. The fundamental difference of this device is the feedback paths to the AND gates of the input, i.e. Q is AND-ed with K and CP and Q’ with J and CP.

Circuit Diagram:

 Characteristic table:



The JK flip-flop has the following characteristics:

• If one input (J or K) is at logic 0, and the other is at logic 1, then the output is set or reset (by J and K respectively), just like the RS flip-flop.

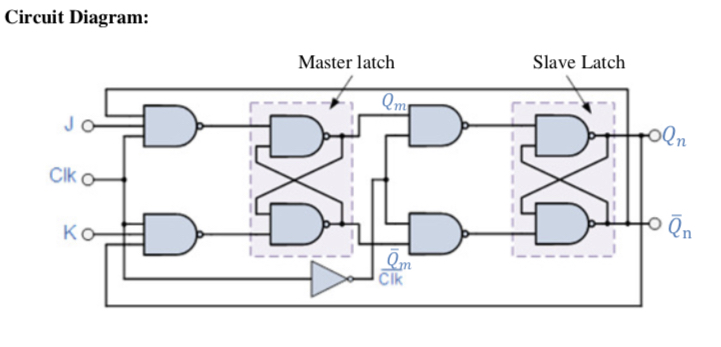
• If both inputs are 0, then it remains in the same state as it was before the clock pulse occurred; again like the RS flip flop. CP has no effect on the output.

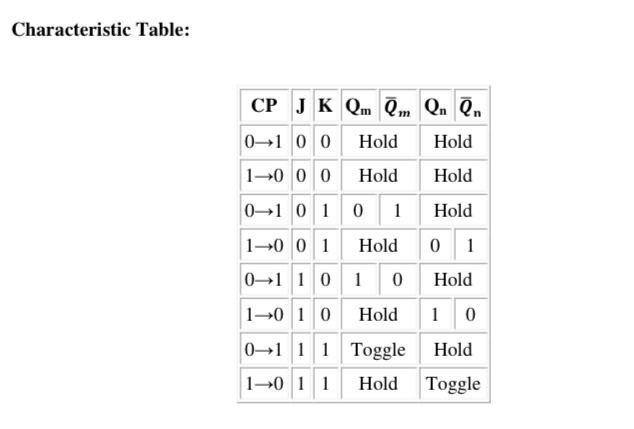
• If both inputs are high, however the flip-flop changes state whenever a clock pulse occurs; i.e., the clock pulse toggles the flip-flop again and again until the CP goes back to 0 as shown in the shaded rows of the characteristic table above. Since this condition is undesirable, it should be eliminated by an improvised form of this flip-flop as discussed in the next section.

MASTER-SLAVE JK FLIP-FLOP:

Although JK flip-flop is an improvement on the clocked SR flip-flop it still suffers from timing problems called "race" if the output Q changes state before the timing pulse of the clock input has time to go "OFF", so the timing pulse period (T) must be kept as short as possible (high frequency). As this is sometimes not possible with modern TTL IC's the much improved Master-Slave J-K Flip-Flop was developed. This eliminates all the timing problems by using two SR flip-flops connected together in series, one for the "Master" circuit, which triggers on the leading edge of the clock pulse and the other, the "Slave" circuit, which triggers on the falling edge of the clock pulse.

The master-slave JK flip flop consists of two flip flops arranged so that when the clock pulse enables the first, or master, it disables the second, or slave. When the clock changes state again (i.e., on its falling edge) the output of the master latch is transferred to the slave latch. Again, toggling is accomplished by the connection of the output with the input AND gates.



T FLIP-FLOP:

The T flip-flop is a single input version of the JK flip-flop. The T flip-flop is obtained from the JK type if both inputs are tied together.

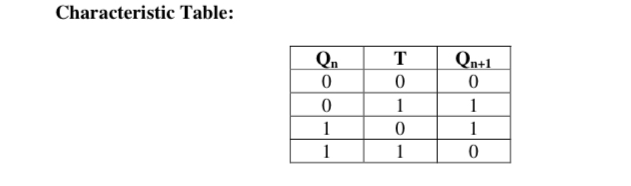
Circuit Diagram:

Same as Master-Slave JK flip-flop with J=K=1

• The toggle, or T, flip-flop is a bistable device, where the output of the T flip-flop "toggles" with each clock pulse.

• Till CP=0, the output is in hold state (three input AND gate principle).

• When CP=1, for T=0, previous output is memorized by the circuit. When T=1 along with the clock pulse, the output toggles from the previous value as given in the

characteristic table below.

COUNTERS AND REGISTERS

What is a counter?

Counter is a sequential circuit. A digital circuit which is used for a counting pulses is known counter. Counter is the widest application of flip-flops. It is a group of flip-flops with a clock signal applied. Counters are of two types.

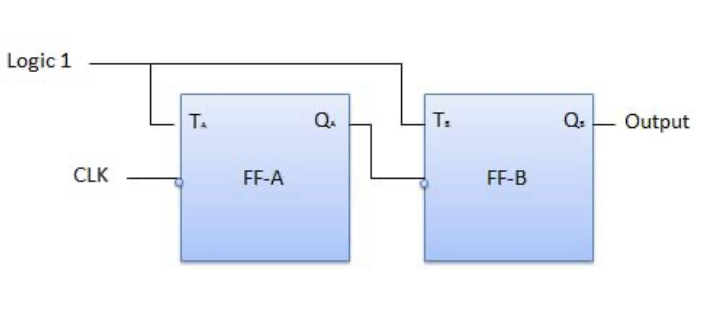
Asynchronous or ripple counters.

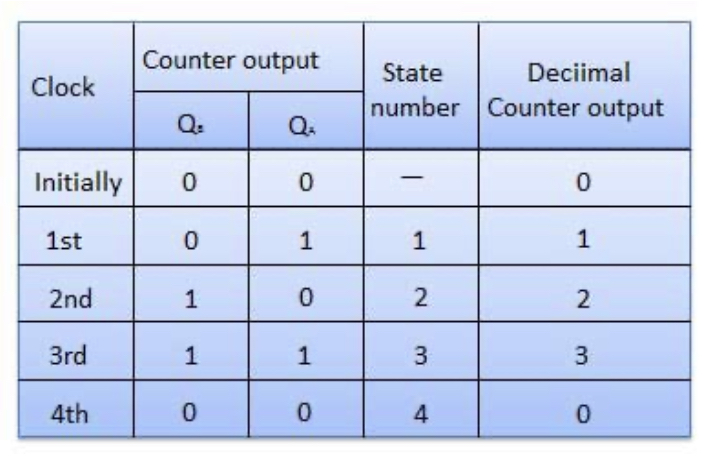
Synchronous counters.

Asynchronous or ripple counters

The logic diagram of a 2-bit ripple up counter is shown in figure. The toggle (T) flip-flop are being used. But we can use the JK flip-flop also with J and K connected permanently to logic 1. External clock is applied to the clock input of flip-flop A and QA output is applied to the clock input of the next flip-flop i.e. FF-B.

Logical Diagram

Truth table



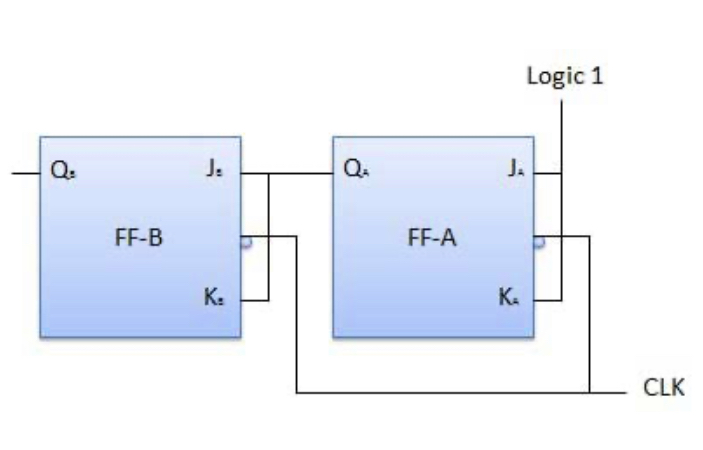
Synchronous counters

If the "clock" pulses are applied to all the flip-flops in a counter simultaneously, then such a counter is called as synchronous counter.

2-bit Synchronous up counter

The JA and KA inputs of FF-A are tied to logic 1. So FF-A will work as a toggle flip-flop. The JB and KB inputs are connected to QA.

Logical Diagram



Classification of counters

Depending on the way in which the counting progresses, the synchronous or asynchronous counters are classified as follows −

Up counters

Down counters

Up/Down counters

UP/DOWN Counter

Up counter and down counter is combined together to obtain an UP/DOWN counter. A mode control (M) input is also provided to select either up or down mode. A combinational circuit is required to be designed and used between each pair of flip-flop in order to achieve the up/down operation.

Type of up/down counters

UP/DOWN ripple counters

UP/DOWN synchronous counter

UP/DOWN Ripple Counters

In the UP/DOWN ripple counter all the FFs operate in the toggle mode. So either T flip-flops or JK flip-flops are to be used. The LSB flip-flop receives clock directly. But the clock to every other FF is obtained from (Q = Q bar) output of the previous FF.

UP counting mode (M=0) − The Q output of the preceding FF is connected to the clock of the next stage if up counting is to be achieved. For this mode, the mode select input M is at logic 0 (M=0).

DOWN counting mode (M=1) − If M = 1, then the Q bar output of the preceding FF is connected to the next FF. This will operate the counter in the counting mode.

Example

3-bit binary up/down ripple counter.

3-bit − hence three FFs are required.

UP/DOWN − So a mode control input is essential.

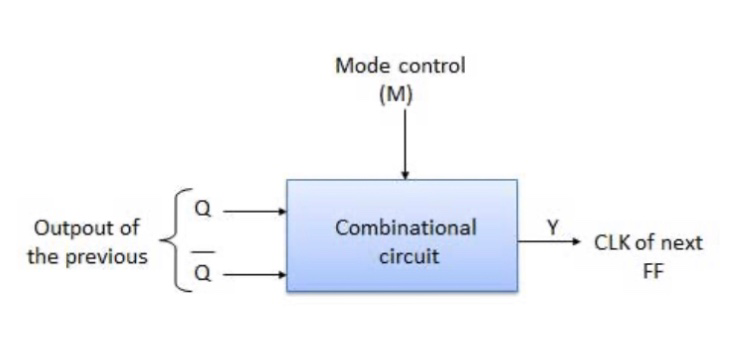
For a ripple up counter, the Q output of preceding FF is connected to the clock input of the next one.

For a ripple up counter, the Q output of preceding FF is connected to the clock input of the next one.

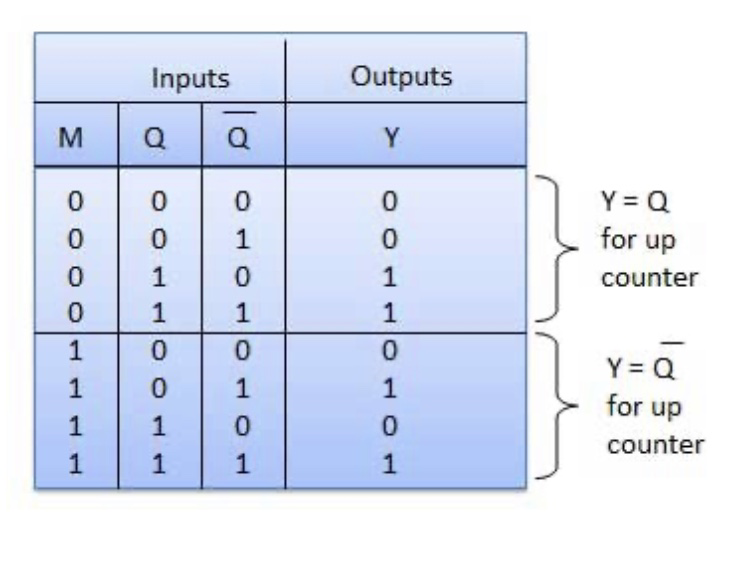
For a ripple down counter, the Q bar output of preceding FF is connected to the clock input of the next one.

Let the selection of Q and Q bar output of the preceding FF be controlled by the mode control input M such that, If M = 0, UP counting. So connect Q to CLK. If M = 1, DOWN counting. So connect Q bar to CLK.

Block Diagram



Truth table



Modulus Counter (MOD-N Counter)

The 2-bit ripple counter is called as MOD-4 counter and 3-bit ripple counter is called as MOD-8 counter. So in general, an n-bit ripple counter is called as modulo-N counter. Where, MOD number = 2n.

Type of modulus

2-bit up or down (MOD-4)

3-bit up or down (MOD-8)

4-bit up or down (MOD-16)

REGISTERS

What is a register?

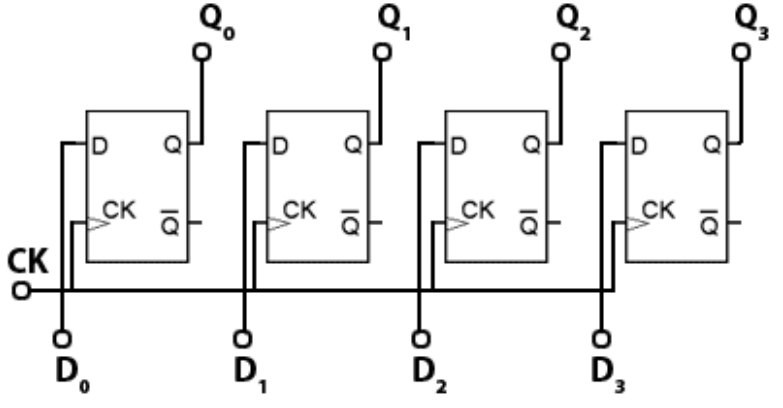
Registers are groups of flip-flops , where each flip-flop is capable of storing one bit of information. An n-bit register is a group of n flip-flops. The basic function of a register is to hold information in a digital system and make it available to the logic elements for the computing process. Registers consist of a finite number of flip-flops. Since each flip-flop is capable of storing either a "0" or a "1", there is a finite number of 0-1 combinations that can be stored into a register. Each of those combinations is known as state or content of the register. With flip-flops we can store data bitwise but usually data does not appear as single bits. Instead it is common to store data words of n bit with typical word lengths of 4, 8, 16, 32 or 64 bit. Thus, several flip-flops are combined to form a register to store whole data words. Registers are synchronous circuits thus all flip-flops are controlled by a common clock line. As registers are often used to collect serial data they are also called accumulators. There exist several types of registers as there are –

Parallel In - Parallel Out (PIPO) Registers

An electronic register is a form of memory that uses a series of flip-flops to store the individual bits of a binary word, such as a byte (8 bits) of data. The length of the stored binary word depends on the number of flip-flops that make up the register. A simple 4-bit register is illustrated and consists of four D Type flip-flops, sharing a common clock input, providing synchronous operation ensuring all bits are stored at exactly the same time.

The binary word to be stored is applied to the four D inputs and is remembered by the flip-flops at the rising edge of the next clock (CK) pulse. The stored data can then be read from the Q outputs at any time, as long as power is maintained, or until a change of data on the D inputs is stored by a further clock pulse, which overwrites the previous data.

Different types of register are generally classified by the method of storage and readout used; this basic form of register is therefore classified as a ‘Parallel In/Parallel Out’ (PIPO) register.

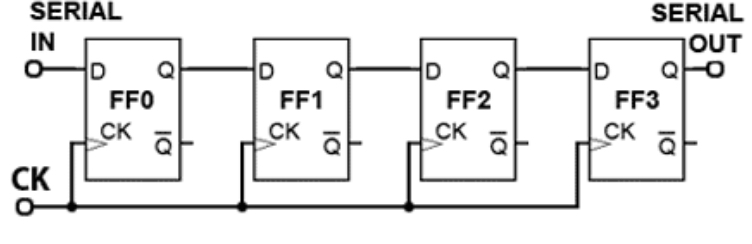


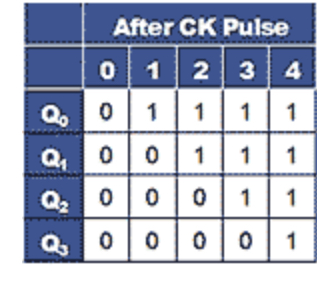
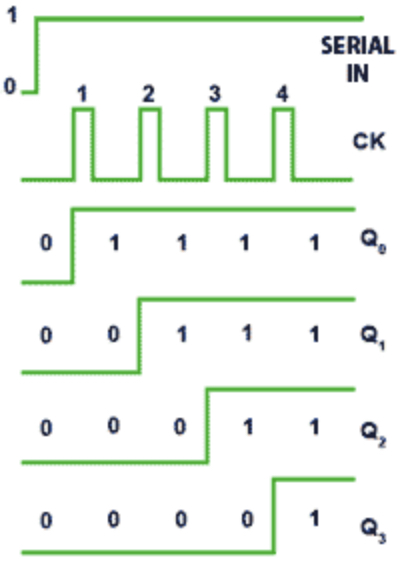
Shift register

Shift registers have a similar structure to the PIPO register but have the added ability to shift the stored binary word left or right, one bit at a time. This makes them extremely useful for many applications. They are used in handling serial data and converting it to parallel form or back again to serial form, and therefore are an essential component in communication systems. Shift registers are also essential in arithmetic circuits where binary numbers may be shifted right (and so divided by two), or left (multiplied by two) as part of a calculation. Shift registers can be used to delay the passage of data at a particular point in a circuit. As the data is shifted one bit at a time from input to output, the amount of delay will depend on the number of flip-flops in the register and the frequency of the clock pulses driving the shift register. Because a number of serial bits of data are stored as they enter the input, and are then recovered from the output at some later time, this action can also be described as a serial memory, or as a digital delay line.

The simple storage register shown can be modified to a shift register by connecting the output of one flip-flop into the input of the next, as shown . The basis of shift register circuits is the D-type flip-flop, but the clocked SR or the JK flip-flop may also be converted to D-types by the inclusion of an inverter between S and R or between J and K. In all cases the clock input is in synchronous mode.

The serial input of the shift register is the D input of the first flip-flop, and the serial output is the Q output of the last flip-flop in the chain. The logic state at the serial input appears at the output, a number of clock pulses (equal to the number of flip flops) later.

State table

Timing diagram

Modes of operating shift register operation

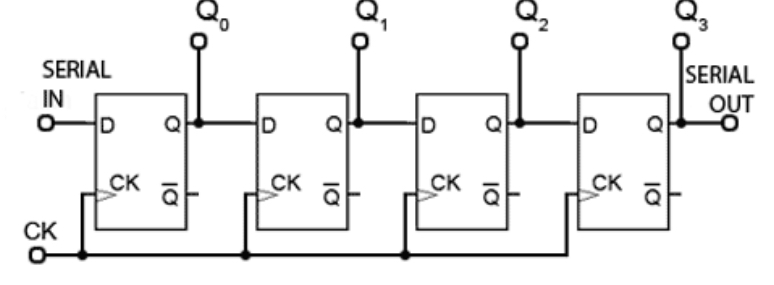
SISO

A State Table and Timing Diagram illustrating the operation of is shown where the timing diagram shows the time relationship between the CK pulses and changes at the Q outputs of the circuit. It can be seen that if the serial input goes from 0 to 1 just before CK pulse 1, the Q output of flip-flop FF0 will go high at the rising edge of CK pulse 1. At the next clock pulse rising edge, the logic 1 will be transferred to FF1 and so on until it reaches FF3, and the serial output.

The same action can also be illustrated by a State Table, which, rather than showing timing data, shows the states of the four Q outputs after each clock pulse. After each CK pulse one more flip-flop output is set to 1 until, after 4 pulses, column 4 shows that all Q outputs, including the serial output, are at logic 1. This form of operation is called ‘serial in/serial out’ or SISO.

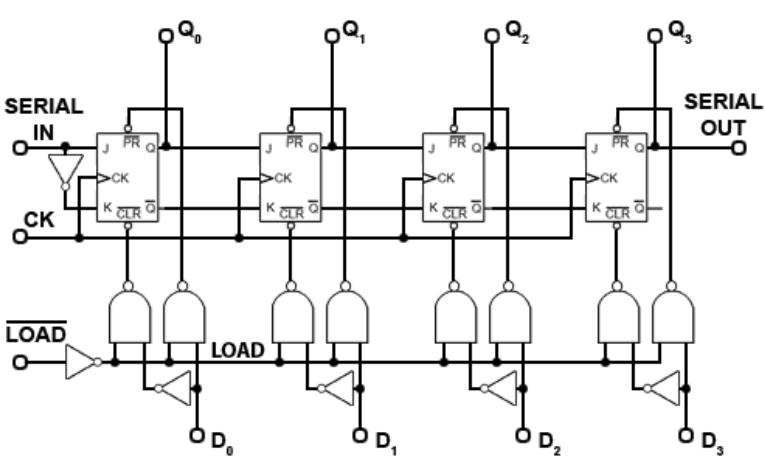
SIPO

The shift register is modified to include additional Q outputs from each flip-flop, so allowing the register to input serial data, and output it in both serial and parallel form. The register could therefore now be called both a ‘Serial In/Serial Out and Serial In/Parallel Out’ (SISO/SIPO) register. This format is the basis for converting serial data to parallel data.



PISO

If use is also made of the Q output, and the additional preset (PR) and clear (CLR) inputs available on many flip-flops, the shift register could be made more versatile still.

This shows a shift register modified to enable it to be loaded with a 4-bit parallel number, which may then be shifted right to appear at the serial output one bit at a time. As the ‘Parallel In/Serial Out’ or PISO register also has a serial input, it can also be used as a SISO register, and if extra outputs from each Q output were also included, the register would also have Serial In/Parallel Out (SIPO) operation.

Loading Parallel Data

If the LOAD input is taken to logic 0, the LOAD control line connected to the four pairs of NAND gates associated with the four flip-flops will be at logic 1, and all four pairs of NAND gates will be enabled. Therefore a logic 1 appearing on any of the D inputs will be inverted by the NOT gate connected to the D input, making the inputs to the left hand NAND gate of the relevant pair of gates, logic 1 and logic 0. This will cause logic 1 to be applied to the CLR input of the flip-flop.

The right hand NAND gate of the pair will have both inputs at logic 1, due to the logic 1 on LOAD line and logic 1 on the D input, and so will output logic 0 (NAND gate rules) to the PR input of the flip-flop, setting the Q output to logic 1.

If the D input is at logic 0, the left hand gate of the NAND gate pair will output logic 0 and the right hand NAND gate will output logic 1, causing the CLR input to clear the Q output of the relevant flip-flop to logic 0.

Notice that as JK flip-flops are being used in this design, a NOT gate is connected between J and K of the first flip-flop of the chain to make the JK flip-flop mimic a D Type. The remaining flip-flops of the shift register have J and K connected to the previous Q and Q outputs, so will also be at opposite logic states.

A 4-bit reversible shift register.

The shift register could be operated as:

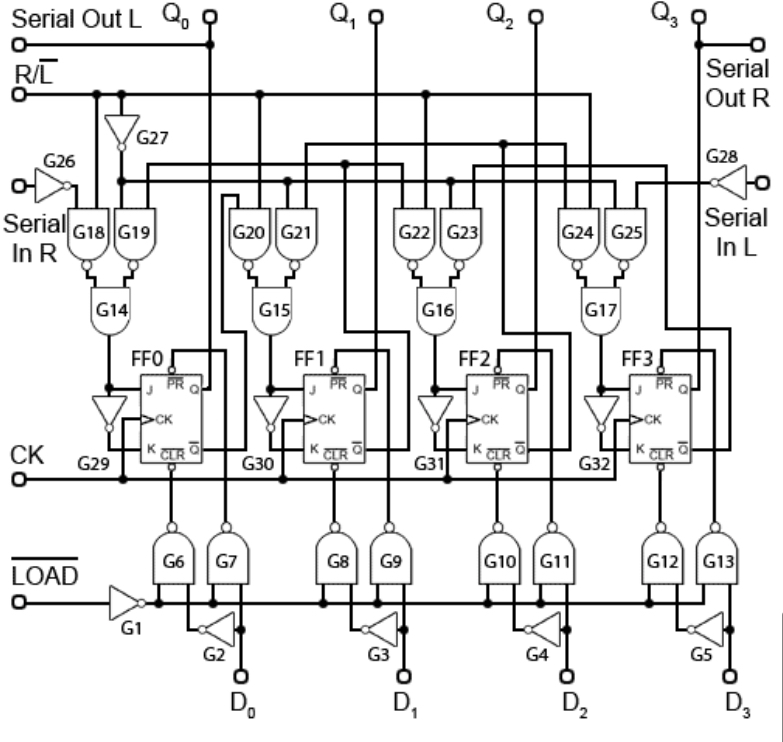
A parallel in/parallel out register. (PIPO)

A Serial in/serial out register. (SISO)

A serial in/parallel out register. (SIPO)

A parallel in/serial out register. (PISO)

However can only shift data in one direction, i.e. left to right. To be truly versatile it could be an advantage to be able to shift data in both directions and in any of the four shift register operating modes. achieves this by adding data steering circuitry.



The gating arrangement at the bottom (gates G1 to G13) is exactly the same as that described above , and these gates control the loading of parallel data.

Gates G14 to G28 control the direction of data flow through the register. The JK flip-flops use the inverter gates G29 to G32 to ensure that J and K are at opposite logic states, so the flip-flops are mimicking D Type operation, with J being used as the data input. Notice also that the clock is connected in the familiar synchronous mode.

Operation.

In any of the modes involving serial operation, data may be shifted left or shifted right by the application of a suitable logic level at the shift control (R/L) input.

With a logic 1 at this input the register is in the shift right mode, and data is taken into the ‘Serial in R’ input to be shifted right by application of successive clock pulses, appearing as parallel data, changing with each clock pulse, on the flip flop Q outputs. After four clock pulses the data begins to appear in serial form on the Q3 output, which is also the ‘Serial Out R’ output.

The logic 1 on the shift control (R/L) enables gates G18, 20, 22 & 24, but because the logic 1 is inverted by G27, gates G19, 21, 23 & 25 are disabled.

The path of serial data (e.g. a logic 1) from left to right is as follows; the logic 1 appearing at the input to G26 is inverted and passes through G18 which re-inverts it to logic 1 and, as G19 is disabled its output must also be at logic 1. Both inputs to the AND gate G14 are at logic 1 and therefore so is its output, (AND gate rules) making the J input of FF0 logic 1.

On the arrival of a clock pulse, the logic 1 input to FF0 will appear on the output Q0. Its inverse (logic 0) will also appear on the Q output of FF0. This logic 0 forms the input to the next multiplexer arrangement, gates G20, 21 & 15. As G20 is enabled (and G21 disabled) the logic 0 becomes logic 1 at G15 output and so is fed to the J input of FF1. This method is used to transfer data to each flip-flop in the chain.

To achieve shift left operation, the shift control (R/L) is set to logic 0 and so enables gates G19, 21, 23 & 25 while disabling gates G18, 20, 22 & 24. Therefore the Q output of FF3 is connected via G23 and G16 to the D input of FF2, the Q output of FF2 is connected to the J input of FF1 via G21 and G15 (remember that G24 is disabled, so FF3 is isolated from this path). Finally, the Q output of FF1 is connected via G19 and G14 to the J input of FF0, the Q0 output of which is also the ‘Serial Out L’ output. The ability to shift data in either direction, together with the parallel input and output facilities make this register a very versatile device.

It is common to connect shift register ICs in cascade, using the serial output of one register to connect to the serial input of the next register in the chain. For this reason both the data and clock inputs and outputs of register ICs are normally buffered.

Some examples from the many commercially available IC registers using these and similar methods, available in both CMOS and TTL versions, are listed below.

74HC164 8-Bit SIPO Shift register from NXP

74HC594 8-Bit SIPO/SISO with PIPO output storage register and dual clocks - from NXP.

74HC595 8-Bit SIPO/SISO with tri-state output PIPO storage register and dual clocks - from NXP.

HEF4014B PISO Register with 8-bit synchronous parallel LOAD and outputs from Q5, Q6 & Q7 only - from NXP.

CD4031B 64 Stage SISO shift register with re-circulation mode