**Chapter One**

**Introduction**

**Register**

Registers are logic units used for storing strings of bits in a sequential logic circuit. You will first need to understand the concept of [flip-flops](https://hkn.illinois.edu/wiki/wiki:logic_design:flip-flops) in order to understand how registers work. Registers are generally constructed using D flip-flops; therefore, the examples provided here will use these flip-flops.

The simplest register is a 1-bit register. A 1-bit register is simply a single D flip-flop. It holds a logical value of exactly one bit in length. Larger registers can hold longer strings of bits.

A register is a temporary storage area built into a [CPU](https://techterms.com/definition/cpu). Some registers are used internally and cannot be accessed outside the [processor](https://techterms.com/definition/processor), while others are user-accessible. Most modern CPU [architectures](https://techterms.com/definition/architecture) include both types of registers.

Internal registers include the instruction register (IR), memory buffer register (MBR), memory data register (MDR), and memory address register (MAR). The instruction register fetches instructions from the program counter (PC) and holds each instruction as it is executed by the processor. The memory registers are used to pass data from [memory](https://techterms.com/definition/memory) to the processor. The storage time of internal registers is extremely temporary, as they often hold data for less than a millisecond.

User-accessible registers are larger than internal registers and typically hold data for a longer time. For example, a data register may store individual values referenced being by a currently running program. An address register contains memory addresses, which reference different blocks of memory within the system [RAM](https://techterms.com/definition/ram). Many CPUs now have general purpose registers (GPRs), which may contain both data and memory addresses.

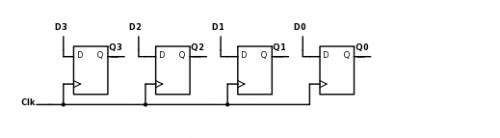
Registers vary in both number and size, depending on the CPU architecture. Some processors have 8 registers while others have 16, 32, or more. For many years, registers were 32-bit, but now many are 64-bit in size. A 64-bit register is necessary for a 64-bit processor, since it enables the CPU to access 64-bit memory addresses. A 64-bit register can also store 64-bit instructions, which cannot be loaded into a 32-bit register. Therefore, most [programs](https://techterms.com/definition/program) written for 32-bit processors can run on 64-bit computers, while 64-bit programs are not backwards compatible with 32-bit machines.

**Chapter Two**

**TYPES OF DIGITAL ELECTRONIC REGISTERS**

## Parallel-Load Registers

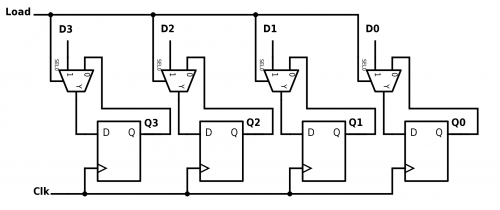
Parallel-load registers are a type of register where the individual bit values in the register are loaded simultaneously. More specifically, every flip-flop within the register takes an external data input, and these inputs are loaded into the flip-flops on the same edge in a clock cycle.



Pictured above is a simple 4-bit parallel-load register where D0, D1, D2, and D3 are the individual data bits; Q0, Q1, Q2, and Q3 form the output value (as a 4-bit word Q3Q2Q1Q0); and Clk is the single clock signal.

### Load Signal

Many commonly used parallel-load registers, however, also implement a **load** signal. A load signal is used to synchronize the loading of a register's flip-flops in cases where the individual data inputs are not provided simultaneously. When the load signal is set, all the flip-flops in the register are loaded with the provided data inputs during the next clock cycle. When the load signal is clear, all the flip-flops retain their current value. This functionality is often implemented using [multiplexers](https://hkn.illinois.edu/wiki/wiki:logic_design:muxes_demuxes_and_decoders), or muxes.



A schematic of a 4-bit parallel load register with a load signal is depicted above. D0, D1, D2, and D3 are the individual data bits. Q0, Q1, Q2, and Q3 form the output value (as a 4-bit word Q3Q2Q1Q0). Load is the single load signal. Clk is the single clock signal.

# Shift Register

A **shift register** is a type of [digital circuit](https://en.wikipedia.org/wiki/Digital_circuit) using a cascade of [flip flops](https://en.wikipedia.org/wiki/Flip-flop_(electronics)) where the output of one flip-flop is connected to the input of the next. They share a single [clock signal](https://en.wikipedia.org/wiki/Clock_signal), which causes the data stored in the system to shift from one location to the next. By connecting the last flip-flop back to the first, the data can cycle within the shifters for extended periods, and in this form they were used as a form of [computer memory](https://en.wikipedia.org/wiki/Computer_memory). In this role they are very similar to the earlier [delay line memory](https://en.wikipedia.org/wiki/Delay_line_memory) systems and were widely used in the late 1960s and early 1970s to replace that form of memory.

In most cases, several parallel shift registers would be used to build a larger memory pool known as a "[bit array](https://en.wikipedia.org/wiki/Bit_array)". Data was stored into the array and read back out in parallel, often as a [computer word](https://en.wikipedia.org/wiki/Computer_word), while each bit was stored serially in the shift registers. There is an inherent trade-off in the design of bit arrays; putting more flip-flops in a row allows a single shifter to store more bits, but requires more clock cycles to push the data through all of the shifters before the data can be read back out again.

Shift registers can have both [parallel](https://en.wikipedia.org/wiki/Parallel_communication) and [serial](https://en.wikipedia.org/wiki/Serial_communication) inputs and outputs. These are often configured as "serial-in, parallel-out" (SIPO) or as "parallel-in, serial-out" (PISO). There are also types that have both serial and parallel input and types with serial and parallel output. There are also "bidirectional" shift registers which allow shifting in both directions: L→R or R→L. The serial input and last output of a shift register can also be connected to create a "circular shift register". A PIPO register (parallel in, parallel out) is very fast – an output is given within a single clock pulse.

**Types of shift registers**

**Serial In Serial Out (SISO) shift registers**

**Serial In Serial Out (SISO) shift registers** are a kind of shift registers where both data loading as well as data retrieval to/from the [shift register](https://www.electrical4u.com/shift-registers/) occurs in serial-mode. Here the data word which is to be stored is fed bit-by-bit at the input of the first [flip-flop](https://www.electrical4u.com/latches-and-flip-flops/). Further it is seen that the inputs of all other flip-flops (except the first flip-flop FF1) are driven by the outputs of the preceding ones say for example, the input of FF2 is driven by the output of FF1. At last the data stored within the register is obtained at the output pin of the nth flip-flop in serial-fashion.

Initially all the flip-flops in the register are cleared by applying high on their clear pins. Next the input data word is fed serially to FF1.  
This causes the bit appearing at the D1 pin (B1) to be stored into FF1 as soon as the first leading edge of the clock appears. Further at the second clock tick, B1 gets stored into FF2 while a new bit enters into FF1 (B2).

This kind of shift in data bits continues for every rising edge of the clock pulse. This indicates that for every single clock pulse the data within the register moves towards right by a single bit. Thus the design shown in Figure 1 is regarded as a right-shift **SISO shift register**. Following the data transmission as explained, one can note that the first bit of an input word appears at the output of nth flip-flop for the nth clock tick. On applying further clock cycles, one gets the next successive bits of the input data word as the serial output.

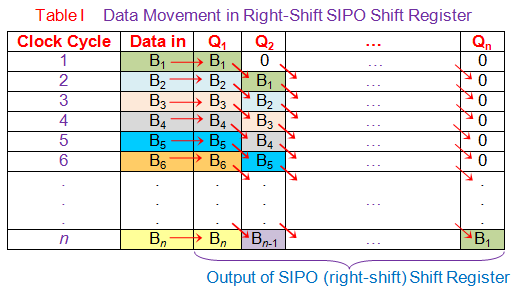
**Serial In Parallel Out (SIPO) shift registers**

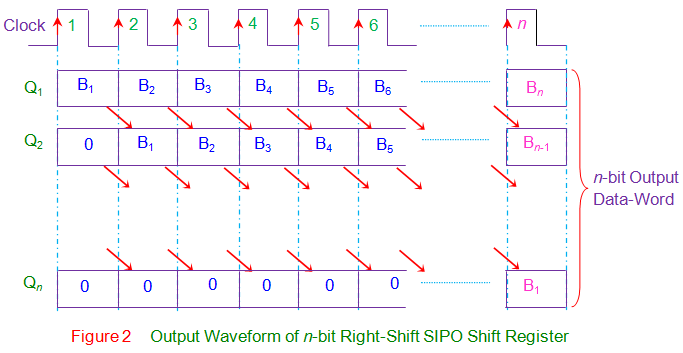
In **Serial In Parallel Out (SIPO) shift registers**, the data is stored into the register serially while it is retrieved from it in parallel-fashion. Here the data word which is to be stored (Data in) is fed serially at the input of the first [flip-flop](https://www.electrical4u.com/latches-and-flip-flops/) (D1 of FF1). It is also seen that the inputs of all other flip-flops (except the first flip-flop FF1) are driven by the outputs of the preceding ones say for example, the input of FF2 is driven by the output of FF1. In this kind of [shift register](https://www.electrical4u.com/shift-registers/), the data stored within the register is obtained as a parallel-output data word (Data out) at the individual output pins of the flip-flops (Q1 to Qn).

In general, the register contents are cleared by applying high on the clear pins of all the flip-flops at the initial stage. After this, the first bit, B1 of the input data word is fed at the D1 pin of FF1.

This bit (B1) will enter into FF1, get stored and thereby appears at its output Q1 on the appearance of first leading edge of the clock. Further at the second clock tick, the bit B1 right-shifts and gets stored into FF2 while appearing at its output pin Q2 while a new bit, B2 enters into FF1. Similarly at each clock tick the data within the register moves towards right by a single bit while a new bit of the input word enters into the register. Meanwhile one can extract the bits stored within the register in parallel-fashion at the individual flip-flop outputs.

Analyzing on the same grounds, one can note that the n-bit input data word is obtained as an n-bit output data word from the shift register at the rising edge of the nth clock pulse. This working of the shift-register can be summarized as in Table I and the corresponding wave forms are given by Figure 2.

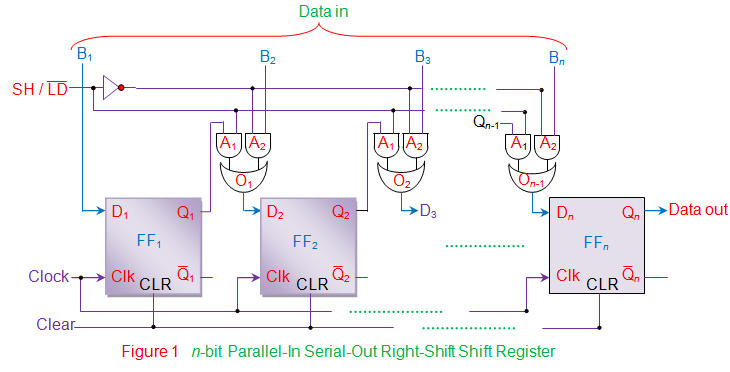




In the right-shift SIPO shift-register, data bits shift from left to right for each clock tick. However if the data bits are made to shift from right to left in the same design, one gets a left-shift SIPO shift-register as shown by Figure 3. Nevertheless the basic working principle remains the same except the fact that now Bn down to B1 is stored in Qn down to Q1 i.e. Q1 = B1, Q2 = B2 … Qn = Bn at the nth clock tick.

**Parallel In Serial Out (PISO) shift registers**

In **Parallel In Serial Out (PISO) shift registers**, the data is loaded onto the register in parallel format while it is retrieved from it serially. Figure 1 shows a **PISO shift register** which has a control-line https://www.electrical4u.com/images/2017/december17/1513684463.PNGand combinational circuit (AND and [OR gates](https://www.electrical4u.com/logical-or-gate/)) in addition to the basic register components ([flip-flops](https://www.electrical4u.com/latches-and-flip-flops/)) fed with clock and clear pins.

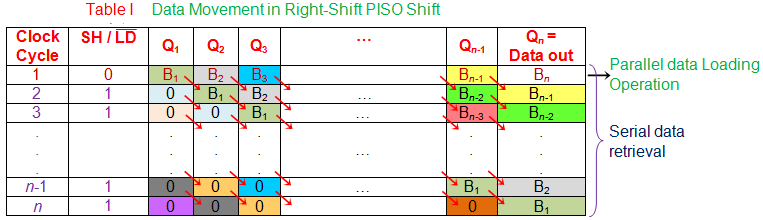


Here https://www.electrical4u.com/images/2017/december17/1513684279.PNGcontrol line is used to select the functionality of the shift register amongst shift or load at a given instant of time. This is because when the https://www.electrical4u.com/images/2017/december17/1513684279.PNGline is made low, A2 [AND gates](https://www.electrical4u.com/logical-and-gate/) of all the combinational circuits become active while A1 gates become inactive.

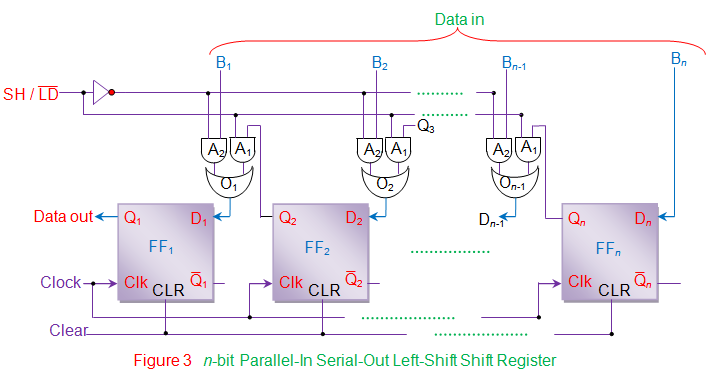
Thus the bits of the input data word (Data in) appearing as inputs to the gates A2 are passed on as the outputs of [OR gates](https://www.electrical4u.com/logical-or-gate/) at each individual combinational circuit. This causes the individual bits of the Data in to be loaded/stored into respective flip-flops at the appearance of first leading edge of the clock (except the bit B1 which gets directly stored into FF1 at the first clock tick). This indicates that all the bits of the input data word are stored into the register components at the same clock tick.

Next, https://www.electrical4u.com/images/2017/december17/1513684279.PNGline is driven high to activate the gates A1 of the combinational circuits which inturn disables the gates A2. This causes output bit of each [flip-flop](https://www.electrical4u.com/latches-and-flip-flops/) to appear at the output of the OR gate driving the very-next flip-flop (except the last flip-flop FFn) i.e. output bit of FF1 (Q1) appears as the output of [OR gate](https://www.electrical4u.com/logical-or-gate/) 1 (O1) connected to D2; Q2 = output of O2 = D3 and so on. At this stage, if the rising edge of the clock pulse appears, then Q1 appears at Q2, Q2 appears at Q3, … and Qn-1 appears at Qn.

This is nothing but right-shift of the data stored within the register by one-bit. Similarly it is seen that for each of the further clock pulses applied, one bit exits the PISO shift register through the output pin of nth flip-flop (Data out = Qn of FFn), which is nothing but the serial output. Thus one requires n clock cycles to obtain the entire n-bit input data word as a serial output of PISO shift register.  
The truth table of the **PISO shift register** emphasizing the loading and retrieval processes is shown by Table I, while the corresponding wave forms are shown by Figure 2.

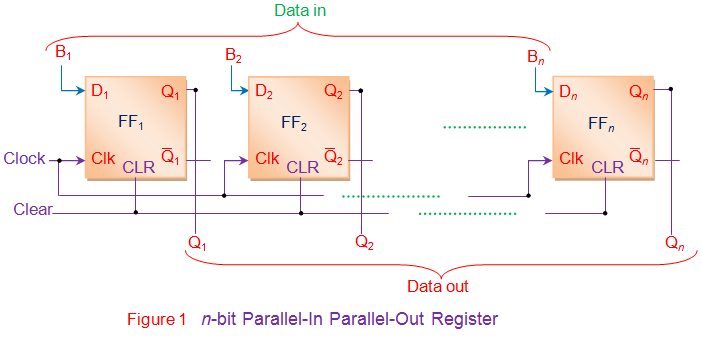


By slightly modifying the design of Figure 1, one can make the data bits within the register to shift from right to left, thus obtaining a left-shift PISO shift-register (Figure 3). However the basic working principle remains unaltered.

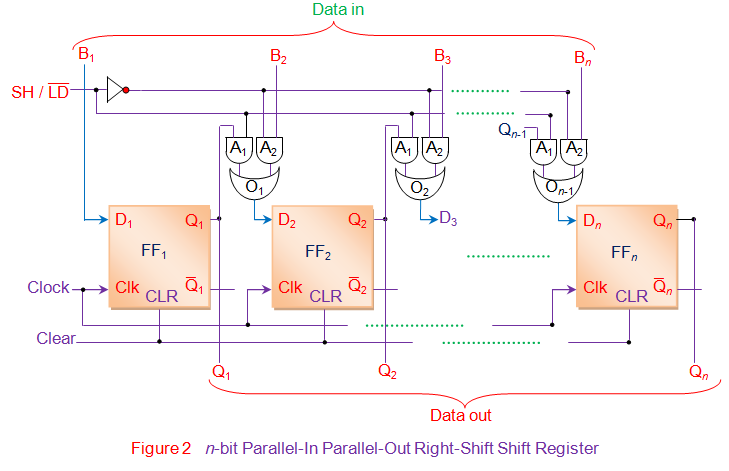


**Parallel In Parallel Out (PIPO) shift registers**

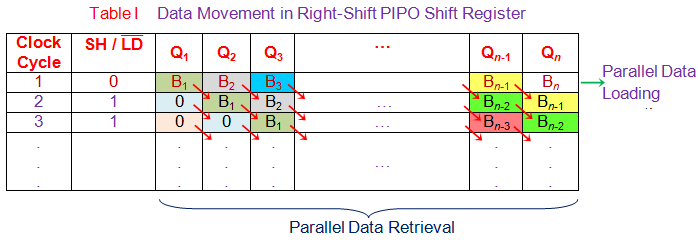
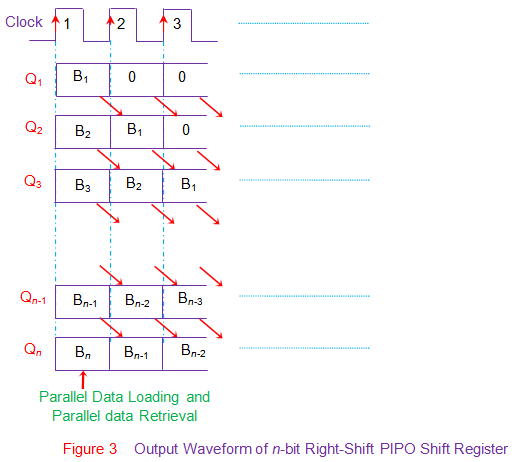
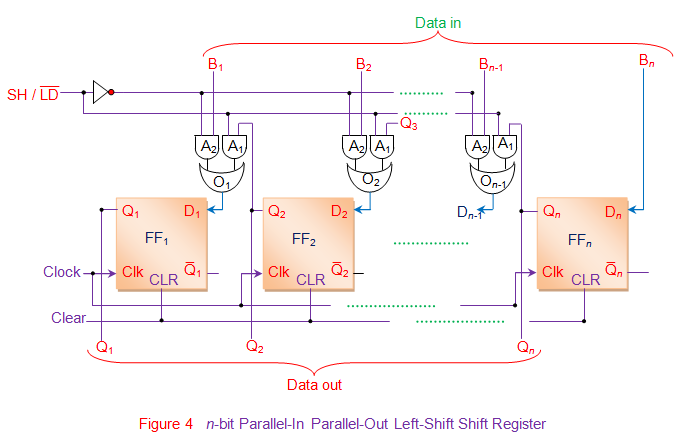
**Parallel In Parallel Out (PIPO) shift registers** are the type of storage devices in which both data loading as well as data retrieval processes occur in parallel mode. Figure 1 shows a PIPO register capable of storing n-bit input data word (Data in). Here each flip-flop stores an individual bit of the data in appearing as its input (FF1 stores B1 appearing at D1; FF2 stores B2 appearing at D2 … FFn stores Bn appearing at Dn) at the instant of first clock pulse. Further, at the same instant, the bit stored in each individual [flip-flop](https://www.electrical4u.com/latches-and-flip-flops/) also appears at their respective output pins (Q1 = D1; Q2 = D2 … Qn = Bn). This indicates that both data storage as well as data recovery occur at a single (and at the same) clock pulse in PIPO registers.



However one has to note that the PIPO register shown in Figure 1 is not capable of shifting the data bits. In order to convert PIPO register of Figure 1 into PIPO shift register, one has to modify its circuit by adding combinational circuit and control line https://www.electrical4u.com/images/2017/december17/1513684279.PNGas shown by Figure 2.



Here if https://www.electrical4u.com/images/2017/december17/1513684279.PNGline goes low, A2 AND gates of all the combinational circuits become active while A1 gates become inactive.  
Thus the bits of the input data word (Data in) appearing as inputs to the gates A2 are passed on as the [OR gate](https://www.electrical4u.com/logical-or-gate/) outputs which are further loaded/stored into respective flip-flops at the appearance of first leading edge of the clock (except the bit B1 which gets directly stored into FF1 at the first clock tick). This indicates that all the bits of the input data word are stored into the register components at the same clock tick. At the same time, these bits also appear at the output pins of the respective flip-flops thus yielding parallel-output data word at the same clock tick.

Further when https://www.electrical4u.com/images/2017/december17/1513684279.PNGline is made high, A1 gates of all the combinational circuits enable while A2 gates get disabled. This causes the output bit of each flip-flop to appear at the output of the OR gate driving the very-next flip-flop (except the last flip-flop FFn) i.e. output bit of FF1 (Q1) appears as the output of [OR gate](https://www.electrical4u.com/logical-or-gate/) 1 (O1) connected to D2; Q2 = output of O2 = D3 and so on. At this stage, if the rising edge of the clock pulse appears, then Q1 appears at Q2, Q2 appears at Q3, … and Qn-1 appears at Qn. This is nothing but right-shift of the data stored within the register by one-bit. This working is further emphasized in the Table I and Figure 3.  
  
  
Similar to the right-shift PIPO shift register, there can also be a left-shift PIPO shift register as shown by Figure 4. Nevertheless the mode of working remains the same.

**Chapter Three**

**Master Slave JK flip flop**   
The Master-Slave Flip-Flop is basically a combination of two JK flip-flops connected together in a series configuration. Out of these, one acts as the **“master”** and the other as a **“slave”**. The output from the master flip flop is connected to the two inputs of the slave flip flop whose output is fed back to inputs of the master flip flop.

In addition to these two flip-flops, the circuit also includes an **inverter**. The inverter is connected to clock pulse in such a way that the inverted clock pulse is given to the slave flip-flop. In other words if CP=0 for a master flip-flop, then CP=1 for a slave flip-flop and if CP=1 for master flip flop then it becomes 0 for slave flip flop.

**Working of a master slave flip flop**

1. When the clock pulse goes to 1, the slave is isolated; J and K inputs may affect the state of the system. The slave flip-flop is isolated until the CP goes to 0. When the CP goes back to 0, information is passed from the master flip-flop to the slave and output is obtained.
2. Firstly the master flip flop is positive level triggered and the slave flip flop is negative level triggered, so the master responds before the slave.
3. If J=0 and K=1, the high Q’ output of the master goes to the K input of the slave and the clock forces the slave to reset, thus the slave copies the master.
4. If J=1 and K=0, the high Q output of the master goes to the J input of the slave and the Negative transition of the clock sets the slave, copying the master.
5. If J=1 and K=1, it toggles on the positive transition of the clock and thus the slave toggles on the negative transition of the clock.
6. If J=0 and K=0, the flip flop is disabled and Q remains unchanged.

**Pseudo Random Generators**

The pseudo random generators are widely used in modern science and technology [1], [2], [3], [4]. One of the pseudo random signal generators class is a linear feedback shift register generators – LFSR [5], [6]. The LFSR generators are easy in hardware and software implementation, and they generate one pseudo random bit on each clock cycle [6]. In many applications the pseudo random signal rate, generated by the shift register generator, is too low [9]. In case when the statistical test results [10] are not a primary requirement, but the pseudo random signal rate is essential, the proposed additional output blocks might be used with the LFSR generator.

Basic pseudo random generator types include the congruential generators [1], its modifications and the generators that uses shift registers [3], [5]. The LFSR generators and majority of the complex pseudo random signal generators build on a shift registers [7], [8], generate only one bit of the pseudo random sequence for each clock cycle [5].

**Applications of Shift registers**

Shift registers are used in a lot of applications some of which are;

**1. Parallel to serial conversion**, where they are used to reduce the number of wires, or lines needed for communication between two devices, since serial communication generally require just two wires compared to parallel which depends on the number of bits being sent.

**2. IO expansion for microcontrollers**. In modern day electronics, microcontrollers IO pins are referred to as real estates and one needs as much as possible for certain application like turning on 100 leds or reading 100 reed switches with something like an Arduino or the Atmeg328p microcontroller. For example, the circuit diagram below illustrates how a serial to parallel shift register can be used to control 8 LEDs, using just three of the microcontrollers IO pins.

3. They are used in state registers which are used in **sequential devices**. Like a finite memory machine, the next state of the device is always determined by shifting and inserting a new data into the previous position.

4. **Time delays**. Shift registers are used for time delay in devices, with the time being adjusted by the clock, or increased by cascading shift registers or reduced by taking the output from a lower significant bit.

The time delay is usually calculated using the formula;

**t = N \* (1 / fc)**

N is the number of flip flop stage at which the output is taken, Fc is the frequency of the clock signal and t which is the value being determined is the amount of time for which the output will be delayed.

**Some of the most popular shift registers are**;

74HC 194 4-bit bidirectional universal shift register

74HC 198 8-bit bidirectional universal shift register

74HC595 Serial-In-Parallel-Out shift register

74HC165 Parallel-In-Serial-Out shift register

IC 74291 4-bit universal shift register, binary up/down counter, synchronous.

IC 74395 4-bit universal shift register with three-state outputs.

IC 74498 8-bit bidirectional shift register with parallel inputs and three-state outputs.

IC 74671 4-bit bidirectional shift register.