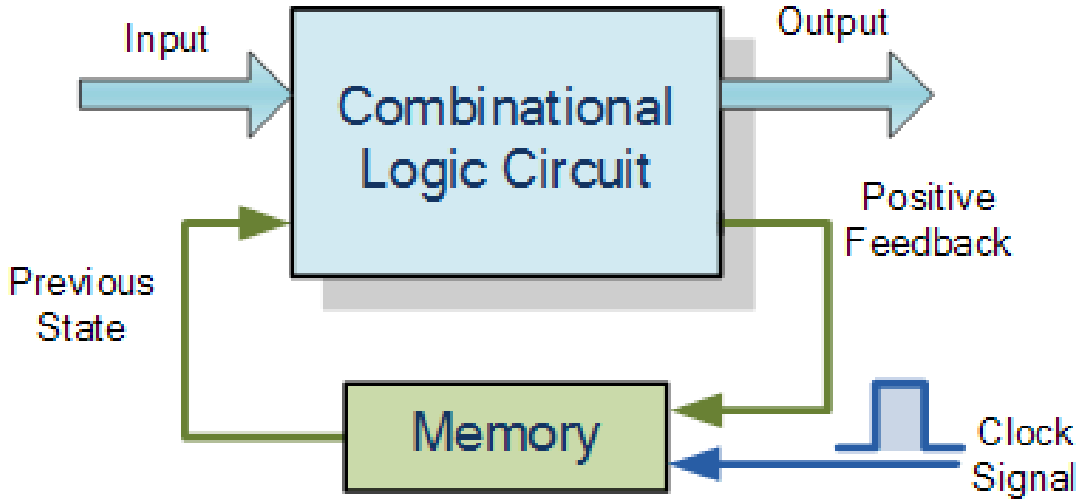
Unit-1 Sequential Circuits

- **Sequential Logic** circuits have some form of inherent “Memory” built in.
- sequential logic circuits are able to take into account their previous input state as well as those actually present.
- In other words, the output state of a “sequential logic circuit” is a function of the following three states, the “present input”, the “past input” and/or the “past output”.
- *Sequential Logic circuits* remember these conditions and stay fixed in their current state until the next clock signal changes one of the states, giving sequential logic circuits “Memory”.

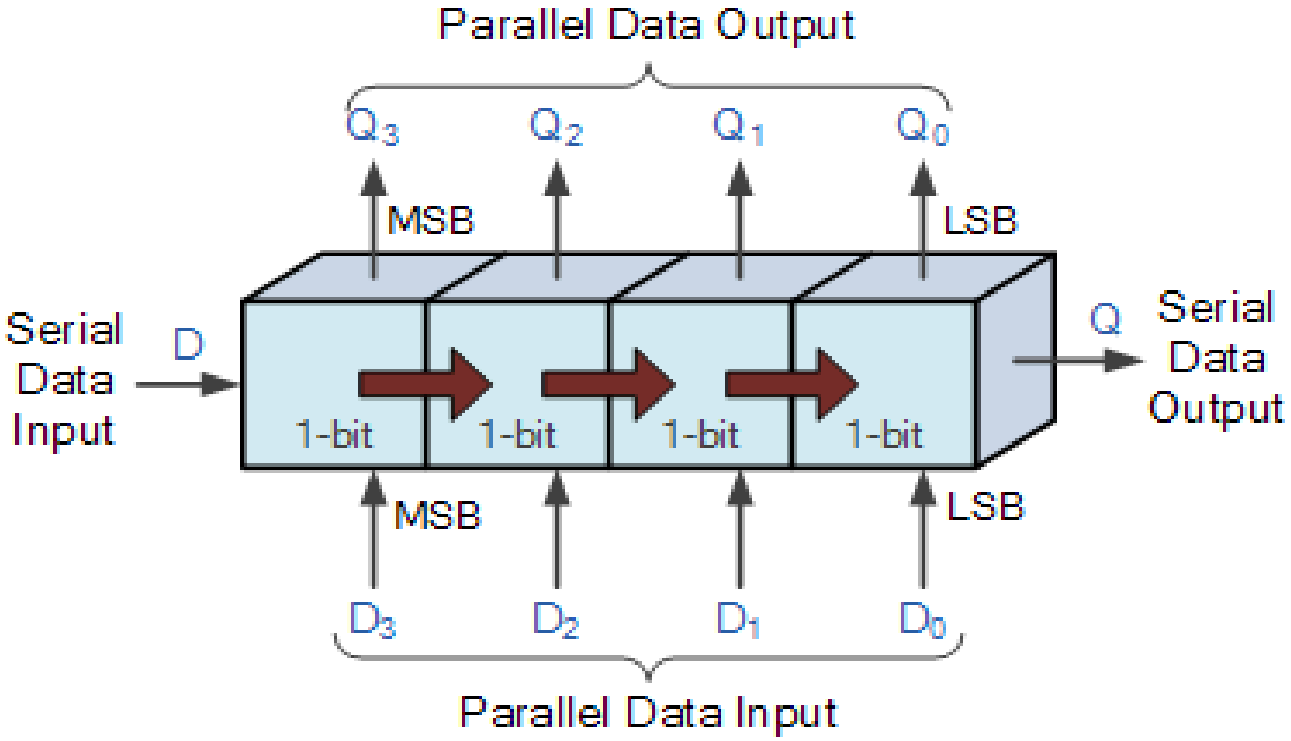


- Sequential circuits (together with combinatorial circuits) make it possible to build several useful applications, such as counters, registers, arithmetic/logic unit (ALU), all the way to microprocessors.
- **Flip Flops:**
- Memory element used in sequential circuits.
- Binary cells capable of storing one bit of information.
- A flip flop has two outputs, one is the normal value and other is the complement.
- It is also named as latch.

- NAND Gate Latch
- NOR Gate Latch
- RS Flip flop
- D Flip flop
- JK Flip flop

Shift Register and Types

- This sequential device loads the data present on its inputs and then moves or “shifts” it to its output once every clock cycle, hence the name **Shift Register**.



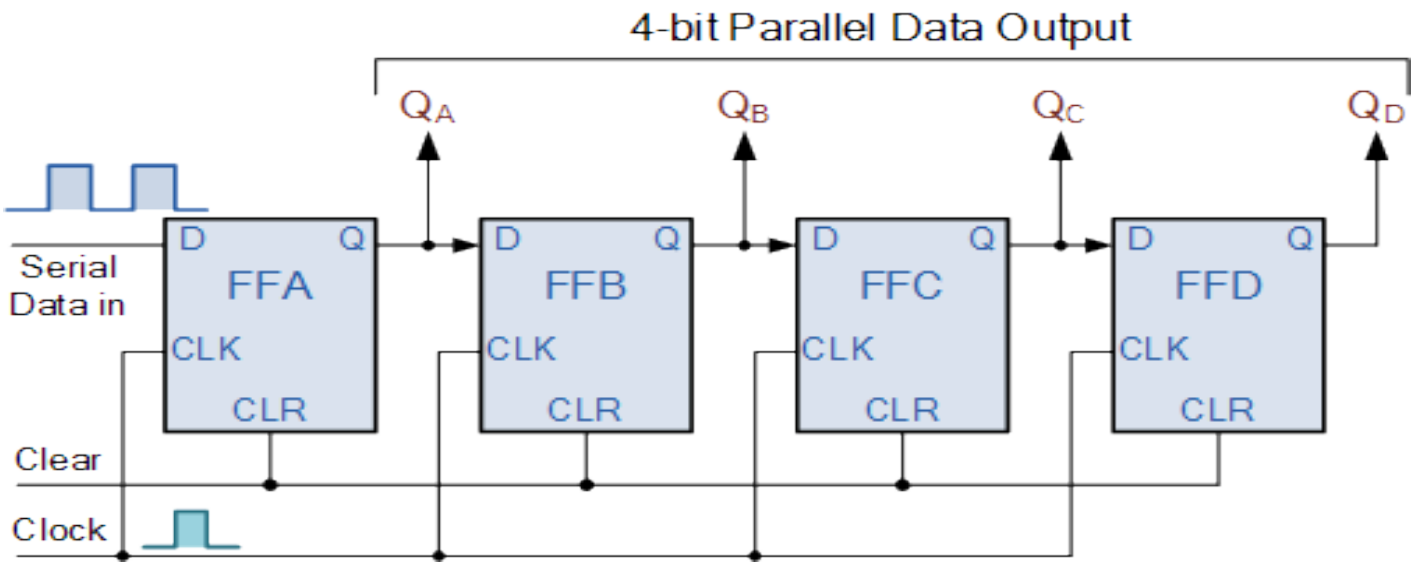
Types

- **Serial-in to Parallel-out (SIPO**) - the register is loaded with serial data, one bit at a time, with the stored data being available at the output in parallel form.
- **Serial-in to Serial-out (SISO)** - the data is shifted serially “IN” and “OUT” of the register, one bit at a time in either a left or right direction under clock control.
- **Parallel-in to Serial-out (PISO)** - the parallel data is loaded into the register simultaneously and is shifted out of the register serially one bit at a time under clock control.
- **Parallel-in to Parallel-out (PIPO)** - the parallel data is loaded simultaneously into the register, and transferred together to their respective outputs by the same clock pulse.

Features of shift Registers

- A simple **Shift Register** can be made using only D-type flip-Flops, one flip-Flop for each data bit.
- The output from each flip-Flop is connected to the D input of the flip-flop at its right.
- Shift registers hold the data in their memory which is moved or “shifted” to their required positions on each clock pulse.
- Each clock pulse shifts the contents of the register one bit position to either the left or the right.
- The data bits can be loaded one bit at a time in a series input (SI) configuration or be loaded simultaneously in a parallel configuration (PI).
- Data may be removed from the register one bit at a time for a series output (SO) or removed all at the same time from a parallel output (PO).
- One application of shift registers is in the conversion of data between serial and parallel, or parallel to serial.
- Shift registers are identified individually as SIPO, SISO, PISO, PIPO, or as a Universal Shift Register with all the functions combined within a single device.

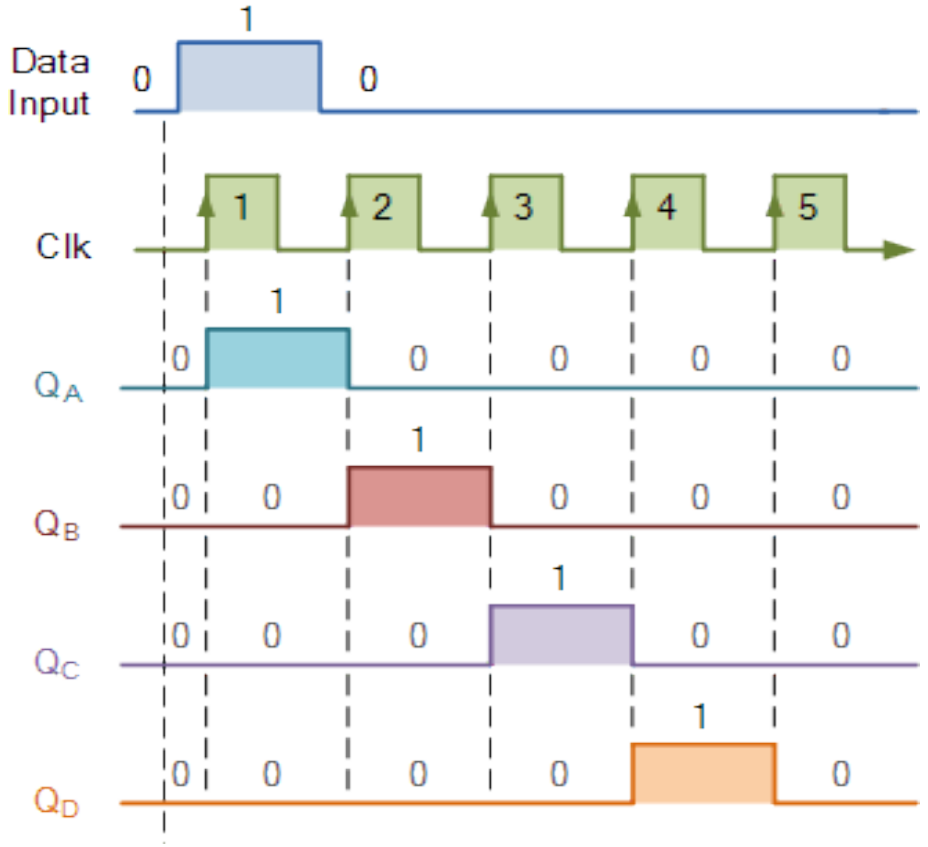
4-bit Serial-in to Parallel-out Shift Register



- The operation is as follows. Lets assume that all the flip-flops (FFA to FFD) have just been RESET (CLEAR input) and that all the outputs Q_A to Q_D are at logic level “0” ie, no parallel data output.

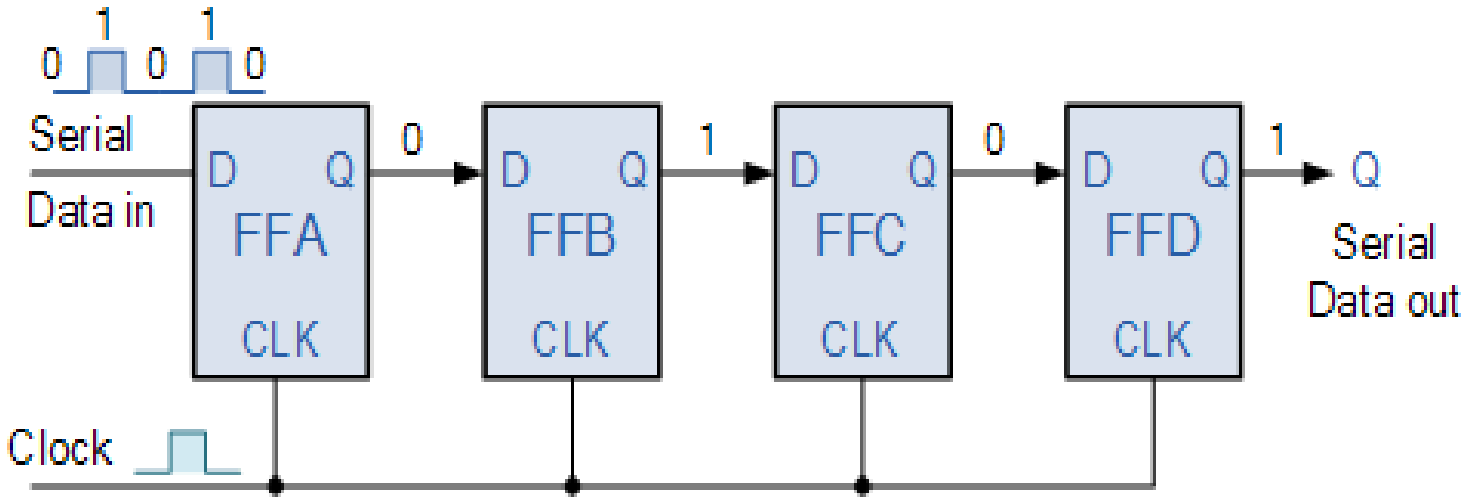
Basic Data Movement Through A Shift Register

Clock Pulse No	QA	QB	QC	QD
0	0	0	0	0
1	1	0	0	0
2	0	1	0	0
3	0	0	1	0
4	0	0	0	1
5	0	0	0	0



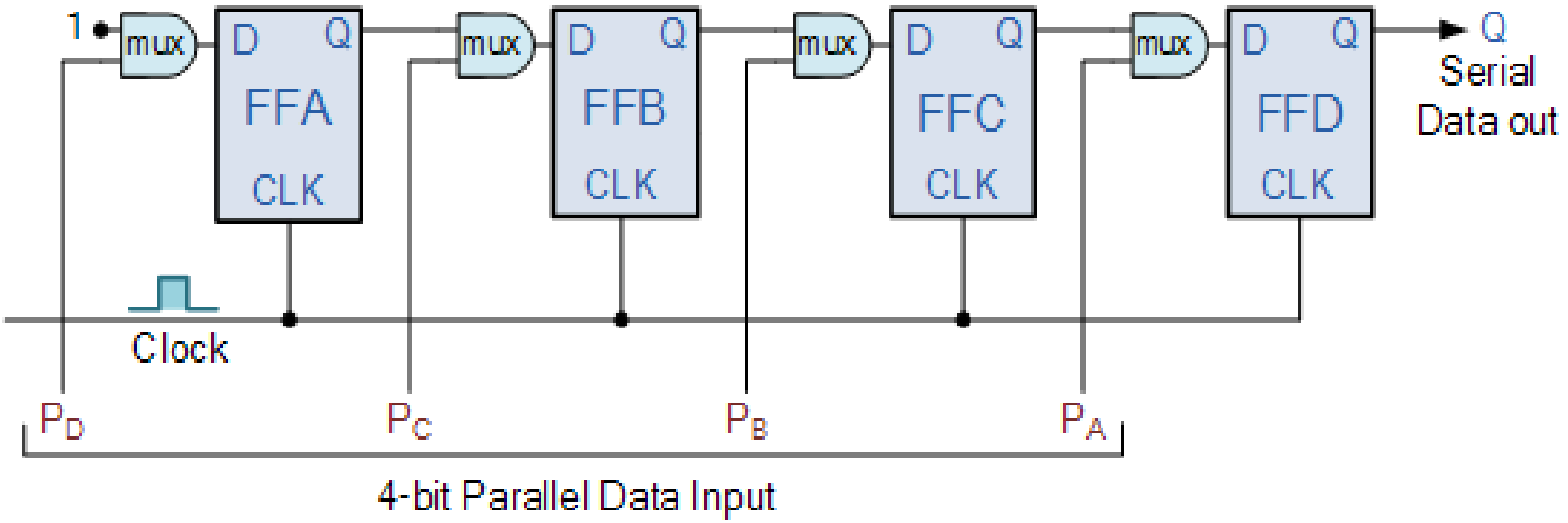
4 bit Serial-in to Serial-out (SISO) Shift Register

- This **shift register** is very similar to the SIPO above, except were before the data was read directly in a parallel form from the outputs Q_A to Q_D, this time the data is allowed to flow straight through the register and out of the other end. Since there is only one output, the DATA leaves the shift register one bit at a time in a serial pattern, hence the name **Serial-in to Serial-Out Shift Register** or **SISO**.



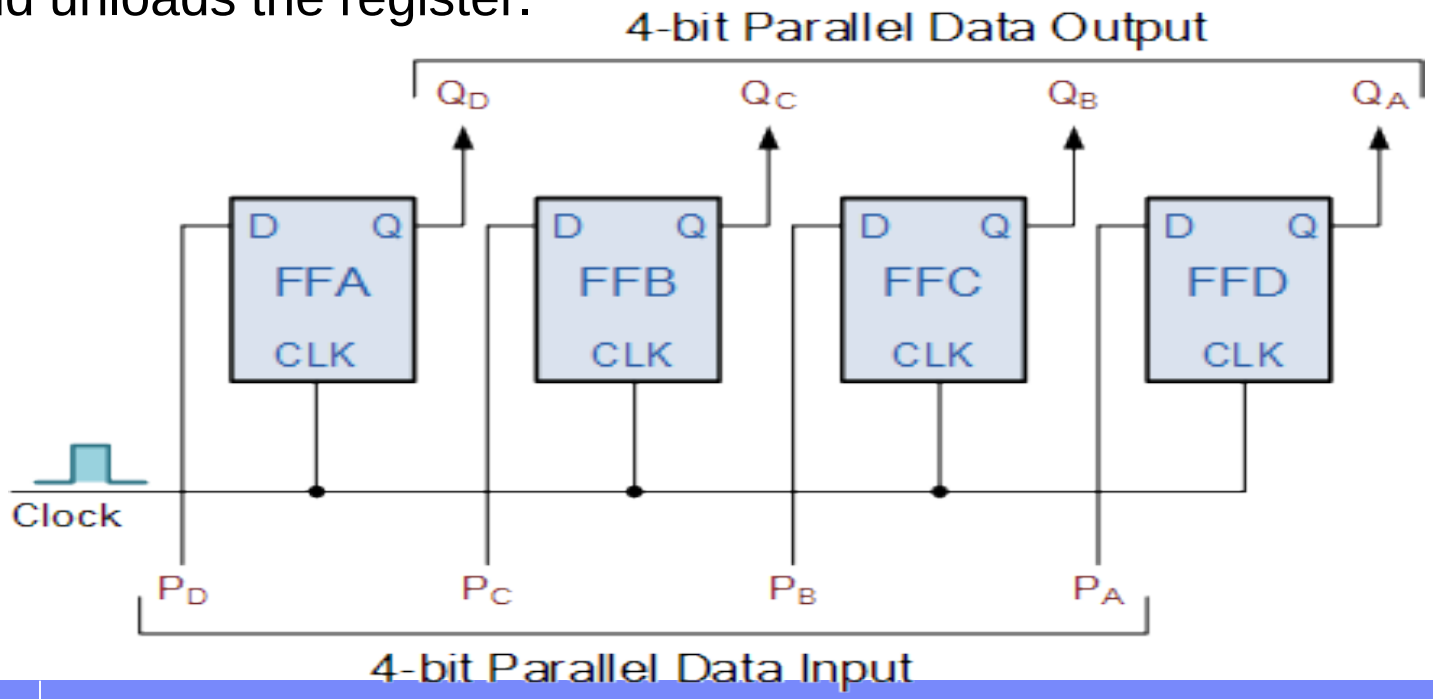
4-bit Parallel-in to Serial-out Shift Register

- The Parallel-in to Serial-out shift register acts in the opposite way to the serial-in to parallel-out one above. The data is loaded into the register in a parallel format in which all the data bits enter their inputs simultaneously, to the parallel input pins P_A to P_D of the register. The data is then read out sequentially in the normal shift-right mode from the register at Q representing the data present at P_A to P_D.



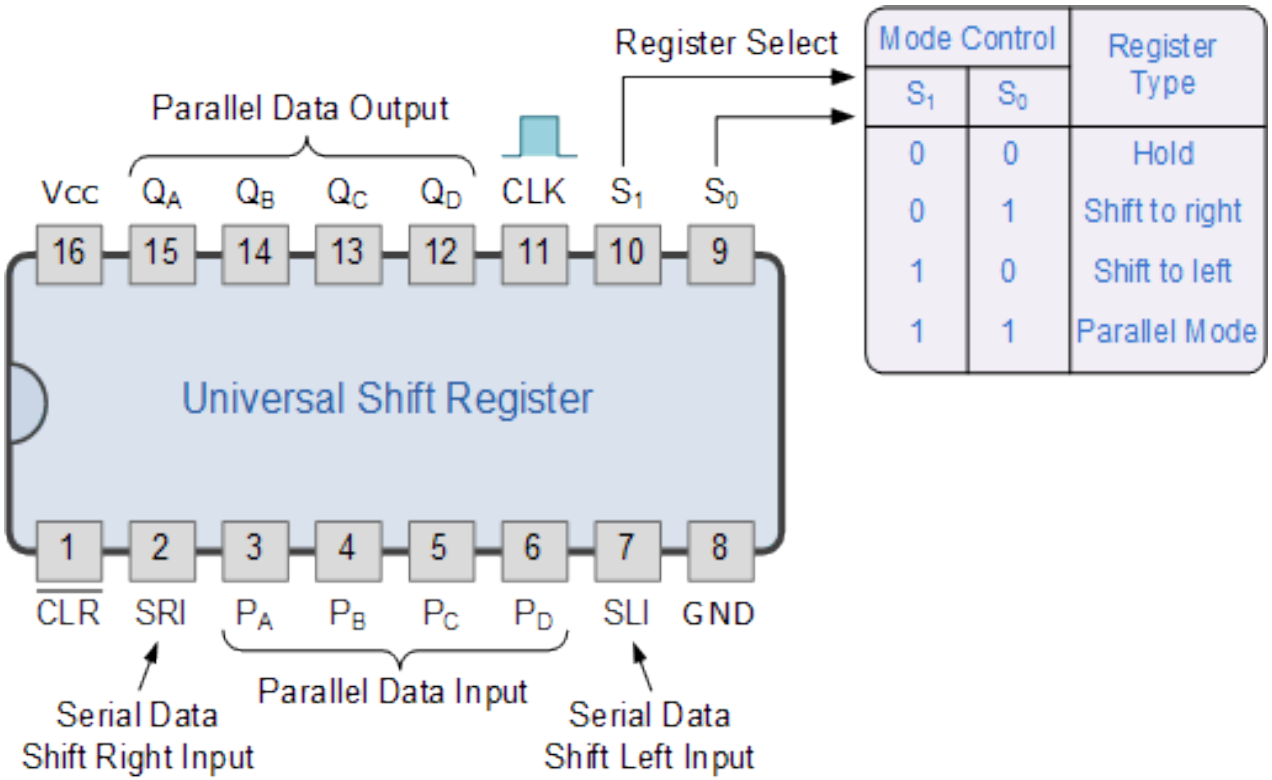
4-bit Parallel-in to Parallel-out (PIPO) Shift Register

- The final mode of operation is the Parallel-in to Parallel-out Shift Register. This type of shift register also acts as a temporary storage device or as a time delay device similar to the SISO configuration above.
- The data is presented in a parallel format to the parallel input pins P_A to P_D and then transferred together directly to their respective output pins Q_A to Q_D by the same clock pulse. Then one clock pulse loads and unloads the register.



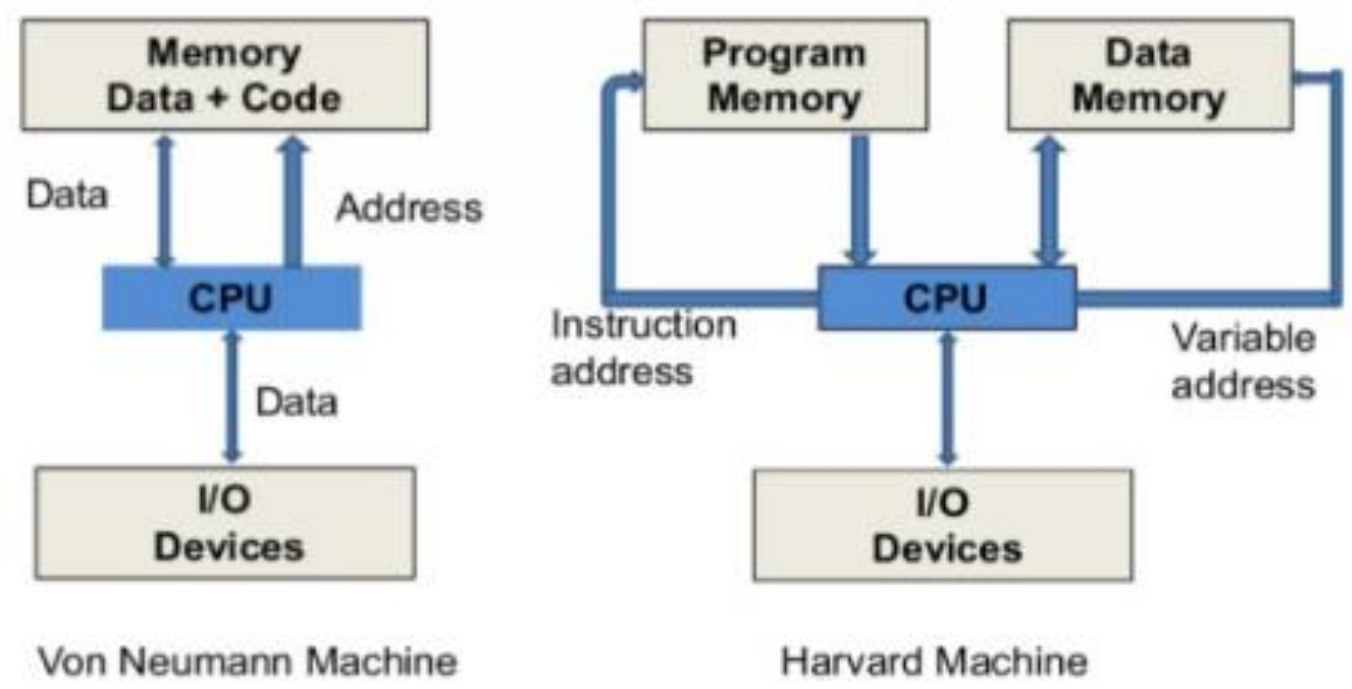
Universal Shift Register

- Universal shift registers can perform any combination of parallel and serial input to output operations but require additional inputs to specify desired function and to pre-load and reset the device

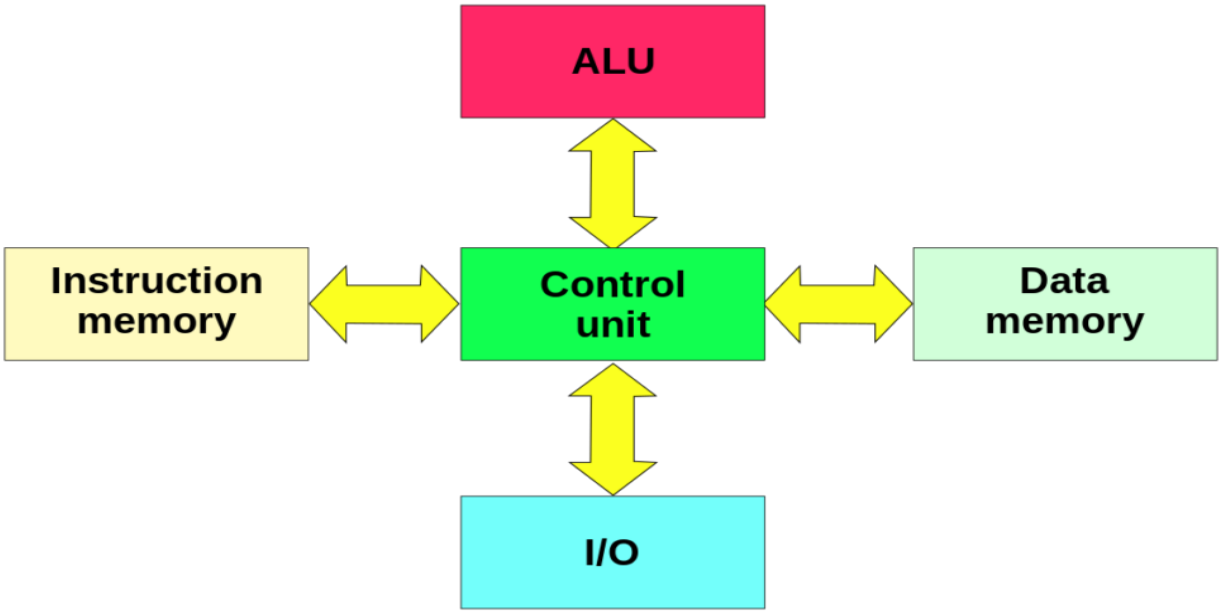


Mode Control		Register Type
S_1	S_0	
0	0	Hold
0	1	Shift to right
1	0	Shift to left
1	1	Parallel Mode

Von Neumann and Harvard Architecture

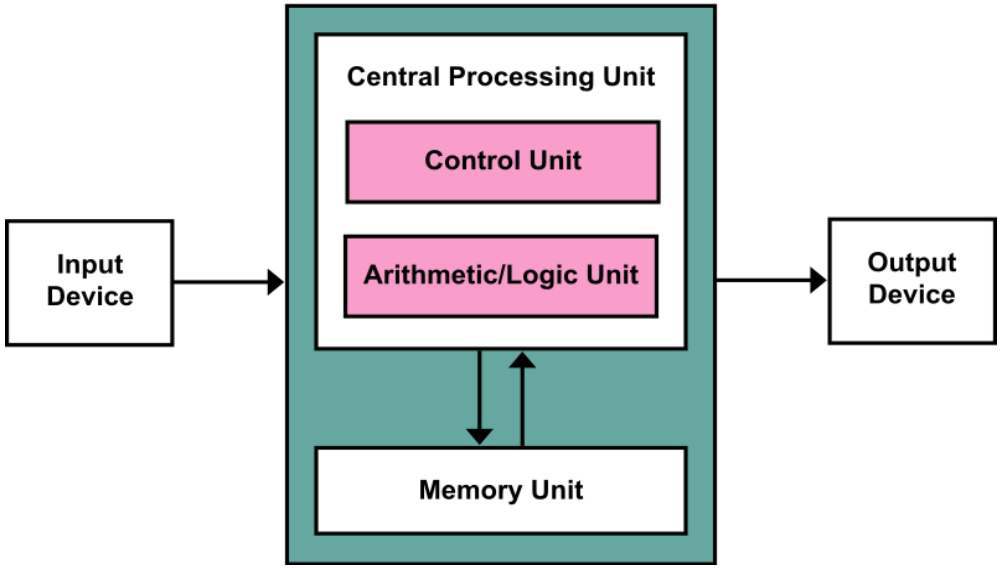


- In a Von-Neumann architecture, the same memory and bus are used to store both data and instructions that run the program. Since you cannot access program memory and data memory simultaneously, the Von Neumann architecture is susceptible to bottlenecks and system performance is affected.



Harvard Architecture

The Harvard architecture stores machine instructions and data in separate memory units that are connected by different busses. In this case, there are at least two memory address spaces to work with, so there is a memory register for machine instructions and another memory register for data. Computers designed with the Harvard architecture are able to run a program and access data independently, and therefore simultaneously. Harvard architecture has a strict separation between data and code.



VON NEUMANN ARCHITECTURE	HARVARD ARCHITECTURE
It is ancient computer architecture based on stored program computer	It is modern computer architecture based on Harvard Mark I relay based

hicles >

Same physical memory address is used for instructions and data.	Separate physical memory address is used for instructions and data.
There is common bus for data and instruction transfer.	Separate buses are used for transferring data and instruction.
Two clock cycles are required to execute single instruction.	An instruction is executed in a single cycle.
It is cheaper in cost.	It is costly than van neumann architecture.
CPU can not access instructions and read/write at the same time.	CPU can access instructions and read/write at the same time.
It is used in personal computers and small computers.	It is used in micro controllers and signal processing.

- **Reduced Set Instruction Set Architecture (RISC) –**
  The main idea behind is to make hardware simpler by using an instruction set composed of a few basic steps for loading, evaluating and storing operations just like a load command will load data, store command will store the data.
- **Complex Instruction Set Architecture (CISC) –**
  The main idea is that a single instruction will do all loading, evaluating and storing operations just like a multiplication command will do stuff like loading data, evaluating and storing it, hence it’s complex.

Characteristic of RISC –

- Simpler instruction, hence simple instruction decoding.
- Instruction come under size of one word.
- Instruction take single clock cycle to get executed.
- More number of general purpose register.
- Simple Addressing Modes.
- Less Data types.
- Pipeline can be achieved.

Characteristic of CISC –

- Complex instruction, hence complex instruction decoding.
- Instruction are larger than one word size.
- Instruction may take more than single clock cycle to get executed.
- Less number of general purpose register as operation get performed in memory itself.
- Complex Addressing Modes.
- More Data types.

- To improve the performance of a CPU we have two options:
  1) Improve the hardware by introducing faster circuits.
  2) Arrange the hardware such that more than one operation can be performed at the same time.
- Since, there is a limit on the speed of hardware and the cost of faster circuits is quite high, we have to adopt the 2nd option.
- **Pipelining :** Pipelining is a process of arrangement of hardware elements of the CPU such that its overall performance is increased. Simultaneous execution of more than one instruction takes place in a pipelined processor.

Difference Between RISC and CISC

RISC	CISC
Reduced instruction set computer	Complex instruction set computer.
The microprocessor is designed using hardwired control.	The microprocessor is designed using code control
It executes at least one instruction in a cycle	Several cycles may be required to execute one instruction
The instructions have simple fixed formats with few addressing modes.	The instructions have variable formats with several complex addressing modes.
It has several general purpose resisters and large cache memory.	It has a small number of general purpose registers
The instruction set of RISC micro- processors typically includes only register to register load and store.	The instruction set of CISC microprocessor includes several instructions to access memory and CPU registers.
Pipelining and superscalar architectures are the base methods to design a RISC processor	Pipelining and superscalar features are not the bases to design such processors, although, many CISC microprocessors use several features if RISCs such as: pipelining, to increase its performance.

Types of Computer

Microcomputer

Minicomputer

Personal computer

Supercomputer

Laptop

Tablet

www.InformationQ.com