EXPT.-08

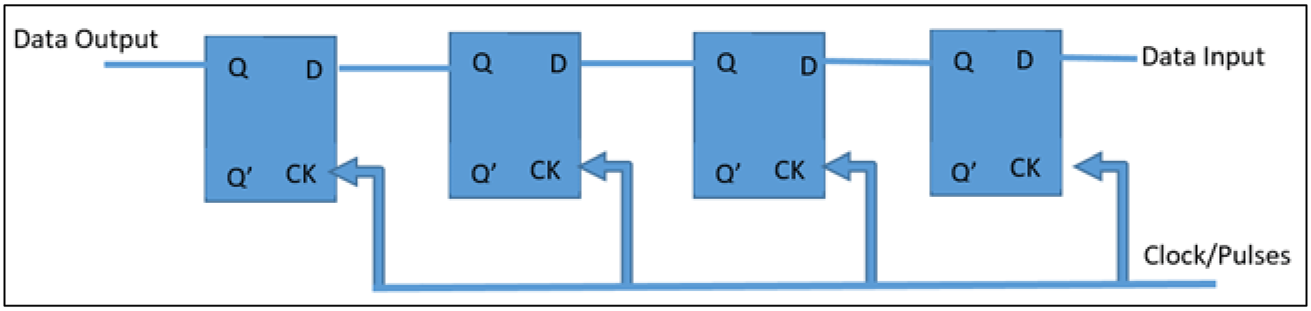
AIM: - Design and Simulate SIPO register using Verilog HDL.

Software: - Xilinx Vivado 2019.1

Theory: -

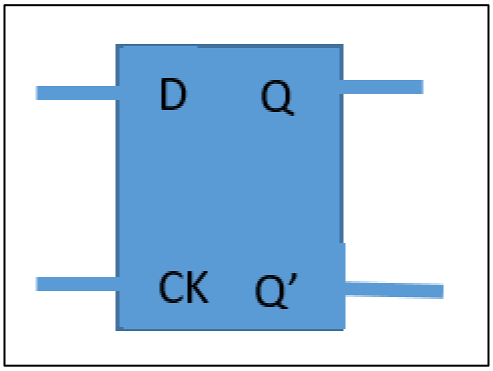
What is Shift Register?

Shift Registers are **sequential logic circuits**, capable of storage and transfer of data. They are made up of Flip Flops which are connected in such a way that the output of one flip flop could serve as the input of the other flip-flop, depending on the type of shift registers being created.



D-Flip Flop shift Register

Shift registers are basically a type of register which have the ability to transfer ("shift") data. Registers are generically storage devices which are created by connecting a specific number of flip flops together in series and the amount of data (number of bits) which can be stored by the register is always directly proportional to the number of flip flops, as each flip flop is capable of storing only one bit at a time. *When the flip-flops in a register are connected in such a way that the output of one flip flop, becomes the input of the other, a shift register is created.*



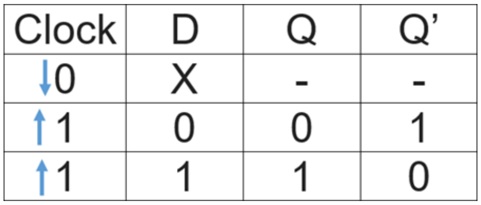
D-Flip Flop

Flip Flops are devices with an operation similar to that of a **latch**. It can be referred to as a bistable vibrator that can move between two states (0 or 1) and is capable of storing data in bits. New data is read into a flip flop with each clock cycle and the previous data sent at the output.

Shift Registers Comprise of which flip-flops?

This however depends on the kind of flip flop, as the Input, Output, and clock cycle relationship between flip flops vary. There are different kinds of flip flops, but the most commonly used in the creation of shift registers are the D (Delay)-flip flops.

For the operation of the D flip flops which makes them so desirable for shift registers,
Whenever there is a change on the clock of a D flip flop (either rising or falling edge, depending on the specifications of the flip flop). The data at the output **“Q”** becomes the same data as the one at the input **“D”.** The Output **“Q”** of the flip flop will stay at that value until the next clock cycle, where it will then change again to the value(High or low, 1 or 0) at the input.



Clock	D	Q	Q’
↓0	X	-	-
↑1	0	0	1
↑1	1	1	0

D Flip Flop Truth Table

Now that we know what Sift Registers are, we will proceed to take a deeper dive into the types of flip-flop and their applications. But before that, to give a more practical exposure on where shift registers are used let's take a look at the popular shift register 74HC595 which we have used with different microcontrollers to interface a display or sequence of LEDs.

- Shift Register with 74HC595 with Arduino to control a sequence of LEDs
- Shift Register with ESP32 to interface 7-Segment Display
- Shift Register with Raspberry Pi to control multiple LEDs
- Shift Register with PIC to control sequence of LEDs

Types of Registers in Digital Electronics

Shift registers are categorized into types majorly by their **mode of operation, either serial or parallel.**

There are six (6) basic **types of shift registers** which are listed below although some of them can be further divided based on the direction of data flow either shift right or shift left.

1. Serial in – Serial out Shift Register (SISO)

2. Serial In – Parallel out shift Register (SIPO)

3. Parallel in – Parallel out Shift Register (PIPO)

4. Parallel in – Serial out Shift Register (PISO)

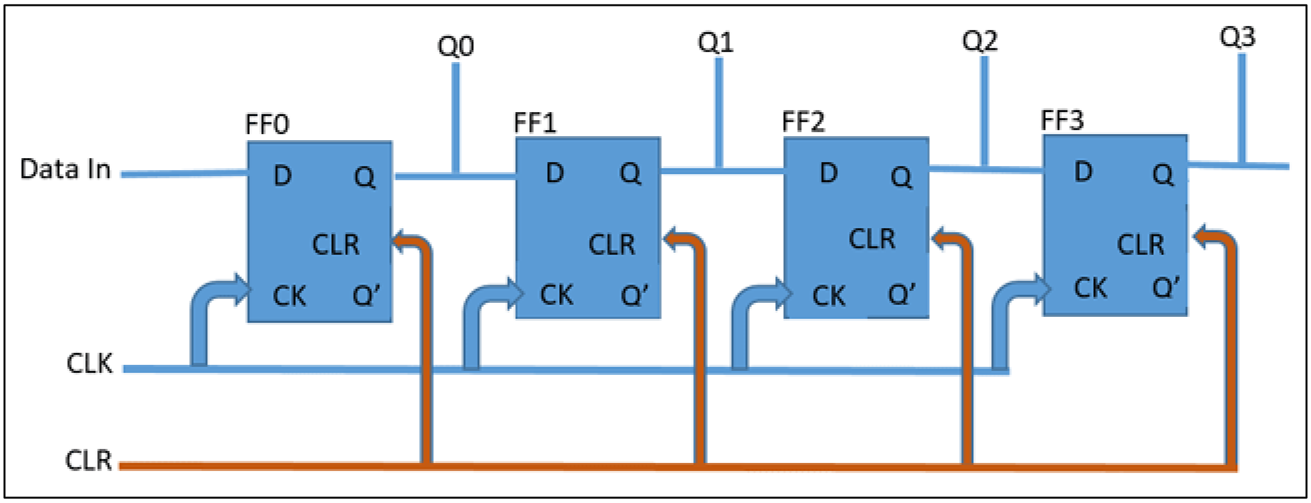
5. Bidirectional Shift Registers

6. Counters

Serial in – Parallel out Shift Register

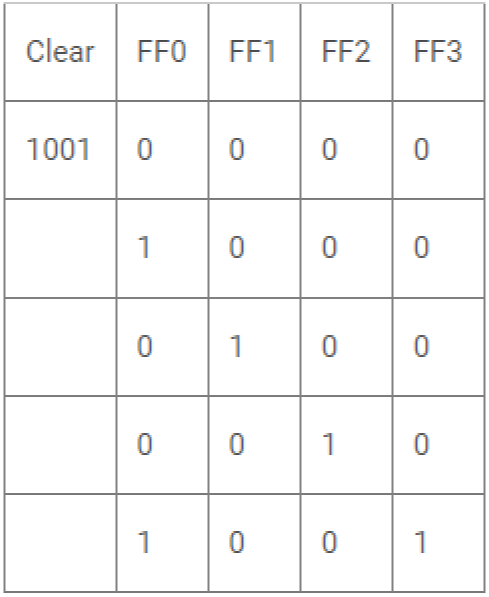
The second type of shift register we will be considering is the Serial in – Parallel out shift register also known as **SIPO Shift Register**. These types of shift registers are used for the conversion of data from serial to parallel. The data comes in one after the other per clock cycle and can either be shifted and replaced or be **read off at each output**. This means when the data is read in, each read in bit becomes available simultaneously on their respective output line (Q0 – Q3 for the 4-bit shift register shown below).

A 4-bits serial in – Parallel out shift register is illustrated in the Image below.



Serial in – Parallel out Shift Register

A table showing how data gets shifted out of serial in –parallel out 4 bit shift register is shown below, with the data in as 1001.



Clear	FF0	FF1	FF2	FF3
1001	0	0	0	0
	1	0	0	0
	0	1	0	0
	0	0	1	0
	1	0	0	1

A good example of the serial in – parallel out shift register is the 74HC164 shift register, which is an 8-bit shift register.

Code: -

```
//DFF
`timescale 1ns / 1ps

module DFF(clk,d,q);
input d,clk;
output reg q;

initial
q<=1'b0;

always @(posedge clk)
q<=d;

endmodule
```

```
//SIPO_4bit
`timescale 1ns / 1ps

module SIPO(d,clk,q);
input d;
input clk;
output [3:0] q;

DFF FF3(clk,d,q[3]);
DFF FF2(clk,q[3],q[2]);
DFF FF1(clk,q[2],q[1]);
DFF FF0(clk,q[1],q[0]);

endmodule
```

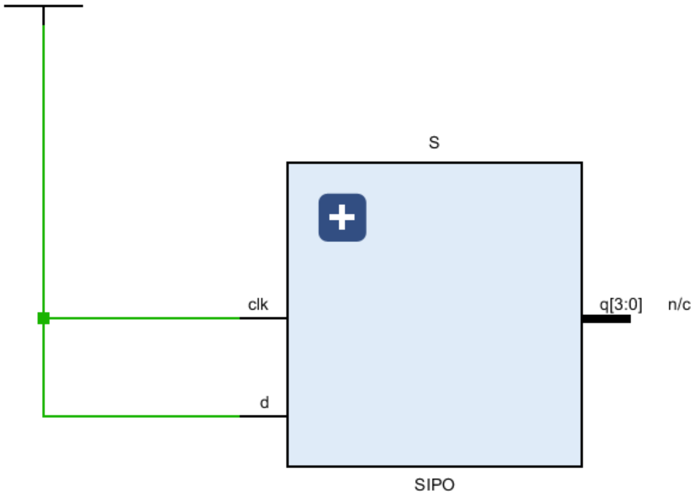
```
//tb 4bit
`timescale 1ns / 1ps

module tb();
reg d;
reg clk;
//wire [7:0] q;
wire [3:0] q;

SIPO S(.d(d),.clk(clk),.q(q));
//SIPO_8BIT S(.d(d),.clk(clk),.q(q));
initial
begin
clk=1'b0;
forever #5 clk=~clk;
end

initial
begin
d=1'b1;
#10 d=1'b0;
#10 d=1'b1;
#10 d=1'b1;
/*#10 d=1'b1;
#10 d=1'b0;
#10 d=1'b1;
#10 d=1'b1;
*/
end

initial
$monitor($time,"d=%b,clk=%b,q3=%b,q2=%b,q1=%b,q0=%b",d,clk,q[3],q[2],q[1],q[0]);

initial
#40 $finish;
//#80 $finish;

endmodule
```

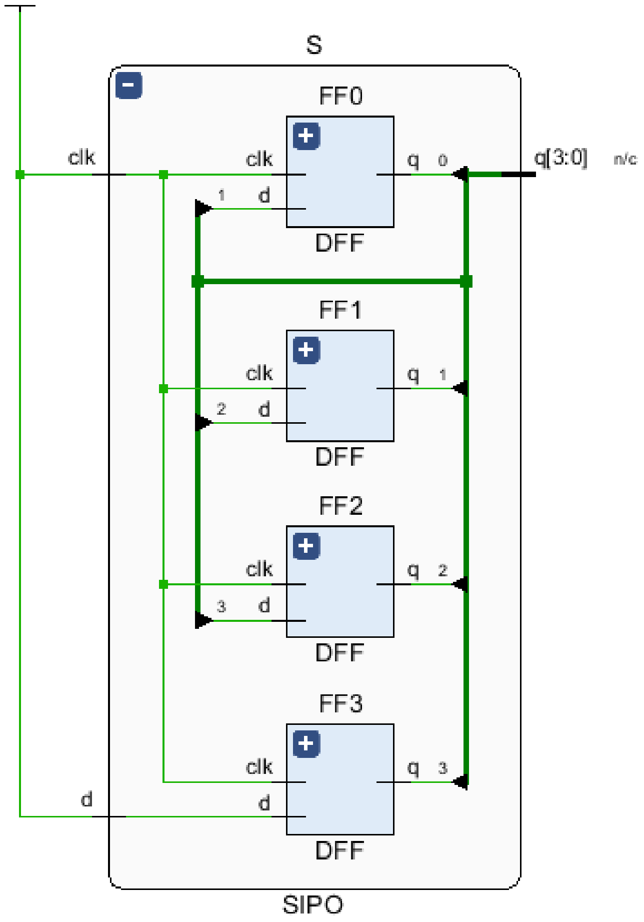
```
//tb 8bit
`timescale 1ns / 1ps

module tb();
reg d;
reg clk;
wire [7:0] q;

//SIPO S(.d(d),.clk(clk),.q(q));
SIPO_8BIT S(.d(d),.clk(clk),.q(q));
initial
begin
clk=1'b0;
forever #5 clk=~clk;
end

initial
begin
d=1'b1;
#10 d=1'b0;
#10 d=1'b1;
#10 d=1'b1;
#10 d=1'b1;
#10 d=1'b0;
#10 d=1'b1;
#10 d=1'b1;
end

initial
$monitor($time,"d=%b,clk=%b,q=%b",d,clk,q);

initial
#80 $finish;

endmodule
```

Schematics and Simulated Waveforms: -

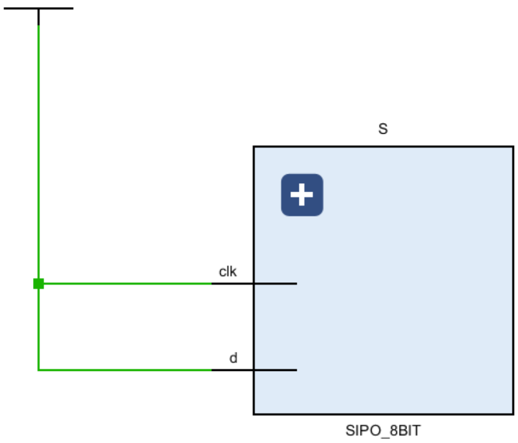
Schematic:

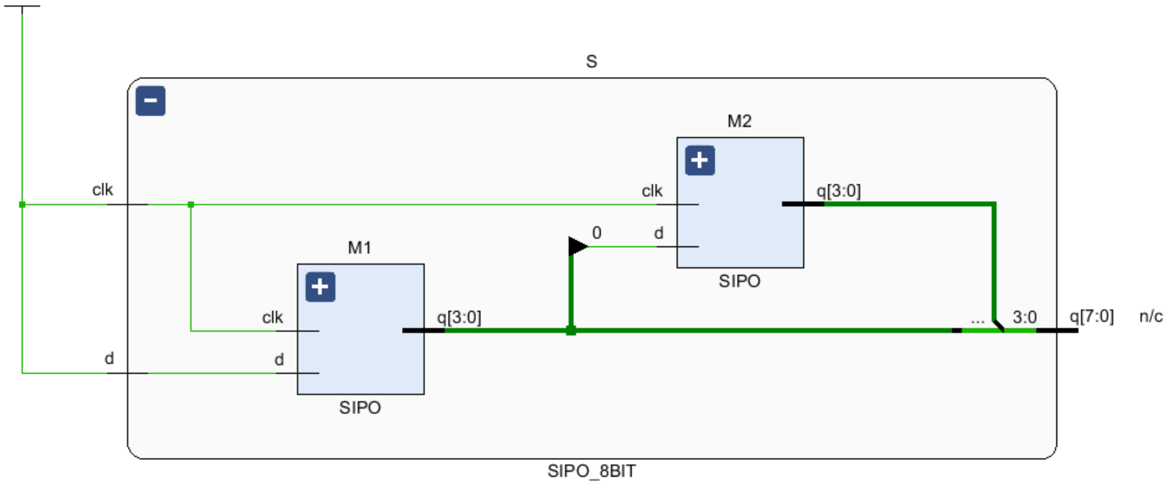
SIPO 4bit

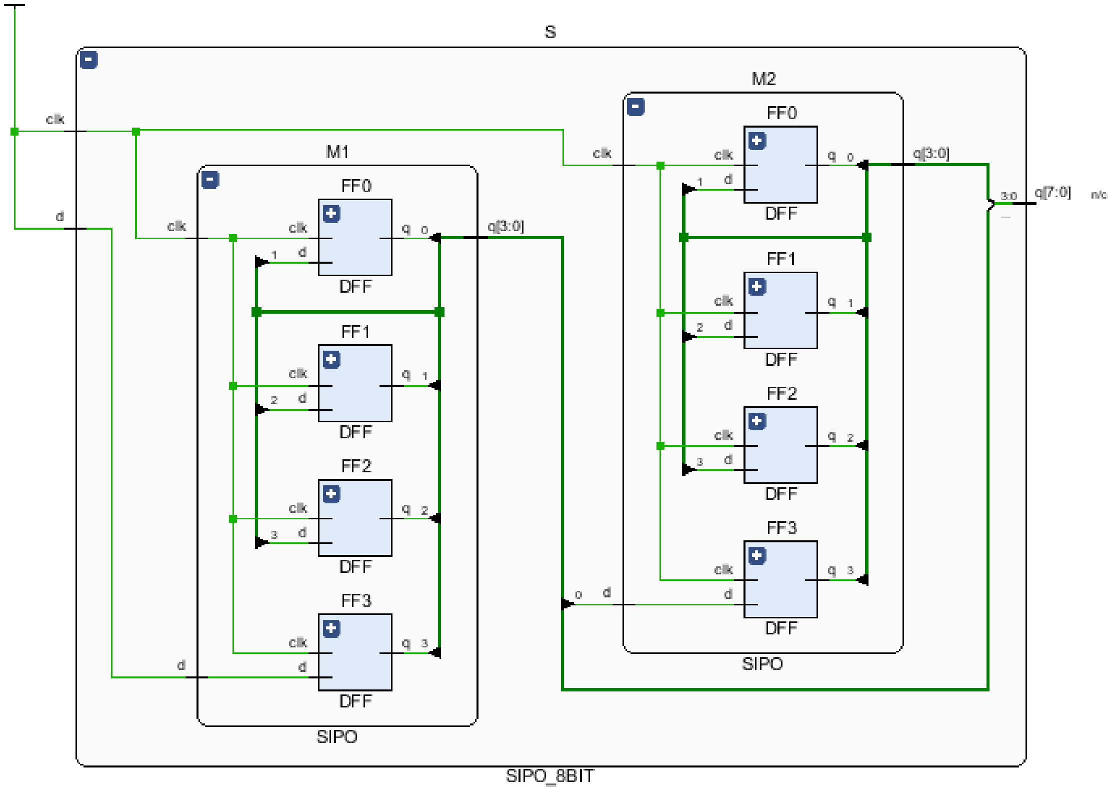




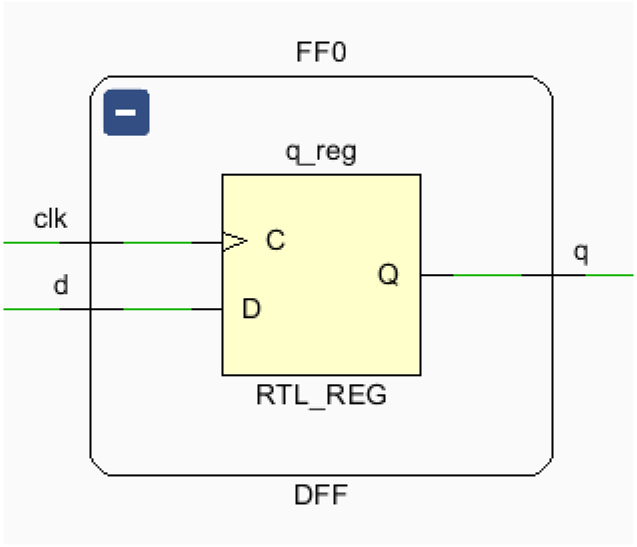
SIPO 8bit


S
clk
d
SIPO_8BIT


S
clk
d
M2
clk
d
q[3:0]
0
SIPO
M1
clk
d
q[3:0]
SIPO
...
3:0
q[7:0]
n/c
SIPO_8BIT

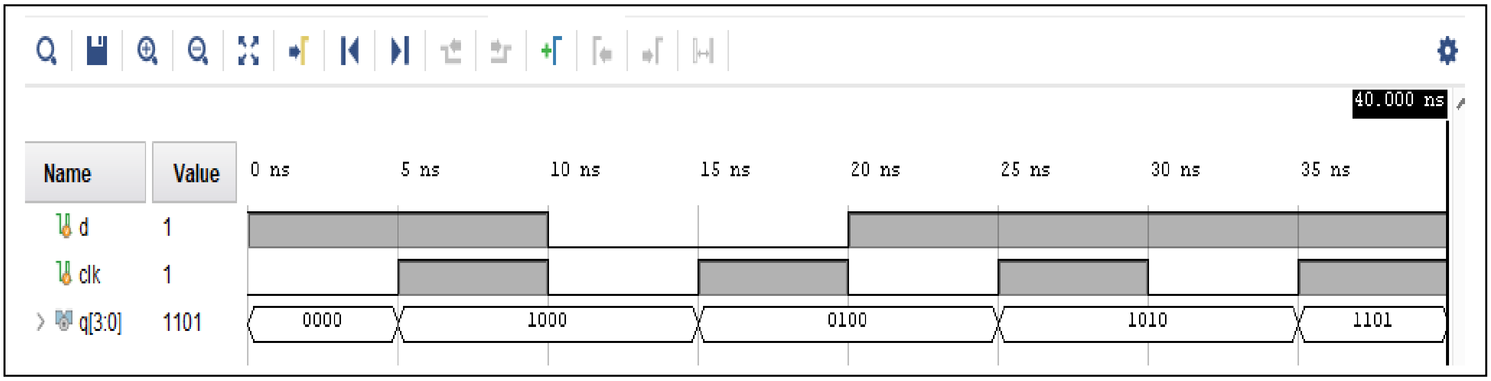


DFF

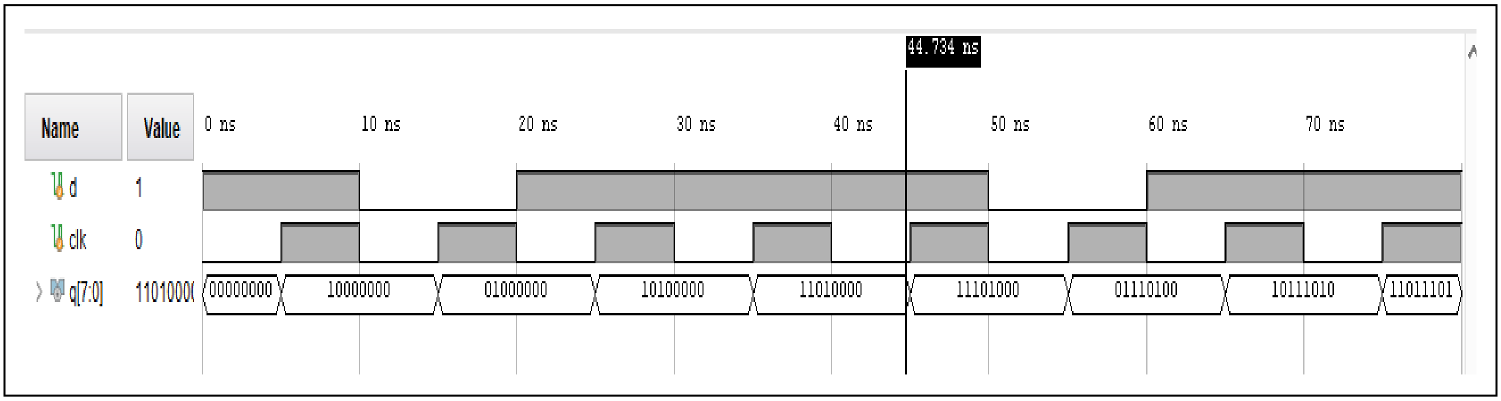


Waveform:

SIPO 4bit



SIPO 8bit



Conclusion: -

Hence , we have successfully designed and simulated SIPO register using behavioral modeling style in Verilog HDL. Also verified simulated waveforms with actual truth tables using Xilinx Vivado 2019.1