CS221-L Data Structures and Algorithms Lab

Faculty of Computer Science and Engineering

GIK Institute of Engineering Sciences and Technology

Lab # 12
Shift Registers in Verilog

Submitted by: **Shayan Rizwan Yazdanie** [**2024585**]

Submitted to: Engr. Irfanullah

Semester: 3rd

Serial-In Serial-Out Shift Register (SISO):

Verilog Code:

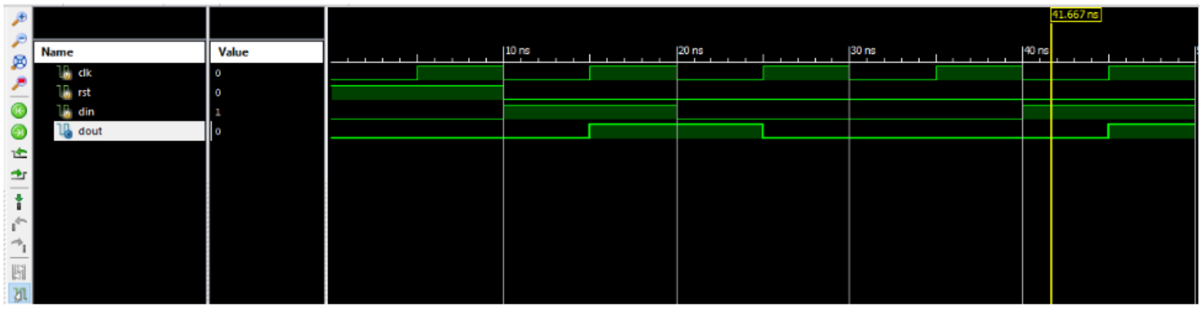
```
  `timescale 1ns / 1ps
module SISO(input clk, input rst, input din, output dout);
reg [3:0] q;
always @(posedge clk or posedge rst) begin
if (rst)
q <= 4'b0000;
else
q[0] <= q[1];
q[1] <= q[2];
q[2] <= q[3];
q[3] <= din;
end
assign dout = q[3];
endmodule
```

Test Bench

```
`timescale 1ns / 1ps
module SISO_TB();
reg clk, rst, din;
wire dout;
// Instantiate the DUT
SISO uut(.clk(clk), .rst(rst), .din(din), .dout(dout));
always #5 clk = ~clk;
initial begin
// Initialize signals
clk = 0;
rst = 1;
din = 0;
#10;
rst = 0;
din = 1;
#10;
din = 0;
#10;
din = 0;
```

```
#10;
din = 1;
#10;
$finish;
end
endmodule
```

Output:



Serial-In Parallel-Out Shift Register (SIPO):

Verilog Code:

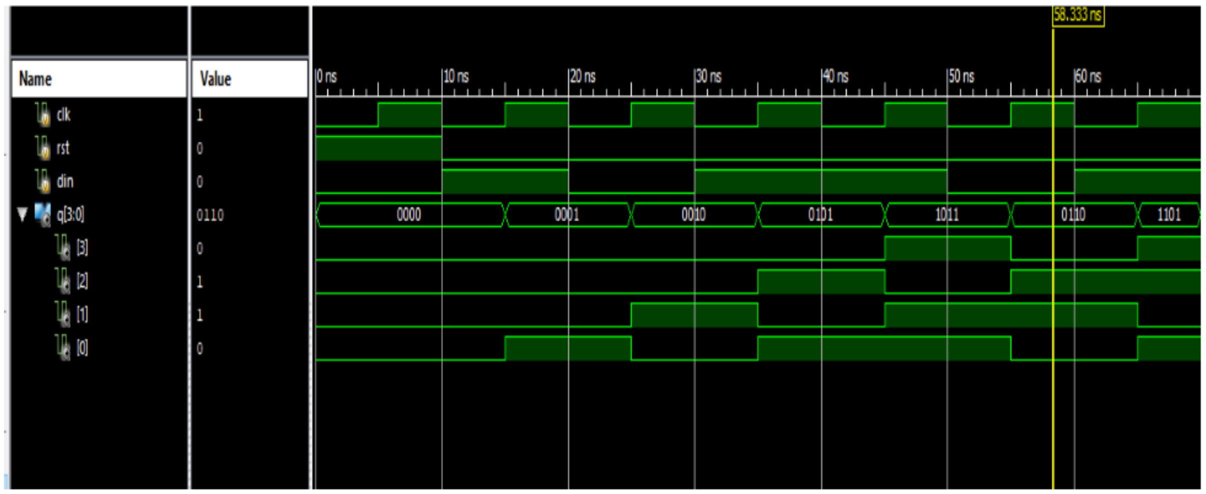
```
`timescale 1ns / 1ps
module SIPO (
input clk,
input rst,
input din,
output reg [3:0] q
);
always @(posedge clk or posedge rst) begin
if (rst)
q <= 4'b0000;
else begin
q[3] <= q[2];
q[2] <= q[1];
q[1] <= q[0];
q[0] <= din;
end
end
endmodule
```

Test Bench

```
`timescale 1ns / 1ps
module tb_SIPO;
reg clk;
reg rst;
reg din;
wire [3:0] q;
// Instantiate DUT
SIPO uut ( .clk(clk), .rst(rst),
.din(din),
.q(q)
initial begin
clk = 0;
forever #5 clk = ~clk;
end
initial begin
rst = 1;
din = 0;
#10;
```

```
rst = 0;
din = 1; #10; // Bit 1
din = 0; #10; // Bit 2
din = 1; #10; // Bit 3
din = 1; #10; // Bit 4
din = 0; #10;
din = 1; #10;
$finish;
end
endmodule
```

Output:



Universal Shift Register (USR):

Verilog Code:

```
`timescale 1ns / 1ps
module universal_SR (
input clk, rst, input [1:0] mode, input SI_left,
input SI_right, input [3:0] D,
output reg [3:0] Q);
always @(posedge clk) begin
if (rst) begin
Q[3] <= 0;
Q[2] <= 0;
Q[1] <= 0;
Q[0] <= 0;
end
else begin
case(mode)
2'b00: begin // HOLD
// Do nothing
end
2'b01: begin // SHIFT RIGHT
Q[3] <= SI_left;
Q[2] <= Q[3];
Q[1] <= Q[2];
Q[0] <= Q[1];
end
2'b10: begin // SHIFT LEFT
Q[3] <= Q[2];
Q[2] <= Q[1];
Q[1] <= Q[0];
Q[0] <= SI_right;
end
2'b11: begin // PARALLEL LOAD
Q[3] <= D[3];
Q[2] <= D[2];
Q[1] <= D[1];
Q[0] <= D[0];
end
endcase
```

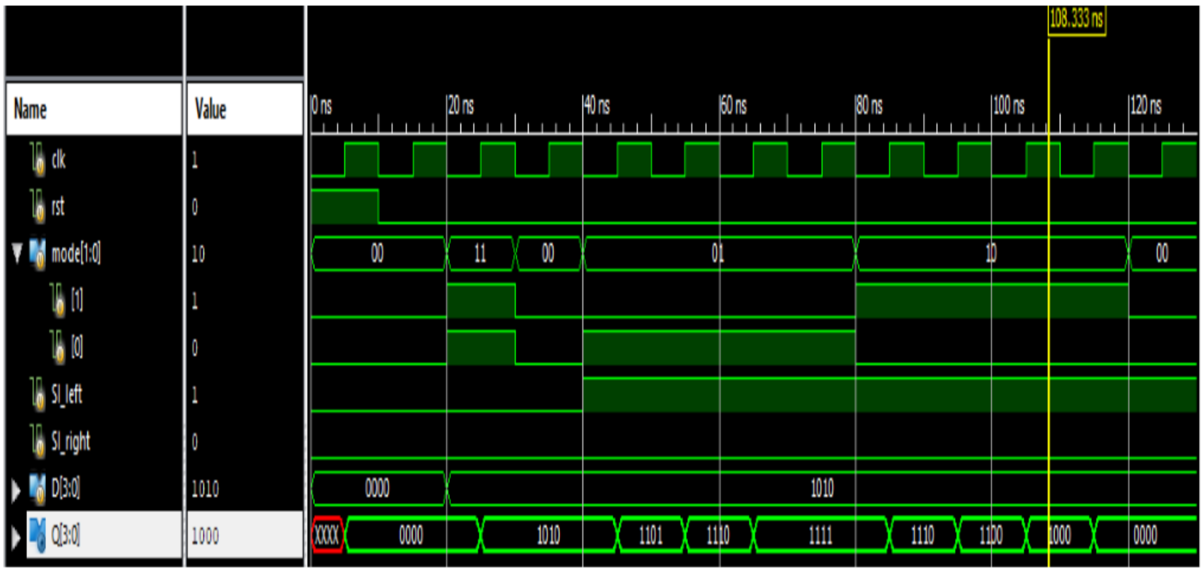
```
end
end
endmodule
```

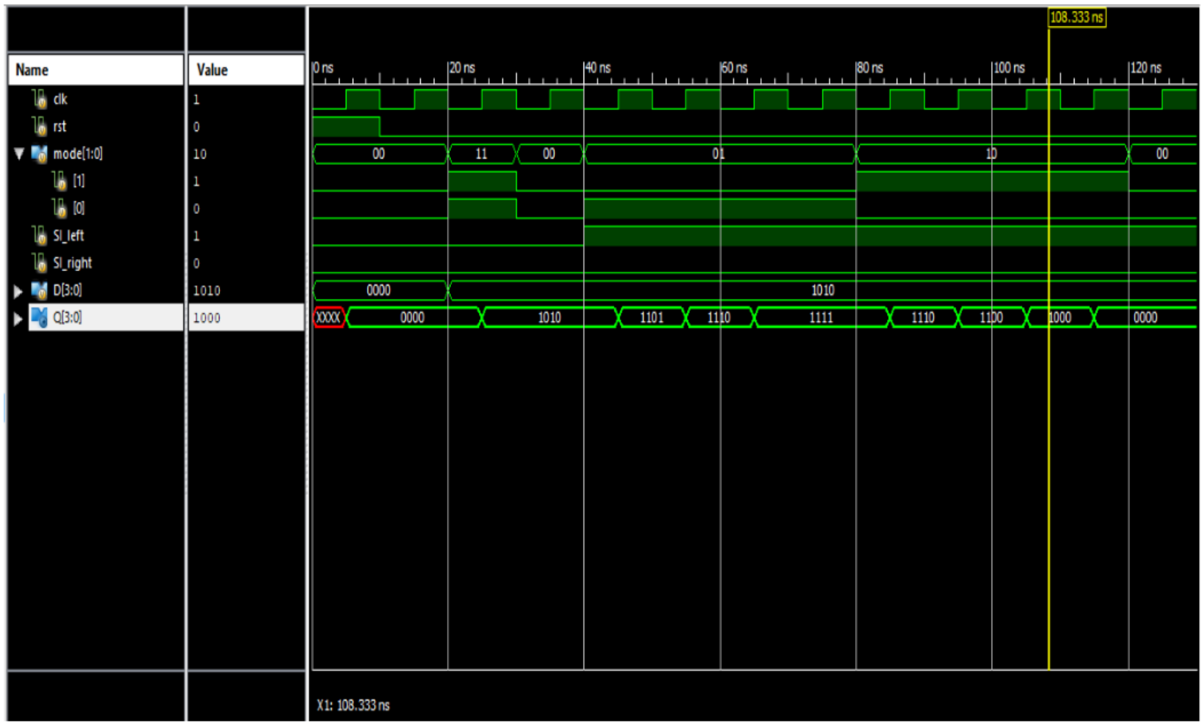
Test Bench

```
`timescale 1ns / 1ps
module universal_SR_TB;
reg clk;
reg rst;
reg [1:0] mode;
reg SI_left;
reg SI_right;
reg [3:0] D;
wire [3:0] Q;
universal_SR uut (
.clk(clk),
.rst(rst),
.mode(mode),
.SI_left(SI_left),
.SI_right(SI_right),
.D(D),
.Q(Q)
);
initial begin
clk = 0;
forever #5 clk = ~clk;
end
initial begin
// Initialize inputs
rst = 1;
mode = 2'b00;
SI_left = 0;
SI_right = 0;
D = 4'b0000;
#10;
rst = 0;
#10;
mode = 2'b11;
D = 4'b1010;
#10;
mode = 2'b00;
#10;
mode = 2'b01;
```

```
SI_left = 1;
#40;
mode = 2'b10;
SI_right = 0;
#40;
mode = 2'b00;
#10;
$finish;
end
endmodule
```

Output:




108.333 ns
Name
Value
0 ns
20 ns
40 ns
60 ns
80 ns
100 ns
120 ns
clk 1
rst 0
mode[1:0] 10
[1] 1
[0] 0
SI_left 1
SI_right 0
D[3:0] 1010
Q[3:0] 1000
00
11
00
01
10
00
0000
1010
XXXX
0000
1010
1101
1110
1111
1110
1100
1000
0000
X1: 108.333 ns