CS 207 Digital Logic - Spring 2020
Lab 13

James Yu

May 13, 2020

Objective

1. Try implementing shift registers.

Lab Exercise Submission

1. You should name all source code as instructed. Mis-named files will not be recognized.
2. You should submit all source code files with an extension "`.v`".
3. You should submit all source code directly into the sakai system below. **Do not compress them into one folder.**

   `https://sakai.sustech.edu.cn/portal/site/ebf68254-68c5-4cfe-9d42-b26758f854ee`
4. Lab exercises should be submitted before the deadline, typically one week after the lab session. No late submission policy applies to lab exercises.

1 Binary Multiplier

During the lecture, we introduced the digital implementation of shift registers. In this lab, we provide an implementation of 4-bit shift-left registers.

`l_shift.v`

```
module L_shifter(din,CLK,Reset,Q);
input din,CLK,Reset;
output reg[3:0] Q;

always @(posedge CLK) begin
```

```
  if (!Reset)
    Q <= 4'b0;
  else begin
    Q <= Q << 1;
    Q[0] <= din;
    end
  end
endmodule
```

This implementation can be tested with the following testbench:

l_shift.v

```
module shifter_tb;
reg clk, din, clr;
wire [3:0] Q;

initial begin
clk <= 1'b0;
end
always @(clk)
begin
  #1 clk <= !clk;
end

L_shifter sf(din, clk, clr, Q);

initial begin
  $monitor("%3t: D_in is %b, Q is %b,%b,%b,%b.", $time, din, Q[3], Q[2], Q[1], Q[0]);
  #2 clr <= 1'b0;
  #2 clr <= 1'b1; din <= 1'b1;
  #8 din <= 1'b0;
  #8 $finish;
end
endmodule
```

which should produce the following output:

```
  0: D_in is x, Q is x,x,x,x.
  1: D_in is x, Q is x,x,x,x.
  3: D_in is x, Q is 0,0,0,0.
  4: D_in is 1, Q is 0,0,0,0.
  5: D_in is 1, Q is 0,0,0,1.
  7: D_in is 1, Q is 0,0,1,1.
  9: D_in is 1, Q is 0,1,1,1.
 11: D_in is 1, Q is 1,1,1,1.
```

```
12: D_in is 0, Q is 1,1,1,1.
13: D_in is 0, Q is 1,1,1,0.
15: D_in is 0, Q is 1,1,0,0.
17: D_in is 0, Q is 1,0,0,0.
19: D_in is 0, Q is 0,0,0,0.
```

2 Exercise

2.1 Exercise 1

Implement a 4-bit binary universal shift register 74LS194. The circuit has the following functions: In this table, s, a, b, c, and d are new input data. The module should follow the design below:

CLR	S_1	S_0	CLK	S_L	S_R	D_0	D_1	D_2	D_3	Q_0^{n+1}	Q_1^{n+1}	Q_2^{n+1}	Q_3^{n+1}	Func
0	X	X	X	X	X	X	X	X	X	0	0	0	0	clear
1	0	0	X	X	X	X	X	X	X	Q_0	Q_1	Q_2	Q_3	keep
1	0	1	↑	X	s	X	X	X	X	s	Q_0	Q_1	Q_2	shift right
1	1	0	↑	s	X	X	X	X	X	Q_1	Q_2	Q_3	s	shift left
1	1	1	↑	X	X	a	b	c	d	a	b	c	d	parallel load

74LS194.v

```
module universal(Q, CLR, CLK, S, D, SR, SL);
// Module Code
endmodule
```

and save in file 74LS194.v.

Assignment

Save the source code in **74LS194.v**. Upload this file to Sakai under **Assignments → Lab Exercise 13**.

2.2 Exercise 2

Implement a sequential signal generator with the counter design. Upon positive clock pulse, the generator should yield the next bit within sequence 1101000101 and repeat. The module should follow the design below:

seq.v

```
module seq(Z, CLK);
// Module Code
endmodule
```

and save in file `seq.v`.

Assignment

Save the source code in **`seq.v`**. Upload this file to Sakai under **Assignments → Lab Exercise 13**.

Assignment

When you finished Lab Exercise 13, there should be two `.v` files in the Sakai system: **`74LS194.v`** and **`seq.v`**.