Class:	CPE-300L	Semester:	Spring 2020

Points		Document author:	Brysen Kokubun
		Author's email:	Brysenkokubun96@gmail.com
		Document topic:	Prelab 4

Instructor's comments:

Introduction / Theory of operation

- For this lab experiment we will be implementing various latches and flip-flops, Verilog models, synchronous and asynchronous operation and asynchronous system design. We will be verifying our models by using the RTL viewer, Waveforms and DE2 board implementations.

Prelab main content

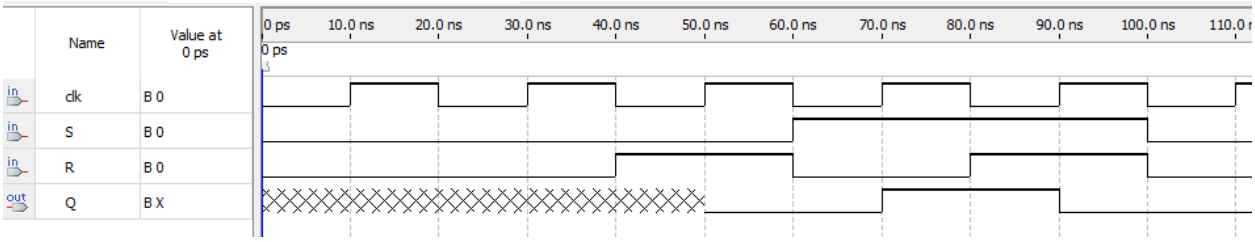
1. Implementation of SR latch: Structural Verilog Models

```
1   module part1 (clk, R, S, Q);
2       input clk, R, S;
3       output Q;
4       wire R_g, S_g, Qa, Qb;
5
6       assign R_g = R & clk;
7       assign S_g = S & clk;
8       assign Qa = ~(R_g | Qb);
9       assign Qb = ~(S_g | Qa);
10      assign Q = Qa;
11
12  endmodule
13
```

Component Instantiation

```
1   module part1 (clk, R, S, Q);
2       input clk, R, S;
3       output Q;
4       wire R_g, S_g, Qa, Qb;
5       and (R_g, R, clk);
6       and (S_g, S, clk);
7       nor (Qa, R_g, Qb);
8       nor (Qb, S_g, Qa);
9       assign Q = Qa;
10  endmodule
11
```

Structural Model

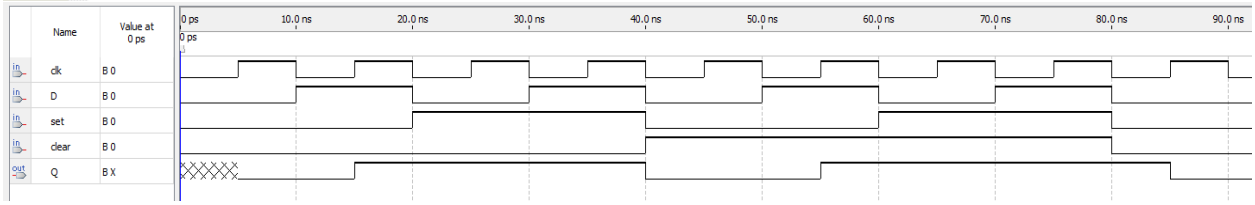


Waveforms of both modules created the same output. The difference between the 2 codes are the way they are written. One code uses component instantiation with assign statements and another code uses a structural model with gate primitives. Both will perform same task just written differently.

2. Asynchronous set/clear

S-R latch with added set and clear input

```
1   module part1 (clk, D, set, clear, Q);
2      input clk, D, set, clear;
3      output Q;
4      wire R_g, S_g, Qa, Qb, S, R;
5
6      assign R = ~D;
7      assign S = D;
8      nand (R_g, R, clk);
9      nand (S_g, S, clk);
10     nand (Qb, ~set, S_g, Qa);
11     nand (Qa, ~clear, R_g, Qb);
12     assign Q = Qb;
13
14  endmodule
15
```



3. JK flip-flop

JK #1

```
module J_K (Q, QN, J, K, Clock);
   output Q, QN; //data output
   input J, K;   //data input
   input Clock;
   reg Q;
   always @(posedge Clock)
      begin
      Q = J&(~Q)|(~K)&Q;
   end
   assign QN = ~Q;
endmodule
```

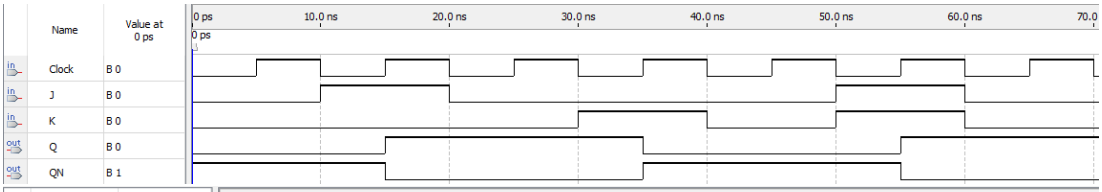
JK #2

```
module J_K (Q, QN, J, K, Clock);
   output Q, QN; //data output
   input J, K;   //data input
   input Clock;
   reg Q;
   always @(posedge Clock)
      begin
      if ({J,K} == 2'b01) Q = 1'b0;
      else if ({J,K} == 2'b10) Q = 1'b1;
      else if ({J,K} == 2'b11) Q = ~Q;
      else Q = Q;
   end
      assign QN = ~Q;
endmodule
```

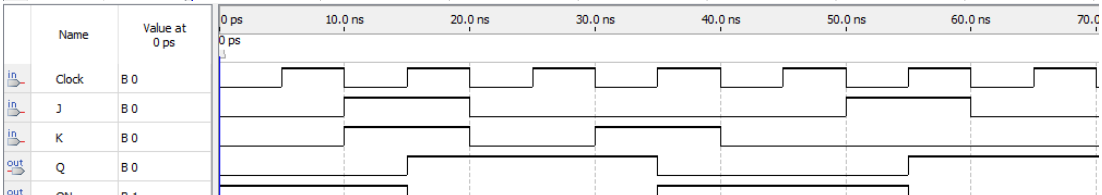
JK #3

```
module J_K (Q, QN, J, K, Clock);
   output Q, QN; //data output
   input J, K;   //data input
   input Clock;
   reg Q;
   always @(posedge Clock)
      begin
      case ({J,K})
      2'b01: Q = 1'b0;
      2'b10: Q = 1'b1;
      2'b11: Q = ~Q;
      default: Q = Q;
      endcase
   end
   assign QN = ~Q;
endmodule
```

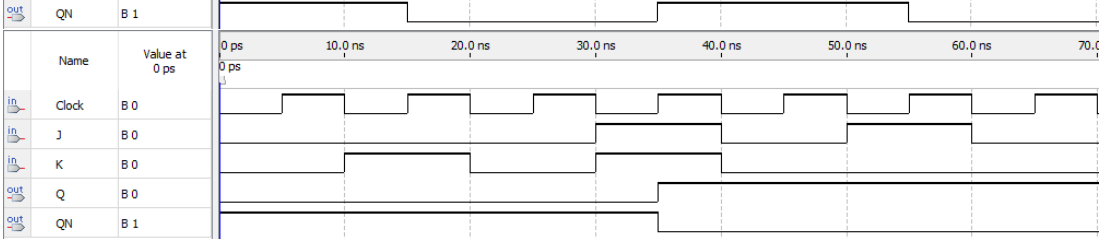
Waveforms



JK #1



JK #2



JK #3

Utilization

JK #1

Resource	Usage
Total logic elements	1 / 14,400 (< 1 %)
-- Combinational with no register	0
-- Register only	0
-- Combinational with a register	1
Logic element usage by number of LUT inputs	
-- 4 input functions	0
-- 3 input functions	1
-- <=2 input functions	0
-- Register only	0
Logic elements by mode	
-- normal mode	1
-- arithmetic mode	0
Total registers*	1 / 14,733 (< 1 %)
-- Dedicated logic registers	1 / 14,400 (< 1 %)
-- I/O registers	0 / 333 (0 %)
Total LABs: partially or completely used	1 / 900 (< 1 %)
Virtual pins	0
I/O pins	5 / 81 (6 %)
-- Clock pins	0 / 6 (0 %)
-- Dedicated input pins	0 / 12 (0 %)
Global signals	0
M9Ks	0 / 60 (0 %)
Total block memory bits	0 / 552,960 (0 %)
Total block memory implementation bits	0 / 552,960 (0 %)

JK #2

Resource	Usage
Total logic elements	1 / 14,400 (< 1 %)
-- Combinational with no register	0
-- Register only	0
-- Combinational with a register	1
Logic element usage by number of LUT inputs	
-- 4 input functions	0
-- 3 input functions	1
-- <=2 input functions	0
-- Register only	0
Logic elements by mode	
-- normal mode	1
-- arithmetic mode	0
Total registers*	1 / 14,733 (< 1 %)
-- Dedicated logic registers	1 / 14,400 (< 1 %)
-- I/O registers	0 / 333 (0 %)
Total LABs: partially or completely used	1 / 900 (< 1 %)
Virtual pins	0
I/O pins	5 / 81 (6 %)
-- Clock pins	0 / 6 (0 %)
-- Dedicated input pins	0 / 12 (0 %)
Global signals	0
M9Ks	0 / 60 (0 %)
Total block memory bits	0 / 552,960 (0 %)
Total block memory implementation bits	0 / 552,960 (0 %)
PLLs	0 / 3 (0 %)

JK #3

Resource Usage Summary

Resource	Usage
Total logic elements	1 / 14,400 (< 1 %)
-- Combinational with no register	0
-- Register only	0
-- Combinational with a register	1
Logic element usage by number of LUT inputs	
-- 4 input functions	0
-- 3 input functions	1
-- <=2 input functions	0
-- Register only	0
Logic elements by mode	
-- normal mode	1
-- arithmetic mode	0
Total registers*	1 / 14,733 (< 1 %)
-- Dedicated logic registers	1 / 14,400 (< 1 %)
-- I/O registers	0 / 333 (0 %)
Total LABs: partially or completely used	1 / 900 (< 1 %)
Virtual pins	0
I/O pins	5 / 81 (6 %)
-- Clock pins	0 / 6 (0 %)
-- Dedicated input pins	0 / 12 (0 %)
Global signals	0
M9Ks	0 / 60 (0 %)
Total block memory bits	0 / 552,960 (0 %)
Total block memory implementation bits	0 / 552,960 (0 %)
PLLs	0 / 3 (0 %)

4. JK with synchronous clear

```
1  module J_K (Q, QN, J, K, ClearN, Clock);
2     output Q, QN;          //Output Data
3     input J, K, ClearN;    //Input Data
4     input Clock;
5     reg Q;
6     always @(posedge Clock)
7        begin
8        case ({J,K,ClearN})
9        3'bxx0: Q = 1'b0;
10       3'b011: Q = 1'b0;
11       3'b101: Q = 1'b1;
12       3'b111: Q = ~Q;
13       default: Q = Q;
14       endcase
15       end
16    assign QN = ~Q;
17 endmodule
```