Fall 2019

California State University, Northridge

Department of Electrical and Computer Engineering

ECE 526L

Experiment # 4
Behavioral Modeling of a Counter

October 03, 2019

Authors: Ridge Tejuco
Professor: Ronald Mehler Ph.D

I hereby attest that this lab report is entirely my own work. I have not copied either code or text from anyone, nor have I allowed or will I allow anyone to copy my work.

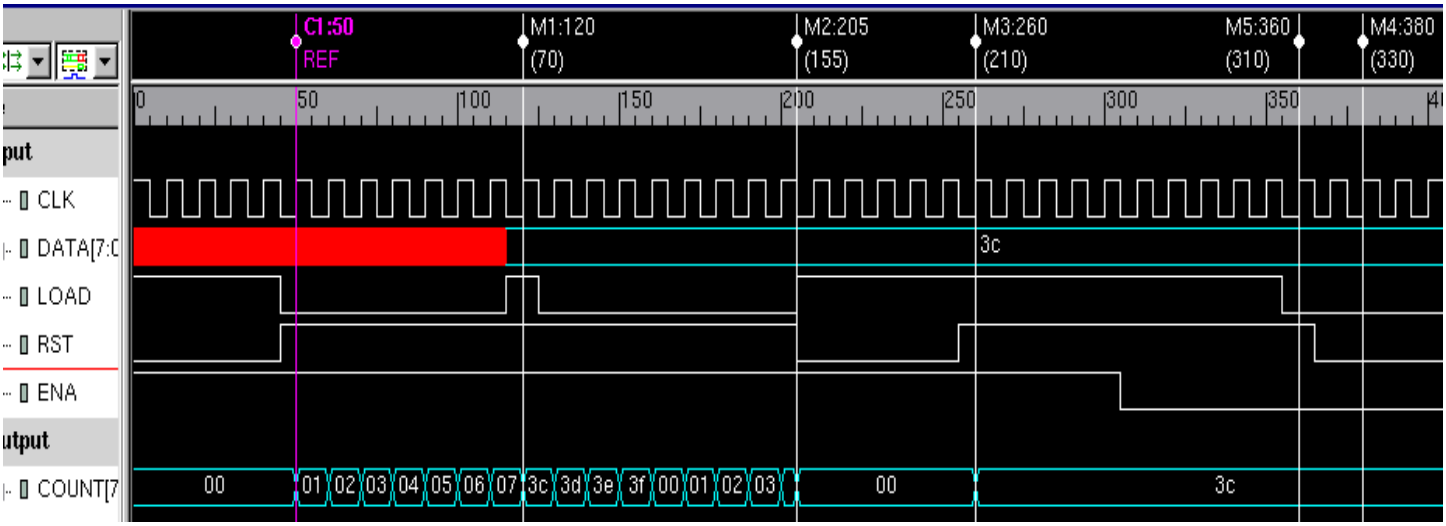
Name (printed) _____

Name (signed) _____

Date _____

Analysis

Figure 1. Top Counter waveform



For the first 50 ns, RST is set to 0. The COUNT is constantly held at 00_h. From 50 ns to 120 ns, The COUNT is incrementing from 00 to 07 with a period of 10 ns. Looking closely, the correct inputs of LOAD = 0, RST = 1, ENA = 1 are set to perform this behavior. The increment and RST behavior works correctly.

Before the CLK edge at 120 ns, LOAD is set to 1 and DATA is set to $(3c)_h$. At 120 ns , 3c is clocked to the COUNT. The COUNT continues to increment and then rolls over to 00 then continues to count.

At 205 ns , before the next clock pulse, RST is set to 0 and the COUNT is instantly reset to 00 before the next clock pulse. This demonstrates that the reset function is asynchronous.

At 260 ns, LOAD is set to 1 and 3C is again loaded. This time LOAD is held at 1 which demonstrates that the counter does not increment while the load operation is held.

At 360 ns, ENA = 0 and LOAD = 0 is tested. As expected, there is no increment in the count while ENA = 0. At 365 ns, RST is set to 0, but the COUNT did not reset to 00. This shows that the ENA input has priority over the reset and the load behavior.

One way to make bigger counters with instances of this design is to add a max pulse detector when the count rolls over from 63 to 00. The max pulse would input to the ENABLE of the next instance for 1 clock cycle allowing the next counter to increment by 1 with input LOAD = 0.

```
/*==================================
CSUN - ECE 526L
Ridge Tejuco
==================================
AASD.sv
GOALS
DFF1 <= RST,DFF2 <= DFF1 @ posedge clk when RST = 1
DFF 1&2 <= 0 @ negedge RST
==================================*/
`timescale 1 ns/ 100 ps
module AASD (OUT,CLK,RST);
    output reg OUT;
    input wire CLK,RST;
    reg DFF0_OUT;
    always @(posedge(CLK), negedge(RST)) begin
      if(RST == 0) begin
            DFF0_OUT <= 1'b0;
            OUT <= 1'b0;
      end else begin
            OUT <= DFF0_OUT;
            DFF0_OUT <= RST;
      end
    end
endmodule
```

```
/*==================================
CSUN - ECE 526L
Ridge Tejuco
Sept. 26, 2019
==================================
COUNTER.sv
GOALS
- COUNT[5:0] <== DATA [5:0] IF      LOAD = 1,
                  ENABLE = 1,
                  RESET = 1;
- COUNT[5:0] <== COUNT[5:0]+1 IF LOAD = 0,
                  ENABLE =1,
                  RESET = 1;
- COUNT[5:0] <== 6'b000000  IF      RESET = 0,
                  CLK = DC;
- COUNT[5:0] <== COUNT[5:0] IF      ENABLE = 0,
                  RESET = 1;
====================================*/
`timescale 1ns / 100 ps

module COUNTER(COUNT,DATA,CLK, RST,ENA, LOAD);
    parameter SIZE = 6;
    output reg [SIZE - 1:0] COUNT;
    input wire [SIZE - 1:0] DATA;
    input wire CLK,RST,ENA,LOAD;

    //start flip flop
    always @ (posedge(CLK) or negedge(RST)) begin
      if(RST == 0)
           COUNT <= 6'b000000;
      else begin
           if(ENA == 1) begin
                if(LOAD == 1)
                     COUNT <= DATA;
                else
                     COUNT <= COUNT + 6'b000001;
           end
      end
    end
endmodule
```

```
/*================================
CSUN - ECE 526L
Ridge Tejuco
Sept. 26, 2019
================================
TOP_COUNTER_AASD.sv
- Attach AASD to the reset of COUNTER
- AASD_OUTPUT <= RESET of COUNTER
================================*/
`timescale 1 ns/ 100 ps
module TOP_COUNTER_AASD (COUNT,DATA,CLOCK,RESET,ENABLE,LOAD);
    parameter SIZE = 6;
    output reg [SIZE - 1:0] COUNT;
    input wire [SIZE - 1:0] DATA;
    input wire CLOCK,RESET,ENABLE,LOAD;
    wire AASD_RESET;
    //COUNTER (COUNT,DATA,CLK, RST,ENA, LOAD);
    COUNTER C0(COUNT,DATA,CLOCK,RESET,ENABLE,LOAD);
    //AASD (OUT,CLK,RST);
    AASD(AASD_RESET,CLOCK,RESET);
endmodule
```

```
/*================================
CSUN - ECE 526L
Ridge Tejuco
Sept. 26, 2019
================================
TB_COUNTER_AASD.sv
GOALS
- instantiate TOP_COUNTER_AASD.sv [x]
- CLOCK of 10 ns [x]
- Test asynch RST [x]
- Increment COUNTER to 7 @ EN = 1 RST = 1 LOAD = 0; [x]
- after 7, load 60    [x]
- Test RST priority over load and increment [x]
- Test enable    [x]
- How to make bigger? 8bit? 32 bit?
================================*/
`timescale 1 ns / 100 ps
module TB_COUNTER_AASD ();
    parameter SIZE = 8,
         PERIOD = 5,
         DELAY = 50;
    reg [SIZE-1:0] COUNT, DATA;
    reg CLK,RST,LOAD,ENA;
    //TOP_COUNTER_AASD (COUNT,DATA,CLOCK,RESET,ENABLE,LOAD)
    TOP_COUNTER_AASD CA0(COUNT,DATA,CLK,RST,ENA,LOAD);
    // CLOCK GEN
  always begin
    #PERIOD CLK = ~ CLK;
  end
  initial begin
     $vcdpluson;
      RST = 0; CLK = 1; ENA = 1; LOAD = 1; // Test asnych RST
      #DELAY RST = 1;LOAD = 0;     // Increment COUNTER to 7
      #(PERIOD*14) LOAD = 1; DATA = 6'b111100; // LOAD 60
      #1  LOAD = 0;
      #(DELAY + 30) RST = 0; LOAD = 1;    //Test RST priority
      #DELAY RST = 1;
      #DELAY ENA = 0;       //Test ENA
      #DELAY LOAD = 0;
     #(PERIOD*2) RST = 0;
      #DELAY $finish;
  end
endmodule
```