FPGA Application Week 13 Exercise

CYEE 10828241 Chen Da-Chuan

May 20, 2023

Contents

List of Figures

List of Tables

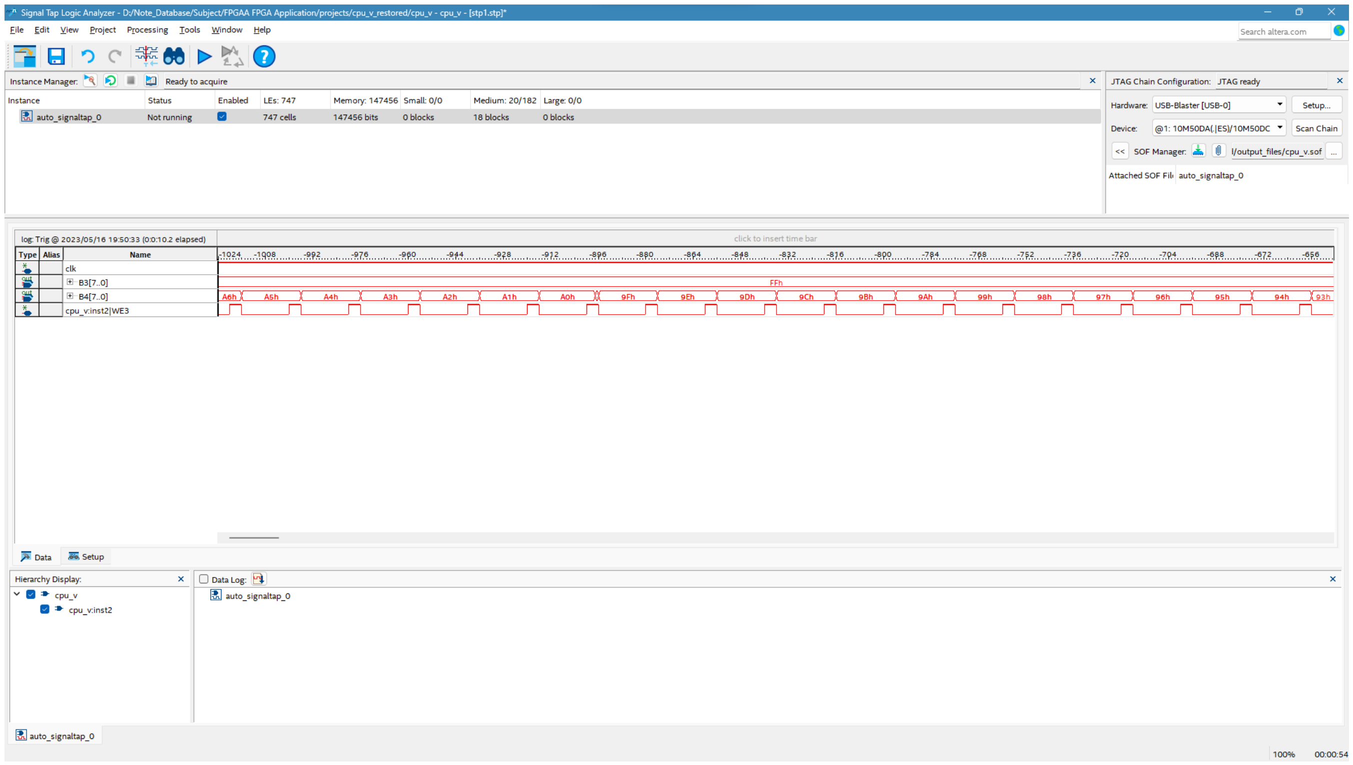
1 Exercise 13-1 Signal Tap Logic Analyzer

1.1 Objective

Add additional nodes of counter output to the Signal Tap Logic Analyzer and observe the waveform.
This can be achieved by modify the WIDTH parameter of the clock divider module.

1.2 Result

With much larger WIDTH, the clock divider module will output a much slower clock signal, which can be observed by the Signal Tap Logic Analyzer.



Instance Manager: Ready to acquire

Instance	Status	Enabled	LEs: 747	Memory: 147456	Small: 0/0	Medium: 20/182	Large: 0/0
auto_signaltap_0	Not running	✓	747 cells	147456 bits	0 blocks	18 blocks	0 blocks

JTAG Chain Configuration: JTAG ready

Hardware: USB-Blaster [USB-0]

Device: @1: 10M50DA(.|ES)/10M50DC

SOF Manager: l/output_files/cpu_v.sof

Attached SOF File: auto_signaltap_0

log: Trig @ 2023/05/16 19:50:33 (0:0:10.2 elapsed)

Type	Alias	Name	-1024 … -656
		clk	
		B3[7..0]	FFh
		B4[7..0]	A6h A5h A4h A3h A2h A1h A0h 9Fh 9Eh 9Dh 9Ch 9Bh 9Ah 99h 98h 97h 96h 95h 94h 93h
		cpu_v:inst2|WE3	

Figure 1: Signal Tap Logic Analyzer

2 Exercise 13-2 Delay functionality with 7-segment display

2.1 Objective

Use different clock speed to display CPU register value on the 7-segment display.

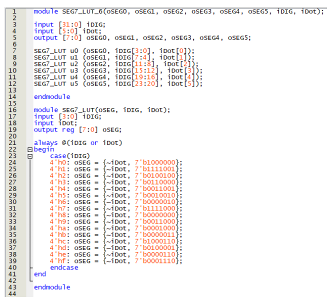
2.2 Operation

Table 1: Homework 13-2 Operation detail

type	var	operation
input	clk	clock signal
inout	B1-2	memory bank 1-2
output	HEX0-5	7-segment display
output	B0, B3-15	memory bank 0, 3-15

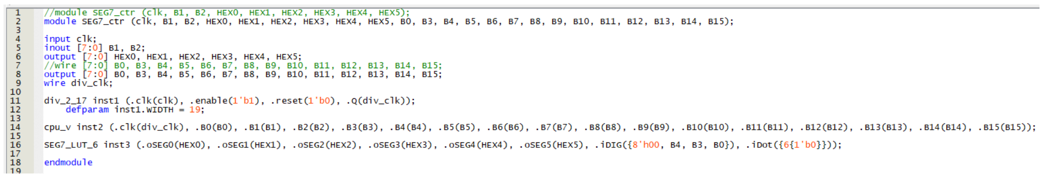
2.3 Code

We need to modify the PIN assignments with DE10 user manual.



```
1   module SEG7_LUT_6(oSEG0, oSEG1, oSEG2, oSEG3, oSEG4, oSEG5, iDIG, iDot);
2
3   input [31:0] iDIG;
4   input [5:0] iDot;
5   output [7:0] oSEG0, oSEG1, oSEG2, oSEG3, oSEG4, oSEG5;
6
7   SEG7_LUT u0 (oSEG0, iDIG[3:0], iDot[0]);
8   SEG7_LUT u1 (oSEG1, iDIG[7:4], iDot[1]);
9   SEG7_LUT u2 (oSEG2, iDIG[11:8], iDot[2]);
10  SEG7_LUT u3 (oSEG3, iDIG[15:12], iDot[3]);
11  SEG7_LUT u4 (oSEG4, iDIG[19:16], iDot[4]);
12  SEG7_LUT u5 (oSEG5, iDIG[23:20], iDot[5]);
13
14  endmodule
15
16  module SEG7_LUT(oSEG, iDIG, iDot);
17  input [3:0] iDIG;
18  input iDot;
19  output reg [7:0] oSEG;
20
21  always @(iDIG or iDot)
22  begin
23      case(iDIG)
24      4'h0: oSEG = {~iDot, 7'b1000000};
25      4'h1: oSEG = {~iDot, 7'b1111001};
26      4'h2: oSEG = {~iDot, 7'b0100100};
27      4'h3: oSEG = {~iDot, 7'b0110000};
28      4'h4: oSEG = {~iDot, 7'b0011001};
29      4'h5: oSEG = {~iDot, 7'b0010010};
30      4'h6: oSEG = {~iDot, 7'b0000010};
31      4'h7: oSEG = {~iDot, 7'b1111000};
32      4'h8: oSEG = {~iDot, 7'b0000000};
33      4'h9: oSEG = {~iDot, 7'b0011000};
34      4'ha: oSEG = {~iDot, 7'b0001000};
35      4'hb: oSEG = {~iDot, 7'b0000011};
36      4'hc: oSEG = {~iDot, 7'b1000110};
37      4'hd: oSEG = {~iDot, 7'b0100001};
38      4'he: oSEG = {~iDot, 7'b0000110};
39      4'hf: oSEG = {~iDot, 7'b0001110};
40      endcase
41  end
42
43  endmodule
44
```

Figure 2: SEG7_LUT_6, 7-segment display driver



```
1   //module SEG7_ctr (clk, B1, B2, HEX0, HEX1, HEX2, HEX3, HEX4, HEX5);
2   module SEG7_ctr (clk, B1, B2, HEX0, HEX1, HEX2, HEX3, HEX4, HEX5, B0, B3, B4, B5, B6, B7, B8, B9, B10, B11, B12, B13, B14, B15);
3
4   input clk;
5   inout [7:0] B1, B2;
6   output [7:0] HEX0, HEX1, HEX2, HEX3, HEX4, HEX5;
7   //wire [7:0] B0, B3, B4, B5, B6, B7, B8, B9, B10, B11, B12, B13, B14, B15;
8   output [7:0] B0, B3, B4, B5, B6, B7, B8, B9, B10, B11, B12, B13, B14, B15;
9   wire div_clk;
10
11  div_2_17 inst1 (.clk(clk), .enable(1'b1), .reset(1'b0), .Q(div_clk));
12      defparam inst1.WIDTH = 19;
13
14  cpu_v inst2 (.clk(div_clk), .B0(B0), .B1(B1), .B2(B2), .B3(B3), .B4(B4), .B5(B5), .B6(B6), .B7(B7), .B8(B8), .B9(B9), .B10(B10), .B11(B11), .B12(B12), .B13(B13), .B14(B14), .B15(B15));
15
16  SEG7_LUT_6 inst3 (.oSEG0(HEX0), .oSEG1(HEX1), .oSEG2(HEX2), .oSEG3(HEX3), .oSEG4(HEX4), .oSEG5(HEX5), .iDIG({8'h00, B4, B3, B0}), .iDot({6{1'b0}}));
17
18  endmodule
19
```

Figure 3: SEG7_ctr, main module

1. Figure 2 Line1-12: Define 7-segment display driver consisted with 6 identical submodules.
2. Figure 3 Line21-43: Define submodules that turns on and off each segment and dot of the 7-segment display.
3. Figure 3 Line11-12: Generate a divided clock signal with clock divider module.
4. Figure 3 Line14-14: Use CPU to execute program in ROM.
5. Figure 3 Line16-16: use register value in CPU to drive the 7-segment display.

2.4 Result

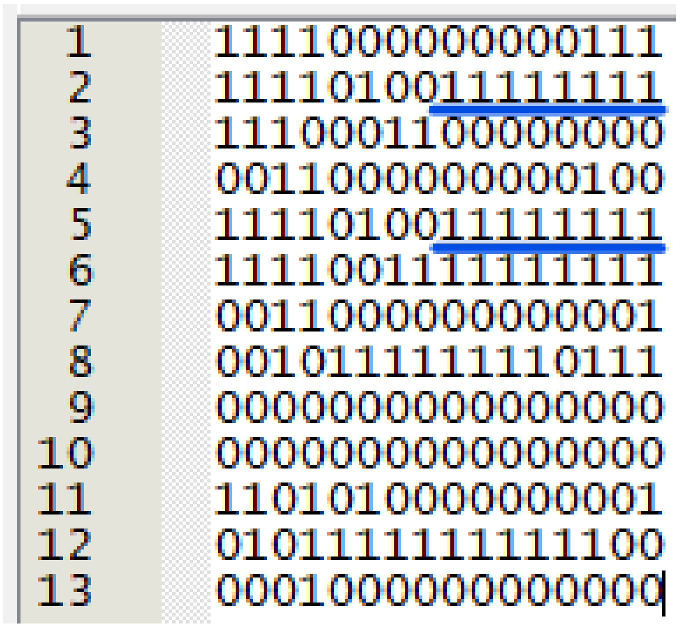
Recorded video is available at `https://youtu.be/AqM1ElJ0kSM`.

3 Exercise 13-3 Modify ROM instruction

3.1 Objective

Set the same delay time for $B3 = FF$ and $B3 = 00$ by modifying ROM commands.

3.2 Code



```
1    1111000000000111
2    1111010011111111
3    1110001100000000
4    0011000000000100
5    1111010011111111
6    1111001111111111
7    0011000000000001
8    0010111111110111
9    0000000000000000
10   0000000000000000
11   1101010000000001
12   0101111111111100
13   0001000000000000
```

Figure 4: New ROM commands

For line 2 and 5, they are responsible for setting register value that controls how long do we want to delay. Therefore we should give them the same value in both lines.

3.3 Result

Recorded video is available at `https://youtu.be/yLjpn83MYnM`.