FPGA Application Week 12 Exercise

CYEE 10828241 Chen Da-Chuan

May 17, 2023

Contents

List of Figures

List of Tables

1 Exercise 12-1 Design a frequency divider

1.1 Objective

Code an adjustable frequency divider.

1.2 Operation

Table 1: Homework 12-1 Operation detail

type	var	operation
input	clk	clock signal
input	enable	enable signal
input	reset	reset signal
output reg	count	counter register number
output	Q	divided clock signal

1.3 Code

```
1   // Quartus Prime Verilog Template
2   // Binary counter
3
4   module div
5   #(parameter WIDTH=4)
6   (
7      input clk, enable, reset,
8      output reg [WIDTH-1:0] count,
9      output Q
10  );
11
12     // Reset if needed, or increment if counting is enabled
13     always @ (posedge clk or posedge reset)
14     begin
15        if (reset)
16           count <= 0;
17        else if (enable == 1'b1)
18           if (count <  5'b01010)
19              count <= count + 1;
20           else
21              count <= 0;
22     end
23
24     assign Q = (count>{1'b0, 4'b0101} ) ? 1:0;
25
26  endmodule
27
```

Figure 1: Main file

```
1   `timescale 1ns/1ns
2   module test_div;
3   reg clk;
4   reg enable, reset;
5   wire [3:0] count;
6   wire Q;
7
8   div DUT(
9        .clk(clk),
10       .enable(enable),
11       .reset(reset),
12       .count(count),
13       .Q(Q)
14  );
15
16  initial
17  begin
18       clk = 0;
19       reset = 0;
20       enable = 1;
21  end
22
23  always #5 clk = ~clk;
24
25  endmodule
```

Figure 2: Test file

Note: Line24 in Figure 1 should have "$>=$" instead of "$=$". Or else the output signal will not have exactly 50% duty cycle.

1. Figure 1 Line4-10: Declare module, parameter, input, output.
2. Figure 1 Line15-16: If triggered by clock or reset signal, if reset is high, reset counter register number to 0.
3. Figure 1 Line18-19: If triggered by clock or reset signal, if enable is high, if counter register number is less then 10, add 1 to counter register number.
4. Figure 1 Line21-22: If triggered by clock or reset signal, if enable is high, if counter register number is 10, reset counter register number to 0.
5. Figure 1 Line24-24: If counter register number is larger then 5, output Q is high, else output Q is low.
6. Figure 2 Line1-6: Declare timescale, module, register, and wire.
7. Figure 2 Line8-14: Include main file for testing.
8. Figure 2 Line16-21: Set initial value for input.
9. Figure 2 Line23-23: Alternate clock signal every 5ns.

1.4 Result

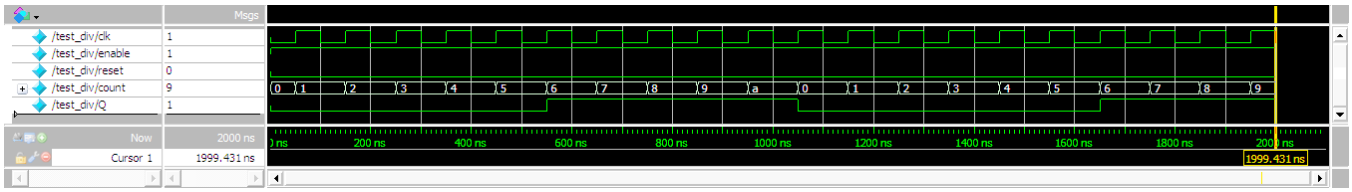

Figure 3: 12-1

2 Exercise 12-2 LEDs blink with different parameter

2.1 Objective

Use the frequency divider to slow down clock speed and make LEDs blink with previously built frequency divider.

2.2 Operation

Table 2: Homework 12-2 Operation detail

type	var	operation
input	clk	clock signal
inout	B1	memory bank 1
inout	B2	memory bank 2
output	B0	memory bank 0
output	B3	memory bank 3
output	B4	memory bank 4
output	B5	memory bank 5
output	B6	memory bank 6
output	B7	memory bank 7
output	B8	memory bank 8
output	B9	memory bank 9
output	B10	memory bank 10
output	B11	memory bank 11
output	B12	memory bank 12
output	B13	memory bank 13
output	B14	memory bank 14
output	B15	memory bank 15

2.3 Code

```
1  module led_ctr (clk, B0, B1, B2, B3, B4, B5, B6, B7, B8, B9, B10, B11, B12, B13, B14, B15);
2  input clk;
3  inout [7:0] B1, B2;
4  output [7:0] B0, B3, B4, B5, B6, B7, B8, B9, B10, B11, B12, B13, B14, B15;
5  wire div_clk;
6
7  div_2_17 inst1 (.clk(clk), .enable(1'b1), .reset(1'b0), .Q(div_clk));
8  defparam inst1.WIDTH = 15;
9
10 cpu_v inst2 (.clk(div_clk), .B1(B1), .B2(B2), .B0(B0), .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));
11
12 endmodule
```

Figure 4: Main file

1. Figure 4 Line1-5: Declare module, input, output, and wire.
2. Figure 4 Line7-8: Include frequency divider module and divide $50MHz$ clock signal by $2^{15} = 32768$ resulting in $1525.87890625Hz$.
3. Figure 4 Line10-10: Include CPU module with divided clock signal.

Table 3: Homework 12-2 16bits ROM commands

16'b	16'h	operation	description
1111000000000111	F007	IOR A0 D07	set B1 and B2 as output
1111010000001111	F40F	IOR A4 D0f	set R4 as 8h'07
1110001100000000	E300	IAND A3 D00	set R3 as 8h'00
0011000000000100	3004	CALL PC+P004+3	call delay function at 4h'3+4'h4+4'h3=4'ha
1111010011111111	F4FF	IOR A4 Dff	set R4 as 8'hff
1111001111111111	F3FF	IOR A3 Dff	set R3 as 8'hff
0011000000000001	3001	CALL PC+P001+3	call delay function at 4'h6+4'h1+4'h3=4'ha
0010111111110111	2FF7	JUMP PC+Pff7+3	call loop function at 4'h7+12'hff7+4'h3=4'h1
0000000000000000	0000	NOP	
0000000000000000	0000	NOP	
1101010000000001	F401	IOR A4 D01	set R4 as R4-1
0101111111111100	5FFC	JNZ PC+Pffc+3	jnz at 4'hb+12'hffc+4'h3=4'ha
0001000000000000	1000	RET	return

Note: A is address, D is data, P is program counter

2.4 Result

Result video is available at:

1. Divider width=15: `https://youtu.be/ihPXbj8fZjU`.
2. Divider width=17: .

3 Exercise 12-3 Find PIN assignments for 7-segment displays

3. 4 Using the 7-segment Displays

The DE10-Lite board has six 7-segment displays to display numbers. Figure 3-17 shows the connection of seven segments (common anode) to pins on MAX 10 FPGA. The segment can be turned on or off by applying a low logic level or high logic level from the FPGA, respectively.

Each segment in a display is indexed from 0 to 6 and DP (decimal point), with corresponding positions given in Figure 3-17. Table 3-6 shows the pin assi zgnment of FPGA to the 7-segment displays.

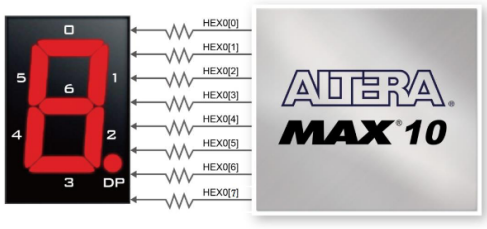

Figure 3-17 Connections between the 7-segment display HEX0 and the MAX 10 FPGA

Table 3-6 Pin Assignment of 7-segment Displays

Signal Name	FPGA Pin No.	Description	I/O Standard
HEX00	PIN_C14	Seven Segment Digit 0[0]	3.3-V LVTTL
HEX01	PIN_E15	Seven Segment Digit 0[1]	3.3-V LVTTL
HEX02	PIN_C15	Seven Segment Digit 0[2]	3.3-V LVTTL
HEX03	PIN_C16	Seven Segment Digit 0[3]	3.3-V LVTTL
HEX04	PIN_E16	Seven Segment Digit 0[4]	3.3-V LVTTL
HEX05	PIN_D17	Seven Segment Digit 0[5]	3.3-V LVTTL
HEX06	PIN_C17	Seven Segment Digit 0[6]	3.3-V LVTTL
HEX07	PIN_D15	Seven Segment Digit 0[7], DP	3.3-V LVTTL
HEX10	PIN_C18	Seven Segment Digit 1[0]	3.3-V LVTTL
HEX11	PIN_D18	Seven Segment Digit 1[1]	3.3-V LVTTL
HEX12	PIN_E18	Seven Segment Digit 1[2]	3.3-V LVTTL
HEX13	PIN_B16	Seven Segment Digit 1[3]	3.3-V LVTTL

Figure 5: Page 1

HEX14	PIN_A17	Seven Segment Digit 1[4]	3.3-V LVTTL
HEX15	PIN_A18	Seven Segment Digit 1[5]	3.3-V LVTTL
HEX16	PIN_B17	Seven Segment Digit 1[6]	3.3-V LVTTL
HEX17	PIN_A16	Seven Segment Digit 1[7] , DP	3.3-V LVTTL
HEX20	PIN_B20	Seven Segment Digit 2[0]	3.3-V LVTTL
HEX21	PIN_A20	Seven Segment Digit 2[1]	3.3-V LVTTL
HEX22	PIN_B19	Seven Segment Digit 2[2]	3.3-V LVTTL
HEX23	PIN_A21	Seven Segment Digit 2[3]	3.3-V LVTTL
HEX24	PIN_B21	Seven Segment Digit 2[4]	3.3-V LVTTL
HEX25	PIN_C22	Seven Segment Digit 2[5]	3.3-V LVTTL
HEX26	PIN_B22	Seven Segment Digit 2[6]	3.3-V LVTTL
HEX27	PIN_A19	Seven Segment Digit 2[7] , DP	3.3-V LVTTL
HEX30	PIN_F21	Seven Segment Digit 3[0]	3.3-V LVTTL
HEX31	PIN_E22	Seven Segment Digit 3[1]	3.3-V LVTTL
HEX32	PIN_E21	Seven Segment Digit 3[2]	3.3-V LVTTL
HEX33	PIN_C19	Seven Segment Digit 3[3]	3.3-V LVTTL
HEX34	PIN_C20	Seven Segment Digit 3[4]	3.3-V LVTTL
HEX35	PIN_D19	Seven Segment Digit 3[5]	3.3-V LVTTL
HEX36	PIN_E17	Seven Segment Digit 3[6]	3.3-V LVTTL
HEX37	PIN_D22	Seven Segment Digit 3[7] , DP	3.3-V LVTTL
HEX40	PIN_F18	Seven Segment Digit 4[0]	3.3-V LVTTL
HEX41	PIN_E20	Seven Segment Digit 4[1]	3.3-V LVTTL
HEX42	PIN_E19	Seven Segment Digit 4[2]	3.3-V LVTTL
HEX43	PIN_J18	Seven Segment Digit 4[3]	3.3-V LVTTL
HEX44	PIN_H19	Seven Segment Digit 4[4]	3.3-V LVTTL
HEX45	PIN_F19	Seven Segment Digit 4[5]	3.3-V LVTTL
HEX46	PIN_F20	Seven Segment Digit 4[6]	3.3-V LVTTL
HEX47	PIN_F17	Seven Segment Digit 4[7] , DP	3.3-V LVTTL
HEX50	PIN_J20	Seven Segment Digit 5[0]	3.3-V LVTTL
HEX51	PIN_K20	Seven Segment Digit 5[1]	3.3-V LVTTL
HEX52	PIN_L18	Seven Segment Digit 5[2]	3.3-V LVTTL
HEX53	PIN_N18	Seven Segment Digit 5[3]	3.3-V LVTTL
HEX54	PIN_M20	Seven Segment Digit 5[4]	3.3-V LVTTL
HEX55	PIN_N19	Seven Segment Digit 5[5]	3.3-V LVTTL
HEX56	PIN_N20	Seven Segment Digit 5[6]	3.3-V LVTTL
HEX57	PIN_L19	Seven Segment Digit 5[7] , DP	3.3-V LVTTL

Figure 6: Page 2

4 Appendix

Table 4: 16bits ROM commands

Instruction	Description	[15:12] (4'd/4'h)	[11:8]	[7:4]	[3:0]
NOP	No Operation	0000/0	N/A	N/A	N/A
RET	Return	0001/1	N/A	N/A	N/A
JUMP	Unconditional Jump	0010/2	P[11:8]	P[7:4]	P[3:0]
CALL	Unconditional Call	0011/3	P[11:8]	P[7:4]	P[3:0]
JZ	Jump if Zero	0100/4	P[11:8]	P[7:4]	P[3:0]
JNZ	Jump if Not Zero	0101/5	P[11:8]	P[7:4]	P[3:0]
JC	Jump if carry flag is zero	0110/6	P[11:8]	P[7:4]	P[3:0]
JNC	Jump if carry flag is not zero	0111/7	P[11:8]	P[7:4]	P[3:0]
ADD	Add two registers	1000/8	R1	R2	R3
SUB	Subtract two registers	1001/9	R1	R2	R3
AND	Bitwise AND two registers	1010/A	R1	R2	R3
OR	Bitwise OR two registers	1011/B	R1	R2	R3
IADD	Add a register and an immediate value	1100/C	R1	I[7:4]	I[3:0]
ISUB	Subtract a register and an immediate	1101/D	R1	I[7:4]	I[3:0]
IAND	Bitwise AND a register and an imm.	1110/E	R1	I[7:4]	I[3:0]
IOR	Bitwise OR a register and an imm.	1111/F	R1	I[7:4]	I[3:0]

Note: R3 stores the result of operation R1 and R2, P stands for PC.