



EGR680 High Level Implementation on FPGA

Laboratory 02

Seven-Segment Display Applications Using PYNQ

Professor: Dr. C. Parikh

Student: Dimitri Häring

September 12, 2018

# Contents

<b>1</b>	<b>Introduction</b>	<b>3</b>
<b>2</b>	<b>Design</b>	<b>4</b>
2.1	Part 1 - Behavioral Modeling of a Seven-Segment Display . . . . .	4
<b>3</b>	<b>Simulation</b>	<b>5</b>
3.1	Part I: behavioral simulation of decoder . . . . .	5
<b>4</b>	<b>Conclusion</b>	<b>6</b>
<b>5</b>	<b>Appendix</b>	<b>7</b>
5.1	Lesson Learned . . . . .	7
5.2	Part I:Decoder module . . . . .	7
5.3	Part I: behavioral simulation of decoder code test bench . . . . .	8

# 1 Introduction

The goal of the applied research is to show that closely spaced power and ground planes aid to the capacitance of the PCB board. This would mean that at higher switching frequencies the embedded capacitance of the PCB board would add to the capacitance of decoupling capacitors.

## 2 Design

In this section the design and decisions that were made to achieve it are discussed.

### 2.1 Part 1 - Behavioral Modeling of a Seven-Segment Display

Verilog is used to describe a decoder that shall control a seven-segment display with two switches and one button. The given task is as followed:

In this part, you will simulate a pattern decoder using the buttons, switches, and a single seven-segment display. Using switches, SW0 and SW1 as the pattern input, write a Verilog program that takes the binary input combinations of the switches and displays the decimal value of the binary switch combination on the seven-segment display. Table 1 shows the switch combinations with the expected output value on the seven-segment display.

**Table 1: Input Combinations and Expected Output**

SW1	SW0	BTN0	Display
0	0	0	OFF
0	0	1	0
0	1	1	1
1	0	1	2
1	1	1	3

As software package to implement the decoder in Verilog Vivado 2017.2 is used.

### 3 Simulation

Describes the result of the behavioral simulation based on the synthesized hardware description language.

#### 3.1 Part I: behavioral simulation of decoder

After a successful simulation of the synthesized hardware description language that implements an decoder a test bench was written, see listening 2. The time variant simulation is shown in figure 1 which shows steps trough the different possible switch positions and shows the output on the decoder bus seg. In comparison to the simulation given in Lab 02 Part 1 it could be confirmed that there are mostly identical.

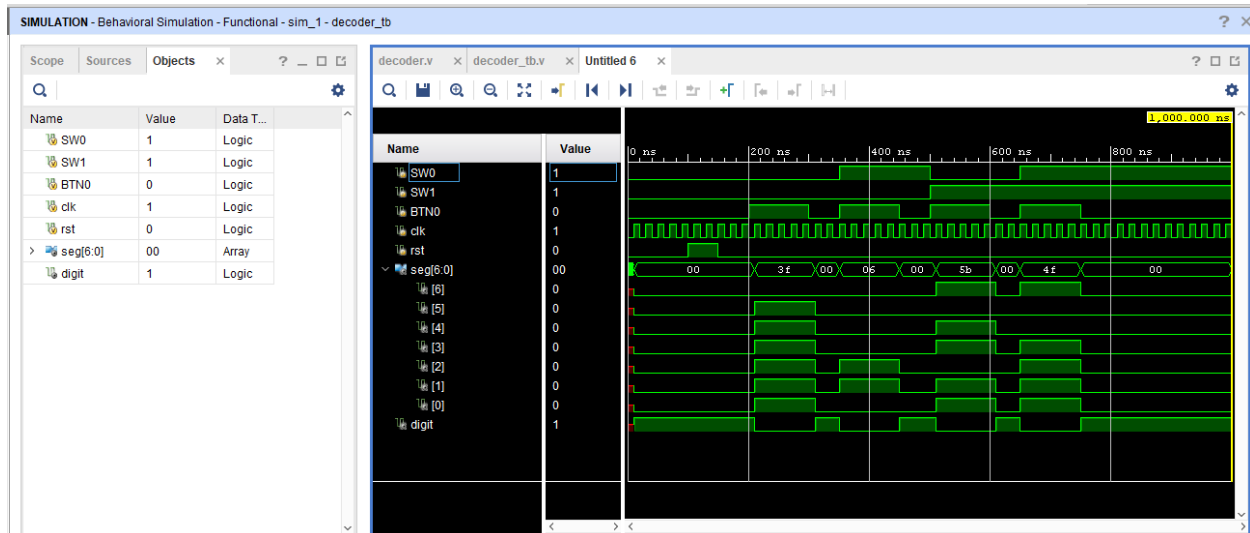


Figure 1: Vivado behavioral simulation of the decoder module.

## 4 Conclusion

## 5 Appendix

The appendix contains code listening and other large information parts that contain partial or complete relevance to the reports topic.

### 5.1 Lesson Learned

Figure 2 shows a handy tool build into Vivado software package that is called Language Templates and it seems to contain most of the verilog syntax.

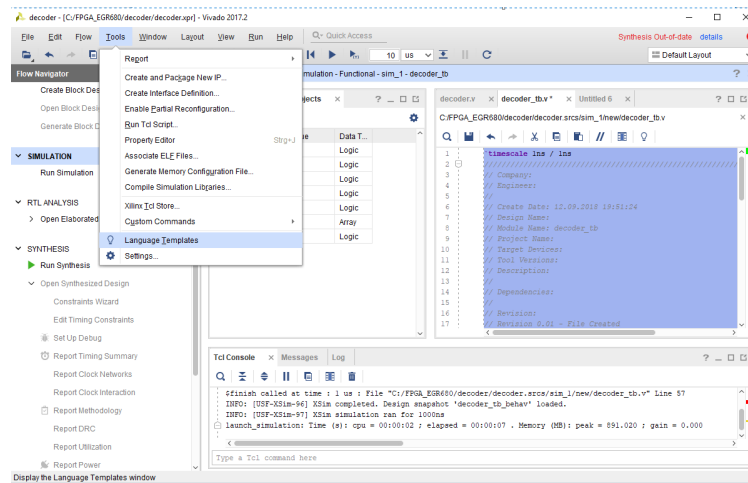


Figure 2: Vivado Language Templates.

### 5.2 Part I:Decoder module

Listing 1: Testbanche decoder part I.

```
'timescale 1ns / 1ns
////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////
// Company:
// Engineer:
//
// Create Date: 12.09.2018 19:31:54
// Design Name:
// Module Name: decoder
// Project Name:
// Target Devices:
// Tool Versions:
// Description:
//
// Dependencies:
//
// Revision:
// Revision 0.01 - File Created
// Additional Comments:
//
////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////
```

```

module decoder(
input SW0,
input SW1,
input BTN0,
input clk,
input rst,
output [6:0] seg,
output digit
);

reg [6:0] seg;
reg digit;

always @(posedge clk or posedge rst)
begin
if(rst)
begin
seg <= 7'b00000000;
digit <= 1'b1; // Digit = 1 (7-segment OFF Digit= 0 (7-segment ON)
end
else
begin
// my code or from webpage http://verilogcodes.blogspot.com/2015/10/verilog-code-for-b
digit = ~BTN0;
if (BTN0 == 1) begin
case ({SW1, SW0}) //case statement
2'b00 : seg = 7'b0111111; //~7'b0000001;
2'b01 : seg = 7'b0000110;
2'b10 : seg = 7'b1011011;
2'b11 : seg = 7'b1001111;
4 : seg = ~7'b1001100;
5 : seg = ~7'b0100100;
6 : seg = ~7'b0100000;
7 : seg = ~7'b0001111;
8 : seg = ~7'b0000000;
9 : seg = ~7'b0000100;
//switch off 7 segment character when the bcd digit is not a decimal number.
default : seg = 7'b0000000;
endcase
end
else begin
seg = 7'b0000000;
end
end
end
endmodule

```

### 5.3 Part I: behavioral simulation of decoder code test bench

Listing 2: Testbanche decoder part I.

```

`timescale 1ns / 1ns
////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////
// Company:

```



```

// Engineer:
//
// Create Date: 12.09.2018 19:51:24
// Design Name:
// Module Name: decoder_tb
// Project Name:
// Target Devices:
// Tool Versions:
// Description:
//
// Dependencies:
//
// Revision:
// Revision 0.01 - File Created
// Additional Comments:
//
////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////

module decoder_tb;

reg SW0, SW1, BTNO, clk, rst;
wire [6:0] seg;
wire digit;

decoder X1(SW0, SW1, BTNO, clk, rst, seg, digit); // Initialistaion

initial
begin
SW0 = 0;
SW1 = 0;
BTNO = 0;
clk = 0;
rst = 0;
end

always #10 clk = ~clk;

initial begin
#100;
rst = 1; #50;
rst = 0; #50;
SW0 = 0; SW1 = 0; BTNO = 1; #100;
BTNO = 0; #50;
rst = 0; SW0 = 1; SW1 = 0; BTNO = 1; #100;
BTNO = 0; #50;
rst = 0; SW0 = 0; SW1 = 1; BTNO = 1; #100;
BTNO = 0; #50;
rst = 0; SW0 = 1; SW1 = 1; BTNO = 1; #100;
BTNO = 0;

end
initial #1000 $finish;
endmodule

```