Remi de Ferrieres
X664611
CS 120A Section 021
Lab 3 – Programming Combinatorial Logic on the Basys FPGA Board
Yann Abou Jaoude
X663581

OVERVIEW :

In this lab, we first took the lab_2 source code (decoder) and we applied it to the FPGA. Each combination of inputs lit up a led. In the next part we coded the Verilog module of the BCD-to-7seg along with the code that linked the inputs/outputs to the one on the basys2 board. Each combination of inputs displayed a hexadecimal digit on the basys2 board.

NEW CONCEPTS:

A field-programmable gate array (FPGA) is an integrated circuit (IC) that can be programmed in the field after manufacture. FPGAs are similar in principle to, but have vastly wider potential application than, programmable read-only memory (PROM) chips. FPGAs are used by engineers in the design of specialized ICs that can later be produced hard-wired in large quantities for distribution to computer manufacturers and end users.

BCD-to-7seg is a Display Decoder. A Display Decode is a combinational circuit which decodes and n-bit input value into a number of output lines to drive a display. 7-segment LED type displays provide a very convenient way of displaying information or digital data in the form of numbers, letters or even alpha-numerical characters. In our case we had to display the hexadecimal digit from 1 to 15 (F).

ANALYSIS:

Truth table of the 3x8 decoder :

E	A	B	C	D0	D1	D2	D3	D4	D5	D6	D7
0	x	x	x	0	0	0	0	0	0	0	0
1	0	0	0	1							
1	0	0	1		1						
1	0	1	0			1					
1	0	1	1				1				
1	1	0	0					1			
1	1	0	1						1		
1	1	1	0							1	
1	1	1	1								1

Equations:

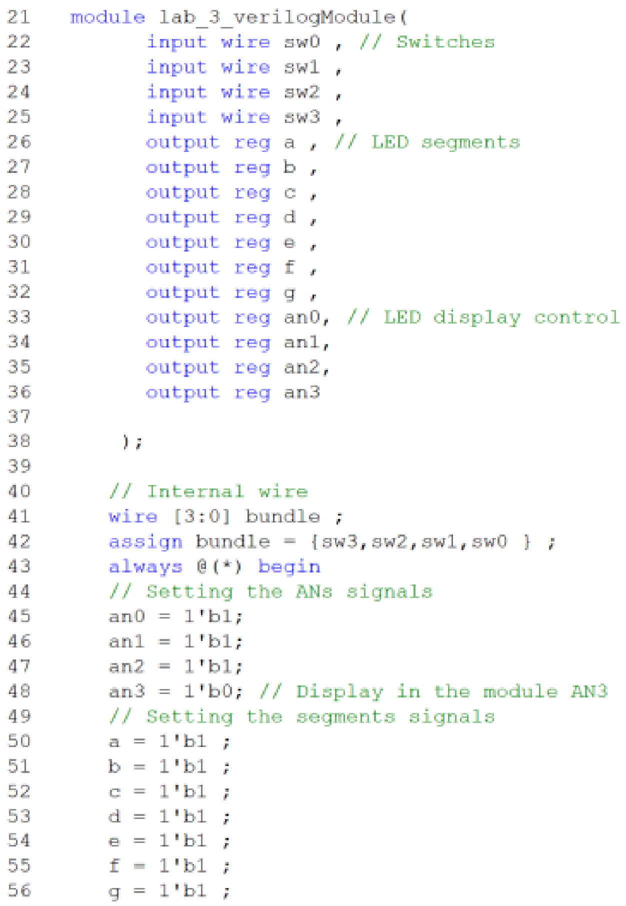
D0 = EA’B’C’	D4 = EAB’C’
D1 = EA’B’C	D5 = EAB’C
D2 = EA’BC’	D6 = EABC’
D3 = EA’BC	D7 = EABC

Truth table of the BCD-to-7seg :

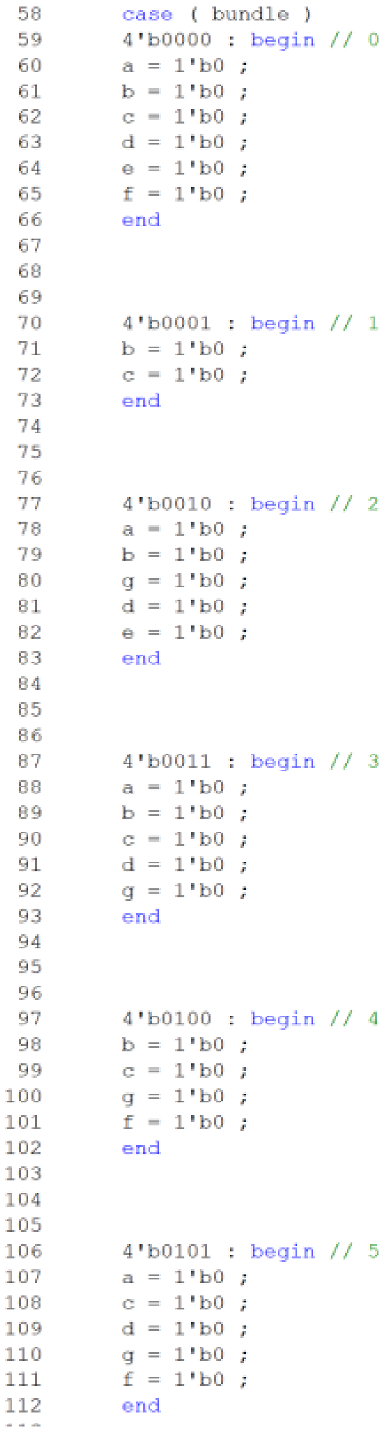
Sw0	Sw1	Sw2	Sw3	a	b	c	d	e	f	g
0	0	0	0	1	1	1	1	1	1	0
0	0	0	1	0	1	1	0	0	0	0
0	0	1	0	1	1	0	1	1	0	1
0	0	1	1	1	1	1	1	0	0	1
0	1	0	0	0	1	1	0	0	1	1
0	1	0	1	1	0	1	1	0	1	1
0	1	1	0	1	0	1	1	1	1	1
0	1	1	1	1	1	1	0	0	0	0
1	0	0	0	1	1	1	1	1	1	1
1	0	0	1	1	1	1	1	0	1	1

RECORDS:

Source code Verilog module BCD-to-7seg :

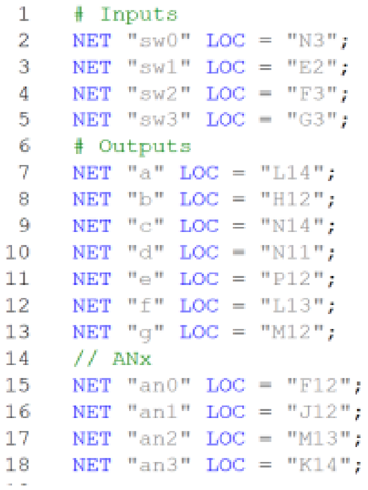


```
module lab_3_verilogModule(
      input wire sw0 , // Switches
      input wire sw1 ,
      input wire sw2 ,
      input wire sw3 ,
      output reg a , // LED segments
      output reg b ,
      output reg c ,
      output reg d ,
      output reg e ,
      output reg f ,
      output reg g ,
      output reg an0, // LED display control
      output reg an1,
      output reg an2,
      output reg an3

    );

   // Internal wire
   wire [3:0] bundle ;
   assign bundle = {sw3,sw2,sw1,sw0 } ;
   always @(*) begin
   // Setting the ANs signals
   an0 = 1'b1;
   an1 = 1'b1;
   an2 = 1'b1;
   an3 = 1'b0; // Display in the module AN3
   // Setting the segments signals
   a = 1'b1 ;
   b = 1'b1 ;
   c = 1'b1 ;
   d = 1'b1 ;
   e = 1'b1 ;
   f = 1'b1 ;
   g = 1'b1 ;
```



```
case ( bundle )
4'b0000 : begin // 0
a = 1'b0 ;
b = 1'b0 ;
c = 1'b0 ;
d = 1'b0 ;
e = 1'b0 ;
f = 1'b0 ;
end



4'b0001 : begin // 1
b = 1'b0 ;
c = 1'b0 ;
end



4'b0010 : begin // 2
a = 1'b0 ;
b = 1'b0 ;
g = 1'b0 ;
d = 1'b0 ;
e = 1'b0 ;
end



4'b0011 : begin // 3
a = 1'b0 ;
b = 1'b0 ;
c = 1'b0 ;
d = 1'b0 ;
g = 1'b0 ;
end



4'b0100 : begin // 4
b = 1'b0 ;
c = 1'b0 ;
g = 1'b0 ;
f = 1'b0 ;
end



4'b0101 : begin // 5
a = 1'b0 ;
c = 1'b0 ;
d = 1'b0 ;
g = 1'b0 ;
f = 1'b0 ;
end
```

Implementation constraint file :



```
# Inputs
NET "sw0" LOC = "N3";
NET "sw1" LOC = "E2";
NET "sw2" LOC = "F3";
NET "sw3" LOC = "G3";
# Outputs
NET "a" LOC = "L14";
NET "b" LOC = "H12";
NET "c" LOC = "N14";
NET "d" LOC = "N11";
NET "e" LOC = "P12";
NET "f" LOC = "L13";
NET "g" LOC = "M12";
// ANx
NET "an0" LOC = "F12";
NET "an1" LOC = "J12";
NET "an2" LOC = "M13";
NET "an3" LOC = "K14";
```

DISCUSSION:

The system works according to provided specifications . We had difficulties to navigate in the software. To improve the system, we can do a decimal rather than hexadecimal display using 2 digit We could also extend the system to display more digit

CONCLUSION:

Today we have learned to design a decoder. It is an example of a system using combinatorial logic. We have done a module that we can now use in any of our project. Every time we may have 4 inputs, we will be able to visualize them through a7 segment display.

QUESTIONS:

1. Can there be a difference in logical behavior between the intended logic entered and simulated and, the logic actually synthesized for FPGA? Why? Yes , there can be a difference. Our system is a simulation, witch use several gates to simulate one. Moreover, component are not perfect. This causes delay that can interfere with the logic 2. Why do we need a configuration file? For the verilog module, input and output are just variables. We need a configuration file to link them with real port of the FPGA. 3. Is there a functional difference in circuitry between Lab 1, Part 3andBasys board for this particular application? There is no functional difference. Both system have the same input and output. 4.What must be done in order to use switches SW3 and SW7 instead of SW0 and SW1? How about using LED5 instead of LED0? We just have to replace the port names in the .ucf file : For the switches, we can change P11 and L3 to B4 and N3. For the LEDwe can change M5 to N4.