COL215
Digital Logic & System Design

Assignment 2

Submitted by:

Dhairya Kuchhal	(2024CS50396)
Siddhant Agrawal	(2024CS50469)

August 13, 2025

1 Task

The objective of this lab was to design, simulate, and implement a Verilog module that displays decimal digits (0–9) on the seven-segment display of the Basys-3 FPGA board. The inputs are provided via the board's 10 slide switches (SW0 – SW9), each corresponding to a specific digit. Only the rightmost display digit (AN0) is activated, while the other digits remain off. The module uses logic while implementing arrays to determine which segments (a–g) should be turned on for a given input.

2 Design Decisions

We kept a minimal design for correctness and clarity. Firstly, we only enabled the AN0 in order to reduce redundancy and show anode control. This was done using using an active - low logic in array form.

```
        AN = 4’b1110
```

Secondly, each input switch was mapped to represent the corresponding index of the array SW0 – SW9 As per the problem statement we also implemented the priority logic by using the if-else statements making sure that the higher priority is allocated to state of higher-number switch. Display was initialised to an all-off state. Additionally in verilog, the seven-segment display on Basys-3 uses active-low control; a 0 turns a segment ON, a 1 turns it OFF. Finally, The segment outputs are generated in an always @(*) block, ensuring instantaneous updates when the switch inputs change

```
    always @(*) begin
        seg = 7'b1111111; // code excerpt
        if (sw[9]) seg = 7'b0010000;
        else if(sw[8]) seg = 7'b0000000;
        else if(sw[7]) seg = 7'b1111000;
        else if(sw[6]) seg = 7'b0000010;
        else if(sw[5]) seg = 7'b0010010;
        else if(sw[4]) seg = 7'b0011001;
        else if(sw[3]) seg = 7'b0110000;
        else if(sw[2]) seg = 7'b0100100;
        else if(sw[1]) seg = 7'b1111001;
        else if(sw[0]) seg = 7'b1000000;
        end
    endmodule
```

2.1 Simulation using Testbench

We wrote a dedicated testbench module (seven_tb.v) for simulation. Inputs were toggled at 10ns time intervals (Inertial delay) and showed each possible case of the switches lighting up. We also ended up using Used $display to print a header and $monitor to track real-time changes in sw, seg, and an. All this was done to verify the functionality in simulation before hardware implementation.

```
module seven_tb();
  reg [9:0] sw;
  wire [6:0] seg;
  wire [3:0]an;
  Seven_segment_display uut(
    .sw(sw),
    .seg(seg),
    .an(an)
  );
  initial begin
    $display("Time(ns)\tSwitches\tSegments\tAnodes");
    $monitor("%0t\t%b\t%b\t%b", $time, sw, seg, an);

    sw = 10'b0000000001;
    #10 sw = 10'b0000000010;
    #10 sw = 10'b0000000100;
    #10 sw = 10'b0000001000;
    #10 sw = 10'b0000010000;
    #10 sw = 10'b0000100000;
    #10 sw = 10'b0001000000;
    #10 sw = 10'b0010000000;
    #10 sw = 10'b0100000000;
    #10 sw = 10'b1000000000;
    #10 sw = 10'b0100010001;
    #20 sw = 10'b0000000000;
    $finish;
   end
endmodule
```

2.2 Mapping of Pins on Basys3

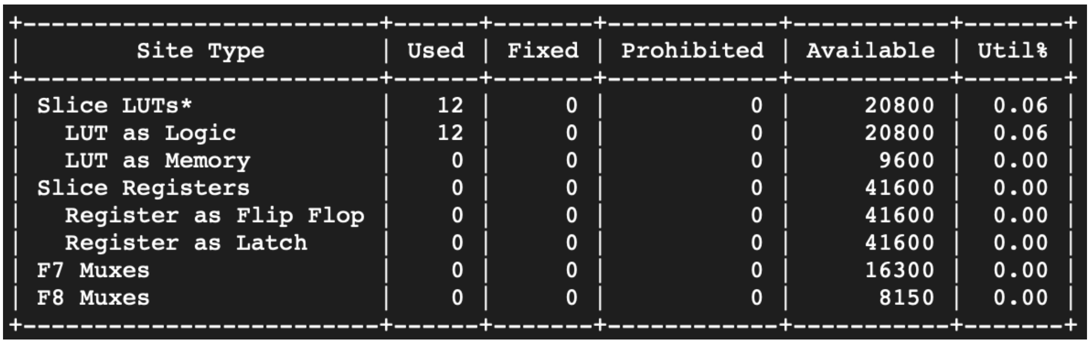
The constraint file (basys3.xdc) was edited to bind logical inputs and outputs to physical switches and the display segments in a user-friendly, simple left to right manner. The following table summarizes the mapping:

Signal Type	Logical Name	FPGA Pin
Input	$sw[0]$	V17
Input	$sw[1]$	V16
Input	$sw[2]$	W16
Input	$sw[3]$	W17
Input	$sw[4]$	W15
Input	$sw[5]$	V15
Input	$sw[6]$	W14
Input	$sw[7]$	W13
Input	$sw[8]$	V2
Input	$sw[9]$	T3
Output	$an[0]$	U2
Output	$an[1]$	U4
Output	$an[2]$	V4
Output	$an[3]$	W4
Output	$seg[0]$	W7
Output	$seg[1]$	W6
Output	$seg[2]$	U8
Output	$seg[3]$	V8
Output	$seg[5]$	U5
Output	$seg[6]$	V5
Output	$seg[6]$	U7

Table 1: Mapping of inputs and outputs on Basys 3 FPGA board

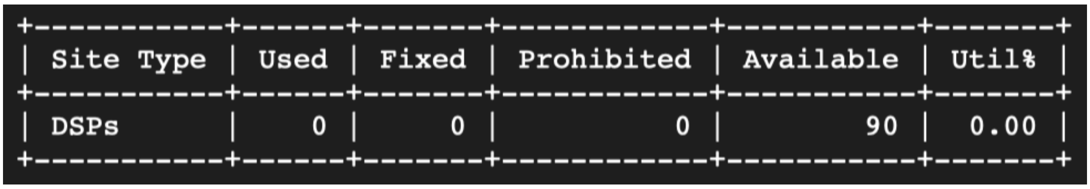
3 Synthesis Report

The following tables show the main resource counts. Other details are present in the synthesis report.



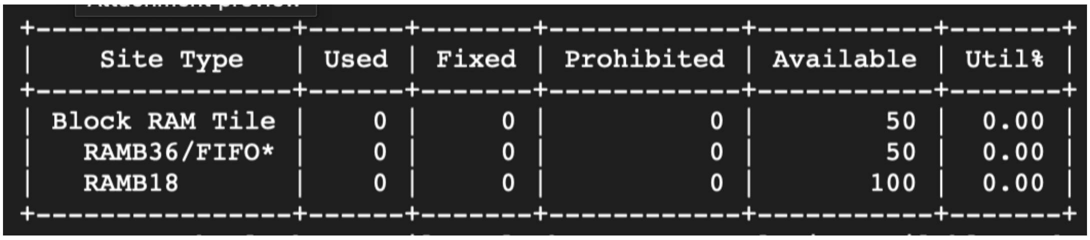
```
+-------------------------+------+-------+------------+-----------+-------+
|        Site Type        | Used | Fixed | Prohibited | Available | Util% |
+-------------------------+------+-------+------------+-----------+-------+
| Slice LUTs*             |   12 |     0 |          0 |     20800 |  0.06 |
|   LUT as Logic          |   12 |     0 |          0 |     20800 |  0.06 |
|   LUT as Memory         |    0 |     0 |          0 |      9600 |  0.00 |
| Slice Registers         |    0 |     0 |          0 |     41600 |  0.00 |
|   Register as Flip Flop |    0 |     0 |          0 |     41600 |  0.00 |
|   Register as Latch     |    0 |     0 |          0 |     41600 |  0.00 |
| F7 Muxes                |    0 |     0 |          0 |     16300 |  0.00 |
| F8 Muxes                |    0 |     0 |          0 |      8150 |  0.00 |
+-------------------------+------+-------+------------+-----------+-------+
```

Figure 1: Flip-Flops and LUT



```
+-----------+------+-------+------------+-----------+-------+
| Site Type | Used | Fixed | Prohibited | Available | Util% |
+-----------+------+-------+------------+-----------+-------+
| DSPs      |    0 |     0 |          0 |        90 |  0.00 |
+-----------+------+-------+------------+-----------+-------+
```

Figure 2: DSP usage

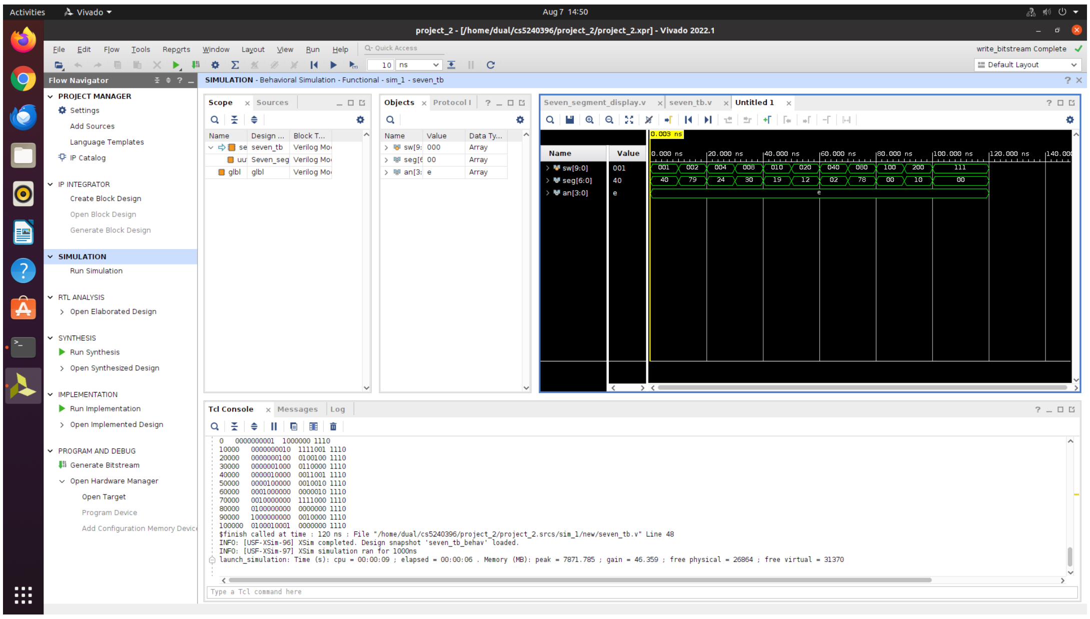


```
+---------------+------+-------+------------+-----------+-------+
|   Site Type   | Used | Fixed | Prohibited | Available | Util% |
+---------------+------+-------+------------+-----------+-------+
| Block RAM Tile |    0 |     0 |          0 |        50 |  0.00 |
|   RAMB36/FIFO* |    0 |     0 |          0 |        50 |  0.00 |
|   RAMB18       |    0 |     0 |          0 |       100 |  0.00 |
+---------------+------+-------+------------+-----------+-------+
```

Figure 3: BRAM usage

4 Simulation Snapshots

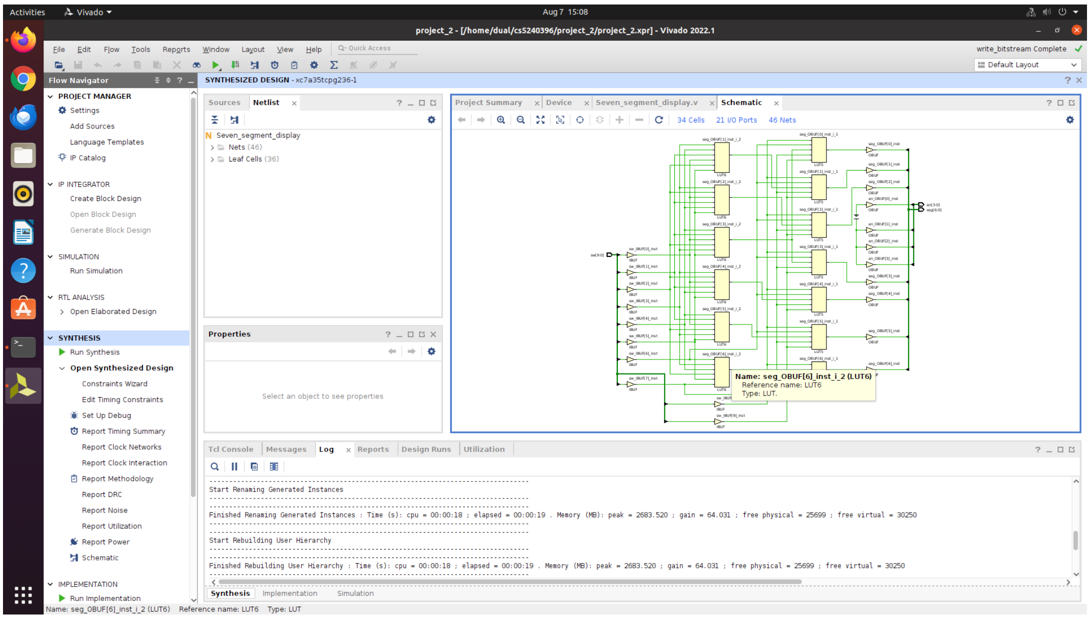
Figure shows the behavioural simulation results on running seven_tb.v file



```
0   0000000001  1000000 1110
10000   0000000010  1111001 1110
20000   0000000100  0100100 1110
30000   0000001000  0110000 1110
40000   0000010000  0011001 1110
50000   0000100000  0010010 1110
60000   0001000000  0000010 1110
70000   0010000000  1111000 1110
80000   0100000000  0000000 1110
90000   1000000000  0010000 1110
100000  0100010001  0000000 1110
$finish called at time : 120 ns : File "/home/dual/cs5240396/project_2/project_2.srcs/sim_1/new/seven_tb.v" Line 48
INFO: [USF-XSim-96] XSim completed. Design snapshot 'seven_tb_behav' loaded.
INFO: [USF-XSim-97] XSim simulation ran for 1000ns
launch_simulation: Time (s): cpu = 00:00:09 ; elapsed = 00:00:06 . Memory (MB): peak = 7871.785 ; gain = 46.359 ; free physical = 26864 ; free virtual = 31370
```

Figure 4: Simulation waveform showing the transitions of the array that controls the seven segment display outputs

5 Generated Schematics



```
---------------------------------------------------------------------------------
Start Renaming Generated Instances
---------------------------------------------------------------------------------
---------------------------------------------------------------------------------
Finished Renaming Generated Instances : Time (s): cpu = 00:00:18 ; elapsed = 00:00:19 . Memory (MB): peak = 2683.520 ; gain = 64.031 ; free physical = 25699 ; free virtual = 30250
---------------------------------------------------------------------------------
---------------------------------------------------------------------------------
Start Rebuilding User Hierarchy
---------------------------------------------------------------------------------
---------------------------------------------------------------------------------
Finished Rebuilding User Hierarchy : Time (s): cpu = 00:00:18 ; elapsed = 00:00:19 . Memory (MB): peak = 2683.520 ; gain = 64.031 ; free physical = 25699 ; free virtual = 30250
---------------------------------------------------------------------------------
```

Figure 5: Generated Schematic design

6 Conclusion

The designed module successfully displayed decimal digits on a single digit of the seven-segment display using switch inputs. The simulation matched hardware behavior, confirming correct .xdc mapping and logic implementation. This assignment reinforced understanding of:

- Array combinational logic design in Verilog
- Active-low seven-segment control
- Priority-based conditional statements
- FPGA pin mapping using .xdc files