

Lab 4: 7-Segment Display SOLUTIONS

Digital Design and Computer Architecture: RISC-V Edition (Harris & Harris, Elsevier © 2021)

1. Please indicate how many hours you spent on this lab. This will be helpful for calibrating the workload for next time the course is taught.
2. Completed Table 1.

Table 1. Truth table for 7-segment display decoder

Hexadecimal Digit	Inputs				Outputs							(in hex)
	D ₃	D ₂	D ₁	D ₀	S _g	S _f	S _e	S _d	S _c	S _b	S _a	
0	0	0	0	0	1	0	0	0	0	0	0	40
1	0	0	0	1	1	1	1	1	0	0	1	79
2	0	0	1	0	0	1	0	0	1	0	0	24
3	0	0	1	1	0	1	1	0	0	0	0	30
4	0	1	0	0	0	0	1	1	0	0	1	19
5	0	1	0	1	0	0	1	0	0	1	0	12
6	0	1	1	0	0	0	0	0	0	1	0	02
7	0	1	1	1	1	1	1	1	0	0	0	78
8	1	0	0	0	0	0	0	0	0	0	0	00
9	1	0	0	1	0	0	1	1	0	0	0	18
A	1	0	1	0	0	0	0	1	0	0	0	08
B	1	0	1	1	0	0	0	0	0	1	1	03
C	1	1	0	0	0	1	0	0	1	1	1	27
D	1	1	0	1	0	1	0	0	0	0	1	21
E	1	1	1	0	0	0	0	0	1	1	0	06
F	1	1	1	1	0	0	0	1	1	1	0	0E

3. SystemVerilog modules: sevenseg.sv and sevenseg_wrapper.sv.

sevenseg.sv:

```
module sevenseg(input  logic [3:0] data,
                output logic [6:0] segments);
```

```
    always_comb
    case (data)
        //Sg - Sa
        4'h0: segments = 7'b1000000;
        4'h1: segments = 7'b1111001;
        4'h2: segments = 7'b0100100;
        4'h3: segments = 7'b0110000;
        4'h4: segments = 7'b0011001;
        4'h5: segments = 7'b0010010;
```

```

    4'h6: segments = 7'b0000010;
    4'h7: segments = 7'b1111000;
    4'h8: segments = 7'b0000000;
    4'h9: segments = 7'b0011000;
    4'hA: segments = 7'b0001000;
    4'hB: segments = 7'b0000011;
    4'hC: segments = 7'b0100111;
    4'hD: segments = 7'b0100001;
    4'hE: segments = 7'b0000110;
    4'hF: segments = 7'b0001110;
endcase
endmodule

```

sevensseg_wrapper.sv:

```

module sevensseg_wrapper(input  logic [3:0] SW,
                        output logic [6:0] HEX0);

    sevensseg sseg1(SW, HEX0);
endmodule

```

4. Testbench module and test vector files (testbench_sevensseg.sv and sevensseg.txt).

testbench_sevensseg.sv:

```

module testbench_sevensseg();
    logic      clk, reset;
    logic [6:0] segments, segments_expected;
    logic [3:0] data;
    logic [31:0] vectornum, errors;
    logic [10:0] testvectors[10000:0];

    // instantiate device under test
    sevensseg dut(data, segments);

    // generate clock
    always
    begin
        clk = 1; #5; clk = 0; #5;
    end

    // at start of test, load vectors

```

```

// and pulse reset
initial
begin
    $readmemh("sevenseg.txt", testvectors);
    vectornum = 0; errors = 0;
    reset = 1; #27; reset = 0;
end

// apply test vectors on rising edge of clk
always @(posedge clk)
begin
    #1; {segments_expected, data} = testvectors[vectornum];
end

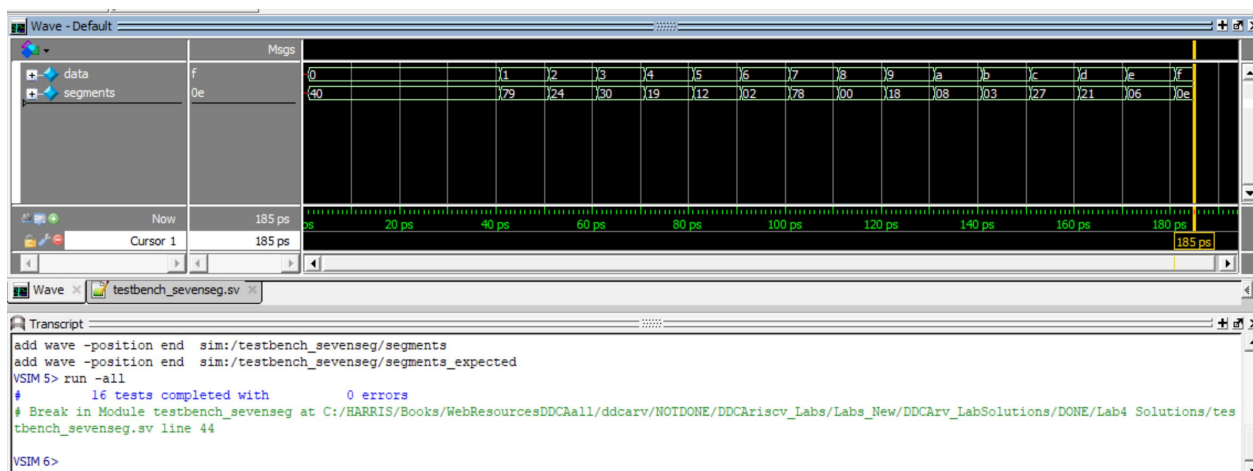
// check results on falling edge of clk
always @(negedge clk)
begin
    if (~reset) begin // skip during reset
        if (segments != segments_expected) begin // check result
            $display("Error: inputs = %h", data);
            $display("  outputs = %h (%h expected)", segments,
                segments_expected);
            errors = errors + 1;
        end
        vectornum = vectornum + 1;
        if (testvectors[vectornum] === 11'bx) begin
            $display("%d tests completed with %d errors",
                vectornum, errors);
            $stop;
        end
    end
end
endmodule

```

sevensseg.txt:

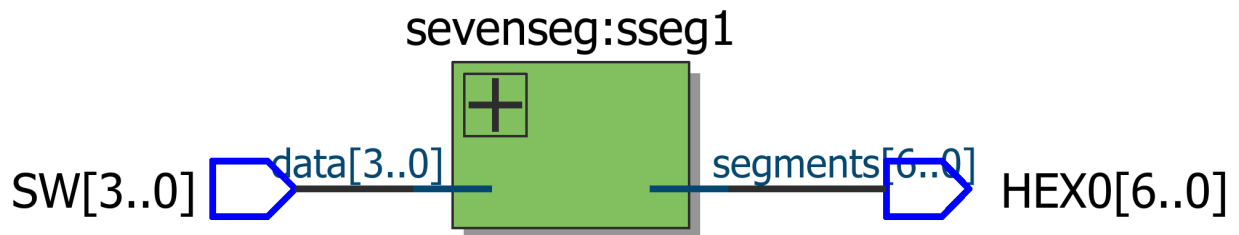
```
// segments_data
40_0
79_1
24_2
30_3
19_4
12_5
02_6
78_7
00_8
18_9
08_A
03_B
27_C
21_D
06_E
0E_F
```

5. Simulation waveforms showing the inputs (data) and outputs (segments) in that order, from top to bottom, and in hexadecimal. Do they pass your testbench and match expectations? **YES.**

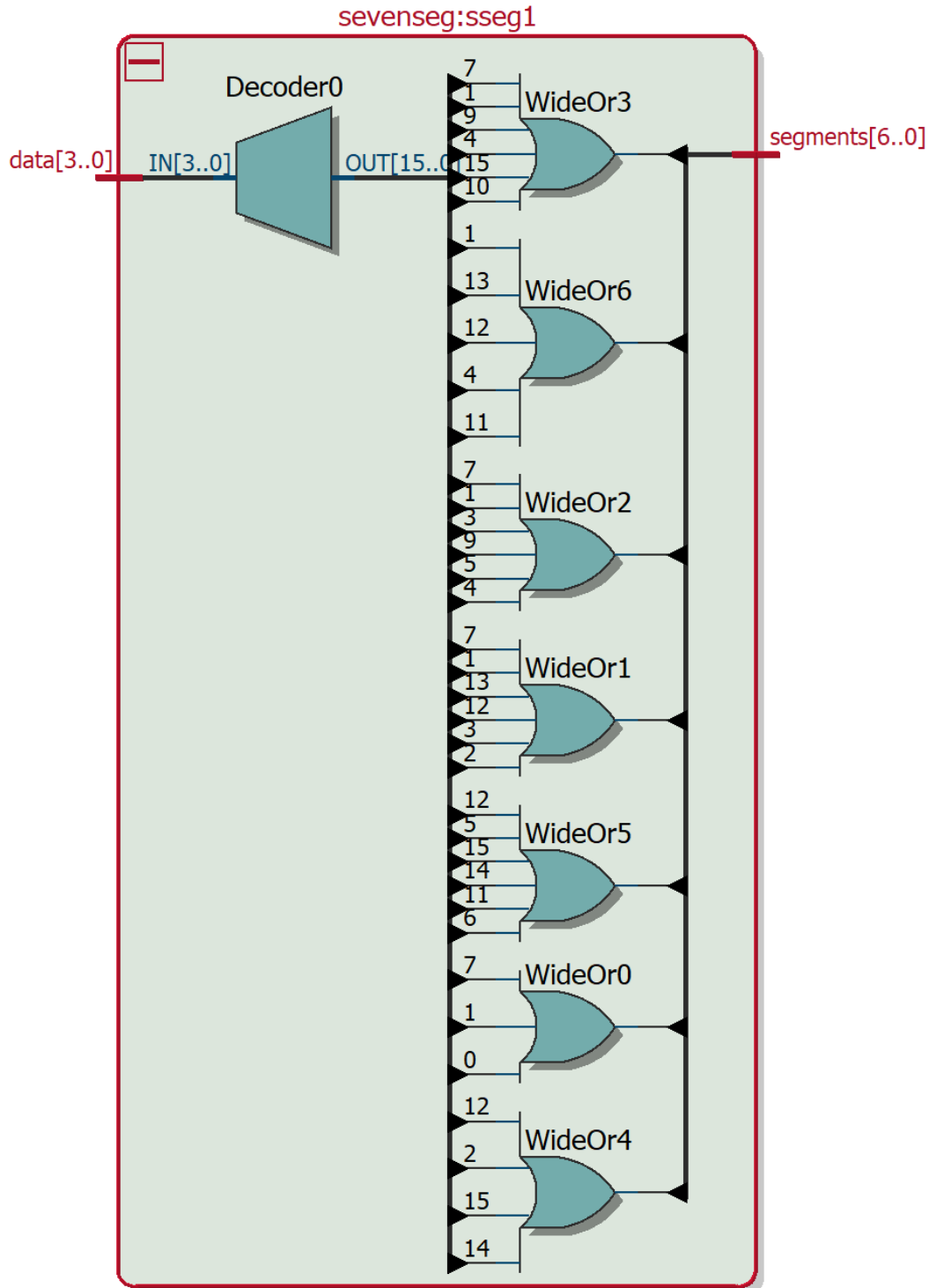


6. RTL Viewer schematics of your sevenseg_wrapper.sv and sevenseg.sv modules (two schematics). Do they match your expectation? **YES**.

sevenseg_wrapper.sv:



sevensseg.sv:



Please indicate any bugs you found in this lab manual, or any suggestions you would have to improve the lab.