

Assignment 01

COMBINATORIAL LOGIC

EMBEDDED SYSTEMS 3

FACHHOCHSCHULE VORARLBERG
MASTER MECHATRONICS

Eingereicht bei

Dr. Andrè Mitterbacher

Vorgelegt von

ROMAN PASSLER

DORNBIRN, 13.12.2017

Inhaltsverzeichnis

A	bbild	ungsverzeichnis	H
Ta	abelle	enverzeichnis	III
Li	\mathbf{sting}	SS .	ΙV
1	Sieb	pensegmentanzeige	1
	1.1	Einleitung	1
	1.2	Implementierung	1
	1.3	Test Bench	2
	1.4	Simulationsscript	4
	1.5	Transkript und Waveform Window	5
	1.6	Vor- und Nachteile der Implementierung	8

Abbildungsverzeichnis

1.1	Waveform	Window																5	

Tabellenverzeichnis

Listings

1.1	Implementierung	1
1.2	Testbench für die Siebensegmentanzeige	2
1.3	Simulationsscript	4
1.4	Commandline Output A	6
1.5	Commandline Output B	6
1.6	Commandline Output C	7
1.7	Commandline Output D	7

1 Siebensegmentanzeige

1.1 Einleitung

In der ersten Aufgabe soll eine Siebensegmentanzeige mit Verilog implementiert werden. Die Anforderungen sind:

- die Binärzahl korrekt den Ausgängen zuweisen ("hex").
- "hex n" ist die negierte Version von "hex"¹.
- Die Zahlen 10 bis 15 nicht vergessen!

Zusätzlich zu den Anforderungen muss die Implementierung getestet und verifiziert werden. Dies bedeutet, dass alle Eingänge stimuliert und alle Signale im "Wave Window" angezeigt werden. Außerdem sollte der aktuelle Zustand der Ein- und Ausgänge als Lookup-Tabelle in der Modelsim-Konsole ("Transkript-Fenster") angezeigt werden.

1.2 Implementierung

Für die Implementierung wurde die Verhaltensbeschreibung verwendet, da die "Gate-Primitive" nicht empfohlen wurde und die kontinuierliche Zuweisung zu umständlich wäre. Der Code kann in Listing 1.1 nachgelesen werden.

```
Project: Seven Segment
    Purpose: Implement a Seven segment BCD
    Author: rpa2306
    Version: 00, 11.12.2017
             F
                                 В
10
                       G
11
13
                                 \mathbf{C}
14
             Е
15
16
17
                       \mathbf{D}
19
20
    module sevenseg(
```

¹Ist hilfreich für, active low" Anzeigen.

```
Descibe the IO of the module
22
                 logic
                           [3:0] bin,
        input
23
        output
                 logic
                            6:0
                                 hex,
24
                           [6:0] hex n
25
        output
                 logic
        // no comma after the last entry!
26
   );
27
28
29
30
   always comb begin
31
        case (bin)
32
                 4'h0 : hex = 7'b011_1111;
33
                 4'h1 : hex = 7'b000_0110;
34
                 4'h2 : hex = 7'b101_1011;
35
                 4'h3 : hex = 7'b100_1111;
36
                 4'h4 : hex = 7'b110_0110;
37
                 4'h5 : hex = 7'b110_1101;
38
                 4'h6 : hex = 7'b111_1101;

4'h7 : hex = 7'b000_0111;
39
40
                 4'h8 : hex = 7'b111^{-}1111;
41
                 4'h9 : hex = 7'b110_1^-1111;
42
                 4'ha : hex = 7'b111_0111;
                                                  // A
43
                 4'hb : hex = 7'b111_1'1100;
                                                  // B
44
                 4'hc : hex = 7'b011 1001;
45
                                                  // D
                 4'hd : hex = 7'b101 1110;
46
                                                  // E
                 4'he : hex = 7'b111 1001;
47
                                                  // F
                 4'hf : hex = 7'b111 0001;
48
                 default: hex = 7'b000 000;
49
50
        endcase
        assign hex_n = ~hex;
51
52
   end
53
   endmodule
54
```

Listing 1.1: Implementierung

1.3 Test Bench

Das Testkonzept ist eine for-Schleife, die abhängig von der Bit-Länge bis zum Maximum iteriert, siehe Codezeile 44 in Listing 1.3. Die Iterationsvariable wird in jedem Iterationsschritt "bin" zugewiesen. Dabei wird die tatsächliche Anzahl der Ein- und Ausgänge in binärer Darstellung im "Transcript" der Modelsim-Software dargestellt. Darüber hinaus wird die Siebensegmentanzeige im "Transcript" simuliert, siehe Codezeile 51 bis 91.

```
(1) Create wires to connect the DUT
13
        Like footprints for an IC on a PCB
14
15
             [3:0]
   logic
                           bin;
16
   logic
             6:0
                           hex;
17
             6:0
                           hex n;
18
   logic
19
        (2) Create an instance of the DUT
20
21
22
                           DUT
23
   sevenseg
   // name of the moule name of this instance of the module
24
25
        IO connection syntax:
26
27
        . pin name
                          (track name),
        .bin
28
                                (bin),
        . hex
                                (hex),
29
        .hex n
                                (hex n)
30
31
32
   );
33
34
        (3) Create stimuli for all inputs
35
36
37
38
   initial begin
   // All initial blocks are started at simulation time = 0. Only used in
39
        the test bench
    // Execution is line by line -> it 's software!!!
40
        $display("-
41
                                                    ");
        $display(" tb_sevenseg started
42
                                                    -");
        $display("-
43
        for (int i =0; i < $bits(bin)<<2;i++) begin //
44
             bin = i;
45
             \#100\,\mathrm{ns};
                           // wait for 100 ns
46
             $display("");
47
             $display("bin\thex\thex n");
48
             \frac{\text{sdisplay}(\text{"\%b} t\%b n", bin, hex, hex n)}{\text{tmplay}(\text{"\%b} t\%b n", bin, hex, hex n)};
49
50
             /* the next lines are only for nice displaying*/
             if (hex[0]) begin /
51
                  $display("%s",'HORIZONTALA);
52
             end else begin
53
                  if (hex [5]) begin
54
                                             ");
                       $ write ( " \ t#
                  end else begin
56
                       $write("\t
                                             ");
57
                  end
58
                  if (hex [1]) begin
59
                       $write("#");
60
61
                  end
                  display("");
62
63
             end
             for (int j = 0; j\!<\!2; j\!+\!\!+\!\!+\!\!) begin
64
                  for (int n = 0; n < 4; n++) begin
65
                       if (hex[4] \&\& j=1 || hex[5] \&\& j=0) begin
66
                           $write("\t|");
67
                      end else begin
68
                           $write("\t ");
69
                       end
70
                       $write("
71
                       if (hex[1] \&\& j=0 || hex[2] \&\& j=1) begin
72
```

```
$display("|");
73
                       end else begin
74
                             $display("");
75
76
                       end
                  end
77
                      (\text{hex} [6] \&\& j=0 \mid | \text{hex} [3] \&\& j=1) \text{ begin}
78
                        $\display(\"\s",'HORIZONTALA);
79
80
                  end else begin
                        if (hex[4] \&\& j=1 || hex[5] \&\& j=0) begin
81
                                                    ");
                             $ write ( " \ t#
82
                       end else begin
83
                                                    ");
                             $write("\t
84
                       end
85
                        if (hex[1] \&\& j=0 || hex[2] \&\& j=1) begin
86
                             $write("#");
87
88
                        $display("");
89
                  end
90
             end //
91
        end
92
         $display("-
93
                                                       ");
         $display("
                         sb_sevenseg finished
94
         $display("-
95
    end
96
97
    endmodule
```

Listing 1.2: Testbench für die Siebensegmentanzeige

1.4 Simulationsscript

In Listing 1.3 ist das Simulationsscript dargestellt. Es beinhaltet dieselben Befehle wie in der letzten Lehrveranstaltung, natürlich angepasst an die Siebensegmentanzeige.

```
# Create simulation environment
   vlib work
   vmap work work
   # Compile desing files -> use file names
   vlog ../src/sevenseg.sv
  # Compile the test bench
   vlog tb sevenseg.sv
   # Init simulation -> use module name
   vsim tb sevenseg
  # -r recursive
   log -r *
   do wave_tb_sevenseg.tcl
13
14
   # Run simulation
15
  run - all
16
   # run 100us
17
   # Show results
18
   view wave
```

Listing 1.3: Simulationsscript

1.5 Transkript und Waveform Window

In Abbildung 1.1 können die Ausgänge "hex" und "hex_n" im Wavefomr Window überprüft werden. In Listings 1.4 bis 1.7 ist der Commandline Output abgebildet, welcher ebenfalls zeigt, dass der Test erfolgreich war.

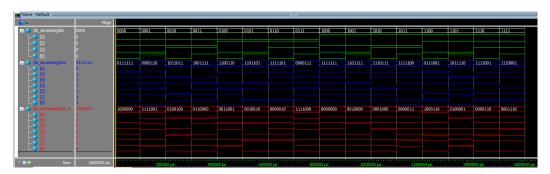
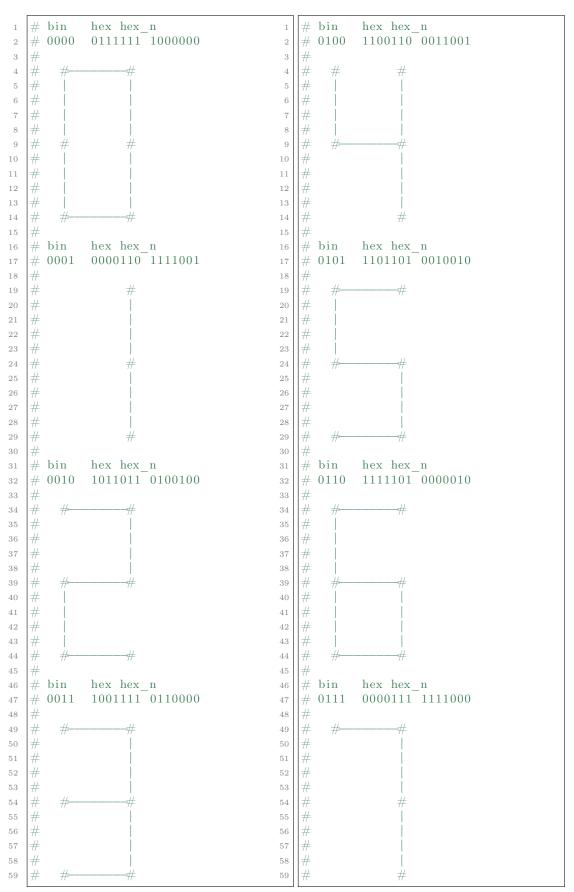
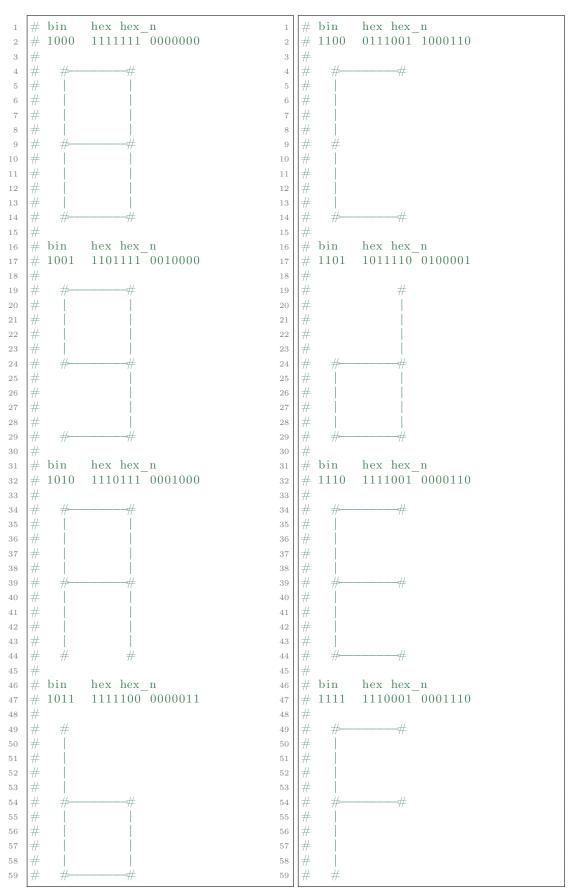


Abbildung 1.1: Waveform Window Quelle: eigene Ausarbeitung



Listing 1.4: Commandline Output A

Listing 1.5: Commandline Output B



Listing 1.6: Commandline Output C

Listing 1.7: Commandline Output D

1.6 Vor- und Nachteile der Implementierung

Folgend sind die Vor- und Nachteile der Implementierung gelistet:

Vorteile

- Keine Counter
- einfach

Nachteile

• Mögliche Verzögerungen, da die Implementierung asynchron ist.