FH Vorarlberg
University of Applied Sciences

Assignment 05

Projektzusammenfassung

Embedded Systems 3

Fachhochschule Vorarlberg

Master Mechatronics

Eingereicht bei

Dr. Andrè Mitterbacher

Vorgelegt von

Roman Passler

Dornbirn, 26.01.2018

Inhaltsverzeichnis

Abbildungsverzeichnis

Tabellenverzeichnis

Listings

1 DAC

1.1 Einleitung

Um die entwickelte Logik in ein reales System einzubetten, müssen einige Aufgaben erledigt werden:

- Geräteauswahl
- Pinbelegung
- Fertige Module (IP) hinzufügen
- Toplevel-Routing
- Top-Level-Simulation
- Synthese des Projekts in die Zielhardware
- Banküberprüfung
- Validierung (wenn möglich)

1.2 Top-Level Design

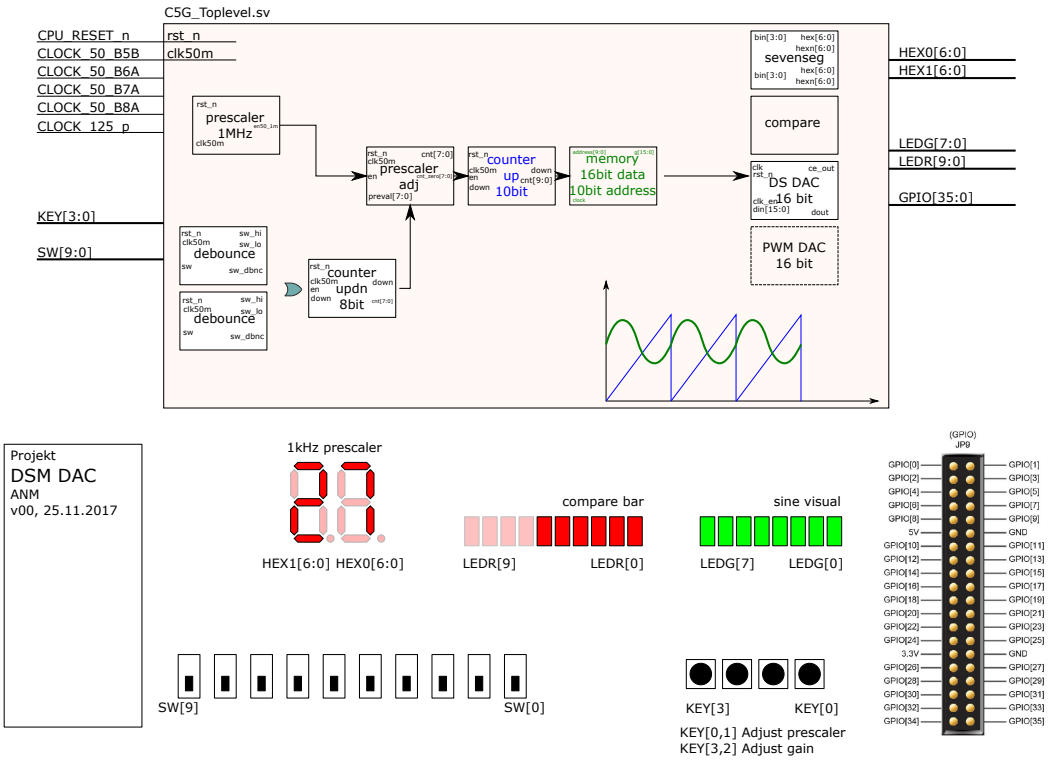

Abbildung 1.1: Top-Level Design
Quelle: eigene Ausarbeitung

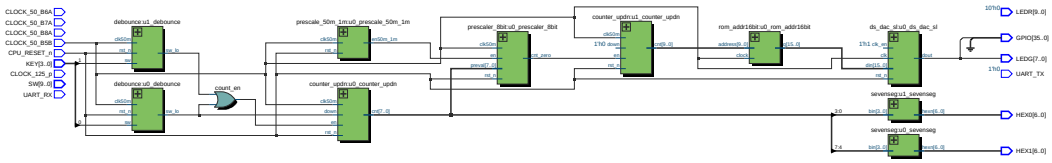

Abbildung 1.2: Top-Level Design
Quelle: eigene Ausarbeitung

1.3 Toplevel Simulation

```
1  /* -----------------------------------------------------
2  Project : DS DAC
3  Purpose : Toplevel delta sigma DAC
4  Author  : ANM
5  Date    : 25.11.2017
6  ----------------------------------------------------- */
7
8  module toplevel_c5g_led_switch_7segx2_gpio_uart(
9
10
11
12     ////////////// CLOCK ///////////
```

```
    input   logic                   CLOCK_125_p,
    input   logic                   CLOCK_50_B5B,
    input   logic                   CLOCK_50_B6A,
    input   logic                   CLOCK_50_B7A,
    input   logic                   CLOCK_50_B8A,

    ///////////// LED /////////
    output  logic       [7:0]       LEDG,
    output  logic       [9:0]       LEDR,

    ///////////// KEY /////////
    input   logic                   CPU_RESET_n,
    input   logic       [3:0]       KEY,

    ///////////// SW /////////
    input   logic       [9:0]       SW,

    ///////////// SEG7 /////////
    output  logic       [6:0]       HEX0,
    output  logic       [6:0]       HEX1,

    ///////////// Uart to USB /////////
    input   logic                   UART_RX,
    output  logic                   UART_TX,

    ///////////// GPIO, GPIO connect to GPIO Default /////////
    output  logic       [35:0]      GPIO
);



//=======================================================
//  REG/WIRE declarations
//=======================================================

    logic                           rst_n;
    logic                           clk50m;
    logic                           ds_bitstream;
    logic                           count_up;
    logic                           count_low;
    logic                           count_en;
    logic               [7:0]       cnt_8bit;
    logic                           en50m_1m;
    logic                           cnt_zero;
    logic               [9:0]       cnt_10bit;
    logic               [15:0]      analogue_sin;

//=======================================================
//  Structural coding
//=======================================================

// --- Map outputs ---

    assign UART_TX      = 1'b0;
    assign LEDG         = {8{ds_bitstream}};
    assign LEDR[9:0]    = '0;
    assign GPIO[35:1]   = '0;
    assign GPIO [0]     = ds_bitstream;

// --- Map inputs ---

```

```
74      assign                          rst_n       = CPU_RESET_n;
75      assign                          clk50m      = CLOCK_50_B5B;
76      assign                          count_en    = count_low || count_up
            ;
77
78  // --- Modules ---
79
80  prescale_50m_1m u0_prescale_50m_1m(
81              .rst_n,
82              .clk50m,
83              .en50m_1m(en50m_1m)
84  );
85
86  debounce u0_debounce(
87              .rst_n,
88              .clk50m,
89              .sw(KEY[0]),
90              .sw_hi(),
91              .sw_lo(count_low),
92              .sw_dbnc()
93  );
94
95  debounce u1_debounce(
96              .rst_n,
97              .clk50m,
98              .sw(KEY[1]),
99              .sw_hi(),
100             .sw_lo(count_up),
101             .sw_dbnc()
102 );
103
104 counter_updn #(.WIDTH (8)) u0_counter_updn(
105             .rst_n,
106             .clk50m,
107             .en(count_en),
108             .down(count_low),
109             .cnt(cnt_8bit)
110 );
111
112 prescaler_8bit  u0_prescaler_8bit(
113             .rst_n,
114             .clk50m,
115             .en(en50m_1m),
116             .preval(cnt_8bit),
117             .cnt(),
118             .cnt_zero(cnt_zero)
119 );
120
121 counter_updn #(.WIDTH (10)) u1_counter_updn(
122             .rst_n,
123             .clk50m,
124             .en(cnt_zero),
125             .down(1'b0),
126             .cnt(cnt_10bit)
127 );
128
129 rom_addr16bit u0_rom_addr16bit(
130             .address(cnt_10bit),
131             .clock(clk50m),
132             .q(analogue_sin)
133 );
```

```
134
135 ds_dac_sl u0_ds_dac_sl(
136             .clk(clk50m),
137             .rst_n,
138             .clk_en(1'b1),
139             .din(analogue_sin),
140             .ce_out(),
141             .dout(ds_bitstream)
142 );
143
144 sevenseg u0_sevenseg(
145             .bin(cnt_8bit[7:4]),
146             .hex(),
147             .hexn(HEX1[6:0])
148 );
149
150 sevenseg u1_sevenseg(
151             .bin(cnt_8bit[3:0]),
152             .hex(),
153             .hexn(HEX0[6:0])
154 );
155
156 endmodule
```

Listing 1.1: Toplevel Design

```
1  /* -----------------------------------------
2  Project : DS DAC
3  Purpose : Toplevel delta sigma DAC
4  Author  : ANM
5  Date    : 25.11.2017
6  ----------------------------------------- */
7  `timescale 10ns/10ns
8  module tb_toplevel_c5g_led_switch_7segx2_gpio_uart();
9      // (1) DUT wiring
10     ////////////// CLOCK //////////
11     logic                   CLOCK_125_p;
12     logic                   CLOCK_50_B5B;
13     logic                   CLOCK_50_B6A;
14     logic                   CLOCK_50_B7A;
15     logic                   CLOCK_50_B8A;
16
17     ////////////// LED //////////
18     logic         [7:0]     LEDG;
19     logic         [9:0]     LEDR;
20
21     ////////////// KEY //////////
22     logic                   CPU_RESET_n;
23     logic         [3:0]     KEY;
24
25     ////////////// SW //////////
26     logic         [9:0]     SW;
27
28     ////////////// SEG7 //////////
29     logic         [6:0]     HEX0;
30     logic         [6:0]     HEX1;
31
32     ////////////// Uart to USB //////////
33     logic                   UART_RX;
34     logic                   UART_TX;
35
```

```
36      //////////// GPIO; GPIO connect to GPIO Default //////////
37      logic        [35:0]       GPIO;
38
39
40      // (2) DUT instance
41      toplevel_c5g_led_switch_7segx2_gpio_uart     dut(.*);
42
43      // (3) DUT stimuli
44      logic run_sim = 1'b1;
45      int error_cnt = 0;
46      string action = "init";
47
48      // --- Clocks and Reset ---
49      initial begin : clk_gen_125m
50          CLOCK_125_p = 1'b0;
51          while (run_sim) begin
52              #4ns;
53              CLOCK_125_p = ~CLOCK_125_p;
54          end
55      end
56
57      initial begin : clk_gen_50m
58          CLOCK_50_B5B = 1'b0;
59          while (run_sim) begin
60              #10ns;
61              CLOCK_50_B5B = ~CLOCK_50_B5B;
62          end
63      end
64      assign CLOCK_50_B6A = CLOCK_50_B5B;
65      assign CLOCK_50_B7A = CLOCK_50_B5B;
66      assign CLOCK_50_B8A = CLOCK_50_B5B;
67
68      initial begin : rst_gen
69          CPU_RESET_n = 1'b0;
70          #99ns
71          CPU_RESET_n = 1'b1;
72      end
73
74      initial begin : load_memory
75          $readmemh("../fpga/Toplevel_C5G_led_switch_7segx2_gpio_uart/ip/
                mem_sine_01.txt", dut.u0_rom_addr16bit.altsyncram_component.
                m_default.altsyncram_inst.mem_data);
76      end
77
78      // --- Stimulate inputs ---
79
80
81      initial begin
82          $display("-----------------------------------");
83          $display("tb_toplevel_c5g_led_switch_7segx2_gpio_uart started."
                );
84          $display("-----------------------------------");
85          KEY = '1;
86          SW = '0;
87          UART_RX = '0;
88          #1us;
89          action="Push KEY[0] 300 times";
90          $display("\t%s",action);
91          repeat(300) begin
92              @ (negedge CLOCK_50_B5B);
93              KEY[0] = 1'b0;
```

```
94              #1us;
95              @ (negedge CLOCK_50_B5B);
96              KEY[0] = 1'b1;
97          end
98
99          #100us;
100
101         action="Push KEY[1] 555 times";
102         $display("\t%s",action);
103         repeat(555) begin
104             @ (negedge CLOCK_50_B5B);
105             KEY[1] = 1'b0;
106             #1us;
107             @ (negedge CLOCK_50_B5B);
108             KEY[1] = 1'b1;
109         end
110         CPU_RESET_n = 1'b0;
111         #99ns
112         CPU_RESET_n = 1'b1;
113         #2000us;
114
115         run_sim = 1'b0;
116         $display("-----------------------------------------");
117         $display("tb_toplevel_c5g_led_switch_7segx2_gpio_uart finished.
                ");
118         $display("-----------------------------------------");
119     end
120
121 endmodule
```

Listing 1.2: Testbench

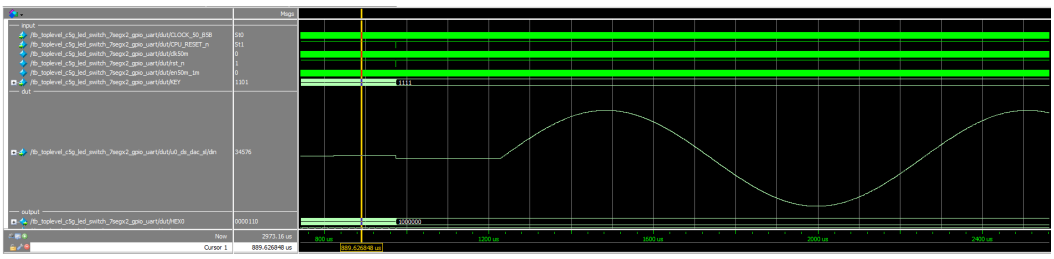

Abbildung 1.3: Sinus Verifikation in der Simulation
Quelle: eigene Ausarbeitung

1.4 Bench Verifikation

Der Bitstream wird mit einem Tiefpassfilter erster Ordnung gefiltert (Abbildung 1.4).

$$f_g = \frac{1}{2 \cdot \pi \cdot R \cdot C} \tag{1.1}$$

Mit $C = 1\ nF$ und $R = 10\ k\Omega$ folgt für f_g

$$f_g = 15,91\ kHz \tag{1.2}$$

Der erzeugte Sinus hat eine Frequenz von $f_{max} = 976,56\ Hz$ unter der Annahme, dass der Prescaler eine Frequenz von 10 MHz und der gespeicherte Sinus 1024 Werte für eine Periode hat.

$$f_{max} = \frac{10^6\ Hz}{1024} \tag{1.3}$$

$$= 976,56\ Hz \tag{1.4}$$

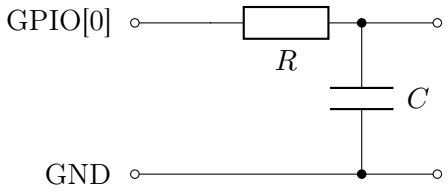

Abbildung 1.4: Tiefpass erster Ordnung
Quelle: eigene Ausarbeitung

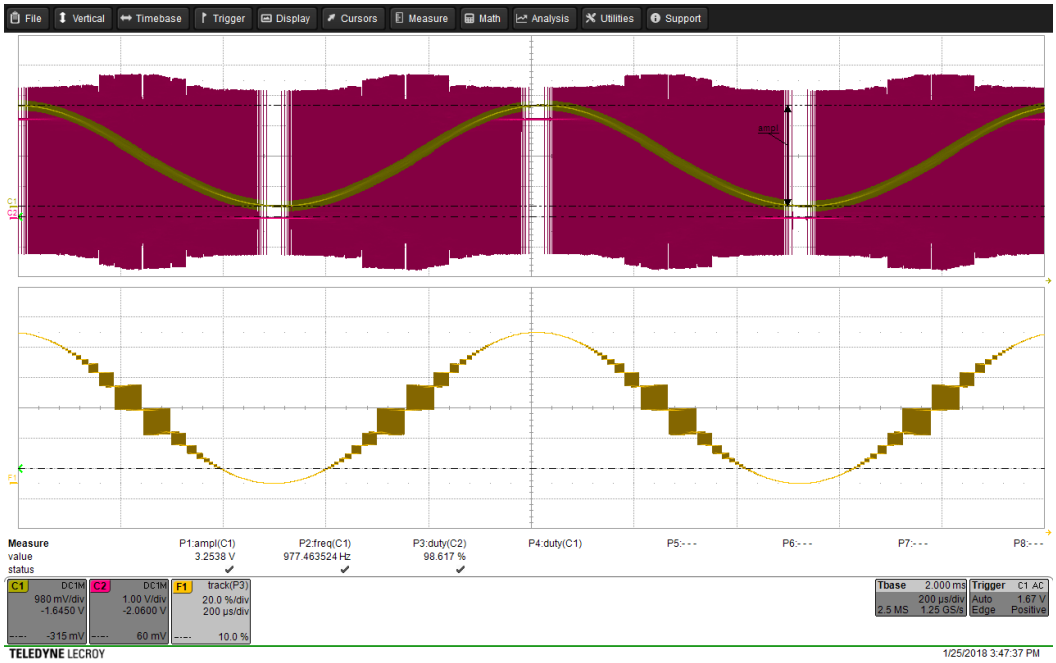

Abbildung 1.5: Sinus Verifikation
Quelle: eigene Ausarbeitung

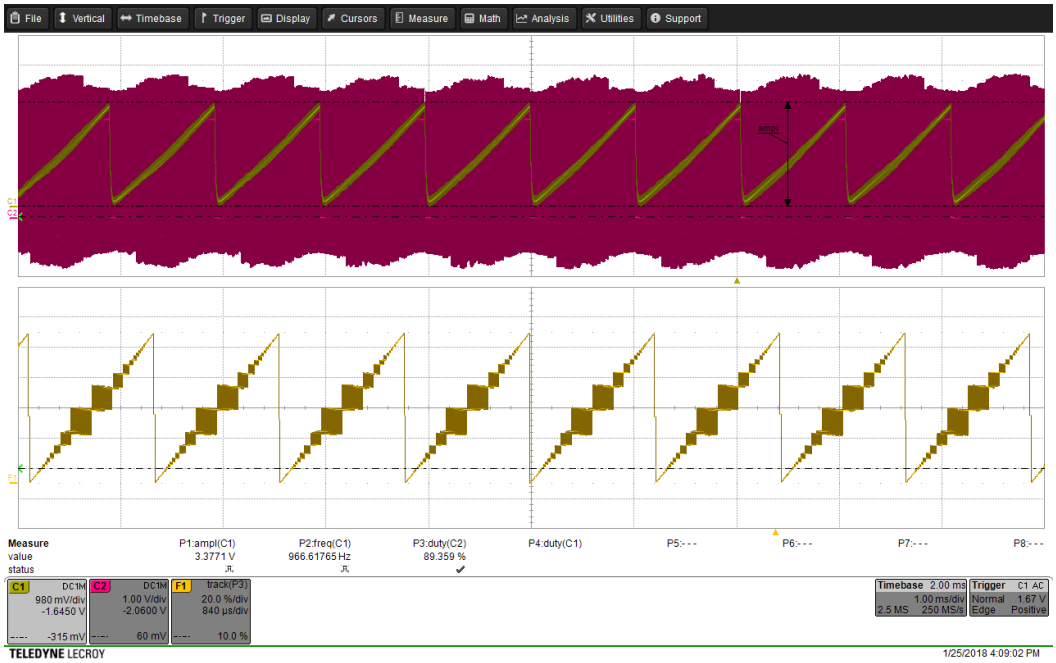

Abbildung 1.6: Sägezahn Verifikation
Quelle: eigene Ausarbeitung

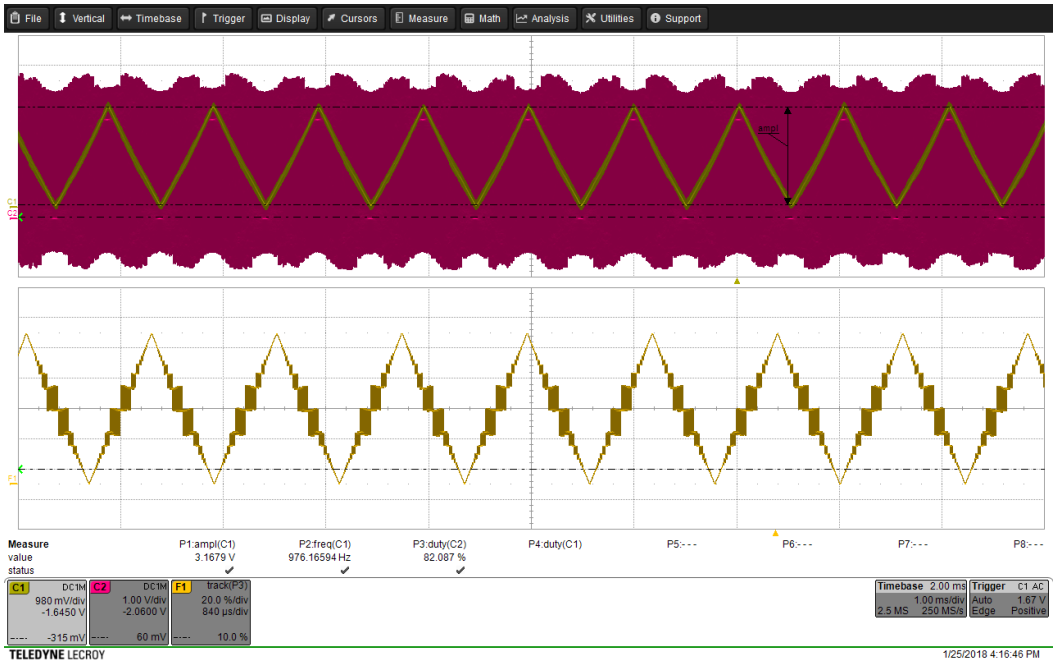

Abbildung 1.7: Dreieck Verifikation
Quelle: eigene Ausarbeitung

Abbildung 1.8: Rechteck Verifikation
Quelle: eigene Ausarbeitung