电子技术基础实验第十四周实验报告

王磊　2022012972

2023 年 12 月 27 日

1　task1&2

task2是在task1基础上完成的，因此只需要介绍task2。

1.1　模块设计

本任务中除了MUX模块，其余均在上周已经完成，因此只需要介绍MUX模块的设计。

1.1.1　MUX模块

mux模块的功能是将两个输入信号中的一个输出。代码如下：

```
module mux2(
    input [7:0] s,
    input [23:0] d1,
    input [23:0] d2,
    output reg [23:0] y
);

    always @ (s or d1 or d2)
        case (s)
            8'd0: y <= d1;
            8'd1: y <= d2;
            default: y <= 24'd0;
        endcase

endmodule
```

模块的作用是根据输入的s信号，选择输出d1或d2。当s为0时，输出d1；当s为1时，输出d2；其他情况输出0。

1.1.2 顶层模块

顶层模块的代码如下：

```
        `include "rom_data_tri.v"
`include "rom_base.v"
`include "rom_harmony.v"
`include "dac_controller_new_1213.v"
`include "mypll2.v"
`include "adc_controller_new_1213.v"
`include "adc_data_ready_tri.v"
`include "uart_NbyteTran_3byteData_controller.v"
`include "uart_tx_byte.v"
`include "uart_rx_byte.v"
`include "uart_rx_Nbyte_controller.v"
`include "uart_1byte_data_reg.v"
`include "mux2.v"
`include "mymult.v"

module task2_top(
    input clk,
    input rst,
    input sdout_adc,
    input sci_rx,
    output lrck_dac,
    output sclk_dac,
    output mclk_dac,
    output sdata_dac,
    output mclk_adc,
    output sclk_adc,
    output lrck_adc,
    output reg  buzz,
    output [23:0] dataL_adc,
    output [23:0] dataR_adc,
    output  send_en,
    output sci_tx
);
```

```
34
35  initial begin
36      buzz = 1'b1;
37  end
38
39  initial begin
40      buzz = 1'b1;
41  end
42
43  wire[23:0] data_dac_chL;
44  wire[23:0] data_dac_chR;
45  wire[7:0] addr_chL;
46  wire[7:0] addr_chR;
47  wire c0;//6553600hz
48  wire c1;//1.28Mhz
49
50  wire tx_done;
51  wire tx_en;
52  wire [23:0] data;
53  wire [7:0] tx_d;
54  wire send_done;
55
56  wire rx_done;
57  wire [7:0] uart_data;
58  wire [7:0] rx_data_out;
59  wire [23:0] mux_data_out;
60
61  wire [7:0] byte1,byte2,byte3;
62
63  wire [23:0] result1,result2;
64  rom_data_tri uut1(
65      .lr_ch_tri_clk(lrck_dac),
66      .rst(rst),
67      .addr_chL(addr_chL),
68      .addr_chR(addr_chR)
69  );
70
71  rom_base uut2(
72      .clock(clk),
73      .address(addr_chL),
```

```
74      .q(data_dac_chL)
75  );
76
77  rom_harmony uut3(
78      .clock(clk),
79      .address(addr_chR),
80      .q(data_dac_chR)
81  );
82
83  dac_controller_new_1213 uut4(
84      .clk(c0),
85      .rst(rst),
86      .data_dac_chL(result1),
87      .data_dac_chR(result2),
88      .mclk_dac(mclk_dac),
89      .sclk_dac(sclk_dac),
90      .lrck_dac(lrck_dac),
91      .sdata_dac(sdata_dac)
92  );
93
94  adc_controller_new_1213 uut6(
95      .clk(c1),
96      .rst(rst),
97      .sdout_adc(sdout_adc),
98      .mclk_adc(mclk_adc),
99      .sclk_adc(sclk_adc),
100     .lrck_adc(lrck_adc),
101     .dataL_adc(dataL_adc),
102     .dataR_adc(dataR_adc)
103 );
104
105 mypll2 uut5(
106     .inclk0(clk),
107     .c0(c0),
108     .c1(c1)
109 );
110
111 adc_data_ready_tri uut7(
112     .clk(clk),
113     .rst(rst),
```

```
114     .lrck(lrck_adc),
115     .send_en(send_en)
116 );
117 
118 uart_NbyteTran_3byteData_controller uut8(
119     .clk(clk),
120     .rst(rst),
121     .send_en(send_en),
122     .data(mux_data_out),
123     .tx_d(tx_d),
124     .tx_en(tx_en),
125     .tx_done(tx_done)
126 );
127 
128 uart_tx_byte uut9(
129     .clk(clk),
130     .rst(rst),
131     .rx_d(tx_d),
132     .tx_en(tx_en),
133     .tx_done(tx_done),
134     .sci_tx(sci_tx)
135 );
136 
137 uart_rx_byte uut10(
138     .clk(clk),
139     .rst(rst),
140     .sci_rx(sci_rx),
141     .rx_done(rx_done),
142     .uart_data(uart_data)
143 );
144 
145 mux2 uut12(
146     .s(byte1),
147     .d1(dataL_adc),
148     .d2(dataR_adc),
149     .y(mux_data_out)
150 );
151 
152 mymult uut13(
153     .dataa(data_dac_chL),
```

```
154      .datab(byte2),
155      .result(result1)
156  );
157
158  mymult uut14(
159      .dataa(data_dac_chR),
160      .datab(byte3),
161      .result(result2)
162  );
163
164  uart_rx_Nbyte_controller uut15(
165      .clk(clk),
166      .rst(rst),
167      .rx_done(rx_done),
168      .uart_data(uart_data),
169      .byte1(byte1),
170      .byte2(byte2),
171      .byte3(byte3)
172  );
173
174  endmodule
```

对应的RTL电路图如图1所示。

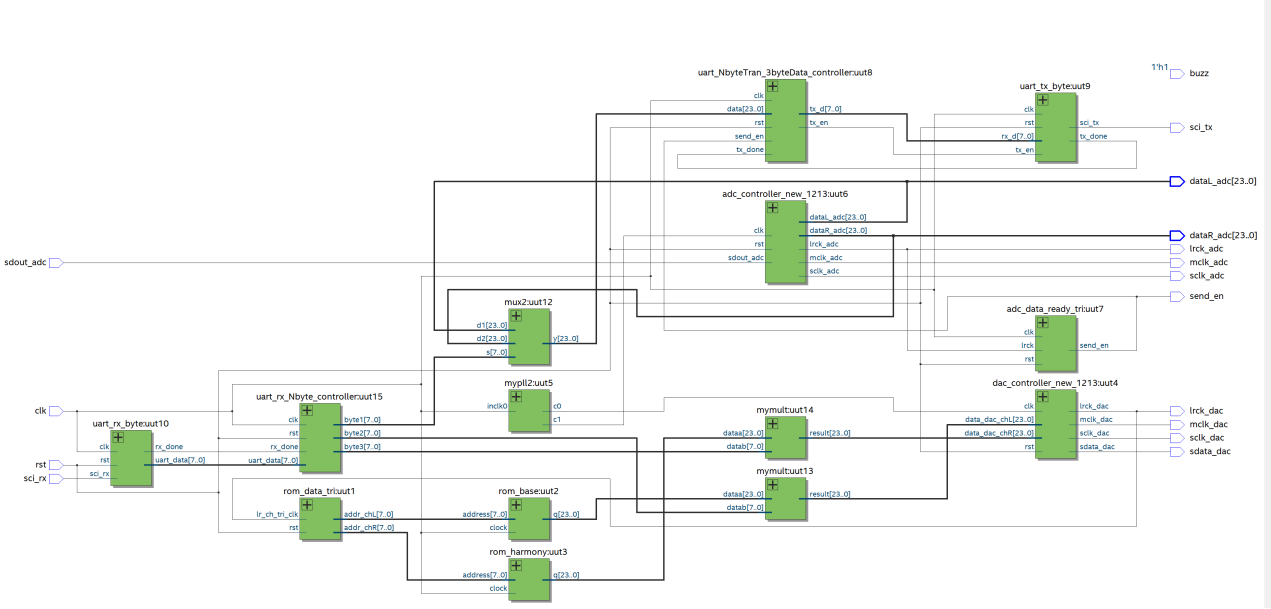


图 1: task2顶层模块RTL电路图

1.2 仿真结果

按照ppt中的要求，搭建simulink仿真模型，如图2所示。

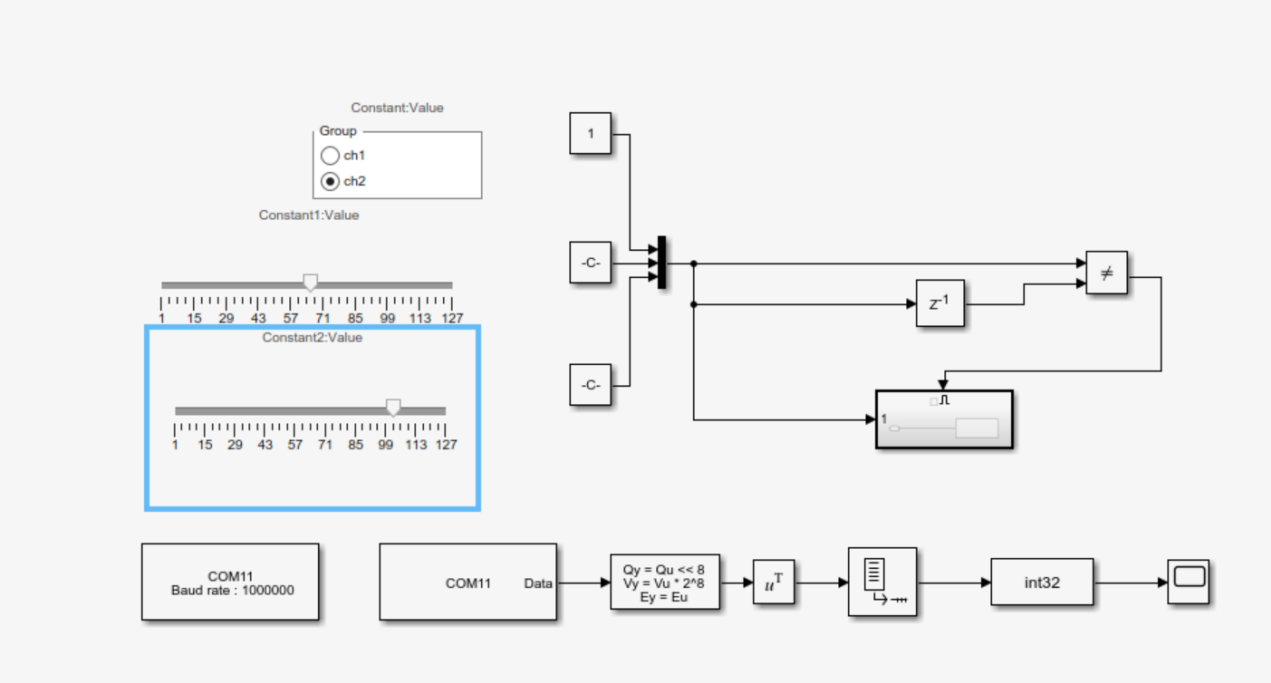
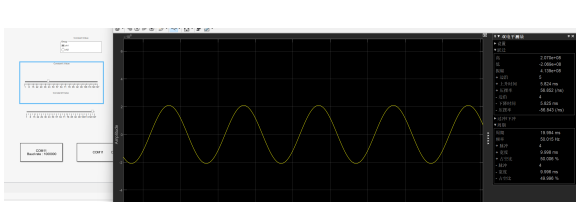


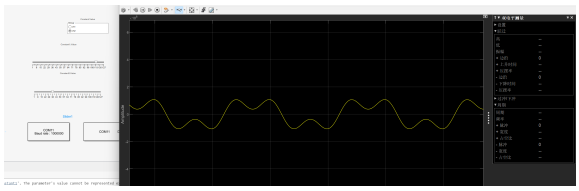
图 2: simulink仿真模型

仿真结果如图3所示。

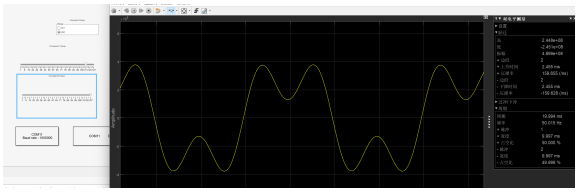


(a) 波形图1

(b) 波形图2



(c) 波形图3



(d) 波形图4

图 3: simulink接收波形结果

可以看出通过选择channel，可以选择输出的信号。同时调节滑块可以调节输出波形的赋值。