

Computation II: embedded system design (5EIB0)

[Dashboard](#) / [My courses](#) / [5EIB0](#) / [Final Exam 2024-04-12](#) / [Part 2f - 2024-04-12 Implement UART Re-transmission Module \(6pt\)](#)

Part 2f - 2024-04-12 Implement UART Re-transmission Module (6pt)

 [Description](#)

 [Submission view](#)

Grade


Reviewed on Monday, 15 April 2024, 4:02 PM by Automatic grade

Grade: 0.00 / 6.00

Assessment report [-]

[\[+\]](#) **Test 1: Meets Reference Specification**

[\[+\]](#) **Summary of tests**

 Submitted on Friday, 12 April 2024, 4:29 PM ( [Download](#))

uart_retrans.v

```

1  module uart_retrans(
2      input clk,
3      input reset,
4      input ack,
5      input signal,
6      output trigger,
7      output valid,
8      output error,
9      output request_resend,
10     output[4:0] resend_count);
11     reg [4:0] out;
12
13
14     timer timer_inst(
15         .clk(clk),
16         .reset(reset),
17         .enable(1),
18         .value(5'h8),
19         .valid(valid),
20         .count(count),
21         .trigger(trigger));
22
23
24     uart_retrans_fsm uart_retrans_fsm_inst(
25         .clk(clk),
26         .reset(reset),
27         .ack(ack),
28         .frame_valid(frame_valid),
29         .parity_error(parity_error),
30         .timeout(timeout),
31         .error(error),
32         .request_resend(request_resend),
33         .valid(valid));
34
35
36     uart_parity_even uart_parity_even_inst(
37         .clk(clk),
38         .reset(reset),
39         .signal(signal),
40         .error(error),
41         .valid(valid));
42
43     assign count = out;
44     assign complete = ~(|out);
45
46     assign MUX1 = (~complete & enable)?(out - 'd1):(out);
47     assign MUX2 = (valid)?(value):(MUX1);
48     assign out_next = (reset)?(5'h0):(MUX2);
49
50     always @(posedge clk) begin
51         out <= out_next;
52     end
53 endmodule
54

```

uart_retrans_fsm.v

```

1  module uart_retrans_fsm(
2      input clk,
3      input reset,
4      input ack,
5      input ,
6      input ,
7      input ,
8      output error,
9      output request_resend,
10     output valid);
11
12     reg [2:0] state, state_next;
13     parameter WAIT_ = 'd0;
14     parameter WAIT_RESEND1 = 'd1;
15     parameter WAIT_RESEND2 = 'd2;
16     parameter ERROR = 'd3;
17     parameter RELEASE_ = 'd4;
18
19     always @(posedge clk) begin
20         if(reset & ~(valid &)) begin
21             state <= IDLE;
22         end else begin
23             state <= state_next;
24         end
25     end
26
27     always @(*) begin
28         state_next = state;
29
30         case(state)
31             WAIT_: begin
32                 if((enable==0) & (complete==0)) begin
33                     state_next = PAUSED;
34                 end else if((enable==1) & (complete==0)) begin
35                     state_next = COUNTING;
36                 end
37             end
38             WAIT_RESEND1: begin
39                 if((enable==0) & (complete==1)) begin
40                     state_next = IDLE;
41                 end else if((enable==1) & (complete==1)) begin
42                     state_next = DONE;
43                 end else if((enable==0) & (complete==0)) begin
44                     state_next = PAUSED;
45                 end
46             end
47             WAIT_RESEND2: begin
48                 if((enable==1) & (complete==1)) begin
49                     state_next = DONE;
50                 end else if((enable==1) & (complete==0)) begin
51                     state_next = COUNTING ;
52                 end else if((enable==0) & (complete==1)) begin
53                     state_next = IDLE;
54                 end
55             end
56             ERROR: begin
57                 if(complete==1) begin
58                     state_next = IDLE;
59                 end else if((enable==1) & (complete==0)) begin
60                     state_next = COUNTING;
61                 end else if((enable==0) & (complete==0)) begin
62                     state_next = PAUSED;
63                 end
64             end
65             RELEASE_: begin
66                 if(complete==1) begin
67                     state_next = IDLE;
68                 end else if((enable==1) & (complete==0)) begin
69                     state_next = COUNTING;
70                 end else if((enable==0) & (complete==0)) begin
71                     state_next = PAUSED;
72                 end
73             end
74         endcase
75     end
76 endmodule
77

```

[VPL](#)

You are logged in as Thomas Stirling Valdez (Log out)
5EIB0

Data retention summary