

```

1  `timescale 1ns/1ps
2  module UART_TX_RX_MODULE_TB4#(
3      parameter UART_BAUD_RATE          = 9600,
4      parameter CLOCK_FREQUENCY         = 38400,
5      parameter PARITY                   = 2,
6      parameter NUM_OF_DATA_BITS_IN_PACK = 8,
7      parameter NUMBER_STOP_BITS        = 1
8  );
9  localparam PERIOD_IN_CLOCK_NS=1000000000/CLOCK_FREQUENCY;
10 //Входы
11 reg IN_CLOCK_1, IN_CLOCK_2;
12 reg IN_TX_LAUNCH_1, IN_TX_LAUNCH_2;
13 reg [NUM_OF_DATA_BITS_IN_PACK-1:0] IN_TX_DATA_1, IN_TX_DATA_2;
14 //Выходы
15 wire OUT_TX_ACTIVE_1, OUT_TX_ACTIVE_2;
16 wire OUT_TX_DONE_1, OUT_TX_DONE_2;
17 wire OUT_TX_STOP_BIT_ACTIVE_1, OUT_TX_STOP_BIT_ACTIVE_2;
18 wire OUT_TX_START_BIT_ACTIVE_1, OUT_TX_START_BIT_ACTIVE_2;
19 wire OUT_RX_DATA_READY_1, OUT_RX_DATA_READY_2;
20 wire [NUM_OF_DATA_BITS_IN_PACK-1:0] OUT_RX_DATA_1, OUT_RX_DATA_2;
21 wire OUT_RX_ERROR_1, OUT_RX_ERROR_2;
22
23 wire BUS_TRANSMIT_1_TO_2, BUS_TRANSMIT_2_TO_1;
24
25
26 UART_TX_RX_MODULE #(
27     .UART_BAUD_RATE(UART_BAUD_RATE),
28     .CLOCK_FREQUENCY(CLOCK_FREQUENCY),
29     .PARITY(PARITY),
30     .NUM_OF_DATA_BITS_IN_PACK(NUM_OF_DATA_BITS_IN_PACK),
31     .NUMBER_STOP_BITS(NUMBER_STOP_BITS)
32 )
33 UTRM_1
34 (
35     .IN_CLOCK(IN_CLOCK_1),
36     .IN_TX_LAUNCH(IN_TX_LAUNCH_1),
37     .IN_TX_DATA(IN_TX_DATA_1),
38
39     .OUT_TX_ACTIVE(OUT_TX_ACTIVE_1),
40     .OUT_TX_DONE(OUT_TX_DONE_1),
41     .OUT_TX_STOP_BIT_ACTIVE(OUT_TX_STOP_BIT_ACTIVE_1),
42     .OUT_TX_START_BIT_ACTIVE(OUT_TX_START_BIT_ACTIVE_1),
43     .OUT_RX_DATA_READY(OUT_RX_DATA_READY_1),
44     .OUT_RX_DATA(OUT_RX_DATA_1),
45     .OUT_RX_ERROR(OUT_RX_ERROR_1),
46
47     .IN_RX_SERIAL(BUS_TRANSMIT_2_TO_1),
48     .OUT_TX_SERIAL(BUS_TRANSMIT_1_TO_2)
49 );
50
51 UART_TX_RX_MODULE #(
52     .UART_BAUD_RATE(UART_BAUD_RATE),
53     .CLOCK_FREQUENCY(CLOCK_FREQUENCY),
54     .PARITY(PARITY),
55     .NUM_OF_DATA_BITS_IN_PACK(NUM_OF_DATA_BITS_IN_PACK),
56     .NUMBER_STOP_BITS(NUMBER_STOP_BITS)
57 )
58 UTRM_2
59 (
60     .IN_CLOCK(IN_CLOCK_2),
61     .IN_TX_LAUNCH(IN_TX_LAUNCH_2),
62     .IN_TX_DATA(IN_TX_DATA_2),
63
64     .OUT_TX_ACTIVE(OUT_TX_ACTIVE_2),
65     .OUT_TX_DONE(OUT_TX_DONE_2),
66     .OUT_TX_STOP_BIT_ACTIVE(OUT_TX_STOP_BIT_ACTIVE_2),
67     .OUT_TX_START_BIT_ACTIVE(OUT_TX_START_BIT_ACTIVE_2),
68     .OUT_RX_DATA_READY(OUT_RX_DATA_READY_2),
69     .OUT_RX_DATA(OUT_RX_DATA_2),
70     .OUT_RX_ERROR(OUT_RX_ERROR_2),
71
72     .IN_RX_SERIAL(BUS_TRANSMIT_1_TO_2),
73     .OUT_TX_SERIAL(BUS_TRANSMIT_2_TO_1)
74 );
75 always
76 begin
77     #(PERIOD_IN_CLOCK_NS/2)
78     IN_CLOCK_1=!IN_CLOCK_1;
79     IN_CLOCK_2=!IN_CLOCK_2;

```

```
80     end
81     initial begin
82         IN_CLOCK_1=1'b1; IN_CLOCK_2=1'b0;
83         IN_TX_LAUNCH_1=0; IN_TX_LAUNCH_2=0;
84         IN_TX_DATA_1=8'bz; IN_TX_DATA_2=8'bz;
85         #(PERIOD_IN_CLOCK_NS*10)
86         IN_TX_DATA_1=8'b00000001;
87         #(PERIOD_IN_CLOCK_NS*12)
88         IN_TX_LAUNCH_1=1'b1;
89         #(PERIOD_IN_CLOCK_NS*12)
90         IN_TX_LAUNCH_1=1'b0; //убираем
91         #(PERIOD_IN_CLOCK_NS*20)
92         IN_TX_DATA_1=8'bz;
93     end
94
95     initial begin
96         forever
97         begin
98             @(posedge OUT_RX_DATA_READY_2)
99             begin
100                 if(OUT_RX_DATA_2<64)
101                 begin
102                     IN_TX_DATA_2=OUT_RX_DATA_2*4; //заряжаем принятые данные, чтоб
103                     отправить обратно
104                     #(PERIOD_IN_CLOCK_NS*25)
105                     IN_TX_LAUNCH_2=1'b1;
106                     #(PERIOD_IN_CLOCK_NS*25)
107                     IN_TX_LAUNCH_2=1'b0;
108                     #(PERIOD_IN_CLOCK_NS*20)
109                     IN_TX_DATA_2=8'bz;
110                 end
111                 else
112                 begin
113                     IN_TX_DATA_2=8'bz;
114                 end
115             end
116         end
117     end
118
119     initial begin
120         forever
121         begin
122             @(posedge OUT_RX_DATA_READY_1)
123             begin
124                 if(OUT_RX_DATA_1<64)
125                 begin
126                     IN_TX_DATA_1=OUT_RX_DATA_1*4; //заряжаем принятые данные, чтоб
127                     отправить обратно
128                     #(PERIOD_IN_CLOCK_NS*25)
129                     IN_TX_LAUNCH_1=1'b1;
130                     #(PERIOD_IN_CLOCK_NS*25)
131                     IN_TX_LAUNCH_1=1'b0;
132                     #(PERIOD_IN_CLOCK_NS*20)
133                     IN_TX_DATA_1=8'bz;
134                 end
135                 else
136                 begin
137                     IN_TX_DATA_1=8'bz;
138                 end
139             end
140         end
141     end
142
143 endmodule
```