```
/*-----------------------------------------
Name Lamin Jammeh
CLass: EE417 Summer 2024
Lesson 08 HW Question 2
Group: Ron Kalin/ Lamin Jammeh
Project Description: TestBench for teh UART_Rx
-----------------------------------------*/

module UART_Rx_tb ();

//define the registers and wires
reg   Serial_in, read_not_ready_in, Sample_clk, rst_b;
wire  [7:0]    Rx_datareg;
wire  read_not_ready_out;
wire  Error1, Error2;

//add wires to monitor the inputs and outputs on the submodules
wire        Ser_in_0;
wire        clr_Sample_counter;
wire        inc_Sample_counter;
wire        clr_Bit_counter;
wire        inc_Bit_counter;
wire [7:0]  Rx_shftreg;
wire [3:0]  Sample_counter;
wire [4:0]  Bit_counter;
wire        SC_eq_3;
wire        SC_lt_7;
wire [1:0]  state;
wire        shift;
wire        load;
wire        BC_eq_8;

//define the unit under test UUT
UART_Rx #(8, 4) UUT (
                     .Rx_datareg(Rx_datareg),
                     .read_not_ready_out(read_not_ready_out),
                     .Error1(Error1),
                     .Error2(Error2),

                     .Serial_in(Serial_in),
                     .read_not_ready_in(read_not_ready_in),
                     .Sample_clk(Sample_clk),
                     .rst_b(rst_b)
                     );

//assign internal probe monitor
assign Ser_in_0             = UUT.M1.Ser_in_0;
assign clr_Sample_counter   = UUT.M1.clr_Sample_counter;
assign inc_Sample_counter   = UUT.M1.inc_Sample_counter;
assign clr_Bit_counter      = UUT.M1.clr_Bit_counter;
assign inc_Bit_counter      = UUT.M1.inc_Bit_counter;
assign Rx_shftreg           = UUT.M1.Rx_shftreg;
assign Sample_counter       = UUT.M1.Sample_counter;
assign Bit_counter          = UUT.M1.Bit_counter;
assign SC_eq_3              = UUT.M0.SC_eq_3;
assign SC_lt_7              = UUT.M0.SC_lt_7;
assign state                = UUT.M0.state;
assign shift                = UUT.M0.shift;
assign load                 = UUT.M0.load;
assign BC_eq_8              = UUT.M0.BC_eq_8;

//clock cycle
always
   begin
      Sample_clk = 0;
      forever #5 Sample_clk = ~Sample_clk;
   end

//initialie reset and run enough clk cycle to get all desired counts
```

```
70   initial
71      begin
72         // Initialize Inputs
73       Serial_in = 0;
74       read_not_ready_in = 0;
75       Sample_clk = 0;
76       rst_b = 0;
77
78       // Apply reset
79       rst_b = 1;
80       #10;
81       rst_b = 0;
82       #10;
83       rst_b = 1;
84
85       // Test Case 1: Transmit a byte (e.g., 8'b10101010)
86       Serial_in = 1; // Start bit
87       #100;
88
89       Serial_in = 1; // Bit 0
90       #100;
91
92       Serial_in = 0; // Bit 1
93       #100;
94
95       Serial_in = 1; // Bit 2
96       #100;
97
98       Serial_in = 0; // Bit 3
99       #100;
100
101      Serial_in = 1; // Bit 4
102      #100;
103
104      Serial_in = 0; // Bit 5
105      #100;
106
107      Serial_in = 1; // Bit 6
108      #100;
109
110      Serial_in = 0; // Bit 7
111      #100;
112
113      Serial_in = 1; // Stop bit
114      #100;
115
116
117       // Force load signal to load the data
118      force UUT.M0.load = 1;
119      #100;
120      force UUT.M0.load = 0;
121      //relese load signal
122      release UUT.M0.load;
123      // Wait for a few cycles
124      #500;
125      #100;
126      $stop;
127        begin
128           $monitor ($time ,, "Serial_in = %h  Rx_datareg = %h", Serial_in, Rx_datareg);
129        end
130     end
131  endmodule
132
133
```