آزمایشگاه طراحی سیستمهای دیجیتال
گزارش آزمایش ۴

۱۳ مرداد ۱۴۰۳

امیرحسین ملک محمدی ۴۰۱۱۰۶۵۷۷
متین محمدی ۴۰۱۱۱۰۳۲۹
صادق محمدیان ۴۰۱۱۰۹۴۷۷

```
module UART(
    input clk,
    input start_sending,
    input [6:0] data_to_send,
    input serial_in,

    output serial_out,
    output sent,
    output [6:0] recived_data,
    output valid,
    output recived

);

UART_sender sender (clk, start_sending, data_to_send, serial_out, sent);
UART_reciver reciver (clk, serial_in, valid, recived, recived_data);


endmodule
```

شکل ۱: کد وریلاگ قطعه اصلی

۱ شرح آزمایش

در این آزمایش می‌خواهیم یک UART طراحی کنیم. این قطعه از دو بخش فرستنده و گیرنده تشکیل شده است.

۱.۱ ورودی‌ها

ورودی‌های دستگاه شامل سیگنال کلاک، ورودی شروع ارسال داده، داده ۷ بیتی، و ورودی سریال است.

۲.۱ خروجی‌ها

خروجی‌های قطعه شامل خروجی سریال، سیگنال اتمام ارسال، داده دریافت شده، صحت داده دریافت شده(مقایسه بیت توازن)، و اتمام دریافت داده است.

۲ کد وریلاگ

۱.۲ قطعه اصلی

در شکل .Fig ۱ کد قطعه اصلی آمده.

این قطعه از دو بخش فرستنده و گیرنده تشکیل شده و ورودی‌ها را به آن‌ها می‌دهد و خروجی‌ها را از آن‌ها می‌گیرد.

۲.۲ ارسال‌کننده

در شکل .Fig ۲ کد ارسال‌کننده آمده.

این بخش یک متغییر ۴ بیتی به نام index دارد که نشان می‌دهد بیت چندم را باید ارسال کنیم. اول یک بیت که همان بیت شروع است را ارسال می‌کنیم. سپس بیت توازن را ارسال می‌کنیم و بعد داده ورودی را به ترتیب ارسال می‌کنیم. در نهایت نیز یک بیت ۰ که بیت پایان است را ارسال می‌کنیم.

در هر لبه بالارونده کلاک، اگر ورودی start فعال بود، مراحل فرستادن را شروع می‌کنیم و گرنه یا ارسال کردن را ادامه می‌دهیم یا اگر ارسال تمام شده بود، ۰ را خروجی می‌دهیم و خروجی اتمام ارسال را فعال می‌کنیم

۳.۲ گیرنده

در شکل .Fig ۳ کد گیرنده آمده.

این مدار دو حالت دارد، یکی حالت pre-start است (یعنی هنوز ورودی شروع نیامده) و حالت دیگر حالت recive است که یعنی ورودی شروع آمده و مدار در حال ورودی گرفتن است.

در این بخش یک متغییر ۳ بیتی به نام index داریم که نشان می‌دهد بیت ورودی را در کدام خانه ذخیره خواهیم کرد. زمانی که این متغییر به مقدار ۷ برسد یعنی فرایند ورودی گرفتن تمام شده.

خروجی valid وقتی ۱ می‌شود که فرایند دریافت تمام شده باشد و بیت توازن صحیح باشد.

۳ تست بنچ

در شکل .Fig ۴ کد تست‌بنچ آمده.

در این تست بنچ از UART دو نمونه می‌گیریم و خروجی اولی را به ورودی دومی وصل می‌کنیم. سپس به قطعه اول یک حرف ورودی می‌دهیم و صبر می‌کنیم تا قطعه دوم کامل دریافت کند. این کار را برای یک حرف دیگر نیز انجام می‌دهیم.

در شکل .Fig ۵ خروجی تست‌بنچ آمده.

خروجی طبق انتظار است. UART دوم داده را به درستی دریافت کرده است.

۴ نتیجه‌گیری

در این آزمایش ابتدا با زبان وریلاگ یک UART ساختیم سپس از آن دو نمونه گرفتیم و آن‌ها را به هم وصل کردیم. دیدم که قطعه دوم به درستی ورودی را دریافت می‌کند.

```
1   module UART_sender (
2       input clk,
3       input start,
4       input [6:0] data_in,
5
6       output reg data_out,
7       output reg done
8   );
9
10  wire [9:0] data_to_send ;
11  assign data_to_send[0] = 1'b1;
12  assign data_to_send[1] = ^data_in;
13  assign data_to_send[8:2] = data_in[6:0] ;
14  assign data_to_send[9] = 1'b0;
15  reg [3:0] index = 4'd0;
16
17  always @(posedge clk) begin
18      if (start) begin
19          index = 4'd0;
20          done = 1'b0;
21      end
22      else if (index < 10)begin
23          data_out = data_to_send[index];
24          index = index + 1;
25      end
26      else begin
27          data_out = 1'b0;
28          done = 1'b1;
29      end
30  end
31  endmodule
```

شکل ۲: کد وریلاگ ارسال‌کننده

```verilog
module UART_reciver (
    input clk,
    input data_in,

    output valid,
    output reg done,
    output [6:0]data_out
);

reg [7:0] data;

assign data_out = data[7:1];

wire parity = ^data[7:1];
wire parity_recived = data[0];
assign valid = (parity == parity_recived) & done;

localparam pre_start = 1'b0;
localparam recive = 1'b1;
reg state = pre_start;

reg [2:0] index = 3'd0;

always @(posedge clk) begin
    if (state == pre_start) begin
        if (data_in == 1'b1) state = recive;
        else state = pre_start;

        index = 3'd0;
        done = 1'b0;
    end
    else if (state == recive) begin
        if (index ==  3'd7) begin
            state = pre_start;
            done = 1'b1;
        end
        data[index] = data_in;
        index = index + 1;
    end
end
endmodule
```

شکل ۳: کد وریلاگ دریافت‌کننده

```
module TB;

reg clk, start_sending1, serial_in1;
reg [6:0] data_to_send1;

wire serial_out1, sent1, valid1, recived1;
wire [6:0] recived_data1;

reg start_sending2, serial_in2;
reg [6:0] data_to_send2;

wire serial_out2, sent2, valid2, recived2;
wire [6:0] recived_data2;

UART u1 (clk, start_sending1, data_to_send1, serial_in1, serial_out1, sent1, recived_data1, valid1, recived1);
UART u2 (clk, start_sending2, data_to_send2, serial_out1, serial_out2, sent2, recived_data2, valid2, recived2);

always #5 clk = ~clk;

initial begin
    clk = 0;
    start_sending1 = 1;
    start_sending2 = 0;
    data_to_send1 = "A";
    $display("data to send : %c", data_to_send1);
    #10
    start_sending1 = 0;
    wait(recived2);
    $display("recived data : %c", recived_data2);
    #10
    start_sending1 = 1;
    data_to_send1 = "*";
    $display("data to send : %c", data_to_send1);
    #10
    start_sending1 = 0;
    wait(recived2);
    $display("recived data : %c", recived_data2);
end
endmodule
```

شکل ۴: کد تست بنچ

```
VSIM 57> run
# data to send : A
run
# recived data : A
# data to send : *
VSIM 58> run
# recived data : *
```

شکل ۵: خروجی تست بنچ