# **实验 3-Verilog 综合实验：串口 uart实验报告**

姓名： 刘贤彬 学号： 23336159

1. **实验内容**

由于新的 FPGA 实验班没有七段数码管，然而后续实验中我们需要增加一种实验手段 来输出执行结果。因此，本次实验需要完成串口通讯实验，完成计算机（PC 电脑）与 FPGA 芯片之间的通讯交互，为后面的MIPS CPU 设计提供测试交互接口，输出打印CPU 的 计算结果。

实验目的：

掌握串口原理和时序协议标准

读懂串口Verilog 代码

掌握测试串口通讯的方法。

1. **实验过程**
2. 导入uart\_top,uart\_tx,uart\_rx,uart\_loop\_tb等文件。
3. 编写uart\_top文件。

uart\_top代码：

module uart\_top#(

parameter baudrate = 115200

)(

input wire CLK,

input wire RSTN,

input wire rx\_i, // Receiver input

output wire tx\_o // Transmitter output

);

localparam baud\_div = 434 ;

// receive buffer register, read only

wire [7:0] rx\_data;

wire [7:0] tx\_data;

// parity error

wire parity\_error;

wire [3:0] IIR\_o;

reg [3:0] clr\_int;

// tx flow control

wire tx\_ready;

// rx flow control

wire rx\_valid;

reg fifo\_tx\_valid;

reg tx\_valid;

wire fifo\_rx\_valid;

reg fifo\_rx\_ready;

wire rx\_ready;

wire fifo\_wren;

wire fifo\_rden;

wire fifo\_full;

wire fifo\_empty;

uart\_rx u\_uart\_rx(

.clk\_i ( CLK ),

.rstn\_i ( RSTN ),

.rx\_i ( rx\_i ),

.cfg\_en\_i ( 1'b1 ),

.cfg\_div\_i ( 16'd0431 ),

.cfg\_parity\_en\_i ( 1'b0 ),

.cfg\_parity\_sel\_i ( 2'b00 ),

.cfg\_bits\_i ( 2'b11 ), // 2'b00 -> 5bit, 2'b01 -> 6bit, 2'b10 -> 7bit, 2'b11 -> 8bit

// .cfg\_stop\_bits\_i ( regs\_q[(LCR \* 8) + 2] ),

.busy\_o ( ),

.err\_o ( parity\_error ),

.err\_clr\_i ( 1'b0 ),

.rx\_data\_o ( rx\_data ),

.rx\_valid\_o ( rx\_valid ),

.rx\_ready\_i ( rx\_ready )

);

uart\_tx u\_uart\_tx(

.clk\_i ( CLK ),

.rstn\_i ( RSTN ),

.tx\_o ( tx\_o ),

.busy\_o ( ),

.cfg\_en\_i ( 1'b1 ),

.cfg\_div\_i ( 16'd0431 ),

.cfg\_parity\_en\_i ( 1'b0 ),

.cfg\_parity\_sel\_i ( 2'b00 ),

.cfg\_bits\_i ( 2'b11 ), // 2'b00 -> 5bit, 2'b01 -> 6bit, 2'b10 -> 7bit, 2'b11 -> 8bit

.cfg\_stop\_bits\_i ( 1'b1 ),

.tx\_data\_i ( tx\_data ),

.tx\_valid\_i ( tx\_valid ),

.tx\_ready\_o ( tx\_ready )

);

// loop

fifo\_8b u\_fifo\_8b (

.clk(CLK), // input wire clk

.srst(~RSTN), // input wire srst

.din(rx\_data), // input wire [7 : 0] din

.wr\_en(fifo\_wren), // input wire wr\_en

.rd\_en(fifo\_rden), // input wire rd\_en

.dout(tx\_data), // output wire [7 : 0] dout

.full(fifo\_full), // output wire full

.empty(fifo\_empty) // output wire empty

);

// your code here

assign fifo\_wren = rx\_valid && ~fifo\_full;

assign fifo\_rden = ~fifo\_empty && tx\_ready;

always @(posedge CLK or negedge RSTN)

if (!RSTN)

tx\_valid <= 1'b0;

else

tx\_valid <= fifo\_rden;

assign rx\_ready = ~fifo\_full;

// your code end

endmodule

1. **编写uart\_loop\_tb文件：**

**uart\_loop\_tb代码：**

`timescale 1ns / 1ps

module axi\_uartlite\_0\_exdes\_tb;

// Internal signals

parameter clk\_freq = 10;

reg clk;

reg rst\_n;

reg tx\_release;

reg [7:0] rx\_data;

reg rx\_valid;

wire rx\_ready;

wire [7:0] tx\_data;

reg tx\_valid;

reg tx\_ready;

wire fifo\_wren;

wire fifo\_rden;

wire fifo\_full;

wire fifo\_empty;

// Clock process to generate clock signals

always begin

# (clk\_freq / 2) clk = ~clk;

end

// Reset signals

initial begin

clk = 1'b1;

rst\_n = 1'b0;

#50 rst\_n = 1'b1;

end

// DUT instantiation

fifo\_8b u\_fifo\_8b (

.clk(clk), // input wire clk

.srst(~rst\_n), // input wire srst

.din(rx\_data), // input wire [7 : 0] din

.wr\_en(fifo\_wren), // input wire wr\_en

.rd\_en(fifo\_rden), // input wire rd\_en

.dout(tx\_data), // output wire [7 : 0] dout

.full(fifo\_full), // output wire full

.empty(fifo\_empty) // output wire empty

);

// simulation code

initial begin

rx\_data = 8'h25;

rx\_valid = 1'b0;

tx\_ready = 1'b1;

tx\_release = 1'b0;

// oper after rst release

#80 tx\_release = 1'b0;

// your code here

#10;

rx\_valid = 1'b1;

rx\_data = 8'h12;

#10;

rx\_valid = 1'b0;

// your code end

// this signal used for release tx\_ready

#80 tx\_release = 1'b1;

#10 tx\_release = 1'b0;

end

always @(posedge clk) begin

if(tx\_valid)

tx\_ready <= 1'b0;

else if (tx\_release)

tx\_ready <= 1'b1;

end

// loop code here

assign fifo\_wren = rx\_valid && ~fifo\_full;

assign fifo\_rden = ~fifo\_empty && tx\_ready;

always @(posedge clk or negedge rst\_n)

if (!rst\_n)

tx\_valid <= 1'b0;

else

tx\_valid <= fifo\_rden;

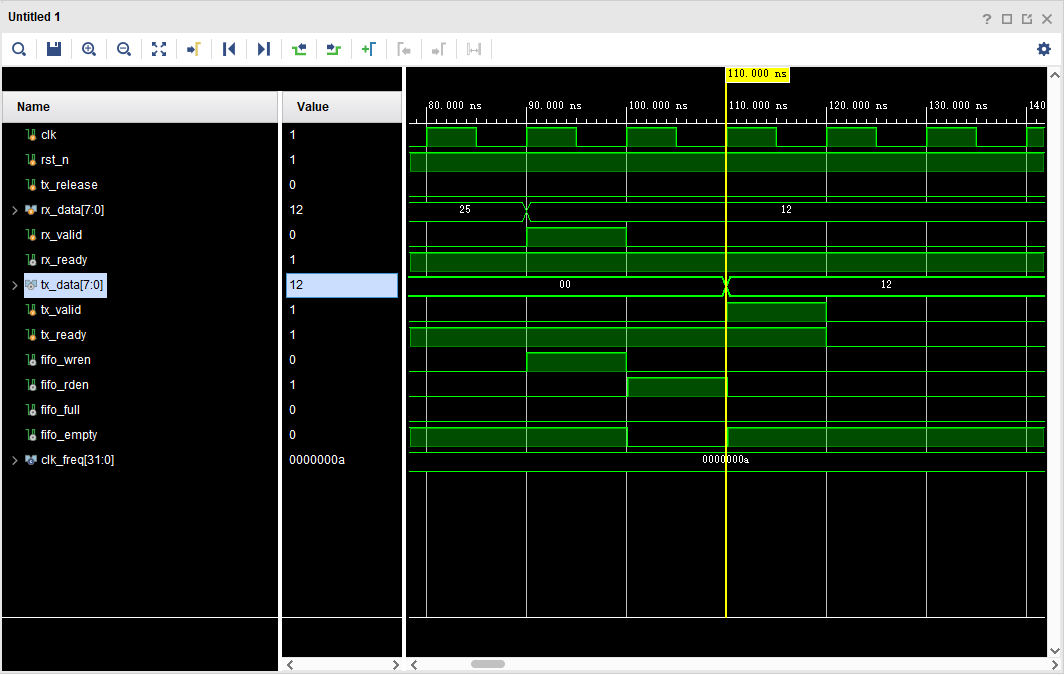
assign rx\_ready = ~fifo\_full;

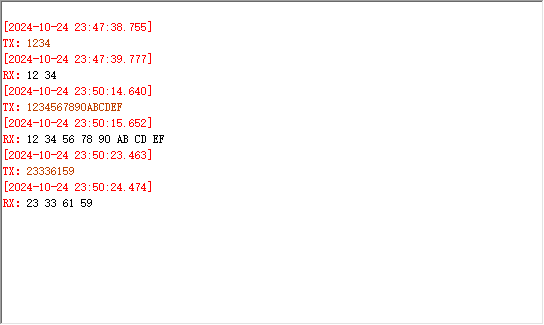
// loop code end

endmodule

1. **结果分析**

仿真图如下： 分析知图像符合实验要求。



串口调试助手结果：成功实现实验要求。

1. **总结**

本次实验实现串口通讯，进一步加深了对verilog代码的理解，掌握了串口原理和时序协议标准及测试串口通讯的方法。