

**数 字 电 路**

**实验报告**

**班级：**教221

**组号：----**

**姓名：**唐嘉良

**学号：**2020K8009907032

**实验名称：**FIFO实验

**2021年12月9日**

**一、实验目的**

1、 熟悉 verilog 编程、调试

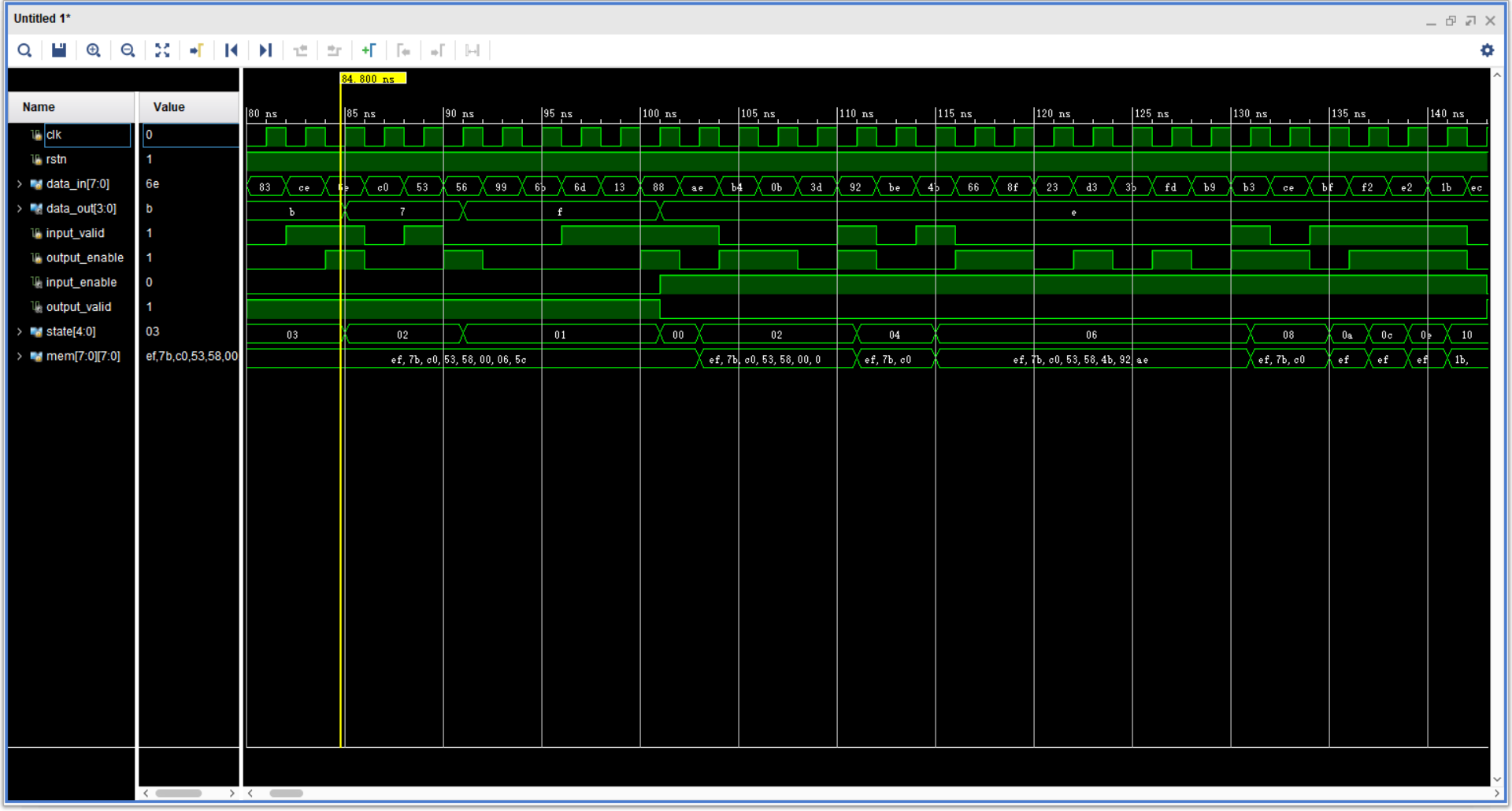
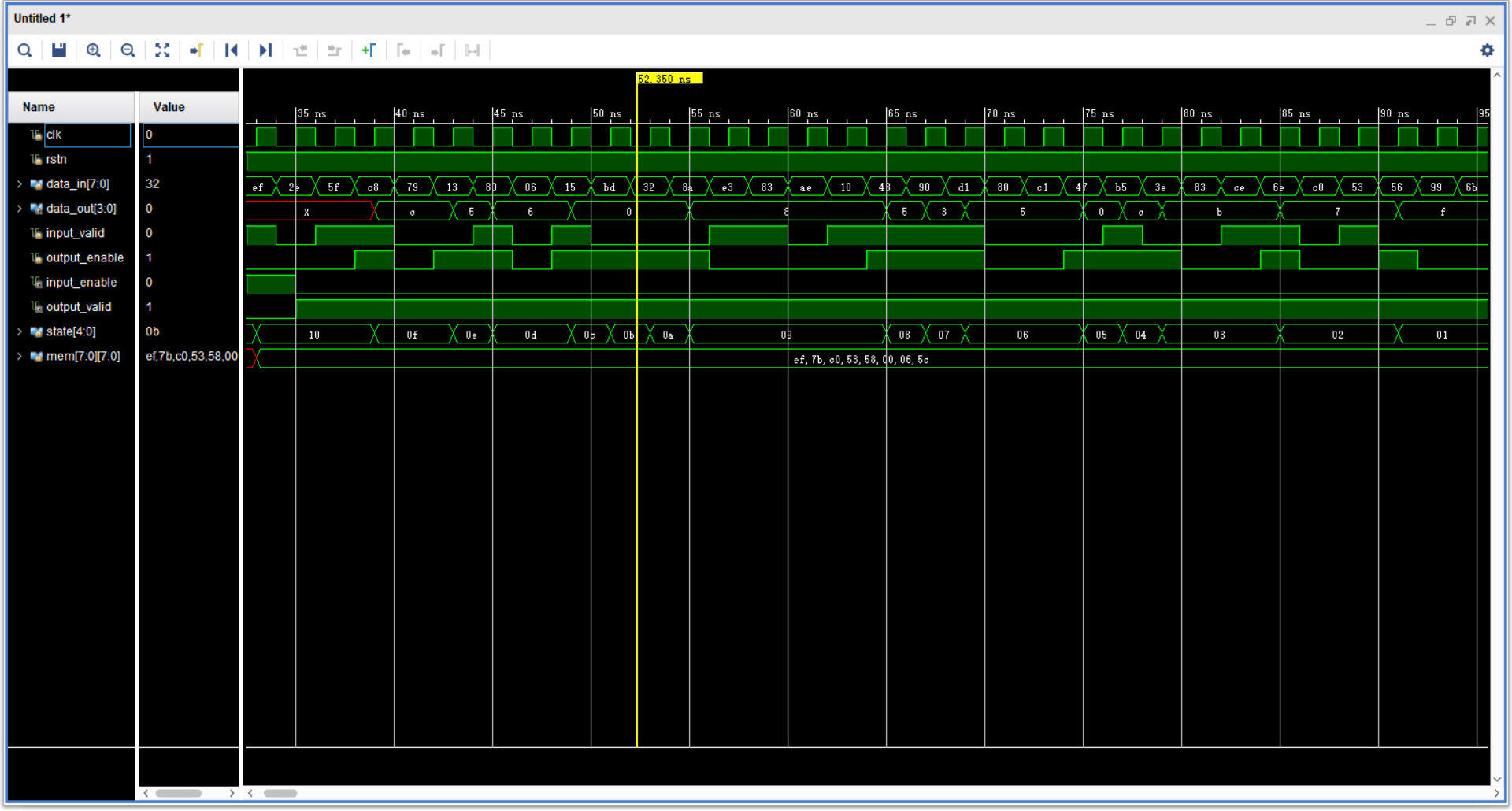
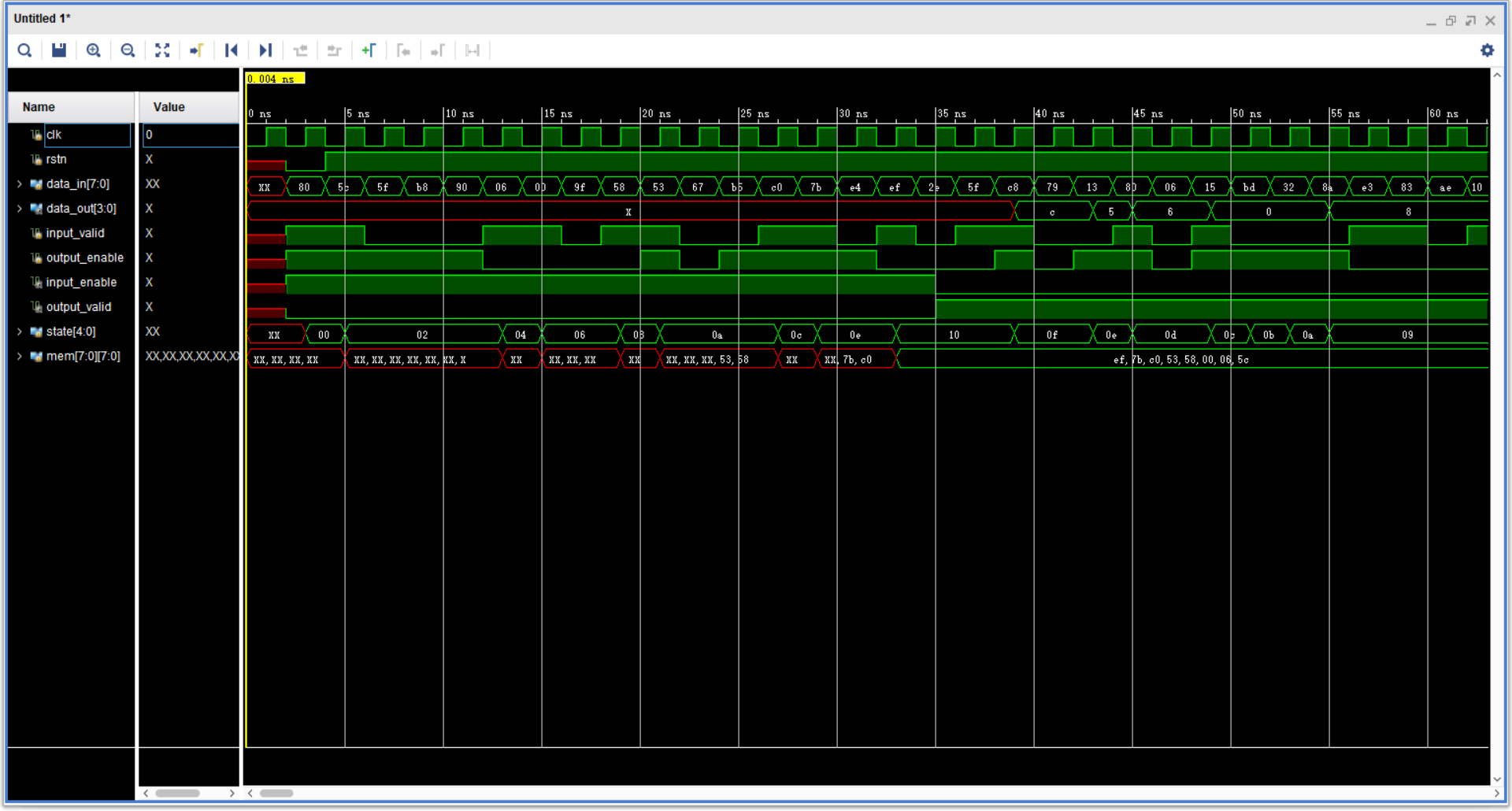
2、 熟悉 FIFO 工作原理

3、 实现功能较复杂的数字电路

1. **实验环境**

本次实验我采用的是vivado 2017.4版本。

1. **调试过程**

FIFO

1. **实验总结**

在此次实验中，我更加熟悉vivado平台的操作流程，现在能够创建激励文件并进行调试。同时，通过构建同步FIFO，我对Verilog语言的掌握程度大大提升，能够更加熟练自如地对数据进行基本的读写操作。

1. **源代码**

**FIFO**

**源代码**

module FIFO(

input [0:0] clk,

input [0:0] rstn,

input input\_valid,

input output\_enable,

output reg input\_enable,

output reg output\_valid,

input [7:0] data\_in,

output reg [3:0] data\_out

);

reg [4:0] state;

reg [7:0] mem [7:0];

reg [2:0] write\_addr;

reg [3:0] read\_addr;

always @(posedge clk or negedge rstn)begin

if(rstn == 0)begin

write\_addr <=3'b0;

input\_enable<=1'b1;

state<=0;

output\_valid<=1'b0;

read\_addr <=4'b0;

end

else if(rstn)

if(input\_valid==1 && input\_enable==1)

begin

if(state==16)

begin

end

else

begin

mem[write\_addr][7:0]<=data\_in;

state<=state+2;

write\_addr<=write\_addr+1;

end

end

end

always@(posedge clk or negedge rstn)begin

if((state==16 && input\_valid==1 && input\_enable==1)||(output\_valid==1 && output\_enable==1))

begin

if(read\_addr[0]==0)begin

data\_out<=mem[read\_addr>>1][3:0];

read\_addr<=read\_addr+1;

state<=state-1;

end

else if(read\_addr[0]==1)begin

data\_out<=mem[read\_addr>>1][7:4];

read\_addr<=read\_addr+1;

state<=state-1;

end

end

end

always@(posedge clk or negedge rstn)begin

if(input\_valid==1&&input\_enable==1)

begin

end

else if(output\_valid==1&&output\_enable==1)

begin

end

else

state<=state;

end

always@(posedge clk or negedge rstn)

if(state==16&&input\_enable==1)

begin

output\_valid<=1;

input\_enable<=0;

end

always@(posedge clk or negedge rstn)

begin

if(state==1&&output\_valid==1&&output\_enable==1)

begin input\_enable<=1;

output\_valid<=0;

end

end

endmodule

**激励文件**

module test\_FIFO(

);

reg clk,rstn;

reg [7:0] data\_in;

wire [3:0] data\_out;

reg input\_valid;

reg output\_enable;

wire input\_enable;

wire output\_valid;

FIFO inst\_FIFO(

clk,

rstn,

input\_valid,

output\_enable,

input\_enable,

output\_valid,

data\_in,

data\_out);

always begin

#1 clk=~clk;

end

initial begin

clk=0;

end

initial begin

#2 rstn=0;

#2 rstn=1;

end

always begin

#2;

data\_in<=$random()%255;

input\_valid<=$random()%2;

output\_enable<=$random()%2;

end

endmodule