中国科学技术大学计算机学院
《计算机组成原理实验》报告

实验题目：单周期 CPU 设计

学生姓名：阿卜杜赛米江·萨吾提

学生学号：PB19111752

完成日期：2021-5-7

计算机实验教学中心制

2020 年 09 月

【实验题目】

单周期 CPU 设计

【实验步骤】

1.修改 Lab2 寄存器堆模块，增加 1 个用于调试的读端口，且使其 r0 内容恒为 0

2. 结构化描述单周期 CPU，并进行功能仿真

3. 将 CPU 和 PDU 下载至 FPGA 中测试，使用 Lab3 实验内容 3 生成的 COE 文件对指令存储器和数据存储器初始化

【实验环境】

VIVADO 软件

【实验各模块】

1.数据通路代码：

```
module DATA_PATH(
    input clk,
   output [31:0] curpc,
    output [31:0] nextpc,
    output [31:0] immgen,
    output [31:0] pcadder,
    output [31:0] alures,regread1,regread2,mux1_o,
    output [31:0] instrc,
    /*output [31:0] pcadd_o,
    output [31:0] bran_o,
    output  selct_of_muxto,
```

```
    output [2:0] alucnt,
    output [31:0] regwritedata,*/

    //pdu block
   //IO_BUS
  input [7:0] io_addr,
  input [31:0] io_dout,
  input io_we,
  output [31:0] io_din,

  //Debug_BUS
  output [7:0] m_rf_addr,
  input [31:0] rf_data,
  input [31:0] m_data,
  input [31:0] pc
    );
  wire [2:0] alu_op,alu_cnt;
  wire
branch,memread,memtoreg,memwrite,alusrc,regwrite,jal,zero;
  wire [31:0]
next_pc,current_pc,instr,mux1_out,alu_result,mem_read_data,
  pc_adder_out,shft_l_out,branch_adder_out;
```

```
  wire [6:0] control_instr;
  wire [31:0]
imm_gen,reg_write_data,reg_read_data1,reg_read_data2;
  wire selct_of_mux2;
  wire [31:0] mux3_out;



    //pc
    PC
pc1(.clk(clk),.next_pc(next_pc),.current_pc(current_pc));
     //instruction memory

     INSTR_MEM IM(.pc(current_pc),.instruction(instr));

    //control unit
    CONTROL
control(.control_instr(instr[6:0]),.aluop(alu_op),.branch(b
ranch)
    ,.memread(memread),.memtoreg(memtoreg),.memwrite(memwri
te),.alusrc(alusrc)
    ,.regwrite(regwrite),.jal(jal));
```

```
    //register file

    REGISTER REG1(
        .clk(clk),
        .reg_write_en(regwrite),
        .reg_write_dest(instr[11:7]),
        .reg_write_data(mux3_out),
        .reg_read_addr1(instr[19:15]),
        .reg_read_addr2(instr[24:20]),
        .reg_read_data1(reg_read_data1),
        .reg_read_data2(reg_read_data2),
        .reg_read_addr3(m_rf_addr),
        .reg_read_data3(rf_data)
    );



//imm gen
   IMM_GEN imm_gen1(.instr(instr),.imm(imm_gen));

//mux1
    MUX
```

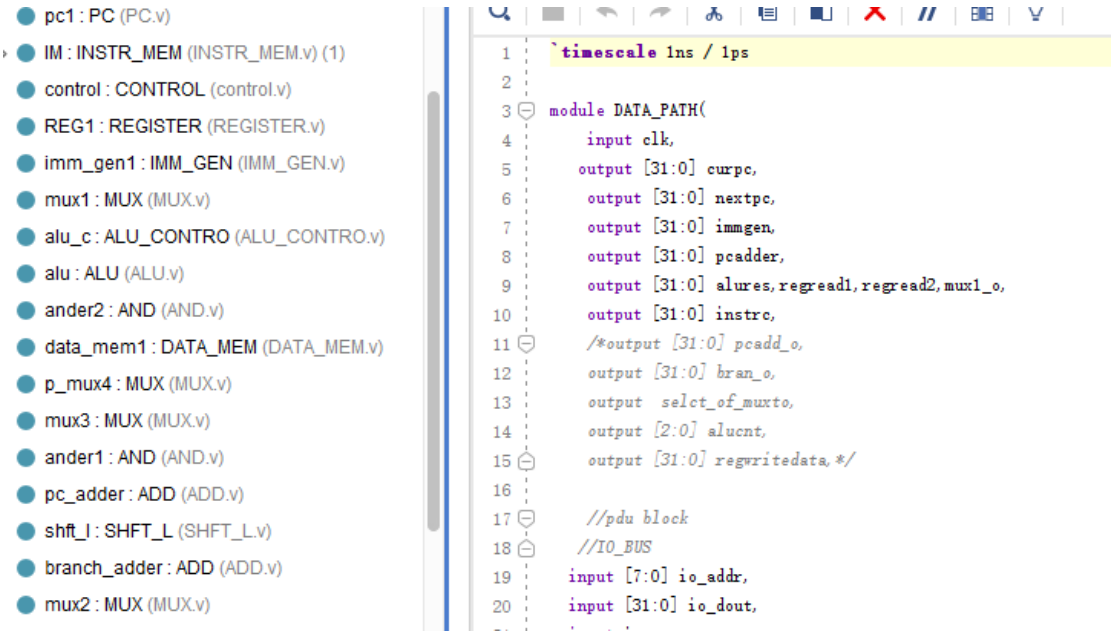
```
mux1(.a(reg_read_data2),.b(imm_gen),.selct({1'b0,alusrc}),.
c(mux1_out));

//alu controler
    ALU_CONTRO
alu_c(.alu_op(alu_op),.opcode(instr[6:0]),.alu_cnt(alu_cnt)
);

//alu
    ALU
alu(.a(reg_read_data1),.b(mux1_out),.zero(zero),.alu_contro
l(alu_cnt),.alu_result(alu_result));



//and2
    wire and_out2;
    AND
ander2(.a(memwrite),.b(~io_addr[10]),.out(and_out2));
//data_memory

    DATA_MEM
data_mem1(.clk(clk), .w_en(and_out2),.r_en(memread)
```

```
        ,.adr1(reg_read_data1)
        , .writedata(reg_read_data2),.rd1(mem_read_data)
        ,.adr2(m_rf_addr),.rd2(m_data));

//p_mux4
    wire [31:0] mux4_out;
   MUX
p_mux4(.a(mem_read_data),.b(io_din),.selct({2'b0,io_addr[10
]}),.c(mux4_out));

//mux3
    MUX
mux3(.a(alu_result),.b(mux4_out),.d(pc_adder_out),.selct({1
'b0,memtoreg}),.c(mux3_out));

//and1
    wire and_out1;
    AND ander1(.a(branch),.b(zero),.out(and_out1));

//pc add
    ADD
pc_adder(.a(current_pc),.b(32'd1),.sum(pc_adder_out));
```

```
//shift left
    SHFT_L shft_l(.a(imm_gen),.b(shft_l_out));

//branch add
    ADD
branch_adder(.a(current_pc),.b(shft_l_out),.sum(branch_adde
r_out));
//get the or of branch and jal
   assign  selct_of_mux2=and_out1||jal;
//mux2
    MUX
mux2(.a(pc_adder_out),.b(branch_adder_out),.selct({1'b0,sel
ct_of_mux2}),.c(next_pc));
 //test


assign curpc=current_pc;
assign nextpc=next_pc;
assign immgen=imm_gen;
assign pcadder=pc_adder_out;
assign alures=alu_result;
```

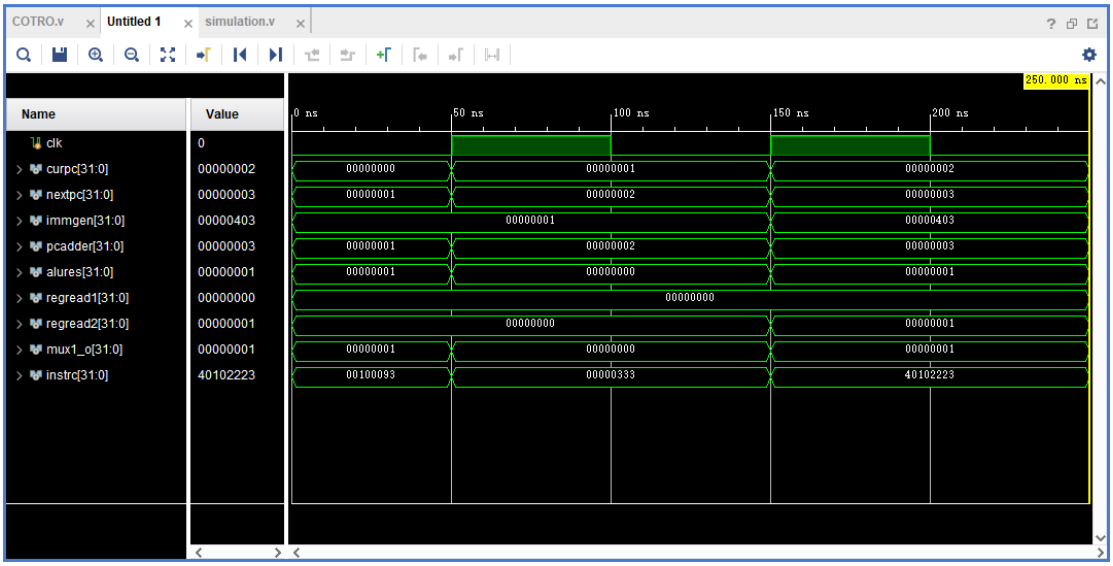
```
assign instrc=instr;

/*assign pcadd_o=pc_adder_out;

assign bran_o=branch_adder_out;

assign selct_of_muxto=selct_of_mux2;

assign alucnt=alu_cnt;

assign regwritedata=mux3_out;*/

assign regread1=reg_read_data1;

assign regread2=reg_read_data2;

assign mux1_o=mux1_out;



//pdu_block


assign pc=current_pc;

assign io_addr=alu_result[7:0]  ;

assign io_dout=reg_read_data2;

assign io_we=io_addr[10]&&memwrite;


endmodule
```

2. 数据通路相关模块：

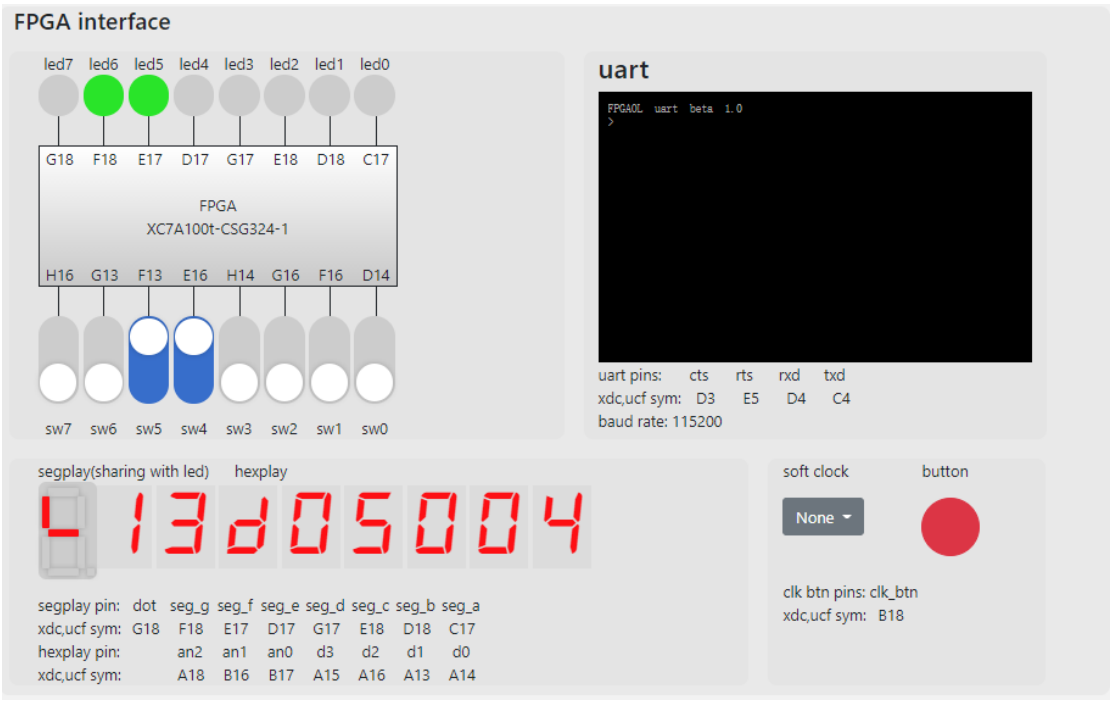


（因为代码文件实在太多，所以截了个图。。。。。）

3. 仿真结果



4.



【总结与思考】

通过这次的实验对单周期risc-v cpu有了很深入的了解,verilog代码能力也有了很大提升!