杭州电子科技大学

计算机组成原理（甲）

实 验 报 告

学　　院	网络空间安全学院
专　　业	网络工程
班　　级	19272401
学　　号	19061440
学生姓名	F001
教师姓名	袁理峰
完成日期	2020.12.26
成　　绩	

实验七 取指令与指令译码实验 （实验名称）

一、 实验目的

（1） 学习指令存储器的设计

（2） 掌握 CPU 取指令操作与指令译码的方法和过程

二、 实验原理

设计一个指令寄存器，只读，物理大小 32*64 位

设计 PC 及其自增电路；

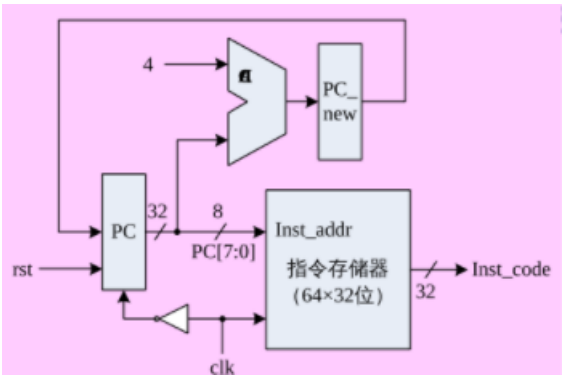
MIPS 地址 32 位，按字节编址，指令存储器：256*8 位

最终目标：设计一个单周期 MIPS CPU

在指令周期 clk 上跳沿，执行取指令操作，在 clk 下降沿更新 PC 值。

复位信号 rst=1 时，PC 清零，即指定 MIPS CPU 从 0 号主存开始执行程序

生成只读的指令存储器时，使用 Memory IP 核，同实验 5，但是选择 Single Port ROM



三、 实验环境

所用电脑的软硬件配置：自己的笔记本电脑、Windows10 操作系统

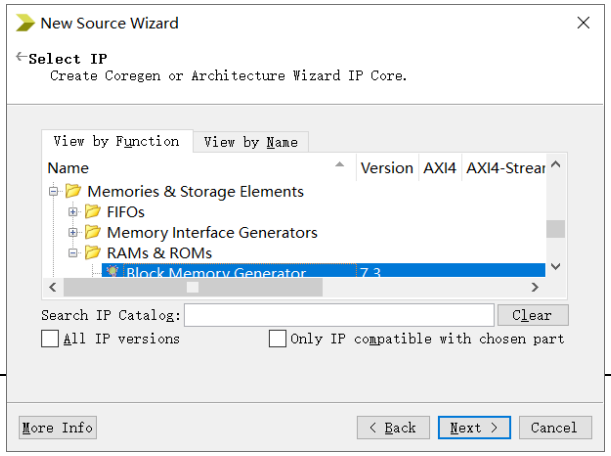
实验所用的软件：ISE design suite

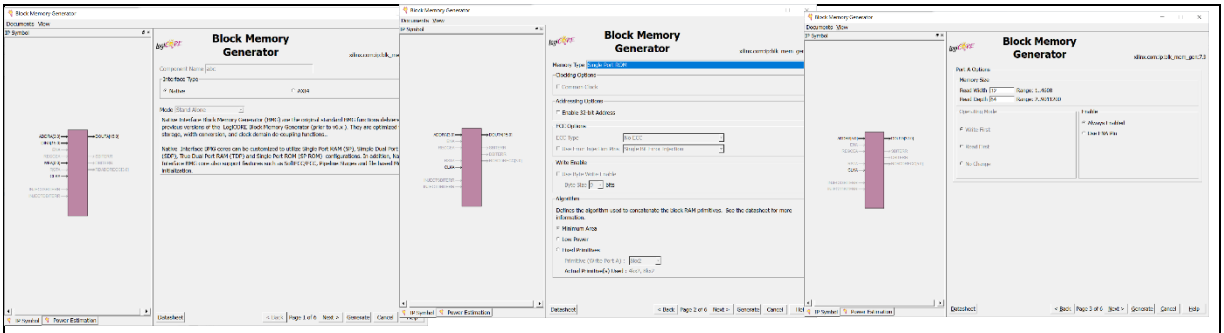
四、 主要操作步骤及实验结果记录（不能光截图，要有相应的文字说明）

（对实验过程中的主要操作步骤进行描述，并随时记录实验过程中观察到的结果，必要时可辅助截图）

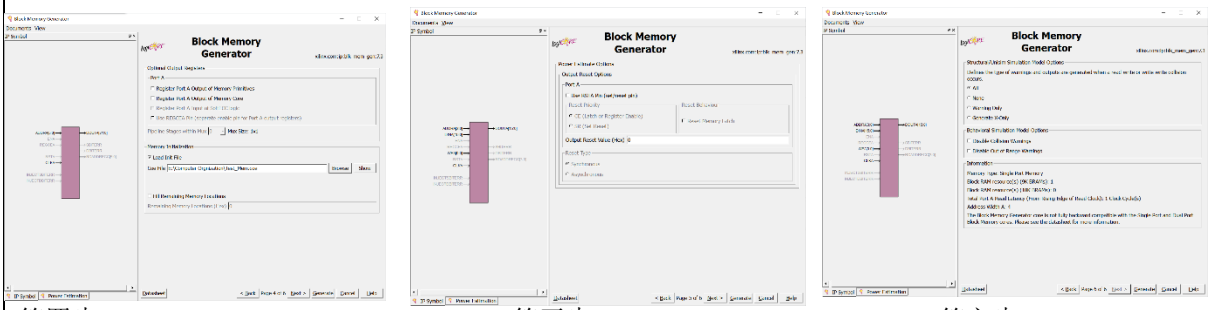
任务一：在 ISE 中使用 Memory IP 核生成一个 Inst_ROM，当作指令存储器，并且关联一个实验六生成的*.coe 文件

以下为生成 ROM 的相关操作记录





第一步　　第二步　　第三步



第四步　　第五步　　第六步

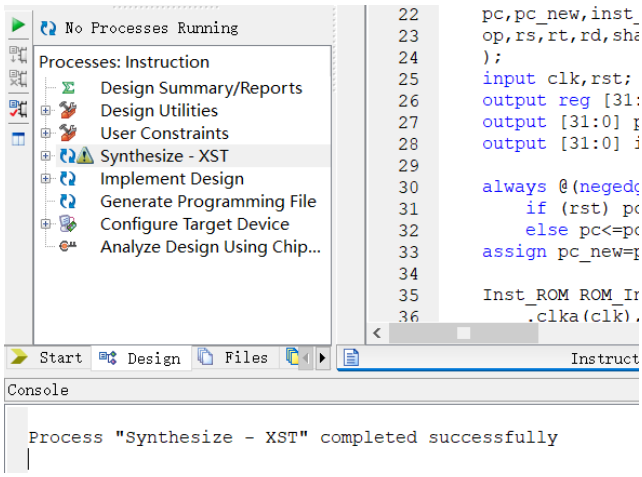
```
21  module Instruction(clk,rst,
22      pc,pc_new,inst_code,
23      op,rs,rt,rd,shamt,func,imm,addr
24      );
25      input clk,rst; //
26      output reg [31:0] pc;
27      output [31:0] pc_new;
28      output [31:0] inst_code;
29
30      always @(negedge clk)
31          if (rst) pc<=32'h0000_0000;
32          else pc<=pc_new;
33      assign pc_new=pc+4;
34
35      Inst_ROM ROM_Instruction_Instance(
36          .clka(clk),
37          .addra(pc[7:2]),
38          .douta(inst_code));
39
40      output [5:0] op,func;
41      output [4:0] rs,rt,rd,shamt;
42      output [15:0] imm;
43      output [25:0] addr;
44      assign op=inst_code[31:26];
45      assign rs=inst_code[25:21];
46      assign rt=inst_code[20:16];
47      assign rd=inst_code[15:11];
48      assign shamt=inst_code[10:6];
49      assign func=inst_code[5:0];
50      assign imm=inst_code[15:0];
51      assign addr=inst_code[25:0];
52  endmodule
```

任务二：编程实现取指令模块，调用 Inst_ROM 指令存储器模块，在此基础上编写一个实验验证的顶层模块

由于 ISE 中不支持 UTF-8 的编码，因此在这里书写注释。

第 25 行表示时钟及重置信号
第 26 行表示 Program Counter，即程序计数器
第 27 行表示 PC 自增值
第 28 行表示机器指令码
第 31 行表示同步清零操作
第 32 行表示更新 PC
第 33 行表示组合逻辑，暂存 PC 自增值
第 35 行是实例化指令存储器 IP 核
第 36 行是输入线 clka
第 37 行是地址线 6 根
第 38 行是输出线 32 根
第 40 行开始是指令分段
第 42 行是 offset 或者立即对于 I 类型
第 43 行是 J 类型的地址

任务四：在 Xilinx ISE 中创建工程，源编码，然后编译、综合

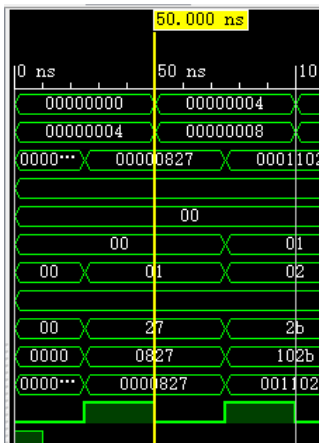


可以看到，该代码通过了语法检查

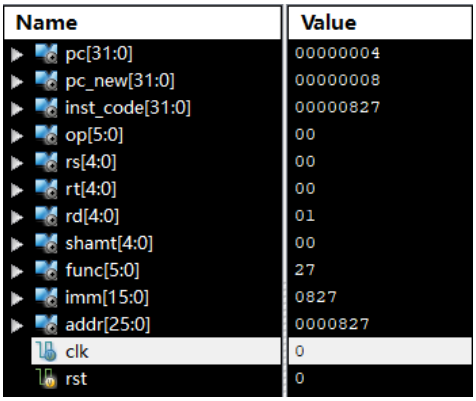
任务五：编写激励代码，观察仿真波形，直至验证正确。

```
    // Instantiate the Unit Under Test (UUT)
    Instruction uut (
        .clk(clk),
        .rst(rst),
        .pc(pc),
        .pc_new(pc_new),
        .inst_code(inst_code),
        .op(op),
        .rs(rs),
        .rt(rt),
        .rd(rd),
        .shamt(shamt),
        .func(func),
        .imm(imm),
        .addr(addr)
    );
    always #25 clk=~clk;
    initial begin
        // Initialize Inputs
        clk = 0;
        rst = 1;
        #10;
        rst=0;
        // Wait 100 ns for global reset to finish

        // Add stimulus here

    end

endmodule
```

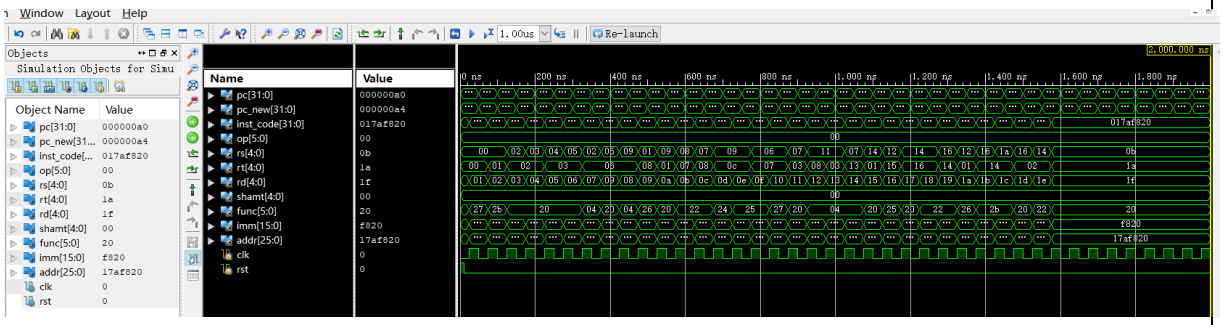
图一



图二



图三



图一为激励代码，显然同样通过了语法检查；上图为仿真波形的全部展示，但难以看清。图三是上图的左侧采用了 16 进制对数据进行展示，图二可以看出每当时钟沿下降时取出相应的地址，实验成功。

五、 实验分析总结及心得

（结合所学知识对实验过程中观察到的实验结果进行分析总结，以便加深对知识的理解，并总结通过实验学到的知识或技术）

实验中，在复制 PCspim 的过程中，发现了地址卡在了 0296b820 中，回忆实验六时，当时也是卡在第 22 条语句之后程序不再执行，因此我重新回去检查了实验六的程序，发现是实验六时，小金同学给我的代码出现了问题，与书上的代码不一样，最后的十条操作全部都是针对寄存器 22 进行的操作，因此发生了栈溢出错误，导致程序无法继续执行。因此重新书写了代码，发现情况正常了。

实验中，一开始的跑出来的结果与书上的结果不一样，经过观察后发现存在 aaaaaaaa，bbbbbbbb 这类特点显著的地址，思考后发现这是实验五的 test 文件所保存的地址值，检查之后发现在导入时导入错了一个 test，因此地址发生了相应的改变。这启示我们在操作时要注意观察目标文件的目录，不要单纯看名字，容易导入成为同名文件但是内容不同。重新导入 coe 文件之后实验正常。

最后对比取出的指令代码与指令存储器关联文件中的指令码是一致的。判断实验成功