CPU 实验报告

班级：智能 1602

学号：201608010722

姓名：孙传骐

实验目标

实现一个单周期 CPU 。

实验要求

- 采用 VHDL 编写程序

CPU 执行指令

CPU 执行指令的流程为：

> 取指-》译码-》执行

设计框架

```
1.  参数定义 0
```

```
entity cpu is
    port(
        clk: in std_logic;
        reset: in std_logic;
        inst_addr: out std_logic_vector(31 downto 0);
        inst: in std_logic_vector(31 downto 0);
        data_addr: out std_logic_vector(31 downto 0);
        data_in: in std_logic_vector(31 downto 0);
        data_out: out std_logic_vector(31 downto 0);
        data_read: out std_logic;
        data_write: out std_logic
    );
end entity cpu;
```

;

2. 声明需要使用的变量；

```
architecture behav of cpu is
        signal ir: std_logic_vector(31 downto 0);
        signal pc: std_logic_vector(31 downto 0);

        signal next_pc: std_logic_vector(31 downto 0);

        -- Fields in instruction
        signal opcode: std_logic_vector(6 downto 0);
        signal rd: std_logic_vector(4 downto 0);
        signal funct3: std_logic_vector(2 downto 0);
        signal rs1: std_logic_vector(4 downto 0);
        signal rs2: std_logic_vector(4 downto 0);
        signal funct7: std_logic_vector(6 downto 0);

        signal Imm11_0I : std_logic_vector(31 downto 0);
        signal Imm20_1J : std_logic_vector(31 downto 0);
        signal Imm12_1B : std_logic_vector(31 downto 0);
        signal Imm11_0S : std_logic_vector(31 downto 0);

        signal src1: std_logic_vector(31 downto 0);
        signal src2: std_logic_vector(31 downto 0);
        signal subresult: std_logic_vector(31 downto 0);
```

3. 取地址，拼凑立即数；

```
-- Instruction Fetch 取出各个值，拼凑指令要求的立即数：
inst_addr <= pc;
ir <= inst;

-- Decode
-- Not finished
opcode <= ir(6 downto 0);
rd <= ir(11 downto 7);
funct3 <= ir(14 downto 12);
rs1 <= ir(19 downto 15);
rs2 <= ir(24 downto 20);
funct7 <= ir(31 downto 25);

Imm11_0I <= "11111111111111111111" & ir(31 downto 20) when ir(31)='1' else
                        "00000000000000000000" & ir(31 downto 20);
Imm20_1J <= "111111111111" & ir(31) & ir(19 downto 12) & ir(20) & ir(30 downto 21) when ir(31)='1' else
                        "000000000000" & ir(31) & ir(19 downto 12) & ir(20) & ir(30 downto 21);
Imm11_0S <= "11111111111111111111" & ir(31 downto 25) & ir(11 downto 7) when ir(31)='1' else
                        "00000000000000000000" & ir(31 downto 25) & ir(11 downto 7);
Imm12_1B <= "11111111111111111111" & ir(31) & ir(7) & ir(30 downto 25) & ir(11 downto 8) when ir(31)='1' else
                        "00000000000000000000" & ir(31) & ir(7) & ir(30 downto 25) & ir(11 downto 8);

-- Read operands from register file
src1 <= regs(TO_INTEGER(UNSIGNED(rs1)));
src2 <= regs(TO_INTEGER(UNSIGNED(rs2)));
```

4. 执行对应的操作。
5. 写回；

测试

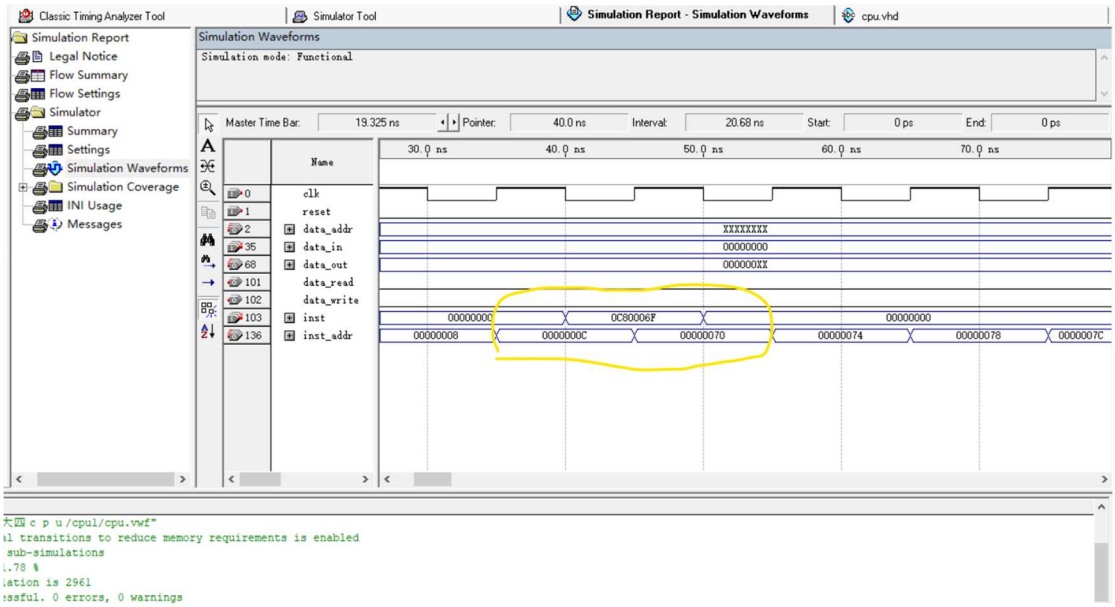
测试平台

模块	配置
CPU	Core i7-6700
操作系统	Windows10
运行环境	C++

测试指令

测试跳转指令，跳到 pc+100 位置。

测试结果



成功实现跳转

结果分析

模拟的给果可认为 CPU 实现。