本次实验测试程序结果不正确大多是因为PC相关的控制信号设计有误，导致PC跑歪，可以按照下面的方法快速定位PC跑歪的位置。

思路很简单，先拿正确的CPU跑一次（比如实验二的CPU），输出WB阶段的PC变化轨迹；然后拿有问题CPU再跑一次，同样输出WB阶段的PC变化轨迹，这样一对比就可以定位到PC出错的位置。

我们可以使用 **fdsiaplay** 函数输出PC信号的值到文件中

- 首先给RV32core添加一个输出信号，引出PC_WB
- 然后在core_sim.v（即仿真的顶层文件）添加输出相关的代码：

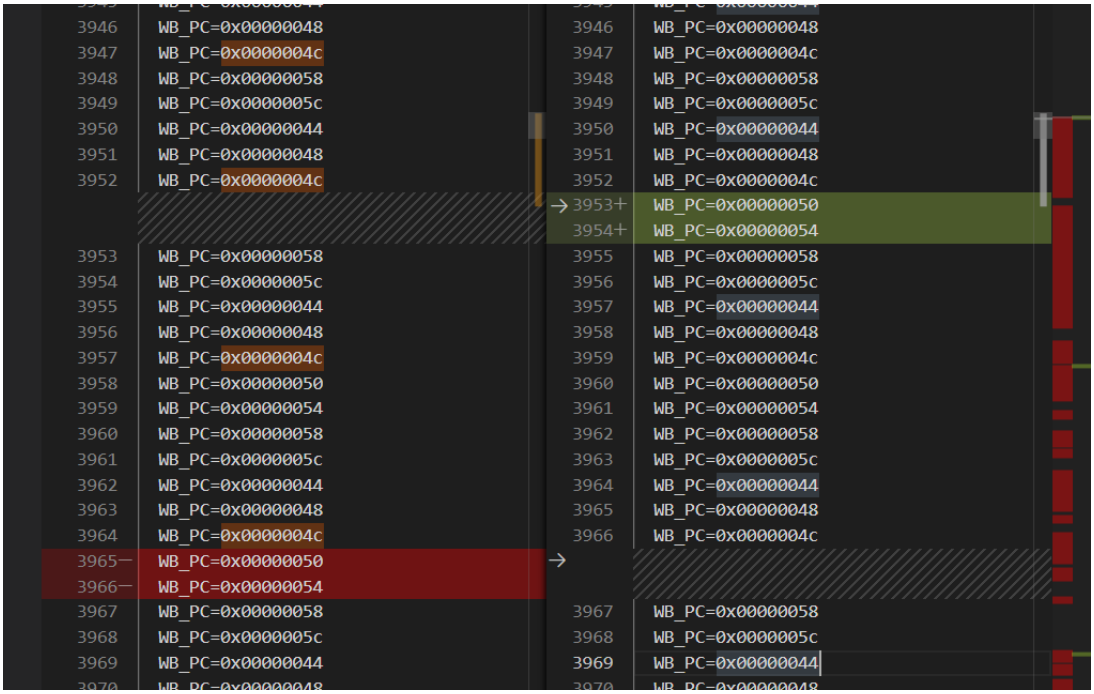
```
module core_sim;
    reg clk, rst;
    wire [31:0] debug_WB_PC;
    // save pc at wb
    reg [31:0] wb_pc;

    RV32core core(
        .debug_en(1'b0),
        .debug_step(1'b0),
        .debug_addr(7'b0),
        .debug_data(),
        .clk(clk),
        .rst(rst),
        .interrupter(1'b0),
        .debug_wb_PC(debug_WB_PC)  //add output here
    );

    integer traceout;
    initial begin
        // open trace file
        traceout = $fopen("trace.out");
        clk = 0;
        rst = 1;
        wb_pc = 0;
        #2 rst = 0;
    end
    always #1 clk = ~clk;

    always@(clk)begin
        if(wb_pc != debug_WB_PC)begin
            // output signal values to file
            $fdisplay(traceout, "  WB_PC=0x%8h",debug_WB_PC );
            wb_pc <= debug_WB_PC;
        end
    end
endmodule
```

用正确和错误的CPU测试后，我们可以得到两个trace文件，然后使用vscode等工具进行比对，可定位出错位置。



例如在这张图里，左侧是正确的，右侧是错误的，我们可以看到0x0000004c后应该跳到0x00000058，但是右侧的并没有。

我们还可以在输出中加入时间信息，以此定位出错的时刻（单位是ns）：

```
$fdisplay(traceout, "[%t]  WB_PC=0x%8h", $time, debug_WB_PC );
```

感兴趣的同学还可以尝试实例化两个CPU，一个正确的和一个待调试的，在仿真运行阶段就比较PC或其他寄存器的信号，这其实就是调试CPU常用的差分测试方法。