寄存器组设计实验

（练习实验二）

设计一个 32 位而拥有 32 个寄存的寄存器组。

1、项目名：RegFile

2、代码文件内容，如下：

```
module RegFile(
        input CLK,
        input RST,
        input RegWre,
        input [4:0] ReadReg1,
        input [4:0] ReadReg2,
        input [4:0] WriteReg,
        input [31:0] WriteData,
        output [31:0] ReadData1,
        output [31:0] ReadData2
        );
        reg [31:0] regFile[1:31]; //   寄存器定义必须用 reg 类型
        integer i;

        assign   ReadData1 = (ReadReg1 == 0) ? 0 : regFile[ReadReg1]; // 读寄存器数据
        assign   ReadData2 = (ReadReg2 == 0) ? 0 : regFile[ReadReg2];

        always @ (negedge CLK or negedge RST) begin     // 必须用时钟边沿触发
            if (RST==0) begin
                for(i=1;i<32;i=i+1)
                regFile[i] <= 0;
            end
            else if(RegWre == 1 && WriteReg != 0) // WriteReg != 0，$0 寄存器不能修改
                regFile[WriteReg] <= WriteData;   // 写寄存器
        end
endmodule
```

3、测试文件内容，如下：

```
module RegFile_sim();
    // Inputs
    reg CLK;
    reg RST;
    reg RegWre;
    reg [4:0] ReadReg1;
    reg [4:0] ReadReg2;
```

```
 reg [4:0] WriteReg;
 reg [31:0] WriteData;

 // Outputs
 wire [31:0] ReadData1;
 wire [31:0] ReadData2;

 // Instantiate the Unit Under Test (UUT)
 RegFile   uut (
       .CLK(CLK),
       .RST(RST),
       .RegWre(RegWre),
       .ReadReg1(ReadReg1),
       .ReadReg2(ReadReg2),
       .WriteReg(WriteReg),
       .WriteData(WriteData),
       .ReadData1(ReadData1),
       .ReadData2(ReadData2)
 );

always #50 CLK = !CLK;
 initial begin
       // Initialize Inputs
       CLK = 0;
       RST = 0;            // 复位
       RegWre = 0;        // 读
       ReadReg1 = 0;
       ReadReg2 = 0;
       WriteReg = 0;
       WriteData = 0;

       // Wait 100 ns for global RST to finish
       #100;    // 写
       RST = 1;
       RegWre = 1;
       ReadReg1 = 0;
       ReadReg2 = 0;
       WriteReg = 1;
       WriteData = 1;

       #100;    // 写
       RST = 1;
       RegWre = 1;
       ReadReg1 = 0;
```

```
ReadReg2 = 0;
WriteReg = 2;
WriteData = 2;

#100;    // 写
RST = 1;
RegWre = 1;
ReadReg1 = 0;
ReadReg2 = 0;
WriteReg = 3;
WriteData = 3;

#100;    // 读
RST = 1;
RegWre = 0;
ReadReg1 = 1;
ReadReg2 = 0;
WriteReg = 0;
WriteData = 0;

#100;    // 读
RST = 1;
RegWre = 0;
ReadReg1 = 0;
ReadReg2 = 2;
WriteReg = 0;
WriteData = 0;

#100;    // 读
RST = 1;
RegWre = 0;
ReadReg1 = 1;
ReadReg2 = 3;
WriteReg = 0;
WriteData = 0;

#200; // 延迟

#100; // 复位
RST = 0;
RegWre = 0;
ReadReg1 = 0;
ReadReg2 = 0;
WriteReg = 3;
```

```
        WriteData = 3;

        // Add stimulus here
    end

endmodule
```

4、相关波形