# Lab4 寄存器与内存

周鼎

April 4, 2016

## 1 Register

### 1.1 Register 结构

Register 需要实现读和写的功能，读 register 时输入两个五位寄存器地址，写 register 时需要 regWrite 信号有效，并选择 clock 下降沿作为同步信号，避免 writeReg,regWrite,writeData 信号的先后次序产生影响。除此之外，为了后续实验的需要，还额外增加了 reset 接口实现寄存器清零功能，和两个额外接口输出 $1,$2 寄存器的值用于上板测试。

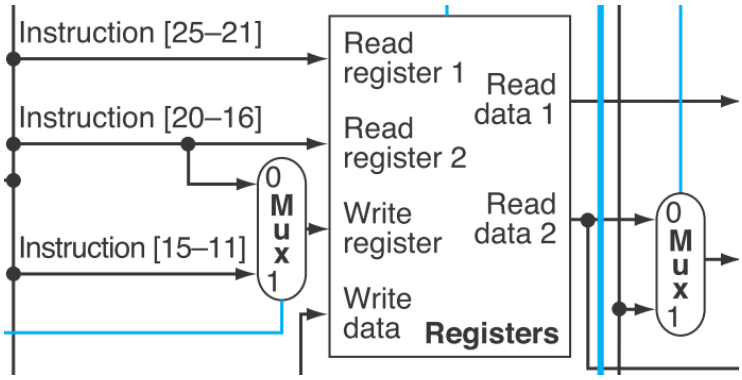


Figure 1: Register

### 1.2 程序实现

```
1  module Register(
2      input clock_in,
3      input regWrite,
4      input [4:0] readReg1,            //address to be read1
5      input [4:0] readReg2,
6      input [4:0] writeReg,            //address to be write
7      input [31:0] writeData,
8      input reset,                    //reset the register
9      output [31:0] readData1,
10     output [31:0] readData2,
11     output [31:0] reg1,             //for IO on board
12     output [31:0] reg2
13     );
14    reg [31:0] regFile[31:0];
```

```
15
16      assign readData1 = regFile[readReg1];
17      assign readData2 = regFile[readReg2];
18      assign reg1 = regFile[1];
19      assign reg2 = regFile[2];
20
21      always @(negedge clock_in)  begin
22        if(reset) $readmemh("./src/regFile.txt",regFile);
23         else if(regWrite) regFile[writeReg] = writeData;
24      end
25  endmodule
```

reg.v

### 1.3 仿真

使用 ModelSim 进行行为级仿真，设置时钟周期 200ns, 仿真过程和仿真结果见 Table??, Figure ??。

| # | regWrite | writeDst | writeData | readReg |
|---|---|---|---|---|
| 1 | 0 | 1 | 255 | 1 |
| 2 | 1 | 1 | 255 | 1 |
| 3 | 1 | 2 | 255 | 2 |
| 4 | 1 | 3 | 233 | 3 |

Table 1: 仿真序列

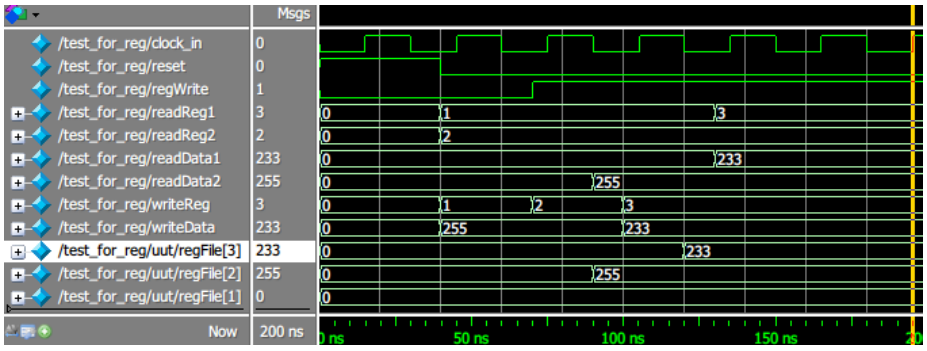


Figure 2: ModelSim Reg 仿真波形

## 2 Memory

### 2.1 Memory 结构

Memory 与 Register 比较类似，同样需要支持读和写操作，不需要 reset，但需要启动时初始化其中的数据。

## 2.2 程序实现

```
module dataMemory(
    input clock_in,
    input [31:0] address,
    input [31:0] writeData,
    output wire [31:0] readData,
    input memWrite,
    input memRead                          //memRead is actually not used
    );
   initial $readmemh("./src/memFile.txt",memFile);     //intialize
    memory

   wire [31:0] index;
   assign index = address>>2;              //data is fetched wordwise
   reg [31:0] memFile [127:0];
   assign readData= memFile[index];

   always @(negedge clock_in)
    if(memWrite)  memFile[index] <= writeData;

endmodule
```

mem.v

## 2.3 仿真

使用 ModelSim 进行行为级仿真，设置时钟周期 200ns, 可见符合在下降沿、memWrite 有效时写入的要求。

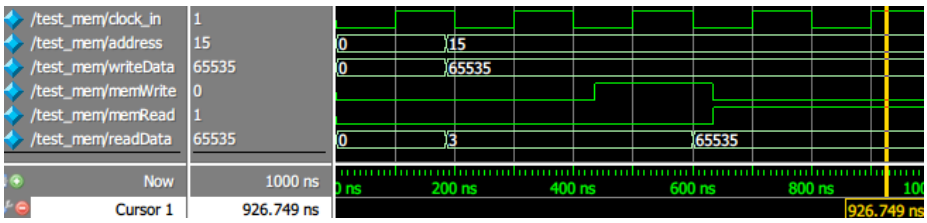


Figure 3: ModelSim Mem 仿真波形

**注**：Index0 和 3 在初始化时分别写入了 0 和 3，