实验四 存储器阵列设计

**一、实验目的**

1 掌握Verilog语言和Vivado、Logisim开发平台的使用；

2 掌握存储器和寄存器组的设计和测试方法。

**二、实验内容**

1 存储器设计与测试

2 寄存器组设计与测试

**三、实验要求**

1 掌握Vivado或Logisim开发工具的使用，掌握以上电路的设计和测试方法；

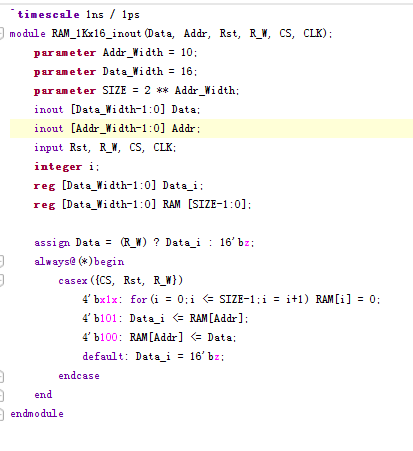
2 记录设计和调试过程（Verilog代码/电路图/表达式/真值表，Vivado仿真结果，Logisim验证结果等）；

3 分析Vivado仿真波形/Logism验证结果，注重输入输出之间的对应关系。

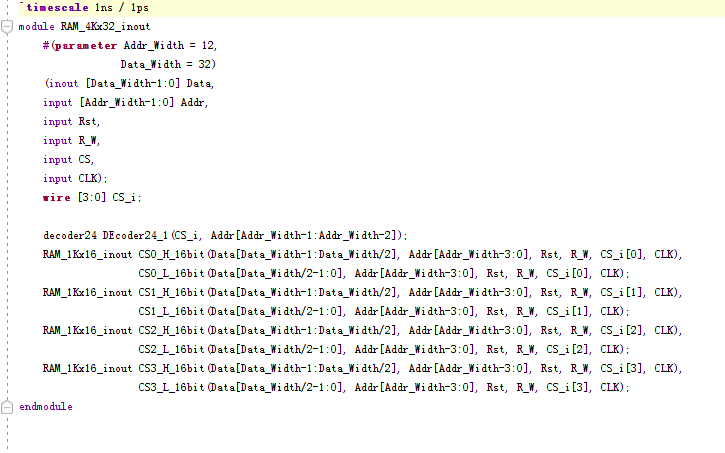
**四、实验过程及分析**

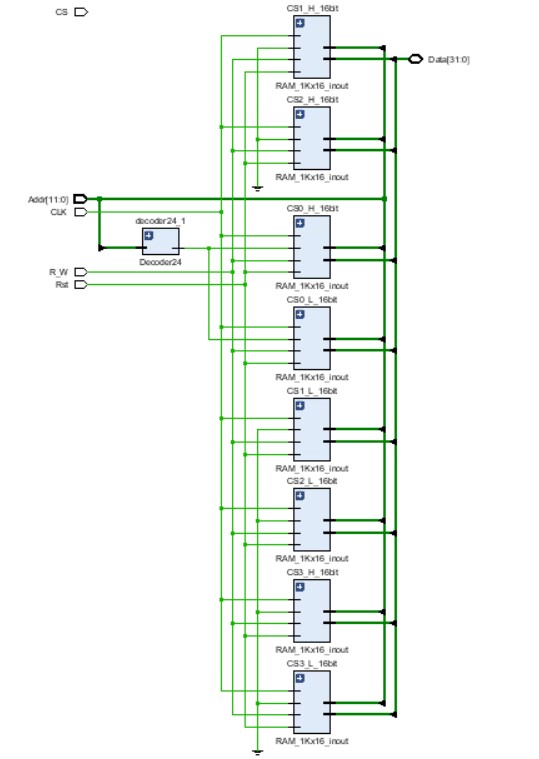
**实验1：用1K×16位芯片构成4K×32位的存储器**

**设计代码：**

 **1X16代码**

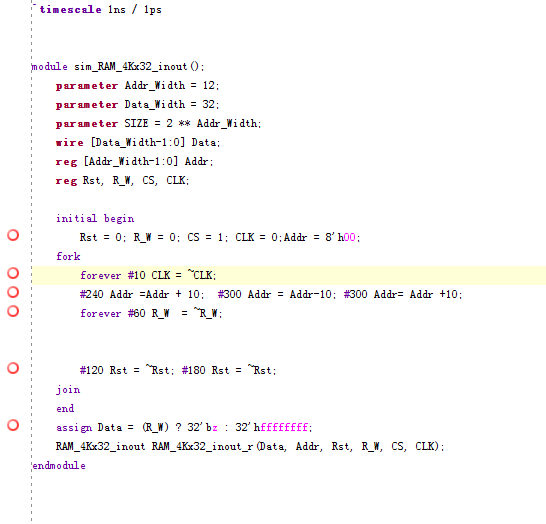
参数化地址线宽和数据线宽。参数化存储器大小为1024，根据CS（片选信号），Rst（复位信号），R\_W(读写信号)判断执行什么操作（读、写、复位）。

**4X32代码** 

通过拓展用8个1K×16位芯片构成4K×32位的存储器，根据34译码器生成的片选信号选择芯片组。

**RTL分析：**

由RTL分析图中可以看出，8个芯片进行字拓展和位拓展。根据24译码器产生的片选信号，选择两个芯片作为Data数组的高16位和低16位。

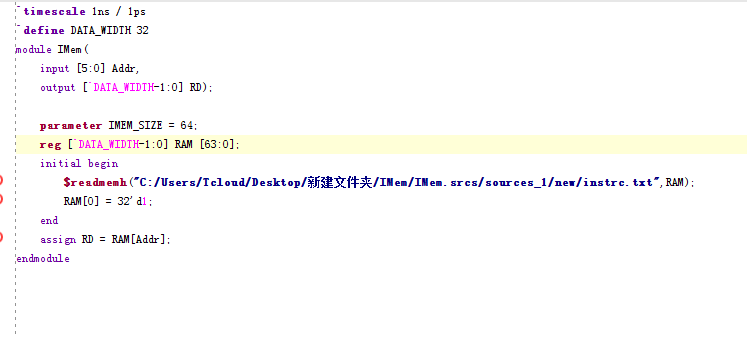
**仿真代码**

仿真代码初始化Rst = 0， R\_W(执行写操作)，Addr置为00，时钟频率位10ns，每隔60ns交替执行一次读和写操作。为体现寄存器被执行了读和写操作，设计Addr规律变化，在初地址执行了两次读操作，另一个地址执行了一次写操作。

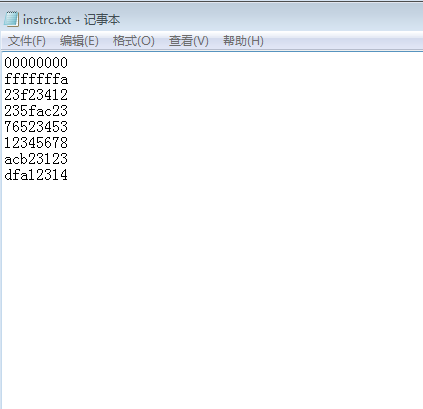
**时序图：**

由时序图中可以看出最开始未进行读写操作时，Data为ffffffff，Rst信号将RAM的存储单元全部清零（data = 00000000），240ns之前data 为00000000，执行的是读操作，将存储器里的00000000读到data中，当240ns之后执行写操作，根据仿真文件可以看出data清置为ffffffff.

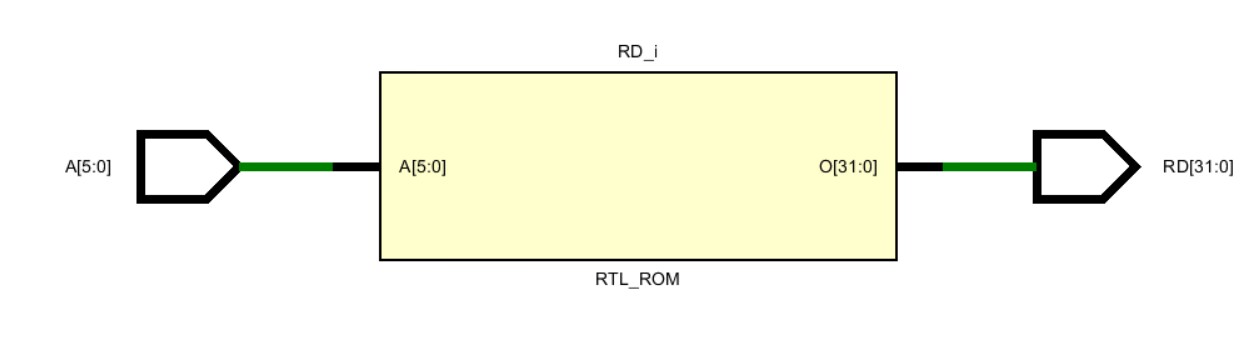
**实验二：指令存储器设计**

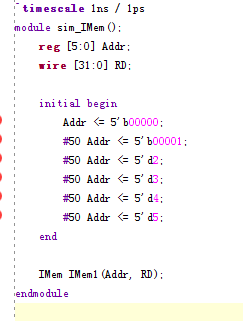
**设计代码：**

使用 $readmemh 函数从文件中读取指令，并将其存储到 RAM 数组中。同时，将 RAM 数组中第一个位置的值设置为 1。最后，使用 assign 关键字将 RAM 数组中指定地址的指令赋值给输出端口 RD，实现了 IMem 模块的功能。



Txt中记录了各指令内容，均为8位16进制数（32位二进制）

**RTL分析：**

**仿真代码：**

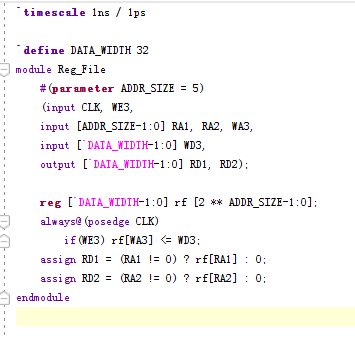
每隔50ns改变一次Addr，读取各个位置的内容。

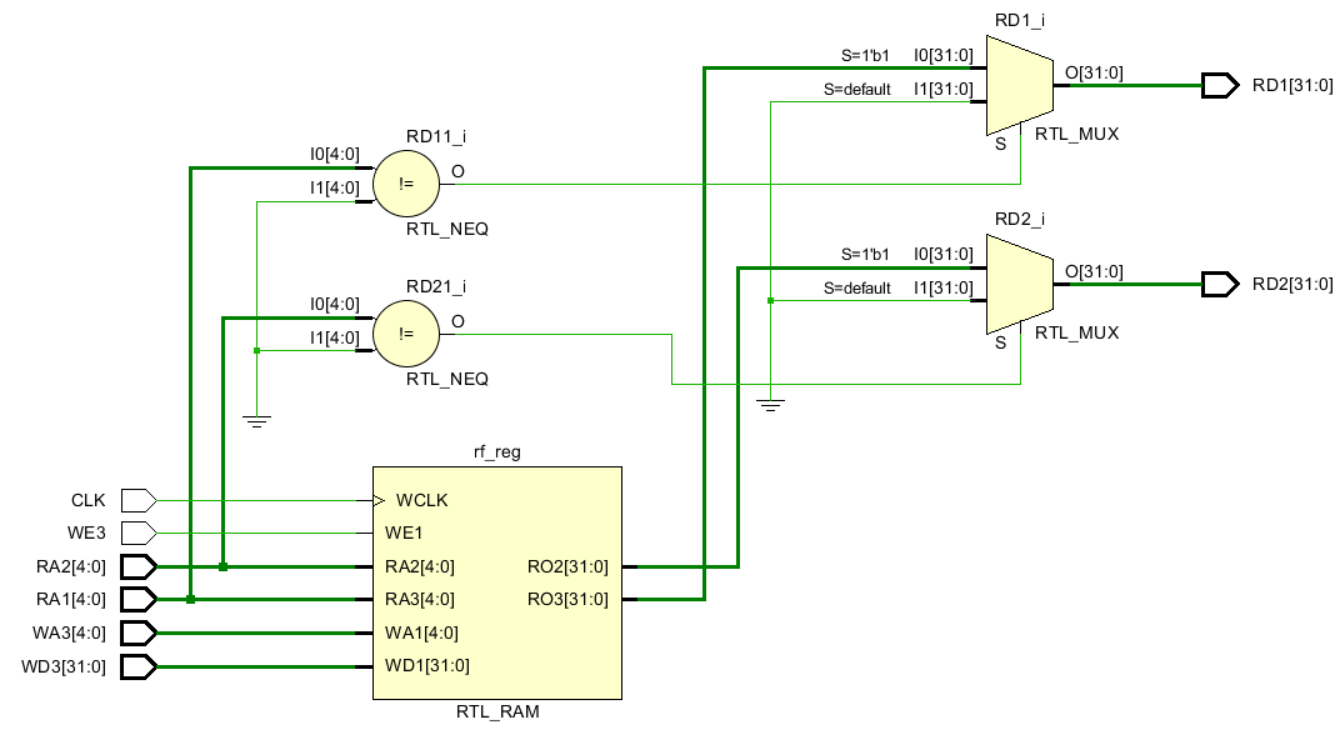
**时序图：**

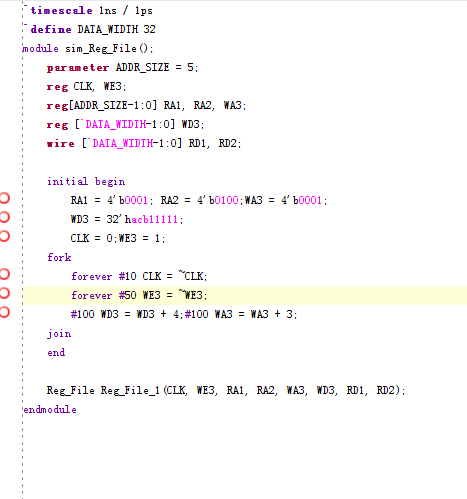
跟txt对应起来看各寄存器内容正确。当Addr = 01时对应txt的第二栏fffffffa，可以由图中看出RD的值也为fffffffa。

**实验三：寄存器文件设计**

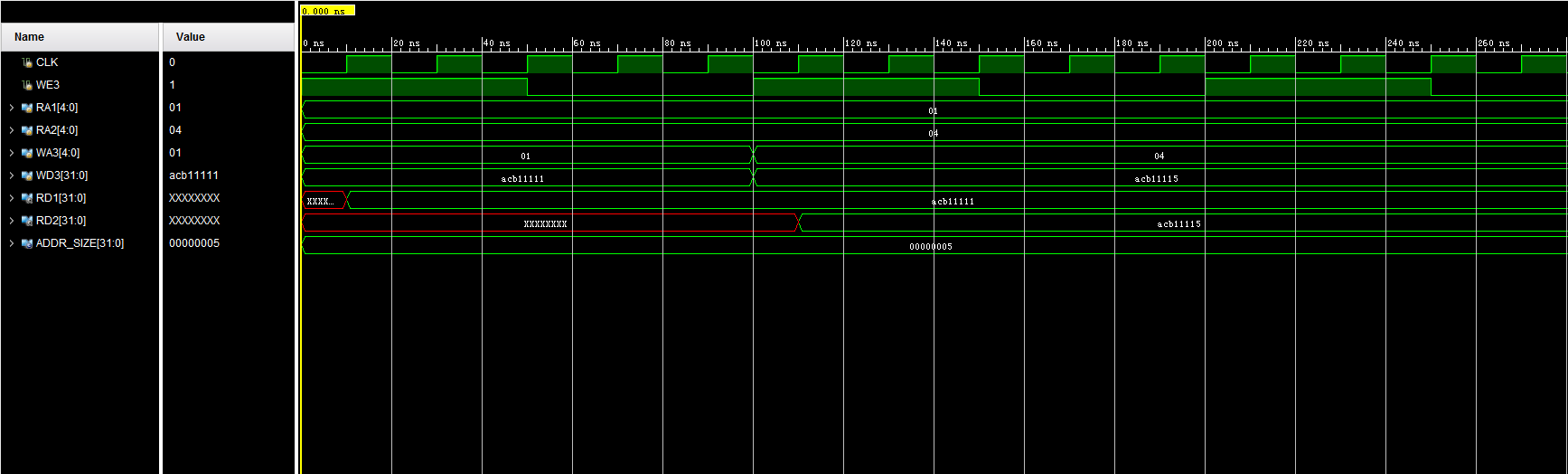
**设计代码：**

RD1和RD2分别对应RA1、RA2地址对应寄存器，当WE3写使能信号有效时，将WD3对应的值放入WA3指向的寄存器地址。

**RTL分析：**

**仿真代码：**

设置RA1的地址为0001，RA2的地址为0100，WA3初始指向地址为0001（RA1的地址）。在100ns时，将WA指向的地址改为0100（RA2的地址）。

**时序图：**

由图中可知，在WA3将初值acb11111赋给了一开始指向的RD1，后来WD3指向RD2，WA

3将修改后的值acb11115赋给了RD2，与仿真文件相匹配。

**五、调试和心得体会**

在这次存储器和寄存器的设计与测试中，我更加深入地了解到了存储器模块端口定义以及如何实现字扩展和位扩展、读和写的操作。并认识到inout类型接口的使用，一端为有效数据，另一端为高阻态，不能同时进行读写操作。