实验四 存储器模块的设计及应用

**一、实验目的**

1. 掌握Verilog语言框架、编程和调试方法。
2. 掌握Verilog中的存储器电路工作原理。
3. 掌握存储器的实际应用。

**二、实验内容**

1. 设计一个静态存储器RAM，容量为256\*32bit。
2. 建立存储器的访问所需要的各种信号。
3. 各种信号的产生。
4. 对存储单元0#,1#,3#,5#,7#,9#或0#，2#，4#，6#，246#进行读写操作。
5. 观察、记录和分析仿真波形。

**三、实验要求**

1. 分析各模块的的程序结构，画出其流程图。
2. 画出模块的电路图。
3. 分析电路的仿真波形，标出关键的数值。
4. 记录设计和调试过程。

**四、实验代码及结果**

**流程图：**

Memory

256\*32bit

Data

Data\_out

读——

cs==1, we==0,oe==1

写——

cs==1, we==1

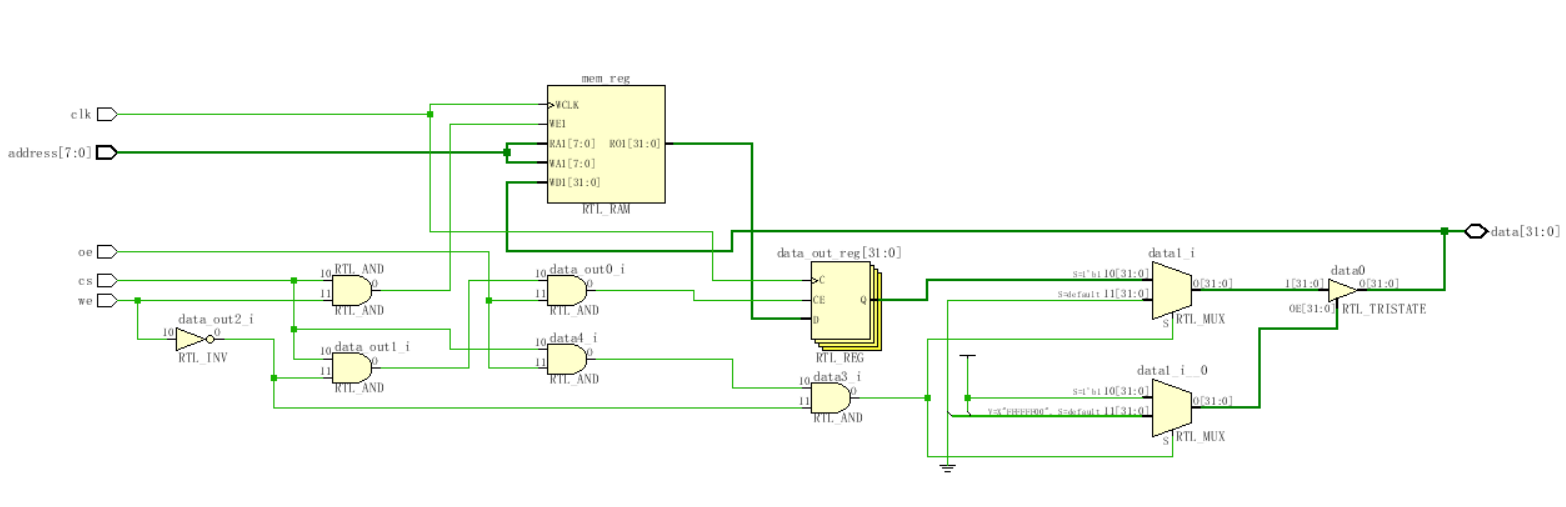
Address

Address

Data

If cs==1, we==0,oe==1

**电路图：**



**器件源代码：**

module ram(

clk,address,data,cs,we,oe

);

parameter DATA\_WIDTH=32;

parameter ADDR\_WIDTH=8;

parameter RAM\_DEPTH=1 << ADDR\_WIDTH;

input clk;

input [ADDR\_WIDTH-1:0] address;

input cs;

input we;

input oe;

inout [DATA\_WIDTH-1:0] data;

reg [DATA\_WIDTH-1:0] data\_out;

reg [DATA\_WIDTH-1:0] mem [0:RAM\_DEPTH-1];

assign data=(cs&&oe&&!we)?data\_out:8'bz;

always @(posedge clk)

begin:MEM\_WRITE

if(cs&&we) begin

mem[address]=data;

end

end

always @(posedge clk)

begin:MEM\_READ

if(cs&&!we&&oe) begin

data\_out=mem[address];

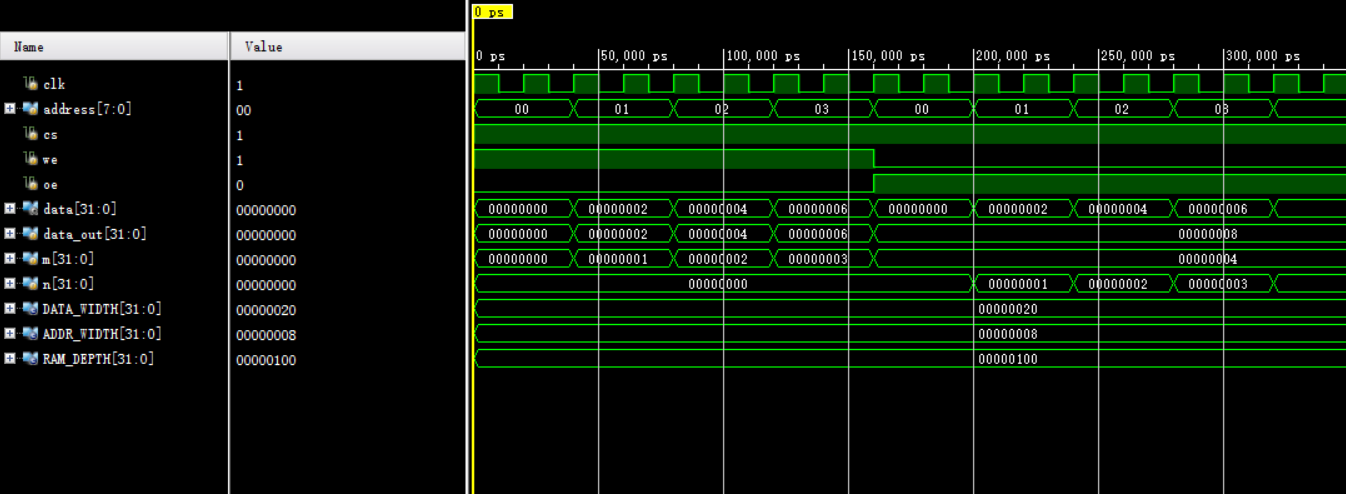
end else begin

end

end

endmodule

**仿真图：**



**仿真源代码：**

module sim\_ram;

parameter DATA\_WIDTH=32;

parameter ADDR\_WIDTH=8;

parameter RAM\_DEPTH=1 << ADDR\_WIDTH;

reg clk;

reg [ADDR\_WIDTH-1:0] address;

reg cs;

reg we;

reg oe;

wire [DATA\_WIDTH-1:0] data;

reg [DATA\_WIDTH-1:0] data\_out;

assign data=(cs&&!oe&&we)?data\_out:32'bz;

initial clk=1'b1;

always

#10 clk<=~clk;

integer m,n;

initial

begin

m=0;

n=0;

cs=1;

we=1;

oe=0;

data\_out=32'b00;

address=8'b00;

end

initial

begin

cs=1;

we=1;

oe=0;

for(m=0;m<4;m=m+1)

begin

#40

address=address+8'b01;

data\_out=data\_out+32'd02;

end

address=8'b00;

cs=1;

we=0;

oe=1;

for(n=0;n<4;n=n+1)

begin

#40

address=address+8'b01;

end

address=8'b00;

end

ram ram\_32(clk,address,data,cs,we,oe);

endmodule

**五、调试和心得体会**

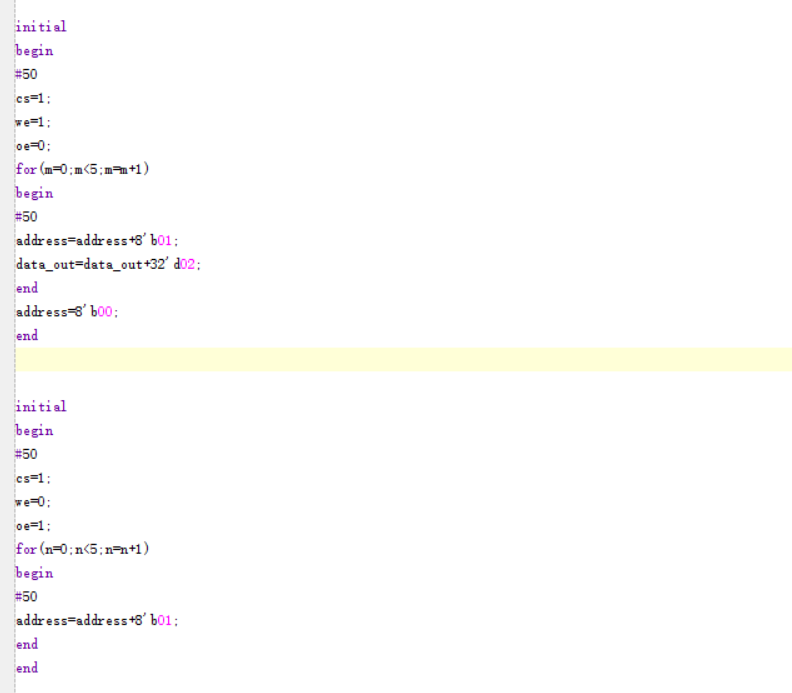
最开始的仿真源代码，总共写了三个initial；

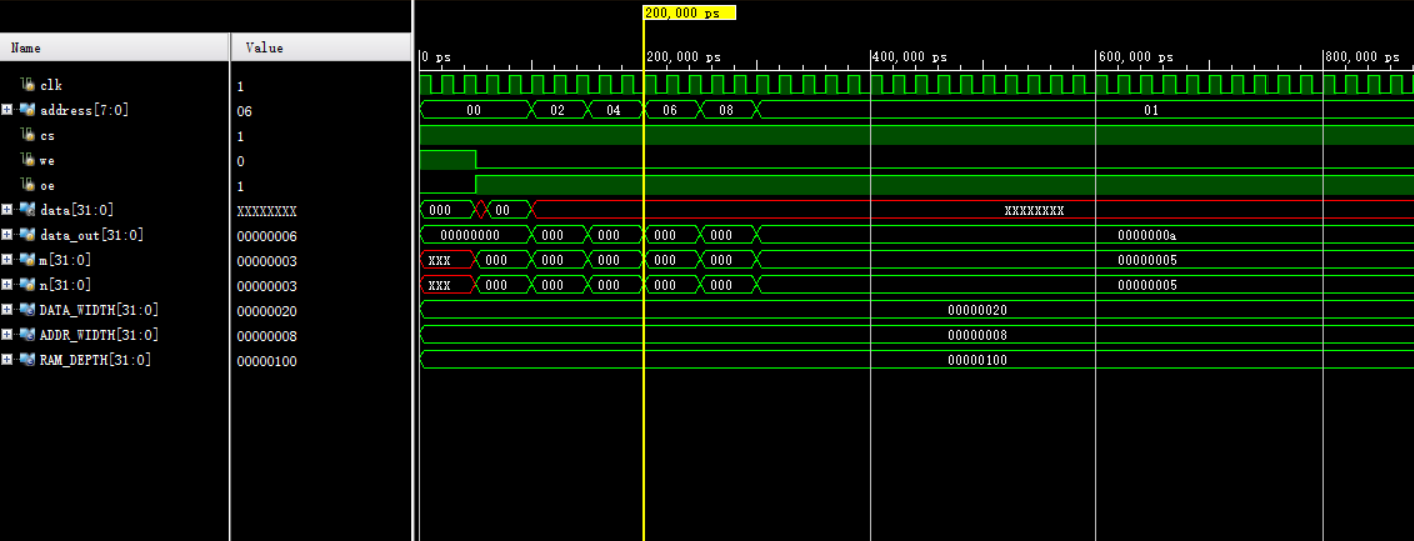
第一个initial初始化cs we oe data\_out address；

第二个initial为写入，cs=1，we=1，oe=0，循环使data\_out，与address自加；

第三个initial为读出，cs=1，we=0，oe=1，循环使address自加；

运行发现，第三个initial中的语句没有执行；





最后查阅资料发现，initial是并行；

也就是多个initial，他们会同时独立运行，从而导致了冲突；

弄懂了这个之后，更改就很简单了，把第三个initial语句中的内容放第二个initial

语句中所有语句的后面，经执行后输出正常。

细节决定成败，我即是因为没有弄懂initial是并行而被卡了很久，自己熟悉语法相关的知识很重要。