笔记本：　FPGA练习
创建时间：　2019/10/31 13:54　　更新时间：　2019/11/1 18:36
作者：　2322900041@qq.com

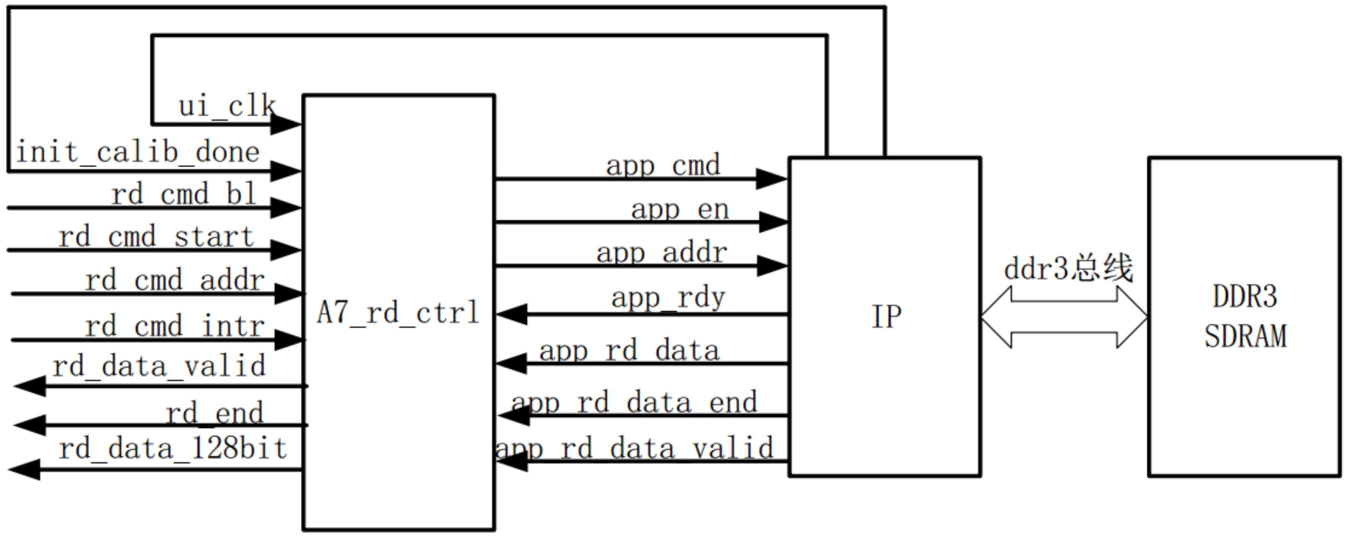
高级练习3　基于 A7 的 DDR3 SDRAM IP 读和仲裁实现

第一部分　DDR3 SDRAM IP 读实现

一、练习内容

ddr3 sdram读实现

二、系统框图



三、设计分析

1.关键信号分析

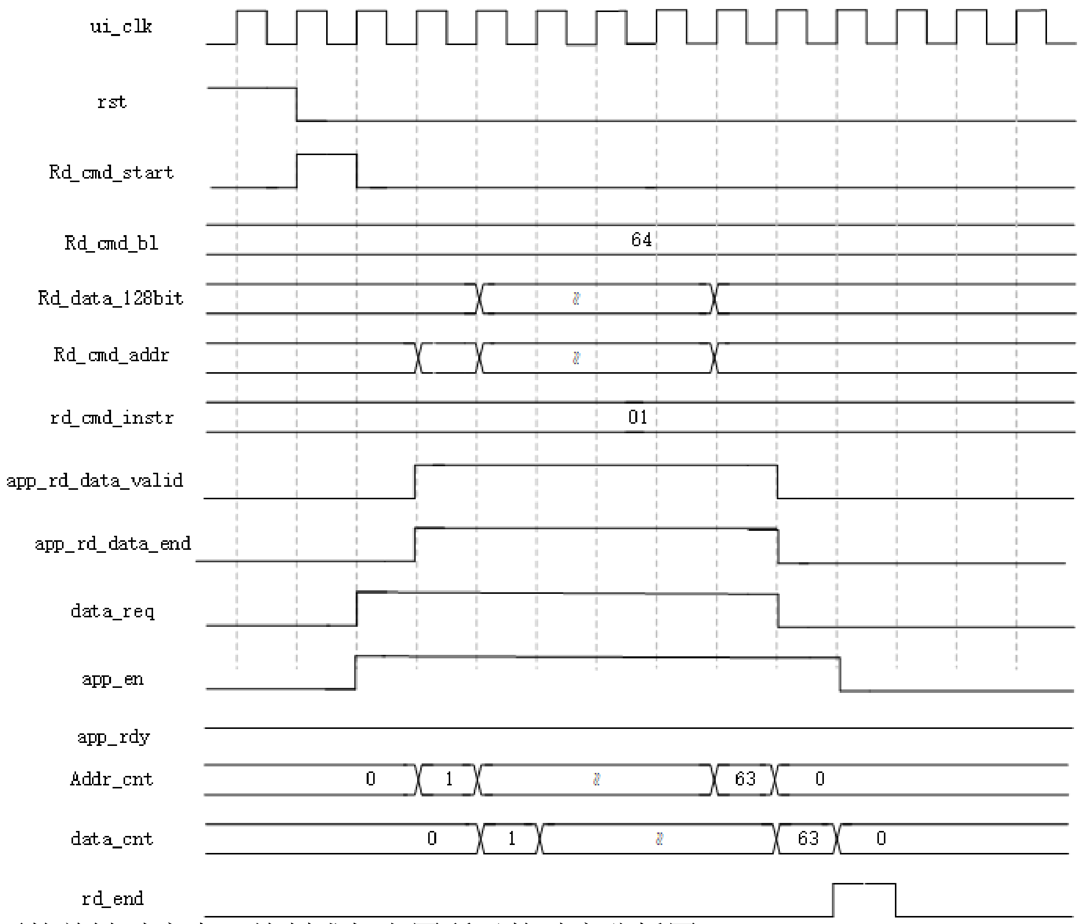
（1）在校验完成前，系统一直处于复位状态，即rst信号一直为高，校验完成后，rst信号才为低

（2）rd_cmd_start出现上升沿时，锁存rd_cmd_bl、rd_cmd_addr和rd_cmd_intr信号

（3）读数据时，要求先发地址，再发数据，地址需要在app_rpy和app_en同时有效时，输出有效数据；地址则需要在app_rd_data_valid有效时，读取正确的数据

（4）rd_end信号在读取完rd_cmd_bl突发长度的最后一个数据后，拉高一个时钟周期后，拉低信号

2.时序分析



按照上面的关键时序点，绘制成如上图所示的时序分析图。

四、练习步骤

1. 先编写verilog代码

根据关键点时序，编写代码。

2. 然后添加testbench文件，将写部分的时序屏蔽掉

（1）读这块的时序较为简单，只要加入rd_cmd_start信号即可

（2）其余部分只需赋值即可，而cmd_bl设置为64，cmd_instr设置为1，即为读命令，这点是很重要的

3. 最后编译，使用modelsim仿真

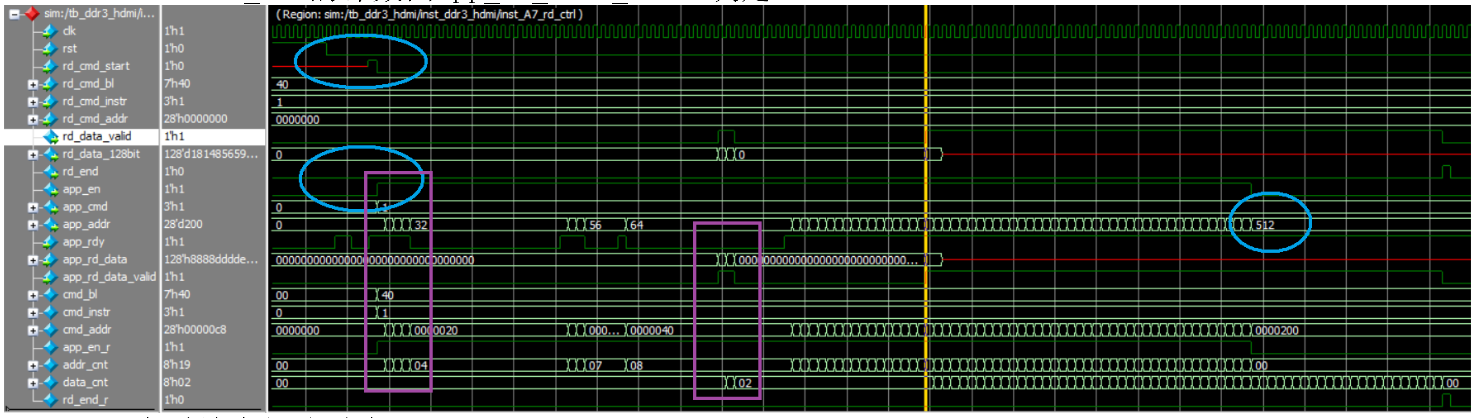
五、实际波形仿真

1. 查看仿真波形，把握关键点

（1）rd_cmd_start上升沿时，app_en置高

（2）addr_cnt的计数由app_en和app_rdy共同决定

（2）data_cnt的计数由app_rd_data_valid决定

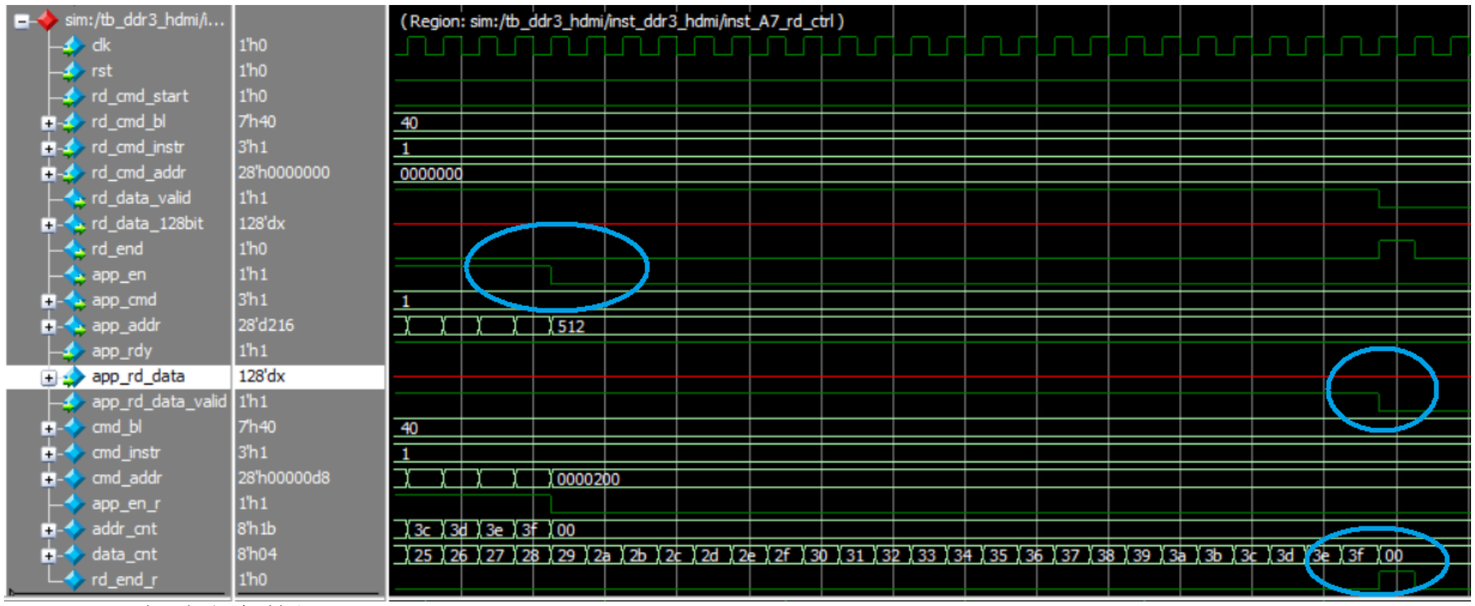

```
sim:/tb_ddr3_hdmi/i...
  clk                 1'h1
  rst                 1'h0
  rd_cmd_start        1'h0
  rd_cmd_bl           7'h40
  rd_cmd_instr        3'h1
  rd_cmd_addr         28'h0000000
  rd_data_valid       1'h1
  rd_data_128bit      128'd181485659...
  rd_end              1'h0
  app_en              1'h1
  app_cmd             3'h1
  app_addr            28'd200
  app_rdy             1'h1
  app_rd_data         128'h8888dddde...
  app_rd_data_valid   1'h1
  cmd_bl              7'h40
  cmd_instr           3'h1
  cmd_addr            28'h00000c8
  app_en_r            1'h1
  addr_cnt            8'h19
  data_cnt            8'h02
  rd_end_r            1'h0
```

2. 查看结束部分时序

（1）rd_end在数据读取完毕后，拉高

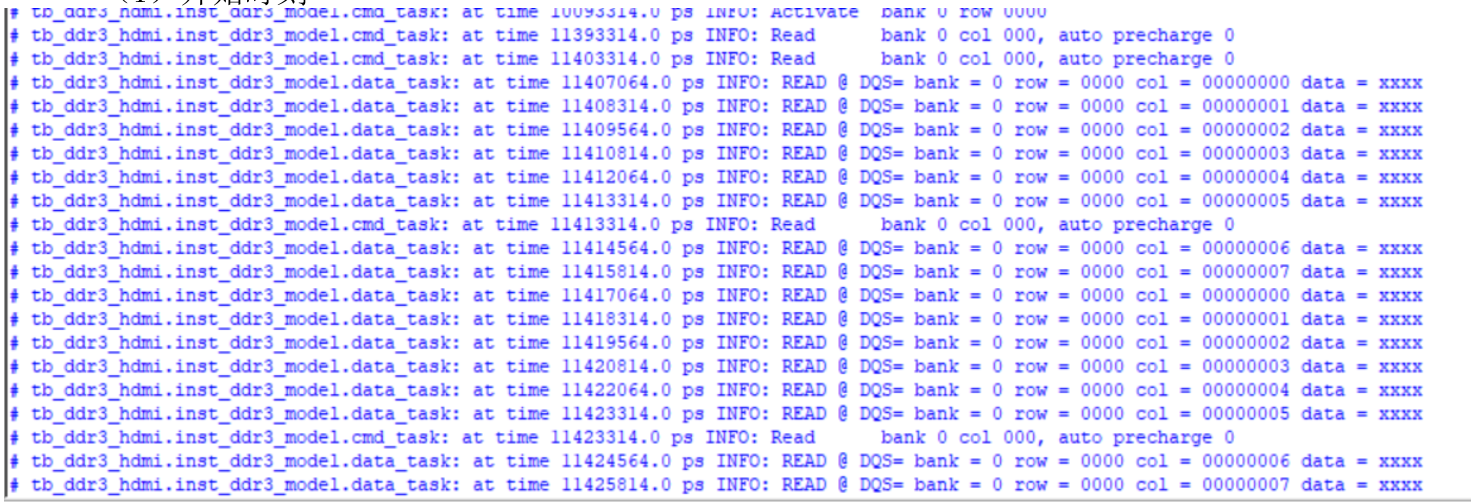
（2）rd_data_valid在读取完毕的时刻，拉低

（3）app_end也在地址发送完毕后，拉低



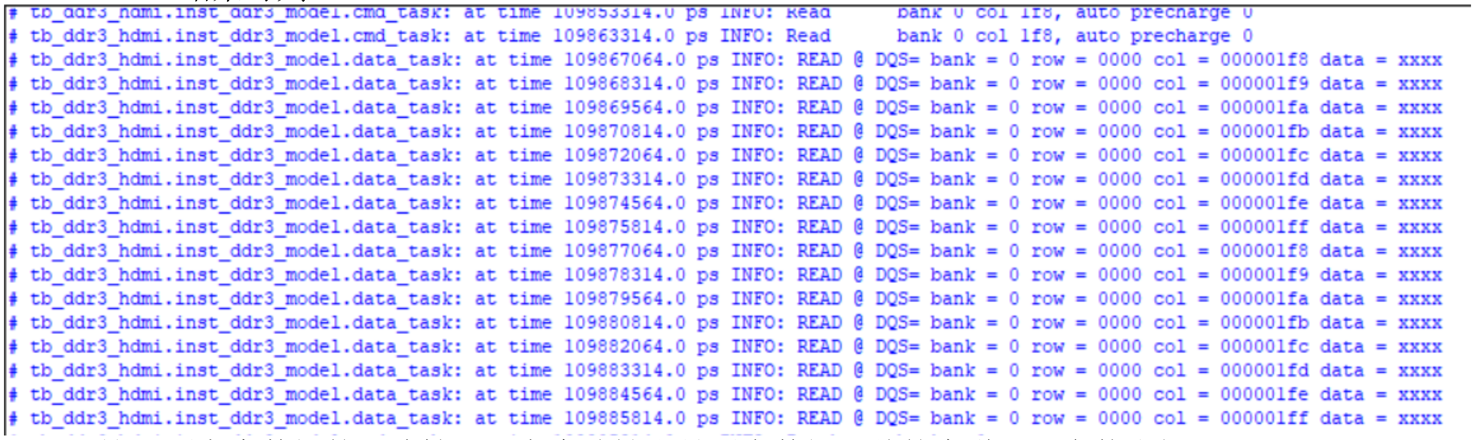
3. 查看仿真数据

（1）开始时刻



```
# tb_ddr3_hdmi.inst_ddr3_model.cmd_task: at time 11093314.0 ps INFO: Activate  bank 0 row 0000
# tb_ddr3_hdmi.inst_ddr3_model.cmd_task: at time 11393314.0 ps INFO: Read      bank 0 col 000, auto precharge 0
# tb_ddr3_hdmi.inst_ddr3_model.cmd_task: at time 11403314.0 ps INFO: Read      bank 0 col 000, auto precharge 0
# tb_ddr3_hdmi.inst_ddr3_model.data_task: at time 11407064.0 ps INFO: READ @ DQS= bank = 0 row = 0000 col = 00000000 data = xxxx
# tb_ddr3_hdmi.inst_ddr3_model.data_task: at time 11408314.0 ps INFO: READ @ DQS= bank = 0 row = 0000 col = 00000001 data = xxxx
# tb_ddr3_hdmi.inst_ddr3_model.data_task: at time 11409564.0 ps INFO: READ @ DQS= bank = 0 row = 0000 col = 00000002 data = xxxx
# tb_ddr3_hdmi.inst_ddr3_model.data_task: at time 11410814.0 ps INFO: READ @ DQS= bank = 0 row = 0000 col = 00000003 data = xxxx
# tb_ddr3_hdmi.inst_ddr3_model.data_task: at time 11412064.0 ps INFO: READ @ DQS= bank = 0 row = 0000 col = 00000004 data = xxxx
# tb_ddr3_hdmi.inst_ddr3_model.data_task: at time 11413314.0 ps INFO: READ @ DQS= bank = 0 row = 0000 col = 00000005 data = xxxx
# tb_ddr3_hdmi.inst_ddr3_model.cmd_task: at time 11413314.0 ps INFO: Read      bank 0 col 000, auto precharge 0
# tb_ddr3_hdmi.inst_ddr3_model.data_task: at time 11414564.0 ps INFO: READ @ DQS= bank = 0 row = 0000 col = 00000006 data = xxxx
# tb_ddr3_hdmi.inst_ddr3_model.data_task: at time 11415814.0 ps INFO: READ @ DQS= bank = 0 row = 0000 col = 00000007 data = xxxx
# tb_ddr3_hdmi.inst_ddr3_model.data_task: at time 11417064.0 ps INFO: READ @ DQS= bank = 0 row = 0000 col = 00000000 data = xxxx
# tb_ddr3_hdmi.inst_ddr3_model.data_task: at time 11418314.0 ps INFO: READ @ DQS= bank = 0 row = 0000 col = 00000001 data = xxxx
# tb_ddr3_hdmi.inst_ddr3_model.data_task: at time 11419564.0 ps INFO: READ @ DQS= bank = 0 row = 0000 col = 00000002 data = xxxx
# tb_ddr3_hdmi.inst_ddr3_model.data_task: at time 11420814.0 ps INFO: READ @ DQS= bank = 0 row = 0000 col = 00000003 data = xxxx
# tb_ddr3_hdmi.inst_ddr3_model.data_task: at time 11422064.0 ps INFO: READ @ DQS= bank = 0 row = 0000 col = 00000004 data = xxxx
# tb_ddr3_hdmi.inst_ddr3_model.data_task: at time 11423314.0 ps INFO: READ @ DQS= bank = 0 row = 0000 col = 00000005 data = xxxx
# tb_ddr3_hdmi.inst_ddr3_model.cmd_task: at time 11423314.0 ps INFO: Read      bank 0 col 000, auto precharge 0
# tb_ddr3_hdmi.inst_ddr3_model.data_task: at time 11424564.0 ps INFO: READ @ DQS= bank = 0 row = 0000 col = 00000006 data = xxxx
# tb_ddr3_hdmi.inst_ddr3_model.data_task: at time 11425814.0 ps INFO: READ @ DQS= bank = 0 row = 0000 col = 00000007 data = xxxx
```

（2）结束时刻



```
# tb_ddr3_hdmi.inst_ddr3_model.cmd_task: at time 109853314.0 ps INFO: Read      bank 0 col 1f8, auto precharge 0
# tb_ddr3_hdmi.inst_ddr3_model.cmd_task: at time 109863314.0 ps INFO: Read      bank 0 col 1f8, auto precharge 0
# tb_ddr3_hdmi.inst_ddr3_model.data_task: at time 109867064.0 ps INFO: READ @ DQS= bank = 0 row = 0000 col = 000001f8 data = xxxx
# tb_ddr3_hdmi.inst_ddr3_model.data_task: at time 109868314.0 ps INFO: READ @ DQS= bank = 0 row = 0000 col = 000001f9 data = xxxx
# tb_ddr3_hdmi.inst_ddr3_model.data_task: at time 109869564.0 ps INFO: READ @ DQS= bank = 0 row = 0000 col = 000001fa data = xxxx
# tb_ddr3_hdmi.inst_ddr3_model.data_task: at time 109870814.0 ps INFO: READ @ DQS= bank = 0 row = 0000 col = 000001fb data = xxxx
# tb_ddr3_hdmi.inst_ddr3_model.data_task: at time 109872064.0 ps INFO: READ @ DQS= bank = 0 row = 0000 col = 000001fc data = xxxx
# tb_ddr3_hdmi.inst_ddr3_model.data_task: at time 109873314.0 ps INFO: READ @ DQS= bank = 0 row = 0000 col = 000001fd data = xxxx
# tb_ddr3_hdmi.inst_ddr3_model.data_task: at time 109874564.0 ps INFO: READ @ DQS= bank = 0 row = 0000 col = 000001fe data = xxxx
# tb_ddr3_hdmi.inst_ddr3_model.data_task: at time 109875814.0 ps INFO: READ @ DQS= bank = 0 row = 0000 col = 000001ff data = xxxx
# tb_ddr3_hdmi.inst_ddr3_model.data_task: at time 109877064.0 ps INFO: READ @ DQS= bank = 0 row = 0000 col = 000001f8 data = xxxx
# tb_ddr3_hdmi.inst_ddr3_model.data_task: at time 109878314.0 ps INFO: READ @ DQS= bank = 0 row = 0000 col = 000001f9 data = xxxx
# tb_ddr3_hdmi.inst_ddr3_model.data_task: at time 109879564.0 ps INFO: READ @ DQS= bank = 0 row = 0000 col = 000001fa data = xxxx
# tb_ddr3_hdmi.inst_ddr3_model.data_task: at time 109880814.0 ps INFO: READ @ DQS= bank = 0 row = 0000 col = 000001fb data = xxxx
# tb_ddr3_hdmi.inst_ddr3_model.data_task: at time 109882064.0 ps INFO: READ @ DQS= bank = 0 row = 0000 col = 000001fc data = xxxx
# tb_ddr3_hdmi.inst_ddr3_model.data_task: at time 109883314.0 ps INFO: READ @ DQS= bank = 0 row = 0000 col = 000001fd data = xxxx
# tb_ddr3_hdmi.inst_ddr3_model.data_task: at time 109884564.0 ps INFO: READ @ DQS= bank = 0 row = 0000 col = 000001fe data = xxxx
# tb_ddr3_hdmi.inst_ddr3_model.data_task: at time 109885814.0 ps INFO: READ @ DQS= bank = 0 row = 0000 col = 000001ff data = xxxx
```

这里不考虑数据的正确性，只考虑地址和是否有数据，总的来说，没有什么问题。

六、总结与讨论

1. 写都搞定了，读还会远吗

（1）学会举一反三

（2）只是改改而已，而且更简单

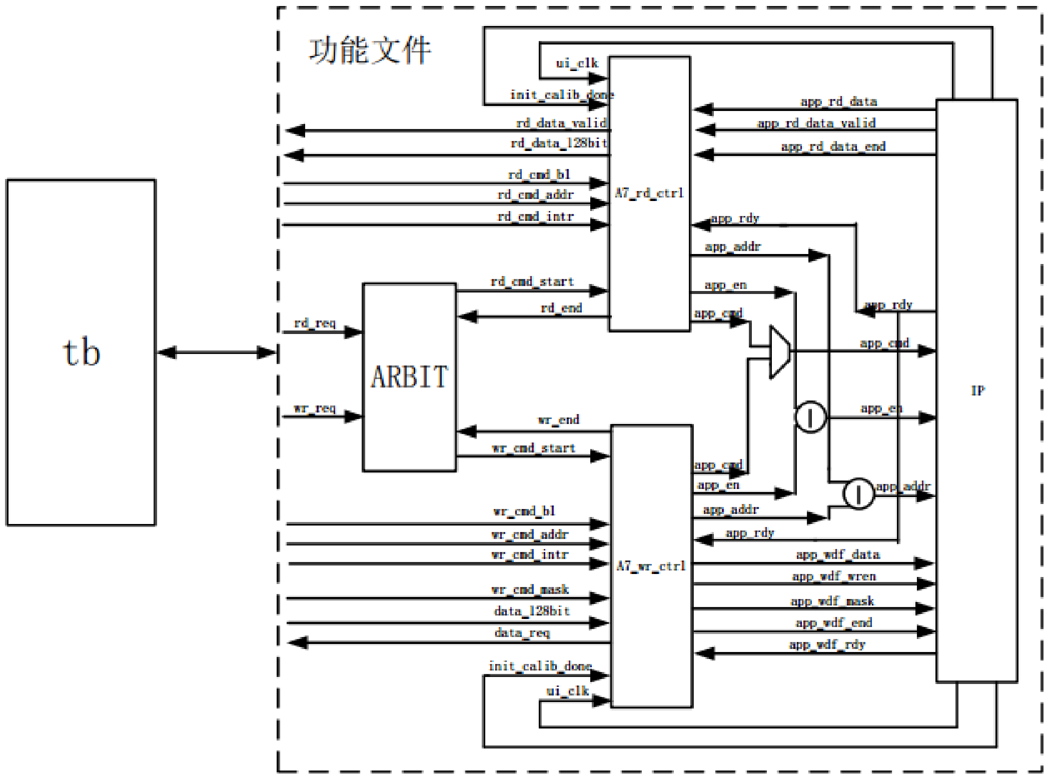
（3）没什么说的，干就是了

第二部分　DDR3 SDRAM IP　仲裁实现

一、练习内容

ddr3 sdram仲裁实现

二、系统框图



三、设计分析

1.地址、命令共用总线

主要涉及app_addr，app_en和app_cmd三根线

（1）app_addr在读写完毕后，地址清零，这样读写就不会互相干扰了

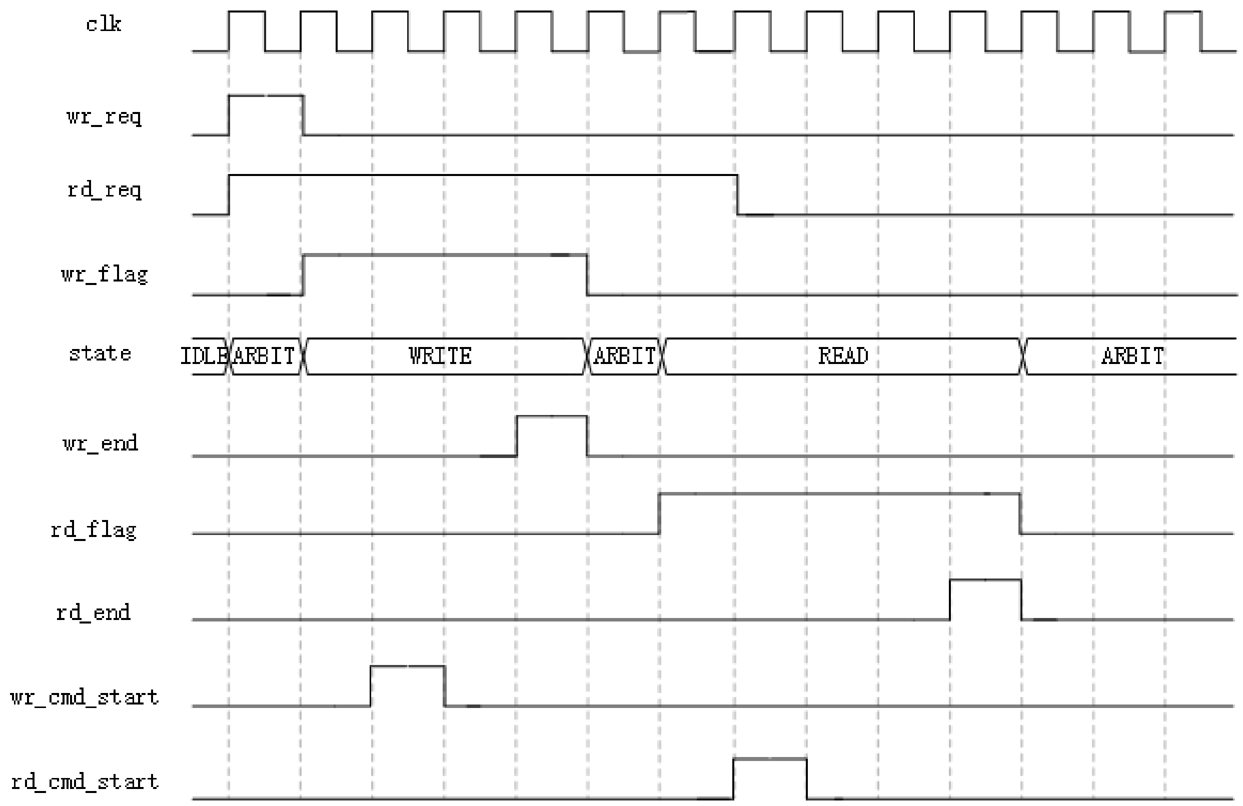
（2）app_en在读写部分相或即可

（3）app_cmd在读写部分不同步即可，即写的时候为0，其余时刻为1

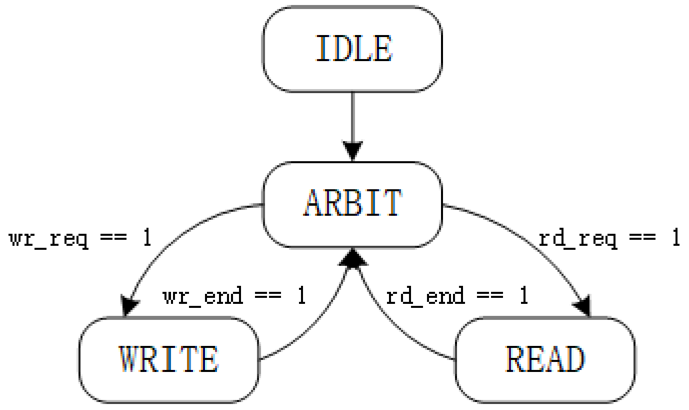
2.仲裁器实现

（1）关键时序点分析

- 写请求的优先级高于读请求，写请求有效进入写状态，读请求有效进入读状态
- 写结束标志和读结束标志都进入总裁状态，重新进行信号的判定



（2）状态机实现



四、练习步骤

1.按照时序图和状态转移图编写代码，然后编译仿真，查看结果。

（1）注意关键点时序，编写代码

```
`timescale 1ns / 1ps

module A7_arbit(
        input         wire         clk,
        input         wire         rst,
        input        wire        wr_req,
        input         wire         rd_req,
        input         wire         wr_end,
        input         wire         rd_end,
        output         wire         wr_cmd_start,
        output         wire         rd_cmd_start
        );

localparam        IDLE        = 4'b0001,
                        ARBIT        = 4'b0010,
                        WRITE        = 4'b0100,
                        READ        = 4'b1000;
```

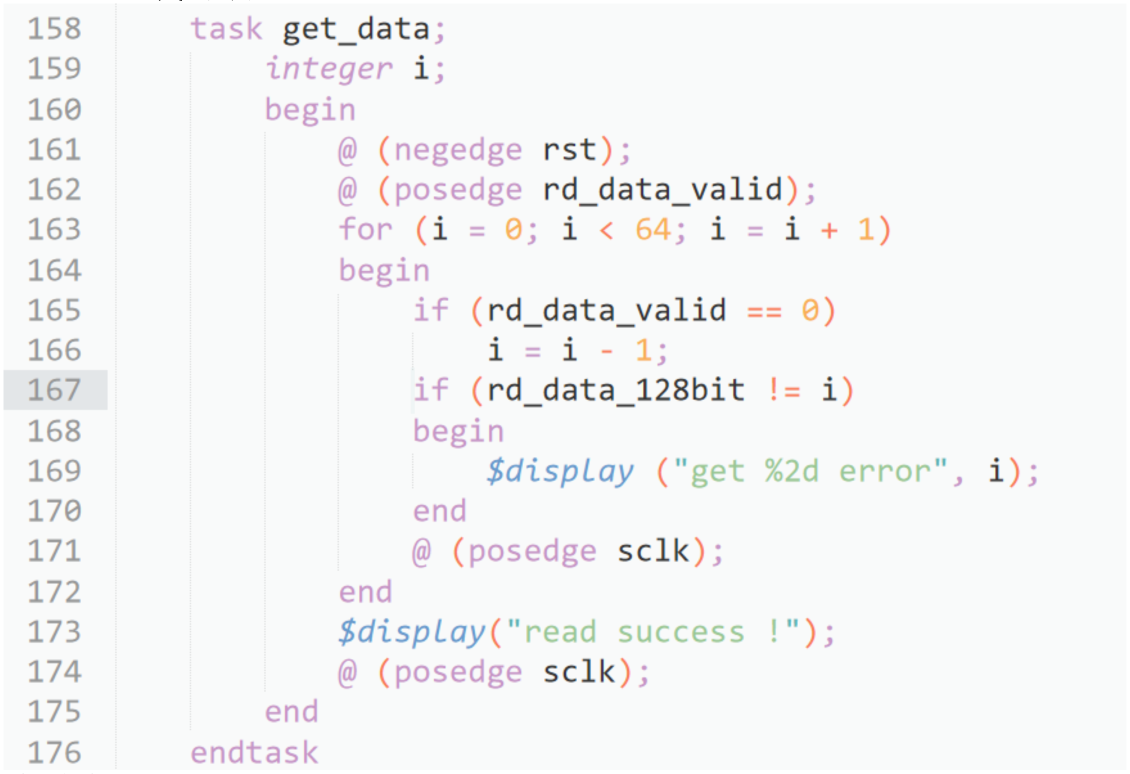
```
reg         [3:0]       state;
reg                               wr_flag;
reg                         rd_flag;
reg                         wr_start;
reg                         rd_start;

assign wr_cmd_start = wr_start;
assign rd_cmd_start = rd_start;

//state
always @ (posedge clk)
begin
        if (rst == 1'b1)
                state <= IDLE;
        else
        begin
                case (state)
                        IDLE:
                                state <= ARBIT;
                        ARBIT:
                                if (wr_req == 1'b1)
                                        state <= WRITE;
                                else if (rd_req == 1'b1)
                                        state <= READ;
                        WRITE:
                                if (wr_end == 1'b1)
                                        state <= ARBIT;
                        READ:
                                if (rd_end == 1'b1)
                                        state <= ARBIT;
                        default:
                                state <= IDLE;
                endcase
        end
end

//wr_flag
always @ (posedge clk)
begin
        if (rst == 1'b1)
                wr_flag <= 1'b0;
        else if (state == ARBIT && wr_req == 1'b1)
                wr_flag <= 1'b1;
        else if (state == WRITE && wr_end == 1'b1)
                wr_flag <= 1'b0;
end

//rd_flag
always @ (posedge clk)
begin
        if (rst == 1'b1)
                rd_flag <= 1'b0;
        else if (state == ARBIT && rd_req == 1'b1 &&    wr_req == 1'b0)
                rd_flag <= 1'b1;
        else if (state == READ && rd_end == 1'b1)
```

```
                rd_flag <= 1'b0;
end

//wr_cmd_start
always @ (posedge clk)
begin
        if (rst == 1'b1)
                wr_start <= 1'b0;
        else if (state == ARBIT && wr_req == 1'b1)
                wr_start <= 1'b1;
        else
                wr_start <= 1'b0;
end

//rd_cmd_start
always @ (posedge clk)
begin
        if (rst == 1'b1)
                rd_start <= 1'b0;
        else if (state == ARBIT && rd_req == 1'b1 &&    wr_req == 1'b0)
                rd_start <= 1'b1;
        else
                rd_start <= 1'b0;
end

endmodule
```

（2）编译后使用modelsim仿真，查看结果。

2. 然后将数据比较部分添加至testbench中，再次查看是否获得了正确的结果。

（1）testbench代码添加



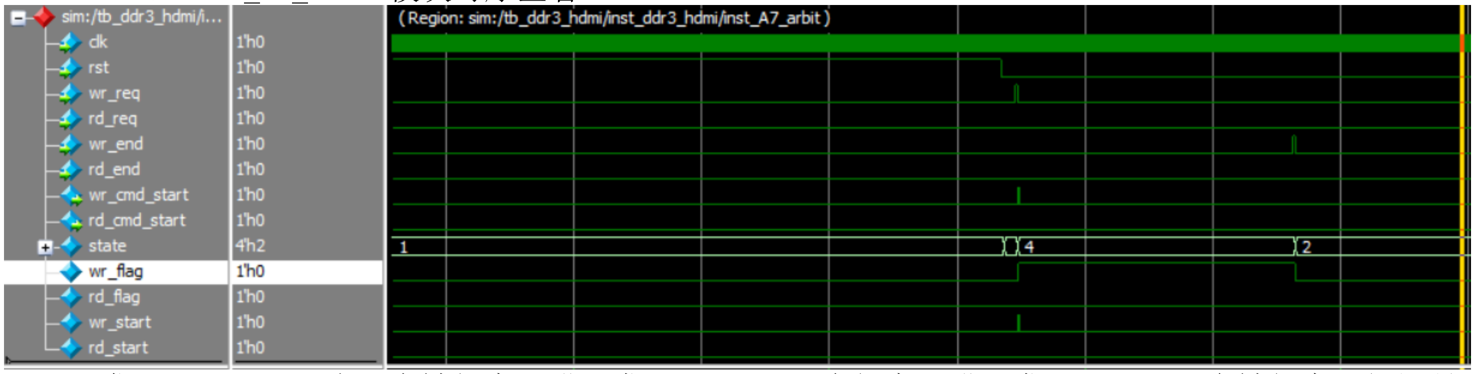
```
task get_data;
    integer i;
    begin
        @ (negedge rst);
        @ (posedge rd_data_valid);
        for (i = 0; i < 64; i = i + 1)
        begin
            if (rd_data_valid == 0)
                i = i - 1;
            if (rd_data_128bit != i)
            begin
                $display ("get %2d error", i);
            end
            @ (posedge sclk);
        end
        $display("read success !");
        @ (posedge sclk);
    end
endtask
```

（2）查看结果

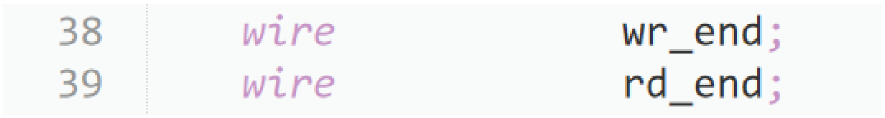
五、实际波形仿真

1. 查看仿真波形

（1）inst_A7_arbit模块时序查看

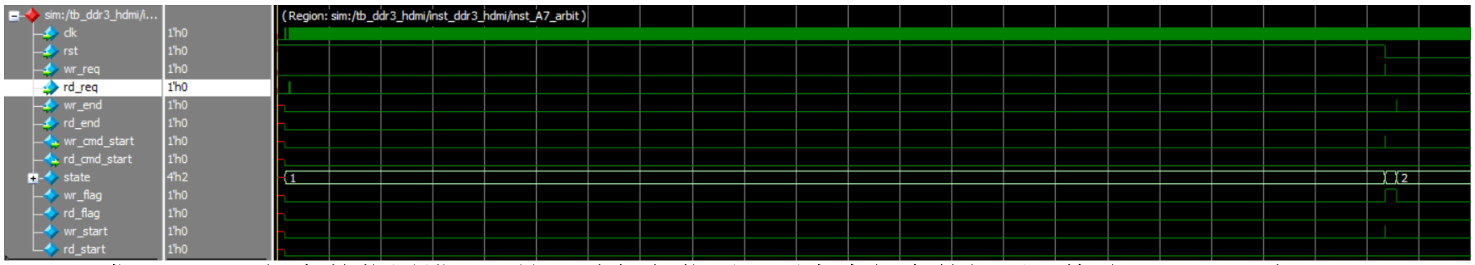


发现rd_start一直没有被拉高，进而发现rd_flag没有拉高，进而发现rd_req没有被拉高，怀疑是testbench出错。

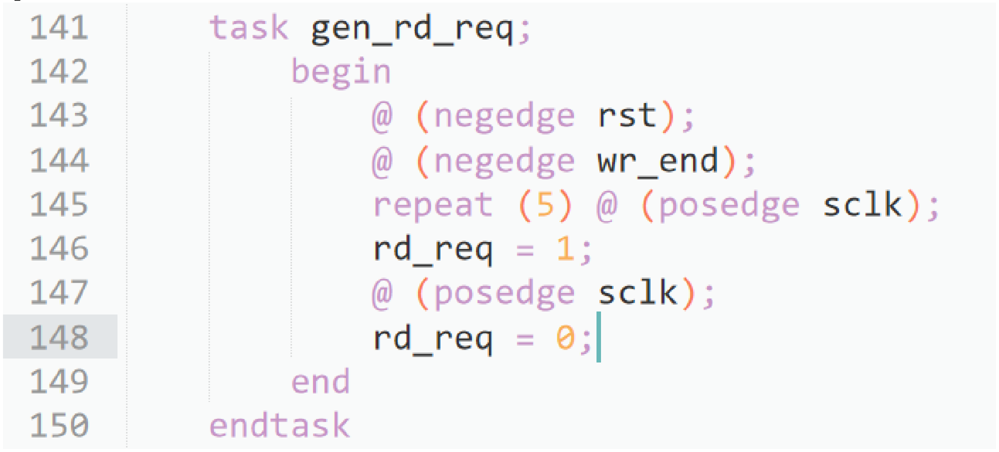


```
wire            wr_end;
wire            rd_end;
```

发现testbench中wr_end的变量类型是wire,改成reg重新编译，查看结果。

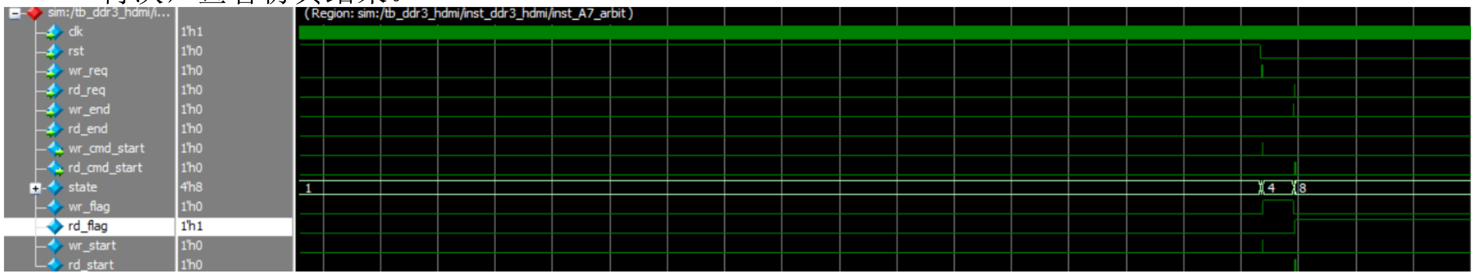


发现rd_req拉高的位置错误，这里选择复位后，再考虑拉高的问题，修改testbench如下。



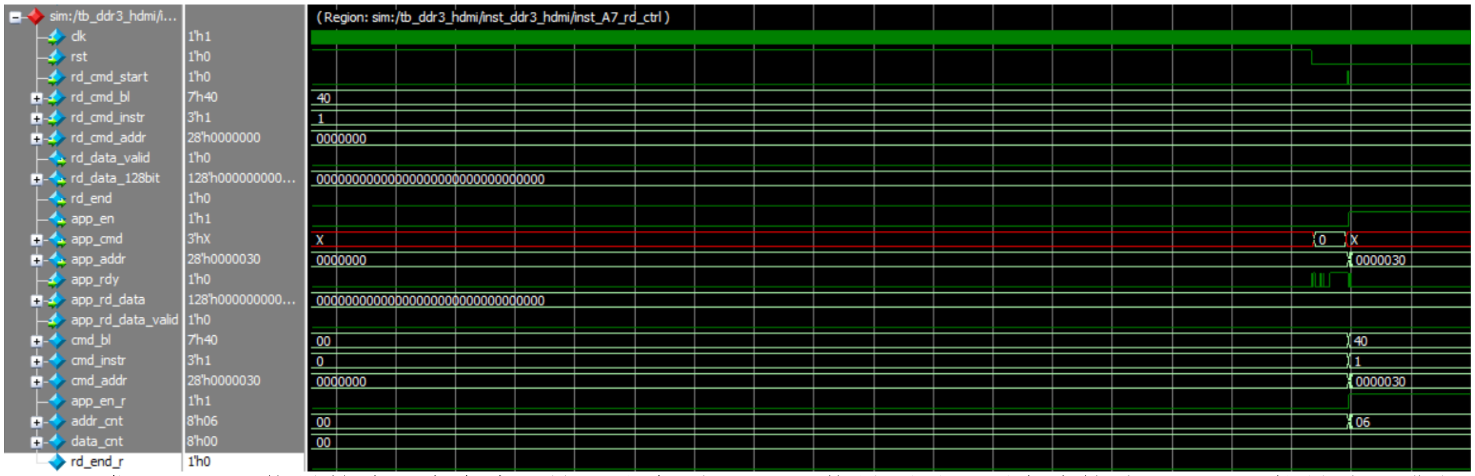
```
task gen_rd_req;
    begin
        @ (negedge rst);
        @ (negedge wr_end);
        repeat (5) @ (posedge sclk);
        rd_req = 1;
        @ (posedge sclk);
        rd_req = 0;
    end
endtask
```

再次，查看仿真结果。



解决了rd_req不拉高的问题，这又出现了一个新问题，那就是rd_end不拉高。

（2）inst_A7_rd_ctrl模块时序查看



发现rd_end信号的确没有产生，这里注意到app_cmd信号不对，不应该的1'bx，而只能是1'b1或者是1'b0，这里是存在问题的。

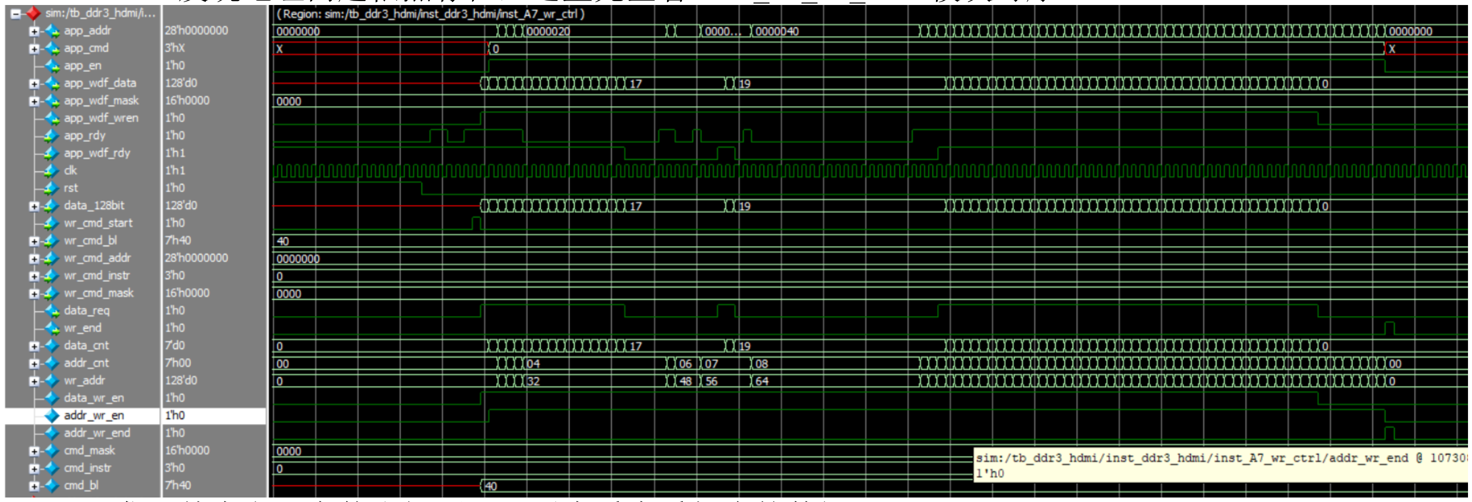
而rd_ctrl模块信号只有rd_cmd_addr修改过，这里没有问题。

同时，我注意到rd_cmd_addr也是不正确的，读写完毕后相应的地址应该清零，而不应该是0x30，这是不对的。

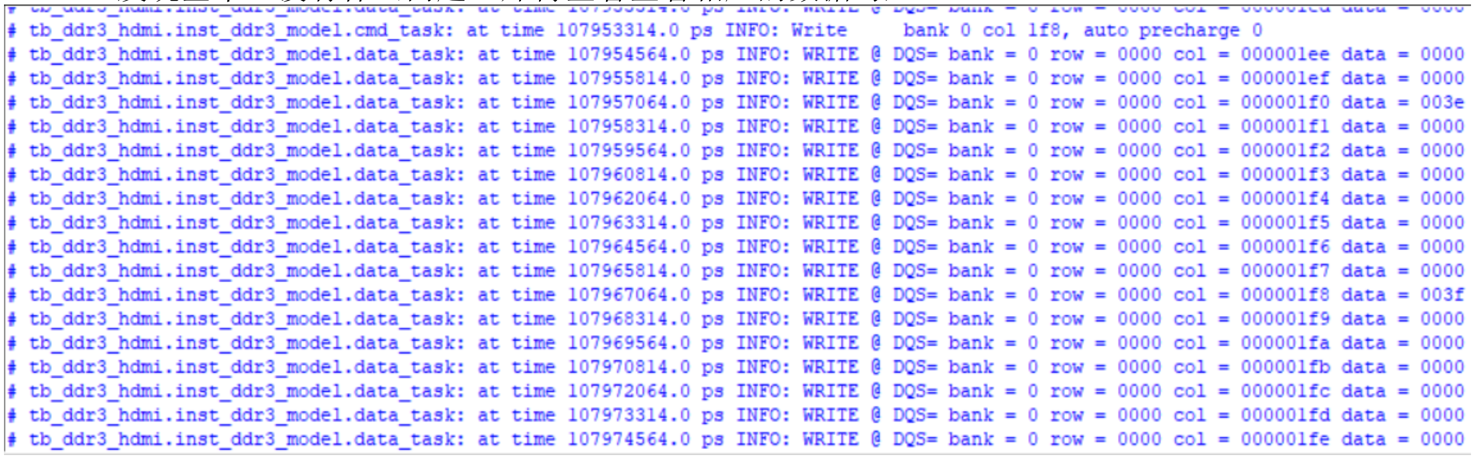
对于addr的操作在wr_ctrl模块也进行了修改，这里查看相关的verilog代码。

发现了问题，这里重新编译，然后在查看结果。

（3）发现地址问题依然存在，这里先查看inst_A7_wr_ctrl模块时序

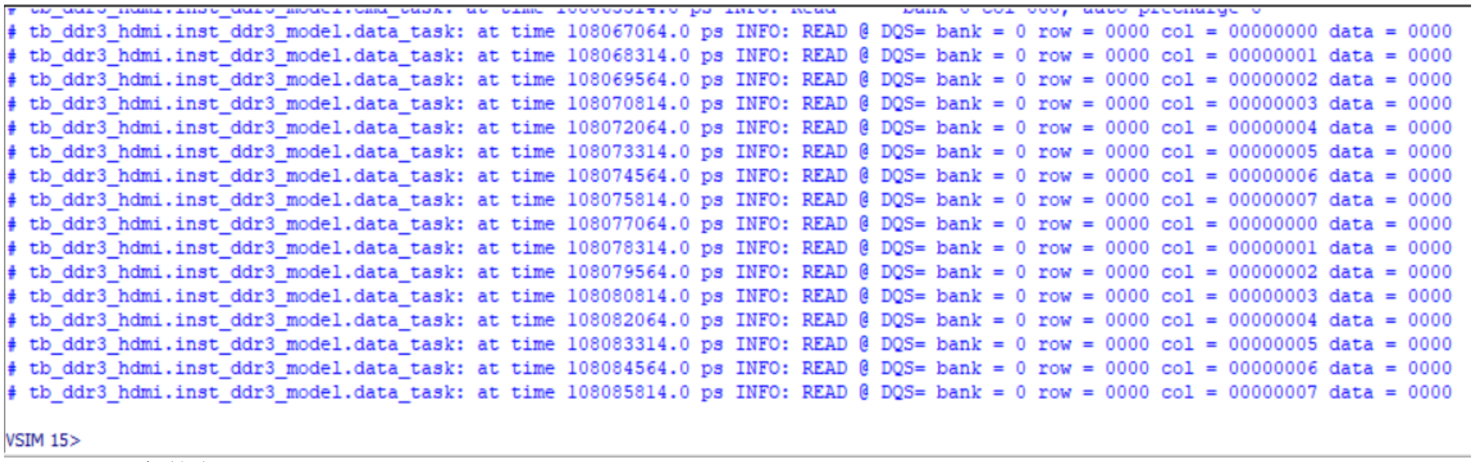


发现基本上没有什么问题，那再查看查看相应的数据写入。



```
# tb_ddr3_hdmi.inst_ddr3_model.cmd_task: at time 107953314.0 ps INFO: Write     bank 0 col 1f8, auto precharge 0
# tb_ddr3_hdmi.inst_ddr3_model.data_task: at time 107954564.0 ps INFO: WRITE @ DQS= bank = 0 row = 0000 col = 000001ee data = 0000
# tb_ddr3_hdmi.inst_ddr3_model.data_task: at time 107955814.0 ps INFO: WRITE @ DQS= bank = 0 row = 0000 col = 000001ef data = 0000
# tb_ddr3_hdmi.inst_ddr3_model.data_task: at time 107957064.0 ps INFO: WRITE @ DQS= bank = 0 row = 0000 col = 000001f0 data = 003e
# tb_ddr3_hdmi.inst_ddr3_model.data_task: at time 107958314.0 ps INFO: WRITE @ DQS= bank = 0 row = 0000 col = 000001f1 data = 0000
# tb_ddr3_hdmi.inst_ddr3_model.data_task: at time 107959564.0 ps INFO: WRITE @ DQS= bank = 0 row = 0000 col = 000001f2 data = 0000
# tb_ddr3_hdmi.inst_ddr3_model.data_task: at time 107960814.0 ps INFO: WRITE @ DQS= bank = 0 row = 0000 col = 000001f3 data = 0000
# tb_ddr3_hdmi.inst_ddr3_model.data_task: at time 107962064.0 ps INFO: WRITE @ DQS= bank = 0 row = 0000 col = 000001f4 data = 0000
# tb_ddr3_hdmi.inst_ddr3_model.data_task: at time 107963314.0 ps INFO: WRITE @ DQS= bank = 0 row = 0000 col = 000001f5 data = 0000
# tb_ddr3_hdmi.inst_ddr3_model.data_task: at time 107964564.0 ps INFO: WRITE @ DQS= bank = 0 row = 0000 col = 000001f6 data = 0000
# tb_ddr3_hdmi.inst_ddr3_model.data_task: at time 107965814.0 ps INFO: WRITE @ DQS= bank = 0 row = 0000 col = 000001f7 data = 0000
# tb_ddr3_hdmi.inst_ddr3_model.data_task: at time 107967064.0 ps INFO: WRITE @ DQS= bank = 0 row = 0000 col = 000001f8 data = 003f
# tb_ddr3_hdmi.inst_ddr3_model.data_task: at time 107968314.0 ps INFO: WRITE @ DQS= bank = 0 row = 0000 col = 000001f9 data = 0000
# tb_ddr3_hdmi.inst_ddr3_model.data_task: at time 107969564.0 ps INFO: WRITE @ DQS= bank = 0 row = 0000 col = 000001fa data = 0000
# tb_ddr3_hdmi.inst_ddr3_model.data_task: at time 107970814.0 ps INFO: WRITE @ DQS= bank = 0 row = 0000 col = 000001fb data = 0000
# tb_ddr3_hdmi.inst_ddr3_model.data_task: at time 107972064.0 ps INFO: WRITE @ DQS= bank = 0 row = 0000 col = 000001fc data = 0000
# tb_ddr3_hdmi.inst_ddr3_model.data_task: at time 107973314.0 ps INFO: WRITE @ DQS= bank = 0 row = 0000 col = 000001fd data = 0000
# tb_ddr3_hdmi.inst_ddr3_model.data_task: at time 107974564.0 ps INFO: WRITE @ DQS= bank = 0 row = 0000 col = 000001fe data = 0000
```

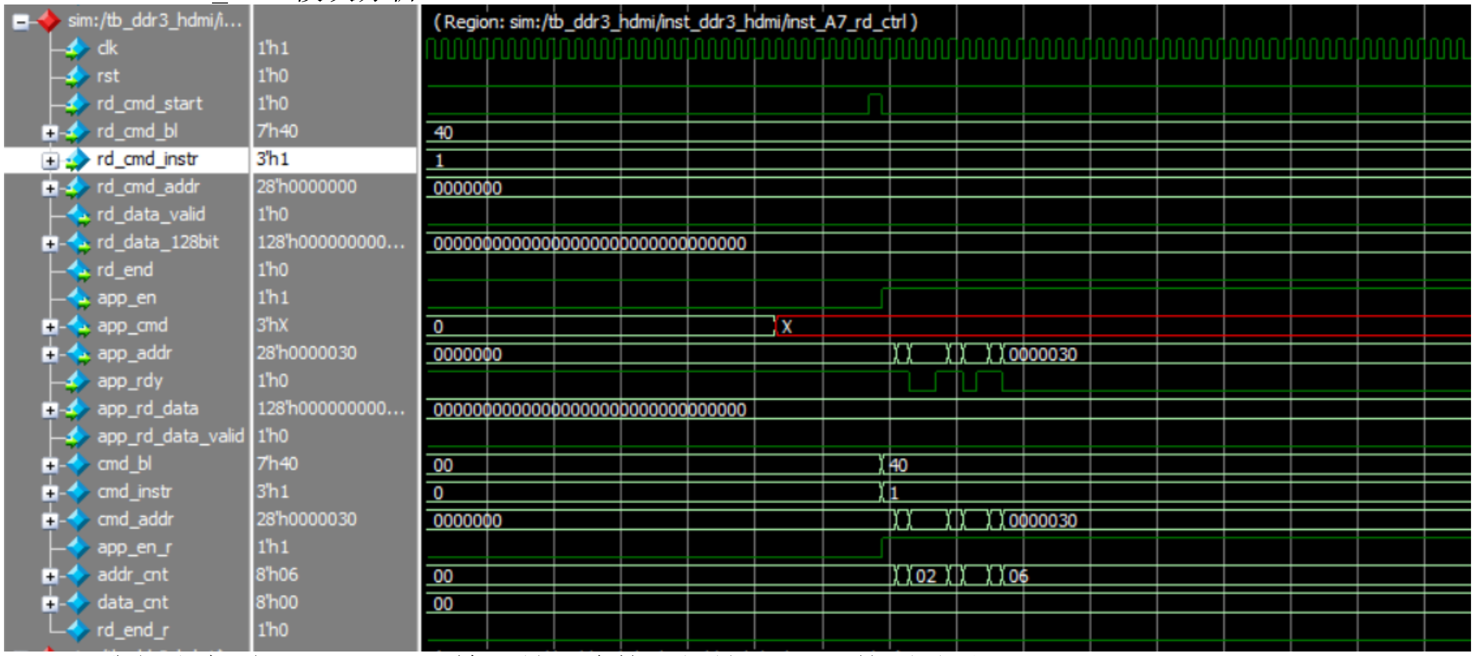
发现数据的写入没有什么问题。读取数据方面除了问题。



```
# tb_ddr3_hdmi.inst_ddr3_model.cmd_task: at time 108065314.0 ps INFO: Read    bank 0 col 000, auto precharge 0
# tb_ddr3_hdmi.inst_ddr3_model.data_task: at time 108067064.0 ps INFO: READ @ DQS= bank = 0 row = 0000 col = 00000000 data = 0000
# tb_ddr3_hdmi.inst_ddr3_model.data_task: at time 108068314.0 ps INFO: READ @ DQS= bank = 0 row = 0000 col = 00000001 data = 0000
# tb_ddr3_hdmi.inst_ddr3_model.data_task: at time 108069564.0 ps INFO: READ @ DQS= bank = 0 row = 0000 col = 00000002 data = 0000
# tb_ddr3_hdmi.inst_ddr3_model.data_task: at time 108070814.0 ps INFO: READ @ DQS= bank = 0 row = 0000 col = 00000003 data = 0000
# tb_ddr3_hdmi.inst_ddr3_model.data_task: at time 108072064.0 ps INFO: READ @ DQS= bank = 0 row = 0000 col = 00000004 data = 0000
# tb_ddr3_hdmi.inst_ddr3_model.data_task: at time 108073314.0 ps INFO: READ @ DQS= bank = 0 row = 0000 col = 00000005 data = 0000
# tb_ddr3_hdmi.inst_ddr3_model.data_task: at time 108074564.0 ps INFO: READ @ DQS= bank = 0 row = 0000 col = 00000006 data = 0000
# tb_ddr3_hdmi.inst_ddr3_model.data_task: at time 108075814.0 ps INFO: READ @ DQS= bank = 0 row = 0000 col = 00000007 data = 0000
# tb_ddr3_hdmi.inst_ddr3_model.data_task: at time 108077064.0 ps INFO: READ @ DQS= bank = 0 row = 0000 col = 00000000 data = 0000
# tb_ddr3_hdmi.inst_ddr3_model.data_task: at time 108078314.0 ps INFO: READ @ DQS= bank = 0 row = 0000 col = 00000001 data = 0000
# tb_ddr3_hdmi.inst_ddr3_model.data_task: at time 108079564.0 ps INFO: READ @ DQS= bank = 0 row = 0000 col = 00000002 data = 0000
# tb_ddr3_hdmi.inst_ddr3_model.data_task: at time 108080814.0 ps INFO: READ @ DQS= bank = 0 row = 0000 col = 00000003 data = 0000
# tb_ddr3_hdmi.inst_ddr3_model.data_task: at time 108082064.0 ps INFO: READ @ DQS= bank = 0 row = 0000 col = 00000004 data = 0000
# tb_ddr3_hdmi.inst_ddr3_model.data_task: at time 108083314.0 ps INFO: READ @ DQS= bank = 0 row = 0000 col = 00000005 data = 0000
# tb_ddr3_hdmi.inst_ddr3_model.data_task: at time 108084564.0 ps INFO: READ @ DQS= bank = 0 row = 0000 col = 00000006 data = 0000
# tb_ddr3_hdmi.inst_ddr3_model.data_task: at time 108085814.0 ps INFO: READ @ DQS= bank = 0 row = 0000 col = 00000007 data = 0000

VSIM 15>
```

2. 查找问题

（1）rd_ctrl模块分析



```
sim:/tb_ddr3_hdmi/i...
clk              1'h1
rst              1'h0
rd_cmd_start     1'h0
rd_cmd_bl        7'h40
rd_cmd_instr     3'h1
rd_cmd_addr      28'h0000000
rd_data_valid    1'h0
rd_data_128bit   128'h000000000...
rd_end           1'h0
app_en           1'h1
app_cmd          3'hX
app_addr         28'h0000030
app_rdy          1'h0
app_rd_data      128'h000000000...
app_rd_data_valid 1'h0
cmd_bl           7'h40
cmd_instr        3'h1
cmd_addr         28'h0000030
app_en_r         1'h1
addr_cnt         8'h06
data_cnt         8'h00
rd_end_r         1'h0
```

我们注意到rd_cmd_instr输入是正确的，但是app_cmd就不对了。

我们再查看对应的verilog代码，发现app_cmd的实现是这样的。

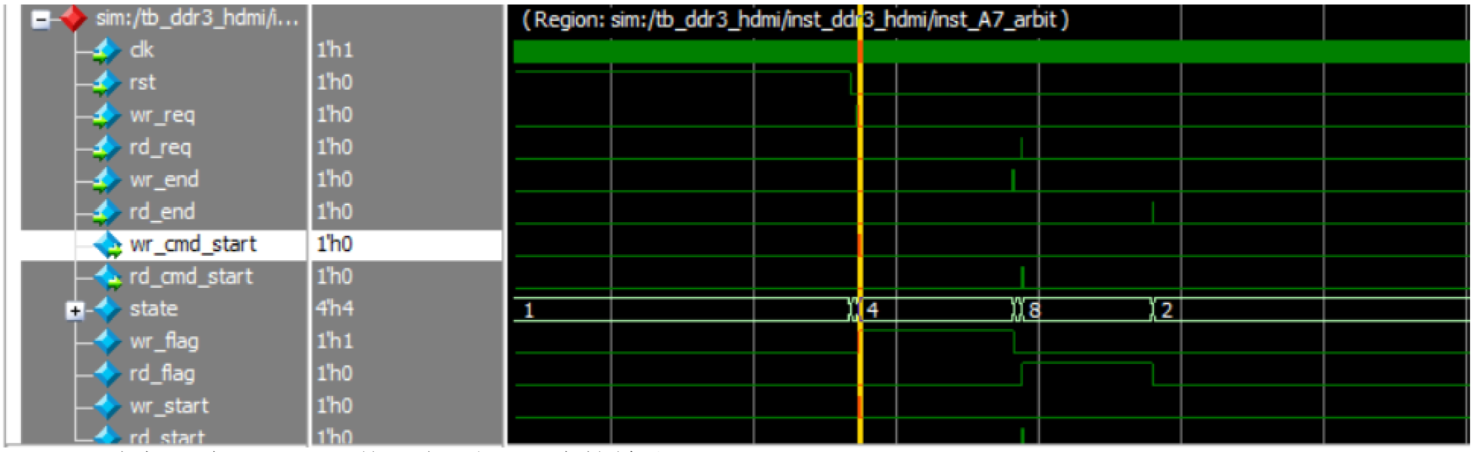
```
assign app_cmd = (wr_app_en == 1'b1) ? 1'b0 : 1'b1;
```

实际上在wr_ctrl和rd_ctrl模块的内部也存在对app_cmd的赋值，这里稍微修改下，只从ddr3_hdmi赋值，不从testbench赋值，

```
assign app_cmd = (wr_app_en == 1'b1) ? wr_app_cmd : rd_app_cmd;
```

（2）修改代码后，重新编译进行modelsim仿真。

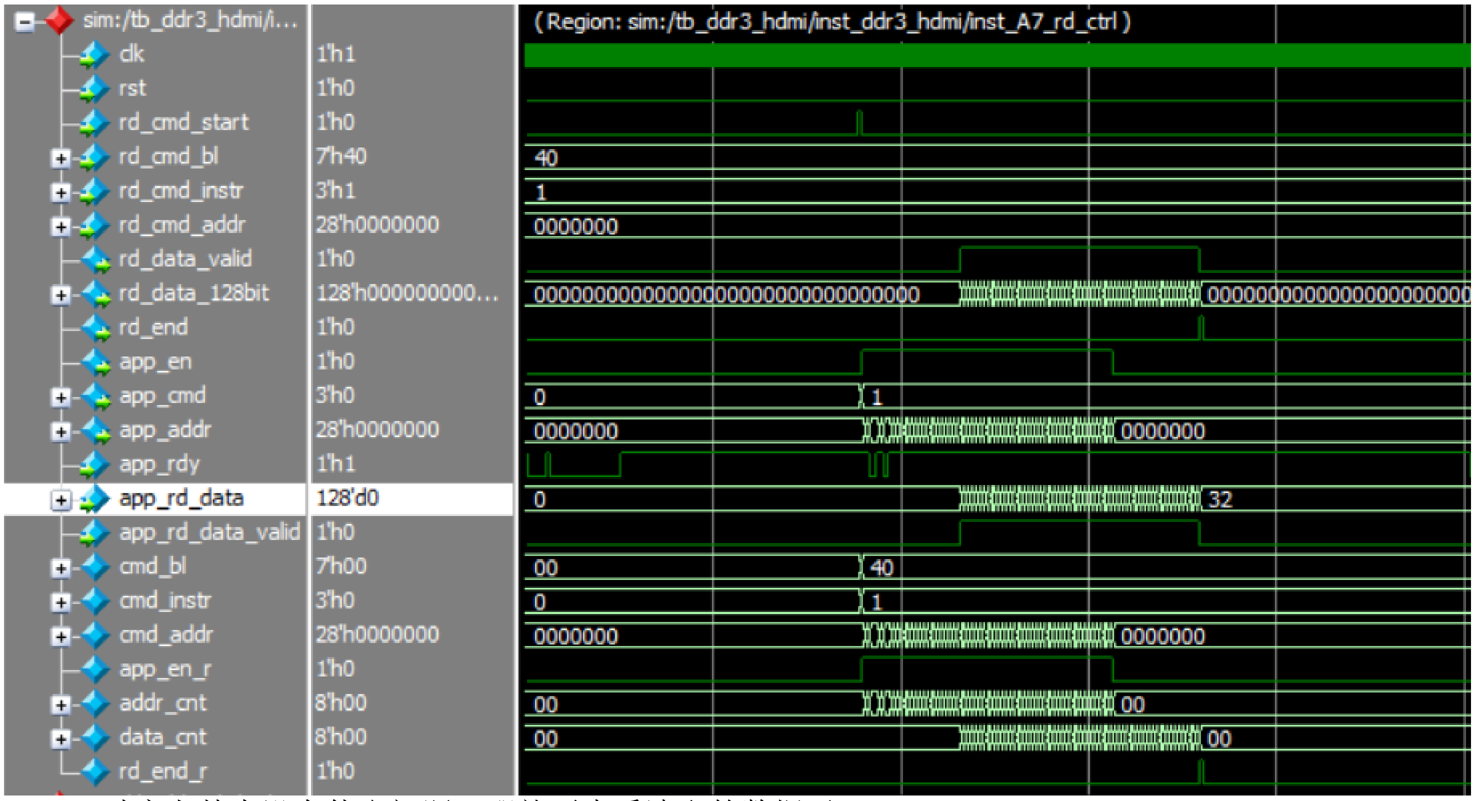
- 先观察A7_aribt模块时序



```
clk           1'h1
rst           1'h0
wr_req        1'h0
rd_req        1'h0
wr_end        1'h0
rd_end        1'h0
wr_cmd_start  1'h0
rd_cmd_start  1'h0
state         4'h4     1    4    8    2
wr_flag       1'h1
rd_flag       1'h0
wr_start      1'h0
rd_start      1'h0
```

时序正确，rd_end终于得到了正确的输出。

- 再观察rd_ctrl模块

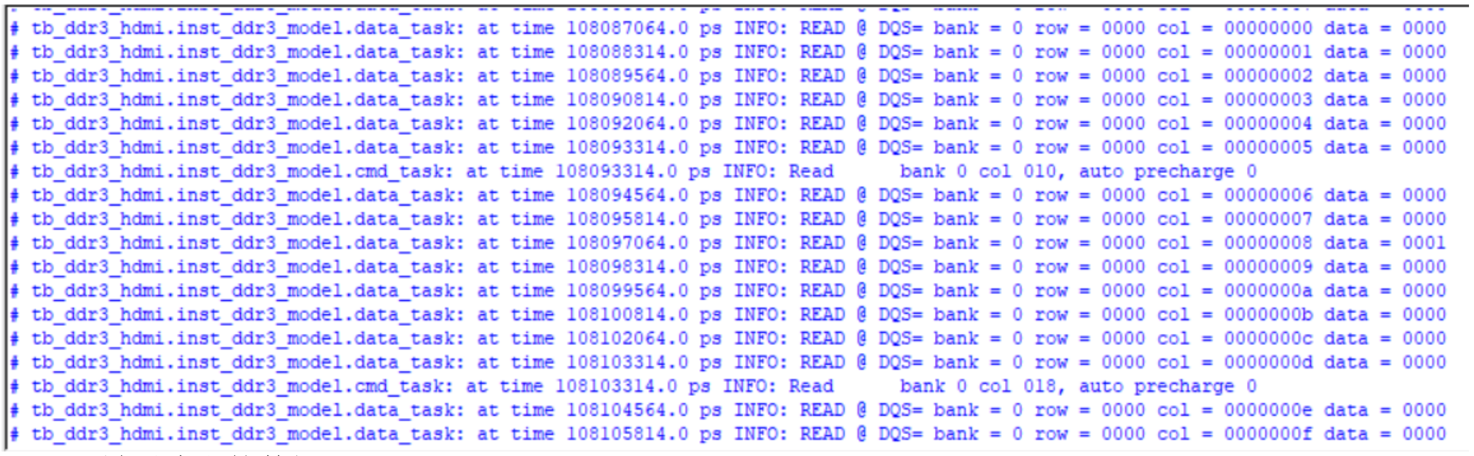


```
clk                 1'h1
rst                 1'h0
rd_cmd_start        1'h0
rd_cmd_bl           7'h40            40
rd_cmd_instr        3'h1             1
rd_cmd_addr         28'h0000000      0000000
rd_data_valid       1'h0
rd_data_128bit      128'h000000000...  00000000000000000000000000000000   00000000000000000000000
rd_end              1'h0
app_en              1'h0
app_cmd             3'h0             0   1
app_addr            28'h0000000      0000000   0000000
app_rdy             1'h1
app_rd_data         128'd0           0   32
app_rd_data_valid   1'h0
cmd_bl              7'h00            00   40
cmd_instr           3'h0             0   1
cmd_addr            28'h0000000      0000000   0000000
app_en_r            1'h0
addr_cnt            8'h00            00   00
data_cnt            8'h00            00   00
rd_end_r            1'h0
```

时序上基本没有什么问题。那就要查看读取的数据了。

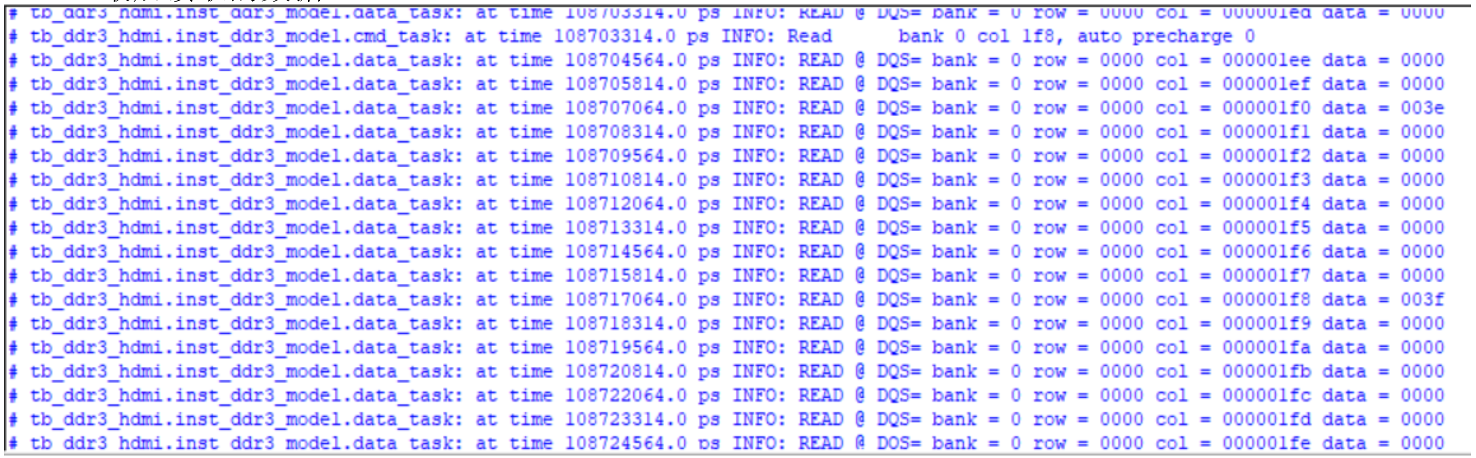
- 最后观察读取的数据

开始读取的数据



```
# tb_ddr3_hdmi.inst_ddr3_model.data_task: at time 108087064.0 ps INFO: READ @ DQS= bank = 0 row = 0000 col = 00000000 data = 0000
# tb_ddr3_hdmi.inst_ddr3_model.data_task: at time 108088314.0 ps INFO: READ @ DQS= bank = 0 row = 0000 col = 00000001 data = 0000
# tb_ddr3_hdmi.inst_ddr3_model.data_task: at time 108089564.0 ps INFO: READ @ DQS= bank = 0 row = 0000 col = 00000002 data = 0000
# tb_ddr3_hdmi.inst_ddr3_model.data_task: at time 108090814.0 ps INFO: READ @ DQS= bank = 0 row = 0000 col = 00000003 data = 0000
# tb_ddr3_hdmi.inst_ddr3_model.data_task: at time 108092064.0 ps INFO: READ @ DQS= bank = 0 row = 0000 col = 00000004 data = 0000
# tb_ddr3_hdmi.inst_ddr3_model.data_task: at time 108093314.0 ps INFO: READ @ DQS= bank = 0 row = 0000 col = 00000005 data = 0000
# tb_ddr3_hdmi.inst_ddr3_model.cmd_task: at time 108093314.0 ps INFO: Read      bank 0 col 010, auto precharge 0
# tb_ddr3_hdmi.inst_ddr3_model.data_task: at time 108094564.0 ps INFO: READ @ DQS= bank = 0 row = 0000 col = 00000006 data = 0000
# tb_ddr3_hdmi.inst_ddr3_model.data_task: at time 108095814.0 ps INFO: READ @ DQS= bank = 0 row = 0000 col = 00000007 data = 0000
# tb_ddr3_hdmi.inst_ddr3_model.data_task: at time 108097064.0 ps INFO: READ @ DQS= bank = 0 row = 0000 col = 00000008 data = 0001
# tb_ddr3_hdmi.inst_ddr3_model.data_task: at time 108098314.0 ps INFO: READ @ DQS= bank = 0 row = 0000 col = 00000009 data = 0000
# tb_ddr3_hdmi.inst_ddr3_model.data_task: at time 108099564.0 ps INFO: READ @ DQS= bank = 0 row = 0000 col = 0000000a data = 0000
# tb_ddr3_hdmi.inst_ddr3_model.data_task: at time 108100814.0 ps INFO: READ @ DQS= bank = 0 row = 0000 col = 0000000b data = 0000
# tb_ddr3_hdmi.inst_ddr3_model.data_task: at time 108102064.0 ps INFO: READ @ DQS= bank = 0 row = 0000 col = 0000000c data = 0000
# tb_ddr3_hdmi.inst_ddr3_model.data_task: at time 108103314.0 ps INFO: READ @ DQS= bank = 0 row = 0000 col = 0000000d data = 0000
# tb_ddr3_hdmi.inst_ddr3_model.cmd_task: at time 108103314.0 ps INFO: Read      bank 0 col 018, auto precharge 0
# tb_ddr3_hdmi.inst_ddr3_model.data_task: at time 108104564.0 ps INFO: READ @ DQS= bank = 0 row = 0000 col = 0000000e data = 0000
# tb_ddr3_hdmi.inst_ddr3_model.data_task: at time 108105814.0 ps INFO: READ @ DQS= bank = 0 row = 0000 col = 0000000f data = 0000
```

最后读取的数据

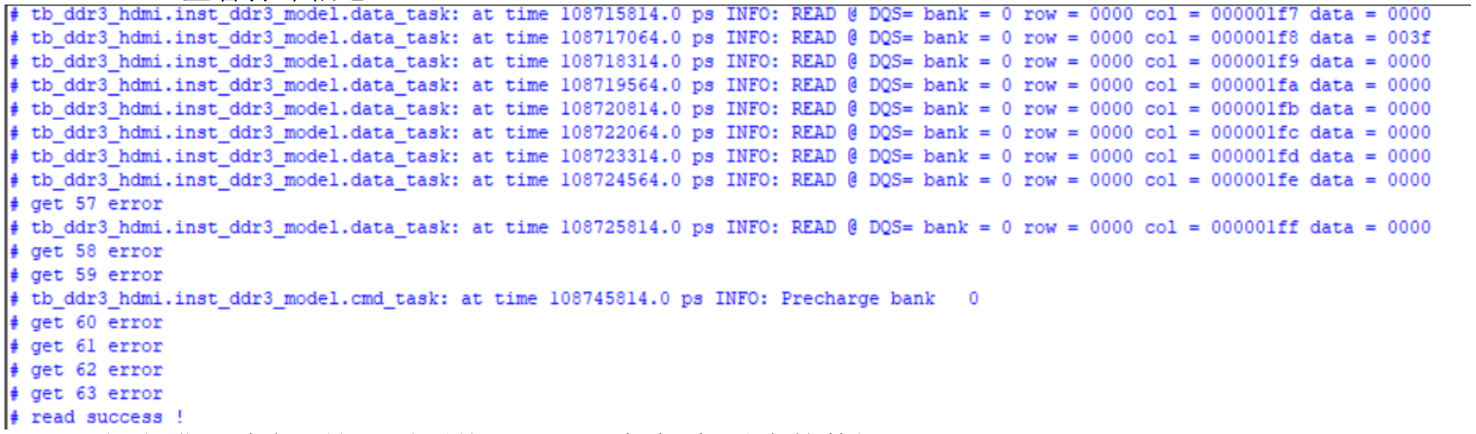


```
# tb_ddr3_hdmi.inst_ddr3_model.data_task: at time 108703314.0 ps INFO: READ @ DQS= bank = 0 row = 0000 col = 000001ed data = 0000
# tb_ddr3_hdmi.inst_ddr3_model.cmd_task: at time 108703314.0 ps INFO: Read      bank 0 col 1f8, auto precharge 0
# tb_ddr3_hdmi.inst_ddr3_model.data_task: at time 108704564.0 ps INFO: READ @ DQS= bank = 0 row = 0000 col = 000001ee data = 0000
# tb_ddr3_hdmi.inst_ddr3_model.data_task: at time 108705814.0 ps INFO: READ @ DQS= bank = 0 row = 0000 col = 000001ef data = 0000
# tb_ddr3_hdmi.inst_ddr3_model.data_task: at time 108707064.0 ps INFO: READ @ DQS= bank = 0 row = 0000 col = 000001f0 data = 003e
# tb_ddr3_hdmi.inst_ddr3_model.data_task: at time 108708314.0 ps INFO: READ @ DQS= bank = 0 row = 0000 col = 000001f1 data = 0000
# tb_ddr3_hdmi.inst_ddr3_model.data_task: at time 108709564.0 ps INFO: READ @ DQS= bank = 0 row = 0000 col = 000001f2 data = 0000
# tb_ddr3_hdmi.inst_ddr3_model.data_task: at time 108710814.0 ps INFO: READ @ DQS= bank = 0 row = 0000 col = 000001f3 data = 0000
# tb_ddr3_hdmi.inst_ddr3_model.data_task: at time 108712064.0 ps INFO: READ @ DQS= bank = 0 row = 0000 col = 000001f4 data = 0000
# tb_ddr3_hdmi.inst_ddr3_model.data_task: at time 108713314.0 ps INFO: READ @ DQS= bank = 0 row = 0000 col = 000001f5 data = 0000
# tb_ddr3_hdmi.inst_ddr3_model.data_task: at time 108714564.0 ps INFO: READ @ DQS= bank = 0 row = 0000 col = 000001f6 data = 0000
# tb_ddr3_hdmi.inst_ddr3_model.data_task: at time 108715814.0 ps INFO: READ @ DQS= bank = 0 row = 0000 col = 000001f7 data = 0000
# tb_ddr3_hdmi.inst_ddr3_model.data_task: at time 108717064.0 ps INFO: READ @ DQS= bank = 0 row = 0000 col = 000001f8 data = 003f
# tb_ddr3_hdmi.inst_ddr3_model.data_task: at time 108718314.0 ps INFO: READ @ DQS= bank = 0 row = 0000 col = 000001f9 data = 0000
# tb_ddr3_hdmi.inst_ddr3_model.data_task: at time 108719564.0 ps INFO: READ @ DQS= bank = 0 row = 0000 col = 000001fa data = 0000
# tb_ddr3_hdmi.inst_ddr3_model.data_task: at time 108720814.0 ps INFO: READ @ DQS= bank = 0 row = 0000 col = 000001fb data = 0000
# tb_ddr3_hdmi.inst_ddr3_model.data_task: at time 108722064.0 ps INFO: READ @ DQS= bank = 0 row = 0000 col = 000001fc data = 0000
# tb_ddr3_hdmi.inst_ddr3_model.data_task: at time 108723314.0 ps INFO: READ @ DQS= bank = 0 row = 0000 col = 000001fd data = 0000
# tb_ddr3_hdmi.inst_ddr3_model.data_task: at time 108724564.0 ps INFO: READ @ DQS= bank = 0 row = 0000 col = 000001fe data = 0000
```

发现数据基本正确。

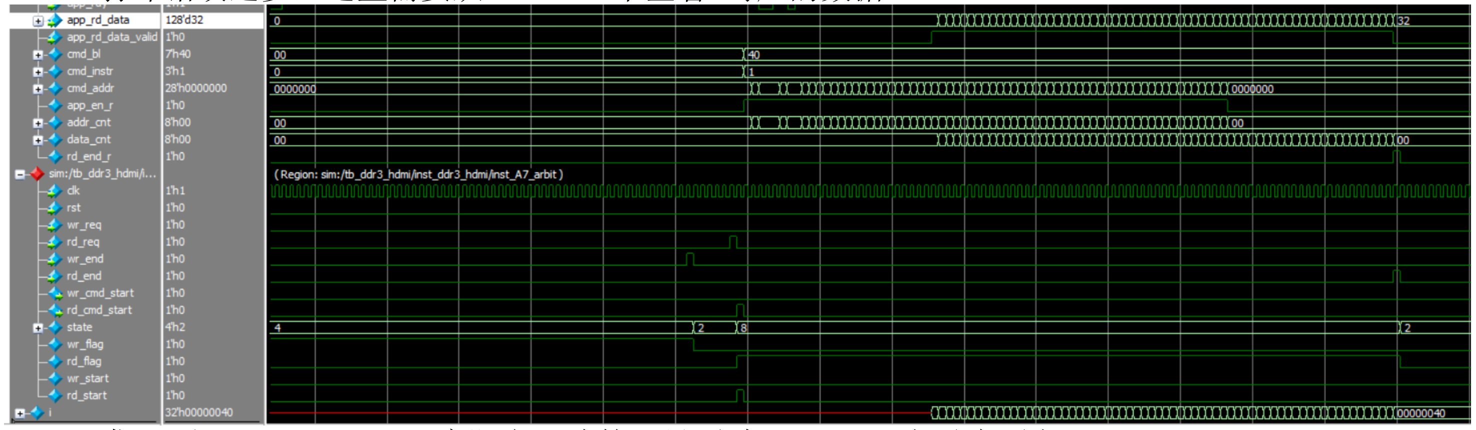
3. 查看读写是否一致

（1）查看打印信息



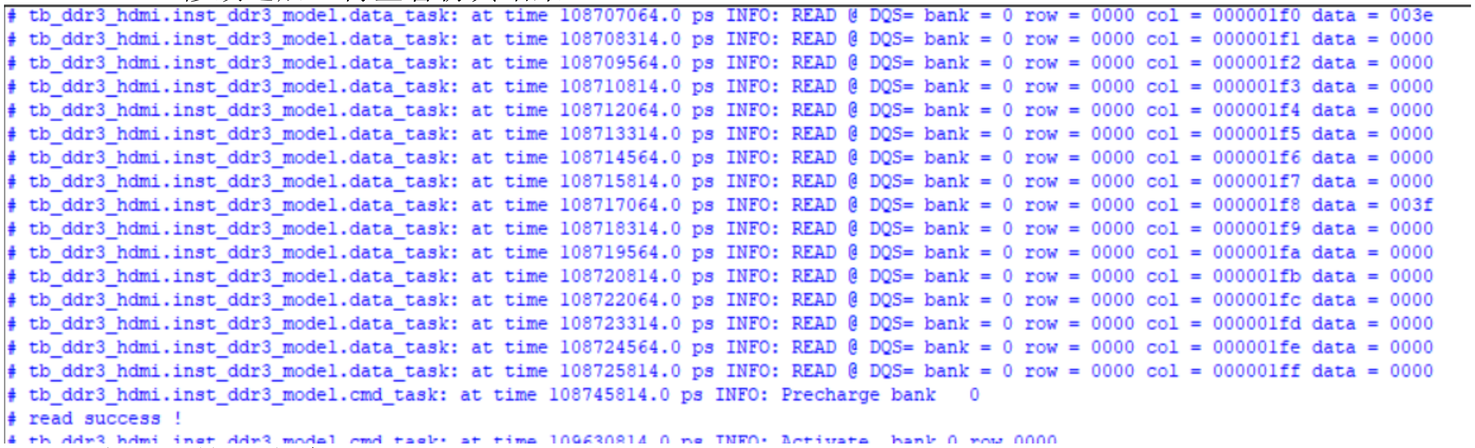
```
# tb_ddr3_hdmi.inst_ddr3_model.data_task: at time 108715814.0 ps INFO: READ @ DQS= bank = 0 row = 0000 col = 000001f7 data = 0000
# tb_ddr3_hdmi.inst_ddr3_model.data_task: at time 108717064.0 ps INFO: READ @ DQS= bank = 0 row = 0000 col = 000001f8 data = 003f
# tb_ddr3_hdmi.inst_ddr3_model.data_task: at time 108718314.0 ps INFO: READ @ DQS= bank = 0 row = 0000 col = 000001f9 data = 0000
# tb_ddr3_hdmi.inst_ddr3_model.data_task: at time 108719564.0 ps INFO: READ @ DQS= bank = 0 row = 0000 col = 000001fa data = 0000
# tb_ddr3_hdmi.inst_ddr3_model.data_task: at time 108720814.0 ps INFO: READ @ DQS= bank = 0 row = 0000 col = 000001fb data = 0000
# tb_ddr3_hdmi.inst_ddr3_model.data_task: at time 108722064.0 ps INFO: READ @ DQS= bank = 0 row = 0000 col = 000001fc data = 0000
# tb_ddr3_hdmi.inst_ddr3_model.data_task: at time 108723314.0 ps INFO: READ @ DQS= bank = 0 row = 0000 col = 000001fd data = 0000
# tb_ddr3_hdmi.inst_ddr3_model.data_task: at time 108724564.0 ps INFO: READ @ DQS= bank = 0 row = 0000 col = 000001fe data = 0000
# get 57 error
# tb_ddr3_hdmi.inst_ddr3_model.data_task: at time 108725814.0 ps INFO: READ @ DQS= bank = 0 row = 0000 col = 000001ff data = 0000
# get 58 error
# get 59 error
# tb_ddr3_hdmi.inst_ddr3_model.cmd_task: at time 108745814.0 ps INFO: Precharge bank   0
# get 60 error
# get 61 error
# get 62 error
# get 63 error
# read success !
```

打印错误过多，这里需要从modelsim中查看i对应的数据。



发现i和rd_data_128bit实际上一致的，只不过testbench上对应不上。

（2）修改之后，再查看仿真结果。



```
# tb_ddr3_hdmi.inst_ddr3_model.data_task: at time 108707064.0 ps INFO: READ @ DQS= bank = 0 row = 0000 col = 000001f0 data = 003e
# tb_ddr3_hdmi.inst_ddr3_model.data_task: at time 108708314.0 ps INFO: READ @ DQS= bank = 0 row = 0000 col = 000001f1 data = 0000
# tb_ddr3_hdmi.inst_ddr3_model.data_task: at time 108709564.0 ps INFO: READ @ DQS= bank = 0 row = 0000 col = 000001f2 data = 0000
# tb_ddr3_hdmi.inst_ddr3_model.data_task: at time 108710814.0 ps INFO: READ @ DQS= bank = 0 row = 0000 col = 000001f3 data = 0000
# tb_ddr3_hdmi.inst_ddr3_model.data_task: at time 108712064.0 ps INFO: READ @ DQS= bank = 0 row = 0000 col = 000001f4 data = 0000
# tb_ddr3_hdmi.inst_ddr3_model.data_task: at time 108713314.0 ps INFO: READ @ DQS= bank = 0 row = 0000 col = 000001f5 data = 0000
# tb_ddr3_hdmi.inst_ddr3_model.data_task: at time 108714564.0 ps INFO: READ @ DQS= bank = 0 row = 0000 col = 000001f6 data = 0000
# tb_ddr3_hdmi.inst_ddr3_model.data_task: at time 108715814.0 ps INFO: READ @ DQS= bank = 0 row = 0000 col = 000001f7 data = 0000
# tb_ddr3_hdmi.inst_ddr3_model.data_task: at time 108717064.0 ps INFO: READ @ DQS= bank = 0 row = 0000 col = 000001f8 data = 003f
# tb_ddr3_hdmi.inst_ddr3_model.data_task: at time 108718314.0 ps INFO: READ @ DQS= bank = 0 row = 0000 col = 000001f9 data = 0000
# tb_ddr3_hdmi.inst_ddr3_model.data_task: at time 108719564.0 ps INFO: READ @ DQS= bank = 0 row = 0000 col = 000001fa data = 0000
# tb_ddr3_hdmi.inst_ddr3_model.data_task: at time 108720814.0 ps INFO: READ @ DQS= bank = 0 row = 0000 col = 000001fb data = 0000
# tb_ddr3_hdmi.inst_ddr3_model.data_task: at time 108722064.0 ps INFO: READ @ DQS= bank = 0 row = 0000 col = 000001fc data = 0000
# tb_ddr3_hdmi.inst_ddr3_model.data_task: at time 108723314.0 ps INFO: READ @ DQS= bank = 0 row = 0000 col = 000001fd data = 0000
# tb_ddr3_hdmi.inst_ddr3_model.data_task: at time 108724564.0 ps INFO: READ @ DQS= bank = 0 row = 0000 col = 000001fe data = 0000
# tb_ddr3_hdmi.inst_ddr3_model.data_task: at time 108725814.0 ps INFO: READ @ DQS= bank = 0 row = 0000 col = 000001ff data = 0000
# tb_ddr3_hdmi.inst_ddr3_model.cmd_task: at time 108745814.0 ps INFO: Precharge bank   0
# read success !
# tb_ddr3_hdmi.inst_ddr3_model.cmd_task: at time 109630814.0 ps INFO: Activate  bank 0 row 0000
```

好了，这下成功了。

六、总结与讨论

1.多做多练多写

（1）前期的时序分析准备

（2）中期的写代码要认真

（3）最后调试要小心