**哈尔滨理工大学**

计算机科学与技术学院

课程设计报告

（2021-2022第二学期）

课 程 高级数字IC设计

题 目 基于AHB总线的SRAM控制器的设计

专 业 集成电路设计与集成系统

班 级 集成19-2

学 生 黄羽铧

学 号 1914020208

指导教师 杨兵

2022**年**6**月**

### 目录

[目录 2](#_Toc107149436)

[基于AHB总线的SRAM控制器的设计 3](#_Toc107149437)

[1.前言 3](#_Toc107149438)

[2.工作原理和整体结构 3](#_Toc107149439)

[3.模块设计 3](#_Toc107149440)

[3.1 AHB总线控制slave (ahb\_slave\_if.v) 4](#_Toc107149441)

[3.2 SRAM控制器（sramc\_top.v） 4](#_Toc107149442)

[3.3 SRAM 5](#_Toc107149443)

[4.RTL 设计 5](#_Toc107149444)

[5.仿真结果和分析 7](#_Toc107149445)

[6.总结 8](#_Toc107149446)

[参考文献 9](#_Toc107149447)

[附录 10](#_Toc107149448)

### 基于AHB总线的SRAM控制器的设计

### 前言

本设计为AHB接口的SRAM设计。为了初步对AHB总线进行了解与学习而进行的设计，主要为了设计SRAM与AHB总线通信的控制器，实现SRAM存储器与AHB总线的数据信息交换。

### 2.工作原理和整体结构

本设计一共分三个模块，分别是顶层模块sram\_top，从设备interface模块sram\_slave\_if与比较常见的SRAM模块sram\_core。本次设计中主要涉及的是SRAM从设备的interface的设计。sram\_top顶层模块主要负责把前面两个模块例化并连接起来。SRAM在整个系统中作为缓存，SRAM控制器实现SRAM存储器与AHB总线的数据信息交换，一端连接AHB总线，另一端连接SRAM，将AHB总线上的读写操作转换成标准的SRAM读写操作。

SRAM的工作原理很简单，SRAM控制单元根据接收到的总线控制信号，将这些信号处理转化为SRAM存储器可以识别的信号，发送到 SRAM存储器；之后，将经过地址译码的物理地址传送到存储器的地址总线，并将数据路径处理的数据送到SRAM存储器的数据总线。最后，SRAM存储器进行相应的读写访问。如果是写操作，SRAM 控制单元的任务就完成了，SRAM 存储器已将数据信息按照要求写入。如果是读操作，SRAM控制单元需要接收返回的读数据，将其送到数据路径，由数据路径将信息传输给AHB总线，最终实现总线对SRAM的读操作。

在设计过程中发现需要注意的是，AHB的总线传输数据位宽有8/16/32这几种传输模式，在本设计中，采用了4个SRAM，每个SRAM对应一个字。

### 3.模块设计

设计中采用的SRAM具有内嵌的MBIST，使用的是从网上下载的IP。设计的SRAM控制器一端连接AHB，另一端连接SRAM存储器。控制器设计目标是实现SRAM存储器与AHB总线的数据信息交换。

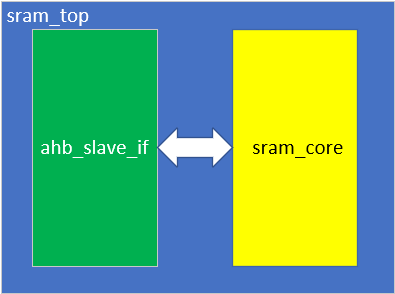


Figure 3.1 结构示意图

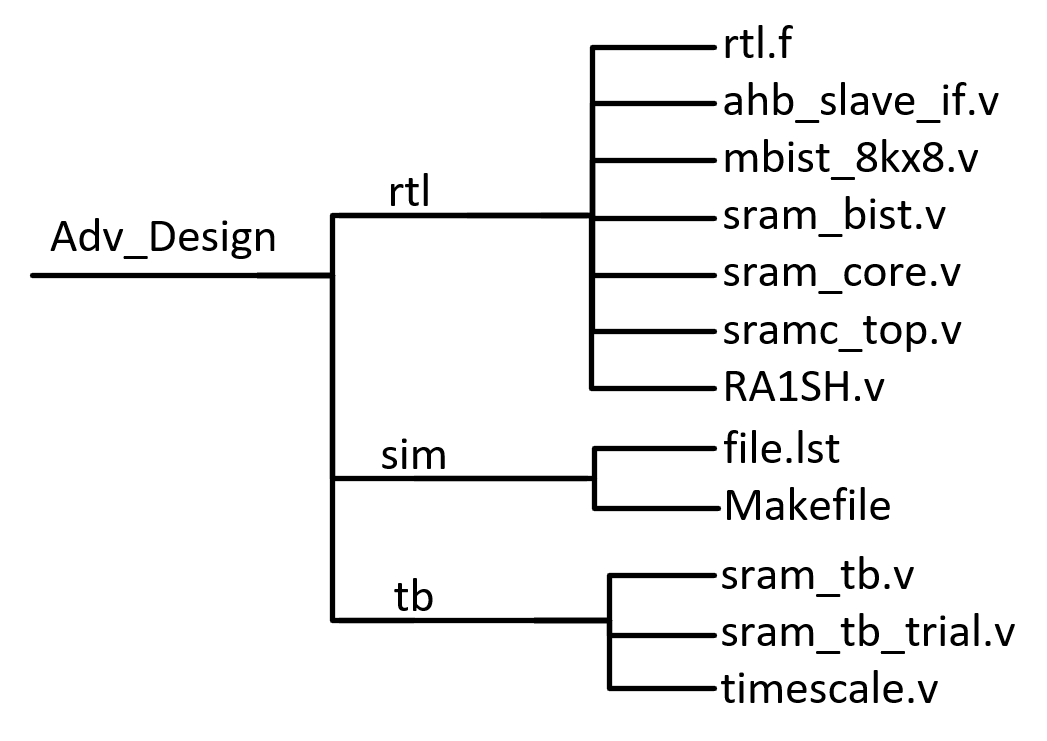


Figure 3.2 文件目录结构

#### 3.1 AHB总线控制slave (ahb\_slave\_if.v)

SRAM与控制器连接的接口，从AHB传输过来的数据信号都会经过总线控制单元的转换处理。来自官方手册的接口图如下所示：

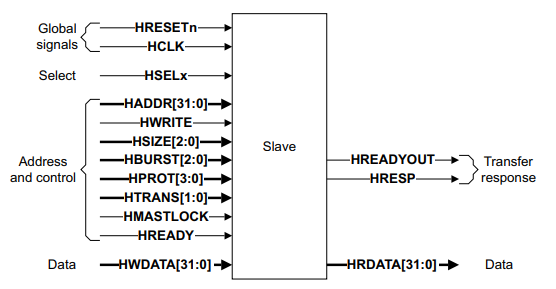


Figure 3.3 来自arm官方手册的Figure1-3对于slave的示例图

图中当HSEL信号有效时，总线控制单元与SRAM存储器控制单元之间可以进行信息传输，实现对SRAM的读写操作。总线控制单元做的就是接收来自AHB总线的地址和数据信息等等，将这些信息处理后送到SRAM单元，读写地址信息送到地址路径，写数据送到数据路径，并接收路径返回的数据。

#### 3.2 SRAM控制器（sramc\_top.v）

SRAM控制单元根据接收到的总线控制信号，将这些信号转化为SRAM存储器可以识别的信号，发送到SRAM存储器，之后将经过译码的物理地址传送到存储器的地址总线，并将数据路径处理的数据送到SRAM存储器的数据总线，SRAM存储器根据控制信号完成相应的读写访问操作。为了设计报告观感，模块代码放入下一小节。

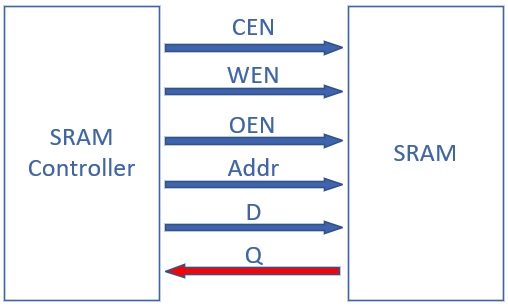


Figure 3.4 SRAM控制器模块的示意图

#### 3.3 SRAM

之前在课上学习的时候有学到，对于SRAM这种比较简单的结构来说，其数量，容量等参数是影响电路的功耗与性能的主要因素。本设计采用的是8k x 8的SRAM。根据不同地址控制器会选择不同块的SRAM。同时SRAM中分为上下两个库，库0包含sram\_bist 0~3，地址范围是0到31；库1包括sram\_bist 4~7，地址范围是32到63。

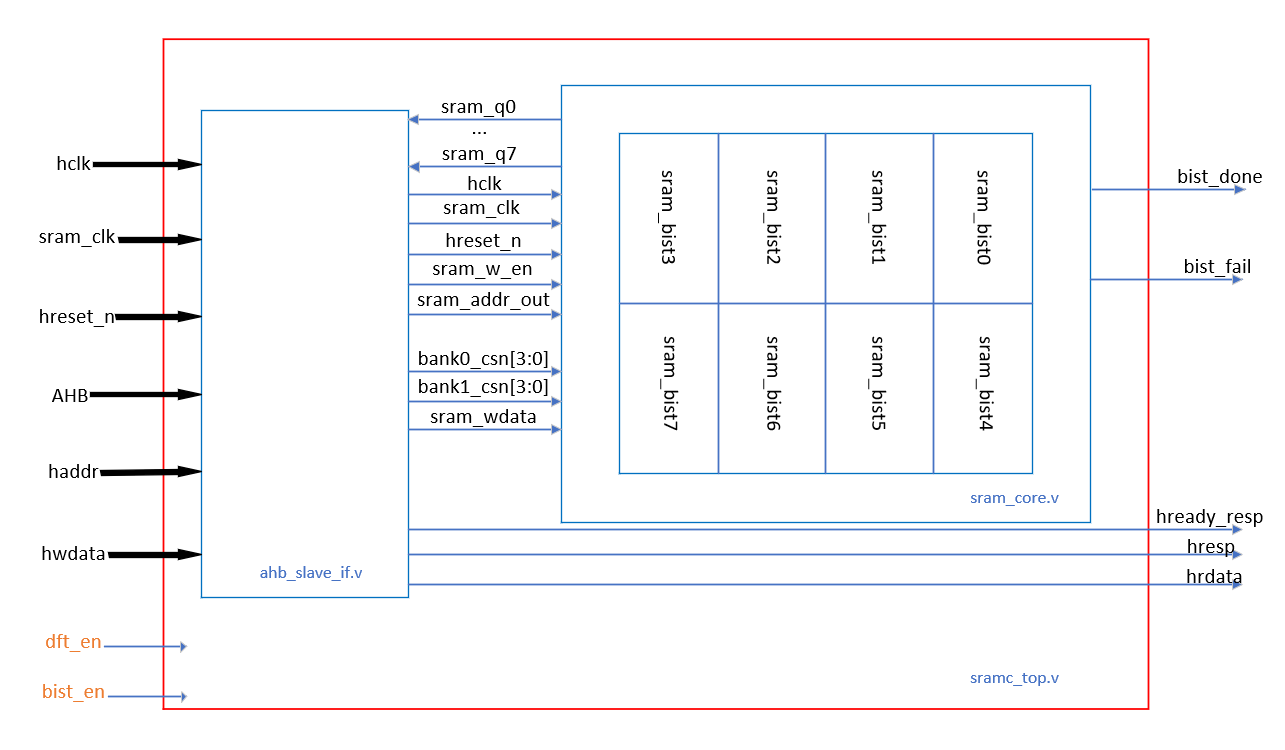


Figure 3.5 总体信号示例

这样设计可以使得在片选中未被选中的RAM片功耗很低，优化了相关PPA指标。

### 4.RTL 设计

SRAM控制器部分关键代码（ahb\_sramc）

/\*选用单周期读写SRAM且SRAM一直处于“OK”状态\*/

assign hready\_resp =1’b1;

assign hresp =2’b00;

/\*数据从SRAM传输到AHB总线 \*/

assign hrdata = sram\_data\_out

/\* 根据AHB总线协议，只有当TRANS信号为NONSEQ或者SEQ时，数据读写才有效，因此，需要保证SRAM读写时，TRANS信号有效。SRAM进行写操时，hwrite信号为高；SRAM进行读操作时，hwrite信号为低 \*/

assign sram\_write = ((htrans\_r == NONSEQ) || (htrans\_r == SEQ)) && hwrite\_r;

assign sram\_read = ((htrans\_r == NONSEQ) || (htrans\_r == SEQ)) && (!hwrite\_r);

/\*写使能信号低有效\*/

assign sram\_w\_en = !sram\_write;

/\*每个bank有4片SRAM，每一片SRAM有一个片选信号 \*/

always@(hsize\_sel or haddr\_sel)

begin

if(hsize\_sel == 2'b10)

sram\_csn = 4'b0;

/\*一次片选两片SRAM \*/

else if(hsize\_sel == 2'b01)

begin

if(haddr\_sel[1] == 1'b0)

sram\_csn = 4'b1100;

else

sram\_csn = 4'b0011;

end

/\*选择一片SRAM，具体到4片中的某一片\*/

else if(hsize\_sel == 2'b00)

begin

case(haddr\_sel)

2'b00 : sram\_csn = 4'b1110;

2'b01 : sram\_csn = 4'b1101;

2'b10 : sram\_csn = 4'b1011;

2'b11 : sram\_csn = 4'b0111;

default : sram\_csn = 4'b1111;

endcase

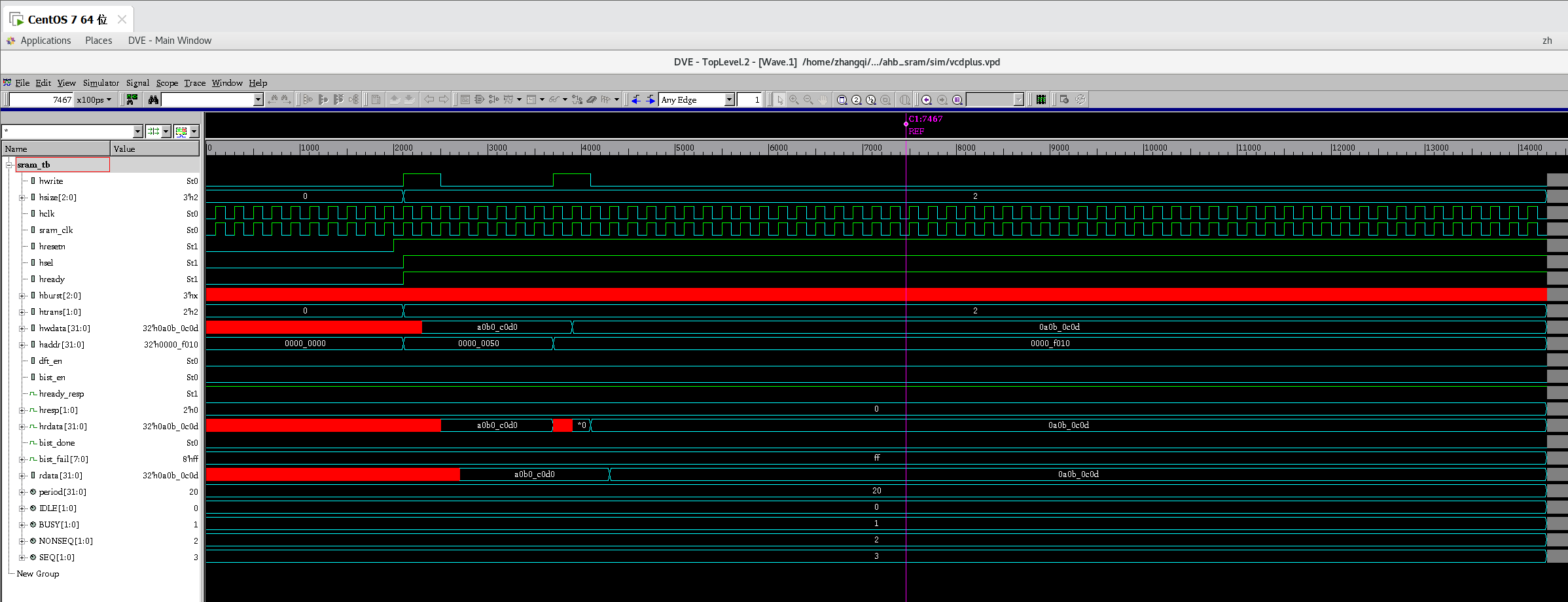
end

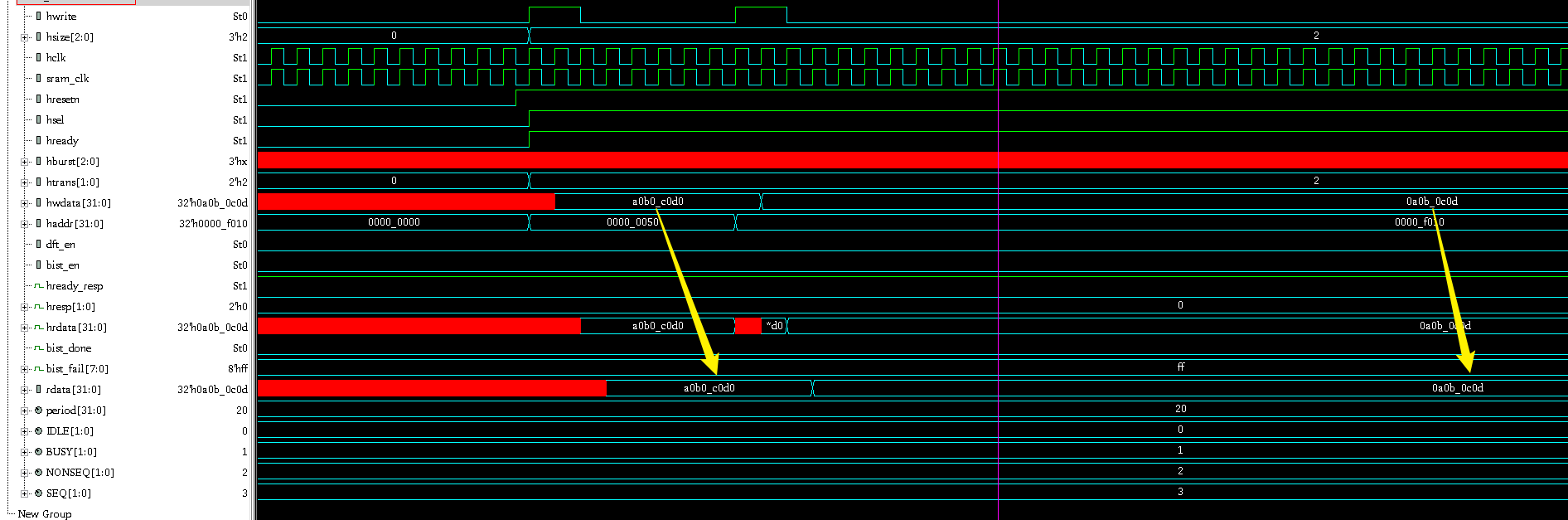
else

sram\_csn = 4'b1111;

end

### 5.仿真结果和分析

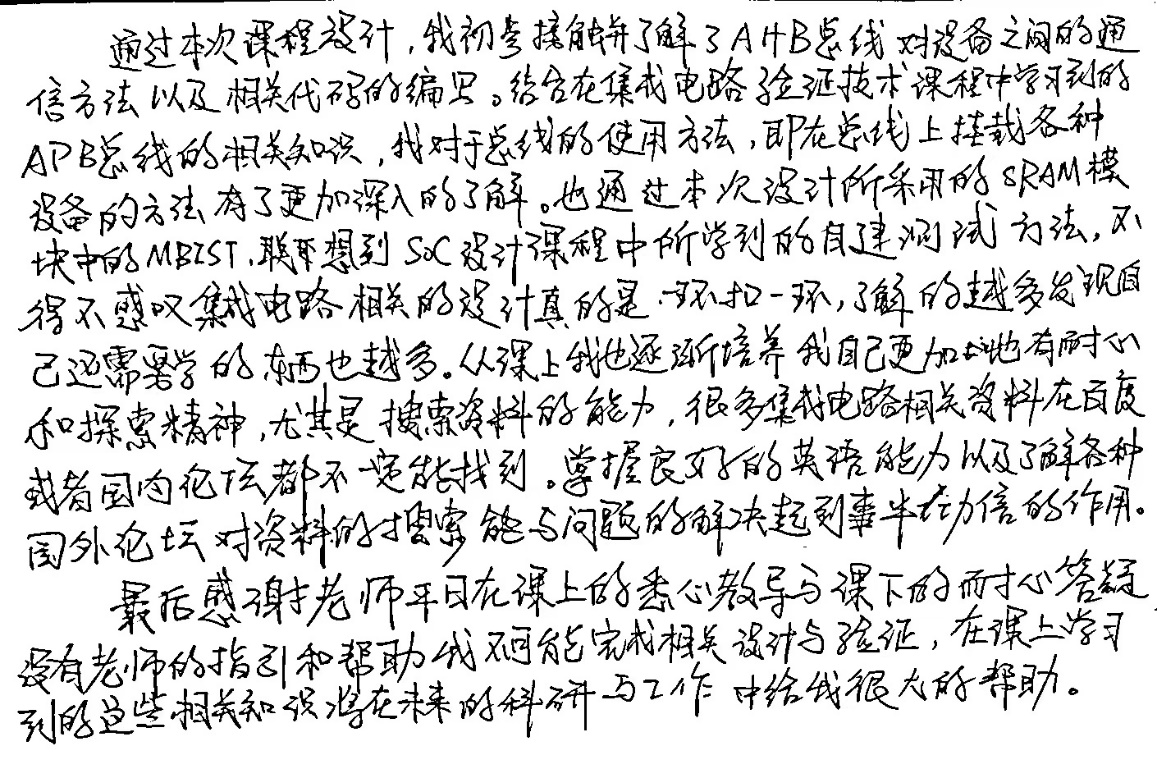




设计能够正确地把AHB总线输入的数据存到SRAM中，并且能够成功正确地读取出来。

通过波形图可以看到设计中的各种信号都能够正常工作，设计目标的功能也成功实现，通过波形图可以看到控制器对于SRAM的地址的操作以及数据的操作。

### 6.总结

****

（草稿）通过本次课程设计，我初步接触并了解了AHB总线与设备之间的通信方法以及相关代码的编写。结合在集成电路验证技术课程中学习到的APB总线的相关知识，我对于总线的使用方法，即在总线上挂载各种设备的方法有了更加深入的了解。也通过本次设计所采用的SRAM模块中的MBIST，联想到SoC课程中所学到的自建测试方法，不得不感叹集成电路相关的设计真的是一环扣一环，了解的越多发现自己还需要学的东西也越多。从课上我也逐渐培养我自己更加的有耐心和探索精神，尤其是搜索资料的能力，很多集成电路相关资料在百度或者国内论坛都不太好搜索，掌握良好的英语能力以及了解各种国外论坛能对与资料的搜索与问题的解决起到事半功倍的作用。

最后感谢老师在平日上课或课下的悉心教导，没有老师的指引和帮助我不可能完成相关设计，在课上学习到的这些相关知识将在未来的科研与工作中给我很大的帮助。

参考文献

[1]. ARM AMBA 5 AHB Protocol Specification AHB5, AHB-Lite Copyright © 2001, 2006, 2010, 2015 ARM Limited or its affiliates. All rights reserved.

[2].Modeling and analysis of the AMBA bus using CSP and B January 2007 [Concurrent Systems Engineering Series](https://www.researchgate.net/journal/Concurrent-Systems-Engineering-Series-1383-7575) 65:379-398

[3]. IEEE Standard for Universal Verification Methodology Language Reference Manual Developed by the Design Automation Standards Committee of the IEEE Computer Society Approved 4 June 2020 IEEE SA Standards Board

### 附录

代码附录

module ahb\_slave\_if(

//input signals

input hclk,

input hresetn,

input hsel,

input hwrite,

input hready,

input [2:0] hsize,

input [1:0] htrans,

input [2:0] hburst, // hard code -> parameter

input [31:0] hwdata,

input [31:0] haddr,

//output signals

output hready\_resp,

output [1:0] hresp,

output [31:0] hrdata,

//sram output

input [7:0] sram\_q0, // 8bits

input [7:0] sram\_q1,

input [7:0] sram\_q2,

input [7:0] sram\_q3,

input [7:0] sram\_q4,

input [7:0] sram\_q5,

input [7:0] sram\_q6,

input [7:0] sram\_q7,

output sram\_w\_en, // 0:write, 1:read

output [12:0] sram\_addr\_out,

output [31:0] sram\_wdata, //写sram数据

output [3:0] bank0\_csn, //四字节可以单独写入

output [3:0] bank1\_csn

);

//-------------------------------------------------------

//internal registers used for temp the input ahb signals

//-------------------------------------------------------

//temperate all the AHB input signals

reg hwrite\_r;

reg [2:0] hsize\_r ;

reg [2:0] hburst\_r;

reg [1:0] htrans\_r;

reg [31:0] haddr\_r;

reg [3:0] sram\_csn;

//------------------------------------------------------

//Internal signals

//------------------------------------------------------

//"haddr\_sel " and "hsize\_sel" used to generate banks of

//sram: "bank0\_sel" and "bank1\_sel".

wire [1:0] haddr\_sel;

wire [1:0] hsize\_sel;

wire bank\_sel;

wire sram\_csn\_en; //sram chip select enable

wire sram\_write; //sram write enable signal from AHB bus

wire sram\_read; //sram read enable signal from AHB bus

wire [15:0] sram\_addr; //sram address from AHB bus

wire [31:0] sram\_data\_out; //data read from sram and send to AHB bus

parameter IDLE = 2'b00,

BUSY = 2'b01,

NONSEQ = 2'b10,

SEQ = 2'b11;

//---------------------------------------------------------

// Combinatorial portion

//---------------------------------------------------------

//assign the response and read data of the ahb slave

//In order to implement the sram function-writing or reading

//in one cycle, the value of hready\_resp is always "1".

assign hready\_resp = 1'b1; // Singal Cycle

assign hresp = 2'b00; // OK

//---------------------------------------------------------

//sram data output to AHB bus

//---------------------------------------------------------

assign hrdata = sram\_data\_out; //组合逻辑读，CPU读SRAM，地址有效则立即读

//Choose the right data output of the two banks(bank0, bank1) according

//to the value of bank\_sel. If bank\_sel = 1'b1, bank0 sleceted, or

//bank1 selected.

assign sram\_data\_out = (bank\_sel) ?

{sram\_q3, sram\_q2, sram\_q1, sram\_q0} :

{sram\_q7, sram\_q6, sram\_q5, sram\_q4} ;

//Generate sram write and read enable signals.

assign sram\_write = ((htrans\_r == NONSEQ) || (htrans\_r == SEQ)) && hwrite\_r;

assign sram\_read = ((htrans\_r == NONSEQ) || (htrans\_r == SEQ)) && (!hwrite\_r);

assign sram\_w\_en = !sram\_write;

//generate sram address

//SRAM总寻址64K 0x0--0xffff

assign sram\_addr = haddr\_r [15:0];//64K 8\*8K

assign sram\_addr\_out = sram\_addr[14:2]; // word

//Generate bank select signals by the value of sram\_addr[15].

//Each bank(32kx32) comprises of four sram block(8kx8), and

//the width of the address of the bank is 15 bits(14~0), so

//the sram\_addr[15] is the minimun of the next bank. If its

//value is "1", it means the next bank is selcted.

assign sram\_csn\_en = (sram\_write || sram\_read);

//低32K bank0 高32K bank1

assign bank\_sel = (sram\_csn\_en && (sram\_addr[15] == 1'b0)) ? 1'b1 : 1'b0;

assign bank0\_csn = (sram\_csn\_en && (sram\_addr[15] == 1'b0)) ? sram\_csn : 4'b1111;

assign bank1\_csn = (sram\_csn\_en && (sram\_addr[15] == 1'b1)) ? sram\_csn : 4'b1111;

//signals used to generating sram chip select signal in one bank.

assign haddr\_sel = sram\_addr[1:0];

assign hsize\_sel = hsize\_r [1:0];

//-------------------------------------------------------

//data from ahb writing into sram

//-------------------------------------------------------

assign sram\_wdata = hwdata;

//-------------------------------------------------------

//Generate the sram chip selecting signals in one bank.

//The resluts show the AHB bus write or read how many data

//once a time: byte, halfword or word.

//---------------------------------------------------------

always@(hsize\_sel or haddr\_sel) begin

if(hsize\_sel == 2'b10)//32bit

sram\_csn = 4'b0;

else if(hsize\_sel == 2'b01) begin//16bit

if(haddr\_sel[1] == 1'b0) //little-endian

sram\_csn = 4'b1100;

else

sram\_csn = 4'b0011;

end

else if(hsize\_sel == 2'b00) begin//8bit

case(haddr\_sel)

2'b00 : sram\_csn = 4'b1110;

2'b01 : sram\_csn = 4'b1101;

2'b10 : sram\_csn = 4'b1011;

2'b11 : sram\_csn = 4'b0111;

default : sram\_csn = 4'b1111;

endcase

end

else

sram\_csn = 4'b1111;

end

//--------------------------------------------------------

// Sequential portion

//--------------------------------------------------------

//tmp the ahb address and control signals

always@(posedge hclk , negedge hresetn) begin

if(!hresetn) begin

hwrite\_r <= 1'b0 ;

hsize\_r <= 3'b0 ;

hburst\_r <= 3'b0 ;

htrans\_r <= 2'b0 ;

haddr\_r <= 32'b0 ;

end

else if(hsel && hready) begin //hsel要片选，否则信号一直在翻转，可能会功能错误，并且功耗大

hwrite\_r <= hwrite ;//写SRAM时，把控制信号寄存。因为写操作时，要把地址打一拍，和数据对齐

hsize\_r <= hsize ;

// hburst\_r <= hburst ;//AHB中master会把burst传输所有地址传递过来，AXI只传递起始地址。此处用处不大。减少一个REG

htrans\_r <= htrans ;

haddr\_r <= haddr ;

end else begin

hwrite\_r <= 1'b0 ;

hsize\_r <= 3'b0 ;

hburst\_r <= 3'b0 ;

htrans\_r <= 2'b0 ;

haddr\_r <= 32'b0 ;

end

end

endmodule