

SoC 微体系结构设计在线教学案例展示

一、案例基本信息

课程负责人：张剑贤

所在学校：西安电子科技大学

课程名称：SoC 微体系结构设计

课程教材：自编 SoC 微体系结构设计讲义，参考教材《SoC 设计方法与实现（第三版）》

授课对象：本科三年级

授课平台：雨课堂/腾讯课堂/学在西电/腾讯会议/QQ 群

二、案例综述

（包括本课程运用信息技术在课程体系、教学内容和教学方法等方面的改革情况，教学方案综述，应对疫情的教学设计，主要教学模式实施经验分享，教学过程数据统计分析，教学效果分析或学生反馈）

针对计算机硬件系列课程内容相对独立，知识结构关联性不足，本课程采用基于基础应用解决课程之间基本概念的横向联系和工作原理的纵向支撑的综合教学方式，突破知识孤岛的封闭特性，将多个课程知识相互联系，通过学生动手实践，将多课程的知识点聚集成计算机学科的基础知识面。

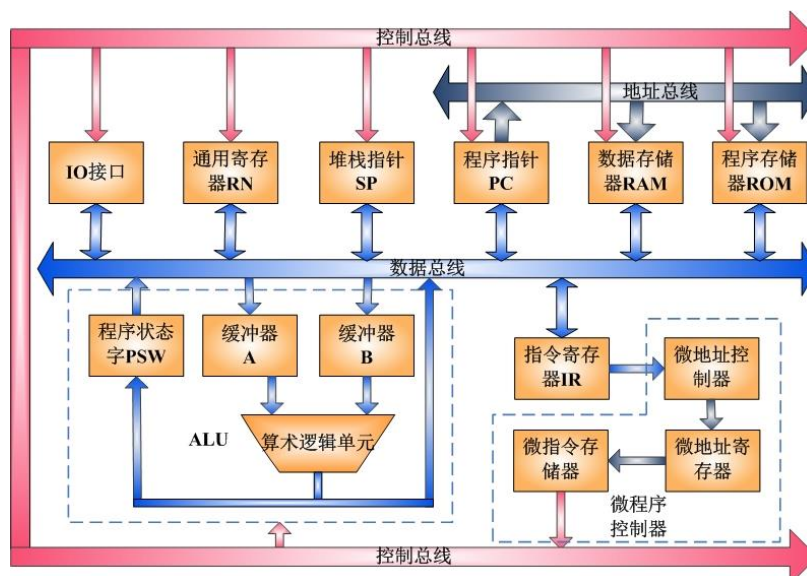
（1）目标导向，注重实践的改革思路。以 8 位 SoC 系统设计为目标，以硬件描述语言设计实现为主线的改革主线。从逻辑电路的工作方式深入剖析计算机的工作原理，从接口电路设计应用角度诠释处理器与逻辑电路之间的关系，注重于 CPU 的设计与逻辑电路的设计实现及 FPGA 验证。

（2）以计算机微体系结构实现为目标，构建计算机专业硬件类理论知识纵向体系结构。综合运用数字电路、计算机组成原理、微机原理、硬件描述语言和 FPGA 等知识和技术方法完成一个完整的 SoC 系统设计。计算机组织与体系结构课程为 SoC 设计提供了 SoC 结构及片上 IP 核模块互连方式的基础理论。数字电路提供了组合和时序逻辑电路设计的方法。硬件描述语言 VHDL 为 SoC 功能模块提供了实现的途径。汇编语言或 C 语言可实现 SoC 具体应用功能。通过 SoC 微体系结构设计课程，可以将所学的计算机组成原理、数字电路。硬件描述语言、汇编，数据结构等基础知识关联起来，融合成新的知识结构。建立计算机专业硬件类理论知识纵向体系结构，有利于提高学生系统分析与设计能力、软硬件协同综合开发实践能力以及创新能力。

（一）教学方案

本课程面向计算机科学与技术专业嵌入式方向大三学生。课程注重基础知识的应用、理论到实践的转化。课程主要内容包括 SoC 系统的基本结构及设计方法、基于硬件描述语言的数字系统基本逻辑电路设计方法、加减乘除算术运算部件原理及设计方法、存储器结构及设计方法，以及 8 位 SoC 系统设计方法及优化等内容。重点讨论基于 FPGA 的 8 位 SoC 系统设计方法以及各个功能部件的设计开发，包括指令集、程序指针 PC、指令寄存器 IR、

算术逻辑单元 ALU、通用寄存器 RN、程序存储器 ROM、数据存储器 RAM、堆栈指针 SP、通用 IO 接口以及微程序控制器等功能模块设计实现。



课程以“学生为中心，教师为引导”，采用从简单到复杂、从功能模块到系统综合的逐层递进的启发式教学方法。以设计实现为教学重点，从设计的角度来剖析计算机功能模块的结构及基于硬件描述语言的实现思路。重点理解及分析计算机指令执行过程中各个功能模块之间的信号传递及时序关系。在较为完整的综合设计项目实现过程中引导学生了解计算机内部工作原理及结构，掌握硬件描述语言设计方法及 FPGA 设计验证方法；引导学生根据需求设计功能结构，设计并验证功能要求，并通过测试与分析提高综合实践能力。

(二) “学讲论辅”四位一体

因为疫情突发，大部分学生没有携带教材或相关资料回家。为了确保疫情期间“延学不停学”的顺利开展，学校紧急筹备建设了“学在西电”学习平台。本课程将课件及往年现场教学视频按照章节知识点进行分解整理，在“学在西电”平台上建立了具有 36 个任务节点的视频录像及课件。

西安电子科技大学

SOC微体系结构设计课程门户

首页 活动 统计

目录 编辑 发放对象

发放 统计 新建话题

- 第1章 SoC设计概论
 - 1.1 SoC设计关键技术 3 ✓ 8%
 - 1.2 SoC总线结构 0 ✓ 0%
 - 1.3 FPGA结构分析 1 ✓ 8%
 - 第2章 硬件描述语言VHDL设计
 - 2.1 VHDL程序结构 2 ✓ 6%
 - 2.2 VHDL基本元素 1 ✓ 4%
 - 2.3 VHDL基本逻辑语句 1 ✓ 3%
 - 2.4 VHDL描述方式 2 ✓ 3%
 - 2.5 组合逻辑电路设计 1 ✓ 3%
 - 2.6 时序逻辑电路设计 1 ✓ 3%
 - 2.7 有限状态机设计 1 ✓ 1%
 - 2.8 EDA工具使用 1 ✓ 4%

为了确保教学质量，提高学生的自主学习能动性，本课程采用“录播视频+直播课堂+集中讨论+个性辅导”相结合的“学讲论辅”四位一体的教学方法开展教学。学生通过课件或录播视频进行课前预习，直播课堂重点讲述知识难点重点以及方法引导。课后对较为集中的问题进行集体答疑解惑，QQ群随时随地进行“一对一”个性辅导答疑。



(1) 自主学习

在课前布置学习任务，以“问题为驱动，激发学生的学习兴趣”。学生在“学在西电”平台上观看现场教学录播视频及课件，学习相关章节的知识内容。“学在西电”平台会自动记录及统计学生的学习情况，包括访问的方式、访问时段，观看教学视频时长等功能。考虑到部分学生家在边远山区网络信号不好，或者家境贫困，反复在线观看视频会产生较多的流量费用，课程组决定将课件资料及视频录像直接分享给选课学生，方便学生利用闲暇时间自主学习。此外本课程还利用“学在西电”平台发布课程作业，学生需在规定时间内提交作业。作业在线批改后，学生可以查看到作业分数、正确答案及老师评语，可以进行自我分析。

(2) 重点讲述

直播课堂上主要根据学生的学习情况，重点讲述知识难点及重点部分，传授分析及设计的方法及技巧。充分利用雨课堂平台提供的“单选题、多选题、投票及问答题”等现场测试随机点名等方式，加强与学生的互动，提高学生的注意力及课程关注度。本次课程安排了72个现场测试，5次编程作业。雨课堂对答题情况（答对、答错和未作答）能够实时统计及显示，方便老师掌握课堂的动态。

利用腾讯课堂共享桌面进行代码及仿真结果展示，以及重点知识的标注及剖析。此外雨课堂和腾讯课堂均提供了视频回放功能，在课后学生也可以回看线上课堂的视频。

(3) 集中讨论

针对学生在学习过程中遇到的共性问题及疑难困惑地方，召开腾讯会议，进行集中答疑解惑。虽然大家天各一方，却相聚网络。学生没有了线下面对老师时的紧张忐忑，可以畅所欲言，发表个人看法。有时某人的话语引发了大家的思考，进而转向更进一步的讨

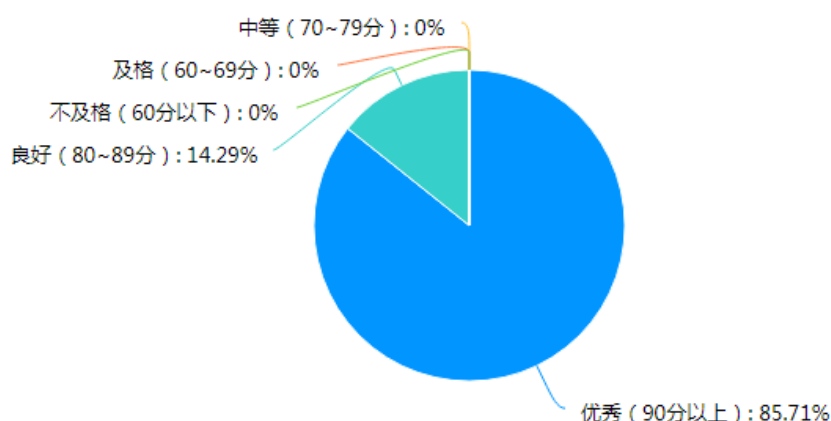
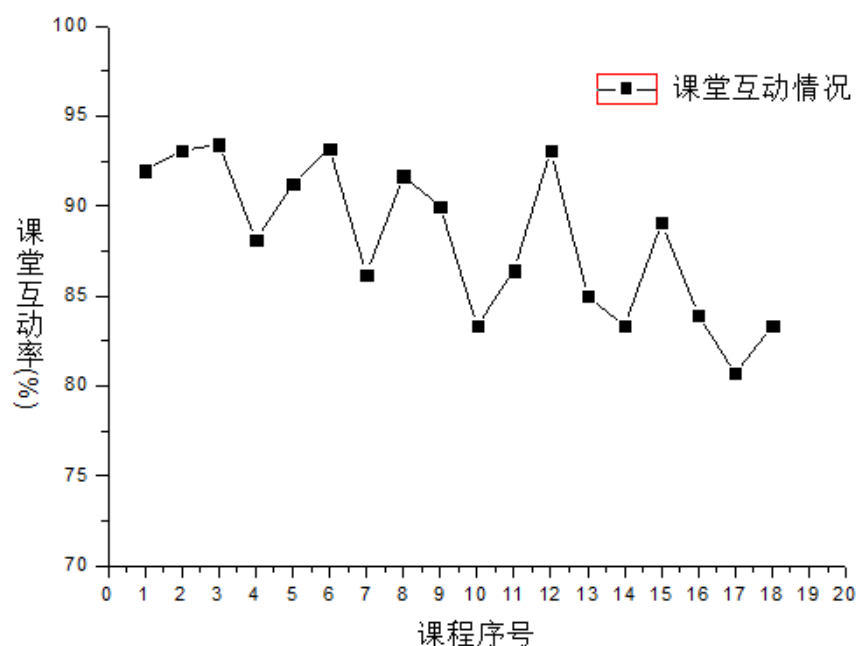
论，培养创新思想。思想碰撞，互学互助，教学相长。讨论使得大家对知识的理解更加深刻，集体力量的强大得到了进一步的体现。

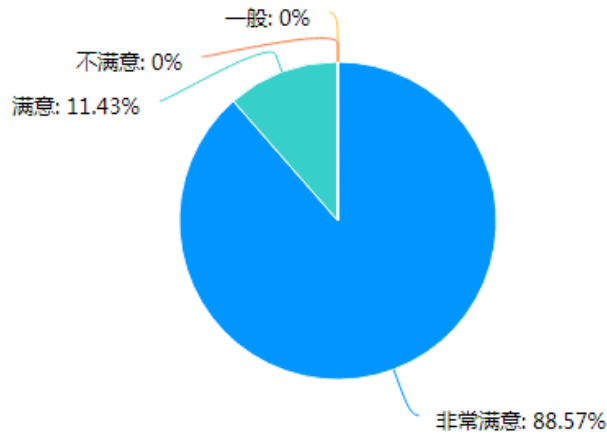
(4) 个性辅导

集中讨论解决的是学生共性问题，但是每个学生对知识的理解程度和掌握程度都不相同，需要个性辅导。每个学生在学习过程中遇到问题，可以随时随地在QQ群或私信给老师进行提问。QQ提问突破了时间空间的限制，学生可以有问即发。QQ群里面的问题，不仅老师回答，其他同学也会根据自己的理解进行回答，相互讨论，相互促进。此外，根据个人情况，学生还可以在“学在西电”平台或者直播平台（雨课堂或腾讯课堂）反复观看录播视频，研究探索加深理解。

(三) 教学过程数据统计分析

本课程学生平均到课率为94%，课堂平均互动率为88%。学生对课程授课方式及内容具有很高的评价（85.71%优秀评价）及满意度（非常满意比例为88.57%）。





大部分学生反馈表示：课程内容安排由易到难，由基础到综合，通过任务的分解和逐层递进的方法激发了学习兴趣和积极性。在任务需求分析过程中，老师引导大家进行辩证分析和信息提炼来设计功能结构，并通过硬件描述语言对 SoC 系统各个功能模块进行编程实现及工程验证，培养了自主探究式学习、思辨性和工程设计思维方式，提高了分析问题和解决问题的能力，以及综合实践动手能力。

第3题：学习完SOC课程后，你的收获及心得体会。

序号	提交答卷时间	答案文本	查看答卷
1	6月7日 13:03	对soc各个功能模块的结构性以及功能更加的了解，同时也对vhdl这门硬件描述语言有了初步的认识，对硬件电路设计更加感兴趣了	查看答卷
2	6月7日 13:22	这门课程让我收获了很多，虽然实验是在线上完成的，但是在老师的认真讲解下，大部分实验内容我都有所掌握，课本上的知识也在实验中得到了验证，虽然没有在学校的实验室动手操作，但是和老师、同学们的线上互动也让我对理论知识更为熟练，总之在老师和同学们的帮助下我在soc这门课程上还是学到了很多实用的知识。	查看答卷
3	6月7日 13:48	对于vhdl语言的运用和逻辑模块的设计有了一定的掌握，更好的学习到了这些设计的思想，拓展了设计的思维	查看答卷
4	6月7日 13:49	学会了vhdl语言，加深了对硬件的理解，同时，课程的实验安排也让我学会并设计了很多cpu部件，获益匪浅。	查看答卷
5	6月7日 14:07	对于SOC课程的学习，我觉得最主要就是掌握了一门的语言同时加深学习了以前的知识，例如计组，微原还有其他的一些课程。虽然实验由于疫情原因没法到校来完成，但是线上的实验完成度也是很好的。	查看答卷
6	6月7日 17:11	刚开始我觉得很不能理解所学的vhdl设计方法，不明白时序的作用是什么，但是在实验中将理论知识用于实践，同时通过老师课堂上的讲解及课后的答疑，我觉得我逐渐掌握了vhdl设计方法。	查看答卷
7	6月7日 18:07	一定程度理解了芯片内各功能模块的作用，以及之间的信号传输，通过熟悉VHDL语言进行软硬件之间的交互。	查看答卷
8	6月7日 18:28	在本门课程的学习过程中，我对VHDL语言的语法有了大方向的了解，并通过多次实验，可以熟练应用VHDL语言编写程序，完成实验，感到很有收获。	查看答卷
9	6月7日 22:50	通过此次课程设计，使我更加扎实的掌握了有关soc方面的知识，在设计过程中虽然遇到了一些问题，但经过一次又一次的思考，一遍又一遍的检查终于找出了原因所在，也暴露出了前期我在这方面的知识欠缺和经验不足。实践出真知，通过亲自动手制作，使我们掌握的知识不再是纸上谈兵。	查看答卷
10	6月7日 22:52	学习了vhdl与相关设计，感觉是与计组和模电知识相结合的 其中计组的那些运算操作都忘记了但通过实验基本都想起来了	查看答卷

13	6月8日 19:28	我了解了SoC的基本概念。从狭义角度讲,它是信息系统核心的芯片集成,是将系统关键部件集成在一块芯片上;从广义角度讲, SoC是一个微小型系统,如果说中央处理器(CPU)是大脑,那么SoC就是包括大脑、心脏、眼睛和手的系统。国内外学术界一般倾向将SoC定义为将微处理器、模拟IP核、数字IP核和存储器(或片外存储控制接口)集成在单一芯片上,它通常是客户定制的,或是面向特定用途的标准产品。SoC技术有很多的特点:半导体工艺技术的系统集成、软件系统和硬件系统的集成、SoC具有很多的优势,因而创造其产品价值与市场需求如:降低耗电量、减少体积、增加系统功能、提高速度、节省成本等	查看答案
14	6月8日 23:03	对我的学习很有帮助,老师很负责,课堂教学活跃,形象生动,虽然只是网上听课,但是老师的教学效果不输于线下教学	查看答案
15	6月8日 23:07	收获良多,老师上课讲解详细,幽默风趣,实验课注重实践,对问题及时解答,使得我学习了这门课之后收获很多,从而增长了知识与经验	查看答案
16	6月8日 23:07	收获很大,感觉进入硬件的设计以后,与软件有很大差别,需要考虑的点很多,但也同时更具体化具象化了,因为有实体电路作为基础。	查看答案
17	6月8日 23:08	通过这次的课程,我学会了数字电路的硬件实现方式,同时对于cpu的体系结构有了更加深入的理解,模块化的概念也更加的清晰。	查看答案
18	6月8日 23:11	对vhdl语言有了初步认识,并且对硬件逻辑设计有了更深的认识。而且对soc的认识从无到有,老师也细心认真。	查看答案

第3题：学习完SOC课程后，你的收获及心得体会。

过滤空选项

?

序号	提交答卷时间	答案文本	查看答案
21	6月8日 23:16	我觉得老师讲的特别好特别仔细,下课督促我们认真写作业,实验课讲的也非常仔细,让我们很快就适应了线上教学。	查看答案
22	6月8日 23:18	我了解了更多关于SOC方面的知识,对于vhdl方面编程知识有了一定的学习,通过对于soc的学习,知道了它对于cpu的作用所在。	查看答案
23	6月8日 23:31	对整个CPU的运作有了深入了解,学习并掌握了基本的vhdl包括如何用vhdl设计硬件,并进行一定的优化	查看答案
24	6月8日 23:32	对CPU的内部结构有更深入的了解,对硬件电路的认知更清晰。这是一门包括了数电计组等知识点的综合课程,能让我在原先知识上学到更多。	查看答案
25	6月8日 23:46	原本在对数电、计组的学习中,很多内容都是老师在纸上谈兵,从来没有想过该怎么实现,有什么用的。在学习了SOC的设计之后,我可以对计算机底层的理解大大加深了。	查看答案
26	6月9日 08:09	SoC课程重点学习了硬件描述语言,基于其的处理器系统构架和优化与测试。最大的收获在于VHDL的了解和适用,一种全新的硬件语言描述方式是需要重点掌握的部分。	查看答案

三、案例展示效果图

(一)“学在西电”课程平台

(1) 课程章节及教学资源

SOC微体系结构设计

西安电子科技大学

[编辑本页](#) [设置](#) 课程打开次数: 15667 课程评价: 0.0 (0人评价)

首页	课程章节	教学资源	教学方法	教学条件	教学效果	参考教材
----	------	------	------	------	------	------

1 SoC设计概论

1.1 SoC设计关键技术

1.2 SoC总线结构

1.3 FPGA结构分析

2 硬件描述语言VHDL设计

2.1 VHDL程序结构

2.2 VHDL基本元素

2.3 VHDL基本逻辑语句

2.4 VHDL描述方式

◀ 上一课 下一课 ▶

本课程在深入分析SoC系统的基本结构、系统研究内容以及关键技术等基础知识的基础上, 讲述基于硬件描述语言的数字系统基本逻辑电路设计方法、基于硬件描述语言的SoC系统中算术运算部件原理及设计方法以及存储器结构及设计方法等内容。重点讨论基于FPGA的SoC系统设计方法以及各个功能部件的设计开发, 包括指令集、程序指针、指令寄存器、算术逻辑单元、通用寄存器、程序存储器、数据存储器、堆栈指针以及通用IO接口等功能模块设计实现。最后探讨SoC测试验证方法以及典型SoC系统和最新技术进展情况。

本课程设计了多个与课程内容相配套的实验。学生需要采用硬件描述语言完成8位SoC系统功能模块设计及综合仿真验证, 并按要求完成基于FPGA的原型验证。

SOC微体系结构设计

西安电子科技大学

[编辑本页](#) [设置](#) 课程打开次数: 15668 课程评价: 0.0 (0人评价)

首页	课程章节	教学资源	教学方法	教学条件	教学效果	参考教材
----	------	------	------	------	------	------

<ul style="list-style-type: none"> 第一讲 soc 设计概论.pdf 2020-02-13 第一章 第2节课.mp4 2020-02-13 1.5 FPGA结构分析.mp4 2020-02-13 第二章 第1节课 VHDL基本介绍.mp4 2020-02-13 第二章 第2节.mp4 2020-02-13 第二章 第4节(1).mp4 2020-02-13 第二章 第5节 组合电路设计.mp4 2020-02-13 第二章 第7节 有限状态机.mp4 2020-02-13 Xilinx_Spartan3E FPGA中文教程.pdf 2020-02-14 第三讲 加法器设计.ppt 2020-02-13 第四讲 乘法器的设计.ppt 2020-02-13 	<ul style="list-style-type: none"> 第一章 第1节课.mp4 2020-02-13 1.5 FPGA结构分析.ppt 2020-02-13 第二讲 硬件描述语言VHDL设计.ppt 2020-02-13 第二章 第1节课 VHDL程序结构.mp4 2020-02-13 第二章 第3节.mp4 2020-02-13 第二章 第4节结构描述.mp4 2020-02-13 第二章 第6节 时序电路设计.mp4 2020-02-13 XILINX_ISE_13.1设计教程.ppt 2020-02-14 第11节课 FPGA开发板及开发软件使用.mp4 2020-02-14 第三章 加法器设计.mp4 2020-02-13 第四章 4.1 常用机器码格式.mp4 2020-02-13
--	---

(2) 录播视频

6.3 CPU设计方法

- ① CPU基本结构
- ② 指令集设计 (功能设计)
- ③ 指令编码设计
- ④ 整体设计思路
- ⑤ 确定指令周期
- ⑥ CPU功能模块设计
- ⑦ CPU模块联合调试测试

Detailed Processor Diagram

2019年05月14日 星期二 09:49:31

6.6.4 指令存储器IR设计

- IR功能分析
 - 传送指令编码到微控制器
 - 生成PC的新地址
 - 生成RAM的读写地址

A-214

Clk_CM \leftarrow W0 & Clk2
 Clk_PC \leftarrow nClk2
 Clk_RAM \leftarrow Clk2 & nClk1
 Clk_IR \leftarrow nClk2
 Clk_RAM \leftarrow nClk2
 Clk_ALU \leftarrow nClk2
 Clk_Mem \leftarrow nClk1 & W1
 Clk_SP \leftarrow W1 & Clk2 & X
 Clk_PO \leftarrow Clk1
 Clk2 \leftarrow nClk2

模块组合

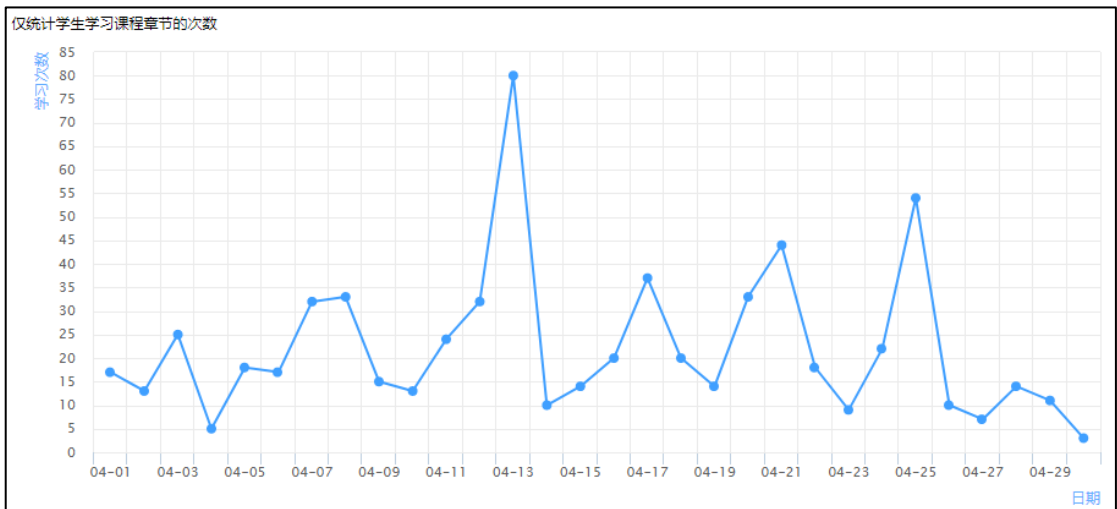
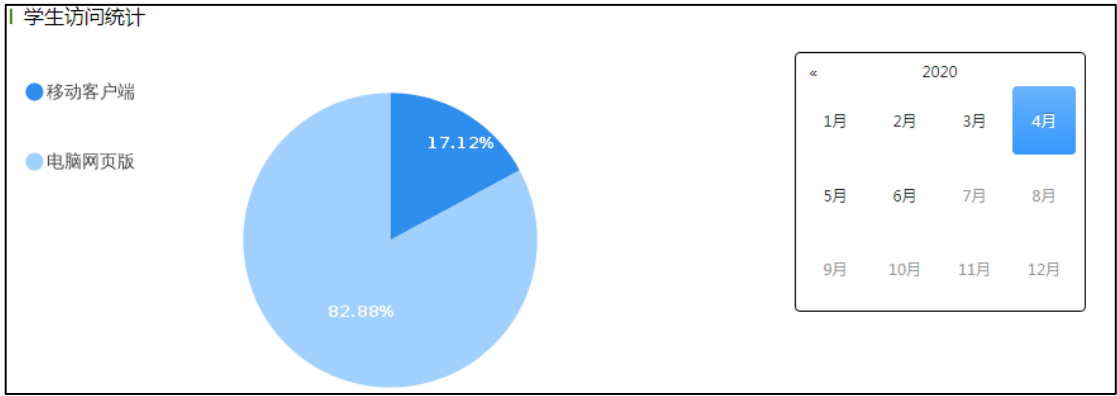
```

SIGNAL a, b, c:STD_LOGIC;
BEGIN
  u1: half_adder PORT MAP (x, y, b, a);
  u2: half_adder PORT MAP (c_in, b, sum, c);
  u3: or_gate PORT MAP (c, a, c_out);
END structural;

```

A-214

(3) 平台 4 月份访问统计



(4) 任务点学习

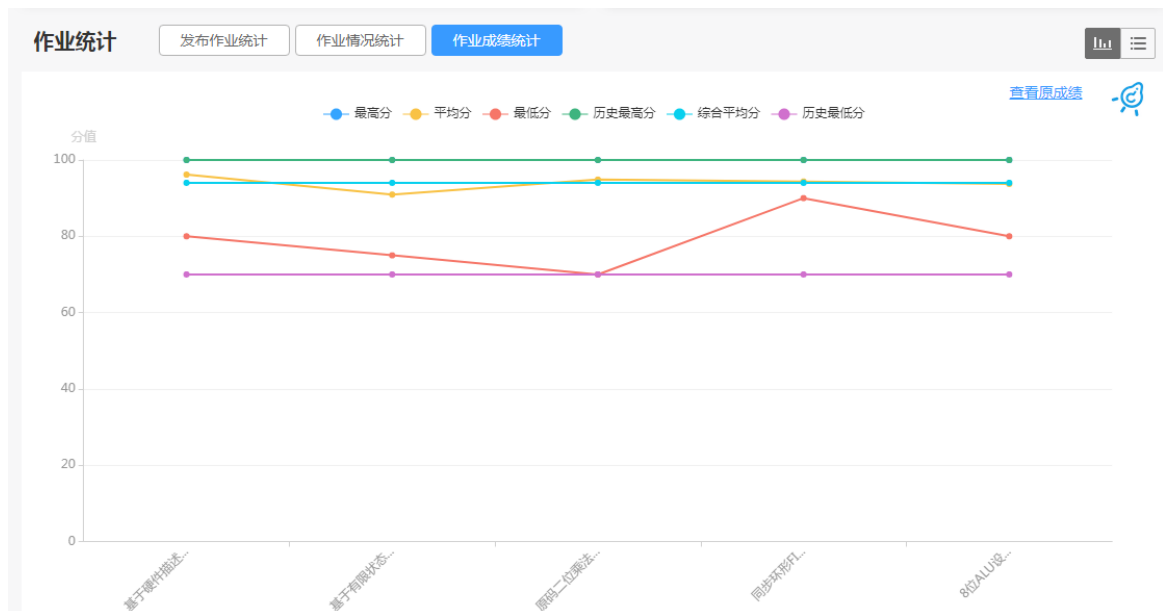
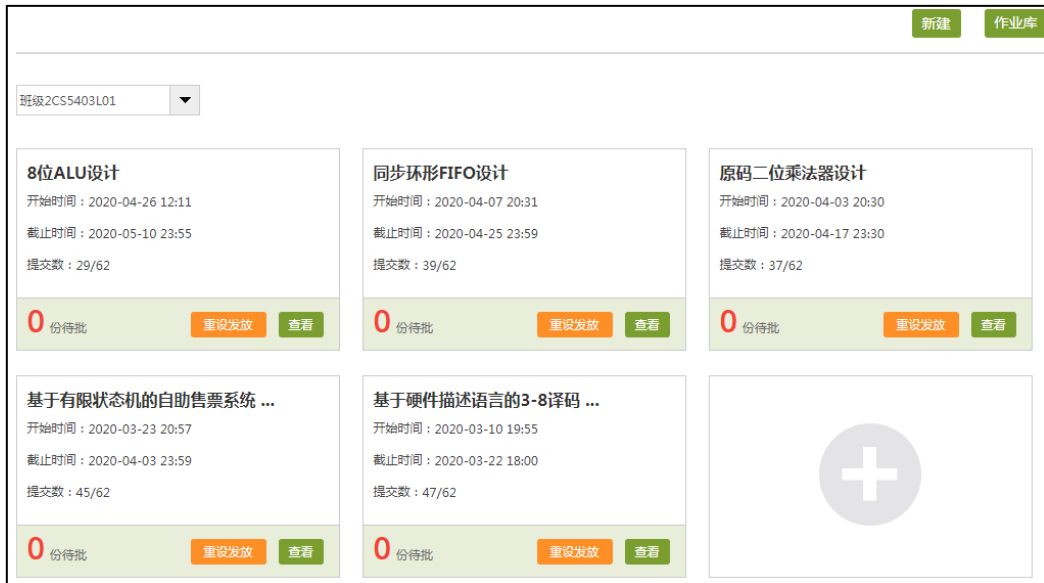
因为课件及视频录像提前分享给了学生，所以在平台上观看视频的学生人数偏少。

班级2CS5403L01 > 任务点 返回

任务点 | 非任务点 选择章节

序号	任务名	类型	说明	学生完成数	详情
1.1、SoC设计关键技术					
任务点1	第一讲 soc 设计概论.pdf	文档		10/62	查看
任务点2	第一章 第1节课.mp4	视频	50.9分钟	4/62	查看
任务点3	第一章 第2节课.mp4	视频	39.7分钟	3/62	查看
1.3、FPGA结构分析					
任务点1	1.5 FPGA结构分析.mp4	视频	43.1分钟	5/62	查看
2.1、VHDL程序结构					
任务点1	第二章 第1节课 VHDL基本介绍.mp4	视频	11.8分钟	6/62	查看
任务点2	第二章 第1节课 VHDL程序结构.mp4	视频	33.9分钟	5/62	查看
2.2、VHDL基本元素					
任务点1	第二章 第2节.mp4	视频	46.6分钟	3/62	查看

(5) 课程作业统计



(二) 在线直播

(1) 雨课堂现场测试



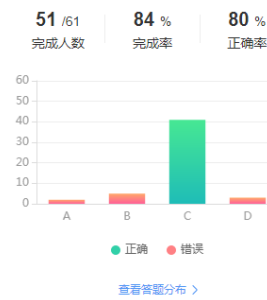


单选题 1分

若7段数码管的编码顺序为gfedcba，共阳极数码管要显示数字“3”的编码是

- A 0101010
- B 0110110
- C 0110000
- D 1010101

西安电子科技大学



单选题 1分

通过外部可以改变电路内部结构和规模的是

- A PORT
- B BUFFER
- C Generic
- D Signal

西安电子科技大学



多选题 1分

本课程所设计的微程序控制器发出的控制信号是多少

- A 16
- B 32
- C 48
- D 64

西安电子科技大学

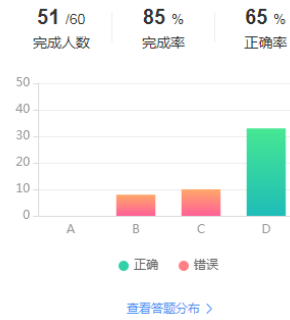


单选题 1分

原码二位乘法器设计中， $-|X|$ 操作需要满足条件

- A $Y_{i+1}=Y_i=C$
- B $Y_{i+1}=0 \ \&\& \ Y_i \oplus C=1$
- C $Y_{i+1} \oplus Y_i=1 \ \&\& \ Y_i=C$
- D $Y_{i+1}=1 \ \&\& \ Y_i \oplus C=1$

西安电子科技大学

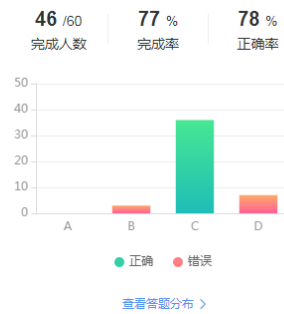


单选题 1分

布斯补码一位乘法运算的符号位应当进行

- A 相与
- B 相或
- C 参与运算
- D 异或

西安电子科技大学



(2) 腾讯课堂录播

腾讯课堂 老师版

张剑贵

首页 > 历史课程

全部考勤记录

课程序号	授课内容	授课时间	授课时长	操作
10	第六讲 6.2 数据通路设计	2020-04-07 16:24	97分钟	考勤 查看 分享
9	第六章 6.1 Richard CPU分析	2020-04-03 16:23	96分钟	考勤 查看 分享
8	第五章 存储器设计	2020-03-31 16:25	100分钟	考勤 查看 分享
7	第四讲 乘除法器设计	2020-03-24 16:22	102分钟	考勤 查看 分享
6	第二讲 2.4 有限状态机及EDA工具使用	2020-03-20 16:25	96分钟	考勤 查看 分享
5	第二讲 2.6 有限状态机设计	2020-03-17 16:27	95分钟	考勤 查看 分享
4	第二讲 第3次课 VHDL描述方式	2020-03-10 16:25	104分钟	考勤 查看 分享

< 1 2 3 >

首页 > 历史课程

全部考勤记录

课程序号	授课内容	授课时间	授课时长	操作
17	第十讲 RISC-V分析	2020-05-19 15:52	101分钟	考勤 查看 分享
16	第九讲 SOC优化设计2	2020-05-12 16:24	104分钟	考勤 查看 分享
15	第九讲 SOC优化设计	2020-05-10 16:23	103分钟	考勤 查看 分享
14	第七讲 SOC测试及技术展望	2020-04-28 16:25	108分钟	考勤 查看 分享
13	第六讲 6.4 微程序控制器时序分析	2020-04-21 16:27	98分钟	考勤 查看 分享
12	第六讲 6.4 微程序控制器设计	2020-04-17 16:25	96分钟	考勤 查看 分享
11	第六讲 6.3 算术逻辑单元ALU设计	2020-04-14 16:24	103分钟	考勤 查看 分享

第六章 6.1 Richard CPU分析 ✓ 已报名

(3) 仿真波形

Signal Timing for Processor

The diagram shows the signal timing for a processor during a memory fetch cycle. Key signals include:

- mem_en**: Memory enable signal.
- pc**: Program Counter value on the bus.
- abus**: Address bus.
- dbus**: Data bus.
- PC value onabus**: Label for the PC signal on the bus.
- enable memory for reading**: Label for the mem_en signal.
- Fetch**: The initial state of the processor.
- Negate Branch**: A control signal.
- mLoad**: Memory load signal.
- add**: ALU operation.
- CM**: Control Memory.
- PC incremented**: Label for the PC signal after the increment.
- ACC loaded**: Label for the ACC signal.
- ireq**: Interrupt request signal.
- acc**: Accumulator register.

Handwritten annotations in red:

- PC → ROM
- ROM → ALU
- ALU → CM

第二讲 2.6 有限状态机设计 ✓ 已报名

(1) D触发器VHDL设计

```

library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
entity d is
    port (cp,din:in std_logic;
          q:out std_logic);
end d;

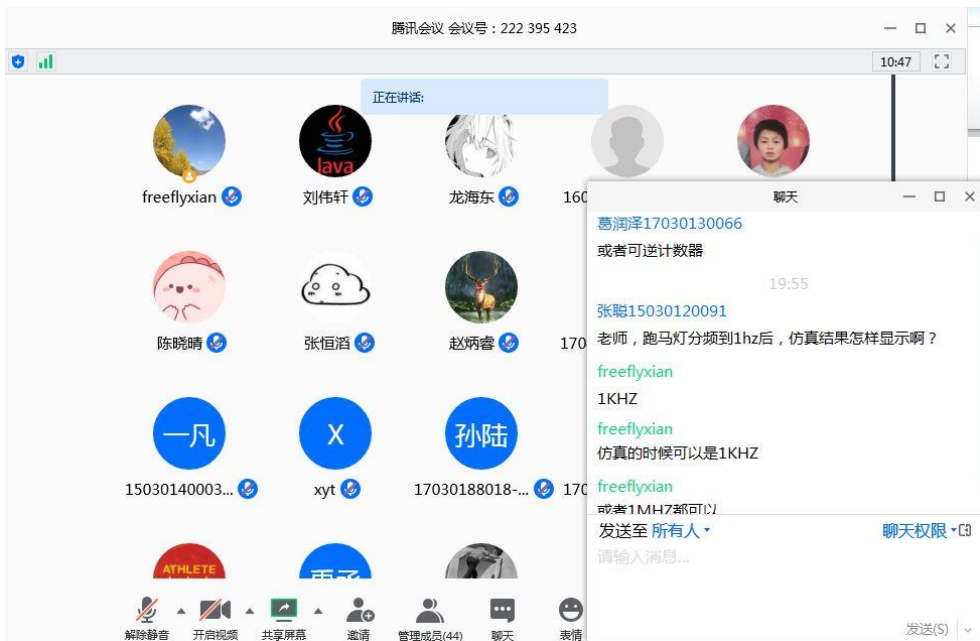
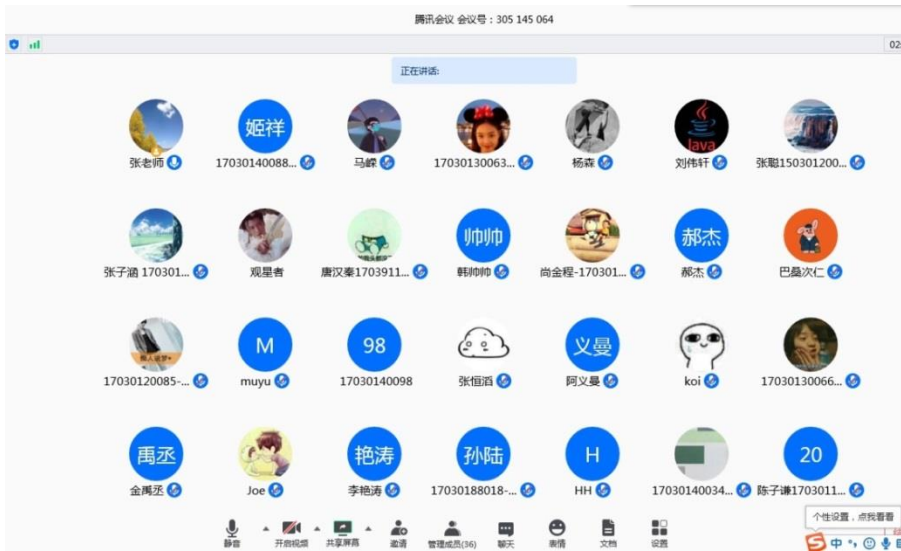
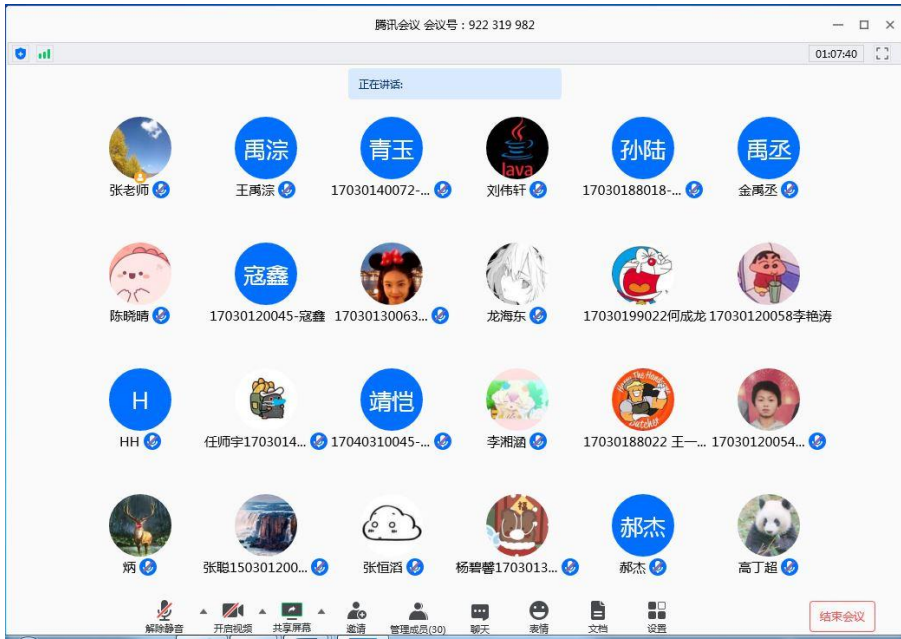
architecture Behavioral of d is
begin
    process(cp)
    begin
        process(cp)
        begin
            if(cp'event and cp='1') then
                q<=din;
            end if;
        end process;
    end process;
end Behavioral;
    
```

The VHDL code defines a D flip-flop. The clock signal (cp) is shown as a square wave. The data input (din) is shown as a signal that changes at various points. The output (q) is shown as a signal that updates only when the clock (cp) transitions from 0 to 1, following the value of din at that time.

第六讲 6.4 微程序控制器时序分析 ✓ 已报名

② 微指令码分析

指令码	位	入口地址	47-46:45:44	43-42:41:40	39-38:37:36	35-34:33:32	31-30:29:28	25-24	21-20:19:18
M_A	M_B	M_F	S3S2S1S0	S4	F1	F0	RAM_CS	RDR1	LDIR1
M_C	M_D	M_E	S5S4S3S2	S6	F2	F3	RAM_CS	RDR2	LDIR2
M_G	M_H	M_I	S7S6S5S4	S8	F4	F5	RAM_CS	RDR3	LDIR3
M_J	M_K	M_L	S9S8S7S6	S10	F6	F7	RAM_CS	RDR4	LDIR4
M_M	M_N	M_O	S11S10S9S8	S12	F8	F9	RAM_CS	RDR5	LDIR5
M_P	M_Q	M_R	S13S12S11S10	S14	F10	F11	RAM_CS	RDR6	LDIR6
M_S	M_T	M_U	S15S14S13S12	S16	F12	F13	RAM_CS	RDR7	LDIR7
M_V	M_W	M_X	S17S16S15S14	S18	F14	F15	RAM_CS	RDR8	LDIR8
M_Y	M_Z	M_0	S19S18S17S16	S20	F16	F17	RAM_CS	RDR9	LDIR9
M_1	M_2	M_3	S21S20S19S18	S22	F18	F19	RAM_CS	RDR10	LDIR10
M_4	M_5	M_6	S23S22S21S20	S24	F20	F21	RAM_CS	RDR11	LDIR11
M_7	M_8	M_9	S25S24S23S22	S26	F22	F23	RAM_CS	RDR12	LDIR12
M_10	M_11	M_12	S27S26S25S24	S28	F24	F25	RAM_CS	RDR13	LDIR13
M_13	M_14	M_15	S29S28S27S26	S30	F26	F27	RAM_CS	RDR14	LDIR14
M_16	M_17	M_18	S31S30S29S28	S32	F28	F29	RAM_CS	RDR15	LDIR15
M_19	M_20	M_21	S33S32S31S30	S34	F30	F31	RAM_CS	RDR16	LDIR16
M_22	M_23	M_24	S35S34S33S32	S36	F32	F33	RAM_CS	RDR17	LDIR17
M_25	M_26	M_27	S37S36S35S34	S38	F34	F35	RAM_CS	RDR18	LDIR18
M_28	M_29	M_30	S39S38S37S36	S40	F36	F37	RAM_CS	RDR19	LDIR19
M_31	M_32	M_33	S41S40S39S38	S42	F38	F39	RAM_CS	RDR20	LDIR20
M_34	M_35	M_36	S43S42S41S40	S44	F40	F41	RAM_CS	RDR21	LDIR21
M_37	M_38	M_39	S45S44S43S42	S46	F42	F43	RAM_CS	RDR22	LDIR22
M_40	M_41	M_42	S47S46S45S44	S48	F44	F45	RAM_CS	RDR23	LDIR23
M_43	M_44	M_45	S49S48S47S46	S50	F46	F47	RAM_CS	RDR24	LDIR24
M_46	M_47	M_48	S51S50S49S48	S52	F48	F49	RAM_CS	RDR25	LDIR25
M_49	M_50	M_51	S53S52S51S50	S54	F50	F51	RAM_CS	RDR26	LDIR26
M_52	M_53	M_54	S55S54S53S52	S56	F52	F53	RAM_CS	RDR27	LDIR27
M_55	M_56	M_57	S57S56S55S54	S58	F54	F55	RAM_CS	RDR28	LDIR28
M_58	M_59	M_60	S59S58S57S56	S60	F56	F57	RAM_CS	RDR29	LDIR29
M_61	M_62	M_63	S61S60S59S58	S62	F58	F59	RAM_CS	RDR30	LDIR30
M_64	M_65	M_66	S63S62S61S60	S64	F60	F61	RAM_CS	RDR31	LDIR31
M_67	M_68	M_69	S65S64S63S62	S66	F62	F63	RAM_CS	RDR32	LDIR32
M_70	M_71	M_72	S67S66S65S64	S68	F64	F65	RAM_CS	RDR33	LDIR33
M_73	M_74	M_75	S69S68S67S66	S70	F66	F67	RAM_CS	RDR34	LDIR34
M_76	M_77	M_78	S71S70S69S68	S72	F68	F69	RAM_CS	RDR35	LDIR35
M_79	M_80	M_81	S73S72S71S70	S74	F70	F71	RAM_CS	RDR36	LDIR36
M_82	M_83	M_84	S75S74S73S72	S76	F72	F73	RAM_CS	RDR37	LDIR37
M_85	M_86	M_87	S77S76S75S74	S78	F74	F75	RAM_CS	RDR38	LDIR38
M_88	M_89	M_90	S79S78S77S76	S80	F76	F77	RAM_CS	RDR39	LDIR39
M_91	M_92	M_93	S81S80S79S78	S82	F78	F79	RAM_CS	RDR40	LDIR40
M_94	M_95	M_96	S83S82S81S80	S84	F80	F81	RAM_CS	RDR41	LDIR41
M_97	M_98	M_99	S85S84S83S82	S86	F82	F83	RAM_CS	RDR42	LDIR42
M_100	M_101	M_102	S87S86S85S84	S88	F84	F85	RAM_CS	RDR43	LDIR43
M_103	M_104	M_105	S89S88S87S86	S90	F86	F87	RAM_CS	RDR44	LDIR44
M_106	M_107	M_108	S91S90S89S88	S92	F88	F89	RAM_CS	RDR45	LDIR45
M_109	M_110	M_111	S93S92S91S90	S94	F90	F91	RAM_CS	RDR46	LDIR46
M_112	M_113	M_114	S95S94S93S92	S96	F92	F93	RAM_CS	RDR47	LDIR47
M_115	M_116	M_117	S97S96S95S94	S98	F94	F95	RAM_CS	RDR48	LDIR48
M_118	M_119	M_120	S99S98S97S96	S100	F96	F97	RAM_CS	RDR49	LDIR49
M_121	M_122	M_123	S101S100S99S98	S102	F98	F99	RAM_CS	RDR50	LDIR50
M_124	M_125	M_126	S103S102S101S100	S104	F100	F101	RAM_CS	RDR51	LDIR51
M_127	M_128	M_129	S105S104S103S102	S106	F102	F103	RAM_CS	RDR52	LDIR52
M_130	M_131	M_132	S107S106S105S104	S108	F104	F105	RAM_CS	RDR53	LDIR53
M_133	M_134	M_135	S109S108S107S106	S110	F106	F107	RAM_CS	RDR54	LDIR54
M_136	M_137	M_138	S111S110S109S108	S112	F108	F109	RAM_CS	RDR55	LDIR55
M_139	M_140	M_141	S113S112S111S110	S114	F110	F111	RAM_CS	RDR56	LDIR56
M_142	M_143	M_144	S115S114S113S112	S116	F112	F113	RAM_CS	RDR57	LDIR57
M_145	M_146	M_147	S117S116S115S114	S118	F114	F115	RAM_CS	RDR58	LDIR58
M_148	M_149	M_150	S119S118S117S116	S120	F116	F117	RAM_CS	RDR59	LDIR59
M_151	M_152	M_153	S121S120S119S118	S122	F118	F119	RAM_CS	RDR60	LDIR60
M_154	M_155	M_156	S123S122S121S120	S124	F120	F121	RAM_CS	RDR61	LDIR61
M_157	M_158	M_159	S125S124S123S122	S126	F122	F123	RAM_CS	RDR62	LDIR62
M_160	M_161	M_162	S127S126S125S124	S128	F124	F125	RAM_CS	RDR63	LDIR63
M_163	M_164	M_165	S129S128S127S126	S130	F126	F127	RAM_CS	RDR64	LDIR64
M_166	M_167	M_168	S131S130S129S128	S132	F128	F129	RAM_CS	RDR65	LDIR65
M_169	M_170	M_171	S133S132S131S130	S134	F130	F131	RAM_CS	RDR66	LDIR66
M_172	M_173	M_174	S135S134S133S132	S136	F132	F133	RAM_CS	RDR67	LDIR67
M_175	M_176	M_177	S137S136S135S134	S138	F134	F135	RAM_CS	RDR68	LDIR68
M_178	M_179	M_180	S139S138S137S136	S140	F136	F137	RAM_CS	RDR69	LDIR69
M_181	M_182	M_183	S141S140S139S138	S142	F138	F139	RAM_CS	RDR70	LDIR70
M_184	M_185	M_186	S143S142S141S140	S144	F140	F141	RAM_CS	RDR71	LDIR71
M_187	M_188	M_189	S145S144S143S142	S146	F142	F143	RAM_CS	RDR72	LDIR72
M_190	M_191	M_192	S147S146S145S144	S148	F144	F145	RAM_CS	RDR73	LDIR73
M_193	M_194	M_195	S149S148S147S146	S150	F146	F147	RAM_CS	RDR74	LDIR74
M_196	M_197	M_198	S151S150S149S148	S152	F148	F149	RAM_CS	RDR75	LDIR75
M_199	M_200	M_201	S153S152S151S150	S154	F150	F151	RAM_CS	RDR76	LDIR76
M_202	M_203	M_204	S155S154S153S152	S156	F152	F153	RAM_CS	RDR77	LDIR77
M_205	M_206	M_207	S157S156S155S154	S158	F154	F155	RAM_CS	RDR78	LDIR78
M_208	M_209	M_210	S159S158S157S156	S160	F156	F157	RAM_CS	RDR79	LDIR79
M_211	M_212	M_213	S161S160S159S158	S162	F158	F159	RAM_CS	RDR80	LDIR80
M_214	M_215	M_216	S163S162S161S160	S164	F160	F161	RAM_CS	RDR81	LDIR81
M_217	M_218	M_219	S165S164S163S162	S166	F162	F163	RAM_CS	RDR82	LDIR82
M_220	M_221	M_222	S167S166S165S164	S168	F164	F165	RAM_CS	RDR83	LDIR83
M_223	M_224	M_225	S169S168S167S166	S170	F166	F167	RAM_CS	RDR84	LDIR84
M_226	M_227	M_228	S171S170S169S168	S172	F168	F169	RAM_CS	RDR85	LDIR85
M_229	M_230	M_231	S173S172S171S170	S174	F170	F171	RAM_CS	RDR86	LDIR86
M_232	M_233	M_234	S175S174S173S172	S176	F172	F173	RAM_CS	RDR87	LDIR87
M_235	M_236	M_237	S177S176S175S174	S178	F174	F175	RAM_CS	RDR88	LDIR88
M_238	M_239	M_240	S179S178S177S176	S180	F176	F177	RAM_CS	RDR89	LDIR89
M_241	M_242	M_243	S181S180S179S178	S182	F178	F179	RAM_CS	RDR90	LDIR90
M_244	M_245	M_246	S183S182S181S180	S184	F180	F181	RAM_CS	RDR91	LDIR91
M_247	M_248	M_249	S185S184S183S182	S186	F182	F183	RAM_CS	RDR92	LDIR92
M_250	M_251	M_252	S187S186S185S184	S188	F184	F185	RAM_CS	RDR93	LDIR93
M_253	M_254	M_255	S189S188S187S186	S190	F186	F187	RAM_CS	RDR94	LDIR94
M_256	M_257	M_258	S191S190S189S188	S192	F188	F189	RAM_CS	RDR95	LDIR95
M_259	M_260	M_261	S193S192S191S190	S194	F190	F191	RAM_CS	RDR96	LDIR96
M_262	M_263	M_264	S195S194S193S192	S196	F192	F193	RAM_CS	RDR97	LDIR97
M_265	M_266	M_267	S197S196S195S194	S198	F194	F195	RAM_CS	RDR98	LDIR98
M_268	M_269	M_270	S199S198S197S196	S200	F196	F197	RAM_CS	RDR99	LDIR99
M_271	M_272	M_273	S201S200S199S198	S202	F198	F199	RAM_CS	RDR100	LDIR100
M_274	M_275	M_276	S203S202S201S200	S204	F200	F201	RAM_CS	RDR101	LDIR101
M_277	M_278	M_279	S205S204S203S202	S206	F202	F203	RAM_CS	RDR102	LDIR102
M_280	M_281	M_282	S207S206S205S204	S208	F204	F205	RAM_CS	RDR103	LDIR103
M_283	M_284	M_285	S209S208S207S206	S210	F206	F207	RAM_CS	RDR104	LDIR104
M_286	M_287	M_288	S211S210S209S208	S212	F208	F209	RAM_CS	RDR105	LDIR105
M_289	M_290	M_291	S213S212S211S210	S214	F210	F211	RAM_CS	RDR106	LDIR106
M_292	M_293	M_294	S215S214S213S212	S216	F212	F213	RAM_CS	RDR107	LDIR107
M_295	M_296	M_297	S217S216S215S214	S218	F214	F215	RAM_CS	RDR108	LDIR108
M_298	M_299	M_300	S219S218S217S216	S220	F216	F217	RAM_CS	RDR109	LDIR109
M_301	M_302	M_303	S221S220S219S218	S222	F218	F219	RAM_CS	RDR110	LDIR110
M_304	M_305	M_306	S223S222S221S220	S224	F220	F221	RAM_CS	RDR111	LDIR111
M_307	M_308	M_309	S225S224S223S222	S226	F222	F223	RAM_CS	RDR112	LDIR112
M_310	M_311	M_312	S227S226S225S224	S228	F224	F225	RAM_CS	RDR113	LDIR113
M_313	M_314	M_315	S229S228S227S226	S2					



(四) QQ群个性辅导

(1) QQ群师生讨论

2020年SOC微体系结构设计

聊天 公告 相册 文件 作业 设置

【班主任】SOC微体系结构设计-张老师(121583182) 2020/4/28 星期二 23:35:12

原码符号位8位

【匿名】郭奕 2020/5/2 星期六 16:26:48

问一下在着手写ALU的同学们，你们的算术运算有考虑加入乘除法的功能吗？

感觉要加入乘除法模块的话，工作量会爆增啊。

【班主任】SOC微体系结构设计-张老师(121583182) 2020/5/2 星期六 16:30:54

ALU模块不需要加入乘除法模块，按照课件的要求做就可以了

【匿名】雄蛇 2020/5/4 星期一 22:22:44

老师，输入到alu中的数据是以原码形式输入，还是以补码形式输入？

【班主任】SOC微体系结构设计-张老师(121583182) 2020/5/4 星期一 23:26:08

按照补码形式输入，原则上是可以输入正负数的。但是其实Alu是不知道现在操作的是正数还是负数。它只是将相应的数据根据指令要求进行运算，然后输出结果，并设置相应的状态信息。

2020年SOC微体系结构设计

聊天 公告 相册 文件 作业 设置

张聪(592847566) 2020/4/11 星期六 21:27:03

乘法器设计那个加法进位应该怎样处理啊？求助

17030120010谭雯丹(936997839) 2020/4/11 星期六 21:36:10

直接做成10加法 肯定不会有进位

张聪(592847566) 2020/4/11 星期六 21:40:04

那就和其他模块匹配不上了啊

17030120010谭雯丹(936997839) 2020/4/11 星期六 21:46:01

emmm输入可以不是10位的吧 高位补上符号位凑成10位

输出的话不是正好放到移位寄存器里面吗

张聪(592847566) 2020/4/11 星期六 21:51:01

- 我知道了
- 是我看错了，16位寄存器的输入是9位，正好加上进位

2020年SOC微体系结构设计

聊天 公告 相册 文件 作业 设置

【匿名】廉颇 2020/5/17 星期日 18:51:08

咱们ppt上原码一位乘法器的锁存器移位为什么部分积不右移呀？这里锁存器中乘数右移一次，在下一次移位寄存器中乘数第二次右移，是不是重复了？

张聪(592847566) 2020/5/17 星期日 18:56:29

没有重复D的输入是带进位的，相当于右移了一次，而上面只右移了低八位

【匿名】廉颇 2020/5/17 星期日 20:00:15

谢谢小姐姐！但是代码中为什么只右移了部分积的最后一位，如果部分积不整体右移，后续每次和16位锁存器高九位里存的部分积相加不是就乱了吗

【匿名】廉颇 2020/5/17 星期日 20:01:45

等等我好像明白了什么

【匿名】廉颇 2020/5/17 星期日 20:01:49

我再瞅瞅

【匿名】廉颇 2020/5/17 星期日 20:04:16

啊 我明白了 我一直以为这个锁存器模块里存的是部分积+乘数，原来只存了部分积，只不过输出了16位，另一个移位寄存器才是乘数右移的部分

2020年SOC微体系结构设计

聊天 公告 相册 文件 作业 设置

【匿名】梅花 2020/5/8 星期五 16:56:27

怎么才能正确的用inout口给别的信号赋值啊

【匿名】梅花 2020/5/8 星期五 16:56:27

但就是哪里不对，每次调试看的data都是u

17030140034 孟新宇<lhqeng@vip.qq.com> 2020/5/8 星期五 16:57:50

也许是逻辑控制部分？

【匿名】梅花 2020/5/8 星期五 16:59:40

我先把DATA赋值给data_in，但是我在调试的时候看data_in也是u

【匿名】梅花 2020/5/8 星期五 17:00:36

```
yiweishuchu:process(clk)
begin
if rising_edge(clk) then
if Alu_EN='1' then
case F is
when '00'=>DATA<=res;
when '10'=>DATA<=res(0)&res(7 downto 1);
when '11'=>DATA<=res(6 downto 0)&res(7);
when others=>null;
end case;
else
DATA<="ZZZZZZZZZ";
end if;
end process yiweishuchu;
```

【匿名】含羞草 2020/5/8 星期五 17:28:05

你要保证在输出数据占用总线时，你给的datain的输入要是高阻

聊天 公告 相册 文件 作业 设置

这个移位应该是移了你的第一位置成1应该就看得出来了

【匿名】海桐 2020/5/19 星期二 11:52:13

变量是不能综合的吧

【匿名】沙棘 2020/5/19 星期二 11:52:22

d是仿真的信号f0，感觉就是加上了信号赋值之后，移位就被屏蔽了

【匿名】沙棘 2020/5/19 星期二 11:52:52

对单纯锁存器仿真真是直接用输入

【匿名】海桐 2020/5/19 星期二 11:53:04

没屏蔽 d仿真信号你改成f1就看得出来了

【匿名】沙棘 2020/5/19 星期二 11:53:46

好 那我再去看下

【匿名】沙棘 2020/5/19 星期二 12:05:45



换了个仿真输入信号电光石火之间就想清楚了，输入信号设置得比较特别，只右移四次f0是看不出效果的，相当于后面8个0不断右移，一开始去掉D赋值由于没有后续的值覆盖上去，所以才能看到的移动效果

聊天 公告 相册 文件 作业 设置

老夫子, 启动老师

【匿名】老夫子 2020/5/30 星期六 19:24:36

老师，那jmp后的pc地址是谁传给他的

【匿名】兰陵王 2020/5/30 星期六 19:24:59

应该是pc指针？

【班主任】SOC微体系结构设计-张老师(121583182) 2020/5/30 星期六 19:26:32

读出JMP指令后，译码后微控制器发出读取地址的控制信号，地址存在ROM里面，地址送到IR，然后由IR发给PC

【匿名】兰陵王 2020/5/30 星期六 19:32:41

老师，在pc加一控制信号有效时，第一次输出的addr是不是应该是0，不是1？

【班主任】SOC微体系结构设计-张老师(121583182) 2020/5/30 星期六 19:35:02

PC+1控制有效，输出就是1啊

GIF 剪贴板 文件夹 分享 通知 更多

(2) “一对一” 个性辅导

17030110015-朱生辉(9627..

17030110015-朱生辉(9627..

madao 2020/6/7 星期日 19:10:22

老师，我想问一下就是我在alu的testbench里面结构体给data赋初始值条件是当总线向A寄存器写数据的条件

madao 2020/6/7 星期日 19:11:48

但是当满足条件后，data在赋给A寄存器是值是uu

```

process
begin
wait for 10 ns;
M_A <= '1';
M_B <= '0';
nALU_EN<='1';
nPSW_EN<='0';
M_F <= '0';
wait for 20 ns;
M_A <= '0';
clk_ALU_process:process
begin
clk_ALU<='0';
wait for 10 ns;
clk_ALU<='1';
wait for 10 ns;
end process clk_ALU_process;

```

赋值条件不应当放到testbench里面，按照操作步骤进行赋值操作就可以了。

M_A有效时间不对，在时钟上升沿来的时候，才变成高电平，应当提前有效

madao 2020/6/7 星期日 21:35:08

老师，那对data的赋值应该放在哪里？

M_A就是在clk上升沿的时候才变成高电平

我一开始wait for 10 ns

蓝枫 2020/6/7 星期日 21:55:12

你没有用复位信号啊，一般复位一个时钟周期，然后开始将各种控制信号陆续有效。因为当激励信号进入到被测试ALU模块，理论上是立即到达。但是实际上不一定，所以在时钟到达的时候，不一定能检测到M_A信号

17030130066葛润泽(18508...)

24H进入到微程序的入口地址，24H的微指令就是取ROM里面的立即数

止痛药 2020/4/21 星期二 18:11:26

啾啾

止痛药 2020/4/21 星期二 18:14:16

老师24H,25H,26H每执行一条，pc都要加1么

蓝枫 2020/4/21 星期二 18:22:34

不是啊，PC加1操作都统一在取指令操作里面，取出一条指令后，PC+1. 24H,25,26H是取立即数这条指令具体执行过程，PC不加1

止痛药 2020/4/21 星期二 18:24:43

- 了解
- 取指令操作得到的指令会包含程序的入口对嘛

蓝枫 2020/4/21 星期二 18:25:37

对的

17030130066葛润泽(18508...)

止痛药 2020/4/8 星期三 21:03:20

老师我想问一下如果一个8位的输入，我只给了7位，那剩下一位默认补0的嘛

止痛药 2020/4/8 星期三 21:12:07

还是端口都要对齐的

蓝枫 2020/4/8 星期三 21:18:27

输入信号没有给输入值的话，可能是随机值（可能是1也可能是0）最好给确定的电平输入

止痛药 2020/4/20 星期一 22:57:49

老师我想问一下

止痛药 2020/4/20 星期一 22:58:16

- fifo里读写指针为什么设置成 depth -1呢
- 如果深度为8
- 那7-0的向量组可以表示道256了

蓝枫 2020/4/20 星期一 23:13:07

设计成depth的话，可以通过顶层调用时修改Fifo深度。相当于用户可修改配置Fifo深度参数

学生柏志伟

学生柏志伟 2020/3/25 星期三 22:30:09

- 老师，这个设计不使用时钟信号可以吗

学生柏志伟 2020/3/25 星期三 22:32:01

我弄了半天，感觉要是用时钟信号来控制状态转移的那个process的话，投入的钱的高电平持续时间我没法确定 就是说可能这个人只投入了5块钱，但时钟太快了，5块钱对应的输入信号还是高电平还没变低下一个时钟又来了，就相当于这个人投了10块钱

蓝枫 2020/3/25 星期三 23:00:51

- 不错，很多同学没有考虑到这个问题的。可以增加投入输入的处理电路来解决这个问题

自动售货机控制...实现.pdf (244.09KB)

成功发送离线文件，文件助手暂存7天

打开 打开文件夹 转发

对方已成功接收了你发送的离线文件“自动售货机控制系统VHDL有限状态机实现.pdf” (244.09KB)。

17030140034 孟新宇<lhge...>

- 老师，这一段代码是什么意思呢？我理解，四组里每一组的一个数码管拼起来显示的才是真实输入值？

该用户通过“2020年SOC微体系结构设计”群向你发起临时会话，前往设置。

蓝枫 2020/5/11 星期一 10:53:06

我们板上总共16个数码管，分为4组，1组有4位数码管，每个数码管可以显示0~F。比如你要显示“1234”，每个数字需要4bit。

上面的代码是通过拨动开关来设置，数码管显示的数据

蓝枫 2020/5/11 星期一 10:55:31

这个代码用4bit开关的数据，然后加上一个数，就是简化了输入端的操作而已

17030120058李艳涛(10153...)

临时会话，前往设置。

十七 2020/5/8 星期五 09:34:41

- 老师，就是黄线对应的那一刻

蓝枫 2020/5/8 星期五 10:40:02

在时钟上升沿的时候，才会发生变化，在F9为02a的后，只有遇到上升沿，才会把结果输出到output

十七 2020/5/8 星期五 10:54:11

老师~我把 计算得到结果f9的操作 和 f9移位后送入output的操作 是写在一个进程里的，这两个操作不是在时钟上升沿同时发生得到结果吗

蓝枫 2020/5/8 星期五 10:56:56

不是同时的，output，f9是在进程结束的时候才更新，也就是说在同一个时钟上升沿output得到的是前一个f9，不是当前新的数据

17030130052-王禹淙(7450...)

- 初步发现有个复位信号没有初始化，被综合误认为无效了，现在一些模块可以综合出来了，但还是有逻辑丢失
- 主要是仿真正确的，这点我不通

蓝枫 2020/4/7 星期二 15:59:34

- verilog 内部要使用寄存器，而不能光是使用wire类型
- 有些参数只能行为级仿真使用，不能综合的

Shepard 2020/4/7 星期二 16:00:19

是的，wire不可以赋值

蓝枫 2020/4/7 星期二 16:00:34

你查下verilog哪些情况下不能被综合

Shepard 2020/4/7 星期二 16:00:41

- 好的，长知识了
- fpga调试感觉比较教人😂

蓝枫 2020/4/7 星期二 16:04:55

FPGA调试就是比较麻烦，不仅要考虑功能的正确性，还要保证时序有效性