

21世纪高等学校计算机**专业**实用规划教材

计算机组成原理 实践教程

C

肖铁军 丁伟 赵蕙 马学文 编著

O

P

清华大学出版社



21世纪高等学校计算机**专业**实用规划教材

计算机组成原理 实践教程

肖铁军 丁伟 赵蕙 马学文 编著

清华大学出版社

内 容 简 介

本书以教学模型计算机的设计为主线,组织了10个实验和2个课程设计。第1章介绍 Verilog HDL 硬件描述语言的基础知识;第2章介绍16位微程序控制计算机的设计;第3章是计算机部件的实验指导;第4章给出了两个课程设计方案,分别是微程序设计和CPU设计;第5章介绍FPGA设计工具和实验系统。

本书主要作为高等院校计算机专业“计算机组成原理”课程的实践教材,也可作为自学CPU设计的参考书。

本书封面贴有清华大学出版社防伪标签,无标签者不得销售。

版权所有,侵权必究。侵权举报电话:010-62782989 13701121933

图书在版编目(CIP)数据

计算机组成原理实践教程/肖铁军等编著. —北京:清华大学出版社,2015

21世纪高等学校计算机专业实用规划教材

ISBN 978-7-302-38280-5

I. ①计… II. ①肖… III. ①计算机组成原理—高等学校—教材 IV. ①TP301

中国版本图书馆CIP数据核字(2015)第002814号

责任编辑:黄 芝 薛 阳

封面设计:傅瑞学

责任校对:梁 毅

责任印制:杨 艳

出版发行:清华大学出版社

网 址: <http://www.tup.com.cn>, <http://www.wqbook.com>

地 址:北京清华大学学研大厦A座 邮 编:100084

社总机:010-62770175 邮 购:010-62786544

投稿与读者服务:010-62776969, c-service@tup.tsinghua.edu.cn

质 量 反 馈:010-62772015, zhiliang@tup.tsinghua.edu.cn

课 件 下 载: <http://www.tup.com.cn>, 010-62795954

印 装 者:北京嘉实印刷有限公司

经 销:全国新华书店

开 本:185mm×260mm

印 张:13.75

字 数:345千字

版 次:2015年2月第1版

印 次:2015年2月第1次印刷

印 数:1~2000

定 价:29.00元

产品编号:060893-01

出版说明

随着我国改革开放的进一步深化,高等教育也得到了快速发展,各地高校紧密结合地方经济建设发展需要,科学运用市场调节机制,加大了使用信息科学等现代科学技术提升、改造传统学科专业的投入力度,通过教育改革合理调整和配置了教育资源,优化了传统学科专业,积极为地方经济建设输送人才,为我国经济社会的快速、健康和可持续发展以及高等教育自身的改革发展做出了巨大贡献。但是,高等教育质量还需要进一步提高以适应经济社会发展的需要,不少高校的专业设置和结构不尽合理,教师队伍整体素质亟待提高,人才培养模式、教学内容和方法需要进一步转变,学生的实践能力和创新精神亟待加强。

教育部一直十分重视高等教育质量工作。2007年1月,教育部下发了《关于实施高等学校本科教学质量与教学改革工程的意见》,计划实施“高等学校本科教学质量与教学改革工程(简称‘质量工程’)”,通过专业结构调整、课程教材建设、实践教学改革、教学团队建设等多项内容,进一步深化高等学校教学改革,提高人才培养的能力和水平,更好地满足经济社会发展对高素质人才的需要。在贯彻和落实教育部“质量工程”的过程中,各地高校发挥师资力量强、办学经验丰富、教学资源充裕等优势,对其特色专业及特色课程(群)加以规划、整理和总结,更新教学内容、改革课程体系,建设了一大批内容新、体系新、方法新、手段新的特色课程。在此基础上,经教育部相关教学指导委员会专家的指导和建议,清华大学出版社在多个领域精选各高校的特色课程,分别规划出版系列教材,以配合“质量工程”的实施,满足各高校教学质量和教学改革的需要。

本系列教材立足于计算机专业课程领域,以专业基础课为主、专业课为辅,横向满足高校多层次教学的需要。在规划过程中体现了如下一些基本原则和特点。

(1) 反映计算机学科的最新发展,总结近年来计算机专业教学的最新成果。内容先进,充分吸收国外先进成果和理念。

(2) 反映教学需要,促进教学发展。教材要适应多样化的教学需要,正确把握教学内容和课程体系的改革方向,融合先进的教学思想、方法和手段,体现科学性、先进性和系统性,强调对学生实践能力的培养,为学生知识、能力、素质协调发展创造条件。

(3) 实施精品战略,突出重点,保证质量。规划教材把重点放在公共基础课和专业基础课的教材建设上;特别注意选择并安排一部分原来基础比较好的优秀教材或讲义修订再版,逐步形成精品教材;提倡并鼓励编写体现教学质量和教学改革成果的教材。

(4) 主张一纲多本,合理配套。专业基础课和专业课教材配套,同一门课程有针对不同层次、面向不同应用的多本具有各自内容特点的教材。处理好教材统一性与多样化,基本教材与辅助教材、教学参考书,文字教材与软件教材的关系,实现教材系列资源配套。

(5) 依靠专家,择优选用。在制定教材规划时要依靠各课程专家在调查研究本课程教

材建设现状的基础上提出规划选题。在落实主编人选时,要引入竞争机制,通过申报、评审确定主题。书稿完成后要认真实行审稿程序,确保出书质量。

繁荣教材出版事业,提高教材质量的关键是教师。建立一支高水平教材编写梯队才能保证教材的编写质量和建设力度,希望有志于教材建设的教师能够加入到我们的编写队伍中来。

21 世纪高等学校计算机专业实用规划教材

联系人:魏江江 weijj@tup.tsinghua.edu.cn

前 言

“计算机组成原理”作为计算机学科的一门专业基础课程,介绍计算机的基本结构、基本原理和基本分析方法。它不以某种商业机型为蓝本,侧重于基础性和一般性,内容比较抽象。作者通过长期的“计算机组成原理”理论教学和实践教学,提出了“从设计的角度理解计算机的组成和工作原理”的教学理念,强调实践教学的重要性,对实验内容、实验手段、实验教学方法进行了一系列的改革,结合课程的重点和难点,设计了伴随理论课进行的实验项目;在理论课结束之后的课程设计项目中,通过模型 CPU 的设计,深入理解计算机的组成和工作原理。开发的实验系统对实践教学的实施效果起到了很好的帮助作用。

第 1 章是 Verilog 硬件描述语言的入门教程。本书实验的设计输入完全采用 Verilog HDL 来描述,没有采用原理图的输入方法,目的是使学生更多地接触现代数字系统的工程设计手段。Verilog HDL 的语法与 C 语言相似,本书假定读者具有 C 语言基础,因此不对 Verilog 语法作详细介绍,而是通过一些实例介绍基本的组合逻辑和时序逻辑的描述方法。例题尽可能地采用 Verilog—2001 的新的改进特性,使学生从入门开始接触的就是被大多数设计软件支持的主流 Verilog 版本。

第 2 章完整地介绍了一个教学模型计算机的设计。该模型机字长为 16 位,具有 38 条常用指令、8 种基本寻址方式,采用微程序控制方式,主存寻址空间为 64K 字,采用向量中断机制,并且内置调试器以支持实验系统。该模型机已经设计实现,能够在 FPGA 上以 10MHz 的主频运行。本章介绍了模型机的指令系统设计,运算器、控制器、基本输入输出接口等硬件设计,以及微程序设计方法和部分微程序实例。

第 3 章设计了 10 个实验项目,涵盖了计算机组成原理课程的核心内容。实验的过程是一个从设计到验证的过程。所有的实验均采用 Verilog HDL 进行逻辑设计,在 FPGA 上实现。考虑到课程性质和教学要求,在硬件设计上并不要求训练学生达到熟练的设计水平,而是通过示范和有限的设计引导学生深入理解硬件结构。在验证环节,通过预先设计的步骤和记录表引导学生发现其中的原理问题,例如在加减运算实验中,通过记录表反映出既可以将运算数看作无符号数,也可以看作有符号补码,引导学生发现加减运算电路的本质;每个实验的操作和记录之后还设计了一些填空和问题,启发学生运用课堂上学习的理论知识对实验现象和结果进行解释,理论和实践相结合,从而加深对计算机部件工作原理的理解。

第 4 章包含两个课程设计项目,分别是微控制器的微程序设计和 CPU 设计,满足不同的教学要求。微程序设计项目针对第 2 章所述的教学模型机,设计控制器的微程序,实现该模型机的指令系统。CPU 设计项目在第 3 章部件实验的基础上从一个仅支持有限指令的初级 CPU 开始,不断扩充硬件以及微程序,由浅入深、循序渐进地设计实现完整的 CPU,使学生深入理解硬件与微程序以及指令系统的联系,进而加深对计算机的结构和工作原理的

理解。课程设计采用任务驱动的教学机制,将整个设计分解成若干个环环相扣的子任务,既便于教学实施,也有利于调动学生的主动性。

第5章设计工具与实验环境介绍了FPGA设计软件和实验系统。FPGA设计软件介绍了Altera的Quartus II和Xilinx的ISE两种主流的FPGA设计工具,本书的实验和课程设计项目在Altera和Xilinx的FPGA上均可以实现。相应地介绍了两种FPGA开发板,Altera/Terasic DE2-115教学开发板和Xilinx/Digilent Nexys3开发板。最后介绍了作者开发的实验系统,该实验系统基于JTAG技术,具有很好的适应性,能够与现有的Altera或Xilinx的FPGA开发板配套,只要该开发板有4个可用的FPGA引脚引出即可;所有的实验都可以通过实验系统软件的虚拟实验板操作,甚至不需要实验板有实际的开关、指示灯等元件。实验软件还具有虚拟面板的定制功能,教师和学生可以设计自己的实验项目,并定制图形化的虚拟面板,而不仅限于本书设计的实验项目。

本书实验和课程设计的相关材料可以向教师开放,需要者请向作者索取。书中错误、不当之处,敬请读者批评指正。作者联系邮箱:fpgalab@qq.com。

作者

2014年6月

目 录

第 1 章 Verilog HDL 快速入门	1
1.1 Verilog HDL 概述	1
1.2 Verilog HDL 语法概要	2
1.2.1 数据类型及数的表示	2
1.2.2 运算符	5
1.2.3 模块	7
1.2.4 编译指示字	7
1.3 组合逻辑的 Verilog HDL 描述	8
1.3.1 用 assign 持续赋值语句描述组合逻辑	8
1.3.2 用 always 过程语句描述组合逻辑	10
1.4 时序逻辑的 Verilog HDL 描述	12
1.4.1 触发器	12
1.4.2 同步复位和异步复位	13
1.4.3 门控时钟和时钟使能	14
1.4.4 数据寄存器	15
1.4.5 计数器和移位寄存器	16
1.4.6 锁存器	17
1.4.7 存储器	18
1.4.8 阻塞赋值和非阻塞赋值	18
1.5 层次化和参数化设计	19
1.5.1 层次化设计	19
1.5.2 参数化设计	20
1.5.3 generate 结构	21
第 2 章 16 位微程序控制计算机的设计	24
2.1 概述	24
2.2 指令系统设计	25
2.2.1 指令格式及寻址方式	25
2.2.2 指令类型	26
2.3 运算器设计	28

2.3.1	补码加减运算电路	28
2.3.2	运算结果的特征标志	29
2.3.3	多功能加减运算电路	29
2.3.4	算术逻辑单元 ALU 设计	31
2.3.5	移位寄存器设计	33
2.3.6	运算器数据通路	35
2.4	微程序控制器设计	37
2.4.1	微程序控制器的基本组成	37
2.4.2	微指令寄存器 μ IR 和微指令译码	37
2.4.3	微地址寄存器和微地址的形成	39
2.4.4	微程序控制时序	46
2.5	微程序设计	47
2.5.1	指令执行过程	47
2.5.2	微程序的设计方法	47
2.5.3	取指令的微程序设计	48
2.5.4	取操作数阶段微程序设计	50
2.5.5	执行阶段微程序设计举例	51
2.6	主存储器	56
2.7	输入输出	57
2.7.1	概述	57
2.7.2	输出接口	60
2.7.3	输入接口	61
2.7.4	中断控制器	62
2.7.5	CPU 对中断的支持	64
2.8	片上调试器	65
2.8.1	JTAG 简介	65
2.8.2	JTAG 调试原理及结构	67
第 3 章	实验项目	70
3.1	信号和传输	70
3.1.1	实验目的	70
3.1.2	实验原理	70
3.1.3	实验操作和记录	74
3.2	加减运算及特征标志	76
3.2.1	实验目的	76
3.2.2	实验原理	76
3.2.3	预习要求	79
3.2.4	实验操作和记录	79
3.3	运算器数据通路	81

3.3.1	实验目的	81
3.3.2	实验原理	81
3.3.3	预习要求	86
3.3.4	实验操作和记录	86
3.4	主存储器组织	88
3.4.1	实验目的	88
3.4.2	实验原理	88
3.4.3	预习要求	92
3.4.4	实验操作和记录	92
3.5	高速缓冲存储器	93
3.5.1	实验目的	93
3.5.2	实验原理	93
3.5.3	预习要求	97
3.5.4	实验操作和记录	97
3.6	指令和寻址方式	100
3.6.1	实验目的	100
3.6.2	实验原理	100
3.6.3	实验操作和记录	100
3.7	微程序控制器	103
3.7.1	实验目的	103
3.7.2	实验原理	103
3.7.3	预习要求	110
3.7.4	实验操作和记录	110
3.8	微程序设计	115
3.8.1	实验目的	115
3.8.2	实验原理	115
3.8.3	预习要求	116
3.8.4	实验操作和记录	116
3.9	中断电路	117
3.9.1	实验目的	117
3.9.2	实验原理	118
3.9.3	预习要求	121
3.9.4	实验操作和记录	121
3.10	输入输出和中断	124
3.10.1	实验目的	124
3.10.2	实验原理	124
3.10.3	预习要求	127
3.10.4	实验操作和记录	127
3.11	实验电路的调试支持	129

第4章 课程设计项目	133
4.1 项目1——微程序设计	133
4.1.1 熟悉微程序的设计和调试方法	133
4.1.2 双操作数指令的设计与调试	135
4.1.3 条件转移指令的设计与调试	135
4.1.4 移位指令的设计与调试	136
4.1.5 堆栈相关指令的设计与调试	137
4.1.6 中断系统的设计与调试	137
4.1.7 考核	138
4.2 项目2——CPU设计	138
4.2.1 CPU的初步设计与验证	138
4.2.2 扩充输出接口	140
4.2.3 扩充条件转移指令	142
4.2.4 扩充移位指令	143
4.2.5 扩充堆栈类指令	143
4.2.6 中断系统的设计	144
4.2.7 考核	145
第5章 设计工具与实验环境	146
5.1 Altera Quartus II 使用入门	146
5.1.1 设计流程	146
5.1.2 片内存储器块的使用	159
5.1.3 系统存储器数据编辑器	166
5.2 Xilinx ISE 使用入门	167
5.2.1 设计流程	167
5.2.2 片内存储器块的使用	181
5.3 实验开发板	185
5.3.1 Altera/Terasic DE2-115 教学开发板	185
5.3.2 Xilinx/Digilent Nexys3 FPGA 开发板	190
5.4 实验系统	194
5.4.1 实验系统组成	194
5.4.2 实验系统软件	196
5.4.3 逻辑部件实验的操作	197
5.4.4 模型机实验的操作	201

第1章

Verilog HDL 快速入门

1.1 Verilog HDL 概述

Verilog HDL(简称 Verilog)最初是 GDA(Gateway Design Automation)公司在 1983 年开发的一种硬件描述语言(Hardware Description Language),GDA 公司随后推出了相应的逻辑仿真器产品 Verilog-XL。1987 年,Synopsys 公司发布了第一个 Verilog 的逻辑综合工具,提高了数字电路的设计效率。1989 年,著名 EDA 公司 Cadence Design Systems 并购了 GDA 公司,并于 1990 年公开发表了 Verilog HDL,成立了 OVI(Open Verilog Internation)组织来负责 Verilog HDL 的发展。在 OVI 组织的推动下,Verilog HDL 于 1995 年被接受为 IEEE 标准,即 IEEE std 1364—1995,之后在 2001 年和 2005 年分别进行了修订,通常简称为 Verilog—1995、Verilog—2001 和 Verilog—2005。Verilog—2001 是对 Verilog—1995 的一个重大改进版本,引入了一些新的特性;Verilog—2005 只是对 Verilog—2001 做了一些细微修正。Verilog—2001 是目前 Verilog 的主流版本,被大多数商业电子设计自动化软件支持,本书以 Verilog—2001 为主介绍。

和传统的原理图输入方式相比,硬件描述语言有以下优点:

(1) HDL 比原理图更有效率,可以描述更复杂的系统。

传统的原理图设计方法,描述一个系统需要几十张至几百张图纸。但是随着设计规模日益增大,原理图描述变得过于复杂,设计和维护就变得很不方便。采用文本的输入方式,可以将精力集中到系统的功能描述,而不必顾及绘图的方法;能够以较少的时间完成更大的、更复杂的设计。

(2) HDL 比原理图具有更高层次的抽象表达能力。

原理图设计是将已有的电路元件按照某种设计意图连接起来,实现一定的逻辑功能;设计者必须有一种能力,用逻辑元件的组合来实现所需要的系统行为。而硬件描述语言具有行为级的描述能力,能够以较抽象的形式描述系统;将行为描述转变为逻辑电路的工作交由综合工具完成。

(3) 有利于设计的维护和重用。

代码的重用比原理图更有优势,例如硬件描述语言支持参数化设计,易于从整体上修改设计,例如修改总线宽度。文本的代码也比原理图有更好的可移植性,易于迁移到不同的设计环境。

(4) 不仅可以用于逻辑设计,还可以用于系统建模和仿真。

原理图只能用于逻辑设计,而硬件描述语言不仅可以用于逻辑设计,还可以用于系统建

模和仿真。在传统的硬件设计中,仿真和调试通常只能在后期进行,一旦出现问题,重新修改设计的代价很大。基于 HDL 的设计可以在早期就进行仿真,较早发现设计错误,极大地缩短设计周期,降低设计成本。

Verilog HDL 的语法非常类似于 C 语言,很多关键字也是相同的。这使得熟悉 C 语言的工程师学习 Verilog 变得容易。但是 C 语言和 Verilog 又有着本质的不同,C 语言是程序设计语言(Programming Language),而 Verilog 是描述语言(Description Language)。Verilog 和 C 语言最大的不同,就是 C 语言的语句最终是由某个 CPU 依次执行的,而 Verilog 语句并不是被某个 CPU 执行,它描述的是并行工作的硬件。要想用好 Verilog,必须有数字逻辑的基础知识和硬件的思维方式。

作为一种硬件描述语言,Verilog HDL 既可用于仿真建模,也可用于逻辑综合。所谓综合,就是将抽象的 HDL 描述依据约束条件转变成门级逻辑网络。逻辑综合通常由综合工具软件完成。Verilog 能在多个层次上进行设计描述,从开关级、门级、寄存器传输级(RTL),到算法级甚至系统级,都可以胜任;但是用 Verilog 描述的电路模型,不一定都能够被综合,有些语句只能用于仿真,本书着重介绍可综合的 Verilog 描述。Verilog HDL 支持多种描述风格:行为描述、结构描述、数据流描述。结构描述是通过基本元件(如与、或、非门,触发器等)和相互连接关系描述电路,本质上和画原理图相同,只是将原理图的符号和连线用语句来描述。行为描述是对设计的功能进行描述,设计者不需要考虑具体用什么逻辑电路来实现,具有一定的抽象性,比较适合 RTL 级和算法级的设计。数据流描述通常是指用 assign 赋值语句对组合逻辑电路功能的描述。在实际应用中,往往需要三种描述方式混合使用。

1.2 Verilog HDL 语法概要

Verilog 的语法非常类似于 C 语言,例如标识符是大小写敏感的,所有的关键字都是小写,单行注释以//开头,多行注释以/*开头、以*/结尾,标识符的组成等。这里假定读者具有 C 语言语法基础,因此不对 Verilog 语法作详细介绍,重点介绍 Verilog 特有的语言现象。

1.2.1 数据类型及数的表示

1. 四值逻辑

Verilog HDL 有下列四种逻辑值:

0: 逻辑 0 或“假”;

1: 逻辑 1 或“真”;

x: 不确定值(Unknown Value);

z: 高阻。

2. 数据类型

有两类重要的数据类型,变量类型和线网类型。它们的主要区别在于赋值和维持数据的方式。

线网类型包括 wire、tri、wand、wor 等,在 FPGA 设计中主要使用 wire 型。正如 wire 所表达的含义,可以把它理解为电路中的连线。wire 型不存储值,它的值是由驱动端的值

决定的。wire 型的初始值是 z。wire 是系统缺省的线网类型,也可以用'default_nettype 改变,参见 1.2.4 节。

变量类型包括 reg、integer、time、real 和 realtime 等。reg、integer、time 型的初值是 x, real 和 realtime 型的初值是 0.0。变量类型是数据存储特性的抽象,在一次赋值之后它保持其值直到下一次赋值。reg 是可综合为物理元件的类型,其他几个主要用于高层次的抽象建模和仿真。需要特别注意的是,reg 型变量并不一定就是逻辑电路中的寄存器,这里不能望文生义,在 1.3.2 节将有具体实例。在最初的 Verilog—1995 标准中,reg 类型被称为寄存器(Register)类型,在 Verilog—2001 中用“变量”(Variable)代替了术语“寄存器”(Register),就是为了避免将 reg 型变量理解为寄存器。

3. 标量和向量

(1) 标量和向量的声明

线网和 reg 型可以指定数据宽度。如果没有指定,缺省为 1 位,称为标量(Scalar),声明方式举例如下:

```
wire w;           // wire 型标量
reg a;            // reg 型标量
wire w1, w2;      // 声明 2 个 wire 型标量
```

如果指定了位宽,就称为向量(Vector)。声明方式举例如下:

```
wire [15:0] busa; // 16 位总线
reg [3:0] v;      // 4 位 reg 型向量
reg [-1:4] b;     // 6 位 reg 向量
reg [4:0] x, y, z; // 声明 3 个 5 位 reg 型向量
```

缺省情况下,线网和 reg 型向量是无符号的;可以用关键字 signed 声明为有符号的(注:Verilog—1995 没有有符号的向量),例如:

```
reg signed [3:0] signed_reg; // 有符号的 4 位 reg 向量,取值范围是 -8~+7
wire signed [7:0] s;         // 有符号的 8 位 wire 向量
```

(2) 向量的位选择和部分选择

从向量中抽取一位称为位选择(Bit-Selects)。可以用一个表达式指定选择的位,语法如下:

```
vect[expr]
```

从向量中抽取几个相邻的位称为部分选择(Part-Selects)。部分选择有两种表达方式:常量部分选择(Constant Part-Select)和可变部分选择(Indexed Part-Select)。常量部分选择的语法如下:

```
vect[msb_expr:lsb_expr]
```

其中 msb_expr 和 lsb_expr 都只能是常量。可变部分选择的语法如下:

```
[base_expr + : width_expr]
[base_expr - : width_expr]
```

width_expr 是选择的位宽,必须是常量; base_expr 是选择的起始位,可以是常量或变量; + 表示由 base_expr 向上增长 width_expr 位, - 表示由 base_expr 向下递减 width_expr 位。例如:

```
reg [31: 0] big_vect;
reg [0 :31] little_vect;
reg [63: 0] dword;
integer sel;
big_vect[ 0 + : 8]           // == big_vect[ 7 : 0]
big_vect[15 - : 8]           // == big_vect[15 : 8]
little_vect[ 0 + : 8]         // == little_vect[0 : 7]
little_vect[15 - : 8]         // == little_vect[8 :15]
dword[8 * sel + : 8]          // 具有固定宽度的可变部分选择
```

最后说明一下, integer 类型不能指定数据宽度,而是使用系统的设置,缺省是 32 位。integer 类型不存在标量和向量之分。integer 类型的变量是有符号的。

4. 常数的表示

Verilog HDL 的定长(Sized)整型常数的表示格式如下:

<位数> ' <基> <数字>

<位数>是用十进制表示的数字的位数; ' <基>用来定义此数为十进制('d 或'D)、二进制('b 或'B)、十六进制('h 或'H)、八进制('o 或'O); <数字>即用相应进制表示的数,也可以包含 x 和 z,不区分大小写。举例如下:

```
4'b1001           // 4 位二进制数
5'D3              // 5 位十进制数
12'habc           // 12 位十六进制数
3'b01x            // 3 位二进制数,最低位是不确定值
16'hz             // 16 位高阻
4'b10??          // ? 和 z 相同,即 4'b10zz
```

如果不指定<位数>,则称为不定长(Unsigned)数;此时若<基>省略,则表示十进制。举例如下:

```
659              // 十进制数
'h 837FF         // 十六进制数
'o7460           // 八进制数
4af              // 非法(十六进制格式应该有'h)
```

表示负数的负号应放在最前面,如:

```
- 8'd3           // 用 8 位二进制补码表示的 -3
```

可以用下划线增强可读性,如:

```
12'b1111_0000_1010           // 即 12'b111100001010
```

在基的符号前加入 s 符号,可以显式地表明常数是有符号数。如:

```
8'sh5d           // 8 位十六进制有符号常数 + 5DH
```

Verilog 中的实数既可以用小数(如 0.5),也可以用科学计数法(如 3e6,1.7E8)来表达,带小数点的实数在小数点两侧都必须至少有一位数字。

5. 数组

各种数据类型都可以定义数组。数组的维数没有限制(注:Verilog—1995 只能定义一维数组)。举例如下:

```
reg [7:0] mema[0:255];           // 声明一个字长为 8 位、有 256 个单元的存储器
reg arrayb[7:0][0:255];         // 声明一个二维数组,字长 1 位
wire w_array[7:0][5:0];         // 声明 wire 型二维数组
integer inta[1:64];             // 声明有 64 个元素的 integer 型数组
time chng_hist[1:1000]         // 声明有 1000 个元素的 time 型数组
```

1.2.2 运算符

Verilog 的运算符见表 1.1,形式上大部分和 C 语言类似,只有归约运算符(Reduction)、并接/复制运算符(Concatenation,Replication)以及算术移位运算符是 Verilog 特有的,运算符的优先级见表 1.2。

表 1.1 Verilog 的运算符

分类	运算符及功能	简要说明
算术运算符	+ 加 - 减 * 乘 / 除 % 取余 ** 乘方	二元运算符,即有两个操作数。 % 是求余运算符,在两个整数相除基础上,取余数。 例如,5%6 的值是 5;13%5 余数 3
比较运算符	> 大于 < 小于 >= 大于等于 <= 小于等于 == 逻辑相等 != 逻辑不等 === 全等 !== 非全等 && 逻辑与 逻辑或 ! 逻辑非	! 为一元运算符,其他是二元运算符,关系运算的结果是 1 位逻辑值。如果操作数之间的关系成立,返回值为 1;关系不成立,则返回值为 0;若某一个操作数为不定值 x,则关系是模糊的,返回值是不定值 x。 逻辑相等与全等运算符的区别:逻辑相等运算,如果两个操作数中含有不定值或高阻值,则结果为不定值;而全等运算的结果要么为 1,要么为 0。例如: A = 8'b1101xx01 B = 8'b1101xx01 则 A = B 运算结果为 x(不定); A == B 运算结果为 1(真)
位逻辑运算符	~ 按位非 & 按位与 按位或 ^ 按位异或 ^~(~ ^) 按位同或	~ 是一元运算符,其余都是二元运算符。将操作数按位进行逻辑运算

续表

分类	运算符及功能	简要说明
归约运算符	& 归约与 ~& 归约与非 归约或 ~ 归约或非 ^ 归约异或 ~^(^~) 归约同或	一元运算符,对操作数各位的值进行运算。如“&”是对操作数各位的值进行逻辑与运算,得到一个一位的结果值 1 或 0。例如: A=8'b11010001,则 &A=0, A=1。归约与运算 A 中的数字全为 1 时,结果才为 1; 归约或运算 A 中的数字全为 0 时,结果才为 0
移位运算符	<< 逻辑左移 >> 逻辑右移 <<< 算术左移 >>> 算术右移	二元运算符,对左侧的操作数进行它右侧操作数指明的位数的移位。逻辑移位和算术左移时空出的位用 0 补全。算术右移时空出位的补全取决于结果的数据类型,如果是无符号型,补 0; 如果是有符号型,复制最高位。如果操作数有不定值 x 或高阻值 z,结果为 x
条件运算符	?:	三元运算符,即条件运算符有三个操作数。 操作数=条件? 表达式 1: 表达式 2; 当条件为真(值为 1)时,操作数=表达式 1; 为假(值为 0)时,操作数=表达式 2
并接运算符 复制运算符	{, } { { } }	将两个或两个以上用逗号分隔的表达式按位连接在一起。还可以用常数来指定重复的次数。例如 {a, {2{a,b}}} 等价于 {a,a,b,a,b}

表 1.2 运算符的优先级

+ - ! ~ & ~& ~ ^ ~ ^ ^ ~ (一元)	高优先级 ↓ 低优先级
**	
* / %	
+ - (二元)	
<< >> <<< >>>	
< <= > >=	
== != === !==	
& (二元)	
^ ^ ~ ~^ (二元)	
(二元)	
&&	
?:	
{ } { { } }	