高等学校电子与通信工程类专业“十二五”规划教材

数字逻辑与EDA设计

主编　丁　磊　冯永晋　张海笑
主审　林福宗

西安电子科技大学出版社
http://www.xduph.com

高等学校电子与通信工程类专业“十二五”规划教材

数字逻辑与EDA设计

主　编　丁磊　冯永晋　张海笑

主　审　林福宗

西安电子科技大学出版社

内 容 简 介

全书共分两个部分：经典篇与现代篇。经典篇(第 1～3 章)主要介绍数字电路的基本概念和基础知识以及组合与时序逻辑电路的分析和设计方法。现代篇(第 4～7 章)介绍 Verilog HDL 的基本语法以及基于 Verilog HDL 和 EDA 工具的数字电路的设计方法，其中第 4、5 章介绍基本概念和 Verilog HDL 语法，并给出了几个例子的详细设计流程，是设计的基础；第 6、7 章介绍基本组合逻辑电路和时序电路的设计、综合及验证方法，其中第 7 章的综合例子由浅入深，尝试引导读者进行实际应用的设计。

本书可作为高等院校计算机、信息、自动化、电子等专业本科生、研究生的教材，亦可供从事数字电路设计的工程人员使用。

图书在版编目(CIP)数据

数字逻辑与 EDA 设计/丁磊，冯永晋，张海笑主编. —西安：西安电子科技大学出版社，2012.8
高等学校电子与通信工程类专业“十二五”规划教材
ISBN 978-7-5606-2854-7

Ⅰ. ① 数… Ⅱ. ① 丁… ② 冯… ③ 张… Ⅲ. ① 数字电路—高等学校—教材
② 逻辑电路—高等学校—教材 ③ 电子电路—电路设计—计算机辅助设计—高等学校—教材
Ⅳ. ① TN79 ② TN702

中国版本图书馆 CIP 数据核字(2012)第 155618 号

策　　划　邵汉平
责任编辑　邵汉平
出版发行　西安电子科技大学出版社(西安市太白南路 2 号)
电　　话　(029)88242885　88201467　　邮　　编　710071
网　　址　www.xduph.com　　电子邮箱　xdupfxb001@163.com
经　　销　新华书店
印刷单位　陕西天意印务有限责任公司
版　　次　2012 年 8 月第 1 版　2012 年 8 月第 1 次印刷
开　　本　787 毫米×1092 毫米　1/16　印　张　26.5
字　　数　631 千字
印　　数　1～3000 册
定　　价　45.00 元

ISBN　978-7-5606-2854-7/TN • 0665
XDUP 3146001-1

前　言

2008年仲夏，我们研发团队第一块基于8位Atmel单片机及Actel FPGA的TFT LCD驱动板调试通过，虽然很简单，但我们还是难以按捺心中的喜悦，同时也在思考如何将与EDA工具及FPGA相关的课程引入本科教学。

将自己本科学习“数字电子技术基础”的教材与现在的教材对比了一下，两者在内容上居然没有太大的变化(24年!)，教学学时也基本持平，这在数字技术飞速发展的今天是难以想象的。

VLSI、FPGA等的迅速发展已经给出了课程迫切需要改革的信号，传统的教学及科研方法已产生众多需要优化及整合的内容。于是就先从“数字系统设计——基于EDA工具”公选课开始了尝试，但是一年下来，效果很差。安排在三年级的课程与“数字电子技术基础”的时间跨度太大，前后衔接不上，而且实验环节也未跟上。

正是在这种背景下，就有了将“数字电子技术基础”与“HDL硬件描述语言”合并的初步想法，并萌生了写教材的冲动。从那时起，几乎将书店能够见到的相关书籍都买了回来参考，再加上前几年的一些科研项目积累，开始没觉得有什么太大的难度，觉得写一本本科生的教材易如反掌。真正做起来才知道不容易，特别是原先想重点突出的“超越课程”的部分内容，很难把握。现在回头来看，还好坚持了下来，不论怎样，都是想将最好的内容呈现给读者。

本书创新地将数字电子技术划分成经典篇与现代篇两大部分，在介绍经典数字电子技术的基本概念、分析及设计理论的前提下，以现代数字电子技术设计流程为主线，利用先进的集成设计环境，对组合及时序等实用电路进行综合、简化和验证，通过大量完整实例描述了使用Verilog HDL进行超大规模集成电路设计的关键步骤以及设计验证方法等，旨在鼓励读者精通描述和验证他们的设计。

注重验证及测试环节的学习，注重方法论的学习，强调理论与实际应用的结合，学有所用、学以致用是本书追求的目标。

在初步掌握逻辑设计方法的基础上，进一步充实读者的设计背景，鼓励读者精炼、明晰化并验证他们自己的设计。

许多有关HDL的参考书都非常好，但其中多数定位于对语言语法的讲解，不太适合于课堂教学。而我们的着眼点主要放在使用HDL的设计方法上，这是本书的独特之处。

集中而有重点地介绍Verilog语言仅仅是为了支持设计实例的需要。本书中列举的实例表明了在使用Verilog HDL的超大规模集成电路设计方法中如何应用关键步骤进行设计。所列举的实例都是完整的，并且包括了已被ModelSim工具验证正确的源代码。所有实例

的源代码和一些重要测试程序集均完整提供给读者。

读者对象

本书可作为高等院校计算机、信息、自动化、电子等专业本科生、研究生的教材，亦可供从事数字电路设计的工程人员使用。

本书的目的

(1) 学习组合、时序逻辑的基本原理；
(2) 学习基本组合、时序逻辑芯片的使用方法；
(3) 介绍 Verilog HDL 的基本语法；
(4) 介绍 Verilog HDL 在数字系统设计中的应用；
(5) 着重讨论能使读者快速设计适于 FPGA 实现的电路描述方式；
(6) 提供大量使用现代 EDA 设计工具进行高层次设计的实例。

章节安排

本书由丁磊、冯永晋、张海笑主编，其中丁磊统稿并负责第 5、7 章的编写，张海笑负责第 1、2、3 章的编写，冯永晋负责第 4、6 章的编写，江志文主要负责配套实验指导书的编写及全书的排版工作。林小平、邓杰航、李峥、张静、侯艳、周永根均对教材中的错漏提出了精准的修改意见。

研究生荣晶、肖丽萍、简芳完成了大量插图的绘制工作。第一批使用讲义的 2010 级本科生中很多学生都针对讲义中的问题提出了有益的修改意见。

定稿后有幸邀得清华大学计算机系林福宗教授审稿。林福宗教授在肯定教材内容的同时，提出了详细的修改、修正意见，大到全书的结构，小到排版及标点符号，使我们得到了很大的启发及提高，同时为林福宗教授严谨的治学态度所折服。在此一并致谢。

本书编写过程中参考了国内外同行的大量文献，在核稿时均作了引用标注，疏漏之处敬请读者指出。

由于编者的水平及能力所限，书中的错漏之处在所难免，恳请读者批评指正。

来信请寄编者邮箱：gzeking@sina.com。

编　者

2012 年 3 月于广州

目　录

第 1 章　数字逻辑基础

数字系统所处理的信息都是二进制形式的数字信号。本章介绍数字信号的表示方式，数制和码制；重点介绍数字逻辑设计的基础，包括逻辑代数的基本公式和定理、逻辑函数的表示方法、逻辑函数的化简以及逻辑门电路的基本结构和常用的集成门电路。

1.1　概　　述

1.1.1　数字信号及模拟信号

客观世界中存在着各种各样的物理量，用于量度物体属性或描述物体运动状态及其变化过程。根据各物理量变化规律的特点可将它们划分为两大类。

一类物理量的变化在时间和数值上都是连续的。这一类物理量叫做模拟量，如电压、频率、压力、温度等。时间上的“连续”是指在一个指定的时间范围里，物理量的数值个数有无穷多个。而数值上的“连续”是指物理量的数值本身的数目有无穷多个。我们把在时间上和数值上都连续的物理量称为模拟物理量，表示模拟量的信号叫做模拟信号，如图 1-1(a)所示，而处理模拟信号的电子电路被称为模拟电路。

另一类物理量的变化在时间上和数量上都是离散的。例如产品的数量、学生的成绩、开关的状态等，它们的值都是离散值。时间上的“离散”是指在一个指定的时间范围里，物理量的数值的个数是有限的。而数值上的“离散”是指物理量的数值本身的数目是有限的。我们把在时间上和数值上都离散的物理量称为数字量，表示数字量的信号称为数字信号，如图 1-1(b)所示，直接对数字量进行处理的电子线路被称为数字电路。

由于数字电路中存储、处理和传输的信号都是数字信号，因而对于模拟信号而言，需要将其转换为数字信号，才能在数字电路中进行存储、处理和传输。图 1-1 显示出了这种转换的方法。图 1-1 中，经过了采样、量化和编码后，模拟信号转换成了数字信号。所谓采样，是指以相等的时间间隔，将时间上连续的模拟信号截取成时间上离散的数字信号，也就是在时间上将模拟信号离散化。量化是指将采样得到的瞬间幅度值离散化，也就是用有限个幅度值近似表示原来连续变化的幅度值，即把模拟信号的连续幅度值变为有限数量的有一定间隔的离散值。例如，用离散值

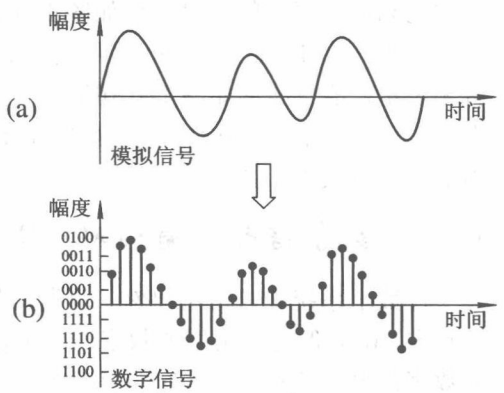


图 1-1　模拟信号与数字信号

–5 V，–4 V，…，4 V，5 V 表示电压的幅度值，当采样得到的幅度值是 1.23 V 时，量化后的值取 1 V；当采样得到的幅度值是 –2.68 V 时，量化后的值取 –3 V。编码则是按照一定的规律，把量化后的值用二进制数字表示。例如，1 V 用二进制数 0001 表示，–3 V 用二进制数 1101 表示。

1.1.2 数字抽象

数字系统是一个能对数字信号进行处理和传输的实体，由各种能实现特定功能的数字逻辑电路相互连接组合而成。在数字系统中，表示信息所使用的都是离散型的信号。

早期机械式的数字系统，如 Charles Babbage(查儿斯·巴贝奇)设计的分析机(现代电子计算机的前身)，使用蒸汽作动力，用 10 个齿轮分别表示 0～9 十个数字，是一个具有 10 个离散值变量的数字系统。

现在，大部分数字系统使用二进制处理信息，这是由于：

(1) 电路容易实现。二进制只有两个数码，在电路中用高、低电平分别表示“1”和“0”这两种状态，区分两种电压值要比区分 10 种电压值容易得多。

(2) 物理上容易实现存储。二进制在物理上最容易实现存储，如通过磁极的取向、表面的凹凸等来记录“0”、“1”信息。

(3) 便于运算。与十进制数相比，二进制数的运算规则要简单得多(例如二进制乘法只有 4 条规则)。这不仅可使运算器的硬件结构大大简化，而且有利于运算速度的提高。

(4) 便于逻辑判断。二进制中的 0 和 1 两个数码，正好与逻辑命题中的“真(True)”、“假(False)”相对应，为逻辑运算提供了便利。

目前，在计算机、数字通信及其他数字设备中，都是用两个电平状态来表示数字信号的。采用的两个状态符号就是 0 和 1(在二进制中，它们表示两个数字信号)，它们是构成数字信息的基本元素。也就是说，不论数字系统处理的信息是表示一个数，或者是表示控制某个装置动作的命令，也都不过是 0 和 1 的某种形式的组合。

对于 0 和 1 来说，既可以用电位的高低来表示，也可以用脉冲信号的有无来表示。例如在图 1-2 中，为表示 1101110010 这样的数字信号，用电位的高低来表示的信号波形如图 1-2(a)所示，以高电平表示 1，低电平表示 0；若以脉冲信号有无表示，信号波形如图 1-2(b)所示，以有脉冲表示 1，无脉冲表示 0。目前的数字系统中，常用前一种方式表示数字信号。

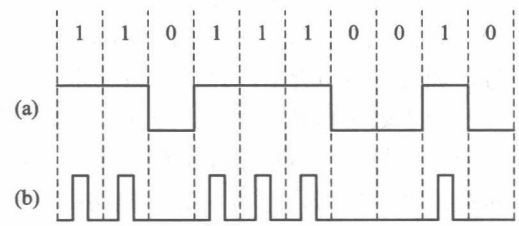


图 1-2 数字信号的表示

1.1.3 数字信号传输时对“0”、“1”的处理

在数字系统中，常用自然二进制对信号进行编码，即高电平表示数据“1”，低电平表示数据“0”。但这种类型的数字信号不适合于在信道中直接传输，原因在于：① 这种类型的数字信号往往存在直流分量和低频分量，而具有电容耦合电路的设备或频带低端受限的信道会过滤掉这些分量；② 当出现连续的“0”或“1”数据时，数字信号会出现长时间的低电平或高电平，接收端无法获取定时信息(即同步信息)；③ 接收端无法判断是否包含错

码。因此，数字信号在传输时需要选择其他的编码方式，常用的有以下四种。

1．不归零编码(NRZ)

不归零编码(NRZ)的编码规则是：“1”和“0”都分别由不同的电平状态来表现，用正电平表示“1”，用负电平表示“0”，除此之外，没有中性状态及其他状态。图 1-3 为不归零编码示意图。不归零编码发送能量大，直流分量小，抗干扰能力比较强，但使用这种编码需要另外传输同步信号。

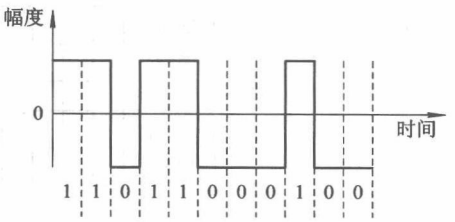


图 1-3　NRZ 编码示意图

2．NRZ-Inverted(NRZI)编码

NRZ-Inverted(NRZI)的编码规则是：如果输入为 0，则输出保持它的前一个值；如果输入为 1，则输出为前一个输出值的相反值。因此，只要输入为 0，输出就会保持不变；如果输入为 1，输出就会发生翻转。图 1-4 为 NRZ-Inverted 编码示意图。使用这种编码也同样需要另外传输同步信号。

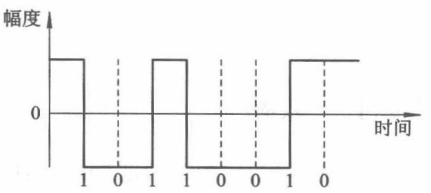


图 1-4　NRZ-Inverted 编码示意图

3．归零编码

归零编码(Return to Zero，RZ)的编码规则是：高电平表示“1”，低电平表示“0”，但整个码元分两部分，前半部分用高低电平表示数据，后半部分则归零位。显然，信号“0”在整个位的时间内以低电平发送；信号“1”在前半个位时间内以高电平发送，在剩余位时间内以低电平发送。图 1-5 为归零编码示意图。归零编码自带了同步信号，但当出现长串“0”时，将丢失同步信号。

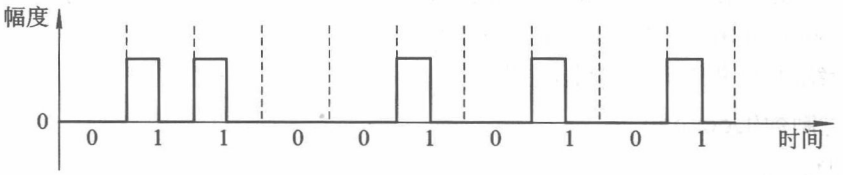


图 1-5　归零编码示意图

4．曼彻斯特编码

曼彻斯特编码(Manchester Encoding)也叫做相位编码(Phase Encoding，PE)，是一种用电平跳变来表示 1 或 0 的编码，这种编码带同步信号，即时钟同步信号就隐藏在数据波形中。在曼彻斯特编码中，每一位的中间有一电平的跳变，位中间的跳变既作时钟信号，又作数据信号；从高到低跳变表示“0”，从低到高跳变表示“1”。

还有一种是差分曼彻斯特编码，每位中间的跳变仅提供同步信号，而用每位开始时有无跳变表示“0”或“1”，如果有跳变则表示“0”，如果无跳变则表示“1”。

两种曼彻斯特编码将时钟同步信号和数据包含在数据流中，在传输代码信息的同时，也将时钟同步信号一起传输到对方。每位编码中有一跳变，不存在直流分量，因此具有自同步能力和良好的抗干扰性能。但每一个码元都被调成两个电平，所以数据传输速率只有调制速率的 1/2。图 1-6 为两种曼彻斯特编码示意图。

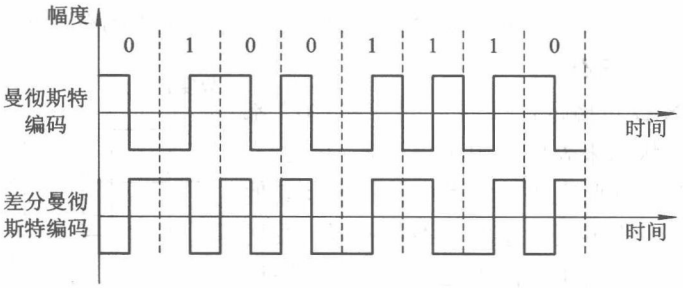


图 1-6　两种曼彻斯特编码示意图

1.2　数制与码制

1.2.1　数制

数制也称计数制，是指用一组固定的符号和统一的规则来表示数值的方法。由于按进位的方法进行计数，故也称进位计数制。在日常生活中，人们普遍采用十进制计数方式，而在数字系统中，最广泛使用的是二进制计数方式。

学习数制，必须首先掌握数码、基数和位权这三个概念。

数码：数制中为表示基本数值大小所使用的不同数字符号。例如，十进制有 10 个数码：0～9；二进制有 2 个数码：0、1。

基数：数制中所使用数码的个数。例如，十进制计数制使用 10 个数码，故基数为 10；二进制计数制使用 2 个数码，故基数为 2。

位权：数制中某位置上的数字 1 所表示数值的大小。例如，十进制的 435 中，4 所在位置上的位权是 100，3 的位权是 10，5 的位权是 1。

下面介绍常用的进位计数制及数制之间的转换。

1. 十进制数(Decimal)

十进制数的特点是：

(1) 由 10 个数码 0～9 组成。

(2) 基数是 10，运算规则是逢十进一。

(3) 在小数点左边，从右至左各位的位权依次是：10^0、10^1、10^2、10^3 等；在小数点右边，从左至右各位的位权依次是：10^{-1}、10^{-2}、10^{-3}、10^{-4} 等。

任意一个十进制数，都可以用位权展开式表示为

$$(d_n\cdots d_1d_0.d_{-1}d_{-2}\cdots d_{-m})_{10}=d_n\times10^n+\cdots+d_1\times10^1+d_0\times10^0+d_{-1}\times10^{-1}+d_{-2}\times10^{-2}+\cdots+d_{-m}\times10^{-m}$$

$$=\sum_{i=-m}^{n}d_i\times10^i$$

其中：d_i 表示各个位置上的十进制数码。例如，十进制数 826.78 的位权展开式为

$$(826.78)_{10}=8\times10^2+2\times10^1+6\times10^0+7\times10^{-1}+8\times10^{-2}$$

2. 二进制数(Binary)

二进制数的特点是：

(1) 由两个数码 0、1 组成。

(2) 基数是 2，运算规则是逢二进一。

(3) 在小数点左边，从右至左各位的位权依次是：2^0、2^1、2^2、2^3 等；在小数点右边，从左至右各位的位权依次是：2^{-1}、2^{-2}、2^{-3}、2^{-4} 等。

任意一个二进制数，都可以用位权展开式表示为

$$(b_n\cdots b_1b_0.b_{-1}b_{-2}\cdots b_{-m})_2 = b_n\times 2^n+\cdots+b_1\times 2^1+b_0\times 2^0+b_{-1}\times 2^{-1}+b_{-2}\times 2^{-2}+\cdots+b_{-m}\times 2^{-m}$$

$$=\sum_{i=-m}^{n} b_i\times 2^i$$

其中：b_i 表示各个位置上的二进制数码。例如，二进制数 1011.101 的位权展开式为

$$(1011.101)_2 = 1\times 2^3+0\times 2^2+1\times 2^1+1\times 2^0+1\times 2^{-1}+0\times 2^{-2}+1\times 2^{-3}$$

3. 八进制数(Octal)

八进制数的特点是：

(1) 由 8 个数码 0～7 组成。

(2) 基数是 8，运算规则是逢八进一。

(3) 在小数点左边，从右至左各位的位权依次是：8^0、8^1、8^2、8^3 等；在小数点右边，从左至右各位的位权依次是：8^{-1}、8^{-2}、8^{-3}、8^{-4} 等。

任意一个八进制数，都可以用位权展开式表示为

$$(o_n\cdots o_1o_0.o_{-1}o_{-2}\cdots o_{-m})_8 = o_n\times 8^n+\cdots+o_1\times 8^1+o_0\times 8^0+o_{-1}\times 8^{-1}+o_{-2}\times 8^{-2}+\cdots+o_{-m}\times 8^{-m}$$

$$=\sum_{i=-m}^{n} o_i\times 8^i$$

其中：o_i 表示各个位置上的八进制数码。例如，八进制数 723.24 的位权展开式为

$$(723.24)_8 = 7\times 8^2+2\times 8^1+3\times 8^0+2\times 8^{-1}+4\times 8^{-2}$$

4. 十六进制数(Hexidecimal)

十六进制数的特点是：

(1) 由 16 个数码 0～9 及 A～F 组成。

(2) 基数是 16，运算规则是逢十六进一。

(3) 在小数点左边，从右至左各位的位权依次是：16^0、16^1、16^2、16^3 等；在小数点右边，从左至右各位的位权依次是：16^{-1}、16^{-2}、16^{-3}、16^{-4} 等。

任意一个十六进制数，都可以用位权展开式表示为：

$$(x_n\cdots x_1x_0.x_{-1}x_{-2}\cdots x_{-m})_{16} = x_n\times 16^n+\cdots+x_1\times 16^1+x_0 16^0+x_{-1}16^{-1}+x_{-2}16^{-2}+\cdots+x_{-m}16^{-m}$$

$$=\sum_{i=-m}^{n} x_i\times 16^i$$

其中：x_i 表示各个位置上的十六进制数码。例如，十六进制数 2D9.A8 的位权展开式为

$$(2D9.A8)_{16} = 2\times 16^2+13\times 16^1+9\times 16^0+10\times 16^{-1}+8\times 16^{-2}$$

5. 数制之间的转换

1) 十进制数与非十进制数之间的转换

(1) 非十进制数转换成十进制数。非十进制数转换成十进制数的方法是：将非十进制

数按位权展开后求和。

【例 1-1】 将$(1011.101)_2$、$(723.24)_8$、$(2D9.A8)_{16}$分别转换成十进制数。

解

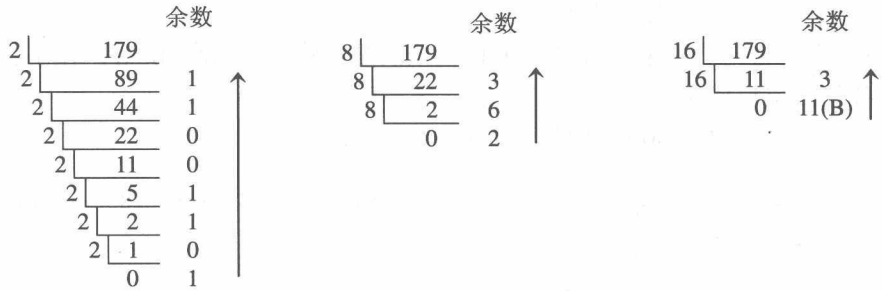
$$(1011.101)_2 = 1\times2^3+0\times2^2+1\times2^1+1\times2^0+1\times2^{-1}+0\times2^{-2}+1\times2^{-3}$$
$$= 8+0+2+1+0.5+0+0.125 = 11.625$$
$$(723.24)_8 = 7\times8^2+2\times8^1+3\times8^0+2\times8^{-1}+4\times8^{-2}$$
$$= 448+16+3+0.25+0.0625 = 467.3125$$
$$(2D9.A8)_{16} = 2\times16^2+13\times16^1+9\times16^0+10\times16^{-1}+8\times16^{-2}$$
$$= 512+208+9+0.625+0.03125 = 729.65625$$

(2) 十进制数转换成非十进制数。十进制数转换成非十进制数，需要分别对整数部分和小数部分进行转换。

① 整数部分的转换方法：不断除以基数取余数，直到商为 0，从下到上读取余数。

【例 1-2】 将$(179)_{10}$分别转换成二进制、八进制、十六进制数。

解

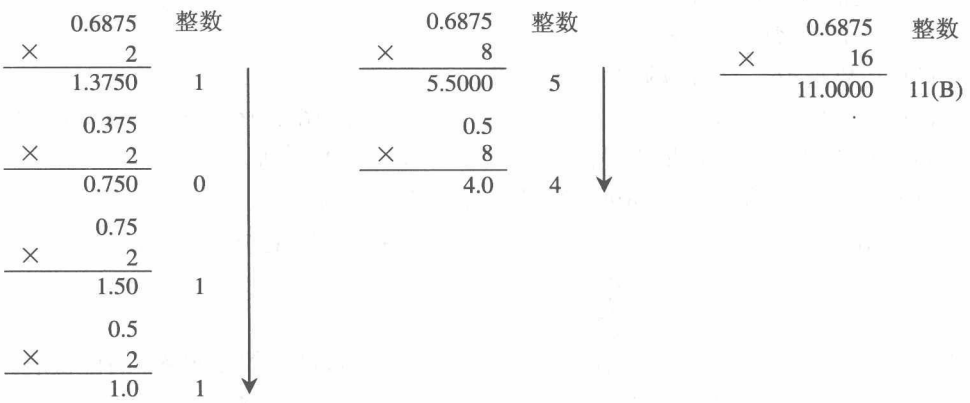


结果：$(179)_{10}=(10110011)_2=(263)_8=(B3)_{16}$

② 小数部分的转换方法：不断乘以基数取整数，从上到下读取整数，直到满足精度要求为止。

【例 1-3】 将$(0.6875)_{10}$分别转换成二进制、八进制、十六进制数。

解



结果：$(0.6875)_{10}=(0.1011)_2=(0.54)_8=(0.B)_{16}$

2) 二进制数与八进制或十六进制数之间的转换

虽然在数字系统中使用的是二进制数，但由于二进制数往往很长，不便书写及识别，

因此常使用八进制数或十六进制数来表示二进制数，以减少书写的长度。

由于八进制数、十六进制数的基数都是 $2^n(n=3,4)$，因此 1 位 2^n 进制数所能表示的数值能恰好用 n 位二进制数表示，即八(2^3)进制数中的数码 0～7 与 3 位二进制数 000～111 一一对应，十六(2^4)进制数中的数码 0～9、A～F 与 4 位二进制数 0000～1111 一一对应。所以，二进制数与 2^n 进制数之间可以按位进行转换。

(1) 二进制数转换成八进制、十六进制数。将二进制数转换成 $2^n(n=3,4)$进制数的方法是：以小数点为界，分别向左、右两个方向按 n 位进行分组，左右两端不足 n 位的，分别用 0 补够 n 位，再将每组二进制数转换为对应的 2^n 进制数。

【例 1-4】 将二进制数$(11010101001101.11001)_2$分别转换为八进制数及十六进制数。

解

二进制数	011	010	101	001	101	.	110	010
八进制数	3	2	5	1	5	.	6	2
二进制数	0011	0101	0100	1101	.	1100	1000	
十六进制数	3	5	4	D	.	C	8	

结果：　$(11010101001101.11001)_2=(32515.62)_8=(354D.C8)_{16}$

(2) 八进制数或十六进制数转换成二进制数。将 $2^n(n=3,4)$进制数转换成二进制数的方法与上面正好相反，将 2^n 进制数中的每一位直接转换成 n 位二进制数即可。

【例 1-5】 将八进制数$(7301.24)_8$、十六进制数$(4A3.E6)_{16}$分别转换成二进制数。

解

八进制数	7	3	0	1	2	4
二进制数	111	011	000	001	010	100

结果：　$(7301.24)_8=(111011000001.0101)_2$

十六进制数	4	A	3	.	E	6
二进制数	0100	1010	0011	.	1110	0110

结果：　$(4A3.E6)_{16}=(10010100011.1110011)_2$

1.2.2 码制

1. 数字的存储形式

前面所提到的二进制数，没有考虑符号问题，所指的都是无符号数。但实际上数字是有正、负符号的，那么在数字系统中对数字进行运算操作时，正负符号该如何表示呢？

以数字 6 为例，按习惯的表示方法，正 6 用 +6 表示，二进制数为 +110；负 6 用 −6 表示，二进制数为 −110。但在数字系统中，符号“+”、“−”也要数字化，一般将数的最高位设为符号位，用“0”表示“+”、用“1”表示“−”，即

+6	→	+110	→	0110
−6	→	−110	→	1110
(十进制数)		(真值)		(机器数)

为了区分一个符号数的“+”、“−”符号数字化前后的两种表示方式，引入真值和机器数两个术语。在二进制数之前用“+”、“−”符号表示正、负数的这种符号数称为真值。将符号用二进制码“0”、“1”表示的这种符号数称为机器数。

机器数常用的表示方法有原码、反码、补码等。下面以带符号的整数为例介绍原码、

反码及补码的定义规则。

2. 原码

将数的真值形式中正数符号用符号位 0 表示，负数符号用符号位 1 表示，叫做数的原码形式，简称原码。原码的定义可用下式表示：

$$A_{原}=\begin{cases}A & 2^{n-1}>A\geq 0\\ 2^{n-1}+|A| & 0\geq A>-2^{n-1}\end{cases}$$

式中 A 为真值，$|A|$ 为 A 的绝对值的真值，$A_{原}$ 为 A 的原码，n 为二进制数码的位数。

例如，绝对值为 9 的数，它的真值形式和原码形式如下所示(用 8 位数码表示，最高位为符号位)：

数	真值	原码
+9	+0001001	0 0001001
−9	−0001001	1 0001001

原码的优点是易于辨认，因为它的数值部分就是该数的绝对值，而且与真值和十进制数的转换十分方便。但在采用原码进行运算时，运算比较复杂，例如当两个数相加时，如果是同号，则数值直接相加；如果是异号，则要进行减法运算，首先要判定哪个数的绝对值大，绝对值大的作为被减数，小的作为减数，所得差值的符号与绝对值大的数符号一致。这样就要增加判定数大小的设备，而且要用减法器来完成减法运算，显然增加了设备量。为了减少设备量，将减法变为加法运算，就引进了反码和补码表示法。

3. 反码

与原码相比较，反码也是在数码左边加上一位符号位，0 代表正数，1 代表负数。与原码不一样的是，反码的数码与它的符号位有关：对于正数，反码与原码相同；对于负数，反码的数码由原码逐位求反而得(不包括符号位)。反码定义可用下式表示：

$$A_{反}=\begin{cases}A & 2^{n-1}>A\geq 0\\ (2^{n}-1)-|A| & 0\geq A>-2^{n-1}\end{cases}$$

式中：A 为原码，$|A|$ 为 A 的绝对值的原码，$A_{反}$ 为 A 的反码，n 为二进制数码的位数。

例如，绝对值为 9 的数，如用 8 位二进制反码表示，则 +9 的反码为

$$A_{反}=A=0\ 0001001$$

−9 的反码为

$$A_{反}=2^{8}-1-|A|=(100000000-1)-(00001001)=11110110$$

即

数	真值	原码	反码
+9	+0001001	0 0001001	0 0001001
−9	−0001001	1 0001001	1 1110110

由此可看出：正数的反码与原码相同，负数的反码为其绝对值的原码按位取反。

4. 补码

机器数的第三种表示是补码形式。对于正数来说，其原码、反码和补码的表示是相同