面向21世纪高等院校课程规划教材

汇编语言程序设计

——基于ARM体系结构（第2版）

ASSEMBLY LANGUAGE PROGRAM DESIGN

ARM

Advanced RISC Machines

文全刚 张 平 主 编
张荣高 副主编

CD-ROM INCLUDED 配套多媒体教学课件

北京航空航天大学出版社

面向21世纪高等院校课程规划教材

汇编语言程序设计

——基于ARM体系结构(第2版)

文全刚　张　平　主　编
张荣高　副主编

北京航空航天大学出版社

内 容 简 介

随着嵌入式技术的发展，越来越需要一套很好的嵌入式系列教材。本书是学习嵌入式技术的入门教材，是学习嵌入式系统原理与接口技术、嵌入式系统设计与应用等知识的前导教材。

本书基于ARM体系结构进行汇编语言的教学。全书的内容分成三个部分：第一部分主要介绍汇编语言程序设计的基础知识和ARM系列微处理器，包括第1、2章；第二部分主要介绍基于ARM体系结构的指令系统，汇编程序设计，以及异常中断编程，包括第3、4、5章；第三部分主要是MDK集成开发环境的使用和ARM汇编语言程序实验，包括第6、7章。书中含光盘1张，内含相关章节程序源代码及其他相关资料。

本书可作为高等院校计算机及相关专业本科生和专科生的汇编语言程序设计课程的教材或参考书，也可供使用汇编语言的工程技术人员参考。

图书在版编目(CIP)数据

汇编语言程序设计：基于ARM体系结构/文全刚等主编．--2版．--北京：北京航空航天大学出版社，2010.8

ISBN 978-7-5124-0187-7

Ⅰ.①汇… Ⅱ.①文… Ⅲ.①汇编语言—程序设计—高等学校—教材 Ⅳ.①TP313

中国版本图书馆CIP数据核字(2010)第161084号

汇编语言程序设计

——基于ARM体系结构(第2版)

文全刚 张 平 主 编

张荣高 副主编

责任编辑 董立娟

*

北京航空航天大学出版社出版发行

北京市海淀区学院路37号(邮编100191) http://www.buaapress.com.cn

发行部电话：(010)82317024 传真：(010)82328026

读者信箱：emsbook@gmail.com 邮购电话：(010)82316936

北京市媛明印刷厂印装 各地书店经销

*

开本：787×960 1/16 印张：23 字数：515千字

2007年3月第1版 2010年8月第2版 2010年8月第1次印刷 印数：3000册

ISBN 978-7-5124-0187-7 定价：39.00元（含光盘）

前 言

以 ARM 为核心的嵌入式技术逐渐成为我国嵌入式教学的主流，五年前我们就规划了嵌入式方向的系列教材，包括《汇编语言程序设计》、《嵌入式系统接口原理与应用》、《嵌入式 Linux 操作系统原理与应用》、《嵌入式系统原理与应用》。在这个系列教材中，《汇编语言程序设计》是学习嵌入式技术的入门教材。

三年时间过去了，无论是硬件还是开发平台都发生了翻天覆地的变化，相比第一版，第二版主要有如下变化：

1. 对第一版中进行了修订，如图表、文字、公式中出现的一些问题。

2. 增加了 ARM 架构方面新的内容，如 ARMv6、ARMv7 方面的介绍。

3. 重新编写了第 6 章，将原来的 ADS 开发平台换成了 RealView MDK 集成开发环境。MDK 支持的 Cortex - M3 核，是 ARM 公司最新推出的针对微控制器应用的内核，它提供业界领先的高性能和低成本的解决方案，未来几年将成为 MCU 应用的热点和主流。目前，国内只有 MDK 和 RVDS 开发工具可以支持 Cortex - M3 芯片的应用开发。MDK 的编译器与 ADS 1.2 比较，代码更小，性能更高。在代码密度方面，比 ADS 1.2 编译的代码尺寸小 10%；在代码性能方面，比 ADS 1.2 编译的代码性能高 20%。

4. 重新编写了第 7 章，每个实验与具体硬件平台无关，都在 MDK 集成开发环境中通过软件模拟来实现。通过学习者能掌握 MDK 开发环境的使用，掌握 ARM 汇编语言程序设计的编辑、编译、链接和调试过程。

本书在编写的过程中得到了北京航空航天大学何立民教授、北京航空航天大学出版社马广云博士的很多帮助和鼓励。本书的出版也得到了吉林大学珠海学院各级领导的大力支持。我的同事张荣高、纪绪、王艺璇等也为本书做出了很大贡献。在此一并表示诚挚的谢意。本书成书仓促，作者水平有限，错误和不足之处在所难免，谨请读者和同行专家批评指正。

有兴趣的读者，可以发送电子邮件到：wen_sir_125@163.com，与作者进一步交流；也可以发送电子邮件到.xdhydcd5@sina.com，与本书策划编辑进行交流。

文全刚

2010.8 于珠海

目 录

第 1 章

基础知识

本章首先介绍常用的十进制，然后引入各种不同的进位计数制以及它们之间的转换。同时还介绍了二进制数的基本运算，以及数字、字符、汉字的编码、语音编码、差错控制编码。接下来介绍计算机中数据的三种表示形式：原码、反码和补码，并讨论它们的一些基本性质。最后概述与、或、非等基本逻辑运算。

1.1 数制与数制转换

1.1.1 数制的基本概念

所谓数制就是指计数的方法，日常生活中常用的计数方法是十进制。在计算机中为了便于数的存储及物理实现，采用了二进制数。为了便于人们阅读和书写，经常使用八进制和十六进制来表示二进制数。数据无论使用哪种进位制，都涉及两个基本要素：基数(radix)与各数位的“位权”(weight)。为了更好地理解这两个概念，这里借助最常用的十进制来说明这个问题。十进制数有两个主要特点：

① 用 0,1,2,…,9 这十个基本符号表示。

② 遵循“逢十进一”原则。

一般地，任意一个十进制数 N 都可以表示为：

$$N = K_{n-1} \times 10^{n-1} + K_{n-2} \times 10^{n-2} + \cdots + K_1 \times 10^1 + K_0 \times 10^0 +$$

$$K_{-1} \times 10^{-1} + K_{-2} \times 10^{-2} + \cdots + K_{-m} \times 10^{-m} = \sum_{-m}^{i=n-1} K_i \times 10^i \qquad (1.1)$$

在上述十进制中，每一位上允许选用 0,1,2,…,9，共 10 个不同数码中的一个，则称 10 为十进制的基数，每位计满十时向高位进一。因此，一种计数制允许选用基本数字符号(数码)的个数叫基数。常用 R 来表示，在基数为 R 的计数制中，包含 R 个不同的数字符号，每个数位计满 R 就向高位进一，即“逢 R 进一”。

一个数字符号处在不同位时，它所代表的数值是不同的。在上述十进制数中的百分位、十分位、个位、十位、百位、千位数的大小权依次是以 10^{-2},10^{-1},10^0,10^1,10^2,10^3 为单位计算的。

每个数字符号所表示的数值等于该数字符号值乘以一个与数码所在位有关的常数 10^i,这个常数 10^i 叫做“位权”,简称“权”。权的大小是以基数为底,数码所在位置的序号为指数的整数次幂,常用 R^i 表示。注意整数位是从 0 开始计算的。

式(1.1)是十进制按权的展开式。其中 10 称十进制数的基数,i 表示数的某一位,10^i 称该位的权,K_i 表示第 i 位的数码,它可以是 0～9 中的任意一个数,由具体的数 N 确定。m 和 n 为正整数,n 为小数点左边的位数,m 为小数点右边的位数。一个数左移一位相当于乘以 10,右移一位相当于除以 10。

1.1.2 二进制数

式(1.1)可以推广到任意进位计数制。二进制数中只有 0 和 1 两个字符,基数为 2,满足“逢二进一”。权用 2^i 表示,二进制的按权展开式为

$$N = \sum_{-m}^{i=n-1} K_i \times 2^i \tag{1.2}$$

尽管人们习惯使用十进制数,但计算机中却采用二进制数。这是因为二进制数与其他进制数相比,有以下特点:

(1) 数制简单、容易表示

二进制数只有“0”和“1”两种表示符号,这两种状态很容易用电路来实现。如晶体管的导通和截止、电容的充电和放电。在计算机中通常采用电平的“高”、“低”或脉冲的“有”、“无”来分别表示“1”和“0”。这种状态工作可靠,抗干扰能力强。若采用十进制数表示,有 0～9 这十个数码,需要 10 个不同的设备状态来表示。这十个不同的状态用电路来表示就变得很复杂,不容易实现。

(2) 运算规则简单,可以利用逻辑代数进行分析和综合

二进制的运算规则非常简单,所以在计算机中实现二进制运算的电路也大为简化。同时还可以使用逻辑代数对计算机逻辑线路进行分析和综合,便于机器结构的简化。

尽管在计算机内部采用二进制操作,但是对于人们来说,使用二进制并不方便,如书写冗长、阅读不方便。为此,通常采用八进制或十六进制。根据式(1.1),读者可以很容易地掌握八进制和十六进制的表示方法。

对于八进制,$R=8$,K 为 0～7 中的任意一个数,“逢八进一”。权用 8^i 表示,八进制的按权展开式如(1.3)所示。

$$N = \sum_{-m}^{i=n-1} K_i \times 8^i \tag{1.3}$$

对于十六进制,$R=16$,K 为 0～9,A,B,C,D,E,F,共 16 个数码中的任意一个,“逢十六进一”。权用 16^i 表示,十六进制的按权展开式如式(1.4)所示

$$N = \sum_{-m}^{i=n-1} K_i \times 16^i \tag{1.4}$$

综上所述，以上几种进位制有以下共同点：每种进位制都有一个确定的基数 R，每一位的系数 K 有 R 种可能的取值；按“逢 R 进一”方式计数；在混合小数中，一个数左移一位相当于乘以 R，右移一位相当于除以 R。

在计算机中，通常用数字后面跟一个英文字母来表示该数的数制。通常二进制数用 B(binary)、十进制数用 D(decimal)、八进制用 O(octal)、十六进制用 H(hexadecimal)表示。如果省略该字母则一般表示该数为十进制数。有些教材中为了避免“O”与“0”混淆，也用 Q 表示八进制。十六以内的各种数制的值如表 1-1 所列。

表 1-1　十六以内的各种数制对照表

十进制 D	二进制 B	八进制 O	十六进制 H
0	0	0	0
1	1	1	1
2	10	2	2
3	11	3	3
4	100	4	4
5	101	5	5
6	110	6	6
7	111	7	7
8	1000	10	8
9	1001	11	9
10	1010	12	A
11	1011	13	B
12	1100	14	C
13	1101	15	D
14	1110	16	E
15	1111	17	F
16	10000	20	10

1.1.3　十进制和二进制之间的转换

十进制数转换成二进制数时，需要把整数部分与小数部分分别转换，然后再合并起来。其中整数部分的转换方法是：除 2 取余，高位在下。小数部分的转换方法是：乘 2 取整，高位在上。例如：要将十进制数 153.787 5D 转换为二进制数，则首先将该数分为整数部分和小数部分。

1. 整数部分的转换

整数部分的转换采用辗转相除法,用 2 不断去除要转换的十进制数,直到商为 0,将各次计算所得的余数,按最后的余数为最高位,第一位为最低位,从下往上取,依次排列,即得转换结果。计算过程如下:

	余数	
153/2=76	1	低位
76/2=38	0	↑
38/2=19	0	
19/2=9	1	
9/2=4	1	
4/2=2	0	
2/2=1	0	
1/2=0	1	高位

因此,153D=(10011001)B

2. 小数部分的转换

与整数部分转换不同,小数部分采用乘基数取整数的方法,即不断用 2 去乘需要转换的十进制小数,直到满足要求的精度或小数部分等于 0 为止,然后取每次乘积结果的整数部分,以第一次取整为最高位,从上往下依次排列,即可得到转换结果。整个转换过程如下:

	整数部分	
0.7875×2=1.575	1	高位
0.575×2=1.15	1	↑
0.15×2=0.30	0	
0.3×2=0.6	0	
0.6×2=1.2	1	低位

这里精确到小数点后五位,因此,0.7875D=(0.11001)B。综上所述,十进制数 153.7875 转换为二进制数为(10011001.11001)B。

以上方法可以推广到十进制数转换为八进制、十六进制甚至是任意进制数的方法。对于整数部分,转换方法为:除基取余,高位在下。对于小数部分转换方法为:乘基取整,高位在上。有兴趣的读者可以通过一些例子去验证。

1.1.4 二进制和其他进制之间的转换

1. 二进制数转换为十进制数

二进制转换为十进制的方法是:按权展开。例如:

$$(110.01)B = 1\times2^2+1\times2^1+0\times2^0+0\times2^{-1}+1\times2^{-2}=(6.25)D$$

同理，以上方法可以推广为八进制、十六进制甚至是任意进制数转换为十进制数。例如：

对于八进制数：　(175)O＝$1\times8^2+7\times8^1+5\times8^0$＝(125)D

对于十六进制数：(B2C)H＝$11\times16^2+2\times16^1+12\times16^0$＝(2860)D

2. 二进制和八进制、十六进制之间数的转换

由于 $8=2^3$，$16=2^4$，因此二进制与八进制或十六进制之间的转换就很简单。将二进制数从小数点位开始，向左每 3 位产生一个八进制数字，不足 3 位的左边补零，这样得到整数部分的八进制数；向右每 3 位产生一个八进制数字，不足 3 位的右边补 0，得到小数部分的八进制数。同理，将二进制数转换成十六进制数时，只要按每 4 位分割即可。例如：

$(101\ 101.101\ 001)_B$＝(55.51)O

$(0010\ 1101.1010\ 0100)_B$＝(2D.A4)H

很明显，八进制数或十六进制数要转换成二进制数，只需将八进制数或十六进制数分别用对应的 3 位或 4 位二进制数表示即可。

1.2　二进制数的基本运算

二进制运算中基本的运算是加法和减法。利用加法和减法，就可以进行乘法、除法以及其他数值运算。

1. 二进制加法

二进制加法的运算规则为：

0＋0＝0　　1＋0＝1

0＋1＝1　　1＋1＝0　进位 1

例如：若有两数 1101 和 1011 相加，则加法过程为

进　位		1	1	1	
被加数		1	1	0	1
加　数	＋	1	0	1	1
	1	1	0	0	0

可见，两个二进制数相加，每一位有三个数：相加的两个数以及低位的进位。用二进制的加法规则得到本位的和以及向高位的进位。

2. 二进制减法

二进制减法的运算规则为：

0－0＝0　　1－0＝1

1－1＝0　　0－1＝1　借位 1

例如：若有两数 11000100 和 01100101 相减，则减法过程为

```
借  位        1 1 1 1 1 1
被加数      1 1 0 0 0 1 0 0
加  数    - 0 1 1 0 0 1 0 1
          ---------------
            0 1 0 1 1 1 1 1
```

与减法类似,每一位有三个数参加运算:本位的被减数和减数,以及低位来的借位。二进制的乘法和除法运算通过一系列变换,最终可以转换为加法或减法运算,这里不再赘述。

1.3 计算机中的编码

1.3.1 数字的编码

计算机内部采用的是二进制数,但是人们习惯了十进制,因此计算机在输入和输出时,通常仍采用十进制数,只不过它要用二进制编码来表示。这就是二进制编码的十进制数,即 BCD(二～十进制)码,这种编码法分别将每位十进制数字编成 4 位二进制代码,从而用二进制数来表示十进制数,它广泛应用于计算机中。

4 位二进制数可以有 16 种不同的组合,原则上可以任选其中的 10 种作为代码,分别表示十进制中 0～9 这 10 个数字。为了便于记忆和比较直观,最常用的 BCD 码是标准 BCD 码或称 8421 码(这是根据这种表示中各位的权值而定的,其权值与普通的二进制相同)。表 1-2 列出了标准 BCD 码与十进制数字的编码关系。为了避免格式与纯二进制码混淆,常在每 4 位二进制数之间留一空格。这种表示也适合十进制小数。如十进制小数 0.456 可以表示成:0.0100 0101 0110。

表 1-2 8421BCD 码

十进制数	8421BCD 码	十进制数	8421BCD 码
0	0000	8	1000
1	0001	9	1001
2	0010	10	0001 0000
3	0011	11	0001 0001
4	0100	12	0001 0010
5	0101	18	0001 1000
6	0110	62	0110 0010
7	0111	79	0111 1001

把一个十进制数变成它的 8421BCD 码数串,仅对十进制数的每一位单独进行即可。例如将 1986 转换为相应的 8421BCD 码表示,结果为 0001 1001 1000 0110。反转换过程也类似,例