



浙江省高等教育重点建设教材

8XC196KC/KD

十六位单片机原理及应用

道克刚

主编

道克刚 徐军
王玉巧 王东辉

编著

浙江大学出版社

浙江省高等教育重点建设教材

十六位单片机原理及应用

8XC196KC/KD

道克刚 主 编
道克刚 徐 军 编 著
王玉巧 王东辉

TP368.1

3847

Yang Guangjun

7 计 1

浙江大学出版社

内容简介

本书以十六位单片机 8XC196KC/KD 为主,系统地介绍了单片机的运算基础,电路组成,存储器原理和扩展方法,汇编语言程序设计,I/O 接口的组成,工作原理和管理方法。并围绕当前单片机应用较为热门的技术领域列举了多种实例,并分别以程序查询、中断和外设事件服务(PTS)方式作了编程和分析。

本书参考了近年来,单片机发展和应用的最新成果,以及 Intel 公司对 8XC196KC/KD 提供的最新资料,使内容上具有新颖性、实用性。

本书是作为教材编写的,注意了原理的阐明,内容的精炼和深入浅出,理论与实践的结合,典型例题的分析和注释等,每章末尾还列有深化理解课文内容的习题。

本书用于大专院校的《单片机原理和应用》、《微机原理和应用》、《接口技术》等的相关教材,对于从事单片机应用开发的工程技术人员也是一本实用的参考书。

浙江省高等教育重点建设教材

十六位单片机原理及应用

8XC196KC/KD

道克刚 主 编

道克刚 徐 军 编 著

王玉巧 王东辉

责任编辑 龚建勋

* * *

浙江大学出版社出版

(杭州玉古路 20 号 邮政编码 310027)

(E-mail: zupress@mail. hz. zj. cn)

浙江大学出版社电脑排版中心排版

杭州金融管理干部学院印刷厂印刷

浙江省新华书店发行

* * *

787mm×1092mm 16 开 25.25 印张 645 千字

1999 年 3 月第 1 版 1999 年 3 月第 1 次印刷

印数 0001—2060

ISBN 7-308-02099-1/TP·178 定价:38.00 元

目 录

第一章 计算机的运算基础	1
§ 1.1 无符号二进制整数的运算	1
1.1.1 两个 n 位无符号数的加法	1
1.1.2 两个 n 位无符号数的减法	1
1.1.3 计算机用加法器完成减法运算	2
§ 1.2 二进制有符号数的表示和运算规则	4
1.2.1 机器数与真值	4
1.2.2 原码	5
1.2.3 补码	6
1.2.4 反码	10
§ 1.3 带小数点的数——定点和浮点表示法	11
1.3.1 数的定点表示法	11
1.3.2 数的浮点表示法	12
§ 1.4 BCD 码及其运算规则	14
1.4.1 BCD 码的引入	14
1.4.2 BCD 码的加法	15
1.4.3 BCD 码的减法	16
§ 1.5 字符的编码	17
§ 1.6 逻辑运算	18
1.6.1 逻辑或运算	18
1.6.2 逻辑与运算	18
1.6.3 逻辑非运算	19
1.6.4 逻辑异或运算	19
§ 1.7 代码的校验	19
1.7.1 校验编码的基本概念	19
1.7.2 奇偶校验码	20
1.7.3 纵横奇偶冗余校验	21
习题一	21
第二章 计算机电路基础	24
§ 2.1 运算器	24
2.1.1 算术逻辑运算部件	25
2.1.2 寄存器、累加器 ACC、标志寄存器	29
2.1.3 多路开关	29
§ 2.2 控制器	30
2.2.1 程序计数器 PC	30
2.2.2 指令寄存器 IR 和指令译码器	31

2.2.3 微操作控制部件	31
2.2.4 定时部件	32
2.2.5 中断控制部件	32
2.2.6 RISC 和 CISC 计算机	33
习题二	33
第三章 存储器	35
§ 3.1 存储器的分类和指标	35
3.1.1 存储器的分类	35
3.1.2 存储器的主要技术指标	36
§ 3.2 随机访问存储器 RAM	37
3.2.1 静态 RAM(SRAM)	37
3.2.2 动态 RAM(DRAM)	41
§ 3.3 只读存储器 ROM	42
3.3.1 ROM 的存储原理	43
3.3.2 可编程 ROM(PROM)	43
3.3.3 可擦除可编程 ROM(EPROM)	44
3.3.4 电可擦除可编程 ROM(EEPROM)	46
§ 3.4 非易失性 RAM	49
3.4.1 “闪存”的特点	49
3.4.2 通用闪存的结构和编程	49
3.4.3 闪存的应用	52
§ 3.5 存储器和 CPU 的连接	52
3.5.1 地址空间的分配	53
3.5.2 地址译码	54
习题三	60
第四章 8XC196KC/KD 单片机硬件组成	61
§ 4.1 8XC196KC/KD 单片机内部的硬件组成	61
4.1.1 8XC196KC/KD 内部结构	62
4.1.2 8XC196KC/KD 引脚	63
§ 4.2 CPU 和 MCR 的结构和工作原理	66
4.2.1 寄存器算术逻辑单元(RALU)	66
4.2.2 寄存器文件	67
4.2.3 存储控制器	67
§ 4.3 时钟和供电管理	68
4.3.1 内部定时	68
4.3.2 供电管理	69
§ 4.4 存储区的管理	70
4.4.1 地址空间映像图	71
4.4.2 芯片配置寄存器	72
4.4.3 系统总线的时序	77

§ 4.5 应用系统的基本组成	79
4.5.1 一个 8 位总线应用系统	79
4.5.2 系统的复位	81
习题四	84
第五章 指令系统	85
§ 5.1 指令系统基础	85
5.1.1 寄存器编程模型	85
5.1.2 程序状态字 PSW	87
5.1.3 指令格式	89
5.1.4 操作数的类型	93
5.1.5 操作数的寻址方式	95
§ 5.2 指令的分类和传送指令	102
5.2.1 指令的分类	102
5.2.2 正向传送指令	105
5.2.3 反向传送指令	110
5.2.4 带扩展正向传送指令	111
5.2.5 堆栈及栈操作指令	111
§ 5.3 算术运算指令	113
5.3.1 不带进位位的加法与不带借位位的减法指令	113
5.3.2 带进位位的加法和带借位位的减法指令	114
5.3.3 比较指令	114
5.3.4 乘法指令	114
5.3.5 除法指令	115
§ 5.4 逻辑运算指令	116
5.4.1 “与”运算指令	116
5.4.2 “或”运算指令	116
5.4.3 “异或”运算指令	117
§ 5.5 单操作数指令	117
5.5.1 增量与减量指令	117
5.5.2 求反与求补指令	118
5.5.3 清零和扩展指令	118
§ 5.6 移位指令	119
5.6.1 向左移位指令	119
5.6.2 逻辑右移指令	120
5.6.3 算术右移指令	120
5.6.4 规格化指令	121
§ 5.7 调用与转移指令	121
5.7.1 调用与返回指令	121
5.7.2 无条件转移指令	123
5.7.3 条件转移指令	123
5.7.4 位测试转移指令	125
5.7.5 循环控制指令	126

§ 5.8 特殊控制指令	127
5.8.1 标志位操作指令	127
5.8.2 开/关中断指令	127
5.8.3 软件陷阱(或称软件中断)指令	127
5.8.4 空操作指令	127
5.8.5 复位指令	127
§ 5.9 8XC196KC/KD 扩展的指令	128
习题五	129
第六章 MCS-96 汇编语言程序设计	132
§ 6.1 MCS-96 汇编语言简介	132
6.1.1 汇编语言语句格式	132
6.1.2 MCS-96 汇编语言的伪指令	133
§ 6.2 定点数运算程序	135
6.2.1 无符号定点数运算	135
6.2.2 有符号定点数运算	139
§ 6.3 浮点数运算程序	140
§ 6.4 数制转换程序	145
6.4.1 整数十翻二	146
6.4.2 整数二翻十	146
§ 6.5 数据处理程序	147
6.5.1 排序	147
6.5.2 求平均值	148
6.5.3 代码转换和 BCD 码运算	149
6.5.4 表格处理	150
§ 6.6 子程序设计	151
6.6.1 主-子程序的参数传送	151
6.6.2 子程序的现场保护	154
6.6.3 子程序嵌套	154
习题六	155
第七章 输入/输出(I/O)接口和中断方式	162
§ 7.1 I/O 接口的基础知识	162
7.1.1 I/O 接口的主要功能	162
7.1.2 I/O 接口的编址方式	165
7.1.3 I/O 数据传送的控制方式	167
7.1.4 8XC196KC/KD 单片机的 I/O 接口及扩展方法	170
§ 7.2 中断的概念	177
7.2.1 引进中断的优点	177
7.2.2 中断源的种类、中断登记、中断屏蔽	177
7.2.3 中断处理过程	178
§ 7.3 8XC196KC/KD 单片机中断系统	180

7.3.1	中断系统的组成	180
7.3.2	中断处理过程	186
7.3.3	中断响应时间	188
7.3.4	中断系统编程实例	189
7.3.5	中断方式应用小结	194
7.3.6	垂直窗口技术用于中断的现场保护	196
7.3.7	采用 PTS 提高中断处理的速度	198
习题七		204
第八章 并行输入输出接口		206
§ 8.1	简单通用的集成电路组成的并行接口	206
8.1.1	用触发器构成的并行输出接口	206
8.1.2	采用三态门电路的并行输入接口	207
§ 8.2	可编程并行输入输出接口	208
8.2.1	概述	209
8.2.2	8255A 控制字和状态字	210
8.2.3	并行传送模式 0	212
8.2.4	模式 1	216
§ 8.3	8XC196KC/KD 片内的并行接口	218
8.3.1	输入口	218
8.3.2	输出口	219
8.3.3	准双向口	220
8.3.4	总线口	221
§ 8.4	并行接口的应用	222
8.4.1	并行接口在键盘显示中的应用	222
8.4.2	并行接口在液晶显示控制方面的应用	239
8.4.3	并行接口用于打印机控制	250
8.4.4	并行口对步进电机的控制	252
习题八		255
第九章 8XC196KC/KD 定时器		263
§ 9.1	定时器的工作原理	263
9.1.1	定时器 T1	263
9.1.2	定时器 T2	264
9.1.3	监视定时器 WDT	265
§ 9.2	定时器的应用	266
9.2.1	定时器 T1 的应用	266
9.2.2	定时器 T2 的应用	268
习题九		269
第十章 8XC196KC/KD 高速输出 HSO 的原理及应用		271
§ 10.1	HSO 的硬件结构及 CAM 的工作原理	271

§ 10.2 HSO 的控制	273
10.2.1 CAM 的预置	273
10.2.2 HSO 命令寄存器	273
10.2.3 时间寄存器	274
10.2.4 状态寄存器	275
§ 10.3 CPU 对 HSO 的管理方法	276
10.3.1 HSO 的查询工作方式	276
10.3.2 HSO 的中断工作方式	277
10.3.3 HSO 的 PTS 工作方式	277
§ 10.4 HSO 的应用	278
10.4.1 定时功能的应用	278
10.4.2 产生脉冲信号波形	279
10.4.3 用 HSO 模拟串行发送端口	282
10.4.4 HSO 生成 PWM 波形	283
10.4.5 HSO 的 PTS 工作方式的应用	284
10.4.6 HSO 应用中要注意的几个问题	285
习题十	285
第十一章 8XC196KC/KD 高速输入 HSI 的原理及应用	289
§ 11.1 HSI 的硬件结构及工作原理	289
11.1.1 输入的选择和功能	289
11.1.2 跳变检测器和方式选择逻辑	290
11.1.3 FIFO 和保持寄存器	291
11.1.4 HSI STATUS 和 HSI TIME 寄存器	291
§ 11.2 HSI 的管理	292
11.2.1 HSI 初始化	292
11.2.2 HSI 的程序查询管理方式	292
11.2.3 HSI 的中断管理方式	293
11.2.4 HSI 的 PTS 工作方式	293
§ 11.3 HSI 的应用	294
11.3.1 作一般输入端口用	294
11.3.2 测量脉冲参数	294
11.3.3 HSI 模拟串行输入端口	296
11.3.4 HSI PTS 工作方式的应用	297
习题十一	298
第十二章 A/D 和 D/A 接口原理及应用	301
§ 12.1 A/D 接口的设计要点	301
12.1.1 A/D 转换器的字长选择	301
12.1.2 A/D 采样的速度	301
12.1.3 采样保持器	302
12.1.4 A/D 的分类	303
12.1.5 数据采集系统的结构	303

§ 12.2	A/D 转换器和微机的接口	304
12.2.1	AD1674 的内部结构和使用方法	304
12.2.2	AD1674 总线扩展方式	307
12.2.3	AD1674 并行接口扩展方式	309
§ 12.3	8XC196KC/KD 单片机内集成的 A/D 功能模块	311
12.3.1	片内 A/D 的结构	311
12.3.2	片内 A/D 的控制	313
12.3.3	片内 A/D 的管理方法——查询、中断、PTS	315
12.3.4	片内 A/D 转换器的应用	317
§ 12.4	D/A 转换器的基本原理	320
12.4.1	D/A 转换原理	320
12.4.2	D/A 转换器的主要指标	321
12.4.3	8 位 D/A 转换器 DAC0832 的介绍	322
12.4.4	DAC0832 与单片机的接口	322
§ 12.5	8XC196KC/KD 片内的 D/A 转换器	325
12.5.1	PWM 波发生器及其工作原理	325
12.5.2	PWM(D/A)的输出	326
12.5.3	PWM 有关的控制寄存器	327
12.5.4	PWM 的应用程序	328
习题十二		330
第十三章	串行通信接口及应用	334
§ 13.1	串行通信的基本知识	334
13.1.1	串行通信的分类	334
13.1.2	串行通信的物理连接	335
13.1.3	串行通信接口和有关标准	336
13.1.4	串行通信的三种工作方式	337
§ 13.2	8XC196KC/KD 的串行通信接口	339
13.2.1	四种工作方式	339
13.2.2	串行通信接口的寄存器	341
13.2.3	串行口的管理	344
§ 13.3	串行通信接口中的应用	348
13.3.1	扩展的显示和键盘接口	348
13.3.2	串行口的电平转换电路	350
13.3.3	串行口自检	351
13.3.4	双机通信	352
13.3.5	带掉电(PD)工作方式的采样终端	354
13.3.6	主从式多机通信	355
§ 13.4	单片机和 PC 机的通信	359
13.4.1	单片机串行接口的扩展	359
13.4.2	用在单片机中的 Xmodem 协议	361
习题十三		363

第十四章 单片机应用系统的设计和安全措施.....	367
§ 14.1 单片机应用系统的设计.....	368
14.1.1 总体设计	368
14.1.2 抗干扰措施	369
§ 14.2 8XC196KC/KD 编程方式和安全措施.....	375
14.2.1 编程的基本方式和控制方法	375
14.2.2 片内程序区的安全措施和编程方式	377
14.2.3 自动编程方式及编程算法	381
14.2.4 运行一定时编程方式	383
习题十四.....	384
附录 8XC196KC/KD 速查表.....	385
参考文献.....	392

先运行 A96 B TRY↓
TMSD TRY↓

第一章 计算机的运算基础

电子数字计算机是一种对数字和代码进行运算和处理的电子设备。由“0”和“1”表示的二进制码是电子数字计算机唯一能识别的代码。所有需要计算机加以处理的数、字母、符号、文字、图像都必须采用二进制编码表示。计算机常处理的数据和代码包括：

1. 无符号二进制整数；
2. 带符号的二进制数；
3. 带小数点的二进制数；
4. 十进制数的二进制编码(BCD 码)；
5. 二进制编码的字母、符号、文字等；
6. 逻辑代码。

本章主要介绍这 6 类数和码的表示方法以及它们的运算和处理规则。

§ 1.1 无符号二进制整数的运算

无符号二进制数是指 n 位字长全部用于表示数值的数，它可表示 2^n 个整数。例如一个 8 位无符号数，可表示的数值范围为 $00000000B \sim 11111111B$ ，即 $0 \sim 255$ 共 256 个数值。

1.1.1 两个 n 位无符号数的加法

例 1-1 已知 $a=11010011$, $b=11100001$ ，完成 $a+b$ 的操作如下：

$$a = 11010011 \quad (211)$$

$$+b = 11100001 \quad (225)$$

$$\boxed{1}10110100 \quad (\text{结果 } 180)$$

溢出 (出错现象) $\rightarrow$ 解决方法: 变 8 位为 16 位数

..... 有向高位的进位

对于 n 位字长的运算器，通常设定了一个进位标志位 C (Carry)。若加的结果有向 $n+1$ 位进位，则 C 置“1”；若无进位，则 C 置“0”。 C 记录了两个 n 位无符号二进制数相加后产生溢出的结果。在多字长的二进制数相加时 (也称作高精度加法) 低位字 (或字节) 的进位位 C 可以参加到高位字的累加中，实现带低位进位的加法。

1.1.2 两个 n 位无符号数的减法

例 1-2 已知 $a=11010011$, $b=11100001$ ，完成 $a-b$ 的操作如下：

$$\begin{array}{r} a = 11010011 \quad (211) \\ - b = 11100001 \quad (225) \end{array}$$

$$\boxed{1} \quad 11110010 \quad (\text{结果 } 242)$$

.....有向高位的进位

计算机的运算器做减法时,遇到不够减的情况,总是向高位借一位来完成减的运算。借来这一位的值是 2^n , (上例中, $n=8$, 相当于借了一个 256 的数值)。这种安排为多字长的减法(即高精度运算)提供了方便。它的借位来自高位字,后面高位字的减操作就应当带低位的借位减。把低位的借位扣除。为了简化电路,借位信息也是由 C 标志位来传递的。在加法之后 $C=1$ 表示有进位。而在减法之后 $C=1$ 表示有借位。(对于 Intel 8086、Intel MCS-51 系列等大多数微机是这样定义的,但对 MCS-96 系列,减法之后 $C=0$ 才表示有借位,这在后面还要说明。)

上述无符号数的减法结果和人们的习惯不同,通常 $211-225$,在不够减时应当得到 -14。但是这里进行的是 n 位无符号数的减法,它的结果也应当是 n 位无符号数,这里的负号在 n 位中是无法表示的。如果运算的字长只有 n 位,没有高位字可以借位的情况,同时又需要把减的结果按人们的习惯通过输出,设备输出,上述结果还需要再作一次减法,即扣除借来这位的数值 (2^n)。本例中则需要再减去 256。

$$\begin{aligned} a-b &= 242-256 \\ &= -(256-242) \\ &= -(14) \end{aligned}$$

通过输出设备先输出符号“-”,再输出绝对值 14。

1.1.3 计算机用加法器完成减法运算

计算机是用加法器完成减法运算的,这样可简化运算器的电路。为了说明这点,首先要了解有模运算的概念和性质。

我们把一个计量器的容量称为模或模数(mod),记为 M 或 $\text{mod } M$ 。例如:一个 n 位二进制计数器,它的容量为 2^n ,所以它的模为 2^n (即可表示 2^n 个不同的数)。又如时钟可表示 12 个钟点,它的模为 12。

模具有这样的性质,当模为 2^n 时, 2^n 和 0 的表示形式相同。例如一个 n 位二进制计数器,可以从 0 计数到 2^n-1 ,如果再加 1,计数器就变成了零,所以 2^n 和 0 在 n 位计数器中的表示形式是一样的。同样时钟的零点和 12 点在钟表上的表示形式也是相同的。

如果有两个整数 a 和 b ,当用某一个正整数 M 去除,所得余数相等时,则称 a 和 b 对模 M 是同余的。

当 a 和 b 对 M 同余时就称 a, b 在以 M 为模时是相等的,记为:

$$a \equiv b \pmod{M}$$

例如: $a=16, b=4$, 若模为 12, 则 16 和 4 在以 12 为模时是同余的:

$$16 \equiv 4 \pmod{12}$$

又如:对于 1 位十进制计数器,模为 10, 若 $a=12, b=2$ 则是同余的:

$$12 \equiv 2 \pmod{10}$$

由同余的概念可以得出:

$$M+a=a \pmod{M}$$

$$2M+a=a \pmod{M}$$

因此,当 a 为负数时,如 $a=-3$,有

$$10+(-3)=-3 \pmod{10} \quad \text{即} \quad 7=-3 \pmod{10}$$

$$12+(-3)=-3 \pmod{12} \quad \text{即} \quad 9=-3 \pmod{12}$$

1 位 10 进制计数器的 7 可看成 10 扣除 3, 钟表上的 9 点可以看成 12 点缺 3 个小时。这样,以 10 为模时负数 (-3) 可以转化为正数 $(+7)$, 以 12 为模时, (-3) 可以转化为正数 $(+9)$ 。

例如: $5-3=5+(-3)$

$$=5+(10-3) \pmod{10}$$

$$=5+7 \pmod{10}$$

$$=10+2=2 \pmod{10}$$

此处 10 自动丢失其值正好等于模 10

在有模运算中,减法可以转换成加法来完成。

$$\text{又如: } 6-3=6+(12-3) \pmod{12}$$

$$=6+9 \pmod{12}$$

$$=12+3=3 \pmod{12}$$

此处 12 自动丢失,其值正好等于模 12

此例实际是一个钟表对时的例子。假定现在标准时间是 3 点整,而一只钟却指在 6 点整。为了校准时钟,可以把时针从 6 倒拨 3 格,即 $6-3=3$, 表明时针最后停在 3 点上。但是,有的钟表不允许倒拨时针,只允许顺时针转动,这时可以把时针从 6 顺拨 9 格到达 3 点整。这是因为时针顺拨时,当到达 12 点就从 0 重新开始,相当于自动丢失一个数字 12。

电子计算机中,运算器字长是有限的,都属于有模运算。字长为 n 的运算器,其模为 2^n 。因此,两个无符号数相减,也可以利用有模运算的一些性质,用加法来实现:

$$\text{如} \quad a-b=M+(a-b) \pmod{M}$$

$$=a+(M-b)$$

$$=a+[b]_{\text{求补}}$$

$$\text{其中} \quad [b]_{\text{求补}}=M-b$$

模减去一个数的操作称作对该数求补。

上例中, b 可以是 0 到 2^n-1 中的任何数。计算机中减法运算一律采用减数求补再和被减数相加来完成。 n 位字长的二进制数求补就是该数逐位求反,再在末位加 1 (简称求反加 1)。它和用模 2^n 减去减数的方法获得的结果完全相同。在数字电路中,对数的各位取反操作很容易实现。因此只有二进制在求补操作中完全避开了减法,两数相减完全用加法器来完成。

计算机在无符号数减法中会遇到下述两种情况。

$a \geq b$ 或 $a < b$ 。我们分别讨论它们的运算结果。

(1) 若 $a \geq b$, 则 $a-b$ 无借位, 由上所述 $a-b=a+[b]_{\text{求补}}=a+[M-b]$

$$=M+[a-b] \pmod{M}$$

等式右边一定大于模 M , 也就是在作加法时, 一定有进位 $C=1$ 。因此 MCS-96 逻辑上规定, 无符号数减法, 采用减数求补与被减数相加完成时, $C=1$ 有进位表示够减, 无借位。

例 1-3 已知: $a=11000000(192)$ $b=00000010(2)$ $[b]_{\text{求补}}=11111110$

$$\begin{array}{r}
 a - b = a + [b]_{\text{求补}} \\
 a = 11000000 \quad (192) \\
 + [b]_{\text{求补}} = 11111110 \quad [2]_{\text{求补}} \\
 \hline
 \boxed{1} \quad 10111110 \quad (\text{结果: } 190)
 \end{array}$$

↑
有进位表示无错位

一个数从高位起逐位求反遇到最后一个1停止求反,这就是手算求补(求反加1)的简便方法。

(2) 若 $a < b$, 则 $a - b$ 有借位, 由前所述

$$\begin{aligned}
 a - b &= a + [b]_{\text{求补}} = a + (M - b) \\
 &= M + (a - b) \pmod{M}
 \end{aligned}$$

其中 $a - b$ 一定是负数, 等式右边一定小于模 M , 在做加法操作时一定无进位 $C = 0$ 。因此 MCS-96 系列逻辑上规定无符号数减法采用减数求补与被减数相加完成时, $C = 0$ 无进位表示有借位。

例 1-4 已知: $a = 00000010 \quad (2) \quad b = 11000000 \quad (192) \quad [b]_{\text{求补}} = 01000000$
 $a = 00000010 \quad (2)$

$$\begin{array}{r}
 + [b]_{\text{求补}} = 01000000 \quad [192]_{\text{求补}} \\
 \hline
 c = \boxed{0} \quad 01000010 \quad (\text{结果: } 66)
 \end{array}$$

↑
无进位表示有借位

结果表示, 被减数不够减时, 是向高位借一位完成减法的, 借来的这位数值是 256, 所以上例中减的结果是 66。

同样如果这已是最后结果并需要按人们的习惯输出的话, 只要先判断是否有借位(进位位等于 0), 若有, 则对运算结果求补, 获得输出结果的绝对值(完成扣除借位数值的操作)送到输出设备, 并在该数的前面加一个负号, 若无借位则直接把结果输出。

上例中 $a - b = 01000010 \quad (66)$

标志位 $c = 0$ 表示有借位

扣除借位值的操作就是把上述结果减去 256, 即:

$$\begin{aligned}
 01000010 - M &\pmod{M = 256} \\
 &= -(M - 01000010) \\
 &= -[01000010]_{\text{求补}}
 \end{aligned}$$

也就是对括号内的八位二进制数求反加 1, 最后输出的是 $-10111110 (-190)$ 。

§ 1.2 二进制有符号数的表示和运算规则

本节主要介绍计算机中二进制带符号数的表示方法和运算规则。

1.2.1 机器数与真值

通常数的正负是用符号“+”、“-”来表示的。

在计算机中, 数是存放在一组存储单元组成的寄存器中, 二进制数码 1 和 0 是由寄存器单元的两种不同的状态(比如高、低电平)来表示的。对于正号“+”或负号“-”, 也只能用这两种

不同的状态来区别。因此,数的符号在计算机中也数码化了。通常规定在数的前面增设一位符号位,正数符号位用“0”表示,负数符号位用“1”表示。

若以 8 位字长的整数单元为例,设有:

$$N_1 = +1001100$$

$$N_2 = -1001100$$

则 N_1 和 N_2 在计算机中表示为

$$N_1: 01001100$$

$$N_2: 11001100$$

为了区别原来的数与它在计算机中的表示形式,我们将已经数码化了的带符号数称为机器数,而把原来的带符号数称为真值。上面例子提到的 $N_1 = +1001100$ 、 $N_2 = -1001100$ 为真值,其在计算机中的表示 01001100 和 11001100 为机器数。其中符号位除外的部分称为尾数。

在计算机中,带符号的机器数常用符号—原码、符号—补码和符号—反码三种表示法表示。简称原码、补码和反码。下面分别进行讨论。

1.2.2 原码

一、原码(true form)表示法的定义

在 n 位字长的机器数中,原码可由下式定义:

$$[X]_{\text{原}} = \begin{cases} X & (0 \leq X \leq 2^{n-1}-1, \text{即正数}) \\ 2^{n-1} - X & (-2^{n-1}+1 \leq X \leq 0, \text{即负数}) \end{cases}$$

其中, X 为真值, $[X]_{\text{原}}$ 为原码表示的机器数。 X 为正数时, $[X]_{\text{原}}$ 最高位为 0, 表示符号为“+”; X 为负数时, $[X]_{\text{原}}$ 最高位为 1, 表示符号为“-”。实际上原码表示的机器数就是把真值 X 前面的符号用 0(对“+”号)或用 1(对“-”号)替换来完成的。以 $n=8$ 为例:

如 $X = +1001000$ 则 $[X]_{\text{原}} = 01001000$

$X = -1001000$ 则 $[X]_{\text{原}} = 11001000$

在原码表示法中有两个 0

原码表示法

$$\begin{cases} [+0]_{\text{原}} = 00000000 \\ [-0]_{\text{原}} = 10000000 \end{cases}$$

计算机将这两种情况都作为 0 来处理。

根据以上定义可知 8 位二进制有符号数的数值范围为 $+127 \sim -127$ 。

二、原码表示法的运算规则

(1) 两数相加

若符号相同,则尾数相加,符号不变;若符号不同,则先判断哪个尾数值大,然后大数尾数减去小数尾数,结果的符号取大数的符号。

(2) 两数相减

若符号相异,则尾数相加,符号取被减数的符号;若符号相同,则先判断哪个尾数值大,然后大数尾数减去小数尾数。结果的符号这样定:若大数是被减数则结果的符号和两数同号,若大数是减数则结果的符号和两数异号。

(3) 乘法

两数的尾数进行乘除运算,它们的积或商的符号为两数符号的异或运算,即:两符号相同,结果的符号为 0;两符号相异,结果的符号为 1。

原码表示法的乘除规则较简便,而加、减规则却很繁琐,这意味着实现它的运算器线路会复杂化。

1.2.3 补码

一、补码(two's complement)表示法的定义

在 n 位字长的机器数中

$$[X]_{\text{补}} = 2^n + X = \begin{cases} X & 0 \leq X \leq 2^{n-1} - 1 \\ 2^n + X & (-2^{n-1} \leq X \leq -1) \end{cases} \pmod{2^n}$$

式中, X 是真值, $[X]_{\text{补}}$ 是补码表示的机器数。

当 X 为正数时, $[X]_{\text{补}}$ 机器数就是真值 X 本身, 其最高位为 0, 它代表了符号“+”。

当 X 为负数时, $2^n + X = 2^n - |X|$ (模 $M = 2^n$), 右式就是对 X 的绝对值 $|X|$ 求补的操作。

$\therefore |X|$ 的范围是 $1 \leq |X| \leq 2^{n-1}$

$\therefore$ 机器数 $[X]_{\text{补}}$ 表示的负数, 其最高位是 1, 它代表了符号“-”。

以字长 n 取 8 为例:

若 $X = +1001101$ (+77)

则 $[X]_{\text{补}} = 01001101$

若 $X = -1001101$ (-77)

则 $[X]_{\text{补}} = 100000000 - 1001101 \pmod{2^8}$

$$= 10110011$$

补码表示法中只有一个机器数表示数值 0, $[0]_{\text{补}} = 00000000$ 。

补码表示法比原码表示法增加了一个 -128 的机器数

$$[-128]_{\text{补}} = 10000000$$

这样, 补码表示法中, 8 位二进制的 256 个编码得以充分的利用, 这一点比原码表示法要优越。在计算机中, 有符号数一般都以补码的形式存放和处理。

这里要指出, $[X]_{\text{补}}$ 和 $[X]_{\text{求补}}$ 是两个不同的概念, 前者表示 X 的机器数是用补码表示的; 后者表示完成模 M 减 X 的操作。

根据定义: $[X]_{\text{补}} = M + X \pmod{M}$

在补码表示的有符号数中 $[-X]_{\text{补}} = M - X = [X]_{\text{求补}} \pmod{M}$

求补即求 $-X$ 的补码, 因此在补码表示的有符号数中, 求补也称为求负。

2. 补码的性质

性质 1: $[X+Y]_{\text{补}} = [X]_{\text{补}} + [Y]_{\text{补}} \pmod{M=2^n}$

性质 2: $[X-Y]_{\text{补}} = [X]_{\text{补}} - [Y]_{\text{补}} \pmod{M=2^n}$

性质 3: $[X-Y]_{\text{补}} = [X]_{\text{补}} + [-Y]_{\text{补}} \pmod{M=2^n}$

性质 1 证明:

等式左边 $= M + (X+Y)$ (根据定义)

$= (M+X) + (M+Y) \pmod{M}$

$= [X]_{\text{补}} + [Y]_{\text{补}}$ (根据定义)

$=$ 等式右边

性质 2 证明:

等式左边 $= M + (X-Y)$ (根据定义)