MCS－51/196 单片机浮点程序和实用程序

张克彦　编著

北京航空航天大学出版社

http：//www. buaapress. com. cn

内 容 简 介

本书结合 MCS－51/196 系列单片机软件资源的特点，设计了 MCS－51/196 浮点程序库以及线性拟合、线性内插、断电保护、CRC 校验等很有参考价值的实用程序，并提供了详实程序清单及说明；给出了严格按照 IEEE 标准判断浮点运算溢出的方法；采取优化数学模型和措施，显著提高了单片机（尤其对 MCS－196 系列单片机）的浮点运算速度。

本书也介绍了浮点数、浮点运算等基本概念，并有演示实例，有助于初学者入门。

本书中的程序可在 http://www.buaapress.com.cn 中下载。

本书可作为微机应用工程技术人员的设计参考书或大专院校的教学参考书。

图书在版编目(CIP)数据

MCS－51/196 单片机浮点程序和实用程序/张克彦编著.
—北京：北京航空航天大学出版社，2001.10
ISBN 7－81077－102－7

Ⅰ.M… Ⅱ.张… Ⅲ.①单片微型计算机，MCS－51/196—浮点程序包②单片微型计算机，MCS－51/196—应用程序 Ⅳ.TP368.1

中国版本图书馆 CIP 数据核字(2001)第 066756 号

MCS－51/196 单片机浮点程序和实用程序

张克彦 编著

责任编辑 胡晓柏

*

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(100083) 发行部电话：(010)82317024 传真：(010)82328026

http://www.buaapress.com.cn

E-mail：pressell@publica.bj.cninfo.net

北京市朝阳区科普印刷厂印刷 各地书店经销

开本：787×1092 1/16 印张：13.5 字数：346 千字

2001 年 10 月第 1 版 2001 年 10 月第 1 次印刷 印数：5000 册

ISBN 7－81077－102－7/TP·053 定价：21.00 元

前　言

MCS－51/196 系列单片机以其众多各具特色的产品，成为功能强大、应用广泛的两种系列单片机，目前在工业过程控制、复杂电机控制、航天设备检测、智能仪器仪表、电子电力工程等方面起着主力军或重要角色作用。因此，MCS－51/196 单片机浮点运算和实用程序设计，也成为单片机应用的一个主要内容。

本书内容主要分为两部分：浮点程序和实用程序。浮点程序中主要讲述基本概念，按 IEEE 浮点数标准设计浮点运算、函数计算和数制转换等子程序的方法。结合单片机资源特点，对主要子程序进行优化设计，显著提高其运行速度。书中重要计算都有公式推导、误差估算，并附有流程图。有严格按照 IEEE 标准检测运算溢出的方法。实用程序部分主要作为浮点程序的应用和补充。该部分包括用最小二乘法实现直线拟合、线性内插、滑动平均、断电保护、循环冗余检测、用软件实现串口功能等等众多具参考价值的程序。

全书分为五章，第 1 章介绍浮点数、浮点运算、浮点程序库等基本概念；第 2 章介绍结合 MCS－51 单片机资源特点设计 MCS－51 浮点程序库的方法；第 3 章介绍 MCS－51 单片机实用程序的设计方法；第 4 章介绍结合 MCS－196 单片机资源特点设计 MCS－196 浮点程序库的方法；第 5 章介绍 MCS－196 单片机实用程序设计的实例。

本书中对程序进行优化的方法也适用于其他类型单片机。

本书以 MCS－51 单片机浮点程序和实用程序设计为主，在 MCS－196 单片机部分中与其内容相同的章节不再细述，可参阅前者的对应部分。

本书可作为单片机应用工程技术人员的设计参考书或大专院校的教学参考书。

本书由北京航空航天大学何立民教授审阅，成书过程中得到公司同仁大力相助，在此表示感谢。

欢迎读者对书中错误批评、指正。

编著者 2001 年 4 月于珠海

通讯地址：珠海市南屏科技工业园虹达路 3 号

珠海志美电子有限公司

邮　　编：519060

目　录

第 1 章　基本概念

1.1　浮点数 …… 1
1.1.1　阶码的选择　移码 …… 1
1.1.2　浮点数格式 …… 2
1.1.3　移码的特点 …… 4
1.1.4　浮点数的规格化 …… 5
1.1.5　对　阶 …… 5
1.1.6　溢　出 …… 5
1.1.7　浮点运算中尾数的处理 …… 6
1.1.8　浮点数的局限性 …… 7
1.2　浮点程序库 …… 7
1.2.1　浮点程序库的结构 …… 7
1.2.2　浮点程序库的优化 …… 7

第 2 章　MCS－51 浮点程序库

2.1　MCS－51 系列单片机软件资源特点及 MCS－51 浮点程序库的特点 …… 9
2.1.1　MCS－51 系列单片机软件资源的特点 …… 9
2.1.2　MCS－51 浮点程序库的特点 …… 9
2.2　基本运算子程序的设计方法 …… 11
2.2.1　支持基本运算的辅助子程序 …… 11
2.2.2　浮点数比较大小子程序 FPCP 的设计方法 …… 14
2.2.3　浮点加法子程序 FPAD 的设计方法 …… 15
2.2.4　浮点减法子程序 FPSU 的设计方法 …… 15
2.2.5　浮点乘法子程序 FPMU 的设计方法 …… 18
2.2.6　浮点除法子程序 FPDI 的设计方法 …… 21
2.2.7　浮点数模拟手算开平方子程序 FPSQ 的设计方法 …… 23
2.2.8　浮点数牛顿迭代开平方子程序 FSQR 的设计方法 …… 26
2.2.9　基本运算子程序的演示程序 …… 27
2.3　函数计算子程序的设计方法 …… 29
2.3.1　函数计算子程序的设计总则 …… 29
2.3.2　函数计算子程序的辅助子程序 …… 29
2.3.3　用荷纳计算法计算多项式值子程序 FPLN1 和 FPLN2 …… 33
2.3.4　对数函数子程序 LNX 的设计方法及截断误差的估算 …… 35

2.3.5 指数函数子程序 EXP 的设计方法和截断误差的估算 …… 40
2.3.6 正弦函数子程序 SINX 的设计方法及截断误差的估算 …… 44
2.3.7 反正弦函数子程序 ASINX 的设计方法及截断误差的估算 …… 47
2.3.8 函数计算子程序的演示程序 …… 52
2.3.9 阶乘子程序 NP 的设计方法 …… 53
2.4 数制转换子程序的设计方法 …… 54
2.4.1 定点数制转换 …… 54
2.4.2 浮点数制转换 …… 57

第 3 章 MCS－51 系列单片机实用程序

3.1 浮点程序库的应用程序和多字节定点运算子程序 …… 65
3.1.1 拟合直线程序 …… 65
3.1.2 多字节定点乘、除法子程序 …… 69
3.1.3 定点整数模拟手算开平方子程序 INTSQR …… 73
3.1.4 滑动平均子程序 SLPAV …… 74
3.1.5 多字节压缩 BCD 码减法子程序 BCDSB …… 76
3.2 查表(子)程序 …… 78
3.2.1 线性内插计算子程序 CHETA …… 78
3.2.2 功能数据表格项目浏览、查找、修改程序 …… 82
3.3 输入输出子程序 …… 88
3.3.1 处理 ASCII 码浮点数子程序 ACUM …… 88
3.3.2 时钟/日历芯片读写程序——OKI MSM 62×42B 时钟/日历芯片的应用 … 98
3.3.3 模数转换器 AD7701 的应用 …… 102
3.3.4 计算键值——LED 显示管理子程序 DSPA …… 104
3.3.5 键入数字序列左移处理子程序 LFDD8 …… 108
3.3.6 双键配合输入数字子程序 KEYIN …… 110
3.3.7 精确定时程序及其应用 …… 111
3.3.8 通用宽行打印机检测及打印子程序 PPRNT …… 115
3.3.9 步进电机控制(驱动)程序 SPDRV1 和 SPDRV2 …… 117
3.4 通讯程序 …… 120
3.4.1 串口中断发送/接收程序 …… 120
3.4.2 用外部中断配合查询接收串行数据 …… 122
3.4.3 以定时器和输出口配合用中断方式发送 ASCII 码字符串 …… 125
3.4.4 高波特率发送数据程序 …… 128
3.5 可靠性方面的实用程序 …… 130
3.5.1 断电保护程序 …… 131
3.5.2 循环冗余检测(CRC)子程序 CRCST …… 136
3.6 格莱码与二进制数间的转换 软件伪随机序列发生器 …… 138
3.6.1 格莱码翻为二进制数子程序 GTOB8 和 GTOB9 …… 138

3.6.2　二进制数翻为格莱码子程序 BTOG8 …… 139
3.6.3　伪随机序列发生器　通讯加密 …… 140

第 4 章　MCS－196 浮点程序库

4.1　MCS－196 系列单片机软件资源的特点及 MCS－196 浮点程序库的特点 …… 141
4.1.1　MCS－196 浮点程序库在设计上与 MCS－51 浮点程序库的主要区别 …… 143
4.1.2　MCS－196 浮点程序库的特点 …… 143
4.2　基本运算子程序的设计方法 …… 145
4.2.1　浮点数比较大小子程序 FPCPR 的设计方法 …… 145
4.2.2　浮点加法子程序 FPADD 的设计方法 …… 146
4.2.3　浮点减法子程序 FPSUB 的设计方法 …… 146
4.2.4　浮点乘法子程序 FPMUL 的设计方法 …… 147
4.2.5　浮点除法子程序 FPDIV 的设计方法 …… 150
4.2.6　快速浮点除法子程序 FPDI2 的设计方法 …… 154
4.2.7　浮点数模拟手算开平方子程序 FPSQ 的设计方法 …… 156
4.2.8　浮点数快速牛顿迭代开平方子程序 FSQR 的设计方法 …… 157
4.2.9　基本运算子程序的演示程序 DMST 的说明 …… 161
4.3　函数计算子程序的设计方法 …… 162
4.3.1　支持函数计算的辅助子程序 …… 162
4.3.2　函数计算子程序的设计方法 …… 165
4.4　数制转换子程序的设计方法 …… 174
4.4.1　定点数制转换 …… 174
4.4.2　浮点数制转换 …… 181
4.4.3　二进制浮点数快速翻为十进制定点数子程序 FBTD 的设计方法 …… 185

第 5 章　MCS－196 系列单片机实用程序

5.1　线性内插计算子程序 CHETA …… 187
5.2　多功能键盘管理——显示程序 …… 189
5.3　精确定时程序 …… 193
5.4　用软件定时器 0 定时中断发送 ASCII 码字串程序 …… 194
5.5　用高速输出器件 HSO_0 中断发送 ASCII 码字串程序 …… 196
5.6　用高速输入器件 HSI_0 和软件定时器 1 中断接收 ASCII 码字串程序 …… 197
5.7　循环冗余检测子程序 CRC0B 和 CRC0W …… 201
5.8　格莱码与二进制数相互转换子程序 GB16 和 BG16 …… 202

参考文献

第 1 章　基本概念

本章主要介绍浮点数格式、浮点运算以及浮点程序库组成等基本概念，提出浮点程序库优化条件。为基本运算函数计算和数制转换等子程序的设计做好准备工作。

1.1　浮点数

人们研制电子计算机的初衷就是为了用于科学计算。时至今日，尽管现在单片机应用领域宽广、色彩缤纷，但复杂计算仍为不可或缺的内容。

针对定点数不能胜任复杂计算的缺点，人们在实践中约定了不同格式、不同精度的浮点数，实现了浮点运算。

因为计算机只能识别二进制数，完成二进制数的计算，所以我们所说的浮点数一般都是指二进制浮点数。与定点数相比，浮点数能较好兼顾表达数值范围和精度范围，能简捷地表示出很大或很小的数值。

浮点数由阶码和尾数两部分组成，阶码为带符号整数，尾数为小于 1 带符号的小数（如尾数的绝对值还满足大于或等于 1/2，则称该浮点数为规格化浮点数）。计算过程中主要以足够长的尾数来保证数据的精度，以阶码来调整数模（绝对值）的大小（即改变小数点位置），并自动进行符号处理。因此，浮点数具有精度高、数的表示范围宽等特点，特别适用于计算过程复杂、精度要求高的场合。

1.1.1　阶码的选择　移码

一个 8 位的二进制计算器，我们让它不停地计数，其状态就在 00H～0FFH 间周而复始地变化，可计 256 个数。若把这 256 个数看作带符号数：最高位为符号位，0 为正，1 为负，并认为在 7FH 与 80H 之间产生间断，则得到补码：00H＝0，40H＝＋64，7FH＝＋127，80H＝－128，0C0H＝－64 和 0FFH＝－1，如图 1－1 所示。

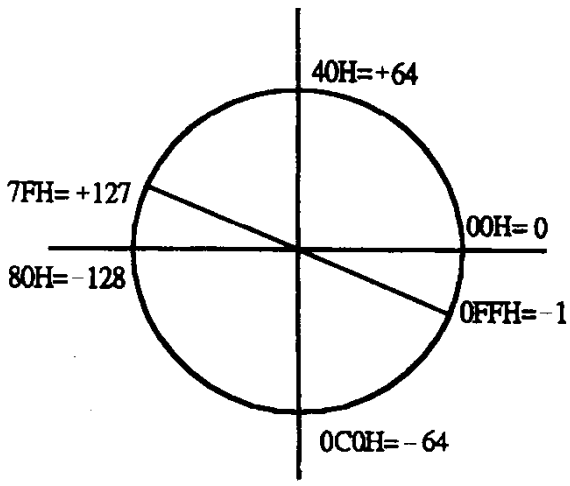


图 1－1　补 码

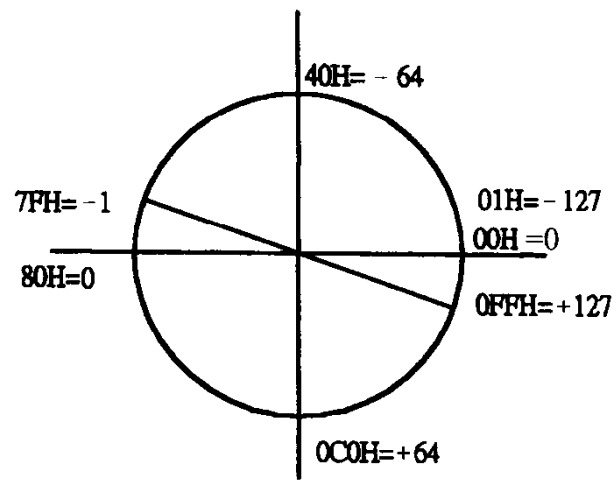


图 1－2　移 码

补码的优点为：加减运算方便——符号位可参与运算，可直接使用访问 OV 标志位的指令来判断溢出及比较符合习惯等。以补码作为浮点数的阶码，给运算带来一定方便。

若把图 1－1 中 0FFH 与 00H 间、00H 与 01H 间都视为间断点（00H 为一孤立点，只作为浮点数 0 的阶码），令 01H＝－127，40H＝－64，7FH＝－1，80H＝0，0C0H＝＋64 及 0FFH＝＋127，如图 1－2 所示，即把最高位也视为符号位，但与补码相反：1 为正，0 为负，则得到移码。我们将看到，以移码作为浮点数的阶码，要比以补码作为阶码优越得多。

1.1.2　浮点数格式

人们根据实践要求和实用性约定了不同的浮点数，浮点数的不同格式决定了不同的运算、处理方法、数据处理精度和数模表示范围，改变了浮点运算程序，而人们对浮点数的认识也有一个提高的过程。

单精度浮点数因其精度、运算速度和数模范围都能满足一般场合需要，因而为各种浮点运算和浮点程序库所采用。下面主要介绍几种单精度浮点数格式。

1. 双符号单精度：这是一种早期的单精度格式。阶码为 1 字节，模 2 补码形式，0～7FH 表示 0～＋127，80H～0FFH 表示 －128～－1（见图 1－1）。尾数为 3 字节，其最高两位为符号位，00 表示正数，11 表示负数。因最高 2 位为符号所占，所以精度只有 22 位，为模 4 补码形式。数模范围为 2.9×10^{-39} ～1.7×10^{38}。其格式如图 1－3 所示。

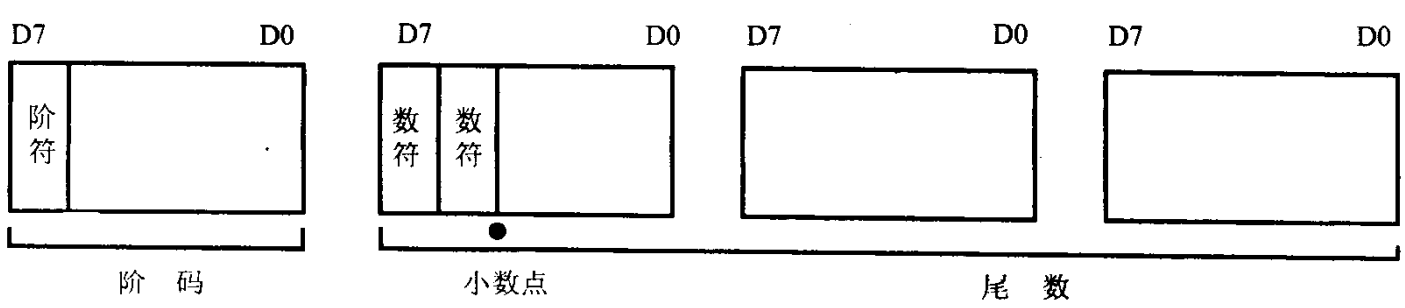


图 1－3　双符号单精度浮点数格式

早期微处理器如 INTEL8080/8085 等都没有测试加、减运算溢出的指令，故采用双符号位尾数。双符号位与数据位一样可参加加法运算（减法运算时将减数求补后转为加法运算），运算结果最高两位为 01 或 10 为溢出，为 00 或 11 则为规格化浮点数或非规格化（小于 0.5）浮点数。因此，对尾数的进一步处理提供了必要条件，对浮点加法判断溢出和规格化等带来一定方便。但对 Z80 微处理器、Z8 单片机、MCS－51/96 系列单片机等有溢出标志的机种来说，当且仅当采用 1 位符号位（模 2 补码）时，才能使用访问 OV 指令来判断溢出，这样可以简化程序、提高运算速度，采用双符号位尾数既浪费精度又浪费机器时间。

双符号格式还有以下缺点：①只对加、减运算方便，对乘、除运算没有任何方便。②阶码为补码造成浮点数无序化现象。例如：浮点数为零的阶码为零，浮点数 1/8 的阶码为 0FEH（即是－2，$1/8=0.5\times2^{-2}$），而 0＞0FEH，使得数值大的浮点数其阶码反倒小。③即使是＋1 这样简单数的浮点数，求补也是麻烦的，须对求补结果规格化才行，而且与＋1 比较起来有面目全非的感觉。这也是双符号格式无序化现象之一（见表 1－1）。

表 1-1　三种单精度浮点常数对照表

格式 / 数	双符号位	IEEE 标准	SICE 仿真器
0	00000000	00000000	00000000
1	01200000	81000000	01800000
−1	00C00000	81800000	81800000
0.5	00200000	80000000	00800000
−0.5	FFC00000	80800000	80800000
0.1	FD333333	7D4CCCCD	7DCCCCCD
−0.1	FDCCCCCD	7DCCCCCD	FDCCCCCD
$\frac{\pi}{180}$	FB23BE8D	7B0EFA35	7B8EFA35
ln 2	002C5C84	80317218	00B17218
ln 10	0224D763	82135D8E	02935D8E
$\sqrt{2}$	012D413D	813504F3	01B504F3
e≈2.7182818	022B7E15	822DF854	02ADF854
90	072D0000	87340000	07B40000
10^{-10}	DFC90640	5F5BE6FF	5FDBE6FF
10^{10}	222540BE	A21502F9	229502F9
88.02969	072C03CD	87300F34	07B00F34
$\frac{\pi}{2}$	013243F7	81490FDB	01C90FDB

但是这种双符号位格式，对用组合、时序逻辑实现加、减、乘、除运算较为容易理解，故现在仍为某些计算机原理教科书所采用。用双符号来判断溢出的原理导致后来带加、减溢出判断功能 CPU 的出现，二者产生溢出的逻辑完全相同，只不过后者涉及的是累加器的最高 2 位上的进(借)位(见 1.1.6 小节)。

2. IEEE 标准单精度：这是针对双符号单精度缺点的改进型。阶码采用 1 字节移码，以 80H～0FFH 表示 0～＋127，以 01H～7FH 表示－127～－1(见图 1-2)。尾数 3 字节，采用原码形式，24 位精度，7 位十进制数据有效。数符附在尾数最高位上，0 为正，1 为负，数模范围为 5.8×10^{-39}～1.7×10^{38}。其格式如图 1-4 所示。

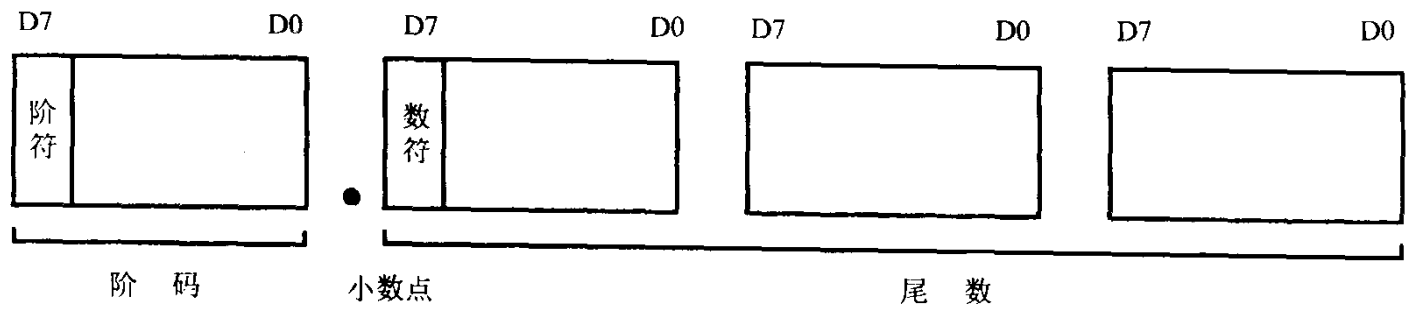


图 1-4　IEEE 标准单精度格式

这种浮点数加、减、乘、除运算都是对绝对值进行的，运算前保存数符(乘、除运算还要预先对乘积、商数符号进行处理)，运算后装配结果数符；具有精度高，乘、除运算方便，求补也方便

(只须求反数符,因此也称为求负)的特点。移码加、减运算虽不能像补码那样可利用 OV 标志位直接判断加、减溢出,但可通过简单运算间接判断溢出。

3. 其他格式单精度:除以上两种单精度格式外,还有多种格式。其中一种是名为 SICE 的仿真器软件包采用的格式,阶码取为 1 字节模 4 补码,阶符放在阶码的次高位上,数符放在阶码的最高位上,尾数为 3 字节原码。现将三种单精度浮点常数列于表 1-1 中,以便比较。

1.1.3 移码的特点

IEEE 标准浮点数以移码作为阶码,以原码为尾数,具以下优点:

1. 阶码为 0 是浮点数等于 0 的充要条件,即阶码为 0,浮点数亦为 0。反过来浮点数等于 0,其阶码也等于 0,而非 0 数不能具有 0 阶码。运算过程中出现阶码变为 0(尾数绝对值不为 0),当作溢出处理。而若出现尾数等于 0(减法运算当被减数与减数值相等时,或两数相乘有一个乘数数值为 0 时等情况),则将阶码清为 0,表示得到了浮点数 0。体现了阶码最小,数模(绝对值)最小的事实。若以补码作阶码,则无此优点。

2. 从图 1-1 与图 1-2 的对比中看出,移码的零点是补码的间断点,反之补码的零点也是移码的间断点(相差 80H)。移码与其对应的补码有如下简单关系:移码=补码±80H,这也是 IEEE 标准浮点数阶码与阶之间的换算公式。移码与补码一样符合求补规律。

3. 由移码特点可知容易做移码加、减运算和判断溢出:移码之和(或差)加上(或减去)80H 即为和(或差)的移码。下面给出移码求和及判断溢出的方法。移码求和有进位,将和加上 80H,如还有进位,即为溢出。此时,对 MCS-51/196 系列单片机来说,CY 和 OV 两标志同时置位,也可能出现 ACC=0(MCS-51)或 Z=1(MCS-196)。移码求和无进位,将和减去 80H,如溢出标志 OV 置位(对 MCS-51/196 同时有 CY=1),或 ACC=0(MCS-51/196),或 Z=1(MCS-196),也产生溢出。无溢出则得到和的移码。对于 MCS-51 系列单片机,可用 CY=1 或 ACC=0 两条件来判断溢出。对于 MCS-196 系列单片机,可用 V=1 或 Z=1 两条件来判断溢出。对于浮点除法中移码减法,可先将除数的阶码求补,转为加法处理。对 MCS-196系列单片机也可直接相减(因减原和加补对进位位影响相同)。浮点乘、除法子程序中调用的子程序 DP,其功能就是求积或商的阶码并判断溢出。

4. 可使检测浮点运算溢出的操作简化。计算结果可能产生非规格化浮点数,这时必须对结果进行左规或右规处理。左规 1 位,阶码减 1,阶码从 01H 变为 00H 为溢出(称为下溢,可将计算结果取为 0);右规 1 位,阶码增 1,阶码从 0FFH 变为 00H 也为溢出(称为上溢)。溢出统归为阶码变为 0。

5. 移码可看作是无符号数按 00H 到 0FFH 从小到大排列,阶码排序与阶值排序相一致,故可按无符号数(绝对值)比较阶的大小,使浮点数的模与阶码具有良好的协调性、有序性,因此优于采用补码作阶码。

本书采用的浮点数即为 IEEE 标准格式。

附录 1:十进制定点数转换为浮点数(IEEE 标准)的方法。(需一个带十进制和十六进制转换功能、10 位十进制精度的计算器。)

1. 整数(可带小数)转换为浮点数:若 $2^{n-1} \leqslant x < 2^n$ 则有 $0.5 \leqslant x/2^n < 1$,将 $x/2^n \cdot 2^{32}$ 转成十六进制数即为 x 的尾数,而阶为 n,阶码为 80H+n。

例:$2^6 < 100 < 2^7$,则将 $100/128 \cdot 2^{32}$ 翻为十六进制数,得到 C8000000H 为尾数,而阶为

7。故有 100＝0. C8000000H · 2^7，尾数为 48000000H(正数)，阶码为 87H(移码)。

2. 小数转换为浮点数：若 $2^{-(n+1)} \leqslant x < 2^{-n}$，则有 $0.5 \leqslant x \cdot 2^n < 1$，将 $x \cdot 2^n \cdot 2^{32}$ 转换成十六进制数为浮点数的尾数，而阶为 $-n$，阶码为 80H $-n$。

例：$1/128 < 0.01 < 1/64$，将 $0.01 \times 64 \times 2^{32}$ 翻为十六进制数，得到 A3D70A3DH 为尾数，而阶为 -6，故有 $0.01 = 0.\text{A3D70A3DH} \times 2^{-6}$，规格化后，尾数为 23D70A3DH，阶码为 7AH(移码)。

仿此可将任意十进制数转换为浮点数。(对负数取其绝对值进行转换，最后保留其尾数最高位 1。对双符号单精度，其尾数绝对值为 IEEE 标准的 1/4，而阶码取为补码)。

1.1.4　浮点数的规格化

非零二进制浮点数表示为 $t \cdot 2^p$，其中 t 为尾数，p 为阶码。若 t 满足 $0.5 \leqslant t < 1$，则称该浮点数为规格化浮点数。凡参加运算的浮点数都必须是规格化的，不然可能造成错误或精度降低。非规格化浮点数来源主要为经过浮点运算后产生的非规格化浮点数。现将双符号格式和 IEEE 标准格式浮点数规格化情况分述如下：

1. 不论是哪一种格式浮点数，在规格化之前，都要对尾数进行测试。如尾数等于 0，则置阶码为 0，表示该浮点数为 0。

2. 双符号单精度浮点数其尾数最高 3 位为 001 或 110 时为规格化浮点数尾数。不然，以 000 或 111 打头者要对浮点数进行左规(左移尾数 1 位，阶码减 1，直到尾数以 001 或 110 打头为止)。以 01 或 10 打头者为加法或求补运算溢出，对其进行右规(尾数算术右移 1 位，阶码增 1)。因此规格化双符号格式浮点数的尾数，写成十六进制数时，只能以 2 或 3 打头(正数)，或以 C 或 D 打头(负数)。以其他数打头的皆为非规格化浮点数，必须对其进行规格化处理。

3. IEEE 标准格式浮点数以原码为尾数，故对绝对值进行规格化。如在加法运算时，尾数相加产生进位，要使尾数连同进位右移 1 位(同时阶码增 1)。如减法或乘法运算使尾数最高位变为 0，则要进行左移尾数，使最高位为 1(同时按移位次数减少阶码值)。如为负数，保留尾数最高位 1，不然将其清除。这种格式浮点数写成十六进制形式，可以 0～F 中任意数打头。

1.1.5　对　阶

对阶是为加、减运算做准备工作，其实质是将相加、减的两个浮点数的小数点位置对齐，亦即小阶向大阶“看齐”。对双符号格式小阶浮点数的尾数算术右移 1 位，对 IEEE 格式须逻辑右移 1 位，阶码增 1，这样反复进行，直到小阶增到与大阶相等为止，这一过程称为对阶。对于 MCS－196 系列单片机，只用阶差控制小阶尾数逻辑右移，执行 1 条指令即可，对阶之后，两浮点数便可相加了。

减法运算时，将减数求补(负)转为加法运算。

1.1.6　溢　出

浮点运算中的溢出是一个内涵丰富的概念，溢出的本意是运算结果超出了数值表示范围。现将其归类如下：

1. 模 2 补码加、减运算溢出

此为最常遇到的一种溢出。当今流行的微处理器、单片机的溢出标志主要是针对模 2 补

码(模 2 补码形式,实际上应是模 256,但把它看作为小数,则为模 2)设计的,如单片机带符号整数加、减法。若两正数相加其和大于 127(含正－负＞127),两负数相加其和小于－128(含负－正＜－128),都产生溢出。又如两正尾数相加其和大于或等于 1(含正－负≥1),两负尾数相加其和小于或等于－1(含负－正≤－1),也产生溢出。溢出表现为正＋正＝负(正－负＝负),或负＋负＝正(负－正＝正)。溢出由累加器最高 2 位上产生进、借位情况进行判断:$OV=C'_7 \oplus C'_6$,即最高位和次高位上有且仅有 1 位向其高位产生进位或借位时,才产生溢出。例如:115＋100＝215＞127,化为二进制计算:73H＋64H＝0D7H＞7FH,考查 C'_7 和 C'_6,发现 $C'_7=0$ 而 $C'_6=1$,故 $OV=0 \oplus 1=1$,确实产生溢出(本例属正＋正＝负)。若 73H 和 64H 为两乘数的阶,则为乘积溢出。若两数相除时,阶码(补码)相减出现正－负＝负,则为商数溢出。

如采用补码作为阶码,则在加法运算时,可能由于两阶大小相差悬殊,对阶时会产生溢出,这时只要取大阶浮点数为和即可,不必进行计算。

2. 除法运算溢出

两数相除,如除得商数超过规定范围(或除数为 0),则产生溢出。例如:MCS-196 浮点程序库中将双字尾数相除转化为双字除以字,字加、减、乘运算时,若不对尾数预先判断就做除法,那么当｜被除数高位字｜≥｜除数高位字｜时,即产生溢出。

3. 用软件判断的溢出

对非模 2 补码,可针对其特点,经简单运算,间接判断溢出。例如:移码加、减运算虽不能直接判断溢出,但经简单处理,就可判断溢出(见 1.1.3 小节)。又如双符号浮点数,加法运算后用软件测得尾数最高 2 位若为 01 或 10,即可断定为溢出。

4. 暂时溢出(假溢出)和真正溢出

暂时溢出是指经规格化处理之后即变为正常的溢出,真正溢出是指不可挽回的溢出。

在加法运算中,若尾数相加产生溢出,多数情况下为暂时溢出,只要右规 1 位,阶码增 1,即变为规格化浮点数。但若阶码增 1 后超出其表示范围(对 IEEE 标准浮点数来说表现为 0FFH＋1＝0),则成为真正溢出。又如乘法运算时,两乘数分别为 0.5×2^{64} 和 0.6×2^{64},在处理乘积阶码时,虽因 64＋64＝128 超出正阶表示范围,但并末产生溢出。因尾数相乘结果为 0.3,规格化后为 0.6,阶减 1 变为 127,实际乘积为 0.6×2^{127},故阶码相加表现为假溢出。除法运算时,若除数为 0 或阶相减之差大于 127,计算 e^x 时若 x＞88.029 691 9 等等场合都是真正溢出的例子。

1.1.7 浮点运算中尾数的处理

为尽量不影响精度,加、减、乘、除、开平方运算后都有对尾数截去部分做四舍五入处理;对阶右移时,将最后移出位四舍五入。乘、除、开平方时,将结果尾数都多取 1 位,再将其四舍五入。除法运算第 1 次试商时,若被除数大于或等于除数(指尾数绝对值),不是简单地将被除数右规 1 位,而是先求差,商阶增 1,直接进入求商循环中。

四舍五入处理,只争得一位半位精度,却增加了程序复杂性,降低了可读性。但从统计规律看,多数情况下连续计算中的每次四舍五入,起着相互补偿作用,可使最终误差保持较低水平;若略去四舍五入,也就失去相互补偿效果,可能造成结果有显著的误差。

1.1.8　浮点数的局限性

浮点数是一种规范性好，运算功能强的数据，但它仍有不足之处，表现如下：

1. 数值相差较大的两个浮点数相加(减)，会产生明显的“大克小”现象，特别表现在长时间做累加(雨量、泾流量)运算的仪器中。如 256 个大致相等的浮点数相加之后再有新的浮点数加入，累计值在对阶时，将使该浮点数的尾数截去约 1 个字节，使 3 字节精度变为 2 字节精度。这种现象若属偶然发生，尚不严重，但在大量累加运算的场合就会造成明显的累计误差。为避免“大克小”现象，在只需要做累计、平均的仪器中采用定点运算，在需要提供浮点形式的累加和、平均值时才将定点数规格化为浮点数。

2. 数值接近的两个浮点数相减时，会使尾数有效位数大大减少，精度降低，故也应尽量避免此种情况发生。可采用加长尾数的方法(MCS－196 浮点程序库采用 4 字节尾数)或其他措施，减少精度降低。

1.2　浮点程序库

浮点程序库是一个结构严谨、嵌套合理、功能强、效率高的浮点运算子程序集。因采用 IEEE 标准浮点数有利于浮点程序库的优化，故多数浮点程序库都采用 IEEE 标准浮点数。少数浮点程序库仍沿用双符号浮点数或采用其他形式的浮点数。本书中提供两个浮点程序库，即 MCS－51 系列单片机浮点程序库和 MCS－196 系列单片机浮点程序库，分别简称为 MCS－51 浮点程序库和 MCS－196 浮点程序库。

1.2.1　浮点程序库的结构

浮点程序库在结构上可分为 3 个层次：以定点运算、定点数制转换和辅助性子程序为基础；以浮点加、减、乘、除和开平方运算为中坚；以对数函数、指数函数、正弦函数、反正弦函数和它们的衍生函数，以及十翻二、二翻十两种浮点数制转换子程序为高层。如表 1－2 所列，程序库中上层对下层为调用关系(可越级调用)，下层对上层则为支持关系。其中十翻二、二翻十子程序是连接十进制和二进制的纽带和桥梁。

由幂级数展开式，我们知道 $\sin x=(e^{ix}-e^{-ix})/2i$。正弦函数和指数函数并无本质差别，只是定义域不同而已，或者说正弦函数只不过是指数函数的一个衍生。因此，浮点程序库中的高层只不过是由指数函数、十翻二子程序以及它们的反函数子程序等组成的。

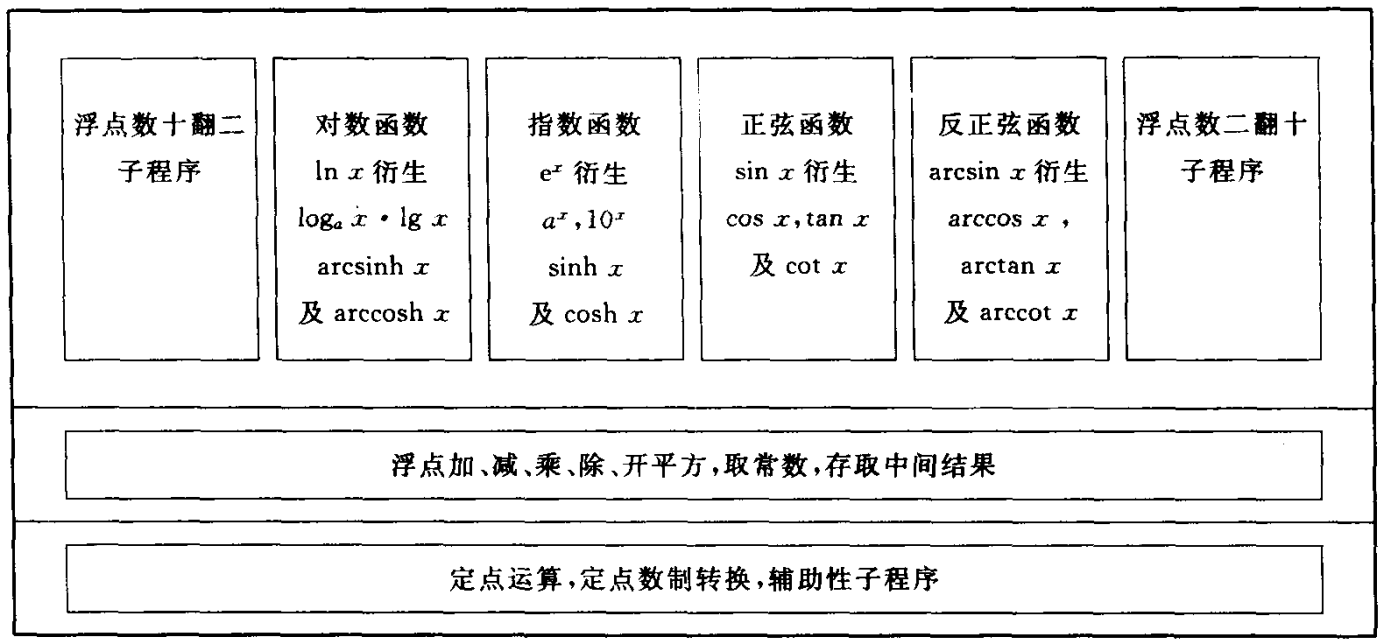
1.2.2　浮点程序库的优化

一个优良的浮点程序库应具备以下优点和特征：

1. 没有隐患和漏洞，功能强，内容丰富，精度高，数的表示范围宽，运行速度快。

2. 密切结合机器资源特点，采用最佳数学模型。如 MCS－51 浮点程序库中用模拟手算开平方。又如 MCS－196 浮点程序库中用牛顿二项式展开双字尾数相除，将其转化为双字除以字、字加、减、乘，大大提高浮点除法运算速度；将浮点数压缩到[0.5,2)半开区，再用牛顿迭代实现快速开平方，极大地提高了浮点数开平方速度。数学模型的优化才是根本性的、彻底的优化，其他优化措施只能起辅助作用。

表 1－2 浮点程序库的结构层次



3. 采用优化组合(移码为阶码，原码为尾数)的浮点数—— IEEE 标准浮点数，有利于规范化编程和优化程序设计。

4. 采用模块化设计，突出模块功能和模块间的联系，透明度高，可读性、可移植性、可维护性强。

5. 入、出口条件规范化，方便用户。

6. 空间应能浮动，用户可灵活安排其占用空间。

7. 尽量少占资源，将较多资源留给用户。

本书介绍的 MCS－51 浮点程序库和 MCS－196 浮点程序库，即是按上述原则设计的浮点程序库。

第 2 章　MCS－51 浮点程序库

本章主要讲述结合 MCS－51 系列单片机软件资源的特点，设计 MCS－51 浮点程序库的方法。详细介绍了基本运算子程序、函数计算子程序和数制转换子程序的设计方法。

2.1　MCS－51 系列单片机软件资源特点及 MCS－51 浮点程序库的特点

MCS－51 浮点程序库的特点与 MCS－51 系列单片机软件资源的特点密切相关，故本节首先讲述 MCS－51 系列单片机软件资源的特点，再讲述优化设计的 MCS－51 浮点程序库。

2.1.1　MCS－51 系列单片机软件资源的特点

MCS－51 系列单片机软件资源特点与 MCS－51 浮点程序库的设计密切相关，具有以下特点：

1. MCS－51 系列单片机有四组通用寄存器，具有双重角色：即作为寄存器，又可作为内存 RAM。作为寄存器，可与累加器快速交换、传送数据，也可与内存 RAM 以及特殊功能寄存器传送数据；若把部分寄存器看作内存，还可实现寄存器间数据的直接传送（寄存器与内存、内存与内存，不占用指针，提高传送速度）。寄存器组间还可实现快速切换，故在寄存器组里可实现高速浮点运算。

2. 在片内 RAM 里已能直接与立即数或 ACC 进行逻辑运算，数据已能在 RAM 间直接传送，RAM 单元能直接装入立即数，并能执行增/减 1 操作，可与寄存器一样作为计数器控制循环，已具有 16 位机的部分功能。

3. 有 OV 标志位供判断加、减运算溢出。

4. 有 2 KB 空间内短调用、短转移指令。利用这一特点加上优化措施，可实现浮点程序库空间的浮动性。

5. 有字节乘、除法，位寻址、位处理等指令及特有的 CJNE 以及其他多种条件转移指令，为优化程序提供了方便条件。MCS－51 浮点程序库的设计正是利用了上述资源的特点。

2.1.2　MCS－51 浮点程序库的特点

MCS－51 浮点程序库是基于 MCS－51 系列单片机软件资源特点，采取优化措施设计的，其特点如下：

1. 浮点运算使用第 1、3 组寄存器，第 0、2 两组寄存器用于保护现场。在使用 MCS－51 浮点程序库之前，将寄存器从 0 组切换到 1 组，使用完毕后，再从 1 组切换回 0 组。以寄存器组间的快速切换，替代对存储器的存取操作，提高了浮点运算的速度。

2. 采用了 IEEE 单精度浮点数，有利于优化编程，具有 7 位十进制数据精度。

3. 程序库内容丰富，优化程度高，嵌套合理，结构谨严。在 2 KB 空间内安排了对数、指数、三角函数、反三角函数、双曲函数、反双曲函数等 20 余种函数子程序以及浮点加、减、乘、

除、开平方等基本运算子程序共 100 余个。库内皆为短调用、短转移指令，因此程序库空间具有浮动性。

4. 将开平方也作为加、减、乘、除同一层次的运算(模拟手算，快于牛顿迭代)。因开平方子程序与反正弦函数(及其衍生函数)、反双曲函数等函数有关，故也提高了相关函数的计算速度。

5. 采取严密措施，阻塞漏洞，封锁一切可能的运算溢出。用户还可以直接改写各溢出转移地址(如程序中 LJMP 0FFFFH 可改为转向 64 KB 空间任意地址去处理溢出，甚至可直接改写机器码)；若不改写，溢出时，程序自动转向初始化。

6. 浮点运算各子程序入、出口规范统一。具体安排如下：对加、减、乘、除、开平方等基本运算子程序和各种函数计算子程序，将操作数和运算结果都放在第 1 组寄存器(08H～0FH)中。单操作数如 FPSQ、sin x 中的 x，一律放在第 1 组寄存器 R4～R7(片内 RAM 0CH～0FH)中，其中 R4 内为阶码，R5～R7 内为尾数，其中 R5 内为高位字节。双操作数中将含“被”意义的第 1 操作数如被加数、被减数、被乘数、被除数、被求对数数(即 $\log_a x$ 中的 x)、被乘方数(即 a^x 中的 a)等放在 R0～R3(片内 RAM 08H～0BH)之中，R0 内为阶码，R1～R3 内为尾数，存放顺序与 R5～R7 对应。第 2 操作数则放在 R4～R7 之中。运算结果占据 R4～R7。如不希望破坏原始数据或中间运算结果，应在运算之前将它们转入内存保护。

7. 浮点数比较大小子程序入口条件与浮点减法相同，但不做减法运算，以 ACC 的内容和 CY 状态决定两数大小。

8. 数制转换子程序入、出口条件如下：

(1) 定点十进制数翻为二进制浮点数子程序 $DTOB_1$ 的入口条件为：十进制整数部分在 09H～0BH 内，09H 为高位字节。小数部分在 0CH～0FH 内，0CH 为高位字。整数、小数部分都是压缩 BCD 码。位 7BH＝0 表示正数，7BH＝1 表示负数。出口条件为：二进制浮点数在 0CH～0FH 内。

(2) 浮点数十翻二子程序 DTOB 的入口条件为：十进制浮点数在 09H～0FH 单元内。09H 内为阶符，0AH 内为数符，00H 为正，0FFH 为负，0BH 内为阶，0CH～0FH 内为纯小数尾数，其中 0CH 内为高位字节，阶与尾数都是压缩 BCD 码。出口条件为：二进制浮点数在 0CH～0FH 内。

(3) 浮点数二翻十子程序 BTOD 的入口条件为：二进制浮点数在 0CH～0FH 内(单操作数位置)。出口条件为：十进制浮点数在 08H～0EH 内，其中 08H 内为阶，09H～0CH 内为纯小数尾数，09H 内为高位字节，阶与尾数都是压缩 BCD 码，0DH 内为阶符，0EH 内为数符，00H 为正，非 0 为负。

9. MCS-51 浮点程序占用资源情况如下：

如只使用基本运算子程序和定点数制转换子程序，占用累加器 ACC、寄存器 B、片内 08H～0FH、1DH～1FH 及 2FH 共 12 个 RAM 单元。如使用函数计算子程序，还要占用 18H～1CH 和 30H～37H 等共 13 个 RAM 单元以及数据指针 DPTR。

堆栈深度：最多 6 级。

2.2　基本运算子程序的设计方法

基本运算子程序包括：加、减、乘、除运算子程序，浮点数比较大小子程序浮点和浮点数开平方子程序，它们是浮点程序库的核心部分。它们既可以独立使用（在多数场合有它们已足够），又是对函数计算和浮点数制转换必不可少的支持。其设计方法完全是模拟手算，即（循环）移位相加或相减。

2.2.1　支持基本运算的辅助子程序

它们包括双精度加、减法、尾数逻辑/算术移位、交换数据、积商符号处理、尾数增 1 处理、以及浮点数规格化等。它们也对函数计算提供支持。

程序清单：

清单 1

```
        ORG     2000H
ADD0:   CLR     C               ;双精度加法子程序
ADC0:   XCH     A,R7            ;双精度带进位位加法子程序
        ADDC    A,R3
        XCH     A,R7
        XCH     A,R6
        ADDC    A,R2
        XCH     A,R6
        RET
ADD1:   CLR     C               ;双精度加法子程序
ADC1:   XCH     A,R7            ;双精度带进位位加法子程序
        ADDC    A,R5
        XCH     A,R7
        XCH     A,R6
        ADDC    A,R4
        XCH     A,R6
        RET
ADD2:   CLR     C               ;双精度加法子程序
ADD2:   XCH     A,R7            ;双精度带进位位加法子程序
        RLC     A
        XCH     A,R7
        XCH     A,R6
        RLC     A
        XCH     A,R6
        RET
SUBO:   CLR     C               ;双精度减法子程序
SBCO:   XCH     A,R7            ;双精度带进位位减法子程序
        SUBB    A,R3
```