第3章 数据类型

本章介绍 Intel 64 和 IA-32 体系结构定义的数据类型，本章最后一节描述在 x87FPU、SSE、SSE2、SSE3 和 SSSE3 扩展中所用的实数和浮点数概念。

3.1 基本数据类型

IA-32 结构的基本数据类型是字节、字、双字、四字和双四字，如图 3-1 所示。

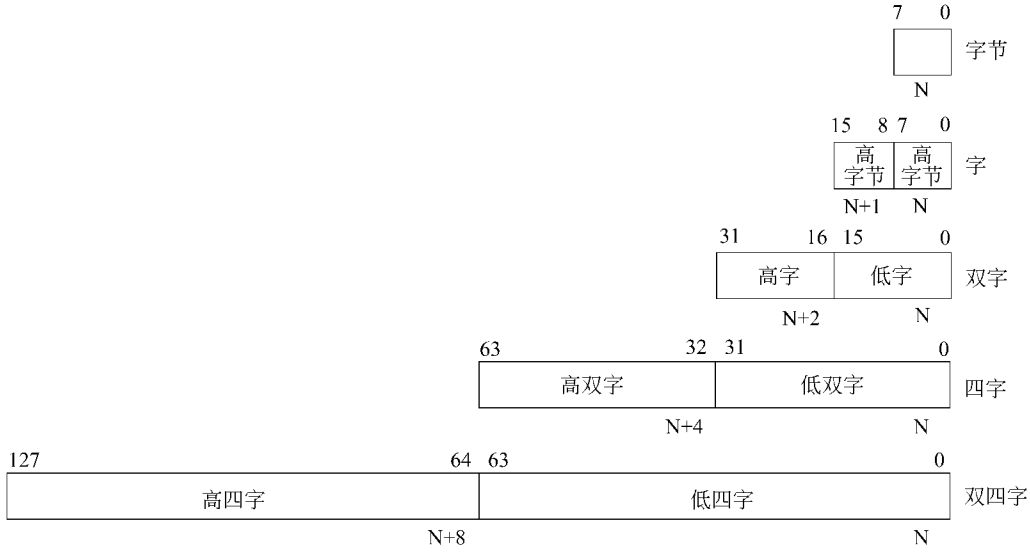

图 3-1 基本数据类型

一个字节是 8 位，一个字是两个字节(16 位)，双字是 4 字节(32 位)，四字是 8 字节(64 位)，双四字是 16 字节(128 位)。IA-32 体系结构指令的子集在这些基本的数据类型上操作，无任何附加的操作数类型。

四字是在 Intel 486 中引入 IA-32 结构的，双四字是在具有 SSE 扩展的 Pentium Ⅲ处理器中引入的。

图 3-2 显示了基本数据类型作为内存中的操作数引用时的字节顺序。

低字节(位 0 到 7)占用内存中的最低地址，此地址也是此操作数的地址。

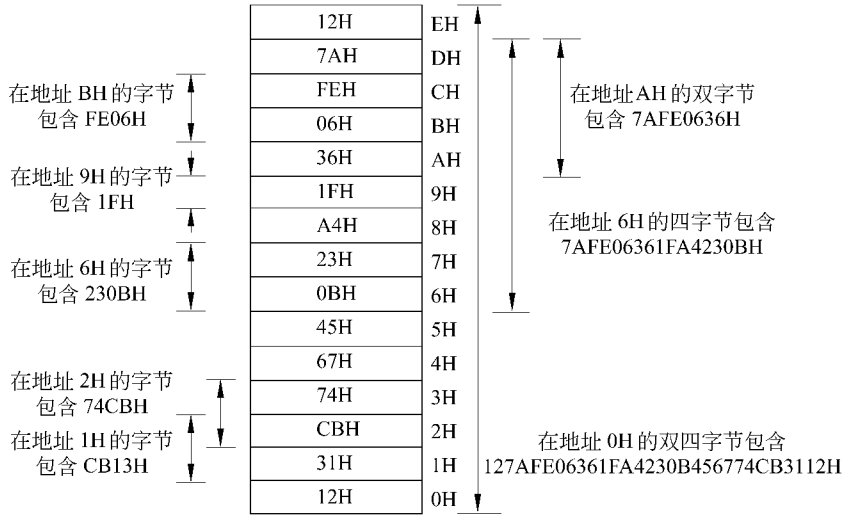

图 3-2　基本数据类型在内存中的字节顺序

字、双字和四字在内存中并不需要对齐至自然边界(字、双字和四字的自然边界是偶数编号的地址,对于双字和四字来说,地址要分别能被 4 和 8 除尽)。

然而,为改进程序的性能,数据结构(特别是堆栈)只要可能应对齐在自然边界上。这样做的理由是:对于不对齐的存储访问,处理器要求做两次存储访问操作;而对于对齐的访问只要做一次存储访问操作。跨越 4 字节边界的字或双字操作数或跨越 8 字节边界的操作数被认为是未对齐的,要访问它们需要两个分别的总线周期;起始于奇数地址但不跨越字边界的字,认为是对齐的,仍能在一个总线周期内访问。

某些操作双四字的指令要求存储操作数对齐在自然边界上。对于双四字的自然边界是能被 16 整除的偶数地址。对于这些指令,若规定了未对齐的操作数,则产生通用保护异常(#GP)。有些操作双四字操作数的指令,允许操作数不对齐,但要求额外的总线访问周期。

3.2　数字数据类型

虽然字节、字和双字是 IA-32 结构的基本数据类型,但某些指令对这些数据类型的附加解释允许在数字数据类型(带符号的或无符号整数和浮点数)上操作。这些数字数据类型如图 3-3 所示。

3.2.1　整数

Intel 64 和 IA-32 结构定义两种类型整数:无符号和符号整数。无符号整数是原始二进制值,范围从 0 到所选择的操作数尺寸能编码的最大正数。符号整数是 2 的补码二进制值能用于表示正的和负的整数值。

某些整数指令(例如 ADD、SUB、PADDB 和 PSUBB 指令)可在无符号或符号整数上操作。而一些整数指令(例如 IMUL、MUL、IDIV、DIV、FIADD 和 FISUB)只能在一种整数类型上操作。

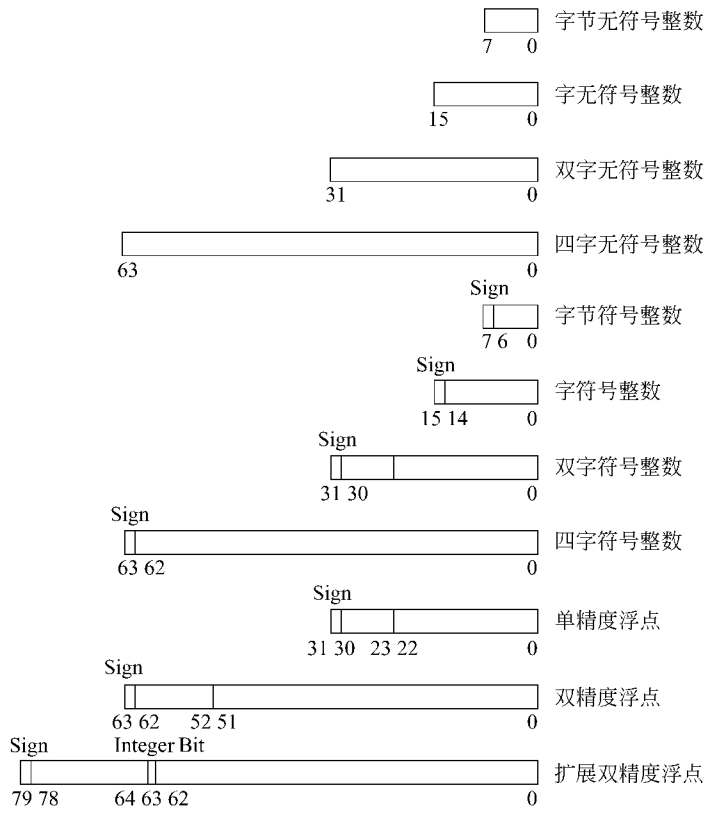

图 3-3 数字数据类型

下面描述两种整数类型的编码与范围。

1. 无符号整数

无符号整数是包含字节、字、双字和四字中的无符号的二进制数。它们的值的范围，对于字节是从 0 到 255；对于字，从 0 到 65 535；对于双字，从 0 到 $2^{32}-1$；对于四字，从 0 到 $2^{64}-1$。无符号整数有时作为原始数引用。

2. 符号整数

符号整数是保存在字节、字、双字或四字中的带符号的二进制数。对于符号整数的所有操作都假定用 2 的补码表示。符号位定位在字节整数的位 7、字整数的位 15、双字整数中的位 31 和四字整数中的位 63(见表 3-1)。

负数的符号位为 1，正数的符号位为 0。整数值的范围，对于字节，从－128 到＋127；对于字从－32 768 到＋32 767；对于双字，从-2^{31}到$+2^{31}-1$；对于四字，从-2^{63}到$+2^{63}-1$。

当在内存中存储整数值时，字整数存放在两个连续字节中；双字整数存放在 4 个连续字节中；四字整数存放在 8 个连续的字节中。

整数无穷大是一特殊值，有时由在整个值上操作的 x87 FPU 返回。

3.2.2 浮点数据类型

IA-32 结构定义和操作 3 种浮点数据类型：单精度浮点、双精度浮点和扩展的双精度浮点(见图 3-3)。

表 3-1 符号整数编码

类别		2的补码	
		符号	
正数	最大	0	11…11
	最小	0	00…01
零		0	00…00
负	最小	1	11…11
	最大	1	00…00
整数无穷大		1	00…00
		符号字节整数	7位
		符号字整数	15位
		符号双字整数	31位
		符号四字整数	63位

这些数据类型的数据格式与IEEE标准754二进制浮点算术所规定的格式直接相应，见表3-2。

表 3-2 浮点数格式

数据类型	长度	精度(位)	近似的规格化的范围	
			二进制	十进制
单精度	32	24	$2^{-126}\sim2^{127}$	$1.18\times10^{-38}\sim3.40\times10^{38}$
双精度	64	53	$2^{-1322}\sim2^{1023}$	$2.23\times10^{-308}\sim1.79\times10^{308}$
双扩展精度	80	64	$2^{-16382}\sim2^{16383}$	$3.37\times10^{-4932}\sim1.18\times10^{4932}$

注：第3.8节“实数和浮点数格式”中给出IEEE标准754浮点格式和定义术语整数位、QNaN、SNaN和未规格化值的概要。

表3-3显示了对于零、未规格化的有限数、规格化有限数、无穷大和NaN在3种浮点数据类型的浮点编码。它也给出了QNaN浮点无穷大的格式。

表 3-3 浮点数和NaN编码

类		符号	偏移阶	有效数	
				整数①	分数
正	+∞	0	11…11	1	00…00
	+规格化	0 ⋮ 0	11…10 ⋮ 00…01	1 ⋮ 1	11…11 ⋮ 00…00
	+未规格化	0 ⋮ 0	00…00 ⋮ 00…00	0 ⋮ 0	11…11 ⋮ 00…01
	+零	0	00…00	0	00…00

续表

类		符号	偏移阶	有效数	
				整数①	分数
负	−零	1	00…00	0	00…00
	−未规格化	1 ⋮ 1	00…00 ⋮ 00…00	0 ⋮ 0	00…01 ⋮ 11…11
	−规格化	1 ⋮ 1	00…01 ⋮ 11…10	1 ⋮ 1	00…00 ⋮ 11…11
	−∞	1	11…11	1	00…00
NaN	SNaN	×	11…11	1	0×…××②
	QNaN	×	11…11	1	1×…××
	QNaN 浮点无穷大	1	11…11	1	10…00
	单精度 双精度 扩展的双精度		←8 位→ ←11 位→ ←15 位→		←23 位→ ←52 位→ ←63 位→

注：① 整数位是隐含的，对于单精度和双精度格式是不存储的。

② 对 SNaN 编码分数必须为非零，而最高有效位为 0。

每种浮点数据类型的阶使用偏移阶格式；对于单精度格式，偏移常数是 127，对于双精度格式是 1023，对于扩展的双精度格式是 16 383。

当存储浮点值至内存时，单精度值存储在内存的 4 个连续字节内；双精度值存储在 8 个连续字节内；扩展的双精度值存储在 10 个连续字节内。

单精度和双精度浮点数据类型由 x87 FPU 和 SSE/SSE2/SSE3 指令操作，扩展的双精度浮点数据格式只由 x87 FPU 操作。

3.3 指针数据类型

指针是内存单元的地址(见图 3-4)。

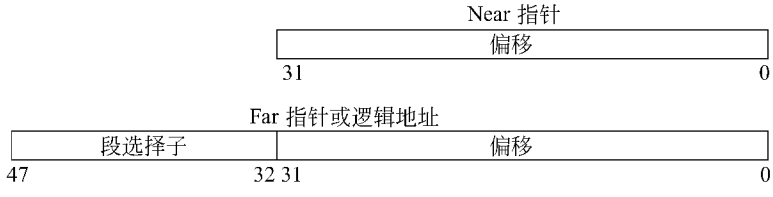

图 3-4 指针数据类型

在非 64 位方式下，体系结构定义两种类型的指针：近(Near)指针(32 位)和远(Far)指针(48 位)。Near 指针是段内的 32 位偏移量(也称为有效地址)。Near 指针在平面存储模式中

用于所有存储器引用;在分段存储模式中用于同一段内的存储器引用。正在访问的段隐含是相同的。

Far 指针是一个 48 位的逻辑地址,包含 16 位段选择子和 32 位的偏移。Far 指针用于在分段存储模式中的跨段存储引用。正在访问的段必须显示规定。具有 32 位偏移量的 Near 和 Far 指针如图 3-4 所示。

在 64 位方式(IA-32e 方式的一种子方式),Near 指针是 64 位。这等于有效地址。在 64 位方式中,Far 指针可能是以下 3 种形式之一:

- 若操作数尺寸是 32 位,16 位段选择子、16 位偏移量;
- 若操作数尺寸是 32 位,16 位段选择子、32 位偏移量;
- 若操作数尺寸是 64 位,16 位段选择子、64 位偏移量。

具体如图 3-5 所示。

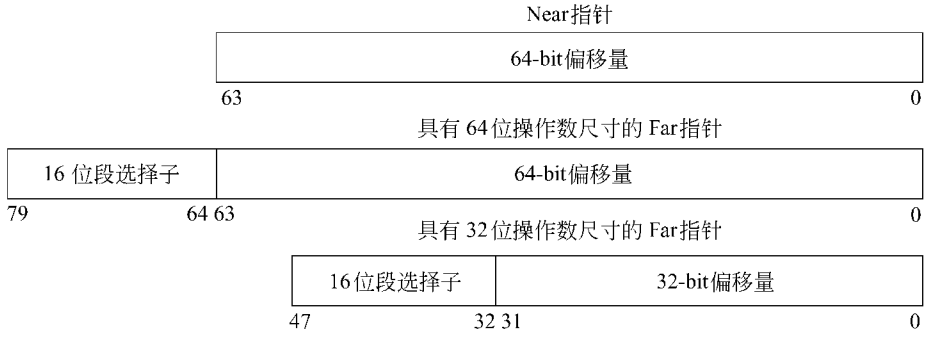

图 3-5　在 64 位方式的指针

3.4　位字段数据类型

一位字段(见图 3-6)是连续的位序到。它能在内存中任何字节的任一位位置开始并能包含多至 32 位。

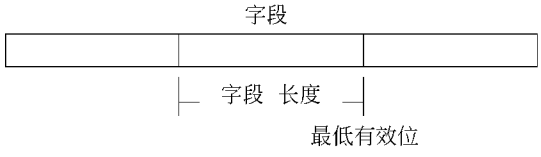

图 3-6　字段数据类型

3.5　串数据类型

串是位、字节、字或双字的连续序到。位串能从任一字节的任一位开始并能包含多至 $2^{32}-1$ 位。字节串能包含字节、字或双字,其范围从 0～$2^{32}-1$ 字节(4G 字节)。

3.6 组合的 SIMD 数据类型

Intel 64 和 IA-32 体系结构定义在一组 64 位和 128 位组合的数据类型上用 SIMD 操作。这些数据类型由基本数据类型(组合的字节、字、双字和四字)和用作组合的整数以及组合的浮点数的基本数据类型的数字的解释组成。

3.6.1 64 位 SIMD 组合的数据类型

64 位组合的数据类型是在 Intel MMX 技术中引入 IA-32 体系结构的。它们在 MMX 寄存器上操作。基本的 64 位组合的数据类型是组合的字节、组合的字和组合的双字(见图 3-7)。当在这些数据类型上执行数字操作时,这些数据类型作为包括字节、字或双字的整数值解释。

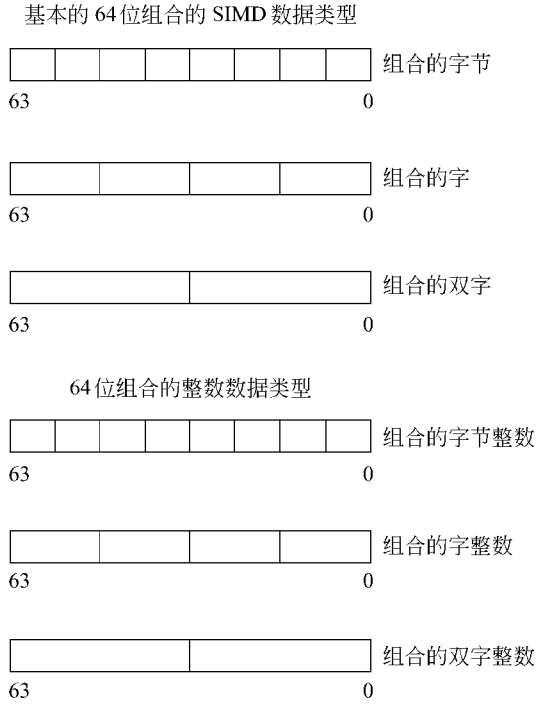

图 3-7 64 位组合的 SIMD 数据类型

3.6.2 128 位组合的 SIMD 数据类型

128 位组合的数据类型是在 SSE 扩展中引入 IA-32 体系结构并用在 SSE2、SSE3 和 SSSE3 扩展中。它们主要在 128 位 XMM 寄存器和内存中操作。基本的 128 位组合的数据类型是组合的字节、组合的字、组合的双字和组合的四字(见图 3-8)。当在这些在 XMM 寄存器中的基本的数据类型上执行 SIMD 操作时,这些数据类型作为包含组合的或标量单精度浮点数或双精度浮点数或作为包含组合的字节、字、双字或四字整数值解释。

图 3-8　128 位组合的 SIMD 数据类型

3.7　BCD 和组合的 BCD 整数

二进制编码的十进制整数(BCD 整数)是有效值为从 0～9 的无符号 4 位整数。IA-32 体系结构定义在定位在一个或多个通用寄存器或一个或多个 x87 FPU 寄存器(见图 3-9)上的整数上操作。

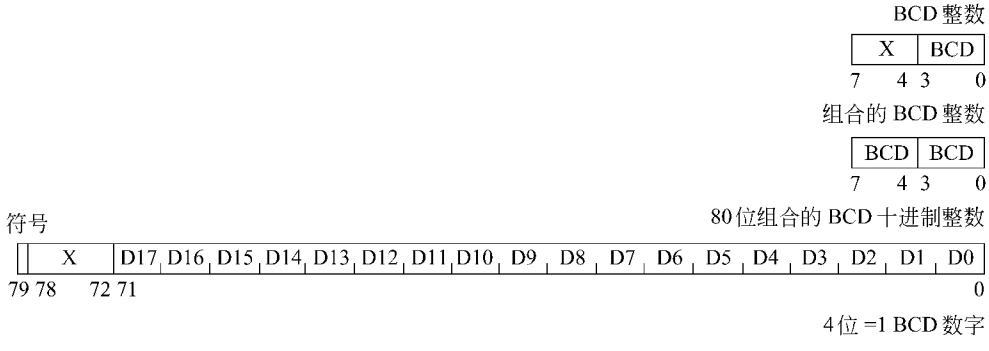

图 3-9　BCD 数据类型

当在通用寄存器中的BCD整数上操作时，BCD值可以是未组合的(每字节一位BCD数字)或组合的(每字节两位BCD数字)。未组合的整数的值是低半字节(位3～0)的二进制值。高半字节(位7～4)在做加法或减法期间可能是任意值，但在乘法和除法期间必须是0。组合的BCD整数允许两个BCD数字包含在一个字节中。其中，在高半字节中的BCD数字为有更高有效位。

当在x87 FPU数据寄存器中的BCD整数上操作时，BCD值组合进一80位格式，作为十进制整数引用。在这样的格式中，前9个字节保持18个数字，每字节两个。最低有效数字包含在字节0的低半字节。字节10的最高有效位包含符号位(0＝正和1＝负)；字节10的位0～6无用。负的十进制整数不用2的补码格式存储；从正至负，只是符号位不同。在这种格式下能编码的十进制整数的范围是$-10^{18}+1$～$+10^{18}-1$。

十进制数格式只在内存中存在。当十进制数装入至一x87 FPU数据寄存器时，它自动转换为扩展的双精度浮点数格式。所有十进制整数能用扩展的双精度格式精确表示。

表3-4给出了十进制数据类型的值的可能的编码。

表3-4 组合的十进制数编码

类 别	符号		数 量					
			数字	数字	数字	数字	…	数字
正的最大值	0	0000000	1001	1001	1001	1001	…	1001
⋮	⋮	⋮	⋮	⋮	⋮	⋮	…	⋮
正的最小值	0	0000000	0000	0000	0000	0000	…	0001
零	0	0000000	0000	0000	0000	0000		0000
负零	1	0000000	0000	0000	0000	0000	…	0000
负的最小值	1	0000000	0000	0000	0000	0000	…	0001
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮
负的最大值	1	0000000	1001	1001	1001	1001	…	1001
组合的BCD整数无穷大	1	1111111	1111	1111	1100	0000	…	0000
	←1个字节→		←9个字节→					

组合的整数无穷大编码(FFFFC000000000000000H)由FBSTP指令在响应屏蔽的浮点无效操作异常时存储的。企图用指令FBLD装入这样的值产生未定义的结果。

3.8 实数和浮点格式

本节描述实数在x87FPU和SSE/SSE2/SSE3浮点指令中如何用浮点格式表示。也介绍例如规格化数、未规格化数、偏移阶、符号零和NaN等术语。

3.8.1 实数系统

如图3-10所示，实数系统由从负无穷大(－∞)至正无穷大(＋∞)的实数连续体组成。

3.8.2 浮点格式

为增加实数计算的速度和效率，计算机和微处理器典型地用二进制浮点数格式表示实数。

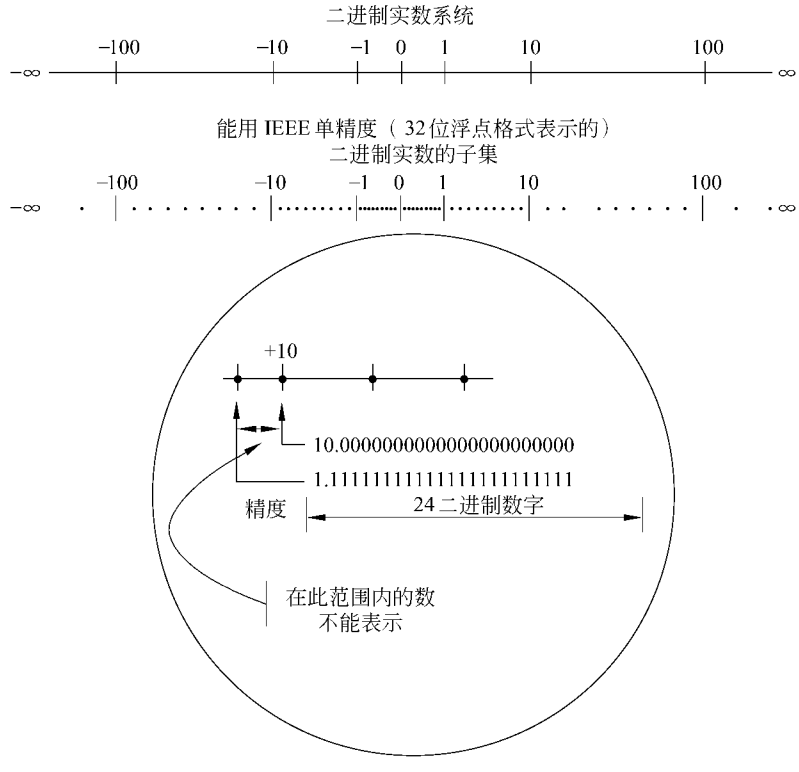

图 3-10　二进制实数系统

在这样的格式中，一实数有 3 部分：符号、有效数和阶（见图 3-11）。

符号是指示数是正（0）或负（1）的一二进制值。有效数有两部分：一位二进制整数（也称为 J 位）和一二进制分数。整数位常常不表示，代之以一隐含值。指数是一二进制整数，它是以 2 为基的由有效数相乘的权。

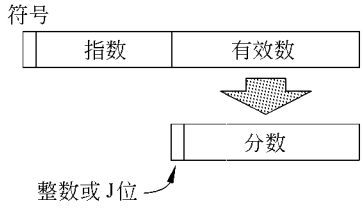

图 3-11　二进制浮点格式

表 3-5 显示了实数 178.125（以普通的十进制格式表示）如何用 IEEE 标准 754 浮点格式存储。此表列出实数符号的级数，它导致单精度、32 位浮点格式（在此格式中）的有效数是格式化的和阶是偏移的。对于单精度浮点格式，偏移常数是＋127。

表 3-5　实数和浮点数符号

普通十进制	178.125		
科学记数法十进制数	$1.78125E_{10}2$		
科学记数法二进制数	$1.0110010001E_{2}111$		
科学记数法二进制数（偏移阶）	$1.0110010001E_{2}10000110$		
IEEE 单精度格式	符号 0	偏移阶 10000110	规格化的有效数 01100100010000000000000 1.（隐含的）