UDC 681.3.053:007(083.73)
L 71

中华人民共和国国家标准

GB/T 14815.1—93
ISO 9282.1—1988

信息处理 图片编码表示 第一部分:在七位或八位环境中图片表示的编码原则

Information processing—Coded representation of pictures—Part 1:Encoding principles for picture representation in a 7-bit or 8-bit environment

1993-12-24 发布　　1994-08-01 实施

国家技术监督局 发布

中华人民共和国国家标准

信息处理　图片编码表示 第一部分：在七位或八位环境中 图片表示的编码原则

GB/T 14815.1—93
ISO 9282.1—1988

Information processing—Coded representation of pictures—Part 1: Encoding principles for picture representation in a 7-bit or 8-bit environment

本标准等同采用国际标准 ISO 9282.1—1988《信息处理　图片编码表示　第一部分：在七位或八位环境中图片表示的编码原则》。

本标准规定了图片编码的标准方法，以便协助编码系统的设计，并防止各种非关连的编码技术激增。

本标准的这一部分规定了能由大多数计算机图形应用所生成的图片表示的编码方案；此编码方案基于七位结构且可在七位或八位环境中使用。

1　主题内容与适用范围

本标准的这一部分规定了：

a.　在七位或八位环境中用于交换由图像组成的图片信息的编码原则；

b.　用于表示描述图片用的图原的数据结构；

c.　可用作图原中的操作数的通用数据类型。

本标准的这一部分不涉及图片的表示语义，这些内容在其他有关的标准中规定。

本标准的这一部分适用于由按照 GB 10022 中规定的图片编码方法而构造的数据所组成的数据流。

2　引用标准

下列标准包含的条文，通过在本标准中引用而构成为本标准的条文。在标准出版时，所示版本均为有效。所有标准都会被修订，使用本标准的各方应探讨使用下列标准最新版本的可能性。

GB 1988　信息处理　信息交换用七位编码字符集

GB 2311　信息处理　七位和八位编码字符集代码扩充技术

GB 5261　文字和符号成形设备用的增补控制功能

GB 10022　信息处理　图片编码方法的标识

3　术语和记数法

3.1　术语

本标准的这一部分采用下列术语定义：

3.1.1　位组；字节

若干个二进制位的有序集合，用于表示一个操作码或操作数，或用作一个操作码或操作数代码表示

国家技术监督局 1993-12-24 批准　　1994-08-01 实施

的一部分。

3.1.2 代码

一组明确的规则，用于确定一个集内的每个操作码或操作数与其由一个或多个位组构成的编码表示之间的一一对应关系。

3.1.3 代码表

表明代码中的操作码和操作数与其位组的总体分配表。

3.1.4 操作码

一字节或多字节的编码表示，用于标识图片标准所需的一种功能。操作码后可接零个或多个操作数。

3.1.5 操作码表

表明分配到为操作码预留的每个位组的控制功能的表。

3.1.6 操作数

用于规定操作码所需的诸参数的一个或多个编码表示。

3.2 记数法

3.2.1 七位字节

七位字节的各位由 b_7、b_6、b_5、b_4、b_3、b_2、b_1 标识，这里 b_7 是最高位，b_1 是最低位。

位组由 x/y 形式的记数法标识，这里 x 是 0～7 范围内的数，y 是 0～15 范围内的数，分别对应于代码表中标明的列和行。

x/y 形式的记数法和 b_7 至 b_1 位所组成的位组之间的对应关系如下：

a. x 是 b_7、b_6 和 b_5 表示的数，赋予 b_7、b_6 和 b_5 的权值分别为 4、2 和 1；

b. y 是 b_4、b_3、b_2 和 b_1 表示的数，赋予 b_4、b_3、b_2 和 b_1 的权值分别为 8、4、2 和 1。

3.2.2 八位字节

八位字节的各位由 b_8、b_7、b_6、b_5、b_4、b_3、b_2 和 b_1 标识，这里 b_8 是最高位，b_1 是最低位。

位组由 xx/yy 形式的记数法标识，这里 xx 和 yy 是00～15范围内的数。xx/yy 形式的记数法和 b_8 至 b_1 位所组成的位组之间的对应关系如下：

a. xx 是 b_8、b_7、b_6 和 b_5 表示的数，赋予 b_8、b_7、b_6 和 b_5 的权值分别为 8、4、2 和 1；

b. yy 是 b_4、b_3、b_2 和 b_1 表示的数，赋予 b_4、b_3、b_2 和 b_1 的权值分别为 8、4、2 和 1。

3.2.3 字节解释

利用下列权值对各个位进行加权，则可以把一个字节内的各个位解释为表示二进制记数法中的数：

七位字节的位	—	b_7	b_6	b_5	b_4	b_3	b_2	b_1
八位字节的位	b_8	b_7	b_6	b_5	b_4	b_3	b_2	b_1
权　　值	128	64	32	16	8	4	2	1

使用这些权值，七位字节中的位组可表示 0～127 范围内的数。八位字节中的位组可表示 0～255 范围内的数。

3.3 代码表布局

3.3.1 七位代码表示

在七位代码表示中，代码表由排列在 8 列和 16 行上的 128 个位置组成，这里列的编号为 0～7，行的编号为 0～15。

代码表的位置由 x/y 形式的记数法标识，其中 x 是列号，y 是行号。

代码表的位置与位组有一一对应关系。用 x/y 形式表示的代码表位置的记数法与相应的位组记数

法相同。

3.3.2 八位代码表示

在八位代码表示中，代码表由排列在16列和16行上的256个位置组成，这里列和行的编号为00～15。

代码表位置用 *xx/yy* 形式的记数法标识，其中 *xx* 是列号，*yy* 是行号。

代码表的位置与位组有一一对应关系。由 *xx/yy* 形式表示的代码表位置的记数法与相应的位组记数法相同。

4 编码原则

本标准的这一部分涉及：

a. 图原的操作码的编码原则；

b. 图原的操作数的编码原则。

所有这样的编码均以七位字节来定义。当在八位环境中使用时，每个字节的第8位便为0（“串”格式除外）。

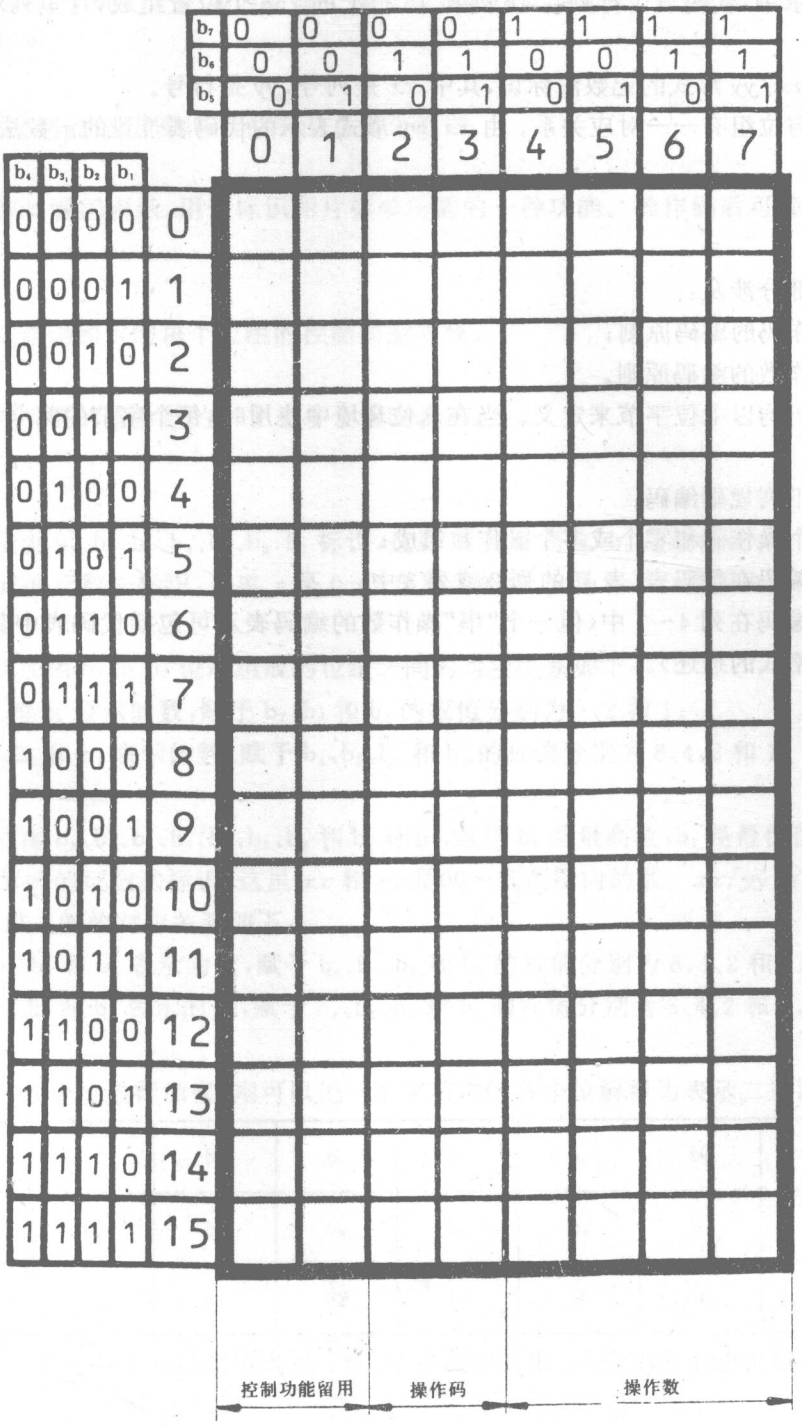
每个图原依据下列规则编码：

a. 图原由一个操作码和零个或多个操作数组成；

b. 操作码的编码在代码表（表1）的列2或列3中；

c. 操作数的编码在列4～7中（但一个“串”操作数的编码表示可包括代码表中其他列内的位组，详见6.2.3中对串格式的描述）。

表 1　图片编码用的代码表



5　操作码的编码原则

5.1　概述

为定义代码表而组织操作码时，可选用下列两种编码技术之一：

a. 若基于这些编码原则的特定标准中所需的操作码的个数少于或等于 32 时，则可用 5.2 中所述的紧致结构；

b. 若需要更多的操作码，则可用 5.3 中所述的可扩充结构。

这样，当需要少量操作码，同时又能制定需要不限数量的操作码的标准时，允许定义更有效的代码表。

上述任一种操作码结构均通过 GB 10022 中定义的标识机制进行标识。

5.2 紧致操作码编码

在需要 32 或少于 32 个操作码的情况下，只需从代码表列 2 和列 3 中的 32 个代码表位置中为每个操作码分配一个代码表位置便可简单地完成操作码的编码。这类操作码的通用结构如图 1 所示。

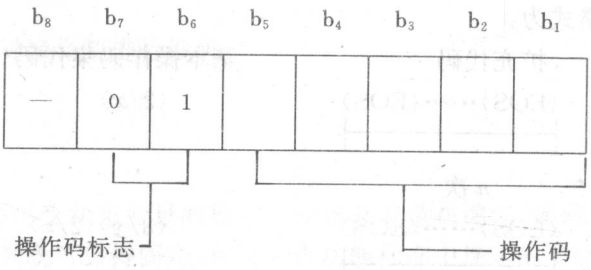


图 1 紧致操作码编码结构

5.3 可扩充操作码编码

在可能需要操作码的数量不受限制的情况下，对操作码的编码需要将操作码分为：

a. 基本操作码集；

b. 扩充操作码集。

5.3.1 中给出了对基本操作码集编码技术的描述。5.3.2 则给出了对扩充机制的描述。

5.3.1 基本操作码集的编码技术

基本操作码集由单字节及双字节的操作码组成。这类操作码的通用结构如图 2 所示。

对于单字节操作码，其操作码长度指示符 b_5 位为 0(列 2 的操作码)，b_4 至 b_1 位用作操作码的编码。

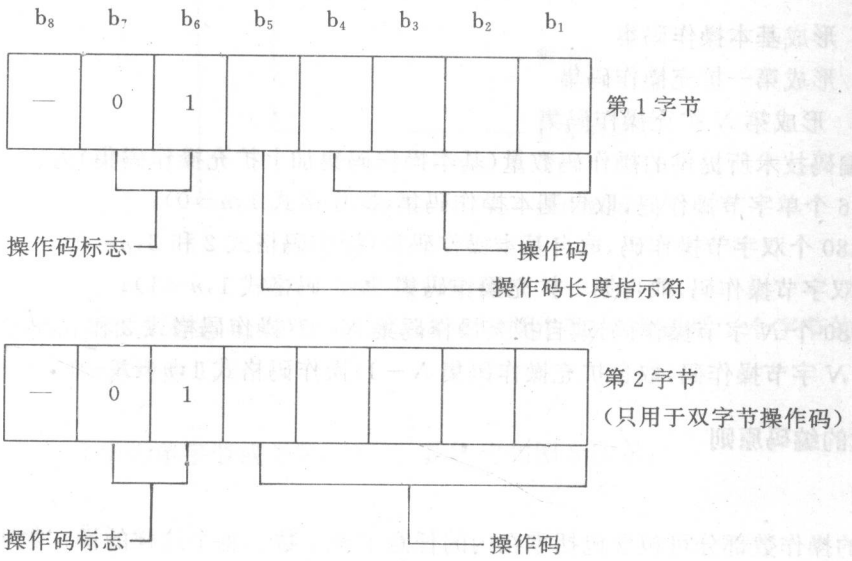


图 2 操作码编码结构

对于双字节操作码，其操作码长度指示符——第 1 字节的 b_5 位为 1，第 1 字节的 b_4 至 b_1 位及第 2

字节的 b_5 至 b_1 位则用作操作码的编码。

位组 3/15，扩充操作码间隔(EOS)则另有它用(见 5.3.2)。

因此，这种编码技术便可提供一个包含 496 个操作码的基本操作码集，它们是：

a. 16 个单字节操作码(代码表中列 2)；

b. 480(即 15×32)个双字节操作码(第 1 字节取自列 3 中除 3/15 以外的位组，第 2 字节则取自列 2 或列 3)。

5.3.2 扩充机制

可以利用扩充操作码间隔(EOS，3/15)将基本操作码集扩充为可容纳不限数量的操作码的扩充操作码集。

第 N 个扩充操作码集由基本操作码集中的操作码和前置的 N 个 EOS 组成。第 N 个扩充操作码集中的操作码的三种可能的格式为：

操作码格式	扩充代码	基本操作码集代码
1	〈EOS〉……〈EOS〉（n 次）	〈2/x〉
2	〈EOS〉……〈EOS〉（n 次）	〈3/y〉〈2/z〉
3	〈EOS〉……〈EOS〉（n 次）	〈3/y〉〈3/z〉

EOS=3/15

x=0、1、……、15

y=0、1、……、14

z=0、1、……、15

n=0、1、……、……

这里：

n=0　形成基本操作码集

n=1　形成第一扩充操作码集

n=N　形成第 N 扩充操作码集

这种编码技术所提供的操作码数量(基本操作码集加上扩充操作码集)为：

a. 16 个单字节操作码，取自基本操作码集(操作格式 1，n=0)；

b. 480 个双字节操作码，取自基本操作码集(操作码格式 2 和 3，n=0)；

16 个双字节操作码，取自第一扩充操作码集(操作码格式 1，n=1)；

c. 480 个 N 字节操作码，取自扩充操作码集 $N-2$(操作码格式 2 和 3，n=$N-2$)；

16 个 N 字节操作码，取自扩充操作码集 $N-1$(操作码格式 1，n=$N-1$)。

6 操作数的编码原则

6.1 概述

图原的操作数部分可包含包括零在内的任意个操作数。每个这样的操作数可由单字节或多字节构成。

操作数字节的一般格式由图 3 给出。

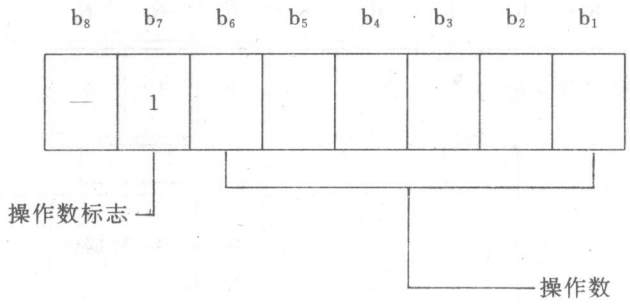


图 3　操作数编码结构

6.2　格式定义

可以用下列三种格式对操作数进行编码：

a.　基本格式；

b.　位流格式；

c.　串格式。

此外，操作数的编码还可受状态变量的控制。该状态变量在编码/译码过程初始化时由具体应用置为所需的一个值。状态变量既可保持固定，也可以由功能标准中规定的功能进行动态修改，这取决于本标准的这一部分中定义的编码原则在何种功能标准下使用。

本标准的这一部分中所用的状态变量在附录 A(补充件)中列出。

本标准的这一部分只是参照状态变量的值，但并不涉及允许修改它们的功能定义。

6.2.1　基本格式

基本格式中的操作数表示为单字节或多字节序列，其结构如图 4 所示。

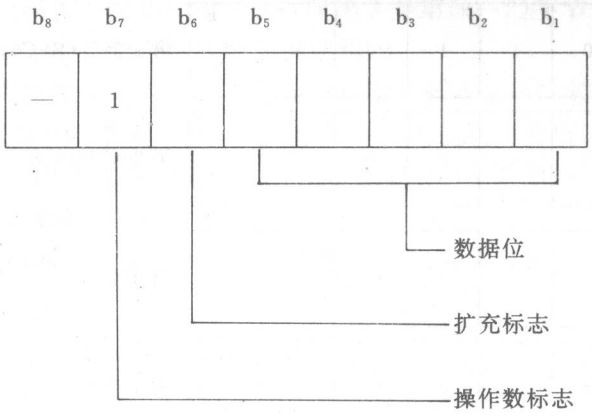


图 4　基本格式结构

对于单字节的操作数，其扩充标志 b_6 位为 0。而在多字节操作数中，除最后一个字节的扩充标志为 0 外，其所有字节中的扩充标志均为 1。

6.2.2　位流格式

位流格式中的操作码表示为单字节或多字节序列，其结构如图 5 所示：

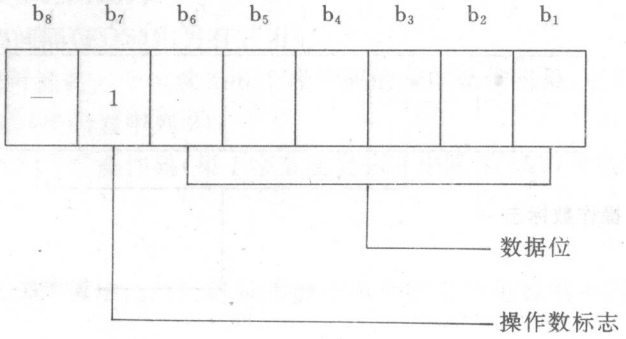


图 5　位流格式结构

位流格式中的数据从位流数据有效部分的第 1 字节的高位至低位连续装在操作数字节的数据位中。

位流格式操作数的末端无法由位流格式自身导出(该格式不能自定界)。

为位流格式操作数末端定界的是:

a. 下一个操作码;或

b. 〈块结束〉值,它标识了以位流格式编码的操作数中的数据的结束;或

c. 状态变量值,它定义了以位流格式编码的操作数的长度。

当以位流格式编码的数据与所有字节的总数不匹配时,则最高字节的剩余位应用"0"填满。

6.2.3　串格式

串格式中的操作数以字节序列形式编码,其结构如图 6 和图 7 所示。

图 6　七位环境中的串格式结构(用 c 标记数据位)

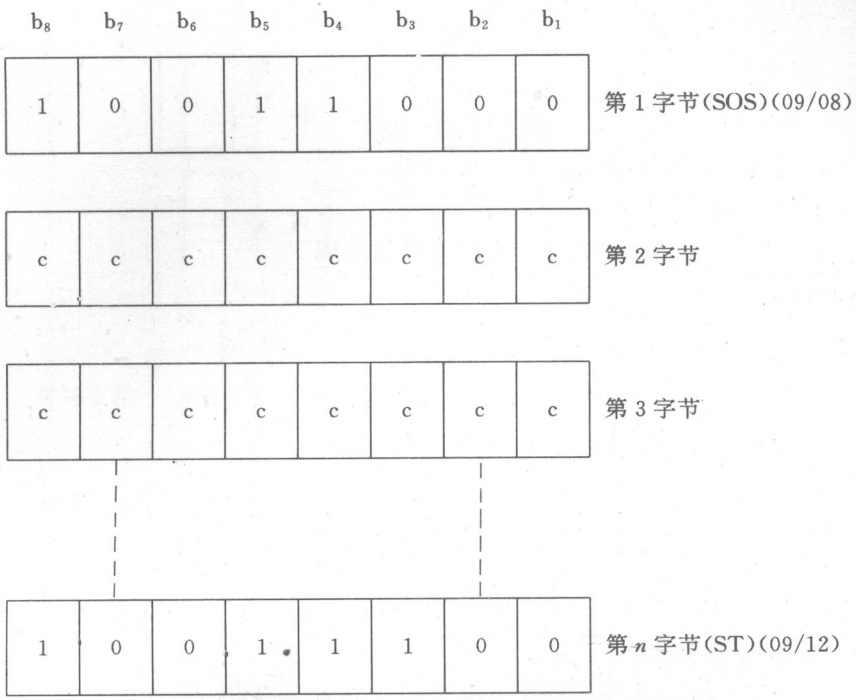


图 7　八位环境中的串格式结构(用 c 标记数据位)

字符串的开始由定界符串开始(SOS)标明。该定界符在七位环境中由控制序列 ESC 5/8 表示(ESC=1/11),而在八位环境中则由 09/08 表示。

字符串的结束由定界符串终止符(ST)标明。该定界符在七位环境中由转义序列 ESC 5/12 表示(ESC=1/11),而在八位环境中则由 09/12 表示。

在七位环境(图 6)中,代码表中列 0～7 的位组,即第 3 至第(n－2)字节,可用作数据字节。使用串格式操作数的标准可能限制代码表列 0～1 中位组的使用。

在八位环境(图 7)中,代码表中列 00～15 的位组,即第 2 至第(n－1)字节,可用作数据字节。使用串格式操作数的标准可能限制代码表列 00～01 和列 08～09 中位组的使用。

在七位环境中,对一个串操作数编码所需的字节数等于串的字符数加 4(这 4 个字节用作 SOS 和 ST,分别编码为 ESC　5/8 和 ESC5/12)。

在八位环境中,对一个串操作数编码所需的字节数等于串的字符数加 2(这两个字节用作 SOS 和 ST,分别编码为 09/08 和 09/12)。

串的编码是 6.1 中所述通用编码规则的唯一例外。

6.3　通用数据类型

本条中描述了使用 6.2 中定义的格式的几种通用数据类型的编码。

这些通用数据类型可以用于图片表示标准中。如果某个特定的标准使用了某通用数据类型,则那个数据类型在该标准中的编码必须与本条中所述的通用数据类型的编码一致。

6.3.1　无符号整数

无符号整数用基本格式或位流格式进行编码,而用哪种格式则由使用本标准编码原则的功能标准确定。

每个无符号整数表示为单字节或多字节序列。数据位则从操作数最有效部分的第 1 字节的最高位至最低位开始连续装入各字节中。

6.3.1.1　基本格式中的无符号整数

以基本格式编码的无符号整数的例子见图 8 和图 9。

b_8	b_7	b_6	b_5	b_4	b_3	b_2	b_1
—	1	0	1	1	1	1	1

操作数数值:31

图 8　无符号整数编码

b_8	b_7	b_6	b_5	b_4	b_3	b_2	b_1	
—	1	1	0	0	0	1	0	第 1 字节
—	1	1	0	0	0	0	0	第 2 字节
—	1	0	1	1	1	1	1	第 3 字节

操作数数值:2079

图 9　无符号整数编码(多于 1 个字节)

6.3.1.2　位流格式中的无符号整数

操作数所用的字节数由无符号整数长度状态变量决定。

以位流格式编码的无符号整数的例子见图 10 和图 11。

b_8	b_7	b_6	b_5	b_4	b_3	b_2	b_1
—	1	1	1	1	1	1	1

操作数数值:63

无符号整数长度:1

图 10　位流格式的无符号整数编码(1 字节)

	b_8	b_7	b_6	b_5	b_4	b_3	b_2	b_1	
	—	1	0	0	0	0	0	0	第 1 字节
	—	1	0	1	1	1	1	1	第 2 字节

操作数数值:31

无符号整数长度:2

	b_8	b_7	b_6	b_5	b_4	b_3	b_2	b_1	
	—	1	1	0	0	0	0	0	第 1 字节
	—	1	0	1	1	1	1	1	第 2 字节

操作数数值:2079

无符号整数长度:2

图 11　位流格式的无符号整数编码(多字节)

6.3.2　带符号整数

带符号整数用模数与符号记数法或二进制补码记数法进行表示。在这两种情况下,都可用基本格式或位流格式进行编码。

带符号整数表示为单字节或多字节序列。数据位则从操作数最有效部分第一字节的高位开始至低位连续装入各字节中。

6.3.2.1　使用基本格式的模数与符号记数法中的带符号整数

带符号的整数区域分为非负数区域和负数区域,第 1 字节的 b_5 位将用作符号位,即:

a.　若 b_5 位置"0",则该整数为非负数;

b.　若 b_5 位置"1",则该整数为负数。

正零被视为非负数。

负零的编码只限于特定编码用。带符号整数不允许具有负零值。

第 1 字节的 b_4～b_1 位和后面字节的 b_5～b_1 位用作带符号整数的模的编码。

图 12 包含了带符号整数编码的例子。

b_8	b_7	b_6	b_5	b_4	b_3	b_2	b_1	
—	1	0	1	0	0	1	0	

操作数数值：－2

b_8	b_7	b_6	b_5	b_4	b_3	b_2	b_1	
—	1	1	1	1	0	0	0	第 1 字节
—	1	0	0	0	0	0	0	第 2 字节

操作数数值：－256

图 12　使用基本格式的模数与符号记数法中的带符号整数

6.3.2.2　使用基本格式的二进制补码记数法中的带符号整数

带符号整数的区域分为非负数区域和负数区域，第 1 字节的 b_5 位将用作符号位，即：

a.　若 b_5 位置“0”，则该整数为非负数；

b.　若 b_5 位置“1”，则该整数为负数。

负数由二进制补码表示。

正零被视为非负数。

负零的编码只限于特定编码用。带符号整数不允许具有负零值。

图 13 包含了使用基本格式的二进制补码记数法中的带符号整数的编码的例子。

b_8	b_7	b_6	b_5	b_4	b_3	b_2	b_1	
—	1	0	0	0	1	1	1	

操作数数值：＋7

b_8	b_7	b_6	b_5	b_4	b_3	b_2	b_1	
—	1	1	1	1	1	1	0	第 1 字节
—	1	0	0	0	1	0	1	第 2 字节

操作数数值：－59

图 13　使用基本格式的二进制补码记数法中的带符号整数编码

6.3.2.3　使用位流格式的模数与符号记数法中的带符号整数

操作数所用的字节数由带符号整数长度状态变量决定。

带符号的整数区域分为非负数区域和负数区域，第 1 字节的 b_6 位用作符号位，即：

a. 若 b_6 位置“0”，则该整数为非负数；

b. 若 b_6 位置“1”，则该整数为负数。

正零被视作非负数。

负零的编码只限于特定编码用。带符号整数不允许具有负零值。

第 1 字节的 b_5～b_1 位和后面字节的 b_6～b_1 位用作带符号整数的模的编码。

图 14 包含了使用位流格式的模数与符号记数法中的带符号整数的编码的例子。

b_8	b_7	b_6	b_5	b_4	b_3	b_2	b_1
—	1	0	0	0	1	1	1

操作数数值：+7

带符号整数长度=1

b_8	b_7	b_6	b_5	b_4	b_3	b_2	b_1	
—	1	1	0	0	0	0	0	第 1 字节
—	1	1	1	1	1	1	1	第 2 字节

操作数数值：−59

带符号整数长度=2

图 14　使用位流格式的模数与符号记数法中的带符号整数编码

6.3.2.4　使用位流格式的二进制补码记数法中的带符号整数

操作数所用的字节数由带符号整数长度状态变量决定。

带符号的整数区域分为非负数区域和负数区域，第 1 字节的 b_6 位用作符号位，即：

a. 若 b_6 位置“0”，则该整数为非负数；

b. 若 b_6 位置“1”，则该整数为负数。

负数用二进制补码形式表示。

图 15 包含了使用位流格式的二进制补码记数法中带符号整数编码的例子。

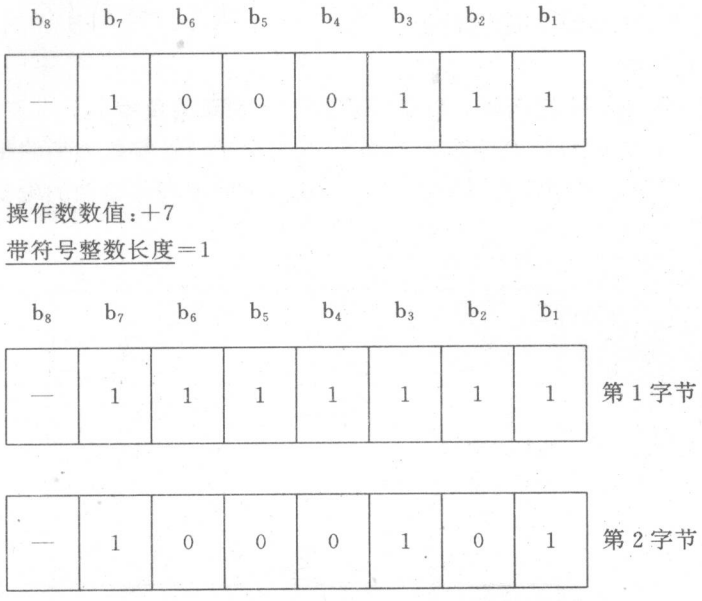


图 15　使用位流格式的二进制补码记数法中的带符号整数编码

6.3.3　实数

实数由尾数部分和一个指数部分组成，尾数和指数均由基本格式或位流格式中的带符号整数表示，既可使用模数和符号记数法，也可使用二进制补码记数法。

当使用模数和符号记数法时，尾数和指数不允许出现负零，该值留作将来使用。

指数是尾数用于所乘的 2 的幂次。指数可隐含地定义为缺省指数，这时在实数中可以略去指数。否则指数便会被编码为实数的第二个部分：

〈实数〉=〈尾数部分〉[〈指数部分〉]

指数是否会作为实数格式的一部分而明确地给出其编码取决于：

a.　状态变量实数显式指数的当前值；

b.　实数的尾数部分中的指数后继标志的值。

若实数显式指数的当前值为允许，则尾数第 1 字节的一个位便用作指数后继位：

a.　若有显式指数后随尾数，则为 1；

b.　若无指数后随尾数，则为 0。

指数后继位的顺序取决于尾数采用的编码方式(基本或位流格式)，它在 6.3.3.1 和 6.3.3.2 中规定。

若实数显式指数的当前值为禁止，则尾数第 1 字节中的指数后继位便用作数据位。

若实数显式指数的当前值为允许且尾数中的指数后继位为 1，则尾数后将跟随一个显式的指数。

若实数显式指数的当前值为禁止，或实数显式指数的当前值为允许，但尾数部分中的指数后继位的值为 0，则编码中便不出现指数，并同时设定了一个隐含的缺省指数值。该值由状态变量实数缺省指数给出。

6.3.3.1　基本格式中的实数

在基本格式中，尾数是一个带符号整数，它既可用模数与符号记数法，也可用二进制补码记数法表示。尾数的第 1 个字节中，b_5 位用作符号位。

若实数显式指数状态变量为允许，则尾数第 1 字节的 b_4 位用作指数后继位：

a. 若无指数后随尾数，则为 0；

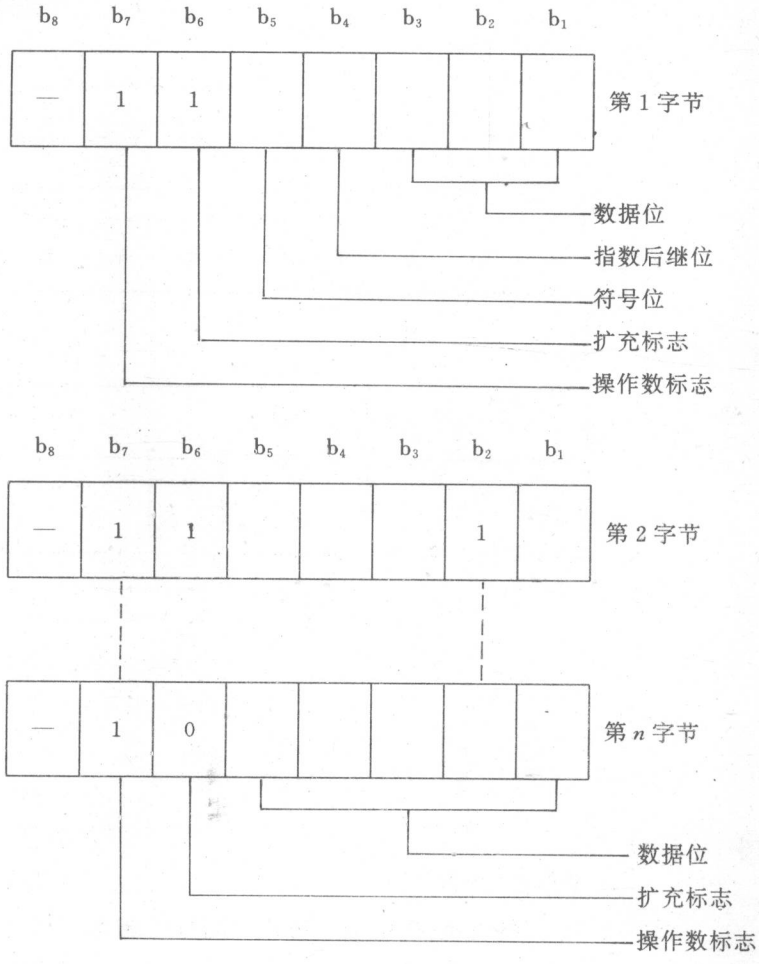
b. 若有显式指数后随尾数，则为 1。

若实数显式指数状态变量为禁止，则尾数第 1 字节的 b_4 位用作数据位。

在基本格式中，指数编码为带符号整数，它既可使用模数与符号记数法，也可使用二进制补码记数法表示。

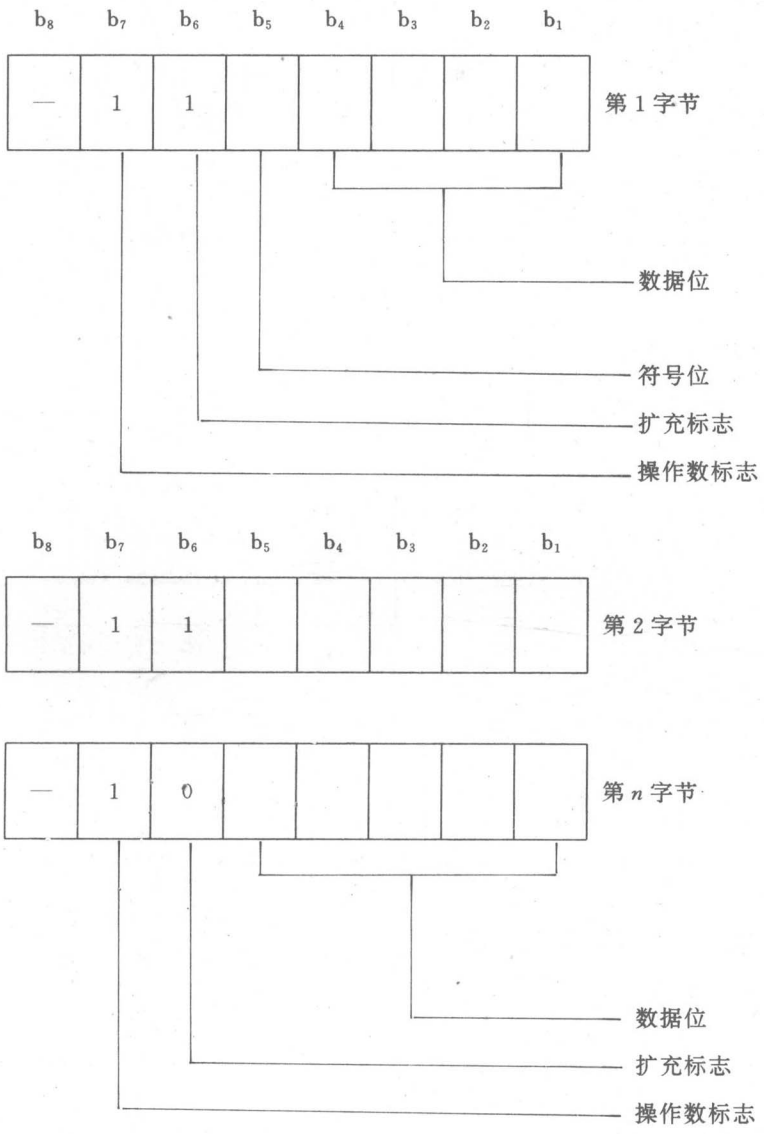
6.3.2 详述了带符号整数的编码。

图 16 和图 17 给出了尾数的通用格式，图 18 给出了多字节尾数的例子。



实数显式指数＝允许

图 16　使用基本格式的尾数的编码



实数显式指数＝禁止

图 17　使用基本格式的尾数的编码