ICS 35.040
L 71

中华人民共和国国家标准

GB/T 27766—2011

二维条码　网格矩阵码

Two-dimensional barcode—Grid matrix code

2011-12-30 发布　　2012-05-01 实施

中华人民共和国国家质量监督检验检疫总局
中国国家标准化管理委员会　发布

目　次

前　言

本标准按照 GB/T 1.1—2009 给出的规则起草。

本标准由中华人民共和国工业和信息化部提出。

本标准由全国物品编码标准化技术委员会(SAC/TC 287)归口。

本标准起草单位:武汉矽感科技有限公司、中国电子技术标准化研究所。

本标准主要起草人:张伟、张也平、刘波、张得煜、樊旭川。

引　言

本文件的发布机构提请注意，声明符合本文件时，可能涉及第5章、第6章、第9章、第10章相关的专利的使用。

本文件的发布机构对于该专利的真实性、有效性和范围无任何立场。

该专利持有人已向本文件的发布机构保证，他愿意同任何申请人在合理且无歧视的条款和条件下，就专利授权许可进行谈判。该专利持有人的声明已在本文件的发布机构备案。相关信息可通过以下联系方式获得：

专利所有人：　武汉矽感科技有限公司

地址：　武汉市东西湖区吴家山经济开发区金一路　武汉矽感光电产业园

邮政编码：　430040

网址：　http://www.syscantech.cn

联系人：　何柳青

联系电话：　027-61675589

传真：　027-61675592

E-mail：　helq@syscangroup.com

请注意除上述专利外，本文件的某些内容仍可能涉及专利。本文件的发布机构不承担识别这些专利的责任。

二维条码　网格矩阵码

1　范围

本标准规定了网格矩阵码的符号结构、信息编译码方法、纠错编译码方法、信息排布方法、参考译码算法以及符号质量要求等技术内容。

本标准适用于网格矩阵码的生成与识读。

2　规范性引用文件

下列文件对于本文件的应用是必不可少的。凡是注日期的引用文件，仅注日期的版本适用于本文件。凡是不注日期的引用文件，其最新版本(包括所有的修改单)适用于本文件。

GB/T 1988　信息技术　信息交换用七位编码字符集

GB/T 12905　条码术语

GB 18030　信息技术　中文编码字符集

GB/T 23704　信息技术　自动识别与数据采集技术　二维条码符号印制质量的检验

ISO/IEC 15424　信息技术　自动识别与数据采集技术　数据载体标识符

AIM 国际技术规范　扩展解释：第 1 部分：识别方案与协议(简称“AIM ECI 规范”)

3　术语、定义、缩略语和约定

3.1　术语和定义

GB/T 12905 中界定的以及下列术语和定义适用于本文件。

3.1.1

纠错块　error correction codeword block

对码字分组后用于纠错的一组码字。

3.1.2

边框　frame

宏模块的最外 20 个单元模块，这些单元模块同为深色(低反射率)或同为浅色(高反射率)。

3.1.3

层　layer

环绕中心宏模块的宏模块圈。

3.1.4

层标识号　layer ID number

赋予宏模块左上角的两个单元模块的值，该值根据纠错等级以及宏模块所在的层号确定，可用于指明 GM 码的方向。

3.1.5

宏模块　macromodule

GM 码的子结构，由 6×6 个单元模块组成。

3.1.6

单元模块 module

组成 GM 码的基本单元，每个单元模块表示一个二进制位。

3.1.7

填充位 padding bit

用于填充数据位流最后一个码字后面容量的无含义位，其值为 0。

3.1.8

填充码字 padding codeword

当数据码字和纠错码字不能填满 GM 码的容量时，用于填充 GM 码的剩余容量的码字。填充码字不表示有效数据，但参与 Reed-Solomon 纠错运算。

3.1.9

版本 version

用于表示 GM 码规格的序列号。

3.1.10

功能码 function code

用于指示属于特定应用或特定功能的 GM 码符号的代码。

3.1.11

纠错等级 error correction level

指明 GM 码中纠错码字所占比例的参数。

3.2 缩略语

下列缩略语适用于本文件：

ABS——绝对值(Absolute Value)

DIV——整除运算(Division)

ECI——扩展解释(Extended Channel Interpretation)

FNC——功能码(Function Code)

GF——伽罗瓦有限域(Galois Field)

GM 码——网格矩阵码(Grid Matrix Code)

MOD——模运算，求整除后的余数(Modulus)

3.3 约定

下列表示适用于本文件：

$(\cdots)_{BIN}$——表示括号中的内容使用二进制表示

$(\cdots)_{HEX}$——表示括号中的内容使用十六进制表示

$\lfloor x \rfloor$——表示不超过 x 的最大整数

$\lceil x \rceil$——表示不小于 x 的最小整数

4 符号描述

4.1 基本特征

4.1.1 可编码信息

GM 码可编码以下信息：

a) 数字字符(数字 0～9,GB/T 1988 中值 48 至 57);
b) 大写字母(字母 A～Z,GB/T 1988 中值 65 至 90);
c) 小写字母(字母 a～z,GB/T 1988 中值 97 至 122);
d) 汉字字符(GB 18030);
e) 8 位字节型数据。

4.1.2 数据表示法

深色单元模块表示二进制"1",浅色单元模块表示二进制"0"。

4.1.3 符号规格

GM 码的规格为 3×3 宏模块到 27×27 宏模块,对应于版本 1 到版本 13,每一版本 GM 码比前一版本每边增加 2 个宏模块,见表 1。

表 1 各版本 GM 码的结构

版本	宏模块数	单元模块数 (不包括空白区)	层数 (不包括中心宏模块)	总码字数 (数据+纠错码字)
1	3×3	18×18	1	18
2	5×5	30×30	2	50
3	7×7	42×42	3	98
4	9×9	54×54	4	162
5	11×11	66×66	5	242
6	13×13	78×78	6	338
7	15×15	90×90	7	450
8	17×17	102×102	8	578
9	19×19	114×114	9	722
10	21×21	126×126	10	882
11	23×23	138×138	11	1058
12	25×25	150×150	12	1250
13	27×27	162×162	13	1458

4.1.4 符号容量

使用最低纠错等级的最大版本 GM 码(纠错 1 级版本 13)的容量如下:

a) 2 751 个数字;
b) 1 836 个大写字母;
c) 1 836 个小写字母;
d) 1 529 个数字字母混合字符;
e) 705 个 GB 18030 双字节 1 区或双字节 2 区内的字符,或 571 个 GB 18030 双字节字符,或 285 个 GB 18030 四字节字符;
f) 1 143 个字节。

4.1.5 纠错等级

版本 1 的 GM 码有 2 级到 5 级纠错，版本 2 到版本 13 的 GM 码有 1 级到 5 级纠错，每级中纠错码字数占总码字数的比例为：

a) 1 级：10%（不适用于版本 1）；

b) 2 级：20%；

c) 3 级：30%；

d) 4 级：40%；

e) 5 级：50%。

纠错码字的个数为总码字个数的上述百分比（向下舍入），见附录 A。

4.2 附加特征

4.2.1 结构链接

允许用不多于 16 个的 GM 码在逻辑上连续地表示数据文件。在多顺序扫描状态下应保持原始顺序与数据正确连接。

4.2.2 支持 ECI 协议

ECI 协议（见“AIM ECI 规范”）使 GM 码可以表示缺省字符集以外的字符（如阿拉伯字符、古斯拉夫字符、希腊字符等），及其他数据解释（如用一定的压缩方式表示的数据），或者具体应用的编码要求。

5 符号结构

5.1 概述

GM 码由深色边宏模块和浅色边宏模块交错排列而成的正方形宏模块矩阵组成，矩阵每边为奇数个宏模块，且 GM 码的中心与四个角上均为深色边宏模块，GM 码的四周为空白区，见图 1。

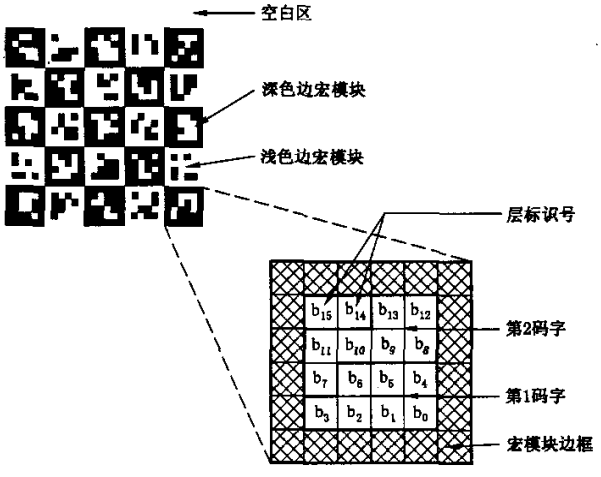


图 1 GM 码结构图

版本 2 纠错 5 级的 GM 码示意图见图 2。

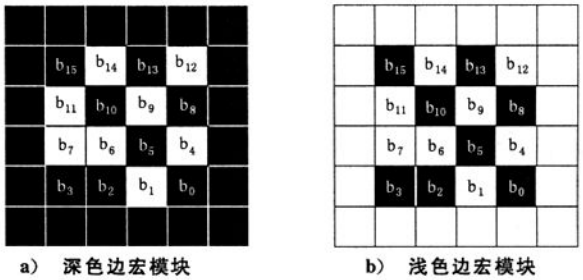
图 2　版本 2 纠错 5 级的 GM 码示意图

5.2　宏模块结构

宏模块的内部结构见图 3，包括边框、两个 7 位的码字和层标识号。

每个宏模块由 6×6 个单元模块无缝排列而成，深色边宏模块的最外一圈单元模块全部是深色，浅色边宏模块的最外一圈单元模块全部是浅色。宏模块的最外一圈单元模块不表示数据，用于识别与定位。

在 20 个边框单元模块内部总共有 16 个单元模块：b_0，b_1，…，b_{15}。每个单元模块表示 1 位二进制数，深色对应“1”，浅色对应“0”。b_{15} 和 b_{14} 单元模块用来表示层标识号，b_{15} 为高位。b_6 到 b_0 表示第 1 个码字，b_{13} 到 b_7 表示第 2 个码字，b_{13} 和 b_6 分别是码字的最高位。



a)　深色边宏模块　　b)　浅色边宏模块

图 3　宏模块结构

图 3 中两个宏模块的数据均为$(2D)_{HEX}$和$(4A)_{HEX}$。图中第 1 个码字 $b_6 \cdots b_0$ 为$(0101101)_{BIN}$，即$(2D)_{HEX}$；第 2 个码字 $b_{13} \cdots b_7$ 为$(1001010)_{BIN}$，即$(4A)_{HEX}$。层标识号 $b_{15}b_{14}$ 为$(10)_{BIN}$，即$(2)_{HEX}$。

5.3　宏模块的分层

GM 码由边长为奇数个宏模块的方阵组成。见图 4，方阵中心的宏模块称为中心宏模块，中心宏模块(第 0 层)周围的 8 个宏模块为第 1 层宏模块，第 1 层宏模块外侧的 16 个宏模块为第 2 层宏模块，……，直至最外层宏模块。

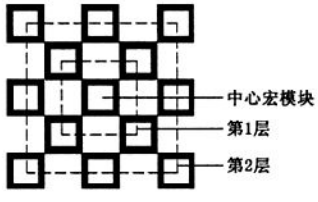


图 4　宏模块的分层

宏模块的层数(不包括中心宏模块)等于 GM 码的版本(见表 1)。

5.4 层标识号

每个宏模块都有一个层标识号，层标识号的取值为 0～3。同一层宏模块的层标识号相同。宏模块的层标识号由 GM 码的纠错等级和该宏模块所在的层号共同决定。表 2 是不同纠错等级的 GM 码各层宏模块的层标识号。

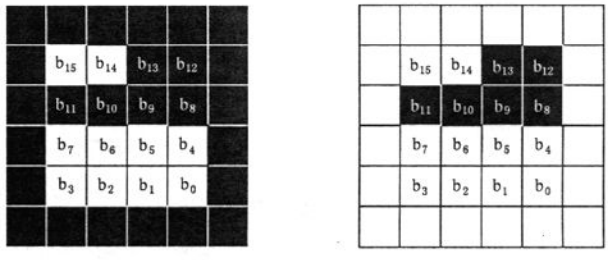
表 2 层标识号分布

纠错等级	从中心到第 13 层													
	中心	1	2	3	4	5	6	7	8	9	10	11	12	13
5	0	1	2	3	0	1	2	3	0	1	2	3	0	1
4	1	2	3	0	1	2	3	0	1	2	3	0	1	2
3	2	3	0	1	2	3	0	1	2	3	0	1	2	3
2	3	0	1	2	3	0	1	2	3	0	1	2	3	0
1	3	2	1	0	3	2	1	0	3	2	1	0	3	2

5.5 填充码字

当数据码字和纠错码字不能正好填满 GM 码的容量时，在数据码字后加入填充码字。

当宏模块的第 1 码字（b_8 到 b_0）是填充码字时，应填充 $(0000000)_{BIN}$；当第 2 码字（b_{13} 到 b_7）是填充码字，并且是码字流中的第 1 个填充码字时，应填充 $(0000000)_{BIN}$，否则应填充 $(1111110)_{BIN}$，见图 5。



a） 填充的深色边宏模块　　b） 填充的浅色边宏模块

图 5 填充的宏模块

5.6 空白区

空白区为环绕在 GM 码四周的不小于 6 个单元模块宽的区域，其反射率应与浅色单元模块相同。

6 符号生成

6.1 生成过程

GM 码的生成过程包括以下六个步骤：

a） 数据分析：分析输入的数据，确定数据的数据编码模式。对不同的数据类型，GM 码采用不同的数据编码模式进行编码，见 6.3。每种模式有各自的编码规则。

b） 数据编码：将输入数据按照其编码模式对应的编码规则转换为位流。当需要进行模式切换时，

在新模式数据编码前输出模式转换码。将编码产生的位流按每7位对应一个码字的方式转换为数据码字流，最后一个码字不足7位时用0填充。

c) 计算GM码版本：用户应选取可接受的最小纠错等级，根据表1可得到能容纳数据码字和纠错码字的GM码版本。若用户未选取纠错等级，使用推荐的纠错等级（见6.6.2）计算GM码版本。根据该GM码版本，采用可以容纳给定数据的最高纠错等级，并在码字流的最后添加需要的填充码字。

d) 纠错编码：若数据码字和纠错码字总数大于127，应将数据码字进行分块（见6.6.3）。对每块码字分别生成纠错码字，并将纠错码字添加到该块数据码字的后面。

e) 在矩阵中布置网格图形：根据GM码的版本和纠错等级，将每个宏模块的边框以及层标识号排列到矩阵中。

f) 排列数据码字和纠错码字：若码字被分块，则对各块码字进行交错排列后得到一个单一的码字流。将码字流按顺序排列到矩阵中，完成编码。

6.2 数据分析

对输入数据进行类型分析，按类型划分成多个段，使编码得到的位流尽量短。位流长度优化的一种方法参见附录B。

6.3 模式指示

6.3.1 模式分类

GM码的编码模式分数据编码模式、ECI模式和功能码模式三类，各种模式由确定的模式指示符指示。表3列出了所有的模式指示符。

表3 模式指示符

模式分类	模式名称	模式指示符	说明
数据编码模式	汉字模式	$(0001)_{BIN}$	每个字符用13位二进制进行编码。见6.4.1
	数字模式	$(0010)_{BIN}$	每3个字符用10位二进制进行编码。见6.4.2
	小写字母模式	$(0011)_{BIN}$	每个字符用5位二进制进行编码。见6.4.3
	大写字母模式	$(0100)_{BIN}$	每个字符用5位二进制进行编码。见6.4.4
	数字字母混合模式	$(0101)_{BIN}$	每个字符用6位二进制进行编码。见6.4.5
	控制字符模式[a]	—	每个字符用6位二进制进行编码。见6.4.6
	字节模式	$(0111)_{BIN}$	每个字符用8位二进制进行编码。见6.4.7
ECI模式	ECI	$(1100)_{BIN}$	见6.4.8
功能码模式	ENC1	$(1000)_{BIN}$	功能码1，GS1应用标识。见6.4.9.1
		$(1011)_{BIN}$	功能码1，AIM应用标识。见6.4.9.1
	FNC2	$(1001)_{BIN}$	功能码2，结构链接功能。见6.4.9.2
	FNC3	$(1010)_{BIN}$	功能码3，识读设备初始化数据。见6.4.9.3
[a] 只允许从小写字母模式、大写字母模式或数字字母混合模式进行切换（见6.4.6.2和6.5.1）。			

6.3.2 数据编码模式

数据编码模式包括汉字模式、数字模式、小写字母模式、大写字母模式、数字字母混合模式、控制字

符模式和字节模式，见表 3。

6.3.3 ECI 模式

ECI 模式只能出现在数据的开头或“模式结束”转换码（见 6.5.1）之后。ECI 模式的模式指示符之后为 ECI 任务号，编码方法见 6.4.8。

6.3.4 功能码模式

功能码分 FNC1、FNC2 和 FNC3 三类，其中 FNC1 包括两种模式指示符，分别对应两种应用标识，见表 3。功能码只能在 GM 码的开头出现。一个 GM 码使用功能码时，其模式指示符应出现在数据编码位流的前面。一个 GM 码最多可以使用两个功能码。

6.3.5 无效的模式指示符

模式指示符$(0000)_{BIN}$、$(0110)_{BIN}$、$(1101)_{BIN}$、$(1110)_{BIN}$和$(1111)_{BIN}$表示无效。

6.4 数据编码模式

6.4.1 汉字模式

6.4.1.1 编码字符

可编码字符包括：

a) GB 18030 双字节 1 区及双字节 2 区的字符（即第一字节值在$(A1)_{HEX}$至$(A9)_{HEX}$或$(B0)_{HEX}$至$(F7)_{HEX}$之间，且第二字节值在$(A0)_{HEX}$至$(FF)_{HEX}$之间的部分）；

b) “回车换行”（GB/T 1988 中值 13、10 的组合）；

c) 数字对“00”到“99”；

d) 8 位字节型数据。

注：GB 18030 除双字节 1 区及双字节 2 区以外的字符不能用汉字模式编码，可用字节模式编码。

6.4.1.2 编码规则

汉字模式采用 13 位二进制进行编码。

当一个 GB 18030 双字节字符第一字节值在$(A1)_{HEX}$至$(A9)_{HEX}$之间，且第二字节值在$(A0)_{HEX}$至$(FF)_{HEX}$之间时，按式(1)计算该字符的 13 位编码：

$$N=(C_1-(A1)_{HEX})\times(60)_{HEX}+(C_2-(A0)_{HEX}) \quad \cdots\cdots(1)$$

式中：

N——字符的 13 位编码；

C_1——GB 18030 编码的第一字节值；

C_2——GB 18030 编码的第二字节值。

当一个 GB 18030 双字节字符第一字节值在$(B0)_{HEX}$至$(F7)_{HEX}$之间，且第二字节值在$(A0)_{HEX}$至$(FF)_{HEX}$之间时，按式(2)计算该字符的 13 位编码：

$$N=(C_1-(B0)_{HEX}+9)\times(60)_{HEX}+(C_2-(A0)_{HEX}) \quad \cdots\cdots(2)$$

式中：

N——字符的 13 位编码；

C_1——GB 18030 编码的第一字节值；

C_2——GB 18030 编码的第二字节值。

式(1)及式(2)定义了 0 至 7775 之间的编码值，以下方式用于定义 7776 至 8191 的编码值：

a) 7776 赋给“回车换行”符；

b) 7777 至 8032 赋给 8 位字节数据(0 至 255)，用于编码混在汉字信息中的非汉字数据，减小个别非汉字模式的数据嵌在一段汉字中导致的模式转换开销；

c) 8033 至 8132 赋给数字对“00”到“99”；

d) 8160 至 8165 用于实现模式的转换，见 6.5.1；

e) 编码值 8133 至 8159 及编码值 8166 至 8191 是无效的。

两个编码示例见表 4。

表 4 汉字编码示例

步骤	说明	例 1	例 2
1	输入字符	¥	多
2	GB 18030 编码	$(A3A4)_{HEX}$	$(B6E0)_{HEX}$
3	代入公式(1)或公式(2)	$((A3)_{HEX}-(A1)_{HEX})\times(60)_{HEX}+((A4)_{HEX}-(A0)_{HEX})$	$((B6)_{HEX}-(B0)_{HEX}+9)\times(60)_{HEX}+((E0)_{HEX}-(A0)_{HEX})$
4	计算结果	$(C4)_{HEX}$	$(5E0)_{HEX}$
5	转化为 13 位二进制值	0000011000100	0010111100000

6.4.2 数字模式

6.4.2.1 编码字符

可编码字符包括：

a) 数字 0 至 9(GB/T 1988 值 48～57)；

b) “空格”(GB/T 1988 值 32)；

c) “+”(GB/T 1988 值 43)；

d) “－”(GB/T 1988 值 45)；

e) “.”(GB/T 1988 值 46)；

f) “,”(GB/T 1988 值 44)；

g) “回车换行”(GB/T 1988 值 13、10 的组合)。

6.4.2.2 编码规则

以连续的三个数字为一组将数据分组，每 3 个数字采用 10 位二进制进行编码。遇到非数字字符则将该字符包含到分组中，每组中最多只能有一个非数字字符，多余的非数字字符不能按数字模式编码。末尾一组不够三个数字用 0 填充。在输出第一组数字的编码前先输出 2 位计数器，记录最后一个分组填充的数字个数，译码时根据该计数器丢弃填充数字：

a) “00”表示没有填充数字；

b) “01”表示有 1 个填充数字；

c) “10”表示有 2 个填充数字；

d) “11”为无效编码。

编码只有数字字符的组时，按式(3)计算该组的 10 位编码：

$$N = 100D_1 + 10D_2 + D_3 \qquad \cdots\cdots(3)$$

式中：

N ——数字组的 10 位编码；

D_1——数字组的第一个数字；

D_2——数字组的第二个数字；

D_3——数字组的第三个数字。

当分组中包含非数字字符时，非数字字符出现在分组中的位置有三种情况，分别是（X 表示非数字字符）：第 1 位置为 $X\ D_1\ D_2\ D_3$；第 2 位置为 $D_1\ X\ D_2\ D_3$；第 3 位置为 $D_1\ D_2\ X\ D_3$。

同一个非数字字符处在不同的位置有不同的编码，非数字字符在不同位置时的赋码见表 5。

表 5　非数字字符赋码表

字符	在分组中的位置	编码（十进制数）
“空格”(GB/T 1988 值 32)	1	1000
	2	1001
	3	1002
“+”(GB/T 1988 值 43)	1	1003
	2	1004
	3	1005
“−”(GB/T 1988 值 45)	1	1006
	2	1007
	3	1008
“.”(GB/T 1988 值 46)	1	1009
	2	1010
	3	1011
“,”(GB/T 1988 值 44)	1	1012
	2	1013
	3	1014
“回车换行”(GB/T 1988 值 13、10 的组合)	1	1015
	2	1016
	3	1017

编码含有非数字字符的分组时，先输出非数字字符的 10 位二进制编码，然后再按式(3)计算并输出 3 个数字的 10 位二进制编码。

剩下的编码值 1018 至 1023 用于实现模式的转换，见 6.5.1。

示例：

输入数据：“1,234,567.899”

分组：		1,23		4,56		7.89		900
编码十进制值：	2	1013	123	1013	456	1010	789	900
转换为二进制：	10	1111110101	0001111011	1111110101	0111001000	1111110010	1100010101	1110000100

6.4.3　小写字母模式

6.4.3.1　编码字符

可编码字符 27 个，包括 26 个小写英文字母 a～z 以及“空格”(GB/T 1988 中值 32)。

6.4.3.2 编码规则

小写字母模式采用5位二进制进行编码，按顺序从a到z最后“空格”递增编码，字母“a”的编码为$(00000)_{BIN}$。剩下的5个编码值$(11011)_{BIN}$至$(11111)_{BIN}$用于实现模式的转换，见6.5.1。

示例：

输入数据：	b	a	r	“空格”	c	o	d	e
编码十进制值：	1	0	17	26	2	14	3	4
转换为二进制：	00001	00000	10001	11010	00010	01110	00011	00100

6.4.4 大写字母模式

6.4.4.1 编码字符

可编码字符27个，包括26个大写英文字母A～Z以及“空格”(GB/T 1988中值32)。

6.4.4.2 编码规则

大写字母模式采用5位二进制进行编码，按顺序从A到Z最后“空格”递增编码，字母“A”的编码为$(00000)_{BIN}$。剩下的5个编码值$(11011)_{BIN}$至$(11111)_{BIN}$用于实现模式的转换，见6.5.1。

示例：

输入数据：	B	A	R	“空格”	C	O	D	E
编码十进制值：	1	0	17	26	2	14	3	4
转换为二进制：	00001	00000	10001	11010	00010	01110	00011	00100

6.4.5 数字字母混合模式

6.4.5.1 编码字符

可编码字符63个，包括：

a) 数字0至9(GB/T 1988中值48～57)；

b) 大写英文字母A～Z(GB/T 1988中值65至90)；

c) 小写英文字母a～z(GB/T 1988中值97至122)；

d) “空格”(GB/T 1988中值32)。

6.4.5.2 编码规则

数字字母混合模式采用6位二进制进行编码，按顺序从数字、大写英文字母、小写英文字母最后“空格”递增编码，数字0的编码为$(000000)_{BIN}$。剩下的1个编码值$(111111)_{BIN}$用于实现模式的转换，见6.5.1。

示例：

输入数据：	0	A	b	“空格”
编码十进制值：	0	10	37	62
转换为二进制：	000000	001010	100101	111110

6.4.6 控制字符模式

6.4.6.1 编码字符

可编码字符64个，包括除以下字符外的GB/T 1988字符：

a) “空格”(GB/T 1988 中值 32)；
b) 数字字符(GB/T 1988 中值 48 至 57)；
c) 大写英文字母(GB/T 1988 中值 65 至 90)；
d) 小写英文字母(GB/T 1988 中值 97 至 122)；
e) DEL(GB/T 1988 中值 127)。

6.4.6.2 编码规则

控制字符模式采用 6 位二进制进行编码，按字符的 GB/T 1988 中值由小至大顺序编码，第一个字符编码为$(000000)_{BIN}$。该模式的数据长度固定为 1，编码后自动切换回之前的数据模式。输入数据的第一个字符不能分类为该模式。控制字符编码见表 6。

表 6 控制字符编码表

字符	编码	字符	编码	字符	编码	字符	编码
NUL	0	DLE	16	!	32	;	48
SOH	1	DC1	17	"	33	<	49
STX	2	DC2	18	#	34	=	50
ETX	3	DC3	19	¥	35	>	51
EOT	4	DC4	20	%	36	?	52
ENQ	5	NAK	21	&	37	@	53
ACK	6	SYN	22	'	38	[	54
BEL	7	ETB	23	(	39	\	55
BS	8	CAN	24	)	40	]	56
HT	9	BM	25	*	41	^	57
LF	10	SUB	26	+	42	_	58
VT	11	ESC	27	,	43	`	50
FF	12	FS	28	-	44	{	60
CR	13	GS	29	.	45	|	61
SO	14	RS	30	/	46	}	62
SI	15	US	31	:	47	~	63

6.4.7 字节模式

字节模式采用 8 位二进制数编码 0 到 255 的字节数据。

设输入数据的长度为 L 个字节，则先输出 9 位二进制无符号数 $L-1$，用于记录字节数，随后直接输出字节数据本身。

当输入数据的长度大于 512 字节时，将输入数据分割成多个数据段，每段长度不超过 512 字节，对每段数据分别编码。从第二段开始的每段数据都需要以模式转换码$(0111)_{BIN}$和用 9 位二进制无符号数编码的该段数据长度开始。

6.4.8 ECI 模式

6.4.8.1 ECI 编码

将输入的数据转换为一个位流。以缺省的 ECI 开始时，位流的开头为第一个数据类型的模式指示

符，否则，其前面要有 ECI 标头，后面为一个或多个不同模式的段。ECI 标头由 ECI 模式指示符 $(1100)_{BIN}$ 和 ECI 任务号组成。ECI 的任务号为 000000～811799（十进制）之间的 6 位数。ECI 任务号的编码见表 7。

表 7 ECI 任务号的编码

ECI 任务号	任务号编码
000000～001023	0bbbbbbbbbb
001024～032767	10bbbbbbbbbbbbbb
032768～811799	11bbbbbbbbbbbbbbbbbbb
注：b…b 是 ECI 任务号的二进制值。	

ECI 模式指示符只能在数据的开头或“模式结束”转换码（见 6.5.1）之后出现。输入的 ECI 数据需要编码系统作为一系列 8 位字节的值进行处理，可以采用汉字、数字、小写字母、大写字母、数字字母混合、控制字符、字节等一种或几种模式进行最高效的编码，而不必考虑其实际意义。例如，值为 30_{Hex} 到 39_{Hex} 的数据序列可以当作一个数字序列，用数字模式进行编码，即使实际上它并不表示数字字符。

示例：

ECI 编码表示：

ECI 任务号：400123

待编码数据的字节值：$(31)_{HEX}$，$(32)_{HEX}$，$(33)_{HEX}$，$(34)_{HEX}$，$(35)_{HEX}$，$(36)_{HEX}$，$(37)_{HEX}$，$(38)_{HEX}$，$(39)_{HEX}$。

编码位流：

a) ECI 模式指示符：1100；

b) ECI 任务号：11 01100001101011111011；

c) 数据模式指示符（数字）：0010；

d) 数据编码：00 0001111011 0111001000 1100010101；

e) 最终的位流：1100 11 01100001101011111011 0010 00 0001111011 0111001000 1100010101。

6.4.8.2 ECI 与结构链接

ECI 可以在单个 GM 码或 GM 码结构链接符号的任意位置出现。引入的任一 ECI 一直保持有效，直至数据结束或一个新的 ECI 被引入，ECI 将跨结构链接中的两个或多个 GM 码一直保持有效。

6.4.9 功能码模式

6.4.9.1 FNC1

FNC1 模式指示符应在 GM 码的开头编码。结构链接模式同时被应用时，FNC1 的模式指示符只在结构链接的第一个符号出现，并且 FNC1 的模式指示符在 FNC2 的模式指示符之前。

FNC1 模式指示符 $(1000)_{BIN}$ 用于标识按 GS1 系统规则格式化信息的符号。

FNC1 模式指示符 $(1011)_{BIN}$ 用于标识按 AIM 同意的特定行业或者特定应用规范格式化信息的符号。在第一数据字符位置的字符（a～z，A～Z，或两位数字）用于指定特定的应用。

6.4.9.2 FNC2

FNC2 功能码用于实现结构链接功能，输入的数据可用最多 16 个 GM 码链接起来。每个结构链结中的符号都是由一个 4 字段（20 位）链接控制头开始的：

a) 第一字段是 4 位的 FNC2 模式指示符 $(1001)_{BIN}$；

b) 第二字段是 8 位的文件签名；

c) 第三字段用 4 位数 n 表示链接中的 GM 码总个数为 $n+1$；

d) 第四字段用 4 位数 m 表示当前 GM 码在结构链接中的序号。

m 应小于或等于 n，否则该 GM 码是无效的。

文件签名是用某种签名算法对输入的整体数据产生的签名，同一个结构链接中的所有符号的文件签名应相同，防止不同结构链接之间的符号互相串扰。

FNC2 应是符号中的最后一个功能码，FNC2 链接控制头之后应是数据模式指示符或 ECI 模式指示符。

在传输结构链接的符号数据之前，结构链接中的所有符号应全部被解码成功并且将数据还原为正确的顺序。

6.4.9.3 FNC3

FNC3 功能码用于实现将符号编码的内容用作识读设备的初始化参数。FNC3 的模式指示符 $(1010)_{BIN}$ 应出现在数据编码位流之前。当 FNC3 和 FNC2 同时被应用时，FNC3 的模式指示符应在 FNC2 的模式指示符之前，且只在结构链接的第一个符号出现。FNC3 不能与 FNC1 同时使用。

6.5 混合模式编码

6.5.1 编码模式转换

数据编码时的模式转换是通过输出模式转换码来实现的，不是任何两个模式都可以转换的。表 8 列出了全部的模式转换码，括号中是模式转换码的二进制位数。

表 8 数据模式转换码

当前编码模式	下一编码模式							
	模式结束	汉字	数字	小写字母	大写字母	数字字母混合	控制字符（切换）	字节
汉字	8160 (13 位)	*	8161 (13 位)	8162 (13 位)	8163 (13 位)	8164 (13 位)	*	8165 (13 位)
数字	1018 (10 位)	1019 (10 位)	*	1020 (10 位)	1021 (10 位)	1022 (10 位)	*	1023 (10 位)
小写字母	27 (5 位)	28 (5 位)	29 (5 位)	*	30 (5 位)	124 (7 位)[a]	125 (7 位)[a]	126 (7 位)[a]
大写字母	27 (5 位)	28 (5 位)	29 (5 位)	30 (5 位)	*	124 (7 位)[a]	125 (7 位)[a]	126 (7 位)[a]
数字字母混合	1008 (10 位)	1009 (10 位)	1010 (10 位)	1011 (10 位)	1012 (10 位)	*	1014 (10 位)	1015 (10 位)
字节	0 (4 位)	1 (4 位)	2 (4 位)	3 (4 位)	4 (4 位)	5 (4 位)	*	7 (4 位)

注："*"号表示不允许的模式转换。

[a] 小写字母、大写字母模式到数字字母混合、控制字符、字节模式的 7 位转换码是 5 位的"11111"分别加上 2 位的"00"，"01"，"10"。

6.5.2 数据编码总流程

数据编码流程如下：

a) 以编码产生的二进制位流最短为目标，将输入数据按类型划分成段，对输入数据的分段进行优化的方法参见附录 B。

b) 对所有数据段按照下面的步骤逐段编码：

1) 存在功能码时，按表 3 进行编码；

2) 当前数据段是第一段时，输出该段数据的模式指示符，见表 3；

3) 按照当前数据模式的编码规则编码当前数据段；

4) 下一段数据编码前，首先输出当前模式到下一模式的模式转换码，见表 8；

5) 需要编码 ECI 时，首先输出模式转换码“模式结束”，见表 8，然后编码 ECI 模式指示符 $(1100)_{BIN}$ 及 ECI 任务号(见表 7)，之后是下一数据段的模式指示符，见表 3；

6) 最后一个数据段完成后，输出模式转换码“模式结束”，见表 8。

c) 需要填充位时，最后一个码字填充“0”。

d) 需要填充码字时，第一个填充码字应当取 $(000000000)_{BIN}$，后面的填充码字根据 5.5 规定的规则进行填充。

6.6 纠错编码

6.6.1 纠错等级

GM 码采用伽罗瓦有限域 $GF(2^7)$ 的 Reed-Solomon 纠错算法生成纠错码字，有限域的本原多项式为 x^7+x^3+1，码字的位长为 7 位。纠错码字应添加在数据码字流后。GM 码有 5 个用户可选纠错等级，对应的纠错码字容量见表 9。

表 9 纠错码字容量

纠错等级	1	2	3	4	5
纠错码字占总码字百分比(向下舍入)	10%	20%	30%	40%	50%

6.6.2 纠错的选择

根据 GM 码质量、识读设备精度以及应用的物理环境选择最佳纠错等级。表 10 给出了各版本 GM 码的推荐纠错等级。一般情况下，用户不应选择小于表 10 中给出的最低推荐纠错等级的 GM 码。纠错等级选择的用户导则参见 C.2。

表 10 推荐的纠错等级

版本	推荐的纠错等级	推荐纠错等级下的数据码字数	最低推荐纠错等级	最低推荐纠错等级下的数据码字数
1	5	9	4	11
2	4	30	2	40
3	4	59	1	89
4	3	114	1	146
5～13	3	总码字的 70%(向上舍入)	1	总码字的 90%(向上舍入)

纠错码字可以纠正两种类型的错误，拒读错误(错误码字的位置已知)和替代错误(错误码字的位置未知)。可纠正的替代错误数和拒读错误数与纠错码字数和错误检测码字数之间的关系由式(4)给出。

$$e+2t=d-p \qquad \cdots\cdots(4)$$

式中：

e——拒读错误数；

t——替代错误数；

d——纠错码字数；

p——错误检测码字数。

在一般情况下，$p=0$。当大部分纠错容量用于纠正拒读错误时，则检不出替代错误的概率增加；当拒读错误的总数大于纠错码字总数的一半时，$p=3$；当 GM 码的纠错码字总数小于 6 时，只允许纠正替代错误($e=0, p=1$)。

6.6.3 码字的分块与纠错码字的分配

运用 Reed-Solomon 纠错算法生成纠错码字时码字序列的长度受到所选用有限域的限制，GM 码采用的是 GF(2^7)有限域，码字序列的长度应小于 128。当数据码字的个数加上纠错码字的个数大于 127 时需要将数据码字分割成多个纠错块，然后分别对每个纠错块运用纠错算法生成各自的纠错码字。

设 GM 码的总码字容量为 C。将 C 个码字分成 B_1 个长度为 N_1 的块，以及 B_2 个长度为 N_2 的块，满足式(5)。

$$C=B_1\times N_1+B_2\times N_2 \qquad \cdots\cdots(5)$$

总分块数 B 见式(6)。

$$B=(C+126)\ \mathrm{DIV}\ 127 \qquad \cdots\cdots(6)$$

当 C 是 B 的整数倍时，码字分块参数为：$B_1=B, N_1=C\ \mathrm{DIV}\ B, B_2=0, N_2=0$。

当 C 不是 B 的整数倍时，码字分块参数为：$N_1=(C\ \mathrm{DIV}\ B)+1, N_2=N_1-1, B_1=C-B\times N_2, B_2=B-B_1$。

设选定的纠错等级为 $R(1\leqslant R\leqslant 5)$，需要生成的纠错码字总数 E 见式(7)。

$$E=(C\times R)\ \mathrm{DIV}\ 10 \qquad \cdots\cdots(7)$$

B 个纠错块中，前 B_3 块中每块分配 E_1 个纠错码字，后 B_4 块每块分配 E_2 个纠错码字，分配结果满足式(8)。

$$\begin{aligned} E&=E_1\times B_3+E_2\times B_4\\ B&=B_1+B_2=B_3+B_4 \end{aligned} \qquad \cdots\cdots(8)$$

当 E 是 B 的整数倍时，纠错码字分配参数为：$B_3=B, E_1=E\ \mathrm{DIV}\ B, B_4=0, E_2=0$。

当 E 不是 B 的整数倍时，纠错码字分配参数为：$E_1=(E\ \mathrm{DIV}\ B)+1, E_2=E_1-1, B_3=E-B\times E_2, B_4=B-B_3$。

GM 码码字分块参数表及分块计算法则的 C 语言源代码见附录 A。

6.6.4 生成纠错码字

构造数据码字多项式，多项式系数是数据码字，第一个数据码字为最高次项的系数，依次排列，最后一个数据码字是最低次项(常数项)的系数。设 k 是纠错码字的个数，则纠错码字多项式是数据码字多项式乘以 x^k 再除以纠错生成多项式得到的余式。其中余式的最高次项的系数为第一个纠错码字，最低次项的系数为最后一个纠错码字。

GM 码生成 k 个纠错码字的生成多项式 $G(x)$ 见式(9)。

$$G(x)=(x-\alpha^1)(x-\alpha^2)\cdots(x-\alpha^k) \qquad \cdots\cdots(9)$$

式中：

$G(x)$——纠错生成多项式；

α ——有限域 GF(2^7)的生成元；

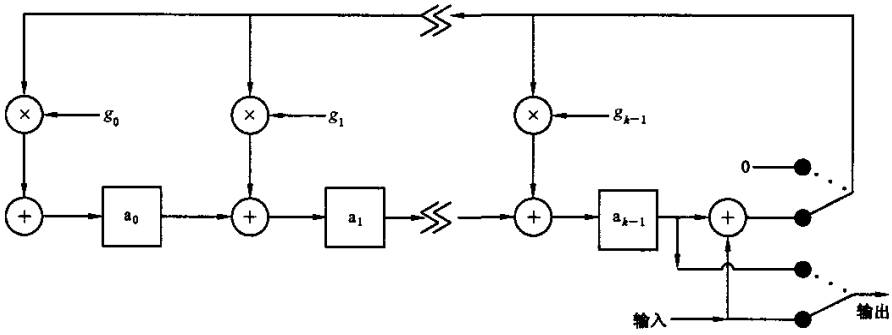
k ——要生成的纠错码字个数。

计算生成多项式系数的 C 语言源代码见附录 D。

纠错码字的生成可用图 6 的电路实现。寄存器 a_0 到 a_{k-1} 的初始值为 0，g_0 到 g_{k-1} 为生成多项式由低次到高次的系数。编码分两个阶段完成：

a) 第一阶段两个开关的位置都向下，n 个时钟脉冲后结束，输入的数据码字被直接导向输出，同时寄存器 a_0 到 a_{k-1} 的值都被更新。这里 n 是数据码字的个数。

b) 第二阶段两个开关的位置都向上，输入保持为零，k 个脉冲后寄存器的值被顺序移位输出，从而生成 k 个纠错码字。



说明：

⊗——GF(2^7)乘法

⊕——GF(2^7)加法

图 6 纠错编码电路

6.7 生成最终码字流

6.7.1 生成数据码字

编码输入数据生成二进制的位流。将二进制位流分成 7 位一组即生成数据码字流，位流分组中的第一个位对应于码字的最高位。最后一个码字不足 7 位时，低位以“0”填充，这样处理后的结果就是数据码字流。

6.7.2 确定 GM 码的各项参数

6.7.2.1 确定 GM 码版本

选择可接受的最小纠错等级 R_{min}，计算所需的最小码字数 C_{min} 见式(10)。

$$C_{min} = (10 \times D)/(10 - R_{min})\ (\text{向上舍入}) \quad \cdots\cdots (10)$$

式中：

D ——由 6.7.1 得到的数据码字数量。

查表 1 选择可以编码 C_{min} 的对应版本。

6.7.2.2 确定 GM 码的实际纠错等级

版本 V 的 GM 码码字容量 C 是 GM 码宏模块数量的两倍，见式(11)。

$$C=2\times(2\times V+1)^2 \qquad \cdots\cdots(11)$$

对于确定的版本 V，使用的纠错等级 R 见式(12)。

$$R=[(C-D)\times 10]\text{DIV } C \qquad \cdots\cdots(12)$$

当算出的 R 大于 5 时，R 即为最大值 5。

6.7.2.3 计算填充码字

纠错等级为 R，纠错码字数 E 见式(7)。需要加到数据码字流后的填充码字数 P 见式(13)。

$$P=C-E-D \qquad \cdots\cdots(13)$$

6.7.3 纠错块的交错排列

当 GM 码包括不止一个纠错块时，应对纠错块进行交错排列，规则如下：

a) 从码字流的第一个块的第一个码字开始提取，然后是第二块的第一个码字，直到最后一个块的第一个码字；

b) 继续提取第一个块的第二个码字，然后是第二块的第二个码字，……，直到最后一个块的第二个码字；

c) 继续提取直到码字流中所有码字被提取完。

6.8 码字在符号中的排列

码字流按照固定的路径填充到宏模块中，每个宏模块先填充第 1 码字，然后填充第 2 码字。以 GM 码的中心宏模块为起点，按顺时针螺旋式依次填充所有宏模块，见图 7。

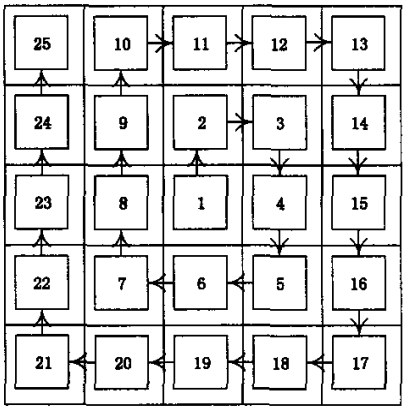


图 7 版本 2 的 GM 码的码字填充顺序示意图

图 7 所示为版本 2 的 GM 码码字填充顺序，从 1 号宏模块开始，到 25 号宏模块结束。1 号宏模块位于 GM 码的正中心，2 号宏模块在 1 号宏模块的上方。

对于版本 3 或者更高版本的 GM 码，接下来的宏模块(26)填充在 25 上面，依次不断填充，直到 GM 码所有层填充完毕。

6.9 编码示例

6.9.1 数据分析

本示例使用推荐的纠错等级编码数据“Grid Matrix”生成 GM 码。

根据附录 B 的选择方法，用数字字母混合模式编码开头 6 个字符(“Grid M”)，用小写字母模式编码后续的 5 个字符。

6.9.2 数据编码

编码顺序如下：

a) $(0101)_{BIN}$表示数字字母混合模式；

b) “Grid M”编码见表 11；

表 11 数据编码

字符	G	r	i	d	“空格”	M
十进制值	16	53	44	39	62	22
二进制值	010000	110101	101100	100111	111110	010110

c) $(1111110011)_{BIN}$(十进制 1011) 转换编码模式到小写字符；

d) “atrix”编码见表 12；

表 12 数据编码

字符	a	t	r	i	x
十进制值	0	19	17	8	23
二进制值	00000	10011	10001	01000	10111

e) $(11011)_{BIN}$(十进制 27)表示数据编码结束。

编码得到的二进制位流是：(中间空格仅用于方便阅读)

0101 010000 110101 101100 100111 111110 010110 1111110011 00000 10011 10001 01000 10111 11011

分成 7 位一组的码字后需要 4 个填充位(圆括号中所示)来填充最后一个码字。

0101010 0001101 0110110 0100111 1111100 1011011 1111001 1000001 0011100 0101000 1011111 011(0000)

该数据共 12 个码字。

6.9.3 确定 GM 码版本

查表 10，确定用版本 2 的 GM 码编码以上数据(推荐最小纠错等级为 4)。

由式(11)计算得到版本 2 的符号容量为：$C = 2(2V + 1)^2 = 2\times(2\times2 + 1)^2 = 50$。

由式(12)计算使用的纠错等级为：$R = 10(C-D)$ DIV $C = 10(50-12)$ DIV $50 = 7$。$R>5$，应采用 5 级纠错。

由式(7)得出纠错码字的数量为：$E = (C\times R)$ DIV $10 = (50\times5)$ DIV $10 = 25$。

由式(13)得出填充码字数量为：$P = C-E-D = 50-25-12 = 13$。

紧接在数据后的填充码字为：$(0000000)_{BIN}(1111110)_{BIN}(0000000)_{BIN}(1111110)_{BIN}\cdots(0000000)_{BIN}(1111110)_{BIN}(0000000)_{BIN}$。

6.9.4 生成纠错码字

十进制的数据码字加上填充码字是：

42 13 54 39 124 91 121 65 28 40 95 48 0 126 0 126 0 126 0 126 0 126 0 126 0

相应的 25 个纠错码字是：

123 47 2 20 54 112 35 23 100 89 55 17 101 4 14 33 48 62 98 52 2 79 92 70 102

所有码字以二进制表示为：

25 个数据和填充码字

0101010 0001101 0110110 0100111 1111100 1011011 1111001 1000001 0011100 0101000
1011111 0110000 0000000 1111110 0000000 1111110 0000000 1111110 0000000 1111110
0000000 1111110 0000000 1111110 0000000

加上 25 个纠错码字

1111011 0101111 0000010 0010100 0110110 1110000 0100011 0010111 1100100 1011001
0110111 0010001 1100101 0000100 0001110 0100001 0110000 0111110 1100010 0110100
0000010 1001111 1011100 1000110 1100110

6.9.5 宏模块的排列

版本 2 的 GM 码由 25 个宏模块组成(13 个深色边宏模块和 12 个浅色边宏模块)，见图 8。

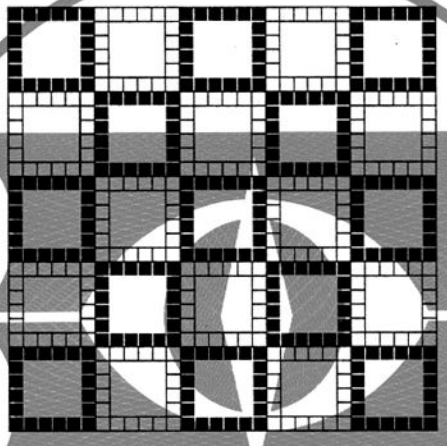


图 8 版本 2 的 GM 码宏模块的排列

将层标识号插入到每个宏模块，见图 9。

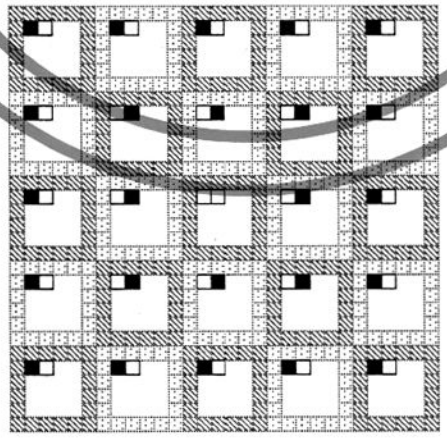


图 9 版本 2 纠错等级 5 的层标识号

6.9.6 数据码字的排列

向宏模块中填入码字的顺序是从中心向外顺时针排列，见图 7。图 10 显示的是排列的头几步：

a) 第一个码字$(0101010)_{BIN}$填在中心深色边宏模块的第 1 码字(b_6～b_0)位置上；

b) 第二个码字$(0001101)_{BIN}$填在同一个宏模块的第 2 码字(b_{13}～b_7)位置上；

c) 第三个码字$(0110110)_{BIN}$填在中间靠上的浅色边宏模块的第 1 码字位置上；

d) 第四个码字填在同一个宏模块的第 2 码字位置上，后续的码字以此顺序依次排列。

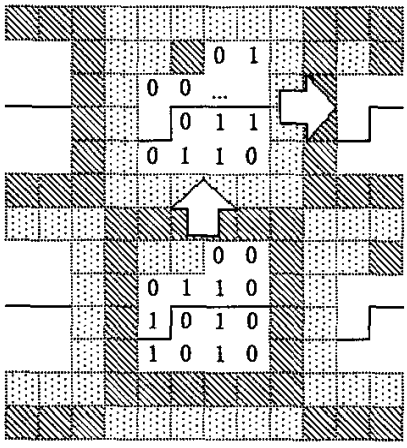


图 10 头几个码字在 GM 码中的排列

图 11 展示了在 GM 码中填入码字的过程。填入码字的过程如左图所示：12 个数据码字以螺旋排列的顺序填到头 6 个宏模块中；其后 13 个填充码字填入到接下来的 6½个宏模块中，在 GM 码的右上角宏模块的第 1 码字位置结束；纠错码字从该宏模块的第 2 码字位置开始填入，在图中仅显示了一部分。填充完毕后，生成了右图。

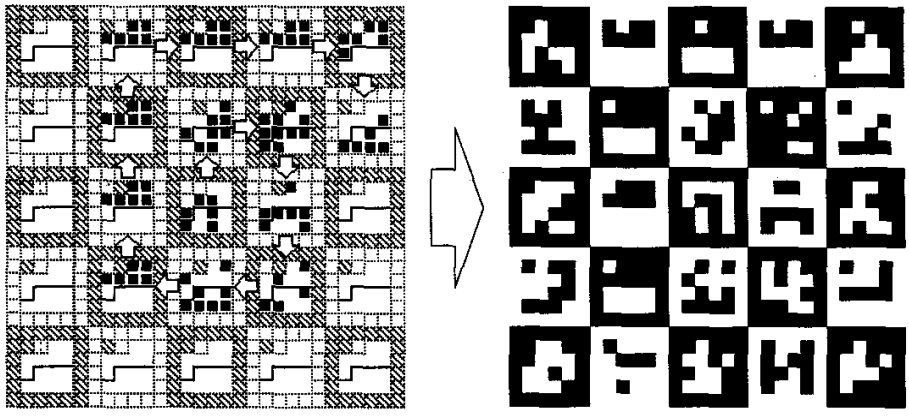


图 11 编码了数据“Grid Matrix”的 GM 码

7 符号印制

7.1 尺寸

GM 码符号尺寸的确定：

X 尺寸：单元模块宽度根据应用要求、采用的扫描技术以及符号生成技术来确定。

Y 尺寸：单元模块高度应与单元模块宽度相等。

最小空白区：在 GM 码符号周围的空白区宽度尺寸为 6*X*。

7.2 供人识读字符

GM 码可以编码数百个字符，可用描述性的文本而不是全部数据原文与 GM 码符号同时印制在一起。

字符尺寸与字体不作具体规定，可印制在 GM 码周围的任意区域，但不能影响 GM 码符号本身及空白区。

7.3 符号制作指南

可用多种不同的技术制作 GM 码符号，参见附录 C。

8 符号质量

8.1 符号质量评级

GM 码符号采用 GB/T 23704 中规定的矩阵式二维条码印制质量测试导则进行质量评级。

评级参数包括符号译码、符号反差、调制比、轴向不一致性、网格不一致性、未使用的纠错以及固有图形污损，其中固有图形污损的质量评级方法见 8.2。

一次扫描获得的符号译码、符号反差、调制比、轴向不一致性、网格不一致性、未使用的纠错以及固有图形污损各参数等级的最低值为单次扫描等级。

符号等级为从不同角度进行的 5 次扫描获得的单次扫描等级的算术平均值。两次扫描译码获得的数据不同时，符号等级为 0。符号等级按质量高低以 4.0～1.0 的数字形式表示，小数点后应保留一位。

8.2 固有图形污损质量等级

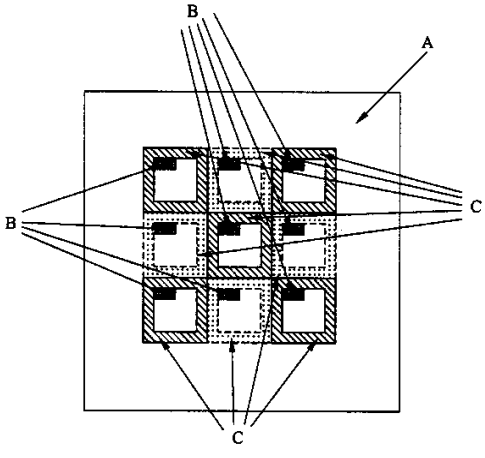
8.2.1 需评估的固有图形

GM 码需要评级的固有图形包括(见图 12)：

a) 空白区；

b) 层标识号区域；

c) 宏模块边框。



说明：

A——空白区；

B——层标识号区域；

C——宏模块边框。

图 12 版本 1 的 GM 码的固有图形

固有图形污损的质量等级为空白区、层标识号区域和宏模块边框三个质量等级的最低值。

8.2.2 空白区质量评级

空白区环绕在GM码周围，为6个单元模块宽且空白区单元模块均为浅色。空白区内单元模块数量为144 + 144 (2V +1)，其中V是GM码的版本。将译码过程中建立的采样网格延拓到空白区，在空白区内统计深色单元模块的数量，求出空白区内深色单元模块占空白区总单元模块数的百分比，评级规则见表13。

8.2.3 宏模块边框质量评级

每个宏模块均有一个边框，每个边框有20个单元模块，当其中任何一个单元模块颜色读取错误时，该边框被视为错误边框，统计错误边框数量，根据错误边框数占总边框数的百分比对边框区域进行评级，见表13。

8.2.4 层标识号区域质量评级

每个宏模块有两个层标识号单元模块，GM码中层标识号单元模块总数为$2(2V+1)^2$，其中V为版本。统计层标识号单元模块错误的数量，根据错误的层标识号单元模块数占总层标识号单元模块数量的百分比确定质量等级，见表13。

表13 固定区域损失的评级阈值

空白区	宏模块边框	层标识号	等级
错误单元模块比率	错误边框比率	错误单元模块比率	
0%	0%	0%	4
≤10%	≤5%	≤3%	3
≤20%	≤10%	≤7%	2
≤30%	≤15%	≤11%	1
>30%	>15%	>11%	0

9 译码过程

9.1 概述

GM码的译码过程如下：

a) 灰阶图像二值化

抓取的图像通常是灰阶图像。使用附录E.1中的二值化算法将灰阶图像转化为二值图像。以下的步骤均是在二值图像上实施的。

b) 获取边界图像及连通分支

称二值图像中的一个像素点为边界点是指该像素点本身为白色且其周围的4个像素至少有一个为黑色。由二值图像的所有边界点组成的图像称为其边界图像。分别提取边界图像的所有连通分支(8连通)，从边界图像中删去尺寸小的连通分支。

c) 估计GM码的水平斜率和竖直斜率

根据边界图像估计GM码的水平斜率和竖直斜率。

d) 估计宏模块的面积

根据边界图像分别估计宏模块的宽度和高度，从而得到宏模块的面积。从边界图像删去所有面积小于该估计面积值70%的连通分支，得到的图像称为框架图像。

e) 计算宏模块的中心

根据框架图像和GM码的水平斜率和竖直斜率计算所有宏模块的中心。

f) 计算宏模块的顶点

对邻近的宏模块中心插值可以计算得到宏模块的顶点。

g) 数据采样

根据宏模块的顶点计算采样网格。黑色像素表示二进制的"1"，白色像素表示二进制的"0"。

h) 推导GM码的方向、中心宏模块和纠错等级

根据层标识号可以推导得到GM码的方向、中心宏模块和纠错等级。

i) 还原GM码码字并纠错

根据码字排列规则得到GM码码字，进行纠错后得到原始信息的编码位流。

j) 编码位流解码

根据6.3、6.4和6.5定义的编码方法对编码位流进行解码。

9.2 参考译码算法

GM码的参考译码算法参见附录E。

10 数据传输

10.1 符号标识符

ISO/IEC 15424提供了一个标准的程序，根据译码器的设置和GM码的自身特性报告被读取的码制。

一旦数据结构(包括使用的ECI模式)被识别，译码器将适当的符号标识符作为一个段首标记追加到被传输的数据上。GM码使用了ECI模式或FNC1功能码时，符号标识符应被传输。

GM码的符号标识符是"]gm"，其中：

a)]是符号标识符(GB/T 1988值为93)；

b) g是GM码的编码字符；

c) m是变数值，取值为表14中的某一值。

表14 符号标识符与变数值

变数值	含义
0	未使用ECI协议
1	已使用ECI协议
2	未使用ECI协议，FNC1用于指示GS1应用
3	已使用ECI协议，FNC1用于指示GS1应用
4	未使用ECI协议，FNC1用于指示AIM应用
5	已使用ECI协议，FNC1用于指示AIM应用

10.2 扩展信道解释

在支持ECI协议的系统中，每一传输都要求传输符号标识符。当遇到ECI模式指示符时，它应作

为转义字符“\”(GB/T 1988 中值 92)被传输。根据表 7 定义的规则,将紧跟在 ECI 模式指示符后的 11、17 或 22 位转化为 6 位数字,这些数字将按其 GB/T 1988 值进行传输(48 到 57)。

应用软件识别到\nnnnnn 之后,将所有后续字符解释为来自 6 位数字的指示符定义的 ECI。该解释一直有效直至数据结束或遇到另一个 ECI 指示符。

当字符“\”需要作为被编码的数据时,应按如下方式进行传输:每当字符“\”作为数据出现,应传输两个该值的字节,因此每当单个值出现,总是一个转义字符,连续两次出现则表示真正的数据。

示例:

被编码的数据:按 ECI 123456 规则编码 A\\B\C

被传输的数据:]g1\123456A\\\\B\\C

注:除非 GM 码确实使用了 ECI 模式指示符,符号标识符]g1,]g3,]g5 不应被传输,并且反斜杠字符“\”不应被传输两次。

10.3 FNC1

功能码没有对应的字节值,不能被直接传输,应通过相关的符号标识符(]g2,]g3,]g4,]g5)指示功能码 FNC1 被使用的情形(见 10.1)。

10.4 FNC2

功能码 FNC2 用于结构链接,译码器在传输前将数据文件重新链接,不传输结构链接头。如果结构链接中任一 GM 码读取失败,译码器不传输任何数据。

10.5 FNC3

功能码 FNC3 用于指示识读器将该 GM 码中的数据作为初始化参数,这些数据不应被传输。识读器可以根据 GM 码中的数据进行初始化。

附 录 A
（规范性附录）
码字分块参数 C 语言源代码

6.6.3 定义的码字分块参数的 C 语言源代码如下：

```
/*
功能:为指定版本和纠错等级的 GM 码计算码字分块参数
输入:GM 码的版本和纠错等级
输出:分块结果
      码字个数为 B1 的块有 N1 块,码字个数为 B2 的块有 N2 块
      前 B3 块每块纠错码字 E1 个,后 B4 块每块纠错码字 E2 个
*/
#define N 127
int B1,N1,B2,N2;
int B3,E1,B4,E4;
void rs_block(int V,/* GM 码的版本,1—13 */
int R) /* 纠错等级, 1—5 */
{
     int C;/* GM 码的总码字数 */
     int E;/* 纠错码字数 */
     int B;/* 总分块数 */
     C = (2 * V + 1) * (2 * V + 1) * 2;
     B = (C + N-1) / N;
     if (0 = = C % B){
        B1 = B;
        N1 = C/B;
        B2 = 0;
        N2 = 0;
     } else {
        N1 = C / B + 1;
        N2 = N1-1;
        B1 = C-B * N2;
        B2 = B-B1;
     }
     E = (int)(C * 0.1 * R);
     if (0 = = E % B){
        B3 = B;
        E1 = E / B;
        B4 = 0;
        E2 = 0;
     } else {
        E1 = E / B + 1;
```

```
            E2 = E1 - 1;
            B3 = E - B * E2;
            B4 = B - B3;
        }
        return;
    }
```

表 A.1 列出了所有版本和纠错等级的 GM 码的分块参数。

表 A.1 GM 码分块参数

版本	纠错等级	码字总数	数据码字数	分块参数							
				N1	B1	N2	B2	E1	B3	E2	B4
1	1	N/A	—	—	—	—	—	—	—	—	—
	2	18	15	18	1	0	0	3	1	0	0
	3	18	13	18	1	0	0	5	1	0	0
	4	18	11	18	1	0	0	7	1	0	0
	5	18	9	18	1	0	0	9	1	0	0
2	1	50	45	50	1	0	0	5	1	0	0
	2	50	40	50	1	0	0	10	1	0	0
	3	50	35	50	1	0	0	15	1	0	0
	4	50	30	50	1	0	0	20	1	0	0
	5	50	25	50	1	0	0	25	1	0	0
3	1	98	89	98	1	0	0	9	1	0	0
	2	98	79	98	1	0	0	19	1	0	0
	3	98	69	98	1	0	0	29	1	0	0
	4	98	59	98	1	0	0	39	1	0	0
	5	98	49	98	1	0	0	49	1	0	0
4	1	162	146	81	2	0	0	8	2	0	0
	2	162	130	81	2	0	0	16	2	0	0
	3	162	114	81	2	0	0	24	2	0	0
	4	162	98	81	2	0	0	32	2	0	0
	5	162	81	81	2	0	0	41	1	40	1
5	1	242	218	121	2	0	0	12	2	0	0
	2	242	194	121	2	0	0	24	2	0	0
	3	242	170	121	2	0	0	36	2	0	0
	4	242	146	121	2	0	0	48	2	0	0
	5	242	121	121	2	0	0	61	1	60	1

表 A.1（续）

版本	纠错等级	码字总数	数据码字数	分块参数							
				N1	B1	N2	B2	E1	B3	E2	B4
6	1	338	305	113	2	112	1	11	3	0	0
	2	338	271	113	2	112	1	23	1	22	2
	3	338	237	113	2	112	1	34	2	33	1
	4	338	203	113	2	112	1	45	3	0	0
	5	338	169	113	2	112	1	57	1	56	2
7	1	450	405	113	2	112	2	12	1	11	3
	2	450	360	113	2	112	2	23	2	22	2
	3	450	315	113	2	112	2	34	3	33	1
	4	450	270	113	2	112	2	45	4	0	0
	5	450	225	113	2	112	2	57	1	56	3
8	1	578	521	116	3	115	2	12	2	11	3
	2	578	463	116	3	115	2	23	5	0	0
	3	578	405	116	3	115	2	35	3	34	2
	4	578	347	116	3	115	2	47	1	46	4
	5	578	289	116	3	115	2	58	4	57	1
9	1	722	650	121	2	120	4	12	6	0	0
	2	722	578	121	2	120	4	24	6	0	0
	3	722	506	121	2	120	4	36	6	0	0
	4	722	434	121	2	120	4	48	6	0	0
	5	722	361	121	2	120	4	61	1	60	5
10	1	882	794	126	7	0	0	13	4	12	3
	2	882	706	126	7	0	0	26	1	25	6
	3	882	618	126	7	0	0	38	5	37	2
	4	882	530	126	7	0	0	51	2	50	5
	5	882	441	126	7	0	0	63	7	0	0
11	1	1 058	953	118	5	117	4	12	6	11	3
	2	1 058	847	118	5	117	4	24	4	23	5
	3	1 058	741	118	5	117	4	36	2	35	7
	4	1 058	635	118	5	117	4	47	9	0	0
	5	1 058	529	118	5	117	4	59	7	58	2

表 A.1（续）

版本	纠错等级	码字总数	数据码字数	分块参数							
				N1	B1	N2	B2	E1	B3	E2	B4
12	1	1 250	1 125	125	10	0	0	13	5	12	5
	2	1 250	1 000	125	10	0	0	25	10	0	0
	3	1 250	875	125	10	0	0	38	5	37	5
	4	1 250	750	125	10	0	0	50	10	0	0
	5	1 250	625	125	10	0	0	63	5	62	5
13	1	1 458	1 313	122	6	121	6	13	1	12	11
	2	1 458	1 167	122	6	121	6	25	3	24	9
	3	1 458	1 021	122	6	121	6	37	5	36	7
	4	1 458	875	122	6	121	6	49	7	48	5
	5	1 458	729	122	6	121	6	61	9	60	3

附 录 B
（资料性附录）
位流长度的优化

B.1 GM码编码数据类型的分析

B.1.1 按数据类型分段

GM码总共有6种基本的数据类型：汉字、数字、小写字母、大写字母、控制字符、字节。

首先将需要编码的数据按这6种基本的数据类型进行分段。将输入数据看作字节流，按以下步骤，给每个字节赋予一个类型，连续的相同的数据类型被归为一个数据段。

a) 遍历字节流，若任意两个连续的字节可以组成一个GB 18030双字节1区及双字节2区的字符，则将这两个字节归为汉字类型。对一个或多个连续的“回车换行”字符，如果它们前面的两字节或后面的两字节为汉字类型，则将它们归为汉字类型，否则它们的类型是未定的。若数字对（“00”到“99”）的前两个字节及后两个字节均为汉字类型，则将该数字对归为汉字类型。

b) 从“a”到“z”的所有字节归为小写字母类型。

c) 从“A”到“Z”的所有字节归为大写字母类型。

d) 对一个或多个连续的空格，它们的类型根据这些空格前面的数据类型或后面的数据类型确定：

1) 若这些空格前面的数据类型是小写字母类型或大写字母类型，则将这些空格归为其前面的数据类型，否则转2)；

2) 若这些空格后面的数据类型是小写字母类型或大写字母类型，则将这些空格归为其后面的数据类型，否则转3)；

3) 这些空格的类型是未定的。

e) 从“0”到“9”的所有字节归为数字类型。满足6.4.2规定的可以归入数字类型的“空格”、“+”、“—”、“.”、“,”、“回车换行”也同时归为数字类型。

f) 剩下的所有未归类的字节被归为字节类型。但如果某一段字节类型满足以下所有条件，则该段数据被归为控制字符类型：

1) 该段数据不是第一个数据段；

2) 该段数据内的所有字符都是控制字符（见表6）；

3) 该段数据的长度不超过3字节；

4) 该段数据的前一数据段不是汉字类型。

B.1.2 数据类型调整

根据一个数据段及其邻近数据段编码的位流长度，对一个数据段的类型进行调整。调整的规则为：根据连续的3（或4）个数据段，对所有可能的调整方案计算编码位流的长度，取其中位流最短的方案。调整从第一个数据段开始直至最后一个数据段结束：

a) 对第一个数据段，考虑它和其后的两个数据段，对所有可能的调整方案计算编码位流的长度，根据位流最短的方案确定第一个数据段的编码类型。

b) 从第二个数据段开始，考虑4个数据段（前一数据段，当前数据段，其后的两个数据段），前一数据段的编码类型固定，对当前数据段和其后的两个数据段，计算所有可能的调整方案的编码位流长度，根据位流最短的方案确定当前数据段的编码类型。直至最后的三个数据段。

c) 最后三个数据段的编码类型由最后4段数据的位流最短的调整方案确定。

如果有几种调整方案得到的编码位流长度是一样的，则根据以下规则选取当前数据段的编码类型：

1) 若存在一种方案保持当前段数据类型不变，则保持数据段的初始类型不变；

2) 若几种调整方案当前段数据类型均变化，则根据以下优先级选取当前数据段的编码类型：数字＞小写字母＞大写字母＞数字字母混合＞控制字符＞字节＞汉字。

表B.1列出了每一种数据类型允许的编码类型。若数字类型被调整为数字字母混合类型，则数字类型中的非数字字符被调整为控制字符。

表B.1 各数据类型可能的编码类型

原数据类型	可能的编码类型
汉字	汉字 字节
数字	数字 数字字母混合 字节 汉字
小写字母	小写字母 数字字母混合 字节 汉字
大写字母	大写字母 数字字母混合 字节 汉字
控制字符	控制字符 字节 汉字
字节	字节 汉字

B.2 编码数据类型分析的例子

以下给出一个编码数据类型分析的例子。

输入数据：国外通信教材 Matlab6.5

输入数据首先被分为4段，见表B.2。

表B.2 根据数据类型分段

国外通信教材	〈空格〉M	atlab	6.5
汉字	大写字母	小写字母	数字

对开头的3个数据段(6个汉字、2个大写字母、5个小写字母),表B.3列出了所有可能的调整方案及其编码位流的长度:

表B.3 编码位流长度的计算

可能的编码方案			位流长度计算						结果
当前段	其后第一段	其后第二段							
汉字	大写字母	小写字母	4	13×6	13	5×2	5	5×5	135
汉字	大写字母	混合	4	13×6	13	5×2	7	6×5	142
汉字	大写字母	字节	4	13×6	13	5×2	7	9+8×5	161
汉字	大写字母	汉字	4	13×6	13	5×2	5	13×5	175
汉字	混合	小写字母	4	13×6	13	6×2	10	5×5	142
汉字	混合	混合	4	13×6	13	6×2	0	6×5	137
汉字	混合	字节	4	13×6	13	6×2	10	9+8×5	166
汉字	混合	汉字	4	13×6	13	6×2	10	13×5	182
汉字	字节	小写字母	4	13×6	13	9+8×2	4	5×5	149
汉字	字节	混合	4	13×6	13	9+8×2	4	6×5	154
汉字	字节	字节	4	13×6	13	9+8×2	0	8×5	160
汉字	字节	汉字	4	13×6	13	9+8×2	4	13×5	189
汉字	汉字	小写字母	4	13×6	0	13×2	13	5×5	146
汉字	汉字	混合	4	13×6	0	13×2	13	6×5	151
汉字	汉字	字节	4	13×6	0	13×2	13	9+8×5	170
汉字	汉字	汉字	4	13×6	0	13×2	0	13×5	173
字节	大写字母	小写字母	4	9+16×6	4	5×2	5	5×5	153
字节	大写字母	混合	4	9+16×6	4	5×2	7	6×5	160
字节	大写字母	字节	4	9+16×6	4	5×2	7	9+8×5	179
字节	大写字母	汉字	4	9+16×6	4	5×2	5	13×5	193
字节	混合	小写字母	4	9+16×6	4	6×2	10	5×5	160
字节	混合	混合	4	9+16×6	4	6×2	0	6×5	155
字节	混合	字节	4	9+16×6	4	6×2	10	9+8×5	184
字节	混合	汉字	4	9+16×6	4	6×2	10	13×5	200
字节	字节	小写字母	4	9+16×6	0	8×2	4	5×5	154
字节	字节	混合	4	9+16×6	0	8×2	4	6×5	159
字节	字节	字节	4	9+16×6	0	8×2	0	8×5	165
字节	字节	汉字	4	9+16×6	0	8×2	4	13×5	194
字节	汉字	小写字母	4	9+16×6	4	13×2	13	5×5	177
字节	汉字	混合	4	9+16×6	4	13×2	13	6×5	182
字节	汉字	字节	4	9+16×6	4	13×2	13	9+8×5	201
字节	汉字	汉字	4	9+16×6	4	13×2	0	13×5	204

从表 B.3 可以看出，编码方案汉字、大写字母、小写字母需要的位流最短，故第一段数据(6 个汉字)使用汉字类型进行编码。

为了得到第二个数据段的编码类型，固定其前一数据段(第一段)为汉字类型，计算当前段(第二段)和其后两个数据段(第三、四段)所有可能的调整方案的编码位流长度，计算结果列于表 B.4。

表 B.4 位流长度计算

可能的编码方案			位流长度计算							结果
当前段	其后第一段	其后第二段								
大写字母	小写字母	数字	13	5×2	5	5×5	5	2+10+10	10	90
大写字母	小写字母	混合	13	5×2	5	5×5	7	6×2+16	10	98
大写字母	小写字母	字节	13	5×2	5	5×5	7	9+8×3	4	97
大写字母	小写字母	汉字	13	5×2	5	5×5	5	13×3	13	110
大写字母	混合	数字	13	5×2	7	6×5	10	2+10+10	10	102
大写字母	混合	混合	13	5×2	7	6×5	0	6×2+16	10	98
大写字母	混合	字节	13	5×2	7	6×5	10	9+8×3	4	107
大写字母	混合	汉字	13	5×2	7	6×5	10	13×3	13	122
大写字母	字节	数字	13	5×2	7	9+8×5	4	2+10+10	10	115
大写字母	字节	混合	13	5×2	7	9+8×5	4	6×2+16	10	121
大写字母	字节	字节	13	5×2	7	9+8×5	0	9+8×3	4	116
大写字母	字节	汉字	13	5×2	7	9+8×5	4	13×3	13	135
大写字母	汉字	数字	13	5×2	5	13×5	13	2+10+10	10	138
大写字母	汉字	混合	13	5×2	5	13×5	13	6×2+16	10	144
大写字母	汉字	字节	13	5×2	5	13×5	13	9+8×3	4	143
大写字母	汉字	汉字	13	5×2	5	13×5	0	13×3	13	145
混合	小写字母	数字	13	6×2	10	5×5	5	2+10+10	10	97
混合	小写字母	混合	13	6×2	10	5×5	7	6×2+16	10	105
混合	小写字母	字节	13	6×2	10	5×5	7	9+8×3	4	104
混合	小写字母	汉字	13	6×2	10	5×5	5	13×3	13	117
混合	混合	数字	13	6×2	0	6×5	10	2+10+10	10	97
混合	混合	混合	13	6×2	0	6×5	0	6×2+16	10	93
混合	混合	字节	13	6×2	0	6×5	10	9+8×3	4	102
混合	混合	汉字	13	6×2	0	6×5	10	13×3	13	117
混合	字节	数字	13	6×2	10	9+8×5	4	2+10+10	10	120
混合	字节	混合	13	6×2	10	9+8×5	4	6×2+16	10	126
混合	字节	字节	13	6×2	10	9+8×5	0	9+8×3	4	121
混合	字节	汉字	13	6×2	10	9+8×5	4	13×3	13	140
混合	汉字	数字	13	6×2	10	13×5	13	2+10+10	10	145
混合	汉字	混合	13	6×2	10	13×5	13	6×2+16	10	151
混合	汉字	字节	13	6×2	10	13×5	13	9+8×3	4	150

表 B.4（续）

可能的编码方案			位流长度计算							结果
当前段	其后第一段	其后第二段								
混合	汉字	汉字	13	6×2	10	13×5	0	13×3	13	152
字节	小写字母	数字	13	9+8×2	4	5×5	5	2+10+10	10	104
字节	小写字母	混合	13	9+8×2	4	5×5	7	6×2+16	10	112
字节	小写字母	字节	13	9+8×2	4	5×5	7	9+8×3	4	111
字节	小写字母	汉字	13	9+8×2	4	5×5	5	13×3	13	124
字节	混合	数字	13	9+8×2	4	6×5	10	2+10+10	10	114
字节	混合	混合	13	9+8×2	4	6×5	0	6×2+16	10	110
字节	混合	字节	13	9+8×2	4	6×5	10	9+8×3	4	119
字节	混合	汉字	13	9+8×2	4	6×5	10	13×3	13	134
字节	字节	数字	13	9+8×2	0	9+8×5	4	2+10+10	10	123
字节	字节	混合	13	9+8×2	0	9+8×5	4	6×2+16	10	129
字节	字节	字节	13	9+8×2	0	9+8×5	0	9+8×3	4	124
字节	字节	汉字	13	9+8×2	0	9+8×5	4	13×3	13	143
字节	汉字	数字	13	9+8×2	4	13×5	13	2+10+10	10	152
字节	汉字	混合	13	9+8×2	4	13×5	13	6×2+16	10	158
字节	汉字	字节	13	9+8×2	4	13×5	13	9+8×3	4	157
字节	汉字	汉字	13	9+8×2	4	13×5	0	13×3	13	159
汉字	小写字母	数字	0	13×2	13	5×5	5	2+10+10	10	101
汉字	小写字母	混合	0	13×2	13	5×5	7	6×2+16	10	109
汉字	小写字母	字节	0	13×2	13	5×5	7	9+8×3	4	108
汉字	小写字母	汉字	0	13×2	13	5×5	5	13×3	13	121
汉字	混合	数字	0	13×2	13	6×5	10	2+10+10	10	111
汉字	混合	混合	0	13×2	13	6×5	0	6×2+16	10	107
汉字	混合	字节	0	13×2	13	6×5	10	9+8×3	4	116
汉字	混合	汉字	0	13×2	13	6×5	10	13×3	13	131
汉字	字节	数字	0	13×2	13	9+8×5	4	2+10+10	10	124
汉字	字节	混合	0	13×2	13	9+8×5	4	6×2+16	10	130
汉字	字节	字节	0	13×2	13	9+8×5	0	9+8×3	4	125
汉字	字节	汉字	0	13×2	13	9+8×5	4	13×3	13	144
汉字	汉字	数字	0	13×2	0	13×5	13	2+10+10	10	136
汉字	汉字	混合	0	13×2	0	13×5	13	6×2+16	10	142
汉字	汉字	字节	0	13×2	0	13×5	13	9+8×3	4	141
汉字	汉字	汉字	0	13×2	0	13×5	0	13×3	13	143

从表 B.4 可以看出，编码方案大写字母、小写字母、数字需要的位流最短，故当前段（第二段数据、〈空格〉M）使用大写字母进行编码。因为这是最后的三个数据段，故第三数据段和第四数据段分别使用小写字母、数字进行编码。

最终的编码类型为：汉字、大写字母、小写字母、数字。

B.3 GM 码编码的例子

以下是一个 GM 码编码的示例。

输入数据：AAT2556 电池充电器＋降压转换器 200mA 至 2A tel:86 010 82512738。

将输入数据分为 12 段并根据 B.1 的规则对各数据段类型进行调整后结果见表 B.5。

表 B.5 各数据段类型编码

数据段	字符	初始数据类型	编码数据类型
1	AAT	大写字母	混合
2	2556“空格”	数字	混合
3	电池充电器＋降压转换器	汉字	汉字
4	“空格”200	数字	混合
5	m	小写字母	混合
6	A	大写字母	混合
7	至	汉字	汉字
8	2	数字	混合
9	A“空格”	大写字母	混合
10	tel	小写字母	小写字母
11	:	控制字符	控制字符
12	86“空格”010“空格”82512738	数字	数字

第 1 段数据的编码模式为数字字母混合，首先输出数字字母混合模式指示符“0101”，其后是第 1 段数据编码结果，见表 B.6。

表 B.6 第 1 段数据编码

第 1 段数据	编码的十进制值	编码的二进制值
A	10	001010
A	10	001010
T	29	011101

第 2 段数据的编码模式与第 1 段相同，故不需要模式转换码，编码结果见表 B.7。

表 B.7 第 2 段数据编码

第 2 段数据	编码的十进制值	编码的二进制值
2	2	000010
5	5	000101

表 B.7（续）

第 2 段数据	编码的十进制值	编码的二进制值
5	5	000101
6	6	000110
“空格”	62	111110

第 3 段数据的编码模式为汉字，首先输出数字字母混合模式到汉字模式的模式转换码“1111110001”，其后是第 3 段数据的编码结果，见表 B.8。

表 B.8 第 3 段数据编码

第 3 段数据	编码的十进制值	编码的二进制值
电	1415	0010110000111
池	1208	0010010111000
充	1220	0010011000100
电	1415	0010110000111
器	3063	0101111110111
十	203	0000011001011
降	2133	0100001010101
压	4057	0111111011001
转	4618	1001000001010
换	1947	0011110011011
器	3063	0101111110111

第 4 段数据的编码模式为数字字母混合，首先输出汉字模式到数字字母混合模式的模式转换码“1111111100100”，其后是第 4 段数据编码结果，见表 B.9。

表 B.9 第 4 段数据编码

第 4 段数据	编码的十进制值	编码的二进制值
“空格”	62	111110
2	2	000010
0	0	000000
0	0	000000

第 5 段数据的编码模式与第 4 段相同，故不需要模式转换码，编码结果见表 B.10。

表 B.10 第 5 段数据编码

第 5 段数据	编码的十进制值	编码的二进制值
m	48	110000

第 6 段数据的编码模式与第 5 段相同，故不需要模式转换码，编码结果见表 B. 11。

表 B. 11　第 6 段数据编码

第 6 段数据	编码的十进制值	编码的二进制值
A	10	001010

第 7 段数据的编码模式为汉字，首先输出数字字母混合模式到汉字模式的模式转换码“1111110001”，其后是第 7 段数据编码结果，编码结果见表 B. 12。

表 B. 12　第 7 段数据编码

第 7 段数据	编码的十进制值	编码的二进制值
至	4545	1000111000001

第 8 段数据的编码模式为数字字母混合，首先输出汉字模式到数字字母混合模式的模式转换码“1111111100100”，其后是第 8 段数据编码结果，编码结果见表 B. 13。

表 B. 13　第 8 段数据编码

第 8 段数据	编码的十进制值	编码的二进制值
2	2	000010

第 9 段数据的编码模式与第 8 段相同，故不需要模式转换码，编码结果见表 B. 14。

表 B. 14　第 9 段数据编码

第 9 段数据	编码的十进制值	编码的二进制值
A	10	001010
“空格”	62	111110

第 10 段数据的编码模式为小写字母，首先输出数字字母混合模式到小写字母模式的模式转换码“1111110011”，其后是第 10 段数据编码结果，编码结果见表 B. 15。

表 B. 15　第 10 段数据编码

第 10 段数据	编码的十进制值	编码的二进制值
t	19	10011
e	4	00100
l	11	01011

第 11 段数据的编码模式为控制字符，首先输出小写字母模式到控制字符模式的模式转换码(shift)“1111101”，其后是第 11 段数据编码结果，编码结果见表 B. 16。

表 B. 16　第 11 段数据编码

第 11 段数据	编码的十进制值	编码的二进制值
:	47	101111

第 12 段数据的编码模式为数字，首先输出控制字符模式到数字的模式转换码“11101”，其后是第 12 段数据编码结果，编码结果见表 B.17。

表 B.17 第 12 段数据编码

第 12 段数据	编码的十进制值	编码的二进制值
计数(填充数字的个数)	2	10
86“空格”0	1002 860	1111101010 1101011100
10“空格”8	1002 108	1111101010 00011011000
251	251	0011111011
273	273	0100010001
800	800	1100100000

编码位流的结尾是从数字模式转换到结束的转换码：1111111010。

最终的编码位流为(以下空格是为了便于阅读，并非位流的数据)：

0101 001010 001010 011101 000010 000101 000101 000110 111110 1111110001 00101100000111 0010010111000 0010011000100 0010110000111 0101111110111 0000011001011 0100001010101 0111111011001 1001000001010 0011110011011 0101111110111 1111111100100 111110 000010 000000 000000 110000 001010 1111110001 1000111000001 1111111100100 000010 001010 111110 1111110011 10011 00100 01011 1111101 101111 11101 10 1111101010 1101011100 1111101010 00011001100 0011111011 0100010001 1100100000 1111111010

将位流每 7 位一组分割成码字(总共 62 个码字)，这些码字的十进制值如下：

41 34 78 66 10 20 55 111 98 44 28 75 65 24 66

97 107 123 65 75 33 42 126 102 32 81 115 53 125 127

114 62 4 0 6 2 95 70 28 15 124 64 69 62 126

57 72 95 109 126 111 85 87 31 40 54 15 90 17 100

15 116

指定使用的纠错等级为 3 级，故要求总码字数至少为 89 个(89×(1−30%)=62.3)，从而需要使用版本 3 的 GM 码(总码字数为(3×2+1)×(3×2+1)×2=98)。版本 3 纠错 3 级的 GM 码可以容纳 69 个数据码字，故需要 7 个填充码字，这 7 个填充码字为：0,126,0,126,0,126,0。

接下来需要生成纠错码字。GM 码使用 Reed-Solomon 纠错码，使用由本原多项式 x^7+x^3+1 定义的有限域 GF(2^7)。生成的纠错码字如下：

105 75 25 67 18 58 38 105 45 7 73 82 2 11 79

68 47 79 15 24 86 70 89 60 87 30 53 118 17

最终的码字流如下：

41 34 78 66 10 20 55 111 98 44 28 75 65 24 66

97 107 123 65 75 33 42 126 102 32 81 115 53 125 127

114 62 4 0 6 2 95 70 28 15 124 64 69 62 126

57 72 95 109 126 111 85 87 31 40 54 15 90 17 100

15 116 0 126 0 126 0 126 0 105 75 25 67 18 58

38 105 45 7 73 82 2 11 79 68 47 79 15 24 86

70 89 60 87 30 53 118 17

图 B.1 显示了最终的编码结果：编码数据为“AAT2556 电池充电器＋降压转换器 200mA 至 2A

tel:86 010 82512738"的3级纠错的GM码：

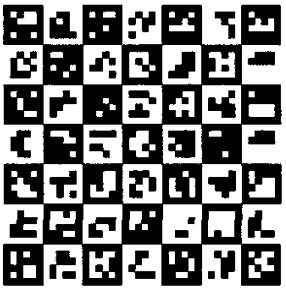


图 B.1 GM码图

附 录 C
（资料性附录）
GM 码印制的用户导则

C.1 总则

应将 GM 码的应用看作整个系统的解决方案。组成系统的各个部分（打印机、标签、识读器）需要作为一个整体来考虑，在该系统的任一环节出现问题，或各环节之间的错误匹配将损害整个系统的运行效果。

符合标准要求是保证整个系统成功的关键之一，同时其他因素也会影响系统的运行。以下导则是建议在确定或者采用 GM 码时应考虑的一些因素。

——选择适当的印制密度，使符号的允许偏差是所使用的印制技术能达到的。保证模块尺寸是打印头分辨率的整数倍（在平行和垂直于印刷方向的两个方向），也要保证印制增量的调整，这种调整是通过单个深色模块或毗连的深色模块组边缘（由深色到浅色或由浅色到深色）改变等量的整数像素来实现的，这样可以保证模块中心的间距保持不变，虽然对每个深色（或浅色）模块的位图表示的尺寸进行了调整。

——选择识读器，其分辨率应与特定印制技术所生成的符号密度和质量相适应。

——保证印制的 GM 码的光学特性与扫描器光源的波长或传感器的感光特性相适应。

——检查在最终标签或外包装上的 GM 码是否合格。遮盖、透光、弯曲或不规则表面都会影响 GM 码的识读性能。

——应考虑光滑的符号表面产生的镜面反射。扫描系统应考虑在深色与浅色特性之间的漫反射的改变量。在某些扫描角度，反射光的镜面反射部分大大地超过希望的漫反射部分，从而改变了扫描特性。如果能改变材料表面或材料表面的某部分，那么，选择粗糙的、非光滑的表面有助于减小镜面效果。否则，应保证识读符号的照明使所希望的对比度达到最佳。

C.2 纠错等级的用户选择

纠错等级的选择与下列因素相关：

——预计的 GM 码质量水平：预计的 GM 码质量等级越低，应用的纠错等级就应越高；

——潜在的 GM 码损毁可能性：潜在的 GM 码损毁可能性越高，应用的纠错等级就应越高；

——印刷 GM 码的空间限制了使用较高的纠错等级。

纠错等级 1、2 适用于具有高质量的 GM 码以及/或者要求使 GM 码的尺寸尽可能小的情况。等级 3、4 的 GM 码具有适中的尺寸和较好的可靠性。等级 5 适用于一些重要的或 GM 码印制质量差的场合。

对版本 1 的 GM 码，推荐使用 5 级纠错，对版本 2、3 的 GM 码，使用 4 级以上的纠错，对其他版本的 GM 码，使用 3 级以上的纠错。

注意：GM 码的编码过程会自动提高纠错等级以充分利用 GM 码的码字容量。

需要注意的是，当 GM 码的质量降低到一定程度后，即使提高纠错等级也无法提高 GM 码的识读可靠性。建议 GM 码印制后测试 GM 码的剩余纠错能力，根据测试结果调整纠错等级以使剩余纠错能力处于一个适中的水平，这样可以在保证可靠性的前提下使得 GM 码的尺寸达到最小。

用户应确定合适的纠错等级来满足应用需求。从 1 级到 5 级的纠错码字占 GM 码容量的百分比逐

渐增加，相应的纠错性能也逐步提高，其代价是对表示给定长度数据的 GM 码的尺寸逐步增加。表 C.1～表 C.4 列出了各版本和纠错等级的 GM 码编码各种数据类型时的编码容量。

表 C.1 GM 码容量表(字节)

GM 码版本	1 级纠错的容量	2 级纠错的容量	3 级纠错的容量	4 级纠错的容量	5 级纠错的容量
1	—	11	9	7	5
2	37	32	28	24	19
3	75	67	58	49	40
4	125	111	97	83	68
5	188	167	146	125	103
6	264	235	205	175	145
7	352	312	273	234	194
8	453	403	352	301	250
9	565	503	440	377	313
10	691	614	537	461	383
11	830	737	644	551	460
12	980	871	761	652	543
13	1 143	1 017	889	761	634

表 C.2 GM 码容量表(数字)

GM 码版本	1 级纠错的容量	2 级纠错的容量	3 级纠错的容量	4 级纠错的容量	5 级纠错的容量
1	—	24	21	18	12
2	87	78	66	57	45
3	180	159	138	117	96
4	300	267	234	201	165
5	453	402	351	300	249
6	633	564	492	420	348
7	843	750	654	561	465
8	1 089	966	843	723	600
9	1 359	1 209	1 056	906	753
10	1 662	1 476	1 293	1 107	921
11	1 995	1 773	1 551	1 326	1 104
12	2 355	2 094	1 830	1 569	1 305
13	2 751	2 445	2 139	1 830	1 524

表 C.3 GM 码容量表(GB 18030 双字节 1 区或双字节 2 区内的字符)

GM 码版本	1 级纠错的容量	2 级纠错的容量	3 级纠错的容量	4 级纠错的容量	5 级纠错的容量
1	—	6	5	4	3
2	22	20	17	14	12
3	46	41	35	30	25
4	77	68	60	51	42
5	116	103	90	77	63
6	162	144	126	108	89
7	216	192	168	144	119
8	279	248	216	185	154
9	348	309	271	232	193
10	426	378	331	284	236
11	511	454	397	340	283
12	604	537	469	402	335
13	705	627	548	469	391

表 C.4 GM 码容量表(字母)

GM 码版本	1 级纠错的容量	2 级纠错的容量	3 级纠错的容量	4 级纠错的容量	5 级纠错的容量
1	—	19	16	13	10
2	61	54	47	40	33
3	122	108	94	80	66
4	202	180	157	135	111
5	303	269	236	202	167
6	425	377	330	282	234
7	565	502	439	376	313
8	727	646	565	484	402
9	908	807	706	605	503
10	1 109	986	863	740	615
11	1 332	1 184	1 035	887	738
12	1 573	1 398	1 223	1 048	873
13	1 836	1 632	1 427	1 223	1 018

附 录 D
（规范性附录）
纠错生成多项式

以下C语言源代码用于计算GM码纠错生成多项式的系数：

```
//power table for GF(2^7)
const unsigned char pow_tab[] = {/* for x^7 + x^3 + 1 */
  1,   2,   4,   8,  16,  32,  64,   9,  18,  36,  72,  25,  50, 100,  65,  11,
 22,  44,  88,  57, 114, 109,  83,  47,  94,  53, 106,  93,  51, 102,  69,   3,
  6,  12,  24,  48,  96,  73,  27,  54, 108,  81,  43,  86,  37,  74,  29,  58,
116,  97,  75,  31,  62, 124, 113, 107,  95,  55, 110,  85,  35,  70,   5,  10,
 20,  40,  80,  41,  82,  45,  90,  61, 122, 125, 115, 111,  87,  39,  78,  21,
 42,  84,  33,  66,  13,  26,  52, 104,  89,  59, 118, 101,  67,  15,  30,  60,
120, 121, 123, 127, 119, 103,  71,   7,  14,  28,  56, 112, 105,  91,  63, 126,
117,  99,  79,  23,  46,  92,  49,  98,  77,  19,  38,  76,  17,  34,  68,   0
};
//log table for GF(2^7)
const unsigned char log_tab[] = {/* for x^7 + x^3 + 1 */
127,   0,   1,  31,   2,  62,  32, 103,   3,   7,  63,  15,  33,  84, 104,  93,
  4, 124,   8, 121,  64,  79,  16, 115,  34,  11,  85,  38, 105,  46,  94,  51,
  5,  82, 125,  60,   9,  44, 122,  77,  65,  67,  80,  42,  17,  69, 116,  23,
 35, 118,  12,  28,  86,  25,  39,  57, 106,  19,  47,  89,  95,  71,  52, 110,
  6,  14,  83,  92, 126,  30,  61, 102,  10,  37,  45,  50, 123, 120,  78, 114,
 66,  41,  68,  22,  81,  59,  43,  76,  18,  88,  70, 109, 117,  27,  24,  56,
 36,  49, 119, 113,  13,  91,  29, 101,  87, 108,  26,  55,  40,  21,  58,  75,
107,  54,  20,  74,  48, 112,  90, 100,  96,  97,  72,  98,  53,  73, 111,  99
};
//multiplication of GF(2^7)
unsigned char prod(unsigned char x,unsigned char y)
{
  if (x = = 0 || y = = 0)
   return 0;
  else
   return pow_tab[(log_tab[x] + log_tab[y]) % 127];
}
/* obtaining the coefficients of generator polynomials
* parameters
* g: pointer to array of the coefficients of generator polynomials to be calculated
* g[i] is the coefficent of term of degree i
* k: number of error correction codewords to be generated. It equals to the degree of
* the generator polynomial.
*/
```

```
void generator_polynomial(unsigned char *g,int k)
{
  int i,j;
  if (k<=0 || k>=127)
   return;
g[0] = 1;
for (i = 1;i <= k;i++)
   g[i] = 0;
for (i = 1;i <= k;i++)
{
   g[i] = g[i-1];
   for (j = i-1;j >= 1;j--)
   {
   g[j] = g[j-1]^prod(g[j],pow_tab[i]);
   }
   g[0] = prod(g[0],pow_tab[i]);
  }
}
```

表 D.1 给出了 GM 码使用的所有纠错生成多项式。

表 D.1 纠错生成多项式

纠错码字数	生成多项式
3	$x^3+\alpha^{104}x^2+\alpha^{116}x+\alpha^6$
4	$x^4+\alpha^{94}x^3+\alpha^{41}x^2+\alpha^{99}x+\alpha^{10}$
5	$x^5+\alpha^{52}x^4+\alpha^{116}x^3+\alpha^{119}x^2+\alpha^{61}x+\alpha^{15}$
6	$x^6+\alpha^{111}x^5+\alpha^6x^4+\alpha^{126}x^3+\alpha^{13}x^2+\alpha^{125}x+\alpha^{21}$
7	$x^7+\alpha^{100}x^6+\alpha^{54}x^5+\alpha^5x^4+\alpha^9x^3+\alpha^{66}x^2+\alpha^{120}x+\alpha^{28}$
8	$x^8+\alpha^{91}x^7+\alpha^{34}x^6+\alpha^{44}x^5+\alpha^6x^4+\alpha^{53}x^3+\alpha^{52}x^2+\alpha^{118}x+\alpha^{36}$
9	$x^9+\alpha^{14}x^8+\alpha^{75}x^7+\alpha^{74}x^6+\alpha^{95}x^5+\alpha^{100}x^4+\alpha^{89}x^3+\alpha^{100}x^2+\alpha^{49}x+\alpha^{45}$
10	$x^{10}+\alpha^7x^9+\alpha^{118}x^8+\alpha^{108}x^7+\alpha^{118}x^6+\alpha^{55}x^5+\alpha^2x^4+\alpha^3x^3+\alpha^{24}x^2+\alpha^{51}x+\alpha^{55}$
11	$x^{11}+\alpha^4x^{10}+\alpha^{108}x^9+\alpha^{21}x^8+\alpha^{22}x^7+\alpha^{75}x^6+\alpha^{81}x^5+\alpha^{40}x^4+\alpha^{51}x^3+\alpha^{23}x^2+\alpha^{58}x+\alpha^{66}$
12	$x^{12}+\alpha^{125}x^{11}+\alpha^{99}x^{10}+\alpha^5x^9+\alpha^{56}x^8+\alpha^{100}x^7+\alpha^{95}x^6+\alpha^{113}x^5+\alpha^{82}x^4+\alpha^{44}x^3+\alpha^{24}x^2+\alpha^{63}x+\alpha^{78}$
13	$x^{13}+\alpha^{61}x^{12}+\alpha^{29}x^{11}+\alpha^{59}x^{10}+\alpha^{103}x^9+\alpha^{70}x^8+\alpha^{56}x^7+\alpha^{63}x^6+\alpha^{91}x^5+\alpha^{11}x^4+\alpha^{108}x^3+\alpha^{92}x^2+\alpha^{11}x+\alpha^{91}$
14	$x^{14}+\alpha^{103}x^{13}+\alpha^7x^{12}+\alpha^{31}x^{11}+\alpha^{72}x^{10}+\alpha^{32}x^9+\alpha^{68}x^8+\alpha^{66}x^7+\alpha^{83}x^6+\alpha^{62}x^5+\alpha^{117}x^4+\alpha^{91}x^3+\alpha^{82}x^2+\alpha^{66}x+\alpha^{105}$
15	$x^{15}+\alpha^{33}x^{14}+\alpha^{106}x^{13}+\alpha^{66}x^{12}+\alpha^{101}x^{11}+\alpha^{58}x^{10}+\alpha^{87}x^9+\alpha^8x^8+\alpha^{16}x^7+\alpha^{111}x^6+\alpha^{98}x^5+\alpha^{30}x^4+\alpha^{11}x^3+\alpha^{67}x^2+\alpha^{10}x+\alpha^{120}$
16	$x^{16}+\alpha^{85}x^{15}+\alpha^{88}x^{14}+\alpha^{90}x^{13}+\alpha^{61}x^{12}+\alpha^{12}x^{11}+\alpha^{38}x^{10}+\alpha^{79}x^9+\alpha^{10}x^8+\alpha^{96}x^7+\alpha^{72}x^6+\alpha^{63}x^5+\alpha^2x^4+\alpha^{48}x^3+\alpha^{53}x^2+\alpha^{77}x+\alpha^9$

表 D.1（续）

纠错码字数	生成多项式
17	$x^{17}+\alpha^{39}x^{16}+\alpha^{94}x^{15}+\alpha^{26}x^{14}+\alpha^{39}x^{13}+\alpha^{53}x^{12}+\alpha^{73}x^{11}+\alpha^{111}x^{10}+\alpha^{35}x^{9}+\alpha^{44}x^{8}+\alpha^{11}x^{7}+\alpha^{116}x^{6}+\alpha^{116}x^{5}+\alpha^{120}x^{4}+\alpha^{125}x^{3}+\alpha^{84}x^{2}+\alpha^{47}x+\alpha^{26}$
18	$x^{18}+\alpha^{58}x^{17}+\alpha^{67}x^{16}+\alpha^{51}x^{15}+\alpha^{121}x^{14}+\alpha^{50}x^{13}+\alpha^{6}x^{12}+\alpha^{38}x^{11}+\alpha^{86}x^{10}+\alpha^{88}x^{9}+\alpha^{105}x^{8}+\alpha^{76}x^{7}+\alpha^{63}x^{6}+\alpha^{126}x^{5}+\alpha^{89}x^{4}+\alpha^{38}x^{3}+\alpha^{73}x^{2}+\alpha^{83}x+\alpha^{44}$
19	$x^{19}+\alpha^{76}x^{18}+\alpha^{104}x^{17}+\alpha^{42}x^{16}+\alpha^{37}x^{15}+\alpha^{23}x^{14}+\alpha^{21}x^{13}+\alpha^{116}x^{12}+\alpha^{31}x^{11}+\alpha^{30}x^{10}+\alpha^{40}x^{9}+\alpha^{61}x^{8}+\alpha^{39}x^{7}+\alpha^{91}x^{6}+\alpha^{113}x^{5}+\alpha^{20}x^{4}+\alpha^{45}x^{3}+x^{2}+\alpha^{119}x+\alpha^{63}$
20	$x^{20}+\alpha^{44}x^{19}+\alpha^{90}x^{18}+\alpha^{47}x^{17}+\alpha^{123}x^{16}+\alpha^{34}x^{15}+\alpha^{89}x^{14}+\alpha^{99}x^{13}+\alpha^{77}x^{12}+\alpha^{70}x^{11}+\alpha^{77}x^{10}+\alpha^{91}x^{9}+\alpha^{119}x^{8}+\alpha^{35}x^{7}+\alpha^{46}x^{6}+\alpha^{12}x^{5}+\alpha^{122}x^{4}+\alpha^{67}x^{3}+\alpha^{4}x^{2}+\alpha^{106}x+\alpha^{83}$
21	$x^{21}+\alpha^{10}x^{20}+\alpha^{24}x^{19}+\alpha^{126}x^{18}+\alpha^{94}x^{17}+\alpha^{85}x^{16}+\alpha^{66}x^{15}+\alpha^{6}x^{14}+\alpha^{26}x^{13}+\alpha^{82}x^{12}+\alpha^{83}x^{11}+\alpha^{94}x^{10}+\alpha^{115}x^{9}+\alpha^{81}x^{8}+\alpha^{83}x^{7}+\alpha^{38}x^{6}+\alpha^{80}x^{5}+\alpha^{110}x^{4}+\alpha^{37}x^{3}+\alpha^{84}x^{2}+\alpha^{92}x+\alpha^{104}$
22	$x^{22}+\alpha^{38}x^{21}+\alpha^{18}x^{20}+\alpha^{88}x^{19}+\alpha^{74}x^{18}+\alpha^{85}x^{17}+\alpha^{19}x^{16}+\alpha^{11}x^{15}+\alpha^{88}x^{14}+\alpha^{59}x^{13}+\alpha^{123}x^{12}+\alpha x^{11}+\alpha^{19}x^{10}+\alpha^{105}x^{9}+\alpha^{30}x^{8}+\alpha^{103}x^{7}+\alpha^{7}x^{6}+\alpha^{96}x^{5}+\alpha^{108}x^{4}+\alpha^{18}x^{3}+\alpha^{98}x^{2}+\alpha^{14}x+\alpha^{125}$
23	$x^{23}+\alpha^{86}x^{22}+\alpha^{94}x^{21}+\alpha^{3}x^{20}+\alpha^{84}x^{19}+\alpha^{113}x^{18}+\alpha^{66}x^{17}+\alpha^{12}x^{16}+\alpha^{14}x^{15}+\alpha^{42}x^{14}+\alpha^{21}x^{13}+\alpha^{89}x^{12}+\alpha^{101}x^{11}+\alpha^{57}x^{10}+\alpha^{102}x^{9}+\alpha^{98}x^{8}+\alpha^{120}x^{7}+\alpha^{71}x^{6}+\alpha^{15}x^{5}+\alpha^{10}x^{4}+\alpha^{80}x^{3}+\alpha^{68}x^{2}+\alpha^{84}x+\alpha^{22}$
24	$x^{24}+\alpha^{26}x^{23}+\alpha^{82}x^{22}+\alpha^{19}x^{21}+\alpha^{66}x^{20}+\alpha^{83}x^{19}+\alpha^{34}x^{18}+\alpha^{126}x^{17}+\alpha^{82}x^{16}+\alpha^{35}x^{15}+\alpha^{71}x^{14}+\alpha^{54}x^{13}+\alpha^{2}x^{12}+\alpha^{79}x^{11}+\alpha^{121}x^{10}+\alpha^{110}x^{9}+\alpha^{55}x^{8}+\alpha^{124}x^{7}+\alpha^{57}x^{6}+\alpha^{111}x^{5}+\alpha^{12}x^{4}+\alpha^{117}x^{3}+\alpha^{78}x^{2}+\alpha^{47}x+\alpha^{46}$
25	$x^{25}+\alpha^{56}x^{24}+\alpha^{52}x^{23}+\alpha^{37}x^{22}+\alpha^{112}x^{21}+\alpha^{75}x^{20}+\alpha^{14}x^{19}+\alpha^{124}x^{18}+\alpha^{99}x^{17}+\alpha^{6}x^{16}+\alpha^{94}x^{15}+\alpha^{7}x^{14}+\alpha^{124}x^{13}+\alpha^{10}x^{12}+\alpha^{46}x^{11}+\alpha^{32}x^{10}+\alpha^{97}x^{9}+\alpha^{89}x^{8}+\alpha^{13}x^{7}+\alpha^{56}x^{6}+\alpha^{16}x^{5}+\alpha^{79}x^{4}+\alpha^{30}x^{3}+\alpha^{71}x^{2}+\alpha^{101}x+\alpha^{71}$
26	$x^{26}+\alpha^{25}x^{25}+\alpha^{51}x^{24}+\alpha^{103}x^{23}+\alpha^{99}x^{22}+\alpha^{90}x^{21}+\alpha^{122}x^{20}+\alpha^{73}x^{19}+\alpha^{66}x^{18}+\alpha^{119}x^{17}+\alpha^{34}x^{16}+\alpha^{126}x^{15}+\alpha^{46}x^{14}+\alpha^{101}x^{13}+\alpha^{73}x^{12}+\alpha^{53}x^{11}+\alpha^{115}x^{10}+\alpha^{100}x^{9}+\alpha^{74}x^{8}+\alpha^{108}x^{7}+\alpha^{57}x^{6}+\alpha^{52}x^{5}+\alpha^{88}x^{4}+\alpha^{119}x^{3}+\alpha^{94}x^{2}+\alpha^{95}x+\alpha^{97}$
27	$x^{27}+\alpha^{87}x^{26}+\alpha^{82}x^{25}+\alpha^{37}x^{24}+\alpha^{100}x^{23}+\alpha^{12}x^{22}+\alpha^{72}x^{21}+\alpha^{116}x^{20}+\alpha^{77}x^{19}+\alpha^{21}x^{18}+\alpha^{82}x^{17}+\alpha x^{16}+\alpha^{100}x^{15}+\alpha^{85}x^{14}+\alpha^{99}x^{13}+\alpha^{15}x^{12}+\alpha^{71}x^{11}+\alpha^{53}x^{10}+\alpha^{20}x^{9}+\alpha^{104}x^{8}+\alpha^{44}x^{7}+\alpha^{28}x^{6}+\alpha^{123}x^{5}+\alpha^{112}x^{4}+\alpha^{77}x^{3}+\alpha^{23}x^{2}+\alpha^{58}x+\alpha^{124}$
28	$x^{28}+\alpha^{109}x^{27}+\alpha^{39}x^{26}+\alpha^{90}x^{25}+\alpha^{56}x^{24}+\alpha^{35}x^{23}+\alpha^{16}x^{22}+\alpha^{88}x^{21}+\alpha^{15}x^{20}+\alpha^{54}x^{19}+\alpha^{6}x^{18}+\alpha^{71}x^{17}+\alpha^{124}x^{16}+\alpha^{34}x^{15}+\alpha^{105}x^{14}+\alpha^{63}x^{13}+\alpha^{55}x^{12}+\alpha^{31}x^{11}+\alpha^{122}x^{10}+\alpha^{72}x^{9}+\alpha^{62}x^{8}+\alpha^{37}x^{7}+\alpha^{121}x^{6}+\alpha^{42}x^{5}+\alpha^{92}x^{4}+\alpha^{28}x^{3}+\alpha^{6}x^{2}+\alpha^{105}x+\alpha^{25}$
29	$x^{29}+\alpha^{71}x^{28}+\alpha^{23}x^{27}+\alpha^{9}x^{26}+\alpha^{71}x^{25}+\alpha^{80}x^{24}+\alpha x^{23}+\alpha^{121}x^{22}+\alpha^{76}x^{21}+\alpha^{81}x^{20}+\alpha x^{19}+\alpha^{84}x^{18}+\alpha^{29}x^{17}+\alpha^{20}x^{16}+\alpha^{16}x^{15}+\alpha^{31}x^{14}+\alpha^{65}x^{13}+\alpha^{104}x^{12}+\alpha^{62}x^{11}+\alpha^{9}x^{10}+\alpha^{119}x^{9}+\alpha^{17}x^{8}+\alpha^{92}x^{7}+\alpha^{2}x^{6}+\alpha^{111}x^{5}+\alpha^{5}x^{4}+\alpha^{100}x^{3}+\alpha^{17}x^{2}+\alpha^{95}x+\alpha^{54}$
30	$x^{30}+\alpha^{98}x^{29}+\alpha^{10}x^{28}+\alpha^{18}x^{27}+\alpha^{15}x^{26}+\alpha^{120}x^{25}+\alpha^{71}x^{24}+\alpha^{4}x^{23}+\alpha^{7}x^{22}+\alpha^{40}x^{21}+\alpha^{53}x^{20}+\alpha^{104}x^{19}+\alpha^{67}x^{18}+\alpha^{77}x^{17}+\alpha^{27}x^{16}+\alpha^{94}x^{15}+\alpha^{58}x^{14}+\alpha^{12}x^{13}+\alpha^{33}x^{12}+\alpha^{101}x^{11}+\alpha^{81}x^{10}+\alpha^{99}x^{9}+\alpha^{97}x^{8}+\alpha^{125}x^{7}+\alpha^{96}x^{6}+\alpha^{49}x^{5}+\alpha^{102}x^{4}+\alpha^{9}x^{3}+\alpha^{32}x^{2}+\alpha^{22}x+\alpha^{84}$
31	$x^{31}+\alpha^{98}x^{30}+\alpha^{37}x^{29}+\alpha^{7}x^{28}+\alpha^{25}x^{27}+\alpha^{66}x^{26}+\alpha^{113}x^{25}+\alpha^{76}x^{24}+\alpha^{19}x^{23}+\alpha^{100}x^{22}+\alpha^{14}x^{21}+\alpha^{31}x^{20}+\alpha^{89}x^{19}+\alpha^{117}x^{18}+\alpha^{86}x^{17}+\alpha^{107}x^{16}+\alpha^{123}x^{15}+\alpha^{7}x^{14}+\alpha^{70}x^{13}+\alpha^{74}x^{12}+\alpha^{48}x^{11}+\alpha^{63}x^{10}+\alpha^{54}x^{9}+\alpha^{5}x^{8}+\alpha^{94}x^{7}+\alpha^{36}x^{6}+\alpha^{21}x^{5}+\alpha^{13}x^{4}+\alpha^{26}x^{3}+\alpha^{88}x^{2}+\alpha^{54}x+\alpha^{115}$

表 D.1（续）

纠错码字数	生成多项式
32	$x^{32}+\alpha^{73}x^{31}+\alpha^{14}x^{30}+\alpha^{9}x^{29}+\alpha^{117}x^{28}+\alpha^{52}x^{27}+\alpha^{34}x^{26}+\alpha^{93}x^{25}+\alpha^{66}x^{24}+\alpha^{87}x^{23}+\alpha^{49}x^{22}+\alpha^{94}x^{21}+\alpha^{118}x^{20}+\alpha^{114}x^{19}+\alpha^{101}x^{18}+\alpha^{14}x^{17}+\alpha^{111}x^{16}+\alpha^{47}x^{15}+\alpha^{40}x^{14}+\alpha^{86}x^{13}+\alpha^{123}x^{12}+\alpha^{5}x^{11}+\alpha^{120}x^{10}+\alpha^{64}x^{9}+\alpha^{76}x^{8}+\alpha^{9}x^{7}+\alpha^{110}x^{6}+\alpha^{34}x^{5}+\alpha^{5}x^{4}+\alpha^{57}x^{3}+\alpha^{95}x^{2}+\alpha^{60}x+\alpha^{20}$
33	$x^{33}+\alpha^{54}x^{32}+\alpha^{97}x^{31}+\alpha^{94}x^{30}+\alpha^{100}x^{29}+\alpha^{124}x^{28}+\alpha x^{27}+\alpha^{122}x^{26}+\alpha^{64}x^{25}+\alpha^{115}x^{24}+\alpha^{17}x^{23}+\alpha^{110}x^{22}+\alpha^{35}x^{21}+\alpha^{124}x^{20}+\alpha^{79}x^{19}+\alpha^{10}x^{18}+\alpha^{126}x^{17}+\alpha^{16}x^{16}+\alpha^{61}x^{15}+\alpha^{37}x^{14}+\alpha^{116}x^{13}+\alpha^{61}x^{12}+\alpha^{43}x^{11}+\alpha^{111}x^{10}+\alpha^{116}x^{9}+\alpha^{99}x^{8}+\alpha^{64}x^{7}+\alpha^{104}x^{6}+\alpha^{7}x^{5}+\alpha^{17}x^{4}+\alpha^{45}x^{3}+\alpha^{82}x^{2}+\alpha^{73}x+\alpha^{53}$
34	$x^{34}+\alpha^{108}x^{33}+\alpha^{5}x^{32}+\alpha^{104}x^{31}+\alpha^{112}x^{30}+\alpha^{34}x^{29}+x^{28}+\alpha^{16}x^{27}+\alpha^{20}x^{26}+\alpha^{40}x^{25}+\alpha^{99}x^{24}+\alpha^{5}x^{23}+\alpha^{105}x^{22}+\alpha^{95}x^{21}+\alpha^{16}x^{20}+\alpha^{42}x^{19}+\alpha^{49}x^{18}+\alpha^{85}x^{17}+\alpha^{84}x^{16}+\alpha^{112}x^{15}+\alpha^{121}x^{14}+\alpha^{108}x^{13}+\alpha^{26}x^{12}+\alpha^{88}x^{11}+\alpha^{90}x^{10}+\alpha^{66}x^{9}+\alpha^{81}x^{8}+\alpha^{112}x^{7}+\alpha^{4}x^{6}+\alpha^{73}x^{5}+\alpha^{59}x^{4}+\alpha^{86}x^{3}+\alpha^{22}x^{2}+\alpha^{33}x+\alpha^{87}$
35	$x^{35}+\alpha^{88}x^{34}+\alpha^{39}x^{33}+\alpha^{119}x^{32}+\alpha^{102}x^{31}+\alpha^{26}x^{30}+\alpha^{17}x^{29}+\alpha^{122}x^{28}+\alpha^{21}x^{27}+\alpha^{103}x^{26}+\alpha^{4}x^{25}+\alpha^{67}x^{24}+\alpha^{107}x^{23}+\alpha^{18}x^{22}+\alpha^{94}x^{21}+\alpha^{86}x^{20}+\alpha^{61}x^{19}+\alpha^{115}x^{18}+\alpha^{6}x^{17}+\alpha^{115}x^{16}+\alpha^{49}x^{15}+\alpha^{93}x^{14}+\alpha^{53}x^{13}+\alpha^{51}x^{12}+\alpha^{47}x^{11}+\alpha^{20}x^{10}+\alpha^{28}x^{9}+\alpha^{109}x^{8}+\alpha^{119}x^{7}+\alpha^{50}x^{6}+\alpha^{95}x^{5}+\alpha^{80}x^{4}+\alpha^{6}x^{3}+\alpha^{89}x^{2}+\alpha^{47}x+\alpha^{122}$
36	$x^{36}+\alpha^{19}x^{35}+\alpha^{77}x^{34}+\alpha^{84}x^{33}+\alpha^{48}x^{32}+\alpha^{74}x^{31}+\alpha^{67}x^{30}+\alpha^{70}x^{29}+\alpha^{58}x^{28}+\alpha^{35}x^{27}+\alpha^{125}x^{26}+\alpha^{30}x^{25}+\alpha^{100}x^{24}+\alpha^{78}x^{23}+\alpha^{75}x^{22}+\alpha^{95}x^{21}+\alpha^{36}x^{20}+\alpha^{58}x^{19}+\alpha^{94}x^{18}+\alpha^{95}x^{17}+\alpha^{110}x^{16}+\alpha^{79}x^{15}+\alpha^{96}x^{14}+\alpha^{9}x^{13}+\alpha^{68}x^{12}+\alpha^{35}x^{11}+\alpha^{40}x^{10}+\alpha^{114}x^{9}+\alpha^{47}x^{8}+\alpha^{96}x^{7}+\alpha^{3}x^{6}+\alpha^{47}x^{5}+\alpha^{58}x^{4}+\alpha^{4}x^{3}+\alpha^{34}x^{2}+\alpha^{13}x+\alpha^{31}$
37	$x^{37}+\alpha^{107}x^{36}+\alpha^{96}x^{35}+\alpha^{83}x^{34}+\alpha^{101}x^{33}+\alpha^{108}x^{32}+\alpha^{76}x^{31}+\alpha^{81}x^{30}+\alpha^{94}x^{29}+\alpha^{33}x^{28}+\alpha^{18}x^{27}+\alpha^{112}x^{26}+\alpha^{24}x^{25}+\alpha^{32}x^{24}+\alpha^{96}x^{23}+\alpha^{37}x^{22}+\alpha^{8}x^{21}+\alpha^{121}x^{20}+\alpha^{125}x^{19}+\alpha^{17}x^{18}+\alpha^{51}x^{17}+\alpha^{101}x^{16}+\alpha^{43}x^{15}+\alpha^{13}x^{14}+\alpha^{114}x^{13}+\alpha^{17}x^{12}+\alpha^{16}x^{11}+\alpha^{87}x^{10}+\alpha^{13}x^{9}+\alpha^{112}x^{8}+\alpha^{10}x^{7}+\alpha^{43}x^{6}+\alpha^{113}x^{5}+\alpha^{17}x^{4}+\alpha^{37}x^{3}+\alpha^{88}x^{2}+\alpha^{10}x+\alpha^{68}$
38	$x^{38}+\alpha^{55}x^{37}+\alpha^{5}x^{36}+\alpha^{50}x^{35}+\alpha^{48}x^{34}+\alpha^{109}x^{33}+\alpha^{58}x^{32}+\alpha^{38}x^{31}+\alpha^{53}x^{30}+\alpha^{17}x^{29}+\alpha^{91}x^{28}+\alpha^{80}x^{27}+\alpha^{54}x^{26}+\alpha^{31}x^{25}+\alpha^{125}x^{24}+\alpha^{6}x^{23}+\alpha^{23}x^{22}+\alpha^{39}x^{21}+\alpha^{9}x^{20}+\alpha^{123}x^{19}+\alpha^{48}x^{18}+\alpha^{117}x^{17}+\alpha^{13}x^{16}+\alpha^{35}x^{15}+\alpha^{66}x^{14}+\alpha^{11}x^{13}+\alpha^{73}x^{12}+\alpha^{11}x^{11}+\alpha^{61}x^{10}+\alpha^{26}x^{9}+\alpha^{101}x^{8}+\alpha^{125}x^{7}+\alpha^{57}x^{6}+\alpha^{20}x^{5}+\alpha^{125}x^{4}+\alpha^{39}x^{3}+\alpha^{33}x^{2}+\alpha^{122}x+\alpha^{106}$
39	$x^{39}+\alpha^{27}x^{38}+\alpha^{52}x^{37}+\alpha^{58}x^{36}+\alpha^{114}x^{35}+\alpha^{28}x^{34}+\alpha^{31}x^{33}+\alpha^{119}x^{32}+\alpha^{109}x^{31}+\alpha^{75}x^{30}+\alpha^{47}x^{29}+\alpha^{125}x^{28}+\alpha^{121}x^{27}+\alpha^{33}x^{26}+\alpha^{96}x^{25}+\alpha^{7}x^{24}+\alpha^{91}x^{23}+\alpha^{28}x^{22}+\alpha^{26}x^{21}+\alpha^{106}x^{20}+\alpha^{126}x^{19}+\alpha^{86}x^{18}+\alpha x^{17}+\alpha^{104}x^{16}+\alpha^{80}x^{15}+\alpha^{62}x^{14}+\alpha^{39}x^{13}+\alpha^{40}x^{12}+\alpha^{84}x^{11}+\alpha^{46}x^{10}+\alpha^{114}x^{9}+\alpha^{61}x^{8}+\alpha^{111}x^{7}+\alpha^{63}x^{6}+\alpha^{100}x^{5}+\alpha^{99}x^{4}+\alpha^{83}x^{3}+\alpha^{117}x^{2}+\alpha^{5}x+\alpha^{18}$
40	$x^{40}+\alpha^{118}x^{39}+\alpha^{115}x^{38}+\alpha^{69}x^{37}+\alpha^{86}x^{36}+\alpha^{58}x^{35}+\alpha^{41}x^{34}+\alpha^{56}x^{33}+\alpha^{27}x^{32}+\alpha^{95}x^{31}+\alpha^{49}x^{30}+\alpha^{45}x^{29}+\alpha^{3}x^{28}+\alpha^{64}x^{27}+\alpha^{82}x^{26}+\alpha^{69}x^{25}+\alpha^{56}x^{24}+\alpha^{50}x^{23}+\alpha^{106}x^{22}+\alpha^{87}x^{21}+\alpha^{73}x^{20}+\alpha x^{19}+\alpha^{61}x^{18}+\alpha^{56}x^{17}+\alpha^{93}x^{16}+\alpha^{20}x^{15}+\alpha^{54}x^{14}+\alpha^{97}x^{13}+\alpha^{77}x^{12}+\alpha^{33}x^{11}+\alpha^{98}x^{10}+\alpha^{38}x^{9}+\alpha^{11}x^{8}+\alpha^{81}x^{7}+\alpha^{107}x^{6}+\alpha^{38}x^{5}+\alpha^{107}x^{4}+\alpha^{4}x^{3}+\alpha^{91}x^{2}+\alpha^{8}x+\alpha^{58}$
41	$x^{41}+\alpha^{36}x^{40}+\alpha^{124}x^{39}+\alpha^{50}x^{38}+\alpha^{15}x^{37}+\alpha^{75}x^{36}+\alpha^{116}x^{35}+\alpha^{111}x^{34}+\alpha^{9}x^{33}+\alpha^{58}x^{32}+\alpha^{7}x^{31}+\alpha^{112}x^{30}+\alpha^{95}x^{29}+\alpha^{118}x^{28}+\alpha^{11}x^{27}+\alpha^{80}x^{26}+\alpha^{36}x^{25}+\alpha^{70}x^{24}+\alpha^{56}x^{23}+\alpha^{85}x^{22}+\alpha^{99}x^{21}+\alpha^{120}x^{20}+\alpha^{21}x^{19}+\alpha^{34}x^{18}+\alpha^{90}x^{17}+\alpha^{98}x^{16}+\alpha^{57}x^{15}+\alpha^{30}x^{14}+\alpha^{52}x^{13}+\alpha^{71}x^{12}+\alpha^{3}x^{11}+\alpha^{67}x^{10}+\alpha^{33}x^{9}+\alpha^{26}x^{8}+\alpha^{43}x^{7}+\alpha^{90}x^{6}+\alpha^{91}x^{5}+\alpha^{73}x^{4}+\alpha^{23}x^{3}+\alpha^{12}x^{2}+\alpha^{93}x+\alpha^{99}$
42	$x^{42}+\alpha^{50}x^{41}+\alpha^{56}x^{40}+\alpha^{73}x^{39}+\alpha^{10}x^{38}+\alpha^{18}x^{37}+\alpha^{20}x^{36}+\alpha^{73}x^{35}+\alpha^{78}x^{34}+\alpha^{54}x^{33}+\alpha^{111}x^{32}+\alpha^{64}x^{31}+\alpha^{49}x^{30}+\alpha^{97}x^{29}+\alpha^{79}x^{28}+\alpha^{43}x^{27}+\alpha^{61}x^{26}+\alpha^{64}x^{25}+\alpha^{80}x^{24}+\alpha^{49}x^{23}+\alpha^{111}x^{22}+\alpha^{33}x^{21}+\alpha^{27}x^{20}+\alpha^{8}x^{19}+\alpha^{82}x^{18}+\alpha^{109}x^{17}+\alpha^{22}x^{16}+\alpha^{47}x^{15}+\alpha^{126}x^{14}+\alpha^{60}x^{13}+\alpha^{55}x^{12}+\alpha^{113}x^{11}+\alpha^{76}x^{10}+\alpha^{92}x^{9}+\alpha^{2}x^{8}+\alpha^{40}x^{7}+\alpha^{30}x^{6}+\alpha^{71}x^{5}+\alpha^{108}x^{4}+\alpha^{85}x^{3}+\alpha^{111}x^{2}+\alpha^{21}x+\alpha^{14}$

表 D.1（续）

纠错码字数	生成多项式
43	$x^{43}+\alpha^{46}x^{42}+\alpha^{68}x^{41}+\alpha x^{40}+\alpha^{29}x^{39}+\alpha^{9}x^{38}+\alpha^{86}x^{37}+\alpha^{100}x^{36}+\alpha^{36}x^{35}+\alpha^{119}x^{34}+\alpha^{103}x^{33}+\alpha^{37}x^{32}+\alpha^{124}x^{31}+\alpha^{47}x^{30}+\alpha^{54}x^{29}+\alpha^{107}x^{28}+\alpha^{20}x^{27}+\alpha^{85}x^{26}+\alpha^{70}x^{25}+\alpha^{69}x^{24}+\alpha^{71}x^{23}+\alpha^{41}x^{22}+\alpha^{53}x^{21}+\alpha^{10}x^{20}+\alpha^{52}x^{19}+\alpha^{97}x^{18}+\alpha^{29}x^{17}+\alpha^{8}x^{16}+\alpha^{12}x^{15}+\alpha^{3}x^{14}+\alpha^{40}x^{13}+\alpha^{34}x^{12}+\alpha^{118}x^{11}+\alpha^{101}x^{10}+\alpha^{34}x^{9}+\alpha^{122}x^{8}+\alpha^{103}x^{7}+\alpha^{6}x^{6}+\alpha^{100}x^{5}+\alpha^{37}x^{4}+\alpha^{53}x^{3}+\alpha^{35}x^{2}+\alpha^{59}x+\alpha^{57}$
44	$x^{44}+\alpha^{106}x^{43}+\alpha^{122}x^{42}+\alpha^{71}x^{41}+\alpha^{17}x^{40}+\alpha^{88}x^{39}+\alpha^{10}x^{38}+\alpha^{99}x^{37}+\alpha^{123}x^{36}+\alpha^{10}x^{35}+\alpha^{101}x^{34}+\alpha^{89}x^{33}+\alpha^{30}x^{32}+\alpha^{55}x^{31}+\alpha^{64}x^{30}+\alpha^{15}x^{29}+\alpha^{17}x^{28}+\alpha^{104}x^{27}+\alpha^{24}x^{26}+\alpha^{119}x^{25}+\alpha^{24}x^{24}+\alpha^{61}x^{23}+\alpha^{4}x^{22}+\alpha^{106}x^{21}+\alpha^{114}x^{20}+x^{19}+\alpha^{77}x^{18}+\alpha^{75}x^{17}+\alpha^{33}x^{16}+\alpha^{76}x^{15}+\alpha^{43}x^{14}+\alpha^{79}x^{13}+\alpha^{99}x^{12}+\alpha^{76}x^{11}+\alpha^{6}x^{10}+\alpha^{87}x^{9}+\alpha^{118}x^{8}+\alpha^{12}x^{7}+\alpha^{95}x^{6}+\alpha^{91}x^{5}+\alpha^{65}x^{4}+\alpha^{37}x^{3}+\alpha^{6}x^{2}+\alpha^{35}x+\alpha^{101}$
45	$x^{45}+\alpha^{20}x^{44}+\alpha^{98}x^{43}+\alpha^{41}x^{42}+\alpha x^{41}+\alpha^{117}x^{40}+\alpha^{3}x^{39}+\alpha^{64}x^{38}+\alpha^{36}x^{37}+\alpha^{11}x^{36}+\alpha^{33}x^{35}+\alpha x^{34}+\alpha^{123}x^{33}+\alpha^{2}x^{32}+\alpha^{113}x^{31}+\alpha^{66}x^{30}+\alpha^{93}x^{29}+\alpha^{15}x^{28}+\alpha^{84}x^{27}+\alpha^{114}x^{26}+\alpha^{115}x^{25}+\alpha^{55}x^{24}+\alpha^{65}x^{23}+\alpha^{88}x^{22}+\alpha^{124}x^{21}+\alpha^{103}x^{20}+\alpha^{21}x^{19}+\alpha^{37}x^{18}+\alpha^{14}x^{17}+\alpha^{11}x^{16}+\alpha^{30}x^{15}+\alpha^{123}x^{14}+\alpha^{58}x^{13}+\alpha^{98}x^{12}+\alpha^{22}x^{11}+\alpha^{100}x^{10}+\alpha^{124}x^{9}+\alpha^{68}x^{8}+\alpha^{15}x^{7}+x^{6}+\alpha^{33}x^{5}+\alpha^{90}x^{4}+\alpha^{49}x^{3}+\alpha^{23}x^{2}+\alpha^{120}x+\alpha^{19}$
46	$x^{46}+\alpha^{75}x^{45}+\alpha^{85}x^{44}+\alpha^{70}x^{43}+\alpha^{26}x^{42}+\alpha^{29}x^{41}+\alpha^{87}x^{40}+\alpha^{112}x^{39}+\alpha^{56}x^{38}+\alpha^{106}x^{37}+\alpha^{89}x^{36}+\alpha^{115}x^{35}+\alpha^{90}x^{34}+\alpha^{23}x^{33}+\alpha^{115}x^{32}+\alpha^{43}x^{31}+\alpha^{72}x^{30}+\alpha^{19}x^{29}+\alpha^{50}x^{28}+\alpha^{102}x^{27}+\alpha^{38}x^{26}+\alpha^{74}x^{25}+\alpha^{114}x^{24}+\alpha^{77}x^{23}+\alpha^{34}x^{22}+\alpha^{41}x^{21}+\alpha^{52}x^{20}+\alpha^{36}x^{19}+\alpha^{31}x^{18}+\alpha^{47}x^{17}+\alpha^{20}x^{16}+\alpha^{38}x^{15}+\alpha^{30}x^{14}+\alpha^{112}x^{13}+\alpha^{99}x^{12}+\alpha^{44}x^{11}+\alpha^{65}x^{10}+\alpha^{2}x^{9}+\alpha^{126}x^{8}+\alpha^{102}x^{7}+\alpha^{124}x^{6}+\alpha^{113}x^{5}+\alpha^{30}x^{4}+\alpha^{121}x^{3}+\alpha^{36}x^{2}+\alpha^{93}x+\alpha^{65}$
47	$x^{47}+\alpha^{59}x^{46}+\alpha^{104}x^{45}+\alpha^{23}x^{44}+\alpha^{39}x^{43}+\alpha^{38}x^{42}+\alpha^{110}x^{41}+\alpha^{53}x^{40}+\alpha^{88}x^{39}+\alpha^{110}x^{38}+\alpha^{41}x^{37}+\alpha^{28}x^{36}+\alpha^{61}x^{35}+\alpha^{101}x^{34}+\alpha^{120}x^{33}+\alpha^{29}x^{32}+\alpha^{33}x^{31}+\alpha^{109}x^{30}+\alpha^{38}x^{29}+\alpha^{52}x^{28}+\alpha^{10}x^{27}+\alpha^{108}x^{26}+\alpha^{117}x^{25}+\alpha^{110}x^{24}+\alpha^{7}x^{23}+\alpha^{62}x^{22}+\alpha^{101}x^{21}+\alpha^{51}x^{20}+\alpha^{14}x^{19}+\alpha^{48}x^{18}+\alpha^{40}x^{17}+\alpha^{12}x^{16}+\alpha^{56}x^{15}+\alpha^{68}x^{14}+\alpha^{97}x^{13}+\alpha^{105}x^{12}+\alpha^{120}x^{11}+\alpha^{54}x^{10}+\alpha^{44}x^{9}+\alpha^{70}x^{8}+\alpha^{83}x^{7}+\alpha^{61}x^{6}+\alpha^{37}x^{5}+\alpha^{86}x^{4}+\alpha^{118}x^{3}+\alpha^{120}x^{2}+\alpha^{123}x+\alpha^{112}$
48	$x^{48}+\alpha^{82}x^{47}+\alpha^{111}x^{46}+\alpha^{85}x^{45}+\alpha^{15}x^{44}+\alpha^{74}x^{43}+\alpha^{15}x^{42}+\alpha^{99}x^{41}+\alpha^{52}x^{40}+\alpha^{38}x^{39}+\alpha^{68}x^{38}+\alpha^{3}x^{37}+\alpha^{124}x^{36}+\alpha^{95}x^{35}+\alpha^{94}x^{34}+\alpha^{57}x^{33}+\alpha^{42}x^{32}+\alpha^{93}x^{31}+\alpha^{24}x^{30}+\alpha^{63}x^{29}+\alpha^{110}x^{28}+\alpha^{103}x^{27}+\alpha^{47}x^{26}+\alpha^{9}x^{25}+\alpha^{63}x^{24}+\alpha^{58}x^{23}+\alpha^{18}x^{22}+\alpha^{123}x^{21}+\alpha^{52}x^{20}+\alpha^{54}x^{19}+\alpha^{64}x^{18}+\alpha^{55}x^{17}+\alpha^{53}x^{16}+\alpha^{117}x^{15}+\alpha^{76}x^{14}+\alpha^{126}x^{13}+\alpha^{77}x^{12}+\alpha^{5}x^{11}+\alpha^{119}x^{10}+\alpha^{11}x^{9}+\alpha^{74}x^{8}+\alpha^{43}x^{7}+\alpha^{8}x^{6}+\alpha^{116}x^{5}+\alpha^{106}x^{4}+\alpha^{98}x^{3}+\alpha^{46}x^{2}+\alpha^{66}x+\alpha^{33}$
49	$x^{49}+\alpha^{6}x^{48}+\alpha^{58}x^{47}+\alpha^{16}x^{46}+\alpha x^{45}+\alpha^{101}x^{44}+\alpha^{102}x^{43}+\alpha^{55}x^{42}+\alpha^{22}x^{41}+\alpha^{53}x^{40}+\alpha^{47}x^{39}+\alpha^{81}x^{38}+\alpha^{23}x^{37}+\alpha^{82}x^{36}+\alpha^{12}x^{35}+\alpha^{82}x^{34}+\alpha^{121}x^{33}+\alpha^{26}x^{32}+\alpha^{59}x^{31}+\alpha^{100}x^{30}+\alpha^{45}x^{29}+x^{28}+\alpha^{93}x^{27}+\alpha^{117}x^{26}+\alpha^{13}x^{25}+\alpha^{38}x^{24}+\alpha^{65}x^{23}+\alpha^{91}x^{22}+\alpha^{48}x^{21}+\alpha^{16}x^{20}+\alpha^{121}x^{19}+\alpha^{3}x^{18}+\alpha^{20}x^{17}+\alpha^{38}x^{16}+\alpha^{49}x^{15}+\alpha^{29}x^{14}+\alpha^{22}x^{13}+\alpha^{13}x^{12}+\alpha^{121}x^{11}+\alpha^{10}x^{10}+\alpha^{66}x^{9}+\alpha^{85}x^{8}+\alpha^{41}x^{7}+\alpha^{11}x^{6}+\alpha^{60}x^{5}+\alpha^{10}x^{4}+\alpha^{75}x^{3}+\alpha^{40}x^{2}+\alpha^{38}x+\alpha^{82}$
50	$x^{50}+\alpha^{15}x^{49}+\alpha^{118}x^{48}+\alpha^{99}x^{47}+\alpha^{68}x^{46}+\alpha^{96}x^{45}+\alpha^{11}x^{44}+\alpha^{24}x^{43}+\alpha^{114}x^{42}+\alpha^{32}x^{41}+\alpha^{71}x^{40}+\alpha^{69}x^{39}+\alpha^{110}x^{38}+\alpha^{117}x^{37}+\alpha^{8}x^{36}+\alpha^{9}x^{35}+\alpha^{28}x^{34}+\alpha^{114}x^{33}+\alpha x^{32}+\alpha^{17}x^{31}+\alpha^{91}x^{30}+\alpha^{71}x^{29}+\alpha^{126}x^{28}+\alpha^{45}x^{27}+\alpha^{3}x^{26}+\alpha^{124}x^{25}+\alpha^{54}x^{24}+\alpha^{20}x^{23}+\alpha^{25}x^{22}+\alpha^{21}x^{21}+\alpha^{92}x^{20}+\alpha^{69}x^{19}+\alpha^{104}x^{18}+\alpha^{14}x^{17}+\alpha^{106}x^{16}+\alpha^{11}x^{15}+\alpha^{61}x^{14}+\alpha^{94}x^{13}+\alpha^{11}x^{12}+\alpha^{21}x^{11}+\alpha^{74}x^{10}+\alpha^{86}x^{9}+\alpha^{92}x^{8}+\alpha^{53}x^{7}+\alpha^{91}x^{6}+\alpha^{100}x^{5}+\alpha^{123}x^{4}+\alpha^{78}x^{3}+\alpha^{21}x^{2}+\alpha^{96}x+\alpha^{5}$
51	$x^{51}+\alpha^{54}x^{50}+\alpha^{49}x^{49}+\alpha^{81}x^{48}+\alpha^{73}x^{47}+\alpha^{85}x^{46}+\alpha^{55}x^{45}+\alpha^{109}x^{44}+\alpha^{5}x^{43}+\alpha^{46}x^{42}+\alpha^{99}x^{41}+\alpha^{15}x^{40}+\alpha^{20}x^{39}+\alpha^{126}x^{38}+\alpha^{92}x^{37}+\alpha^{54}x^{36}+\alpha^{4}x^{35}+\alpha^{70}x^{34}+\alpha^{11}x^{33}+\alpha^{8}x^{32}+\alpha^{57}x^{31}+\alpha^{39}x^{30}+\alpha^{119}x^{29}+x^{28}+\alpha^{107}x^{27}+\alpha^{38}x^{26}+\alpha^{62}x^{25}+\alpha^{58}x^{24}+\alpha^{3}x^{23}+\alpha^{47}x^{22}+\alpha^{19}x^{21}+\alpha^{89}x^{20}+\alpha^{92}x^{19}+\alpha^{20}x^{18}+\alpha^{4}x^{17}+\alpha^{117}x^{16}+\alpha^{92}x^{15}+\alpha^{55}x^{14}+\alpha^{14}x^{13}+\alpha^{87}x^{12}+\alpha^{7}x^{11}+\alpha^{16}x^{10}+\alpha^{15}x^{9}+\alpha^{26}x^{8}+\alpha^{55}x^{7}+\alpha^{53}x^{6}+\alpha^{8}x^{5}+\alpha^{48}x^{4}+\alpha^{108}x^{3}+\alpha x^{2}+\alpha^{68}x+\alpha^{56}$

表 D.1（续）

纠错码字数	生成多项式
52	$x^{52}+\alpha^{80}x^{51}+\alpha^{114}x^{50}+\alpha^{28}x^{49}+\alpha^{71}x^{48}+\alpha^{106}x^{47}+\alpha^{50}x^{46}+\alpha^{42}x^{45}+\alpha^{106}x^{44}+\alpha^{80}x^{43}+\alpha^{2}x^{42}+\alpha^{59}x^{41}+\alpha^{109}x^{40}+\alpha^{52}x^{39}+\alpha^{117}x^{38}+\alpha^{27}x^{37}+\alpha^{85}x^{36}+\alpha^{62}x^{35}+\alpha^{110}x^{34}+\alpha^{34}x^{33}+\alpha^{64}x^{32}+\alpha^{21}x^{31}+\alpha^{103}x^{30}+\alpha^{9}x^{29}+\alpha^{78}x^{28}+\alpha^{29}x^{27}+\alpha^{117}x^{26}+\alpha^{82}x^{25}+\alpha^{57}x^{24}+\alpha^{41}x^{23}+\alpha^{61}x^{22}+\alpha^{32}x^{21}+\alpha x^{20}+\alpha^{24}x^{19}+\alpha^{26}x^{18}+\alpha^{31}x^{17}+\alpha^{87}x^{16}+\alpha^{102}x^{15}+\alpha^{118}x^{14}+\alpha^{106}x^{13}+\alpha^{89}x^{12}+\alpha^{92}x^{11}+\alpha^{88}x^{10}+\alpha^{52}x^{9}+\alpha^{44}x^{8}+\alpha^{33}x^{7}+\alpha^{104}x^{6}+\alpha^{76}x^{5}+\alpha^{94}x^{4}+\alpha^{104}x^{3}+\alpha^{116}x^{2}+\alpha^{8}x+\alpha^{108}$
53	$x^{53}+\alpha^{43}x^{52}+\alpha^{93}x^{51}+\alpha^{56}x^{50}+\alpha^{108}x^{49}+\alpha^{67}x^{48}+\alpha^{44}x^{47}+\alpha^{10}x^{46}+\alpha^{2}x^{45}+\alpha^{17}x^{44}+\alpha^{126}x^{43}+\alpha^{52}x^{42}+\alpha^{116}x^{41}+\alpha^{104}x^{40}+\alpha^{6}x^{39}+\alpha^{15}x^{38}+\alpha x^{37}+\alpha^{86}x^{36}+\alpha^{55}x^{35}+\alpha^{96}x^{34}+\alpha^{53}x^{33}+\alpha^{118}x^{32}+\alpha^{48}x^{31}+\alpha^{83}x^{30}+\alpha^{50}x^{29}+\alpha^{90}x^{28}+\alpha^{73}x^{27}+\alpha^{100}x^{26}+\alpha^{44}x^{25}+\alpha^{58}x^{24}+\alpha^{18}x^{23}+\alpha^{37}x^{22}+\alpha^{34}x^{21}+\alpha^{23}x^{20}+\alpha^{120}x^{19}+\alpha^{16}x^{18}+\alpha^{91}x^{17}+\alpha^{60}x^{16}+\alpha x^{15}+\alpha^{46}x^{14}+\alpha^{71}x^{13}+\alpha^{10}x^{12}+x^{11}+\alpha x^{10}+\alpha^{73}x^{9}+\alpha^{112}x^{8}+\alpha^{47}x^{7}+\alpha^{8}x^{6}+\alpha^{85}x^{5}+\alpha^{53}x^{4}+\alpha^{55}x^{3}+\alpha^{19}x^{2}+\alpha^{23}x+\alpha^{34}$
54	$x^{54}+\alpha^{77}x^{53}+\alpha^{90}x^{52}+\alpha^{69}x^{51}+\alpha^{43}x^{50}+\alpha^{11}x^{49}+\alpha^{39}x^{48}+\alpha^{28}x^{47}+\alpha^{4}x^{46}+\alpha^{74}x^{45}+\alpha^{97}x^{44}+\alpha^{83}x^{43}+\alpha^{16}x^{42}+\alpha^{18}x^{41}+\alpha^{32}x^{40}+\alpha^{65}x^{39}+\alpha^{23}x^{38}+\alpha^{56}x^{37}+\alpha^{123}x^{36}+\alpha^{85}x^{35}+\alpha^{22}x^{34}+\alpha^{14}x^{33}+\alpha^{52}x^{32}+\alpha^{62}x^{31}+\alpha^{31}x^{30}+\alpha^{96}x^{29}+\alpha^{41}x^{28}+\alpha^{90}x^{27}+\alpha^{96}x^{26}+\alpha^{79}x^{25}+\alpha^{89}x^{24}+\alpha^{28}x^{23}+\alpha^{73}x^{22}+\alpha^{90}x^{21}+\alpha^{26}x^{20}+\alpha^{17}x^{19}+\alpha^{110}x^{18}+\alpha^{98}x^{17}+\alpha^{120}x^{16}+\alpha^{90}x^{15}+\alpha^{45}x^{14}+\alpha^{26}x^{13}+\alpha^{73}x^{12}+\alpha^{74}x^{11}+\alpha^{16}x^{10}+\alpha^{48}x^{9}+\alpha^{33}x^{8}+\alpha^{112}x^{7}+\alpha^{51}x^{6}+\alpha^{78}x^{5}+\alpha^{38}x^{4}+\alpha^{119}x^{3}+\alpha^{68}x^{2}+\alpha^{110}x+\alpha^{88}$
55	$x^{55}+\alpha^{123}x^{54}+\alpha^{43}x^{53}+\alpha^{112}x^{52}+\alpha^{102}x^{51}+\alpha^{119}x^{50}+\alpha^{29}x^{49}+\alpha^{69}x^{48}+\alpha^{68}x^{47}+\alpha^{122}x^{46}+\alpha^{73}x^{45}+\alpha^{100}x^{44}+\alpha^{93}x^{43}+\alpha^{91}x^{42}+\alpha^{52}x^{41}+\alpha^{70}x^{40}+\alpha^{119}x^{39}+\alpha^{124}x^{38}+\alpha^{12}x^{37}+\alpha^{62}x^{36}+\alpha^{57}x^{35}+\alpha^{29}x^{34}+\alpha^{121}x^{33}+\alpha^{112}x^{32}+\alpha^{56}x^{31}+\alpha^{123}x^{30}+\alpha^{93}x^{29}+\alpha^{104}x^{28}+\alpha^{5}x^{27}+\alpha^{50}x^{26}+\alpha^{9}x^{25}+\alpha^{125}x^{24}+\alpha^{110}x^{23}+\alpha^{48}x^{22}+\alpha^{12}x^{21}+\alpha^{96}x^{20}+\alpha^{30}x^{19}+\alpha^{36}x^{18}+\alpha^{77}x^{17}+\alpha x^{16}+\alpha^{8}x^{15}+\alpha^{46}x^{14}+\alpha^{14}x^{13}+\alpha^{72}x^{12}+\alpha^{8}x^{11}+\alpha^{37}x^{10}+\alpha^{15}x^{9}+\alpha^{17}x^{8}+\alpha^{74}x^{7}+\alpha^{90}x^{5}+\alpha^{109}x^{5}+\alpha^{21}x^{4}+\alpha^{87}x^{3}+\alpha^{74}x^{2}+\alpha^{83}x+\alpha^{16}$
56	$x^{56}+\alpha^{121}x^{55}+\alpha^{87}x^{54}+\alpha^{63}x^{53}+\alpha^{16}x^{52}+\alpha^{49}x^{51}+\alpha^{8}x^{50}+\alpha^{57}x^{49}+\alpha^{107}x^{48}+\alpha^{57}x^{47}+\alpha^{119}x^{46}+\alpha^{74}x^{45}+\alpha^{108}x^{44}+\alpha^{39}x^{43}+\alpha^{123}x^{42}+\alpha^{28}x^{41}+\alpha^{122}x^{40}+\alpha^{91}x^{39}+\alpha^{78}x^{38}+\alpha^{76}x^{37}+\alpha^{32}x^{36}+\alpha^{62}x^{35}+\alpha^{7}x^{34}+\alpha^{52}x^{33}+\alpha^{104}x^{32}+\alpha^{19}x^{31}+\alpha^{118}x^{30}+\alpha^{27}x^{29}+\alpha^{17}x^{28}+\alpha^{84}x^{27}+\alpha^{105}x^{26}+\alpha^{63}x^{25}+\alpha^{78}x^{24}+\alpha^{83}x^{23}+\alpha^{95}x^{22}+\alpha^{80}x^{21}+\alpha^{107}x^{20}+\alpha^{81}x^{19}+\alpha^{13}x^{18}+\alpha^{83}x^{17}+\alpha^{44}x^{16}+\alpha^{7}x^{15}+\alpha^{32}x^{14}+\alpha^{5}x^{13}+\alpha^{4}x^{12}+\alpha^{27}x^{11}+\alpha^{2}x^{10}+\alpha^{124}x^{9}+\alpha^{104}x^{8}+\alpha^{111}x^{7}+\alpha^{119}x^{6}+\alpha^{90}x^{5}+\alpha^{114}x^{4}+\alpha^{91}x^{3}+\alpha^{45}x^{2}+\alpha^{9}x+\alpha^{72}$
57	$x^{57}+\alpha^{9}x^{56}+\alpha^{100}x^{55}+\alpha^{122}x^{54}+\alpha^{109}x^{53}+\alpha^{105}x^{52}+\alpha^{80}x^{51}+\alpha^{51}x^{50}+\alpha^{110}x^{49}+\alpha^{111}x^{48}+\alpha^{69}x^{47}+\alpha^{8}x^{46}+\alpha^{97}x^{45}+\alpha^{69}x^{44}+\alpha^{86}x^{43}+\alpha^{114}x^{42}+\alpha^{95}x^{41}+\alpha^{109}x^{40}+\alpha^{60}x^{39}+\alpha^{30}x^{38}+\alpha^{61}x^{37}+\alpha^{52}x^{36}+\alpha^{55}x^{35}+\alpha^{80}x^{34}+\alpha^{59}x^{33}+\alpha^{82}x^{32}+\alpha^{29}x^{31}+\alpha^{67}x^{30}+\alpha^{82}x^{29}+\alpha^{111}x^{28}+\alpha^{27}x^{27}+\alpha^{47}x^{26}+\alpha^{31}x^{25}+\alpha^{66}x^{24}+\alpha^{18}x^{23}+\alpha^{51}x^{22}+\alpha^{106}x^{21}+\alpha^{48}x^{20}+\alpha^{73}x^{19}+\alpha^{34}x^{18}+\alpha^{14}x^{17}+\alpha^{58}x^{16}+\alpha^{8}x^{15}+\alpha^{38}x^{14}+\alpha^{79}x^{13}+\alpha^{38}x^{12}+\alpha^{7}x^{11}+\alpha^{126}x^{10}+\alpha^{99}x^{9}+\alpha^{29}x^{8}+\alpha^{28}x^{7}+\alpha^{115}x^{6}+\alpha^{71}x^{5}+\alpha^{6}x^{4}+\alpha^{77}x^{3}+\alpha^{113}x^{2}+\alpha^{80}x+\alpha^{2}$
58	$x^{58}+\alpha^{45}x^{57}+\alpha^{24}x^{56}+\alpha^{44}x^{55}+\alpha^{77}x^{54}+\alpha^{107}x^{53}+\alpha^{45}x^{52}+\alpha^{32}x^{51}+\alpha^{13}x^{50}+\alpha^{23}x^{49}+\alpha^{32}x^{48}+\alpha^{121}x^{47}+\alpha^{67}x^{46}+\alpha^{94}x^{45}+\alpha^{25}x^{44}+\alpha^{113}x^{43}+\alpha^{90}x^{42}+\alpha^{116}x^{41}+\alpha^{114}x^{40}+\alpha^{48}x^{39}+\alpha^{51}x^{38}+\alpha^{117}x^{37}+\alpha^{81}x^{36}+\alpha^{37}x^{35}+\alpha^{123}x^{34}+\alpha^{73}x^{33}+\alpha x^{32}+\alpha^{14}x^{31}+\alpha^{31}x^{30}+\alpha^{85}x^{29}+\alpha^{90}x^{28}+\alpha^{5}x^{27}+\alpha^{51}x^{26}+\alpha^{55}x^{25}+\alpha^{87}x^{24}+\alpha^{10}x^{23}+\alpha^{113}x^{22}+\alpha^{81}x^{21}+\alpha^{74}x^{20}+\alpha^{3}x^{19}+\alpha x^{18}+\alpha^{64}x^{17}+\alpha^{95}x^{16}+\alpha^{50}x^{15}+\alpha^{21}x^{14}+\alpha^{22}x^{13}+\alpha^{54}x^{12}+\alpha^{40}x^{11}+\alpha^{10}x^{10}+\alpha^{60}x^{9}+\alpha^{109}x^{8}+\alpha^{60}x^{7}+\alpha^{5}x^{6}+\alpha^{126}x^{5}+\alpha^{28}x^{4}+\alpha^{54}x^{3}+\alpha^{93}x^{2}+\alpha^{46}x+\alpha^{60}$
59	$x^{59}+\alpha^{51}x^{58}+\alpha^{56}x^{57}+\alpha^{101}x^{56}+\alpha^{5}x^{55}+\alpha^{81}x^{54}+\alpha^{53}x^{53}+\alpha^{3}x^{52}+x^{51}+\alpha^{59}x^{50}+\alpha^{77}x^{49}+\alpha^{90}x^{48}+\alpha^{59}x^{47}+\alpha^{70}x^{46}+\alpha^{56}x^{45}+\alpha^{58}x^{44}+\alpha^{95}x^{43}+\alpha^{119}x^{42}+\alpha^{2}x^{41}+\alpha^{108}x^{40}+\alpha^{75}x^{39}+\alpha^{113}x^{38}+\alpha^{25}x^{37}+\alpha^{69}x^{36}+\alpha^{86}x^{35}+\alpha^{16}x^{34}+\alpha^{125}x^{33}+\alpha^{119}x^{32}+\alpha^{111}x^{31}+\alpha^{40}x^{30}+\alpha^{70}x^{29}+\alpha^{74}x^{28}+\alpha^{15}x^{27}+\alpha^{81}x^{26}+\alpha^{32}x^{25}+\alpha^{35}x^{24}+\alpha^{78}x^{23}+\alpha^{94}x^{22}+\alpha^{115}x^{21}+\alpha^{10}x^{20}+\alpha^{103}x^{19}+\alpha^{57}x^{18}+\alpha^{107}x^{17}+\alpha^{16}x^{16}+\alpha^{39}x^{15}+\alpha^{97}x^{14}+\alpha^{44}x^{13}+\alpha^{93}x^{12}+\alpha^{57}x^{11}+\alpha^{104}x^{10}+\alpha^{19}x^{9}+\alpha^{20}x^{8}+\alpha^{83}x^{7}+\alpha^{66}x^{6}+\alpha^{27}x^{5}+\alpha^{11}x^{4}+\alpha^{40}x^{3}+\alpha^{65}x^{2}+\alpha^{110}x+\alpha^{119}$

表 D.1（续）

纠错码字数	生成多项式
60	$x^{60}+\alpha^{95}x^{59}+\alpha^{116}x^{58}+\alpha^{60}x^{57}+\alpha^{106}x^{56}+\alpha^{53}x^{55}+\alpha^{71}x^{54}+\alpha^{55}x^{53}+\alpha^{15}x^{52}+\alpha^{90}x^{51}+\alpha^{30}x^{50}+\alpha^{52}x^{49}+\alpha^{72}x^{48}+\alpha^{106}x^{47}+\alpha^{76}x^{46}+\alpha^{6}x^{45}+\alpha^{84}x^{44}+\alpha^{41}x^{43}+\alpha^{47}x^{42}+\alpha^{40}x^{41}+\alpha^{52}x^{40}+\alpha^{54}x^{39}+\alpha^{65}x^{38}+\alpha^{57}x^{37}+\alpha^{35}x^{36}+\alpha^{23}x^{35}+\alpha^{112}x^{34}+\alpha^{33}x^{33}+\alpha^{8}x^{32}+\alpha^{37}x^{31}+\alpha^{69}x^{30}+\alpha^{98}x^{29}+\alpha x^{28}+\alpha^{89}x^{27}+\alpha^{102}x^{26}+\alpha^{74}x^{25}+\alpha^{20}x^{24}+\alpha^{103}x^{23}+\alpha^{45}x^{22}+\alpha^{95}x^{21}+\alpha^{27}x^{20}+\alpha^{76}x^{19}+\alpha^{17}x^{18}+\alpha^{72}x^{17}+\alpha^{49}x^{16}+\alpha^{32}x^{15}+\alpha^{36}x^{14}+x^{13}+\alpha^{27}x^{12}+\alpha^{68}x^{11}+\alpha^{107}x^{10}+\alpha^{101}x^{9}+\alpha^{87}x^{8}+\alpha^{81}x^{7}+\alpha^{11}x^{6}+\alpha^{54}x^{5}+\alpha^{41}x^{4}+\alpha^{56}x^{3}+\alpha^{46}x^{2}+\alpha^{86}x+\alpha^{52}$
61	$x^{61}+\alpha^{72}x^{60}+\alpha^{10}x^{59}+\alpha^{87}x^{58}+\alpha^{42}x^{57}+\alpha^{4}x^{56}+\alpha^{20}x^{55}+\alpha^{50}x^{54}+\alpha^{44}x^{53}+\alpha^{82}x^{52}+\alpha^{38}x^{51}+\alpha^{109}x^{50}+\alpha^{11}x^{49}+\alpha^{96}x^{48}+\alpha^{89}x^{47}+\alpha^{3}x^{46}+\alpha^{9}x^{45}+\alpha^{7}x^{44}+\alpha^{73}x^{43}+\alpha^{62}x^{42}+\alpha^{88}x^{41}+\alpha^{8}x^{40}+\alpha^{110}x^{39}+\alpha^{74}x^{38}+x^{37}+\alpha^{76}x^{36}+\alpha^{96}x^{35}+\alpha^{124}x^{34}+\alpha^{24}x^{33}+\alpha^{36}x^{32}+\alpha^{43}x^{31}+\alpha^{74}x^{30}+\alpha^{2}x^{29}+\alpha^{52}x^{28}+\alpha^{87}x^{27}+\alpha^{121}x^{26}+\alpha^{36}x^{25}+\alpha^{22}x^{24}+\alpha^{31}x^{23}+\alpha^{2}x^{22}+\alpha^{89}x^{21}+\alpha^{104}x^{20}+\alpha^{13}x^{19}+\alpha^{86}x^{18}+\alpha^{82}x^{17}+\alpha^{19}x^{16}+\alpha^{75}x^{15}+\alpha^{98}x^{14}+\alpha^{38}x^{13}+\alpha^{15}x^{12}+\alpha^{48}x^{11}+\alpha^{39}x^{10}+\alpha^{18}x^{9}+\alpha^{42}x^{8}+\alpha^{110}x^{7}+\alpha^{15}x^{6}+\alpha^{61}x^{5}+\alpha^{34}x^{4}+\alpha^{14}x^{3}+\alpha^{126}x^{2}+\alpha^{123}x+\alpha^{113}$
62	$x^{62}+\alpha^{99}x^{61}+\alpha^{14}x^{60}+\alpha^{8}x^{59}+\alpha^{96}x^{58}+\alpha^{94}x^{57}+\alpha^{125}x^{56}+\alpha^{26}x^{55}+\alpha^{66}x^{54}+\alpha^{11}x^{53}+\alpha^{57}x^{52}+\alpha^{17}x^{51}+\alpha^{95}x^{50}+\alpha^{62}x^{49}+\alpha^{106}x^{48}+\alpha^{43}x^{47}+\alpha^{33}x^{46}+\alpha^{86}x^{45}+\alpha^{66}x^{44}+\alpha^{115}x^{43}+\alpha^{10}x^{42}+\alpha^{71}x^{41}+\alpha^{91}x^{40}+\alpha^{19}x^{39}+\alpha^{44}x^{38}+\alpha^{68}x^{37}+\alpha^{49}x^{36}+\alpha^{8}x^{35}+\alpha^{15}x^{34}+\alpha^{81}x^{33}+\alpha^{69}x^{32}+\alpha^{75}x^{31}+\alpha^{5}x^{30}+\alpha^{80}x^{29}+\alpha^{77}x^{28}+\alpha^{6}x^{27}+\alpha^{110}x^{26}+\alpha^{65}x^{25}+\alpha^{104}x^{24}+\alpha^{15}x^{23}+\alpha^{23}x^{22}+\alpha^{66}x^{21}+\alpha^{68}x^{20}+\alpha^{109}x^{19}+\alpha^{123}x^{18}+\alpha^{79}x^{17}+\alpha^{89}x^{16}+\alpha^{35}x^{15}+\alpha^{34}x^{14}+\alpha^{53}x^{13}+\alpha^{22}x^{12}+\alpha^{7}x^{11}+\alpha^{110}x^{10}+x^{9}+\alpha^{118}x^{8}+\alpha^{14}x^{7}+\alpha^{49}x^{6}+\alpha^{81}x^{5}+\alpha^{19}x^{4}+\alpha^{121}x^{3}+\alpha^{63}x^{2}+\alpha^{84}x+\alpha^{48}$
63	$x^{63}+\alpha^{112}x^{62}+\alpha^{54}x^{61}+\alpha^{25}x^{60}+\alpha^{30}x^{59}+\alpha^{34}x^{58}+\alpha^{101}x^{57}+\alpha^{17}x^{56}+\alpha^{55}x^{55}+\alpha^{46}x^{54}+\alpha^{126}x^{53}+\alpha^{49}x^{52}+\alpha^{16}x^{51}+\alpha^{32}x^{50}+\alpha^{85}x^{49}+\alpha^{73}x^{48}+\alpha^{86}x^{47}+\alpha^{123}x^{46}+\alpha^{31}x^{45}+\alpha^{121}x^{44}+\alpha^{76}x^{43}+\alpha^{6}x^{42}+\alpha^{40}x^{41}+\alpha^{13}x^{40}+\alpha^{2}x^{39}+\alpha^{125}x^{38}+\alpha^{54}x^{37}+\alpha^{101}x^{36}+\alpha^{39}x^{35}+\alpha^{85}x^{34}+x^{33}+\alpha^{114}x^{32}+\alpha^{19}x^{31}+\alpha^{96}x^{30}+\alpha^{118}x^{29}+\alpha^{9}x^{28}+\alpha^{8}x^{27}+\alpha^{25}x^{26}+\alpha^{33}x^{25}+\alpha^{101}x^{24}+\alpha^{49}x^{23}+\alpha^{13}x^{22}+\alpha^{43}x^{21}+\alpha^{50}x^{20}+\alpha^{32}x^{19}+\alpha^{6}x^{18}+\alpha^{35}x^{17}+\alpha^{62}x^{16}+\alpha^{113}x^{15}+\alpha^{62}x^{14}+\alpha^{73}x^{13}+\alpha^{121}x^{12}+\alpha^{91}x^{11}+\alpha^{105}x^{10}+\alpha^{89}x^{9}+\alpha^{35}x^{8}+\alpha^{61}x^{7}+\alpha^{82}x^{6}+\alpha^{79}x^{5}+\alpha^{12}x^{4}+\alpha^{71}x^{3}+\alpha^{37}x^{2}+\alpha^{32}x+\alpha^{111}$

附 录 E
（资料性附录）
参考译码算法

E.1 灰阶图像二值化

将灰阶图像分割为小块，使用 Otsu 算法（见参考文献[1]）分别计算每块的二值化阈值，对相邻块的二值化阈值进行平均平滑后，各图像块使用各自的阈值进行二值化。

将灰阶图像分割为 $M\times M$ 像素的图像块（这里 $M=40$）。如果图像高度或宽度不是 M 的整数倍，则采取以下分块算法：图像宽度 W 被分为 BW 份，前 BW_1 份的宽度为 MW_1，后 BW_2 份的宽度为 MW_2。图像高度 H 被分为 BH 份，前 BH_1 份的高度为 MH_1，后 BH_2 份的高度为 MH_2：

$$BW=(W+M-1)\ \text{DIV}\ M$$
$$MW_1=W\ \text{DIV}\ BW$$
$$MW_2=MW_1+1$$
$$BW_1=BW\times MW_2-W$$
$$BW_2=BW-BW_1$$
$$BH=(H+M-1)\ \text{DIV}\ M$$
$$MH_1=H\ \text{DIV}\ BH$$
$$MH_2=MH_1+1$$
$$BH_1=BH\times MH_2-H$$
$$BH_2=BH-BH_1$$

各图像块的对比度是指该块内最暗与最亮像素间的灰阶差。记对比度最大的图像块的灰阶差为 max_sc。若一个图像块的对比度小于 max_sc/2，该图像块被认为处于背景区域，整个图像块被二值化为白色并且该块的二值化域值设为 0。否则，使用 Otsu 二值化算法计算该图像块的二值化域值。

对相邻块的二值化阈值进行平均平滑。所有的二值化阈值组成一个二维数组，所有非 0 的阈值与其周围 8 个块的非 0 阈值参与平均平滑。各图像块阈值的权值模板为：

1	2	1
2	4	2
1	2	1

根据阈值对各图像块进行二值化，灰阶小于阈值的像素被二值化为黑色，其余像素为白色，从而得到二值图像。

图 E.1 和图 E.2 分别是灰阶图像和二值化的结果。

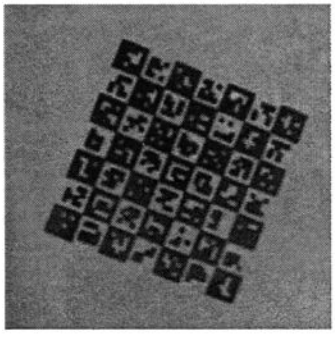


图 E.1 GM 码灰阶图像

图 E.2 二值图像

E.2 获取边界图像及连通分支

称二值图像中的一个像素点为边界点是指该像素点本身为白色且其周围的 4 个像素至少有一个为黑色。由二值图像的所有边界点组成的图像称为其边界图像。根据 8 连通将边界图像分解为连通分支的并。一个连通分支的尺寸定义为它的凸包的面积。删除所有尺寸小于 625 的连通分支(625 是 5×5 个单元模块的最小面积,本参考译码算法要求单元模块至少应包含 5×5 个像素)。

图 E.3 是边界图像。图 E.4 是删去小连通分支后的边界图像。

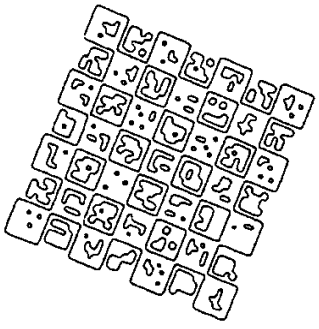


图 E.3 边界图像

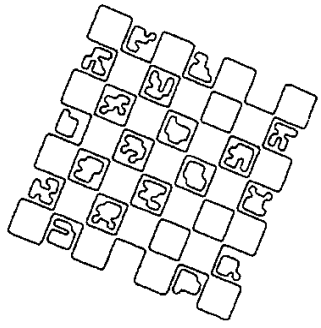


图 E.4 删去小连通分支后的边界图像

E.3 估计 GM 码的水平斜率和竖直斜率

水平斜率是指 GM 码图的水平方向与 x 轴夹角的正切值,竖直斜率是指 GM 码图的竖直方向与 y 轴夹角的正切值。水平斜率和竖直斜率应分别进行计算。

使用删去小连通分支后的边界图像对水平斜率和竖直斜率进行估算。设当前像素点为(x,y)(下面的矩阵中以“*”号表示),它的 8 个相邻像素点分别为 $P_0=(x+1,y-1)$,$P_1=(x+1,y)$,$P_2=(x+1,y+1)$,$P_3=(x,y+1)$,$P_4=(x-1,y+1)$,$P_5=(x-1,y)$,$P_6=(x-1,y-1)$,$P_7=(x,y-1)$。

$$\begin{matrix} P_6 & P_7 & P_0 \\ P_5 & * & P_1 \\ P_4 & P_3 & P_2 \end{matrix}$$

定义数组 dir[8],这里 dir[i]是当前像素与其周围的 P_i 像素点同时为黑色的情形的总数。定义数组 dd[8][8],这里 dd[i][j]是当前像素与其周围的 P_i 及 P_j 像素点同时为黑色的情形的总数。

当 dir[0]+dir[2]<dir[1]+dir[3]且 dd[4][1]+dd[5][0]+dd[6][3]+dd[7][2]>dd[6][1]+dd[5][2] +dd[7][4]+dd[0][3]时:

水平斜率 =−(dd[4][1]+dd[5][0])/(dir[1]×2+ dd[4][1]+dd[5][0]+1)

竖直斜率 = (dd[6][3]+dd[7][2])/(dir[3]×2+dd[6][3]+dd[7][2]+1)

当 dir[0]+dir[2]<dir[1]+dir[3]且 dd[4][1]+dd[5][0]+dd[6][3]+dd[7][2]≤dd[6][1]+dd[5][2] +dd[7][4]+dd[0][3]时:

水平斜率= (dd[6][1]+dd[5][2])/(dir[1]×2+dd[6][1]+dd[5][2]+1)

竖直斜率=−(dd[7][4]+dd[0][3])/(dir[3]×2+dd[7][4]+dd[0][3]+1)

当 dir[0]+dir[2]≥dir[1]+dir[3]且 dd[4][1]+dd[5][0]+dd[6][3]+dd[7][2]>dd[6][1]+dd[5][2] +dd[7][4]+dd[0][3]时:

水平斜率=−(dir[0]×2)/(dir[0]×2+dd[5][0]+dd[4][1]+1)

竖直斜率＝(dir[2]×2)/(dir[2]×2+ dd[7][2]+dd[6][3]+1)

当 dir[0]+dir[2]≥dir[1]+dir[3]且 dd[4][1]+dd[5][0]+dd[6][3]+dd[7][2]≤dd[6][1]+dd[5][2]+dd[7][4]+dd[0][3]时：

水平斜率＝(dir[2]×2)/(dir[2]×2+dd[5][2]+dd[6][1]+1)

竖直斜率＝－(dir[0]×2)/(dir[0]×2+dd[7][4]+dd[0][3]+1)

E.4 估计宏模块的面积

根据删去小连通分支后的边界图像和 GM 码的水平斜率和竖直斜率可以估算出宏模块的面积。

称二值图像中的一个像素点为左边界点是指：

——该像素点属于删去小连通分支后的边界图像；

——该像素点的“右侧像素点”为黑色。

这里像素点(x,y)的“右侧像素点”根据 GM 码的水平斜率和竖直斜率定义如下(0.414 是 22.5°的正切值)：

a) 当(ABS(竖直斜率)＋ABS(水平斜率))≤0.414×2 时，(x,y)的“右侧像素点”为$(x+1,y)$；

b) 当(ABS(竖直斜率)＋ABS(水平斜率))＞0.414×2 且竖直斜率＞0 时，(x,y)的“右侧像素点”为$(x+1,y-1)$；

c) 当(ABS(竖直斜率)＋ABS(水平斜率))＞0.414×2 且竖直斜率≤0 时，(x,y)的“右侧像素点”为$(x+1,y+1)$。

所有左边界点组成的图像称为左边界图像。见图 E.5，左边界图像中有一些孤立的像素点，根据以下规则去除孤立像素点：

当竖直斜率＞0 且$(x,y-1)$，$(x,y+1)$，$(x-1,y-1)$，$(x+1,y+1)$都不是左边界点时，从左边界图像中删去(x,y)；当竖直斜率≤0 且$(x,y-1)$，$(x,y+1)$，$(x+1,y-1)$，$(x-1,y+1)$ 都不是左边界点时，从左边界图像中删去(x,y)。

图 E.6 是去除孤立像素点后的左边界图像。

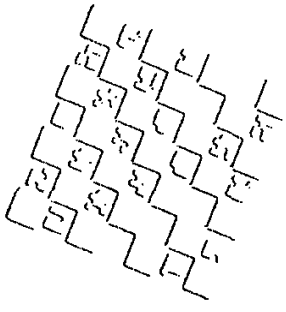


图 E.5 左边界图像

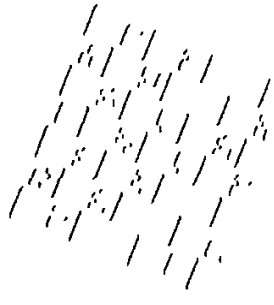


图 E.6 去除孤立像素点后的左边界图像

去除孤立像素点后的左边界图像分解为连通分支的并，各连通分支的直径定义为该连通分支内距离最远的两点直接的距离，并四舍五入到最接近的整数。删去直径小于 26 的连通分支。

对右边界图像实施同样的步骤。设 $len[i]$是左右边界图像中直径为 i 的连通分支的总个数，$P[i]$和 $Q[i]$的定义分别见式(E.1)和式(E.2)。

$$P[i]=\sum_{u\leqslant j\leqslant v} len[j] \qquad \cdots\cdots(E.1)$$

$$Q[i]=\sum_{u\leqslant j\leqslant v} len[j]\times j^2 \qquad \cdots\cdots(E.2)$$

式中：

$u=\lfloor 11\ i/12 \rfloor$

$v=\lceil 13\ i/12 \rceil$

设 $Q[i_0]$是满足 $i\geqslant 26$ 且 $P[i]\geqslant 10$ 的所有 $Q[i]$的最大值，则宏模块高度的估计值见式(E.3)。

$$\text{apprx\_height}=\frac{\sum_{u_0\leqslant i\leqslant v_0} i\times len[i]}{\sum_{u_0\leqslant i\leqslant v_0} len[i]} \qquad \cdots\cdots(E.3)$$

式中：

$u_0=\lfloor 11\ i_0/12 \rfloor$

$v_0=\lceil 13\ i_0/12 \rceil$

使用同样的方法可计算得到宏模块的估计宽度 apprx_width，宏模块的估计面积为：

$$\text{apprx\_area} = \text{apprx\_height}\times\text{apprx\_width}$$

从边界图像删去所有面积小于 apprx_area 的 70%的连通分支，得到的图像称为框架图像，见图 E.7。

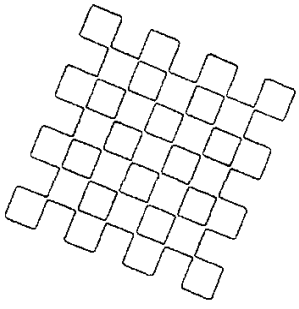


图 E.7 框架图像

得到框架图像后，使用 E.3 的方法重新计算 GM 码的水平斜率和竖直斜率。

E.5 计算宏模块的中心

以下步骤根据框架图像计算宏模块的中心：

a) 计算初始内部点 O 为框架图像的重心。若 O 正好位于框架图像的边上，则从 O 点开始顺时针方向螺旋向外搜索初始内部点，步长为宏模块边长的一半，初始搜索方向为 GM 码的竖直方向，直至得到第一个非边界点，仍记为 O。

b) 经过点 O 以码图的水平方向画一条直线，直线与框架图像左右分别相交于 A、B 两点。AB 的中心记为 O_1，见图 E.8。

c) 经过点 O_1 以码图的竖直方向画一条直线，直线与框架图像上下分别相交于 C、D 两点。CD 的中心记为 O_2，见图 E.8。

d) O_2 是调整后的宏模块中心。将 O_2 作为初始中心重复第 b)、c)步，直至调整后的中心与初始中心间的距离不超过 2 像素。若重复第 b)、c)步超过 10 次后距离仍大于 2 像素，则认为该宏模块中心搜索失败。

e) 至此 AB 为宏模块的宽度，CD 为宏模块的高度。如果 $AB>$ini_width$\times 7/6$ 或$AB<$ini_width$\times$

5/6 或 $CD>$ini_height×7/6 或 $CD<$ini_height×5/6，则认为该宏模块中心搜索失败。

f) 当一个宏模块中心被找到后，根据该宏模块的中心、码图的水平斜率和竖直斜率方向估计其周围 8 个宏模块的初始中心、初始高度和初始宽度，然后对这 8 个宏模块分别重复第 b)、c)、d)、e)步骤对中心进行调整。

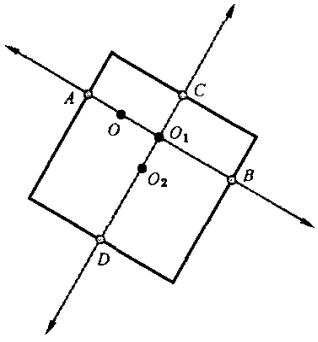
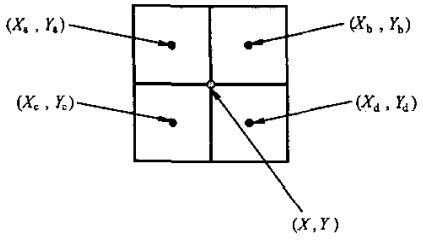


图 E.8 计算宏模块中心

E.6 计算宏模块的顶点

对邻近的宏模块中心插值可以计算得到宏模块的顶点(X,Y)。因为码图可能被污损，部分宏模块中心可能会丢失。以下给出了 5 种可能情况的插值公式：

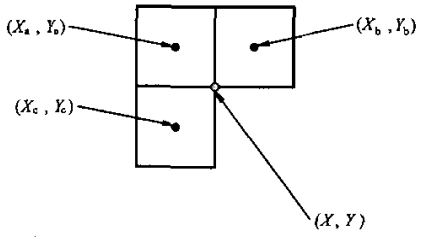
a) 4 个宏模块中心：



$$X=(X_a+X_b+X_c+X_d)/4$$

$$Y=(Y_a+Y_b+Y_c+Y_d)/4$$

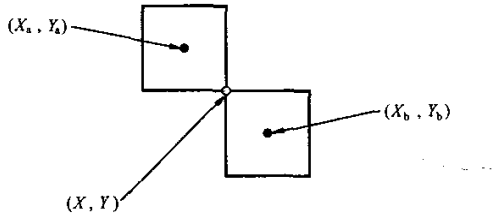
b) 3 个宏模块中心：



$$X=(X_b+X_c)/2$$

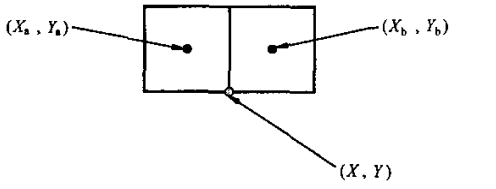
$$Y=(Y_b+Y_c)/2$$

c) 两个对角位置的宏模块中心：



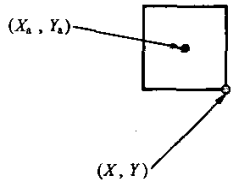
$$X=(X_a+X_b)/2$$
$$Y=(Y_a+Y_b)/2$$

d) 两个并排位置的宏模块中心：



该顶点根据(X_a,Y_a)、(X_b,Y_b)，宏模块的宽度、高度以及GM码的水平斜率和竖直斜率进行计算。本例中为(X_a,Y_a)、(X_b,Y_b)的中点沿竖直方向向下走这两个宏模块的平均高度的一半得到(X,Y)。

e) 1个宏模块中心：



该顶点根据(X_a,Y_a)、宏模块的宽度、高度以及GM码的水平斜率和竖直斜率进行计算。本例中为(X_a,Y_a)沿水平方向向右走宏模块宽度的一半，并沿竖直方向向下走宏模块高度的一半得到(X,Y)。

E.7 数据采样

每个宏模块由6×6单元模块组成。根据宏模块的顶点可以计算这些单元模块的中心位置。

设宏模块的四个顶点为(X_0,Y_0)、(X_1,Y_1)、(X_2,Y_2)、(X_3,Y_3)，坐标变换公式见式(E.4)。

$$X=k_1\times x+k_2\times y+k_3\times x\times y+k_4 \quad \cdots\cdots(E.4)$$
$$Y=k_5\times x+k_6\times y+k_7\times x\times y+k_8$$

式中：

(x,y)——变换前的坐标；

(X,Y)——变换后的坐标；

$k_1\sim k_8$——待定的参数。

图E.9是坐标变换公式的示意图。

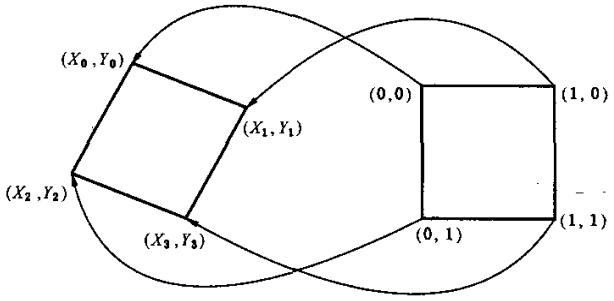


图 E.9 坐标变换

计算参数 k_1 到 k_8 如下：

$$X_0 = k_1\times0 + k_2\times0 + k_3\times0\times0 + k_4 \quad \Rightarrow k_4 = X_0$$
$$X_1 = k_1\times1 + k_2\times0 + k_3\times1\times0 + k_4 \quad \Rightarrow k_1 = X_1 - X_0$$
$$X_2 = k_1\times0 + k_2\times1 + k_3\times0\times1 + k_4 \quad \Rightarrow k_2 = X_2 - X_0$$
$$X_3 = k_1\times1 + k_2\times1 + k_3\times1\times1 + k_4 \quad \Rightarrow k_3 = X_3 - X_1 - X_2 + X_0$$
$$Y_0 = k_5\times0 + k_6\times0 + k_7\times0\times0 + k_8 \quad \Rightarrow k_8 = Y_0$$
$$Y_1 = k_5\times1 + k_6\times0 + k_7\times1\times0 + k_8 \quad \Rightarrow k_5 = Y_1 - Y_0$$
$$Y_2 = k_5\times0 + k_6\times1 + k_7\times0\times1 + k_8 \quad \Rightarrow k_6 = Y_2 - Y_0$$
$$Y_3 = k_5\times1 + k_6\times1 + k_7\times1\times1 + k_8 \quad \Rightarrow k_7 = Y_3 - Y_1 - Y_2 + Y_0$$

最外一层单元模块为边框，不表示数据。内部 4×4 单元模块分别表示 1 位。将 x=3/12,5/12,7/12,9/12 和 y=3/12,5/12,7/12,9/12 分别代入式(E.4)，即可得到所有内部单元模块中心的坐标。

设 w 是单元模块的宽度和高度的平均值，读取以单元模块中心为圆心、$w/4$ 为半径的圆形区域内所有像素点。若多数像素为黑色，该单元模块值为“1”，否则该单元模块值为“0”。

E.8 推导 GM 码的方向、中心宏模块和纠错等级

GM 码没有用于指明方向的功能图形，但通过综合分析宏模块的层标识号可推导出 GM 码的方向。

从正确的方向读取层标识号，则获得的层标识号呈环状循环分布。如果从其他方向读取层标识号，读取到的单元模块不代表层标识号，从而得到的结果是随机内容。由此与期望模式匹配得最好的方向被认为是正确的方向。

层标识号与期望模式的匹配度根据以下参数进行计算：

——8 个方向(正常的 4 个方向和镜像的 4 个方向)。

——5 个纠错等级。

——中心宏模块的位置。宏模块数组中心附近的多个宏模块都可以被认为是中心宏模块。中心宏模块边框的颜色用于确定 GM 码是否反色。

对每一个可能的参数组合，按以下方法计算一个分值：

记 $L[i,j]$为第 i 列第 j 行的宏模块的层标识号。设中心宏模块的位置为第 x_c 列第 y_c 行，纠错等级为 e，记 $LT[i,j;x_c,y_c,e]$为第 i 列第 j 行的宏模块的层标识号的理论值。当 e 等于 1 时，$LT[i,j;x_c,y_c,e] = 3-(\mathrm{MAX}(\mathrm{ABS}(i-x_c),\mathrm{ABS}(j-y_c))\ \mathrm{MOD}\ 4)$；当 e 不等于 1 时，$LT[i,j;x_c,y_c,e] = (\mathrm{MAX}(\mathrm{ABS}(i-x_c),\mathrm{ABS}(j-y_c))+5-e)\ \mathrm{MOD}\ 4$。

第 i 列第 j 行的宏模块的分值 $S[i,j;x_c,y_c,e]$定义为($L[i,j]$ XOR $LT[i,j;x_c,y_c,e]$)中位“1”的个数。

GM 码的总分值 $T[x_c,y_c,e]$见式(E.5)。

$$T[x_c,y_c,e]=\sum_{i,j}S[i,j;x_c,y_c,e] \quad \cdots\cdots(E.5)$$

一般地，得分最低的情况对应于正确的中心、方向和纠错等级。但在以下一些情况下，得分最低的情况不一定正好对应于正确的中心、方向和纠错等级，从而得分最低的情形不能得到正确的解码：

a) 由于 GM 码被污损，(x_c,y_c)可能不在 GM 码的正中心，见图 E.10。

b) 由于 GM 码是以中心宏模块为中心的方阵，我们可以根据假定的中心宏模块计算丢失的宏模块个数。根据 GM 码各纠错等级纠错码字所占的比例，可以得知若丢失宏模块个数大于纠错等级×10%×宏模块总个数，则必定解码失败，故这种情况不必再计算总分值。

c) 在一些特殊的情况下，从错误的方向读取到的层标识号有可能正好能够符合层标识号的分布规律，加上噪声的存在，有可能导致从错误方向得到的分值小于正确方向得到的分值。

所有(总分值＋丢失宏模块个数)＜(纠错等级×10%×宏模块总个数)的情形应当按总分值从小到大的次序依次尝试解码，直至解码成功或所有情形均解码失败。

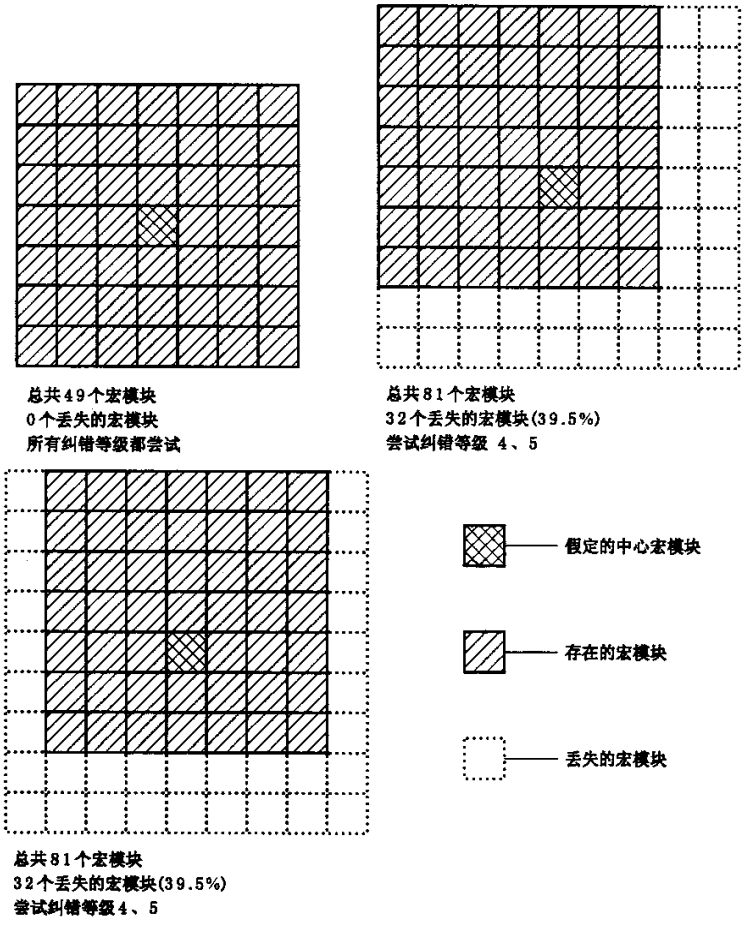


图 E.10 存在的宏模块，假定的中心宏模块和丢失的宏模块

E.9 还原码字并纠错

从 GM 码的中心宏模块开始，按照顺时针螺旋方向(见图 7)依次读取码字。若 GM 码是镜像的则

反转螺旋的方向。每个宏模块中先读取第 1 码字，再读取第 2 码字。如果总码字数大于 127，则按 6.6.3 的分块法则以及 6.7.3 的纠错块交错规则还原码字顺序。对每一纠错块分别应用 Reed-Solomon 纠错算法进行纠错后得到原始的数据码字。

E.10 编码位流解码

错误码字被纠正后，将数据码字组装成二进制位流，按下面的流程译码二进制位流即可还原数据：

a) 读取开头 4 位数据得到当前的数据模式；
b) 根据当前数据模式的编码规则，对二进制位流进行解码，直至遇到模式转换码；
c) 查模式转换码表(表 8)，确定下一编码模式，如果为数据结束标志则整个译码过程结束，否则跳转到步骤 b)。

若位流不是以数据结束标志结束，或填充位、填充码字(如果存在)不符合定义的规则，解码过程应返回失败。

参 考 文 献

[1] IEEE Trans. Sys. ,Man. N. Otsu (1979). "A threshold selection method from gray-level histograms". IEEE Trans. Sys. ,Man. ,Cyber. 9:62-66