

[19] 中华人民共和国国家知识产权局

[51] Int. Cl⁷

H03M 7/14

G11B 7/14



[12] 发明专利说明书

[21] ZL 专利号 02102441.3

[45] 授权公告日 2005 年 6 月 8 日

[11] 授权公告号 CN 1205754C

[22] 申请日 2002.1.18 [21] 申请号 02102441.3

[30] 优先权

[32] 2001. 2. 2 [33] JP [31] 026967/2001

[32] 2001. 6. 12 [33] JP [31] 177408/2001

[71] 专利权人 日本胜利株式会社

地址 日本神奈川县

[72] 发明人 冲 刚 速水淳

审查员 王艳妮

[74] 专利代理机构 中原信达知识产权代理有限责
任公司

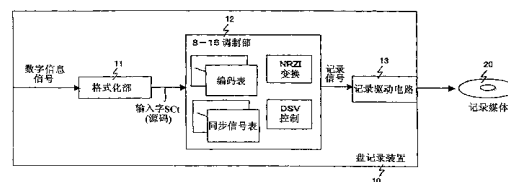
代理人 谢丽娜 方 挺

权利要求书 4 页 说明书 28 页 附图 14 页

[54] 发明名称 同步信号生成方法、记录装置以及
传输装置

[57] 摘要

将重放数据解码用的同步信号插入到代码字串。用多个编码表将 p 比特的输入数据字变换成 q 比特(其中, $q > p$)代码字, 将使该代码字之间直接结合的代码字串记录在光盘或磁盘等记录媒体上和进行重放, 或在通过传输部传输代码字串时, 同步帧由同步信号和满足最小游程长度和最大游程长度限制的代码字串组成, 上述同步信号可与满足规定的游程长度限制规则的代码字串分离, 并且由用于识别一个扇区内的位置的特定码、以及游程长度比规定的游程长度限制规则的最大游程长度大 $1T$ 以上的第一比特码型和接续该第一比特码型游程长度比最小游程长度大的第二比特码型组成的同步码型来构成。



1. 一种同步信号生成方法，用于多个编码表对 p 比特的输入数据字进行调制以获得 q 比特的代码字时，其中， $q > p$ ，所述多个编码表存储对应于各个输入数据字的代码字和表示为了获得即使该代码字之间直接结合生成代码字串也满足规定的游程长度限制规则的下一个代码字、而对下一个输入数据字进行调制所用的编码表的状态信息，所述同步信号生成方法包括：

生成重放数据解码用的同步信号的步骤，能够与满足所述规定的游程长度限制规则的所述代码字串分离，并且由用于识别一个扇区内的位置的特定码、游程长度比所述规定的游程长度限制规则中的最大游程长度大 $1T$ 以上的第一比特码型和接续该第一比特码型游程长度大的第二比特码型组成的同步码型构成，其中， T 是所述代码字的信道比特周期，并且所述同步码型包括接续其后的一部分代码字；

生成同步帧的步骤，将所述重放数据解码用的同步信号以每个规定的代码字数目插入到满足所述规定的游程长度限制规则输出的代码字串中。

2. 如权利要求 1 所述的同步信号生成方法，其特征在于，使所述多个编码表中的特定编码表和另一特定的编码表对应于预先设定的规定的输入数据字所存储的各个代码字中为‘1’的数目可进行数字加和值 DSV 控制，具有如果一方的编码表中的代码字中为偶数个则另一方的编码表中的代码字中为奇数个这样的奇偶性来进行分配，在对所述规定的输入数据字进行调制时，在用所述特定的编码表调制的代码字获得的 DSV 值的绝对值和用另一特定的编码表调制的代码字获得的 DSV 值的绝对值中，通过选择绝对值小的一方的代码字，来进行 DSV 控制，并且将重放数据解码用的同步信号以每个规定的代码字数目插入到满足所述规定的游程长度限制规则输出的代码字串中而生成同步帧。

3. 如权利要求 1 或权利要求 2 所述的同步信号生成方法，其特征在于，对应于所述多个编码表准备多个同步信号表，并且在各同步信号表内设定多个用于生成所述同步信号的同步信号比特码型，并且各同步信号比特码型通过 DSV 控制可选择具有如果一方的同步信号比特码型为‘1’的数目为偶数个则另一方的同步信号比特码型就为奇数个这样的奇偶性的两个比特码型的其中一个。

4. 如权利要求 1 或 2 所述的同步信号生成方法，其特征在于，所述 p 比特为 8 比特，所述 q 比特为 15 比特，所述游程长度限制规则除了所述同步信号以外，对所述代码字进行不归零翻转 NRZI 变换所得的信号的最小游程长度是 3T，并且最大游程长度是 11T、12T、13T、14T 中的某一个。

5. 一种记录装置，将用多个编码表对 p 比特的输入数据字进行调制以获得 q 比特的代码字的记录信号依次记录到记录媒体上，其中， $q > p$ ，所述多个编码表存储对应于各个输入数据字的代码字和表示为了获得即使该代码字之间直接结合生成代码字串也满足规定的游程长度限制规则的下一个代码字、而对下一个输入数据字进行调制所用的编码表的状态信息，所述记录装置包括：

同步信号生成部分，生成重放数据解码用的同步信号，能够与满足所述规定的游程长度限制规则的所述代码字串分离，并且由用于识别一个扇区内的位置的特定码、及游程长度比所述规定的游程长度限制规则中的最大游程长度大 1T 以上的第一比特码型和接续该第一比特码型游程长度比最小游程长度大的第二比特码型组成的同步码型构成，其中，T 是所述代码字的信道比特周期，并且所述同步码型包括接续其后的一部分代码字；

同步帧生成部分，将所述重放数据解码用的同步信号以每个规定的代码字数目插入到满足所述规定的游程长度限制规则输出的代码字串中而生成同步帧；

记录部分，基于所述同步帧将记录信号依次记录到记录媒体上。

6. 如权利要求 5 所述的记录装置，其特征在于，使所述多个编码表中的特定编码表和另一特定的编码表对应于预先设定的规定的输入数据字所存储的各个代码字中为‘1’的数目可进行 DSV 控制，具有如果一方的编码表中的代码字中为偶数个则另一方的编码表中的代码字中就为奇数个这样的奇偶性来进行分配，在对所述规定的输入数据字进行调制时，在用所述特定的编码表调制的代码字获得的 DSV 值的绝对值和用另一特定的编码表调制的代码字获得的 DSV 值的绝对值中，通过选择绝对值小的一方的代码字，来进行 DSV 控制，并且将重放数据解码用的同步信号以每个规定的代码字数目插入到满足所述规定的游程长度限制规则输出的代码字串中而生成同步帧。

7. 一种传输装置，将用多个编码表对 p 比特的输入数据字进行调制以获得 q 比特的代码字的传输信号以无线或有线依次进行传输，其中， $q > p$ ，所述多个编码表存储对应于各个输入数据字的代码字和表示为了获得即使该代码字之间直接结合生成代码字串也满足规定的游程长度限制规则的下一个代码字、而对下一个输入数据字进行调制所用的编码表的状态信息，所述传输装置包括：

同步信号生成部分，生成重放数据解码用的同步信号，能够与满足所述规定的游程长度限制规则的所述代码字串分离，并且由用于识别一个扇区内的位置的特定码、及游程长度比所述规定的游程长度限制规则中的最大游程长度大 $1T$ 以上的第一比特码型和接续该第一比特码型游程长度比最小游程长度大的第二比特码型组成的同步码型构成，其中， T 是所述代码字的信道比特周期，并且所述同步码型包括接续其后的一部分代码字；

同步帧生成部分，将所述重放数据解码用的同步信号以每个规定的代码字数目插入到满足所述规定的游程长度限制规则输出的代码字串中而生成同步帧；

传输部分，基于所述同步帧将传输信号以无线或有线依次进行传输。

8. 如权利要求 7 所述的传输装置，其特征在于，使所述多个编码表中的特定编码表和另一特定的编码表对应于预先设定的规定的输入数据字所存储的各个代码字中为‘1’的数目可进行 DSV 控制，具有如果一方的编码表中的代码字中为偶数个则另一方的编码表中的代码字中就为奇数个这样的奇偶性来进行分配，在对所述规定的输入数据字进行调制时，在用所述特定的编码表调制的代码字获得的 DSV 值的绝对值和用另一特定的编码表调制的代码字获得的 DSV 值的绝对值中，通过选择绝对值小的一方的代码字，来进行 DSV 控制，并且将重放数据解码用的同步信号以每个规定的代码字数目插入到满足所述规定的游程长度限制规则输出的代码字串中而生成同步帧。

同步信号生成方法、记录装置以及 传输装置

5

技术领域

本发明涉及同步信号生成方法、记录装置、传输装置、记录媒体和传输媒体，在用多个编码表将 p 比特的输入数据字变换成 q 比特（其中， $q > p$ ）的代码字，将使该代码字之间直接结合的代码字串记录在光盘或磁盘等记录媒体上并进行重放，或通过传输部来传输代码字串时，随着高密度化而提高了编码率，还将重放数据解码用的同步信号以每个规定的代码字数目插入到代码字串中，从该代码字串中可靠地分离同步信号来高精度地重放数字数据。

15

背景技术

一般来说，被记录在光盘上的比特长度，由于记录重放的光传输特性、根据有关比特生成的物理制约而产生的最小游程长度（最小比特长度或最小游程长度）的限制、根据块重放扩展容易程度而产生的最大游程长度（最大比特长度或最大游程长度）的限制、以及为了保护伺服频带等，必须对记录信号进行调制，使得具有被记录信号的低频分量的抑制特性。

在满足了该限制的现有的调制方式中，最小游程长度（=所谓的最小反转间隔）为 $3T$ （ T =信道比特的周期），最大游程长度（=所谓的最大反转间隔）为 $11T$ ，所以已知 CD（小型盘）所用的 EFM（Eight to Fourteen Modulation: 8-14 调制）方式、或 DVD（数字视频盘）所用的 EFM+ 方式。

首先，在 CD（小型盘）所用的 EFM 调制中，将输入的 8 比特（1 字节）的数字数据变换成满足最小游程长度为 $3T$ 、最大游程长度为 $11T$

的游程长度限制的 14 比特的游程长度限制码（以下记为代码字），并且将在变换过的代码字之间作为用于 DSV（Digital Sum Value: 数字加和值）控制和用于保持游程长度限制规则而附加了 3 比特的连接比特所得的信号作为 EFM 调制信号来生成。

5

此时，在最小游程长度为 3T 时，在代码字中的逻辑值 ‘1’ 和 ‘1’ 之间 ‘0’ 的数最少，包含 $d=2$ 个，另一方面，在最大游程长度为 11T 时，在代码字中的逻辑值 ‘1’ 和 ‘1’ 之间 ‘0’ 的数最多，包含 $d=10$ 个。而且，为了减少被调制信号的直流分量或低频分量，即使将作为用于 DSV 控制和用于保持游程长度限制规则而设置的 3 比特的连接比特连接在 14 比特的代码字之间，EFM 调制信号也满足最小游程长度为 3T、最大游程长度为 11T 的游程长度限制规则 $RLL(d, k) = RLL(2, 10)$ 。

而且，在 CD 中，在该 EFM 调制信号中，记录有附加了同步信号的信号。此时，在这样的 EFM 调制信号的代码字串中，不存在上述最大游程长度 11T 的重复码型、即 11T-11T 组成的重复码型，将该 11T-11T 组成的重复码型作为同步信号。

在用于重现 CD 的 CD 播放机中，从 CD 读取的信号中，通过检测上述的 11T-11T 组成的重复码型，来进行同步信号的提取。

其次，在 DVD（数字通用盘）所用的 EFM+ 方式，是将输入的 8 比特的数字数据变换成 16 比特的代码字，将该代码字之间直接结合而不用连接比特，是满足最小游程长度为 3T、最大游程长度为 11T 的游程长度限制规格 $RLL(2, 10)$ 的 8-16 调制方式。

但是，如上所述，在 CD 播放机中，尽管从 CD 读取的信号中可提取上述 11T-11T 组成的重复码型的同步信号，但在今后比 DVD 密度更高的光盘等记录媒体、以及高密度传输调制信号的传输媒体中，

在读取该信息时受到码间干扰的影响很大。此时，假如采用 CD 所用的同步信号，那么 11T-11T 组成的重复码型的同步信号会变化成 11T-10T、或 10T-11T 那样的码型并被读取。相反，作为 EFM 调制信号的 10T-11T、或 11T-10T 的数据码型会变化成 11T-11T 的重复码型，产生将其误检为同步信号的情况。

伴随着这种情况，在对光盘等记录媒体的高密度记录、或高密度数据传输中，在同步信号的检测中产生差错的频度增加，容易产生同步以外的突发差错。

另一方面，在采用 EFM+方式的 DVD 中，由于将同步信号设定为 14T-4T，所以尽管使上述的问题得到解决，但存在编码率低这样的问题。

因此，希望将 p 比特的输入数据字变换成 q 比特（其中， $q > p$ ）的代码字，将该代码字之间直接结合的代码字串记录重现在光盘或磁盘等记录媒体上，或在通过传输部传输代码字串时，在随着高密度化而提高了编码率之后，以每个规定的代码字数将重放数据解码用的同步信号插入在代码字串中，从该代码字串中可靠地分离同步信号来用于高精度地进行数字数据的重放的同步信号生成方法、记录装置、传输装置、记录媒体及传输媒体。

发明内容

本发明是鉴于上述课题的发明，发明的第一方案是一种同步信号生成方法，在用多个编码表对 p 比特的输入数据字进行调制以获得 q 比特（其中， $q > p$ ）的代码字时，所述多个编码表对应于各个输入数据字，存储代码字和表示为了获得即使与该代码字直接结合也满足规定的游程长度限制规则的下一个代码字而对下一个输入数据字进行调制所用的编码表的状态信息，将重放数据解码用的同步信号以每个规定的代码字数目插入到满足所述规定的游程长度限制规则输出的代码

字串中而生成同步帧，其特征在于：

所述同步信号能够与满足所述规定的游程长度限制规则的所述代码字串分离，并且由用于识别一个扇区内的位置的特定码、游程长度比所述规定的游程长度限制中的最大游程长度大 $1T$ （其中， T 是所述代码字的信道比特周期）以上的第一比特码型和接续该第一比特码型游程长度比最小游程长度大的第二比特码型组成的同步码型构成，并且所述同步码型包括接续其后的一部分代码字。

本发明的第二方案是一种同步信号生成方法，其特征在于，在上述第一方案的同步信号生成方法中，使所述多个编码表中的特定编码表和另一特定的编码表具有奇偶性来进行分配，使得对应于预先设定的规定的输入数据字所存储的各个代码字中的‘1’的数目可进行 DSV 控制，如果一方为偶数个则另一方就为奇数个，在对所述规定的输入数据字进行调制时，在用所述特定的编码表调制的代码字获得的 DSV 值的绝对值和用另一特定的编码表调制的代码字获得的 DSV 值的绝对值中，通过选择绝对值小的一方的代码字，来进行 DSV 控制，并且将重放数据解码用的同步信号以每个规定的代码字数目插入到满足所述规定的游程长度限制规则输出的代码字串中而生成同步帧。

本发明的第三方案的特征在于，在上述第一或第二方案的同步信号生成方法中，对应于所述多个编码表准备多个同步信号表，并且在各同步信号表内设定多个用于生成所述同步信号的同步信号比特码型，并且各同步信号比特码型通过 DSV 控制可选择具有奇偶性的两个比特码型的其中一个，使得如果一方的‘1’的数目为偶数个则另一方就为奇数个。

本发明的第四方案的特征在于，在上述第一~第三中任何一项的同步信号生成方法中，所述 p 比特为 8 比特，所述 q 比特为 15 比特，所述游程长度限制规则除了所述同步信号以外，对所述代码字进行 NRZI（Non Return To Zero Inverted，不归零翻转）变换所得的信号的

最小游程长度是 $3T$ ，并且最大游程长度是 $11T$ 、 $12T$ 、 $13T$ 、 $14T$ 中的某一个。

5 本发明的第五方案是一种记录装置，在用多个编码表对 p 比特的输入数据字进行调制以获得 q 比特（其中， $q > p$ ）的代码字时，所述多个编码表对应于各个输入数据字，存储代码字和表示为了获得即使与该代码字直接结合也满足规定的游程长度限制规则的下一个代码字而对下一个输入数据字进行调制所用的编码表的状态信息，将重放数据解码用的同步信号按每个规定的代码字数目插入到满足所述规定的游程长度限制规则而输出的代码字串中而生成同步帧并依次记录到记录媒体上，其特征在于：

10 所述同步信号能够与满足所述规定的游程长度限制规则的所述代码字串分离，并且由用于识别一个扇区内的位置的特定码、及游程长度比所述规定的游程长度限制中的最大游程长度大 $1T$ （其中， T 是所述代码字的信道比特周期）以上的第一比特码型和接续该第一比特码型游程长度比最小游程长度大的第二比特码型组成的同步码型构成，并且所述同步码型包括接续其后的一部分代码字。

20 本发明的第六方案的特征在于，在上述发明的第五方案的记录装置中，使所述多个编码表中的特定编码表和另一特定的编码表具有奇偶性来进行分配，使得对应于预先设定的规定的输入数据字所存储的各个代码字中的‘1’的数目可进行 DSV 控制，如果一方为偶数个则另一方就为奇数个，在对所述规定的输入数据字进行调制时，在用所述特定的编码表调制的代码字获得的 DSV 值的绝对值和用另一特定的编码表调制的代码字获得的 DSV 值的绝对值中，通过选择绝对值小的一方的代码字，来进行 DSV 控制，并且将重放数据解码用的同步信号以每个规定的代码字数目插入到满足所述规定的游程长度限制规则输出的代码字串中而生成同步帧。

30 本发明的第七方案是一种传输装置，在用多个编码表对 p 比特的

输入数据字进行调制以获得 q 比特（其中， $q > p$ ）的代码字时，所述多个编码表对应于各个输入数据字，存储代码字和表示为了获得即使与该代码字直接结合也满足规定的游程长度限制规则的下一个代码字而对下一个输入数据字进行调制所用的编码表的状态信息，将重放数据解码用的同步信号按每个预定的代码字数目插入到满足所述规定的游程长度限制规则而输出的代码字串中而生成同步帧，并以无线或有线依次进行传输，其特征在于：

所述同步信号能够与满足所述规定的游程长度限制规则的所述代码字串分离，并且由用于识别一个扇区内的位置的特定码、及游程长度比所述规定的游程长度限制中的最大游程长度大 $1T$ （其中， T 是所述代码字的信道比特周期）以上的第一比特码型和接续该第一比特码型游程长度比最小游程长度大的第二比特码型组成的同步码型构成，并且所述同步码型包括接续其后的一部分代码字。

本发明的第八方案的特征在于，在上述第七方案的传输装置中，使所述多个编码表中的特定编码表和另一特定的编码表具有奇偶性来进行分配，使得对应于预先设定的规定的输入数据字所存储的各个代码字中的‘1’的数目可进行 DSV 控制，如果一方为偶数个则另一方就为奇数个，在对所述规定的输入数据字进行调制时，在用所述特定的编码表调制的代码字获得的 DSV 值的绝对值和用另一特定的编码表调制的代码字获得的 DSV 值的绝对值中，通过选择绝对值小的一方的代码字，来进行 DSV 控制，并且将重放数据解码用的同步信号以每个规定的代码字数目插入到满足所述规定的游程长度限制规则输出的代码字串中而生成同步帧。

本发明的第九方案是一种记录媒体，其特征在于，使用上述第一~第四发明的方案中任何一项的同步信号生成方法来进行记录。

本发明的第十方案是一种传输媒体，其特征在于，使用上述第一~第四发明的方案中任何一项的同步信号生成方法来进行传输。

附图说明

图 1 是表示应用本发明的同步信号生成方法、记录装置的一实施形态的盘记录装置的方框图。

5 图 2 是用于说明图 1 所示的 8-15 调制部的方框图。

图 3 是表示编码表的示例图（之 1）。

图 4 是表示编码表的示例图（之 2）。

图 5 是表示编码表的示例图（之 3）。

图 6 是表示编码表的示例图（之 4）。

10 图 7 是表示编码表的示例图（之 5）。

图 8 是表示编码表的示例图（之 6）。

图 9 是表示编码表的示例图（之 7）。

图 10 是表示将图 3~图 9 所示的多个编码表与下一个可获得的状态的编码表划分为 5 种情况的图。

15 图 11 是用于说明相对于输入数据字的多个编码表中的指定编码表和其他指定的编码表之间交换情况的图。

图 12 是表示同步信号表的示例的图。

图 13 是表示同步信号的编码表格式的图。

图 14 是表示一个扇区部分的传输信号格式的图。

20 图 15 是 8-15 调制时的 DSV 控制流程图。

图 16 是表示应用本发明的同步信号生成方法、传输装置的一实施形态的信息传输装置的方框图。

具体实施方式

25 以下，参照图 1 至图 16 按照项目顺序来详细说明本发明的同步信号生成方法、记录装置、传输装置、记录媒体以及传输媒体的一个实施例。

〈同步信号生成方法、记录装置、记录媒体〉

30 图 1 是表示应用本发明的同步信号生成方法、记录装置的一实施

形态的盘记录装置的方框图。

如图 1 所示,应用本发明的同步信号生成方法、记录装置的一实施例的盘记录装置 10 大致由格式化部 11、8-15 调制部 12、以及记录驱动电路 13 构成,是通过将输入到该盘记录装置 10 的与图像或声音等信息有关的数字信号经格式化部 11 由 8-15 调制部 12 进行 8-15 调制,由记录驱动电路 13 将 8-15 调制的信号记录到光盘或磁盘等上,获得本发明的记录媒体 20 的装置。

首先,与图像或声音等信息有关的数字信号是在输入时比特数 $p=8$ 比特的输入数据字 SC_t 的连续信号,并且该输入数据字 SC_t 与一起记录的控制信号等同时地输入到格式化部 11 中,在那里附加了纠错码之后,变换成与记录媒体 20 的记录格式一致的控制格式。然后,从格式化部 11 将作为源码的 8 比特的输入数据字 SC_t 输出到 8-15 调制部 12。

接着,在 8-15 调制部 12 中,将比特数 $p=8$ 比特的输入数据字 SC_t 参照后述的多个编码表变换成比特数 $q=15$ 比特的代码字,同时参照后述的多个同步信号表每隔规定的代码字数(例如 91 个字代码)来插入同步信号,并且在对同步信号和多个代码字组成的代码字串进行 NRZI 变换后进行 DSV (Digital Sum Value 数字加和值)控制,作为记录信号输出到记录驱动电路 13。然后,在供给记录媒体驱动电路 13 的记录信号接受由这里未图示的光调制器的光调制后,经带有物镜的光学系统照射并记录在光盘或磁盘等记录媒体 20 上。此时,通过上述方法获得的记录信号是随着对记录媒体 20 的高密度化而提高了编码率的信号。

这里用图 2 至图 15 更详细地说明作为本发明主要部分的 8-15 调制部 12。

图 2 是说明图 1 所示的 8-15 调制部的方框图。

如图 2 所示，作为本发明主要部分的 8-15 调制部 12 由编码选择
分支有无检测部 121、包括多个编码表 123 的编码表地址运算部 122、
同步帧最终数据检测部 130、包括多个同步信号表 132 的同步信号表
地址运算部 131、NRZI 变换部 133、第一、第二路径存储器 125、127、
第一、第二 DSV 运算存储器 124、126、绝对值比较部 128、以及存
储器控制/记录信号输出部 129 构成。

在说明上述的 8-15 调制部 12 内的各构成部件的操作之前，先说
明编码表地址运算部 122 内包括的多个编码表 123 和在同步信号表地
址运算部 131 内包括的多个同步信号表 132。

（关于编码表）

图 3~图 9 是以其 1~其 7 的顺序来表示编码表的示例的图，图 10
是对于图 3~图 9 所示的多个编码表将后面获得的状态的编码表划分成
5 种情况的图，图 11 是说明对于输入数据字的多个编码表中的指定编
码表和另一指定的编码表之间进行交换情况的图。

如图 3~图 9 所示，在编码表地址运算部 122 内配置的多个编码
表 123 预先准备用于对最初输入的输入数据字设定编码表的选择分支
的初始值的初始表，以及由状态（=State）“0” ~状态“5”组成的 6
个编码表。

上述 6 个编码表的每一个将 8 比特的输入数据字 SCt 以 10 进制
数分配给 ‘0’ ~ ‘255’，并且对分配了 ‘0’ ~ ‘255’ 的各输入数
据字 SCt 变换成用 2 进制数表示的 15 比特的各代码字串，同时各代
码字的右端的数字分别设定表示对下一个输入数据字 SCt 进行调制所
使用的编码表的状态信息（下一状态），以便即使使代码字之间直接
结合生成代码字串，也可获得满足规定的游程长度限制规则的下一个

代码字。更具体地对此加以说明，例如参照图 3 所示的状态“0”的编码表可知，在输入数据字‘0’时，状态信息是“4”，在输入数据字‘1’时，状态信息是“5”，而在输入数据字‘2’时，状态信息是“0”。因此，使用状态“0”的编码表进行输入数据字‘0’的调制（编码）时，对于下一个输入数据字 SCt 来说，使用状态“4”的编码表来进行调制。

设定上述 6 个的各编码表，设定使得在每次输入输入数据字 SCt 时，以满足最小游程长度为 3T、最大游程长度为 11T 的游程长度限制规则 RLL (2, 10) 来变换成 15 比特（1 代码字）的代码字。此时，如在现有技术中说明的那样设定为，在最小游程长度为 3T 时，在 15 比特的代码字中的逻辑值‘1’和‘1’之间‘0’的数目最小包含 $d=2$ 个，而在最大游程长度 11T 时，在 15 比特的代码字中的逻辑值‘1’和‘1’之间‘0’的数目最小包含 $k=10$ 个，满足游程长度限制规则 RLL (d, k) = RLL (2, 10)，并且即使使代码字串之间直接结合，也可以满足游程长度限制规则 RLL (2, 10)。

上述 6 个的各编码表如图 10 所示，根据前面输出的 15 比特的代码字中的 LSB 侧（低比特侧）的零游程长度，下次传递的编码表的获得状态可以区分为情况 0~情况 4 的 5 种情况。

上述 6 个编码表中的指定的编码表和另一指定的编码表的与对应于预先设定的规定的输入数据字 SCt 所存储的各个代码字具有奇偶性，即为了进行 DSV 控制，如果 15 比特中的‘1’的数目在一方的编码表的代码字中为偶数个（或奇数个），则在另一方的编码表的代码字中为奇数个（或偶数个），并且分配代码字，使得在对各个代码字进行 NRZI 变换所得的各信号进行 DSV 控制时两者的 DSV 值的极性为+-相反极性。而且，如后面所述，作为替换代码字的形态，如下设定第一形态~第三形态这 3 个形态，以便可获得与对应于预先设定的规定的输入数据字 SCt 的指定的编码表的代码字，和对应于上述同一

规定的输入数据字 SCt 的另一指定的编码表的代码字之间 DSV 值的绝对值小的代码字（与 DSV 值接近 0 的方向等价）。由此，如后面所述，在与第一形态~第三形态适合的情况下，对规定的输入数据字 SCt 判断为‘有选择分支’，而在除此以外的情况下，对输入数据字 SCt 判断为‘没有选择分支’。

即，在第一形态中，在指定的编码表为状态“0”的编码表，另一指定的编码表为状态“3”的编码表时，尽管对与输入数据字‘0’~‘38’对应的状态“0”和状态“3”的各编码表的各输出代码字进行 NRZI 变换所得的信号的 DSV 值的极性相反（代码字中包含的‘1’的数目的奇偶性不同），但如后述的图 15 的 8-15 调制时的 DSV 控制流程图所示，考虑到解码时的情况，在检测出状态信息“0”时与输入数据字‘0’~‘38’对应的状态“0”的编码表的各输出代码字被设定为可与输入数据字‘0’~‘38’对应的状态“3”的编码表的各输出代码字替换，并且即使进行代码字的替换也可以维持游程长度限制规则，而且可以进行解码。

用图 11（a）、（b）更具体地说明这种情况时，如图 11（a）所示，例如，对输入数据字‘16’用状态“2”的编码表变换为代码字{000000001001001}时，下一个编码表根据状态信息来指定状态“0”的编码表。由此，检测状态信息“0”，在将下次输入的例如输入数据字‘6’用状态“0”的编码表变换成{00000000100100}时，该代码字{00000000100100}中的‘1’的数目是 2 个，为偶数个。

另一方面，如图 11（b）所示，对输入数据字‘16’用状态“2”的编码表变换成代码字{000000001001001}时，下个编码表尽管根据状态信息指定状态“0”的编码表，但由于被设定为可替换为如上所述的状态“3”的编码表，所以在将输入的输入数据字‘6’用状态“3”的编码表变换成代码字{001001000100000}时，该代码字{001001000100000}中的‘1’的数目是 3 个，为奇数个。因此，对于

输入数据字‘6’来说，状态“0”的编码表和状态“3”的编码表相对于‘1’的数目具有奇偶性。

5 然后，对图 11 (a)、(b) 的代码字串进行 NRZI 变换。这里，众所周知，由于 NRZI 变换是进行在比特‘1’中将极性反转、在比特‘0’中不将极性反转的调制，所以可获得图 11 (a)、(b) 所示的各信号。

10 然后，为了进行良好的 DSV 控制，对于图 11 (a)、(b) 的代码字串进行 NRZI 变换所得的各信号，进行 DSV 值比较，选择 DSV 值的绝对值小的一方。众所周知，该 DSV 值是将比特‘1’的值+1，将比特‘0’的值-1，从进行了 NRZI 变换的各信号的开始时刻起求出的累积值，在图 11 (a) 的情况下 DSV 值为+2，在图 11 (b) 的情况下 DSV 值为-10，两者间 DSV 值的极性为相反极性。而且，即使进行
15 替换也可以维持游程长度限制规则，而且可以进行解码。在图 11 (a)、(b) 的例中，由于图 11 (a) 情况的 DSV 值的绝对值小，所以选择它就可以，按照过去的状态，DSV 值一般是变化的值。

20 接着，在第二形态中，在指定的编码表为状态“2”的编码表，另一指定的编码表为状态“4”的编码表时，对于状态“2”和状态“4”的各编码表的输入数据字‘0’~‘11’和‘26’~‘47’来说，与上述同样对‘1’的数目具有奇偶性，这里也如图 15 的 8-15 调制时的 DSV 控制流程图所示，考虑到解码时的情况，在检测出状态信息“2”时与输入数据字‘0’~‘11’和‘26’~‘47’对应的状态“2”的编
25 码表的各输出代码字被设定为可与输入数据字‘0’~‘11’和‘26’~‘47’对应的状态“4”的编码表的各输出代码字替换，并且即使进行代码字的替换也可以维持游程长度限制规则，而且可以进行解码。

30 接着，在第三形态中，是状态“3”的编码表，前面的输出代码字的 LSB 侧的零游程长度为 2~6，并且输入数据字 SCt 在‘156’以

下，即使在下个输出代码字替换为状态“0”的编码表中的输出代码字也不使游程长度限制规则崩溃的范围时，可将状态“3”的编码表的各输出代码字替换为状态“0”的编码表的各输出代码字。

5 以上说明的多个编码表 123 根据上述的编码时的各约束条件，在将比特数 $p=8$ 比特的输入数据字 SC_t 变换成比特数 $q=15$ 比特的代码字时，进行 8-15 调制，使得满足最小游程长度为 $3T$ 、最大游程长度为 $11T$ 的游程长度限制规则 $RLL(d, k) = RLL(2, 10)$ ，但并不限于此，用上述 6 个编码表也可以变更为游程长度限制规则 $RLL(2, 11)$ 、
10 或 $FLL(2, 12)$ 、或 $RLL(2, 13)$ ，这种情况下，在后述的操作流程图（图 15）的步骤 407 中通过改变游程长度限制规则，可以使最小游程长度为 $3T$ 、并且最大游程长度为 $12T$ 、或 $13T$ 、或者除了步骤 403、步骤 405 的条件之外部分地为 $14T$ 。

15 显然，不使用上述 6 个编码表，而按与此相同的技术思想，将 $p=8$ 比特的输入数据字 SC_t 变换为 $q=15$ 比特的代码字时，也可以对编码表内的各代码字和状态信息重新设定，以便满足最小游程长度为 $3T$ 、最大游程长度为 $12T$ 、 $13T$ 或 $14T$ 。于是，通过将最大游程长度设定为比 $11T$ 大的 $12T$ 、 $13T$ 或 $14T$ ，随着最大游程长度增大，可以进一步增加 DSV 控制的机会。
20

（关于同步信号表）

图 12 表示同步信号表的示例，图 13 表示同步信号的编码表的格式的图，图 14 表示 1 扇区的传输信号的格式的图。

25

如图 12 所示，同步信号表地址运算部 131 内包括的多个同步信号表 132 预先准备用于对最初输入的同步信号设定同步信号表的选择分支初始值的初始表，以及与前面说明的编码表 123 的状态信息对应的由状态“0”~状态“5”组成的 6 个同步信号表。

30

上述状态“0”~状态“5”的各同步信号表对应于用于获得同步帧的最终输入数据的下个代码字 SC_t 的状态信息来准备，并且将在各同步信号表内分组成 $SY_0 \sim SY_5$ 组成的 6 种同步信号比特码型。

5 6 种同步信号比特码型 $SY_0 \sim SY_5$ 组成由图示左侧的 1 比特~30 比特构成的同步信号比特码型 SY_{n-1t} （其中， n 为 0~5）、以及图示右侧的 1 比特~30 比特构成的同步信号比特码型 SY_{n-2t} （其中， n 为 0~5）构成的两个同步信号比特码型，具有用于 DSV 控制的‘1’的数目如果在一方的同步信号比特码型 SY_{n-1t} 中为偶数个（或奇数个），
10 则在另一方的同步信号比特码型 SY_{n-2t} 中为奇数个（或偶数个）的奇偶性，并分配比特码型，使得在对各个同步信号比特码型 SY_{n-1t} 、 SY_{n-2t} 进行了 NRZI 变换所得的信号进行 DSV 控制时使两者的 DSV 值的极性为+-相反极性。

15 1 比特~30 比特组成的同步信号比特码型如图 13 放大所示，由 1 比特~13 比特组成的指定码和接续该指定码的 14 比特~30 比特的同步码型的大部分的比特串构成。而且，同步码型由同步信号比特码型中的 14 比特~30 比特和接续连接它的后续代码字中的一部分来构成，并且通过将作为后续代码字的前置比特的最高位比特设定为‘1’，进行调制使得同步信号接续的代码字 SC_t 前置比特为‘1’。此时，作为实施例中代码字的前置比特为‘1’的编码表 123，准备了状态“5”的编码表。
20

25 同步信号比特码型中的指定代码被分配到比特 1~比特 13 中，与作为可识别后述的 1 扇区内的位置的代码一起的是可以进行 DC 控制的代码。

30 上述同步码型是以比 8-15 调制信号中的最大游程长度 $11T$ 大 $2T$ 的 $13T$ 的第一比特码型为核心，在该 $13T$ 的第一比特码型的后方配置固定长度组成的 $4T$ 的第二比特码型的 $13T-4T$ 的排列，即

{100000000000010001}的比特码型，是对所有的同步信号公用的固定码型。此时，在同步码型中的 13T 的第一比特码型的后方使 4T 的第二比特码型为固定长度的原因在于，在将上述的指定码放置在该同步码型的前方时，前方的自由度增大，可充分确保取得指定码的码型的数目。

在上述实施例的同步信号表 132 中，同步信号比特码型中的比特 14~比特 30 和它们后续的代码字的一部分组成的同步码型的最大间隔是以比调制方式的游程长度限制规则的最大游程长度 11T 长 2T 的 13T 的第一比特码型为例表示的，但并不限于此，第一比特码型的最大游程长度也可以比最大游程长度限制大 1T 以上来构成。特别是第一比特码型比最大游程长度长 3T 的情况或长 4T 的情况更有效。

在同步码型中的第一比特码型的后方作为示例示出了 4T 的第二比特码型，但并不限于此，第二比特码型也可以组合 5T 以上的码型。在上述实施例中考虑到调制解调方式的效率用了 13T-4T。

如图 14 所示，上述同步信号比特码型的同步信号在构成输入数据字 SCt 代码字串的例如每 91 个码字中选择同步信号比特码型 SY0~SY5 中的其中一种，将在产生该比特码型的 91 个码字的前头附加的码字作为与 1 个同步帧对应的记录信号来输出。此时，1 个扇区的记录信号格式如图所示，1 个扇区由 13 行组成，在各行中沿列方向分配 4 个同步帧。各同步帧中分配的同步信号是从图 12 所示的同步信号比特码型 SY0~SY5 中选择的信号。例如，第一行的前同步帧中分配的同步信号比特码型是与选择的 SY0 对应的比特码型。在第一行以后、前同步帧中分配的同步信号比特码型对应于行的增加形成如 SY1~SY3 那样循环重复的构造。此时，这样的 SY1~SY3 的各自差异是上述指定码决定的差异。即，各行中存在的 4 个同步信号比特码型的各自指定码内的一个是按照行的增加循环重复的构造。

这里，返回到图 2 来说明 8-15 调制部 12 的工作情况。

5 在该 8-15 调制部 12 中，对同步信号和输入数据字 SCt 进行如上所述的 DSV 控制，决定最终输出的同步信号和输入数据字 SCt 所对应的代码字，但为了清楚容易地说明，首先说明对输入数据字 SCt 的 DSV 控制。

10 在由 8-15 调制部 12 对输入数据字 SCt 进行 DSV 控制的情况下，首先对输入数据字 SCt 选择初始编码表（编码表 123 的选择分支的初始值）。接着，在输入 8 比特的输入数据字 SCt 时，代码字选择分支有无检测部 121 根据本次的输入数据字 SCt 和从编码表地址运算部 122 供给的先行输出代码字（这里选择了初始值）决定的状态信息，检测与本次的输入数据字 SCt 对应的输出代码字是前面说明的第一~第三形态的某一个，有用于 DSV 控制的选择分支，还是第一~第三形态以外的没有选择分支、而由代码字唯一决定的，将检测结果分别输出到
15 编码表地址运算部 122 和绝对值比较部 128。然后，编码表地址运算部 122 根据来自代码字选择分支有无检测部 121 的‘有选择分支’或‘没有选择分支’的检测结果来计算编码表 123 的地址。

20 即，代码字选择分支有无检测部 121 在前面说明的第一形态的情况下，从编码表地址运算部 122 供给的状态信息是状态“0”，而在输入数据字 SCt 为‘0’~‘38’的情况下，输出‘有选择分支’的检测结果。此时，由于编码表地址运算部 122 算出的地址为 2 个，所以编码表 123 通过时分处理输出两种代码字。然后，编码表地址运算部 122 将编码表 123 中与状态“0”的编码表的输入数据字 SCt 对应的输出代码字 OC1t 作为路径 1 的用途而读出，并且将与状态“3”的
25 编码表的输入数据字 SCt 对应的输出代码字 OC2t 作为路径 2 的用途而读出。

30 代码字选择分支有无检测部 121 在前面说明的第二形态的情况

下，从编码表地址运算部 122 供给的状态信息是状态“2”，而在输入数据字 SCt 为‘0’～‘11’或‘26’～‘47’的情况下，也输出‘有选择分支’的检测结果。此时，编码表地址运算部 122 将编码表 123 中与状态“2”的编码表的输入数据字 SCt 对应的输出代码字 OC1t 作为路径 1 的用途读出，并且将与状态“4”的编码表的输入数据字 SCt 对应的输出代码字 OC2t 作为路径 2 的用途读出。

代码字选择分支有无检测部 121 在前面说明的第三形态的情况下，从编码表地址运算部 122 供给的状态信息是状态“3”，前面的输出代码字的 LSB 侧的零游程长度为 2~6，并且输入数据字 SCt 在‘156’以下，即使在下个输出代码字替换为状态“0”的编码表中的输出代码字也处于不使编码规则崩溃的范围内时，也输出‘有选择分支’的检测结果。此时，编码表地址运算部 122 将编码表 123 中与状态“3”的编码表的输入数据字 SCt 对应的输出代码字 OC1t 作为路径 1 的用途读出，并且将与状态“0”的编码表的输入数据字 SCt 对应的输出代码字 OC2t 作为路径 2 的用途读出。

另一方面，代码字选择分支有无检测部 121 在前面说明的第一~第三形态以外的条件下将‘没有选择分支’（唯一的决定）的检测结果输出到编码表地址运算部 122。此时，由于通过编码表地址运算部 122 算出的地址是 1 个，所以编码表地址运算部 122 从编码表地址 122 中仅读出与该地址对应的输出代码字 OC1t。

接着，在 NRZI 变换部 133 中，在适合第一~第三形态的‘有选择分支’的情况下，对输出代码字 OC1t、OC2t 两者分别实施 NRZI 变换。另一方面，在第一~第三形态以外的‘没有选择分支’的情况下，仅对输出代码字 OC1t 实施 NRZI 变换。此时，对从编码表地址运算部 122 输出的各代码字 OC1t、OC2t（‘有选择分支’的情况）或代码字 OC1t（‘没有选择分支’的情况）进行 NRZI 变换时，由于对比当前时刻前一个的代码字（OC1t-1、OC2t-1）进行后述的 DSV

运算所决定的前一个代码字 $OC1t-1$ 、或前一个代码字 $OC2t-1$ 的其中一个被存储在内部的存储器 133A 中，所以参照该存储器 133A 中存储的前一个代码字来进行 NRZI 变换。

5 接着，由 NRZI 变换部 133 进行了 NRZI 变换的各代码字 $OC1t$ 、 $OC2t$ 或代码字 $OC1t$ 如后面所述的操作流程说明的那样，不直接被存储在第一、第二路径存储器 125、127 中，而是先根据第一、第二 DSV 运算存储器 124、126 运算的对从过去到前一个的代码字的 DSV 值的绝对值的比较结果最终地决定了前一个代码字之后，再存储在第一、
10 第二路径存储器 125、127 中。

 这里，说明由第一、第二 DSV 运算存储器 124、126 对从过去到前一个的代码字进行 DSV 值（累积值）运算，在绝对值比较部 128 中比较该 DSV 值的绝对值的情况。在第一路径存储器 125 中，将前
15 一个代码字 $OC1t-1$ 和在前一个代码字 $OC1t-1$ 之前决定的所有代码字在 NRZI 变换的状态下以时间序列顺序存储，在该第一路径存储器 125 中存储的时间序列顺序的代码字被输出到第一 DSV 运算存储器 124。与此相同，在第二路径存储器 127 中，将前一个代码字 $OC2t-1$ 和在前一个代码字 $OC2t-1$ 之前决定的所有代码字在 NRZI 变换的状态下按照时间序列顺序来存储，在该第二路径存储器 127 中存储的时间序列
20 顺序的代码字被输出到第二 DSV 运算存储器 126。第一、第二路径存储器 125、127 在没有任何存储时，看作 0 进行处理，以后再逐渐地存储即可。

25 接着，第一 DSV 运算存储器 124 进行从过去至前一个代码字 $OC1t-1$ 累积的 DSV 值的运算，其结果的 $DSV1t-1$ 输出到绝对值比较部 128。与此同样，第二 DSV 运算存储器 126 进行从过去至前一个代码字 $OC2t-1$ 累积的 DSV 值的运算，其结果的 $DSV2t-1$ 输出到绝对值比较部 128。

30

接着，绝对值比较部 128 比较从第一 DSV 运算存储器 124 输出的前一个代码字 $OC1t-1$ 的 DSV 值的绝对值 $|DSV1t-1|$ 和从第二 DSV 运算存储器 126 输出的前一个代码字 $OC2t-1$ 的 DSV 值的绝对值 $|DSV2t-1|$ 的大小，将其比较结果输出到存储器控制/记录信号输出部 129。

接着，存储器控制/记录信号输出部 129 在从绝对值比较部 128 送出的比较结果是 $|DSV1t-1| < |DSV2t-1|$ 的情况下，将第一路径存储器 125 中所存储的过去的所有输出代码字和之前的代码字 $OC1t-1$ 作为选择的记录信号输出，同时也输出到第二路径存储器 127，对第二存储器 127 进行改写。并且将第二 DSV 运算存储器 126 的存储内容改写为 DSV 值的绝对值小的一方的在第一 DSV 运算存储器 124 中存储的 $DSV1t-1$ 。

相反，存储器控制/记录信号输出部 129 在从绝对值比较部 128 送出的比较结果为 $|DSV1t-1| \geq |DSV2t-1|$ 时，将第二路径存储器 127 中存储的过去的输出代码字和前一个代码字 $OC2t-1$ 作为选择的记录信号输出，同时也输出到第一路径存储器 125，对第一存储器 125 进行改写，并且将第一 DSV 运算存储器 124 的存储内容改写为 DSV 值的绝对值小的一方的第二 DSV 运算存储器 126 中存储的 $DSV2t-1$ 。

因此，绝对值比较部 128 选择 DSV 值的绝对值小的一方的前一个代码字，将对从过去的输出代码字和选择的前一个代码字进行组合所得的代码字串从存储器控制/记录信号输出部 129 输出到记录驱动电路 13（图 1）。

然后，NRZI 变换部 133 在‘有选择分支’时将对输出代码字 $OC1t$ 、 $OC2t$ 进行 NRZI 变换后所得的各信号分别存储在第一、第二路径存储器 125、127 中，另一方面，在‘没有选择分支’时通过将仅对输出代码字 $OC1t$ 进行 NRZI 变换所得的信号存储在第一、第二路径存储

器 125、127 两者中，从而第一、第二路径存储器 125、127 中存储的各信号成为与下一个被编码的输入数据字 SC_{t+1} 对应的对向代码字 OC_{1t+1} 、 OC_{2t+1} 的 DSV 控制时的前一个信号。然后，对于第一、第二路径存储器 125、127 中存储的各信号，如果由第一、第二 DSV 运算存储器 124、126 与上述大致同样地进行 DSV 运算并存储，那么这些信号在下次操作时用于 DSV 值的绝对值的比较。

重复以上的操作直至没有输入数据字 SC_t ，在 NRZI 变换后满足从 $3T$ 至 $11T$ 的游程长度限制规则，并且可以将 DSV 控制的记录信号作为对记录媒体 20 的记录信号输出。

另一方面，输入数据字 SC_t 也被输入到同步帧最终数据检测部 130，在同步帧最终数据检测部 130 中对输入数据字 SC_t 的输入个数（同步帧由 91 个码字构成）进行计数，检测输入数据字 SC_t 是否为同步帧的最终数据，将用于插入同步信号的检测结果输出到同步信号表地址运算部 131。

然后，在检测出输入数据字 SC_t 为同步帧的最终数据、插入同步信号时，同步信号编码表地址运算部 131 根据从编码表地址运算部 122 供给的预先输出代码字（这里是初始表的初始值）所决定的状态信息和表示同步信号表地址运算部 131 中保持的 6 种同步信号比特码型 $SY_0 \sim SY_5$ 的某一个的信息，选择在状态“0”~状态“5”的同步信号表的某一个、并且选择各同步信号表内 6 种同步信号比特码型 $SY_0 \sim SY_5$ 的某一个。这里，对应于选择 $SY_0 \sim SY_5$ 中的某一个种类来计算奇偶性不同的两个同步信号比特码型 SY_{n-1t} 、 SY_{n-2t} （其中， n 为 0~5）的地址，同步信号表 132 将具有相互不同的两个比特码型的同步信号输出到 NRZI 变换部 133。然后，NRZI 变换部 133 对从同步信号表 132 输出的两个同步信号进行 NRZI 变换。

然后，通过与上述的代码字情况时相同的步骤，对由第一、第二

DSV 运算存储器 124、126 运算的此前的代码字进行 DSV 值的绝对值比较，产生 DSV 值的绝对值的比较结果，决定此前的代码字后，将从 NRZI 变换部 133 输出的两个同步信号存储在第一、第二路径存储器 125、127 中。并且，如果对第一、第二路径存储器 125、127 中存储的各同步信号，如果预先用第一、第二 DSV 运算存储器 124、126 进行与上述同样地运算并存储，那么这些信号在下次操作时可用于 DSV 值的绝对值的比较。

此时，在最初插入同步信号时，在第一、第二路径存储器 125、127 中作为未存储此前的代码字处理即可。

而且，在输入了同步信号后，在其后续的输入数据字 SCt 为‘有选择分支’的时刻，通过比较包含同步信号的此前的 DSV 值的绝对值，来决定包含同步信号的此前的 DSV 值的绝对值小的一方的同步信号。然后，同步信号例如按每 91 个的字数据进行插入。

在图 2 所示的 8-15 调制部 12 中，为了暂时存储同步信号和代码字串而设有第一、第二两个路径存储器，但本发明也可以适用于具有更多的路径存储器的情况。

下面，参照图 15 所示的 8-15 调制时的 DSV 控制流程图并用图 2 一起详细说明其操作的具体例。

首先，在步骤 400 中，对同步信号和输入数据字 SCt 选择初始表（同步信号表 132 和编码表 123 的选择分支的初始值）。

接着，在步骤 401 中，同步信号表地址运算部 131 根据从编码表地址运算部 122 供给的预先输出代码字（最初时选择初始值）所决定的状态和表示是否是同步信号表地址运算部 131 中保持的同步信号比特码型 SY0~SY5 的某一个的信息，来选择状态“0”~状态“5”的

同步信号表的某一个，并且是各同步信号表内的 SY0~SY5 的某一个。例如，如果选择状态“n”（其中，n 为 0~5）的同步信号表内的同步信号比特码型 SYn（其中，n 为 0~5），则由于该同步信号比特码型 Syn 当对于‘1’的数目以奇偶性不同来进行 NRZI 变换时，就保持极性不同的两个同步信号 SYn-1t、SYn-2t（其中，n 为 0~5），所以 NRZI 变换部 133 分别对这两个同步信号比特码型 SYn-1t、SYn-2t 进行 NRZI 变换。然后，如上所述，对此前代码字（初次同步信号时没有此前的代码字）进行 DSV 值的绝对值的比较，在决定了此前的代码字后，由 NRZI 变换部 133 进行了 NRZI 变换的两个同步信号 SYn-1t、SYn-2t 被输出到第一、第二路径存储器 125、127，由第一 DSV 运算存储器 124 运算并存储包含有同步信号 SYn-1t 的 DSV 值，而第二 DSV 运算存储器 126 运算并存储包含有同步信号 SYn-2t 的 DSV 值。

接着，在步骤 402 中，接着同步信号来输入 8 比特的输入数据字 SCt。

接着，在步骤 403、步骤 405、步骤 407 中，代码字选择分支有无检测电路 121 根据本次的输入数据字 SCt 和从编码表地址运算部 122 供给的预先输出代码字（最初的情况是选择初始值）所决定的状态来检测本次的输入数据字 SCt 是唯一决定的、还是有选择分支，将该检测结果输出到编码表地址运算部 122 和绝对值比较部 128。

也就是说，在步骤 403 中，如上所述，在图 3~图 9 所示的编码表时，着眼于状态“0”和状态“3”的编码表，根据上述的第一形态，由于对应于输入数据字‘0’~‘38’的状态“0”的编码表的各输出代码字即使与对应于输入数据字‘0’~‘38’的状态“3”的编码表的各输出代码字替换也能够维持编码规则，并且可以解码，所以在该步骤中，代码字选择分支有无检测电路 121 检测出了是否有第一形态选择分支。

然后，编码选择分支有无检测部 121 在从编码表地址运算部 122 供给的状态为状态“0”、输入数据字 SCt 为‘38’以下的适合情况（是时）下，输出‘有选择分支’的检测结果并转到步骤 404。另一方面，在状态“0”、并且输入数据字 SCt 不为‘38’以下的不适合情况（“否”时）下转到步骤 405。

接着，在步骤 404 中，根据步骤 403 的‘有选择分支’的结果，编码表地址运算部 122 从编码表 123 中读出用作路径 1 的与状态“0”的表的输入数据字 SCt 对应的输出代码字 OC1t，并且读出用作路径 2 的与状态“3”的表的输入数据字 SCt 对应的输出代码字 OC2t，由 NRZI 变换部 133 对输出代码字 OC1t、OC2t 分别进行 NRZI 变换。

接着，在步骤 405 中，根据步骤 403 的不适合的结果，着眼于状态“2”和状态“4”的编码表，根据上述的第二形态，由于与输入数据字‘0’～‘11’及‘26’～‘47’对应的状态“2”的编码表的各输出代码字即使替换为与输入数据字‘0’～‘11’及‘26’～‘47’对应的状态“4”的编码表的各输出代码字也可以维持编码规则，并且可以解码，所以在该步骤中代码字选择分支有无检测电路 121 检测出了是否有第二形态选择分支。

然后，编码选择分支有无检测部 121 判断从编码表地址运算部 122 供给的状态是否为状态“2”、输入数据字 SCt 是否在‘11’以下或‘26’～‘47’的范围内，在适合的情况（“是”时）下输出‘有选择分支’的检测结果并转到步骤 406。另一方面，在不适合的情况（“否”时）下转到步骤 407。

接着，在步骤 406 中，根据步骤 405 的‘有选择分支’的结果，编码表地址运算部 122 从编码表 123 中读出与状态“2”的表的输入数据字 SCt 对应的输出代码字 OC1t，并且读出与状态“4”的表的输入数据字 SCt 对应的输出代码字 OC2t，由 NRZI 变换部 133 对输出代

码字 OC1t、OC2t 分别进行 NRZI 变换。

接着，在步骤 407 中，根据步骤 405 的不适合的结果，根据上述的第三形态，作为状态“3”的编码表，在前一个输出代码字的 LSB 侧的零游程长度为 2~6（流程图中示出 2 以上），并且输入数据字 SCt 为‘156’以下（流程图中示出<157），下个输出代码字处于即使替换为状态“0”的编码表中的输出代码字也不使编码规则崩溃的范围内时，由于即使状态“3”的输出代码字和状态“0”的输出代码字进行替换也可以维持编码规则，并且可解码，所以在该步骤中编码选择分支有无检测部 121 检测出了是否有第三形态选择分支。

然后，判断前一个的输出代码字的 LSB 侧的零游程长度是否在 2 以上，以及输入数据字 SCt 是否在‘156’以下，并且下一个输出代码字是否是从状态“3”的编码表中选择的输出代码字，即使替换为状态“0”的编码表中的输出代码字也在不使编码规则崩溃的范围内，在适合的情况下（“是”时），输出‘有选择分支’的检测结果并转到步骤 408。另一方面，在不适合的情况下（“否”时），由于经步骤 403、步骤 405 直至到此都判断为‘没有选择分支’，所以输出该‘没有选择分支’的检测结果并转到步骤 409。在步骤 407 中判断为‘没有选择分支’的情况下，不进行 DSV 值的绝对值的比较或路径的选择，在变为‘有选择分支’之前仅进行对第一、第二路径存储器 125、127 的存储和在第一、第二 DSV 运算存储器 124、126 中的 DSV 计算更新。

此时，在实施例 1 中，在该步骤 407 中设定了满足游程长度限制规则 $RLL(d, k) = RLL(2, 10)$ ，但在该步骤 407 中通过将游程长度限制规则 $RLL(d, k)$ 变更为 $RLL(2, 11)$ 、或者 $RLL(2, 12)$ 、或者 $RLL(2, 13)$ ，使最小游程长度为 3T，并且最大游程长度为 12T、或者 13T、或者是除了步骤 403、步骤 405 的条件以外，可以部分地为 14T。

接着，在步骤 408 中，根据步骤 407 的‘有选择分支’的判断结果，编码表地址运算部 122 从编码表 123 中读出与状态“3”的表的输入数据字 SCt 对应的输出代码字 OC1t，并且读出与状态“0”的表的输入数据字 SCt 对应的输出代码字 OC2t，由 NRZI 变换部 133 对输出代码字 OC1t、OC2t 分别进行 NRZI 变换。

接着，在步骤 409 中，由于根据步骤 407 判断为‘没有选择分支’，所以不进行此前的 DSV 值的绝对值的比较，根据‘没有选择分支’的判断结果，从编码表 123 中仅读出与输入数据字 SCt 对应的输出代码字 OC1t，NRZI 变换部 133 仅对该输出代码字 OC1t 进行 NRZI 变换，并将仅对输出代码字 OC1t 进行了 NRZI 变换所得的信号存储在第一、第二路径存储器 125、127 两者中。这时，路径 1、2 的输出 OC1t、OC2t 的值相同。然后，转到后述的步骤 411，从而进行步骤 414~步骤 415 的处理。

接着，在步骤 410 中，根据步骤 404、或步骤 406、或步骤 408，在 NRZI 变换部 133 对输出代码字 OC1t、OC2t 进行了 NRZI 变换所得的各信号被存储在第一、第二路径存储器 125、127 中的状态下，用绝对值比较部比较第一、第二 DSV 运算存储器 124、126 中存储的对从过去到前一个的代码字进行 DSV 运算所得的各 DSV 值的绝对值 |DSV|。这里，在同步信号后续的代码字的情况下，对步骤 401 运算的各 DSV 值比较绝对值，而在输出代码字 OC1t、OC2t 的情况下，在后述的步骤 414 中对前一个运算并存储的从过去到此前的各 DSV 值比较绝对值。

这里，当来自第一 DSV 运算存储器 124 的 DSV1t-1 的绝对值 |DSV1t-1| 比来自第二 DSV 运算存储器 126 的 DSV2t-1 的绝对值 |DSV2t-1| 小时（“是”时），在步骤 411 中，将第一路径存储器 125 中存储的过去的输出代码字输出到第二路径存储器 127 中并对第二存

5 存储器 127 进行重写,同时用第一 DSV 运算存储器 124 中存储的 $DSV1t-1$ 来对第二 DSV 运算存储器 126 进行重写(使第二 DSV 运算存储器 126 的内容为 $DSV1t-1$)。另一方面,当来自第二 DSV 运算存储器 126 的 $DSV2t-1$ 的绝对值 $|DSV2t-1|$ 小或相等时(“否”时),在步骤 412 中,将第二路径存储器 127 中存储的过去的输出代码字输出到第一路径存储器 125 中并对第一存储器 125 进行重写,同时用第二 DSV 运算存储器 126 中存储的 $DSV2t-1$ 来对第一 DSV 运算存储器 124 进行重写(使第一 DSV 运算存储器 124 的内容为 $DSV2t-1$)。

10 接着,在步骤 411 和步骤 412 后,在步骤 413 中,将由 NRZI 变换部 133 对输出代码字 $OC1t$ 、 $OC2t$ 分别进行 NRZI 变换所得的信号、即与路径 1 的输出代码字 $OC1t$ 对应的信号追加存储到第一路径存储器 125 中,并且将与路径 2 的输出代码字 $OC2t$ 对应的信号追加存储到第二路径存储器 127 中。

15 接着,在步骤 414 中,由第一 DSV 运算存储器 124 运算并存储包含路径 1 的输出代码字 $OC1t$ 的 DSV 值,并且由第二 DSV 运算存储器 126 运算并存储包含路径 2 的输出代码字 $OC2t$ 的 DSV 值。这里,在第一、第二 DSV 运算存储器 124、126 中存储的各 DSV 值在对下个代码字的操作步骤时用于步骤 410 中从过去到此前的 DSV 值的绝对值的比较。

20

25 接着,在步骤 415 中,在同步帧最终数据检测部 130 未检测出输入数据字 SCt 是同步帧的最终数据的情况下(“否”时),返回到步骤 402 重复进行上述步骤 402~步骤 415。另一方面,在检测出输入数据字 SCt 是同步帧的最终数据的情况下(“是”时),在步骤 416 中对同步信号比特码型 $SYn-1$ 、 $SYn-2$ 进行与步骤 410~步骤 414 相同的处理。

30 接着,在步骤 417 中,在有下一个输入数据字 SCt 的情况下(“否”

时)，返回到步骤 401，另一方面，在没有下一个输入数据字 SCt 的情况下（“是”时），在步骤 418 中将第一路径存储器 125（或第二路径存储器 127）中存储的输出代码字的数据串从存储器控制/记录信号输出部 129 输出到记录驱动电路（图 1）。

5

然后，这样编码的 15 比特的记录信号按规定的码字数（例如 91 个字代码）插入同步信号，除了同步信号以外，在满足最小游程长度为 3T（T=信道比特的周期）、最大游程长度为 11T 的游程长度限制规则的同时，还可以提高光盘或磁盘等的记录媒体 20 上的编码率并以高密度进行记录。

10

本发明的同步信号生成方法在使用不同的 DSV 控制方法的情况下也有效。例如，即使用本实施例的编码表，在不进行表内的编码后交换而以每个指定周期插入 DSV 控制比特来进行 DSV 控制的情况下，也可以应用本发明的同步信号生成方法。

15

〈传输装置、传输媒体〉

图 16 是表示应用本发明的同步信号生成方法和传输装置的一实施例的信息传输装置的方框图。

20

如图 16 所示，应用本发明的同步信号生成方法、传输装置的一实施例的信息传输装置 14，大致由格式化部 11、8-15 调制部 12、以及传输部 15 构成，使与输入的图像或声音等信息有关的数字信号经格式化部 11 由 8-15 调制部 12 进行 8-15 调制，8-15 调制过的信号从传输部 15 经无线或有线进行传输。信息传输装置 14 是获得本发明的传输媒体 21 的装置。

25

此时，上述的信息传输装置（传输装置）14 对于前面说明的盘记录装置（记录装置）10 来说，格式化部 11 和 8-15 调制部 12 相同，仅传输部 15 有所不同。这里，即使将由 8-15 调制部 12 进行了 8-15

30

调制的信号从传输部 15 通过空中（无线）或传输电缆（有线）等进行传输时，也可以通过传输部 15 进行适合传输的变换，提高编码率并以少的数据量进行没有错误的传输。

5 根据以上详述的本发明的同步信号生成方法、记录装置、传输装置、记录媒体及传输媒体，在随着高密度而提高了编码率后，在将数字数据收容在多个同步帧组成的扇区中依次传输时，这样的同步帧由同步信号和满足最小游程长度和最大游程长度限制规则的代码字串组成，由于上述同步信号可与满足规定的游程长度限制规则的代码字串分离，并且由用于识别 1 扇区内的位置的特定码、比规定的游程长度限制规格中最大游程长度大 $1T$ （其中， T 是所述码字的信道比特周期）
10 以上的游程长度的第一比特码型及接续在该第一比特码型比最小游程长度大的游程长度的第二比特码型组成比特码型构成，所以即使因码间干扰的影响，使该同步信号及代码字串产生的信号边缘各自边缘移位 $1T$ ，也可以正确地检测两者。
15

 由于上述同步信号的构成在识别扇区内的位置的同时，包含可进行 DC 控制的特定码的结构，所以即使在更高密度的光盘中，无论暂时不能读取扇区开头的同步信号，还是将其他同步信号错误地当成扇区的开头，都可以根据后面存在的同步信号预测正确的扇区开头，所以可良好地进行数字数据的重放。
20

图1

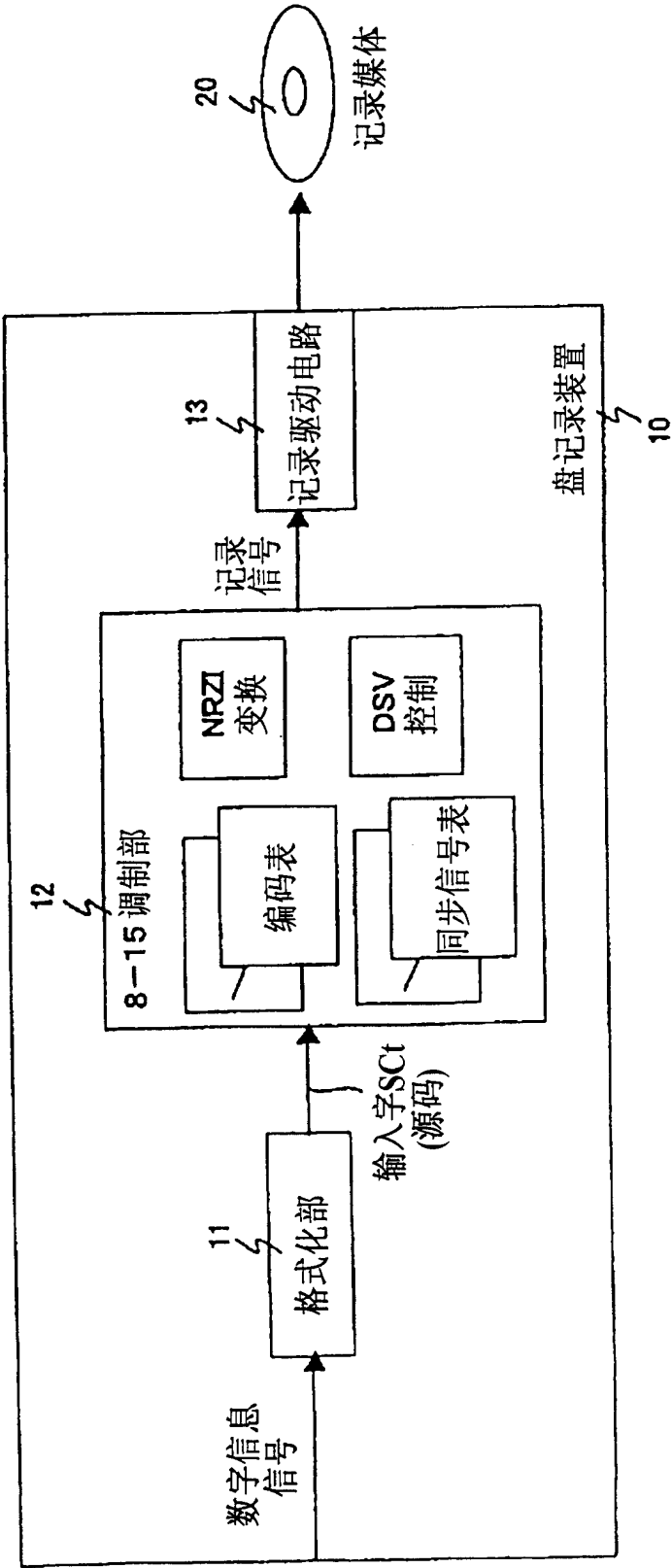


图2

8-15调制部 12

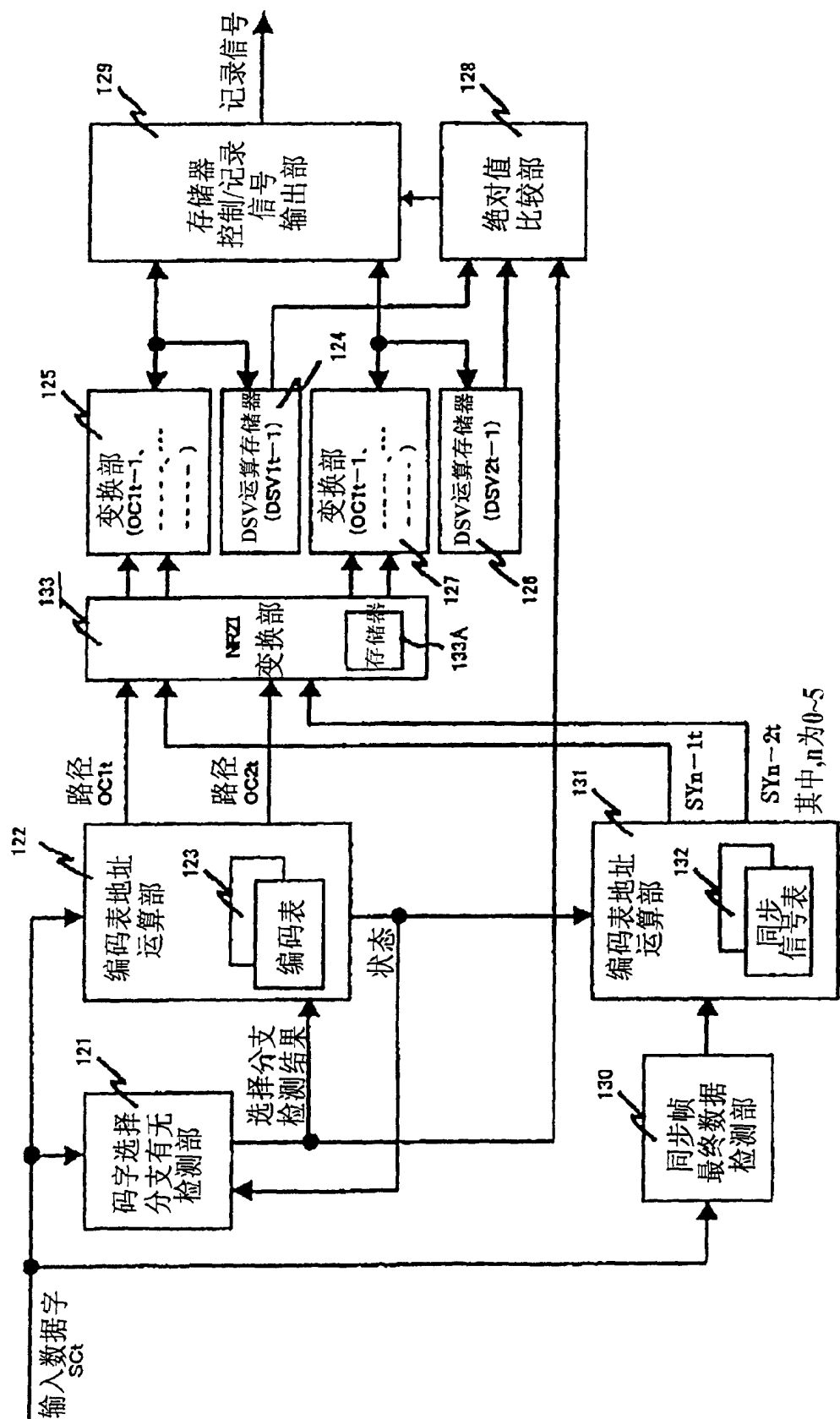


图4

输入字	状态"0"的表	x^1	x^2	x^3	x^4	x^5
39	00000001000001 0	00010001000000 1	00000001000001 0	01000000010001 0	0100100100000001 0	100001001000100 1
40	00000001000001 1	000010001000000 3	00000001000001 1	01000000010001 1	0100100100000001 1	100001001000100 3
41	00000001000010 1	000010001000000 4	00000001000010 1	01000000010001 1	0100100100000001 1	100001001000100 4
42	00000001000010 2	000010001000000 5	00000001000010 2	01000000010001 2	0100100100000001 2	100001001000100 5
43	00000001000010 3	000010001000001 0	00000001000010 3	01000000010001 3	0100100100000001 3	100001001000100 1
44	000001001000100 1	000010001000001 1	000001001000100 1	01000000010001 1	0100100100000001 1	100001001000100 3
45	000001001000100 3	000010001000010 1	000001001000100 3	01000000010001 3	0100100100000001 3	100001001000100 4
46	000001001000100 4	000010001000010 2	000001001000100 4	01000000010001 4	0100100100000001 4	100001001000100 5
47	000001001000100 5	000010001000010 3	000001001000100 5	01000000010001 5	0100100100000001 5	100001001000100 0
48	00000100100010 1	000010001000100 1	00000100100010 1	01000000010001 1	1000000010000001 0	100001001000100 1
49	00000100100010 2	000010001000100 3	00000100100010 2	01000000010001 2	1000000010000001 1	100001000000000 5
50	00000100100010 3	000010001000100 4	00000100100010 3	01000000010001 3	1000000010000001 1	100001000000000 0
51	000001001000100 1	000010001000100 5	000001001000100 1	01000000010001 1	1000000010000001 2	100001000000000 1
52	000001001000100 3	000010001000100 1	000001001000100 3	01000000010001 3	1000000010000001 3	100001000000000 1
53	000001001000100 4	000010001000100 3	000001001000100 4	01000000010001 4	1000000010000001 1	100001000000000 2
54	000001001000100 5	000010001000100 4	000001001000100 5	01000000010001 5	1000000010000001 3	100001000000000 3
55	000001001000100 0	000010001000100 5	000001001000100 0	01000000010001 0	1000000010000001 4	100001000000000 1
56	000001001000100 1	000010001000100 0	000001001000100 1	01000000010001 1	1000000010000001 5	100001000000000 3
57	000000000100010 1	000010001000100 1	000000000100010 1	01000000010001 1	1000000010000001 0	100001000000000 4
58	000000000100010 2	000010010000000 3	000000000100010 2	01000000010001 2	1000000010000001 1	100001000000000 5
59	000000000100010 3	000010010000000 5	000000000100010 3	01000000010001 3	1000000010000001 1	1000010000001000 1
60	000000010000000 3	000010010000000 4	000000010000000 3	0100000001000000 3	1000000010000001 2	1000010000001000 3
61	000000010000000 5	000010010000000 0	000000010000000 5	0100000001000000 5	1000000010000001 3	1000010000001000 4
62	000000010000000 4	000010010000000 1	000000010000000 4	0100000001000000 4	1000000000000000 1	1000010000001000 5
63	000000010000000 0	000010010000000 1	000000010000000 0	0100000001000001 0	1000000000000000 3	1000010000001001 0
64	000000010000000 1	000010010000000 2	000000010000000 1	0100000001000001 1	1000000000000000 4	1000010000001001 1
65	000000010000010 1	000010010000010 3	000000010000010 1	0100000001000010 1	1000000000000000 5	1000010000001000 1
66	000000010000010 2	000010010000010 1	000000010000010 2	0100000001000010 2	1000000000000000 1	1000010000001000 3
67	000000010000010 3	000010010000010 3	000000010000010 3	0100000001000010 3	1000000000000000 3	1000010000001000 4
68	000000010000010 1	000010010000010 4	000000010000010 1	0100000001000100 1	1000000000000000 4	1000010000001000 5
69	000000010000010 3	000010010000010 5	000000010000010 3	0100000001000100 3	1000000000000000 5	1000010000001000 0
70	000000010000010 4	000010010000010 1	000000010000010 4	0100000001000100 4	1000000000000000 3	1000010000001000 1
71	000000010000010 5	000010010000010 3	000000010000010 5	0100000001000100 5	1000000000000000 5	1000010000001001 1
72	000000010000010 1	000010010000010 4	000000010000010 1	0100000001000100 1	1000000000000000 4	1000010000001001 2
73	000000010000010 3	000010010000010 5	000000010000010 3	0100000001000100 3	1000000000000000 5	1000010000001001 3
74	000000010000010 4	000010010000010 1	000000010000010 4	0100000001000100 4	1000000000000000 1	1000010000001000 1
75	000000010000010 5	000010010000010 3	000000010000010 5	0100000001000100 5	1000000000000000 3	1000010000001000 3
76	000000010000000 3	000010010000000 1	000000010000000 3	0100000001000000 3	1000000000000000 5	1000010000001000 4
77	000000010000000 5	000010010000000 3	000000010000000 5	0100000001000000 5	1000000000000000 4	1000010000001000 5
78	000000010000000 4	000010010000000 4	000000010000000 4	0100000001000000 4	1000000000000000 4	1000010000001000 0

图5

输入字	状态"0"的表					2^1	2^2	2^3	2^4	2^5		
79	0000001000000001	0	000010010010000	5	000000100000001	0	000000100000001	0	100000000100100	3	1000100001000001	1
80	0000001000000001	1	000010010010000	0	000000100000001	1	010000010000000	1	100000000100100	4	1000100001000010	1
81	0000001000000010	1	000010010010000	1	000000100000001	1	000000100000001	1	100000000100100	5	1000100001000010	2
82	0000001000000010	2	000010010010010	1	000000100000001	2	010000010000000	2	1000000001000000	1	1000100001000010	3
83	0000001000000010	3	000010010010010	2	000000100000001	3	010000010000000	3	1000000001000000	3	1000100001001000	1
84	0000001000000010	1	000010010010010	3	000000100000001	1	010000010000000	1	1000000001000000	4	1000100001001000	3
85	0000001000000010	3	000100000000000	0	000000100000001	3	010000010000000	3	1000000001000000	5	1000100001001000	4
86	0000001000000010	4	000100000000000	1	000000100000001	4	010000010000000	4	1000000001000000	1	1000100001001000	5
87	0000001000000010	5	000100000000000	1	000000100000001	5	010000010000000	5	1000000001000000	3	1000100001000000	1
88	0000001000000000	1	000100000000000	2	000000100000000	1	010000010000000	1	1000000001000000	4	1000100001000000	3
89	0000001000000000	3	000100000000000	3	000000100000000	3	010000010000000	3	1000000001000000	5	1000100001000000	4
90	0000001000000000	4	000100000000000	0	000000100000000	4	010000010000000	4	1000000001000000	3	1000100001000000	5
91	0000001000000000	5	000100000000000	1	000000100000000	5	010000010000000	5	1000000001000000	1	1000100001000000	0
92	0000001000000000	0	000100000000000	0	000000100000000	0	010000010000000	0	1000000001000000	4	1000100001000000	1
93	0000001000000000	1	000100000000000	1	000000100000000	1	010000010000000	1	1000000001000000	1	1000100001000000	2
94	0000001000000000	1	000100100000010	1	000000100000000	1	010000010000000	1	1000000001000000	1	1000100001000000	3
95	0000001000000000	3	000100100000010	2	000000100000000	3	010000010000000	3	1000000001000000	1	1000100001000000	4
96	0000001000000000	4	000100100000010	3	000000100000000	4	010000010000000	4	1000000001000000	2	1000100001000000	1
97	0000001000000000	5	000100100000010	1	000000100000000	5	010000010000000	5	1000000001000000	3	1000100001000000	3
98	0000001000000000	0	000100100000010	3	000000100000000	0	010000010000000	0	1000000001000000	1	1000100001000000	4
99	0000001000000000	1	000100100000010	4	000000100000000	1	010000010000000	1	1000000001000000	3	1000100001000000	5
100	0000001000000000	1	000100100000010	5	000000100000000	1	010000010000000	1	1000000001000000	4	1000100001000000	1
101	0000001000000000	3	000100100000010	1	000000100000000	3	010000010000000	3	1000000001000000	5	1000100001000000	3
102	0000001000000000	4	000100100000010	3	000000100000000	4	010000010000000	4	1000000001000000	1	1000100001000000	4
103	0000001000000000	5	000100100000010	4	000000100000000	5	010000010000000	5	1000000001000000	3	1000100001000000	5
104	0000001000000000	0	000100100000010	5	000000100000000	0	010000010000000	0	1000000001000000	4	1000100001000000	0
105	0000001000000000	1	000100000000010	1	000000100000000	1	010000010000000	1	1000000001000000	5	1000100001000000	1
106	0000001000000000	0	000100000000010	2	000000100000000	0	010000010000000	0	1000000001000000	1	1000100001000000	3
107	0000001000000000	1	000100000000010	3	000000100000000	1	010000010000000	1	1000000001000000	3	1000100001000000	4
108	0000001000000000	1	000100000000010	1	000000100000000	1	010000010000000	1	1000000001000000	1	1000100001000000	5
109	0000001000000000	2	000100000000010	2	000000100000000	2	010000010000000	2	1000000001000000	3	1000100001000000	4
110	0000001000000000	3	000100000000010	3	000000100000000	3	010000010000000	3	1000000001000000	4	1000100001000000	1
111	0000001000000000	1	000100000000010	1	000000100000000	1	010000010000000	1	1000000001000000	5	1000100001000000	1
112	0000001000000000	3	000100000000010	3	000000100000000	3	010000010000000	3	1000000001000000	0	1000100001000000	2
113	0000001000000000	4	000100000000010	4	000000100000000	4	010000010000000	4	1000000001000000	1	1000100001000000	3
114	0000001000000000	5	000100000000010	5	000000100000000	5	010000010000000	5	1000000001000000	1	1000100001000000	4
115	0000001000000000	1	000100000000010	1	000000100000000	1	010000010000000	1	1000000001000000	3	1000100001000000	5
116	0000001000000000	3	000100000000010	3	000000100000000	3	010000010000000	3	1000000001000000	4	1000100001000000	1
117	0000001000000000	4	000100000000010	4	000000100000000	4	010000010000000	4	1000000001000000	5	1000100001000000	3
118	0000001000000000	5	000100000000010	5	000000100000000	5	010000010000000	5	1000000001000000	0	1000100001000000	4

图6

输入字	状态"0"的表	x_1^*	x_2^*	x_3^*	x_4^*	x_5^*
119	00000100001001	0	000100001001000	1	000001000001001	0
120	000001000001001	1	000100001001000	3	000001000001001	1
121	000001000010000	1	000100001001000	4	000001000010000	1
122	000001000010000	3	000100001001000	5	000001000010000	3
123	000001000010000	4	000100001001001	0	000001000010000	4
124	000001000010000	5	000100001001001	1	000001000010000	5
125	000001000010001	0	000100001000000	3	000001000010001	0
126	000001000010001	1	000100001000000	5	000001000010001	1
127	000001000010010	1	000100001000000	4	000001000010010	1
128	000001000010010	2	000100001000000	1	000001000010010	2
129	000001000010010	3	000100001000000	2	000001000010010	3
130	000001000010000	1	000100001000000	3	000001000010000	1
131	000001000010000	3	000100001000000	1	000001000010000	3
132	000001000010000	4	000100001000000	3	000001000010000	4
133	000001000010000	5	000100001000000	4	000001000010000	5
134	000001000010000	0	000100001000000	5	000001000010000	0
135	000001000010000	1	000100001000000	1	000001000010000	1
136	000001000010010	1	000100001000000	3	000001000010010	1
137	000001000010010	2	000100001000000	4	000001000010010	2
138	000001000010010	3	000100001000000	5	000001000010010	3
139	000001000000001	0	000100001000000	0	000001000010000	0
140	000001000000001	1	000100001000000	1	000001000010000	1
141	000000001000100	1	000100100000000	1	000000001000100	1
142	000000001000100	3	000100100000000	3	000000001000100	3
143	000000001000100	4	000100100000000	4	000000001000100	4
144	000000001000100	5	000100100000000	5	000000001000100	5
145	000000001001000	1	000100100000000	1	000000001001000	1
146	000000001001000	3	000100100000000	3	000000001001000	3
147	000000001001000	4	000100100000000	4	000000001001000	4
148	000000001001000	5	000100100000000	5	000000001001000	5
149	000000001001000	1	000100000000000	0	000000001001000	1
150	000000001001000	3	000100000000000	1	000000001001000	3
151	000000001001000	4	000100000000000	1	000000001001000	4
152	000000001001000	5	000100000000000	2	000000001001000	5
153	000000000000000	4	000100000000000	3	000000000000000	4
154	000000000000000	5	000100000000000	1	000000000000000	5
155	000000000000000	1	000100000000000	3	000000000000000	1
156	000000000000000	3	000100000000000	4	000000000000000	3
157	000000000000000	1	000100000000000	5	000000000000000	1
158	000000000000000	3	000100000000000	0	000000000000000	3

图7

输入字	状态"0"的表											
	x_1^*	x_2^*	x_3^*	x_4^*	x_5^*							
159	000100010010000	4	000100010010001	1	010000000010000	4	0100010000000010	1	010000000010000	4	100100000100000	4
160	000000000010000	5	000100010010010	1	010000000010000	5	0100010000000010	2	010000000010000	5	100100000100000	5
161	000000000010001	0	000100010010010	2	010001000000100	0	0100010000000010	3	010001000000100	0	100100000100001	0
162	000000000010001	1	000100010010010	3	010001000000100	1	0100010000000010	1	010001000000100	1	100100000100001	1
163	000000000010010	1	000010000010000	1	010000000010000	1	0100010000000010	3	010000000010000	1	100100000100010	1
164	000000000010010	2	000010000010000	3	010000000010000	3	0100010000000010	4	010000000010000	3	100100000100010	2
165	000000000010010	3	000010000010000	4	010000000010000	4	0100010000000010	5	010000000010000	4	100100000100010	3
166	001000000100100	1	000010000001000	5	010000000010000	5	001000000100100	1	010000000010000	5	100100000100100	1
167	001000000100100	3	000100000001000	1	010001000000100	0	001000000100100	3	010001000000100	0	100100000100100	3
168	001000000100100	4	000100000001000	3	010001000000100	1	001000000100100	4	010001000000100	1	100100000100100	4
169	001000000100100	5	000100000001000	4	010001000000100	1	001000000100100	5	010001000000100	1	100100000100100	5
170	001000000100101	0	000100000001000	5	010001000000100	2	001000000100101	0	010001000000100	2	100100000100000	1
171	001000000100101	1	000100000100001	0	010001000000100	3	001000000100101	1	010001000000100	3	100100000100000	3
172	001000010000001	0	000100000100001	1	010000000010000	1	001000000100000	0	010000000010000	1	100100000100000	4
173	001000010000001	1	000100000100001	0	010000000010000	3	001000000100000	1	010000000010000	3	100100000100000	5
174	001000010000010	1	000100000100001	1	010000000010000	4	001000000100000	1	010000000010000	4	100100000100001	0
175	001000010000010	2	000100000100010	1	010000000010000	5	001000000100000	2	010000000010000	5	100100000100001	1
176	001000010000010	3	000100000100010	2	010001000000100	0	001000000100000	3	010001000000100	0	100100000100001	1
177	001000010000100	1	000100000100010	3	010001000000100	1	001000000100000	1	010001000000100	1	100100000100001	2
178	001000010000100	3	000100000001000	1	010001000000100	1	001000000100000	3	010001000000100	1	100100000100001	3
179	001000010000100	4	000100000001000	3	010001000000100	2	001000000100000	4	010001000000100	2	100100000100001	1
180	001000010000100	5	000100000001000	4	010001000000100	3	001000000100000	5	010001000000100	3	100100000100001	3
181	001000010000100	1	000100000001000	5	010001000000100	1	001000000100000	1	010001000000100	1	100100000100001	4
182	001000010000100	3	000100000100000	1	010001000000100	3	001000000100000	3	010001000000100	3	100100000100001	5
183	001000010000100	4	000100000100000	3	010001000000100	4	001000000100000	4	010001000000100	4	100100000100001	1
184	001000010000100	5	000100000100000	4	010001000000100	5	001000000100000	5	010001000000100	5	100100000100001	3
185	001000010000101	0	000100000100000	5	010001000000100	1	001000000100000	0	010001000000100	1	100100000100001	4
186	001000010000101	1	000100000000001	0	010001000000000	3	001000000100000	1	010001000000000	3	100100000100000	5
187	001000010000100	1	000100010000001	1	010001000000000	4	001000000100000	1	010001000000000	4	100100000100001	0
188	001000010000100	3	000100000001000	1	010001000000000	5	001000000100000	3	010001000000000	5	100100000100001	1
189	001000010000100	4	000100000001000	3	010001000000000	0	001000000100000	4	010001000000000	0	100100000100000	3
190	001000010000100	5	000100000001000	4	010001000000000	1	001000000100000	5	010001000000000	1	100100000100000	5
191	001000010000100	0	000100000001000	5	010001000000000	1	001000000100000	0	010001000000000	1	100100000100000	4
192	001000010000101	1	000100000001001	0	010001000000000	2	001000000100000	1	010001000000000	2	100100000100000	0
193	001000010000101	1	000100000000001	1	010001000000000	3	001000000100000	1	010001000000000	3	100100000100000	1
194	001000010000101	2	001000000001000	1	010001000000000	4	001000000100000	2	010001000000000	4	100100000100000	1
195	001000010000101	3	001000000001000	2	010001000000000	3	001000000100000	3	010001000000000	3	100100000100000	2
196	001000100000000	3	001000000001000	3	010001000000000	4	001000000100000	4	010001000000000	4	100100000100000	3
197	001000100000000	5	000100000000000	3	010001000000000	5	001000000100000	5	010001000000000	5	100100000100000	1
198	001000100000000	4	000100000000000	5	010001000000000	1	001000000100000	4	010001000000000	1	100100000100000	3

图 8

输入字	状态"0"的表					"1"	"2"	"3"	"4"	"5"		
199	0010001000000001	0	0001001000000000	4	0100010010010001	3	0010001000000001	0	0100010010010000	3	1001000100000100	4
200	0010001000000001	1	0001001000100001	0	0100010010010000	4	0010001000000001	1	0100010010010000	4	1001000100000100	5
201	0010001000000010	1	0001001000100001	1	0100010010010000	5	0010001000000001	0	0100010010010000	5	10010001000001000	1
202	0010001000000010	2	0010000000000010	1	0100000001000001	0	0010001000000001	2	0100000001000001	0	10010001000001000	3
203	0010001000000010	3	0010000000000010	2	0100000001000001	1	0010001000000001	3	0100000001000001	1	10010001000001000	4
204	0010001000000100	1	0010000000000010	3	0100100000000001	0	0010001000000001	1	0100100000000001	0	10010001000001000	5
205	0010001000000100	3	0001000001000000	1	0100100000000001	1	0010001000000001	3	0100100000000001	1	10010001000001001	0
206	0010001000000100	4	0001000001000000	3	0100100000000001	1	0010001000000001	4	0100100000000001	1	10010001000001001	1
207	0010001000000100	5	0001000001000000	4	0100100000000001	2	0010001000000001	5	0100100000000001	2	10010001000001000	1
208	0010001000000100	1	0001000001000000	5	0100100000000001	3	0010001000000001	1	0100100000000001	3	10010001000001000	3
209	0010001000000100	3	0010000000000010	1	0100100000000001	1	0010001000000001	3	0100100000000001	1	10010001000001000	4
210	0010001000000100	4	0010000000000010	3	0100100000000001	3	0010001000000001	4	0100100000000001	3	10010001000001000	5
211	0010001000000100	5	0010000000000010	4	0100100000000001	4	0010001000000001	5	0100100000000001	4	10010001000001000	0
212	0010001000000100	0	0010000000000010	5	0100100000000001	5	0010001000000001	0	0100100000000001	5	10010001000001000	1
213	0010001000000100	1	0010000000000001	0	0100100000000001	1	0010001000000001	1	0100100000000001	0	10010001000001001	1
214	0010001000000100	1	0010000000000001	1	0100100000000001	3	0010001000000001	1	0100100000000001	1	10010001000001001	1
215	0010001000000100	3	0010000000000001	1	0100100000000001	4	0010001000000001	3	0100100000000001	3	10010001000001001	2
216	0010001000000100	4	0010000000000001	3	0100100000000001	5	0010001000000001	4	0100100000000001	4	10010001000001001	3
217	0010001000000100	5	0010000000000001	4	0100100000000001	0	0010001000000001	5	0100100000000001	5	10010001000001000	3
218	0010001000000100	0	0010000000000001	5	0100100000000001	1	0010001000000001	0	0100100000000001	0	10010001000001000	5
219	0010001000000100	1	0010000000000001	0	0100100000000001	1	0010001000000001	1	0100100000000001	1	10010001000001000	4
220	0010001000000100	1	0010000000000001	1	0100100000000001	3	0010001000000001	1	0100100000000001	1	10010001000001000	0
221	0010001000000100	2	0010000000000001	1	0100100000000001	4	0010001000000001	1	0100100000000001	3	10010001000001000	1
222	0010001000000100	3	0010000000000001	2	0100100000000001	5	0010001000000001	2	0100100000000001	4	10010001000001000	1
223	0010001000000100	1	0010000000000001	3	0100100000000001	0	0010001000000001	3	0100100000000001	5	10010001000001000	2
224	0010001000000100	3	0001001001001001	1	0100100000000001	1	0010001000000001	4	0100100000000001	0	10010001000001000	3
225	0010001000000100	4	0001001001001001	3	0100100000000001	2	0010001000000001	5	0100100000000001	1	10010001000001000	1
226	0010001000000100	5	0001001001001001	4	0100100000000001	3	0010001000000001	0	0100100000000001	2	10010001000001000	3
227	0010001000000100	0	0001001001001001	5	0100100000000001	4	0010001000000001	1	0100100000000001	3	10010001000001000	4
228	0010001000000100	1	0010000000000001	0	0100100000000001	1	0010001000000001	2	0100100000000001	4	10010001000001000	5
229	0010001000000100	1	0010000000000001	1	0100100000000001	3	0010001000000001	3	0100100000000001	5	10010001000001000	1
230	0010001000000100	2	0010000000000001	1	0100100000000001	4	0010001000000001	4	0100100000000001	0	10010001000001000	3
231	0010001000000100	3	0010000000000001	2	0100100000000001	5	0010001000000001	5	0100100000000001	1	10010001000001000	4
232	0010001000000100	1	0010000000000001	3	0100100000000001	0	0010001000000001	0	0100100000000001	2	10010001000001000	5
233	0010001000000100	3	0010000000000001	1	0100100000000001	1	0010001000000001	3	0100100000000001	3	10010001000001000	0
234	0010001000000100	4	0010000000000001	3	0100100000000001	1	0010001000000001	4	0100100000000001	4	10010001000001001	1
235	0010001000000100	5	0010000000000001	4	0100100000000001	2	0010001000000001	5	0100100000000001	5	10010001000001000	1
236	0010001000000001	0	0010000000000001	5	0100100000000001	3	0010001000000001	0	0100100000000001	0	10010001000001000	3
237	0010001000000001	1	0001001001000001	0	0100100000000001	1	0010001000000001	1	0100100000000001	1	10010001000001000	4
238	0010001000000001	1	0001001001000001	1	0100100000000001	3	0010001000000001	1	0100100000000001	3	10010001000001000	5
239	0010001000000001	1	0001001001000001	1	0100100000000001	4	0010001000000001	1	0100100000000001	4	10010001000001000	0

图9

输入字	状态"0"的表					*1	*2	*3	*4	*5		
239	001001000000010	2	000100100100010	1	010010000100100	4	001001000000010	2	010010000100100	4	100100100010001	1
240	001001000000010	3	000100100100010	2	010010000100100	5	001001000000010	3	010010000100100	5	100100100010010	1
241	001001000000010	1	000100100100010	3	010010000100000	1	001001000000010	1	010010000100000	1	100100100010010	2
242	001001000000010	3	001000000010000	1	010010000100000	3	001001000000010	3	010010000100000	3	100100100010010	3
243	001001000000010	4	001000000010000	3	010010000100000	4	001001000000010	4	010010000100000	4	100100100100000	1
244	001001000000010	5	001000000010000	4	010010000100000	5	001001000000010	5	010010000100000	5	100100100100000	3
245	001001000000010	1	001000000010000	5	010010000100000	0	001001000000010	1	010010000100000	0	100100100100000	4
246	001001000000010	3	001000000010000	1	010010000100000	1	001001000000010	3	010010000100000	1	100100100100000	5
247	001001000000010	4	001000000010000	3	010010000100000	1	001001000000010	4	010010000100000	1	100100100100000	0
248	001001000000010	5	001000000010000	4	010010000100000	2	001001000000010	5	010010000100000	2	100100100100000	1
249	001001000000010	0	001000000010000	5	010010000100000	3	001001000000010	0	010010000100000	3	100100100100000	1
250	001001000000010	1	001000000010000	0	010010000100000	0	001001000000010	1	010010000100000	0	100100100100000	2
251	001001000000010	1	001000000010000	1	010010000100000	1	001001000000010	1	010010000100000	1	100100100100000	3
252	001001000000010	3	001000000010000	1	010010000000000	3	001001000000010	3	010010000100000	3	100100100100000	1
253	001001000000010	4	001000000010000	3	010010000000000	5	001001000000010	4	010010000100000	5	100100100100000	3
254	001001000000010	5	001000000010000	4	010010000000000	4	001001000000010	5	010010000100000	4	100100100100000	4
255	000010000000000	5	001000001000000	5	010010000000000	5	*1	*2	*3	*4	*5	

*1 前一个数据字的零游程长度为6以下时
7以上时
5
5

*2 前一个数据字的零游程长度为7或8以外
7或8时
5
5

图10

情况	LSB侧的零游程长度	下一个可获得的状态
0	0	0, 1
1	1	1, 2, 3
2	2~6	1, 3, 4, 5
3	7, 8	3, 4, 5
4	9, 10	4, 5

图11

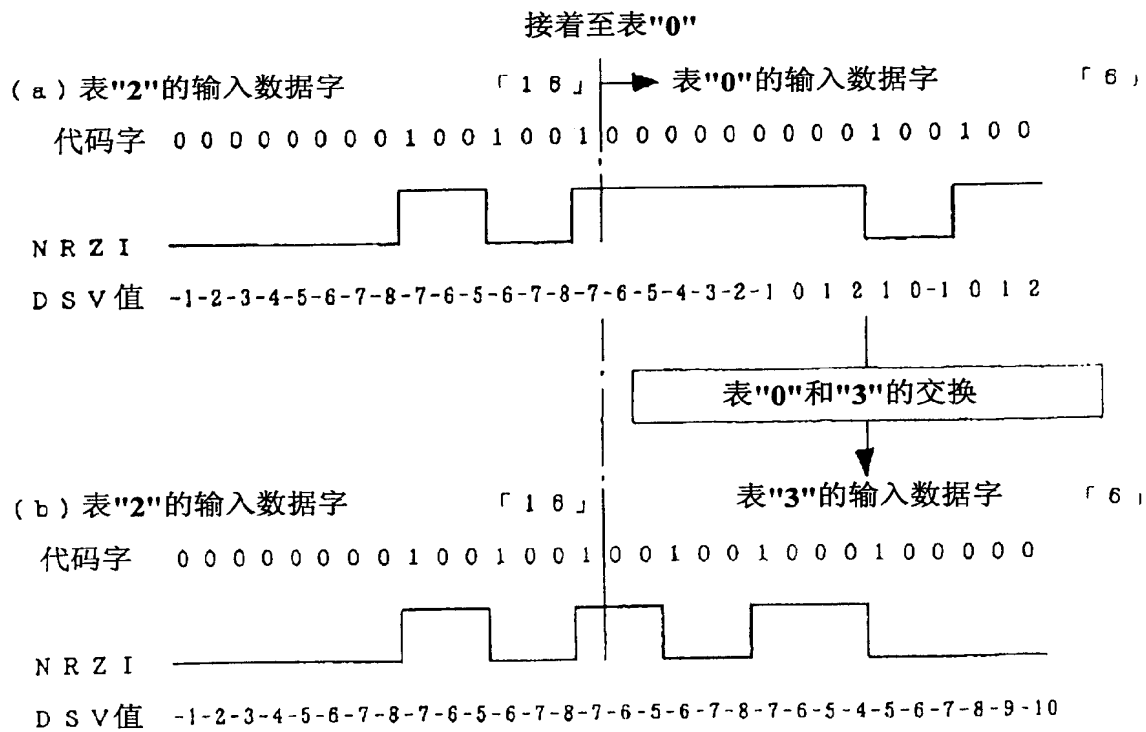


图12

同步信号表 1 3 2

奇偶性
 $\swarrow$ $\searrow$
 SY_n-1t 侧 SY_n-2t 侧
 $\downarrow$ $\downarrow$
 (其中,n为0~5)

状态 = 0												
	1	比特	13	14	比特	30	1	比特	13	14	比特	30
SY0	000000	1000100		10000000000000	1000		00000000	100000		10000000000000	1000	
SY1	000000	100100		10000000000000	1000		00000001	100000		10000000000000	1000	
SY2	000000	1001000		10000000000000	1000		00000010	100000		10000000000000	1000	
SY3	000001	10010000		10000000000000	1000		00000100	100000		10000000000000	1000	
SY4	000001	10000100		10000000000000	1000		00100001	100100		10000000000000	1000	
SY5	000001	10001000		10000000000000	1000		00100010	1000100		10000000000000	1000	
状态 = 1												
	1	比特	13	14	比特	30	1	比特	13	14	比特	30
SY0	000010	00000100		10000000000000	1000		00001000	000000		10000000000000	1000	
SY1	000010	00000100		10000000000000	1000		00001001	000100		10000000000000	1000	
SY2	000010	00010000		10000000000000	1000		00010000	000000		10000000000000	1000	
SY3	000010	00010000		10000000000000	1000		00010001	000100		10000000000000	1000	
SY4	000100	00000100		10000000000000	1000		00010010	000100		10000000000000	1000	
SY5	000100	00000100		10000000000000	1000		00010010	000100		10000000000000	1000	
状态 = 2												
	1	比特	13	14	比特	30	1	比特	13	14	比特	30
SY0	000000	1000100		10000000000000	1000		00000001	100000		10000000000000	1000	
SY1	000000	100100		10000000000000	1000		00000001	100000		10000000000000	1000	
SY2	000000	1001000		10000000000000	1000		00000010	100000		10000000000000	1000	
SY3	000001	10010000		10000000000000	1000		01000100	0000100		10000000000000	1000	
SY4	000001	10000100		10000000000000	1000		01000100	0000100		10000000000000	1000	
SY5	000001	10001000		10000000000000	1000		01000100	0000100		10000000000000	1000	
状态 = 3												
	1	比特	13	14	比特	30	1	比特	13	14	比特	30
SY0	010000	00000100		10000000000000	1000		00100100	0000100		10000000000000	1000	
SY1	010000	00000100		10000000000000	1000		00100100	0000100		10000000000000	1000	
SY2	010000	00010000		10000000000000	1000		00100100	0000100		10000000000000	1000	
SY3	010000	00010000		10000000000000	1000		01000010	0000100		10000000000000	1000	
SY4	010000	10000000		10000000000000	1000		01000010	0000100		10000000000000	1000	
SY5	010001	10000000		10000000000000	1000		01000010	0000100		10000000000000	1000	
状态 = 4												
	1	比特	13	14	比特	30	1	比特	13	14	比特	30
SY0	100000	00000100		10000000000000	1000		01001001	000000		10000000000000	1000	
SY1	100000	00000100		10000000000000	1000		10000010	0000100		10000000000000	1000	
SY2	100000	00010000		10000000000000	1000		10000010	0000100		10000000000000	1000	
SY3	100000	00010000		10000000000000	1000		01000100	0000100		10000000000000	1000	
SY4	100000	10000000		10000000000000	1000		01000100	0000100		10000000000000	1000	
SY5	100001	10000000		10000000000000	1000		01000100	0000100		10000000000000	1000	
状态 = 5												
	1	比特	13	14	比特	30	1	比特	13	14	比特	30
SY0	100010	00000000		10000000000000	1000		10000100	0000100		10000000000000	1000	
SY1	100010	000100100		10000000000000	1000		10000100	0000100		10000000000000	1000	
SY2	100100	00000000		10000000000000	1000		10000100	0000100		10000000000000	1000	
SY3	100100	000100100		10000000000000	1000		10001000	0000100		10000000000000	1000	
SY4	100100	000100100		10000000000000	1000		10001000	0000100		10000000000000	1000	
SY5	100100	000100100		10000000000000	1000		10001000	0000100		10000000000000	1000	

* 同步信号后续的代码的前置比特必须为"1"

图13

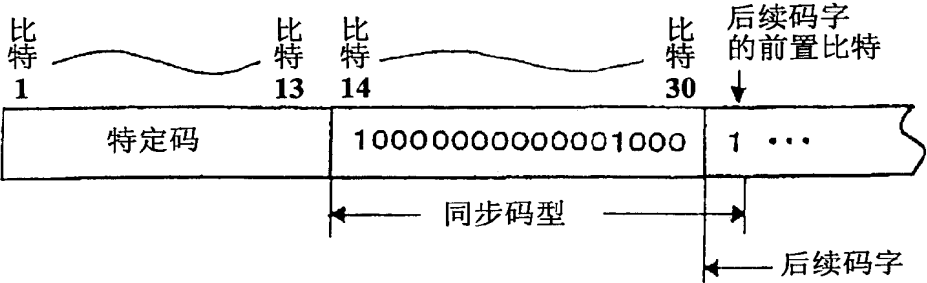


图14

列方向

	30	1365	30	1365	30	1365	30	1365
13行	SY 0		SY 5		SY 5		SY 5	
	SY 1		SY 1		SY 1		SY 1	
	SY 2		SY 2		SY 2		SY 2	
	SY 3		SY 3		SY 3		SY 3	
	SY 1		SY 2		SY 2		SY 1	
	SY 2		SY 3		SY 3		SY 2	
	SY 3		SY 1		SY 1		SY 3	
	SY 1		SY 4		SY 4		SY 1	
	SY 2		SY 1		SY 1		SY 2	
	SY 3		SY 4		SY 4		SY 3	
	SY 1		SY 3		SY 3		SY 1	
	SY 2		SY 4		SY 4		SY 2	
	SY 3		SY 2		SY 2		SY 3	
	SB		SB		SB		SB	

1个扇区的格式

8-15调制时的DSV控制流程图

图15

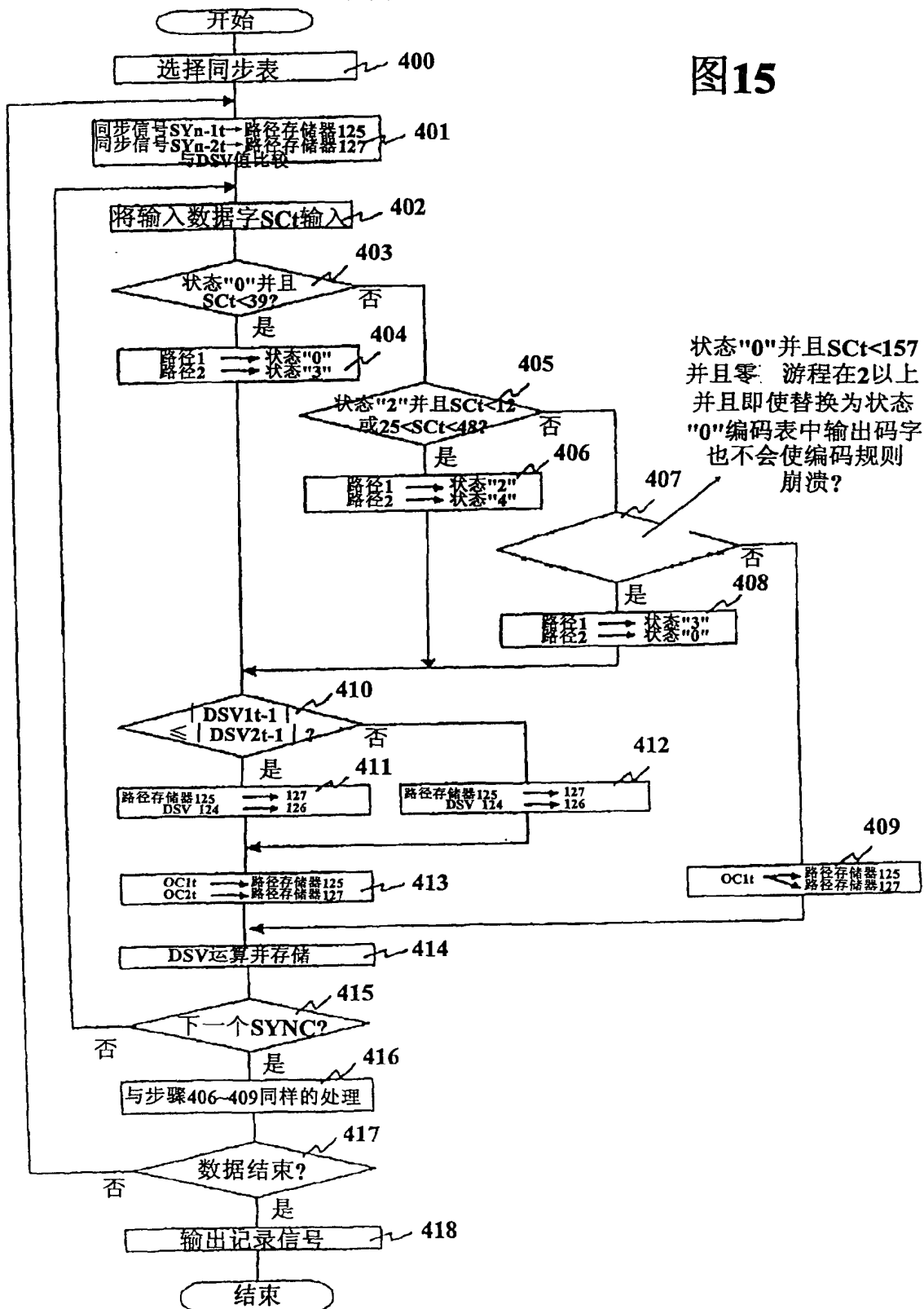


图16

