[19] 中华人民共和国国家知识产权局

[51] Int. Cl.
H03M 7/46 (2006.01)
G11B 20/10 (2006.01)

[12] 发明专利说明书

专利号 ZL 200410008433.8

[45] 授权公告日 2007 年 6 月 6 日

[11] 授权公告号 CN 1320768C

[22] 申请日 2004.3.10
[21] 申请号 200410008433.8
[30] 优先权
[32] 2003.3.11 [33] JP [31] 064778/2003
[73] 专利权人 佳能株式会社
地址 日本东京都
[72] 发明人 梅田嘉伸
[56] 参考文献
US 6195026 B1 2001.2.27
CN 1230845 A 1999.10.6
审查员 韩燕_2

[74] 专利代理机构 北京市金杜律师事务所
代理人 季向冈

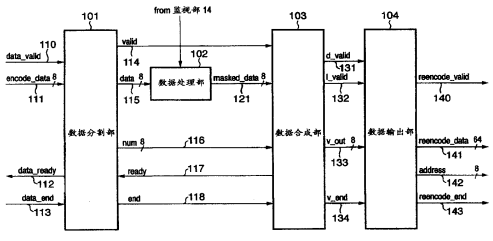
权利要求书 4 页 说明书 36 页 附图 16 页

[54] 发明名称

编码方法和编码装置

[57] 摘要

本发明提供编码方法和编码装置、计算机程序以及计算机可读取存储介质。 可将如压缩位编码数据那样的，以表示相同数据的连续个数的连续数代码部和表示上述数据的数据部，以及表示不同数据流的连续数的连续数代码部和表示上述不同数据流的数据部的数据形式所表达的编码数据，再编码成相同数据形式，使压缩率得以提高，而不用对其进行解码处理。 因此，如果输入在压缩位编码处理被编码的数据，在数据分割部将表示连续的数据数的信息和数据部进行分离，并将其各自作为 num、data 进行输出。 数据处理部根据监视部的指示将预定的位进行屏蔽，并将其结果输出到数据结合部。 数据结合部和数据输出部根据被屏蔽的数据部和 num 数据重新构建压缩位形式的数据，并输出。



1. 一种编码方法，依次输入第 1 类型的编码数据或者第 2 类型的编码数据，再编码成上述第 1 类型或第 2 类型的编码数据形式的编码数据，其中，上述第 1 类型的编码数据用表示相同数据的连续个数的连续数代码部和表示上述数据的数据部所表达，上述第 2 类型的编码数据用表示不同数据的连续个数的连续代码部和上述不同数据的数据部所表达，其特征在于，包括：

判定步骤，根据所输入的编码数据的连续数代码部，判定关注编码数据是否为上述第 1 类型、第 2 类型的任意一种类型的编码数据；

编码数据分离步骤，当在上述判定步骤中判定为关注编码数据是上述第 1 类型的编码数据时，输出关注编码数据中的由连续数代码部所表示的连续个数信息和由数据部所表示的数据，当在上述判定步骤中判定为关注编码数据是上述第 2 类型的编码数据时，在输出关注编码数据中的数据部的各个数据时，输出表示相同数据的连续个数为“1”的连续个数信息；

数据变更步骤，将在上述编码数据分离步骤中所输出的连续个数信息和数据中的上述数据的一部分位变更为预定值并输出；

再构建步骤，输入在上述编码数据分离步骤中所输出的连续个数信息和在上述数据变更步骤中所处理的数据，不进行解码处理，再构建上述第 1 类型或第 2 类型的编码数据的连续数代码部和数据部；以及

将再构建的连续数代码部和数据部作为再编码数据输出的步骤；

上述再构建步骤包括

判断步骤，每当输入来自上述编码数据分离步骤的连续个数信息和来自上述数据变更步骤的变更后的数据组时，判断是正在对相同数据的连续个数进行计数还是正在对不同数据的连续个数进行计数，以及现在输入的数据与此前输入的数据是否相等；和

当在上述判断步骤中判断为正在对相同数据的连续个数进行计

数、且现在输入的数据与此前输入的数据相等时，将在此前的输入中更新的连续个数信息加在现在输入的连续个数信息中，来更新现在连续个数信息，当在上述判断步骤中判断为正在对不同数据的连续个数进行计数、且现在输入的数据与此前输入的数据不相等时，将在此前的输入中更新的连续个数信息加在现在输入的连续个数信息中，来更新现在连续个数信息，当在上述判断步骤中判断为正在对相同数据的连续个数进行计数、且现在输入的数据与此前输入的数据不相等时，或者当正在对不同数据的连续个数进行计数、且现在输入的数据与此前输入的数据相等时，根据正在进行计数的未输出的数据生成数据部，根据计数出的连续数个数信息生成连续数代码部的步骤。

2. 根据权利要求 1 所述的编码方法，其特征在于：

还包括对上述数据变更步骤指定进行变更的位的位置的步骤。

3. 根据权利要求 1 或 2 所述的编码方法，其特征在于：

上述编码数据的数据部，由图像数据的像素的多个图像区属性的标志位所构成。

4. 根据权利要求 1 或 2 所述的编码方法，其特征在于：

上述编码数据的数据部，是对图像数据进行了编码的数据。

5. 根据权利要求 1 所述的编码方法，其特征在于：

上述编码数据，是压缩位编码数据。

6. 一种编码装置，依次输入第 1 类型的编码数据或者第 2 类型的编码数据，再编码成上述第 1 类型或第 2 类型的编码数据形式的编码数据，其中，上述第 1 类型的编码数据用表示相同数据的连续个数的连续数代码部和表示上述数据的数据部所表达，上述第 2 类型的编码数据用表示不同数据的连续个数的连续代码部和上述不同数据的数据部所表达，其特征在于，包括：

判定单元，根据所输入的编码数据的连续数代码部，判定关注编码数据是否为上述第 1 类型、第 2 类型的任意一种类型的编码数据；

编码数据分离单元，当由上述判定单元判定为关注编码数据是上述第 1 类型的编码数据时，输出关注编码数据中的由连续数代码部所

表示的连续个数信息和由数据部所表示的数据，当由上述判定单元判定为关注编码数据是上述第 2 类型的编码数据时，在输出关注编码数据中的数据部的各个数据时，输出表示相同数据的连续个数为“1”的连续个数信息；

数据变更单元，将由上述编码数据分离单元所输出的连续个数信息和数据中的上述数据的一部分位变更为预定值并输出；

再构建单元，输入由上述编码数据分离单元所输出的连续个数信息和由上述数据变更单元所处理的数据，不进行解码处理，再构建上述第 1 类型或第 2 类型的编码数据的连续数代码部和数据部；以及

将再构建的连续数代码部和数据部作为再编码数据输出的单元；

上述再构建单元包括

判断单元，每当输入来自上述编码数据分离单元的连续个数信息和来自上述数据变更单元的变更后的数据组时，判断是正在对相同数据的连续个数进行计数还是正在对不同数据的连续个数进行计数，以及现在输入的数据与此前输入的数据是否相等；和

当由上述判断单元判断为正在对相同数据的连续个数进行计数、且现在输入的数据与此前输入的数据相等时，将在此前的输入中更新的连续个数信息加在现在输入的连续个数信息中，来更新现在连续个数信息，当由上述判断单元判断为正在对不同数据的连续个数进行计数、且现在输入的数据与此前输入的数据不相等时，将在此前的输入中更新的连续个数信息加在现在输入的连续个数信息中，来更新现在连续个数信息，当由上述判断单元判断为正在对相同数据的连续个数进行计数、且现在输入的数据与此前输入的数据不相等时，或者当正在对不同数据的连续个数进行计数、且现在输入的数据与此前输入的数据相等时，根据正在进行计数的未输出的数据生成数据部，根据计数出的连续数个数信息生成连续数代码部的单元。

7. 根据权利要求 6 所述的编码装置，其特征在于：

还包括对上述数据变更单元指定进行变更的位的位置的单元。

8. 根据权利要求 6 或 7 所述的编码装置，其特征在于：

上述编码数据的数据部，由图像数据的像素的多个图像区属性的标志位所构成。

9. 根据权利要求 6 或 7 所述的编码装置，其特征在于：

上述编码数据的数据部，是对图像数据进行了编码的数据。

10. 根据权利要求 6 所述的编码装置，其特征在于：

上述编码数据，是压缩位编码数据。

编码方法和编码装置

技术领域

本发明涉及处理编码数据，进行再编码的技术。

背景技术

作为数据的压缩编码技术众所周知有压缩位编码（PackBits encoding）。此压缩位编码是行程编码（runlength encoding）之一，可不损失编码前的信息地进行解码，即、也是无损编码。

下面，设作为进行编码的单位的一个数据由8位构成来进行压缩位编码的说明（作为数据例如是8位多值像素数据）。另外，将进行编码以前的数据称为原始数据，将编码以前的原始数据的流称为原始数据流。

压缩位编码是指，将原始数据区别成“相同数据连续的部分”和“不同数据连续的部分”，并对各个部分附加数据的“长度信息”。然后，关于“相同数据连续的部分”以“连续的个数＋数据”的形式来表示，关于“不同数据连续的部分”以“不同数据连续的个数＋不同数据流”的形式来表示。然后将通过此处理所生成的数据流作为压缩位编码数据进行输出。

现在，输入下面（1）的原始数据流（24个数据）。各个数据用8位（0～255范围的数据）来表达。

0,0,0,0,1,2,3,4,4,4,4,4,4,4,4,5,5,6,7,8,8,9,10,10 …（1）

在此情况下，将原始数据流如下面那样区别成“相同数据连续的部分”和“不同数据连续的部分”。

0,0,0,0,

1,2,3,

4,4,4,4,4,4,4,4,

5,5,

6,7,

8,8,

9,

10,10 ...（2）

然后，对各自的部分附加“长度信息”，关于“相同数据连续的部分”则将数据部分仅压缩成一个。

0,0,0,0, => (-4),0

1,2,3, => (3),1,2,3,

4,4,4,4,4,4,4,4, => (-8),4

5,5, => (-2),5

6,7, => (2),6,7

8,8, => (-2),8

9, => (1)9,

10,10 => (-2),10 ...（3）

在（3）中“长度信息”用括弧包围来表示。用正负值来表示“长度信息”是为了区别连续的部分是相同数据，还是不同数据。即，负数表示“相同数据的连续部分”，正数表示“不同数据的连续部分”。然后，将此“长度信息”作为具有与数据相同的8位宽度的“长度代码数据”嵌入数据流并输出。即，将长度代码数据的8位的最高位MSB作为正负的符号位来处理。

另外，关于如上述值“9”那样在连续的数据之间仅孤立存在一个的数据，则定义为“不同数据连续一个”。为此，相同数据连续的个数最少也在2个以上。

长度代码数据、“相同值的数据连续的个数”以及“不同值的数据连续的个数”的关系（表）如下所示（注：“0x”表示16进制数）。

不同值的数据连续的个数	长度代码数据
1	0x00
2	0x01

3	0x00
:	
127	0x7E
128	0x7F
相同值的数据连续的个数	长度代码数据
2(-2)	0xFF
3(-3)	0xFE
:	
127(-127)	0x82
128(-128)	0x81

从而，若对先前所示的原始数据流（1）进行压缩位编码，则生成下面所示的编码数据流（4）。

FD,00,02,01,02,03,F9,04,FF,5,01,06,07,FF,08,00,09,FF,10…（4）

如上所述，在压缩位编码中，不同数据连续的个数能够用 1 至 128 来表达，相同数据连续的个数能够用 2 至 128 来表达。从而，假设在相同数据连续的个数为 129 以上的情况下，生成连续至 128 个的代码（长度代码数据 + 连续的数据），关于第 129 以后，则生成以该第 129 个数据为起点的代码。这在不同数据连续的情况下也同样。

此外，如上述表所示那样在压缩位编码中作为“长度代码数据”不生成 0x80。从而，有时也将此“0x80”定义为表示编码数据流最后的列表结束代码数据。

以后，用“长度信息”来表达在压缩位编码数据中“长度代码数据”，用“数据部”来表达表示接着长度信息而输出的实际的数据值的部分。

如上面那样，在压缩位编码中若相同值的数据连续出现的次数增加则编码后的数据量就减少（压缩率变高）。

一般，在压缩数据的情况下，对于原始数据的量（大小），大多确立 1/2 以下（压缩率 2 倍以上），或者 1/4 以下（压缩率 4 倍以上）这样的目标来进行压缩编码。在压缩位编码的情况下，相同数据连续

3 个以上的情况下压缩率提高，但在不同数据连续较多的情况下只有附加了“长度信息”的部分的数据量增加。

从而，进行压缩位编码后不能到达目标的数据量的情况下，只要不施加变更以使相同数据连续，就不能实现。

这里，若将原始数据的某个位固定成 0，则可取的值的种类数就减少一半，相同值连续的概率必然增高，因此在压缩这样的数据的情况下就得到高的压缩率。例如，若作为压缩编码对象以多值像素为例，则在将 LSB 设成 0 的情况下，用 2n，2n+1（n 是 0、1、2...）所表达的像素值都被舍入成“2n”。在半色调图像的情况下原本邻接的像素的浓度和亮度之差较小，因此如上述那样通过变更位值就更容易成为相同值。即，可知如果进行上述位变更处理，则得到较高的压缩率。同样，能够理解若将适当的 2 位设成 0，则得到更高的压缩率。

但是，为了对作为初始的原始数据的集合（例如多值图像数据）的一个个的数据进行上述的舍入处理，就需要与该数据量成比例的处理，不能期望较高的处理速度。另外，为此还需要存储全体初始的原始数据的存储器。

发明内容

因此，本申请发明的目的就是提供一种技术，即将如压缩位编码数据那样的，以表示相同数据的连续个数的连续数代码部和表示上述数据的数据部，以及表示不同数据流的连续数的连续数代码部和表示上述不同数据流的数据部的数据形式所表达的编码数据，以更高的压缩率再编码成相同数据形式，而不用对其进行解码。

为了解决这样的课题，本发明的技术方案提供一种编码方法，依次输入第 1 类型的编码数据或者第 2 类型的编码数据，再编码成上述第 1 类型或第 2 类型的编码数据形式的编码数据，其中，上述第 1 类型的编码数据用表示相同数据的连续个数的连续数代码部和表示上述数据的数据部所表达，上述第 2 类型的编码数据用表示不同数据的连续个数的连续代码部和上述不同数据的数据部所表达，其特征在

于，包括：判定步骤，根据所输入的编码数据的连续数代码部，判定关注编码数据是否为上述第 1 类型、第 2 类型的任意一种类型的编码数据；编码数据分离步骤，当在上述判定步骤中判定为关注编码数据是上述第 1 类型的编码数据时，输出关注编码数据中的由连续数代码部所表示的连续个数信息和由数据部所表示的数据，当在上述判定步骤中判定为关注编码数据是上述第 2 类型的编码数据时，在输出关注编码数据中的数据部的各个数据时，输出表示相同数据的连续个数为“1”的连续个数信息；数据变更步骤，将在上述编码数据分离步骤中所输出的连续个数信息和数据中的上述数据的一部分位变更为预定值并输出；再构建步骤，输入在上述编码数据分离步骤中所输出的连续个数信息和在上述数据变更步骤中所处理的数据，不进行解码处理，再构建上述第 1 类型或第 2 类型的编码数据的连续数代码部和数据部；以及将再构建的连续数代码部和数据部作为再编码数据输出的步骤；上述再构建步骤包括判断步骤，每当输入来自上述编码数据分离步骤的连续个数信息和来自上述数据变更步骤的变更后的数据组时，判断是正在对相同数据的连续个数进行计数还是正在对不同数据的连续个数进行计数，以及现在输入的数据与此前输入的数据是否相等；和当在上述判断步骤中判断为正在对相同数据的连续个数进行计数、且现在输入的数据与此前输入的数据相等时，将在此前的输入中更新的连续个数信息加在现在输入的连续个数信息中，来更新现在连续个数信息，当在上述判断步骤中判断为正在对不同数据的连续个数进行计数、且现在输入的数据与此前输入的数据不相等时，将在此前的输入中更新的连续个数信息加在现在输入的连续个数信息中，来更新现在连续个数信息，当在上述判断步骤中判断为正在对相同数据的连续个数进行计数、且现在输入的数据与此前输入的数据不相等时，或者当正在对不同数据的连续个数进行计数、且现在输入的数据与此前输入的数据相等时，根据正在进行计数的未输出的数据生成数据部，根据计数出的连续数个数信息生成连续数代码部的步骤。

附图说明

图 1 是实施方式中的压缩位再编码部的块结构图

图 2 是表示实施方式中的数据分割部 101 的动作处理内容的流程图。

图 3 是表示实施方式中的数据结合部 103 的动作处理内容的流程图。

图 4 是表示实施方式中的数据结合部 103 的动作处理内容的流程图。

图 5 是表示实施方式中的数据结合部 103 的动作处理内容的流程图。

图 6 是表示实施方式中的数据结合部 103 的动作处理内容的流程图。

图 7 是表示实施方式中的数据结合部 103 的动作处理内容的流程图。

图 8 是表示实施方式中的数据结合部 103 的动作处理内容的流程图。

图 9 是表示实施方式中的数据输出部 104 的动作处理内容的流程图。

图 10 是表示实施方式中的数据分割部 101 和数据处理部 102 的输入输出时间表。

图 11 是表示实施方式中的数据结合部 103 的输入输出时间表 。

图 12 是伴随实施方式中的数据输出部 104 的动作的内部状态的转变图。

图 13 是伴随实施方式中的数据输出部 104 的动作的内部状态的转变图。

图 14 是实施方式适用的数字复印机的块结构图。

图 15 是图 14 中的图像区信息编码部的块结构图。

图 16 是第 2 实施方式中的压缩位再编码装置的块结构图

具体实施方式

下面，按照附图来说明与本发明有关的实施方式。

<概要说明>

在实施方式中，以数字复印机为例进行说明。

图 14 是实施方式中的数字复印机的块结构图。图中，1 是负责控制装置整体的控制部，由 CPU、存储着其控制处理程序的 ROM 以及作为工作区来使用的 RAM 构成。2 是读取原稿的阅读部，搭载有将原稿图像作为彩色图像进行读取的摄像单元以及 ADF（Auto Document Feeder）。3 是前处理部进行各种校正处理（例如包含黑斑校正等）。4 是对读取得到的图像数据进行压缩编码的编码部，例如按照 JPEG 编码方法进行压缩编码。5 是暂时存储压缩编码数据的存储部。从而，若从阅读部 2 依次读取原稿图像，则该压缩编码数据就被储存到存储部 5 中。6 是解码部，对储存在存储部 5 中的压缩编码数据进行解码。7 是图像处理部，对由解码部 12 进行解码得到的每个像素生成记录色成分的数据，同时基于来自后述的解码部 12 的每个像素的信息，对图像区进行特殊的处理。例如，对字符线条区域，其结果被输出到打印机引擎 8，进行打印。设打印机引擎 8 为彩色激光打印机引擎，但当然即使是喷出墨液的类型也没有关系，在打印方式上没有限定。

另一方面，从前处理部 3 输出的各色成分的数据，被供给图像区判定部 9，在此以像素为单位判定是处于怎样的图像区，对每个像素将此判定结果作为 1 字节的属性信息进行输出。

作为由此图像区判定部 9 进行判定的内容，包含关注像素是否是

处于网点中的字符线条的像素、是有彩色/无彩色的哪个、是否是字符（是字符线条区域还是半色调区域）以及是字符线条的情况下其线条部分的粗细信息。

是否是字符线条的判定，通过关注像素和邻接像素之间的亮度变化是否急剧（是否超过阈值）来进行判断。然后，在被判定为其急剧的像素连续预定数的情况下判定为字符线条，在不满预定数的情况下判断为是网点。从而，是否是处于网点中的字符，通过判定关注像素是字符、并判定为在距其预定的距离范围内有网点的像素数是否存在多个来进行判断。此外，在判断为既不是字符线条也不是网点的情况下，判断为半色调区域即可。

另外，有彩色还是无彩色的判断，在关注像素的各亮度色成分RGB的亮度值之差是阈值以下的情况下判断为无彩色。为了进一步精度良好地进行判定，如果在包含关注像素的附近即使存在一个有彩色的像素，也将关注像素判断为有彩色。

另外，所谓字符的粗细信息，计数被判定为字符线条的像素数即可，该字符线条为和沿着被判断为是字符的像素的边缘的方向垂直的方向的字符线条。

如上所述，图像区判定部9，对每个像素将其判定结果作为8位的数据进行输出。8位的详细内容如下。

Bit	含义
7	预留（0）
6	预留（0）
5	字符粗细
4	字符粗细　用位4、5表达字符·线条的粗细0~3
3	预留（0）
2	字符　字符线条=1、非字符线条（半色调）=0
1	网点中字符　处于网点中的字符·线条=1、除此以外=0
0	无彩色　无彩色=1、有彩色=0

于是，由图像区判定部9，如上所述对每个像素输出1字节的图

像区信息，图像区信息编码部 10 通过对此信息进行压缩位编码来进行压缩。本实施方式中的特征部分，在于此图像区信息编码部 10，关于其细节在后面叙述。11 是暂时存储由图像区信息编码部 10 进行了压缩位编码的图像区信息的存储部。12 是解码部，对存储在存储部 11 中的被压缩编码的图像区信息进行解码，并供给图像处理部 7。从而，解码部 6、12 相互同步，将解码结果输出给图像处理部 7。

在上述结构中，实施方式中的图像区信息编码部 10 的结构，例如取图 15 所示的结构。

图中，11 是按照来自后述的监视部的指示，对来自图像区判定部 9 的 1 字节（像素的图像区信息）的各位有条件地进行屏蔽的屏蔽部，在 1 页的初始状态下通过全部的位。

12 是依次输出经由屏蔽部 11 所输入的图像区数据，进行公知的压缩位编码的压缩位编码部。13 是在输出到存储部 12 时，用于暂时进行保存的缓冲存储器。14 是对储存在缓冲存储器 13 中的数据量进行监视的监视部（处于控制部 1 的控制下），在对应于 1 张原稿图像的属性信息的代码数据的保存完成以前就达到了预先设定的代码量（阈值）的情况下，决定用屏蔽部 11 进行屏蔽的位，同时生成对于在下面进行说明的压缩位再编码部 15 的控制信号。15 是压缩位再编码部，对保存在缓冲存储器 13 中的被压缩编码的图像区信息进行再编码，再次保存到缓冲存储器 13。此时的再编码条件（压缩参数），因从监视部 14 输出来的控制信号而异（细节后述）。另外，监视部 14，还对此再编码的数据量进行监视。16 是位于压缩位再编码部 15 和缓冲存储器 13 之间，进行压缩位再编码部 15 的数据读入写入用的存储器控制部。

于是，在上述结构中，实施方式中的监视部 14，在进行 1 张原稿的图像读取的初始阶段，在屏蔽部 11 中进行设定以通过 8 位的全部数据。然后，对储存在缓冲存储器 13 中的图像区信息的编码数据量进行监视，在判断为 1 页的数据被保存以前就达到预先设定的数据量（称为溢出状态）的情况下，对屏蔽部 11 屏蔽预定的位，对在此以

后所输入的图像区信息施加限制。另外，对已储存在缓冲存储器 13 中的编码数据（图像区信息），用压缩位再编码部 15 进行再编码。在进行再编码时，与屏蔽部 11 同样设定为屏蔽预定位。

结果，在缓冲存储器 13 中，对已判断为溢出状态的以后的数据，进行基于被屏蔽的结果的压缩位编码，对比已判断为溢出状态的定时还靠前进行了编码的图像区信息的编码数据用压缩位再编码部 15 进行再编码。监视部 14，在成为溢出状态时，临时复位已监视的数据量，继续监视从压缩位编码部 11 输出的数据量和压缩位再编码部 15 中的再编码数据量的合计值是否再次溢出。然后，在 1 页的属性的压缩编码数据量再次溢出了的情况下，进一步增加进行屏蔽的位，反复上述处理。

实施方式中的溢出（OF）的次数、图像区信息的屏蔽条件及位如下：

OF 次数　条件&屏蔽位

0 没有屏蔽

1 如果字符标志（位 2）是 0，则设无彩色（位 0）为 0

2 如果字符标志是 0，则设字符粗细（位 4、5）为 0

3 如果字符标志是 0，则设网点字符标志（位 1）为 0

4 字符标志以外全部为 0

5 全部为 0

如果使上述内容容易理解地进行说明，则意味着将像素的属性变更成包含它的属性。原因是例如，在最初的溢出中使无彩色成为有彩色，可以说成在有彩色空间中包含无彩色空间。

此外，在屏蔽部 11 中，在溢出次数例如为第 2 次的情况下，以第 1 次和第 2 次的 OR 条件进行屏蔽的位就成为 2。另一方面，压缩位再编码部 15 中，如上所述就可。这是因为，进行再编码的对象在上次的编码、或者上次的再编码中已经屏蔽了。

另外，在上述的处理中，压缩位编码部 12 的处理本身单纯地继续进行与溢出次数无关的处理即可。另一方面，期待压缩位再编码部

15 进行非常高速的处理。

着眼于本实施方式中的压缩位再编码部 15 进行再编码的对象是已进行压缩位编码的数据，该代码数据原样进行再编码。下面，说明其具体的解决对策。

<压缩位再编码部 15 的说明>

图 1 使本实施方式中的压缩位再编码部的框图。下面，使用该图的框图对其结构和处理内容进行说明。

图 1 中，101 是数据分割（分离）部，经由存储器控制部 16 来判别从 encode_data（图中的 111）输入的压缩位编码数据的长度信息和数据部，并通过分析所输入的长度信息，与数据值同时输出紧跟着长度信息所输入的数据部的数据值连续多少个（下面，记为连续个数）。即，若说成对长度信息和数据部进行分离，则容易理解。

本实施方式的数据分割部 101 除接收压缩位编码数据的 encode_data 以外，还使用作为表示从 encode_data 输入压缩位编码数据的有效信号的 data_valid（110）、表示编码数据的输入已结束的结束信号 data_end（113）以及把数据分割部为可输入编码数据的状态通知存储器控制部 16 的就绪信号 data_ready（112），来取得压缩位编码数据。换言之，若此 data_ready 信号变得有效，存储器控制部 16 就从缓冲存储器 13 读出下一数据。

另外，来自数据分割部 101 的输出使用 data（115）输出数据值，使用 num（116）输出连续个数。在本实施方式中，除上述的 data、num 的总线以外，还使用表示从上述 data 和 num 输出的值有效的有效信号 valid（114）、表示接收侧可进行数据的接收的就绪信号 ready（117）以及表示数据的输出已结束的结束信号 end（118）进行数据的发送和接收。后面对这些输入输出信号的动作进行说明。

102 是数据处理部，按照来自监视部 14 的控制信号，对从数据分割部 101 输出的数据值，按照先前所说明的条件，对作为 data（115）所输入的数据值，进行将其屏蔽（设位值为 0）的处理，并作为 masked_data（121）进行输出。

103 是数据结合部，对参照数据处理部 102 作为 masked_data 输出的数据值以及数据分割部 101 从 num 输出的连续个数能够削减数据量的部分，进行压缩位方式的编码处理生成新的编码数据。

此外，在本实施方式中，除 masked_data 以外还使用在数据分割部 101 中说明过的 valid、就绪信号 ready、以及结束信号 end 来取得数据值和连续个数。

由数据结合部所生成的编码数据，使用 v_out（133）进行输出。但是从数据结合部 103 输出的编码数据与通常的压缩位编码数据不同，首先输出数据部的数值，紧跟着数据部长度信息从 v_out 输出。

为此在本实施方式的数据结合部中，为了判别从 v_out 输出的数据是长度信息还是数据部，使用作为长度信息有效信号的 l_valid 和作为数据部有效信号的 d_valid、以及表示数据的输出已结束的结束信号 v_end（134）进行编码数据的输出。

104 是数据输出部，对从数据结合部 103 取得的编码数据进行长度信息和数据部的排列顺序的替换，以成为通常压缩位编码数据的顺序。通过上述数据结合部 103 和数据输出部 104，重新构成压缩位编码数据。

在本实施方式中进行了排列替换的压缩位编码数据汇总 8 个数据作为 64 位从 reencode_data（141）输出，同时从 address（142）输出保存从 reencode_data 输出的数据的后级的存储器地址。另外，在从 reencode_data 以及 address 输出数据和地址时，从作为有效信号的 reencode_vald 输出 1 以表示上述数据和地址是有效的。

另外，若来自数据输出部的地址和数据的输出结束，则通过从 reencode_end（143）输出 1 来告知后级数据的输出已结束。

若将数据分割部 101 的动作以过程来表示，则如图 2 所示的流程图那样。

若数据分割部 101 开始动作则对内部计数器（图 2 中记为 i）进行初始化。具体来讲，就是将计数器 i 设定成 0（步骤 S201），接着进行压缩位编码数据的输入是否结束的判断（步骤 S202）。

数据的输入结束的判断如在先前所说明那样还有“作为长度信息输入 128（用 16 进制数表达是 80h，下面紧跟数字的括弧内表示 16 进制数表达）”这样的方法，在本实施方式中通过从 data_end 输入“TURE = 1”来表示数据的输入结束。

若在步骤 S202 中判断为数据输入没有结束（data_end = 0），则数据分割部等待从 encode_data 输入数据。这里，如在以往的技术中所示的（3）和（4）那样压缩位编码数据最初是输入长度信息，所以具体来讲就是等待输入长度信息。

若从 encode_data 输入作为最初的编码数据的长度信息（图 2 中记为 L）（步骤 S203），则在数据分割部中参照所输入的 L 的值判断长度信息是表示“相同数据连续的个数”还是表示“不同数据连续的个数”。具体来讲，将所输入的长度信息 L 作为没有正负符号的数据，与阈值“128（80h）”进行比较（步骤 S204），在 L 大于 128 的情况下判断为“相同数据连续的个数”，在不足 128 的情况下判断为“不同数据连续的个数”。

在所输入的长度信息 L 比 128（80h）大的情况下，在数据分割部中从长度信息 L 求出相同数据连续的个数并代入变量 N。在本实施方式中由于相同数据连续的个数通过 2 的补码作为负值来表达，所以从长度信息 L 进行连续数的计算（步骤 S205）。具体来讲，通过

$$N = 256 - L + 1$$

求出连续数并代入 N。

若计算出相同数据连续的个数，则数据分割部等待再次从 encode_data 输入压缩位编码数据。在压缩位编码数据中长度信息为 128 以上的情况下，下一数据由于输入连续的数据的值，因此在这里等待数据部（数据值）的输入。

若从 encode_data 输入数据部（图 2 中记为 D）（步骤 S206），则数据分割部从 num 输出（116）输出作为数据连续个数的 N，从 data 输出（115）输出作为数据值的 D（步骤 S207）。

由此，将“数据值 D 为 N 个”这样的信息送给后级，再次返回

步骤 S202 的数据输入结束的判断。

另外，在所输入的长度信息 L 不足 128（80h）的情况下，在数据分割部中从长度信息 L 求出不同数据连续的个数并代入变量 N'（步骤 S208）。在本实施方式中不同数据连续的个数是在长度信息上加了 1 的值，因此具体来讲，通过

N' = L + 1

来求出不同数据的数。

若计算出不同数据的数，则数据分割部等待再次从 encode_data 输入压缩位编码数据。在压缩位编码数据中长度信息为 128 以下的情况下，按跟着长度信息先前计算出的 N'的数相应输入数据的值。从而在这里等待数据部（数据值）的输入。

若从 encode_data 输入数据部 D（步骤 S209），则数据分割部从 num 输出（116）输出 1，从 data 输出（115）输出数据值 D（步骤 S210）。由此，将“从 data 输出的数据值 D 是 1”这样的信息发送给后级。

若从 data 输出数据值 D，则数据分割部在内部计数器 i 上加 1（步骤 S211），与 N'进行比较（步骤 S212）。在计数值 i 比 N'小的情况下，由于接着输入数据值故返回步骤 S209 等待从 encode_data 输入数据部。

另外由步骤 S212 进行了比较的结果是计数值 i 和 N'的值相等的情况下，由于数据值被输入了 N'的个数，所以将计数器 i 初始化成 0（步骤 S213），返回步骤 S202 的数据输入结束的判断。

通过反复上述步骤 S202～213 的动作进行压缩位编码数据的输入，最后若在步骤 S202 中输入 data_end = 1 并检测出压缩位编码数据的输入结束，则数据分割部 101 从 end 输出输出 1 以表示已结束数据输出，并结束动作。

图 10 中示出数据分割部 101 的输入输出时间表的一例。

在图 10 中，clk 表示时钟，各信号同步于 clk 进行输入输出。data_valid、encode_data、data_ready 以及 data_end 各信号是用于将压缩位编码数据输入到数据分割部的信号。N、N'以及 i 表示在数据分

割部中使用的变量以及计数器值，valid、num、data、ready 以及 end 信号是用于将数据部以及连续个数输出到数据处理部 102 以及数据结合部 103 的信号。

此外，在图 10 中从 encode_data 输入的压缩位编码数据之中，对表示长度信息的部分画上双下划线来表示。

在图 10 中若数据分割部 101 的动作开始，则在图 10 的定时 t0 进行内部计数器 i 的初始化动作（步骤 S201）。然后，通过从 t1 开始设 data_valid = 1 压缩位编码数据开始从 encode_data 输入到数据分割部。

由于最初输入到数据分割部的压缩位编码数据是长度信息，所以数据分割部将在 t1 中输入的值 253(= FDh)作为长度信息来取得(步骤 S203)。然后比较所取得的长度信息 L 和 128(80h)(步骤 S204)，由于不满足

$L < 128$

的条件，故使用所取得的长度信息 L，通过步骤 S205 所示的运算计算出数据连续个数 4（04h）并保存在 N 中。

然后，在定时 t2 将从 encode_data 输入的值 0（00h）作为数据值 D 来取得（步骤 S206），在定时 t3 从 num 输出数据连续个数 N，从 data 输出数据值 D 以后（步骤 S207），在步骤 S202 中参照 data_end 判断数据输入是否结束。

由于在定时 t3 没有从 data_end 输入 1，所以进入步骤 S203，将在 t3 从 data_end 输入的值 248（F8h）作为长度信息 L 来取得（步骤 S203）。然后比较长度信息 L 和 128（80h），进入步骤 S205 计算出数据连续个数 9（09h）并保存在 N 中。然后在定时 t4 从 encode_data 作为数据值 D 取得 1（01h）（步骤 S206），在定时 t4 从 num 输出数据连续个数 N，从 data 输出数据值 D。

然后再次在步骤 S202 中进行数据输入结束的判断，由于定时 t5 中的 data_end 是 0 故进入步骤 S203，将在定时 t5 从 encode_data 输入的 6（06h）作为长度信息 L 来取得。

通过步骤 S204 比较长度信息 L 和 128（80h），由于满足

$L < 128$

的条件，故在通过步骤 S208 所示的公式从长度信息 L 计算出跟随长度信息所输入的数据值的个数 7(07h)并保存在 N'中(步骤 S208)以后，等待从 encode_data 输入数据部 D。

若在定时 t6 从 encode_data 作为数据值 D 输入“2（02h）”（步骤 S209），在 t7 从 num 作为数据连续个数输出 1（01h），从 data 输出作为数据值 D 的“2（02h）”（步骤 S210）。然后在内部计数器 i 上加 1（步骤 S211），并与在步骤 S208 中计算出的 N'进行比较（步骤 S212）。

由于在定时 t7 内部计数器的值为 1 故和存储在 N'中的数据输入个数 7 不相等，所以返回步骤 S209 等待再次从 encode_data 输入数据部 D。

由于在定时 t7 data_valid 为 1 故从 encode_data 输入的数据是有效的，但由于后述的数据结合部 103 未能进行数据的输入准备故将 ready 信号设成 0。为此，在数据分割部中也从 data_ready 输出 0 并停止数据的取得。此时，从 encode_data 输入的数据就被保持直到下一 clk。

若在 t8 数据结合部 103 可进行数据的输入，并从 ready 输入 1，则数据分割部也从 data_ready 输出 1 并从 encode_data 作为数据值 D 取得 3（03h）。

以后，通过由步骤 S209～步骤 S212 反复同样的动作 N'次。从 encode_data 作为输入值在定时 t9 取得 2（02h），接着，在 t10 取得 3（03h），在 t11 取得 4（04h），在 t13 取得 3（03h），在 t14 取得 1（01h），从 data 在定时 t9 输出 3（03h），在 t10 输出 2（02h），在 t11 输出 3（03h），在 t12 输出 4（04h），在 t14 输出 3（03h），在 t15 输出 1（01h）。

由于在定时 t15 内部计数器 i 的值成为 7 变得与数据输入个数 N'相等，故在步骤 S212 中的比较以后，将内部计数器 i 初始化成 0（步骤 S213），返回步骤 S202 参照 data_end 来判断数据输入是否结束。

由于在定时 t15 没有从 data_end 输入 1，故进入步骤 S203，将在定时 t15 从 encode_data 输入的值 253（FDh）作为长度信息 L 来取得（步骤 S203），在步骤 S204 中与 128（80h）进行了比较以后，进入步骤 S205 计算出数据连续个数 4（04h）并保存在 N 中。然后在定时 t16 从 encode_data 作为数据值 D 取得 0（00h）（步骤 S206），在定时 t17 从 num 输出数据连续个数 N，从 data 输出数据值 D，返回步骤 S202。

由于在定时 t17 从 encode_data 输入 1，所以由数据分割部在步骤 S202 中判断为数据输入已结束，通过在定时 t18 从 end 输出输出 1 来告知后级数据输出的结束（步骤 S214），结束本动作。

图 10 的 masked_data 是来自数据处理部 102 的输出，在本实施方式中表示对从数据分割部 101 的 data 输出的值，在最初的溢出中由数据处理部 102 所处理的结果的例子。即，在表示从 data 输入的值是字符的位 2 为 0 的情况下，进行将位 0 设成 0 的处理并从 masked_data 输出。此外，可知在下一溢出（第 2 次的溢出）的情况下，条件又不同。

此外，由于在本实施方式的数据处理部 102 中，如图 10 所示那样从 data 将值输入到数据处理部后没有 clk 延迟地输出 masked_data，故从 masked_data 输出的数据，和表示从数据分割部的 num 输出的数据的连续个数的信息在相同的定时被给与数据结合部。

接着，按照图 3 至图 8 的流程图对实施方式中的数据结合部 103 的动作处理进行说明。

若数据结合部 103 开始其处理，则等待从 num 和 masked_data 输入最初的数据连续个数和由数据处理部 102 所处理的数据值。

若输入数据连续个数和数据值（步骤 S301），则将该值作为初始值把从 num 输入的数据连续个数保存在内部变量 cnt 中，把从 masked_data 输入的数据值保存在内部变量 data_buf 中（步骤 S302）。

然后参照保存在 cnt 中的值（步骤 S303），在该值为 1 之外的情况下判断为从 masked_data 输入的数据值是“相同数据值连续”部分

的数据值，为了在数据连续部保持该状态，将内部变量 continue 设定成“1”（步骤 S304）。

另外，在保存在 cnt 中的值为 1 的情况下判断为从 masked_data 输入的数据值是“相同数据值不连续”部分的数据值，为了在数据连续部保持此状态，将内部变量 continue 设定成“0”（步骤 S305）。

通过以上处理，用于输入下一数据的准备结束。

接着数据结合部判断数据输入的结束（步骤 S306）。在本实施方式中通过数据分割部将 end（118）设成“1 = high”判断为数据的输入已结束，分支到图 8 的处理。在没有从 end（118）输入 1 的情况下，数据分割部等待从 num（116）和 masked_data（121）输入下一数据连续个数和数据值。

若从 num（116）和 masked_data（121）输入下一数据连续个数和数据值（步骤 S307），则数据结合部根据内部变量 continue 的值来进行分支（步骤 S308）。在 continue 的值是表示“相同数据值不连续”部分的 0 的情况下，进一步比较从 masked_data 输入的数据值和内部变量 data_buf 的值（步骤 S309）。这里在从 masked_data 输入的数据值与 data_buf 的值不同的情况下分支到图 4 的处理。另外在从 masked_data 输入的数据值与 data_buf 的值相等的情况下，分支到图 5 的处理进行用于结束“相同数据值不连续的部分”的处理，和用于重新开始“相同数据值连续的部分”的处理。

另一方面，在根据步骤 S308 内部变量 continue 的值表示“相同数据值连续”部分的 1 的情况下，同样地比较从 masked_data 输入的数据值和内部变量 data_buf 的值（步骤 S310），在从 masked_data 输入的数据值与 data_buf 的值不同的情况下分支到图 6 的处理。另外在从 masked_data 输入的数据值与 data_buf 的值相等的情况下，分支到图 5 进行用于继续“相同数据连续的部分”的处理。

图 4 是表示数据结合部中，内部变量 continue 为 0 且从 masked_data 输入的数据值与内部变量 data_buf 的值不同的情况下（图 3 中的步骤 S309 为 no）的处理的流程图。

在此情况下，由于为了生成新的压缩位编码数据而需要保存在data_buf 中的值，故将保存在 data_buf 中的值从 v_out 输出（步骤S401）。然后参照从 num 输入的数据连续个数（步骤 S402）。

如果数据连续个数为 1 以外，则由于其表示从 masked_data 输入的值连续一个以上，故“相同数据不连续”部分在此结束。为此将内部变量 continue 的值设定成表示“相同数据连续”部分的 1（步骤S403），通过从保存了迄今所持续的“不同数据连续”部分的数据数的内部计数器 cnt 减 1 来求得表示“不同数据连续的个数”的长度信息，并将该长度信息从 v_out 输出（步骤 S404）。

然后，在将内部变量 cnt 初始化成 0（步骤 S405）后，将从 num 输入的数据连续个数加到 cnt 上（步骤 S406），将从 masked_data 输入的数据值保存到内部变量 data_buf 中（步骤 S407），进行了下一数据接收准备以后，返回图 3 的步骤 S306。

另外，当在步骤 S402 中从 num 输入的数据连续个数为 1 的情况下，表示随后是“相同数据不连续”部分。在此情况下首先参照内部变量 cnt 的值，判断是否是 128（80h）（步骤 S408）。在本实施方式中由于用 8 位来表示压缩位代码数据，故“相同数据连续”部分和“相同数据不连续”部分的数只能表示到 128。为此在内部变量 cnt 的值是 128 的情况下，就需要输出“相同数据不连续的部分”持续 128 次这样的信息。为此通过步骤 S404 计算出长度信息（在此情况下“由于不同数据连续 128 个”故作为长度信息生成 127（7Fh））并从 v_out 输出。

然后，在将内部变量 cnt 初始化成 0（步骤 S405）后，将从 num 输入的数据连续个数（在这里是 1）加到 cnt 上（步骤 S406），将从 masked_data 输入的数据值保存到内部变量 data_buf 中（步骤 S407），进行了下一数据接收准备以后返回图 3 的步骤 S306。

当在步骤 S408 中内部变量 cnt 的值不是 128（不足 128）的情况下，将从 num 输入的数据连续个数（在这里是 1）加到 cnt 上（步骤 S406），将从 masked_data 输入的数据值保存到内部变量 data_buf 中

（步骤 S407），进行了下一数据接收准备以后，返回图 3 的步骤 S306。

图 5 是表示数据结合部中，内部变量 continue 为 0 且从 masked_data 输入的数据值与内部变量 data_buf 的值相同的情况下（图 3 中的步骤 S309 为 yes）的处理的流程图。

在此情况下，由于直到前一处理 continue 都为 0 故进行“相同数据不连续”部分的处理，但由于从 masked_data 输入的数据值和内部变量 data_buf 的值相同故“相同数据不连续”部分在前一数据就结束。因此将内部变量 continue 的值设定成表示“相同数据连续”部分的 1（步骤 S501），参照内部变量 cnt 的值（步骤 S502）。如果 cnt 的值不是 1，就需要计算并输出迄今所持续的“不同数据连续”部分的数据数。

为此，由于保存在当前 data_buf 中的值包含在“相同数据连续”部分，故实际上“相同数据不连续”部分的数据数就成为从内部计数器 cnt 减去 1 的值。另外，由于表示“不同数据连续的个数”的长度信息是从“不同数据连续”的数减去 1 的值，故在这里从内部变量 cnt 减 2 求出长度信息，并将该长度信息从 v_out 输出（步骤 S503）。

然后由于与从 masked_data 所输入的数据值相同的数据保存在 data_buf 中故在将内部变量 cnt 设定成 1（步骤 S504）后，将由 num 输入的数据连续个数加到 cnt 上（步骤 S505）并进行下一数据的接收准备，返回图 3 中的步骤 S306。

另外，当在步骤 S502 中 cnt 值为 1 的情况下，将从 num 输入的数据连续个数加到 cnt 的值上（步骤 S506）。

在本实施方式中，由于用 8 位来表示压缩位编码数据，故“相同数据连续”部分的数只能表示到 128，因此在内部变量 cnt 的值为 129 以上的情况下，就需要输出“相同数据连续”部分持续 128 次这样的信息。为此参照内部变量 cnt 的值（步骤 S507），在该值为 129 以上的情况下从 v_out 输出保存在 data_buf 中的数据值（步骤 S508），跟随其从 v_out 输出作为表示“相同数据连续 128 个”的长度信息的 129（81h）（步骤 S509）。此外，在以往技术中说明过的将长度信息和

代码作为表进行存储保持，此 129 这样的长度信息，既可以参照其来求出，也可以根据表示相同数据持续 n 个的长度信息为

256 - n + 1 来求出。若通过步骤 S508 输出长度信息，则数据结合部在从内部变量 cnt 减去在后级作为压缩位编码数据输出的“128”（步骤 S510），并进行了下一数据的接收准备以后，返回图 3 的步骤 S306。

另外，当在步骤 S507 中内部变量 cnt 的值不足 129 的情况下，返回图 3 的步骤 S306。

此外，在这里由于压缩位编码所需要的数据值已经保存在 data_buf 中，故不需要将从 masked_data 输入的数据值保存在内部变量 data_buf 中，但也可以进行保存。

图 6 是表示数据结合部中，内部变量 continue 为 1 且从 masked_data 输入的数据值与内部变量 data_buf 的值不同的情况下（图 3 中的步骤 S310 为 No）的处理的流程图。

在此情况下，由于从 masked_data 输入的数据值与 data_buf 的值不同，故“相同数据连续”部分在此结束。为此将保存在 data_buf 中的值从 v_out 输出（步骤 S601）。然后参照从 num 输入的数据连续个数（步骤 S602）。

如果数据连续个数为 1 以外，则由于其表示从 masked_data 输入的值连续一个以上，“相同数据连续”部分再次开始。为此内部变量 continue 的值保持表示“相同数据连续”部分的 1 不变，从保存了迄今所持续的“相同数据连续”部分的数据数的内部计数器 cnt 求出长度信息（通过从 256 减去 cnt 的值，进而加上 1 来求出），并将该长度信息从 v_out 输出（步骤 S603）。

然后，在将内部变量 cnt 初始化成 0（步骤 S604）后，从 masked_data 取得新的数据值并保存在 data_buf 中（步骤 S605），进而将从 num 输入的数据连续个数加到 cnt 上（步骤 S606）。这样若下一数据接收准备结束，则返回图 3 的步骤 S306。

另外，当在步骤 S602 中从 num 输入的数据连续个数为 1 的情况

下，表示“相同数据不连续”部分开始。在此情况下首先将内部变量continue的值设定成表示“相同数据不连续”部分的0（步骤S607）。并参照内部变量cnt的值（步骤S608）。

如果cnt的值为1以外，则从保存了“相同数据连续”部分的数据数的内部计数器cnt求出长度信息（通过从256减去cnt的值，进而加上1来求出），并将该长度信息从v_out输出（步骤S603）。

然后，在将内部变量cnt初始化成0（步骤S604）后，从masked_data取得新的数据值并保存在data_buf中（步骤S605），进而将从num输入的数据连续个数（在这里为1）加到cnt上（步骤S606）。这样若下一数据接收准备结束，则返回图3的步骤S306，进行数据输入结束的判断。

另外，在根据步骤S608内部变量cnt的值为1的情况下，将在步骤S601中输出的数据值作为“相同数据不连续”部分的最初的数据进行处理。为此从masked_data取得新的数据值并保存在data_buf中（步骤S605），进而将从num输入的数据连续个数（在这里为1）加到cnt上（步骤S606）。这样若下一数据接收准备结束，则返回图3的步骤S306。

图7是表示数据结合部中，内部变量continue为1且从masked_data输入的数据值与内部变量data_buf的值相同的情况下（图3中的步骤S310为yes）的处理的流程图。

在此情况下，由于从masked_data输入的数据值与data_buf的值相同，所以“相同数据连续”部分进一步持续。因此为求出数据连续的个数而将从num输入的数据连续个数加在内部变量cnt上（步骤S701）。

在本实施方式中，由于用8位来表示压缩位编码数据，故“相同数据连续”部分以及“相同数据不连续”部分的数，如在图5的说明中所说那样只能表示到128，因此在内部变量cnt的值为129以上的情况下，就需要输出“相同数据连续”部分持续128次这样的信息。为此参照内部变量cnt的值（步骤S702），在该值为129以上的情况

下从 v_out 输出保存在 data_buf 中的数据值（步骤 S703），跟随其从 v_out 输出作为表示“相同数据连续 128 个”的长度信息的 129（81h）（步骤 S704）。此外，该“129” 长度信息为了保持在现有技术中说明过的表，既可以参照其来求出，也可以根据表示相同数据持续 n 个的长度信息为

256 - n + 1 来求出。若通过步骤 S704 输出长度信息，则数据结合部在从内部变量 cnt 减去在后级作为压缩位编码数据输出的“128”（步骤 S705），并进行了下一数据的接收准备以后，返回图 3 的步骤 S306。

另外，在根据步骤 S702 内部变量 cnt 为 128 以下的情况下，由于压缩位编码所需要的数据未被生成，故直接返回图 3 的步骤 S306，进行数据输入结束的判断。

此外，在图 7 的流程图中由于压缩位编码所需要的数据值已经保存在 data_buf 中，故不需要将从 masked_data 输入的数据值保存在内部变量 data_buf 中，但也可以进行保存。

图 8 表示通过图 3 的步骤 S306 从 end（118）输入 1，判断为数据的输入已结束情况下的流程图。

在图 8 中首先从 v_out 输出保存在内部变量 data_buf 中的数据值（步骤 S801），然后参照内部变量 continue 的值（步骤 S802）。

在 continue 的值为 0 的情况下，由于在步骤 S801 中输出的数据值是“相同数据不连续”部分的最后的数据值，故从内部变量 cnt 生成“不同数据连续”部分的长度信息并输出。具体来讲就是从内部变量 cnt 的值减去 1 并从 v_out 输出（步骤 S803）。

然后为了向后级告知数据的输出结束而将 1 输出给 v_end（134）（步骤 S804），结束动作。

另一方面当在步骤 S802 中 continue 为 1 的情况下，由于在步骤 S801 中输出的数据值是“相同数据连续”部分的数据值，故从内部变量 cnt 生成“相同数据连续”部分的长度信息并输出。具体来讲就是使用内部变量 cnt 的值，求出

256 - cnt + 1

的值，并从 v_out 输出（步骤 S805）。

然后为了向后级告知数据的输出结束而将 1 输出给 v_end（134）（步骤 S804），结束动作。

此外，在本实施方式的数据结合部 103 中，在图 3 至图 8 中从 v_out 输出保存在内部变量 data_buf 中的数据值的情况下同时从 d_valid（131）输出 1，另外在从 v_out 输出由内部变量 cnt 生成的长度信息的情况下同时从 l_valid（132）输出 1。由此将从 v_out（133）输出的数据的有效信号、以及从 v_out 输出的数据是压缩位编码数据的数据部还是长度信息的区别通知给后级。

另外，在图 3～图 8 中，在如图 6 的步骤 S603 或图 7 的步骤 S704 那样，必须通过从 num 和 data 取得数据连续个数和数据值，来从 v_out 输出数据部和长度信息这两个数据的情况下，需要通过直到来自 v_out 的输出结束将 ready（117）设成 0（即，不进行下一数据的输入请求）使来自数据结合部 101 和图像处理部 102 的数据连续个数和数据输入停止。

图 11 表示实施方式中的数据结合部 103 的输入输出时间表的一例。

在图 11 中，clk 表示时钟，各信号同步于 clk 进行输入输出。valid、num、masked_data、ready 以及 end 信号是用于将由数据分割部所生成的数据连接数以及由数据处理部 102 所生成的数据值输入到数据结合部 103 的信号。

cnt、data_buf 以及 continue 表示在数据结合部内使用的内部变量，d_valid、l_valid、v_out 以及 v_end 信号是用于将压缩位编码数据的数据部和长度信息输出到数据输出部 104 的信号。

此外，在图 11 中从 v_out 输出的压缩位编码数据之中，对表示长度信息的部分画上双下划线来表示。

在图 11 中若在定时 t3 从 num 和 masked_data 输入最初的数据连续个数和数据值（步骤 S301），则在数据结合部 103 中将从 num 输

入的 4（04h）保存到内部变量 cnt，将从 masked_data 输入的 0（00h）保存在内部变量 data_buf 中（步骤 S302）。

然后通过步骤 S303 参照 cnt 的值，由于值不是 1 故通过步骤 S304 将内部变量 continue 设定成 1。

然后通过步骤 S306 进行数据输入结束的判断，由于没有从 end 输入 1 故等待再次从 num 和 masked_data 输入数据连续个数和数据值。

若在定时 t5 从 num 输入 9（09h）、从 masked_data 输入 0（00h）（步骤 S307），则数据结合部 103 通过步骤 S308 参照内部变量 continue 的值，由于该值为 1 故进入步骤 S310，比较从 masked_data 输入的数据值和保存在 data_buf 中的值。在定时 t3 由于 masked_data 的数据值和 data_buf 的值都为 0（00h）即相等，故数据结合部的动作进入图 7 的步骤 S701。

通过步骤 S701 对内部变量 cnt 的值和从 num 输入的数据连续个数进行相加，并将相加后的 cnt 的值与 129 进行比较（步骤 S702）。由于相加后的结果，内部变量 cnt 的值如图 11 的定时 t6 所示那样是 13（0Dh），比 129 还要小故返回图 3 的步骤 S306。

在定时 t6 再次通过步骤 S306 进行数据输入结束的判断，由于没有从 end 输入 1 故等待再次从 num 和 masked_data 输入数据连续个数和数据值。

若在定时 t7 从 num 输入 1（01h）、从 masked_data 输入 2（02h）（步骤 S307），则数据结合部 103 通过步骤 S308 参照内部变量 continue 的值，由于该值为 1 故进入步骤 S310，比较从 masked_data 输入的数据值和保存在 data_buf 中的值。在定时 t7 由于 masked_data 的数据值和 data_buf 的值不同，故数据结合部的动作进入图 6 的步骤 S601。

在步骤 S601 将 d_valid 设成 1，同时从 v_out 输出保存在 data_buf 中的值以后，参照从 num 输入的值（步骤 S602），由于该值是 1 故通过步骤 S607 将内部变量 continue 设定成 0 进行“相同数据不连续”部分开始的准备。

另外，通过步骤 S608 参照内部变量 cnt 的值，由于不是 1（01h）故通过步骤 S603

256 - 13 + 1

来计算长度信息将 l_valid 设成 1，同时从 v_out 输出 244（F4h）。然后将内部变量 cnt 初始化成 0（步骤 S604），将作为从 masked_data 输入的数据值的 2（02h）保存在 data_buf 中（步骤 S605），进而在 cnt 上相加从 num 输入的数据连续个数“1（01h）”，将 cnt 的值设成 1（01h）返回图 3 的步骤 S306。

此外，上述所说明的动作分别在定时 t7 进行步骤 S601 的 data_buf 的值的输出，在定时 t7 利用步骤 S607 进行将 0 设定到内部变量 continue 的动作，在定时 t8 进行步骤 S603 的长度信息的输出，在定时 t8 进行步骤 S605 和步骤 S606 中的数据值的取得以及数据连续个数的相加。

另外，关于在定时 t7 从 num 和 masked_data 输入的数据连续个数和数据值，由于需要从数据结合部的 v_out 输出数据部和长度信息这两个数据，故在定时 t7 从数据结合部的 ready 输出 1，将在定时 t7 从 num 和 masked_data 输入的值保持到定时 t8，在定时 t8 从 num 和 masked_data 取得所需要的值。

若作为下一数据在定时 t9 从 num 输入 1（01h），从 masked_data 输入 2（02h），则数据结合部通过步骤 S308 参照 continue 值。由于定时 t8 中的 continue 值为 0，故通过步骤 S309 比较从 masked_data 输入的值 2 和 data_buf 的值。由于定时 t8 中的 data_buf 的值为 2 故数据结合部的动作进入图 5 的步骤 S501。

通过步骤 S501 将内部变量 continue 的值设定成表示“相同数据连续”部分的 1，通过步骤 S502 参照内部变量 cnt 的值。由于此时的 cnt 的值为 1 故进入步骤 S506 对 cnt 的值和从 num 输入的数据连续个数进行相加。由于定时 t9 中的 num 的值为 1（01h）故 cnt 的值成为 2（02h）。

接着在步骤 S507 中参照 cnt 的值，由于此时的 cnt 的值如上所示

那样是 2，不足 129 故直接返回图 3 的步骤 S306。

由于定时 t10 和定时 t11 的动作，所输入的值和动作与定时 t9 的情况相同故在这里省略说明，通过定时 t10 和定时 t11 的动作内部变量 cnt 的值分别在定时 t11 成为 3（03h），在定时 t12 成为 4（04h），continue 的值仍为 1 不变。

若在定时 t12 中从 num 输入 1(01h)从 masked_data 输入 4(04h)，则数据结合部从步骤 S307 进入步骤 S308，由于内部变量 continue 为 1 故进入步骤 S310。于是对作为从 masked_data 输入的值的 4 和作为保存在 data_buf 中的值的 2 进行比较，由于值不同故进入图 6 的步骤 S601。

通过步骤 S601 从 v_out 输出作为 data_buf 的值的 2（02h），由于从 num 输入的数据连续个数为 1 故从步骤 S602 进入步骤 S607。这里将 continue 设定成 0 以表示“相同数据不连续”，并通过步骤 S608 参照 cnt 的值。由于定时 t12 中的 cnt 的值为 4（04h），故进入步骤 S603 从 cnt 的值计算出表示“相同数据连续”数的长度信息并从 v_out 输出，当在步骤 S604 中对 cnt 进行了初始化以后，将作为 masked_data 的数据值的 4（04h）保存在 data_buf 中，并在初始化后的 cnt 上相加从 num 输入的数据连续个数 1（01h）返回图 3 的步骤 S306。

此外，上述所说明的动作分别在定时 t12 进行步骤 S601 的 data_buf 的值的输出，在定时 t12 利用步骤 S607 进行将 0 设定到内部变量 continue 的动作，在定时 t13 进行步骤 S603 的长度信息的输出，在定时 t13 进行步骤 S605 和步骤 S606 中的数据值的取得以及数据连续个数的相加。

另外，关于在定时 t12 从 num 和 masked_data 输入的数据连续个数和数据值，由于需要从数据结合部的 v_out 输出数据部和长度信息这两个数据，故在定时 t12 从数据结合部的 ready 输出 1，将在定时 t12 从 num 和 masked_data 输入的值保持到定时 t13，在定时 t13 从 num 和 masked_data 取得所需要的值。

若在定时 t14 中从 num 输入 1(01h)从 masked_data 输入 2(02h)，

则数据结合部使处理从步骤 S307 进入步骤 S308，由于内部变量 continue 为 0 故进入步骤 S309。于是对作为从 masked_data 输入的值的 2 和作为保存在 data_buf 中的值的 4 进行比较，由于值不同故进入图 4 的步骤 S401。

通过步骤 S401 从 v_out 输出作为 data_buf 的值的 4（04h），由于从 num 输入的数据连续个数为 1 故从步骤 S402 进入步骤 S408。由于这里的内部变量 cnt 的值为 1(01h)，故从步骤 S408 进入步骤 S406，将 num 的值相加在内部变量 cnt 上（由此 cnt 成为 2）。然后在步骤 S407 中将从 masked_data 输入的值保存在 data_buf 以后，返回图 3 的步骤 S306。

在定时 t15，数据结合部进行与定时 t14 同样的动作。由此从 v_out 输出 2（02h），内部变量 cnt 的值被加 1 而成为 3，data_buf 的值成为从 masked_data 输入的 0（00h）。

若在定时 t17 从 num 输入 4(04h)，从 masked_data 输入 0(00h)，则数据分割部从步骤 S307 进入步骤 S308，由于内部变量 continue 为 0 故进入步骤 S309。于是对作为从 masked_data 输入的值的 0 和作为保存在 data_buf 中的值的 0 进行比较，由于值相同故进入图 5 的步骤 S501。

通过步骤 S501 将内部变量 continue 的值设定成表示“相同数据连续”部分的 1，通过步骤 S502 参照内部变量 cnt 的值。由于此时的 cnt 的值为 3 故进入步骤 S503，从 cnt 值计算出表示“不同数据连续”数的长度信息。具体来讲就是将从作为 cnt 值的 3 减去 2 之后的值 1 作为长度信息从 v_out 输出。

然后通过步骤 S503 将 1 设定到 cnt，通过步骤 S504 将作为从 num 输入的数据连续个数的 4 加到 cnt 值上而成为 5（05h），返回图 3 的步骤 S306。

若在定时 t18 从 end（118）输入 1，则由数据结合部在步骤 S306 中判断为数据输入已结束，并为了用于结束动作的处理而进入图 8 的步骤 S801。

当在步骤 S801 中从 v_out 输出保存在 data_buf 中的数据值以后，通过步骤 S802 参照内部变量 continue 的值。由于定时 t18 中的 continue 的值为 1，则进入步骤 S805 从 cnt 的值生成表示“相同数据连续”部分的长度信息。具体来讲就是在下式中使用作为 cnt 值的 5（05h）作为长度信息计算出 252（FCh）并从 v_out 输出。

256 - cnt + 1

然后，为了向后级告知数据的输出已结束而在步骤 S804 中从 v_end 输出 1 并停止动作。

此外，上述所说明的动作分别在定时 t18 进行步骤 S801 的 data_buf 的值的输出，在定时 t19 利用步骤 S805 进行长度信息的输出，在定时 t20 进行步骤 S804 的从 v_end 输出 1 的处理。

另外，关于在定时 t18 从 end 输入了 1 的状态，由于需要从数据结合部的 v_out 输出数据部和长度信息这两个数据，故在定时 t18 从数据结合部的 ready 输出 1，将在定时 t18 所输入的 end 的状态保持到定时 t19，由此数据结合部对数据输入的结束进行确认。

通过上述说明过的动作，实施方式中的数据结合部 103 与作为通常的压缩位编码数据的排列顺序的“连续的个数 + 数据值”或者“不同数据连续的个数 + 不同数据流”相反，以“数值 + 连续的个数”或者“不同数据流 + 不同数据连续的个数”这样的顺序输出压缩位编码数据。

数据输出部 104，重新排列从数据结合部 103 输出的编码数据，替换长度信息和数据部的排列顺序以成为通常的压缩位编码数据的顺序，并进行输出。

若作为实施方式中的数据输出部 104 进行的处理过程来表示，就成为图 9 的流程图。

若数据输出部开始动作，则进行内部变量的初始化（步骤 S901）。本实施方式中的数据输出部中在作为内部变量，存在有：表示保存从 v_out 输入的长度信息的缓冲区的 L_locate，表示同样从 v_out 输入的数据部的数据值的缓冲器的 D_locate，用于保存长度信息和数据部的

数据值的 16 个 8 位数据缓冲区 data_buf[15] ~ data_buf[0]，表示输出 data_buf[15] ~ data_buf[8]的内容时输出目的地的地址的地址缓冲区 addr_buf[1]以及表示输出 data_buf[7] ~ data_buf[0]的内容时输出目的地的地址的地址缓冲区 addr_buf[0]。

在步骤 S901 中，将这些内部变量设定成，

L_locate = 0

D_locate = 1

addr_buf[0] = 0

addr_buf[1] = 1

data_buf[15] ~ data_buf[0] = 0。

若内部变量的初始化结束，则数据输出部判断来自数据结合部的数据的输入是否结束（步骤 S902）。数据的输入结束的判断，在本实施方式中通过是否从 v_end 输入 1 来进行判断，若输入 1 则设数据的输入结束。

若在步骤 S902 中判断为数据输入没有结束（v_end = 0），则数据输出部等待从 v_out 输入数据。在本实施方式中由于在从 v_out 输入长度信息的情况下从 l_valid 输入 1，在输入数据部的数据值的情况下从 d_valid 输入 1，故具体来讲就是数据输出部通过参照 l_valid 的值（步骤 S903）和参照 d_valid 的值（步骤 S904）来等待来自 v_out 的数据输入。然后反复图 9 的步骤 S902 至 S904，直到从 v_end 输入 1 数据输入结束，或者直到 l_valid 或 d_valid 变成 1 从 v_out 输入数据。

若通过步骤 S903 检测到从 l_valid 输入 1，则数据输出部将从 v_out 输入的长度信息保存在 L_locate 表示的编号的 data_buf（下面表示为 data_buf[L_locate]）中（步骤 S905），将 D_locate 所示的值作为新的长度信息保存位置保存在 L_locate 中，D_locate 在 D_locate 所保存的值上加 1（步骤 S906）。

另外，在步骤 S907 中参照在步骤 S906 中所设定的 L_locate 的值。在本实施方式中由于在比 L_locate 的值还小的编号的数据缓冲区中已

经保存了数据，故在 L_locate 的值比 7 还大的情况下就在 data_buf[7]~data_buf[0]的全部数据缓冲区中保存有值。

为此从address输出表示data_buf[7]~data_buf[0]的保存目的地的地址缓冲区 addr_buf[0]的值，同时从 reencode_data 输出 data_buf[7]~data_buf[0]的值（步骤 S908）。

然后将保存在 addr_buf[1]中的地址作为新地址保存在 addr_buf[0]中，addr_buf[1]在 addr_buf[0]中所保存的值上加 1。另外将保存在 data_buf[15]~data_buf[8]中的值代入 data_buf[7]~data_buf[0]，data_buf[15]~data_buf[8]初始化成 0。进而分别从 D_locate 和 L_locate 的值减去 8（步骤 S909）。

由此由于下一数据的取得准备已结束故数据输出部返回步骤 S902 等待从 v_end 输入 1 或者从 v_out 输入新的数据。

此外，当在步骤 S907 中所参照的 L_locate 的值为 7 以下的情况下，直接返回步骤 S902，等待从 v_end 输入 1 或者从 v_out 输入新的数据。

另外若在步骤 S903 中检测出 l_valid 的值为 0，且在步骤 S904 中检测出从 d_valid 输入 1，则数据输出部将从 v_out 输入的数据部的数据值保存在 D_locate 表示的编号的 data_buf（下面表示为 data_buf[D_locate]）中（步骤 S910），在 D_locate 所保存的值上加 1（步骤 S911）。

另外，参照在步骤 S911 中所设定的 D_locate 的值（步骤 S912）。在本实施方式中由于步骤 S902、S903、S904 中的 L_locate 的值只取 7 以下的值，故在 D_locate 的值比 15 还大的情况下就在 data_buf[15]~data_buf[8]的全部数据缓冲区中保存有值。

为此从 address 输出表示 data_buf[15]~data_buf[8]的保存目的地的地址缓冲区 addr_buf[1]的值，同时从 reencode_data 输出 data_buf[15]~data_buf[8]的值（步骤 S913）。

然后在保存于 addr_buf[1]中的值上加 1，将 data_buf[15]~data_buf[8]的内容初始化成 0 同时从 D_locate 的值减去 8（步骤 S914）。

由此由于下一数据的取得准备已结束故数据输出部返回步骤S902等待从v_end输入1或者从v_out输入新的数据。

此外，当在步骤S912中所参照的D_locate的值为15以下的情况下，直接返回步骤S902，等待从v_end输入1或者从v_out输入新的数据。

另外，若在步骤S902中从v_end输入1数据输入结束，则数据输出部参照L_locate的值（步骤S915）。在L_locate的值为0以外的情况下，由于在数据缓冲区data_buf[7]~data_buf[0]的一些缓冲区中存在未输出的数据，故从address输出表示data_buf[7]~data_buf[0]的保存目的地的地址缓冲区addr_buf[0]的值，同时从reencode_data输出data_buf[7]~data_buf[0]的值（步骤S916）。

然后，为了向后级告知数据的输出结束而从reencode_end输出1（步骤S917）并结束动作。

另外当在步骤S915中L_locate的值为0的情况下，由于在数据缓冲区中没有未被输出的数据，故进入步骤S917，在从reencode_end输出1告知数据的输出结束以后，结束动作。

此外，在上述说明中的步骤S909和S914中附加以下处理，在从reencode_data输出数据，或者从address输出地址时将reencode_valid设成1。

图12、图13是表示在图11所示的定时d_valid、l_valid、v_out以及v_end被输入本实施方式中的数据输出部104的情况下数据输出部的动作的图，并示出内部变量以及输出信号的值的转变。

处于图12中开头的是，图11的定时t0中的数据输出部的状态。在该图同一处，1201表示图11中的时间（定时），1202表示在1201所示的时间向数据输出部的输入值。另外，1202表示1201所示的时间中的内部变量的值，1203表示同样在1201所示的时间中的数据输出部的输出值。此外，在图12的定时t0表示在图9的步骤S901中对内部变量进行了初始化的状态。此外，data_buf[0]~data_buf[15]的值被设定成0，但在定时t0为了区别于从v_out输入的数据data_buf

的值用空白来表示。

在定时 t7，表示从 v_out 输入了数据部的数据值的状态。此时的数据输出部 104 通过步骤 S904 检测出 d_valid 为 1，由于 D_locate 的值为 1 故通过步骤 S910 将从 v_out 输入的值保存在 data_buf[1]中，同时通过步骤 S911 在 D_locate 的值上加 1 而成为 2。然后，通过步骤 S912 参照 D_locate 的值，由于为 15 以下故返回步骤 S902。在上述动作后，成为定时 t8 所示的状态。

图 12 的定时 t8 表示从 v_out 输入了长度信息的状态。此时，数据输出部 104 通过步骤检 S903 测出 l_valid 为 1，由于 L_locate 的值为 0 故通过步骤 S905 将从 v_out 输入的值保存在 data_buf[0]中，同时通过步骤 S906 将保存于 D_locate 的值 2 保存在 L_locate 中，并在 D_locate 的值 2 上加 1 而成为 3。然后，通过步骤 S907 参照 L_locate 的值，由于为 7 以下故返回步骤 S902。上述动作后的内部变量的状态，成为定时 t12 那样。

同样由于在定时 t12、t14、t15 以及图 13 中的定时 t18 从 d_valid 输入 1，故在 D_locate 的值所示的 data_buf 中保存从 v_out 输入的数据部的数据值，并使 D_locate 的值加 1。

另外，在定时 t17 由于从 l_valid 输入 1，故在 L_locate 的值所示的 data_buf 中保存从 v_out 输入的长度信息，将 D_locate 的值保存到 L_locate 并使 D_locate 的值加 1。

在定时 t19 中从 v_out 输入长度信息。由于此时的 L_locate 的值为 7，故通过步骤 S905 从 v_out 输入的值 FCh 被保存在 data_buf[7]中，然后通过步骤 S906 在 L_locate 上保存作为 D_locate 的值的 9，另外在 D_locate 中保存 10。

此时，若在步骤 S907 中参照 L_locate 则由于 L_locate 的值为 7 以上，故通过步骤 S908 从 address 输出作为 addr_buf[0]的值的 0，另外从 reencode_data 输出 data_buf[0] ~ data_buf[7]的内容。

然后，通过步骤 S909 将 addr_buf[0]的内容设定成保存在 addr_buf[1]中的 1，addr_buf[1]的值加 1 而成为 2。另外将保存在

addr_buf[8]～addr_buf[15]中的值设定到 addr_buf[0]～addr_buf[7]中，将 addr_buf[8]～addr_buf[15]初始化成 0（在图 12、图 13 内用空白来表示）。进而从 L_locate 和 D_locate 的值减去 8 将 L_locate 的值设成 1，将 D_locate 的值设成 2。

此外，上述步骤 S905～S909 的动作结果在图 13 的定时 t20 示出。另外，在上述说明中在从 address 和 reencode_data 输出地址和数据时还从 reencode_valid 输出 1。

在图 13 的定时 t20，表示从 v_end 输入 1 的状态。此时的数据输出部 104 通过步骤 S902 检测到 v_end 为 1，通过步骤 S905 参照 L_locate 的值。在定时 t20 由于 L_locate 的值是 1，故通过步骤 S916 输出剩余在 data_buf[0]～data_buf[7]中的数据。为此，从 reencode_valid 输出 1，同时从 address 输出作为 addr_buf[0]的值的 1，从 reencode_data 输出 data_buf[0]～data_buf[7]的值。

然后，通过步骤 S917 从 reencode_end 输出 1 并结束数据输出部的动作。

在上述说明中，步骤 S916 中的动作结果成为图 13 的定时 t21，步骤 S917 中的动作结果成为图 13 的定时 t22。

如以上所说明那样根据本实施方式，就可在压缩位编码装置中在输入压缩位编码数据，对压缩位编码数据的各数据部进行预定的处理以后再次进行压缩位编码处理（压缩位编码的重新构建）。在本实施方式中作为具体例子，在图 10 中从 encode_data 输入的 14 个压缩位编码数据，就能够通过由数据处理部将数据部的值的最低位设成 0，而如图 11 的 v_out 所示那样压缩成 9 个压缩位编码数据。

此外，在实施方式中，说明了对图像区信息进行压缩位编码的情况，但也可以将作为图像数据的各像素值作为编码对象。在此情况下，先前所示的溢出次数和进行屏蔽的位的关系，只要对从像素值的 LSB 向高位的 N 位（N＝溢出次数）进行屏蔽即可。

另外，虽然在实施方式中，以压缩位编码为例进行了说明，但由于也能够适用于用相同数据连续的个数及其数据、不同数据连续的个

数及其数据流所表达的编码，故本申请发明并不只限于上述实施方式。

<第 2 实施方式>

图 16 时本发明的压缩位再编码部的其他例子（第 2 实施方式）。

图 16 的压缩位编码装置，为了不仅可从 encode_data 输入压缩位编码数据还可输入进行压缩位编码以前的原始数据，而对数据分割部 1401 追加 raw_data 信号（1410）。

在本第 2 实施方式的压缩位编码装置中，若从 raw_data 输入 0，则数据分割部 1401 如图 2 所示那样进行从 encode_data（111）输入了压缩位编码数据情况下的动作，若从 raw_data 输入 1 则进行从 encode_data（111）输入了进行压缩位编码以前的原始数据情况下的动作。

若从 encode_data（111）输入原始数据，则所输入的数据从 data（115）输出到数据处理部 102，与此同时从 num（116）作为数据连续个数输出 1（01h）。

由此在数据结合部 103 和数据处理部 104 中，将在从数据分割部 1401 输出后，并由数据处理部 102 所处理的数据值作为“数据连续个数为 1”的数据进行压缩位编码处理。

根据上面的结构，图 16 的压缩位编码装置就可不论输入压缩位编码数据和原始数据的哪个，都生成并输出压缩位编码数据。

如以上所述那样根据本第 1、第 2 实施方式，可将压缩位编码数据分割成长度信息和数据部，对数据部的数据值进行处理，使用处理后的数据值和长度信息重新构建新的压缩位编码数据，而不用对已进行压缩位编码的编码数据进行解码处理，在进行再编码时就不需要很多的存储器，可进行高速的再编码。

另外，虽然在实施方式中，对适用于复印机的例子进行了说明，但本申请发明并不限定于此。

进而，虽然在实施方式中，示出在进行再编码时，决定进行屏蔽（设成 0）的位的位置的例子，但也可以通过设成 1 来应对。即，采

用正逻辑/负逻辑的哪个都可以。总而言之，在进行再编码时，向减少数据可取得的种类数目的方向进行位变更即可。这是因为，其结果就能够使相同数据连续的比例增高，并能够期待更高的压缩率。

另外，使用流程图对在实施方式中所说明的压缩位再编码部 15 具有的各个构成元件进行了说明。如从这样的说明可知那样，也可以用计算机程序来实现与其功能相同的功能。即本发明也将计算机程序作为其范畴。另外，由于通常计算机程序通过在装置中放置存储了它的 CDROM 等计算机可读取的存储介质，并拷贝或者安装到系统而变得可以执行，故可知本发明将这样的计算机可读取的存储介质也作为其范畴。

如以上所说明那样根据本实施方式，就可将如压缩位编码数据那样，以表示相同数据的连续个数的连续数代码部和表示上述数据的数据部，以及表示不同数据流的连续数的连续数代码部和表示上述不同数据流的数据部的数据形式所表达的编码数据，再编码成相同数据形式，使压缩率得以提高，而不用对其进行解码处理。

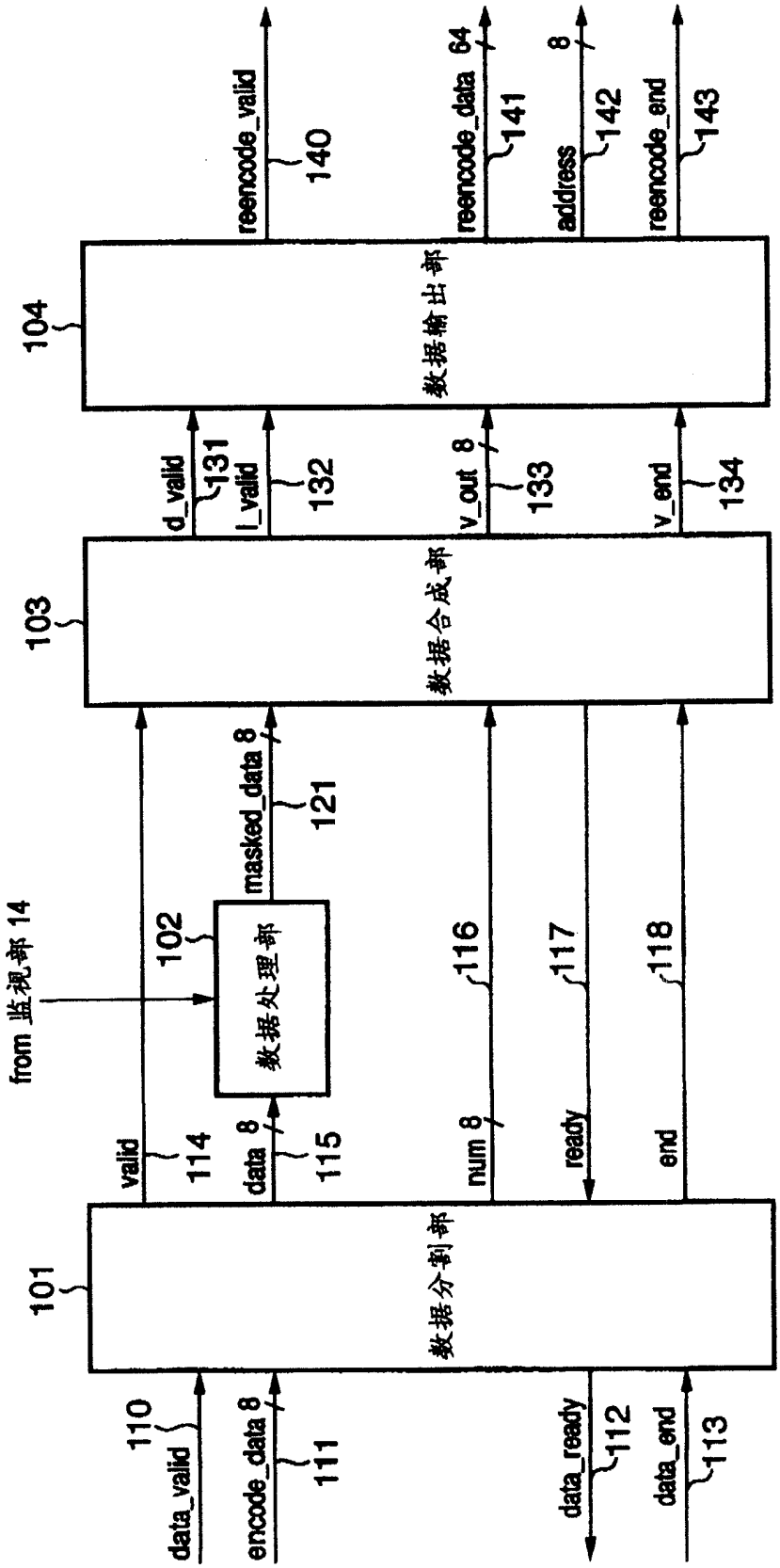

101
数据分割部
103
数据合成部
104
数据输出部
102
数据处理部
from 监视部 14
data_valid
110
encode_data 8
111
data_ready
112
data_end
113
valid
114
data 8
115
num 8
116
ready
117
end
118
masked_data 8
121
d_valid
131
l_valid
132
v_out 8
133
v_end
134
reencode_valid
140
reencode_data 64
141
address 8
142
reencode_end
143

图 1

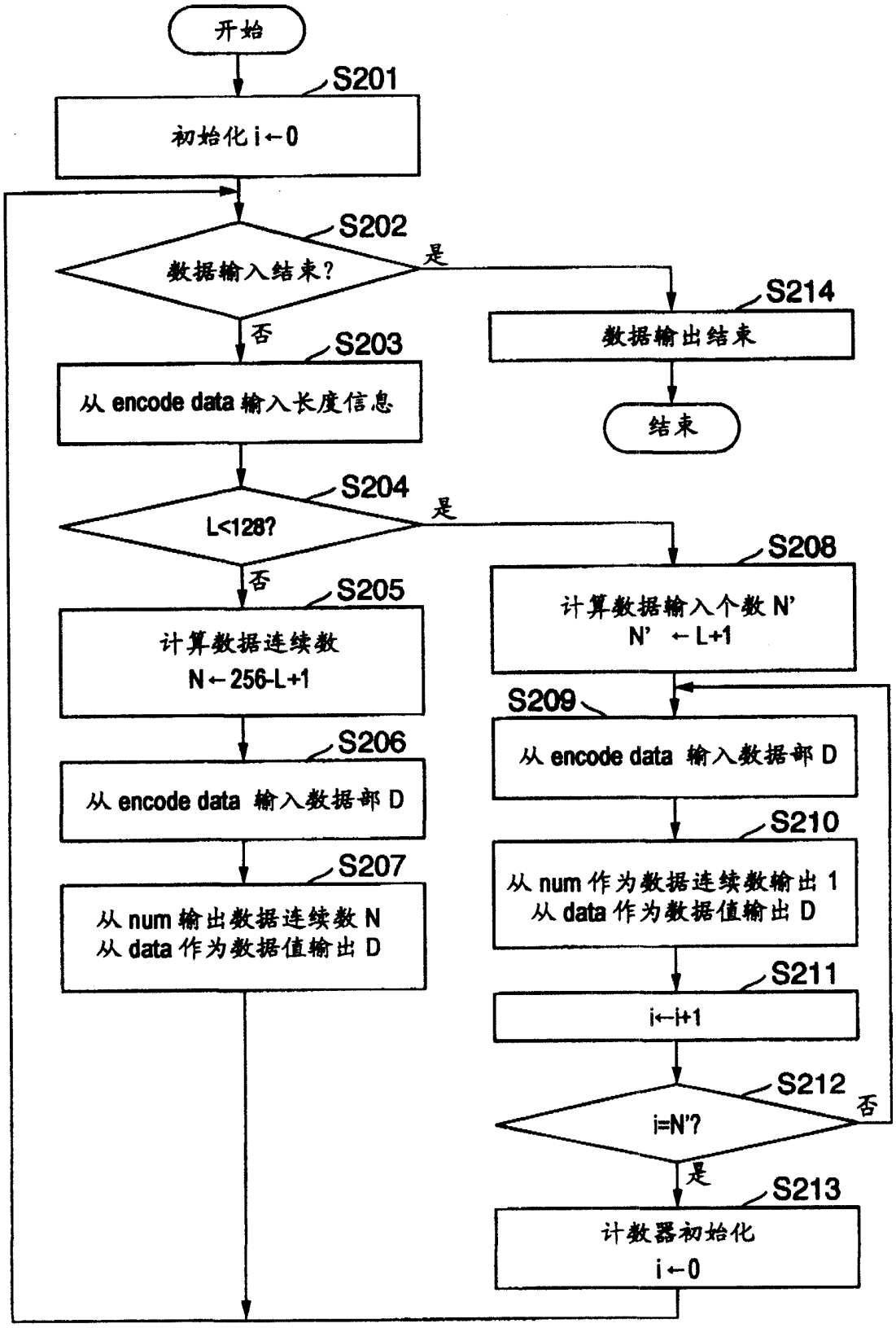

开始
S201
初始化 i←0
S202
数据输入结束?
是
否
S214
数据输出结束
结束
S203
从 encode data 输入长度信息
S204
L<128?
是
否
S205
计算数据连续数
N←256-L+1
S206
从 encode data 输入数据部 D
S207
从 num 输出数据连续数 N
从 data 作为数据值输出 D
S208
计算数据输入个数 N'
N' ←L+1
S209
从 encode data 输入数据部 D
S210
从 num 作为数据连续数输出 1
从 data 作为数据值输出 D
S211
i←i+1
S212
i=N'?
否
是
S213
计数器初始化
i←0

图 2

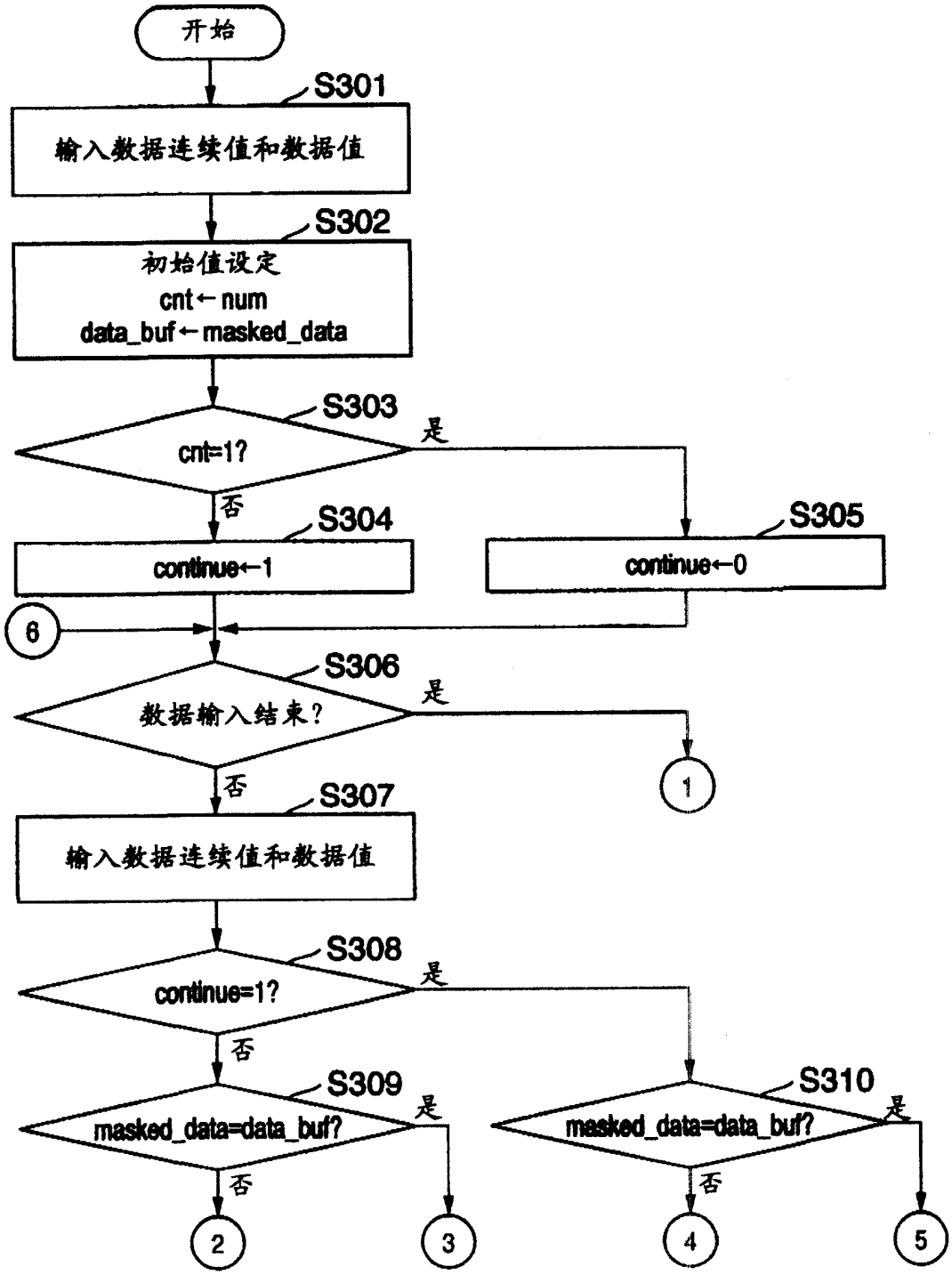

开始
S301
输入数据连续值和数据值
S302
初始值设定
cnt←num
data_buf←masked_data
S303
cnt=1?
是
否
S304
continue←1
S305
continue←0
6
S306
数据输入结束?
是
否
1
S307
输入数据连续值和数据值
S308
continue=1?
是
否
S309
masked_data=data_buf?
是
否
S310
masked_data=data_buf?
是
否
2
3
4
5

图 3

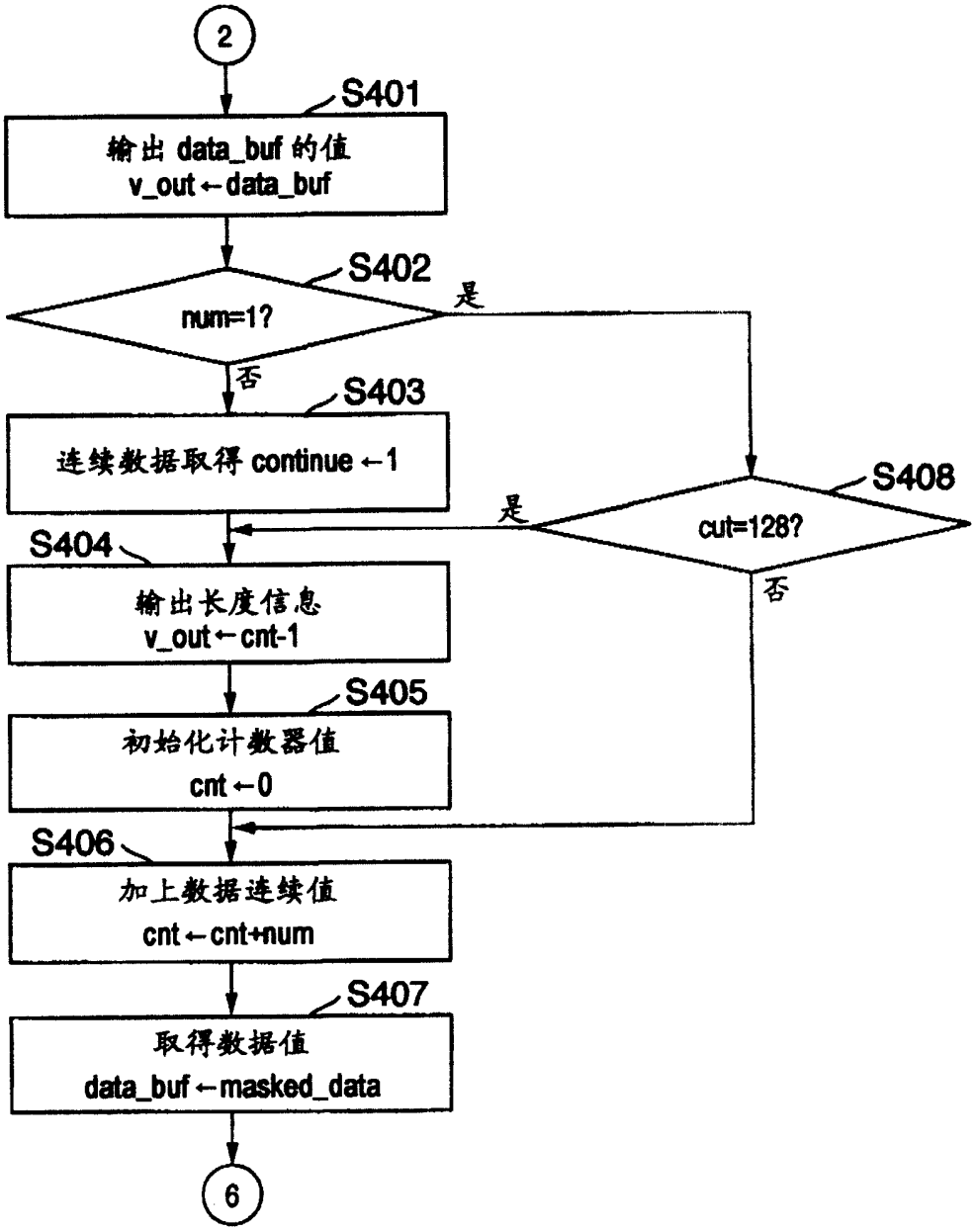

2
S401
输出 data_buf 的值
v_out←data_buf
S402
num=1?
是
否
S403
连续数据取得 continue←1
S408
是
cut=128?
否
S404
输出长度信息
v_out←cnt-1
S405
初始化计数器值
cnt←0
S406
加上数据连续值
cnt←cnt+num
S407
取得数据值
data_buf←masked_data
6

图 4

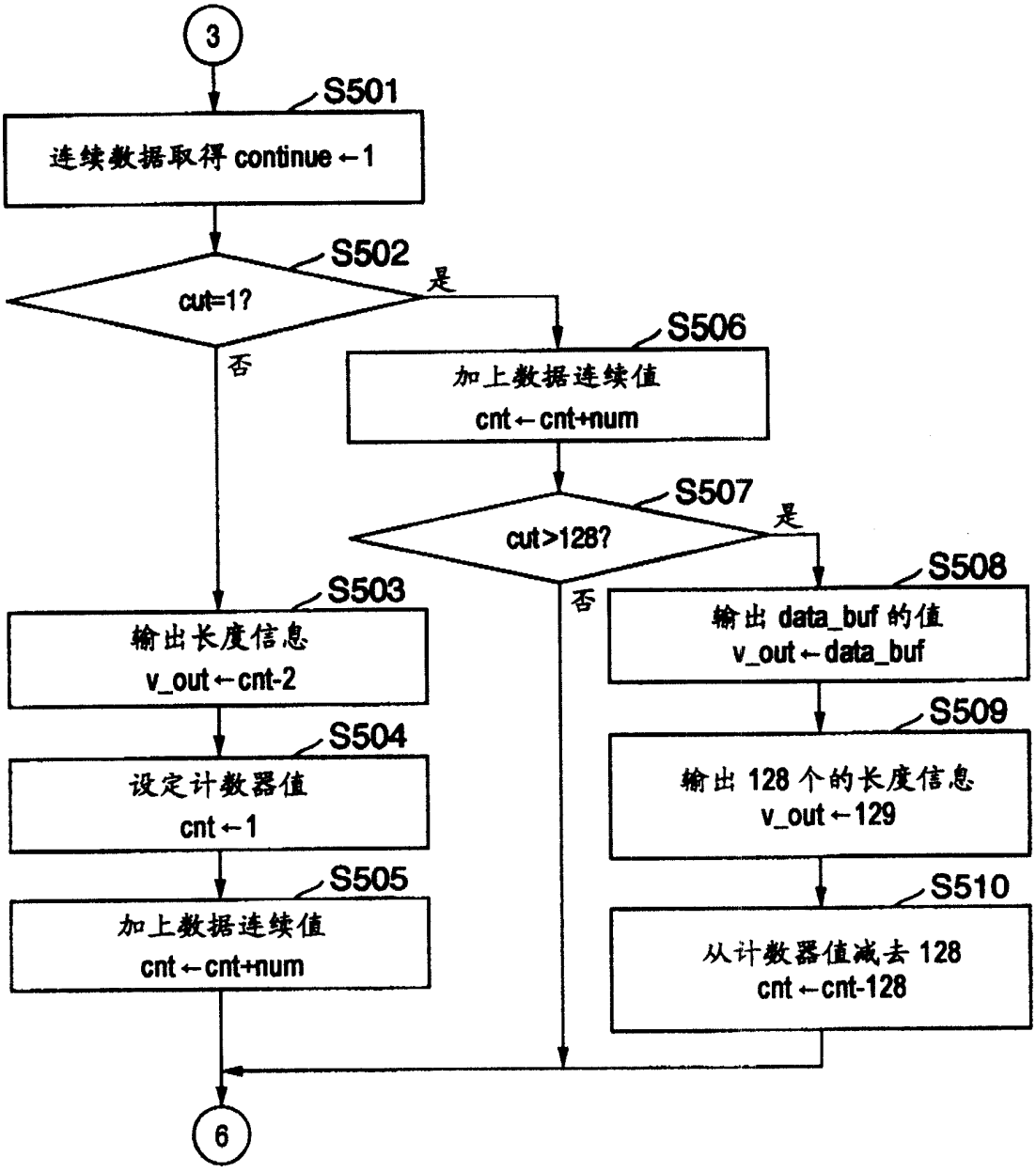

3
S501
连续数据取得 continue←1
S502
cut=1?
是
否
S506
加上数据连续值
cnt←cnt+num
S507
cut>128?
是
否
S503
输出长度信息
v_out←cnt-2
S504
设定计数器值
cnt←1
S505
加上数据连续值
cnt←cnt+num
S508
输出 data_buf 的值
v_out←data_buf
S509
输出 128 个的长度信息
v_out←129
S510
从计数器值减去 128
cnt←cnt-128
6

图 5

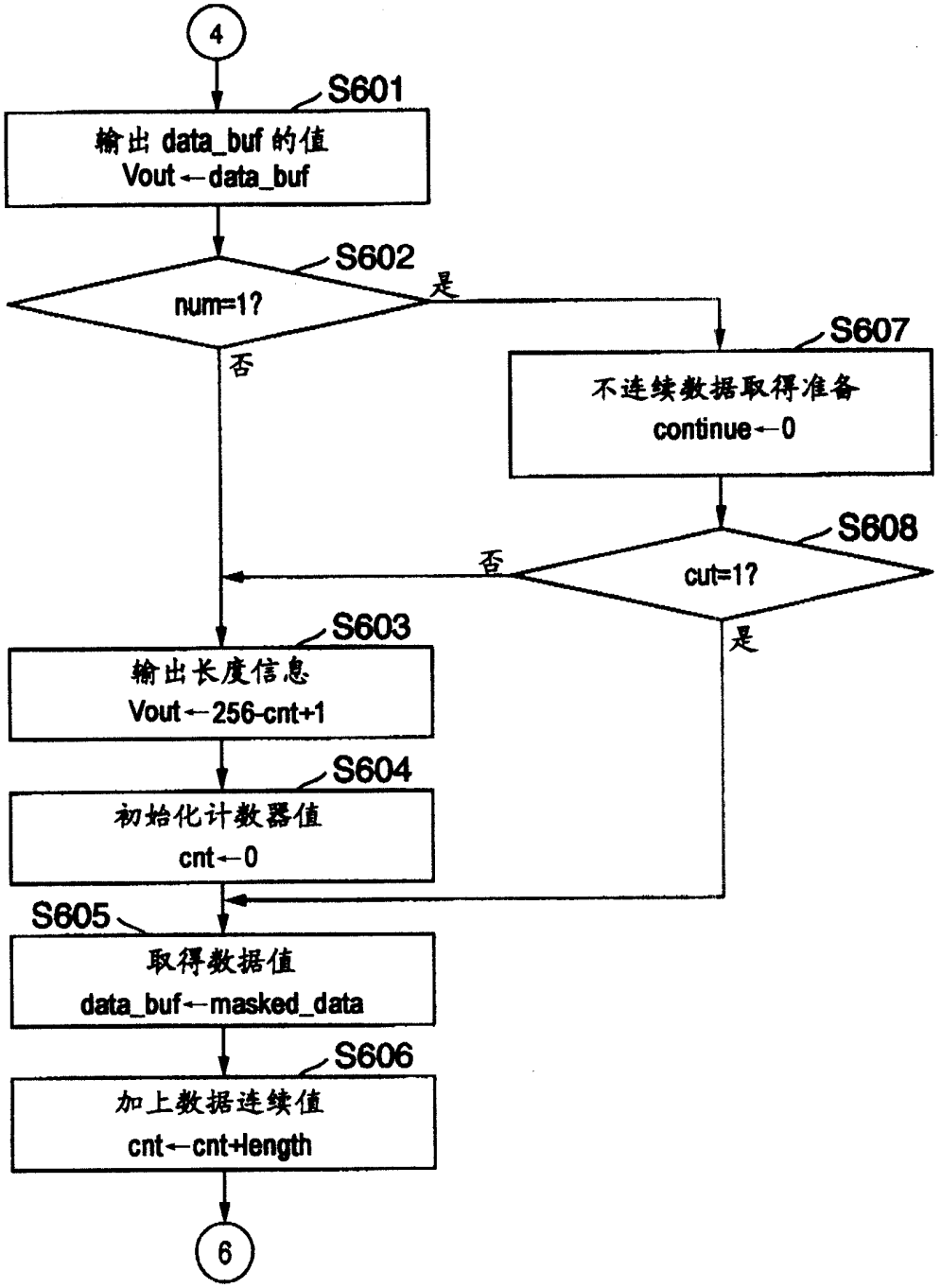

4
S601
输出 data_buf 的值
Vout←data_buf
S602
num=1?
是
否
S607
不连续数据取得准备
continue←0
S608
cut=1?
否
是
S603
输出长度信息
Vout←256-cnt+1
S604
初始化计数器值
cnt←0
S605
取得数据值
data_buf←masked_data
S606
加上数据连续值
cnt←cnt+length
6

图 6

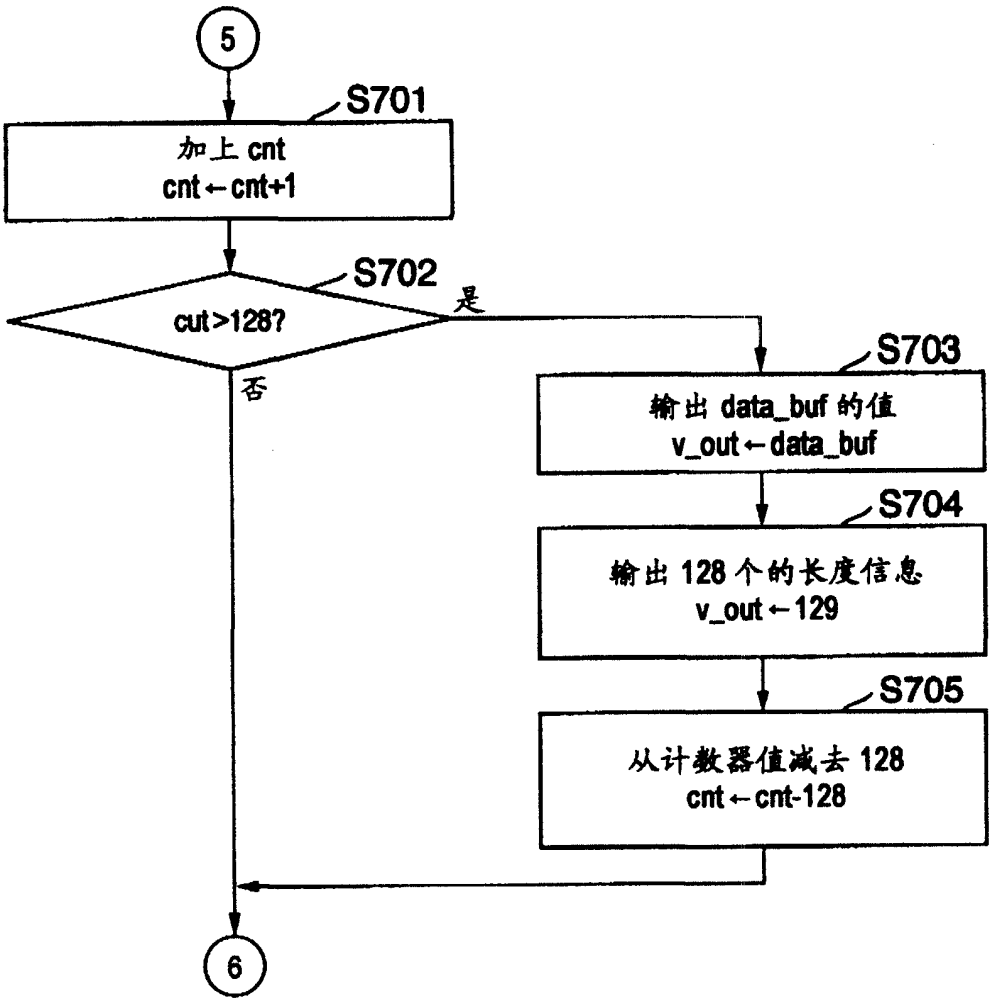

5
S701
加上 cnt
cnt←cnt+1
S702
cut>128?
是
否
S703
输出 data_buf 的值
v_out←data_buf
S704
输出 128 个的长度信息
v_out←129
S705
从计数器值减去 128
cnt←cnt-128
6

图 7

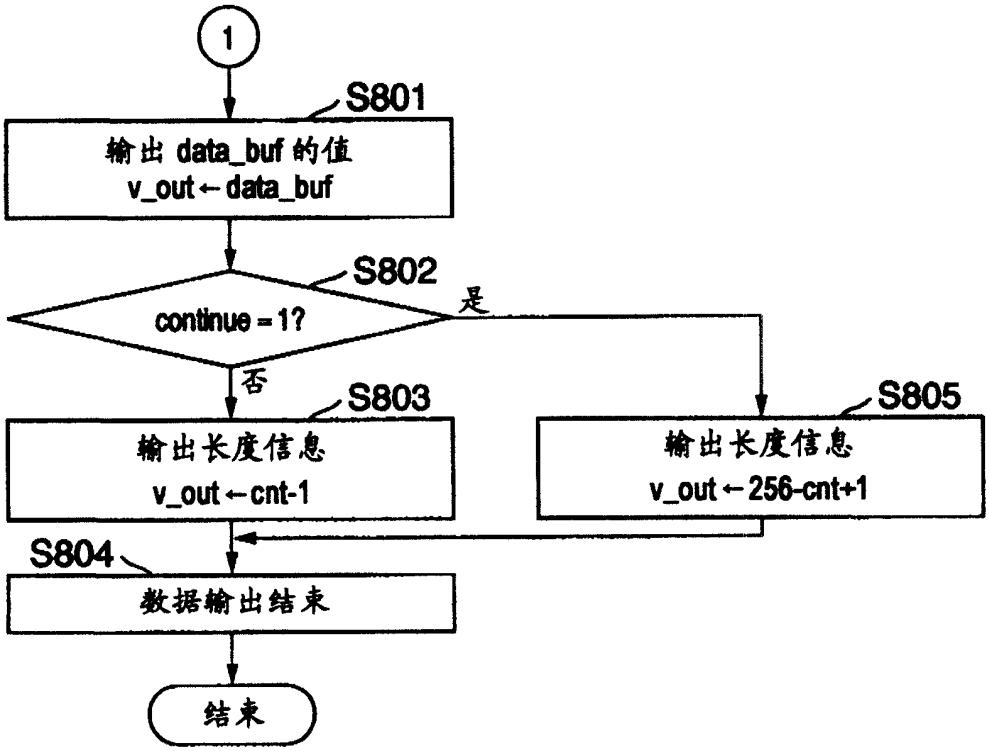

1
S801
输出 data_buf 的值
v_out←data_buf
S802
continue=1?
是
否
S803
输出长度信息
v_out←cnt-1
S805
输出长度信息
v_out←256-cnt+1
S804
数据输出结束
结束

图 8

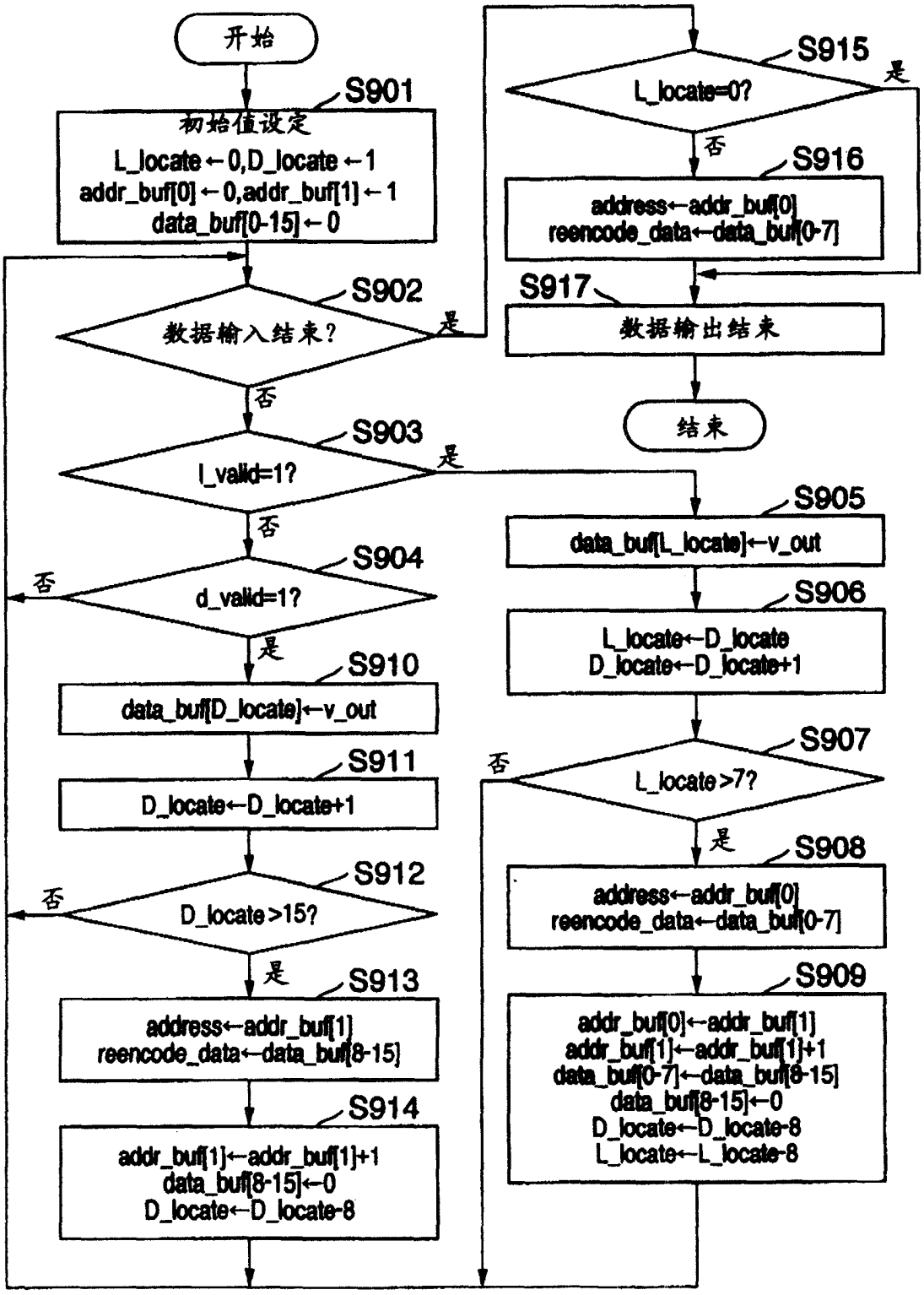

开始
S901
初始值设定
L_locate←0,D_locate←1
addr_buf[0]←0,addr_buf[1]←1
data_buf[0-15]←0
S902
数据输入结束?
是
否
S903
I_valid=1?
是
否
S904
d_valid=1?
否
是
S910
data_buf[D_locate]←v_out
S911
D_locate←D_locate+1
S912
D_locate>15?
否
是
S913
address←addr_buf[1]
reencode_data←data_buf[8-15]
S914
addr_buf[1]←addr_buf[1]+1
data_buf[8-15]←0
D_locate←D_locate-8
S915
L_locate=0?
是
否
S916
address←addr_buf[0]
reencode_data←data_buf[0-7]
S917
数据输出结束
结束
S905
data_buf[L_locate]←v_out
S906
L_locate←D_locate
D_locate←D_locate+1
S907
L_locate>7?
否
是
S908
address←addr_buf[0]
reencode_data←data_buf[0-7]
S909
addr_buf[0]←addr_buf[1]
addr_buf[1]←addr_buf[1]+1
data_buf[0-7]←data_buf[8-15]
data_buf[8-15]←0
D_locate←D_locate-8
L_locate←L_locate-8

图 9

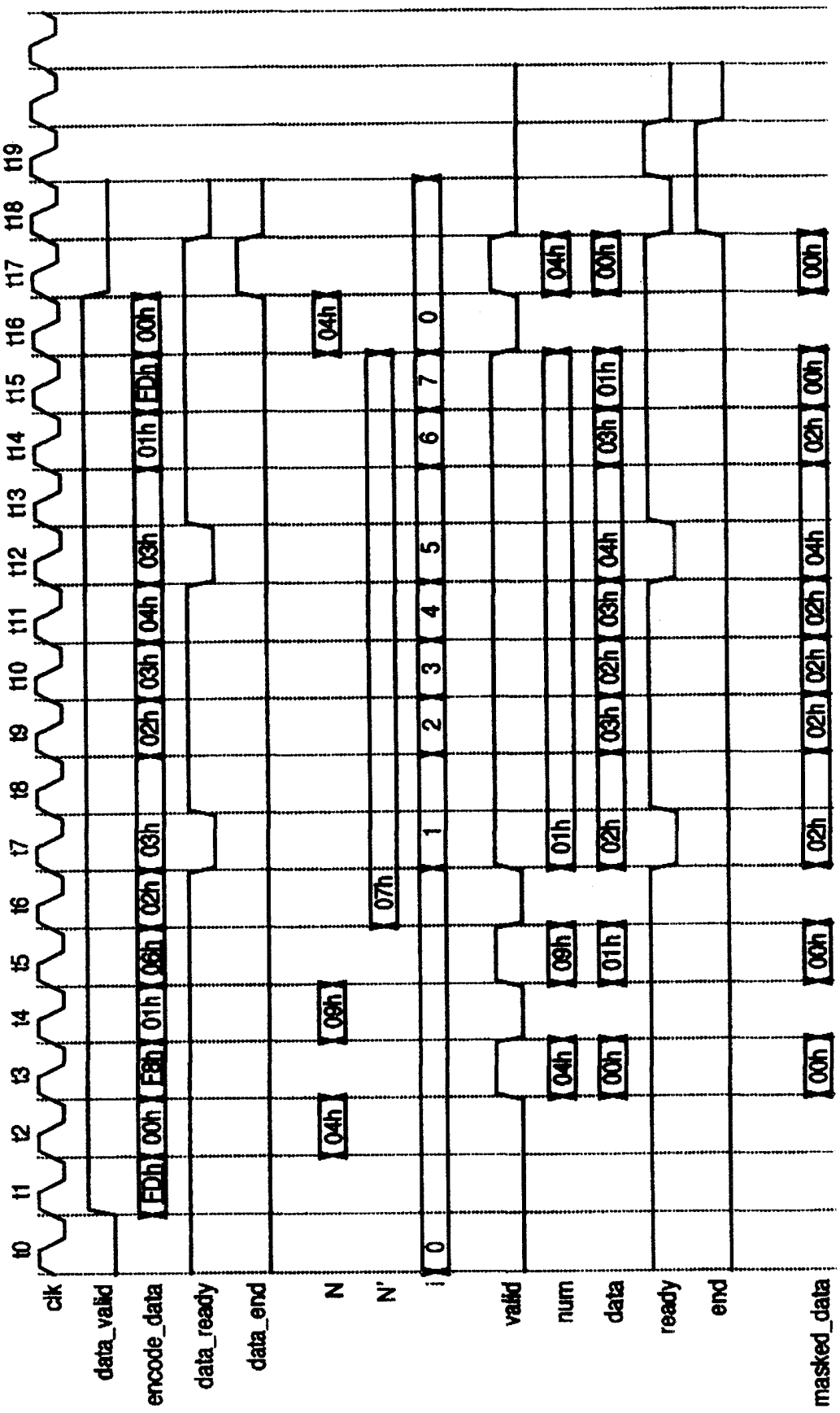

t0
t1
t2
t3
t4
t5
t6
t7
t8
t9
t10
t11
t12
t13
t14
t15
t16
t17
t18
t19
clk
data_valid
encode_data
data_ready
data_end
N
N'
i
valid
num
data
ready
end
masked_data
FDh
00h
F8h
01h
06h
02h
03h
04h
03h
02h
01h
FDh
00h
04h
09h
07h
0
1
2
3
4
5
6
7
01h
02h
03h
04h
09h
01h
00h
00h
00h
00h
02h
02h
02h
02h
04h
02h
00h

图 10

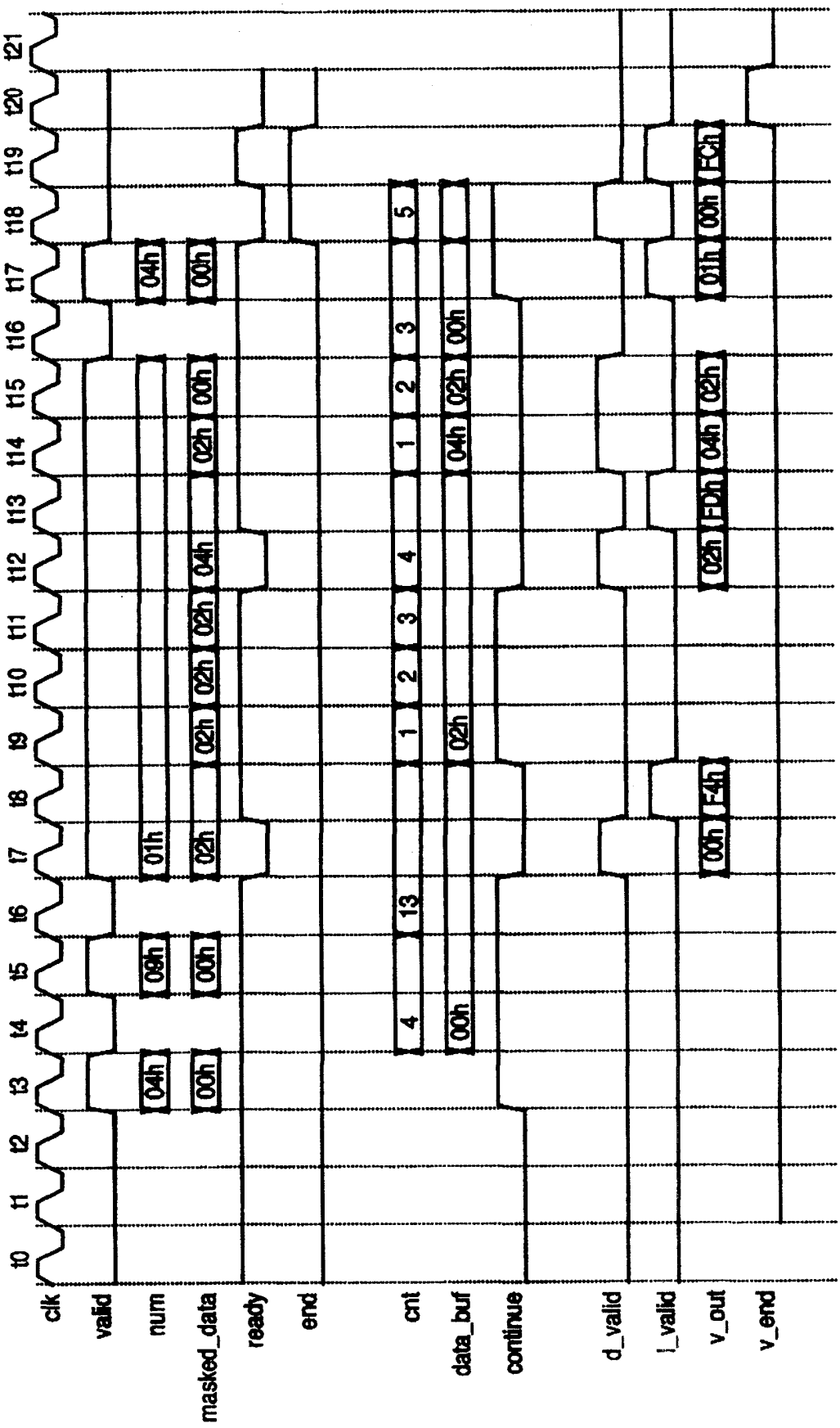

t0
t1
t2
t3
t4
t5
t6
t7
t8
t9
t10
t11
t12
t13
t14
t15
t16
t17
t18
t19
t20
t21
clk
valid
num
masked_data
ready
end
cnt
data_buf
continue
d_valid
l_valid
v_out
v_end
04h
00h
09h
01h
02h
13
4
5
3
1
00h
FCh
FDh
FEh

图 11

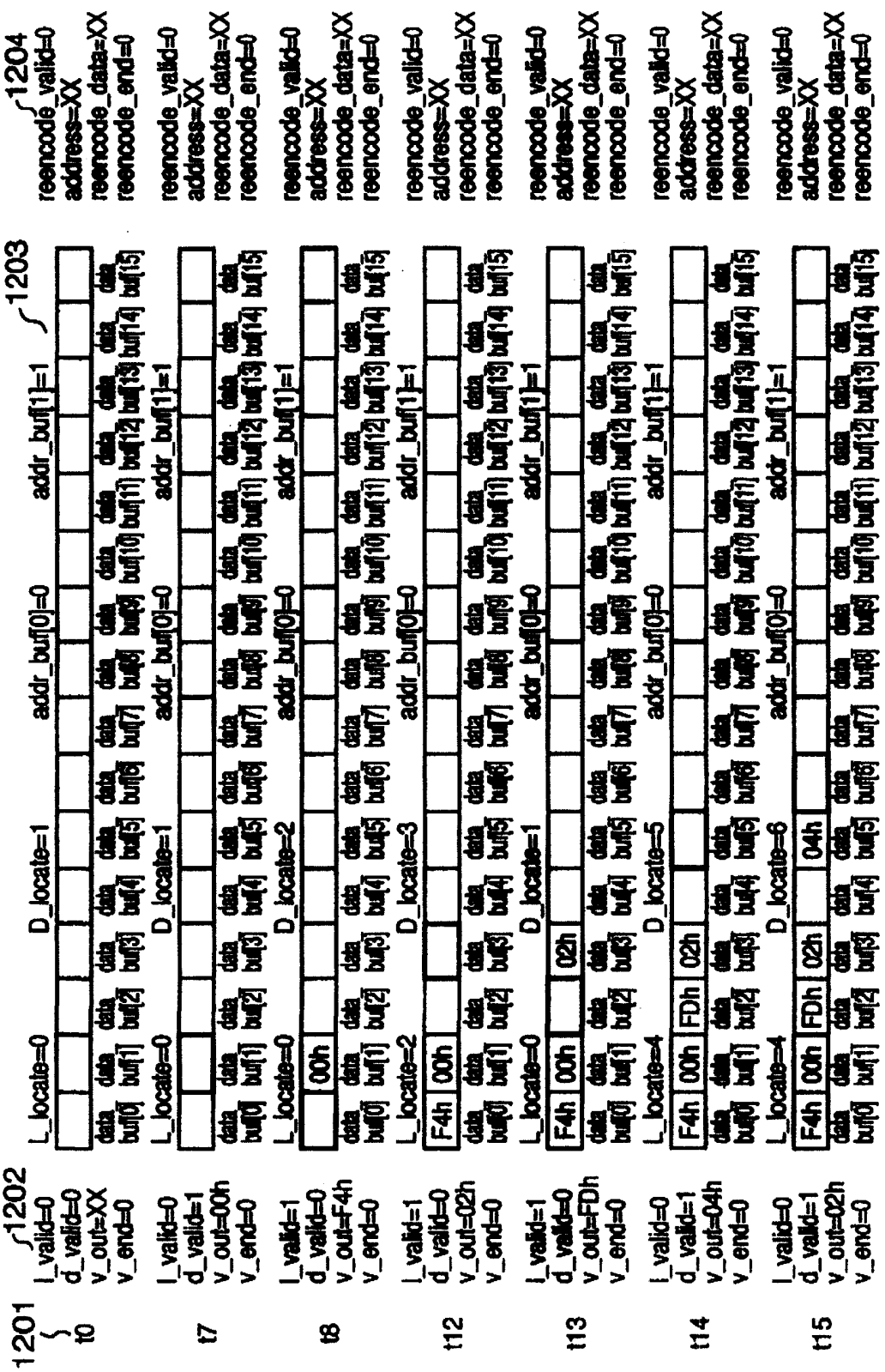

1201
1202
1203
1204
t0
t7
t8
t12
t13
t14
t15
l_valid=0
d_valid=0
v_out=XX
v_end=0
l_valid=1
d_valid=0
v_out=F4h
v_end=0
l_valid=0
d_valid=1
v_out=00h
v_end=0
l_valid=1
d_valid=0
v_out=02h
v_end=0
l_valid=0
d_valid=1
v_out=04h
v_end=0
l_valid=1
d_valid=0
v_out=FDh
v_end=0
l_valid=0
d_valid=1
v_out=02h
v_end=0
L_locate=0
L_locate=1
L_locate=2
L_locate=3
L_locate=4
L_locate=5
L_locate=6
D_locate=1
D_locate=2
D_locate=3
D_locate=4
D_locate=5
D_locate=6
addr_buf[0]=0
addr_buf[1]=1
reencode_valid=0
address=XX
reencode_data=XX
reencode_end=0
F4h
00h
FDh
02h
04h

图 12

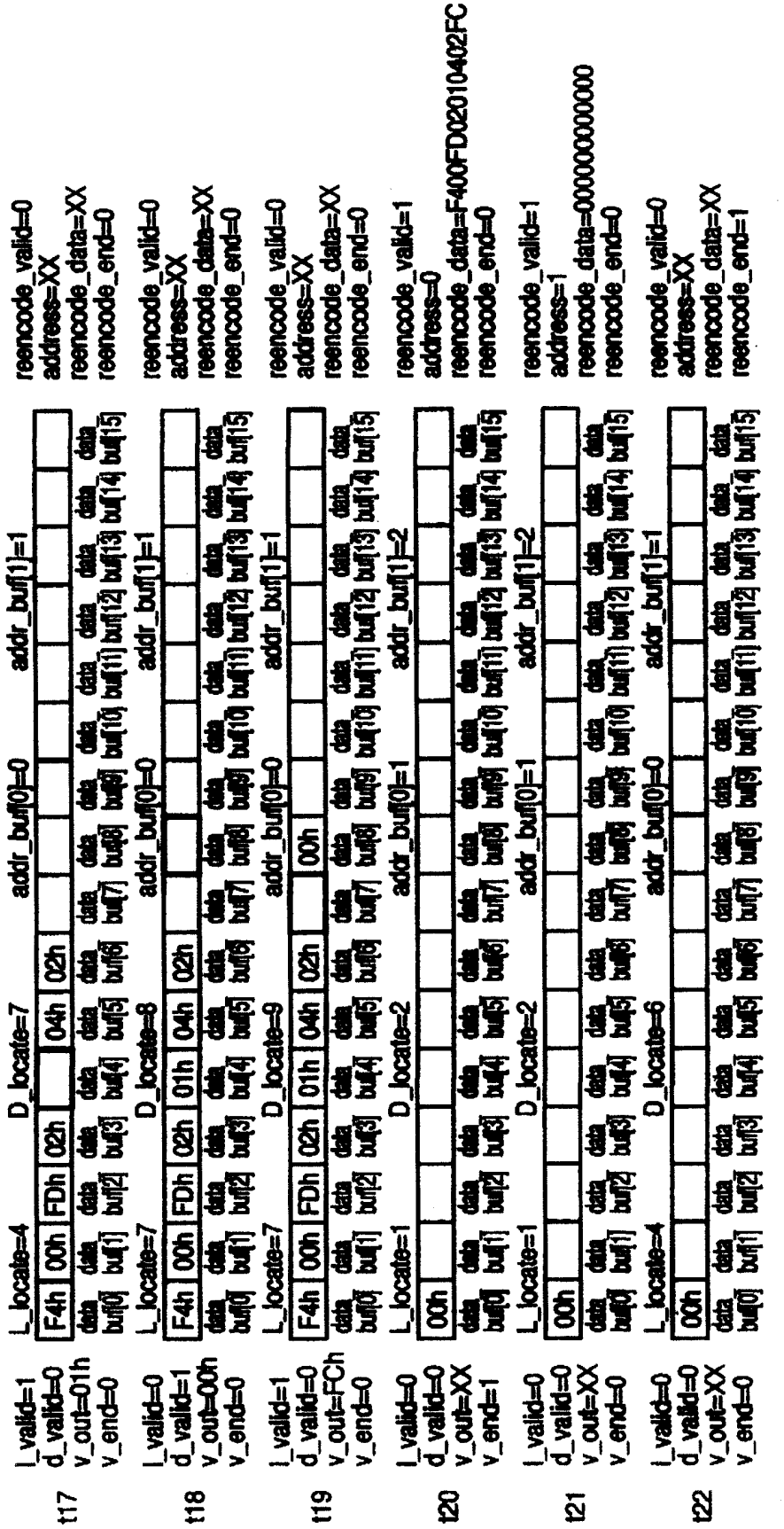

t17
t18
t19
t20
t21
t22
l_valid=1
d_valid=0
v_out=01h
v_end=0
l_valid=0
d_valid=1
v_out=00h
v_end=0
l_valid=1
d_valid=0
v_out=FCh
v_end=0
l_valid=0
d_valid=0
v_out=XX
v_end=1
l_valid=0
d_valid=0
v_out=XX
v_end=0
l_valid=0
d_valid=0
v_out=XX
v_end=0
L_locate=4
D_locate=7
addr_buf[0]=0
addr_buf[1]=1
F4h 00h FDh 02h 04h 02h
L_locate=7
D_locate=8
addr_buf[0]=0
addr_buf[1]=1
F4h 00h FDh 02h 01h 04h 02h
L_locate=7
D_locate=9
addr_buf[0]=0
addr_buf[1]=1
F4h 00h FDh 02h 01h 04h 02h 00h
L_locate=1
D_locate=2
addr_buf[0]=1
addr_buf[1]=2
00h
L_locate=1
D_locate=2
addr_buf[0]=1
addr_buf[1]=2
00h
L_locate=4
D_locate=6
addr_buf[0]=0
addr_buf[1]=1
00h
reencode_valid=0
address=XX
reencode_data=XX
reencode_end=0
reencode_valid=0
address=XX
reencode_data=XX
reencode_end=0
reencode_valid=0
address=XX
reencode_data=XX
reencode_end=0
reencode_valid=1
address=0
reencode_data=F400FD02010402FC
reencode_end=0
reencode_valid=1
address=1
reencode_data=0000000000
reencode_end=0
reencode_valid=0
address=XX
reencode_data=XX
reencode_end=1

图 13

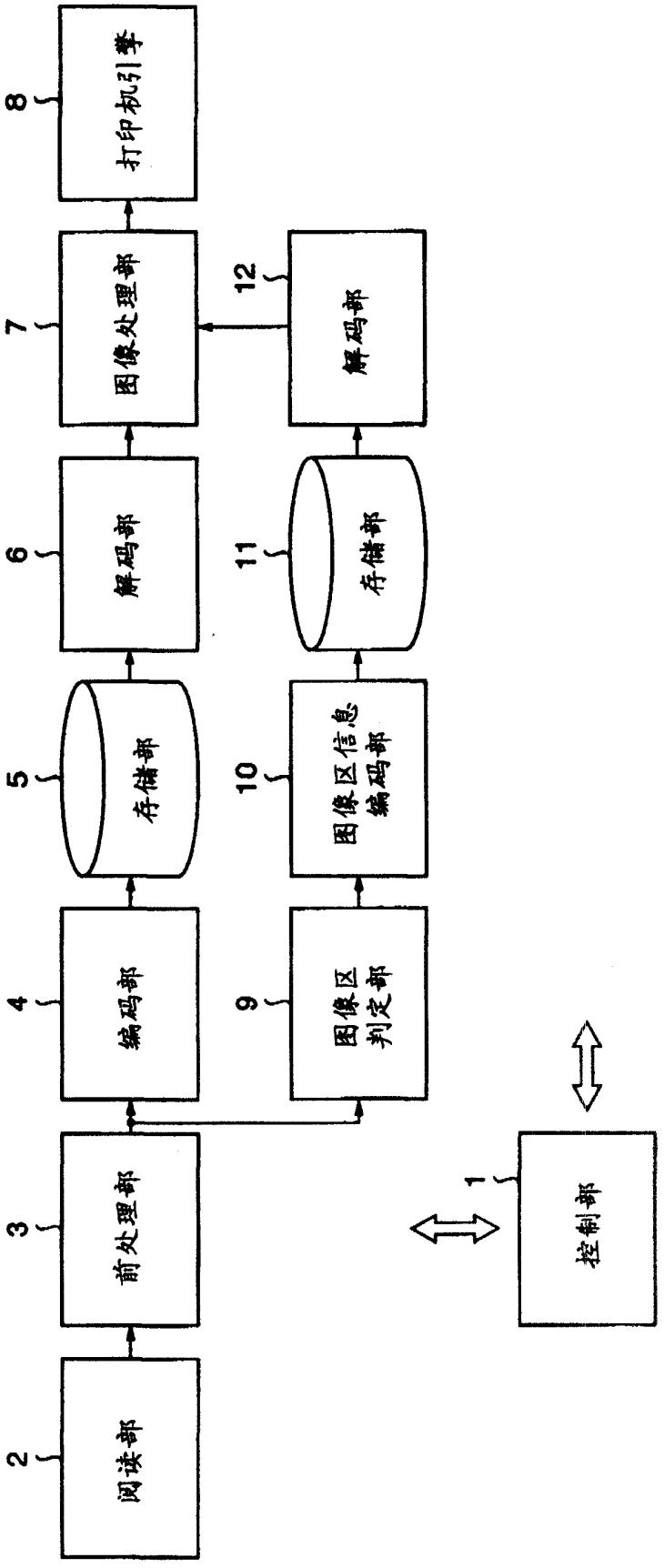

2
阅读部
3
前处理部
4
编码部
5
存储部
6
解码部
7
图像处理部
8
打印机引擎
9
图像区判定部
10
图像区信息编码部
11
存储部
12
解码部
1
控制部

图 14

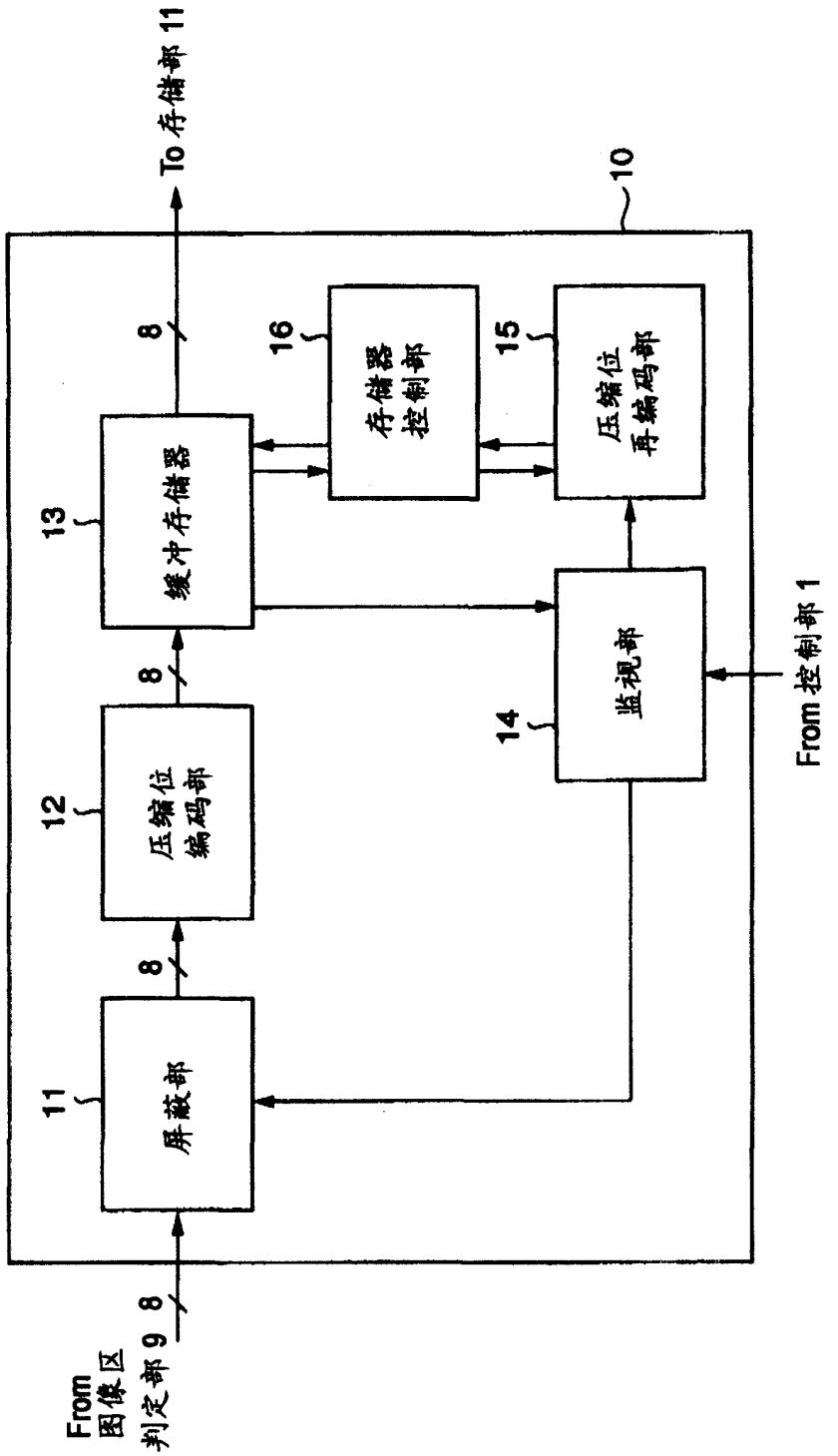

From 图像区判定部 9
8
11 屏蔽部
8
12 压缩位编码部
8
13 缓冲存储器
8
To 存储部 11
10
16 存储器控制部
15 压缩位再编码部
14 监视部
From 控制部 1

图 15

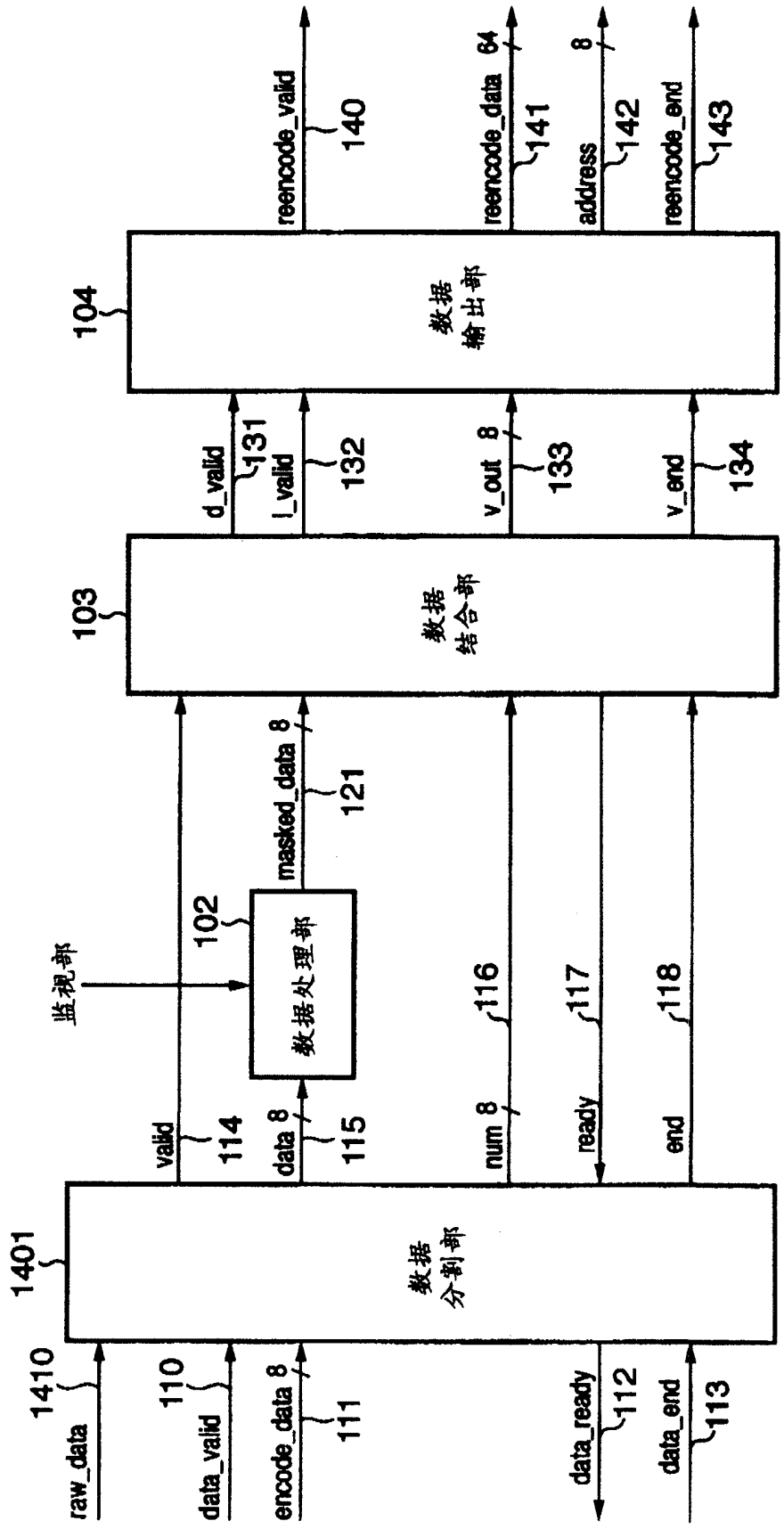

raw_data
1410
data_valid
110
encode_data 8
111
1401
数据分割部
valid
114
data 8
115
num 8
116
ready
117
end
118
data_ready
112
data_end
113
监视部
102
数据处理部
masked_data 8
121
103
数据结合部
d_valid
131
l_valid
132
v_out 8
133
v_end
134
104
数据输出部
reencode_valid
140
reencode_data 64
141
address 8
142
reencode_end
143

图 16