[19] 中华人民共和国国家知识产权局

[51] Int. Cl[7]
H03M 7/30

[12] 发明专利说明书

[21] ZL 专利号 94106536.7

[43] 授权公告日 2003 年 3 月 5 日

[11] 授权公告号 CN 1102812C

[22] 申请日 1994.6.7 [21] 申请号 94106536.7

[30] 优先权

[32] 1993.7.8 [33] US [31] 089211

[71] 专利权人 国际商业机器公司

地址 美国纽约州

[72] 发明人 M·A·哈斯纳 E·D·卡宁
U·施韦格索恩 田村哲也

审查员 王艳坤

[74] 专利代理机构 中国专利代理(香港)有限公司

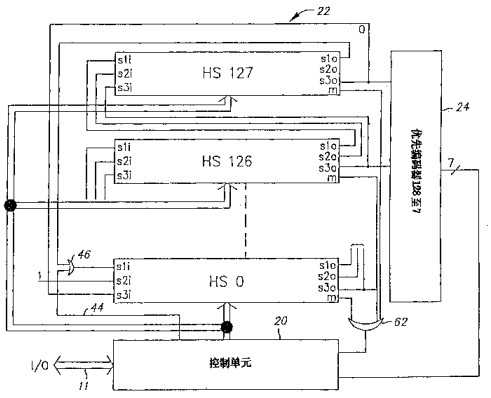
代理人 傅 康 王忠忠

权利要求书 3 页 说明书 11 页 附图 4 页

[54] 发明名称 执行顺序数据压缩算法的方法与装置

[57] 摘要

一种执行顺序数据压缩算法的装置和方法，它用于一个设备控制器中需要的数据压缩。一个历史缓冲器压缩由 i 个相同水平片单元组成的一个阵列。每个片单元存储 j 个符号以定义 j 个独立的块，在块中处在每个片单元的符号刚好由 i 个符号分隔开。在由 i 个到达的符号组成的串中的符号，由 i 个比较器并行地与片单元中先前存储的符号进行比较，以识别出匹配的符号序列。一个控制单元控制顺序算法的执行，以限定比较器并行扫描符号但在每个块中为顺序扫描，并使匹配的符号序列和不匹配的符号序列存储于该阵列中。



ISSN1008-4274

1. 一种执行顺序数据压缩算法的装置，其特点在于包括：

i 个相同的水平片单元构成的一个阵列，每个片单元存储 j 个符号，以定义 j 个独立的块，块中每个片单元的符号刚好被 i 个符号分隔开；

i 个比较器用于将 i 个并行到来的符号组成的串与先前存储在所述片单元中的符号进行比较，以识别出匹配的符号序列；以及

包括控制器装置的装置，用于（1）执行顺序算法，支配比较器顺序地扫描并行的但位于每个所述块中的符号；（2）使匹配的和不匹配的符号序列存储于所述阵列中。

2. 权利要求 1 的装置，其特征在于：

根据算法执行的速度和硬件费用之间的权衡来选择参数 i 和 j，以限定在执行该算法时达到希望的有效程度所需要的所述比较器的数目。

3. 权利要求 1 的装置，其特点在于如果识别出一个区配序列则每个片单元提供一个具有一种状态的输出信号，该装置的特点还在于包括：

一个优先编码器用于从所述输出信号计算出每个从中识别出匹配序列的 j、i 地址，但只输出所述地址，但只输出所述地址中的一个地址。

4. 权利要求 3 的装置，其特点在于每个水平片单元的输出信号当处于所述一种状态时标识出与到来的符号串中最近序列迄今为止相匹配的先前存储序列的最后一个字节。

5. 权利要求 1 的装置，其特点在于包括：

禁止将该算法应用于任何不包含符号的块的装置。

6. 权利要求 1 的装置，其特点在于包括：

在每个单独的片单元中有一单个锁存器，这些锁存器彼此串联，组合起来构成一个移位寄存器；以及

在每次将符号存储于所述阵列中不同单元之前复位所述各锁存器并因此复位所述移位寄存器的装置。

7. 一种数据压缩装置，共特点在于它的组成是：

分成 j 块的一个历史缓冲器，每个块能存储 i 个接续的符号，从而使相邻块中的相应符号位置的符号相隔 i 个符号；

i 个比较器；

一个控制器，用于将 i 个到来符号组成的串中的每个符号并行播发到所述 i 个比较器中；以及

i 个选择器装置，用于使到来的符号串行写入该缓冲器中的接续的符号位置，直至全部缓冲器位置被充满为止，然后使缓冲器中的最老的符号串被到来的符号所替换。

8. 执行一种顺序数据压缩算法的一种方法，其特点在于由下列步骤组成：

提供一个由 i 个相同的水平片单元组成的一个阵列，每个单元用于存储 j 个符号，以定义 j 个独立的块，在块中处于每个片单元的符号刚好被 i 个符号分隔开；

使用 i 个比较器，将 i 个并行到来的符号组成的串中各符号与先前存储在所述各片单元中的符号进行比较，以识别出匹配的符号序列；以及

通过（1）并行扫描符号但序列扫描所述每个块中的符号，以及（2）将匹配的符号序列和不匹配的符号序列存储于一个输入/输出设备，来执行该顺序算法。

9. 权利要求 8 的方法，其特点在于包括下述步骤：

根据算法执行速度和硬件费用之间的权衡来选择参数 i 和 j，以限定为达到希望的有效程度所需要的所述比较器的数目。

10. 权利要求 9 的方法，其特点在于包括下述步骤：

计算出在其中识别出一个匹配序列的每个 j、i 地址；以及

输出所述地址中只是最小的一个的地址。

11. 权利要求 10 的方法，其特点在于包括下述步骤：

识别出与到来的符号串中的最近序列迄今为止相匹配的一个先前存储序列的最后一个字节。

12. 权利要求 8 的方法，其特点在于包括下述步骤：

禁止将该算法应用于不包含有效符号的任何块。

13. 一种数据压缩方法，包括下述步骤：

提供一个分成 j 个块的历史缓冲器，各块能存储 i 个相继的符号，

使得相邻的块中对应的符号位置上的符号相距 i 个符号；

提供 i 个比较器；

并行地将由 j 个输入符号组成的一个串中的各个符号播送到所述 i 个比较器；

将输入符号串行写入缓冲器中相继的符号位置中，直到充满所有的缓冲器位置为止；以及

然后用输入符号来替换缓冲器中最老的符号串。

执行顺序数据压缩算法的方法与装置

本发明涉及对数据存储系统中的数据进行压缩和脱压缩的设备与方法，更具体地说涉及成块并行执行一个顺序数据压缩算法的设备和方法，这种算法特别适用于低速终端数据存储系统。

众所周知，数据压缩与脱压缩的效率主要取决于所用缓存区的大小及编码实现。利用软件执行压缩／脱压缩算法速度慢，因而不适于高速或实时应用。用硬件执行算法所需的硬件所需的硬件量随着所利用的实现技术的并行程度而变化。如果需要太大量的硬件，则可能难于把数据压缩算法集成于一个控制器中。

Lempel和Ziv 在题为“一种顺序数据压缩的普适算法”（发表于ＩＥＥＥ信息论学报１９７７年５月号，第３３７－３４３页）一文中描述了一种有效地压缩数据的方法。这种Lempel－Ziv 1（ＬＺ１）算法是一种顺序算法，它把可变长度二进制数据串压缩成固定长度的压缩二进制格式。它是利用一个“历史缓冲器”来实现的，在该缓冲器中包含按正确顺序存放的一个文件的若干最近字节或字。在方法上，通过重复执行一个基本的例行程度，只要输入的字节序列与历史缓冲器中的一个序列匹配，新字节便被读取，因

此产生一个顺序的数据流。由于每个输入字节要顺序地与历史缓冲器内的每个字节进行比较，所以需要大量的计算时间，使得这种技术不适于实时的应用。

共同转让的美国顺序号07/807,007（1991年12月31日提交），（Docket AT991-030））描述了LZ1算法的一种典型实现并引述了许多专利（不认为是本发明的材料），这些专利覆盖了改善LZ1算法执行速度或实现压缩程度的多种技术。

这一被引用的共同转让申请描述了在硬件中实现LZ1算法的完全并行的体系结构。利用一个按内容寻址存储器（CAM）作为历史缓冲器，每个输入字节同时与历史缓冲器中的所有字节进行比较。这种完全并行硬件技术理想地提供了LZ1算法的最快执行手段。然而，它要求为每个独立缓冲位置（即，CAM地址）准备单独的比较器，而且只有当历史缓冲器满时才能达到最高效率（速度/硬件性能），也就是说，是在对数据存储介质的每个扇区或输入数据段的初始加载阶段之后才能达到最高效率。所以，如果该扇区与历史缓冲区有近似相同的大小时，全并行方法将要求许多冗余的操作。

因为一个设备控制器芯片的大小实质上与实现纯并行压缩所需芯片的大小相同，所以一个并行压缩芯片不能有效地用于完成历史缓冲器中的压缩。这种全并行方法主要是用于主机数据压缩，这时压缩芯片位于主机控制器中。

需要有一种使用模块结构实现LZ1算法的数据压缩/脱压缩装置与方法，它：

1、把历史缓冲器分成多个块，在一个 块中并行比较所有字

节，而对各块顺序扫描；

2、使设计者能够通过选择所希望的并行程度来选择从缓慢的LZ1算法顺序执行到上述最佳并行实现这一范围内的任何速度，以使硬件费用限定在具体应用的实际需要上；

3、特别适合于在设备控制器中完成数据压缩的各种应用，这里所要求的执行速度近似于比在主机控制器中完成的压缩所要求的速度低一个数量级；以及

4、当输入数据扇区和历史缓冲器包含近似相同数量的字节时显得特别具有优越性。

本发明描述了一种执行顺序数据压缩算法的装置和方法，它特别适合于在设备（以与主机相区别）控制器中要求的数据压缩。一个历史缓冲器压缩由i个完全相同的水平片单元组成的一个阵列。每个片单元存储j个符号来定义j个单独的块，在这些块中，每个片单元的符号正好被i个符号分隔开。i个输入符号组成的串中的符号由i个比较器与片单元中先前存储的符号进行并行比较，以识别出匹配的符号序列。一个控制单元控制顺序算法的执行，以支配各比较器并行扫描各符号，但在每个块中则是顺序扫描，并使符号匹配序列和不匹配序列存储于该阵列中。根据算法执行速度和硬件费用之间的权衡来选择参数i和j，以限定执行该算法时为实现希望的有效程度所要求的比较器个数。

一个优先编码器根据片单元输出信号计算每个识别出的匹配序列的j，i地址，但它只输出这些地址中的一个（例如最小地址）。到来符号被串行写入缓冲区中的连续符号位置，直至所有缓冲区位置被充满为止。然后，缓冲区中的最老符号串被到来的符号所取代。

图1是实现本发明的数据压缩系统的框图。

图2是实现本发明的数据压缩／脱压缩装置的示意图，该装置中包括由完全相同的水平片单元组成的一个阵列和一个优先编码器。

图3的示意图详细给出图2所示每一个水平片单元的配置。

图4给出图2所示优先编码器的一种简化形式的电路逻辑图。

如图1所示，实现本发明的一个数据压缩系统包含一个主计算机10，它向包括压缩／脱压缩装置14的设备控制器12传送数据和从该控制器12接收数据。装置14包括一个压缩机15和一个脱压缩机16，它们能被分别启动以压缩机和脱压缩数据。压缩机15向输入／输出（I／O）设备18（例如包含一组盘的盘驱动器）提供被压缩数据输出。设备18向设备控制器12的脱压缩机16提供被压缩数据输入。

如图2所示，数据压缩／脱压缩装置14包括一个控制单元20、一个历史缓冲器22和一个优先编码器24。输入／输出（I／O）总线11对控制单元20取、送数据。数据是采取“符号”的形式的，“符号”是在权利要求中使用的术语，统指字节、半字、字、或任何其他预先选定个数的二进制位。然而，为便于理解，这里假定数据是字节形式的。

如图所示，历史缓冲器22包含一个由128个相同水平片单元HSO至HS127组成的阵列，每个片单元存储4个字节，从而将缓冲器分成4个各存储128字节的块。这种模块结构造成的历史缓冲器22能存储接续的512（128×4）个字节，其中每个HS（水平片）单元中的字节正好按128字节（即1块的大小）分隔开，而且每个HS单元存储4个字节，一个字节来自一个

块。

缓冲器22中的每个字节b有一个由其块索引B和HS索引标识的唯一地址。这样，对于字节0－127，其块索引B为0；对于字节128－255，块索引B为1；对于字节256－383，块索引B为2；而对字节384－511，其块索引B为3。HS索引是字节地址除以块大小的余数。

这样，字节356的地址即为[2,100]。 由来自每个块中同一HS地址0…127的一个字节所组成的4个字节b构成了一个字W，在这种情况下该字W的地址便是（1，L；2，L；3，L；4，L），这里L是一个特定的HS地址。 对于图中所示4×128一字节缓冲器，如果用二进制格式表示一个字节的地址，可以用二个高位表示块索引，而用7个低位表示HS单元索引。

现在参考图3，每个HS单元包括一个存储器寄存器或元件ME，一个地址选择元件SE，一个比较器元件CO，一个多路转换器MU和锁存器S1、S2、S3。控制器20向每个HS单元传送一个输入符号串的相同的8位字节和控制该HS单元的多路转换器MU的4个信号C1－C4。控制器20还向每个HS单元并行提供：信号r1和r2用于分别复位锁存器S1和S2；分别进行块地址选择的信号ad1和ad2；同步所有HS单元的操作的时钟信号CK；以及允许写入信号W。

每个HS单元有输出S1o、S2o、S3o，它们代表各锁存器S1、S2、S3的当前状态。这些输出以循环方式构成下一个紧接着的HS单元的输入S1i、S2i、S3i（见图2），而HS127的输出S1o、S3o作为HSo的输入S1i、

S3i。多路转换器MU使用控制信号C1－C4并产生输出m。

在操作过程中，假定缓冲器22的内容（即1L，2L，3L，4L，S1，S2，S3）是未定义的。每当一个新扇区被压缩写时便发生这种情况。初始时，控制单元20并行地向所有HS单元的与（AND）门42发送信号r1，以便所有HS单元中的锁存器S1复位。只有在下列条件下才能将一个字节写入一个给定HS单元的存储单元中：

1、来自控制单元20的允许写入信号W必须为“1”。

2、该HS单元的锁存器S1必须为“0”（即处于复位状态）

3、对该HS单元的输入S1i必须为“1”。

以图3所示与门（AND）50来满足这些条件。当一个块满时，除最后一个（这里是HS127）以外的所有HS单元的锁存器S1被置位（即为“1”）。S1＝“0”的HS单元是缓冲器22中由新的输入字节代替老字节的位置所在。然而，由于初始时所有HS单元的S1锁存器有值“0”，故控制单元20通过线44和或门（OR）46（如2）提供一个“1”信号。这个“1信号一直保持到除最后一个（这里HS127）外的所有HS单元的S1锁存器为“1”时为止。这一过程重复进行（即对所有HS单元中的S1锁存器复位和在线44上保持初始化“1”信号）直至所有块被充满为止；此时，线44上的控制信号被控制单元20切换成“0”。

每个HS单元的锁存器S1也提供该单元的输出S1o。因此，所有HS单元的S1锁存器起到128－位移位寄存器的作用。

这个移位寄存器被控制单元２０并行传送给所有HS单元的允许写入信号W更新。

允许写入信号W还使来自线４８的到来字节被写入该HS单元的存储单元ME。其存储位置由锁存器Ｓ１和输入Ｓ１ｉ确定为由选择器单元SE确定的位置１Ｌ，２Ｌ，３Ｌ或４Ｌ之一。锁存器Ｓ１的优点之一在于：在一个新扇区的开始不需要对存储单元复位，从而减小了功耗。这一特点对于因蓄电池电源限制而使功耗问题成为关键的设备控制器实现而言具有实际重要性。

为起动与一个新块的串比较操作，控制单元２０通过与门４０向所有HS单元并行发出信号ｒ２以使锁存器Ｓ２复位。Ｓ２的输出控制每个HS单元中的选择元件SE。控制单元２０还向所有HS单元的SE元件发出两个信号ａｄ１和ａｄ２。这些２－位信号ａｄ１和ａｄ２代表相邻的块地址（例如Ｂ１和Ｂ２）。根据每个HS单元中锁存器Ｓ２的状态，ａｄ１和ａｄ２中的一个成为该HS单元的确定块地址ａｄ０。初始时因为所有Ｓ２锁存器被复位，故对所有HS单元都有ａｄ０等于ａｄ１。一个HS单元的Ｓ２锁存器Ｓ２的输出变成输出Ｓ２ｏ，然后变成下一个HS单元的输入Ｓ２ｉ。

这样，所有锁存器Ｓ２构成一个１２８位移位寄存器，它的输入（来自HSｏ的Ｓ２ｉ）总是“１”。随着输入字节串与缓冲器２２内容匹配过程的继续进行，这个移位寄存器从底部起填充“１”，这样使越来越多的HS单元选择ａｄ２而不选ａｄ１。这构成了在匹配过程中块边界的虚拟移动。例如，假定在缓冲器２２中已匹配到字节２５５（即达到字节地址[1,127]）。然后，下一个到

来的字节要与缓冲器中的字节256进行比较，该字节256有一个不同的块地址，即[2,0]。

到来的字节还经由线48的分支传送到每个HS单元的相应的比较器元件Co。在全部128个HS单元中的比较器单元Co将到来的字节同时与选择器元件SE选出的四块中的一个特定块中的128个字节进行比较。如果存储单元ME之一的输出字节与线48中到来的字节匹配，则比较器Co将向多路转换器单元MU提供一个“1”输出信号。

多路转换器单元MU被控制单元20限定在由信号C1-C4确定的4个模式或状态之一。这些信号实现的功能如下：

信号C1表示到来的字节是一个新的匹配串的第一字节，还表示当前工作块（即，它的地址由ad0给出）是满的。在这种情况下锁存器S1的状态可被忽略。

信号C2表示到来的字节也是一个新的匹配串的第一字节，但当前工作块还没有满（即在该块中某些存储单元的内容未被确定），所以只能当锁存器S1为“1”时才发生匹配。

信号C3表示到来的字节不是一个匹配串的第一字节，而且锁存器S1的状态不能被忽略。只有当整个缓冲器是满的，而且一个块的移位区与该匹配串要写入的另一块重迭时，才会发生这种状况。

信号C4表示到来的字节不是一个匹配串的第一字节，但锁存器S1的状态能被忽略。

如果一个到来的字节不是一个匹配串的第一字节，那么接下来的匹配便要求前一个字节也必须是已经匹配的，这是由来自前一个HS单元的输入信号S3i表示的一种状态。

这样，当多路转换器MU由信号C1或C2支配时，信号S3i能被忽略，因为S3i表示一个串的初始字节的匹配。比较器元件Co的m信号有选择地与52处的信号C1、54处的信号C2和S1o、56处的信号C3和S1o和S3i、以及58处的信号C4和S3i进行“与”（AND）操作。各个“与”门52、54、56和58的输出在60处进行“或”（OR）操作，产生输出信号m。如果输出信号m为“1”，它表示缓冲器22中的一个匹配串已经延伸到包括最近到来的字节。所有HS单元的m输出在门62处进行“或”（OR）操作，以通过控制单元20当前输入序列已在当前的工作块中找到了一个有效的匹配。一个HS单元的这个输出信号m还输入到锁存器S3。所以，如果一个HS单元的锁存器S3的输出S3o为“1”，它意味着由该单元和地址ad0确定的缓冲器单元是迄今为止与最近输入序列匹配的先前存储序列的最末字节。每个HS单元的S3o输出与优先编码器24相连。

应该指出，缓冲器22的更新独立于串比较操作并能与串比较操作同时完成。

优先编码器24使用来自128个HS单元的S3o输出，计算和编码与到来的待压缩数据中最近序列匹配的历史缓冲器22中的一个序列的结尾地址。在一个并行实现中，例如这里所描述的情况，可能在历史缓冲器22中有不止一个序列与来到的序列匹配。在这种情况下，根据本发明的一个特点，编码器24将对一个匹配序列只提供单个结尾地址。

将参考图4所示简化的优先编码器24′来说明选择单个结尾

地址的方式。这个编码器编码512字节，但为了简化图示和描述，假定只有8个HS单元，每个单元存储64字节，每个字节来自64块中的一块，只编码成3位（而不是象图3所示对该类128个HS单元的地址进行变换时所需的7位）。

如图4所示，简化的优先编码器24′接收来自全部（现在是8个）HS单元的8个S3o信号输入。然后从S3o为“1”的各HS单元当中确定8个索引号中最低的一个。这由图4所示逻辑实现的，它只保留最小HS地址的S3o信号而忽略所有其他S3o信号。这样，它确定了匹配的序列的唯一结尾地址。这一地址被送到控制单元20。

由于脱压缩算法需要一个匹配序列的起始地址，必须从优先编码器24提供的结尾地址计算出这一起始地址。因为已知该匹配序列起始于当前工作块（所有S2锁存器在初始时都曾被复位），所以只需要计算该地址的末7位。这只要减去匹配序列的长度就可做到，即从结尾地址中减去该序列的字节数，并取其结果除以HS单元个数（图3所示实施例中为128）的余数。匹配序列的长度是压缩的数据的一部分，所以在控制单元20中是可以得到的。在控制单元20中利用一个增量计数器便可容易地得到这个序列长度。

用于对已由申请人的修正的LZ1压缩算法所压缩的数据进行脱压缩的算法原本是顺序性的，所以它不依赖于在压缩机15中使用的并行程度。然而，为了有效地使用硬件，使用了压缩机15的历史缓冲器22。被压缩的到来数据包含一个起始缓冲器地址、一个串长度数值、以及一个字符，该字符是被压缩串中的最后一个符号。控制单元20在起始地址处抽取缓冲器内容，并以按压缩机

15中同样的方式抽取的符号来更新缓冲器22。对被压缩串的整个长度重复这一步骤。在这一处理过程中，以一种循环方式对缓冲器地址不断地增量。最后，这最后一个符号也被送到主机10，于是控制单元20再次准备好接收下一个被压缩串的数据。

尽管已经参考其最佳实施例具体显示和描述了本发明，精通本门技术的人们将会理解，可以在形式和细节上进行各种改变而不偏离本发明的精神和范围。因此，认为本发明除了由权利要求书所要求的内容之外没有其他限制。

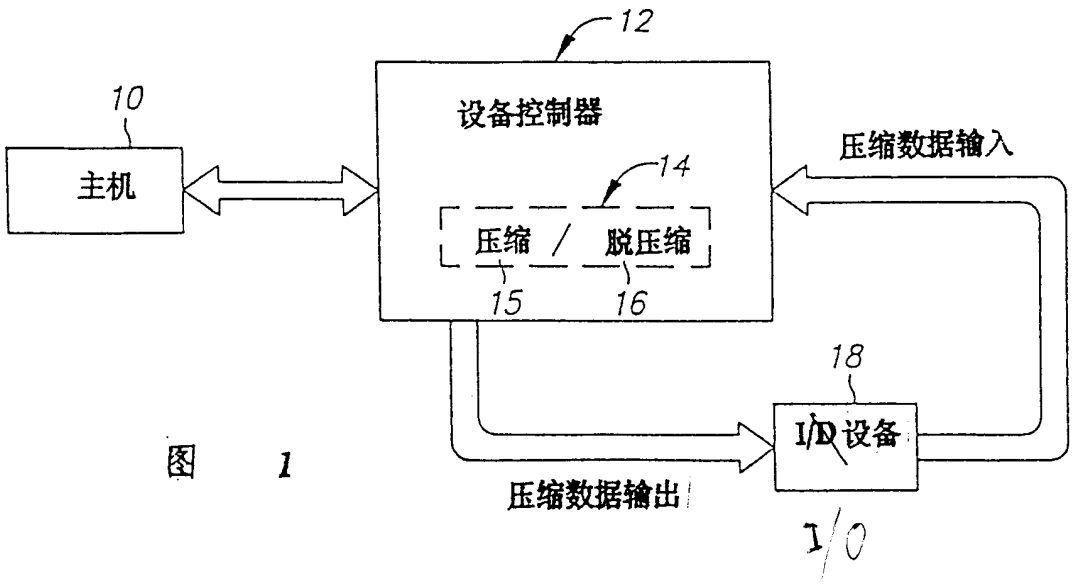
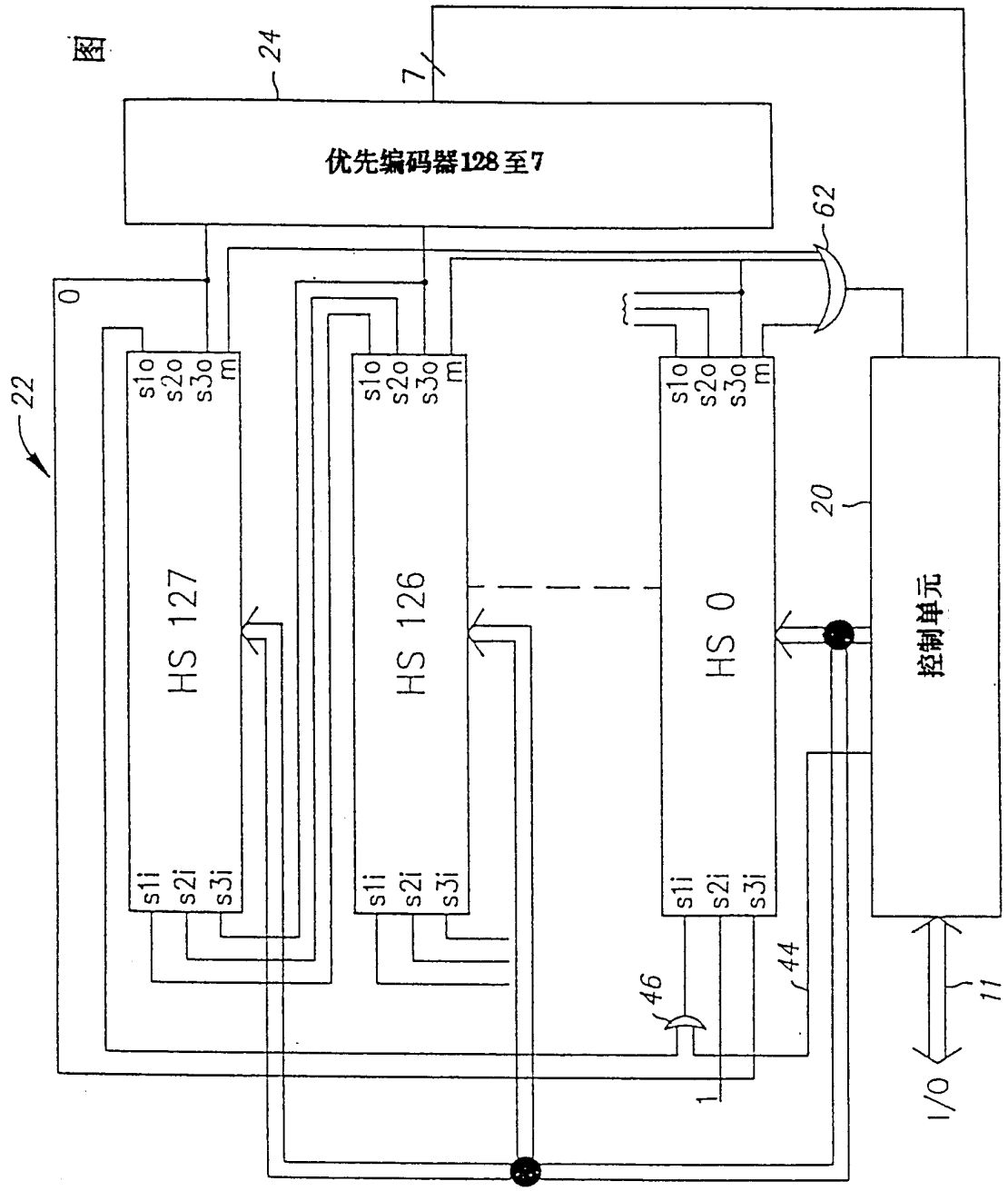


图 1

图 2


优先编码器128至7
24
7
62
22
0
HS 127
HS 126
HS 0
s1o
s2o
s3o
m
s1i
s2i
s3i
20
控制单元
46
44
1
11
I/O

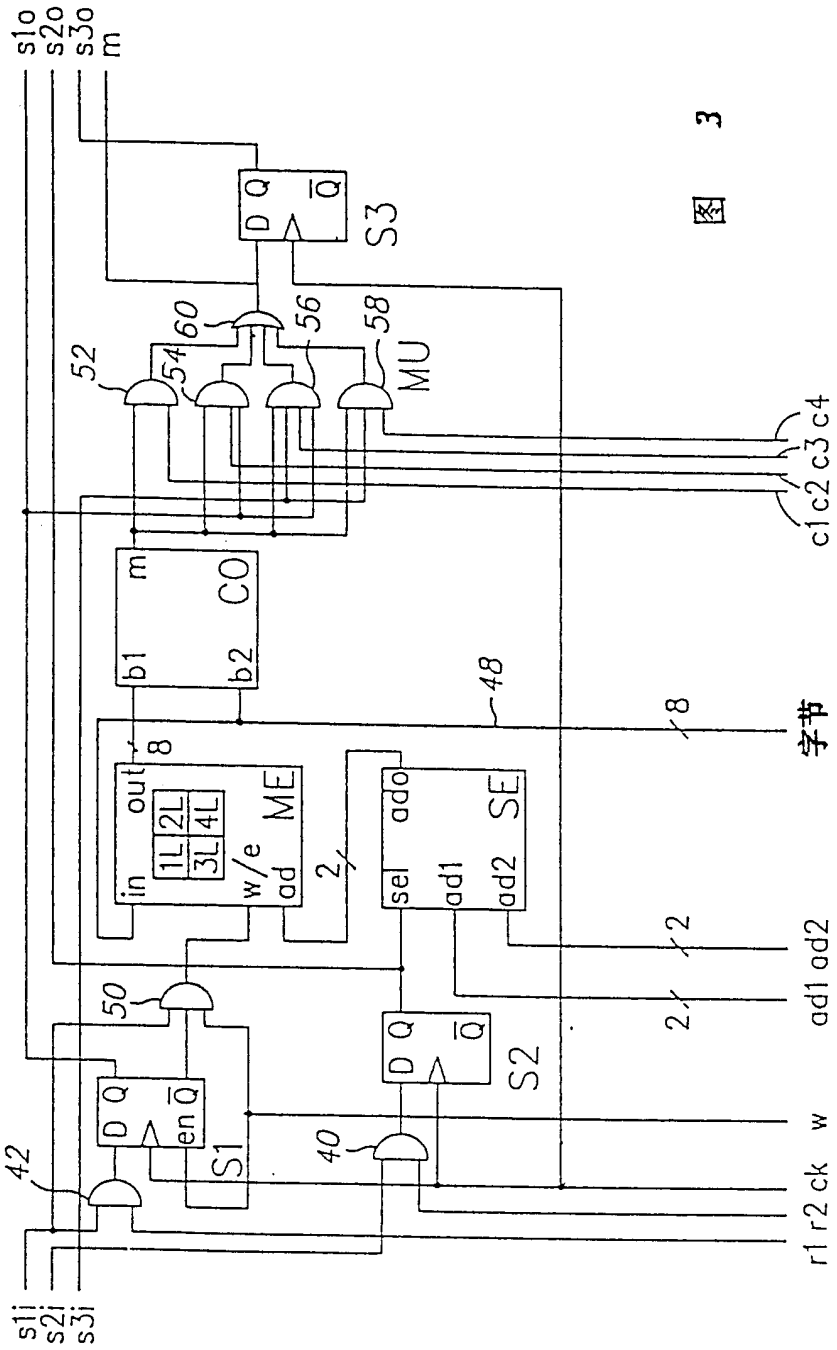

s1o
s2o
s3o
m
S3
MU
52
54
60
56
58
c1 c2 c3 c4
CO
b1
b2
48
8
字节
ME
in
out
1L
2L
3L
4L
w/e
ad
2
SE
sel
ado
ad1
ad2
2
ad1 ad2
50
S2
S1
42
40
w
ck
r1 r2
s1i
s2i
s3i

图 3

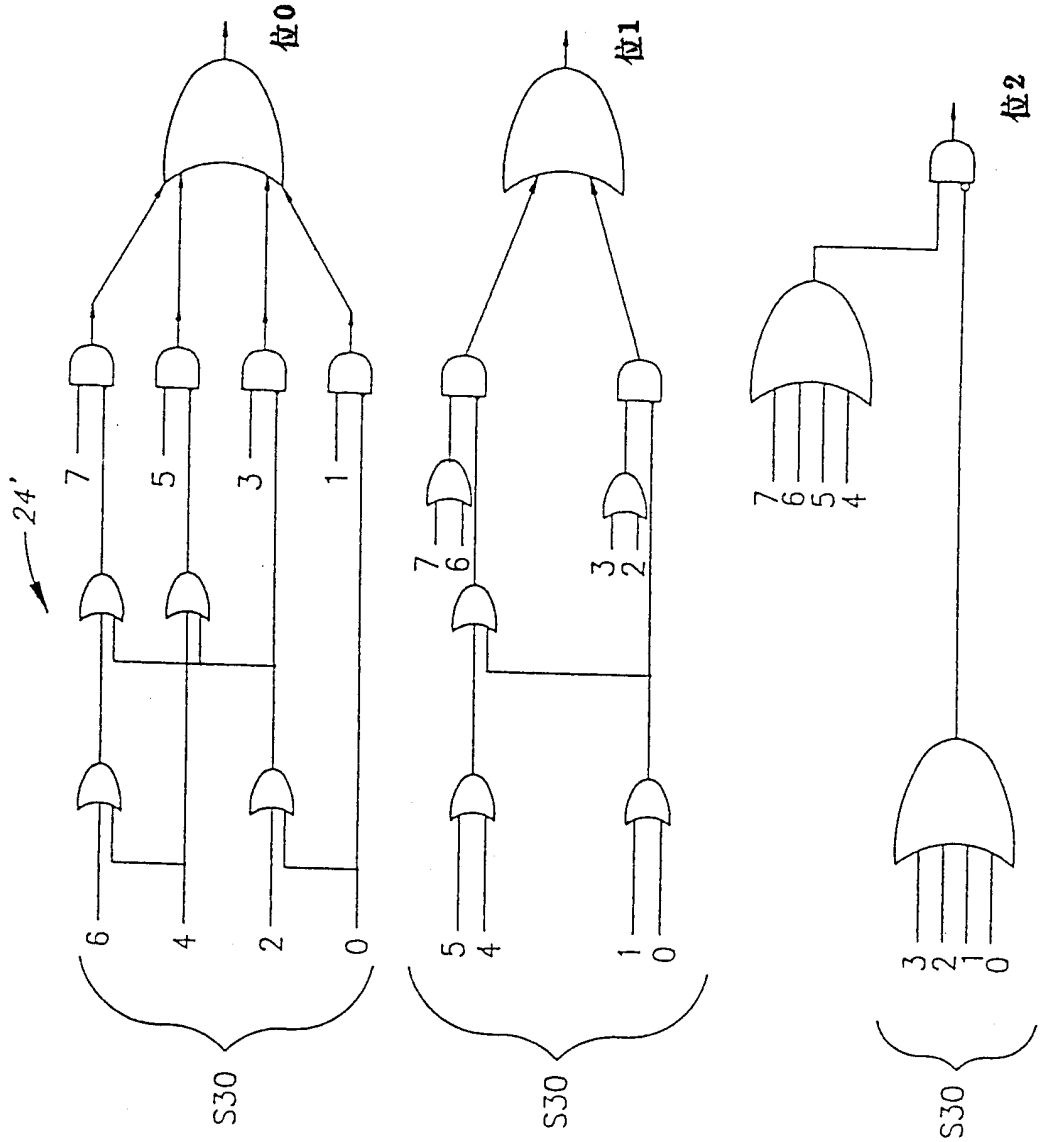


图 4