[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 95197190.5

[43] 授权公告日 2003 年 6 月 4 日

[11] 授权公告号 CN 1110904C

[22] 申请日 1995.12.18 [21] 申请号 95197190.5

[30] 优先权

[32] 1994.12.29 [33] US [31] 08/366,356

[86] 国际申请 PCT/US95/16615 1995.12.18

[87] 国际公布 WO96/21283 英 1996.7.11

[85] 进入国家阶段日期 1997.6.28

[71] 专利权人 尤尼西斯公司

地址 美国宾夕法尼亚

[72] 发明人 阿尔伯特·B·库伯

[56] 参考文献

EP0573208A1 1993.12.08 H03M7/30

US5151697A 1992.09.29 H03M7/30

审查员 王 莉

[74] 专利代理机构 中国国际贸易促进委员会专利商标事务所

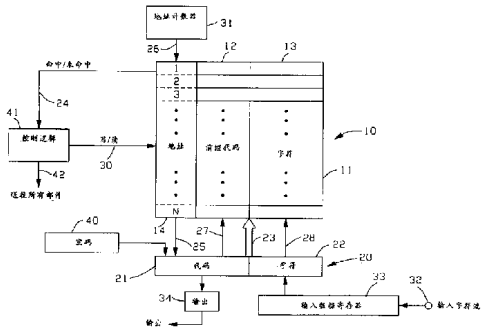
代理人 吴丽丽

权利要求书 8 页 说明书 9 页 附图 2 页

[54] 发明名称 数据压缩方法和装置

[57] 摘要

相联存储器(11)被用来实现 LZW 数据压缩。存储器的各个位置包含一个前缀代码域(12)和一个字符域(13)。 包含一个代码域(21)和一个字符域(22)的寄存器(20)同存储器中的存储位置的内容进行相关比较以确定是否存在一个匹配，如果有，匹配地址(14)被插入寄存器的代码域中，下一个输入字符则插入到寄存器的字符域中。 这一过程连续执行直到没有匹配出现为止。 寄存器代码域中的代码作为字符串的压缩代码被传送(34)，而寄存器的内容被写入到存储器的下一个空位置中。 在下一个周期，通过将寄存器代码域置空来进行初始化，然后重复所描述的步骤。



ISSN1008-4274

1. 一种数据压缩方法，用于将数据字符信号输入流压缩为压缩代码信号流，所述数据字符信号属于包括 A 个字符的数据字符信号字符表，其特征在于:

(a). 使用具有多个位置的相联存储器以存储数据字符信号的字符串，每个位置具有一个前缀代码域和一个字符域，每个位置具有一个与之相关联的地址，该地址为每个存储字符串提供一个压缩代码信号，

(b). 通过把所述存储器 A 个位置的前缀代码域置空，初始化所述存储器以包含所述字符表的 A 个单字符串并将所述字符表的数据字符信号分别插入所述 A 个位置的字符域中，

(c). 使用具有一个代码域和一个字符域的寄存器，

(d). 置空所述寄存器的所述代码域并将一个输入数据字符插入所述寄存器的所述字符域，

(e). 相联地将所述寄存器中的内容与所述存储器位置中的内容相比较，以确定与所述存储器位置中内容的一个匹配，

(f). 若确定了一个匹配，则将与匹配位置相关联的地址插入所述寄存器的所述代码域并且将所述输入流的下一数据字符插入所述寄存器的所述字符域，

(g). 重复步骤(e)和(f)，直到确定没有匹配，从而在所述存储器中找到与所述输入流相匹配的最长存储字符串，

(h). 确定在步骤(e)没有匹配时，提供所述寄存器的所述代码域的内容作为一个压缩代码信号，从而提供所述最长匹配存储字符串的压缩代码信号，

(i). 分配顺序地址以存取所述存储器的顺序空位置，以提供下一个空位置，所述顺序地址从 A+1 开始，将所述寄存器的所述代码域和所述字符域的内容分别写入所述存储器中所述下一个空位置中的前缀代码域和字符域，从而在所述存储器中插入一个扩充字符串，该扩充字符串由所述最长匹配存储字符串组成，该最长匹配字符串是所述输入流中下一个数据

字符信号扩充得到的，所述下一个空位置的地址为插入所述存储器的所述扩充字符串提供压缩代码信号。

(j). 将所述扩充字符串插入所述存储器后，将所述寄存器的代码域置空(40)，及

(k). 重复步骤(e)至(j)，直到没有可以被压缩的数据字符信号输入流(32)。

2. 根据权利要求 1 所述的方法，其中所述初始化步骤进一步包括：除所述 A 个位置外，插入所述存储器位置的字符域，随意位模式不被认为是所述字符表的一个数据字符信号。

3. 根据权利要求 1 所述的方法，进一步包括一种数据解压缩方法(图 2)，用于对所述压缩代码信号流执行解压缩以恢复与之相应的输入数据字符信号流，其中下述步骤(o)到(y)包括一个解压缩周期，所述数据解压缩方法包括：

(l). 使用一个解压缩存储器，其具有多个用于存储数据字符信号的位置，每个位置具有一个前缀代码域和一个字符域，每个位置具有一个与之相关联的地址，该地址为存储在该解压缩存储器中的每个字符串提供一个压缩代码信号，

(m). 通过把所述解压缩存储器 A 位置的前缀代码域置空，初始化所述解压缩存储器以包括所述字符表的 A 个单字符串并将所述字符表的数据字符信号分别插入所述 A 个位置的字符域中，

(n). 使用一个地址寄存器以存取所述解压缩存储器的所述位置，

(o). 接收一个压缩代码信号，将其放入一输入代码寄存器，

(p). 将所述输入代码寄存器的内容传输给所述地址寄存器，

(q). 使用一前置代码寄存器以存储在当前解压缩周期之前的解压缩周期中接收到的压缩代码信号，

(r). 存取与所述地址寄存器的内容相对应的所述解压缩存储器的位置，

(s). 将所存取位置的字符域的内容插入一个堆栈，从而将所存取位置的字符域中的数据字符信号插入所述堆栈，

(t). 将所存取位置的前缀代码域的内容插入所述地址寄存器,

(u). 重复步骤(r)至(t), 直到所述地址寄存器中的内容为空, 从而将与所接收的压缩代码信号相对应的数据字符信号插入所述堆栈,

(v). 分配顺序地址以存取所述解压缩存储器的顺序空位置, 以提供下一个空位置的地址, 所述顺序地址从 A+1 开始, 将所述下一个空位置的所述地址插入所述地址寄存器,

(w). 将一个更新前缀代码和一个更新字符分别写入所述地址寄存器所存取的所述解压缩存储器的位置的前缀代码域和字符域, 该更新前缀代码由所述前置代码寄存器提供, 该更新字符由插入所述堆栈的最后一个字符信号提供, 从而将与插入所述相联存储器的扩充字符串相对应的扩充字符串插入所述解压缩存储器, 所述下一个空位置的地址为插入所述解压缩存储器的所述扩充字符串提供压缩代码信号,

(x). 输出所述堆栈的内容, 从而恢复与所接收的压缩代码信号相对应的数据字符信号的字符串,

(y). 将所接收的所述输入代码寄存器中压缩代码信号传输给所述前置代码存储器,

(z). 重复步骤(o)至(y)直到再没有可被解压缩的压缩代码信号流。

4. 根据权利要求 3 所述的方法, 其中所述输出步骤包括以将数据字符信号插入所述堆栈的顺序相反的顺序从所述堆栈中输出数据字符信号。

5. 根据权利要求 3 所述的方法, 其中所述数据解压缩方法包括一个异常处理方法, 该异常处理情况是当所接收到的压缩代码信号没有相应的存储在所述解压缩存储器中的字符串从而造成所述所接收的压缩代码信号不可识别时发生的, 所述异常处理方法包括:

生成一个异常扩充字符串, 其包括与前置代码寄存器中的压缩代码信号相对应的字符串, 该压缩代码信号由所述更新字符扩充得到,

输出所述异常扩充字符串, 从而输出与所述不可识别的压缩代码信号相对应的字符串, 及

在所述解压缩存储器中存储所述异常扩充字符串, 所述不可识别的压缩代码信号提供与所述所存储的异常扩充字符串相对应的压缩代码信号,

其中所述异常处理方法包括:

将所述更新字符插入所述堆栈,

将所述前置代码寄存器中的内容传输给所述地址寄存器,

执行步骤(r)至(x),从而生成所述异常扩充字符串,输出所述异常扩充字符串并以步骤(i)的地址在所述解压缩存储器中存储所述异常扩充字符串,从而为所述异常扩充字符串提供不可识别的压缩代码信号,

继续步骤(y)的所述解压缩方法。

6. 根据权利要求5所述的方法,其中步骤(e)至(j)包括一个压缩周期,并且所述压缩方法包括在执行前一个压缩周期之后执行当前压缩周期,在当前压缩周期中,所述压缩方法在为上一个压缩周期步骤(i)中所插入的所述相联存储器的扩充字符串提供压缩代码信号的同时,所述压缩方法也提供了所述不可识别的压缩代码信号。

7. 根据权利要求5所述的方法,进一步包括:

使用一个用于分配顺序地址的地址计数器以存取所述解压缩存储器的顺序空位置,从而提供步骤(v)中所述下一空位置的地址,所述顺序空位置从A+1开始,

根据对所述输入代码寄存器中所接收的压缩代码信号和所述地址计数器的内容相比较的结果调用所述异常处理方法。

8. 一种数据压缩装置,用于将数据字符信号输入流压缩为压缩代码信号流,所述数据字符信号属于包括A个字符的数据字符信号字符表,其特征在于:

(a). 一个具有多个位置的相联存储器,用于存储数据字符信号的字符串,每个位置具有一个前缀代码域和一个字符域,每个位置具有一个与之相关联的地址,该地址为每个存储字符串提供一个压缩代码信号,

(b). 通过把所述存储器A个位置的前缀代码域置空,初始化所述存储器以包含所述字符表的A个单字符串并将所述字符表的数据字符信号分别插入所述A个位置的字符域的位置,

(c). 具有一个代码域和一个字符域的一个寄存器,

(d). 置空所述寄存器的所述代码域并将一个输入数据字符插入所述

寄存器的所述字符域的装置，

(e). 与所述存储器和所述寄存器相联接的控制装置，用于操作所述存储器，相联地将所述寄存器中的内容与所述存储器位置中的内容相比较，以确定与所述存储器位置中内容的一个匹配，

(f). 所述控制装置，用于若确定了一个匹配，则将与匹配位置相关联的地址插入所述寄存器的所述代码域并且将所述输入流的下一数据字符插入所述寄存器的所述字符域，

(g). 所述控制装置，用于重复步骤(e)和(f)，直到确定没有匹配，从而在所述存储器中找到与所述输入流相匹配的最长存储字符串，

(h). 所述控制装置，进一步用于确定在步骤(e)没有匹配时，提供所述寄存器的所述代码域的内容作为一个压缩代码信号，从而提供所述最长匹配存储字符串的压缩代码信号，

(i). 一个地址计数器，用于分配顺序地址以访问所述存储器的顺序空位置以提供下一个空位置，所述顺序地址从 A+1 开始，所述控制装置，进一步用于将所述寄存器的所述代码域和所述字符域的内容分别写入所述存储器中所述下一个空位置中的前缀代码域和字符域，从而在所述存储器中插入一个扩充字符串，该扩充字符串包括所述字符信号扩充得到的，所述下一个空位置的地址为插入所述存储器的所述扩充字符串提供压缩代码信号，

(j). 将所述寄存器的代码域置空的装置，及

(k). 所述操作的控制装置，用于重复步骤(e)至(j)，直到再没有可以被压缩的数据字符信号输入流。

9. 根据权利要求 8 所述的装置，其中所述初始化装置进一步包括：除所述 A 个位置外，插入所述存储器的位置的字符域的装置，随意拉模式不被认为是所述字符表的一个数据字符信号。

10. 根据权利要求 8 所述的装置，进一步包括一种数据解压缩装置，用于对所述压缩代码信号流执行解压缩以恢复与之相应的输入数据字符信号流，其中下述步骤(o)到(y)包括一个解压缩周期，所述数据解压缩装置包括：

(l). 一个解压缩存储器，其具有多个用于存储数据字符信号的位置，每个位置具有一个前缀代码域和一个字符域，每个位置具有一个与之相关联的地址，该地址为存储在该解压缩存储器中的每个字符串提供一个压缩代码信号，

(m). 通过把所述解压缩存储器 A 个位置的前缀代码域置空，初始化所述解压缩存储器以包括所述字符表的 A 个单字符串并将所述字符表的数据字符信号分别插入所述 A 个位置的字符域的装置，

(n). 一个地址寄存器，用于存取所述解压缩存储器的所述位置，

(o). 一个输入代码寄存器，用于接收一个压缩代码信号，

(p). 将所述输入代码寄存器的内容传输给所述地址寄存器的装置，

(q). 一个前置代码寄存器，用于存储在当前解压缩周期之前的解压缩周期中接收到的压缩代码信号，

(r).一个堆栈，

(s). 与所述解压缩存储器，所述地址寄存器，所述输入代码寄存器，所述前置代码寄存器及所述堆栈相连接的解压缩控制装置，用于操作所述解压缩存储器，存取与所述地址寄存器的内容相对应的所述解压缩存储器的位置，

(t). 所述解压缩控制装置，用于将所存取的位置的字符域的内容插入所述堆栈，从而将所存储位置中的字符域中的数据字符信号插入所述堆栈，

(u). 所述解压缩控制装置，用于将所存取位置的前缀代码域的内容(61)插入所述地址寄存器，

(v). 所述解压缩控制装置，用于重复步骤(s)至(u)，直到所述地址寄存器中的内容为空，从而将与所接收的压缩代码信号相对应的数据字符信号插入所述堆栈，

(w). 一个地址计数器，用于分配顺序地址以访问所述解压缩存储器的顺序空位置，从而提供下一个空位置的地址，所述顺序地址从 A+1 开始，所述解压缩控制装置，进一步用于将所述下一个空位置的地址插入所述地址寄存器，并且将一个更新前缀代码和一个更新字符分别写入所述地

址寄存器所存取的所述解压缩存储器的位置的前缀代码域和字符域，该更新前缀代码由所述前置代码寄存器提供，该更新字符由插入所述堆栈的最后一个字符信号提供，从而将与插入所述相联存储器的扩充字符串相对应的扩充字符串插入所述解压缩存储器，所述下一个空位置的地址为插入所述解压缩存储器的所述扩充字符串提供压缩代码信号，

(x). 所述解压缩控制装置，进一步用于输出所述堆栈的内容，从而恢复与所接收的压缩代码信号相对应的数据字符信号的字符串，

(y). 所述解压缩控制装置，进一步用于将所接收的所述输入代码寄存器中压缩代码信号传输给所述前置代码存储器，

(z). 所述解压缩控制装置，用于重复步骤(o)至(y)，直到再没有可被解压缩的压缩代码信号流。

11. 根据权利要求 10 所述的装置，其中所述解压缩控制装置进一步用于以将数据字符信号插入所述堆栈的顺序相反的顺序从所述堆栈中输出数据字符信号。

12. 根据权利要求 10 所述的装置，其中在异常处理模式中，所述解压缩控制装置用于操作所述解压缩装置，该异常处理模式是由下面的异常情况发生时被调用的，即当所接收到的压缩代码信号没有相应的存储在所述解压缩存储器中的字符串，从而造成所述所接收的压缩代码信号不可识别，在所述异常处理模式中所述解压缩控制装置用于:

生成一个异常扩充字符串，其包括与前置代码寄存器中的压缩代码信号相对应的字符串，该压缩代码信号由所述更新字符扩充得到，

输出异常扩充字符串，从而输出与所述不可识别的压缩代码信号相对应的字符串，及

在所述解压缩存储器中存储所述异常扩充字符串，所述不可识别的压缩代码信号提供与所述所存储的异常扩充字符串相对应的压缩代码信号，其中，在所述异常处理模式中所述解压缩控制装置用于:

将所述更新字符插入所述堆栈，

将所述前置代码寄存器中的内容传输给所述地址寄存器，

从而生成所述异常扩充字符串，输出所述异常扩充字符串并

执行步骤(s)至(x)，从而生成所述异常扩充字符串，输出所述异常扩充字符串并以步骤(v)的地址在的述解压缩存储器中存储所述异常扩充字符串，从而为所述异常扩充字符串提供不可识别的压缩代码信号，及

继续步骤(y)的所述解压缩周期。

13. 根据权利要求 12 所述的装置，其中步骤(e)至(j)定义了一个压缩周期，所述压缩装置包括在执行前一个压缩周期之后执行当前压缩周期，在当前压缩周期中，所述压缩装置在为上一个压缩周期步骤(i)中所插入的所述相联存储器的扩充字符串提供压缩代码信号的同时，所述压缩装置也提供了所述不可识别的压缩代码信号。

14. 根据权利要求 12 所述的装置，进一步包括:

一个地址计数器，用于分配顺序地址以存取所述解压缩存储器的顺序空位置，从而提供步骤(w)中所述下一空位置的地址，所述顺序空位置从 A+1 开始，及

用于对所述输入代码寄存器中所接收的压缩代码信号和所述地址计数器的内容相比较的装置，所述解压缩控制存储器根据对所述输入代码寄存器中所接收的压缩代码信号和所述地址计数器的内容相比较的结果调用所述异常处理模式。

数据压缩方法和装置

技术领域

本发明涉及数据压缩和解压缩。

技术背景

LZW 是一种相当常用的进行数据压缩和解压缩的方法，例如符合调制解调器通信标准 CCITT V. 42bis 的应用就用到了这一方法。在 1985 年 11 月 10 日授权给 Terry A.Welch 的名称为“高速数据压缩和解压缩装置和方法”的美国专利 4,558,302 中对 LZW 进行了描述。在这里，所述专利 4，558，302 被作为对比文件使用并转让给了本发明的受让人。

LZW 数据压缩法使用了一个字典，该字典用来存储输入的数据字符串，并以比较输入流和存储在字典中的字符串来确定最长匹配(longest match)的方式来搜索输入流。该字典通过存储包含最长匹配的字符串得以不断扩充，而该最长匹配字符串由进行最长匹配后紧接到达的输入数据字符进行扩充，而该最长匹配字符串由进行最长匹配后紧接到达的输入数据字符进行扩充。数据压缩字典以前均用随机存取内存(RAM)实现。Welch 在专利 4,558,302(第 30-34 行，第 52 列)中提出使用按内容寻址存储器或相联存储器而不是使用 RAM，因为这样可以减少控制复杂性。但是，Welch 并没有描述如何实现该方法。因此可以相信，在此之前的现有技术未提供关于 LZW 压缩算法的相联存储器实施例。

另一方面，1982 年 12 月 28 日授权给 Klaus E.Holtz 名称为“相联存储器搜索系统”的美国专利 4,366,551 中公开了一种使用相联存储器进行存储和搜索的系统。但是，所述专利 4,366,551 未公开或提出 LZW 算法的相联存储器实施例。根据 1994 年元月 4 日的重新审查(re-examination)证书 B4,558,302，所述专利 4,366,551 被作为对比文件在所述专利 4,558,302 中提出。

通过将寄存器中所含前缀代码、字符对的内容同相联存储器中存储的多个前缀代码、字码对相比较，数据字符信号流被压缩为压缩代码信号流。寄存器的字符部分按数据字符信号从数据字符信号输入流中接收的顺序保持数据字符信号。如果上述比较结果为命中(Hit)，则用命中地址替代寄

存器中的前缀代码，并用下一数据字符信号替代寄存器中的字符。这一过程不断重复直到未命中(Miss)发生，在每次重复过程中，字符串中的前缀代码被作为压缩代码信号传送。地址计数器提供相联存储器中下一个可用空位置的地址并将寄存器中的内容存储在该位置，同时地址计数器递增。

发明内容

本发明提供一种数据压缩方法，用于将数据字符信号输入流压缩为压缩代码信号流，所述数据字符信号属于包括 A 个字符的数据字符信号字符表，其特征在于:

(a). 使用具有多个位置的相联存储器以存储数据字符信号的字符串，每个位置具有一个前缀代码域和一个字符域，每个位置具有一个与之相关联的地址，该地址为每个存储字符串提供一个压缩代码信号，

(b). 通过把所述存储器 A 个位置的前缀代码域置空，初始化所述存储器以包含所述字符表的 A 个单字符串并将所述字符表的数据字符信号分别插入所述 A 个位置的字符域中，

(c). 使用具有一个代码域和一个字符域的寄存器，

(d). 置空所述寄存器的所述代码域并将一个输入数据字符插入所述寄存器的所述字符域，

(e). 相联地将所述寄存器中的内容与所述存储器位置中的内容相比较，以确定与所述存储器位置中内容的一个匹配，

(f). 若确定了一个匹配，则将与匹配位置相关联的地址插入所述寄存器的所述代码域并且将所述输入流的下一数据字符插入所述寄存器的所述字符域，

(g). 重复步骤(e)和(f)，直到确定没有匹配，从而在所述存储器中找到与所述输入流相匹配的最长存储字符串，

(h). 确定在步骤(e)没有匹配时，提供所述寄存器的所述代码域的内容作为一个压缩代码信号，从而提供所述最长匹配存储字符串的压缩代码信号，

(i). 分配顺序地址以存取所述存储器的顺序空位置，以提供下一个空位置，将所述寄存器的所述代码域和所述字符域的内容分别写入所述存储器中所述下一个空位置中的前缀代码域和字符域，从而在所述存储器中插入一个扩充字符串，该扩充字符串由所述最长匹配存储字符串组成，该最长匹配字符串是所述输入流中下一个数据字符信号扩充得到的，所述下一

个空位置的地址为插入所述存储器的所述扩充字符串提供压缩代码信号。

(j). 将所述扩充字符串插入所述存储器后，将所述寄存器的代码域置空，及

(k). 重复步骤(e)至(j)，直到没有可以被压缩的数据字符信号输入流。

本发明提供一种数据压缩装置，用于将字符信号输入流压缩为压缩代码信号流，所述数据字符信号属于包括 A 个字符的数据字符信号字符表，其特征在于:

(a). 一个具有多个位置的相联存储器，用于存储数据字符信号的字符串，每个位置具有一个前缀代码域和一个字符域，每个位置具有一个与之相关联的地址，该地址为每个存储字符串提供一个压缩代码信号，

(b). 通过把所述存储器 A 个位置的前缀代码域置空，初始化所述存储器以包含所述字符表的 A 个单字符串并将所述字符表的数据字符信号分别插入所述 A 个位置的字符域的位置，

(c). 具有一个代码域和一个字符域的一个寄存器，

(d). 置空所述寄存器的所述代码域并将一个输入数据字符插入所述寄存器的所述字符域的装置，

(e). 与所述存储器和所述寄存器相联接的控制装置，用于操作所述存储器，相联地将所述寄存器中的内容与所述存储器位置中的内容相比较，以确定与所述存储器位置中内容的一个匹配，

(f). 所述控制装置，用于若确定了一个匹配，则将与匹配位置相关联的地址插入所述寄存器的所述代码域并且将所述输入流的下一数据字符插入所述寄存器的所述字符域，

(g). 所述控制装置，用于重复步骤(e)和(f)，直到确定没有匹配，从而在所述存储器中找到与所述输入流相匹配的最长存储字符串，

(h). 所述控制装置，进一步用于确定在步骤(e)没有匹配时，提供所述寄存器的所述代码域的内容作为一个压缩代码信号，从而提供所述最长匹配存储字符串的压缩代码信号，

(i). 一个地址计数器，用于分配顺序地址以访问所述存储器的顺序空位置以提供下一个空位置，所述控制装置，进一步用于将所述寄存器的所述代码域和所述字符域的内容分别写入所述存储器中所述下一个空位置中的前缀代码域和字符域，从而在所述存储器中插入一个扩充字符串，该扩充字符串包括所述字符信号扩充得到的，所述下一个空位置的地址为插

入所述存储器的所述扩充字符串提供压缩代码信号，

(j). 将所述寄存器的代码域置空的装置，及

(k). 所述操作的控制装置，用于重复步骤(e)至(j)，直到再没有可以被压缩的数据字符信号输入流。

附图说明

图 1 是根据本发明实现的数据压缩器原理方框图。

图 2 是对图 1 装置的输出进行解压缩的解压缩器原理方框图。

本发明实施例用到的字典，既可以在初始化时包含所有的单字符串，也可以在初始化时仅包含空字符串。下面首先描述初始化方式为单字符串下的实施例。

具体实施方式

图 1 示出了根据本发明进行配置的数据压缩器 10。数据压缩器 10 包括一个按内容寻址的存储器 11，该存储器具有 N 个存储位置，每个位置具有一个用来存储前缀代码的域 12 和一个用来存储字符的域 13。存储器 11 进一步包括一个用于标识存储器存储位置地址的部分 14。

压缩器 10 对数据字符信号流进行压缩，该数据字符信号属于具有 A 个字符的字符表。例如，如果使用 ASCII 码表示，则可以使用的字符表大小为 256。在实施例为单字符串初始方式的数据压缩器 10 中，存储器 11 前面 A 个位置经初始化后包含 A 个单字符串，对于存储某单字符串的某位置上的域 12，其所含前缀代码被置为 0，域 13 以二进制形式存储所述字符。例如，若使用 ASCII 码，字符域 13 的宽度为 8 位，前缀代码域 12 含有足够的位数以容纳存储器 11 的 N 个存储位置。

存储器 11 的存储位置从 A+1 开始，通过将字符域 13 的所有内容重新设置为随意位模式(arbitrary bit pattern)进行初始化，因此其不能作为字符表中的任何字符来识别。

数据压缩器 10 进一步包括一个寄存器 20，该寄存器具有一个用于存储一个代码的域 21 和一个用于存储一个字符的域 22。存储器 11 可以在相联或读模式下进行操作，操作时把寄存器 20 中的内容同存储器 11 的内容相比较。这一操作用参考标记 23 表示，如果寄存器 20 的内容同存储器 11 中某个存储位置的内容匹配，则命中/未命中(Hit/Miss)输出 24 发出一个命中信号。命中发生处的存储位置的地址由地址部分 14 的输出 25 提供，输

出 25 为寄存器 20 的代码域 21 提供输入。如果在存储器 11 中没有找到寄存器 20 的内容，则输出 24 发出一个未命中指示。

存储器 11 也可以在写模式下操作，操作时寄存器 20 的代码域 21 的内容和字符域 22 的内容将被分别写入存储器 11 某个位置的前缀域 12 和字符域 13 中，存储器 11 的这一写入位置的地址由地址输入 26 提供，地址输入 26 所提供的存储器地址来自地址计数器 31 。写模式下存储器 11 的代码和字符输入分别被标记为标号 27 和 28 。存储器 11 的写/读模式由输入 30 选择。

待压缩字符的输入数据流在 32 输入，通过数据寄存器 33 进入到寄存器 20 的字符域 22 。由寄存器 20 的代码域 21 提供的，经数据压缩器 10 压缩后的代码通过输出方框 34 输出。空代码输入 40 用于将寄存器 20 的代码域 21 置零。

控制逻辑 41 为数据压缩器的所有部件提供输入，如 42 所示。控制逻辑 41 接收存储器输出 24 上的命中/未命中信号并通过存储器输入 30 提供对存储器 11 的写/读控制。

在数据压缩器 10 的单字符串初始化实施例的操作中，存储器 11 的前 A 个位置经初始化后存储所有可能的单字符串。在这些经初始化的存储位置中，前缀代码域 12 被置为 0 ，字符域 13 被置为字符表中各个字符的二进制形式，地址计数器 31 被置为 A +1 。待压缩的输入字符流由输入 32 提供，在输入数据寄存器 33 中进行缓冲。

数据压缩器 10 的一个工作周期如下所述。

寄存器 20 的代码域 21 由空代码 40 置为 0 。字符域 22 存储上一个周期中引起未命中指示的那个字符。但是，如果数据压缩器 10 开始的是其第一个周期，则来自输入数据寄存器 33 的输入流的第一个字符被放置在字符域 22 中。

控制逻辑 41 通过输入 30 控制存储器 11 操作于相联模式下。寄存器 20 的内容与存储器 11 的内容通过路径 23 进行比较，如果匹配则输出 24 登记一次命中并将其送往控制逻辑 41 。产生命中的地址被装入寄存器 20 的代码域 21 中并且下一个输入字符被调入字符域 22 。这一过程不断重复，直到输出 24 上登记一次未命中并送往控制逻辑 41 。当此情况发生时，驻留在寄存器 20 域 21 中的代码通过输出方框 34，作为该周期的压缩代码输出。

接着，控制逻辑 41 通过控制输入 30 控制存储器 11 操作于写模式下，将来自输入 27 的代码和来自输入 28 的字符写入到由地址计数器 31 所标识的存储位置的前缀代码域 12 和字符域 13 中。然后地址计数器 31 加 1 并且代码域 21 通过空代码 40 置为 0 。

这样，一个压缩周期完成，数据压缩器 10 准备进行下一个周期。

控制逻辑 41 通过为数据压缩器 10 的所有部件提供控制信号来控制上面描述的操作。控制逻辑 41 可以方便地用常规状态机器来实现。

通过对上面操作周期的描述，输入流中的输入字符串被数据压缩器 10 接收后，与存储器 11 的内容进行比较直到得到输入的最长匹配。这一最长匹配的前缀代码被输出并且通过存储一个扩充字符串更新存储器，存储器 11 中的该扩充字符串包括由下一个输入流字符扩充的最长匹配。

这样，数据压缩器 10 在没有通常 RAM 搜索方法所带来的额外开销的情况下实现了 LZW 压缩，而且通过比较寄存器 20 和存储器 11 的内容实现了按内容寻址比较。

图 2 示出了对图 1 所示的数据压缩器 10 的输出进行解压缩的数据解压缩器 50 。数据解压缩器 50 接收图 1 输出方框 34 输出的压缩代码并且恢复相应的数据字符流。解压缩器 50 使用 RAM 51 的方式同专利 4,558,302 所描述的方式相似。解压缩器 50 的构造和操作同专利 4,558,302 的图 5 所示的方式相似。

输入 52 接收经压缩的代码并将其放在输入代码寄存器 53 中。寄存器 53 中的输入代码被提供给 RAM 地址寄存器 54 ，依此来存取 RAM 51 中由 RAM 地址寄存器 54 中的压缩代码所表示的存储位置。 RAM 51 中的每个存储位置包括一个前缀代码域 55 和一个字符域 56 。

同上面图 1 所描述的方式类似， RAM 51 在初始化时包含所有的单字符串。因此， RAM 51 的前 A 个位置被初始化，前缀代码域 55 存储 0 ，字符域 56 存储字符表各个字符的二进制表示。

解压缩器 50 也包括一个地址计数器 60 ，该计数器在解压缩操作初始化时置初始值 A +1 。地址计数器 60 的输出将作为 RAM 地址寄存器 54 的输入来存取 RAM 51 。 RAM 51 同图 1 存储器 11 的 N 个存储位置相对应也包含 N 个位置。

RAM 51 操作于读模式下时字符串被恢复，操作于写模式下时 RAM 51 被更新。在读模式下，由 RAM 地址寄存器 54 的地址标识的被存取位置上的前缀代码通过路径 61 提供，而来自被存取位置上的字符则通过路径 63 压入堆栈 62 。路径 61 上的前缀代码作为 RAM 地址寄存器 54 的输入进行使用。堆栈 62 用来保持被恢复的字符，该字符在输出 64 按顺序弹出堆栈。

在 RAM 51 的写模式下，由前置代码寄存器(prior code register)70 提供的代码通过路径 71 被写入到由 RAM 地址寄存器 54 所标识的存取位置的前缀代码域 55 。堆栈 62 提供的字符通过输入 72 被写入所存取位置的字符域 56 中。当一个解压缩周期完成时，输入代码寄存器 53 中的代码被输往前置代码寄存器 70 。

解压缩器 50 进一步包括控制逻辑 73，它为解压缩器 50 的所有部件提供控制输入，如标号 74 所示。零检测器 75 检测 RAM 51 的前缀代码输出 61 何时为 0 并将这一指示信号通过路径 76 送给控制逻辑 73 。

为了提供被描述的“异常情况”处理，解压缩器 50 包括一个比较器 80，该比较器比较输入代码寄存器 53 中的代码和地址计数器 60 的内容，当两者相等时将提供一个指示并将该指示通过路径 81 送给控制逻辑 73 。

在操作过程中，解压缩器 50 为在输入 52 接到的每个压缩代码执行一个解压缩周期进行恢复，并在输出 64 提供同该代码相应的字符串。正常的解压缩周期过程描述如下。

寄存器 53 中的输入代码被送往 RAM 地址寄存器 54 以存取 RAM 51 。控制逻辑 73 控制 RAM 51 处于读模式。存储在被存取位置上的字符由输出 63 中读出并压入堆栈 62 。被存取位置上的前缀代码由输出 61 中读出，然后送到 RAM 地址寄存器 54，以此来定位下一个被存取位置。这一过程持续进行直到零检测器 75 检测到所读到的前缀代码为 0，这时，压入堆栈 62 的字符串从输出 64 反序弹出，提供与在输入 52 接收到的压缩代码相对应的恢复字符串。

控制逻辑 73 接着控制 RAM 51 处于写模式下并将前置代码寄存器 70 的内容写入地址计数器 60 所存取的 RAM 存储位置中。位于堆栈 62 顶部的字符通过堆栈输出 72 被写入到该存取位置的的字符域 56 中。写入字符域 56 的字符是当前所恢复字符串的第一个字符，也是正被存储的扩充字符串的扩

充字符。

在解压缩周期最后，地址计数器 60 加 1 ，输入代码寄存器中的代码被送到前置代码寄存器 70 ，解压缩器 50 接着准备接收下一个代码。

在解压缩器 50 的第一个周期并不执行写操作，因为此时前置代码寄存器 70 中没有前置代码。另外，地址计数器 60 在这一周期中并不增值。

当图 1 的压缩器输出存储在上一个压缩周期中的代码字符串时会发生一种“异常情况”。此时，不能识别解压缩器接收到的压缩代码，因为解压缩器在接收该压缩代码时还没有存储该代码字符串。当寄存器 53 中所接收到的输入压缩代码同地址计数器 60 相等时，上述异常情况发生。

异常情况按如下方式处理：前置代码寄存器 70 中的代码通过路径 90 被转往 RAM 地址寄存器 54 。堆栈 62 是专利 4,558,302 所述的类型，从堆栈中弹出的最后一个字符仍然驻留在栈顶寄存器中。对于正常处理，该字符可提供扩充字符，接着在输入 63 接收到字符时该字符被覆盖。对于异常情况处理，上述字符被压入堆栈，紧随其后的是同 RAM 地址寄存器 54 中所驻留的代码对应的被恢复字符串。然后这个字符串从堆栈 62 中弹出，在输出 64 提供被恢复的字符串。接着地址计数器 60 通过 RAM 地址寄存器 54 来存取 RAM 51 ，堆栈 62 顶部的字符则被写入所存取位置的字符域 56 中，而前置代码寄存器 70 中的代码则被写入前缀代码域 55 中。然后，输入代码寄存器 53 中的代码被传送到前置代码寄存器 70 ，并且地址计数器 60 递增，异常情况处理周期完成。

从前述可以了解到，利用类似专利 4,558,302 所描述的方式，对应每个输入代码，字符串以反序得到恢复，然后利用堆栈 62 再次反序，以正确的顺序提供所恢复的字符串。

上面所描述的发明实施例是根据下面的前提进行解释的，即用所有的单字符串初始化图 1 中的存储器 11 和图 2 中的 RAM 51 。由前可知，本发明也可以使用空字符串初始化方式来实现实施例。在这种实施例中，整个存储器 11 和 51 被置空，地址计数器 31 和 60 以 1 开始计数。整个处理以上面所描述的方式进行，除了首次遇到的字符以非压缩方式进行传送以便解压缩器和压缩器保持同步以外，这可以通过压缩器 10 传送一个零代码，然后在后面接可识别的、交由解压缩器 50 恢复的字符来实现。在这一实施例

中，零代码可由零检测器在输入代码寄存器 53 检测到。

零代码和字符可以通过从寄存器 20 的字符域到输出方框 34 的路径来传送。输出方框 34 将把来自寄存器 20 的域 21 的零代码和来自域 22 的字符组装在一起作为传送输出。另外，可以对图 2 的输入代码寄存器 53 进行修改，使之通过堆栈 62 提供送往输出 64 的单字符。该字符和零前缀代码被存于 RAM 51 中。地址寄存器 60 适当递量以适应同上面描述的单字符串初始化实施例的这些不同。

上面所描述的实施例可以用软件、固件(firmware)、逻辑、硬件及其他类似或它们的组合来实现。

图 1

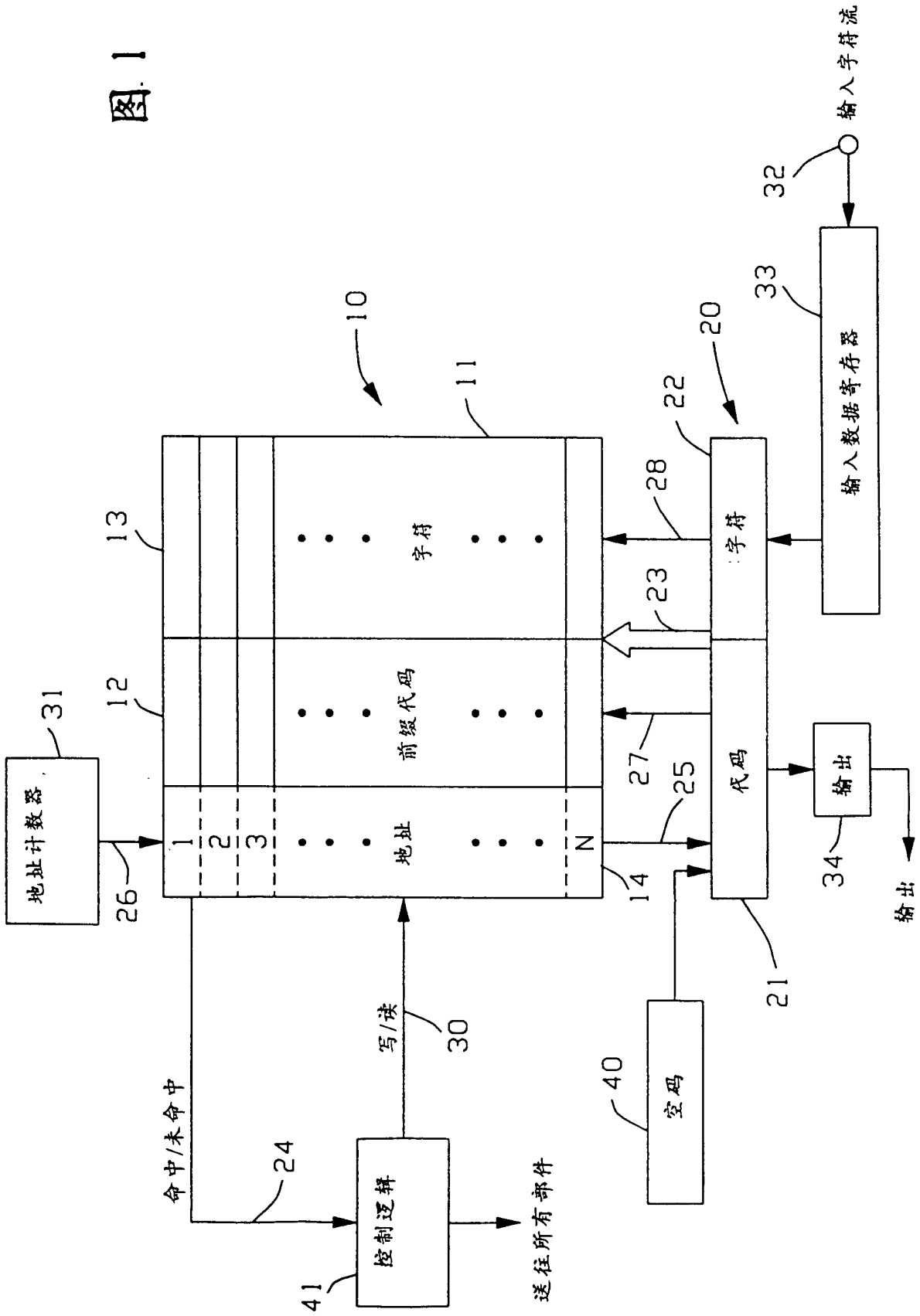
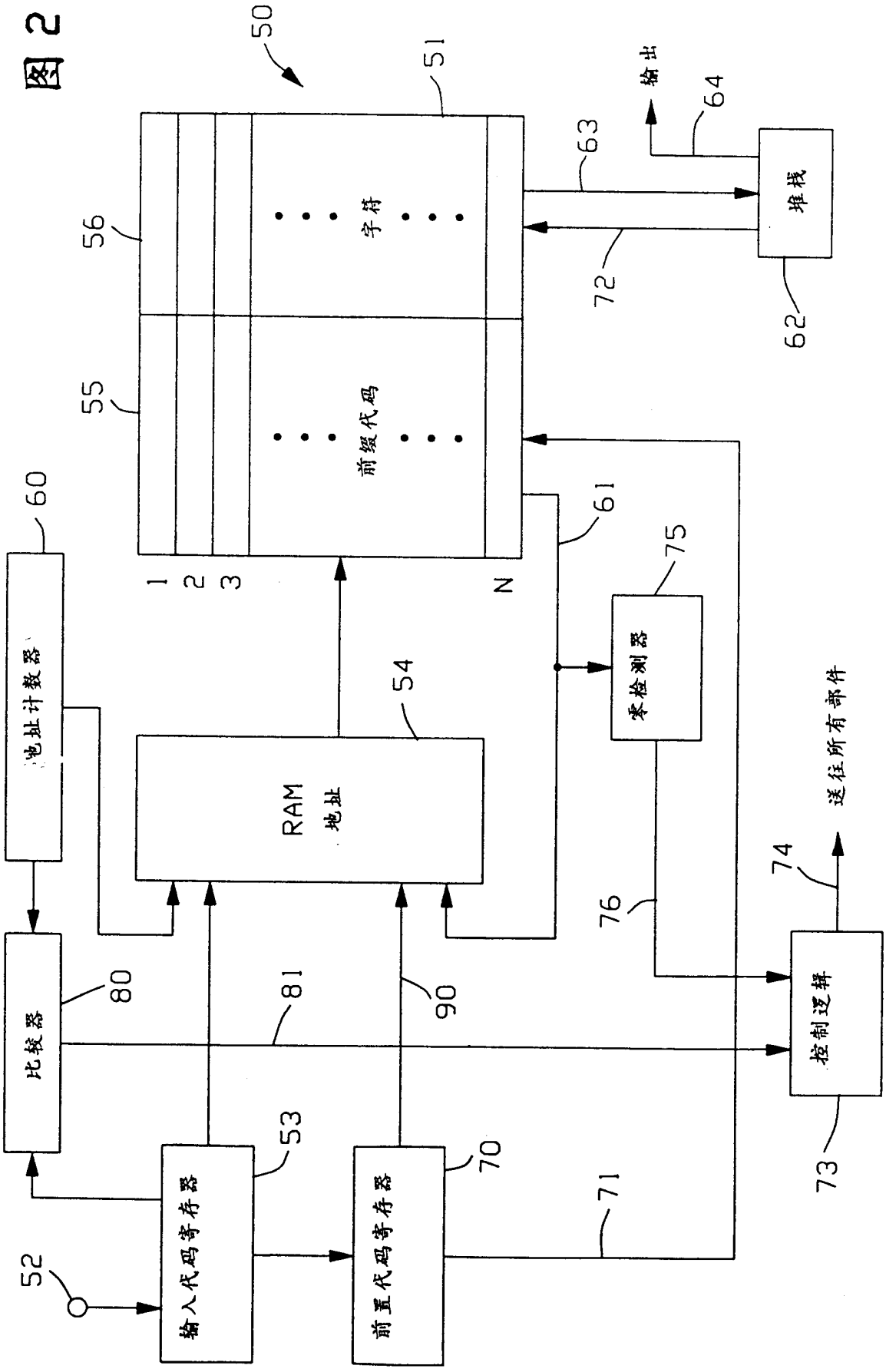

输入字符流
32
33
输入数据寄存器
20
22
字符
28
23
10
11
13
字符
前缀代码
12
31
地址计数器
1
2
3
地址
N
26
14
27
25
代码
输出
34
输出
21
40
空码
写/读
30
命中/未命中
24
控制逻辑
41
送往所有部件

图 2


50
51
字符
前缀代码
1
2
3
N
55
56
63
64
输出
堆栈
72
62
60
地址计数器
61
75
零检测器
54
RAM
地址
76
74
送往所有部件
比较器
80
81
90
控制逻辑
73
53
70
71
52
输入代码寄存器
前置代码寄存器