

[19] 中华人民共和国国家知识产权局

[51] Int. Cl.

G06F 9/45 (2006.01)



[12] 发明专利说明书

专利号 ZL 200380106165.7

[45] 授权公告日 2008 年 11 月 5 日

[11] 授权公告号 CN 100430896C

[22] 申请日 2003. 10. 3

[21] 申请号 200380106165.7

[30] 优先权

[32] 2002. 10. 29 [33] US [31] 60/421,773

[32] 2002. 10. 29 [33] US [31] 60/421,774

[32] 2002. 10. 29 [33] US [31] 60/421,775

[32] 2002. 12. 31 [33] US [31] 10/331,315

[86] 国际申请 PCT/US2003/031314 2003. 10. 3

[87] 国际公布 WO2004/040446 英 2004. 5. 13

[85] 进入国家阶段日期 2005. 6. 15

[73] 专利权人 洛克希德马丁公司

地址 美国马里兰州

[72] 发明人 迈克尔·C·达普

埃里克·C·莱特

[56] 参考文献

US5995963A 1999. 11. 30

CN1173676A 1998. 2. 18

CN1118899A 1996. 3. 20

US5414833A 1995. 5. 9

审查员 魏 峰

[74] 专利代理机构 北京英赛嘉华知识产权代理有限公司

代理人 余 滕 方 挺

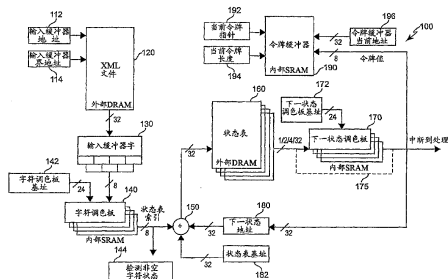
权利要求书 2 页 说明书 12 页 附图 6 页

[54] 发明名称

硬件解析器加速器

[57] 摘要

使用专用硬件以在大量减少的时间内执行文件(例如 XML™ 文件)的解析,同时从主 CPU 中消除大量的处理负担。对状态表的传统的应用被分为字符调色板、简缩形式的状态表、以及下一状态调色板。调色板可在专用的高速存储器中执行,并且高速缓存装置可被用于加速对简缩的状态表的存取。处理在可以为部分并发的并行流水线中被执行。专用的寄存器可以也被并行地更新,并且由字符调色板所适应的任意长度的特殊字符的串跳过处于标记位的控制下的特征以进一步加速文件的解析。



1. 一种解析器加速器, 包括:

文件存储器;

字符调色板, 包含多个地址, 所述地址对应文件中的字符;

状态表, 包含与所述字符对应的多个条目;

下一状态调色板, 包括状态地址或偏移量; 以及

根据状态表条目存储令牌的令牌缓冲器, 其中

所述状态表中的所述条目包括进入所述下一状态调色板和令牌的地址中的至少一个, 且状态表中的令牌地址存储在令牌缓冲器中。

2. 如权利要求 1 所述的解析器加速器, 其中, 所述字符调色板、所述状态表和所述下一状态调色板形成流水线。

3. 如权利要求 2 所述的解析器加速器, 其中, 所述字符调色板、所述状态表和所述下一状态调色板中的各个分别包括压缩形式的状态表信息的各自部分。

4. 如权利要求 1 所述的解析器加速器, 其中, 所述下一状态调色板包括进入所述状态表中的条目的地址的下一状态地址部分和待被存储的令牌值。

5. 如权利要求 1 所述的解析器加速器, 进一步包括用于在串中检测不会导致状态改变的字符的装置。

6. 如权利要求 5 所述的解析器加速器, 进一步包括无需进行用于状态表存储的进一步存储器操作即可快速处理下一字符的装置。

7. 如权利要求 2 所述的解析器加速器, 其中, 所述流水线以硬件的方式实施。

8. 如权利要求 2 所述的解析器加速器, 其中, 所述流水线构成包括用于将下一状态地址与来自所述字符调色板的状态表索引相结合的装置的环。

9. 一种解析电子文档以识别所关心串的方法, 所述方法包括以下步骤:
在字符调色板中存储与字符对应的多个地址, 在状态表中存储多个条目, 以及在下一状态调色板中存储状态地址以形成环状流水线, 以检测所述关心的串的部分;

从所述状态表中获得令牌信息, 并将令牌存储到令牌缓冲器;
并行地执行对所述令牌信息的存储以及对所述关心的串的部分的检测。

10. 如权利要求 9 所述的方法, 包括以下进一步的步骤:
检测所关心的串的序列; 以及
响应所述检测序列的步骤发出特殊令牌以控制处理。

11. 如权利要求 10 所述的方法, 其中, 所述关心的串的序列包括嵌套的串。

12. 如权利要求 10 所述的方法, 其中, 所述关心的串的序列与文件中的文本字或文本短语对应。

13. 如权利要求 10 所述的方法, 其中, 所述进一步的处理执行消息的阻止。

14. 如权利要求 10 所述的方法, 其中, 所述进一步的处理执行基于内容的路由选择。

硬件解析器加速器

说明

发明背景

发明领域

本发明一般涉及对用于控制通用计算机操作的应用程序的解析 (parsing)，尤其涉及对应用程序、网络数据包的文件和/或其它逻辑序列执行解析操作。

现有技术的描述

近几年来，在计算机和将计算机连接到网络中的链路之间的数字通信领域得到了迅速的发展，这与几年前个人计算机的迅速增长在许多方面都是相类似的。这种在远程处理的互连性和可能性方面的增长极大地增加了在上述网络系统中的单个计算机的有效性能和功能。然而，单个计算机和系统的用途以及当计算机被置入服务时它们的用户的喜好和技术状态这些方面的多样性已经造成了个人机器及它们的操作系统的性能和配置产生了基本程度的变化，个人机器及它们的操作系统共同地被称为“平台”，这些平台通常在某些程度上相互不兼容，尤其是在操作系统和程序语言的层面上。

平台特性和对通信能力和远程处理能力以及为了支持它的足够程度的兼容性的同时需求的这种不兼容已经导致了面向对象程序设计的发展（其接纳了这样一种概念，即，通过实体、属性和关系的参考系统将应用程序以及数据集合为一组或多或少通用的模块）和大量体现这种概念的程序设计语言。可扩展标记语言™（XML™）就是这样的语言，它其已经得到广泛的使用，并且能够作为文件而在任意构造和体系结构的网络上传输。

在这样的语言中，某些字符串与某些指令或标识符相对应，包括特殊字符和其它重要的数据（它们集合起来被称作控制字），其允许数据或操作有效地进行自我识别以便它们可以在随后被当作“对象”，这样，相关的数据和命令可被

以不同语言翻译成不同应用程序的适当格式和命令，从而产生各个相连的平台的支持在给定机器处的期望的处理的一定程度的兼容性。这些字符串的检测通过称为解析（parsing）的操作来实现，这与将表达式（例如句子）的语法分解为其组成部分以及对其进行语法描述的更常规的应用类似。

当解析 XMLTM 文件时，大部分和可能主要的中央处理器（CPU）执行时间被用来遍历该文件搜索控制字、特殊字符和其它被定义作为正在被处理的具体 XMLTM 的规范的重要数据。此举典型地通过软件来完成，该软件查询每个字符并确定它是否属于预定义的感兴趣字符串的集合，例如，一组包括“<command>”、“<data type = dataword>”、“</command>”等的字符串。如果目标字符串的任意一个被检测到，则令牌被保存，该令牌具有在文件中指向用于令牌的起点和长度的位置的指针。这些令牌被积累直到全部文件被解析。

传统的实现方式是在软件中执行基于表格（table）的有限状态机（FSM）来搜寻所要关心的那些字符串。状态表驻留于存储器中并被设计以在文件中搜寻特殊的模式。当前状态被用作状态表中的基址，输入字符的 ASCII 表示（representation）是该表的索引。例如，假定上述状态机处于状态 0（零）并且第一输入字符的 ASCII 值为 02，则状态条目的绝对地址将是基址（状态 0）和索引/ASCII 字符（02）的和/级联（concatenation）。FSM 从 CPU 从存储器中取出输入文件的第一个字符开始。接着 CPU 在存储器中的状态表中构造相应于初始化的/当前状态和输入字符的绝对地址，接着从该状态表中获取状态数据。基于被返回的状态数据，如果不同（表示该字符相应于所感兴趣的字符串的第一字符），CPU 更新当前状态到新的值，并执行在状态数据中指示的其它操作（例如，如果基于前述的进一步重复，单个字符是特殊字符或如果当前的字符被发现是所要关心的串的最后字符，则发出令牌或中断）。

随着所要关心的串连续字符被找到，上述的处理被重复，状态被改变。也就是，如果初始的字符是所关心的串的初始字符，则上述 FSM 的状态能够被提升为新的状态（例如，从初始状态 0 到状态 1）。如果该字符不是所要关心的，则通过在从状态表地址返回的状态表条目中指定同样的状态（例如，状态 0）或不命令状态更新）而将（通常）状态机保持不变。可能的操

作包括但不限于设置中断、存储令牌和更新指针。对后来的字符随后将重复执行上述处理。应该注意到，在所关心的串正被跟随以及上述 FSM 处于非状态 0 的状态(或其它表示所要关心的串还没有发现为当前正在跟随的字符的其它状态)时，与当前串不一致但是所要关心的其它串的初始字符的字符可被发现。在这样一种情况下，状态表条目将指示适当的操作以指示和识别先前被跟随的串片段或部分，并跟随所要关心的可能的新串直到该新串被彻底识别或被发现为不是所要关心的串为止。换句话说，所关心的串可以是嵌套的，状态机必须能够在所要关心的另一个串内检测出所要关心的串，等等。这就要求 CPU 遍历 XMLTM 文件很多次以彻底解析该 XMLTM 文件。

全部的 XMLTM 文件或其它语言文件通过上述方式被一个字符接一个字符地解析。随着潜在的目标串被认出，上述 FSM 一个字符接一个字符地逐步通过各种状态，直到所关心的串被完全识别出或遇到与所关心的可能的串不一致的字符(例如，在上述串被完全/完整地匹配时，或字符偏离目标串时)为止。在后一种的情况下，除了返回到初始的状态或与其它目标串的初始字符的检测相应的状态外，通常不会发生操作。在前一种情况下，令牌与输入文件中的开始地址以及令牌的长度一起被存储到存储器中。在解析完成时，所有的对象将会被识别出，并且基于本地平台或给定平台的处理可以开始。

由于上述搜寻通常是针对所关心的多个串来执行的，所以状态表能够提供从任意给定状态的多种转换。这种实现方式在便利地适应嵌套串的同时，允许当前的字符被解析以同时用于多个目标串。

从前述说明可以看出，对例如 XMLTM 文件的文件解析需要许多重复和用于各次重复的许多存储器存取。因此，在通用 CPU 中的处理时间是最基本必需的。而且，处理多个串的另一个主要的复杂性在于，它需要产生大的状态表，并且是在脱离实时数据包处理的情况下处理。然而，这就需要大量的 CPU 周期获取输入的字符数据、获取状态数据、更新用于文件中各个字符的各种指针和状态地址。这样，解析例如 XMLTM 文件的文件以完全抢先(pre-empt) CPU 或平台中的其它处理，并基本地延迟所需的处理是相对普通(common)的。

在本领域中应该认识到，通过程序编制能够使得通用的硬件模仿

(emulate) 专用硬件的功能，专用的数据处理硬件通常比被程序编制的通用硬件运行更快，即便是它们的结构和程序精确地相互一致，这是因为需要更少的开销来管理和控制专用硬件。然而，专用硬件的需要用于特定处理的硬件资源可以大的惊人 (prohibitively)，尤其是在处理速度的增长可能较小的情况下。而且，专用硬件必需具有功能限制，并为特定的应用提供足够的灵活性，例如，提供搜寻字符的任意数目的任意组合的能力也可以是有限制的。这样，为了切实可行，在提供非常充分的硬件节约的同时专用硬件在处理速度中必须提供大的增长；在所需的处理功能中，由于增长的功能灵活性或可程序编制性而愈加难以同步适应 (accommodate) 的需求是需要的。

在这点上，互相连接能力和需要用于解析例如 XMLTM 文件的文件处理时间量还引出了系统安全问题。一方面，以相对高的优先级进行的、需要非常多的处理时间的任何处理在某些方面相似于在系统或其节点中的拒绝服务 (DOS) 攻击，或能够是在这样一种攻击中使用的工具。

为了有敌意地消耗并最终使可用资源过载，频繁 DOS 攻击表现为琐碎或不良的系统服务请求。硬件加速器的适当配置能够极大地减少或消除可用资源过载的潜在性。此外，在过载时，系统经常发生故障或暴露出脆弱的安全性。这样，消除过载是重要的安全性所要考虑的事情。

而且，由于状态表必须能够在基础层面上 (basic levels) 包含 CPU 命令，因此，完成解析之前，开始一些处理和执行一些命令是可能的，这在没有对系统的性能进行严格折中 (compromise) 的情况下是困难或不可能的。简而言之，用于折中安全的潜在性将通过减少用于处理例如 XMLTM 解析的处理时间而必要地减少。

发明内容

本发明提供了一种用于在限制所需硬件和存储器的数量的同时，加速对例如 XMLTM 文件的文件进行解析处理的专用处理器以及相关硬件。

为了完成本发明的这些目的和其它目的，提供了一种硬件解析器加速器，包括：文件存储器；字符调色板，包含与所述文件中的字符对应的地址；状态表，包含与所述字符对应的多个条目；下一状态调色板，包括状态地址或偏移量；以及令牌缓冲器，其中所述状态表中的所述条目包括进入所述下

一状态调色板和令牌的地址中的至少一个。

附图的简要说明

前述和其它目标,方面和有益效果在参照附图对本发明的优选实施方式进行详细的描述后将会被更好地理解,其中:

图 1 显示了在文件解析中使用的状态表的一部分;

图 2 是根据本发明的解析器加速器的高层次示意图;

图 3 描述了如图 2 所描述的优选的字符调色板 (palette) 的格式;

图 4A 和 4B 描述了如图 2 描述的本发明优选形式所述的状态表格式和与状态表一起使用的状态表控制寄存器;

图 5 描述了如图 2 所示的优选的下一状态调色板格式; 以及

图 6 是如图 5 所描述的优选令牌格式。

本发明优选实施方案的详细描述

现在参考附图,更具体地是参照图 1,其中显示了对理解本发明有用的状态表的部分表示。应该理解在图 1 中显示的状态表是可用于解析 XMLTM 文件的非常小的部分,并在本质上是用于示例性的目的。应该理解,XMLTM 文件在这里被用作能够使用根据本发明的加速器来处理的一种类型的逻辑数据序列的实施例。其它的逻辑数据序列也能够从网络数据包内容构建,例如用于共享服务器计算机执行的用户终端命令串。尽管全部状态表没有物理地存在(至少在本发明显示的表格中),图 1 还是能够被用来便于对公知软件解析器的操作的理解,申请人声明图 1 中没有属于任何与本发明相关的现有技术。

观察到在图 1 中描述的状态表部分的许多条目是复制性的也是有帮助的,对于本发明的理解而言,不再需要使硬件适应于图 1 中表示出的整个状态表是重要的。相反,尽管本发明能够在软件中实现(可能使用专用的处理器),但是根据本发明的硬件需求被充分地限制,用于通过软件来解析的增加处理时间的性能损失 (penalty) 不会通过任意可能的硬件节约 (hardware economy) 来评判 (justify)。

在图 1 中,状态表被分割成任意数目的行,每一行具有与一种状态相应的基址。基址的行被分割成与被用来表示待被解析的文件中的字符的编码的

个数一致的列；在该实施例中，有二百五十六（256）列与用于作为状态表的索引的字符的基础 8 位字节相对应。

注意到显示的状态表条目的几个方面是有帮助的，尤其在传递在图 1 中描述的示例性的状态表的一小部分是如何支持许多字的检测的理解方面：

1. 在显示的状态表中，在用于状态 0 的行中只有两个条目包括不是“位于状态 0”的条目，在被测试的字节与所关心的任意字符串的初始字符不匹配时，该条目保持了初始的状态。提供前进到状态 1 的单个条目对应于这样一种特殊情况下，其中全部所关心的串都以相同的字符开始。可以提供前进到其它状态的任意其它字符将通常但不是必须地前进到除状态 1 以外的状态，但是，对通过其它字符能够被达到的相同状态的进一步参考可被用来例如检测嵌套的串。在{状态 0, FD}处示出的命令（例如，“特殊的中断”）与“位于状态 0”的结合可被用来检测和操作特殊的单个字符。

2. 在高于状态 0 的状态中，“保持于状态 n”的条目是为待要通过一个或多个字符的可能的长时间运行而得到保持的状态而规定的，像通常遇到的情况那样，这种状态可能在例如命令的数值增加中遇到。本发明提供了这种类型的字符串的特殊处理以提供增强的加速，如在下面将详细描述的那样。

3. 在高于状态 0 的状态，“到达状态 0”表示检测到了从所关心的任意串中区分该串的字符，而不管有多少匹配的字符在先前被检测到，并将解析处理返回到初始/缺省状态以开始搜寻所关心的其它串。（因此，到目前为止，“到达状态 0”条目通常是状态表中出现最频繁和最多的条目）。返回到状态 0 可要求解析操作返回到文件中的在区分字符被检测到时、开始被跟随的字符的后面的字符。

4. 包括“到达状态 0 的命令的条目表示完成了对所关心的完整串的检测。通常，该命令将会是存储一个令牌（带有该令牌的地址和长度），之后，该令牌使得串被作为一个对象来处理。然而，具有“到达状态 n”的命令规定在继续跟随能够潜在地与所关心的串匹配的串的同时，在中间点（intermediate point）发起操作。

5. 为了避免在其上搜寻在所关心的两个串（例如，具有 n-1 个相同的初始字符但第 n 个字符不同的串，或具有不同的初始字符的串）之间出现分枝的任意点处出现不确定性（ambiguity），通常需要向不同的（例如，非连

续性的) 状态行进, 如在{状态 1, 01}和{状态 1, FD}处所示的状态. 除了被包括的特殊字符的串和所关心的具有共同初始字符的串的特殊情况外, 对任意长度为 n 的串的完整识别需要 $n-1$ 个状态. 为此, 状态和状态表行的数目通常必须非常大, 即便是对于所关心的相对适度数目的串也如此.

7. 与前段内容相反, 大部分的状态能够通过一个或两个唯一的条目和缺省的“到达状态 0”而得到完全表征. 图 1 的状态表的这些特性在本发明中被用来产生高程度的硬件节约性并大幅度地加速解析处理以用于所关心的串的通常情况.

如上所暗示的一样, 解析处理(, 如传统执行一样) 开始于在给定的缺省/初始状态下的系统, 在图 1 中被描述为状态 0, 接着, 在重复处理后随着匹配字符被发现而前进到较高序号的状态. 在所关心的串被完全识别时, 或者在具有潜在匹配性的串的中间位置处指定一个特定的操作时, 例如存储令牌或发出中断的操作被执行. 然而, 在用于文件的各个字符的每次重复中, 字符必须从 CPU 存储器获取, 状态表必须被(再次从 CPU 存储器) 获取, 并且各种指针(例如, 指向文件的字符和状态表中的基址) 和寄存器(例如, 指向串的初始匹配字符地址和累加的长度) 必须在顺序的操作中被更新. 因此, 很容易认识到上述解析处理耗费大量的处理时间.

根据本发明的解析器加速器 100 的高层次示例性的方框图如图 2 所示. 如本领域的普通技术人员可以认识到的那样, 图 2 还可以理解为根据本发明执行解析的执行步骤的流程图. 如下面将连同图 3、4A、4B、5 和 6 进行详细论述的那样, 本发明在表示状态表时使用了一些硬件节约措施, 以使得多个硬件流水线(pipelines) 被开发出来, 这些硬件流水线尽管在时间上有轻微的不对称(skewed) 但基本上以并行的方式操作. 这样, 指针和寄存器的更新能够被基本上并行地和与其它操作并发地执行, 同时通过较快地访问以并行方式操作的硬件以及从关于状态表和文件的 CPU 存储器中预取(prefetching), 存取存储器所需的时间被大量降低.

作为一般的总体观点, 例如 XMLTM 文件的文件被外部地存储在由寄存器 112、114 编索引的 DRAM 120 中, 并优选地通过 32 位字传输到为上述流水线起到多路复用器的作用的输入缓冲区 130. 各个流水线包括字符调色板(palette) 140、状态表 160 和下一状态调色板 170 的拷贝; 其每一个都

适应于压缩形式的状态表的部分。下一状态调色板 170 的输出包含进入状态表 160 中的条目的地址的下一状态地址部分和待被存储的令牌值(如果有的话)。字符调色板 140 和下一状态调色板 170 中的操作是对高速内部 SRAM 简单的存储器访问,内部 SRAM 可以相互并行地运行,并且与对形成状态表 160 的外部高速 DRAM(其还可实现为高速缓冲器)的简单存储器访问并行。因此,只需要 CPU 初始地控制这些硬件元件(但是,其一旦启动,仅能够用偶然的 CPU 操作调用来自发地起作用以更新文件数据和存储令牌)的相对少的时钟周期以用于文件中的各个字符的评估。基础的加速增益是在 CPU 中各个字符的所有操作持续时间加上在高速 SRAM 或 DRAM 中单一自发地执行的存储器操作的持续时间的总和的减少。

应该理解,在这里所称为“外部的”存储器构造是用来暗示存储器 120、140 的配置,考虑到所需的存储的量和从上述硬件解析器加速器和/或主 CPU 的存取,其目前是本发明人所优选的。换句话说,令牌处理和一些其它操作提供根据本发明的解析器加速器的体系结构以便利存储器共享,或至少便利由主机 CPU 以及硬件加速器存取是具有有益效果的。根据这些论述,本领域的普通技术人员应该认识到没有其它的预期内含和更广范围的例如同步 DRAM (SDRAM) 的硬件替代物是适合的。

现在参照图 3 到图 6 以支持图 2 中优选的实现的硬件节约措施为例对字符调色板 140、状态表 160、下一状态表 170 和下一状态和令牌的格式进行讨论。可以使用其它的技术/格式,同样,上述描述的格式可以理解为示例性的但目前是优选的。

图 3 描述了字符调色板优选形式,该调色板与被包括在或可以被包括在所关心的串中的字符对应。相对应于图 1 中的状态表中列的数目,这些格式优选地提供了编号为 0-255 的条目。(术语“调色板 (palette)”与包含用于支持各颜色的数据并且被集合性地称为全色域 (gamut) 的术语“调色板 (color palette)”以相同的意思使用,调色板的使用减少了状态表中的条目/列)。例如,被称为“空字符”的不会导致任何状态变化的字符能够在状态表中的单列中表述,而不是在许多列中表述。在 144 处为空字符输出进行测试是所期望的,这能够基本地加速用于解析的处理,这是因为它允许对下一字符立刻进行处理,而无需对状态表存取的进一步的存储器操作。这

种格式能够被单个寄存器所适应，或者被由（例如）在指向特殊字符的调色板（被图 2 中用重叠的存储板示意地描述）的基址寄存器 142 中的数据进行配置的存储单元所适应。来自文件（例如 XMLTM 文件）的当前 8 位字符（四个中的一个从输入缓冲器 130 提供，和从外部 DRAM 120 接收的四字节字一样）寻址调色板中的条目，该调色板接着输出地址作为状态存储器的索引或局部指针。这样，通过上述格式提供调色板，图 1 的功能的一部分能够以相对被限制容量的单个寄存器的形式提供；这样，允许它们中的多个被形成并以并行的方式操作，同时保持充分的硬件节约并支持状态表 160 中的其它功能。

图 4A 显示了优选的状态表格式，其以与字符调色板（例如，基本上和寄存器一样）相似的方式构成或配置。与图 3 中的字符调色板的主要不同在于，寄存器的长度依赖于对所期望字符的响应次数以及所关心字符串的个数和长度。因此，如果能够被经济地提供的内部存储器的数量在具体的情况中是不充足的，则提供在 CPU 或其它外部 DRAM（可能具有内部或外部的高速缓冲）中实现这种存储器的可能性是考虑期望得到的。尽管如此，由于在图 1 的状态表中的具有较高复制性的条目能够被减少到单个条目，所以提供充分的硬件节约是很清楚的；单个条目的地址通过如根据图 3 的字符调色板如上所述而提供的数据而被接纳。状态表 160 的输出优选地是一位、两位或四位，但是规定多达 32 位的位数可以提供增加的灵活性，如在下面参照图 4B 论述的那样。在任何情况下，状态表的输出提供了访问下一状态调色板 170 的地址或指针。

现在参照图 4B，如在后面关注的本发明的完美特征一样，本发明优选的实现特征包括状态表控制寄存器 162，其允许更进一步的充分硬件节约，尤其是在如果状态表 160 的 32 位输出将被提供的情况下。实质上，状态表控制寄存器通过允许变长字被存储到状态表并将其从其中读出而提供状态表信息的压缩。

更具体地说，状态表控制寄存器 162 存储和提供图 4A 的状态表 160 中的各个条目的长度。由于图 1 中的一些状态表条目是高度复制性的（例如，“到达状态 0”、“保持于状态 n”），所以这些条目不仅能够通过状态表 160 中的单个条目或至少远少于图 1 中的条目来表示，而且还可被较少的位

表示,这些较少的位可以少到即便是大部分或所有的复制性条目被包括在状态表时仍能够产生充分的硬件节约,就像在一些状态表中被发现是方便的那样。本领域的普通技术人员应该认识到这种减少的原理相似于所谓的熵(entropy)编码。

现在参照图 5 对下一状态调色板 170 的优选格式进行讨论。下一状态调色板 170 优选地以与上述讨论过的字符调色板 140 非常相同的方式实现。然而,由于具有状态存储器 160,可能需要的条目的个数不会被事先知道,并且各自条目的长度优选地非常长(例如,两个 32 位的字节)。另一方面,由于只有相对小和可预见的地址范围需要包含在任意给定的时间,所以下一状态调色板 170 能够作为高速缓冲来操作(例如,使用下一状态调色板基址寄存器 172)。而且,如果状态表 160 的 32 位输出被提供,一些上述数据能够被用来补充下一状态调色板 170 的条目中的数据,可能在后者中允许较短的条目或可能完全地跳过下一状态调色板,如短划线 175 所示。

如图 5 中所示,从下一状态调色板 170 输出的低地址 32 位字是待被保存的令牌。这些令牌优选地被形成为具有 16 位令牌值、8 位令牌标记和 8 位控制标记,其中,令牌值和令牌标记被存储在令牌缓冲器 190 的由指向串的开头的指针 192 和通过计算成功的字符比较而积累的长度共同提供的地址处。控制标记设置中断到主 CPU 或控制解析器加速器中的处理。上述后者控制标记中的一个被优选地用来设置能够跳过字符的功能,这不会导致在除了状态 0 之外的状态处的状态变化,例如在所关心的串中出现的任意长度的相同或相关字符的串,如上所能推导出的一样。在这样一种情况下,下一状态表条目无需从 SRAM/SDRAM 获取就能够被重新使用。输入缓冲地址 112 无需额外的处理就可被递增;从而允许用于字符的确定串的解析的充分附加的加速。第二个 32 位字是反馈到寄存器 180 和加法器 150 的地址偏移,其待与来自字符调色板的索引输出连接(concatenate)以形成指向用于下一字符的状态表指针。相应于状态 0 的初始地址通过寄存器 182 提供。

这样可以看出,字符调色板的使用、简化形式的状态存储器和下一状态存储器将传统的状态存储器的操作的功能清楚地表达为单独的阶段;各个阶段能够用相对稍高速度的存储器来极快地执行,这样,各阶段可被复制以依次形成与令牌的其它操作和存储并行的对文件各自字符操作的并行流水线。

因此，解析处理能够被极大地加速，即便是相对于在其它的字符处理能够被开始之前必须顺次执行所有上述功能的专用处理器。

总之，该加速器存取主机 CPU 的字符数据（有时被称作暗含网络传输的数据包）和状态表所定位的程序存储器。加速器 100 经由存储器映射寄存器而处于主 CPU 的控制下。该加速器能够中断主 CPU 以指出例外、报警和终止。在解析开始时，指针（112，114）被设置到待被解析的输入缓冲器 130 的数据的开端和结尾，待被使用的状态表（如基址 182 所示和其它控制信息（例如，142）被建立在加速器内。

为了启动加速器的操作，CPU 发出命令到加速器，作为对该命令的响应，加速器从 CPU 程序存储器（例如，120 或高速缓冲）获取第一个 32 位字，并将其置入到输入缓冲器 130，第一字节/ASCII 字符从输入缓冲 130 中选择。加速器获取相应于输入字符（也就是，图 4A 对应图 1 中的完整状态表的单个字符或单个列）的状态信息和当前状态。该状态信息包括下一状态地址和待执行的例如中断 CPU 或终止处理的任一特定操作。这样，状态信息的提前不但支持对所关心的单个串的检测，而且支持对嵌套的串（如上所暗示）以及串的序列或者相应的令牌（例如文件中文本的字或短语）的检测。响应于上述检测发出的中断和/或者异常并不限制于解析器的内部控制和令牌的发出，而是可能产生报警或者另外地启动其它提供其它功能的处理，例如截取不希望的电子邮件或者阻止应该反对的（objectionable）主题或基于内容的路由选择（可能通过特殊令牌的发出来实现）。

加速器接下来从输入缓冲器 130 中选择待被分析的下一字节，并用加法器 150 利用可得到的新状态信息重复上述处理。上述操作或令牌信息的存储能够被并发地执行。这种执行在输入字的所有四个字符被分析之前继续。接着（或与解析预获取的第四个字符并发地），缓冲器 112、114 被比较以确定是否到达了文件缓冲器 120 的末尾，如果达到了文件缓冲器的末尾，则中断被发送回 CPU。否则，获取新的字，缓冲器 112 被更新，上述处理被重复。

由于指针和计数器在专用的硬件中实现，所以它们能够被并行地更新，而不是像如果以软件实现时所需的串行更新。这就将解析数据的字节的时间减少至执行以下动作所需的时间，即，从局部输入缓冲器中获取字符、从高

速局部字符调色板存储器中产生状态表地址、从存储器中获取相应的状态表条目和再次从局部高速缓冲器中取出下一状态信息。上述操作的一些操作能够在单独的并行流水线中并行地执行，在状态表信息（部分地或全部地通过下一状态调色板来提供）中指定的其它操作可在继续更进一步的字符解析的同时被执行。

因此可以很清楚地看出，本发明通过小而经济数量的专用硬件来提供解析处理的充分加速。在解析器加速器能够中断 CPU 时候，上述处理的操作在初始命令后被完全从 CPU 移到解析器加速器。

尽管本发明根据单独的优选实施方式来描述，本领域的普通技术人员将会认识到在本发明的精神和所附的权利要求的范围内，也可对本发明做出修改。

		ASCII 值						
索引 →	基址 ↓	00	01	02	03	FD	FE	FF
	状态0	停留 状态0	停留 状态0	到达 状态1	停留 状态0	停留 状态0	停留 状态0
	状态1	停留 状态1	到达 状态2	到达 状态0	停留 状态1	到达 状态4	到达 状态0
	状态2	停留 状态2	到达 状态3	到达 状态0	到达 状态3, 特殊 中断	到达 状态0	到达 状态0
	状态3	停留 状态3	到达 状态0	到达 状态0, 存储 令牌	到达 状态0	到达 状态0	到达 状态0
	状态4	停留 状态4	到达 状态0, 存储 令牌	到达 状态0	到达 状态0	到达 状态0	到达 状态0
	状态5	停留 状态5	到达 状态0	到达 状态0	到达 状态0	到达 状态0	到达 状态0, 存储 令牌

图 1

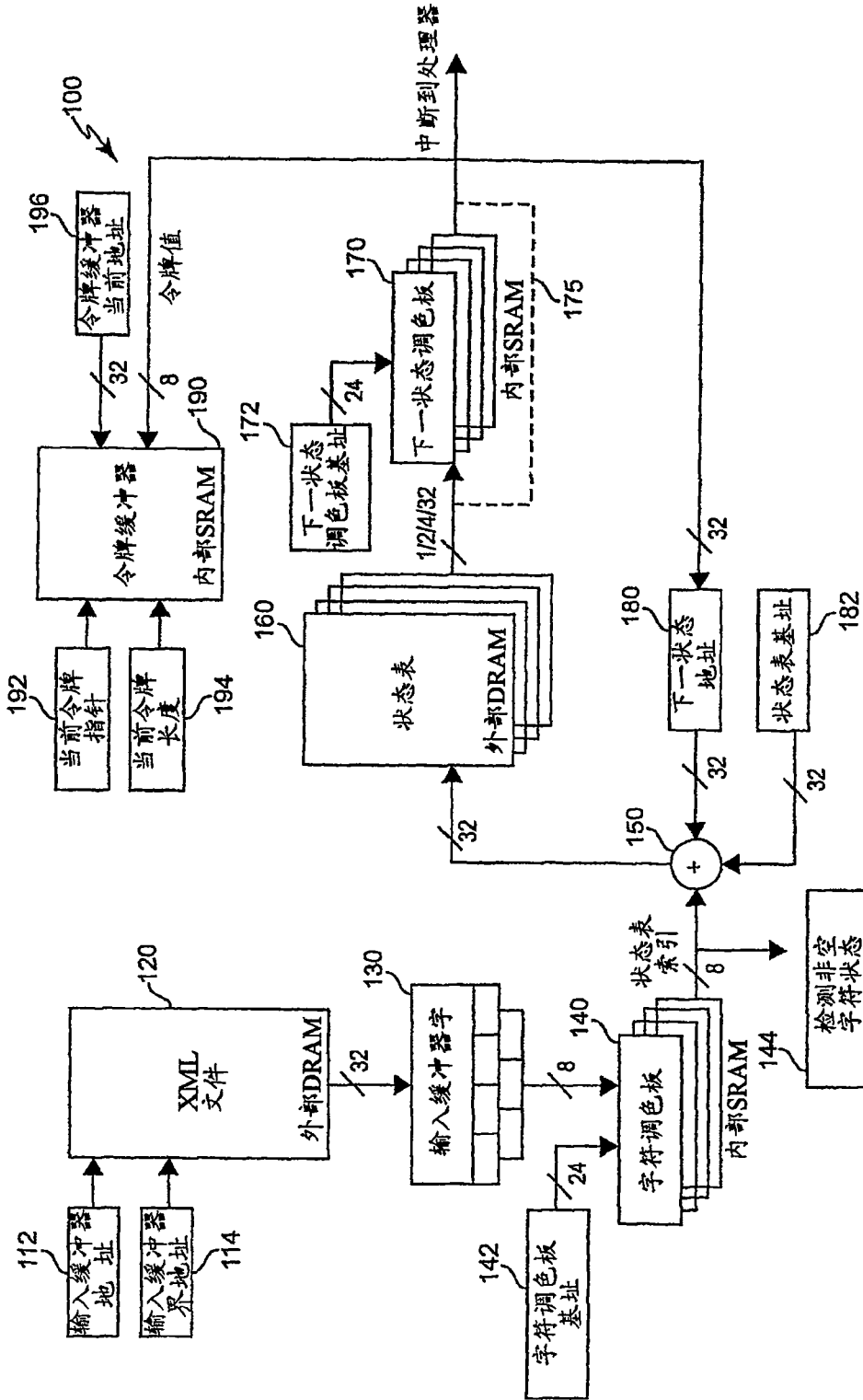


图 2

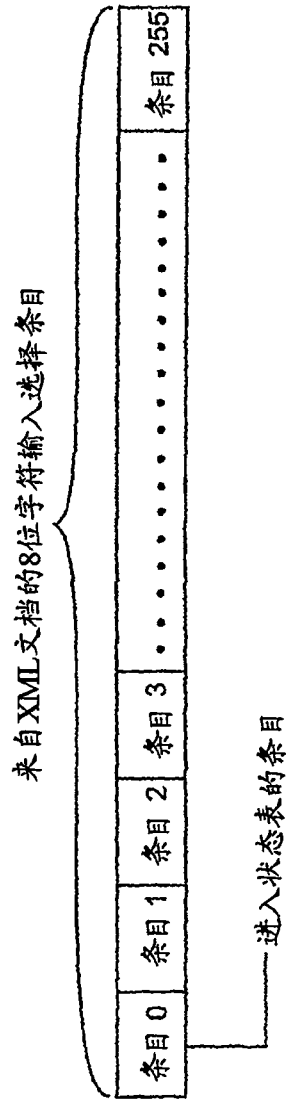


图 3

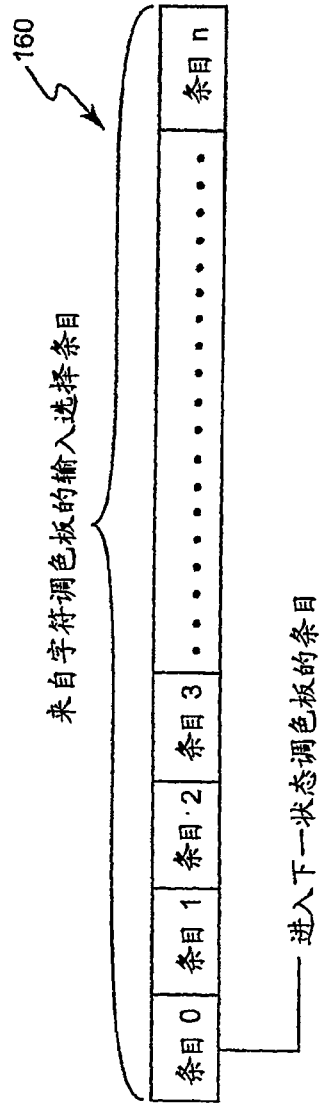


图 4A

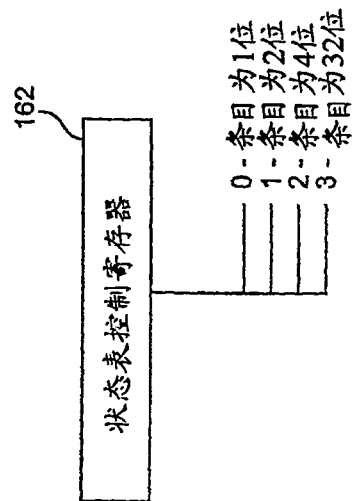


图 4B

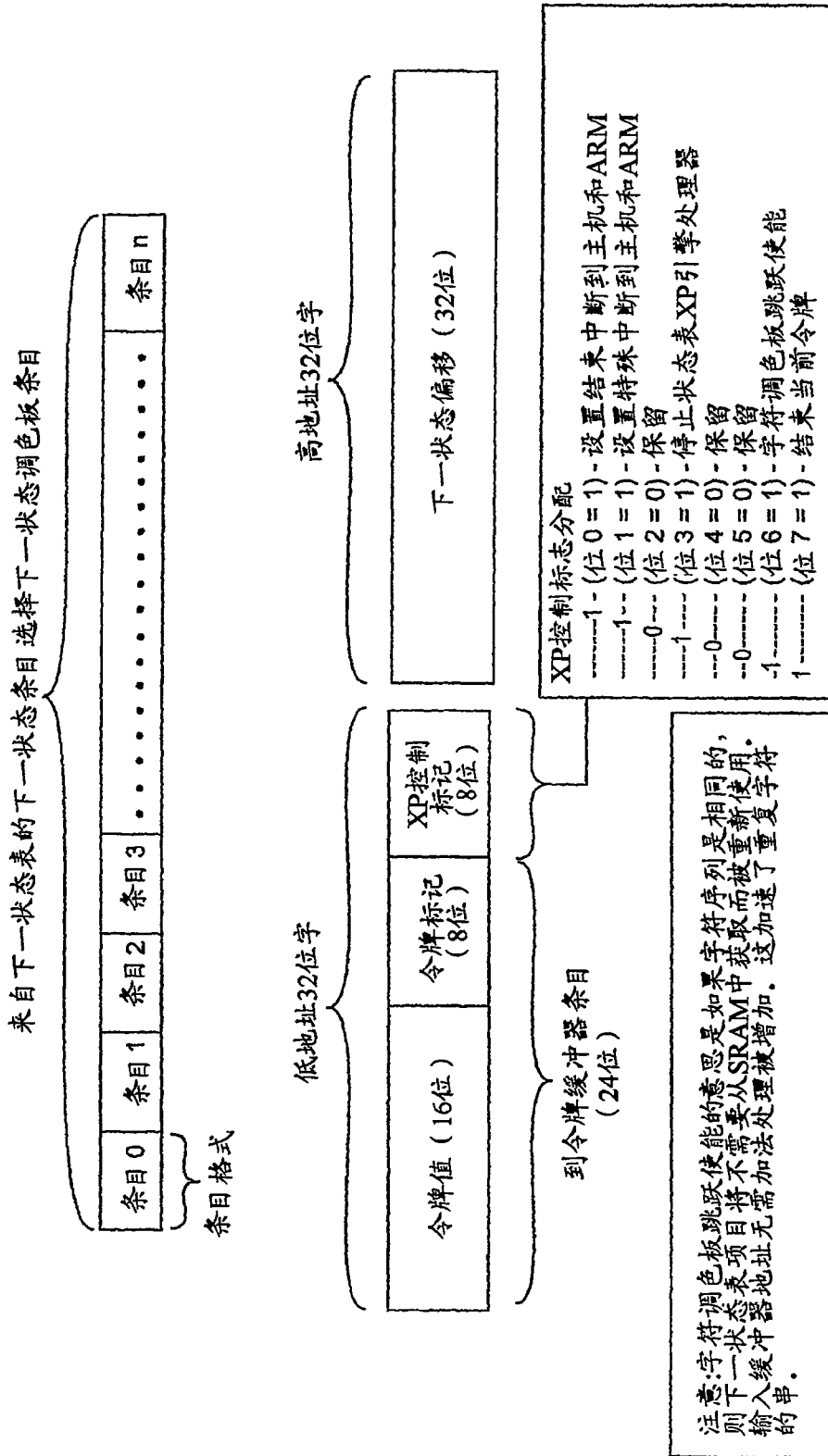


图 5

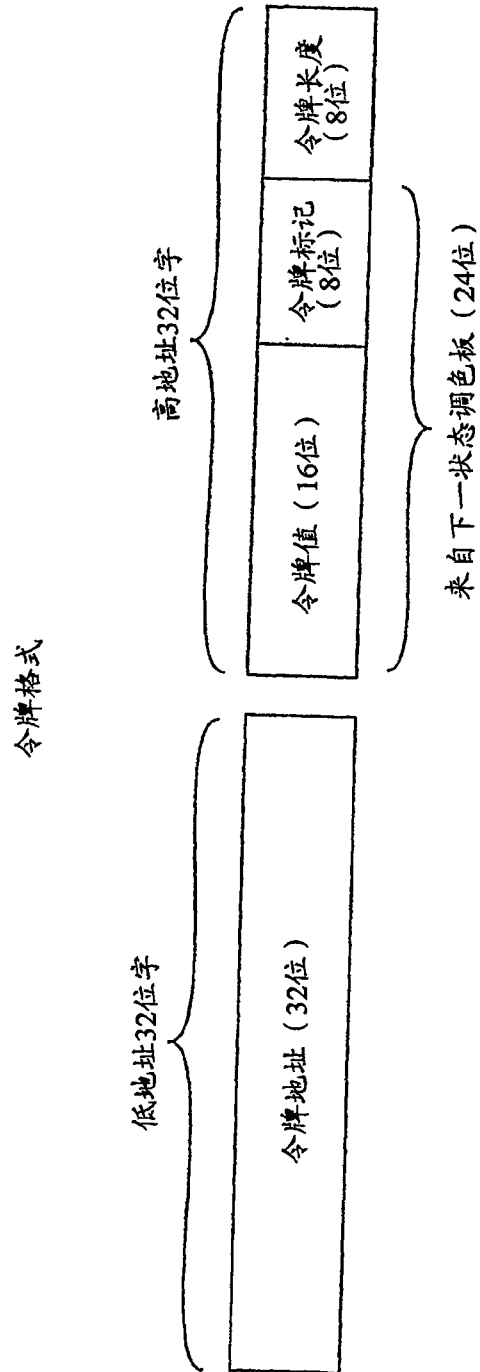


图 6