



(12) 发明专利

(10) 授权公告号 CN 113454593 B

(45) 授权公告日 2025. 06. 27

(21) 申请号 202080015707.3

(22) 申请日 2020.02.20

(65) 同一申请的已公布的文献号
申请公布号 CN 113454593 A

(43) 申请公布日 2021.09.28

(30) 优先权数据
16/286,990 2019.02.27 US

(85) PCT国际申请进入国家阶段日
2021.08.20

(86) PCT国际申请的申请数据
PCT/EP2020/054518 2020.02.20

(87) PCT国际申请的公布数据
W02020/173808 EN 2020.09.03

(73) 专利权人 国际商业机器公司
地址 美国纽约

(72) 发明人 M·克莱纳 B·C·贾梅伊
A·T·索菲亚 M·法雷尔
S·斯瓦尼 T·什莱格尔

(74) 专利代理机构 北京市中咨律师事务所
11247
专利代理师 于静 刘薇

(51) Int.Cl.
G06F 9/455 (2006.01)

(56) 对比文件
US 2011320825 A1,2011.12.29
US 8701109 B1,2014.04.15

审查员 池梦群

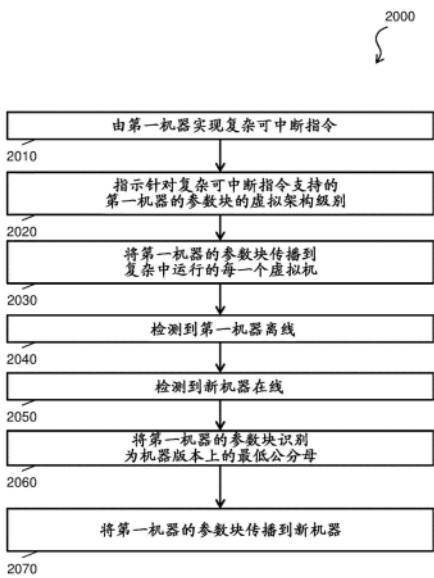
权利要求书2页 说明书48页 附图28页

(54) 发明名称

保持多个机器生成上的复杂功能的兼容性

(57) 摘要

包括多个机器的系统包括第一生成机器和第二生成机器。多个机器中的每一个包括机器版本。第一生成机器执行第一虚拟机和虚拟架构级。第二生成机器执行第二虚拟机和虚拟架构级。虚拟架构级别为第一虚拟机和第二虚拟机的复杂可中断指令提供兼容性级别。针对跨多个机器的最低公分母机器版本构建兼容性等级。兼容性等级包括标识最低公分母机器版本的最低公分母指示符。



1. 一种用于保持多个机器生成上的复杂功能的兼容性的系统,包括:

多个机器,包括第一生成机器、第二生成机器和第三生成机器,所述多个机器中的每一个包括机器版本,所述第一生成机器、所述第二生成机器和所述第三生成机器各自包含不同的机器版本,所述第一生成机器执行第一虚拟机和虚拟架构级别,所述第二生成机器执行第二虚拟机和所述虚拟架构级别,所述第三生成机器执行第三虚拟机和所述虚拟架构级别,其中,

所述虚拟架构级别为第一虚拟机、第二虚拟机和第三虚拟机的复杂可中断指令提供兼容性级别;

所述兼容性级别是针对所述多个机器中的最低公分母机器版本构建的;

所述兼容性级别包括标识所述最低公分母机器版本的最低公分母指示符;

所述复杂可中断指令具有需要保存下来的状态并符合参数块格式;

所述兼容性级别包括与最低公分母指示符相关联的参数块格式;

所述兼容性级别包括用于所述多个机器中的每一个的所述复杂可中断指令的本地参数块格式,所述本地参数块格式是针对所述多个机器中的每一个本地的所述机器版本而架构化的;

所述最低公分母指示符由所述多个机器中的机器生成以首先执行所述复杂可中断指令;

所述最低公分母指示符从所述多个机器中的机器传播以首先执行所述复杂可中断指令至所述多个机器的剩余数目。

2. 根据权利要求1所述的系统,其中,所述兼容性级别内的所述最低公分母指示符通过标识哪些函数在特定虚拟机中可用的一系列设施位来控制。

3. 根据权利要求1所述的系统,其中所述复杂可中断指令包括DEFLATE转换调用指令。

4. 根据权利要求1所述的系统,其中,所述复杂可中断指令包括来自在加速器上运行的复杂指令集的指令。

5. 一种用于保持多个机器生成上的复杂功能的兼容性的方法,包括:

由多个机器中的第一生成机器执行第一虚拟机和虚拟架构级别,所述多个机器包括所述第一生成机器、第二生成机器和第三生成机器,所述多个机器中的每一个包括不同的机器版本;

由所述第二生成机执行第二虚拟机和所述虚拟架构级别;

由所述第三生成机器执行第三虚拟机和所述虚拟架构级别;并且

由所述虚拟架构级别将复杂可中断指令的兼容性级别提供给所述第一、第二和第三虚拟机,其中,

所述兼容性级别是针对所述多个机器中的最低公分母机器版本而架构的;

所述兼容性级别包括标识所述最低公分母机器版本的最低公分母指示符;

所述复杂可中断指令具有需要保存下来的状态并符合参数块格式;

所述兼容性级别包括与最低公分母指示符相关联的参数块格式;

所述兼容性级别包括用于所述多个机器中的每一个的所述复杂可中断指令的本地参数块格式,所述本地参数块格式是针对所述多个机器中的每一个本地的所述机器版本而架构化的;

所述最低公分母指示符由所述多个机器中的机器生成以首先执行所述复杂可中断指令；

所述最低公分母指示符从所述多个机器中的机器传播以首先执行所述复杂可中断指令到所述多个机器的剩余数目。

6. 根据权利要求5所述的方法, 其中, 所述兼容性级别内的所述最低公分母指示符通过识别哪些函数在特定虚拟机中可用的一系列设施位来控制。

7. 根据权利要求5所述的方法, 其中, 所述复杂可中断指令包括DEFLATE转换调用指令。

8. 根据权利要求5所述的方法, 其中, 所述复杂可中断指令包括来自在加速器上运行的复杂指令集的指令。

9. 一种计算机程序产品, 所述计算机程序产品包括程序指令, 所述程序指令可执行以致使包括以下各项的操作:

由多个机器中的第一生成机器执行第一虚拟机和虚拟架构级别, 所述多个机器包括所述第一生成机器、第二生成机器和第三生成机器, 所述多个机器中的每一个包括不同的机器版本;

由所述第二生成机器执行第二虚拟机和所述虚拟架构级别;

由所述第三生成机器执行第三虚拟机和所述虚拟架构级别; 并且

由所述虚拟架构级别将复杂可中断指令的兼容性级别提供给所述第一、第二和第三虚拟机, 其中,

所述兼容性级别是针对所述多个机器中的最低公分母机器版本而架构的;

所述兼容性级别包括标识所述最低公分母机器版本的最低公分母指示符;

所述复杂可中断指令具有需要保存下来的状态并符合参数块格式;

所述兼容性级别包括与最低公分母指示符相关联的参数块格式;

所述兼容性级别包括用于所述多个机器中的每一个的所述复杂可中断指令的本地参数块格式, 所述本地参数块格式是针对所述多个机器中的每一个本地的所述机器版本而构造的;

所述最低公分母指示符由所述多个机器中的机器生成以首先执行所述复杂可中断指令;

所述最低公分母指示符从所述多个机器中的机器传播以首先执行所述复杂可中断指令到所述多个机器的剩余数目。

10. 根据权利要求9所述的计算机程序产品, 其中, 所述兼容性级别内的所述最低公分母指示符通过识别哪些函数在特定虚拟机中可用的一系列设施位来控制。

11. 根据权利要求9的计算机程序产品, 其中, 所述复杂可中断指令包括DEFLATE转换调用指令。

12. 根据权利要求9的计算机程序产品, 其中, 所述复杂可中断指令包括来自在加速器上运行的复杂指令集的指令。

保持多个机器生成上的复杂功能的兼容性

背景技术

[0001] 本发明总体上涉及维持多个机器生成 (generation) 上的复杂功能的兼容性。

[0002] 关于允许工作负荷在不同机器生成的机器之间迁移,每个机器需要能够继续由另一机器启动的工作。例如,当在系统可以在不同计算机之间迁移的环境中运行时,需要在这些环境之间存在共同的架构级别。对于具有可在架构之间改变的复杂参数块格式的指令,机器需要能够使用这些环境之间的公共架构级别来创建和消耗每一先前产生的参数块格式。然而,创建和消耗每一先前产生的参数块格式可引起实施和验证复杂性。例如,对于可能在五生成中看到不同参数块定义的复杂指令,最新一生成机器需要理解并产生五个不同参数块。

发明内容

[0003] 根据本发明的实施例,提供了一种系统,该系统包括多个机器。所述多个机器包括第一生成机器和第二生成机器。多个机器中的每一个包括机器版本。第一生成机器执行第一虚拟机和虚拟架构级。第二生成机器执行第二虚拟机和虚拟架构级。虚拟架构级别为第一虚拟机和第二虚拟机的复杂可中断指令提供兼容性级别。针对跨所述多个机器的最低公分母 (lowest common denominator) 机器版本构造兼容性等级。兼容性等级包括标识最低公分母机器版本的最低公分母指示符。本文中的实施例的技术效果和益处可包括当针对复杂指令/函数,每个机器生成最多仅支持两个不同的参数块 (其自身的参数块和兼容性级别) 时,验证和确认被减少到仅针对参数块的当前和兼容性版本进行测试。

[0004] 根据本发明的一个或多个实施例或上述系统实施例,兼容性级别包括用于多个机器中的每一个的复杂可中断指令的本地参数块格式,所述本地参数块格式是针对多个机器中的每一个本地的机器版本构建的。

[0005] 根据本发明的一个或多个实施例或上述系统实施例中的任一个,最低公分母指示符由多个机器中的机器生成以首先执行复杂可中断指令。

[0006] 根据本发明的一个或多个实施例或上述系统实施例中的任一个,最低公分母指示符从多个机器中的机器传播以首先执行复杂可中断指令至多个机器的剩余数量。本文中的实施例的技术效果和益处可以包括:对于第一生成,其参数块格式表示“起源”格式并且仅需要支持/测试一种格式。由此,任何后续生成的机器特定参数块内容都可以针对该机器进行优化,而不会对未来的机器生成产生下游影响。

[0007] 根据本发明的一个或多个实施例或上述系统实施例中的任一个,兼容性等级内的最低公分母指示符通过一系列设施位 (facility bits) 来控制,所述设施位标识哪些功能在特定虚拟机中可用。

[0008] 根据本发明的一个或多个实施例或上述系统实施例中的任一个,复杂可中断指令包括DEFLATE转换调用指令。

[0009] 根据本发明的一个或多个实施例或上述系统实施例中的任一个,复杂可中断指令包括来自在加速器上运行的复杂指令集的指令。

[0010] 根据本发明的一个或多个实施例,任何上述系统实施例可以被实现为方法或计算机程序产品。

[0011] 根据一个或多个实施例,提供了一种由第一机器实现复杂可中断指令并在虚拟架构级别中指示针对所述复杂可中断指令支持所述第一机器的参数块的方法。而且,第一机器的参数块被传播到在复合体中运行的每个虚拟机。本文中的实施例的技术效果和益处可包括减少针对参数块的当前和兼容性版本进行测试的验证和验证。

[0012] 根据本发明的一个或多个实施例或上述方法实施例,该方法可以包括:检测新机器在线;将第一机器的参数块标识为跨越复合体的机器版本的最低公分母;以及将第一机器的参数块传播到新机器

[0013] 根据本发明的一个或多个实施例,以上方法实施例中的任一个可以被实现为系统或计算机程序产品。

[0014] 通过本发明的技术实现了附加的技术特征和益处。在此详细描述了本发明的实施例和方面,并且这些实施例和方面被认为是所要求保护的主体的一部分。为了更好地理解,参考具体实施方式和附图。

附图说明

[0015] 在说明书结论的权利要求书中特别指出并清楚地要求保护本文描述的独占权利的细节。从以下结合附图的详细描述中,本发明的实施例的前述和其他特征和优点是显而易见的,其中:

[0016] 图1A描绘了结合和使用本发明的一个或多个方面的计算环境的一个实例;

[0017] 图1B描绘了根据本发明的一个或多个方面的图1A的进一步细节;

[0018] 图2描绘并入且使用本发明的一个或多个方面的计算环境的另一实例;

[0019] 图3A描绘了根据本发明的一个方面的DEFLATE转换调用(DFLTCC)指令的一种格式;

[0020] 图3B描绘了根据本发明的一个方面的由DEFLATE转换调用指令使用的隐式寄存器(通用寄存器0)的字段的一个示例;

[0021] 图3C描绘根据本发明的一个方面的用于DEFLATE转换调用指令的函数代码的一个实例;

[0022] 图3D描绘根据本发明的一个方面的由DEFLATE转换调用指令使用的隐式寄存器(通用寄存器1)的字段的一个实例;

[0023] 图3E描述了根据本发明的一个方面的由DEFLATE转换调用指令指定的寄存器R1的内容的一个实例;

[0024] 图3F描绘了根据本发明的一个方面的由DEFLATE转换调用指令使用的寄存器R1+1的内容的一个实例;

[0025] 图3G描绘根据本发明的一个方面的由DEFLATE转换调用指令指定的寄存器R2的内容的一个实例;

[0026] 图3H描绘了根据本发明的一个方面的由DEFLATE转换调用指令使用的寄存器R2+1的内容的一个实例;

[0027] 图3I描述了根据本发明的一个方面的由DEFLATE转换调用指令指定的寄存器R3的

内容的一个实例；

[0028] 图3J描述了根据本发明的一个方面的由DEFLATE转换调用指令的DFLTCC-QAF (查询可用函数) 函数使用的参数块的内容的一个实例；

[0029] 图3K描述了根据本发明的一个方面的由DEFLATE转换调用指令的DFLTCC-GDHT (生成动态霍夫曼表) 函数使用的参数块的内容的一个实例；

[0030] 图3L描述了根据本发明的一个方面的由DEFLATE转换调用指令的DFLTCC-CMPR (压缩) 和DFLTCC-XPND (扩展) 函数使用的参数块的内容的一个实例；

[0031] 图4描绘了根据本发明的一个或多个方面的子字节边界的一个实例；

[0032] 图5A-5C描述了根据本发明的一方面的示出子字节边界如何应用于DFLTCC-CMPR函数的实例；

[0033] 图6描绘了根据本发明的一个方面的没有压缩的数据块的一个实例；

[0034] 图7描绘了根据本发明的一个方面的具有使用固定-霍夫曼表 (FHT) 的压缩数据的块的一个示例；

[0035] 图8根据本发明的一个方面描绘了使用动态霍夫曼表 (DHT) 的具有压缩数据的块的一个示例；

[0036] 图9描绘了根据本发明的一个方面的存储器中的压缩数据集合的一个示例；

[0037] 图10描绘了根据本发明的一个方面的将数据压缩成压缩数据集的三个块的程序的样本的一个示例；

[0038] 图11描述了根据本发明的一个方面的在集合的第一压缩数据块上操作的DFLTCC-CMPR函数的参数块内容的一个实例；

[0039] 图12描述了根据本发明的一个方面的在集合的第二压缩数据块上操作的DFLTCC-CMPR函数的参数块内容的一个实例；

[0040] 图13描绘了根据本发明的一个方面的从压缩数据集解压缩数据的程序的样本的一个示例；

[0041] 图14A-14C描绘了根据本发明的一个方面的在执行DFLTCC-CMPR多次之前和之后的在线历史缓冲器的示例；

[0042] 图15A-15E描绘了根据本发明的一个方面的在执行DFLTCC多次之前和之后的循环历史缓冲器的示例；

[0043] 图16A-16C描绘了根据本发明的一个方面的在执行DFLTCC-XPND多次之前和之后的在线历史缓冲器的示例；

[0044] 图17描绘了根据本发明的一个方面的使用DEFLATE转换调用指令的一个示例；

[0045] 图18描绘根据本发明的一个方面的使用循环历史缓冲器的一个实例；

[0046] 图19描绘了根据本发明的实施方式的系统；

[0047] 图20描绘了根据本发明的实施例的工艺流程；

[0048] 图21描绘了根据本发明的实施例的系统；并且

[0049] 图22描绘了根据一个或多个实施例的处理系统。

[0050] 本文所描绘的图是说明性的。在不脱离本发明的精神的情况下, 这里描述的图或操作可以有許多变化。例如, 这些动作可按不同次序执行, 或动作可被添加、删除或修改。同样, 术语“耦合”及其变体描述了在两个元件之间具有通信路径并且不暗示这些元件之间的

直接连接,而在它们之间没有中间元件/连接。所有这些变化被视为说明书的一部分。

[0051] 在附图和所披露实施例的以下详细说明中,附图中展示的这些不同元件配备有两个或三个数字附图标记。在次要例外的情况下,每个参考号的最左边的数字对应于首先展示其元件的图。

具体实施方式

[0052] 根据一个或多个实施例,本发明涉及一种虚拟架构级,其包括复杂可中断指令的兼容性级,使得任何虚拟机可利用复杂可中断指令。对此,针对所有虚拟机中的最低公分母机器版本构建兼容性等级,其中最低公分母指示符标识该机器版本。

[0053] 根据本发明的一个方面,提供了一种促进计算环境中的处理的能力。作为一个示例,提供单个指令(例如,硬件/软件接口处的单个架构化硬件机器指令)来执行功能(也被称为操作),诸如压缩或解压缩功能,以压缩和/或解压缩(也被称为解压缩)数据。指令是由通用处理器上的程序(例如,操作系统或用户程序)分派的通用处理器指令集架构(ISA)的一部分。通过ISA指令进行压缩/解压缩,不需要操作系统进行任务切换来进行压缩/解压缩操作,节省了执行周期。进一步,通过使用单个指令来压缩和/或解压缩数据,减少了处理器(例如通用处理器)内的执行时间。

[0054] 在一个实例中,所述指令执行符合工业标准(被称作DEFLATE标准)的压缩和解压操作,且所述指令被称作DEFLATE转换调用指令。DEFLATE标准包括压缩数据符号的描述,其表示数据的原始形式(数据的未压缩形式)的重复串。这样的符号包括指针和重复串的长度,其描述先前被处理的与被处理的数据的当前位置相关的重复串的位置和长度。先前处理的未压缩形式的数据被称为历史。在一个示例中,历史是存储器中的连续字节数,其可以大到例如32K字节。

[0055] 参见图1A描述并入和使用本发明的一个或多个方面的计算环境的一个实施例。计算环境100包括例如处理器102(例如,中央处理单元),存储器104(例如,主存储器;又名,系统存储器、主存储器、中央存储器、存储器),以及经由彼此耦合的一个或多个输入/输出(I/O)设备和/或接口106,例如,一个或多个总线108和/或其他连接。

[0056] 在一个示例中,处理器102基于由纽约州阿蒙克市的国际商业机器公司提供的z/Architecture®硬件架构,并且是诸如IBM Z®服务器之类的服务器的一部分,该服务器也由国际商业机器公司提供并实现z/Architecture硬件架构。z/Architecture硬件架构的一个实施例在名称为“z/Architecture Principles of Operation”的出版物中描述,IBM公开号SA22-7832-11,第12版,2017年9月(“z/Architecture Principles of Operation,”IBM Publication No.SA22-7832-11,12th edition,September 2017),特此将其通过引用全文并入本文。然而,z/Architecture硬件架构仅是一个示例架构;其他架构和/或其他类型的计算环境可包括和/或使用本发明的一个或多个方面。在一个示例中,处理器执行操作系统,诸如也由国际商业机器公司提供的z/OS®操作系统。

[0057] 处理器102包括用于执行指令的多个功能组件。如图1B中所描绘的,这些功能组件包括:例如,指令提取组件120,用于提取要执行的指令;指令解码单元122,用于对所获取的指令进行解码并且获得经解码的指令的操作数;指令执行组件124,用于执行解码的指令;存储器访问组件126,用于在必要时访问存储器以用于指令执行;以及回写组件130,用于提

供所执行的指令的结果。根据本发明的一个或多个方面,这些组件中的一个或多个组件可以包括在压缩/解压缩处理(或可以使用本发明的一个或多个方面的其他处理)中使用的一个或多个其他组件的至少一部分或具有对在压缩/解压缩处理(或可以使用本发明的一个或多个方面的其他处理)中使用的一个或多个其他组件的访问权,如本文所描述的。一个或多个其他组件包括例如压缩/解压缩组件(或其他组件)136。

[0058] 参看图2来描述用以并入且使用本发明的一个或一个以上方面的计算环境的另一实例。在一个示例中,计算环境基于z/Architecture硬件架构;然而,计算环境可基于由国际商业机器公司或其他公司提供的其他架构。

[0059] 参照图2,在一个示例中,计算环境包括中央电子复合体(CEC)200。CEC200包括多个部件,例如耦接到一个或多个处理器(又名中央处理单元(CPU))204和输入/输出子系统206的存储器202(又名系统存储器、主存储器、主存储器、中央存储器、存储装置)。

[0060] 存储器202包括例如一个或多个逻辑分区208、管理逻辑分区的管理程序210和处理器固件212。管理程序210的一个示例是由纽约Armonk的国际商业机器公司提供的处理器资源/系统管理器(PR/SM™)管理程序。如本文所使用的,固件包括例如处理器的微代码。其包括例如在更高级机器代码的实现中使用的硬件级指令和/或数据结构。在一个实施例中,其包括例如通常作为微代码递送的专有代码,所述微代码包括可信软件或特定于底层硬件的微代码,并控制对系统硬件的操作系统访问。

[0061] 每个逻辑分区208能够充当单独的系统。即,每个逻辑分区可以被独立地重置,运行诸如z/OS操作系统或另一操作系统的客户操作系统220,并且用不同的程序222操作。在逻辑分区中运行的操作系统或应用程序看上去具有对完整和完整系统的访问,但是实际上,其仅一部分是可用的。

[0062] 存储器202耦合到处理器(例如,CPU)204,其是可以被分配给逻辑分区的物理处理器资源。例如,逻辑分区208包括一个或多个逻辑处理器,每个逻辑处理器表示可以被动态地分配给逻辑分区的物理处理器资源204的全部或共享。

[0063] 进一步,存储器202耦合到I/O子系统206。I/O子系统206可以是中央电子复合体的一部分或与其分离。它引导主存储器202和输入/输出控制单元230以及耦接到中央电子复合体的输入/输出(I/O)设备240之间的信息流。

[0064] 可以使用许多类型的I/O设备。一个特定类型是数据存储设备250。数据存储设备250可以存储一个或多个程序252、一个或多个计算机可读程序指令254和/或数据等。计算机可读程序指令可以被配置成执行本发明的各方面的实施例的功能。

[0065] 作为一个实例,每一处理器204包含高速缓冲存储器层级的至少一个高速缓冲存储器260(例如,本地高速缓冲存储器),所述高速缓冲存储器层级包含多个高速缓冲存储器级,包含一个或一个以上本地高速缓冲存储器和/或一个或一个以上共享高速缓冲存储器。进一步,在一个实施例中,本地高速缓存和存储器202耦合到压缩/解压缩组件(或其他组件)262,其用于执行数据的压缩和/或解压缩(和/或本发明的一个或多个方面的其他操作)中的一个或多个。在不同实例中,可存在执行这些任务的一个或一个以上组件。许多变化是可能的。

[0066] 在一个实施例中,处理器(例如,处理器204)获得指令(例如,DEFLATE转换调用指令),解码所述指令,执行所述指令的设置,包括翻译将由所述指令使用的地址,并且将指令

的命令发送到耦合到处理器的组件,例如组件262,执行由所述指令指定的功能。组件262可访问高速缓存层级和存储器,使得在执行指定功能时,它读取数据、处理它并将经处理的数据存储回。作为示例,组件262是硬件组件。

[0067] 在进一步的实施例中,组件262的至少一部分被包括作为处理器的一部分。许多变化是可能的。

[0068] 中央电子复合体200可以包括和/或耦合到可移除/不可移除、易失性/非易失性计算机系统存储媒质。例如,它可以包括和/或耦合到不可移动的非易失性磁性介质(通常称为“硬盘驱动器”),一个磁盘驱动器,用于从一个可拆卸的非易失性磁盘(例如,“软盘”)中读取和向其写入,和/或光盘驱动器,用于从可移除的非易失性光盘读取或向可移除的非易失性光盘写入,例如CD-ROM、DVD-ROM或其他光学介质。应当理解,其他硬件和/或软件组件可以与中央电子复合体200结合使用。示例包括但不限于:微代码、设备驱动器、冗余处理单元、外部磁盘驱动器阵列、RAID系统、磁带驱动器和数据归档存储系统等。

[0069] 进一步,中央电子复合体200可以与许多其他通用或专用计算系统环境或配置一起操作。可以适用于中央电子复合体200的公知的计算系统、环境和/或配置的例子包括,但不限于个人计算机(PC)系统、服务器计算机系统、瘦客户机、厚客户机,手持式或膝上型设备、多处理器系统、基于微处理器的系统、机顶盒、可编程消费电子产品,网络PC、小型计算机系统、大型计算机系统和包括任何上述系统或设备的分布式云计算环境,等等。

[0070] 虽然本文描述了计算环境的不同实例,但本发明的一个或一个以上方面可与许多类型的环境一起使用。本文提供的计算环境仅是示例。

[0071] 根据本发明的一个方面,诸如计算环境100或中央电子复合体200的计算环境采用提供压缩和解压缩数据的机制的转换设施。在一个示例中,转换设施是DEFLATE转换设施,其提供使用DEFLATE压缩数据格式来压缩和解压缩数据的机制。在一个示例中,当设施指示符被设置为例如一个时,转换设施被安装在系统中。作为z/Architecture硬件架构的一个特定实例,当以z/Architecture架构模式安装转换设施时,将设施位151设定为(例如)一。该设施包括例如DEFLATE转换调用指令,其实施例在下面描述。

[0072] 在一个示例中,DEFLATE转换调用指令执行与在数据的原始(未压缩)形式之间转换数据的状态相关的功能,以及数据的压缩表示,如由所选标准(诸如IETF(互联网工程任务组)RFC(请求注解)1951规范)指定的,DEFLATE压缩数据格式规范版本1.3互联网工程任务组,请求评论1951,1996年5月(DEFLATE Compressed Data Format Specification version 1.3Internet Engineering Task Force,Request for Comments 1951,May 1996)中描述。

[0073] 在一个示例中,未压缩数据是字节序列,并且数据的压缩表示包括符号。符号表示未压缩数据的单独字节,称为字面字节,或者表示未压缩数据的字节的重复出现的序列,称为重复串。作为实例,霍夫曼表指定压缩数据符号和未压缩数据之间的编码和解码。存在两种类型的霍夫曼表:固定-霍夫曼表(FHT),其是包括例如所有可能的编码的预定规范;以及动态霍夫曼表(DHT),其是专门针对要被压缩的数据创建的编码的集合,其可以是所有可能编码的子集。用DHT生成的数据的压缩表示通常小于用FHT生成的相同数据的压缩表示。最近处理的未压缩数据的一部分(被称为历史)被维持用于对表示重复串的压缩数据符号进行编码和解码。历史是重复字符串的参考源。当在操作期间处理数据时更新历史。如所指

示,在一个实例中,DEFLATE转换调用指令使用DEFLATE压缩数据格式,其在RFC1951,DEFLATE压缩数据格式规范版本1.3中描述。应用于DEFLATE转换调用指令的DEFLATE标准的属性包括例如:

[0074] • 压缩的数据集包括一系列块。存在三种类型的块。一种类型包括3字节报头,其后是长度信息和未压缩数据,两种类型的块包括3字节报头,其后是压缩数据元素。

[0075] • 压缩的数据元素可包括动态霍夫曼表的压缩的表示、压缩的数据符号和块结束(EOB)符号。

[0076] • 压缩的数据元素具有不同位长度。

[0077] • 经压缩的数据元素可在存储中的字节边界之间开始或结束。

[0078] • 压缩的数据元素按照例如从最右边的字节位置到最左边的字节位置的顺序被加载到字节中。

[0079] 当压缩的数据元素占据存储中的字节的一部分而不是全部时,访问存储中的整个字节。储存操作数长度指定可寻址字节的数目,其可指定比压缩数据占据的位多的位。

[0080] 下文进一步描述关于经压缩数据块的额外细节。

[0081] 参见图3A-3L描述DEFLATE转换调用(DFLTCC)指令的一个实施例。在一个示例中,使用通用处理器(例如,处理器102或204)来执行指令。在本文的描述中,指示特定位置、特定字段和/或字段的特定大小(例如,特定字节和/或位)。然而,可以提供其他位置、字段和/或大小。进一步,虽然指定将位设定为特定值(例如,一或零),但这仅是实例。在其他实例中,可将所述位设定为不同值,例如相反值或另一值。许多变化是可能的。

[0082] 在一个实施例中,程序(例如,操作系统或用户程序)可执行DEFLATE转换调用指令多次以压缩或解压缩单个数据流。例如,当应用压缩或解压缩大数据流(例如,大于1M字节)时,操作可包括多个调用以压缩或解压缩数据流的经缓冲部分。根据本发明的一个方面,程序声明缓冲器(例如,32K字节的缓冲器),该缓冲器用于累积在跨越DEFLATE转换调用指令的多个执行的操作期间处理的未压缩数据的历史。缓冲器被称为循环历史缓冲器,其使用DEFLATE转换调用指令来定义,如本文所描述的。

[0083] 参见图3A,在一个示例中,DEFLATE转换调用(DFLTCC)指令300的格式是表示具有扩展操作码(操作码)字段和附加寄存器字段的寄存器和寄存器操作的RRF格式。作为示例,指令包括:操作码字段302(例如,位0-15),其具有指示DEFLATE转换调用操作的操作码;指定第一对通用寄存器的第一寄存器字段(R1)304(例如,位24-27);指定第二对通用寄存器的第二寄存器字段(R2)306(例如,位28-31);以及指定第三通用寄存器的第三寄存器字段(R3)308(例如,位16-19)。由R1字段304指定的寄存器的内容指定第一操作数的位置(在存储中);R2字段306指定的寄存器的内容指定第二操作数(在存储中)的位置;并且R3字段308指定的寄存器的内容指定(存储装置中的)第三操作数的位置。R1+1的内容指定第一操作数的长度,R2+1的内容指定第二操作数的长度。在一个实例中,保留指令的位20到23且应含有零;否则,程序在未来可能不兼容地操作。如本文所使用的,程序是发布DEFLATE转换调用指令的程序。它可以是用户程序、操作系统或另一种类型的程序。

[0084] 在一个实施例中,指令的执行包括使用一个或多个隐含的通用寄存器(即,指令未明确指定的寄存器)。例如,通用寄存器0和1用于执行DEFLATE转换调用指令,如本文所述。在一个实例中,使用通用寄存器0来指定待执行的功能(以及下文描述的历史缓冲器类型),

且使用通用寄存器1来提供由指令使用的参数块的位置。

[0085] 作为实例,参考图3B,通用寄存器0(309)包括历史缓冲器类型字段310和功能代码字段312。在一个特定实例中,通用寄存器0的位位置56包含历史缓冲器类型,且通用寄存器0的位位置57到63含有函数代码;但是在其他实施例中,可以使用其他位来包含历史缓冲器类型和/或功能代码。在一个实例中,当通用寄存器0的位57-63指定未指派或卸载的功能代码时,辨识出规格异常。

[0086] 用于DEFLATE转换调用指令的示例分配的函数代码在图3C中示出,并且包括例如:指示DFLTCC-QAF(查询可用函数)函数的函数代码0(313);指示DFLTCC-GDHT(生成动态霍夫曼表)函数的函数代码1(315);指示DFLTCC-CMPR(压缩)函数的函数代码2(317);以及指示DFLTCC-XPND(扩展)函数的函数代码4(319)。每一码使用参数块,且在一实例中,参数块的大小取决于函数。例如,对于DFLTCC-QAF函数,参数块为32字节;对于DFLTCC-GDHT函数,参数块为384字节;对于DFLTCC-CMPR和DFLTCC-XPND函数,参数块为1536字节。在此实例中,未指派其他函数代码。尽管描述了示例功能和功能代码,但是可以使用其他功能和/或功能代码。

[0087] 当所指定的函数是DFLTCC-CMPR或DFLTCC-XPND时,通用寄存器0的位56指定在操作期间使用的历史缓冲器类型(HBT)。当HBT为零时,历史缓冲器称为在线历史缓冲器。当使用在线历史缓冲器时,当指定DFLTCC-CMPR时,历史例如紧靠第二操作数的左侧,并且当指定DFLTCC-XPND时,历史例如紧靠第一操作数的左侧。当HBT为1时,历史缓冲器称为循环历史缓冲器。当使用循环历史缓冲器时,当指定DFLTCC-CMPR或DFLTCC-XPND时,该历史是第三操作数的一部分或全部。当指定DFLTCC-QAF或DFLTCC-GDHT函数时,忽略一般寄存器0的位56。在一个实例中,忽略通用寄存器0的位位置0至31。进一步,在一个实例中,保留通用寄存器0的位位置32到55且应含有零;否则,程序在未来可能不兼容地操作。

[0088] 参见图3D描述关于由DEFLATE转换调用指令使用的另一隐式寄存器(通用寄存器1)的进一步细节。通用寄存器1(314)的内容例如指定存储装置中的参数块的最左边字节的逻辑地址316。在一个实例中,在4K字节的边界上指定参数块;否则识别出规格异常。下文进一步描述关于参数块的进一步细节。

[0089] 对于指定的函数(例如,DFLTCC-QAF、DFLTCC-GDHT、DFLTCC-CMPR、DFLTCC-XPND),不修改通用寄存器0、1和R3的内容。进一步,在一个实例中,R1字段304指定通用寄存器的奇偶对。指定偶数寄存器,不指定通用寄存器0;否则,识别规格异常。

[0090] 如图3E-3F中所描绘的并且在此进一步详细描述,通用寄存器R1 318的内容指示第一操作数地址320,并且通用寄存器R1+1 322的内容用于确定第一操作数的长度324。例如,当所指定的函数为DFLTCC-CMPR或DFLTCC-XPND时,通用寄存器R1318的内容指定第一操作数的最左字节的逻辑地址。当所指定的函数为DFLTCC-CMPR时,通用寄存器R1+1的内容连同新任务(NT)和参数块(下文描述)的子字节边界(SBB)字段的值指定第一操作数的长度。下表提供说明DFLTCC-CMPR函数的第一操作数的长度为通用寄存器R1+1、NT字段和SBB字段的内容的函数的实例:

	GR R1+1 的内容 (十六进制)	NT	SBB (二进制)	第一操作数的长度
[0091]	00000000 00000002	0	001	15 位
	00000000 00000001	1	---	8 位
	00000000 00000001	0	000	8 位
	00000000 00000001	0	011	5 位
	00000000 00000001	0	111	1 位
	00000000 00000000	-	---	0 位

[0092] 当指定函数为DFLTCC-XPND时,通用寄存器R1+1的内容指定第一操作数的长度。当指定函数为DFLTCC-CMPR或DFLTCC-XPND时,将数据压缩或解压缩的结果存储在第一操作数位置处。当指定DFLTCC-QAF或DFLTCC-GDHT功能时,忽略通用寄存器R1和R1+1的内容。

[0093] 此外,在一个示例中,对于指定的函数(例如,DFLTCC-QAF、DFLTCC-GDHT、DFLTCC-CMPR和DFLTCC-XPND),R2字段306指定偶数-奇数对通用寄存器。指定偶数寄存器,不指定通用寄存器0;否则,识别规格异常。

[0094] 如图3G-3H中所描绘且在本文中进一步详细描述,通用寄存器R2 326的内容指示第二操作数地址328,且通用寄存器R2+1330的内容用以确定第二操作数的长度332。例如,当指定函数为DFLTCC-GDHT、DFLTCC-CMPR或DFLTCC-XPND时,通用寄存器R2的内容指定第二操作数的最左字节的逻辑地址。当指定函数为DFLTCC-CMPR或DFLTCC-GDHT时,通用寄存器R2+1的内容指定第二操作数的长度。当指定函数为DFLTCC-XPND时,通用寄存器R2+1的内容连同参数块的NT字段和SBB字段的值指定第二操作数的长度。当第二操作数长度被引用并且在指令的执行开始时具有非零值时,从第二操作数位置取回数据。当引用第二操作数长度时,在指令的执行开始时具有值0,并且参数块的继续标志(CF)字段在指令的执行开始时为1,第二操作数不被访问。

[0095] 当指定DFLTCC-QAF函数时,忽略通用寄存器R2和R2+1的内容。当指定DFLTCC-GDHT函数且通用寄存器R2+1的内容指定等于零的长度时,辨识出指定异常,且不存取第二操作数。当指定DFLTCC-CMPR或DFLTCC-XPND函数时,在指令的执行开始时参数块的继续标记(CF)字段为零,并且通用寄存器R2+1的内容指定等于零的长度,识别指定异常,并且不访问第二操作数。

[0096] 如图3I所示,当指定的功能是DFLTCC-CMPR或DFLTCC-XPND并且历史缓冲器类型(HBT)是环形的(例如,HBT310=1),通用寄存器R3335的内容指定循环历史缓冲器地址337。例如,指定第三操作数的最左字节的逻辑地址。指定例如4K字节的边界;否则识别出规范异常。在一个示例中,循环历史缓冲器位于第三操作数位置处。当指定的函数是DFLTCC-CMPR或DFLTCC-XPND并且HBT是零时,忽略通用寄存器R3的内容。当指定DFLTCC-QAF或DFLTCC-GDHT功能时,忽略通用寄存器R3的内容。对于指定的函数(例如,DFLTCC-QAF、DFLTCC-GDHT、DFLTCC-CMPR和DFLTCC-XPND),R3字段不指定通用寄存器0或通用寄存器1;否则,在一个示例中,识别规格异常。

[0097] 作为操作的部分,当指定函数为DFLTCC-CMPR时,通用寄存器R1中的地址递增包括处理位位置0的第一操作数的处理的字节数,且通用寄存器R1+1中的长度递减相同数;通用寄存器R2中的地址递增第二操作数的处理的字节的数目,且通用寄存器R2+1中的长度递减相同数目。包括处理位位置0的第一操作数的处理的字节数是例如由整数除法产生的整数

商,其中,除数是所处理的输出位的数量与SBB的原始值的总和,除数是值8。地址和长度的形成和更新取决于寻址模式,如下所述。

[0098] 作为操作的一部分,当指定函数为DFLTCC-XPND时,通用寄存器R1中的地址递增第一操作数的处理的字节数,并且通用寄存器R1+1中的长度递减相同的数;通用寄存器R2中的地址递增包含处理位位置0的第二操作数的处理的字节的数目,且通用寄存器R2+1中的长度递减相同数目。包括处理位位置0的第二操作数的处理的字节的数量是从整数除法得到的整数商,其中除数是所处理的输入位的数量和SBB的原始值的总和,除数是值8。地址和长度的形成和更新取决于寻址模式,如下所述。

[0099] 在24位寻址模式中,在一个实施例中,以下适用:

[0100] • 通用寄存器1、R1、R2和R3的位位置40-63的内容分别构成参数块、第一操作数、第二操作数和循环历史缓冲器的地址,并且位位置0-39的内容被忽略。

[0101] • 经更新的第一操作数及第二操作数地址的位40至63分别替换通用寄存器R1及R2中的对应位。忽略更新地址的位位置40的执行,且通用寄存器R1和R2的位位置32-39的内容设定为零。通用寄存器R1和R2的位位置0至31的内容保持不变。当指令以部分或正常完成结束,并且更新的操作数地址等于指令的执行开始时的操作数地址时,对应的通用寄存器的位位置32-39被设定为零。

[0102] • 通用寄存器R1+1和R2+1的位位置32-63的内容形成例如32位无符号二进制整数,其分别指定第一和第二操作数中的字节数。忽略通用寄存器R1+1和R2+1的位位置0至31的内容。

[0103] • 经更新的第一操作数及第二操作数长度的位32-63分别替换通用寄存器R1+1及R2+1中的对应位。通用寄存器R1+1和R2+1的位位置0至31的内容保持不变。

[0104] 在31位寻址模式中,在一个实施例中,以下适用:

[0105] • 通用寄存器1、R1、R2和R3的位位置33-63的内容分别构成参数块、第一操作数、第二操作数和循环历史缓冲器的地址,并且位位置0-32的内容被忽略。

[0106] • 经更新的第一操作数及第二操作数地址的位33-63分别替换通用寄存器R1及R2中的对应位。忽略更新地址的位位置33的执行,且通用寄存器R1和R2的位位置32的内容设定为零。通用寄存器R1和R2的位位置0至31的内容保持不变。当指令以部分或正常完成结束,并且更新的操作数地址等于指令的执行开始时的操作数地址时,对应的通用寄存器的位位置32被设定为零。

[0107] • 通用寄存器R1+1和R2+1的位位置32-63的内容形成分别规定第一和第二操作数中的字节数的32位无符号二进制整数。忽略通用寄存器R1+1和R2+1的位位置0至31的内容。

[0108] • 经更新的第一操作数及第二操作数长度的位32-63分别替换通用寄存器R1+1及R2+1中的对应位。通用寄存器R1+1和R2+1的位位置0至31的内容保持不变。

[0109] 在64位寻址模式中,在一个实施例中,以下适用:

[0110] • 通用寄存器1、R1、R2和R3的位位置0至63的内容分别构成参数块、第一操作数、第二操作数和循环历史缓冲器的地址。

[0111] • 经更新的第一操作数及第二操作数地址的位0至63分别替换通用寄存器R1及R2中的对应位。忽略更新地址的位位置0的执行。

[0112] • 通用寄存器R1+1和R2+1的位位置0至63的内容形成64位无符号二进制整数,其

分别指定第一和第二操作数中的字节数。

[0113] • 经更新的第一操作数及第二操作数长度的位0至63分别替换通用寄存器R1+1及R2+1中的对应位。

[0114] 在访问寄存器模式中,访问寄存器1、R1、R2和R3分别指定包含参数块、第一操作数、第二操作数和循环历史缓冲器的地址空间。当在存取寄存器模式中指定具有在线历史(in-line)缓冲器的DFTCC-CMPR时,存取寄存器R2指定含有在线历史的地址空间。当在访问寄存器模式中指定具有在线历史缓冲器的DFTCC-XPND时,访问寄存器R1指定包含在线历史的地址空间。

[0115] 关于不同功能的进一步细节描述如下:

[0116] 函数代码0:DFLTCC-QAF(查询可用函数)

[0117] DFLTCC-QAF(查询可用函数)函数提供用于指示所安装函数和所安装参数块格式的可用性的机制。参考图3J描述DFLTCC-QAF函数的参数块的一个示例格式。在一个示例中,DFLTCC-QAF函数(例如,函数代码0)的参数块340包括安装的函数向量342和安装的参数块格式向量346。在一个特定实例中,这些向量分别存储到参数块的字节0到15和字节24到25。下文进一步描述这些载体中的每一者。

[0118] 作为实例,所安装函数向量342的位0至127分别对应于DEFLATE转换调用指令的函数代码0至127。当一个字节是例如1时,安装相应的功能;否则,不安装功能。

[0119] 进一步,在一个示例中,所安装的参数块格式向量346的位0-15分别对应于DFLTCC-GDHT、DFLTCC-CMPR和DFLTCC-XPND函数的参数块格式0-15。当某个字节位为1时,则安装相应的参数块格式,否则不安装。在一个实例中,将零存储到参数块的保留字节16到23和26到31。

[0120] 尽管关于参数块340描述了某些字段,但是在其他实施例中可以包括附加的、更少的和/或其他的字段。

[0121] 在一个实施例中,DFLTCC-QAF函数忽略通用寄存器R1、R2、R3、R1+1和R2+1的内容。

[0122] 当可应用时,PER(程序事件记录)储存变更事件被识别用于参数块。当可应用时,PER零地址检测事件被识别用于参数块。

[0123] 在一个示例中,当DFLTCC-QAF函数的执行完成时,设置条件码0;在一个实例中,条件码1、2和3不适用于查询函数。

[0124] 函数代码1:DFLTCC-GDHT(生成动态霍夫曼表)

[0125] 当指定DFLTCC-GDHT函数时,第二操作数例如用作源以生成如由DEFLATE标准指定的动态霍夫曼表(DHT)的压缩表示。

[0126] 在一个实例中,DFLTCC-GDHT功能使用参数块,参照图3K描述该参数块的实例。在本文所描述的实例参数块中,指示用于特定字段和字段的特定大小的参数块内的特定位置(例如,特定字节和/或位)。然而,可以为一个或多个字段提供其他位置和/或大小。进一步,虽然指定将位设定为特定值(例如,一或零),但这仅是实例。在其他实例中,可将所述位设定为不同值,例如相反值或另一值。许多变化是可能的。

[0127] 另外,在一个实例中,参数块包含一个或一个以上保留字段和一个或一个以上保留字段。DFLTCC-GDHT功能不修改保留字段。将保留字段与保留字段区分开,以使得程序能够初始化单个存储位置,将该存储位置用于DFLTCC-GDHT函数的参数块,并且随后将相同的

存储位置用于DFLTCC-CMPR函数的参数块。保留字段包含零;否则,程序在未来可能不兼容地操作。当操作结束时,保留字段可以被存储为零或者可以保持不变。

[0128] 更进一步,一些字段由其他函数(例如,DFLTCC-CMPR或DFLTCC-XPND)使用,并且因此,与这些函数相关的方面也可以与这些字段的描述一起描述。在一个实例中,用于DFLTCC-GDHT函数的参数块360包括以下字段:

[0129] 参数块版本号(PBVN) 362:参数块的字节0-1指定参数块的版本和大小。PBVN的字节0-11被保留并且应该包含零;否则,程序将来可能不能兼容地操作。PBVN的字节12-15包含指定参数块的格式的无符号二进制整数。DFLTCC-QAF功能提供机制以指示可用的参数块格式。当指定的参数块的格式不被模型支持时,识别通用操作数数据异常。PBVN由程序指定,并且在指令的执行期间不被修改。

[0130] 模型版本号(MVN) 363:参数块的字节2是识别执行指令的模型的无符号二进制整数。不需要程序来初始化MVN。MVN在指令的执行期间被更新。MVN中存储的值是依赖于模型的。

[0131] 动态霍夫曼表(DHT)生成控制(DHTGC) 364:参数块的字节17的位2适用于生成动态霍夫曼表(DHT)。DHT指定用于表示字面字节、重复串长度、块结束(EOB)符号和重复串指针距离的符号的霍夫曼码。特定符号的霍夫曼代码的值是以数据的未压缩形式表示的实体的出现计数的函数。当符号的计数为零时,在符号的DHT中不存在霍夫曼代码。DHTGC指定等于零的计数将被如下处理,在一个示例中:

[0132] DHTGC 含义

[0133] 0 处理字面字节、重复串长度和等于零的指针距离的计数为等于1(生成通用DHT)。

[0134] 1 处理等于零的重复串长度和指针距离的计数为等于一。

[0135] 指定用于字面字节、EOB符号、复制串长度和复制串指针距离的每个可能值的霍夫曼代码的DHT被称为通用DHT。不指定用于以未压缩形式的数据出现的字面字节、重复串长度或重复串指针距离的值的霍夫曼代码的DHT被称为非通用DHT。

[0136] 对于DHTGC的所有值,所得DHT为所有可能的复制串长度和指针距离指定霍夫曼码,如由DEFLATE标准所定义的。因此,下面进一步描述的所得DHT的压缩形式的HLIT(霍夫曼字面)和HDIST(霍夫曼距离)子元素各自包含例如29的值。

[0137] 当指定DFLTCC-GDHT功能时,DHTGC是操作的输入。当指定DFLTCC-CMPR或DFLTCC-XPND功能时,DHTGC不适用于操作。在一个实施例,在执行指令期间不修改DHTGC。

[0138] 操作结束补充代码(OESC) 365:参数块的字节19是在被报告给程序的条件下提供附加信息的无符号二进制整数。由于该字段由多个函数使用,所以一些条件涉及由其他函数使用的参数块(例如,由DFLTCC-CMPR和DFLTCC-XPND函数使用的图3L的参数块)的字段。当所报告的条件是通用操作数数据异常时,尽管更新了参数块的OESC字段,但是认为操作受到抑制,在这种情况下,在一个实例中,定义如下:

OESC (十六进制)	含义
00	不提供附加信息。
01	模型不支持参数块版本号 362 指定的参数块格式。
02	DFLTCC-CMPR 或 DFLTCC-XPND 函数被指定, 历史长度字段 385 (图 3L) 大于例如 32, 768, 并且新任务字段 374 (图 3L) 为零。
11	遇到 BTYPE (块类型) 等于 11 二进制的压缩数据块。
12	遇到 BTYPE 等于 00 二进制且 NLEN 不等于 LEN (长度) 的一的补码的经压缩数据块。
21	CDHTL 字段 366 (图 3L) 适用并且例如小于 42 或大于 2283。
22	在操作期间使用的压缩 DHT 的 HLIT 子元素例如大于 29 (无效 DHT)。
23	在操作期间使用的压缩 DHT 的 HDIST 子元素例如大于 29 (无效 DHT)。
24	在操作期间使用的压缩 DHT 指定代码, 该代码在指定针对压缩 DHT 定义的可能代码长度 (例如, 19) 的字节长度的代码序列中, 并且小于霍夫曼算法指定功能霍夫曼树 (无效 DHT) 所需的长度。
26 [0139]	在操作期间使用的压缩的 DHT 将代码长度 16 (复制先前的代码长度) 指定为由字面字节、EOB 符号和重复串长度 (无效 DHT) 组成的元素集合的第一代码长度。
27	在操作期间使用的压缩 DHT 指定代码, 该代码在指定字面字节的代码长度的代码序列中, 并且该代码不匹配被确定为表示所引用的代码长度的集合的代码中的任何代码, 如之前在压缩 DHT (无效 DHT) 中所指定的。
28	在操作期间使用的压缩的 DHT 指定将码长 0 (CLO) 分配给 EOB 符号的码。在这种情况下, 相应的 DHT 不指定霍夫曼代码来表示 EOB 符号 (无效 DHT)。
29	在操作期间使用的压缩 DHT 指定代码序列中的代码, 所述代码指定用于重复串长度和指针距离的代码长度, 并且所述代码不匹配被确定为表示参考代码长度的集合的代码中的任何代码, 如压缩 DHT (无效 DHT) 中较早指定的。
2A	在操作期间使用的压缩的 DHT 指定大于 DHT 中的霍夫曼代码的数量的代码长度的数量, 如由 HLIT 字段、HDIST 字段和例如 258 中的值的总和指定的。作为示例 (无效 DHT), 这可能具有代码长度 16、17 和 18 的不正确使用。
2B	在操作期间使用的压缩 DHT 指定用于字面字节集、EOB 符号和重复串长度的代码长度, 其小于由霍夫曼算法指定功能霍夫曼树 (无效 DHT) 所需的长度。

2	在操作期间使用的压缩的 DHT 指定用于复制串指针距离的集合的代码长度,其小于由霍夫曼算法指定功能霍夫曼树(无效 DHT)所需的长度。
2F	CDHTL 字段 366 (图 3L) 适用并且不等于在操作期间使用的 CDHT 字段 367 (图 3L) 中的压缩 DHT 的长度。
31	在操作期间使用的压缩 DHT 不指定对应于在操作期间处理的字面字节或重复串长度的霍夫曼代码(缺乏非通用 DHT),或者,规定 DFLTCC-XPND 函数和 BTYPE 等于 01 二进制的压缩数据块中遇到的压缩数据符号,指定重复串长度(11000110 或 11000111 二进制)的无效代码。
[0140]	
32	在操作期间使用的压缩的 DHT 不指定与在操作期间处理的复制串指针距离对应的霍夫曼代码(缺乏的非通用 DHT),或者,规定 DFLTCC-XPND 函数和 BTYPE 等于 01 二进制的压缩数据块中遇到的压缩数据符号,指定用于复制串指针距离(11110 或 11111 二进制)的无效代码。
40	遇到压缩的数据符号,其是复制串指针并且指定比在处理符号的点处可用的历史的长度更大的距离。

[0141] 当操作结束而不报告一般操作数数据异常时,将零存储到 OESC 字段。

[0142] 对除了零之外的补充代码的支持是依赖于模型的。当存在多个条件时,在 OESC 字段中报告哪个代码(如果有的话)是依赖于模型的。

[0143] 压缩动态霍夫曼表长度(CDHTL) 366:参数块的从字节56的位4开始到字节57的位7的12位包含无符号二进制整数,该整数在参数块的CDHT字段(例如,CDHT367)中将DHT的压缩格式的长度指定为位计数。

[0144] CDHTL是当指定DFLTCC-GDHT功能时来自操作的输出。

[0145] 当指定DFLTCC-CMPR功能并且霍夫曼表类型(例如,图3L的HTT 376)是1时,CDHTL是对操作的输入。当CDHTL未指定CDHT的适当长度时,识别一般操作数数据异常。当指定DFLTCC-CMPR功能时,不修改CDHTL。

[0146] 当指定DFLTCC-XPND函数并且在仅解码具有BTYPE 10二进制的块的一部分之后操作结束时,将块中的DHT的压缩表示的长度存储到该字段。当指定DFLTCC-XPND函数并且操作在块边界处结束时,或者在利用BTYPE 00或01二进制解码块的仅一部分之后,将零存储到该字段。当在具有BTYPE 10二进制的块内恢复解压操作时(即,当CF(图3L的继续标记373)等于1且IFS(不完整功能状态383)等于C或D hex时,如下所述),该字段是操作的输入。

[0147] 压缩动态霍夫曼表(CDHT) 367:参数块的字节64-351包含动态霍夫曼表(DHT)的压缩格式。

[0148] DHT指定霍夫曼码(字节序列)以表示两组元素。一个集合的元素包括字面字节、EOB符号和重复串长度。另一集合的元素包括重复的串指针距离。DHT的压缩表示定义代码长度的集合并指定每一集合的每一元素的代码长度(CL)。期望在操作期间参考的元素的霍夫曼代码从为该元素指定的CL和具有相同指定CL的相同集合中的元素的数目导出。具体地,DHT的压缩表示例如包括:

[0149] • HLIT字段,用于指定表示文字量字节、EOB符号和重复串长度的霍夫曼码的数目。

- [0150] • HDIST字段,用于指定表示重复串指针距离的霍夫曼码的数目。
- [0151] • HLEN(霍夫曼码长度) 字段,用于指定表示码长的霍夫曼码的数目。
- [0152] • 一码序列,其指定每一位长度,例如,为该压缩DHT定义的19码长度。
- [0153] • 代码序列,所述代码序列指定由所述字面字节、EOB符号和重复串长度组成的所述集合的所述元素中的每个元素的代码长度。
- [0154] • 代码序列,其指定由重复串指针距离组成的所述集合的所述元素中的每一者的代码长度。
- [0155] 下文参考具有块类型10二进制的经压缩数据块的描述来描述DHT的经压缩表示的进一步细节。
- [0156] 在一个示例中,DHT的压缩表示在CDHT字段中保持对齐。即,字节64的最右边位包含DHT的压缩表示的HLIT子元素的最低有效位。
- [0157] 当指定DFLTCC-GDHT功能时,DHT的压缩表示是来自操作的输出。
- [0158] 当指定DFLTCC-CMPR功能并且下文描述的HTT为1时,DHT的压缩表示是对操作的输入。CDHT字段不被DFLTCC-CMPR功能修改。
- [0159] 当DFLTCC-XPND函数被指定并且操作仅在用BTYP 10二进制解码块的一部分之后结束时,块中的DHT的压缩表示被存储到该字段。当指定DFLTCC-XPND函数并且操作在块边界处结束时,或者在利用BTYP 00或01二进制解码块的仅一部分之后,将零存储到该字段。当在具有BTYP10二进制的块内恢复解压缩操作时(即,当CF等于1且IFS等于C或D十六进制时),该字段是操作的输入。
- [0160] 当CDHT被修改时,未被用于表示DHT的压缩表示的字段的位被存储为零。
- [0161] 尽管上文相对于参数块360描述了不同字段,但是在其他实施例中可以包括附加的、更少的和/或其他的字段。
- [0162] DHT生成的各方面由程序使用参数块的动态霍夫曼表生成控制(DHTGC) 字段364来指定给机器。既定源包含未压缩数据,且在完成操作之后,用DFLTCC-CMPR功能指定所产生的结果以压缩同一源。
- [0163] 在一个实施例中,在处理当前操作时不存在来自先前操作的要引用的历史。
- [0164] 在一个实例中,当通用寄存器R2+1的内容指定大于例如32K字节的长度时,应用以下内容:
- [0165] • 仅第二操作数的前32K字节用于生成DHT。
- [0166] • 对于超出第二操作数的第一32K字节的位置,不辨识存取异常。
- [0167] 当通用寄存器R2+1的内容指定等于零的长度时,辨识出指定异常,且不存取第二操作数。
- [0168] 所产生的压缩DHT包括表示块结束(EOB)符号的霍夫曼代码。
- [0169] 所生成的DHT的压缩格式被存储到参数块的压缩动态霍夫曼表(CDHT) 字段367。所生成的DHT的压缩格式的长度被存储在参数块的CDHTL字段366中。
- [0170] 所述操作包含将模型识别存储到参数块的模型版本号字段363。
- [0171] 当操作结束而不识别一般操作数数据异常时,将零存储到参数块的操作结束补充代码(OESC) 字段365。
- [0172] 当DFLTCC-GDHT函数的执行完成时设置条件码0;条件码1、2和3不适用于DFLTCC-

GDHT函数。

[0173] 操作不修改通用寄存器R2和R2+1。

[0174] 当指定DFLTCC-GDHT功能时,忽略通用寄存器R1、R1+1和R3的内容。

[0175] 当可应用时,PER零地址检测事件被识别为第二操作数位置和参数块。

[0176] 函数代码2:DFLTCC-CMPR(压缩)

[0177] 当指定DFLTCC-CMPR功能时,执行压缩操作。所述操作包括将来自第二操作数位置的数据编码为压缩数据符号,所述压缩数据符号存储到第一操作数位置。

[0178] 在一个实例中,DFLTCC-CMPR函数使用参数块,参照图3L描述该参数块的实例。上文已关于参数块360描述了一些字段,且因此下文用相同参考数字列出且不进一步详细描述。

[0179] 在一个实例中,参数块370包含:

[0180] 参数块版本号(PBVN)362。

[0181] 模型版本号(MVN)363。

[0182] 继续旗标(CF)373:参数块的位63(当为1时)指示操作部分完成,且继续状态缓冲器的内容(例如,在继续状态缓冲器字段392中)可用以恢复操作。程序将继续旗标(CF)初始化为零,且在为了恢复操作的目的而重新执行指令的情况下不修改CF;否则结果是不可预测的。

[0183] 新任务(NT)374:参数块的字节16的位0在为1时指示操作应用于经压缩数据集的开始。因此,没有来自先前操作的历史和检查值适用于当前操作。当NT在操作的开始处为1且操作在部分完成之后结束时,将零存储到NT字段。当NT为零时,来自先前操作的历史和检查值应用于当前操作。

[0184] 校验值类型(CVT)375:参数块的字节16的位2指定包含在参数块的校验值字段(例如,字段387)中的校验值的类型。当CVT是零时,校验值类型是例如32位循环冗余校验(CRC-32)。在CVT为1的情况下,校验值种类例如为32位的Adlerchecksum(Adler-32)。CVT钻头在指令的执行期间不修改。

[0185] 霍夫曼表类型(HTT)376:当零时,参数块的字节16的位4指定包含由DEFLATE标准定义的固定霍夫曼码(FHT)的表在压缩操作期间被使用。当HTT是1时,在压缩操作期间使用包含如在参数块的CDHT字段中指定的动态霍夫曼码(DHT)的表。HTT不适用于解压缩操作。HTT位在指令的执行期间不被修改。

[0186] 块连续标志(BCF)377:当指定DFLTCC-CMPR功能时,应用参数块的字节16的位5。当零时,3字节块报头以及当适用时,在存储任何压缩的数据元素之前将如在参数块的CDHT字段(例如,字段367)中指定的动态霍夫曼表的压缩格式存储到第一操作数位置。当为1时,DHT的块头部和压缩格式都不存储到第一操作数位置。当NT为1时,BCF被视为等于0。BCF位在指令的执行期间不被修改。

[0187] 块关闭控制(BCC)378:当指定DFLTCC-CMPR功能时,应用参数块的字节16的位6。当在存储所有压缩数据符号之后为1时,将块结束(EOB)符号存储到第一操作数位置。当HTT使用FHT指定时,作为实例,霍夫曼代码00000000二进制(其对应于指定用于文字量字节、EOB符号和重复串长度的代码的表中的256的中间整数表示)用于EOB符号。当HTT使用DHT指定时,在DHT中指定用于EOB符号的霍夫曼代码。当BCC位为零时,EOB符号不存储到第一操作数位

置。BCC位在指令的执行期间不修改。

[0188] 块报头最终 (BHF) 379: 当指定DFLTCC-CMPR函数并且BCF377为0或NT374为1时, 应用参数块的字节16的位7; 否则不适用BHF。当可应用且为1时, 在将块标头存储到第一操作数位置之前, 将块标头 (BFINAL) 的第一位设定为1。当可应用且为零时, 在将块标头存储到第一操作数位置之前, 将块标头的第一位 (BFINAL) 设定为零。BHF位在指令的执行期间不被修改。

[0189] DHT生成控制 (DHTGC) 364: 当指定DFLTCC-CMPR功能时, DHTGC不适用于操作。在执行指令期间不修改DHTGC。

[0190] 子字节边界 (SBB) 381: 参数块的字节18的位5-7含有无符号二进制整数, 其指定经压缩数据流的字节内的经处理位与未经处理位之间的边界。所引用的流的字节是当操作结束时的最后所引用的字节 (意味着最右边的字节), 并且是当操作开始或恢复时要引用的第一字节 (意味着最左边的字节)。当指定DFLTCC-CMPR函数时, SBB应用于由第一操作数地址指定的字节。当指定DFLTCC-XPND函数时, SBB应用于第二操作数地址指定的字节。SBB指定已处理的最右边的位的数目。SBB是操作的输入和操作的输出。

[0191] 图4中描绘了当SBB具有011二进制的值时的压缩数据流的一个示例。在400处描绘了在操作结束之后已经被处理的数据; 并且在402处描绘了在操作开始之前要处理的数据。

[0192] 进一步地, 图5A-5C提供了展示SBB如何应用于DFLTCC-CMPR功能的实例。例如, 在图5A中描绘了在执行DFLTCC-CMPR功能之前和之后SBB如何应用的一个实例。在图5B-5C中描绘了其他实例。当NT374是1时, SBB381被视为等于000二进制。

[0193] 返回图3L, 描述参数块370的附加字段:

[0194] 操作结束补充代码 (OESC) 365。

[0195] 不完整功能状态 (IFS) 383: 当某些操作结束时, 参数块的字节21的位4-7含有状态信息。在解压缩操作结束时, IFS如下传递关于第二操作数的信息, 在一个示例中:

IFS (二进制)	含义
0000	在解码具有等于一的 BFINAL 的块的最后元素之后, 操作结束。
1000	在对 BTYPE 等于 00 二进制且 BFINAL 等于 0 的块的除了最后元素之外的元素进行解码之后, 操作结束。
1001	在对 BTYPE 等于 00 二进制并且 BFINAL 等于 1 的块的除了最后元素之外的元素进行解码之后, 操作结束。
1010	在对 BTYPE 等于 01 二进制且 BFINAL 等于 0 的块的除了最后元素之外的元素进行解码之后, 操作结束。
[0196] 1011	在解码具有等于 01 二进制的 BTYPE 和等于一的 BFINAL 的块的除了最后元素之外的元素之后结束操作。
1100	在对 BTYPE 等于 10 二进制且 BFINAL 等于 0 的块的除了最后元素之外的元素进行解码之后, 操作结束。
1101	在对 BTYPE 等于 10 二进制且 BFINAL 等于 1 的块的除了最后元素之外的元素进行解码之后, 操作结束。
1110	操作结束于块边界, BFINAL 等于 1 的块的最后元素尚未被解码, 并且后续块的块头第一元素尚未被处理。

[0197] 在一个实施例中, 解压缩操作可以以等于0000二进制的IFS结束, 并且不满足正常完成。在这样的情况下, 以设置的条件码1或3结束操作。

[0198] 当压缩操作结束时,IFS字段是未定义的,但是可以被修改。

[0199] IFS不是对操作的输入。

[0200] 不完整函数长度 (IFL) 384: 当某些操作结束时,参数块的字节22-23包含长度信息。对于解压缩操作,IFL应用于第二操作数。当解压缩操作在对具有等于00二进制的BTTYPE的块中的一些但不是全部进行解码之后结束时,IFL包含指定第二操作数中的块的尚未被处理的字节数的无符号二进制整数。字节22至23包含例如大端字节次序的IFL,不同于具有等于00二进制的BTTYPE的块的LEN字段,其例如小端字节次序 (little-endian byte order)。

[0201] 当解压缩操作在解码具有等于00二进制的BTTYPE和等于1的BFINAL的完整块之后结束时,将零存储到IFL字段。当解压缩操作在解码具有非零BTTYPE的块的一些但不是全部之后结束,或者在块边界处结束时,IFL字段是未定义的,但是可以被修改。

[0202] 当压缩操作结束时,IFL字段是未定义的,但是可以被修改。

[0203] IFL不是对操作的输入。

[0204] 历史长度 (HL) 385: 参数块的字节44-45包含无符号二进制整数,其指定在操作期间可以参考的历史缓冲器中的历史的字节数。HL应用于在线和循环历史缓冲器。当新任务 (NT) 等于1时,没有历史应用于操作的开始,并且历史长度被视为零作为对操作的输入。

[0205] 当历史长度大于例如32,768且NT等于零时,识别一般操作数数据异常。

[0206] 历史长度在压缩和解压缩操作期间被修改。当原始HL与在操作期间处理的未压缩数据字节的数量的和小于或等于例如32,768时,更新的HL等于原始HL与在操作期间处理的未压缩数据字节的数量的和;否则,更新的HL等于32,768的值。

[0207] 历史偏移 (H0) 386: 参数块的从字节46的位1开始到字节47的位7的十五个位含有无符号二进制整数,其在历史缓冲器类型为循环时指定第三操作数中的偏移。R3的内容与历史偏移的总和指定循环历史缓冲器内的历史的第一字节的位置,其是缓冲器中的未压缩数据的最近最少处理的字节。当历史缓冲器类型是循环时,历史偏移是对操作的输入并且在操作结束时被更新。当原始HL和在操作期间处理的未压缩数据字节的数量的和小于或等于例如32,768时,更新的H0等于原始H0;否则,更新的H0等于原始H0、原始HL和在操作期间处理的未压缩数据字节的数量的总和,模32,768。

[0208] 当历史缓冲类型在线时,参数块的H0字段是未定义的,但可被修改。

[0209] 校验值387: 参数块的字节48-51包含校验值。作为操作的一部分,生成检查值。检查值应用于未压缩数据操作数。即,检查值应用于DFLTCC-CMPR函数的第二操作数并且应用于DFLTCC-XPND函数的第一操作数。当CVT字节375是零时,产生例如32字节循环冗余校验校验值 (CRC-32)。当CVT位为1时,产生例如32位Adler校验和校验值 (Adler-32)。

[0210] 生成校验值的输入是例如4字节基数并且在操作期间处理未压缩数据。基本输入提供用于计算一组压缩数据块的单个和一致的校验值的手段,而不管DFLTCC指令被执行以处理完整的一组压缩数据块的次数。当NT位为0时,校验值字段中的原始值用于生成校验值的基本输入。

[0211] 在一个示例中,当生成Adler-32校验值时,以下适用:

[0212] • 当NT位为1时,值1用于4字节基本输入。

[0213] • 在Adler-32校验值生成中定义的和是模65,521。

[0214] • 结果以大字节序存储到校验值字段。即,校验值的最高有效字节位于字节48中,并且校验值的最低有效字节位于字节51中。

[0215] 在一个实施例中,当CRC-32校验值被生成时,以下适用:

[0216] • 当NT位为1时,值0用于4字节基本输入。

[0217] • 在生成CRC-32校验值时用作除数的多项式为 $x^{32}+x^{26}+x^{23}+x^{22}+x^{16}+x^{12}+x^{11}+x^{10}+x^8+x^7+x^5+x^4+x^2+x^1+x^0$,表示为104C11DB7十六进制。在此表示中,最左边位对应于最高有效位。

[0218] • 生成校验值的第一阶段和最后阶段分别是在存储结果之前计算基本输入的一的补码和计算结果的一的补码。

[0219] • 结果以小字节序存储到校验值字段。即,校验值的最低有效字节位于字节48中,并且校验值的最高有效字节位于字节51中。

[0220] 在一个示例中,当操作结束于条件码0设置时,检查值仅对程序有意义;否则,校验值只是中间结果,只对恢复操作有意义。当指定DFLTCC-CMPR函数并且操作结束于条件码1、2或3时,将由第二操作数地址指定的字节的左边的一些字节设定为可不包含在所得检查值的计算中。当指定DFLTCC-XPND函数并且操作结束于条件码1、2或3集合时,尚未存储在由第一操作数地址指定的字节右侧的一些结果字节可能已经包括在结果校验值的计算中。

[0221] 块结束符号 (EOBS) 388: 参数块的从字节52的字节0开始到字节53的字节6的十五个字节包含块结束 (EOB) 符号。参数块的块结束长度 (EOBL) 字段389指定EOBS字段中的EOB符号的长度。EOB符号在EOBS字段中被左对齐。EOBS字段中未被EOB符号占用的字节存储为零。EOBS字段是当压缩数据时的操作的输出,而不管应用哪种类型的霍夫曼表。EOBS字段不用作操作的输入。

[0222] 字节52的位0包含EOB符号的最高有效位。当EOB符号的长度为7字节时,字节52的字节6包含EOB符号的最低有效字节。当EOB符号的长度为15字节时,字节53的字节6包含EOB符号的最低有效字节。

[0223] 对于使用FHT的块,EOB符号是由DEFLATE标准定义的0000000二进制。对于使用DHT的块,EOB符号由DHT定义。传送EOB符号以便为程序提供关闭块的能力。

[0224] 当指定DFLTCC-XPND功能时,EOBS字段是未定义的,但是可以被修改。

[0225] 块结束长度 (EOBL) 389: 参数块的字节54的位0-3包含指定参数块的EOBS字段388中的块结束 (EOB) 符号的长度的无符号二进制整数。该长度规定EOB符号在EOBS字段中占用的字节数。EOBL字段是当压缩数据时的操作的输出,而不管应用哪种类型的霍夫曼表。EOBL字段不用作操作的输入。

[0226] 当指定DFLTCC-XPND函数时,EOBL字段是未定义的,但是可以被修改。

[0227] 压缩动态霍夫曼表长度 (CDHTL) 366。

[0228] 压缩动态霍夫曼表 (CDHT) 367: 当指定DFLTCC-CMPR功能并且HTT为1时,DHT的压缩表示是对操作的输入。CDHT字段不被DFLTCC-CMPR功能修改。

[0229] 连续状态缓冲器 (CSB) 392: 当条件使得值1被存储在CF字段373中时,内部状态数据被存储到参数块的字节384-1535;否则,参数块的字节384-1535是未定义的,并且可以被修改。所存储的内部状态数据是依赖于模型的,并且可随后用于恢复操作。预期但不要求程序初始化继续状态缓冲器以包含例如全零。在指令以非零条件码集合结束之后,且在为了

恢复操作的目的是而重新执行指令之前,程序不应修改继续状态缓冲器;否则结果是不可预测的。

[0230] 尽管以上参考参数块370描述了不同字段,但是在其他实施例中可以包括附加的、更少的和/或其他的字段。

[0231] 下文相对于压缩数据描述压缩操作的一个实例。

[0232] DFLTCC-CCMPR函数的正常完成发生在整个第二操作数被压缩并存储到第一操作数位置时。当操作由于正常完成而结束时,在一个实例中,发生以下情形: • 模型相关值被存储到参数块的模型版本号 (MVN) 字段363。

[0233] • 参数块的继续标记 (CF) 字段373被设置为零。

[0234] • 参数块的子字节边界 (SBB) 字段381被更新。

[0235] • 参数块的块结束长度 (EOBL) 389和块结束符号 (EOBS) 388字段被更新。

[0236] • 参数块的历史长度 (HL) 字段385被更新。

[0237] • 参数块的历史偏移 (HO) 字段386在适用时被更新。

[0238] • 参数块的操作结束补充代码 (OESC) 字段365被设置为零。

[0239] • 参数块的校验值字段387被更新。

[0240] • 通用寄存器R1中的地址递增包括处理位0的第一操作数的处理的字节数,并且通用寄存器R1+1中的长度递减相同的数。包括处理位0的第一操作数的处理的字节数是由整数除法产生的整数商,其中除数是所处理的输出位的数量与SBB的原始值的和,除数是值8。

[0241] • 通用寄存器R2中的地址递增所处理的源字节的数量,并且通用寄存器R2+1中的长度递减相同的数量。

[0242] • 设置条件码0。

[0243] 地址和长度的形成和更新取决于寻址模式。

[0244] 当发生正常完成时,在操作结束之后不定义参数块的CSB字段392。

[0245] 在一个例子中,当CPU确定的字节数已被处理时,操作结束且发生以下情形:

[0246] • 参数块中的继续标记 (CF) 位373被设置为1。

[0247] • 参数块中的连续状态缓冲器 (CSB) 字段392被更新。

[0248] • 参数块的子字节边界 (SBB) 字段381被更新。

[0249] • 参数块的历史长度 (HL) 字段385被更新。

[0250] • 当适用时,更新参数块的历史偏移 (HO) 字段386。

[0251] • 参数块的校验值字段387被更新。

[0252] • 依赖于模型的值被存储到参数块的模型版本号 (MVN) 字段363。

[0253] • 参数块的块结束长度 (EOBL) 389和块结束符号 (EOBS) 388字段被更新。

[0254] • 参数块的操作结束补充代码 (OESC) 字段365被设置为零。

[0255] • 通用寄存器R1中的地址递增包括处理位0的第一操作数的处理的字节数,并且通用寄存器R1+1中的长度递减相同的数。包括处理位0的第一操作数的处理的字节数是由整数除法产生的整数商,其中除数是所处理的输出位的数量与SBB的原始值的和,除数是值8。

[0256] • 通用寄存器R2中的地址递增所处理的源字节的数量,并且通用寄存器R2+1中的

长度递减相同的数量。

[0257] • 设置条件码3。

[0258] 地址和长度的形成和更新取决于寻址模式。

[0259] CPU确定的字节数取决于模型,且可为每次执行指令时不同的数目。

[0260] 在指令以条件码3设置结束之后,期望程序不修改指令的任何输入或输出规范并且分支回来重新执行指令以恢复操作。

[0261] 在某些情况下,尽管以条件码3设置结束指令,但不更新参数块和通用寄存器。这些情形可在CPU在执行DEFLATE转换调用指令的同时执行静默操作或CPU重试时发生。在这些情况下,所处理的CPU确定的字节数为零,数据可能已存储到第一操作数位置,数据可能已存储到第三操作数位置(在适用时),且已设定对应改变位。

[0262] 在一个实例中,当应用以下条件中的任一者时,第一操作数长度不足以完成操作:

[0263] • 如由通用寄存器R1+1的内容指定的第一操作数长度在指令的执行开始时为零。

[0264] • 第一操作数长度在指令的执行期间变为等于零,并且不发生正常完成。

[0265] 在一个实例中,当通用寄存器R1+1的内容为零时,第一操作数长度为零,而不管参数块的NT字段和SBB字段中的值如何。

[0266] 在一个实施例中,当第一操作数长度在指令的执行期间变为等于零时,操作结束并且发生以下情形:

[0267] • 参数块中的继续标记(CF)位373被设置为1。

[0268] • 参数块中的连续状态缓冲器(CSB)字段392被更新。

[0269] • 参数块的子字节边界(SBB)字段381被更新。

[0270] • 参数块的历史长度(HL)字段385被更新。

[0271] • 当适用时,更新参数块的历史偏移(H0)字段386。

[0272] • 参数块的校验值字段387被更新。

[0273] • 依赖于模型的值被存储到参数块的模型版本号(MVN)字段363。

[0274] • 参数块的块结束长度(EOBL)389和块结束符号(EOBS)388字段被更新。

[0275] • 参数块的操作结束补充代码(OESC)字段365被设置为零。

[0276] • 通用寄存器R1中的地址递增包括处理位0的第一操作数的处理的字节数,并且通用寄存器R1+1中的长度递减相同的数。包括处理位0的第一操作数的处理的字节数是由整数除法产生的整数商,其中除数是所处理的输出位的数量与SBB的原始值的和,除数是值8。

[0277] • 通用寄存器R2中的地址递增所处理的源字节的数量,并且通用寄存器R2+1中的长度递减相同的数量。

[0278] • 设置条件码1。

[0279] 地址和长度的形成和更新取决于寻址模式。

[0280] 在一个实施例中,当在指令的执行开始时第一操作数长度是零时,操作结束并且发生以下操作:

[0281] • 设置条件码1。

[0282] 在指令以条件码1设置结束之后,预期程序修改第一操作数长度、第一操作数地址或两者,并且重新执行指令以恢复操作。

[0283] 当可应用时,PER储存变更事件被识别如下:

[0284] • 存储到参数块,如下所述。

[0285] • 存储到该第一操作数位置。

[0286] • 存储第三操作数位置,其例如当历史缓冲器类型(HBT)为一(循环)时发生。

[0287] 当整个参数块与PER存储区域指定重叠时,当可应用时,对于参数块识别PER储存变更事件。当参数块的仅一部分与PER存储区域指定重叠时,发生以下中的哪一个是依赖于模型的:

[0288] • PER储存变更事件在适用于参数块时被识别。

[0289] • 当可应用时,PER储存变更事件被识别用于储存的参数块的部分。

[0290] 当可应用时,对于参数块,在HBT为1(循环)时,识别PER零地址检测事件,第一操作数位置、第二操作数位置和第三操作数位置。

[0291] 条件码2不适用于DFLTC-CCMPR函数。

[0292] 当指令以设置条件码1或3结束时,可完全地或仅部分地处理从第二操作数位置引用的输入数据。当输入数据仅被部分地处理时,导致参数块的第一操作数位置、第一操作数地址、第一操作数长度和SBB字段不表示与更新的第二操作数地址和长度一致的状态。在这些情况下,部分处理的数据和内部状态信息可被放置在参数块的CSB字段中。部分处理数据的量取决于在操作结束时存在的条件和模型。尽管一些数据可能仅被部分地处理,但是存储在由经更新的第一操作数地址指定的位置的左侧的结果是完整的,并且在操作恢复时将不被修改。此外,预期程序随后重新执行指令以恢复操作,此时在恢复操作之前参考CSB字段的内容。当指令以设置条件码0结束时,所有数据被完全处理,并且与输入和输出数据相关联的所有结果表示一致状态。

[0293] 在指令以非零条件码集结束之后,且在为了恢复操作的目的而重新执行指令之前,程序不应修改参数块的任何字段;否则结果是不可预测的。

[0294] 函数代码4:DFLTCC-XPND(Expand)(扩展)

[0295] 当指定DFLTCC-XPND功能时,执行解压缩操作。该操作包括将来自第二操作数位置的压缩数据符号解码成未压缩数据,该未压缩数据被存储到第一操作数位置。

[0296] 在一个示例中,DFLTCC-XPND函数使用参数块,以上关于图3K-3L描述了该参数块的示例。

[0297] 下文相对于解压缩数据描述DFLTCC-XPND操作的一个实例。

[0298] 当第二操作数中的数据集合的最终块的所有元素被解码并且所有未压缩数据被存储到第一操作数位置时,正常完成发生。当块头的BFINAL位为1时,识别数据集合的最后块。当操作由于正常完成而结束时,在一个实施例中发生以下情形:

[0299] • 模型相关值被存储到参数块的模型版本号(MVN)字段363。

[0300] • 参数块的继续标记(CF)字段373被设置为零。

[0301] • 参数块的子字节边界(SBB)字段381被更新。

[0302] • 参数块的历史长度(HL)字段385被更新。

[0303] • 当适用时,更新参数块的历史偏移(H0)字段386。

[0304] • 参数块的压缩动态霍夫曼表(CDHT)367和压缩动态霍夫曼表长度(CDHTL)字段366被设置为零。

- [0305] • 参数块的操作结束补充代码 (OESC) 字段365被设置为零。
- [0306] • 参数块的校验值字段387被更新。
- [0307] • 通用寄存器R1中的地址递增存储在第一操作数位置处的字节的数目,且通用寄存器R1+1中的长度递减相同数目。
- [0308] • 通用寄存器R2中的地址递增包含处理位0的第二操作数的处理的字节的数目,且通用寄存器R2+1中的长度递减相同数目。包括处理位0的第二操作数的处理的字节的数量是从整数除法得到的整数商,其中除数是所处理的输入位的数量和SBB的原始值的总和,除数是值8。
- [0309] • 设置条件码0。
- [0310] 地址和长度的形成和更新取决于寻址模式。
- [0311] 当发生正常完成时,在操作结束之后不定义参数块的CSB字段392。
- [0312] 当CPU确定数量的字节已经被处理时,操作结束并且发生以下情形,在一个实施例中:
 - [0313] • 参数块中的继续标记 (CF) 位373被设置为1。
 - [0314] • 参数块中的连续状态缓冲器 (CSB) 字段392被更新。
 - [0315] • 参数块的子字节边界 (SBB) 字段381被更新。
 - [0316] • 参数块的压缩动态霍夫曼表 (CDHT) 367和压缩动态霍夫曼表长度 (CDHTL) 366字段被更新。当在处理具有10二进制的BTYPE值的块的同时发生部分完成时,将表示该表不需要的CDHT字段的字节存储为零。当在处理具有00或01二进制的BTYPE值的块时发生部分完成时,将零存储到CDHT和CDHTL字段。
- [0317] • 参数块的历史长度 (HL) 字段385被更新。
- [0318] • 当适用时,更新参数块的历史偏移 (HO) 字段386。
- [0319] • 参数块的校验值字段387被更新。
- [0320] • 模型相关值被存储到参数块的模型版本号 (MVN) 字段363。
- [0321] • 参数块的操作结束补充代码 (OESC) 字段365被设置为零。
- [0322] • 参数块的不完整功能状态 (IFS) 字段383被更新。
- [0323] • 当可应用时,更新参数块的不完整函数长度 (IFL) 字段384。
- [0324] • 通用寄存器R1中的地址递增存储在第一操作数位置处的字节的数目,且通用寄存器R1+1中的长度递减相同数目。
- [0325] • 通用寄存器R2中的地址递增包含处理位0的第二操作数的处理的字节的数目,且通用寄存器R2+1中的长度递减相同数目。包括处理位0的第二操作数的处理的字节的数量是从整数除法得到的整数商,其中除数是所处理的输入位的数量和SBB的原始值的总和,除数是值8。
- [0326] • 设定条件码3。
- [0327] 地址和长度的形成和更新取决于寻址模式。
- [0328] CPU确定的字节数取决于模型,且可为每次执行指令时不同的数目。
- [0329] 在指令以条件码3设置结束之后,期望程序不修改指令的任何输入或输出规范并且分支回来重新执行指令以恢复操作。
- [0330] 在某些情况下,尽管以条件码3设置结束指令,但不更新参数块和通用寄存器。这

些情形可在CPU在执行DEFLATE转换调用指令的同时执行静默操作或CPU重试时发生。在这些情况下,所处理的CPU确定的字节数为零,数据可能已存储到第一操作数位置,数据可能已存储到第三操作数位置(在适用时),且已设定对应改变位。

[0331] 第二操作数长度不足以在应用以下操作时完成操作,例如:

[0332] 在操作期间,BFINAL等于1的经压缩数据块的最后元素尚未被解码,以及由第二操作数长度和SBB指定的第二操作数中的位数,小于要解码的下一个元素的位数,并且来自对来自第二操作数位置的数据进行解码的所有结果已经被放置在第一操作数位置处。

[0333] 在一个实施例中,当第二操作数长度不足以完成操作时,操作已经部分完成,操作结束,并且发生以下情况:

[0334] • 参数块中的继续标记(CF)位373被设置为1。

[0335] • 参数块中的连续状态缓冲器(CSB)字段392被更新。

[0336] • 参数块的子字节边界(SBB)字段381被更新。

[0337] • 参数块的压缩动态霍夫曼表(CDHT)367和压缩动态霍夫曼表长度(CDHTL)字段366被更新。当在处理具有10二进制的BTYP值值的块的同时发生部分完成时,将表示该表不需要的CDHT字段的字节存储为零。当在处理具有00或01二进制的BTYP值值的块时发生部分完成时,将零存储到CDHT和CDHTL字段。

[0338] • 参数块的历史长度(HL)字段385被更新。

[0339] • 当适用时,更新参数块的历史偏移(H0)字段386。

[0340] • 参数块的校验值字段387被更新。

[0341] • 模型相关值被存储到参数块的模型版本号(MVN)字段363。

[0342] • 参数块的操作结束补充代码(OESC)字段365被设置为零。

[0343] • 参数块的不完整功能状态(IFS)字段383被更新。

[0344] • 当可应用时,更新参数块的不完整函数长度(IFL)字段384。

[0345] • 通用寄存器R1中的地址递增存储在第一操作数位置处的字节的数目,且通用寄存器R1+1中的长度递减相同数目。

[0346] • 通用寄存器R2中的地址递增包含处理位0的第二操作数的处理的字节的数目,且通用寄存器R2+1中的长度递减相同数目。包括处理位0的第二操作数的处理的字节的数量是从整数除法得到的整数商,其中除数是所处理的输入位的数量和SBB的原始值的总和,除数是值8。

[0347] • 设置条件码2。

[0348] 地址和长度的形成和更新取决于寻址模式。

[0349] 在指令结束于条件码2设置之后,预期程序修改第二操作数长度、第二操作数地址或两者,并且重新执行指令以恢复操作。

[0350] 当应用以下操作时,第一操作数长度不足以完成操作,例如:

[0351] • 来自解码来自第二操作数位置的数据的结果由于第一操作数长度等于零而不能被放置在第一操作数位置处。

[0352] 当第一操作数长度不足以完成操作时,操作已部分完成,操作结束,且在一个实施例中发生以下情形:

[0353] • 参数块中的继续标记(CF)位373被设置为1。

- [0354] • 参数块中的连续状态缓冲器 (CSB) 字段392被更新。
- [0355] • 参数块的子字节边界 (SBB) 字段381被更新。
- [0356] • 参数块的压缩的动态霍夫曼表 (CDHT) 367和压缩的动态霍夫曼表长度 (CDHTL) 字段366被更新。当在处理具有10二进制的BTYP E值的块的同时发生部分完成时,将表示该表不需要的CDHT字段的字节存储为零。当在处理具有00或01二进制的BTYP E值的块时发生部分完成时,将零存储到CDHT和CDHTL字段。
- [0357] • 参数块的历史长度 (HL) 字段385被更新。
- [0358] • 当适用时,更新参数块的历史偏移 (H0) 字段386。
- [0359] • 参数块的校验值字段387被更新。
- [0360] • 模型相关值被存储到参数块的模型版本号 (MVN) 字段363。
- [0361] • 参数块的操作结束补充代码 (OESC) 字段365被设置为零。
- [0362] • 参数块的不完整功能状态 (IFS) 字段383被更新。
- [0363] • 当可应用时,更新参数块的不完整函数长度 (IFL) 字段384。
- [0364] • 通用寄存器R1中的地址递增存储在第一操作数位置处的字节的数目,且通用寄存器R1+1中的长度递减相同数目。
- [0365] • 通用寄存器R2中的地址递增包含处理位0的第二操作数的处理的字节的数目,且通用寄存器R2+1中的长度递减相同数目。包括处理位0的第二操作数的处理的字节的数量是从整数除法得到的整数商,其中除数是所处理的输入位的数量和SBB的原始值的总和,除数是值8。
- [0366] • 设定条件码1。
- [0367] 地址和长度的形成和更新取决于寻址模式。
- [0368] 在指令以条件码1设置结束之后,预期程序修改第一操作数长度、第一操作数地址或两者,并且重新执行指令以恢复操作。
- [0369] 当可应用时,PER储存变更事件被识别如下:
- [0370] • 如本文所述,存储到参数块。
- [0371] • 存储到该第一操作数位置。
- [0372] • 存储第三操作数位置,其例如当历史缓冲器类型 (HBT) 为一 (循环) 时发生。
- [0373] 在一个实例中,当整个参数块与PER存储区域指定重叠时,当可应用时,针对参数块识别PER储存变更事件。在一个实施例中,当参数块的仅一部分与PER存储区域指定重叠时,与模型无关地发生以下中的哪一个:
- [0374] • PER储存变更事件在适用于参数块时被识别。
- [0375] • 当可应用时,PER储存变更事件被识别用于储存的参数块的部分。
- [0376] 当可应用时,对于参数块,在HBT为1 (循环) 时,识别PER零地址检测事件,第一操作数位置、第二操作数位置和第三操作数位置。
- [0377] 当指令结束于条件码1、2或3设置时,可完全地或仅部分地处理从第二操作数位置引用的输入数据。当输入数据仅被部分地处理时,产生第一操作数位置,第一操作数地址,第一操作数长度、所述参数块的SBB字段、所述参数块的校验值字段,参数块的HL字段、参数块的IFS字段以及当适用时,参数块的第三操作数位置和H0字段不表示与更新的第二操作数地址和长度一致的状态。在这些情况下,部分处理的数据和内部状态信息可被放置在参

数块的CSB字段中。部分处理数据的量取决于在操作结束时存在的条件和模型。尽管一些数据可能仅被部分地处理,但是存储在由经更新的第一操作数地址指定的位置的左侧的结果是完整的,并且在操作恢复时将不被修改。此外,预期程序随后重新执行指令以恢复操作,此时在恢复操作之前参考CSB字段的内容。当操作结束于条件码0设置时,所有数据被完全处理,并且与输入和输出数据相关联的所有结果表示一致状态。

[0378] 在指令以非零条件码集结束之后,且在为了恢复操作的目的而重新执行指令之前,程序不应修改参数块的任何字段;否则结果是不可预测的。

[0379] 压缩的数据块

[0380] 在一个示例中,例如从左到右处理存储装置中的压缩数据块的字节。压缩的数据块可以或可以不在字节边界上开始或结束。压缩的数据块例如是字节流。一次一位地将块的元素加载到存储器中。位流例如在存储装置的每一字节内从右向左加载且按字节次序从(例如)从左向右加载。当元素是霍夫曼码时,按从例如元素的最高有效位到最低有效位的顺序存储位。当元素不是霍夫曼码时,按从例如元素的最低有效位到最高有效位的顺序存储位。

[0381] 图6示出了具有块类型00二进制的块600的示例,其不包含压缩的数据符号。在一个实施例中,以下内容适用于该实例:

[0382] • 经压缩的数据块600由字节流602组成,该字节流602以字节0的位4(标识为b0)开始,并且以字节7的位0(标识为b60)结束。

[0383] • 位流中遇到的第一元素是字节0的位4中的BFINAL(块标头最终位)。

[0384] • 在位流中遇到的第二元素是字节0的位2-3中的BTYP(块类型)。在该示例中,BTYP是00二进制的。

[0385] • 当BTYP为00二进制(其在此实例中为字节0的位0-1)时,忽略BTYP左边及字节边界右边的位。

[0386] • 位流中遇到的第三元素是LEN字段的最低有效字节(LSB),其之后是LEN字段的最高有效字节(MSB)。LEN字段指定具有文字量数据的块中的字节数。例如,原始数据是未压缩的数据。具有文字量数据的字节跟随位流中的NLEN字段。NLEN是LEN的互补序列。在一个示例中,字节1-2以小字节次序包含LEN字段。

[0387] • 在LEN字段之后的字节流中遇到的元素分别是NLEN字段的最低有效字节,之后是NLEN字段的最高有效字节。字节3到4以小字节次序含有NLEN字段。NLEN字段是LEN字段的1的补码。

[0388] • 在NLEN字段之后的字节流中遇到的元素是未压缩的数据,被标识为文字量字节。字节5到7包含未压缩数据,其与用以产生此块的源数据未改变。

[0389] • 在该块中包含的元素都不是霍夫曼码。此块中的每一元素按从元素的最低有效位到最高有效位的次序存储到位流次序,如由DEFLATE标准定义。由于LEN、NLEN和文字量元素各自是在字节边界上对齐的整数数量的字节,因此这些元素可以被处理为字节单位,并且不一定处理为位单位。

[0390] 图7示出了具有块类型01二进制的块700的示例,该块700包含使用固定的霍夫曼表(FHT)生成的压缩的数据符号。在一个实施例中,以下内容适用于该实例:

[0391] • 压缩数据块700由字节流702组成,字节流702以字节0的位4开始,标识为b0,并

且以字节11的位3结束,标识为b89。

[0392] • 位流中遇到的第一元素是字节0的位4中的BFINAL。

[0393] • 在位流中遇到的第二元素是字节0的位2-3中的BTYPE。在该示例中,BTYPE是01二进制的。

[0394] • 固定霍夫曼表(FHT)不是块的组件。

[0395] • 位流中遇到的第三元素是第一经压缩数据符号,其在字节0的位1中开始。在一个实例中,经压缩数据符号由在位流中遇到的以列出所述子元素的次序的次序遇到的以下子元素组成:

[0396] 1. 可变长度的霍夫曼码。代码的最高有效位指定代码的长度。在字节流中遇到该码,以码的最高有效位开始并且以码的最低有效位结束。当代码表示文字量值或块结束符号时,代码是经压缩的数据符号的唯一子元素。当代码表示指向历史缓冲器的指针的长度时,代码之后是压缩数据符号的后续子元素。

[0397] 2. 当适用时,如由DEFLATE标准指定的,额外长度位可以遵循表示指针长度的霍夫曼代码。在位流中遇到额外长度位,所述额外长度位以额外长度位的最低有效位开始且以最高有效位结束。

[0398] 3. 位流中遇到的下一子元素为到历史缓冲器的指针的5位距离码。在字节流中遇到距离码,该距离码以例如该码的最高有效位开始并以该距离码的最低有效位结束。

[0399] 4. 在适用时,如由DEFLATE标准指定的,额外的距离位可遵循距离代码。在位流中遇到额外距离位,所述额外距离位从所述额外距离位的最低有效位开始且以所述额外距离位的最高有效位结束。

[0400] • 作为实例,字节0的位0-1、字节1到9的所有位和字节10的位2-7含有经压缩数据符号的位。

[0401] • 在位流中遇到的最后元素是包含单个子元素的压缩的数据符号,其是表示块结束(EOB)符号的霍夫曼码。具有BTYPE 01二进制的块的EOB符号是0000000二进制。在此实例中,字节10的位1含有EOB符号的最高有效位,且字节11的位3含有EOB符号的最低有效位。

[0402] • 字节11的位3包含位流的最后一位,其为经压缩数据块的最后一位。

[0403] • 图8示出了具有块类型10二进制的块800的示例,该块800包含使用动态霍夫曼表(DHT)生成的压缩的数据符号。在一个实施例,以下内容适用于该实例:

[0404] • 压缩的数据块800由字节流802组成,字节流802以字节0的位4开始,标识为b0,并且以字节11的位3结束,标识为b89。

[0405] • 位流中遇到的第一元素是字节0的位4中的BFINAL。

[0406] • 在位流中遇到的第二元素是字节0的位2-3中的BTYPE。在该示例中,BTYPE是10二进制的。

[0407] • 在位流中遇到的第三元素是动态霍夫曼表(DHT)的压缩表示,其在字节0的位1中开始。DHT的压缩表示包括以下子元素,在一个示例中,这些子元素在位流中以列出它们的顺序遇到:

[0408] 1. HLIT:5位HLIT子元素和257的总和指定表示文字量字节、EOB符号和重复串长度的霍夫曼码的数量。HLIT的有效值范围从例如0至29。在位流中遇到HLIT位,其以HLIT子元件的最低有效位开始且以最高有效位结束。在此实例中,字节0的位1(识别为b3)为HLIT子

元件的最低有效位。

[0409] 2.HDIST:5位HDIST子元素和1的总和指定表示复制串指针距离的霍夫曼码的数量。HDIST的有效值范围从例如0至29。在从HDIST子元件的最低有效位开始且以最高有效位结束的位流中遇到HDIST位。

[0410] 3.HCLEN:4位HCLEN子元素和4的总和指定表示代码长度的霍夫曼代码的数量。HCLEN的有效值例如为0至15。在位流中遇到HCLEN位,其以HCLEN子元件的最低有效位开始且以最高有效位结束。

[0411] 4.代码序列,所述代码序列指定针对经压缩的DHT定义的代码长度中的每个代码长度的位长度。码的数目等于HCLEN与4的总和。每个码是3字节。

[0412] 5.代码序列,所述代码序列为由字面字节、EOB符号和重复串长度组成的所述集中的所述元素中的每个元素指定代码长度。指定的码长的数目等于HLIT与257的总和。

[0413] 当用于字面字节集合、EOB符号和重复串长度的最后代码长度(CL)是16、17或18并且CL之后的额外位指定针对比针对集合定义的元素更多的元素重复CL时,代码长度还适用于重复串指针距离集合。为字面字节集合、EOB符号和重复串长度指定代码长度的代码序列、之后为重复串指针距离指定代码长度的代码序列是两个集合的连续序列。

[0414] 6.代码序列,其指定由重复串指针距离组成的所述集合的所述元素中的每一者的代码长度。指定的码长的数量等于HDIST和1的和。

[0415] • 位流中遇到的第四元素是第一经压缩数据符号。在一个实施例中,压缩的数据符号由以下子元素组成,这些子元素在位流中以列出它们的顺序遇到:

[0416] 1.可变长度的霍夫曼码。代码的最高有效位指定代码的长度。在字节流中遇到该码,以码的最高有效位开始并且以码的最低有效位结束。当代码表示文字量值或块结束符号时,代码是经压缩的数据符号的唯一子元素。当代码表示指向历史缓冲器的指针的长度时,代码之后是压缩数据符号的后续子元素。

[0417] 2.当适用时,如由DEFLATE标准指定的,额外长度位可以遵循表示指针长度的霍夫曼代码。在字节流中遇到额外长度字节,其以例如最低有效字节开始并以额外长度字节的最高有效字节结束。

[0418] 3.位流中遇到的下一子元素为到历史缓冲器的指针的5位距离码。在字节流中遇到距离码,该距离码以例如该码的最高有效位开始并以该距离码的最低有效位结束。

[0419] 4.在适用时,如由DEFLATE标准指定的,额外的距离位可遵循距离代码。在位流中遇到额外距离位,其以例如额外距离位的最低有效位开始且以最高有效位结束。

[0420] • 位流中遇到的后续位(高达且包含(例如)字节10的位5)含有经压缩数据符号的位。

[0421] • 在位流中遇到的最后元素是包含单个子元素的压缩的数据符号,其是表示块结束(EOB)符号的霍夫曼码。在此实例中,字节10的位4含有EOB符号的最高有效位,且字节11的位3含有EOB符号的最低有效位。

[0422] • 字节11的位3包含位流的最后一位,其为经压缩数据块的最后一位。

[0423] 在以上对不同块类型的描述中,指定某些恒定值以及特定位、字节、方向等。这些仅是实例。在其他实施例中,可以指定其他恒定值、位、字节、方向等。

[0424] 处理压缩的数据集

[0425] 提供处理经压缩数据集合以说明DEFLATE转换调用指令的实例使用和增强参数块的不同字段的描述的实例。示例没有描述所有可能的场景、要求和能力,但是示出了不同场景、要求和/或能力。示例和描述例如应用于存储装置中的压缩数据集合,其示例在图9中示出。如图所示,压缩数据集合900包括多个压缩数据块902,并且数据集合900的开始由压缩数据集合开始地址(CDSBA)904指示。

[0426] 对于在此描述的实例,在一个实施例中,旨在用于处理压缩数据集合的程序考虑以下各项:

[0427] • 单个参数块可由DEFLATE转换调用指令的多个用途定义和引用以处理整个压缩数据集合。参数块的校验值387和校验值类型375字段将应用于压缩数据集合中的压缩数据块(例如,所有块)。参数块的子字节边界字段381将应用于各个块之间的转换。历史长度385和历史偏移386可应用于多个块。在一个实例中,参数块的剩余字段仅应用于正由DEFLATE转换调用指令的特定执行处理的个别经压缩数据块。

[0428] • 个别检查值例如应用于由压缩数据集合表示的所有未压缩数据。

[0429] • 不存在块1中的第一压缩的数据符号要引用的历史。块1中的后续符号可参考对应于块1中的先前遇到的符号的历史。块2中的符号可参考对应于块2和块1中的先前遇到的符号的历史。块3中的符号可参考对应于块3、2和1中的先前遇到的符号的历史。

[0430] 图10列出了用于压缩图9中描述的压缩数据集合900中的数据的样本程序1000的一部分的一个示例。进一步地,图11列出了在位于图10中标记为IABLK1 (1002) 的指令地址处的DFLTCC指令的执行期间使用的参数块的某些字段的值。例如,图11描绘了不同参数块字段1100;在压缩操作1102开始时的那些字段的值;当条件码1、2或3被设置1104时,在操作结束时那些字段的值;以及当条件码0被设置1106时那些字段在操作结束时的值。

[0431] 类似地,图12列出了在位于图10的标为IABLK2 (1004) 的指令地址处的DFLTCC指令的执行期间使用的参数块的某些字段的值。这些图展示了与多次使用DEFLATE转换调用指令来处理整个压缩数据集合相关联的细节中的一些细节。

[0432] 此外,参考图13,描绘了用于从图9的压缩的数据集合解压缩数据的样本程序1300的一部分的一个示例。

[0433] 压缩数据

[0434] 压缩数据的过程包括生成一个或多个压缩数据块。DEFLATE转换调用指令的压缩函数用以建构个别块的一部分。所述部分可为整个块。该函数生成具有块类型(BTYPE) 01或10二进制而不是00二进制的块的部分。当参数块的新任务位(NT)为1时,生成第一压缩数据块,并且没有来自先前执行的压缩操作的要引用的历史。

[0435] 在一个示例中,单个块按它们被列出的顺序包含以下元素:

[0436] 1. 最终块指示(BFINAL)。

[0437] 2. 块类型(BTYPE)。

[0438] 3. 动态霍夫曼表的压缩格式,当适用时。

[0439] 4. 压缩的数据符号。

[0440] 5. 块结束(EOB)符号。

[0441] 压缩操作生成以针对块定义的顺序指定的元素。元素可以在存储装置中的字节边界之间开始或结束。子字节边界(SBB)适用于将第一元素存储到第一操作数位置。压缩的数

据块是字节流。一次一位地将块的组件加载到存储器中。作为实例,位流在存储装置的每一字节内从右向左且按字节次序从左向右加载。

[0442] 当SBB为非零时,对第一操作数位置处的第一字节的引用为更新引用。

[0443] 来自第二操作数位置的未压缩数据被压缩并且作为压缩数据符号存储到第一操作数位置。

[0444] 当在指令的执行开始时第一操作数长度是零时,不访问第一操作数,并且分别在通用寄存器R1和R1+1中的第一操作数地址和第一操作数长度不改变。这在指令的执行开始时CF字段373(图3L)的值为0或1时适用。

[0445] 当在指令的执行开始时第二操作数长度是零时,第二操作数不被访问,并且通用寄存器R2和R2+1中的第二操作数地址和第二操作数长度分别不被改变。例如,对于以下情况,第二操作数长度在指令的执行开始时为零:

[0446] • 指令被重新执行以恢复操作(参数块的CF字段373在指令的执行开始时为1),并且完成操作可参考参数块的CSB字段392并且不参考第二操作数来执行。

[0447] • 在一个实施例中,所述程序不使用DEFLATE转换调用指令来执行以下操作: • 生成空的压缩数据块。空压缩数据块由例如块报头、DHT的压缩格式(当适用时)和EOB符号组成。

[0448] • 关闭打开的压缩数据块。即只存储EOB符号到压缩后的数据块的末尾。

[0449] • 压缩算法包括在最近压缩的数据的更新历史中搜索与来自第二操作数位置的当前正被压缩的数据匹配的字串。在一个实施例中,在压缩操作开始或恢复之前,应用以下内容:

[0450] • 当新任务(NT)374是1时,不存在可用于引用的初始历史。

[0451] • 当NT为零,且通用寄存器0(HBT)的位56为零(在线)时,可用于参考的初始历史位于第二操作数的最左字节的左侧且邻近于所述最左字节,且初始历史的长度由参数块的历史长度(HL)字段385指定。

[0452] • 当NT为零,且通用寄存器0(HBT)的位56为一(循环)时,可用于参考的初始历史位于第三操作数位置中,如由参数块的历史偏移(H0)386和历史长度(HL)385字段指定。

[0453] 在压缩操作期间,可以对整个历史进行获取型引用,而不管使用哪些字节的历史来执行操作。此外,当历史缓冲器类型是循环时,可以对整个32K字节的历史缓冲器进行提取类型引用,而不管使用哪些字节的历史来执行操作。

[0454] 在压缩操作期间,更新历史。在将源数据的一个或多个字节编码成压缩数据符号而不遇到一般操作数数据异常情况之后,源字节被级联到历史的结尾。源数据的最近处理的字节(最多32K字节)构成在处理源数据的后续字节的同时可用于参考的经更新的历史。

[0455] 当压缩操作结束时,在一个示例中,以下内容应用于可用于随后恢复操作或开始另一操作的结果历史:

[0456] • 当HBT在线时,当历史更新时不需要对第二操作数位置的存储更新。更新的第二操作数地址和更新的HL指定结果历史的更新位置和更新长度。

[0457] • 当HBT为循环时,当更新历史时执行对第三操作数位置的存储更新。第三操作数地址、更新的H0和更新的HL指定结果历史的更新的位置和更新的长度。

[0458] 作为示例,图14A-14C示出了当每个执行在部分完成的情况下结束时在具有指定

的DFLTCC-CMPR函数以及指定的在线历史(例如,位310=0)的DEFLATE转换调用指令的多次执行之前和之后在线历史缓冲器相对于第二操作数的位置。例如,图14A描绘了DFLTCC-CMPR执行编号1之前的在线历史;图14B描绘了DFLTCC-CMPR执行编号2之前和执行编号1之后的在线历史;以及图14C描绘了DFLTCC-CMPR执行编号2之后的在线历史。图14C中提供的解释也适用于图14A和14B。

[0459] 当由通用寄存器0的位56指定的HBT(历史缓冲器类型)为循环(例如,位310=1)时,将历史维持在例如位于第三操作数位置处的32K字节缓冲器中。通过例如通用寄存器R3的内容与历史偏移(H0)386(图3L)的总和指定缓冲器(HB)内的历史的第一字节的位置。历史的第一字节是缓冲器中的未压缩数据的最近最少处理的字节。例如,缓冲区(HE)内的历史的最后字节的位置由以下等式指定:

[0460] $HE = R3 + \text{modulo}(\text{模}) 32K (H0 + HL - 1)$

[0461] 历史的最后一个字节是缓冲器中的未压缩数据的最近处理的字节。当历史偏移(H0)386(图3L)与历史长度(HL)385的总和超过第三操作数的大小(例如,32K字节)时,历史从第三操作数的结尾缠绕至第三操作数的开始。

[0462] 作为实例,图15A-15E说明当每一执行以部分完成结束时在具有指定的DFLTCC-CMPR函数的DEFLATE转换调用指令的多次执行以及指定的循环历史缓冲器(位310=1)之前和之后的循环历史缓冲器内的历史的位置。例如,图15A描绘了DFLTCC执行编号1之前的循环历史缓冲器;图15B描绘了DFLTCC执行编号2之前和执行编号1之后的循环缓冲器;图15C描绘了DFLTCC执行编号3之前和执行编号2之后的循环缓冲器;图15D描绘了DFLTCC执行编号4之前和执行编号3之后的循环缓冲器;以及图15E描绘了DFLTCC执行编号4之后的循环缓冲器。图15E中提供的解释也适用于图15A-15D。

[0463] 在一个实例中,当HBT为循环且从第二操作数位置处理的字节的数目小于例如32,768时,以下适用:

[0464] • 对所述第三操作数位置中的字节范围进行存储。字节范围包括例如以下指定的位置并且从以下指定的位置开始:

[0465] $R3 + \text{modulo} 32K (H00 + HL0)$, 其中

[0466] H00:指令执行之前的历史偏移。

[0467] HL0:指令执行之前的历史长度。

[0468] 字节范围包含例如由以下各者指定的位置且以所述位置结束:

[0469] $R3 + \text{modulo} 32K (H00 + HL0 + BP - 1)$, 其中

[0470] BP:在指令的执行期间从第二操作数位置处理的字节的数量。

[0471] 作为实例,对刚刚描述的字节范围进行的存储经受存储类型访问异常、PER存储更改事件和设置改变位。

[0472] • 不对存储位置的内容进行修改并且不是必须的存储可以对第三操作数位置中不包括在刚刚描述的范围中的字节做出。到这些位置的存储还经受存储类型访问异常、PER存储变更事件和设置改变位。

[0473] 当HBT为循环且从第二操作数位置处理的字节的数目大于或等于例如32,768时,对第三操作数位置的所有字节进行存储且经受存储类型存取异常、PER存储更改事件和设置改变位。

[0474] 当块继续旗标 (BCF) 377 为 0 时, 3 位块标头 (包含 BFINAL, 之后是 BTYPE) 被存储到第一操作数位置。块报头的 BFINAL 字节被设置为等于参数块的块报头最终字节 (BHF) 379。当霍夫曼表类型 (HTT) 376 是 0 时, 块报头的 BTYPE 字段被设置为例如 01 二进制, 并且当 HTT 是 1 时, 块报头的 BTYPE 字段被设置为例如 10 二进制。当存储块标头时, 将 BFINAL 位存储到由 SBB 在第一操作数的第一字节中指定的位。随后, 将 BTYPE 存储到第一操作数位置。当 BCF 为 1 时, 不存储块头。

[0475] 当霍夫曼表类型 (HTT) 是 1 时, 针对一般操作数数据异常条件, 检查在参数块中指定的动态霍夫曼表 (DHT) 367 的压缩格式。当 DHT 的指定压缩格式存在通用操作数数据异常情况时, 压缩后的 DHT 称为无效, 不用于压缩数据。下面进一步描述一般操作数数据异常条件的示例定义。当 DHT 的压缩格式指定用于代码长度的字节长度时, 或字面字节的代码长度、EOB 符号、重复串长度, 或者复制串指针距离, 其大于霍夫曼算法指定适当和功能的霍夫曼树所需的长度, 压缩的 DHT 仍然用于导出功能 DHT 和压缩数据。当块继续旗标 (BCF) 为 0 且 HTT 为 1 时, 将如参数块的 CDHT 字段 367 中指定的 DHT 的压缩格式存储到第一操作数位置。

[0476] 在压缩操作期间, 将来自第二操作数位置的源数据编码成经压缩数据符号。作为编码的一部分, 将源数据与历史进行比较。当未找到匹配时, 源数据的中间表示是文字量字节, 其与源数据相同。当发现匹配时, 源数据的中间表示是指向历史内包含源数据的副本的位置的指针。指针由长度和距离组成。长度是与历史中的字符串匹配的源数据字节的数量。该距离是从历史的结尾到与源数据匹配的字符串的开始的字节的数量。在一个示例中, 使用来自霍夫曼表的两个霍夫曼代码树来将源数据的中间表示编码成压缩的数据符号。当霍夫曼表类型 (HTT) 是零时, 如由 DEFLATE 标准描述的固定-霍夫曼表 (FHT) 指定用于编码中间结果的两个霍夫曼代码树。当 HTT 376 为 1 时, 从在参数块的 CDHT 字段 367 中指定的 DHT 的压缩表示导出的动态霍夫曼表 (DHT) 指定用于编码中间结果的两个霍夫曼码树。如 DEFLATE 标准所述执行编码。当使用非通用 DHT (其未指定将用于对源数据的中间表示进行编码的霍夫曼代码) 时, 识别通用操作数数据异常。在将结果存储到第一操作数位置之前, 按 DEFLATE 标准指定的次序排列所得压缩数据符号的位。

[0477] 在一个示例中, 副本串长度范围从 3 字节至 258 字节。

[0478] 在处理进一步的源数据之前, 如本文所述, 更新历史。

[0479] 在一个示例中, 重复该过程, 直到已经处理了所有源字节。

[0480] 在已处理源字节 (例如, 所有源字节) 且块关闭控制 (BCC) 378 为 1 之后, 将块结束 (EOB) 符号存储到第一操作数位置。当使用固定霍夫曼表时, 霍夫曼码 00000000 二进制用于 EOB 符号。当使用动态霍夫曼表 (DHT) 时, 用于 EOB 符号的霍夫曼代码由 DHT 指定。在将 EOB 符号存储到第一操作数位置之前, EOB 符号的位按照 DEFLATE 标准指定的顺序排列。

[0481] 在一个示例中, 当操作的最后一个压缩的数据符号 (包括 EOB 符号) 仅占用最后一个字节的一部分来存储时, 不包含最后一个符号的一部分的位被存储为零。

[0482] 在一个实施例, 在处理最后的压缩数据符号之后, 发生以下情况:

[0483] • 模型相关值被存储到参数块的模型版本号 (MVN) 字段 363。

[0484] • 参数块的子字节边界 (SBB) 字段 381 被更新。

[0485] • 参数块的块结束长度 (EOBL) 389 和块结束符号 (EOBS) 388 字段被更新。

[0486] • 通用寄存器 R1 中的地址递增包括处理位 0 的第一操作数的处理的字节数, 并且

通用寄存器R1+1中的长度递减相同的数。包括处理位0的第一操作数的处理的字节数是由整数除法产生的整数商,其中除数是所处理的输出位的数量与SBB的原始值的和,除数是值8。

[0487] • 通用寄存器R2中的地址递增所处理的源字节的数目,且通用寄存器R2+1中的长度递减相同数目。

[0488] 地址和长度的形成和更新取决于寻址模式。

[0489] 与压缩源数据一致,源数据是用于生成如上所述的32位校验值的输入。将所得到的校验值存储到参数块的校验值字段387中。

[0490] 解压缩数据

[0491] 在一个实施例中,DEFLATE转换调用指令的扩展函数用于将压缩的数据集解码成未压缩的数据。第二操作数位置中的压缩数据集包括一个或多个连续压缩数据块。在一个实例中,从左到右处理数据集的块,且例如从左到右处理块的字节。块可以或可以不在字节边界上开始或结束。每个块独立于数据集中的其他块被解码。通用寄存器R2指定数据集中的第一块的最左边字节的逻辑地址。数据集中的最后一个块是在BFINAL位等于1的处理期间遇到的块。在一个实例中,存在要处理的三种类型的块。对块的内容进行解码的技术是块类型(BTYPE)的函数。

[0492] 当操作开始时(例如,当参数块的继续旗标字段373为零时),由通用寄存器R2、新任务(NT)字段374和子字节边界(SBB)字段381指定的位被解释为经压缩数据块的第一位(块标头的BFINAL位)。

[0493] 扩展功能包括引用最近解码的未压缩数据的更新的历史。在一个实施例中,在解压操作开始或恢复之前,以下适用:

[0494] • 当新任务(NT)374是1时,不存在可用于引用的初始历史。

[0495] • 当NT为零,且通用寄存器0(HBT)的位56为零(在线)时,可用于参考的初始历史位于第一操作数的最左字节的左侧且邻近于最左字节,且初始历史的长度由参数块的历史长度(HL)字段385指定。

[0496] • 当NT为零,且通用寄存器0(HBT)的位56为一(循环)时,可用于参考的初始历史位于第三操作数位置中,如由参数块的历史偏移(H0)386和历史长度(HL)385字段指定。

[0497] 在操作期间,可以对整个历史进行获取型引用,而不管使用哪些字节的历史来执行操作。此外,当历史缓冲器类型为循环时,可对整个历史缓冲器(例如,32K字节)进行提取类型引用,而不管使用哪些字节的历史来执行操作。

[0498] 在解压操作期间,更新历史。在解码源数据而不遇到一般操作数数据异常情况之后,将未压缩数据的所得字节串接到历史的结尾。直到例如32K字节的最大值的未压缩数据的最近解码字节构成在处理后续源数据时可用于参考的经更新历史。

[0499] 在一个实例中,当解压缩操作结束时,以下内容适用于可用于随后恢复操作或开始另一操作的所得历史:

[0500] • 当HBT在线时,对第一操作数位置的存储更新也构成对所得历史的更新。更新的第一操作数地址和更新的HL指定结果历史的更新的位置和更新的长度。

[0501] • 当HBT为循环时,当更新历史时执行对第三操作数位置的存储更新。第三操作数地址、更新的H0和更新的HL指定结果历史的更新的位置和更新的长度。

[0502] 作为示例,图16A-16C示出了当每个执行以部分完成结束时在具有指定的DFLTCC-XPND函数以及指定的在线历史的DEFLATE转换调用指令的多次执行之前和之后在线历史缓冲器相对于第一操作数的位置的示例。在操作期间修改历史长度(HL) 385。例如,图16A描绘了在DFLTCC-XPND执行号1之前的在线历史的一个实例;图16B描绘了DFLTCC-XPND执行编号2之前和执行编号1之后的在线历史的示例;并且图16C描绘了DFLTCC-XPND执行编号2之后的在线历史的示例。图16C中提供的解释也适用于图16A-16B。

[0503] 当由通用寄存器0的位56指定的HBT为循环时,在例如位于第三操作数位置处的32K字节缓冲器中维持历史。通过通用寄存器R3的内容和历史偏移(HO) 386的总和来指定缓冲器(HB)内的历史的第一字节的位置。历史的第一字节是缓冲器中的未压缩数据的最近最少处理的字节。缓冲区(HE)内的历史的最后字节的位置由例如以下等式指定:

[0504] $HE = R3 + \text{modulo}32K(HO + HL - 1)$ 。

[0505] 历史的最后一个字节是缓冲器中的未压缩数据的最近处理的字节。当历史偏移(HO)与历史长度(HL)的总和超过第三操作数的大小(例如,32K字节)时,历史从第三操作数的结尾卷绕到第三操作数的开始。在此所描述的图15A-15E展示了当每个执行以部分完成结束时在具有指定的DFLTCC-XPND函数和循环历史缓冲器的DEFLATE转换调用指令的多次执行之前和之后的循环历史缓冲器内的历史的位置的示例。

[0506] 在一个实例中,当HBT为循环且存储到第一操作数位置的字节的数目小于例如32,768时,以下情况适用:

[0507] • 对所述第三操作数位置中的字节范围进行存储。字节的范围包括由以下指定的位置并且从以下指定的位置开始:

[0508] $R3 + \text{modulo}32K(HO0 + HL0)$, 其中

[0509] HO0: 指令执行之前的历史偏移。

[0510] HL0: 指令执行之前的历史长度。

[0511] 字节范围包含例如由以下各者指定的位置且以所述位置结束:

[0512] $R3 + \text{modulo}32K(HO0 + HL0 + BP - 1)$, 其中

[0513] BP: 在指令的执行期间存储到第一操作数位置的字节的数目。

[0514] 对刚刚描述的字节范围进行的存储经受存储类型访问异常、PER储存变更事件和设置改变位。

[0515] • 可以对第三操作数位置中不包括在刚刚描述的范围中的字节进行不修改存储位置的内容并且不是必需的存储。到这些位置的存储还经受存储类型访问异常、PER储存变更事件和设置改变位。

[0516] 当HBT为循环且存储到第一操作数位置的字节的数目大于或等于例如32,768时,对例如第三操作数位置的所有字节进行存储,且经受存储类型存取异常、PER存储更改事件和设置改变位。

[0517] 当BTYPE为00二进制时,该块不包含压缩的数据。在此描述的图6展示了具有等于00二进制的BTYPE的块的一个实例。LEN字段指定块中的文字量字节的数量。LEN字段的字节顺序是小字节序。LEN字段可以指定零文字量字节。块的字面字节被放置在第一操作数位置处。如先前所描述的,还利用块的每个文字量字节来更新历史。

[0518] 当BTYPE为01二进制时,该块包含使用固定的霍夫曼表(FHT)生成的压缩的数据符

号。FHT由DEFLATE标准定义并且不是块的一部分。如本文所述,图7示出了具有BTTYPE等于01二进制的块的一个实例。在解释块报头之后,按照压缩数据符号在块中出现的顺序对压缩数据符号进行解码。例如从左到右处理块的字节,且例如从右到左处理块的每一字节内的位。在一个实例中,在处理块中的下一符号之前完全处理每一符号。不是块结束(EOB)符号的每个符号表示字面值或指向先前在历史缓冲器中解码的子串的指针。先前解码的子串也被称为重复串。在一个示例中,复制串长度范围从3字节至258字节。指针由表示子串长度和从历史末尾到子串开头的距离的代码组成。当符号表示历史中的子串时,从历史缓冲器中引用该子串。将由解码符号产生的未压缩数据放置在第一操作数位置处。

[0519] 在处理进一步的源数据之前,如先前描述的那样更新历史。

[0520] 经更新的历史适用于解码块的下一符号。当遇到EOB符号时,块的处理完成。

[0521] 当BTTYPE是10二进制时,块包含使用动态霍夫曼表(DHT)生成的压缩的数据符号。所使用的DHT的压缩格式为压缩数据块的元素。在此描述的图8展示了具有等于10二进制的BTTYPE的块的一个实例。在解释块头部之后,检查在压缩数据块内提供的DHT的压缩格式以寻找一般操作数数据异常情况。当提供的DHT的压缩格式存在通用操作数数据异常情况时,DHT的压缩格式称为无效,不用于解压缩数据。当DHT的压缩格式指定用于代码长度的字节长度时,或字面字节的代码长度、EOB符号、重复串长度,或者复制串指针距离,其大于霍夫曼算法指定适当和功能的霍夫曼树所需的长度,压缩的DHT仍然用于导出功能DHT和压缩数据。在检查DHT的压缩格式之后,按照压缩数据符号出现在块中的顺序对压缩数据符号进行解码。例如从左到右处理块的字节,且例如从右到左处理块的每一字节内的位。在一个实例中,在处理块中的下一符号之前完全处理每一符号。具有BTTYPE 10二进制的块中的符号的处理与先前描述的用于处理具有BTTYPE 01的块中的符号的处理相同,除了前者使用被提供用于解码符号的DHT,而后者使用FHT来解码符号。当提供非通用DHT(其未指定要用于对压缩的数据符号进行解码的霍夫曼代码)时,识别通用操作数数据异常。

[0522] 与解压缩第二操作数一致,未压缩数据是用以产生检查值(例如,32位检查值)的输入。将所得到的校验值存储到参数块的校验值字段387中。

[0523] 在一个实施例中,在处理数据集合的最后一个块之后,发生以下情况:

[0524] • 模型相关值被存储到参数块的模型版本号(MVN)字段363。

[0525] • 参数块的子字节边界(SBB)字段381被更新。

[0526] • 通用寄存器R1中的地址递增存储在第一操作数位置处的字节的数目,且通用寄存器R1+1中的长度递减相同数目。

[0527] • 通用寄存器R2中的地址递增包含处理位0的第二操作数的处理的字节的数目,且通用寄存器R2+1中的长度递减相同数目。包括处理位0的第二操作数的处理的字节的数量是从整数除法得到的整数商,其中除数是所处理的输入位的数量和SBB的原始值的总和,除数是值8。

[0528] 地址和长度的形成和更新取决于寻址模式。

[0529] 当在指令的执行开始时第一操作数长度是零时,不访问第一操作数,并且分别在通用寄存器R1和R1+1中的第一操作数地址和第一操作数长度不改变。这在指令的执行开始时CF字段373的值为零或一时适用。

[0530] 当第二操作数长度在指令的执行开始时为零时,第二操作数不被存取,且通用寄

寄存器R2及R2+1中的第二操作数地址及第二操作数长度分别不改变。在一个实施例中,对于以下情况,第二操作数长度在指令的执行开始时为零:

- [0531] • 指令被重新执行(例如,参数块的CF字段373在指令的执行的开始处为1),
- [0532] 并且当指令先前被执行时整个第二操作数被处理。

[0533] 解压缩操作可在不将任何结果存储到第一操作数位置的情况下结束,即使数据是从第二操作数位置处理的。在一实例中,当从第二操作数位置处理的数据仅含有以下经压缩数据块元素中的任一者时,此情形发生:

- [0534] • 块报头。
- [0535] • 具有块类型00二进制的块的LEN字段。
- [0536] • 具有块类型00二进制的块的NLEN字段。
- [0537] • 动态霍夫曼表的压缩格式。
- [0538] • 块结束(EOB)符号。

[0539] 在一个或多个实施例中,以下条件适用于执行DEFLATE转换调用指令:

[0540] 在一个示例中,当指定DFLTCC-GDHT函数并且发生以下条件时,识别一般操作数数据异常:

- [0541] • 模型不支持参数块版本号362指定的参数块格式。

[0542] 在一个实例中,当指定DFLTCC-CMPR函数且发生以下条件中的任一者时,识别一般操作数数据异常:

- [0543] • 模型不支持参数块版本号362指定的参数块格式。
- [0544] • NT374为零且HL385大于例如32768。
- [0545] • HTT 376是1并且CDHTL 366小于例如42或大于例如2283。
- [0546] • HTT376为1,并且CDHTL366不等于CDHT字段367中指定的DHT的压缩格式的长度。
- [0547] • HTT 376是1并且DHT的压缩格式的HLIT子元素大于例如29(无效DHT)。
- [0548] • HTT376为1,并且DHT的压缩格式的HDIST子元素大于例如29(无效DHT)。

[0549] • HTT 376是1,并且DHT的压缩格式(CDHT字段367的内容)指定代码,该代码在指定用于的字节长度的代码序列中,例如,针对压缩的DHT定义的19个可能的代码长度,并且小于霍夫曼算法指定功能霍夫曼树(无效DHT)所需的长度。

[0550] • HTT 376是1并且DHT的压缩格式(CDHT字段367的内容)指定代码长度(例如,16(复制前一代码长度))作为用于由字面字节、EOB符号和重复串长度(无效DHT)组成的元素集合的第一代码长度。

[0551] • HTT376为1,并且DHT的压缩格式(CDHT字段367的内容)指定代码,该代码在指定用于字面字节的代码长度的代码序列中,并且所述代码与被确定为表示所述参考代码长度集合的代码中的任何代码不匹配,如之前在压缩的DHT(无效DHT)中所指定的。

[0552] • HTT376是1,并且DHT的压缩格式(CDHT字段367的内容)指定将码长0(CLO)分配给EOB符号的码。在这种情况下,相应的DHT不指定霍夫曼代码来表示EOB符号(无效DHT)。

[0553] • HTT376是1,并且DHT的压缩格式(CDHT字段367的内容)指定在代码序列中的代码,该代码序列指定用于重复串长度和指针距离的代码长度,并且所述代码与被确定为表示所述参考代码长度集合的代码中的任何代码不匹配,如之前在压缩的DHT(无效DHT)中所指定的。

[0554] • HTT376是1,并且DHT的压缩格式(CDHT字段367的内容)指定大于DHT中的霍夫曼码的数目的码长的数目,如由HLIT字段、HDIST字段和例如258中的值的总和指定的。作为示例(无效DHT),这在不正确地使用代码长度16、17和18的情况下是可能的。

[0555] • HTT376是一个并且DHT的压缩格式(CDHT字段367的内容)指定用于字面字节集、EOB符号和重复串长度的代码长度,其小于由霍夫曼算法指定功能霍夫曼树(无效DHT)所需的长度。

[0556] • HTT376是1,并且DHT的压缩格式(CDHT字段367的内容)指定用于复制串指针距离的集合的代码长度,其小于由霍夫曼算法指定功能霍夫曼树(无效DHT)所需的长度。

[0557] • CPU尝试生成压缩的数据符号以表示第二操作数中的字面字节,并且从CDHT字段的内容导出的DHT是非通用的并且不指定对应于该字面字节的霍夫曼代码。

[0558] • CPU尝试生成压缩的数据符号以表示第二操作数中的复制串,并且从CDHT字段的内容导出的DHT是非通用的,并且不指定对应于该复制串长度或指针距离的霍夫曼代码。

[0559] 作为实例,当例如指定DFLTCC-XPND函数并且发生以下条件中的任一个时,识别通用操作数数据异常:

[0560] • 模型不支持参数块版本号362指定的参数块格式。

[0561] • NT374为零且HL385大于例如32768。

[0562] • 遇到BTYP等于11二进制的压缩数据块。

[0563] • 遇到BTYP等于00二进制且NLEN不等于LEN的1's补码的经压缩数据块。

[0564] • 遇到DHT的压缩格式(BTYP等于10二进制的压缩数据块的内容),并且压缩的DHT的HLIT子元素大于例如29(无效DHT)。

[0565] • 遇到DHT的压缩格式(BTYP等于10二进制的压缩数据块的内容),并且压缩的DHT的HDIST子元素大于例如29(无效DHT)。

[0566] • 遇到DHT的压缩格式(BTYP等于10二进制的压缩数据块的内容),其指定代码序列中的代码,该代码序列指定位长度,例如,针对压缩的DHT定义的19个可能的代码长度,并且小于霍夫曼算法指定功能霍夫曼树(无效DHT)所需的长度。

[0567] • 遇到DHT的压缩格式(具有等于10二进制的BTYP的压缩数据块的内容),其指定代码长度(例如,16(复制先前代码长度))作为用于由字面字节、EOB符号和重复串长度(无效DHT)组成的元素集合的第一代码长度。

[0568] • 遇到DHT的压缩格式(具有等于10二进制的BTYP的压缩数据块的内容),其指定代码序列中的代码,该代码指定字面字节的代码长度,并且所述代码与被确定为表示所述参考代码长度集合的代码中的任何代码不匹配,如之前在压缩的DHT(无效DHT)中所指定的。

[0569] • 遇到DHT的压缩格式(BTYP等于10二进制的压缩数据块的内容),其指定将代码长度0(CLO)分配给EOB符号的代码。在这种情况下,相应的DHT不指定霍夫曼代码来表示EOB符号(无效DHT)。

[0570] • 遇到DHT的压缩格式(BTYP等于10二进制的压缩数据块的内容),其指定代码序列中的代码,该代码序列指定用于重复串长度和指针距离的代码长度,并且所述代码与被确定为表示所述参考代码长度集合的代码中的任何代码不匹配,如之前在压缩的DHT(无效DHT)中所指定的。

[0571] • 遇到DHT的压缩格式(具有等于10二进制的BTYPE的压缩数据块的内容),其指定大于DHT中的霍夫曼码的数目的码长的数目,如由HLIT字段、HDIST字段和例如258中的值的总和所指定。例如(无效DHT),这有可能不正确地使用码长16、17和18。

[0572] • 遇到DHT的压缩格式(具有等于10二进制的BTYPE的压缩数据块的内容),其指定用于字面字节集、EOB符号和重复串长度的代码长度,其小于由霍夫曼算法指定功能霍夫曼树(无效DHT)所需的长度。

[0573] • 遇到DHT的压缩格式(具有等于10二进制的BTYPE的压缩数据块的内容),其指定用于复制串指针距离的集合的代码长度,其小于由霍夫曼算法指定功能霍夫曼树(无效DHT)所需的长度。

[0574] • 在BTYPE等于10二进制的压缩数据块中遇到的压缩数据符号指定霍夫曼代码,该霍夫曼代码不是由在同一块中从DHT的压缩格式导出的非通用DHT定义的。在这种情况下,第二操作数的可用于处理的位数(出于识别通用操作数数据异常的目的)是依赖于模型的。更具体来说,尝试解码未定义代码的模型可在辨识异常之前处理例如15位,即使可在处理较少位之后确定异常。

[0575] • 遇到压缩的数据符号,其是复制串指针并且指定大于在处理符号的点处可用的历史的长度的距离。

[0576] • 在BTYPE等于01二进制的压缩数据块中遇到的压缩数据符号指定无效代码(例如,用于复制串长度的11000110或11000111二进制的代码,或用于复制串指针距离的11110或11111二进制的代码)。在这种情况下,第二操作数的可用于处理的位数(出于识别通用操作数数据异常的目的)是依赖于模型的。更具体地,尝试对无效代码进行解码的模型可以在识别异常之前处理例如8位(在重复串长度的情况下)或5位(在重复串指针距离的情况下),即使可以在处理较少的位之后确定异常。

[0577] 当识别到一般操作数数据异常时,即使更新参数块的操作结束补充代码(OESC)365和模型版本号(MVN)字段363以提供与异常相关联的额外信息,也认为操作受到抑制。

[0578] 当DFLTCC-CMPR或DFLTCC-XPND函数正在被执行并且一般操作数数据异常是由于针对第二操作数被识别时,结果是异常被识别,或操作以部分完成和条件码(例如,3被设置)结束。如果条件码3被设置,则当再次执行指令以继续处理相同的操作数并且异常条件仍然存在时,将识别异常。

[0579] 其他条件包括例如:

[0580] 指令的执行是可中断的。当发生中断时,更新通用寄存器R1和R2中的地址、通用寄存器R1+1和R2+1中的长度以及参数块的特定字段,以使得指令在重新执行时在中断点处恢复。

[0581] 当DFLTCC-CMPR或DFLTCC-XPND函数正在被执行并且访问异常是由于针对第一或第二操作数被识别时,结果是异常被识别,或操作以部分完成和条件码(例如,3被设置)结束。如果条件码3被设置,则当再次执行指令以继续处理相同的操作数并且异常条件仍然存在时,将识别异常。

[0582] 如由此CPU、其他CPU及通道程序所观察,对参数块、第一、第二及第三操作数的引用可为多路存取引用,对这些存储位置的存取未必为并行的块,且这些存取或引用的序列未定义。

[0583] 在一个实施例中,如果指定了DFLTCC-CMPR或DFLTCC-XPND功能并且应用了以下各项中的任一项,则结果是不可预测的:

[0584] • 参数块与第一或第二操作数重叠。

[0585] • 第一操作数与第二操作数重叠。

[0586] • 指定的历史缓冲器类型(HBT)是循环,并且第三操作数与第一操作数、第二操作数或参数块重叠。

[0587] • 指定的历史缓冲器类型(HBT)在线,指定DFLTCC-CMPR函数,并且历史与第一操作数或参数块重叠。

[0588] • 指定历史缓冲器类型(HBT)在线,指定DFLTCC-XPND函数,并且历史与第二操作数或参数块重叠。

[0589] 在某些情况下,尽管结束DEFLATE转换调用指令的执行,其中处理的CPU确定的字节数为零,但数据可能已存储到第一操作数位置,数据可能已存储到第三操作数位置(在适用时),且对应的改变位已设置(在适用时)。在这些情况下,参数块和通用寄存器的内容尚未从原始值修改。当CPU在执行DEFLATE转换调用指令的同时执行静默操作或者CPU重试时,可能发生这些情况。

[0590] 以下是从执行DEFLATE转换调用指令得到的示例条件码:

[0591] 0 正常完成

[0592] 1 第一操作数长度不足以完成操作

[0593] 2 第二操作数长度不足以完成操作(DFLTCC-XPND)

[0594] 3 CPU确定的处理数据的量

[0595] 程序异常:

[0596] • 访问(提取,操作数2,在线历史;提取和存储,参数块,操作数1,操作数3)

[0597] • 具有DXC 0的数据,一般操作数

[0598] • 操作(如果未安装DEFLATE转换设施)

[0599] • 规范

[0600] • 交易约束

[0601] DEFLATE CONVERSION CALL指令的执行的示例优先级如下所示:

[0602] 1.-6.具有与用于一般情况的程序中断条件的优先级相同的优先级的异常。

[0603] 7.A第二指令半字的访问异常。

[0604] 7.B操作异常。

[0605] 7.C事务限制。

[0606] 8.A归因于无效功能代码或无效寄存器编号的规范异常。

[0607] 8.B归因于未在4K字节边界上指定的参数块的规范异常。

[0608] 8.C由于在4K字节边界上未指定循环历史缓冲器而引起的规范异常。

[0609] 9.访问参数块的访问异常。

[0610] 10.该模式不支持参数块的指定格式时通用操作数数据异常。

[0611] 11.由于在指令的执行开始时第二操作数长度等于零且CF等于零而引起的规范异常。

[0612] 12.由于在指令的执行的开始处第一操作数长度等于零并且指定DFLTCC-CMPR而

引起的条件码1。

[0613] 13.A当指定DFLTCC-CMPR或DFLTCC-XPND时,由于历史长度字段大于32,768和新任务字段引起的一般操作数数据异常为零。

[0614] 13.B用于访问所述第一操作数的访问异常,并且所述第一操作数长度为非零。13.C访问第二操作数的异常,并且第二操作数长度非零。

[0615] 13.D用于访问在指令的执行的开始处指定的在线历史的访问异常。

[0616] 13.E对第三操作数的访问的访问异常。

[0617] 14.A由于除了以上项目10和13.A中包括的条件之外的条件引起的一般操作数数据异常。

[0618] 14.B由于除上述项目12中包括的条件之外的条件码1、2或3。

[0619] 15.条件码0。

[0620] 在使用之前,检查DHT的压缩格式是否存在一般操作数数据异常情况。当由于一般操作数数据异常情况而未精确地定义DHT的压缩格式的长度时,所解释的长度可以取决于条件、取决于模型并且不超过例如286字节。结果,当指定DFLTCC-XPND函数并且在例如第二操作数的最右边286字节中遇到具有一般操作数数据异常状况的DHT的压缩格式时,识别异常状况(优先级14.A)还是状况代码2(优先级14.B)是依赖于模型的。

[0621] 以下提供了示例编程注释:

[0622] 1.当压缩或解压缩数据时,当以执行DEFLATE转换调用指令的最小次数执行操作时,总体来说可能更有效率。换言之,执行具有大操作数的DFLTCC可能比执行具有小操作数的DFLTCC多倍更有效。

[0623] 2.对于压缩和解压缩操作,当条件码3被设置时,由指令和参数块使用的通用寄存器已经被更新,使得程序可分支回到指令以继续操作。

[0624] 3.在一个实施例中,在执行由指令的参数指定的处理的CPU确定的子部分之后,可完成DEFLATE转换调用指令。当在仅执行CPU确定量的处理而不是所有指定处理之后指令完成时,指令集条件码3。在这样的完成时,PSW(程序状态字)中的指令地址指定下一顺序指令,并且指令的操作数参数已经被调整,使得指令的处理可以通过分支回到指令以再次执行指令来恢复。当指令已执行所有指定处理时,其设定除3以外的条件码。

[0625] 4.当指定DFLTCC-CMPR函数并且操作以参数块的子字节边界(SBB)字段中的非零值结束时,操作包括存储到由得到的第一操作数地址指定的字节。当指定DFLTCC-XPND函数并且操作以SBB中的非零值结束时,操作包括提取由所得第二操作数地址指定的字节。

[0626] 5.当操作结束于非零条件码集时,参数块的CSB字段392可包含部分处理的数据,并且预期程序重新执行指令以恢复操作。

[0627] 6.在以非零条件码集结束的操作之后,并且在为了恢复操作的目的而重新执行指令之前,程序不修改参数块的任何字段;否则结果是不可预测的。

[0628] 7.当指定DFLTCC-GDHT功能时,根据霍夫曼算法,生成的DHT的压缩表示描述三个适当的完整霍夫曼代码树。即,没有描述不完整的霍夫曼代码树。从DHT的压缩表示中导出不完整的霍夫曼代码树,DHT指定大于霍夫曼算法指定适当和功能的霍夫曼树所需的长度的元素的代码长度。

[0629] 当指定DFLTCC-CMPR功能时,HTT为1,并且DHT的压缩表示包括欠满霍夫曼码树的

描述,利用DFLTCC-XPND函数将压缩数据结果转换为原始未压缩数据,但不是符合DEFLATE标准的所有解码器都能够将结果转换为原始未压缩数据。例如,当由于执行DFLTCC-GDHT功能而没有生成由DFLTCC-CMPR功能的程序指定的DHT的压缩表示时,这可能发生。

[0630] 8. 当DFLTCC-CMPR函数以设置的条件码1结束时,存储到参数块的子字节边界(SBB)字段381的结果是000二进制。识别此情形可与分配输出缓冲器以供与DEFLATE转换调用指令一起使用的程序相关。

[0631] 如本文中所述,在一个方面中,提供单个指令(例如,硬件/软件接口处的单个架构化机器指令,例如,DEFLATE转换调用指令)以使用通用处理器执行压缩和/或解压缩操作。该指令例如是指令集架构(ISA)中定义的硬件指令。因此,降低了与压缩和/或解压缩操作相关的程序的复杂性。此外,提高了操作的性能,并因此提高了处理器的性能。

[0632] 有利地,DEFLATE转换调用指令由例如程序员在通用处理器(例如,中央处理单元,在本文中称为处理器)上而不是专用处理器(诸如I/O设备、通过I/O接口连接的应用专用设备或其他类型的专用处理器)上分派。与软件实现方式相比,执行所披露的指令需要显著更少的执行周期来执行相同的操作。进一步,与向I/O设备分派操作相比,执行所公开的指令不要求操作系统的I/O操作,并且不触发操作系统在等待操作完成时执行任务切换。

[0633] 尽管描述了不同字段和寄存器,本发明的一个或多个方面可以使用其他、附加或更少的字段或寄存器,或其他大小的字段和寄存器等。许多变化是可能的。例如,可以使用隐式寄存器来代替显式指定的指令寄存器或字段,和/或可以使用显式指定的寄存器或字段来代替隐式寄存器或字段。其他变化也是可能的。

[0634] 参见图17描述使用DEFLATE转换调用指令的一个实施例。在一个实例中,在处理器(例如,通用处理器)上执行的程序指定将在存储装置中的参数块中执行的操作的细节且指定参数块的位置(步骤1700)。例如,取决于待执行的功能,提供或设置参数块(例如,参数块340、360或370)的字段中的一者或一者以上。进一步,在步骤1702,程序指定要执行的操作(例如,查询、生成、压缩、扩展等)。另外,程序指定或更新输入数据在存储中的位置和量(步骤1704)以及结果缓冲器在存储中的位置和大小(步骤1706)。

[0635] 此后,程序执行DEFLATE转换调用(DFLTCC)指令,步骤1708。在一个示例中,在通用处理器上分派指令。作为示例,其在通用处理器上被处理,或至少部分地由耦合到通用处理器且不使用I/O接口可访问的硬件来处理。

[0636] 基于指令终止,确定由执行产生的条件码是否等于第一定义值(例如,0)(询问1710)。如果条件码等于第一定义值,那么指令的处理完成(步骤1712)。然而,如果条件码不等于第一定义值,则进一步确定条件码是否等于第二定义值(例如,3)(询问1714)。如果条件码等于指示存在要处理的附加数据的第二定义值,则重新执行指令(步骤1708)。然而,如果条件码不等于第二定义值,则进行关于条件码是否被设置为第三定义值(例如,1)的另一确定(询问1716)。如果条件码被设置为指示第一操作数长度不足的第三定义值,则处理继续步骤1706;否则,第二操作数长度不足以用于函数且处理以步骤1704继续。

[0637] 如所指示的,DEFLATE转换调用指令可被执行多次以压缩或解压缩单个数据流。因此,在一个方面,DEFLATE转换调用指令包括提供程序声明缓冲器(例如,32K字节缓冲器)的机制的属性,该缓冲器用于累积在跨越DEFLATE转换调用指令的多次执行的操作期间处理的未压缩数据的历史。缓冲器例如是循环历史缓冲器。

[0638] 在一个方面中,DEFLATE转换调用指令使用隐式寄存器(例如,GR0.56)中的指示符(例如,位)来指示循环历史缓冲器的使用。当循环历史缓冲区被指示并且DEFLATE转换调用指令要执行的指定函数是压缩或解压缩数据时,指令的字段(例如,R3)指定例如存储器中的位置,32K字节的缓冲器,处理器使用该缓冲器来从操作开始时提取历史并且将历史存储到操作结束时为止。循环历史缓冲器内的历史的长度由与DEFLATE转换调用指令相关联的参数块的字段(例如,HL字段385)指定,且缓冲器内的历史的开始由包含在参数块的另一字段(例如,H0字段386)中的偏移指定。

[0639] 参见图18描述使用循环历史缓冲器的进一步细节。在一个实例中,在处理器(例如,通用处理器)上执行的程序指定将在存储装置中的参数块中执行的操作的细节且指定参数块的位置(步骤1800)。例如,根据要执行的功能,提供或设置参数块(例如,参数块360或370)的一个或多个字段。进一步,程序指定要执行的操作(例如,压缩、扩展等)。

[0640] 进一步,在一个示例中,程序分配和指定预定义大小(例如,32K字节)的环形缓冲器在存储器中的位置(步骤1802)。另外,程序将未压缩数据流的一部分放置到缓冲器中,并且指定缓冲器的位置和大小作为对DEFLATE转换调用指令的输入(步骤1804),并且指定或更新存储装置中的结果缓冲器的位置和大小(步骤1806)。

[0641] 接着执行DEFLATE转换调用指令(步骤1808)。基于执行指令,处理器从(例如)循环历史缓冲器提取历史作为对操作的输入(步骤1820),且执行指定操作(步骤1822),如本文中所描述。进一步,处理器修改循环历史缓冲器中的历史作为操作的输出(步骤1824)。进行关于整个数据流是否已被处理的确定,查询1826。如果不是,则处理继续步骤1804。否则,处理完成。

[0642] 循环历史缓冲器的使用提供以下内容,作为实例:

[0643] 当输入或输出缓冲器的大小被指定用于DEFLATE转换调用指令的个别执行时,相对较小(例如,512字节),历史跨越缓冲数据的多个段,多达例如32K字节可用作DEFLATE转换调用指令的输入,其处理少量的字节。

[0644] 当输入或输出缓冲器的大小被指定用于DEFLATE转换调用指令的个别执行时,相对较大(例如,128K字节),是缓冲数据的先前段的历史,多达例如32K字节可用作到正处理前32K字节的数据的DEFLATE转换调用指令的输入。

[0645] 在两种情况下,相比以其他方式可用的历史,更多的历史可用于处理数据。结果,检测重复串的有效性得到改善,从而导致改善的整体压缩比。这有利于在计算环境内的处理并且改善性能。

[0646] 本发明的一个或多个方面与计算机技术不可扩展地关联并且促进计算机内的处理,从而改进其性能。使用单个架构机器指令来执行压缩和/或解压缩改进了计算环境内的性能。压缩/解压缩数据可用于管理和/或使用数据的许多技术领域,诸如计算机处理、医疗处理、安全、库存控制等。通过在压缩/解压缩中提供优化,通过减少执行时间来改进这些技术领域。

[0647] 当涉及本发明的一个或多个方面时,参见图19-22描述一个或多个实施例的进一步细节。

[0648] 现在转向与本发明的方面更具体地相关的技术的概述,DEFLATE是一种用于压缩数据的算法,该数据的大小可以是几千兆字节(GB),其中应用一次可能仅具有小的缓冲区,

并且压缩必须在相对小的块中完成,该相对小的块可以是1兆字节(MB)或更小。DEFLATE可通常指代在加速器或NXU上运行的复杂指令集,其可用类似于用于直接存储器存取(DMA)的I/O接口的输入/输出(I/O)接口附接到芯片上相干性子系统(例如,L3高速缓冲存储器)。从架构角度来看,DEFLATE需要遵循某些主要规则

[0649] 进一步,复杂可中断指令是具有需要保存下来的状态的指令(例如,DEFLATE;DEFLATE转换调用指令;来自在加速器上运行的复杂指令集的指令),使得在中断被服务之后可重新启动指令。注意,该状态可以是其需要实现重启的任何状态,提供满足特定格式化要求(例如,参数块格式)。特定格式化要求可专门针对原始机器模型进行微架构。进而,当引入具有相同复杂可中断指令的新机器模型时,针对新机器模型具体微架构化相同复杂可中断指令的状态。从软件预期来看,状态的区域是不透明的,因为它仅可由为其创建的机器模型理解。由此,在引入新机器模型之后,计算环境包括执行具有两个不同参数块格式的相同复杂可中断指令的两个机器(例如,原始机器模型和新机器)。

[0650] 对于具有复杂函数和参数块的指令,根据本文的一个或多个实施例采取新的虚拟架构级(VAL)方法。即,在不同复杂计算环境中,VAL用于处理迁移系统,使得通过利用针对所有虚拟机设置到最低公共机器级别的指示符(在VAL内),所有指令支持来自任何较低机器级别的所有输入。由此,本发明的一个或多个实施例通过在多个机器生成上(例如,在计算环境的原始机器模型和新机器之间)维持复杂功能的兼容性来解决上述缺点。进一步,本文的一个或多个实施例定义各代之间的兼容性级别,该兼容性级别要求每个机器仅理解两个不同的参数块格式。对此,每个机器实现其自己的指令和参数块的级别以及最低公分母。

[0651] 图19描绘了根据本发明的实施例的系统1900。系统1900包括支持具有兼容性级别1929的虚拟机1920的机器版本(n)1910,所述兼容性级别1929在其中包含汇编程序版本n1930。汇编版本n1930包括用于所有机器版本的最低公分母的指示符(例如,最低公分母指示符1931)。

[0652] 系统1900还包括支持具有兼容性级别1959的虚拟机1950的机器版本(n+1)1940,所述兼容性级别1959在其中包含解析锁版本n1930和解析锁版本n+11960。系统1900还包括支持具有兼容性级别1989的虚拟机1980的机器版本(n+2)1970,该虚拟机1980中包含参数块(parmblock)版本n1930和参数块版本n+21990。兼容性级别1929、1959和1989内的最低公分母指示符1931通过一系列设施位来控制,所述设施位识别哪些函数在特定虚拟机1920、1950和1980中可用。

[0653] 注意,“n”是生成的指定;进而,如果n=0,则机器版本1910是第一生成机器,机器版本1940是第二生成机器,并且机器版本1970是第三代机器。机器版本1910、1940和1970可以是包括支持一个或多个虚拟机(例如,虚拟机1920、1950和1980)的硬件/软件组合的任何设备。例如,每个机器版本1910、1940和1970可以等效于图22的系统2200的一个实例。

[0654] 通常,因为操作系统可以在虚拟机1920、1950和1980之间迁移,所以每个虚拟机1920、1950和1980必须具有相同的VAL。相同的VAL意味着如果复合体(例如,系统1900;图2的CEC200)具有机器版本1910(例如,第一生成机器)和机器版本1940(例如,第二生成机器)并且虚拟机1950正从第一生成机器移动到第二生成机器,虚拟机1950一旦移动就可以执行第一代机器指令。注意,第二代机器指令将不会在第一代机器上工作。还注意,第一和第二生成机器指令可以是分别满足第一和第二生成机器的特定格式化要求的复杂可中断指令

(例如,DEFLATE转换调用指令或来自在加速器上运行的复杂指令集的指令)。

[0655] 系统1900针对每个虚拟机1920、1950和1980的VAL内维护复杂可中断指令的兼容性级别1929、1959和1989。针对跨机器版本1910、1940和1970的最低公分母机器版本和针对复杂可中断指令的本地参数块格式构造每个兼容性级别1929、1959和1989。注意,本地参数块格式是针对机器版本1910、1940和1970中的每一个本地的机器版本构建的。对此,虚拟机1920支持可在机器版本1910(因为这是最早版本)上执行的参数块格式;虚拟机1940支持可在机器版本1910(最低公分母)和机器版本1950(本地)上执行的参数块格式;并且虚拟机1970支持可在机器版本1910(最低公分母)和机器版本1980(本地)上执行的参数块格式。更具体地,因为版本n是最低公分母,所以在每个虚拟机1920、1950和1980上在每个兼容性级别1929、1959和1989中复制分组锁版本n1930。

[0656] 根据一个或多个实施例,兼容性级别1929、1959和1989可以由机器版本1910、1940和1970中的机器生成以执行对应的复杂可中断指令。进一步,最低公分母指示符1931可由机器版本1910、1940和1970中的机器生成以首先执行对应的复杂可中断指令。VAL利用兼容性级别1929来携带该最低公分母指示符1931,使得当生成后续兼容性级别1959和1989时复制分组锁版本n1930。最低公分母指示符1931可以与所示的分组块版本n1930相关联,或者是兼容性级别1929本身的独立部分。根据一个或多个实施例,最低公分母指示符和/或兼容性等级可从机器版本1910、1940和1970中的机器传播以首先执行复杂可中断指令到机器版本1910、1940和1970的剩余数量。

[0657] 图20描绘了根据本发明的实施例的工艺流程2000。图21描绘了根据本发明的实施例的系统2100。系统2100描绘最初包括具有机器版本n、n-1和n-2(分别为2110、2140和2170)的机器,每个机器版本实现不同的参数块格式(分别为解析锁版本n2130、解析锁版本n-12160和解析锁版本n-22190)。根据一个或多个实施例,每个参数块2130、2160和2190可被单独地存储在机器版本n、n-1和n-2内(分别为2110、2140和2170)。

[0658] 过程流程2000可以由系统2100执行。过程流程2000开始于框2010,其中级别n-2的机器(2170)是实现复杂可中断指令(例如,DEFLATE;DEFLATE转换调用指令;来自在机器2170的加速器上运行的复杂指令集的指令)的第一机器。在框2020处,系统2100的虚拟架构级别经由最低公分母指示符2191指示针对复杂可中断指令执行/支持第一机器(例如,汇编程序版本n-22190)的参数块。分组锁版本n-22190格式可被称为兼容性水平或起源格式

[0659] 在框2030处,将具有最低公分母指示符2191的汇编版本n-22190格式传播到当前在系统2100中运行的每一虚拟机。如图21中B箭头所示,与虚拟机2120和2150共享组合版本n-22190格式。现在,在机器级n和n-1上执行的虚拟机2120和2150两者可实现具有解析锁版本n-22190格式的复杂可中断指令。另外,在机器级n上执行的虚拟机2120可实现对应于解析锁版本n2130格式的复杂可中断指令,并且在机器级n-1上执行的虚拟机2150可实现对应于解析锁版本n-12160格式的复杂可中断指令。

[0660] 在框2040,系统2100检测到第一机器离线。如图21中X所示,在级别n-2的机器(2170)离线。在框2050,系统2100检测到新机器在线。例如,级别n+1的机器(2191)上线。在框2060处,系统2100将第一机器的参数块识别为机器版本上的最低公分母。

[0661] 在框2070,系统2100将第一机器的参数块传播到新机器。如图21所示,系统2100现在包括具有机器版本n+1(2191)的机器,其实现参数块格式(组合锁版本n+12193)。如图21

中D箭头所示,与虚拟机2192共享组合版本n-22190格式。由此,当迁移任何虚拟机时,其使用函数的n-2级。

[0662] 本文中的实施例的技术效果和益处包括:当针对复杂指令/函数,每个机器生成最多仅支持两个不同的参数块(其自身的参数块和兼容性级别)时,验证和确认被减少以对照参数块的当前和兼容性版本进行测试。在此的实施例的技术效果和益处进一步包括,对于第一代,其参数块格式表示“起源”格式并且仅需要支持/测试一种格式。由此,任何后续代的机器特定参数块内容都可以针对该机器进行优化,而不会对未来的机器生成产生下游影响。

[0663] 现在转向图22,根据本发明的一个或多个实施例示出了用于实现本文的教导的系统2200。在该实施例中,系统2200具有处理器2201,其可以包括一个或多个中央处理单元(CPU) 2201a、2201b、2201c等。

[0664] 处理器2201(也被称为处理电路、微处理器、计算单元)经由系统总线2202耦合到系统存储器2203和不同其他组件。系统存储器2203包括只读存储器(ROM) 2204和随机存取存储器(RAM) 2205。ROM2204耦合到系统总线2202,并且可以包括基本输入/输出系统(BIOS),其控制系统2200的某些基本功能。RAM是耦接到系统总线2202以供处理器2201使用的读写存储器。

[0665] 图22的系统2200包括硬盘2207,其是可由处理器2201执行的可读的有形存储介质的示例。硬盘2207存储软件2208和数据2209。软件2208被存储为由处理器2201在系统2200上执行的指令(以执行过程,如图21-22的过程流2100、2200)。数据2209包括以不同数据结构组织以支持软件2208的操作和由软件2208的操作使用的定性或定量变量的一组值。

[0666] 图22的系统2200包括互连并支持处理器2201、系统存储器2203、硬盘2207和系统2200的其他部件(例如,外围设备和外部设备)之间的通信的一个或多个适配器(例如,硬盘控制器、网络适配器、图形适配器等)。在本发明的一个或多个实施例中,一个或多个适配器可以连接到经由中间总线桥连接到系统总线2202的一个或多个I/O总线,并且一个或多个I/O总线可以利用公共协议,例如外围组件互连(PCI)。

[0667] 如图所示,系统2200包括将键盘2221、鼠标2222、扬声器2223和麦克风2224互连到系统总线2202的接口适配器2220。系统2200包括将系统总线2202与显示器2231互连的显示适配器2230。显示适配器2230(和/或处理器2201)可以包括图形控制器以提供图形性能,诸如GUI2232的显示和管理。通信适配器2241将系统总线2202与网络2250互连,从而使得系统2200能够与其他系统、设备、数据和软件(如服务器2251和数据库2252)通信。在本发明的一个或多个实施例中,软件2208和数据2209的操作可以由服务器2251和数据库2252在网络2250上实现。例如,网络2250、服务器2251、和数据库2252可以组合以提供软件2208和数据2209的内部迭代,作为平台作为服务、软件作为服务、和/或基础设施作为服务(例如,作为分布式系统中的web应用)。

[0668] 因此,如图22中配置的,软件2208和数据2209(例如,系统2200)的操作必须以处理器2201和/或服务器2251的计算能力为根,以克服和解决工作负荷在不同机器的机器之间的当代迁移的在此描述的缺点。就这一点而言,软件2208和数据2209通过维持多个机器生成上的复杂功能的兼容性(从而减少验证和验证并且优化任何后续生成的机器特定参数块内容而不产生下游影响)来改进系统2200的处理器2201和/或服务器2251的计算操作。

[0669] 在此参考相关附图描述本发明的不同实施例。在不脱离本发明的范围的情况下，可以设计本发明的替代实施例。在以下描述和附图中的元件之间阐述了各种连接和位置关系(例如，上方、下方、相邻等)。除非另有说明，这些连接和/或位置关系可以是直接的或间接的，并且本发明在这方面并不意图进行限制。因而，实体的耦合可以指直接或间接耦合，并且实体之间的位置关系可以是直接或间接位置关系。此外，本文所述的各种任务和工艺步骤可并入到具有本文未详细描述的增加步骤或功能的更全面的程序或工艺中。

[0670] 以下定义和缩写用于解释权利要求书和说明书。如在此使用的，术语“包含”(comprises)、“包含”(comprising)、“包括”(includes)、“包括”(including)、“具有”(has)、“具有”(having)、“含有”(contains)或“含有”(containing)或其任何其他变体旨在覆盖非排他性的包含。例如，包含一系列元素的组合物、混合物、工艺、方法、制品或设备不一定仅限于那些元素，而是可包括未明确列出的或此类组合物、混合物、工艺、方法、制品或设备固有的其他元素。

[0671] 另外，术语“示例性”在此用于意指“充当实例、实例或说明。”在此描述为“示范性”的任何实施例或设计不一定被解释为比其他实施例或设计优选或有利。术语“至少一个”和“一个或多个”可以被理解为包括大于或等于一的任何整数，即，一个、两个、三个、四个等。术语“多个”可以被理解为包括大于或等于二的任何整数，即，两个、三个、四个、五个等。术语“连接”可以包括间接“连接”和直接“连接”两者。

[0672] 术语“约”、“基本上”、“大约”及其变体旨在包括与基于在提交本申请时可用的设备的具体量的测量相关联的误差程度。例如，“约”可以包括给定值的 $\pm 8\%$ 或 5% 、或 2% 的范围。

[0673] 为了简洁起见，与制作和使用本发明的多个方面相关的常规技术可以或可以不在本详细描述。具体地，用于实现本文描述的不同技术特征的计算系统和特定计算机程序的不同方面是公知的。因而，为了简洁起见，许多常规实现细节在此仅简要地提及，或者完全省略，而不提供公知的系统和/或过程细节。

[0674] 本发明可以是任何可能的集成技术细节水平的系统、方法和/或计算机程序产品。所述计算机程序产品可包含上面具有计算机可读程序指令的计算机可读存储媒体(或媒体)，所述计算机可读程序指令用于致使处理器执行本发明的方面。计算机可读存储媒质可以是保留和存储指令以供指令执行设备使用的有形设备。计算机可读存储媒质可以是例如但不限于电子存储设备、磁存储设备、光存储设备、电磁存储设备、半导体存储设备或前述各项的任何合适的组合。计算机可读存储媒质的更具体例子的非穷举列表包括以下：便携式计算机盘、硬盘、随机存取存储器(RAM)、只读存储器(ROM)、可擦除可编程只读存储器(EPROM或闪存)、静态随机存取存储器(SRAM)、便携式致密盘只读存储器(CD-ROM)、数字通用盘(DVD)、记忆棒、软盘、机械编码设备(诸如穿孔卡片)或具有记录在其上的指令的凹槽中的凸起结构)，以及上述的任意合适的组合。如本文中所使用的计算机可读存储媒质不应被解释为瞬态信号本身，诸如无线电波或其他自由传播的电磁波、通过波导或其他传输媒质传播的电磁波(例如，通过光纤电缆的光脉冲)、或通过导线传输的信号。

[0675] 本文所述的计算机可读程序指令可从计算机可读存储介质下载到相应的计算/处理设备，或经由网络(例如，互联网、局域网、广域网和/或无线网络)下载到外部计算机或外部存储设备。网络可以包括铜传输电缆、光传输光纤、无线传输、路由器、防火墙、交换机、网

关计算机和/或边缘服务器。每个计算/处理设备中的网络适配器卡或网络接口从网络接收计算机可读程序指令,并转发计算机可读程序指令以存储在相应计算/处理设备内的计算机可读存储介质中。

[0676] 用于执行本发明的操作的计算机可读程序指令可以是汇编指令,指令集架构(ISA)指令、机器指令、机器相关指令、微代码、固件指令、状态设置数据,集成电路的配置数据,或以一种或多种编程语言的任何组合编写的源代码或目标代码,包括面向对象的Smalltalk、C++等编程语言,以及过程式编程语言,如“C”编程语言或类似的编程语言。计算机可读程序指令可完全在用户的计算机上执行、部分在用户的计算机上执行、作为独立软件包执行、部分在用户的计算机上部分在远程计算机上执行、或者完全在远程计算机或服务器上执行。在后一种情形中,远程计算机可以通过任何类型的网络(包括局域网(LAN)或广域网(WAN))连接到用户的计算机,或者可以连接到外部计算机(例如,通过使用互联网服务提供商的互联网)。在一些实施例中,电子电路(包括例如可编程逻辑电路、现场可编程门阵列(FPGA)或可编程逻辑阵列(PLA))可以通过利用计算机可读程序指令的状态信息来执行计算机可读程序指令以使电子电路个性化,以便执行本发明的方面。

[0677] 本文中参考根据本发明的实施例的方法、设备(系统)和计算机程序产品的流程图说明和/或框图描述本发明的方面。应当理解,流程图和/或框图的每个框以及流程图和/或框图中各框的组合,都可以由计算机可读程序指令来实现。

[0678] 这些计算机可读程序指令可以被提供给通用计算机的处理器,专用计算机或其他可编程数据处理装置,以产生机器,其通过计算机或其他可编程数据处理装置的处理器执行,创建用于实现在流程图和/或框图的一个或多个框中指定的功能/动作的装置。这些计算机可读程序指令还可存储在可指导计算机的计算机可读存储介质中,可编程数据处理装置,和/或以特定方式起作用的其他设备,使得具有存储在其中的指令的计算机可读存储介质包括制品,该制品包括实现流程图和/或框图中的一个或多个框中规定的功能/动作的各方面的指令。

[0679] 计算机可读程序指令还可以加载到计算机、其他可编程数据处理装置上,或使得在计算机上执行一系列操作步骤的其他装置,其他可编程装置或其他设备,以产生计算机实现的过程,使得在计算机上执行的指令,其他可编程装置或其他设备实现流程图和/或框图中的一个或多个框中规定的功能/动作。

[0680] 附图中的流程图和框图图示了根据本发明的不同实施例的系统、方法和计算机程序产品的可能实现的架构、功能和操作。对此,流程图或框图中的每个框可以代表模块、段或指令的一部分,其包括用于实现规定的逻辑功能的一个或多个可执行指令。在一些替代实施例中,框中所标注的功能可以不以图中所标注的次序发生。例如,取决于所涉及的功能,连续示出的两个框实际上可以基本上同时执行,或者这些框有时可以以相反的顺序执行。还将注意的是,框图和/或流程图中的每个框、以及框图和/或流程图中的框的组合可以由基于专用硬件的系统来实现,所述基于专用硬件的系统执行指定的功能或动作或执行专用硬件与计算机指令的组合。

[0681] 已经出于说明的目的呈现了本发明的不同实施例的描述,但并不旨在是穷尽性的或局限于所披露的实施例。在不背离所描述的实施例的范围和精神的情况下,许多修改和变化对本领域的普通技术人员而言将是显而易见的。选择本文中所使用的术语以最佳地解

释实施例的原理、实际应用或对市场中所发现的技术的技术改进,或使得所属领域的其他普通技术人员能够理解本文中所描述的实施例。

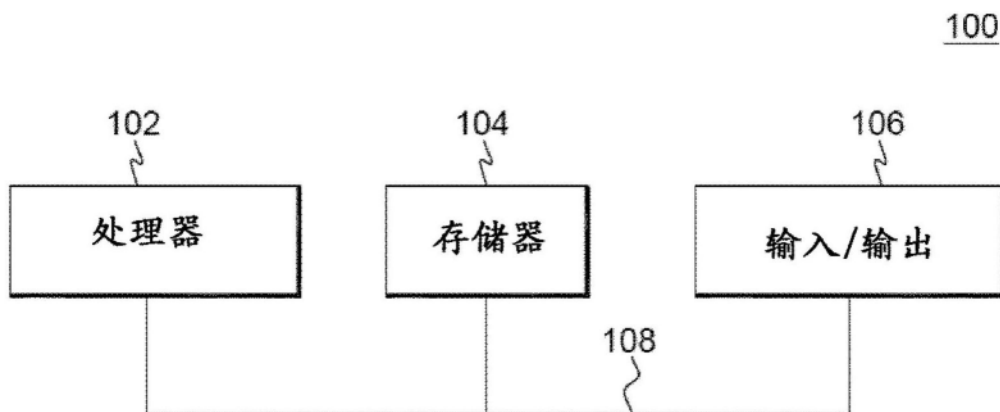


图1A

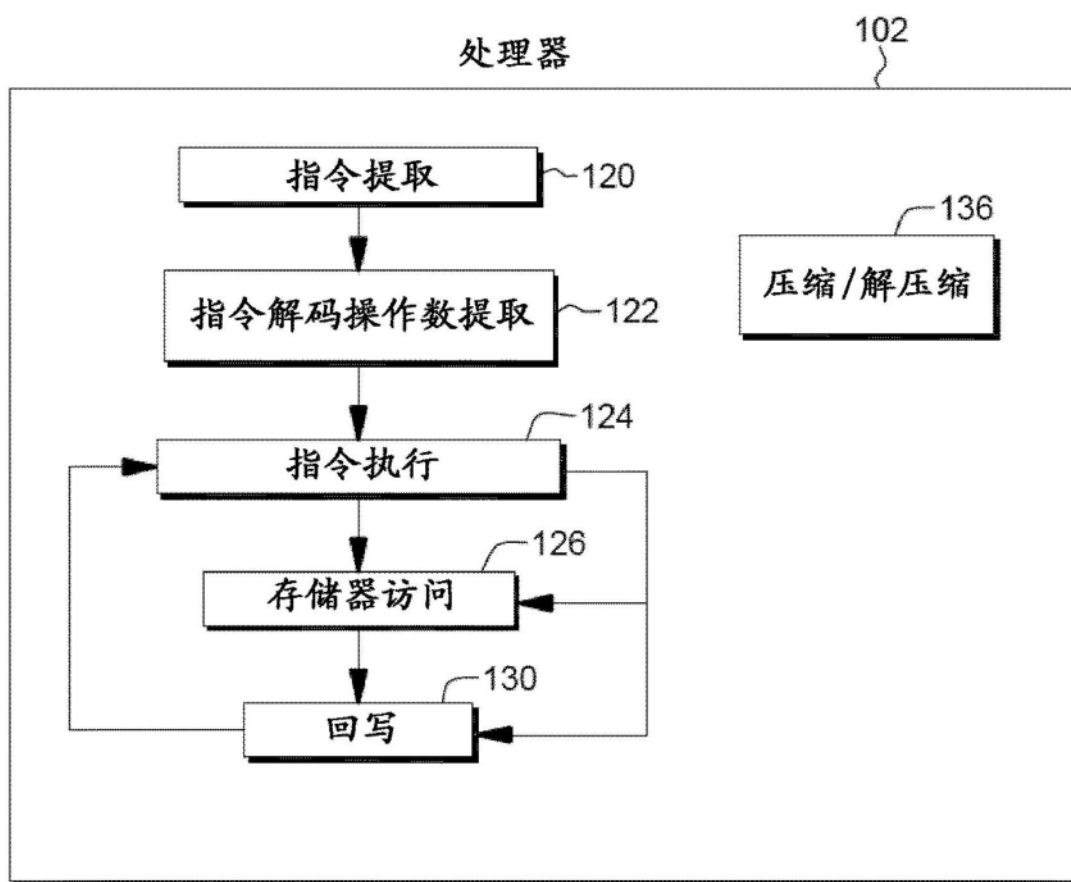


图1B

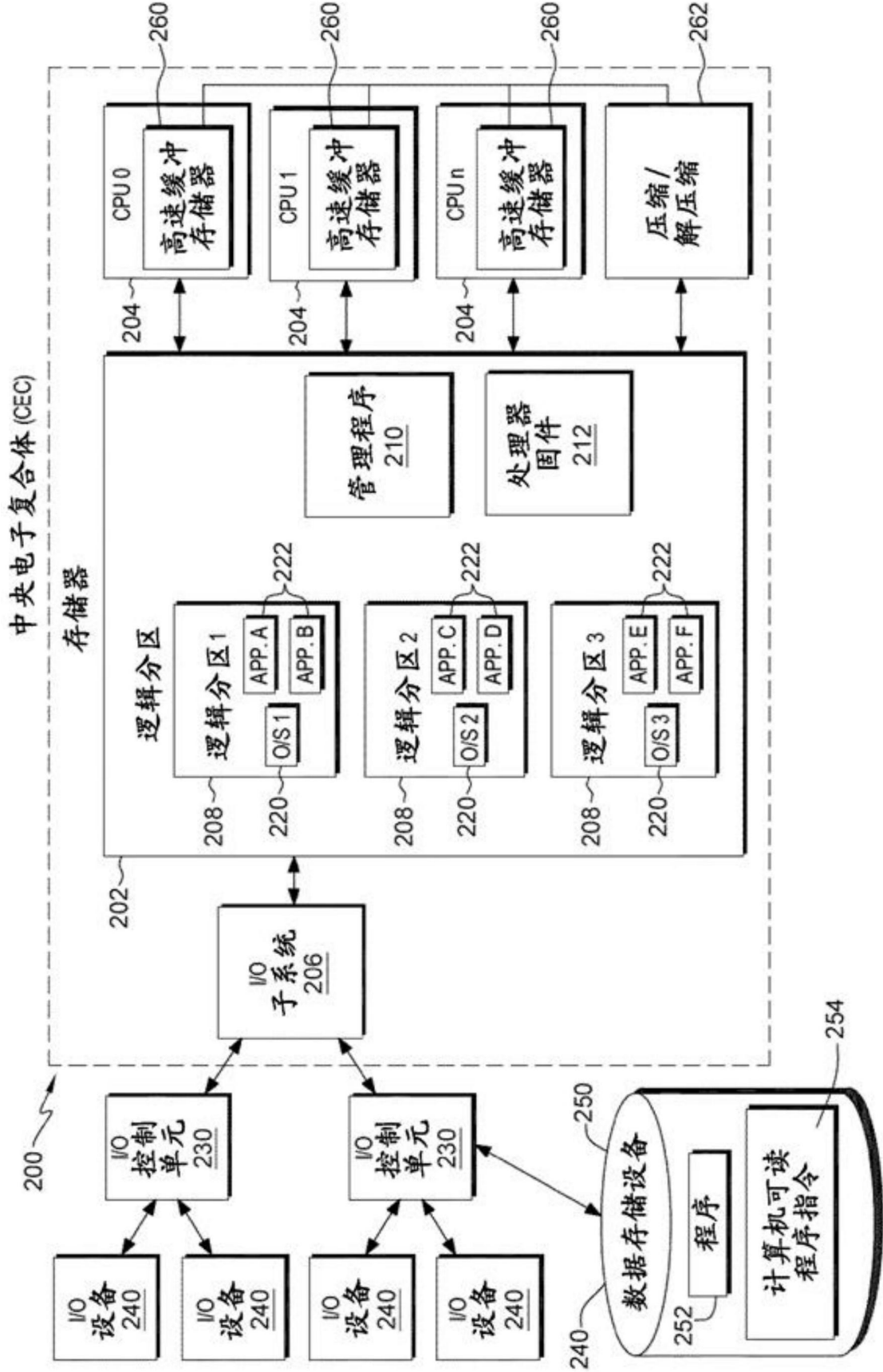


图2

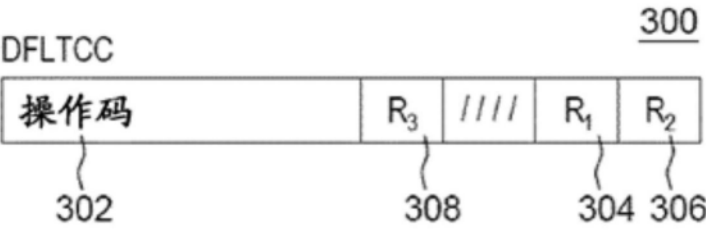


图3A

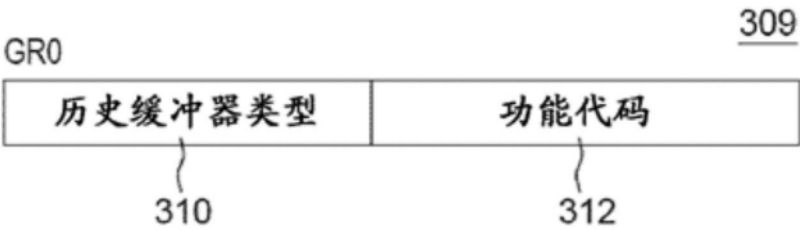


图3B

	代码	功能	参数块大小 (字节)
313	0	DFLTCC - QAF	32
315	1	DFLTCC - GDHT	384
317	2	DFLTCC - CMPR	1536
319	4	DFLTCC - XPND	1536

图3C

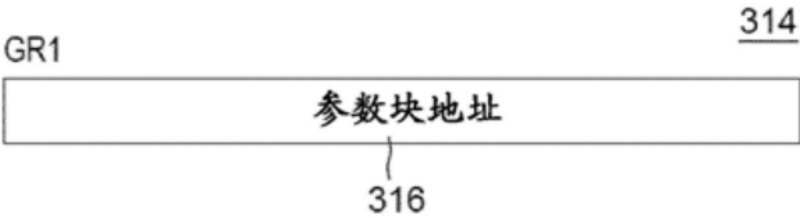


图3D

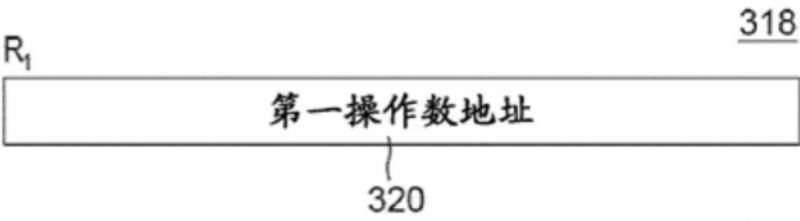


图3E

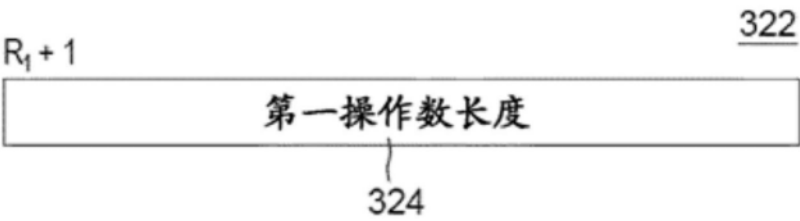


图3F

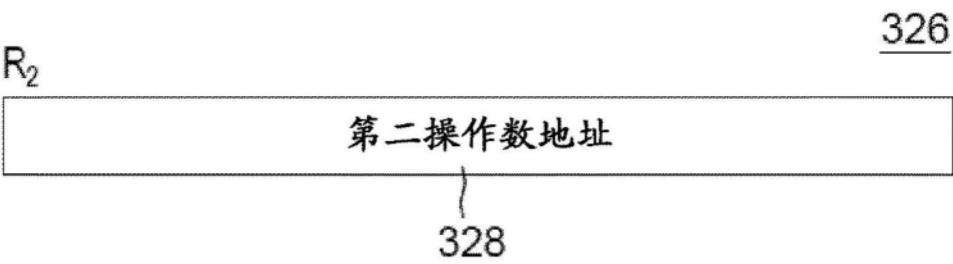


图3G

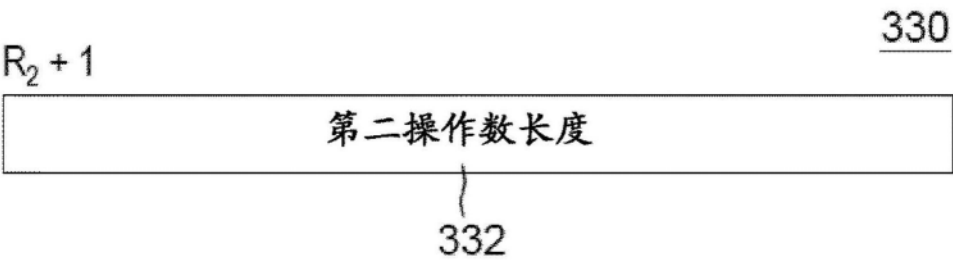


图3H

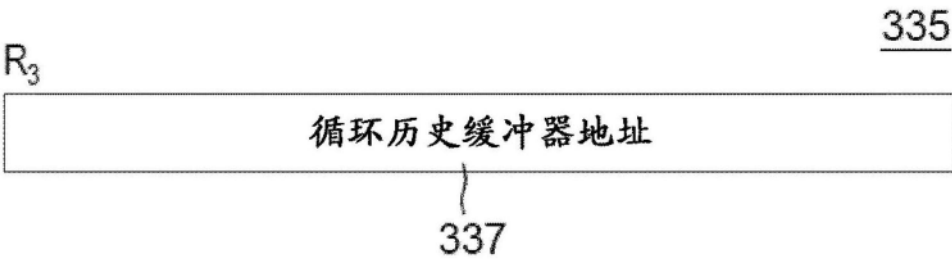


图3I



图3J

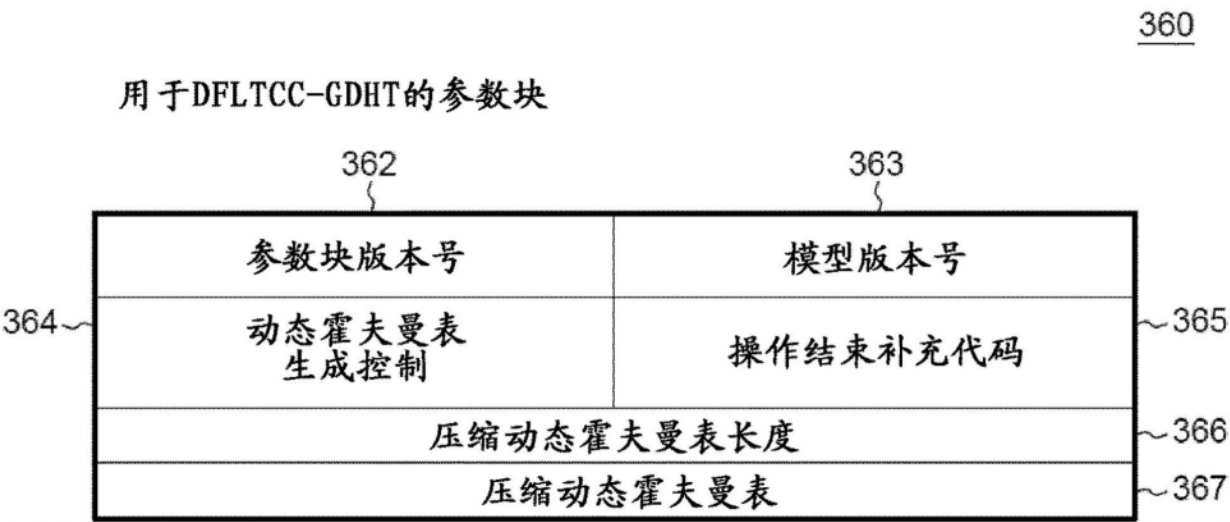


图3K

370

用于DFL TCC-CMPR+DFL TCC-XPND的参数块

参数块版本号 362					模型版本号 363			继续标记 373		
新任务 374	校验值 类型 375	霍夫曼表 类型 376	块连续 标志 377	块关闭 控制 378	块报头 最终 379	DHTGC 364	子字节 边界 381	操作结束 补充代码 365	不完整 功能状态 383	不完整 函数长度 384
历史长度 385					历史偏移 386					
校验值 387			块结束符号 388			块结束长度 389				
压缩动态霍夫曼表长度 366										
压缩动态霍夫曼表 367										
继续状态缓冲器 392										

图3L

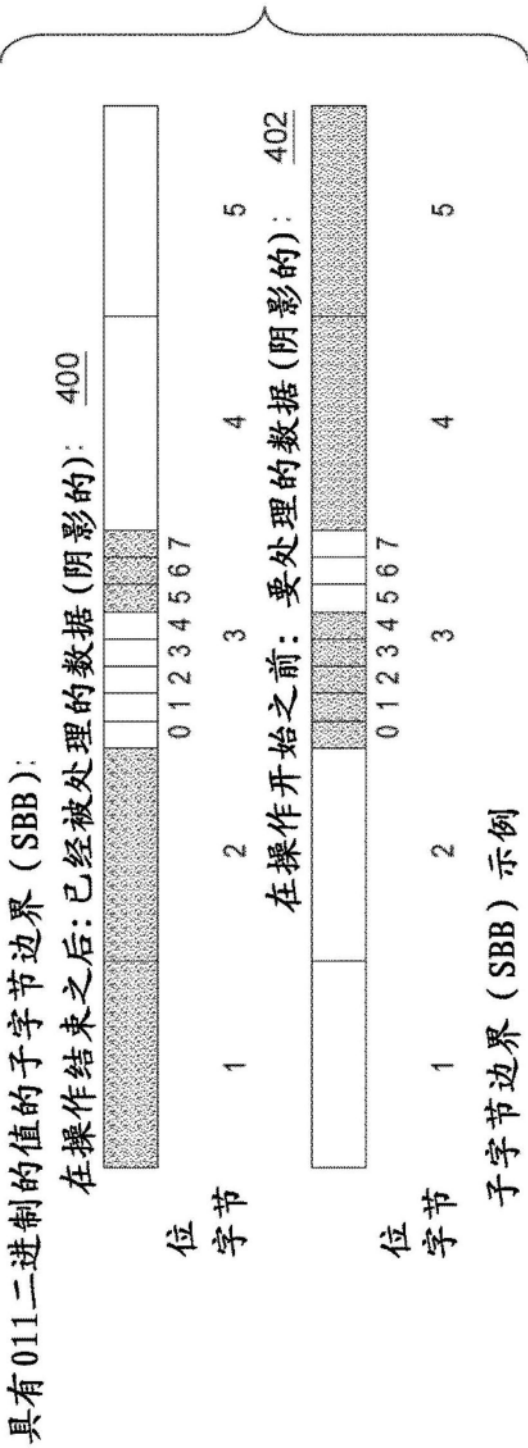


图4

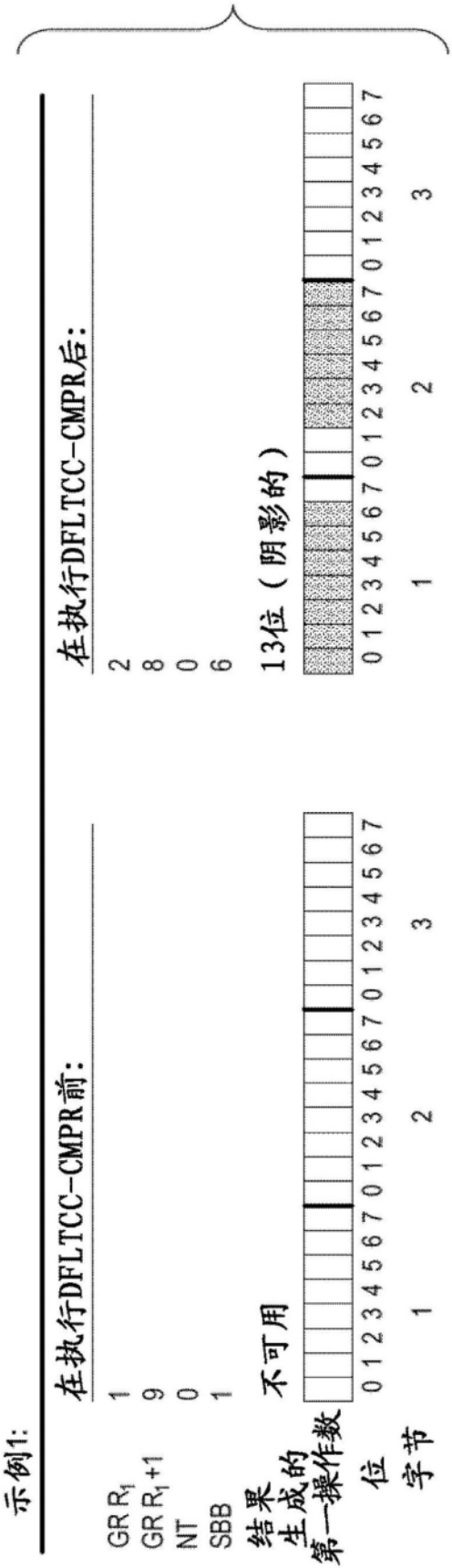


图5A

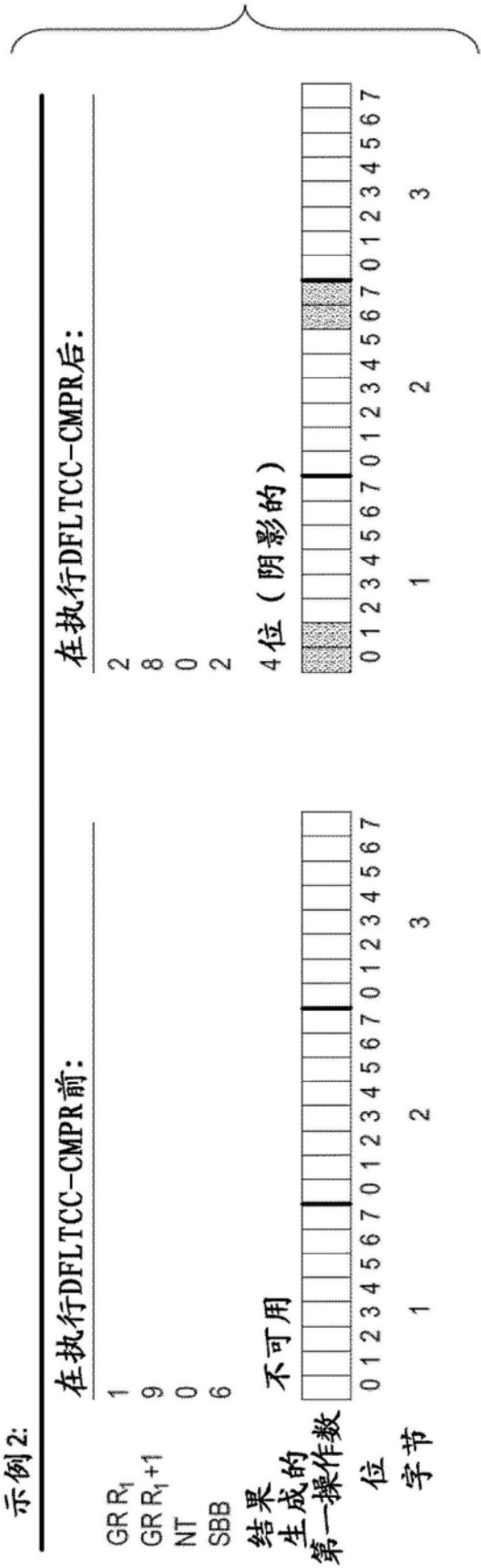


图5B

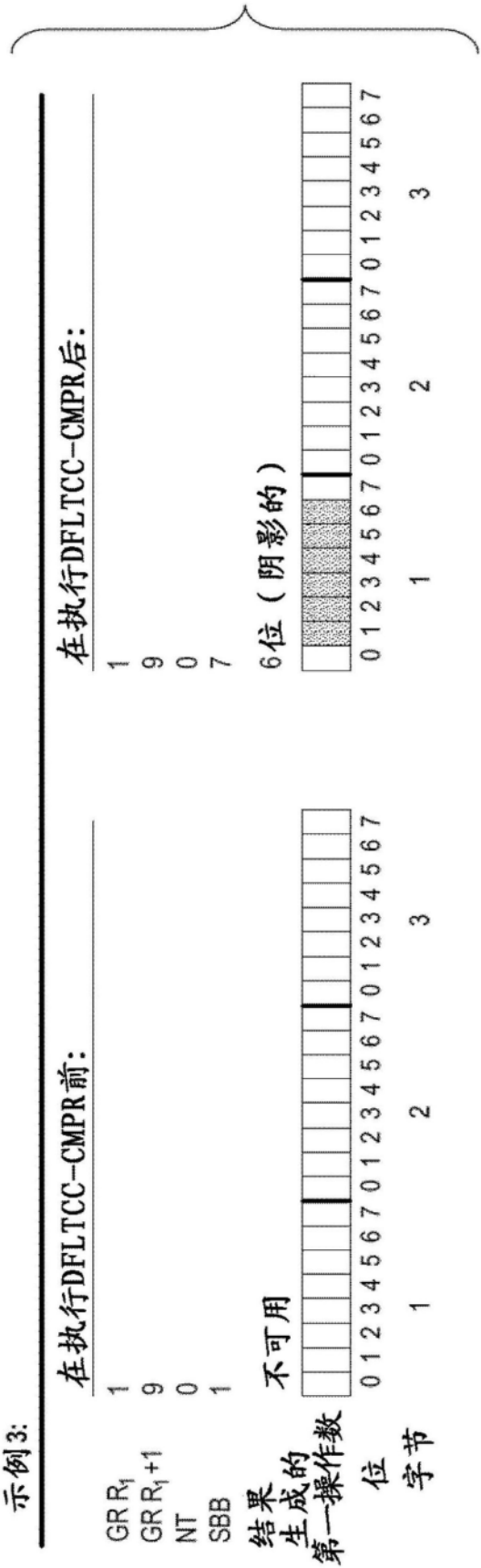


图5C

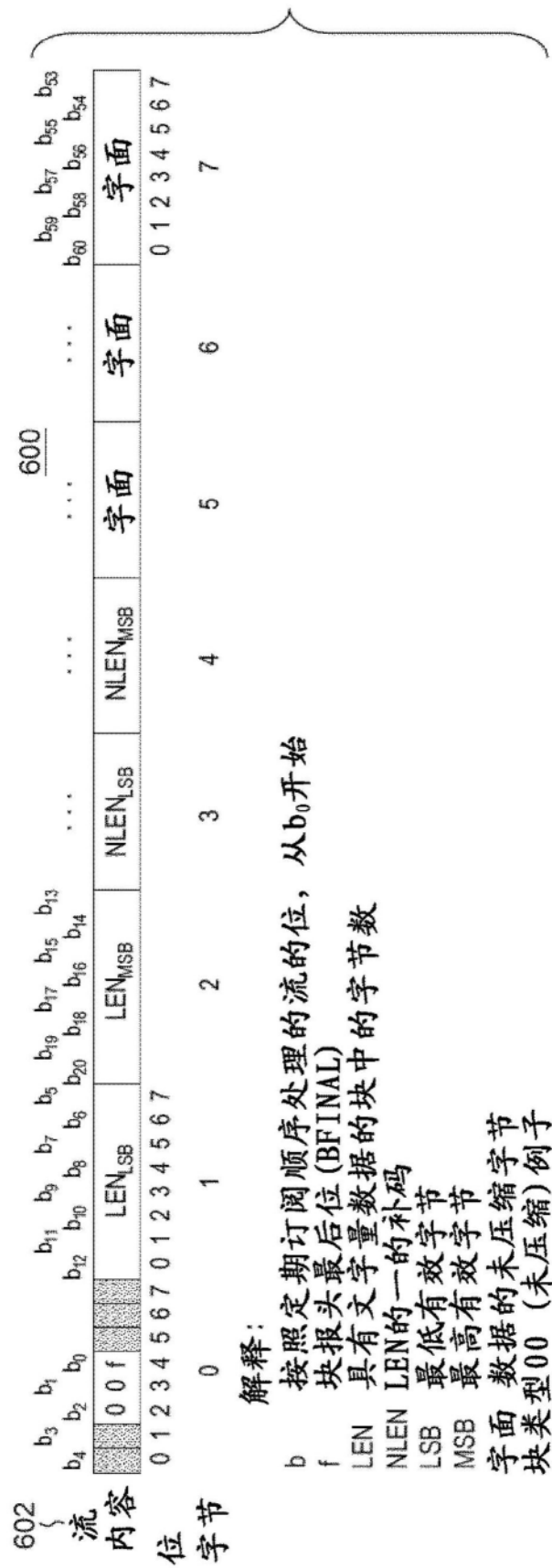


图6

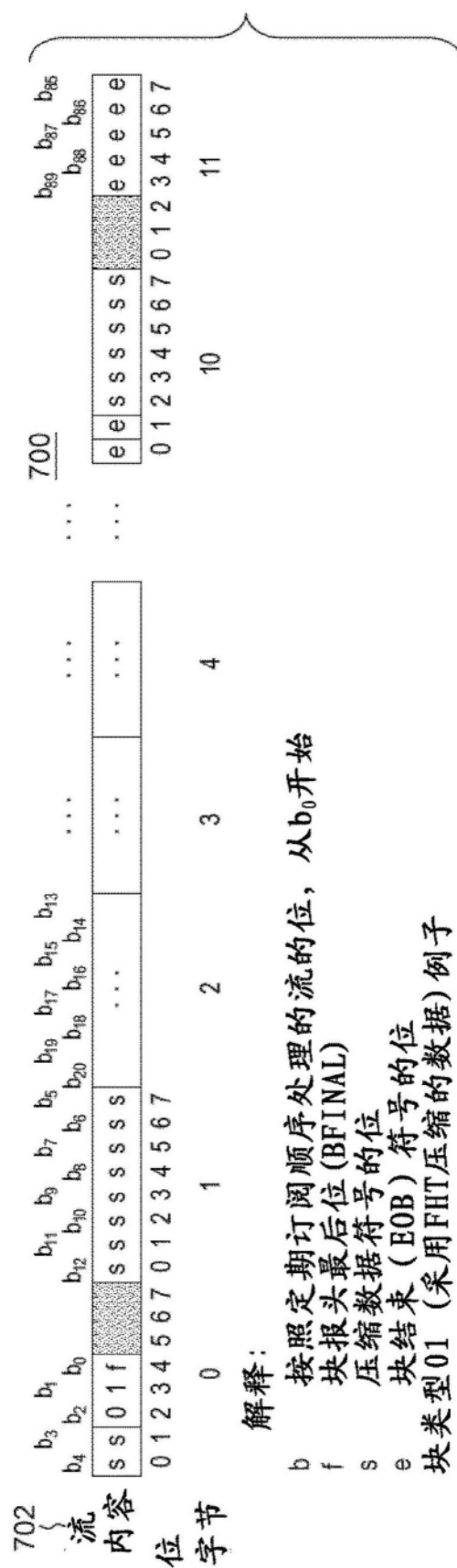


图7

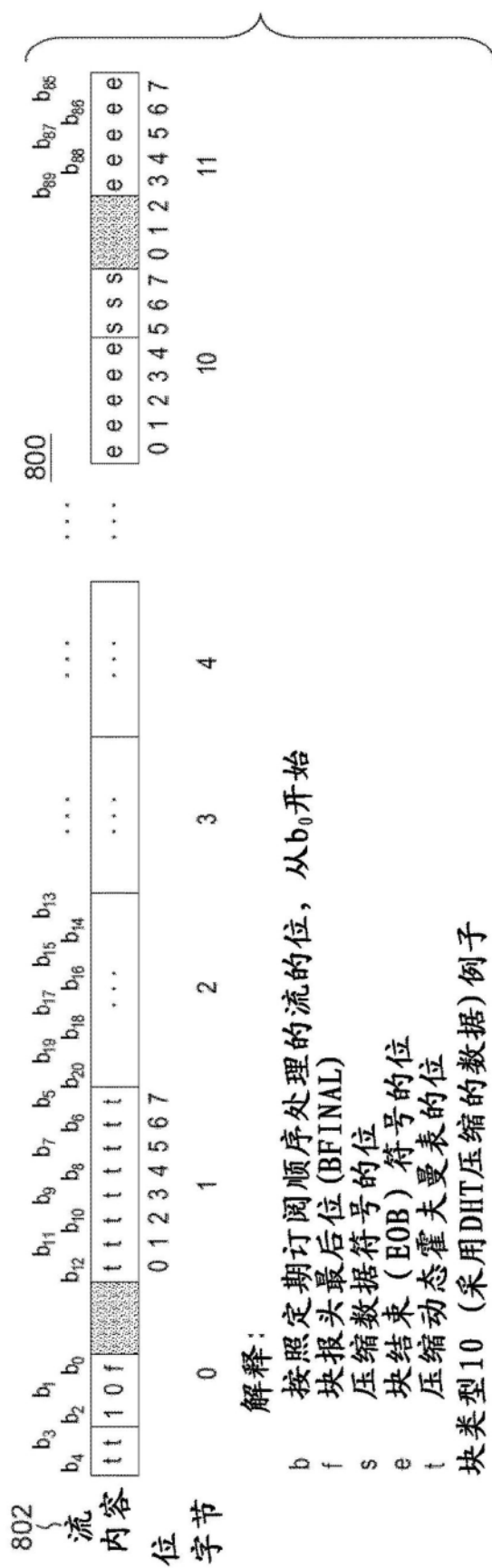


图8

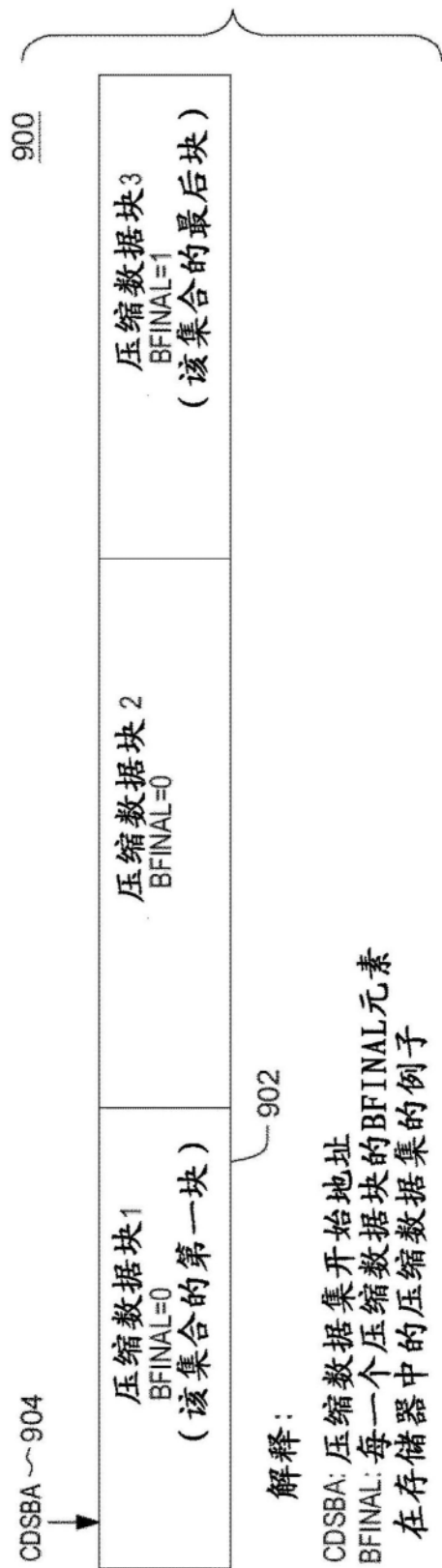
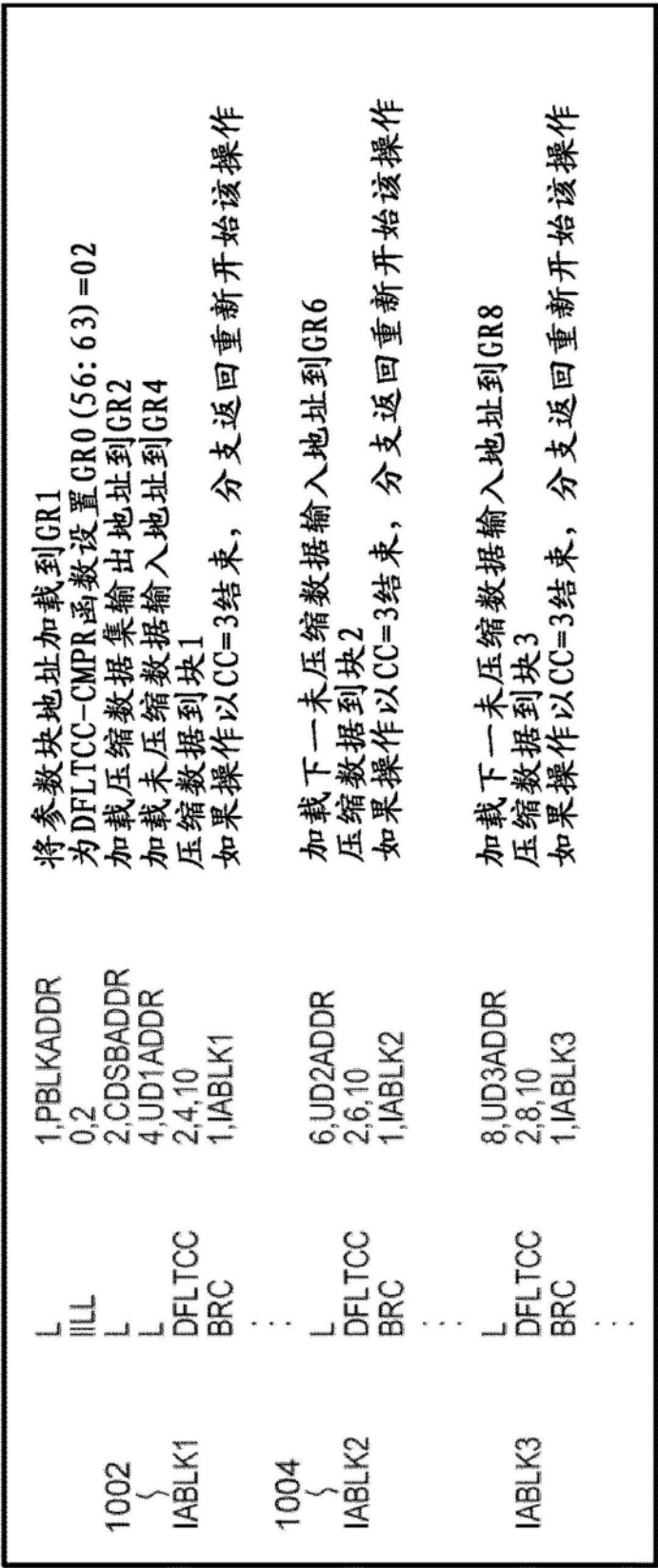


图9

1000



将数据压缩成/压缩数据集的三个块的程序的例子

图10

参数块字段 1100	操作开始的字段值 ~1102	当条件码1、2或3被 设置(同样在操作 重新开始时的值)时, 在操作结束时的字段值 ~1104	当条件码0被设置时 在操作结束时的字段值 ~1106
新任务(NT)	1(由程序指定)	0	0
继续标记(CF)	由程序初始化为零	1	0
连续状态缓冲器(CSB)	由程序初始化为全零	在操作结束时记录的内部 状态数据, 其为当操作后来 重新开始时要求	未定义
校验值	从NT为一, 作为 初始值对待	从操作开始以及重新开始 该操作的开始值, 从处理的 数据一部分生成的值	由操作以及后续操作 开始值处理的全部 数据生成的值
子字节边界(SBB)	从NT为一, 作为 等于零对待	当操作后来重新开始, 应用到由操作处理的 最后字节和第一字节	对后续操作, 应用到 由操作处理的最后 字节和第一个字节

在集合的第一压缩数据块上操作的DFLTCC-CMPR函数的参数块内容

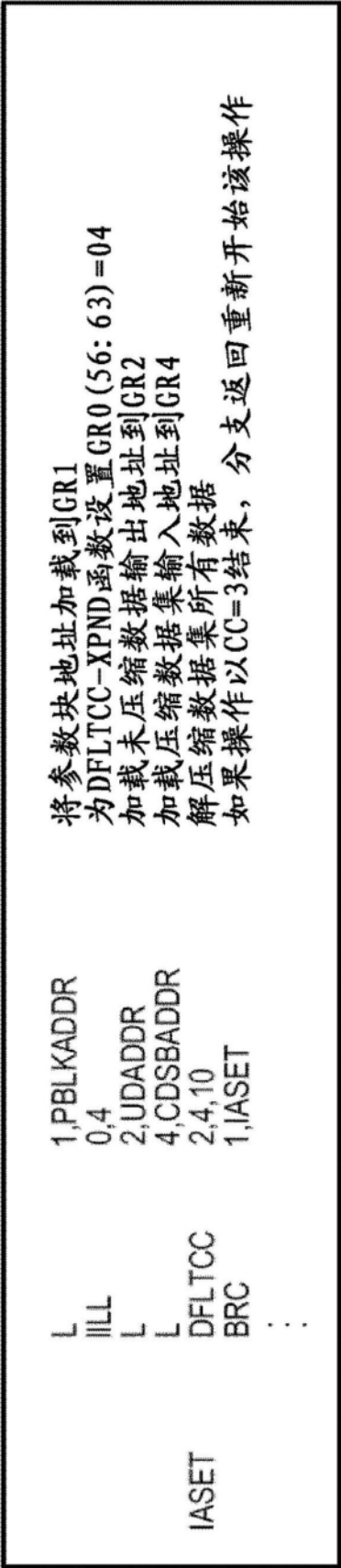
图11

参数块字段	操作开始的 字段值	当条件码1、2或3 被设置（同样的值）时， 重新开始时在操作结束时的 字段值	当条件码0被 设置时在操作 结束时的字段值
新任务 (NT)	0 (由程序指定)	0	0
继续标记 (CF)	由程序初始化为零	1	0
连续状态缓冲器 (CSB)	由程序初始化为全零	在操作结束时记录的 内部状态数据，当操作 后来重新开始时要求	未定义
校验值	由前面操作生成的值 (由程序保持)	从目前操作开始以及 重新开始时前操作的 开始值，从处理的 数据一部分生成的值	由操作以及后续操作 开始值处理的全部 数据生成的值
子字节边界 (SBB)	与由前面操作处理的 最后字节关联的值 (由程序保持)	当当前操作后来重新开始， 应用到由当前操作处理的 最后字节和第一字节	对后续操作， 应用到由操作 处理的最后一个 字节和第一个

在集成的第二压缩数据块上操作的DFLTCC-CMPR函数的参数块内容

图12

1300



从压缩数据集解压数据的程序的样本的示例

图13

DFLTCC-CMPR 执行编号1之前:

HL=0

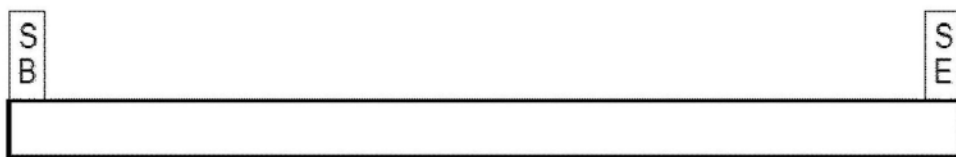


图14A

在 DFLTCC-CMPR 执行编号1期间:

BP=10K

DFLTCC-CMPR 执行编号1之后:

HL=10K

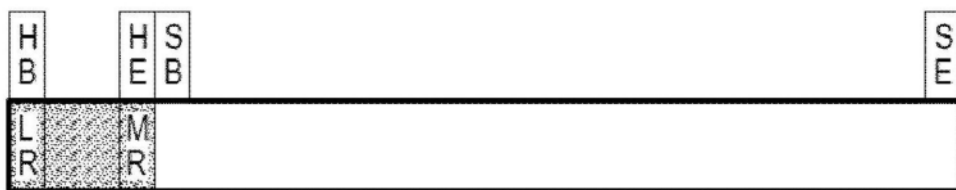


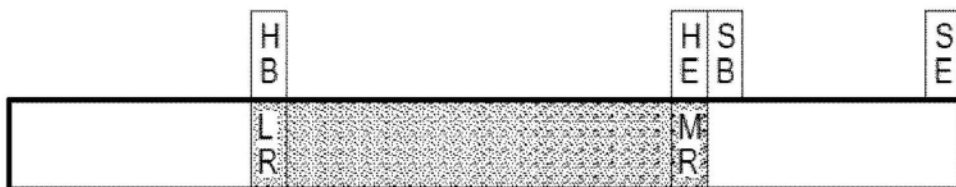
图14B

在 DFLTCC-CMPR 执行编号2期间:

BP=40K

DFLTCC-CMPR 执行编号2之后:

HL=32K



解释:

阴影区域: 历史字节

SB 源开始地址: 由 R_2 指定位置SE 源结束地址: 由 $R_2 + (R_2 + 1) - 1$ 指定位置HB 历史开始地址: 由 $R_2 - HL$ 指定位置HE 历史结束地址: 由 $R_2 - 1$ 指定位置

LR 最近最少处理的历史字节

MR 最近最多处理的历史字节

BP 处理的未压缩数据字节

HL 历史长度

在执行DFLTCC-CMPR多次之后和之前的在线历史

图14C

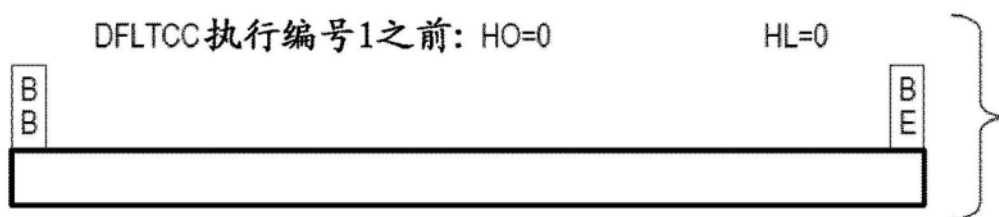


图15A

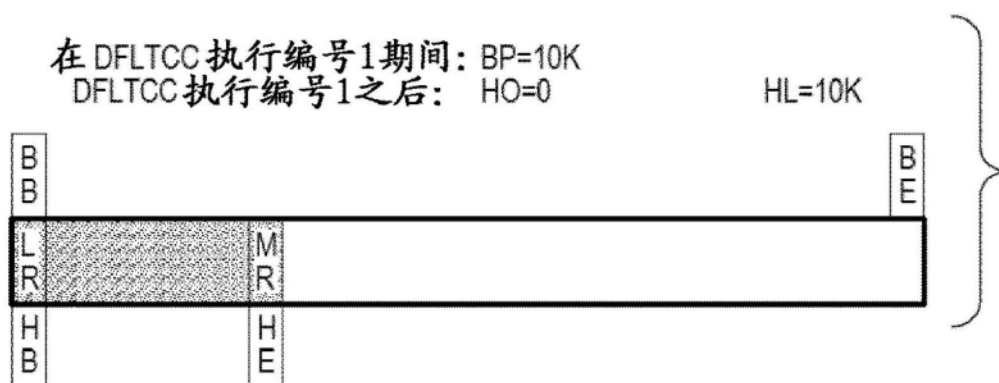


图15B

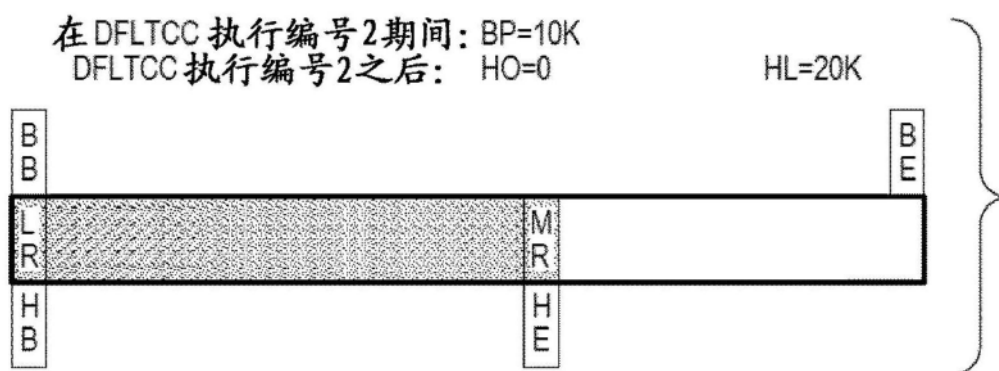


图15C

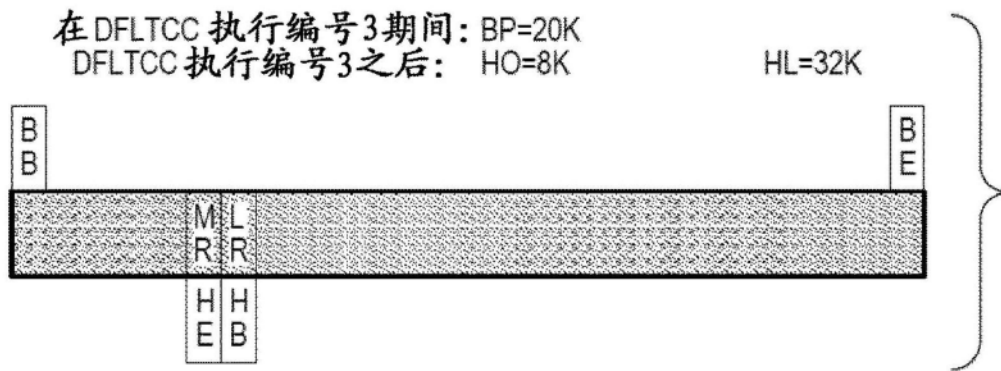


图15D

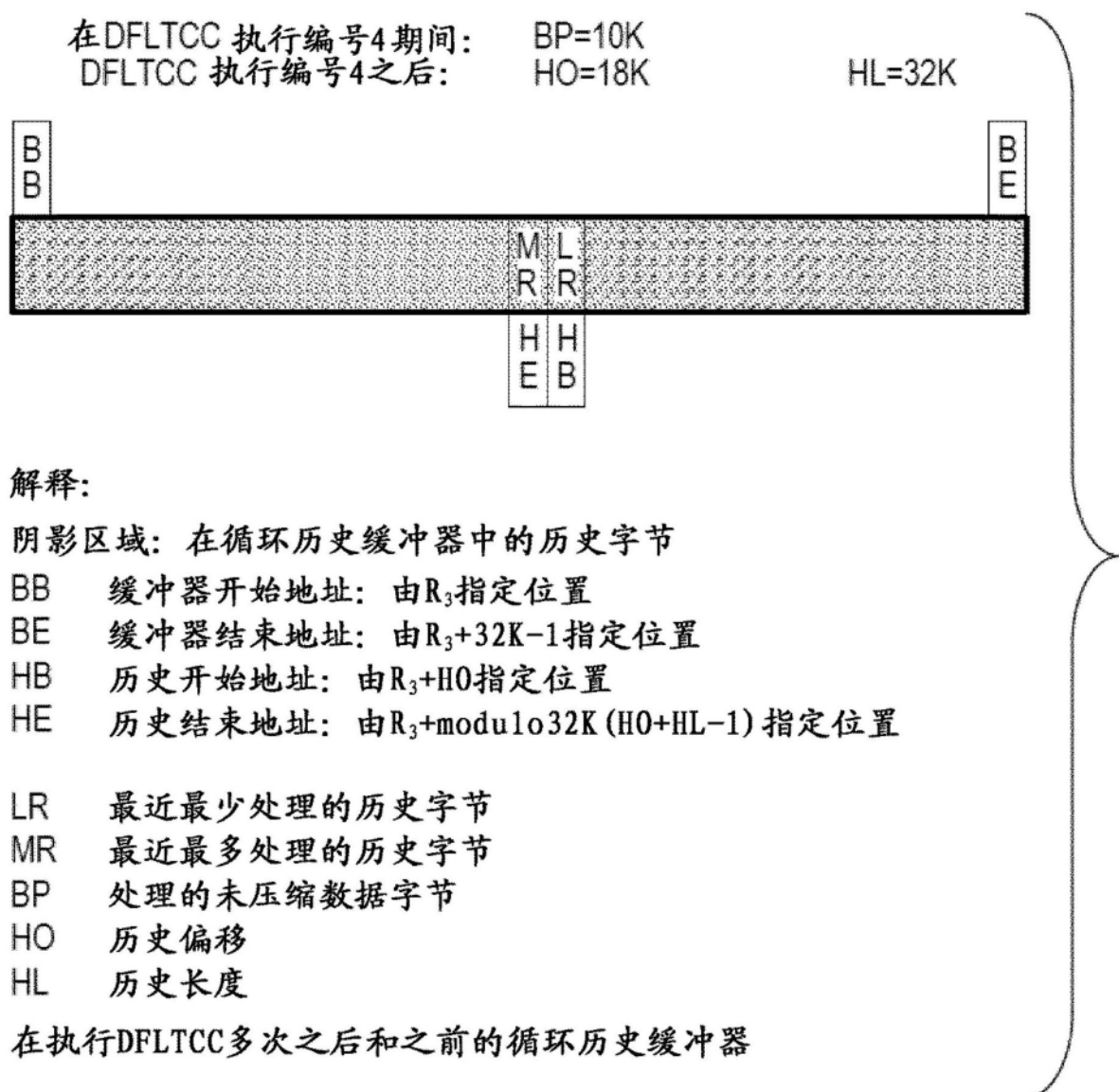


图15E

DFLTCC-XPND 执行编号1之前:

HL=0

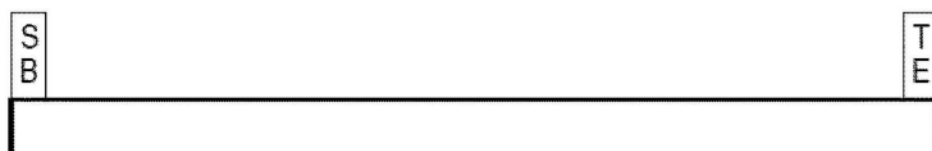


图16A

在DFLTCC-XPND 执行编号1期间:
DFLTCC-XPND 执行编号1之后:

BP=10K

HL=10K

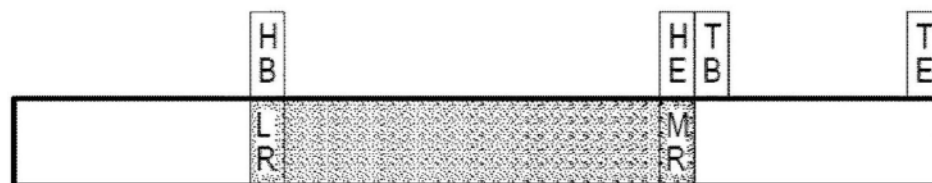


图16B

在DFLTCC-XPND 执行编号2期间:
DFLTCC-XPND 执行编号2之后:

BP=40K

HL=32K



解释:

阴影区域: 历史字节

TB 目标开始地址: 由R1指定位置

TE 目标结束地址: 由 $R_1 + (R_1 + 1) - 1$ 指定位置HB 历史开始地址: 由 $R_1 - HL$ 指定位置HE 历史结束地址: 由 $R_1 - 1$ 指定位置

LR 最近最少处理的历史字节

MR 最近最多处理的历史字节

BP 处理的未压缩数据字节

HL 历史长度

在执行DFLTCC-XPND多次之后和之前的在线历史

图16C

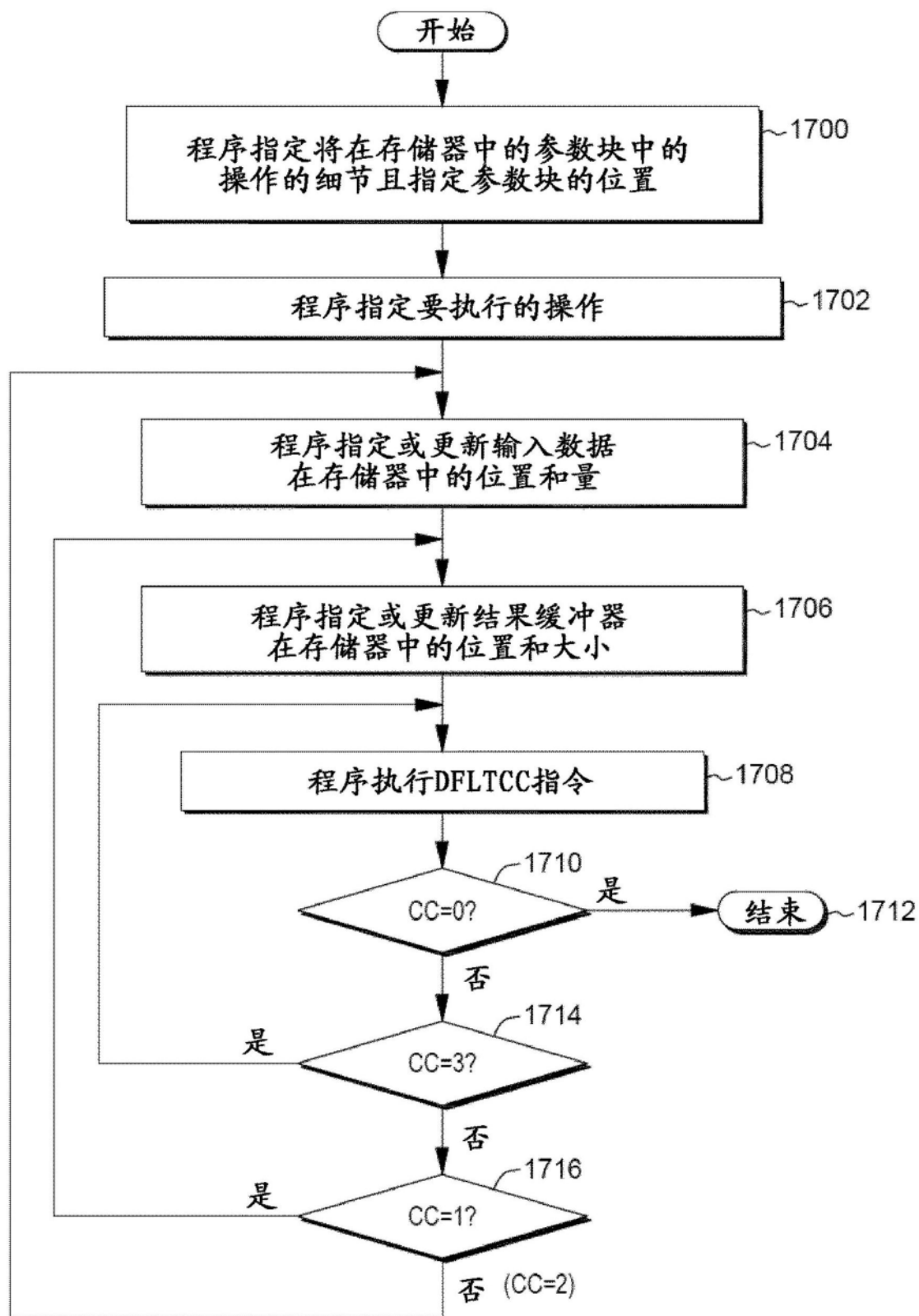


图17

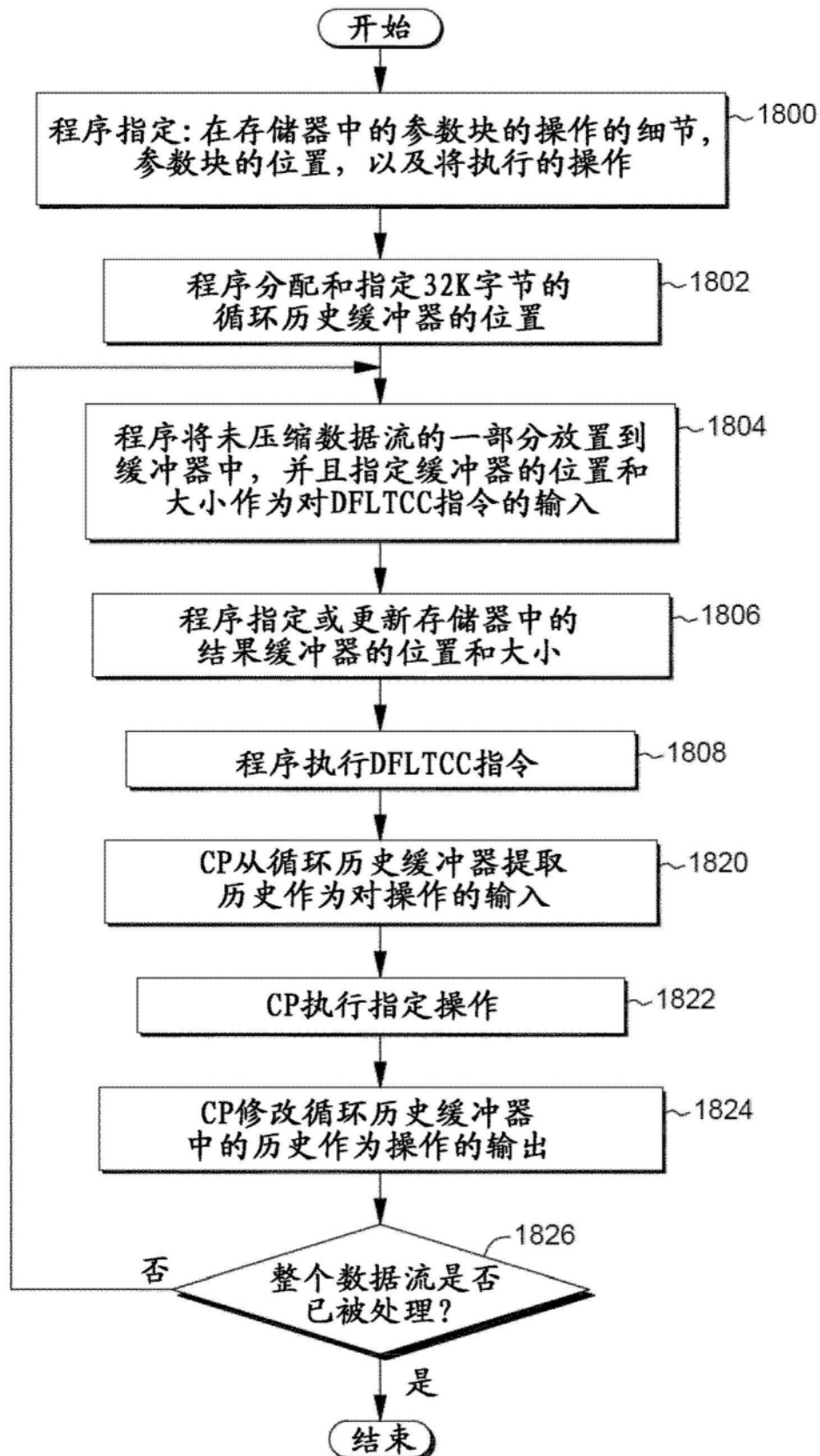


图18

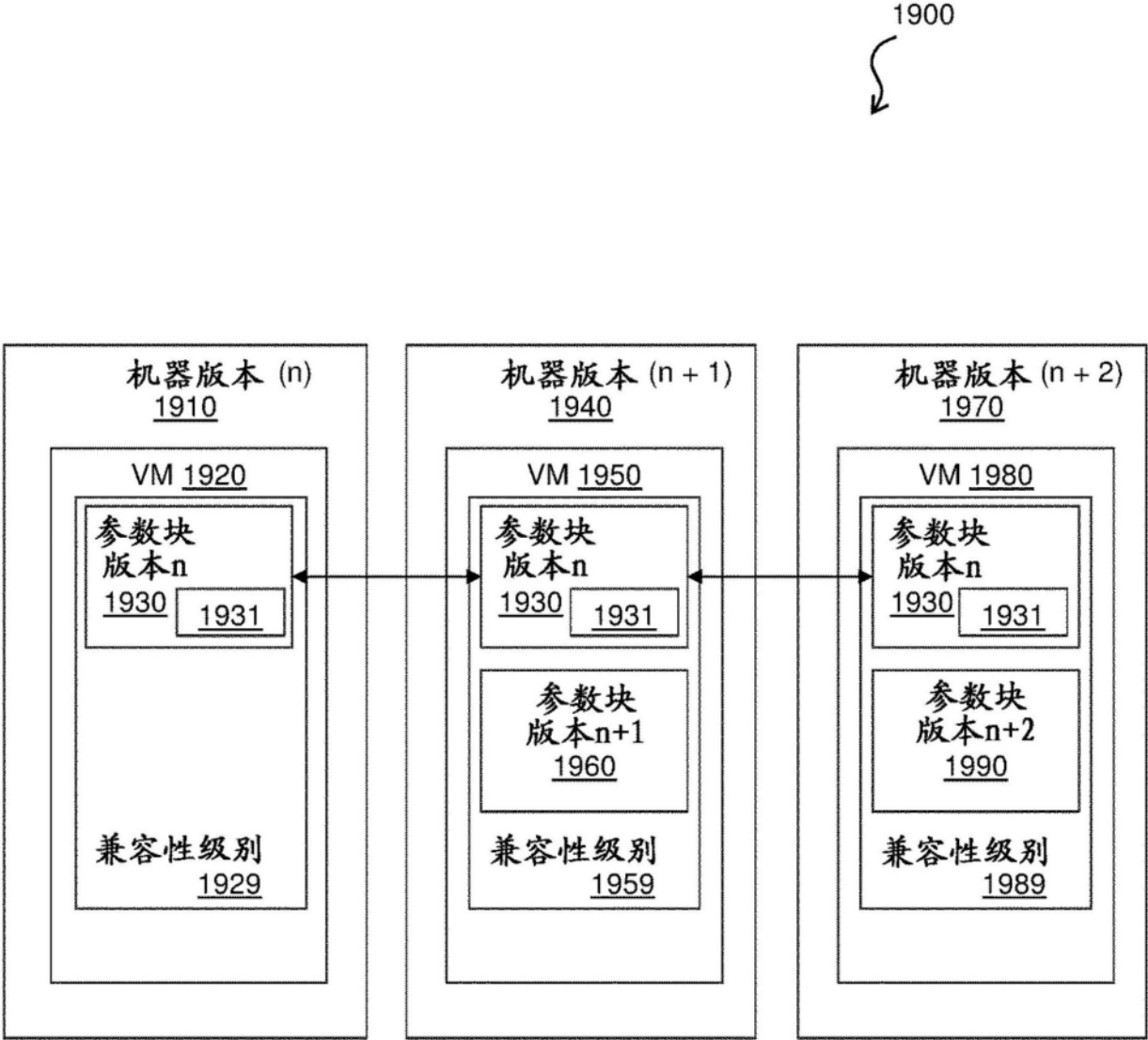


图19

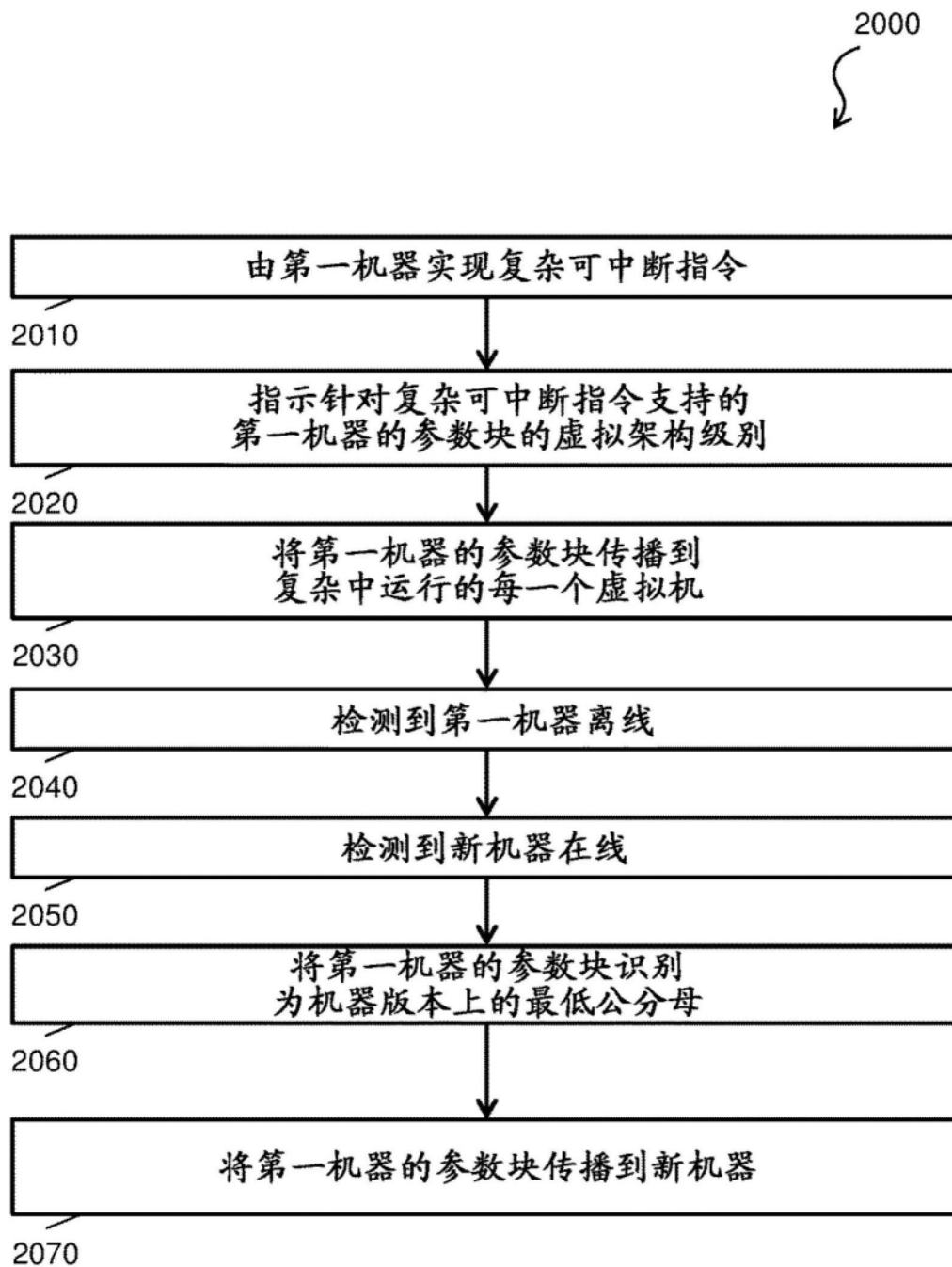


图20

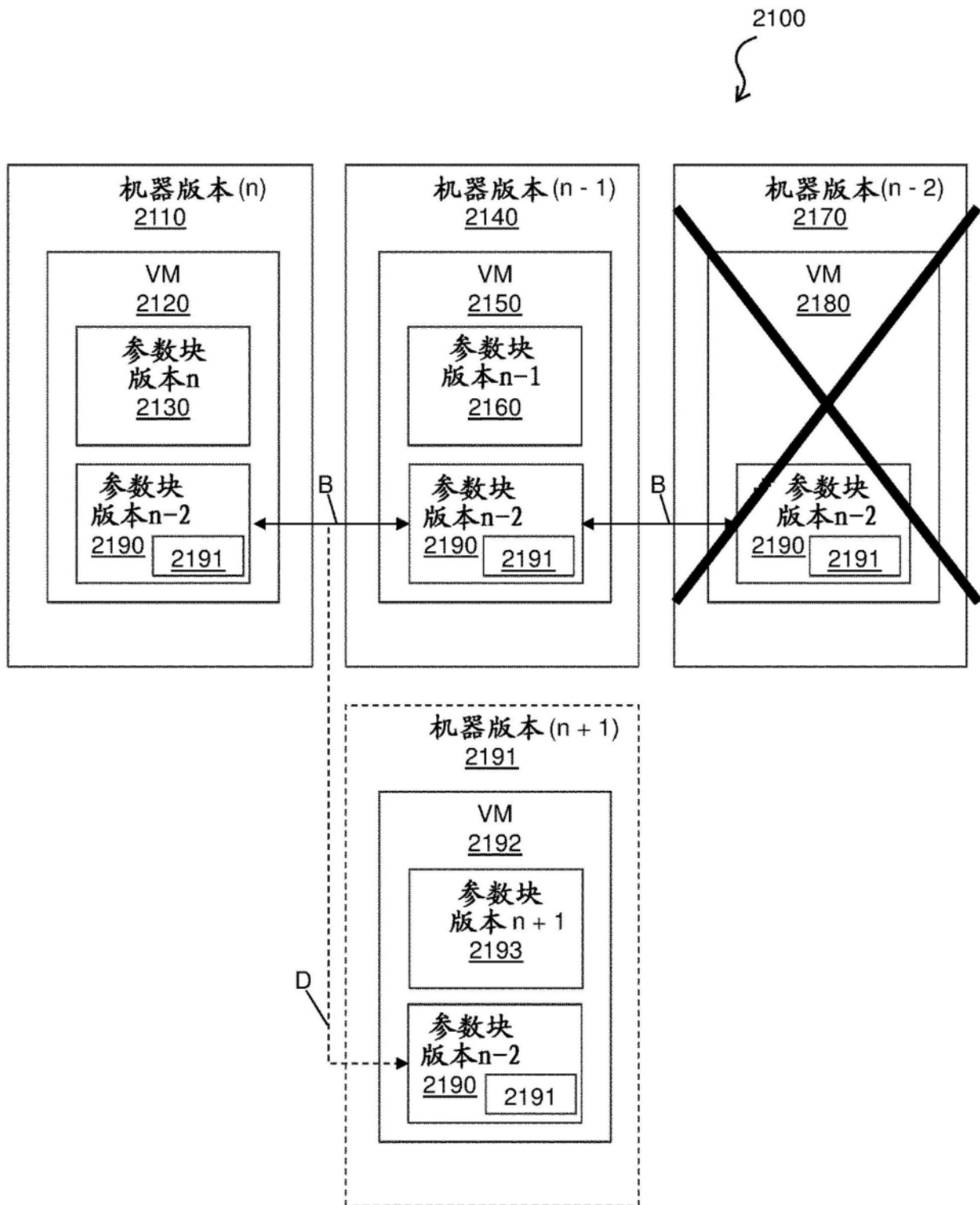


图21

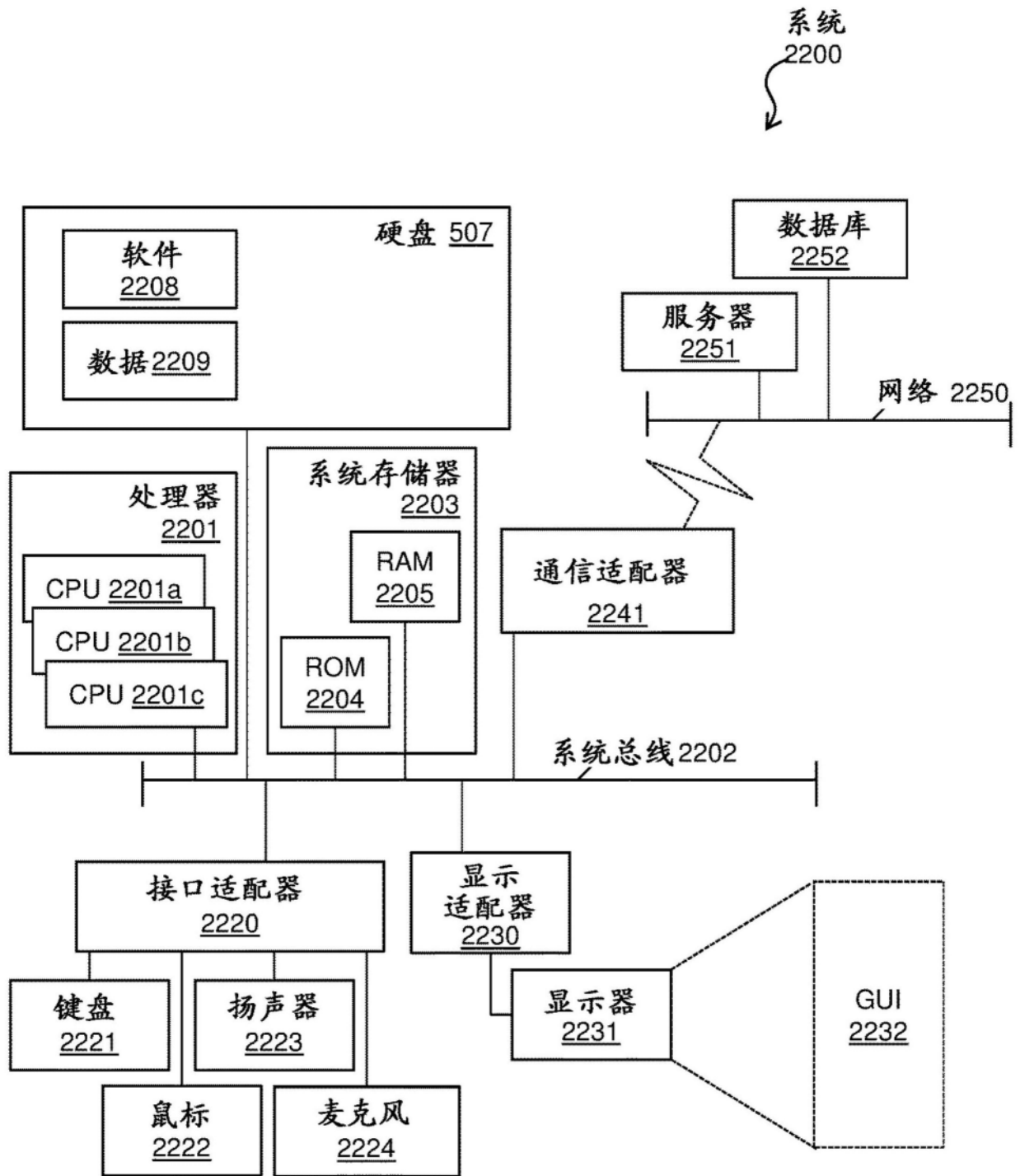


图22