[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 200310115738.4

[45] 授权公告日 2007 年 3 月 21 日

[11] 授权公告号 CN 1306399C

[22] 申请日 2003.11.28
[21] 申请号 200310115738.4
[30] 优先权
[32] 2002.12.10 [33] US [31] 10/316,606
[73] 专利权人 英特尔公司
地址 美国加利福尼亚州
[72] 发明人 艾伦·E·斯通
[56] 参考文献
CN1266513A 2000.9.13
JP20001067234A 2001.3.16
EP0926594A1 1999.6.30
审查员 王 丹

[74] 专利代理机构 北京东方亿思知识产权代理有限责任公司
代理人 杜 娟

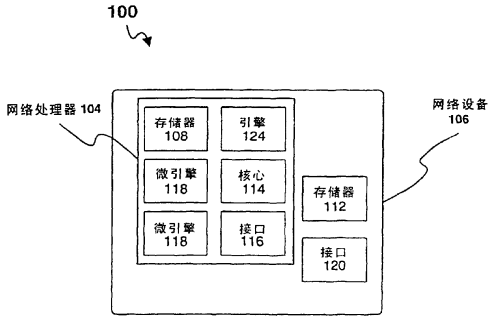
权利要求书 4 页 说明书 11 页 附图 3 页

[54] 发明名称

用于网络处理器的虚拟机

[57] 摘要

本发明公开了一种编译数据的方法、系统，以及存储有实现所述方法的指令的制品。 本发明公开了用于网络处理器的虚拟机的多个实施例。



1. 一种编译数据的方法，包括：

由虚拟机接收第一代码集；

将所述第一代码集的至少一部分提供给编译器；

将所述第一代码集的所述至少一部分划分为多个代码分组；

将所述代码分组的至少一个编译成多个第二代码集；以及

将至少两个所述第二代码集提供给不同的处理器资源。

2. 如权利要求 1 所述的方法，其中，所述接收包括：

被提供一个或多个字节码模块，其中所述接收由所述虚拟机的逻辑字节码接收器接口进行。

3. 如权利要求 1 所述的方法，其中，所述接收包括：

被提供软件程序，其中，所述软件程序被接收为一个或多个字节码模块。

4. 如权利要求 1 所述的方法，其中，所述划分基于执行所述编译器的设备可用的资源，以及所述第一代码集的至少一个功能。

5. 如权利要求 4 所述的方法，其中，所述资源包括实现在一个处理器设备中的一个或多个处理单元。

6. 如权利要求 1 所述的方法，还包括产生一个或多个代码集，以提供在不同处理单元上的所述两个或多个代码集之间的呼叫函数的逻辑链接。

7. 一种编译数据的方法，包括：

接收两个或多个字节码集；

将所述两个或多个字节码集分组为两个或多个任务，并定义所述两个或多个字节码集的至少一部分之间的关系结构以产生抽象语法图；

根据所述抽象语法图和执行所述两个或多个任务的资源需要将所述两个或多个任务划分为两个或多个子任务；以及

将所述两个或多个子任务的至少一部分提供给处理设备的不同资源。

8. 如权利要求 7 所述的方法，其中所述划分还包括为所述两个或多个子任务的至少一部分确定处理器映射，并且至少部分地基于所述处理器映

射提供逻辑分区。

9. 如权利要求 7 所述的方法，其中所述接收包括：

被提供一个或多个字节码模块，其中所述接收由所述虚拟机的字节码接口进行，以及将所述一个或多个字节码模块提供给编译器。

10. 如权利要求 7 所述的方法，其中所述划分基于执行至少一部分所述划分的资源可用的资源，以及两个或多个字节码集的至少一个功能。

11. 如权利要求 7 所述的方法，其中处理设备的所述一个或多个资源包括多核心处理器的一个或多个处理单元。

12. 如权利要求 7 所述的方法，其中所述划分至少部分地基于从前面的划分获得的反馈，其中，所述反馈从基于一个或多个度量标准来度量所述划分的适宜度分数获得。

13. 如权利要求 12 所述的方法，其中所述划分至少部分地基于所述反馈，在所述提供之前被执行一次或多次。

14. 一种编译数据的方法，包括：

由设备接收软件程序，其中所述程序包括用高级语言编写的多个指令；

在编译器中为所述多个指令的至少一部分确定关系结构；

确定设备的可用资源；

在编译器中将所述程序划分为多个任务，并将所述多个任务的至少一部分分别提供给设备的一个或多个资源，其中所述划分至少部分地基于所述关系结构和可用资源。

15. 如权利要求 14 所述的方法，其中，所述软件程序包括多个字节码。

16. 如权利要求 14 所述的方法，其中，所述关系结构包括抽象语法图，所述抽象语法图是一个或多个字节码的两个或多个功能之间的关系结构。

17. 如权利要求 14 所述的方法，其中，所述可用资源包括实现在一个处理设备中的一个或多个处理单元。

18. 如权利要求 14 所述的方法，其中，所述划分还包括将所述两个或

多个任务中的两个或多个提供给不同的可用资源。

19. 如权利要求 14 所述的方法，被提供软件程序，其中所述软件程序被作为一个或多个字节码模块接收。

20. 如权利要求 14 所述的方法，其中，所述划分基于对进行至少一部分所述划分的资源可用的资源，以及所述指令的至少一个功能。

21. 如权利要求 14 所述的方法，其中，处理设备的所述一个或多个资源包括多核心处理器的一个或多个处理单元。

22. 如权利要求 14 所述的方法，其中，所述划分至少部分地基于从前面的划分获得的反馈，其中所述反馈从基于一个或多个度量标准来度量所述划分的适宜度分数获得。

23. 一种编译数据的系统，包括：

网络设备；以及

编译器，其中所述编译器包括：字节码分析器，用于接收相关的字节码集；代码划分启发式引擎，用于将相关的字节码集划分为多个被划分的代码；以及至少一个合成器，用于将所述多个被划分的代码的至少一部分分别提供给多个处理单元，其中所述划分至少部分地基于网络设备的可用资源，以及实现在所述字节码集的至少一部分中的功能。

24. 如权利要求 23 所述的系统，其中所述接收包括：

被提供一个或多个字节码模块，其中所述接收被虚拟机的字节码接口执行。

25. 如权利要求 23 所述的系统，其中所述划分基于所述虚拟机可用的资源和所述字节码集的至少一个功能。

26. 如权利要求 23 所述的系统，其中所述资源包括与网络设备相关联的一个或多个硬件元件。

27. 如权利要求 23 所述的系统，其中所述资源可以包括一个或多个专用硬件加速器，其中所述编译器产生用于基于指令的处理资源的代码，以使用所述一个或多个专用硬件加速器来执行一个或多个字节码，其中，所述一个或多个字节码从一个或多个处理资源上被卸载。

28. 如权利要求 23 所述的系统，其中所述相关的字节码集包括机器语

言。

29. 如权利要求 23 所述的系统，其中网络设备的一个或多个资源包括多核心处理器的一个或多个异构处理单元。

30. 如权利要求 23 所述的系统，其中所述划分至少部分地基于从前面的划分获得的反馈，其中所述反馈从基于一个或多个度量标准来度量所述划分的适宜度分数获得。

31. 如权利要求 30 所述的系统，其中所述划分至少部分地基于所述反馈，在所述提供之前被进行一次或多次。

用于网络处理器的虚拟机

技术领域

本发明一般地涉及网络处理器，更具体地，本发明涉及用于网络处理器的虚拟机。

背景技术

软件一般用高级语言编写，然后被编译成机器可读的语言，或者本机指令，以被处理器执行。虚拟机可以帮助编译高级语言。作为编译过程的一部分，软件的各种功能可以被映射到处理器的特定模块来执行。这种处理器编程方法对于单个处理器或者包含对称多处理器平台的设备的编程可能是足够有效的，但是对于如网络处理器的具有多个异构处理单元的设备的编程，可能就不是特别有效的方法了。

对如网络处理器的多核心处理器编程可能需要作出诸如哪个处理器将执行特定的模块的判断，并且可能要求程序员作出关于功能划分的决定，以及怎样最好地利用每个处理核心。这可能导致对多核心处理器进行编程变得繁重和复杂，还使得对可用的处理资源的使用不那么有效。因此，需要改进的编写代码的方法，以及包含更有效地使用处理资源的能力的虚拟机。

发明内容

为了解决上述问题，本发明的一个方面提供了一种编译数据的方法，该方法包括：由虚拟机接收第一代码集；将第一代码集的至少一部分提供给编译器；将第一代码集的至少一部分划分为多个代码分组；将代码分组的至少一个编译成多个第二代码集；将至少两个第二代码集提供给不同的处理器资源。

本发明的另一个方面还提供了一种编译数据的方法，该方法包括：接

收两个或多个字节码集；将两个或多个字节码集分组为两个或多个任务，其中两个或多个任务的至少一个可以包括复合的任务；将两个或多个任务划分为两个或多个子任务；将两个或多个子任务的至少一部分提供给处理设备的不同资源。

本发明的另一个方面还提供了一种编译数据的方法，该方法包括：由设备接收软件程序，其中程序包括用高级语言编写的多个指令；为多个指令的至少一部分确定关系结构；确定设备的可用资源；将程序划分为多个任务，并将多个任务的至少一部分分别提供给设备的多个资源，其中划分至少部分地基于关系结构和可用资源。

本发明的另一个方面还提供了一种编译数据的系统，该系统包括网络设备和编译器，其中编译器在操作中被配置来接收相关的字节码集，将字节码集划分为多个被划分的代码，以及将多个被划分的代码的至少一部分分别提供给多个处理单元，其中划分至少部分地基于网络设备的可用资源，以及包含在字节码集的至少一部分中的功能。

本发明的另一个方面还提供了一种制品，该制品包括存储介质，存储介质包括所存储的指令，当这些指令被处理器执行时，导致接收两个或多个代码集，将两个或多个代码集分组为一个或多个任务，将一个或多个任务划分成两个或多个子任务，并将两个或多个子任务提供给一个或多个处理器。

根据本发明的、用于网络设备的虚拟机具有可以更有效地使用处理资源的能力。

附图说明

在说明书的结束部分具体指出并清楚地要求了被作为所要求保护主题的实施例的主题。但是，通过结合附图参照下面更详细的描述，可以更好地理解所要求保护主题的实施例，这关于组织和操作的方法，以及其目的、特征和优点，在附图中：

图 1 是适合于实施所要求保护主题的一个实施例的设备；

图 2 是根据所要求保护主题的一个实施例的虚拟机的框图；

图 3 是根据所要求保护主题的一个实施例的编译器的框图。

具体实施方式

所要求保护主题的实施例可以包括用于网络处理器的虚拟机。如网络设备的计算设备通常可以包含处理器，如网络处理器，并且可以由一个或多个处理单元组成。网络处理器可以执行被称为汇编语言的代码，这种语言是为特定类型的处理器特别编写的低级语言。用于一种类型或类别的处理器的汇编语言代码一般与另一类型的处理器是不兼容的。例如，特别为 Intel® Xscale RISC 处理器编写的低级汇编语言有可能不能在 Intel® Pentium™ 处理器上执行，因为这两个处理器使用不同类型的指令集。这可能导致需要根据执行软件的处理器类型来编写同一软件的几个版本以实现相同的功能。为了帮助解决这个问题，已经开发了也可以被称为通用编程语言的更高级的编程语言，这在很大程度上使得底层处理器变得不为人知。例如，C、C++、BASIC 以及 FORTRAN 语言都允许程序员编写非特定处理器专用的代码。用于特定处理器的编译器接收以这种语言所编写的源代码，并且可以产生用于特定目标处理器的指令。但是这些方法不包括以多个异构处理单元为目标的能力。

近来，这些所谓的通用编程语言可以为程序员提供这样的能力：只编写一次用于特定功能的代码，并通过抽象处理器指令集将代码用于几个不同的计算设备或处理器，使得在不同的平台间计算功能是相似的。这些非处理器专用的、并且可能依赖一个或多个虚拟机来抽象用于底层处理器的指令的编程语言的另外的例子，例如可以包括 Java 和 C#。这些类型的语言一般被编译成固定而标准化的虚拟机指令集合，该集合可以保持恒定而与将接收代码的目标处理器无关。这些虚拟指令被统称为字节码。字节码可以被分组成一个或多个字节码模块，并且一个或多个字节码模块可以构成字节码汇编。这些字节码指令最终被编译成特定设备原有的机器码的形式，其中，接收字节码的设备可以使用包含虚拟机的字节码编译器来进行编译。

网络处理器在编译代码时可以利用一种或多种类型的虚拟机。虚拟机

可以在高级指令集、编译器和作为结果产生的编译后的代码之间提供某种级别的功能性。例如，虚拟机可以确定运行时间联接（linkage），或者将字节码模块映射到处理器的各模块来执行。但是，如在后面详细示出的一样，为了编程的目的，虚拟机不会将一个或多个异构处理单元作为单个处理器。

此外，网络处理器，如基于 Intel® IXP 2800 的网络处理器，可以包含多个处理单元，这些处理单元例如可以包括处理器核心和多个微引擎。一般地，如字节码的代码是为单独的处理单元编写的，并且虚拟机没有将为一个特定处理单元指定的字节码模块映射到另一个处理单元来执行的能力，或者作出任何关于在一个或多个处理单元之间对字节码模块或字节码模块的部分进行划分的决定的能力。此外，字节码模块一般被看作为是关系清楚的，这意味着，例如，当一套模块被提交给虚拟机时，即使模块可能需要使用相同的处理单元或者执行补充或相关的功能，但当确定如运行时间联接的性质时，也不考虑所有模块的功能性。至少部分地由于这些限制，可能有必要为每个处理单元编写软件，并且在软件被提交给虚拟机时作出怎样对每个任务进行划分、以及哪个处理单元将执行每个任务的决定。这可能导致对网络处理器的处理能力的低效率的使用，并给如网络处理器的多核心处理器的编程增加了复杂性，就如在后面详细示出的一样。

所要求保护的主题的实施例可以包括虚拟机，该虚拟机包含有接收一个或多个字节码模块、并将一个或多个字节码模块映射到多核心处理器的一个或多个处理单元的能力，其中多核心处理器可以由多个异构处理器单元组成，例如由一个或多个微引擎和/或一个或多个处理器核心组成。所述的映射可以至少部分地基于处理器的可用资源、以及一个或多个字节码模块的一个或多个期望的功能来进行。

值得注意的是，说明书中引用的任何“一个实施例”或“实施例”是指结合实施例描述的具体的特征、结构或特点被包含在所要求保护主题的至少一个实施例中。在说明书的不同地方出现的短语“在一个实施例中”，不一定都指同一个实施例。

为了提供对所要求保护主题的实施例的完整的理解，这里可能阐述了

大量的特定细节。但是，对于本领域的普通技术人员，所要求保护主题的实施例可以在没有这些特定细节的情形被来实施。在其他情况下，为了不使所要求保护主题的实施例变得模糊，没有对公知的方法、过程、元件和电路进行描述。可以理解，这里公开的特定结构或功能细节可以是代表性的，而不一定限制所要求保护主题的范围。

现在具体地参照附图，在附图中，相似的部件全部用相似的标号指定，图 1 中图示了适合于实施所要求保护主题的一个实施例的设备。图 1 是设备 100 的框图，设备 100 可以包含如网络设备 106 的设备。网络设备 106 例如可以包含多业务交换器、宽带接入平台、网络交换机（Web switch）或者网络应用设备。在这个实施例中，网络设备 106 可以由如 Intel® IXP2800 网络处理器的网络处理器 104 组成。网络设备 106 还可以包含一种或多种类型的存储器 112，存储器可以包含例如同步动态随机存储器（SDRAM）和静态随机存储器（SRAM），以及一个或多个接口 120，接口具有与一个或多个其他设备通信的能力。网络设备 106 的一个或多个元件可以通过例如 PCI 总线的一个或多个总线（未示出）通信。网络处理器 104 可以由例如 SRAM 或 SDRAM 的一种或多种类型的内部存储器 108，以及可以包含例如总线接口的一种或多种类型的接口 116 组成。网络处理器 104 还可以包含如处理核心 114 的一个或多个处理单元，例如处理核心可以是 Intel® StrongARM® 核心（ARM 是英国 ARM 有限公司的商标）。处理器核心 114 还可以包含中央控制器（未示出），例如，其帮助为网络处理器的其他资源加载代码，并执行其他的通用目的计算类型功能，如处理协议、例外以及对包处理的额外支持。网络处理器 104 还可以包含引擎 124，引擎可以包含一个或多个硬件专用的加速功能，例如，例如加密引擎、CRC 校验器或乘法引擎。引擎 124 可以包括执行一个或多个专门功能的能力，这些功能例如可以从一个或多个处理单元卸载下来，但是不可以被归类为一个处理器单元。

网络处理器 104 可以包括一个或多个额外的如微引擎 118 的处理单元，在操作中，额外的处理单元可以被配置来与处理器核心 114 相互作用。一个或多个微引擎 118 可以包括存储器（未示出），例如存储器具有

存储指令的能力。在一个实施例中，有十六个微引擎，并且每个微引擎都包括处理有八个程序线程的能力。十六个微引擎可以用例如包括存储器系统 108、引擎 124 和总线接口 116 的共享资源来操作，但是存在另外的构造。所要求保护主题可以应用于任何具有多于一个处理器核心和/或多于一个微引擎的处理器，注意到这点很重要。

图 2 图示了系统 200，该系统并入了可以被实现为由处理器执行的软件的功能。执行软件的处理器可以是通用或者专用的处理器，如来自 Intel 公司制造的处理器家族的处理器，或者前面参照图 1 的网络处理器 104 描述的一个或多个处理器。软件可以包括编程逻辑、指令或数据以实现所要求保护主题的实施例的一定的功能。软件可以被存储在机器可访问介质或计算机可读的介质中，这些介质如只读存储器（ROM）、随机存取存储器（RAM）、磁盘（例如软盘和硬盘）、光盘（例如 CD-ROM）或者任何其他数据存储介质。在所要求保护主题的一个实施例中，介质可以存储压缩和/或加密格式的编程指令，以及是在被处理器执行前可能必须被编译的或被安装程序（installer）安装的指令。或者，所要求保护主题的实施例可以被实现为特定的硬件元件，元件包含用于执行所列举的功能的硬布线逻辑，或者通过程序通用目的计算元件和常规硬件元件的任何组合来实现。

在所要求保护主题的一个实施例中，系统 200 可以包含虚拟机。如前所述，虚拟机可以被配置来接收可以是非专用于特定处理单元的一个或多个字节码，解释一个或多个字节码，并将一个或多个字节码进行编译和/或映射以由一个或多个处理单元使用。在操作中，如所述的虚拟机可以为程序员提供正被编程的处理器的统一视图，并为程序员提供一个单片处理器的共用范式，而与组成正被编程的处理器的处理单元的数量无关。

在操作中，虚拟机 202 可以接收一个或多个字节码模块 204。字节码装入程序 206 可以接收一个或多个模块 204，但是也可以使用多种接收一个或多个字节码模块的方法。在这种环境下，接收可能包括下述功能，该功能至少部分地包括解释或读取一个或多个字节码模块，并且接收不限于对模块的物理接收。字节码装入程序 206 一般作为用于创建或汇编一个或多个字节码的程序和虚拟机 202 之间的逻辑字节码接收器接口。在接收一

个或多个字节码模块 204 之后，模块的至少一部分内容可以被提供到如即时（JIT）编译器 208 的编译器。在接收一个或多个字节码模块 204 之后，JIT 编译器 208 可以执行一个或多个编译任务，例如包括确定资源分配，如一个或多个字节代码的运行时间联接。在一个实施例中，运行时间联接可以是一种类型的资源分配，其中，模块 204 可以全部或部分地被分配给一个或多个处理单元来执行。

例如，装入程序 206 可以接收字节码模块。装入程序 206 可以将这个模块提供给 JIT 编译器 208。JIT 编译器 208 可以通过将模块的一部分编译成处理器 214 的本机指令，即这里示出的核心本机二进制码(core native binaries)210，来将模块的一部分功能映射到网络处理器的核心处理器 214。此外，JIT 编译器 208 可以通过将模块的另一部分编译成微引擎 218 的本机指令，即这里示出的、也被称为微块的微引擎本机二进制码 216，来将模块的另一个部分映射到如引擎 124 或微引擎 218 的一个或多个其他的资源上。接收一个或多个微块的核心处理器和一个或多个微引擎，然后可以执行一个或多个微块。JIT 编译器的特定功能可以参照图 3 被更好地理解。

图 3 是 JIT 编译器的框图，其可以与图 2 的 JIT 编译器 208 在功能上是相似的。图 3 中示出的是 JIT 编译器 300 的框图，说明了由根据所要求保护主题的一个实施例的虚拟机执行的编程逻辑。在这个实施例中，虚拟机可以指用来实现这里描述的一个或多个块的功能的软件和/或硬件。在所要求保护主题的这个实施例中，管理系统可以被实现为网络设备 106 的一部分。但是，可以理解，这个功能可以被位于通信网络中任何地方的任何设备或设备的组合实现，这仍旧落在所要求保护主题的范围内。

在操作中，图 3 中图示的编程逻辑的实现可以执行下面的功能，但是所给出的顺序并不意味着在实现时进行下面的功能的特定顺序。字节码分析器 302 可以被提供一个或多个字节码模块或者一个或多个字节码汇编，并且一个或多个模块或汇编可以从如装入程序 206 的字节码装入程序被接收。分析器 302 可以解释一个或多个字节码，这种解释包括读一个或多个字节码的至少一部分内容，以至少部分地确定模块中表示的一个或多个功

能。分析器 302 可以包括用来确定一个或多个字节码的一个或多个功能之间的关系结构的功能，这将在后面更详细地解释，这种关系结构也可以称为抽象语法图。在一个实施例中，分析器 302 可以将一个或多个字节码汇编成一个或多个任务，并且在这种情况下，例如，一个任务可以包括一个或多个字节码的一部分。

代码划分启发式引擎(Code Partitioning Heuristics Engine)304 可以接收至少一部分语法图，语法图可以由一个或多个任务组成。至少部分地基于抽象语法图、一个或多个任务和/或一个或多个与引擎相关的启发，引擎 304 可以确定例如执行一个或多个任务的资源需要、一个或多个处理器顺序或并行执行任务的能力以及对任务进行划分的能力的这些因素。基于上面提到的因素，引擎 304 可以至少部分地将一个或多个任务划分为一个或多个子任务。例如，一个或多个任务或子任务可以包括一个或多个字节码。当进行划分时，引擎 304 可以考虑一个或多个字节码中所包含的所有任务，以及例如可以用来处理任务的资源。引擎 304 还可以包括产生一个或多个在被划分的任务之间的一个或多个逻辑链接的能力，所述链接例如可以是一个或多个代码集，这些代码集提供在一个或多个任务之间的通信，这些任务可以互相依赖并且可以在不同的处理单元上被处理。例如，呼叫函数可以从一个处理单元被提供给另一个处理单元，并且用于提供呼叫函数的代码例如可以由引擎 304 产生。

一旦一个或多个任务被划分以被一个或多个处理单元处理，被划分的任务就被发送到合成器 310，其中，合成器可以是专用于一个处理单元或其他资源的，不过重要的是注意所要求保护的主题不限于此。例如，合成器可以进行将非专用的字节码译为与其相关的处理单元或资源专用的本机指令。被划分的任务被发送给与被指定来处理一个或多个任务的处理单元或资源相关的合成器 310。例如，除将被划分的任务提供给处理单元之外，合成器 310 可能能够提供一个或多个任务或子任务到与其相关的处理单元的额外的映射。例如，合成器 310 可以改变由引擎 304 提出的映射方案，并提供额外的优化，或者可以利用与微引擎相关的如硬件加速的特定硬件特征，例如图 1 的引擎 124。

所提出的映射信息然后可以被提供给评分模块 312，该模块可以包括确定关于由一个或多个合成器 310 或代码划分引擎 304 确定的映射的效率的一个或多个量度(measure)或度量标准(metrics)的能力。评分模块 312 可以接收一个或多个目标（goal）314，目标被用来确定一个或多个度量标准。目标可以以多种方式被引入，包括但不限于源代码实现（instrumentation）或注释、编辑元数据、配置文件或流、或者外部管理设备。评分模块 312 可以将适宜度分数作为适应反馈 316 提供给代码划分引擎 304。虚拟机的至少一部分功能可以至少部分地基于适宜度分数被重复。例如，在一个实施例中，处理可以一直被重复直到达到阈值容许分数，或者可以被重复特定的次数。在本上下文中，分数可以是任何可以被用来确定具体映射方案的期望度的量度或度量标准。度量标准的例子可以包括例如设备利用率、执行路径等待时间或者存储器的利用。一旦适宜度分数达到了使得对虚拟机的至少一部分功能的重复不再是所期望的，迭代就可以停止，并且可以使用接收最期望的结果如用于划分的最高分数的分区。或者，如果一个或多个目标没有被满足，则映射方案可以停止，并且这些结果例如可以通过虚拟机接口被作为通知提供。参照编译器 308 描述的功能块，在下面还可以进行更详细的描述。

在一个实施例中，字节代码分析器 302 可以接收或读取一个或多个字节码，并将它们汇编成抽象语法图。例如抽象语法图可以包括字节码如程序的关系结构。在这种情况下，抽象语法图可以包括一个或多个字节码的两个或多个功能之间的任何关系结构。例如，抽象语法图可以包括边和顶点的集合，其中，顶点表示功能，边代表功能的执行路经。当然，应该理解，这只是一个实施例，对多个任务的任何抽象都可以被用在所要求保护主题的至少一个实施例中，所述抽象提供将任务作为组的理解。在这个实施例中，字节码被汇编成程序，这可以提供一个或多个其他模块来确定一个或多个字节码的一个或多个目的的能力，这可以作为等同的实施例。

在一个实施例中，代码划分启发式引擎 304 可以从分析器 302 接收抽象语法图，并且，可以作为该语法图的一部分接收一个或多个任务。在这个实施例中，引擎 304 可以具有确定设备可用资源的能力，例如核心处理

器和微引擎的数量和能力。此外，引擎 304 可能能够确定设备的其他可以资源，例如加密引擎，其可能和处理单元不相关，但是可能能够对在一个或多个任务中所公开的相同功能进行仿真。在这个可选的实施例中，语法图公开的一个或多个功能可以被映射到除一个或多个处理单元之外的资源，例如，网络设备的一个或多个硬件元件，如同图 1 的引擎 124。

设备的可用资源可以至少部分地从资源描述符 306 提供，其例如可以包括关于例如网络设备的处理单元的一个或多个资源，或者如硬件或软件的其他可用资源的信息。资源描述符 306 还可以提供关于一个或多个资源的信息，如处理能力、限制或可用性。此外，引擎 304 可以确定任务的各种目的，并且可以确定如任务所期望的定时和优先权的因素。引擎 304 可以确定一个或多个任务是否可以被划分为一个或多个子任务，或者例如如果一些任务是被串起来的或彼此依赖，它们是否可以与其他任务并行处理或它们是否必须被顺序处理。引擎 304 可以基于一个或多个这些因素，来将字节码或任务映射到设备的特定处理器。这个映射信息可以被提供给一个或多个合成器 310。

在一个实施例中，一个或多个逻辑合成器 310 可以从代码划分启发式引擎 304 接收一个或多个任务。合成器可以与运行虚拟机的网络设备的单独的处理单元相关联。例如，对于在例如具有十七个处理单元的 Intel® IXP 1200 网络处理器上执行的虚拟机，JIT 编译器 308 可以由十七个逻辑合成器组成。或者，单个合成器可以与一类处理单元相关联，而不是与单个处理单元相关联。一个或多个合成器 310 可以评估由引擎 304 提供的任务的映射结构，并且可以改进或改变映射结构。这种改进或改变可以是因为任何数量的原因，例如，如效率、资源的可用性、处理器的优先权或者限制。在其他实施例中，一个或多个合成器 310 可以不对由代码划分引擎 304 执行的映射进行任何改变。但是，如果一个或多个合成器 310 不改变映射，那么一个或多个合成器 310 可以将一个或多个任务的映射度量标准提供给评分模块 312。这些映射度量标准可以被用来确定适宜度分数，该适宜度分数可以包括对引擎 304 进行划分以及将一个或多个字节码翻译成本地二进制码的效率如何的度量。

在一个实施例中，适宜度分数包括确定一些因素，如由引擎 304 执行的映射的效率，基于映射成功处理任务的能力，或者与映射或任务分配相比，处理单元或其他资源的可用容量。评分可以是基于一个或多个目标 314，目标可以包括这些因素，例如，如优先权、所期望的功能或定时，并且，在一个实施例中，可以由用户提供。或者，例如，目标 314 可以被自动产生，并且可以包括程序专用属性。此外，一个或多个度量标准可以从一个或多个合成器 310 被提供给评分模块 312，并且评分模块 312 可以至少部分地基于从一个或多个合成器 310 提供的一个或多个度量标准来确定分数。适宜度分数可以将至少一部分数据提供给适应反馈模块 316。

在一个实施例中，适应反馈模块 316 可以从评分模块 312 接收一个或多个值，如适宜度分数，并且可以具有对由引擎 304 产生的映射进行改变的能力。例如至少部分地基于一个或多个值，映射可以被改变。例如，模块 316 可以确定分数，或者块流程图 300 的一个或多个映射迭代上的分数的改变，并且可以基于例如在一个或多个迭代上的分数改变的趋势来对代码划分引擎 304 提出一个或多个改变。在这个实施例中，阈值可以被用来确定对映射或划分的改变是否是期望的。如果一个或多个改变是期望的，那么该信息被提供给引擎 304，在那里一个或多个改变可以被实现。信息例如可以通过逻辑接口（未示出）被提供，该接口允许从适应反馈模块 316 的输入。

可以理解，实施例可以被应用于包含至少一个处理单元的任何设备。此外，这里已经描述了所要求保护主题的实施例的特定特征，但是，对于本领域的技术人员，将可以想到很多修改、替换、改变和等效变化。此外，虽然这里已经详细描述了一个功能块以及它们之间的关系，但是于本领域的技术人员可以考虑到，几个操作可以不用使用其他的操作而被执行，或者可以建立额外的功能或者功能之间的关系，而仍旧符合所要求保护的主题。因此，应该理解，所附的权利要求意于覆盖所有这种落在所要求保护主题的实施例的真正精神之内的修改和变换。

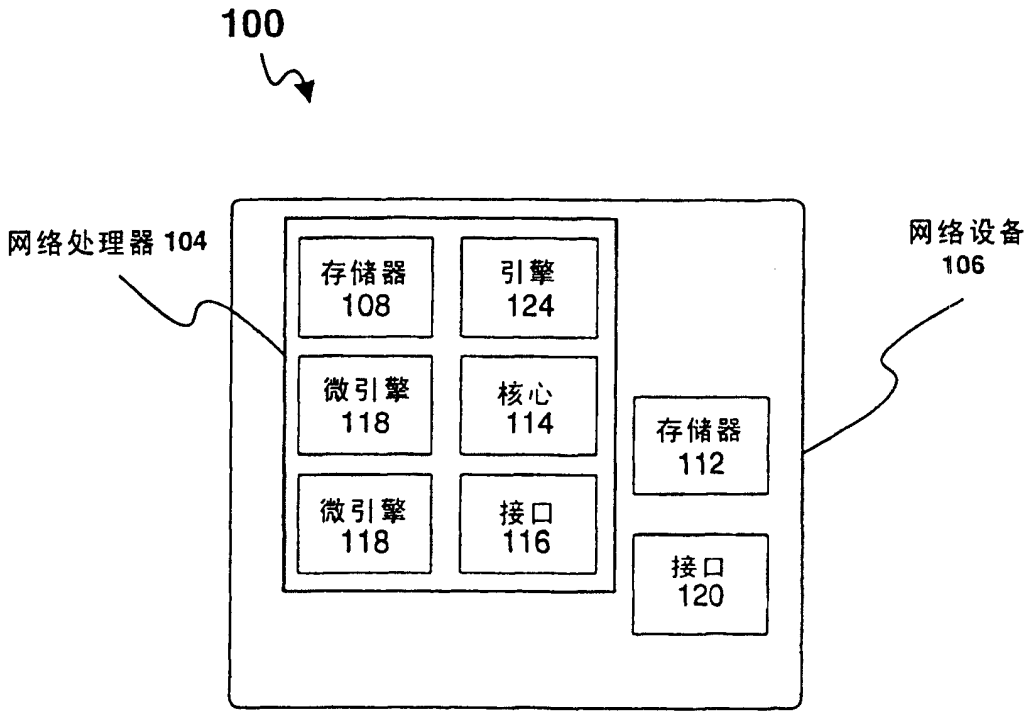

100
网络处理器 104
存储器 108
引擎 124
微引擎 118
核心 114
微引擎 118
接口 116
存储器 112
接口 120
网络设备 106

图1

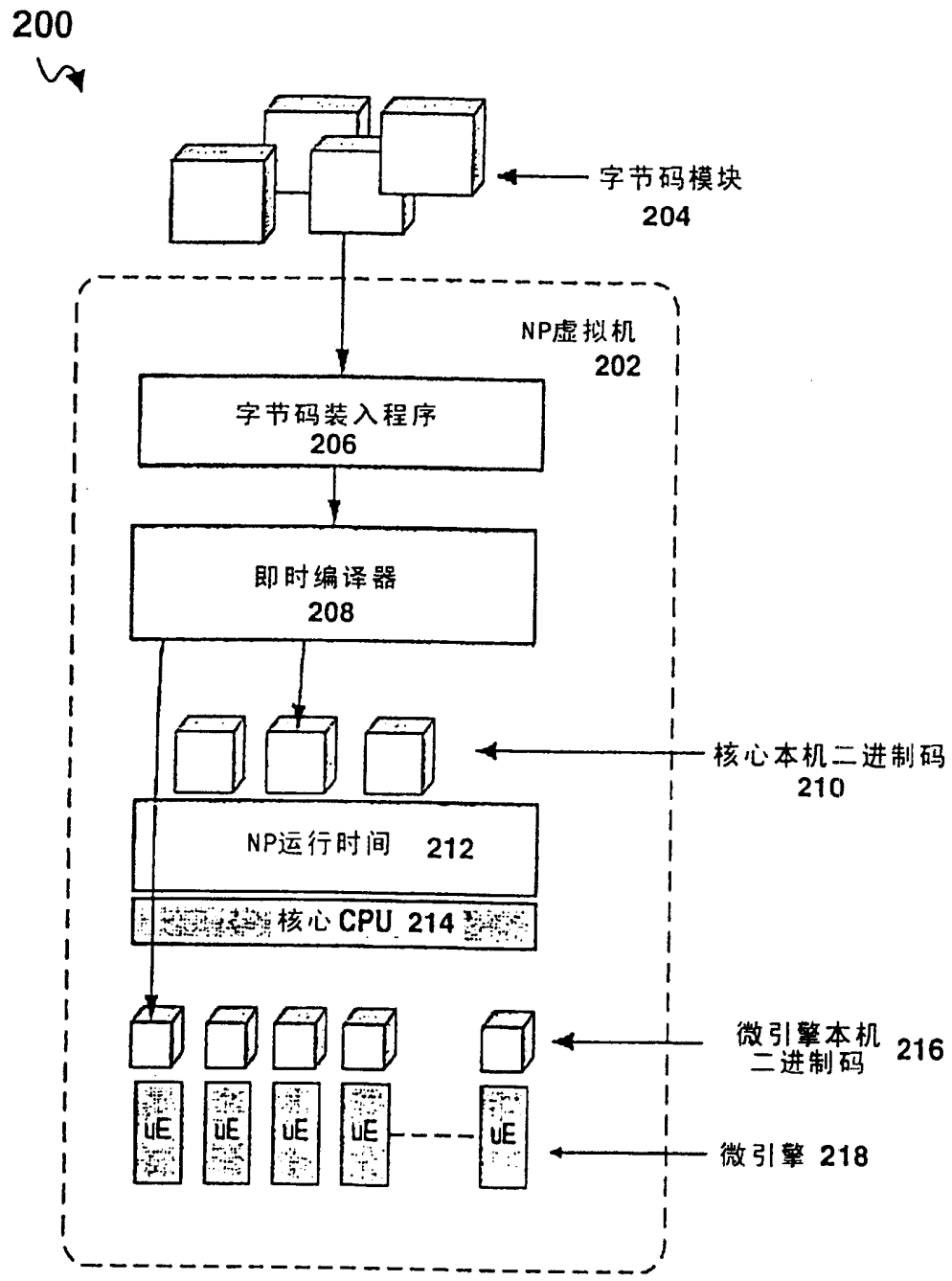

200
字节码模块
204
NP虚拟机
202
字节码装入程序
206
即时编译器
208
核心本机二进制码
210
NP运行时间 212
核心 CPU 214
微引擎本机
二进制码 216
uE
uE
uE
uE
uE
微引擎 218

图2

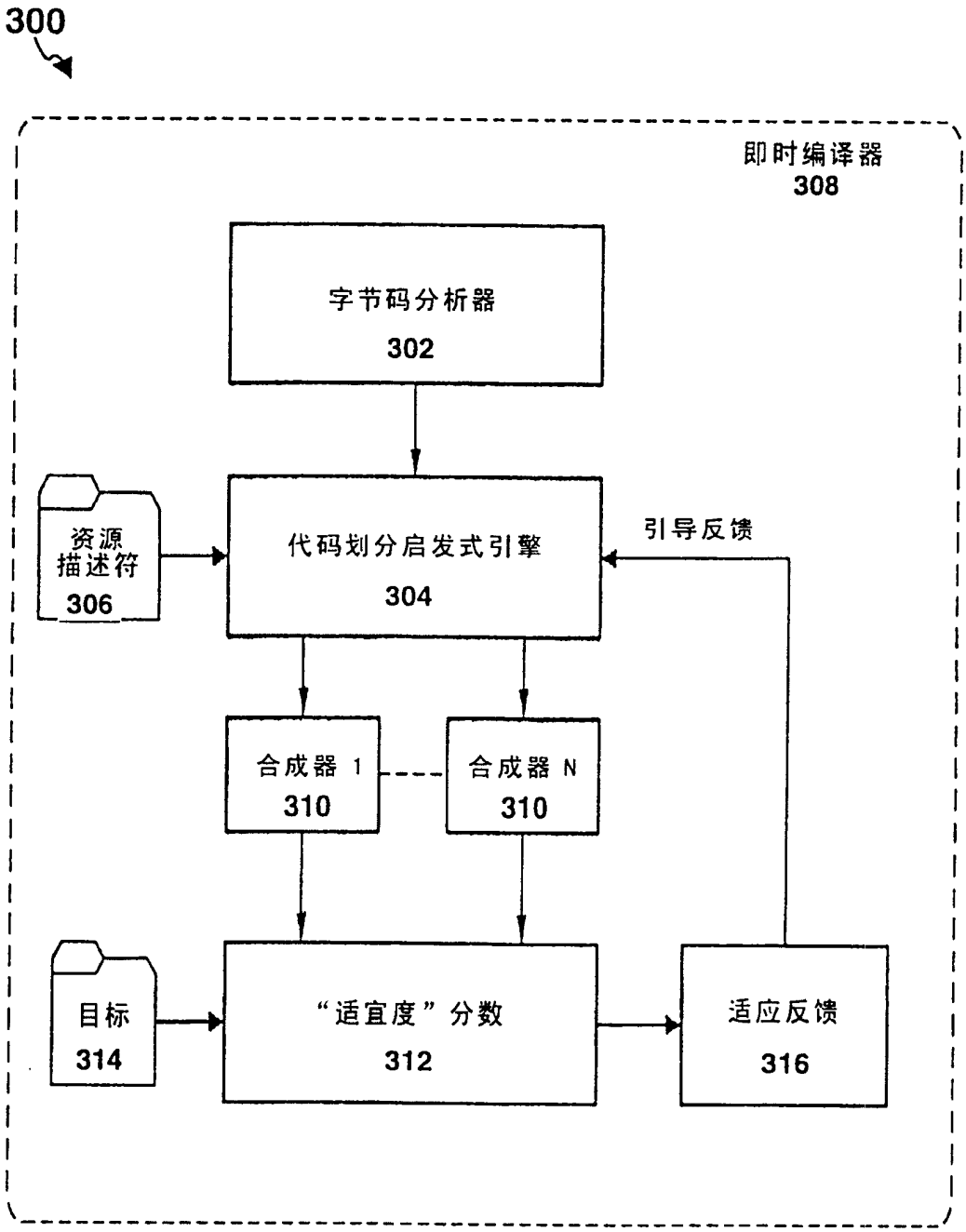

300
即时编译器
308
字节码分析器
302
资源
描述符
306
代码划分启发式引擎
304
引导反馈
合成器 1
310
合成器 N
310
目标
314
“适宜度”分数
312
适应反馈
316

图3