[19] 中华人民共和国国家知识产权局

[51] Int. Cl[7]
H04N 5/00
H04N 7/24
G06F 9/455
G06F 9/45

[12] 发明专利申请公开说明书

[21] 申请号 03814907.9

[43] 公开日 2005 年 8 月 31 日

[11] 公开号 CN 1663237A

[22] 申请日 2003.6.25 [21] 申请号 03814907.9

[30] 优先权

[32] 2002. 6.27 [33] EP [31] 02014326.9

[86] 国际申请 PCT/EP2003/050265 2003.6.25

[87] 国际公布 WO2004/003740 英 2004.1.8

[85] 进入国家阶段日期 2004.12.24

[71] 申请人 汤姆森许可贸易公司

地址 法国布洛里

[72] 发明人 玛丽 - 卢克·尚佩尔 劳伦·科涅

威廉·卢贝斯

[74] 专利代理机构 中科专利商标代理有限责任公司

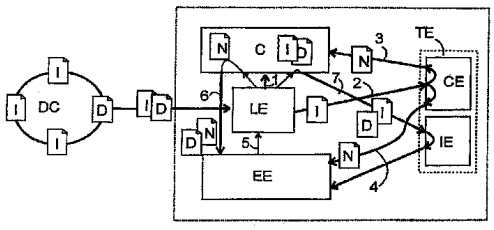
代理人 罗松梅

权利要求书 4 页 说明书 13 页 附图 3 页

[54] 发明名称 用于交互式电视的数据处理设备和方法

[57] 摘要

一种用于数字 TV 机顶盒的可编程数据处理设备，包括：加载引擎(LE)，用于接收来自 DSM - CC 转盘(DC)的第一类代码的部分和/或数据，存储装置(C)，用于存储由所述加载引擎接收到的所述部分，执行引擎(EE)，用于执行由所接收的部分具体化的应用程序；以及转换引擎(TE)，用于将所述第一类代码转换为所述执行引擎(EE)的本机代码所述转换引擎(TE)适合于将所述接收到的部分的至少特定的一个部分编译为本机代码，并且将这样编译的部分存储在所述存储装置(C)中，并解译代码的其它部分，所述执行引擎(EE)适合于处理相同应用程序中的已编译代码和已解译代码。



ISSN1008-4274

知识产权出版社出版

1.一种可编程的数据处理设备，包括：

加载引擎（LE），用于接收来自广播网络的流（DC）的第一类代码和/或数据的部分，所述部分在广播网络中被重复传送，

存储装置（C），用于存储由所述加载引擎接收到的所述部分，

执行引擎（EE），用于执行由所接收到的部分具体实现的应用程序，

转换引擎（TE），用于将所述第一类代码转换为所述执行引擎（EE）的本机代码，

其特征在于：所述转换引擎（TE）用于将这样编译的部分存储在所述存储装置（C）中，以便当从流中接收到预定信号通知信息时，将所接收部分中的至少特定的一部分编译为本机代码，并解译代码的其它部分，以及所述执行引擎（EE）用于处理相同应用程序中的已编译代码和已解译代码。

2. 根据权利要求1所述的数据处理设备，其特征在于：所述转换引擎（TE）用于根据由其接收的控制信息，选择所述特定部分。

3. 根据权利要求1或2所述的数据处理设备，其特征在于：所述流（DC）是DSM-CC转盘。

4. 根据权利要求3所述的数据处理设备，其特征在于：所述部分是DSM-CC模块（I）。

5. 根据权利要求3所述的数据处理设备，其特征在于：所述部分是DSM-CC模块的片断。

6. 根据权利要求2到5之一所述的数据处理设备，其特征在于：所述转换引擎（TE）用于接收来自所述流（DC）的所述控制信息。

7. 根据权利要求3、4和6所述的数据处理设备，其特征在于：所述控制信息是DSM-CC预取信号通知。

8. 根据权利要求3和6以及权利要求4和5之一所述的数据处理设备，其特征在于：所述转换引擎（TE）用于从所述DSM-CC转盘（DC）

的净荷模块（T）中提取所述控制信息。

9. 根据权利要求8所述的数据处理设备，其特征在于：所述转换引擎（TE）用于从所述净荷模块（T）中提取与要编译的代码部分相关的编译优化信息，并在编译所述代码的部分的处理中，注意编译优化。

10. 根据权利要求2到9之一所述的数据处理设备，其特征在于：所述转换引擎（TE）用于接收来自执行引擎（EE）的控制信息。

11. 根据权利要求1到10之一所述的数据处理设备，其特征在于：所述转换引擎（TE）用于根据从执行引擎（EE）所接收的控制信息，确定是对第一类代码的指定部分进行编译还是进行解译。

12. 根据权利要求11所述的数据处理设备，其特征在于：所述转换引擎（TE）用于在编译给定第一类代码部分期间，忽略需要对所述部分进行解译的控制信息，并完成对所述部分的编译。

13. 根据权利要求11所述的数据处理设备，其特征在于：所述转换引擎（TE）用于当在编译所述部分期间，接收到需要对给定第一类代码部分进行解译的控制信息时，放弃编译，并开始解译所述部分。

14. 一种数据处理方法，包括以下步骤：

a）接收来自广播网络的流（DC）的第一类代码（I）和/或数据（D）的部分，所述部分（I、D）在广播网络中重复传送，其中，在所述流（DC）中传送的部分的集合具体实现了一个或多个数据处理应用程序（a1、a1'）；

b）将所述部分中的预定部分存储在存储装置（C）中（a6、a9'、b3）；

c）当从流中接收到预定信号通知信息时，在转换引擎中将包括第一类代码的至少一个所述部分编译成执行引擎的本机代码；

d）在执行引擎（EE）中，通过执行属于所述一个应用程序的选择部分的已编译的本机代码（N）（c6），以及通过解译此应用程序的未选择部分（c7），来执行所述数据处理应用程序之一。

15. 根据权利要求14所述的数据处理方法，其特征在于：在步骤c和d之间包括以下步骤：从用户接收用于指定要在步骤d中执行的应用程序的指令。

16. 根据权利要求14或15所述的数据处理方法，其特征在于：在所述步骤c中，根据提供给转换引擎（TE）的控制信息，选择所述至少一个部分（a2、a3'、a4'、a5'）。

17. 根据权利要求14到16之一所述的数据处理方法，其特征在于：所述流（DC）是DSM-CC转盘。

18. 根据权利要求17所述的数据处理方法，其特征在于：所述部分是DSM-CC模块。

19. 根据权利要求17所述的数据处理方法，其特征在于：所述部分是DSM-CC模块（I）的片断。

20. 根据权利要求16到19之一所述的数据处理方法，其特征在于：从所述流（DC）接收所述控制信息（a2、a5'）。

21. 根据权利要求18或20所述的数据处理方法，其特征在于：所述控制信息是DSM-CC预取信息。

22. 根据权利要求18到20之一所述的数据处理方法，其特征在于：所述控制信息是DSM-CC转盘（DC）的净荷模块（T）。

23. 根据权利要求22所述的数据处理方法，其特征在于：所述控制信息还包括与要编译的代码部分相关的编译优化信息，并当编译所述代码部分时，所述转换引擎注意所述编译优化信息。

24. 根据权利要求14到19之一所述的数据处理方法，其特征在于：从所述执行引擎（EE）中接收所述控制信息（a4'）。

25. 根据权利要求24所述的数据处理方法，其特征在于：所述转换引擎（TE）根据来自执行引擎（EE）的所述控制信息，确定是对给定的第一类代码部分进行编译还是进行解译。

26. 根据权利要求25所述的数据处理方法，其特征在于：如果所述转换引擎（TE）在编译所述部分期间，接收到需要对给定部分进行解译的控制信息，则该转换引擎忽略所述控制信息（c11）并完成所述部分的编译。

27. 根据权利要求25所述的数据处理方法，其特征在于：所述转换引擎（TE）在编译所述部分期间，接收需要对给定部分进行解译的控制信息，所述转换引擎放弃编译（c11'）并开始解译所述部分。

28. 根据权利要求14到27之一所述的数据处理方法，其特征在于：在步骤c）之后，释放分配给已编译部分的第一类代码的存储空间，以便进行覆写。

用于交互式电视的数据处理设备和方法

技术领域

本发明涉及一种数据处理设备和方法，具体地，其适用于将应用程序从网络的中央局传送到多个终端站，并在终端站执行该应用程序，其中下行链路传输（中央局到终端站）的网络数据速率远高于上行链路（终端站到中央局）传输的网络数据速率。

背景技术

此类型的典型网络是交互式TV网络。在这样的网络中，终端站由装备有所谓的机顶盒的电视机构成，中央局是广播站。在下行链路中，以高速率将音频/视频数据（AV数据）和应用程序数据传输到各个TV电视机；上行链路数据速率远低得多，且针对不需要反馈到广播站的应用程序，可以为零。

尽管这样的网络的拓扑与具有连接到公用服务器的多个工作站的公用计算机网络类似，这些网络的性质引起许多新的问题。一个问题在于：在交互式电视网络中，终端站的数量非常大，可能是一百万个终端或更多。而在本地计算机网络中，每个终端可以在任何给定时间，免费请求将特定应用程序下载到其，在交互式网络中实现其将需要极高的数据传输容量。在交互式电视的技术中，利用所谓的DSM-CC（数字存储介质命令与控制）转盘（carousel）来解决这个问题。DSM-CC转盘是与其中传送模块序列的AV数据一起、由广播站传送的数据流，每个模块均包括：代码和/或可以由终端站执行的一个或多个应用程序的数据。无限重复该模块序列的传输。由于应用程序的数量是有限的，在一定时间间隔之后，重复每一个模块的传输。为了能够执行应用程序，终端站必须收集属于这个应用程序的所有模块。

如果仅在接收到来自用户的指令之后，终端站开始收集给定应用

程序程序的模块以启动该应用程序，则可能会导致指令和启动之间显著延迟，这对于用户而言是令人烦扰的。

为了解决这个问题，根据DVB（数字视频广播论坛，www.dvb.org）的MHP（多媒体家用平台）标准，广播站可以传送所谓的预取信号通知，用于针对在转盘上传送的每个模块，指示其是否值得将该模块存储在该终端站处，这是由于存在在不久的将来可能必须执行该模块的可能性。因此，终端站可以具有存储装置，用于在其中存储由预取信号通知所指定的模块，从而在必须执行这样的模块的情况下，可以立即开始执行，而不用必须等待该模块的重传。

如果在DSM-CC转盘上传播的代码是可以由终端站执行的本机代码，上述方案可能是令人满意的，而无需另外的初步处理。但是，由于使用本机代码将有以下缺点：需要终端站的极其高度的标准化，且由于存在许多不同的这样的终端站制造商，对此，这样的标准化难以施加于寻求给终端站提供Java虚拟机，并将应用程序作为Java中间代码从广播站传送到终端站的行业。这种解决方案确实具有以下优点：具有在无论什么特定的基础硬件的任意DVB-MHP平台上运行的DVB-MHP应用程序。

然而，这种方案的代价在于：需要强有力的Java虚拟机，意味着需要显著的处理器速度和大量的存储器。在市场上，这种相对昂贵的硬件平台的需要通常被视为对交互式数字电视的DVB-MHP解决方案的重大缺陷，并且可以还被视为严重限制了DVB-MHP解决方案的快速发展。

在传统的Java执行环境中，经常通过减少执行时间来增加应用程序处理速度。这通过将Java类编译为可以由Java虚拟机的处理器直接执行的本机代码来进行，而不将其编译为在执行其之前首先必须由机器解译的中间代码。

在上述类型的数字TV网络中，显而易见，仅在终端站处执行这样的编译。如果广播站已经传送了编译就绪的应用程序，则将失去互操作性。如果在终端站处进行该编译，则可以改进执行速度，但将增加加载等待时间，这是由于当用户给出指令来启动给定应用程序时，直

到已经从转盘中接收到此应用程序的所有模块为止，他不但必须等待，而且另外，需要等待对其进行编译所需的时间。

发明内容

因此，本发明的目的是提供一种可编程的数据处理设备和数据处理方法，用于其中通过公用流的部分传送应用程序的网络中，其中，可以实现在输入启动应用程序的决定和实际启动应用程序之间较小的延迟，而无需高处理能力。

通过一种可编程数据处理设备来实现该目的，所述可编程数据处理设备包括：

加载引擎，用于接收来自广播网络的流中的第一类代码和/或数据的部分，所述部分在广播网络中重复传送，

存储装置，用于存储由所述加载引擎接收到的所述部分，

执行引擎，用于执行由所接收的部分具体实现的应用程序，

转换引擎，用于将所述第一类代码转换为所述执行引擎（EE）的本机代码，

其特征在于：所述转换引擎（TE）用于将已编译的部分存储在所述存储装置（C）中，以便当从流中接收到预定信号通知信息时，将所接收的部分中的至少特定部分编译为本机代码，并解译其它代码部分，以及所述执行引擎（EE）用于处理相同应用程序中的已编译代码和已解译代码。

通过一种数据处理方法来实现该目的，所述数据处理方法包括以下步骤：

a）接收来自广播网络的流的第一类代码和/或数据的部分，所述部分在广播网络中重复传送，其中，在所述流中传送的部分的集合具体实现一个或多个数据处理应用程序，

b）将所述部分的预定部分存储在存储装置中，

c）当从流中接收到预定信号通知信息时，在转换引擎中将包括第一类代码的至少一个所述部分编译成执行引擎的本机代码，

d）在执行引擎中，通过执行属于所述一个应用程序的选择部分

的已编译的本机代码，以及通过解译此应用程序的未选择部分，来执行所述数据处理应用程序之一。

在本设备和方法中，可以通过使转换引擎对已经在数据处理设备处接收到的代码部分进行编译，来利用在将指定的应用程序的代码部分传送到数据处理设备时的延迟。一旦接收到应用程序的整个代码，则可以开始执行应用程序；可以通过解译来执行在此时仍未编译的代码部分。

优选地，仅当用户已经输入达到该效果的指令时，将开始执行应用程序。在此之前，设备应该最好已经收集和编译了此应用程序的代码部分。

另外，当执行启动时编译代码部分的事实相对于其中解译所有代码的传统设备，提高了执行速度。在其中执行引擎和转换引擎共享处理能力的数据处理设备中，速度提高的主要原因在于：当解译时，每次执行该部分时，处理能力花费在将中间代码部分转换为执行引擎的本机代码上；而当编译时，一次性转换所有代码，因此，解译总是占有了最好用于执行应用程序的处理能力。但是，即使在执行引擎和转换引擎不竞争处理能力的系统中，速度提高也是可能的，这是因为可以使编译代码比解译代码更有效。

如果该设备的执行引擎和转换引擎是物理上分离的单元，通过使解译单元在应用程序的执行时间处，编译所述代码的另外部分，可以实现进一步改善。这可能无需提供功能特别强大的转换引擎，因为当执行引擎执行已编译代码时，转换引擎具有多余的时间，可用于编译。按照该方式，在执行期间，编译代码与解译代码的比率将持续增加，最后，直到编译了应用程序的所有代码为止。

如果转换引擎和执行引擎是在公用处理器中以分时方式实现的虚拟设备，则如上所述，编译还可以在执行时继续，或者可选地，当正在执行已编译代码时，公用处理器的所有处理能力可以赋予该执行引擎。

更有利地，转换引擎用于根据其接收到的信号通知信息来选择由其进行编译的代码部分。

所述信号通知信息的一个来源可以是其中传送代码部分的流。如果所述流是DSM-CC转盘，在这样的转盘中通常传送的预取信号通知可以用作所述信号通知信息。即，与传统相反，DVB-MHP平台将仅存储按照传送其而在以后最终将对其进行解译的形式的预取信号通知指定的DSM-CC模块，本发明的数据处理设备可以另外编译这样的模块。

当然，要编译的部分也可能是DSM-CC模块的片断。在这种情况下，为了指定要编译的片断，将必需在转盘上传送除DSM-CC预取信号通知的其他类型的信号通知信息。可以由开发该应用程序的人根据诸如执行一个或另一代码部分的相对频率或已编译代码相对于已解译代码的效率的预计增加（对于代码的不同部分可能是不同的），来定义这样的信号通知信息。

可选地或附加地，在第一时间，预取信息针对在转盘上传送的每个模块，指示是否必须将其以第一类的形式存储在终端站处。然后，转换引擎从执行引擎中接收表示要编译的代码部分的预定信号通知信息。这在应用程序运行时尤其有用的。广播者可以在适宜的时间启动该转换，例如，正好在执行应用程序前。这样的应用程序可以由多个模块或部分组成，其中一些可能会执行几百次，而其它的则根本不执行。广播者知道哪些模块或部分将在很快被执行，并可以预测该执行。通常，当开发应用程序时，能够评估哪些代码部分会频繁执行，而哪些不会，或者哪些模块将很快被执行。可以在应用程序的其它代码部分中，例如在初始化部分中包含用于频繁执行的代码部分的编译指令，从而当执行初始化部分时，可以从执行引擎向转换引擎发送根据这些编译指令的信号通知信息。

转换引擎还应该适合于根据其从执行引擎中接收的信号通知信息，确定对第一类代码的给定部分是进行编译还是进行解译。即，当执行引擎必须执行对地址的跳跃或分支指令时，转换引擎还必须接受指定了代码中的特定地址的信号通知信息。如果在必须执行该指令时，相应代码部分尚未编译，则转换引擎将必须对在跳跃或分支指令中指定的代码进行解译，以便避免处理延迟。

如果在执行指令时，包括指令中跳跃或分支的目标地址的代码部

分处于待编译状态，发生的一个可能性在于：转换引擎忽略来自执行引擎的、需要解译的信号通知信息，并完成编译所述代码部分。在这种情况下，执行引擎必须等待一些时间，直到其可以执行跳跃或分支指令为止，但是，这样的延迟可以通过随后增加的处理能力来进行补偿。

一种可选的可能性是转换引擎放弃编译此代码部分并开始解译其的情况，以防止执行引擎等待。此外，还可以进行设置，以致于所述转换引擎根据完成所述部分的编译所需的处理实际的估计，在这两个选项之间进行选择。

附图说明

根据随后结合附图所给出的优选实施例的描述，本发明的其它特点和优点变得明显。

图1是根据本发明的可编程数据处理设备的示意性方框图；

图2是由图1所示的设备执行的处理的第一流程图；

图3是由图1所示的设备执行的可选择的处理的第二流程图；

图4A、4B示出了图3的细节的两个可能的实施例；以及

图5是本发明的设备的第二实施例的方框图

具体实施方式

由图1中的虚线框架界定的数据处理设备包括：加载引擎LE、由编译引擎CE和解译引擎IE构成的转换引擎TE、高速缓存器C以及执行引擎EE。该设备是用于数字电视的机顶盒的一部分。

加载引擎LE适合于接收来自DSM-CC转盘DC的代码和数据的模块。在该转盘中，循环转送净荷模块组。存在两种类型的净荷模块：数据模块，用带有字母D的纸张表示；以及代码模块，用具有字母I的纸张表示。因此，代码模块和数据模块还分别被称为I模块和D模块。在代码模块I中的代码可以属于能够由执行引擎EE执行的一个或多个应用程序。在代码模块I中包含的代码是中间代码，即，比作为最初编写应用程序的高级语言的代码更紧凑（compact）的代码，但该中间代码没

有在先解译，是不可由执行引擎执行的。优选地，从中获得中间代码的高级语言是Java编程语言。使用中间代码是必需的，以便确保在由各种不同制造商制造的各种机顶盒中的代码的可操作性。

在加载引擎LE和高速缓存器C之间存在连接1，通过其，加载引擎LE可以将从转盘DC接收到的I和D模块存储在高速缓存器C中。在加载引擎LE和编译引擎CE之间存在另一连接2，通过其，加载引擎LE可以将I模块转送到用于编译的编译引擎CE。编译引擎CE将其编译为本机代码模块，用带有字母N的纸张表示，如图1如示，通过连接3将其转送到高速缓存器C，或者还可以通过连接4将其直接转送到执行引擎EE，以便立即执行。因此，高速缓存器可以包含三种类型的模块，即I、D以及N模块。

当执行单元EE执行应用程序时，其经由连接5，将所需的代码模块通知给加载引擎LE。然后，如果模块是已编译的，加载引擎LE将从高速缓存器C中提取相应的模块，并经由连接6，将其转送到执行引擎EE，或者如果模块是中间代码的，则经由连接7将其转送到解译引擎IE。

明显地，该设备的各个引擎的每一个可以是与其他电路不同的电路，或者这些引擎的一个或多个可以由微处理器具体实现，所述微处理器在不同的时间处，执行如上所述的各个引擎的不同任务。

现在，将参考图2和3来详细描述由该设备的各个引擎执行的处理。

在本描述中，假设本发明的设备是用于数字电视的机顶盒，其可以根据操作员的选择，执行由DSM-CC转盘连续传输的各种应用程序。

图2示出了在操作员已经选择了要执行的应用程序前，在接通之后立即由机顶盒中的加载引擎LE和编译引擎CE执行的、优选为处理a和处理b的两个处理。同时执行处理a和b。在处理a的步骤a1中，加载引擎LE接收来自DSM-CC转盘DC的模块。模块可以是I模块或D模块。在步骤a2中，加载引擎LE判断该模块是否伴随有预取信号通知。原则上，该模块是否信号通知为要预取的问题基于转盘DC的操作员的判断。通常，如果模块包含需要开始执行在转盘DC上传送的任何应用程序的代码或数据，则模块将至少被信号通知为要预取，由于这将使得一旦操

作员选择了应用程序，机顶盒将无任何延迟地开始执行。

如果模块没有被信号通知为要预取，则加载引擎将忽略该模块，并将等待下一个模块到达。

如果模块被信号通知为要预取，则加载引擎LE将在步骤a3中检查其是否已经存在于高速缓存器C中。如果是，则处理可以返回到步骤a1以等待下一个模块。可选地，如果存在在转盘DC上的后续传输之间更新模块的可能性，则可以提供：由新版本覆写高速缓存器C中存在的模块的较早版本。

如果模块未存在于高速缓存器C中，则加载引擎LE在步骤a4中检查在高速缓存器C中是否存在存储该模块的可用空间。如果不存在可用空间，则加载引擎LE选择可以步骤a5中被覆写的、高速缓存器中所存储的模块。选择标准可以是模块已经被未使用地存储在高速缓存器中多长时间。可选地，预取信号通知（signalization）可以指定各种优先级，并且为覆写而选择的模块可能具有低于当前接收到的模块的预取优先级。如果没有模块具有低于当前接收到的模块的预取优先级，则将必须丢弃当前所接收到的模块。如果在高速缓存器中存在可用空间，或如果已经通过选择进行覆写的模块而获得了可用空间，则在步骤a5中，将当前接收到的模块存储在高速缓存器中，加载引擎LE准备就绪以接收来自转盘DC的另一模块。

同时，编译引擎CE循环重复以下步骤：选择未编译模块的步骤b1；对其进行编译的模块b2；将已编译模块存储在高速缓存器C中的步骤b3。一旦编译了该模块，则释放包含原始中间代码模块的高速缓存器C中的存储空间，从而可以在其中存储任何新接收到的模块。

处理a和b可以是完全不同步的，即，无论何时编译引擎CE针对一个模块完成了步骤b3时，其立即返回到步骤b1，并在高速缓存器C处搜索另一要编译的模块。当在高速缓存器中存在多于一个的未编译模块，并对这些模块定义了预取优先级时，编译引擎CE将总是选择具有最高优先级的模块。

当加载引擎LE和编译引擎CE在单个微处理器或其它合适的电子电路中具体实现，并共享其处理能力时，处理a和b都不必使微处理器

保持被完全占用。在这种情况下，优选地，无论何时当接收到模块并且其必须判定是否存储该模块时，使微处理器充当加载引擎LE，例如，由中断来触发；以及无论何时当加载引擎处于空闲时，使其充当编译引擎CE，即，作为前台任务或高优先级任务来执行处理a，并作为低优先级任务或后台任务来执行处理b，无论何时当需要时，将中断其以有利于处理a。

在可选实施例中，处理a和b也可以是同步的，即，当已经执行了步骤a6时，触发处理b，并选择在步骤a6中存储的模块。

如果处理a、b是同步的，根据另一选择方案，可以省去步骤a6，并作为将接收到的模块存储在高速缓存器C中的替代，将其直接从加载引擎LE提供给加载引擎CE，并仅在编译之后，将其写入高速缓存器C。

如果加载引擎LE和编译引擎CE共享公用处理电路的处理能力，并且当接收到伴随有预取信号通知的另一模块时，存在模块的编译可能未完成的危险，则未同步的处理a和b是更优选的。然而，这样的危险通过选择其中在转盘DC上传送模块的序列可以得到降低，从而使要预取的模块有规律地散布在模块之间，而无需预取信号通知。

通常，信号通知为要预取的、DSM-CC转盘DC中的模块是包含有启动各种应用程序的任一个所需的代码和数据的模块。因此，如果高速缓存器的容量足够大，则在转盘DC上传送的、针对所有应用程序的启动代码部分将以已编译的形式存在于高速缓存器中，从而当操作员在机顶盒处选择特定应用程序时，可以开始执行该应用程序，而没有延迟。如果高速缓存器的存储容量足够大，则也可以将各种应用程序的其它重要模块存储在其中，并最终对其进行编译。如果其具有经常被执行的高可能性，则在上下文中，代码部分被称为重要的，以及/或者随后被称为应用程序的启动部分。

由于广播站的DSM-CC转盘可以服务于具有不同高速缓冲存储容量的各种机顶盒，优选地，所述预取信号通知应该包括优先级，表示使加载引擎LE能够仅选择最重要的代码部分，以将其存储在高速缓存器中。

当操作员在机顶盒上选择了要执行的应用程序并输入该选择时，

存在以下可能性：应用程序所需要的所有代码已经存在于高速缓存器中，以已编译或未编译的形式。如果这样，则可以停止处理a，且执行引擎EE将开始运行图3所示的处理c。

然而，在多数情况下，并非所述应用程序的所有代码均将存在于高速缓存器中，从而用图3所示的修改处理a'来代替处理a，其目的在于：从转盘DC处收集丢弃的代码部分。

在处理a'中，在步骤a1'中，加载引擎LE从转盘DC处接收模块。如果在步骤a2'中，发现相同模块已经存在于高速缓存器中，以已编译或未编译的形式，则该处理返回到步骤a1'等待下一个模块。如果该模块不在高速缓存器中，则步骤a3'检查所接收的模块是否为选择来要执行的应用程序的一部分。如果是，则处理分支到步骤a6'，稍后描述。如果回答是“否”，则加载引擎LE在步骤a4'中检查其是否从执行引擎EE接收到指示该模块可能是该执行引擎所想要的任何控制信息。如果存在当前应用程序调用第二应用程序的可能性，则该执行引擎可以发送这样的控制信息，所述控制信息涉及到第二应用程序的特定模块当前未根据当前执行的应用程序中这样做的指令来执行。

如果在步骤a4'中的回答是“是”，则处理分支到步骤a6'，如果不是，则步骤a5'检查该模块是否是信号通知为要预取。如果是，则该处理还切换到a6'，如果不是，则其返回到a1'以等待另一模块。

在步骤a6'中，给模块赋予优先级。如果其是预取信号通知模块，且预取信号通知指定了优先级，则可以赋予该优先级。如果模块属于当前应用程序，或者如果其是执行引擎EE所想要的，则给其赋予高于任何预取信号通知优先级的优先级。

然后，步骤a7'检查是否存在可用的高速缓存器空间。如果没有，则在步骤a8'中，必须选择在高速缓存器C中的模块来进行覆写。此选择考虑到在步骤a6'中赋予当前模块的优先级、以及已经处于高速缓存器C中的模块的优先级。由于属于当前应用程序或由执行引擎想要的模块与可能属于任何其它应用程序的预取信号通知模块相比，具有更高的优先级，因此，这些预取信号通知模块总是被覆写，以便能够存储当前应用程序的模块或所需模块。如果当前模块的优先级非常低以致

于在高速缓存器中不存在可以被覆写的其它模块，则将丢弃该模块。如果在高速缓存器C中存在可用的空闲空间，或者如果在高速缓存器C中存在可以被覆写的模块，则在步骤a9'中，将当前模块存储在高速缓存器C中。

如果高速缓存器容量是足够的，在转盘DC的一个循环之后，应用程序的所有模块都将处于高速缓存器C中。

在加载引擎LE正在执行处理a'的同时，编译引擎CE可以继续图2所示的处理b，从而在一定时间之后，所选择的应用程序的整个代码将以已编译形式的可用。

在编译引擎CE和执行引擎EE共享公用处理电路的处理能力且处理能力不足的实施例中，当开始执行应用程序时，还可以停止处理b。

图3所示的处理涉及由执行单元EE来执行所选的应用程序。

在第一步骤c1中，执行单元EE将执行应用程序的启动模块。假设该启动模块以已编译的形式存在于高速缓存器C中。最后，其可能需要执行单元EE在步骤c2中，跳到应用程序代码的另一模块。为此，执行单元EE向识别所需模块的加载引擎LE发送控制信息。

在步骤c3，加载引擎LE确定该所需的模块是否存在于高速缓存器C中，如果不是，则需要等待，直到从转盘DC处接收到该模块为止（步骤c4）。如果该模块存在于高速缓存器中，则加载引擎LE进一步确定该模块是否已编译（步骤c5）。如果该模块是已编译的，则在步骤c6中，将已编译的代码直接从高速缓存器C转送到执行单元EE；否则，将其转送到用于解译的解译引擎IE，并且在步骤c7中，解译引擎将已解译的指令提供给执行引擎EE。继续执行已编译的代码或解译，直到任一处理回到步骤c2，以便完成到另一模块或应用程序的跳跃。

根据处理c的改进实施例，在处理c的流程图中所示的虚线框c'可以包括如图4A或4B所示的附加方法步骤。也就是，如果处理b与处理c同时地继续进行，则可能出现以下情形：在步骤c2中执行引擎EE试图跳到其上的模块正在由编译引擎CE进行编译（步骤c10）。在这种情况下，存在两种避免冲突的可能性。如图4A所示，可以使执行引擎EE不得不等待，直到编译引擎CE已经完成编译代码为止（步骤c11），然后，

转移到上述步骤c6。可选地，如图4B中所示，在步骤c10中识别冲突，在步骤c11'中可以中止编译，并在步骤c7中解译模块。

在到目前为止所描述的实施例中，已经假设：当加载引擎LE从DSM-CC转盘DC接收模块时，其将使用转盘的信号通知信息，例如预取信号通知，以确定编译所接收到的代码模块是否合适。当然，这是执行本发明的一种简单且方便的方法，具有的优点在于：其可以采用传统的DSM-CC转盘，而无需对由其传送的信号通知和净荷进行任何修改。

在本发明的改进实施例中，净荷模块用作控制信号，用于使本发明的设备能够确定是否应该编译接收到的代码模块。因此，提供了针对DSM-CC转盘的第三类的净荷模块。该模块可参考以下的代码分析表，并用图5所示的带有字母T的纸张表示。

代码分析表T可以包含：

a）控制信息，用于识别转盘的中间代码模块I或这样的模块中应该预编译的部分。对模块部分的编译可能是有利的，这是由于转盘的一个I模块可以包括属于各个应用程序的代码，而并不是所有代码都是对预编译有用的。此外，在与单个应用程序有关的代码中，可能存在很可能比其他部分更经常被执行的部分。因此，为了从开始就实现应用程序的最佳执行速度，仅对I模块中的重要的部分进行预编译是明智的，留下较不重要的部分来进行解译，从而所节省的编译引擎CE的处理时间可以用于编译另一I模块的重要部分。例如，这些部分可以中间模块的单独功能或方法。

该控制信息还可以表示对各个部分进行编译的优先级。这使编译引擎能够首先编译I模块中具有最大优先级的部分，然后，编译其它模块中具有相同优先级的部分，并且一旦编译时间可能是多余的，则开始编译模块具有较低优先级的部分。

b）编译优化信息。该信息可以给予编译引擎CE关于如何对给定的代码部分进行编译的提示，以便实现最优系统性能。在此信息中指定的优化可能涉及：诸如函数直接插入（inlining）、环路展开、寄存器使用、管道技术、开关重写等。此信息由给定应用程序的开发者准备。机顶盒不需要留意在T模块中给出的编译提示。如果其不这样做，

其将还能够正确地编译，但是如果这样做，其将能够产生高效的本机代码，尽管编译引擎的算法可能会非常简单。因此，根据其的先进程度，机顶盒能够不注意这些提示、或注意这些提示的部分或全部。

通过本发明，实现了以下主要优点。首先，由于对应用程序的至少部分代码进行编译，强有力地提高了应用程序的执行速度。代码的解译明显慢于已编译代码的执行，这主要由于以下事实：当进行解译时，每次在执行该部分时，处理能力花费在将中间代码的部分转换为执行引擎EE的本机代码上，而当进行编译时，仅一次就转换所有代码。

当应用程序的瓶颈部分得到编译时，本发明及其有效。事实上，当仅应用程序的某些部分引起执行引擎上的重负载时，则可能不需要对应用程序的所有部分进行编译。明显地，当存在多次重复的代码部分时，通过编译其所实现的效率的增加是相当可观的。另一方面，可能存在仅执行一次或具有较低执行可能性的代码部分。为此，编译将不会引起效率上的提高，从而可能不值得对其进行编译。第二个优点在于：通常，编译使代码更为紧凑。因此，如果编译代码，与未编译相比，将能够在指定的高速缓存器尺寸中存储更多的模块。

由于更有效地使用了存储器，能够使给定MHP同时容纳多得多的应用程序。从用户的观点来看，这使机顶盒更为强大。

最后但并非最不重要地，通过明智地准备该代码分析表，应用程序的开发者在如何将他编写的代码转换为执行引擎的本机指令方面具有增强的控制。按照该方式，利用相当简单的编译引擎和/或解译引擎，可以产生高效的本机代码。

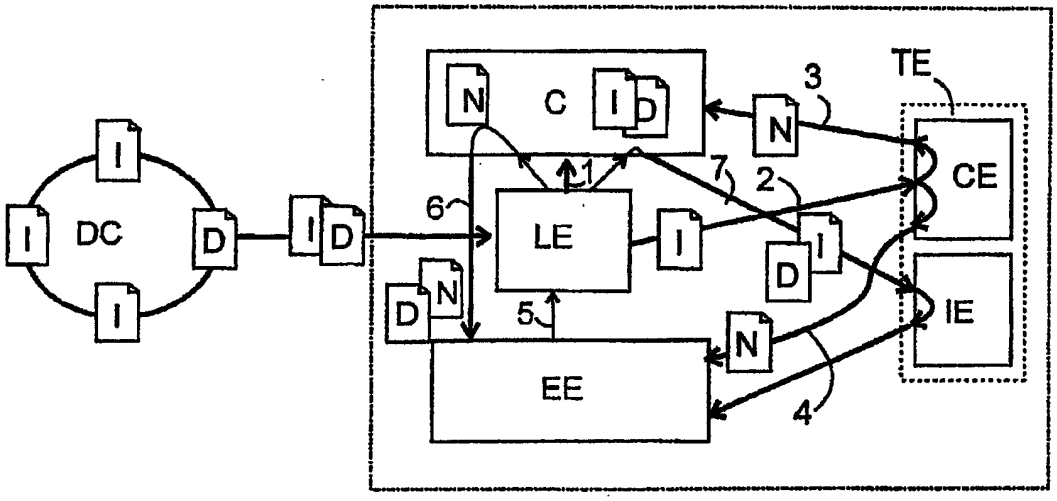

I
I
DC
D
I
I
I D
6
LE
N
C
I D
1
N
3
TE
7
2
CE
I
D
I
IE
D N
5
N
EE
4

图 1

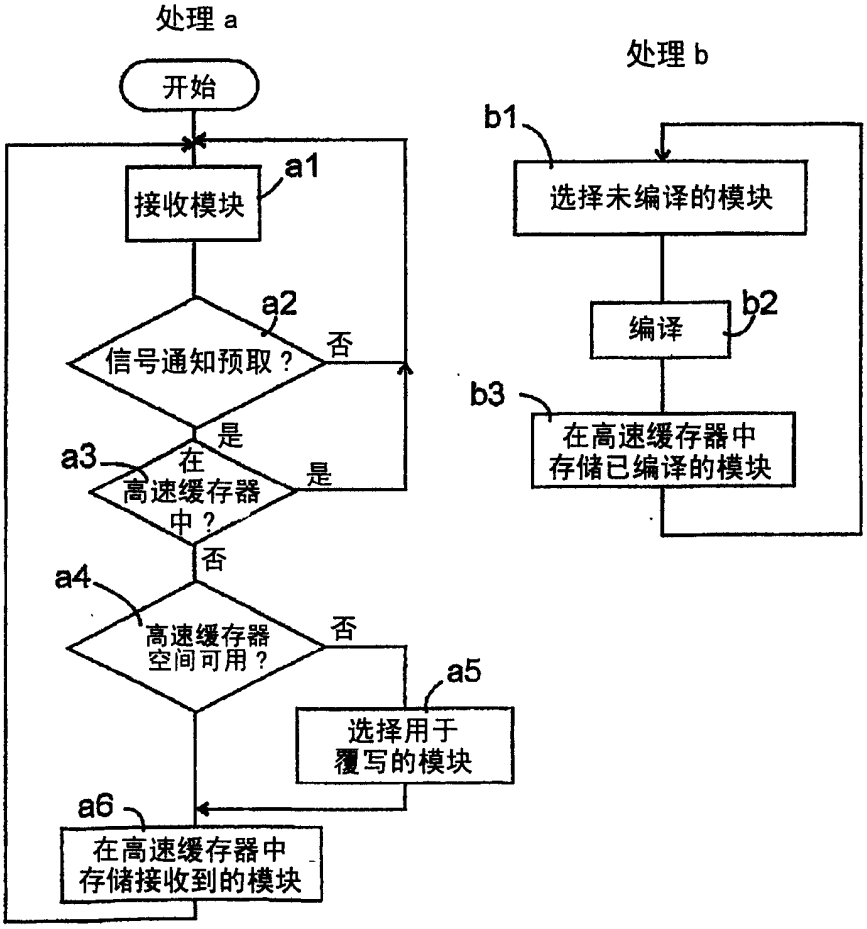

处理 a
开始
接收模块
a1
a2
信号通知预取？
否
是
a3
在
高速缓存器
中？
是
否
a4
高速缓存器
空间可用？
否
a5
选择用于
覆写的模块
a6
在高速缓存器中
存储接收到的模块
处理 b
b1
选择未编译的模块
编译
b2
b3
在高速缓存器中
存储已编译的模块

图 2

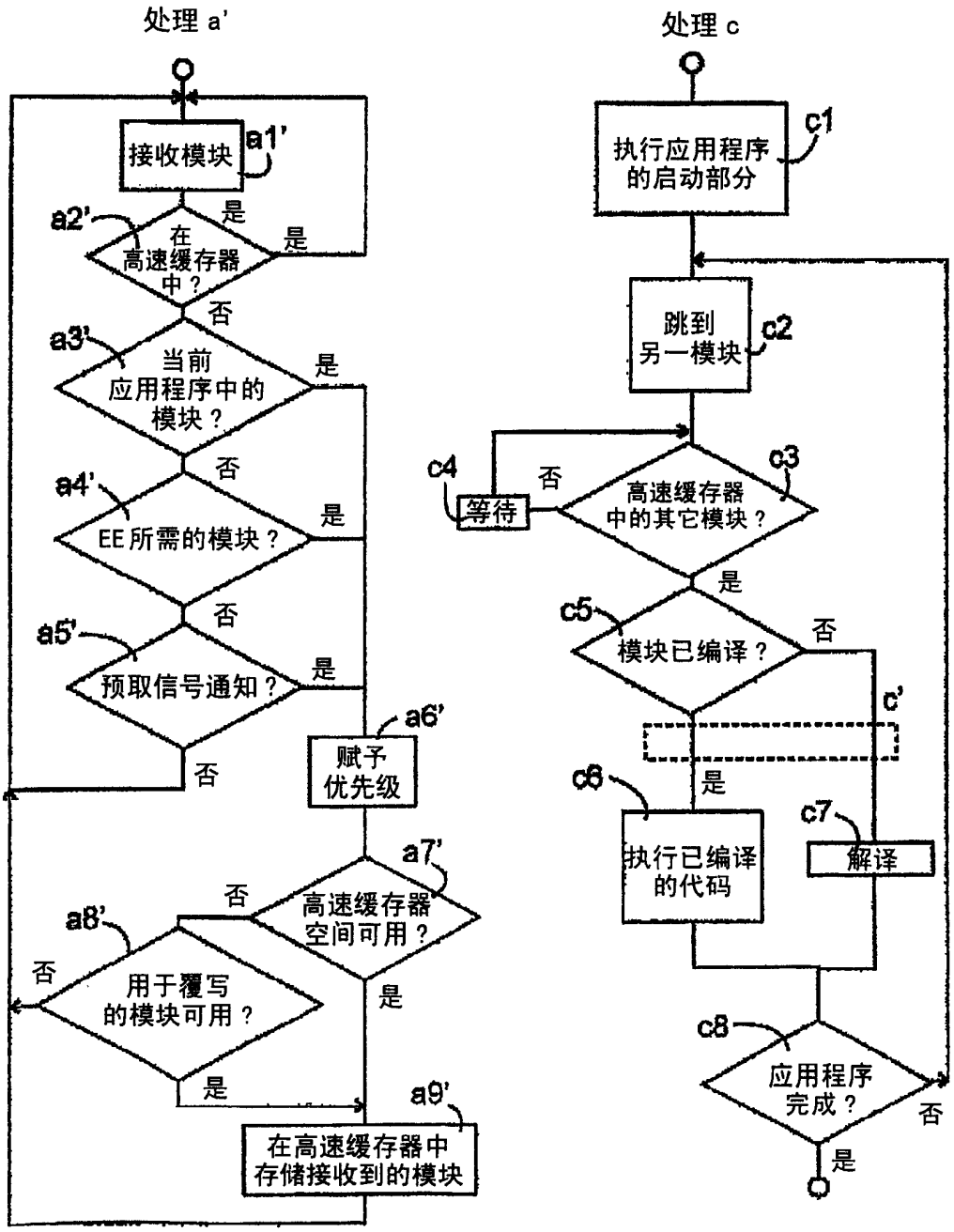

处理 a'
接收模块
a1'
a2'
在高速缓存器中？
是
否
a3'
当前应用程序中的模块？
是
否
a4'
EE所需的模块？
是
否
a5'
预取信号通知？
是
否
a6'
赋予优先级
a7'
高速缓存器空间可用？
否
是
a8'
用于覆写的模块可用？
否
是
a9'
在高速缓存器中存储接收到的模块
处理 c
执行应用程序的启动部分
c1
跳到另一模块
c2
c3
高速缓存器中的其它模块？
否
c4
等待
是
c5
模块已编译？
否
是
c'
c6
执行已编译的代码
c7
解译
c8
应用程序完成？
否
是

图 3

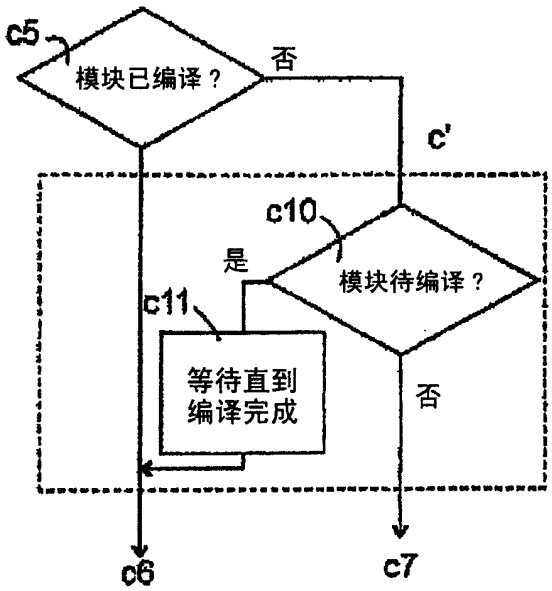

c5
模块已编译？
否
c'
c10
是
模块待编译？
c11
等待直到
编译完成
否
c6
c7

图 4A

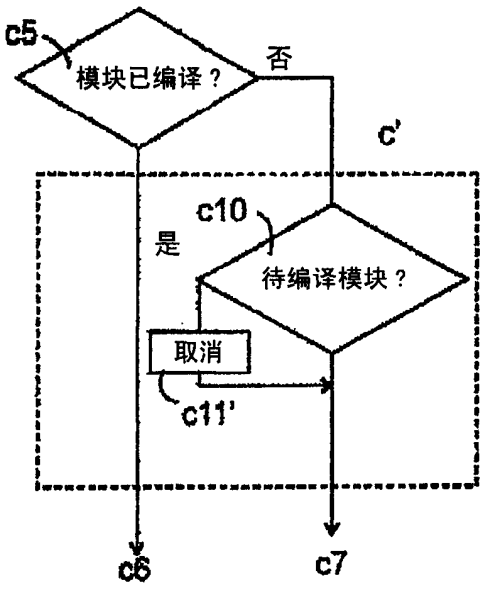

c5
模块已编译？
否
c'
c10
是
待编译模块？
取消
c11'
c6
c7

图 4B

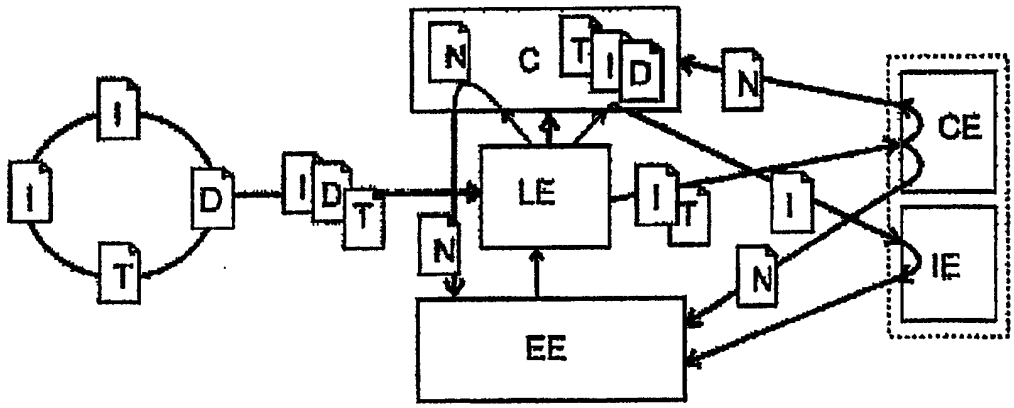

N
C
T
I
D
N
CE
I
D
I
T
D
T
LE
I
T
I
N
N
IE
EE

图 5