[19] 中华人民共和国国家知识产权局

[51] Int. Cl.
G06F 11/00 (2006.01)
G06F 7/58 (2006.01)

[12] 发明专利说明书

专利号 ZL 01103212.X

[45] 授权公告日 2007 年 3 月 7 日

[11] 授权公告号 CN 1303529C

[22] 申请日 2001.2.5 [21] 申请号 01103212.X

[30] 优先权

[32] 2000. 2. 8 [33] US [31] 09/500345

[73] 专利权人 国际商业机器公司

地址 美国纽约州

[72] 发明人 小 G·W·威廉

[56] 参考文献

US4310720 1982.1.12

CN1201545A 1998.12.9

CN1159108A 1997.9.10

审查员 刘 栩

[74] 专利代理机构 中国专利代理(香港)有限公司

代理人 程天正 梁 永

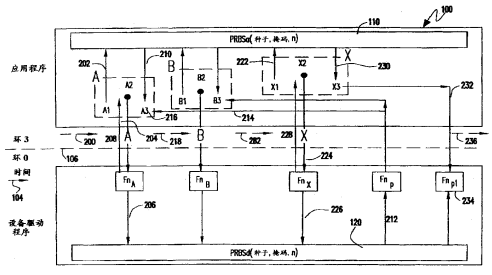
权利要求书 2 页 说明书 10 页 附图 5 页

[54] 发明名称

利用伪随机数二进制序列程序互锁来防止窜改

[57] 摘要

通过调用以排它方式存在处理器及其外围设备的一个不同部分中的功能和来自另一不同部分的驱动程序来防止窜改，以便防止观察软件各部分的执行过程。 伪随机二进制序列发生器与各不同部分相关并分别随调用和执行前述功能而递增。 在允许或阻止进一步执行软件之前比较伪随机二进制序列值。 可在任何特权级下进行比较。 可在一种特权级下用 PRBS 值来破坏数据或程序文件并执行逆功能以便用本身的 PRBS 值在另一种特权级下恢复数据或程序文件。



1、一种防止窜改在处理器上运行的软件的方法，所述处理器具有多个不同的部分，其中第一不同部分可以执行在第二不同部分中不存在的功能，所述方法包括下列步骤:

使与上述第二不同部分相关的第一伪随机二进制序列发生器递增;

从所述第二不同部分中调用上述第一不同部分的所述功能;

使与上述第一不同部分相关的第二伪随机二进制序列发生器递增;

执行所述功能;

对所述第一和第二伪随机二进制序列发生器中的伪随机二进制序列值进行比较; 以及

根据比较步骤的结果控制所述软件的执行。

2、如权利要求 1 的方法，其特征在于，所述功能限于使上述第一不同部分相关的第二伪随机二进制序列发生器递增。

3、如权利要求 1 的方法，其特征在于，所述第一不同部分中存在的另一个功能调用来自上述第二不同部分相关的第一伪随机二进制序列发生器中的伪随机二进制序列值并在所述第一不同部分中执行上述比较步骤。

4、如权利要求 3 的方法，其特征在于，所述另一个功能使上述第一不同部分相关的第二伪随机二进制序列发生器递增，并且，在调用上述另一功能时，使所述第二不同部分相关的第一伪随机二进制序列发生器递增。

5、如权利要求 2 的方法，其特征在于，所述第一不同部分中存在的另一个功能调用来自上述第二不同部分相关的第一伪随机二进制序列发生器中的伪随机二进制序列值并在所述第一不同部分中执行上述比较步骤。

6、如权利要求 5 的方法，其特征在于，所述另一个功能使上述第一不同部分相关的第二伪随机二进制序列发生器递增，并且，在调用上述另一功能时，使所述第二不同部分相关的第一伪随机二进制序列发生器递增。

7、如权利要求1的方法，其特征在于，该方法还包括下列步骤：

用一个算法以及所述第二不同部分相关的第一伪随机二进制序列发生器中的伪随机二进制序列值来改变所述软件的数据；以及

用具有其功能是上述算法的逆算法的另一个算法以及所述第一不同部分相关的第二伪随机二进制序列发生器中的伪随机二进制序列值来改变所述软件的数据；

从而，在所述第一不同部分和所述第二不同部分相关的第一和第二伪随机二进制序列发生器的伪随机二进制序列值相同时，可恢复用上述算法改变的数据。

8、如权利要求1的方法，其特征在于，所述不同部分具有不同的特权级。

9、一种防止窜改在处理器上运行的软件的方法，所述处理器具有不同的部分，其中第一不同部分可以执行在第二不同部分中不存在的功能，所述方法包括下列步骤：

用一个算法和所述第二不同部分中的伪随机二进制序列值来改变所述软件的数据；以及，

用具有其功能是上述算法的逆算法的另一个算法和所述第一不同部分中的伪随机二进制序列值来改变所述软件的数据；

从而，当所述第一不同部分和所述第二不同部分的伪随机二进制序列值相同时，恢复由上述算法改变的数据。

10、如权利要求9的方法，其特征在于，所述不同部分具有不同的特权级。

利用伪随机数二进制序列程序互锁来防止窜改

技术领域

本发明在总体上涉及计算机软件应用程序的安全措施，具体地说，本发明涉及用于防止非法人员观察应用程序执行过程的装置。

背景技术

在过去的几年中，计算机应用软件的数量和多样性大大增加。计算机硬件的发展使得这些应用程序的功能有极大的扩展，从而，必须要写入的代码量和其它开发成本有极大的增长。在同一时期内，软件销售商对因非法复制这类程序而可能损失的价值的认识也有所提高。因此，目前软件销售商的通常作法是要求注册各个程序拷贝或者是出售用于特定用户的定点许可，并在程序中包括这样的特性，使得能够防止除已对该程序拷贝注册过的人员以外的人运行应用程序或应用程序中的主要部分。

软件注册一般是通过以下方法来强制执行的：如果不依照注册来授权访问，则使软件的大部分无效。以相同方式，为了促销软件，通常可通过在线方式将呈下述形式的软件提供给用户：所述软件具有与演示拷贝相类似的减少了的功能，但这种软件能有效地运行，足以使潜在的购买者决定是否购买。然后，在购买、付款和注册时以远程的方式使得在初始时被禁止的功能生效。按这种可转换的形式发布软件还能提供这样的优点即：可以避免需要生产如演示拷贝那样的独立版本并能避免一个拷贝破坏另一个拷贝的可能性。

但是，尽管这种性能业已发展到了一定程度的完善性和复杂性并取得了良好的安全性，但可以通过修改应用程序以绕过这些性能而使任何的这种性能失去作用，这一点是不言自明的。使受保护的软件的功能复原并访问受保护的功能的方法一般称为“黑客攻击（hacking）”，它一般用于使口令保护、用户识别或验证等失去作用。修改程序的方法通常称为“打补丁（patching）”。使得难以通过黑客攻击行为来修改程序的特性称为“防窜改”。当然，对诸如数据库之类的多种软件以及应用程序来说，“防窜改”是应有的特性。

形成防窜改软件通常涉及到包括在处理器内的特权级以及所谓的特权指令的概念。一般需要特权指令以便允许访问处理器体系结构的受限性能。例如，特权指令或功能可包括加载和卸载处理器中的受保护的专用调试寄存器、勾连中断向量和处理关键的操作系统（ＯＳ）存储区以及输入／输出（Ｉ／Ｏ）功能。这些特权功能在低特权级下受控于应用程序（例如由应用程序来请求），但实际上在应用程序的正常操作过程中要通过更高特权操作系统组件在较高的特权级下由其它结构或软件所执行。具体地说，Ｉ／Ｏ功能的执行受限于ＯＳ的最高特权级并且不能由在较低特权级下操作或执行的应用程序来执行，而只能由该应用程序来请求。

也就是说，为了使得应用软件更安全、符合软件开发的要求，通常需要开发特权伴随软件模块（通常呈设备驱动程序的形式），这种软件模块可按与应用程序相串行的方式工作并且能代表有较低特权的应用程序来执行上述特权操作。当应用程序需要（或者，应该需要，这取决于编程者）验证系统的完整性或需要确定是否出现了某些类型的窜改时，应用程序就可调用上述特权模块以进行上述维护。不幸的是，这种组织结构及其较为广泛的应用使得应用程序与特权模块之间的接口成为企图使安全特征失去作用的逻辑攻击点。例如，撤掉与安全性有关的对特权模块的调用在许多情况下会使得它们要执行的防窜改性和完整性检查功能失去作用。

尽管没有一种防窜改的形式是完全安全的，但是，一般都认为，应用程序和相应特权模块中以这种方式（例如通过包括更多的技术、陷阱和相关性）提供的复杂程度越大，在给定的时间量和工作量内使应用程序和相应特权模块失去作用从而危及应用程序或数据完整性或获得包含在其中的秘密的成功可能性就越低。不幸的是，这种复杂性还会使软件应用程序的可靠性和／或稳定性（或者例如数据库和／或错误恢复系统的牢靠性）以及软件本身的执行速度和效率打折扣。所以，迄今为止，在应用程序执行效率与应用程序本身及其数据的安全性之间存在着不可避免的折衷。

发明内容

所以，本发明的一个目的是提供一种通过检测窜改并在检测到窜

改时终止执行文件和／或使文件破坏来提高软件安全性的技术。

本发明的另一个目的是在不使软件执行效率明显打折扣的情况下提供软件的有效安全性。

本发明的又一个目的是防止非法人员观察软件执行过程。

本发明的还一个目的是防止以非法的方式使软件中被禁止的或受保护的部分变为有效、或者观察这些被禁止的或受保护的部分。

为了实现本发明上述和其它目的，提供了一种防止窜改在处理器上运行的软件的方法，所述处理器具有多个不同的部分，第一不同部分可以执行在第二不同部分中不存在的功能，以及提供一个含有用于执行上述方法的命令的软件，所述方法包括下列步骤：使与所述第二不同部分相关的第一伪随机二进制序列递增；从第二不同部分中调用第一不同部分的功能；使与所述第一不同部分相关的第二伪随机二进制序列发生器递增；执行所述功能（该功能可包括对伪随机序列的值进行比较或者除了使伪随机二进制序列发生器递增以外还可包括或不包括一种或多种操作）；对所述第一和第二伪随机二进制序列发生器中的伪随机二进制序列值进行比较；以及，根据比较的结果控制所述软件的执行。

依照本发明的另一个方面，提供了这样一种防止窜改在处理器上运行的软件的方法，所述处理器具有多个不同的部分，一个不同部分可以执行在第二不同部分中不存在的功能，以及提供含有用于执行上述方法的命令的软件，所述方法包括下列步骤：用一个算法和第一不同部分中的伪随机二进制序列值来改变所述软件的数据；以及，用逆算法和在上述第一不同部分中伪随机二进制序列值来改变所述软件的数据；从而，当所述不同部分和所述另一个不同部分的伪随机二进制序列值相同时，可恢复由上述算法改变的数据。

附图说明

参照附图从以下对本发明最佳实施例的详细说明中可以更好地理解上述和其它目的、方面和优点，在附图中：

图 1 是本发明最佳实施例的高层框图，它说明了该实施例的操作；

图 2 A和 2 B 包括说明图 1 所示的本发明实施例的操作的流程

图；

图 3 是说明图 1 的实施例的优选增强形式的高层框图；

图 4 是说明图 3 所示的本发明增强形式的流程图.

具体实施方式

参照附图，具体参照图 1 、 2 A和 2 B ，以高层概略框图的形式

示出了本发明的实施例１００．应该注意，如箭头１０４所示，在图１中设置了任意的时间标度（从左至右），其中，实框表示可用硬件或软件来实现的功能元素，而用虚线表示的框则表示应对其进行保护的软件序列．在图２Ａ和２Ｂ中，所述序列自上向下进行并且以与图１用虚线１０６来区分特权级的相同的方式将在其上执行每一个所述功能的软件分隔到左方或右方．因此，图１与图２Ａ和２Ｂ按不同的重点和观点说明了同样的主题．在可能的情况下使用相对应的标号．（在图３和４中也分别使用了同样的制图约定．）

正如本领域的技术人员所认识到的那样，伪随机二进制序列（ＰＲＢＳ）发生器是公知的，可按多种方式加以使用和实现．一般地说，ＰＲＢＳ发生器被实现为一个逻辑阵列，它可包括多个存储设备．ＰＰＢＳ发生器的一种常用形式使用了串行连接的触发器串，它们呈移位寄存器的形式．业已表明在具有不同长度的这种移位寄存器中，某些存储级的输出可按原码值或补码值反馈给其它存储级并且可与其它存储级的输出（更一般地称为掩码）作逻辑组合，可以生成一长串逻辑值，它们具有基本上与纯随机数值相同的统计学属性．当然，可用软件以本技术中周知的方式来模拟这种硬件实现形式，并且，通常都进行这种模拟．（事实上，就实施本发明而言，正如以下将详细说明的那样，软件实现形式是更佳的．）

相对所提供的存储级的数量而言，可以把ＰＲＢＳ发生器的重复周期做成极长．可以从单个存储级中串行地取出ＰＲＢＳ发生器的输出，或者从内部抽头中按任意长度（多达存储级的数量）的字节并行地取出ＰＲＢＳ发生器的输出．尽管ＰＲＢＳ在多个有任意长度的二进制数字之后重复，但在开始时作为例如存储设备的初始存储状态提供给ＰＲＢＳ发生器的特定值可以使所述序列的起点始于任何点．所述初始状态总称为种子值．所以，可以从ＰＲＢＳ发生器得出的唯一序列的数量等于重复序列中的位数．

此外，ＰＲＢＳ发生器中的各个不同的逻辑阵列的互连或掩码（例如各相应存储级的反馈和逻辑组合）可提供一唯一的二进制序列．但是，同样原因，同样的二进制序列仅可源于功能相同的ＰＲＢＳ发生器，这些发生器有同样的掩码、存储级数量并用同样的种子值

来初始化．所以，ＰＲＢＳ序列在不事先知道这些参数的情况下难以复制．

本发明在原理上使用了两个相同的ＰＲＢＳ发生器１１０和１２０（或有同样掩码和存储级ｎ的ＰＲＢＳ发生器），这两个ＰＲＢＳ发生器用相同种子值来初始化它们在请求和执行特权调用时分别递增．这两个相同ＰＲＢＳ发生器中的一个发生器被提供给所述各特权级的每一级，或者可以被各特权级访问，在所述的各特权级之间，可依照本发明的原理防窜改．在更高特权级或更低特权级上（或者在不同的特权级根据各相应特权级中的ＰＲＢＳ所执行的其它功能中）比较来自各ＰＲＢＳ发生器的ＰＲＢＳ，并且，在所述相应ＰＲＢＳ信号不相等时中断执行并阻断更高特权功能的结果．这种基本的操作模式可非常快地执行，并且，即使是经常地重复也不会显著地使软件的执行效率打折扣，而且，能如以下参照图３和４详细说明的那样很容易地得以增强．

参照图１以及图２Ａ和２Ｂ，应该明白，这里示出了两种特权级，每种特权级均有相应的ＰＲＢＳ发生器１１０或１２０．这两种特权级被命名为与Windows-Intel环境术语相一致的环３（较低的特权）和环０（较高的特权）．但是，应该认识到，本发明完全适用于其中不同功能可在不同特权级下加以执行或者以其它方式限于由专用的独立硬件、软件或固件来加以执行的任何环境．例如，可能需要用于执行命令的专门数据路径或寄存器．

也就是说，就成功地实施本发明以达到其预定目的而言，仅需要处理器或其外围设备或所用的驱动程序有（或可配置成有）不同的操作部分，这些部分可与离散的相同（或同样配置和初始化）的ＰＲＢＳ发生器相关，并且，一个这样不同部分需要能执行在另一个不同部分上得不到的功能．因此，术语“不同部分” 应为理解为包括满足上述条件的所有这种可能结构．

这方面，应该明白，用软件来实现ＰＲＢＳ发生器在理论上是优选的，因为，在较低的特权级下对硬件访问可能被拒绝，从而需要ＰＲＢＳ发生器的软件实现形式．此外，ＰＲＢＳ发生器的软件实现形式有助于提供相同的ＰＲＢＳ发生器并且有助于在应用程序和其伴

随设备驱动程序或其它较高特权级软件中对这些发生器的初始化．还应注意，本发明可以并且应该用于对ＰＲＢＳ发生器本身进行保护．

就以下的说明而言，假定应用程序或其它软件正在执行尚未按本发明加以保护的功能，如虚箭头２００所示。还假定ＰＲＢＳ发生器１１０、１２０已被初始化成相同的种子值、相同的掩码以及相同的值ｎ，从而，ＰＲＢＳ发生器１１０、１２０会输出相同的值并且可被看作是同步的．

在程序指令列表中在应加以保护的程序部分之前的某一点上插入操作系列Ａ（和／或Ｂ和／或Ｘ）．这些操作不一定要按紧挨着的次序执行而是介于它们之间可以具有其它操作，以便使这些操作的目的不大可能被非法人员所识别．为了能说明本发明，将指令序列Ａ表示为包括三项操作Ａ１、Ａ２和Ａ３．这些操作中的每一项操作均可包括多个命令．

当在程序中达到序列Ａ以便保护一个或多个按不同特权级或用专用软件加以执行的后续操作（例如Ａ２）时，步骤Ａ１如标号２０２所示那样使ＰＲＢＳ发生器递增并前进至受保护的步骤Ａ２，该步骤如标号２０４所示那样在特权环０（例如在设备驱动程序中）调用预定的功能Ａ（Fn_A）．然后，功能Ａ使ＰＲＢＳ发生器１２０递增．功能Ａ执行预定的操作并如标号２０８所示返回至应用程序．

然后，应用程序前进至操作Ａ３并如标号２１０所示取出ＰＲＢＳ发生器１１０的输出且如标号２１２所示调用Fn_p以取出ＰＲＢＳ发生器１２０的输出，而且如标号２１４所示将所述输出返回给应用程序．然后，操作Ａ３比较ＰＲＢＳ发生器１１０和１２０的输出，由于所述发生器两者在开始时是同步的并且两者在序列Ａ期间均递增一次，故比较是成功的，应用程序如标号２１８所示继续进行．

序列Ｂ以类似的方式操作，因而不需要单独加以说明．如果在操作Ｂ３（与操作Ａ３相对应）中对ＰＲＢＳ发生器输出的比较是成功的，则应用程序如标号２２２所示那样继续．

但是，如果因为应用程序尚未注册，操作Ａ２尚未被授权，从而对环０或其中的任何功能（或要由功能执行的操作）的访问均会被拒绝，那么，应用程序就不能继续．如果非法用户试图通过撤掉步骤Ａ

1（200）来使拒绝访问失去作用，则步骤A2会使PRBS发生器120递增，而PRBS发生器110不会递增，从而A3（216）处的比较会失败.

相反，如果通过在操作A期间撤掉使PRBS发生器120递增而试图使拒绝访问失去作用，则所述比较也会失败.通过撤掉功能A（例如通过删除或不开放操作A2）试图使拒绝对应用程序后续部分的访问失去作用（例如，对代码进行比较）也会导致前述比较失败，因为，PRBS发生器110会递增，而PRBS发生器120不递增.通过插入使PRBS发生器120或110递增或递减而试图使拒绝访问失去作用也会使A3处的比较失败，因为，依照本发明，Fn_A和其它特权功能会在执行所述功能之前使PRBS发生器120递增.

即使执行了特权功能，也会在将结果报告给应用程序之前进行以较.在比较216失败时，就会以编程者所希望的任何方式例如通过转换至任何预定的子例程（包括从存储器或者甚至从将应用程序加载进计算机的存储介质中删除应用程序或者删除该应用程序中或多或少的关键部分）来终止应用程序的执行.

必须同时且适当地执行至少两种修改，才能使拒绝对特权功能的访问失去作用，而事先不知道应用程序执行过程的非法用户要成功地作到这一点几乎是完全不可能的.应该记得，对系统执行过程的观察正是本发明所要防止的一种特别类型的窜改.所以，应该认识到，尽管本发明在应用程序中即使多次使用或重复的情况下仍能简单和迅速地执行，但本发明仍能相当有效地防止将应用程序的执行过程暴露给非法用户去窜改.

就实施本发明的基本原理而言，在哪里进行比较或者是否在比较时执行额外的也是特权功能的功能并不重要.但是，通过如以下就图1和图2B中受保护的操作X所述那样执行这种额外的功能可以提供某种额处的保护.在这种情况下，操作X1如以前在操作A1中如标号222所述那样使PRBS发生器110递增.操作X2调用Fn_X（224）并如以前那样使PRBS发生器120（226）递增.同样，按上述方式再返回至应用程序（228）.

然后，操作X 3从P R B S发生器1 1 0中取出（2 3 0）P R B S值，但这时调用操作 $Fn_{p'}$ 且将P R B S值从P R B S发生器1 1 0提供（2 3 2）给设备驱动程序．然后特权功能 $Fn_{p'}$ 从P R B S发生器1 2 0中取出P R B S值并进行比较．$Fn_{p'}$ 不使P R B S发生器1 2 0进一步递增，或者，X 3和 $Fn_{p'}$ 两者都可设置成使P R B S发生器1 1 0和1 2 0分别递增．除了是在较高的（或者至少是不同的）特权级下进行比较以外，上述操作与操作序列A相类似．但是，在实践中，与操作A 3和其它特权功能相比，提供了更多的位置，在这些位置处，非法人员会不适当地进行修改，并且，在X 3或 $Fn_{p'}$ 处提供了通常的不一致性．

以下参照图3和4说明本发明的增强形式．尽管本发明的上述实施例会在对P R B S输出的比较失败之后开始终止应用程序或者毁坏或破坏文件，但图3和4所示的本发明增强形式则将文件破坏和修复包括为防窜改的一部分，从而在性能上更牢靠并且会在检测到窜改之前的更早时间采取保护动作．现将所述增强形式说明为上述受保护的操作A的变化形式A′．

如以前那样执行操作A 1，以便使P R B S发生器1 1 0递增．但是，如标号4 0 2所示，递增后的P R B S返回给应用程序．操作A 2′如标号4 0 4所示利用该P R B S值以便用算法X和来自P R B S发生器1 1 0的当前P R B S值有意地毁坏或破坏数据和／或应用程序代码．然后，操作A 3′调用特权功能 $Fn_{A'}$，以便在使P R B S发生器1 2 0递增且返回该新的P R B S值之后执行操作A′．然后，使用来自P R B S发生器1 2 0的新P R B S值，该值目前与P R B S发生器1 1 0的值相同，并利用来自P R B S发生器1 2 0的P R B S值和算法X的逆算法X′重构在标号4 0 4处破坏的数据或代码，如标号4 1 0所示．

在这种情况下，不需要单独的比较，因为，对数据或代码的不正确的重构会终止应用程序或返回一个不正确和无意义的结果（因为数据或代码实际上是被随机二进制代码序列所破坏的）．此外，在操作A′期间不需要执行对数据进行破坏和重构以外的特权操作，因而实际上试图观察程序执行过程的非法人员不能从应用程序中获得信

息。所述增强形式可单独使用或者与特权功能结合使用，所述特权功能利用比较作为诸如上述A、B或X之类的特权功能的特定实例的替代方案。

就以上内容而言，可以看出，本发明尽管能简单能够快速地执行，但是在单个实例中却能实现很高程度的防窜改。可提供本发明的多个实例，并且，所述防窜改过程可具体在变化形式（例如图2B）和增强形式（例如图4）中进行重复，以便在应用程序执行速度不打折扣的情况下提供在实践上难以超越的保护。

尽管就单个的实施例说明了本发明，但是本领域的技术人员应该认识到，可用后附权利要求的精神和范围内的改进形式来实施本发明。

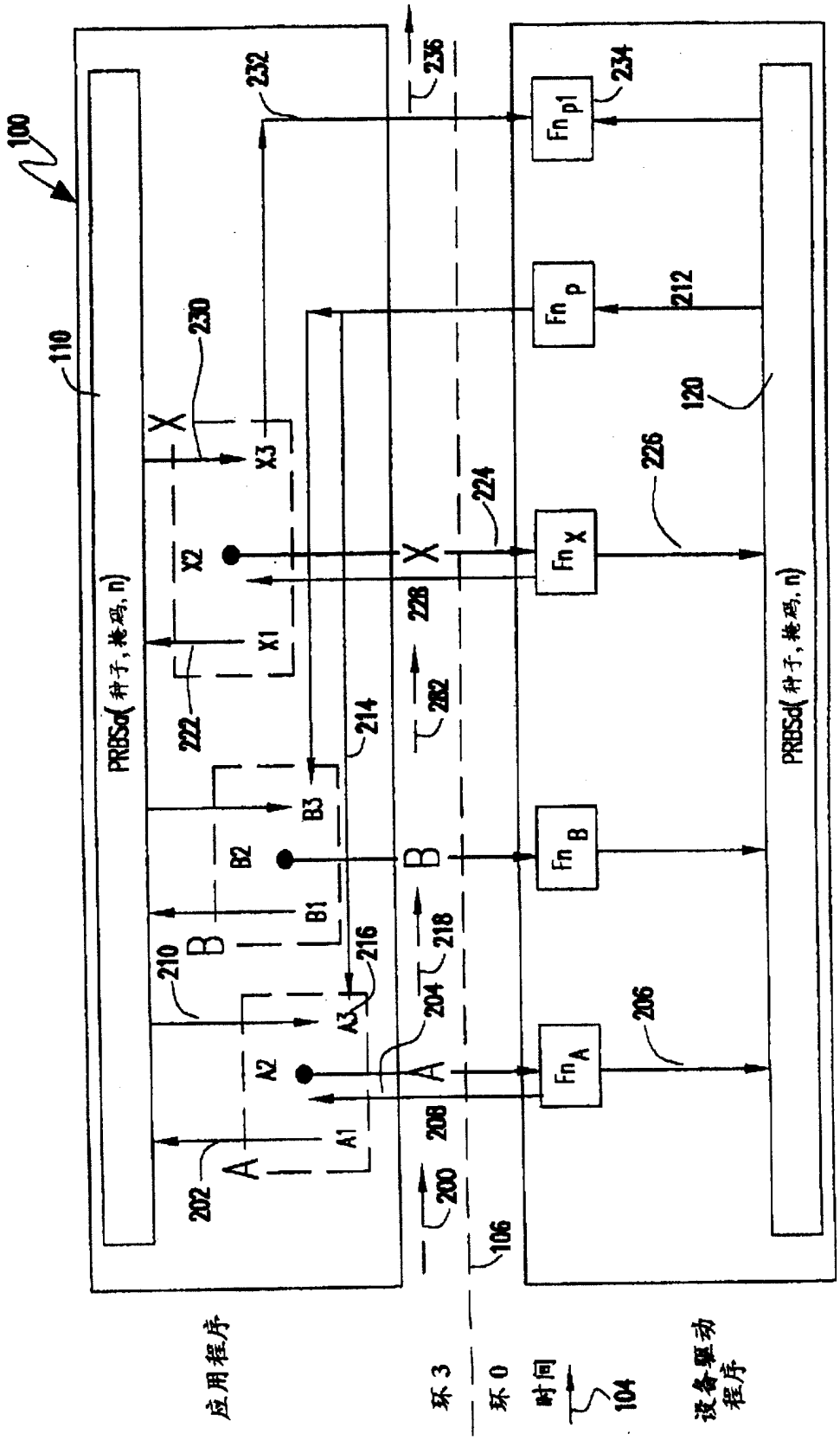

100
110
120
PRBSd(种子,掩码,n)
PRBSd(种子,掩码,n)
232
234
236
230
212
226
224
228
282
222
214
210
216
218
204
206
202
208
200
106
Fn p1
Fn p
Fn X
Fn B
Fn A
X3
X2
X1
B3
B2
B1
A3
A2
A1
应用程序
环 3
环 0
时间
104
设备驱动程序

图 1

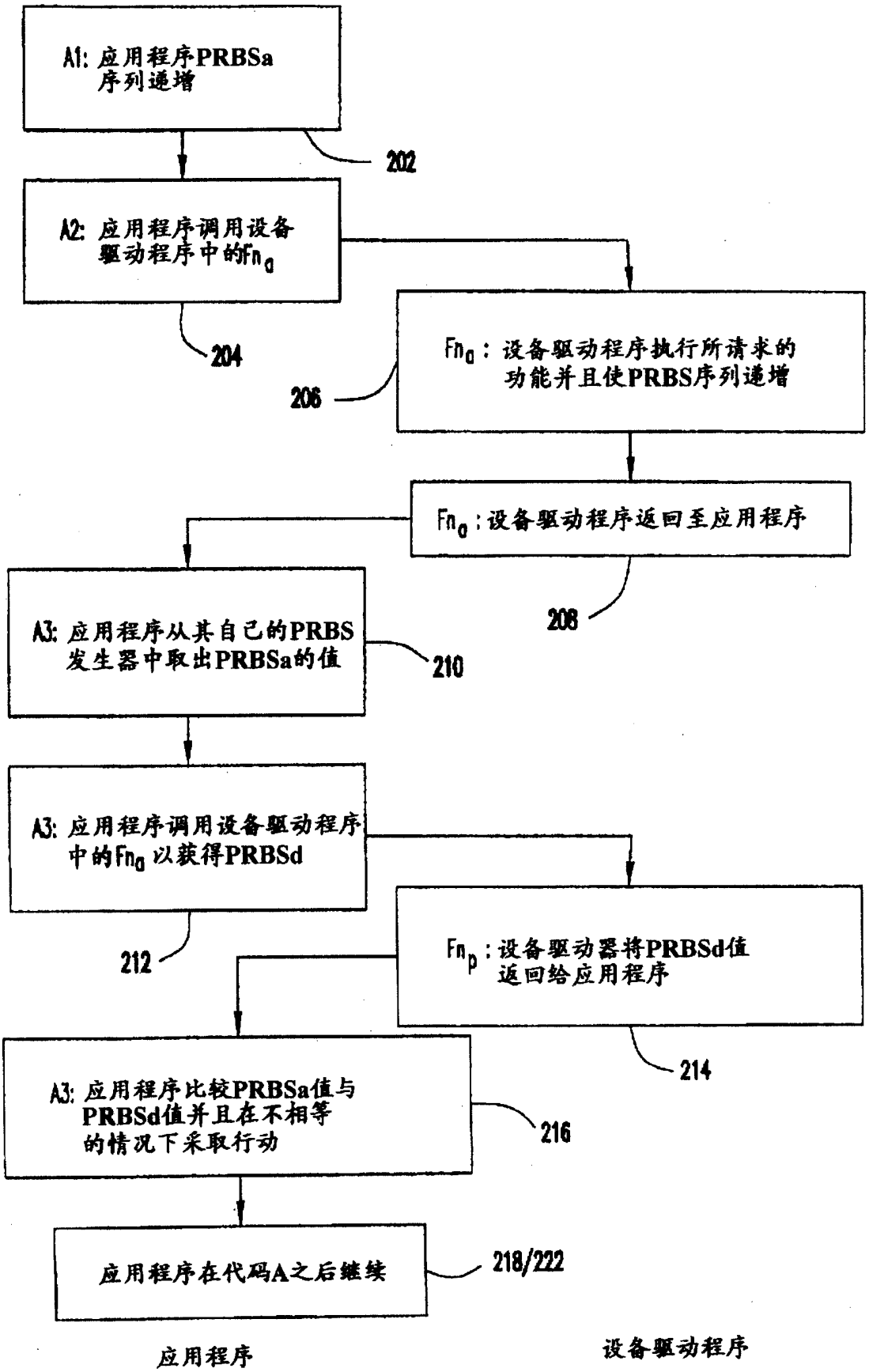

A1: 应用程序PRBSa序列递增
202
A2: 应用程序调用设备驱动程序中的Fn_a
204
206
Fn_a：设备驱动程序执行所请求的功能并且使PRBS序列递增
Fn_a：设备驱动程序返回至应用程序
208
A3: 应用程序从其自己的PRBS发生器中取出PRBSa的值
210
A3: 应用程序调用设备驱动程序中的Fn_a以获得PRBSd
212
Fn_p：设备驱动器将PRBSd值返回给应用程序
214
A3: 应用程序比较PRBSa值与PRBSd值并且在不相等的情况下采取行动
216
应用程序在代码A之后继续
218/222
应用程序
设备驱动程序

图 2A

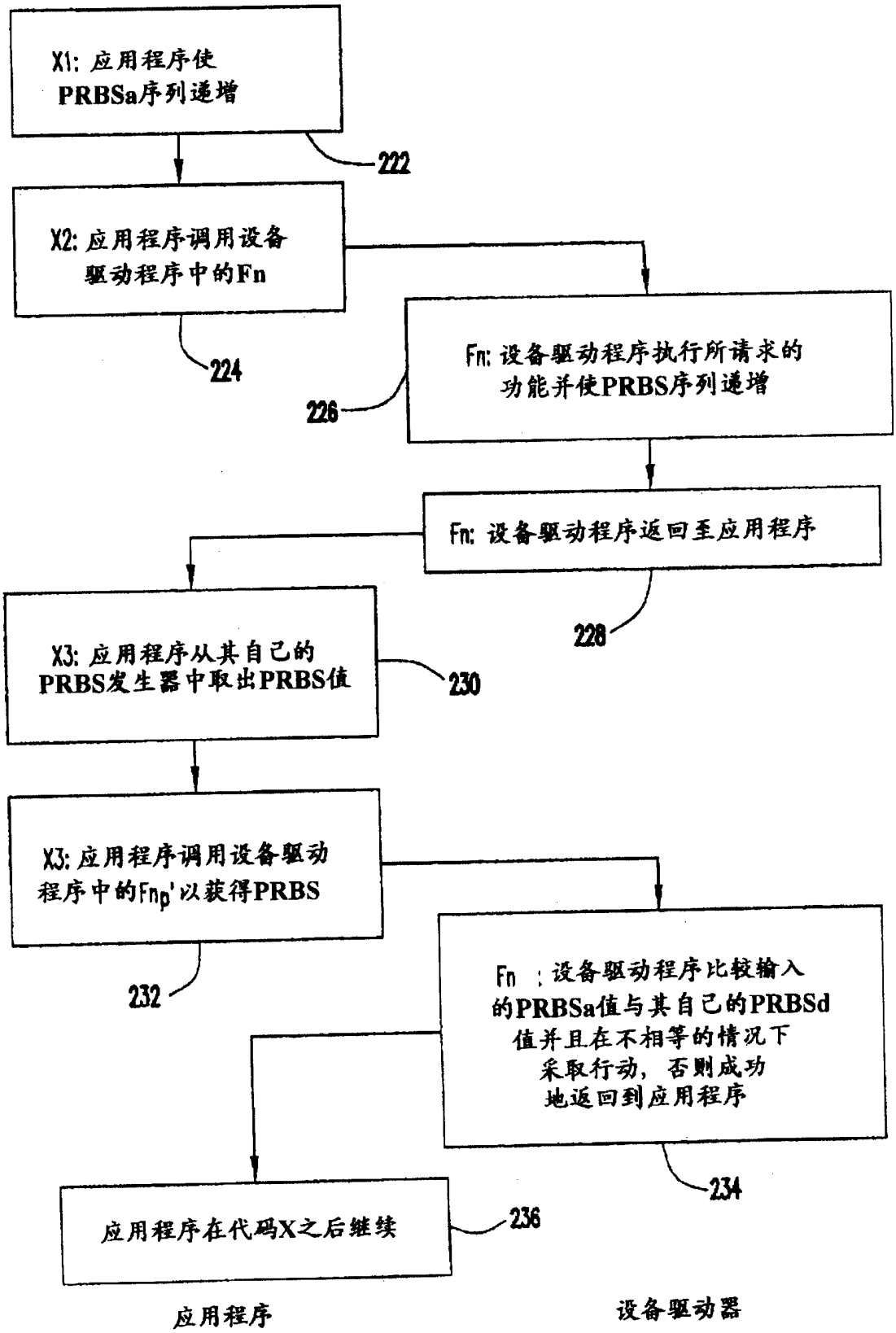

X1: 应用程序使
PRBSa序列递增
222
X2: 应用程序调用设备
驱动程序中的Fn
224
226
Fn: 设备驱动程序执行所请求的
功能并使PRBS序列递增
Fn: 设备驱动程序返回至应用程序
228
X3: 应用程序从其自己的
PRBS发生器中取出PRBS值
230
X3: 应用程序调用设备驱动
程序中的Fnp'以获得PRBS
232
Fn ：设备驱动程序比较输入
的PRBSa值与其自己的PRBSd
值并且在不相等的情况下
采取行动，否则成功
地返回到应用程序
234
应用程序在代码X之后继续
236
应用程序
设备驱动器

图 2B

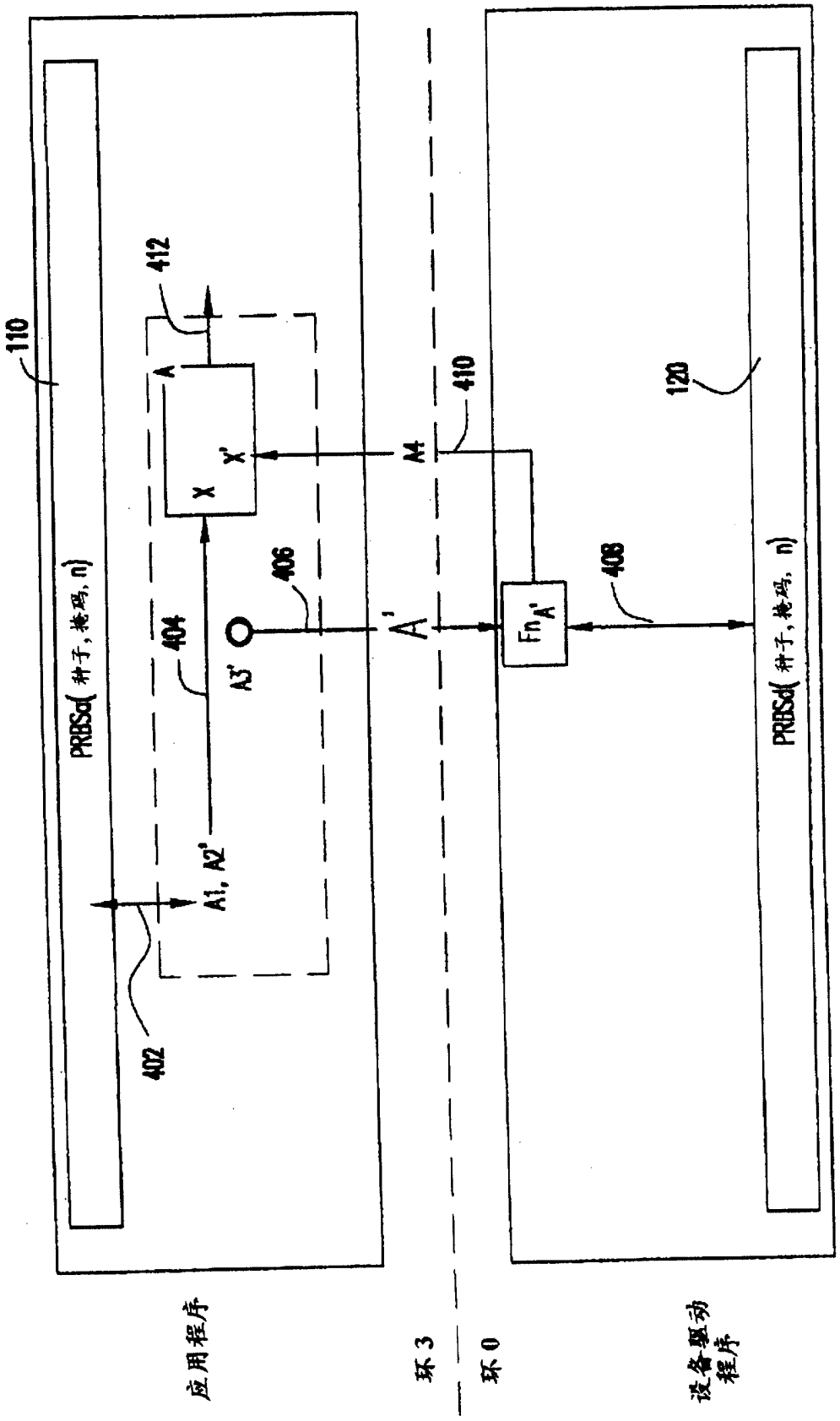

110
412
A
X
X'
410
A4
120
406
404
A3'
A'
408
$Fn_{A'}$
402
A1, A2'
PRBSd(种子,抽码,n)
PRBSd(种子,抽码,n)
应用程序
环 3
环 0
设备驱动程序

图 3

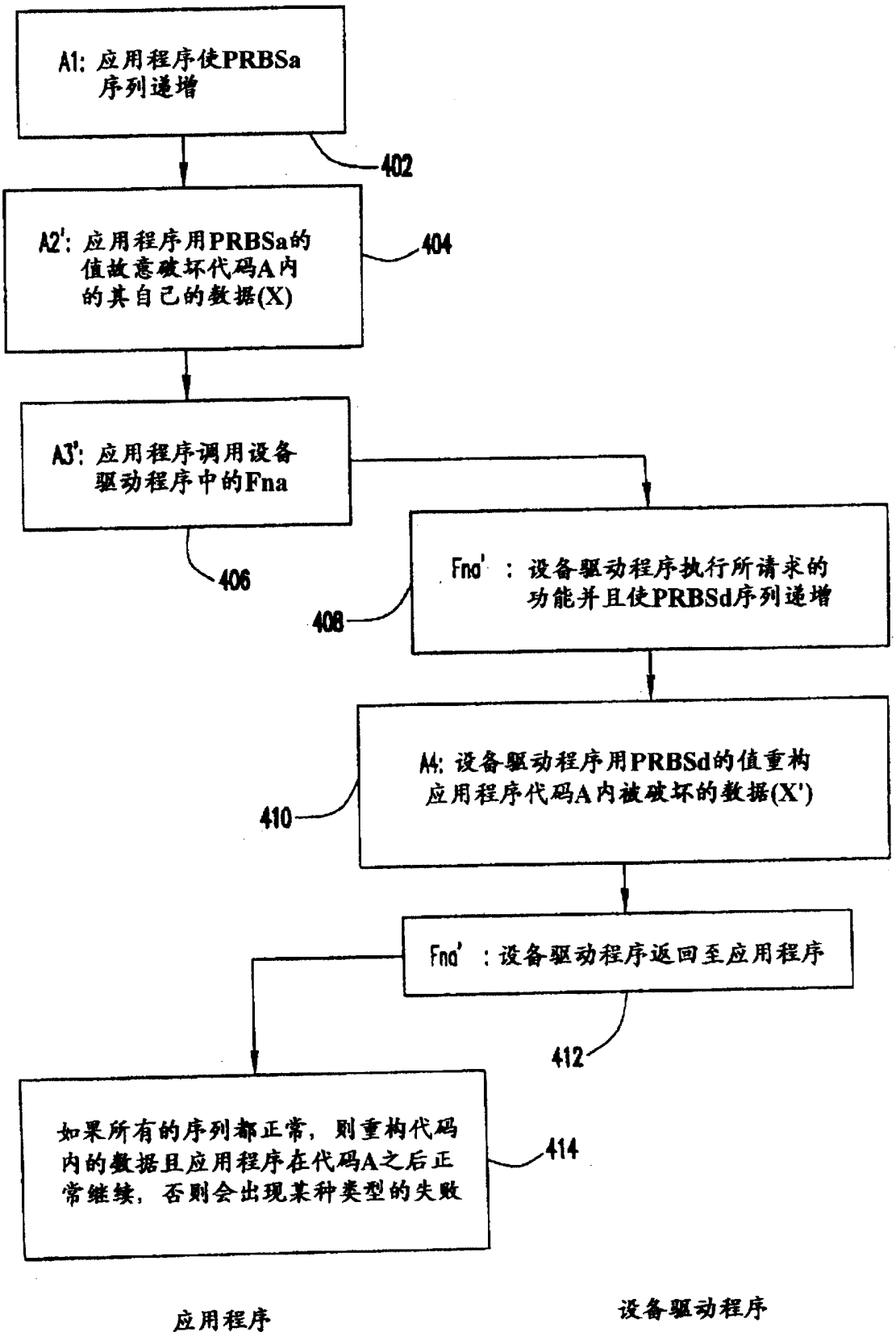

A1: 应用程序使PRBSa
序列递增
402
A2': 应用程序用PRBSa的
值故意破坏代码A内
的其自己的数据(X)
404
A3': 应用程序调用设备
驱动程序中的Fna
406
Fna' : 设备驱动程序执行所请求的
功能并且使PRBSd序列递增
408
A4: 设备驱动程序用PRBSd的值重构
应用程序代码A内被破坏的数据(X')
410
Fna' : 设备驱动程序返回至应用程序
412
如果所有的序列都正常，则重构代码
内的数据且应用程序在代码A之后正
常继续，否则会出现某种类型的失败
414
应用程序
设备驱动程序

图 4