

(12) 发明专利申请

(10) 申请公布号 CN 102667716 A

(43) 申请公布日 2012. 09. 12

(21) 申请号 200980162575. 0

G06F 15/16(2006. 01)

(22) 申请日 2009. 12. 18

(85) PCT申请进入国家阶段日
2012. 05. 24

(86) PCT申请的申请数据
PCT/US2009/068701 2009. 12. 18

(87) PCT申请的公布数据
W02011/075139 EN 2011. 06. 23

(71) 申请人 惠普发展公司, 有限合伙企业
地址 美国德克萨斯州

(72) 发明人 约翰·朗德里

(74) 专利代理机构 北京德琦知识产权代理有限公司 11018
代理人 于未茗 罗正云

(51) Int. Cl.
G06F 9/44(2006. 01)
G06F 9/22(2006. 01)

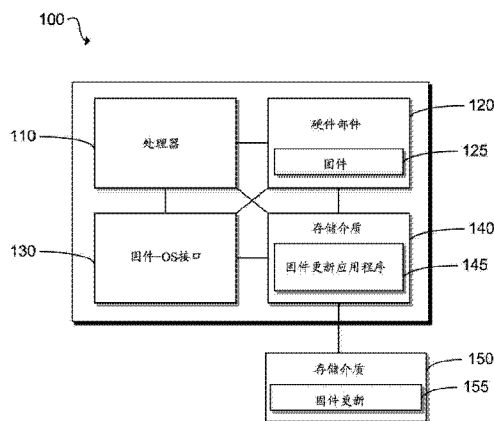
权利要求书 2 页 说明书 9 页 附图 7 页

(54) 发明名称

使用固件更新应用程序更新部件的固件的方法及装置

(57) 摘要

示例实施例涉及对在计算装置中包括的硬件部件的固件进行更新的方法。示例方法可以将用于硬件部件的可执行的固件更新存储在机器可读存储介质上。然后该方法可以在计算装置启动时由固件-操作系统(OS)接口发起固件更新应用程序。最后,该方法可以通过开始执行可执行的固件更新来触发硬件部件的固件的更新。还公开了相关计算装置和机器可读存储介质。



1. 一种对计算装置中包括的硬件部件的固件进行更新的方法,该方法包括:
由所述计算装置将用于所述硬件部件的可执行的固件更新存储在机器可读存储介质上;
当所述计算装置启动时,由固件-操作系统(OS)接口发起固件更新应用程序;以及
由所述固件更新应用程序通过开始执行所述可执行的固件更新来触发所述硬件部件的所述固件的更新。
2. 根据权利要求1所述的方法,其中所述固件-OS接口是统一的可扩展固件接口(UEFI)。
3. 根据权利要求1所述的方法,其中:
所述存储是通过在所述计算装置的操作系统(OS)中运行的应用程序执行的,以及
所述方法进一步包括由所述OS设置应当在重启时执行更新的指示。
4. 根据权利要求3所述的方法,其中发起固件更新应用程序是在由所述计算装置的所述固件-OS接口检测到所述指示时执行的。
5. 根据权利要求1所述的方法,其中:
所述固件更新应用程序由所述计算装置的制造商提供,以及
所述可执行的固件更新由所述硬件部件的制造商提供。
6. 根据权利要求5所述的方法,其中所述可执行的固件更新符合由所述计算装置的制造商规定的应用协议,以返回结果给所述固件应用程序。
7. 根据权利要求5所述的方法,其中所述可执行的固件更新由所述硬件部件的制造商通过使用为所述固件-OS接口设计的软件开发工具包(SDK)开发。
8. 根据权利要求1所述的方法,进一步包括:
将所述固件的更新的结果写到所述计算装置的操作系统(OS)能访问的位置;以及
通过在所述OS中运行的应用程序将所述结果报告给所述计算装置的制造商和所述硬件部件的制造商中的至少一个。
9. 根据权利要求1所述的方法,进一步包括:
将用于所述硬件部件的固件备份存储在所述机器可读存储介质上;以及
当检测到所述更新的失败时,使用所述固件备份触发所述硬件部件的所述固件的恢复。
10. 一种计算装置,包括:
处理器;和
机器可读存储介质,被编码有可由所述处理器执行的指令,所述指令包括:
用于在所述计算装置启动期间检测所述计算装置的硬件部件的固件可获得的可执行的固件更新的指令,
用于在检测到可获得所述固件更新时由固件-操作系统(OS)接口发起固件更新应用程序的指令,以及
用于通过在所述固件更新应用程序中开始所述可执行的固件更新来对所述硬件部件的所述固件进行更新的指令。
11. 根据权利要求10所述的计算装置,其中所述可执行的固件更新位于所述机器可读存储介质的对所述计算装置的用户隐藏的分区中。

12. 根据权利要求 10 所述的计算装置,其中所述可执行的固件更新包括访问和更改所述硬件部件的所述固件的指令。

13. 一种机器可读存储介质,其被编码有可由计算装置的处理器的处理器执行以运行固件更新应用程序的指令,所述指令包括:

用于从所述计算装置的固件-操作系统(OS)接口接收对所述计算装置的硬件部件的固件进行更新的指示的指令;

用于响应于更新所述固件的指示来定位可执行的固件更新的指令;和

用于在所述固件更新应用程序中开始执行所述可执行的固件更新以更新所述硬件部件的所述固件的指令。

14. 根据权利要求 13 所述的机器可读存储介质,进一步包括:

用于在开始执行以前使用在所述机器可读存储介质上存储的数字签名验证所述可执行的固件更新的指令。

15. 根据权利要求 13 所述的机器可读存储介质,其中所述机器可读存储介质是也包括可执行的固件更新的外部存储装置。

使用固件更新应用程序更新部件的固件的方法及装置

背景技术

[0001] 台式计算机、笔记本和其它计算装置中使用的硬件部件一般包括固件，固件是硬件和控制该硬件部件的底层功能的可执行指令的结合。固件使计算装置中安装的硬件部件之间能进行交互，因此对计算装置的正常工作是十分重要的。作为示例，硬驱通常包括在通电时配置硬驱、解释和执行来自处理器的指令以及使硬驱安全掉电的固件。

[0002] 与任何程序一样，实现硬件部件的固件时使用的指令易受错误和其它问题影响，这些错误和其它问题经常出现在最终用户购买的部件中。在错误会影响性能或者使部件无法使用的情况下，制造商可能发现其必须发布固件更新来纠正问题。制造商还可以发布固件更新来添加附加特征或者提高部件性能。

[0003] 开发和安装固件更新经常是复杂的容易出错的过程。硬件部件的制造商一般必须花很多时间来为固件更新定制开发安装器。而且一般的固件更新过程需要用户的大量交互，因而负面地影响用户的体验质量并且增加了损坏部件的可能。在不断竞争的个人计算机生意中，对固件更新进行开发、安装和故障检修所需的时间和成本可能对制造商的利润率产生明显影响并导致客户满意度下降。

附图说明

[0004] 在附图中，相同的附图标记表示相同的部件或步骤。下面的具体实施方式参考附图，其中：

[0005] 图 1 是包括机器可读存储介质的计算装置的实施例的框图，该机器可读存储介质被编码有用于执行固件更新应用程序的指令；

[0006] 图 2 是包括固件更新应用程序的计算装置的实施例的框图，该固件更新应用程序用于更新硬件部件的固件；

[0007] 图 3 是用于对硬件部件的固件进行更新的方法的示例的流程图；

[0008] 图 4A 和图 4B 是由操作系统执行的用于对硬件部件的固件进行更新的方法的示例的流程图；

[0009] 图 5A 和图 5B 是在计算装置启动时执行的用于对硬件部件的固件进行更新的方法的示例的流程图。

具体实施方式

[0010] 如上文描述的，对固件更新进行开发、安装和故障检修所需的时间和成本可能给制造商带来大量开支并导致客户满意度下降。因此，如下面描述的，各个实施例涉及允许高成功率同时从固件更新开发者和用户角度看保持简单的固件更新过程。具体地，在一些实施例中，由固件 - 操作系统 (OS) 接口发起的固件更新应用程序对固件更新过程进行管理。固件更新应用程序可以发起可执行的固件更新，在一些实施例中其可以验证更新的可靠性，向 OS 报告结果以及如果更新失败恢复前面的固件映像。因此，固件更新可能在被管理的环境下发生，以保证安全、向用户提供状态更新、允许文件管理以及简化固件更新过程。

当本领域的技术人员阅读和理解下面的描述时,附加实施例以及附加实施例的应用将对本领域的技术人员来说显而易见。

[0011] 在下面的描述中,提到术语“机器可读存储介质”。本文中使用的术语“机器可读存储介质”指的是任何包含或存储可执行指令或其它数据的电子存储装置、磁存储装置、光存储装置或其它物理存储装置(例如硬盘驱动器、闪存等)

[0012] 现在参考附图,附图中相同的附图标记指相同的部件或步骤。图 1 是包括机器可读存储介质 140 的计算装置 100 的实施例的框图,该机器可读存储介质 140 被编码有用于执行硬件更新应用程序 145 的指令。计算装置 100 可以是例如台式计算机、笔记本计算机、服务器、手持计算装置等等。在图 1 的实施例中,计算装置 100 包括处理器 110、硬件部件 120、固件 -OS 接口 130 以及机器可读存储介质 140 和 150。

[0013] 处理器 110 可以是中央处理单元(CPU)、基于半导体的微处理器或任何其它适合于检索和执行在机器可读存储介质 140 和 150 中存储的指令的硬件设备。具体地,处理器 110 可以取得、解码和执行固件更新应用程序 145 和固件更新 155,以执行下面描述的功能。

[0014] 硬件部件 120 可以是任何包括用来实现计算装置 100 的功能的机械元件、磁元件、电子元件和/或电气元件的物理设备。例如,硬件部件 120 可以是硬盘驱动器、固态驱动器、光盘驱动器、视频卡或网卡、笔记本电池或任何其它的与计算装置 100 连接的内部或外部物理设备。应当明白,虽然图中示出计算装置 100 仅包括一个部件 120,但是计算装置 100 可以包括多个部件 120,每个部件包括其自己的固件 125。

[0015] 每个硬件部件 120 可以包括固件 125,固件 125 可以包括非易失性机器可读存储介质,该非易失性机器可读存储介质被编码有可由处理器 110 或者在硬件部件 120 中包括的控制硬件部件 120 的功能的处理器(未示出)执行的指令。例如,固件 125 可以提供从硬件部件 120 读取或向硬件部件 120 写入的功能,管理硬件部件 120 的供电的功能和执行在硬件部件 120 工作期间使用的其它功能。当制造商为硬件部件 120 的固件 125 提供更新时,可以使用下面详细描述固件更新功能施加更新。

[0016] 固件 -OS 接口 130 可以包括在机器可读存储介质上编码的可执行指令,以提供硬件部件 120 的固件 125 和计算装置 100 的操作系统之间的功能链接。具体地,当计算装置 100 通电时,固件 -OS 接口 130 可以执行一系列指令来初始化、配置和测试硬件部件 120 以及加载操作系统。此外,固件 -OS 接口 130 可以发起固件更新应用程序 145 来更新在硬件部件 120 中包括的固件 125。作为示例,固件 -OS 接口 130 可以检测固件更新 155 是可用的,于是其从预定位置发起固件更新应用程序 145。

[0017] 在一些实施例中,固件 -OS 接口 130 是根据统一的可扩展固件接口(UEFI)标准实现的接口,以提供初始化计算装置 100 的服务。作为另一示例,固件 -OS 接口 130 可以是结合基本输入/输出系统(BIOS)实现的 UEFI 接口,使得 BIOS 执行初始配置(例如开机自检),而 UEFI 接口执行余下配置并与 OS 通信。固件 -OS 接口 130 的其它适合实现将对本领域的技术人员来说显而易见。

[0018] 机器可读存储介质 140 可以被编码有用于运行固件更新应用程序 145 的可执行指令。如上文所述,固件更新应用程序 145 可以由固件 -OS 接口 130 发起,以管理固件更新过程。在一些实施例中,在初始化时,固件更新应用程序 145 可以定位固件更新 155,然后执行固件更新 155。此外,固件更新应用程序 145 可以例如在执行前验证固件更新 155、监控执

行、向用户提供反馈(例如完成的百分比或剩余时间)以及判断更新是否成功。

[0019] 在一些实施例中,固件更新应用程序 145 可以由计算装置 100 的制造商提供。例如,固件更新应用程序 145 可以在制造或定制期间包含在硬盘驱动器或存储介质内,使得计算装置 100 可以生来就支持本文中描述的固件更新安装。可选地,在计算装置 100 上运行的应用程序可以自动地或在用户的指示下将固件更新应用程序 145 下载到存储介质 140 上。

[0020] 机器可读存储介质 150 可以编码有用于对硬件部件 120 的固件 125 应用固件更新 155 的可执行指令。例如,固件更新 155 可以包括通过使用由硬件部件 120 的控制器支持的指令访问和修改包含固件 125 的存储介质的指令。作为更具体的示例,当硬件部件 120 是硬盘驱动器时,固件更新 155 可以包括由硬盘驱动器接受的向在硬盘的 ROM 或类似存储器上存储的固件映像的特定部分写入的指令。以此方式,固件更新 155 可以处理对固件 125 的所有访问,同时固件更新应用程序 145 可以管理更新过程。

[0021] 在一些实施例中,固件更新 155 可以由硬件部件 120 的制造商提供,以此方式,硬件部件 120 的制造商可以在固件更新 155 中包括底层指令,这些底层指令在不需要计算装置 100 的制造商知道这些特定指令的情况下访问和更改固件。相反,由于计算装置 100 的制造商可以提供固件更新应用程序 145 来管理更新过程,所以硬件部件 120 的制造商可以避免配置固件更新 155 以实现图形用户接口、接收用户输入、管理文件以及与固件-OS 接口 130 通信的需要。

[0022] 此外,在一些实施例中,固件更新 155 可以服从由开发固件更新应用程序 145 的计算装置 100 的制造商或另一实体规定的协议,使得固件更新 155 可以将更新结果返回至固件更新应用程序 145。具体地,该协议可以规定固件更新 155 应当返回什么信息,应当如何传递该信息和对该信息编排格式等等。作为一个示例,在完成以后,固件更新 155 可以将表示更新成功还是失败、任何失败的原因以及类似信息的多个返回代码传递给固件更新应用程序 145。

[0023] 此外,在一些实施例中,硬件部件 120 的制造商可以在为具体的固件-OS 接口 130 设计的软件开发工具包(SDK)内开发固件更新 155。例如,当固件-OS 接口 130 是 UEFI 接口时,该制造商可以使用 UEFI SDK 来开发固件更新 155。这样的实施例是有优势的,因为它们通过允许使用专门的 UEFI 应用程序接口(API)、开发和调试工具等,而不再需要固件更新 155 的定制开发。

[0024] 应当注意,虽然存储介质 150 图示为位于计算装置 100 外部,但是其也可以位于计算装置 100 内部。因此,固件更新 155 可以存储在内部的硬盘驱动器、光盘或别的存储介质上。而且,虽然存储介质 140 和存储介质 150 图示为分离的装置,但是它们可以是位于计算装置 100 内部或外部的同一介质。作为示例,在一些实施例中,固件更新应用程序 145 和固件更新 155 都可以存储在外部存储介质(例如闪存驱动器或光盘)上,使得固件更新应用程序 145 和固件更新 155 可以从外部介质上启动。这样的实施例是有优势的,例如,将(例如由网络管理员)在多个计算装置 100 上安装相同的更新的环境下或者在固件更新应用程序 145 不能从计算装置 100 的内部存储介质启动的情况下。

[0025] 图 2 是包括固件更新应用程序 240 的计算装置 200 的实施例的框图,该固件更新应用程序 240 用于更新硬件部件 260 的固件。如图所示,计算装置 200 可以包括支持应用

程序 210、操作系统 220、固件 -OS 接口 230、固件更新应用程序 240、固件更新 250 和硬件部件 260。

[0026] 支持应用程序 210 可以包括为获得固件更新而配置成在 OS 220 上运行的应用程序。作为示例,具体的支持应用程序 210 可以使用例如每个硬件部件 260 的标识符周期地轮询远程服务器,以确定硬件部件 260 中一个或多个的固件更新的可获得性。当确定可获得更新时,支持应用程序 210 可以下载该更新到计算装置 200 能访问的存储介质上。可选地,支持应用程序 210 可以在用户的指示下下载和存储固件更新 250。作为另一示例,用户可以使用互联网浏览器或使用不同的计算装置获得固件更新。

[0027] 无论用来获得固件更新 250 的方法如何,固件更新 250 都可以存储在固件更新应用程序 240 能访问的位置上。作为一个示例,支持应用程序 210 可以将固件更新存储在硬驱或其它机器可读存储介质的专用分区上。在一些实施例中,该分区可以对于计算装置 200 的用户隐藏起来,使得用户不会通过操作系统 220 或其它方式访问固件更新 250。以此方式,支持应用程序 210 可以保证在固件更新 250 被执行以前用户不破坏或以其它方式更改固件更新 250。

[0028] 在一些实施例中,支持应用程序 210 还可以包括将结果报告给计算装置 200 的制造商、报告给特定硬件部件 260 的制造商或报告给某一其它实体的应用程序。具体地,当操作系统 220 在固件更新以后重新启动时,支持应用程序 210 可以通过 OS 220 访问这些结果。然后支持应用程序 210 可以通过网络连接(例如通过互联网)将这些结果发送给制造商或其它实体中的一个或多个。

[0029] 操作系统 220 可以充当支持应用程序 210 的宿主并且担当用户和硬件部件 260 之间的接口。在一些实施例中,当支持应用程序 210 存储固件更新时,OS 220 可以从支持应用程序那里接收这样存储的通知。然后 OS 220 可以设置在计算装置 200 重新启动时应当执行更新的指示并且请求重新启动计算装置。指示可以是例如布尔值、字符串、一系列数字或者任何其它足以表示当可获得固件更新 250 时的值。OS 220 可以将该指示存储在固件 -OS 接口 230 能访问的任何存储位置,只要固件 -OS 接口 230 在启动时检查该位置以获得该指示。作为一个示例,OS 220 可以使用视窗管理规范(WMI)将该指示写到固件 -OS 接口 230 能访问的非易失性存储介质(例如固件 -OS 接口 230 的 ROM)上。

[0030] 此外,在一些实施例中,OS 220 可以使支持应用程序 210 有权访问从固件更新应用程序 240 接收的结果。例如,OS 220 可以从预设的位置检索固件更新结果,然后将这些结果提供给支持应用程序 210。作为一个示例,OS 220 可以使用 WMI 从固件 -OS 接口 230 的 ROM 检索这些结果。作为另一示例,OS 220 可以从在指定位置存储的文本文件中检索这些结果。

[0031] 如上文所述,固件 -OS 接口 230 可以包括在存储介质上编码的用于提供硬件部件 260 的固件和 OS 220 之间的功能链接的可执行指令。在一些实施例中,固件 -OS 接口 230 可以包括被编码有检测指令 232 和发起指令 234 的机器可读存储介质。

[0032] 检测指令 232 可以在计算装置 200 启动期间判断一个或多个硬件部件 260 的固件是否可获得固件更新 250。作为示例,检测指令 232 可以访问 OS220 写入指示的位置,以判断是否可获得固件更新。该位置可以是例如存储介质(例如硬盘)上预设的位置或者 ROM 或其它可用来实现固件 -OS 接口 230 的存储介质中的位置。

[0033] 当检测指令 232 确定可获得固件更新时,发起指令 234 可以开始固件更新应用程序 240。然后,发起指令 234 可以访问存储固件更新应用程序 240 的存储介质并发出指令来开始该固件更新应用程序。例如,当固件更新应用程序 240 存储在专用分区上时,发起指令 234 可以知道该位置,因此从该专用分区中启动该固件更新应用程序。作为另一示例,发起指令 234 可以确定固件更新应用程序 240 位于外部驱动器(例如闪存)中,于是其从该外部驱动器启动该固件更新应用程序。然后固件更新过程的管理可以转交给固件更新应用程序 240。

[0034] 如上文所述,固件更新应用程序 240 可以包括在存储介质上编码的用于管理固件更新过程的可执行指令。在一些实施例中,固件更新应用程序 240 可以包括接收指令 242、定位指令 244、验证指令 246、执行指令 248 和备份指令 249。

[0035] 接收指令 242 可以从固件 -OS 接口 230 接收更新硬件部件 260 的固件的指示。例如,当启动固件更新应用程序 240 时,接收指令 242 可以接收由固件 -OS 接口 230 规定的应用参数。这样的参数可以规定例如可获得更新,标识硬件部件 260 以及指示固件更新 250 的存储位置。作为可选,接收指令 242 可以在不使用应用参数的情况下接收应当执行更新的指示。例如,固件 -OS 接口 230 启动固件更新应用程序 240 可以足以传达可获得更新的指示。

[0036] 定位指令 244 可以响应于更新固件的指示来定位固件更新 250。作为一个示例,定位指令 244 可以根据固件 -OS 接口 230 使用的应用参数确定固件更新 250 的位置。作为另一示例,当固件更新 250 存储在预设的位置,例如专用分区上时,定位指令 244 可以访问那个位置中的固件更新 250。作为又一示例,定位指令 244 可以对一个或多个存储介质执行查找来定位固件更新 250。

[0037] 验证指令 246 可以在发起执行固件更新 250 以前判断固件更新 250 是否有效。在一些实施例中,验证指令 246 可以使用与固件更新 250 对应的数字签名,该数字签名可以与固件更新 250 存储在相同位置。例如,在发布固件更新 250 以前,计算装置 250 或硬件部件 260 的制造商可以计算固件更新 250 的哈希值,然后使用专用密钥加密该哈希值。在执行固件更新 250 以前,可以有权访问相应公开密钥的固件更新应用程序 240 可以执行验证指令 246 来解密被加密的哈希值。然后,验证指令 246 可以将已解密的哈希值与新计算的固件更新 250 的哈希值相比较。当这些值匹配时,验证指令 246 可以确定批准执行固件更新 250。

[0038] 如上文所述,固件更新 250 的哈希值可以由计算装置 200 的制造商通过使用专用密钥加密。以此方式,计算装置 200 的制造商可以验证固件更新 250 的正常作用,计算固件更新 250 的数字签名,然后将固件更新 250 和该数字签名提供其顾客。这样的实施例是有优势的,因为制造商可能是唯一知道专用密钥的一方,从而保证固件更新应用程序 240 仅会安装由制造商批准的更新。因此这些实施例可以提高安全性并且防止安装恶意或错误固件更新。

[0039] 执行指令 248 可以开始执行固件更新 250,以更新特定硬件部件 260 的固件。具体地,执行指令 248 可以从由定位指令 244 确定的位置启动的固件更新 250 并且管理更新过程。例如,执行指令 248 可以估计完成百分比并将其显示给用户。执行指令 248 还可以接收由固件更新 250 返回的、表示更新是否成功以及如果没有成功失败原因的值。此外,当更新完成时,执行指令 248 可以将任何结果写到 OS 220 能访问的位置(例如固件 -OS 接口

230 的 ROM、预设的文本文件等等中的位置)。

[0040] 备份指令 249 可以实现备份程序,以保证万一固件更新失败计算装置 200 仍继续工作。具体地,在执行指令 248 启动固件更新 250 以前,备份指令 249 可以通过例如产生当前映像的副本从硬件部件 260 的固件中取回固件备份。可选地,如果例如固件的备份是与固件更新 250 一起下载的,那么可以在执行固件更新应用程序 240 以前获得固件的备份。在固件更新 250 将执行返回给固件更新应用程序 240 以后,备份指令 249 可以判断更新是否成功,如果未成功,则通过开始可执行的固件备份或者通过将固件备份复制到硬件部件 260 的固件中,触发备份的恢复。

[0041] 然而,应当注意,固件更新 250 可以包括备份指令 249。在这样的实施例中,固件更新 250 可以包括在开始更新程序以前产生固件映像的备份的指令。例如,当固件更新 250 初始化时,备份指令可以执行复制固件当前内容所需的存取。以此方式,硬件部件 260 的制造商可以在固件更新 250 中包括适当的存取指令,不需要计算装置 100 的制造商知道这些指令。如果需要,固件更新 250 还可以包括应用固件更新以后验证更新的指令以及恢复备份的指令。

[0042] 同样如上文详细描述,固件更新 250 可以由硬件部件 260 的制造商以可由固件更新应用程序 240 执行的文件的形式提供。硬件部件 260 可以是任何包括用来实施计算装置 200 的功能的机械元件、磁元件、电子元件和 / 或电气元件的装置。虽然未以独立的框示出,但用于指令执行的处理器可以包括在硬件部件 260 中。

[0043] 图 3 是用于对硬件部件的固件进行更新的方法 300 的示例的流程图。虽然方法 300 的执行在下文中是关于计算装置 100 的部件描述的,但是其它适合的用于方法 300 的执行的部件将对本领域的技术人员来说是显而易见的。方法 300 可以在机器可读存储介质上存储的可执行指令的形式实施。

[0044] 方法 300 可以从框 305 开始并继续进行到框 310,在这里固件更新 155 可以存储在计算装置 100 能访问的机器可读存储介质上。作为一个示例,处理器 110 可以执行由计算装置 100 的操作系统提供的指令,以将固件更新 155 存储在硬盘或其它内部或外部存储装置上。作为另一示例,用户可以手动复制固件更新 155 到机器可读介质上。

[0045] 然后方法可以继续到框 320,在这里当计算装置 100 启动时,固件 -OS 接口 130 可以发起固件更新应用程序 145。具体地,固件 -OS 接口 130 可以访问存储固件更新应用程序 145 的存储介质并由处理器 110 触发固件更新应用程序 145 的执行。

[0046] 在发起固件更新应用程序 145 以后,方法 300 可以继续到框 330,在这里固件更新应用程序 145 可以通过开始执行固件更新 155 触发硬件部件 120 的固件 125 的更新。具体地,处理器 110 可以执行固件更新 155 来访问和更改包含固件 125 的存储介质。在固件更新 155 完成以后,执行然后可以返回到固件更新应用程序 145。最后,方法 300 继续到框 335,在这里方法 300 结束。

[0047] 图 4A 和图 4B 是由操作系统执行的用于对硬件部件的固件进行更新的方法 400 的示例的流程图。虽然方法 400 的执行在下文中是关于计算装置 200 的部件描述的,但是其它适合的用于方法 400 的执行的部件将对本领域的技术人员来说是显而易见的。方法 400 可以在机器可读存储介质上存储的可执行指令的形式实施。

[0048] 方法 400 可以从框 405 开始并且继续进行到框 410,在这里支持应用程序 210 可以

检查在计算装置中包括的一个或多个硬件部件 260 的固件更新。如上文详细介绍的,支持应用程序 210 可以通过例如周期地查询远程服务器执行该检查。方法 400 然后可以继续到框 415,在这里支持应用程序 210 可以判断是否可获得固件更新。

[0049] 当确定不可获得计算装置 200 的硬件部件 260 的更新时,方法 400 可以返回框 410,在这里支持应用程序 210 可以在经过预定的时间段后检查固件更新。可选地,当可获得更新时,方法 400 可以继续到框 420。

[0050] 在框 420 中,支持应用程序 210 可以将固件更新 250 从远程服务器下载到计算装置 200 能访问的存储介质上。方法 400 然后可以继续到框 425,在这里支持应用程序 210 可以将固件更新 250 存储在固件更新应用程序 240 能访问的位置上。作为一个示例,支持应用程序 210 可以将固件更新存储在硬驱或其它机器可读存储介质的预定的目录中或专用分区中。

[0051] 在存储固件更新 250 以后,方法 400 可以继续到框 430,在这里操作系统 220 可以设置应执行更新的指示。应当注意,在一些实施例中,方法 400 的执行可以直接跳至框 430。例如,在用户已经手动将固件更新 250 存储在固件更新应用程序 240 能访问的位置中的情况下,可以跳过框 410、415、420 和 425 的执行。不管怎样,在框 430 中,OS 220 可以将应执行更新的指示存储在固件-OS 接口 230 知道的位置中。然后,方法 400 可以继续到下文对照图 4B 详细描述框 435。

[0052] 现在参考图 4B,在框 435 中,计算装置 200 可以在 OS 220 的要求下重启或者由用户手动重启。如下文结合图 5A 和图 5B 详细描述,然后固件-OS 接口 230、固件更新应用程序 240 和固件更新 250 可以尝试更新硬件部件 260 的固件。在尝试更新固件以后,计算装置 200 然后可以在框 440 中重启 OS220。

[0053] 在 OS 220 初始化以后,方法 400 可以继续到框 445,在这里支持应用程序 210 可以访问预定位置来判断是否可获得固件结果。例如,支持应用程序 210 可以通过 OS 220 访问提供固件更新结果的固件-OS 接口 230 的接口。作为另一示例,支持应用程序 210 可以从可在指定位置中访问的文本文件中读取更新结果。

[0054] 当在框 445 中确定不可获得固件更新结果时,方法 400 可以继续到框 455,在这里方法 400 结束。如果例如用户在启动期间指示不应启动固件更新应用程序 240,因而不会提供更新结果,那么上述场景是可能出现的。

[0055] 可选地,当在框 445 中确定可获得固件更新结果时,方法 400 可以继续到框 450,在这里支持应用程序 210 可以将更新结果报告给一个或多个制造商或其它实体。例如,支持应用程序 210 可以将更新结果报告给计算装置 200 的制造商和 / 或硬件部件 260 的制造商。这些结果可以包括例如更新是否成功,应用更新的时间以及应用更新所需时间量。如果更新失败,那么这些结果还可以包括失败详情(例如,无效签名、在过程期间的失败、用户中止该过程等等)。方法 400 然后可以继续到框 455,在这里方法 400 结束。

[0056] 图 5A 和图 5B 是在计算装置启动时执行的用于对硬件部件的固件进行更新的方法 500 的示例的流程图。虽然方法 500 的执行在下文中是关于计算装置 200 的部件描述的,但是其它适合的用于方法 500 的执行的部件将对本领域的技术人员来说是显而易见的。方法 500 可以在机器可读存储介质上存储的可执行指令的形式实施。

[0057] 方法 500 可以从框 505 开始并且继续进行到框 510,在这里可以启动计算装置

200。具体地,可以向计算装置 200 的硬件部件 260 供电,固件 -OS 接口 230 可以执行这些部件 260 的初始化和配置。然后方法 500 可以继续到框 515,在这里固件 -OS 接口 230 可以判断是否可获得用于安装的固件更新。作为示例,固件 -OS 接口 230 可以访问 OS 220 将可获得更新的指示写入的位置。该位置可以是例如存储介质上预定的位置或者 ROM 或其它用来实施固件 -OS 接口 230 的存储介质上的位置。可选地,计算装置 200 的用户可以通过在计算装置 200 启动时按压预定的键手动触发固件更新的安装。

[0058] 当在框 515 中确定不可获得固件更新时,方法 500 可以继续到框 540,在这里方法 540 结束并且固件 -OS 接口 230 继续其正常工作来加载 OS 220。可选地,当确定可获得固件更新时,方法 500 可以继续到框 520。在框 520 中,固件 -OS 接口 230 可以加载固件更新应用程序 240 来开始固件更新过程。应当注意,固件 -OS 接口 230 可以加载来自任何机器可读介质(无论是计算装置 200 内部的还是外部的)的固件更新应用程序 240。方法 500 然后可以继续到框 525。

[0059] 在框 525 中,固件更新应用程序 240 可以定位固件更新 250。如上文详细介绍的,固件更新应用程序 240 可以根据由固件 -OS 接口 230 使用的应用参数、通过访问预定位置或者通过执行查找确定固件更新 250 的位置。其它适合的用于定位固件更新 250 的方法将对本领域的技术人员来说是显而易见的。

[0060] 在定位固件更新 250 以后,方法 500 可以继续到框 530,在这里固件更新应用程序 240 可以尝试验证与固件更新 250 有关的数字签名。作为一个示例,同样如上文详细描述,固件更新应用程序 240 可以使用公开密钥来解密用相应的专用密钥加密的哈希值,然后将所解密的哈希值与固件更新 250 的计算的哈希值相比较。在该示例中,当这些哈希值匹配时,固件更新应用程序 240 可以确定固件更新 250 有效。

[0061] 当在框 530 中确定验证了该数字签名以及因此固件更新 250 有效时,那么方法 500 可以继续到下文结合图 5B 详细描述框 545。可选地,方法 500 可以继续到框 535,在这里固件更新应用程序 240 可以向 OS 220 能访问的位置写入失败结果,表示由于固件更新 250 的失败验证的原因,固件更新失败。然后方法 500 可以继续到框 540,在这里方法 540 结束,固件 -OS 接口 230 继续其正常工作来加载 OS 220。

[0062] 现在参考图 5B,在框 545 中,在执行更新 250 以前,固件更新应用程序 240 可以产生将要更新的硬件部件 260 的固件的备份。作为示例,固件更新应用程序 240 可以产生当前在硬件部件 260 的固件上存储的映像的副本。作为另一示例,可以在执行固件更新应用程序 240 以前获得该备份,使得固件更新应用程序 240 不需要产生备份。可选地,可以由固件更新 250 在开始运行以后执行硬件部件 260 的固件的备份。

[0063] 然后方法 500 可以继续到框 550,在这里固件更新应用程序 240 可以开始固件更新 250 的执行。固件更新 250 然后可以按需要更改硬件部件 260 的固件。以此方式,由于固件更新应用程序 240 启动固件更新 250,所以固件更新应用程序 240 可以管理更新过程。例如,固件更新应用程序 240 可以显示估计剩余时间并且从固件更新 250 那里接收表示更新结果的反馈。

[0064] 在固件更新 250 的执行已完成以后,方法 500 可以继续到框 555,在这里固件更新应用程序 240 可以判断更新是否成功。作为示例,固件更新应用程序 240 可以从固件更新 250 那里接收表示更新是否成功以及如果更新未成功那么一个或多个失败原因的一个或多

个返回代码。

[0065] 当在框 555 中确定更新未成功时,方法 500 可以继续到框 560,在这里固件更新应用程序 240 可以将失败详情写到 OS 220 能访问的位置上。然后,方法 500 可以继续到框 565,在这里固件更新应用程序 240 可以通过开始可执行的固件备份或者通过将备份复制至硬件部件 260 的固件,发起备份的恢复。然后,方法 500 可以继续到框 575。

[0066] 可选地,当在框 555 中确定更新已成功,方法 500 可以继续到框 570。在框 570 中,固件更新应用程序 240 可以将成功详情(例如日期和时间、更新持续时间、固件版本号等)写到 OS 220 能访问的位置上。然后方法 500 可以继续到框 575。

[0067] 在框 575 中,可以重新启动计算装置 200,使得可以应用对硬件部件 260 的固件的更改,如果有的话。然后方法 500 可以继续到框 580,在这里方法 500 结束。

[0068] 根据上述内容,本发明公开的示例实施例提供了由固件 -OS 接口发起的执行固件更新并管理固件更新过程的固件更新应用程序。以此方式,固件更新应用程序可以验证固件更新、向用户提供反馈以及向 OS 报告结果。此外,由于固件更新应用程序可以在固件 -OS 接口内执行,所以固件更新应用程序可以有权访问硬盘驱动器以及其它存储装置,使得其可以从多个位置运行更新以及执行文件管理操作。更新的开发也可以简化,因为在一些实施例中,部件制造商可以提供更改固件所需的可执行文件,同时计算装置制造商可以提供执行更新的环境。因此,示例实施例提供提高用户满意度的可靠用户友好的固件更新过程,同时降低由系统和部件制造商带来的成本。

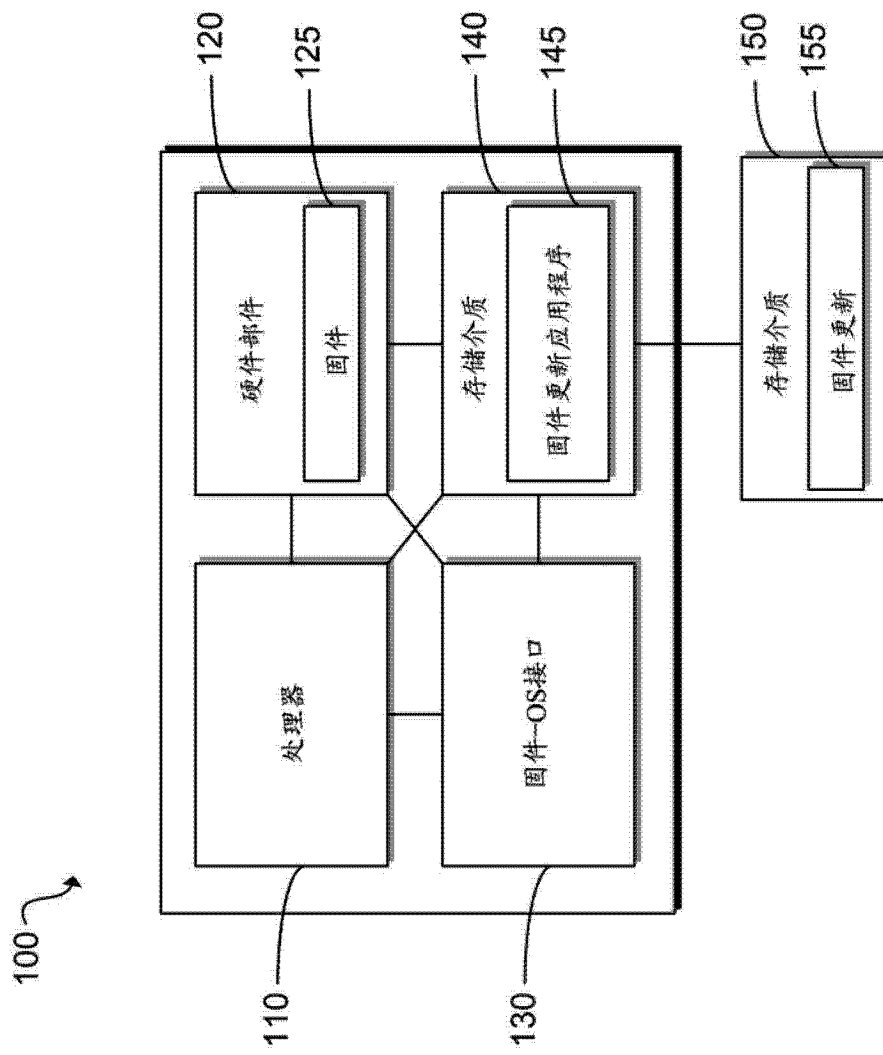


图 1

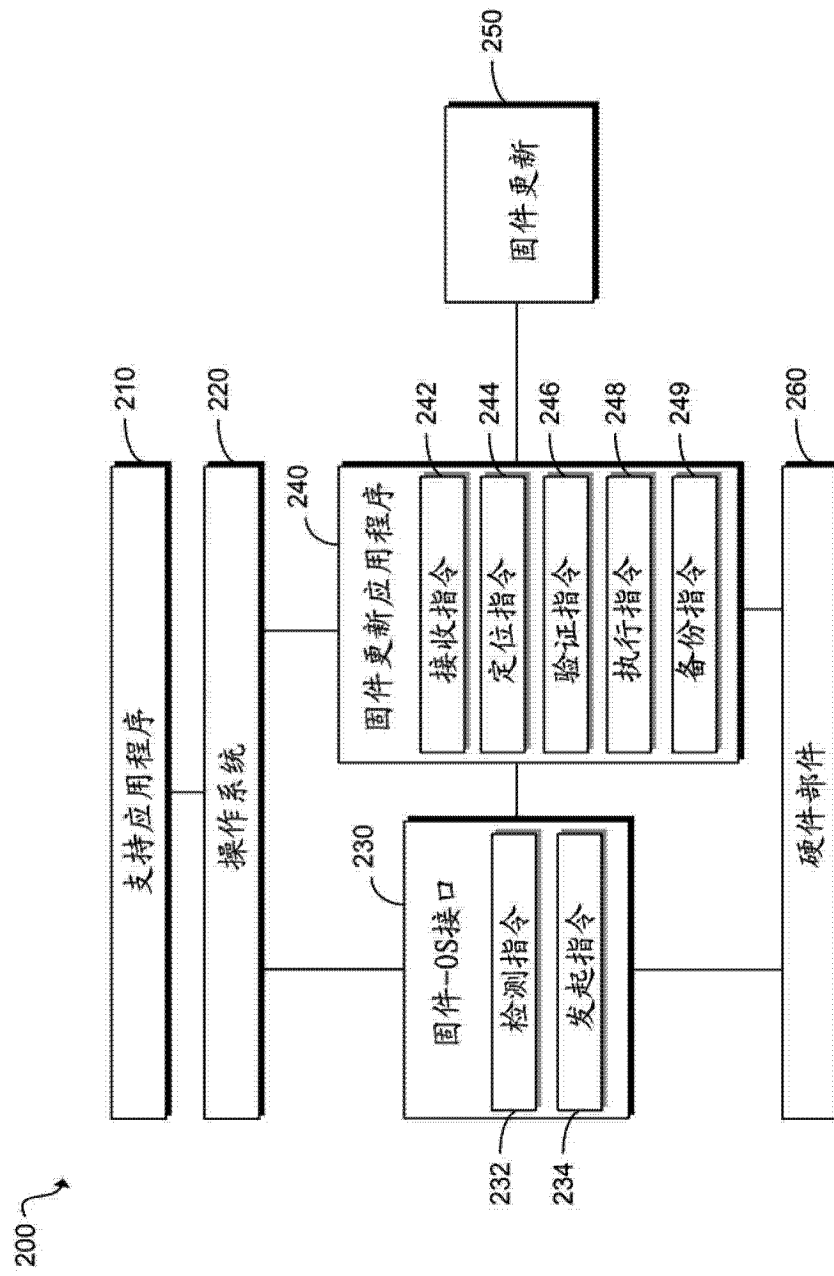


图 2

300

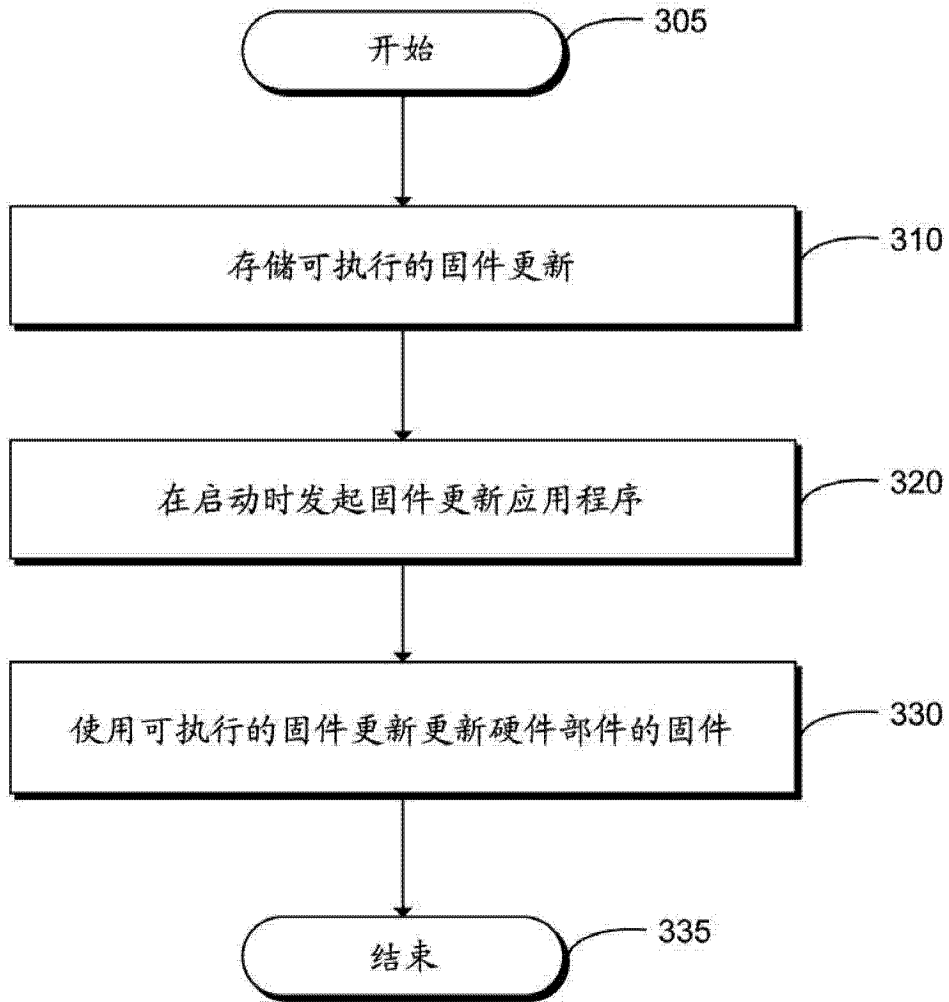


图 3

400

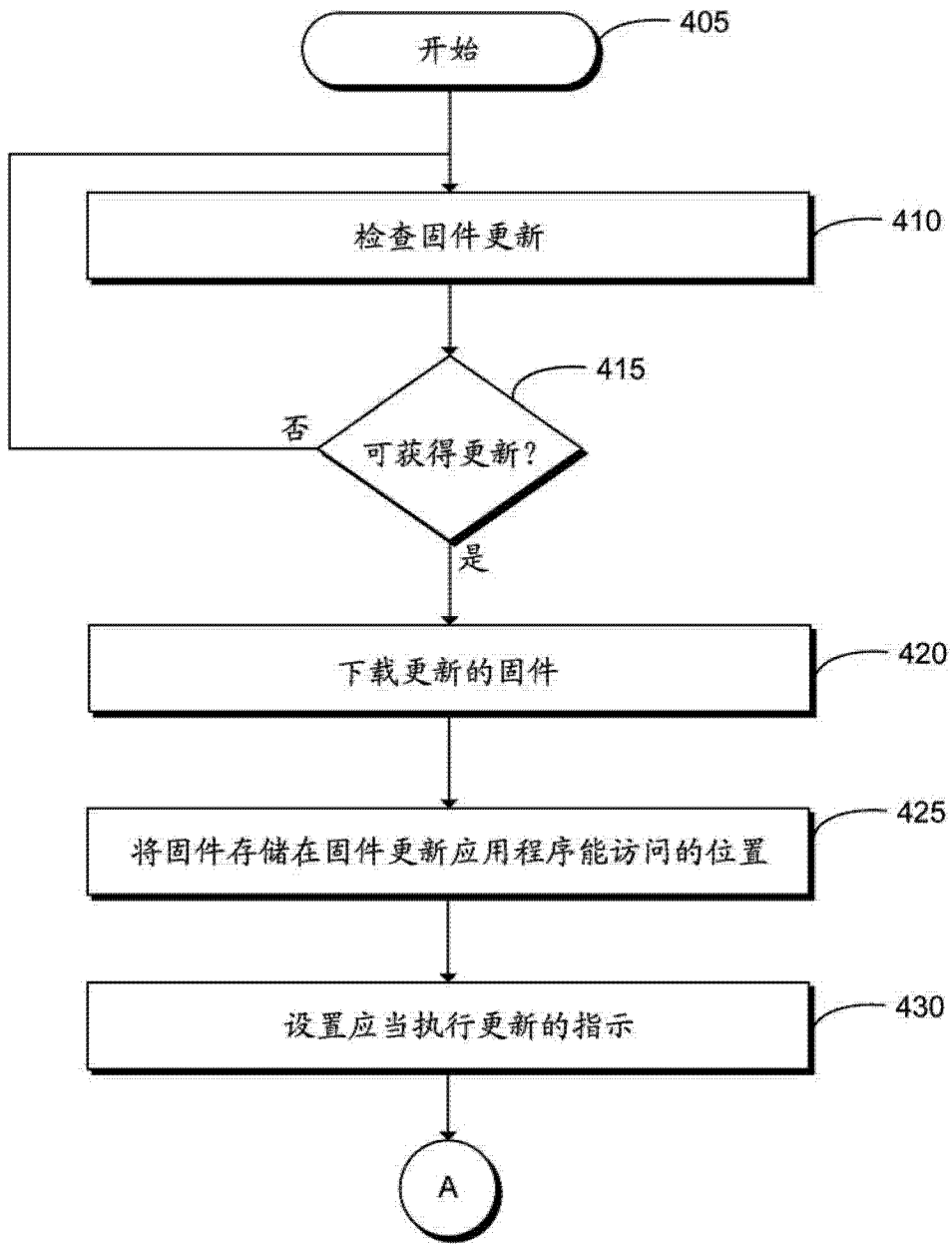


图 4A

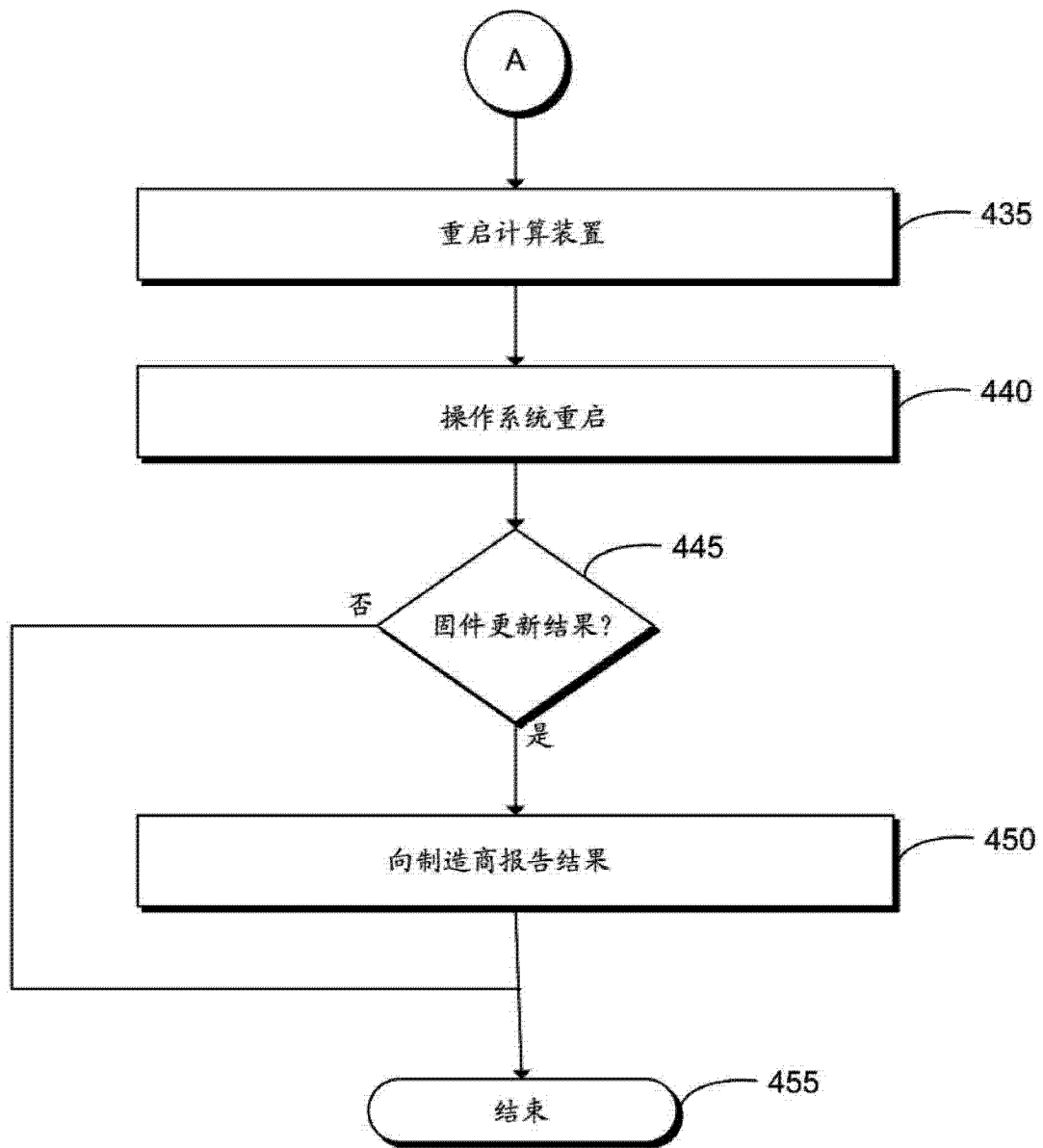


图 4B

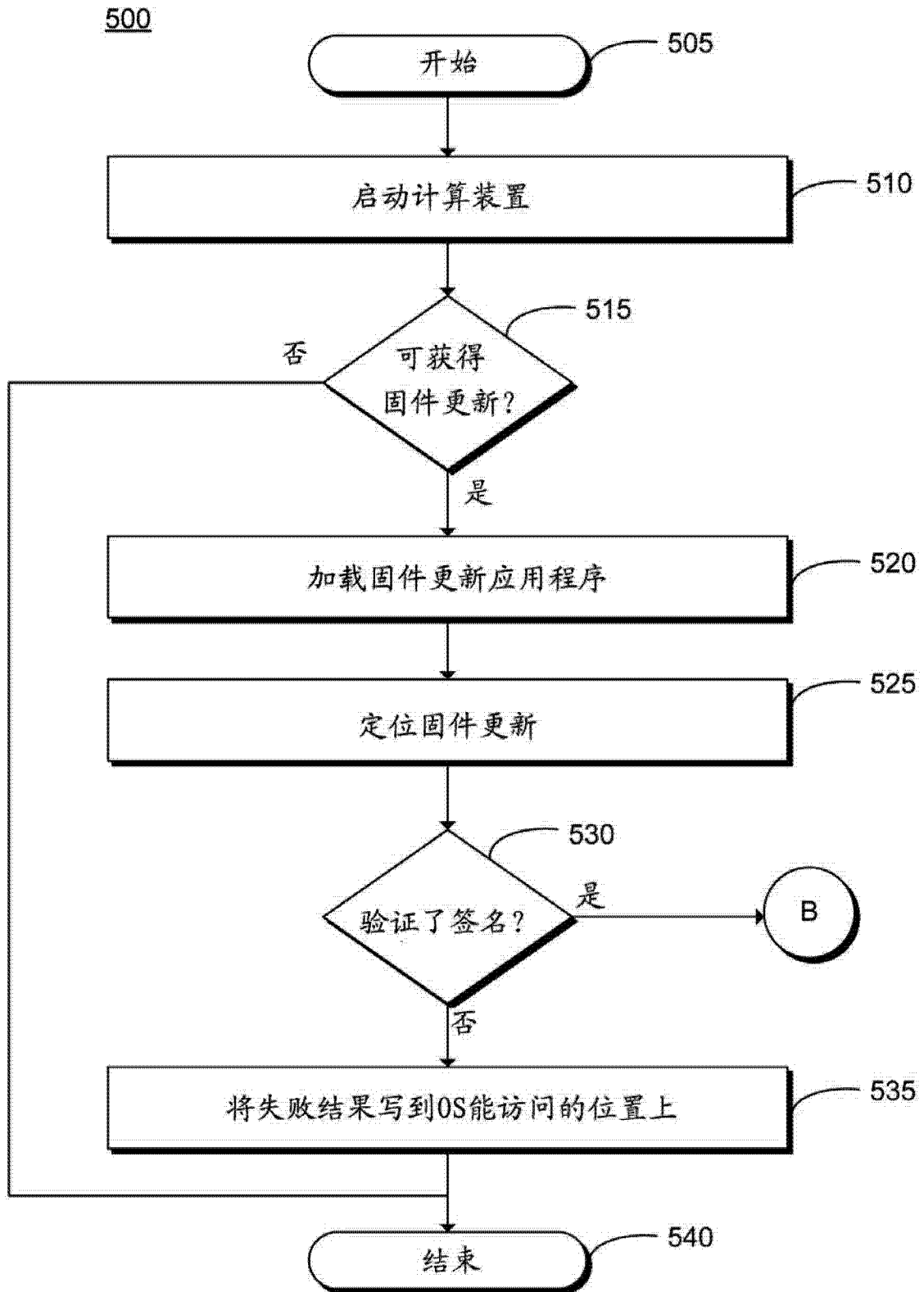


图 5A

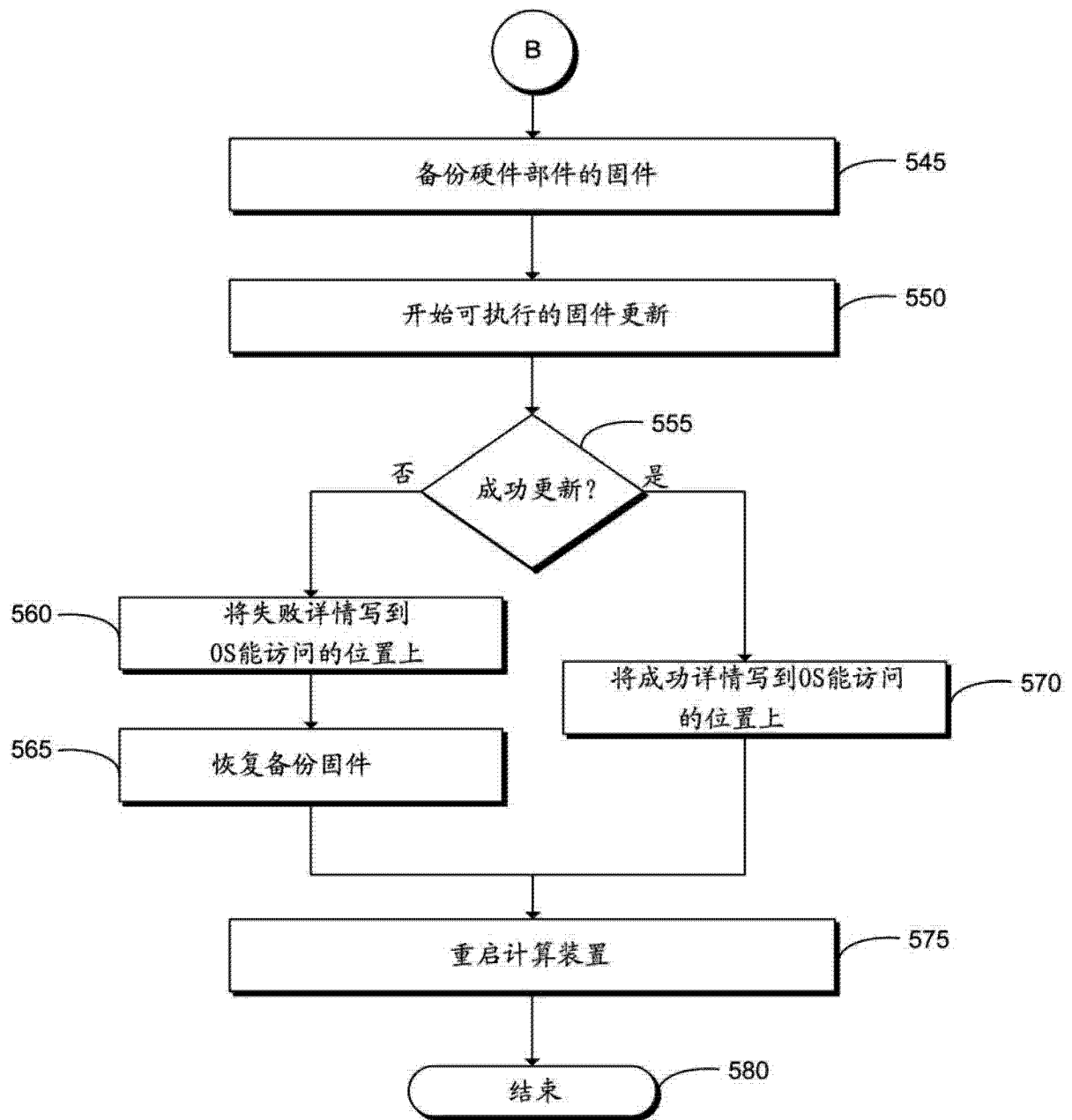


图 5B