(12) 按照专利合作条约所公布的国际申请

(19) 世界知识产权组织
国 际 局

(43) 国际公布日
2025 年 6 月 26 日 (26.06.2025)

WIPO | PCT

(10) 国际公布号
WO 2025/133751 A2

(51) 国际专利分类号:
*G06F 21/57* (2013.01) *G06F 9/445* (2018.01)

(21) 国际申请号: PCT/IB2024/061559

(22) 国际申请日: 2024 年 11 月 19 日 (19.11.2024)

(25) 申请语言: 中文

(26) 公布语言: 中文

(30) 优先权:
202311766515.8 2023年12月20日 (20.12.2023) CN

(71) 申请人: 云智能资产控股（新加坡）私人股份有限公司(**CLOUD INTELLIGENCE ASSETS HOLDING (SINGAPORE) PRIVATE LIMITED**) [SG/SG]; 新加坡新加坡勿拉士峇沙路51号来赞达一号大厦#03-06 189554 (SG)。

(72) 发明人:梅建强(**MEI, Jianqiang**); 中国浙江省杭州市西湖区三墩镇灯彩街1008号 310030 (CN)。 常琳(**CHANG, Lin**); 中国广东省深圳市南山区科苑南路3331号阿里中心 518052 (CN)。 黄明(**HUANG, Ming**); 中国广东省深圳市南山区科苑南路3331号阿里中心 518052 (CN)。 赵哲(**ZHAO, Zhe**); 中国浙江省杭州市西湖区三墩镇灯彩街1008号 310030 (CN)。 郭和益(**GUO, Heyi**); 中国浙江省杭州市西湖区三墩镇灯彩街1008号 310030 (CN)。

(81) 指定国(除另有指明，要求每一种可提供的国家保护): AE, AG, AL, AM, AO, AT, AU, AZ, BA, BB, BG, BH, BN, BR, BW, BY, BZ, CA, CH, CL, CN, CO, CR, CU, CV, CZ, DE, DJ, DK, DM, DO, DZ, EC, EE, EG, ES, FI, GB, GD, GE, GH, GM, GT, HN, HR, HU, ID, IL, IN, IQ, IR, IS, IT, JM, JO, JP, KE, KG, KH, KN, KP, KR, KW, KZ, LA, LC, LK, LR, LS, LU, LY, MA, MD, MG, MK, MN, MU, MW, MX, MY, MZ, NA, NG, NI, NO, NZ, OM, PA, PE, PG, PH, PL, PT, QA, RO, RS, RU, RW, SA, SC, SD,

(54) **Title:** TEE OS STARTING METHOD, DEVICE, AND STORAGE MEDIUM

(54) 发明名称: 一种TEE OS的启动方法、设备及存储介质

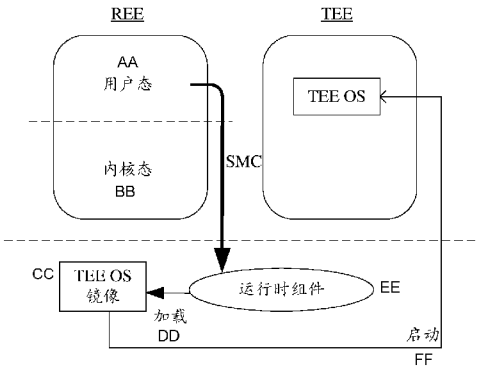


图 3

AA User state
BB Kernel state
CC TEE OS image
DD Load
EE Runtime component
FF Start

(57) **Abstract:** Embodiments of the present application provide a TEE OS starting method, a device, and a storage medium. After cold start is completed on the basis of BIOS firmware, in a user state of an REE, an SMC message used for instructing to perform TEE OS starting can be sent to a runtime component in trusted basic firmware; moreover, a compiled TEE OS image in the user state of the REE can be loaded to a specified memory address by the runtime component, and on the basis, under the instruction of the SMC message, by means of the runtime component, the image loaded to the specified memory address can be triggered to run in a TEE, so as to start a TEE OS. Accordingly, loading of the TEE OS image can be achieved by means of mutual cooperation of the user state of the REE and the runtime component in the trusted basic firmware, and the BIOS firmware does not need to be relied on any more, so that the maintainability of the TEE OS can be effectively improved.

(57) 摘要: 本申请实施例提供一种TEE OS的启动方法、设备及存储介质。在基于BIOS固件完成冷启动之后，可在REE的用户态下，发起用于指示进行TEE OS启动的SMC消息，至可信基础固件中的运行时组件；而且，可由运行时组件将在REE的用户态下为TEE OS编译完成的镜像加载至指定内存地址，基于此，可在SMC消息的指示下，借助运行时组件触发在TEE中运行已加载至指定内存地址中的所述镜像，从而启动TEE OS。据此，可通过REE的用户态与可信基础固件中的运行时组件相互配合实现TEE OS镜像的加载，而不再需要依赖BIOS固件，因此可有效改善TEE OS的可维护。

WO 2025/133751 A2

[见续页]

SE, SG, SK, SL, ST, SV, SY, TH, TJ, TM, TN, TR, TT, TZ, UA, UG, US, UZ, VC, VN, WS, ZA, ZM, ZW。

**(84)** 指定国(除另有指明，要求每一种可提供的地区保护): ARIPO (BW, CV, GH, GM, KE, LR, LS, MW, MZ, NA, RW, SC, SD, SL, ST, SZ, TZ, UG, ZM, ZW), 欧亚 (AM, AZ, BY, KG, KZ, RU, TJ, TM), 欧洲 (AL, AT, BE, BG, CH, CY, CZ, DE, DK, EE, ES, FI, FR, GB, GR, HR, HU, IE, IS, IT, LT, LU, LV, MC, ME, MK, MT, NL, NO, PL, PT, RO, RS, SE, SI, SK, SM, TR), OAPI (BF, BJ, CF, CG, CI, CM, GA, GN, GQ, GW, KM, ML, MR, NE, SN, TD, TG)。

本国际公布:

— 不包括国际检索报告，在收到该报告后将重新公布(细则48.2(g))。

— 黑白；提交的国际申请包含彩色或灰度，并且可通过PATENTSCOPE下载

# 一种 TEE OS 的启动方法、设备及存储介质

## 技术领域

本申请涉及可信计算技术领域，尤其涉及一种 TEE OS 的启动方法、设备及存储介质。

## 背景技术

TEE，全名为可信执行环境（Trusted Execution Environment），是指在计算平台上使用软硬件方法构建的一个安全隔离区域，进而保证在这个安全隔离的区域内拥有的数据和代码的生命周期在机密性和完整性方面得到保护。TEE OS（Trusted Execution Environment-Operating System）则是指运行在可信执行环境中的操作系统。

目前， TEE OS 启动过程中所需的 TEE OS 镜像需要打包至 BIOS（Basic Input Output System，基本输入输出系统）的固件镜像中，并随 BIOS 固件镜像一同预烧录到闪存 Flash 空间中，因此，TEE OS 的启动依赖于 BIOS 固件。而这种依赖于 BIOS 固件的 TEE OS 启动方式导致 TEE OS 的可维护性不佳，无法满足用户对 TEE OS 灵活多变的使用需求。

## 发明内容

本申请的多个方面提供一种 TEE OS 的启动方法、设备及存储介质，用以改善 TEE OS 的可维护性。

本申请实施例提供一种 TEE OS 的启动方法，包括：

在常规执行环境 REE 的用户态下，发起用于指示进行 TEE OS 启动的安全模式调用 SMC 消息，至可信基础固件中的运行时组件；

利用所述运行时组件，将在所述 REE 的用户态下为所述 TEE OS 编译完成的镜像加载至指定内存地址；

利用所述运行时组件，按照所述 SMC 消息触发在可信执行环境 TEE 中运行所述指定内存地址中的所述镜像，以启动所述 TEE OS。

本申请实施例还提供一种计算设备，包括存储器和处理器；

所述存储器被设置为存储一条或多条计算机指令；

所述处理器与所述存储器耦合，被设置为执行所述一条或多条计算机指令，以执行前述的 TEE OS 的启动方法。

本申请实施例还提供一种存储计算机指令的计算机可读存储介质，当所述计算机指令被一个或多个处理器执行时，致使所述一个或多个处理器执行前述的 TEE OS 的启动方法。

在本申请实施例中，在基于 BIOS 固件完成冷启动之后，可在常规执行环境 REE 的用户态下，发起用于指示进行 TEE OS 启动的安全模式调用 SMC 消息，至可信基础固件中的运行

时组件；而且，可由运行时组件将在REE的用户态下为TEE OS编译完成的镜像加载至指定内存地址，基于此，可在SMC消息的指示下，借助运行时组件触发在可信执行环境TEE中运行已加载至指定内存地址中的所述镜像，从而启动TEE OS。据此，本申请实施例中，可通过REE的用户态与可信基础固件中的运行时组件相互配合实现TEE OS镜像的加载，而不再需要依赖BIOS固件，因此，在无需重启BIOS固件的情况，即可实现按需在线启动TEE OS，这可有效改善TEE OS的可维护性。

## 附图说明

此处所说明的附图用来提供对本申请的进一步理解，构成本申请的一部分，本申请的示意性实施例及其说明用于解释本申请，并不构成对本申请的不当限定。在附图中：

图1为一种示例性的现有TEE OS的启动逻辑示意图；

图2为本申请一示例性实施例提供的一种TEE OS的启动方法的流程示意图；

图3为本申请一示例性实施例提供的一种TEE OS的启动方法的逻辑示意图；

图4为本申请一示例性实施例提供的另一种TEE OS的启动方法的流程示意图；

图5为本申请一示例性实施例提供的又一种TEE OS的启动方法的逻辑示意图；

图6为本申请一示例性实施例提供的一种与主核对应的示例性引导流程的示意图；

图7为本申请另一示例性实施例提供的一种计算设备的结构示意图。

## 具体实施方式

为使本申请的目的、技术方案和优点更加清楚，下面将结合本申请具体实施例及相应的附图对本申请技术方案进行清楚、完整地描述。显然，所描述的实施例仅是本申请一部分实施例，而不是全部的实施例。基于本申请中的实施例，本领域普通技术人员在没有做出创造性劳动前提下所获得的所有其他实施例，都属于本申请保护的范围。

在开始描述本申请各实施例提供的技术方案之前，先对本申请中涉及到的几个技术概念进行简单说明如下：

BIOS：Basic Input/Output System，是计算机微处理器上电后用来启动计算机系统的程序。它还管理计算机操作系统和附加设备(如硬盘、视频适配器、键盘、鼠标和打印机)之间的数据流。

TEE：Trusted Execution Environment，可信执行环境。是处理器中的一个安全区域，它有助于在保密性和完整性方面保护其中加载的代码和数据。

TEE OS（Trusted Execution Environment-Operating System）则是指运行在可信执行环境TEE中的操作系统。

REE：Rich Execution Environment，常规执行环境。包含常规操作系统及其与该执行环境其余部分的组合，不具备足够的安全性来满足设备所需的任务。

CA：Client Application，常规应用。通常运行在 REE 环境下的应用简称为 CA 。

TA：Trusted Application，可信应用。通常运行在 TEE 环境下的应用简称为 TA。

Hafnium：用于实现 Armv8.4-A S-EL2 扩展的一种安全分区管理器(SPM)，它允许多个隔离的安全分区(SP)在 S-EL1 层级上运行。

SMC：Secure Monitor Call，安全模式调用。用于进入安全模式并执行安全管控器（Secure Monitor，SM）内核服务调用，此指令只能在特权模式下执行。

ATF:Trusted Firmware-A,一种可信基础固件。是 Arm A-profile 架构(Armv8-A 和 Armv7-A)的安全世界软件的参考实现，包括 EL3 层级的安全管控器。

Runtime，ATF 中的运行时组件，用于在运行时状态下，支持 TEE 中的 TA 和 REE 中的 CA 之间进行交互。

FF-A：Firmware Framework-A profile，一种固件框架。

SPMD：FF-A 规范下，在 ATF 的运行时组件中部署的用于对安全分区管理器(SPM)进行调度的调度器。

正如背景技术中介绍的那样，目前 TEE OS 的启动需要依赖于 BIOS 固件，也即是，在基于 BIOS 固件进行冷启动的过程中，TEE OS 的启动是冷启动过程中的其中一个环节。图 1 为一种示例性的现有 TEE OS 的启动逻辑示意图。现有的启动方案中，CPU 中包含多个处理核 core 的情况下，通常会指定主核和从核。参考图 1，冷启动流程整体上可分为两个阶段：第一个阶段是主核上电启动，此时，其它从核并未上电，针对主核可按照安全启动链路（由图 1 中的 SCP ROM 及 BL2 及 BL31 等环节构成，为现有方案，在此不做过多说明）逐一加载各个组件，从而基于 BIOS 固件（图 1 中示为 UEFI）完成 REE 中操作系统内核 OS kernel 的启动，以完成主核的开核过程；之后可进入第二个阶段，主核完成 OS kernel 主要模块的初始化流程之后,OS kernel 可遵循从 BIOS 固件中获得的相关配置信息,调用 PSCI 接口 (Power State Coordination Interface，电源管理接口，是一种被设置为处理多核处理器的标准接口）进入可信基础固件 ATF 的运行时组件，为从核上电，随后进入从核的启动入口，并完成图 1 中的 Hafnium 及 TEE OS 的初始化等环节，以完成从核的开核过程。另外，在从核为多个的情况下，OS kernel 将重复执行上述关于从核的开核动作，进而完成其它从核的开核工作。

基于图 1 示出的冷启动流程可知，TEE OS 的启动穿插在冷启动的流程中，且正如背景技术中介绍的，TEE OS 的镜像需要烧录在 BIOS 的固件镜像中，因此，现有的 TEE OS 启动方案对 BIOS 固件有强依赖性。

显然，这会导致 TEE OS 的可维护性不佳，这里的可维护性不佳可理解为 TEE OS 的启动时机不灵活或启动后无法进行镜像再编译等，因此，现有的 TEE OS 启动方案无法满足用户对 TEE OS 灵活多变的使用需求。除此之外，由于 TEE OS 的镜像需要打包在 BIOS 固件镜像中，而 BIOS 固件镜像又被烧录在闪存 flash 空间中，因此，在处理器中需要部署多个 TEE OS 的情况下，BIOS 固件镜像的大小将随着 TEE OS 数量的增加而不断增加，这给 BIOS flash 空间带来比较严重的存储压力，甚至可能超过 flash 空间的存储上限。而且，TEE OS 的启动还将导致计算机冷启动的时延变长，由于对 BIOS 固件的依赖性较大，因此，在 TEE OS 数量不断增多的情况下，计算机的冷启动时延甚至可能达到小时级，导致用户体验不佳。

为此，本申请实施例提出了一种新的 TEE OS 启动方案，支持 TEE OS 按需在线启动，

而不再依赖于 BIOS 固件。

以下结合附图，详细说明本申请各实施例提供的技术方案。

图 2 为本申请一示例性实施例提供的一种 TEE OS 的启动方法的流程示意图。图 3 为本申请一示例性实施例提供的一种 TEE OS 的启动方法的逻辑示意图。参考图 2，该方法可包括：

步骤 100、在常规执行环境 REE 的用户态下，发起用于指示进行 TEE OS 启动的安全模式调用 SMC 消息，至可信基础固件中的运行时组件；

步骤 101、利用运行时组件，将在 REE 的用户态下为 TEE OS 编译完成的镜像加载至指定内存地址；

步骤 102、利用运行时组件，按照 SMC 消息触发在可信执行环境 TEE 中运行指定内存地址中的镜像，以启动 TEE OS。

本实施例提供的 TEE OS 的启动方法可应用于各种需要使用到 TEE OS 的场景中，以在场景中支持按需在线启动 TEE OS，本实施例对应用场景不做限定。

应当理解的是，本实施例提供的 TEE OS 的启动方法是在完成前文中提及的常规的冷启动流程之后实施的。其中，对于期望按照本实施例提供的 TEE OS 的启动方法进行启动的 TEE OS，不再需要将对应的镜像打包至 BIOS 固件镜像中，也不再需要将其对应的镜像烧录中 flash 中。也即是，在前文中提及的常规的冷启动流程中，不再需要对这些未烧录镜像的 TEE OS 进行启动，而是可在完成冷启动流程之后，按照本实施例提供的启动方法对这些未烧录镜像的 TEE OS 进行按需在线启动。因此，本实施例提供的 TEE OS 的启动方法可独立在常规的冷启动流程之后，不再依赖于 BIOS 固件，也不再需要占用冷启动流程中的 flash 空间及工作时延，这使得冷启动流程的效率更高且 flash 存储压力更小。

值得说明的是，本实施例并不禁止在冷启动流程中进行 TEE OS 的启动，也即是，部分期望开机即启动完成的 TEE OS 仍可按照传统的冷启动方式完成启动，本实施例对这部分 TEE OS 的冷启动过程不做干预。但是，针对这部分开机即启动完成的 TEE OS，可通过本实施例提供的 TEE OS 的启动方法来实现版本升级，换句话说，本实施例提供的 TEE OS 的启动方法可支持对已启动的 TEE OS 进行在线重启，进而支持 TEE OS 的在线升级。

参考图 3，本实施例中，计算机中提供了两种执行环境：TEE 和 REE。应当理解的是，这两种执行环境在前述的冷启动流程中已经构建完成。实际应用中，可采用 Trustzone 等技术来支持这两种执行环境的构建。以 TrustZone 为例，其可提供 Secure 空间和 Non-Secure 空间两种空间，分别对应于 TEE 和 REE 两种执行环境，并在两种执行环境之间进行硬件强制隔离，且可支持计算机的 CPU 进行工作模式的切换，以按需在两种执行环境中工作。示例性地，可通过切换安全配置 Secure Configuration Register 系统寄存器来使能工作模式的切换，该寄存器最后 1bit 为 0，表示当前 CPU 处于安全模式 secure mode。并且 Trustzone 技术还支持将系统资源配置成安全 secure 状态，通过操作相关寄存器可以将系统总线、内存、DMA、cache 等资源配置成安全 secure 状态，配置成安全 secure 状态之后，REE 中运行的程序 CA 无法访问这些硬件资源，从而实现两种执行环境之间的硬件强制隔离。针对两种执行环境的构建过程在此不作更详细的说明。

这样，本实施例中，在完成冷启动流程后，计算机中的 CPU 进入运行时（runtime），且在运行时支持将工作状态划分为两种：安全状态和非安全状态。当 CPU 处于安全状态时只能运行 TEE 侧的代码，且具有 REE 侧地址空间的访问权限。当 CPU 处于非安全状态时只能运行 REE 侧的代码，且只能通过事先定义好的接口来获取 TEE 侧中特定的数据和调用特定的功能。

另外，参考图 3，在构建出上述 TEE 和 REE 两种执行环境的基础上，本实施例中，还引入了可信基础固件来进一步保证 TEE OS 启动的可信性。示例性地，本实施例中的可信基础固件可采用 ATF（ARM Trusted Firmware）。基于 ATF 可在保有 Trustzone 提供的 Secure 空间和 Non-Secure 空间的同时，划分出 EL0（异常等级 Exception level 0）到 EL3 四个异常等级，以进一步保证启动和运行时阶段中的安全性。其中，前述的 CA 和 TA 可运行在 EL0，前述的 OS kernel 和本实施例中启动后的 TEE OS 则可运行在 EL1，BISO 则可运行在 EL2。

参考图 3，本实施例中提出，采用可信基础固件中的运行时组件来支持本实施例中的 TEE OS 的启动方法。其中，该运行时组件运行在前述的 EL3，且一直位于 Secure 空间中，这可有效保证本实施例中 TEE OS 的启动方法的安全性和可信性。

在此基础上，参考图 2，在步骤 100 中，可在常规执行环境 REE 的用户态下，发起用于指示进行 TEE OS 加载的安全模式调用 SMC 消息，至可信基础固件中的运行时组件。

本实施例中，可在 REE 的用户态中部署一自定义 CA（常规应用），并由该自定义 CA 实施需要在 REE 的用户态下进行的处理逻辑。当然，本实施例中还可采用其它类型的逻辑载体作为在 REE 的用户态下的方法执行主体，例如，除了前述的自定义 CA 之外，也可采用自定义脚本或者其它类型的自定义程序等，本实施例对在 REE 的用户态中部署的方法执行主体的形态不做限定。后文中在需要提及 REE 的用户态中方法执行主体的情况下，将以自定义 CA 作为示例。这里，在步骤 100 中，可由该自定义 CA 向可信基础固件中的运行时组件发送 SMC 消息。

其中，正如前文提供的 SMC 的定义，REE 的用户态中发起的 SMC 消息是一种用于安全模式调用的消息。这里可理解为，REE 的用户态通过发出 SMC 信息至可信基础固件中的运行时组件，将 CPU 的控制权从 REE 的用户态转移至可信基础固件中的运行时组件。

另外，本实施例中，REE 的用户态中发起的 SMC 消息是用于指示进行 TEE OS 启动。实际应用中，REE 的用户态中发起的 SMC 消息可用于指示进行单个 TEE OS 的启动，也可用于指示多个 TEE OS 一同启动，对于后一种情况下，可触发运行时组件自动重复执行针对单个 TEE OS 的启动逻辑。为便于理解，后文中，将从单个 TEE OS 的视角进行启动逻辑的描述。

继续参考图 2 和图 3，在步骤 101 中，可利用运行时组件，将在 REE 的用户态下为 TEE OS 编译完成的镜像加载至指定内存地址。这里存在两个方面的处理逻辑：

一方面，本实施例中，可在 REE 的用户态下为 TEE OS 编译镜像。实际应用中，可由前述的自定义 CA 完成镜像编译工作。基于此，本实施例中，可支持按照用户灵活多变的 TEE OS 使用需求，为 TEE OS 按需编译镜像，而且，可支持对 TEE OS 进行镜像再编译。也即是，本实施例中，可在 REE 的用户态下为单个 TEE OS 多次进行镜像编译，而在执行步骤 101 的

过程中，使用的是为 TEE OS 最新编译出的镜像，因此，本实施例中，可通过切换 TEE OS 启动时使用的镜像版本，来实现对 TEE OS 进行版本更新。实际应用中，当 REE 的用户态下对某个 TEE OS 的镜像进行重新编译后，可按照本实施例提供的 TEE OS 的启动方法，对该 TEE OS 进行重新启动，以实现版本升级。

本实施例中，在 REE 的用户态下为 TEE OS 编译完成的镜像，可存储至 REE 的用户态下的指定目录中。在此基础上，本实施例中，在 REE 的用户态下，前述的自定义 CA 还可调用 TEE 驱动提供的自定义接口，以利用该自定义接口从 REE 的用户态下指定目录获取为 TEE OS 编译完成的镜像，并将镜像存储至指定的缓冲区域中；之后，将缓冲区域的地址配置至第二寄存器中；其中，第二寄存器为 TEE 驱动与运行时组件之间约定的缓冲区域的地址传递媒介。实际应用中，可在 TEE 驱动中添加自定义接口函数 ffa_tos_load，在 REE 的用户态下，可调用该自定义接口函数，而该自定义接口函数的函数功能可定义为：调用 Linux 固件子系统接口函数 request_firmware，来获取 OS 文件系统指定目录下新编译得到的 TEE OS 镜像，并申请缓冲区域 buffer 暂存该镜像，最后，将 buffer 地址连同 Function ID 等写入第二寄存器。

另一方面，运行时组件可获取到在 REE 的用户态下为 TEE OS 编译完成的镜像，并将镜像加载至指定内存地址中。

其中，可在可信基础固件中为 TEE OS 镜像预先配置指定内存地址，这样，在步骤 101 中，运行时组件可按照可信基础固件中的指定，将 TEE OS 的镜像存储至指定内存地址。后续的处理逻辑中，也将继续遵循可信基础固件中的该指定，从而保证在 TEE 中能够准确地查找到镜像，进而运行镜像。

承接前文中的第二寄存器，实际应用中，运行时组件可从第二寄存器中查询到 TEE OS 镜像所处的缓冲区域的地址，从而可从该缓冲区域中读取到 TEE OS 镜像，进而可将读取到的 TEE OS 镜像转移至指定内存地址。

值得说明的是，前述的步骤 101 中的镜像加载环节，可在步骤 100 之前即预先完成，也即可在 TEE OS 进入启动流程之前预先完成镜像编译及加载；当然，该环节也可在步骤 100 之后再执行，也即是在步骤 100 中发起的 SMC 消息的触发下，再进行 TEE OS 镜像的加载。两种加载时机均可，本实施例对此不做限定，能够保证在步骤 102 之前已经正确完成 TEE OS 的镜像加载即可。

另外，本实施例中，运行时组件在读取到 REE 的用户态为 TEE OS 所编译的镜像之后，还可对读取到的镜像进行签名校验；在确定镜像通过签名校验的情况下，再执行将镜像加载至指定内存地址的操作。本实施例中，不限定签名校验的实现方式，在一种示例性的解决方案中，可将 TEE 的公钥存储在硬件可信根中，而在 REE 的用户态可通过 TEE 私钥对 TEE OS 镜像进行加密，这样，由于运行时组件可通过 TEE 公钥对读取到的 TEE OS 进行签名校验。由于 TEE 公钥仅可能在 TEE 中被获知，因此可保证 TEE 公钥的安全性，进而保证签名校验的可信性。本实施例中，在镜像加载之前增加对 TEE OS 镜像的签名校验环节，可有效保障 TEE OS 镜像的安全性和可信性，避免因 REE 的用户态参与 TEE OS 的启动过程而引入安全性问题。

可知，本实施例中，在REE的用户态和可信基础固件中运行时组件的相互配合下，可支持在REE的用户态下为TEE OS按需编译镜像，且将编译完成的镜像顺利加载至TEE OS启动时默认的镜像读取地址（也即前文的指定内存地址）中。整个镜像加载的过程完全无需依赖BIOS固件，完全脱离了前述的冷启动流程。

因此，本实施例中，在无需重启BIOS固件的情况下，即可实现按需在线完成TEE OS镜像的编译及加载。可为TEE OS的启动提供了可维护的启动基础。

继续参考图2和图3，基于加载好的TEE OS镜像，在步骤102中，可在运行时组件的引导下，在可信执行环境TEE中运行指定内存地址中的镜像，以启动所需的TEE OS。

由于TEE OS的镜像已经按约定存储到了指定内存地址，因此，运行时组件只需按约定进行启动流程的引导，即可进入到TEE OS镜像的运行入口，进而无障碍地开始在可信执行环境TEE中运行TEE OS的镜像，以完成TEE OS的启动。本实施例中，可在可信基础固件的代码中定义用于完成步骤102的引导流程，本实施例中不限定具体的引导流程，实际应用中，可按需进行引导流程的定义，本实施例中保证该引导流程能够触发在可信执行环境TEE中运行指定内存地址中的镜像即可，后文中将提供示例性的引导流程，在此不做展开说明。

综上，本实施例中，在基于BIOS固件完成冷启动之后，可在常规执行环境REE的用户态下，发起用于指示进行TEE OS启动的安全模式调用SMC消息，至可信基础固件中的运行时组件；而且，可由运行时组件将在REE的用户态下为TEE OS编译完成的镜像加载至指定内存地址，基于此，可在SMC消息的指示下，借助运行时组件触发在可信执行环境TEE中运行已加载至指定内存地址中的所述镜像，从而启动TEE OS。据此，本申请实施例中，可通过REE的用户态与可信基础固件中的运行时组件相互配合实现TEE OS镜像的加载，而不再需要依赖BIOS固件，因此，在无需重启BIOS固件的情况，即可实现按需在线启动TEE OS，这可有效改善TEE OS的可维护性。

除了改善TEE OS的可维护性之外，本实施例提供的按需在线启动TEE OS的方案，TEE OS的启动时延可达到秒级，相较于传统启动方案中小时级的启动时延，本实施例中可有效改善TEE OS的启动效率。而且，本实施例中，在线启动的TEE OS不再需要占用BIOS的flash空间，因此，可有效减轻flash空间的存储压力。

图4为本申请一示例性实施例提供的另一种TEE OS的启动方法的流程示意图。参考图4，该方法可包括：

步骤400、在常规执行环境REE的用户态下，发起用于指示进行TEE OS启动的安全模式调用SMC消息，至可信基础固件中的运行时组件；

步骤401、在REE的用户态下，发起用于传递资源需求的SMC消息，至运行时组件；

步骤402、利用运行时组件，为TEE OS编译与资源需求相适配的资源配置文件，以供运行时组件按照资源配置文件为TEE OS分配资源；

步骤403、利用运行时组件，将在REE的用户态下为TEE OS编译完成的镜像加载至指定内存地址；

步骤404、利用运行时组件，按照SMC消息触发在可信执行环境TEE中运行指定内存

地址中的镜像，以启动 TEE OS。

其中，步骤 400 和步骤 403-步骤 404 可参考前述实施例中的描述，本实施例中，基于步骤 401 和步骤 402 提供了一种进一步改进的 TEE OS 的启动方法。

发明人在研究过程中发现，传统冷启动流程中，TEE OS 的启动过程中，除了需要依赖 BIOS 固件进行 TEE OS 的镜像加载之外，为 TEE OS 进行资源分配时也需要依赖 BIOS 支持。这种依赖 BIOS 进行的资源分配方式，导致 TEE OS 的资源配置过于死板，无法支持资源弹性伸缩，进而也就无法满足用户针对 TEE OS 的灵活多变的资源占用需求。

为此，本实施例中提出了一种按需在线进行资源配置的技术构思，以进一步改善 TEE OS 的可维护性。

参考图 4，在步骤 401 中，可在 REE 的用户态下，发起用于传递资源需求的 SMC 消息，至运行时组件。其中，用于传递资源需求的 SMC 消息中可包含对资源配置参数，用于对所需占用的资源规格进行描述。资源配置参数可包括但不限于内存空间大小、物理核的个数或 IO 设备列表等，在此不做更多示例。这样，通过在 REE 的用户态下，发起用于传递资源需求的 SMC 消息，可将前述的资源配置参数传递至运行时组件。

继续参考图 4，在步骤 402 中，响应于 REE 的用户态中所发起的用于传递资源需求的 SMC 消息，运行时组件可为相应的 TEE OS 编译资源配置文件。而本实施例中提出，可有运行时组件负责基于资源配置文件为 TEE OS 进行资源配置，其中，正如前文提及的，运行时组件运行在 EL3 级别，因此，运行时组件具备对计算机中整个硬件资源的访问权限，且可用于进行硬件资源的调度。实际应用中，可由运行时组件中部署的 SPMD，来实施该资源配置过程，本实施例并不限于此。

这样，本实施例中，可在 REE 的用户态为 TEE OS 按需调整资源配置参数，并通过向运行时组件发起 SMC 消息的方式，来驱使运行时组件在线为 TEE OS 编译资源配置文件。因此，本实施例中，可支持 TEE OS 的资源弹性配置。在 TEE OS 所分配到的资源过多的情况下，可在 REE 的用户态为 TEE OS 调整资源配置参数，并将调整后的资源配置参数通过 SMC 消息传递至运行时组件，而运行时组件可为 TEE OS 更新对应的资源配置文件，在此基础上，可对 TEE OS 进行重新启动，即可使更新后的资源配置文件生效，从而为 TEE OS 分配更少的资源，以解决资源浪费问题。在 TEE OS 所分配到的资源不足的情况下，也可通过本实施例提供的按需在线进行资源配置的技术构思，来为 TEE OS 分配更多的资源，以提高 TEE OS 的性能。

由于本实施例中可支持对 TEE OS 的资源配置参数进行多次编译，为此，本实施例中，对于运行时组件来说，若已经存在与 TEE OS 对应的资源配置文件，则可按照本次接收到的用于传递资源需求的 SMC 消息，修改 TEE OS 对应的资源配置文件，从而实现资源配置文件的更新。

另外，值得强调的是，本实施例中，在 REE 的用户态支持为 TEE OS 进行资源配置参数及镜像的按需在线编译，发明人在研究过程中发现，TEE OS 的镜像中可包含了一部分与资源配置相关的配置内容，因此，本实施例中，若在 REE 的用户态为 TEE OS 进行了资源配置参

数的调整，则应同步调整 TEE OS 镜像中相关的配置内容。也即是，运行时组件中的资源配置文件与为 TEE OS 编译完成的镜像中所描述的资源需求保持一致。

综上，本实施例中，可支持在 REE 的用户态下为 TEE OS 在线编译资源配置参数，并可在将 TEE OS 的镜像加载至指定内存地址之前，将编译后的资源配置参数传递至运行时组件中，以供运行时组件为 TEE OS 编译资源配置文件，这样，运行时组件可按照资源配置文件为 TEE OS 动态分配资源，从而支持 TEE OS 的资源弹性伸缩，这可进一步改善 TEE OS 的可维护性，更好地满足用户对 TEE OS 所占用资源的灵活多变的规格要求。通过 TEE OS 的创建及删除等动作，可对系统资源进行按需申请和释放，提高系统资源利用率。

图 5 为本申请一示例性实施例提供的又一种 TEE OS 的启动方法的逻辑示意图。参考图 5，本实施例中仍遵照图 1 所示的 TEE OS 的启动流程，本实施例中为在 REE 的用户态下的处理构思和运行时组件中的处理构思提供了示例性的实现方式。

先来看 REE 的用户态。

本实施例中提出了一种发起用于指示进行 TEE OS 启动的安全模式调用 SMC 消息的示例性实现方式。

参考图 5，在该实现方式中，可在 REE 的用户态为 TEE OS 所需占用的处理核创建触发线程。其中，若 TEE OS 需占用多个处理核，则在 REE 的用户态下为多个处理核 core 分别创建触发线程。正如前文提及的，可在 REE 的用户态下为 TEE OS 编译资源配置参数，因此，REE 的用户态对 TEE OS 所需占用的处理核是已知的，基于此，可在 REE 的用户态下，为 TEE OS 所需占用的处理核分别创建触发线程。实际应用中，可调用 glibc 中的 pthread_create 及 pthread_setaffinity_np 等接口函数，以在 REE 的用户态中为处理核创建触发线程。

在该实现方式中，触发线程是用于发起 SMC 消息的，具体地，单个触发线程发起的 SMC 消息用于指示为 TEE OS 启动相应的处理核。值得说明的是，这里，启动处理核不再是冷启动流程中的开核动作，而是可理解为在相应的处理核为 TEE OS 进行相关的初始化工作。包括但不限于内存初始化、页表初始化、VCPU 初始化、镜像文件中相关代码的执行以及其它需在该处理核上为 TEE OS 运行做出的准备工作。值得强调的是，在该实现方式中，用于指示进行 TEE OS 启动的 SMC 消息并不会导致处理核的断电，也不会影响处理核上其它负载的运行。

在该实现方式中，在创建触发线程的过程中：可在 REE 的用户态下创建与 TEE OS 需占用的处理核数量一致的触发线程；并可为创建的各个触发线程分别配置处理核亲和度，以通过处理核亲和度形成触发线程与处理核之间的绑定关系。实际应用中，相互绑定的处理核和触发线程之间的处理核亲和度足够高，以支持按照处理核亲和度即可为触发线程唯一确定出去所绑定的处理核。当然，还可采用其它方式来实现处理核与触发线程之间的绑定关系，而并不限于此，在此不做更多示例。

这样，可在 REE 的用户态下，按照为 TEE OS 所需占用的多个处理核设定的启动顺序，依次运行多个处理核各自对应的触发线程，以针对多个处理核分别发起 SMC 消息。实际应用中，可通过添加互斥锁使得各个触发线程能够按照启动顺序运行。其中，发明人在研究过程

中发现，在 TEE OS 需要占用多个处理核的情况下，通常会设定主核和从核，且通常是先启动主核，再依次启动各个从核。为此，在该实现方式中，可在 REE 的用户态下，按照先主后从的顺序，设定多个处理核之间的启动顺序。这里再次强调的是，这里的启动并不是开核动作。

另外，实际应用中，在 REE 的用户态下，针对多个处理核分别发起的 SMC 消息可以是相同的，主要是用于驱动运行时组件进入后续处理逻辑。

再来看运行时组件。

参考图 5，在该实现方式中，在可信基础固件中针对主核和从核分别为运行时组件定义了不同的引导流程。运行时组件在接收到 REE 的用户态下所发起的 SMC 消息后，可先判断当前 SMC 消息对应的是主核还是从核，以便进入对应的引导流程。

为了支持运行时组件准确判断出接收到的 SMC 消息对应的是主核还是从核，在一种示例性的支持方案中：可在 REE 的用户态下，为多个处理核中主核和从核分别配置核标识；在第一寄存器中分别记录属于主核的核标识和属于从核的核标识。基于此，运行时组件可访问第一寄存器，若识别到目标 SMC 消息对应的处理核的核标识为属于主核的核标识，则确定目标 SMC 消息对应的处理核为主核。实际应用中，第一寄存器可采用 mpidr 寄存器，通过记录各个处理核之间的拓扑关系来表征处理核之间的主从关系。当然，这仅是示例性的，在此对第一寄存器不做限定。

以下将以运行时组件接收到目标 SMC 消息为例，对运行时组件中为主核和从核分别定义的引导流程进行详细说明。其中，目标 SMC 消息可以是针对 TEE OS 所需占用的多个处理核所分别发起的 SMC 消息中的任意一个。

参考图 5，对于运行时组件来说，可在接收到目标 SMC 消息后，判断目标 SMC 消息是否对应于主核；

若是，则可执行与主核对应的引导流程；

若否，则可执行与从核对应的引导流程。

图 6 为本申请一示例性实施例提供的一种与主核对应的示例性引导流程的示意图。参考图 6，继续以目标 SMC 消息为例，若运行时组件确定目标 SMC 消息对应于主核，则可在目标 SMC 消息的触发下，执行前述的将在 REE 的用户态下为 TEE OS 编译完成的镜像加载至指定内存地址的操作；在完成镜像加载之后，触发在可信执行环境 TEE 中运行与主核对应的初始化逻辑。也即是，可将图 2 中的步骤 101 放在主核对应的引导流程中进行执行。相应地，前文提及的针对 TEEOS 镜像进行的签名验签环节也可放在主核对应的引导流程中进行执行。参考图 6，按照主核的引导流程中，运行时组件可在完成镜像加载操作之后，触发在可信执行环境 TEE 中运行与主核对应的初始化逻辑。

应当理解的是，本实施例中，针对与主核对应的 SMC 消息，运行时组件执行完前述的镜像验签以及镜像加载操作之后，引导 TEE 中的相关组件开始运行与主核对应的初始化逻辑即可，运行时组件可只承担引导工作，具体的初始化逻辑可由 TEE 中的相关组件来实施。

在一种示例性的解决方案中：参考图 6，可在 TEE 中引入安全分区管理器，例如前文提

及的 Hafnium 等，用以管理多个 TEE OS。运行时组件可向 TEE 中的安全分区管理器发送第一触发通知，已将主核的控制权转移至安全分区管理器；安全分区管理器则可根据第一触发通知对主核执行针对 TEE OS 的初始化配置操作，且在完成初始化配置操作后调用预设指令以触发进入镜像中与主核对应的运行入口。这里，对主核执行针对 TEE OS 的初始化配置操作，可包括但不限于内存初始化、页表初始化及 VCPU 初始化等，在此不做穷举。另外，由于多处理核的原因，TEE OS 镜像中通常针对主核和从核提供不同的运行入口，例如，主核对应的运行入口通常为 reset_primary 函数，而从核的运行入口则通常为 reset_secondary 函数，当然，这仅是示例性的，本实施例对此不做限定。基于此，实际应用中，安全分区管理器可通过调用 start 函数（TEE OS 镜像的入口函数）来触发 TEE OS 的镜像运行，之后，start 函数又可继续调用 reset 函数，而 reset 函数又可继续调用 reset_primary 函数，从而得以运行 TEE OS 中与主核对应的镜像代码。其中，镜像的运行过程可位于前述的 EL1 中进行。

当然，以上在 TEE 中执行的初始化逻辑仅是示例性的部分逻辑，本实施例中，还支持定义其它初始化逻辑，例如，安全分区管理器在接收到第一触发通知后，可更新其管理的 TEE OS 的数量，在监测到 TEEOS 镜像中的相关镜像代码执行完成后，重新接管对主核的控制权，并在主核上为 TEE OS 启动所需的 VCPU 等，在此不做初始化逻辑的穷举。

至此可完成在主核上与 TEE OS 启动相关的工作。

还是以目标 SMC 消息为例，若目标 SMC 消息对应于从核，运行时组件按照目标 SMC 消息执行与从核对应的引导流程，以触发在可信执行环境 TEE 中运行与从核对应的初始化逻辑。

在与从核对应的引导流程中，运行时组件不再需要进行镜像的验签及加载操作，这一点区别于与主核对应的引导流程。其它的则可参考与主核对应的引导流程。同样，在 TEE 中存在多个 TEE OS 的情况下，运行时组件按照目标 SMC 消息，向 TEE 中用于管理 TEE OS 的安全分区管理器发送第二触发通知；安全分区管理器根据第二触发通知对从核执行针对 TEE OS 的初始化配置操作，且在完成初始化配置操作后调用预设指令以触发进入镜像中与从核对应的运行入口。

关于在 TEE 中运行与从核对应的初始化逻辑的过程，与前述在 TEE 中运行与从核对应的初始化逻辑的过程基本一致，为节省篇幅，在此不再重复赘述。

可知，本实施例中，REE 的用户态可与运行时组件相互配合，为 TEE OS 完成在主核和从核上的相关启动工作，整个过程中并无需依赖 BIOS 固件。

进一步优选地，参考图 5，本实施例中，运行时组件在监测到针对当前处理核发起的 SMC 消息已完成处理流程后，可向 REE 的用户态返回流程结束通知；REE 的用户态在接收到流程结束通知后，再针对下一个处理核发起 SMC 消息。实际应用中，运行时组件在监测到针对当前 SMC 消息的处理流程已经完成后，可恢复 EL3 层级上下文，再调用 eret 指令返回 REE 中的 OS kernel，最终 REE 中的自定义 CA 可通过调用 TEEC_FinalizeContext 接口函数，结束针对当前 SMC 消息的启动流程，并可针对下一个处理核发起 SMC 消息。如此往复循环，可为 TEE OS 在其所需占用的各个处理核上完成相关启动工作。

综上，本实施例中，可分别定义与主核和从核对应的引导流程，运行时组件可针对 TEE OS 所需占用的主核和从核分别执行相应的引导流程，以与 TEE 中的其它组件相互配合，在 TEE OS 所需占用的主核和从核上分别完成与 TEE OS 启动相关的工作，以完成 TEE OS 的启动。整个过程无需依赖 BIOS 固件，而且，相关组件各司其职，配合流畅，使得 TEE OS 的启动效率、安全性、准确性及可信性得以保障。

需要说明的是，在上述实施例及附图中的描述的一些流程中，包含了按照特定顺序出现的多个操作，但是应该清楚了解，这些操作可以不按照其在本文中出现的顺序来执行或并行执行，操作的序号如 801、802 等，仅仅是用于区分开各个不同的操作，序号本身不代表任何的执行顺序。另外，这些流程可以包括更多或更少的操作，并且这些操作可以按顺序执行或并行执行。需要说明的是，本文中的“第一”、“第二”等描述，是用于区分不同的触发通知或寄存器等，不代表先后顺序，也不限定“第一”和“第二”是不同的类型。

图 7 为本申请另一示例性实施例提供的一种计算设备的结构示意图。如图 7 所示，该计算设备包括：存储器 70 及处理器 71。

处理器 71，与存储器 70 耦合，被设置为执行存储器 70 中的计算机程序，以用于：

在常规执行环境 REE 的用户态下，发起用于指示进行 TEE OS 启动的安全模式调用 SMC 消息，至可信基础固件中的运行时组件；

利用所述运行时组件，将在所述 REE 的用户态下为所述 TEE OS 编译完成的镜像加载至指定内存地址；

利用所述运行时组件，按照所述 SMC 消息触发在可信执行环境 TEE 中运行所述指定内存地址中的所述镜像，以启动所述 TEE OS。

本实施例中，处理器中可包含多个处理核，可由其中的一个或多个处理核执行存储器 70 中的上述计算机程序。

在一可选实施例中，处理器 71 还可被设置为：

在将所述镜像加载至指定内存地址之前，在所述 REE 的用户态下，发起用于传递资源需求的 SMC 消息，至所述运行时组件；

利用所述运行时组件，为所述 TEE OS 编译与所述资源需求相适配的资源配置文件，以供所述运行时组件按照所述资源配置文件为所述 TEE OS 分配资源。

在一可选实施例中，处理器 71 在利用所述运行时组件，为所述 TEE OS 编译与所述资源需求相适配的资源配置文件时，具体可被设置为：

若已经存在与 TEE OS 对应的资源配置文件，则按照本次接收到的用于传递资源需求的 SMC 消息，修改所述资源配置文件；

其中，修改后的所述资源配置文件与为所述 TEE OS 编译完成的镜像中所描述的资源需求保持一致。

在一可选实施例中，处理器 71 在发起用于指示进行 TEE OS 启动的安全模式调用 SMC 消息时，具体可被设置为：

若所述 TEE OS 需占用多个处理核，则在所述 REE 的用户态下为所述多个处理核分别创

建触发线程；

按照为所述多个处理核设定的启动顺序，在所述 REE 的用户态下依次运行所述多个处理核各自对应的触发线程，以针对所述多个处理核分别发起所述 SMC 消息。

在一可选实施例中，所述多个处理核包括主核和从核，处理器 71 在利用所述运行时组件，按照所述 SMC 消息触发在可信执行环境 TEE 中运行所述指定内存地址中的所述镜像时，具体可被设置为：

所述运行时组件在接收到多条所述 SMC 消息中的目标 SMC 消息后，判断所述目标 SMC 消息是否对应于主核；

若所述目标 SMC 消息对应于主核，利用所述运行时组件按照所述目标 SMC 消息执行与主核对应的引导流程，以触发在可信执行环境 TEE 中运行与主核对应的初始化逻辑；

其中，所述目标 SMC 消息为所述多条 SMC 消息中的任意一条。

在一可选实施例中，处理器 71 在利用所述运行时组件按照所述目标 SMC 消息执行与主核对应的引导流程，以触发在可信执行环境 TEE 中运行与主核对应的初始化逻辑时，具体可被设置为：

在所述目标 SMC 消息的触发下，利用运行时组件执行所述将在所述 REE 的用户态下为所述 TEE OS 编译完成的镜像加载至指定内存地址的操作；

在完成镜像加载之后，利用运行时组件触发在可信执行环境 TEE 中运行与主核对应的初始化逻辑。

在一可选实施例中，所述处理器 71 在利用运行时组件触发在可信执行环境 TEE 中运行与主核对应的初始化逻辑时，具体可被设置为：

在所述 TEE 中存在多个 TEE OS 的情况下，利用所述运行时组件按照所述目标 SMC 消息，向 TEE 中用于管理 TEE OS 的安全分区管理器发送第一触发通知；

利用所述安全分区管理器根据所述第一触发通知对所述主核执行针对所述 TEE OS 的初始化配置操作，且在完成所述初始化配置操作后调用预设指令以触发进入所述镜像中与主核对应的运行入口。

在一可选实施例中，处理器 71 还可用于：

若所述目标 SMC 消息对应于从核，利用所述运行时组件按照所述目标 SMC 消息执行与从核对应的引导流程，以触发在可信执行环境 TEE 中运行与从核对应的初始化逻辑。

在一可选实施例中，处理器 71 在利用所述运行时组件按照所述目标 SMC 消息执行与从核对应的引导流程，以触发在可信执行环境 TEE 中运行与从核对应的初始化逻辑时，可用于：

在所述 TEE 中存在多个 TEE OS 的情况下，利用所述运行时组件按照所述目标 SMC 消息，向 TEE 中被设置为管理 TEE OS 的安全分区管理器发送第二触发通知；

利用所述安全分区管理器根据所述第二触发通知对所述从核执行针对所述 TEE OS 的初始化配置操作，且在完成所述初始化配置操作后调用预设指令以触发进入所述镜像中与从核对应的运行入口。

在一可选实施例中，处理器 71，还可被设置为：

在所述 REE 的用户态下，为所述多个处理核中主核和从核分别配置核标识；

在第一寄存器中分别记录属于主核的核标识和属于从核的核标识；

处理器 71 在利用运行时组件判断所述目标 SMC 消息对应的处理核是否为主核，包括：

利用所述运行时组件访问所述第一寄存器，若识别到所述目标 SMC 消息对应的处理核的核标识为属于主核的核标识，则确定所述目标 SMC 消息对应的处理核为主核。

在一可选实施例中，处理器 71 在所述 REE 的用户态下为所述多个处理核分别创建触发线程时，具体可被设置为：

在所述 REE 的用户态下创建与所述 TEE OS 需占用的处理核数量一致的触发线程；

为创建的各个触发线程分别配置处理核亲和度，以通过处理核亲和度形成触发线程与处理核之间的绑定关系。

在一可选实施例中，处理器 71 还可被设置为：

利用所述运行时组件在监测到针对当前处理核发起的 SMC 消息已完成处理流程后，向所述 REE 的用户态返回流程结束通知；

所述 REE 的用户态在接收到所述流程结束通知后，再针对下一个处理核发起 SMC 消息。

在一可选实施例中，处理器 71 还可被设置为：

利用所述运行时组件从指定的寄存器中读取所述 REE 的用户态为所述 TEE OS 所编译的镜像的存储地址；

对所述存储地址中的镜像进行签名校验；

在确定所述镜像通过签名校验的情况下，再执行将所述镜像加载至指定内存地址的操作。

在一可选实施例中，处理器 71 还可被设置为：

在所述 REE 的用户态下，调用 TEE 驱动提供的自定义接口，以利用所述自定义接口从所述 REE 的用户态下指定目录获取为所述 TEE OS 编译完成的所述镜像，并将所述镜像存储至指定的缓冲区域中；

将所述缓冲区域的地址配置至第二寄存器中；

其中，所述第二寄存器为所述 TEE 驱动与所述运行时组件之间约定的缓冲区域的地址传递媒介。

进一步，如图 7 所示，该计算设备还包括：通信组件 72、电源组件 73 等其它组件。图 7 中仅示意性给出部分组件，并不意味着计算设备只包括图 7 所示组件。

值得说明的是，上述关于计算设备各实施例中的技术细节，可参考前述的方法实施例中的相关描述，为节省篇幅，在此不再赘述，但这不应造成本申请保护范围的损失。

相应地，本申请实施例还提供一种存储有计算机程序的计算机可读存储介质，计算机程序被执行时能够实现上述方法实施例中执行的各步骤。

上述图 7 中的存储器，被设置为存储计算机程序，并可被配置为存储其它各种数据以支持在计算平台上的操作。这些数据的示例包括用于在计算平台上操作的任何应用程序或方法的指令，联系人数据，电话簿数据，消息，图片，视频等。存储器可以由任何类型的易失性或非易失性存储设备或者它们的组合实现，如静态随机存取存储器（SRAM），电可擦除可编

程只读存储器(EEPROM),可擦除可编程只读存储器(EPROM),可编程只读存储器(PROM),只读存储器(ROM),磁存储器,快闪存储器,磁盘或光盘。

上述图 7 中的通信组件,被配置为便于通信组件所在设备和其他设备之间有线或无线方式的通信。通信组件所在设备可以接入基于通信标准的无线网络,如 WiFi, 2G、3G、4G/LTE、5G 等移动通信网络,或它们的组合。在一个示例性实施例中,通信组件经由广播信道接收来自外部广播管理系统的广播信号或广播相关信息。在一个示例性实施例中,所述通信组件还包括近场通信(NFC)模块,以促进短程通信。例如,在 NFC 模块可基于射频识别(RFID)技术,红外数据协会(IrDA)技术,超宽带(UWB)技术,蓝牙(BT)技术和其他技术来实现。

上述图 7 中的电源组件,为电源组件所在设备的各种组件提供电力。电源组件可以包括电源管理系统,一个或多个电源,及其他与为电源组件所在设备生成、管理和分配电力相关联的组件。

本领域内的技术人员应明白,本申请的实施例可提供为方法、系统、或计算机程序产品。因此,本申请可采用完全硬件实施例、完全软件实施例、或结合软件和硬件方面的实施例的形式。而且,本申请可采用在一个或多个其中包含有计算机可用程序代码的计算机可用存储介质(包括但不限于磁盘存储器、CD-ROM、光学存储器等)上实施的计算机程序产品的形式。

本申请是参照根据本申请实施例的方法、设备(系统)、和计算机程序产品的流程图和/或方框图来描述的。应理解可由计算机程序指令实现流程图和/或方框图中的每一流程和/或方框、以及流程图和/或方框图中的流程和/或方框的结合。可提供这些计算机程序指令到通用计算机、专用计算机、嵌入式处理机或其他可编程数据处理设备的处理器以产生一个机器,使得通过计算机或其他可编程数据处理设备的处理器执行的指令产生用于实现在流程图一个流程或多个流程和/或方框图一个方框或多个方框中指定的功能的装置。

这些计算机程序指令也可存储在能引导计算机或其他可编程数据处理设备以特定方式工作的计算机可读存储器中,使得存储在该计算机可读存储器中的指令产生包括指令装置的制造品,该指令装置实现在流程图一个流程或多个流程和/或方框图一个方框或多个方框中指定的功能。

这些计算机程序指令也可装载到计算机或其他可编程数据处理设备上,使得在计算机或其他可编程设备上执行一系列操作步骤以产生计算机实现的处理,从而在计算机或其他可编程设备上执行的指令提供用于实现在流程图一个流程或多个流程和/或方框图一个方框或多个方框中指定的功能的步骤。

还需要说明的是,术语“包括”、“包含”或者其任何其他变体意在涵盖非排他性的包含,从而使得包括一系列要素的过程、方法、商品或者设备不仅包括那些要素,而且还包括没有明确列出的其他要素,或者是还包括为这种过程、方法、商品或者设备所固有的要素。在没有更多限制的情况下,由语句“包括一个……”限定的要素,并不排除在包括所述要素的过程、方法、商品或者设备中还存在另外的相同要素。

需要说明的是，本申请所涉及的用户信息（包括但不限于用户设备信息、用户个人信息等）和数据（包括但不限于用于分析的数据、存储的数据、展示的数据等），均为经用户授权或者经过各方充分授权的信息和数据，并且相关数据的收集、使用和处理需要遵守相关国家和地区的相关法律法规和标准，并提供有相应的操作入口，供用户选择授权或者拒绝。

以上所述仅为本申请的实施例而已，并不用于限制本申请。对于本领域技术人员来说，本申请可以有各种更改和变化。凡在本申请的精神和原理之内所作的任何修改、等同替换、改进等，均应包含在本申请的保护范围之内。

# 权利要求书

1、一种 TEE OS 的启动方法，，包括：

在常规执行环境 REE 的用户态下，发起用于指示进行 TEE OS 启动的安全模式调用 SMC 消息，至可信基础固件中的运行时组件；

利用所述运行时组件，将在所述 REE 的用户态下为所述 TEE OS 编译完成的镜像加载至指定内存地址；

利用所述运行时组件，按照所述 SMC 消息触发在可信执行环境 TEE 中运行所述指定内存地址中的所述镜像，以启动所述 TEE OS。

2、根据权利要求 1 所述的方法，其中，还包括：

在将所述镜像加载至指定内存地址之前，在所述 REE 的用户态下，发起用于传递资源需求的 SMC 消息，至所述运行时组件；

利用所述运行时组件，为所述 TEE OS 编译与所述资源需求相适配的资源配置文件，以供所述运行时组件按照所述资源配置文件为所述 TEE OS 分配资源。

3、根据权利要求 2 所述的方法，其中，利用所述运行时组件，为所述 TEE OS 编译与所述资源需求相适配的资源配置文件，包括：

若已经存在与 TEE OS 对应的资源配置文件，则按照本次接收到的用于传递资源需求的 SMC 消息，修改所述资源配置文件；

其中，修改后的所述资源配置文件与为所述 TEE OS 编译完成的镜像中所描述的资源需求保持一致。

4、根据权利要求 1 所述的方法，其中，发起用于指示进行 TEE OS 启动的安全模式调用 SMC 消息，包括：

若所述 TEE OS 需占用多个处理核，则在所述 REE 的用户态下为所述多个处理核分别创建触发线程；

按照为所述多个处理核设定的启动顺序，在所述 REE 的用户态下依次运行所述多个处理核各自对应的触发线程，以针对所述多个处理核分别发起所述 SMC 消息。

5、根据权利要求 4 所述的方法，其中，所述多个处理核包括主核和从核，利用所述运行时组件，按照所述 SMC 消息触发在可信执行环境 TEE 中运行所述指定内存地址中的所述镜像，包括：

所述运行时组件在接收到多条所述 SMC 消息中的目标 SMC 消息后，判断所述目标 SMC 消息是否对应于主核；

若所述目标 SMC 消息对应于主核，所述运行时组件按照所述目标 SMC 消息执行与主核对应的引导流程，以触发在可信执行环境 TEE 中运行与主核对应的初始化逻辑；

其中，所述目标 SMC 消息为所述多条 SMC 消息中的任意一条。

6、根据权利要求 5 所述的方法，其中，所述运行时组件按照所述目标 SMC 消息执行与

主核对应的引导流程，以触发在可信执行环境 TEE 中运行与主核对应的初始化逻辑，包括：

在所述目标 SMC 消息的触发下，执行所述将在所述 REE 的用户态下为所述 TEE OS 编译完成的镜像加载至指定内存地址的操作；

在完成镜像加载之后，触发在可信执行环境 TEE 中运行与主核对应的初始化逻辑。

7、根据权利要求 6 所述的方法，其中，所述运行时组件触发在可信执行环境 TEE 中运行与主核对应的初始化逻辑，包括：

在所述 TEE 中存在多个 TEE OS 的情况下，所述运行时组件按照所述目标 SMC 消息，向 TEE 中被设置为管理 TEE OS 的安全分区管理器发送第一触发通知；

所述安全分区管理器根据所述第一触发通知对所述主核执行针对所述 TEE OS 的初始化配置操作，且在完成所述初始化配置操作后调用预设指令以触发进入所述镜像中与主核对应的运行入口。

8、根据权利要求 5 所述的方法，其中，还包括：

若所述目标 SMC 消息对应于从核，所述运行时组件按照所述目标 SMC 消息执行与从核对应的引导流程，以触发在可信执行环境 TEE 中运行与从核对应的初始化逻辑。

9、根据权利要求 8 所述的方法，其中，所述运行时组件按照所述目标 SMC 消息执行与从核对应的引导流程，以触发在可信执行环境 TEE 中运行与从核对应的初始化逻辑，包括：

在所述 TEE 中存在多个 TEE OS 的情况下，所述运行时组件按照所述目标 SMC 消息，向 TEE 中被设置为管理 TEE OS 的安全分区管理器发送第二触发通知；

所述安全分区管理器根据所述第二触发通知对所述从核执行针对所述 TEE OS 的初始化配置操作，且在完成所述初始化配置操作后调用预设指令以触发进入所述镜像中与从核对应的运行入口。

10、根据权利要求 5 所述的方法，其中，

在所述 REE 的用户态下，为所述多个处理核中主核和从核分别配置核标识；

在第一寄存器中分别记录属于主核的核标识和属于从核的核标识；

判断所述目标 SMC 消息对应的处理核是否为主核，包括：

所述运行时组件访问所述第一寄存器，若识别到所述目标 SMC 消息对应的处理核的核标识为属于主核的核标识，则确定所述目标 SMC 消息对应的处理核为主核。

11、根据权利要求 4 所述的方法，其中，在所述 REE 的用户态下为所述多个处理核分别创建触发线程，包括：

在所述 REE 的用户态下创建与所述 TEE OS 需占用的处理核数量一致的触发线程；

为创建的各个触发线程分别配置处理核亲和度，以通过处理核亲和度形成触发线程与处理核之间的绑定关系。

12、根据权利要求 4 所述的方法，其中，还包括：

所述运行时组件在监测到针对当前处理核发起的 SMC 消息已完成处理流程后，向所述 REE 的用户态返回流程结束通知；

所述 REE 的用户态在接收到所述流程结束通知后，再针对下一个处理核发起 SMC 消息。

13、根据权利要求 1 所述的方法，其中，还包括：

利用所述运行时组件从指定的寄存器中读取所述 REE 的用户态为所述 TEE OS 所编译的镜像的存储地址；

对所述存储地址中的镜像进行签名校验；

在确定所述镜像通过签名校验的情况下，再执行将所述镜像加载至指定内存地址的操作。

14、根据权利要求 1 所述的方法，其中，还包括：

在所述 REE 的用户态下，调用 TEE 驱动提供的自定义接口，以利用所述自定义接口从所述 REE 的用户态下指定目录获取为所述 TEE OS 编译完成的所述镜像，并将所述镜像存储至指定的缓冲区域中；

将所述缓冲区域的地址配置至第二寄存器中；

其中，所述第二寄存器为所述 TEE 驱动与所述运行时组件之间约定的缓冲区域的地址传递媒介。

15、一种计算设备，，包括存储器和处理器；

所述存储器被设置为存储一条或多条计算机指令；

所述处理器与所述存储器耦合，被设置为执行所述一条或多条计算机指令，以执行权利要求 1-14 任一项所述的 TEE OS 的启动方法。

16、一种存储计算机指令的计算机可读存储介质，，当所述计算机指令被一个或多个处理器执行时，致使所述一个或多个处理器执行权利要求 1-14 任一项所述的 TEE OS 的启动方法。

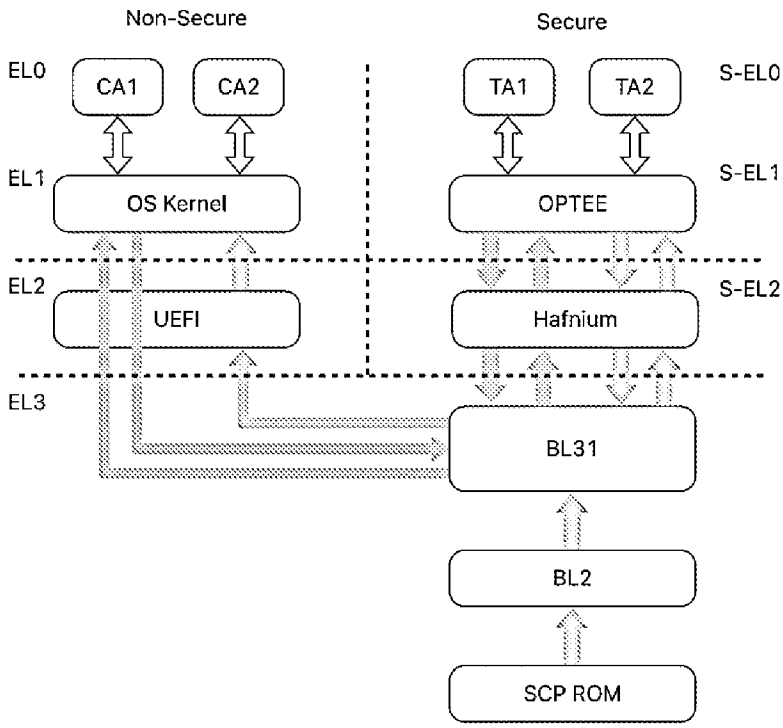
Non-Secure
Secure
EL0
CA1
CA2
TA1
TA2
S-EL0
EL1
OS Kernel
OPTEE
S-EL1
EL2
UEFI
Hafnium
S-EL2
EL3
BL31
BL2
SCP ROM

图 1

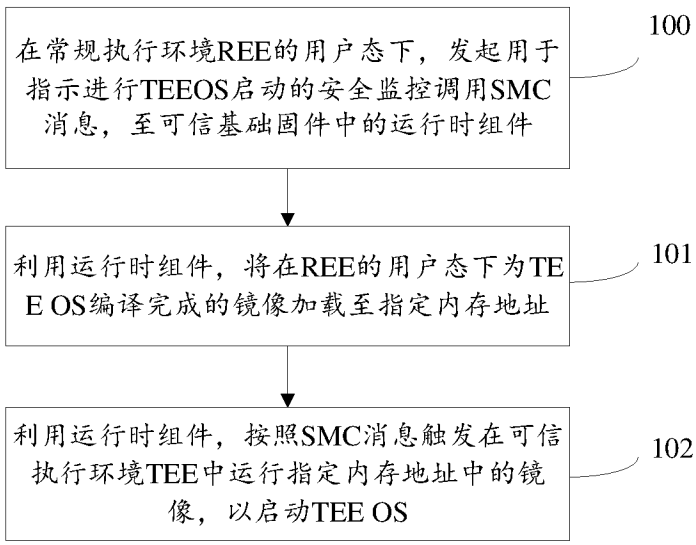
在常规执行环境REE的用户态下，发起用于指示进行TEEOS启动的安全监控调用SMC消息，至可信基础固件中的运行时组件
100
利用运行时组件，将在REE的用户态下为TEE OS编译完成的镜像加载至指定内存地址
101
利用运行时组件，按照SMC消息触发在可信执行环境TEE中运行指定内存地址中的镜像，以启动TEE OS
102

图 2

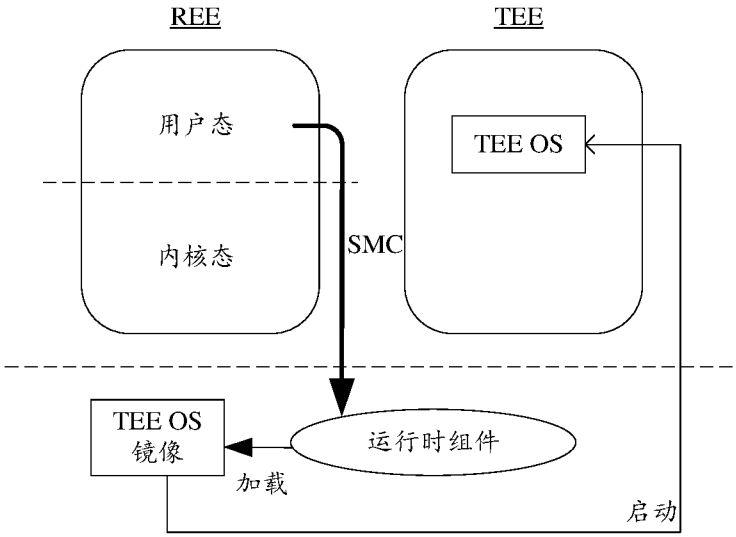
REE
TEE
用户态
内核态
SMC
TEE OS
TEE OS
镜像
运行时组件
加载
启动

图 3

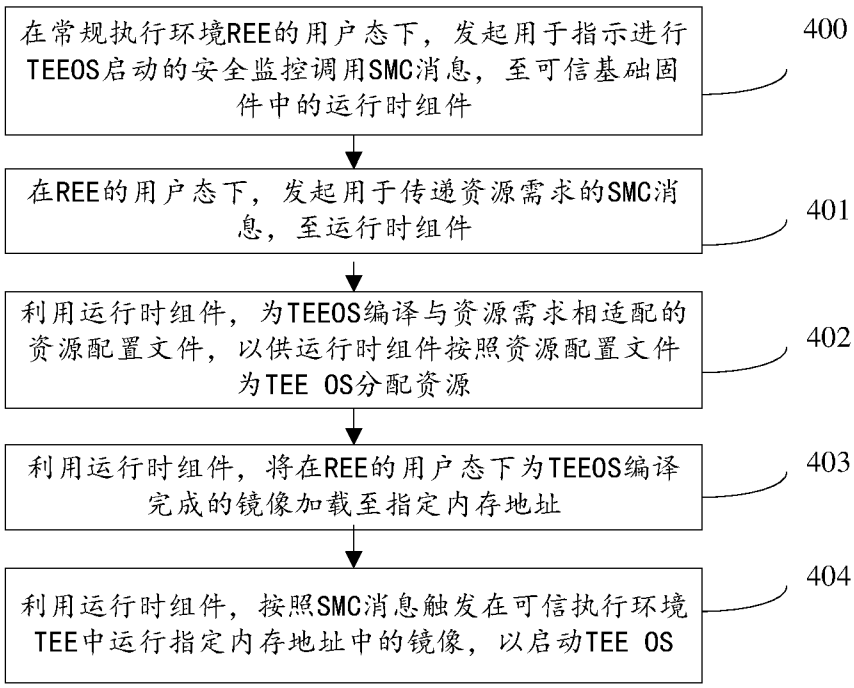
在常规执行环境REE的用户态下，发起用于指示进行TEEOS启动的安全监控调用SMC消息，至可信基础固件中的运行时组件
400
在REE的用户态下，发起用于传递资源需求的SMC消息，至运行时组件
401
利用运行时组件，为TEEOS编译与资源需求相适配的资源配置文件，以供运行时组件按照资源配置文件为TEE OS分配资源
402
利用运行时组件，将在REE的用户态下为TEEOS编译完成的镜像加载至指定内存地址
403
利用运行时组件，按照SMC消息触发在可信执行环境TEE中运行指定内存地址中的镜像，以启动TEE OS
404

图 4

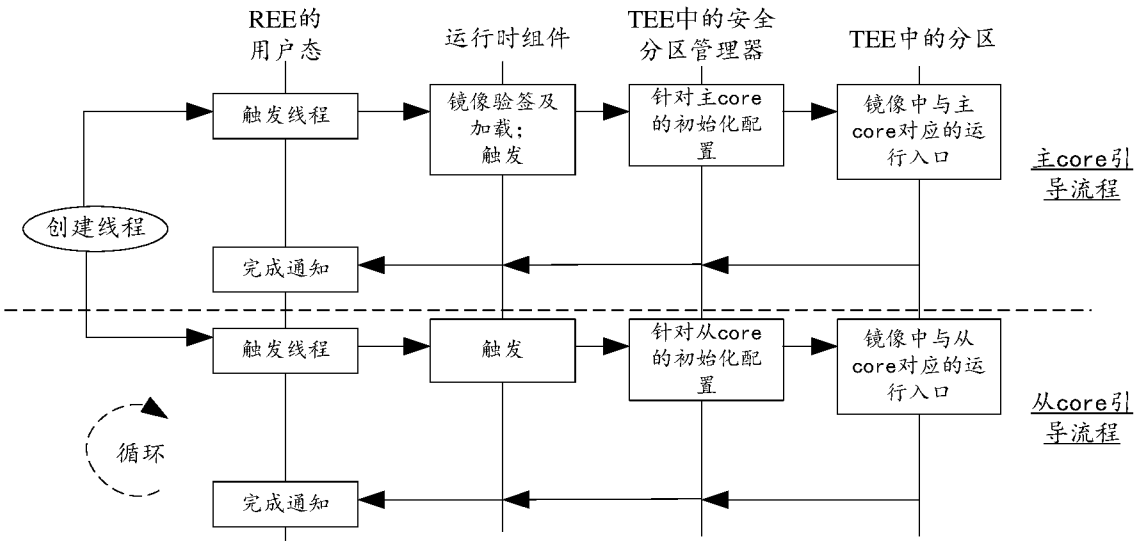
REE的
用户态
运行时组件
TEE中的安全
分区管理器
TEE中的分区
触发线程
镜像验签及
加载；
触发
针对主core
的初始化配
置
镜像中与主
core对应的运
行入口
主core引
导流程
创建线程
完成通知
触发线程
触发
针对从core
的初始化配
置
镜像中与从
core对应的运
行入口
从core引
导流程
循环
完成通知

图 5

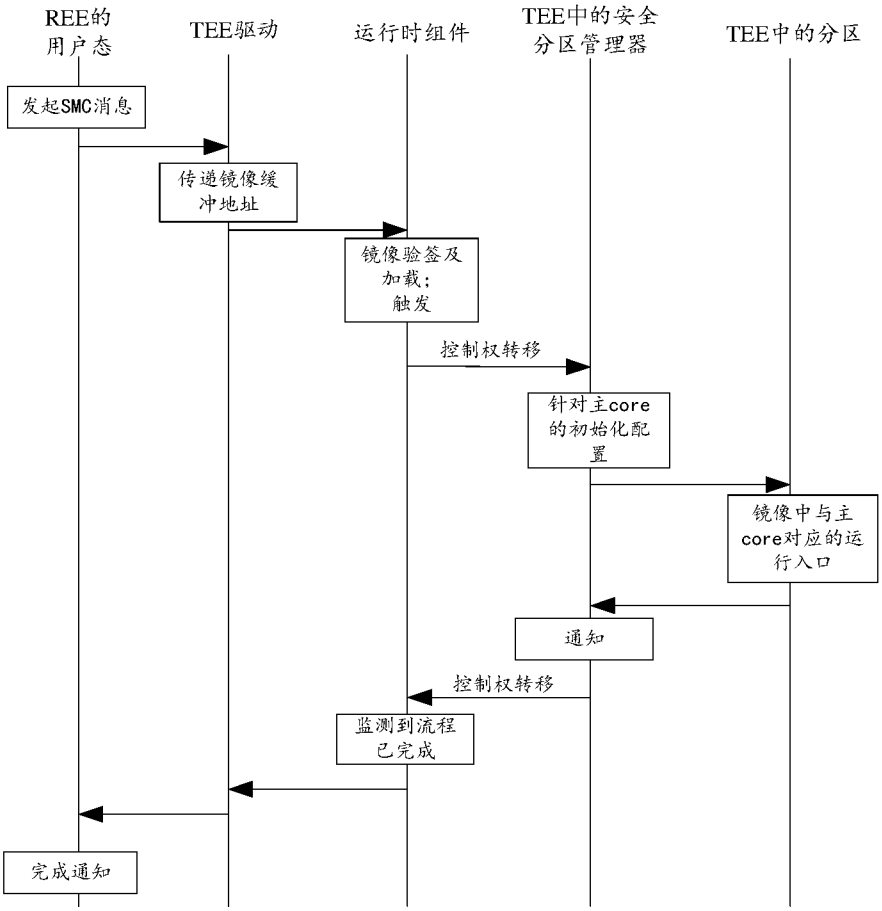
REE的
用户态
TEE驱动
运行时组件
TEE中的安全
分区管理器
TEE中的分区
发起SMC消息
传递镜像缓
冲地址
镜像验签及
加载；
触发
控制权转移
针对主core
的初始化配
置
镜像中与主
core对应的运
行入口
通知
控制权转移
监测到流程
已完成
完成通知

图 6

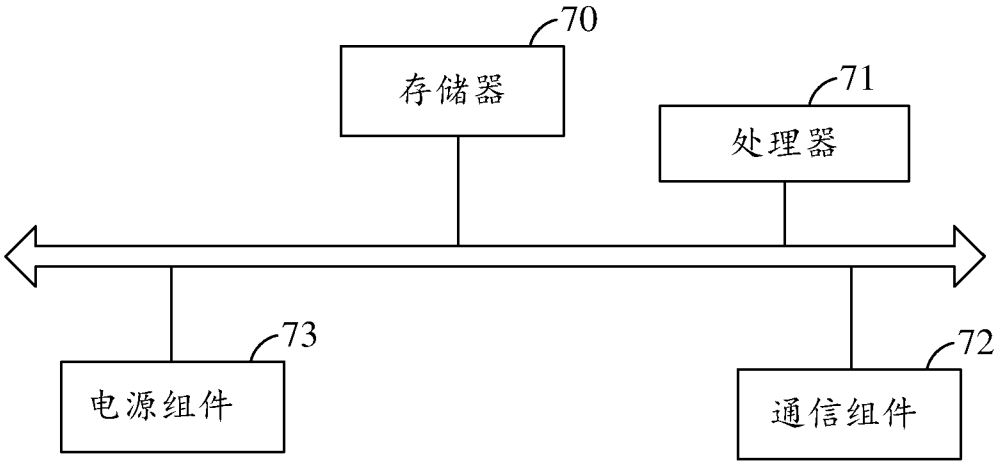


图 7