(19)国家知识产权局

(12)发明专利申请

(10)申请公布号 CN 117075974 A
(43)申请公布日 2023.11.17

(21)申请号 202310948530.8

(22)申请日 2023.07.31

(71)申请人 山东大学
地址 250100 山东省济南市历城区山大南路27号

(72)发明人 戴鸿君 张真瑜 李冰 翟明杰

(74)专利代理机构 济南金迪知识产权代理有限公司 37219
专利代理师 杨树云

(51)Int.Cl.
G06F 9/4401(2018.01)
G06F 9/445(2018.01)

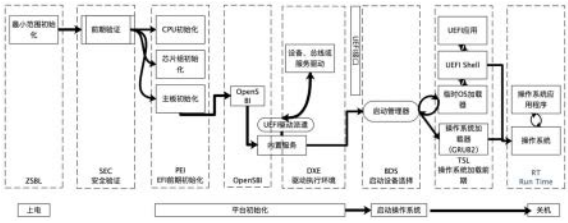
权利要求书2页 说明书8页 附图1页

(54)发明名称

一种基于RISC-V服务器CPU的新型EDK2启动引导方法

(57)摘要

本发明涉及一种基于RISC-V服务器CPU的新型EDK2启动引导方法,包括:在服务器CPU上电后,进行最小范围初始化,并将DTB和EDK2加载到内存中;EDK2完成安全性验证和efi前期初始化等工作;与硬件进行交互,完成底层初始化和设备初始化;对设备程序进行加载并执行,同时初始化硬件抽象层并运行DXE服务;第五步,GRUB加载真正的Linux内核映像和设备树,顺利启动Linux内核。本发明减少了SEC阶段OpenSBI库的冗余,使得OpenSBI在EDK2中具有灵活性和可拓展性,简化了调试和更新。在OpenSBI安全性验证和模式切换中增加了多样的可能性,实现了更可靠和灵活的Linux内核启动。



CN 117075974 A

1.一种基于RISC-V服务器CPU的新型EDK2启动引导方法,其特征在于,包括:

在RISC-V服务器CPU上电后,ZSBL阶段进行最小范围初始化,并将DTB和EDK2加载到内存中;

在SEC阶段和PEI阶段,EDK2完成安全性验证和EDK2前期初始化工作,并跳转到OpenSBI;

OpenSBI阶段,与硬件进行交互,完成底层初始化和设备初始化,并跳转回EDK2的DXE阶段;

DXE阶段,RISC-V服务器处于S-mode,对设备程序进行加载并执行,同时初始化硬件抽象层并运行DXE服务,同时,构建DXE阶段数据结构为BDS阶段提供服务信息;

GRUB加载真正的Linux内核映像和设备树,并设置必要的硬件配置和参数,必要的硬件配置包括硬件设备的检测、初始化和配置,参数包括启动命令行参数、内核参数、启动设备信息,顺利启动Linux内核。

2.根据权利要求1所述的一种基于RISC-V服务器CPU的新型EDK2启动引导方法,其特征在于,ZSBL阶段进行最小范围初始化,此时RISC-V服务器处于M-mode,包括:ZSBL阶段,初始化处理器的基本环境,包括设置寄存器、中断向量表。

3.根据权利要求1所述的一种基于RISC-V服务器CPU的新型EDK2启动引导方法,其特征在于,在RISC-V服务器CPU上电后,ZSBL阶段进行最小范围初始化,并将DTB和EDK2加载到内存中,之后还包括:

ZSBL阶段,进行内存DDR的初始化工作,内存DDR的初始化工作包括内存控制器初始化,内存训练,初始化DRAM以及内存映射,将EDK2的程序代码加到内存的指定位置,并通过跳转指令将控制权传递给EDK2。

4.根据权利要求1所述的一种基于RISC-V服务器CPU的新型EDK2启动引导方法,其特征在于,SEC阶段时,RISC-V服务器处于M-mode,在与OpenSBI分离后进行安全性验证:在安全环境中,SEC阶段进行一系列安全性验证,包括验证引导程序和固件的签名以及检查系统启动配置的完整性,寻址并跳转至PEI Core;

进一步优选的,PEI阶段时,RISC-V服务器处于M-mode,进行EDK2前期初始化工作:运行PEIM。

5.根据权利要求1所述的一种基于RISC-V服务器CPU的新型EDK2启动引导方法,其特征在于,在SEC阶段和PEI阶段,EDK2完成安全性验证和EDK2前期初始化工作,之后还包括:OpenSBI阶段时,RISC-V服务器处于M-mode,在与硬件进行交互时,完成处理器、内存和设备关键组件的初始化工作,设置处理器状态、初始化内存子系统,并与外部设备建立通信连接;

进一步优选的,底层初始化,包括:判断hart id,代码重定位、清除除保存设备树地址的寄存器值、清除BSS段、设置栈指针、读取设备树中的设备信息、设备树重定位;

通过执行sbi_init来进行设备初始化,包括:控制台的初始化、irq中断和核间中断的初始化、MMU的TLB表的初始化、timer的初始化;

利用sbi_hsm_prepare_next_jump函数跳转到UEFI固件的DXE阶段入口函数。

6.根据权利要求1所述的一种基于RISC-V服务器CPU的新型EDK2启动引导方法,其特征在于,RISC-V服务器处于S-mode下,EDK2继续执行还包括:

EDK2继续执行DXE阶段将设备程序进行加载并执行,BDS阶段进行DXE阶段驱动与实际硬件匹配连接,让相应的设备进行读写;TSL阶段为临时系统阶段,RT阶段提供运行时服务。

7.根据权利要求1所述的一种基于RISC-V服务器CPU的新型EDK2启动引导方法,其特征在于,对设备程序进行加载并执行,同时初始化硬件抽象层并运行DXE服务,同时,构建DXE阶段数据结构为BDS阶段提供服务信息,包括:

设备程序加载和执行:在DXE阶段,EDK2加载和执行设备程序,包括驱动程序和DXE Core;

硬件抽象层初始化:在DXE阶段,EDK2初始化并提供硬件抽象层,以便DXE服务与硬件设备进行交互;

DXE服务运行:DXE服务包括:提供系统配置管理、内存管理、引导设备选择的功能和服务;在DXE阶段,EDK2运行这些DXE服务;

构建DXE阶段数据结构:DXE阶段数据结构包括系统配置表和带有服务描述符的服务节点。

8.根据权利要求1-7任一所述的一种基于RISC-V服务器CPU的新型EDK2启动引导方法,其特征在于,GRUB加载真正的Linux内核映像和设备树,并设置必要的硬件配置和参数,顺利启动Linux内核,还包括:

GRUB使用EXT4文件系统驱动,进行EXT4分区文件系统的初始化和管理,包括创建分区、格式化、挂载和卸载操作;

GRUB在EXT4分区中寻找并解析Linux内核的启动配置文件,其中包括多个Linux内核的路径和参数设置,以确保正确启动Linux操作系统。

9.一种计算机设备,包括存储器和处理器,所述存储器存储有计算机程序,其特征在于,所述处理器执行所述计算机程序时实现基于RISC-V服务器CPU的新型EDK2启动引导方法的步骤。

10.一种计算机可读存储介质,其上存储有计算机程序,其特征在于,所述计算机程序被处理器执行时实现基于RISC-V服务器CPU的新型EDK2启动引导方法的步骤。

一种基于RISC-V服务器CPU的新型EDK2启动引导方法

技术领域

[0001] 本发明涉及计算机技术领域,特别涉及一种基于RISC-V服务器CPU的新型的EDK2启动引导方法。

背景技术

[0002] RISC-V是一种基于开放指令集架构(Open Instruction Set Architecture,ISA)的处理器架构,在服务器CPU领域正逐步崭露头角。与市场上主流的X86和ARM两大平台架构相比,RISC-V架构在服务器领域还处于起步阶段,但它已经开始在MCU产品领域展现出了强大的潜力并成为了ARM架构以外的另一种选择。RISC-V架构支持三种不同的工作模式:用户态模式U-mode(User Mode)、监管态模式S-mode(Supervisor Mode)和机器态模式M-mode(Machine Mode)。用户态模式提供了应用程序运行的标准模式,监管态模式用于操作系统和虚拟机管理,而机器态模式是用于进行系统级的低级操作和初始化等。随着RISC-V架构的不断发展和成熟,我们有理由相信它可以在服务器CPU领域逐渐获得一席之地,并展现出更加活跃的发展态势。

[0003] UEFI(Unified Extensible Firmware Interface)是一种新一代的系统固件接口标准,旨在取代传统的BIOS(Basic Input/Output System),UEFI最初是由Intel公司针对X86处理器进行定制的,但是UEFI标准是与架构无关的,通过不断的发展UEFI的接口标准已经普遍使用在了X86、ARM等处理器平台,其支持多个操作系统、驱动程序和应用程序的加载和执行。提供了一个可扩展的固件环境,具有更强大的功能和灵活性。

[0004] EDK2(EFIDevelopment Kit 2)是一个开发套件,可以帮助开发人员设计、创建和调试以UEFI(Unified Extensible Firmware Interface)为基础的固件。它包含用于构建UEFI BIOS和UEFI应用程序的开发库、工具、示例代码和文档。EDK2是一个开源项目,由Intel、HP、AMD、Insyde Software、Microsoft等公司共同维护。

[0005] OpenSBI(Open Source Secure Boot and Runtime Firmware是一个开放的固件,可以用于支持多个RISC-V设备。它提供了通用的启动和运行时固件功能,包括安全引导、运行时环境和操作系统启动,这些功能使RISC-V设备更具可靠性和安全性。OpenSBI旨在为RISC-V生态系统提供标准的启动和运行时环境,从而使RISC-V设备更易于开发、移植和维护。它可以在不同的处理器架构、操作系统和设备上运行,包括嵌入式设备、服务器、网络设备和智能手机。

[0006] 现有的基于RISC-V服务器CPU的EDK2启动引导方法实体表现为开源的UEFI的实现代码EDK2,作为固件代码其可以为RISC-V服务器的UEFI固件提供丰富的功能和可扩展性,如模块化架构、高级驱动支持、安全启动、图形界面。然而面向RISC-V服务器CPU使用EDK2时,所使用的EDK2代码引用OpenSBI固件作为库使用。具有以下问题:

[0007] 1.代码维护困难:由于EDK2和OpenSBI是两个不同的开源项目,分别由不同的团队维护,因此在EDK2中增加了代码的复杂性。当EDK2或OpenSBI其中一个更新时,可能需要耗费额外的工作来进行集成和解决可能出现的冲突。

[0008] 2.性能下降:OpenSBI的引入可能会导致额外的运行时开销,影响系统的性能。由于OpenSBI作为一种中间层,在调用EDK2的功能时可能会有额外的延迟。

[0009] 3.安全性问题:融合两个不同的开源项目可能增加攻击面和潜在的安全漏洞。由于EDK2和OpenSBI由不同的团队开发和维护,其中一个项目的安全漏洞可能会影响到整个系统的安全性。

发明内容

[0010] 本发明的目的主要为解决RISCV服务器CPU的启动问题,并针对现有的UEFI启动过程中edk与OpenSBI的冗余,做出了分离与重组的创新性流程,本发明提供了一种基于RISC-V服务器CPU的新型的EDK2启动引导方法。该方法将OpenSBI从EDK2的SEC阶段拆分出来并置于PEI和DXE阶段之间,使得OpenSBI在EDK2中具有灵活性和可拓展性,简化了调试和更新。在OpenSBI安全性验证和模式切换中增加了多样的可能性。最终实现了更可靠和灵活的Linux内核启动。

[0011] 术语解释:

[0012] 1、ZSBL(Zero Stage Bootloader):是一种引导加载程序,zsbl是系统启动过程中的第一个阶段,负责初始化硬件和一些基本设定,以及将系统控制权转交给下一个引导加载程序。zsbl通常是在ROM或者SoC芯片内部存储器中嵌入的,它使用最基本的指令集和硬件资源,以最小的系统资源消耗来完成其功能。

[0013] 2、DTB(Device Tree Binary):中文含义为设备树二进制文件,该文件描述设备树(device tree)架构。DTS(Device Tree Source,设备树源)文件是便于人类阅读和理解的源文件,dtb文件是dts文件被dtc(device tree compiler,设备树编译器)编译后形成的二进制格式文件,可以被linux内核解析。

[0014] 3、DDR(Double Data Rate):DDR是一种内存技术,能够在每个时钟周期内进行两次数据传输,提供更高的带宽和数据传输速度。在计算机中,DDR通常指代计算机内存中使用的双倍数据率类型的内存模块。

[0015] 4、PEI Core:PEI(Pre-EFI Initialization)是edk2框架中的一个阶段,PEI Core是PEI阶段的核心组件。PEI阶段是UEFI固件启动过程的一部分,它位于edk2中的前期初始化阶段。在PEI阶段,系统硬件和资源的初始化工作被执行,包括处理器的初始化、内存的初始化、中断控制器的配置等。PEI Core是PEI阶段的核心组件,它负责管理PEI阶段的执行流程和各个模块的初始化。

[0016] 5、PEIM(Pre-EFI Initialization Module):PEIM是由UEFI驱动程序提供的模块,用于初始化特定设备或提供系统服务。

[0017] 6、Hart ID:Hart ID是OpenSBI中的一个名词,代表处理器核心的唯一标识符。在RISC-V系统中,每个处理器核心(Hart)都有一个唯一的Hart ID,用于区分不同的处理器核心。Hart ID可以是一个数字或者其他形式的标识符,例如0、1、2等。Hart ID通常用于标识和区分处理器核心,在多核系统中可以用来进行进程调度、资源分配和通信等操作。

[0018] 7、BSS(Block Started by Symbol):代表程序的未初始化数据块,在计算机程序中,BSS段是存放全局未初始化的静态变量和全局静态变量的一块内存空间。这些变量在编译时已经声明,但并没有给予初始化的数值。BSS段通常在程序加载时会被清零,即所有的

变量都会被初始化为0或者空值。

[0019] 8、sbi_init:sbi_init是OpenSBI中的一个函数,用于初始化OpenSBI环境。sbi_init函数在OpenSBI的启动过程中被调用,它负责执行一些基本的初始化操作以准备OpenSBI运行。这些初始化操作包括:设置初始化中断控制器、时钟硬件环境,初始化处理器核心的状态信息、系统配置全局变量,设置异常处理用于处理系统中可能出现的页错误、非法指令异常情况。

[0020] 9、Irq(Interrupt Request):代表中断请求,在计算机系统中,中断是一种机制,可以中断处理器的正常执行流程,以便处理紧急事件或外部设备的请求。中断请求是指外部设备或软件触发的中断信号,用于通知处理器需要处理某个事件或响应某个请求。

[0021] 10、MMU(Memory Management Unit):代表内存管理单元,内存管理单元(MMU)是计算机系统中的硬件组件,用于管理处理器和系统之间的内存访问。它负责将虚拟地址(Virtual Address)转换为物理地址(Physical Address),并提供内存保护、地址映射、页表管理等功能。

[0022] 11、TLB(Translation Lookaside Buffer):TLB是一种高速缓存,用于存储虚拟地址到物理地址的转换结果。TLB用于加快处理器访问内存的速度,通过缓存最近使用的页表项,避免每次访问内存时的完整地址转换过程。

[0023] 12、Timer:Timer是系统中的计时器设备,用于提供时间基准和定时功能。初始化Timer涉及配置计时器的频率和模式,以及设置中断以实现定时器中断功能。

[0024] 13、sbi_hsm_prepare_next_jump:sbi_hsm_prepare_next_jump是OpenSBI中的一个函数,用于准备下一次跳转到安全监控模式(Hypervisor,Secure Monitor)的操作。

[0025] 本发明所采用的技术方案如下:

[0026] 一种基于RISC-V服务器CPU的新型EDK2启动引导方法,包括:

[0027] 在RISC-V服务器CPU上电后,ZSBL阶段进行最小范围初始化,并将DTB和EDK2加载到内存中;

[0028] 在SEC阶段和PEI阶段,EDK2完成安全性验证和EDK2前期初始化工作,并跳转到OpenSBI;

[0029] OpenSBI阶段,与硬件进行交互,完成底层初始化和设备初始化,并跳转回EDK2的DXE阶段;

[0030] DXE阶段,RISC-V服务器处于S-mode,对设备程序进行加载并执行,同时初始化硬件抽象层并运行DXE服务,同时,构建DXE阶段数据结构为BDS阶段提供服务信息;

[0031] GRUB加载真正的Linux内核映像和设备树,并设置必要的硬件配置和参数,必要的硬件配置包括硬件设备的检测、初始化和配置,参数包括启动命令行参数、内核参数、启动设备信息,顺利启动Linux内核。

[0032] 根据本发明优选的,ZSBL阶段进行最小范围初始化,此时RISC-V服务器处于M-mode,包括:ZSBL阶段,初始化处理器的基本环境,包括设置寄存器、中断向量表。

[0033] 根据本发明优选的,在RISC-V服务器CPU上电后,ZSBL阶段进行最小范围初始化,并将DTB和EDK2加载到内存中,之后还包括:

[0034] ZSBL阶段,进行内存DDR的初始化工作,内存DDR的初始化工作包括内存控制器初始化,内存训练,初始化DRAM以及内存映射,将EDK2的程序代码加到内存的指定位置,并通

过跳转指令将控制权传递给EDK2。

[0035] 根据本发明优选的,SEC阶段时,RISC-V服务器处于M-mode,在与OpenSBI分离后进行安全性验证:在安全环境中,SEC阶段进行一系列安全性验证,包括验证引导程序和固件的签名以及检查系统启动配置的完整性,寻址并跳转至PEI Core。

[0036] 根据本发明优选的,PEI阶段时,RISC-V服务器处于M-mode,进行EDK2前期初始化工作:运行PEIM。

[0037] 根据本发明优选的,在SEC阶段和PEI阶段,EDK2完成安全性验证和EDK2前期初始化工作,之后还包括:OpenSBI阶段时,RISC-V服务器处于M-mode,在与硬件进行交互时,完成处理器、内存和设备关键组件的初始化工作,设置处理器状态、初始化内存子系统,并与外部设备建立通信连接。

[0038] 根据本发明优选的,底层初始化,包括:判断hart id,代码重定位、清除除保存设备树地址的寄存器值、清除BSS段、设置栈指针、读取设备树中的设备信息、设备树重定位;

[0039] 通过执行sbi_init来进行设备初始化,包括:控制台的初始化、irq中断和核间中断的初始化、MMU的TLB表的初始化、timer的初始化;

[0040] 利用sbi_hsm_prepare_next_jump函数跳转到UEFI固件的DXE阶段入口函数。

[0041] 根据本发明优选的,RISC-V服务器处于S-mode下,EDK2继续执行还包括:

[0042] EDK2继续执行DXE阶段将设备程序进行加载并执行,BDS阶段进行DXE阶段驱动与实际硬件匹配连接,让相应的设备进行读写;TSL阶段为临时系统阶段,RT阶段提供运行时服务。

[0043] 根据本发明优选的,对设备程序进行加载并执行,同时初始化硬件抽象层并运行DXE服务,同时,构建DXE阶段数据结构为BDS阶段提供服务信息,包括:

[0044] 设备程序加载和执行:在DXE阶段,EDK2加载和执行设备程序,包括驱动程序和DXE Core;

[0045] 硬件抽象层初始化:在DXE阶段,EDK2初始化并提供硬件抽象层,以便DXE服务与硬件设备进行交互;

[0046] DXE服务运行:DXE服务包括:提供系统配置管理、内存管理、引导设备选择的功能和服务;在DXE阶段,EDK2运行这些DXE服务;

[0047] 构建DXE阶段数据结构:DXE阶段数据结构包括系统配置表(System Configuration Tables)和带有服务描述符的服务节点(Service Nodes with Service Descriptors)。

[0048] 根据本发明优选的,GRUB加载真正的Linux内核映像和设备树,并设置必要的硬件配置和参数,顺利启动Linux内核,还包括:

[0049] GRUB使用EXT4文件系统驱动,进行EXT4分区文件系统的初始化和管理,包括创建分区、格式化、挂载和卸载操作;

[0050] GRUB在EXT4分区中寻找并解析Linux内核的启动配置文件,其中包括多个Linux内核的路径和参数设置,以确保正确启动Linux操作系统。

[0051] 一种计算机设备,包括存储器和处理器,所述存储器存储有计算机程序,所述处理器执行所述计算机程序时实现基于RISC-V服务器CPU的新型EDK2启动引导方法的步骤。

[0052] 一种计算机可读存储介质,其上存储有计算机程序,所述计算机程序被处理器执

行时实现基于RISC-V服务器CPU的新型EDK2启动引导方法的步骤。

[0053] 本发明的有益效果在于：

[0054] 1、启动过程更加灵活：将OpenSBI置于PEI和DXE阶段之间，可以更灵活地使用OpenSBI功能，避免仅局限于SEC阶段进行初始化。

[0055] 2、易于扩展：将OpenSBI置于PEI和DXE之间，更容易扩展OpenSBI的功能，例如添加新的硬件支持和调整引导策略等。

[0056] 3、简化调试与更新：将OpenSBI置于PEI和DXE之间，整个启动过程变得更易于调试和理解，OpenSBI的更新和调试也变得更加简单。

[0057] 4、提高安全性：可通过在OpenSBI中进行安全性验证，提高安全性监控的效率。在EDK2中，还可以更灵活地使用UEFI固件提供的安全协议和策略。

[0058] 5、提供模式切换：在RISC-V启动阶段提供了SEC与PEI阶段置于M-MODE的可选方案。

[0059] 本发明通过将OpenSBI置于PEI和DXE之间、减少SEC阶段OpenSBI库的冗余，使得OpenSBI在EDK2中具有灵活性和可拓展性，简化了调试和更新。在OpenSBI安全性验证和模式切换方面增加了多种可能性。本发明适用于RISC-V架构的服务器领域，为服务器系统提供了可靠和灵活的启动技术解决方案。

附图说明

[0060] 为了更清楚地说明本发明实施例或现有技术中的技术方案，下面将对实施例或现有技术描述中所需要使用的附图作简单地介绍，显而易见地，下面描述中的附图仅仅是本发明的一些实施例，对于本领域普通技术人员来讲，在不付出创造性劳动的前提下，还可以根据这些附图示出的结构获得其他的附图。

[0061] 图1为本发明提出的一种基于RISC-V服务器CPU的新型的EDK2启动引导方法的流程示意图。

[0062] 图2为本发明提出的一种基于RISC-V服务器CPU的新型的EDK2启动引导方法的在RISC-V服务器CPU上基于EDK2各个阶段的详细启动流程示意图。

具体实施方式

[0063] 下面结合说明书附图和实施例对本发明作进一步限定，但不限于此。

[0064] 实施例1

[0065] 一种基于RISC-V服务器CPU的新型EDK2启动引导方法，如图1所示，体现了OpenSBI分离与重构的思想，指出了OpenSBI在EDK2中段运行的特征。包括：

[0066] 在RISC-V服务器CPU上电后，ZSBL阶段进行最小范围初始化，并将DTB和EDK2加载到内存中；

[0067] 在SEC阶段和PEI阶段，EDK2完成安全性验证和EDK2前期初始化工作，并跳转到OpenSBI；

[0068] OpenSBI阶段，与硬件进行交互，完成底层初始化和设备初始化，并跳转回EDK2的DXE阶段；

[0069] DXE阶段，RISC-V服务器处于S-mode，在s-mode下，EDK2继续执行，在后续阶段进入

UEFI SHELL并加载操作系统加载器(GRUB)。对设备程序进行加载并执行,同时初始化硬件抽象层并运行DXE服务,同时,构建DXE阶段数据结构为BDS阶段提供服务信息;

[0070] GRUB加载真正的Linux内核映像和设备树,并设置必要的硬件配置和参数,必要的硬件配置包括硬件设备的检测、初始化和配置,例如,处理器、内存控制器、磁盘控制器、图形显示适配器;参数包括启动命令行参数、内核参数、启动设备信息,顺利启动Linux内核。硬件初始化:GRUB需要初始化和配置计算机的硬件设备,以便操作系统在启动后能够正确地与这些设备进行通信。这可能等。引导参数设置:GRUB可以配置和传递一些引导参数给操作系统内核。这些参数可以非常重要。

[0071] 本发明提供一种基于RISC-V服务器CPU的新型的EDK2启动引导方法,能够提供提供安全和灵活的服务器CPU操作系统启动过程,确保处理器、内存和设备等关键组件正确初始化和配置,从而实现可靠的Linux操作系统的启动。该方法将OpenSBI从SEC阶段拆分出来并置于PEI和DXE阶段之间,提供了更简单的调试和更新方案,适用于RISC-V架构的服务器CPU。

[0072] 实施例2

[0073] 根据实施例1所述的一种基于RISC-V服务器CPU的新型EDK2启动引导方法,其区别在于:

[0074] 如图2所示,ZSBL阶段进行最小范围初始化,此时RISC-V服务器处于M-mode,包括:ZSBL阶段,初始化处理器的基本环境,包括设置寄存器、中断向量表。

[0075] 在RISC-V服务器CPU上电后,ZSBL阶段进行最小范围初始化,并将DTB和EDK2加载到内存中,之后还包括:

[0076] ZSBL阶段,进行内存DDR的初始化工作,内存DDR的初始化工作包括内存控制器初始化,内存训练,初始化DRAM以及内存映射,将EDK2的程序代码加到内存的指定位置,并通过跳转指令将控制权传递给EDK2。等待EDK2进行进一步的初始化任务。

[0077] SEC阶段时,RISC-V服务器处于M-mode,在与OpenSBI分离后进行安全性验证:在安全环境中,SEC阶段进行一系列安全性验证,以确保服务器的启动是可信的。这包括验证引导程序和固件的签名以及检查系统启动配置的完整性,寻址并跳转至PEI Core。

[0078] PEI阶段时,RISC-V服务器处于M-mode,进行EDK2前期初始化工作:运行PEIM。这些PEIM可以被认为是UEFI固件的“早期驱动程序”,用于EDK2中管理硬件设备。

[0079] 在SEC阶段和PEI阶段,EDK2完成安全性验证和EDK2前期初始化工作,之后还包括:OpenSBI阶段时,RISC-V服务器处于M-mode,在与硬件进行交互时,完成处理器、内存和设备关键组件的初始化工作,设置处理器状态、初始化内存子系统,并与外部设备建立通信连接。

[0080] 底层初始化,包括:判断hart id,代码重定位、清除除保存设备树地址的寄存器值、清除BSS段、设置栈指针、读取设备树中的设备信息、设备树重定位;以为后续EDK2(UEFI固件)阶段提供信息。

[0081] 当底层初始化完成后,通过执行sbi_init来进行设备初始化,包括:控制台的初始化、irq中断和核间中断的初始化、MMU的TLB表的初始化、timer的初始化;

[0082] 利用sbi_hsm_prepare_next_jump函数跳转到UEFI固件的DXE阶段入口函数。当这些组件初始化完成后,OpenSBI将利用sbi_hsm_prepare_next_jump函数跳转到EDK2(UEFI

固件)的DXE阶段入口函数,并处于S-mode状态下,将控制权交给DXE阶段,从而继续系统的启动过程。通过这个过程,确保硬件和基本的运行环境正确初始化,为后续的操作系统启动做好准备。

[0083] 对设备程序进行加载并执行,同时初始化硬件抽象层并运行DXE服务,同时,构建DXE阶段数据结构为BDS阶段提供服务信息,包括:

[0084] 设备程序加载和执行:在DXE阶段,EDK2加载和执行设备程序,包括驱动程序(device drivers)和DXE Core(DXE核心);

[0085] 硬件抽象层(HAL)初始化:硬件抽象层是一个软件接口层,用于提供对硬件设备的访问和管理。在DXE阶段,EDK2初始化并提供硬件抽象层,以便DXE服务与硬件设备进行交互;

[0086] DXE服务运行:DXE服务是在DXE Core的基础上运行的,DXE服务包括:提供系统配置管理、内存管理、引导设备选择的功能和服务;在DXE阶段,EDK2运行这些DXE服务;以完成各种任务。

[0087] 构建DXE阶段数据结构:DXE阶段还会构建一些数据结构,用于为BDS阶段提供服务信息。DXE阶段数据结构包括系统配置表(System Configuration Tables)和带有服务描述符的服务节点(Service Nodes with Service Descriptors)。系统配置表用于存储系统的相关信息,提供给后续的BDS阶段使用。服务节点则用于描述DXE服务的属性和功能,也会在BDS阶段被使用。

[0088] 实施例3

[0089] 根据实施例2所述的一种基于RISC-V服务器CPU的新型EDK2启动引导方法,其区别在于:

[0090] RISC-V服务器处于S-mode下,EDK2继续执行还包括:

[0091] EDK2继续执行DXE阶段将设备程序进行加载并执行,BDS阶段进行DXE阶段驱动与实际硬件匹配连接,让相应的设备进行读写;TSL阶段为临时系统阶段,是EDK2(UEFI固件)与操作系统的过度阶段,此阶段UEFI提供了人机交互接口,可以进行部分系统工具的运行,并使用操作系统加载器如GRUB。RT阶段提供运行时服务。本阶段,操作系统加载器获得全部的控制权,EDK2(UEFI固件)只进行保留运行时服务提供给操作系统加载器与操作系统使用。

[0092] GRUB加载真正的Linux内核映像和设备树,并根据设备树的描述信息进行硬件初始化和配置。它设置处理器特性、内存映射、中断和定时器等参数。最后,EDK2将控制权转交给Linux内核,从而启动操作系统的完整环境。这个过程确保了Linux内核和硬件之间的正确交互和配置,为操作系统的正常运行提供了必要的基础。

[0093] GRUB加载真正的Linux内核映像和设备树,并设置必要的硬件配置和参数,顺利启动Linux内核,还包括:

[0094] GRUB使用EXT4文件系统驱动,进行EXT4分区文件系统的初始化和管理,包括创建分区、格式化、挂载和卸载操作;

[0095] GRUB在EXT4分区中寻找并解析Linux内核的启动配置文件,其中包括多个Linux内核的路径和参数设置,以确保正确启动Linux操作系统。

[0096] 实施例4

[0097] 一种计算机设备,包括存储器和处理器,存储器存储有计算机程序,处理器执行计算机程序时实现实施例1-3任一所述的基于RISC-V服务器CPU的新型EDK2启动引导方法的步骤。

[0098] 实施例5

[0099] 一种计算机可读存储介质,其上存储有计算机程序,计算机程序被处理器执行时实现实施例1-3任一所述的基于RISC-V服务器CPU的新型EDK2启动引导方法的步骤。

[0100] 上述本发明实施例序号仅仅为了描述,不代表实施例的优劣。

[0101] 通过以上的实施方式的描述,本领域的技术人员可以清楚地了解到上述实施例方法可借助软件加必需的通用硬件平台的方式来实现,当然也可以通过硬件,但很多情况下前者是更佳的实施方式。基于这样的理解,本发明的技术方案本质上或者说对现有技术做出贡献的部分可以以软件产品的形式体现出来,计算机软件产品存储在一个存储介质(如ROM/RAM、磁碟、光盘)中,包括若干指令用以使得一台终端(可以是手机,计算机,服务器,空调器,或者网络设备等)执行本发明各个实施例所述的方法。

[0102] 上面结合附图对本发明的实施例进行了描述,但是本发明并不局限于上述的具体实施方式,上述的具体实施方式仅仅是示意性的,而不是限制性的,本领域的普通技术人员在本发明的启示下,在不脱离本发明宗旨和权利要求所保护的范围情况下,还可做出很多形式,这些均属于本发明的保护之内。

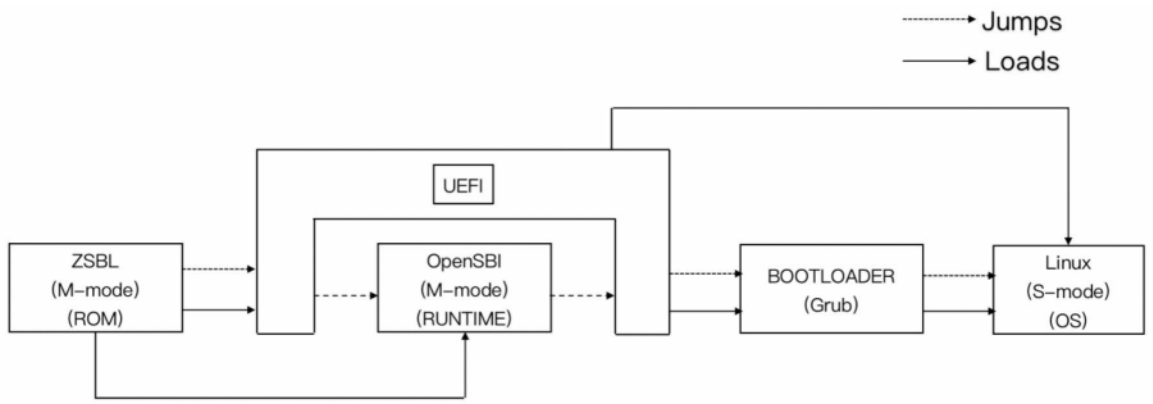

Jumps
Loads
UEFI
ZSBL
(M-mode)
(ROM)
OpenSBI
(M-mode)
(RUNTIME)
BOOTLOADER
(Grub)
Linux
(S-mode)
(OS)

图1

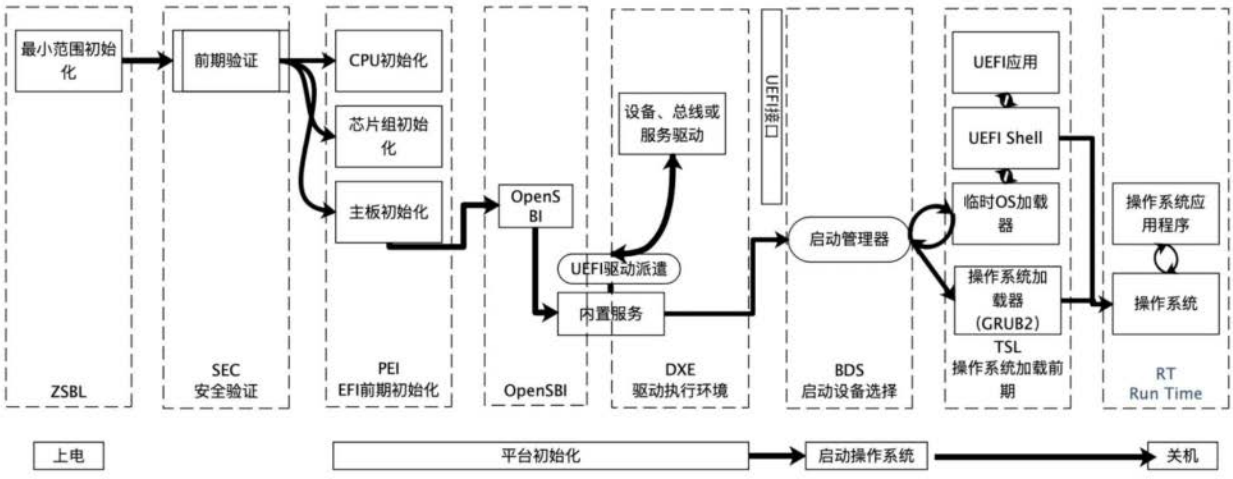

最小范围初始化
前期验证
CPU初始化
芯片组初始化
主板初始化
OpenSBI
设备、总线或服务驱动
UEFI驱动派遣
内置服务
UEFI接口
启动管理器
UEFI应用
UEFI Shell
临时OS加载器
操作系统加载器(GRUB2)
操作系统应用程序
操作系统
ZSBL
SEC
安全验证
PEI
EFI前期初始化
OpenSBI
DXE
驱动执行环境
BDS
启动设备选择
TSL
操作系统加载前期
RT
Run Time
上电
平台初始化
启动操作系统
关机

图2