(19)国家知识产权局

(12)发明专利

(10)授权公告号 CN 115421864 B
(45)授权公告日 2023.04.28

(21)申请号 202211114604.X

(22)申请日 2022.09.14

(65)同一申请的已公布的文献号
申请公布号 CN 115421864 A

(43)申请公布日 2022.12.02

(73)专利权人 北京计算机技术及应用研究所
地址 100854 北京市海淀区永定路51号

(72)发明人 贾张涛 孔祥炳 付修锋 安恒勉斌 邵飒 安顺 李雅斯 金政宇

(74)专利代理机构 中国兵器工业集团公司专利中心 11011
专利代理师 刘瑞东

(51)Int.Cl.
G06F 9/455(2006.01)

(56)对比文件
CN 111026504 A,2020.04.17
US 2005076186 A1,2005.04.07
US 2006026563 A1,2006.02.02
US 2008222384 A1,2008.09.11
US 6212614 B1,2001.04.03
WO 2021249193 A1,2021.12.16
邸强.基于TTA可配置处理器的指令集仿真器及集成开发环境的设计.中国优秀硕士学位论文全文数据库信息科技辑(月刊).2012,(第2期),I137-73.

审查员 赵静

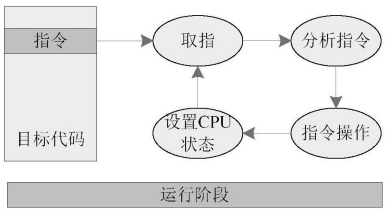
权利要求书3页 说明书9页 附图1页

(54)发明名称

一种通用的PowerPC架构处理器指令集虚拟化仿真方法

(57)摘要

本发明涉及一种通用的PowerPC架构处理器指令集虚拟化仿真方法,属于虚拟仿真领域。本发明针对飞思卡尔(FreeScale)基于Power Architecture的32位微处理器核心e300、e500、e600,设计一种通用的处理器仿真框架,同时支持多种架构处理器的仿真运行。提供PowerPC处理器嵌入式处理器软件的运行环境,为PowerPC处理器嵌入式处理器软件的开发提供支撑。本发明提出的方案,能够实现PowerPC指令架构处理器的指令集仿真,仿真精度高;本发明采用数组、链表等实现寄存器、内存的仿真,能够完成仿真PowerPC指令架构处理器。



CN 115421864 B

1.一种通用的PowerPC架构处理器指令集虚拟化仿真方法,其特征在于,该方法包括如下步骤:

S1、PowerPC指令架构目标文件解析

打开PowerPC指令架构编译后的可执行文件*.exe,获取Program Header对应的代码段;

S2、PowerPC指令架构内存仿真及代码加载

采用链表结构对PowerPC指令架构处理器内存进行仿真,记为PowerPC-VMemory,并加载Program Header对应的代码段,模拟PowerPC指令架构的内存读写操作;

S3、PowerPC指令架构寄存器仿真

通过数组模拟寄存器,通过数组操作模拟寄存器读写操作,实现寄存器的模拟;

S4、PowerPC指令架构指令描述

每条指令有一个唯一的表示,每条指令分为不同的段;采用将指令分段的方式进行描述,获取指令Instruction的操作码opcode信息,构建指令操作码信息数组PowerPC-InstructionDesp[425],对POWERPC架构指令集架构的425条指令进行描述,并存储到指令操作码信息数组PowerPC-InstructionDesp[425]中;

S5、PowerPC指令架构读取指令

根据程序计数器PC的值,从仿真内存PowerPC-VMemory中读取指令,获取当前PC对应的指令PC-Instruction;

S6、PowerPC指令架构指令译码

根据PowerPC指令架构指令描述,逐个计算PC-Instruction与PowerPC-InstructionDesp[425]的对应bit位是否相同,若PC-Instruction与PowerPC-InstructionDesp[425]中第k个指令描述对应的位置相同,则完成指令译码,记为Instruction_k;

S7、PowerPC指令架构指令翻译

对PowerPC指令架构指令集进行功能翻译,设计在虚拟仿真处理器运行的对每条指令进行操作模拟的函数,保证每一条指令和函数处理后,硬件处理器和虚拟仿真处理器的内存、寄存器保持一致;并将函数指针存储到PowerPC-InstructionInterp[425]数组中,存储顺序与PowerPC-InstructionDesp[425]的指令顺序保持一致;

S8、PowerPC指令架构Linux系统调用实现

解析系统调用,并根据系统调用的类型,对系统调用进行相应的处理,并对相应的寄存器进行置位操作;

S9、PowerPC指令架构指令执行

根据指令数量,进行循环取指、译码、指令翻译,并根据函数指针,执行指令操作;连续仿真,直到完成所有指令执行,并处理系统调用,实现针对PowerPC指令架构的处理器指令集仿真。

2.如权利要求1所述的通用的PowerPC架构处理器指令集虚拟化仿真方法,其特征在于,所述步骤S1具体包括:

S11、打开PowerPC指令架构编译后的可执行文件*.exe,读取文件信息;

S12、按照PowerPC指令架构编译后的可执行文件的格式,读取文件信息中File

Header、Section Header、Program Header信息，获取Program Header对应的代码段。

3.如权利要求2所述的通用的PowerPC架构处理器指令集虚拟化仿真方法，其特征在于，所述步骤S2具体包括：

S21、采用链表结构对PowerPC指令架构处理器内存进行仿真，其中链表结构体PowerPC-MemoryPage大小为256个字；

S22、将PowerPC指令架构内存记为PowerPC-VMemory；将S1获取的Program Header对应的代码段，写入到PowerPC指令架构芯片仿真内存PowerPC-VMemory中，并根据代码段的大小，维护整个仿真内存结构。

4.如权利要求3所述的通用的PowerPC架构处理器指令集虚拟化仿真方法，其特征在于，所述链表结构体包括：起始虚拟地址MemoryPageBeginAddress、结束虚拟地址MemoryPageEndAddress、内存块MemoryPage、下一个内存节点指针NEXT和前一个内存节点指针PRE。

5.如权利要求3所述的通用的PowerPC架构处理器指令集虚拟化仿真方法，其特征在于，所述步骤S22中，如果代码段不大于一个MemoryPage的大小256，创建一个MemoryPage，并将代码段写到对用的MemoryPage数组中去；如果代码段大于256，创建多个MemoryPage，并将代码段按顺序写到对应的MemoryPage数组中去，并维护各个节点的MemoryPageBeginAddress、MemoryPageEndAddress信息，保证链表结构信息的正确性。

6.如权利要求2-5任一项所述的通用的PowerPC架构处理器指令集虚拟化仿真方法，其特征在于，所述步骤S3具体包括：

S31、对PowerPC指令架构e300、e500、e600芯片共计197个寄存器进行仿真，通过数组PowerPC-Register模拟寄存器，实现寄存器的模拟；

S32、通过访问PowerPC-Register，对相应的数组中的元素进行操作模拟PowerPC指令架构的寄存器操作，实现对PowerPC指令架构的寄存器仿真。

7.如权利要求6所述的通用的PowerPC架构处理器指令集虚拟化仿真方法，其特征在于，所述步骤S4具体包括：指令操作码信息数组PowerPC-InstructionDesp的结构包括：name、contentNumber和content，其中，name是指令名称，contentNumber指令分段数量，content用于存储opcode指令译码信息。

8.如权利要求7所述的通用的PowerPC架构处理器指令集虚拟化仿真方法，其特征在于，所述步骤S5具体包括：

S51、根据程序计数器PC的值，从S2维护的PowerPC-VMemory中读取指令；

S52、根据PowerPC-MemoryPage中的MemoryPageBeginAddress、MemoryPageEndAddress的值，计算出PC属于PowerPC-MemoryPage，记为PowerPC-MemoryPage-i，地址偏移为PC-MemoryPageBeginAddress，指令内容为PowerPC-MemoryPage-i对应的MemoryPage[PC-MemoryPageBeginAddress]，记为PC-Instruction。

9.如权利要求1所述的通用的PowerPC架构处理器指令集虚拟化仿真方法，其特征在于，所述步骤S7具体包括：

S71、对PowerPC指令架构指令集中的指令进行功能翻译，并设计在虚拟仿真处理器运行的对每条指令进行操作模拟的函数，保证每一条指令和函数处理后，硬件处理器和虚拟仿真处理器的内存、寄存器保持一致；

S72:翻译PowerPC指令架构的425条指令,并将函数指针存储到指令翻译译码数组PowerPC-InstructionInterp[425]数组中,存储顺序与PowerPC-InstructionDesp[425]的指令顺序保持一致;

S73、根据S6计算出的Instruction_k,获得该指令对应的函数中指针。

10.如权利要求9所述的通用的PowerPC架构处理器指令集虚拟化仿真方法,其特征在于,所述步骤S9具体包括:

S91、构建PowerPC指令架构的仿真框架,根据指令数量,进行循环取指、译码、指令翻译,并根据函数指针,执行指令操作;

S92、连续仿真,直到完成所有指令执行,并处理系统调用,实现针对PowerPC指令架构的处理器指令集仿真。

一种通用的PowerPC架构处理器指令集虚拟化仿真方法

技术领域

[0001] 本发明属于虚拟仿真领域,具体涉及一种通用的PowerPC架构处理器指令集虚拟化仿真方法。

背景技术

[0002] 指令集仿真是处理器虚拟化技术最重要的支撑技术,指令集仿真允许特定指令集上的软件运行在另一类异构的指令集上。在指令集层次上实现虚拟化,实际上就是将某个硬件平台上的二进制代码转换为另一个硬件平台上的二进制代码,从而实现不同指令集间的兼容,这一技术也被称为二进制翻译。虚拟化技术实现有2种主要方式:解释执行、动态二进制翻译。

[0003] 解释器对源二进制代码逐条进行分析,根据译码结果即指令类型,分解相应的解释例程执行。解释例程在一个由软件维护的源体系结构(包括各种结构寄存器、内存状态等)上用等价的一条或多条目标指令来模拟源指令的执行,获得和源指令同样的执行效果。解释器工作过程主要包括“取指令—>分析指令—>完成指令所需的操作—>修改处理器状态”等步骤,如此循环。

[0004] 基于解释执行的仿真器在主机中维护一个精确的处理器数据结构,具有很高的仿真精度,可以实现精确的寄存器、存储器、流水线,除了模拟源程序的功能外,可以得到精确的性能指标,如每条指令在流水线中的时钟周期,堆栈模拟等。

[0005] PowerPC(英语:Performance Optimization With Enhanced RISC-Performance Computing,有时简称PPC)是一种精简指令集(RISC)架构的中央处理器(CPU),其基本的设计源自IBM的POWER(Performance Optimized With Enhanced RISC;《IBM Connect电子报》2007年8月号译为“增强RISC性能优化”)架构。

[0006] PowerPC处理器有非常强的嵌入式表现,因为它具有优异的性能、较低的能量损耗以及较低的散热量。除了像串行和以太网控制器那样的集成I/O,该嵌入式处理器与台式机CPU存在非常显著的区别。PowerPC处理器有32个(32位或64位)GPR(通用寄存器)以及诸如PC(程序计数器,也称为IAR/指令地址寄存器或NIP/下一指令指针)、LR(链接寄存器)、CR(条件寄存器)等各种其它寄存器。

发明内容

[0007] (一)要解决的技术问题

[0008] 本发明要解决的技术问题是如何提供一种通用的PowerPC架构处理器指令集虚拟化仿真方法,以解决PowerPC架构处理器指令集虚拟化仿真的问题。

[0009] (二)技术方案

[0010] 为了解决上述技术问题,本发明提出一种通用的PowerPC架构处理器指令集虚拟化仿真方法,该方法包括如下步骤:

[0011] S1、PowerPC指令架构目标文件解析

[0012] 打开PowerPC指令架构编译后的可执行文件*.exe,获取Program Header对应的代码段;

[0013] S2、PowerPC指令架构内存仿真及代码加载

[0014] 采用链表结构对PowerPC指令架构处理器内存进行仿真,记为PowerPC-VMemory,并加载Program Header对应的代码段,模拟PowerPC指令架构的内存读写操作;

[0015] S3、PowerPC指令架构寄存器仿真

[0016] 通过数组模拟寄存器,通过数组操作模拟寄存器读写操作,实现寄存器的模拟;

[0017] S4、PowerPC指令架构指令描述

[0018] 每条指令有一个唯一的表示,每条指令分为不同的段;采用将指令分段的方式进行描述,获取指令Instruction的操作码opcode信息,构建指令操作码信息数组PowerPC-InstructionDesp[425],对POWERPC架构指令集架构的425条指令进行描述,并存储到指令操作码信息数组PowerPC-InstructionDesp[425]中;

[0019] S5、PowerPC指令架构读取指令

[0020] 根据程序计数器PC的值,从仿真内存PowerPC-VMemory中读取指令,获取当前PC对应的指令PC-Instruction;

[0021] S6、PowerPC指令架构指令译码

[0022] 根据PowerPC指令架构指令描述,逐个计算PC-Instruction与PowerPC-InstructionDesp[425]的对应bit位是否相同,若PC-Instruction与PowerPC-InstructionDesp[425]中第k个指令描述对应的位置相同,则完成指令译码,记为Instruction_k;

[0023] S7、PowerPC指令架构指令翻译

[0024] 对PowerPC指令架构指令集进行功能翻译,设计在虚拟仿真处理器运行的对每条指令进行操作模拟的函数,保证每一条指令和函数处理后,硬件处理器和虚拟仿真处理器的内存、寄存器保持一致;并将函数指针存储到PowerPC-InstructionInterp[425]数组中,存储顺序与PowerPC-InstructionDesp[425]的指令顺序保持一致;

[0025] S8、PowerPC指令架构Linux系统调用实现

[0026] 解析系统调用,并根据系统调用的类型,对系统调用进行相应的处理,并对相应的寄存器进行置位操作;

[0027] S9、PowerPC指令架构指令执行

[0028] 根据指令数量,进行循环取指、译码、指令翻译,并根据函数指针,执行指令操作;连续仿真,直到完成所有指令执行,并处理系统调用,实现针对PowerPC指令架构的处理器指令集仿真。

[0029] 进一步地,所述步骤S1具体包括:

[0030] S11、打开PowerPC指令架构编译后的可执行文件*.exe,读取文件信息;

[0031] S12、按照PowerPC指令架构编译后的可执行文件的格式,读取文件信息中File Header、Section Header、Program Header信息,获取Program Header对应的代码段。

[0032] 进一步地,所述步骤S2具体包括:

[0033] S21、采用链表结构对PowerPC指令架构处理器内存进行仿真,其中链表结构体PowerPC-MemoryPage大小为256个字;

[0034] S22、将PowerPC指令架构内存记为PowerPC-VMemory;将S1获取的Program Header对应的代码段,写入到PowerPC指令架构芯片仿真内存PowerPC-VMemory中,并根据代码段的大小,维护整个仿真内存结构。

[0035] 进一步地,所述链表结构体包括:起始虚拟地址MemoryPageBeginAddress、结束虚拟地址MemoryPageEndAddress、内存块MemoryPage、下一个内存节点指针NEXT和前一个内存节点指针PRE。

[0036] 进一步地,所述步骤S22中,如果代码段不大于一个MemoryPage的大小256,创建一个MemoryPage,并将代码段写到对用的MemoryPage数组中去;如果代码段大于256,创建多个MemoryPage,并将代码段按顺序写到对应的MemoryPage数组中去,并维护各个节点的MemoryPageBeginAddress、MemoryPageEndAddress信息,保证链表结构信息的正确性。

[0037] 进一步地,所述步骤S3具体包括:

[0038] S31、对PowerPC指令架构e300、e500、e600芯片共计197个寄存器进行仿真,通过数组PowerPC-Register模拟寄存器,实现寄存器的模拟;

[0039] S32、通过访问PowerPC-Register,对相应的数组中的元素进行操作模拟PowerPC指令架构的寄存器操作,实现对PowerPC指令架构的寄存器仿真。

[0040] 进一步地,所述步骤S4具体包括:指令操作码信息数组PowerPC-InstructionDesp的结构包括:name、contentNumber和content,其中,name是指令名称,contentNumber指令分段数量,content用于存储opcode指令译码信息。

[0041] 进一步地,所述步骤S5具体包括:

[0042] S51、根据程序计数器PC的值,从S2维护的PowerPC-VMemory中读取指令;

[0043] S52、根据PowerPC-MemoryPage中的MemoryPageBeginAddress、MemoryPageEndAddress的值,计算出PC属于PowerPC-MemoryPage,记为PowerPC-MemoryPage-i,地址偏移为PC-MemoryPageBeginAddress,指令内容为PowerPC-MemoryPage-i对应的MemoryPage[PC-MemoryPageBeginAddress],记为PC-Instruction。

[0044] 进一步地,所述步骤S7具体包括:

[0045] S71、对PowerPC指令架构指令集中的指令进行功能翻译,并设计在虚拟仿真处理器运行的对每条指令进行操作模拟的函数,保证每一条指令和函数处理后,硬件处理器和虚拟仿真处理器的内存、寄存器保持一致;

[0046] S72:翻译PowerPC指令架构的425条指令,并将函数指针存储到指令翻译译码数组PowerPC-InstructionInterp[425]数组中,存储顺序与PowerPC-InstructionDesp[425]的指令顺序保持一致;

[0047] S73、根据S6计算出的Instruction_k,获得该指令对应的函数中指针。

[0048] 进一步地,所述步骤S9具体包括:

[0049] S91、构建PowerPC指令架构的仿真框架,根据指令数量,进行循环取指、译码、指令翻译,并根据函数指针,执行指令操作;

[0050] S92、连续仿真,直到完成所有指令执行,并处理系统调用,实现针对PowerPC指令架构的处理器指令集仿真。

[0051] (三)有益效果

[0052] 本发明提出一种通用的PowerPC架构处理器指令集虚拟化仿真方法,为嵌入式处

理软件提供虚拟化的运行环境，本发明提出一种基于解释执行的PowerPC指令架构处理器虚拟化仿真技术方案，本发明针对飞思卡尔(FreeScale)基于Power Architecture的32位微处理器核心e300、e500、e600，设计一种通用的处理器仿真框架，同时支持多种架构处理器的仿真运行。提供PowerPC处理器嵌入式处理器软件的运行环境，为PowerPC处理器嵌入式处理器软件的开发提供支撑。

[0053] 本发明提出的方案，能够实现PowerPC指令架构处理器的指令集仿真，仿真精度高；本发明采用数组、链表等实现寄存器、内存的仿真，能够完成仿真PowerPC指令架构处理器。

附图说明

[0054] 图1为解释执行技术原理图；

[0055] 图2为指令信息示意图。

具体实施方式

[0056] 为使本发明的目的、内容和优点更加清楚，下面结合附图和实施例，对本发明的具体实施方式作进一步详细描述。

[0057] 为嵌入式处理软件提供虚拟化的运行环境，本发明提出一种基于解释执行的PowerPC指令架构处理器虚拟化仿真技术方案，本发明针对飞思卡尔(FreeScale)基于Power Architecture的32位微处理器核心e300、e500、e600，设计一种通用的处理器仿真框架，同时支持多种架构处理器的仿真运行。提供PowerPC处理器嵌入式处理器软件的运行环境，为PowerPC处理器嵌入式处理器软件的开发提供支撑。

[0058] S1、PowerPC指令架构目标文件解析

[0059] 打开PowerPC指令架构编译后的可执行文件*.exe，获取Program Header对应的代码段；

[0060] 其中，PowerPC处理器文件格式为ELF，根据ELF文件格式文件信息，读取文件信息中的File Header、Section Header、Program Header等信息，获取Program Header对应的代码段；

[0061] S2、PowerPC指令架构内存仿真及代码加载

[0062] 为提高内存仿真的可扩展性，采用链表结构对PowerPC指令架构处理器内存进行仿真，记为PowerPC-VMemory，并加载Program Header对应的代码段，模拟PowerPC指令架构的内存读写操作。

[0063] S3、PowerPC指令架构寄存器仿真

[0064] 对PowerPC指令架构e300、e500、e600芯片共计197个寄存器进行仿真，分为32个GPR通用寄存器、32个FPR浮点处理寄存器、51个特殊寄存器、41个e600专用特殊寄存器、41个e500专用特殊寄存器，寄存器共计197个，本发明通过数组模拟寄存器，通过数组操作模拟寄存器读写操作，实现寄存器的模拟。

[0065] S4、PowerPC指令架构指令描述

[0066] 每条指令有一个唯一的表示，每条指令分为不同的段；为准确描述指令结构，本发明采用将指令分段的方式进行描述，获取指令Instruction的操作码opcode信息，构建指令

操作码信息数组PowerPC-InstructionDesp[425],对POWERPC架构指令集架构的425条指令进行描述,并存储到指令操作码信息数组PowerPC-InstructionDesp[425]中。

[0067] S5、PowerPC指令架构读取指令

[0068] 根据程序计数器PC(Program Count)的值,从仿真内存PowerPC-VMemory中读取指令,获取当前PC对应的指令PC-Instruction;

[0069] S6、PowerPC指令架构指令译码

[0070] POWERPC架构指令架构共包含425条指令集,根据PowerPC指令架构指令描述,逐个计算PC-Instruction与PowerPC-InstructionDesp[425]的对应bit位是否相同,若PC-Instruction与PowerPC-InstructionDesp[425]中第k个指令描述对应的位置相同,则完成指令译码,记为Instruction_k。

[0071] S7、PowerPC指令架构指令翻译

[0072] 对PowerPC指令架构指令集进行功能翻译,设计在虚拟仿真处理器运行的对每条指令进行操作模拟的函数,保证每一条指令和函数处理后,硬件处理器和虚拟仿真处理器的内存、寄存器保持一致;并将函数指针存储到PowerPC-InstructionInterp[425]数组中,存储顺序与PowerPC-InstructionDesp[425]的指令顺序保持一致;

[0073] S8、PowerPC指令架构Linux系统调用实现

[0074] 解析系统调用,并根据系统调用的类型,对系统调用进行相应的处理,并对相应的寄存器进行置位操作;

[0075] S9、PowerPC指令架构指令执行

[0076] 根据指令数量,进行循环取指、译码、指令翻译,并根据函数指针,执行指令操作;连续仿真,直到完成所有指令执行,并处理系统调用,实现针对PowerPC指令架构的处理器指令集仿真。

[0077] 实施例1:

[0078] 本发明提出一种基于解释执行的PowerPC指令架构处理器虚拟化仿真技术方案,通过指令集仿真

[0079] S1、PowerPC指令架构目标文件解析

[0080] S11、打开PowerPC指令架构编译后的可执行文件*.exe,读取文件信息;

[0081] S12、按照PowerPC指令架构编译后的可执行文件的格式,读取文件信息中File Header、Section Header、Program Header等信息,获取Program Header对应的代码段;

[0082] S2、PowerPC指令架构内存仿真及代码加载

[0083] S21、为提高内存仿真的可扩展性,采用链表结构对PowerPC指令架构处理器内存进行仿真,其中链表结构体PowerPC-MemoryPage大小为256个字(word),链表结构体包括:起始虚拟地址MemoryPageBeginAddress、结束虚拟地址MemoryPageEndAddress、内存块MemoryPage、下一个内存节点指针NEXT和前一个内存节点指针PRE;

[0084]
```
PowerPC-MemoryPage {
Unsigned int MemoryPageBeginAddress; //虚拟地址
Unsigned int MemoryPageEndAddress; //虚拟地址
MemoryPage[ 256 ];//内存块大小
PowerPC-MemoryPage NEXT; //指向下一个内存节点
PowerPC-MemoryPage PRE; //指向前一个内存节点
    }
```

[0085] S22、将PowerPC指令架构内存记为PowerPC-VMemory;将S1获取的Program Header对应的代码段,写入到PowerPC指令架构芯片仿真内存PowerPC-VMemory中,并根据代码段的大小,维护整个仿真内存结构

[0086] 如果代码段不大于256(一个MemoryPage的大小),创建一个MemoryPage,并将代码段写到对用的MemoryPage数组中去;

[0087] 如果代码段大于256(一个MemoryPage的大小),创建多个MemoryPage,并将代码段按顺序写到对应的MemoryPage数组中去,并维护各个节点的MemoryPageBeginAddress、MemoryPageEndAddress等信息,保证链表结构信息的正确性。

[0088] S3、PowerPC指令架构寄存器仿真

[0089] S31、对PowerPC指令架构e300、e500、e600芯片共计197个寄存器进行仿真,分为32个GPR通用寄存器、32个FPR浮点处理寄存器、51个特殊寄存器、41个e600专用特殊寄存器、41个e500专用特殊寄存器,寄存器共计197个,本发明通过数组PowerPC-Register模拟寄存器,通过数组操作模拟寄存器读写操作,实现寄存器的模拟,主要模拟结果如下:

[0090]
```
PowerPC-Register {
        /*e300、e500、e600 特殊寄存器*/
        Unsigned int   GPR [ 32 ]; //32 个通用寄存器
        Unsigned int   FPR [ 32 ]; //32 个浮点处理寄存器
        Unsigned int   cr;
        Unsigned int   fpcsr;
```

[0091]

```
Unsigned int   xer;
Unsigned int   xer_ca;
Unsigned int   lr;
Unsigned int   ctr;
Unsigned int   msr;
Unsigned int   pvr;
Unsigned int   pc;
Unsigned int   npc;
Unsigned int   phys_pc;
Unsigned int   ibatu[4];
Unsigned int   ibati[4];
Unsigned int   ibat_bl17[4];
Unsigned int   dbatu[4];
Unsigned int   dbatl[4];
Unsigned int   dbat_bl17[4];
Unsigned int   sdr1;
Unsigned int   sr[16];
/*e600 特殊寄存器*/
Unsigned int   e600_ibatu[4];
Unsigned int   e600_ibati[4];
Unsigned int   e600_dbatu[4];
Unsigned int   e600_dbatl[4];
Unsigned int   e600_pte[2];
Unsigned int   e600_tlbmiss;
Unsigned int   e600_ictc[2];
Unsigned int   e600_hid[2];
Unsigned int   e600_upmc[6];
Unsigned int   e600_usiar;
Unsigned int   e600_ummcr[3];
Unsigned int   e600_sprg[4];
Unsigned int   e600_ictrl;
Unsigned int   e600_mmcr2;
Unsigned int   e600_bamr;
/*e500 特殊寄存器*/
Unsigned int   e500_ibatu[4];
Unsigned int   e500_ibati[4];
Unsigned int   e500_dbatu[4];
Unsigned int   e500_dbatl[4];
Unsigned int   e500_pte[2];
Unsigned int   e500_tlbmiss;
Unsigned int   e500_ictc[2];
Unsigned int   e500_hid[2];
Unsigned int   e500_upmc[6];
Unsigned int   e500_usiar;
Unsigned int   e500_ummcr[3];
Unsigned int   e500_sprg[4];
```

```
        Unsigned int   e500_ictrl;
        Unsigned int   e500_mmcr2;
        Unsigned int   e500_bamr;
```

[0092]

```
    }
```

[0093] S32、通过访问PowerPC-Register，对相应的数组中的元素进行操作模拟PowerPC指令架构的寄存器操作，实现对PowerPC指令架构的寄存器仿真。

[0094] S4、PowerPC指令架构指令描述

[0095] S41、PowerPC指令架构e300、e500、e600处理器的所有425条指令，每条指令有一个唯一的表示，每条指令分为不同的段；为准确描述指令结构，本发明采用将指令分段的方式进行描述，获取指令Instruction的操作码opcode信息，并设计指令操作码信息数组PowerPC-InstructionDesp的结构包括：name、contentNumber和content。描述结构如下：

```
PowerPC-InstructionDesp {
        Const char *name;
        Int contentNumber;
        Int content[21]
    }
```

[0096]

[0097] 其中，name是指令名称，contentNumber指令分段数量，content用于存储opcode指令译码信息。

[0098] 以addx指令为例，对指令结构进行描述，其中ABS指令信息如图2所示(来自指令集参考手册)

[0099] Name为“addx”；该指令opcode(指令中固定不变的bit位)分为两段，第一段为bit0-bit5位，第二段为bit22-bit30位，contentNumber为“2”；content[21]存储opcode信息，content[21]为“0、5、0x1f、22、30、0x10A”。三个数字表示一个段，(2、11、0x05)表示bit0-bit5位，值为0x1f；(22、30、0x10A)表示bit22-bit30位，值为0x10A。

[0100] S42、构建指令操作码信息数组PowerPC-InstructionDesp[425]，对POWERPC架构指令集架构的425条指令进行描述，并存储到指令操作码信息数组PowerPC-InstructionDesp[425]中。

[0101] S5、PowerPC指令架构读取指令

[0102] S51、根据程序计数器PC(Program Count)的值，从S2维护的PowerPC-VMemory中读取指令；

[0103] S52、根据PowerPC-MemoryPage中的MemoryPageBeginAddress、MemoryPageEndAddress的值，计算出PC属于PowerPC-MemoryPage，记为PowerPC-MemoryPage-i，地址偏移为PC-MemoryPageBeginAddress，指令内容为PowerPC-MemoryPage-i对应的MemoryPage[PC-MemoryPageBeginAddress]，记为PC-Instruction。

[0104] S6、PowerPC指令架构指令译码

[0105] S61、根据S5读取的PC-Instruction，逐个计算PC-Instruction与PowerPC-InstructionDesp[425]的对应bit位是否相同，若PC-Instruction与PowerPC-InstructionDesp[425]中第k个指令描述对应的位置相同，则该指令记为Instruction_k。

[0106] S7、PowerPC指令架构指令翻译

[0107] S71、对PowerPC指令架构指令集中的指令进行功能翻译，并设计在虚拟仿真处理器运行的对每条指令进行操作模拟的函数，保证每一条指令和函数处理后，硬件处理器和虚拟仿真处理器的内存、寄存器保持一致；

[0108] S72：翻译PowerPC指令架构的425条指令，并将函数指针存储到指令翻译译码数组PowerPC-InstructionInterp[425]数组中，存储顺序与PowerPC-InstructionDesp[425]的指令顺序保持一致；

[0109] S73、根据S6计算出的Instruction_k，可获得该指令对应的函数中指针；

[0110] S8、PowerPC指令架构Linux系统调用实现

[0111] S81、定义Linux系统318种系统调用，并对系统调用的名称按照Linux系统调用的顺序进行标号，定义如下：

[0112] #define SYSCALL_restart_system 0

[0113] ……

[0114] #define SYSCALL_dup2 63

[0115] ……

[0116] #define SYSCALL_inotify_init1 318

[0117] S82、解析系统调用，并根据系统调用的类型，对系统调用进行相应的处理，并对相应的寄存器进行置位操作。

[0118] S9、PowerPC指令架构指令执行

[0119] S91、构建PowerPC指令架构的仿真框架，根据指令数量，进行循环取指、译码、指令翻译，并根据函数指针，执行指令操作；

[0120] S92、连续仿真，直到完成所有指令执行，并处理系统调用，实现针对PowerPC指令架构的处理器指令集仿真。

[0121] 本发明提出的方案，能够实现PowerPC指令架构处理器的指令集仿真，仿真精度高；本发明采用数组、链表等实现寄存器、内存的仿真，能够完成仿真PowerPC指令架构处理器。

[0122] 以上所述仅是本发明的优选实施方式，应当指出，对于本技术领域的普通技术人员来说，在不脱离本发明技术原理的前提下，还可以做出若干改进和变形，这些改进和变形也应视为本发明的保护范围。

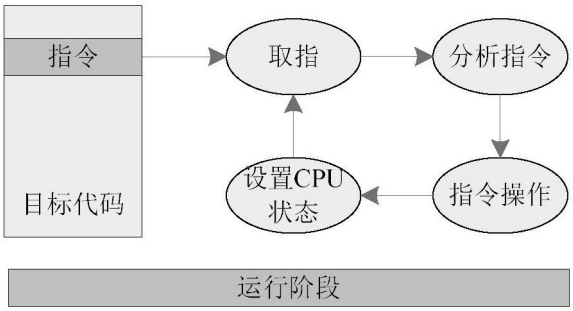

指令
目标代码
取指
分析指令
设置CPU状态
指令操作
运行阶段

图1

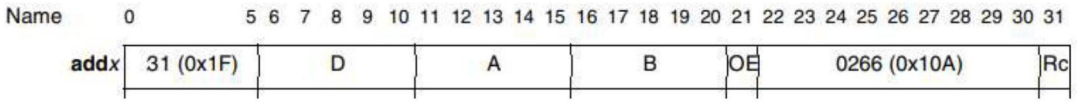

Name
0 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31
addx
31 (0x1F)
D
A
B
OE
0266 (0x10A)
Rc

图2