

SkyEye 平台 LEON2 VxWorks 仿真研究

赵 愚, 席 隆

(中国科学院 空间应用工程与技术中心, 北京 100094)

摘 要: 在分析 SkyEye 逻辑结构和 VxWorks 操作系统启动原理的基础上, 探索了 LEON2 模拟环境下 VxWorks 仿真的基本方法, 完善了 SkyEye 对 BIN 文件的装载兼容和 LEON2 模拟器的部分指令, 并对现有问题进行了分析, 为完整操作系统仿真提供了基础。

关键词: SkyEye; LEON2; 模拟器; VxWorks; 系统仿真

中图分类号: TP316.2

文献标识码: A

文章编号: 1000-7180(2017)05-0063-04

Research of LEON2 VxWorks Simulation on SkyEye Platform

ZHAO Yu, XI Long

(Technology and Engineering Center for Space Utilization, Chinese Academy of Sciences, Beijing 100094, China)

Abstract: Based on analysis of logical architecture of SkyEye and thesis of VxWorks operating system booting, this paper researched the basic approach of simulating VxWorks in LEON2 simulation environment, improved the compatibility of SkyEye for BIN files, added some instructions for LEON2 simulator, analyzed the existing problems of the LEON2 simulator on SkyEye, and provided the foundation for the whole operating system simulation.

Key words: SkyEye; LEON2; simulator; VxWorks; system simulation

1 引言

体系结构模拟器, 是指利用软件开发技术, 在一台计算机设备上仿真另一台具有不同指令集或体系结构的设备运行环境的软件, 它使用软件模拟计算机硬件的活动, 提供硬件接口供其他体系结构上的操作系统或软件调用运行, 模拟器的出现使得跨架构平台软件运行成为可能。

在模拟器上运行软件有诸多优点: 低成本、高透明度、硬件可配置等, 因此, 体系结构模拟器越来越普遍地被应用到硬件开发和软件开发生命周期中。

SkyEye 是一款开源的体系结构仿真平台, 由清华大学研发, 它支持的指令集架构有 x86、ARM、SPARC、MIPS 等, 并提供了对同指令架构多款处理器的模拟。

LEON2 是 Gaisler Research 公司于 2003 年研制完成的一款 32 位、符合 IEEE-1754(SPARC V8)

结构的处理器 IP 核^[1]。LEON2 具有高可靠性、可配置性、高性价比等特点, 常用于航天部门的产品研发。SkyEye 对 LEON2 处理器的模拟建立在两个基础上: 一是对通用 SPARC V8 指令集的模拟; 二是对 LEON2 外设的模拟。

目前在 SkyEye 平台的 LEON2 模拟器上可以运行经过特殊编译的二进制程序, 本文将探索在该模拟器上仿真运行 VxWorks 操作系统的方法。

2 SkyEye 结构及 VxWorks 映像

2.1 SkyEye 的结构分析

SkyEye 仿真平台可以分成两个部分: 核心库和外围动态模块。核心库实现了仿真平台与用户的交互部分, 并完成对外围动态模块的管理功能。外围动态模块包括处理器仿真模块、外设仿真模块、统计分析模块。

核心库主要位于源码目录下 common 目录, 该目录下有配置文件解析、断点管理、总线管理、处理

器管理等多个模块. 核心库是多种类处理器仿真所公用的部分.

源代码目录下的 arch 目录包含了所有 SkyEye 支持处理器的仿真模块. 由于 SkyEye 的软件结构很复杂, 下面介绍与操作系统仿真相关的目录及文件.

2.1.1 SkyEye 配置文件

使用 SkyEye 运行独立的程序或者操作系统需要建立一个文件夹, 文件夹内包含一个配置文件和可执行文件集合.

该配置文件默认命名为 skyeye.conf, 描述了运行程序所需的环境. 如目标架构、目标处理器、处理器寻址空间划分、终端显示方式等.

2.1.2 SkyEye 仿真平台的入口

utils/main/skyeye.c: 处理命令行输入指令, 判断指令是否正确, 判断文件映像格式是否正确, 根据指令以及 skyeye.conf 生成结构体保存相应配置信息; 执行 SIM_init() 函数.

2.1.3 控制台指令处理

common/ctrl/sim_ctrl.c: SIM_init() 函数位于该模块, 该函数中注册了多个控制台指令: start、stepi、run、list 等.

其中 start 命令通过已经生成的配置信息结构体初始化一个处理器内核, 并将可执行文件装载到内存并指定 pc 值. 执行 start 命令后, 仿真的处理器处于待运行状态.

run 命令将使仿真处理器从 pc 值指向的位置开始执行指令, 完成可执行程序的运行. stepi 属于单步调试命令, 其后跟数字表示执行指令数目.

2.1.4 可执行文件装载模块

common/loader/loader_elf.c: load_exec() 函数主要完成 ELF 可执行文件的分析和装载. 该函数通过 bfd 工具分析 ELF 文件头, 获取 text、data、bss 段的起始位置和大小, 分步装载到内存相应位置.

2.1.5 LEON2 处理器模块

该模块位于 arch/sparc 目录, 包含 common、instructions、mach、shell 四个目录.

common 目录内包含对 SPARC V8 通用架构的模拟, 如寄存器读写函数、内存读写函数、处理器单步执行函数、trap 处理等.

instructions 目录描述了所有模拟器支持的指令, 共 74 条, 没有完全实现 V8 指令集所有内容, 因此实现操作系统仿真的过程中需要完善指令集. 该目录中每个指令使用一个 c 文件描述, 每条指令对

应一个结构体. 这些结构体在 common 目录中 iu.c 文件中注册, 在单步执行函数中会利用字符匹配的方式确定正在执行指令的类型.

mach 目录是外设模拟部分, 模拟了三个基本外设 timer、uart、中断处理器. SkyEye 处理外设接口的方式是将外设接口注册到核心库的外设读写函数中, 因此多个处理器的对外封装完全一样, 具有良好的可扩展性和可移植性.

2.2 VxWorks 映像分类

VxWorks 映像可以分为两类: 一种是可加载的 VxWorks 映像; 一种是可引导的 VxWorks 映像^[2]. 可引导映像包括不驻留 ROM 的映像和驻留 ROM 的映像两种类型^[3].

可加载型映像存放于开发机上, 它的启动依赖于启动引导程序 BootRom, 常用于系统开发的调试阶段. BootRom 程序在生成后被下载到开发板 ROM, 开发板上电时程序会自动执行. BootRom 主要完成可加载型 VxWorks 映像的从开发机到开发板 RAM 的搬移工作, 具体搬移方式可以通过串口、网络、软盘等.

基于 ROM 的映像是 BootRom 程序和可加载型 VxWorks 映像的合并产物, 一般开发完成品可以使用这种格式的映像. 该映像存储于开发板 ROM, 上电时同样会运行搬移程序, 将 VxWorks 操作系统代码从 ROM 搬移到 RAM, 再从操作系统入口位置执行.

ROM 驻留型映像和基于 ROM 的映像基本相同, 也是存储在开发板 ROM 中, 不同的是, 该映像只会将 VxWorks 映像的数据段和 BSS 段搬移到 RAM 空间, 而代码段的执行仍然在 ROM 中. 因此, 与前者相比, 它具有更快的引导速度, 但运行速度会下降.

3 VxWorks 引导程序仿真

3.1 BootRom 引导程序

3.1.1 BootRom 生成

BootRom 的生成依赖于 Tornado BSP 文件夹内的 romInit.s、bootInit.c、sysALib.s、sysLib.c、bootConfig.c 以及设备驱动程序.

在 Tornado 集成环境中, 选择 Build 菜单—> Build Boot ROM, 选择目标板 BSP 生成对应的 BootRom 引导程序.

在命令行编译对应的 bin 文件: C:\Tornado\host\x86-win32\bin\objcopysparc -O binary

bootrom bootrom. bin,编译汇编代码:C:\Tornado\host\x86-win32\bin\objdumpsparc -D bootrom >bootrom. dis. dis 文件是 bin 文件的汇编描述,对 VxWorks 操作系统仿真研究有重要作用。

3. 1. 2 BootRom 启动原理

BootRom 启动时执行的第一段代码为汇编代码,即 romInit. s,这段代码完成寄存器初始化、关中断、禁止 Cache 以及初始化内存等功能,最后调用 romStart()函数。

romStart()位于 romStart. c,该函数主要作用是将引导程序的代码段和数据段搬移到内存中 RAM_HIGH_ADRS 位置并跳转到该位置执行,LEON2 BSP 定义的 RAM_HIGH_ADRS 为 0x40400000。

引导程序随后安装异常向量、设置 Cache 并进行内核初始化。引导程序最终从指定位置(软驱、硬盘、串口或网络等)下载 VxWorks 映像,将其装载到 RAM_LOW_ADRS(0x40200000)位置并跳转,操作系统开始接管硬件。

由于 BootRom 启动过程中与 VxWorks 硬件初始化动作基本相同,并且加载了部分 VxWorks 内核,所以选择 BootRom 的仿真作为 VxWorks 仿真的基础。

3. 2 SkyEye 引导模块设计

由于 SkyEye 默认引导 ELF 程序,而 Tornado 集成环境生成的 BootRom 及 VxWorks 映像并非 ELF 文件,在通过批处理指令获得 BIN 格式的映像文件后,需要对 SkyEye 引导模块重新设计。

utils/main/skyeye. c;在该文件中,需要对 ELF 判读部分进行修改。删除对映像文件是否为 ELF 文件的判断部分,并指定处理器模式为大端模式: endian = Big_endian。

common/ctrl/sim_ctrl. c;在 SIM_start()函数中,需要通过读取 ELF 文件的头部获取映像的入口地址。对于 BIN 文件的加载,需要指定地址。如果要仿真运行可加载型 VxWorks 映像,地址应指定为 RAM_LOW_ADRS(0x40200000);如果仿真运行 BootRom 程序,地址指定为 0 即可。

common/loader/loader_elf. c:重写 load_exec()函数,将 BIN 文件直接加载到指定的地址位置:

```
tmp_ker = fopen(file, "r");
fseek(tmp_ker, 0, SEEK_END);
size = ftell(tmp_ker);
printf("File length: %d bytes.\n", size);
```

```
tmp_str = (char *)skyeye_mm (size);
fseek(tmp_ker, 0, SEEK_SET);
fread(tmp_str, size, sizeof(char), tmp_ker);
```

```
if(tmp_str)
printf("Memory allocated.\n");
write_phys(addr, tmp_str, size);
```

至此 SkyEye 能够正确装载并运行 BIN 文件。

3. 3 SkyEye 指令模块设计

SkyEye 的指令模块包含了 74 条指令,部分实现了 SPARC V8 指令集。为了实现 VxWorks 操作系统仿真运行,需要逐步完善指令集模块。

目前研究中发现一条需要补充的指令:LDFSR。该指令动作是:从内存中取四个字节数据放入浮点状态寄存器即 FSR。

SkyEye 的指令模块设计需要遵循一定的模式。

3. 3. 1 指令设计

在 instructions 文件夹内加入指令描述文件:i_ldfsr. c。指令描述文件中包含一个结构体,该结构体就是对指令定义以及动作的描述。

```
sparc_instruction_t i_ldfsr = {
execute,
disassemble,
LDFSR_CODE_MASK,
PRIVILEGE,
OP,
};
```

LDFS 指令结构如图 1 所示。

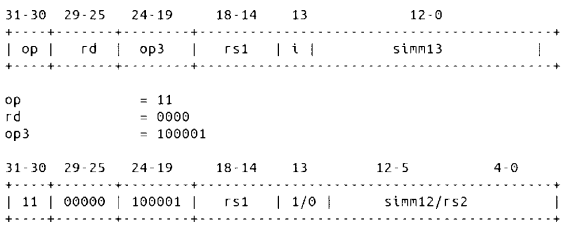


图 1 LDFS 指令结构

OP 表示该指令的 op 段内容,即指令高 32、31 的内容;PRIVILEGE 为特权等级;LDFSR_CODE_MASK 为该指令特征掩码,用于判别指令是否为该类型。

disassemble 为函数入口,该函数完成对指令类型以及分支判别。函数对当前指令进行操作,取出 op 和 op3 段内容,再结合特征掩码判断当前指令是否为 LDFSR。指令的第 13 位是分支标识 i,i 的取值可以是 0 或 1,分别代表两条分支:指令的 0—12 位表示寄存器或是内存地址。

execute 也是函数入口,该函数根据分支类型完成指令动作. LDFSR 指令通过 PSR 寄存器的 EF 段判断 FPU 模块是否有效,若无效则陷入 trap,若有效则继续执行. 根据 disassemble 函数得到的分支结果,execute 函数通过两种方式生成内存地址 addr: 寄存器 rs1 和 rs2 的值相加; 寄存器 rs1 和 simml2 内存偏移值相加. 然后取出内存 addr 地址位处的 4 个字节数据并存入 32 位的 FSR 寄存器. 最后完成 PC 和 NPC 寄存器赋值:

```
PCREG = NPCREG;
NPCREG += 4;
```

3.3.2 指令注册及编译

位于 arch/sparc/common 文件夹中的 iu. c 文件中注册了所有的指令结构体,因此新添加的指令应当在文件中注册.

在文件头部声明外部结构体变量:extern sparc_instruction_t i_ldfsr,iu_isa_register() 函数中加入语句:iu_i_register(&i_ldfsr).

将新加入的指令加入自动编译文件 Makefile. am,该文件位于 arch/sparc/common,在 sparc_insn 变量中加入 instructions/i_ldfsr. c.

3.4 SkyEye 配置文件编写

skyeye. conf 文件定义了模拟设备的存储地址划分,表 1 为 LEON2 的地址空间. 根据此表设计 skyeye. conf 内容如下:

```
arch:sparc
mach:leon2
mem_bank: map=M, type=R, addr=0x00000000,
size=0x20000000
mem_bank: map=I, type=RW, addr=0x20000000,
size=0x20000000
mem_bank: map=M, type=RW, addr=0x40000000,
size=0x40000000
mem_bank: map=I, type=RW, addr=0x80000000,
size=0x80000000
uart:mod=term
```

表 1 LEON2 地址空间分布

地址范围	映射含义
0x00000000—0x1FFFFFFF	PROM
0x20000000—0x3FFFFFFF	I/O
0x40000000—0x7FFFFFFF	RAM
0x80000000—0xFFFFFFFF	Internal(AHB)

文件中 map 的值有两类,值为 M 表示存储地址,I 表示外设地址. type 表示读写属性,R 表示只

读,W 表示只写,RW 表示可读可写.

3.5 重新编译

在 SkyEye 的源码文件夹下执行. /autogen. sh 和 automake 命令,确保 arch/sparc/common 文件夹下 Makefile. am 文件能够正确生成 Makefile. in 文件;再执行. /configure 通过 Makefile. in 生成最终的 Makefile 文件,这样才能保证新添加的指令被编译识别;最后执行 make &&. make install 重新编译安装被改变的文件. 重新编译后运行界面如图 2 所示.

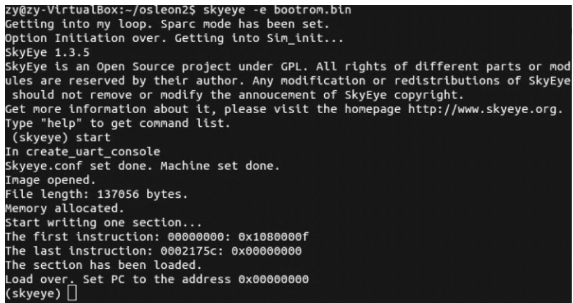


图 2 重编译后运行界面

3.6 模拟器运行测试及问题分析

在 bootrom. bin 以及 skyeye. conf 文件夹内输入 skyeye - e bootrom. bin 启动 SkyEye 仿真平台,输入 start 指令装载映像并初始化处理器核,再输入 run 即可运行映像.

目前映像运行仍然存在一定问题,经过调试分析发现两处需要改进的地方.

arch/sparc/common/traps. c 中 trigger() 函数是 trap 处理函数,在 ET 值为零的时候该函数触发会导致模拟器停止运行,因为 ET=0 表示处理器禁止中断. 解决这个问题可以利用 bootrom. dis 汇编文件以及 SkyEye 单步调试命令定位出错位置,出现这个问题的可能原因是 SkyEye 内部架构设计缺陷导致操作系统指令分支出现错误,从而在处理器禁中断时出现 trap.

arch/sparc/mach/mach_leon2_io. c: 该文件统一描述了 LEON2 外设动作:读、写、中断、重置. handle_irq() 函数用于处理外设中断,但是这个函数目前的设计较为简单,只能同时处理一个中断,在后续设计中,可以通过队列实现多中断处理和优先级触发机制.

4 结束语

本文对 LEON2 模拟平台 VxWorks 的仿真运 (下转第 72 页)

法,提高了前导数字纠错单元的运算性能和工作效率.这样就可以提升整个浮点运算单元的规格化移位的效率,并且可对浮点加法器的运算速度改进起到很大的改进作用.

参考文献:

- [1] Javier D Bruguera, Tomas Lang. Leading-one prediction with concurrent position correction[J]. IEEE Transactions Computers, 1999, 48(10):1083-1097.
- [2] Suzuki H, Morinaka H, Makino H, et al. Leading-zero anticipatory logic for high-speed floating point addition[J]. IEEE Journal of Solid State Circuits, 1996, 31(8): 1157-1164.
- [3] Kershaw R, Bays L, Freyman R, et al. A programmable digital signal processor with 32-bit floating-point arithmetic[C]//IEEE Solid Circuits Conference, Digest of Papers, IEEE, 1985.
- [4] Hays W, Kershaw R, Bays L, et al. A 32-bit VLSI

digital signal processor[J]. IEEE Journal of Solid State Circuits, 1985, SC-20(5):998-1004.

- [5] Zhang Ge, Qi Zichu, Hu Weiwu. Anovel design of leading zero anticipation circuit with parallel error detection[C]//IEEE International Symposium on Circuits and Systems. IEEE, 2005.
- [6] Yao Tao, Gao Deyuan. A novel concurrent error detection circuit for leading zero anticipator[C]// Proc. of 2nd International Conference on Computer Engineering and Technology (ICCET), 2010. IEEE, 2010.

作者简介:

魏思捷 女, (1992-), 硕士研究生. 研究方向为计算机智能控制、电子设计自动化.

富坤(通讯作者) 女, (1979-), 博士, 副教授. 研究方向为可重构计算、计算机系统结构.

E-mail: fukun@hebut.edu.cn.

(上接第 66 页)

行进行了初步探索,分析了 SkyEye 的内部架构和 VxWorks 启动原理,并选择 BootRom 仿真作为系统仿真的第一步,BootRom 的汇编代码是对 VxWorks 启动过程的表述,也是逐步完成研究的重要部分.通过改写 SkyEye 的装载模块,实现了对 BIN 文件的装载兼容;以 LDFSR 指令的设计为例介绍了通用指令设计方式.

本文旨在为完整的 VxWorks 操作系统的仿真运行提供研究基础.

参考文献:

- [1] 王宇舟,王二超. Leon2 微处理器 IP 核原理及应用[J]. 单片机与嵌入式系统应用, 2005, 6(8): 41-45.
- [2] 王雷,樊晓桢,王党辉. 龙芯 3A 平台 Vxworks 移植的研究和实现[J]. 微电子学与计算机, 2012, 29(2): 86-90.
- [3] 张忠,樊留群. VxWorks 在 S3C2410 上的 BSP 设计[J]. 微型电脑应用, 2005, 21(10): 22-25.

作者简介:

赵愚 男, (1992-), 硕士研究生. 研究方向为计算机应用技术. E-mail: zhaoyu1504@qq.com.

席隆 男, (1965-), 正高级工程师. 研究方向为空间综合电子学.