

面向 RISC 体系结构的 Linux 系统移植原理

王亮亮^{1,3}, 杨 琴², 芮 雪²

- (1. 新疆教育学院 现代教育技术中心, 新疆 乌鲁木齐 830043;
2. 新疆教育学院 信息科学与技术学院, 新疆 乌鲁木齐 830043;
3. 新疆教育学院 远程教育技术重点实验室, 新疆 乌鲁木齐 830043)

摘 要:移动互联网时代嵌入式处理器得到了快速发展与广泛应用。由于主流嵌入式处理器区别于传统 X86 架构处理器,需要对操作系统和应用软件进行移植。文中总结了 CISC 与 RISC 体系结构的 X86、ARM、MIPS 等主流处理器体系架构;介绍了我国龙芯处理器体系架构与各型号的发展现状及特点;对 Linux 操作系统跨处理器体系架构的移植原理进行了初步研究,介绍了交叉编译概念和原理,交叉编译工具集 gcc 编译器、binutils 连接器、glibc 运行库的创建原理,交叉编译工具 CLFS,本地编译的两种方式。

关键词:RISC;龙芯;Linux;系统移植

中图分类号:TP316

文献标识码:A

文章编号:1673-629X(2014)08-0076-04

doi:10.3969/j.issn.1673-629X.2014.08.018

Principle of Transplantation of Linux Operating System for RISC Architecture

WANG Liang-liang^{1,3}, YANG Qin², RUI Xue²

- (1. Modern Education Technology Center, Xinjiang Education Institute, Urumqi 830043, China;
2. College of Information Science and Technology, Xinjiang Education Institute, Urumqi 830043, China;
3. Key Laboratory of Distance Education Technology, Xinjiang Education Institute, Urumqi 830043, China)

Abstract: The embedded processor has developed rapidly and applied widely in mobile Internet times. The operating systems and software applications need to be transplanted because the mainstream embedded processor is different from traditional X86 processors. Not only the mainstream processor architecture based on the CISC and RISC, such as X86, ARM, MIPS, but also the development status and characteristics of the domestic Loongson processor architecture and each models are summarized in this paper. Meanwhile, the concept and principle of cross-compiling, the creation principle of cross-compiler toolsets which include gcc compiler, binutils connector and glibc runtime, cross-compiler tool CLFS, two ways of native compilation are also introduced after the principle of transplantation of Linux operating system which is cross-architecture is studied preliminarily.

Key words: RISC; Loongson; Linux; system transplantation

0 引言

近几年随着移动互联网的快速发展,嵌入式处理器得到了广泛的应用,人们对处理器的认识已不仅仅只有 Intel 和 AMD。在选择设备时非常关心处理器的品牌、主频、内核数量以及 RAM、ROM、操作系统等参数,在日常使用时对苹果 MAC 类 Unix 操作系统和基于 Linux 内核的 Android 操作系统、文件系统存储结构、管理方式等方面都有了初步的了解和认识,可以说

对全民进行了普及,整体提升了这方面的知识,改善了 Windows 一统天下的局面。

主流嵌入式处理器区别于传统 X86 架构,需要对操作系统和应用软件进行移植。目前 Linux 操作系统所支持的包括 X86、ARM、MIPS、MIPS64、Sun SPARC、Power PC、Alpha、IA64 等主要的体系结构^[1-2]。文中阐述 Linux 操作系统向 RISC 体系结构移植的原理,目的是能够快速地了解基础理论知识。

收稿日期:2013-10-08

修回日期:2014-01-14

网络出版时间:2014-04-24

基金项目:新疆维吾尔自治区自然科学基金项目(2011211B29)

作者简介:王亮亮(1977-),男,副教授,CCF 会员,研究方向为 Linux 操作系统。

网络出版地址:<http://www.cnki.net/kcms/detail/61.1450.TP.20140424.0839.086.html>

1 主流处理器

依据设计制造处理器的指令集特点可以分为两大体系结构:CISC(Complex Instruction Set Computer,复杂指令集计算机)和 RISC(Reduced Instruction Set Computer,精简指令集计算机)^[3-4]。现今三大主流处理器架构其中之一是以 Intel 和 AMD 为代表的 X86,另外一个现在是手机、平板处理器所使用的 ARM,最后一个是 RISC 处理器的典型代表 MIPS。

1.1 X86

早期的 CPU 全部是 CISC 体系结构,为了提高运行效率加入了越来越多的复杂指令,从而逐步形成复杂指令集计算机体系结构。Intel 公司 X86 架构 CPU 是典型的 CISC 体系结构,从最初的 8086 到后来的 Pentium 系列,每出一代新的 CPU,都会有自己新的指令,而为了兼容以前的 CPU 平台上的软件,旧的 CPU 指令集又必须保留,这就使指令的解码系统越来越复杂。

日益庞大的指令系统不仅使计算机研制周期变长,而且还有难以调试、难以维护等一些自身无法克服的缺点。大量的统计数字表明,大概有 20% 的简单的指令被反复使用,使用量约占整个程序的 80%;而 80% 左右的指令则很少使用,使用量约占整个程序的 20%^[5-6]。因此人们开始研制精简指令系统计算机(RISC),于是 RISC 技术在高端服务器和工作站上更是得到了广泛的应用。

1.2 ARM

随着智能手机与平板的快速发展,一个鲜为大众所知的平台逐渐地呈现在世人的面前。这就是 ARM(Advanced RISC Machines,高级精简指令集制造)处理器,基于 32 位精简指令集(RISC)处理器体系结构,ARM7、ARM9、ARM9E 和 ARM10E 为 4 个通用处理器系列,每一个系列提供一套相对独特的性能来满足不同应用领域的需求^[7]。ARM 技术广泛应用在嵌入式领域,涉及手机、数字机顶盒以及汽车智能系统和网络路由器等。当今,全球 90% 以上的手机以及超过四分之一的电子设备都在使用 ARM 技术^[8]。

ARM 是英国的一家微处理器企业,总部位于英国剑桥,专门从事基于 RISC 技术芯片设计开发。ARM 本身不直接从事芯片生产,通过出售芯片技术授权,建立起新型微处理器设计、生产和销售模式。全世界各大半导体公司都购买了 ARM 公司的授权,其中包括 Intel、IBM、三星半导体、NEC、SONY、飞利浦这样的大公司,目前高通、苹果和 NVIDIA 这三家公司是 ARM 体系中应用范围较广的。这些半导体生产商根据各自不同的应用领域,加入适当的外围电路,从而形成自己的 ARM 微处理器芯片进入市场。

1.3 MIPS

MIPS 是美国著名的芯片设计公司,总部位于美国加州,是一家采用精简指令集(RISC)设计制造高性能、高档次及嵌入式 32 位和 64 位处理器的厂商。在通用方面 MIPS R 系列微处理器用于高性能工作站、服务器和超级计算机系统。在嵌入式方面,MIPS K 系列微处理器是目前仅次于 ARM 的用得最多的处理器之一,其应用领域覆盖游戏机、路由器、打印机、掌上电脑等各个方面。

MIPS 的意思是“无内部互锁流水级的微处理器”(Microprocessor without Interlocked Piped Stages),在设计理念上强调软硬件协同提高性能,具有设计更简单、设计周期更短等优点。核心机制是尽量利用软件办法避免流水线中的数据相关问题^[9]。MIPS 的系统结构及设计理念比较先进,其指令系统经过通用处理器指令体系 MIPS I、MIPS II、MIPS III、MIPS IV 到 MIPS V,嵌入式指令体系 MIPS16、MIPS32 到 MIPS64 的发展已经十分成熟^[10]。

1.4 VLIW

RISC 和 CISC 各有优势,但为了更进一步提升处理器并行处理能力,衍生了 VLIW(Very Long Instruction Word,超长指令集架构),由美国 Multiflow 和 Cydrome 公司于 20 世纪 80 年代设计的体系结构。是一种非常长的指令组合,把许多条指令连在一起,增加了运算的速度。超长指令字(VLIW)是指令级并行,超线程(Hyper-Threading)是线程级并行,而多内核则是芯片级并行,这三种方式都是提高并行计算性能的有效途径^[11-12]。

VLIW 的最大优点是简化了处理器的架构,删除了处理器内部许多复杂的控制电路,降低了芯片制造成本且功耗低。例如 IA-64 每时钟周期可运行 20 条指令,而 CISC 通常只能运行 1-3 条指令,RISC 能运行 4 条指令,可见 VLIW 要比 CISC 和 RISC 强大得多。目前基于这种指令架构的微处理器主要有 Intel 的 IA-64 和 AMD 的 x86-64 两种。超长指令集 CPU 由于融合了 RISC 和 CISC 的优势,成为未来的 CPU 发展方向之一。

2 Loongson 与 MIPS

我国龙芯(Loongson)处理器采用的也是 MIPS 指令架构^[13],它与大家平常所知道的 X86 指令架构互不兼容,属于 RISC 体系。2001 年 5 月,中国科学院计算技术研究所成立龙芯课题组研制龙芯系列处理器。2006 年 3 月龙芯购买了 MIPS 的结构授权。2011 年曙光 5000A 使用 8 000 颗 AMD 的 CPU 达到了 200 万亿次,而 2012 年曙光 6000 将使用 8 000 颗龙芯 3 号达到

1 000 万亿次。

1) 龙芯 1 号是兼顾通用及嵌入式 CPU 特点的 32 位处理器,采用类 MIPS III 指令集,具有七级流水线、32 位整数单元和 64 位浮点单元,频率为 266 MHz。

2) 龙芯 2 号是 64 位处理器,采用先进的四发射超标量超流水结构,片内一级指令和数据高速缓存各 64 kB,二级高速缓存最多可达 8 MB,最高频率为 1 000 MHz,功耗为 3~5 W,远低于国外同类芯片,实测性能是 1.3 GHz 的威盛处理器的 2~3 倍,已达到中等 Pentium4 水平。龙芯 2 号是一个系列,其中 2E 和 2F 为典型代表。

3) 龙芯 3 号目前有 3A(四核)和 3B(八核)。龙芯 3A 集成了四个 64 位超标量处理器核、4 MB 的二级 Cache,频率为 900 MHz~1 GHz,指令系统与 MIPS 64 兼容并能通过指令扩展支持 X86 二进制翻译。

3 系统移植

为什么要移植?主要是目前绝大多数计算机、服务器等均采用 X86 架构,常用的操作系统和应用软件也是基于 X86 架构,无法直接运行在 ARM、MIPS、龙芯等硬件平台。移植方法主要有两种:交叉编译和本地编译。

3.1 交叉编译原理

Linux 操作系统由于其开源、免费、多硬件体系结构支持、高度可定制性等特点在服务器系统、嵌入式领域、操作系统研究等得到了快速发展与广泛应用。

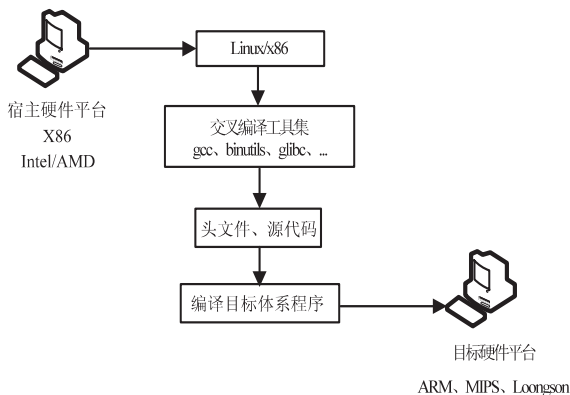


图 1 交叉编译流程图

交叉编译就是在一种体系结构的机器上编译出运行于另一种体系结构的程序,两种体系也可以是相同的,但通常是以不同的体系来说明。Linux 操作系统的交叉编译多是采用在传统 X86 架构的计算机上安装 32 或 64 位 Linux 操作系统,完善本地编译与开发环境,建立交叉编译工具集,编译目标体系程序,在目标硬件平台安装测试等环节实现系统移植,如图 1 所示。

3.2 交叉编译工具集

交叉编译工具主要由 gcc 编译器、binutils 连接器

和 glibc 运行库三个部分组成。它们既要运行于当前体系的系统下,又要能生成和处理目标体系的文件,因此称它们为交叉编译工具集。

1) gcc(gnu collect compiler)是一组编译工具的总称。主要完成的工作任务是“预处理”和“编译”,以及提供了与编译器紧密相关的运行库的支持,如 libgcc_s.so、libstdc++.so 等。

主要编译参数如下,其中环境变量有 $\$ \{CLFS\}$ 为编译根目录, $\$ \{CLFS_HOST\}$ 为宿主体系目录, $\$ \{CLFS_TARGET\}$ 为目标体系目录。

```

./configure --prefix = $ {CLFS}/cross-tools \
--host = $ {CLFS\_HOST} \
--target = $ {CLFS\_TARGET} \
--with-sysroot = $ {CLFS} \
--disable-nls \
--disable-shared \
--disable-multilib \
--enable-languages = c
  
```

如要编译出目标体系的 gcc 代码,除了需要 gcc 程序的源代码外还需要符合目标体系的头文件,其中最为主要的就是 kernel 的头文件和 glibc 的头文件,如图 2 所示。

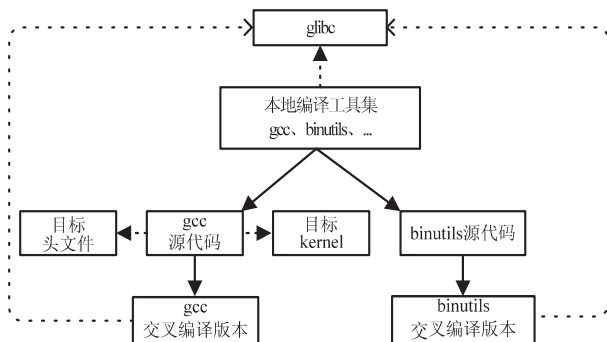


图 2 创建交叉编译工具集

目标体系的 glibc-headers 需要通过编译 glibc 源代码获得,但 glibc 需依赖 kernel 来识别目标体系的内核特征,故需先行生成目标体系的 kernel-headers,有关内核的交叉编译另行论述。

2) binutils 是一组开发工具,包括连接器、汇编器和其他用于目标文件和档案的工具。如汇编(as)、连接(ld)、静态库归档(ar)、反汇编(objdump)、elf 结构分析工具(readelf)、无效调试信息和符号的工具(strip)等。通常,binutils 与 gcc 是紧密相集成的,如果没有 gcc 是不能正常工作的。主要编译参数如下:

```

./configure --prefix = $ {CLFS}/cross-tools \
--host = $ {CLFS\_HOST} \
--target = $ {CLFS\_TARGET} \
--with-sysroot = $ {CLFS} \
--disable-nls \
--enable-shared \
  
```

```
--disable-multilib
```

编译交叉版本 binutils 需要宿主系统 (X86 系统) 的本地编译工具集 binutils、gcc、glibc 等以及 binutils 源代码来实现,如图 2 所示。

3) glibc 是 gnu 发布的 libc 库,即 c 运行库,是 Linux 操作系统中最底层的 api,几乎其他任何的运行库都会依赖于 glibc。主要编译参数如下:

```
BUILD_CC="gcc" \
CC=" ${CLFS_TARGET} -gcc" \
AR=" ${CLFS_TARGET} -ar" \
RANLIB=" ${CLFS_TARGET} -ranlib" \
./configure --prefix=/usr \
--libexecdir=/usr/lib/glibc \
--host= ${CLFS_TARGET} \
--build= ${CLFS_HOST} \
--enable-kernel=2.6.x \
--with-binutils= ${CLFS}/cross-tools/bin \
--with-headers= ${CLFS}/usr/include \
.....
```

上述参数中 BUILD_CC 用于指定在编译过程中用 gcc 程序进行编译,CC 用于调用 \${CLFS_TARGET} 环境变量指定交叉编译版本 (目标体系) 的 gcc 程序存储路径。AR 和 RANLIB 是 binutils 中的工具,作用是处理二进制文件,这里调用交叉编译后生成的目标体系版本。

--enable-kernel 指定内核版本

--with-binutils 指定目标体系的 binutils 存储路径

--with-headers 指定目标体系的头文件存储路径

3.3 交叉编译工具 CLFS

CLFS (Cross Linux From Scratch) 是支持交叉编译的工具集,支持 X86、X86_64、sparc、mips、powerpc 等多体系硬件平台^[14]。正式版本为 1.0,另有两个变种,其中 CLFS Sysroot 采用 sysroot 的方法来进行编译,优点是软件包编译次数可以减少,缺点是不能支持编译后对软件进行测试,必须要到目标平台上才能测试。CLFS Embedded 一般用于极小系统以及嵌入式系统,使用 uclibc 而不是一般 LFS 使用的 glibc。使用 CLFS 的主要优点如下:

1) 可以指导人们如何构造一个交叉编译器,比如你可以在 X86 的机器上构建出一套 ARM 或 MIPS 工具链,然后利用这套工具链完全从源代码级编译出按照定制要求的 Linux 发行版。

2) 能够充分了解 Linux 系统基础组件功能与系统工作机制,如何使用多库共存的系统,充分挖掘目标系统的潜力;对学习和开发操作系统具有重要作用。

3) 高度可定制性,可以生成一个紧凑的系统;可以提供额外的安全性,可以在编译阶段便把某个安全

代码补丁加入到系统中。

3.4 本地编译

一般在系统上能够直接进行编译叫本地编译,也就是在目标软硬件平台进行编译,编译得到的程序也是在本地执行,如图 3 所示。相对于交叉编译,本地编译是一种更简单、快捷的编译方法,但有时会受限于当前系统组件功能制约的情况,如 kernel、gcc、glibc 等版本较低等。

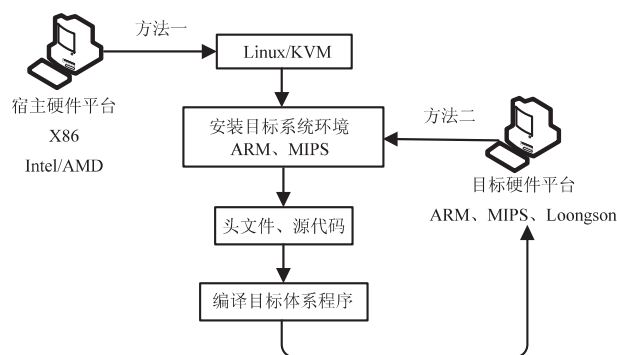


图 3 本地编译流程图

常见的本地编译方式有两种:一种是通过 Linux KVM 内核虚拟机安装目标操作系统 (ARM、MIPS),如图 3 中的方式一;另一种是目标硬件平台已有相关软件系统,如图 3 中的方式二。本地编译也需要完善的编译工具集,一般情况下系统能提供基本的编译环境,但有时编译目标体系程序还需要安装依赖开发包或相关组件,这时需要先对依赖需求进行本地编译,然后安装与完善编译环境,编译过程中可以根据具体需要进行定制、优化、集成。

4 结束语

文中总结了目前主流处理器的发展现状及特点,并对 Linux 操作系统跨体系结构的移植原理进行了初步研究,分析了交叉编译与本地编译的概念和流程。目前,我国龙芯处理器也需通过移植 Linux 操作系统及应用软件完善对其支撑,掌握移植技术对国产核心硬件平台和国产基础软件系统的发展应用具有积极意义。

参考文献:

- [1] 朱 嘉. 基于 MIPS32 平台的 Linux 操作系统移植[J]. 单片机与嵌入式系统应用, 2006(11): 24-26.
- [2] Linux arch 目录下处理器体系架构介绍[EB/OL]. 2011. <http://blog.csdn.net/xdw1985829/article/details/6863804>.
- [3] 庞春江, 孟建良, 王晓华. 计算机体系结构的现状及其发展趋势[J]. 计算机工程与应用, 2002, 38(5): 105-107.
- [4] 施志林. RISC、CISC 技术比较与研究[J]. 中国科技信息,

从图 3 的结果可以看出,聚类数在演化周期上的变化为: $K=4$ 时,持续了 10 步; $K=3$ 时,持续了 189 步(由于演化周期较长,省略了后面的演化过程),所以理想的聚类结果是 3 类,而离群点 O_1 被定义为弱离群点, O_2 被定义为强离群点。

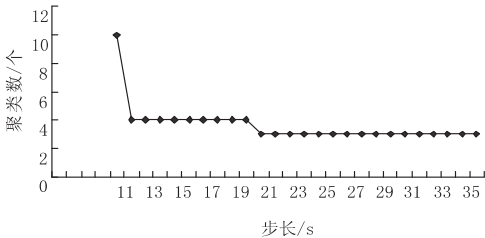


图 3 聚类数及演化周期

实验数据表明,基于元胞自动机模型的凝固聚类算法能够用于离群点检测与分析。它可以通过稳定聚类数的演化周期发现离群点,并通过离群点到簇的平均距离将其划分为强离群点和弱离群点,并在元胞演化的过程中得到各个层次上的聚类结果为离群点的属性分析提供参考。

4 结束语

离群点的挖掘和分析是数据挖掘的重要研究内容之一,而发现离群点并分析离群点的属性能够获得进一步的扩展知识,为决策支持提供依据。通过将凝固层次聚类算法应用于离群点检测与分析可以看出,凝固层次聚类能够在发现离群点的过程中同时完成聚类过程,具有并行计算特性。根据凝固聚类过程中的元胞周期演化过程可以将离群点分成强离群点和弱离群点,为数据分析提供了方便。通过对元胞自动机邻居的定义,可以将算法有效地扩展到高维空间,为离群点分析算法的改进提供新的思路。

参考文献:

[1] Han Jiawei, Altman R B, Kumar V, et al. Emerging scientific applications in data mining [J]. Communications of the
2008(11):131-132.
[5] 孟开元. 计算机体系结构的发展与变化[J]. 福建电脑, 2005(7):18-19.
[6] 王艳芳. RISC 技术发展研究[J]. 科技情报开发与经济, 2005, 15(13):237-238.
[7] 沈建华. ARM 处理器与嵌入式系统[J]. 单片机与嵌入式系统应用, 2010(11):5-7.
[8] ARM 官网公司简介[EB/OL]. 2013. [http://www. arm. com/zh/about/compan y-profile/index. php](http://www.arm.com/zh/about/compan y-profile/index. php).
[9] 范学英, 张明新, 王登磊. 嵌入式系统概述[J]. 自动化技术与应用, 2008, 27(2):113-115.

ACM, 2002, 45(8):54-58.
[2] Knorr E M, Ng R T. Algorithms for mining distance-based outliers in large datasets[C]//Proceedings of the 24th international conference on very large data bases. New York: Morgan Kaufmann, 1998:392-403.
[3] Knorr E M, Ng R T. Finding intensional knowledge of distance-based outliers[C]//Proceedings of the 25th international conference on very large data bases. Edinburgh, Scotland: Morgan Kaufmann, 1999:211-222.
[4] 邓玉洁, 朱庆生. 基于聚类的离群点分析方法[J]. 计算机应用研究, 2012, 29(3):865-868.
[5] 陈蓉, 李艳萍. 一种基于离群点的聚类迭代检测算法[J]. 科学技术与工程, 2012, 20(35):9725-9729.
[6] 李旭辉, 郑丽英, 徐项, 等. 一种基于高维空间聚类的离群数据发现算法[J]. 微电子学与计算机, 2007, 24(12):44-46.
[7] 魏黎, 宫学庆, 钱卫宁, 等. 高维空间中的离群点发现[J]. 软件学报, 2002, 13(2):280-290.
[8] 薛安荣, 姚林, 鞠时光, 等. 离群点挖掘方法综述[J]. 计算机科学, 2008, 35(11):13-18.
[9] 张廷, 高宝俊, 宣慧玉. 基于元胞自动机的创新扩散模型综述[J]. 系统工程, 2006, 24(12):6-15.
[10] 王安麟, 姜涛. 基于进化元胞自动机的结构拓扑优化[J]. 机械工程学报, 2005, 41(2):1-5.
[11] Zheng Bojin, Li Yuanxiang. Function optimization algorithm based on cellular automata[J]. Computer Engineering, 2003, 29(19):66-67.
[12] 毛宽民, 陈吉红, 唐小琦, 等. 基于元胞自动机的曲线曲面迭代逼近算法[J]. 华中科技大学学报:自然科学版, 2001, 29(4):38-39.
[13] 黎夏, 刘小平. 基于案例推理的元胞自动机及大区域城市演变模拟[J]. 地理学报, 2007, 62(10):1097-1109.
[14] Ester M, Kriegel H, Sander J, et al. A density-based algorithm for discovering clusters in large spatial databases with noises[C]//Proceedings of the 2nd international conference on knowledge discovery and data mining. Portland, Oregon: AAAI Press, 1996:226-231.
[10] Sweetman D. MIPS 体系结构透视[M]. 屈建勤, 译. 第 2 版. 北京:机械工业出版社, 2008.
[11] 超长指令集架构[EB/OL]. 2012. <http://baike. baidu. com/view/2 72962. htm>.
[12] VLIW 架构服务器[EB/OL]. 2013. <http://baike. baidu. com/view/2 5595. htm>.
[13] 龙芯获 MIPS 指令系统授权[EB/OL]. 2009. http://www. loongson. cn/lo ongson/article/2009/0622/article_220. ht-ml.
[14] Cross Linux From Scratch (CLFS)[EB/OL]. 2013. <http://trac. cross-lfs. org/wiki/WikiStart>.

面向RISC体系结构的Linux系统移植原理


作者:

王亮亮, 杨琴, 芮雪, [WANG Liang-liang](#), [YANG Qin](#), [RUI Xue](#)

作者单位:

[王亮亮, WANG Liang-liang\(新疆教育学院 现代教育技术中心, 新疆 乌鲁木齐 830043; 新疆教育学院 远程教育技术重点实验室, 新疆 乌鲁木齐 830043\), 杨琴, 芮雪, \[YANG Qin\]\(#\), \[RUI Xue\\(新疆教育学院 信息科学与技术学院, 新疆 乌鲁木齐, 830043\\)\]\(#\)](#)

刊名:

[计算机技术与发展](#) 

英文刊名:

[Computer Technology and Development](#)

年, 卷(期):

2014(8)

本文链接: http://d.wanfangdata.com.cn/Periodical_wjz201408018.aspx