# 龙芯Linux内核移植和优化

**张福新**
**江苏中科龙梦科技有限公司**
**中科院计算所**

**2008.10.20**

# 内容提要

- 准备工作
- 移植步骤
- 性能优化案例

# 准备工作

- 打好基础

  学习**C**编程，大致了解**MIPS**汇编

  掌握操作系统的基本概念：中断，进程，内存管理等

  学习配置和编译内核
- **获得和熟悉龙芯硬件平台**
  - **阅读龙芯处理器手册**
  - **阅读相关平台的主板手册**
- **准备**编译环境

  交叉编译

  本地编译
- 取得内核源代码

# 准备工作——参考资源

- 龙芯相关：
  中科龙梦，www.lemote.com
  龙芯技术服务中心: www.loongson.cn/comapany
- **Linux/MIPS资源**
  www.linux-mips.org
  内核代码、工具链、模拟器和各类文档

# 参考平台

- 我们就以目前放**PPT**的这台龙芯笔记本的内核开发为例

# 移植步骤

1. 试通我们的开发环境
2. 添加代码
3. 编译第一个龙芯内核并用早期的printk输出内核信息
4. Kgdb
5. CPU支持
6. 主板相关支持
7. PCI子系统
8. 驱动程序
9. 调试，调试，调试，再调试

# 试通开发环境

- 我们采用串口来作为调试信息窗口
  本主板预留串口接口
  串口简单、容易获得，适合做调试界面
- 串口设置
  终端仿真程序：**minicom(Linux),hyper term(windows)**
  端口设置，波特率，奇偶校验，流控等
  设置：龙芯**PMON**缺省设置**115200 8 N1**
- 连接龙芯笔记本主板和调试主机
- 启动
  应该能够看到**PMON**启动信息

# 添加代码

- 代码位置

  需要添加的代码包括板级支持、驱动和一些与系统接口的添加信息。板极支持代码按照习惯应该放在**arch/mips**目录下，驱动应该在**driver/**目录下，接口代码主要在**include/asm-mips**和**arch/mips**下

  建立**arch/mips/lemote**目录

# 添加代码(续)

- **arch/mips/lemote**

  提供主板相关的中断分派，**PMON**接口，
  时钟，**IO**，内存等平台相关资源设置

  **Linux-2.6.23**中，需要实现的接口如下：
  - **PMON**相关：**prom_init, prom_putchar**
  - 中断相关：**plat_irq_dispatch,arch_init_irq**
  - 资源设置：**plat_mem_setup,plat_timer_setup,pcibios_init,set_io_port_base, get_system_type,_machine_restart,_machine_halt,pm_power_off, board_time_init, mips_rtc_get_time, _wbflush**等

# 添加代码(续）

- **和linux/mips内核集成**
  - **Include/asm-mips/｛cpu.h,module.h,chacheops.h,bootinfo.h｝：龙芯相关宏定义**
  - **include/asm-mips/mach-lemote/｛dma-coherency.h, mc146818rtc.h｝：由于平台特殊行，需要覆盖mach-generic的内容**
  - **Arch/mips/kernel/cpu-info.c：添加龙芯CPU处理代码**
  - **Arch/mips/kernel/proc.c：把机器组名字添加到mach_group_to_name数组**
  - **Arch/mips/kernel/Makefile: 根据config把龙芯CPU需要用的存储管理模块选上**
  - **Arch/mips/Makefile: 增加一节把新增的代码代码链进去**
  - **Arch/mips/Kconfig: 添加必要的配置选项**

# Makefile修改样例

- **#**
- **# lemote fulong mini-PC board**
- **#**
- **core-$(CONFIG_LEMOTE_FULONG) +=arch/mips/lemote/lm2e/**
- **load-$(CONFIG_LEMOTE_FULONG) +=0xffffffff80100000**
- **cflags-$(CONFIG_LEMOTE_FULONG) += -linclude/asm-mips/mach-lemote**

# Kconfig修改

- 加一个**CPU**节，一个平台节
- **config CPU_LOONGSON2**
- **bool "Loongson 2"**
- **depends on SYS_HAS_CPU_LOONGSON2**
- **select CPU_SUPPORTS_32BIT_KERNEL**
- **select CPU_SUPPORTS_64BIT_KERNEL**
- **select CPU_SUPPORTS_HIGHMEM**
- **help**
- **The Loongson 2F processor imple**

# Kconfig修改

- **config LEMOTE_FULONG**
- **bool "Lemote Fulong mini-PC"**
- **select ARCH_SPARSEMEM_ENABLE**
- **.....**
- **select CPU_HAS_WB**
- **select GENERIC_ISA_DMA_SUPPORT_BROKEN**
- **help**
- **Lemote Fulong mini-PC board based on the Chinese Loongson-2E CPU and a FPGA northbridge**

# 配置和编译内核

- 选择**LEMOTE_FULONGG**板，别选其它任何板
- 选择**LOONGSON2 CPU**
- 选择**'character devices'**下的串口设备和串口控制台支持，别选**virtual console**支持
- 在**'kernel hacking'**下选择**cross-compilation**
- 其它选项要么缺省，要么选择**no**
- 检查交叉编译工具路径和**Makefile**。**make dep; make**

# 早期的printk

- 终端设备初始化太迟
- 简单的串口输出**prom_printf**

# 串口驱动和控制台

- **Prom_printf不能满足要求(中断方式)**
- 添加串口支持的两种方式：

  静态定义 **include/asm/serial.h**
  - **2.6.23废止，可采用platform_device方式**

  运行时设置
- 串口参数(标准串口)：

  **Io_type:**
  - **SERIAL_IO_MEM memory mapped方式(writeb,readb)访问. Iomem_base + shifted offset**
  - **SERIAL_IO_PORT inb,outb访问.**

- **KGDB--内核调试**
- **支持代码： putDebugChar,getDebugChar**
- **使用:**

  **选择'kernel hacking'下的CONFIG_REMOTE_DEBUG**

  **准备串口连接**

  **主机: mipsel-linux-gdb vmlinux, target remote /dev/ttyS0**

  **目标机：运行，停于断点处**
- **注意事项**

  **__init函数影响断点精确**

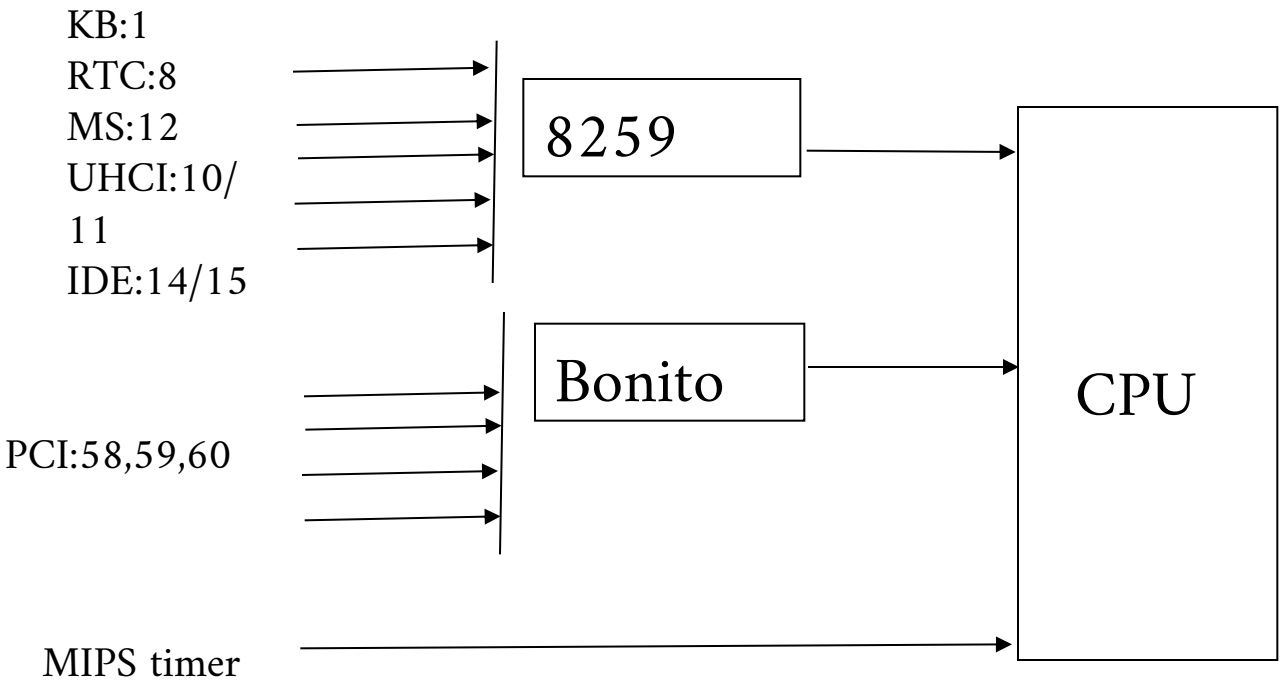
  **断点别设置在函数入口(防止为初始化的指针被kgdb引用)**

# CPU相关的支持代码

- 不支持指令的模拟或者替代
  早期龙芯的除法指令，非对齐访问指令模拟
- **cache和TLB管理**
  龙芯**cache**管理指令有特殊性

# 板级支持--中断

- 了解硬件：中断源，控制器，路由(连线)
- 列出方案：
  一个静态的中断路由图
  中断源列表
  相应的中断控制器
  中断控制器的级连方式


KB:1
RTC:8
MS:12
UHCI:10/
11
IDE:14/15
8259
Bonito
PCI:58,59,60
CPU
MIPS timer

# 中断概述

- 四部分代码完成中断服务：

  中断检测和分派：genex.S->plat_irq_dispatch,功能是确定那个中断源发生了中断，调用do_IRQ

  Do_IRQ: arch/mips/irq.c，通用框架，调用相应的中断控制器禁止/使能相应的中断，调用真正的驱动

  Hw_irq_handler：每个中断源对应一个这个结构，告诉do_IRQ如何处理某个中断. 如果你有一个新的中断控制器，那么就要写一段这个代码。

  驱动代码：做真正的事情

# 系统时间和时钟

- **rtc**用于保持日历时间(启动时读)
- 一个系统时钟(常常是**mips cpu**提供的**counter/compare**硬件时钟)用于计算**jiffies**,推进系统时间

# PCI子系统

- **Documentation/mips/pci/pci.README**
- **PCI总线的一些简单概括**
- 启动次序
- 驱动接口
- 板级相关的函数和变量

# PCI总线的一些简单概括

- **PCI总线有三个独立的地址空间: config ,IO,memory.**
- 每个**PCI**设备都能响应配置命令，可以响应**IO**和**/**或**memory**访问。
- 启动时，**BIOS**或者**OS**通过配置空间设置基址寄存器。基址寄存器决定了一个设备应该响应的**IO/mem**范围。一条总线上的设备基址寄存器范围不能重叠
- 多条**pci**总线可以通过**pci-pci**桥连接

# pci启动次序

- **Do_basic_setup调用pci_init(). drivers/pci/pci.c**
- **Pci_init()首先调用pcibios_init**
- **pcibios_init会调用pciauto_assign_resources做一个资源分配**
- **Pcibios_init调用pci_scan_bus**
- **Pcibios_init调用一些修正代码，调整资源分配**
- **从pcibios_init返回后，pci_init会做最后一次基于设备的修正**

# PCI驱动接口

- **Inb/outb/inw/outw/inl/outl: include/asm/io.h**用于访问**pci io**空间. **Mips_io_port_base**应该初始化为**pci io**空间的起始地址
- **Readb/writeb etc. include/asm/io.h** 驱动程序用它们来访问**pci mem**空间。
- **Pci_{read/write}_config_{byte,word,dword}: include/linux/pci.h**。驱动程序用它们来访问**pci**配置空间
- **Pci_map_single**等。用于把虚拟地址映射到总线地址，例如**DMA**时

# 板级函数和变量接口

- **Struct pci_ops my_pci_ops:** 提供配置空间读写例程
- **Mips_pci_channels[]：**用于给**PCIAUTO**的代码指定系统所有的**PCI**总线和地址范围
- **Pcibios_fixup_resources：**传给**pci_scan_bus,**发现新设备时调用
- **Pcibios_fixup: pci_scan_bus**结束时调用
- **Pcibios_fixup_irqs：pci_scan_bus**结束时调用，用于修正中断分配。
- **Pcibios_assign_all_busses**

# *调试*

- **Oops**
- 四个关：
  串口出字 → 基本硬件配置**ok**
  **cache**打开 → 存储管理初始化**ok**
  **bogoMips**计算通过(打开中断) → 中断通路至少部分正确。
  **Init** 启动 → 用户态**ok**
- 思考，通路法

# 代码整理和提交

- 编码风格
  **Kernel source Documentation dir**
- 命名、文件组织整理
- 生成**patch**
- 提交和修改交互
  **Documenatio/SubmittingPatches**

# 性能分析和改进

- 存储管理
  例：**pagesize/TLB**
- 网络性能

# TLB优化

- 通过增加页的大小来减少TLB失效的次数
- 通过软TLB减少TLB失效开销

在内存系统中建立一个硬件TLB的缓冲区

- 通过FAST_TLB_REFILL减少TLB失效开销

在64位操作系统内核中缓存第3级页表指针

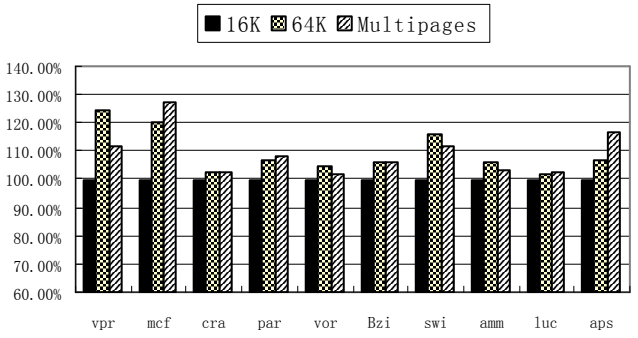
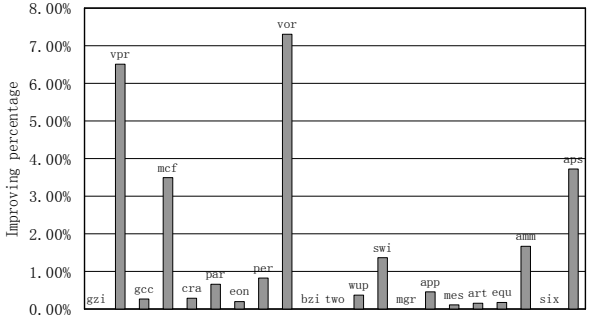
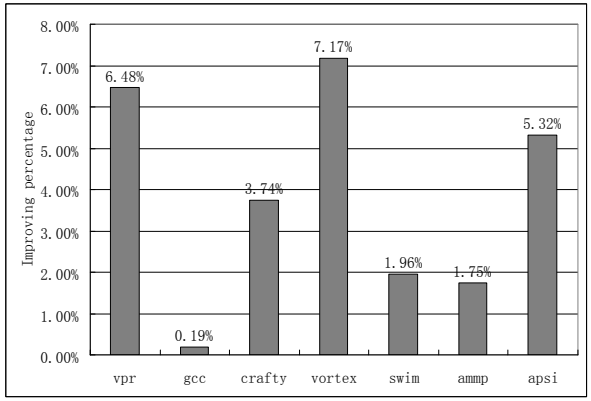



图4 多种页系统下部分SPEC测试程序性能比较

多种页系统下SPEC测试程序性能比较




软TLB对SPEC2000分数的提高率




**FAST_TLB_REFILL对SPEC2000分数的提高**

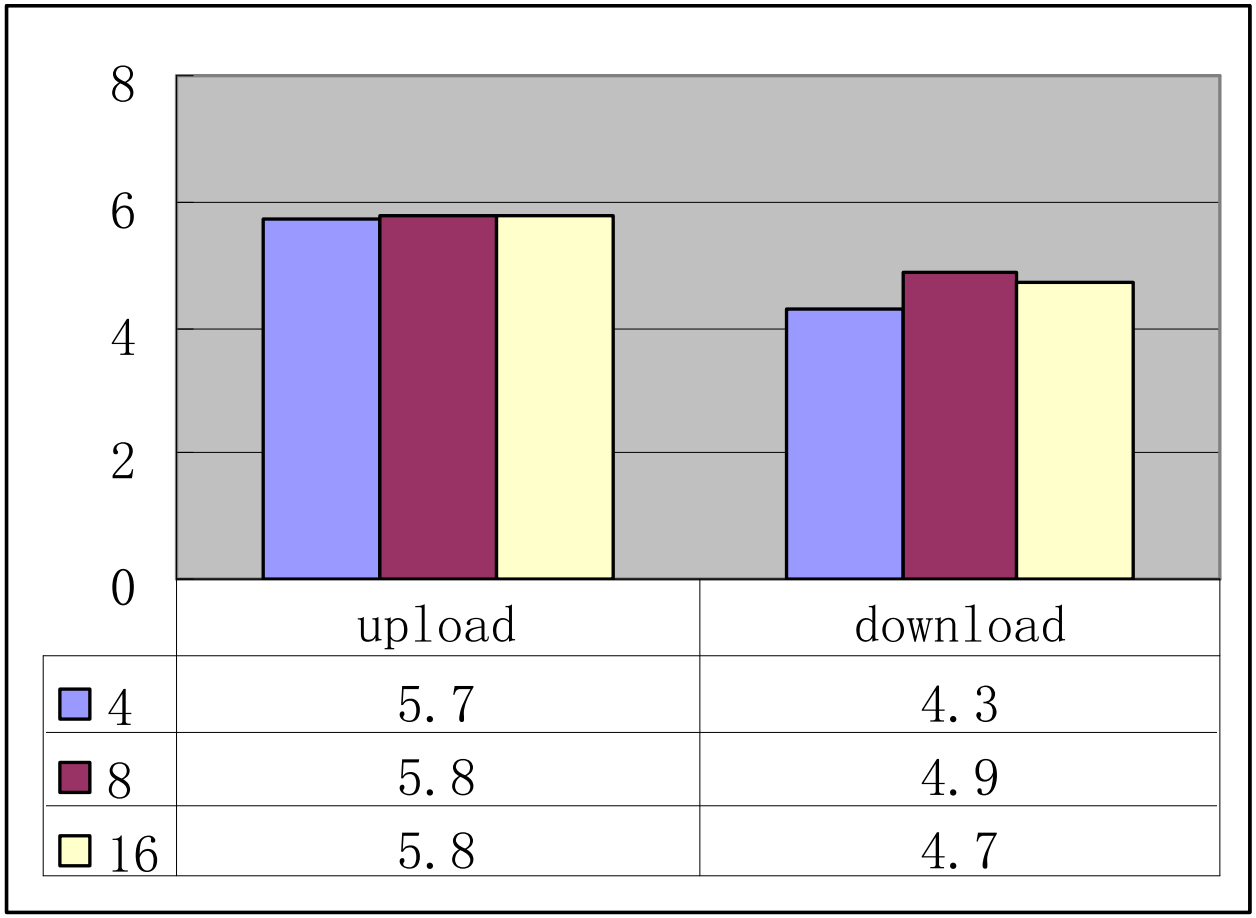
# Oprofile 统计ssh/scp

| scp文件下载 | 文件名 | 采样次数 | 所占比例 | 总共采样次数 |
|---|---|---|---|---|
| ssh | libcrypto.so | 723467 | 58.1% | 1244387 |
| | vmlinux | 428483 | 34.4% | |
| | libc | 69824 | 5.6% | |
| | ssh | 22176 | 1.8% | |
| scp | vmlinux | 113061 | 97.7% | 115754 |
| | libc | 1359 | 1.2% | |
| | scp | 1131 | 1.0% | |

# 内核时间分布

| | | | |
|---|---|---|---|
| vmlinux | src_unaligned_dst_aligned | 215851 | 17.35 |
| | both_aligned | 40941 | 3.29 |
| | rtl8139_poll | 24445 | 1.96 |
| | handle_IRQ_event | 17212 | 1.38 |
| | move_128bytes | 13525 | 1.07 |

# 优化memcpy效果：循环展开




| | upload | download |
|---|---|---|
| 4 | 5.7 | 4.3 |
| 8 | 5.8 | 4.9 |
| 16 | 5.8 | 4.7 |

# 抛砖引玉

- **敬请指教，谢谢！**