基于龙芯 2K1000 的 Loongnix 系统实时性优化方法研究*

俞冠中，韦　雄，田青旺，史旭明
（国核自仪系统工程有限公司，上海 200241）

摘　要： 在高可靠性要求的工业自动化应用中，Loongnix 系统作为一种分时操作系统，其实时性不能满足可靠性要求，需要使用实时操作系统。在分析研究 Loongnix 的 Linux 内核实时性缺陷和 RT-Preempt 实时性优化方法的基础上，提出一种基于 RT-Preempt-Linux 实时内核替换 Loongnix 系统原生 Linux 内核的方法，实现 Loongnix 实时性优化和实时化改造，用自设计测试软件和专用实时性工具 Cyclictest 对改造后的 Loongnix 系统进行验证测试。测试结果表明改造后的 Loongnix 系统的实时性能获得大幅提升，进程切换时间、线程切换时间以及任务延时都能达到微秒级，满足工业自动化应用的可靠性要求。

关键词： 龙芯；Loongnix；Linux；实时抢占补丁；实时系统；实时性
中图分类号： TP311；TP316.2　　**文献标识码：** A　　DOI：10.16157/j.issn.0258-7998.212461

中文引用格式： 俞冠中，韦雄，田青旺，等. 基于龙芯 2K1000 的 Loongnix 系统实时性优化方法研究[J].电子技术应用，2022，48(6)：98-102，111.
英文引用格式： Yu Guanzhong，Wei Xiong，Tian Qingwang，et al. Research on method of real-time performance optimization of Loongnix operation system based on Loongson 2K1000[J]. Application of Electronic Technique，2022，48(6)：98-102，111.

Research on method of real-time performance optimization of Loongnix operation system based on Loongson 2K1000

Yu Guanzhong，Wei Xiong，Tian Qingwang，Shi Xuming
(State Nuclear Power Automation System Engineering Corp，Shanghai 200241，China)

Abstract： Because LoongnixOS is the time-sharing operating system，it cannot meet the requirement of high reliability in industrial automation application which requires to use real-time operating system(RTOS). On the basis of analysis of and study on defects of real-time performance in Linux kernel of LoongnixOS and the method to optimize real-time performance in RT-Preemption patch，the method of real-time transformation of LoongnixOS and optimizing real-time performance of LoongnixOS is put forward，in which a RT-Preempt-Linux kernel is a substitute for original Linux kernel of LoongnixOS. Self-developing real-time performance softwares and Cyclictest are utilized to verify the method. Results of test show that the method can be effective in improvement of real-time performance of LoongnixOS. Real-time performance，including process-switching time，thread-switching time and task-switching delay，of the LoongnixOS transferred to RTOS achieves the grade of microsecond.
Key words： Loongson；Loongnix；Linux；RT-Preempt；real-time opration；real-time performance

0 引言

龙芯 2K1000 处理器[1-2]是一款面向工业自动化与工业控制应用场景的高性能低功耗通用处理器，基于 MIPS64 架构，采用 40 nm 制造工艺[3]，主频最高 1 GHz，功耗小于 5 W，支持 64 位 DDR2/3-1066 内存，提供 SPI、UART、I²S、I²C、USB2.0 等通用外设接口。

目前市场上，龙芯 2K1000 板卡一般预装 Loongnix 操作系统。Loongnix 操作系统是一种基于 Linux 内核的图形化界面操作系统。和 Linux 系统一样，Loongnix 系统也是分时系统[4]，不能满足对实时性要求较高的工业自动化场景(如电站控制[5-6])的要求。因此，需要针对 Linux 内核影响实时性能的因素进行实时性改造和优化。

目前，Linux 内核实时化的有效方法是在 Linux 内核源文件中加入实时补丁，再编译内核，生成 Linux 实时内核。Linux 内核实时补丁主要有三种：RED-Linux 补丁，Kurt-Linux 补丁以及实时抢占(RT-preempt)补丁[7-8]。RED-Linux 补丁是美国加州大学欧文分校(University of California Irvine，UCI)开发的一种软实时补丁[8]。Kurt-Linux 是美国堪萨斯大学(University of Kansas，KU)开发的一种 Linux 实时补丁，其内核同时运行实时任务和非实时任务

* 基金项目：国家科技重大专项(2019ZX06002035)

时，内核不能被抢占[8-9]。RT-Preempt 补丁是由 Ingo Molnar 和 Thomas Gleixner 开发和维护的一种完全可抢占式内核的硬实时补丁[10]，其实时性要明显优于前两种 Linux 实时补丁，且 RT-Preempt 是开源补丁，拥有强大社区支持[11]，支持 Linux 内核版本也比前两者丰富。综上所述，本文提出一种基于 RT-Preempt-Linux 实时内核替换 Loongnix 系统原生 Linux 内核的方法，实现 Loongnix 实时性优化和实时化改造。

1 Linux 内核实时性限制因素

1.1 非完全可抢占内核

Linux 进程切换机制依赖进程用户态和进程内核之间互相切换实现的[12]。进程需通过系统调用或中断触发来完成进程用户态到进程内核态的切换。进程切换时，内核使用自旋锁来确保数据的不冲突。进程进入临界区操作数据时，其他进程只能阻塞，那么任务的时间确定性就无法保证[13]。所以，标准 Linux 内核的临界区是不可抢占的。

1.2 中断延迟

Linux 中断响应处理分为顶半(top-halves)部和底半(buttom-halves)部[11]，也称为上半部和下半部。上半部属于硬中断，会关闭中断，屏蔽其他任何中断请求。在关闭中断时，系统外部事件无法得到响应，导致任务响应延迟。若标准 Linux 出现大量外部 IO 事件(如磁盘操作[11])，其他任务的延时时间会大大增加。

1.3 优先级反转

优先级反转(Priority Inversion)[13-14]是指高优先级的任务被低优先级的任务阻塞，反而中等优先级的任务先于高优先级的任务执行的现象。低优先级的进程 PL 首先执行，并占用了共用资源 Rsrc，此时高优先级进程 PH 开始执行，进程 PL 挂起。当进程 PH 尝试获取 Rsrc 时，因 Rsrc 被进程 PL 占据，进程 PH 也挂起，进程 PL 恢复运行。此时，不需要 Rsrc 的中等优先级进程 PM 就绪运行，进程 PL 挂起。进程 PM 执行结束后，进程 L 恢复运行，直至放弃 Rsrc，此时，高优先级进程 PH 才获得 CPU 使用权。中优先级进程 PM 先于高优先级进程 PH 获取 CPU 使用权，优先级发生反转。优先级反转对操作系统实时性危害很大，增加了任务调度时间的不确定性，严重时引起系统崩溃[8]。目前，标准 Linux 内核并无应对优先级反转的机制。

2 RT-Preempt 补丁实时原理

2.1 内核可完全抢占

RT-Preempt 补丁使用优先级可继承的互斥锁(rt_mutex)重新实现自旋锁来实现内核锁的可抢占性[13,15]。自旋锁 spin_lock()宏函数内用禁止迁移 migrate_disable()替代禁止抢占 preempt_disable()，使自旋锁可抢占。实时自旋锁 rt_spin_lock()替代原自旋锁 _raw_spin_lock()。rt_spin_lock()函数的实现中调用了 rt_spin_lock_fastlock()函数。rt_spin_lock_fastlock()函数中调用了 might_sleep()函数。might_sleep()函数的作用是允许当前进程进入睡眠状态，进程进入睡眠状态则该进程交出了 CPU 的使用权。那么，进程可以被抢占。需要指出的是内核中非线程化的中断不能被抢占，不能使用 rt_mutex[10]。

2.2 中断线程化处理

RT-Preempt 补丁的中断线程化处理是将中断服务程序转变为可被操作系统调度的线程断线程的优先级并不固定，用户按需设置其优先级，默认优先级为 50。

中断线程化处理包含硬中断的线程化处理和软中断的线程化处理两部分[15]。硬中断线程化在 __setup _irq()函数中实现，__setup_irq()函数调用 kthread_create()函数创建线程，采用先进先出调度策略(SCHED_FIFO)。软中断线程化在 spawn_ksoftirqd()中实现，其线程建立过程与硬中断线程化相同。需要指出的是：不是所有的中断都需要中断线程化处理，例如时钟中断应为最高优先级，不能中断线程化处理。struct irqaction 结构体中的 flag 成员变量用来设置是否需要中断线程化处理。

2.3 优先级继承策略

在高优先级任务 TH 等待低优先级任务 TL 占据共用资源 Rsrc 时，为了使低优先级任务尽快运行并释放 Rsrc，操作系统会将 TL 的优先级提高到和 TH 的优先级一样，直到 TL 释放 Rsrc。当 TL 的优先级继承了 TH 的优先级，中优先级任务 TM 就无法抢先 TH 获得 CPU 的使用权。因此，优先级反转就不会产生。RT-Preempt 补丁通过优先级可继承 rt_mutex 实现优先级继承策略。加入 RT-Preempt 补丁编译 Linux 后，rt_mutex 成为 Linux 核心(kernel)的组成部分。

2.4 高精度时钟

RT-Preempt 补丁的时钟系统不像依赖标准 Linux 一样依赖系统滴答中断计时，而是提供了一套新的时钟架构，可以提供纳秒级的精度。标准 Linux 系统为了提高时钟分辨率而升高系统滴答中断的频率情况不会在 RT-Preempt-Linux 出现，从而避免了系统符合变重性能降低的发生。

3 实时性能测试软件设计

3.1 进程切换时间统计软件设计

在一个时刻里，一个 CPU 只能执行一个任务。多个任务共享一个 CPU 需要依赖上下文切换(Context Switch)。上下文切换时间是指 CPU 从一个任务切换到另一个任务所需的时间开销。上下文切换时间决定了任务调度速度。因此，上下文切换时间是衡量操作系统实时性的关键指标[16]。Linux 支持多进程运行，所以，Linux 的上下文切换时间就是进程切换时间。

进程切换时间统计软件的设计思路是通过读管道(pipe)和写管道来实现父子进程之间的同步，其程序流程如图 1 所示。父进程获取当前时间，把当前时间写入管道，后读管道阻塞。子进程读管道获取父进程切换前的时间，再获取当前时间，计算进程切换时间。进程调度

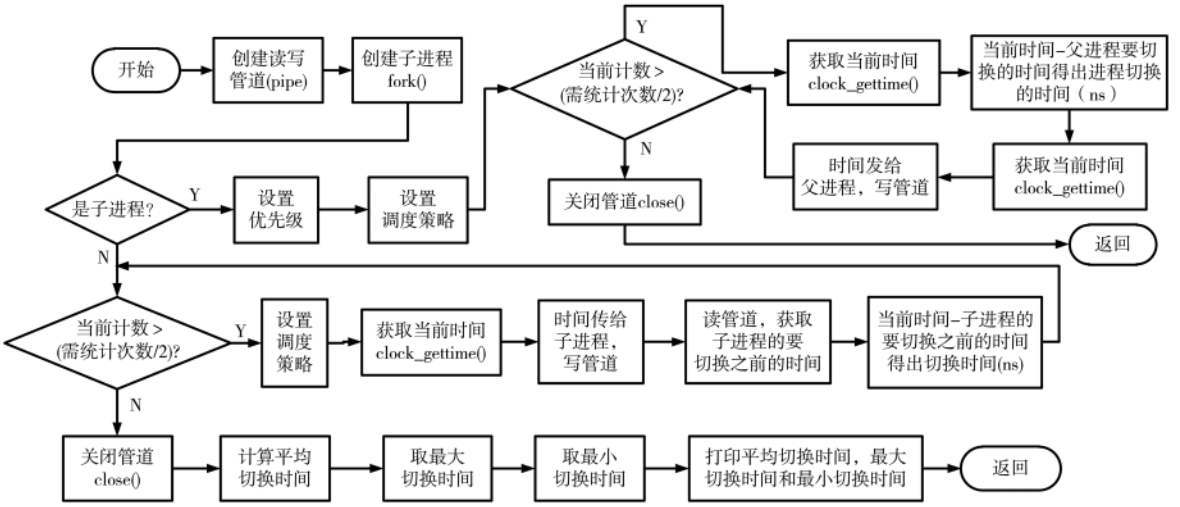

图 1　进程切换时间统计软件流程图

策略采用时间片轮转(SCHED_RR)，进程优先级设置为 99，切换统计次数设定为 1 000 次。每运行满 1 000 次，打印切换平均用时，打印切换最大用时和切换最小用时。

3.2 线程切换统计软件设计

线程是进程内共享进程资源的一个最小执行单元。线程切换时间大小体现了一个进程内的任务调度速度。因此，线程切换时间也是评判操作系统实时性能的重要标志。

线程切换时间统计软件由三个模块组成：主线程，线程 1 和线程 2。主线程内初始化功能所需的全局变量，如 timespec 结构体对象等，初始化信号量，创建线程 1 和线程 2，计算线程切换时间平均值，统计线程切换时间的最大值和最小值，并打印线程切换的平均用时、线程切换的最大用时和线程切换时间的最小用时。

线程 1 的逻辑设计图如图 2 所示，线程 1 先等待信号量 1，收到信号量 1 后获取当前时间，计算线程切换时间，再获取当前时间，最后发送信号量 2。信号量 2 发送后，线程 2 就被唤醒执行。在主线程中初始化信号量 1 时，其参数 Value 设置为 1，首先运行线程 1。

线程 2 的设计逻辑与线程 1 相同。线程切换时间统计软件通过两个信号量实现线程 1 和线程 2 的同步，其设计思想和进程切换时间统计软件相似。

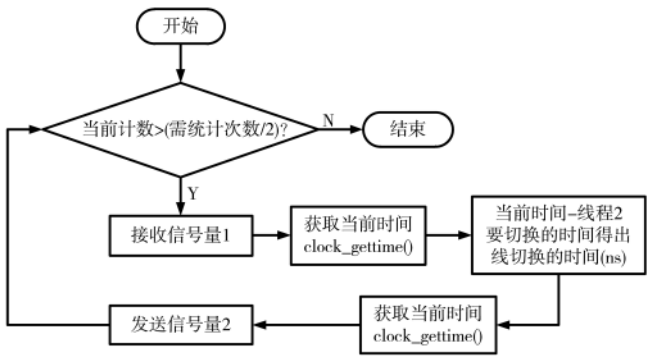

图 2　线程切换时间统计软件流程图

4 实时 Loongnix 实时性测试与分析

实时 Loongnix 系统是用 RT-Preempt-Linux 内核替代 Loongnix 系统原生标准 Linux 内核后的 Loongnix 系统。RT-Preempt-Linux 内核是在标准 Linux 内核源码上加入 RT-Preempt 补丁后编译生成的。对安装实时 Loongnix 系统的龙芯 2K1000 平台进行性能测试。测试分为用自设计软件测试进程切换时间和线程切换时间，以及用专用实时性能测试工具 Cyclictest 测试任务响应延时时间。对未替换实时内核的 Loongnix 系统进行相同的性能测试并进行比较研究。

4.1 测试环境

实时 Loongnix 系统和原生 Loongnix 系统性能测试的软硬件环境如表 1 所示。龙芯 2K1000 处理器的工作主频设置为 800 MHz。由于电站控制项目要求的应用软件需要在 Linux 4.0 版本以上才能运行，因此，实时 Loongnix 系统选用的 Linux 4.19 内核，并在其基础上加入对应版本的 RT-Preempt 补丁。

表 1　测试软硬件环境

	原生 Loongnix	实时 Loongnix
处理器	龙芯 2K1000	龙芯 2K1000
处理器工作主频/MHz	800	800
Loongnix 版本	1.0	1.0
Linux 内核版本	3.10	4.19
RT-Preempt 补丁版本	-	4.19

4.2 进程切换时间测试

进程切换时间统计软件的进程优先级设置为 99，统计次数设定为 1 000 次。在龙芯 2K1000 平台上的实时 Loongnix 系统运行 20 次获取总共 2 万次进程切换时间的统计数据。在龙芯 2K1000 平台上的原生 Loongnix 系统进行相同的测试。测试结果见表 2。

表 2　实时 Loongnix 与原生 Loongnix 进程切换时间对比

被测对象	线程切换最大时间/μs	线程切换平均时间/μs	线程切换最小时间/μs
实时 Loongnix	151.61	13.82	3.89
原生 Loongnix	4 168.12	13.38	5.26

实时 Loongnix 进程切换时间为微秒级，而未使用 RT-Preempt-Linux 内核的原生 Loongnix 进程切换时间达到 2.51 ms。实时 Loongnix 系统可以满足电站控制应用的系统任务切换时间不大于 1 ms 的性能需求。

对实时 Loongnix(实线)与原生 Loongnix(虚线)进程最大切换时间每千次切换统计一次的对比如图 3 所示。实时 Loongnix 系统每千次切换的最大切换时间连线比较平滑，而原生 Loongnix 系统每千次切换的最大切换时间连线抖动幅度比较大。因此，RT-Preempt-Linux 内核对 Loongnix 系统的上下文切换时间确定性提高明显。

4.3 线程切换时间测试

线程切换时间统计软件的统计次数设定为 1 000 次。在龙芯 2K1000 平台上的实时 Loongnix 系统运行 20 次获取总共 2 万次线程切换时间的统计数据。在龙芯 2K1000 平台上的原生 Loongnix 系统进行相同的测试。测试结果见表 3。

实时 Loongnix 线程平均切换时间和未使用 RT-Preempt-Linux 内核的原生 Loongnix 线程切平均切换时间接近，但其线程切换最大用时也是微秒级的。原生 loongnix 的最大线程切换用时要超过 4 ms。

表 3　实时 Loongnix 与原生 Loongnix 线切换时间对比

被测对象	进程切换最大时间/μs	进程切换平均时间/μs	进程切换最小时间/μs
实时 Loongnix	385.85	25.45	12.13
原生 Loongnix	2 511.59	17.6	7.98

对实时 Loongnix(实线)与原生 Loongnix(虚线)线程最大切换时间每千次切换统计一次的对比如图 4 所示。实时 Loongnix 系统每千次切换的最大切换时间连线比较平滑且都在 100 μs 左右，而原生 Loongnix 系统每千次切换的最大切换时间连线抖动幅度大。因此，RT-Preempt-Linux 内核对 Loongnix 系统的线程切换时间确定性提高明显。

4.4 Cyclictest 工具测试

Cyclictest 是一种开源的专业 Linux 实时性能测试工具软件，可以精确地测量任务唤醒延时。Cyclictest 测量线程线程时间唤醒的时间间隔，这个实际的时间间隔与线程睡眠设定时间的差就是任务唤醒时间的延时。这个延时由定时器中断延时和线程调度延时组成。中断响应时间和保存上下文的时间决定了中断延时的大小。本次实验使用 Cyclictest 1.0。

4.4.1 单线程测试

Cyclictest 测试指令为：sudo ./cyclictest -l100000 -m -t1 -n -p90 -i200 -h2000 -q，其中，-l(loops)为循环个数，本次测试设定为 100 000，缺省为 0；-m(mlockall)为锁定当前和未来的内存分配；-t[NUM] (threads=NUM)为

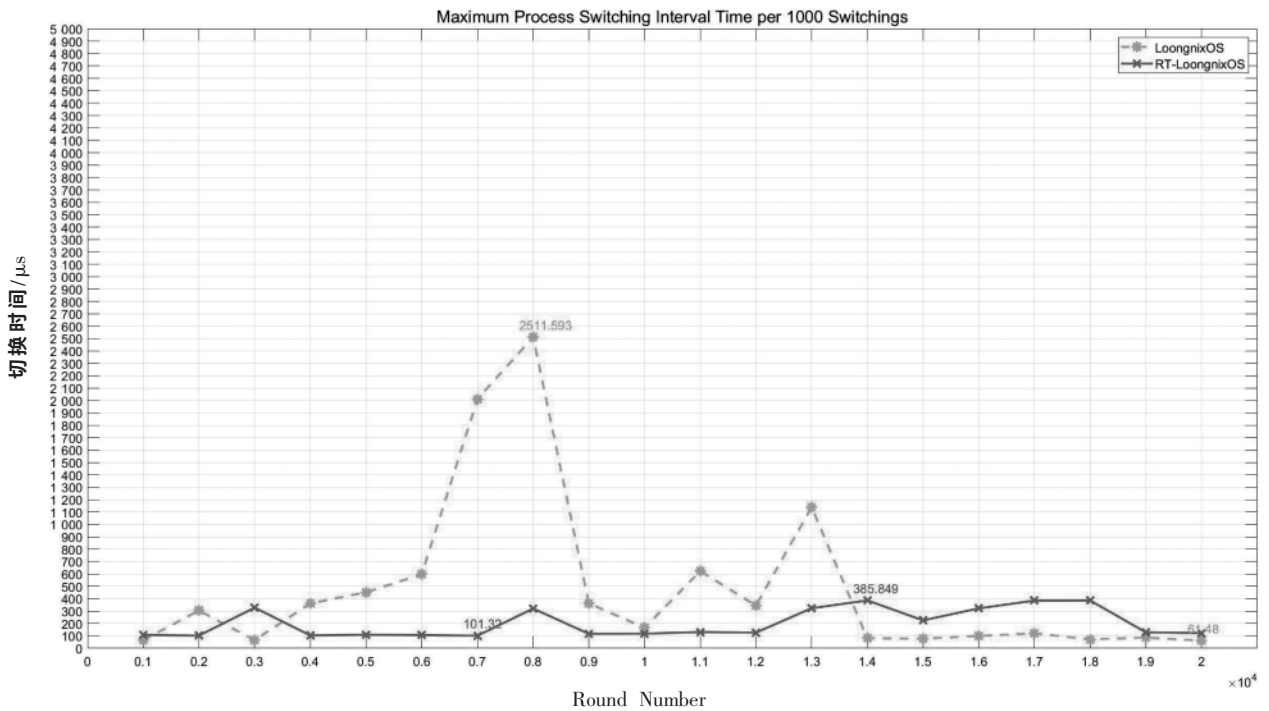

图 3　实时 Loongnix 与原生 Loongnix 最大进程切换时间对比

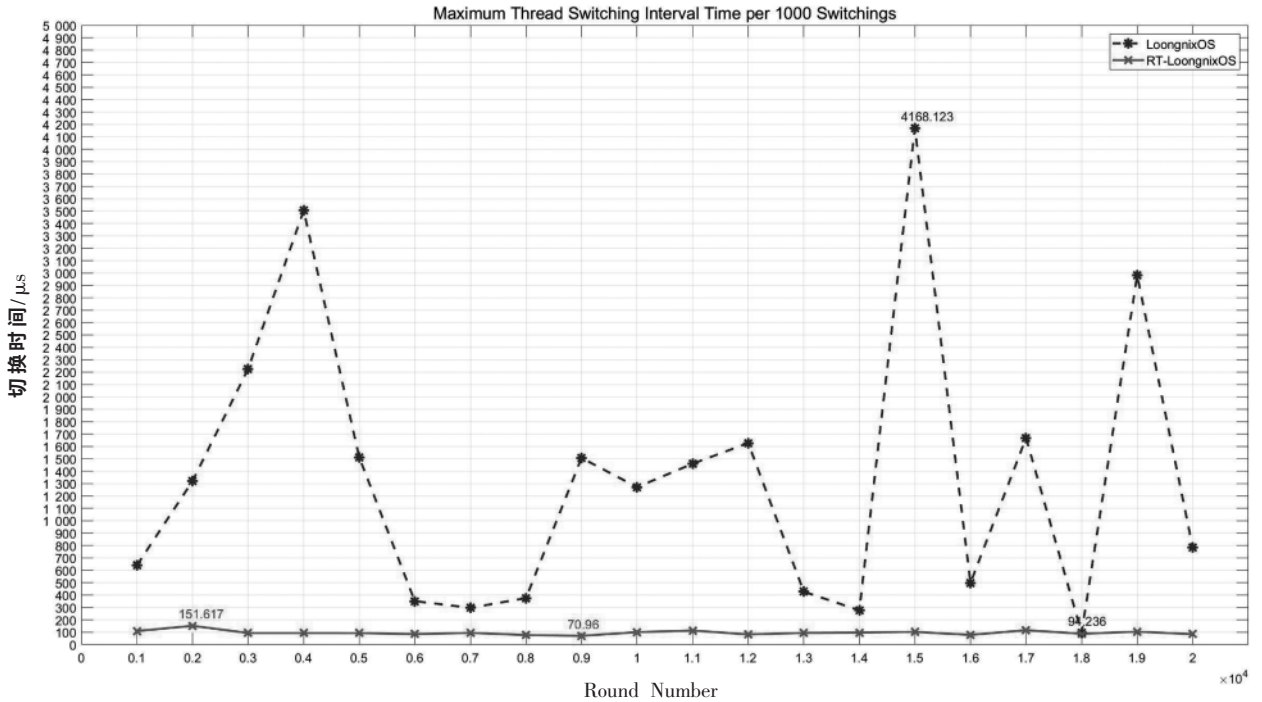

图 4 实时 Loongnix 与原生 Loongnix 最大线程切换时间对比

启动线程个数，本次测试为单线程测试，故设定为 1；-n (nanosleep)使用精度为纳秒的睡眠时间设置；-p(prio)为线程设置的优先级，本次实验优先级设置为 90；-i(interval)为线程的时间间隔，本次实验设置为 200 μs，缺省为 1 000 μs；-h(histogram)为记录延时时间，本次实验跟踪 2 000 μs 以内的延时；-q(quiet)为退出前打印结果。实时 Loongnix 的测试结果如图 5 所示，原生 Loongnix 的测试结果如图 6 所示。

```
# Total: 000100000
# Min Latencies: 00014
# Avg Latencies: 00021
# Max Latencies: 00044
# Histogram Overflows: 00000
# Histogram Overflow at cycle number:
# Thread 0:
```

图 5 实时 Loongnix 系统 Cyclictest 单线程测试结果

```
# Total: 000099996
# Min Latencies: 00011
# Avg Latencies: 00021
# Max Latencies: 04908
# Histogram Overflows: 00004
# Histogram Overflow at cycle number:
# Thread 0: 40003 40024 90690 90737
```

图 6 原生 Loongnix 系统 Cyclictest 单线程测试结果

RT-Preempt-Linux 内核替换后成为实时系统的 loongnix 系统单线程最大延迟时间微秒级。未实时化改造的原生 Loongnix 系统单线程最大延时超过 4 ms。

4.4.2 多线程测试

Cyclictest 测试指令为：sudo ./cyclictest -l100000 -m -t5 -n -p90 -i200 -q。其中线程数量设置为 5。实时 Loongnix 的测试结果如图 7 所示，原生 Loongnix 的测试结果如图 8 所示。

RT-Preempt-Linux 内核替换后成为实时系统的 loongnix 系统 5 个线程最大延迟时间均为微秒级。未实时化改造的原生 Loongnix 系统所有 5 个线程最大延时都大于 2 ms。

```
[root@localhost rt-tests_1]# sudo ./cyclictest -l100000 -m -t5 -n -p90 -i200 -q
# /dev/cpu_dma_latency set to 0us
T: 0 ( 3034) P:90 I:200 C: 100000 Min:    11 Act:   21 Avg:   21 Max:     102
T: 1 ( 3035) P:90 I:700 C:  28576 Min:    12 Act:   22 Avg:   19 Max:      67
T: 2 ( 3036) P:90 I:1200 C:  16657 Min:   13 Act:   17 Avg:   27 Max:     107
T: 3 ( 3037) P:90 I:1700 C:  11749 Min:   12 Act:   15 Avg:   23 Max:      71
T: 4 ( 3038) P:90 I:2200 C:   9072 Min:   13 Act:   22 Avg:   26 Max:      95
```

图 7 实时 Loongnix 系统 Cyclictest 多线程测试结果

```
[root@localhost rt-tests_1]# sudo ./cyclictest -l100000 -m -t5 -n -p90 -i200 -q
# /dev/cpu_dma_latency set to 0us
T: 0 ( 5388) P:90 I:200 C: 100000 Min:    10 Act:   24 Avg:   23 Max:   25003
T: 1 ( 5389) P:90 I:700 C:  28392 Min:    10 Act:   28 Avg:   28 Max:   24704
T: 2 ( 5390) P:90 I:1200 C:  16597 Min:   12 Act:   23 Avg:   33 Max:   24137
T: 3 ( 5391) P:90 I:1700 C:  11708 Min:   10 Act:   17 Avg:   36 Max:   23919
T: 4 ( 5392) P:90 I:2200 C:   9071 Min:   10 Act:   21 Avg:   28 Max:   24632
```

图 8 原生 Loongnix 系统 Cyclictest 多线程测试结果

5 结论

本文首先分析了 Loongnix 系统的标准 Linux 内核影响实时性能的 3 个重要因素，探究了 RT-Preempt-补丁的实时性优化原理，提出一种基于 RT-Preempt-Linux 实时内核替换 Loongnix 系统原生 Linux 内核的方法，实现 Loongnix 实时性优化和实时化改造，给出了两种实时性能测试软件的设计方法，并用自设计软件和专用实时性测试工具软件对实时化的 Loongnix 系统和原生 Loongnix 系统进行实时性能测试与分析。测试结果表明，改造后的 Loongnix 系统的实时性较改造前有了大幅提升，进程切换时间、线程切换时间以及任务延时都远小于原生 Loongnix 系统，都能达到微秒级。

参考文献

[1] 徐意泊，陈富浩，丁振华，等.基于国产龙芯 2K1000 龙芯

(下转第 111 页)

[9] 王豪,纪少波,刘振革,等.高速行驶车辆位置信息快速检测系统设计[J].测控技术,2019,38(8):84-88.
[10] 沈健.一种基于PIC18F45芯片车辆检测系统设计[J].电子技术,2018,47(10):76-77,69.

(收稿日期:2021-09-18)

作者简介:
周亚(1990-),男,硕士,工程师,主要研究方向:硬件电路设计。
戴伟(1990-),通信作者,男,硕士,工程师,主要研究方向:物联网应用,E-mail:daiwei@cetc-iot.com。
张鑫(1994-),男,硕士,工程师,主要研究方向:硬件电路设计。

扫码下载电子文档

(上接第102页)

派的内核系统启动[J].现代信息科技,2018,2(12):29-34.
[2] 任礼虎.一种基于龙芯2K1000的COM-E功能主板产品设计方法[J].电子技术,2018,47(4):46-47.
[3] 齐刚.龙芯拟发布多款芯片——国产唯一支持多路互联的"中国芯"诞生[J].计算机与网络,2017,43(9):24-25.
[4] 钱家乐.计算机操作系统的应用与发展分析[J].无线互联科技,2015(19):95-96.
[5] 俞冠中,王巍,刘玉升.基于状态机的AI/AO模块HART通信软件实现方法[J].测控技术,2020,39(11):106-112.
[6] 王巍,俞冠中,靳子洋.基于Cortex-M的嵌入式设备固件更新方法研究[J].单片机与嵌入式系统应用,2021,21(6):69-73.
[7] 徐海亚,赵增基,朱波,等.基于中标麒麟的POWERLINK节点实时性解决方法[J].火力与指挥控制,2014,39(5):164-167.
[8] 刘剑,仲宇,王琦.嵌入式Linux实时性改造技术综述[J].航天控制,2018,36(2):93-97.
[9] 赵旭,夏靖波,王哲.Linux内核进程调度的研究与改进[J].测控技术,2009,28(Z1):65-67.
[10] 王朴,周晴.基于龙芯1E的嵌入式Linux实时性的优化与可靠性设计[J].微电子学与计算机,2019,36(11):17-21.
[11] 王娜,李彦峰,孙菲艳,等.Linux中断线程化分析及中断延时测试[J].智能计算机与应用,2018,8(6):20-23,27.
[12] 袁辉建,陈曾汉,晏来成.基于ARM9的嵌入式Linux测控系统实时性增强研究[J].测控技术,2007(1):61-63.
[13] 刘宇帅,苏宇,王金波,等.航天嵌入式Linux实时性能优化研究[J].航天控制,2018,36(3):57-62,78.
[14] 张晓龙,郭锐锋,陶耀东,等.Linux实时抢占补丁研究及实时性能测试[J].计算机工程,2014,40(10):304-307.
[15] 王帅华,杨东升,王允森,等.基于ARM的Linux实时抢占补丁的研究与实现[J].组合机床与自动化加工技术,2015(9):1-4.
[16] 张帆,求伟,韩大鹏.Linux在嵌入式实时系统的研究与改进[J].制造业自动化,2011,33(3):87-89.

(收稿日期:2021-12-11)

作者简介:
俞冠中(1989-),男,硕士,工程师,主要研究方向:智能通信网络与信息处理、计算机网络与系统安全、云计算与物联网安全、操作系统及虚拟化技术、分布式计算机控制技术。
韦雄(1989-),男,硕士,工程师,主要研究方向:智能通信网络与信息处理、计算机网络与系统安全、云计算与物联网安全、操作系统及虚拟化技术、分布式计算机控制技术。
史旭明(1972-),通信作者,男,博士,教授级高级工程师,主要研究方向:智能通信网络与信息处理、计算机网络与系统安全、云计算与物联网安全、操作系统及虚拟化技术、分布式计算机控制技术,E-mail:shixuming@snpas.com.cn。

扫码下载电子文档

(上接第106页)

tional Geometry:Towards Geometric Engineering.Berlin:Springer,1996:741-778.
[6] RENKA R.Algorithm 772:STRIPACK:Delaunay triangulation and Voronoi diagram on the surface of a sphere[J].ACM Transactions on Mathematical Software,1997,23:416-434.

(收稿日期:2022-01-28)

作者简介:
刘壮(1988-),男,博士,工程师,主要研究方向:高性能计算。
黄小猛(1980-),通信作者,男,博士,副教授,主要研究方向:海洋模式、并行计算与大数据,E-mail:hxm@tsinghua.edu.cn。

扫码下载电子文档

版权声明