第32卷 第23期 计 算 机 工 程 2006年12月
Vol.32 No.23 Computer Engineering December 2006
·软件技术与数据库· 文章编号：1000—3428(2006)23—0070—03 文献标识码：A 中图分类号：TP316

提高嵌入式 Linux 时钟精度的方法

王 霞[1]，马忠梅[1]，何小庆[2]，江文瑞[2]，黄武陵[3]
(1. 北京理工大学信息学院，北京 100081；2. 北京麦克泰软件技术有限公司，北京 100094；3. 中国科学院自动化研究所，北京 100080)

摘 要：突破 Linux 内核在实时应用方面的缺陷主要体现在增加 Linux 内核的可抢占性、细化时钟粒度和调度算法上。该文从时钟精度的角度出发，介绍了目前流行的嵌入式操作系统在实时性方面的改进方法，分析了 MontaVista Linux 采用的高精度定时器 HRT 机制的原理、HRT 对 Linux 内核的改造方法及其在 ARM 平台上的实现方法等。

关键词：嵌入式 Linux；时钟精度；高精度定时器；实时系统；ARM

Method of Timer Resolution Improvement for Embedded Linux System

WANG Xia[1], MA Zhongmei[1], HE Xiaoqing[2], JIANG Wenrui[2], HUANG Wuling[3]
(1. School of Information, Beijing Institute of Technology, Beijing 100081; 2. Beijing Microtech Software Technology Ltd., Beijing 100094; 3. Institute of Automation, Chinese Academy of Sciences, Beijing 100080)

【Abstract】 Linux with real-time constraints are growing in the field of embedded system. In order to break through obstacle in real-time applications aspects, it is sensible to add preempt feature in Linux kernel, improve timer resolution and modify schedule strategy to satisfy real-time requirments. The thesis summarizes several modification methods of improving timer resolution and focuses on popular method named HRT, which applies comprehensively in embedded Linux system.

【Key words】 Embedded Linux; Timer resolution; HRT; Real-time system; ARM

时钟是操作系统基本活动的基准，系统用它来维持系统时间、监督系统运作。一般的 Linux 内核缺乏高精度的时钟，而依赖低精度时钟无法分辨高精度实时任务的到来，实时应用得不到满足。

为了实现对实时任务的精确控制，可以通过改进使内核支持高精度的时钟来满足系统的需求。不同的体系结构采用了不同的措施来提高时钟的精度。对于 ARM 处理器来说，Linux 内核的调度单位(scheduling time slice)的最大值为 10ms，而时钟精度主要取决于 RTC 的精度；MIPS、PPC、x86 等处理器还提供了基于总线仲裁的计数器，称为时间印记计数器 (Timer Stamp Counter, TSC)来提高时钟精度。后者时钟的频率一般是系统总线时钟频率的 1/4。

1 提高时钟精度的方法

开源的嵌入式操作系统对改进时钟精度提出了一些方案和设想，主要有 KURT-Linux、RT-Linux 和 MontaVista Linux 等。它们采用了不同的思路和技术方案，各有优劣，以下简要说明。

1.1 KURT-Linux

KURT-Linux(Kansas University Realtime OS)由 Kansas 大学研制开发，通过对 Linux 内核进行内部改造来满足实时应用需求，是第 1 个面向硬实时应用的 Linux 变种。

在时钟精度方面，KURT-Linux 将系统时钟的精度从原来的 10ms 提高到了μs 级。KURT-Linux 的解决办法很巧妙，它改变时钟中断的固定频率模式，将时钟芯片设置为单次触发模式(One Shot Mode)，即每次时钟芯片设置一个超时时间，然后到该超时事件发生时，在时钟中断处理程序中再次根据需要给时钟芯片设置一个超时时间(以μs 为单位)。

例如，在 Pentium 架构下，KURT-Linux 利用 CPU 的 TSC 来跟踪系统时间。它以到期 TSC 时标和当前系统时间 TSC 时标之差作为时钟发生器芯片的精度，设置硬件时钟发生频率，这样就可以动态地改变系统的时钟精度，精度可达 CPU 主频的时间精度。

KURT-Linux 提供的这种变长的时钟精度的方法，既保证了特定实时任务的精度需求，又避免了不必要的调度负担。这种方法需要频繁地对时钟芯片进行编程设置。

1.2 RT-Linux

RT-Linux 是新墨西哥工学院研制的一个基于 Linux 的硬实时系统。它采用双内核方法，在原有 Linux 基础上设计一个用于专门处理实时进程的内核，然后把整个 Linux 作为这个微内核上运行的一个进程。

在时钟精度方面，RT-Linux 类似 KURT-Linux，也是通过将系统的实时时钟设置为单次触发状态，然后利用 CPU 的计数寄存器提供高达 CPU 时钟频率的定时精度，可以提供十几个微秒级的调度粒度。特别地，使 Intel 8354 定时器芯片工作在 interrupt-on-terminal-count 模式。使用这种模式，可以使中断调度得到 1μs 左右的精度。这种方法的定时器精度高而系统开销是最小的。

作者简介：王 霞(1982－)，女，硕士生，主研方向：基于嵌入式 Linux 的实时性技术；马忠梅，副教授；何小庆、江文瑞，硕士；黄武陵，硕士、高工

收稿日期：2005-12-01 **E-mail：**guaimaow@gmail.com

1.3 MontaVista Linux

MontaVista Linux 是 MontaVista Software 的创立者 James Ready 领导开发的嵌入式 Linux，它是面向各种嵌入式应用的 Linux 发布，其前身是 HardHat Linux。MontaVista Linux 通过对 Linux 内核进行内部改造，直接修改原有 Linux 内核的数据结构等来满足实时需要。

在时钟精度方面，MontaVista Linux 采用高精度定时器 HRT(High Resolution POSIX Timers)使得定时器可以产生任何微秒级的中断，无需每一个微秒都产生中断，将系统时钟的精度从原来的 10ms 提高到了μs 级。

HRT是由MontaVista软件公司的George Anzinger维护的开源项目，该项目旨在为Linux操作系统提供符合POSIX API标准的高精度定时器。它抛开传统的周期中断CPU的方法，在最早需要调度时间的那一刻中断CPU，即one-shot模式，这与KURT-Linux和RT-Linux类似。除此之外，HRT向应用程序提供接口，符合POSIX 1003.1b API标准，在嵌入式领域中应用广泛。目前MontaVista Linux的所有版本都支持HRT，并被电信级Linux(CGL)工作组和CELF(Consumer Electronics Linux Forum)论坛发布的规范采纳，其开源网址是http://sourceforge.net/projects/high-res-timers。另外，开源社区中Ingo Molnar主持并开发的Linux实时实现(Ingo's RT patch)最近发布的 2.6.13-rt6 patch中对HRT提高时钟精度的方法给予肯定，并重写HRT部分代码实现“ktimers”框架。

1.4 Linux-SRT

Linux-SRT 是剑桥大学 David Ingram 的博士论文项目，它属于软实时的 Linux，自从 Ingram 在 2000 年从剑桥毕业以后，该项目就再没有人维护。

Linux-SRT 也提高了系统的定时精度。但它并没有采用惯用的将时钟芯片置于单次触发模式的做法，而是简单地修改了 Linux 内核中 HZ 的定义，将 Linux 的时钟频率由每秒 100 次提高到了 1 024。

提高 HZ 值是 Linux 中提高时钟精度的最简单的方法。HZ 代表时钟频率，它是一个与体系结构相关的常数。诚然，增加时钟频率可以提高时钟中断精度，例如 Linux 2.6 内核中 x86 等架构中将 HZ 值从 2.4 下的 100 提高到 1 000，即每秒中断 1 000 次，时钟频率为 1ms。但是，简单地提高 HZ 值将使时钟中断更频繁地产生，必定引起调度负载的增加，这样减少了处理器处理其他工作的时间，而且还会频繁打乱处理器的高速缓存，在嵌入式领域中使用这种方法并不见效。

2 高精度定时器的实现

2.1 HRT 原理

尽管不同体系结构中系统定时器实现不尽相同，但其根本思想都是为 Linux 内核提供一种周期性触发中断的机制。以 ARM 体系结构为例(如未特殊说明，以下内容均以 ARM 体系结构为例)分析时钟中断服务程序，Linux 在每次时钟中断上半部，给 jiffies_64 变量增加 1，更新墙上时间 xtime；在时钟中断下半部，Linux 采用 softirq 软中断机制推迟中断处理，执行已到期的动态定时器。

这个周期的时钟中断是导致低时间精度的一个原因。HRT 绕过这个时钟，利用附加的硬件资源，添加一个高精度的时钟，利用这个时钟使系统无需每一个微秒都产生中断，而在任何微秒需要中断时再产生。这要求修改定时器 timer_list 结构，增加用来设定高精度定时器的超时值的成员，然后进一步修改时钟中断上半部处理函数、下半部的软中断处理函数以及一些接口函数，通过灵活的安排定时器中断时间，在不扰乱系统时钟中断的情况下提高定时器精度。

2.2 HRT 实现

HRT 利用附加的硬件资源得到比 10ms 更高的定时精度，这个硬件资源因体系结构不同而不同，以 X86 和 ARM 体系结构为例。

X86 体系结构下，CPU 包含一个 64 位的时间印记计数器(TSC)的寄存器，该寄存器是一个不断增加的计数器，它在 CPU 的每个时钟信号到来时加 1。利用 TSC 实现 HRT 可得到更为精确的时间度量。

ARM 架构下没有类似 TSC 的计数器，它一般设有多个可编程计数器/定时器，可以利用系统未使用到的可编程定时器来实现 HRT，注意该可编程定时器必须能达到微秒级精度。例如 AT91RM9200(ARM920T 核)处理器的 Timer Counter(TC)模块包含 3 个 16 位定时器，每个定时器可编程实现计数、频率测量等用途，利用其中一个定时器实现 HRT 便可；S3C2410(ARM920T 核)处理器包含 5 个 16 位定时器，系统使用其中一个作为系统时钟，那么可以利用系统未使用到的其余 4 个中的一个来实现 HRT。

2.2.1 HRT 中断请求队列(HRT IRQ)

系统中增加了一个高精度定时器之后，便要维护两个与时钟有关的中断请求队列：系统时钟中断请求队列(timer_irq)和 HRT 中断请求队列(hr_timer_irq)，它们分别对应各自的中断服务程序。如图 1 所示为带有 HRT 的时钟中断服务程序执行流图。

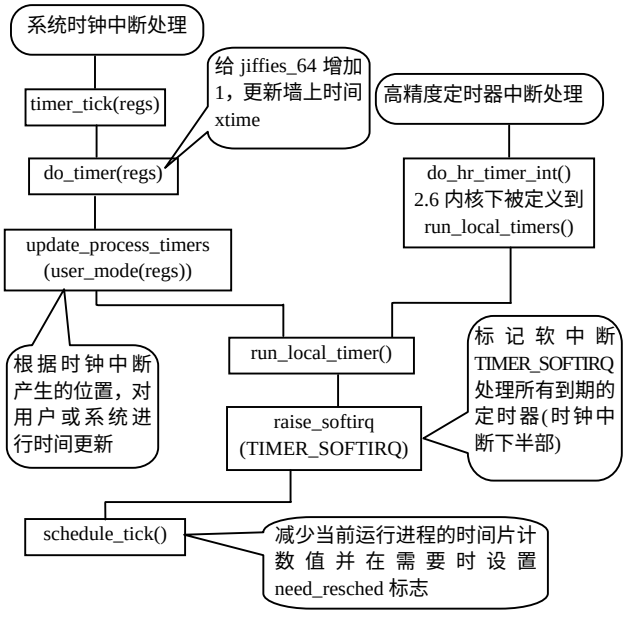

图 1 时钟中断服务程序执行流图

图 1 的左半部是系统维护的原有时钟中断服务程序。这个处理过程从系统开机后就一直在执行，其中 do_timer()函数给 jiffies_64 增加 1，更新墙上时间(xtime 变量)；update_process_timers()函数与进程有关，更新当前进程与时间相关的变量。

图 1 的右半部是 HRT 中断服务程序。如果系统中的设备驱动程序或应用程序利用了高精度定时器产生中断，那么两次产生中断之间的间隔肯定比正常的系统时钟中断短，系统便调用高精度定时器服务程序。

从图 1 中可以看出，两个时钟中断服务程序最终都执行

run_local_timer()，之后标记软中断 TIMER_SOFTIRQ，系统将在稍后合适的时机执行时钟中断下半部。时钟中断下半部主要处理所有到期的定时器，这是 HRT 修改的重点部分，将在下节详细介绍。

2.2.2 时钟中断下半部的修改

系统中添加一个 HRT 中断请求队列之后，所有与定时器相关的数据结构、函数等都要做相应修改。值得一提的是，内核中使用 CONFIG_HIGH_RES 宏定义来控制 HRT，如果内核中配置了 CONFIG_HIGH_RES，则高精度定时器机制有效，否则 HRT 不起作用。

(1)增加 sub_expires 成员

为了获得微秒级的定时粒度，HRT 在定时器 time_list 结构中增加一个成员 sub_expires 来指示在一个高精度定时器到期时处理 jiffy 值之外的机器指令周期数，该成员提供了高精度的判断标准。

(2)修改与定时器相关的函数

这些函数包括初始化定时器数据结构 init_timer()、激活定时器的函数 add_timer()、更改已经激活的定时器超时时间的函数 mod_timer()等。这些函数由内核提供，是一组与定时器相关的接口，用来简化管理定时器的操作。

(3)修改时钟中断下半部

从图 1 中看到，在时钟中断服务程序快要结束时，raise_softirq()函数标记软中断 TIMER_SOFTIRQ，使得内核在稍后一段时间执行时钟中断下半部，即执行 TIMER_SOFTIRQ 对应的 run_timer_softirq()函数。这个函数是实现 HRT 最关键的地方，因为时钟中断下半部执行所有到期的定时器，包括到期的高精度定时器。

在带有 HRT 的系统中并没有 tv1、tv2、……、tv5 这样的数据结构，而是添加一个大小为 512 的 new_tvec 指针数组，每个指针都指向了一个高精度定时器队列。判断某高精度定时器是否到期，可从以下 3 种情况来考查：

1)timer->expires 小于当前时间(jiffies)；

2)timer->expires 等于当前时间(jiffies)，同时 timer->sub_expires 小于等于 sub_jiffies_f；

3)timer->expires 等于当前时间(jiffies)，同时 timer->sub_expires 大于 sub_jiffies_f。

其中 sub_expires 是系统新增加的成员，sub_jiffies_f 根据 jiffies 计算出来，表示一个 jiffies 内包含的机器指令周期数。如果有高精度定时器满足前 2 种情况，则该高精度定时器到期，系统执行其挂接的服务程序。如果满足第 3 种情况，则该高精度定时器没有到期，那么安排该下一个高精度定时器。

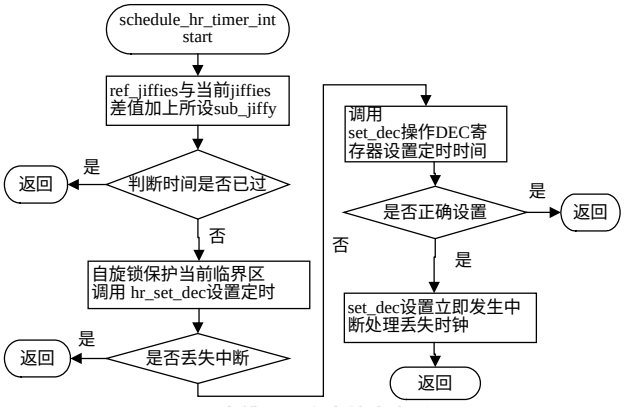

图 2 安排下一个高精度定时器

安排下一个到期的高精度定时器(schedule_hr_timer_int)是很重要的一步，如图 2 所示的是其执行流图。从图 2 中可以看出，下一个高精度定时器的到期时间由参考 jiffies(ref_jiffies) 和 sub_jiffy 两个参数决定，如果想安排的时间已经过了，则返回一个非零值；否则使用自旋锁保护临界区后设置定时时间，再判断安排的中断是否发生，若未发生，则设置新的定时时间。

综上所述，HRT 对 Linux 内核改造的关键之处在于修改定时器结构、定时器操作的所有函数以及时钟中断下半部的代码，其中时钟中断下半部的代码略为复杂。下面列出 HRT 对 Linux 内核改造之后在时钟中断下半部的伪码：

```
void __run_timers()
do
run_timer_list_again:
        for each timer in queue
          do
              if timer.expire<= jiffes OR timer.expires == jiffies and
                                  timer.sub_expire <= sub_jiffies
                        then      fn=timer.function
                                  data= timer.data
                                  detach timer from queue
                                  timer.next=timerl; prev=NULL
                        set timer to be running timer
                                  run fn(data)
              else
                  if  timer.expires = jiffies and
                        timer.sub_expire > sub_jiffies
                        //means there is a sub jiffy timer to shoot
                        then
                                  schedule the next hr timer interrupt,
                                  if return value says we have missed it
                                          then goto  run_timer_list_again
                                  fi
                        exit loop
                  fi
          done
done
        if timer.expire > jiffes  and no sub_jiffy
                then just goto run_timer_list_again or wait the jiffy to change
```

2.2.3 POSIX 接口

HRT 的一个很重要的特性是向应用程序提供接口，它符合 POSIX 1003.1b API 标准，在嵌入式领域中应用广泛。

这部分修改主要是实现 kernel/posix-timers.c，这个文件主要是针对 sub_jiffy 值而相应地添加一些接口操作，例如 sys_timer_create()、sys_timer_gettime()、sys_timer_settime()、sys_clock_settime()、sys_clock_nanosleep()等。

值得说明的是，由于添加了 sub_jiffy 作一个 jiffy 以下精度的睡眠唤醒动作，在 nanosleep 一些函数中还必须实现传递睡眠的高精度 sub_jiffy 参数的功能，例如 do_clock_nanosleep()中：

new_timer.sub_expires = restart_block->arg4;

另外，在 include/linux/posix-timers.h 中定义实现 posix_get_now(), posix_time_before(timer, now) 以及 posix_bump_timer(timr)等函数。

至此，内核具备完整的高精度定时器功能。

3 总结

操作系统中，使用周期时钟并不能得到要求的计时器定时精度，这也是导致低时钟精度的一个原因。系统设计者必须在时钟中断处理函数开销与计时精度之间做一个折中。

（下转第 96 页）