NTP 时间同步性能研究

徐怡山 陶 克 贺 鹏

(三峡大学 电气信息学院,湖北 宜昌 443002)

摘要:网络化计算和分布式系统的应用,对计算机系统的时间同步精度要求越来越高.分析了 NTP (Network Time Protocol)同步效果对操作系统的依赖性,描述了该模型的执行特征和系统配置部件,并给出了一个嵌入式系统的时间同步模型.

关键词:时间同步; 时间管理; 嵌入式系统设计; NTP

中图分类号:TP391 **文献标识码**:A **文章编号**:1672-948X(2004)06-0537-03

Study of Performance for NTP Time Synchronization

Xu Yishan Tao Ke He Peng

(College of Electrical Engineering & Information Science, China Three Gorges Univ., Yichang 443002, China)

Abstract With computing of network and application of distribution system, there is more and more precise time synchronization needed for computer system. This paper analyses the performance of NTP depending on operating system. Also the implementing features of the prototype are described; and the system configuration components are shown. In the outcome of the analysis, this paper shows a clock model for embedded system.

Keywords time synchronization; time management; embedded system design; network time protocol (NTP)

一般来说,计算机时钟在可靠性和精确度上都是有限的,由于温度变化、电磁干扰、振荡器老化和生产调试等原因,时钟的振荡频率和标准频率之间存在一些误差.因此,如果计算机需要精确的时间处理,计算机时钟就必须与一个标准的时间同步.由美国特拉华大学的 David L. Mills 教授提出的 NTP 是互联网上公认的时间同步工具.本文描述了计算机系统的时间同步并给出了利用 NTP 在 pSOS 这样的 RTOS(实时操作系统)上时间同步模型的设计和实现.

1 NTP 对操作系统的依赖性分析

计算机的时钟模型一般包括硬件时钟和由硬件时钟维持的软件时钟.软件时钟或者是内核时钟是由硬件时钟上的定时器产生的周期性中断维持的,这些周期性中断称为 *tick*[1,2].时间同步方法应考虑计算机系统的时钟模型和操作系统中与时间操作相关的服务.

同步进程、OS、硬件时钟 3 者之间的操作关系不同,会形成不现的同步特点.尤其是同步进程和操作系统之间的关系对于时间同步而言是非常关键的.3 种时间同步层次结构如图 1 所示:(a)容易实现且更具有移植性,但不易优化.另外,同步的特点也依赖于操作系统内部,如时间操作模块和实时性等.如果一个操作系统有强大时间操作模块功能,那么(a)就是一个比较理想的同步方法.(b)和(c)的同步进程可以通过直接与硬件打交道来使其优化,但它们并不易实现,可移植性也不强.在(b)和(c)中,同步进程可以在效果和可移植性之间找到一个理想的折衷点,那些对时间同步起关键作用的部分则采用直接操作硬件的方式来实现.由于现有操作系统的时间功能有限性,本文以(b)图中的模型为目标.NTP 和操作系统的关

收稿日期:2004-03-08
作者简介:徐怡山(1975—),男,硕士研究生.

系如图1(a)所示. NTP通过操作系统的系统调用来实现时间同步,具有以下两个特点:

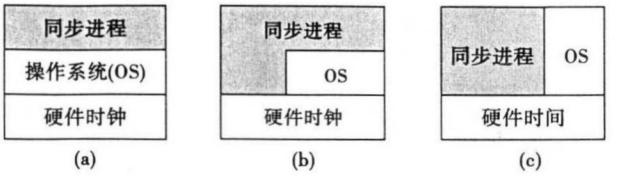


图1 时间同步进程、OS、硬件时钟关系

①时间操作

a. 利用系统API函数调整时钟

NTP建立了其自身的逻辑时钟模型,约束逻辑时钟并计算修正量. NTP将更新内核时钟的任务交给了操作系统. 因此,当NTP将修正的量作为API函数参数调整时钟时,如果NTP运行的操作系统上没有提供相应的API函数,则NTP的同步效果就很难令人满意. 也就是说,NTP的同步性能随着操作系统时间操作模块的不同而不同[3].

b. 缺少实时时钟管理

NTP通过修正参数调用操作系统API函数同步内核时钟,但该参数并不用于同步物理硬件时钟. 硬件时钟调整因操作系统时间操作模块的不同而不同. 当操作系统重起,内核时钟被硬件时钟重置. 如果NTP的同步结果没有写入硬件时钟,则同步效果就不能得以保证. NTP的同步效果受操作系统硬件时钟修正策略的影响.

②实时特征

NTP的实时性是通过一些技术来实现的. 例如:异步事件通知,异步定时器(计数器),非阻塞套接字I/O. 但是NTP不能够保证时间戳的实时处理和重要I/O事件的实时处理,这些是通过操作系统来达到的. 图2说明了NTP的时间戳的执行过程:当有包到达时,在接收方打上时间戳标记,在发送方发出包之前,在发送方打上时间戳标记. NTP不能保证虚线部分的原子性. 例如,如果内核中断在包到达 t_0 时刻和打上时间戳标记之间发生,NTP进程将在 α 期间被挂起,则打上时间戳标记也将有误差 α[4], $T_{\text{arrival}}=t_0+\alpha$.

2 时间操作模型的设计与实现

本节描述了一个实时时间同步系统的设计和实现. 在本文的设计中,综合处理器性能,需要的最小ROM,以及多任务内核特点,选择pSOS实时操作系统. 但是pSOS系统时间操作模块功能较弱,这就为在pSOS上实现同步时钟带来了一定的困难.

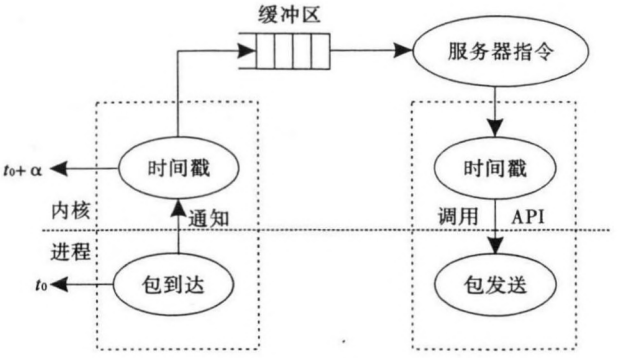


图2 时间戳执行图表

2.1 时间服务模型的组成

采用pSOS为其操作系统,时间服务器用普通的晶振作为本地时钟,选择互联网上的公共时间服务器作为外部时钟源,IC采用三星公司的KS32C5000.

2.2 时间操作模型的设计

pSOS时间操作单元是一个时钟 *tick*,一个 *tm-tick* 系统调用之间的间隔,称为 *tick*,而 *tm-tick* 系统调用是在每次计数器中断时被中断服务例程调用的. 针对pSOS系统上的时间操作能力有限的缺陷,希望设计的系统有如下特点. 新系统应当有强大的本地时钟管理能力,它应该支持逐步调整,具有容错性,和高时间分辨率等特点. 这些都依赖于如何控制 *tick* 值,下面就说明了如何控制这个 *tick* 值.

①pSOS内核I/O模型

NTP利用硬件时钟驱动操作时钟I/O. pSOS内核有I/O特权来配制用户从应用层访问这些驱动的标准接口,pSOS内核利用I/O切换表访问这些驱动. 和本地时钟操作相关的I/O服务记录在切换表中,同时它们在本地时钟设备入口的I/O切换表也进行了注册,在I/O切换表中定义了从应用层到pSOS的接口以及从pSOS到驱动的接口. 这些组成了基本的时钟调整过程.

②本地时钟操作

pSOS上的本地时钟操作要求4种I/O服务,ClockInit,ClockWrite,ClockRead,ClockCntrl. 图3描述了在pSOS上的时钟操作模型. 黑体字部分是与时钟I/O服务相关的时钟驱动. ClockCntrl部件位于"计数器频率控制"部分

③逐步调整:频比法

ClockCntrl驱动遵循Unix 4.3 bsd *adjtime* 模型. 工作模型如图4所示. *tick* 和 $tick_{\text{adj}}$ 两个参数决定了 *adjtime* 模型的时钟速率,时钟以3个不同的速率运行,分别是 *tick*, $tick+tick_{\text{adj}}$, $tick-tick_{\text{adj}}$. 如果调用了 *adjtime* 并给出了参数 δ,则调整间隔 Δt 和最大调整频率 f_{adj} 可以从下面的公式算出[2,5].

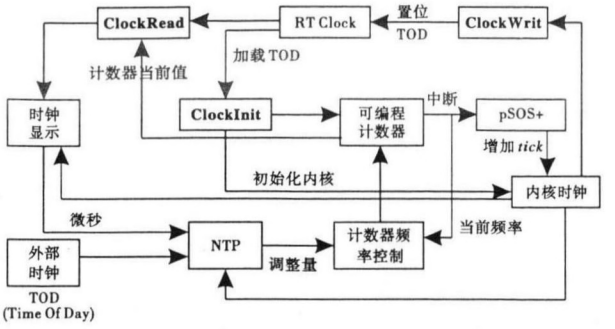


图 3　pSOS 时钟操作性模型

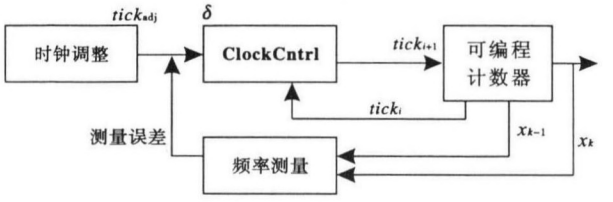
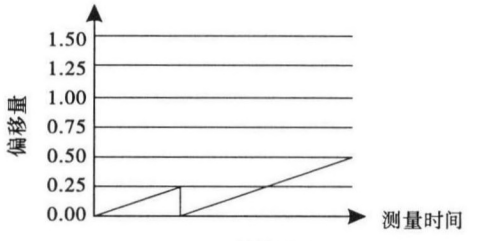
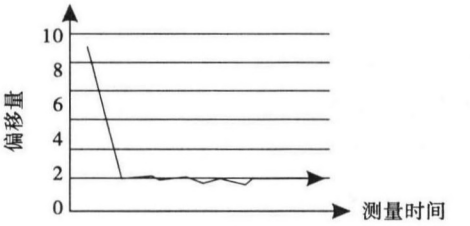


图 4　pSOS+内核本地时钟调整模型

$$\Delta t=\frac{1}{f_{\text{adj}}}=\delta\frac{tick}{tick_{\text{adj}}}\qquad \varphi=\frac{1}{tick},\varphi_{i+1}=\varphi_i+f_{\text{adf}}$$



(a)*tm-set*累计误差



(b)频比法同步效果

图 5　改进前后系统误差

测量误差用计数器中的采样 x_{k-1} 和 x_k 计算得到[2]. *tick* 间隔的新值，$tick_{i+1}$ 由调整模型的公式计算出来. 因此，内核时钟能够通过控制 *tick* 的间隔大小前进或后退.

2.3　同步效果

改进前系统工作过程：NTP 调用 *tm-set* 来将本地时钟每秒前进或后退±500 us. 但这种调整精度很难保证，因为 pSOS 的调整分辨率是 *tick* 单元，但 NTP 是微秒级，因此，用微秒级的调整量调用 *tm-set* 后，本地时钟和外部时钟的差额并不在零左右，这个差额会随着调用次数的增加而增加. 图 5(a)给出了在 pSOS 上用 *tm-set* 的 NTP 的调整图.

采用改进后的时间模型，减少了 NTP 同步性能对操作系统的依赖性，提高了 NTP 同步效果，图 5(b)给出了改进后的时间操作模型的实验结果. 我们不难看出，图 5(a)中的偏移量在增加，图 5(b)中的偏移量收敛趋向于零.

3　结　语

本文系统地分析了 NTP 同步效果对操作系统的依赖性. 例如，未知执行时间的内核过程会导致时间同步的误差，虽然出现这样的情况很少，但操作系统应该提供实时特性和相应时间操作模块以提高 NTP 的同步效果. 针对 pSOS 上时间操作能力有限的缺陷. 本文提出的时间操作模型包括逐步调整量的 tick 计数器频率的控制，保证同步效果更新实时时钟，以及高精度的微秒级分辨率. 实验表明，使用该模型的 NTP 同步性能大大提高.

未来的工作是进一步增强系统的实时特性，并采用时间同步的误差模型，以便更好地提高 NTP 的性能.

参考文献：

[1] Levine J. An Agorithm to Synchronize the Time of a Computer to Universal Time[J]. IEEE/ACM Trans On Netwroking, 1995, 3(1): 42～50.

[2] RFC1305－1992, Network Time Protocol 3 Specification, Implementation and Aanalysis[S].

[3] Mills D. Adaptive Hybrid Clock Discipline Algorithm for the NTP[J]. IEEE/ACM Trans On NetWroking, 1998, 6(5): 505～514.

[4] Yi Y S. An Analysis of Dynamic Timing Error for Clock Synchronication in Computer NetWorks[D]. TEHRAN, IRAN: PNU, 1997.

[5] RFC1589－1994, A Kernel Model for Precision Timekeeping [S].

[责任编辑　张　莉]