一种嵌入式实时操作系统 Nucleus Plus

贺 磊

(信息工程大学,国家数字交换系统工程技术研究中心,河南 郑州 450002)

摘要:本文介绍了美国 ATI 公司的嵌入式多任务操作系统 Nucleus Plus 的基本内容,及其内核(kernel)设计及实现中的一些典型的数据结构、概念及方法。

关键词:嵌入式实时操作系统;RTOS;内核;任务;任务调度;任务通信;任务同步
中图分类号:TP316.21 文献标识码:A

1 前言

实时性(Real-Time)指系统能对外部和内部事件在合适的时间内产生正确的反应。实时系统不仅要求有很好的灵活性和可靠性,而且要求系统的操作具有时间特性。实时系统分为"硬实时系统"和"软实时系统"。"硬实时系统"是指系统中所有的截止期限必须被严格的保证,否则将导致灾难性的后果;而"软实时系统"在截止期限被错过的情况下,只造成系统性能下降而不会带来严重恶果。

近些年来,随着嵌入式系统飞速的发展,嵌入式实时操作系统广泛地应用在制造工业、过程控制、通讯、仪器、仪表、汽车、船舶、航空、航天、军事装备、消费类产品等方面。今天嵌入式系统带来的工业年产值已超过了 1 万亿美元。嵌入式实时操作系统(E-RTOS)是运用在嵌入式系统中的实时操作系统。操作系统(Operationg System)作为管理计算机软硬件资源的工具,为用户提供一台比物理计算机更易使用的虚拟计算机。它可以分为批处理操作系统、分时操作系统、实时操作系统、网络操作系统、分布式操作系统等。利用嵌入式实时操作系统(E-RTOS)提供的各种便利,用户在实现自己的系统时,不用亲自编写相应的监控程序,可以节省大量的时间和精力。

当前嵌入式操作系统有数百种,比较流行的有 Nucleus Plus, QNX, LynxOS, Psos, VRTX, VXWorks, WINDOWS CE, HOPEN, CHORUSOS, PALM OS 等。它们各有特色,用户可以根据自身的软硬件环境的要求,选择合适的操作系统。

本文介绍的 Nucleus Plus 是为实时嵌入式应用而设计的一个抢先式多任务操作系统,其 95% 的代码是用 ANSI C 写成的,因此非常便于移植并能够支持大多数类型的处理器。从实现角度来看,Nucleus Plus 是一组 C 函数库,将应用程序与核心函数库链接在一起,生成一个目标代码,下载到目标板的 RAM 中或直接烧录到目标板的 ROM 中执行。在典型的目标环境中,Nucleus Plus 核心代码区一般不超过 20K 字节大小。

Nucleus Plus 采用了软件组件(COMPONENT)的方法。每个组件具有单一明确目的,通常由几个 C 或汇编模块构成,提供清晰的外部接口,对组件的引用就是通过这些接口完成的。除了少数一些特殊情况外,不允许从外部对组件内的变量进行访问。由于采用了软件组件的方法,Nucleus Plus 各个组件非常易于替换和复用。Nucleus Plus 的组件包括任务控制、内存管理、任务间通信、任务的同步与互斥、中断管理、定时器管理及 I/O 驱动管理等。

Nucleus Plus 提供 C 源代码给每一个用户。这样用户不但可以进行 RTOS 的学习和研究,而且产品在量产时也不必支付 License,可以省去大量费用。除内核(Kernel)外,Nucleus Plus 还提供种类丰富的功能模块,例如 TCP/IP 网络模块、嵌入式文件系统、嵌入式图形模块 Grafix、支持 Internet 的 WEB 模块、工控机 BIOS 模块、Nucleus RMON、Nucleus SNMP 等。

Nucleus Plus 支持大部分流行的 CPU,如 X86, 68xxx, 68HCxx, NEC V25, ColdFire, I960, MIPS, SPARC-

收稿日期:2000-10-20

作者简介:贺磊(1974-),男,河南隧平人,信息工程大学教员,主要研究方向为嵌入式操作系统内核研究。

Clite, ARM6/7, StrongARM, PowerPC 等。针对不同的 CPU 类型, Nucleus 还提供编译器、动态连接器、多任务调试器等相应的工具来配置用户的开发环境, 方便软件的开发和调试。

2 Nucleus Plus 内核的系统结构

Nucleus Plus 内核(Kernel)的主要目的是管理实时任务的竞争运行(共享 CPU), 为应用提供各种便利, 快速响应外部事件, 实现实时性。在 Nucleus Plus 中, 任务之间可以按照优先级和时间片方式来共享 CPU 资源, 通过邮箱(mailbox), 队列(queue)和管道(pipe)进行通信, 任务之间的同步和互斥是通过信号量(semaphore)、事件组(event group)和信号(signal)进行。Nucleus Plus 提供动态和分区内存(dynamic/partition memory)两种存储器管理机制。Nucleus Plus 还提供定时器(timer)来处理周期性事件和任务的睡眠和挂起超时。Nucleus Plus 将这些机制称之为软件组件(SOFTWARE COMPONENT)。Nucleus Plus 为每一个软件组件提供了一系列的系统调用, 任务与 Nucleus Plus 的交互是在系统调用的界面上进行的。

Nucleus Plus 的系统结构如图 1 所示:

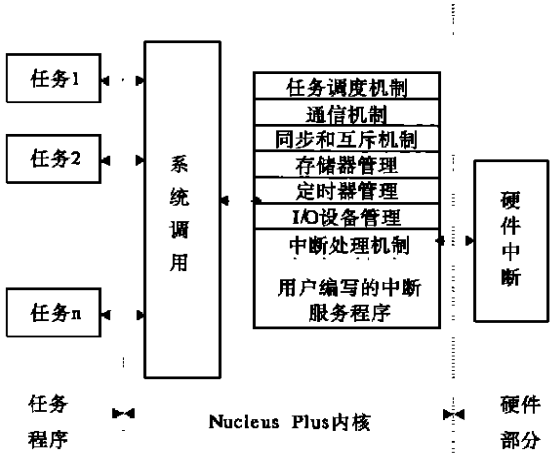

图 1 Nucleus Plus 系统结构

从图 1 可以看出, 利用 Nucleus Plus 开发平台, 用户只需编写任务(Task)代码和中断服务程序代码(ISR)。在任务代码(Task)和中断服务程序代码(ISR)中利用系统调用实现和 Nucleus Plus 的交互, 由 Nucleus Plus 来调度多个任务并行执行, 实现 CPU 的共享。由于 Nucleus Plus 根据优先级和时间片方式来共享 CPU 资源, 所以只要任务和中断服务程序的优先级设置得当, 系统的实时性就能保障。下面介绍 Nucleus Plus 的各个软件组件。

2.1 任务调度(Task Schedule)

Nucleus Plus 中的任务(Task)是有着明确目标的半独立的程序段, 在 C 中表示为 void task_name (UNSIGNED argc, VOID * argv)形式的函数。每个任务都需要一个任务控制块(TCB)和一个堆栈(STACK)。任务(Task)具有五种状态: 运行态(executing)、就绪态(ready)、挂起态(suspended)、终止态(terminated)、完成态(finished)。具体见表 1。

表 1 任务的五种状态

状态	含义
运行态(executing)	任务正在运行
就绪态(ready)	任务已经就绪, 但其他任务正在运行(拥有 CPU 的控制权)
挂起态(suspended)	任务在等待服务完成时挂起。当服务完成后, 转入就绪态
终止态(terminated)	任务未完成就被杀掉(killed)
完成态(finished)	任务完成了处理工作并返回到初始入口模块

大多数的现代实时应用需要多个任务并行运行, 而且由于任务的重要性不同, 各任务拥有的优先级不同。Nucleus Plus 拥有从 0 到 255 共 266 个优先级。优先级的数字越小则表示任务的优先级越高。一个任务的优先级在创建时设置, 并可以动态修改。

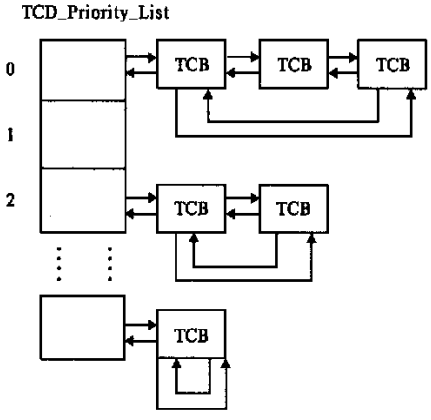

图 2 双向链表数组 TCD_Priority_List

Nucleus Plus 对任务的调度有两种方式, 优先级调度和时间片调度。当一个更高优先级的任务就绪时, Nucleus Plus 中断低优先级的任务, 保存现场, 并先运行更高优先级的任务, 这叫抢占(Preemption)。通过优先级调度可以保障高优先级的任务优先运行。相同优先级的任务间也可以采用时间片的方式轮流使用 CPU 资源, 用户只需指定每一个任务的时间片大小, Nucleus Plus 通过时钟中

断来计算任务的运行时间, 当任务的时间片耗尽后, Nucleus Plus 会自动进行任务切换。Nucleus Puls 对任务的调度利用类似双向链表数组 TCD _ Priority _ List 数据结构实现, 如图 2。其中, TCB 是任务控制块, TCD _ Priority _ List 是 TCB 指针链表的数组, 0 ~ 255 代表任务的优先级。

对于实时应用, 设置任务的优先级必须十分小心, 否则可能会造成任务饥饿(task starvation)和额外的系统开销。由于一个正在运行的任务必须是拥有最高优先级的就绪任务, 如果一个高优先级任务总是处在就绪状态, 则所有其他优先级比它低的任务就永远不会运行, 这种情形就叫任务饥饿(task starvation)。有几种解决任务饥饿的方法: 首先, 高优先级任务应该主动挂起, 以允许低优先级任务运行。其次, 几乎持续不断运行的任务应当拥有较低的优先级。另一方法是操作系统应不断提高饥饿任务的优先级。Nucleus Plus 的任务可以动态的创建和删除。对一个应用的任务数没有限制。

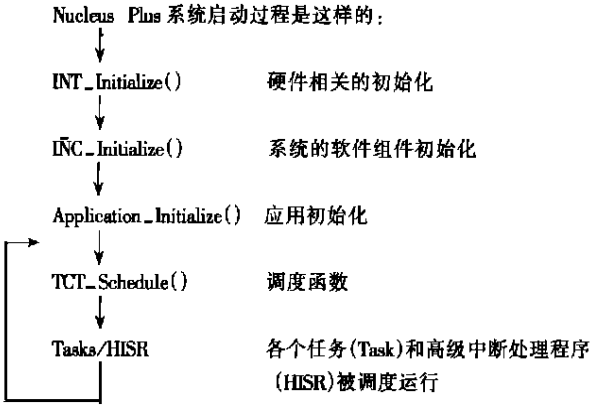

图 3　系统启动过程

应用初始化 Application _ Initialize()由用户编写, 负责应用和任务所需内存的申请和分配、任务所需的变量和软件组件的申请、任务代码、任务创建等。在 Application _ Initialize()结束后, 系统控制权就交给任务调度函数 TCT _ Schedule(), 它是一个特殊的函数, 按照优先级和时间片调度原则负责任务(Task)和高级中断处理程序(HISR)的调度和运行。

Nucleus Plus 提供信号量(semaphore)、事件组(event group)和信号(signal)三种手段进行任务同步和互斥。信号量(semaphore)、事件组(event group)是独立的公共系统工具。而信号(signal)则与特定任务相关。

信号量(semaphore)提供一种控制同时访问程序关键区(critical section)的机制。资源分配中最常使用信号量机制。Nucleus Plus 的信号量值域范围从 0 到 4, 294, 967, 294。对信号量的两种基本操作是获取信号量 NU _ Obtain _ Semaphore()和释放信号量 NU _ Release _ Semaphore()。获取信号量操作会使信号量减一, 释放信号量操作则会加一。当一个任务试图获取一个值已经是零的信号量可以被挂起。当一个释放信号量操作发生后, 任务被恢复运行。

事件组(event group)提供了一种提示某个系统事件发生的机制。一个事件表示为事件组中的一位(a bit), 这一位即一个事件标识(Event Flag)。一个事件组可以拥有 32 个事件标识。任务主动地查看事件组以确认事件是否发生, 事件在本质上是同步的。信号(signal)则更像软中断。用户要自己编写信号处理函数。当信号发生时, 任务被中断, 用户定义的特殊信号处理模块被运行, 信号处理在本质上是异步模式。

2.2　任务通信(Task Communication)

Nucleus Plus 提供了邮箱(mailbox), 队列(queue)和管道(pipe)三种通信机制, 用于任务之间的相互联系。这三种通信机制的最大不同是所通信的数据类型不同。

邮箱(mailbox)提供一种传送简单消息的低开销机制。每个邮箱可以容纳一个消息(大小是 4 个 32 位的字(word))。收发邮箱消息按值传递, 即通过消息复制进行。它是日常生活中寄信过程的抽象, 需要通信的任务通过对特定邮箱收发消息来完成通信。Nucleus Plus 中的邮箱消息格式如下:

```
NU _ Mailbox　myMailbox;
```

在 Nucleus Plus 中, 提供 NU _ Receive _ From _ Mailbox(), NU _ Send _ To _ Mailbox()两个基本消息函数。收/发消息服务提供无条件挂起、有超时挂起、不挂起三种选项。

队列(queue)和管道(pipe)提供一次传送多个消息的机制。收发队列消息和管道消息也都是按值传递, 即通过复制消息进行。一个队列消息包含一个或多个 32 位字(Word), 一个管道消息包含一个或多个 8 位字节(Byte)。定长和变长的队列和管道消息都被支持。

2.3　定时器管理(Timers)和存储器管理(Memory Management)

在实时应用中, 定时器(Timer)管理一直占据着主要的位置。Nucleus Plus 将定时器也作为一个软件组件加以实现, 可以简化我们在定时器方面的编程工作。每个 Nucleus Plus 任务都拥有一个内置

的定时器,这个定时器被用来提供任务睡眠和系统调用挂起超时服务。Nucleus Plus 中定时器的时标来自硬件时钟中断。跳(Tick)是 Nucleus Plus 的基本时钟单位。每一跳(Tick)代表一次硬件时钟中断。

Nucleus Plus 提供的定时器有两种定时方式:一种是周期性定时方式,另一种是一次性定时方式。使用定时器首先要定义一个 NU _Timer 类型的定时器变量,然后利用 NU _Creat _Timer()函数设置定时器的各种属性:包括定时器超时处理函数(用户自定义)、定时器初始时间(第一次定时器超时时间)、定时器周期(为零时是一次性定时方式,非零时代表定时器的间隔周期)等。可以利用 NU _Control _Timer()开关定时器。此外,Nucleus Plus 还具有时钟功能,可以方便的管理系统时间。

Nucleus Plus 提供分区(Partition)内存管理和动态(Dynamic)内存管理两种存储器管理方式,它们各有特点。分区内存管理具有非常好的确定性,但不够灵活。动态内存管理非常灵活,但内存分配和回收时的时延不够确定。用户可以根据不同的要求,选择合适的内存池。

2.4 中断管理(Interrupts)

中断(Interrupts)提供了一种机制可以对外部和内部事件作出立即的反应。当中断发生时,Nucleus Plus 挂起当前运行的任务,并将控制权转交给合适的中断处理模块(ISP).Nucleus Plus 在调用中断处理模块(ISR)之前自动地保留现场,并在其返回后恢复现场,用户只需要编写中断服务模块即可。

中断对所有的实时操作系统内核都产生一定的影响。中断延迟(Interrupt Latency)指系统封锁中断的时间,是判断实时操作系统(RTOS)优劣的重要指标。由于 Nucleus Plus 无例外(Exception)功能,而中断处理模块(ISP)需要访问系统提供的服务。因此,需要一种保护机制来防止多个 ISR 同时访问 Nucleus Plus 内部数据结构。最简单的方法是在系统提供服务的时候封锁中断。但快速反应是实时操作系统的目标,封锁中断显然不是一种好的办法。Nucleus Plus 将 ISR 共分为低级中断处理模块(Low-level ISR, LISR)和高级中断处理模块(High-level ISR, HISR)。LISR 访问系统服务少,而且可以快速完成,所以不需封锁中断。HISR 可以完成较多功能,需要系统服务多,占用 CPU 时间长,但可以被抢占(Preemption)或被系统调度。因为不依靠锁中断来防止多个 ISR 同时访问 Nucleus PLUS 内部数据结构,所以 Nucleus PLUS 内核的中断延迟小而确定。

3 结束语

Nucleus Plus 是一种实时、多任务的嵌入式操作系统。它不但管理各种系统资源、调度任务运行,还为软件开发提供了一个方便的虚拟机界面,为实时任务开发打下了一个良好的基础。使用实时、多任务的嵌入式操作系统在实用开发中起到省时、省力、提高效率的作用,成为嵌入式应用的潮流和方向。

参考文献:

[1] Nucleus Plus Reference Manual[Z]. Accelerated Technology Inc., 1998.

[2] Nucleus Plus Internals Manual[Z]. Accelerated Technology Inc., 1998.

[3] 操作系统:设计与实现(第二版). Andrews[M]. 北京:电子工业出版社, Tanenbaum, Alberts Woodhull, 1998.

Nucleus Plus: an Embedded Real-Time Operationg System

HE Lei

Abstract: This article has introduced the basic content of ATI company's embedded RTOS: Nucleus Plus, and some typical data structures, ideas, methods of the system kernel.

Key words: embedded real-time operating system; RTOS; kernel; task; task schedule; task communication; task synchronization