

$\mu\text{C}/\text{OS}-\text{II}$ 内核扩展接口的低功耗模式

清华大学 陈萌萌

摘要

在嵌入式实时操作系统中,如何在操作系统层面尽量降低系统功耗已成为一个值得研究的问题。本文以嵌入式实时操作系统 $\mu\text{C}/\text{OS}-\text{II}$ 为例,以飞思卡尔 8 位单片机 HCS08GT60 作为硬件平台,详细讨论如何实现一个低功耗的实时操作系统,如何利用 $\mu\text{C}/\text{OS}-\text{II}$ 内核扩展接口省电;详细分析如何选择一种合适的单片机低功耗模式,说明利用 $\mu\text{C}/\text{OS}-\text{II}$ 内核扩展接口实现一个低功耗系统的可行性。

关键词 $\mu\text{C}/\text{OS}-\text{II}$ 内核扩展接口 HCS08GT60 低功耗模式

引言

随着消费类电子产品的功能日益复杂,在其中移植或固化实时操作系统已不是新鲜事了,如手机、PDA 等等。对于该类产品,低功耗特性往往占有举足轻重的地位。如何在操作系统层面上,尽量降低系统功耗,是一个值得探讨的问题。一般来说,嵌入式 CPU 都具有低功耗的工作模式,如果在任务调度的空闲时间,使 CPU 进入这种模式,就能大幅度降低系统功耗。

本文以嵌入式实时操作系统 $\mu\text{C}/\text{OS}-\text{II}$ 在飞思卡尔 8 位单片机 HCS08GT60 上的移植为例,详细讨论如何利用 $\mu\text{C}/\text{OS}-\text{II}$ 给出的内核扩展接口,实现一个低功耗的嵌入式实时系统;进一步分析如何选择一种合适的低功耗模式。

$\mu\text{C}/\text{OS}-\text{II}$ 是一种可移植、可固化、可裁剪的可剥夺型多任务内核。由于其源码公开、注释详尽、内核设计概念清晰,已成为世界上学习和使用频率较高的实时操作系统。2000 年 7 月, $\mu\text{C}/\text{OS}-\text{II}$ V2.52 通过了美国航空航天管理局的安全认证,其可靠性得到了进一步的验证。

利用任务调度的空闲时间使 CPU 进入低功耗模式,以降低系统功耗这一思想在 $\mu\text{C}/\text{OS}-\text{II}$ 内核设计之初就被注意到了。为此设计者特意留出了相应的内核扩展接口。用户可以利用此接口,实现一个实时的低功耗系统。

1 利用空闲任务扩展接口使 CPU 进入低功耗模式

实现 $\mu\text{C}/\text{OS}-\text{II}$ 低功耗特性的方法很简单:用户可以利用 $\mu\text{C}/\text{OS}-\text{II}$ 中空闲任务的扩展接口,使系统在空闲状态下进入某种低功耗模式,降低系统功耗;同时利用 RTI 信号作为时钟节拍,周期性地唤醒 CPU。CPU 被唤醒之

后,将执行节拍中断服务程序,重新判断是否有任务处于就绪态。如果有,就执行该任务;如果没有,则重复上面的过程。

$\mu\text{C}/\text{OS}-\text{II}$ 最多可以管理 64 个任务,并为每一个任务分配一个不同的优先级。每一个任务有五种可能的状态——睡眠态、就绪态、运行态、等待态和中断服务态。 $\mu\text{C}/\text{OS}-\text{II}$ 属于可剥夺型内核,也就是说, $\mu\text{C}/\text{OS}-\text{II}$ 总是运行进入就绪状态的优先级最高的任务。一旦优先级高的任务进入就绪态,就可以将 CPU 从低优先级任务中抢过来。

在 $\mu\text{C}/\text{OS}-\text{II}$ 初始化时,会建立一个优先级最低的任务——空闲任务,在没有任务进入就绪态的时候,空闲任务就会开始运行。空闲任务会调用一个函数——OSTaskIdleHook()。这是留给用户使用的内核扩展接口。空闲任务实际上并没有什么事情可做,只是一个等待中断的无限循环。因此用户可以利用 OSTaskIdleHook(),使 CPU 进入低功耗模式。

用户不必担心整个内核因为系统进入低功耗模式而停止运行。因为 HCS08GT60 允许 RTI 时钟周期性地使 CPU 唤醒。唤醒之后的系统会和遇到节拍中断一样,进入 OSTickISR() 中断服务程序,查看是否有任务进入了就绪态。如果还没有,就再次进入低功耗模式。

对于 HCS08GT60,允许 RTI 时钟的低功耗模式有 WAIT 模式、STOP2 模式和 STOP3 模式三种,其功耗、系统恢复时间、唤醒中断源等各不相同。下面介绍如何选择一种合适的低功耗模式。

事实上,空闲任务可以为统计任务提供一个计数,用以统计 CPU 的利用率,但该工作完全可以在改动 OSTaskIdleHook() 之前运行。



2 选择合适的低功耗模式

2.1 HCS08GT60 的低功耗模式

考虑到后面的讨论要涉及到具体的低功耗模式,所以首先介绍一下单片机 HCS08GT60 的低功耗特性。HCS08GT60 属于飞思卡尔(原 Motorola) HCS08 系列单片机。该系列单片机的低功耗特性很突出;工作电压可以在 1.8~3.6 V 之间选择,有 WAIT 和 STOP 两种低功耗模式。STOP 模式可细分为 STOP3、STOP2 和 STOP1 三种,功耗依次降低。WAIT 模式下,CPU 停止运行,但其他外围模块并不断电,因此,系统随时可以响应各种中断。HCS08GT60 的三种 STOP 模式如表 1 所列。

表 1 三种 STOP 子模式的特点

模式	CPU/Flash	RAM	ICG	ATD	KBI	Regulator	I/O	RTI
STOP1	关闭	关闭	关闭	禁用	关闭	关闭	复位值	关闭
STOP2	关闭	备用	关闭	禁用	关闭	备用	锁存	可选
STOP3	备用	备用	备用	禁用	可选	备用	锁存	可选

从表 1 可以看出,在 STOP1 模式中,唤醒 CPU 只能通过 IRQ 中断或复位信号,由于无法提供时钟节拍,内核的任务调度无法实现;而在 STOP2 和 STOP3 中,RTI 都可以作为系统的唤醒中断源,内核可以使用 RTI 作为时钟节拍。

STOP2 模式与 STOP3 模式相比功耗更低;但是,STOP2 模式下 I/O 寄存器是关闭的,必须在进入模式之前将 I/O 寄存器的值保存在 RAM 中,而在唤醒之后再从 RAM 拷贝到 I/O 寄存器。唤醒 STOP2 可以使用 IRQ、复位信号和 RTI。STOP3 模式下,RAM 和 I/O 寄存器内容将保持。另外,除 STOP2 模式允许的唤醒中断源外,还允许键盘中断唤醒 CPU。

2.2 实时性、中断源和功耗

影响低功耗模式的选择有三个主要因素:功耗、中断源和实时性。

(1) 功耗

前文中已经提到,适用于 $\mu\text{C}/\text{OS}-\text{II}$ 的低功耗模式(即允许 RTI 唤醒)有三种:WAIT 模式、STOP3 模式和 STOP2 模式。系统在这三种模式下的功耗逐渐降低。表 2 列出了 3.12 V 供电下,三种模式的典型功耗。

表 2 STOP2、STOP3 和 WAIT 模式下的功耗

模式	系统消耗电流/ μA
STOP2(使用 RTI)	0.89
STOP3(使用 RTI)	1.1
STOP3(使用以 32kHz 晶振为时钟源的 RTI)	14.5
WAIT(1MHz 总线时钟)	560

$\mu\text{C}/\text{OS}-\text{II}$ 为用户提供了一个统计任务,用以计算 CPU 的利用率,并保存在变量 OSCPUUsage(%) 中。用户可以在加入低功耗处理前,使用统计任务计算出 CPU 利用率,从而粗略地估算出系统的功耗。

假设系统正常运行时,消耗电流为 1 mA,CPU 利用率是 1%,则以下是选择三种不同低功耗模式后的消耗电流。

STOP2: $1 \text{ mA} \times 1\% + 890 \text{ nA} \times 99\% = 10.881 \mu\text{A}$, 系统功耗降低 98.9%。

STOP3: $1 \text{ mA} \times 1\% + 14.5 \mu\text{A} \times 99\% = 24.355 \mu\text{A}$, 系统功耗降低 97.6%。

WAIT: $1 \text{ mA} \times 1\% + 560 \mu\text{A} \times 99\% = 564.4 \mu\text{A}$, 系统功耗降低 43.6%。

系统功耗当然越小越好,但当考虑到其他因素时,系统功耗就未必能够达到最低了。

(2) 中断源

系统用到的中断源限制了低功耗模式的使用。为了保证 $\mu\text{C}/\text{OS}-\text{II}$ 正常运行,系统所用到的中断必须能够唤醒处于低功耗模式下的 CPU。

WAIT 模式虽然功耗较大,但能够响应任何中断源;STOP3 模式下,系统保留了 RTI、IRQ、KBI 和复位作为唤醒中断;而在 STOP2 模式下,只有 IRQ、复位和 RTI 可以唤醒系统。

(3) 实时性

毫无疑问,使 CPU 进入低功耗模式会减弱系统的实时性。这种减弱来自于两个方面,一是使中断响应时间变长;二是使响应的的时间变得不易预测。

当系统从低功耗模式中被唤醒后,时钟往往需要一段时间稳定,有时候还需要软件做内核运行环境的恢复工作(如 STOP2 下的寄存器恢复),中断的响应时间就被拉长了。同时,由于时钟恢复的时间和供电电压、时钟源、环境温度都有密切的关系,实际上不可能给出一个准确的恢复时间,中断响应的的时间也就变得不易预测了。在实时系统中,响应时间的不可预测往往比响应得慢更为致命,一个响应速率时快时慢的系统只能以最坏的情况作估计。所幸的是,大多数低功耗应用(如手机、PDA 等)都不是硬实时系统,换句话说,并没有一个绝对的响应时间限制。大多数情况下,采用低功耗处理所带来的实时性减弱可以被忍受。

WAIT 模式对响应时间影响最小。由于没有停止系统时钟,WAIT 模式对中断的响应基本都是同步的。

计算必须在改动 OSTaskIdleHook() 之前进行,因为一旦系统进入任何一种低功耗模式,空闲任务将不能给变量 OSIdleCtr 继续加 1。使用以 32 kHz 晶振为时钟源的 RTI。

STOP3 模式恢复的时间和时钟设置关系很大。除了 FBE 时钟方案外(使用外时钟、不使用锁相环),恢复时间都在 100 μ s 左右。如果采用 FBE,恢复时间就和晶振频率密切相关了。一般 32 kHz 晶振需要 180 ~ 300 ms 恢复稳定,假如在 STOP3 模式下将晶振保持打开,则只需要 2.42 ms。

STOP2 模式的恢复时间在 50 μ s 左右。但是,因为需要将在 RAM 中保存的 I/O 寄存器恢复,可能另外还需要几十个指令周期。

表面上 STOP2 的恢复时间比 STOP3 的恢复时间短,但是考虑到进入 STOP2 之后 RTI 时钟源会从外部晶振调整为内部晶振,最多可能与实际系统相差 1 个时钟节拍。

3 μ C/OS-II 在 HCS08GT60 上的移植

μ C/OS-II 的 95% 代码是由 ANSI C 写成的,具有很好的移植性。如何移植 μ C/OS-II 可以参阅文献[1]。这里只强调一下时钟节拍的选择。

为了实现时间延时和确认超时, μ C/OS-II 需要系统提供一个 10 ~ 100 Hz 的周期性信号。我们选择实时时钟中断(RTI)作为 μ C/OS-II 的时钟。这主要是考虑在 HCS08GT60 处于 WAIT 或者 STOP2/3 模式下,RTI 仍然可以作为唤醒系统的中断源。需要注意,在运行和等待模式下,RTI 的时钟只能由外部晶振提供;在 STOP3 模式下,RTI 时钟可以由外部晶振或是内部晶振提供;在 STOP2 模式下,RTI 只能由内部时钟提供。为了尽量不改动时钟源,建议使用 1 个 32.768 kHz 的外部晶振提供系统时钟和 RTI 时钟,在运行、WAIT 和 STOP3 模式下,RTI 的时钟源始终不变;而在 STOP2 模式下,用户只能使

用内部时钟发生器提供的 RTI 时钟源。

4 结论

仔细分析 1 个低功耗实时系统会发现,有很多因素左右着系统功耗,各因素之间往往会相互影响,相互制约。例如,为了保证实时性,尽量不改动时钟设置,使用了 32.768 kHz 的外部晶振作为 RTI 时钟源,并利用锁相环将该频率升高,作为系统总线时钟。从操作系统角度分析,CPU 可以进入低功耗模式,系统功耗降低了。但是,因为使用了锁相环,也会给系统带来额外的功耗。对于一个实际系统,这种做法到底是提高还是降低了系统功耗,只能通过 CPU 占用率、节拍频率等条件具体分析了。

因此,要选择一套合理的软硬件设置来降低功耗,就必须全盘考虑,不能仅仅局限于操作系统的角度。

参考文献

- 1 Labrosse Jean J. 嵌入式实时操作系统 μ C/OS-II. 邵贝贝译. 第 2 版. 北京:北京航空航天大学出版社,2003
- 2 刘慧银,程建平,龚光华,等. Motorola (Freescale) 微控制器 MC68HC08 原理及其嵌入式应用,北京:清华大学出版社,2005
- 3 邵贝贝. 单片机嵌入式应用的在线开发方法. 北京:清华大学出版社,2004
- 4 Donnie Garcia, Scott Pape, MC9S08GB/GT Low-Power Modes. Freescale Semiconductor, Rev 2, 2004-11
- 5 Freescale Semiconductor. MC9S08GB/GT Data Sheet. Rev 2.2, 2004
- 6 Freescale Semiconductor. HCS08 Family Reference Manual. 2003

(收稿日期:2005-09-08)

以上恢复时间均为实测值。

缩 略 语

ADO	ActiveX Data Object	ActiveX 数据对象
CMPP	China Mobile Peer to Peer	中国移动点对点协议
DBB	Digital Basic Blocks	数字基本模块
DCB	Digital Communication Blocks	数字通信模块
DDR	Double Data Rate	双数据速率
ECU	Electronic Control Unit	电子控制单元
FACCH	Fast Associated Control Channel	快速相关控制信道
IVR	Interactive Voice Response	互动式语音应答
ME	Mobile Equipment	移动设备
MFC	Microsoft Foundation Class Library	Microsoft 基本类库
MSC	Mobile Service switching Center	移动业务交换中心
OLED	Organic Light Emitting Display	有机发光显示
SMPP	Short Message Peer to Peer	短信点对点
SMSC	Short Message Service Center	短信服务中心
SP	Service Provider	服务提供商
TA	Terminal Adapter	终端适配器
TE	Terminal Equipment	终端设备
USSD	Unstructured Supplementary Service Data	非结构化补充数据