μC/OS-II 在倒立摆控制系统中的应用

杨兴明[1],孙 锐[1],赵 鹏[2],张培仁[2]
YANG Xing-ming[1],SUN Rui[1],ZHAO Peng[2],ZHANG Pei-ren[2]

1.合肥工业大学 计算机与信息学院,合肥 230009
2.中国科技大学 自动化系,合肥 230026
1.School of Computer & Information,Hefei University of Technology,Hefei 230009,China
2.Department of Automation,University of Science & Technology of China,Hefei 230026,China
E-mail:yxm@ustc.edu

YANG Xing-ming,SUN Rui,ZHAO Peng,et al.Application of μC/OS-II in inverted pendulum control system.Computer Engineering and Applications,2009,45(22):59-61.

Abstract: Pendulum is a rapid response system and a classic problem in the field of control theory research,it requests real-time control.μC/OS-II is a multi-task real-time operating system,some urgent event can be disposed by it.μC/OS-II is used as operating system to control inverted pendulum in this paper,and good character is achieved.
Key words: μC/OS-II;pendulum;TMS320LF2407

摘 要:倒立摆是一个快速响应的控制系统,是控制理论研究中的经典问题,要求能够快速执行控制;μC/OS-II 是一个多任务实时操作系统,能够对紧要任务给予快速的处理。采用 μC/OS-II 作为操作系统,对倒立摆进行控制,取得了良好控制效果。
关键词:μC/OS-II;倒立摆;TMS320LF2407
DOI:10.3778/j.issn.1002-8331.2009.22.020 **文章编号**:1002-8331(2009)22-0059-03 **文献标识码**:A **中图分类号**:TP316

1 引言

在嵌入式系统软件开发中，多采用前后台的设计模式,即应用程序是一个无限的循环,循环中按一定的顺序调用函数完成相应的任务;中断服务程序处理异步事件。在相对简单的应用中这种模式可以胜任,而对于实时性要求较高、处理任务多的应用,就可能达不到应用要求,且系统稳定性差、可靠性低。引入实时操作系统,可以较好解决这个问题。

倒立摆是控制理论研究中的经典问题,它是一个快速响应控制系统,要求能够快速执行控制,系统采样、控制周期约 10 ms,对任务的执行具有较高的实时性,其结构图如图 1。

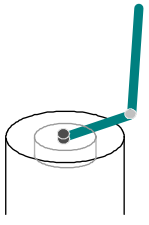

图 1 倒立摆结构图

μC/OS-II 是目前一个流行的实时操作系统[1],该内核是源代码公开的、占先式、嵌入式实时内核,并且简单,易于使用,可根据应用裁减、具有良好的稳定性和可靠性,大部分代码用 C 语言写成,容易移植。

论文针对倒立摆控制对任务执行实时性要求高的特点,提出应用 μC/OS-II 作为软件设计平台，在实际的实验中取得了良好的控制效果。

2 移植

倒立摆系统采用 TI 公司的电机控制专用 DSP——TMS320LF2407 作为核心处理器,所以,首先需将 μC/OS-II 移植到 2407 中。

对 μC/OS-II 的移植[2],主要包括对数据类型的定义,堆栈空间的定义和初始化,开关中断指令的定义和初始化,在任务切换或中断时保存和恢复任务信息,以及一些接口函数的定义。

2.1 对数据类型的定义

TI 公司的 2407 是 16 位的[3],移植时,需在文件 OS_CPU.H 中,根据编译器对数据类型的定义给出实际的数据类型定义。

2.2 堆栈的定义和初始化

在文件 OS_CPU.H 中定义堆栈入口宽度为 16 位，地址增长方向为向上递增；在文件 OS_CPU_C.C 中，改写函数 OSTaskStkInit,根据需要保存的数据量,初始化堆栈。

2.3 开关中断指令

2407 的关中断指令[4]为 asm("setc intm"),开中断指令为

基金项目:安徽省"十五"二期科技攻关项目(No.04012501)。
作者简介:杨兴明(1977-),男,讲师,博士,主要从事计算机控制、小波信号处理等研究。
收稿日期:2008-06-25 **修回日期**:2008-10-07

asm("clrc intm"),开关中断方式为方式 1。由于编译器支持嵌入式汇编指令,所以可用嵌入式汇编代码改写 OS_CPU.H 中对应的宏定义。

2.4 保存和恢复任务信息相关内容

在任务切换时,需保存将被挂起任务信息,恢复即将运行任务信息到 CPU 寄存器;在中断发生时,需保存被打断任务信息,直到中断返回时,将信息恢复[5]。μC/OS-II 中相关的函数和内容如下:

在 OS_CPU.H 中定义宏 OS_TASK_SW(),它用于引起任务切换,是一个软件中断。中断服务程序实现任务的切换。

在 OS_CPU_A.ASM 中,用汇编语言改写函数 OSCtxSw,它是 OS_TASK_SW()对应软中断的中断服务程序,主要用于保存将挂起任务的信息,然后恢复即将运行任务信息。由于 TI 公司的编译器 CCS 有自动保存被中断信息的库函数 I$$SAVE,所以可直接调用该函数实现保存信息的操作[6]。需要注意的是,在调用后,要将用于保存信息的堆栈指针指向即将挂起任务堆栈;恢复即将运行任务信息也可调用库函数 I$$REST 实现,调用前,需将即将运行任务堆栈指针放入 CPU 的寻址寄存器中。OSCtxSw 的实现代码如下[7]:

```
_OSCtxSw:
    call  I$$SAVE          ;调用 I$$SAVE 保存当前任务信息
    ;将 I$$SAVE 保存数据后的栈顶保存到当前任务 TCB 堆栈指针,AR1 中保存
    ; 着 I$$SAVE 保存数据后的栈顶;AR3 保存着当前优先级任务 TCB 堆栈指针
      ldpk  _OSTCBCur
      lar  ar3,_OSTCBCur        ;将当前任务 TCB 地址装载到 AR3
      mar  *,ar3                ;以 AR3 内容为数据存储器地址
      sar  ar1,*,ar1            ;将 AR1 内容保存到数据存储器中
      call  _OSTaskSwHook
      ldpk  _OSTCBHighRdy
      lar  ar1,_OSTCBHighRdy;最高优先级任务 TCB 地址装载到 AR1
      sar  ar1,_OSTCBCur    ; 将 AR1 内容保存到 OSTCBCur 指向的数据存储单元
          ;更改当前任务优先级为即将开始运行任务的优先级[8]
      lacl  _OSPrioHighRdy
      sacl  _OSPrioCur
      lar  ar1,*                ;装载最高优先级任务 TCB 地址到 AR1
      b  I$$REST,ar1            ;调用 I$$REST 从即将运行的任务堆栈中恢复寄存器
```

启动任务时调用的 OSStartHighRdy,也在 OS_CPU_A.ASM 中,需用汇编语言编写。它主要是在系统刚启动时,使就绪任务中最高优先级任务运行,实质上是将最高优先级任务信息的恢复到 CPU 寄存器,操作同 OSCtxSw 的恢复任务部分[6]。

中断任务切换函数 OSIntCtxSw,在 OS_CPU_A.ASM 中,用汇编语言编写。该函数的内容同 OSCtxSw 大部分相同,只是由于中断时保存了任务信息,在 OSIntCtxSw 中,不需要再次保存。

2.5 中断服务程序

时钟节拍中断服务程序 OSTickISR,用于为应用提供一个周期性的时钟节拍。在中断服务程序中,先调用 I$$SAVE 保存被中断任务的信息到对应的堆栈中,然后清除中断标志,并调用 OSTimeTick 函数对系统中各任务延时时间进行更新,OSIntExit 查看是否需要作任务切换,最后调用 I$$REST 返回。

2.6 各个接口函数

为减少应用中的代码量,没有使用 μC/OS-II 提供的各个接口函数,它们都被定义为空函数。

3 应用

采用 μC/OS-Ⅱ作为操作系统时,底层软件在完成初始化工作后,将控制权交给操作系统,在系统运行过程中,由操作系统根据任务状态,对各个任务进行调度和管理,以实现重要操作的实时性。

倒立摆控制中,主要执行的任务有:对转杆、摆杆位置的采样,并进行滤波,按位置差分方法求相应的角速度;根据控制算法计算控制量并输出;由于倒立摆的位移是相对于一个初始零点测量的,为避免零点漂移对控制的影响,系统有一个调零操作,对初始位置进行校正;此外在控制过程中,系统还需将同上位机保持通信,将控制中的一些数据发给上位机以用于分析控制效果,并从上位机接收控制命令。所以,系统共有 5 个任务:采样(包括滤波和求角速度)、控制、系统调零、发送和接收数据。

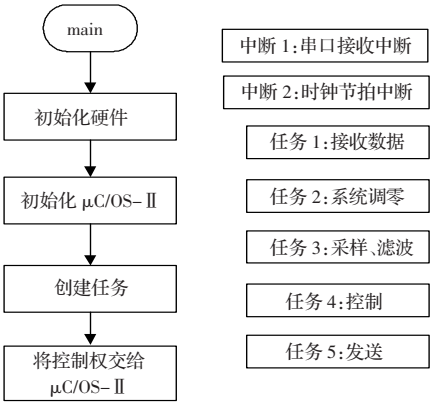

图 2 应用 μC/OS-Ⅱ时倒立摆控制程序流程

接收数据是紧要事件,所以接收数据任务的优先级最高,五个任务优先级为:

接收任务>调零任务>采样任务>控制任务>发送任务

调零操作时,需使倒立摆系统保持静止状态,从而不能执行控制任务,此外调零任务时也不需要采样,所以调零任务和采样、控制任务不能同时就绪,通过在接收任务中收到控制命令时,挂起可能同目标操作有冲突的任务,来处理冲突:收到调零命令时,挂起采样和控制任务,使调零任务就绪,从而在调零操作时,不会有采样操作和控制输出;收到控制命令时,挂起调零任务,使采样和控制任务就绪,执行正常的控制操作。

系统共有两个外部中断,定时节拍中断,发生频率为 100 Hz,为系统提供计时标准;串口接收中断,接收上位机发送来的命令。系统第一个执行的任务为接收数据任务,时钟节拍中断在接收任务中启动。

接收数据任务创建后,一直等待串口通信事件发生。当有串口通信事件时,串口中断服务子程序向数据接收任务发送消息,该任务收到消息后,恢复运行,根据收到的命令进行处理,处理完本次操作后,挂起自己,等待下一个串口通信事件发生。

为简单起见,各个任务间的数据传递由全局变量实现。数据接收任务接收到的控制量或控制参数保存到全局变量中,由控制任务从该全局变量读取;采样任务和校正任务得到的位置

值也放到全局变量中，控制任务和发送任务从对应全局变量中取数据。

4　实验

在倒立摆控制中采用 μC/OS-Ⅱ作为操作系统，采样、控制周期设为 10 ms，采用 LQR 算法进行控制实验，控制效果如图 3~图 5。

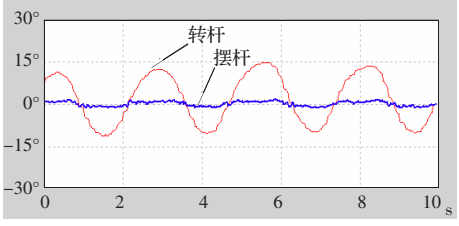

图 3　控制中转动角度变化曲线

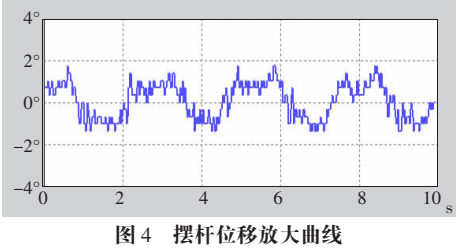

图 4　摆杆位移放大曲线

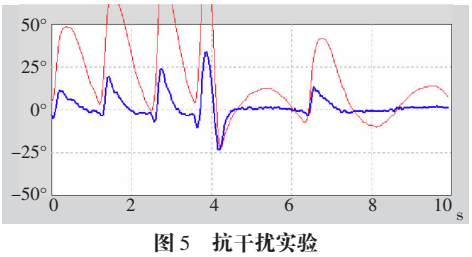

图 5　抗干扰实验

控制过程中，摆杆角位移偏离平衡位置 2°以内，转杆位移也在以内，取得了良好的控制效果。为检测系统受干扰时响应速度，验证系统抗干扰性能，对摆杆施加冲击，从图 5 中可见，在受到干扰的时候，系统仍能有效实现控制，可见，系统具有良好的响应速度，能够对突发的情况快速处理。

5　结论

对于倒立摆这种要求快速进行控制的设备，μC/OS-Ⅱ具有实时、快速响应的特点，能够满足系统对实时性的要求，并在实验中取得了良好的控制效果，此外，采用操作系统后，增加了系统稳定性和可靠性，软件实现也变得容易。

参考文献：

[1] Labrosse J J.嵌入式实时操作系统 μC/OS-Ⅱ[M].邵贝贝，译.北京：北京航空航天大学出版社，2003.
[2] 赵鹏.基于 DSP 的环形倒立摆系统设计与研究[D].合肥：中国科学技术大学，2007-05.
[3] 徐科军.TMS320LF/LC24 系列 DSP 指令和编程工具[M].北京：清华大学出版社，2004.
[4] 徐科军.TMS320LF/LC24 系列 DSP 的 CPU 与外设[M].北京：清华大学出版社，2004.
[5] 张小芳，李向华，陆起涌.uC/OS-Ⅱ在仪器仪表中的应用[J].仪表学报，2003(S1).
[6] 李纪奎，向怀坤，胡泓，等.基于 uCOS-II 的视频动态交通信息采集系统研究[J].哈尔滨工业大学学报，2007(2).
[7] 戴琪华，戴曙光，穆平安.uCOS 实时响应问题的解决方法[J].上海理工大学学报，2002(1).
[8] 钟汉如，郭建飞.基于 uC/OS 无刷直流电机控制的研究[J].微计算机信息，2007(2)：49-51.

(上接 48 页)

程序编写，并导出其函数符号，供虚拟网络接口的驱动程序调用。本设计中采用第二种设计方法。

6　总结

实现虚拟网络驱动程序及 Netlink 程序后，即可按照以下步骤进行系统的部署与测试，并对结果进行分析。

(1)由于虚拟网络接口的驱动中需要使用到 Netlink 内核态的函数，所以应该首先以内核模块的方式编译 Netlink 程序，成功后使用 insmod 进行该模块的加载，可以使用 lsmod 查看模块的加载情况。编译 Netlink 的用户态程序并运行，并单独测试 Netlink 的用户态程序与内核程序通信是否正常。

(2)以内核模块方式将虚拟网络驱动程序编译成 Linux 的内核模块，使用 insmod 进行网络驱动模块的加载。当网络驱动模块加载成功后，则会在 Linux 内核中增加了一个虚拟的网络接口 eth1，使用 ifconfig 命令对该网络接口进行配置，包括 IP 地址、子网掩码、MAC 地址等等参数，最后使用 ping 命令测试该虚拟网络接口是否正常工作。

(3)打开串口用户程序，测试其与 Netlink 用户态程序之间的通信是否正常。

(4)整体测试，按照系统设计的模型，在各个 IDU 上运行路由软件 Zebra，进行路由信息发送测试，分析出现的问题，进行进一步的修正。

测试结果表明，该系统对设备数量较少的网络进行网络管理时，能够满足实际需求。当站点数量很多时，由于每个站点上都需要运行路由协议用来跟踪网络的拓扑结构，大量数据需要通过带宽较低的串口进行转发，导致系统性能下降。另外，在进行基于串口的网络数据转发的过程中，网络数据经历了从用户态到内核态或者从内核态到用户态的数据通信，在一定程度上影响了系统的性能，增加了系统开销，也需要进行进一步的优化。

参考文献：

[1] 李驹光.ARM 应用系统开发详解[M].北京：清华大学出版社，2004.
[2] Yaghmour K.Building embedded Linux systems[M].New York：O'Reilly & Associates，2003.
[3] Rubini A，Corbet J.Linux device drivers[M].2nd ed.New York：O'Reilly & Associates，2001.
[4] Salim J.RFC3549 Linux Netlink as an IP services protocol[S].2003.