多硬件中断环境下的 μC/OS-Ⅱ操作系统的移植与驱动开发*

刘世隆[1] 祖家奎[1] 黄 海[2] 缪克钻[2]
(1. 南京航空航天大学 自动化学院 南京 211106;2. 嘉兴中创航空技术有限公司 嘉兴 314000)

摘 要: 以 ARM 处理器和 μC/OS-Ⅱ操作系统为核心的嵌入式技术在低成本、微小型等实时系统中得到了广泛应用。本文的研究对象是微小型无人直升机飞行控制系统普遍采用的 LPC2468 微控制器,应用软件设计基于 μC/OS-Ⅱ操作系统,针对飞行控制软件在多硬件中断和 μC/OS-Ⅱ多任务管理调度相结合的运行机制下,为满足工程应用的系统稳定性和可靠性要求,展开了 μC/OS-Ⅱ操作系统移植、相关驱动开发以及飞行控制软件结构设计等相关研究,最后对整个飞行控制系统进行了测试和验证。

关键词: 无人直升机;飞行控制系统;μC/OS-Ⅱ操作系统;多硬件中断;驱动开发
中图分类号: TP242 **文献标识码**: A **国家标准学科分类代码**: 590.35

Transplantation and driver development of μC/OS-Ⅱ operating system in multi-hardware interrupt environment

Liu Shilong[1] Zu Jiakui[1] Huang Hai[2] Miao Kezuan[2]
(1. College of Automation Engineering, Nanjing University of Aeronautics and Astronautics, Nanjing 210016, China;
2. Jiaxing Zhongchuang Aviation Technology Company limited, Jiaxing 314000, China)

Abstract: ARM processor and μC/OS-Ⅱ operating system as the core of embedded technology has been widely applied in low cost and micro real-time system. In the paper, the research object of micro unmanned helicopter flight control system using LPC2468 micro controller, application soft design based on the μC/OS-Ⅱ operating system. For multi-hardware interrupt and μC/OS-Ⅱ many task management scheduling mechanism combing the running of flight control software. To meet the requirements of the system stability and reliability of the engineering application, launches the corresponding research of μC/OS-Ⅱ system transplantation driver development and flight control software structure design. Finally, we tested the whole system.

Keywords: unmanned helicopter, flight control system, μC/OS-Ⅱ operating system, multi-hardware interrupt, driver development

1 引 言

当前,以 ARM 处理器和 μC/OS-Ⅱ操作系统为核心的嵌入式技术在低成本、微小型等实时系统中得到了广泛应用。其中,ARM 处理器作为片上系统,其接口多且资源丰富,用户不需要做复杂的接口变换;而 μC/OS-Ⅱ是一款源代码开放,且可多平台移植、可固化、可裁剪的多任务实时内核。因此,这两者的技术结合和拓展可以满足众多系统级应用的需求。

本文的研究对象是某微小型无人直升机飞行控制系统,其系统构架如图 1 所示。该系统的飞控机采用 ARM7TDMI-S 为内核的 LPC2468 为核心处理器,应用软件是在 μC/OS-Ⅱ操作系统之上开发的,外部接口以 PWM 信号和串口信号为主。鉴于微小型飞行器机动性强、操控的实时性要求高以及低成本硬件约束等因素,飞控软件是在多硬件中断和 μC/OS-Ⅱ多任务管理调度相结合的机制下运行。为了保证系统运行的稳定性和可靠性,本文针对 ARM 系统多硬件中断环境下的 μC/OS-Ⅱ操作系统的移植和驱动问题展开研究,具体包括多中断环境下的 μC/OS-Ⅱ的移植方法、底层驱动技术、移植和驱动的特殊处理方法、飞控软件设计结构以及系统的测试与验证结果等。

收稿日期:2015-08
* 基金项目:南京航空航天大学基本科研业务费专项科研(NS2013031)资助项目

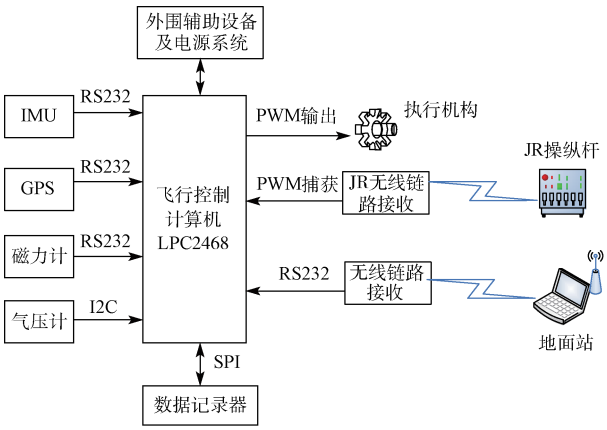

图 1　微小型无人直升机的飞控系统架构图

2　技术需求与解决方法

在微小型无人直升机飞行控制系统中，需要实时地采集操纵杆信号、传感器数据等，然后通过控制律解算，将舵面控制信息传给无人直升机的各个舵机，从而实现控制。在这个过程中，需要同时对多路信号源的数据进行实时采集处理，对飞行器的导航、制导和控制回路的多个控制律解算任务进行调度，因此操控的实时性以及系统的稳定性是微小型无人直升机飞行控制系统在设计中最优考虑的两个因素[1]。

针对上述的两个关键因素，在设计飞控软件时采用多硬件中断和 μC/OS-Ⅱ多任务管理调度相结合的设计思想和运行机制。中断能够大大地提高 CPU 的利用率，而任务的合理调度能够有效地管理整个系统，从而最大满足微小型无人直升机飞行控制系统的技术需求。本系统中所涉及的外部硬件中断处理主要是操纵杆 PWM 信号的实时捕获、串口通信和 SPI 通信的底层处理；而基于 μC/OS-Ⅱ系统的多任务调度处理主要是数据的接收发送处理、传感器数据的融合、控制律的解算以及导航解算等。

3　多中断环境下的 μC/OS-Ⅱ移植

3.1　μC/OS-Ⅱ内核结构与移植方法

要使 μC/OS-Ⅱ在处理器中正常运行，必须满足以下要求[2]：1）处理器的 C 编译器能产生可重入型代码；2）处理器支持中断，并且能产生定时中断（通常为 10～100 Hz）；3）用 C 语言就可以开/关中断；4）处理器能支持一定数量的数据存储硬件堆栈（可能是几千字节）；5）处理器有将堆栈指针以及其他 CPU 寄存器的内容读出、并存储到堆栈或内存中去的指令。本文中所使用的是以 ARM7TDMI-S 处理器为内核的 LPC2468 微控制器，符合以上的移植要求。

μC/OS-Ⅱ的移植模块主要分为 3 部分：1）与处理器无关的代码部分；2）与应用相关的部分；3）与处理器相关的代码部分。其中，第一部分和第二部分只需根据应用程序的要求，略加修改，μC/OS-Ⅱ的移植的重点在与和处理器相关的三个文件：OS_CPU. H、OS_CPU_A. ASM 和 OS_CPU_C. C，需要根据处理器的不同进行相应的修改。

3.2　实时系统节拍与多中断定义

时钟节拍（clock tick）是特定的周期性中断，这种节拍式中断可以为内核任务延时提供 μC/OS-Ⅱ在 ARM 的移植过程中，时钟节拍是通过定时器 0 的中断产生的，这个过程有 3 步实现[3]：

1）在 IRQ. S 文件中添加程序：

Timer0_Handler HANDLER Timer0_Exception

以此来增加对汇编语言接口的支持；

2）对向量中断控制器进行初始化，安装定时器 0 的中断；

3）对定时器 0 进行初始化操作，设置时钟节拍的周期，然后 ARM 就会周期性地调用定时器 0 的中断服务子程序 OSTickISR()，在子程序中调用时钟节拍函数 OSTimeTick()，进行时钟节拍服务。这里我们需要注意的是必须将定时器 0 的中断优先级设置为最高，否则有外部中断有可能打乱时钟节拍，从而导致系统延时不准确，进一步影响任务的调度和运行。

与此同时，还需增加所有使用的中断对汇编语言接口支持的程序，具体如表 1 所示。

表 1　汇编语言支持接口程序

中断名称		中断服务函数
Timer1_Handler	HANDLER	PWMCap_Timer1Exception
Timer2_Handler	HANDLER	PWMCap_Timer2Exception
Timer3_Handler	HANDLER	PWMCap_Timer3Exception
PCAP10_Handler	HANDLER	PWMCap_PCAP10Exception
Uart0_Handler	HANDLER	Com_Uart0Exception
Uart1_Handler	HANDLER	Com_Uart1Exception
Uart2_Handler	HANDLER	Com_Uart2Exception
Uart3_Handler	HANDLER	Com_Uart3Exception
SSP0_Handler	HANDLER	SSP_Com0Exception
SSP1_Handler	HANDLER	SSP_Com1Exception

3.3　中断嵌套及优先级的定义

关于中断嵌套，即对有设定中断优先级的系统，当高优先级的中断源响应中断时，将正在执行的低优先级的中断源暂时终止，先处理高优先级的中断源服务程序，待处理完毕后再重新返回到中断了的中断服务程序中继续执行，这样的一个过程就是中断嵌套。

本系统中所使用的中断、优先级以及相关说明的设定如表 2 所示；中断的优先级取决于该中断对系统的影响程度而设定。

表 2　中断优先级设置

名称	优先级	说明
Time 0	1	产生系统时钟节拍
Time 1	2	通过 3 个定时器中断和一个 PCAP 中断获取 10 路 PWM 捕获通道的脉冲宽度，采集遥控器各个通道的数据
Time 2	3	
Time 3	4	
PCAP	5	
Uart 0	6	通过 4 个串口中断实现对串口通信数据的实时接收和发送，传输 IMU、GPS、磁力计以及地面站的数据
Uart 1	7	
Uart 2	8	
Uart 3	9	
SPI 1	10	通过 2 个 SPI 中断实现数据的实时传输，写入数据到记录仪中
SPI 2	11	

在这里使用定时器 0 作为 μC/OS-Ⅱ的系统时钟节拍，为了使其不被外部的中断打乱将其优先级设置为最高，同时也需要注意在使用外部中断源时要尽量避免使用 FIQ(快速中断请求)，由于 FIQ 具有最高优先级，所以 FIQ 有可能打乱系统的时钟节拍，进而导致一系列的问题。

本系统应用在微小型无人直升机的飞行控制系统中，最重要的就是无人直升机各个舵机的信号输入，故将应用于 PWM 信号捕获的定时器 1、定时器 2、定时器 3 及 PCAP 中断的中断优先级次之。然后是整个系统的数据的串口传输；最后将应用于记录数据的 SPI 中断的优先级设置为最低。在这里需要注意的是，即便可以通过硬件中断优先级来判断软件优先级相同的响应先后顺序，但我们应该在软件中将优先级规划好，避免相同软件优先级情况的出现。

3.4　中断或任务切换的堆栈处理

在 μC/OS-Ⅱ系统中，任务有休眠态、就绪态、运行态、挂起态、被中断态五种状态。在正常的任务切换过程中，堆栈处理如图 2 所示，即将被切换的私有任务堆栈区存储当前 CPU 的寄存器内容，即入栈，进入就绪态；并将即将运行的私有任务堆栈区数据复制到 CPU 寄存器中，即出栈，该任务进入运行态，掌握 CPU 使用权。从而实现任务的切换。在有中断响应情况中，任务的堆栈情况一样，而该任务进入被中断态，若中断结束后，有更高优先级任务进入就绪态，则不再运行原来被中断了的任务。

3.5　中断服务程序

当系统响应硬件中断并保护现场后，将跳转入中断服务程序中，来执行这段特定的程序，对事件进行处理，完成后回到中断之前处理的程序中。中断服务的处理时间应该尽可能地短，即仅将重要并且能够快速完成的工作放在中断服务程序中，而把不太重要的需要花费较长时间完成的工作放在任务中完成。这样能够有效地减少每次中断处理对 CPU 的消耗，进而提高中断响应的速度和 μC/OS-Ⅱ系统的时钟精度。

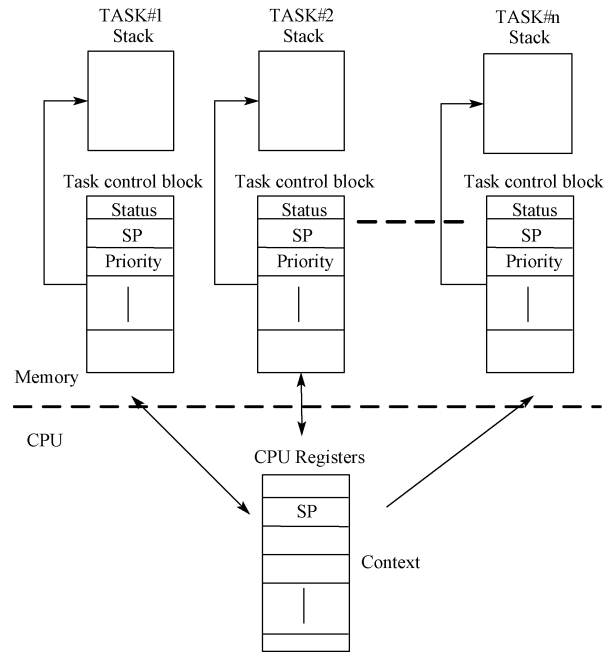

图 2　μC/OS-Ⅱ堆栈处理

中断是在 CPU 有事件发生时才去处理，这就需要微处理器不断的查询是否有事件的发生，而为了减少对 CPU 的消耗，在 μC/OS-Ⅱ系统中，通过 OS_ENTER_CRITICAL (void)(关中断)和 OS_EXIT_CRITICAL(void)(开中断)让 CPU 是否响应中断。需要注意的是关中断的时间要尽可能地短，否则可能导致中断响应的丢失。

3.6　中断与 μC/OS-II 系统的接口实现

在 ARM7 处理器中，用户任务具有两种处理器模式：用户模式和系统模式，可以使用两种指令集：ARM 指令集和 THUMB 指令集，为了使底层接口函数与处理器状态无关，让处理器在不同的模式下均可以不受限制地调用 μC/OS-Ⅱ的底层接口函数，采用 ARM7 的软中断指令 SWI，用关键字_swi 作为前缀来声明一个利用软中断的调用，其格式如下：

_swi(功能号)　返回值类型　名称(参数列表)；

用户可以像调用一个普通函数一样实现软中断的调用，μC/OS-Ⅱ中须用软中断实现的函数及其软中断功能号分配如表 3 所示[4]。

4　多中断环境下的接口驱动开发

4.1　接口驱动的实现方法

接口驱动是微小型无人直升机飞行控制计算机中的数据交换区，负责接收来自外部的传感器数据，同时把内部处理过后的数据传输至执行机构。基于 μC/OS-Ⅱ操作系统，本文通过统一对底层中断和任务实时管理调度，实现具体思路如下：

表 3　须用软中断来实现的函数

功能号	函数名	说明
0x00	void OS_TASK_SW(void)	任务级任务切换函数
0x01	void _OSStartHighRdy(void)	运行优先级最高的任务，由 OSStartHighRdy()产生
0x02	void OS_ENTER_CRITICAL(void)	关中断
0x03	void OS_EXIT_CRITICAL(void)	开中断
0x80	void ChangeToSYSMode(void)	任务切换到系统模式
0x81	void ChangeToUSRMode(void)	任务切换到用户模式
0x82	void TaskIsARM(INT8U prio)	任务代码是 ARM 代码
0x83	void TaskIsTHUMB(INT8U prio)	任务代码是 THUMB 代码

1）飞行控制计算机通过底层中断实时接收来自传感器的数据，存入缓冲区，并置信号量标识；

2）实时操作系统任务根据信号量读取软件缓冲区的数据，然后通过任务对数据进行处理，信号量释放。这样设计的优点是屏蔽了中断处理与顶层系统任务之间的联系，使得软件设计清晰、合理[5-6]。

4.2　PWM 信号捕获

微小型无人直升机执行任务主要通过操纵杆来控制，因此如何稳定实时地获取操纵杆的信号是核心问题。操纵杆的每个通道输出为 PWM 的脉冲信号，因此采用捕获 PWM 脉宽的方法来实现获取操纵杆信号。捕获的基本原理为在 PWM 信号的上升沿产生一次中断，定时器获得一个计数值 $T1$，接着在下降沿产生中断，获得一个计数值 $T2$，然后判断是否跨越定时器周期，最后根据跨越的定时器周期数和两次计数值 $T2$、$T1$ 之差计算得到 PWM 高电平时间[10]。PWM 捕获流程如图 3 所示[4]。

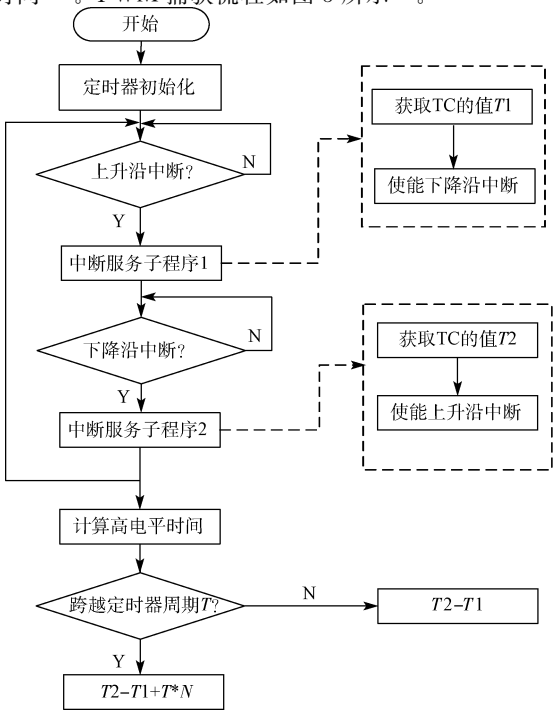

图 3　定时器捕获 PWM 的流程

4.3　串口通信

微小型无人直升机中需要通过串口通信获取 GPS、IMU 等传感器的数据，然后通过传感器数据融合、控制律解算等过程实现对无人直升机控制，数据传输的实时性决定了控制性能，因此采用中断的串口通信方式[7]。其基本原理是对串口初始化后，若是发送数据，则使能 THRE 中断，当 TXFIFO 为空时，THRE 中断被激活，然后发送数据，并复位。若是接收数据，则使能 RBR 中断，当 RXFIFO 到达所定义的触发点时，RDA 中断就会被激活，CPU 可以读出由触发点所定义的数据块，当 RXFIFO 的深度低于触发点时，RDA 中断复位[9]。串口的收发流程如图 4 所示。

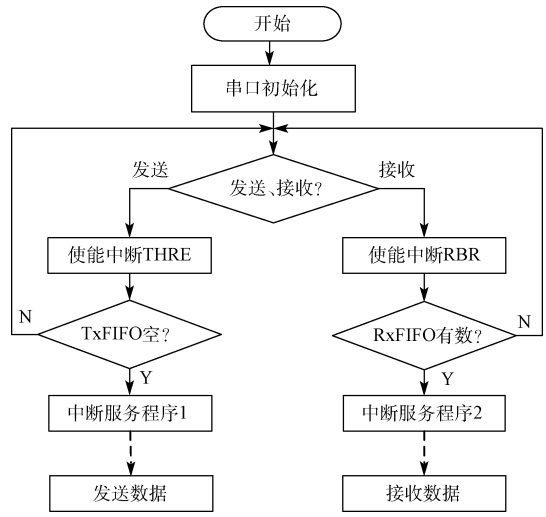

图 4　串口收发流程

5　系统移植和驱动的测试与验证

本文中采用类似于文献[2]中示例 2 的方式来进行系统的测试验证。使用 13 个 OSTaskCreateExt()函数创建的任务，以便能够进行堆栈检测，以及 μC/OS-Ⅱ的 2 个内部任务（空闲任务和 CPU 利用率的统计任务），同时创建的 13 个任务中涉及本文中的硬件中断的使用以及飞行控制软件中的任务调度。通过这样处理，该系统在涉及多硬件中断的使用的情况下，测试了 μC/OS-Ⅱ的堆栈检查功

能，每个任务已经使用了的堆栈空间、剩余的堆栈空间、每个任务的执行时间、任务数、任务切换次数、CPU 利用率，通过一个串口将 μC/OS-Ⅱ任务的测试信息通过上位机显示，如图 5 所示[8]。

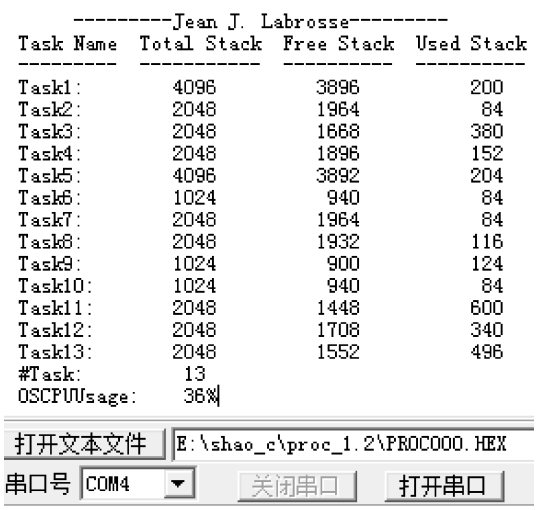

图 5　软件测试和验证结果

经过长时间系统测试，μC/OS-Ⅱ系统中的各个任务运行正常，每个任务的堆栈使用情况稳定，CPU 利用率稳定在 36%左右，实时时钟准确，各个驱动数据处理无误，整套系统运行正常，足够证明 μC/OS-Ⅱ系统移植成功，也能够反映在多硬件中断的影响 μC/OS-Ⅱ系统的稳定性基本不受干扰。

5 结　论

针对 μC/OS-Ⅱ系统在多硬件中断环境下移植过程中产生的一系列不稳定因素，本文基于 μC/OS-Ⅱ系统在 ARM 平台下移植的相应处理，提出合理设定中断优先级和优化中断服务程序的解决方法，能够有效保证系统的稳定运行，提高系统的实时性，减少对 CPU 的占用率。本文中所涉及的解决方法已在微小型无人直升机上进行了试飞验证，满足了飞行控制系统的工程应用需求。

参考文献

[1] 杨一栋. 直升机飞行控制(第 2 版)[M]. 北京：国防工业出版社，2011. 6：25-29.

[2] LABROSSE J J，邵贝贝. 嵌入式实时操作系统 μC/OS-Ⅱ[M]. 北京：北京航空航天大学出版社，2003.

[3] 任哲，嵌入式实时操作系统 μC/OS-Ⅱ原理及应用(第 3 版)[M]. 北京：北京航空航天大学出版社 2014(1)：235-253.

[4] 陈是知，μC/OS-II 内核分析、移植与驱动程序开发[M]. 人民邮电出版社，2007：30-45.

[5] 徐海龙，邱建，王晓娜，等. μC/OS-Ⅱ的优化移植和设备驱动框架设计[J]. 计算机测量与控制，2012，20(9)：2501-2503.

[6] 吴绍根. μC/OS-Ⅱ下通用驱动框架的设计与实现[J]. 微计算机信息，2006，22(12-2)：58-60.

[7] 王奕. 基于 μC/OS-Ⅱ的无人直升机飞行控制系统软件设计[D]. 南京：南京航空航天大学，2008.

[8] 刘培强. 小型无人直升机飞行控制系统软件及仿真系统的开发[D]. 南京：南京航空航天大学，2010.

[9] 张旭，亓学广，李世光，等. 基于 STM32 电力数据采集系统的设计[J]. 电子测量技术，2010，33(11)：90-92.

[10] 袁雪，张志文，司庆丹. 基于 ARM 的智能数据采集系统设计[J]. 国外电子测量技术，2014，33(11)：66-69.

作者简介

刘世隆，1991 年出生，硕士研究生，研究方向为无人直升机飞控技术。