

文章编号:1005 - 0523(2009)06 - 0040 - 04

基于 $\mu\text{C}/\text{OS- II}$ 的微机保护系统的设计

杨丰萍,柳 亮

(华东交通大学 电气与电子工程学院,江西 南昌 330013)

摘要:针对传统微机保护装置存在的不足,围绕一种以 ARM 和 $\mu\text{C}/\text{OS- II}$ 为核心的微机保护系统这一课题进行研究,介绍了该系统的总体设计、 $\mu\text{C}/\text{OS- II}$ 在 ARM 处理器上的移植,并对微机保护的数据采集程序设计作了详细阐述。实践证明,引入了 RTOS 的微机保护具有更好的实时性、可靠性和可扩展性。
关键词:微机保护; $\mu\text{C}/\text{OS- II}$;ARM;数据采集
中图分类号:TP274 **文献标识码:**A

自从 20 世纪 80 年代我国第一套微机线路保护投入运行到现在,微型机继电保护装置凭借其优越性能在国内得到了广泛的认可和使用。随着电力系统自动化技术的发展及微机保护装置的大量投入使用,电力系统对微机保护的要求不断提高,并且由于电力系统本身的特殊性,它需要微机保护装置具有很强的实时性、高可靠性、高扩展性、更强的网络通信能力及更强大的人机交互界面。它们不仅对硬件要求高,而且对嵌入式软件的要求也不断增加,从而引进了嵌入式实时操作系统(Real Time Operating System, RTOS),这些操作系统具有可裁减的内核结构、高效的多任务优先级管理、强大的可移植性和扩展性以及微秒级的中断管理等都更加有利于控制效率的提高,并且更容易满足这些方面的需要^[1]。

1 微机保护系统的总体设计

整个系统主要由模拟信号采集系统,ARM 主控板,显示设备,外部开入设备和上位机组成。如图 1 所示。

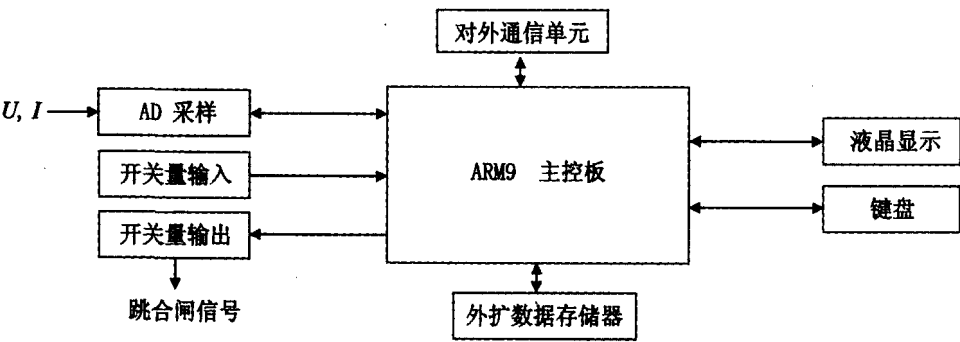


图 1 微机保护系统结构图

本文选用的 ARM 处理器是 Samsung 公司生产的 S3C2410X,这款芯片集成了以下部件:16KB 指令 Cache、16KB 数据 Cache、MMU、外部存储器控制器、LCD 控制器、NAND FLASH 控制器、4 个 DMA 通道、3 个 UART 通道、1 个 IIC 总线控制器、1 个 IIS 总线控制器、4 个 PWM 定时器、1 个内部定时器、通用 IO 口、实时时钟、8 通道 10 位 ADC 和触摸屏接口等。

对于传统微机保护的设计,它采用的是主循环加中断的线性结构,也就是前后台系统。随着电力系统自动化的发展,对微机保护功能要求也越来越多。这种设计方法,会导致软件复杂度上升,开发及维护时

收稿日期:2009 - 06 - 29
作者简介:杨丰萍(1969 -),女,江西萍乡人,教授,研究方向为交通信息工程及控制等。

间增加,成本提高,并可能造成中断响应的不及时或任务的阻塞或死锁。本系统将引入实时操作系统 $\mu\text{C}/\text{OS- II}$,它是一个开放式的实时内核,它没有软件支持包、文件系统、I/O 系统及网络系统等。图 2 所示为 $\mu\text{C}/\text{OS- II}$ 硬件/软件体系结构^[2]。

2 $\mu\text{C}/\text{OS- II}$ 的移植^[3,4]

2.1 OS-CPU.H 代码的移植

文件 OS - CPU.H 中包括了用 # define 语句定义的与处理器相关的常数、宏以及类型。移植时主要修改的内容有:

- (1) 设置与编译器相关的数据类型;
- (2) 用 # define 语句定义 2 个宏开关中断;

```
# define OS_ENTER_CRITICAL() ARMDisableInt() /
* 关中断 * /
# define OS_EXIT_CRITICAL() ARMEEnableInt() / *
开中断 * /
```

(3) 设置 OS_STK_GROWTH 为 1,表示堆栈从上(高地址)往下(低地址)递减;

(4) OS_TASK_SW()是一个宏,用以实现任务从低优先级向高优先级切换,它用于任务级代码中。其代码如下: # define OS_TASK_SW OSCtxSw。

2.2 OS-CPU-C.C 代码的移植

OS-CPU-C.C 代码需要移植 10 个简单的相关函数,但在一般情况下,仅仅需要移植任务栈初始化函数 OSTaskStkInit(),其它 9 个函数可以用简单的空函数来实现。根据 ARM 处理器的结构和特点确定任务栈的结构,也就是在堆栈增长方向上定义每个需要保存的寄存器位置。其任务堆栈空间由高到低依次为 PC, LR, R12, R11, R10, ..., R1, R0, CPSR, SPSR。在进行堆栈初始化以后,OSTaskStkInit() 返回新的堆栈栈顶指针。

2.3 OS-CPU-A.S 代码的移植

- (1) OSStartHighRdy()用来使就绪态任务中优先级最高的任务开始运行。
- (2) OSCtxsw()是任务级的任务切换函数,通过执行软中断指令,或者依据处理器的不同,执行 TRAP 指令来实现的。中断服务子程序、陷阱或异常处理的向量地址必须指向 OSCtxSw()。
- (3) OSIntCtxSw()是中断级的任务切换函数,OSIntExit()通过调用 OSIntCtxSw()在 ISR 中执行任务切换功能。
- (4) OSTickISR()是时钟节拍中断服务函数,实现时间的延迟和超时功能。

3 基于 $\mu\text{C}/\text{OS- II}$ 的微机保护系统的软件设计

3.1 软件设计的总体思想

对于基于 $\mu\text{C}/\text{OS- II}$ 的微机保护系统的软件设计,本文将从两部分予以分析。

(1) 多任务模块运行前的程序流程

在多任务环境运行前,系统上电后也会首先执行一个入口函数程序,即平常所说的主函数。该主函数主要是完成一些应用系统和操作系统的初始化工作,并让系统进入实时操作系统管理下的多任务运行环境。其中应用系统的初始化主要包括 ARM 本身初始化、与 ARM 相连的一些外围设备的初始化和应用系

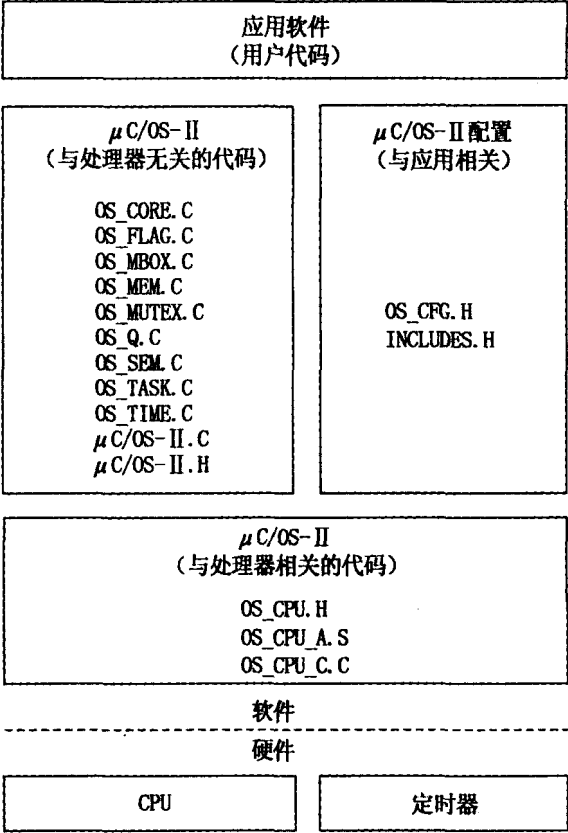


图 2 $\mu\text{C}/\text{OS- II}$ 硬件/软件体系结构

统所用的一些全局变量、参数等的初始化。其程序流程图如图3所示。

(2) 多任务模块进入运行环境的程序流程

当多任务模块进入运行环境,各个任务的调度全部交由嵌入式实时操作系统去完成,其程序结构与传统的线形程序结构也发生了很大的变化,并没有一个具体的程序流程。但这并不等于在该环境下各任务的调度是无序的,任务根据运行前设置好的任务优先级和触发事件,以任务链的形式去完成系统的各个功能。

当没有中断事件发生并激活其他任务时,该实时嵌入式操作系统不断的循环调用开关量输入输出任务、键盘任务、显示任务、通信数据接收任务、测量任务以及系统的空闲任务。

当有采样定时器发生中断时,以该事件为原发事件,在中断服务程序中激活定时采样任务,由于此时定时采样任务优先级为最高优先级,当前任务立即切换为定时采样任务;当数据采样完成后,首先运行启动元件,判断故障判别与处理任务启动条件是否满足,当满足时,立即激活故障判别与处理任务,由于此时故障判别与处理任务的优先级为最高,任务立即从定时采样任务切换到故障判别与处理任务,当完成故障判别与处理任务后重新返回;如果启动条件不满足,当采样值数据窗体满足采样算法所需窗体大小时,则激活数据处理任务,计算相关数据。以采样定时器中断作为原发事件的任务链程序流程图如图4所示。

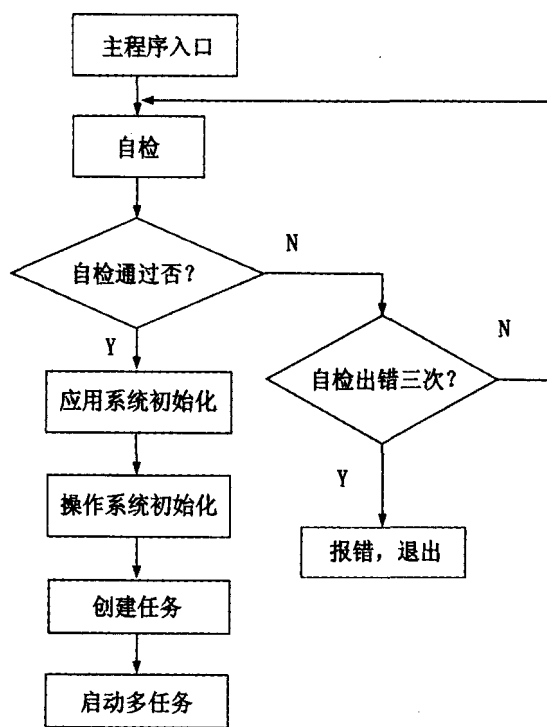


图3 多任务模块运行前的程序流程图

3.2 数据采集程序模块

本系统的软件设计所采取的方法如图5所示的正弦电流采样,每工频周期采12点,在正弦函数过零点前后取采样点 i_1, i_2 和下一个周波过零点前后的 i_1', i_2' 。在过零点前后,正弦函数很接近直线,作线性处理时,有如下关系

$$\Delta t_1 = \frac{i_2}{i_2 - i_1} T_{S1} \quad \Delta t_2 = \frac{i_2'}{i_2' - i_1'} T_{S2}$$

在 i_1 之前,采样周期为 T_{S1} ,在 i_2 之后一个工频周

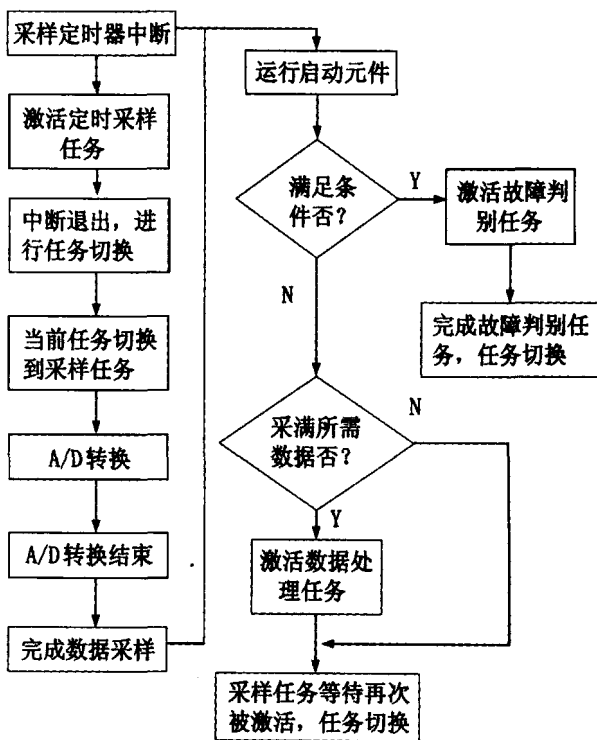


图4 任务链程序流程图

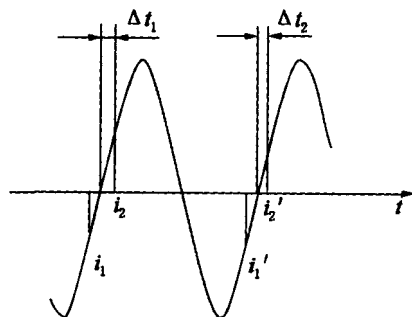


图5 采样频率跟踪法

期,采样周期为 T_{S2} 。当一个工频周期恰好采样12次即 $T = 12 T_{S2}$ 时,应有 $t_1 = t_2$ 。若 $t_1 > t_2$ 意味着采样周期过长,应该减小,其减小量为

$$\Delta t = \frac{\Delta t_1 - \Delta t_2}{12}$$

即 $T_{S3} = T_{S2} - t$ 。这种方法每一工频周期调节一次,便能够满足微机保护的要求。

采样中断是系统中的最高中断,不能够被其余中断打断。根据每周采样12点,采样频率为1200 Hz,则ARM的定时器T1的采样中断时间为0.833 33 ms,相继对保护电流、测量电流、电压和零序电流等进行采样。每个采样点结束后就要启动A/D转换,等将所有数据转换完成后,读取采样数据到内存缓冲区,进行差分保存,等采满一个周期,然后给数据处理任务发信号,进入数据处理任务。其流程图如图6所示。

4 结论

基于 $\mu C/OS-II$ 的微机保护系统,充分发挥了 $\mu C/OS-II$ 的作用,把各功能模块根据其功能特点和运行特点划分不同的任务,任务的功能划分清晰,易于实现。在数据采集方面,大大提高了其实时性和高速性。在硬件模块化的基础上,实现了软件的模块化,从而缩短了开发和维护时间。

参考文献:

- [1] 陈杰,刘亚平,张旭光等.多任务机制在继电保护装置中的应用[J].继电器,2001(8):49-51.
- [2] 刘玉娟.基于RTOS的微机保护平台的研究[D].保定:华北电力大学,2004.
- [3] Jean J Labrosse.嵌入式实时操作系统 $\mu C/OS-II$ (第二版).北京:北京航空航天大学出版社,2003.
- [4] 吴永忠,程文娟,郑淑丽.嵌入式实时操作系统 $\mu C/OS-II$ 教程[M].西安:西安电子科技大学出版社,2007.

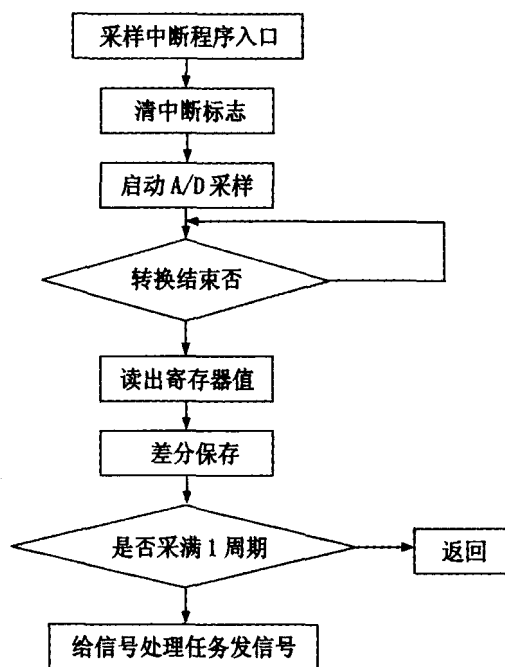


图6 数据采集程序流程图

A Design of Protective Relay System Based on $\mu C/OS-II$

YANG Feng-ping, LIU Liang

(School of Electrical and Electronic Engineering, East China Jiaotong University, Nanchang 330013, China)

Abstract: In view of the insufficiency of traditional Protective Relay devices, this paper studies Protective Relay System by using ARM and $\mu C/OS-II$. Firstly, it introduces overall design of the system and the transplant of $\mu C/OS-II$ fsystem to ARM processor. Then the data acquisition programming of the Protective Relay is elaborated in detail. The test proves that Protective Relay with RTOS has a better timeliness, reliability and extendibility.

Key words: protective relay; $\mu C/OS-II$; ARM; data acquisition

(责任编辑:王建华)