基于 ARM 与 WINCE 的 RS485 驱动程序设计

范忠强[1],徐 平[2*],姜周曙[1]
(1. 杭州电子科技大学 自动化学院,浙江 杭州 310018;
2. 杭州电子科技大学 生命信息与仪器工程学院,浙江 杭州 310018)

摘要:随着工业设备控制要求的日益复杂,嵌入式的工业控制系统成为必然趋势。由于 RS485 具有良好的抗干扰性、传输距离长和多站能力等优点,在工业控制领域中应用最广泛。为了开发基于 ARM9 内核和 Windows CE 5.0 系统的 RS485 驱动程序,首先给出了硬件电路设计,接着阐述了 Windows CE 系统的驱动模型以及中断处理方法;然后详细描述了 RS485 驱动程序的具体设计过程,包括驱动程序的结构设计、关键函数设计、驱动程序的注册和加载等。实验结果表明,RS485 驱动程序运行稳定可靠。

关键词:Windows CE 5.0;RS485;流驱动;中断
中图分类号:TP23;TH39 **文献标志码**:A **文章编号**:1001-4551(2011)11-1327-05

Design of RS485 driver based on ARM and WINCE

FAN Zhong-qiang[1], XV Ping[2], JIANG Zhou-shu[1]
(1. College of Automation, Hangzhou Dianzi University, Hangzhou 310018, China; 2. College of Life Information Science & Instrument Engineering, Hangzhou Dianzi University, Hangzhou 310018, China)

Abstract: As the industrial equipment control requirements become increasingly complex, embedded industrial control systems become an inevitable trend. Due to its advantage such as excellent anti-interference ability, long transmission distance and multimode transmission ability, RS485 is applied most widely in the industrial control field. In order to develop the RS485 driver based on the ARM9 core and Windows CE system, firstly, the hardware design for RS485 was presented; secondly, the driver model and interrupt handling methods in Windows CE system was introduced; thirdly, the RS485 driver's design process was given in detail, including the driver's structure design, key functions design, the registration and load of driver. The experimental results indicate that RS485 driver runs stably and reliably.

Key words: Windows CE 5.0; RS485; stream interface driver; interrupt

0 引 言

Windows CE 系统是微软为开发嵌入式智能设备精心打造的 32 位、嵌入式、实时、多任务和模块化的操作系统,具有可靠性好、实时性强、内核体积小以及开放源代码等特点, 被广泛地应用于工业控制、汽车电子、个人电子消费品等各个领域,其开发流程快,软、硬件选择灵活,成为嵌入式开发的首选平台之一[1-2]。驱动是联系操作系统和硬件平台的枢纽,在嵌入式开发中具有重要地位。在硬件上选择抗噪声干扰能力强、具有多机通讯能力的 RS485 接口, 具有价格低廉、操作方便等优势,在过程控制、工业自动化、远程控制、建筑自动化以及安全系统 (尤其是电机控制动作控制)等场合应用广泛[3]。

由于在 Windows CE 5.0 系统中没有现成的 RS485 驱动可用,本研究探讨在 Windows CE 5.0 系统中如何进行 RS485 驱动开发。该驱动使用 Platform Builder 5.0 集成开发工具进行开发。

1 RS485 的硬件电路设计

目前,采用 32 位 ARM 架构 CPU 的嵌入式产品代表嵌入式市场的主流。该系统采用 32 位的基于 ARM920T

收稿日期:2011-04-20
作者简介:范忠强(1985-),男,山东滕州人,主要从事嵌入式软件系统方面的研究. E-mail:fanzq1314521@126.com
通信联系人:徐 平,男,副教授,硕士生导师. E-mail:xuping@hdu.edu.cn

的 Samsung S3C2410A 处理器，其内部集成了 3 个独立的 UART 控制器，每个 UART 均有 2 个 16 Bytes 的 FIFO 用于数据发送、接收，都可以工作在中断模式或者 DMA 模式[4]。系统设置成当接收 FIFO(发送 FIFO)的内容不为空时就会触发接收中断(发送中断)通知 CPU 接收(发送)数据。RS485 的硬件电路设计如图 1 所示，选用 S3C2410A 的 UART1 作为串口控制器，选用广泛应用于工业控制局域网、集成服务数字网络、分组交换技术等场合的 MAX3485 作为电平转换芯片，其传输速率最高可达 10 Mbps，满足串口数据传输的要求[5]。由于 MAX3485 是半双工工作方式，要选取一个 GPIO 端口作为发送控制引脚控制数据的发送，本研究选用的发送控制引脚是 GPC8。

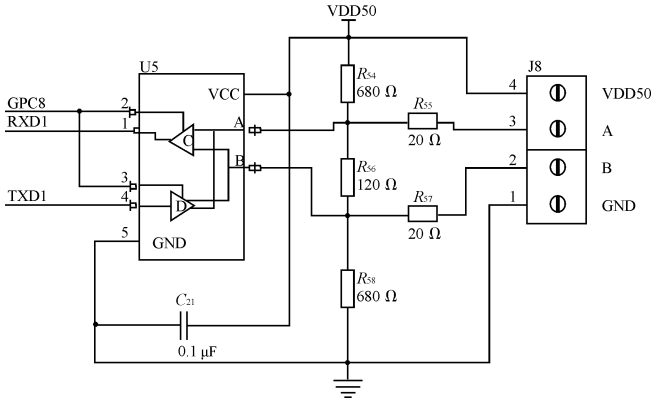

图 1 RS485 模块设计图

2 Windows CE 驱动设计基础

在进行 RS485 驱动的具体开发之前，需要先掌握 Windows CE 下的驱动模型和中断机制。

2.1 Windows CE 下的驱动模型

Windows CE 系统下的驱动模型主要有本地设备驱动和流设备驱动，分别导出定制的用户接口和流接口。按驱动是否分层，Windows CE 下的驱动又可以分为分层驱动和单体驱动，驱动模型如图 2 所示，与单层驱动相比，分层驱动包含两层：模型设备接口 MDD 层和平台设备接口 PDD 层，PDD 层直接操作具体硬件设备，并向 MDD 层提供 DDSI 接口；MDD 层针对某一类型的驱动设计通用的操作，调用 DDSI 函数，并向操作系统提供接口[6]。

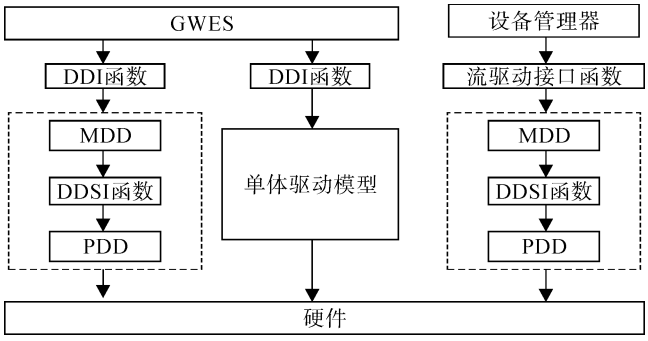

图 2 Windows CE 驱动模型

综上所述，一方面，RS485 驱动属于典型的流设备驱动，需要导出流接口以便和系统或应用程序交互；另一方面，RS485 驱动也属于分层驱动，中间使用 DDSI 函数通信，由于 MDD 层由微软提供，这样就加快了 RS485 驱动的开发进度，降低了开发难度。

2.2 Windows CE 下的中断机制

中断设计是 Windows CE 驱动程序中的一个关键，在驱动设计中经常使用事件关联某一个特定中断。Windows CE 系统将中断处理过程分为 ISR 和 IST 两个过程，ISR 负责把物理中断请求 IRQ 转化成逻辑中断并返回给内核，IST 则负责中断的逻辑处理。具体过程如图 3 所示：某一个硬件中断发生后，它被发送至内核的异常中断处理器(如图 3 中①所示)，由内核处理中断异常。内核的中断支持处理器调用 OAL 函数 OEMInterruptDisable 屏蔽这个硬件中断（如图 3 中②所示)，然后内核调用中断服务例程 ISR(如图 3 中③所示)并根据返回的逻辑中断标识产生一个已注册事件(如图 3 中④所示)，该事件激活一个处于阻塞的中断服务线程 IST(如图 3 中⑥所示)，进入具体中断处理过程。必要时，IST 调用各种 I/O 函数操作硬件(如图 3 中⑦所示)。当中断处理完成后，IST 调用函数 InterruptDone 通知内核（如图 3 中⑧所示)，内核调用 OEMInterruptDone 完成此中断的所有处理（如图 3 中⑨所示)，并通知硬件重新开启这个中断[7]。

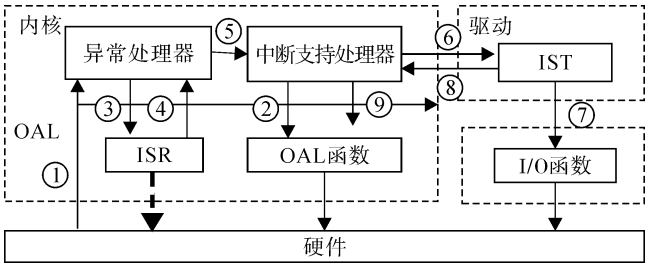

图 3 中断处理流程

下面介绍 RS485 的 IST 的具体处理流程：① 在 MDD 层初始化里使用 CreateEvent 创建了 4 个事件对象：串口事件对象 hSerialEvent、串口接收数据事件对象 hReadEvent、串口发送数据事件对象 hTransmitEvent 及线程退出事件对象 hKillDispatchThread；② 通过 GetSerialObject 获取关联硬件寄存器参数的 HWOBJ 结构体；③ 通过调用系统函数 InterruptInitialize 关联事件和中断，并调用 InterruptDone 函数确保串口中断能正常产生；④ 创建串口处理线程 SerialDispatchThread，调用 WaitForSingleObject( ) 函数等待事

件被触发。⑤ 一旦事件被捕获，调用 HWGetIntrType 函数判断中断的类型，从而调用不同的中断处理方法。⑥ 中断处理完毕后调用 InterruptDone 重新打开被屏蔽的中断,以便能处理重复的中断。

3　RS485 驱动程序设计

在 Windows CE 5.0 下驱动程序实际上是一个 DLL 形式的动态链接库,它导出了一系列操作硬件设备的接口函数,以供其他程序调用。

3.1　RS485 驱动程序分层驱动设计

RS485 驱动整体结构如图 4 所示,RS485 驱动程序实际上由 MDD 层和 PDD 层组成,MDD 层由微软已经开发好了,可直接使用,PDD 需要具体开发:

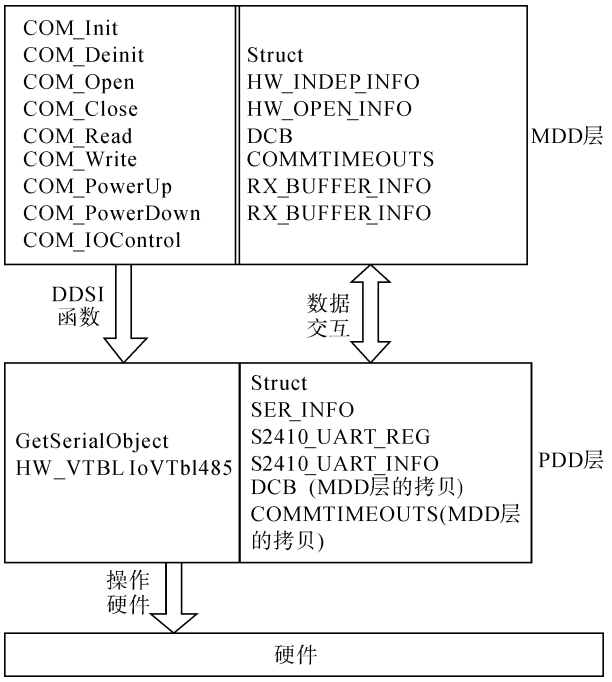

图 4　RS485 驱动整体结构

(1) GetSerialObject 是在 MDD 层 COM_Init 中被调用,其函数实现是在 PDD 层。由于 S3C2410A 处理器有 3 个 UART 控制器，需要根据此驱动注册表中 DeviceArrayIndex 的值来判断加载的是 COM1、COM2 还是 COM3，并返回相应 PHWOBJ 结构体指针，HWOBJ 结构体包含中断号 dwIntID、中断标志位 BindFlags 和 PHW_VTBL 结构体指针。

(2) 指针 PHW_VTBL 指向 HW_VTBL 结构体，HW_VTBL 结构体由一系列函数指针组成,MDD 层通过调用该结构体实现对 PDD 层的函数调用,实现了具体的硬件操作,其实现的接口如下所示的结构体成员:

```
const HW_VTBL IoVTbl485={
SerInitSerial485,
SL_PostInit,
SerDeinit,
SerOpen,
SerClose,
SL_GetInterruptType,
SL_RxIntr,
SL_TxIntrEx,
SL_ModemIntr,
SL_LineIntr,
SL_GetRxBufferSize,
SerPowerOff,
SerPowerOn,
SL_ClearDTR,
SL_SetDTR,
SL_ClearRTS,
SL_SetRTS,
SerEnableSerial,
SerDisableSerial,
SL_ClearBreak,
SL_SetBreak,
SL_XmitComChar,
SL_GetStatus,
SL_Reset,
SL_GetModemStatus,
SerGetCommProperties,
SL_PurgeComm,
SL_SetDCB,
SL_SetCommTimeouts,
};
```

(3) 结构体作为数据处理和数据存储的基本单元,是实现数据同步和数据共享的基本手段。MDD 层和 PDD 层的交互数据主要是 EventCallback 事件回传机制和上述的 HWOBJ 结构体。串口事件发生时，EventCallback 将发生的事件号由 PDD 层传递到 MDD 层的串口等待事件 WaitCommEvent 函数和 EvaluateEventFlag 函数,再由 MDD 层处理。PDD 层也会复制一份 MDD 层的 COMMTIMEOUTS 和 DCB 结构体,从硬件角度对串口超时参数和串口数据流传输的控制参数进行配置。

3.2　相关函数设计

本研究在 PDD 层要实现 GetSerialObject 和 HW_VTBL IoVTbl485 所包含的成员函数,由于接口众多,下面介绍几个关键函数的设计思路:

(1) SerInitSerial485:初始化函数,主要过程有读串口注册表得到串口中断号、I/O 口物理地址、I/O 口地址长度等参数，为 UART 寄存器、中断寄存器、I/O 寄存器、时钟寄存器分配虚拟内存空间并映射相应的物理地址,配置串口属性结构体,映射发送数据寄存

器和接收数据寄存器到 S2410_UART_INFO 结构体，配置 485 串口工作在有 16 Bytes 缓冲区的使能 FIFO 模式，关闭所有中断，清除所有中断标志，默认设置 485 串口使能数据接收标志位、使能数据发送中断，直到 SL_PostInit 才重新打开中断。

（2）SerOpen：打开串口函数。主要进行串口寄存器的配置，获取应用程序发送给串口驱动的 DCB 结构体的波特率、数据位、停止位、有无校验等参数并进行配置，配置串口所用时钟与中断触发电平，使能串口接收、发送、错误与超时中断，清空并使能串口 FIFO，配置串口 FIFO 模式下的中断触发条件。

（3）SerClose 与 SerDeinit：SerClose 是关闭串口函数，主要负责清除并禁止串口所有中断。SerDeinit 是卸载串口函数，主要负责释放串口所占用的内存，释放 HWOBJ 和 SER_INFO 结构体。

（4）SL_GetInterruptType：获得中断类型，其返回值为以下其中之一：INTR_NONE、INTR_LINE、INTR_RX、INTR_TX，分别代表没有中断、线路中断、接收数据中断、发送数据中断。

（5）SL_RxIntr：进行接收数据，基本流程图如图 5 所示。在接收到 INTR_RX 中断后该函数被调用，它把进入 UART 的接收 FIFO 中的数据逐个字节读取出来放到系统接收 buffer，然后再由 MDD 层传递给应用程序的接收缓冲区。这个函数的关键点是在每接收一个字节时都需要判断接收 FIFO 中是否有数据，没有数据则结束接收。而在 MDD 层接收数据的关键是用头、尾双指针标识数据在 buffer 中的位置并完成同步，每次都需要判断是否越界读取数据。不过值得注意的是对于半双工的 RS485，要将发送控制引脚配置为接收模式才能接收到数据，否则会导致不能接收数据或接收到的都是乱码。

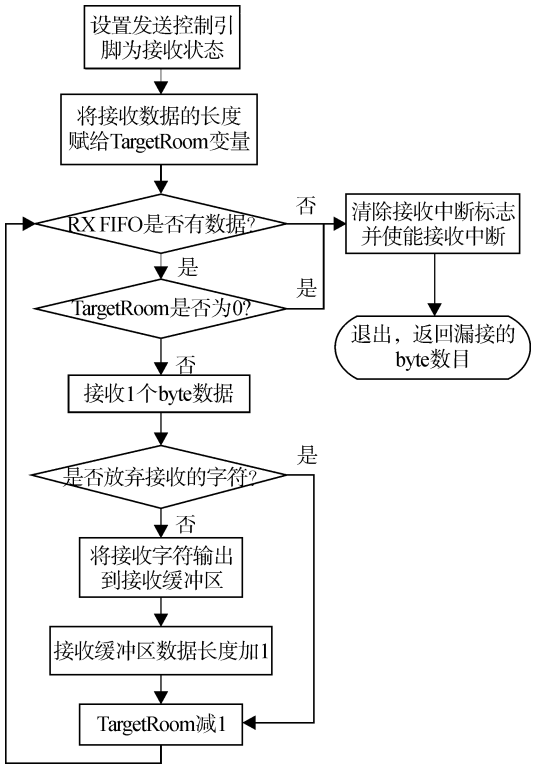

图 5 PDD 层接收数据流程图

（6）SL_TxIntrEx：进行发送数据，在接收到一次 INTR_TX 中断后被调用，负责将应用程序发送 buffer 中的数据发送到 UART 的发送 FIFO，流程图如图 6 所示。这个函数的关键是在发送数据时调用 EnterCriticalSection 函数对进入发送数据临界区的资源进行保护，防止多线程对数据的并发操作。由于发送 FIFO 最多可容纳 16 Bytes，MDD 层控制此函数最多发送 16 Bytes，大于 16 Bytes 则要连续调用该函数直至发送完毕。该函数使用的策略是每次发送数据都要检查发送 FIFO 是否已经满，发送后还要等待数据被发送出去，最后返回发送数据的数目。对于半双工的 RS485 来说，还要注意在发送开始时要关闭接收中断，发送控制引脚设置为发送模式；结束发送时开启接收中断，发送控制引脚设置为接收模式。

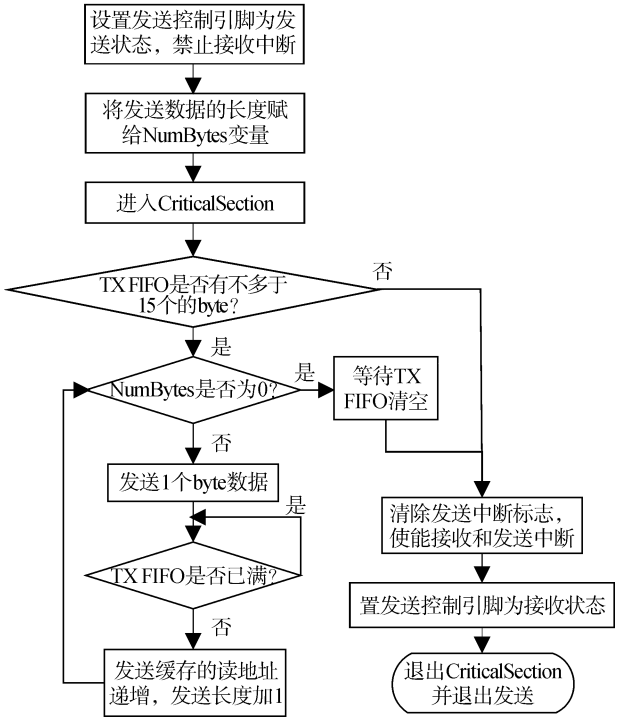

图 6 PDD 层发送数据流程图

3.3 RS485 驱动程序的注册和加载

RS485 驱动程序的注册和加载：

（1）Windows CE 系统冷启动以后，操作系统陆续启动 NK.EXE、DEVICE.EXE、FILESYS.EXE、GWES.EXE 等进程，其中 DEVICE.EXE 进程负责加载流驱动。DEVICE.EXE 根据注册表的设置，在加载总线枚举

器 BusEnum.dll 后，总线枚举器会枚举［HKEY_LOCAL_MACHINE\Drivers\BuiltIn］下的所有子键,并加载相应的驱动程序[8-9]。因此在开发流驱动时要在［HKEY_LOCAL_MACHINE \Drivers\BuiltIn］注册表项下加入设备注册信息。对于本研究的 RS485 驱动应在注册表中添加以下的注册信息[10-12]：

```
［HKEY_LOCAL_MACHINE\Drivers\BuiltIn\SER 2410_485］
"DeviceArrayIndex"=dword:1 //串口编号
"Irq"=dword:13 //系统使用的逻辑中断号
"IoBase"=dword:50004000 //串口寄存器的物理地址
"IoLen"=dword:2C
"Prefix"="COM"//RS485 驱动前缀
"Dll"="SER2410_485.dll"//驱动以 DLL 的形式存在
"Order"=dword:0 //RS485 驱动启动顺序
"Priority"=dword:0
"Port"="COM2:"
"DeviceType"=dword:0//设备类型
"FriendlyName"="Serial Cable on COM2:"
```

（2）在注册完驱动后将 RS485 驱动加入系统是通过修改 Plateform.bib 文件完成的：

SER2410_485.dll $（_FLATRELEASEDIR）\ SER 2410_485.dll　NK　SH

这表示 RS485 驱动对应的 DLL 文件存放在映像的 NK 区域，以系统隐藏文件的形式加载到系统内存中。

（3）在 Platform Builder 中选择“Build OS->build and sysgen”编译 Windows CE 系统映像,编译好后将映像文件下载到硬件开发板,就可以使用 RS485 接口进行通信了。

4　测　试

本研究使用 EVC 集成开发工具开发一个适合在 Windows CE 系统运行的串口调试助手程序，其在 ARM 上的运行界面如图 7 所示，主要借助 MFC 的消息映射机制和多线程技术实现。

图 7　运行在 WINCE 上的 RS485 测试工具界面

实验过程如下：采用 ARM 与 PC 机互连方式对 RS485 采用多种波特率方式进行收发测试，选取 115 200 bps、19 200 bps、9 600 bps 等典型常用波特率点作为测试点,对其丢包率进行测试,测试数据如表 1 所示。其他调试参数为:8 个数据位,1 个停止位,无校验。

经多次重复测试和系统联机调试表明:RS485 在多个波特率下工作稳定可靠,能够正常接收、发送数据。

表 1　RS485 测试实验数据

波特率 /bps	发送或接收的数据量 /Bytes	发送数据丢包率 /(%)	接收数据丢包率 /(%)
115 200	300 000	0	0
19 200	300 000	0	0
9 600	300 000	0	0

5　结束语

对于 Windows CE5.0 的驱动，其 ISR 是运行在内核级别的,而 IST 是运行在用户级别的,这样既保证了中断具有较高的优先级,也保证了系统的稳定正常运行,不会因驱动的崩溃而崩溃。RS485 能够正常、稳定地接收发送数据的关键是串口能够快速监测到中断并迅速传至 IST 处理。本研究采用多种措施加强 RS485 数据收发的稳定性,例如:提高串口线程的优先级;提高串口中断优先级;精简 ISR,减少 ISR 的延迟;UART FIFO 中的触发接收和发送中断的字节数要设置的尽量小,如可设置为 4 Bytes 或 8 Bytes,本研究设置 TX FIFO 和 RX FIFO 中多于 4 Bytes 就触发中断,这样可尽量避免数据溢出;RS485 属于低速设备,而工业级通信常用 9 600 bps 作为其波特率,故其波特率范围可设置在 0 ~ 115 200 bps 内,不需要设置过大。通过多次重复测试并实际联机调试证明,RS485 驱动能够稳定工作,实现了预期目标。

参考文献(References):

[1] 张冬泉,谭南林. Windows CE 开发实例精粹[M]. 北京:电子工业出版社,2008.

[2] WILLIAMS R. Computer Systems Architecture:a Networking Approach[M]. 2nd ed. Beijing:China Machine Press,2007.

[3] 平毅斌. 基于 RS-485 网络的控制系统研究[D]. 成都:西南交通大学材料科学与工程学院，2003.

[4] Samsung Corporation.S3C2410X 32-Bit RISC Microprocessor User's Manual［M］. Samsung Electronics,2003.

[5] MAXIM. 3.3 V-Powered, 10 Mbps and Slew-Rate-Limited True RS-485/RS-422 Transceivers[EB/OL]. [2011-04-02]. http://www.waveshare.net/shop/MAX3485-price.htm.

（下转第 1357 页）

PID 参数初始值为:K_{p0}=12.0, K_{i0}=0.15, K_{d0}=1.0,采样时间为 10 ms。

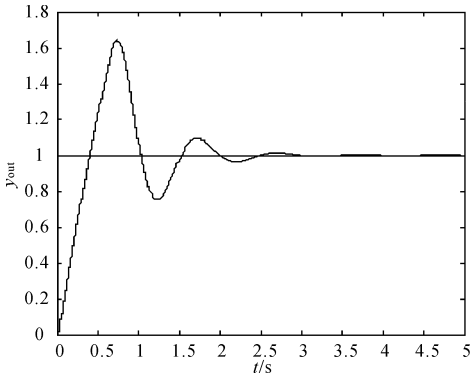

图 4　定尺飞锯常规 PID 控制仿真图

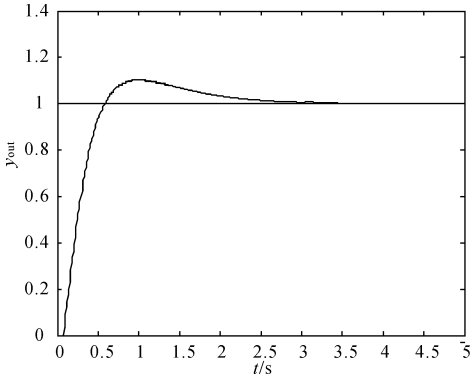

图 5　定尺飞锯模糊 PID 控制仿真图

对比图 4 和图 5 可知,模糊 PID 控制在不牺牲上升时间的前提下, 调整时间比常规 PID 控制要好,而且超调量小,所以模糊 PID 控制比常规 PID 控制拥有更好的动态响应性能。

5　结束语

本研究通过将模糊控制和常规 PID 控制相结合,设计了一个基于模糊自适应 PID 控制的定尺飞锯控制系统。仿真实验表明,该技术可大大提高系统的鲁棒性,改善动态特性,而且具有更大的灵活性、适应性,以及更强的实时性与控制精度,可在定尺飞锯控制系统等领域推广应用。

参考文献(References):

[1] WOO Z W,CHUNG H Y,LIN J J. A PID type fuzzy controller with self-tuning scaling factors [J]. **Fuzzy Sets and System**,2000(115):321-326.

[2] 童自惠,甘永兴,沈启飙. 半闭环、全闭环兼容的位置控制数控飞锯系统[J]. 电气传动,2003,33(5):34-37.

[3] 宁辰校,李　兰,张戌社. 基于 PLC 的定长切割飞锯智能控制系统研究[J]. 河北科技大学学报,2009,30(3):219-222.

[4] 李士勇. 模糊控制. 神经控制和智能控制论[M]. 哈尔滨:哈尔滨工业大学出版社,1996.

[5] KUKOLJ D D,KUZMANOVIC S B,LEVI E. Design of a PID-like compound fuzzy logic controller[J]. **Engineering Application of Artficial Intelligence**,2001,14(6):785-803.

[6] REZNIK L,GHANAYEM O,BOURMISTROV A. PID plus fuzzy controller structures as a design base for industrial application[J]. **Engineering Application of Artificial Intelligence**,2000(13):419-430.

[7] 李先银,胡乾斌,李光斌. 最优 PID 控制算法在飞锯位置伺服系统中的应用[J]. 电气传动,2001,31(1):41-43.

[8] 赵一鹏,姜　伟. 基于模糊 PID 电液伺服控制系统的设计和仿真[J]. 轻工机械,2010,28(3):69-72.

[9] 杨红波,徐振越,高德山. 全闭环飞锯运动控制的实现[J]. 电气自动化,2009,31(6):19-20.

[10] 李雪莲. 基于 Matlab 的 PID 参数调整方法的仿真研究[J]. 机电技术,2011(1):4-6.

[**编辑**:张　翔]

(上接第 1331 页)

[6] 何宗键. Windows CE 嵌入式系统[M]. 北京:北京航空航天大学出版社,2006.

[7] 张冬泉,谭南林,苏树强. Windows CE 实用开发技术[M]. 北京:电子工业出版社. 2009.

[8] 周建设. Windows CE 设备驱动及 BSP 开发指南 [M]. 北京:中国电力出版社,2009.

[9] Microsoft. Platform builder for Microsoft Windows CE 5.0 help-device Manager[EB/OL]. [2006-12-01]. ms-help://MS.WindowsCE.500/wceddk5/html/wce50conDeviceManager.htm.

[10] 戴钦来, 马均华. 基于 ARM 和 DSP 的多轴伺服系统以太网通信[J]. 轻工机械,2011,29(1):62-66.

[11] [作者不详]. ARM-WINCE.WinCE BSP 的 BIB 文件介绍[EB/OL].[2008-09-16]http://blog.csdn.net/nanjianhui/archive/2008/09/16/2931991.aspx.

[12] [作者不详]. RS485 总线[EB/OL]. [2010-05-24]. http://www.bbfar.com.cn/article/2c/1053.html.

[**编辑**:李　辉]