文章编号:2095-476X(2015)01-0095-05

基于 NicheStack TCP/IP 协议栈的嵌入式以太网控制器的设计与实现

刘黎明[1], 王用玺[2]

(1. 商丘师范学院 物理与电气信息学院, 河南 商丘 476000;
2. 河南省城乡规划设计研究总院有限公司智能化研究所, 河南 郑州 450000)

摘要:针对传统嵌入式以太网系统存在的数据传输速率低、硬件不能升级、实时性和通用性不足等问题,提出了基于内嵌有 Nios II CPU 的 Altera Cyclone 系列 FPGA 的以太网控制器设计方案. 该方案针对以太网协议利用 Quartus II 和 Nios II IDE 为开发工具,对硬件进行重新配置,以提高系统集成度;采用 SOPC 技术构建了嵌入式网络硬件平台;基于 μC/OS-II 实现了 NicheStack TCP/IP 协议栈的移植及顶层应用程序的编写. 系统测试结果表明,数据能够以 400 Mb/s 的速率正确收发,满足了以太网通信速率的要求,并可根据实际情况灵活配置.

关键词:NicheStack TCP/IP 协议栈;以太网控制器;FPGA;SOPC;μC/OS-II
中图分类号:TP393 **文献标志码**:A **DOI**:10.3969/j.issn.2095-476X.2015.01.020

Design and realization of the embedded ethernet controller based on NicheStack TCP/IP protocol stack

LIU Li-ming[1], WANG Yong-xi[2]

(1. *School of Physics and Electrical Information, Shangqiu Normal University, Shangqiu* 476000, *China*;
2. *Intelligent Research Institute, He'nan Urban and Rural Planning and Design Research Institute Co., Ltd. Zhengzhou* 450000, *China*)

Abstract: In order to solve the problems of the traditional embedded Ethernet system, such as low data transmission rate, hardware cannot be upgraded, poor real-time and universality etc, the Ethernet controller design scheme based on Altera Cyclone series FPGA with Nios II CPU was proposed. The design aimed at Ethernet protocal using the Quartus II and Nios II IDE as the development tools for reconfiguration of the hardware in order to reduce the cost and improve the system integration. The SOPC technology was used to construct the embedded network hardware platform and based on μC/OS-II to achieve the migration of NicheStack TCP/IP and compiling of the top application. The system test results showed that the design met the requirement of the Ethernet communication rate at the data rate of about 400 Mb/s, and could be flexible configurated according to actual situation.

Key words: NicheStack TCP/IP protocol stack; ethernet controller; FPGA; SOPC; μC/OS-II

收稿日期:2014-06-18

作者简介:刘黎明(1982—),男,河南省商丘市人,商丘师范学院助教,主要研究方向为嵌入式系统应用及复杂系统仿真.

0 引言

随着网络技术的发展,网络通信技术的应用日益广泛,将嵌入式设备和以太网技术结合[1]进行高速数据传输、远程实时监控及远程视频会议等,已经成为嵌入式领域的研究热点,而这些研究都对网络性能、实时性和灵活性等提出了严苛的要求.

目前,传统的嵌入式网络系统多采用 8 b 或 16 b 的低速控制单位(MCU),在运行协议栈时,可靠性和实时性都难以保证. 以 ARM7 为例,在没有其他任务占用资源时,CPU 向 PC 传送数据的最大传输速率仅为 31 Mb/s. 文献[2]基于以 LPC2478 嵌入式处理器为核心的硬件平台实现了以太网通信,但只适用于小型协议;文献[3]基于 SAMSUNG S3C44B0X 微处理器和 RTL8019S,通过 Socket 编程在嵌入式 uCLinux 系统上实现了网络通信,但仅适用于系统网络数据较小且传输速率为 10 Mb/s 的以太网传输. 在以上方案中,如果处理器在同时执行其他运算和控制任务,数据传输速率将会大幅减小. 随着电子技术的发展,研究者开始尝试采用可编程逻辑器件实现网络数据的传输、交换、控制等功能[4-5]. 本文拟将性价比高、配置灵活的现场可编程门阵列(FPGA)用于以太网控制器设计,采用 NicheStack TCP/IP 协议栈和 Nios II 处理器,通过定制 IP 核实现整个系统,以提高系统实时性和扩展性.

1 NicheStack TCP/IP 协议栈

NicheStack TCP/IP 是专用于嵌入式系统的精简协议栈,具有移植简单、可靠性高和代码精简等特点,包含 Socket API,ANSI C 代码和基本网络通信功能,是 TCP/IP 组的 smal_footprint 实现. 目前,NicheStack 支持 ICMP,IP,TCP,UDP 等多种协议和服务[6]. NicheStack TCP/IP 协议栈具有以下特点:

1)占用内存少. Boot 最小启动客户(包括 ARP,IP,ICMP,UDP,DHCP 和 TFTP)只占12.8 KB,完整的 TCP/IP 只需 42.4 KB,增加 Socket API 需要 51.5 KB.

2)支持两种任务模式,即支技主循环查询方式和任务挂起/恢复方式.

3)通用性的内存管理. 在使用内存时,利用宏定义对其进行分配和释放,分配时其大小固定.

4)与 RTOS 无关.

5)可靠性高,在数据通信中,TCP 数据包传输只受传输介质的频带宽度限制.

6)可支持多种网络硬件,包括令牌环网、以太网和 SLIP.

2 系统硬件平台的搭建

本设计以 Cyclone II 系列的 FPGA 为核心硬件平台,采用嵌入 FPGA 的 Nios II 软核为主处理器,基于 Altera 提供的 Enternet IP 协议,采用 DAVICOM 公司的 DM9000 网络控制芯片为以太网控制芯片(该芯片具有通用处理器接口,1 个 10/100 PHY 和 4 KB 的 SRAM,可实现网络协议中 MAC 层和传输层的功能,是一款低功耗高性能的控制器),其他硬件还包括 JTAG 接口、Flash 存储器接口等. 系统结构如图 1 所示.

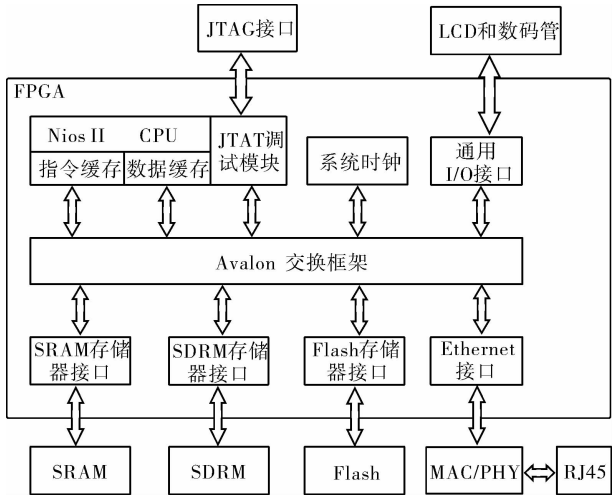

图 1 系统结构图

系统硬件平台设计过程如下:

首先,在 SOPC builder 中构建 SOPC 系统,包括 Nios II 软核 CPU、存储器接口、以太网接口等,这也是系统的主要硬件部分,并通过 Avalon 总线将它们相连;其次,将 SOPC builder 生成的原理图模块进行布局连线设计;最后,由系统生成配置文件,下载到开发板上. 由上述过程生成的以太网 SOPC 系统如图 2 所示.

3 软件设计

本系统软件设计主要包括三部分,即μC/OS-II 移植、NicheStack TCP/IP 网络协议栈的移植、顶层应用程序的编写和实现. 其软件结构如图 3 所示.

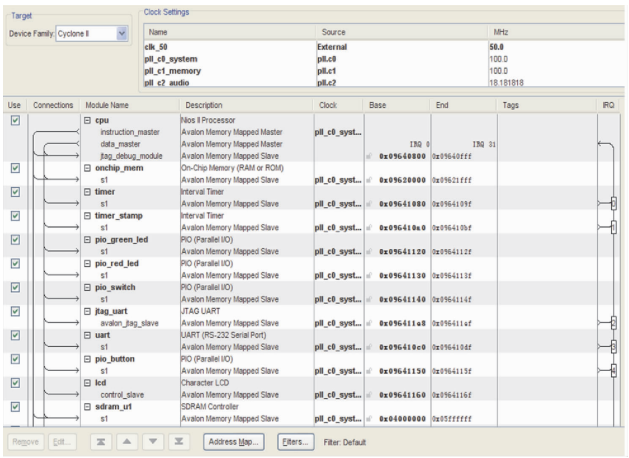

图 2　以太网 SOPC 系统结构图

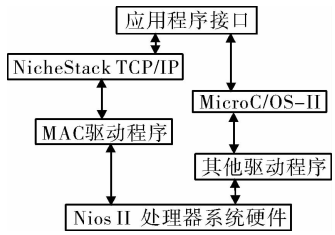

图 3　以太网 SOPC 系统软件结构图

3.1　μC/OS-II 移植

μC/OS-II 移植就是将 μC/OS-II 实时内核移植到特定的微处理器,运行 μC/OS-II 时需提供源代码和系统运行软件[7-8].

μC/OS-II 是一个多任务调度器,添加了与多任务操作系统相关的信号量、邮箱等服务. 其中,实现多任务之间的切换是移植过程的主要工作,因为这部分代码是用来恢复和保存处理器现场的,因此只能使用处理器的汇编语言完成,而不能用 C 语言完成[9].

本设计将 μC/OS-II 移植到 Nios II 软核上,需要对与 Nios II 体系结构相关的 OS_CPU. H 文件、OS_CPU_C. C 文件和 OS_CPU A. ASM 文件进行修改. μC/OS-II 通过以上文件实现任务切换和中断机制,通过编写相应的中断响应函数使系统在目标处理器上运行.

μC/OS-II 建立在安全线程 HAL 系统库上,与 Nios II 系统兼容,是系统软件设计和 Nios II 系统的一部分. HAL 提供了 μC/OS-II 在 Nios II 上运行所需的驱动,因此 μC/OS-II 不仅可以使用 HAL 服务,而且可以调用 API 函数. 对于 Nios II 处理器而言,μC/OS-II 作为 HAL 库的一个子集,是 HAL 环境的一种扩展,包括相应的 μC/OS-II API 功能和 μC/OS-II 系统的调度[10-11]. 利用 Nios II IDE 对 RTOS 模块进行配置,其结果保存在 OS_CFG. H 文件中,无需编写头文件或源代码.

3.2　NicheStack TCP/IP 协议栈的移植

本文采用 NicheStack TCP/IP 协议栈作为系统的数据传输协议,通过该协议栈调度实现系统通信功能. NicheStack TCP/IP 在 Nios II 处理器实现移植须满足以下要求:

1)系统程序要在基于 μC/OS-II 的实时操作系统运行;

2)μC/OS-II 的运行需要有时钟控制机制和时钟节拍,建立任务数不小于 4;

3)FPGA 硬件系统需要提供带有中断的以太网接口;

4)必须由专用的定时器件提供系统时钟.

3.2.1　初始化 NicheStack TCP/IP 协议栈　NicheStack TCP/IP 协议栈的初始化可通过两个函数来完成,一个是 alt_iniche_init( ) 函数,另一个是 netmain( ) 函数,并设置 iniche_net_ready 变量为一个非零值. 系统在初始化设备过程中,调用 get_mac_addr( )函数和 get_IP_addr( )函数来完成 MAC 地址、IP 地址的设置.

3.2.2　调用套接字接口　在以太网硬件初始化完成后,程序主要通过定义套接字的结构体来管理套接字的链接. 程序中使用套接字接口访问 IP 协议栈需利用 TK_NEWTASK( ) 函数,它通过调用 μC/OS-II 的 OSTaskCreat( ) 函数来创建线程,且进行其他 NicheStack TCP/IP 协议栈特定的操作. 本设计通过 TK_NEWTASK (&ssstask)来建立网络线程.

3.3　顶层应用程序的编写和实现

通信的软件部分主要通过编写相应的应用程序并调用移植的协议栈库函数来实现. 在通信应用程序设计之前,首先要将与系统有关的以太网协议栈、驱动程序和操作系统移植,通过 Nios II IDE 对其进行加载,然后在 Nios II IDE 中完成系统通信应用程序的设计和编写. 系统软件部分的通信流程如图 4 所示.

系统上电后,整个系统(包括硬件和任务)都需要初始化. 通信的整个过程为:通过任务建立网络连接,首先在网络连接中建立一个 TCP/IP 服务的 socket,通过调用 socket( )函数来创建一个套接字;

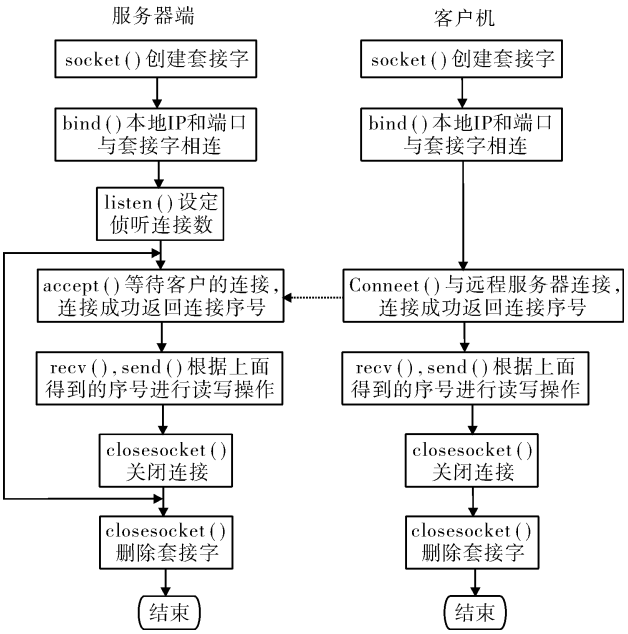

图 4 以太网通信程序流程图

然后,listen( )函数设定侦听的连接数,若此时有来自客户机端的请求,就调用 accept( )函数接受请求,与此同时客户机端则通过 socket( )函数创建一个套接字,通过 bind( )函数将本地 IP 及端口号与套接字相连,再利用 connect( )函数与远程服务器连接,连接成功后,客户机与服务器将通过调用读、写套接字来进行通信,发送数据直至传输完毕后断开连接.

4 系统测试与结果

在本设计中,使用 DE2_70 开发板的快速以太网控制器发送和接收数据包,采用 Nios II 处理器,并通过 DM9000A 以太 PHY/MAC 控制器来发送和接收以太网数据包.

在发送端,Nios II 处理器每 0.5 s 向 DM9000A 发送 64 B 的数据包,DM9000A 接收完成后,给其附上一个 4 B 的校验和,并把它发送给以太网接口,在接收和发送数据时对其进行检验. 符合 IEEE802.3 标准的 CRC 多项式为

$$FCS(s) = x^{32} + x^{26} + x^{23} + x^{22} + x^{16} + x^{12} + x^{11} + x^{10} + x^{8} + x^{7} + x^{5} + x^{4} + x^{2} + x + 1$$

在第 1 个字节最后的重要位,32 位的 CRC(cyclic redundancy check,循环冗余校验)校验值占据着带有 x^{31} 的 FCS(frame check sequence,侦校验). CRC 位按命令 $x^{31}, x^{30}, x^{29}, \cdots, x^{1}, x^{0}$ 接收和发送,在接收端检验是否丢弃.

在接收端,DM9000A 检验每个接收到的数据包,确定数据包的目的地 MAC 地址是否与 DE2-70 开发板的 MAC 地址一致,如果地址一致或这是一个广播包,DM9000A 将接收数据包并发送一个中断给 Nios II 处理器,处理器就会在 Niso II IDE 控制窗口中显示数据包内容. 测试结果如图 5 所示.

通过上位机对数据传输速率进行仿真测试,测试结果如图 6 所示. 由图 6 可知,数据在正确传输过程中能够以 400 Mb/s 的速率正确收发.

图 5 系统测试结果

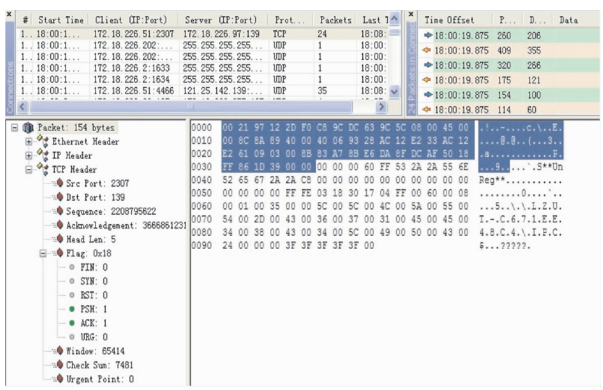

图 6 数据传输速率仿真测试结果

5 结语

本设计使用 Altera Cyclone 系列 FPGA 实现了 NicheStack TCP/IP Stack 协议栈,并移植到嵌入式实时操作系统,测试结果验证了该方法的有效性和可行性:满足了系统实时性和稳定性的要求,充分利用 FPGA 配置的灵活性,使得功能具备较好的扩展性,为今后嵌入式网络控制器在工业控制系统等领

域的应用提供了有效的技术支持.

参考文献:

[1] 杨新华,王用玺,刘欣. 基于 FPGA 的以太网高速数据传输系统的设计[J]. 仪表技术与传感器,2013(12):80.

[2] 王国法,刘炜,段明玮. 基于嵌入式系统的以太网通信开发[J]. 光通信技术,2012(4):36.

[3] 徐海军,刘金刚,王益华. 基于 ARM 核的嵌入式 TCP/IP 协议栈简化实现[J]. 计算机应用研究,2006,23(10):251.

[4] 王鹏,姚明眸,鲍民权. 基于 FPGA 的航空全双工以太网交换芯片[J]. 计算机工程,2008,34(23):261.

[5] Yang H C,Wang F,Zhang J J,et al. Design of embedded tri-mode ethernet based on FPGA[J]. Computer Engineering,2007(15):139.

[6] 李硕. 基于 FPGA 的 SOPC 嵌入式系统设计技术的研究与应用[D]. 北京:北京工业大学,2012.

[7] 赵星星,罗克露,张军,等. 嵌入式实时操作系统移植技术的研究与应用[J]. 计算机工程,2007(17):90.

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

[9] 黄布毅,刘国梅,郑安平,等. 嵌入式家庭网络中央控制器的开发[J]. 郑州轻工业学院学报:自然科学版,2004,19(2):41.

[10] 杨威. 基于 FPGA 的以太网和串口数据传输系统设计与实现[D]. 哈尔滨:哈尔滨工程大学,2013.

[11] 张志刚. FPGA 与 SOPC 设计教程/DE2 实践[M]. 西安:西安电子科技大学出版社,2007.

(上接第 80 页)

果的区别. 将施加均布载荷的三辐板与实际模型三辐板进行对比,发现施加均布载荷的二维模型应力分布与三维模型的应力分布情况相差较远,施加非轴对称载荷的二维模型的应力分布与三维模型的应力分布情况较为接近. 与只施加均布载荷的方法相比,本方法提高了发动机风扇盘二维有限元模型应力分布的准确性.

参考文献:

[1] 刘长福,邓明. 航空发动机结构分析[M]. 西安:西北工业大学出版社,2006:60 - 78.

[2] 陈光. CFM56 系列发动机结构设计与研究特点[M]. 北京航空航天大学出版社,2006:38 - 40.

[3] 陈光. 用于波音 787 客机的 GEnx 发动机设计特点[J]. 航空发动机,2010,36(1):1.

[4] Forrester J M. Circular arc multi-bore fan disk[P]. US Patent:6520742B1,2003 - 02 - 18.

[5] 宋健. 多辐板风扇盘拓扑优化方法研究[D]. 南京:南京航空航天大学,2013.

[6] 李伦未,陆山. 基于 ANSYS 的多辐板风扇盘结构优化设计技术[J]. 航空动力学报,2011,26(10):2245.