文章编号:1001 - 9081(2003)09 - 0082 - 03

微型 TCP/IP 协议栈的设计与实现

姚光开,于永棠,柴乔林
(山东大学 计算机科学与技术学院,济南 山东 250061)

摘 要:TCP/IP 协议已经成为当今开放互联网络系统中不可缺少的部分,详细说明了微型 TCP/IP 协议栈的设计方法,它具有紧凑、轻便、模块化、高效和可升级等特点,实现了标准 TCP/IP 协议栈的主要功能,但对系统资源的要求少,可以用于系统资源少的嵌入式系统进行网络通讯。

关键词:微型 TCP/IP;以太网控制器;协议栈
中图分类号: TP393 **文献标识码**:A

Design and Realization of Tiny TCP/IP Stack

YAO Guang-kai, YU Yong-tang, CHAI Qiao-lin
(*School of Computer Science and Technology, Shandong University, Jinan Shandong* 250061, *China*)

Abstract: TCP/IP has been an indispensable part of the current open internetworking system. This article describes the design methods of tiny TCP/IP stack in detail. The tiny TCP/IP stack is compact, light, modularized, efficient and upgradable. It realizes the main functions of the standard TCP/IP stack with little requirement for system resource, so that it can be used for network communication in embedded system with little system resource.

Key words: tiny TCP/IP; Ethernet controller; protocol stack

1 前言

随着 Internet 网络软、硬件的快速发展,互联网用户正在以指数增长。以太网经过 20 多年的发展,已经成为当今互联网络中底层链接不可缺少的部分;TCP/IP 协议栈[1]是一个非常复杂和庞大的系统,它是 Internet 互联网安全可靠通讯的重要组成部分,通常在通用计算机系统和操作系统下实现。而各种家电设备、PDA、仪器仪表、工业生产中数据的采集与控制等设备也正在逐渐地走向网络化,以便共享互联网络中庞大的资源。在某些应用领域,嵌入式设备在价格、体积及实时性等方面,有着通用计算机无法比拟的优点,因此,嵌入式设备的网络化开发有着广阔的前景。为了实现各种小型设备能够通过互联网进行远程监控,除了在网络硬件接口建设方面的开发以外,一个重要的问题就是根据嵌入式系统的资源有限,功能单一的特点开发一套微型 TCP/IP 互联网络协议。如何在价格低廉、资源有限的嵌入式系统的环境中实现网络通讯功能,已经成为嵌入式网络开发人员面对的重要问题。我们设计的微型 TCP/IP 协议栈能满足这一需要,能够完成常用的网络通讯功能。

2 微型 TCP/IP 协议栈的特点和实现

2.1 微型 TCP/IP 协议栈的特点

TCP/IP 协议栈已经成为开放系统互联的标准,与其它任何协议相比,TCP/IP 协议能够提供更好的交互操作性能,可以在大多数系统中使用,并且它可兼容多种网络技术,因而 TCP/IP 非常流行并且已经得到了广泛使用。微型 TCP/IP 协议栈具有 TCP/IP 协议栈的最基本功能,运行于以太网环境下,底层可以处理 ARP 请求;IP 层支持 ICMP、TCP 和 UDP 三种协议。

收稿日期:2003 - 03 - 24

作者简介:姚光开(1974 -),男,山东德州人,讲师,主要研究方向:计算机网络应用; 于永棠(1973 -),男,山东青岛人,讲师,主要研究方向:网络数据库应用; 柴乔林(1956 -),男,山东青岛人,教授,主要研究方向:计算机网络及应用、神经网络应用.

器在局域网甲,客户端 A 与它在同一个子网内,可以直接与它通讯。客户端 B 和客户端 C 与 DHCP 服务器不在同一个子网,它们与 DHCP 服务器通讯的初始化阶段必须由子网路由器上的转发代理来转发消息包。

5 不足与展望

本系统初步实现了 DHCPv6 的部分功能,但还有一些实用的功能有待完善。比如对 DDNS(动态 DNS)和其它的一些网络配置信息(如 NTP 网络时间协议)的支持还没有,在地址冲突检测方面也没有很好的处理。另外,全状态和无状态地址自动配置这两种方式可以配合使用,由无状态配置协议生成地址,然后由全状态配置协议提供其它网络信息,这样将简化配置,提高网络效率。

参考文献

[1] Droms R, Bound J, Volz B, *et al*. Dynamic Host Configuration Protocol for IPv6(DHCPv6)[S]. draft-ietf-dhc-dhcpv6-28.txt, 2002.

[2] Droms R. Dynamic Host Configuration Protocol[S]. RFC2131. IETF, 1998.

[3] Ldshin P. IPv6 详解[M]. 沙斐,等译. 北京:机械工业出版社, 2000.

[4] Naganand D, Dan H. IPSec ——新一代因特网安全标准[M]. 京京工作室,译. 北京:机械工业出版社, 2000.

其中 ICMP 协议能对 ECHO 命令响应,TCP 协议提供了一个快速的、可靠的连接。高层协议比如 HTTP、TELNET 等能够稳定运行于 TCP 之上。各层分别采用独立的模块,整个系统具有紧凑、轻便、模块化、高效和可升级等特点。由于微型 TCP/IP 协议栈主要应用于基于单片机的嵌入式系统,该系统的各种资源有限,如 CPU 的处理速度、字长、RAM 和 ROM 存储器的容量,接口数量等和通用计算机相比有很大的差距,因此,如何使微型 TCP/IP 协议栈做到精细、通讯可靠、功能相对完善,而且又能发挥单片机的特点成为微型 TCP/IP 协议栈设计的关键问题。为了减少系统的资源开销,微型 TCP/IP 协议栈在功能上根据需要进行了精简,我们作了如下限制:

1) 每次仅仅进行一个 TCP 或 UDP 会话;
2) 不能重组接收的 IP 帧;
3) 在错误的顺序号下不能缓冲 TCP 数据段;
4) 不支持服务类型 ToS 和安全选项;
5) 忽略接收的 TCP 选项。

2.2　微型 TCP/IP 协议栈的实现

2.2.1　软件模型

我们使用 4 层实现 TCP/IP 协议栈软件,如图 1 所示。为了有效利用单片机系统有限的资源并提高效率,各层协议的具体实现和通常的 TCP/IP 协议相比作了较大的简化,但它具有最基本的功能。

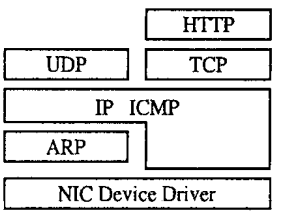

图 1　微型 TCP/IP 协议栈软件模型

最底层是物理层,它定义了以太网控制器的工作方式,以太网帧的封装和分析、发送以及接收等,我们使用了 RealTek 公司的以太网控制器芯片 RTL8019AS。

第二层是 Internet 层,它完成了 IP 数据报的封装和 IP 头的分析,并根据帧的类型(ICMP、TCP 或 UDP 等)进行相应的处理。

第三层是传输层,主要是 TCP,它对 Internet 提供点到点服务,在这层上只允许单个连接。它实现了 RFC793 中相应的功能,包括三次握手建立连接,各种状态之间的转换,超时重传,连接的撤销等。

最上层是应用层,解决用户特定的应用,而不用关心底层的具体实现。在我们的系统中实现了一个简单的 Web 服务。

2.2.2　硬件框架和工作过程

系统硬件主要由图 2 所示部分组成:RJ45 接口连接至以太网络,以太网控制器完成以太帧的接收与发送;外部扩展 RAM 用来存放接收与发送数据的缓冲区以及系统中使用的变量;EEPROM 用来存放 MAC 地址、IP 地址、子网掩码、默认网关等信息。系统的工作过程如图 3 所示。当系统加电时,系统首先读取保存在 AT24C01 EEPROM 中的 MAC 地址、IP 地址、子网掩码和默认网关等参数。然后对 RTL8019AS 以太网控制器进行初始化,初始化完成后,系统进入监听状态,被动地等待数据的到达。当 RTL8019AS 接收到请求数据包后,引发系统产生外部中断,系统主机开始接收数据包,根据以太帧(即刚刚接收的数据包)中包含的帧的类型分别进行 ARP、ICMP、TCP 和 UDP 的处理,并准备相应的回答帧,交给以太网控制器,由以太网控制器发送至以太网络。处理结束后,系统继续等待接收下一个数据包。如此循环。

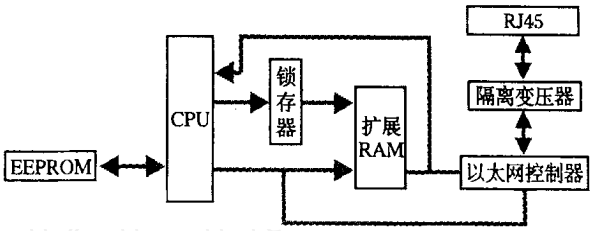

图 2　微型 TCP/IP 系统硬件框图

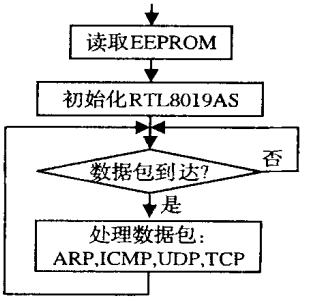

图 3　系统的工作过程

CPU 使用 ATMEL 的 89C52,具有 8K 的片内 ROM,足以存储可执行代码,不需再使用外部 ROM。我们使用 Keil C51 进行编程,程序中的变量主要在 CPU 的内部 RAM 分配,从而使得系统的速度得以提高并充分利用系统有限的资源。用于存储以太帧的缓冲区以及读写 EEPROM 的数据结构则在外部 RAM 中分配。

3　RTL8019AS 以太网控制器[2,3]

RTL8019AS 是一个高度集成的、全双工的、低功耗的 10MHz 以太网控制器,具有即插即用和 NE2000 兼容的性能。对 RTL8019AS 的初始化设置是系统运行的重要组成部分,RTL8019AS 是通过内部寄存器的设置进行工作的,首先介绍一下 RTL8019AS 的几个重要寄存器:

1) CR(命令寄存器):用于选择寄存器页(RTL8019AS 的寄存器组分布在四个页中),使能或禁止远程 DMA 操作,并且发出命令,它是 RTL8019AS 内部最重要的寄存器。

2) ISR(中断状态寄存器):反映了以太控制器当前的状态,CPU 通过读取它判断引起中断的原因。

3) IMR(中断屏蔽寄存器):各位分别控制 ISR 各位,设置 IMR 的单独位,可以响应相应的中断,从而置位 ISR 相应位。

4) DCR(数据控制寄存器):控制 FIFO 缓冲区的大小,远程 DMA 是否自动初始化,字节顺序选择,DMA 字节/字传输模式选择。

5) TCR、RCR(发送,接收控制寄存器):控制接收、发送的方式。

6) PSTART、PSTOP、BNRY、CURR:这四个寄存器设置接收缓冲区(RTL8019AS 有 16K 的 RAM,每 256 字节称为一页,页地址为 0X40～0X7F)。PSTART 设置起始页,PSTOP 设置停止页,这两个寄存器设置了接收缓冲区的首尾;BNRY 指示最后一个被取走的缓冲区页,CURR 指示第一个用于接收的缓冲区页,这两个寄存器保证接收缓冲区不会溢出。

7) TPSR、TBCR0、TBCR1:TPSR 设置发送缓冲区的起始页,

TBCR0、TBCR1 设置发送字节数。

8) RSAR0、RSAR1、RBCR0、RBCR1：RTL8019AS 通过远程 DMA 和系统交换数据，前两个寄存器设置远程 DMA 的起始地址，后两个设置远程 DMA 数据字节数。

4 功能函数[4]

各个协议处理函数所处理的协议帧由两个全局无符号字符数组存放，一个用于接收数据，一个用于发送数据。因为单片机的资源有限，所以并不是采用各类通用操作系统使用的 MBUF 数据结构；而且内部 RAM 只有 256 字节，因此把它分配在扩展的外部 RAM 中。下面简单介绍几个主要的 API。

4.1 物理层(Ethernet Layer)

以太网控制器初始化函数 Init8019：它用来复位以太网控制器(包括硬复位和软复位)，初始化 RTL8019AS 的寄存器组。设置 RTL8019AS 的工作方式，中断的响应，发送、接受缓冲区的开始地址、大小，MAC 地址等。

```
reset = 1;                                          //硬复位
for(i = 0;i < 256;i ++);                            //delay for some times
  reset = 0;
  ByteRead = RSTPORT;                               //软复位
for(i = 0;i < 256;i ++);                            //delay for some times
  RSTPORT = ByteRead;
for(i = 0;i < 256;i ++);
  CR = 0x21;                                        //选择页 0，网卡停止工作
  DCR = 0xc0;                                       //8 字节 FIFO，lookback mode
  RBCR0 = 0;RBCR1 = 0;                              //清除远程 DMA 字节计数器
  RCR = 0xc0;
  //可以接收广播和发送至本机的包，包的长度大于 60
  TCR = 0xe4;                                       //Lookback mode
  PSTART = 0x46;                                    //接收缓冲区开始页
  PSTOP = 0x60;                                     //接收缓冲区停止页
  BNRY = 0x46;                                      //BNRY
  ISR = 0xff;                                       //清除中断状态寄存器
  IMR = 0x11;                                       //允许 PRX OVW 中断
  CR = 0x61;                                        //page 1
  CURR = 0x46;                                      //当前页寄存器
  PAR0 = MYMAC[0];                                  //设置 MAC 地址
  PAR1 = MYMAC[1];
  PAR2 = MYMAC[2];
  PAR3 = MYMAC[3];
  PAR4 = MYMAC[4];
  PAR5 = MYMAC[5];
  CR = 0x22;                                        //page 0，启动网卡工作
  TCR = 0xe0;
```

接收数据包(以太帧)的函数 ReadETHFrameFrom8019：它从以太网控制器的接收缓冲区通过远程 DMA 接收数据。

发送数据包的函数 SendETHFrameTo8019：通过远程 DMA 向以太网控制器的发送缓冲区写入数据，并发出发送命令(CR = 0X24)，指示以太网控制器向网络发送。

4.2 网络层(Internet Layer)

DoNetWork：当 RTL8019AS 接收到正确的数据包时，调用 ReadETHFrameFrom8019 接收，然后处理：如果是 ARP 请求包，则发送一个 ARP 响应包；如果是 IP 包，则根据 IP 包的类型(ICMP、TCP、UDP)，调用相应的处理函数。最后，如果有数据包发送，调用 SendETHFrameTo8019。

```
switch(ProtocolType)
{
 case IP-PROTOTYPE-ICMP: ProcessICMP();break;
 case IP-PROTOTYPE-TCP: ProcessTCP();break;
 case IP-PROTOTYPE-UDP: ProcessUDP();break;
}
```

ProcessARP：它和通常的 ARP 协议不同，只是响应 ARP 请求，而不会主动发出 ARP 请求。在完成 ARP 响应帧的封装后，设置发送标志，等待 DoNetWork 处理。

ProcessICMP：它只是完成 PING 命令应答帧的封装，设置发送标志，等待 DoNetWork 处理。

4.3 传输层(Transfer Layer)

TCPListen：设置系统被动监听。

PrepareTCPFrame、PrepareTCPDataFrame：分别完成 TCP 控制帧、TCP 数据帧的封装，并设置响应的发送标志。

ProcessTCP：完成 RFC793 描述的主要功能。TCP 协议建立连接和释放连接所需的步骤用 11 种状态表示：CLOSED，LISTENING，SYNRCVD，SYNSEND，ESTABLISHED，FINWAIT1，FINWAIT2，TIMEDWAIT，CLOSING，CLOSEWAIT，LASTACK。每个连接均开始于 CLOSED 状态。当一方执行了主动的连接或被动的连接时，便会脱离 CLOSED 状态。如果此时另一方执行了相应的连接，连接便会建立，并且状态转变为 ESTABLISHED 状态，此时数据可以传输。任何一方均可以首先请求释放连接。当连接释放后，状态又回到 CLOSED。如图 4 所示(实线代表客户的正常路径，虚线代表服务器的正常路径，每条线上均标有事件/动作)。数据传输时，首先检查是否有超时的 TCP 数据包，若有，再检查重传次数是否大于设定值，若是，释放连接，否则重传；然后接收，处理；接下来根据不同的状态调用 PrepareTCPFrame 或者 PrepareTCPDataFrame 进行数据封装，设置相应的发送标志，并启动相应的定时器计时。

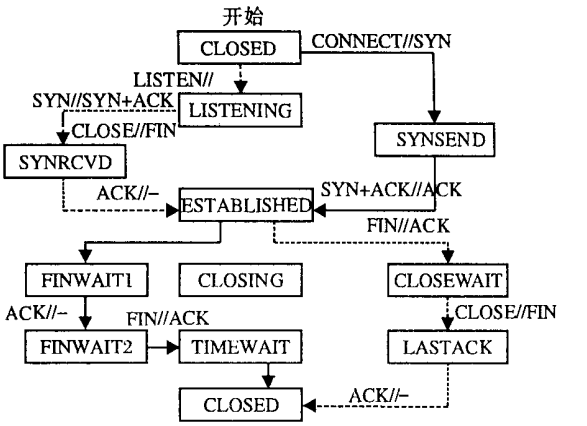

图 4 TCP 连接管理状态

5 小结

该文详细地说明了微型 TCP/IP 协议栈的特点和设计方法，它适用于资源较少的嵌入式系统，如 ROM、RAM 和接口很少的 8 位或 16 位处理器，利用它可以实现一些简单的远程监控系统，由于采用单进程，因此每次仅仅支持一种服务。如果实现较复杂的远程控制，需要增加 ROM 和 RAM 以及支持多进程；如果支持图形，需要加入虚拟文件系统。

参考文献

[1] (美) Comer DE. 用 TCP/IP 进行网际互联(第二卷)：设计、实现与内核[M]. 张娟，王海，等译. 北京：电子工业出版社，2001.

[2] http://www.laogu.com/download/rtl8019as.pdf[EB/OL]，2003 - 01.

[3] http://www.national.com/ds/DP/DP8390D.pdf[EB/OL]，2003 - 01.

[4] Tanenbaum AS. 计算机网络(第三版)[M]. 熊桂喜，王小虎，译. 北京：清华大学出版社，1998.