维普资讯 http://www.cqvip.com

EI内核中TCP状态机约简的研究*)

林 涛 王济勇 韩光洁 王金东 赵 海
(东北大学信息科学与工程学院 沈阳110006)

摘 要 在嵌入式Internet(EI)环境中,可以利用的资源十分有限,而要在这种条件下实现Internet接口是很困难的,通常的做法就是扩充系统资源,但是这种方法增加了系统成本,大材小用。而且嵌入式Internet设备将要最终嵌入到设备中去,因此必须要在尽量小的空间内实现所需要的功能。本文提出一种TCP状态机的约简原则,减少状态机的处理流程,将复杂状态变换转化为简单变化,将状态机中的状态数和跳转条件进行约简与合并,提高了性能,减少实现代码。在不改变协议接口的前提下在原有的资源空间内实现通用的TCP/IP协议栈,并保证了协议功能上的完备性。根据这套约简原则,在一个8位单片机上实现了一个功能完善、体积微小的嵌入式Internet内核,使其可以嵌入到任何一个小型设备和家用电器中。

关键词 嵌入式Internet,TCP,状态机,约简,Web server

Reasearch on TCP State Machine Reduction in Embedded Internet Kernel

LIN Tao WANG Ji-Yong HAN Guang-Jie WANG Jin-Dong ZHAO Hai
(School of Information Science and Engineering, Northeastern University, Shenyang, 110006)

Abstract In Embedded Internet (EI) environment there are limited resources, so it is very difficult to realize a Internet in EI environment. In general some high performance processor and memory are used to solve the problem. It is clearly is the wrong way as it incurs additional development time, testing time and price. In this paper a proper method is presented which is based on TCP state machine reduction. In TCP/IP protocol suite TCP is the most complex protocol and consumed most resources. We delete unimportant and unnecessarily state in EI environment and simplify complex relations. All these reduction are complete transparent for user because the interfaces are untouched. According to the reduction principle proposed in paper we realize an mini EI kernel in a 8 bit processor, in this kernel there are full function Web Server and correlative protocols.

Keywords Embedded Internet, TCP, State machine, Reduction, Web server

1 引言

在传统的嵌入式系统中,用户在本地可以方便地完成对设备的管理和控制。随着需求的发展,用户希望远程访问和控制嵌入式设备,有时候一个嵌入式系统中不止具有一个芯片,这就要求系统中不同部件之间可以进行通信,传统的方法一般采用系统总线,例如I^2C总线,CAN总线等,但是它们的缺点是有距离限制,实现起来比较复杂,而且是专用网络环境,不能和其他网络兼容[1]。随着Internet技术的快速发展,大量的计算设备联入互联网,而且随着基于IP协议的标准制定,互联网内的协议达成了一致,这使得用户之间的接口互相通用,Internet成了互联全球的技术。对于嵌入式芯片来说,它们并不一定被应用于路由器或交换机等具有Internet接口的设备上,也许是家用电器或工业设备,而它们没有与Internet的接口,因此,给传统的嵌入式设备一个Internet接口是目前的迫切要求,传统的嵌入式系统也延伸为嵌入式Internet(EI)系统,因为这样给了开发者和使用者一个通用的界面。

2 传统解决方案及其局限性

传统的嵌入式技术对Internet接口的解决方案是在原有内核的基础上,将TCP/IP协议栈完整地搬移过去,这样的作法在功能上解决了Internet接口问题,但是在本地资源上却造成了浪费,因为TCP/IP协议栈中包含了许多协议,并且有些协议很复杂,完整的移植必然占据大部分的系统空间,用户因此不得不使用高端处理器来解决资源不足的问题,而且普通的TCP/IP协议栈并不是为嵌入式Internet专门设计的,没有考虑嵌入式开发的特点,如实时性。这样造成的后果必然是使嵌入式服务器"胖"起来,这不但违背了嵌入式系统的设计规则,也造成了使用中的诸多不便。而本文提出的是一种"瘦"的TCP/IP协议栈,尤其是在对TCP协议的裁减,使其更加适合于嵌入式系统的应用,并且节省了大部分编程空间。

3 TCP/IP协议栈的分析及裁减

要实现嵌入式Web Server的功能,首先必须在其底层实现TCP协议以及服务器端的HTTP协议,HTTP服务器进程只是简单返回客户进程所请求的文件,因此HTTP服务器软件比HTTP客户软件要小得多,例如,NCSA版本1.3的Unix服务器由大约6500行C代码写成,而X Window环境下的Unix Mosaic2.5浏览器有约80000行代码[5]。以上分析可以看到,主要的工作集中在TCP协议的实现上,因此,能否在嵌入式芯片的少量资源上实现Web功能,主要看是否能够实现TCP协议。嵌入式Internet应用主用针对设备和家电的控制问题,控制命令一般比较简单,因此复杂的管理协议可以去除,而使用简单有效的传输协议来完成控制功能。

*)本课题得到国家自然科学基金资助(69873007)。林 涛 博士,主要研究领域为嵌入式Internet。赵 海 教授,博士生导师,研究方向为嵌入式Internet,数据融合。

维普资讯 http://www.cqvip.com

3.1 TCP 状态机的约简

TCP 是传输层一个重要的面向连接的、可靠的字节流服务的传输层协议，它是实现 Web Server 的基础，通过分析可以得知它的大部分工作量集中在两点：第一：维持连接，包括连接的建立，保持和断开；第二：确保数据发送的可靠性。由于连接的维护，使 TCP 协议存在多种状态，在不同状态之间进行切换，要保证系统的完备，不致进入死锁状态。为了确保数据的可靠发送，TCP 协议维护了几种定时器和超时处理功能，这些定时器分为保活定时器，超时重传定时器，坚持定时器等。下面就针对这两方面给出约简规则。

普通的 TCP 状态机转换图如图1所示，图中给出了 TCP 协议的状态和它们之间主要的转换条件。从图中可以看出，状态图中的状态比较多，大部分都集中在断开连接和超时重传部分，而在正常的 TCP 连接建立起来之后，传送字节流的过程却很简单，鉴于此，提出了下面的约简的原则：

(1) 减少状态机中的状态个数。

(2) 尽量减少超时状态在状态机中出现的机会。

(3) 减少每个 TCP 连接的保持时间。

(4) 将状态机中的双向输入输出简化为单向。

在状态转换图1中，状态 close 并没有很大的作用，它不是一个真正的状态，而是这个状态图的假想起点和终点，所以根据原则(1)可以将其拿掉。在图1所示的状态机转换图中支持同时打开和同时关闭功能，两个应用程序同时彼此主动打开和关闭的情况尽管发生的可能性极小，但也是可能的[8]。这种情况下状态机的转换关系变得复杂，因为它打破了正常的状态变迁，增加了不稳定的因素，如果没有大量工作对状态转换进行检测，很容易导致死锁以及资源的无限期阻塞[9]。在 TCP 状态机约简过程中，根据原则(5)在服务器端已经发出断开请求后，把客户端的突发性请求用状态机函数进行判断，不进行处理，而使状态流图方向按照正常的状态流动。根据以上原则及方法，对 TCP 状态机进行合理的约简，保证 TCP 协议的功能完整性，得到新的状态机状态转换如图2所示。

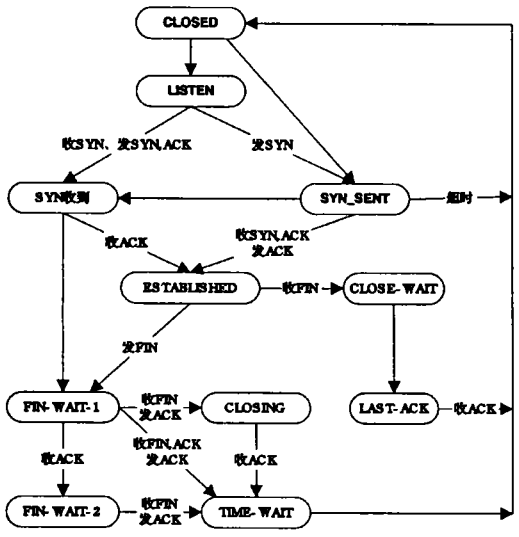

图1 一般 TCP 协议状态机模型

作为 Web 服务器，其主要的功能就是通过 HTTP 协议获取用户所需的主页文件，在这类操作中，用户的交互非常多，客户端的浏览器可以在任何时候关闭浏览器，或者客户端主机发生故障，如停电、死机等现象，而嵌入式 Internet 服务器上的资源很有限，可以建立的 TCP 连接数也很有限，如果在客户端异常断开连接后，仍然保持足够长的时间，大部分的 TCP 连接控制块就处于等待状态，对与请求建立连接请求的报文就无法进行答复，直接影响了服务器的性能，因此，应该根据原则(2)，对超时重传的操作次数进行缩减。

3.2 定时器的约简

TCP 是一个可靠的传输层协议，为了保证可靠性，它使用了许多定时器，而定时器是嵌入式系统中比较关键而且缺少的资源，如果不能有效使用，会造成系统的混乱。在连接过程中 TCP 协议使用了一个保活定时器的功能，以避免在客户端异常消失的时候在服务器上留下一个半开放连接，而服务器却又在等待来自客户的数据，这样服务器将永远等待下去，在本文的方法中，将保活定时器的时间适当地缩短，在一般的保活定时器中，保活时间默认为两个小时，在本文的方法中，将保活时间改为两分钟，因为嵌入式 Web Server 一般工作在以太网环境中，网络延迟比较小[7]，如果探测两分钟仍然没有客户端反馈消息，那么可以认为是异常断开了，在 Internet 环境中，虽然可能做出错误判断，但由于客户端是功能强大的 PC，远比嵌入式设备的资源丰富，功能也要强，可以很快弥补这一失误，并马上进行新的连接操作，这种方法就是在不影响协议的前提下将服务器端的工作转移到客户端。

由于同时访问嵌入式 Web Server 的用户可能不止一个，因此对每个连接的状态都要进行记录，这个工作一般是由 TCP 传输控制块来完成的，在本文的模型中也不例外，由于嵌入式系统的特点，我们对 TCP 控制块的个数进行了动态的控制，并以数组的形式组织控制块的结构，这样可以很方便地增减控制块的个数。当一个连接占用了某个控制块后，其他连接就无法使用它，当所有的 TCP 控制块都被占用之后，就不能建立新的连接了，因此，另一个约简的原则就是尽量减少 TCP 控制块的占用时间。在本文提出的方法中，就采用了这种方法，在客户与服务器交互过程中，每一个报文都要进行确认，然而对于最后一个报文，对于服务器来说，只要在收到后给客户端发送一个 ACK 确认帧即可，但是一般的 TCP 状态机中(如图1)，要进入一个 TIME_WAIT 状态，这是一个非常容易出现延迟的状态，如果确认报文丢失了，而客户端又没有重新发送 FIN 请求，则要等待一个2MSL 时间[6]，这个时间根据 RFC793可知为2分钟，那么对于 TCP 模型来说要等待4分钟，这对一个嵌入式服务器来说是非常可怕的，因为在这个时间内，它所控制的设备很有可能出现动作，而 Web 服务器却无法响应，因此，在本文的 TCP 状态机中就省略了此状态，而直接释放 TCP 控制块回到 LISTEN 状态，这样做既减少了超时状态出现的机会，又将 TCP 的连接时间降低为最短，同时贯彻了约简原则(2)和(3)。

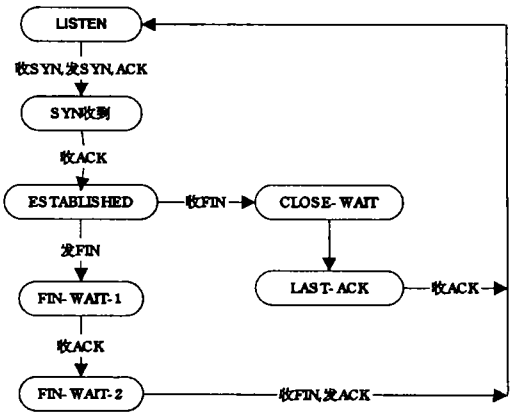

图2 约简后的状态机模型

维普资讯 http://www.cqvip.com

3.3 裁减后状态机的完备性和可扩展性

在本文提出的方法中，将 TCP 状态机进行了约简，将不必要和很少用到的状态筛选出去，最终得到一个适合于资源非常有限的嵌入式 Internet 应用环境的状态机，由于 Internet 是一个世界范围内的网络，它的接口也是标准的[10]，所以，无论对 TCP 状态机做了多少修改，对最终用户来说应该是透明的，用户应该不知道他访问的是一个嵌入式 Web Server 还是一个普通 PC 上的 Web Server，这就要求在约简的时候做到接口和功能的完备性，本文提出的约简方法保留了协议的入口和出口完整性，将耗费资源的状态转换划归到异常处理部分，例如，在同时断开连接时，客户端不会感觉到服务器端没有处理他发出来的 FIN，而是认为这个报文丢失了，在它等待的这段时间内，由服务器发出的断开请求 FIN 已经到达客户端，有效地避免了同时断开给嵌入式服务器增加的负担。状态机函数处理流程如图3所示。

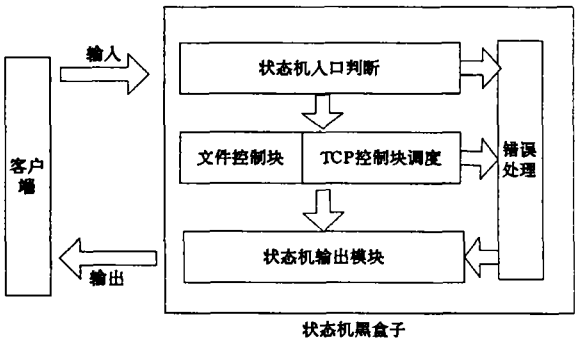

图3 状态机函数处理流程

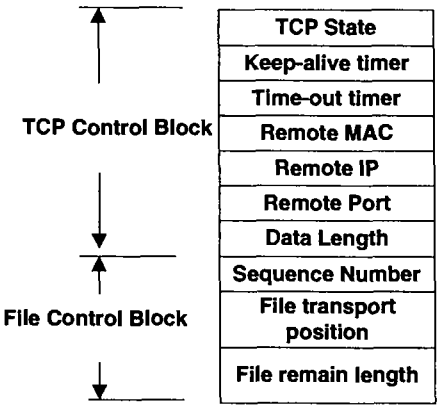

图4 可扩展的 TCP 控制块

由于嵌入式 Internet 设备是嵌入到其他设备中去的，它的应用是和具体的设备相关的[11]，因此针对不同的设备就要开发不同的应用，而嵌入式 Internet 内核也是与它所采用的芯片密切相关的，所以，在嵌入式 Internet 系统中系统的升级很重要，而且经常随着设备性能的提高而采用不同的芯片，在本文的方法中，充分考虑到了这一点，所有设计采用模块化，在 RAM 扩大后，只要简单地修改几个参数即可使用。例如，本文的 TCP 的设计是面对多用户多连接的，为了区分不同的连接采用了 TCP 控制模块，可是由于芯片内 SRAM 的限制，模块中只开辟了十个连接，在每一瞬间只能有十个 TCP 连接，和它相对应的是十个文件传输控制模块，这两个模块为线性组织方式，所以由首地址可以遍历到每一个控制块，在资源得到扩展后只要修改控制块的个数参数即可，而内核的其他功能模块并不受到影响。

4 性能分析及结论

使用本文提出的 TCP 状态机约简方法，我们在一个8位的芯片内实现了一个嵌入式 Web Server，它的资源条件是可编程 Flash 8kbyte，时钟频率为8M，网卡芯片10M，SRAM 为512字节，在这个芯片内实现了 TCP/IP 协议栈内的大部分协议，如 ARP，RARP，ICMP，UDP，TCP 等，其中以 TCP 协议为主，占用编程空间为内核总空间的30%，为了检验利用本文提出的状态机约简方法实现的内核的有效性及可靠性，采用了以下方法进行分析测试。

测试1：速度测试

通过浏览器访问嵌入式 Web Server 的主页。我们将所有 Web 页面及相关文件存放在 EEPROM 中，大小仅为30k，利用抓包工具分析 TCP 数据包的传输情况。由于浏览器对不同文件的解释时间不一样，我们选用了三种具有代表性的文件进行测试，对每种数据都是在不同的时段进行若干次试验求得其平均值，所得数据如表1所示。

表1 裁减前内核速度测试结果

文件种类	总文件大小 (kb)	文件个数	传输时间 (s)	传输速度 (bps)
纯文本	30	1	21.36	11.24
HTML 及图片	28.4	6	20.35	11.16
HTML 及 Java Applet	28	4	28.48	7.87

表2 裁减后内核速度测试结果

文件种类	总文件大小 (kb)	文件个数	传输时间 (s)	传输速度 (bps)
纯文本	30	1	18.29	13.12
HTML 及图片	28.4	6	17.91	12.64
HTML 及 Java Applet	28	4	25.45	8.8

表3 稳定性测试结果

ICMP 包个数	TCP 包个数	ICMP 丢包率	TCP 访问次数
86400	207360	0	69120

测试2：健壮性测试

由于我们实现的嵌入式文件系统比较小，因此在一个用户终端文件系统当主页显示到浏览器上以后，我们使用了 Java Applet 来与后台的嵌入式 Web Server 来进行通信，由于控制信息很少，传输速度并不是问题，而是要检验 TCP 的状态转换是否合理，以及控制块的分配和释放是否能造成系统资源死锁。测试时间为24小时，用户端使用客户端软件每隔5秒访问 Web Werver 的主页，同时打开10个 console 来向 Web server 发送 ping 请求，这是为了制造复杂的网络环境，人为的给 Web Server 的网络端制造干扰。测试结果见表格3。由此可见内核的结构很合理，而且也适合于网络条件恶劣的环境下使用。

性能分析：设 X，Y 为传输数据速度的随机变量，X 为传输文本数据的速度，Y 为超文本和 Java Applet 的传输速度，根据概率论，对两组数据分别求其裁减前和裁减后的均值和方差。对随机变量 X，裁减前的均值和方差分别表示为 E

维普资讯 http://www.cqvip.com

($X_{前}$)，$Var(X_{前})$，裁减后均值和方差分别表示为 $E(X_{后})$，方差 $Var(X_{后})$如下：

$$E(x_{前})=\frac{1}{n}\sum_{i=1}^{n}x_i=11.24$$

$$E(x_{后})=\frac{1}{n}\sum_{i=1}^{n}x_i=13.12$$

$$Var(x_{前})=\frac{1}{n}\sum_{i=1}^{n}(x_i-\bar{x})^2=0.004174$$

$$Var(x_{后})=\frac{1}{n}\sum_{i=1}^{n}(x_i-\bar{x})^2=0.001481$$

同样对随机变量 Y 可以得到裁减前后的均值和方差如下：

$$E(y_{前})=\frac{1}{n}\sum_{i=1}^{n}x_i=7.87$$

$$E(y_{后})=\frac{1}{n}\sum_{i=1}^{n}x_i=8.8$$

$$Var(y_{前})=\frac{1}{n}\sum_{i=1}^{n}(x_i-\bar{x})^2=0.059372$$

$$Var(y_{后})=\frac{1}{n}\sum_{i=1}^{n}(x_i-\bar{x})^2=0.004038$$

从以上计算结果中可以看到：

$E(X_{前})<E(X_{后})$　$Var(X_{前})>Var(X_{后})$

$E(Y_{前})<E(Y_{后})$　$Var(Y_{前})>Var(Y_{后})$

从这个结果中可以看到，裁减后的速度均值都有所增加，这说明裁减后的内核速度比裁减前有所提高。而裁减后速度的方差比裁减前有所减小说明了裁减后内核的性能变得比较稳定。

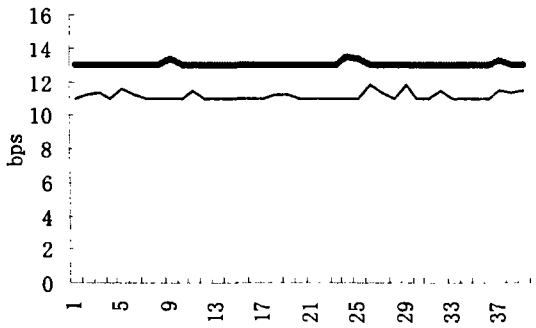

图5　文本数据传输性能曲线图

为了得到更准确的比较数据，对裁减前后的内核使用了多组数据进行多次试验，表1和表2中给出的只是平均速度，为了更清楚地看到裁减前后性能的变化，给出了两种内核的性能比较曲线图，(因为篇幅有限数据的值在此未详细列出)，将多次在不同时间测得的大量数据以曲线图的形式给出，可以清晰地看到内核性能波动情况。曲线1和曲线2中，横坐标表示试验数据序号，也就是样本点的顺序，纵坐标为传输速度，单位为 bps，其中曲线图1是对纯文本数据的传输评价，曲线图2是对超文本及 Java Applet 文件的传输评价。两组曲线图中较粗曲线为裁减后的速度曲线，较细曲线为裁减前的内核速度曲线。从这两组曲线中均可以看出，裁减后的内核性能曲线平均速度有了提高，而速度的波动减小了，波动的减小说明内核的性能稳定性增强了，这主要是由于裁减后的内核将复杂的情况进行了简化处理，提高了内核的效率。波动性的减小在嵌入式 Internet 中有很重要的意义，因为在嵌入式系统中，在不影响性能的前提下，稳定性是衡量一个系统是否可行的标准，这种改善是符合嵌入式应用需要的。

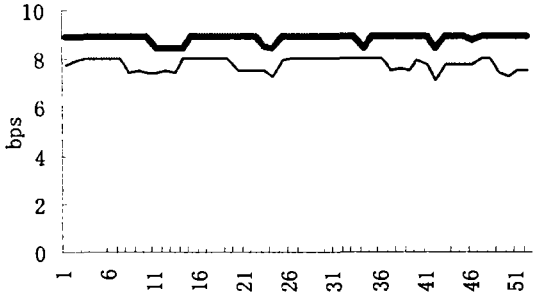

图6　Java Applet 及 HTML 文件传输性能曲线图

结论　从以上数据及图表可以看出，对嵌入式内核的裁减有效地减小了内核的总体积，使系统的有限资源得到了有效的利用，为用户的应用程序开发提供了更大的自由度，而且，在系统性能上，速度有了一定的提高，内核的稳定性有了改进。

参 考 文 献

1 Burton L. Embedded Internet systems: poised for take off[J]. IEEE Internet Computing, 1998, 14(2): 24～29

2 Jacek W. Embedded Internet technology in process control devices[J]. IEEE, 2000, 34(3): 301～308

3 Kasemir K U, Dalesio L R. Interfaceing the controllogix PLC over Ethernet/IP

4 Robert F. Embedded Internet systems come home [J]. IEEE Internet Computing, 2001, 40(14): 52～53

5 Stenens W R. TCP/IP Illustrated Volume 3 [J] China Machine Press, 2000. 130～131

6 Kopetzky T. Visual preview for link traversal on the World Wide Web[J]. Published by Elsevier Science, 1999

7 Yoshida A. MOWS: distributed Web and cache server in Java[J]. Computer Networks and ISDN System, 1997, 29: 965～975

8 Morrison L W, Purves R S. Customizable landscape visualizations [J]. Elsevier Science, 2002

9 Lin C M, Chen T F. Dynamic memory management for real-time embedded java chips[J]. IEEE Network , 2000, 34: 49～56

10 Adnan H. Distributed VRLA battery management organisation with provision for embedded Internet interface [J]. IEEE Computer Internet Computing, 2000, 35(12): 713～720

11 van der Wolf P. System-Level Design of Embedded Media Systems [C]. IEEE International Conference on VLSI Design, 2002, 15: 59～61