基于 ARM9 的 UDP 协议栈的设计与实现

为了满足以太网通信过程中大数据量的快速传输的需求，往往可以牺牲一些可靠性换来高速的数据传输.根据方案，文中设计了一套基于 YLP2440 的 UDP 通信系统，实现了简单实用的 UDP 通信协议.首先介绍了系统整体硬件结构，然后完成了以太网通信系统软件设计，以 DM9000A 以太网卡驱动程序为基础，通过裁剪移植 TCP/ IP 协议栈，实现了系统数据的接收和发送.对系统 UDP 和 ARP 通信进行了测试，结果表明 UDP 通信系统整体稳定可靠，并且系统开销小.数据传输速度快，能够满足实际应用需求.

0 引 言

随着嵌入式技术和网络技术的迅速发展，以太网接口在嵌入式系统中的应用越来越广泛.以太网通信速度快.通用，可直接与 Internet 相连接，提供更大范围的远程访问.目前在工控嵌入式领域，网络通信通常采用 UDP 和 TCP 协议.UDP 与 TCP 相比，UDP 使用非连接的.不可靠的通信方式，因此网络传输速度快，实时性相对较好.文中设计实用 S3C2440.以太网控制器 DM9000 和经过自行裁剪的 TCP / IP 协议栈，构成嵌入式系统的以太网接口，实现 UDP 通信.

1 系统的硬件介绍该系统采用优龙科技公司 YLP2440 作为开发的硬件系统，YLP2440 采用三星 S3C2440A 作为 CPU,最高主频 400MHz,带有 64MB SDRAM 和 64MB NANDFlash 的外部存储器，有两个五线异步串行口，波特率高达 115200bps,一个 10M / 100M DM900AEP 网络接口卡，带有连接和传输指示灯.DM9000A 是一个全集成.功能强大.性价比高的快速以太网 MAC 控制器，它带有一个通用处理接口.EEPROM 接口.10/ 100MPHY 和 SRAM,采用单电源供电，可兼容 3. 3V.5V 的 IO 接口电平.DM9000A 同样支持 MII （Media IndependentInterface,介质无关接口），它包含一系列可被访问控制的状态寄存器，这些寄存器是字节对齐的，在硬件或者软件复位时被设置成初始化.

硬件框图如图 1 所示.

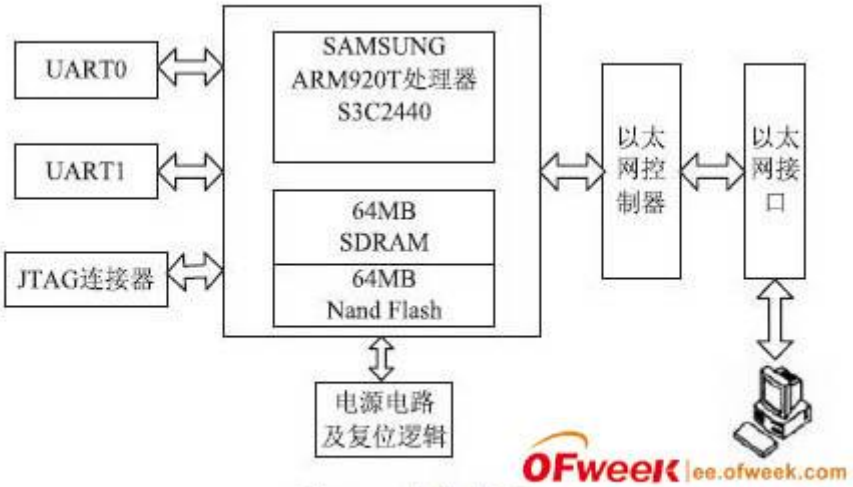

图 1　硬件框图

2 以太网软件的设计

2. 1 以太网卡控制器的初始化

首先 DM9000A 自检，读取 DM9000 的生产厂家 ID 和设备 ID 与已经设定好的 ID 进行比对，判断 DM9000 网卡是否存在，初始化 DM9000A,它的过程就是适当配置 DM9000A 寄存器的过程，具体过程分为以下几个步骤：

（1）启动 DM9000A,设置 CPCR[REG_1E]＝0×1,使 DM9000 的 GPIO3 为输出，GPR[REG_1F] =0×0,使 DM9000 的 GPIO3 输出为低以激活内部 PHY.延时 2ms 以上以等待 PHY 上电.

（2）进行两次软复位，设置 DM9000 为正常工作模式，根据芯片设计要求，要想使芯片在上电之后工作正常就要进行两次软复位，设置为 NCR[REG_00] =0×01,NCR[REG_00] =0×00,这两步操作进行两次.

（3） 清除各种状态标志位和中断标志位，NSR[REG_01] =0x2c,ISR[REG_FE] =0x3f.

（4）设置接收和发送控制寄存器，并且设置 FIFO 的大小，RCR[REG_05] =0×39.TCR[REG_02] =0×00.FCTR[REG_09] =0×38.

（5）设置板子自身的 MAC 地址.

（6）再一次清除各种状态标志位和中断标志位，NSR[REG_01] =0x2c,ISR[REG_FE] =0x3f.

（7）设置中断屏蔽寄存器，打开接收中断，IMR[REG_FF] =0×81.

当进行了以上步骤的设置之后，DM9000A 芯片就处于正常工作状态了.在以后进行通信的过程中，如果发生异常引起芯片重启，则再一次进行同样的设置.

2. 2 以太网卡数据的发送和接收

DM9000A 发送数据采用的是循环查询模式，接收数据采用的是中断模式，DM9000 内部有 0x3FF 大小的 SRAM 用于接收和发送数据缓存.在发送或接收数据包之前，数据是暂存在这个 SRAM 中的.当需要连续发送或接收数据时，需要分别把 DM9000 寄存器 MWCMD 或 MRCMD 赋予数据端口,这样就指定了 SRAM 中的某个地址，并且在传输完一个数据后，指针会指向 SRAM 中的下一个地址，从而完成了连续访问数据的目的.但当发送或接收一个数据后， 指向 SRAM 的数据指针不需要变化时，则要把 MWCMDX 或 MRCMDX 赋予数据端口 .

发送数据比较简单，接收数据就略显复杂，因为它是有一定格式要求的.在接收到的一包数据中的首字节如果为 0×01,则表示这是一个可以接收的数据包;如果为 0×0,则表示没有可接收的数据包.因此在读取其他字节时，一定要先判断首字节是否为 0×01.数据包的第二个字节为数据包的一些信息，它的高字节的格式与 DM9000 的寄存器 RSR 完全一致.第三个和第四个字节为数据包的长度.后面的数据就是真正要接收的数据了.

2. 2. 1 UDP 协议栈的裁剪实现

在系统中主要使用 UDP 通信，只需要实现 ARP 协议.IP 协议，对 TCP/ IP 协议进行部分的实现.UDP 协议通信（即用户数据报协议）与 TCP 一样都是属于传输层协议，位于 IP（网际协议）协议的顶层.UDP 相对于 TCP 是一种简单协议，提供的是最少的服务，编写的代码量也小，所需的程序和内存空间少，运行速度快.ARP 为 IP 地址对应的硬件地址之间提供动态映射，发送终端把以太网数据帧发送到位于同一局域网上的另一台主机时，是根据 48bit 的以太网地址来确定目的接口的.设备驱动程序从不检查 IP 数据报中的目的 IP 地址.IP 协议是 TCP/ IP 协议中最为核心的协议，它提供不可靠.无连接的数据报传送服务 .

2. 2. 2 数据的发送过程

数据发送过程如图 2 所示.发送终端在第一次发送数据的时候，要知道接收端的 IP 地址和端口号，还要得到对方的物理 MAC 地址，因为两个终端最后通信是通过寻找对方的 MAC 地址来进行的，因此首先得通过 ARP 协议，把对方的 IP 地址转换为 MAC 地址，得到了物理地址之后才能通信.如果长时间不能得到这个物理地址，则只能说明请求失败，需要重新发送 ARP 请求，ARP 的封装过程如图 3（b）所示.

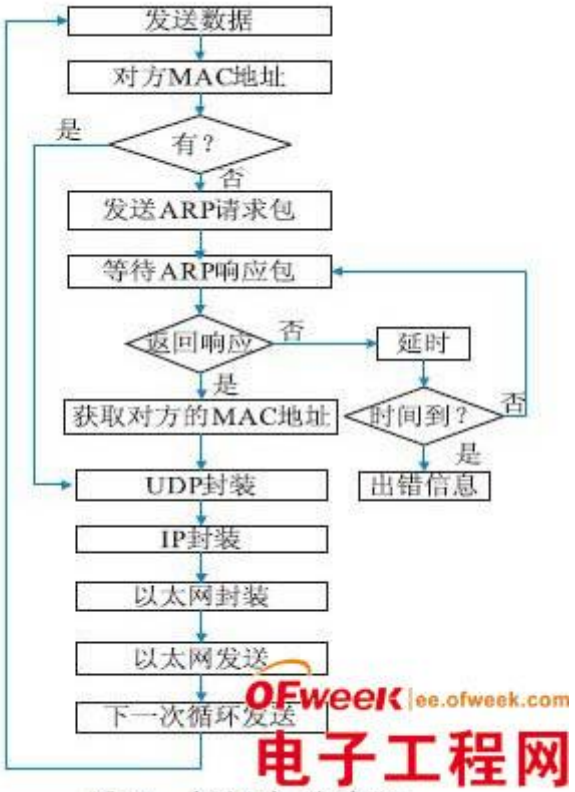

图 2　数据发送过程

图 3　数据封包过程

2. 2. 3 数据包的封装过程

UDP 协议数据包的封装在运输层进行，打好包的 UDP 数据将送往网络层进行 IP 协议的打包，UDP 要完成进程到进程的通信，把报文交付给正确的进程.当进程有报文要通过 UDP 发送时，它就把这个报文连同一对套接字地址以及数据长度传递给 UDP.UDP 收到数据后就加上 UDP 首部，也就是 UDP 数据包的封装如图 3（c）所示.然后 UDP 就把该用户数据包连同 IP 加上自己首部，在协议字段使用值 17,指出该数据是从 UDP 协议来的，这个过程就是 IP 数据包的封装过程如图 3（a）所示.这个 IP 数据包再传递给数据链路层.数据链路层收到 IP 数据包之后，加上自己的首部（可能还有尾部），再传递给物理层.物理层把这些位编码为电信号或者光信号，然后把它发送到远程的机器.

2. 2. 4 数据的接收

系统接收数据采用的是中断模式.当网卡接收到数据时，就触发一个中断，启动中断服务程序.在中断服务程序中首先清除中断标志位，以防在接收数据的时候再次引发中断，然后判断寄存器 MRMDX 的值，确定网卡是否接收到了数据，如果接收到了数据就要进行数据处理，也就是对数据包的解封，得到应用程序发送来的数据，如果没有得到数据则说明网卡初始化失败，重新初始化网卡.中断接收程序的流程图如图 4 所示.

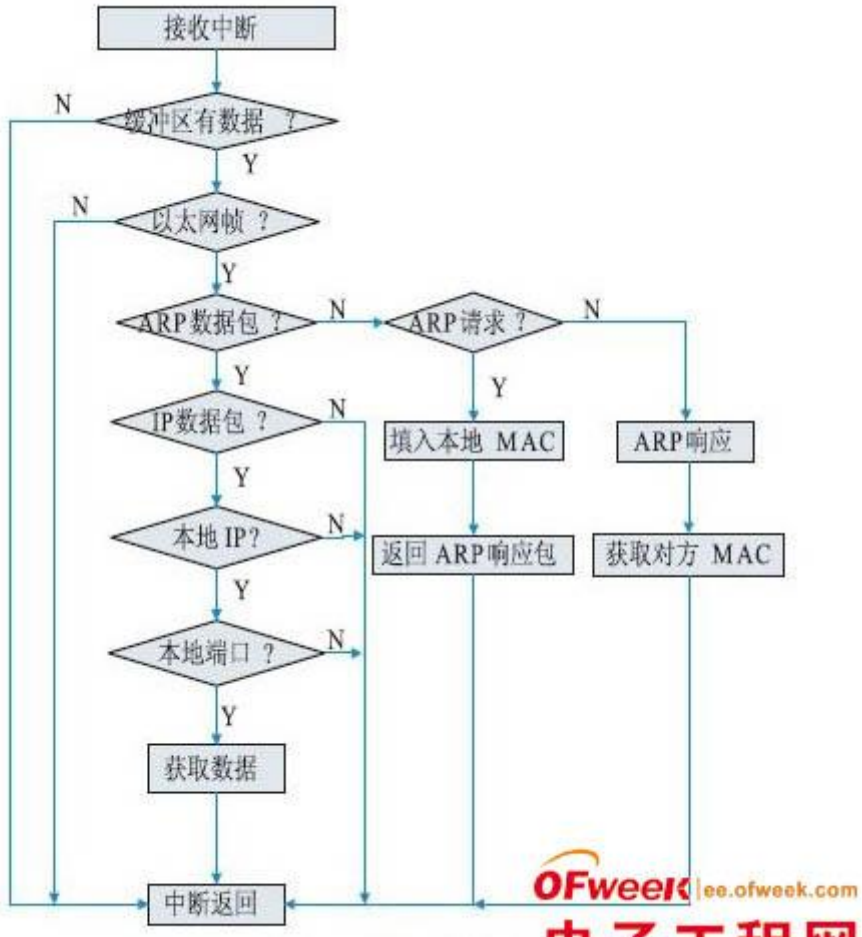

图 4　数据接收过程

在接收到以太网数据帧中，首先判断数据类型字段，如果是 ARP 协议，则进入 ARP 处理流程，如果是 IP 协议，则进入 IP 协议流程.ARP 协议处理过程：

首先判断 ARP 包目的 IP 地址是否与本地 IP 地址一致，如果不一致，丢弃不处理；如果一致，再判断 ARP 类型，操作类型字段为 1 时表示 ARP 请求，调用 ARP 发送函数发送 ARP 响应包.操作类型字段为 2 时，记录下对方的 MAC 地址，以后通信就是根据这个 MAC 传送数据的.

IP 协议处理过程如下所述：首先判断 IP 包目的 IP 地址是否与本地 IP 一致，如果不一致，丢弃不处理，如果一致，则再判断协议类型，是否为 UDP 数据包，是就进入 UDP 处理过程，不是就进入其他协议处理过程.

3 实验结果和分析

3. 1 ARP 通信测试

实验中测试了 ARP 请求和 UDP 通信，设置 ARM 开发板的 IP 地址为 219. 243. 50. 187,MAC 地址为 0×52,0×54,0x4c,0×38,0xf7,0×42,PC 机的 IP 地址为

219. 243. 50. 188,MAC 地址为 0×00,0×23,0x5A,0×15,0×73,0xF4.经过测试当开板向 PC 发送 ARP 请求时，PC 能够正确应答响应包，并且多次请求都能够得到正确的响应，证明 ARP 协议能够稳定运行，实验结果如图 5 所示.

图 5 ARP 通信实验结果

当开发板得到了 PC 机的 MAC 地址之后，就可以与它进行正常的 UDP 通信了.在中断程序之中判定协议类型，接收到的数据包是 ARP 协议还是 UDP 协议，当接收的是 ARP 协议，则进行 ARP 协议处理，得出图 5 的实验结果，当得到的是 UDP 协议，则进行 UDP 协议处理，得出的实验结果如图 6 所示.经过长时间的通信可以发现，传输的数据没有出现误码，可以说明这种通信方式比较稳定可行.

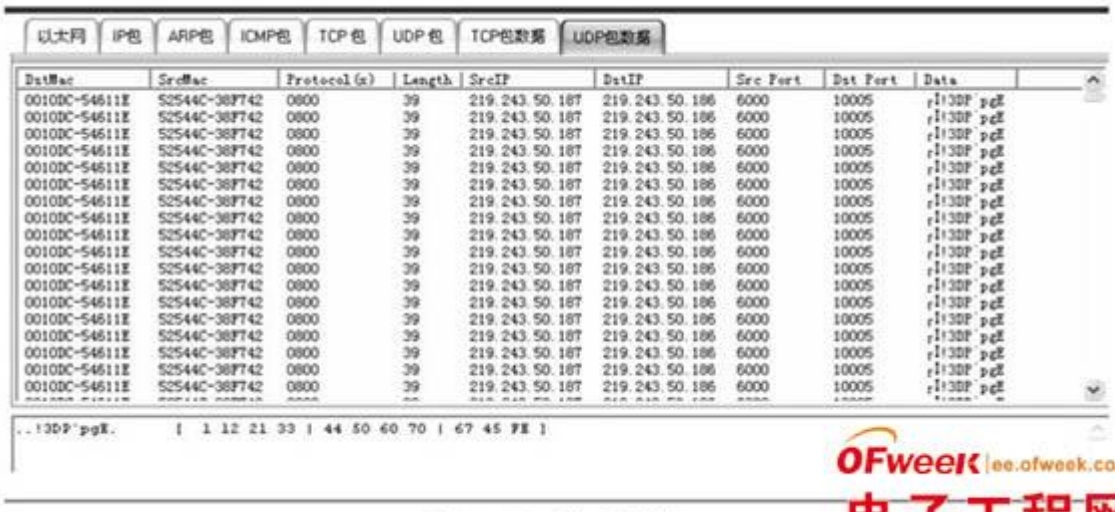

图 6 UDP 通信实验结果

3. 2 UDP 通信测试

在 UDP 通信实验中，设置两个数据终端的 IP 地址和通信端口分别为 219.243.50.187:6000,219.243.50.186:10005,然后发送数据，用抓包工具 sniffer 抓包的结果如图 6 所示.通过图中结果可以看出，SrcIP 为 219. 243. 50. 87,Src Port 为 6000,DestIP 为 219. 243.50. 186,Dst Port 为 10005,这都与设置的相同.Protocol 为 0×800 表示为 UDP 协议类型，而且能够正确地接收到发送的数据，并且经过多次的实验，结果都是正确的，这证明系统通信稳定可靠，通过移植的协议栈能够正常的工作，达到了预期的目标.

4 结束语

文中实现了基于 ARM9 和 DM9000 芯片的 UDP 通信，成功地对 TCP/ IP 协议栈裁剪移植实现 UDP.ARP 等协议通信.详细介绍了 DM9000 网卡驱动程序过程，并且实现了网口接收发送数据的功能，通过对大量数据的传输实验，证明了 ARM9 和 DM9000 构成的通信系统性能的稳定性.能够较好地解决大量数据通过 UDP 协议通信的问题.