基于 NDK 的 DSP 网络接口移植开发设计

张 枫,郑力新,周凯汀
(华侨大学 信息科学与工程学院,福建 厦门 361021)

摘 要:给出了一个基于 TMS320DM6437 DSP 的嵌入式网络实现方案,对该 DSP 的网络控制模块和 NDK 进行了深入的研究与分析,实现了 NDK 在不同外设的移植,并以 SEED-DEC6437 与 PC 之间网络通信为例,介绍了 PC 端 Winsock 与 DSP 的 NDK 开发流程。实验结果表明,使用移植过的 NDK 进行开发,可以有效地提高开发速度,减少开发时间。

关键词:嵌入式;网络接口;DSP;NDK

中图分类号:TP368 **文献标识码**:A **文章编号**:1674-7720(2013)01-0045-03

Transplant of DSP's network interface based on the NDK

Zhang Feng, Zheng Lixin, Zhou Kaiting
(College of Information Science and Engineering, Huaqiao University, Xiamen, 361021, China)

Abstract: This article provides an embedded network implementation based on TMS320DM6437 DSP. It also includes further research and analysis on the network control module and NDK of the DSP in order to realize the NDK in different peripherals transplantation. Then, taking the network communication between SEED-DEC6437 and PC as an example, this paper introduces the development processes on PC Winsock and DSP NDK. Experimental results show that it can improve the develop speed and reduce the develop time effectively by using the transplanted NDK to develop.

Key words: embedded; network interface; DSP; NDK

随着网络通信技术的不断发展,嵌入式设备间的网络通信起到了越来越重要的作用,TI 公司推出的嵌入式数字信号处理器(DSP)都集成了以太网介质存取控制器(EMAC),使得因特网终端的连接成本降低了 50%以上。TI 同时推出了相对应的开发工具包(NDK),使用 NDK 不仅能够快速地开发网络程序,缩短开发周期,而且应用程序能够方便地在不同型号 DSP 上进行移植。本文主要针对 NDK 的结构进行详细介绍,阐述了移植的详细过程,最后进行实验,实验结果证明,移植过的 NDK 可以进行良好的运用。

1 DM6437 的网络模块

TMS320DM6437 是 TI 公司的一款专用于数字媒体应用的高性能 32 bit 定点 DSP 处理器,集成了以太网媒质访问控制器(EMAC)和物理层设备的数据输入输出管理(MDIO)模块用于网络配置[1]。其网络功能模块如图 1 所示。

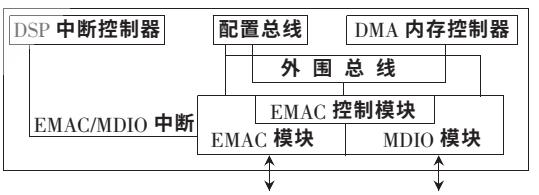

图 1 DSP 网络功能模块

DM6437 的网络功能主要由 EMAC 控制模块、EMAC 模块和 MDIO 模块 3 部分组成[2]。

EMAC 控制模块提供了 DSP 核与 EMAC 模块和 MDIO 模块之间的接口,它的作用主要是控制中断和有效地利用设备内存。EMAC 控制模块内部具有 8 KB 的随机存储器用来保存信息包的缓冲描述符。

MDIO 模块采用串行接口控制器来对以太网的物理层进行监视和控制[3],最多支持 32 个物理层设备。它主要负责管理与 EMAC 相连的所有 PHY 芯片,包括对 PHY 芯片进行状态检测、配置等操作。使用 MDIO 可以减少额外的 CPU 开销。

EMAC 模块提供了 DSP 核与网络之间通信的高效接

口，DM6437 的 EMAC 模块支持 10 Mb/s 和 100 Mb/s。在半双工或全双工模式下，同时具有硬件流控制及服务质量保证（QoS）支持。

内置的 EMAC/MDIO 仅仅需要连接一个物理层设备即可，大大减少了开发时间，因此成为高速嵌入式网络连接一个很好的选择。在 PHY 的接口设计中 SEED-DEC6437 使用的是 DAVICOM 公司的 DM9161A 作为 10/100Base-TX 以太网收发器[4]，DM9161A 提供有 MII 接口，可以实现与 TMS320DM6437 的 MII 接口无缝对接。RJ45 连接器选用 AMP 公司的 406549-1，其上带两个 LED 指示灯，右边的 LED 为绿色，用作指示连接状态；左边的为黄色，正常情况下，用来指示数据传输。接口原理图[5]如图 2 所示。

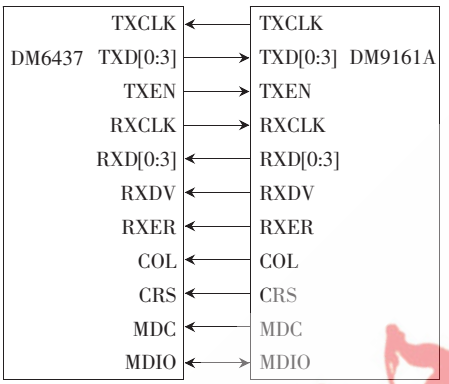

图 2 DM6437 网络接口连接图

2 NDK 结构的介绍与移植

为了加速 C6000 系列 DSP 的网络开发进程，TI 公司结合 C6000 系列芯片推出 TCP/IP 的 NDK 开发套件，目的是为了通过使用较少资源消耗来支持 TCP/IP 服务，例如应用层 Telnet、DHCP、HTTP 服务等。由于 NDK 中内置了常用的服务程序，因此通过使用 NDK 不仅可以有效地减少服务占用的资源，而且可以减少开发时间，以便于更快地推出产品。

NDK 建立在 TI 的嵌入式操作系统 DSP/BIOS 之上，主要由 TCP/IP 网络协议栈（STACK.LIB）、网络工具库（NETTOOL.LIB）、操作系统层和打印层（OS.LIB and MiniPrintf.LIB）、硬件接口层（HAL.LIB）以及网络控制层（NETCTRL.LIB）5 部分重要的库构成，如图 3 所示。

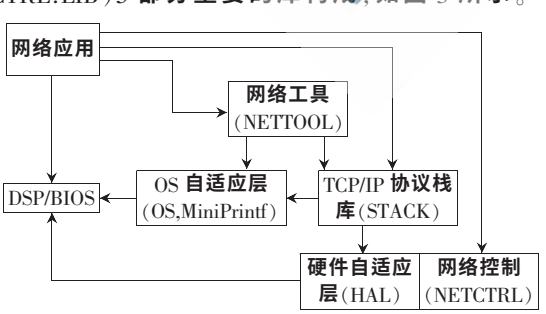

图 3 协议栈结构示意图

协议栈（STACK.LIB）指的是主要的 TCP/IP 网络栈，这个库文件主要是跟 DSP/BIOS 系统有关，移植的时候不需要对它进行改变。

网络工具库（NETTOOL.LIB）包含 NDK 提供的所有网络服务的套接字和一些帮助用户开发的工具。

操作系统层和打印层（OS.LIB and MiniPrintf.LIB）提供了一些抽象的函数以供 DSP/BIOS 调用，例如任务线程管理、内存分配和包缓冲管理等。而打印层提供了 CCS 使用的 RTS 输出函数等。

网络控制层（NETCTRL.LIB）是协议栈的中心，它控制 TCP/IP 与外界交互，主要作用是初始化 NDK 以及底层设备，启动网络系统配置，响应和调度设备，退出时卸载系统配置清除驱动程序。

硬件接口层（HAL.LIB）是连接硬件设备与 NDK 之间的接口，包含时钟、LED 指示灯、以太网设备及串口的驱动。需要对这个文件进行修改用来适应不同的外设。由于 HAL.LIB 是个静态链接库，需要找到开发包中提供的源程序文件，修改之后重新进行编译。源文件位于%NDK_INSTALL_DIR%\packages\ti\ndk\src\hal\<board_name>\eth_<device_name>中。以 NDK1.92 版本为例，在 ndk_1_92_00_22_eval\packages\ti\ndk\src\hal\evmdm6437 中找到 eth_dm6437 和 userled_dm6437 文件夹，其中 eth_dm6437 中是以太网设备驱动程序，而 userled_dm6437 中是 LED 指示灯的驱动程序，LED 的驱动程序采用默认的即可。NDK 的以太网驱动程序所包含的文件如表 1 所示。

表 1 NDK 以太网驱动程序所含文件

	文件名	文件内容
<SRC\HAL\EVMDM6437\ETH_DM6437>	LLPACKET.C	与硬件无关的以太网底层包驱动程序
	LLPACKET.H	LLPACKET.C 的头文件
	DM64LC.C	DM6437 的 EMAC 数据包微型驱动
	DM64LC_MDIO.C	DM6437 的 EMAC 的 MDIO 控制函数
	DM64LC_MDIO.H	MDIO 的头文件
<SRC\HAL\EVMDM6437\ETH_DM6437\CSLR>	CSLR.H	定义了一些主要的寄存器
	DM64LC_COMMON.H	通用的外围寄存器的结构以及定义
	CSLR_ECTL.H	ECTL 的寄存器描述
	CSLR_EMAC.H	EMAC 的寄存器描述
	CSLR_MDIO.H	MDIO 的寄存器描述

LLPACKET.C 是与硬件设备无关的以太网包驱动，不需要进行修改。DM64LC_MDIO.C 是与硬件设备有关的，包括 PHY 设备的初始化、配置等，需要进行修改。文件首先定义了相应的 PHY 控制寄存器地址，经过与 DM9161A 手册对比，地址符合，不需要进行更改。在接下来的初始化 PHY 设备 MDIO_initPHY 的函数中，应该进行适当的修改以符合不同的 PHY 设备。TI 为了加快 PHY 的初始化速度，当 PHY 设备启动时，采用宏定义

PHY_MASK 来屏蔽不使用的 PHY 位。通过查看 DM9161A 手册可以看到，管脚 29、28、27、26、35 分别代表的是 PHYAD[0]~PHYAD[4]，通过查看原理图上引脚是否有上拉电阻即可确定其值，一般通过 10 kΩ 的上拉，设置为 1，不上拉设置为 0。经过查看 PHYAD=01111，换成十进制为 15，则 PHY_MASK 就为 0x8000。在文件开始位置修改 PHY_MASK 即可。如果需要连接多个 PHY 设备，在 MDIO_initPHY()函数中，需要把注释“Shutdown all other PHYs”下面的代码删除。这段代码是为加速单个 PHY 设备启动而设计的，当连接多个 PHY 设备时就会阻止其正常工作，经过修改之后需要重新编译生成 lib 文件，注意要添加编译包含的路径，如图 4 所示。

图 4 编译包含路径

将生成的文件复制到 NDK 中的 lib\hal\evmdm6437 目录下，名称为 hal_eth_dm64lc.lib 即可。

3 NDK 的开发与测试

在使用 NDK 开发之前，需要设置 NDK 的开发环境，需要进行以下操作[6]：

（1）硬件层需要一个 100 ms 的周期函数来作为它的时钟驱动，每 100 ms 周期驱动一次 llTimerTick()函数。所以在 BIOS 里面 PRD 下面新建一个 prdNDK 变量，并在其属性中设置周期为 100 ms，函数为 _llTimerTick。

（2）系统需要钩子函数为 TCP/IP 堆栈加载和保存私人变量指针，所以在 BIOS 中的 HOOK 下面新建一个 hookNdk，并将初始化函数设置为 _NDK_hookInit，创建函数设置为 _NDK_hookCreat。

（3）需要包含一些文件头文件。必须把工程项目的 IncludeSearchingPath 指向 NDK 安装目录下的 inc 目录，如%NDK_INSTALL_DIR%\packages\ti\ndk\inc。

（4）为了确保能够正确编译项目文件，需要在 CCS 项目中的“build options”的“Link Order”一栏按一定顺序添加库文件，最佳的顺序是：NETCTRL.LIB、HAL_xxx.LIB、NETTOOL.LIB、STACK.LIB、OS.LIB 以及 MiniPrintf.LIB。

（5）NDK 中的 OS 和 HAL 会创建 3 个内存段，分别是 PACKETMEM、MMBUFFER 和 OBJMEM，必须为这 3 个段分配内存，在 CMD 中写入以下内容：

```
SECTIONS
{
.far:PACKETMEM:{}>MYSDRAM
.far:MMBUFFER:{}>MYSDRAM
.far:OBJMEM:{}>MYSDRAM
}
```

这里将 3 个段分配到自定义的 MYSDRAM 中。

（6）如果 cache L2 完全设置为 SRAM，HAL 驱动将不能正常工作，这里需要把 Cache 设置至少为 32 KB 大小。Cache 的大小对网络传输速度的有很大的影响，Cache 越大速度越快。

设置好开发环境之后，在 DSP 端开发一个 UDP 程序来与上位机通信。具体实现的功能是：上位机发送一个字符串，DSP 接收后进行回传，在上位机进行显示。程序的基本流程如图 5 所示。

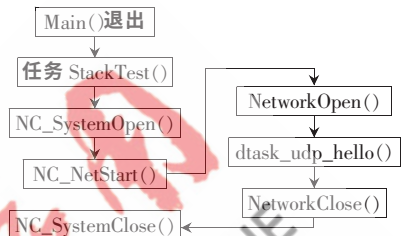

图 5 程序流程图

dtask_udp_hello()中代码如下：

```
int dtask_udp_hello(SOCKET s, UINT32 unused)
{
    struct sockaddr_in sin1;
    int                i, tmp;
    char               *pBuf;
    HANDLE             hBuffer;
    for(;;)
    {
tmp=sizeof(sin1);
i =(int)recvncfrom (s, (void**)&pBuf, 0, &sin1, &tmp, &hBuffer);
    //将数据重新回送
    if(i>=0)
        {
            sendto(s, pBuf, i, 0, &sin1, sizeof(sin1));
            recvncfree(hBuffer);
        }
        else
            break;
    }
```

程序运行后的结果在 DSP 端 stdout 显示，如图 6 所示，上位机显示如图 7 所示。

图 6 DSP 端显示结果

图 7 上位机显示结果

为了研究 NDK 开发工具的移植方法，本文讨论了 DM6437 上 NDK 的结构以及工作原理，并进行了与 PC 数据通信的测试。测试表明经过移植的 NDK 可以正确地开发网络程序。本文重点介绍了 NDK 的移植方法，使开发者可以将其使用在不同的 PHY 设备上，这大大拓展了 NDK 的使用范围。TI 推出的 NDK 网络开发工具可以使技术人员快速开发基于 DSP 的网络应用程序，而且具有十分可靠的性能，使用本文的方法使得基于 NDK 开发的程序可以快速地移植到不同 PHY 设备的 DSP 上。随着 DSP 网络通信的不断普及，该方法的应用范围将越来越广阔。希望通过本文的介绍，使得相关的开发研究人员得到方便，减少开发时间。

参考文献

[1] Texas Instruments. TMS320C6000 network developer's kit support package for EVMDM6437 user's guide[Z]. 2006.

[2] Texas Instruments. TMS320C6000 network developer's kit (NDK) software user's guide[Z]. 2007.

[3] Texas Instruments. TMS320DM643x DMP ethernet aedia access controller (EMAC)/management data input/output (MDIO)module user's guide[Z]. 2007.

[4] 合众达.SEED-DEC6437 用户指南[Z].2008.

[5] 合众达.SEED-DEC6437 V1.1 原理图[Z].2008.

[6] Texas Instruments. TMS320C6000 network developer's kit (NDK) software programmer's reference guide[Z]. 2007.

（收稿日期：2012-05-11）

作者简介：

张枫，男，1987 年生，硕士研究生，主要研究方向：智能相机图像处理。

郑力新，男，1967 年生，博士，教授，主要研究方向：工业自动化技术和人工智能。

周凯汀，女，1969 年生，副教授，主要研究方向：信号处理、模式识别、智能控制、电子测量与仪器领域。