# NET-SNMP 网管代理在 ARM9 上实现的技术①

刘 明，陈 磊，宋 佳

(北京邮电大学 信息与通信工程学院，北京 100876)

**摘 要**：简单网络管理协议简化了复杂的网络管理工作，使得网络管理可以标准化。数字电视的综合网管基于简单网络管理协议，通过在数字电视前端设备上建立网管代理，把数字电视管理起来，给用户管理大量的数字电视带来了极大的便利。考虑到 NET-SNMP 是一个开源的软件包，选择 NET-SNMP 开发嵌入式网管系统既便于移植，也利于代理的扩展。使用 NET-SNMP 在数字电视前端设备的嵌入式 ARM9 平台上成功建立了网管代理，实现了网管代理的高效运行，为监控数字电视的工作情况打下了坚实的基础。

**关键词**：嵌入式系统；SNMP；NET-SNMP；网管代理；ARM9

## Implementing NET-SNMP Agent in ARM9

LIU Ming, CHEN Lei, SONG Jia

(School of Information and Communication Engineering, Beijing University of Posts and Telecommunications, Beijing 100876, China)

**Abstract**: SNMP is convenient for complicated network management, which is able to standardize network management. The network management of digital television is based on SNMP. The agent is built on the ARM9 platform of digital television front-end equipment. All the digital televisions are managed by the same platform, which is easy for user management. NET-SNMP is an open source software package. For the sake of code migration and agent extension, NET-SNMP is selected to research and develop the embedded network management system. This article chose NET-SNMP to build agent on ARM9 and realized the running of agent, which laid a solid foundation for monitoring the running of digital television.

**Key words**: embedded system; SNMP; NET-SNMP; agent; ARM9

当前，数字化通信正席卷全球，模拟电视在经历了几十年的发展以后也走到了尽头。随着我国三网融合的推进，数字电视正经历着前所未有的黄金发展阶段。因为 IP 是通信的主流发展方向，所以 IPTV 在数字电视的市场上占据着绝对优势。以 SNMP 为基础的数字电视集中网管系统能够给用户管理数字电视的相关设备带来极大的方便，简单的设置就能完成用户所要求达到的效果。

数字电视综合网管的实现包括在后台的嵌入式操作系统中建立网管代理和在前台界面监控网络的运行情况等。本研究课题主要解决的是在后台的嵌入式操作系统中建立网管代理。只有在所有的数字电视前端设备上都成功建立网管代理，才能在统一的前台界面监控整个数字电视网络的运行状况。当然，只要能在一套数字电视前端设备上成功安装网管代理，也就能够在所有的数字电视设备上都安装网管代理。

NET-SNMP 网管代理在数字电视集中网管系统中的成功应用对相关平台的开发具有借鉴和应用意义，对推动数字电视的广泛传播起着良好的推动作用。

## 1 简单网络管理协议简介

SNMP 是专门设计并用于在 IP 网络上管理网络节点（服务器、工作站、路由器、交换机及集线器等）的一种标准协议，它是一种应用层协议。SNMP 使网络管理员能够监控网络性能，发现并解决网络问题以及规划网络扩容。通过 SNMP 的陷阱机制，网络管理系统获知网络出现的问题。

从体系结构上来说，SNMP 框架由主代理、子代

① 收稿时间:2011-02-24;收到修改稿时间:2011-04-09

理和管理站组成。

### 1.1 主代理

主代理是一个在可执行 SNMP 的网络设备上运作的软件，可回应从管理站发出的 SNMP 请求。它的角色类似客户机/服务器（client/server）体系结构中的服务器。主代理依赖子代理提供特定功能的管理咨询。

如果系统当前拥有多个可管理的子系统，主代理就会传递它从一个或多个子代理处收到的请求。这些子代理在一个子系统以及对那个子系统进行监测和管理操作的界面为前台的设备上建模。主代理和子代理的角色可以合并在一起，在这种情况下我们可以简单的称之为代理（agent）。

### 1.2 子代理

子代理是一个在可执行 SNMP 的网络设备上运行的软件，执行在特定子系统的特定管理信息库（MIB）中定义的请求和管理功能。子代理的主要功能有：

① 收集主代理的咨询；

② 配置主代理的参数；

③ 回应管理者的请求；

④ 产生警告或陷阱。

对协定和管理咨询结构的良好分离使得使用 SNMP 来监测和管理同一网络内上百个不同子系统非常简单。MIB 模型管理 OSI 参考模型的所有层，并可以扩展至诸如资料库、电子邮件以及 J2EE 参考模型之类的应用之中。

### 1.3 管理站

管理者或者管理站提供第三个元素。它和一个客户机/服务器体系结构下的客户端一样工作。它根据一个管理员或应用程序的行为发出管理操作的请求，也接收从代理处获得的 TRAP。

在典型的 SNMP 用法中，有许多设备被管理，而且是有一个或多个管理者在管理这些设备。每一个被管理的设备上运行一个代理（agent），通过发送陷阱对管理者报告设备信息。

SNMP 代理上维护着管理信息库（MIB）。管理者通过 GET，GETNEXT 和 GETBULK 等操作获得管理信息，或是代理在没有被询问的情况下，使用 TRAP 或 INFORM 发送管理信息。管理者也可以设置设备的管理信息，通过 SET 操作达到主动管理系统的目的。SET 操作只有当网络基本结构需要改变的时候使用，而监控操作则是常态性的工作。

目前，SNMP 有 3 种版本： SNMPv1、SNMPv2 和 SNMPv3。第 1 版和第 2 版没有太大区别，但 SNMPv2 是增强版本，包含了各种协议操作。与前两种相比，SNMPv3 则包含更多安全和远程配置。为了解决不同 SNMP 版本间的不兼容问题， RFC3584 中定义了三者共存的策略。

## 2 NET-SNMP软件包介绍

NET-SNMP 是一个开源的 SNMP 实现项目。本研究课题所用的 NET-SNMP 版本是 5.4.2.1。NET-SNMP 支持 SNMPv1、SNMPv2 和 SNMPv3，支持基于 IPv4 和 IPv6 的 SNMP 应用程序开发。

NET-SNMP 包括以下内容：

① NET-SNMP 提供完整的 API 用于 SNMP 应用程序开发，包括 C 和 Perl 的 API；

② 一个可扩展的、功能强大的 SNMP 代理：snmpd，开发者可以开发动态模块扩展 snmpd，NET-SNMP 内置扩展子代理与主代理的通信协议；

③ 提供众多命令行工具检查和使用 SNMP 协议；

④ 一个图形化的 MIB 浏览工具 tkmib；

⑤ 一个 Trap 接收进程，用于接收和显示 Trap，并可以将 Trap 记录到日志文件里。

NET-SNMP 被很多商业化 Linux 包含，大多数的 Linux 使用 NET-SNMP 的主代理实现 Linux 的 SNMP 支持[1]。

## 3 搭建硬件平台

准备一台 PC，安装 linux 操作系统，内核的版本在 2.6 以上；再准备一块目标板，目标板上的核心控制器是 AT91RM9200[2]，使用 Flash 作为嵌入式系统的存储器，在 Flash 上安装 Linux 操作系统，搭建的硬件平台如下图 1 所示。

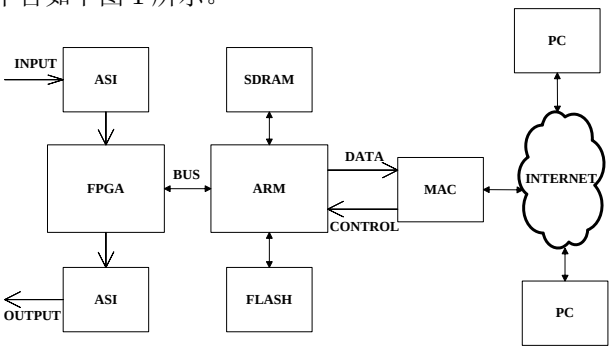


图 1 网管代理硬件平台结构图

## 4 源码测试

源码测试的主要目的是验证NET-SNMP源码和硬件平台的适用性。

首先，打开命令行终端界面，进入源码的目录/root/net-snmp-5.4.2.1，运行命令./configure --prefix=/usr/local/net-snmp --with-perl-modules，完成配置工作。

各选项的说明如下：

--prefix=/usr/local/net-snmp：指明编译后的net-snmp代理端软件在PC上的安装路径；

--with-perl-modules：指明获得perl模块的支持。

接下来，输入命令make，完成编译工作。

最后，输入命令make install，完成安装工作。

还有一步非常重要的工作，即添加环境变量。由于使用的不是默认安装，所以需要添加寻找配置文件的环境变量SNMPCONFPATH：

在目录/etc下找到环境变量的配置文件profile，添加如下三行：

SNMPCONFPATH="$PATH:/usr/local/etc/snmp"
PATH="$PATH:/usr/local/arm/3.4.1/bin:/usr/local/net-snmp/sbin:/usr/local/net-snmp/bin"

export PATH SNMPCONFPATH

安装完成后，打开命令行终端界面，运行snmpd，此时可以通过查看日志文件检验代理是否正常运行，日志文件位于目录/var/log下，打开此目录下的日志文件snmpd.log，查看内容：

Turning on agentX master support.

NET-SNMP version 5.4.2.1

如果日志文件snmpd.log显示以上两行信息，说明网管代理成功运行，NET-SNMP软件包测试完毕。

## 5 源码移植

以root的身份登录Linux操作系统。

从NET-SNMP的官方网站www.net-snmp.org下载源码压缩包net-snmp-5.4.2.1.tar.gz，将此源码压缩包放置在目录/root下，打开命令行终端界面，直接输入tar -xzvf net-snmp-5.4.2.1.tar.gz，解压后的源码net-snmp-5.4.2.1位于当前目录下。

首先，在命令行终端界面输入arm-linux-gcc –v，查看是否安装交叉编译器及其版本。如果没有安装交叉编译器，建议把交叉编译器Arm-linux-gcc 3.4.1安装在/usr/local/arm/3.4.1/目录下。

其次，添加环境变量，在/etc目录下的profile文件中找到包含export PATH字样的行，在该行前面加上PATH="$PATH:/usr/local/arm/3.4.1/bin",接着重新启动操作系统。

回到源码的主目录/root/net-snmp-5.4.2.1下输入下列命令：

CC=arm-linux-gccAR=arm-linux-ar
LD=arm-linux-ld./configure--with-cc=$CC
--with-ar=$AR--prefix=/usr/local/net-snmp-arm
--build=i686-linux--host=arm-linux
--with-endianness=little--disable-applications
--disable-manuals--disable-debugging
--disable-snmptrapd-subagent--disable-ipv6
--disable-scripts--enable-mini-agent
--with-mib-modules="notification"
--with-mib-modules="agentx"
--disable-ucd-snmp-compatibility
--disable-embedded-perl--disable-perl-cc-checks
--without-perl-modules

全部输入完毕后再按回车键，开始对源码进行交叉编译的配置。

交叉编译时加入了很多的编译选项，以使得代理端软件尽可能的精简。编译各选项的说明如下：

--with-cc=$CC：指明编译时使用的编译器；

--with-ar=$AR：指明编译库的工具；

--prefix=/usr/local/net-snmp-arm：指定交叉编译后的net-snmp代理端软件在PC上的安装路径；

--build=i686-linux：指明编译源码的主机；

--host=arm-linux：指明要运行该程序的机器；

--with-endianness=little：指明采用小端模式；

--disable-applications：指明不编译SNMP的应用程序；

--disable-manuals：指明取消编译帮助信息；

--disable-debugging：关闭调试信息；

--disable-snmptrapd-subagent：指明不支持snmptrapd子代理；

--disable-ipv6：指明不支持ipv6协议；

--disable-scripts：指明不安装脚本；

--enable-mini-agent：指明安装最简单的代理；

--with-mib-modules="notification"：指明加载notification模块，此模块用于发送trap；

--with-mib-modules="agentx"：指明加载 agentx 模块，此模块支持主代理和子代理，默认情况下启动的是子代理，但是在配置文件 snmpd.conf 可以配置为主代理；

--disable-ucd-snmp-compatibility：指明不安装 UCD-SNMP 的头文件和库；

--disable-embedded-perl：指明不支持嵌入式 perl；

--disable-perl-cc-checks：在支持嵌入式 Perl 的情况下，不对编译 Perl 的 C 编译器进行兼容性检查；

--without-perl-modules：指明不支持 Perl 模块。

需要注意的是，这里选项输入的顺序不能随便更改，否则可能无法完成代理的移植。

配置完成后，需要编译程序，输入命令 make，以默认的动态共享库的方式编译。采用动态共享库的方式编译时，需要动态共享库的支持才能正常运行网管代理。

安装编译好的程序，输入命令 make install。

至此，用于移植的源码就完全编译好了，并且编译好的软件就安装在目录/usr/local/net-snmp-arm 下，此目录下各文件的说明如下：

bin:net-snmp 配置脚本文件；

include:网管代理用到的头文件；

lib:共享库文件；

sbin:代理程序 snmpd；

share:mib 文件。

接下来，打开命令行终端界面，输入命令 cd /usr/local/net-snmp-arm/sbin，进入到相应的目录，再输入命令 arm-linux-strip snmpd，把代理 smpd 进行压缩，减少其在目标板上占用的存储空间。

同时，输入命令 cd /usr/local/net-snmp-arm/lib，进入到相应的目录，输入命令 arm-linux-strip libnetsnmp.so.15.1.2，压缩库文件 libnetsnmp.so.15.1.2，按同样的方式压缩其它的库文件 libnetsnmpmibs.so.15.1.2，libnetsnmpagent.so.15.1.2 和 libnetsnmphelpers.so.15.1.2。

库文件压缩好以后，准备配置文件 snmpd.conf 如下：

```
createUser liuming MD5 12345678 DES 87654321
rwuser liuming auth
com2sec local default public
com2sec mynetwork default private
group MyRWGroup  v1 local
group  MyRWGroup v2c local
group  MyRWGroup usm local
group  MyROGroup v1 mynetwork
group  MyROGroup v2c mynetwork
group  MyROGroup usm mynetwork
view all included. 180
access MyROGroup""any noauth exact all none none
access MyRWGroup""any noauth exact all all none
master on
trapsink 192.168.0.19:162
trap2sink 192.168.0.19:162
informsink 192.168.0.19:162
trapsess 192.168.0.19
```

现在开始把源码移植到 arm9 目标板上。首先，将刚才生成的文件 snmpd 和 snmpd.conf 通过 FTP 服务器上传到 ARM9 上，放在目录/mnt/mtd 下。

打开命令行终端界面，输入 kermit –c，进入 arm9 的 linux 操作系统，输入命令 cd /mnt/mtd，进入相应的目录。输入 ls，确认已经成功安装文件 snmpd 和 snmpd.conf。输入命令 mkdir mibs，创建文件夹 mibs,用于存放 mib 文件。如果成功创建，把 PC 上 /usr/local/net-snmp-arm/share/snmp/mibs 目录下的 mib 文件 SNMPv2-SMI.txt、SNMPv2-MIB.txt 和 RFC1213- MIB. txt 移植到 arm9 的 /mnt/mtd/mibs 目录下。如果成功完成，再次在/mnt/mtd 目录下创建文件夹 lib，把压缩好的共享库文件 libnetsnmp.so.15.1.2、libnetsnmpmibs.so.15.1.2、libnetsnmpagent.so.15.1.2 和 libnetsnmphelpers.so.15.1.2 移植过来。

接下来的工作是建立符号链接，打开命令行终端界面，输入 kermit –c，进入 arm9 的 linux 操作系统，输入命令 cd /mnt/mtd，进入相应的目录。输入命令 ln –s /mnt/mtd/lib/libnetsnmp.so.15.1.2 /lib/libnetsnmp.so.15, 建立 /mnt/mtd/lib 下的库文件 libnetsnmp.so.15.1.2 和/lib 下的库文件 libnetsnmp.so.15 的符号链接。同理，完成其它库文件的符号链接。至此，全部的移植工作完成。

## 6 系统测试

打开命令行终端界面，输入 kermit –c，进入 arm9 的 linux 操作系统，输入命令 cd /mnt/mtd，进入相应的目录。

输入命令./ snmpd-Lfsnmpd.log-c /mnt/mtd/ snmpd.

（下转第 149 页）

位，实现数据模式到自动报站模式的转变。其软件实现流程如图 6 所示：

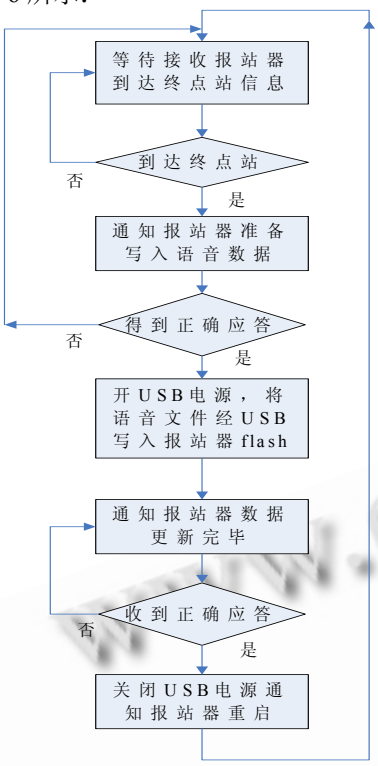


图 6 模块与报站器通信流程图

## 3 结语

本文中的智能公交无线传输模块已在小范围安装使用。实际情况表明，该模块具有良好的可靠性与稳定性，通过该模块能有效地将已有公交报站器与 POS 机纳入到整个长沙市智能公交网络，有效提高对车载电子设备的管理与维护，该模块作为长沙市智能公交网络的核心车载电子设备，在公交网络中起到一个信息枢纽的作用，具有良好的应用前景与经济价值。随着运营商 3G 网络覆盖的日益完善，以及 3G 资费的逐步下调，未来该模块可考虑升级 WCDMA 作为组网技术。

## 参考文献

1 文沛.应用智能调度系统改变城市公交传统管理.城市公共交通,2007,(6):13–14.
2 潘良,刘宏立.GPRS 技术在智能公交管理系统中的应用与研究.计算机工程与科学,2009,(11):153–155.
3 温泉,李炳煜,焦毅.基于 GPRS 的无线数据传输系统解决方案.现代电子技术,2006,29(23):15–20.
4 华清远见嵌入式培训中心.嵌入式应用程序开发.北京:人民邮电出版社,2009.

（上接第 235 页）

conf-M /mnt/mtd/mibs，启动 arm9 上的网管代理。配置选项说明如下：

-Lf snmpd.log：指明在当前目录下生成日志文件 snmpd.log；

-c /mnt/mtd/snmpd.conf：指明使用的配置文件；

-M /mnt/mtd/mibs：指明使用的 mib 文件；

为了检查代理是否在 arm9 上正常运行，需要查看日志文件。此时，通过 FTP 把日志文件 snmpd.log 传输到 PC 上，打开日志文件。

如果在配置文件 snmpd.conf 中指明了代理以主代理的方式运行，则日志文件的内容为：

Turning on agentX master support.

NET-SNMP version 5.4.2.1

如果没有在配置 snmpd.conf 中指明代理以主代理的方式运行，则代理以默认的子代理方式运行，此时日志文件的内容为：

NET-SNMP version 5.4.2.1

如果没有出现以上情况，则网管代理很有可能没有正常运行。这时候，需要重新移植，再次启动网管代理，直到正常启动网管代理为止。

## 7 结论

本文把 NET-SNMP 网管代理移植到数字电视设备上，成功地实现了网管代理在嵌入式系统上的运行，这样就建立了数字电视综合网管系统的后台部分。需要特别注意的是，交叉编译时的配置必须正确，各个配置选项的顺序不能随便打乱，否则，编译后的共享库文件可能出现问题。

## 参考文献

1 李明江.SNMP 简单网络管理协议.北京:电子工业出版社,2007.3–14.
2 李驹光.ARM 应用系统开发详解.基于 S3C4510B 的系统设计.第 2 版.北京:清华大学出版社,2004.40–126.