应用报告
ZHCA542A–December 2006–Revised May 2008

构建小型嵌入式 Linux 内核示例

Loc Truong and Brijesh Singh

摘要

这份应用说明演示了如何使用 DaVinci DM644x 数字评估模块 (DVEVM) 包进行NOR内核构建和电路板设置。 目标是构建尽可能小的内核，此构建使用 MontaVista® Linux 支持软件包 (LSP)，此软件包支持 HTTP 服务器，TCP/IP 堆栈和以太网以及针对串行调试终端的通用异步收发器 (UART) 所需的驱动程序。 内核驻留在 NOR 闪存内，并且使用一个 RAM 基于磁盘的文件系统，此系统也保存在闪存内。

诸如路由器和打印服务器等嵌入式器件中的这种设置可用作更加复杂器件的开始点，例如 I/O 监视器、网络摄像头和多媒体播放器。

此应用说明包括以下部分：

- 可用所需硬件和软件的概述
- 构建内核
- 构建 RAM 磁盘文件系统
- 设置应用
- 存储到闪存

内容

图片列表

图表列表

SPRAAH2 — http://www-s.ti.com/sc/techlit/SPRAAH2

1 概述

为了创建一个独立且可引导的嵌入式 Linux 系统，三个主要的软件段必须被存入 EVM：

- 在这个情况下为一个引导加载程序，u-Boot
- 一个用于 DaVinci DM644x 器件的 Linux 内核（具有内置驱动程序）
- 一个包含了 shell 程序、应用和运行支持工具和堆栈的 ARM 目标 Linux 文件系统

这一部分快速检查了 DVEVM，其中包括所需的硬件和软件组件。

1.1 *DM644x 数字评估模块包*

DM644x EVM 套件是一个针对嵌入式 Linux 开发人员社区的硬件和软件包集合。

这些硬件组件包括：

- TMS320DM6446 基于器件的开发板
- NTSC/PAL 摄像机（视地区而定）
- NTSC/PAL LCD 显示器（视地区而定）
- 麦克风
- 红外 (IR) 遥控
- 40GB，2.5 英寸 IDE 硬盘

此开发板有多个附件和 I/O 接口，例如 USB，10/100Mbps 以太网，视频输入（复合），视频输出（模拟或数字），音频输入（线路或麦克风），音频输出（S.PDIF，模拟）和 UART。 此电路板还包括 4MB SRAM 存储器，16MB NOR 内存，64MB NAND 内存，40GB HDD 和 256MB DDR2 内存。

要获得 DVEVM 可用特性的详细列表，请翻阅技术参考。 这个项目所需的硬件特性请见表 1。

表 1. 所需的 DVEVM 硬件特性

类型	器件	描述
CPU	DM6446	具有视频加速硬件的双内核多媒体处理器
非易失性内存	NOR 闪存	16MB 可用，0x0200 0000 至 0x02FF FFFF
易失性内存	DDR2	256MB 可用，0x8000 0000 至 0x8FFF FFFF
I/O	发光二级管 (LED)	总共 8 个，可被用于特性指示和/或用户反馈
	以太网	10/100Mbps
	UART0	串行调试端口，设置为 115200 1 个停止位，无奇偶校验，无流控

1.2 *软件组件*

DVEVM 包提供多种软件组件，其中包括使用不同编解码格式的音频、语音和视频编码以及解码的多媒体演示。 然而，在这个项目中，只需 ARM Linux 工具链、引导加载程序和 Linux 支持包即可完成构建尽可能小的基于闪存的 Linux 内核（具有针对 DM664x DVEVM 的 HTTP 服务器）的构建目标。

表 2显示了认为可以与这个项目一同使用的可用组件列表。 虽然列表中包括了软件包版本，但是有可能提供更新的版本。

表 **2.** 所需的 **DVEVM** 软件包

项目	版本	注释
ARM Linux 工具链	MVL Pro 4.0.0	包括在 DVEVM 软件包内
Linux 支持包	MVL-401c	包括在 DVEVM 1.10 发布版本中
引导加载程序	u-boot-1.1.3	包括在 DVEVM 1.10 发布版本中
RAM 磁盘	MVL Pro 4.0.0	包括在 DVEVM 软件包内
HTTP 网页服务器	MVL Pro 4.0.0	包括在 DVEVM 软件包内

2 特性选择和内核构建步骤

如果从裸芯片开始的话，构建一个嵌入式 Linux 内核会比较复杂。 驱动程序必须被移植或开发、测试并且在交叉开发工具链上兼容，并且针对 DM644x 器件上的 ARM926EJS 处理器对上层协议栈进行更新或重定位。 DVEVM 包已经包含了大多数可用工具，诸如一个 ARM GNU 工具套件，一个具有 ARM Linux 内核 v2.6 的 Linux 支持包和您的项目需要的全部驱动程序。

这个部分假定您已经按照《*DVEVM 入门指南*》(SPRUE66) 的4 节中描述的那样安装了 DVEVM 软件。《*DVEVM 入门指南*》的4 节还记录了构建一个 Linux 内核的常见命令。

因此，构建一个嵌入式 Linux 内核包括两个简单的步骤：

- 配置内核来选择所需的驱动程序和特性
- 编译内核以创建一个合适的镜像，uImage，这样 u-boot 可在 DVEVM 上加载

2.1 内核配置

Linux 内核特性集中在内核目录顶层的*.config*文件内。 这个文件在构建过程中由 GNU 制造工具使用。 虽然您可以直接编辑*.config*文件来将特性打开或关闭，我们还是提供了几个菜单操作方法来简化这个步骤。 虽然最早的方法是*make menuconfig*，但是受欢迎的方法是诸如使用 X-windows 环境的*make xconfig*，或者使用 GTK+ 环境的*make gconfig*的图形化方法。

下面的部分描述了使用 xconfig 来执行配置的示例。 如果您已经熟悉配置步骤，请使用表 3来确定必须从 DVEVM 的缺省 LSP 中选择或取消选中的特性，

表 **3.** 配置汇总

启用	禁用
ARM 系统类型 (TI-Davinci)	支持可加载模块
基于 TI DM644x 的系统	支持可加载的内置固件
TI Davinci EVM	MTD 支持
TI Davinci I2C 扩展器	支持回路器件
ARM EABI 支持	ATA/ATAPI 支持
高分辨率定时器	SCSI 支持
支持网络连接	支持输入器件
支持初始化 RAM 磁盘	支持 Linux 视频
内核 .config 文件支持	Ext3/XFS/Minix/Automounter/MSDOS/VFAT/CRAMFS/NFS 支持
针对小型器件配置内核	支持数据帧缓冲器件
POSIX 消息队列	USB 支持
系统 V IPC	声卡支持
ELF 支持	多媒体卡 (MMC) 支持
8250 串行驱动器支持	

2.1.1 配置步骤

以下的步骤假定缺省安装的内核树已经在编译前被复制到*/home/user/workdir/lsp*上的一个私有位置中。 此外，请注意，内核树的目录名称可以从软件包的一个版本改成另外一个。

> 注: DVEVM 和 DVSDK-L 或 -3L 软件包也许会在内核配置和构建命令方面有轻微的不同。 始终检查与软件包一同提供的文档以获得与构建步骤相关的准确命令的更新信息，诸如 DVEVM 版本说明，DVEVM 快速开始指南或《*DVEVM 入门指南*》(SPRUE66)。 下面是针对 DVEVM 软件包的内核树的步骤。

1. 在主机 Linux 工作站上，前往内核树的根目录：
   ```
   host $ cd /home/user/working/lsp/ti-davinci
   ```
2. 启动 Linux 内核配置工具：
   ```
   host $ make ARCH=arm CROSS_COMPILE=arm_v5t_le- xconfig
   ```
3. 在*Loadable module support*下，取消选中*Enable loadable module support（启用可加载模块支持）*来禁用模块加载特性。 请见图 1。

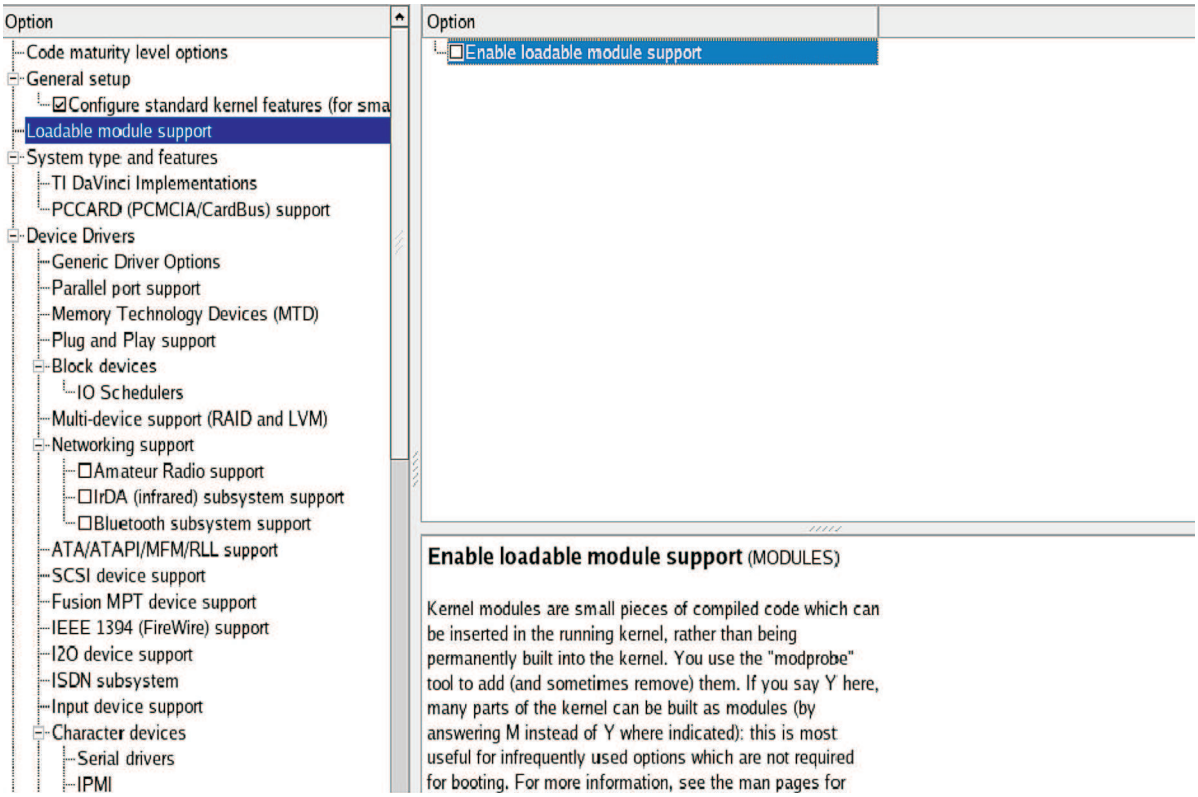

图 1. 可加载模块支持

4. 在*Device Drivers（器件驱动程序）→Generic Driver Options（通用驱动程序选项）*下，取消选中 *Select only drivers that don't need compile-time external firmware（只选择不需要编译时间外部固件的驱动程序）*和*Prevent firmware from being built（防止固件被构建）*复选框以禁用固件加载特性。
5. 在*Device Drivers→Memory Technology Devices (MTD)（内存技术器件）*下，取消选中*Memory Technology Devices (MTD) support（MTD 支持）*复选框以禁用内存技术驱动程序支持。
6. 在*Device Drivers→Block devices（阻断器件）*下，取消选中*Loopback device support（回路器件支持）*复选框来禁用被用来安装一个 ISO 镜像文件的回路器件支持。
7. 在*Device Drivers→ATA/ATPI/MFM/RRL support*下，取消选中*ATA/ATPI/MFM/RRL support*复选框来禁用被用来访问 EVM 硬盘的 ATA 支持。
8. 在*Device Drivers→ SCSI device support（SCSI）器件支持*，取消选中*legacy /proc/scsi/ support* 和*SCSI disk support* 复选框来禁用 EVM 上的 SCSI 磁盘支持。

9. 在*Device Drivers→Input device support（输入器件支持）*下, 取消选中*Mouse interface（鼠标接口），Event interface（事件接口）*和*Keyboards（键盘）*复选框来禁用输入器件支持。
10. 在*Device Drivers→ Multimedia devices（多媒体器件）*下，取消选中*Video For Linux（针对 Linux 的视频）*复选框来禁用被用来从摄像机捕捉视频图像的 V412 驱动程序支持。
11. 在*Device Drivers→File systems（文件系统）*下，取消选中*Ext3 journalling file system support（Ext3 日志文件系统支持），XFS file system support，Minix fs support，Dnotify support*和*Kernel automounter version 4 support（内核自动加载版本 4 支持）*复选框以禁用文件系统支持。 不要取消 ext2 文件系统支持的选中，这是因为 ext2 文件系统用于初始化 RAM 磁盘。 请见图 2。

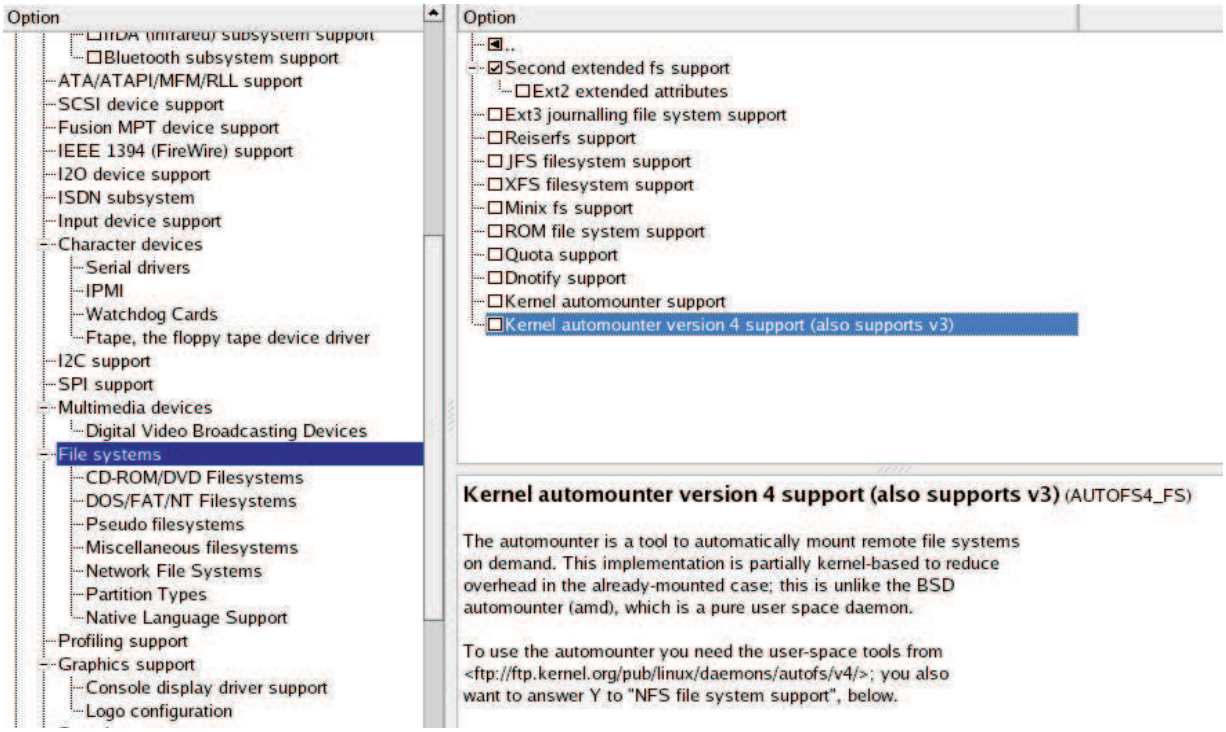

图 **2.** 禁用文件系统

12. 在*Device Drivers→ File systems→ DOS/FAT/NT Filesystems*下，取消选中*MSDOS fs support*和*VFAT (Windows 95) fs support*复选框以禁用 Windows 文件系统支持。
13. 在*Device Drivers→File systems→Miscellaneous filesystems（混合文件系统）*下，取消选中*Compressed ROM file system support (cramfs)（压缩 ROM 文件系统支持）*复选框以禁用 cramfs 文件系统支持。
14. 在*Device Drivers→ File systems→ Network File Systems*下，取消选中*NFS file system support，NFS server support（NFS 服务器支持）*和*SMB file system support*复选框以禁用网络文件系统支持。
15. 在*Device Drivers→ File systems→Partition Types（分区类型）*下，取消选中*Advanced Partition Selection（高级分区选择）*复选框以禁用硬盘上的分区支持。
16. 在*Device Drivers→Graphics Support（图形化支持）*下，取消选中*Support for frame buffer devices（支持数据帧缓冲器件）*复选框来禁用 Linux 数据帧缓冲支持。
17. 在*Device Drivers→Sound（声音）*下，取消选中*Sound card support（声卡支持）*复选框以禁用 Linux 声音支持。
18. 在*Device Drivers →USB Support*下，取消选中*Support for Host-side USB（支持主机侧 USB）*和*Inventra USB Highspeed Dual Role Controller Support（Inventra USB 高速双重角色控制器支持）*复选框以禁用 USB 驱动器支持。
19. 在 *Device Drivers→MMC/SD Card Support*下，取消选中*MMC Support*复选框以禁用多媒体卡支持。

2.2 内核编译

这个部分描述了内核编译的步骤。

> **注:** DVEVM 和 DVSDK-L 或 -3L 软件包也许具有不同的内核配置和构建命令。 始终检查诸如 DVEVM 版本说明，DVEVM 快速启动指南或《*DVEVM 入门指南*》(SPRUE66) 等随软件包一同提供的文档以获得针对构建步骤的准确命令。 下面步骤针对 DVEVM 软件包中的内核树。

1. 如果未以*user*（*用户*）的身份登录，那么在构建内核前请登录为*user*。
2. 使用这个命令来构建 Linux 内核：
   ```
   host$ make ARCH=arm CROSS_COMPILE=arm_v5t_le- uImage
   ```

请注意，上面的内核配置禁用了大多数外设支持，除了网络互连堆栈、以太网和串行驱动器。 如果除了这个应用报告内使用的应用程序之外还需要额外的应用程序，那么也许需要启用其它支持特性。

生成的内核，u-boot 兼容已压缩二进制文件*uImage*，位于 arch/arm/boot 目录下。 将这个文件复制到 /tftpboot 目录下，这样在晚些时候可在 DVEVM 上将这个文件存入闪存。

在下一个步骤中，您将构建一个 RAM 磁盘文件系统来保存至闪存。

3 构建一个初始化 RAM 磁盘文件系统

虽然这个部分与之前的部分无关，但是我们认为您已经按照《*DVEVM 入门指南*》(SPRUE66) 中概括的步骤将 DVEVM 软件安装在 Linux 主机上。

在引导内核前，一个初始化 RAM 磁盘依靠一个引导加载程序（例如 u-boot）将其从非易失性内存（例如 NOR 闪存）加载到易失性内存（例如 DDR2）内。 RAM 磁盘内的文件系统是指一个初始化 RAM 磁盘文件系统，或者 initrd。 这个文件系统可作为引导文件系统安装，而应用程序可以从这个文件系统被执行。 这是内核的本地存储。 由于它安装在易失性内存中，它的内容在系统断电时丢失。 对于大多数嵌入式系统，这是一个希望得到的运行环境。 如果您必须保存在运行期间生成的一些参数，您将需要一个 NOR 闪存文件系统，而这个系统不在本项目的范围内。

借助于 Davinci EVM 平台，您可以使用 MontaVista® DevRocket™ IDE（在 -L 或 -3L 软件包内提供），或者命令行脚本（在 DVEVM 和 DVSDK 软件包内提供）来构建一个 RAM 磁盘文件系统。

要执行网页服务器，初始化 RAM 磁盘文件系统应该包含以下 GNU 软件包。 他们可在 <tool chain install directory>/pro/devkit/arm/v5t_le/packages/pro 或 pro/optional 下的 DVEVM 软件 MVL Pro 安装目录中找到。

表 4. 用于 RAM 磁盘文件系统的 Linux 软件包

项目	版本	说明
busybox	1.00r3-5.0.0	将很多常见 Linux 工具的小版本组合在一起。
initscript	2.85-3.0.0	包含用来引导系统的基本系统脚本。
netbase	4.17-1.0.1	为 TCP/IP 网络连接提供必要的基础结构
thttpd	2.25b-1.0.0	包含一个小型、快速且安全的网页服务器，其中包括 CGI 支持，基于 URL 流量的流量控制和基本认证。

注: 交叉构建这里提供的或者源于 GNU 源代码树的这些软件包超出了这份应用报告的范围。 请翻阅合适的文档或嵌入式 Linux 书籍来了解如何执行这些任务。

有几个可用的选项来完成这个步骤，其中包括使用现有的 RAM 磁盘，或者按照您的需要构建一个。

3.1 使用一个现有的 RAM 磁盘

为了节省时间，RAM 磁盘与 DVEVM ARM Linux 软件工具链一同提供。 它位于：

<tool chain install directory>/pro/devkit/arm/v5t_le/images

注: 在 DVSDK 软件包的晚些版本中，同样的 RAM 磁盘位于：

<dvsdk install dir>/<PSP dir>/bin

在这个目录中，RAM 磁盘文件被称为 ramdisk.gz（gunzipped 压缩后大约 2.1MB）。 运行时，它占用 6.3MB 的 DDR 空间。 这个文件系统包含一些用于这个项目的不必要的工具，但是适合用于一个典型的嵌入式系统。

3.2 按照使用要求设置 RAM 磁盘

1. 将现有的初始化 RAM 磁盘复制到一个临时的位置：

```
host $ mkdir /mnt/def_cd
host $ cp <tool chain install dir>/pro/devkit/arm/v5t_le/images/ramdisk.gz
/mnt/def_cd
```

2. 解压缩文件，生成一个名为*ramdisk 的文件*：

```
host $ gzip –d /mnt/def_cd/ramdisk.gz
```

3. 创建一个安装点，并将 RAM 磁盘安装使用：

```
host $ mkdir –p /mnt/def_cd/ram0
host $ mount –o loop /mnt/def_cd/ramdisk /mnt/def_cd/ram0
```

您可以通过改变已安装目录并列出内容来浏览 RAM 磁盘内容：

```
host $ cd /mnt/def_cd/ram0
host $ ls
```

控制台输出显示典型 Linux 目录结构。

在下个步骤中，您将合适的应用包和 http 网页服务器以及某些初始化添加到 RAM 磁盘中，并在将其再次压缩至闪存中之前使用脚本。

4 应用支持。

这个部分描述了如何将一个小型网页服务器 (thttpd) 添加到初始化 RAM 磁盘文件系统，并针对 DVEVM 对其进行配置。

网页服务器 thttpd 是一款简单、小型、便携、快速且安全的 HTTP 服务器，它具有以下特性：

- 简单：它只处理执行 HTTP/1.1 所必须的最少信息。
- 小型：它运行时间较短，这是因为它谨慎地分配内存，并且不进行分岔程序操作。
- 便携：它在大多数与 Unix 类似的操作系统内完全编译，其中包括 FreeBSD，SunOS 4，Solaris 2，BSD/OS，Linux 和 OSF。
- 快速：在常见使用中，它的速度与最全面特性的服务器 (Apache，NCSA，Netscape) 一样快。 在极端负载下，它的速度更快。
- 安全：它保护网页服务器机器不受其他网站的攻击和侵入。

4.1 构建 http 网页服务器

您可以使用 ARM gcc 工具链在主机开发 PC 或本地 EVM 上构建网页服务器。 这个部分说明了如何在主机开发 PC 上交叉构建网页服务器。

- 从开发人员网站上下载最新的 thttpd：http://www.acme.com/software/thttpd/，或者使用作为 DVEVM 软件包一部分而提供的源代码。 下面的指令用于打开且编译开源版本。 DVEVM 软件包已经将源文件安装在 <dvevm install dir>/examples/thttpd-2.25b 目录下。 在更高版本中，到源文件的路径也许是 <dvsdk install dir>/examples/<device>/thttpd-2.25b ，而 thttpd 二进制数据也许已经存在于这个位置内。
  - `host $ cd ~/workdir`
  - `host $ tar xzf path-to-tar-file/thttpd-2.25b.tar.gz`
  - `host $ cd thttpd-2.25b`
- 验证到 ARM 交叉编译工具链的路径已经按照《*DVEVM 入门指南*》(SPRUE66) 中描述的那样被导出。
- 按如下命令编译网页服务器：
  - `host $ CC=arm_v5t_le-gcc ./configure`
  - `host $ make`

4.2 测试网页服务器

1. 将可执行的 thttpd 复制到 EVM 板。 《*DVEVM 入门指南*》描述了 HDD 或 NFS 配置的过程。
2. 使用一个端口号为 8000 的仲裁端口在 EVM 上运行 thttpd。
   `dvevm $./thttpd –p 8000`
3. 使用 PC 的浏览器来连接到全新的网页服务器。 URL 是 EVM 板的 IP 地址和附加端口号：http://evm-ip-address:8000 当您在 DVEVM 上运行 thttpd 程序时，URL 即被建立。 您应该看到类似的输出：

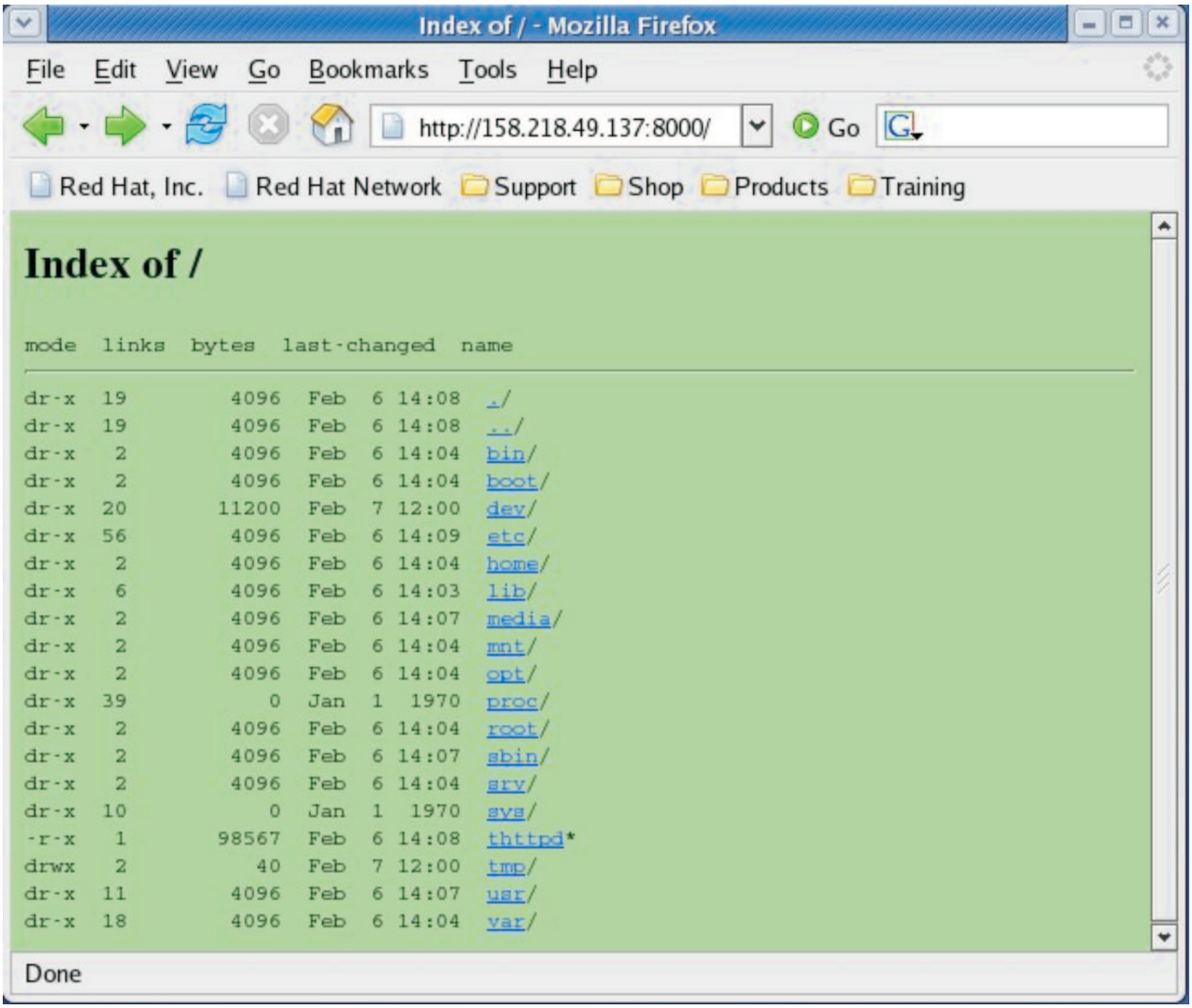

图 **3. Mozilla Firefox** 的索引

4.3 将网页服务器添加到初始化 RAM 磁盘

这个部分描述了如何修改现有的初始化 RAM 磁盘以包括 thttpd 网页服务器和 cgi 脚本。 为了节省时间，网页服务器和示例 index.html 页所使用的 TI 图形文件已经包含在 DVEVM/DVSDK 软件包中，它们可从http://www.ti.com/dvevmupdates内下载。 与创建这个演示中所使用的 index.html 文件和 cgi 脚本相关的更多信息请参见附录 A。

1. 在 RAM 磁盘上创建一个网页目录，并将 thttpd 可执行文件复制到这个目录中：

```
host $ mkdir -p /mnt/def_cd/ram0/opt/dvevm/web
host $ cp thttpd /mnt/def_cd/ram0/opt/dvevm/web
```

2. 写入 index.html 文件，并将其复制到初始化 RAM 磁盘文件系统中。 要获得与本演示中所使用的 index.html 文件相关的信息，请参考附录 A。

```
host $ cp index.html /mnt/def_cd/ram0/opt/dvevm/web
```

3. 写入 cgi 脚本并将其复制到初始化 RAM 磁盘文件系统的 cgi-bin 目录中。 要获得与本演示中所使用的 cgi 脚本相关的信息，请参考附录 A。

```
host $ mkdir -p /mnt/def_cd/ram0/opt/dvevm/web/cgi-bin
host $ cp <cgi files> /mnt/def_cd/ram0/opt/dvevm/web/cgi-bin/
```

4. 将 index.html 文件中使用的 TI 图形复制到 RAM 磁盘文件系统。 这些图形位于 DVEVM/DVSDK 软件包内，可从http://www.ti.com/dvevmupdates内下载。

```
host $ cp <dvevm/dvsdk install dir>/examples/web/*.gif
/mnt/def_cd/ram0/opt/dvevm/web
```

注: 对于更高版本的 DVEVM/DVSDK 软件，TI 图形文件的位置可能更像是 <dvsdk install dir>/examples/<device>/web

5. 编辑一个名为 startweb 的文件，并添加以下脚本行以启动网页服务器：

```
#!/bin/sh
# script to start web server
echo "Start web service..."
/opt/dvevm/web/thttpd -d /opt/dvevm/web -c "/cgi-bin/*"
```

6. 将这个脚本复制到 RAM 磁盘的 /etc/init.d 目录中以使其成为引导序列的一部分：

```
host $ cp startweb.sh /mnt/def_cd/ram0/etc/init.d
host $ chmod +x /mnt/def_cd/ram0/etc/init.d/startweb.sh
host $ cd /mnt/def_cd/ram0/etc/rc.d/rcS.d
host $ ln –s ../init.d/startweb.sh S42startweb
```

7. 最后，重新压缩这个 RAM 磁盘用于闪存存储：

```
host $ cd /mnt/def_cd
host $ umount /mnt/def_cd/ram0
host $ gzip ramdisk
host $ cp ramdisk.gz /tftpboot
```

5 将信息复制到 NOR 闪存

这个部分需要完成2 节，3 节和4 节。 在这个部分中，您将把内核镜像文件和初始化 RAM 磁盘复制到 NOR 闪存中。

1. 将内核镜像文件复制到 /tftpboot 目录中，如果您还没有这么做的话：

```
host $ cp ~/workdir/lsp/ti-davinci/arch/arm/boot/uImage /tftpboot
```

2. 将初始化 RAM 磁盘文件系统复制到 /tftpboot 目录中，如果您还没有这么做的话：

```
host $ cp /mnt/def_cd/ramdisk.gz /tftpboot/
```

3. 通过 TFTP 下载 Linux 内核：

```
DVEVM # setenv serverip <tftp server ip address>
DVEVM # setenv bootfile uImage
DVEVM # dhcp
BOOTP broadcast 1
 *** Unhandled DHCP Option in OFFER/ACK: 44
*** Unhandled DHCP Option in OFFER/ACK: 46
DHCP client bound to address <dvem ip address>
TFTP from server <tftp server ip address>; our IP address is <dvevm ip
```

```
address>
Filename 'uImage'.
Load address: 0x80700000
Loading:
#################################################################
#################################################################
done
Bytes transferred = 823844 (c9224 hex)
```

dhcp 命令包含 IP 设置，然后下载 Linux 内核镜像文件（正如 serverip 和 bootfile 环境变量所指定的那样）。 请注意载入地址 (0x80700000) 和传输的字节 (0xc9224)，这是因为下面的步骤需要这些数字。

4. 通过 TFTP 下载 RAM 磁盘文件系统：

```
DVEVM # tftp 0x85000000 ramdisk.gz
TFTP from server <tftp server ip address>; our IP address is <dvevm ip
address>
Filename 'ramdisk.gz'.
Load address: 0x85000000
Loading:
#################################################################
#################################################################
done
Bytes transferred = 2304639 (232a7f hex)
```

tftp 命令下载 0x85000000 地址上的 ramdisk.gz 文件。 请注意载入地址 (0x85000000) 和传输的字节 (0x232a7f)，这是因为下面的步骤需要这些数字。

5. 确定闪存中存储镜像文件的位置：

```
EVM # flinfo
Bank # 1: MY AMD 29LV256M (256 Mbit)
 Size: 16 MB in 256 Sectors
Sector Start Addresses:
```

02000000	02010000	02020000	02030000	02040000 (RO)
02050000	02060000	02070000	02080000	02090000
020A0000	020B0000	020C0000	020D0000	020E0000

U-Boot 编码和数据保存在头五个扇区内，起始地址 0x2000000。 请注意，尾标*(RO)*表示此扇区是只读的或者受到擦除和写入保护。 Linux 内核镜像文件应该被保存在 0x2050000 位置上，U-Boot 之后的第一个空闲扇区。

注: 对于 Intel NOR 闪存芯片，您应该使用闪存地址 0x2060000，这是 U-Boot 之后的第一个起始地址，而不是 0x2050000。 应该对下面的命令进行调整以使用这个地址。

6. 擦除闪存：

```
DVEVM # protect off 0x2050000 +0x2FBCA3
```

注: 0x2FBCA3 是内核镜像和 ramdisk 镜像尺寸的和。

```
DVEVM # erase 0x2050000 +0x2FBCA3
Erasing sector 5 ... done.
```

```
Erasing sector 6 ... done.
```

protect off（*保护关闭*）命令使得闪存可被写入（对这个示例不是必须的），而擦除命令通过擦除之前的内容为闪存写入做好准备。 请注意，起始地址 (0x2050000) 源自 flinfo 的输出和长度（Linux 内核镜像的大小加上通过 TFTP 服务器下载的 RAM 磁盘文件系统的大小）。

7. 从 RAM 复制到闪存：

```
DVEVM # cp.b 0x80700000 0x2050000 0xc9224
```

Copy to Flash.../done

```
DVEVM # cp.b 0x85000000 0x2119224 0x232a7f
```

注： 目的地址 0x2119224 源自将内核大小 0xc9224 与起始地址 0x2050000 相加。 这样 RAM 磁盘镜像被写入内核后的闪存扇区内。

Copy to Flash.../done

cp（复制）命令被用来将 RAM 中的 Linux 内核镜像复制到可访问的闪存存储器。 自变量是源地址、目标地址和长度。 cp 命令上的 .b 扩展名指定一个字节复制。

8. 防止写入闪存：

```
DVEVM # protect on 0x2050000 +0x2FBCA3
```

保护命令使得闪存扇区只读，以确保内核镜像和 RAM 磁盘文件系统不会被意外覆盖。

9. 设定 U-Boot 命令和 Linux 内核命令行：

注： Intel NOR 闪存芯片使用闪存地址 0x2060000 而非 0x2050000

```
DVEVM # setenv bootargs console=ttyS0,115200n8 ip=dhcp root=/dev/ram0 rw
initrd=0x85000000,6M
DVEVM# setenv bootcmd ‘cp.b 0x2119224 0x85000000 0x232a7f; bootm 0x2050000’
```

引导命令被设定以使用 0x2050000 地址上的内核镜像。 bootcmd 首先在步骤 7 将 RAM 磁盘镜像从它被写入的位置复制到 RAM 中的位置。 Linux 内核命令行自变量 (bootargs) 被设置成将 RAM 磁盘用作引导文件系统，并且在 RAM 指定其位置，在这个位置上，RAM 磁盘在 bootcmd 操作中被复制。

10. 现在，系统已经为引导做好准备，所有保存 u-boot 环境变量：

```
DVEVM # saveenv
DVEVM # boot
```

现在 Linux 应该从闪存引导，而引导文件系统应该被挂接到 /dev/ram0 中。

表 5总结了 u-boot，Linux 内核和已压缩的 RAM 磁盘在 16MB NOR 闪存存储器上被载入的位置。

表 5. DVEVM NOR 闪存内引导加载程序，内核和 RAM 磁盘的存储器位置

地址	内容
0x0200 0000 – 0x0204 FFFF	u-boot 和 u-boot 参数 (327KB)
0x0205 0000 – 0x0211 9223	uImage – Linux 内核 (823KB)
0x0211 9224 – 0x0234BCA3	已压缩的 RAM 磁盘 (2.1MB)
0x0234BCA4 – 0x02FF FFFF	未使用 (12.70MB)

6 引导

这个部分要求5 节完成。 它描述了如何访问网页服务器以及登陆到 EVM 中。

1. 为 EVM 板供电。 成功引导时，会出现登陆提示。 请见图 4。

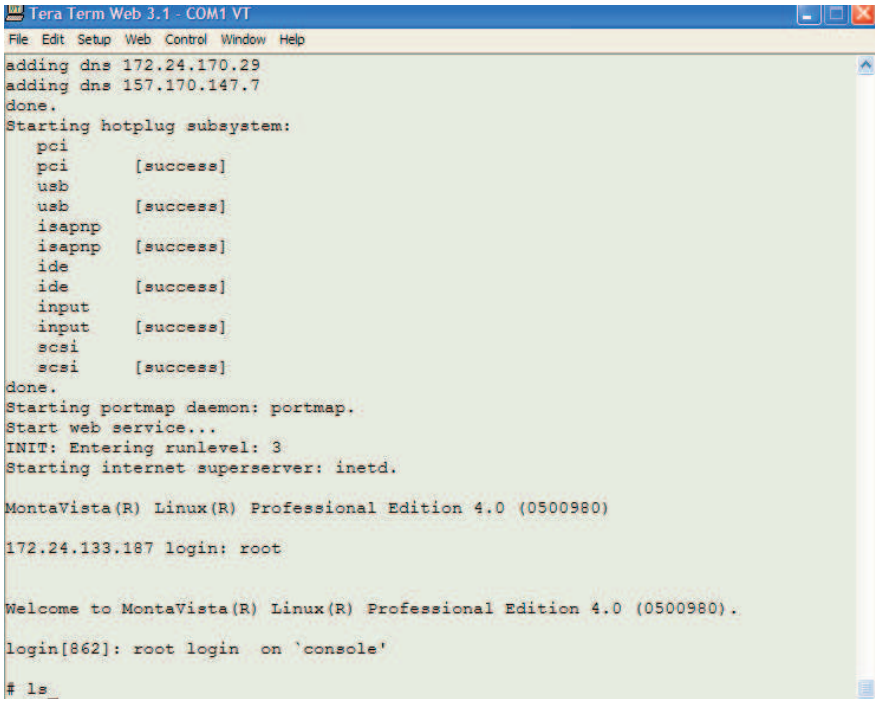

图 **4. EVM** 引导屏幕

2. 登陆为*root*。
3. 打开主机上的网页浏览器，并连接到 EVM。
4. 通过在 URL 地址框内敲入 EVM IP 地址来连接到 EVM 网页服务器。 请见图 5。

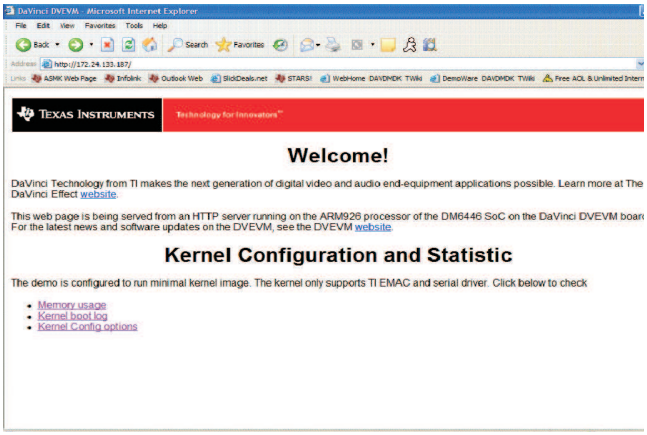

图 **5.** 被连接到 **DVEVM** 网页服务器的网页屏幕显示

5. 单击*Memory usage*（*内存使用量*）链接。 网页显示了*cat /proc/meminfo*命令的输出。
6. 单击*Kernel Config options*（*内核配置选项*）链接。 此网页显示被用来构建内核镜像的*.config*文件。
7. 单击*Kernel boot log*（*内核引导日志*）链接。 网页显示了*dmesg*命令的输出。

7 概要

这份应用报告描述了使用 Davinci DVEVM 软件包进行内核配置和构建步骤。 选择了一个小型内核特性集，而且一个 http 服务器软件包被添加到 RAM 磁盘中以用作引导文件系统。 随后内核和 RAM 磁盘闪存入 NOR 闪存存储器，这里驻留了 u-boot 和引导加载程序。 然后，您启动这个系统，并证明一个网页浏览器能够连接到运行这个内核和 http 服务器的 DVEVM。

正如摘要中提到的那样，这个设置的类型可被用作一个用 Davinci DM644x EVM 进行嵌入式 Linux 系统开发的起始点。 可添加其它特性，诸如：

- NOR 闪存文件系统（例如 jffs2 文件系统），能够提供持续的本地存储。
- V4L2 驱动器，从而使视频图像可以被捕获并能使用运行在 DM644x 器件的 DSP 上的编码器进行压缩。这可将 DVEVM 变成一个视频服务器。

8 参考书目

- 《*Davinci-DM644x 评估模块技术参考*》，Spectrum Digital，508165-0001. 最新版本请访问www.spectrumdigital.com上的 Spectrum Digital 网页。
- 《*DVEVM 入门指南*》()。 www.ti.com/dvevmupdates网站。

附录 A

A.1 创建 index.html 文件

编辑一个名为 index.html 的文件并将以下的命令行添加其中：

```
<HTML>
<HEAD>
<TITLE>DaVinci DVEVM</TITLE>

<STYLE TYPE="text/css">
BODY,H1,H2,H3,H4,H5,H6,P,CENTER,TD,TH,UL,DL,DIV {
     font-family: Geneva, Arial, Helvetica, sans-serif;
}
H1 {
     text-align: center;
}
CAPTION { font-weight: bold }
</STYLE>

</HEAD>

<BODY>

<table width=100%>
<tr>
  <td bgcolor="black" width="1"><a href="http://www.ti.com"><img border=0
src="tilogo.gif"></a></td>
  <td bgcolor="red"><img src="titagline.gif"></td>
</tr>
</table>


<H1>Welcome!</H1>
<P>
DaVinci Technology from TI makes the next generation of digital video and
audio end-equipment applications possible. Learn more at The DaVinci Effect
<A HREF="http://www.thedavincieffect.com">website</A>.
</P>

<P>
This web page is being served from an HTTP server running on the
ARM926 processor of the DM6446 SoC on the DaVinci DVEVM board. For the
latest news and software updates on the DVEVM, see the DVEVM
<A HREF="http://www.ti.com/dvevmupdates">website</A>.
</P>

<H1> Kernel Configuration and Statistic</H1>
<P>
The demo is configured to run minimal kernel image. The kernel only supports TI EMAC and serial
driver. Click below to check

<UL>
     <LI> <A HREF="/cgi-bin/memory">Memory usage</A> </LI>
     <LI> <A HREF="/cgi-bin/log">Kernel boot log</A> </LI>
     <LI> <A HREF="/cgi-bin/config">Kernel Config options</A> </LI>
</UL>

</BODY>
</HTML>
```

A.2 创建 CGI 脚本

1. 编辑一个名为存储器的文件，并添加以下的脚本行来创建存储器使用脚本：

```
#!/bin/sh

cat << EOF
Content-type: text/plain
Cache-control: no-cache

EOF

echo "#cat /proc/meminfo"
cat /proc/meminfo

echo
echo "# free -b"
free -b

echo
echo "# ps -el"
ps -el
```

2. 编辑一个名为存储器的文件，并添加以下的脚本行来创建内核引导日志脚本：

```
#!/bin/sh

cat << EOF
Content-type: text/plain
Cache-control: no-cache

EOF

echo
echo "#dmesg"
dmesg
```

3. 编辑一个名为存储器的文件，并添加以下的脚本行来创建内核配置选项脚本：

```
#!/bin/sh

cat << EOF
Content-type: text/plain
Cache-control: no-cache

EOF

rm -rf /tmp/config*
cp /proc/config.gz /tmp
gzip -d /tmp/config.gz
cat /tmp/config
```

4. 使脚本可执行：

```
chmod +x memory
chmod +x log
chmod +x config
```

重要声明

德州仪器(TI) 及其下属子公司有权根据 JESD46 最新标准, 对所提供的产品和服务进行更正、修改、增强、改进或其它更改， 并有权根据 JESD48 最新标准中止提供任何产品和服务。客户在下订单前应获取最新的相关信息, 并验证这些信息是否完整且是最新的。所有产品的销售都遵循在订单确认时所提供的TI 销售条款与条件。

TI 保证其所销售的组件的性能符合产品销售时 TI 半导体产品销售条件与条款的适用规范。仅在 TI 保证的范围内，且 TI 认为 有必要时才会使用测试或其它质量控制技术。除非适用法律做出了硬性规定，否则没有必要对每种组件的所有参数进行测试。

TI 对应用帮助或客户产品设计不承担任何义务。客户应对其使用 TI 组件的产品和应用自行负责。为尽量减小与客户产品和应 用相关的风险，客户应提供充分的设计与操作安全措施。

TI 不对任何 TI 专利权、版权、屏蔽作品权或其它与使用了 TI 组件或服务的组合设备、机器或流程相关的 TI 知识产权中授予 的直接或隐含权限作出任何保证或解释。TI 所发布的与第三方产品或服务有关的信息，不能构成从 TI 获得使用这些产品或服 务的许可、授权、或认可。使用此类信息可能需要获得第三方的专利权或其它知识产权方面的许可，或是 TI 的专利权或其它 知识产权方面的许可。

对于 TI 的产品手册或数据表中 TI 信息的重要部分，仅在没有对内容进行任何篡改且带有相关授权、条件、限制和声明的情况 下才允许进行复制。TI 对此类篡改过的文件不承担任何责任或义务。复制第三方的信息可能需要服从额外的限制条件。

在转售 TI 组件或服务时，如果对该组件或服务参数的陈述与 TI 标明的参数相比存在差异或虚假成分，则会失去相关 TI 组件 或服务的所有明示或暗示授权，且这是不正当的、欺诈性商业行为。TI 对任何此类虚假陈述均不承担任何责任或义务。

客户认可并同意，尽管任何应用相关信息或支持仍可能由 TI 提供，但他们将独力负责满足与其产品及在其应用中使用 TI 产品 相关的所有法律、法规和安全相关要求。客户声明并同意，他们具备制定与实施安全措施所需的全部专业技术和知识，可预见 故障的危险后果、监测故障及其后果、降低有可能造成人身伤害的故障的发生机率并采取适当的补救措施。客户将全额赔偿因 在此类安全关键应用中使用任何 TI 组件而对 TI 及其代理造成的任何损失。

在某些场合中，为了推进安全相关应用有可能对 TI 组件进行特别的促销。TI 的目标是利用此类组件帮助客户设计和创立其特 有的可满足适用的功能安全性标准和要求的终端产品解决方案。尽管如此，此类组件仍然服从这些条款。

TI 组件未获得用于 FDA Class III（或类似的生命攸关医疗设备）的授权许可，除非各方授权官员已经达成了专门管控此类使 用的特别协议。

只有那些 TI 特别注明属于军用等级或“增强型塑料”的 TI 组件才是设计或专门用于军事/航空应用或环境的。购买者认可并同 意，对并非指定面向军事或航空航天用途的 TI 组件进行军事或航空航天方面的应用，其风险由客户单独承担，并且由客户独 力负责满足与此类使用相关的所有法律和法规要求。

TI 已明确指定符合 ISO/TS16949 要求的产品，这些产品主要用于汽车。在任何情况下，因使用非指定产品而无法达到 ISO/TS16949 要求，TI不承担任何责任。

	产品		应用
数字音频	www.ti.com.cn/audio	通信与电信	www.ti.com.cn/telecom
放大器和线性器件	www.ti.com.cn/amplifiers	计算机及周边	www.ti.com.cn/computer
数据转换器	www.ti.com.cn/dataconverters	消费电子	www.ti.com/consumer-apps
DLP® 产品	www.dlp.com	能源	www.ti.com/energy
DSP - 数字信号处理器	www.ti.com.cn/dsp	工业应用	www.ti.com.cn/industrial
时钟和计时器	www.ti.com.cn/clockandtimers	医疗电子	www.ti.com.cn/medical
接口	www.ti.com.cn/interface	安防应用	www.ti.com.cn/security
逻辑	www.ti.com.cn/logic	汽车电子	www.ti.com.cn/automotive
电源管理	www.ti.com.cn/power	视频和影像	www.ti.com.cn/video
微控制器 (MCU)	www.ti.com.cn/microcontrollers		
RFID 系统	www.ti.com.cn/rfidsys		
OMAP应用处理器	www.ti.com/omap		
无线连通性	www.ti.com.cn/wirelessconnectivity	德州仪器在线技术支持社区	www.deyisupport.com

邮寄地址： 上海市浦东新区世纪大道 1568 号，中建大厦 32 楼 邮政编码： 200122