



追求卓越 创造精品

TO BE BEST

TO DO GREAT

广州友善之臂计算机科技有限公司

# MINI2440 用户手册

2009-02-20



copyright@2007-2009



追 求 卓 越 创 造 精 品

TO BE BEST

TO DO GREAT

广州友善之臂计算机科技有限公司

## 版 权 声 明

本手册版权归属广州友善之臂计算机科技有限公司（以下简称“友善之臂”）所有，并保留一切权力。未经友善之臂同意(书面形式)，任何单位寄个人不得擅自摘录本手册部分或全部，违者我们将追究其法律责任。

敬告：

在售开发板的手册会经常更新，请在 <http://www.arm9.net>网站查看最近更新，并下载最新手册，不再另行通知。



追求卓越 创造精品

TO BE BEST

TO DO GREAT

广州友善之臂计算机科技有限公司

## 目 录

第一章 MINI2440 开发板介绍 .....	- 10 -
1.1 MINI2440 开发板简介 .....	- 10 -
1.1.1 MINI2440 开发板外观 .....	- 10 -
1.1.2 MINI2440 开发板硬件资源特性 .....	- 11 -
1.1.3 用户光盘资源说明 .....	- 12 -
1.2 接口布局及跳线 .....	- 12 -
1.2.1 跳线说明 .....	- 13 -
1.2.2 接口布局 .....	- 13 -
1.3 接口资源说明 .....	- 13 -
1.3.1 地址空间分配和片选信号定义 .....	- 14 -
1.3.2 SDRAM存储系统 .....	- 15 -
1.3.3 FLASH存储系统 .....	- 15 -
1.3.4 电源系统及接口 .....	- 16 -
1.3.5 复位系统 .....	- 18 -
1.3.6 用户LED .....	- 18 -
1.3.7 用户按键 .....	- 19 -
1.3.8 A/D输入测试 .....	- 19 -
1.3.9 PWM控制蜂鸣器 .....	- 20 -
1.3.10 串口 .....	- 20 -
1.3.11 USB接口 .....	- 21 -
1.3.12 LCD接口 .....	- 22 -
1.3.13 EEPROM .....	- 23 -
1.3.14 网络接口 .....	- 23 -
1.3.15 音频接口 .....	- 24 -
1.3.16 JTAG接口 .....	- 25 -
1.3.17 GPIO .....	- 26 -
1.3.18 CMOS CAMERA接口 .....	- 27 -
1.3.19 系统总线接口 .....	- 28 -
1.3 linux特性 .....	- 29 -
1.4 WindowsCE特性 .....	- 30 -
第二章 MINI2440 开发板使用说明 .....	- 31 -
2.1 开发板设置及连接 .....	- 31 -
2.1.1 启动模式选择 .....	- 31 -
2.1.2 外部接口连接 .....	- 31 -
2.1.3 设置超级终端 .....	- 31 -
2.2 开发板BIOS功能及使用说明 .....	- 35 -
2.3.1 开机进入BIOS模式 .....	- 35 -
2.2.2 安装USB驱动 .....	- 36 -
2.3.3 功能主菜单说明 .....	- 39 -



2.2.4 分区子菜单功能说明 .....	- 42 -
2.2.5 设置linux启动参数子菜单功能说明 .....	- 45 -
2.3 非操作系统下的外围资源测试 .....	- 49 -
2.3.1 下载运行测试程序 .....	- 49 -
2.3.2 外围资源测试 .....	- 55 -
2.4 Linux之图形界面Qtopia系统测试与使用说明(预装).....	- 62 -
2.4.1 触摸屏校正 .....	- 64 -
2.4.2 主要界面说明 .....	- 65 -
2.4.3 播放Mp3.....	- 66 -
2.4.4 播放视频 .....	- 67 -
2.4.5 图片浏览 .....	- 68 -
2.4.6 计算器 .....	- 68 -
2.4.7 命令终端 .....	- 69 -
2.4.8 网络设置 .....	- 70 -
2.4.9 Ping测试 .....	- 71 -
2.4.10 浏览器 .....	- 73 -
2.4.11 LED测试.....	- 73 -
2.4.12 EEPROM读写测试 .....	- 75 -
2.4.13 PWM控制蜂鸣器测试 .....	- 76 -
2.4.14 触摸笔测试 .....	- 77 -
2.4.15 条码扫描 .....	- 78 -
2.4.16 语言设置 .....	- 79 -
2.4.17 屏幕旋转 .....	- 80 -
2.4.18 关于关机和亮度调节 .....	- 81 -
2.5 Linux之非图形界面系统测试与使用说明 .....	- 81 -
2.5.1 播放mp3 .....	- 83 -
2.5.2 如何中止程序的运行 .....	- 83 -
2.5.3 使用优盘/移动硬盘 .....	- 84 -
2.5.4 使用SD卡 .....	- 85 -
2.5.5 使用USB摄像头抓图.....	- 85 -
2.5.6 如何通过串口与PC互相传送文件.....	- 86 -
2.5.7 如何通过网络远程控制显示USB摄像头 .....	- 88 -
2.5.8 如何控制板上的LED.....	- 90 -
2.5.9 测试板上的按键 .....	- 91 -
2.5.10 串口 2 和 3 的测试 .....	- 92 -
2.5.11 测试蜂鸣器 .....	- 94 -
2.5.12 控制LCD的背光 .....	- 94 -
2.5.13 测试I2C－EEPROM .....	- 95 -
2.5.14 使用telnet上bbs .....	- 96 -
2.5.15 如何设置网络以访问互联网网址.....	- 98 -
2.5.16 如何设置MAC地址 .....	- 100 -



2.5.17 如何使用Telnet远程登录开发板.....	- 103 -
2.5.18 使用ftp传递文件.....	- 103 -
2.5.19 通过网页控制板上的LED.....	- 104 -
2.5.20 如何挂接使用网络文件系统NFS .....	- 105 -
2.5.21 设置并保存系统实时时钟 .....	- 106 -
2.5.22 如何掉电保存数据到Flash.....	- 106 -
2.5.23 如何设置开机自动运行程序 .....	- 107 -
2.5.24 如何使用命令进行屏幕截图 .....	- 108 -
2.6 预装WindowsCE的功能和外围资源测试 .....	- 109 -
2.6.1 按键测试 .....	- 110 -
2.6.2 LED测试 .....	- 111 -
2.6.3 屏幕旋转测试 .....	- 112 -
2.6.4 串口通信测试 .....	- 112 -
2.6.5 如何使用优盘 .....	- 114 -
2.6.6 如何使用SD/MMC卡 .....	- 115 -
2.6.7 使用Windows Media Player播放mp3.....	- 116 -
2.6.8 如何使用超级播放器流畅播放SD卡中的Mpeg4 电影 .....	- 117 -
2.6.9 以太网测试 .....	- 117 -
2.6.10 通过telnet登录目标板.....	- 118 -
2.6.11 使用ftp向目标版传送文件.....	- 119 -
2.6.12 Web server测试 .....	- 120 -
2.6.13 触摸屏校正保存 .....	- 121 -
2.6.14 使用ActiveSync进行USB同步通讯 .....	- 122 -
2.6.15 无线网卡测试 .....	- 123 -
2.6.16 如何设置实时时钟并保存 .....	- 124 -
2.7 使用H-JTAG快速烧写BIOS到开发板(全部过程鼠标操作) .....	- 125 -
2.7.1 H-JTAG简介.....	- 125 -
2.7.2 安装并设置H-JTAG.....	- 126 -
2.7.3 设置Flash型号并烧写BIOS.....	- 130 -
2.7.4 常见问题 .....	- 136 -
第三章 备份恢复系统及安装更新 .....	- 137 -
3.1 备份和恢复系统 .....	- 137 -
3.1.1 备份系统 .....	- 137 -
3.1.2 使用备份文件恢复系统 .....	- 142 -
3.2 安装Linux系统.....	- 145 -
3.2.1 分区 .....	- 145 -
3.2.2 安装bootloader .....	- 147 -
3.2.3 安装linux内核 .....	- 149 -
3.2.4 安装根文件系统 .....	- 150 -
3.3 安装WinCE系统 .....	- 153 -
3.3.1 分区 .....	- 153 -



3.3.2 安装bootloader .....	- 154 -
3.3.3 安装eboot .....	- 156 -
3.3.4 安装wince内核映象 .....	- 157 -
3.4 下载到内存运行 .....	- 161 -
3.4.1 运行 2440test.....	- 161 -
3.4.2 运行uCOS2 .....	- 166 -
3.4.3 运行Linux.....	- 171 -
3.4.4 运行WinCE .....	- 174 -
第四章 ADS1.2 集成开发环境的使用 .....	- 177 -
4.1 使用ADS创建LED工程.....	- 177 -
4.1.1 建立一个工程 .....	- 177 -
4.1.2 编译和链接工程 .....	- 182 -
4.2 使用H-JTAG进行代码调试.....	- 189 -
4.2.1 为H-JTAG配置AXD DEBUGGER .....	- 189 -
4.2.4 使用H-JTAG在ADS1.2 环境下进行仿真调试 .....	- 192 -
4.3 编译运行烧写 2440test.....	- 192 -
4.3.1 编译和使用H-JTAG调试 2440test .....	- 193 -
4.3.2 通过USB把 2440test下载到运行 .....	- 197 -
4.4.3 把 2440test烧写到Nand Flash运行 .....	- 201 -
4.5 uCos2 的编译和烧写 .....	- 204 -
4.5.1 编译uCOS2 .....	- 204 -
4.5.2 把uCOS2 下载到内存运行 .....	- 206 -
4.5.3 把uCOS2 烧写到Nand Flash运行 .....	- 210 -
4.6 NBOOT的编译和烧写 .....	- 213 -
4.6.1 编译NBOOT.....	- 213 -
4.6.2 把NBOOT烧写到Nand Flash .....	- 216 -
第五章 建立Linux开发环境 .....	- 219 -
5.1 基于Redhat Linux9.0 的开发环境建立.....	- 219 -
5.1.1 完全图解安装Redhat9.0 .....	- 219 -
5.1.2 建立交叉编译环境 .....	- 232 -
5.1.3 配置网络文件系统NFS服务 .....	- 233 -
5.1.4 通过NFS启动系统 .....	- 234 -
5.1.5 配置PC机Linux的ftp服务.....	- 236 -
5.1.6 配置PC机的telnet服务.....	- 237 -
5.1.7 在Redhat中添加新用户 .....	- 238 -
第六章 嵌入式Linux应用开发入门指南 .....	- 240 -
6.1Hello,World! .....	- 240 -
6.1.1 Hello,World源代码.....	- 240 -
6.1.2 编译Hello,World.....	- 240 -
6.1.3 把Hello,World下载到开发板运行 .....	- 240 -
6.2 嵌入式Linux程序开发入门 .....	- 244 -



6.2.1 LED测试程序.....	- 244 -
6.2.2 测试按键 .....	- 245 -
6.2.3 UDP网络编程 .....	- 247 -
6.2.4 数学函数库调用示例 .....	- 252 -
6.2.5 线程编程示例 .....	- 253 -
6.2.6 管道应用编程示例 .....	- 255 -
6.2.7 基于C++的Hello,World .....	- 260 -
6.3 最简单的嵌入式Linux驱动程序模块.....	- 261 -
6.3.1 Hello,Module源代码 .....	- 261 -
6.3.2 把Hello,Module加入内核代码树，并编译 .....	- 262 -
6.3.3 把Hello, Module下载到开发板并安装使用 .....	- 265 -
6.4 简易Linux驱动程序示例.....	- 266 -
6.4.1 LED驱动程序.....	- 266 -
6.4.2 按键驱动程序 .....	- 270 -
6.5 嵌入式Linux程序移植实例.....	- 276 -
6.5.1 mp3 播放器madplay移植过程详解.....	- 276 -
(1)建立工作目录，拷贝源代码包.....	- 277 -
(2)解压源代码包.....	- 277 -
(3)编译madplay所依赖的库文件 .....	- 278 -
(4)编译安装madplay .....	- 279 -
(5)测试PC版的madplay .....	- 282 -
(6)构建编译脚本build-x86 .....	- 282 -
(7)构建并修正ARM版本的编译脚本build-arm .....	- 283 -
(8)下载madplay到开发板运行测试.....	- 287 -
第七章 常见bootloader的配置和编译 .....	- 289 -
7.1 配置和编译vivi .....	- 290 -
7.1.1 使用缺省配置编译 .....	- 290 -
7.1.2 配置vivi从Nor Flash启动 .....	- 293 -
7.2 使用ADS编译YL-BIOS.....	- 294 -
7.2.1 使用ADS编译YL-BIOS.....	- 294 -
7.2.2 把YL-BIOS下载到内存中运行 .....	- 296 -
7.2.3 烧写YL-BIOS到开发板.....	- 299 -
7.3 配置和编译U-Boot .....	- 301 -
7.3.1 配置和编译U-Boot .....	- 302 -
7.3.2 把U-Boot烧写到开发板 .....	- 303 -
第八章 配置和编译linux内核 .....	- 307 -
8.1 使用缺省配置文件编译内核 .....	- 307 -
8.1.1 解压内核源代码 .....	- 307 -
8.1.2 装载缺省配置文件 .....	- 308 -
8.1.3 编译内核 .....	- 309 -
8.1.4 各个Linux驱动程序源代码位置 .....	- 311 -



8.2 定制linux内核 .....	- 312 -
8.2.1 如何配置CPU选项 .....	- 313 -
8.2.2 如何配置各个尺寸的LCD驱动支持 .....	- 315 -
8.2.3 如何配置触摸屏 .....	- 318 -
8.2.4 如何配置USB鼠标和键盘 .....	- 321 -
8.2.5 如何配置优盘的支持 .....	- 324 -
8.2.6 如何配置网眼和中芯微等USB摄像头 .....	- 327 -
8.2.7 如何配置CS8900 网卡驱动 .....	- 330 -
8.2.8 如何配置声卡驱动 .....	- 336 -
8.2.9 如何配置SD/MMC卡驱动 .....	- 340 -
8.2.10 如何配置LED驱动 .....	- 341 -
8.2.11 如何配置按键驱动 .....	- 342 -
8.2.12 如何配置串口驱动 .....	- 343 -
8.2.13 如何配置RTC实时时钟驱动 .....	- 344 -
8.2.14 如何配置yaffs文件系统的支持 .....	- 345 -
8.2.15 如何配置EXT2/VFAT/ NFS等文件系统 .....	- 347 -
8.3 yaffs根文件系统映象的制作 .....	- 352 -
第九章 WinCE开发指南 .....	- 354 -
9.1 基于WinCE5.0 的开发环境 .....	- 354 -
9.1.1 安装Platform Builder 5.0(含 2007 最新补丁) .....	- 354 -
9.1.2 导入安装BSP .....	- 365 -
9.1.3 安装无线网卡驱动程序 .....	- 368 -
9.1.4 编译内核工程示例 .....	- 371 -
9.1.5 导出SDK .....	- 375 -
9.1.6 安装Embedded Visual C++(EVC) .....	- 381 -
9.1.7 安装EVC补丁和导出的SDK .....	- 387 -
9.1.8 定制CE内核 .....	- 396 -
9.2 使用ActiveSync与PC同步通讯(公共) .....	- 410 -
9.2.1 安装ActiveSync .....	- 410 -
9.2.2 为同步通讯安装USB驱动 .....	- 415 -
9.2.3 使用ActiveSync同步传输工具复制文件 .....	- 419 -
9.2.4 使用ActiveSync与Platform Builder连接实现通讯并屏幕截图 .....	- 422 -
9.2.5 使用ActiveSync与Platform Builder在线编辑注册表 .....	- 429 -
9.3 创建EVC的Hello,World, 并编译下载到开发板运行 .....	- 430 -
9.4 创建VS2005/2008 应用程序, 并编译下载到开发板运行 .....	- 437 -
9.4.1 创建项目 .....	- 438 -
9.4.2 设置连接开发板 .....	- 440 -
9.4.3 编译下载程序到开发板运行 .....	- 443 -
9.5 LED驱动程序编写及测试示例 .....	- 444 -
9.5.1 了解硬件连接 .....	- 445 -
9.5.2 编写LED流式驱动程序 .....	- 446 -



9.5.3 把LED驱动程序添加到BSP中以编译.....	452 -
9.5.4 编写并编译LED测试应用程序.....	454 -
9.5.5 把LED测试程序添加到内核，并建立桌面快捷方式.....	457 -
附录 1 Qt嵌入式图形开发入门 .....	459 -
1. 设置开发环境 .....	459 -
2. 编译X86 平台的Qtopia和Hello,World和嵌入式浏览器 .....	459 -
2.1 编译Qt/Embedded .....	460 -
2.2 在PC上模拟运行Qtopia.....	460 -
2.3 编译Hello, World示例.....	460 -
2.4 单独运行Hello, World.....	461 -
2.5 在Qtopia中运行Hello, World.....	461 -
3 编译ARM平台的Qtopia和Hello,World和嵌入式浏览器.....	462 -
3.1 编译Qt/Embedded .....	462 -
3.2 编译Hello, World示例.....	463 -
3.3 把hello,world下载到目标板并运行 .....	463 -
3.4 使用自己编译的Qtopia更新制作文件系统.....	467 -
4 常见问题 .....	467 -
4.1 执行build时出现的错误 .....	468 -
4.2 编译hello时出现的错误 .....	469 -
4.3 编译hello时出现的第二种错误信息.....	469 -
附录 2 使用BIOS的命令行更新和烧写系统 .....	470 -
1.1. 如何进入BIOS的命令行模式 .....	470 -
1.1.1 从功能菜单进入命令行模式 .....	470 -
1.1.2 在Nand Flash启动时进入命令行模式 .....	471 -
2.2 安装linux .....	471 -
2.2.1 对Nand Flash进行分区 .....	472 -
2.2.2 恢复BIOS .....	473 -
2.2.3 烧写linux内核 .....	475 -
2.2.4 烧写基于yaffs的根文件系统.....	477 -
2.2.5 启动系统 .....	479 -
3.3 安装wince.....	479 -
3.3.1 对Nand Flash进行分区 .....	480 -
3.3.2 恢复BIOS .....	481 -
3.3.3 烧写Eboot.....	482 -
3.3.4 烧写wince内核 .....	483 -
附录 3 使用SJF2440 烧写BIOS .....	487 -
1 安装GIVEIO驱动.....	487 -
2 使用SJF2440 烧写BIOS .....	493 -

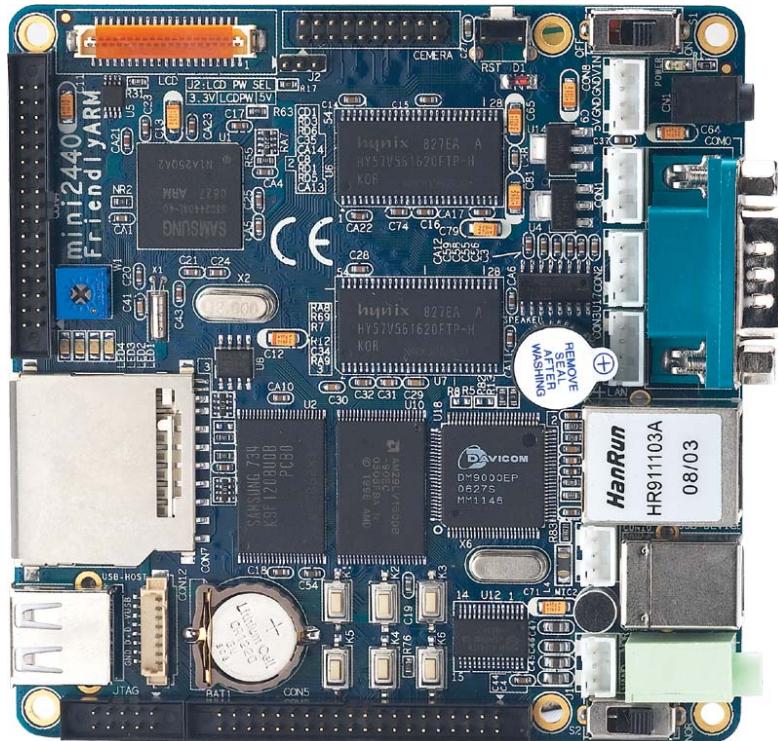
# 第一章 MINI2440 开发板介绍

## 1.1 MINI2440 开发板简介

mini2440 是一款低价实用的 ARM9 开发板，是目前国内性价比最高的一款学习板；它采用 Samsung S3C2440 为微处理器，并采用专业稳定的 CPU 内核电源芯片和复位芯片来保证系统运行时的稳定性。mini2440 的 PCB 采用沉金工艺的四层板设计，专业等长布线，保证关键信号线的信号完整性，生产采用机器贴片，批量生产；出厂时都经过严格的质量控制，配合这本十分详细的手册，可以迅速帮你掌握嵌入式 Linux 和 WinCE 开发的流程，只要有 C 语言基础的人一般 2 周即可入门。

用户可以到我们网站浏览最新通知及下载更新最新的手册和系统网址：  
<http://www.arm9.net>

### 1.1.1 MINI2440 开发板外观





追求卓越 创造精品

TO BE BEST

TO DO GREAT

广州友善之臂计算机科技有限公司

## 1.1.2 MINI2440 开发板硬件资源特性

### ● CPU 处理器

- Samsung S3C2440A, 主频 400MHz, 最高 533Mhz

### ● SDRAM 内存

- 在板 64M SDRAM
- 32bit 数据总线
- SDRAM 时钟频率高达 100MHz

### ● FLASH 存储

- 在板 64M Nand Flash, 掉电非易失
- 在板 2M Nor Flash, 掉电非易失, 已经安装 BIOS

### ● LCD 显示

- 板上集成 4 线电阻式触摸屏接口, 可以直接连接四线电阻触摸屏
- 支持黑白、4 级灰度、16 级灰度、256 色、4096 色 STN 液晶屏, 尺寸从 3.5 寸到 12.1 寸, 屏幕分辨率可以达到 1024x768 像素;
- 支持黑白、4 级灰度、16 级灰度、256 色、64K 色、真彩色 TFT 液晶屏, 尺寸从 3.5 寸到 12.1 寸, 屏幕分辨率可以达到 1024x768 像素;
- 标准配置为 NEC 256K 色 240x320/3.5 英寸 TFT 真彩液晶屏, 带触摸屏;
- 板上引出一个 12V 电源接口, 可以为大尺寸 TFT 液晶的 12V CCFL 背光模块(Inverting)供电。

### ● 接口和资源

- 1 个 100M 以太网 RJ-45 接口(采用 DM9000 网络芯片)
- 3 个串行口
- 1 个 USB Host
- 1 个 USB Slave B 型接口
- 1 个 SD 卡存储接口
- 1 路立体声音频输出接口, 一路麦克风接口;
- 1 个 2.0mm 间距 10 针 JTAG 接口
- 4 USER Leds
- 6 USER buttons(带引出座)
- 1 个 PWM 控制蜂鸣器
- 1 个可调电阻, 用于 AD 模数转换测试
- 1 个 I2C 总线 AT24C08 芯片, 用于 I2C 总线测试
- 1 个 2.0 mm 间距 20pin 摄像头接口
- 板载实时时钟电池
- 电源接口(5V), 带电源开关和指示灯

### ● 系统时钟源

- 12M 无源晶振

### ● 实时时钟

- 内部实时时钟 (带后备锂电池)



追求卓越 创造精品

TO BE BEST

TO DO GREAT

广州友善之臂计算机科技有限公司

- 扩展接口
  - 1 个 34 pin 2.0mmGPIO 接口
  - 1 个 40 pin 2.0mm 系统总线接口
- 规格尺寸
  - 100 x 100(mm)
- 操作系统支持
  - Linux2.6.13
  - WindowsCE.NET 5.0

### 1.1.3 用户光盘资源说明

- (1)ADS1.2 安装程序
- (2)H-JTAG 烧写调试软件
- (3)Windows 下烧写 Flash 的软件 SJF2440
- (4)Linux 下烧写 Nand Flash 的软件 Jflash-2440(含源代码)
- (5)串口工具 CRT, dnw
- (6)图片转 C 语言数组工具
- (7)USB 驱动(WindowXP/2000 下安装使用)
- (8)vivi 源代码, 用于 linux 的 bootloaer
- (9)最简单的测试程序(包含 ADS1.20 的项目文件), 用于点亮板上的 LED 灯。
- (10)2440test 测试程序(包含 ADS1.20 的项目文件, 全部源代码), 测试项目包括: 中断方式按键测试, RTC 实时时钟测试, ADC 数模转换测试, IIS 音频播放 wav 测试, IIS 音频录音测试, 触摸屏测试, I2C 总线读写 AT24C08 测试, 三星 3.5”LCD、640x480 真彩液晶测试等
- (11)WindowsCE BSP 和示例项目文件
- (12)linux 开发工具和内核源代码包:
  - arm-linux-gcc-3.3.2 编译 Qtopia 使用
  - arm-linux-gcc-3.4.1 编译内核使用
  - arm-linux-gcc-2.95.3 编译 vivi 用
  - yaffs 文件系统映象制作工具 mkyaffsimage
  - linux-2.6.13 for MINI2440 内核源代码(包含 DM9000 驱动、各种真彩液晶驱动、声卡驱动、触摸屏驱动、YAFFS 源代码、SD 卡驱动、RTC 驱动、扩展串口驱动、各种 USB 摄像头驱动、USB 鼠标和键盘、优盘驱动等)
- (13)嵌入式图形界面 Qtopia 源代码包, 嵌入式浏览器源代码包
- (14)开发板原理图(Protel99SE 格式/PDF 格式)
- (15)用户手册(pdf 格式)

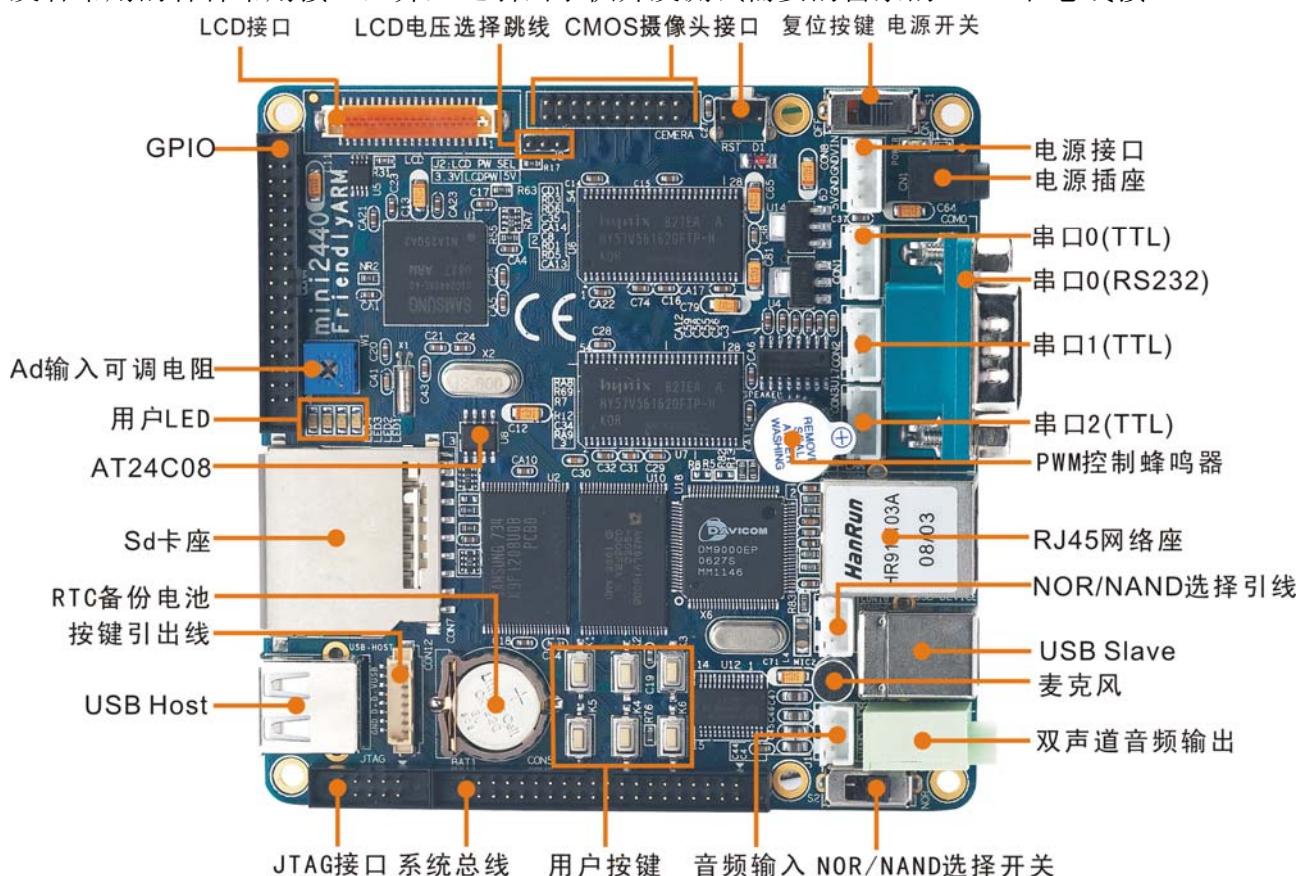
### 1.2 接口布局及跳线

## 1.2.1 跳线说明

开发板上只有一个跳线 J2，它用于选择 LCD 驱动板的输入电压，在标准配置中，所接为 NEC3.5 寸 LCD，电压选择为 5V。

## 1.2.2 接口布局

Mini2440 接口布局如下图所示，它在十分紧凑的 100 x 100mm 面积上精致安排了开发者常用的各种常用接口，并且还引出了供开发测试需要的富余的 IO 口和总线接口：

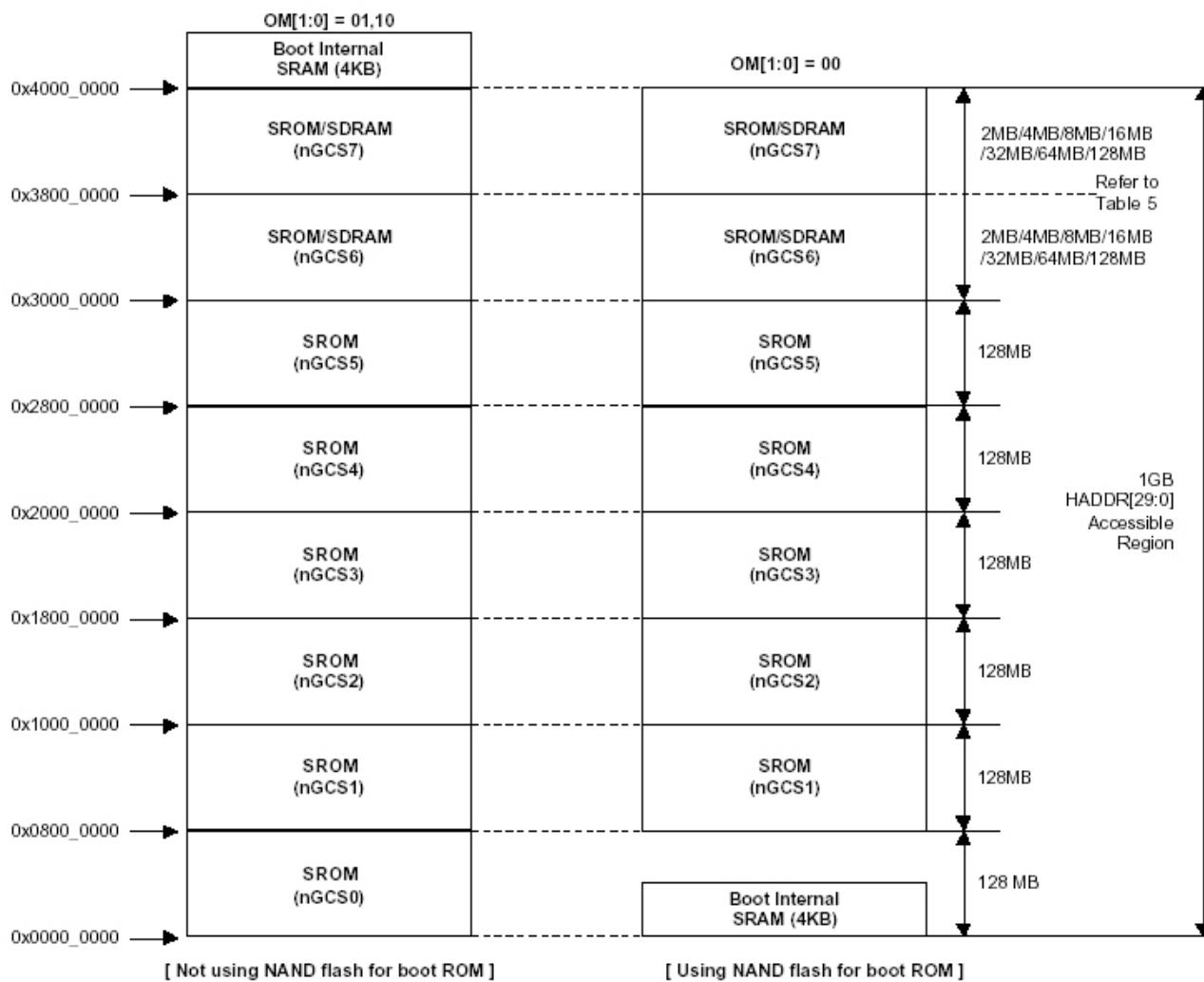


## 1.3 接口资源说明

本小节详细介绍了开发板上每个接口或模块的引脚定义和占用的 CPU 资源，光盘中还有本开发板的完整原理图和封装库(分为 pdf 格式和 Protel99SE 格式)，以供开发板参考使用。

### 1.3.1 地址空间分配和片选信号定义

S3C2440 支持两种启动模式：一种是从 Nand Flash 启动(MINI2440 即是此种)；一种是从 Nor Flash 启动。在此两种启动模式下，各个片选的存储空间分配是不同的，如下图：



上图中，

左边是 nGCS0 片选的 Nor Flash 启动模式下的存储分配图；

右边是 Nand Flash 启动模式下的存储分配图；

说明：SFR Area 为特殊寄存器地址控制

下面是器件地址空间分配和其片选定义

在进行器件地址说明之前，有一点需要注意，nGCS0 片选的空间在不同的启动模式下，映射的器件是不一样的。由上图可以知道：

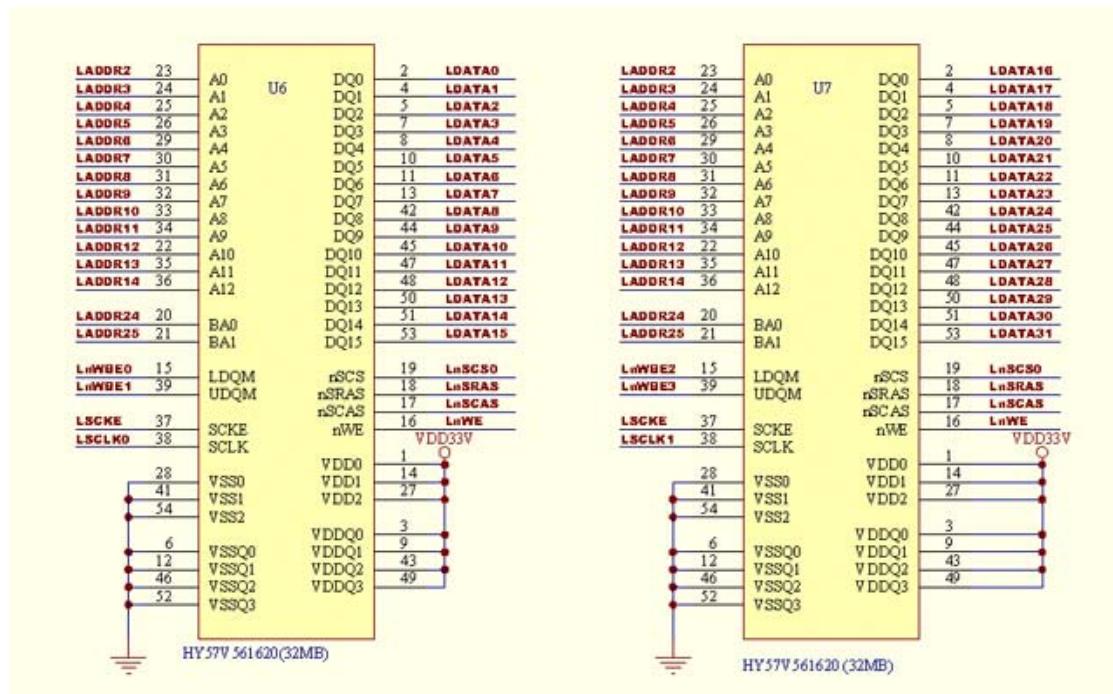
- 在 NAND Flash 启动模式下，内部的 4K Bytes BootSram 被映射到 nGCS0 片选的空间；
- 在 Nor Flash 启动模式下(非 Nand Flash 启动模式)，与 nGCS0 相连的外部存储器

Nor Flash 就被映射到 nGCS0 片选的空间

SDRAM 地址空间: 0x30000000 ~ 0x34000000

### 1.3.2 SDRAM 存储系统

Mini2440 使用了两片外接的 32M bytes 总共 64M bytes 的 SDRAM 芯片(型号为: HY57V561620FTP), 一般称之为内存, 它们并接在一起形成 32-bit 的总线数据宽度, 这样可以增加访问的速度; 因为是并接, 故它们都使用了 nGCS6 作为片选, 根据 CPU 手册 5-2 中的介绍可知, 这就决定了它们的物理起始地址为 0x30000000, 下面是摘自 mini2440 原理图中的 SDRAM 部分原理图。



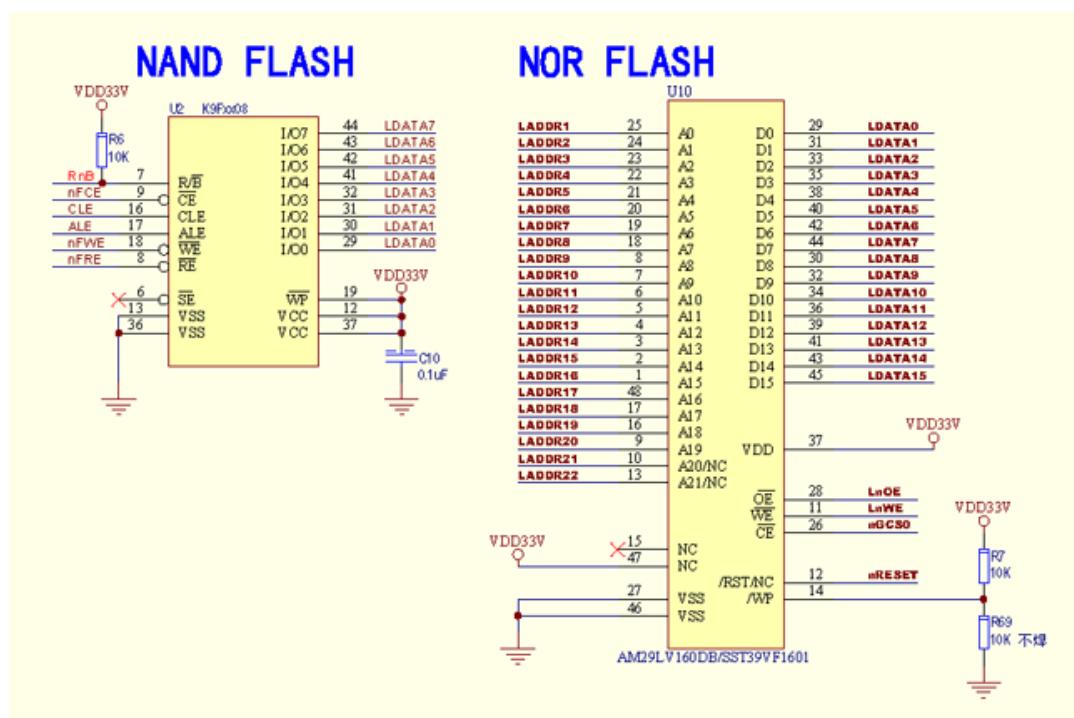
### 1.3.3 FLASH 存储系统

Mini2440 具备两种 Flash, 一种是 Nor Flash, 型号为 SST39VF1601, 大小为 2Mbyte; 另一种是 Nand Flash, 型号为 K9F1208, 大小为 64Mbyte。S3C2440 支持这两种 Flash 启动系统, 通过拨动开关 S2, 你可以选择从 Nor 还是从 Nand 启动系统。实际的产品中大都使用一片 Nand Flash 就够了, 因为我们为了方便用户开发学习, 因此还保留了 Nor Flash。

Nand Flash 不具有地址线, 它有专门的控制接口与 CPU 相连, 数据总线为 8-bit, 但这并不意味着 Nand Flash 读写数据会很慢。大部分的优盘或者 SD 卡等都是 Nand Flash 制成的设备。

从下面的原理图可以看出, Nor Flash 采用了 A1-A22 总共 22 条地址总线和 16 条数据总线与 CPU 连接, 请注意地址是从 A1 开始的, 这意味着它每次最小的读写单位是 2-byte,

因此根据原理图，该设计总共可以兼容支持最大 8Mbyte 的 Nor Flash，实际我们的开发板上只用了 A1-A20 条地址线，因为与 A21、A22 相连的 SST39V1601 的相应引脚是悬空的。

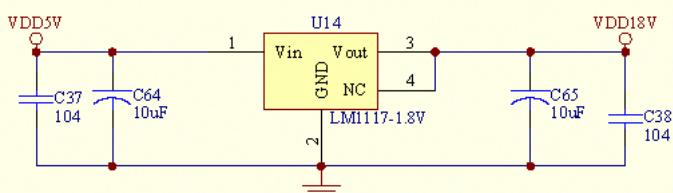


### 1.3.4 电源系统及接口

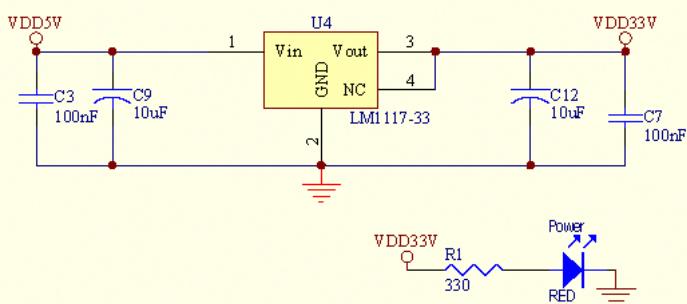
本开发板的电源系统比较简单，直接使用外接的 5V 电源，通过降压芯片产生整个系统所需要的三种电压：3.3V、1.8V、1.25V。

请注意，本开发板并非面向手持移动设备设计，因此它并不具备完善的电源管理电路。整个系统的电源通断是由 S1 拨动开关控制的，它不能通过软件实现开关机。

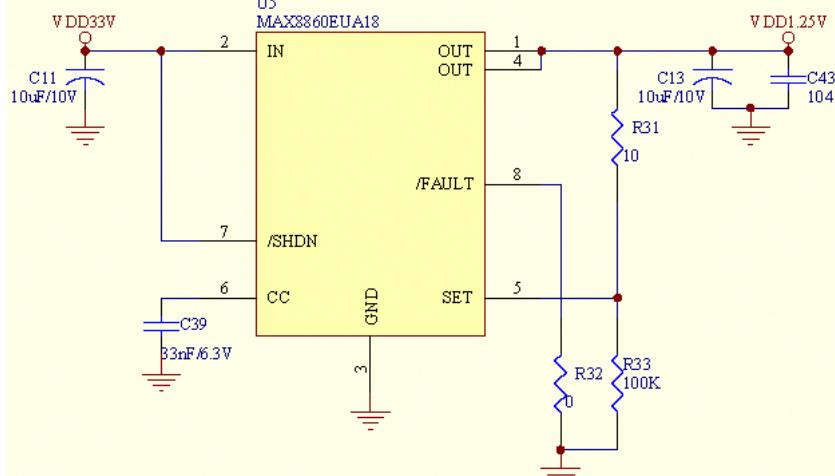
## 1. 1.8V电源产生电路(实测可能有偏差)



## 3.3V电源产生电路(实测可能有偏差)

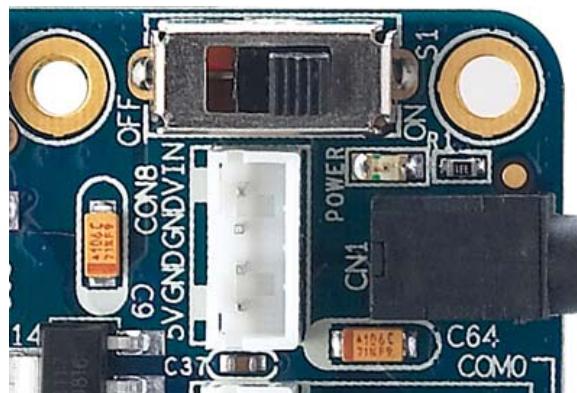
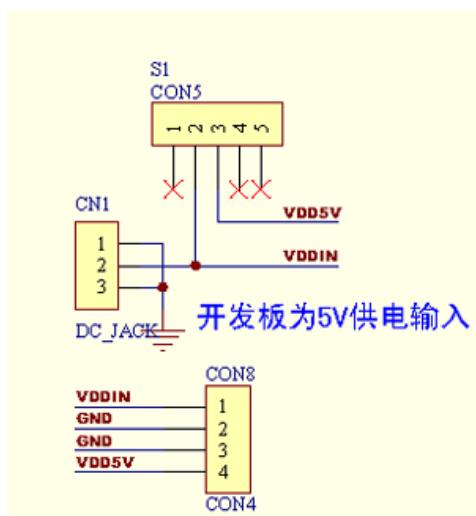


## 1. 2.5V电源产生电路(实测可能有偏差)



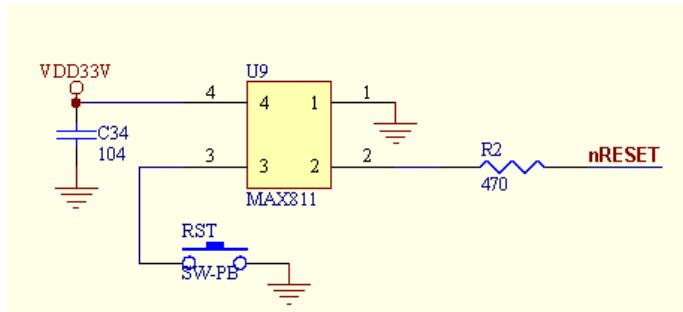
为了方便用户外接其他电源，我们还设计了一个电源接口 CON8，它是一个白色 2.0mm 间距的单排插座，中间均为“地”，两侧均为 5V。注意，这两个 5V 并非是相通的，其中一个连接了外部电源的 5V，另外一个则连接了经过拨动开关 S1 之后的 5V。

它们的连接关系和相应的实物标称见下图：



### 1.3.5 复位系统

本开发板采用专业的复位芯片 MAX811 实现 CPU 所需要的低电平复位，见下图：



### 1.3.6 用户 LED

LED 是开发中最常用的状态指示设备，本开发板具有 4 个用户可编程 LED，它们直接与 CPU 的 GPIO 相连接，低电平有效(点亮)，详细的资源占用如下表：

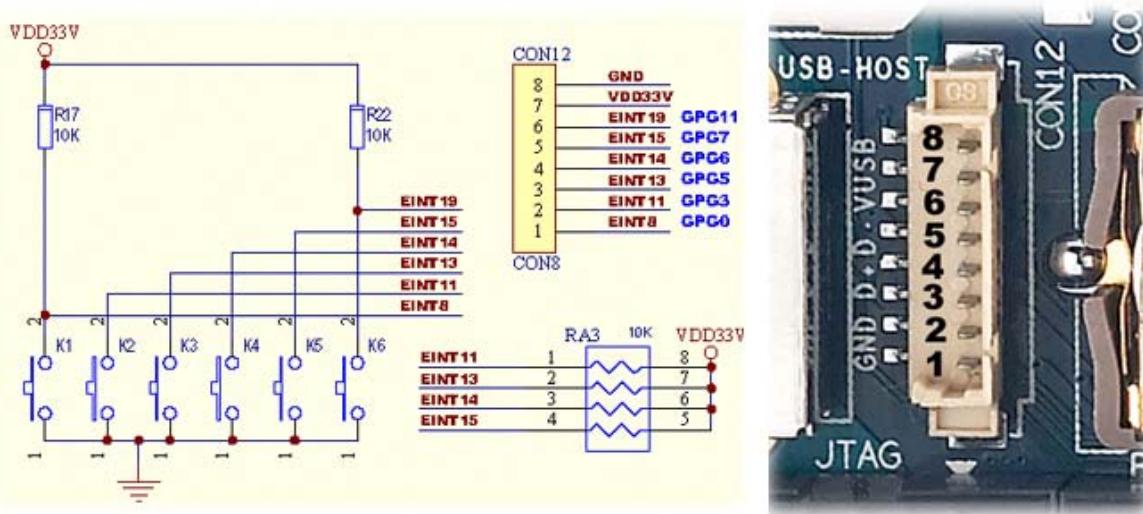
	<b>LED1</b>	<b>LED2</b>	<b>LED3</b>	<b>LED4</b>
<b>GPIO</b>	GPB5	GPB6	GPB7	GPB8
可复用为	nXBACK	nXREQ	nXDACK1	nDREQ1
在原理图中的网络名	nLED_1	nLED_2	nLED_3	nLED_4

### 1.3.7 用户按键

本开发板总共有 6 个用户测试用按键，它们均从 CPU 中断引脚直接引出，属于低电平触发，这些引脚也可以复用为 GPIO 和特殊功能口，为了用户把它们引出作为其他用途，这 6 个引脚也通过 CON12 引出，6 个按键和 CON12 的定义如下：

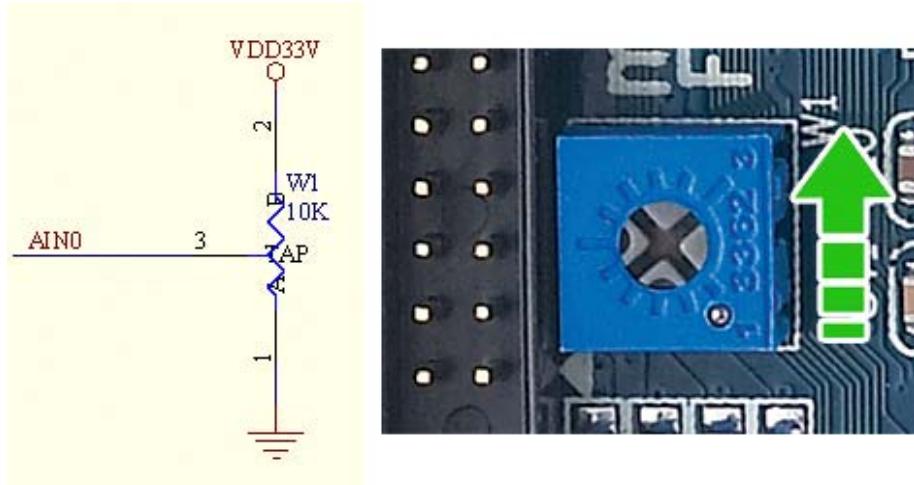
	K1	K2	K4	K4	K5	K6
对应的中断	EINT8	EINT11	EINT13	EINT14	EINT15	EINT19
复用的 GPIO	PGP0	PGP3	PGP5	PGP6	PGP7	PGP11
特殊功能口	无	nSS1	SPIMISO1	SPIMOSI1	SPICLK1	TCLK1
对应的 CON12 引脚	CON12.1	CON12.2	CON12.3	CON12.4	CON12.5	CON12.6

说明：CON12.7 为电源(3.3V)，CON12.8 为地(GND)



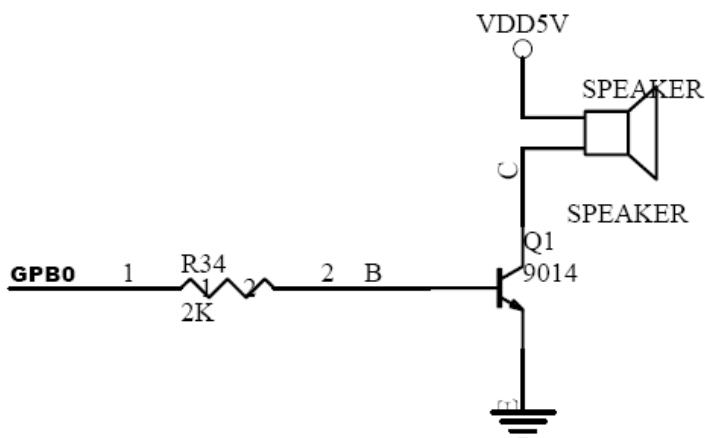
### 1.3.8 A/D 输入测试

本开发板总共可以引出 4 路 A/D(模数转换)转换通道，它们位于板上的 CON4-GPIO 接口(详见 GPIO 接口介绍)，为了方便测试，AIN0 连接到了开发板上的可调电阻 W1，原理图如下所示。



### 1.3.9 PWM 控制蜂鸣器

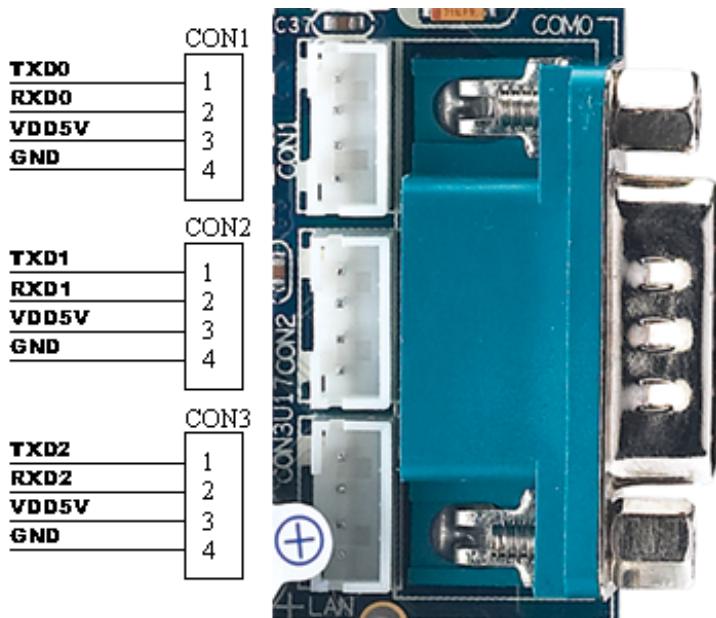
开发板的蜂鸣器 SPEAKER 是通过 PWM 控制的，原理图如下所示，其中 GPB0 可通过软件设置为 PWM 输出。



### 1.3.10 串口

S3C2440 本身总共有 3 个串口 UART0、1、2，其中 UART0,1 可组合为一个全功能的串口，在大部分的应用中，我们只用到 3 个简单的串口功能(本开发板提供的 Linux 和 WinCE 驱动也是这样设置的)，即通常所说的发送(TXD)和接收(RXD)，它们分别对应板上的 CON1、CON2、CON3，这 3 个接口都是从 CPU 直接引出的，是 TTL 电平。为了方便用户使用，其中 UART0 做了 RS232 电平转换，它们对应于 COM0，可以通过附带的直连线与 PC 机互相通讯。

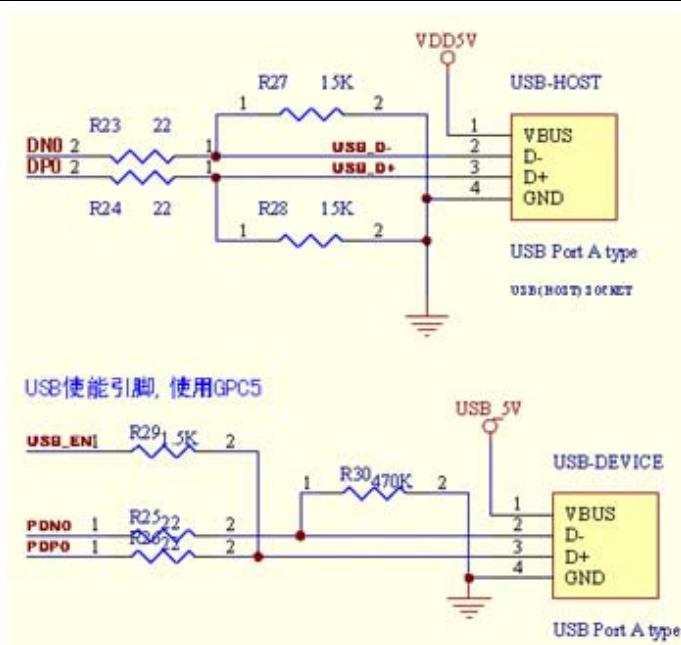
CON1, CON2, CON3 在开发板上的位置和原理图中的连接定义对应关系如下图所示。



### 1. 3. 11 USB 接口

本开发板具有两种 USB 接口，一个是 USB Host，它和普通 PC 的 USB 接口是一样的，可以接 USB 摄像头、USB 键盘、USB 鼠标、优盘等常见的 USB 外设，另外一种是 USB Slave，我们一般使用它来下载程序到目标板，当开发板装载了 WinCE 系统时，它可以通过 ActiveSync 软件和 Windows 系统进行同步，当开发板装载了 Linux 系统时，目前尚无相应的驱动和应用。为了方便用户通过程序控制 USB Slave 和 PC 的通断，我们设置了 USB\_EN 信号，如图，它使用的 CPU 资源为 GPC5。

我们将提更加广泛的 USB Host 外设应用，请经常留意我们网站的更新信息。

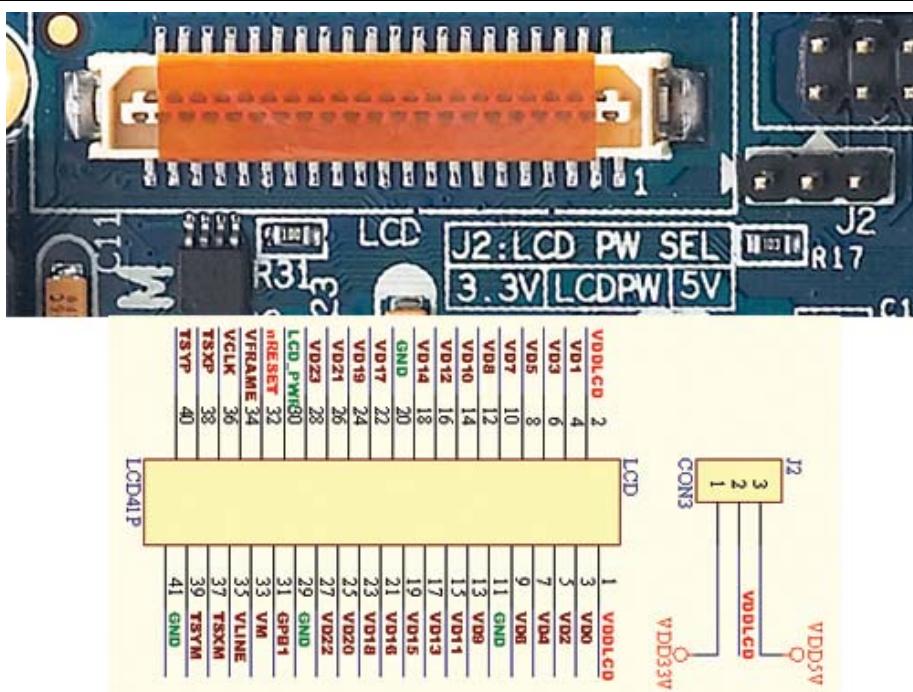


### 1. 3. 12 LCD 接口

本开发板的 LCD 接口是一个 41Pin 0.5mm 间距的白色座, 其中包含了常见 LCD 所用的大部分控制信号(行场扫描、时钟和使能等), 和完整的 RGB 数据信号(RGB 输出为 8: 8: 8, 即最高可支持 1600 万色的 LCD); 为了用户方便试验, 还引出了 PWM 输出(GPB1 可通过寄存器配置为 PWM), 和复位信号(nRESET), 其中 LCD\_PWR 是背光控制信号。

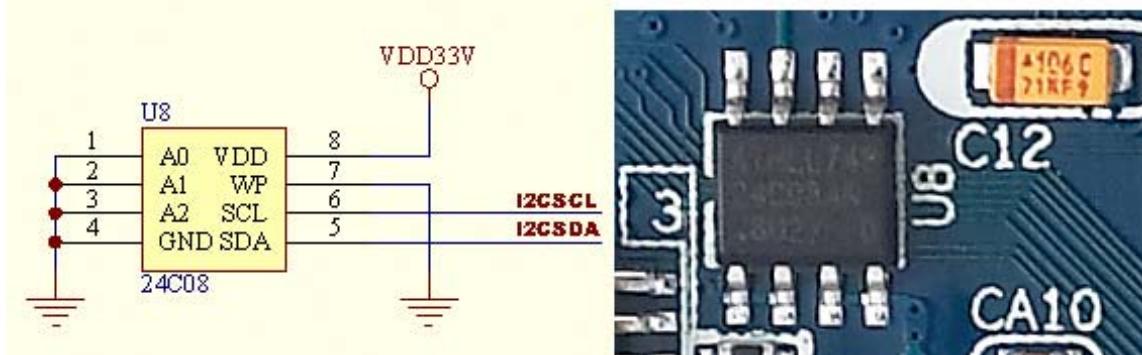
另外, 37、38、39、40 为四线触摸屏接口, 它们可以直接连接触摸屏使用。

图中的 J2 为 LCD 驱动板供电选择信号, 目前我们的驱动板都使用 5V 供电。



### 1.3.13 EEPROM

本开发板具有一个直接连接 CPU 之 I2C 信号引脚的 EEPROM 芯片 AT24C08，它的容量有 256 byte，在此主要是为了供用户测试 I2C 总线而用，它并没有存储特定的参数。

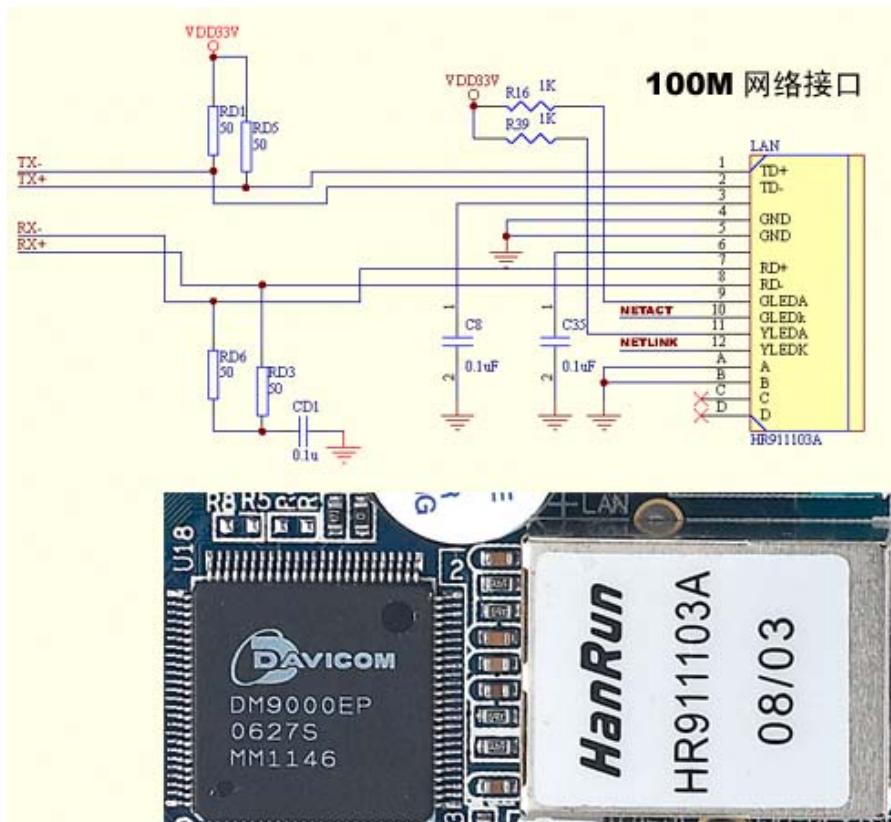


### 1.3.14 网络接口

本开发板采用了 DM9000 网卡芯片，它可以自适应 10/100M 网络，RJ45 连接头内部已经包含了耦合线圈，因此不必另接网络变压器，使用普通的网线即可连接本开发板至你的路由器或者交换机。

注意：每个开发板的网络 MAC 地址都是相同的，它可以通过软件设定，对于 Linux

用户，本手册 2.4 章节有相关介绍；对于 WinCE 用户，您可以参考 BSP 里面的 DM9000 驱动代码和注册表文件(platform.reg)。

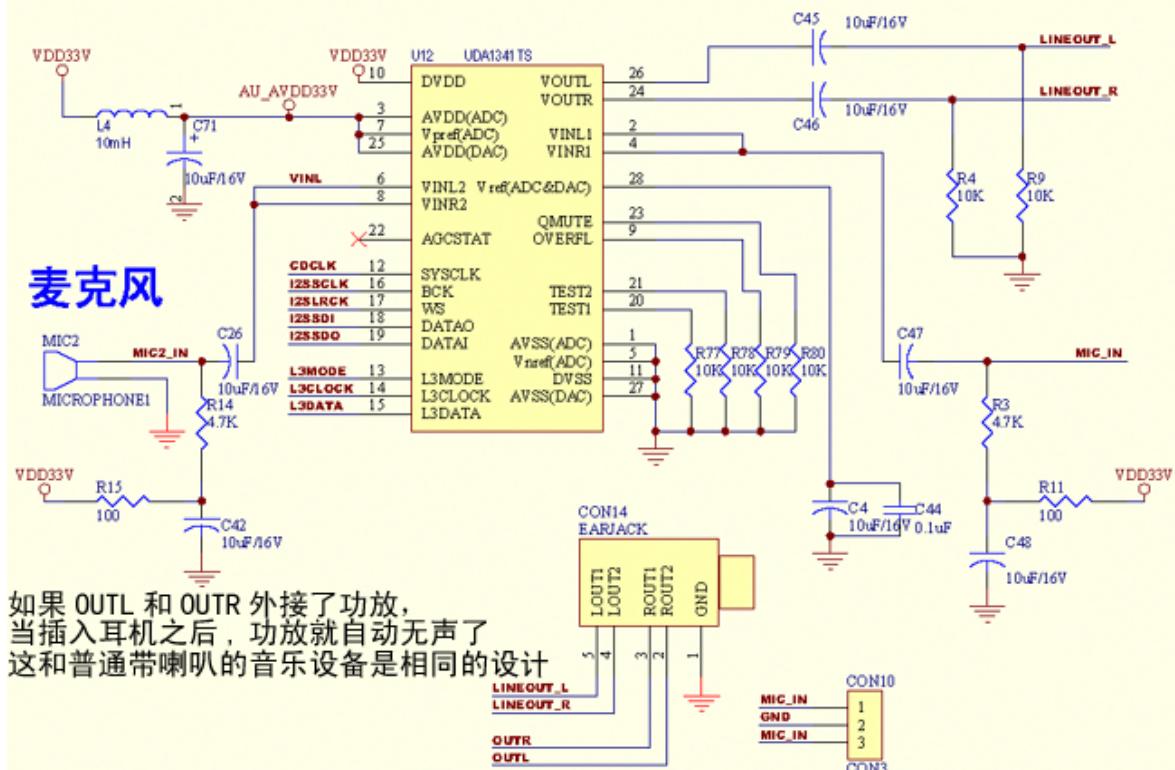


### 1.3.15 音频接口

S3C2440 内置 I2S 总线接口，可直接外接 8/16 比特的立体声 CODEC，本开发板采用基于 I2S 总线的 UDA1341 芯片实现音频解码系统，该芯片内部寄存器的初始化和设置则是采用 L3-bus 总线连接控制实现的，在这里我们沿用了三星公板的设计，分别使用 CPU 的 GPB2、GPB3、GPB4 端口模拟实现 L3-Bus 规范的 L3MODE、L3DATA、L3CLOCK，它们在初始化完 UDA1341 以后就不再有用了，因此这三条控制线也可以使用普通的单片机模拟实现。

音频系统的输出为开发板上的常用 3.5mm 孔径插座，输入分为两路，一路为板载麦克风，另一路通过 CON10 白色 2.0mm 插座引出。两路音频输入的驱动是不同的，目前只有 CON10 接口对应的通道是可以录音使用的，请留意我们网站的更新信息，这两个通道我们最终都会驱动起来。

## 音频输入与输出电路



### 1.3.16 JTAG 接口

当开发板从贴片厂下线，里面是没有任何程序的，这时我们一般通过 JTAG 接口烧写第一个程序，就是 Supervivi，借助 Supervivi 可以使用 USB 口下载更加复杂的系统程序等，这在后面的章节中你可以看到。

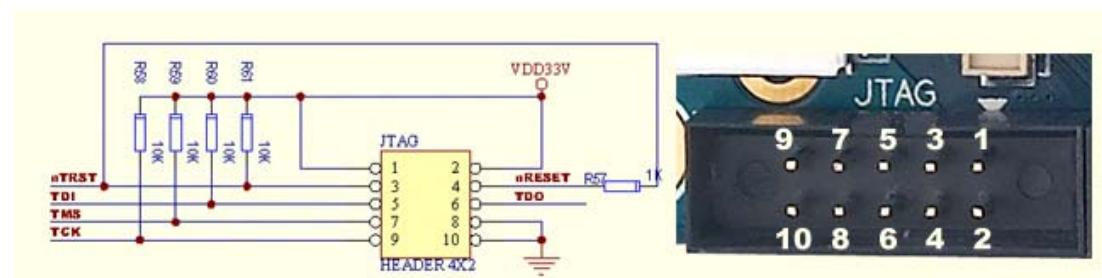
除此之外，JTAG 接口在开发中最常见的用途是单步调试，不管是市面上常见的 JLINK 还是 ULINK，以及其他仿真调试器，最终都是通过 JTAG 接口连接的。标准的 JTAG 接口是 4 线：TMS、TCK、TDI、TDO，分别为模式选择、时钟、数据输入和数据输出线，加上电源和地，一般总共 6 条线就够了；为了方便调试，大部分仿真器还提供了一个复位信号。

因此，标准的 JTAG 接口是指是否具有上面所说的 JTAG 信号线，并不是 20Pin 或者 10Pin 等这些形式上的定义表现。这就如同 USB 接口，可以是方的，也可以扁的，还可以是其他形式的，只要这些接口中包含了完整的 JTAG 信号线，都可以称为标准的 JTAG 接口。本开发板提供了包含完整 JTAG 标准信号的 10 Pin JTAG 接口，各引脚定义如图。

说明：对于打算致力于 Linux 或者 WinCE 开发的初学者而言，JTAG 接口基本是没有任何意义和用途的，因为大部分开发板都已经提供了完善的 BSP，这包括最常用的串口和网络以及 USB 通讯口，当系统装载了可以运行的 Linux 或者 WinCE 系统，用户完全可以通过这些高级操作系统本身所具备的功能进行各种调试，这时是不需要 JTAG 接口的；即使你可以进行跟踪，但鉴于操作系统本身结构复杂，接口繁多，单步调试犹如大海捞针，毫无意义。

可言。想一想你手头使用的 PC 机就知道了，或许你从没有见过甚至听过有谁会在 PC 主板上插一个仿真器，来调试 PCI 这样接口的 WindowsXP 或者 Linux 驱动。这就是为什么你经常见到或者听到那么多人在讲驱动“移植”，因为大部分人都参考前辈的实现来做驱动的。

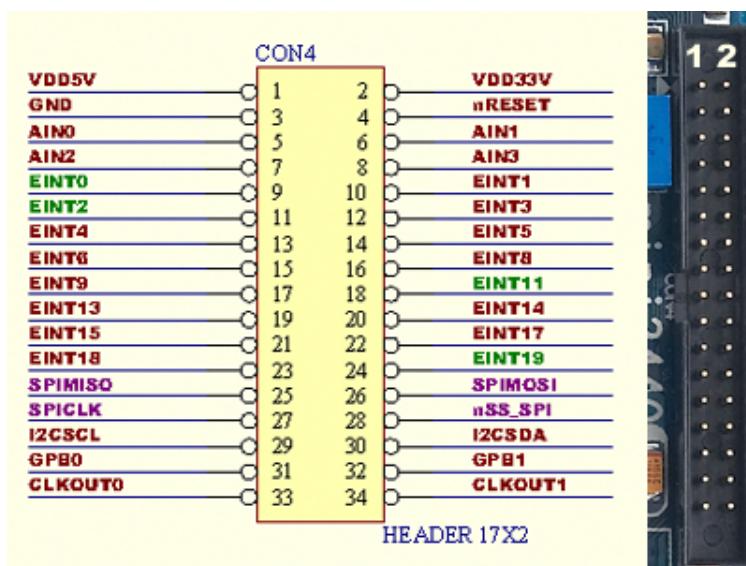
JTAG 仅对那些不打算采用操作系统，或者采用简易操作系统(例如 uCos2 等)的用户有用。大部分开发板所提供的 Bootloader 或者 BIOS 已经是一个基本完好的系统了，因此也不需要单步调试。



### 1. 3. 17 GPIO

GPIO 是通用输入输出口的简称，本开发板带有一个 34 Pin 2.0mm 间距的 GPIO 接口，标称为 CON4，如图。

实际上，CON4 不仅包含了很多富余的 GPIO 引脚，还包含了一些其他 CPU 引脚，如 AD0-AIN3, CLKOUT 等。你所看到的图中的 SPI 接口、I2C 接口、GPB0 和 GPB1 等，它们其实也是 GPIO，不过是以特殊功能接口来标称定义的，这些都可以通过相应的 CPU 寄存器来设置更改它们的用途，详细的接口资源见下表。



CON4	网络名称	说明(有些端口可复用)	CON4	网络名称	说明(有些端口可复用)
1	VDD5V	5V 电源(输入或者输出)	2	VDD33V	3.3V 电源(输出)
3	GND	地	4	nRESET	复位信号(输出)
5	AIN0	AD 输入通道 0	6	AIN1	AD 输入通道 1

<b>7</b>	AIN2	AD 输入通道 2	<b>8</b>	AIN3	AD 输入通道 3
<b>9</b>	EINT0	EINT0/GPF0	<b>10</b>	EINT1	EINT1/GPF1
<b>11</b>	EINT2	EINT2/GPF2	<b>12</b>	EINT3	EINT3/GPF3
<b>13</b>	EINT4	EINT4/GPF4	<b>14</b>	EINT5	EINT5/GPF5
<b>15</b>	EINT6	EINT6/GPF6	<b>16</b>	EINT8	EINT8/GPG0
<b>17</b>	EINT9	EINT9/GPG1	<b>18</b>	EINT11	EINT11/GPG3/nSS1
<b>19</b>	EINT13	EINT13/GPG5/SPIMISO1	<b>20</b>	EINT14	EINT14/GPG6/SPIMOSI1
<b>21</b>	EINT15	EINT15/GPG7/SPICLK1	<b>22</b>	EINT17	EINT17/GPG9/nRST1
<b>23</b>	EINT18	EINT18/GPG10/nCTS1	<b>24</b>	EINT19	EINT19/GPG11
<b>25</b>	SPIMISO	SPIMISO /GPE11	<b>26</b>	SPIMOSI	SPIMOSI /EINT14/GPG6
<b>27</b>	SPICLK	SPICLK /GPE13	<b>28</b>	nSS_SPI	nSS_SPI /EINT10/GPG2
<b>29</b>	I2CSCL	I2CSCL/GPE14	<b>30</b>	I2CSDA	I2CSDA/GPE15
<b>31</b>	GPB0	TOUT0/ GPB0	<b>32</b>	GPB1	TOUT1/ GPB1
<b>33</b>	CLKOUT0	CLKOUT0/GPH9	<b>34</b>	CLKOUT1	CLKOUT1/GPH10

### 1. 3. 18 CMOS CAMERA 接口

S3C2440 带有 CMOS 摄像头接口，在开发板上通过标称为 CAMERA 的接口引出。它是一个 20 脚 2.0mm 间距的针座，用户可以直接使用我们提供的 CAM130 摄像头模块；其实 CAM130 摄像头模块上面没有任何电路，它只是一个转接板，它直接连接使用了型号为 ZT130G2 摄像头模块，它们的定义如下图所示。

说明：CAMERA 接口是一个复用端口，它可以通过设置相应的寄存器改为 GPIO 使用，下表是它对应引脚的 GPIO 列表

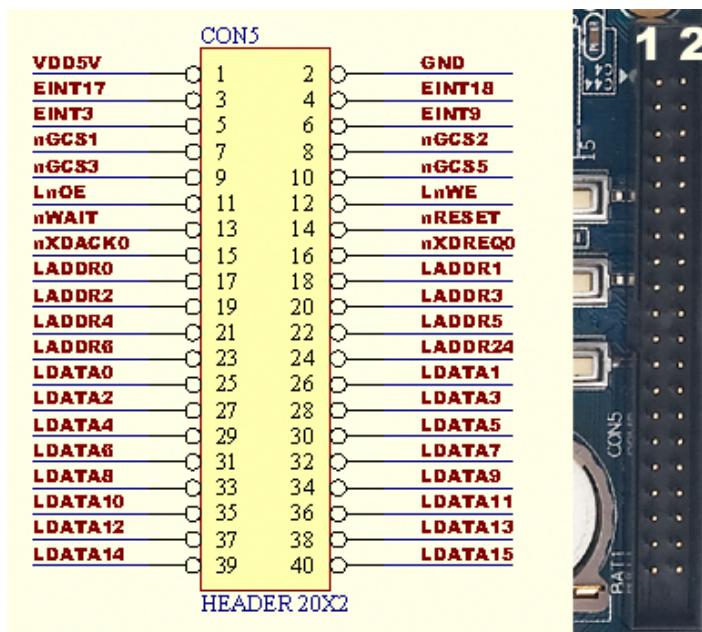


CAMERA	网络名称	可复用为	CAMERA	网络名称	可复用为
<b>1</b>	I2CSDA	GPE15	<b>2</b>	I2CSCL	GPE14
<b>3</b>	EINT20	PGP12	<b>4</b>	CAMRST	GPJ12

<b>5</b>	CAMCLK	GPJ11	<b>6</b>	CAM_HREF	GPJ10
<b>7</b>	CAM_VSYNC	GPJ9	<b>8</b>	CAM_PCLK	GPJ8
<b>9</b>	CAMDATA7	GPJ7	<b>10</b>	CAMDATA6	GPJ6
<b>11</b>	CAMDATA5	GPJ5	<b>12</b>	CAMDATA4	GPJ4
<b>13</b>	CAMDATA3	GPJ3	<b>14</b>	CAMDATA2	GPJ2
<b>15</b>	CAMDATA1	GPJ1	<b>16</b>	CAMDATA0	GPJ0
<b>17</b>	VDD33V	3.3V 电源	<b>18</b>	VDD_CAM	VDD_CAM
<b>19</b>	VDD18V	1.8V 电源	<b>20</b>	GND	地

### 1.3.19 系统总线接口

本开发板上的系统总线接口为 CON5，它总共包含 16 条数据线(D0-D15)、8 条地址线(A0-A6, A24)、还有一些控制信号线(片选、读写、复位等)，CON5 可以向外提供 5V 电压输出；实际上，很少有用户通过总线扩展外设。下面是 CON5 的详细引脚定义说明。



CON5	网络名称	说明(有些端口可复用)	CON5	网络名称	说明(有些端口可复用)
<b>1</b>	VDD5V	5V 电源(输入或者输出)	<b>2</b>	GND	地
<b>3</b>	EINT17	中断 17(输入)	<b>4</b>	EINT18	中断 18(输入)
<b>5</b>	EINT3	中断 3(输入)	<b>6</b>	EINT9	中断 9(输入)
<b>7</b>	nGCS1	片选 1 对应物理地址: 0x08000000	<b>8</b>	nGCS2	片选 2 对应物理地址: 0x10000000
<b>9</b>	nGCS3	片选 3 对应物理地址: 0x18000000	<b>10</b>	nGCS5	片选 2 对应物理地址: 0x28000000
<b>11</b>	LnOE	读使能信号	<b>12</b>	LnWE	写使能



追求卓越 创造精品

TO BE BEST

TO DO GREAT

广州友善之臂计算机科技有限公司

13	nWAIT	等待信号	14	nRESET	复位
15	nXDACK0	nXDACK0	16	nXDREQ0	nXDREQ0
17	LADDR0	地址 0	18	LADDR1	地址 1
19	LADDR2	地址 2	20	LADDR3	地址 3
21	LADDR4	地址 4	22	LADDR5	地址 5
23	LADDR6	地址 6	24	LADDR24	地址 24
25	LDATA0	数据线 0	26	DATA1	数据线 1
27	LDATA2	数据线 2	28	DATA3	数据线 3
29	LDATA4	数据线 4	30	DATA5	数据线 5
31	LDATA6	数据线 6	32	DATA7	数据线 7
33	LDATA8	数据线 8	34	DATA9	数据线 9
35	LDATA10	数据线 10	36	DATA11	数据线 11
37	LDATA12	数据线 12	38	DATA13	数据线 13
39	LDATA14	数据线 14	40	DATA15	数据线 15

## 1.3 linux 特性

- 版本
  - Linux2.6.13
- 支持的文件系统
  - yaffs(可读写的文件系统, 推荐使用)
  - cramfs(压缩的只读文件系统, 不在线更新数据时推荐使用)
  - Ext2
  - Fat32
  - NFS(网络文件系统, 开发驱动程序及应用程序时方便使用)
- 基本驱动程序(以下驱动均以源代码方式提供)
  - 3 串口标准驱动
  - DM9000 驱动程序
  - 声音驱动
  - RTC 驱动(可掉电保存时间)
  - 用户 LED 灯驱动
  - USB Host 驱动
  - 常见液晶驱动
  - 触摸屏驱动
  - USB 摄像头, 支持网眼、中芯微芯片的摄像头
  - USB 鼠标、USB 键盘驱动、优盘、移动硬盘
  - SD 卡驱动, 最大可支持 2G
- Linux 应用及服务程序
  - busybox1.2.0(Linux 工具集, 包含常用 Linux 命令等)



追求卓越 创造精品

TO BE BEST

TO DO GREAT

广州友善之臂计算机科技有限公司

- Telnet、Ftp、inetd(网络远程登录工具及服务)
- boa(web server)
- madplay(基于控制台的mp3播放器)
- snapshot(基于控制台的抓图软件)
- ishow(基于控制台的图片浏览软件)
- ifconfig、ping、route等(常用网络工具命令)
- 嵌入式图形系统(以源代码方式提供)
  - Qt/Embedded

## 1.4 WindowsCE 特性

- 版本
  - WindowsCE.net 5.0
- 特性
  - DM9000 网卡驱动源代码
  - USB 键盘、USB 鼠标驱动、优盘、移动硬盘等
  - 三个串口驱动
  - USB ActiveSync
  - 声音驱动
  - SD 卡驱动
  - 实时时钟
  - 注册表保存
  - Flash 剩余空间掉电保存数据
  - 屏幕可旋转
- 缺省系统特性(简体中文系统)
  - XP 界面风格
  - Windows Media Player 9.0(支持 mp3, mpeg2, mpeg4, wmv, wav 等)
  - 超级播放器(类似 windows 下的暴风影音)
  - 图片浏览器、写字板
  - IE6 浏览器
  - ftp、telnet、httpd 服务器
  - 串口助手



追求卓越 创造精品

TO BE BEST

TO DO GREAT

广州友善之臂计算机科技有限公司

## 第二章 MINI2440 开发板使用说明

出厂之前，如果客户未加说明，我们一般已经烧写缺省的 linux 系统(包含三个文件，对应的光盘二进制文件是 **supervivi**、**zImage\_n35**、**root\_qtopia\_tp.img**)，请注意以下的操作是基于 Windows 环境的。

### 2.1 开发板设置及连接

#### 2.1.1 启动模式选择

本开发板的启动模式选择，是通过拨动开关 S2 来决定的：

**根据目标板提示：**

**S2 接到 Nor Flash 标识一侧时，系统将从 Nor Flash 启动；**

**S2 接到 Nand Flash 标识一侧时，系统将从 Nand Flash 启动。**

出厂的时候开发板的 Nor Flash 和 Nand Flash 已经烧入了相同的 BIOS(因为该 BIOS 同时支持这两种 Flash，只是开机后表现形式不同，请参考“开发板 BIOS 功能及使用说明”一节)，S2 已经被接到 Nand Flash 一侧，系统一开机就从 Nand Flash 启动运行系统。

#### 2.1.2 外部接口连接

- 请使用我们提供的直连串口线连接 MINI2440 的串口 0 和 PC 机的串口
- 用我们提供的交叉网线将 MINI2440 的网络接口与 PC 相连
- 用我们提供的 5V 电源适配器连接到板上的 5V 输入插座
- 把音箱或者耳机的插头接入板上的音频输出口(绿色)
- 如果您有液晶屏，请按照数据线头的方向与 MINI2440 的 LCD 接口相连
- 用 USB 电缆连接 MINI2440 和 PC

#### 2.1.3 设置超级终端

为了通过串口连接MINI2440，必须使用一个模拟终端程序，几乎所有的类似软件都可以使用，其中MS-Windows 自带的超级终端是最常用的选择，当你安装Windows9x 时需要自定义选择安装该项，Windows2000 及更高版本则已经缺省安装。

一般桌面版Linux系统也自带了类似的串口终端软件，叫minicom，它是基于命令行的程序，使用比较复杂一些，感兴趣的用户可以在网上找一下这方面的介绍。



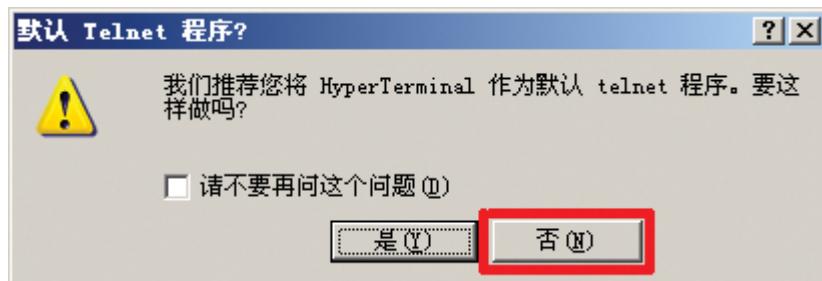
追求卓越 创造精品

TO BE BEST

TO DO GREAT

广州友善之臂计算机科技有限公司

在此着重介绍一下Windows自带的超级终端程序并以WindowsXP为例，或许其他Windows版本的程序界面有所不同。超级终端程序通常位于“开始->程序->附件->通讯”中，选择运行该程序，一般会跳出如图所示窗口，询问你是否要将Hypertrm作为默认的telnet程序，此时你不需要，因此点“否”按钮。



接下来，会跳出如下窗口，点“取消”



此时系统提示“确认取消”，点“是”即可，接着点提示窗口的“确定”，进入下一步。





追求卓越 创造精品

TO BE BEST

TO DO GREAT

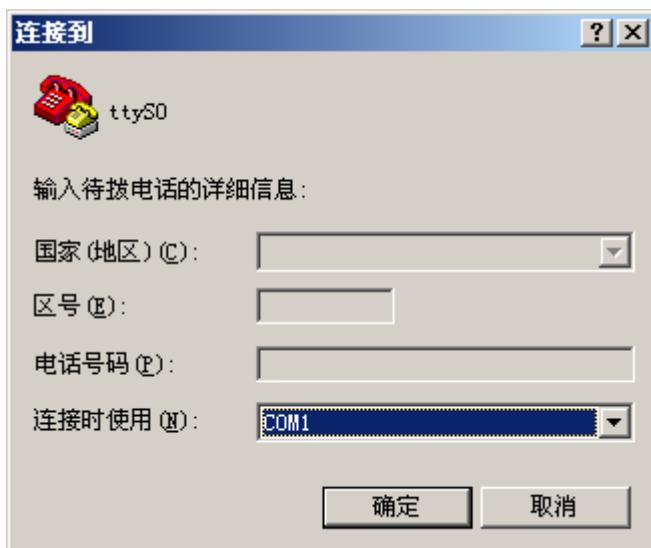
广州友善之臂计算机科技有限公司



超级终端会要求你为新的连接取一个名字，如图所示，这里我取了”ttyS0”，Windows 系统会禁止你取类似”COM1”这样的名字，因为这个名字被系统占用了。



当你命名完以后，又会跳出一个对话框，你需要选择连接 MINI2440 的串口，我这里选择了串口 1，如图所示：



最后，最重要的一步是设置串口，注意必须选择无流控制，否则，或许你只能看到输出而不能输入，另外板子工作时的串口波特率是 **115200**，如图所示。

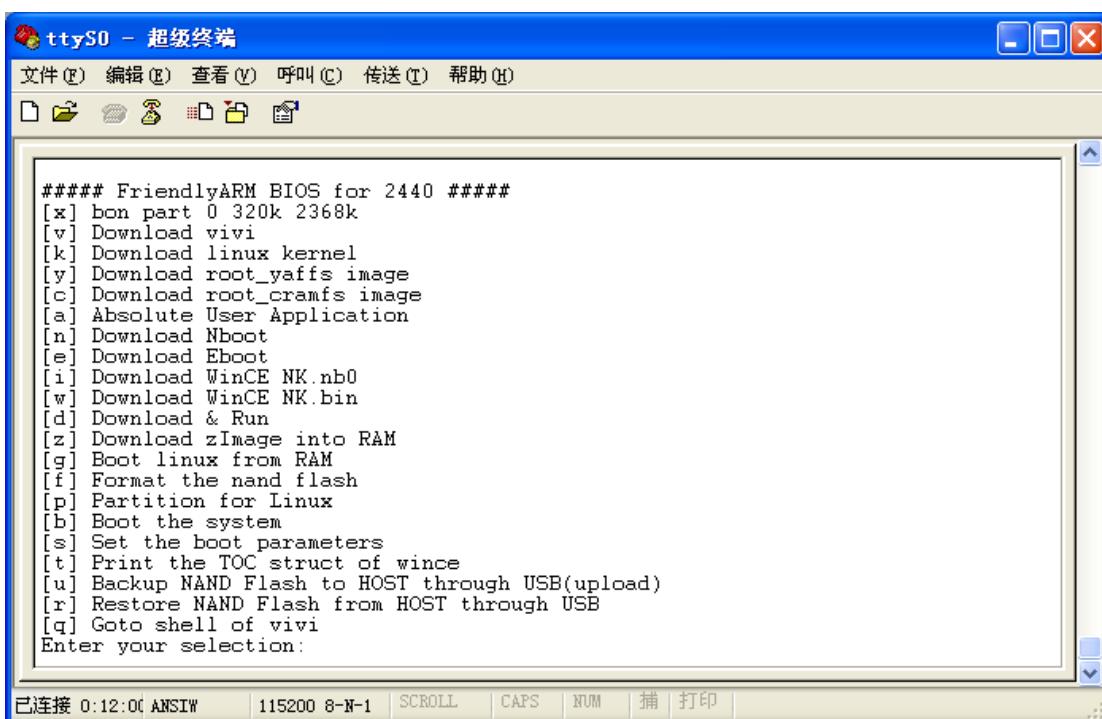


当所有的连接参数都设置好以后，打开电源开关，系统会出现 vivi 启动界面。选择超级终端“文件”菜单下的“另存为...”，保存该连接设置，以便于以后再连接时就不必重新执行以上设置了。

## 2.2 开发板 BIOS 功能及使用说明

### 2.3.1 开机进入 BIOS 模式

Supervivi 在出厂的时候已经预装入板子的 Nor Flash 中，设置拨动开关 S2 为 Nor Flash 启动，即可进入 BIOS 模式，此时开发板上的绿色 LED1 会呈现闪烁状态，其启动界面如下图：



#### Supervivi 简介：

开发板采用的 BIOS 是基于三星原来的 bootloader 之 vivi 改进而来，名为 **Supervivi**，它采用功能菜单的方式，并可以和原来的命令交互模式互相切换。

Supervivi 可以使用 JTAG 板直接烧写入 Nor Flash 中使用，也可以直接烧入 Nand Flash 中运行。当烧入 Nor Flash 并从中时，将会出现菜单模式；当烧入 Nand Flash 并从中运行时，则为命令交互模式（提示：需要在超级终端界面下按住空格键才能进入，否则直接启动系统）。

Supervivi 的菜单模式主要为烧写系统和调试而用，也可以设置参数和进行分区等，它采用 USB 下载的方式，因此搭建烧写环境极为简单，并且下载速度快，使用十分方便。

如果 Supervivi 被烧写入 Nor Flash(默认)，您不仅可以用它来方便的下载更新 linux 和 WinCE 系统，还可以烧写其他任何支持 Nand Flash 启动的操作系统和非操作系统到 Nand Flash，如 uCos2，U-boot，Nboot，2440test 等，然后再选择系统从 Nand Flash 启动，这样您就可以使用各种各样的系统了，我们将会逐步增加这方面的 Demo 文件，请留意我们的网站信息。

如果 Supervivi 被烧写入 Nand Flash，它可以自动识别您烧写的 Linux 或者 WindowsCE



追求卓越 创造精品

TO BE BEST

TO DO GREAT

广州友善之臂计算机科技有限公司

系统、或者其他系统，并自动启动它们。在本手册的“安装和更新系统”一节，我们就直接使用它来作为 bootloader。

另外，使用 Download & Run 功能，您还可以把程序下载到内存马上运行，这对于开发调试是极有帮助的，这样，您甚至不使用仿真器都可以了，我们光盘中的 2440test 程序就是这样一个例子。

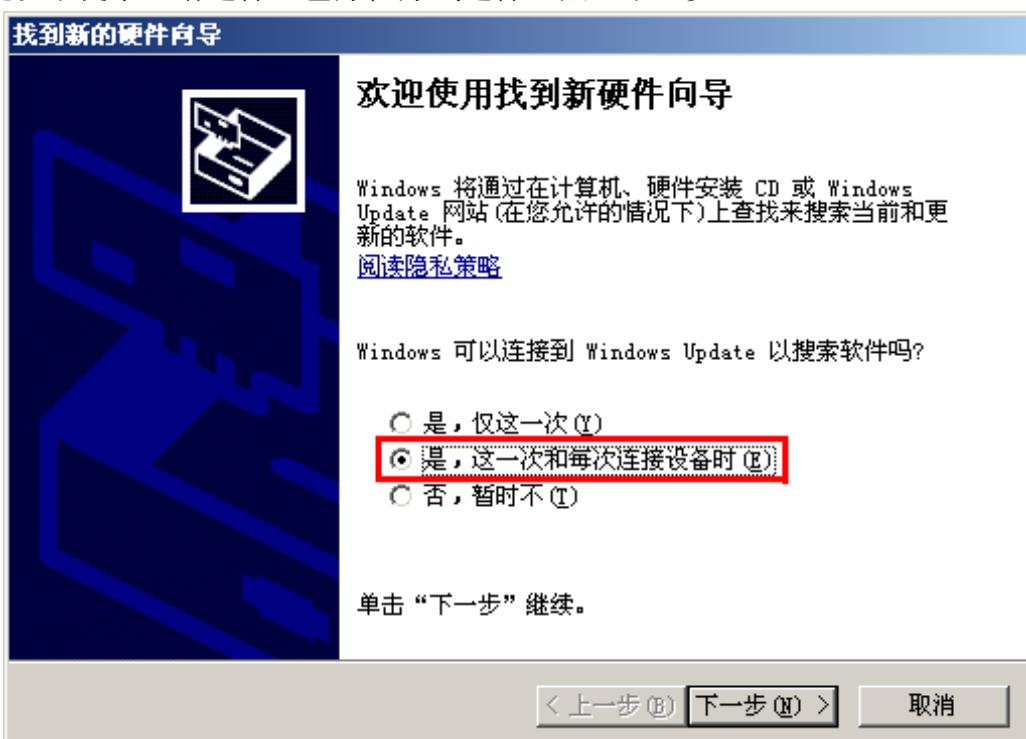
使用 supervivi 还可以把 linux 内核文件 zImage 直接下载到内存中运行，如果您在 supervivi 中设定好网络启动参数，则还可以通过网络启动整个系统；同样的，suerpvivi 也可以把 WinCE 的运行时映像文件 NK.nb0 下载在内存中运行。

## 2.2.2 安装 USB 驱动

**首先设置开发板的拨动开关 S2 为 Nor Flash 启动，连接好附带的 USB 线和电源(可以不必连接串口线)。**

打开电源开关 S1，如果您是第一次使用，WindowsXP 系统会提示您发现了新的 USB 设备，按照以下步骤安装好 USB 驱动：

(1) 出现以下提示，请选择红色方框方式选择，点“下一步”



(2) 出现以下提示，选择“从列表或指定位置安装...”，点“下一步”

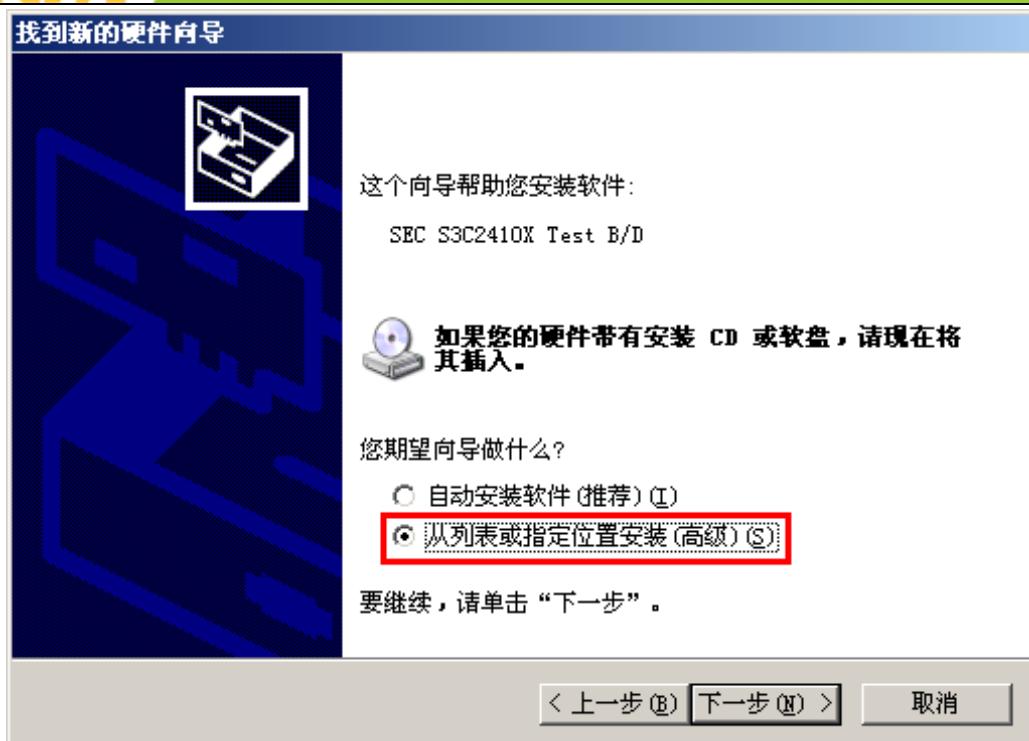


追求卓越 创造精品

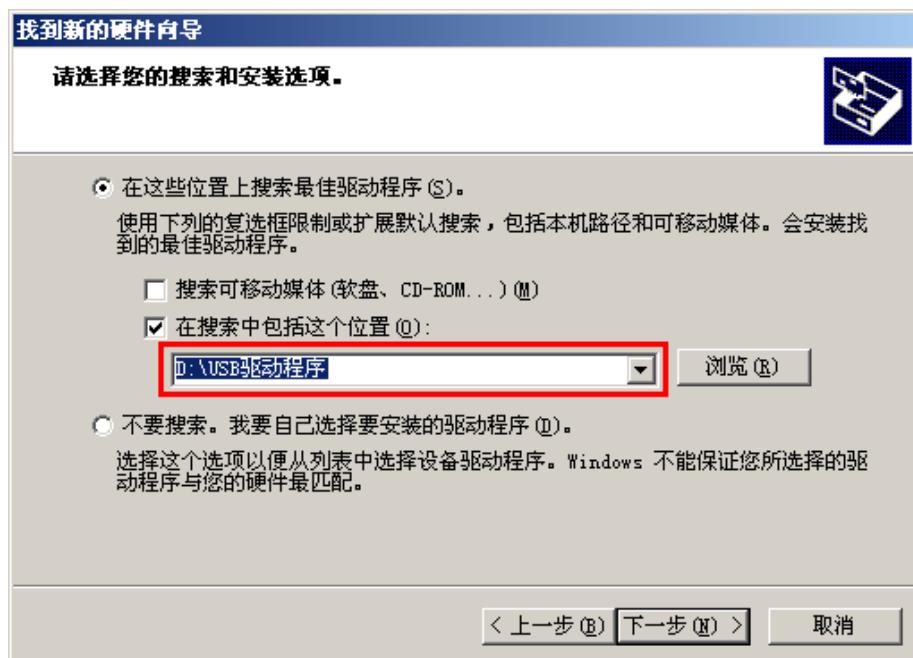
TO BE BEST

TO DO GREAT

广州友善之臂计算机科技有限公司



(3) 选择开发板提供的光盘中 USB 驱动所在的位置，点“下一步”



(4) 开始安装 USB 驱动，如图所示



追求卓越 创造精品

TO BE BEST

TO DO GREAT

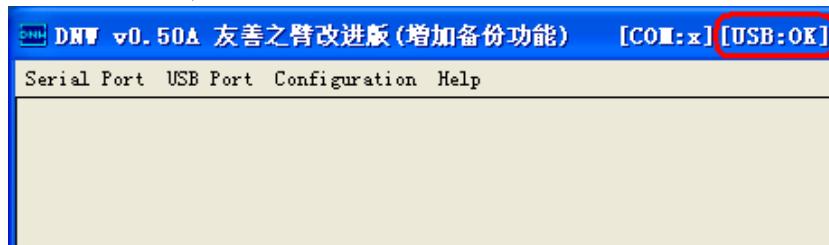
广州友善之臂计算机科技有限公司



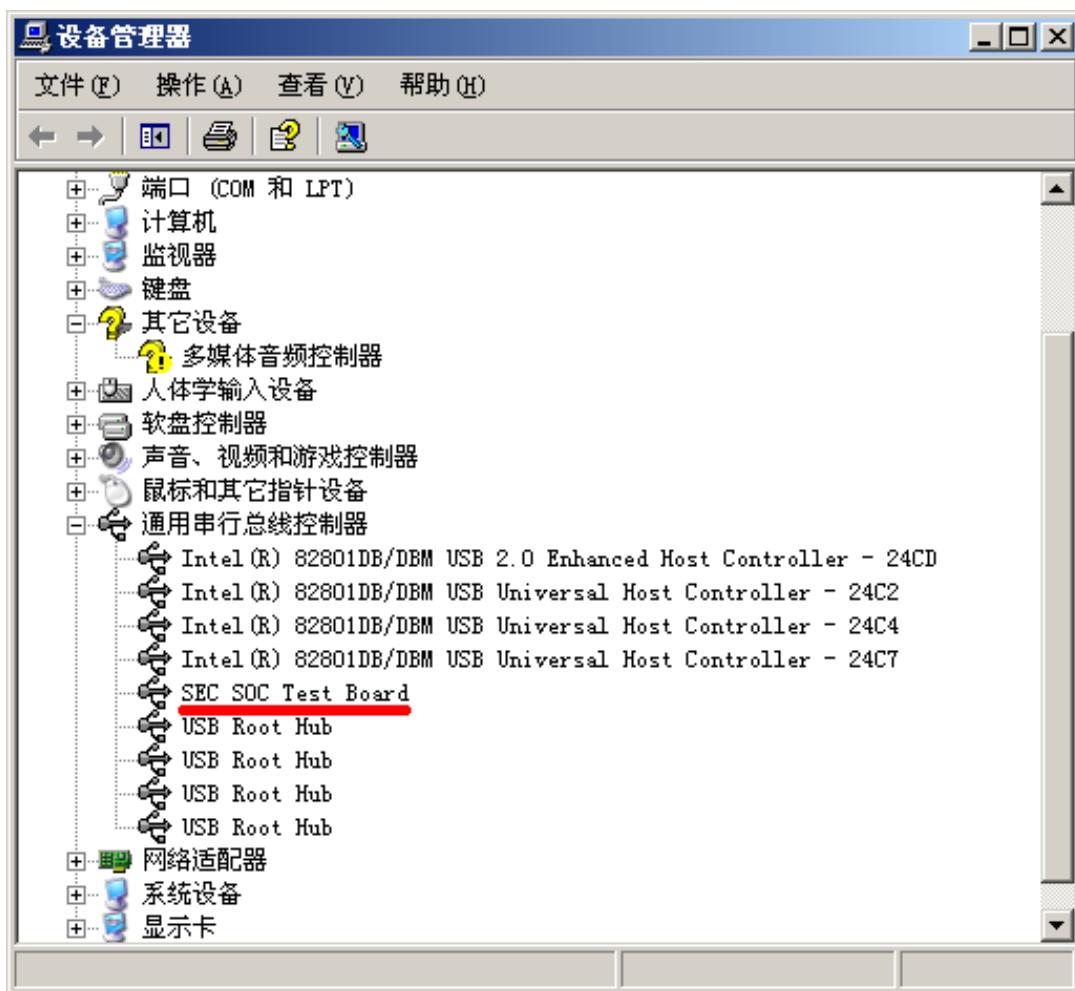
(5) USB 驱动安装成功，出现以下提示。



(6) 打开运行光盘“Windows 平台工具\dnw\”目录里的 dnw.exe 程序(建议把该程序复制到您的电脑中，该程序无需安装)，此时观察 DNW 标题栏，可以看到 USB 连接 OK，如图



(7) 安装好 USB 驱动后，可以在电脑的硬件设备列表中多了如下一项。



### 2.3.3 功能主菜单说明

注意：以下通过 USB 下载的功能均配合 DNW 这个程序使用。

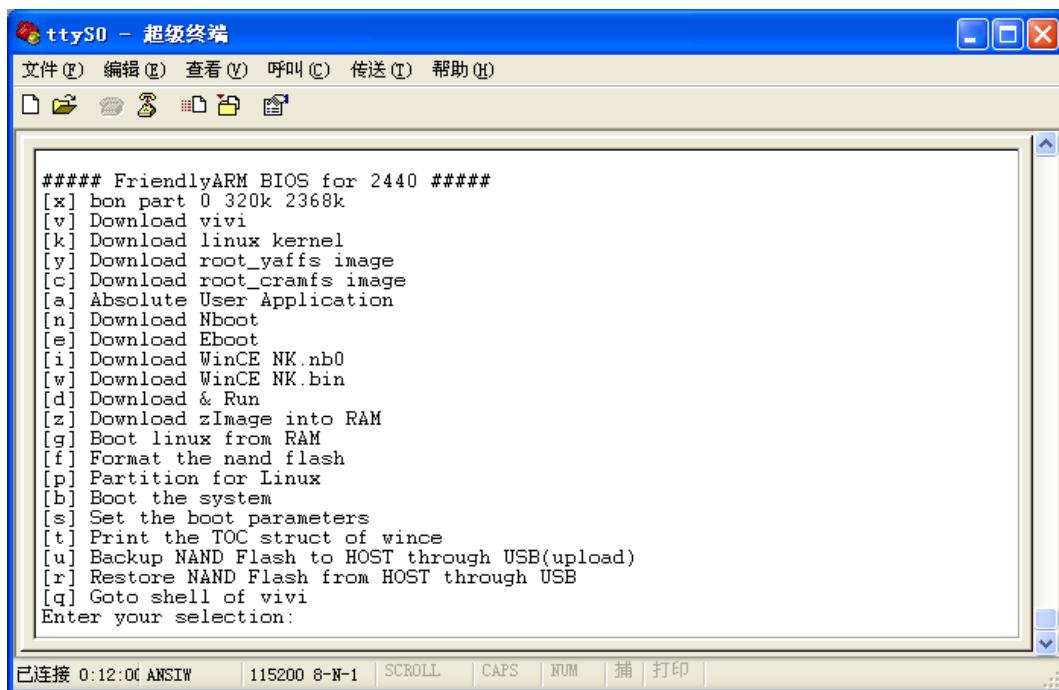


追求卓越 创造精品

TO BE BEST

TO DO GREAT

广州友善之臂计算机科技有限公司



功能[x]: 对 Nand Flash 进行默认分区, 相当于执行命令行的 bon part 0 320k 2368k

功能[v]: 通过 USB 下载 linux bootloader 之 vivi 到 Nand Flash 的 vivi 分区

功能[k]: 通过 USB 下载 linux 内核到 Nand Flash 的 kernel 分区

功能[y]: 通过 USB 下载 yaffs 文件系统映象到 Nand Flash 的 root 分区

功能[c]: 通过 USB 下载 cramfs 文件系统映象到 Nand Flash 的 root 分区

功能[a]: 通过 USB 下载用户程序到 Nand Flash 中, 一般这样的用户程序为 bin 可执行文件, 如 2440test(需要支持超过 4K 限制)、uCOS2(开发板中带的 uCOS2 支持 nand flash 启动)、U-Boot 等; 当然也可以是其他任意大小的 bin 程序。

功能[n]: 通过 USB 下载 WinCE 之启动程序 Nboot 到 Nand Flash 的 Block0

功能[e]: 通过 USB 下载 WinCE Bootloader 之 Eboot 到 Nand Flash 的 Eboot 分区

功能[i]: 通过 USB 下载 WinCE 运行时映象 NK.nb0 到 Nand Flash

功能[w]: 通过 USB 下载 WinCE 发行映象 NK.bin 到 Nand Flash

功能[d]: 通过 USB 下载程序到指定内存地址(通过 DNW 的 Configuration->Option 选项指定运行地址), 并运行。对于本开发板, SDRAM 的物理起始地址是 0x30000000, 结束地址是 0x34000000, 大小为 64Mbytes, 另外 BIOS 本身占用了 0x33DE8000 以上的空间, 因此在用 BIOS 的 USB 下载功能时应指定地址在 **0x30000000 - 0x33DE8000** 之间。

功能[z]: 通过 USB 下载 Linux 内核映像文件 zImage 到内存中, 下载地址为 0x30008000。

功能[g]: 运行内存中的 Linux 内核映像, 该功能一般配合功能[z]一起使用。

功能[f]: 擦除 Nand Flash, 执行此功能将会擦除指定整数地址的 Nand Flash 空间。对于本开发板, Nand Flash 的大小为 64Mbytes, 其空间地址范围是 0-0x4000000, 比较常用的操作是擦除 linux 的分区数据, 和擦除整片 Nand Flash, 其地址空间范围如下:



追求卓越 创造精品

TO BE BEST

TO DO GREAT

广州友善之臂计算机科技有限公司

常见擦除地址范围表

输入“f”，BIOS 将提示你输入起始地址和结束地址，该表为常用地址范围		
	起始地址	结束地址
擦除 vivi 分区数据(block0-13)	0x0	0x50000
擦除 linux 内核分区数据(block14-93)	0x50000	0x250000
擦除文件系统分区数据(block94-4095)	0x250000	0x4000000
擦除整片 Nand Flash(block0-4095)	0x0	0x4000000

提示：本开发板采用的 Nand Flash 总共有 4096 个 Block，每个 Block 有 32page，每个 page 有 512 byte，因此总共为： $4096 \times 32 \times 512 = 64M\text{ bytes}$

功能[p]: 对 Nand Flash 进行分区，主要用于 linux，详细见子菜单说明

功能[b]: 启动系统，如果烧入了 linux 或者 wince，执行从命令将自动辨认识别启动系统。

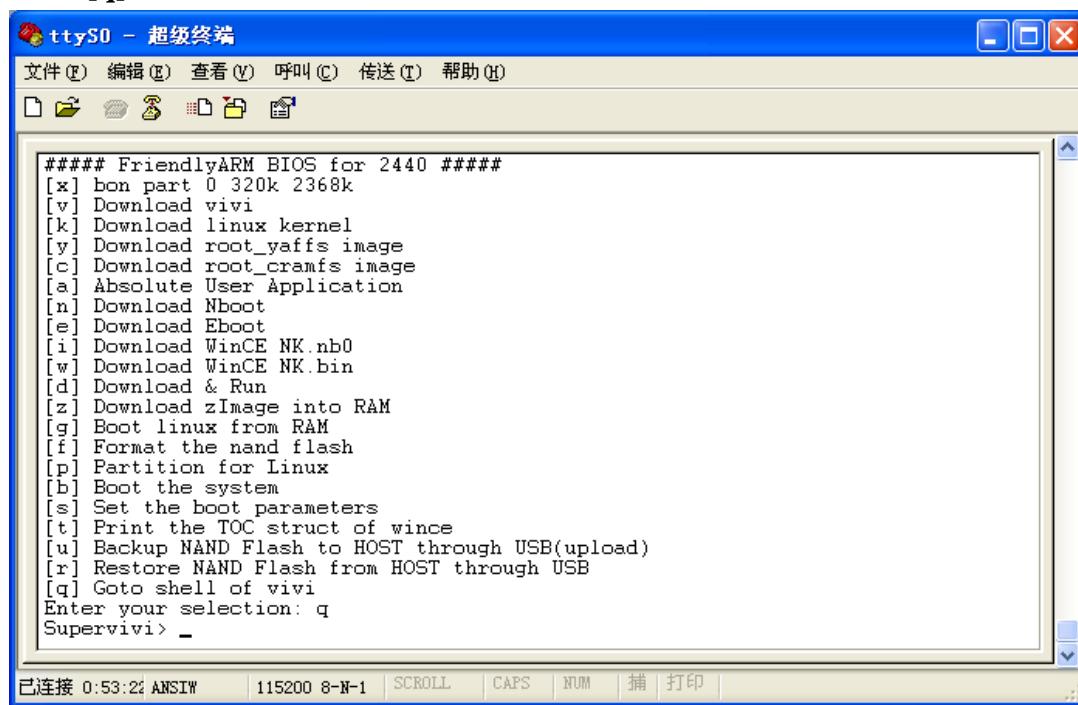
功能[s]: 设置 linux 启动参数，详细见子菜单说明

功能[t]: 打印 wince 内核映象的 TOC(很少用到)

功能[u]: 备份整个 Nand Flash 中的内容，通过 USB 上传到 PC 存储为一个文件，该功能类似于 PC 系统中经常用的 Ghost 工具。

功能[r]: 使用备份出来的文件恢复到 Nand Flash。

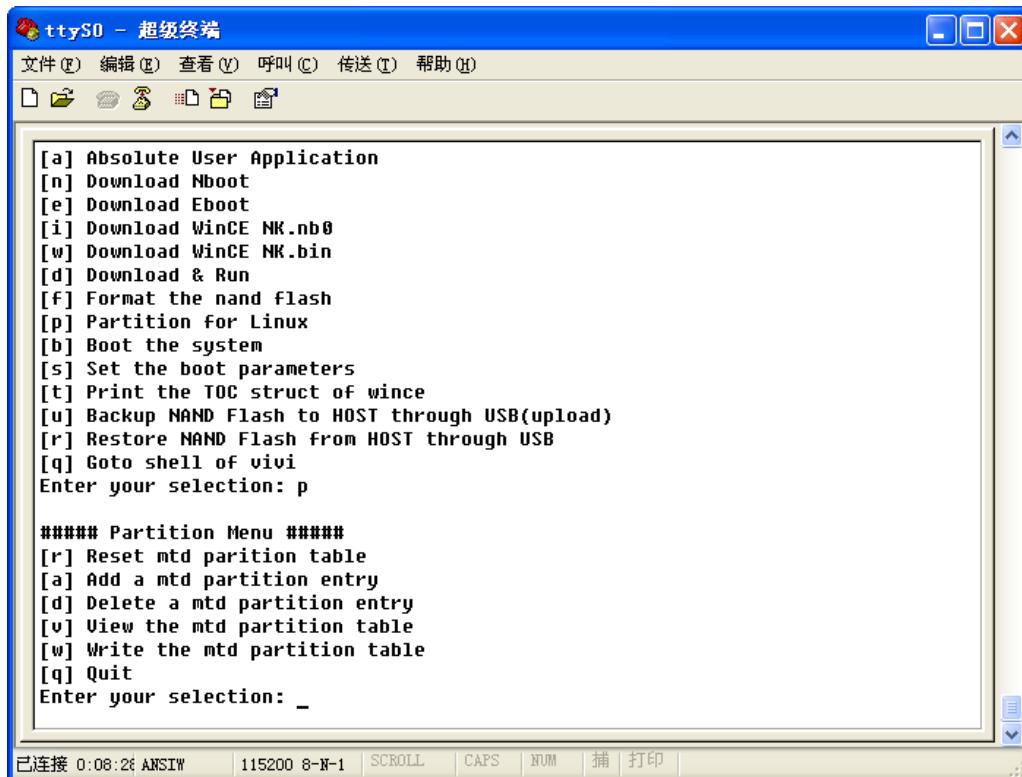
功能[q]: 返回 vivi 的命令交互模式，如图



在交互模式下输入 menu 命令，则可以返回到菜单模式。

## 2.2.4 分区子菜单功能说明

执行功能号[p]，进入其子菜单，如图：



### (1) 浏览分区[v]

输入“v”可以浏览当前的分区表，该表存在于 Nand Flash 中，如果 Nand Flash 是空的或者全新的，则会显示 BIOS 本身缺省的分区表，如图：



追求卓越 创造精品

TO BE BEST

TO DO GREAT

广州友善之臂计算机科技有限公司

ttyS0 - 超级终端

文件 (F) 编辑 (E) 查看 (V) 呼叫 (C) 传送 (T) 帮助 (H)

[r] Reset mtd partition table  
[a] Add a mtd partition entry  
[d] Delete a mtd partition entry  
[v] View the mtd partition table  
[w] Write the mtd partition table  
[q] Quit

Enter your selection: v

Number of partitions: 5

name	:	offset	size	flag
vivi	:	0x00000000	0x00028000	0
eboot	:	0x00028000	0x00018000	0
param	:	0x00040000	0x00010000	0
kernel	:	0x00050000	0x00200000	0
root	:	0x00250000	0x03dac000	0

##### Partition Menu #####

[r] Reset mtd partition table  
[a] Add a mtd partition entry  
[d] Delete a mtd partition entry  
[v] View the mtd partition table  
[w] Write the mtd partition table  
[q] Quit

Enter your selection:

已连接 0:09:12 ANSIW | 115200 8-N-1 | SCROLL | CAPS | NUM | 插 | 打印 |

## (2)删除分区[d]

输入“d”，将提示你输入要删除分区的名字，例如要删除 vivi 分区，则输入“vivi”（引号不需要输入），如图：

ttyS0 - 超级终端

文件 (F) 编辑 (E) 查看 (V) 呼叫 (C) 传送 (T) 帮助 (H)

[r] Reset mtd partition table  
[a] Add a mtd partition entry  
[d] Delete a mtd partition entry  
[v] View the mtd partition table  
[w] Write the mtd partition table  
[q] Quit

Enter your selection: d

Enter partition name : vivi

deleted 'vivi' partition

##### Partition Menu #####

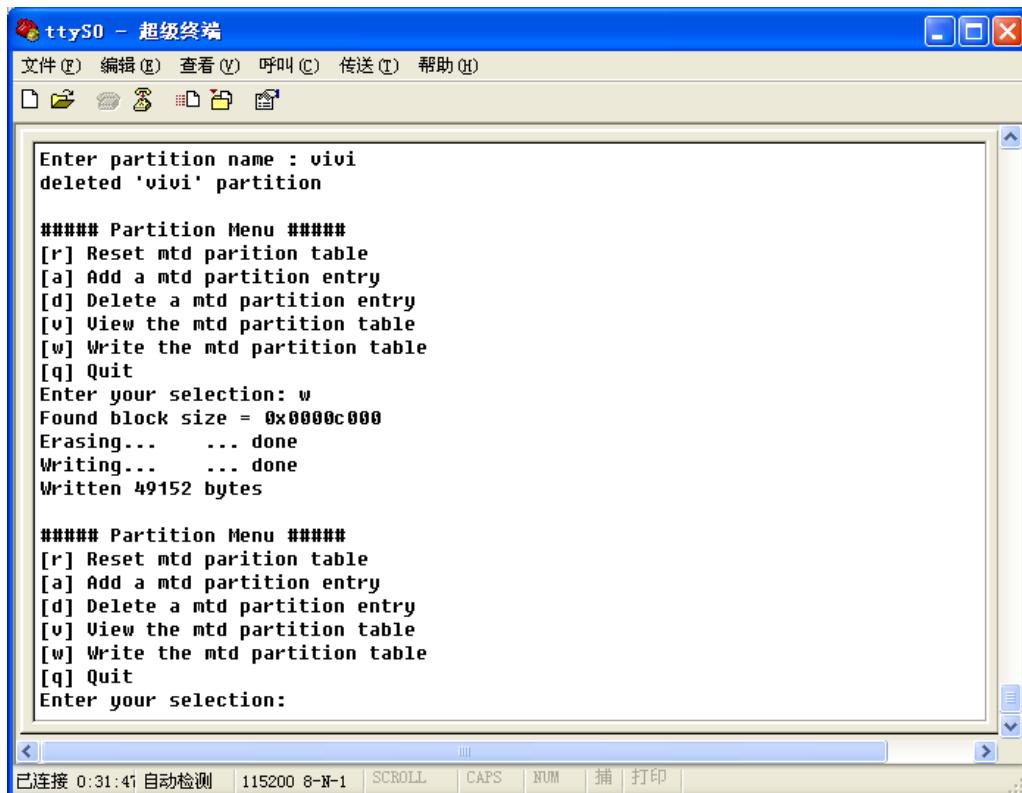
[r] Reset mtd partition table  
[a] Add a mtd partition entry  
[d] Delete a mtd partition entry  
[v] View the mtd partition table  
[w] Write the mtd partition table  
[q] Quit

Enter your selection:

已连接 0:09:45 ANSIW | 115200 8-N-1 | SCROLL | CAPS | NUM | 插 | 打印 |

### (3)保存操作[w]

输入“w”，可以对设置进行保存，如刚刚删除了vivi分区，如果不进行保存的话，下次启动系统浏览分区，您发现该分区会依然存在，执行保存的操作界面如图：



```
ttyS0 - 超级终端
文件(F) 编辑(E) 查看(V) 呼叫(C) 传送(T) 帮助(H)
□ ☐ ×

Enter partition name : vivi
deleted 'vivi' partition

##### Partition Menu #####
[r] Reset mtd partition table
[a] Add a mtd partition entry
[d] Delete a mtd partition entry
[v] View the mtd partition table
[w] Write the mtd partition table
[q] Quit
Enter your selection: w
Found block size = 0x0000c000
Erasing...    done
Writing...    done
Written 49152 bytes

##### Partition Menu #####
[r] Reset mtd partition table
[a] Add a mtd partition entry
[d] Delete a mtd partition entry
[v] View the mtd partition table
[w] Write the mtd partition table
[q] Quit
Enter your selection:

已连接 0:31:41 自动检测 | 115200 8-N-1 | SCROLL | CAPS | NUM | 捕 | 打印 | .
```

### (4)添加分区[a]

输入“a”，系统会提示你要添加分区的一些信息：名字、偏移地址、大小、标志位等，一般参考默认值就可以了，执行添加分区的操作界面如图：



追求卓越 创造精品

TO BE BEST

TO DO GREAT

广州友善之臂计算机科技有限公司

```
Written 49152 bytes

##### Partition Menu #####
[r] Reset mtd partition table
[a] Add a mtd partition entry
[d] Delete a mtd partition entry
[v] View the mtd partition table
[w] Write the mtd partition table
[q] Quit
Enter your selection: a
Enter partition name : vivi
Enter offset fo flash: 0x0
Enter size: 0x280000
Enter flag: 0
vivi: offset = 0x00000000, size = 0x00280000, flag = 0

##### Partition Menu #####
[r] Reset mtd partition table
[a] Add a mtd partition entry
[d] Delete a mtd partition entry
[v] View the mtd partition table
[w] Write the mtd partition table
[q] Quit
Enter your selection: _
```

#### (5)复位分区表[r]

执行“r”，可以使用 BIOS 本身自带的分区表信息，替换当前的所有分区表，当您不小心删除了不清楚的 linux 分区时，可以执行此功能恢复系统的 linux 分区表，当然，恢复之后要输入“w”保存才有效。

#### (6)返回主菜单[q]

执行“q”，将会返回上一级主菜单。

### 2.2.5 设置 linux 启动参数子菜单功能说明

通过该子菜单功能，可以更加灵活的启动 linux 系统，在 BIOS 主菜单执行功能号[s]，进入设置 linux 启动参数子菜单，如图：

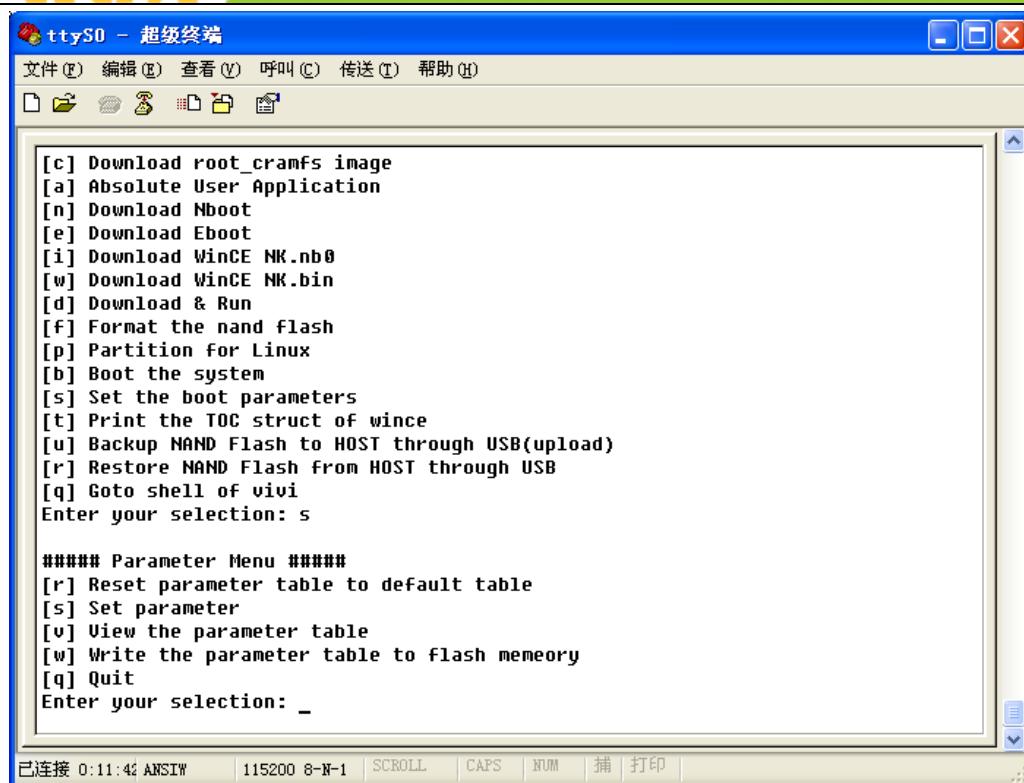


追求卓越 创造精品

TO BE BEST

TO DO GREAT

广州友善之臂计算机科技有限公司



### (1) 浏览当前参数设置[v]

输入“v”可以浏览当前启动参数设置情况：



追求卓越 创造精品

TO BE BEST

TO DO GREAT

广州友善之臂计算机科技有限公司

```
[w] Write the parameter table to flash memory
[q] Quit
Enter your selection: v
Number of parameters: 9
name          :      hex           integer
-----
mach_type     : 00000030e        782
media_type    : 00000003          3
boot_mem_base : 30000000        805306368
baudrate      : 0001c200        115200
xmodem         : 00000001          1
xmodem_one_nak: 00000000          0
xmodem_initial_timeout: 000493e0  300000
xmodem_timeout : 000F4240        1000000
boot_delay    : 01000000        16777216
Linux command line: noinitrd root=/dev/mtdblock2 init=/linuxrc console=ttySAC0

##### Parameter Menu #####
[r] Reset parameter table to default table
[s] Set parameter
[v] View the parameter table
[w] Write the parameter table to flash memory
[q] Quit
Enter your selection:
```

已连接 0:35:08 自动检测 | 115200 8-N-1 | SCROLL | CAPS | NUM | 插 | 打印 |

## (2) 设置参数[s]

输入“s”，可以对上面列出的参数进行设置，比较常用的参数有(其他参数建议不要更改):

- Mach\_type
- Linux command line

下面我们分别举例说明如何设置：

开发板默认的 MACH\_TYPE 为 782，假设你编译的内核使用的 MACH\_TYPE 是 867，则可以通过修改 mach\_type 参数来正常启动内核，根据提示先输入参数的名字“mach\_type”，再输入参数值“867”(引号不要输入)，更改后记得输入“w”保存设置，如图：



COM3 (1) - CRT

File Edit View Options Transfer Script Window Help

[v] View the parameter table  
[w] Write the parameter table to flash memory  
[q] Quit

Enter your selection: s

Enter the parameter's name(mach\_type, media\_type, linux\_cmd\_line, etc): mach\_type

Enter the parameter's value(if the value contains space, enclose it with "'): 782

Change 'mach\_type' value. 0x0000030e(782) to 0x0000030e(782)

##### Parameter Menu #####

[r] Reset parameter table to default table  
[s] Set parameter  
[v] View the parameter table  
[w] Write the parameter table to flash memory  
[q] Quit

Enter your selection: w

Found block size = 0x0000c000

Erasing... ... done

Writing... ... done

Written 49152 bytes

Saved vivi private data

##### Parameter Menu #####

[r] Reset parameter table to default table  
[s] Set parameter  
[v] View the parameter table  
[w] Write the parameter table to flash memory  
[q] Quit

Enter your selection: [ ]

`Linux_cmd_line` 是经常用到的一个内核启动参数，例如要把内核的启动信息和登录终端改为串口 1（默认是串口 0），则这样修改：

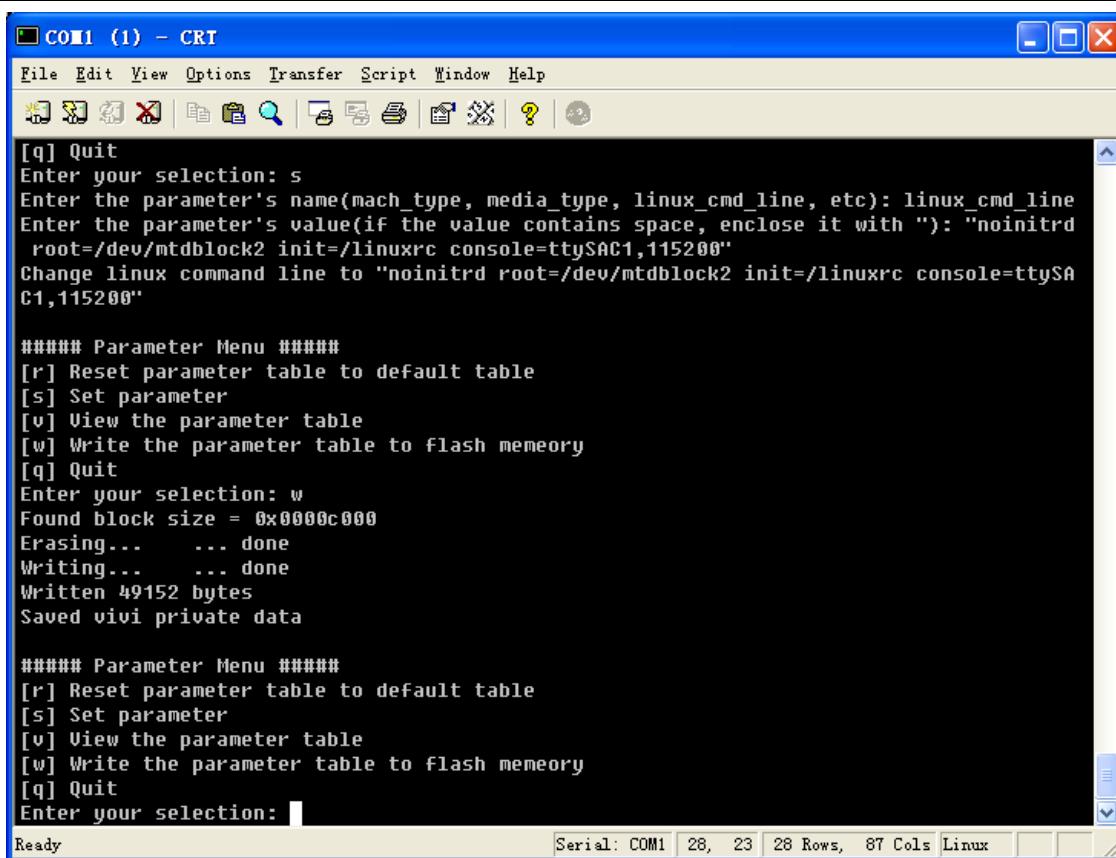
通过浏览参数，可以看到原来的参数：

Linux\_cmd\_line: **noinitrd root=/dev/mtdblock2 init=/linuxrc console=ttySAC0**

输入“s”后，根据提示输入要修改的参数“**linux\_cmd\_line**”，回车，再输入参数值为（因为该参数串中有空格，因此需要输入双引号括起来）：

**“noinitrd root=/dev/mtdblock2 init=/linuxrc console=ttySAC1,115200”**

如图所示：



这样系统启动的时候，内核的启动信息和登录信息都将在串口 1 出现，而 vivi 的输出信息不会改变，还是从串口 0 出来。

#### (3) 保存配置[w]

当设置更改之后，可以输入“w”保存所作的更改。

#### (4) 恢复默认值[r]

输入“r”可以恢复出厂时的内核启动参数。

#### (5) 返回主菜单[q]

输入“q”可以返回 BIOS 功能主菜单。

## 2.3 非操作系统下的外围资源测试

在非操作系统下，主要测试 PWM 控制蜂鸣器，RTC 实时时钟测试，AD 转换测试，按键，触摸屏，各种 LCD，红外测试，I2C 总线测试，音频输入与输出，SD 卡功能。

### 2.3.1 下载运行测试程序

**说明：**2440test 是一个裸机测试程序，它不是一个操作系统，该程序由三星原厂的同名文件修改而来，我们根据实际情况，更改了输出菜单和各项测试内容，使其更加简洁明了，



追求卓越 创造精品

TO BE BEST

TO DO GREAT

广州友善之臂计算机科技有限公司

为了方便用户测试和使用，我们分别编译出了针对不同类型 LCD 显示输出的可执行二进制文件(见下表)，用户使用本节的步骤通过 USB 下载到内存中即可运行，区别之处在于默认显示 LCD 显示输出的类型不同；实际上它们都是使用相同的代码编译出的，只需在头文件中更改一下 LCD 类型(2440test\inc\Option.h 中 “LCD\_TYPE” 定义)即可。

文件名	说明	备注
2440test_N35.bin	默认显示输出支持 NEC3.5 寸液晶屏	
2440test_A70.bin	默认显示输出支持群创 7 寸液晶屏	
2440test_VGA1024x768.bin	默认显示输出支持 VGA(分辨率：1024x768@70Hz)	因为它们的代码都是相同的，以下我们把这几个针对不同显示输出的测试文件统称为 <b>2440test.bin</b>

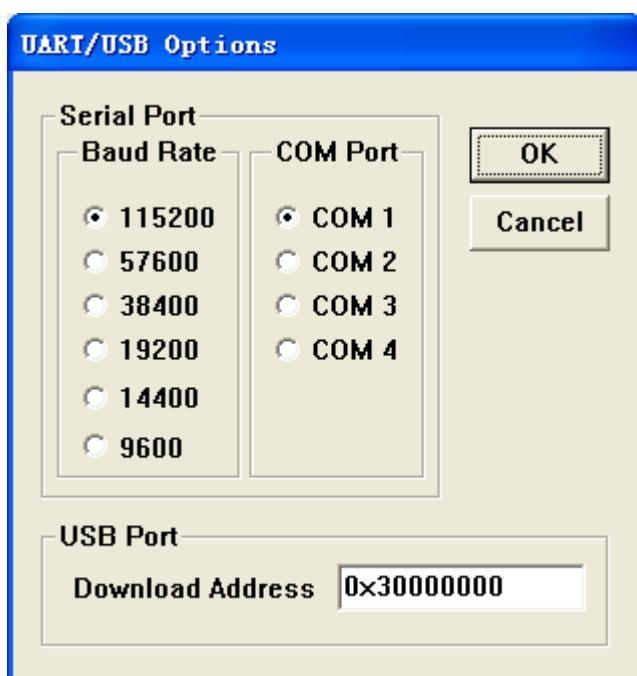
在光盘 “images\” 目录中找到 **2440test.bin** 文件，通过 BIOS 下载运行该测试程序，步骤如下：

(1)连接好开发板电源，串口线，USB 线，并**设置拨动开关 S2 为 Nor Flash 启动系统**，分别打开串口超级终端和 DNW，上电启动开发板。

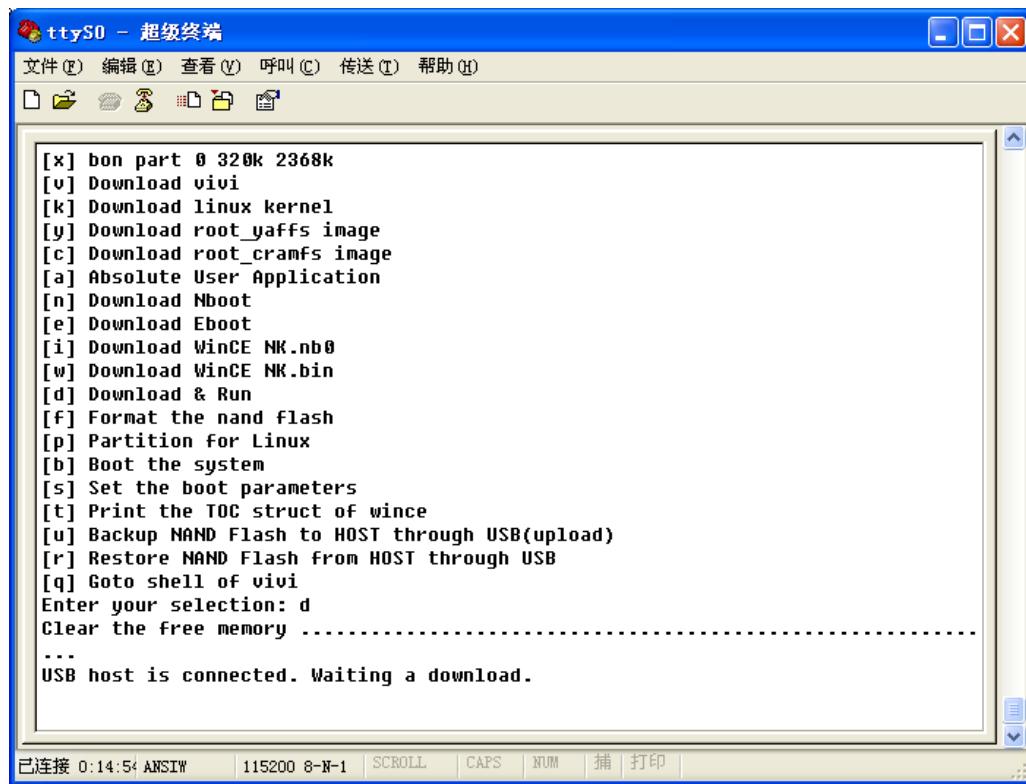
(2)保证 USB 驱动已经安装好(前面已经详细介绍了 USB 驱动的安装方法)，这时可以看到 DNW 的标题栏显示[USB: OK]，如果没有安装好驱动会显示[USB: x]，如图所示：



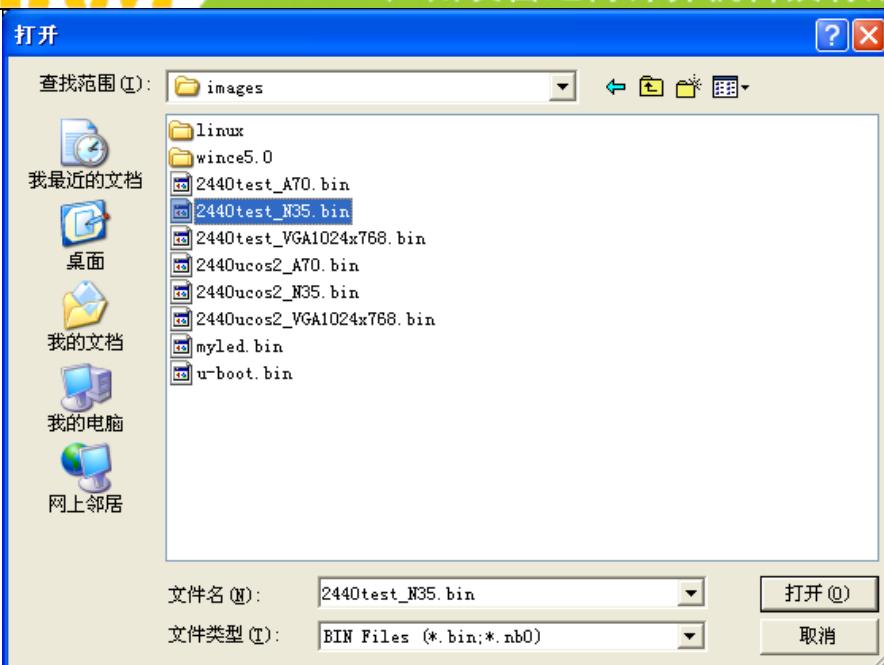
(3)点 DNW 菜单 Configuration，设置 USB 下载运行地址为 0x30000000



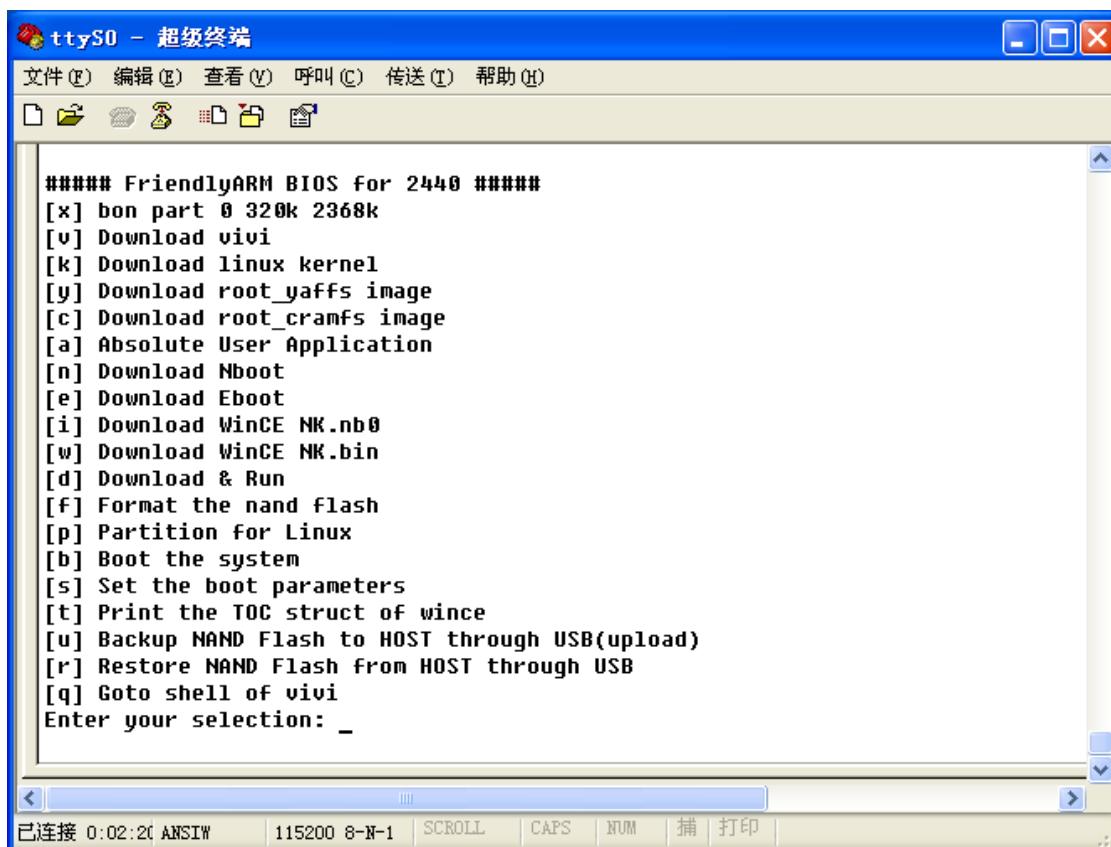
(4)这时在超级终端的 BIOS 功能菜单中选择功能号[d], 出现 USB 下载等待提示信息:



(5)点击 DNW 程序的“USB Port”→“Transmit”，选择 2440test.bin 这个映象文件(在光盘的 images 目录下面)，接着点“打开”，这样就开始下载了。



(6) 下载结束后，会自动运行，出现如下界面：



同时在 LCD 上会出现如下界面。

**说明：** 编译 2440test 的时候，通过“2440test\inc\Option.h”文件中“LCD\_TYPE”的定义



追求卓越 创造精品

TO BE BEST

TO DO GREAT

广州友善之臂计算机科技有限公司

可以选择初始化液晶屏的型号,不能同时选择两种以上的型号,在此默认为 LCD\_TYPE\_N35,即 NEC3.5 寸真彩屏。

2440test\inc\Option.h 中液晶屏型号的定义:

```
#define LCD_TYPE_N35      1 ; NEC3.5 寸真彩屏型号定义  
#define LCD_TYPE_A70      2 ; 7 寸真彩屏型号定义  
#define LCD_TYPE_VGA1024x768 3; VGA 模块, 分辨率: 1024x768 @70Hz  
#define LCD_TYPE_LCD_TYPE_N35
```

若使用 NEC3.5 寸屏(**2440test.bin** 默认), 会出现如下界面:



若使用 7 寸真彩屏, 会出现如下界面:



若使用 VGA 模块(1024x768 @70Hz), 会出现如下界面:

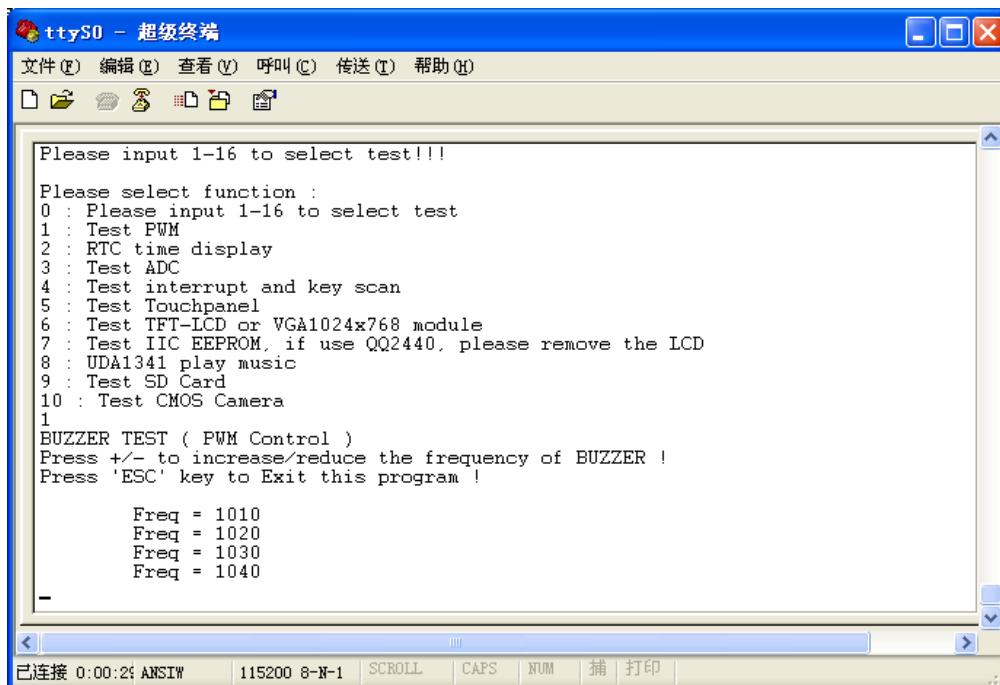


## 2.3.2 外围资源测试

测试程序运行后，就可以进行相应的外围资源测试了，通过选择测试程序主菜单相应的选项，可以执行测试。

### (1)蜂鸣器测试(Test PWM)

在主菜单中，输入“1”，再按“回车”键(即 Enter 键)，将开始进行蜂鸣器测试，蜂鸣器测试运行起来后，将会听到蜂鸣器发出声音。



按“—”键，蜂鸣器频率会降低，按“+”键频率升高，按“ESC”键可以推出该测试，并返回到主菜单中。

### (2)实时时钟测试

在测试程序主菜单中，选择“2”，再按“回车”键，可以看到秒钟在不断的变化，这说明 CPU 的 RTC 在正常工作（注意：该时间并不是当前的时间，因为测试程序对其初始化并进行了赋值）



追求卓越 创造精品

TO BE BEST

TO DO GREAT

广州友善之臂计算机科技有限公司

```
3 : Test ADC
4 : Test interrupt and key scan
5 : Test Touchpanel
6 : Test TFT-LCD or VGA1024x768 module
7 : Test IIC EEPROM, if use QQ2440, please remove the LCD
8 : UDA1341 play music
9 : Test SD Card
10 : Test CMOS Camera
2RTC TIME Display, press ESC key to exit !
RTC time : 2005-06-19 15:21:30
RTC time : 2005-06-19 15:21:31
RTC time : 2005-06-19 15:21:31
RTC time : 2005-06-19 15:21:32
RTC time : 2005-06-19 15:21:33
RTC time : 2005-06-19 15:21:34
RTC time : 2005-06-19 15:21:35
RTC time : 2005-06-19 15:21:36
RTC time : 2005-06-19 15:21:37
RTC time : 2005-06-19 15:21:38
RTC time : 2005-06-19 15:21:39
RTC time : 2005-06-19 15:21:40
RTC time : 2005-06-19 15:21:40
RTC time : 2005-06-19 15:21:41
```

已连接 0:01:04 ANSIW 115200 8-N-1 SCROLL CAPS NUM 插 打印

按“ESC”键退出该测试，并返回到主菜单。

### (3)AD 测试

在主菜单中，输入“3”，再按“回车”键，开始执行 AD 测试。

用户可以使用螺丝刀调节开发板上的 W1 或者 W2(这两个可调电阻接了 AIN0 和 AIN1)，可以看到 AD 的值在跟随调节电压在不断的变化。

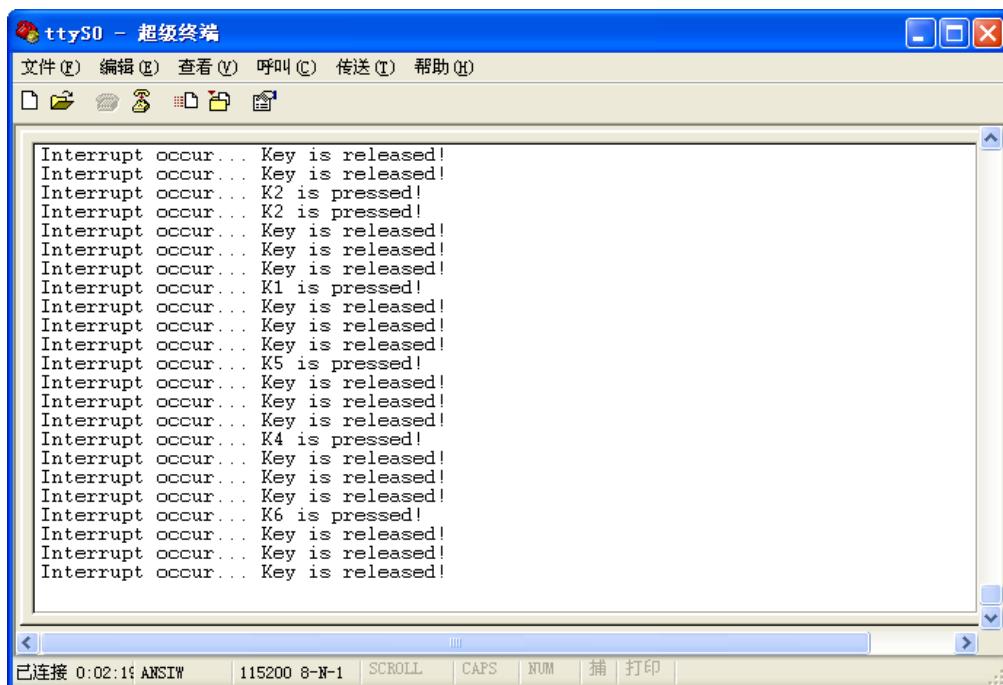
```
1 : Test PWM
2 : RTC time display
3 : Test ADC
4 : Test interrupt and key scan
5 : Test Touchpanel
6 : Test TFT-LCD or VGA1024x768 module
7 : Test IIC EEPROM, if use QQ2440, please remove the LCD
8 : UDA1341 play music
9 : Test SD Card
10 : Test CMOS Camera
3ADC INPUT Test, press ESC key to exit !
ADC conv. freq. = 2500000Hz
PCLK/ADC_FREQ - 1 = 19
AIN0: 0705, AIN1: 0267
AIN0: 0704, AIN1: 0268
AIN0: 0529, AIN1: 0293
AIN0: 0720, AIN1: 0323
AIN0: 1023, AIN1: 0374
AIN0: 1023, AIN1: 0381
AIN0: 0955, AIN1: 0299
AIN0: 0847, AIN1: 0340
AIN0: 0764, AIN1: 0326
AIN0: 0764, AIN1: 0284
```

已连接 0:01:46 ANSIW 115200 8-N-1 SCROLL CAPS NUM 插 打印

按“ESC”键退出该测试，并返回到主菜单。

**(4)按键测试**

在主菜单中输入“4”，再按“回车”键，开始执行按键测试，此时按开发板上的 K1-K6 按键进行测试，可以看到串口终端打印相应的按键信息。



```
Interrupt occur... Key is released!
Interrupt occur... Key is released!
Interrupt occur... K2 is pressed!
Interrupt occur... K2 is pressed!
Interrupt occur... Key is released!
Interrupt occur... Key is released!
Interrupt occur... Key is released!
Interrupt occur... K1 is pressed!
Interrupt occur... Key is released!
Interrupt occur... Key is released!
Interrupt occur... Key is released!
Interrupt occur... K5 is pressed!
Interrupt occur... Key is released!
Interrupt occur... Key is released!
Interrupt occur... Key is released!
Interrupt occur... K6 is pressed!
Interrupt occur... Key is released!
Interrupt occur... Key is released!
Interrupt occur... Key is released!
```

按“ESC”键退出该测试，并返回到主菜单。

**(5)触摸屏测试**

如果选购了 LCD 液晶屏，请使用附带的电缆连接开发板上的 LCD 接口。在主菜单中输入“5”，按“回车”开始进行触摸屏测试，这时用附带的触摸笔点击触摸屏，可以看到串口终端打印触摸点的坐标信息。

```

0 : Please input 1-16 to select test
1 : Test PWM
2 : RTC time display
3 : Test ADC
4 : Test interrupt and key scan
5 : Test Touchpanel
6 : Test TFT-LCD or VGA1024x768 module
7 : Test IIC EEPROM, if use QQ2440, please remove the LCD
8 : UDA1341 play music
9 : Test SD Card
10 : Test CMOS Camera
5ADC touch screen test

Type any key to exit!!!

Stylus Down, please.....
count=000  XP=0395,  YP=0482
count=001  XP=0535,  YP=0480
count=002  XP=0693,  YP=0362
count=003  XP=0483,  YP=0538
count=004  XP=0313,  YP=0461
count=005  XP=0557,  YP=0366
count=006  XP=0533,  YP=0476

```

已连接 0:02:40 ANSIW 115200 8-N-1 SCROLL CAPS NUM 插 打印

按“ESC”键退出该测试，并返回到主菜单。

### (6)LCD 或 VGA 模块输出测试

请烧写相应的 2440test 测试程序，主菜单中输入“6”按“回车”键开始执行测试，接着按照提示按任意键，LCD 将不断变化显示，直到最后显示一幅图片结束，并返回主菜单。

```

8 : UDA1341 play music
9 : Test SD Card
10 : Test CMOS Camera
6
Test TFT LCD 240x320!

LCD clear screen is finished! press any key to continue!
LCD clear screen is finished! press any key to continue!
LCD color test, please look! press any key to continue!
LCD paint a bmp, please look! press any key to continue!

Please select function :
0 : Please input 1-16 to select test
1 : Test PWM
2 : RTC time display
3 : Test ADC
4 : Test interrupt and key scan
5 : Test Touchpanel
6 : Test TFT-LCD or VGA1024x768 module
7 : Test IIC EEPROM, if use QQ2440, please remove the LCD
8 : UDA1341 play music
9 : Test SD Card
10 : Test CMOS Camera
-
```

已连接 0:04:31 ANSIW 115200 8-N-1 SCROLL CAPS NUM 插 打印

### (7)I2C 测试

在主菜单中输入“7”，按“回车”键开始执行测试，程序将对 I2C 总线的芯片 AT24C08



追求卓越 创造精品

TO BE BEST

TO DO GREAT

广州友善之臂计算机科技有限公司

进行读写，该测试主要是通过向 AT24C08 写入 0x-0xFF，然后读取出来。

```
60 61 62 63 64 65 66 67 68 69 6a 6b 6c 6d 6e 6f  
70 71 72 73 74 75 76 77 78 79 7a 7b 7c 7d 7e 7f  
80 81 82 83 84 85 86 87 88 89 8a 8b 8c 8d 8e 8f  
90 91 92 93 94 95 96 97 98 99 9a 9b 9c 9d 9e 9f  
a0 a1 a2 a3 a4 a5 a6 a7 a8 a9 aa ab ac ad ae af  
b0 b1 b2 b3 b4 b5 b6 b7 b8 b9 ba bb bc bd be bf  
c0 c1 c2 c3 c4 c5 c6 c7 c8 c9 ca cb cc cd ce cf  
d0 d1 d2 d3 d4 d5 d6 d7 d8 d9 da db dc dd de df  
e0 e1 e2 e3 e4 e5 e6 e7 e8 e9 ea eb ec ed ee ef  
f0 f1 f2 f3 f4 f5 f6 f7 f8 f9 fa fb fc fd fe ff  
  
Please select function :  
0 : Please input 1-16 to select test  
1 : Test PWM  
2 : RTC time display  
3 : Test ADC  
4 : Test interrupt and key scan  
5 : Test Touchpanel  
6 : Test TFT-LCD or VGA1024x768 module  
7 : Test IIC EEPROM, if use QQ2440, please remove the LCD  
8 : UDA1341 play music  
9 : Test SD Card  
10 : Test CMOS Camera
```

已连接 0:04:49 ANSIW 115200 8-N-1 SCROLL CAPS NUM 捕 打印

这个测试结束后，会自动回到主菜单。

### (8)音频输出测试

先将音箱接到开发板的绿色耳机孔座，在主菜单中输入“8”，按“回车”开始音频输出测试，这时可以从音箱听到 XP 的启动声音。

```
d0 d1 d2 d3 d4 d5 d6 d7 d8 d9 da db dc dd de df  
e0 e1 e2 e3 e4 e5 e6 e7 e8 e9 ea eb ec ed ee ef  
f0 f1 f2 f3 f4 f5 f6 f7 f8 f9 fa fb fc fd fe ff  
  
Please select function :  
0 : Please input 1-16 to select test  
1 : Test PWM  
2 : RTC time display  
3 : Test ADC  
4 : Test interrupt and key scan  
5 : Test Touchpanel  
6 : Test TFT-LCD or VGA1024x768 module  
7 : Test IIC EEPROM, if use QQ2440, please remove the LCD  
8 : UDA1341 play music  
9 : Test SD Card  
10 : Test CMOS Camera  
8  
Sample Rate = 22050, Channels = 2, 16BitsPerSample, size = 243508  
  
err = 0  
Now playing the file  
Press 'ESC' to quit, '+' to inc volume, '-' to dec volume, 'm' to mute, 'p' to pause  
-
```

已连接 0:05:08 ANSIW 115200 8-N-1 SCROLL CAPS NUM 捕 打印

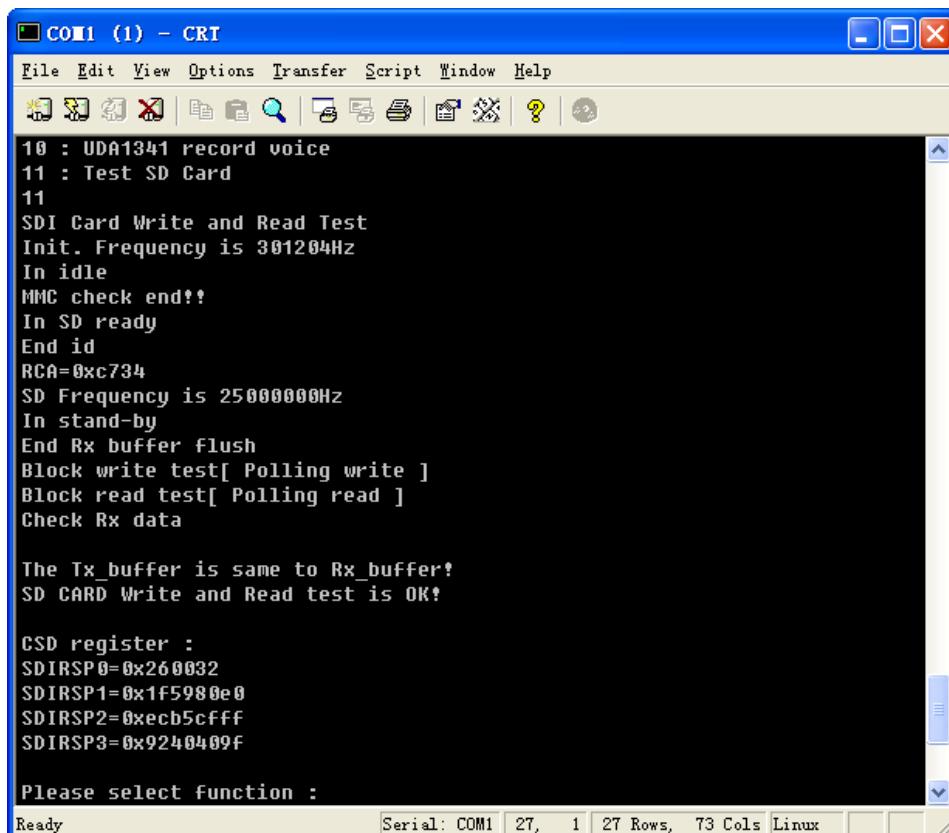
按“+”或者“-”可以增加或者减小音量，按“ESC”键退出测试，返回主菜单。

### (9)SD 卡测试

**注意：**本测试会破坏 SD 卡中的数据，试用前请备份好 SD 卡中的数据。

先将 SD 卡插入开发板的 SD 卡插座。

在主菜单中输入“9”，按“回车”开始执行测试，程序将对 SD 卡进行读写，并出现如下界面：



The screenshot shows a terminal window titled "COM1 (1) - CRT". The window has a menu bar with File, Edit, View, Options, Transfer, Script, Window, Help. Below the menu is a toolbar with icons for copy, paste, cut, find, etc. The main text area displays the following output:

```
10 : UDA1341 record voice
11 : Test SD Card
11
SDI Card Write and Read Test
Init. Frequency is 301204Hz
In idle
MMC check end!!
In SD ready
End id
RCA=0xc734
SD Frequency is 25000000Hz
In stand-by
End Rx buffer flush
Block write test[ Polling write ]
Block read test[ Polling read ]
Check Rx data

The Tx_buffer is same to Rx_buffer!
SD CARD Write and Read test is OK!

CSD register :
SDIRSP0=0x260032
SDIRSP1=0x1F5980e0
SDIRSP2=0xecb5cff
SDIRSP3=0x9240409f

Please select function :
```

At the bottom, it says "Ready" and "Serial: COM1 | 27, 1 | 27 Rows, 73 Cols | Linux".

该界面显示 SD 卡读写成功，测试完毕，自动退回到主菜单。

### (11)测试 CMOS 摄像头

如果您选购了本公司提供的 CAM130 型号的 CMOS 摄像头，可以进行本功能测试。

开机之前，把 CAM130 摄像头模块按照板上箭头方向插到开发板的“CAMERA”插座上，在主菜单中输入“10”，按“回车”开始执行测试，

**注意：**如果使用的是 7 寸屏或者 VGA 输出模块，LCD 显示界面会有所不同。

```

Please select function :
0 : Please input 1-16 to select test
1 : Test PWM
2 : RTC time display
3 : Test ADC
4 : Test interrupt and key scan
5 : Test Touchpanel
6 : Test TFT-LCD or VGA1024x768 module
7 : Test IIC EEPROM, if use QQ2440, please remove the LCD
8 : UDA1341 play music
9 : Test SD Card
10 : Test CMOS Camera
10
Camera Preview Test
CAMERA : UPLL 96000000 UCLK 48000000 CAMCLK 24000000
Check camera ID
Initial Camera now, Please wait several minutes...
Initializing end...

Now Start Camera Preview
preview sc control = 0
preview sc control = 81d5018e
Press 'ESC' key to exit!

```

已连接 0:08:24 ANSIW 115200 8-N-1 SCROLL CAPS NUM 插 打印

使用 NEC3.5”寸屏时，CMOS 摄像头效果：



## 2.4 Linux 之图形界面 Qtopia 系统测试与使用说明(预装)

**注意：本小节内容的介绍均基于 mini2440 + 3.5" LCD 的屏幕截图。**

本开发板提供了两种 Linux 系统以方便用户测试使用，这两种系统所使用的内核文件是完全相同的，只是文件系统的内容不同：

一种是非图形界面的系统，它的安装文件对应于光盘中的 root\_default.img，它的源文件压缩包为 root\_default.tgz(注意：并非源代码)。该系统基于 Linux 工具集 busybox-1.2 和 Linux glibc-2.3.2 库文件制作而成，另外我们还专门针对本开发板集成了很多基于命令行的实用测试工具，如 mp3 播放，图片播放，按键测试，LED 测试等等。正因为如此，这样的系统可以做的比较小型化，系统剩余的空间(即 Flash 剩余空间)也就比较大，用户也可以根据自己的需要，灵活的删除、添加一些程序，以实现灵活的定制化。

另一种是嵌入式图形界面 Qtopia 系统(PDA 版)，它其实是基于非图形界面系统的扩展，使用的 glibc 基本库都是相同的。Qtopia 本身有很多实用的小程序，根据用途不同，这些小程序被分为三类：应用程序、游戏和设置，另外还有一个“文档”。同样地，我们针对本开发板也开发了一些 Qtopia 的实用小程序，例如 USB 摄像头拍照、PWM 控制蜂鸣器、EEPROM 读写、LED 控制等，它们都是基于 3.5" LCD 的程序，我们把这些自主开发的程序归为一类“友善之臂”。下面的内容是详细的使用说明。

**注意：**出厂之前，Mini2440 一般预装了图形界面的 Qtopia 系统，第一次拿到开发板的时候，或许触摸屏已经校正好。

使用 3.5 寸屏时，开机后的显示界面：



使用 7 寸屏时，开机后的显示界面：



追求卓越 创造精品

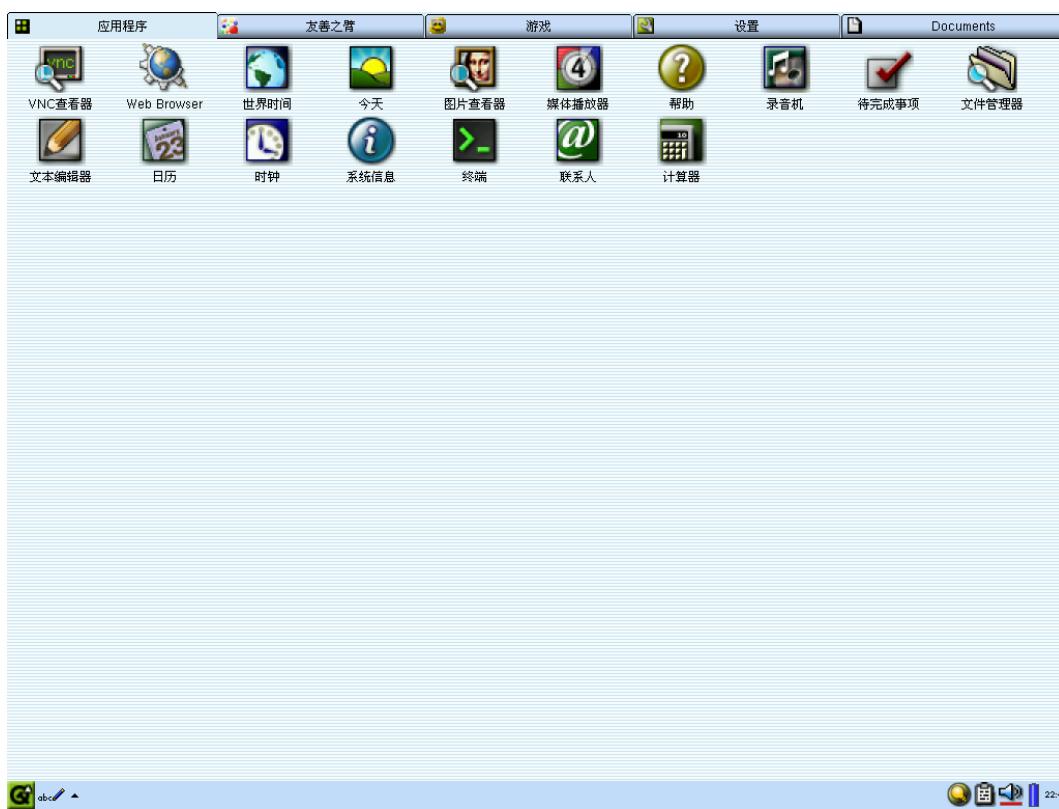
TO BE BEST

TO DO GREAT

广州友善之臂计算机科技有限公司



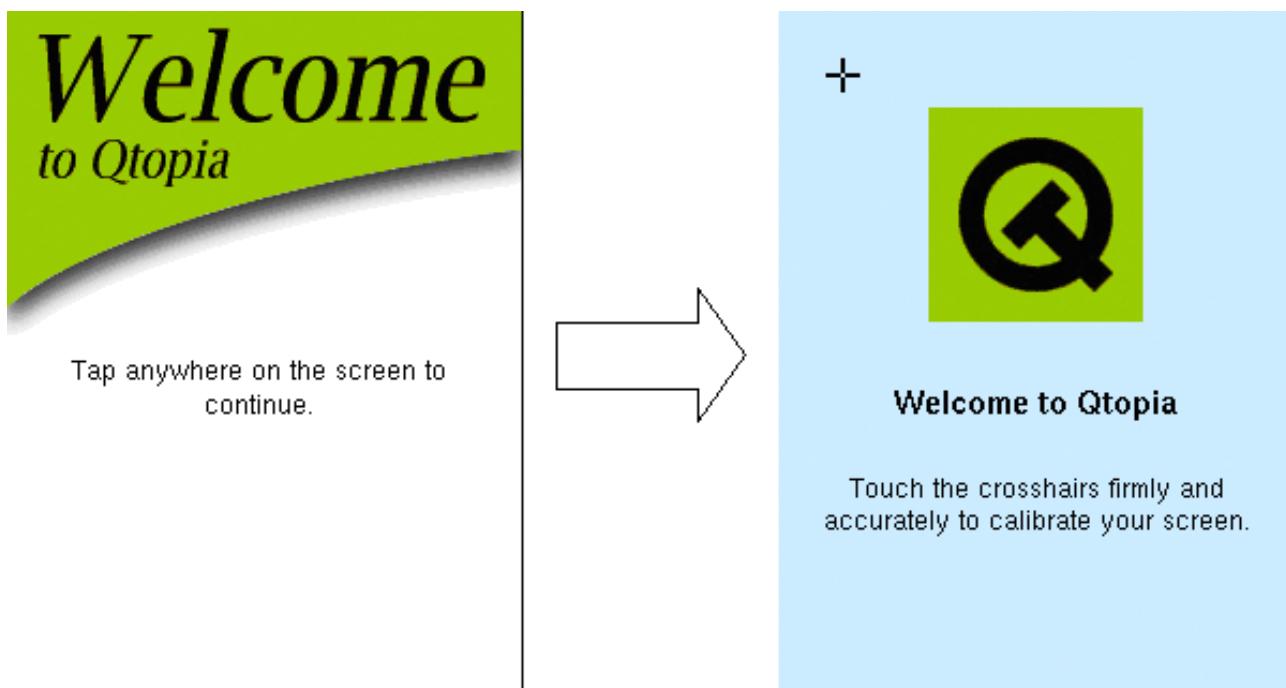
使用 VGA 模块输出时，开机后的显示界面：



## 2.4.1 触摸屏校正

在两种情况下可以会出现触摸屏校正界面：

1. 如果你按照第三章的步骤重新安装了 Qtopia 系统，重启系统时首先出现如下“欢迎”界面，依屏幕提示点击屏幕任何地方开始进行校正；然后依照屏幕提示，使用触摸屏逐步点击“十”型交叉点即可。



2. 进入系统后，点“开始->设置”切换到“设置”界面，再点“触摸屏校正”图标也会出现校正界面；然后依照屏幕提示，使用触摸屏逐步点击“十”型交叉点即可。



## 2.4.2 主要界面说明

进入 Qtopia 系统后主界面如下图所示：



可以看到 Qtopia 系统界面上方有五个图标，它们代表了五类程序/文档，单击任何一个图标都可以进入相应的子类界面，它们都是类似的。

另外点系统界面左下角的“开始”图标，也可以出现五个子类选择菜单，它们和系统界面上方的图标是对应的。



追求卓越 创造精品

TO BE BEST

TO DO GREAT

广州友善之臂计算机科技有限公司

五个子类界面分别如下图所示，根据标题名称，其意自明。



其中“友善之臂”子类中所有程序均为友善之臂公司自主开发，供测试使用。其他子类中程序或为系统自带，或为移植而来。

### 2.4.3 播放 Mp3

在子类“应用程序”中单击“媒体播放器”图标，出现播放器界面，在“Audio”列表中选择一首 mp3 歌曲文件，再点上方的“播放”按钮，开始播放 Mp3 文件。

说明：Audio 列表中的音频文件对应“Documents”子类中的所有有效音频文件。

提示：也可以在“Documents”中直接点击相应的文件名开始播放。

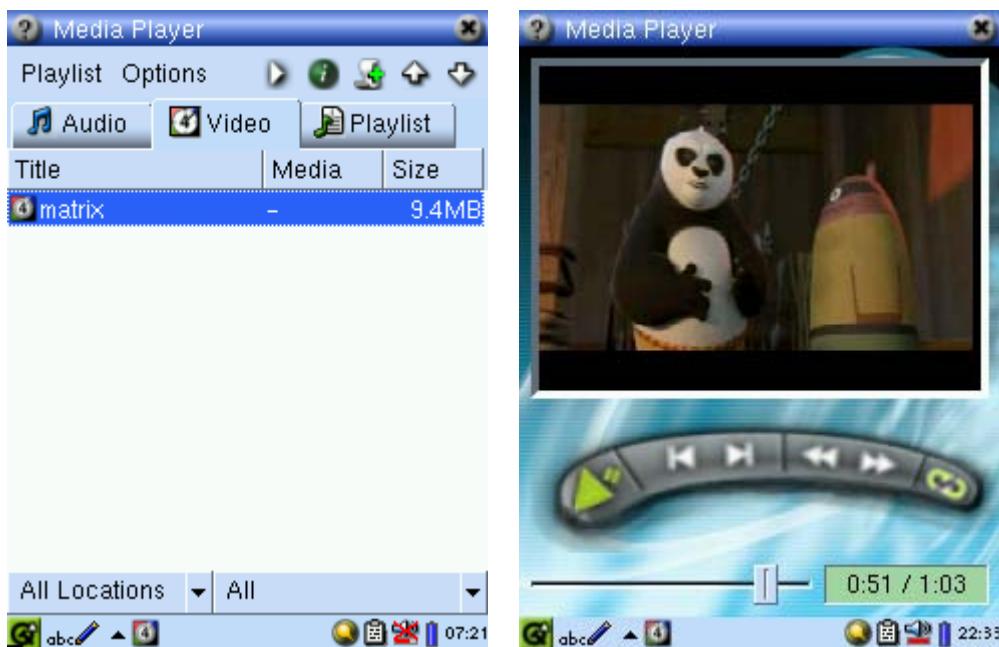


## 2.4.4 播放视频

在子类“应用程序”中单击“媒体播放器”图标，出现播放器界面，在“Video”列表中选择一个视频文件，再点上方的“播放”按钮，开始播放视频。

说明：Video列表中的音频文件对应“Documents”子类中的所有有效视频文件。

提示：也可以在“Documents”中直接点击相应的文件名开始播放。

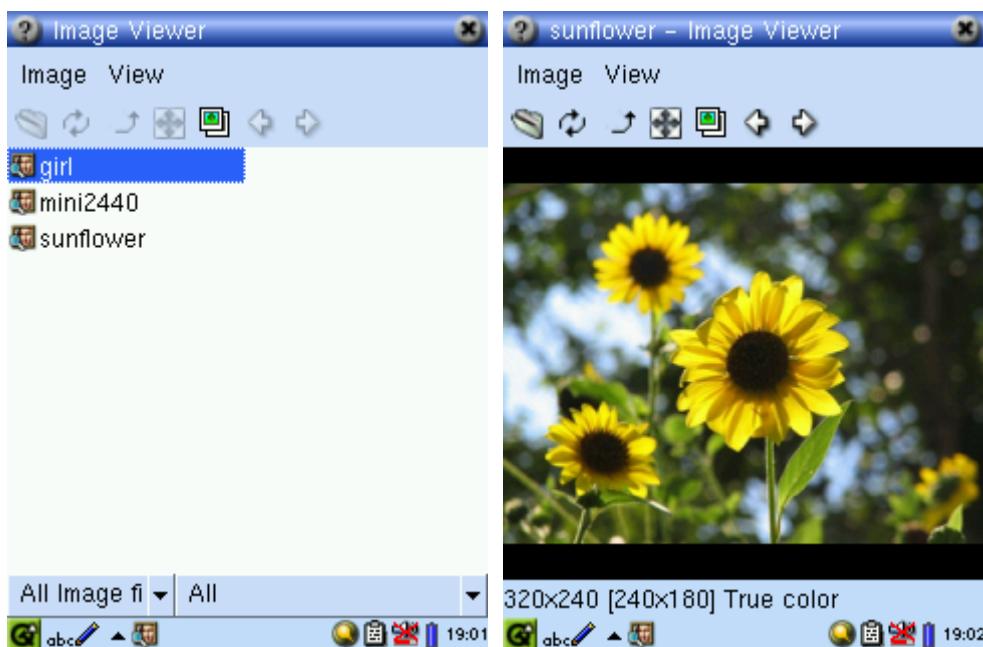


## 2.4.5 图片浏览

在子类“应用程序”中单击“图片查看器”图标，出现查看器界面，直接点击列表中的图片文件即可开始浏览，界面上方还有旋转、全屏、幻灯片播放的按钮，用户可以自行测试；目前本开发板支持 jpg、png、bmp 格式的图片。

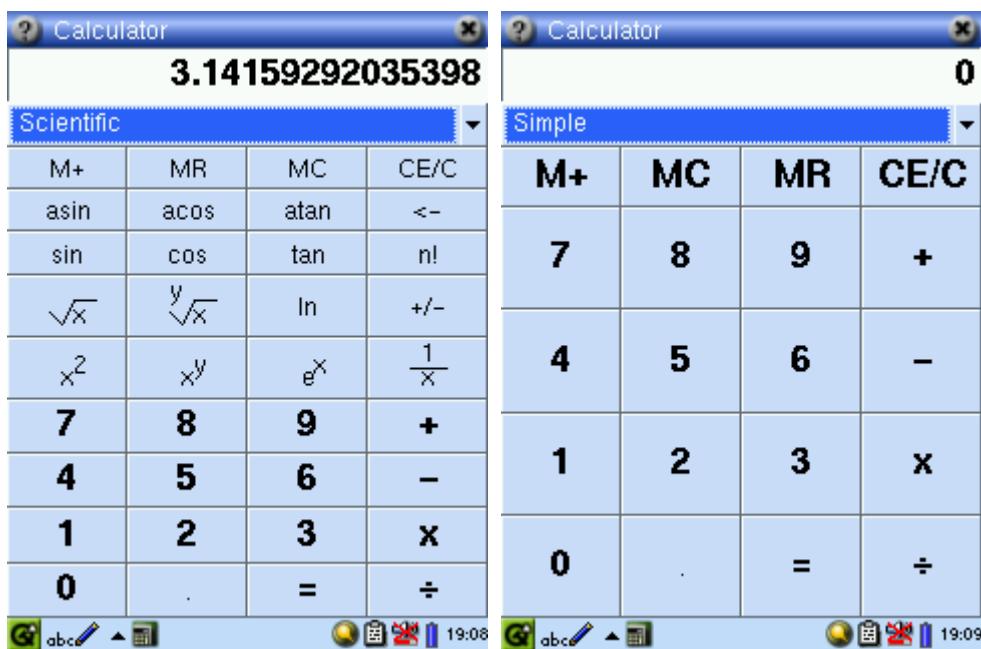
说明：图片查看器列表中的图片文件对应“Documents”子类中的所有有图片文件。

提示：也可以在“Documents”中直接点击相应的文件名开始浏览。



## 2.4.6 计算器

在子类“应用程序”中单击“计算器”图标，出现计算器界面，用户可以通过下拉列表选择“科学计算器”(Scientific)或者“普通计算器”(Simple)，如图：



## 2.4.7 命令终端

“终端”是 Linux 系统中通常用到的交互操作界面，通过“终端”可以运行很多 Linux 命令，查看系统信息等等。

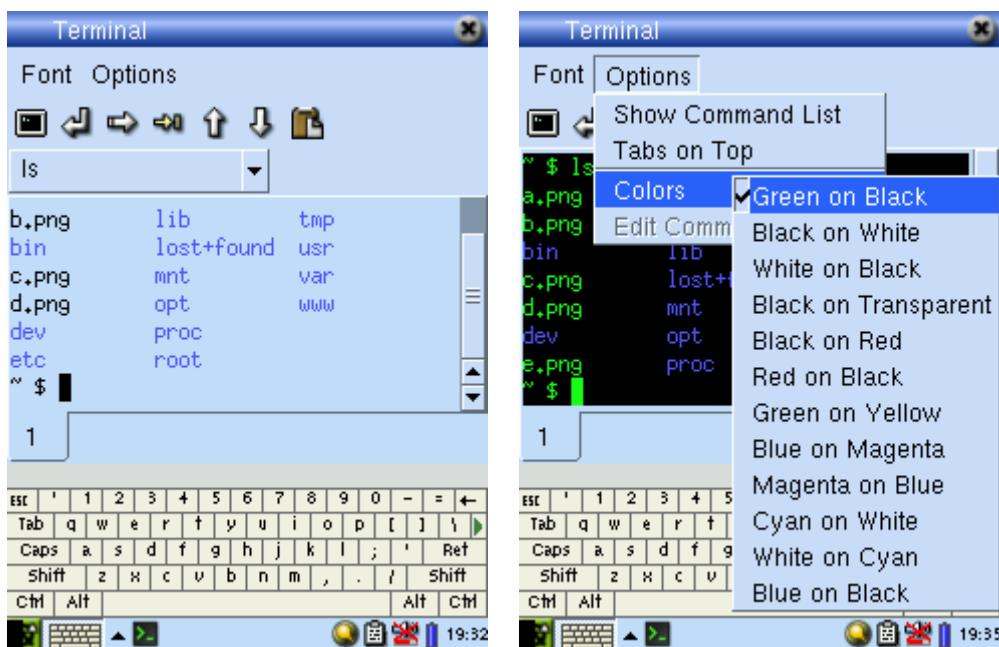
有很多途径可以设置或打开一个“终端”：

在 Linux 系统启动的时候，可以把终端指向串口输出，这样就形成了串口终端，它的输入和输出都是通过串口进行的，无需图形界面，这是嵌入式 Linux 开发中最常用的方式。

在系统启动的时候，也可以把终端输出指向图形显示设备（如 LCD 或者 CRT 等），而把键盘设定为输入，这样就形成了一套独立的“输入输出系统”，它无需借助另外的 PC 即可操作。

当使用了图形显示设备，并且系统软件中增加了图形用户界面(GUI)时，就可以建立一个基于 GUI 系统的“命令终端窗口”，这时既可以通过标准的实体硬件键盘进行交互，也可以通过虚拟的“软键盘”进行交互，此处所讲的就是这种终端方式。

在子类“应用程序”中单击“终端”图标，出现命令终端窗口界面，此时可以接上 USB 键盘(**不要在启动之前接 USB 键盘，否则不能使用**)或者使用屏幕下方的软键盘输入 Linux 命令，你还可以点 Option 菜单中的某些选项进行设置，以改变显示的模式，如图。



## 2.4.8 网络设置

在子类“友善之臂”程序中，点“网络设置”图标打开相应的界面，如图：



在这里，你可以进行常见的网络参数设置：

- 静态的 IP 地址 – 出厂缺省为 192.168.1.230
- 子网掩码 – 出厂缺省为 255.255.255.0
- 网关 – 出厂缺省为 192.168.1.1



追求卓越 创造精品

TO BE BEST

TO DO GREAT

广州友善之臂计算机科技有限公司

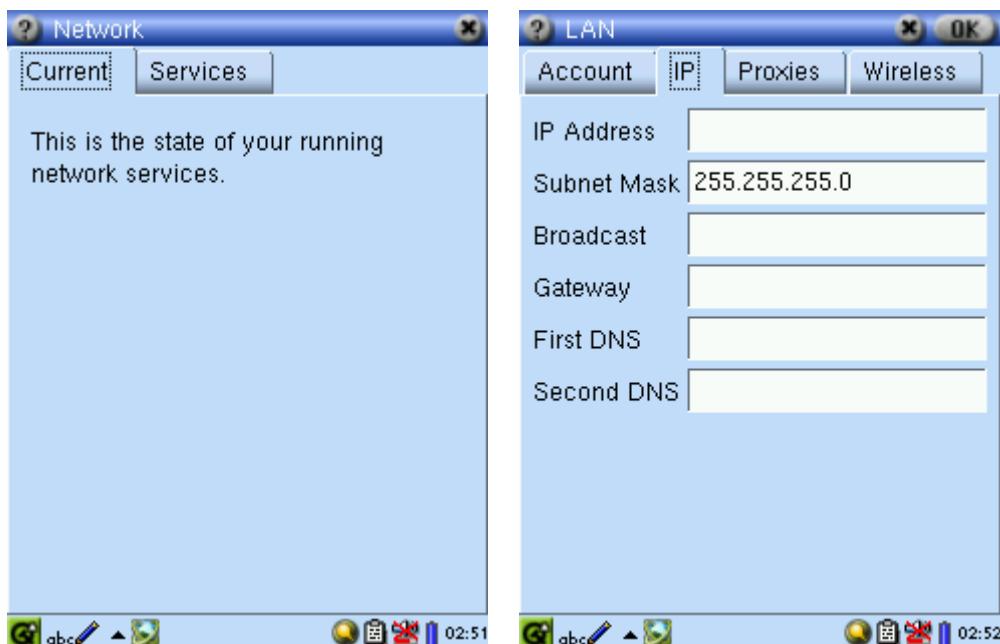
- DNS 解析服务器 IP – 出厂缺省为 192.168.1.1，和网关地址相同
- 网卡的 MAC 地址 – 此地址由驱动程序通过软件设定，是可以修改的，本开发板出厂时所有 MAC 地址都是相同的，为 08:90:91:92:93:94

点“Save”按钮可以保存以上参数，并马上生效，重新启动开发板也可以保留此次的更改设定，与该设置程序相对应的参数文件为/etc/eth0-setting

说明：/etc/eth0-setting 参数文件在系统重装后是不存在的，点“Save”按钮会自动生成；开发板出厂之前需经过测试，因此这个文件是存在的。另外，命令终端的 ifconfig 程序所作的 IP 地址更改对该配置文件没有影响。

其实，Qtopia 本身带有一个网络设置的程序，但配置界面有些复杂，有用户反应其设置也不能有效，为了保持 Qtopia 系统的代码原始性，我们对此并没有深入检测，所以另外自己开发了上面介绍的“网络设置”程序。

Qtopia 自带的网络设置界面如下(注意：其设置不一定有效，本公司不对该程序的设置提供咨询)：



## 2.4.9 Ping 测试

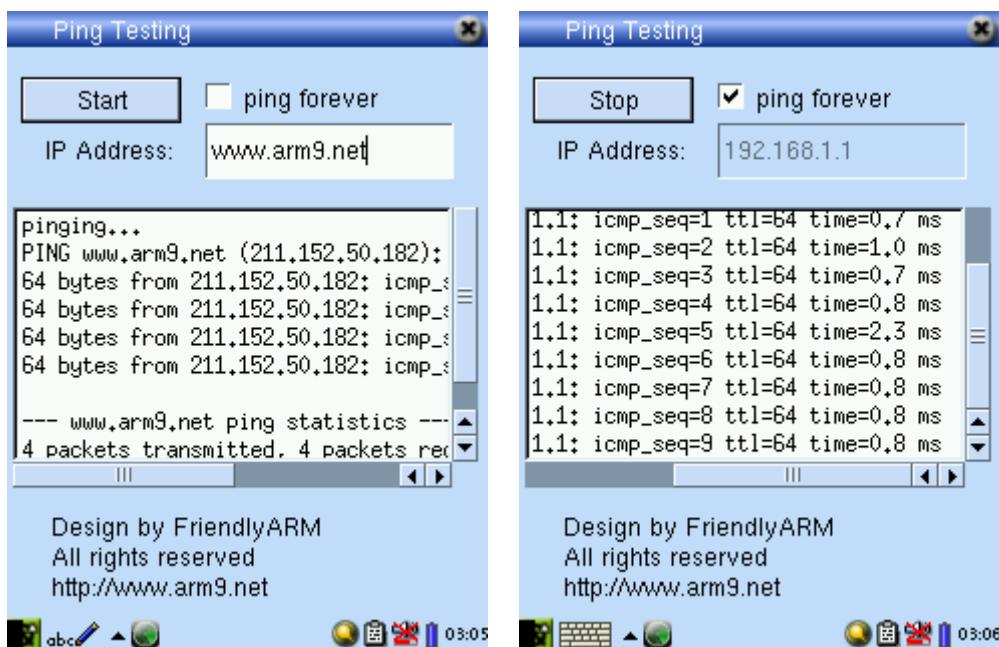
请连接好开发板附带的网线，并设置好有效的网关，DNS 等参数，就可以通过图形界面的 ping 程序来测试网络连通性了。在程序子类“友善之臂”中点“Ping 测试”图标，打开相应界面，如下图：



因为已经设置好了 DNS，所以可支持字符域名和数字 IP 两种方式。默认的 ping 测试次数为 4 次，当勾选上方的“ping forever”后，可以一直 ping，测试结果如下图。

**重要提示：**要 ping 互联网域名，必须要设置好正确有效的网关和 DNS，并且保证你的网络确实可以连通互联网。

点“Start”按钮开始 ping，点“Stop”按钮停止 ping，要关闭“Ping 测试”界面，必须先停止 ping。



说明：ping 是计算机系统中最常见的网络测试工具，不管是各个发行版本的 Linux 系统，还是各种 MS Windows 系统，都可以在命令终端输入“ping”命令。以上的“Ping 测试”程序实际就是调用命令行的 ping，把结果通过图形界面显示出来。

## 2.4.10 浏览器

在“应用程序”中，点“Web Browser”打开浏览器，点开界面下方的软键盘，在界面上方的地址栏中输入一个网址，再点键盘上的“Ret”(回车)按钮，可以打开相应的网站。

说明：本开发板所用的网络浏览器为 Konqueror/Embedded，它是一个开发源代码的浏览器，具体的移植步骤见附录 1 中 Qtopia 的相关脚本(build-all 脚本中包含了移植该浏览器的所有步骤)。



## 2.4.11 LED 测试

在“友善之臂”程序中点“LED 测试”图标，打开如下界面：



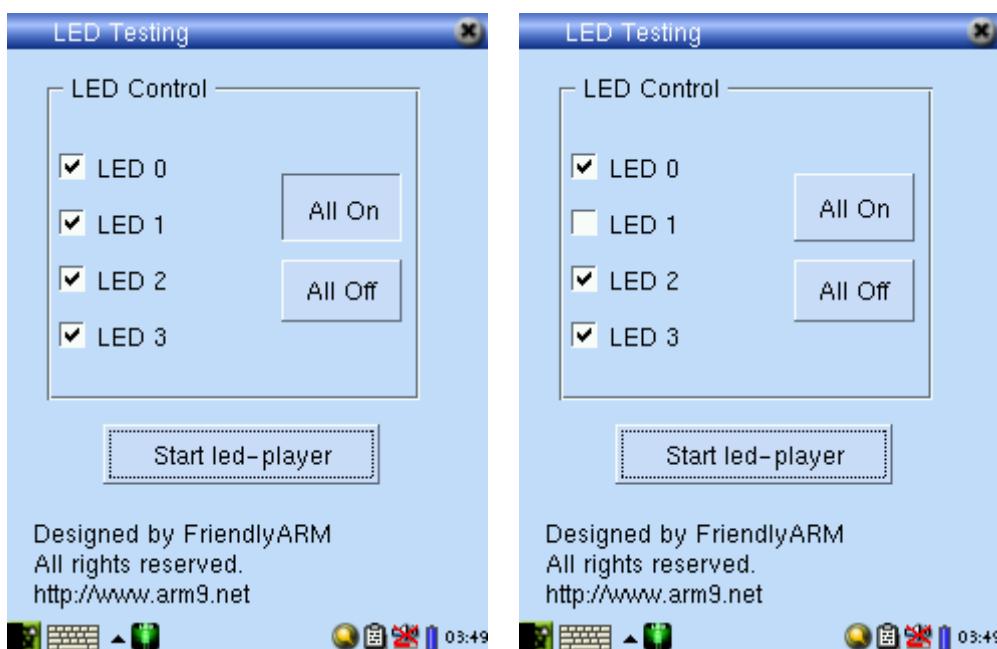
可以看到界面中只有“Stop led-player”按钮有效，这是因为系统启动的时候开启了led-player服务，开机后你所见到的“流水灯”效果就是这个服务控制的，要单独控制某个LED，需要先关闭这个服务，释放LED资源。÷

操作：

点“Stop led-player”按钮，这时它变为“Start led-player”，同时板上所有灯关闭、“LED Control”框中所有按钮由灰色变为有效(如下图)。

这时点“All On”按钮可以点亮所有LED，点“All Off”可以关闭所有LED，勾选左边任意一个框可以点亮相应的LED，取消勾选左边任意一个框可以熄灭相应的LED。

当关闭“LED 测试”界面时，会重新开启 led-player 服务。



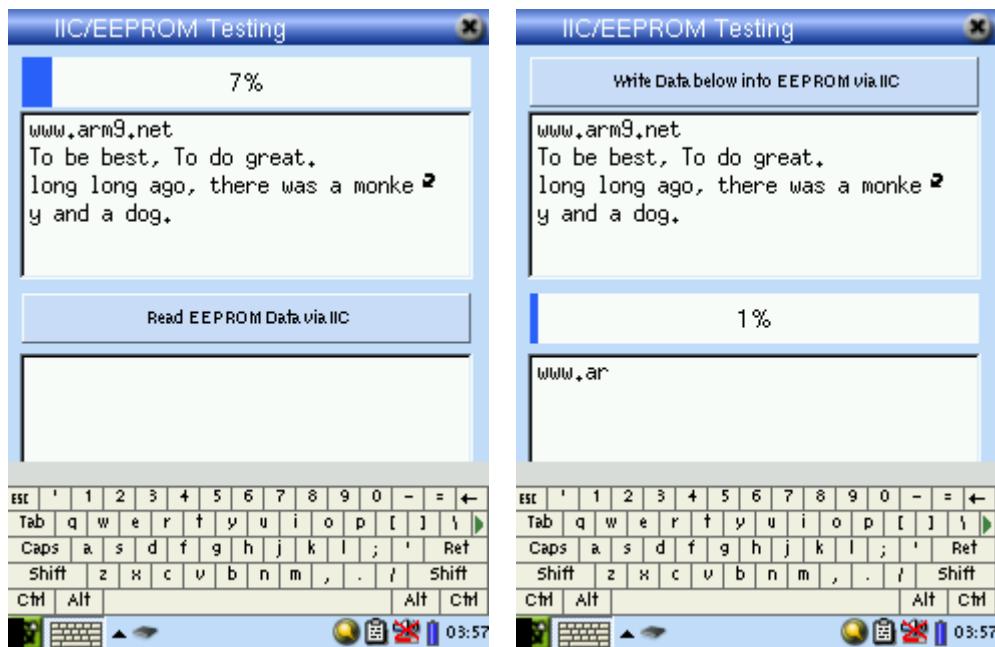
## 2.4.12 EEPROM 读写测试

在“友善之臂”程序中点“I2C-EEPROM 测试”图标，打开如下界面：



由上到下依次可以看到：写 EEPROM 按钮、写入字符编辑区、读取 EEPROM 按钮、读出字符编辑区。

点开任务栏上的“软键盘”按钮，在“写入字符编辑区”输入一些 ASC 字符，点“Write Data below into EEPROM via IIC”按钮，这时该按钮变为进度条，并指示正在写入的进度；点“Read EEPROM Data via IIC”按钮，这时按钮变为进度条，并指示正在读取的进度。如图。





追求卓越 创造精品

TO BE BEST

TO DO GREAT

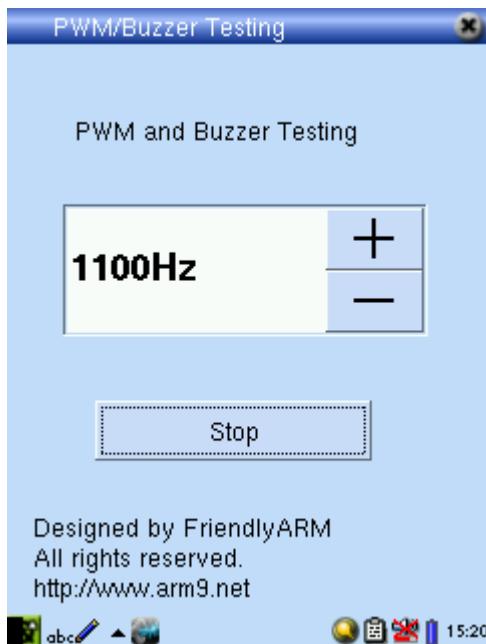
广州友善之臂计算机科技有限公司

## 2.4.13 PWM 控制蜂鸣器测试

在“友善之臂”程序中点“PWM-蜂鸣器测试”图标，打开如下界面：



程序中默认的 PWM 输出为 1000Hz，点“Start”按钮开始驱动蜂鸣器发声，此时可以通过点击“+”或者“-”按钮改变 PWM 输出的频率，同时也可以听到蜂鸣器输出声音的改变。点“Stop”按钮中止 PWM 输出。



## 2.4.14 触摸笔测试

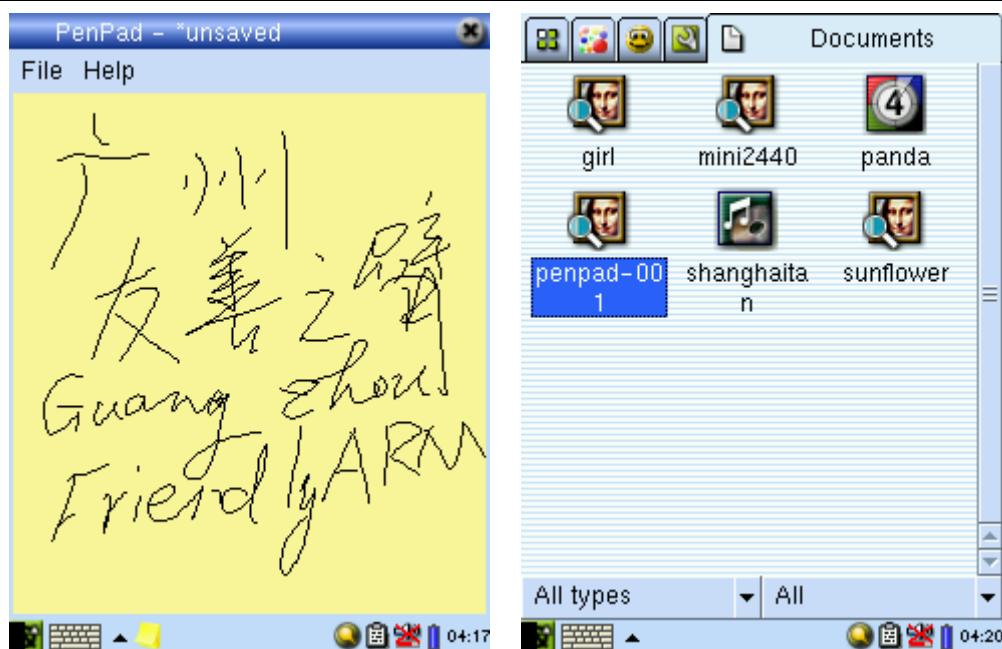
要测试校正后触摸笔的准确性，可以在 LCD 上任意地方画线，对比笔针所到之处画线是否有偏移，是否有抖动，这可以通过“随手写”程序来测试。

在“友善之臂”程序组中点“随手写”图标，打开如下界面：



“随手写”是由友善之臂开发的一个简易画图程序，打开后它出现一个浅黄色的画图区，用触摸笔可以在上面任意涂画(画笔颜色为黑色，宽度为 1 pixel); 点 File->Save 可以把所画保存为 png 格式的图片文件(保存位置：“Documents”，在板子中的位置：/Documents/image/png/)，文件以 001 起始，总共可以保存 999 个文件。

可以看到书写比较流畅，也没有毛刺出现，这说明触摸笔是很准确的。



## 2.4.15 条码扫描

本开发板支持 USB 接口的条码扫描器(英文名称：Barcode Scanner)，它其实是一个 HID 设备，类似 USB 键盘。因此，在任何适合 USB 键盘输入的地方，都可以直接使用这种条码扫描器。

需要注意的是：目前必须在图形界面启动之后插入 USB 扫描器才能正常使用，不能在开机之前就插入。

接上 USB 扫描器，打开“应用程序”中的“文本编辑器”，使用扫描器扫描一些条码(诸如快递单上、杂志等很多商品上都可以找到)，可以看到条码数字被准确扫描到编辑器中了。

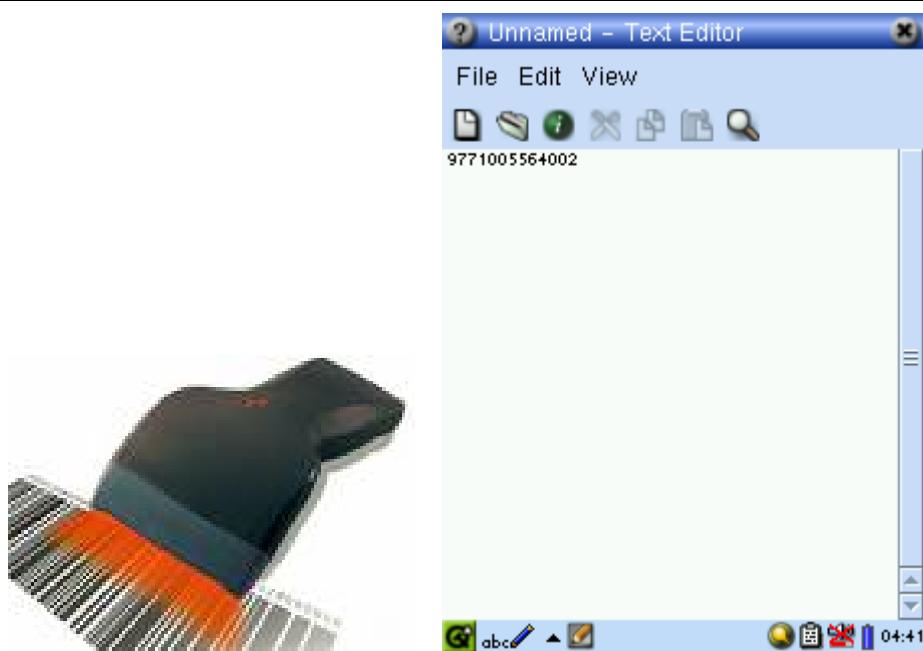


追求卓越 创造精品

TO BE BEST

TO DO GREAT

广州友善之臂计算机科技有限公司



## 2.4.16 语言设置

在“设置”程序组中，找到“语言”图标，点击打开它，可以看到各种语言选择列表，选取相应的语种，如图，点“OK”返回。

说明：因为语言字体文件比较大，本开发板目前只安装了简体中文和英文两种字体，因此只支持这两种语言；另外，本开发板的中文版本只是翻译了程序名称。



## 2.4.17 屏幕旋转

在“设置”程序组点“旋转”图标，进入相应的界面，如图，你可以选择 4 个界面旋转方向，如图。



选择好你想要的方向，点“OK”返回，即可看到结果，如图。

说明：有时需要重新启动 Qtopia 系统才可以看到旋转的效果，因为这个程序是 Qtopia 自带的，为保持其代码原始性，我们没有去修改它。另外，旋转的效果完全是 Qtopia 软件实现的，与 LCD 的底层驱动无关。



## 2.4.18 关于关机和亮度调节

在“设置”程序组中，有个“关机”图标，打开它，如图，其中有四个关机选项：

Shutdown 和 Reboot 是需要硬件支持的，因为本开发板并无相应的硬件电路，因此这两个选择是无效的；

Restart Server 是指重新启动 Qtopia 图形系统，此时并不会影响到基本的 Linux 系统；

Terminates Server 是指关闭 Qtopia 图形系统，点击它之后图形界面就完全失效了，此时屏幕上的显示是遗留在内存中的数据，并不是有效的图形系统界面。



在“设置”程序组中，还有个“亮度和电源”的图标，因为本开发板并没有相应的电源管理电路，因此它们也是无效的。

## 2.5 Linux 之非图形界面系统测试与使用说明

本开发板提供了两种 Linux 系统以方便用户测试使用，这两种系统所使用的内核文件是完全相同的，只是文件系统的内容不同：

一种是非图形界面的系统，它的安装文件对应于光盘中的 root\_default.img，它的源文件压缩包为 root\_default.tgz(注意：并非源代码)。该系统基于 Linux 工具集 busybox-1.2 和 Linux glibc-2.3.2 库文件制作而成，另外我们还专门针对本开发板集成了很多基于命令行的实用测试工具，如 mp3 播放，图片播放，按键测试，LED 测试等等。正因为如此，这样的系统可以做的比较小型化，系统剩余的空间(即 Flash 剩余空间)也就比较大，用户也可以根据自己的需要，灵活的删除、添加一些程序，以实现灵活的定制化，下面的内容是详细的使用说明。

另一种是嵌入式图形界面 Qtopia 系统(PDA 版)，它其实是基于非图形界面系统的扩



追求卓越 创造精品

TO BE BEST

TO DO GREAT

广州友善之臂计算机科技有限公司

展，使用的 glibc 基本库都是相同的。Qtopia 本身有很多实用的小程序，根据用途不同，这些小程序被分为三类：应用程序、游戏和设置，另外还有一个“文档”。同样地，我们针对本开发板也开发了一些 Qtopia 的实用小程序，例如 USB 摄像头拍照、PWM 控制蜂鸣器、EEPROM 读写、LED 控制等，它们都是基于 3.5” LCD 的程序，我们把这些自主开发的程序归为一类“友善之臂”。

对于非图形界面系统，开机后的显示界面：

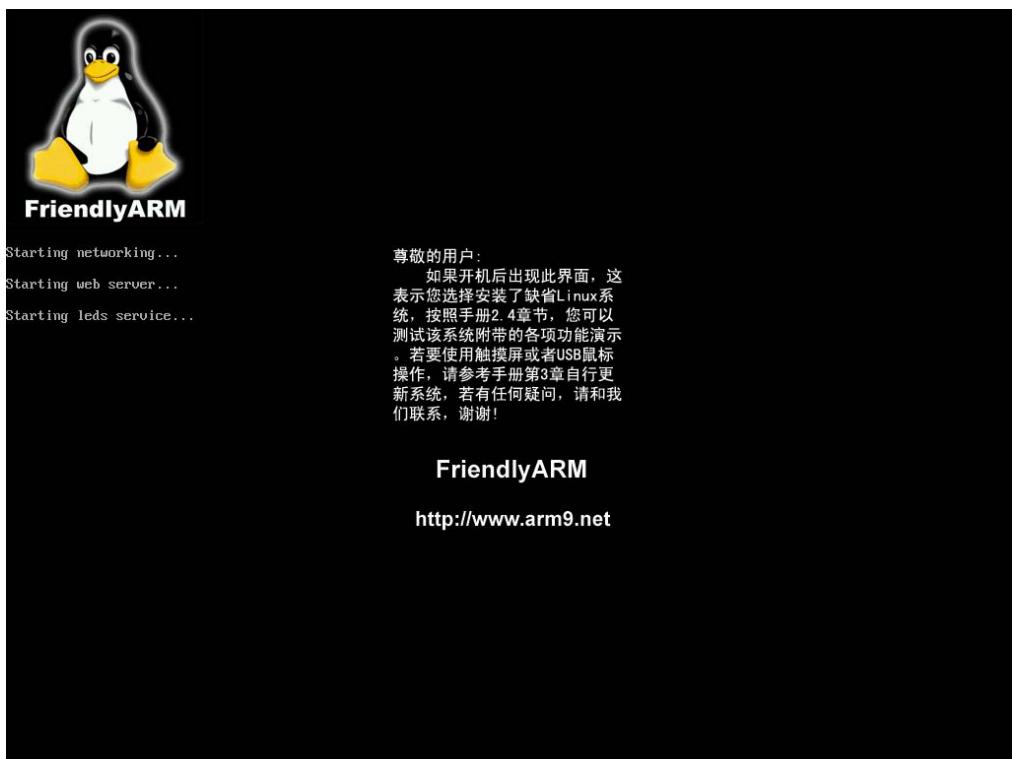
使用 3.5 寸屏时，开机后的显示界面：



使用 7 寸屏时，开机后的显示界面：



使用 VGA 输出时，开机后的显示界面：



## 2.5.1 播放 mp3

名称： madplay	备注
源代码或者源代码包的名称	madplay.tgz
源代码或者源代码包的位置	linux\porting sample
其他：编译移植方法见 6.5	

madplay 是我们移植的一个基于控制台下的 mp3 播放器。它有多种播放控制模式，最简单的使用方法是：

**#madplay your.mp3**

该命令将以缺省模式播放 your.mp3 文件(开发板中并无 your.mp3 文件，这里只是举例说明)。

**注意：系统启动后将会自动播放“/”目录下的 shanghaiwan.mp3 文件。**

可以运行“madplay -h”查看其使用帮助。

## 2.5.2 如何中止程序的运行

要中止程序的运行，可以在终端控制台下同时按下 **Ctrl+c**，注意：先按 **Ctrl**，不要放开，再按下 **c** 键即可。

例如：我们刚刚使用 madplay 命令播放了 mp3，如果要中止这个程序的运行，可以按下 **Ctrl+c** 键。

另外，如果程序是在后台运行，可以使用 **kill** 命令杀掉该进程

### 2.5.3 使用优盘/移动硬盘

在开发板 Linux 系统中，移动存储设备对应的设备文件是 **/dev/scsi/host1/bus0/target0/lun0/part\***，为了能够和标准 Linux 系统中的优盘设备名兼容，这里创建一个连接

```
#ln -s /dev/scsi/host1/bus0/target0/lun0/part1 /dev/sda1
```

**该命令已经写入/etc/init.d/rcS 启动脚本中**，因此系统启动后就已经可以直接使用 **/dev/sda1** 了。当优盘接入 USB HOST 口(或者接入 USB HUB)，可以用以下命令把优盘设备挂接到系统的某一个目录上：

```
#mount /dev/sda1 /mnt
```

表示把优盘挂接到了 **/mnt** 目录。

一般情况下，当优盘插入以后，您将在串口终端看到如图所示信息，以下命令根据信息提示直接把优盘设备挂接到 **/mnt** 目录：

```
#mount /dev/scsi/host1/bus0/target0/lun0/part1 /mnt
```

The screenshot shows a terminal window titled "超级终端". The terminal output is as follows:

```
-sh: can't access tty; job control turned off
[root@FriendlyARM ~]# usb 1-1: new full speed USB device using s3c2410-ohci and
address 2
scsi0 : SCSI emulation for USB Mass Storage devices

[root@FriendlyARM ~]# Vendor: Netac      Model: OnlyDisk          Rev: 4.12
      Type: Direct-Access           ANSI SCSI revision: 02
SCSI device sda: 257824 512-byte hdwr sectors (132 MB)
sda: Write Protect is off
sda: assuming drive cache: write through
SCSI device sda: 257824 512-byte hdwr sectors (132 MB)
sda: Write Protect is off
sda: assuming drive cache: write through
  /dev/scsi/host0/bus0/target0/lun0:<7>usb-storage: queuecommand called
p1
Attached scsi removable disk sda at scsi0, channel 0, id 0, lun 0

[root@FriendlyARM ~]# mount /dev/scsi/host0/bus0/target0/lun0/part1 /mnt/
[root@FriendlyARM ~]# ls /mnt/
dsc00034.jpg dsc00039.jpg hope.mp3 theday~1.mp3
dsc00035.jpg dsc00040.jpg ibelie~1.mp3 tll.mp3
dsc00037.jpg dsc00044.jpg tearin~1.mp3
[root@FriendlyARM ~]# 使用mount命令挂接优盘设备到mnt目录，并查看其中的内容
```

The terminal window has a blue border around the command and its output. The text "插入优盘后跳出的信息" (Information displayed after inserting the USB drive) is highlighted in red over the first few lines of the terminal output. The text "使用mount命令挂接优盘设备到mnt目录，并查看其中的内容" (Using the mount command to mount the USB drive to the /mnt directory and viewing its contents) is also highlighted in red at the bottom of the terminal output.

## 2.5.4 使用 SD 卡

SD/MMC 卡驱动源代码目录(在 2.6.13 内核中仅支持 2G 容量以内的 SD 卡):  
 kernel-2.6.13/drivers/mmc

SD 或者 MMC 卡底使用方法和优盘十分类似，插入存储卡之后，一般会跳出如图所示的信息，同时会在/dev/mmc 目录下出现对应的设备，使用 mount 命令挂接 SD/MMC 卡设备到/mnt 目录，就可以对它进行操作了。

**注意：本开发板的 Linux 系统目前最大可以支持 2G 容量的 SD 卡。**

#mount /dev/mmc/disc0/part1 /mnt

The screenshot shows a terminal window titled "超级终端". The terminal output is as follows:

```

00-2003 Robert Leslie et al.

Please press Enter to activate this console. boa: server version Boa/0.94.13
[01/Jan/1970:00:00:05 +0000] boa: server built Feb 28 2004 at 21:47:23.
[01/Jan/1970:00:00:05 +0000] boa: starting server pid=775, port 80
    Title: 上海滩
    Artist: 叶丽仪
    Year: 2000
    Genre: Goa

-sh: can't access tty; job control turned off
[root@FriendlyARM ~]# MMC: sd_app_op_cond: at least one card is busy - trying again.
MMC: sd_app_op_cond: at least one card is busy - trying again.
MMC: sd_app_op_cond: at least one card is busy - trying again.
mmcblk0: mmc0:c734 SD016 14400KiB
/dev/mmcblk0:<7>MMC: starting cmd 37 arg c7340000 flags 00000009
p1                                         插入SD卡之后跳出的信息
MMC: sd_app_op_cond: at least one card is busy - trying again.

[root@FriendlyARM ~]# mount /dev/mmcblk0/part1 /mnt/
[root@FriendlyARM ~]# ls /mnt/
hero.mp3  hope.mp3  使用mount命令挂接SD卡设备到mnt目录，并浏览其中的内容
[root@FriendlyARM ~]#

```

The terminal window has a blue box around the line "MMC: sd\_app\_op\_cond: at least one card is busy - trying again." and another blue box around the command "mount /dev/mmcblk0/part1 /mnt/". A red box highlights the file names "hero.mp3" and "hope.mp3" in the "/mnt/" directory.

## 2.5.5 使用 USB 摄像头抓图

MINI2440 支持市面上常见的使用中芯微芯片的 USB 摄像头，当把摄像头插入 USB 接口之后，一般会跳如图信息，同时在 dev 目录下出现相应的设备名：/dev/v4l/video0，使用 spcacat 程序可以直接抓取摄像头采集到的图象。

```
[root@FriendlyARM /]# usb 1-1: new full speed USB device using s3c2410-ohci and
address 2
drivers/usb/media/gspca/gspca_core.c: USB SPCA5XX camera found. SONIX JPEG (sn9c
1xx)
```

插入USB摄像头出现的信息

```
[root@FriendlyARM /]# ls /dev/v4l/video0
/dev/v4l/video0
```

此时可以看到在dev目录下增加了/dev/v4l/video0

```
[root@FriendlyARM /]#
```

使用如下命令抓取图象：

```
#spcacat -p 100ms -N 5
```

该命令将每隔 100ms 抓图一次，连续抓图 5 个，之后将在当前目录生成以顺序时间命名的 5 个 jpg 文件：

The screenshot shows a Telnet session connected to 192.168.1.230. The terminal window title is "Telnet 192.168.1.230". The session output is as follows:

```

picture jpeg 06:25:2007-00:14:23-P0004.jpg
GRABBER going out !!!!!
unmapping frame buffer
close video_device
freeing output buffer 0
freeing output buffer 1
freeing output buffer 2
freeing output buffer 3
[root@FriendlyARM /]# ls
06:25:2007-00:14:22-P0000.jpg lost+found
06:25:2007-00:14:22-P0001.jpg mnt
06:25:2007-00:14:23-P0002.jpg opt
06:25:2007-00:14:23-P0003.jpg proc
06:25:2007-00:14:23-P0004.jpg sbin
bin shanghaitan.mp3
dev tmp
etc usr
home var
lib www
linuxrc
[root@FriendlyARM /]#
[root@FriendlyARM /]#
```

Annotations in red text highlight specific parts of the output:

- "抓图过程" (Capture Process) points to the sequence of commands from "GRABBER going out" to "freeing output buffer 3".
- "抓图生成的5个文件" (5 captured files generated) points to the list of files in the current directory: 06:25:2007-00:14:22-P0000.jpg, 06:25:2007-00:14:22-P0001.jpg, 06:25:2007-00:14:23-P0002.jpg, 06:25:2007-00:14:23-P0003.jpg, and 06:25:2007-00:14:23-P0004.jpg.

经试验，要抓取一幅质量较好的图片的命令如下：

```
Spcacat -s 384x288 -p 100ms -N2 -o
```

运行该命令，将在当前目录下抓图生成 SpcaPict.jpg 文件

## 2.5.6 如何通过串口与 PC 互相传送文件

当通过串口终端登录系统之后，可以使用 **rz** 或者 **sz** 命令通过串口与 PC 互相传送文件，具体操作如下。

### (1) 使用 sz 向 PC 发送文件

在超级终端窗口中，点鼠标右键，在弹出的菜单中选择“接收文件”开始设置接收文



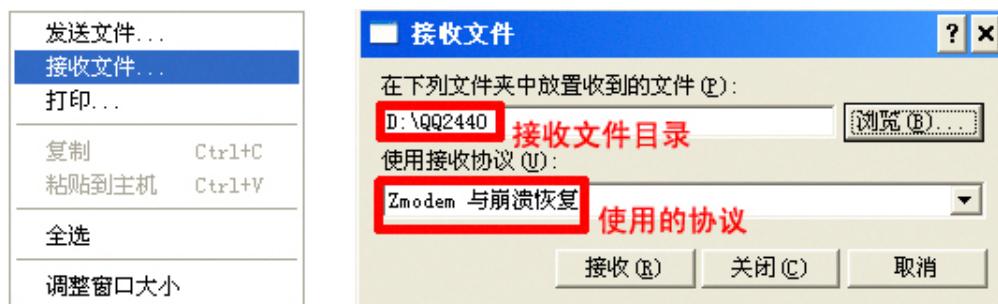
追求卓越 创造精品

TO BE BEST

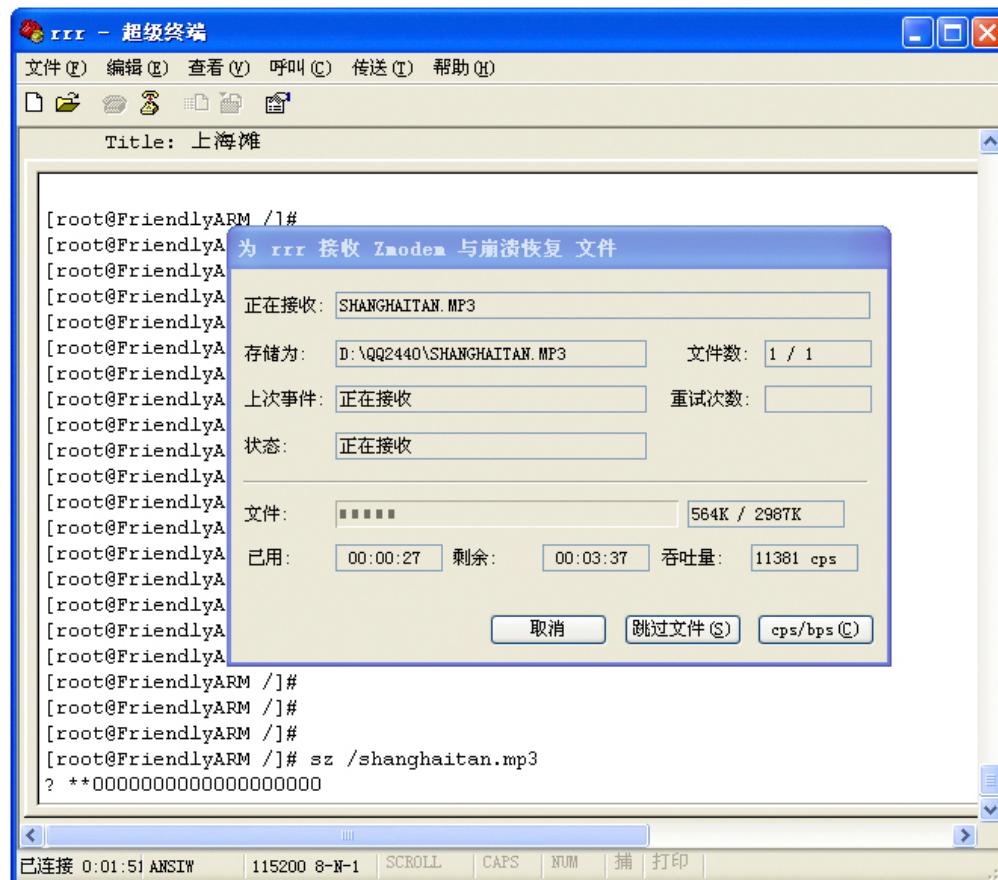
TO DO GREAT

广州友善之臂计算机科技有限公司

件目录和协议，如图所示。



然后在终端的命令行输入“`sz /shanghaitan.mp3`”命令，开始向 PC 传送位于“/”目录的 `shanghaitan.mp3` 文件(或者其他文件，改一下路径和文件名就可以了)，因为该文件比较大，所以需要多等几分钟，发送完毕，系统会自动保存文件到您设置的目录里面，如图。



## (2) 使用 `rz` 命令下载文件到开发板

在串口终端输入“`rz`”命令，开始接收从 PC 传过来的文件。

然后在超级终端窗口中，点鼠标右键，在弹出的菜单中选择“发送文件”，设置好要发送的文件和使用的协议，如图所示，开始向开发板发送文件。



追求卓越 创造精品

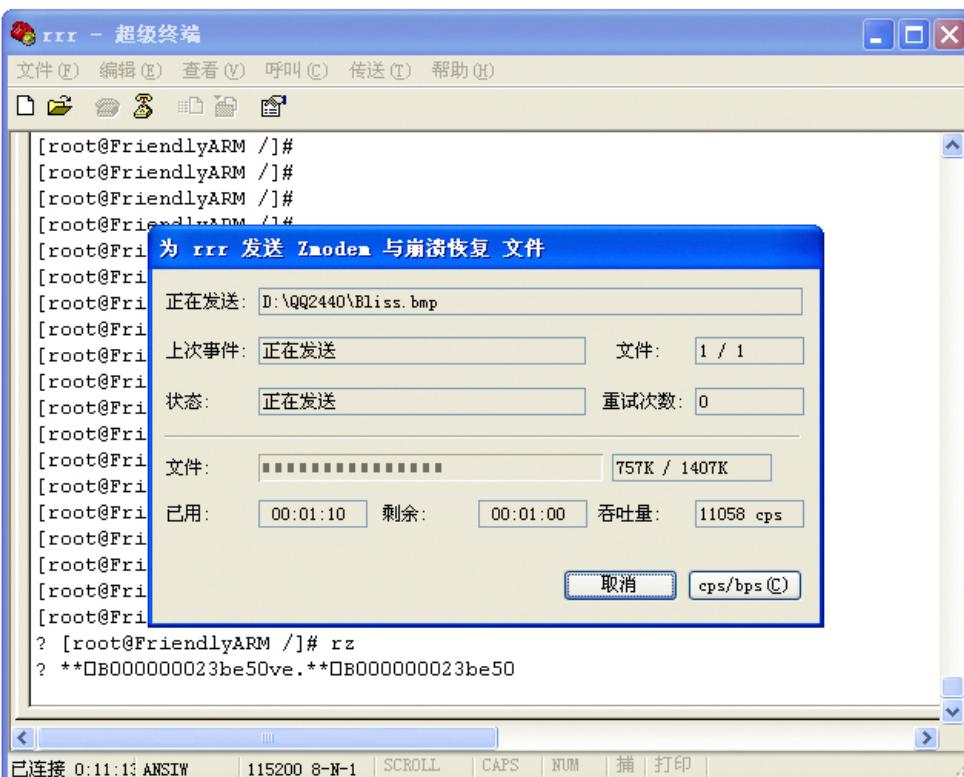
TO BE BEST

TO DO GREAT

广州友善之臂计算机科技有限公司



点“发送”，开发板开始接收文件，如图所示。



接送完毕，将会在当前目录下得到同样文件名的文件，您可以使用 md5sum 命令验证该文件是否和源文件相同。

```
[root@FriendlyARM /]# md5sum Bliss.bmp
216d20df62af34d39089e066d1d4b3af  Bliss.bmp
```

## 2.5.7 如何通过网络远程控制显示 USB 摄像头

系统开机后，并且 LED 计数器显示跳动正常，就可以使用 IE 或者其他浏览器来浏览存放于板子中的网页了，在浏览器的地址栏中输入：



追求卓越 创造精品

TO BE BEST

TO DO GREAT

广州友善之臂计算机科技有限公司

<http://192.168.1.230>

此时会跳出如图所示页面，这是板子自带的 web server 服务器的一个测试页面。



这时点“USB 摄像头远程控制与显示”项，出现操作界面，如图所示。



按照网页上的说明进行操作：

请连接好板上的USB摄像头，然后点右边的“确定”按钮开始浏览

确定

提示：请先安装Java插件方能正常显示，安装程序位于光盘的“Windows平台工具\java”目录。

这时，您就可以在网页中浏览到 USB 摄像头采集的动态图象了。



## 2.5.8 如何控制板上的 LED

名称: led-player	备注
源代码或者源代码包的名称	led-player.c
源代码或者源代码包的位置	linux示例代码\examples\led-player
其他: 源代码原理详细说明详见 6.2.6 的	

名称: leds.cgi	备注
源代码或者源代码包的名称	Leds.cgi
源代码或者源代码包的位置	位于开发板中的\www 目录中
说明:	
leds.cgi 是一个 shell 脚本文件，它并不是二进制程序，该脚本通过 leds.html 被调用，其中使用的是最普通的网页设计技术。	
解压光盘中的 root_default.tgz 也可以在其中的 www 目录得到 leds.cgi 和 leds.html 文件，它们都是脚本，本身就是源代码，使用任何文本编辑器(如 Windows 的“记事本”)都可以打开。	

说明: Led-player 和通过网页控制 LED 均为友善之臂早期为 SBC2410 开发的简易示例程序，因其硬件无关性，所以可以方便的移植到其他系统。目前市面上有的书籍，部分



追求卓越 创造精品

TO BE BEST

TO DO GREAT

广州友善之臂计算机科技有限公司

2410/2440 开发板厂商均采用了这个典型的管道应用示例。

### (1) LED 服务器

开机进入系统后，将会自动运行一个 LED 服务程序(`/etc/rc.d/init.d/leds`)，它其实是调用了 `led-player` 的一个脚本，`led-player` 开始运行后，将会在/tmp 目录下创建一个 `led-control` 管道文件，向该管道发送不同的参数可以改变 led 的闪烁模式：

**#echo 0 0.2 >/tmp/led-control**

运行该命令后，4 个用户 led 将会以每个间隔 0.2 秒的时间运行跑马灯。

**#echo 1 0.2 >/tmp/led-control**

运行该命令后，4 个用户 led 将会以间隔 0.2 秒的时间运行累加器。

**#/etc/rc.d/init.d/leds stop**

运行该命令后，4 个用户 led 将会停止闪动。

**#/etc/rc.d/init.d/leds start**

运行该命令后，4 个用户 led 将会重新开始闪动。

### (2) 单独控制 LED

`/bin/leds` 是一个可以控制单个 led 的实用程序，要使用 leds 必须先停止 led-player，如下命令：

**#/etc/rc.d/init.d/leds stop**

该命令将停止 led-player 对 led 的操纵。led 的使用方法如下：

[root@fa /]# led

Usage: leds led\_no 0|1

led\_no 是要操作的 led(可为 0, 1, 2, 3)，0 和 1 分别代表关闭和点亮。

**#led 2 1**

将点亮 LED3

## 2.5.9 测试板上的按键

在命令行输入“**buttons**”命令，然后按开发板上的按键，可以显示对应的键值，如图

```

Please press Enter to activate this console.
[root@FriendlyARM /]# 7316 frames decoded (0:03:11.1), +0.0 dB peak amplitude, 6
clipped samples

[root@FriendlyARM /]# buttons
key value = 0x01
key value = 0x81
key value = 0x02
key value = 0x02
key value = 0x02
key value = 0x82
key value = 0x03
key value = 0x83
key value = 0x06
key value = 0x86
key value = 0x05
key value = 0x85
key value = 0x04
key value = 0x84
-

```

已连接 0:37:21 ANSIW | 115200 8-N-1 | SCROLL | CAPS | NUM | 捕 | 打印 |

## 2.5.10 串口 2 和 3 的测试

**说明:** armcomtest 是友善之臂为了方便测试而开发的 linux 下的简易实用串口终端程序, 它使用标准的系统调用, 和硬件无关, 该程序可以在大部分 armv4 平台系统上运行使用, 该程序不提供源代码。

**提示:**

串口驱动程序的位置:

**kernel-2.6.13/drivers/serial/s3c2410.c**

系统启动后, 串口 0,1,2 对应的设备名分别为**/dev/tts/0,1,2**

测试串口 2 需要借助另一台带有串口的 PC, 使用我们提供的串口线和扩展小板(**选购配件**), 连接好 COM2 和另一台 PC 的串口, 并如前所述设置该 PC 的超级终端为波特率 115200, 无流控制, 其他默认。

在命令行下输入:

**#armcomtest -d /dev/tts/1 -o**

这时如果输入字符会在另一台 PC 的超级终端出现, 反之亦然。

如果要测试串口 3, 则需要连接扩展小板的 COM3, 并在命令行输入:

**#armcomtest -d /dev/tts/2 -o**

下面是测试时的界面:

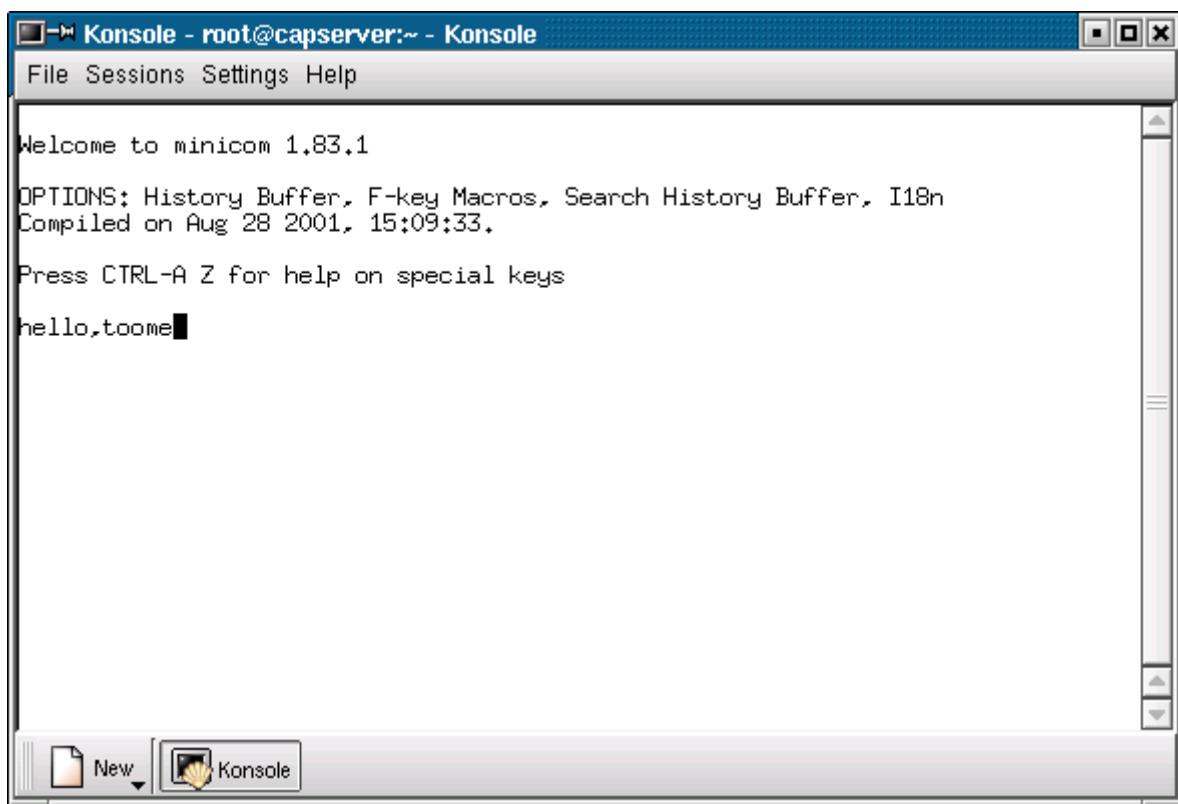
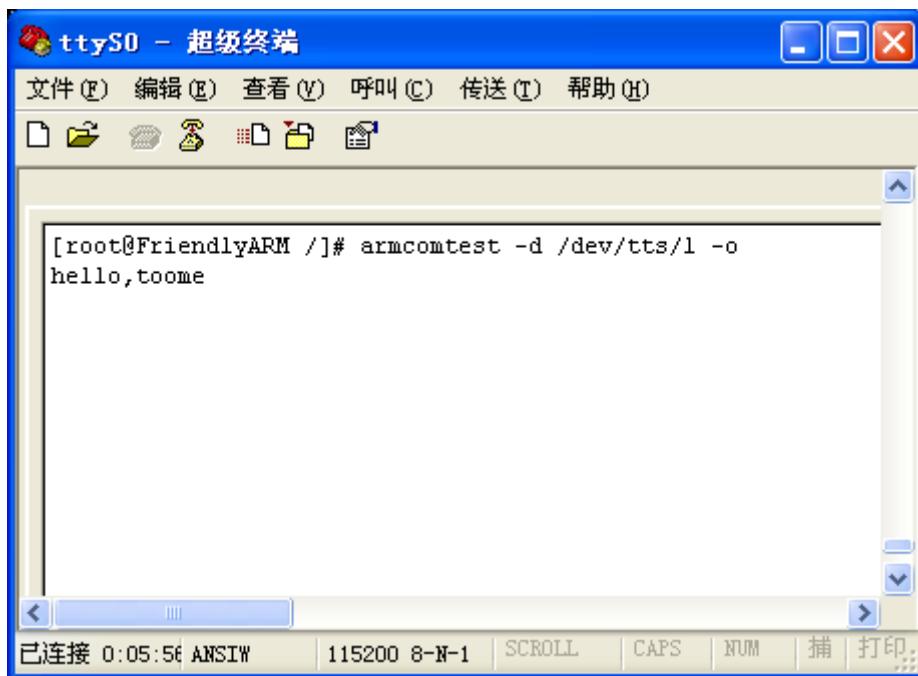


追求卓越 创造精品

TO BE BEST

TO DO GREAT

广州友善之臂计算机科技有限公司



## 2.5.11 测试蜂鸣器

提示:

蜂鸣器驱动程序的位置:

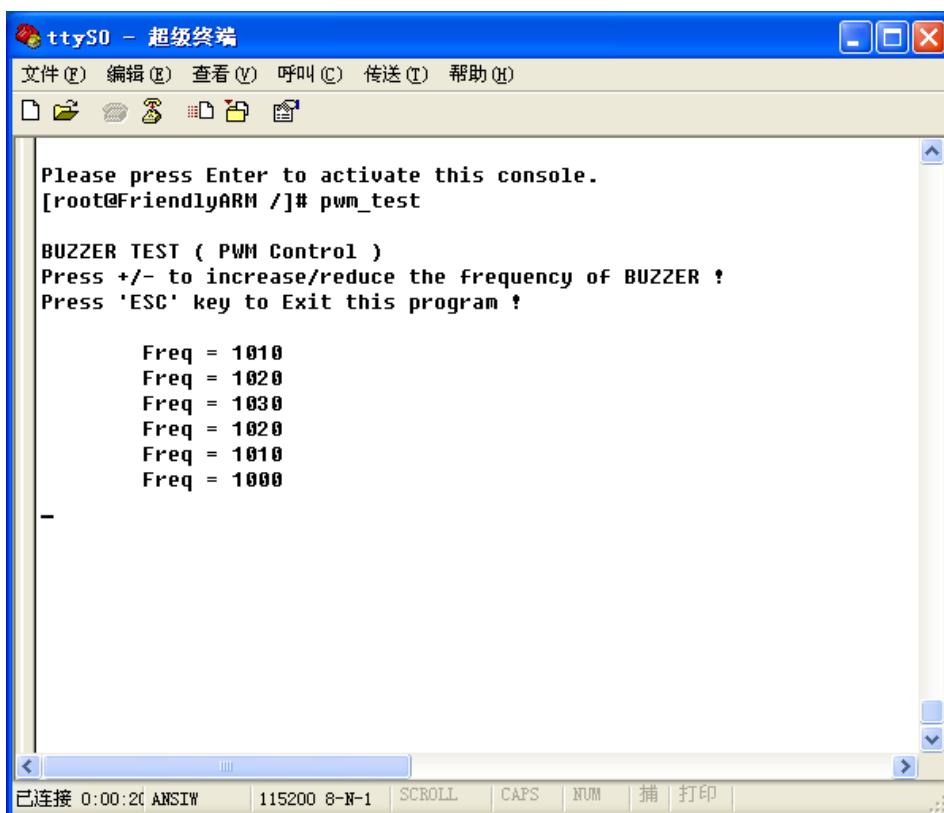
**kernel-2.6.13/drivers/char/qq2440\_pwm.c**

因为 mini2440 和 QQ2440 使用相同的硬件资源，因此它们的驱动是完全相同的。

在命令行种输入: **pwm\_test**

可以听到蜂鸣器的发出的声音，按“+”或者“-”可以改变输出的频率，如图。

按 ESC 键中止该测试。



## 2.5.12 控制 LCD 的背光

提示:

LCD 的背光控制通过一个简单的字符设备驱动驱动程序来实现，它的源代码的位置:

**kernel-2.6.13/drivers/char/mini2440\_backlight.c**

在命令行种输入: **bl 1** 或者 **bl 0** 可以控制 LCD 背光的开和关。

[root@FriendlyARM /]# bl 0

close LCD backlight

[root@FriendlyARM /]# bl 1

open LCD backlight

[root@FriendlyARM /]#

## 2.5.13 测试 I2C—EEPROM

**提示：**

I2C 的驱动程序位置：

**kernel-2.6.13/drivers/i2c/busses/i2c-s3c2410.c**

在命令行种输入： i2c -w 可以向板子的 24C08 器件中写入数据（0x00-0xff）

```

[root@FriendlyARM /]#
[root@FriendlyARM /]#
[root@FriendlyARM /]# i2c -w
Open /dev/i2c/0 with 8bit mode
Writing 0x00-0xff into 24C08

0000| 00 01 02 03 04 05 06 07 08 09 0a 0b 0c 0d 0e 0f
0010| 10 11 12 13 14 15 16 17 18 19 1a 1b 1c 1d 1e 1f
0020| 20 21 22 23 24 25 26 27 28 29 2a 2b 2c 2d 2e 2f
0030| 30 31 32 33 34 35 36 37 38 39 3a 3b 3c 3d 3e 3f
0040| 40 41 42 43 44 45 46 47 48 49 4a 4b 4c 4d 4e 4f
0050| 50 51 52 53 54 55 56 57 58 59 5a 5b 5c 5d 5e 5f
0060| 60 61 62 63 64 65 66 67 68 69 6a 6b 6c 6d 6e 6f
0070| 70 71 72 73 74 75 76 77 78 79 7a 7b 7c 7d 7e 7f
0080| 80 81 82 83 84 85 86 87 88 89 8a 8b 8c 8d 8e 8f
0090| 90 91 92 93 94 95 96 97 98 99 9a 9b 9c 9d 9e 9f
00a0| a0 a1 a2 a3 a4 a5 a6 a7 a8 a9 aa ab ac ad ae af
00b0| b0 b1 b2 b3 b4 b5 b6 b7 b8 b9 ba bb bc bd be bf
00c0| c0 c1 c2 c3 c4 c5 c6 c7 c8 c9 ca cb cc cd ce cf
00d0| d0 d1 d2 d3 d4 d5 d6 d7 d8 d9 da db dc dd de df
00e0| e0 e1 e2 e3 e4 e5 e6 e7 e8 e9 ea eb ec ed ee ef
00f0| f0 f1 f2 f3 f4 f5 f6 f7 f8 f9 fa fb fc fd fe ff

[root@FriendlyARM /]#

```

已连接 1:47:54 ANSIW | 115200 8-N-1 | SCROLL | CAPS | NUM | 插 | 打印 |

在命令行中输入： i2c -r 可以从板子的 24C08 器件中读出输出



追求卓越 创造精品

TO BE BEST

TO DO GREAT

广州友善之臂计算机科技有限公司

```
00f0| f0 f1 f2 f3 f4 f5 f6 f7   f8 f9 fa fb fc fd fe ff
[root@FriendlyARM ~]# i2c -r
Open /dev/i2c-0 with 8bit mode
Reading 256 bytes from 0x0
0000| 00 01 02 03 04 05 06 07   08 09 0a 0b 0c 0d 0e 0f
0010| 10 11 12 13 14 15 16 17   18 19 1a 1b 1c 1d 1e 1f
0020| 20 21 22 23 24 25 26 27   28 29 2a 2b 2c 2d 2e 2f
0030| 30 31 32 33 34 35 36 37   38 39 3a 3b 3c 3d 3e 3f
0040| 40 41 42 43 44 45 46 47   48 49 4a 4b 4c 4d 4e 4f
0050| 50 51 52 53 54 55 56 57   58 59 5a 5b 5c 5d 5e 5f
0060| 60 61 62 63 64 65 66 67   68 69 6a 6b 6c 6d 6e 6f
0070| 70 71 72 73 74 75 76 77   78 79 7a 7b 7c 7d 7e 7f
0080| 80 81 82 83 84 85 86 87   88 89 8a 8b 8c 8d 8e 8f
0090| 90 91 92 93 94 95 96 97   98 99 9a 9b 9c 9d 9e 9f
00a0| a0 a1 a2 a3 a4 a5 a6 a7   a8 a9 aa ab ac ad ae af
00b0| b0 b1 b2 b3 b4 b5 b6 b7   b8 b9 ba bb bc bd be bf
00c0| c0 c1 c2 c3 c4 c5 c6 c7   c8 c9 ca cb cc cd ce cf
00d0| d0 d1 d2 d3 d4 d5 d6 d7   d8 d9 da db dc dd de df
00e0| e0 e1 e2 e3 e4 e5 e6 e7   e8 e9 ea eb ec ed ee ef
00f0| f0 f1 f2 f3 f4 f5 f6 f7   f8 f9 fa fb fc fd fe ff
[root@FriendlyARM ~]#
```

已连接 1:48:14 ANSIW | 115200 8-N-1 | SCROLL | CAPS | NUM | 插 | 打印 |

## 2.5.14 使用 telnet 上 bbs

telnet 是一个经常被使用的远程登录工具，使用 telnet 功能，可以从开发板登录到其他提供了 telnet 服务器的主机，如果您接入开发板的网络可以上互联网，则可以通过 telnet 命令登录外部的 bbs。

首先，确认开发板的 IP 地址是否为 192.168.1.230，并且是否和局域网内其他主机相通，如图为成功的信息。

The screenshot shows a terminal window titled "超级终端" (Super Terminal) with the following content:

```
-sh: can't access tty; job control turned off
[root@FriendlyARM /]# ifconfig
eth0      Link encap:Ethernet HWaddr 08:00:3E:26:0A:5B
          inet addr:192.168.1.230  Bcast:192.168.1.255  Mask:255.255.255.0
                  UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
                  RX packets:14 errors:0 dropped:0 overruns:0 frame:0
                  TX packets:0 errors:0 dropped:0 overruns:0 carrier:0
                  collisions:0 txqueuelen:1000
                  RX bytes:1193 (1.1 KiB)  TX bytes:0 (0.0 B)
                  Interrupt:53 Base address:0x300

lo       Link encap:Local Loopback
          inet addr:127.0.0.1  Mask:255.0.0.0
          UP LOOPBACK RUNNING  MTU:16436  Metric:1
          RX packets:0 errors:0 dropped:0 overruns:0 frame:0
          TX packets:0 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:0
          RX bytes:0 (0.0 B)  TX bytes:0 (0.0 B)

[root@FriendlyARM /]# ping 192.168.1.1
PING 192.168.1.1 (192.168.1.1): 56 data bytes
64 bytes from 192.168.1.1: icmp_seq=0 ttl=64 time=6.5 ms
64 bytes from 192.168.1.1: icmp_seq=1 ttl=64 time=0.9 ms
```

Annotations in red highlight the IP address (192.168.1.230) in the ifconfig output and the ping results.

然后设置路由 IP: **route add default gw 192.168.1.1**

最后使用 telnet 命令登录您要登录的主机，在此登录的是华南木棉 bbs。

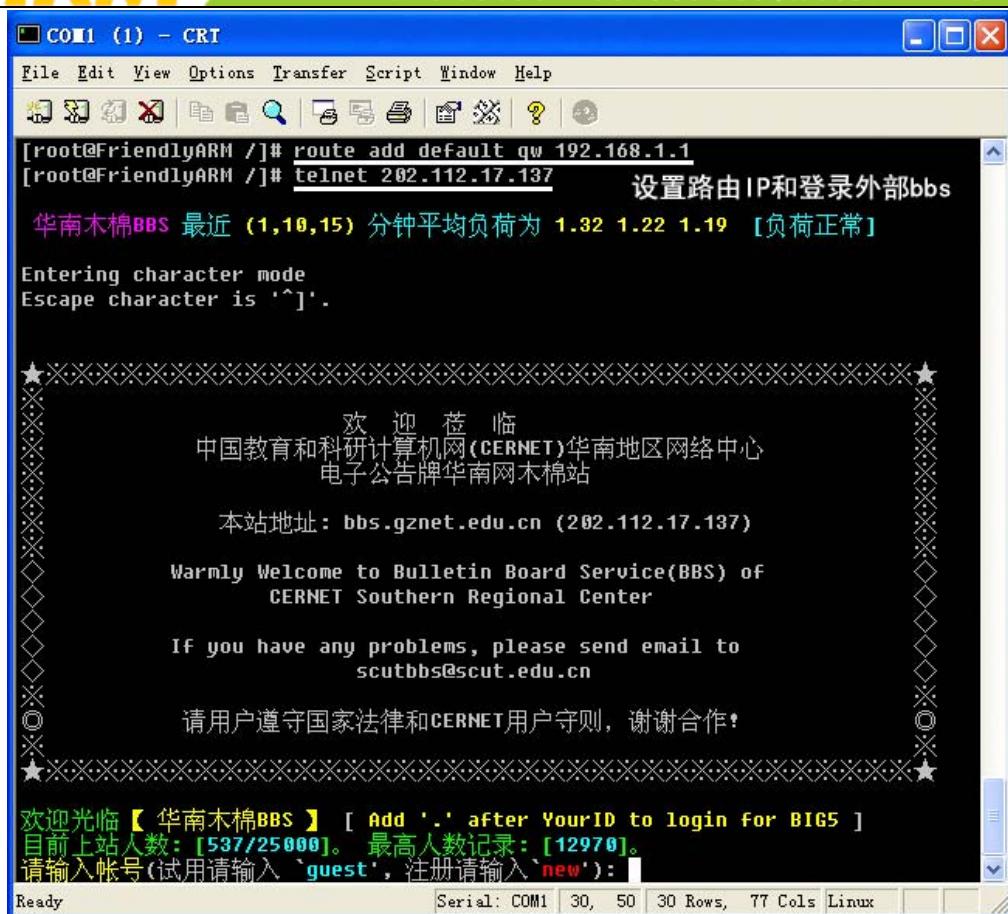


追求卓越 创造精品

TO BE BEST

TO DO GREAT

广州友善之臂计算机科技有限公司



## 2.5.15 如何设置网络以访问互联网网址

首先要确保你的网络环境可以正常登陆互联网，请记下你的网络环境所使用的网关 IP 地址，比如在我这里是 192.168.1.1，然后使用 route 进行设置：

```
# route add default gw 192.168.1.1
```

这时你就可以直接访问互联网上的数字 IP 地址了，比如 ping 一下华南木棉的 BBS(其 IP 地址为 202.112.17.137)：

```
#ping 202.112.17.137
```

如图所示表示可以 ping 通外面的网络：



追求卓越 创造精品

TO BE BEST

TO DO GREAT

广州友善之臂计算机科技有限公司

```
[root@FriendlyARM /]# route add default gw 192.168.1.1
[root@FriendlyARM /]# ping 202.112.17.137
PING 202.112.17.137 (202.112.17.137): 56 data bytes
64 bytes from 202.112.17.137: icmp_seq=0 ttl=52 time=1509.6 ms
64 bytes from 202.112.17.137: icmp_seq=1 ttl=52 time=1426.0 ms
64 bytes from 202.112.17.137: icmp_seq=2 ttl=52 time=1446.8 ms
```

已连接 0:00:24 ANSIW | 115200 8-N-1 | SCROLL | CAPS | NUM | 捕 | 打印 | . . .

要能 ping 通外部网络的实名网址，还需要设置好域名解析服务器，先查看一下您当前网络所使用的 DNS 服务器 IP 地址(可以询问您的网络管理员)：



比如，我这里 DNS 服务器的 IP 为“202.96.128.86”，则在开发板中这样设置：

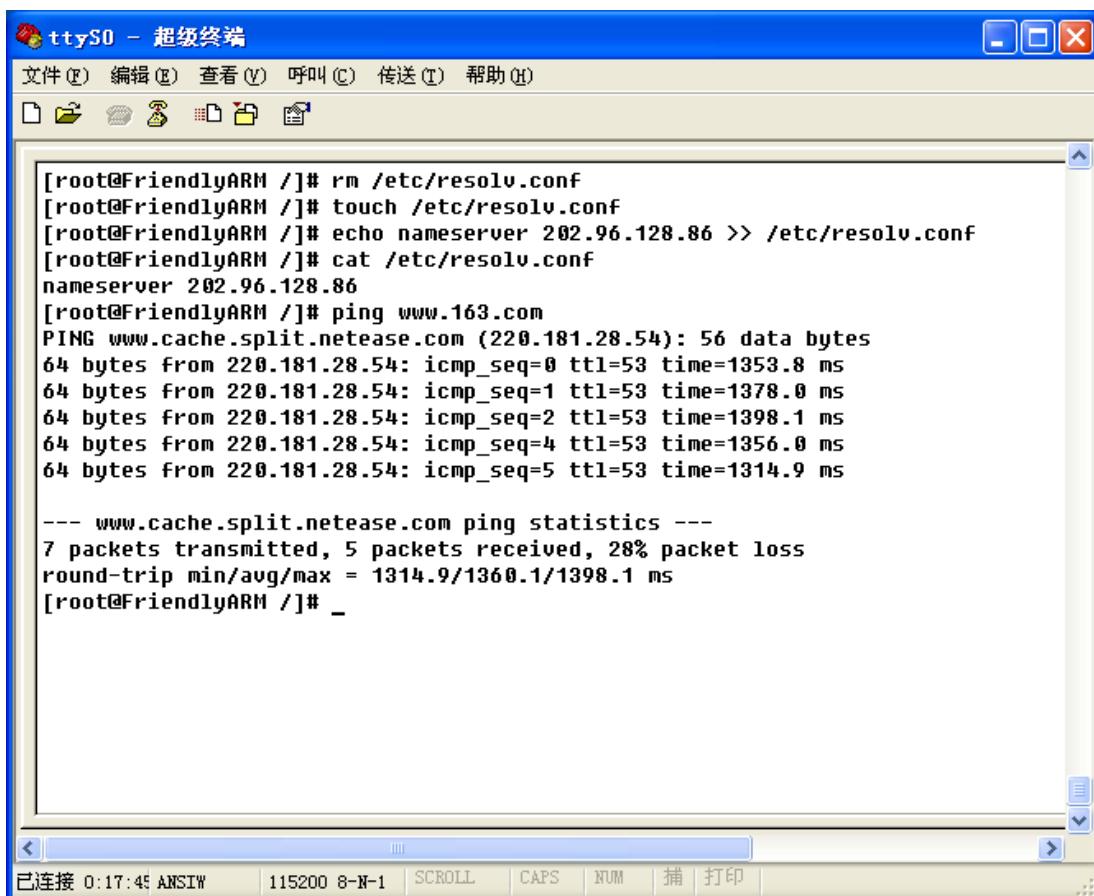
```
#rm /etc/resolv.conf ;首先删除以前的配置文件
```

```
#touch /etc/resolv.conf ; 重新生成一个 resolv.conf 文件
```

```
#echo nameserver 202.96.128.86 >> /etc/resolv.conf ;使用实际的 DNS 服务器 IP 配置  
resolv.conf 文件
```

可以这里主要是修改/etc/resolv.conf 文件，当然你也可以直接使用 vi 进行修改。

全部过程如下图所示：



The screenshot shows a terminal window titled "ttyS0 - 超级终端". The window contains the following command history:

```
[root@FriendlyARM /]# rm /etc/resolv.conf
[root@FriendlyARM /]# touch /etc/resolv.conf
[root@FriendlyARM /]# echo nameserver 202.96.128.86 >> /etc/resolv.conf
[root@FriendlyARM /]# cat /etc/resolv.conf
nameserver 202.96.128.86
[root@FriendlyARM /]# ping www.163.com
PING www.cache.split.163.com (220.181.28.54): 56 data bytes
64 bytes from 220.181.28.54: icmp_seq=0 ttl=53 time=1353.8 ms
64 bytes from 220.181.28.54: icmp_seq=1 ttl=53 time=1378.0 ms
64 bytes from 220.181.28.54: icmp_seq=2 ttl=53 time=1398.1 ms
64 bytes from 220.181.28.54: icmp_seq=4 ttl=53 time=1356.0 ms
64 bytes from 220.181.28.54: icmp_seq=5 ttl=53 time=1314.9 ms

--- www.cache.split.163.com ping statistics ---
7 packets transmitted, 5 packets received, 28% packet loss
round-trip min/avg/max = 1314.9/1360.1/1398.1 ms
[root@FriendlyARM /]# _
```

The terminal window has a blue header bar with icons for minimize, maximize, and close. The bottom status bar shows "已连接 0:17:45 ANSIW" and "115200 8-N-1 | SCROLL | CAPS | NUM | 捕 | 打印 |".

## 2.5.16 如何设置 MAC 地址

开发板中所使用的 MAC 地址是“软”性的，因此你可以通过 ifconfig 命令对它进行重置，以适应于在同一个网络环境中使用多片开发板的情况，具体操作如下：

首先使用 ifconfig 查看一下当前的 mac 地址，运行：

#ifconfig ;注意后面不要跟任何内容



追求卓越 创造精品

TO BE BEST

TO DO GREAT

广州友善之臂计算机科技有限公司

```
Title: 上海滩
Artist: 叶丽仪
Year: 2000
Genre: Goa

[root@FriendlyARM /]# ifconfig
eth0      Link encap:Ethernet HWaddr 08:00:3E:21:C7:F7
          inet addr:192.168.1.230 Bcast:192.168.1.255 Mask:255.255.255.0
          UP BROADCAST RUNNING MULTICAST MTU:1500 Metric:1
          RX packets:0 errors:0 dropped:0 overruns:0 frame:0
          TX packets:0 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:0 (0.0 B) TX bytes:0 (0.0 B)
          Interrupt:53 Base address:0x300

lo       Link encap:Local Loopback
          inet addr:127.0.0.1 Mask:255.0.0.0
          UP LOOPBACK RUNNING MTU:16436 Metric:1
          RX packets:0 errors:0 dropped:0 overruns:0 frame:0
          TX packets:0 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:0
          RX bytes:0 (0.0 B) TX bytes:0 (0.0 B)

[root@FriendlyARM /]#
```

可以看到当前的 mac 地址为“08: 00: 3E: 21: C7: F7”，这是在网卡驱动中默认的 mac 地址，它已经被写死到内核中，除非更改网卡 CS8900 的驱动源代码并重新编译得到新内核。要在运行的系统中动态更改 mac 地址，先关闭当前网络，并使用 ifconfig 重置 mac 地址：

```
#ifconfig eth0 down
```

```
#ifconfig eth0 hw ether 00:11:AA:BB:CC:DD ;提示：a,b,c,d,e,f可以为小写
```

再开启网络，并使用 ifconfig 查看设置以后的 mac 地址，使用 ping 检验网络是否依然可通：

```
#ifconfig eth0 up
```

```
#ifconfig
```

```
#ping 192.168.1.1
```



追求卓越 创造精品

TO BE BEST

TO DO GREAT

广州友善之臂计算机科技有限公司

```
[root@FriendlyARM /]# ifconfig eth0 hw ether 00:11:aa:bb:cc:dd
[root@FriendlyARM /]# ifconfig eth0 up
[root@FriendlyARM /]# ifconfig
eth0      Link encap:Ethernet HWaddr 00:11:AA:BB:CC:DD
          inet addr:192.168.1.230 Bcast:192.168.1.255 Mask:255.255.255.0
          UP BROADCAST RUNNING MULTICAST MTU:1500 Metric:1
          RX packets:60 errors:0 dropped:0 overruns:0 frame:0
          TX packets:8 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:4791 (4.6 Kib) TX bytes:672 (672.0 B)
          Interrupt:53 Base address:0x300

lo       Link encap:Local Loopback
          inet addr:127.0.0.1 Mask:255.0.0.0
          UP LOOPBACK RUNNING MTU:16436 Metric:1
          RX packets:0 errors:0 dropped:0 overruns:0 frame:0
          TX packets:0 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:0
          RX bytes:0 (0.0 B) TX bytes:0 (0.0 B)

[root@FriendlyARM /]# ping 192.168.1.1
PING 192.168.1.1 (192.168.1.1): 56 data bytes
64 bytes from 192.168.1.1: icmp_seq=0 ttl=64 time=3.2 ms
```

已连接 6:41:0 ANSIW | 115200 8-N-1 | SCROLL | CAPS | NUM | 捕 | 打印 |

为了使每次开机后都可以使用新的 mac 地址，可以把以上语句写入启动脚本 /etc/init.d/rcS 文件：

如图：



追求卓越 创造精品

TO BE BEST

TO DO GREAT

广州友善之臂计算机科技有限公司

```
echo "" > /dev/uc/0
echo "Starting networking..." > /dev/uc/0
usleep 300000
/etc/rc.d/init.d/httpd start
echo "" > /dev/uc/0
echo "Starting web server..." > /dev/uc/0
usleep 300000
/etc/rc.d/init.d/leds start
echo "" > /dev/uc/0
echo "Starting leds service..." > /dev/uc/0
echo ""
usleep 300000
/sbin/ifconfig lo 127.0.0.1
/sbin/ifconfig eth0 192.168.1.230 up

/sbin/ifconfig eth0 down
/sbin/ifconfig eth0 hw ether 00:11:aa:bb:cc:dd
/sbin/ifconfig eth0 up

/sbin/madplay /shanghaitan.mp3 &
/bin/hostname -F /etc/sysconfig/HOSTNAME
- /etc/init.d/rcS 66/66 100%
```

### 2.5.17 如何使用 Telnet 远程登录开发板

开发板开机正常运行后，其实已经启动了一个 Telnet 服务，因此用户也可以通过网络远程登录开发板。

在 Windows 的命令行窗口输入“**telnet 192.168.1.230**”，如图出现登录界面，输入“root”(不需要密码)进入系统。

```
FriendlyARM login: root
[root@FriendlyARM ~]#
```

### 2.5.18 使用 ftp 传递文件

无论在 linux 系统还是 windows 系统中，一般安装后都自带一个命令行的 ftp 命令程



追求卓越 创造精品

TO BE BEST

TO DO GREAT

广州友善之臂计算机科技有限公司

序，使用 ftp 可以登录远程的主机，并传递文件，这需要主机提供 ftp 服务和相应的权限；MINI2440 开发板不仅带有 ftp 命令，还在开机时启动了 ftp 服务。为了方便测试，我们可以从 PC 机的命令行窗口登录开发板，并向开发板传递文件。

注意：请确保您执行 ftp 所在的目录有需要上传的文件，这里是 hope.mp3

传送完毕，您可以在串口终端看到目标板的/home/plg 目录下多了一个 hope.mp3 文件。

```
C:\WINDOWS\system32\cmd.exe
Microsoft Windows XP [版本 5.1.2600]
(C) 版权所有 1985-2001 Microsoft Corp.

C:\Documents and Settings\yang>d:

D:>cd ftp

D:>ftp>ftp 192.168.1.230      登录目标板
Connected to 192.168.1.230.
220 FriendlyARM FTP server <Version 6.4/OpenBSD/Linux-ftp-0.17> ready.
User <192.168.1.230:<none>>: plg    用户名: plg
331 Password required for plg.          密码: plg
Password:                                使用bin命令改变文件传输的模式
230 User plg logged in.                 使用put命令上传文件至目标板
ftp> bin
200 Type set to I.                      使用put命令上传文件至目标板
ftp> put hope.mp3                      传送完毕使用by命令退出
200 PORT command successful.
150 Opening BINARY mode data connection for 'hope.mp3'.
226 Transfer complete.
ftp: 发送 4266940 字节, 用时 8.81Seconds 484.22Kbytes/sec.
ftp> by
221 Goodbye.

D:>ftp>
```

## 2.5.19 通过网页控制板上的 LED

在 web server 测试页面中点“网络控制 LED 测试”项，会出现 LED 测试控制页面，如图



追求卓越 创造精品

TO BE BEST

TO DO GREAT

广州友善之臂计算机科技有限公司



您可以使用网页中的各个测试项目进行测试，其中的“LED 测试”将会通过 CGI 程序来控制板上的 LED 灯，其中包括 2 种方式的显示类型和三种不同的显示速度。

如果要停止 web 服务器，则在命令提示符下输入以下命令：

#/etc/rc.d/init.d/httpd stop

要重新启动则输入：

#/etc/rc.d/init.d/httpd start

## 2.5.20 如何挂接使用网络文件系统 NFS

在进行该测试之前，请先按照 4.3 一节搭建好 NFS 服务器系统，然后在命令行输入以下命令（假定服务器的 IP 地址为 192.168.1.111）：

#mount -t nfs -o noblock 192.168.1.111:/opt/FriendlyARM/MINI2440/root\_nfs /mnt

挂接成功，您就可以进入/mnt 目录进行操作了，如下图所示。

取消挂接的命令如下：

#umount /mnt

```

rrr - 超级终端
文件(F) 编辑(E) 查看(V) 呼叫(C) 传送(T) 帮助(H)
[25/Jun/2007:
[root@FriendlyARM /]# clearg server pid=274, port 80+<

[root@FriendlyARM /]# mount -t nfs -o nolock 192.168.1.111:/opt/FriendlyARM/QQ2
40/root_nfs /mnt 挂接NFS网络文件系统到/mnt目录
[root@FriendlyARM /]# ls /mnt/
bin      lib      proc      usr
dev      linuxrc  shbin     var
etc      mnt      shanghai.mp3  www
home    opt      tmp

[root@FriendlyARM /]# cd /mnt/
[root@FriendlyARM /mnt]# madplay shanghai.mp3
MPEG Audio Decoder 0.15.0 (beta) - Copyright (C) 2000-2003 Robert Leslie et al.
Title: 上海滩
Artist: 叶丽仪
Year: 2000
Genre: Goa
播放网络文件系统中的mp3文件

已连接 2:43:34 ANSIW | 115200 8-N-1 | SCROLL | CAPS | NUM | 插 | 打印 |

```

## 2.5.21 设置并保存系统实时时钟

Linux 中更改时间的方法一般使用 **date** 命令，为了把 S3C2440 内部带的时钟与 linux 系统时钟同步，一般使用 **hwclock** 命令，下面是它们的使用方法：

- (1) **date -s 042916352007** #设置时间为 2007-04-29 16:34
- (2) **hwclock -w** #把刚刚设置的时间存入 S3C2440 内部的 RTC
- (3).开机时使用 **hwclock -s** 命令可以恢复 linux 系统时钟为 RTC，一般把该语句放入 **/etc/init.d/rcS** 文件自动执行。

注意：我们提供的系统已经把 **hwclock -s** 命令写入 **rcS** 文件。

## 2.5.22 如何掉电保存数据到 Flash

由于本系统采用了可读写文件系统 yaffs(在嵌入式系统中，专门管理 Flash 存储器的一种文件系统)，因此可以很方便的动态保存数据，掉电后不会丢失。开机后在串口终端运行以下命令：

```
#cp / shanghai.mp3 /home/plg
```

此时将在/home/fa 目录下复制一个同样的文件，然后关机，重新开启系统，可以查看到/home/plg 目录下的文件依然存在。



追求卓越 创造精品

TO BE BEST

TO DO GREAT

广州友善之臂计算机科技有限公司

## 2.5.23 如何设置开机自动运行程序

借助启动脚本可以设置各种程序开机后自动运行，也可以设置其他系统设置，这有点类似于 Windows 系统中的 Autobat 自动批处理文件，启动脚本的位于板子的/etc/init.d/rcS，内容如下(实际内容可能与此不完全一致)：

```
#!/bin/sh

PATH=/sbin:/bin:/usr/sbin:/usr/bin:/usr/local/bin:    ##设置默认有效执行路径
runlevel=S      ##用户等级，在此为：单用户
prevlevel=N
umask 022
export PATH runlevel prevlevel

#
#   Trap CTRL-C &c only in this shell so we can interrupt subprocesses.
#
trap ":" INT QUIT TSTP

#SCSI modules

#Input modules
#/sbin/insmod /lib/input.o
#/sbin/insmod /lib/keybdev.o
#/sbin/insmod /lib/mousedev.o
#/sbin/insmod /lib/evdev.o

#Charactor modules

/bin/ln -s /dev/fb/0 /dev/fb0      ##FrameBuffer 的符号联接
/bin/ln -s /dev/vc/0 /dev/tty1
/bin/ln -s /dev/sound/dsp /dev/dsp #声音设备的符号联接
/bin/ln -s /dev/sound/mixer /dev/mixer #声音设备的符号联接
/bin/ln -s /dev/scsi/host1/bus0/target0/lun0/part1 /dev/sda1

#设置常用临时目录
/bin/mount -t proc none /proc
/bin/mount -t tmpfs none /tmp
/bin/mount -t tmpfs none /var

/bin/mkdir -p /var/lib
```



追求卓越 创造精品

TO BE BEST

TO DO GREAT

广州友善之臂计算机科技有限公司

/bin/mkdir -p /var/run

/bin/mkdir -p /var/log

#Usb devices

#/sbin/insmod /lib/hid-core.o

#/sbin/insmod /lib/usbmouse.o

#/sbin/insmod /lib/usbkbd.o

#Netcard

#/sbin/insmod /lib/cs8900a.o

#各种服务程序

/etc/rc.d/init.d/netd start # telnet/ftp 服务

/etc/rc.d/init.d/httpd start # web server 服务

/etc/rc.d/init.d/leds start # led 服务

/sbin/ifconfig lo 127.0.0.1 #本地回环设备 ip 地址

/sbin/ifconfig eth0 192.168.1.230 up #本机 ip 地址,

/sbin/madplay /shanghaitan.mp3 & #开机后自动运行 madplay 播放 mp3, 用户可以仿照  
此处添加自己的开机程序

/bin/hostname -F /etc/sysconfig/HOSTNAME

## 2.5.24 如何使用命令进行屏幕截图

使用 snapshot 命令可以对当前的 LCD 显示进行截图，并保存为 png 格式的图片。

**#snapshot pic.png**

执行该命令将把当前 LCD 显示进行抓图，并保存为 pic.png 文件。



## 2.6 预装 WindowsCE 的功能和外围资源测试

使用 wince 系统可以测试播放 mp3、网络、优盘的使用、SD 卡的使用、各种常见的服务程序等。

请按照“安装和更新系统程序”一章下载烧写您所需要的 wince 映象文件(此示例中烧写的是 NK\_N35.bin)。

启动 WINCE 系统后，出现一个 WindowsXP 风格的界面，如下图所示。



## 2.6.1 按键测试

驱动程序位置：按键的驱动程序位于 BSP(SMDK2440 文件夹)中。：  
SMDK2440\DRIVERS\Userkey

说明：开发板上的 6 个按键可以模拟 USB 键盘的上、下、左、右、回车和焦点转移  
按键“TAB”，系统启动后，不需要启动任何程序打开它，它们的对应关系如下：

K1 - TAB

K2 - UP

K3 - ENTER

K4 - DOWN

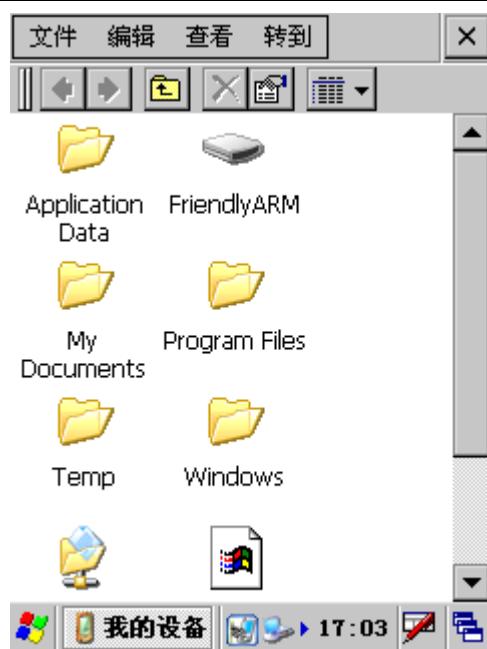
K5 - LEFT

K6 - RIGHT

在桌面状态下，按 K1 键，把焦点转移到桌面的图标，图标的周围会出现虚框，如图中的“回收站”，这时按上下左右键，把焦点转移到其他图标，如“我的设备”，如图：



按下“回车”键，即 K3，就可以打开窗口了：



## 2.6.2 LED 测试

系统启动后，点击运行桌面的“QQ2440 测试”，打开 LED 测试程序，如图：



点击程序中的按钮，可以任意控制开发板上的四个 LED 的亮和灭。

### 2.6.3 屏幕旋转测试

说明：屏幕旋转的驱动程序已经包换在 LCD 驱动中，它不需要特别的硬件支持，是纯软件实现的，因此无需另外的单独驱动。

LCD 驱动的位置：SMDK2440\DRIVERS\DISPLAY

系统启动后，点击运行桌面的“屏幕旋转”，运行界面如左下图，点击其中的“旋转”按钮，屏幕会按逆时针方向 90 度，如右下图。



### 2.6.4 串口通信测试

说明：本开发板提供的 BSP 包含三个串口的标准驱动，要测试串口 2,3，需要使用串口扩展小板。

系统启动后，点击运行桌面的“串口调试助手”，运行界面如左下图：



追求卓越 创造精品

TO BE BEST

TO DO GREAT

广州友善之臂计算机科技有限公司



点“设置”按钮，打开设置窗口，设置串口号为 COM2，波特率为 115200，其他设置如图(右上)，点确定返回主窗口。

同时，连接好扩展板的 COM2 到 PC 一端，并在 PC 端把相应的串口作相同的设置。

在主窗口中点“打开端口”按钮，此时该按钮会变为“关闭端口”，在“发送区”输入一些字符，点“发送”按钮，如左下图，这时会在 PC 端的串口终端接收到从开发板发送来的字符，如右下图：



然后，在串口调试助手的主窗口点“接收”按钮（该按钮会改变为“不接收”），在 PC 端的串口终端输入一些字符(通过超级终端是无法看到的)，在输入的同时，我们看到输入



追求卓越 创造精品

TO BE BEST

TO DO GREAT

广州友善之臂计算机科技有限公司

的字符会在开发板串口调试助手的接收区显示，如图：



我们还可以使用同样的方法测试 COM3，在此就不作详细的说明了。

## 2.6.5 如何使用优盘

在 wince 中使用优盘和使用标准的 windows 使用优盘类似，当 WINCE 系统启动以后，把优盘插入 USB Host 接口，这时板子给优盘供电，优盘的指示灯会闪烁，等待几秒系统就自动加载优盘了，这时可以双击桌面的“我的电脑”图标，打开资源管理器，可以看到优盘的盘符：硬盘，如下图所示。



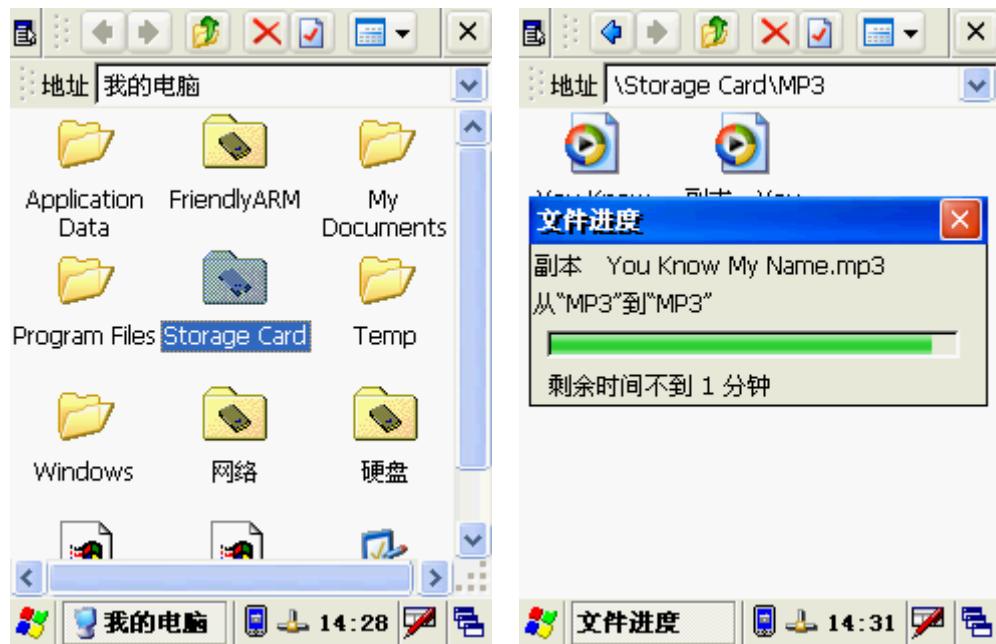
双击硬盘即可进入优盘进行数据读写了。



## 2.6.6 如何使用 SD/MMC 卡

说明：本开发板使用的 SD 卡驱动源自三星公板，只有 dll 驱动文件，没有源代码。

把 SD/MMC 卡插入板上的 SD 插槽，资源管理器中就可以看到 SD 卡的盘符：Storage Card，双击打开进入该目录，就可以对 SD/MMC 卡进行读写了。



## 2.6.7 使用 Windows Media Player 播放 mp3

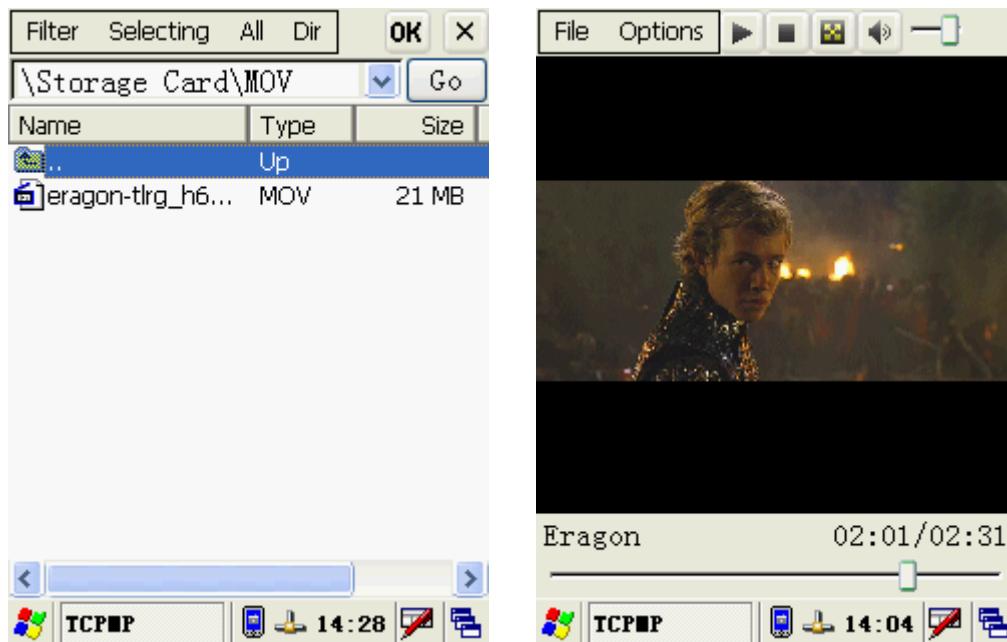
双击打开桌面上的 Media Player 图标，出现 Windows Media Player 播放器，如下图，点 File->Open 根据提示，找到您要播放的 mp3 文件，这样就可以像在 PC Windows 中一样播放 mp3 文件了，另外 Media Player 播放器还可以播放 WMV 格式的影音文件，请自行测试。



## 2.6.8 如何使用超级播放器流畅播放 SD 卡中的 Mpeg4 电影

超级播放器是在 Windows Mobile 中经常用到的一个媒体播放器，它类似于我们经常在电脑上使用的“暴风影音”播放器，现在我们把它集成进去，您可以使用它在 3.5 寸屏上流畅播放各种格式的媒体软件，如 mpeg2, mov, avi 等格式。

双击打开运行桌面上的“超级播放器”，如下图所示，点“File->Open File...”，选择您需要播放的电影文件，下图是播放中的视频截图。

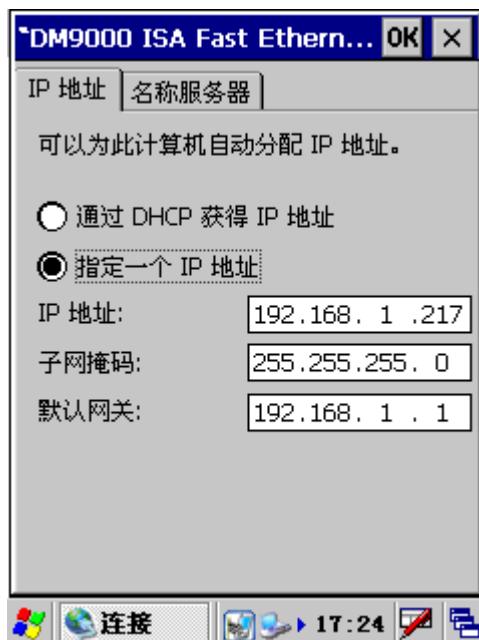


## 2.6.9 以太网测试

点“开始->设置->网络和拨号连接”，打开界面如下。



双击 DM9CE1，打开设置界面，下图是缺省配置，您可以安装实际的网络环境设置。



在 PC 端使用 ping 命令测试网络的连通情况。

### 2.6.10 通过 telnet 登录目标板

我们配置的 WINCE 系统启动后，将会启动 telnet 服务，这时接上网线，可以使用 telnet 命令登录开发板，如下图所示。



**提示:** Telnet 属于 WINCE 内核定制中的组件之一，我们只是在内核的定制中选择加入了它，对其更加详细的使用，如：如何创建其他帐户、设置密码等并不熟悉，请用户自行学习研究。

**注意:** 在 WINCE 系统中，默认的 IP 地址是 **192.168.1.217**，登录时不需要任何密码。

The screenshot shows a Telnet window titled "Telnet 192.168.1.216". The title bar also includes "Welcome to the Windows CE Telnet Service on QQ2440". The command entered is "dir". The output shows a directory listing:

时间	大小	类型	文件名
98/01/01	20:00	<DIR>	网络
98/01/01	20:00	<DIR>	FriendlyARM
07/05/27	15:15		13 PKTSMAP.DAT
07/05/27	15:15	<DIR>	Application Data
07/05/27	15:15		2000000 Printer.swap
07/05/27	23:15		23 控制面板.lnk
07/05/27	23:15	<DIR>	My Documents
07/05/27	23:15	<DIR>	Program Files
07/05/27	23:15	<DIR>	Temp
07/05/27	23:15	<DIR>	Windows

总计文件数 10，总计字节数 2000036。1 个目录，12915260 个可用字节。

## 2.6.11 使用 ftp 向目标版传送文件

**提示:** ftp 服务属于 WINCE 内核定制中的组件之一，我们只是在内核的定制中选择加入了它，对其更加详细的使用，如：如何创建其他帐户、设置密码等并不熟悉，请用户自行学习研究。

我们配置的 WINCE 系统启动后，将会启动 ftp 服务，这时接上网线，可以使用 ftp 命令匿名登录开发板，如下图所示。

**注意:** 在 WINCE 系统中，默认的 IP 地址是 192.168.1.217，用户名和密码均为 **ftp**。



追求卓越 创造精品

TO BE BEST

TO DO GREAT

广州友善之臂计算机科技有限公司

```
C:\WINDOWS\system32\cmd.exe - ftp 192.168.1.216
Microsoft Windows XP [版本 5.1.2600]
(C) 版权所有 1985-2001 Microsoft Corp.

C:\Documents and Settings\yang>ftp 192.168.1.216
Connected to 192.168.1.216.
220 Service ready for new user.
User <192.168.1.216:<none>>: ftp
331 User name okay, need password.
Password:
230 User logged in, proceed.
ftp> ls
200 Command okay.
150 File status okay; about to open data connection.
网络
FriendlyARM
Application Data
Printer.swap
控制面板.lnk
My Documents
Program Files
Temp
Windows
226 Closing data connection.
ftp: 收到 109 字节, 用时 0.39Seconds 0.28Kbytes/sec.
ftp>
```

## 2.6.12 Web server 测试

我们配置的 WINCE 系统启动后，将会启动 http 服务，即通常所说的 web server，这时接上网线，在您的 PC 机浏览器上输入开发板的 IP 地址，可以看到目标板服务器提供的一个简单网页，这说明该服务已经启动了。

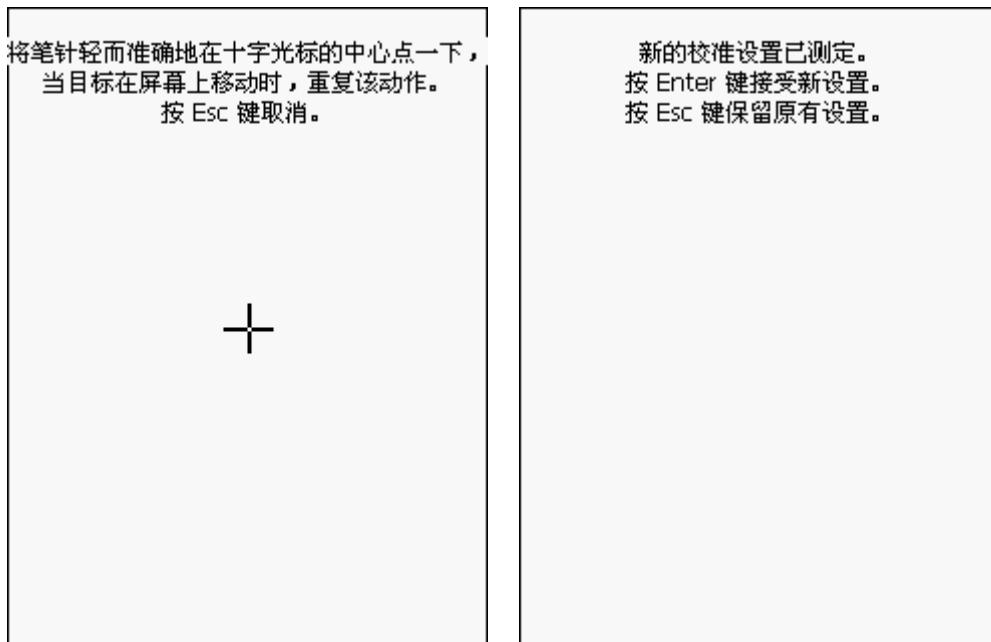


## 2.6.13 触摸屏校正保存

您使用的 WINCE 系统有可能触摸屏不太准确，这时请先接上 USB 鼠标，点开始->设置->控制面板，找到“笔针”图标，双击打开“笔针属性”窗口，点“校准”选项卡的“再校准”按钮，开始重新校正。



根据系统提示，使用五点校正法用触摸笔开始校正，校正完毕，将会跳出如下窗口，这时随便点一个位置即可返回“笔针属性”窗口，点“OK”保存退出。



如果您想保存本次校准的参数，请点“开始->挂起”，然后再重新开机就可以了。



### 2.6.14 使用 ActiveSync 进行 USB 同步通讯

**注意：**安装同步连接的 USB 驱动时，请使用光盘中的“\windows 平台工具\CE 用同步 USB 驱动”

我们假定您已经按照 9.3 节安装好驱动和 ActiveSync 同步软件。

在默认情况下，当 WINCE 系统启动以后，USB Device 连接和设置就已经做好了，这时可以直接用 USB 连接开发板和 PC 机，即可看到如下成功连接的情况：



## 2.6.15 无线网卡测试

我们预装的 WINCE 系统已经集成了无线网卡驱动(型号为: VNUWLC41), 您可以直接把无线网卡模块插入板子上的 USB Host 接口, 将会跳出如下窗口, 同时列表中会出现您附近能够提供无线网络的服务器组名。



双击打开您要连接的无线网络服务提供组进行设置, 这里选择的是“FriendlyARM”, 输入该组的网络密钥(如果没有进行安全设置, 就不用输入了), 这里的密码是明文方式显示的。点“OK”设置完毕, 返回上一个窗口。



可以看到系统正在尝试连接，过一会就可以看到已经与 FriendlyARM 无线网组连接上了。



这时选择点击“IP 信息”选项卡，可以查看该无线网络已经被分配的信息，点“更新”按钮以更新信息，如下图。



## 2.6.16 如何设置实时时钟并保存

点任务栏右下角的时间，出现时间设置窗口，根据提示进行设置就可以了，设置完毕，

点“OK”退出，设置时间不需要点“挂起”保存。



## 2.7 使用 H-JTAG 快速烧写 BIOS 到开发板(全部过程鼠标操作)

### 2.7.1 H-JTAG 简介

**说明：**关于部分 H-JTAG 的说明摘 H-Jtag 说明书，请参考其官方网站  
<http://www.hntag.com>

当前 ARM 的学习与开发非常流行，由于 ARM 的软件开发相对以前单片机而言更加复杂，硬件上的考虑也比较多，因此选择一个好的调试方法将可以使得开发的除错过程变得更加直接和简单。

现在市面上有很多可用于 ARM 调试的仿真器出售，然而其价格往往都比较贵。这些仿真器一般都有其专用的软件和硬件，在速度和 flash 编程等方面有各自的优势。然而对初学者而言，这些仿真器的成本都太高。而简易仿真器的出现，使得大家可以使用甚至自制 ARM 仿真器硬件。

有了调试器的硬件，还要加上调试代理软件，作为中介，将调试器前端软件（比如 AXD）的调试信息与目标板上的目标芯片交互，才能最终完成仿真的任务。目前，可以免费使用的简易 ARM 仿真器的代理软件很多，差别也比较大，主要表现在易用程度，目标器件支持，调试速度等方面。H-JTAG 作为近来新推出的简易 ARM 仿真器调试代理，其支持器件比较多，支持的调试器前端软件也比较多，特别是支持 keil，其调试速度也很有优势。

H-JTAG 是由 twentyone 推出的一款免费调试代理软件。官方主页为：  
<http://www.hntag.com/>



追求卓越 创造精品

TO BE BEST

TO DO GREAT

广州友善之臂计算机科技有限公司

目前的版本为 0.4.4，支持下列特性(更新的版本请到 H-JTAG 网站下载试用):

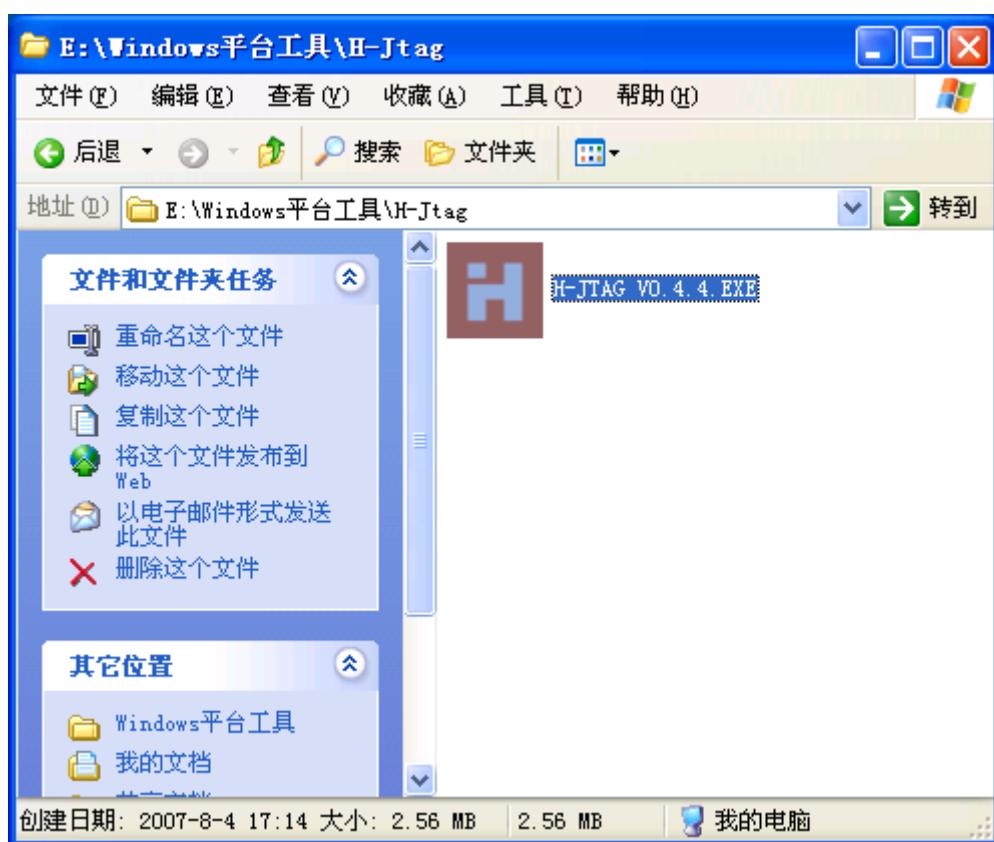
1. 支持 RDI 1.5.0 与 1.5.1;
2. 支持 ARM7 与 ARM9 (包括 ARM9E-S 与 ARM9EJ-S) ;
3. 支持 thumb 与 arm 指令集;
4. 支持 little-endian 与 big-endian;
5. 支持 semihosting;
6. 支持 wiggler, sjf-jtag 以及用户自定义的简易调试器硬件接口;
7. 支持 WINDOWS 9.X/NT/2000/XP;
8. 支持 flash 器件的编程

**注：本开发板所用的 JTAG 板即为 SJF-JTAG**

## 2.7.2 安装并设置 H-JTAG

### (1) 安装 H-JTAG

H-JTAG 安装文件位于光盘的“Windows 平台工具\H-JTAG”目录，双击运行，按照其提示安装即可。

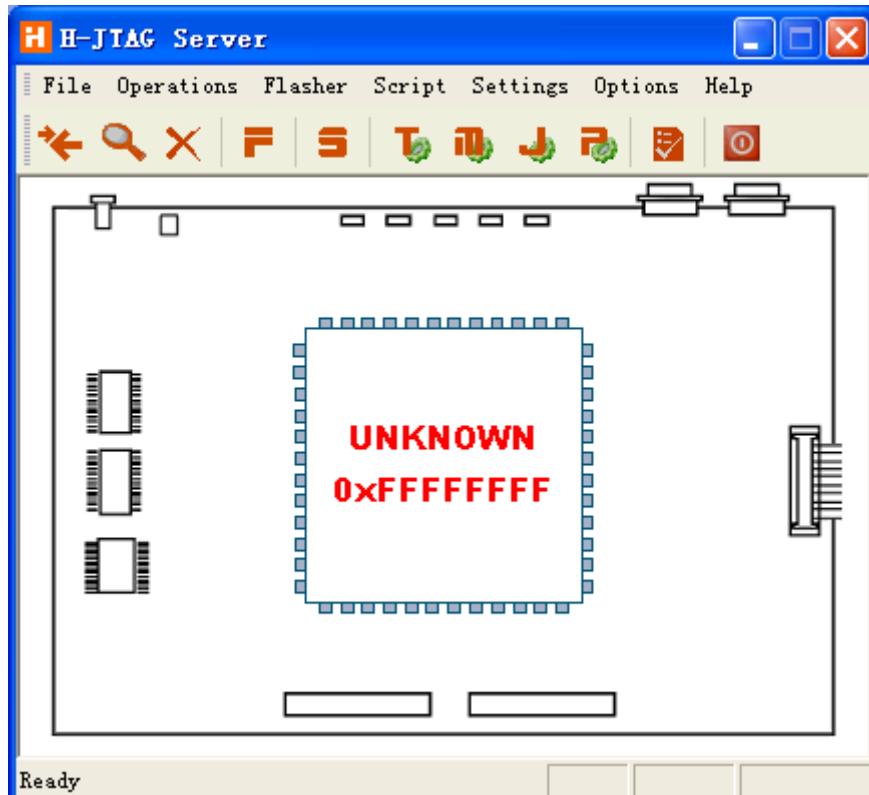


安装完毕，会在桌面生成 H-JTAG 和 H-Flasher 快捷方式，双击运行 H-JTAG，程序将自动检测是否连接了 JTAG 设备，因为之前我们还没有做任何设置，所以会跳出一个提示窗

口：

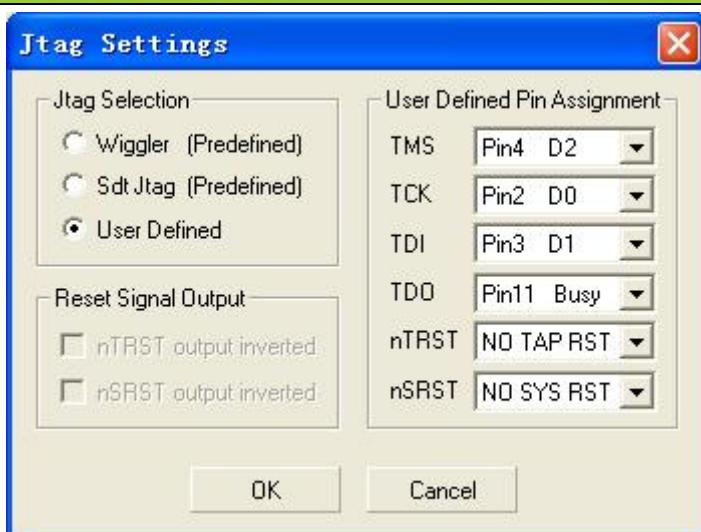


点击确定，进入程序主界面，因为没有连接任何目标器件，因此显示如图所示：



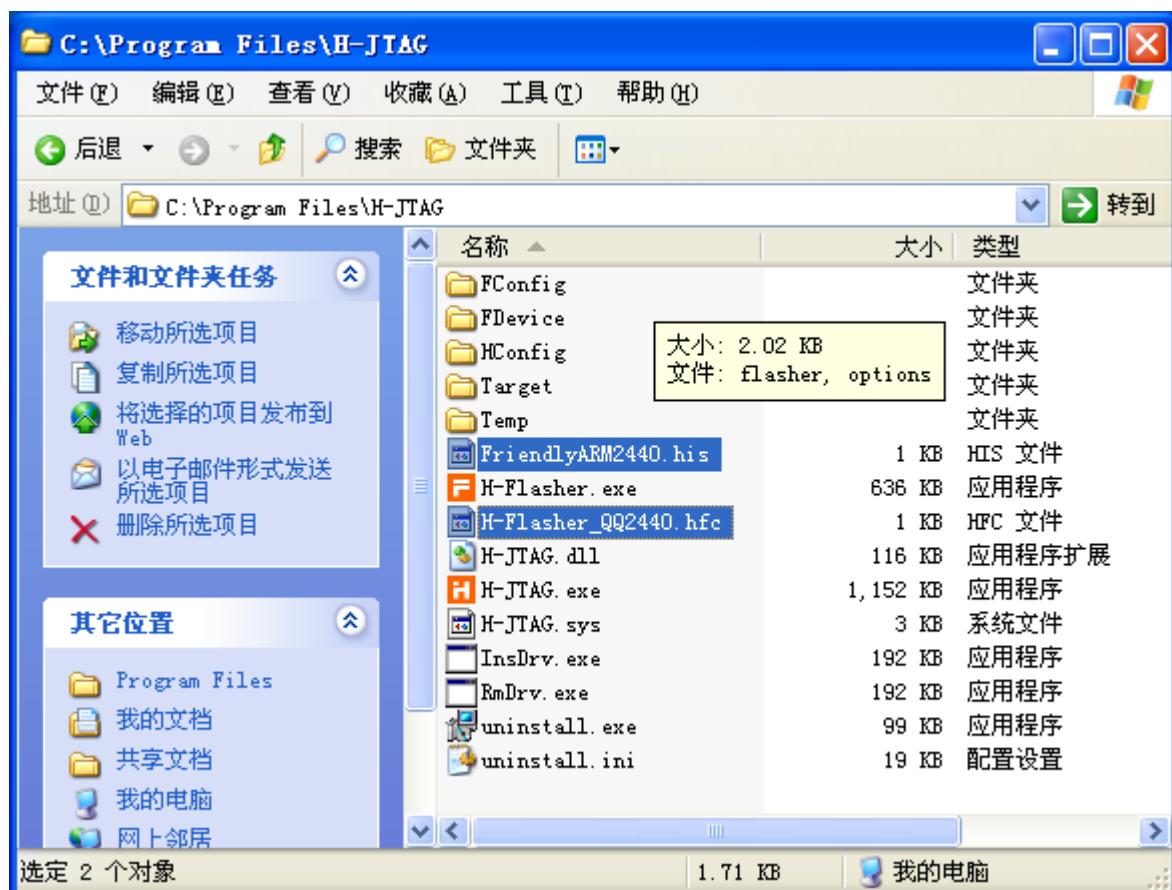
## (2)设置 JTAG 端口

在 H-JTAG 主界面的菜单里点 Setting->Jtag Settings，作如下图所示设置，点 OK 返回主界面。

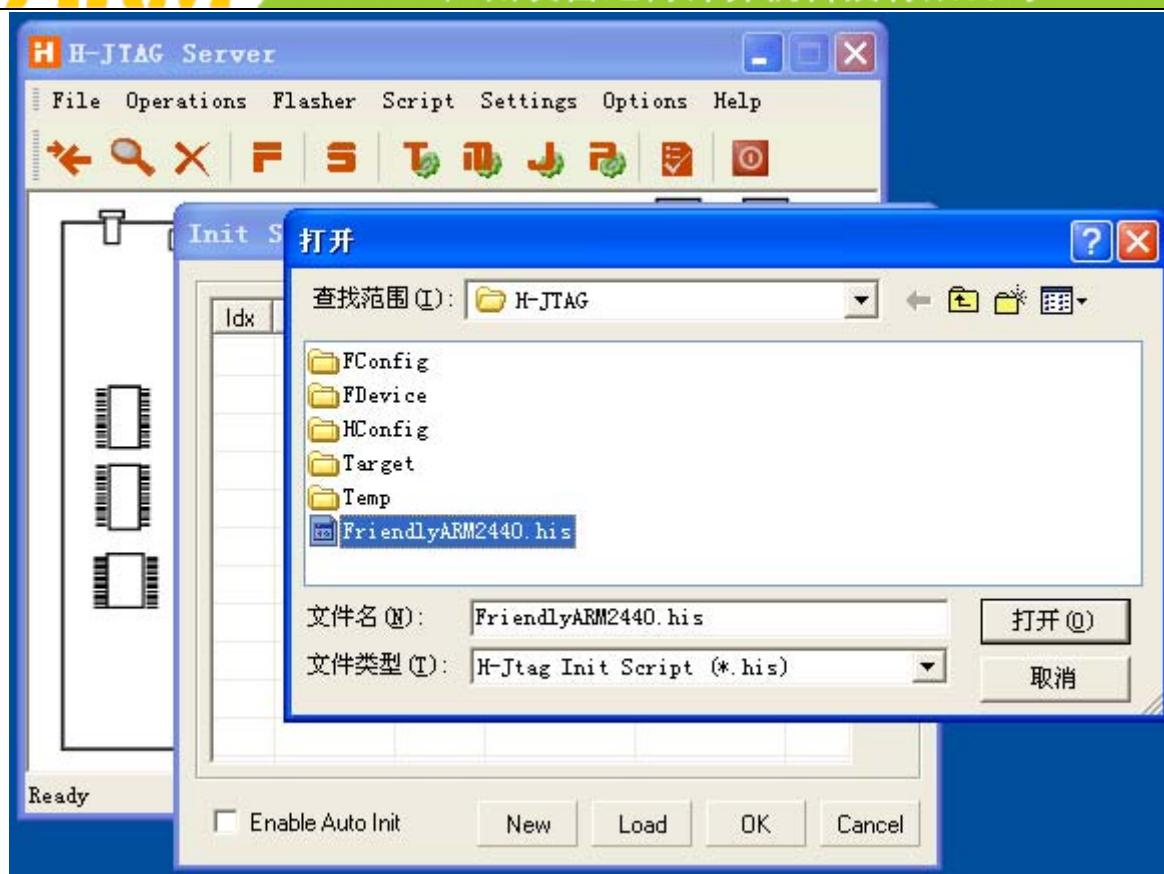


### (3)设置初始化脚本

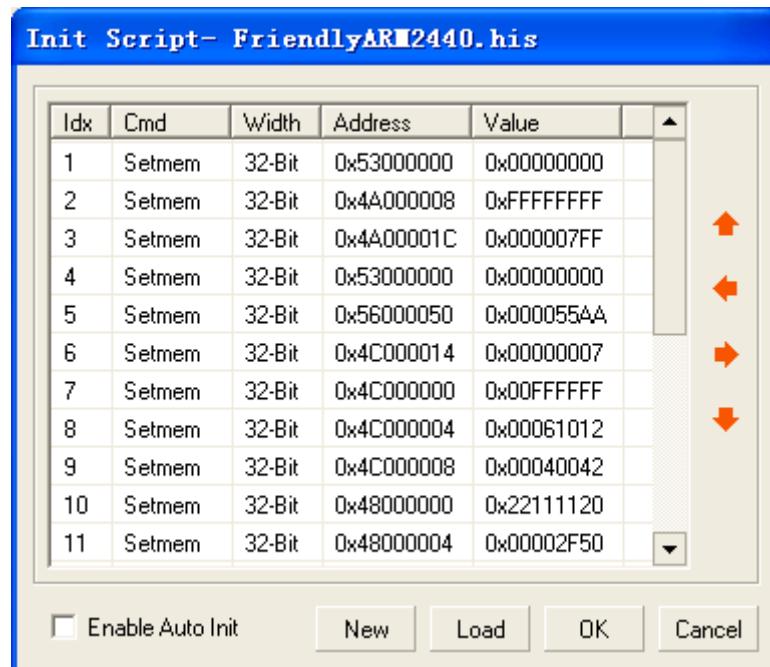
把光盘“Windows 平台工具\H-JTAG”目录中的 FriendlyARM2440.his 和 H-Flasher\_QQ2440.hfc 文件复制到 H-JTAG 的安装目录, 如图:



在 H-JTAG 的主界面, 点 Script->Init Script, 跳出 Init Script 窗口, 点该窗口下面的 Load 按钮, 找到并选择打开刚刚复制的 FriendlyARM2440.his 文件, 如图:



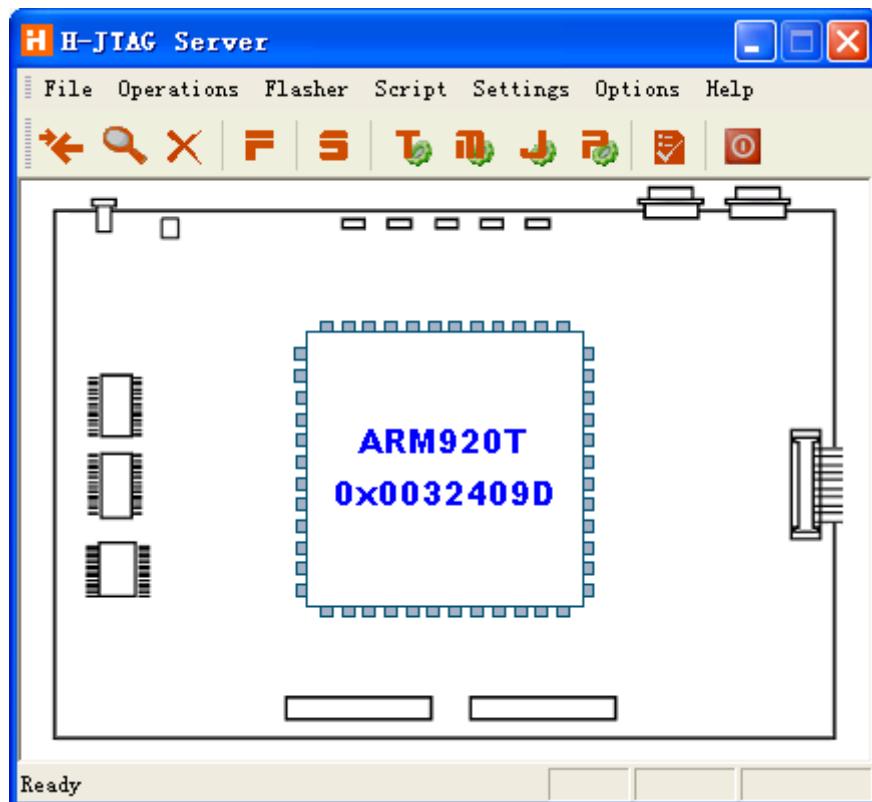
这时，Init Script 窗口会被载入的脚本填充，如图，**注意不要点选“Enable Auto Init”**，点 OK 退回 H-JTAG 主界面：



#### (4)检测目标器件

使用开发板附带的 JTAG 小板连接开发板的 JTAG 接口，并接上打开电源。点主菜单 Operations->Detect Target，或者点工具栏相应的图标也可以，这时就可以看到已经检测到目标器件了。

**提示：如果没有设置初始化脚本，也可以检测到 CPU，但无法进行下面的单步调试。**



### 2.7.3 设置 Flash 型号并烧写 BIOS

**注意：执行以下步骤之前，要确保开发板选择从 Nor Flash 启动，切记！**

接上面的步骤：

(1)点 H-JTAG 主菜单的 Flasher → Start H-Flasher 打开 H-Flasher 烧写程序窗口，如图：

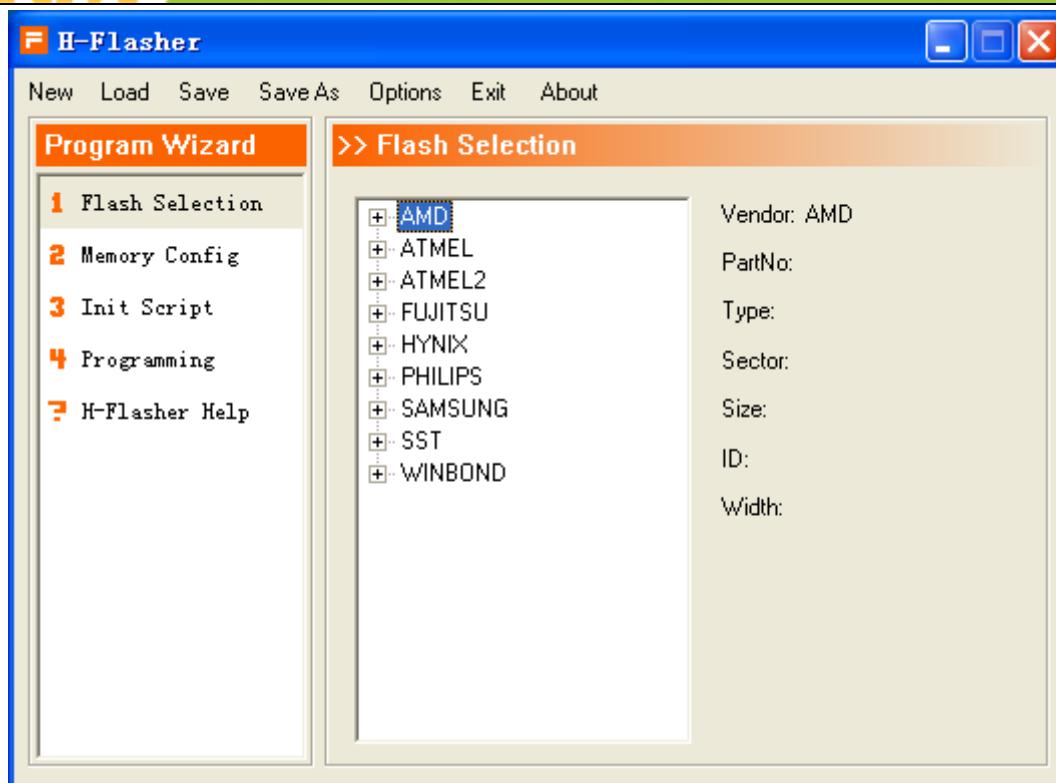


追求卓越 创造精品

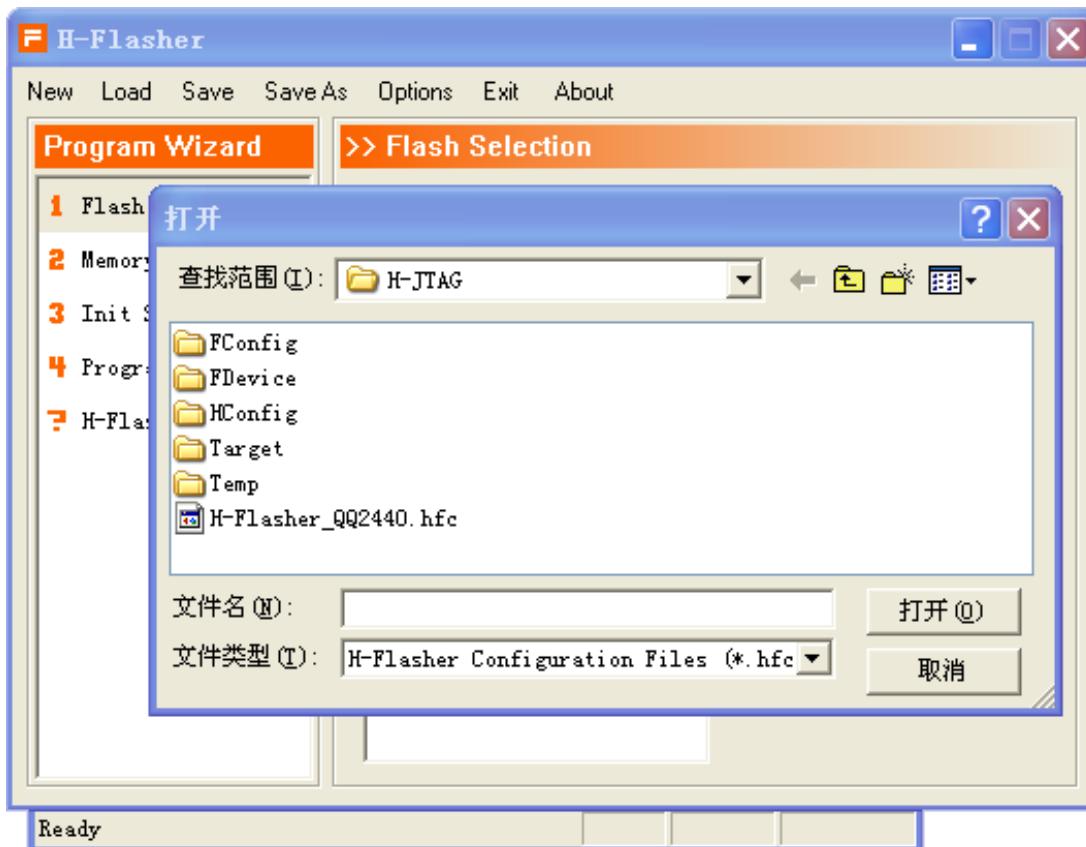
TO BE BEST

TO DO GREAT

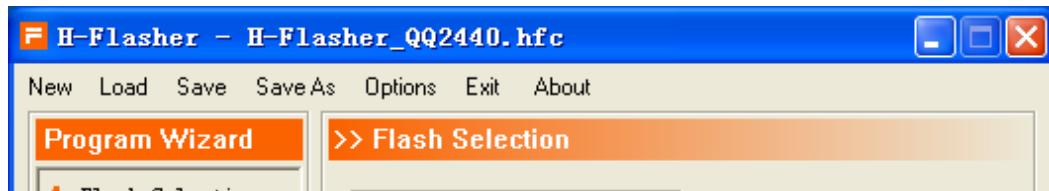
广州友善之臂计算机科技有限公司



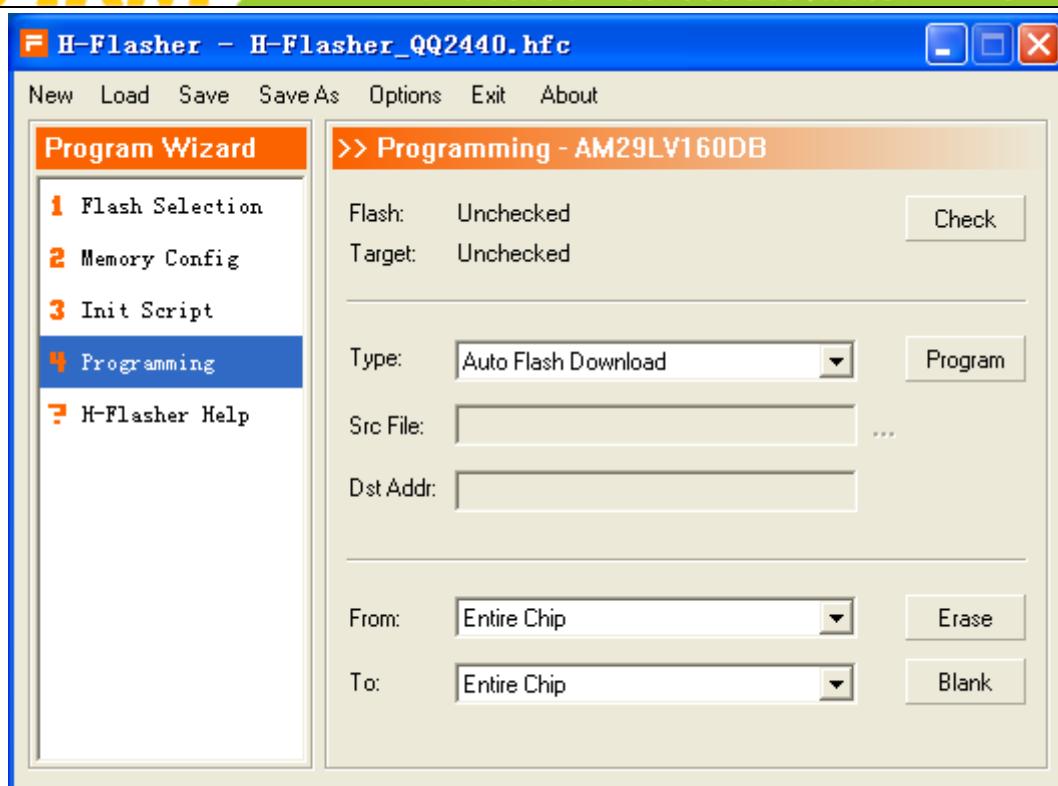
(2)在 H-Flasher 窗口菜单中选择“Load”，出现打开文件选择窗口，选择上面步骤复制的 H-Flasher\_mini2440.hfc 文件，如图：



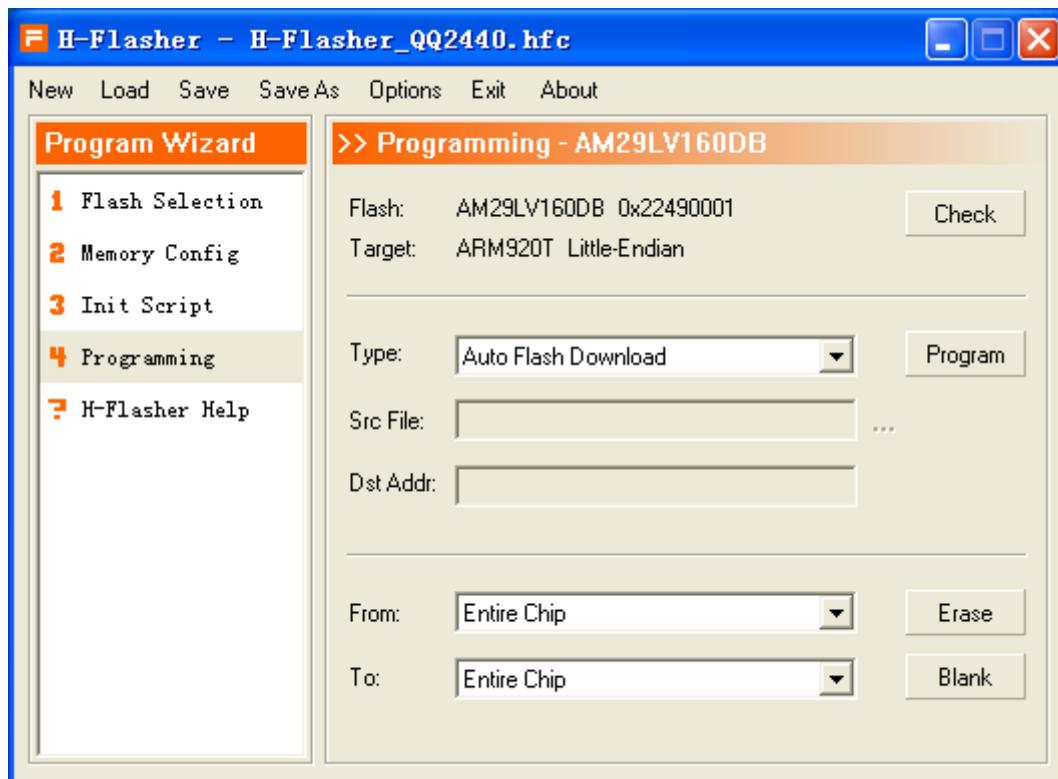
(3) 可以看到 Flash 初始文件已经被载入并显示在 H-Flasher 窗口标题栏中，如图：



这时，点 H-Flasher 左侧导航栏的“4 Programming”，出现如图界面：



(4)点一下“Check”按钮，H-Flasher 将会探测到 mini2440 所用的 Nor Flash 所用的型号为 AM29LV160DB，如图：





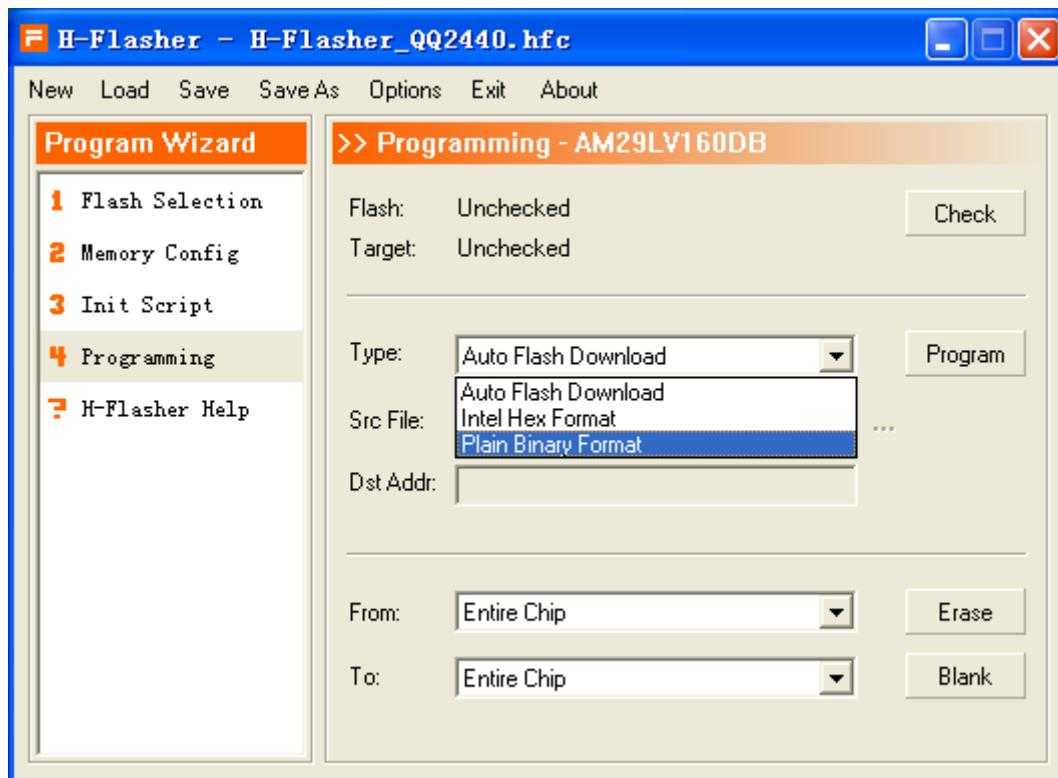
追求卓越 创造精品

TO BE BEST

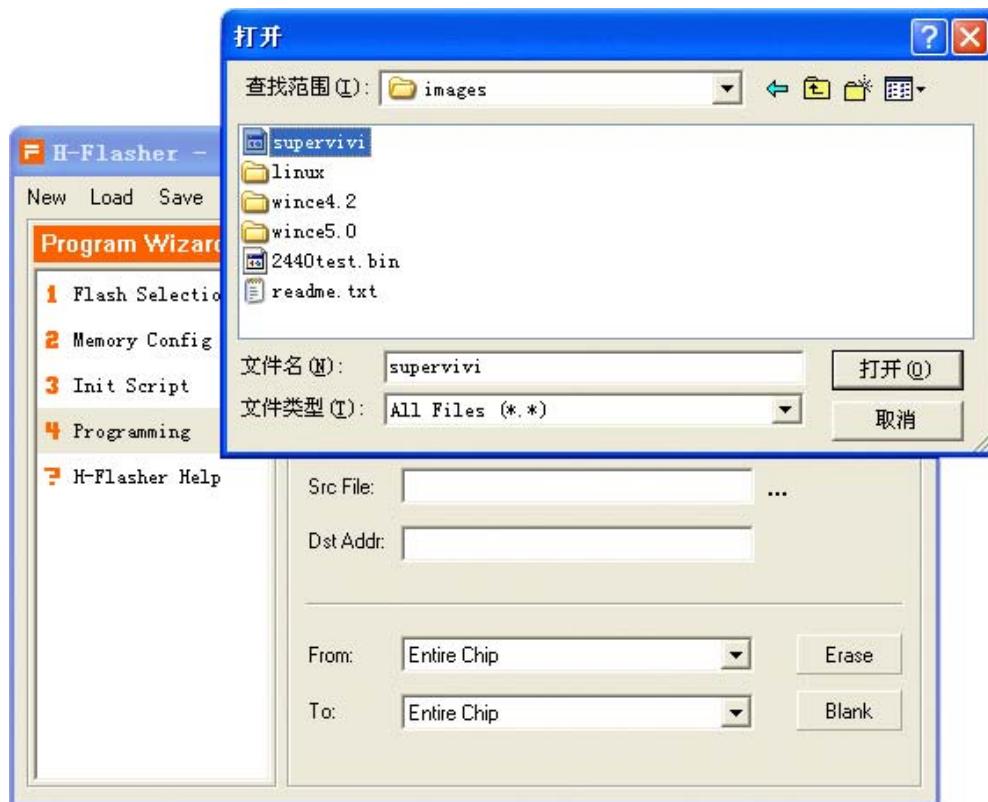
TO DO GREAT

广州友善之臂计算机科技有限公司

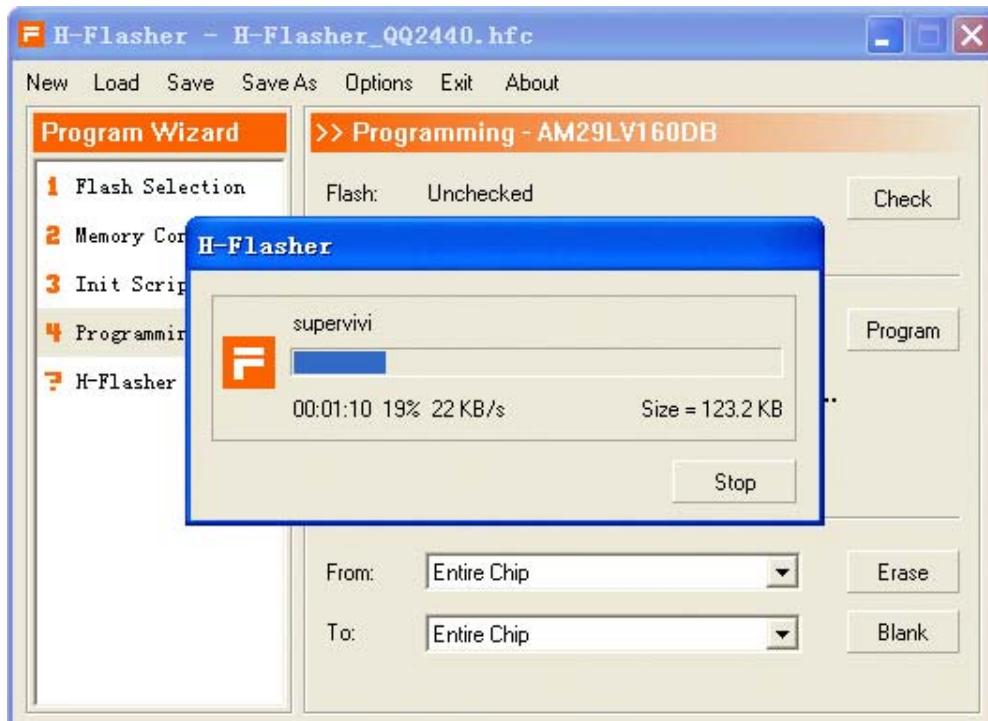
(5)点 Type 下拉列表, 选择 “Plain Binary Format”, 如图:



再点 Src File 右侧的浏览按钮, 选择所要烧写的文件 supervivi, 并在 Dst Addr 一栏中输入烧写的起始地址 “0” ,如图:

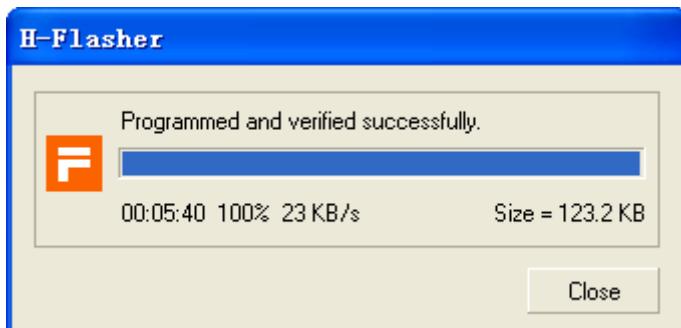


(6)点“Program”按钮开始烧写 supervivi，如图：



烧写结束，“Stop”按钮会变为“Close”，点“Close”结束烧写并取下 JTAG 烧写板，如

图：



(7)至此，您已经把 BIOS 烧写入 Nor Flash 中，如果您需要烧写更多的 mini2440，无需重复以上步骤(下次打开运行 H-JTAG 时会自动载入上次的配置)，可以直接接上 Jtag 线，打开电源，点“Check”先检测一下 Flash，再点“Program”就可以开始新的烧写了。

**注意：目前 H-JTAG 只能用于烧写 Nor Flash，并不能直接烧写 Nand Flash，一些开发板厂商为了节省成本，很多都省掉了 Nor Flash，因此并不能用本节介绍的步骤快速简单的烧写 BIOS。**

## 2.7.4 常见问题

其实我们不推荐初学者使用 H-JTAG 烧写 NOR FLASH，一旦操作错误就会导致无法正常进行后面步骤的操作。

有的用户可能尝试少了其他程序或者文件到 NOR FLASH 中，这时再使用 H-JTAG 烧写 Supervivi 有可能会失败，通常是因为系统复位或者开机后就马上执行 NOR FLASH 中的程序了，这导致 H-JTAG 无法正常执行。

可以尝试这样解决：在复位之后马上点 H-Flaser 的“Progarm”按钮，防止 NOR FLASH 中的程序进一步执行。



追求卓越 创造精品

TO BE BEST

TO DO GREAT

广州友善之臂计算机科技有限公司

## 第三章 备份恢复系统及安装更新

本节内容主要通过图解介绍如何通过 USB 备份和恢复您的开发板系统程序，以及如何安装更新系统。

很多人特别是初学者认为，安装嵌入式系统是十分复杂的事情，需要很高的技术水平，特别是 Linux 系统，总以为要输入很多的命令进行配置安装。通过本节的介绍，你将会发现安装和更新系统简直比在 PC 上安装 Windows 系统还要简单，只要几分钟的时间，输入不超过 5 个字母，您就可以随意更新您的系统了。

如果你认为快速的输入一些命令行指令是一件很酷的事情，可以参考附录 2。

**提示：本节内容的各个小节均具有独立性，用户可以根据自己的目的需要进行独立阅读操作。**

本小节的工作环境为 Windows/2000/XP。

### 3.1 备份和恢复系统

在开发过程中，我们经常需要不断的烧写和更新系统进行调试，有时您的系统程序会配合的很好，让你有大功告成的感觉，这时我们都希望能保存当前的目标系统程序以供以后参考，或者完全复制当前的系统装入到另一个目标板；特别是当项目要求紧迫，而重新构建一个同样的系统又很复杂的时候，这时使用系统自带的备份和恢复功能，就能够快速有效地解决您的问题。

**注意：目前只有广州友善之臂提供了此功能技术。**

#### 3.1.1 备份系统

**注意：本小节假定您已经按照前面的方法安装了 USB 驱动，并把开发板设置为 Nor Flash 启动，设置方法请参考前面的章节。**

**另外，备份并不会破坏任何 Flash 的数据，备份之前请检查您的系统是否能够正常运行使用，以便参考。**

(1) 连接好串口，打开超级终端，上电启动开发板，进入 BIOS 功能菜单：



追求卓越 创造精品

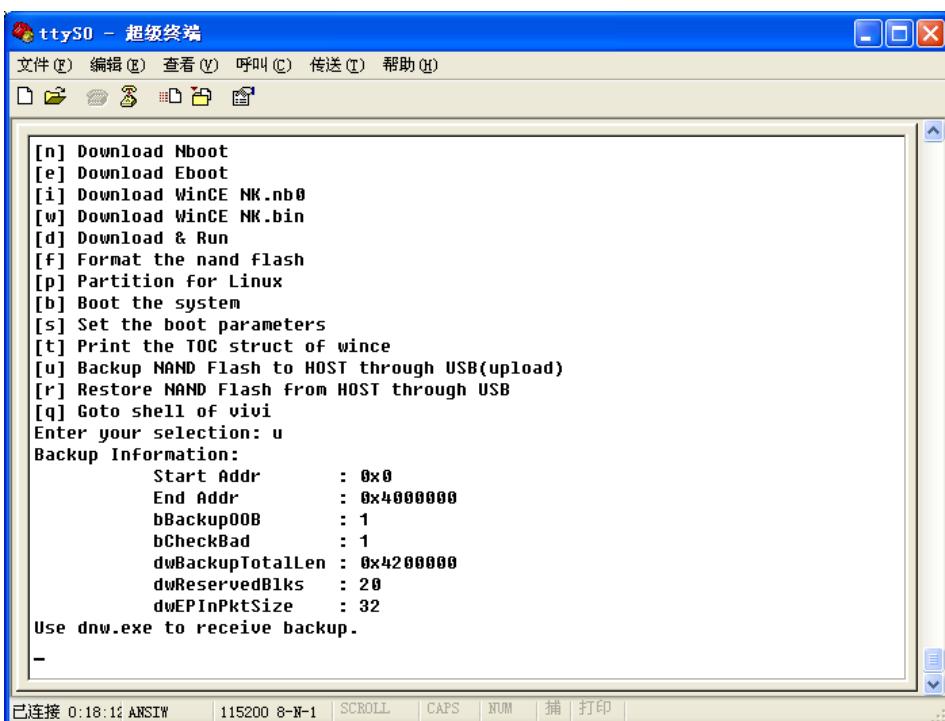
TO BE BEST

TO DO GREAT

广州友善之臂计算机科技有限公司



(2) 选择功能号[u]开始备份 Nand Flash 内容到文件，如图所示：



(3) 打开 DNW 程序，接上 USB 电缆，如果 DNW 标题栏提示[USB: OK]，说明 USB



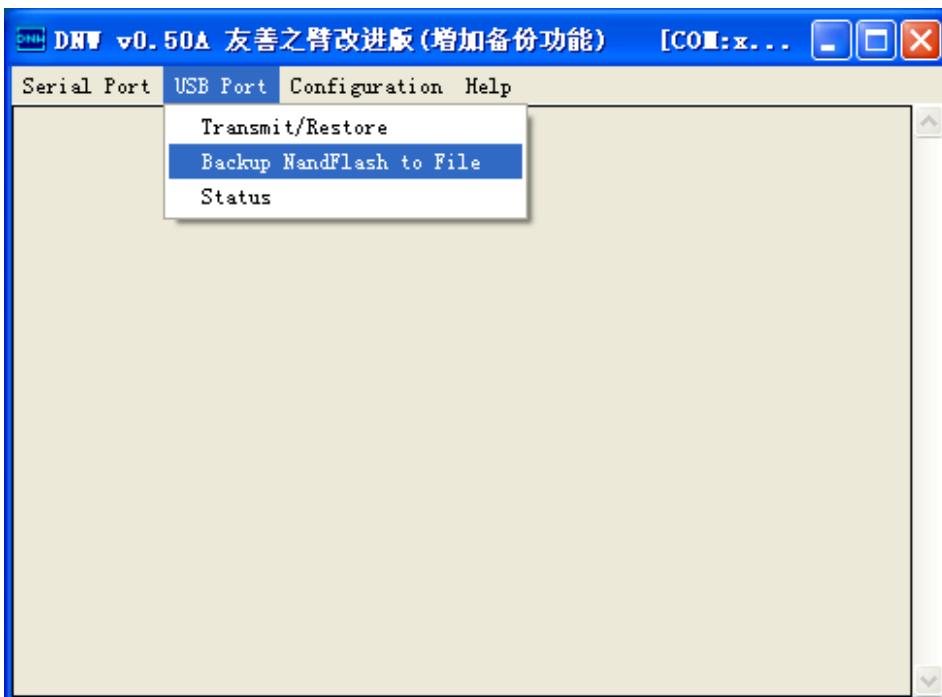
追求卓越 创造精品

TO BE BEST

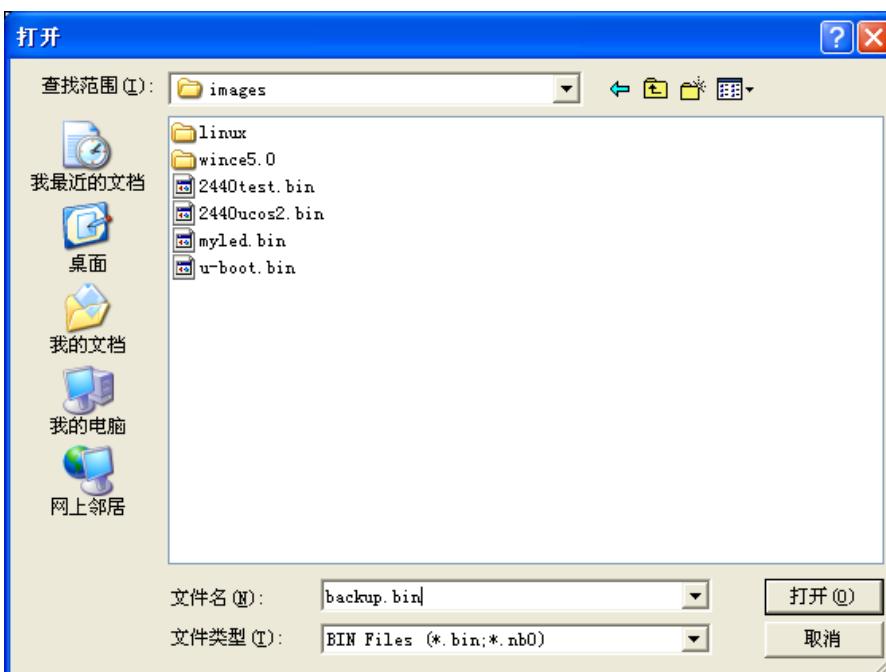
TO DO GREAT

广州友善之臂计算机科技有限公司

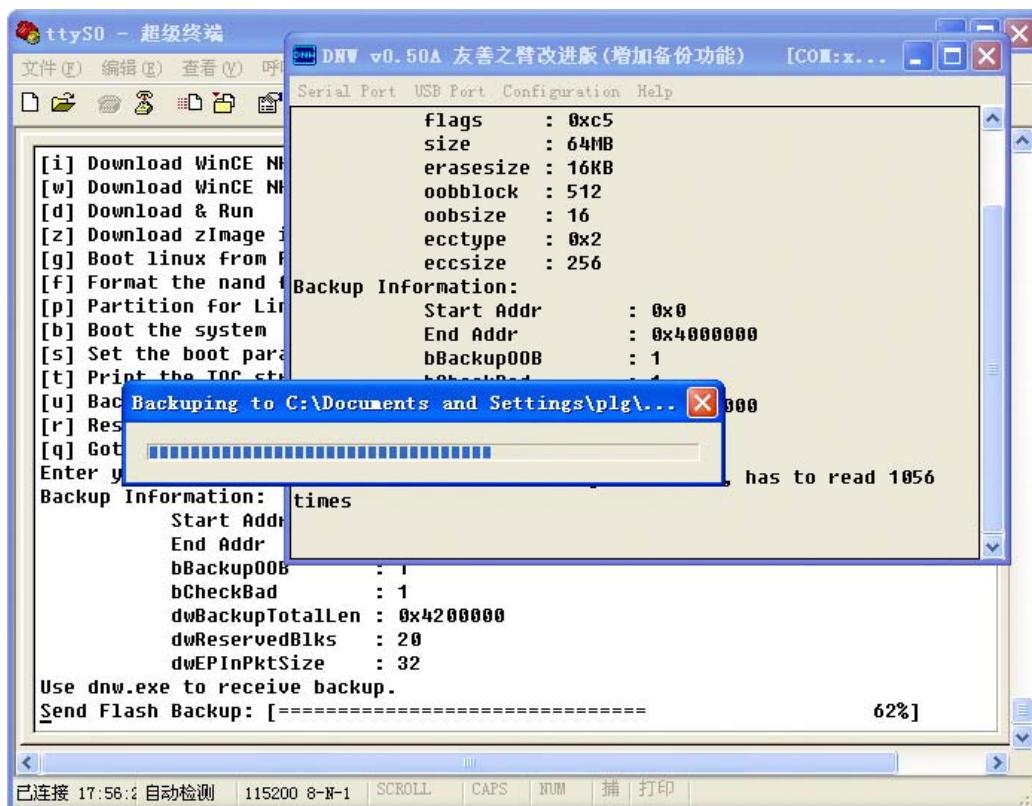
连接成功，这时选择 DNW 菜单的 Usb Port → Backup NandFlash to File，如图所示：



跳出文件保存窗口，选择所要保存文件的位置，并命名(这里保存文件为 backup.bin)，如图：



系统开始备份，备份的进度如图所示：



备份完毕，DNW 窗口会出现如图信息：



追求卓越 创造精品

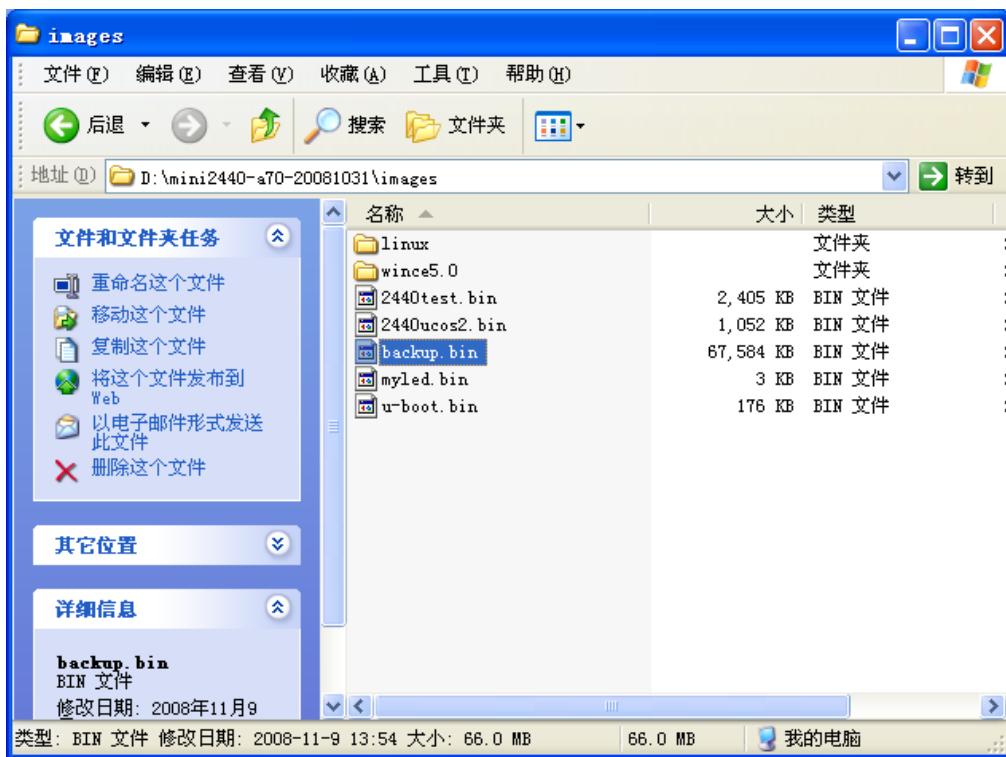
TO BE BEST

TO DO GREAT

广州友善之臂计算机科技有限公司

```
DNW v0.50A 友善之臂改进版(增加备份功能) [COM:x] [...]
Serial Port USB Port Configuration Help
=====
Nand Flash Information:
  type      : 0x4
  flags     : 0xc5
  size      : 64MB
  erasesize : 16KB
  oobblock  : 512
  oobsize   : 16
  ecctype   : 0x2
  eccsize   : 256
Backup Information:
  Start Addr      : 0x0
  End Addr        : 0x4000000
  bBackup00B       : 1
  bCheckBad        : 1
  dwBackupTotalLen : 0x4200000
  dwReservedBlks  : 20
  dwEPInPktSize    : 32
dnw.exe read data 65536 bytes a time, has to read 1056
times
=====
USB Backup End =====
```

最后备份生成的文件大小为 66M byte, 这是因为包含了 Nand Flash 的所有字节信息, 关于 Nand Flash 更多的介绍请查看相应的数据手册。



### 3.1.2 使用备份文件恢复系统

**注意：本小节假定您已经按照前面的方法安装了 USB 驱动，并把开发板设置为 Nor Flash 启动，设置方法请参考前面的章节。**

**另外，要使用备份文件恢复系统，必须先按照上一步创建生成备份文件。恢复系统会擦除整片 Nand Flash！**

下面我们开始介绍如何使用备份文件恢复系统。

(1) 连接好串口，打开超级终端，上电启动开发板，进入 BIOS 功能菜单：



追求卓越 创造精品

TO BE BEST

TO DO GREAT

广州友善之臂计算机科技有限公司

```
##### FriendlyARM BIOS for 2440 #####
[v] bon part 0 320k 2368k
[v] Download vivi
[k] Download linux kernel
[y] Download root_yaffs image
[c] Download root_cramfs image
[a] Absolute User Application
[n] Download Nboot
[e] Download Eboot
[i] Download WinCE NK.nb0
[w] Download WinCE NK.bin
[d] Download & Run
[f] Format the nand flash
[p] Partition for Linux
[b] Boot the system
[s] Set the boot parameters
[t] Print the TOC struct of wince
[u] Backup NAND Flash to HOST through USB(upload)
[r] Restore NAND Flash from HOST through USB
[q] Goto shell of vivi
Enter your selection:
```

(2) 选择功能号[r]开始使用备份文件恢复整个 Nand Flash, 如图所示:

```
[v] Download vivi
[k] Download linux kernel
[y] Download root_yaffs image
[c] Download root_cramfs image
[a] Absolute User Application
[n] Download Nboot
[e] Download Eboot
[i] Download WinCE NK.nb0
[w] Download WinCE NK.bin
[d] Download & Run
[z] Download zImage into RAM
[g] Boot linux from RAM
[f] Format the nand flash
[p] Partition for Linux
[b] Boot the system
[s] Set the boot parameters
[t] Print the TOC struct of wince
[u] Backup NAND Flash to HOST through USB(upload)
[r] Restore NAND Flash from HOST through USB
[q] Goto shell of vivi
Enter your selection: r
USB host is not connected yet.
USB host is connected. Waiting a download.
```

(3) 打开 DNW 程序, 接上 USB 电缆, 如果 DNW 标题栏提示[USB: OK], 说明 USB 连接成功, 这时选择 DNW 菜单的 Usb Port → Transmit/Restore, 如图所示:

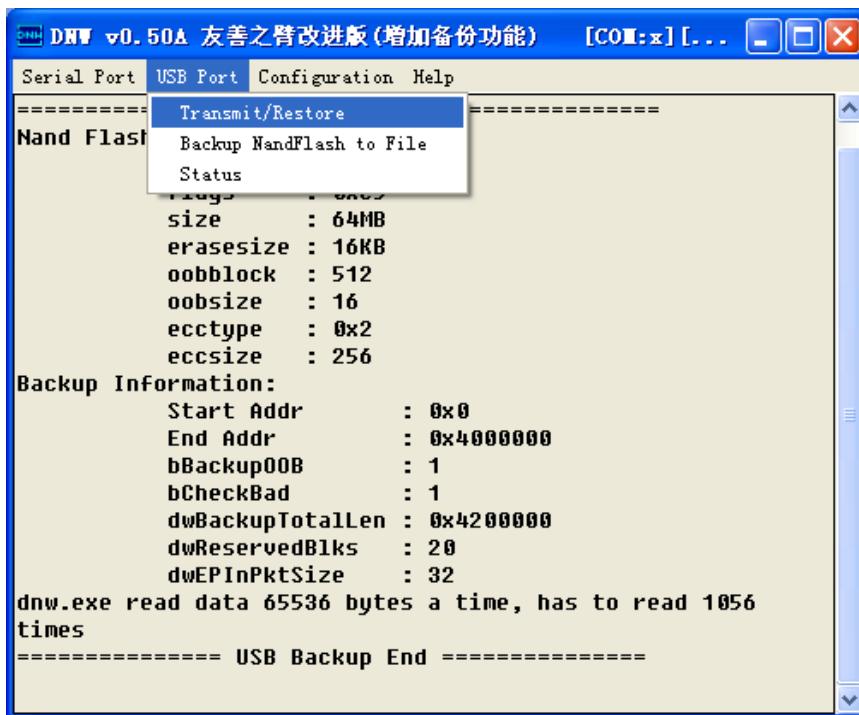


追求卓越 创造精品

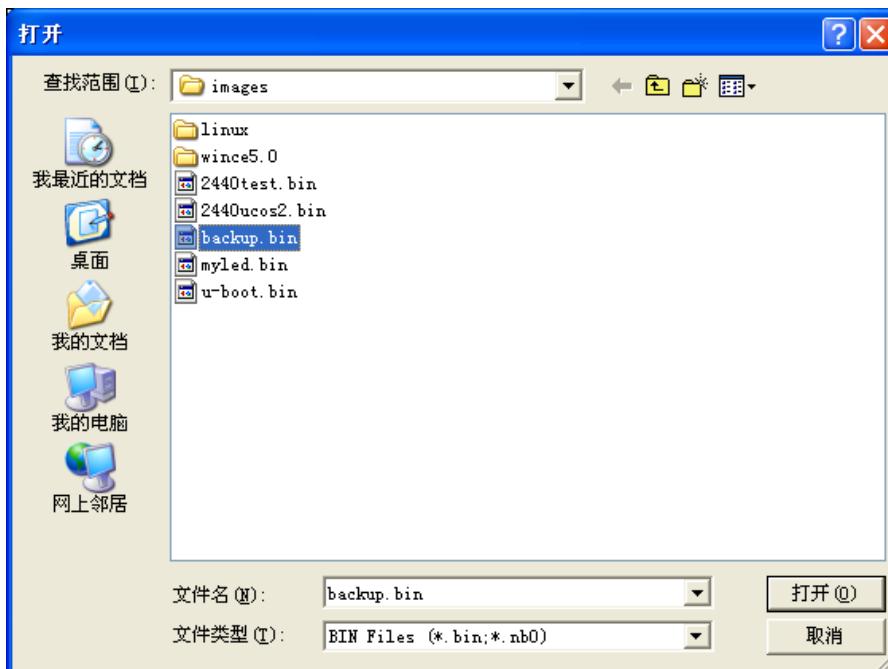
TO BE BEST

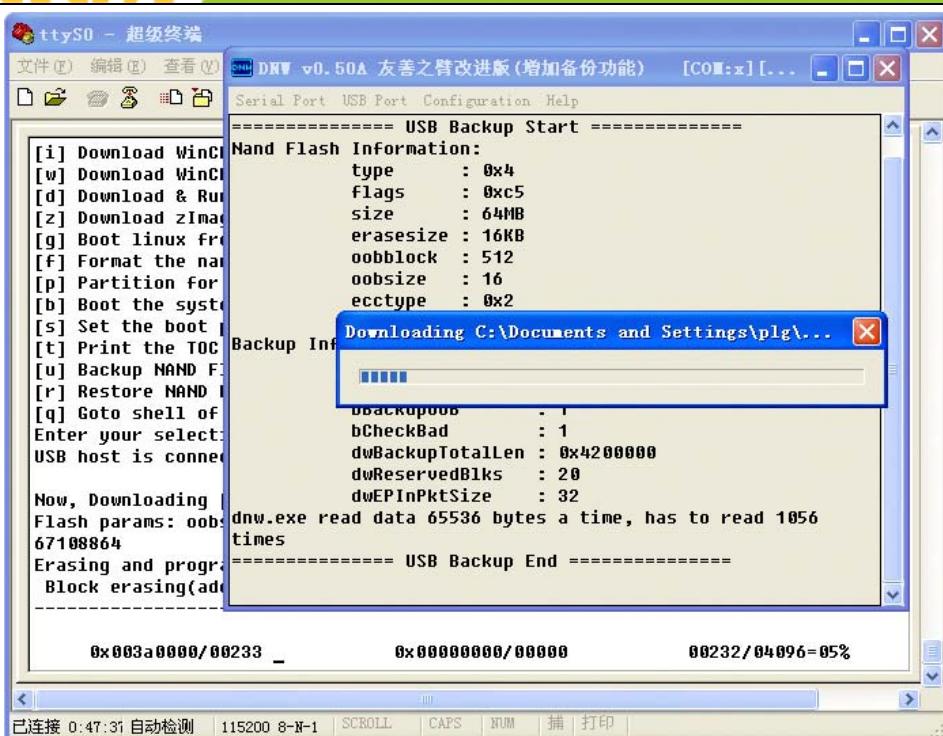
TO DO GREAT

广州友善之臂计算机科技有限公司



跳出文件选择窗口，选择要使用的备份文件(如上一步骤所生成的 backup.bin)，点“打开”开始恢复系，如图所示：





备份完毕，就可以把开发板设置为 Nand Flash 启动，按复位或者电源开关重新启动系统了。

## 3.2 安装 Linux 系统

**注意：本小节假定您已经按照前面的方法安装了 USB 驱动，并把开发板设置为 NOR Flash 启动，系统更新和安装完毕请设置为 Nand Flash 启动，设置方法请参考前面的章节。**

说明：安装 linux 所需要的二进制文件位于光盘的 **image/linux** 目录中。

安装 Linux 系统主要有以下步骤：

(1) 对 Nand Flash 进行分区

(2) 安装 bootloader

(3) 安装内核文件

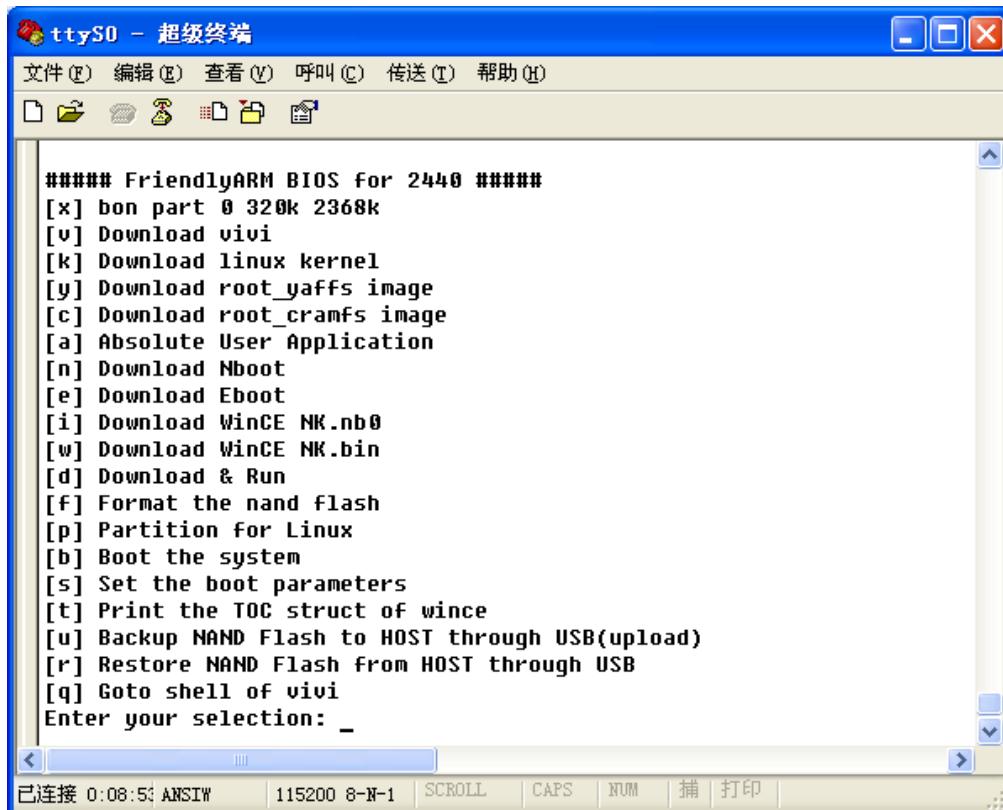
(4) 安装文件系统

下面是详细的步骤。

### 3.2.1 分区

**提示：分区将会擦除 Nand Flash 里面的所有数据**

(1) 连接好串口，打开超级终端，上电启动开发板，进入 BIOS 功能菜单：



(2) 选择功能号[x]开始对 Nand Flash 进行分区，如图所示。

说明：有的 Nand Flash 分区时会出现坏区报告提示，因为 supervivi 会对坏区做检测记录，因此这将不会影响板子的正常使用。



追求卓越 创造精品

TO BE BEST

TO DO GREAT

广州友善之臂计算机科技有限公司

```
[t] Print the TOC struct of wince
[q] Goto shell of vivi
Enter your selection: x
doing partition
size = 0
size = 327680
size = 2424832
check bad block
part = 0 end = 327680
part = 1 end = 2424832
part = 2 end = 67108864
part0:
    offset = 0
    size = 327680
    bad_block = 0
part1:
    offset = 327680
    size = 2097152
    bad_block = 0
part2:
    offset = 2424832
    size = 64667648
    bad_block = 0

##### FriendlyARM BIOS for 2440 #####
[x] bon part 0 320k 2368k
[v] Download vivi
Ready
```

Serial: COM1 | 27, 23 | 27 Rows, 73 Cols | VT100 | NUM

### 3.2.2 安装 bootloader

(1) 打开 DNW 程序，接上 USB 电缆，如果 DNW 标题栏提示[USB: OK]，说明 USB 连接成功，这时根据菜单选择功能号[v]开始下载 supervivi



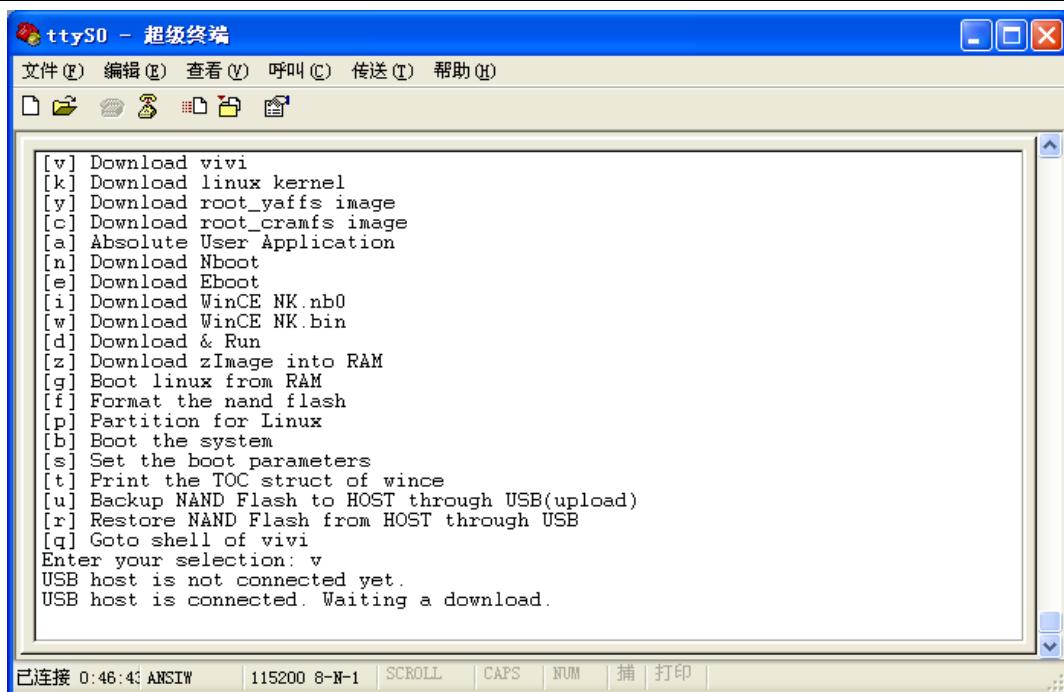


追求卓越 创造精品

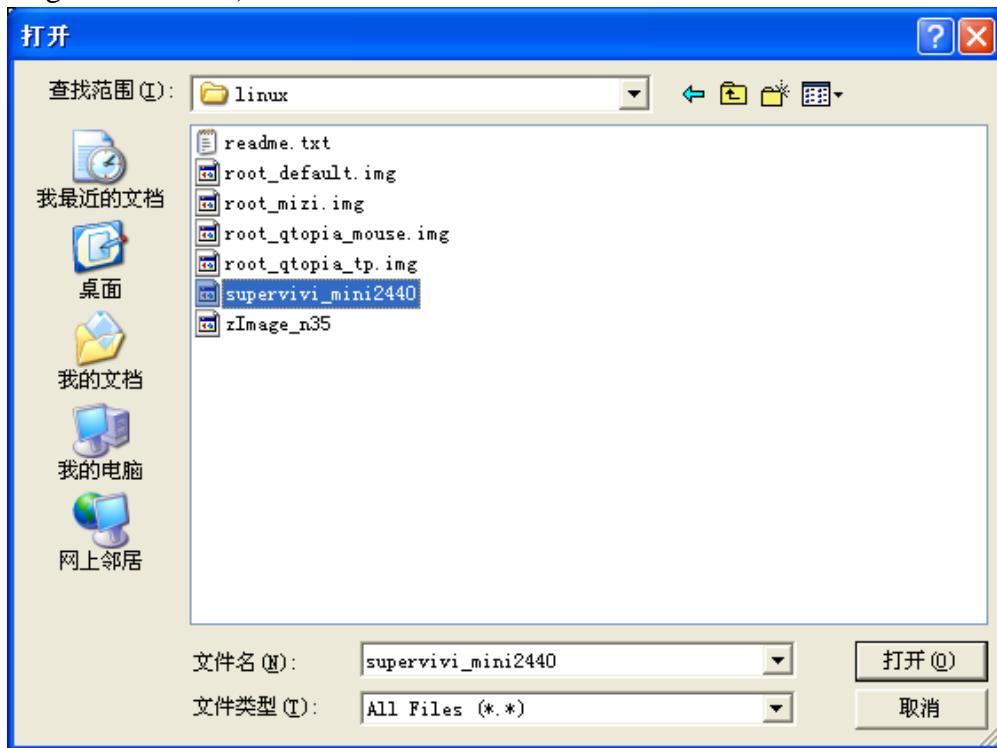
TO BE BEST

TO DO GREAT

广州友善之臂计算机科技有限公司



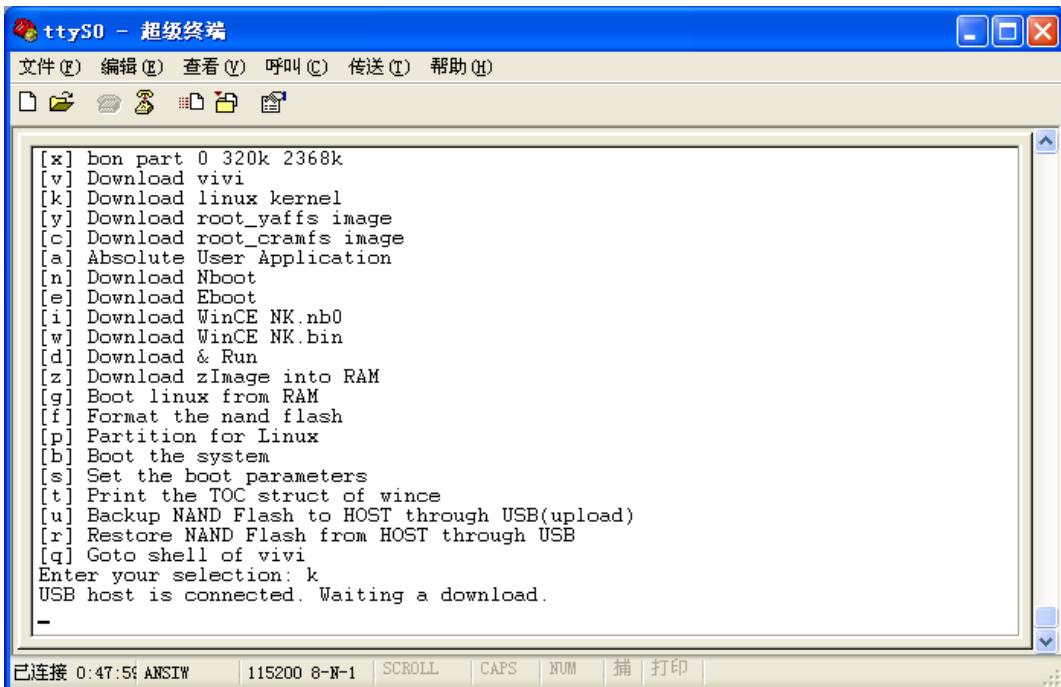
(3)点击“USB Port->Transmit/Restore”选项，并选择打开文件 supervivi(该文件位于光盘的 images/linux/目录)开始下载。



(4)下载完毕，BIOS 会自动烧写 supervivi 到 Nand Flash 分区中，并返回到主菜单。

### 3.2.3 安装 linux 内核

(1) 在 BIOS 主菜单中选择功能号[k], 开始下载 linux 内核 zImage



(2) 点击“USB Port->Transmit”选项，并选择打开相应的内核文件 zImage(该文件位于光盘的 images/linux/目录)开始下载。

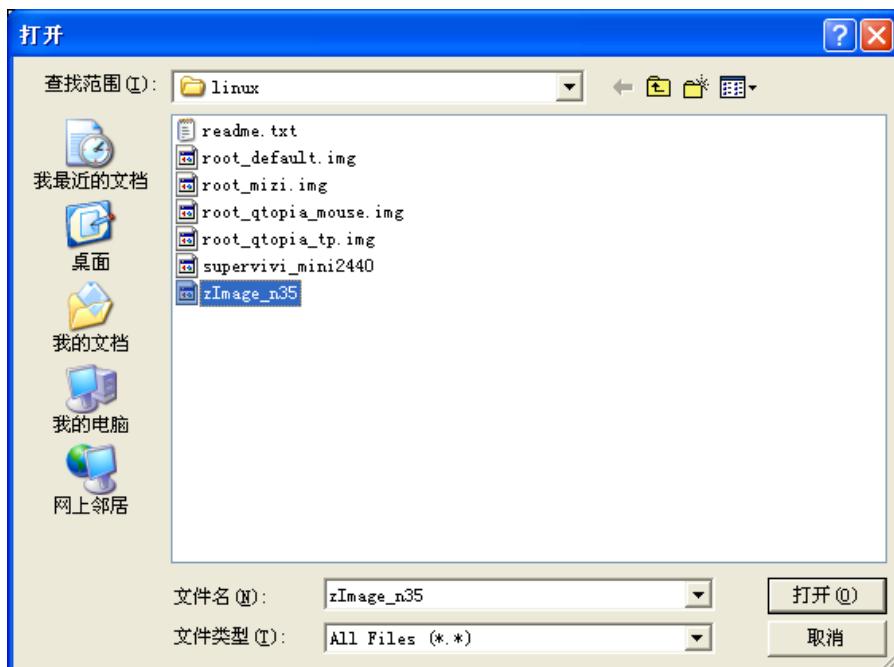
内核文件说明：

zImage\_n35 – 适用于 NEC3.5 寸 LCD

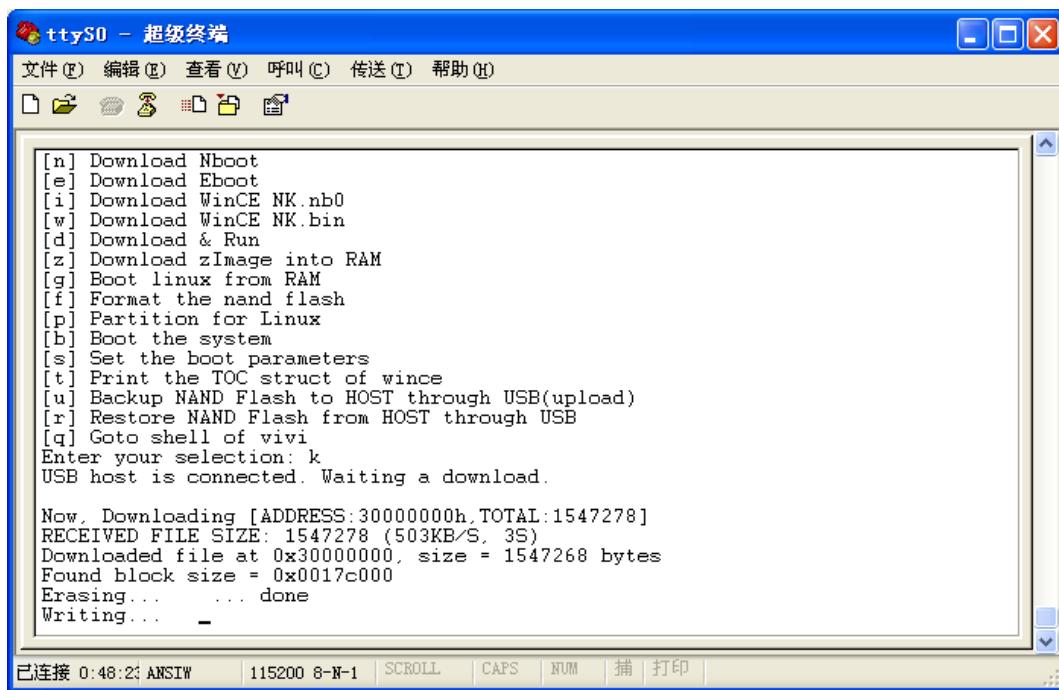
zImage\_a70 – 适用于 7 寸真彩屏，分辨率为 800x480

zImage\_VGA1024x768 – 适用于 VGA 模块输出，分辨率为 1024x768

实际可能与此不完全相同，请参考 images/linux/目录下的 readme.txt 文件说明



(3) 下载完毕，BIOS 会自动烧写内核到 Nand Flash 分区中，并返回到主菜单，如图：



### 3.2.4 安装根文件系统

(1) 在 BIOS 主菜单中选择功能号[y]，开始下载 yaffs 根文件系统映象文件

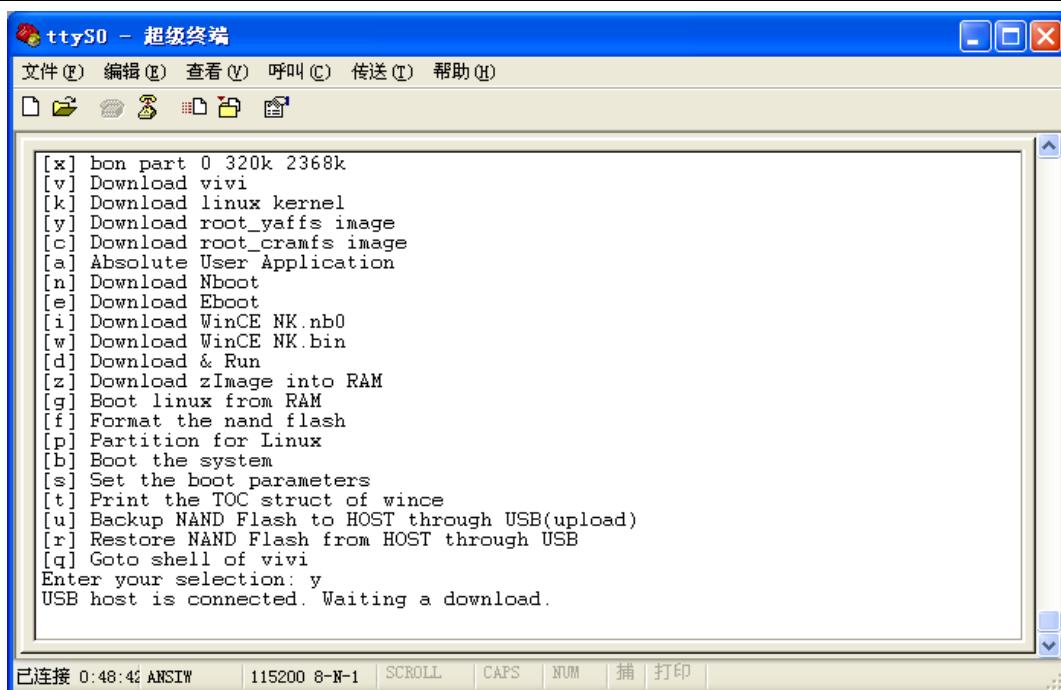


追求卓越 创造精品

TO BE BEST

TO DO GREAT

广州友善之臂计算机科技有限公司



(2) 点击“USB Port->Transmit/Restore”选项，并选择打开相应的文件系统映象文件root\_default.img(该文件位于光盘的images/linux目录)开始下载。

根文件系统映象文件说明：

root\_default.img

- 缺省安装的文件系统映象文件，采用基于 arm-linux-gcc-3.4.1 的函数库

root\_mizi.img

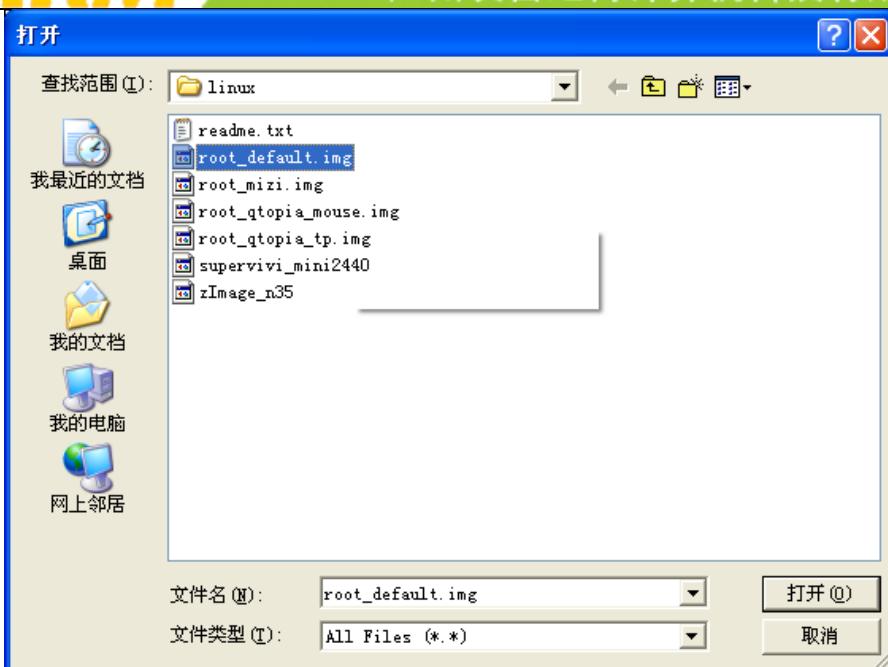
- mizi 公司提供的映象文件，含有中文手写识别，浏览器等。

root\_qtopia\_mouse.img

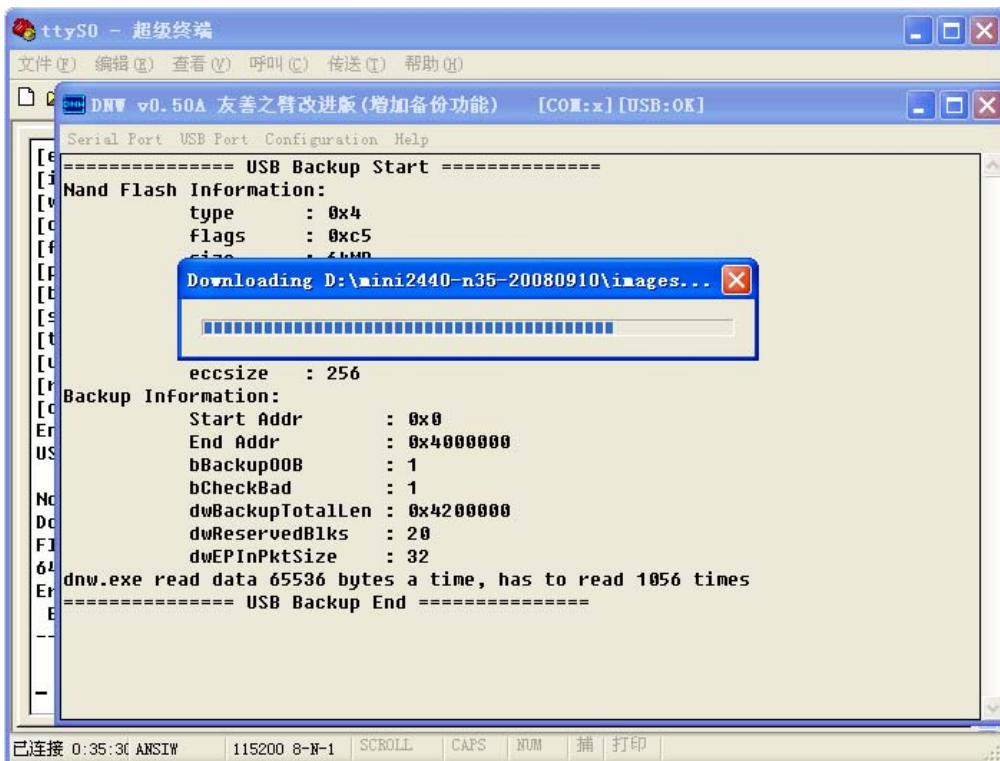
- 标准的 qtopia，使用鼠标操作,采用基于 arm-linux-gcc-3.4.1 的函数库

root\_qtopia\_tp.img

- 标准的 qtopia，使用触摸屏操作。采用基于 arm-linux-gcc-3.4.1 的函数库  
实际可能与此不完全相同，请参考 images/linux/目录下的 readme.txt 文件说明



(3) 下载过程如图所示，下载完毕，BIOS 会自动烧写内核到 Nand Flash 分区中，并返回到主菜单，如图：



提示：此过程大概需要 2-3 分钟，下载的文件越大，下载和烧写的时间就会越长。

下载完毕，请拔下 USB 连接线，如果不取下来，有可能在复位或者启动系统的时候

导致您的电脑死机。

在 BIOS 主菜单中选择功能号**[b]**, 将会启动系统。

如果您把开发板的启动模式设置为 Nand Flash 启动, 则系统会在上电后自动启动。

### 3.3 安装 WinCE 系统

**注意:** 本小节假定您已经按照前面的方法安装了 USB 驱动, 并把开发板设置为 Nor Flash 启动, 系统更新和安装完毕请设置为 Nand Flash 启动, 设置方法请参考前面的章节。

说明: 安装 WinCE 所需要的二进制文件位于光盘的 “\images\wince5.0” 目录中。

安装 WindowsCE 系统主要有以下步骤:

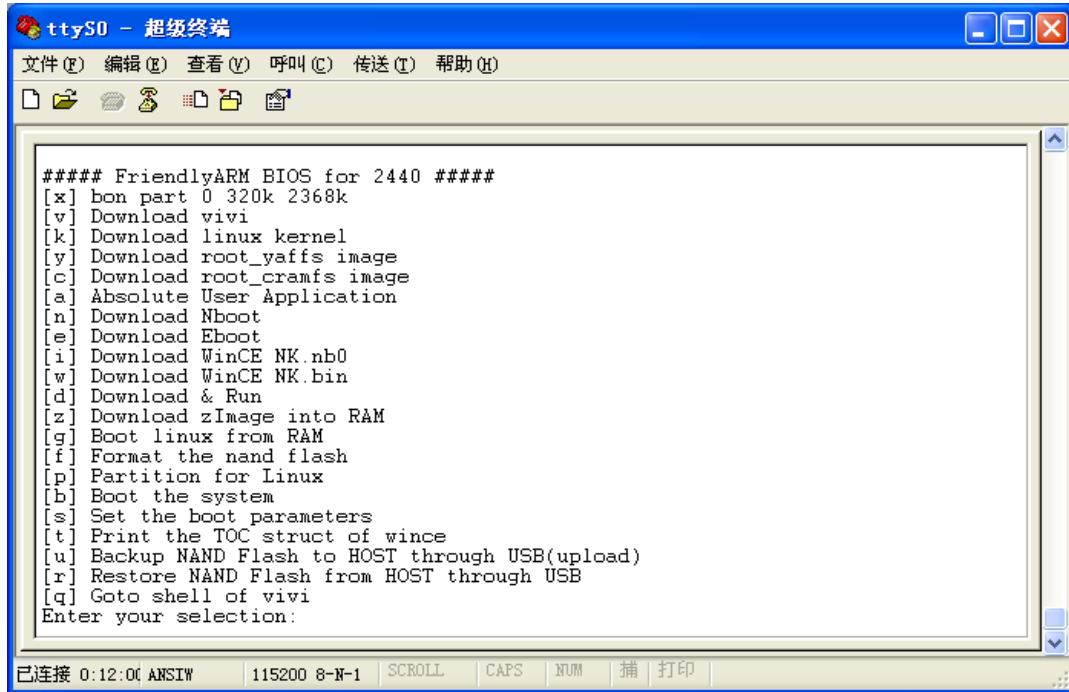
- (1) 分区
- (2) 安装 bootloader
- (3) 安装 Eboot
- (4) 安装 WindowsCE 内核映象

下面是详细的步骤。

#### 3.3.1 分区

**提示:** 分区将会擦除 Nand Flash 里面的所有数据

(1) 连接好串口, 打开超级终端, 上电启动开发板, 进入 BIOS 功能菜单:



(2) 选择功能号[x]开始对 Nand Flash 进行分区, 如图所示。

说明：有的 Nand Flash 分区时会出现坏区报告提示，因为 supervivi 会对坏区做检测记录，因此这将不会影响板子的正常使用。

```

[t] Print the TOC struct of wince
[q] Goto shell of vivi
Enter your selection: x
doing partition
size = 0
size = 327680
size = 2424832
check bad block
part = 0 end = 327680
part = 1 end = 2424832
part = 2 end = 67108864
part0:
    offset = 0
    size = 327680
    bad_block = 0
part1:
    offset = 327680
    size = 2097152
    bad_block = 0
part2:
    offset = 2424832
    size = 64667648
    bad_block = 0

##### FriendlyARM BIOS for 2440 #####
[x] bon part 0 320k 2368k
[v] Download vivi

```

### 3.3.2 安装 bootloader

本开发板提供了两种 bootloader 均可启动 WINCE: supervivi 和 nboot.bin，它们的说明如下：

	supervivi	Nboot
二进制映象文件的位置	\images\wince5.0	\images\wince5.0
源代码位置	无	WindowsCE5.0\NBOOT
项目文件	无	Nboot.mcp
编译器	Arm-linux-gcc	ADS1.2
说明：	<ul style="list-style-type: none"> <li>● supervivi 由友善之臂维护和发展，不提供源代码</li> <li>● NBOOT 的编译、下载烧写见 4.6 章节</li> </ul>	

下面是 supervivi 的下载烧写步骤：



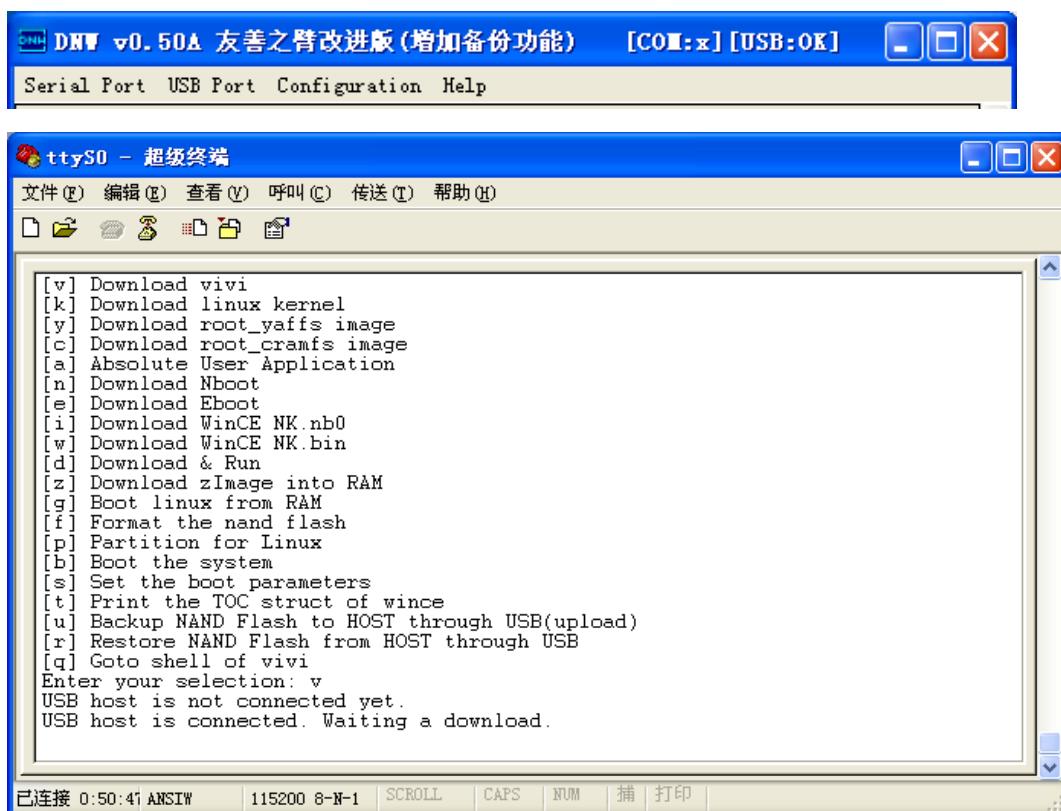
追求卓越 创造精品

TO BE BEST

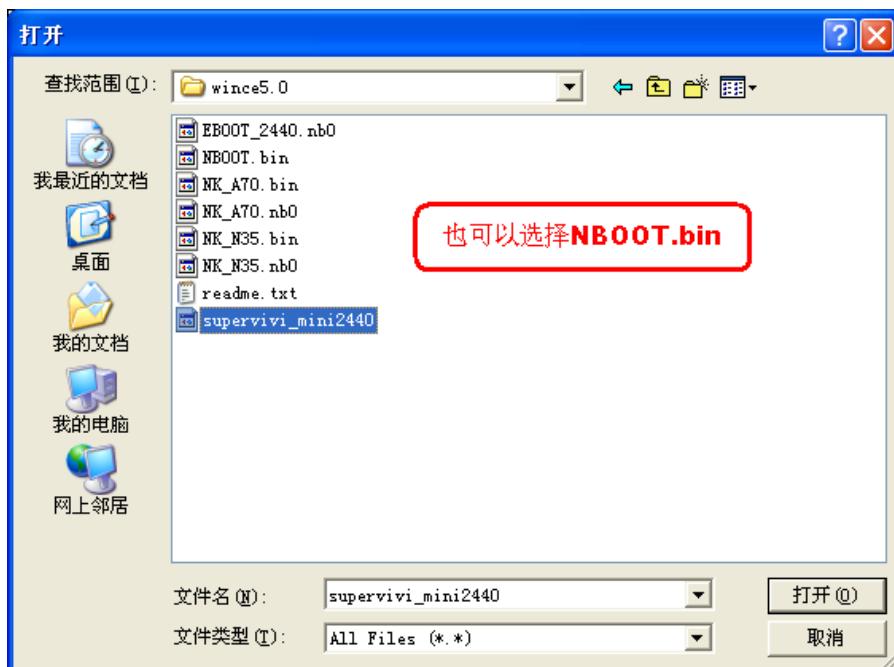
TO DO GREAT

广州友善之臂计算机科技有限公司

(1) 打开 DNW 程序，接上 USB 电缆，如果 DNW 标题栏提示[USB: OK]，说明 USB 连接成功，这时根据菜单选择功能号[v]开始下载 supervivi



(3)点击“USB Port->Transmit”选项，并选择打开文件 supervivi(该文件位于光盘的\images\wince5.0 目录)开始下载。

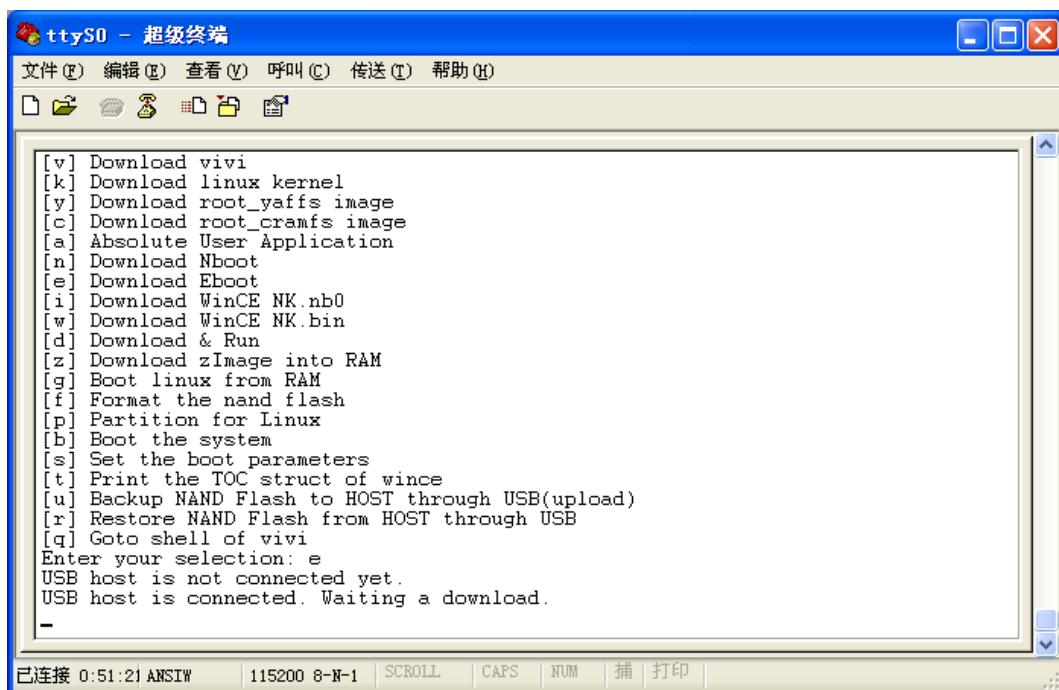


(4) 下载完毕，BIOS 会自动烧写 supervivi 到 Nand Flash 分区中，并返回到主菜单。

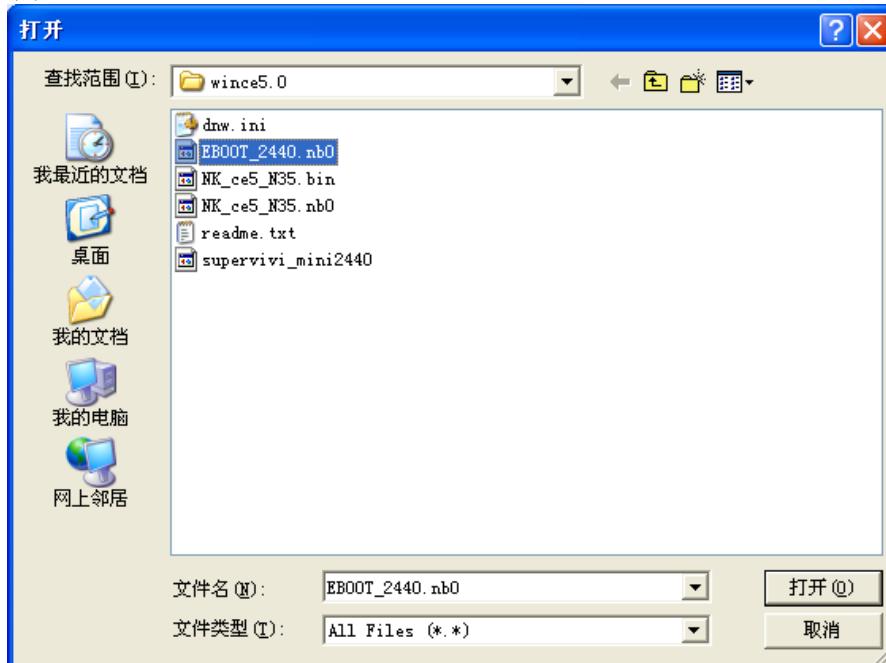
### 3.3.3 安装 eboot

**说明：**Eboot 在此的作用仅为烧写 nk.bin 之用，并无其他用途。

(1) 在 BIOS 主菜单中选择功能号[e]，开始下载 Eboot



(2) 点击“USB Port->Transmit/Restore”选项，并选择文件 Eboot\_2440.nb0



(3) 下载完毕，BIOS 会自动烧写 Eboot 到 Nand Flash 中，并返回到主菜单。

### 3.3.4 安装 wince 内核映象

(1) 在 BIOS 主菜单中选择功能号[w]，开始下载 WINCE 内核



追求卓越 创造精品

TO BE BEST

TO DO GREAT

广州友善之臂计算机科技有限公司

```
[i] Download WinCE NK.nb0
[w] Download WinCE NK.bin
[d] Download & Run
[f] Format the nand flash
[p] Partition for Linux
[b] Boot the system
[s] Set the boot parameters
[t] Print the TOC struct of wince
[u] Backup NAND Flash to HOST through USB(upload)
[r] Restore NAND Flash from HOST through USB
[q] Goto shell of vivi
Enter your selection: w
Clear the free memory .....
...
Read eboot image from flash .....
ReadImageFromNand: TOC is invalidate: 0x1, use default toc to load eboot
Sector addr on NAND: 0x100
TotalSector: 0x100
LoadAddress: 0x30038000
JumpAddr: 0x30038000

Now, to download the wince image(nk.bin) .....
USB host is connected. Waiting a download.
```

已连接 0:44:04 ANSIW | 115200 8-N-1 | SCROLL | CAPS | NUM | 插 | 打印 |

(2) 点击“USB Port->Transmit/Restore”选项，并选择打开相应的内核文件 NK.bin(该文件位于光盘的\images\wince5.0 目录)开始下载。

WINCE 内核文件说明：

NK\_N35.bin – 适用于 NEC 3.5" LCD 的内核文件

NK\_A70.bin – 适用于 7 寸真彩屏

NK\_VGA1024x768.bin – 适用于 VGA 模块输出，分辨率为 1024x768

实际可能与此不完全相同，请参考 images/wince5.0 目录下的 readme.txt 文件说明

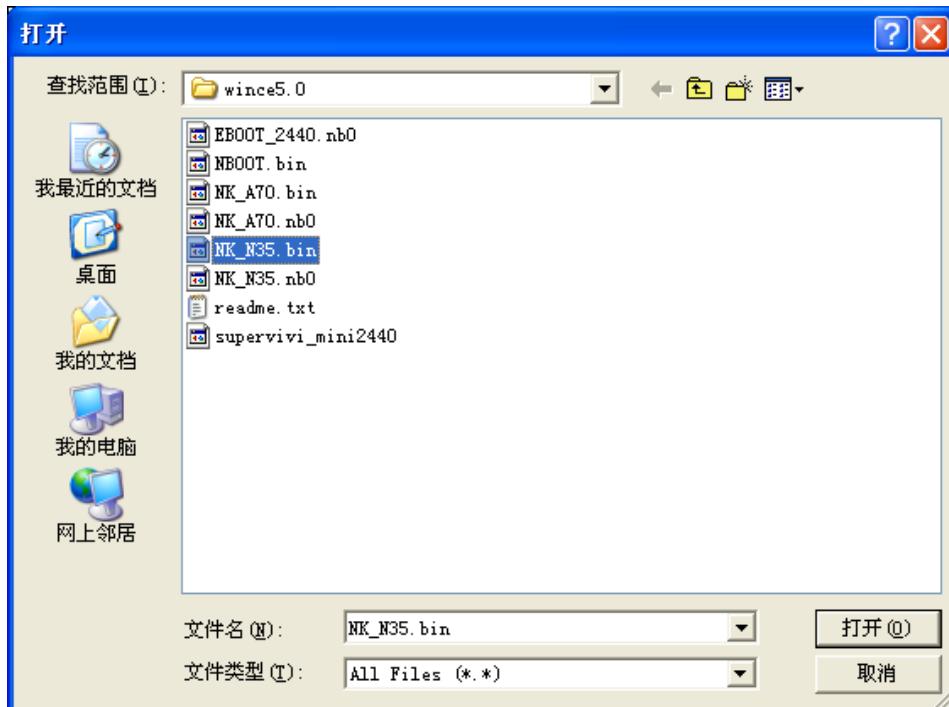


追求卓越 创造精品

TO BE BEST

TO DO GREAT

广州友善之臂计算机科技有限公司



下载完毕, BIOS 会自动调用 Eboot 的烧写功能开始格式化 Nand Flash, 并烧写 WINCE 内核映象文件。



追求卓越 创造精品

TO BE BEST

TO DO GREAT

广州友善之臂计算机科技有限公司

```
sgList[0].dwLength: 0x100
}
ID[1] {
    dwVersion: 0x1
    dwSignature: 0x43465349
    String: ''
    dwImageType: 0x6
    dwTtlSectors: 0x0
    dwLoadAddress: 0x0
    dwJumpAddress: 0x0
    dwStartOffset: 0x0
}
chainInfo.dwLoadAddress: 0X00000000
chainInfo.dwFlashAddress: 0X00000000
chainInfo.dwLength: 0X00000000
}
g_pViviWinceInfo = 0x301D8000, g_pViviWinceInfo->dwViviWinceMagic = 0x12345678
Low-level format nand flash ...
Reserving Blocks [0x0 - 0x13] ...
...reserve complete.
Low-level format Blocks [0x14 - 0xFFFF] ...
...erase complete.
Format nand flash for BinFS, please wait several minutes ...
```

已连接 1:01:45 ANSIW | 115200 8-N-1 | SCROLL | CAPS | NUM | 捕 | 打印 |

(3) 烧写完毕，WINCE 会自动运行，出现如图信息：

```

<PWR_Init:0x307b0
>PWR_Open(0x307b0, 0x0, 0x3)
<PWR_Open:1
>PWR_IoControl(0x321000, 0x0, 0, 0x6030850)
<PWR_IoControl:1
>PWR_Open(0x307b0, 0x0, 0x3)
<PWR_Open:2
PWR_Close(0x307b0)
384 clock
        USB:OhcdPdd_Init
++InitializeOHCI
USB: *pIrq=11, *pioPortBase=0x260000
OHCD: MapIrq2SysIntr(11): 27
OHCD: Memory Object
--InitializeOHCI
+CS8900:DriverEntry
USB enable interrupt
charlie::SDIO::SDHOST::SDCSDCardDllEntry::DLL_PROCESS_ATTACH
charlie::SDIO::SDCInitialize+
charlie::SDIO::SDCInitialize-
--S3C2440DISP::InitializeHardware
Touch Init
RasEntry 'USB Socket Default' Created
    
```

## 3.4 下载到内存运行

### 3.4.1 运行 2440test

文件信息		备注
文件名称和位置	光盘\images\2440test.bin	2440test_N35.bin 适用于 NEC3.5 寸屏; 2440test_A70.bin 适用于 7 寸屏 2440test_VGA1024x768.bin 适用于 VGA 模块，分辨率 1024x768
指定下载运行地址	0x30000000	
对应的源代码位置	非操作系统示例代码\2440test	
项目名称	2440test.mcp	默认项目使用 NEC3.5 寸屏
编译工具	ADS1.2	
说明：	<ul style="list-style-type: none"> <li>若要烧写到 Nand Flash 运行，需选择 supervivi 的[a]功能，无需指定下载地址</li> <li>通过修改 “\非操作系统示例代码\2440test\inc\Option.h” 中 LCD_TYPE 的定义，</li> </ul>	



追 求 卓 越 创 造 精 品

TO BE BEST

TO DO GREAT

广州友善之臂计算机科技有限公司

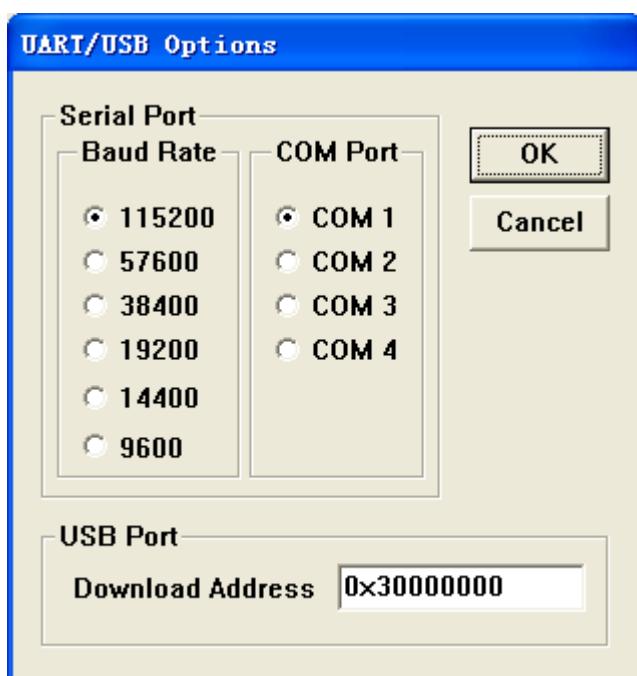
可 以 编 译 出 2440test\_N35.bin 或 者 2440test\_A70 或 者  
2440test\_VGA1024x768.bin, 见 4.3 章节

(1)连接好开发板电源, 串口线, USB 线, 并**设置拨动开关 S2 为 Nor Flash 启动系统**, 分别打开串口超级终端和 DNW, 上电启动开发板。

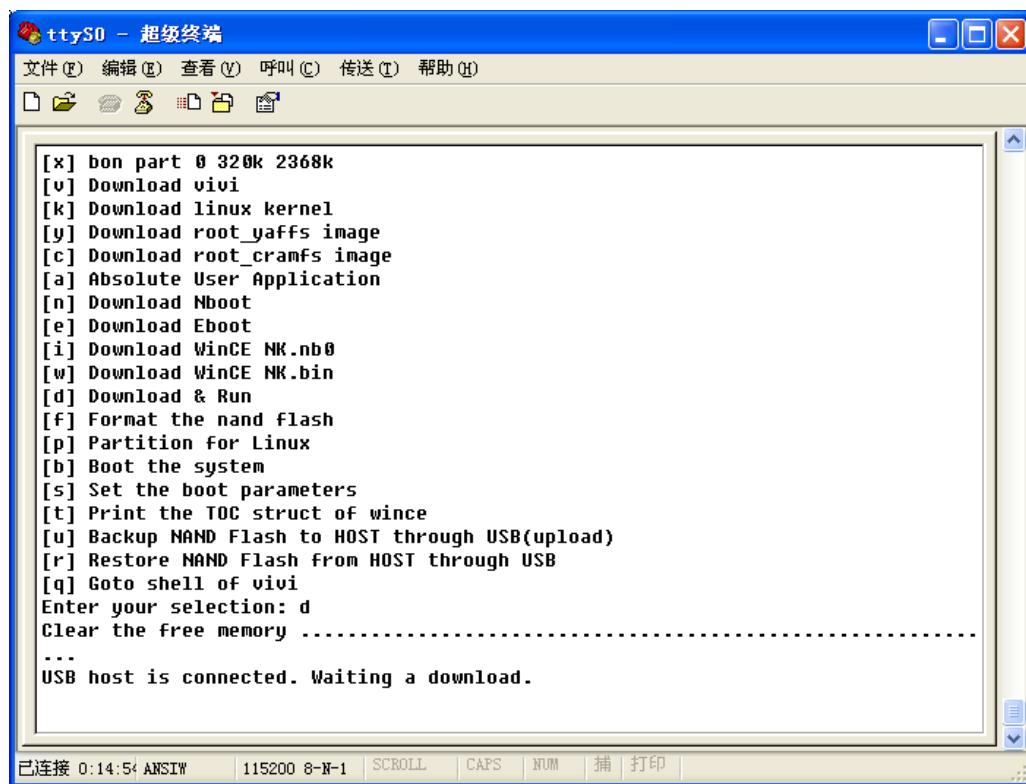
(2)保证 USB 驱动已经安装好(前面已经详细介绍了 USB 驱动的安装方法), 这时可以看到 DNW 的标题栏显示[USB: OK], 如果没有安装好驱动会显示[USB: x], 如图所示:



(3)点 DNW 菜单 Configuration, 设置 USB 下载运行地址为 0x30000000



(4)这时在超级终端的 BIOS 功能菜单中选择功能号[d], 出现 USB 下载等待提示信息:



(5)点击 DNW 程序的“USB Port”→“Transmit”，选择 2440test.bin 映象文件(在光盘的 images 目录下面)，接着点“打开”，这样就开始下载了。

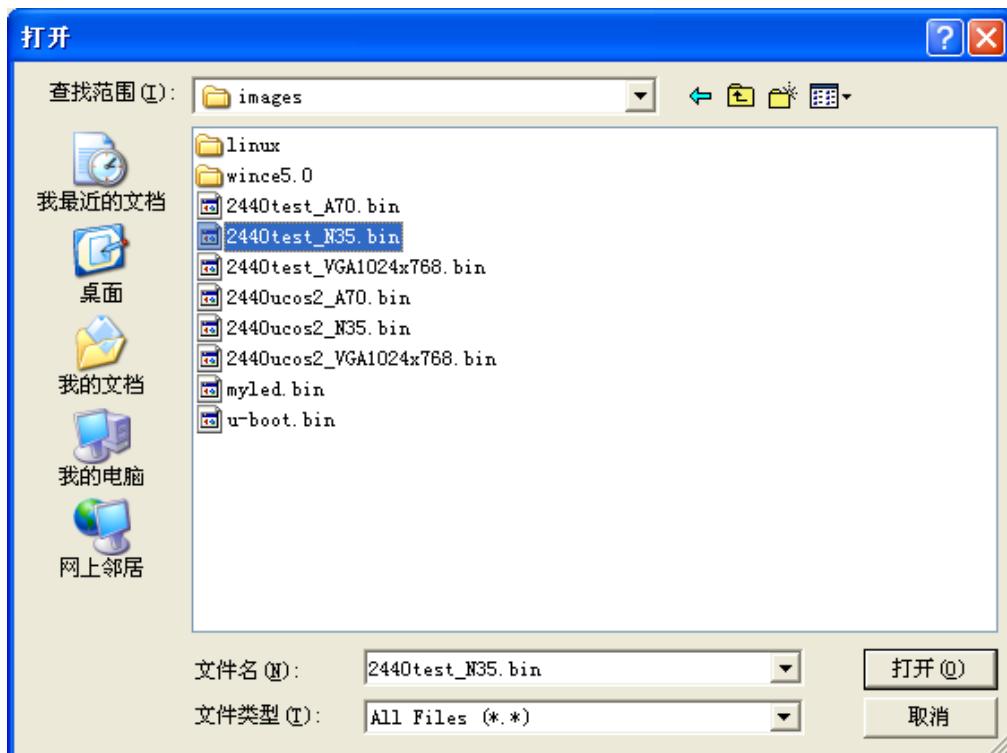


追求卓越 创造精品

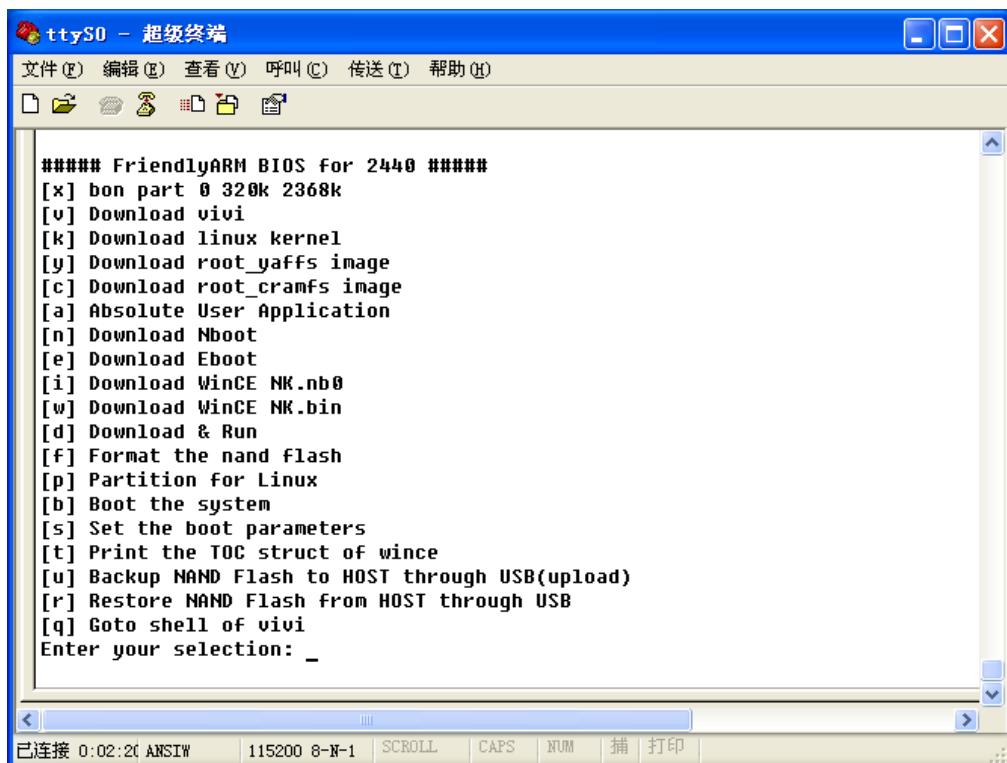
TO BE BEST

TO DO GREAT

广州友善之臂计算机科技有限公司



(6) 下载结束后，会自动运行，出现如下界面，就可以按照 2.3 章节来操作测试了：



若使用 7 寸真彩屏，会出现如下界面：

## FriendlyARM Demo

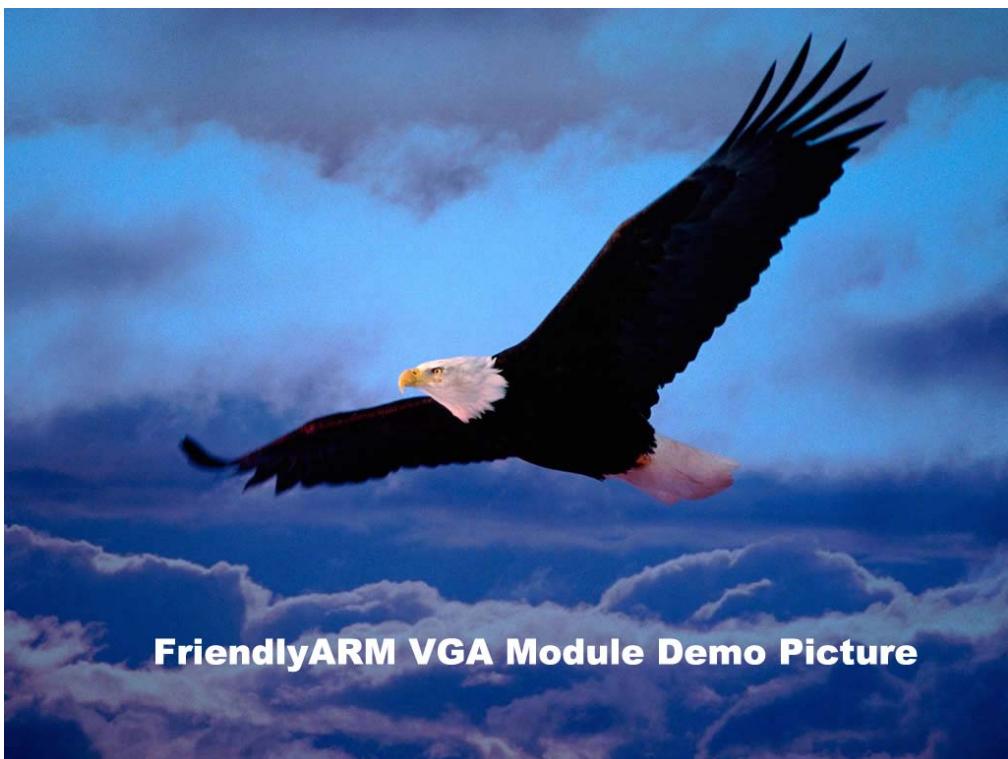


[arm9.net](http://arm9.net)

若使用 NEC3.5 寸屏，会出现如下界面：



若使用 VGA 模块板，会出现如下界面：



### 3.4.2 运行 uCos2

文件信息		备注
文件名称和位置	<a href="#">光盘\images\2440uCOS2.bin</a>	2440uCOS2_N35.bin 适用于 NEC3.5 寸屏； 2440uCOS2_A70.bin 适用于 7 寸屏 2440uCOS2_VGA1024x768.bin 适用于 VGA 显示输出，分辨率：1024x768
指定下载运行地址	<b>0x30000000</b>	
对应的源代码位置	uCOS2\uCOS2	
项目名称	uCOS_2440.mcp	默认项目使用 NEC3.5 寸屏
编译工具	ADS1.2	
说明：	<ul style="list-style-type: none"> <li>若要烧写到 Nand Flash 运行，需选择 supervivi 的[a]功能，无需指定下载地址</li> <li>通过修改 “uCOS2\uCOS2\S3C2440\includes\lcd.h” 中 LCD_TYPE 的定义，可以编译出 2440uCOS2_N35.bin 或者 2440uCOS2_A70 或者 2440uCOS2_VGA1024x768.bin，见 4.4 章节</li> </ul>	

(1)连接好开发板电源，串口线，USB 线，并**设置拨动开关 S2 为 Nor Flash 启动系统**，分别打开串口超级终端和 DNW，上电启动开发板。

(2)保证 USB 驱动已经安装好(前面已经详细介绍了 USB 驱动的安装方法)，这时可以看到 DNW 的标题栏显示[USB: OK]，如果没有安装好驱动会显示[USB: x]，如图所示：



追求卓越 创造精品

TO BE BEST

TO DO GREAT

广州友善之臂计算机科技有限公司



(3)点 DNW 菜单 Configuration，设置 USB 下载运行地址为 0x30000000



(4)这时在超级终端的 BIOS 功能菜单中选择功能号[d]，出现 USB 下载等待提示信息：



追求卓越 创造精品

TO BE BEST

TO DO GREAT

广州友善之臂计算机科技有限公司

```
[x] bon part 0 320k 2368k
[v] Download vivi
[k] Download linux kernel
[y] Download root_yaffs image
[c] Download root_cramfs image
[a] Absolute User Application
[n] Download Nboot
[e] Download Eboot
[i] Download WinCE NK.nb8
[w] Download WinCE NK.bin
[d] Download & Run
[f] Format the nand flash
[p] Partition for Linux
[b] Boot the system
[s] Set the boot parameters
[t] Print the TOC struct of wince
[u] Backup NAND Flash to HOST through USB(upload)
[r] Restore NAND Flash from HOST through USB
[q] Goto shell of vivi
Enter your selection: d
Clear the free memory .....
...
USB host is connected. Waiting a download.
```

(5)点击 DNW 程序的“USB Port”→“Transmit”，选择 2440test.bin 映象文件(在光盘的 images 目录下面)，接着点“打开”，这样就开始下载了。

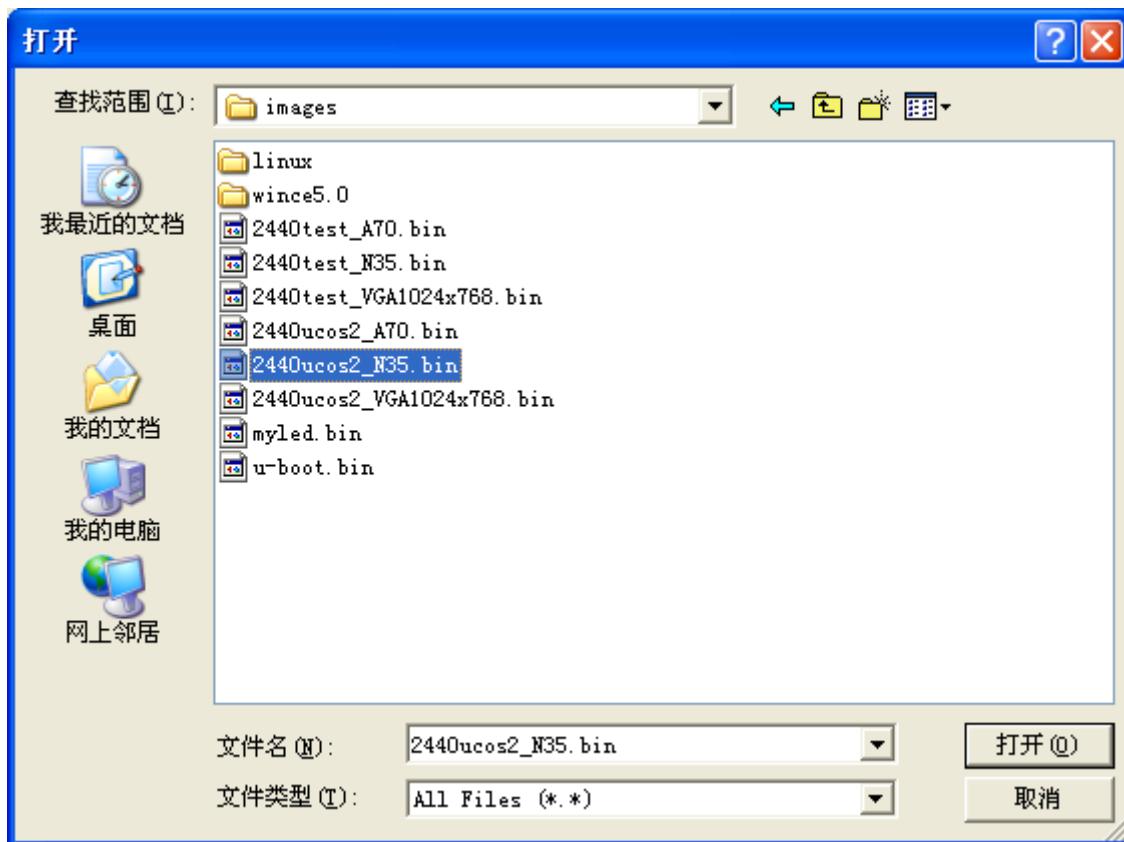


追求卓越 创造精品

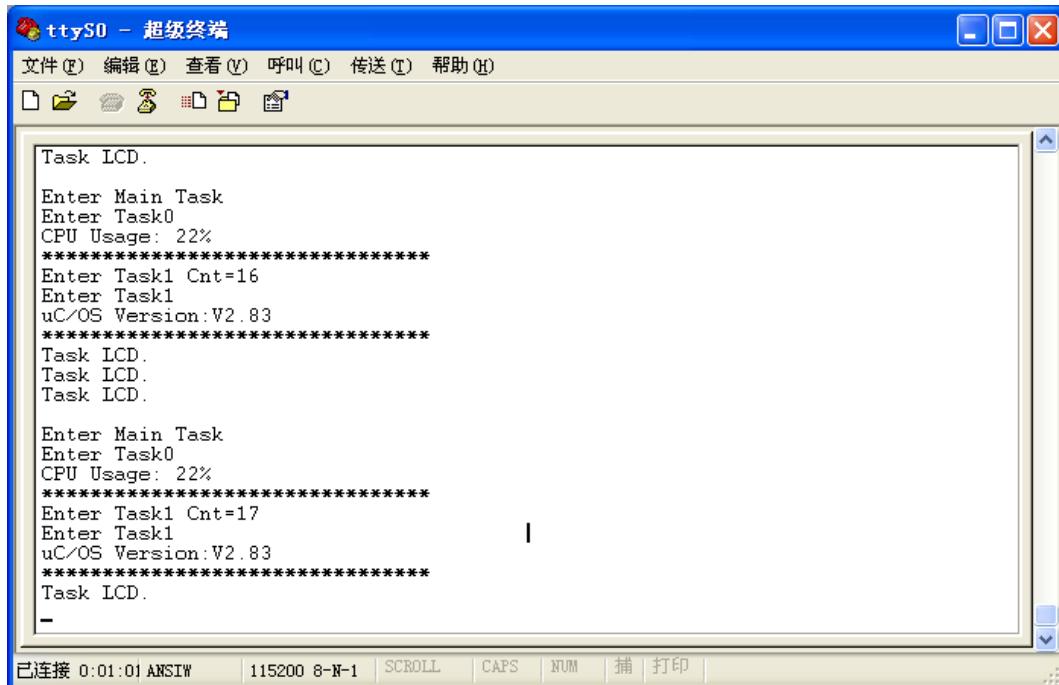
TO BE BEST

TO DO GREAT

广州友善之臂计算机科技有限公司



(6) 下载结束后，会自动运行，出现如下界面，：



使用 7 寸屏时，会出现如下图片背景的界面：



追求卓越 创造精品

TO BE BEST

TO DO GREAT

广州友善之臂计算机科技有限公司



使用 NEC3.5 寸屏时，会出现如下图片背景的界面：



使用 VGA 模块时，会出现如下图片背景的界面：

## FriendlyARM 演示图片: 纯真中国



### 3.4.3 运行 Linux

文件信息		备注
二进制文件位置	<a href="#">光盘\images\linux\zImage</a>	zImage_N35 适用于 NEC3.5 寸屏; zImage_A70 适用于 7 寸屏 zImage_VGA1024x768 适用于 VGA 显示输出, 分辨率: 1024x768
指定下载运行地址	<a href="#">0x30008000</a>	该地址无需通过 dnw 指定
对应的源代码包位置	<a href="#">linux-2.6.13-mini2440-20080910.tgz</a>	因为内核经常更新, 请以最新日期为准
项目名称	无	
编译工具	Arm-linux-gcc-3.4.1	
说明: 配置和编译内核见第 8 章		

说明: 在内存中运行 linux 系统, 一般是指 linux 内核(具体为 zImage 文件), 文件系统是无法通过网络或者 USB 下载到内存中运行的。一般是借助于 linux 的启动命令, 指定 NFS(网络文件系统), 或者使用开发板“本地”文件系统, 如 yaffs(可通过 supervivi 的“y”命令烧写 root\_default.img 或者其他文件系统映象文件)。

#### 如何通过 linux 命令指定 NFS 启动系统?

在本开发板中, 首先进入 supervivi 菜单, 按“q”键进入 supervivi 的命令行模式, 输入(详细见 5.1.4 章节):

```
Supervivi>param      set      linux_cmd_line      "console=ttySAC0      root=/dev/nfs
```



追求卓越 创造精品

TO BE BEST

TO DO GREAT

广州友善之臂计算机科技有限公司

nfsroot=192.168.1.111:/opt/FriendlyARM/mini2440/root\_nfs

ip=192.168.1.70:192.168.1.111:192.168.1.111:255.255.255.0:MINI2440.arm9.net:eth0:off"

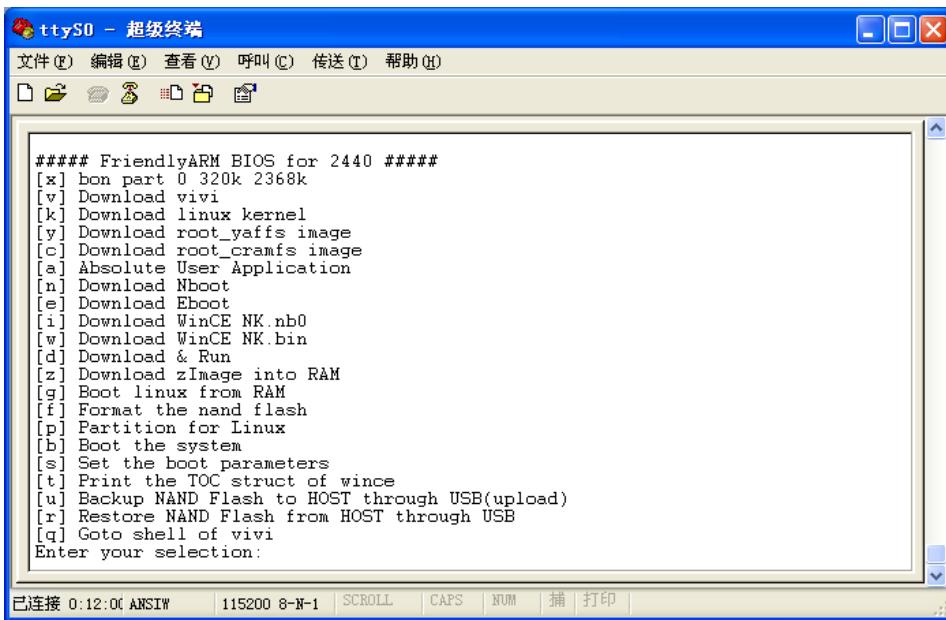
下面是使用 USB 下载 linux 内核到开发板中，并启动运行的步骤，这里使用的是“本地”文件系统 root\_default.img：

(1)连接好开发板电源，串口线，USB 线，并**设置拨动开关 S2 为 Nor Flash 启动系统**，分别打开串口超级终端和 DNW，上电启动开发板。

(2)保证 USB 驱动已经安装好(前面已经详细介绍了 USB 驱动的安装方法)，这时可以看到 DNW 的标题栏显示[USB: OK]，如果没有安装好驱动会显示[USB: x]，如图所示：

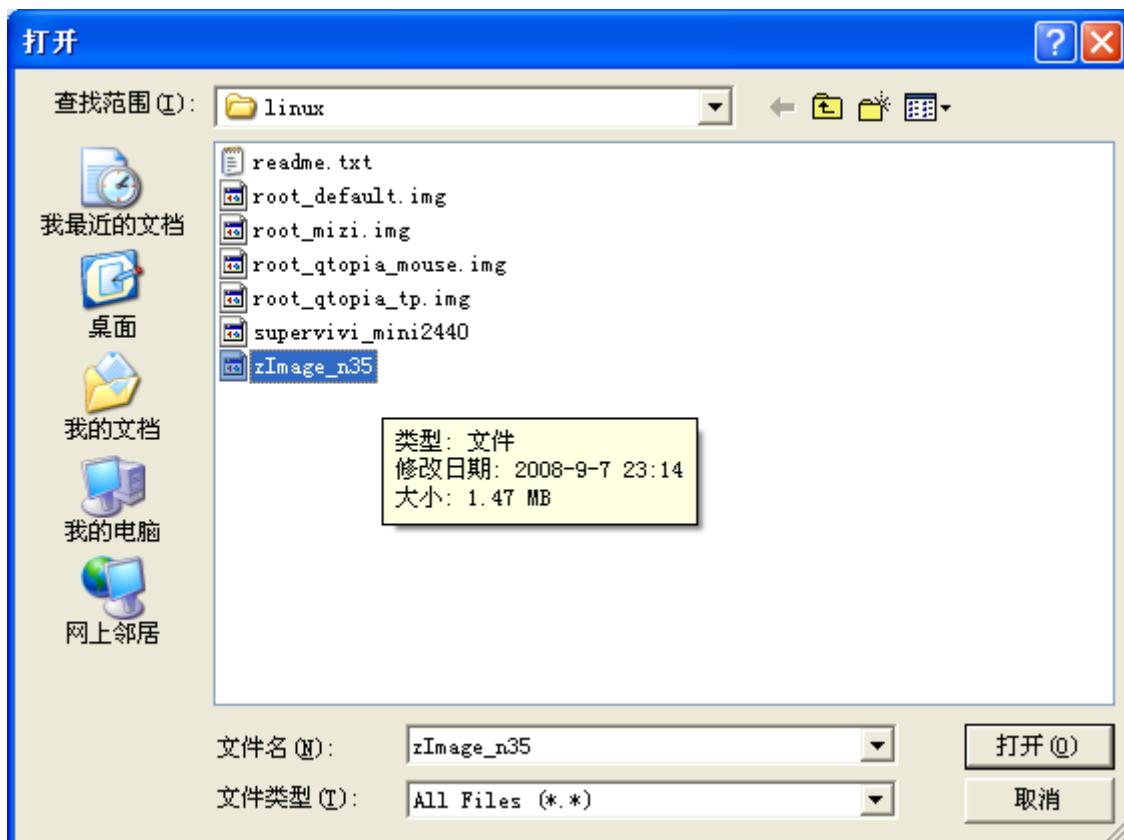


(3)这时在超级终端的 BIOS 功能菜单中选择功能号[z]，出现 USB 下载等待提示信息：



(4)点击 DNW 程序的“USB Port”→“Transmit”，选择 zImage\_n35 或者 zImage\_A70 这个映象文件(在光盘的 images\linux 目录下面)，接着点“打开”，这样就开始下载了。

**说明(仅供参考):** 功能[z]实际是把 zImage 文件下载到内存地址为 0x30008000 的地方，大小为 0x200000。按[q]进入 supervivi 的命令行模式，输入“**load ram 0x30008000 0x200000 u**”也可以实现相同的功能。



(5)下载结束后，回到 supervivi 菜单，这时按功能号[g]，就可以启动系统了。

**说明(仅供参考):** 功能[g]的功能实际是 supervivi 的命令行“**boot ram**”，在 supervivi 的命令行输入“**boot ram**”可以达到相同的功能效果。

若出现如下界面，则说明没有指定好文件系统，可以在 supervivi 菜单中选择[y]烧写一个 root\_default.img，或者使用 NFS 启动系统：

```

s3c2410-sdi s3c2410-sdi: running at 198kHz (requested: 197kHz).
s3c2410-sdi s3c2410-sdi: s3cmci_request: no card
s3c2410-sdi s3c2410-sdi: powered down.
IP route cache hash table entries: 1024 (order: 0, 4096 bytes)
TCP established hash table entries: 4096 (order: 3, 32768 bytes)
TCP bind hash table entries: 4096 (order: 2, 16384 bytes)
TCP: Hash tables configured (established 4096 bind 4096)
TCP reno registered
TCP bic registered
NET: Registered protocol family 1
NET: Registered protocol family 17
yaffs: dev is 32505858 name is "mtdblock2"
yaffs: Attempting MTD mount on 31.2, "mtdblock2"
VFS: Mounted root (yaffs filesystem).
mount_devfs_fs(): unable to mount devfs, err: -2
Freeing init memory: 144K
Warning: unable to open an initial console.
Kernel panic - not syncing: No init found. Try passing init= option to kernel.
-

```

已连接 0:18:23 ANSIW 115200 8-N-1 SCROLL CAPS NUM 插 打印

### 3.4.4 运行 WinCE

文件信息		备注
文件名和位置	<a href="#">光盘\images\wince5.0\NK.nb0</a>	NK_N35.nb0 适用于 NEC3.5 寸屏; NK_A70.nb0 适用于 7 寸屏 NK_VGA1024x768.nb0 适用于 VGA 显示输出，分辨率：1024x768
指定下载运行地址	<b>0x30200000</b>	必须在 dnw 中指定此下载运行地址， 详见以下操作
对应的源代码包位置	<a href="#">smdk2440 目录包</a>	
项目名称	<a href="#">mini2440.pbxml</a>	
编译工具	Platform Builder 5.0	
说明：	smdk2440 目录是本开发板的 wince 5.0 BSP，mini2440.pbxml 是相应的项目文件，按照 9.1 章节步骤可以编译出相应的 wince 内核映象文件 nk.bin 和 nk.nb0.	

**注意：**在内存中运行 nk.nb0，在其启动的时候，因为 wince 启动过程中目录的创建，会破坏 Nand Flash 中一些内容。如原来的 Eboot，或者 linux 内核等文件，从而导致原来的系统不再可用，在此请注意！

下面是使用 USB 下载 WINCE 内核到开发板内存中运行的步骤：

(1)连接好开发板电源，串口线，USB 线，并**设置拨动开关 S2 为 Nor Flash 启动系统**，



追求卓越 创造精品

TO BE BEST

TO DO GREAT

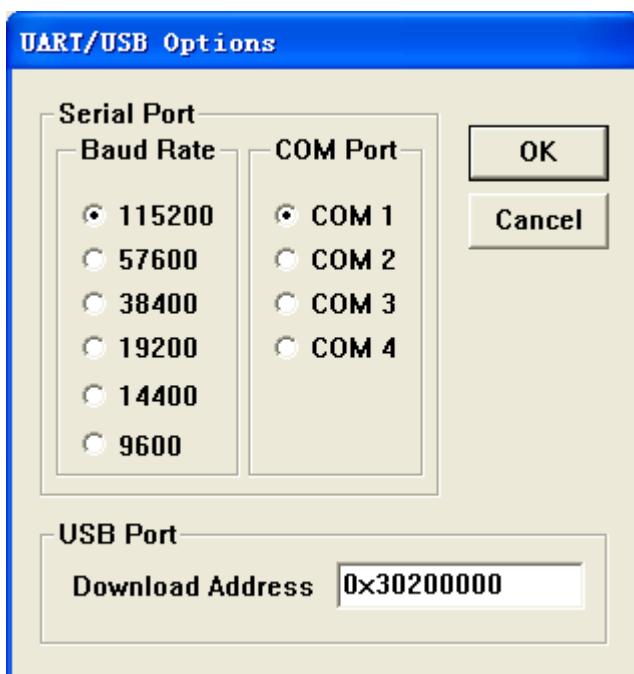
广州友善之臂计算机科技有限公司

分别打开串口超级终端和 DNW，上电启动开发板。

(2)保证 USB 驱动已经安装好(前面已经详细介绍了 USB 驱动的安装方法)，这时可以看到 DNW 的标题栏显示[USB: OK]，如果没有安装好驱动会显示[USB: x]，如图所示：



(3)点 DNW 菜单 Configuration，设置 USB 下载运行地址为 0x30200000





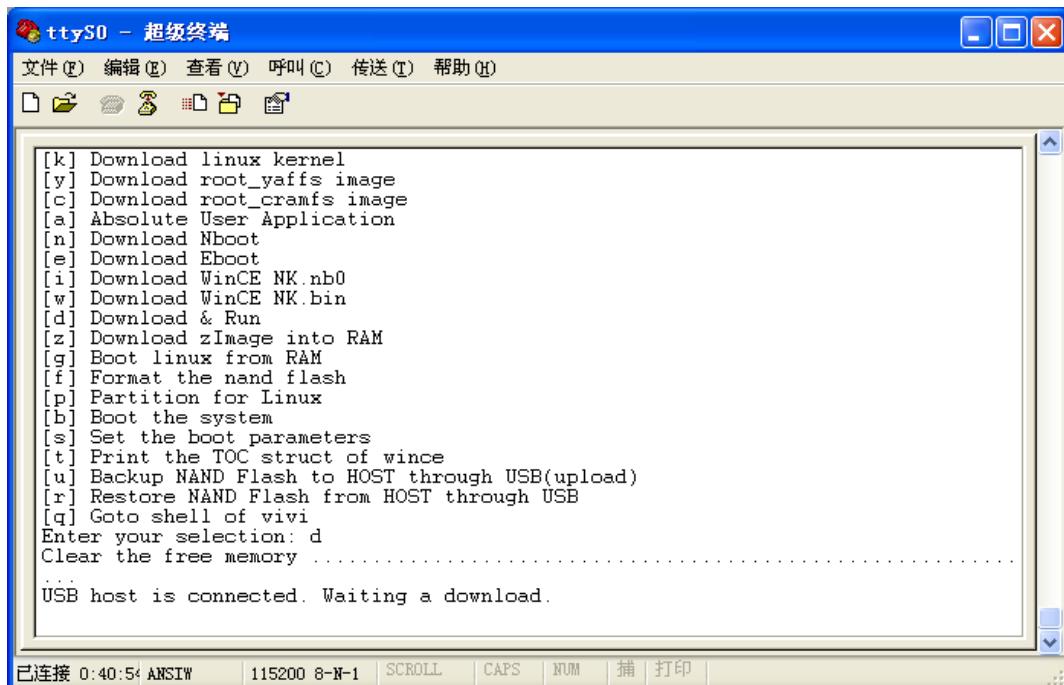
追求卓越 创造精品

TO BE BEST

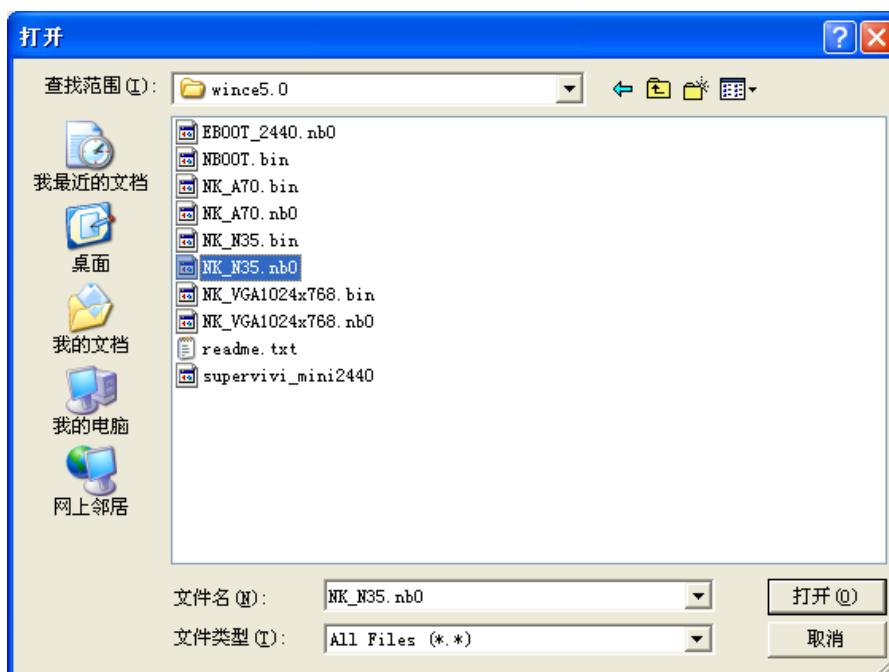
TO DO GREAT

广州友善之臂计算机科技有限公司

(4)这时在超级终端的 BIOS 功能菜单中选择功能号[d], 出现 USB 下载等待提示信息:



(5)点击 DNW 程序的“USB Port”→“Transmit”，选择 NK.nb0 映象文件(在光盘的 images\wince5.0 目录下面)，接着点“打开”，这样就开始下载了。



(6)下载结束后，会自动运行，不再返回 supervivi 菜单。这时 PC 机有可能会出现 USB 无法识别的提示，只要把 USB 拔下来，重新插上，就可以看到同步连接了。

## 第四章 ADS1.2 集成开发环境的使用

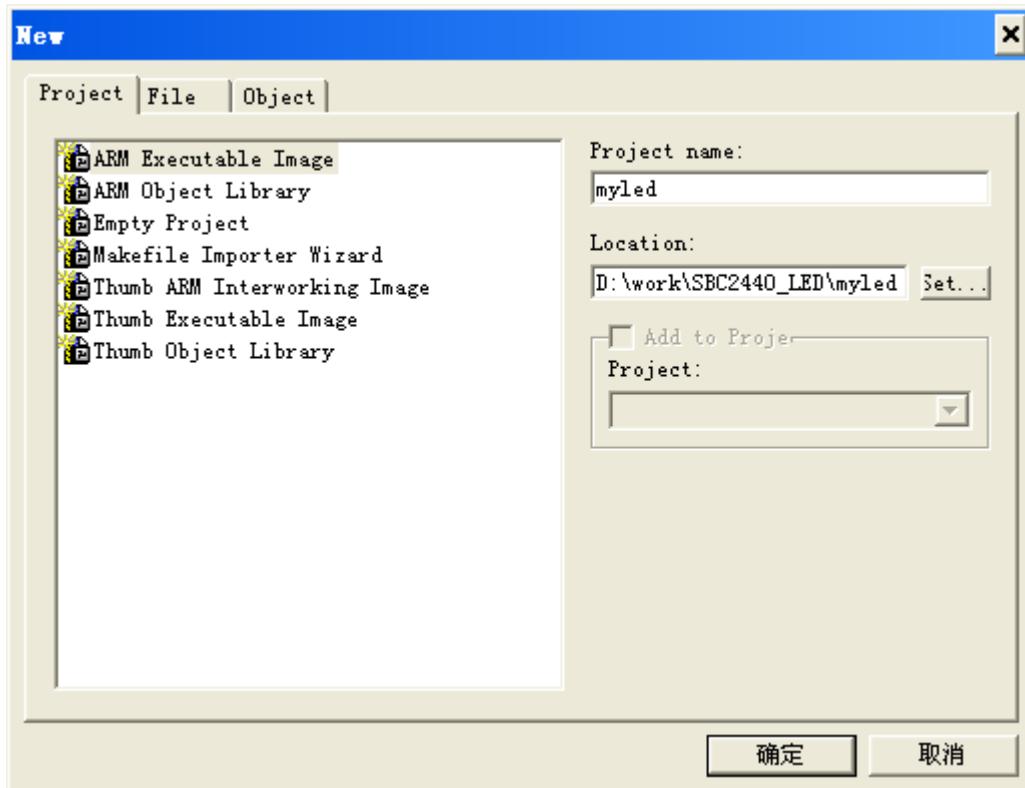
ARM ADS 的全称为 ARM Developer Suite，它是 ARM 公司推出的新一代 ARM 集成开发环境，我们使用的 ADS 为 1.2 版本，它取代了早期的 ADS1.1 和 ADS1.0，它可以安装在 WindowsNT/2000/98/95/XP 上面使用。

### 4.1 使用 ADS 创建 LED 工程

本节通过一个简单的具体实例，介绍如何使用 ADS 集成开发环境。包括如何创建一个新的工程，如何配置编译选项，并编译生成可以直接烧写到 Flash 中的 bin 格式二进制可执行文件。

#### 4.1.1 建立一个工程

在 ADS 集成开发环境中，点 File->New，打开如图所示窗口：



可以看到有 7 种工程类型可以选择：

**ARM Executable Image:** 用于由 ARM 指令的代码生成一个 ELF 格式的可以执行映象

文件。

**ARM Object Library:** 用于由 ARM 指令的代码生成一个 armar 格式的目标文件库。

**Empty Project:** 用于创建一个不包含任何库或者源文件的工程。

**Makefile Importer Wizard:** 用于将 Visual C 的 nmake 或者 GNU make 文件转入到 CodeWarrior IDE 工程文件。

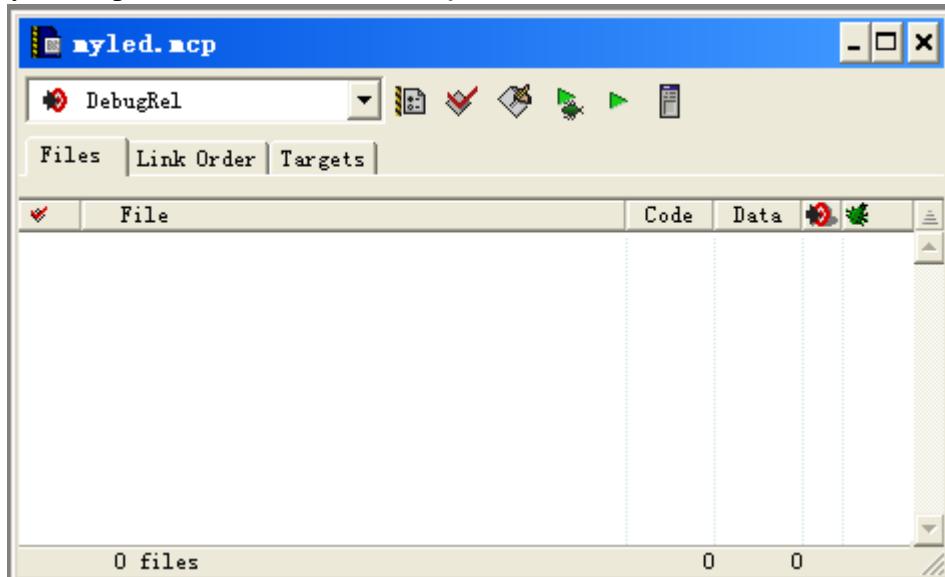
**Thumb ARM Executable Image:** 用于由 ARM 指令和 Thumb 指令的混和代码生成一个可执行的 ELF 格式的映象文件。

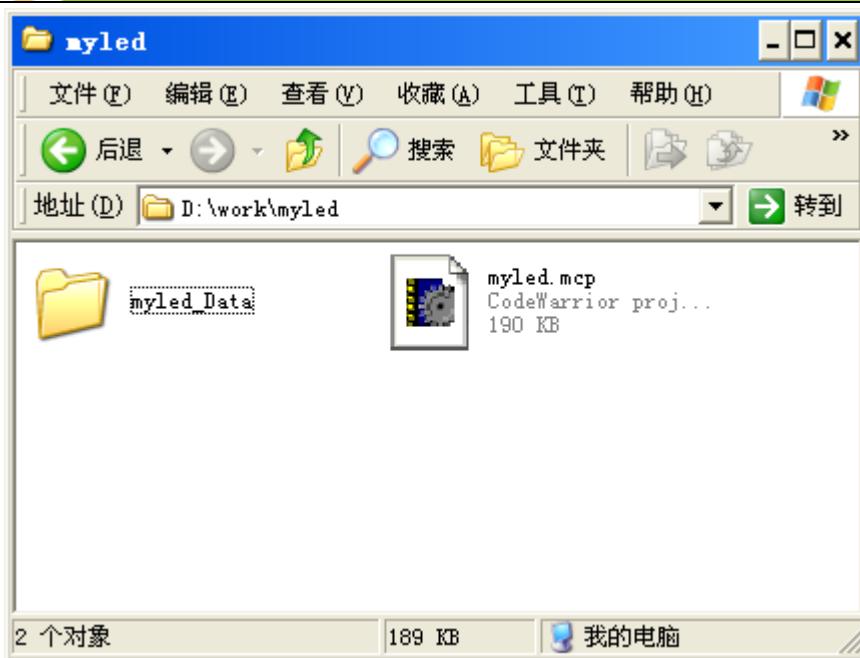
**Thumb Executable image:** 用于由 Thumb 指令创建一个可执行的 ELF 格式的映象文件。

**Thumb Object Library:** 用于由 Thumb 指令的代码生成一个 armar 格式的目标文件库。

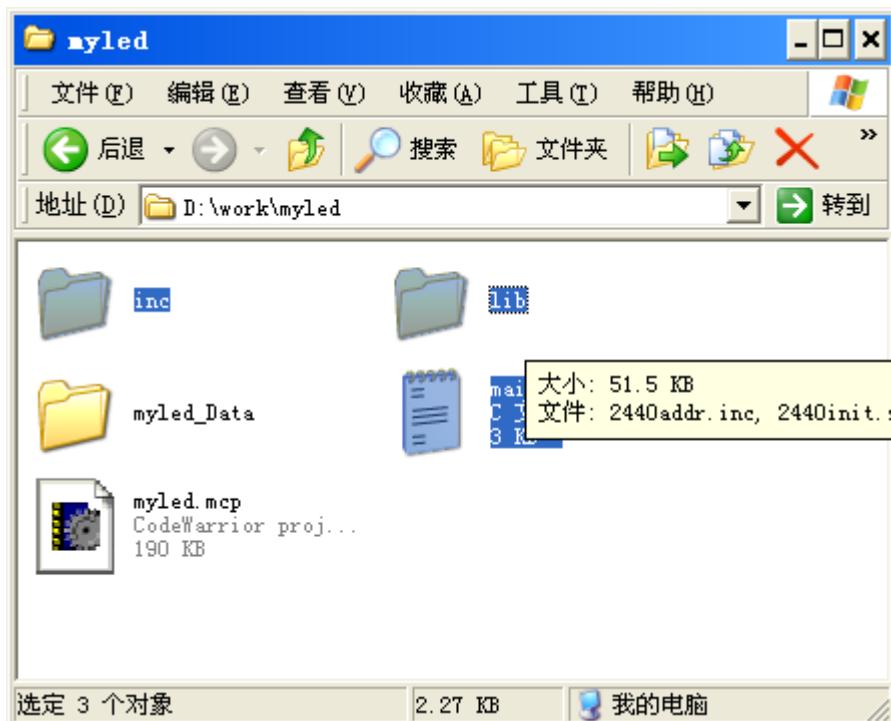
我们在这里选择 ARM Executable Image，在“Project name:”中输入工程文件名，本例为“myled”，点击“Location:”文本框的“Set”按钮，浏览选择想要保存该工程的路径(本例为“D:\work”)，将这些设置好之后，点击“确定”，即可创建一个新的名为 myled 的工程。

这个时候会出现 myled.mcp 窗口，如图所示，同时会在 D:\work 目录下创建一个工程目录 myled，而 myled.mcp 会出现在“D:\work\myled”目录中。





对于本例，我们将已经准备好的源文件及其目录（位于光盘的“非操作系统示例代码\myled”文件夹中）一起复制到 myled 工程目录，如图



然后在 myled.mcp 项目窗口中，点鼠标右键或者 ADS 菜单 Project->Add Files...，如图开始添加该项目需要的源代码。

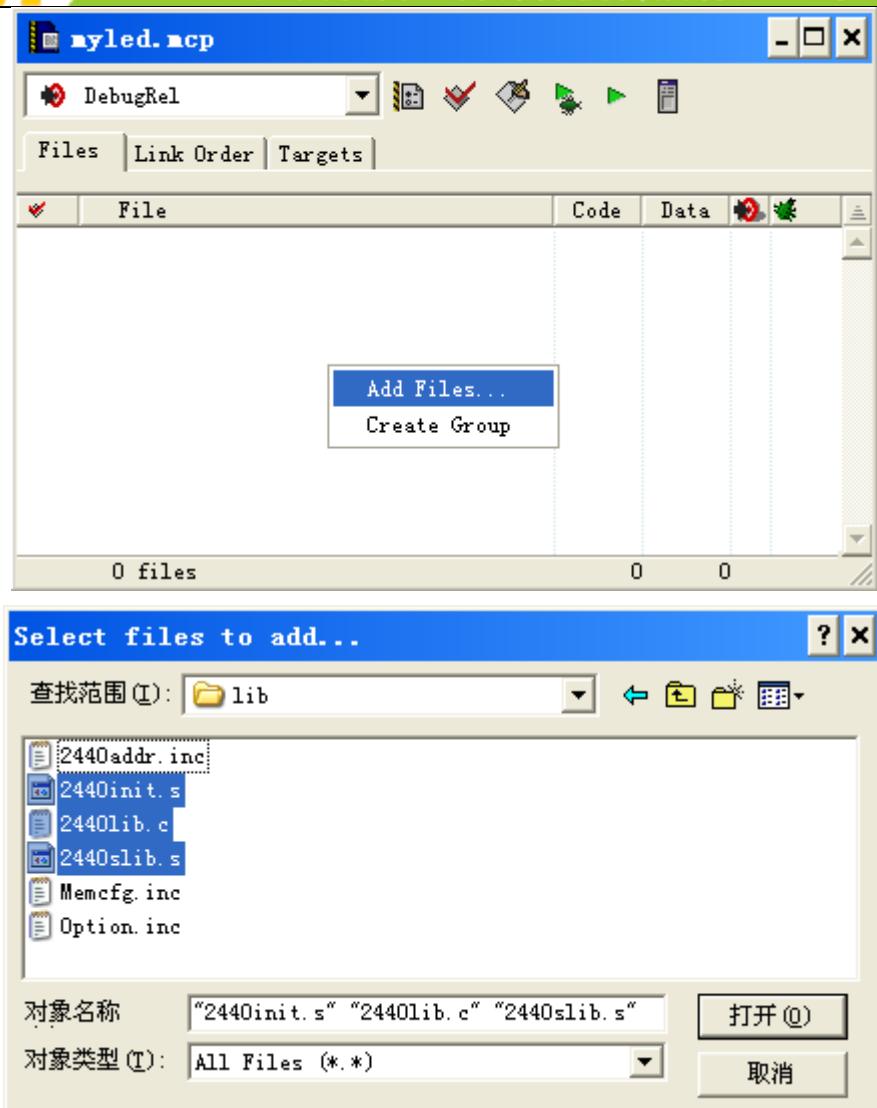


追求卓越 创造精品

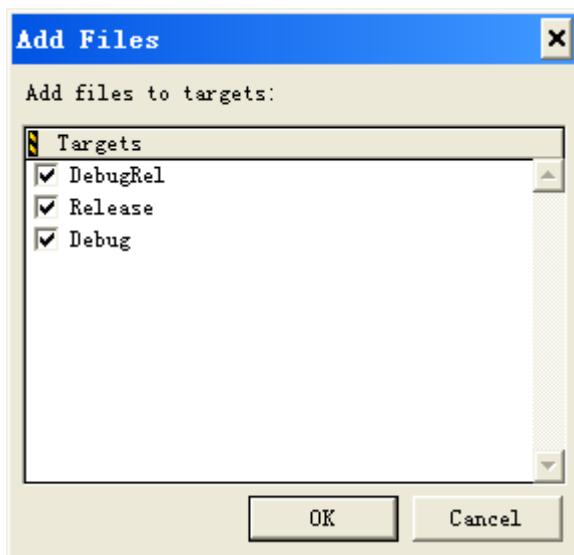
TO BE BEST

TO DO GREAT

广州友善之臂计算机科技有限公司



点击“打开”按钮确定，这时会跳出如下图所示的提示选择窗口



这里请注意，我们在新建一个工程时，ADS 默认的 target 是 DebugRel，另外还有两个可用的 target，分别为 Release 和 Debug，它们的含义分别为：

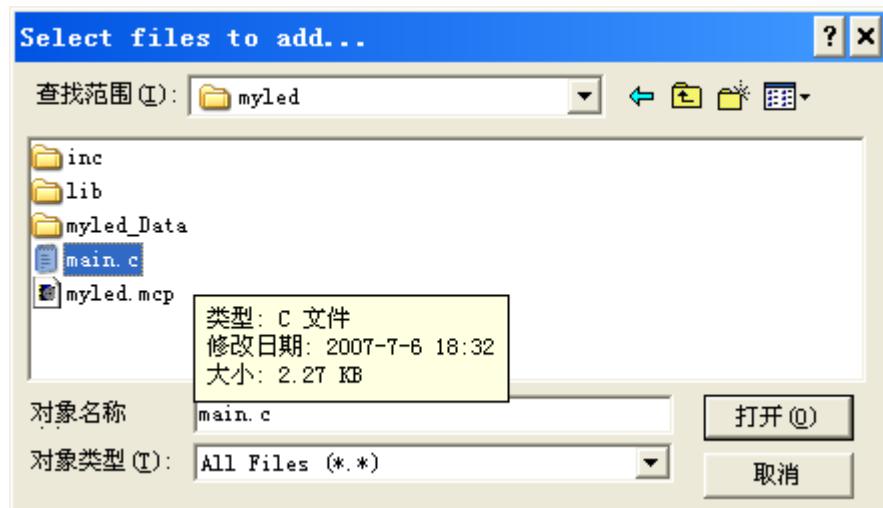
**DebugRel:** 使用该目标选项，在生成目标的时候，会为每一个源文件生成调试信息。

**Debug:** 使用该目标选项，在生成目标的时候，会为每一个源代码生成最完整的调试信息。

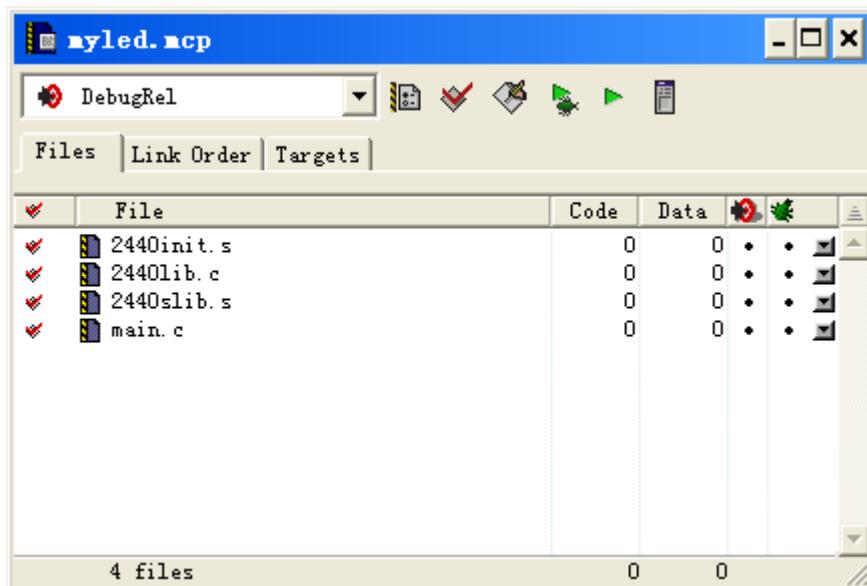
**Release:** 使用该目标选项，在生成目标的时候，不会生成任何调试信息。

在本例中，我们使用默认的 DebugRel 选项。

然后把 main.c 也加入到 myled.mcp 项目工程：

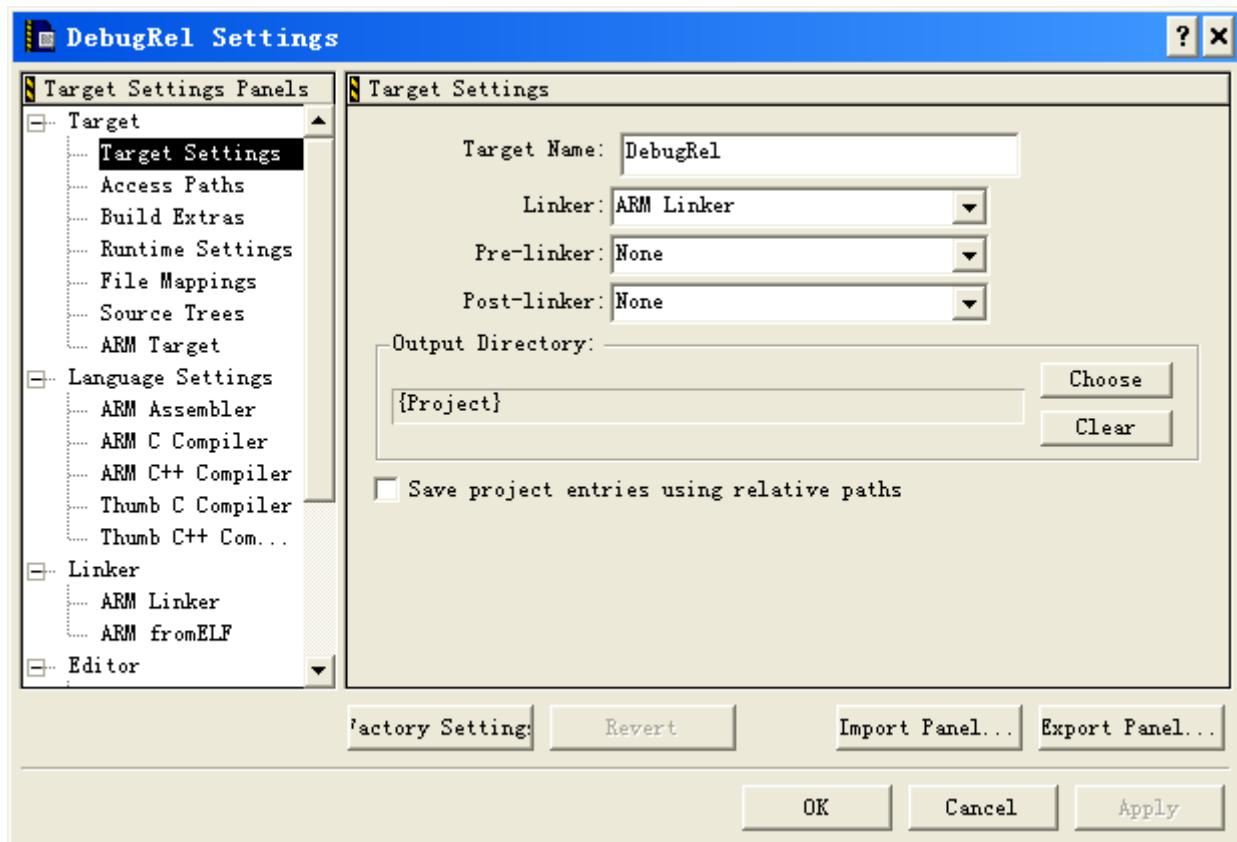


到目前为止，一个完整的工程就已经建立了，下面我们开始对该工程进行编译和链接的配置。



#### 4.1.2 编译和链接工程

在进行编译和链接之前,首先需要对生成的目标进行配置,点 Edit 菜单,选择“DebugRel Setting...”(注意:这个选项会因为用户选择的不同目标而有所不同),出现如图所示的设置窗口。



这里的设置有很多，我们主要介绍最常用的一些选项。

### ● Target Setting

Target Name 文本框显示了当前的目标设置。

Linker 选项为用户提供了要使用的连接器，在这里选择默认的 ARM Linker，使用该连接器，将使用 armlink 链接编译器和汇编器生成相应的工程目标文件。

在 Linker 设置中，还有两个可选项，None 代表不对生成的各个源代码目标文件进行链接，ARM Librarian 表示将编译或者汇编得到的目标文件转换为 ARM 库文件，对于本例，使用默认的连接器 ARM Linker。

Pre-Linker：目前 ADS 并不支持该选项。

Post-Linker：选择在链接完成后，还要对输出文件进行的操作。因为在本例中，希望生成一个可以烧写到 Flash 中去的二进制代码，所以在此选择 ARM fromELF，表示在链接生成映象文件后，再调用 fromELF 命令将含有调试信息的 ELF 格式的映象文件转换为其他格式的文件。

Target Setting 选择最后设置如图所示：

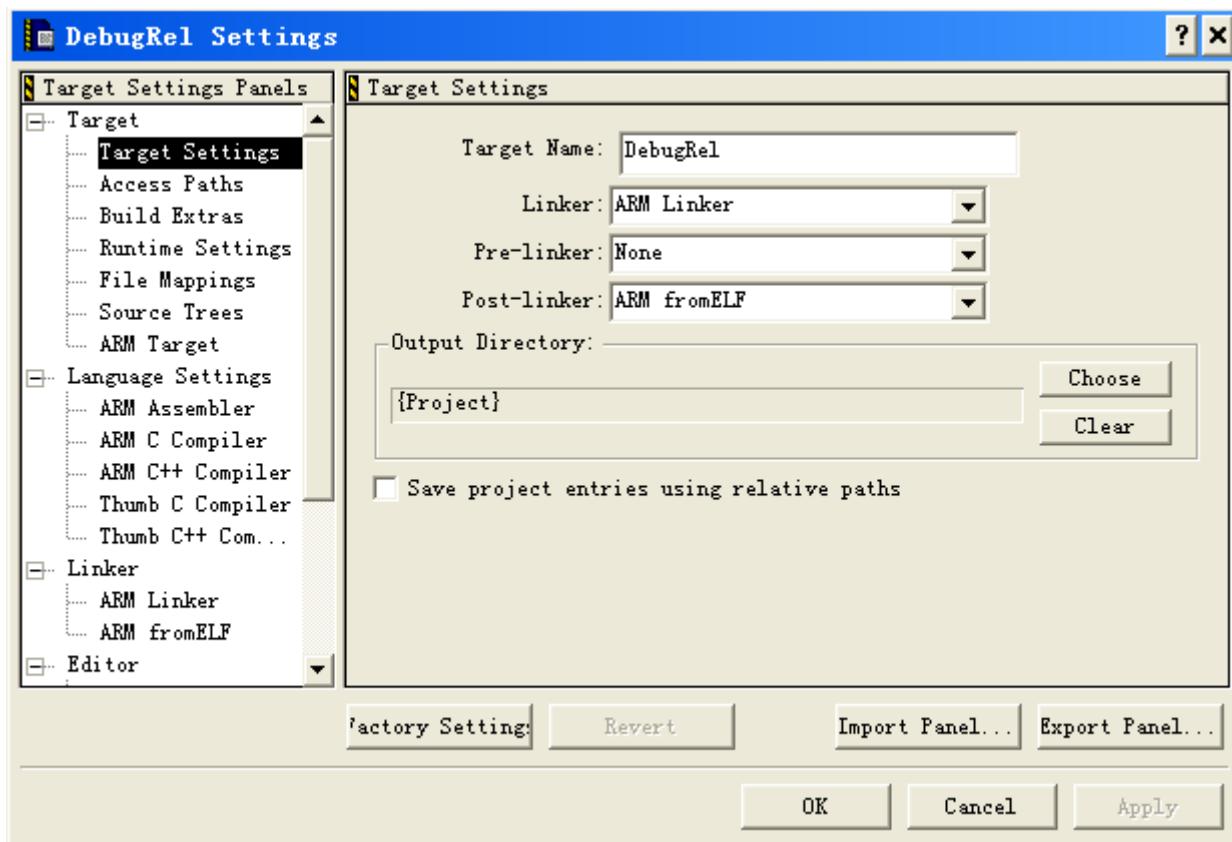


追求卓越 创造精品

TO BE BEST

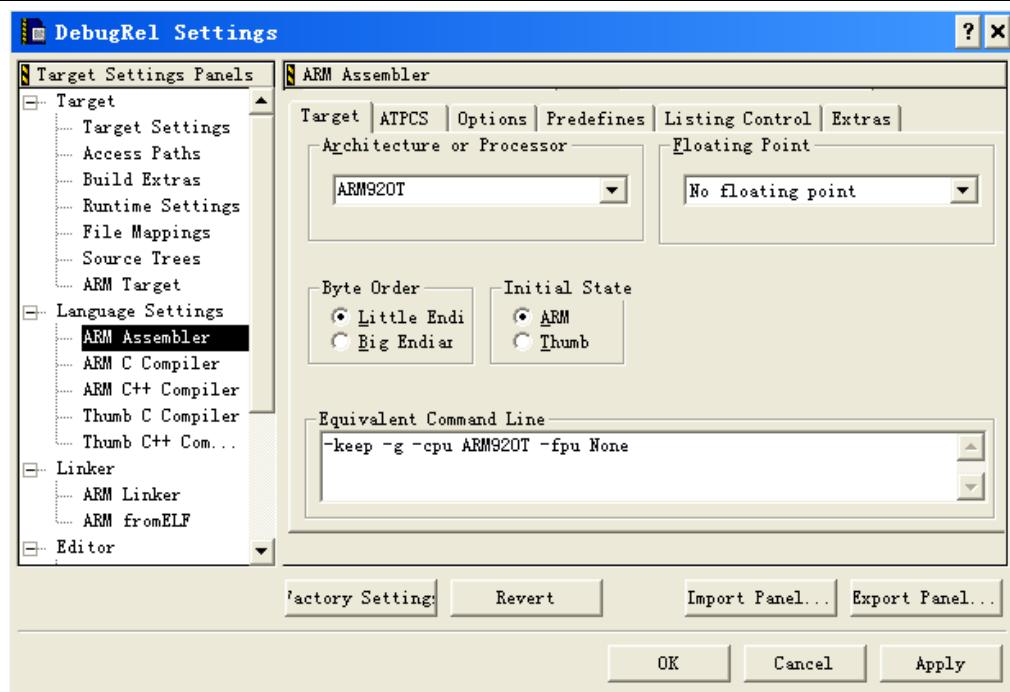
TO DO GREAT

广州友善之臂计算机科技有限公司

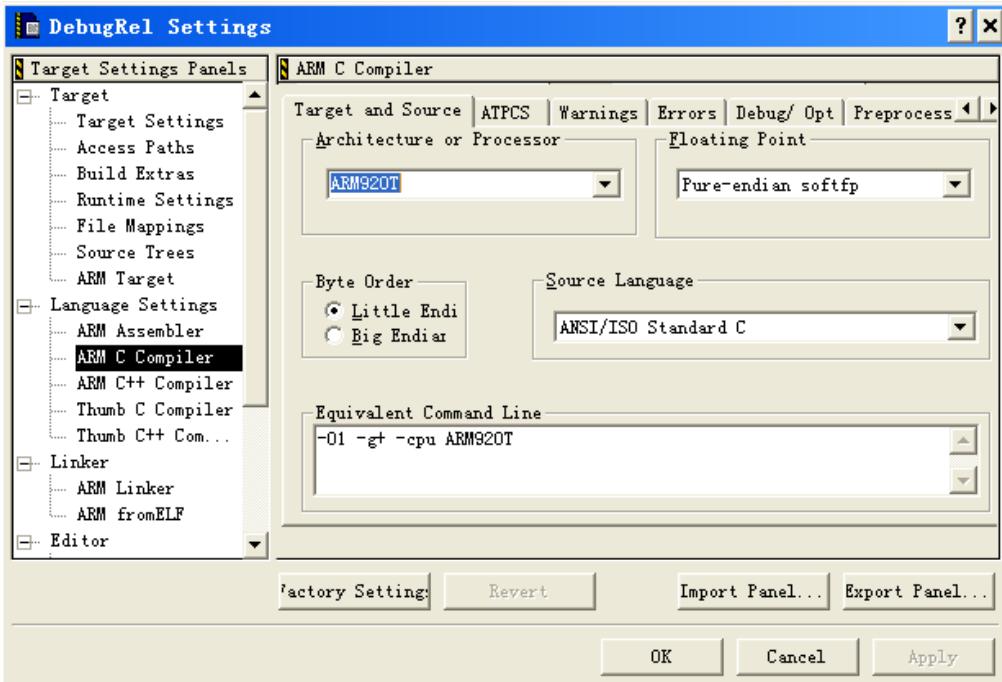


### ● Language Settings

因为在本例中包含汇编代码，所有要用到汇编器，直接选择 **ARM Assembler**，在右侧出现相应的设置选项，在 ADS 集成开发环境中用的汇编器是 armasm，默认的 ARM 体系结构是 ARM7TDMI，在此要改为 ARM920T，字节顺序默认是小端模式，其他设置，采用默认值即可，如图所示：



本例中还包含了 C 语言代码，因此还需要设置 ARM C Compiler 选项，点接选择 ARM C Compiler，在右侧出现相应的设置选项，在 ADS 集成开发环境中用的汇编器是 armcc，默认的 ARM 体系结构是 ARM7TDMI，在此要改为 **ARM920T**，字节顺序默认是小端模式，其他设置，采用默认值即可，如图所示：



细心的读者可能会注意到，在设置框的右下脚，当对某项设置进行了修改，该行中的某个选项就会发生相应的改动，实际上，这行文字显示的就是相应的编译或者链接选项，由



追求卓越 创造精品

TO BE BEST

TO DO GREAT

广州友善之臂计算机科技有限公司

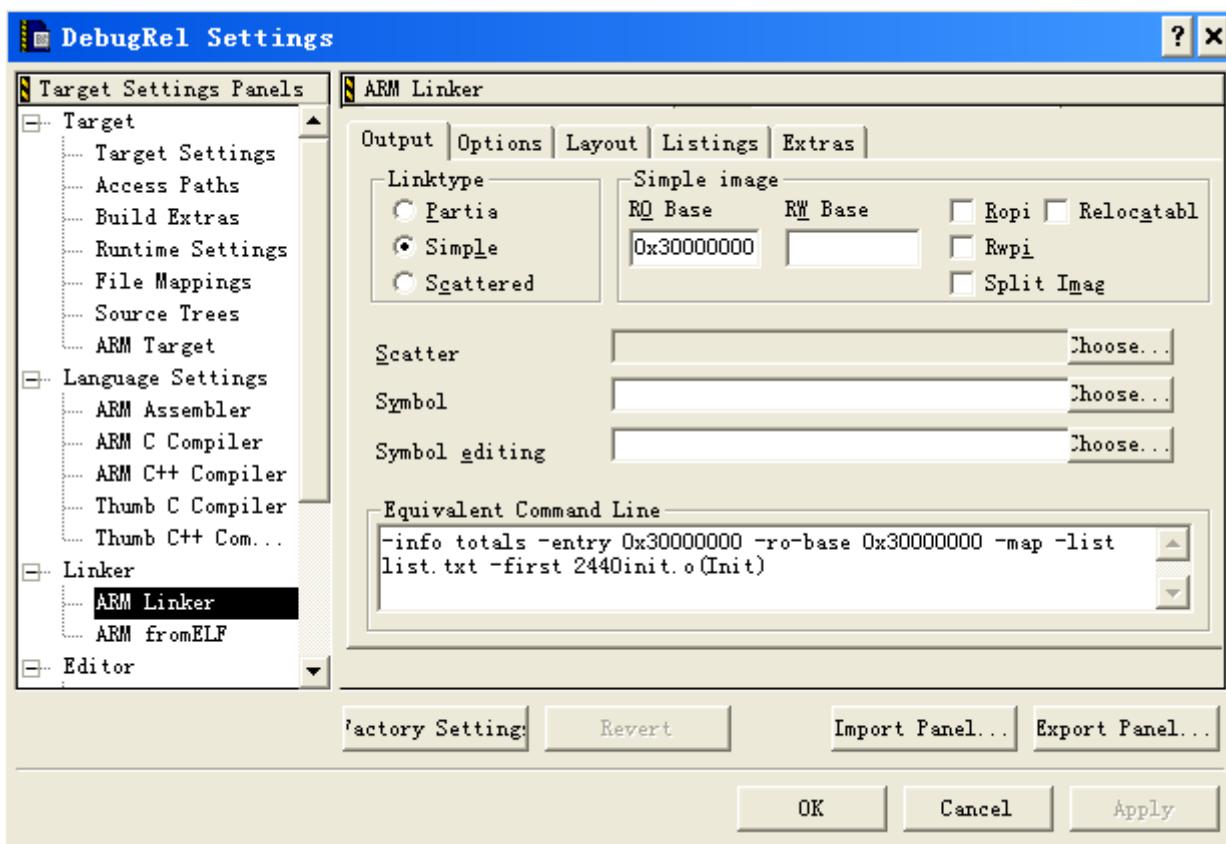
于有了 CodeWarrior，开发人员可以不用再去产科繁多的命令行选项，只要在界面中选择或者撤销某个选项，软件就会自动生成相应的代码，该命令框为习惯在 DOS 下键入命令行的用户提供了极大的方便。

### ● Linker 设置

**提示：如果您对 ADS 设置不熟悉，可以使用我们缺省的项目文件，以下部分仅供参考。**

点接选择 ARM Linker，在右侧出现相应的设置选项，我们在此详细介绍这些设置框，因为这些选项对最终生成的文件有着直接的影响。

在标签 Output 中，Linktype 中提供了三种链接方式。Partial 方式表示链接器只进行部分链接，经过部分链接生成的目标文件，可以作为以后进一步链接时的输入文件。Simple 方式是默认的链接方式，也是最为频繁使用的链接方式，它链接生成简单的 ELF 格式的目标文件，使用的是链接器中指定的地址映象方式。Scattered 方式使得链接器要根据 scatter 格式文件指定的地址映象，生成复杂的 ELF 格式的映象文件，这个选项一般很少用到。



在本例中，我们选择使用 Simple 方式，这里有一些设置：

**RO Base:** 这个文本框设置包含 RO 段的加载域和运行域为同一个地址，默认是 0x8000。这里用户要根据自己的硬件实际 SDRAM 地址空间来修改这个地址，保证这里填写的地址，是程序运行时，SDRAM 地址空间所能到达的范围，针对本目标板，SDRAM 的空间范围是 0x30000000-0x34000000，因此这里设置为 0x30000000。

**RW Base:** 这个文本框设置了包含 RW 和 ZI 输出段的运行域地址。如果选中 split 选项，



链接器生成的映象文件将包含两个加载域和两个运行域，此时在 RW Base 中所输入的地址为包含 RW 和 ZI 输出段的域设置了加载域和运行域地址。

**Ropi:** 选中这个设置将告诉链接器使包含有 RO 输出段的运行域位置无关。使用这个选项，链接器将保证下面的操作：检查各段时间的重寻址是否有效；确保任何由 armlink 自身生成的代码是只读位置无关的。

**Rwpi:** 选中该选项将会告诉链接器使包含 RW 和 ZI 输出段的运行域无关。如果这个选项没有被选中，域就标识为绝对。每一个可写的输入段必须是和读写位置无关的。如果这个选项被选中，链接器将进行下面的操作：

检查可读/写属性的运行域的输入段是否设置了位置无关属性；

检查在各段之间的重地址是否有效；

在 Region\$\$Table 和 ZISection\$\$Table 中添加基于静态存储器 sb 的选项。

该选项要求 RW Base 有值，如果没有给他指定数值的话，默认为 0。

**Split Image:** 选择这个选项把包含 RO 和 RW 的输出段的加载域分成 2 各加载域，一个包含 RO 输出段的域，一个是包含 RW 输出段的域。

这个选项要求 RW Base 有值，如果没有给 RW Base 选项的值，则默认是 0。

**Relocatable:** 选择这个选项保留了映像文件的重寻址偏移量。这些偏移量为程序加载器提供了有用信息。在 Options 选项中，需要读者引起注意的是 Image entry point 文本框。它指定映像文件的初始入口点地址值，当映像文件被加载程序加载时，加载程序会跳转到该地址处执行。如果需要，用户可以在这个文本框中输入下面格式的入口点：

入口点地址：这是一个数值，例如 -entry 0x0

符号：该选项指定映像文件的入口点为该符号所代表的地址处，比如：-entry int\_handler，如果该符号有多处定义存在，armlink 将产生出错信息。

**offset+object(section):** 该选项指定在某个目标文件的段的内部的某个偏移量处为映像文件的入口地址，例如：-entry8+startup(startuoseg)在此处指定的入口点用于设置 ELF 映像文件的入口地址。需要引起注意的是，这里不可以用符号 main 作为入口点地址符号，否则将会出现类似“Image dose not have an entry point(Not specified or not set due to multiple choice)”的错误信息。在 Layout 选项中，需要设置 asm.o 目标文件中的 Init 为整个文件的入口点。关于 ARM Linker 的设置还很多，对于想进一步深入了解的读者，可以查看帮助文件，都有很详细的介绍。

在 Linker 下还有一个 ARM fromELF：

fromELF 是一个实用工具，它实现将链接器，编译器和汇编器的输出代码进行格式转换的功能。例如，将 ELF 格式的可执行映像文件转换成可以烧写到 ROM 的二进制格式文件；对输出文件进行反汇编，从而提取出有关目标文件的大小，符号和字符串表以及重寻址等信息。只有在 Target 设置中选择了 Post-linker，才金可以使用该选项。

在 Output format 下拉框中，为用户提供了多种可以转换的目标格式，本例选择 Plain binary，这是一个二进制格式的可执行文件，可以被烧写到目标板的 Flash 中。

在 Output file name 文本域输入期望生成的输出文件存放的路径，或通过点击 Choose... 按钮从文件对话框中选择输出文件。如果在这个文本域不输入路径名，则生成的二进制文件存放在工程所在的目录下。进行好这些相关的设置后，以后在对工程进行 make 的时候，CodeWarrior IDE 就会在链接完成后调用 fromELF 来处理生成的映像文件。



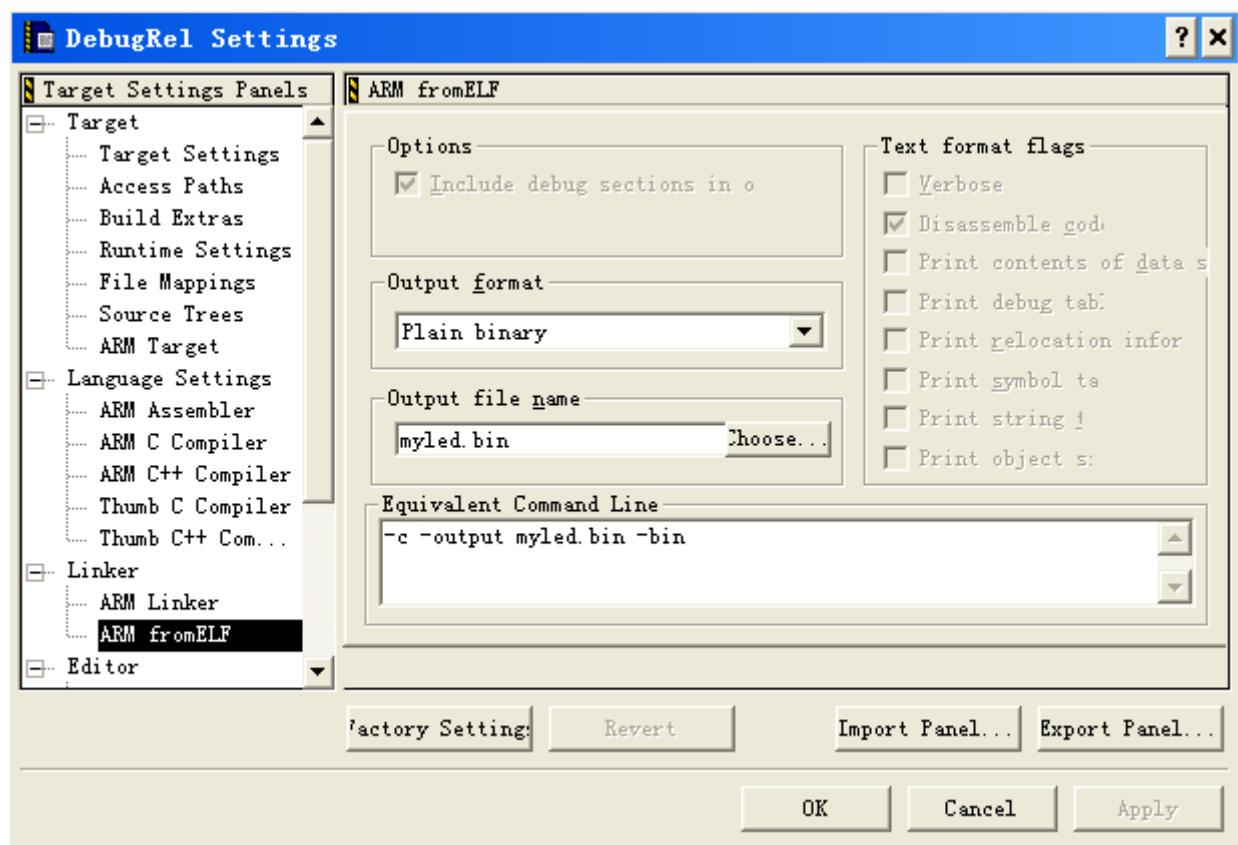
追求卓越 创造精品

TO BE BEST

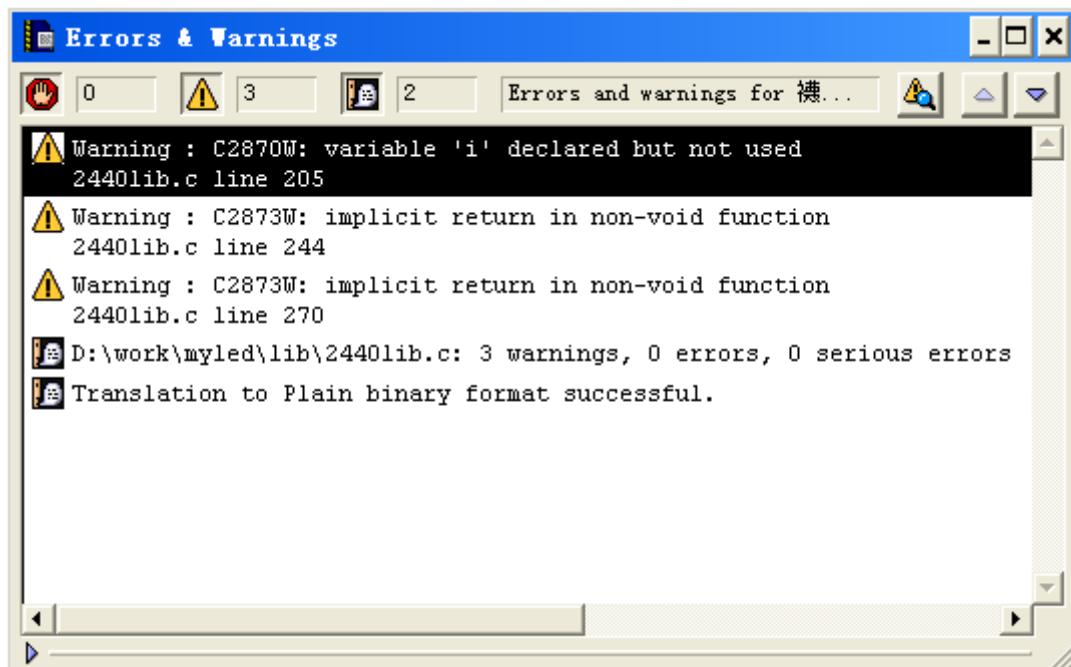
TO DO GREAT

广州友善之臂计算机科技有限公司

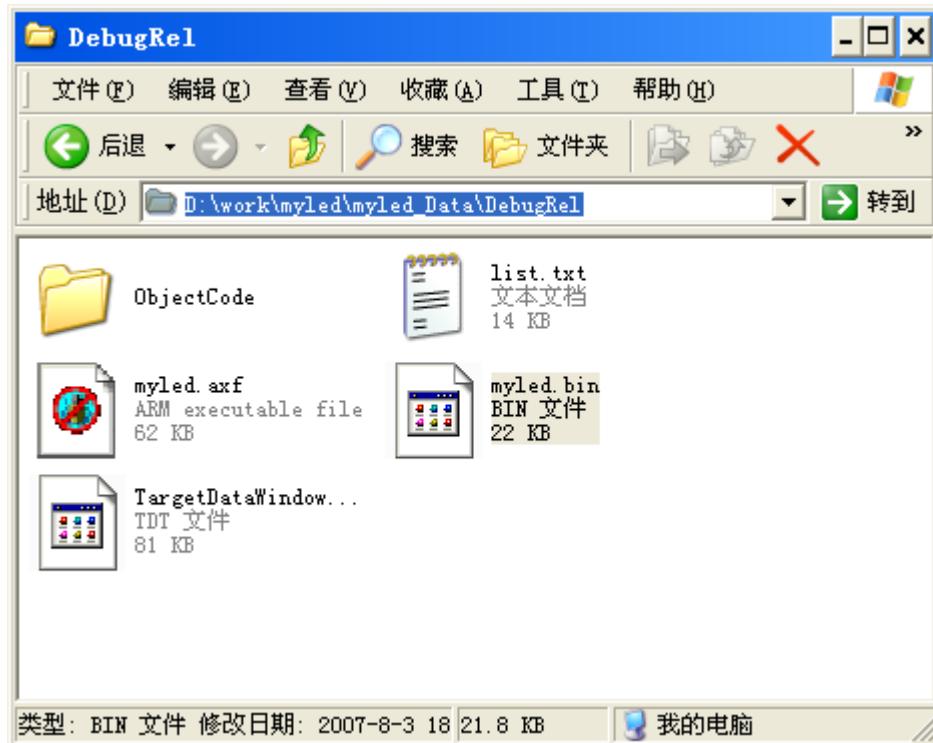
对于本例的工程而言，到此，就完成了 make 之前的设置工作了。



点击 CodeWarrior IDE 的菜单 Project 下的 make 菜单，就可以对工程进行编译和链接了。编译链接结果如图：



最后在“D:\work\myled\myled\_Data\DebugRel”目录下生成 myled.bin, 同时还有 myled.axf 文件, 它是用于调试的, 如图:



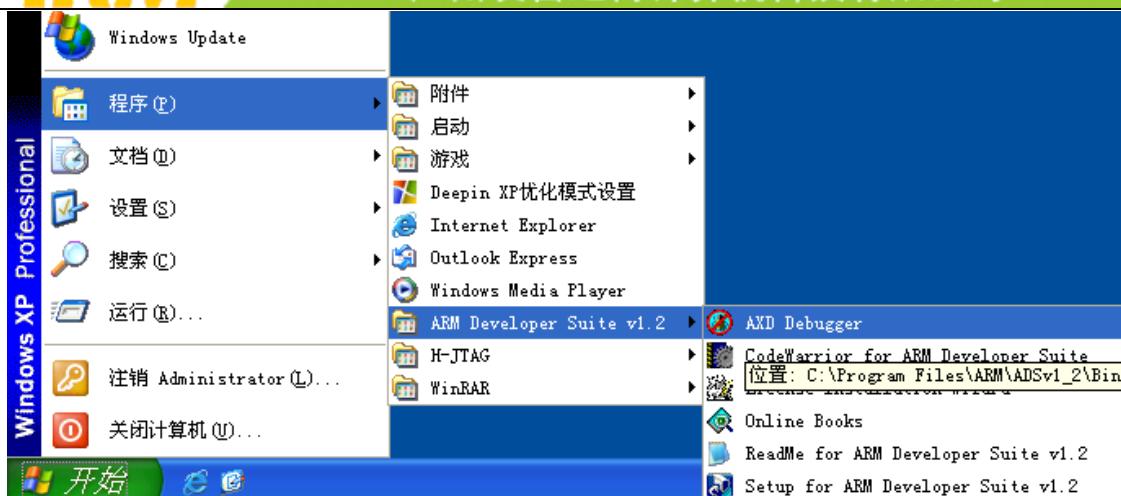
## 4.2 使用 H-JTAG 进行代码调试

请先按照 2.6 一节安装 H-JTAG。

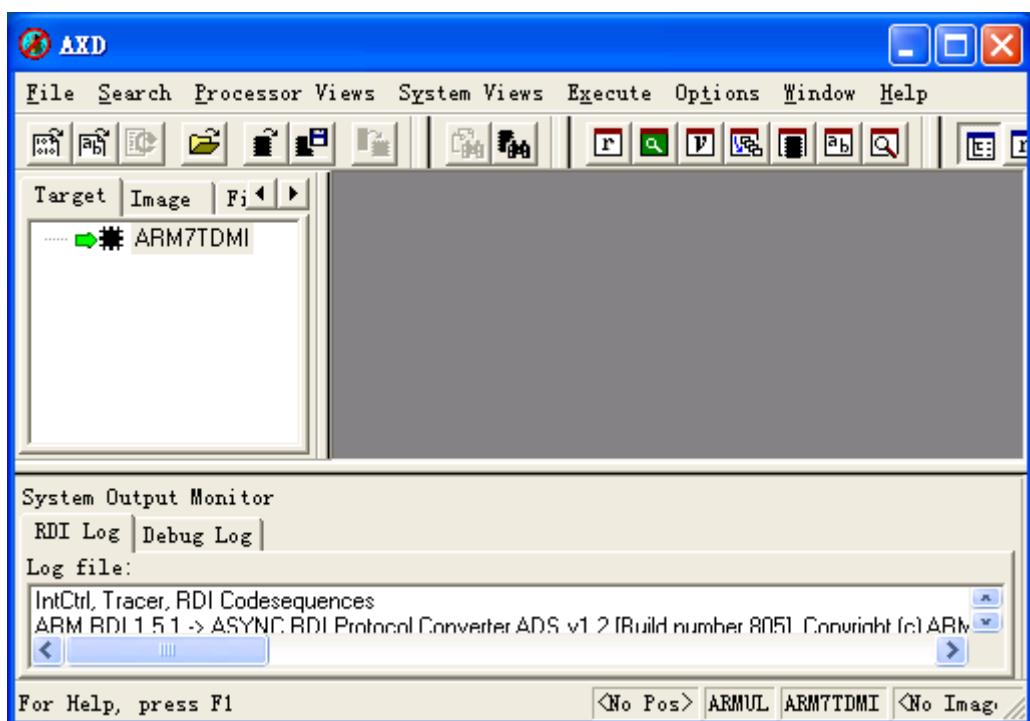
### 4.2.1 为 H-JTAG 配置 AXD DEBUGGER

#### (1)运行 AXD Debugger

如图所示打开运行 ADS1.2 软件的调试软件—AXD Debugger:



AXD Debugger 主界面:



## (2)设置 AXD Debugger

点菜单 Options->Confiuguer Target...， 出现如下窗口：

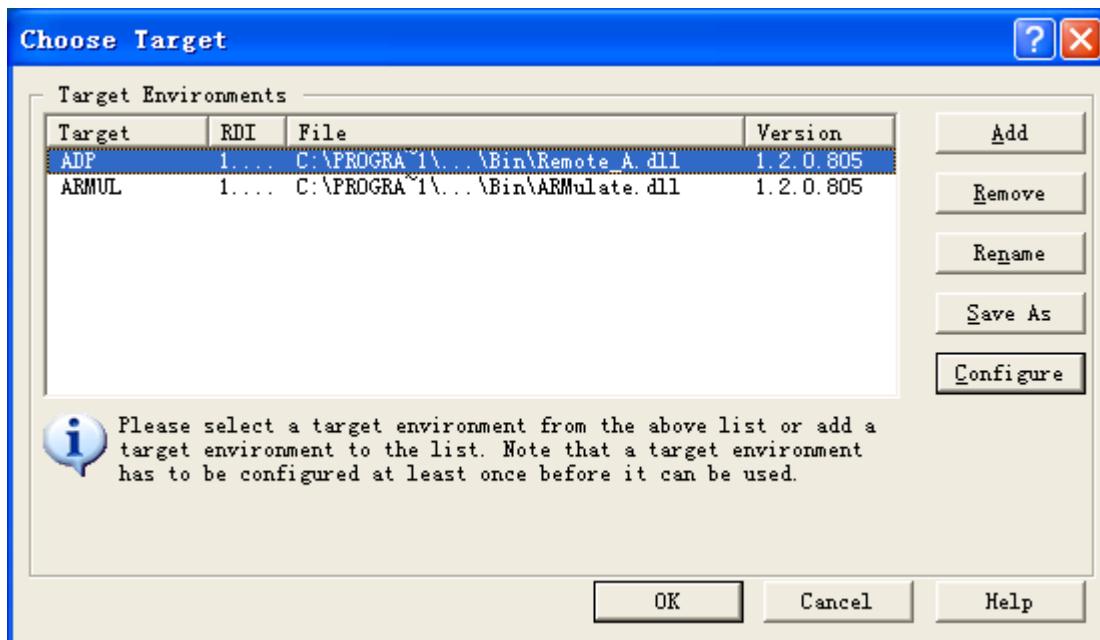


追求卓越 创造精品

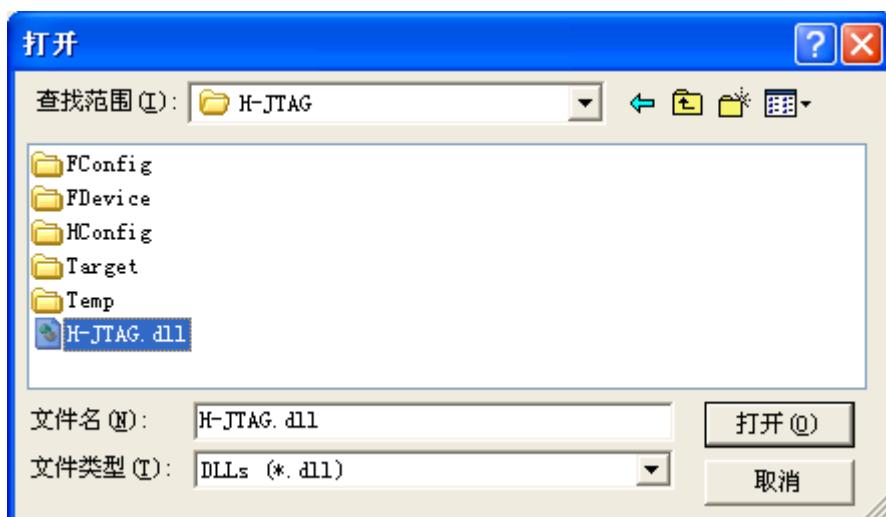
TO BE BEST

TO DO GREAT

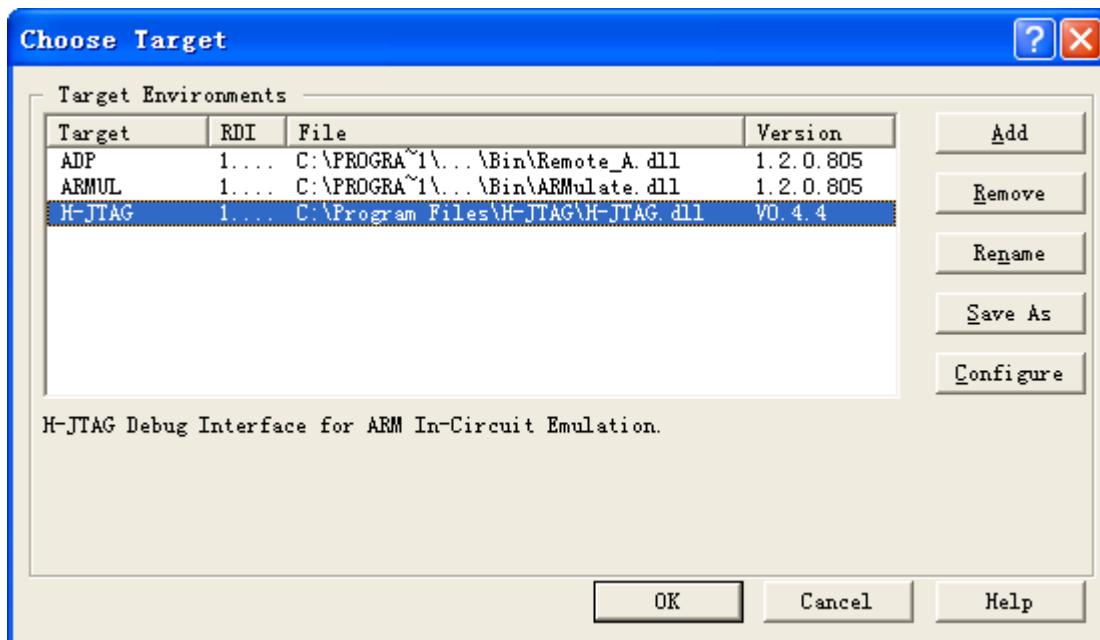
广州友善之臂计算机科技有限公司



在该窗口中点击 Add 按钮，跳出选择文件对话框，找到 H-JTAG 安装目录，选择并打开里面的 H-JTAG.dll 文件，如图。



此时会在 Choose Target 窗口中多了一项 H-JTAG，如图，点 OK 返回 AXD Debugger 主界面。



#### 4.2.4 使用 H-JTAG 在 ADS1.2 环境下进行仿真调试

关闭并重新启动 AXD Debugger，点击菜单 File->Load Image，找到您想调试的调试目标文件(\*.axf 格式)，如我们前面编译生成的 myled.axf 文件，打开它就自动启动目标映象通过 Jtag 下载至目标板，这时在 AXD Debugger 底部的状态栏会出现下载过程提示，下载完毕就可以进行单步或者全速调试了，调试过程中您可以看到 CPU 各个寄存器的值，也可以设置断点等，详细的用法请参考 ADS 附带的英文说明手册，这基本上和常见的 VisualC++ 等集成开发环境是类似的。

### 4.3 编译运行烧写 2440test

2440test 是源自三星的一个非操作系统测试程序，里面集成了很多小型的测试程序，涉及到 GPIO 的配置，中断的编写，常见接口的测试使用等，其中每一部分的测试代码都有很强的独立性，非常适合“ARM 基础性”练习实验。2440test 是基于 ADS1.2 开发环境创建的，它编译出的二进制文件不能下载到 linux 或者 wince 系统中运行，只能下载到内存指定地址（这里是 0x30000000）运行，也可以烧写到 Nand Flash 中运行。

本开发板提供的 2440test 对于开机后 LCD 显示的支持，可以通过\2440test\inc\Option.h 中的 LCD\_TYPE 定义来选择：

```
#define LCD_TYPE_N35      1 /* N35 代表适用于 NEC3.5 寸屏 */
#define LCD_TYPE_A70      2 /* A70 代表适用于 7 寸屏 */
#define LCD_TYPE_VGA1024x768 3 /* VGA1024x768 代表适用于 VGA 输出，分辨率为 1024x768 */
```

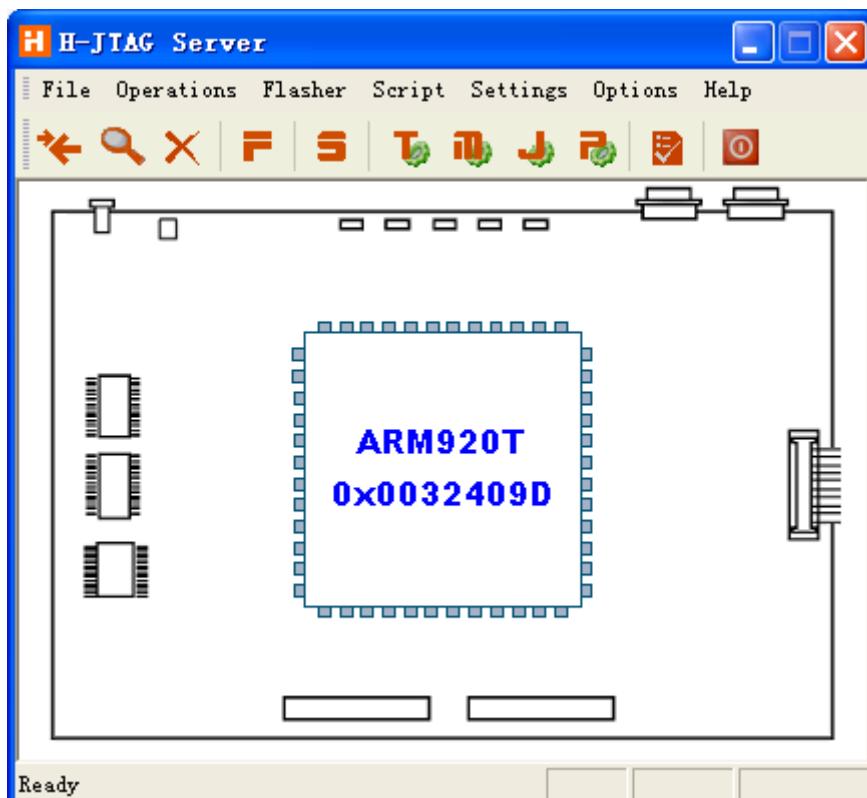
```
#define LCD_TYPE LCD_TYPE_N35
```

光盘中默认的是 LCD\_TYPE\_N35

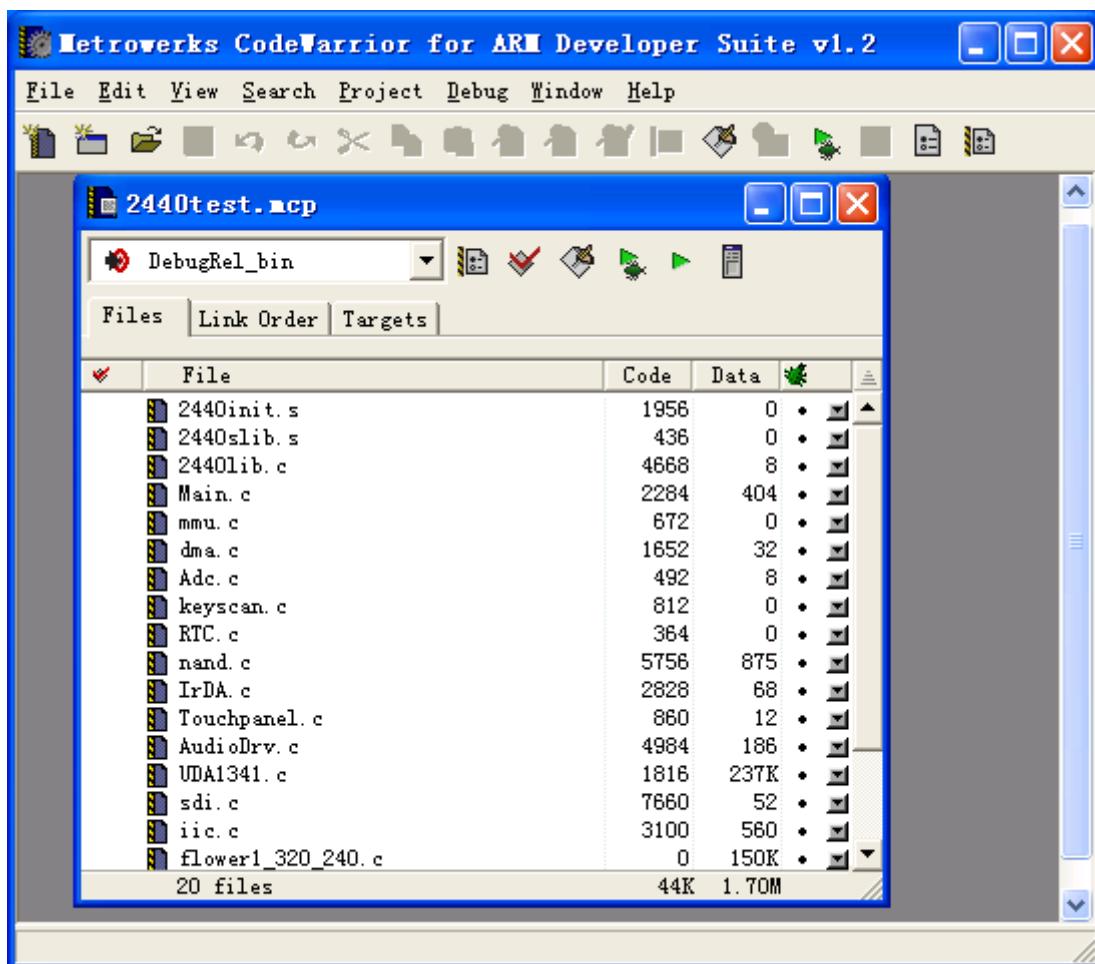
下面是 2440test 的编译、运行和烧写的方法，代码的原理性请用户自行理解。

#### 4.3.1 编译和使用 H-JTAG 调试 2440test

使用开发板附带的 JTAG 小板连接开发板的 JTAG 接口，使用附带的串口线连接开发板的串口，并接上打开电源。这时打开 H-JTAG 软件，它会自动检测到目标板，如图。



把光盘中“非操作系统示例代码”文件夹中的 2440test 目录复制到硬盘的某一个目录（在此为 D:\work），去掉其只读属性，运行 ADS1.2 集成开发环境，点 File->Open... 打开 2440test.mcp 文件，如图。



点 Project->Debug 或者按 F5 键开始编译 2440test 项目，编译完毕会自动启动 AXD Debugger 调试器，并把编译好的 2440test.axf 映象通过 JTAG 下载到内存中，如图。

注意：因为 2440test.axf 比较大，因此下载需要等待一段时间。

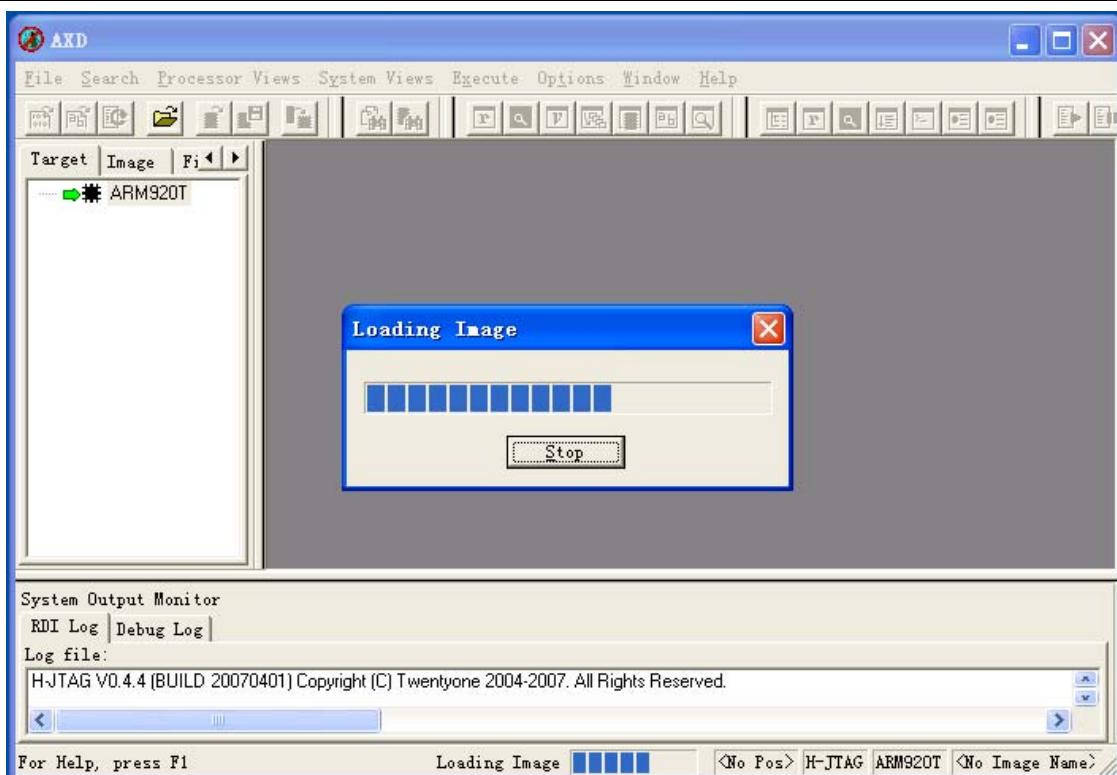


追求卓越 创造精品

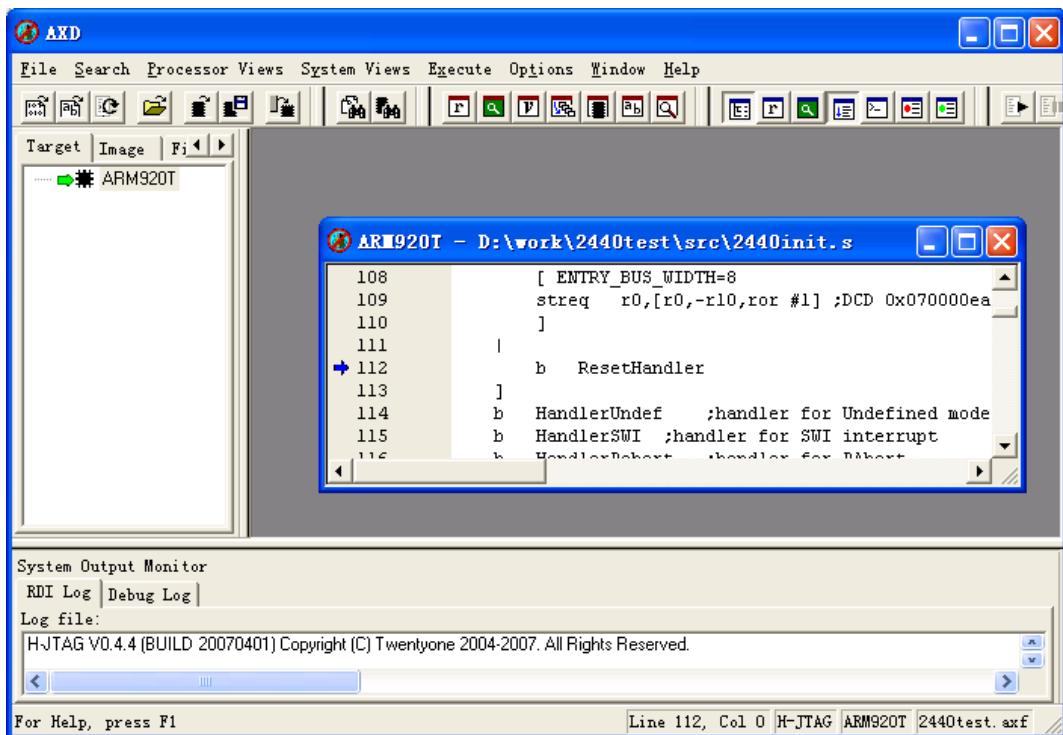
TO BE BEST

TO DO GREAT

广州友善之臂计算机科技有限公司



下载完毕如图所示。



这时点菜单 Execute->Go 或者按 F5 按键，程序将跳转到 Main 函数处，我们就可以进行单步运行调试了。

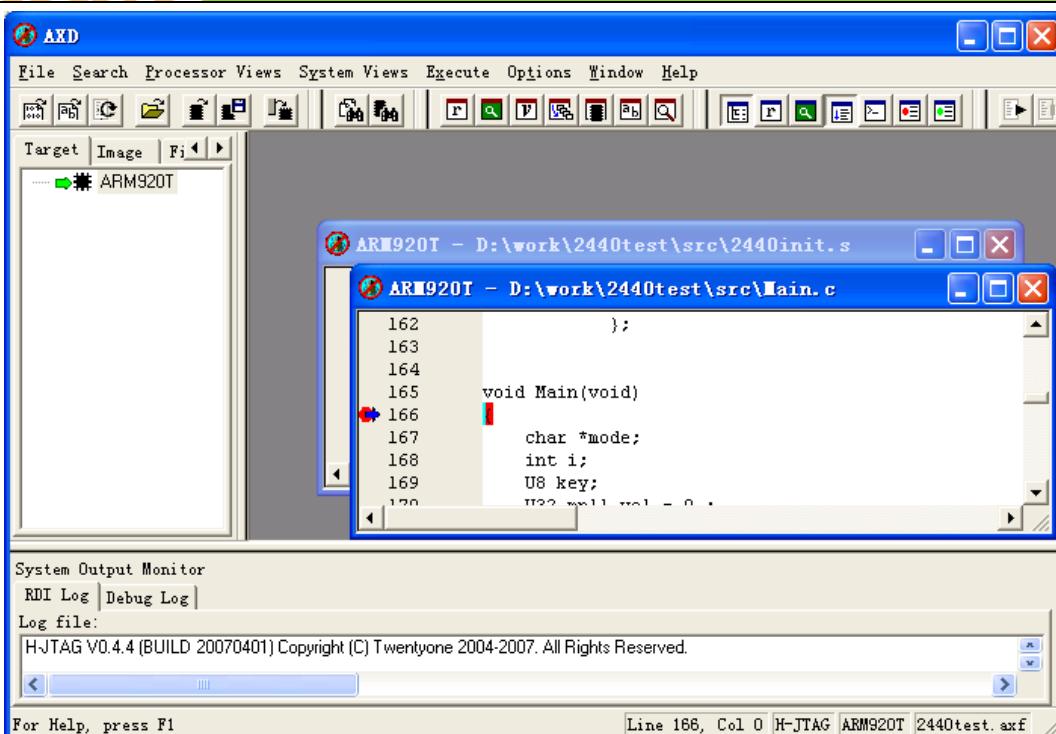


追求卓越 创造精品

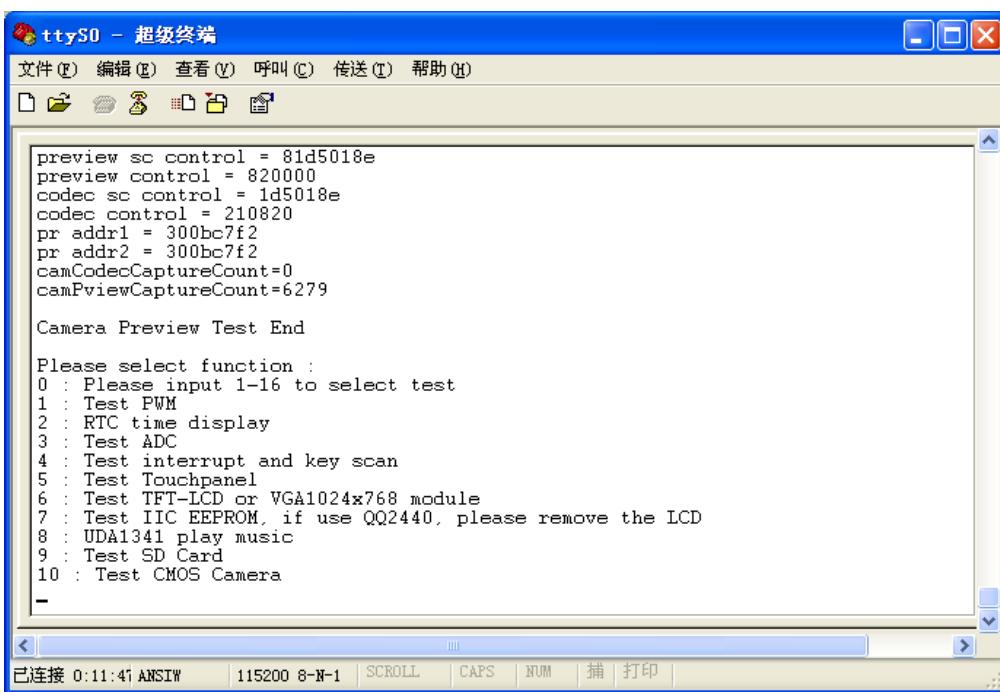
TO BE BEST

TO DO GREAT

广州友善之臂计算机科技有限公司



如果您这时再全速执行程序，可以看到从串口终端打印处如下信息，这和我们之前使用的非操作系统测试程序是一样的。



### 4.3.2 通过 USB 把 2440test 下载到运行

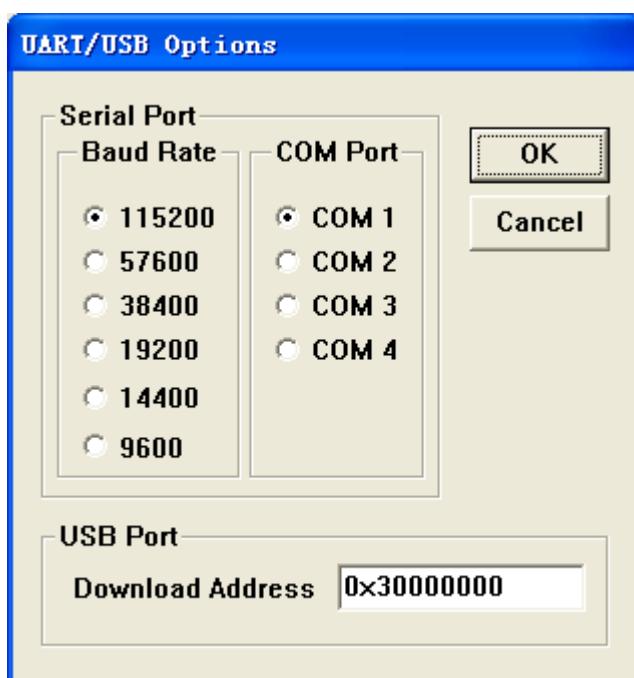
使用 USB 下载运行 2440test 程序不需要并口和 JTAG 板, 借助 Supervivi 的“Download & Run”功能就可以了, 下面是详细的操作步骤:

(1)连接好开发板电源, 串口线, USB 线, 并**设置开发板为 NOR Flash 启动系统**, 分别打开串口超级终端和 DNW, 上电启动开发板。

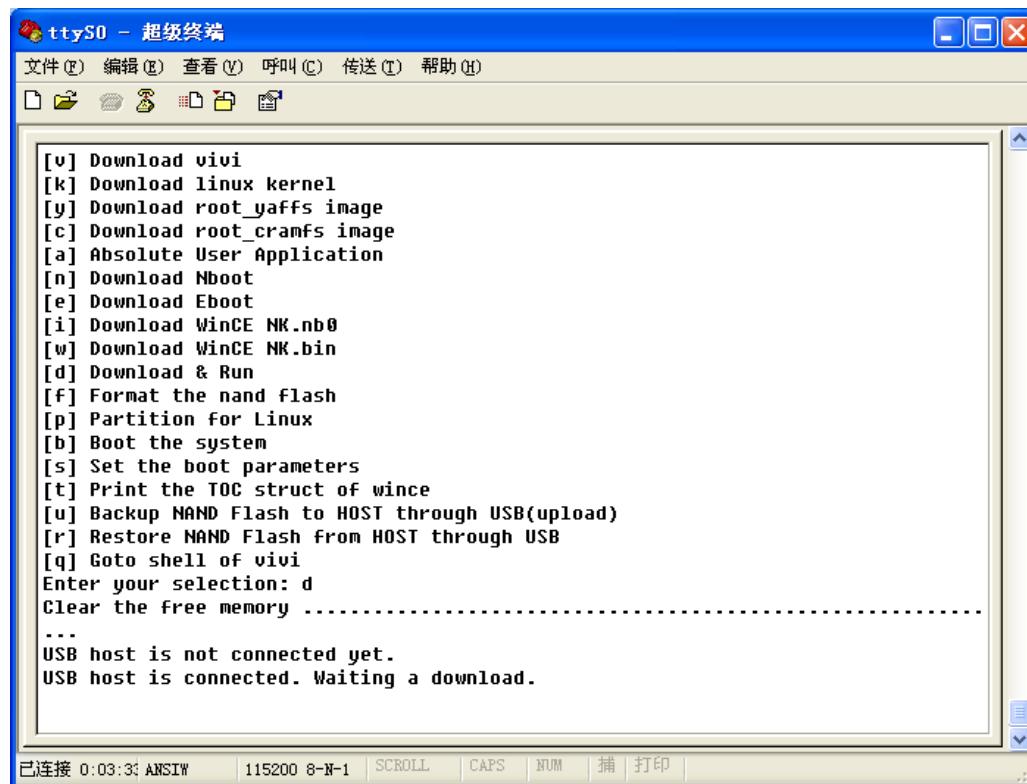
(2)保证 USB 驱动已经安装好(前面已经详细介绍了 USB 驱动的安装方法), 这时可以看到 DNW 的标题栏显示[USB: OK], 如果没有安装好驱动会显示[USB: x], 如图所示:



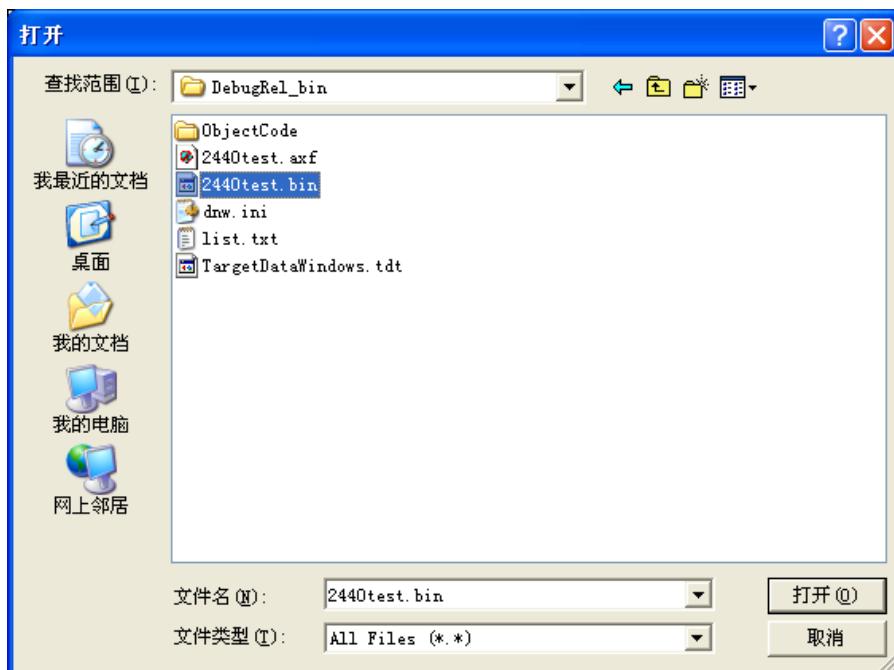
(3)点 DNW 菜单 Configuration, 设置 USB 下载运行地址为 0x30000000



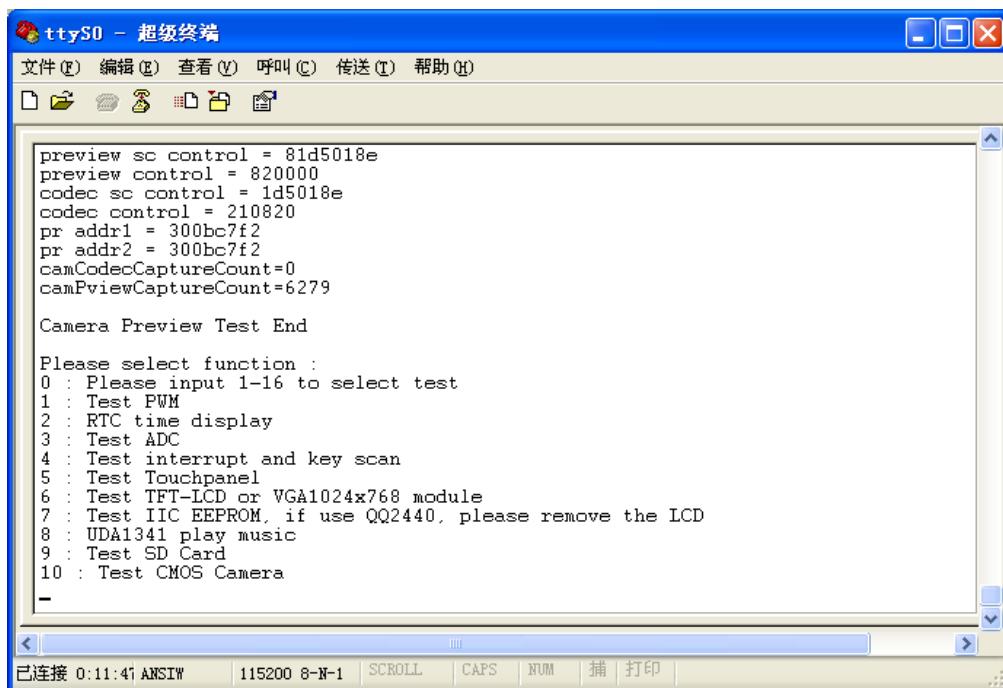
(4)这时在超级终端的 BIOS 功能菜单中选择功能号[d], 出现 USB 下载等待提示信息:



(5)点击 DNW 程序的“USB Port”→“Transmit”，如图选择刚刚编译出的映象文件(光盘“images”目录中有已经编译好的可执行文件)，这样就开始下载了



(6) 下载结束后，会自动运行，串口出现如下信息：



若使用 7 寸真彩屏，会出现如下界面：



若使用 NEC3.5 寸屏，会出现如下界面：



若使用 VGA 输出模块，会在显示器上出现如下界面：



#### 4.4.3 把 2440test 烧写到 Nand Flash 运行

使用 Supervivi 的功能号[a]可以把 2440test.bin 可执行程序烧写到 Nand Flash 中运行，步骤如下：

- (1)连接好开发板电源，串口线，USB 线，并**设置开发板为 NOR Flash 启动系统**，分别打开串口超级终端和 DNW，上电启动开发板。
- (2)保证 USB 驱动已经安装好(前面已经详细介绍了 USB 驱动的安装方法)，这时可以看到 DNW 的标题栏显示[USB: OK]，如果没有安装好驱动会显示[USB: x]，如图所示：



追求卓越 创造精品

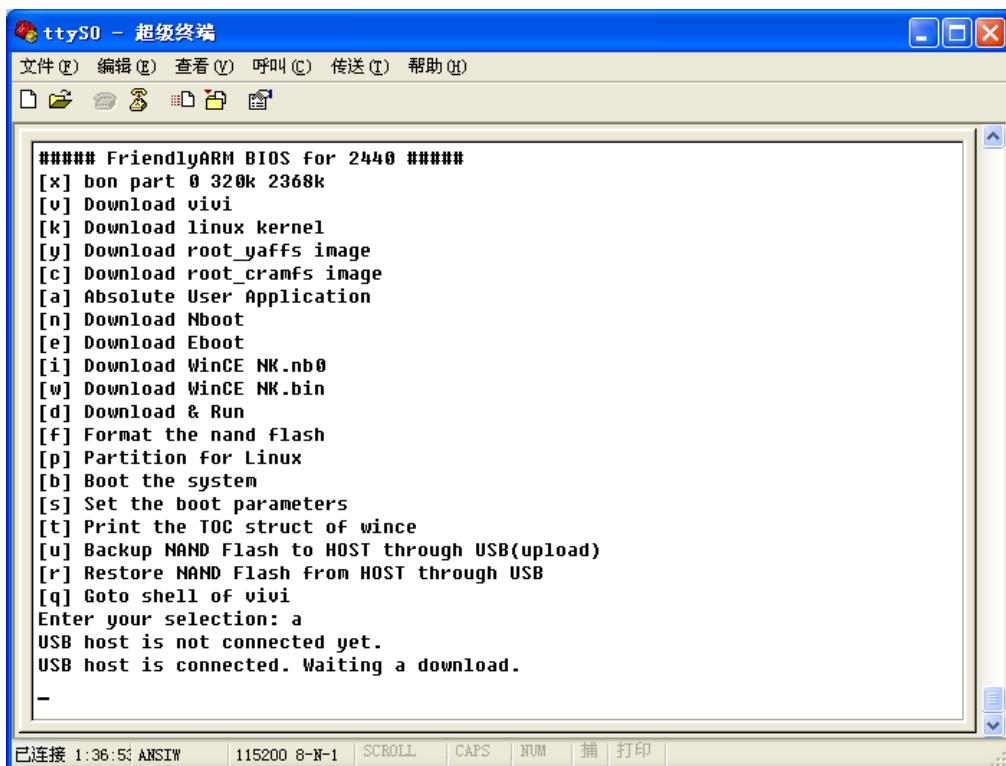
TO BE BEST

TO DO GREAT

广州友善之臂计算机科技有限公司



(3)这时在超级终端的 BIOS 功能菜单中选择功能号[a], 出现 USB 下载等待提示信息:





追求卓越 创造精品

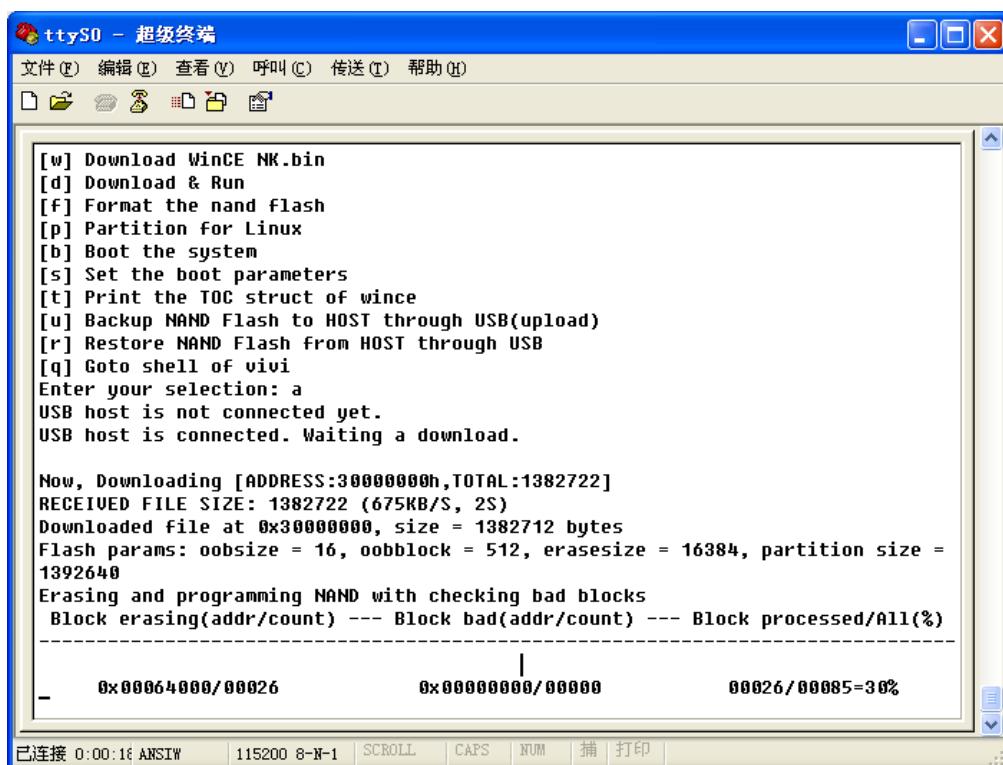
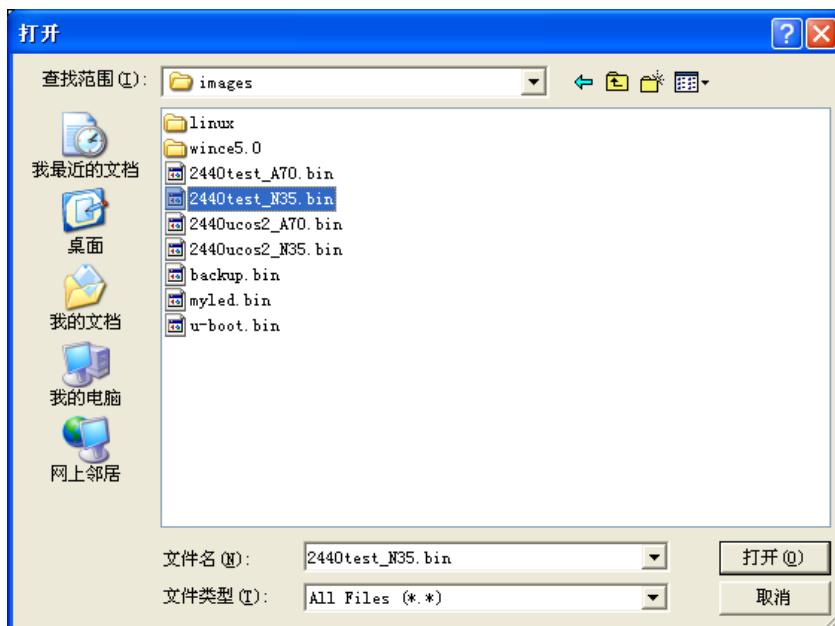
TO BE BEST

TO DO GREAT

广州友善之臂计算机科技有限公司

(4)点击 DNW 程序的“USB Port”→“Transmit”，如图选择刚刚编译出的映象文件(光盘“images”目录中有已经编译好的可执行文件)，这样就开始下载了，下载完毕，Supervivi 会把它自动烧写到 Nand Flash 起始块为 0 的地方，即 Block 0 开始处。

烧写完毕，把开发板设置为 Nand Flash 启动，重新开机或者复位开发板就可以了。





追求卓越 创造精品

TO BE BEST

TO DO GREAT

广州友善之臂计算机科技有限公司

## 4.5 uCos2 的编译和烧写

**说明：**本手册不详细讲述 uCos2 在 2440 上的移植过程及原理，这方面的资料在网上和书店里有很多，虽然都是针对三星 2410 系统的，但同样适用于 2440。

uCos2 的源代码位于光盘的“uCos2”目录，为了方便您的使用，我们分别以目录和压缩文件的方式提供其源代码：您可以把源代码目录复制到硬盘中，去掉其只读属性使用(以下示例就是如此)，也可以解压源代码包使用。

本开发板提供的 uCos2 对于开机后 LCD 显示的支持，可以通过 uCos2\S3C2440\includes\lcd.h 中的 LCD\_TYPE 定义来选择：

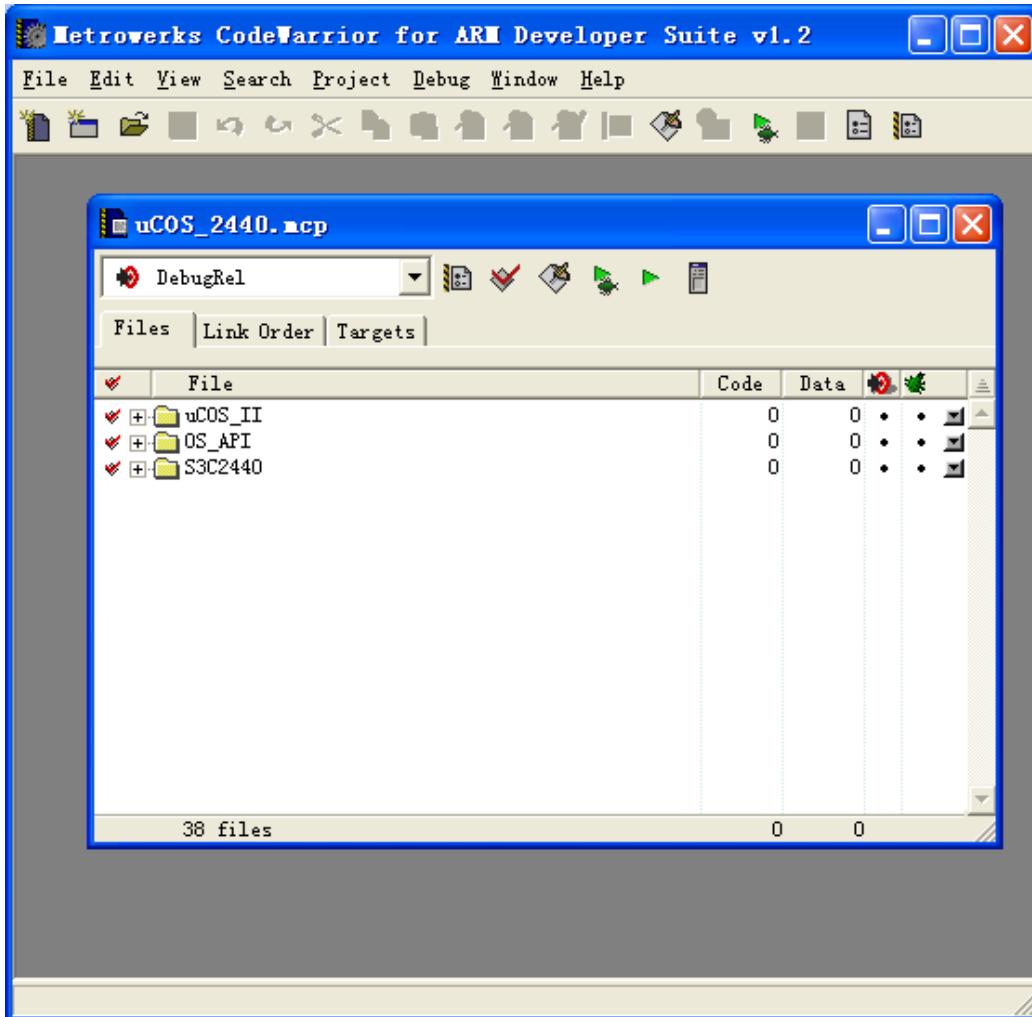
```
#define LCD_TYPE_N35      1 /* N35 代表适用于 NEC3.5 寸屏 */
#define LCD_TYPE_A70      2 /* A70 代表适用于 7 寸屏 */
#define LCD_TYPE_VGA1024x768 3 /*VGA1024x768 代表适用于 VGA 输出，分辨率为 1024x768 */
```

#define LCD\_TYPE LCD\_TYPE\_N35

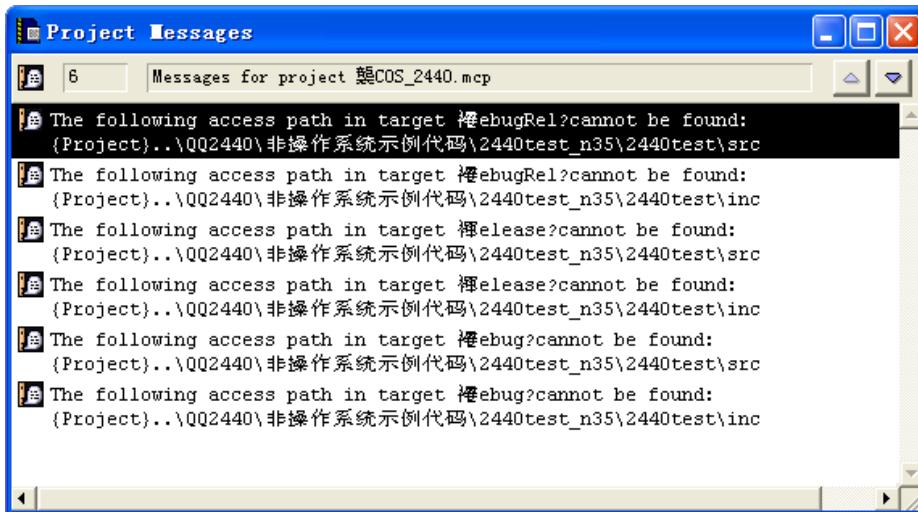
光盘中默认的是 LCD\_TYPE\_N35，

### 4.5.1 编译 uCos2

把光盘中“uCos2”目录夹中的“uCos2”文件夹复制到硬盘的某一个目录(在此为 D:\work)，去掉其只读属性，运行 ADS1.2 集成开发环境，点 **File->Open...** 打开 uCOS\_2440.mcp 文件，如图。

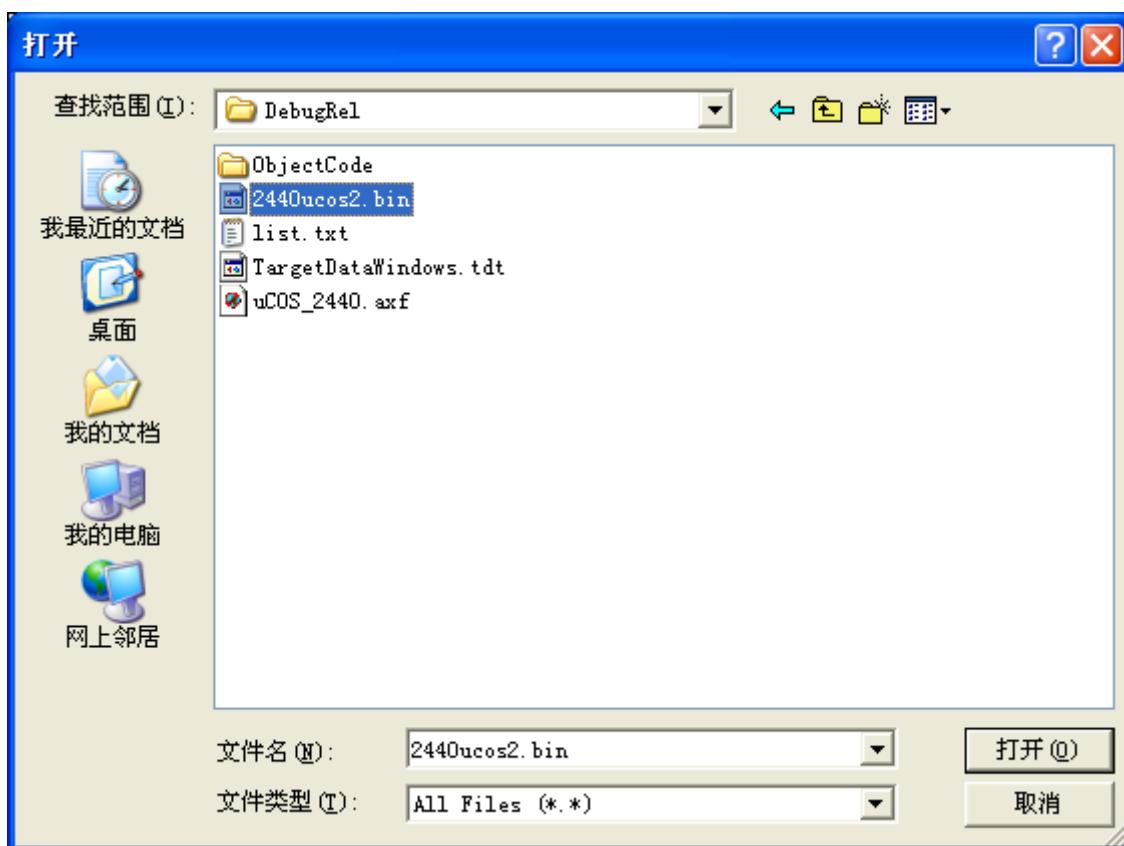


你可能会看到出现如下的一个提示窗口，可以不必理会：



这时点菜单 Project→Make 或者直接按 F7 键，开始编译 uCos2 项目。

编译完毕，在 D:\work\uCos2\uCOS\_2440\_Data\DebugRel 目录下会生成 2440ucos2.bin 可执行文件，如图。



#### 4.5.2 把 uCos2 下载到内存运行

(1)连接好开发板电源，串口线，USB 线，并设置开发板为 Nor Flash 启动系统，分别打开串口超级终端和 DNW，上电启动开发板。

(2)保证 USB 驱动已经安装好(前面已经详细介绍了 USB 驱动的安装方法)，这时可以看到 DNW 的标题栏显示[USB: OK]，如果没有安装好驱动会显示[USB: x]，如图所示：



追求卓越 创造精品

TO BE BEST

TO DO GREAT

广州友善之臂计算机科技有限公司



(3)点 DNW 菜单 Configuration, 设置 USB 下载运行地址为 0x30000000



(4)这时在超级终端的 BIOS 功能菜单中选择功能号[d], 出现 USB 下载等待提示信息:

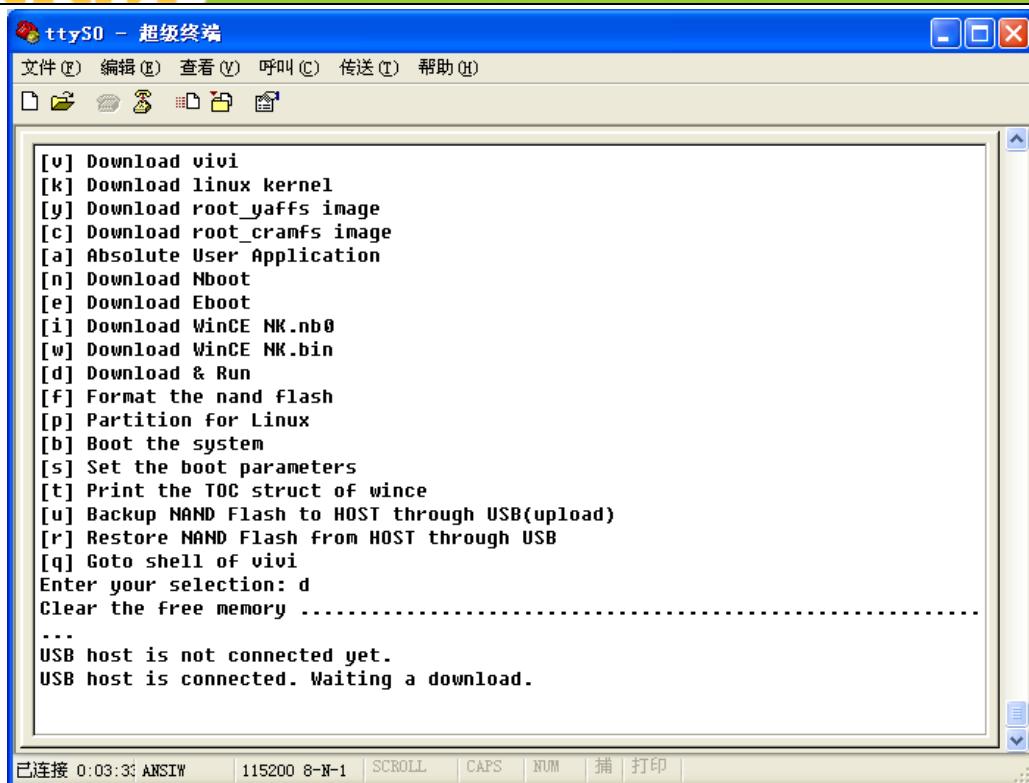


追求卓越 创造精品

TO BE BEST

TO DO GREAT

广州友善之臂计算机科技有限公司



(5)点击 DNW 程序的“USB Port”→“Transmit”，如图选择刚刚编译出的映象文件(光盘“images”目录中有已经编译好的可执行文件)，这样就开始下载了

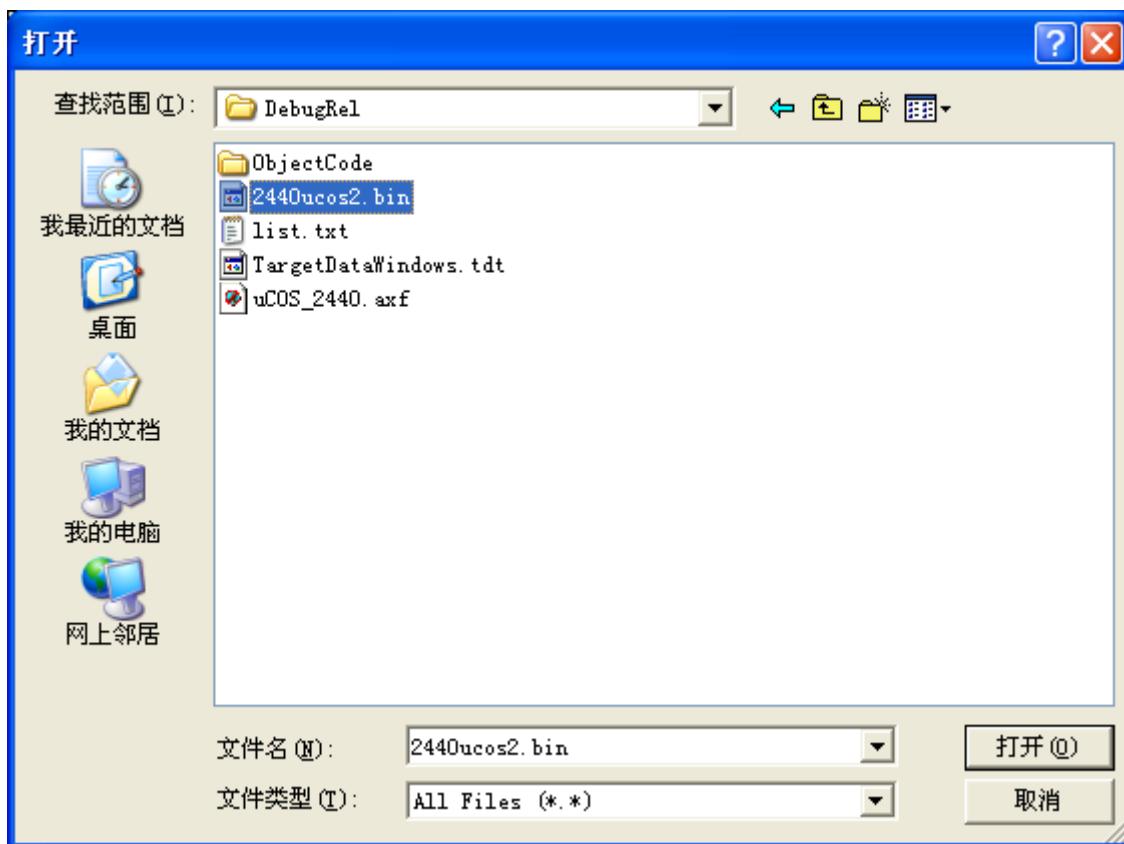


追求卓越 创造精品

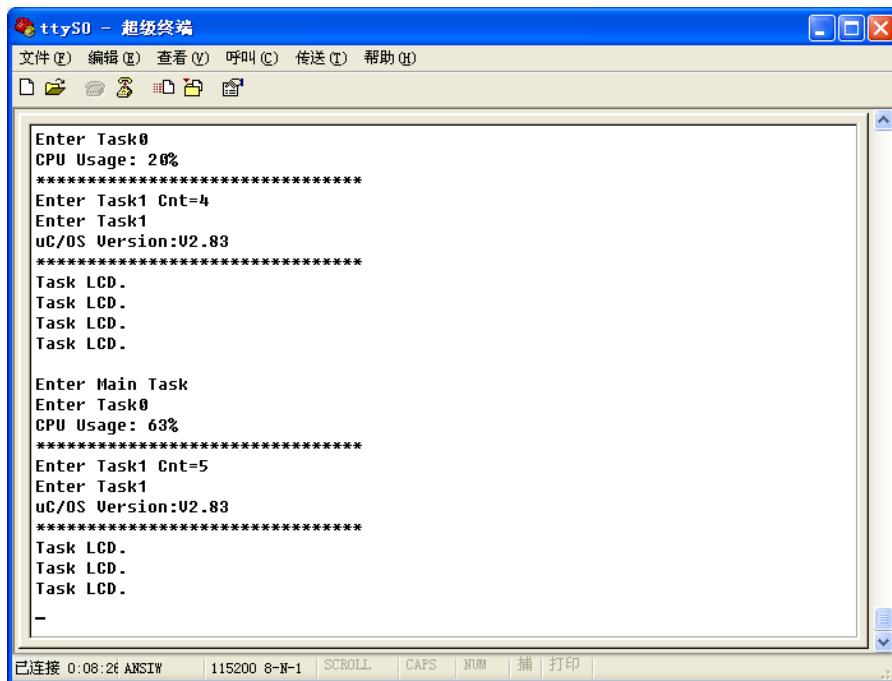
TO BE BEST

TO DO GREAT

广州友善之臂计算机科技有限公司



(6) 下载结束后，会自动运行，串口出现如下信息：



同时

使用 7 寸屏时，会出现如下图片背景的界面：



使用 NEC3.5 寸屏时，会出现如下图片背景的界面：



#### 4.5.3 把 uCos2 烧写到 Nand Flash 运行

上面的步骤是把可执行文件下载到内存中执行，为了脱离 PC 运行，需要使用 Supervivi 的功能[a](Absolute User Application)把它烧写到 Nand Flash 中，步骤如下：

(1)连接好开发板电源，串口线，USB 线，并设置开发板为 Nor Flash 启动系统，分别打开串口超级终端和 DNW，上电启动开发板。



追求卓越 创造精品

TO BE BEST

TO DO GREAT

广州友善之臂计算机科技有限公司

(2)保证 USB 驱动已经安装好(前面已经详细介绍了 USB 驱动的安装方法), 这时可以看到 DNW 的标题栏显示[USB: OK], 如果没有安装好驱动会显示[USB: x], 如图所示:



(3)这时在超级终端的 BIOS 功能菜单中选择功能号[a], 出现 USB 下载等待提示信息:

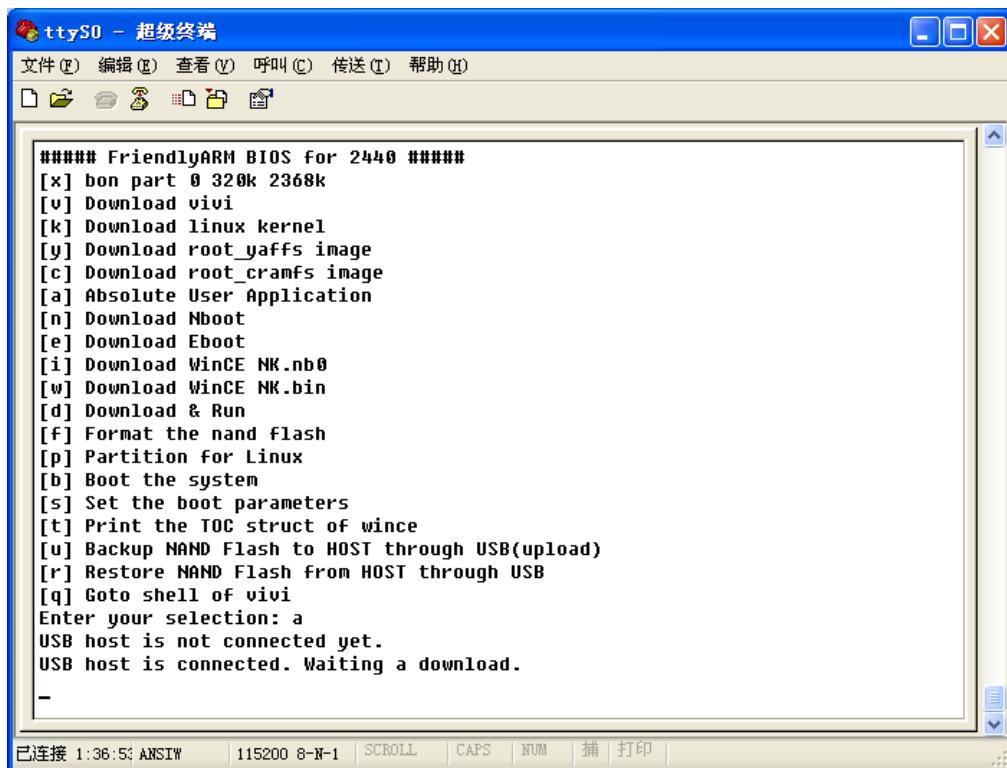


追求卓越 创造精品

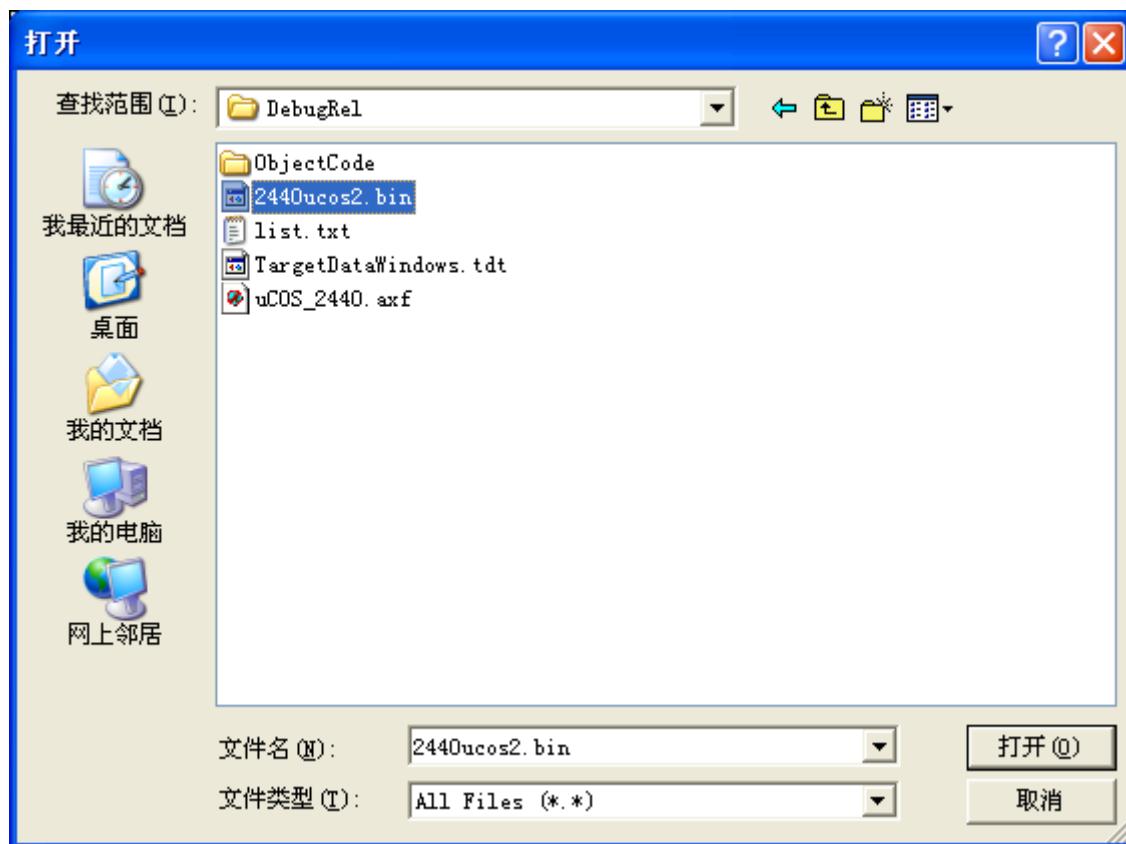
TO BE BEST

TO DO GREAT

广州友善之臂计算机科技有限公司



(4)点击 DNW 程序的“USB Port”→“Transmit”，如图选择刚刚编译出的映象文件(光盘“images”目录中有已经编译好的可执行文件)，这样就开始下载了，下载完毕，supervivi会把它自动烧写到 Nand Flash 起始 block 0 中；把 S2 开关拨至“NAND”一侧，选择从 Nand Flash 启动系统；这时重新开机或者复位，就可以看到和在内存中完全一样的 uCos2 运行界面了。

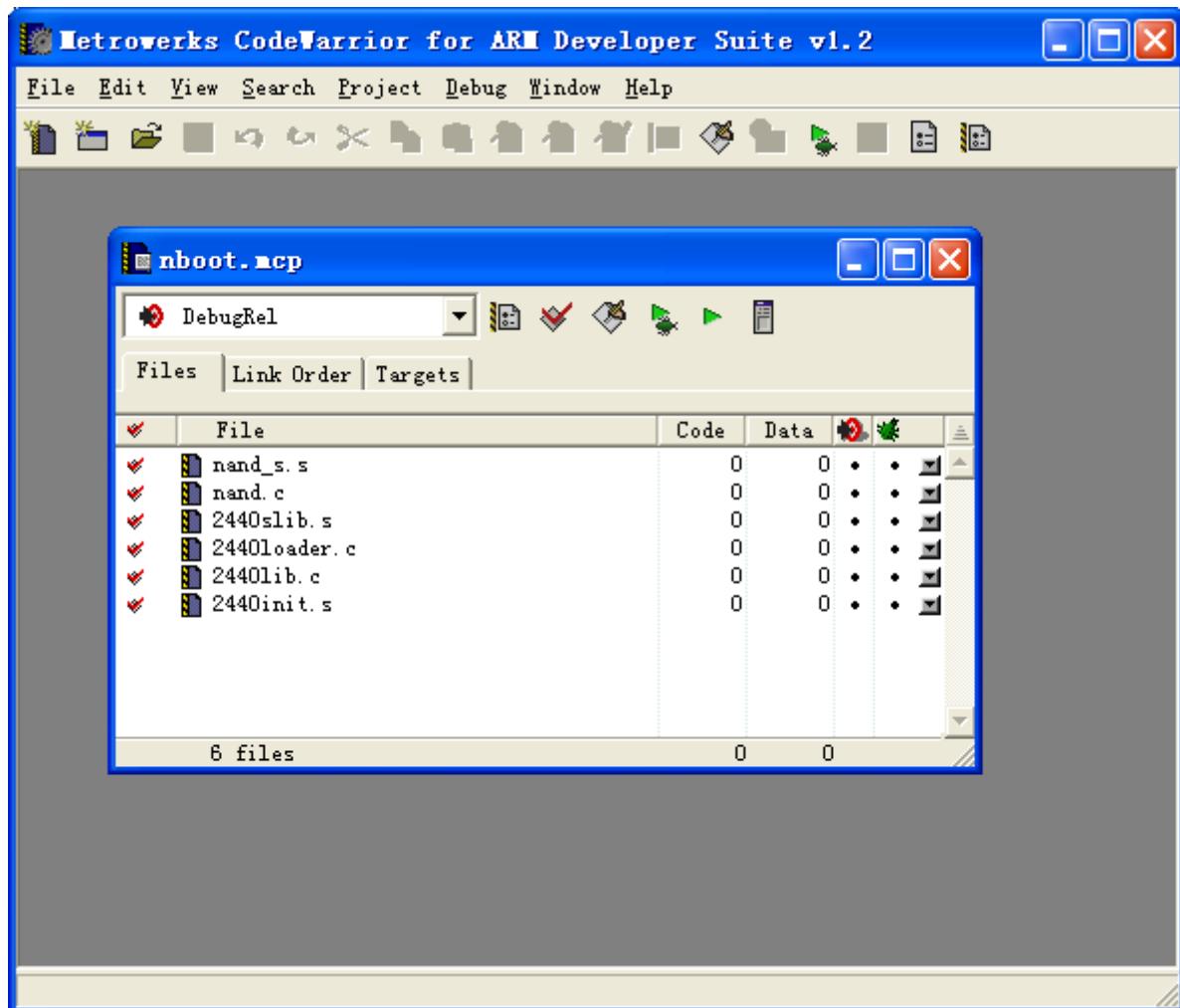


## 4.6 NBOOT 的编译和烧写

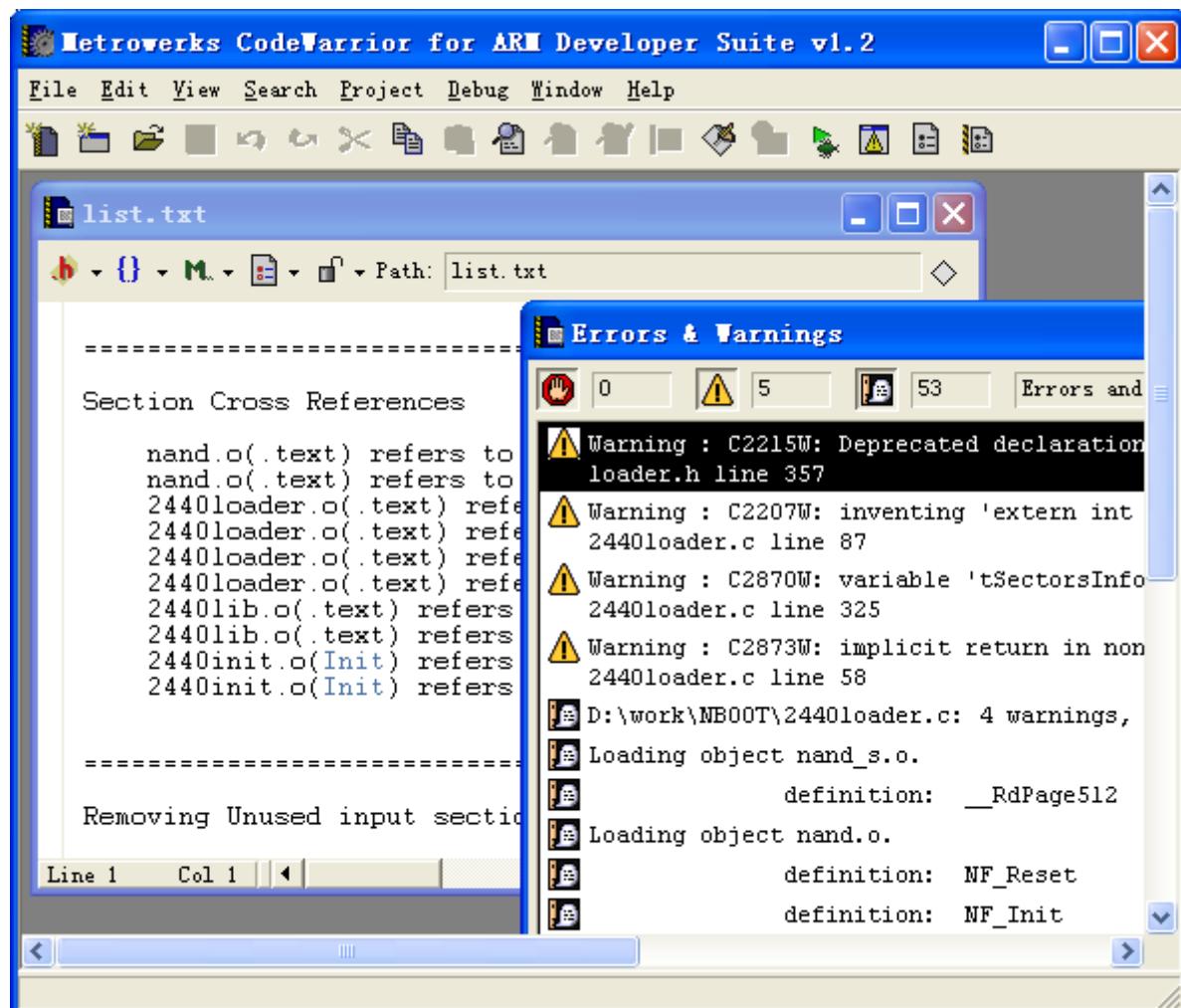
NBOOT 是一个十分简单的 bootloader，它主要用来启动 WINCE 内核，使用 NBOOT 启动 WINCE 比 supervivi 的速度要快一些，下面是 NBOOT 的编译和烧写步骤。

### 4.6.1 编译 NBOOT

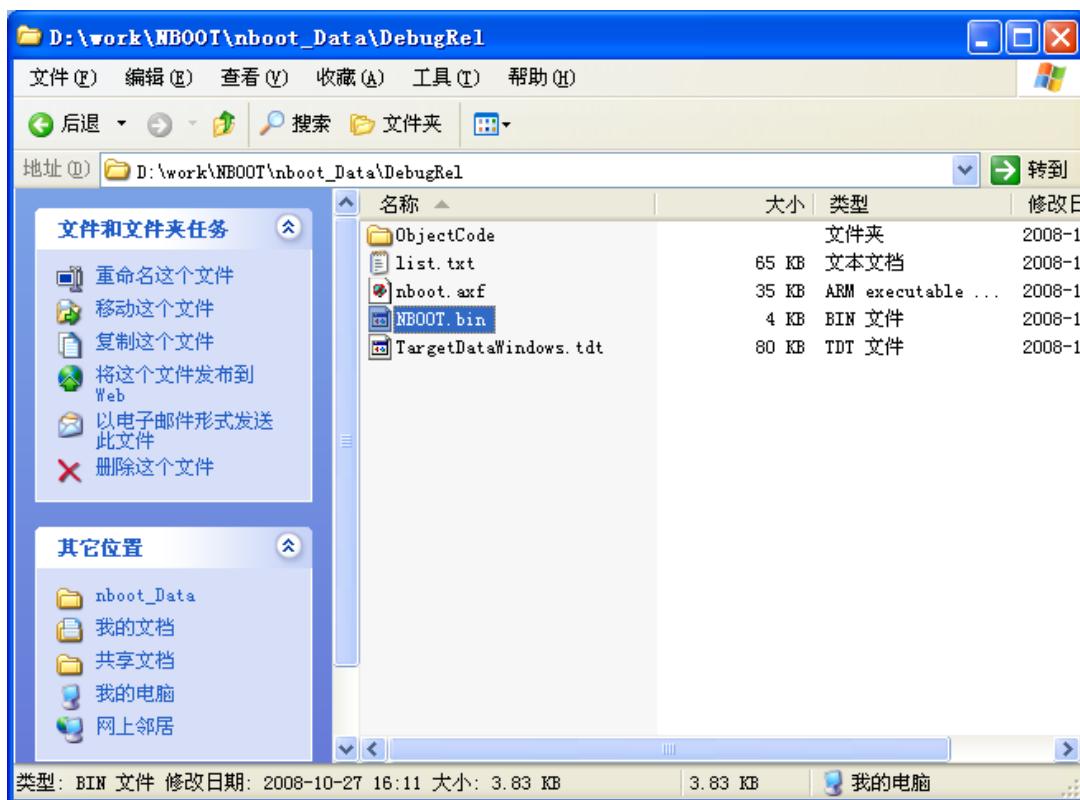
把光盘中“WindowsCE5.0”目录中的文件夹“NBOOT”文件夹复制到硬盘的某一个目录(在此为 D:\work)，去掉其只读属性，运行 ADS1.2 集成开发环境，点 **File->Open...** 打开 nboot.mcp 文件，如图。



这时点菜单 Project→Make 或者直接按 F7 键，开始编译 nboot 项目，编译完毕如图：



在 D:\work\NBOOT\nboot\_Data\DebugRel 目录下会生成 uboot.bin 可执行文件，如图。



## 4.6.2 把 NBOOT 烧写到 Nand Flash

(1)连接好开发板电源，串口线，USB 线，并**设置拨动开关 S2 为 Nor Flash 启动系统**，分别打开串口超级终端和 DNW，上电启动开发板。

(2)保证 USB 驱动已经安装好(前面已经详细介绍了 USB 驱动的安装方法)，这时可以看到 DNW 的标题栏显示[USB: OK]，如果没有安装好驱动会显示[USB: x]，如图所示：



追求卓越 创造精品

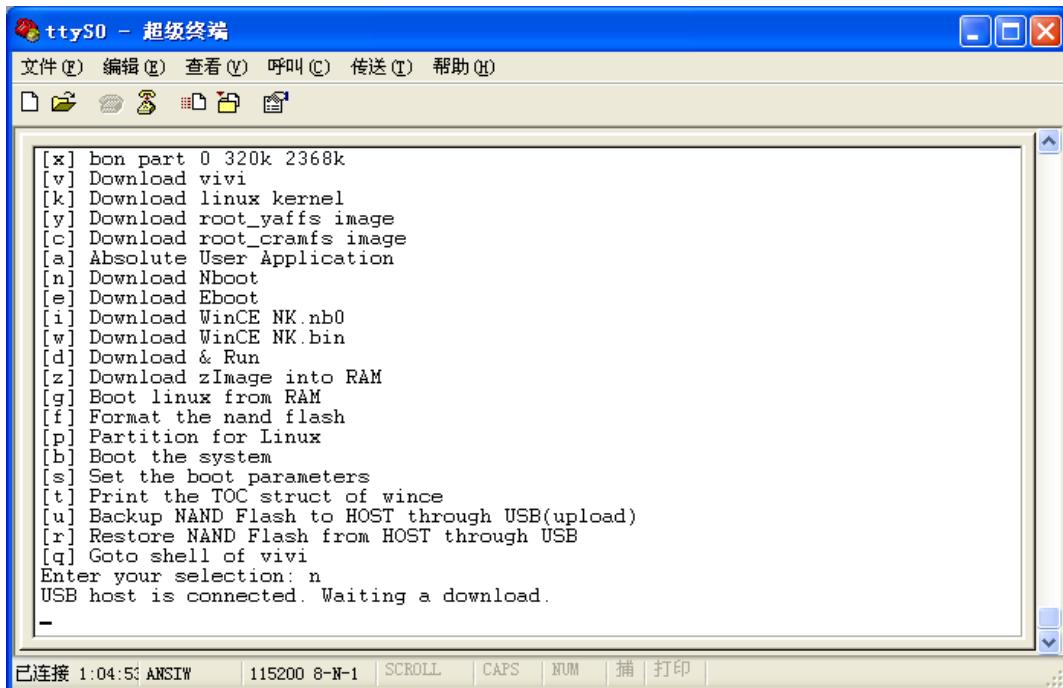
TO BE BEST

TO DO GREAT

广州友善之臂计算机科技有限公司



(3)这时在超级终端的 BIOS 功能菜单中选择功能号[n]，出现 USB 下载等待提示信息：



(4)点击 DNW 程序的“USB Port”→“Transmit”，如图选择刚刚编译出的映象文件(光盘“images”目录中有已经编译好的可执行文件)，这样就开始下载了，瞬间就可下载完毕，



追求卓越 创造精品

TO BE BEST

TO DO GREAT

广州友善之臂计算机科技有限公司

supervivi 会把它自动烧写到 Nand Flash 起始 block 0 中；把 S2 开关拨至“NAND”一侧，选择从 Nand Flash 启动系统；如果系统中已经烧写好了 WINCE 内核映象文件(见手册 3.3 章节)稍等片刻，就会看到 wince 启动了。

使用 NBOOT 启动 wince 的串口信息如下：

```
chainInfo.dwFlashAddress: 0X0000000000
chainInfo.dwLength: 0X0000000000
}

Jumping to image at virtual address 0x8C201000h
+ToPhysicalAddr:0x8C201000
-ToPhysicalAddr:0x30201000

::: Physical Launch Address: 0x30201000h

WinCE NAND Boot v1.00
Oct 27 2008 16:11:22
*****
dwEntry is 0x00000001
Sector addr on NAND: 0x000000520
TotalSector: 0x00000df24

JumpAddr is 0x30201000
Reading Kernel| Image from NAND
dwSector: 0x000000520
dwLength: 0x00000df24
dwRAM: 0x30200000
Launch wince
```

## 第五章 建立 Linux 开发环境

本章从在 PC 机上安装 **Redhat 9.0** 开始，详细介绍了如何建立 Linux 开发环境。

注意：如果您对 Linux 系统的开发环境和开发方法还不熟悉，请一定使用 Redhat 9.0 平台，使用其他版本的 Linux 会出现错误。

### 5.1 基于 Redhat Linux9.0 的开发环境建立

#### 5.1.1 完全图解安装 Redhat9.0

注意: **Redhat 9.0 只有一种发行版本，总共三张安装光盘**

Step1：将的安装光盘放到 CDROM/DVD 里,将 BIOS 改为从 CDROM 启动，启动后系统将会提示您选择安装方式，如果要以图形化安装直接按回车，如果要以 TEXT 模式安装则打入 linux text 然后回车。

Step2：然后进入下一步，检查安装盘，一般不需要检测，所以选择了 Skip（跳过）

Step3：过一会儿就进入安装图形化画面，点击 Next 即可.





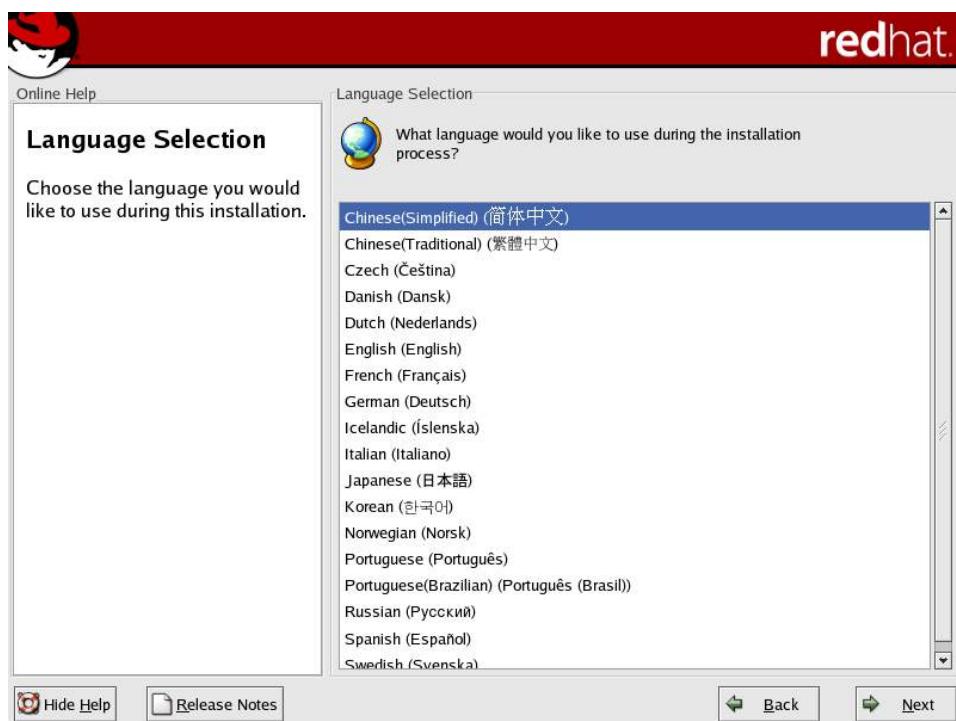
追求卓越 创造精品

TO BE BEST

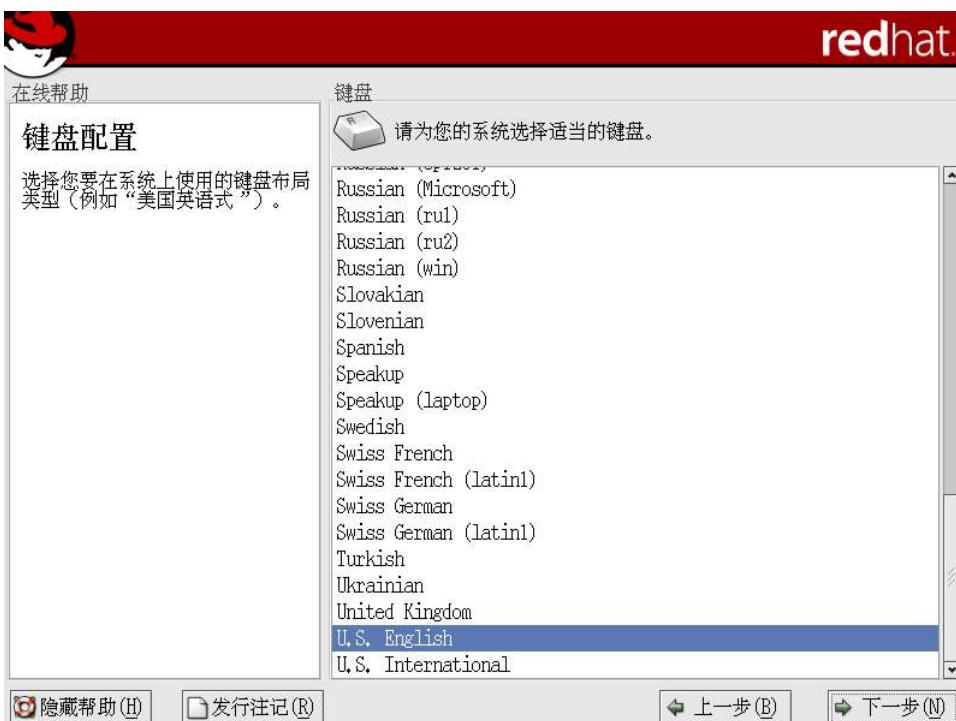
TO DO GREAT

广州友善之臂计算机科技有限公司

Step4: 选择安装过程用什么语言, 这里选择的是简体中文



Step5: 选键盘, 我们一般选美式键盘即可.



Step6: 选择鼠标, 按照自己的鼠标类型选择即可, 我的是 PS/2 的. (注: 不要模拟三键)

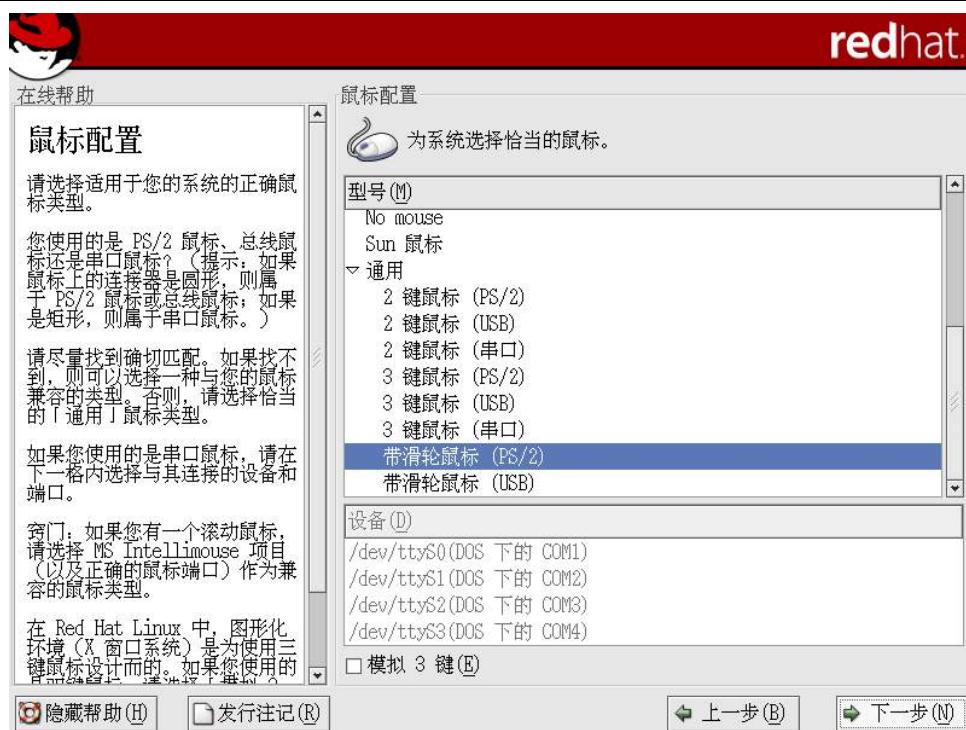


追求卓越 创造精品

TO BE BEST

TO DO GREAT

广州友善之臂计算机科技有限公司



Step7: 选择安装类型,请选择"定制"安装



Step8: 这一步是比较关键的,也是很麻烦的,就是给硬盘分区,请选择手工分区: linux 自带的 DiskDruid



追求卓越 创造精品

TO BE BEST

TO DO GREAT

广州友善之臂计算机科技有限公司



Step9: 先看看分区前的硬盘使用情况



Step10: 按上一步的图例, 点击"删除", 根据提示信息把以前的系统完全删除(如果您的系统有足够的空间, 直接进入下一步也可以)

Step11: 按 Step9 的图例, 点"新建", 跳出"添加新区"窗口, 按图例选择文件系统类型



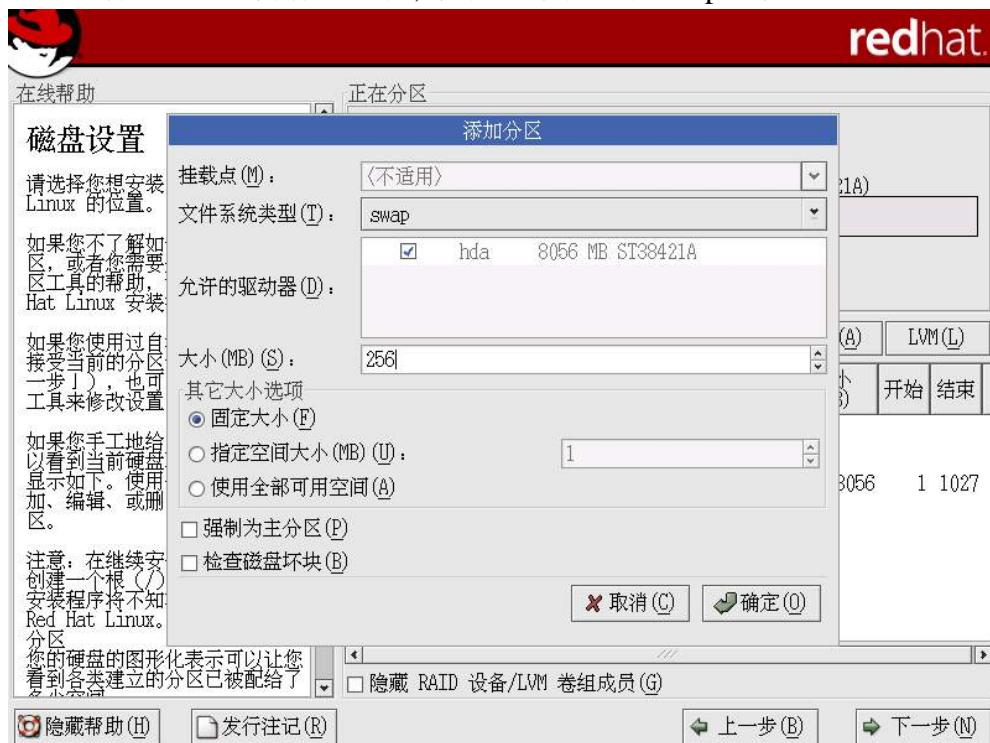
追求卓越 创造精品

TO BE BEST

TO DO GREAT

广州友善之臂计算机科技有限公司

为 swap, 大小一般选择为您 PC 内存的 2 倍, 然后点确定返回 step9 界面



Step12: 再点"新建", 跳出"添加新区"窗口, 按图例选择挂接点为"/", 文件系统类型为 ext3, 使用全部可用空间, 并强制为主分区



Step13: 最后分区表如图即可, 分区成功!



Step14: 编辑引导菜单, 如果你是多系统, 就可以有多个选项, 你可以设置任意一个为默认启动项。





追求卓越 创造精品

TO BE BEST

TO DO GREAT

广州友善之臂计算机科技有限公司

Step15: 置网络, 不要设置为 DHCP, 我们是静态的 IP, 点编辑, 将使用 DHCP 的钩去掉, 在下面输入 IP 和子网掩码



Step16: 手工设置里填入你的主机名, 这里是 FriendlyARM, 网关和 DNS 可有可无



Step17: 防火墙配置, 请选择"无防火墙"



追求卓越 创造精品

TO BE BEST

TO DO GREAT

广州友善之臂计算机科技有限公司



redhat.

在线帮助

### 防火墙配置

防火墙设立在您的计算机和网络之间，用来判定网络中的远程用户有权限访问您的计算机上的哪些资源。一个正确配置的防火墙可以极大地增加您的系统安全性。

为您的系统选择恰当的安全级别。

「高级」 - 选择「高级」，您的系统就不会接受那些没有被您具体指定的连接。只有下列连接是默认允许的：

- DNS 回应
- DHCP - 因此，任何使用 DHCP 的网络接口都可以被正确的配置。

使用「高级」将不允许下列连接：

- 活跃状态 FTP (在多数客户机中默认使用的被动状态)

防火墙配置  
选择系统的安全级别：  
 高级 (G)     中级 (M)     无防火墙 (O)  
 使用默认的防火墙规则 (D)     定制 (C)  
信任的设备 (T) :  eth0  
允许进入 (A) :  
 WWW (HTTP)  
 FTP  
 SSH  
 DHCP  
 Mail (SMTP)  
 Telnet  
其它端口 (P) :  
  
 隐藏帮助 (H)     发行注记 (R)       

Step18：选择系统语言，这里选择简体中文

在线帮助

### 附加语言支持

选择您的默认语言。这个默认语言将会是您在安装完毕后在系统上使用的语言。如果您选择要安装其它语言，在安装后您就能够改变所要使用的默认语言。

Red Hat Linux 可以安装并支持数千种语言。要在您的系统中使用多种语言，您可以指定要安装的语言，或者在您的系统上安装所有可用的语言。

使用「重设」按钮来取消您的选择。

其它语言支持  
选择系统默认语言 (C) : Chinese (P.R. of China)  
选择您想在该系统上安装的其它语言 (A) :  
 Arabic (Yemen)  
 Basque (Spain)  
 Belarusian  
 Bosnian (Bosnia and Herzegovina)  
 Breton (France)  
 Bulgarian  
 Catalan (Spain)  
 Chinese (Hong Kong)  
 Chinese (P.R. of China)  
 Chinese (Taiwan)  
 Cornish (Britain)  
 Croatian  
 Czech  
 Danish  
 Dutch (Belgium)  
 Dutch (Netherlands)  
 English (Australia)  
  
 隐藏帮助 (H)     发行注记 (R)       

Step19：这一步选择时区，我们用亚洲上海即可，即默认即可



追求卓越 创造精品

TO BE BEST

TO DO GREAT

广州友善之臂计算机科技有限公司

redhat.

在线帮助

### 时区选择

您可以通过选择计算机所处的物理位置来设置时区，也可以按照与世界协调时间（又称 UTC）的时区偏差来设置时区。

注意屏幕顶端的两个活页标签。第二个活页标签给您提供按位置配置的能力。

您可以在交互式地图上点击特定的城市（以黄点表示），所选城市旁边会显示一个红色的 X。

您也可以滚动城市列表，然后选择所需的时区。

第二个标签使您可以使用世界协调时间（UTC）时差来设置时区。您可以在此找到一个供您选择的时差列表以及一个用于设置夏令时的选项。

两个标签均提供了一个「系统时钟使用 UTC」的选项。（UTC，又称 GMT，将会允许您的系统正确处理夏令时。）如果您的系统

选择时区

位置 UTC 偏移

南极洲/帕玛 - 帕马站, 安维斯岛

位置 (L)	描述
亚洲/曼谷	东经 100 度 (东经 100 度, 东经 100 度)
亚洲/万象	
亚洲/上海	中国东部 - 北京、广东、上海等地
亚洲/东京	
亚洲/乌兰巴托	蒙古地区

系统时钟使用 UTC (U)

上一步 (B) 下一步 (N)

Step20：设置管理员密码，即 root 用户的密码，root 是超级管理员

redhat.

在线帮助

### 设置根口令

只有在管理时才使用根帐号。在一般使用下，建立一个非根帐号，或在您需要快速修复某项事物时使用 su - 命令获取根访问权。这些基本的规则可以减少由于键入错误或不正确的命令而损害系统的机

会。

设置根口令

请输入该系统的根用户(管理员)口令。

根口令： (P)  \*\*\*\*\*

确认 (C) :  \*\*\*\*\*

根口令被接受。

上一步 (B) 下一步 (N)

Step21：进行验证配置，选择默认即可



追求卓越 创造精品

TO BE BEST

TO DO GREAT

广州友善之臂计算机科技有限公司



Step22: 选择软件包，把右边的滚动条拉到最下方，找到"全部"选项，这样才算真正完全安装 Redhat 9.0



Step23: 软件包安装完毕，点下一步开始进行其他配置



追求卓越 创造精品

TO BE BEST

TO DO GREAT

广州友善之臂计算机科技有限公司



Step24：选择是否创建引导盘，这里选择“否”，点下一步



Step25：系统将自动配置 X，按默认即可



追求卓越 创造精品

TO BE BEST

TO DO GREAT

广州友善之臂计算机科技有限公司

redhat.



Step26: 自动探测到您使用的显示器，这里检测使用的是 MAG 770PF，最新的机器有可能自动检测不到，请到网上查找相关文章自己解决。





追求卓越 创造精品

TO BE BEST

TO DO GREAT

广州友善之臂计算机科技有限公司

Step27: 定制图形化配置, 按默认即可



Step28: 安装完毕



## 5.1.2 建立交叉编译环境

Linux 下开发环境的建立主要就是建立交叉编译环境，在 Redhat 9.0 里面建立一个能编译 arm-linux 内核及驱动、应用程序等开发环境的步骤如下。

先将光盘目录 linux\ 中的 arm-linux-gcc-3.3.2.tgz 、 arm-linux-gcc-2.95.3.tgz 和 arm-linux-gcc-3.4.1.tgz 拷贝到某个目录下如 tmp\, 然后进入到该目录，执行解压命令：

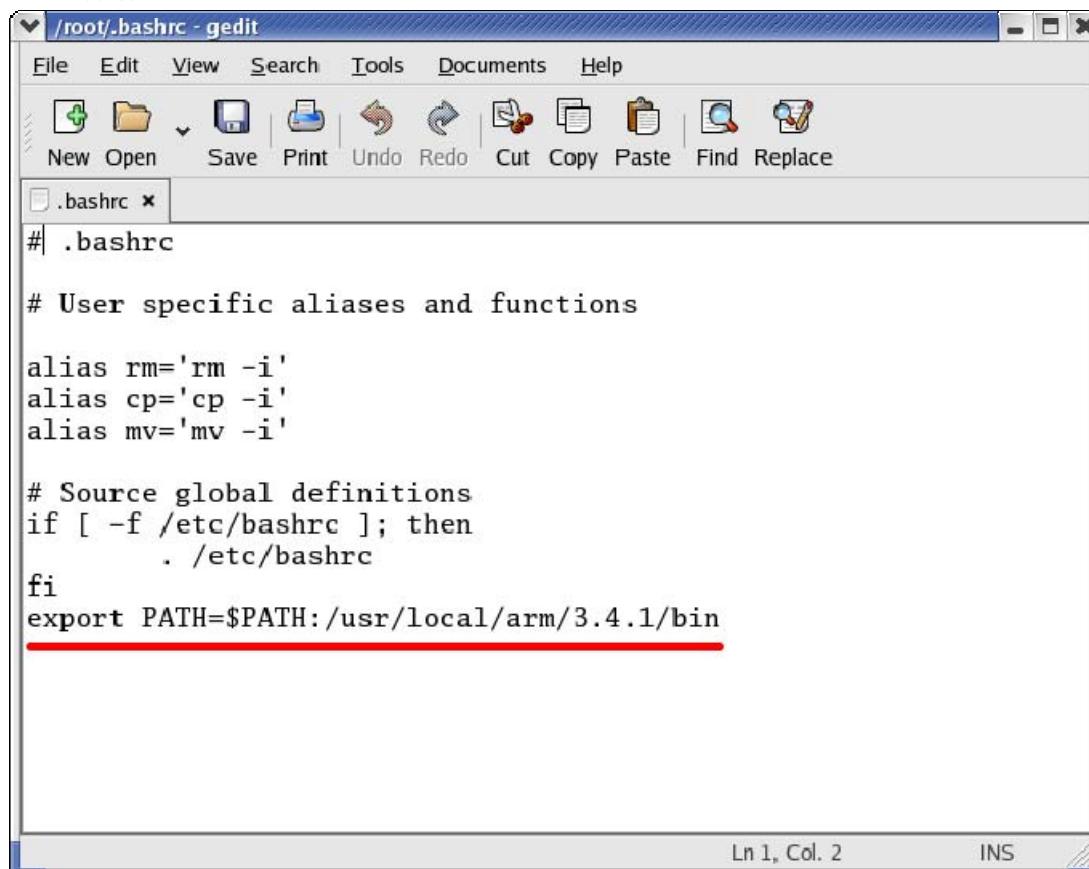
```
#cd /tmp  
#tar xvzf arm-linux-gcc-3.3.2.tgz -C /  
#tar xvzf arm-linux-gcc-2.95.3.tgz -C /  
#tar xvzf arm-linux-gcc-3.4.1.tgz -C /  
#mkdir -p /opt/FriendlyARM/minи2440 ;建立工作目录，备用
```

执行该命令，将把 arm-linux-gcc( 版本 3.3.2, 2.95.3 和 3.4.1) 分别安装到 /usr/local/arm/2.95.3 和 /usr/local/arm/3.4.1 目录，其中 3.3.2 版本是用来编译 Qtopia/Embedded 的， 2.95.3 版本是用来编译 VIVI 的， 3.4.1 版本是用来编译内核的，两个版本均可以用来编译应用程序等。

然后运行命令

```
#gedit /root/.bashrc
```

编辑/root/.bashrc 文件，在最后一行 **export PATH=\$PATH:/usr/local/arm/3.4.1/bin** 如图，保存退出。



```
# .bashrc  
  
# User specific aliases and functions  
  
alias rm='rm -i'  
alias cp='cp -i'  
alias mv='mv -i'  
  
# Source global definitions  
if [ -f /etc/bashrc ]; then  
    . /etc/bashrc  
fi  
export PATH=$PATH:/usr/local/arm/3.4.1/bin
```

重新登录 Redhat 系统(不必重启机器, 开始->logout 即可), 使以上设置生效, 在命令行输入 arm-linux-gcc -v, 会出现如下信息, 这说明交叉编译环境已经成功安装。

```

root@localhost:~# arm-linux-gcc -v
Reading specs from /usr/local/arm/3.4.1/lib/gcc/arm-linux/3.4.1/specs
Configured with: /work/cross tool-0.27/build/arm-linux/gcc-3.4.1-glibc-2.
3.2/gcc-3.4.1/configure --target=arm-linux --host=i686-host_pc-linux-gnu
--prefix=/usr/local/arm/3.4.1 --with-headers=/usr/local/arm/3.4.1/arm-l
inux/include --with-local-prefix=/usr/local/arm/3.4.1/arm-linux --disabl
e-nls --enable-threads=posix --enable-symvers=gnu --enable-__cxa_atexit
--enable-languages=c,c++ --enable-shared --enable-c99 --enable-long-long
Thread model: posix
gcc version 3.4.1
[root@localhost root]#

```

如果设置 2.95.3 版本的编译器, 只需更改 **export PATH=\$PATH:/usr/local/arm/2.95.3/bin** 即可。

**注意: 2.95.3 版本的交叉编译器和3.4.1 版本的交叉编译器不能同时使用。**

### 5.1.3 配置网络文件系统 NFS 服务

如果您已经按照以上章节介绍的方法完全安装好了 Redhat 9.0, 则 NFS 相关软件都已经缺省安装好了, 请按照以下步骤建立和配置 NFS 服务。

(1) 设置共享目录

运行命令

**#gedit /etc/exports**

编辑 nfs 服务的配置文件(注意: 第一次打开时该文件是空的), 添加以下内容:

/opt/FriendlyARM/MINI2440/root\_nfs \*(rw,sync,no\_root\_squash)

其中:

**/opt/FriendlyARM/mini2440/root\_nfs** 表示 nfs 共享目录, 它可以作为开发板的根文件系统通过 nfs 挂接;

\* 表示所有的客户机都可以挂接此目录

rw 表示挂接此目录的客户机对该目录有读写的权力

no\_root\_squash 表示允许挂接此目录的客户机享有该主机的 root 身份

(2) 建立共享目录

拷贝光盘中的 root\_nfs.tgz 文件到某一个目录, 进入此目录, 执行以下命令:

**#tar xvzf root\_nfs.tgz -C /opt/FriendlyARM/mini2440/root\_nfs**

该命令将把 root\_nfs 的内容解压安装到 /opt/FriendlyARM/mini2440/root\_nfs 目录。

### (3) 启动和停止 nfs 服务

在命令行下运行:

```
#/etc/init.d/nfs start
```

这将启动 nfs 服务，可以输入以下命令检验 nfs 该服务是否启动。

```
# mount -t nfs localhost: /opt/FriendlyARM/mini2440/root_nfs /mnt/
```

如果没有出现错误信息，您将可以浏览到 /mnt 目录中的内容和 /opt/FriendlyARM/mini2440/root\_nfs 是一致的。

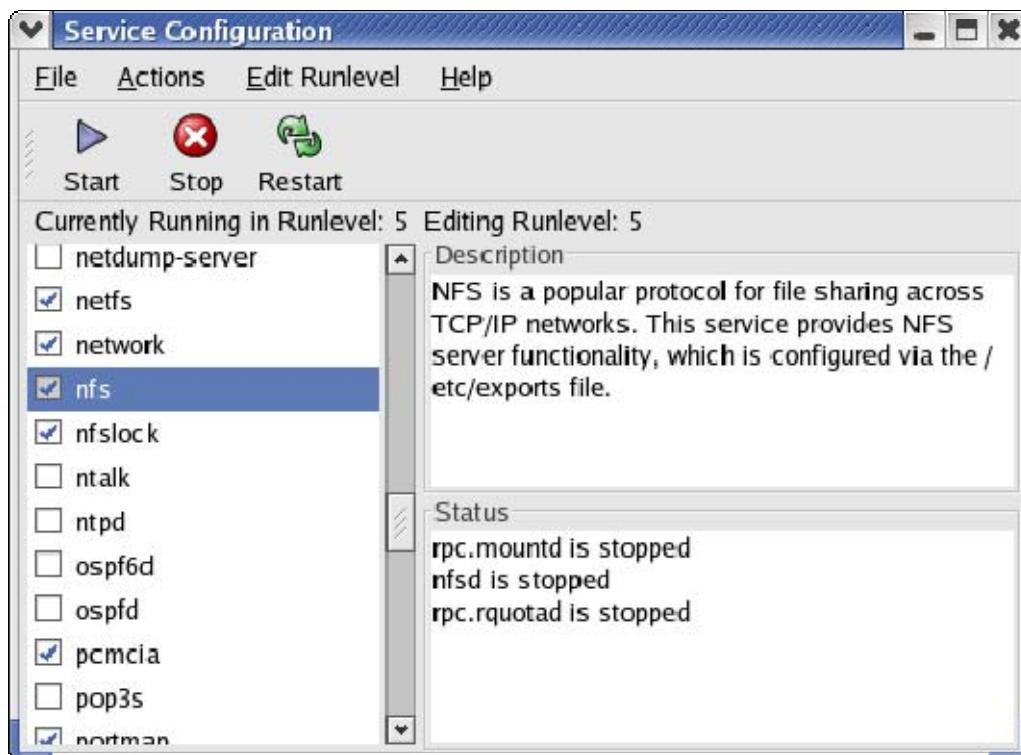
使用这个命令可以停止 nfs 服务:

```
#/etc/init.d/nfs stop
```

为了在每次开机时系统都自动启动该服务，可以输入

```
#redhat-config-services
```

打开系统服务配置窗口，在左侧一栏找到 nfs 服务选项框，并选中它，然后点 File->Save Changes 保存设置，如图。



## 5.1.4 通过 NFS 启动系统

当 NFS 服务设置好并启动后，我们就可以把 NFS 作为根文件系统来启动开发板了。通过使用 NFS 作为根文件系统，开发板的“硬盘”就可以变得很大，因为您使用的是主机的硬盘，这是使用 linux 作为开发经常使用的方法，

设置目标板启动模式为 Nand Flash 启动，连接好电源，串口线，网线；打开串口终端，



追求卓越 创造精品

TO BE BEST

TO DO GREAT

广州友善之臂计算机科技有限公司

在开机或者复位的时候迅速按下 PC 机的空格键，这样我们就进入了 vivi 模式，输入以下命令：

```
Supervivi>param set linux_cmd_line "console=ttySAC0 root=/dev/nfs
nfsroot=192.168.1.111:/opt/FriendlyARM/mini2440/root_nfs
ip=192.168.1.70:192.168.1.111:192.168.1.111:255.255.255.0:MINI2440.arm9.net:eth0:off"
```

其中，param set linux\_cmd\_line 是设置启动 linux 时的命令参数。其各参数的含义如下：

nfsroot 是自己开发主机的 IP 地址。

“ip=” 后面：

第一项(192.168.1.70)是目标板的临时 IP(注意不要和局域网内其他 IP 冲突)；

第二项(192.168.1.111)是开发主机的 IP；

第三项(192.168.1.111)是目标板上网关(GW)的设置；

第四项(255.255.255.0)是子网掩码；

第五项是开发主机的名字(一般无关紧要，可随便填写)

eth0 是网卡设备的名称。

由于该命令比较长，容易输入错误，我们已经把它写入了光盘的 nfs.txt 文件中，这样您直接复制过来就可以了。

```
DIVN_UPPL0
MPLLVal [M:7fh,P:2h,S:1h]
CLKDIVN:5h

+-----+
| FriendlyARM SBC2440 USB Downloader ver1.0 |
+-----+
USB: IN_ENDPOINT:1 OUT_ENDPOINT:3
FORMAT: <ADDR(DATA):4>+<SIZE(n+10):4>+<DATA:n>+<CS:2>
NOTE: Power off/on or press the reset button for 1 sec
      in order to get a valid USB device address.

NAND device: Manufacture ID: 0xec, Chip ID: 0x76 (Samsung K9D1208VOM)
Could not found stored vivi parameters. Use default vivi parameters.
Press Return to start the LINUX/Wince now, any other key for vivi
type "help" for help.
FriendlyARM> param set linux_cmd_line "console=ttySAC0 root=/dev/nfs nfsroot=19
.168.1.111:/opt/FriendlyARM/QQ2440/root_nfs ip=192.168.1.70:192.168.1.111:192.1
8.1.111:255.255.255.0:sbc2440.arm9.net:eth0:off"
Change linux command line to "console=ttySAC0 root=/dev/nfs nfsroot=192.168.1.1
1:/opt/FriendlyARM/QQ2440/root_nfs ip=192.168.1.70:192.168.1.111:192.168.1.111:
55.255.255.0:sbc2440.arm9.net:eth0:off"
FriendlyARM>
```



追求卓越 创造精品

TO BE BEST

TO DO GREAT

广州友善之臂计算机科技有限公司

然后输入 boot，按回车就可以通过 nfs 启动系统了。

The screenshot shows a terminal window titled "超级终端" (Super Terminal) with the command "TCP rno registeredi /etc/initdep 30000" entered. The terminal output details the boot process:

```
mmcblk0: mmc0:c734 SD016 14400KiB
/dev/mmc/blk0:<7>MMC: starting cmd 37 arg c7340000 flags 00000009
p1
IP-Config: Complete:
    device=eth0, addr=192.168.1.70, mask=255.255.255.0, gw=192.168.1.111,
    host=sbc2440, domain=, nis-domain=arm9.net,
    bootserver=192.168.1.111, rootserver=192.168.1.111, rootpath=
Looking up port of RPC 100003/2 on 192.168.1.111
Looking up port of RPC 100005/1 on 192.168.1.111
VFS: Mounted root (nfs filesystem).
Mounted devfs on /dev
Freeing init memory: 148K
[27/Jun/2007:16:02:49 +0000] boa: server version Boa/0.94.13
[27/Jun/2007:16:02:49 +0000] boa: server built Feb 28 2004 at 21:47:23.
[27/Jun/2007:16:02:49 +0000] boa: starting server pid=279, port 80

Please press Enter to activate this console. MPEG Audio Decoder 0.15.0 (beta) -
Copyright (C) 2000-2003 Robert Leslie et al.
    Title: 上海滩
    Artist: 叶丽仪
    Year: 2000
    Genre: Goa

[root@FriendlyARM /]#
```

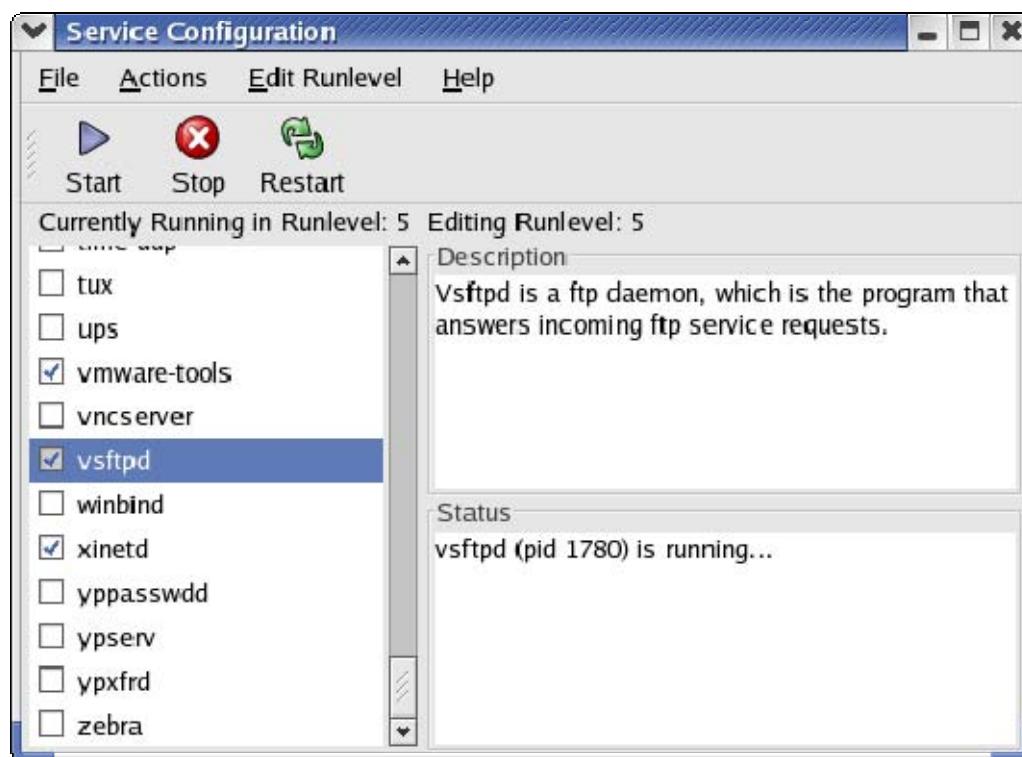
At the bottom of the terminal window, there is a status bar with the text "已连接 1:34:26 ANSIW" and a row of icons: SCROLL, CAPS, NUM, 插, 打印.

## 5.1.5 配置 PC 机 Linux 的 ftp 服务

和配置 nfs 服务相同，在命令行输入

**#redhat-config-services**

打开系统服务配置窗口，在左侧一栏找到 vsftpd 服务选项框，并选中它，然后点 File->Save Changes 保存设置，如图。

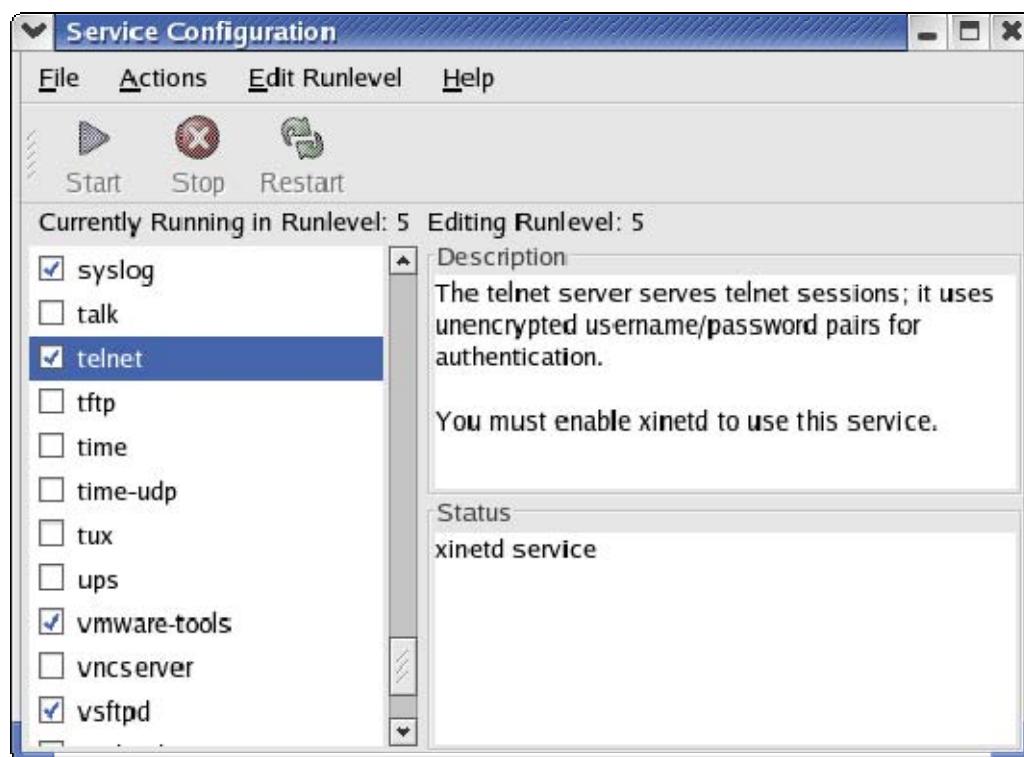


## 5.1.6 配置 PC 机的 telnet 服务

和配置 nfs 服务相同，在命令行输入

**#redhat-config-services**

打开系统服务配置窗口，在左侧一栏找到 nfs 服务选项框，并选中它，然后点 File->Save Changes 保存设置，下次开机的时候该系统就启动了 telnet 服务，如图。



## 5.1.7 在 Redhat 中添加新用户

虽然我们已经设置了 telnet 和 ftp 服务，但一般需要向外面提供一个用户帐户才能使用该服务，增加一个用户帐户的命令如下：

**#adduser tom**

为该帐户设置密码：

**#passwd tom**

这时系统会提示你输入两次密码，按照提示来操作就可以了。

另外一种添加新用户的方法是图形界面操作，命令行下输入：

**#redhat-config-users**

出现如下图所示窗口

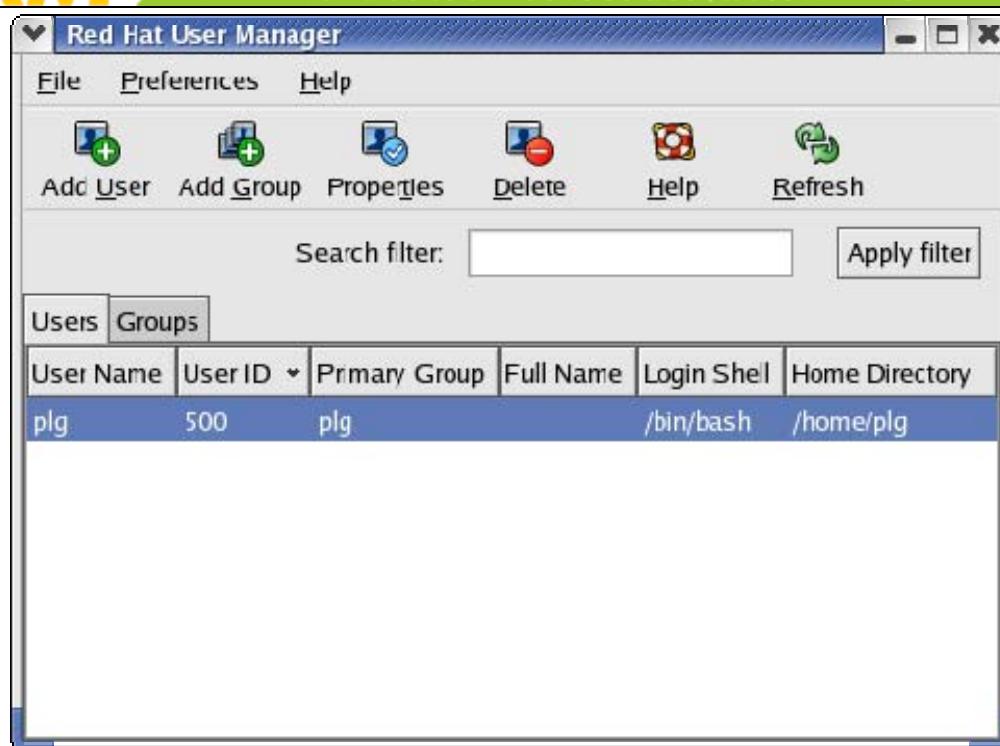


追求卓越 创造精品

TO BE BEST

TO DO GREAT

广州友善之臂计算机科技有限公司



点 Add User 按钮，出现添加新用户窗口，按提示操作就可以了。





## 第六章 嵌入式 Linux 应用开发入门指南

本节内容通过嵌入式 Linux 开发最简单的例子，介绍了如何编写和编译 Linux 应用程序，并下载到开发板运行起来，并介绍了如何制作和装载驱动程序模块，以及移植一些常见的开源软件。

嵌入式 Linux 资源丰富，我们不可能介绍到每一个细节，本文旨在提供一些嵌入式 Linux 经常用到的方法，为你打开奇妙世界的大门。

### 6.1 Hello,World!

#### 6.1.1 Hello,World 源代码

Hello,World 源代码位于光盘的 `linux/examples.tgz` 包中，如果您按上一节的步骤安装了开发环境，它将位于 `/opt/FriendlyARM/mini2440/examples/hello` 目录，其源代码如下：

**Hello, World 源代码:**

```
#include <stdio.h>

int main(void) {
    printf("hello, FriendlyARM!\n");
}
```

#### 6.1.2 编译 Hello,World

首先进入测试程序源代码目录

```
#cd /opt/FriendlyARM/mini2440/examples/hello
```

然后，使用命令行手工编译示例程序

```
#arm-linux-gcc -o hello main.c
```

或者借助编译脚本进行编译

```
#make
```

最后将生成 `hello` 可执行文件

#### 6.1.3 把 Hello,World 下载到开发板运行

将编译好的可执行文件下载到目标板目前主要四种方式：



追求卓越 创造精品

TO BE BEST

TO DO GREAT

广州友善之臂计算机科技有限公司

- 第一种：复制到介质(如优盘)
  - 第二种：通过网络传送文件到开发板
  - 第三种：通过串口传送文件到开发板
  - 第四种：通过 NFS(网络文件系统)直接运行
- 下面分别进行介绍：

### (1) 使用优盘

方法：先把编译好的可执行程序复制到优盘，再把优盘插到目标板上并挂载它，然后把程序拷贝到目标板的可执行目录/bin

步骤：

Step1：复制程序到优盘

把优盘插到 PC 的 USB 接口，执行以下命令把程序复制到优盘

```
#mount /dev/sda1 /mnt      ; 挂接优盘  
#cp hello /mnt            ; 复制刚才编译好的程序到优盘  
#umount /mnt              ; 卸载优盘
```

Step2：把程序从优盘拷贝到目标板并执行

把优盘插入到开发板的 USB Host 接口，执行以下命令复制程序到目标板

```
#mount /dev/sda1 /mnt      ; 挂接优盘  
#cp hello /mnt /sbin       ; 复制程序到目标板的可执行目录/sbin  
#umount /mnt              ; 卸载优盘  
#hello                      ; 执行程序
```

**注意：/dev/sda1 是一个连接文件，有的优盘插到目标板之后，不一定对应此连接文件，请参考上面的如何使用优盘的章节。**

### (2) 使用 ftp 传送文件

方法：使用 ftp 登录目标板，把编译好的程序上传；然后修改上传后目标板上的程序的可执行属性，并执行。

首先，在 PC 端执行，如图所示



追求卓越 创造精品

TO BE BEST

TO DO GREAT

广州友善之臂计算机科技有限公司

```
root@localhost:/opt/FriendlyARM/QQ2440/examples/hello
File Edit View Terminal Go Help
root@localhost:/opt/FriendlyARM/QQ2440          root@localhost:/opt/FriendlyARM/QQ2440/exampl...
[root@localhost hello]# ls
hello  hello.c  Makefile 生成的hello可执行文件
[root@localhost hello]# ftp 192.168.1.230 使用ftp命令登录开发板
Connected to 192.168.1.230.
220 FriendlyARM FTP server (Version 6.4/OpenBSD/Linux-ftpd-0.17) ready.
500 'AUTH GSSAPI': command not understood.
500 'AUTH KERBEROS_V4': command not understood.
KERBEROS_V4 rejected as an authentication type
Name (192.168.1.230:root): plg 输入用户名: plg
331 Password required for plg.
Password: 输入密码: plg
230 User plg logged in.
Remote system type is UNIX.
Using binary mode to transfer files.
ftp> bin 改变传输模式为 binary
200 Type set to I.
ftp> put hello 上传hello
local: hello remote: hello
227 Entering Passive Mode (192,168,1,230,4,1)
150 Opening BINARY mode data connection for 'hello'.
226 Transfer complete.
7414 bytes sent in 0.00024 seconds (3e-04 Kbytes/s)
ftp> by 退出ftp登录
221 Goodbye.
[root@localhost hello]#
```

在PC端执行的步骤

然后，在目标板一端执行，如图所示



COM1 (1) – CRT

[root@FriendlyARM plg]# cd  
[root@FriendlyARM /]# ls  
bin lib opt tmp  
dev linuxrc proc usr  
etc lost+found sbin var  
home mnt shanghaiitan.mp3 www  
[root@FriendlyARM /]# cd /home/plg/ —— 进入 /home/plg , 因为文件上传至此  
[root@FriendlyARM plg]# ls —— 查看该目录下的文件，可以看到hello  
hello  
[root@FriendlyARM plg]# chmod +x hello —— 改变文件的执行权限  
[root@FriendlyARM plg]# ls  
hello —— 可以看到文件的属性已经变了  
[root@FriendlyARM plg]# ./hello —— 执行hello  
hello, FriendlyARM!  
[root@FriendlyARM plg]#

### (3)通过串口传送文件到开发板

通过 2.3.6 一节我们学会了如何通过串口传送文件到开发板，您也可以通过相同的方法传送 hello 可执行程序，具体步骤在此不再详细描述，记得传送完毕把文件的属性改为可执行才能正常运行。

```
#chmod +x hello
```

#### (4) 通过网络文件系统 NFS 执行

Linux 中最常用的方法就是采用 NFS 来执行各种程序，这样可以不必花费很多时间下载程序，虽然在此下载 hello 程序用不了多久，一旦您的应用程序变得越来越大，您就会发现使用 NFS 运行的方便所在。

如同前面所讲述的那样,请先按照4.3一节搭建好NFS服务器系统,然后在命令行输入以下命令(假定服务器的IP地址为192.168.1.111):

```
#mount -t nfs -o noblock 192.168.1.1111:/opt/FriendlyARM/mini2440/root_nfs /mnt
```

挂接成功，您就可以进入/mnt 目录进行操作了，在您的 PC Linux 终端把 hello 复制到 opt/FriendlyARM/mini2440/root nfs 目录，然后在开发板的串口终端执行

```
#cd /mnt
```

#./hello



追求卓越 创造精品

TO BE BEST

TO DO GREAT

广州友善之臂计算机科技有限公司

## 6.2 嵌入式 Linux 程序开发入门

声明：以下 Linux 示例程序均为友善之臂原创，我们发现有的开发板厂商或者个人修改了 copyright 说明，据为己有，虽然国内对这种抄袭行为基本没有法律约束，但对我们对这种无耻的盗窃行为予以鄙视，并敬告大家要尊重原厂家的辛苦劳动。

### 6.2.1 LED 测试程序

程序源代码说明：

驱动源代码所在目录	/opt/FriendlyARM/mini2440/kernel-2.6.13/drivers/char
驱动程序名称	qq2440_leds.c
该驱动的主设备号	231
设备名	/dev/leds
测试程序源代码目录	/opt/FriendlyARM/mini2440/examples/led
测试程序名称	led.c
测试程序可执行文件名称	led

说明：LED 驱动已经被编译到缺省内核中，因此不能再使用 insmod 方式加载。

测试程序清单：

```
#include <stdio.h>
#include <stdlib.h>
#include <unistd.h>
#include <sys/ioctl.h>

int main(int argc, char **argv)
{
    int on;
    int led_no;
    int fd;

    /* 检查 led 控制的两个参数，如果没有参数输入则退出。 */
    if (argc != 3 || sscanf(argv[1], "%d", &led_no) != 1 || sscanf(argv[2], "%d", &on) != 1 ||
        on < 0 || on > 1 || led_no < 0 || led_no > 3) {
        fprintf(stderr, "Usage: leds led_no 0|1\n");
        exit(1);
    }

    /* 打开/dev/leds 设备文件 */
    fd = open("/dev/leds", 0);
```



追求卓越 创造精品

TO BE BEST

TO DO GREAT

广州友善之臂计算机科技有限公司

```
if (fd < 0) {  
    perror("open device leds");  
    exit(1);  
}  
  
/*通过系统调用 ioctl 和输入的参数控制 led*/  
ioctl(fd, on, led_no);  
  
/*关闭设备句柄*/  
close(fd);  
  
return 0;  
}
```

你可以按照上面的 hello 程序的步骤手工编译出 led 可执行文件，然后下载到开发板运行它。

## 6.2.2 测试按键

程序源代码说明：

驱动源代码所在目录	/opt/FriendlyARM/mini2440/kernel-2.6.13/drivers/char
驱动程序名称	mini2440_buttons.c
该驱动的主设备号	232
设备名	/dev/buttons
测试程序源代码目录	/opt/FriendlyARM/mini2440/examples/buttons
测试程序源代码名称	buttons_test.c
测试程序可执行文件名称	buttons
说明：按键驱动已经被编译到缺省内核中，因此不能再使用 insmod 方式加载。	

测试程序清单：

```
#include <stdio.h>  
#include <stdlib.h>  
#include <unistd.h>  
#include <sys/ioctl.h>  
#include <sys/types.h>  
#include <sys/stat.h>  
#include <fcntl.h>  
#include <sys/select.h>  
#include <sys/time.h>  
#include <errno.h>
```



追 求 卓 越 创 造 精 品

TO BE BEST

TO DO GREAT

广州友善之臂计算机科技有限公司

```
int main(void)
{
    int i;
    int buttons_fd;
    int key_value[4];

    /*打开键盘设备文件*/
    buttons_fd = open("/dev/buttons", 0);
    if (buttons_fd < 0) {
        perror("open device buttons");
        exit(1);
    }

    for (;;) {
        fd_set rds;
        int ret;

        FD_ZERO(&rds);
        FD_SET(buttons_fd, &rds);

        /*使用系统调用 select 检查是否能够从/dev/buttons 设备读取数据*/
        ret = select(buttons_fd + 1, &rds, NULL, NULL, NULL);

        /*读取出错则退出程序*/
        if (ret < 0) {
            perror("select");
            exit(1);
        }

        if (ret == 0) {
            printf("Timeout.\n");
        }
        /*能够读取到数据*/
        else if (FD_ISSET(buttons_fd, &rds)) {
            /*开始读取键盘驱动发出的数据，注意 key_value 和键盘驱动中定义为一致的
            类型*/
            int ret = read(buttons_fd, key_value, sizeof key_value);
            if (ret != sizeof key_value) {
                if (errno != EAGAIN)
                    perror("read buttons\n");
                continue;
            }
        }
    }
}
```



```
    } else {
        /*打印键值*/
        for (i = 0; i < 4; i++)
            printf("K%d %s, key value = 0x%02x\n", i,
                   (key_value[i] & 0x80) ? "released"      : \
                   key_value[i] ? "pressed down" : "", \
                   key_value[i]);
    }

}

/*关闭设备文件句柄*/
close(buttons_fd);
return 0;
}
```

你可以按照上面的 hello 程序的步骤手工编译出 buttons 可执行文件，然后下载到开发板运行它

### 6.2.3 UDP 网络编程

程序源代码说明：

驱动源代码所在目录	/opt/FriendlyARM/mini2440/kernel-2.6.13/drivers/net/
驱动程序名称	dm9000x.c
该驱动的主设备号	无
设备名	eth0 (网络设备并不在/dev 目录中出现)
测试程序源代码目录	/opt/FriendlyARM/mini2440/examples/udptak
测试程序源代码名称	udptalk.c
测试程序可执行文件名称	udptalk.c

说明：测试程序编译后可得到 x86 版本和 arm 版本，其源代码是完全一样的。

TCP/IP 提供了无连接的传输层协议：UDP(User Datagram Protocol，即用户数据报协议)。 UDP 与 TCP 有很大的区别，因为无连接的 socket 编程与面向连接的 socket 编程也有很大的差异。由于不用建立连接，因此每个发送个接收的数据报都包含了发送方和接收方的地址信息。

在发送和接收数据之前，先要建立一个数据报方式的套接字，该 socket 的类型为 SOCK\_DGRAM，用如下的调用产生：

```
sockfd=socket(AF_INET, SOCK_DGRAM, 0);
```

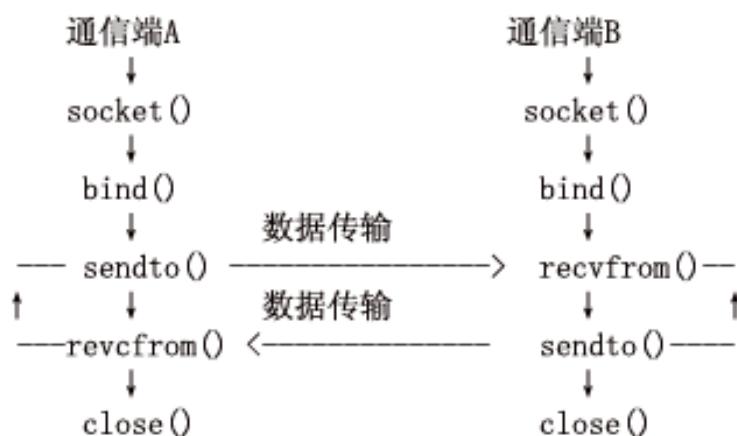
由于不需要建立连接，因此产生 socket 后就可以直接发送和接收了。当然，要接收数据报也必须绑定一个端口，否则发送方无法得知要发送到哪个端口。Sendto 和 recvfrom 两个系统调用分别用于发送和接收数据报，其调用格式为：

```
int sendto(int s, const void *msg, int len, unsigned int flags, const struct sockaddr *to, int tolen);
```

```
int recvfrom(int s, void *buf, int len, unsigned int flags, struct sockaddr *from, int fromlen);
```

其中 s 为所使用的 socket，msg 和 buf 分别为发送和接收的缓冲区指针，len 为缓冲区的长度，flags 为选项标志，此处还用不到，设为 0 即可。to 和 from 就是发送的目的地址和接收的来源地址，包含了 IP 地址和端口信息。tolen 和 fromlen 分别是 to 和 from 这两个 socket 地址结构的长度。这两个函数的返回值就是实际发送和接收的字节数，返回-1 表示出错。

使用无连接方式通信的基本过程如图所示。



### UDP 通信的基本过程

上图描述的是通信双方都绑定自己地址端口的情形，但在某些情况下，也可能有一方不用绑定地址和端口。不绑定的一方的地址和端口由内核分配。由于对方无法预先知道不绑定的一方的端口和 IP 地址(假设主机有多个端口，这些端口分配了不同的 IP 地址)，因此只能由不绑定的一方先发出数据报，对方根据收到的数据报中的来源地址就可以确定回送数据报所需要的发送地址了。显然，在这种情况下对方必须绑定地址和端口，并且通信只能由非绑定方发起。

与 read() 和 write() 相似，进程阻塞在 recvfrom() 和 sendto() 中也会发生。但与 TCP 方式不同的是，接收到一个字节数为 0 的数据报是有可能的，应用程序完全可以将 sendto() 中的 msg 设为 NULL，同时将 len 设为 0。

下面是一个基于以上原理分析的一个 udp 编程的例子(源代码位于 /opt/FriendlyARM/mini2440/examples/udptalk 目录)：

```
/*
```



追求卓越 创造精品

TO BE BEST

TO DO GREAT

广州友善之臂计算机科技有限公司

```
*      udptalk : Example for Matrix V ;说明：本程序同样适用于 mini2440
*
*      Copyright (C) 2004 capbily - friendly-arm
*      capbily@hotmail.com
*/
#include <sys/types.h>
#include <sys/socket.h>
#include <arpa/inet.h>
#include <stdio.h>

#define BUflen 255

int main(int argc, char **argv)
{
    struct sockaddr_in peeraddr, /*存放谈话对方 IP 和端口的 socket 地址*/
                      localaddr; /*本端 socket 地址*/
    int sockfd;
    char recmsg[BUflen+1];
    int socklen, n;

    if(argc!=5){
        printf("%s <dest IP address> <dest port> <source IP address> <source port>\n",
               argv[0]);
        exit(0);
    }

    sockfd = socket(AF_INET, SOCK_DGRAM, 0);
    if(sockfd<0){
        printf("socket creating err in udptalk\n");
        exit(1);
    }
    socklen = sizeof(struct sockaddr_in);
    memset(&peeraddr, 0, socklen);
    peeraddr.sin_family=AF_INET;
    peeraddr.sin_port=htons(atoi(argv[2]));
    if(inet_pton(AF_INET, argv[1], &peeraddr.sin_addr)<=0){
        printf("Wrong dest IP address!\n");
        exit(0);
    }
    memset(&localaddr, 0, socklen);
    localaddr.sin_family=AF_INET;
```



```
if/inet_pton(AF_INET, argv[3], &localaddr.sin_addr)<=0){
    printf("Wrong source IP address!\n");
    exit(0);
}
localaddr.sin_port=htons(atoi(argv[4]));
if(bind(sockfd, &localaddr, socklen)<0){
    printf("bind local address err in udptalk!\n");
    exit(2);
}

if(fgets(recmsg, BUFSIZE, stdin) == NULL) exit(0);
if(sendto(sockfd, recmsg, strlen(recmsg), 0, &peeraddr, socklen)<0){
    printf("sendto err in udptalk!\n");
    exit(3);
}

for(;;){
    /*recv&send message loop*/
    n = recvfrom(sockfd, recmsg, BUFSIZE, 0, &peeraddr, &socklen);
    if(n<0){
        printf("recvfrom err in udptalk!\n");
        exit(4);
    }else{
        /*成功接收到数据报*/
        recmsg[n]=0;
        printf("peer:%s", recmsg);
    }
    if(fgets(recmsg, BUFSIZE, stdin) == NULL) exit(0);
    if(sendto(sockfd, recmsg, strlen(recmsg), 0, &peeraddr, socklen)<0){
        printf("sendto err in udptalk!\n");
        exit(3);
    }
}
}
```

将 udptalk.c 编译好后就可以运行了，/opt/FriendlyARM/mini2440/exampls/udptalk 目录下的 Makefile 指定了两个编译目标可执行文件，一个用于在主机端的 x86-udptalk，一个是用于开发板的 arm-udptalk，运行 make 命令将把这两个程序一起编译出来。可以把 arm-udptalk 使用上面介绍的方法下载到开发板中(预装的 Linux 不含该程序)，假设主机的 IP 地址为 192.168.0.1，开发板的 IP 地址为 192.168.0.230。



追求卓越 创造精品

TO BE BEST

TO DO GREAT

广州友善之臂计算机科技有限公司

在主机的终端上输入：

```
#./x86-udptalk 192.168.0.230 2000 192.168.0.1 2000
```

在开发板上的终端输入

```
#arm-udptalk 192.168.0.1 2000 192.168.0.230 2000
```

则运行结果分别如图所示：

The screenshot shows a terminal window titled "root@capbily:/friendly-arm/examples/udptalk". The window has a menu bar with File, Edit, View, Terminal, Go, and Help. Below the menu is a toolbar with five buttons labeled root@capbily:/..., root@capbily:/f..., root@capbily:~, root@capbily:~, and root@capbily:/f... . The main terminal area displays the following text:

```
[root@capbily udptalk]# ./x86-udptalk
./x86-udptalk <dest IP address> <dest port> <source IP address>
<source port>
[root@capbily udptalk]# ./x86-udptalk 192.168.0.230 2000 192.168
.0.1 2000

peer:

peer:Hello, Capbily
Hello, SBC-2410X!
peer:
[REDACTED]
```

在主机上运行 x86-udptalk



追求卓越 创造精品

TO BE BEST

TO DO GREAT

广州友善之臂计算机科技有限公司

The screenshot shows a terminal window titled "root@capbily:~". The window contains the following text:

```
[02/Dec/2030:18:41:57 +0000] boa: server version Boa/0.94.13
[02/Dec/2030:18:41:57 +0000] boa: server built Feb 28 2004 at 2.
[02/Dec/2030:18:41:57 +0000] boa: starting server pid=34, port 0

Please press Enter to activate this console.

BusyBox v0.60.5 (2003.09.05-09:25+0000) Built-in shell (ash)
Enter 'help' for a list of built-in commands.

sh: can't access tty: job control turned off
[root@fa /]# arm-udptalk 192.168.0.1 2000 192.168.0.230 2000
Hello, Capbily
peer:
peer:
peer:Hello, SBC-2410X!
```

在开发板上运行 arm-udptalk

#### 6.2.4 数学函数库调用示例

程序源代码说明：

测试程序源代码目录	/opt/FriendlyARM/mini2440/examples/math
测试程序源代码名称	mathtest.c
测试程序可执行文件名称	mathtest

**注意：使用数学函数的关键是要包含其头文件 math.h，并且在编译的时候加入数学函数库 libm。**

程序清单：

```
#include <stdio.h>
#include <stdlib.h>
#include <math.h> ;注意：一定要包含此头文件

int main(void)
{
```



追求卓越 创造精品

TO BE BEST

TO DO GREAT

广州友善之臂计算机科技有限公司

```
double a=8.733243;  
  
printf("sqrt(%f)=%f\n", a, sqrt(a));  
  
return 0;  
}
```

相应的 Makefile 内容:

CROSS=arm-linux-

all: mathtest

#注意：该处包含了数学函数库 **libm**, 红色部分  
mathtest:

\$(CROSS)gcc -o mathtest main.c -lm

clean:

@rm -vf mathtest \*.o \*~

你可以按照上面的 hello 程序的步骤手工编译出 mathtest 可执行文件，然后下载到开发板运行它

## 6.2.5 线程编程示例

程序源代码说明:

测试程序源代码目录	/opt/FriendlyARM/mini2440/examples/pthread
测试程序源代码名称	pthread_test.c
测试程序可执行文件名称	pthread_test
注意：使用线程的关键是要包含其头文件 <b>pthread.h</b> , 并且在编译的时候加入线程库 <b>libpthread</b> 。	

程序清单:

```
#include<stddef.h>  
#include<stdio.h>  
#include<unistd.h>  
#include"pthread.h" ;注意：一定要包含此头文件  
  
void reader_function(void);  
void writer_function(void);  
char buffer;
```



```
int buffer_has_item=0;
pthread_mutex_t mutex;
main()
{
    pthread_t reader;
    pthread_mutex_init(&mutex,NULL);
    pthread_create(&reader,NULL,(void*)&reader_function,NULL);
    writer_function();
}
void writer_function(void)
{
    while(1)
    {
        pthread_mutex_lock(&mutex);
        if(buffer_has_item==0)
        {
            buffer='a';
            printf("make a new item\n");
            buffer_has_item=1;
        }
        pthread_mutex_unlock(&mutex);
    }
}
void reader_function(void)
{
    while(1)
    {
        pthread_mutex_lock(&mutex);
        if(buffer_has_item==1)
        {
            buffer='\0';
            printf("consume item\n");
            buffer_has_item=0;
        }
        pthread_mutex_unlock(&mutex);
    }
}
```

相应的 Makefile 内容:

CROSS=arm-linux-



追求卓越 创造精品

TO BE BEST

TO DO GREAT

广州友善之臂计算机科技有限公司

all: pthread

#注意：该处包含了线程库 libpthread，红色部分

pthread:

\$(CROSS)gcc -static -o pthread main.c **-lpthread**

clean:

@rm -vf pthread \*.o \*~

你可以按照上面的 hello 程序的步骤手工编译出 pthread 可执行文件，然后下载到开发板运行它

## 6.2.6 管道应用编程示例

程序源代码说明：

测试程序源代码目录	/opt/FriendlyARM/mini2440/examples/led-player
测试程序源代码名称	led-player.c
测试程序可执行文件名称	led-player
说明：见以下描述	

开机后我们可以通过网页发送命令控制开发板上的 LED 闪烁模式，其实这是进程间通信共享资源的一个典型例子，进程间通信就是 IPC(InterProcess Communication)，进程间通信的一般有：

- (1)数据传输
- (2)共享数据
- (3)通知事件
- (4)资源共享
- (5)进程控制.

Linux 支持多种 IPC 机制，信号和管道是其中的两个。关于更详细的进程间通信的介绍，一般 Linux 编程的书上都有介绍，在此我们不多说了。

通过网页来控制 LED 的闪烁模式就是通过管道机制来实现的，其中 LED 是共享资源，led-player 是一个后台程序，它启动的时候就创建了一个命名管道/tmp/ led-control(当然该管道也可以通过命令 mknod 来创建，那样程序就要改写了，有兴趣的可以自己试试)，并一直监测输入该管道的数据，根据不同的参数(模式 type 和周期 period)来改变 LED 的显示模式；leds.cgi 是一个网关程序，它接收从网页发送过来的字符形式指令 (ping 代表跑马灯模式或者乒乓模式，counter 代表计数器模式，stop 代表停止模式，slow 代表周期为 0.25m，normal 代表周期为 0.125m，fast 代表周期为 0.0625m)，并对这些指令进行赋值转换为实际数字，然后调用 echo 命令输送到管道/tmp/ led-control 以此实现对 LED 的控制，以下是各自的程序清单。

Led-player 源代码位置/opt/FriendlyARM/mini2440/examples/led-player/main.c:

程序清单：



追求卓越 创造精品

TO BE BEST

TO DO GREAT

广州友善之臂计算机科技有限公司

```
#include <stdio.h>
#include <stdlib.h>
#include <unistd.h>
#include <sys/ioctl.h>
#include <sys/types.h>
#include <sys/stat.h>
#include <fcntl.h>
#include <sys/select.h>
#include <sys/time.h>

static int led_fd;
static int type = 1;

static void push_leds(void)
{
    static unsigned step;
    unsigned led_bitmap;
    int i;

    switch(type) {
    case 0:
        if (step >= 6) {
            step = 0;
        }
        if (step < 3) {
            led_bitmap = 1 << step;
        } else {
            led_bitmap = 1 << (6 - step);
        }
        break;
    case 1:
        if (step > 255) {
            step = 0;
        }
        led_bitmap = step;
        break;
    default:
        led_bitmap = 0;
    }
    step++;
    for (i = 0; i < 4; i++) {
```



追求卓越 创造精品

TO BE BEST

TO DO GREAT

广州友善之臂计算机科技有限公司

```
ioctl(led_fd, led_bitmap & 1, i);
led_bitmap >>= 1;
}

}

int main(void)
{
    int led_control_pipe;
    int null_writer_fd; // for read endpoint not blocking when control process exit

    double period = 0.5;

    led_fd = open("/dev/leds", 0);
    if (led_fd < 0) {
        perror("open device leds");
        exit(1);
    }
    unlink("/tmp/led-control");
    mknod("/tmp/led-control", 0666);

    led_control_pipe = open("/tmp/led-control", O_RDONLY | O_NONBLOCK);
    if (led_control_pipe < 0) {
        perror("open control pipe for read");
        exit(1);
    }
    null_writer_fd = open("/tmp/led-control", O_WRONLY | O_NONBLOCK);
    if (null_writer_fd < 0) {
        perror("open control pipe for write");
        exit(1);
    }

    for (;;) {
        fd_set rds;
        struct timeval step;
        int ret;

        FD_ZERO(&rds);
        FD_SET(led_control_pipe, &rds);
        step.tv_sec = period;
        step.tv_usec = (period - step.tv_sec) * 1000000L;
```



追求卓越 创造精品

TO BE BEST

TO DO GREAT

广州友善之臂计算机科技有限公司

```
ret = select(led_control_pipe + 1, &rds, NULL, NULL, &step);
if (ret < 0) {
    perror("select");
    exit(1);
}
if (ret == 0) {
    push_leds();
} else if (FD_ISSET(led_control_pipe, &rds)) {
    static char buffer[200];
    for (;;) {
        char c;
        int len = strlen(buffer);
        if (len >= sizeof buffer - 1) {
            memset(buffer, 0, sizeof buffer);
            break;
        }
        if (read(led_control_pipe, &c, 1) != 1) {
            break;
        }
        if (c == 'r') {
            continue;
        }
        if (c == '\n') {
            int tmp_type;
            double tmp_period;
            if (sscanf(buffer, "%d%lf", &tmp_type, &tmp_period) == 2) {
                type = tmp_type;
                period = tmp_period;
            }
            fprintf(stderr, "type is %d, period is %lf\n", type, period);
            memset(buffer, 0, sizeof buffer);
            break;
        }
        buffer[len] = c;
    }
}
close(led_fd);
return 0;
}
```



追求卓越 创造精品

TO BE BEST

TO DO GREAT

广州友善之臂计算机科技有限公司

使用 make 指令可以直接编译出 led-player 可执行文件，它被作为一个服务器放置在开发板的/sbin 目录中。

Leds.cgi 网关程序源代码(该程序在开发板上的位置：/www/leds.cgi)，可见该网关程序其实就是一个 shell 脚本，它被网页 leds.html 调用为一个执行“action”：

```
#!/bin/sh

type=0
period=1

case $QUERY_STRING in
    *ping*)
        type=0
        ;;
    *counter*)
        type=1
        ;;
    *stop*)
        type=2
        ;;
esac

case $QUERY_STRING in
    *slow*)
        period=0.25
        ;;
    *normal*)
        period=0.125
        ;;
    *fast*)
        period=0.0625
        ;;
esac

/bin/echo $type $period > /tmp/led-control

echo "Content-type: text/html; charset=gb2312"
echo
/bin/cat led-result.template
```



exit 0

追求卓越 创造精品

TO BE BEST

TO DO GREAT

广州友善之臂计算机科技有限公司

## 6.2.7 基于 C++的 Hello,World

程序源代码说明：

测试程序源代码目录	/opt/FriendlyARM/mini2440/examples/c++
测试程序源代码名称	cplus.c++
测试程序可执行文件名称	cplus
说明：无	

程序清单及注释：

程序清单：

```
#include <iostream>
#include <cstring>
using namespace std;

class String
{
private:
    char *str;
public:
    String(char *s)
    {
        int lenght=strlen(s);
        str = new char[lenght+1];
        strcpy(str, s);
    }
    ~String()
    {
        cout << "Deleting str.\n";
        delete[] str;
    }
    void display()
    {
        cout << str << endl;
    }
};

int main(void)
{
```



```
String s1="I like FriendlyARM.";
cout << "s1=";
s1.display();
return 0;
double num, ans;

cout << "Enter num:";
}
```

你可以按照上面的 hello 程序的步骤手工编译出 cplus 可执行文件，然后下载到开发板运行它

## 6.3 最简单的嵌入式 Linux 驱动程序模块

6.1一节我们介绍了一个简单的Linux程序Hello,World,它是运行于用户态的应用程序，现在我们再介绍一个运行于内核态的 Hello, World 程序，它其实是一个最简单的驱动程序模块。

### 6.3.1 Hello,Module 源代码

程序源代码说明：

源代码所在目录	/opt/FriendlyARM/mini2440/kernel-2.6.13/drivers/char
源代码文件名称	qq2440_hello_sample.c
该驱动的主设备号	无
设备名	无
测试程序源代码目录	无
测试程序名称	无
测试程序可执行文件名称	无
说明： mini2440 使用与 QQ2440 板相同的该示例程序。	

Hello,Module 的源代码位于/opt/FriendlyARM/mini2440/kernel-2.6.13/drivers/char 目录，文件名为 qq2440\_hello\_module.c，源代码内容如下：



```
#include <linux/kernel.h>
#include <linux/module.h>

MODULE_LICENSE("GPL");

static int __init qq2440_hello_module_init(void)
{
    printk("Hello, QQ2440 module is installed !\n");
    return 0;
}

static void __exit qq2440_hello_module_cleanup(void)
{
    printk("Good-bye, QQ2440 module was removed!\n");
}

module_init(qq2440_hello_module_init);
module_exit(qq2440_hello_module_cleanup);
```

### 6.3.2 把 Hello,Module 加入内核代码树，并编译

一般编译 2.6 版本的驱动模块需要把驱动代码加入内核代码树，并做相应的配置，如下步骤(注意：实际上以下步骤均已经做好，你只需要打开检查一下直接编译就可以了)：

Step1：编辑配置文件 Kconfig，加入驱动选项，使之在 make menuconfig 的时候出现  
打开 kernel-2.6.13/drivers/char/Kconfig 文件，添加如图所示：

```

root@localhost:/opt/FriendlyARM/QQ2440/kernel-2.6.13#
File Edit View Terminal Go Help
depends on ARCH_S3C2410
help
RTC (Realtime Clock) driver for the clock inbuilt into the
Samsung S3C2410. This can provide periodic interrupt rates
from 1Hz to 64Hz for user programs, and wakeup from Alarm.

config SBC2440_LEDS
    tristate "SBC2440 LEDs Driver"
    depends on ARCH_S3C2410
    help
        SBC2440 User leds.

config QQ2440_HELLO_MODULE
    tristate "QQ2440 module sample"
    depends on ARCH_S3C2410
    help
        QQ2440 module sample.

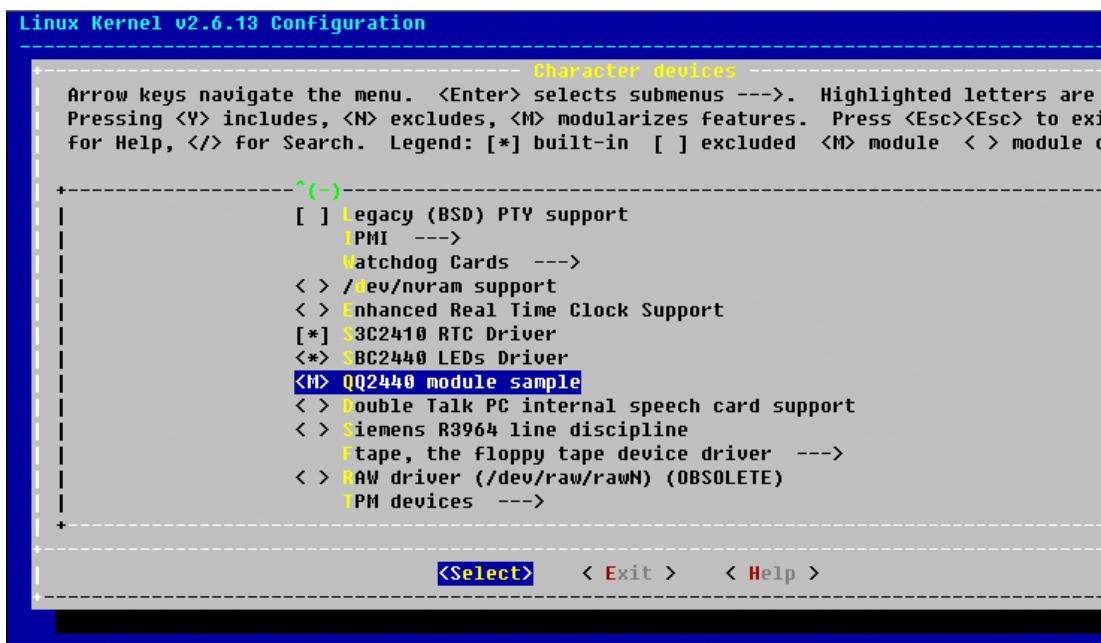
config RTC_VR41XX
    tristate "NEC VR4100 series Real Time Clock Support"
    depends on CPU_VR41XX

config COBALT_LCD

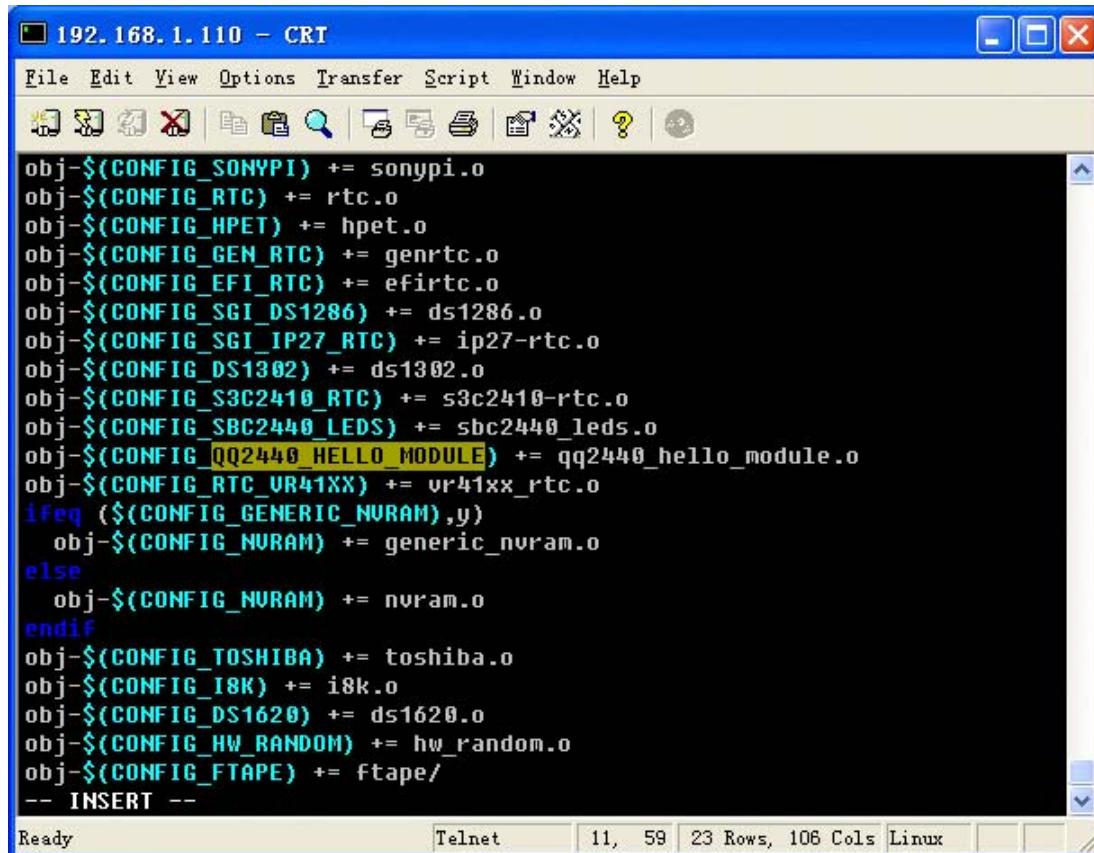
```

787,0-1      77%

保存退出，这时在 kernel-2.6.13 目录位置运行一下 make menuconfig 就可以在 Device Drivers → Character devices 菜单中看到刚才所添加的选项了，按下空格键将会选择为<M>，此意为要把该选项编译为模块方式；再按下空格会变为<\*>，意为要把该选项编译到内核中，在此我们选择<M>，如图：



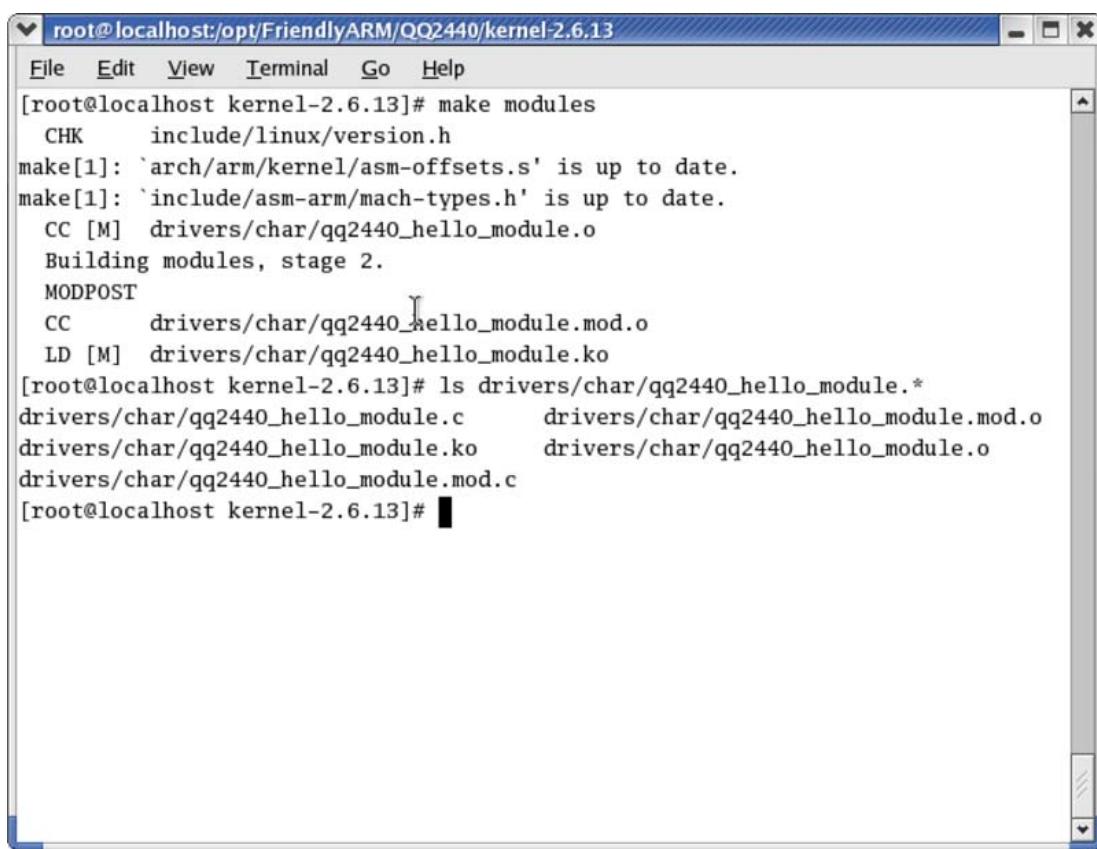
Step2：通过上一步，我们虽然可以在配置内核的时候进行选择，但实际上此时执行编译内核还是不能把 qq2440\_hello\_module.c 编译进去的，还需要在 Makefile 中把内核配置选项和真正的源代码联系起来，打开 kernel-2.6.13/drivers/char/Makefile，如图添加并保存退出：



```
obj-$(CONFIG_SONYPI) += sonypi.o
obj-$(CONFIG_RTC) += rtc.o
obj-$(CONFIG_HPET) += hpet.o
obj-$(CONFIG_GEN_RTC) += genrtc.o
obj-$(CONFIG_EFI_RTC) += efirtc.o
obj-$(CONFIG_SGI_DS1286) += ds1286.o
obj-$(CONFIG_SGI_IP27_RTC) += ip27-rtc.o
obj-$(CONFIG_DS1302) += ds1302.o
obj-$(CONFIG_S3C2410_RTC) += s3c2410-rtc.o
obj-$(CONFIG_SBC2440_LEDS) += sbc2440_leds.o
obj-$(CONFIG_QQ2440_HELLO_MODULE) += qq2440_hello_module.o
obj-$(CONFIG_RTC_VR41XX) += vr41xx_rtc.o
ifeq ($(CONFIG_GENERIC_NVRAM),y)
    obj-$(CONFIG_NVRAM) += generic_nvram.o
else
    obj-$(CONFIG_NVRAM) += nvram.o
endif
obj-$(CONFIG_TOSHIBA) += toshiba.o
obj-$(CONFIG_I8K) += i8k.o
obj-$(CONFIG_DS1620) += ds1620.o
obj-$(CONFIG_HW_RANDOM) += hw_random.o
obj-$(CONFIG_FTAPE) += ftape/
-- INSERT --
```

Step3：这时回到 kernel-2.6.13 源代码根目录位置，执行 make modules，就可以生成我们所需要的内核模块文件 qq2440\_hello\_module.ko 了，如图：

至此，我们已经完成了模块驱动的编译。



The screenshot shows a terminal window titled "root@localhost:opt/FriendlyARM/QQ2440/kernel-2.6.13". The window contains the following text:

```
[root@localhost kernel-2.6.13]# make modules
CHK      include/linux/version.h
make[1]: `arch/arm/kernel/asm-offsets.s' is up to date.
make[1]: `include/arm/arm/mach-types.h' is up to date.
CC [M]  drivers/char/qq2440_hello_module.o
Building modules, stage 2.
MODPOST
CC      drivers/char/qq2440_hello_module.mod.o
LD [M]  drivers/char/qq2440_hello_module.ko
[root@localhost kernel-2.6.13]# ls drivers/char/qq2440_hello_module.*
drivers/char/qq2440_hello_module.c      drivers/char/qq2440_hello_module.mod.o
drivers/char/qq2440_hello_module.ko      drivers/char/qq2440_hello_module.o
drivers/char/qq2440_hello_module.mod.c
[root@localhost kernel-2.6.13]#
```

### 6.3.3 把 Hello, Module 下载到开发板并安装使用

最简单的方法莫过于使用 **ftp** 把编译好的驱动模块传送到板子里，当然你也可以使用 6.1 一节中介绍的其他方法。假定我们已经把 **qq2440\_hello\_module.ko** 放到了板子的 **/home/plg** 目录下，现在执行

```
#insmod qq2440_hello_module.ko
```

可以看到该模块已经被装载了；

再执行，可以看到该模块被卸载

```
#rmmod qq2440_hello_module.ko
```

整个过程如下图：

```

[root@FriendlyARM plg]# ls
qq2440_hello_module.ko
[root@FriendlyARM plg]# insmod qq2440_hello_module.ko
Hello, QQ2440 module is installed !
[root@FriendlyARM plg]# rmmod qq2440_hello_module
Good-bye, QQ2440 module was removed!
[root@FriendlyARM plg]#

```

已连接 0:00:31 ANSIW | 115200 8-N-1 | SCROLL | CAPS | NUM | 捕 | 打印 |

## 6.4 简易 Linux 驱动程序示例

在上一小节，我们介绍了最简单的 HelloModule 驱动程序模块，它只是从串口输出一些信息，并未对应板上的硬件进行操作，下面的几个例子都是和硬件密切相关的实际驱动，在嵌入式 Linux 系统中，大部分的硬件都需要类似的驱动才能操作，比如触摸屏、网卡、音频等，在这里我们介绍的是一些简单典型的例子，实际上复杂的驱动都有参考代码，不必从头写驱动。

在本节中，我们不介绍驱动程序理论概念之类的内容，那些在网上或者书籍中都有比较系统的描述。

### 6.4.1 LED 驱动程序

程序源代码说明：

驱动源代码所在目录	/opt/FriendlyARM/mini2440/kernel-2.6.13/drivers/char
驱动程序名称	qq2440_leds.c
该驱动的主设备号	231



追求卓越 创造精品

TO BE BEST

TO DO GREAT

广州友善之臂计算机科技有限公司

设备名	/dev/leds
测试程序源代码目录	/opt/FriendlyARM/mini2440/examples/led
测试程序名称	led.c
测试程序可执行文件名称	led

说明：LED 驱动已经被编译到缺省内核中，因此不能再使用 insmod 方式加载。  
另，mini2440 使用的 LED 资源与 QQ2440 相同，因此驱动是完全一样的。

要写实际的驱动，就必须了解相关的硬件资源，比如用到的寄存器，物理地址，中断等，在这里，LED 是一个很简单的例子，它用到了如下硬件资源。

mini2440 开发板上所用到的 4 个 LED 的硬件资源

LED	对应的 IO 寄存器名称	对应的 CPU 引脚
LED1	GPB5	K2
LED2	GPB6	L5
LED3	GPB7	K7
LED4	GPB8	K5

要操作所用到的 IO 口，就要设置它们所用到的寄存器，我们需要调用一些现成的函数或者宏，例如：[s3c2410\\_gpio\\_cfgpin](#)

为什么是 S3C2410 的呢？因为三星出品的 S3C2440 芯片所用的寄存器名称以及资源分配大部分和 S3C2410 是相同的，在目前各个版本的 Linux 系统中，也大都采用了相同的函数定义和宏定义。

它们从哪里定义？细心的用户或许很快就想到它们和体系结构有关，因此你可以在 kernel-2.6.13/include/asm/arch-s3c2410/hardware.h 文件中找到该函数的定义，关于该函数的实际实现则可以在 kernel-2.6.13/arch/arm/mach-s3c2410/gpio.c 中找到，它的内容如下：

```
void s3c2410_gpio_cfgpin(unsigned int pin, unsigned int function)
{
    void __iomem *base = S3C2410_GPIO_BASE(pin);
    unsigned long mask;
    unsigned long con;
    unsigned long flags;

    if (pin < S3C2410_GPIO_BANKB) {
        mask = 1 << S3C2410_GPIO_OFFSET(pin);
    } else {
        mask = 3 << S3C2410_GPIO_OFFSET(pin)*2;
    }

    local_irq_save(flags);

    con = __raw_readl(base + 0x00);
```



```
con &= ~mask;
con |= function;

__raw_writel(con, base + 0x00);
```

```
local_irq_restore(flags);
}
```

实际上，我们并不需要关心这些，写驱动时只要会使用他们就可以了，除非你所使用的 CPU 体系平台尚没有被 Linux 所支持，因为大部分常见的嵌入式平台都已经有了很完善的类似定义，你不需要自己去编写。

在下面的驱动程序清单中，你可以看到 `s3c2410_gpio_cfgpin` 被调用的情况。除此之外，你还需要调用一些和设备驱动密切相关的 basic 函数，如注册设备 `register_chrdev`，创建设备 `devfs_mk_cdev`，填写驱动函数结构 `file_operations`，以及像 `HelloModule` 中那样的 `module_init` 和 `module_exit` 函数等。

有些函数并不是必须的，随着你对 Linux 驱动开发的进一步了解和阅读更多的代码，你自然明白。下面是我们为 LED 编写的驱动代码清单，

驱动程序清单：

```
#include <linux/config.h>
#include <linux/module.h>
#include <linux/kernel.h>
#include <linux/fs.h>
#include <linux/init.h>
#include <linux/devfs_fs_kernel.h>
#include <linux/miscdevice.h>
#include <linux/delay.h>
#include <asm/irq.h>
#include <asm/arch/regs-gpio.h>
#include <asm/hardware.h>

#define DEVICE_NAME    "leds"
#define LED_MAJOR 231

static unsigned long led_table [] = {
    S3C2410_GPB5,
    S3C2410_GPB6,
    S3C2410_GPB7,
    S3C2410_GPB8,
};

static unsigned int led_cfg_table [] = {
    S3C2410_GPB5_OUTP,
```



```
S3C2410_GPB6_OUTP,  
S3C2410_GPB7_OUTP,  
S3C2410_GPB8_OUTP,  
};
```

```
static int qq2440_leds_ioctl(  
    struct inode *inode,  
    struct file *file,  
    unsigned int cmd,  
    unsigned long arg)  
{  
    switch(cmd) {  
    case 0:  
    case 1:  
        if (arg > 4) {  
            return -EINVAL;  
        }  
        s3c2410_gpio_setpin(led_table[arg], !cmd);  
        return 0;  
    default:  
        return -EINVAL;  
    }  
}
```

```
static struct file_operations qq2440_leds_fops = {  
    .owner  = THIS_MODULE,  
    .ioctl   = qq2440_leds_ioctl,  
};
```

```
static int __init qq2440_leds_init(void)  
{  
    int ret;  
    int i;  
  
    ret = register_chrdev(LED_MAJOR, DEVICE_NAME, &qq2440_leds_fops);  
    if (ret < 0) {  
        printk(DEVICE_NAME " can't register major number\n");  
        return ret;  
    }
```

```
    devfs_mk_cdev(MKDEV(LED_MAJOR, 0), S_IFCHR | S_IRUSR | S_IWUSR | S_IRGRP,
```



追求卓越 创造精品

TO BE BEST

TO DO GREAT

广州友善之臂计算机科技有限公司

DEVICE\_NAME);

```
for (i = 0; i < 4; i++) {  
    s3c2410_gpio_cfgpin(led_table[i], led_cfg_table[i]);  
    s3c2410_gpio_setpin(led_table[i], 1);  
}  
  
printk(DEVICE_NAME " initialized\n");  
return 0;  
}  
  
static void __exit qq2440_leds_exit(void)  
{  
    devfs_remove(DEVICE_NAME);  
    unregister_chrdev(LED_MAJOR, DEVICE_NAME);  
}  
  
module_init(qq2440_leds_init);  
module_exit(qq2440_leds_exit);
```

## 6.4.2 按键驱动程序

程序源代码说明：

驱动源代码所在目录	/opt/FriendlyARM/mini2440/kernel-2.6.13/drivers/char
驱动程序名称	mini2440_buttons.c
该驱动的主设备号	232
设备名	/dev/buttons
测试程序源代码目录	/opt/FriendlyARM/mini2440/examples/buttons
测试程序源代码名称	buttons_test.c
测试程序可执行文件名称	buttons

说明：按键驱动已经被编译到缺省内核中，因此不能再使用 insmod 方式加载。

mini2440 所用到的按键资源如下：

按键	对应的 IO 寄存器名称	对应的中断
K1	PGP0	EINT8
K2	PGP3	EINT11
K3	PGP5	EINT13
K4	PGP6	EINT14
K5	PGP7	EINT15



追求卓越 创造精品

TO BE BEST

TO DO GREAT

广州友善之臂计算机科技有限公司

K6

PGP11

EINT19

按键驱动源代码清单及注释：

```
#include <linux/module.h>
#include <linux/kernel.h>
#include <linux/fs.h>
#include <linux/init.h>
#include <linux/delay.h>
#include <linux/poll.h>
#include <asm/irq.h>
#include <linux/interrupt.h>
#include <asm/uaccess.h>
#include <asm/arch/regs-gpio.h>
#include <asm/hardware.h>
```

```
#define DEVICE_NAME      "buttons"    /* 加载模式后，执行” cat /proc/devices ” 命令看到的设备名称 */
#define BUTTON_MAJOR      232          /* 主设备号 */
```

```
struct button_irq_desc {
    int irq;
    int pin;
    int pin_setting;
    int number;
    char *name;
};
```

```
/* 用来指定按键所用的外部中断引脚及中断触发方式，名字 */
```

```
static struct button_irq_desc button_irqs [] = {
    {IRQ_EINT8, S3C2410_GPG0, S3C2410_GPG0_EINT8, 0, "KEY1"}, /* K1 */
    {IRQ_EINT11, S3C2410_GPG3, S3C2410_GPG3_EINT11, 1, "KEY2"}, /* K2 */
    {IRQ_EINT13, S3C2410_GPG5, S3C2410_GPG5_EINT13, 2, "KEY3"}, /* K3 */
}
    {IRQ_EINT14, S3C2410_GPG6, S3C2410_GPG6_EINT14, 3, "KEY4"}, /* K3 */
}
    {IRQ_EINT15, S3C2410_GPG7, S3C2410_GPG7_EINT15, 4, "KEY5"}, /* K3 */
}
    {IRQ_EINT19, S3C2410_GPG11, S3C2410_GPG11_EINT19, 5, "KEY6"}, /* K4 */
};
```



```
/* 按键被按下的次数(准确地说, 是发生中断的次数) */
static volatile int key_values [] = {0, 0, 0, 0, 0, 0, 0};

/* 等待队列:
 * 当没有按键被按下时, 如果有进程调用 mini2440_buttons_read 函数,
 * 它将休眠
 */
static DECLARE_WAIT_QUEUE_HEAD(button_waitq);

/* 中断事件标志, 中断服务程序将它置 1, mini2440_buttons_read 将它清 0 */
static volatile int ev_press = 0;

static irqreturn_t buttons_interrupt(int irq, void *dev_id)
{
    struct button_irq_desc *button_irqs = (struct button_irq_desc *)dev_id;
    int up = s3c2410_gpio_getpin(button_irqs->pin);

    if (up)
        key_values[button_irqs->number] = (button_irqs->number + 1) + 0x80;
    else
        key_values[button_irqs->number] = (button_irqs->number + 1);

    ev_press = 1;                      /* 表示中断发生了 */
    wake_up_interruptible(&button_waitq); /* 唤醒休眠的进程 */

    return IRQ_RETVAL(IRQ_HANDLED);
}

/* 应用程序对设备文件/dev/buttons 执行 open(...)时,
 * 就会调用 mini2440_buttons_open 函数
 */
static int mini2440_buttons_open(struct inode *inode, struct file *file)
{
    int i;
    int err;

    for (i = 0; i < sizeof(button_irqs)/sizeof(button_irqs[0]); i++) {
        // 注册中断处理函数
        s3c2410_gpio_cfgpin(button_irqs[i].pin,button_irqs[i].pin_setting);
    }
}
```



```
err = request_irq(button_irqs[i].irq, buttons_interrupt, NULL,
                   button_irqs[i].name, (void *)&button_irqs[i]);
    set_irq_type(button_irqs[i].irq, IRQT_BOTHEDGE);
    if (err)
        break;
}

if (err) {
    // 释放已经注册的中断
    i--;
    for (; i >= 0; i--) {
        disable_irq(button_irqs[i].irq);
        free_irq(button_irqs[i].irq, (void *)&button_irqs[i]);
    }
    return -EBUSY;
}

return 0;
}

/* 应用程序对设备文件/dev/buttons 执行 close(...)时,
 * 就会调用 mini2440_buttons_close 函数
 */
static int mini2440_buttons_close(struct inode *inode, struct file *file)
{
    int i;

    for (i = 0; i < sizeof(button_irqs)/sizeof(button_irqs[0]); i++) {
        // 释放已经注册的中断
        disable_irq(button_irqs[i].irq);
        free_irq(button_irqs[i].irq, (void *)&button_irqs[i]);
    }

    return 0;
}

/* 应用程序对设备文件/dev/buttons 执行 read(...)时,
 * 就会调用 mini2440_buttons_read 函数
 */
```



追求卓越 创造精品

TO BE BEST

TO DO GREAT

广州友善之臂计算机科技有限公司

```
static int mini2440_buttons_read(struct file *filp, char __user *buff,
                                  size_t count, loff_t *offp)
{
    unsigned long err;

    if (!ev_press) {
        if (filp->f_flags & O_NONBLOCK)
            return -EAGAIN;
        else
            /* 如果 ev_press 等于 0, 休眠 */
            wait_event_interruptible(button_waitq, ev_press);
    }

    /* 执行到这里时, ev_press 等于 1, 将它清 0 */
    ev_press = 0;

    /* 将按键状态复制给用户, 并清 0 */
    err = copy_to_user(buff, (const void *)key_values, min(sizeof(key_values), count));
    memset((void *)key_values, 0, sizeof(key_values));

    return err ? -EFAULT : min(sizeof(key_values), count);
}

/******************
 * 当用户程序调用 select 函数时, 本函数被调用
 * 如果有按键数据, 则 select 函数会立刻返回
 * 如果没有按键数据, 本函数使用 poll_wait 等待
******************/

static unsigned int mini2440_buttons_poll(
    struct file *file,
    struct poll_table_struct *wait)
{
    unsigned int mask = 0;
    poll_wait(file, &button_waitq, wait);
    if (ev_press)
        mask |= POLLIN | POLLRDNORM;
    return mask;
}
```



```
/* 这个结构是字符设备驱动程序的核心
 * 当应用程序操作设备文件时所调用的 open、read、write 等函数,
 * 最终会调用这个结构中的对应函数
 */

static struct file_operations mini2440_buttons_fops = {
    .owner      = THIS_MODULE,      /* 这是一个宏, 指向编译模块时自动创建的
__this_module 变量 */
    .open       = mini2440_buttons_open,
    .release   = mini2440_buttons_close,
    .read       = mini2440_buttons_read,
    .poll       = mini2440_buttons_poll,
};

/*
 * 执行 “insmod mini2440_buttons.ko” 命令时就会调用这个函数
 */
static int __init mini2440_buttons_init(void)
{
    int ret;

    /* 注册字符设备驱动程序
     * 参数为主设备号、设备名字、file_operations 结构;
     * 这样, 主设备号就和具体的 file_operations 结构联系起来了,
     * 操作主设备为 BUTTON_MAJOR 的设备文件时, 就会调用
mini2440_buttons_fops 中的相关成员函数
     * BUTTON_MAJOR 可以设为 0, 表示由内核自动分配主设备号
     */
    ret      = register_chrdev(BUTTON_MAJOR,           DEVICE_NAME,
&mini2440_buttons_fops);
    if (ret < 0) {
        printk(DEVICE_NAME " can't register major number\n");
        return ret;
    }
    devfs_mk_cdev(MKDEV(BUTTON_MAJOR, 0), S_IFCHR | S_IRUSR | S_IWUSR |
S_IRGRP, DEVICE_NAME);

    printk(DEVICE_NAME " initialized\n");
    return 0;
}

/*
```



```
* 执行" rmmod mini2440_buttons.ko" 命令时就会调用这个函数
*/
static void __exit mini2440_buttons_exit(void)
{
    /* 卸载驱动程序 */
    devfs_remove(DEVICE_NAME);
    unregister_chrdev(BUTTON_MAJOR, DEVICE_NAME);
}

/* 这两行指定驱动程序的初始化函数和卸载函数 */
module_init(mini2440_buttons_init);
module_exit(mini2440_buttons_exit);

/* 描述驱动程序的一些信息，不是必须的 */
MODULE_AUTHOR("http://www.arm9.net");           // 驱动程序的作者
MODULE_DESCRIPTION("S3C2410/S3C2440 BUTTON Driver"); // 一些描述信息
MODULE_LICENSE("GPL");                           // 遵循的协议
```

## 6.5 嵌入式 Linux 程序移植实例

在此我们介绍几个常见的嵌入式 Linux 程序移植详细过程以作引导，大家可以到网上搜索下载和移植更多自己喜欢的程序，但移植过程往往一波三折，这其中不免会让你遭受挫折，但也充满乐趣，这或许就是 Linux 的魅力所在吧。

提示：本小节的移植文档原基于 QQ2440 开发板编写，故文中提到的目录是 QQ2440 相关的，用户可自行修改为 mini2440 目录测试，在此不再一一重新修改。

### 6.5.1 mp3 播放器 madplay 移植过程详解

**声明：本文系友善之臂原创文章，版权归友善之臂所有。**

我们的缺省 Linux 系统开机时都会播放一首 mp3，这其中所用的播放器就是 madplay，自从 2004 年我们首次在 2410 Linux 系统中采用了该命令模式的播放器，就一直被后来众多的开发板厂商所借鉴采用，下面我们介绍一下该播放器的详细移植过程。

说明：本文中所使用的交叉编译器版本为 **arm-linux-gcc-3.4.1**，移植 madplay 所需的所有文件均已放在光盘的 **linux-apps** 目录相应的文件夹中，带有编译脚本的整个压缩包为光盘中的 **/linux/porting/sample/madplay.tgz** 文件，为了还原一个真实的移植过程，本文从网络上搜索源代码开始。

目前 madplay 的官方网站是 <http://www.underbit.com/products/mad/>，透过该网站的介绍可以得知，它还需要 libmad 和 libid3tag 两个库，从该网站找到下载连接



追求卓越 创造精品

TO BE BEST

TO DO GREAT

广州友善之臂计算机科技有限公司

[http://sourceforge.net/project/showfiles.php?group\\_id=12349](http://sourceforge.net/project/showfiles.php?group_id=12349)

这样我们就得到了移植 madplay 所需要的关键的三个文件：

- madplay-0.15.2b.tar.gz
- libmad-0.15.1b.tar.gz
- libid3tag-0.15.1b.tar.gz

它还会用到其他文件吗？谁知道呢！一般都会遇到一些小麻烦，让我们继续吧。

一般移植嵌入式应用软件的步骤是先在 PC 上配置编译该软件并运行，以了解一下该软件的用途和使用方法等，现在我们就先在 PC 上开始。

### (1)建立工作目录，拷贝源代码包

在/opt/FriendlyARM/QQ2440 目录下建立 madplay 目录，并以此为工作目录，并在该目录中建立以下子目录，以存放不同的文件：

```
#cd /opt/FriendlyARM/QQ2440  
#mkdir madplay  
#cd madplay  
#mkdir tarball src-x86 src-arm target-x86 target-arm
```

目录说明：

tarball 目录用来存放所有的源代码包

src-x86 目录用来存放 X86 版本的所有源代码文件

src-arm 目录用来存放 ARM 版本的所有源代码文件

target-x86 目录是 X86 版本的安装目录

target-arm 目录是 ARM 版本的安装目录

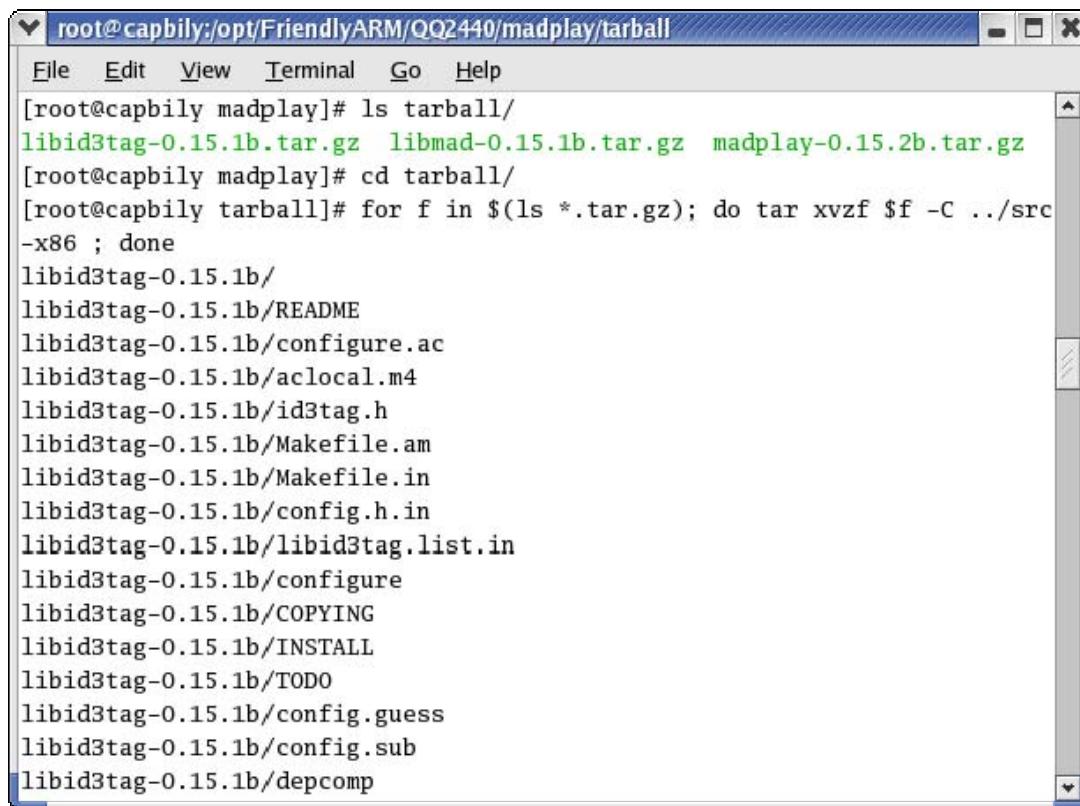
接下来把从网上下载到的源代码包放入 tarball 目录：

The screenshot shows a terminal window titled 'root@capbily:/opt/FriendlyARM/QQ2440/madplay'. The window contains the following command and its output:  
File Edit View Terminal Go Help  
[root@capbily madplay]# ls tarball/  
libid3tag-0.15.1b.tar.gz libmad-0.15.1b.tar.gz madplay-0.15.2b.tar.gz  
[root@capbily madplay]#

### (2)解压源代码包

```
#cd tarball  
#for f in $(ls *.tar.gz); do tar xvzf $f -C ../src-86 ; done
```

如图：



```
root@capbily:/opt/FriendlyARM/QQ2440/madplay/tarball
File Edit View Terminal Go Help
[root@capbily madplay]# ls tarball/
libid3tag-0.15.1b.tar.gz libmad-0.15.1b.tar.gz madplay-0.15.2b.tar.gz
[root@capbily madplay]# cd tarball/
[root@capbily tarball]# for f in $(ls *.tar.gz); do tar xvzf $f -C ../../src-x86 ; done
libid3tag-0.15.1b/
libid3tag-0.15.1b/README
libid3tag-0.15.1b/configure.ac
libid3tag-0.15.1b/aclocal.m4
libid3tag-0.15.1b/id3tag.h
libid3tag-0.15.1b/Makefile.am
libid3tag-0.15.1b/Makefile.in
libid3tag-0.15.1b/config.h.in
libid3tag-0.15.1b/libid3tag.list.in
libid3tag-0.15.1b/configure
libid3tag-0.15.1b/COPYING
libid3tag-0.15.1b/INSTALL
libid3tag-0.15.1b/TODO
libid3tag-0.15.1b/config.guess
libid3tag-0.15.1b/config.sub
libid3tag-0.15.1b/depcomp
```

### (3) 编译 madplay 所依赖的库文件

a) libid3tag

```
#cd ../../src-x86/libid3tag-0.15.1b
./configure --prefix=/opt/FriendlyARM/QQ2440/madplay/target-x86
make
make install
```

b) libmad

```
#cd ../../libmad-0.15.1b
./configure --prefix=/opt/FriendlyARM/QQ2440/madplay/target-x86
make
make install
```

以上过程完毕，将在 target-x86 目录出现编译 madplay 所依赖的库文件和头文件。



追求卓越 创造精品

TO BE BEST

TO DO GREAT

广州友善之臂计算机科技有限公司

```
root@capbily:/opt/FriendlyARM/QQ2440/madplay/target-x86
File Edit View Terminal Go Help
x86/include/mad.h
make[3]: Leaving directory '/opt/FriendlyARM/QQ2440/madplay/src-x86/libmad-0.15.1b'
make[2]: Leaving directory '/opt/FriendlyARM/QQ2440/madplay/src-x86/libmad-0.15.1b'
make[1]: Leaving directory '/opt/FriendlyARM/QQ2440/madplay/src-x86/libmad-0.15.1b'
[root@capbily libmad-0.15.1b]# cd ..
[root@capbily src-x86]# cd ..
[root@capbily madplay]# ls
src-arm src-x86 tarball target-arm target-x86
[root@capbily madplay]# cd target-x86/
[root@capbily target-x86]# ls
include lib
[root@capbily target-x86]# ls include/
id3tag.h mad.h
[root@capbily target-x86]# ls lib/
libid3tag.a libid3tag.so.0 libmad.la libmad.so.0.2.1
libid3tag.la libid3tag.so.0.3.0 libmad.so
libid3tag.so libmad.a libmad.so.0
[root@capbily target-x86]#
```

#### (4) 编译安装 madplay

如果我们还是像刚才那样配置编译选项:

```
#cd madplay-0.15.2b
```

```
./configure --prefix=/opt/FriendlyARM/QQ2440/madplay/target-x86
```

将会出现如图错误:



追求卓越 创造精品

TO BE BEST

TO DO GREAT

广州友善之臂计算机科技有限公司

```
root@capbily:/opt/FriendlyARM/QQ2440/madplay/src-x86/madplay-0.15.2b
File Edit View Terminal Go Help
checking for sys/types.h... (cached) yes
checking fcntl.h usability... yes
checking fcntl.h presence... yes
checking for fcntl.h... yes
checking errno.h usability... yes
checking errno.h presence... yes
checking for errno.h... yes
checking sys/soundcard.h usability... yes
checking sys/soundcard.h presence... yes
checking for sys/soundcard.h... yes
checking machine/soundcard.h usability... no
checking machine/soundcard.h presence... no
checking for machine/soundcard.h... no
checking mad.h usability... no
checking mad.h presence... no
checking for mad.h... no
configure: error: mad.h was not found
*** You must first install libmad before you can build this package.
*** If libmad is already installed, you may need to use the CPPFLAGS
*** environment variable to specify its installed location, e.g. -I<dir>.
[root@capbily madplay-0.15.2b]#
```

提示我们在配置 madplay 之前要先安装 libmad，因为我们之前已经编译并安装了依赖库，根据提示我们要设置 CPPFLAGS 环境变量，采用如下参数重新配置：

```
./configure --prefix=/opt/FriendlyARM/QQ2440/madplay/target-x86 CPPFLAGS=-I/opt/FriendlyARM/QQ2440/madplay/target-x86/include
```

运行结果如图所示：



追求卓越 创造精品

TO BE BEST

TO DO GREAT

广州友善之臂计算机科技有限公司

```
root@capbily:/opt/FriendlyARM/QQ2440/madplay/src-x86/madplay-0.15.2b
File Edit View Terminal Go Help
checking sys/soundcard.h presence... yes
checking for sys/soundcard.h... yes
checking machine/soundcard.h usability... no
checking machine/soundcard.h presence... no
checking for machine/soundcard.h... no
checking mad.h usability... yes
checking mad.h presence... yes
checking for mad.h... yes
checking id3tag.h usability... yes
checking id3tag.h presence... yes
checking for id3tag.h... yes
checking for an ANSI C-conforming const... (cached) yes
checking for inline... (cached) inline
checking whether byte ordering is bigendian... no
checking for mad_decoder_run in -lmad... no
configure: error: libmad was not found
*** You must first install libmad before you can build this package.
*** If libmad is already installed, you may need to use the LDFLAGS
*** environment variable to specify its installed location, e.g. -L<
dir>.
[root@capbily madplay-0.15.2b]#
```

提示哦告诉我们还要设置 LDFLAGS 环境变量，因此再次修改配置参数如下：

```
#cd ./src-x86/madplay-0.15.2b
#./configure --prefix=/opt/FriendlyARM/QQ2440/madplay/target-x86 --CPPFLAGS=-I/opt/
FriendlyARM/QQ2440/madplay/target-x86/include --LDFLAGS=-L/opt/FriendlyARM/QQ
2440/madplay/target-x86/lib
```

执行结果如下：



追求卓越 创造精品

TO BE BEST

TO DO GREAT

广州友善之臂计算机科技有限公司

```
root@capbily:/opt/FriendlyARM/QQ2440/madplay/src-x86/madplay-0.15.2b
File Edit View Terminal Go Help
checking for library containing pow... -lm
checking for library containing log10... none required
checking whether to use mmap... yes
checking for audio support... oss
checking for esd_open_sound in -lesd... yes
checking whether to enable profiling... no
checking whether to enable debugging... default
checking whether to enable experimental code... no
configure: creating ./config.status
config.status: creating Makefile
config.status: creating msvc++/Makefile
config.status: creating intl/Makefile
config.status: creating po/Makefile.in
config.status: creating m4/Makefile
config.status: creating madplay.list
config.status: creating config.h
config.status: executing depfiles commands
config.status: executing default-1 commands
config.status: creating po/POTFILES
config.status: creating po/Makefile
[root@capbily madplay-0.15.2b]#
```

这说明配置已经成功，生成了编译所需要的 Makefile 文件，输入一下命令开始编译安装：

```
#make
#make install
```

执行完毕，可执行文件将被安装在我们指定的目录 /opt/FriendlyARM/QQ2440/madplay/target-x86 目录中：

```
#ls target-x86/bin
abxtest madplay
```

其中 abxtest 是附加生成的测试程序，不必理会。

## (5) 测试 PC 版的 madplay

拷贝一首 mp3 文件到 madplay 所在的目录，执行：

```
./madplay test.mp3
```

如果你的声卡已经正确安装，就可以听到 mp3 的声音了。

## (6) 构建编译脚本 build-x86

通过以上步骤，我们看到配置和编译不仅有一定的顺序，还需要注意一些安装细节，



虽然这次编译通过了，但不免以后会忘记这个过程，特别是当程序更加复杂的时候，因此我们要养成好习惯，把整个过程构建为一个脚本，以后只要执行这个脚本就可以完成所有步骤了，如图所示是 PC 版本 madplay 的构建脚本，该脚本位于 madplay 工作目录的根目录：

```
#!/bin/sh
#This build script is for madplay under PC-Linux

MADPLAY_DIR=/opt/FriendlyARM/QQ2440/madplay
SRC_DIR=src-x86
TARGET_DIR=$MADPLAY_DIR/target-x86

tar xvzf ./tarball/libid3tag-0.15.1b.tar.gz -C $SRC_DIR
tar xvzf ./tarball/libmad-0.15.1b.tar.gz -C $SRC_DIR
tar xvzf ./tarball/madplay-0.15.2b.tar.gz -C $SRC_DIR

cd $SRC_DIR/libid3tag-0.15.1b
./configure --prefix=$TARGET_DIR
make;make install
cd ../../

cd $SRC_DIR/libmad-0.15.1b
./configure --prefix=$TARGET_DIR
make;make install
cd ../../

cd $SRC_DIR/madplay-0.15.2b
./configure --prefix=$TARGET_DIR CPPFLAGS=-I$TARGET_DIR/include
LDFLAGS=-L$TARGET_DIR/lib
make;make install
cd ../../
```

## (7) 构建并修正 ARM 版本的编译脚本 build-arm

既然我们已经构建了一个简单易用的编译脚本，现在就可以通过对它稍作修改来进行交叉编译了，这就是通常所说的移植。简单的移植只要重新指定一下编译器就可以了，可以通过修改环境变量来实现。很多的移植所要修改的环境变量是

CC 编译器，系统默认为 gcc，  
AR 库工具，用以创建和修改库，系统默认 ar  
LD 链接器，系统默认为 LD



追求卓越 创造精品

TO BE BEST

TO DO GREAT

广州友善之臂计算机科技有限公司

RANLIB 随机库创建器，系统默认为 ranlib

AS 汇编器，系统默认为 as

NM 库查看工具，系统默认为 nm

还有一些不常用的其他环境变量，在此就不一一列举了。

需要注意的是，并不是每个移植都需要做全面的环境变量修改，有些是不需要改的，这要根据实际情况，也就是系统提示信息来调整。

除了要修改编译器环境变量，一般还需要在配置中加入目标平台指定标识，在此为“arm-linux”，修改后的脚本如下：

```
#!/bin/sh

MADPLAY_DIR=/opt/FriendlyARM/QQ2440/madplay
SRC_DIR=src-arm
TARGET_DIR=$MADPLAY_DIR/target-arm

tar xvzf ./tarball/libid3tag-0.15.1b.tar.gz -C $SRC_DIR
tar xvzf ./tarball/libmad-0.15.1b.tar.gz -C $SRC_DIR
tar xvzf ./tarball/madplay-0.15.2b.tar.gz -C $SRC_DIR

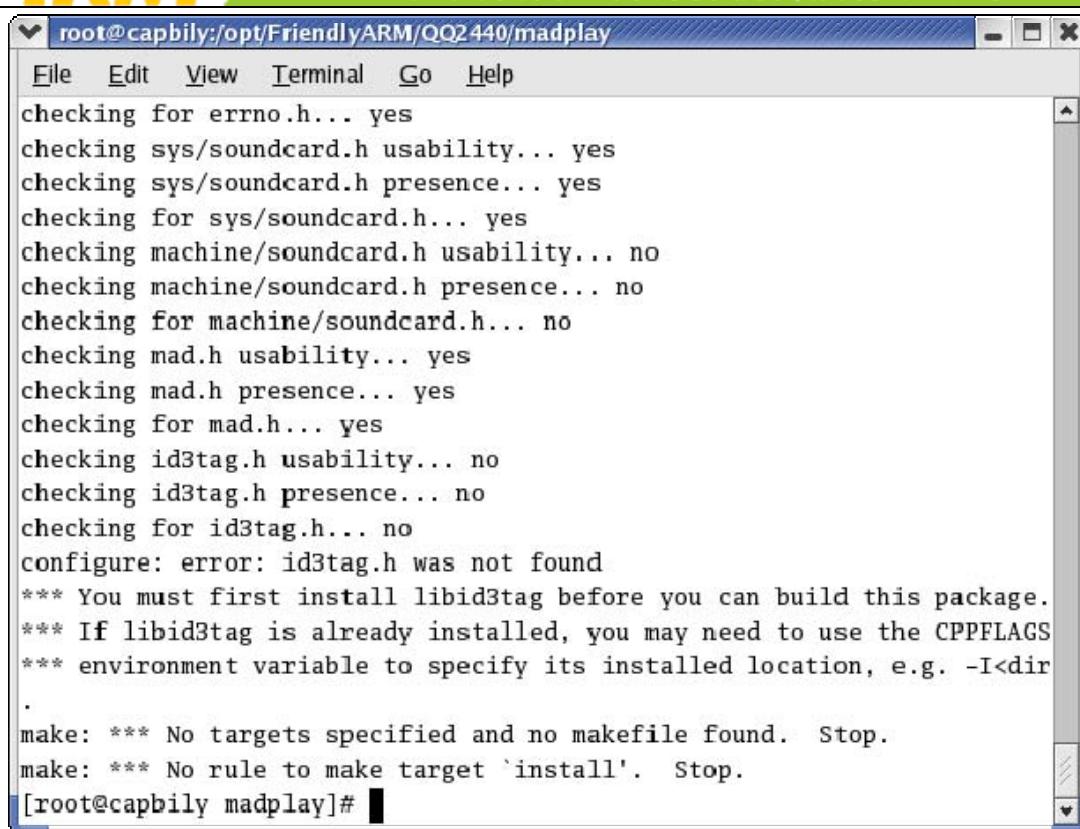
export CC=arm-linux-gcc

cd $SRC_DIR/libid3tag-0.15.1b
./configure --host=arm-linux --prefix=$TARGET_DIR CPPFLAGS=-I$TARGET_DIR/include
LDFLAGS=-L$TARGET_DIR/lib
make;make install
cd ../

cd $SRC_DIR/libmad-0.15.1b
./configure --host=arm-linux --prefix=$TARGET_DIR
make;make install
cd ..

cd $SRC_DIR/madplay-0.15.2b
./configure --host=arm-linux --prefix=$TARGET_DIR CPPFLAGS=-I$TARGET_DIR/include
LDFLAGS=-L$TARGET_DIR/lib
make;make install
cd ../
cd ../../
```

现在我们直接运行一下该脚本，看结果，如图：



```
root@capbily:/opt/FriendlyARM/QQ2440/madplay
File Edit View Terminal Go Help
checking for errno.h... yes
checking sys/soundcard.h usability... yes
checking sys/soundcard.h presence... yes
checking for sys/soundcard.h... yes
checking machine/soundcard.h usability... no
checking machine/soundcard.h presence... no
checking for machine/soundcard.h... no
checking mad.h usability... yes
checking mad.h presence... yes
checking for mad.h... yes
checking id3tag.h usability... no
checking id3tag.h presence... no
checking for id3tag.h... no
configure: error: id3tag.h was not found
*** You must first install libid3tag before you can build this package.
*** If libid3tag is already installed, you may need to use the CPPFLAGS
*** environment variable to specify its installed location, e.g. -I<dir>
.
make: *** No targets specified and no makefile found. Stop.
make: *** No rule to make target 'install'. Stop.
[root@capbily madplay]#
```

根据提示，可以推断 libid3tag 库没有被正确编译出来，再根据更上的提示信息，我们得知该库还依赖于一个叫做“zlib”的库，为什么 PC 版本的没有这个问题呢，是因为我们所使用的 PC Linux 系统中已经有了这个库。因此我们从网上搜索到该库的源代码包，下载下来放到 tarball 目录中，并在编译脚本中参考其他库在相应位置加入以下部分：

```
tar xvzf ./tarball/zlib-1.2.3.tar.gz -C $SRC_DIR
cd $SRC_DIR/zlib-1.2.3
./configure --prefix=$TARGET_DIR
make && make install
cd ../../
```

再次执行编译脚本，这次顺利编译通过，最后在 target-arm/bin 目录中可以看到交叉编译生成的 madplay，使用 file 命令检查一下，如图：



追求卓越 创造精品

TO BE BEST

TO DO GREAT

广州友善之臂计算机科技有限公司

```
root@capbily:/opt/FriendlyARM/QQ2440/madplay/target-arm/bin
File Edit View Terminal Go Help
mkdir -p -- . /opt/FriendlyARM/QQ2440/madplay/target-arm/bin
/usr/bin/install -c abxtest /opt/FriendlyARM/QQ2440/madplay/target-arm/
/bin/abxtest
mkdir -p -- . /opt/FriendlyARM/QQ2440/madplay/target-arm/man/man1
/usr/bin/install -c -m 644 ./madplay.1 /opt/FriendlyARM/QQ2440/madplay/
/target-arm/man/man1/madplay.1
/usr/bin/install -c -m 644 ./abxtest.1 /opt/FriendlyARM/QQ2440/madplay/
/target-arm/man/man1/abxtest.1
make[2]: Leaving directory '/opt/FriendlyARM/QQ2440/madplay/src-arm/mad
play-0.15.2b'
make[1]: Leaving directory '/opt/FriendlyARM/QQ2440/madplay/src-arm/mad
play-0.15.2b'
[root@capbily madplay]# ls
bok build-arm src-arm src-x86 tarball target-arm target-x86
[root@capbily madplay]# cd target-arm/bin/
[root@capbily bin]# ls
abxtest madplay
[root@capbily bin]# file madplay
madplay: ELF 32-bit LSB executable, ARM, version 1 (ARM), for GNU/Linux
2.4.3, dynamically linked (uses shared libs), not stripped
[root@capbily bin]#
```

最后完整的编译脚本如图：



追求卓越 创造精品

TO BE BEST

TO DO GREAT

广州友善之臂计算机科技有限公司

```
#!/bin/sh

MADPLAY_DIR=/opt/FriendlyARM/QQ2440/madplay
SRC_DIR=src-arm
TARGET_DIR=$MADPLAY_DIR/target-arm

tar xvzf ./tarball/libid3tag-0.15.1b.tar.gz -C $SRC_DIR
tar xvzf ./tarball/libmad-0.15.1b.tar.gz -C $SRC_DIR
tar xvzf ./tarball/madplay-0.15.2b.tar.gz -C $SRC_DIR
tar xvzf ./tarball/zlib-1.2.3.tar.gz -C $SRC_DIR

export CC=arm-linux-gcc

cd $SRC_DIR/zlib-1.2.3
./configure --prefix=$TARGET_DIR
make && make install
cd ../../

cd $SRC_DIR/libid3tag-0.15.1b
./configure --host=arm-linux --prefix=$TARGET_DIR CPPFLAGS=-I$TARGET_DIR/include
LDFLAGS=-L$TARGET_DIR/lib
make;make install
cd ../../

cd $SRC_DIR/libmad-0.15.1b
./configure --host=arm-linux --prefix=$TARGET_DIR
make;make install
cd ../../

cd $SRC_DIR/madplay-0.15.2b
./configure --host=arm-linux --prefix=$TARGET_DIR CPPFLAGS=-I$TARGET_DIR/include
LDFLAGS=-L$TARGET_DIR/lib
make;make install
cd ../../
```

## (8) 下载 madplay 到开发板运行测试

为了区别于板子中已经存在的 madplay，我们把新做的改名为 mymadplay，把它以及依



追求卓越 创造精品

TO BE BEST

TO DO GREAT

广州友善之臂计算机科技有限公司

赖库通过 ftp 下载到开发板，并作如下放置：

执行文件：

mymadplay 放在 /usr/bin/ 目录

库文件：

libid3tag.a	libid3tag.la	libid3tag.so	libid3tag.so.0
libid3tag.so.0.3.0	libmad.a	libmad.la	libmad.so
libmad.so.0	libmad.so.0.2.1	libz.a	

放在 /usr/lib 目录。

执行结果如图所示：

```
[root@FriendlyARM bin]# ls /usr/lib/
libid3tag.a      libid3tag.so.0.3.0  libmad.so.0
libid3tag.la     libmad.a          libmad.so.0.2.1
libid3tag.so     libmad.la         libz.a
libid3tag.so.0   libmad.so        telnetlogin
[root@FriendlyARM bin]# ls /usr/bin/mymadplay
/usr/bin/mymadplay
[root@FriendlyARM bin]# mymadplay /shanghaitan.mp3
MPEG Audio Decoder 0.15.2 (beta) - Copyright (C) 2000-2004 Robert Leslie et al
    Title: 上海滩
    Artist: 叶丽仪
    Year: 2000
    Genre: Goa
```



## 第七章 常见 bootloader 的配置和编译

在 S3C2440/2410 系统中，常见的 bootloader 一般有

- Vivi – 由三星提供，韩国 mizi 公司原创，开放源代码，必须使用 arm-linux-gcc 进行编译，目前已经基本停止发展，主要适用于三星 S3C24xx 系列 ARM 芯片，用以启动 Linux 系统，支持串口下载和网络文件系统启动等常用简易功能。
- Supervivi – 由友善之臂提供并积极维护，它基于 vivi 发展而来，不提供源代码，在保留原始 vivi 功能的基础上，整合了诸多其他实用功能，如**支持 CRAMFS, YAFFS 文件系统**，USB 下载，自动识别并启动 Linux, WinCE, uCos, Vxwork 等多种嵌入式操作系统，下载程序到内存中执行，并**独创了系统备份和恢复功能**，非常适合在批量生产中使用，是目前 2440/2410 系统中功能最强大最好用的 bootloader。
- YL-BIOS – 深圳优龙基于三星的监控程序 24xxmon 改进而来，**提供源代码，可以使用 ADS 进行编译**，整合了 USB 下载功能，仅支持 CRAMFS 文件系统，并增加了手工设置启动 Linux 和 WinCE，下载到内存执行测试程序等多种实用功能。因其开源性，故该 bootloader 被诸多其他嵌入式开发板厂商所采用，需要注意的是大部分是未经优龙公司授权的。
- U-Boot – 一个开源的专门针对嵌入式 Linux 系统设计的最流行 bootloader，必须使用 arm-linux-gcc 进行编译，具有**强大的网络功能(失去网络，U-Boot 基本丧失其最独到的优势，在 2440/2410 系统中网络是添加网络芯片外扩的，毫无疑问会增加成本，而 USB 是内置的，只需几个电阻配置)**，支持网络下载内核并通过网络启动系统，U-Boot 处于更加活跃的更新发展之中，但对于 2440/2410 系统来说，它尚未支持 Nand Flash 启动，国内已经有人为此自行加入了这些功能，本章节中的 U-Boot 即是如此。

Bootloader 以其本身的含义来讲就是下载和启动系统，它类似于 PC 中的 BIOS，大部分芯片厂商所提供的嵌入式系统都提供有这样的程序，而且都比较成熟，大可不必自行编写。我们所改进的 supervivi 目标是使之更加人性化，更加适合于批量生产需要。

以下各个版本的 BIOS 均包含原始代码，和我们稍微改造后以适应于 mini2440 的全部原代码，在测试对比过程中我们发现，使用以下几个 BIOS 的过程无异于经历一场噩梦，你并不能十分顺利的使用通过标准方式编译出来的内核和文件系统，你需要一系列的配置、转换，还要搭建一个局域网，这其中会遇到很多挫折、郁闷和彷徨。

使用 supervivi 则会让你更加注重于实际的开发工作，你可以轻松把自己所作下载到开发板并运行起来，只要会操作鼠标就可以，就像一次愉快的旅行，根本不需要专业知识，也不需要输入复杂的命令，毕竟这是一个飞速发展的时代，我们谁都不想浪费时间在不必要的事情上！

下面我们使用图解的方式介绍一下那些开源 BIOS 的配置和编译过程，其使用方法暂时不做介绍。



追求卓越 创造精品

TO BE BEST

TO DO GREAT

广州友善之臂计算机科技有限公司

## 7.1 配置和编译 vivi

注意：编译 vivi 需要设置交叉编译环境为 2.95.3 版本的编译器，请参考“建立 linux 开发环境”一节。

### 7.1.1 使用缺省配置编译

vivi 的源代码包 **vivi.tgz** 位于光盘的/OpenSourceBootloader 目录，把 **vivi.tgz** 复制到某一个目录，进入该目录，运行以下命令：

```
#tar xvzf vivi.tgz -C /opt/FriendlyARM/mini2440
```

执行该命令将把 vivi 源代码解压到/opt/FriendlyARM/mini2440 目录，进入 vivi 源代码目录，执行：

```
#cd /opt/FriendlyARM/mini2440/vivi  
#make clean  
#make menuconfig
```

出现以下界面：

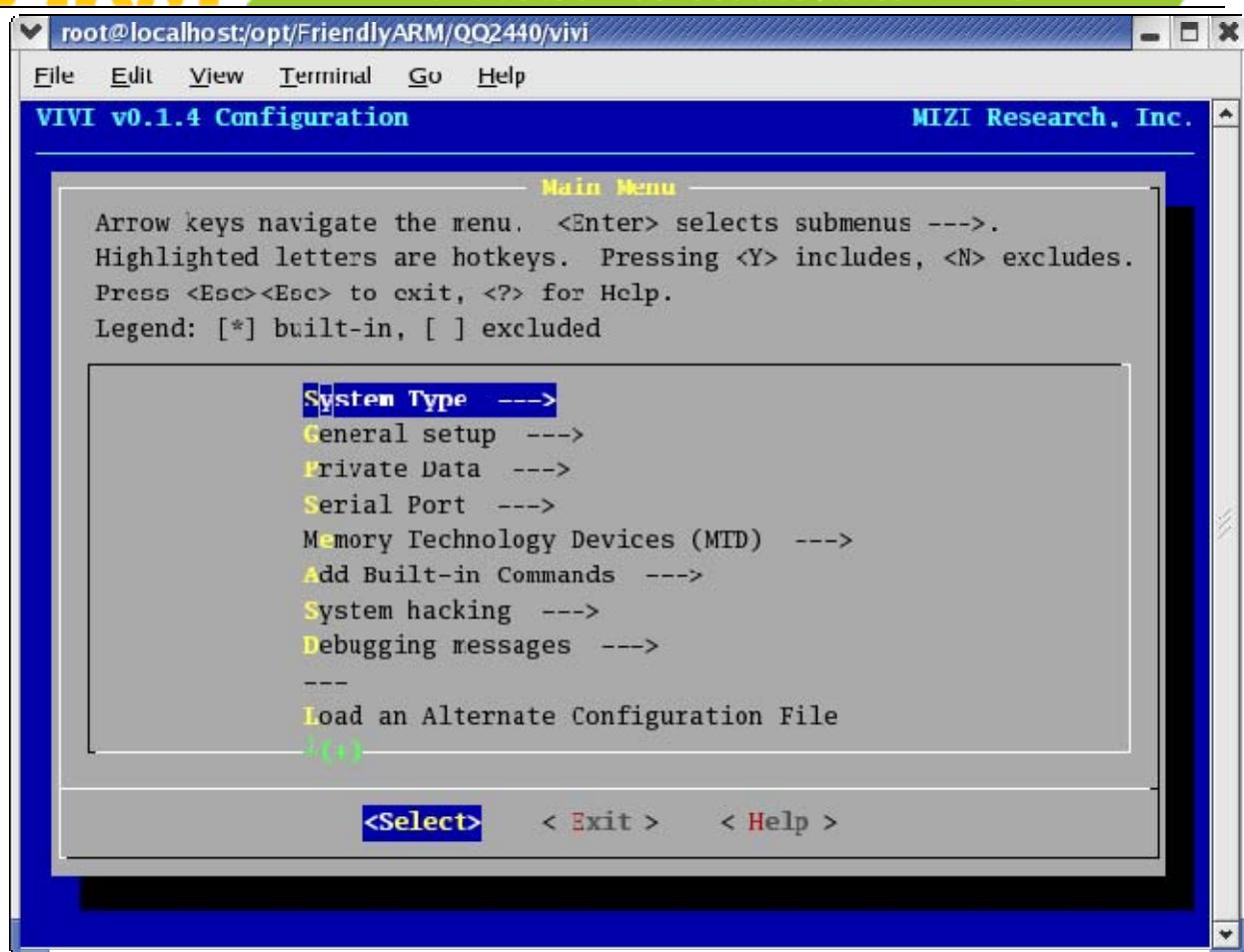


追求卓越 创造精品

TO BE BEST

TO DO GREAT

广州友善之臂计算机科技有限公司



一般不需要更改任何配置，按左右方向键，选择 **<Exit>**，如图：

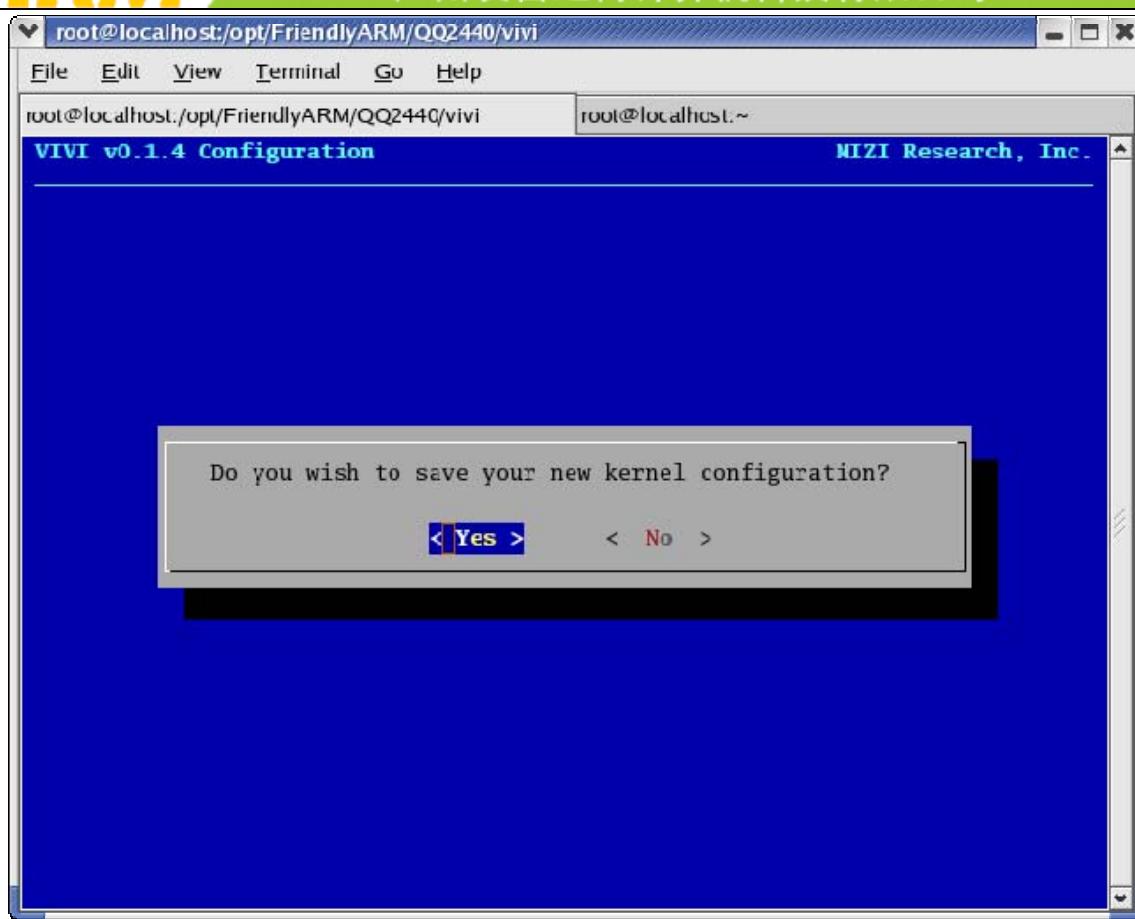


追求卓越 创造精品

TO BE BEST

TO DO GREAT

广州友善之臂计算机科技有限公司



选择<Yes>, 按回车退出, 然后执行“**make**”开始编译, 执行结果如下:

#make

```

root@localhost:/opt/FriendlyARM/QQ2440/vivi
File Edit View Terminal Go Help
root@localhost:/opt/FriendlyARM/QQ2440/vivi          root@localhost:~
make[2]: Nothing to be done for `all_targets'.
make[2]: Leaving directory `/opt/FriendlyARM/QQ2440/vivi/lib'
make[1]: Leaving directory `/opt/FriendlyARM/QQ2440/vivi/lib'
make CFLAGS="-I/opt/FriendlyARM/QQ2440/vivi/include -I/usr/local/arm/2.95.3/include -Wall -Wstrict-prototypes -O2 -fPIC -fomit-frame-pointer -mapcs-32 -nshort-load-bytes -msoft-float" -C arch/s3c2440
make[1]: Entering directory `/opt/FriendlyARM/QQ2440/vivi/arch/s3c2440'
make all_targets
make[2]: Entering directory `/opt/FriendlyARM/QQ2440/vivi/arch/s3c2440'
make[2]: Nothing to be done for `all_targets'.
make[2]: Leaving directory `/opt/FriendlyARM/QQ2440/vivi/arch/s3c2440'
make[1]: Leaving directory `/opt/FriendlyARM/QQ2440/vivi/arch/s3c2440'
/usr/local/arm/2.95.3/bin/arm-linux-ld -v -Tarch/vivi.lds -Bstatic \
    arch/s3c2440/head.o \
    arch/s3c2440/s3c2440.o init/main.o init/version.o lib/lib.o \
    drivers/serial/serial.o drivers/mtd/mtd.o \
    lib/priv_data/priv_data.o \
    -o vivi-elf -L/usr/local/arm/2.95.3/lib/gcc-lib/arm-linux/2.95.3 -lgcc -lc
GNU ld version 2.11.2 (with BFD 2.11.2)
/usr/local/arm/2.95.3/bin/arm-linux-nm -v -l vivi-elf > vivi.map
/usr/local/arm/2.95.3/bin/arm-linux-objcopy -O binary -S vivi-elf vivi -R .comment -R .stab -R .stabstr
[root@localhost vivi]#

```

此时已经在当前目录下生成了 vivi，您可以参考上面的章节把 vivi 烧写到目标板的 Nand Flash 运行。

### 7.1.2 配置 vivi 从 Nor Flash 启动

有的用户抱怨说编译出的 vivi 不能正常运行使用，大部分情况是因为对系统的不了解所导致，上一小节编译出的 vivi 只能烧写到 Nand Flash 运行使用。要使你的 vivi 能够从 Nor Flash 启动运行，需要重新配置一下，它不像 supervivi 那样可以自动识别 Nor Flash/Nand Flash。

进入 vivi 目录，执行“make menuconfig”开始配置：

选择 System Type → Implementations → Support AMD Boot，同时取消选择 Support NandBoot，如图：

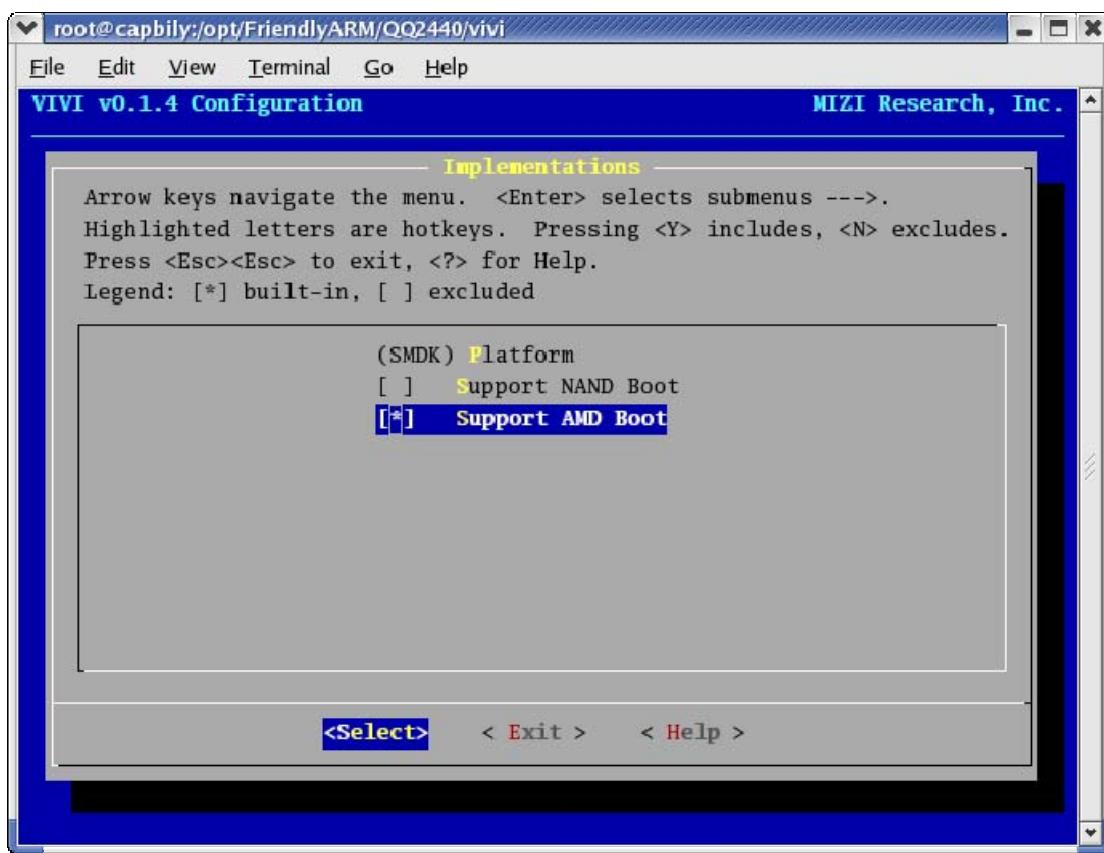


追求卓越 创造精品

TO BE BEST

TO DO GREAT

广州友善之臂计算机科技有限公司



保存退出，执行：

#make clean

#make

这样生成的 vivi 就可以使用 H-JTAG 烧写(参见 2.6 一节)到 Nor Flash 启动了。

## 7.2 使用 ADS 编译 YL-BIOS

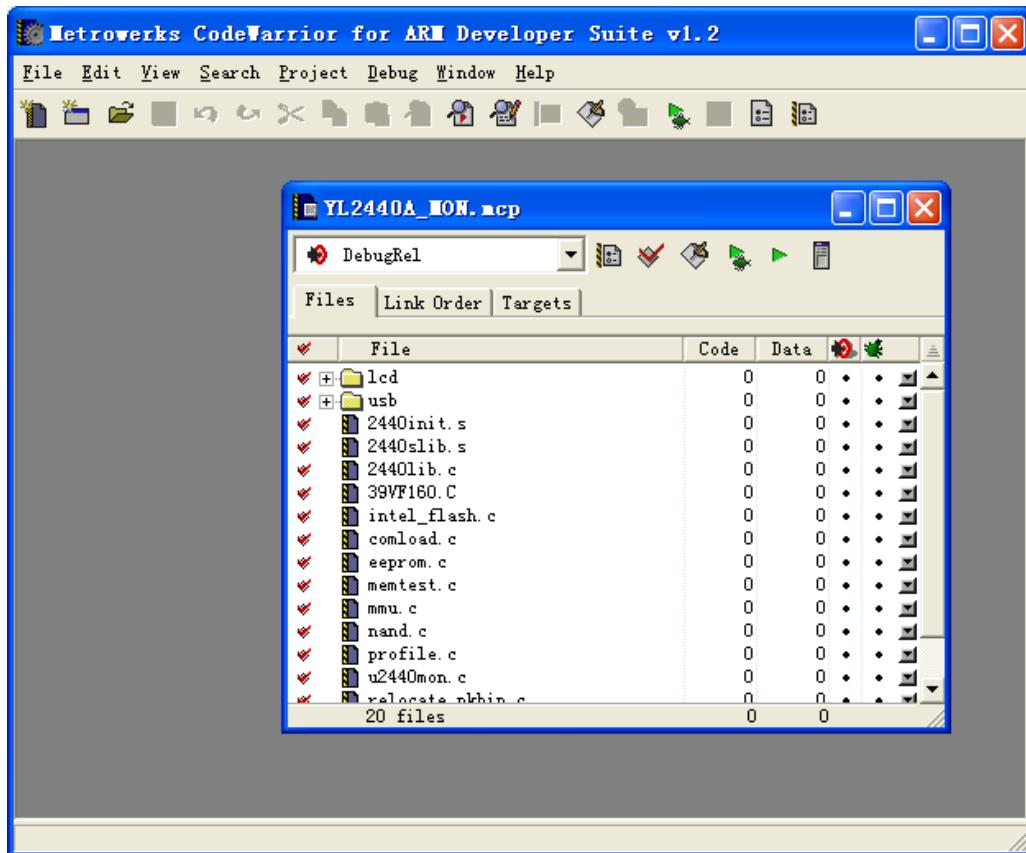
**注意：要编译使用优龙的 BIOS，首先要安装好 ADS 开发环境。**

**说明：对于 YL-BIOS 我们没有做太多测试和说明，毕竟这是第三方 BIOS，也非主流，在此仅作参考，本公司不提供关于该 BIOS 的任何技术支持和咨询。**

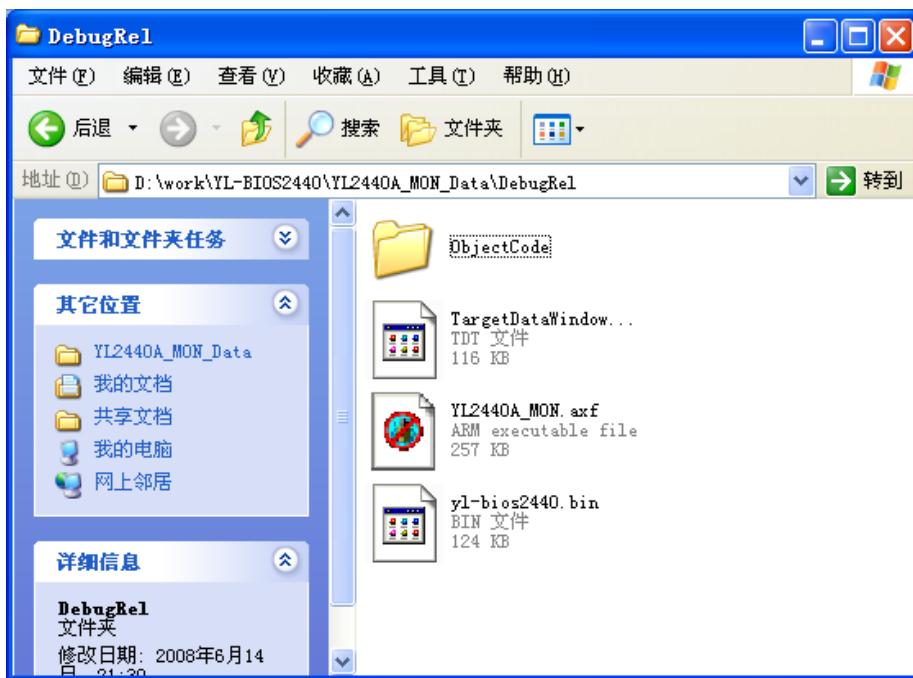
### 7.2.1 使用 ADS 编译 YL-BIOS

(1)YL-BIOS2440 源代码位于光盘的/OpenSourceBootloader 文件夹中，把它复制到您的电脑中，如“D:\work”，去掉其只读属性。

(2)使用 ADS 打开 YL2440A\_MON.mcp 文件：

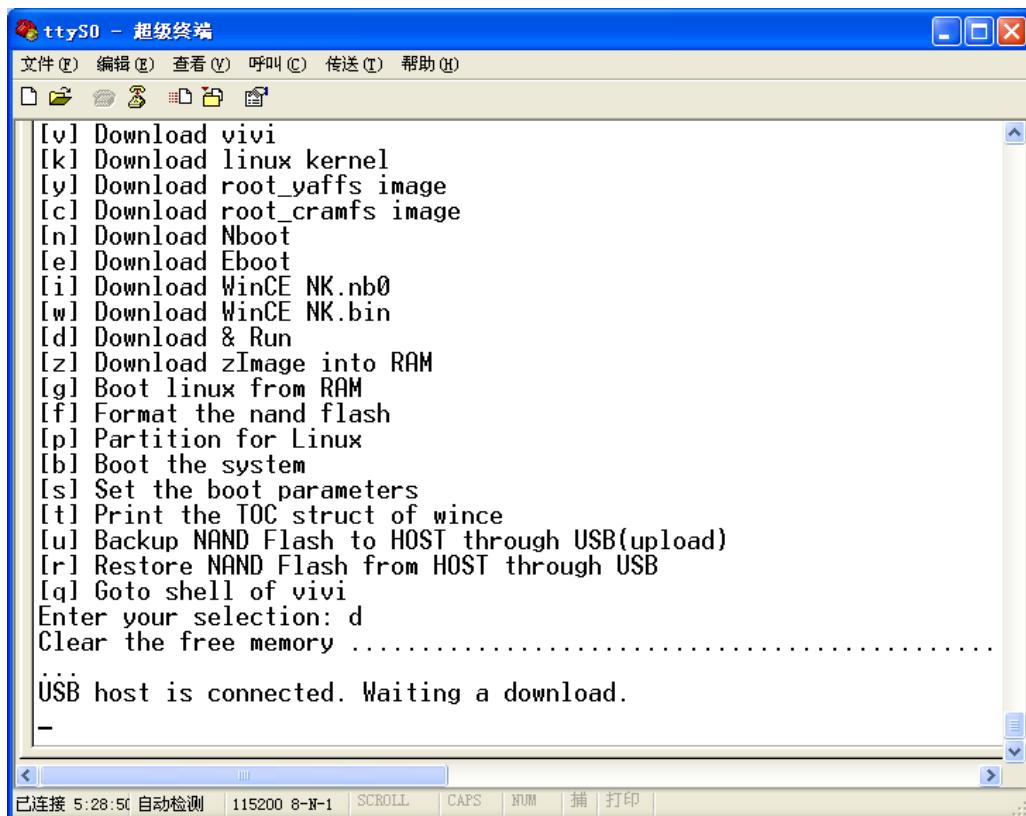


(3)不要修改任何设置，直接点击 Project→Make 或者按 F7 键开始编译，最后在“D:\work\YL-BIOS2440\YL2440A\_MON\_Data\DebugRel”目录下生成 yl-bios2440.bin 文件：



## 7.2.2 把 YL-BIOS 下载到内存中运行

把 mini2440 拨动开关 S2 设置为 Nor Flash 启动，并打开超级终端，进入功能菜单模式，并输入功能命令“d”，如图：



打开运行 DNW，设置下载地址为 0x30100000 (这是 YL-BIOS 的默认运行地址)：



追求卓越 创造精品

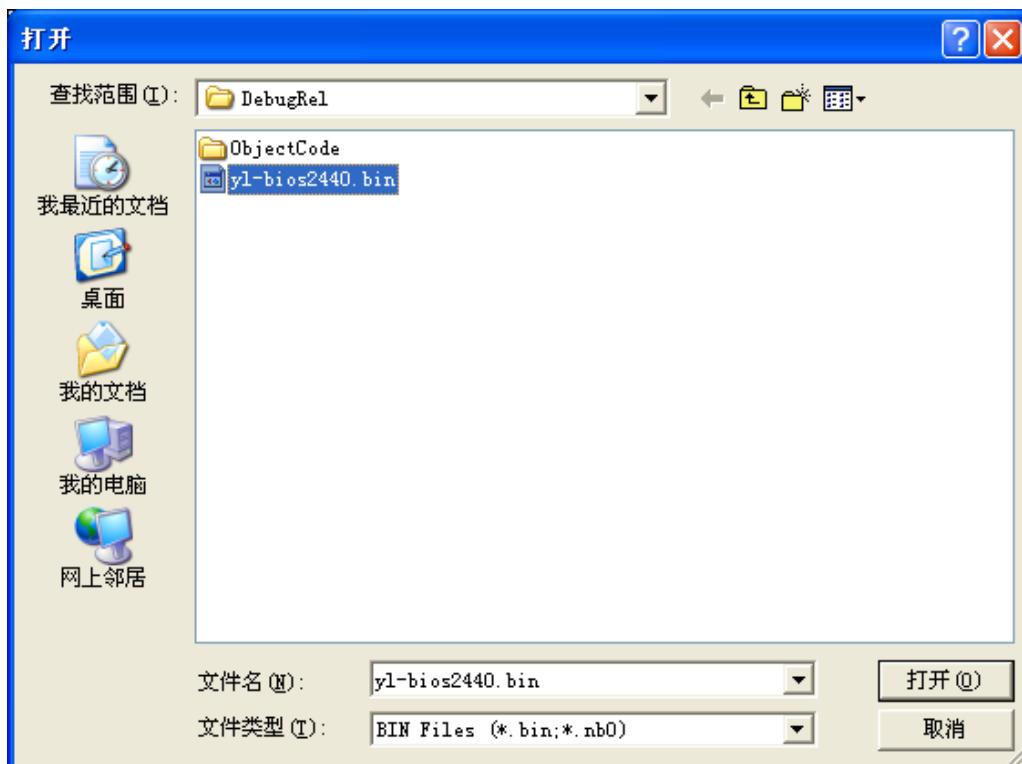
TO BE BEST

TO DO GREAT

广州友善之臂计算机科技有限公司



点 Usb Port → Transmit/Restore，并选择刚才所编译出的 yl-bios2440.bin 开始通过 USB 下载到内存中：



下载很快结束，这时串口终端出现如图所示：



ttyS0 - 超级终端

文件(F) 编辑(E) 查看(V) 呼叫(C) 传送(T) 帮助(H)

Autoboot delay is 0 seconds.

<\*\*\*\*\*>

Check SST39VF160 Flash ID is 0x22490001

NOR Flash Man. ID is 0xb

Unsupported Flash Type!

Fail to write camera IIC!

DIVN\_UPLL0

MPPLLVal [M:5ch,P:1h,S:1h]

CLKDIVN:dh

+-----+  
| S3C2440A USB Downloader ver R0.03 2004 Jan |  
+-----+

FCLK=400.0MHz, DMA mode

USB: IN\_ENDPOINT:1 OUT\_ENDPOINT:3

FORMAT: <ADDR(DATA):4>\*<SIZE(n+10):4>\*<DATA:n>\*<CS:2>

NOTE: 1. Power off/on or press the reset button for 1 sec  
in order to get a valid USB device address.  
2. For additional menu, Press any key.

USB host is not connected yet.

USB host is connected. Waiting a download.

按一下空格键就可以进入 YL-BIOS 的菜单了，如图：

The screenshot shows a terminal window titled "ttyS0 - 超级终端". The menu bar includes "文件(F)", "编辑(E)", "查看(V)", "呼叫(C)", "传送(T)", and "帮助(H)". Below the menu is a toolbar with icons for file operations. The main text area displays the following information:

```
+-----+
| S3C2440A USB Downloader ver R0.03 2004 Jan |
+-----+
FCLK=400.0MHz, DMA mode
USB: IN_ENDPOINT:1 OUT_ENDPOINT:3
FORMAT: <ADDR(DATA):4>+<SIZE(n+10):4>+<DATA:n>+<CS:2>
NOTE: 1. Power off/on or press the reset button for 1 sec
      in order to get a valid USB device address.
      2. For additional menu, Press any key.

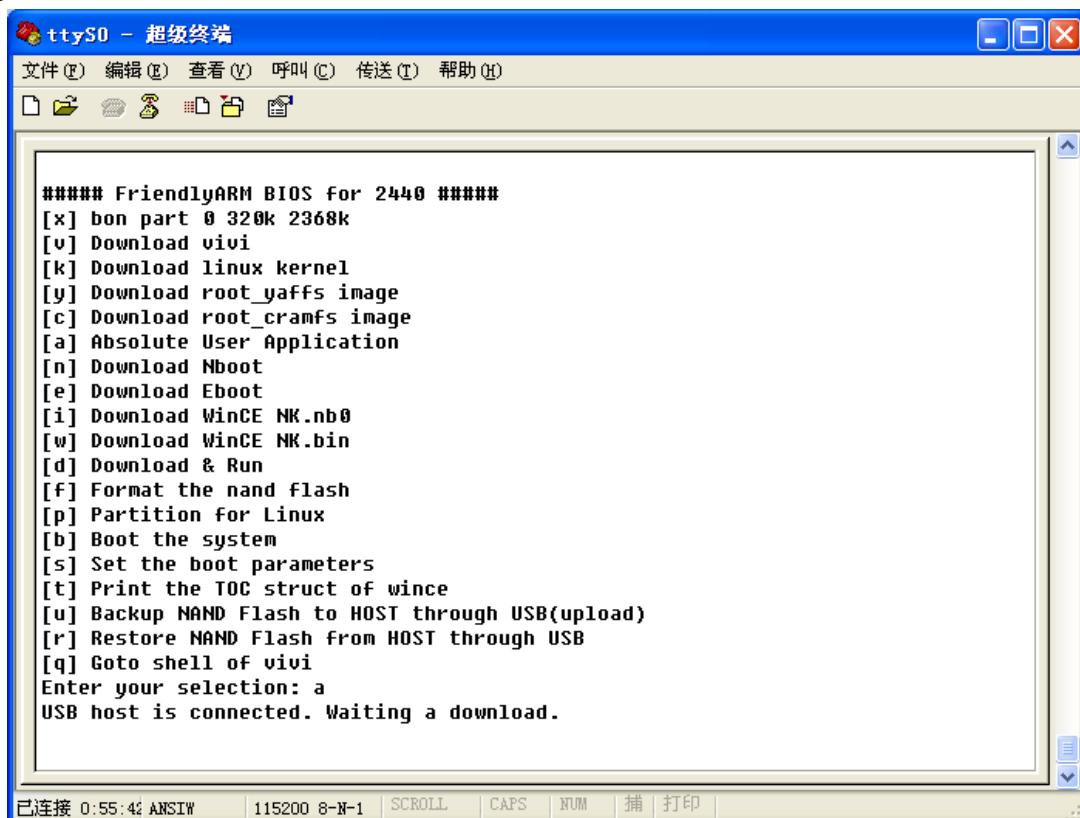
USB host is not connected yet.
USB host is connected. Waiting a download.

##### Select Menu #####
[0] Download & Run
[1] Download Only
[2] Download From UART
[3] Write File to SST39VF160
[4] Write File to NAND Flash
[5] Boot OS
[6] Erase NAND Flash Partition
[7] Config parameters
[8] Relocate NK.bin
```

### 7.2.3 烧写 YL-BIOS 到开发板

YL-BIOS 还可以直接烧写到 Nand Flash 中运行使用。

(1)和上面的步骤类似，首先确定设置 mini2440 为 Nor Flash 启动模式，打开超级终端进入 supervivi 的 BIOS 模式，并输入功能命令“a”：



(2)打开 DNW，确认 USB 连接正常 OK，点 UsbPort→Transmit/Restore，选择刚才所编译的 yl-bios2440.bin，下载和烧写很快就会结束。

(3)把 mini2440 启动模式跳线改为 Nand Flash 启动，重新复位或者重启开机关电源开关，在串口终端可以看到如图信息：



按任意键进入 YL-BIOS 的菜单模式，如图：



追求卓越 创造精品

TO BE BEST

TO DO GREAT

广州友善之臂计算机科技有限公司

### 注意事项：

至此你已经可以编译、烧写并启动了 YL-BIOS，在此之后你在 Nand Flash 中将使用一个全新的第三方 BIOS，你不能再继续使用 supervivi 烧写内核和文件系统，并用 YL-BIOS 启动它们，因为其采用的一些参数和友善之臂并不相同；你必须按照 YL-BIOS 的功能手册来操作剩余的部分，关于如何进一步使用 YL-BIOS 请用户自行查找资料解决。

当然你也可以使用 SJF2440 工具软件来烧写 YL-BIOS，那将会十分慢，但不能使用 H-JTAG，因为它尚不支持 Nand Flash 烧写。

## 7.3 配置和编译 U-Boot

说明：本小节只介绍 U-Boot 的配置编译和烧写，关于其使用方法的详细介绍，我们将会在以后的手册更新中添加，用户也可以自行到网上查找相关资料。

本光盘中的 U-Boot 具有以下功能特性：

1. 同时支持 S3C2410 和 S3C2440
2. 支持串口 xmodem 协议
3. 支持 USB 下载，可以在 PC 上使用 dnw 传数据
4. 支持网卡芯片 CS8900
5. 支持 NAND Flash 读写
6. 支持从 Nor/Nand Flash 启动
7. 支持烧写 yaffs 文件系统映象
8. 可以直接下载到内存运行

9. 即可以支持 CS8900，又可以支持 DM9000，但是，不能同时支持；要选择支持哪个网卡芯片，需要在 include/configs/100ask24x0.h 中进行配置，如下：

```
#if 0
#define CONFIG_DRIVER_CS8900 1 /* we have a CS8900 on-board */
#define CS8900_BASE      0x19000300
#define CS8900_BUS16     1 /* the Linux driver does accesses as shorts */
#endif
```

```
#if !defined(CONFIG_DRIVER_CS8900)
#define CONFIG_DRIVER_DM9000      1
#define CONFIG_DM9000_USE_16BIT   1
#define CONFIG_DM9000_BASE        0x20000000
#define DM9000_IO                 0x20000000
#define DM9000_DATA                0x20000004
#endif
```

下面是具体的编译方法和烧写步骤。

### 7.3.1 配置和编译 U-Boot

**注意：要编译 U-Boot 请使用光盘带的 arm-linux-gcc-3.4.1 编译器，这是带有浮点处理功能的编译器。U-Boot 源代码包位于光盘的/OpenSourceBootloader 目录中**

首先解压源代码包到工作目录：

```
#tar xvzf u-boot-1.1.6-FA24x0.tar.gz -C /opt/FriendlyARM/mini2440
```

执行该命令将把 U-Boot 源代码解压到 /opt/FriendlyARM/mini2440 目录。

再检查一下当前编译器版本，执行：

```
#arm-linux-gcc -v
```

如图，注意是带软浮点运算功能的编译器：

```
root@capcross:~/opt/FriendlyARM/mini2440/u-boot-1.1.6# arm-linux-gcc -v
Reading specs from /usr/local/arm/3.4.1/bin/../lib/gcc/arm-linux/3.4.1/specs
Configured with: /opt/crosstool/crosstool-0.28/build/arm-linux/gcc-3.4.1-glibc-2
.3.2/gcc-3.4.1/configure --target=arm-linux --host=i686-host_pc-linux-gnu --pref
ix=/opt/crosstool/arm-linux/gcc-3.4.1-glibc-2.3.2 --with-float=soft --with-heade
rs=/opt/crosstool/arm-linux/gcc-3.4.1-glibc-2.3.2/arm-linux/include --with-local
-prefix=/opt/crosstool/arm-linux/gcc-3.4.1-glibc-2.3.2/arm-linux --disable-nls -
-enable-threads=posix --enable-symvers=gnu --enable_cxa_atexit --enable-langua
ges=c,c++ --enable-shared --enable-c99 --enable-long-long
Thread model: posix
gcc version 3.4.1
[root@capcross u-boot-1.1.6]#
```

配置 U-Boot 十分简单，进入 U-Boot 目录，执行：

```
#make open24x0_config
```

```
#make
```

就可以开始编译了，编译完毕，如图所示生成 u-boot.bin

```
root@capcross:/opt/FriendlyARM/mini2440/u-boot-1.1.6
File Edit View Terminal Go Help
on.a --end-group -L /usr/local/arm/3.4.1/bin/.../lib/gcc/arm-linux/3.4.1 -lgcc \
    -Map u-boot.map -o u-boot
arm-linux-objcopy --gap-fill=0xff -O srec u-boot u-boot.srec
arm-linux-objcopy --gap-fill=0xff -O binary u-boot u-boot.bin
[root@capcross u-boot-1.1.6]# ls
arm_config.mk           include          mips_config.mk
avr32_config.mk         lib_arm         mkconfig
blackfin_config.mk      lib_avr32       mkconfig.l
board                   lib_blackfin    nand_spl
CHANGELOG                lib_generic     net
CHANGELOG-before-U-Boot-1.1.5 lib_i386        nios2_config.mk
common                  lib_m68k        nios_config.mk
config.mk               lib_microblaze post
COPYING                 lib_mips        ppc_config.mk
cpu                     lib_nios       README
CREDITS                 lib_nios2      rtc
disk                    lib_ppc         rules.mk
doc                     m68k_config.mk System.map
drivers                 MAINTAINERS   tools
dtt                     MAKEALL        u-boot
examples                Makefile       u-boot.bin
fs                      Makefile.l    u-boot.map
i386_config.mk          microblaze_config.mk u-boot.srec
[root@capcross u-boot-1.1.6]#
```

### 7.3.2 把 U-Boot 烧写到开发板

要烧写 U-Boot，需要把开发板拨动开关 S2 设置为 Nor Flash 启动，连接好串口和 USB 线，打开超级终端，打开电源，串口显示如图：

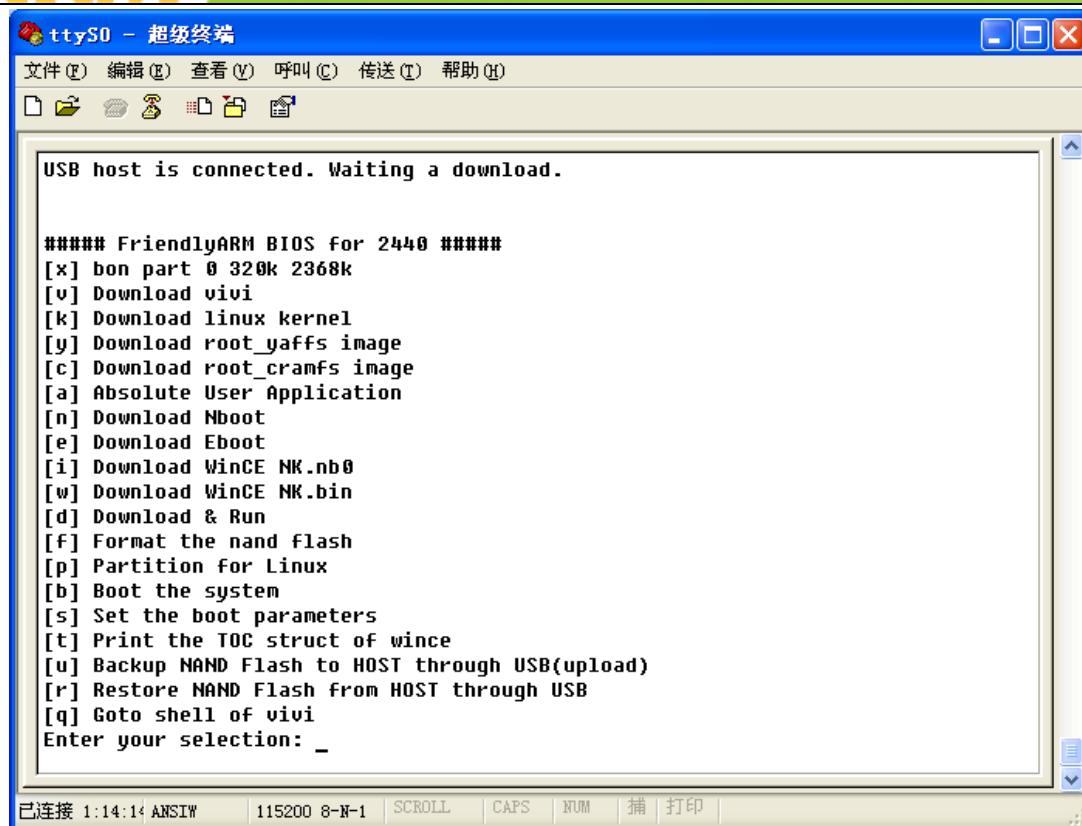


追求卓越 创造精品

TO BE BEST

TO DO GREAT

广州友善之臂计算机科技有限公司



选择功能号“a”，打开 DNW，确认 USB 连接正常 OK，点 UsbPort→Transmit/Restore，选择刚才所编译的 u-boot.bin，下载和烧写很快就会结束。

把 mini2440 启动模式改为 Nand Flash 启动，重新复位或者重启开机关电源开关，在串口终端可以看到如图信息，如果开发板中已经安装了 linux 系统，U-Boot 将会自动启动它，否则会进入 U-Boot 的功能菜单(也可以根据提示，在开机后 3 秒中内按任意键进入)：



追求卓越 创造精品

TO BE BEST

TO DO GREAT

广州友善之臂计算机科技有限公司

```
U-Boot 1.1.6 (Oct 27 2008 - 06:45:29)
DRAM: 64 MB
Flash: 1 MB
NAND: 64 MiB
*** Warning - bad CRC, using default environment

In:    serial
Out:   serial
Err:   serial
UPLLVal [M:38h,P:2h,S:2h]
MPLLVal [M:5ch,P:1h,S:1h]
CLKDIVN:5h

+-----+
| S3C2440A USB Downloader ver R0.03 2004 Jan |
+-----+
USB: IN_ENDPOINT:1 OUT_ENDPOINT:3
FORMAT: <ADDR(DATA):4>+<SIZE(n+10):4>+<DATA:n>+<CS:2>
NOTE: Power off/on or press the reset button for 1 sec
      in order to get a valid USB device address.

Hit any key to stop autoboot: 3
```

已连接 1:57:17 ANSIW 115200 8-N-1 SCROLL CAPS NUM 插 打印

U-Boot 的功能菜单如图，您可以按照功能选项进行测试，它和 supervivi 基本是类似的：

```
Hit any key to stop autoboot: 0
Booting Linux ...

NAND read: device 0 offset 0x0, size 0x2000000

reading NAND page at offset 0x0 failed
Could not read entire image due to bad blocks
2097152 bytes read: ERROR
## Booting image at 32000000 ...
Bad Magic Number

##### open24x0 Bootloader for FA24x0 #####
[u] Download u-boot
[k] Download Linux kernel
[j] Download JFFS2 image
[y] Download YAFFS image
[d] Download to SDRAM & Run
[b] Boot the system
[f] Format the Nand Flash
[s] Set the boot parameters
[r] Reboot u-boot
[q] Quit from menu
Enter your selection:
```

已连接 1:59:45 ANSIW 115200 8-N-1 SCROLL CAPS NUM 插 打印

注意：若使用 U-Boot，这时你需要完全抛弃 Nor Flash 中 supervivi 的那些功能命令，使用 U-Boot 本身的功能来下载和烧写内核，以及文件系统。关于 U-Boot 的使用方法用户可以参考网上的资料，我们将在以后的文档更新中添加。



追 求 卓 越 创 造 精 品

TO BE BEST

TO DO GREAT

广州友善之臂计算机科技有限公司

## 第八章 配置和编译 linux 内核

注意：编译内核需要设置交叉编译环境为 3.4.1 版本的编译器，请参考“建立 linux 开发环境”一节。

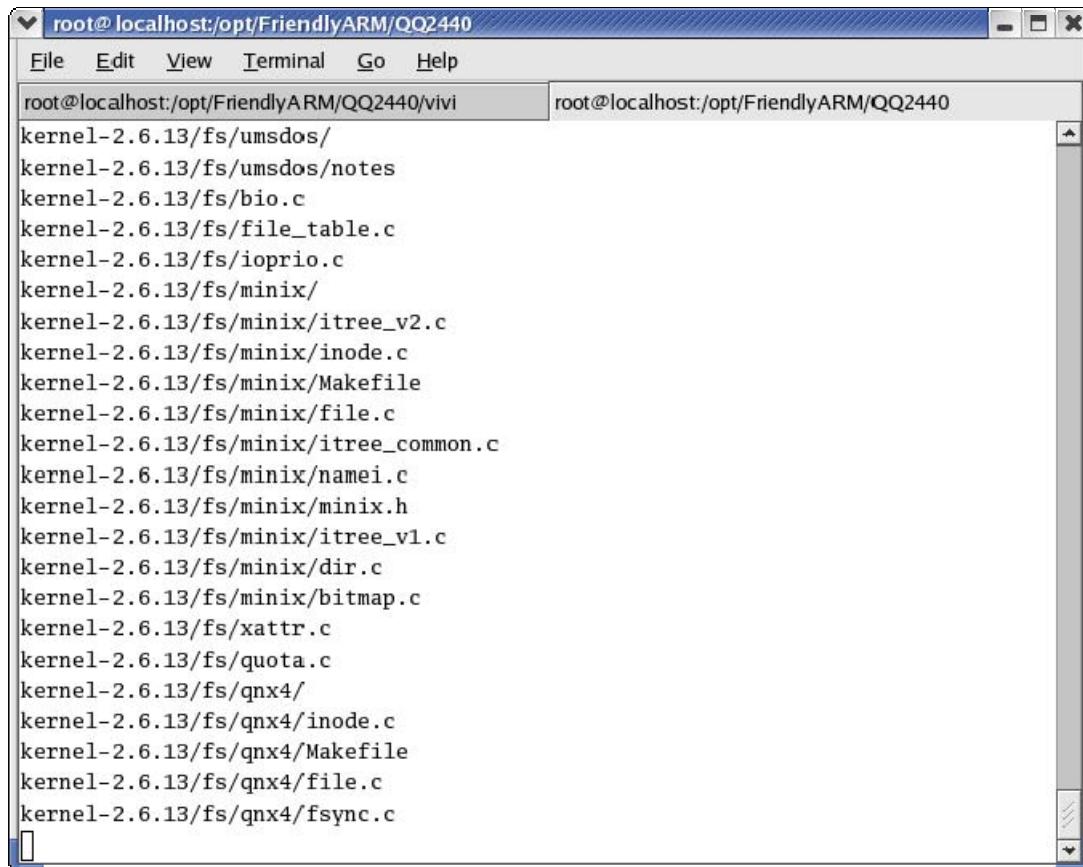
### 8.1 使用缺省配置文件编译内核

#### 8.1.1 解压内核源代码

Linux 内核的源代码包位于光盘的 `linux\` 目录，您在该目录下还会看到类似 `linux-2.6.13-mini2440_20080910.tgz` 的文件。

把内核源代码包复制到某一个目录，进入该目录，运行以下命令：

```
#tar xvzf linux-2.6.13-mini2440_20080910.tgz -C /opt/FriendlyARM/mini2440
```



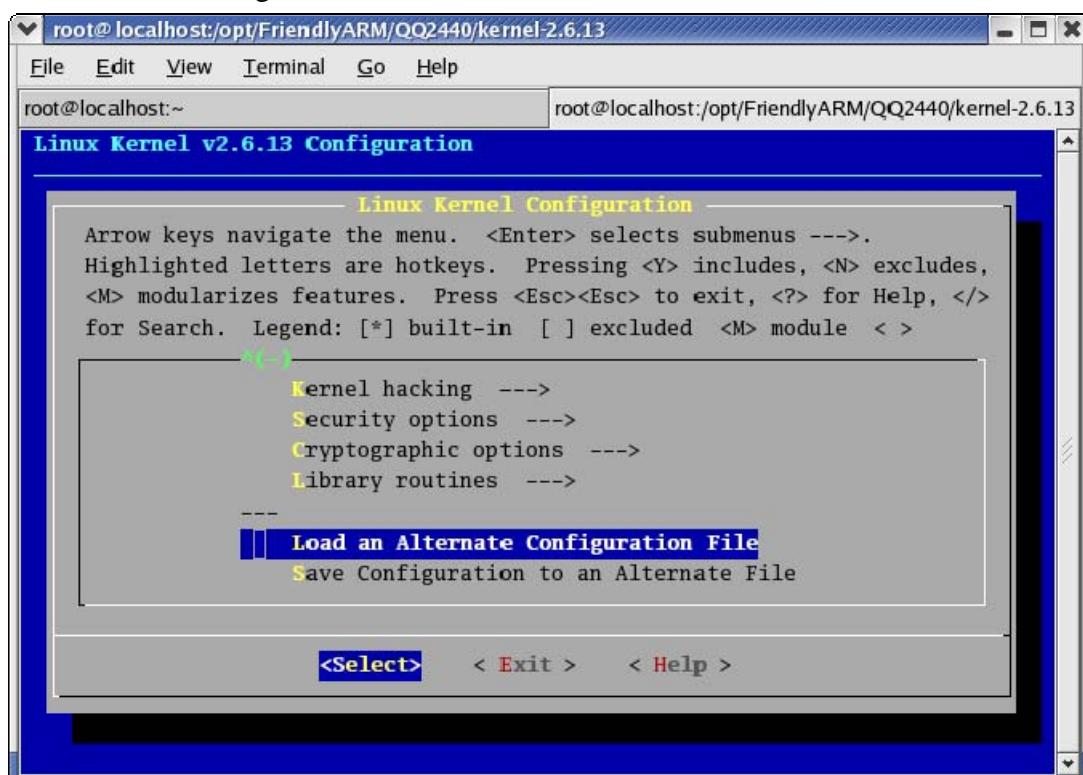
The screenshot shows a terminal window titled "root@localhost:/opt/FriendlyARM/QQ2440". The command "#tar xvzf linux-2.6.13-mini2440\_20080910.tgz -C /opt/FriendlyARM/mini2440" has been run, and the output lists numerous files extracted from the tarball, primarily related to the fs (file system) directory of the Linux kernel version 2.6.13.

```
root@localhost:/opt/FriendlyARM/QQ2440/vivi root@localhost:/opt/FriendlyARM/QQ2440
File Edit View Terminal Go Help
root@localhost:/opt/FriendlyARM/QQ2440/vivi root@localhost:/opt/FriendlyARM/QQ2440
kernel-2.6.13/fs/umsdos/
kernel-2.6.13/fs/umsdos/notes
kernel-2.6.13/fs/bio.c
kernel-2.6.13/fs/file_table.c
kernel-2.6.13/fs/ioprio.c
kernel-2.6.13/fs/minix/
kernel-2.6.13/fs/minix/itree_v2.c
kernel-2.6.13/fs/minix/inode.c
kernel-2.6.13/fs/minix/Makefile
kernel-2.6.13/fs/minix/file.c
kernel-2.6.13/fs/minix/itree_common.c
kernel-2.6.13/fs/minix/namei.c
kernel-2.6.13/fs/minix/minix.h
kernel-2.6.13/fs/minix/itree_v1.c
kernel-2.6.13/fs/minix/dir.c
kernel-2.6.13/fs/minix(bitmap.c
kernel-2.6.13/fs/xattr.c
kernel-2.6.13/fs/quota.c
kernel-2.6.13/fs/qnx4/
kernel-2.6.13/fs/qnx4/inode.c
kernel-2.6.13/fs/qnx4/Makefile
kernel-2.6.13/fs/qnx4/file.c
kernel-2.6.13/fs/qnx4/fsync.c
```

这样将把内核源代码解压到/opt/FriendlyARM/mini2440/kernel-2.6.13 目录

### 8.1.2 装载缺省配置文件

进入内核源代码目录，然后执行“**make menuconfig**”，出现配置内核界面，选择进入“Load an Alternate Configuration File”配置栏：



输入配置文件名如 **config\_n35** 并回车，在主菜单里选择<Exit>退出并保存设置。

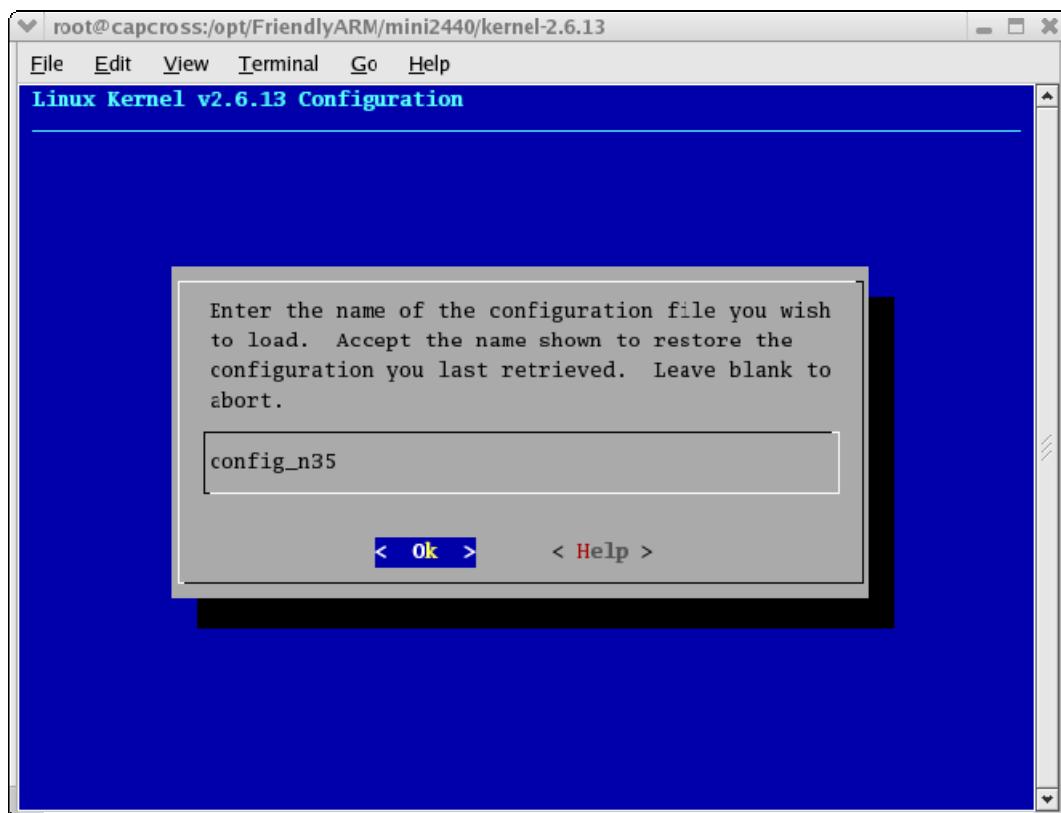


追 求 卓 越 创 造 精 品

TO BE BEST

TO DO GREAT

广州友善之臂计算机科技有限公司



### 8.1.3 编译内核

**再次提示：编译内核需要 arm-linux-gcc-3.4.1 版本的编译器，请务必检查安装好。**

输入以下命令，开始编译内核：

#make zImage



追求卓越 创造精品

TO BE BEST

TO DO GREAT

广州友善之臂计算机科技有限公司

```
root@localhost:/opt/FriendlyARM/QQ2440/kernel-2.6.13
File Edit View Terminal Go Help
root@localhost:~                                     root@localhost:/opt/FriendlyARM/QQ2440/kernel-2.6.13
CHK      include/asm-arm/constants.h
UPD      include/asm-arm/constants.h
Generating include/asm-arm/mach-types.h
CC       init/main.o
CHK      include/linux/compile.h
UPD      include/linux/compile.h
CC       init/version.o
CC       init/do_mounts.o
CC       init/do_mounts_devfs.o
CC       init/do_mounts_rd.o
LD       init-mounts.o
CC       init/initramfs.o
CC       init/calibrate.o
LD       init/built-in.o
HOSTCC   usr/gen_init_cpio
CHK      usr/initramfs_list
UPD      usr/initramfs_list
CPIO     usr/initramfs_data.cpio
GZIP    usr/initramfs_data.cpio.gz
AS      usr/initramfs_data.o
LD      usr/built-in.o
CC      arch/arm/kernel/compat.o
CC      arch/arm/kernel/dma.o
```

编译结束后，会在 arch/arm/boot 目录下生成 linux 内核映象文件：zImage



追求卓越 创造精品

TO BE BEST

TO DO GREAT

广州友善之臂计算机科技有限公司

The terminal window shows the following output:

```
root@localhost:/opt/FriendlyARM/QQ2440/kernel-2.6.13
File Edit View Terminal Go Help
root@localhost:~                                     root@localhost:/opt/FriendlyARM/QQ2440/kernel-2.6.13
GEN      .version
CHK      include/linux/compile.h
UPD      include/linux/compile.h
CC       init/version.o
LD       init/built-in.o
LD       .tmp_vmlinux1
KSYM    .tmp_kallsyms1.S
AS       .tmp_kallsyms1.o
LD       .tmp_vmlinux2
KSYM    .tmp_kallsyms2.S
AS       .tmp_kallsyms2.o
LD       vmlinux
SYSMAP  System.map
SYSMAP  .tmp_System.map
OBJCOPY arch/arm/boot/Image
Kernel: arch/arm/boot/Image is ready
AS       arch/arm/boot/compressed/head.o
GZIP    arch/arm/boot/compressed/piggy.gz
AS       arch/arm/boot/compressed/piggy.o
CC       arch/arm/boot/compressed/misc.o
LD       arch/arm/boot/compressed/vmlinux
OBJCOPY arch/arm/boot/zImage
Kernel: arch/arm/boot/zImage is ready
[root@localhost kernel-2.6.13]#
```

您可以使用以上章节介绍的方法把 zImage 下载到开发板测试。

#### 8.1.4 各个 Linux 驱动程序源代码位置

说明：解压内核源代码(linux-2.6.13 开头的 tgz 压缩文件)可以找到如下驱动，mini2440 提供 linux-2.6.13 内核 100% 完全可以使用的驱动源代码，绝无库文件，敬请放心使用。

##### (1)DM9000 网卡驱动

kernel-2.6.13/drivers/net/dm9000x.c

##### (2)串口(包括三个串口驱动 0,1,2， 对应设备名/dev/tts/0,1,2)

kernel-2.6.13/drivers/serial/s3c2410.c

##### (3)实时时钟 RTC 驱动

kernel-2.6.13/drivers/char/s3c2410-rtc.c

##### (4)LED 驱动

kernel-2.6.13/drivers/char/qq2440\_leds.c

##### (5)按键驱动

kernel-2.6.13/drivers/char/qq2440\_buttons.c

##### (6)触摸屏驱动

kernel-2.6.13/drivers/input/touchscreen/s3c2410\_ts.c



(7)yaffs 文件系统源代码目录

kernel-2.6.13/fs/yaffs2

(8)USB 鼠标、键盘源代码

kernel-2.6.13/drivers/usb/input/hid-input.c

(9)SD/MMC 卡驱动源代码目录(在 2.6.13 内核中仅支持 2G 容量以内的 SD 卡)

kernel-2.6.13/drivers/mmc

(10)Nand Flash 驱动

kernel-2.6.13/drivers/mtd/nand

(11)UDA1341 音频驱动目录

kernel-2.6.13/sound/oss/uda1341.c

kernel-2.6.13/drivers/l3

(12)LCD 驱动(包含 3.5", 7", 8.4", 10.4", 12.4", 15"等大小的驱动)

kernel-2.6.13/drivers/video/s3c2410fb.c

(13)优盘支持驱动

kernel-2.6.13/drivers/usb/storage

(14)中星微 USB 摄像头驱动

kernel-2.6.13/drivers/usb/media/gspca

## 8.2 定制 linux 内核

上面我们使用缺省文件配置和编译了内核，其实 linux 内核的配置选项有很多，下面我们就常见的一些选项分别予以图解，帮助你尽快熟悉内核配置，以便定制自己需要的内核。

运行 make menuconfig 后，进入内核配置主菜单

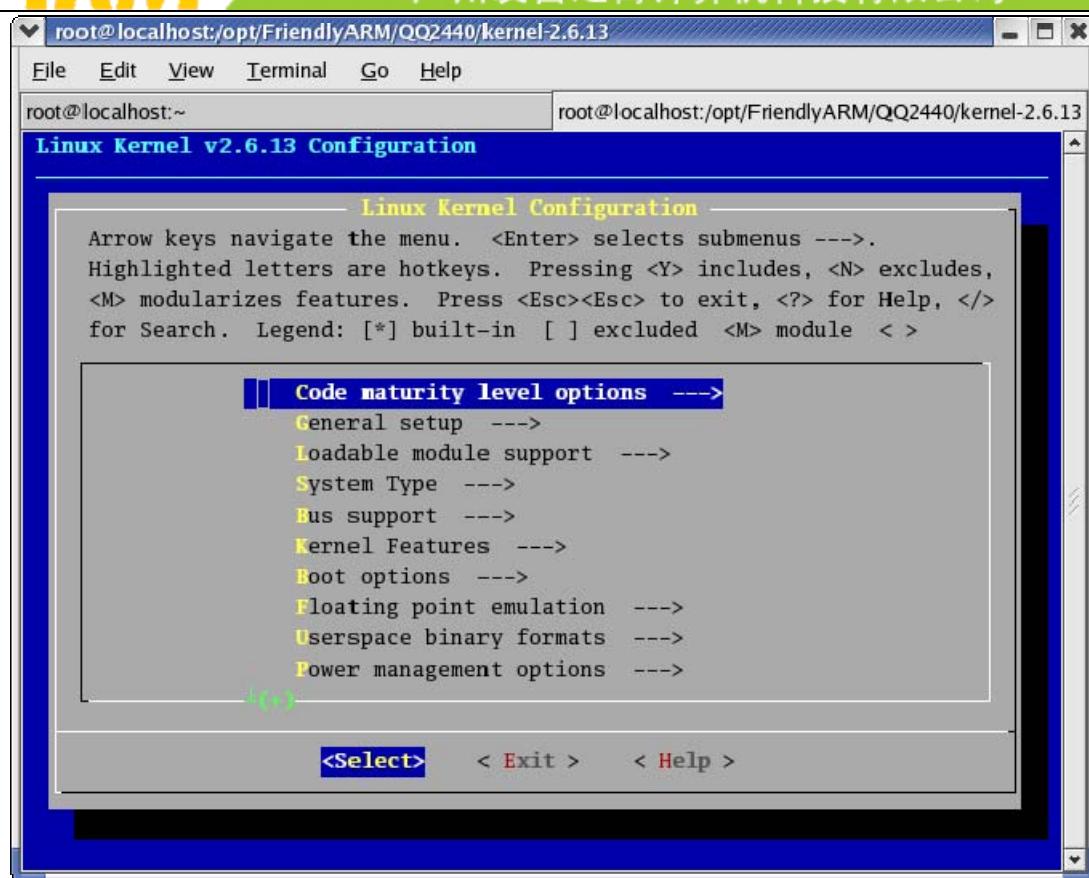


追求卓越 创造精品

TO BE BEST

TO DO GREAT

广州友善之臂计算机科技有限公司



### 8.2.1 如何配置 CPU 选项

在主菜单里面，选择 System Type，按回车进入

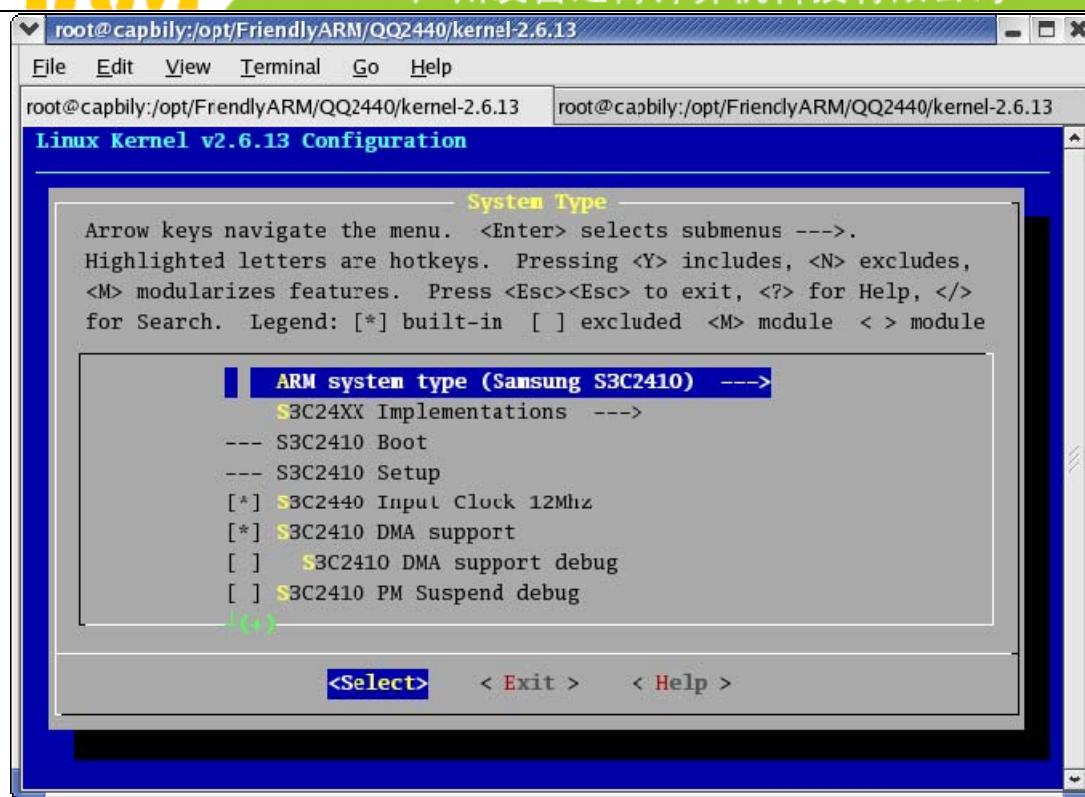


追求卓越 创造精品

TO BE BEST

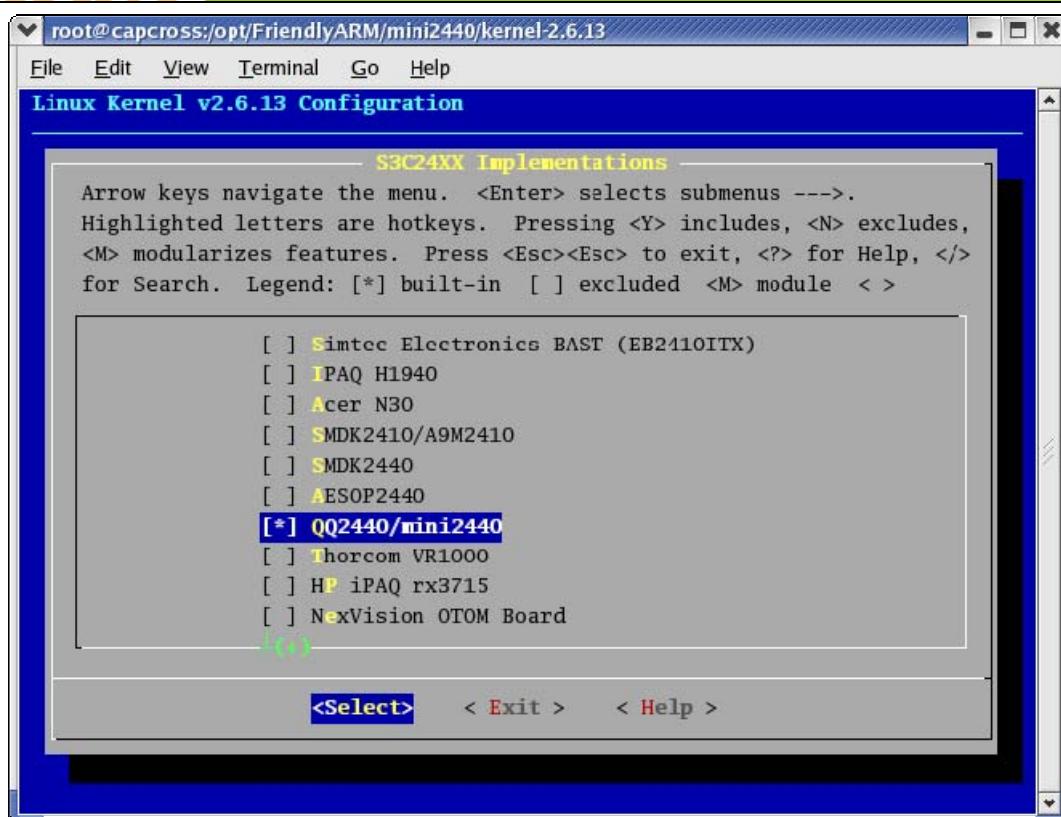
TO DO GREAT

广州友善之臂计算机科技有限公司



可以看到，系统大部分使用了标注了 S3C2410 的选项，这是因为 S3C2410 和 S3C2440 的很多寄存器地址等地址和设置是完全相同的，该版本的 linux-2.6 内核不再对这两种 CPU 分别设置。

如果您要选择板级选项，可以进入 S3C24XX Implementations 子菜单，里面有很多常见的使用基于 S3C2410 和 S3C2440 的目标板平台选项



它们分别对应于 **arch/arm/mach-s3c2410/mach-\*** 开头的文件，如 IPAQ H1940 对应于 **mach-h1940.c**，我们的开发板平台为 **mini2440**，它和 QQ2440 使用基本相同的配置，因此它对应于 **mach-qq2440.c**。另外，在这个文件中，还会用到一个机器码 **MACH\_TYPE**，该机器码的定义文件为 **arch/arm/tools/mach-types**，其中，我们开发板的机器码为 **782**，它还对应于 vivi 源代码中 **include/platform/smdk2440.h** 文件的 **MACH\_TYPE**

### 8.2.2 如何配置各个尺寸的 LCD 驱动支持

在主菜单里面，选择 Device Drivers，按回车进入

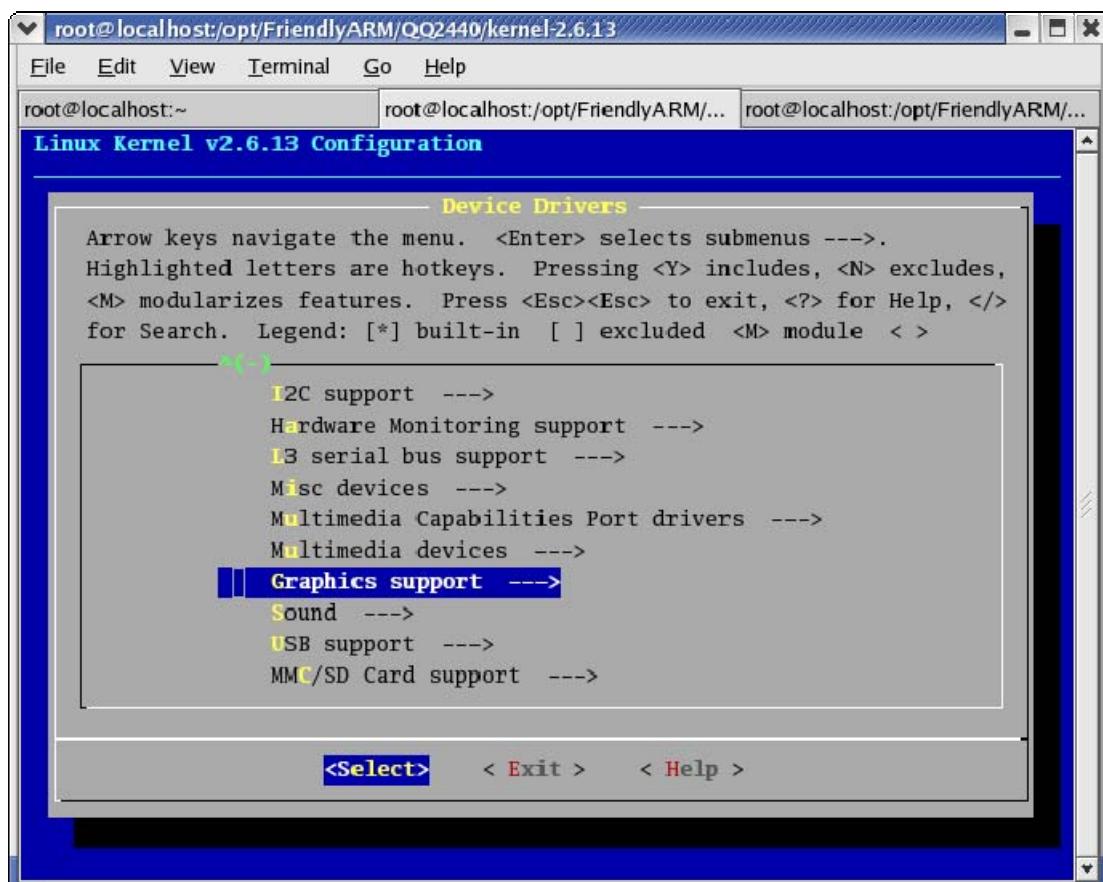


追求卓越 创造精品

TO BE BEST

TO DO GREAT

广州友善之臂计算机科技有限公司



选择 Graphics support，按回车进入，选中：

<\*> Support for frame buffer devices

<\*> S3C2410 LCD framebuffer support

选中 LCD select，按回车进入，如图

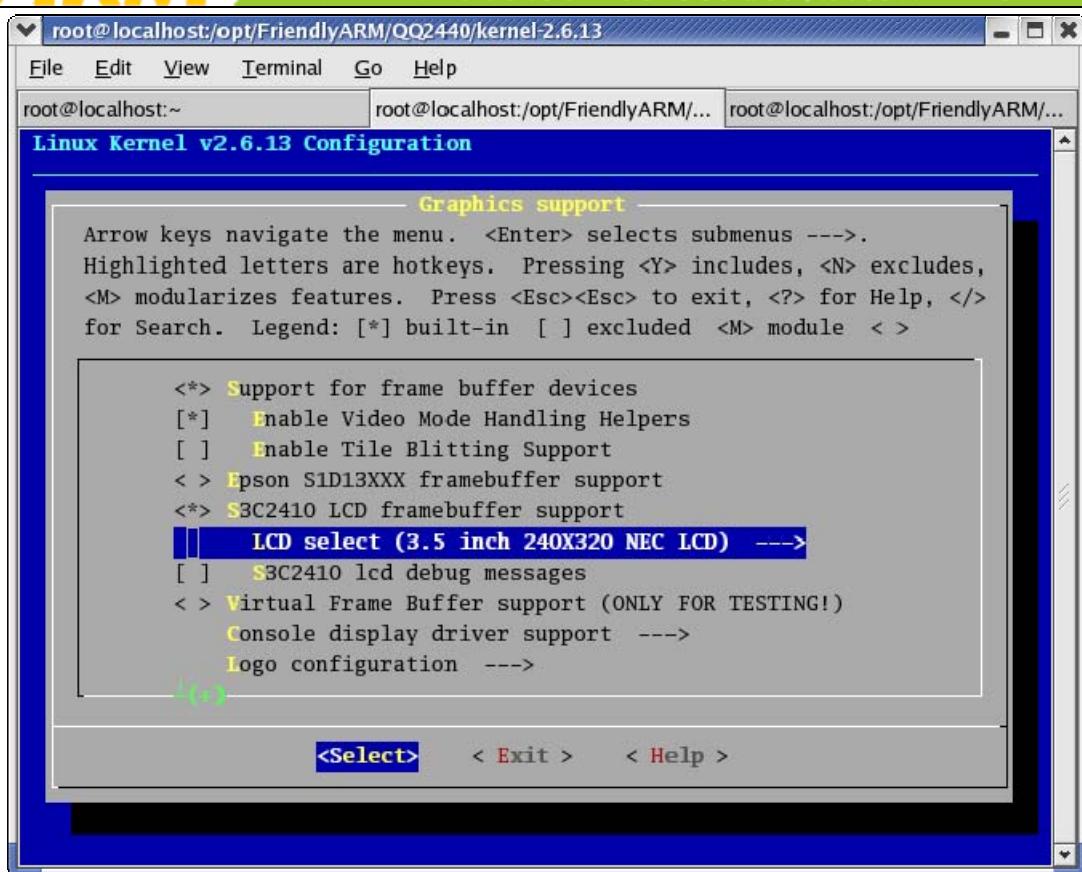


追求卓越 创造精品

TO BE BEST

TO DO GREAT

广州友善之臂计算机科技有限公司



在该子菜单里面您可以选择需要的 LCD 驱动

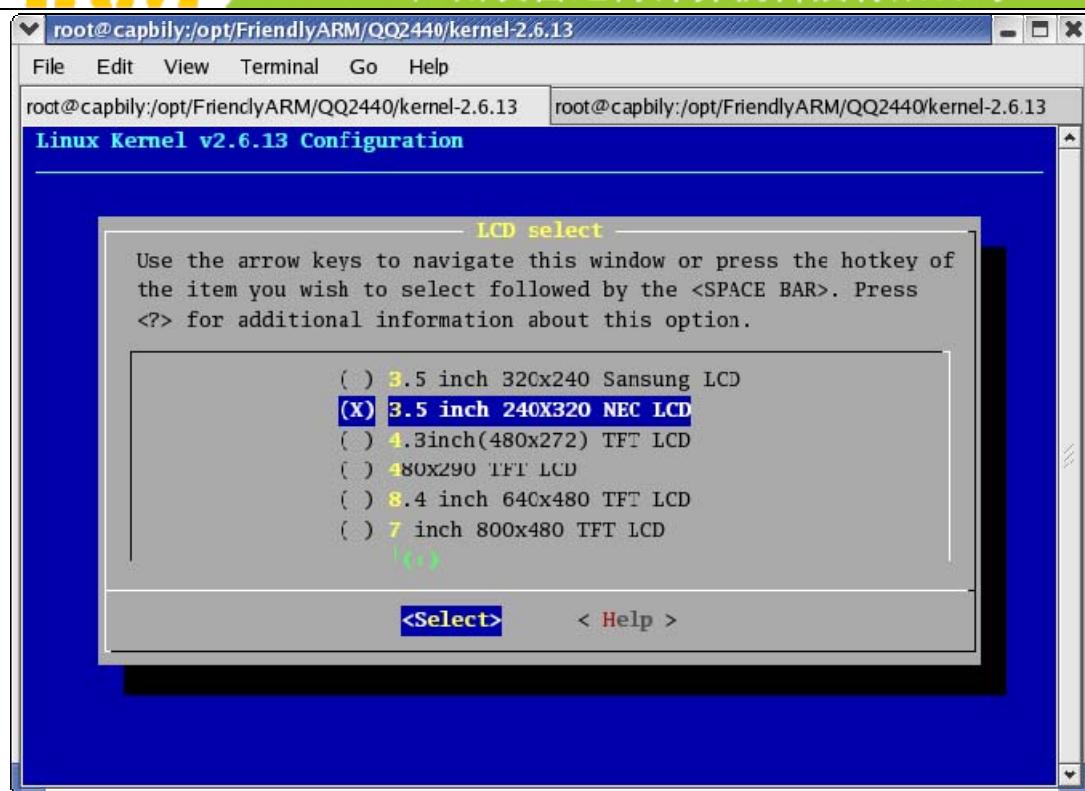


追求卓越 创造精品

TO BE BEST

TO DO GREAT

广州友善之臂计算机科技有限公司



选择完毕，按回车退回到上一层菜单，再选择<Exit>返回 Device Drivers 菜单。

### 8.2.3 如何配置触摸屏

在 Device Drivers 菜单里面，选择 Input device support，按回车进入

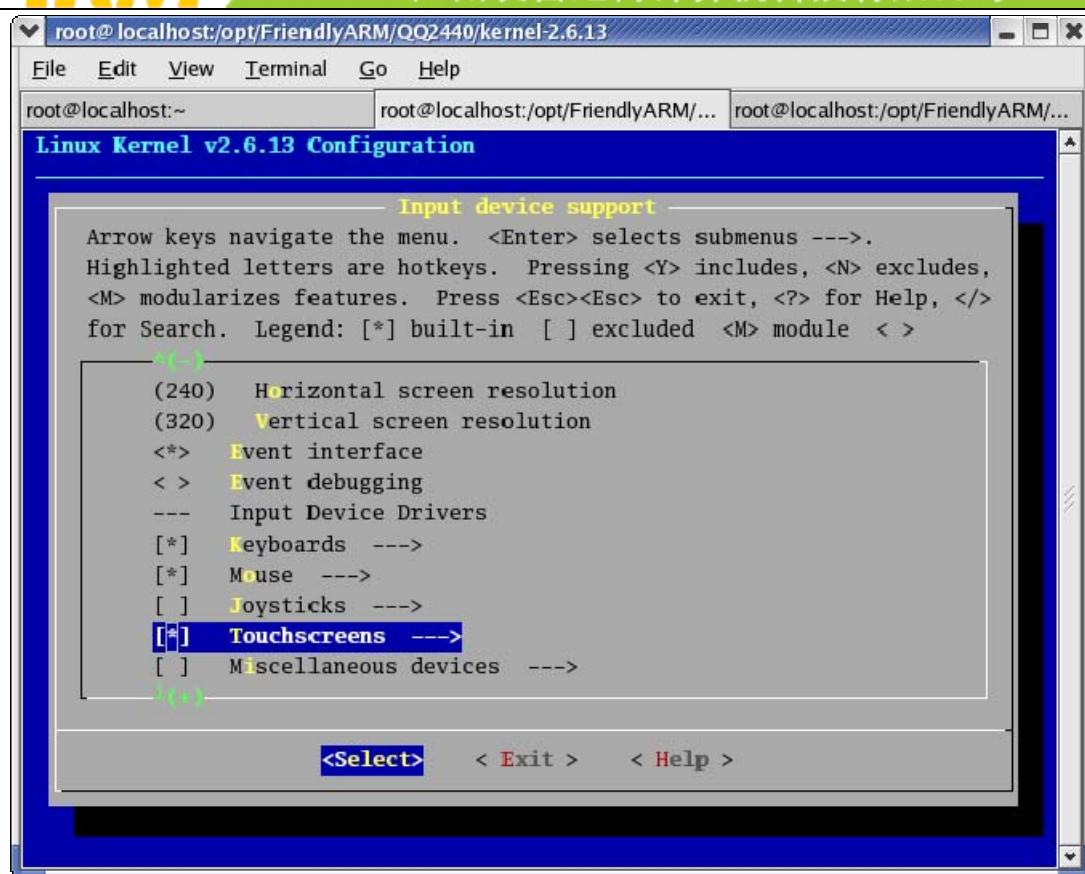


追求卓越 创造精品

TO BE BEST

TO DO GREAT

广州友善之臂计算机科技有限公司



找到并选择 Touchscreens 选项，按回车进入。然后选中里面的

<\*> Touchscreen interface

<\*> Event interface

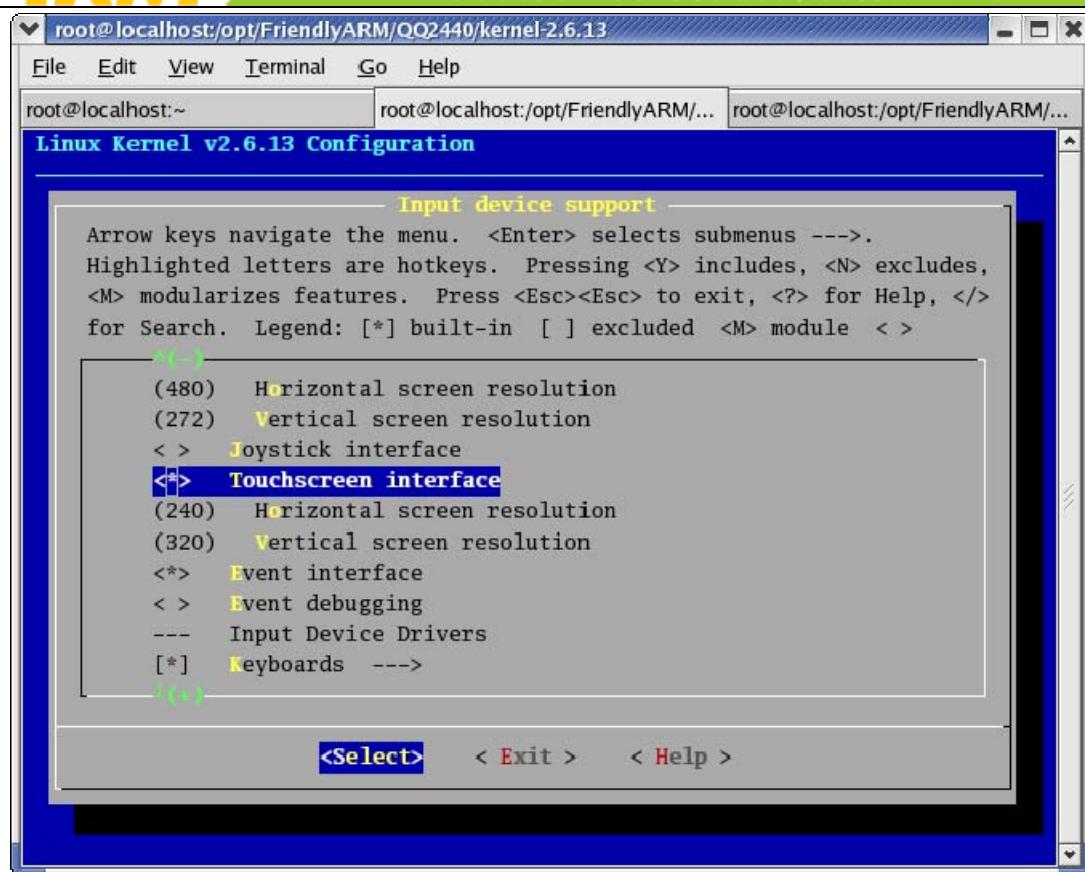


追求卓越 创造精品

TO BE BEST

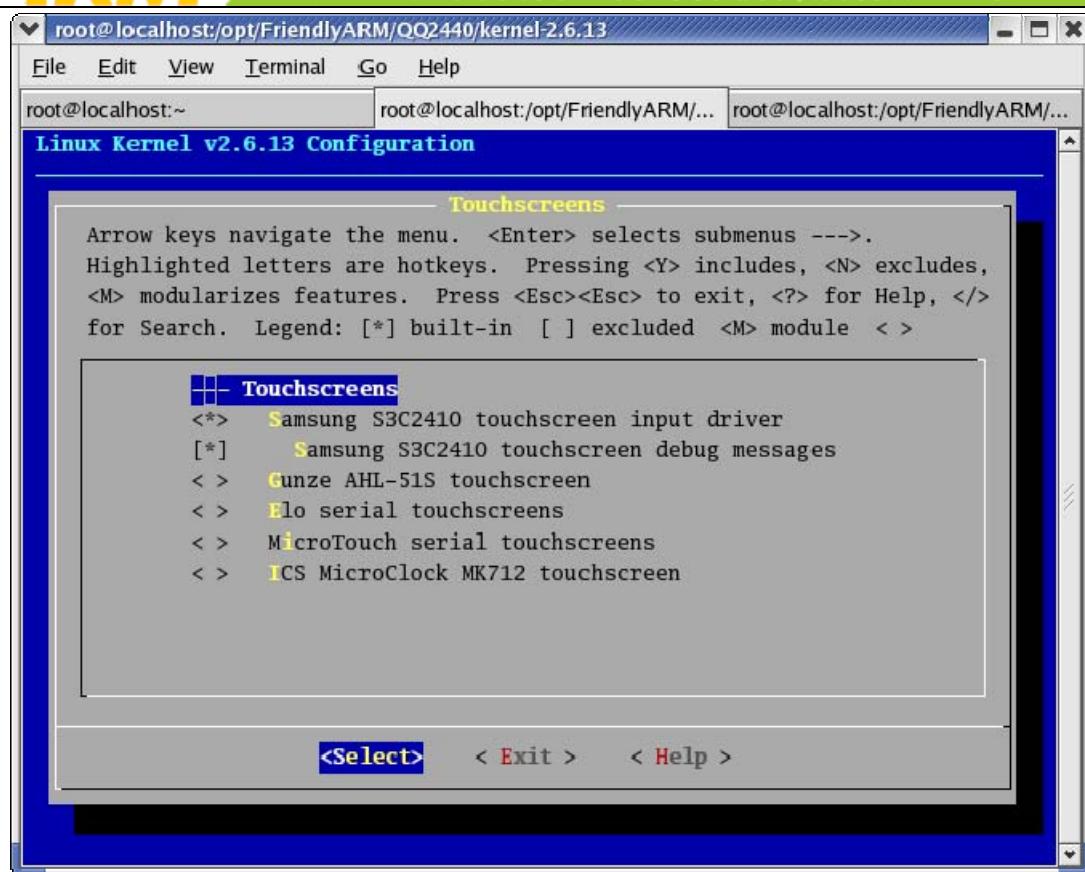
TO DO GREAT

广州友善之臂计算机科技有限公司



选中

<\*>Samsung S3C2410 touchscreen input dirver



再选择<Exit>返回 Input device support 菜单，再选择<Exit>返回 Device Drivers 菜单。

#### 8.2.4 如何配置 USB 鼠标和键盘

在 Device Drivers 菜单里面，找到并选中

<\*>USB supoort

然后回车进入

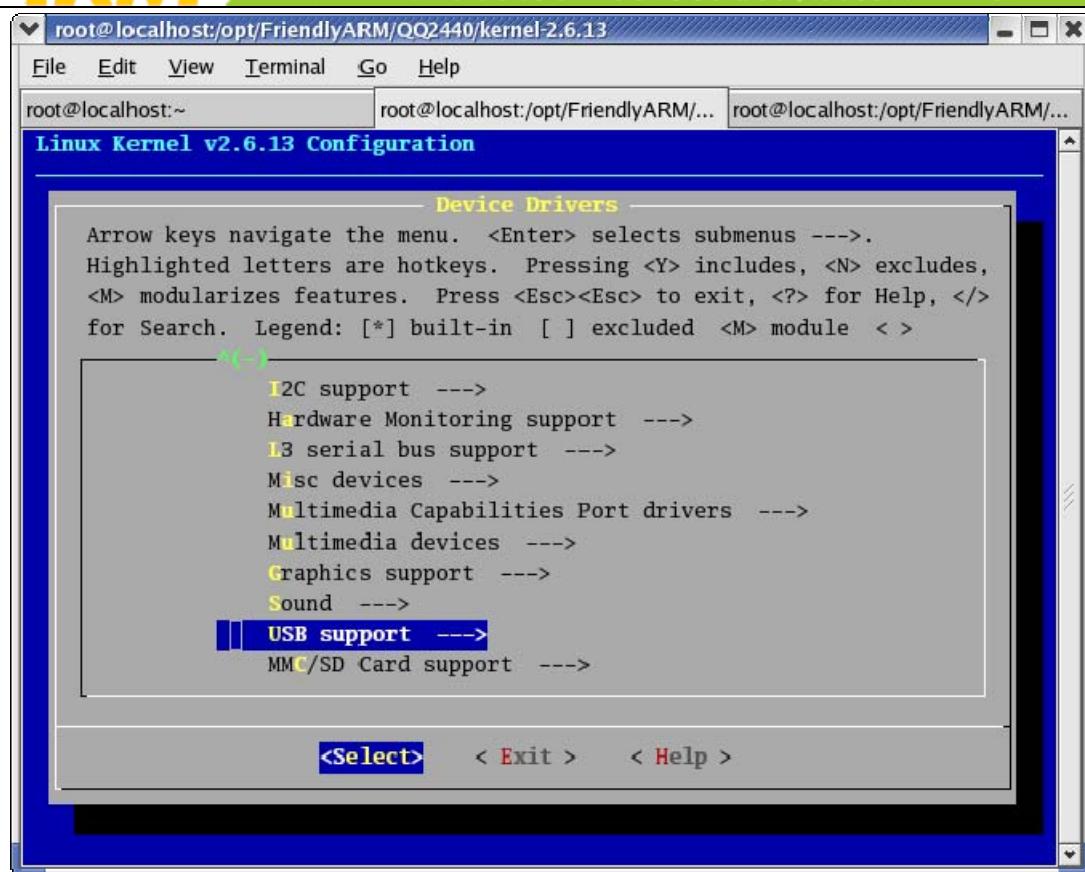


追求卓越 创造精品

TO BE BEST

TO DO GREAT

广州友善之臂计算机科技有限公司



找到并选中：

<\*> Support for Host-side USB

<\*> OHCI HCD support

接着向下移动方向键，寻找 USB 键盘和鼠标部分

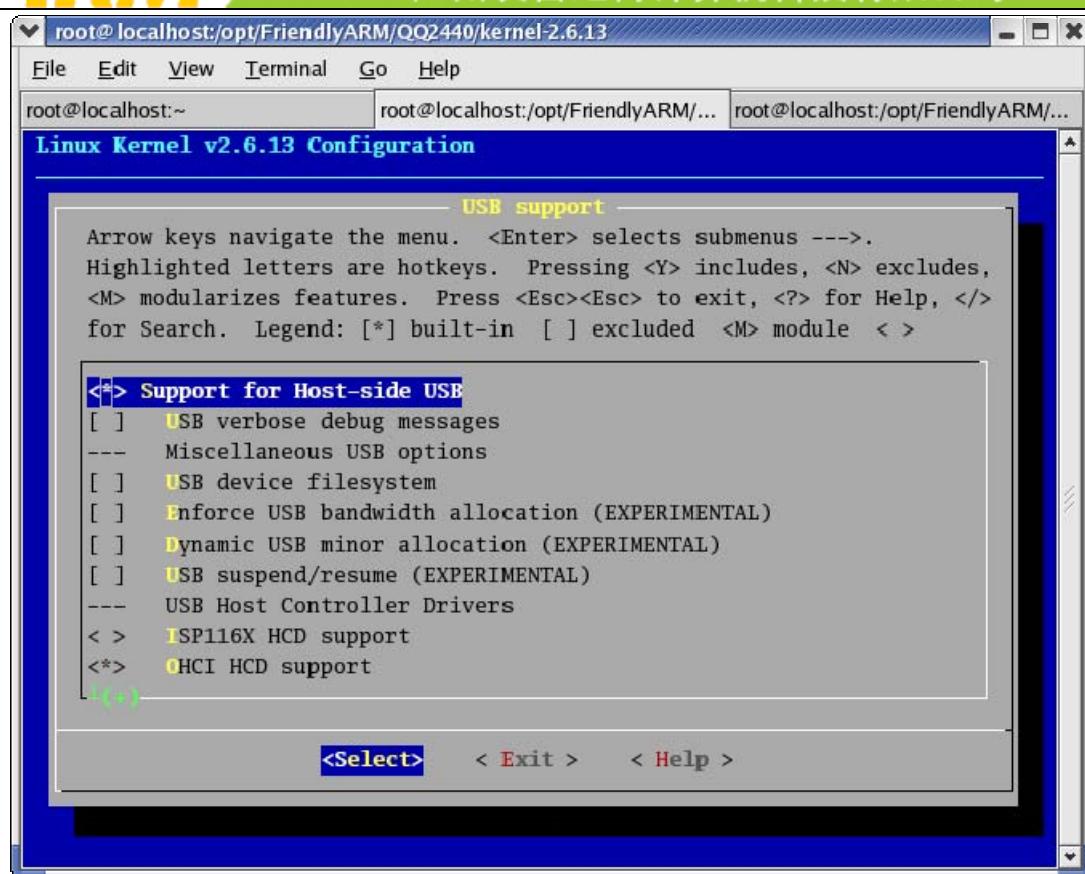


追求卓越 创造精品

TO BE BEST

TO DO GREAT

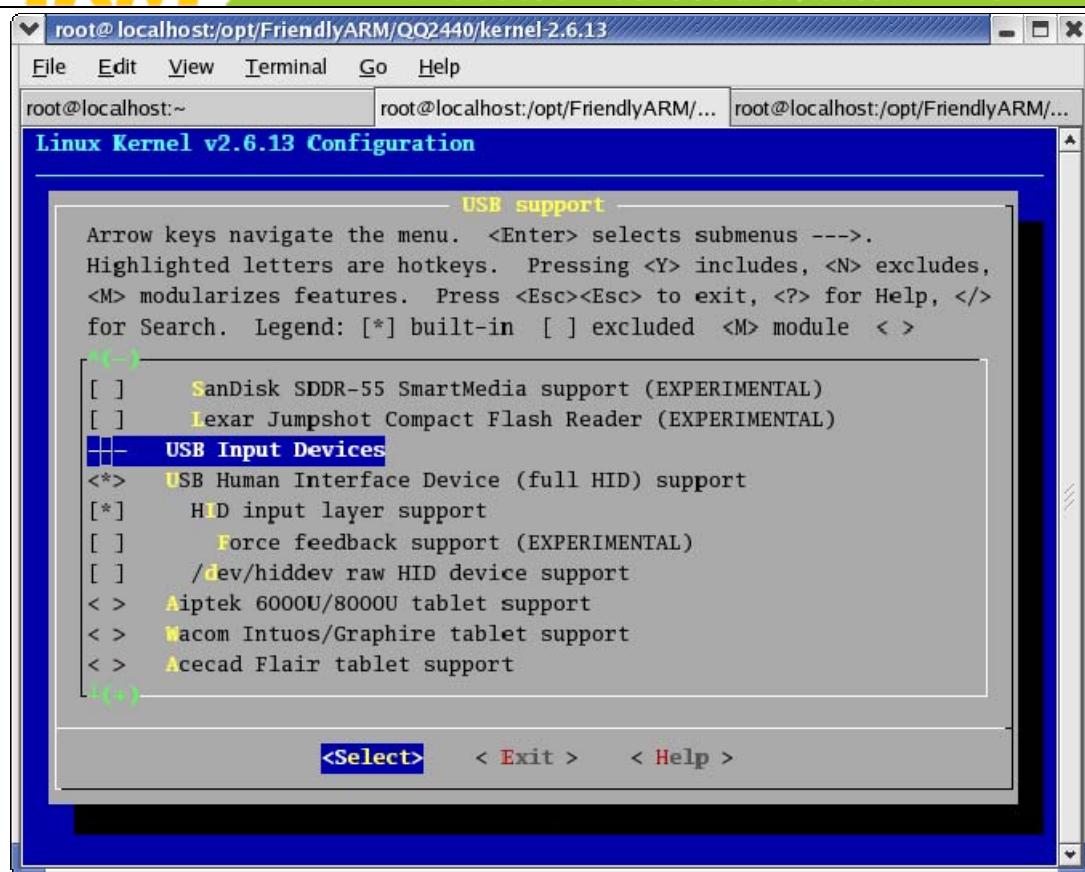
广州友善之臂计算机科技有限公司



在 --- USB Input Devices 部分，我们可以找到并选中：

<\*> USB Human Interface Devices (full HID) support

[\*] HID input layer support



这样就选择配置了 USB 键盘和鼠标，然后选择<Exit>返回 Deice Drivers 菜单。

### 8.2.5 如何配置优盘的支持

因为要优盘用到了 SCSI 命令，所以我们先增加 SCSI 支持。

在 Device Drivers 菜单里面，选择 SCSI device support，按回车进入

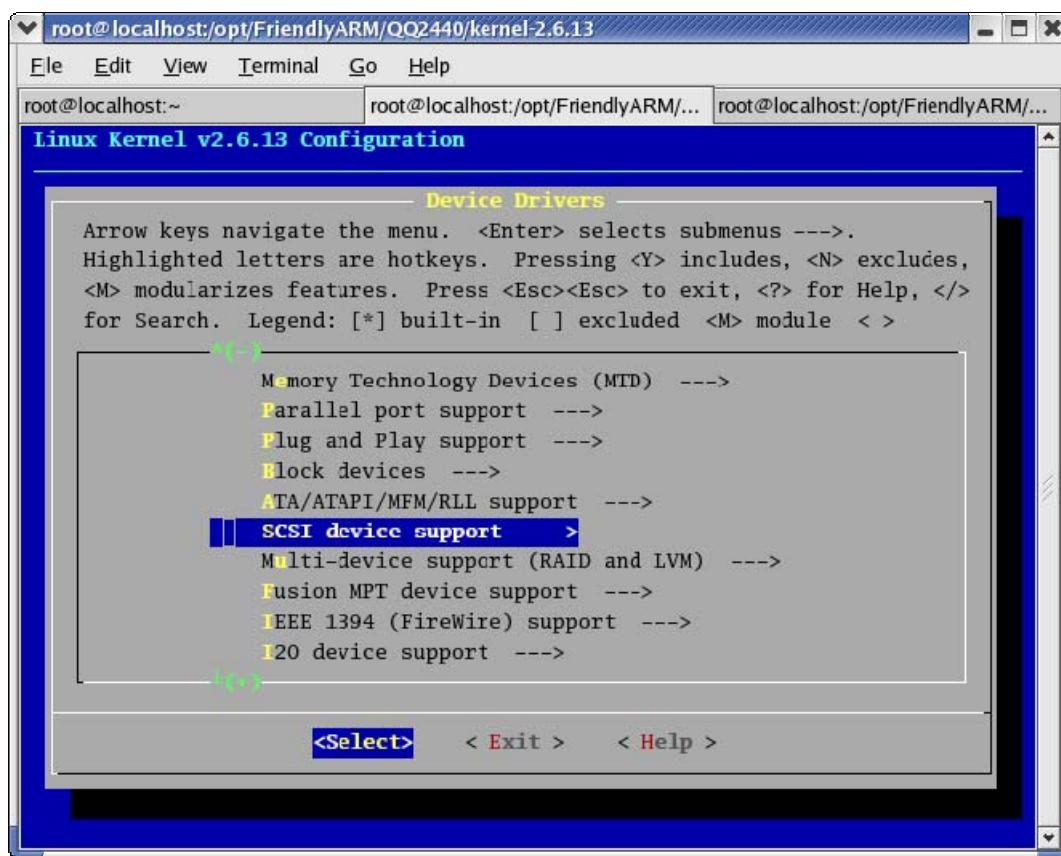


追求卓越 创造精品

TO BE BEST

TO DO GREAT

广州友善之臂计算机科技有限公司



选中：

[\*] legacy /proc/scsi support  
<\*> SCSI disk support

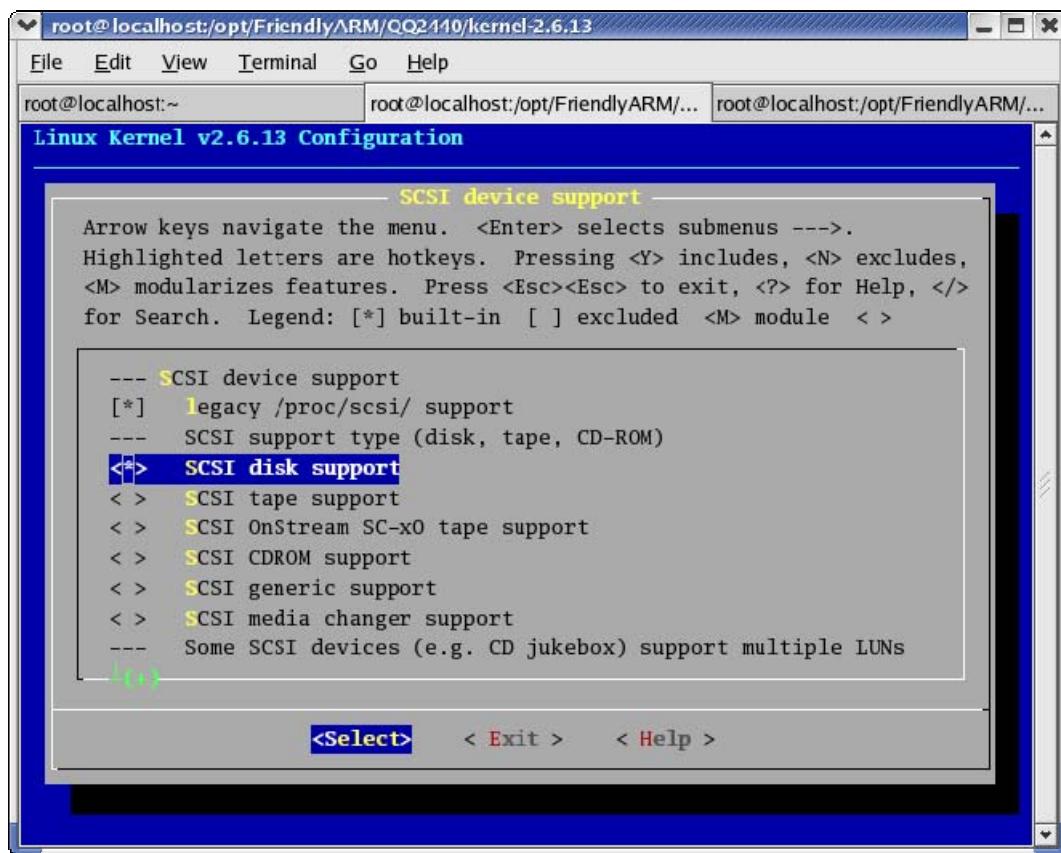


追求卓越 创造精品

TO BE BEST

TO DO GREAT

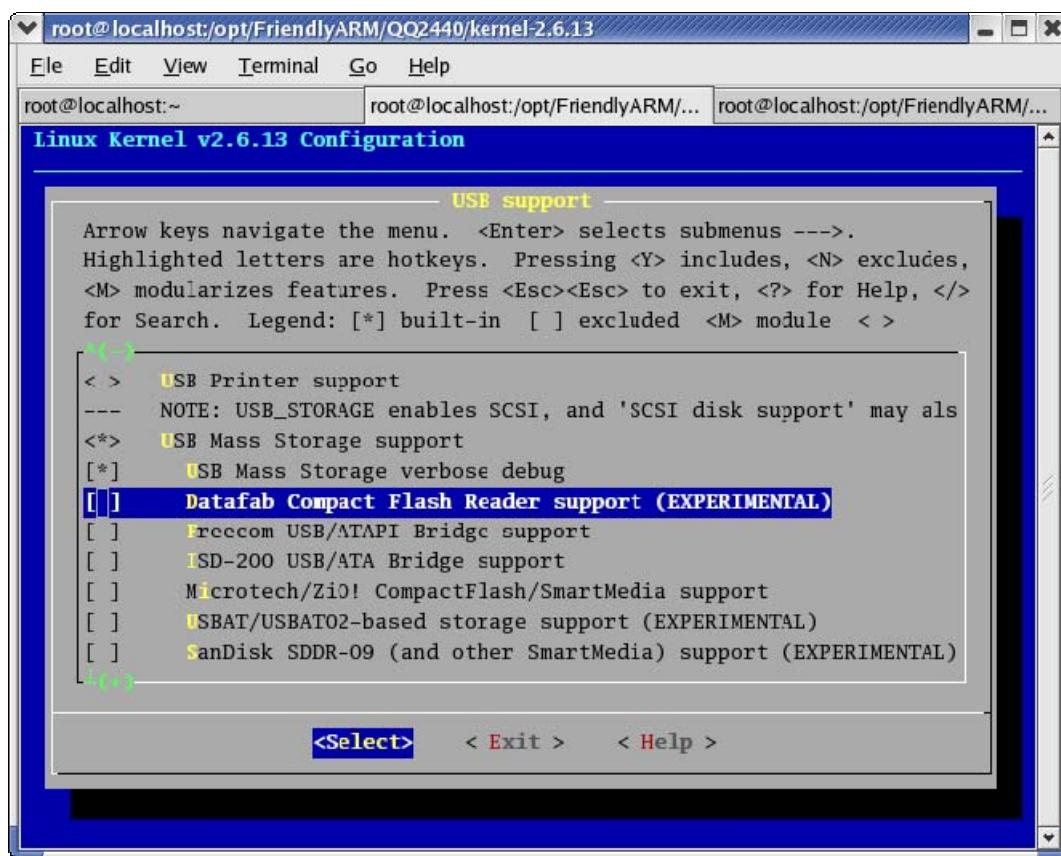
广州友善之臂计算机科技有限公司



然后选择<Exit>返回 Device Drivers 菜单，再选择 USB support，按回车进入 USB support 菜单

找到并选中

<\*> USB Mass Storage support



然后选择<Exit>返回 Device Drivers 菜单

### 8.2.6 如何配置网眼和中芯微等 USB 摄像头

在 Device Drivers 菜单里面，选择 Multimedia devices，回车进入

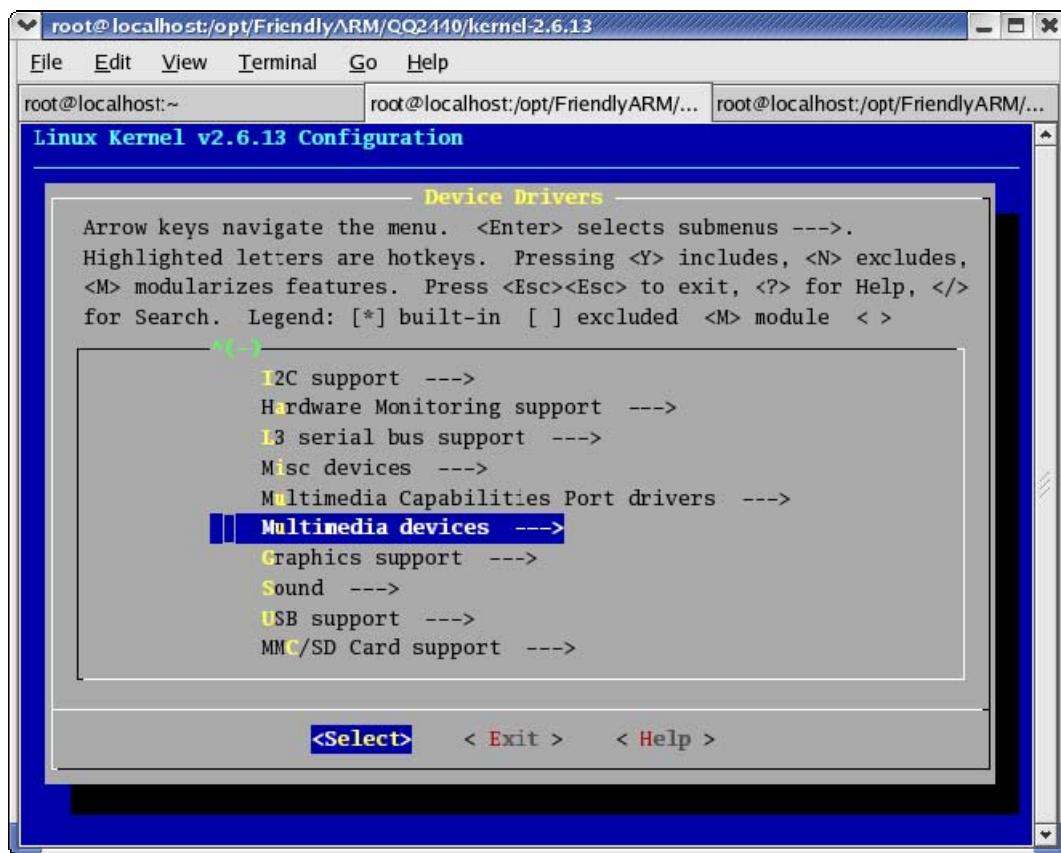


追求卓越 创造精品

TO BE BEST

TO DO GREAT

广州友善之臂计算机科技有限公司



选中

<\*> Video For Linux

并选择 Video For Linux，按回车进入

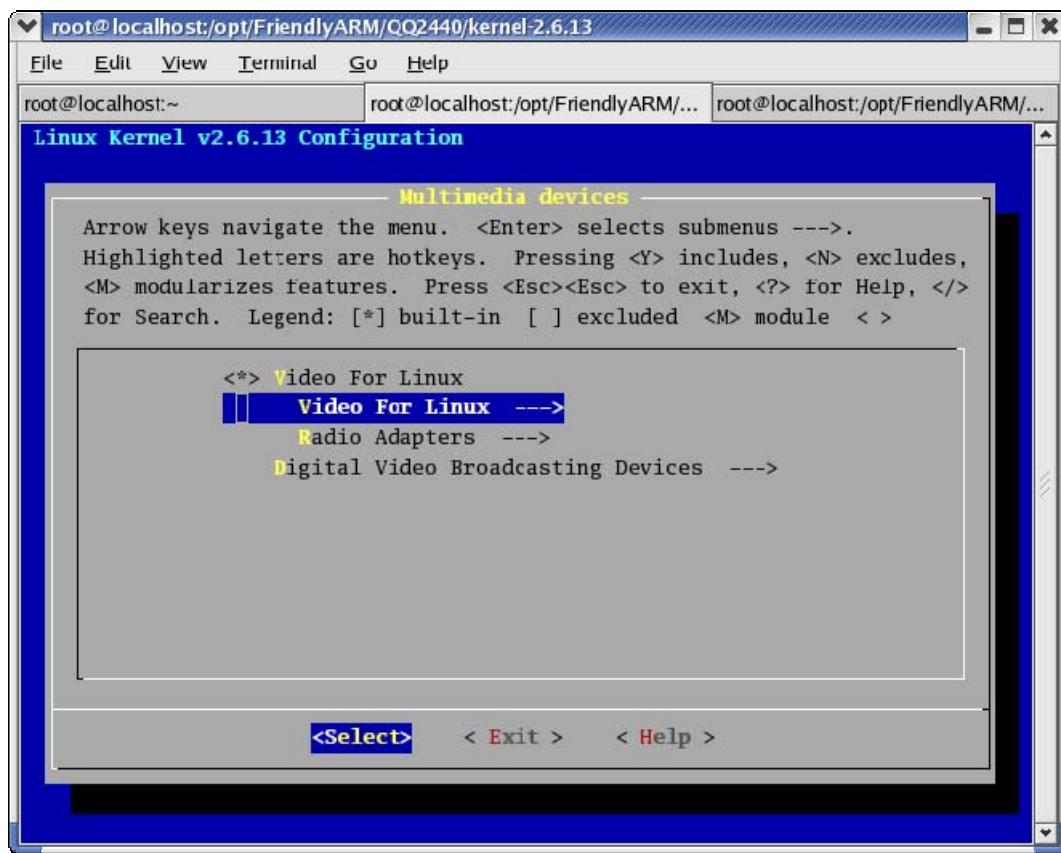


追求卓越 创造精品

TO BE BEST

TO DO GREAT

广州友善之臂计算机科技有限公司



选中

<\*> OmniVision Camera Chip support

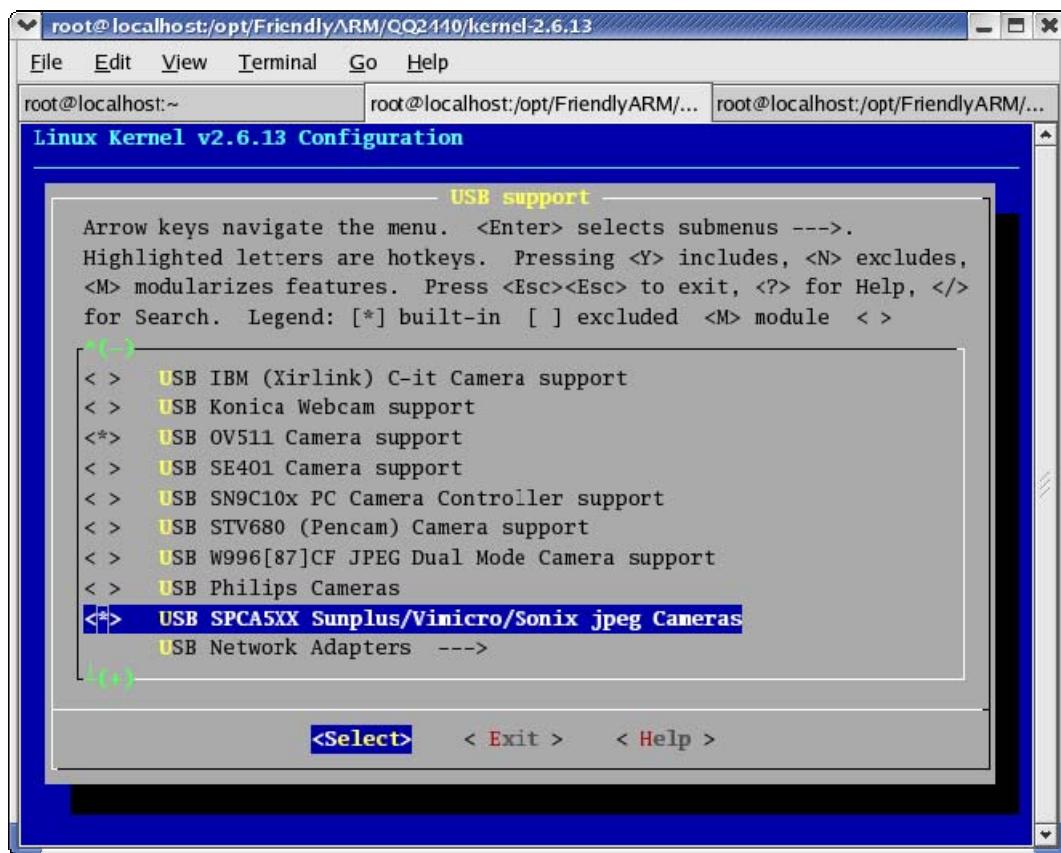
选择<Exit>返回 Multimedia devices，再按<Exit>返回 Device Drivers 菜单。

在 Device Drivers 菜单里面选择 USB support，回车进入，在列表中找到并选中

<\*> USB OV511 Camera support

<\*> USB SPCA5XX Sunplus/Vimicro/Sonix jpeg Cameras

其中 OV511 是支持基于 OV511 芯片的摄像头，SPCA5XX 是支持中芯微 301 系列的摄像头，目前市场上大部分摄像头都是使用该芯片设计的。



选择<Exit>返回 Device Drivers 菜单，再选择<Exit>返回到主菜单。

### 8.2.7 如何配置 CS8900 网卡驱动

要配置 CS8900 网卡驱动，首先要配置网络协议支持  
在主菜单中，选择 Netwoking，回车进入

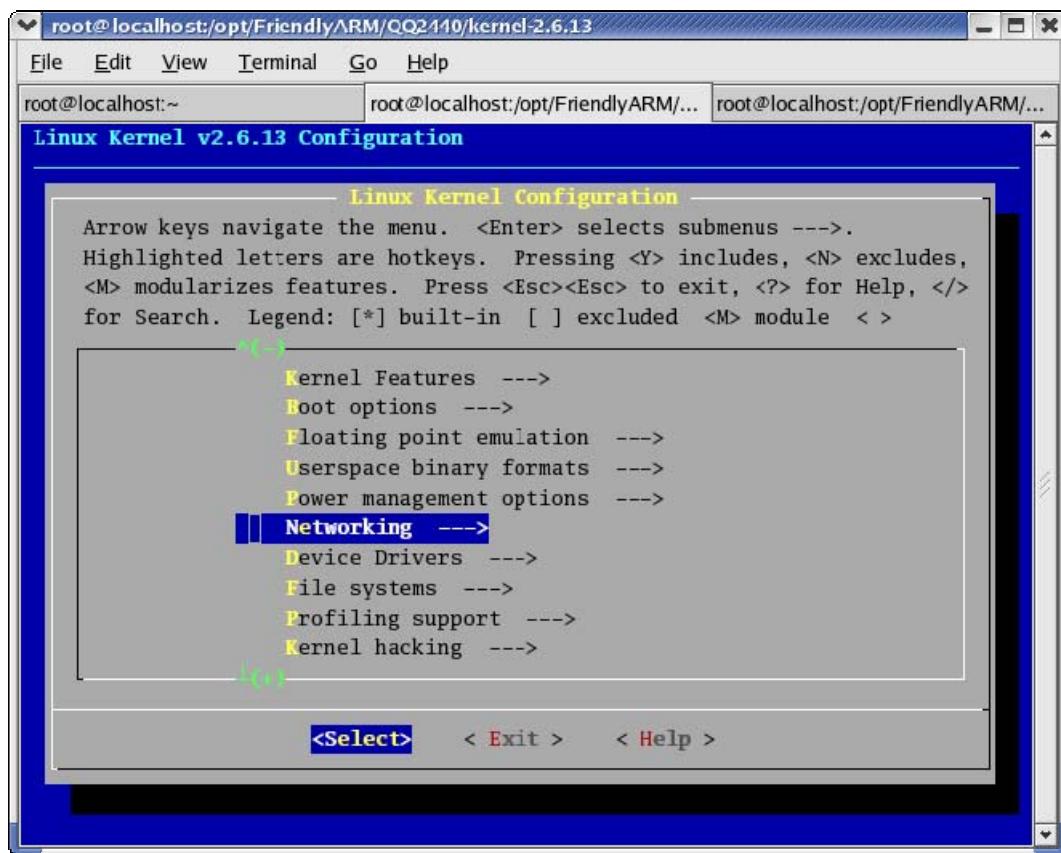


追求卓越 创造精品

TO BE BEST

TO DO GREAT

广州友善之臂计算机科技有限公司



选中

[\*] Networking support

并选择 Networking options，按回车进入

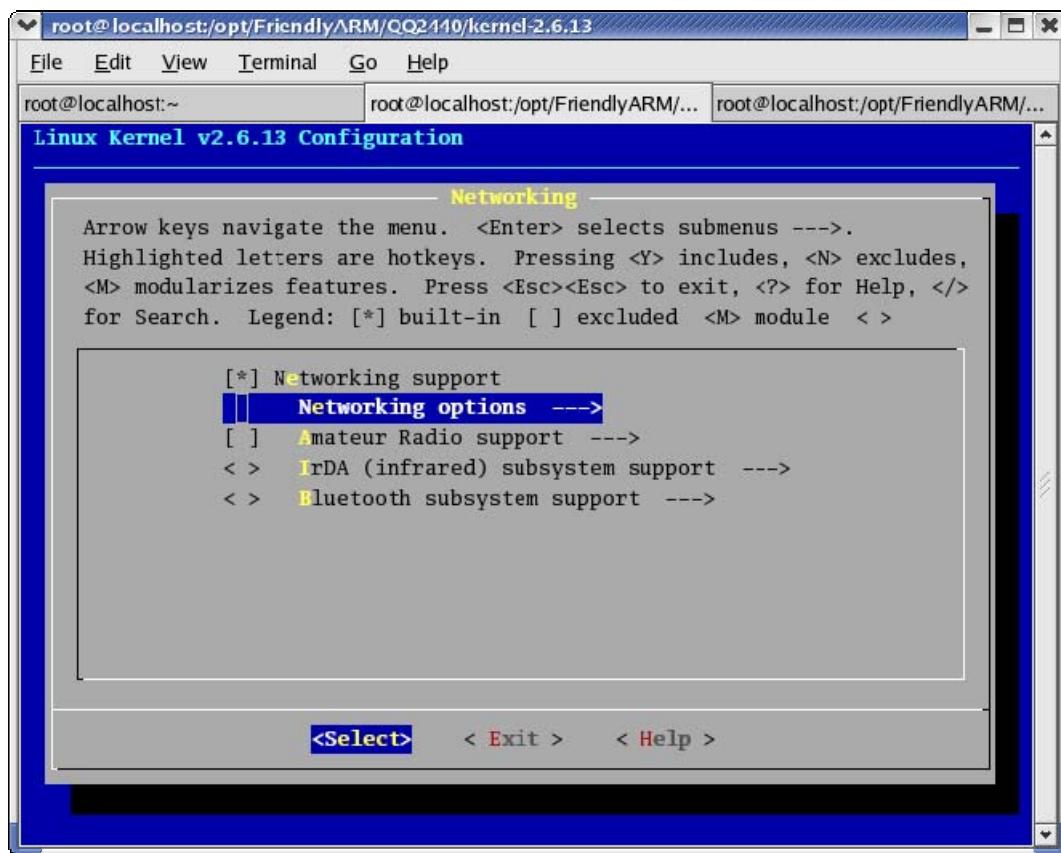


追求卓越 创造精品

TO BE BEST

TO DO GREAT

广州友善之臂计算机科技有限公司



一般我们选择 TCP/IP 协议就够了，但推荐使用我们缺省配置的几个选项，如图

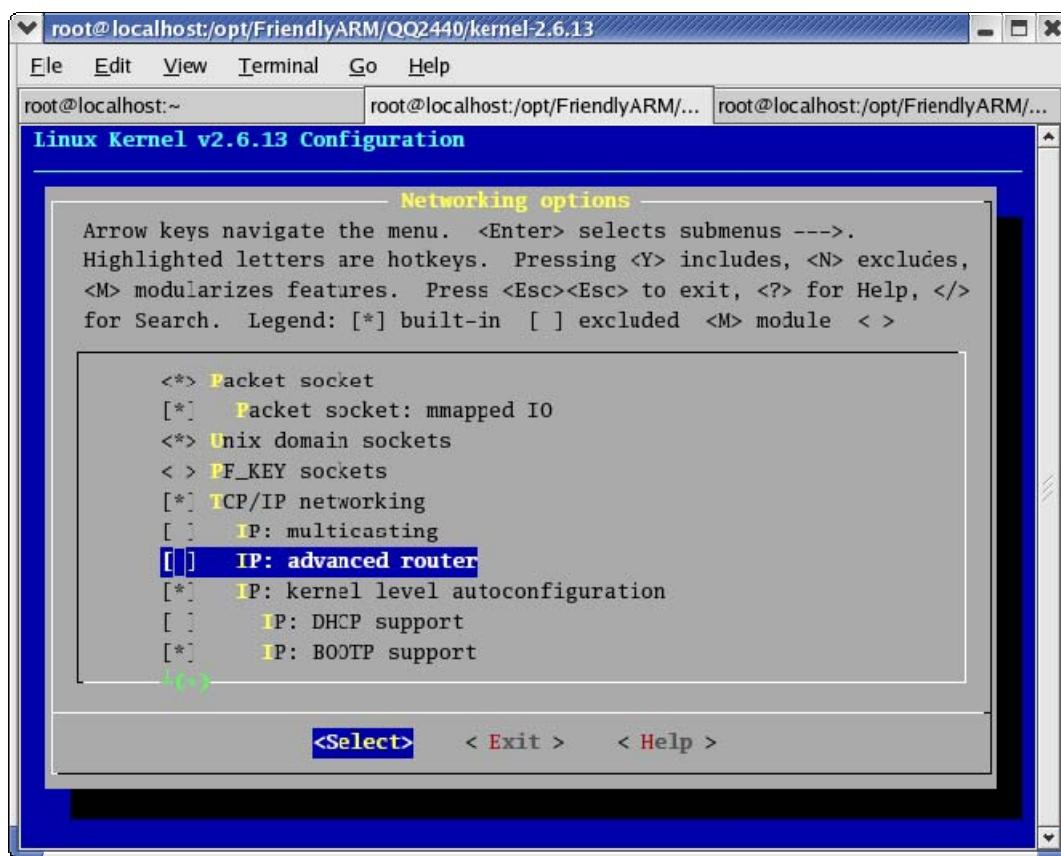


追求卓越 创造精品

TO BE BEST

TO DO GREAT

广州友善之臂计算机科技有限公司



选择完毕，一直退回到主菜单，并选择进入 Device Drivers 菜单。  
找到 Network device support，选择进入

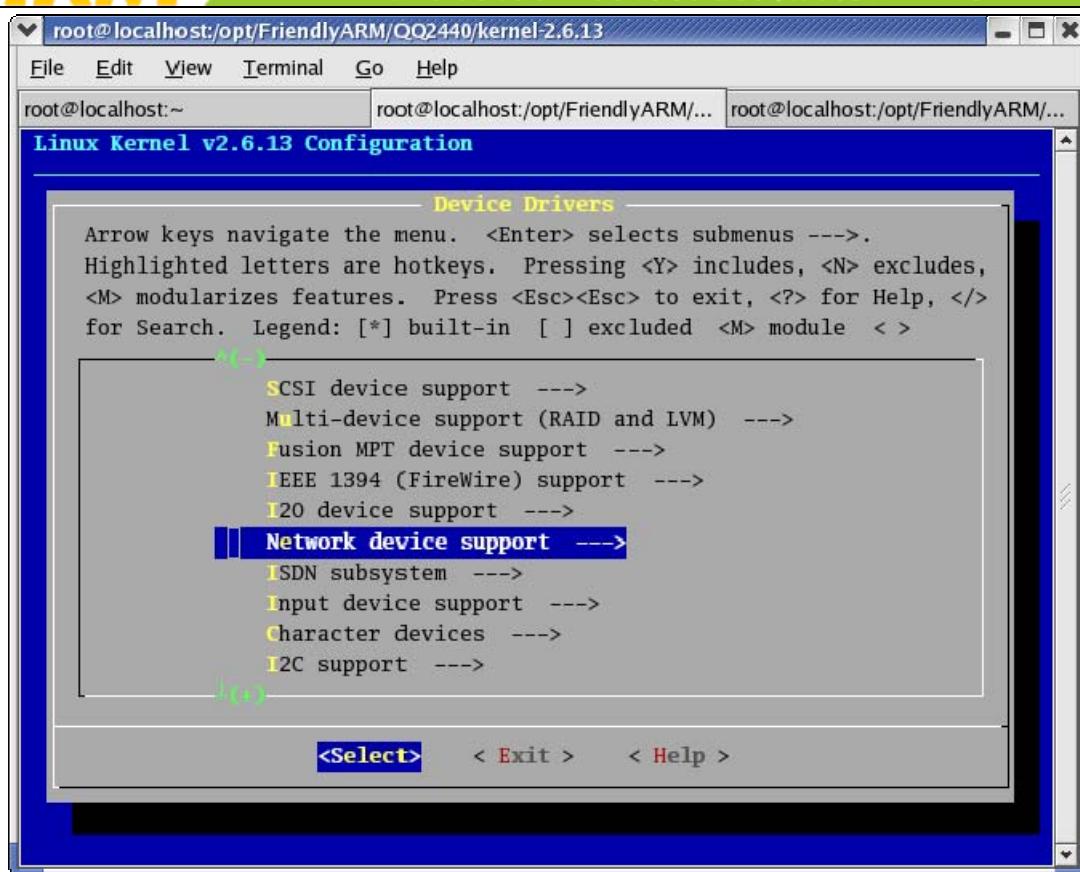


追求卓越 创造精品

TO BE BEST

TO DO GREAT

广州友善之臂计算机科技有限公司



找到并进入 Ethernet (10 or 100Mbit) 选项

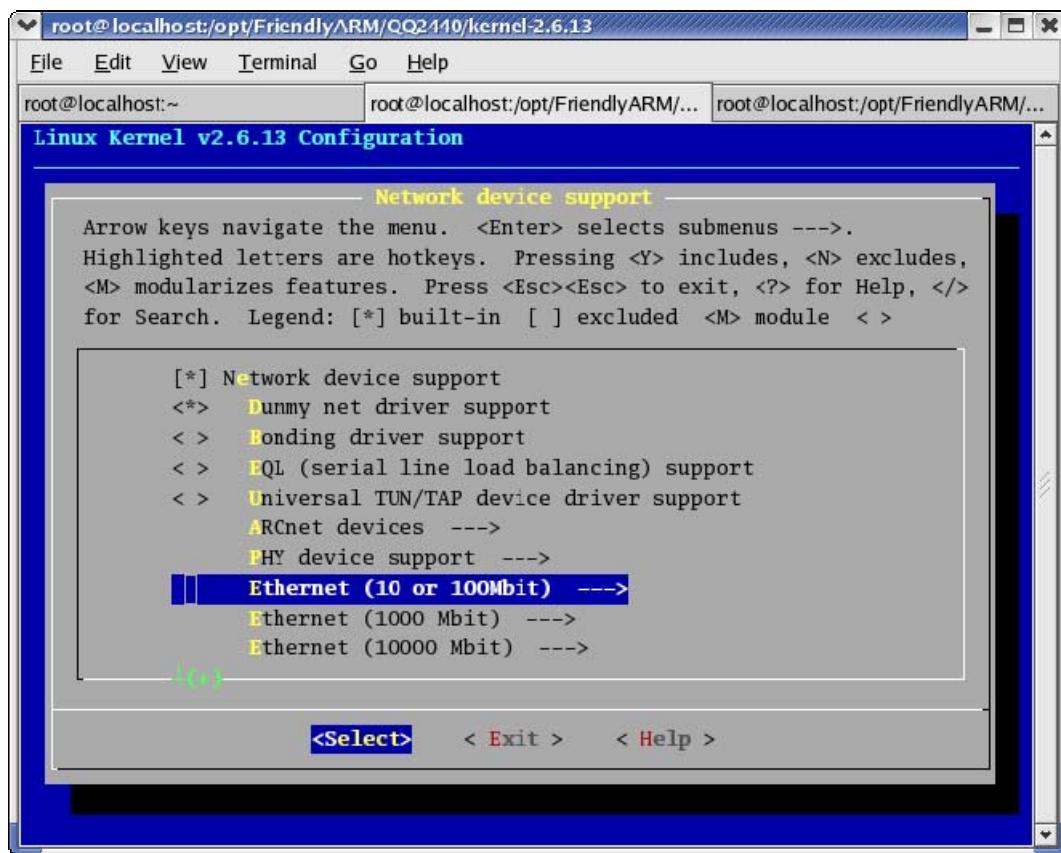


追求卓越 创造精品

TO BE BEST

TO DO GREAT

广州友善之臂计算机科技有限公司



选中：

<\*> CS8900 support

<\*> Generic Media Independent Interface device support

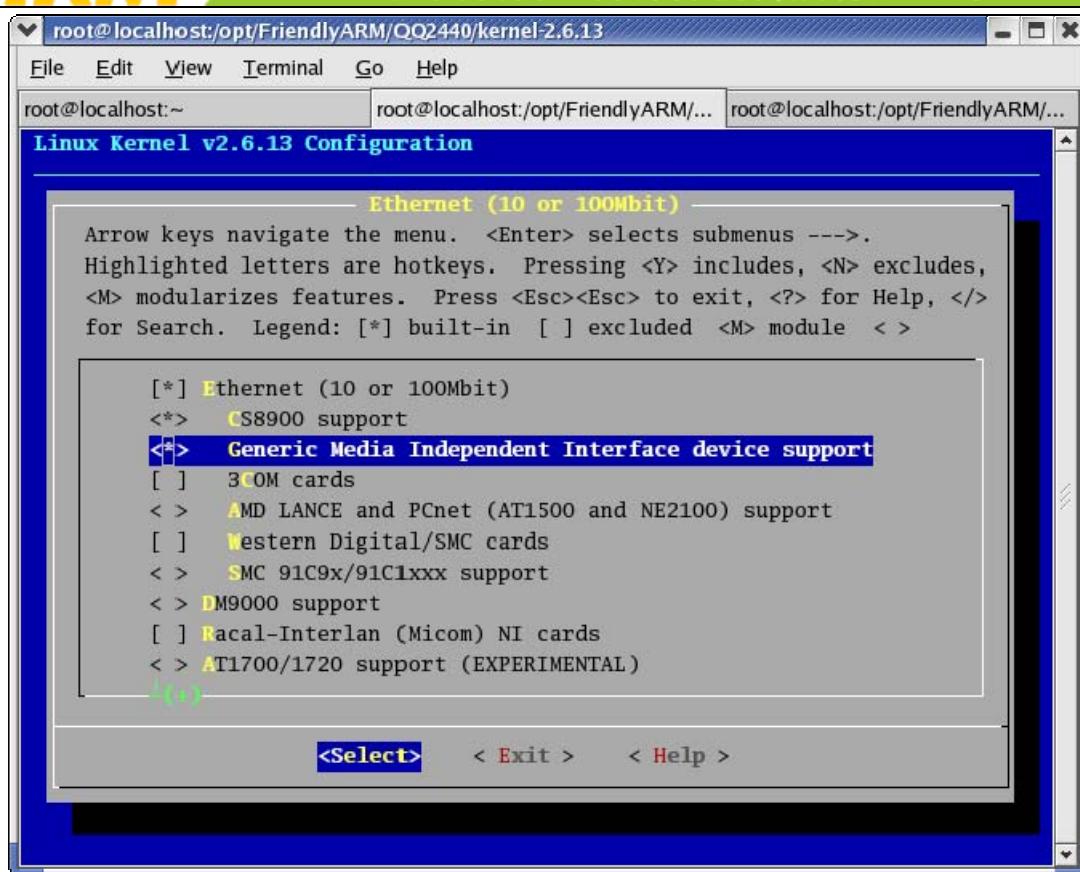


追求卓越 创造精品

TO BE BEST

TO DO GREAT

广州友善之臂计算机科技有限公司



选择<Exit>一直返回到 Device Drivers 菜单。

## 8.2.8 如何配置声卡驱动

在 Device Drivers 菜单中，选择 L3 serial bus supprt，并进入

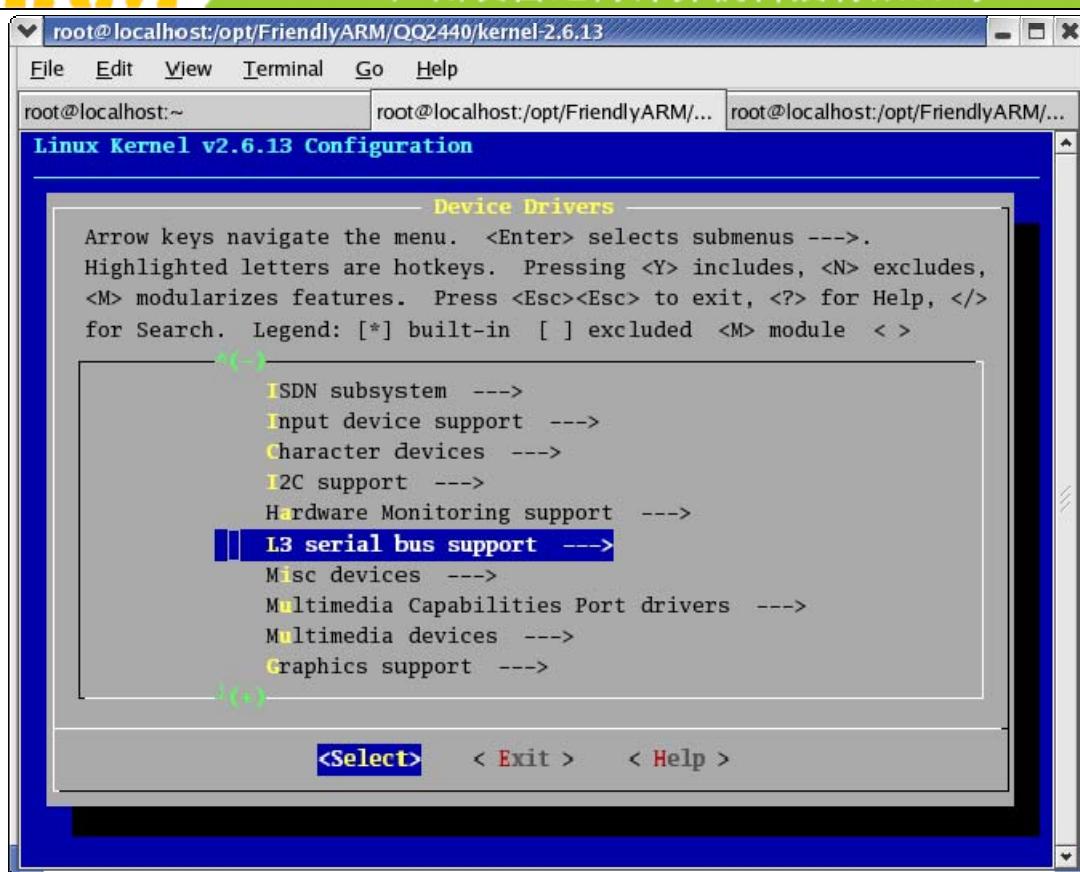


追求卓越 创造精品

TO BE BEST

TO DO GREAT

广州友善之臂计算机科技有限公司



选中里面的所有选项，如图

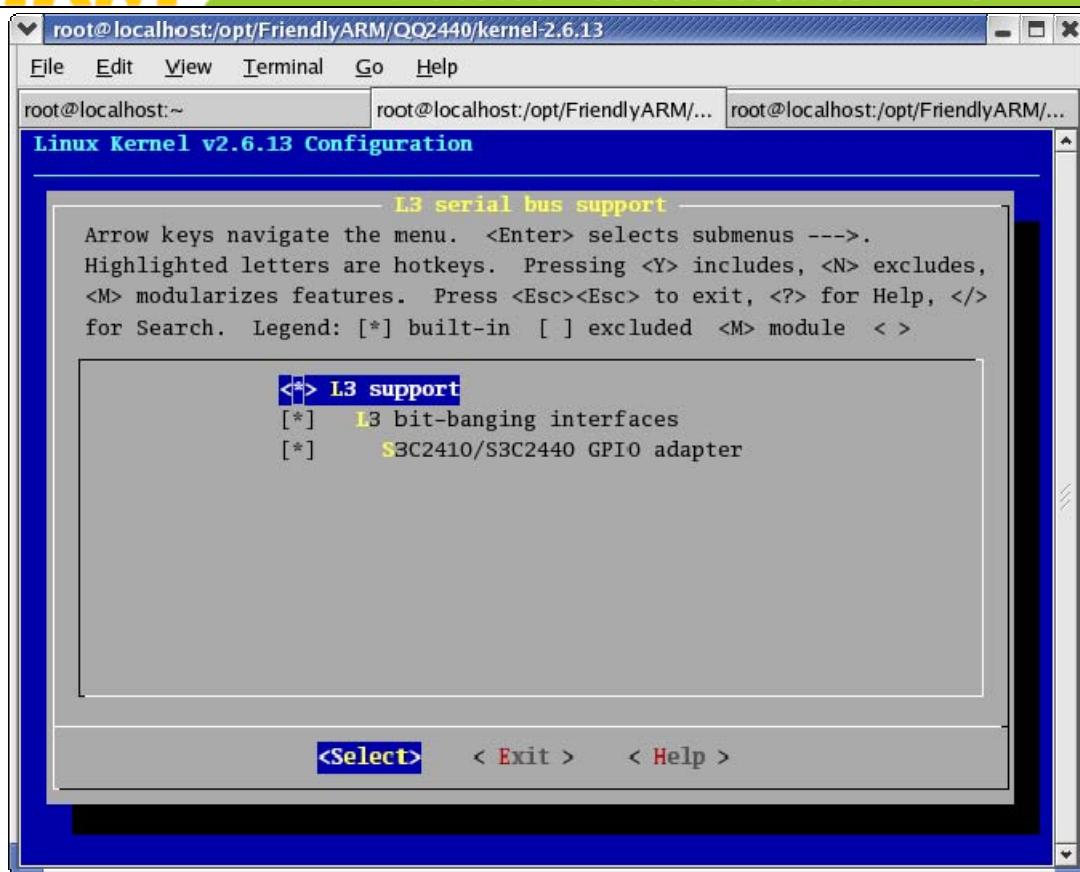


追求卓越 创造精品

TO BE BEST

TO DO GREAT

广州友善之臂计算机科技有限公司



返回到 Device Drivers 菜单，并选择 Sound ---> 进入

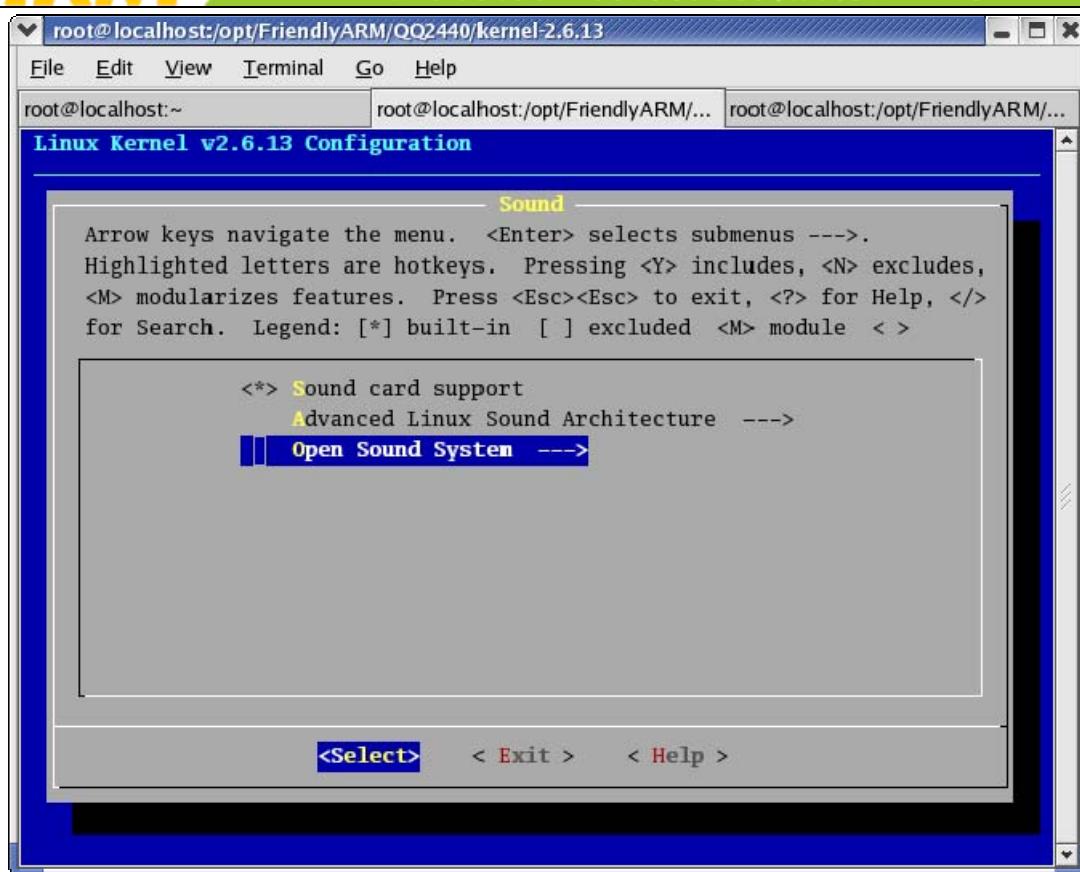


追求卓越 创造精品

TO BE BEST

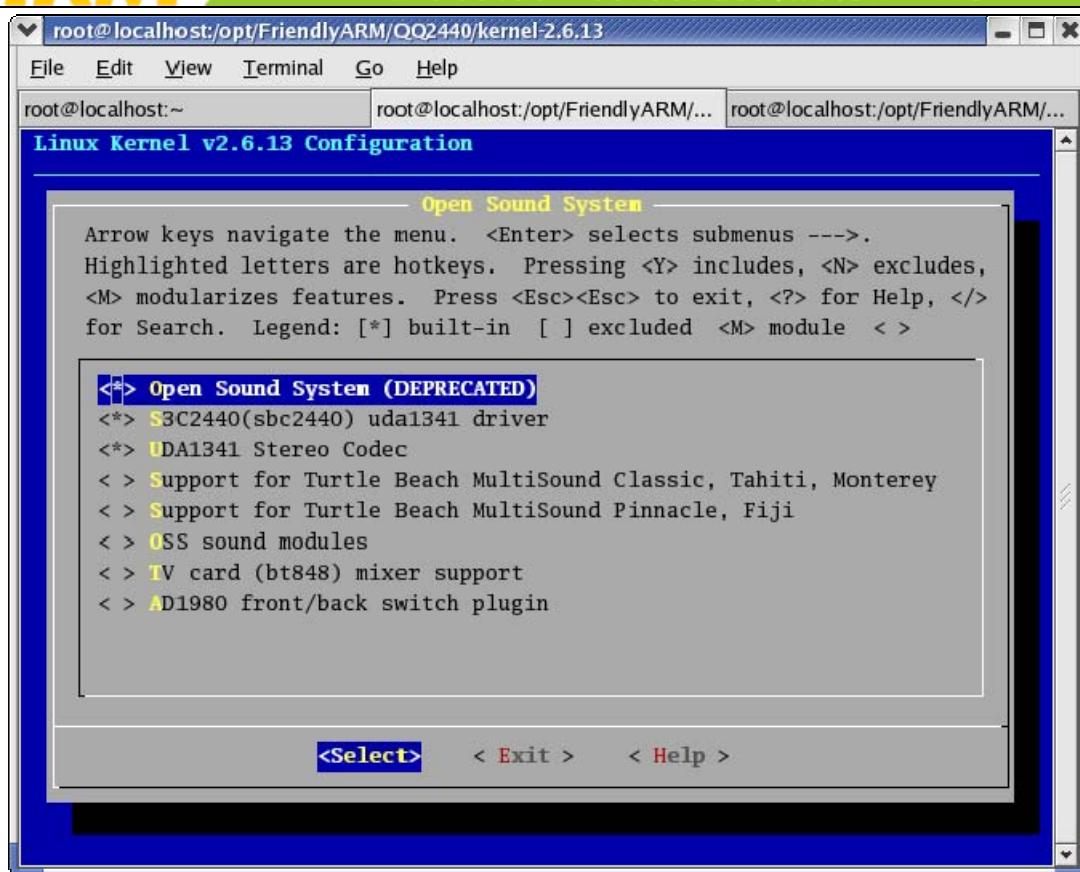
TO DO GREAT

广州友善之臂计算机科技有限公司



选择进入 Open Sound System - - ->

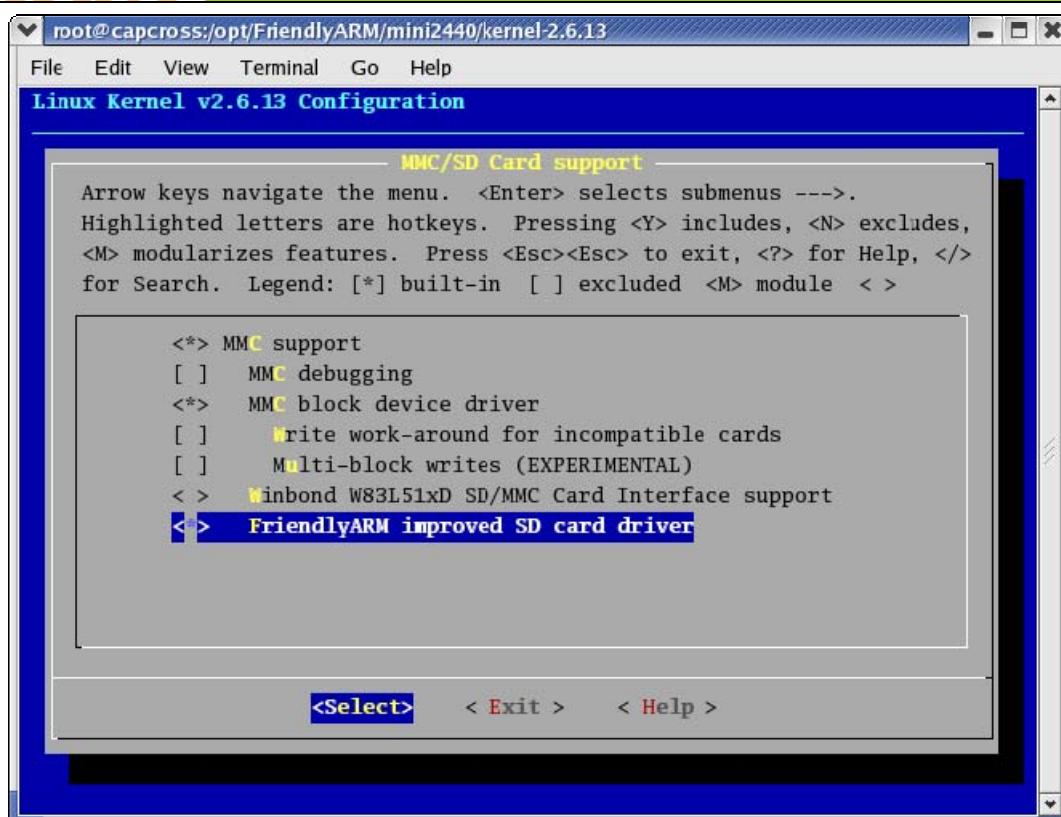
选中如图所示选项



返回到 Device Drivers 菜单

### 8.2.9 如何配置 SD/MMC 卡驱动

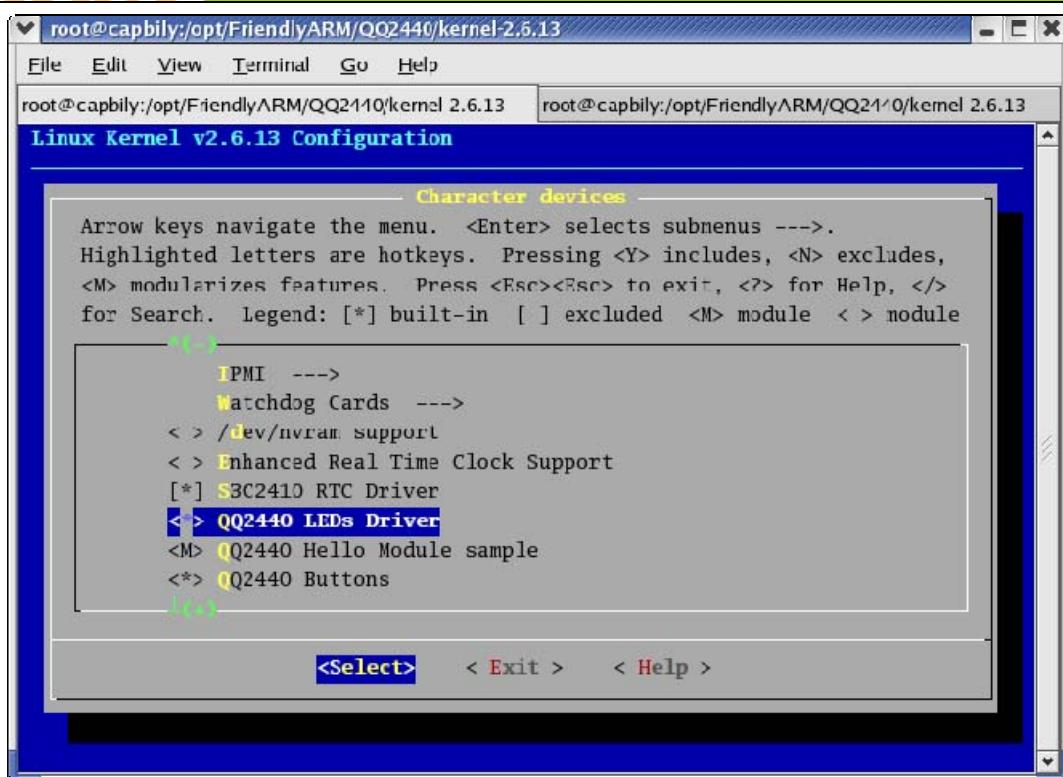
在 Device Drivers 菜单中，选择进入 MMC/SD Card support - - ->，并做如图选择



返回到 Device Drivers 菜单。

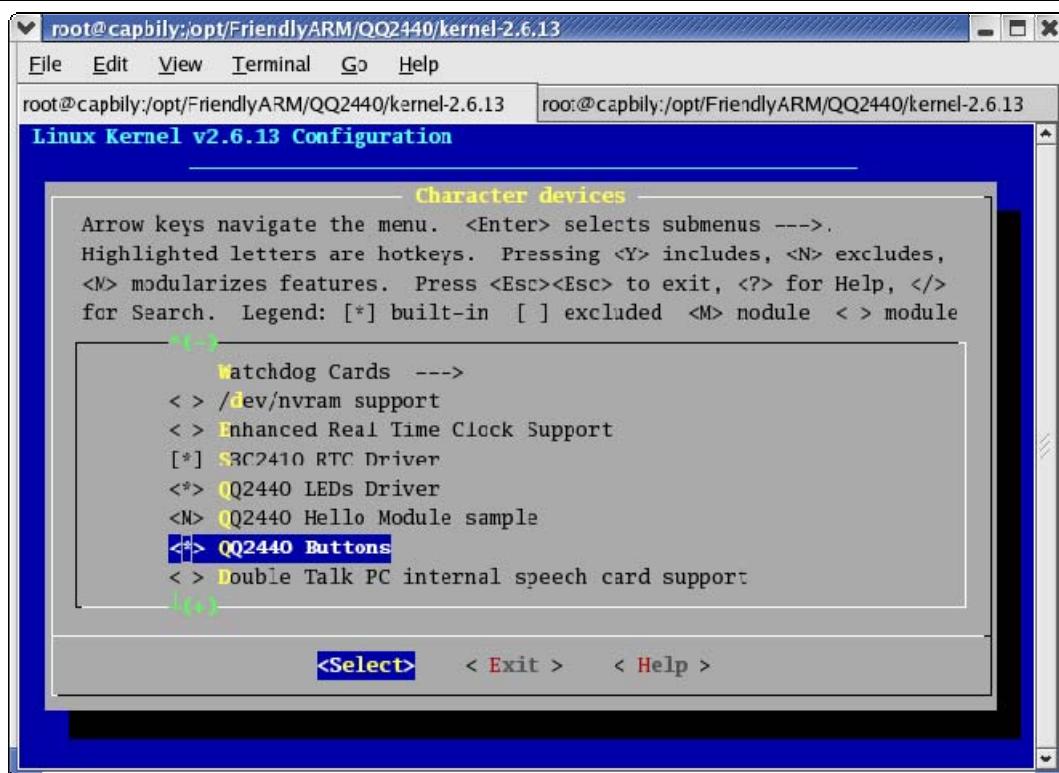
### 8.2.10 如何配置 LED 驱动

在 Device Drivers 菜单中，选择进入 Character devices - - ->，找到并选中 LEDs 驱动支持，如图。



### 8.2.11 如何配置按键驱动

在 Device Drivers 菜单中，选择进入 Character devices - - ->，找到并选中 QQ2440 Buttons 驱动支持，如图。



### 8.2.12 如何配置串口驱动

依然在 Character devices 菜单中，选择进入 Serial drivers - - ->，并做如图选择。

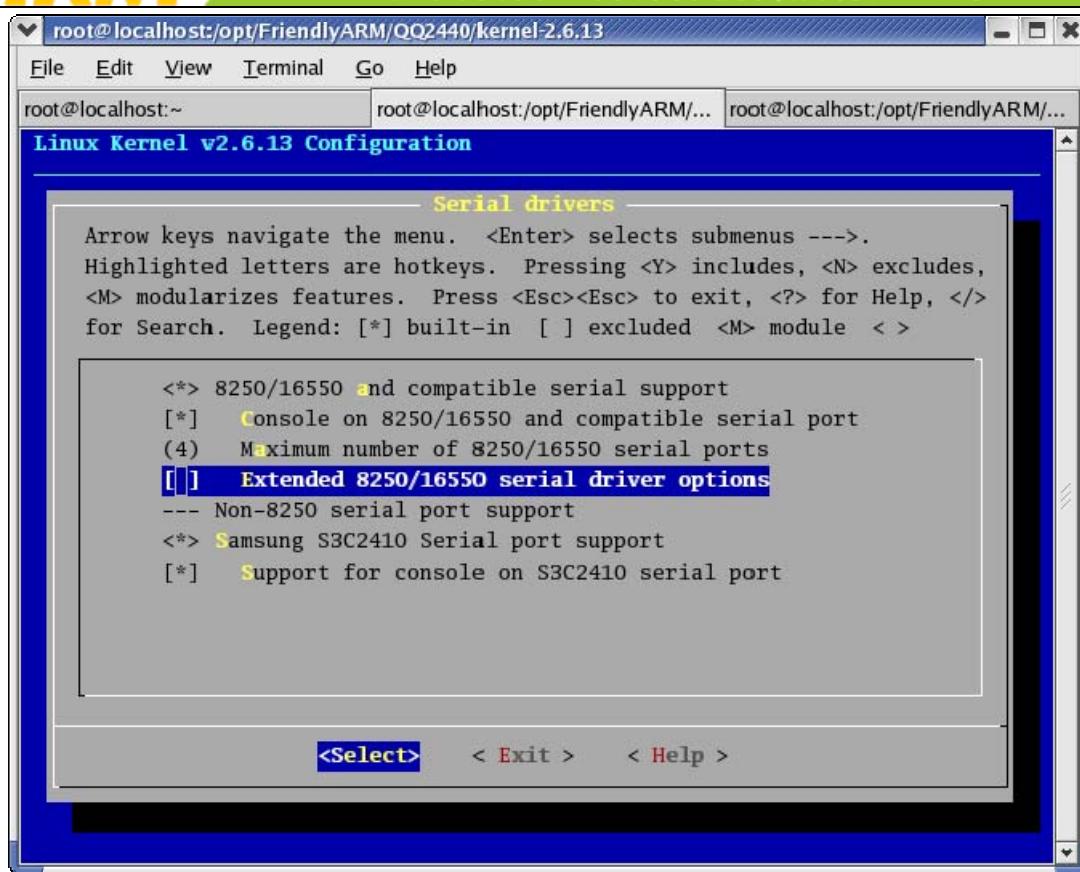


追求卓越 创造精品

TO BE BEST

TO DO GREAT

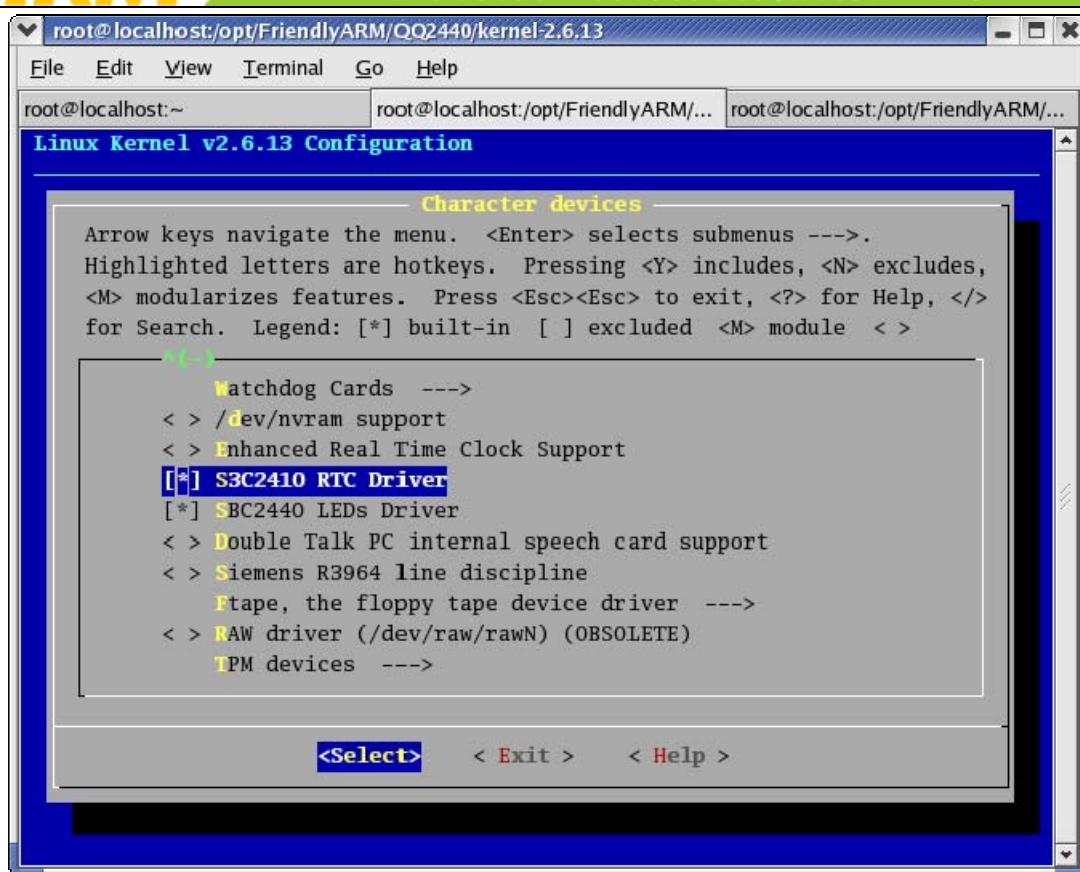
广州友善之臂计算机科技有限公司



### 8.2.13 如何配置 RTC 实时时钟驱动

依然在 Character devices 菜单中，选中 RTC 驱动支持，如图。

**注意：不要选中 Enhanced Real Time Clock Support**



返回到主菜单。

### 8.2.14 如何配置 yaffs 文件系统的支持

选择进入 File Systems --->菜单，再选择进入 Miscellaneous filesystems --->，如图：

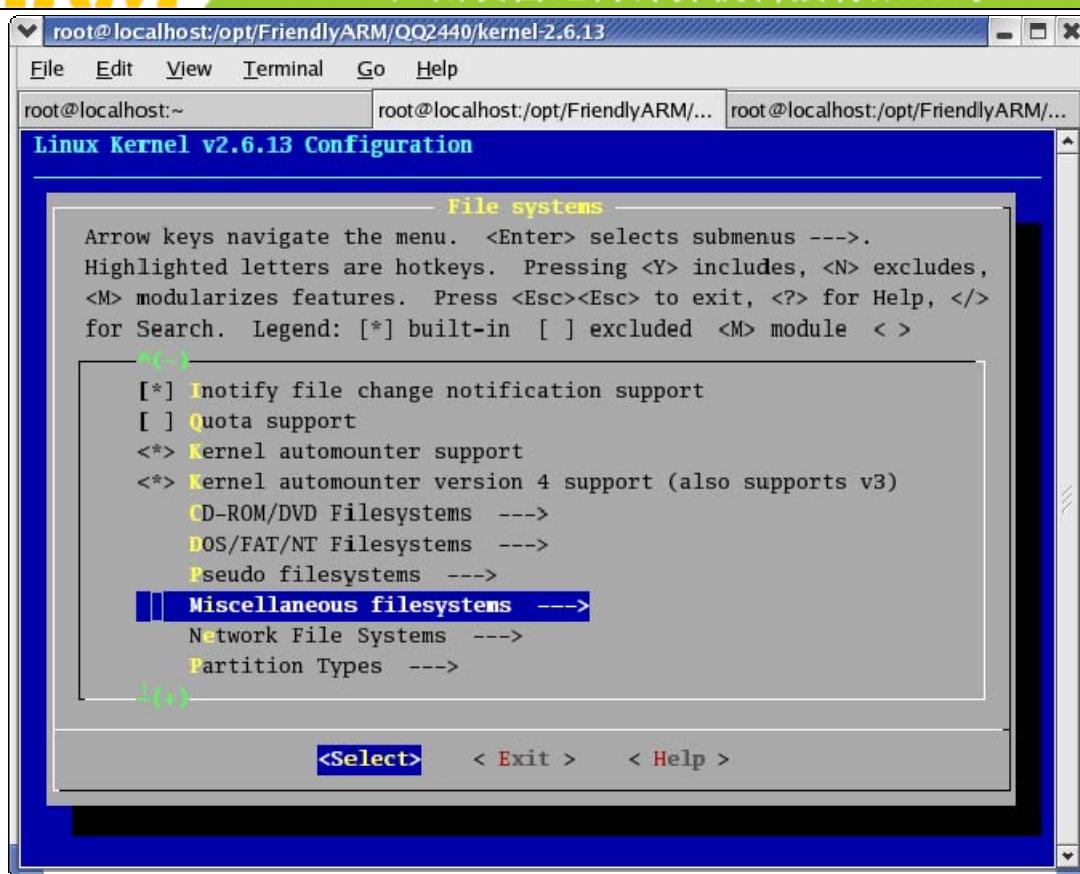


追求卓越 创造精品

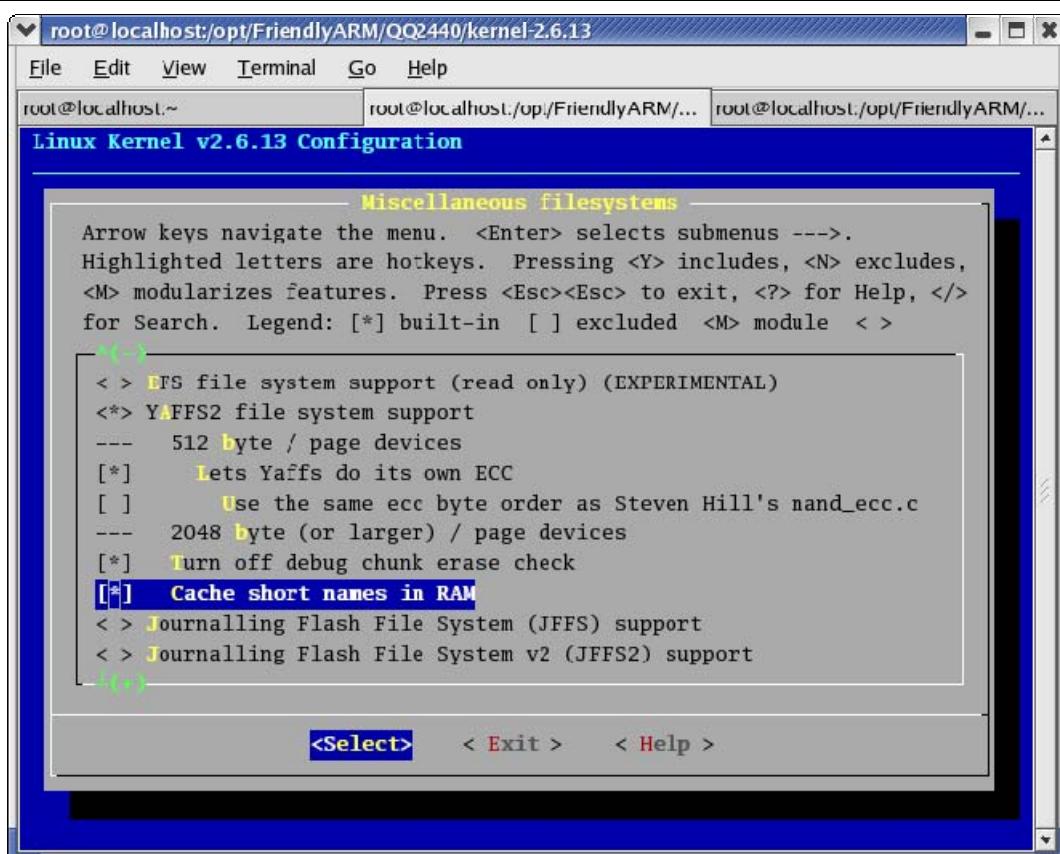
TO BE BEST

TO DO GREAT

广州友善之臂计算机科技有限公司



如图选择 yaffs 文件系统相关选项。



返回到 File systems 菜单。

### 8.2.15 如何配置 EXT2/VFAT/ NFS 等文件系统

在 File System 菜单中，如图选择 EXT2 文件系统的支持。

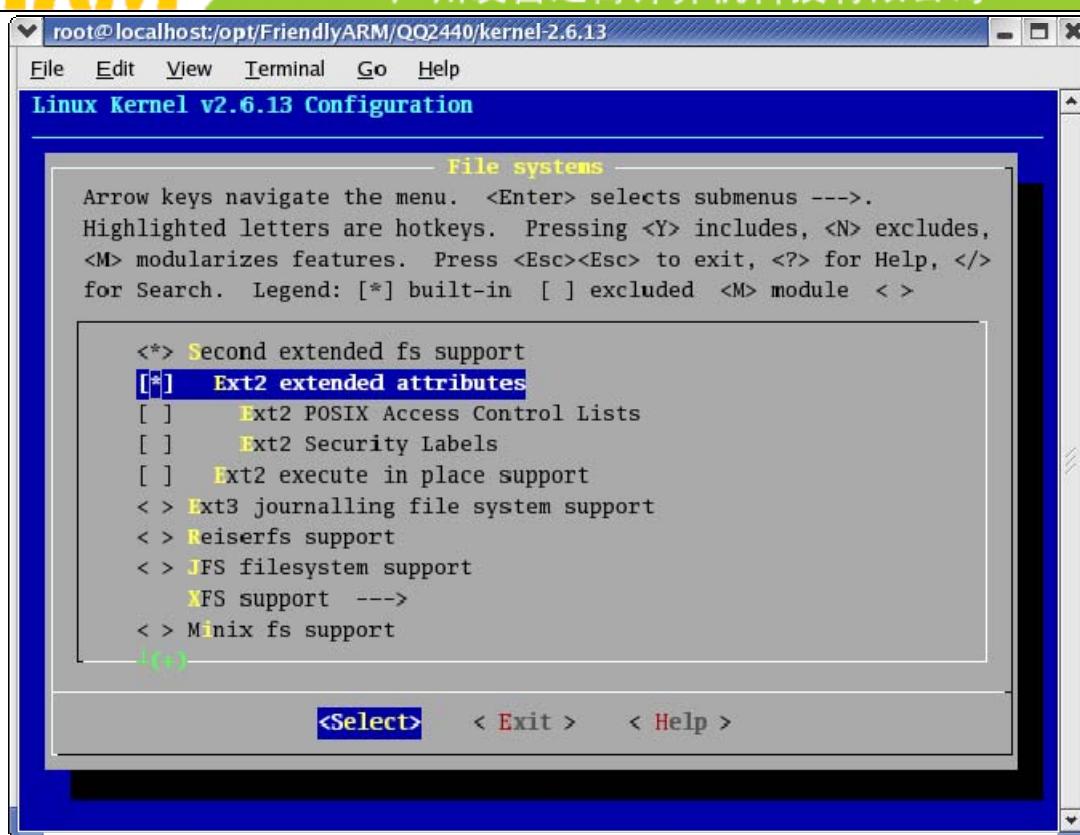


追求卓越 创造精品

TO BE BEST

TO DO GREAT

广州友善之臂计算机科技有限公司



选择内核启动后自动挂接支持，如图。

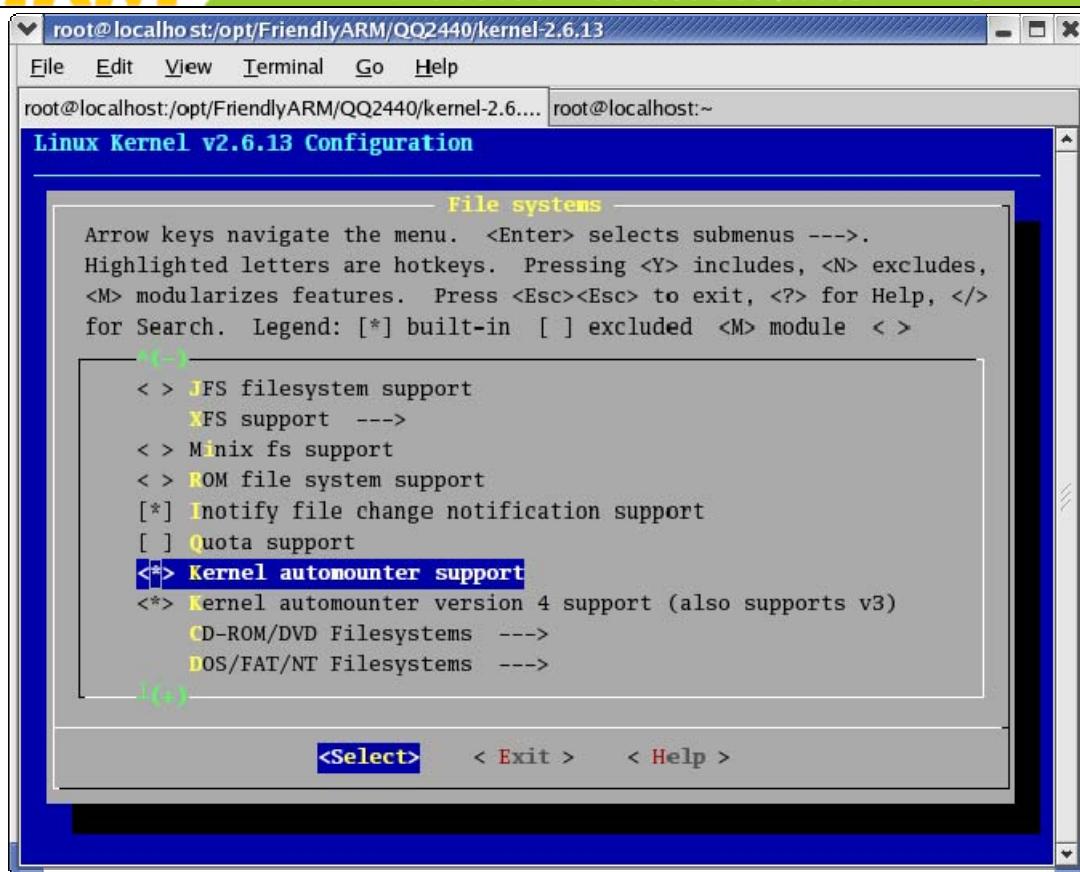


追 求 卓 越 创 造 精 品

TO BE BEST

TO DO GREAT

广州友善之臂计算机科技有限公司



选择进入 DOS/FAT/NT Filesystems，如图选择对 VFAT 的支持。

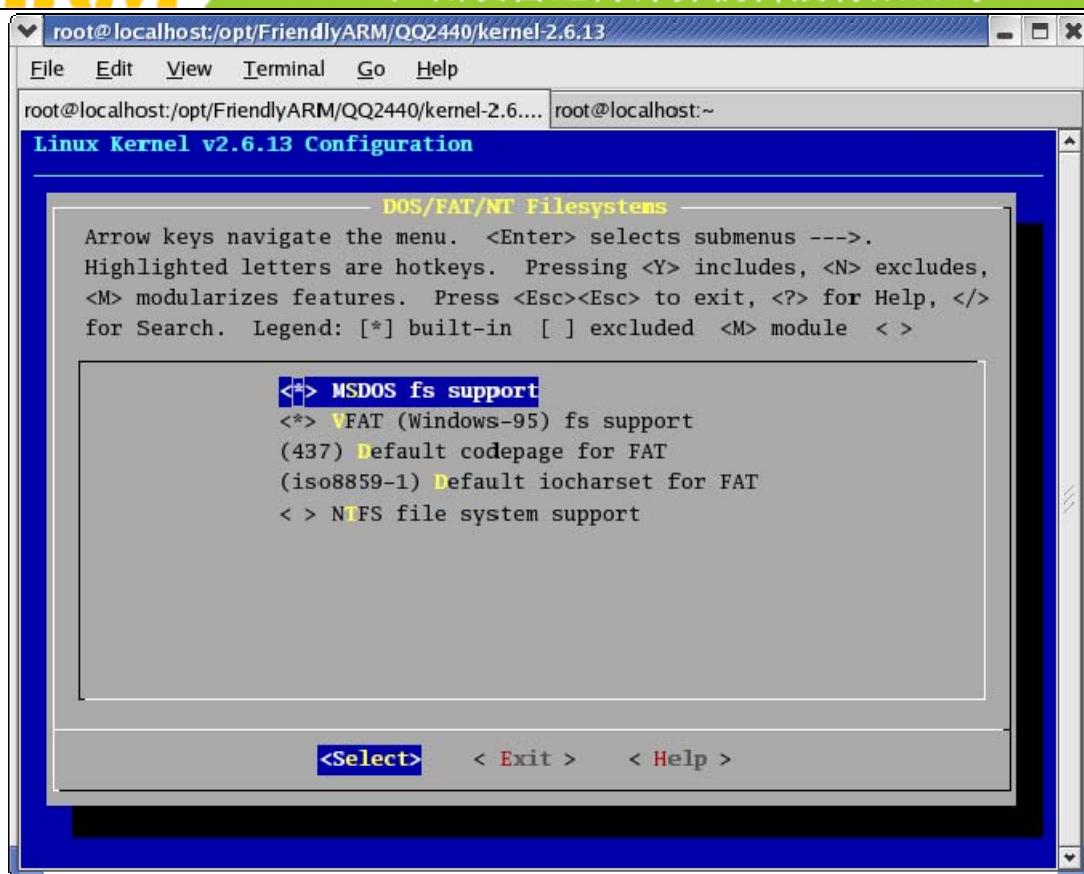


追 求 卓 越 创 造 精 品

TO BE BEST

TO DO GREAT

广州友善之臂计算机科技有限公司



返回 File System 菜单，然后选择进入 Pseudo filesystems - - ->，如图进行选择

**注意：该项必须选择，否则yaffs 文件系统不会正确运行。**

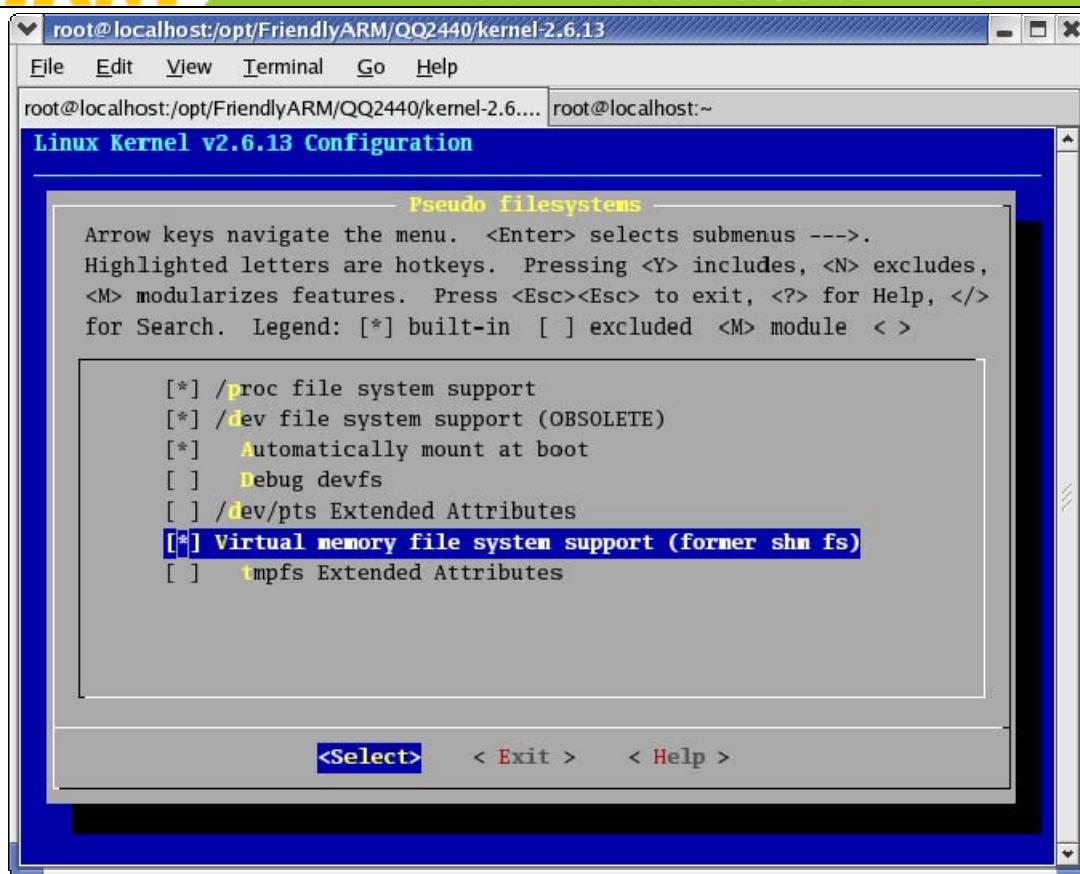


追求卓越 创造精品

TO BE BEST

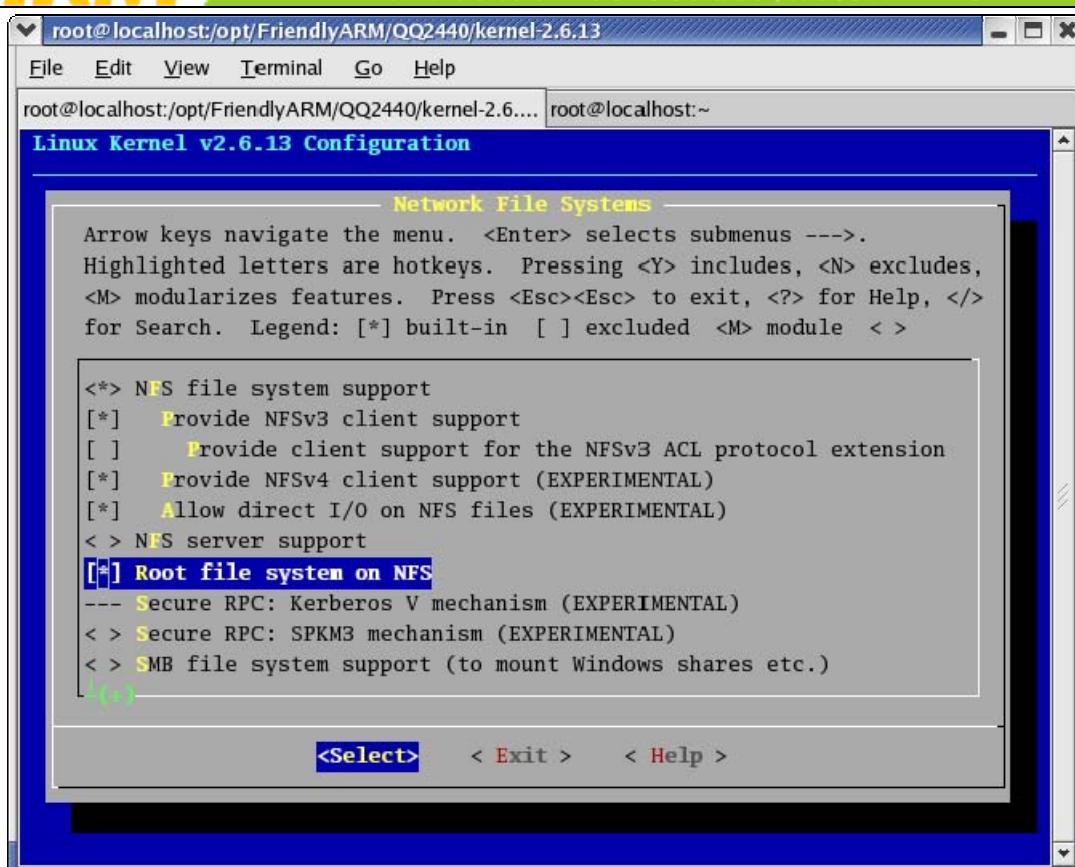
TO DO GREAT

广州友善之臂计算机科技有限公司



返回 File System 菜单，然后选择进入 Network File Systems - - ->，如图选择以配置网络文件系统(NFS)。

**注意：只有正确选择了网络文件系统的支持选项才能使用网络文件系统。**



至此，您已经了解内核的大部分常用选项的配置，更多的内核选项需要您在学习中逐步实践和摸索。

### 8.3 yaffs 根文件系统映象的制作

制作 yaffs 文件系统映象需要使用 mkyaffsimage 工具程序

在此，我们以制作测试用的 root\_default.img 文件系统映象为例，来介绍 yaffs 文件系统映象的制作。

(1) 把光盘中的 mkyaffsimage.tgz 文件拷贝到某一个目录，进入该目录，然后执行以下命令：

```
#tar xvzf mkyaffsimage.tgz -C /usr/sbin
```

这将把制作工具 mkyaffsimage 安装到系统的可执行路径/usr/sbin

(2) 拷贝光盘中的 root\_default.tgz 到某一个目录，进入该目录，然后执行以下解压命令：

```
#tar xvzf root_default.tgz -C /opt/FriendlyARM/mini2440
```

该命令将把 root\_default 文件系统目录解压到/opt/FriendlyARM/mini2440 目录。

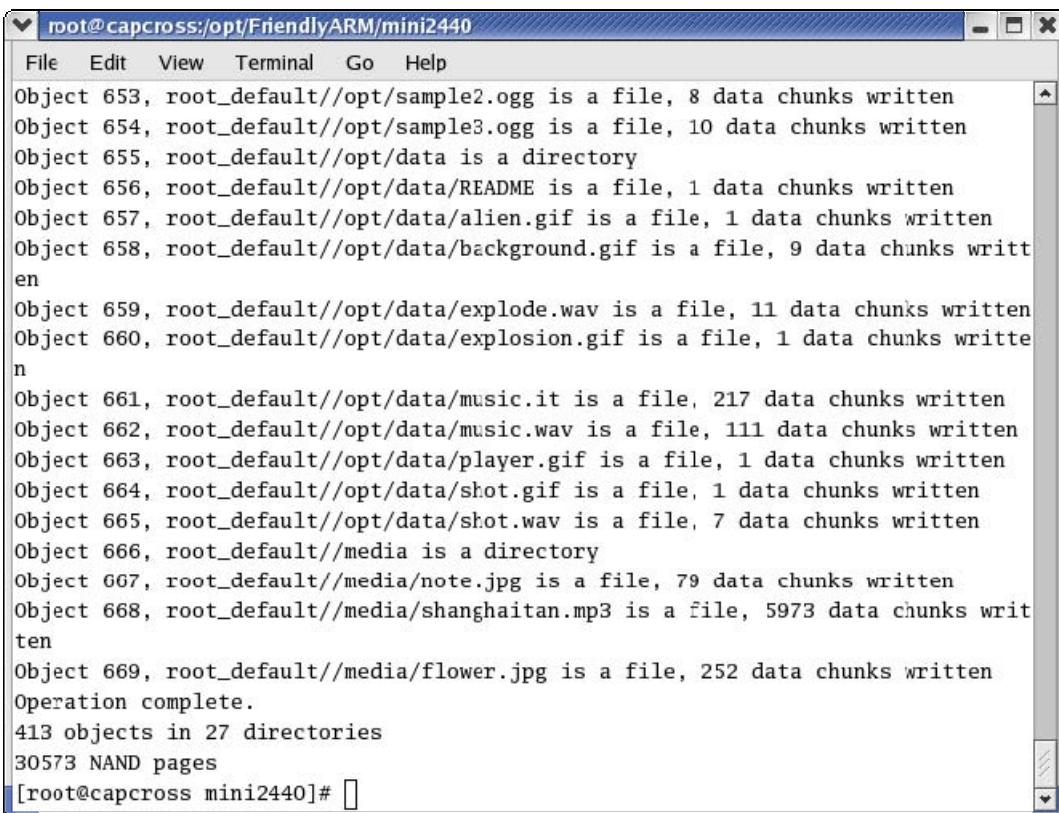


```
root@capcross:~/opt/FriendlyARM/mini2440$ ls
examples      root_default  root_qtopia_mouse
kernel-2.6.13  root_mizi    root_qtopia_tp
root@capcross:~/opt/FriendlyARM/mini2440$
```

(3) 把您自己制作的应用程序或者其他文件放入文件系统，如 hello 例子程序：

```
#cp examples/hello/hello root_default/sbin
```

(4) 使用 mkyaffsimage 制作 yaffs 文件系统映象：



```
root@capcross:~/opt/FriendlyARM/mini2440$ mkyaffsimage -o root_default.img root_default
Object 653, root_default//opt/sample2.ogg is a file, 8 data chunks written
Object 654, root_default//opt/sample3.ogg is a file, 10 data chunks written
Object 655, root_default//opt/data is a directory
Object 656, root_default//opt/data/README is a file, 1 data chunks written
Object 657, root_default//opt/data/alien.gif is a file, 1 data chunks written
Object 658, root_default//opt/data/background.gif is a file, 9 data chunks written
Object 659, root_default//opt/data/explode.wav is a file, 11 data chunks written
Object 660, root_default//opt/data/explosion.gif is a file, 1 data chunks written
Object 661, root_default//opt/data/music.it is a file, 217 data chunks written
Object 662, root_default//opt/data/music.wav is a file, 111 data chunks written
Object 663, root_default//opt/data/player.gif is a file, 1 data chunks written
Object 664, root_default//opt/data/shot.gif is a file, 1 data chunks written
Object 665, root_default//opt/data/shot.wav is a file, 7 data chunks written
Object 666, root_default//media is a directory
Object 667, root_default//media/note.jpg is a file, 79 data chunks written
Object 668, root_default//media/shanghaitan.mp3 is a file, 5973 data chunks written
Object 669, root_default//media/flower.jpg is a file, 252 data chunks written
Operation complete.
413 objects in 27 directories
30573 NAND pages
root@capcross:~/opt/FriendlyARM/mini2440$
```

可以看到，已经在当前目录下生成了 root\_default.img 映象文件，使用前面章节介绍的烧写方法，可以通过 USB 把 root\_default.img 烧写到目标板。

## 第九章 WinCE 开发指南

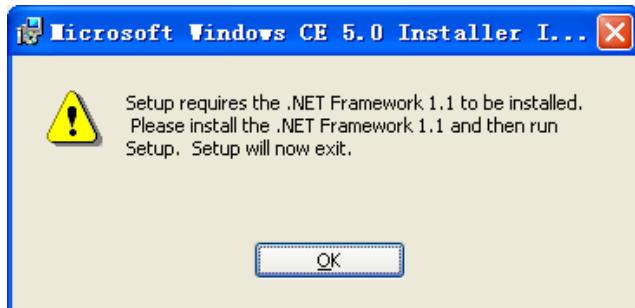
### 9.1 基于 WinCE5.0 的开发环境

#### 9.1.1 安装 Platform Builder 5.0(含 2007 最新补丁)

**提示:** Platform Builder 5.0 的补丁安装文件位于光盘“\WindowsCE5.0\PB5 补丁 2007”目录。

本节以图例介绍如何在 WindowsXP 上安装 Platform Builder 5.0(简称 PB5), 它用来开发和定制 WINCE 内核, 并可以用来调试内核等, 安装 PB 一般需要 5-7G 的硬盘空间。

(1) 安装PB5需要dotnet framework1.1, 如果系统中没有安装此组件, 将会出现如图提示:



在PB5的安装光盘中可以找到此组件的安装文件, 双击运行按照提示安装即可, 如图:

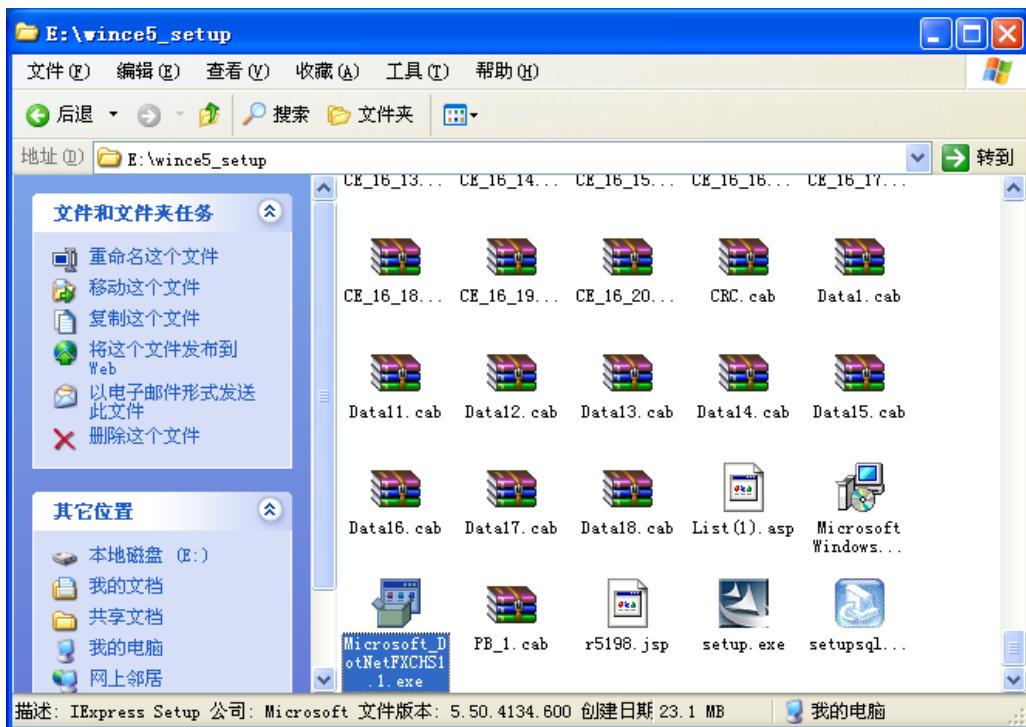


追求卓越 创造精品

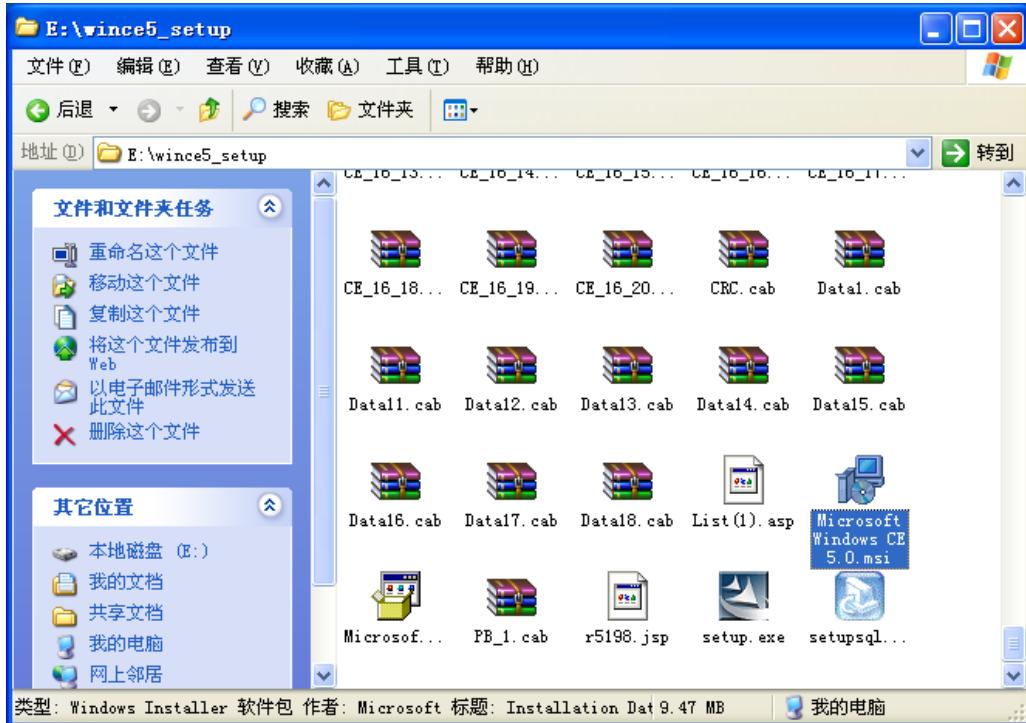
TO BE BEST

TO DO GREAT

广州友善之臂计算机科技有限公司



(2) 在PB5安装光盘中找到Microsoft Windows CE 5.0.msi.exe，双击运行开始安装PB5：



(3) 出现安装向导窗口，点“Next”继续：

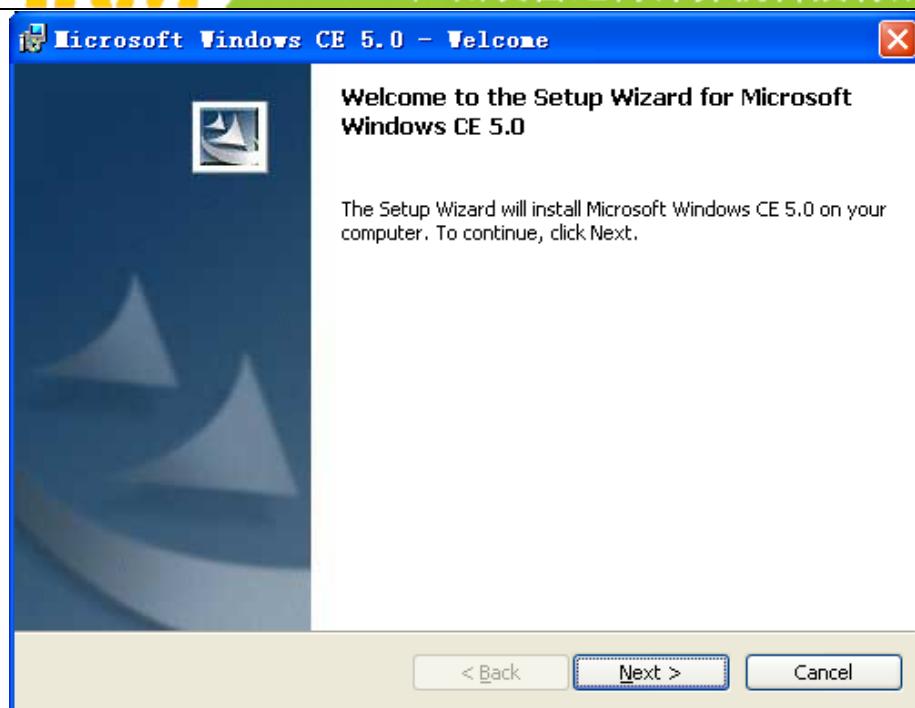


追求卓越 创造精品

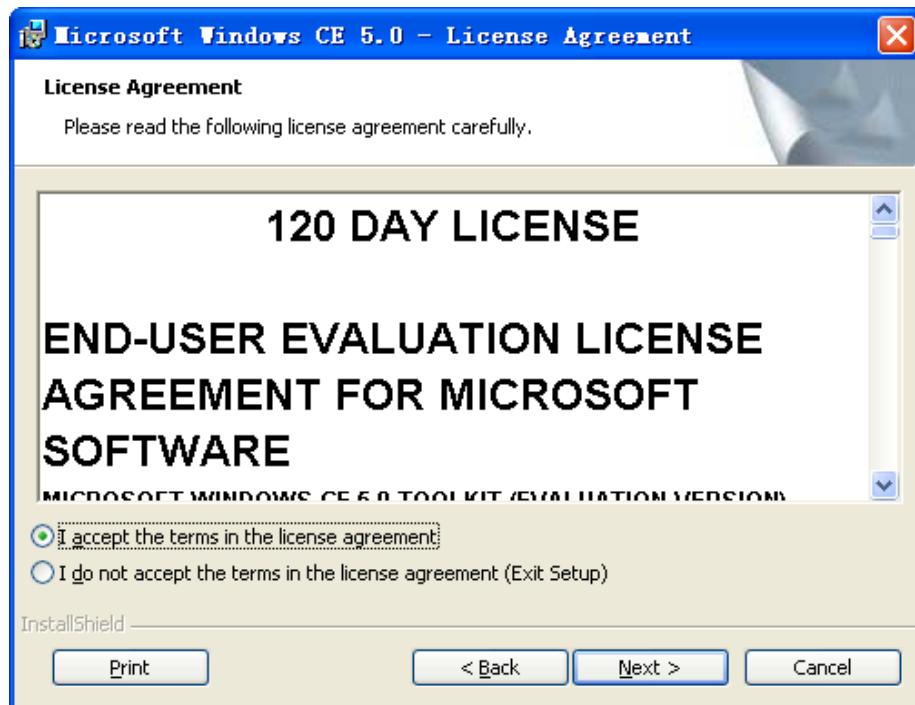
TO BE BEST

TO DO GREAT

广州友善之臂计算机科技有限公司



(4) 出现“License Agreement”窗口，选择“I accept the terms in the license agreement”，并点“Next”继续：



(5) 输入用户信息和序列号，点“Next”继续：

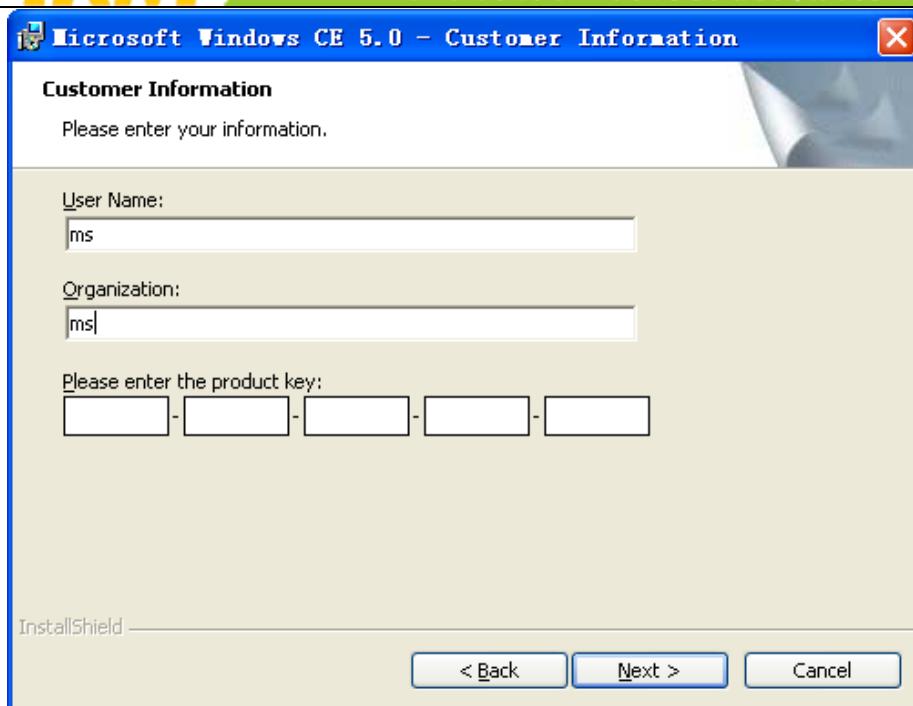


追求卓越 创造精品

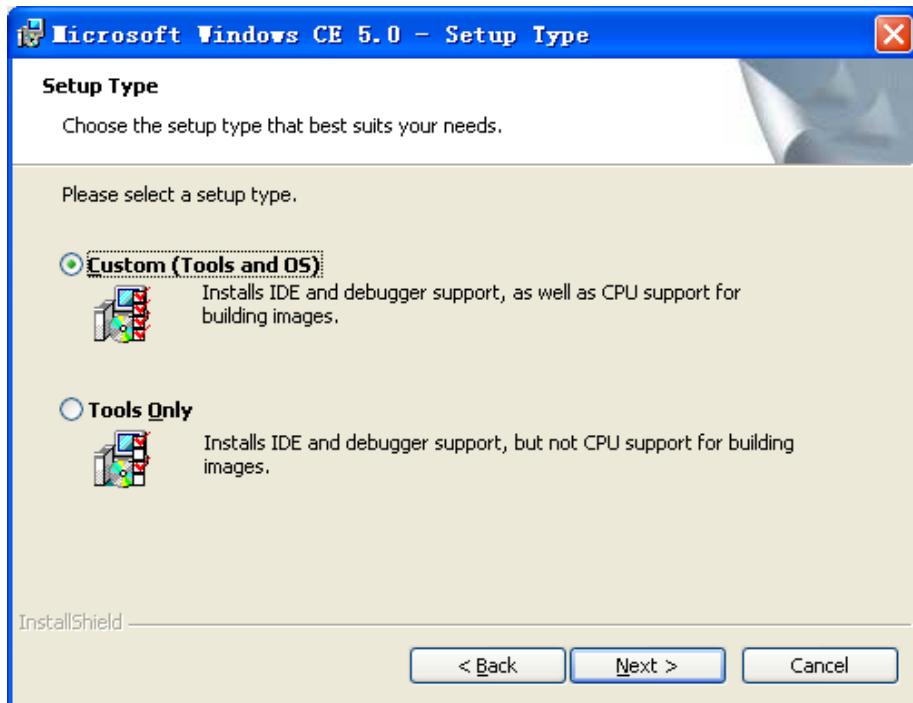
TO BE BEST

TO DO GREAT

广州友善之臂计算机科技有限公司



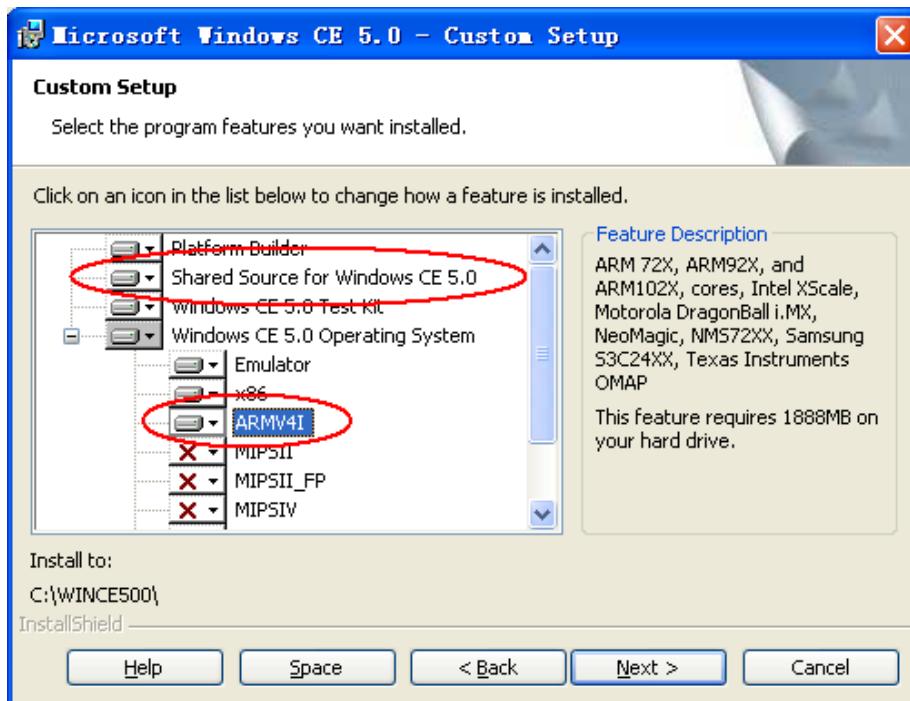
(6) 选择安装类型为定制安装，如图，点“Next”继续



(7) 选择安装目录，这里按默认，点“Next”继续



(8) 在定制安装中选择您所需要的系统平台，在这里一定要选择安装ARMV4I，最好也选择安装“Shared Source for Windows CE 5.0”，如图。点“Next”继续



(9) 出现许可协议窗口，选择 “I accept the terms in the license agreement”，并点“Next”继续：



追求卓越 创造精品

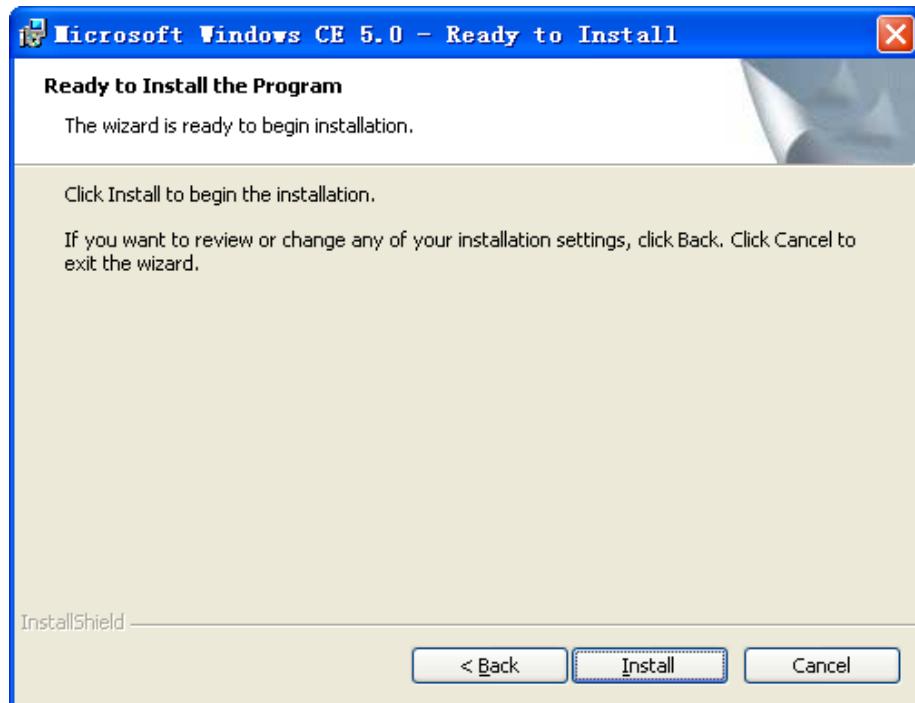
TO BE BEST

TO DO GREAT

广州友善之臂计算机科技有限公司



(10) 出现如图提示窗口，点“Next”继续



(11) 进入安装进程界面，此过程时间较长，请耐心等待，如图

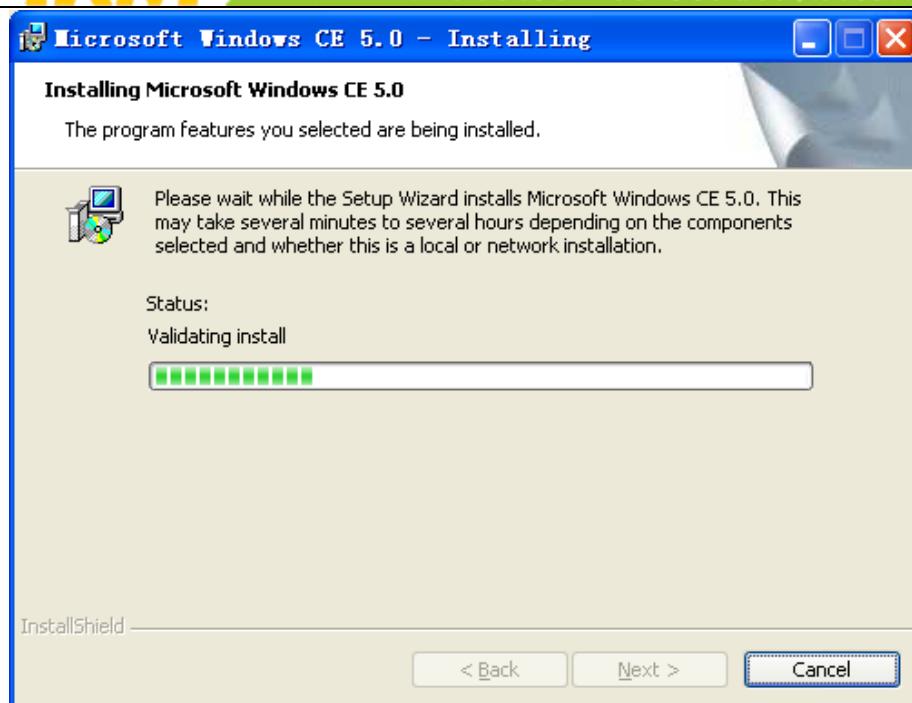


追求卓越 创造精品

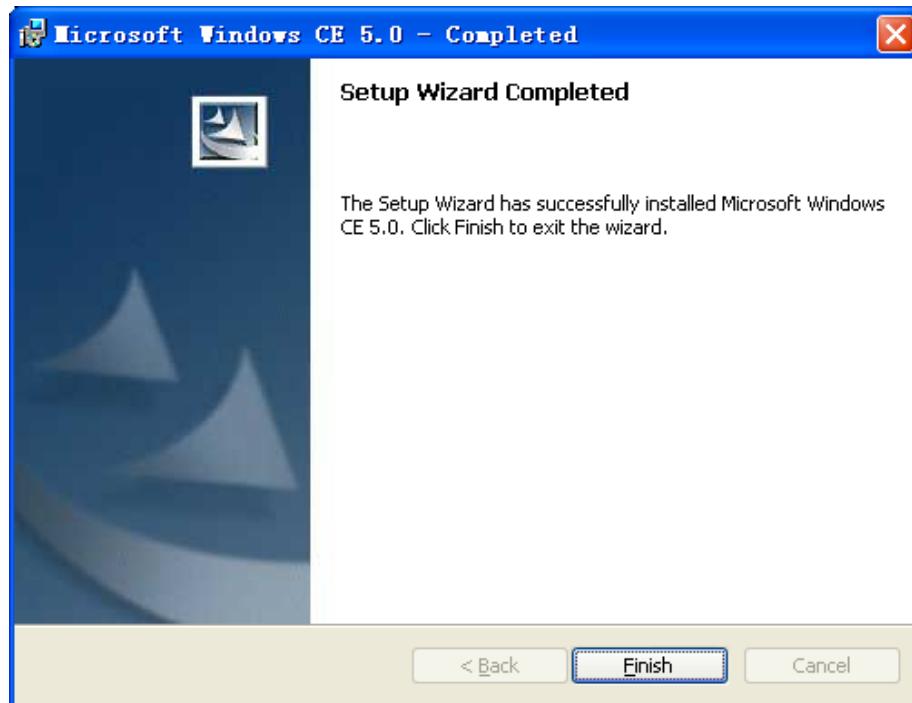
TO BE BEST

TO DO GREAT

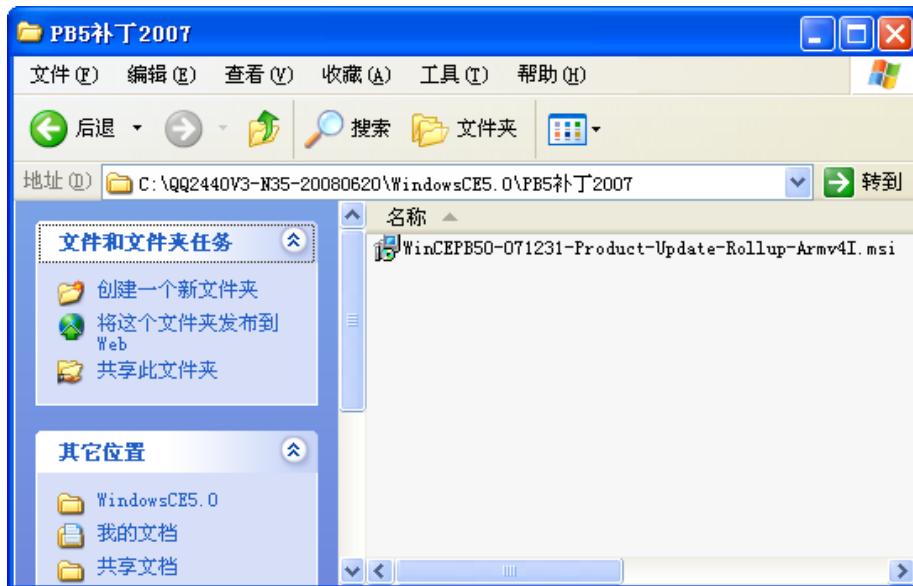
广州友善之臂计算机科技有限公司



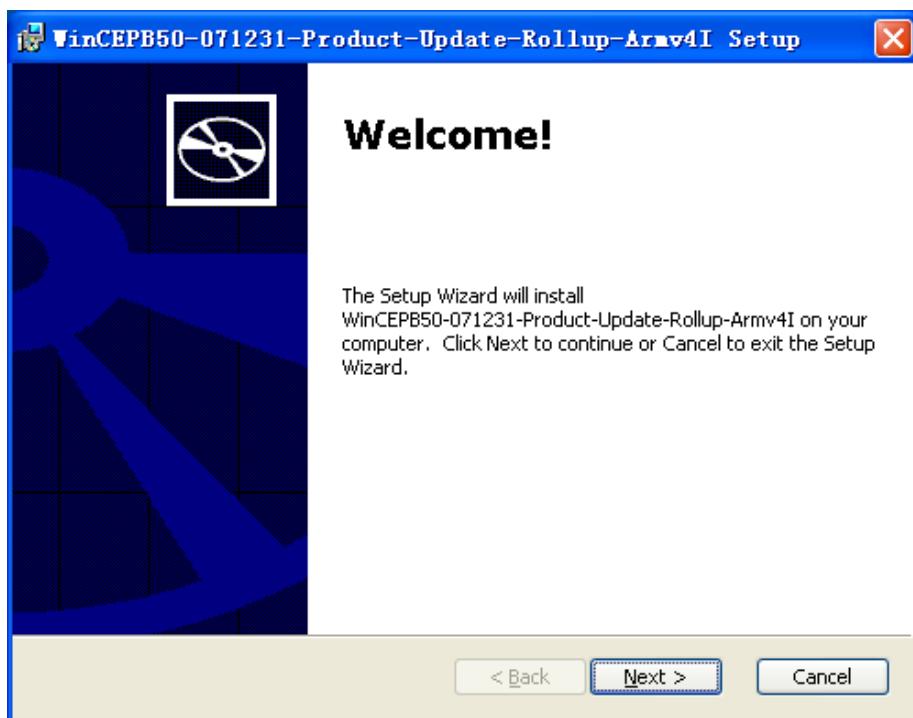
(12) 安装完毕，如图



(13) 现在开始安装PB5补丁程序，其安装文件位于光盘WindowsCE 5.0\PB5补丁2007文件夹中，双击该程序开始安装：



(14) 出现安装向导界面，如图，点“Next”继续



(15) 出现许可协议窗口，选择“*I accept the terms in the license agreement*”，并点“Next”继续：

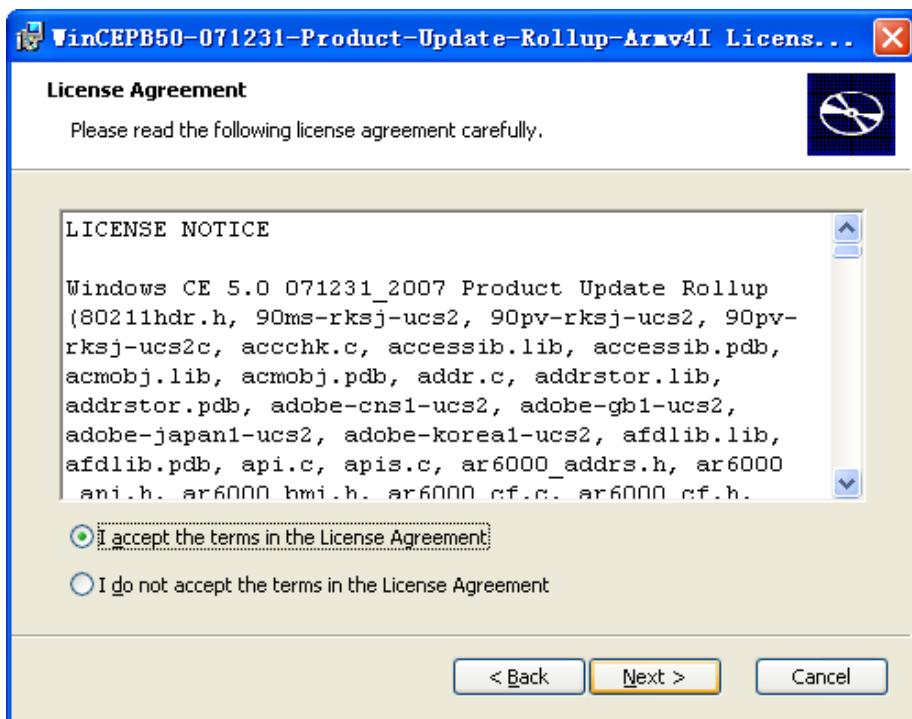


追求卓越 创造精品

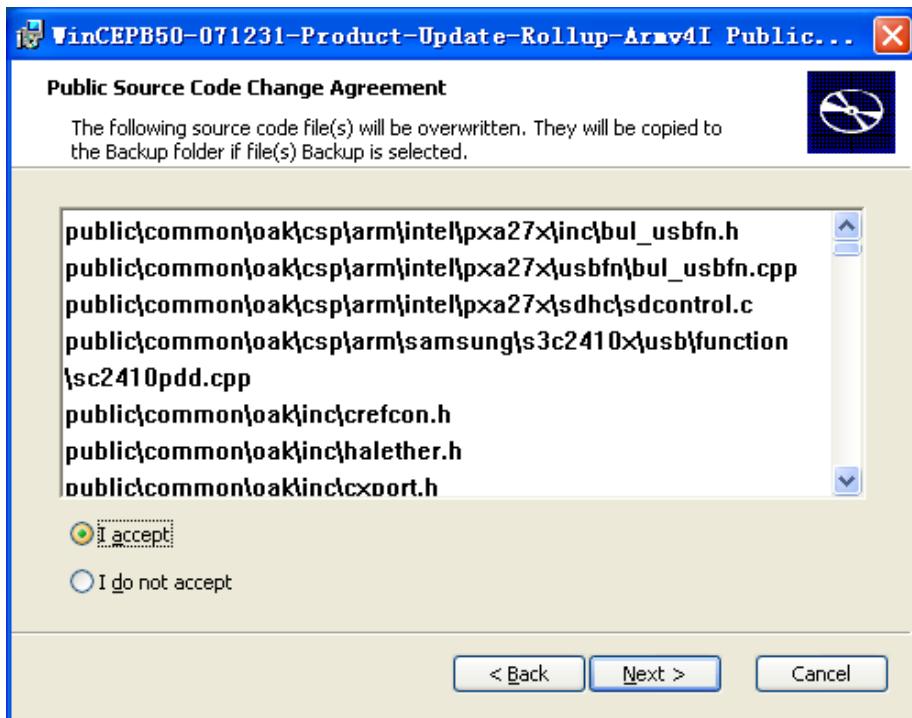
TO BE BEST

TO DO GREAT

广州友善之臂计算机科技有限公司



(16) 出现如图公共代码许可协议窗口，选择“ I accpet ”，点“ Next ”继续



(17) 出现定制安装界面，选择默认安装全部组件，点“ Next ”继续

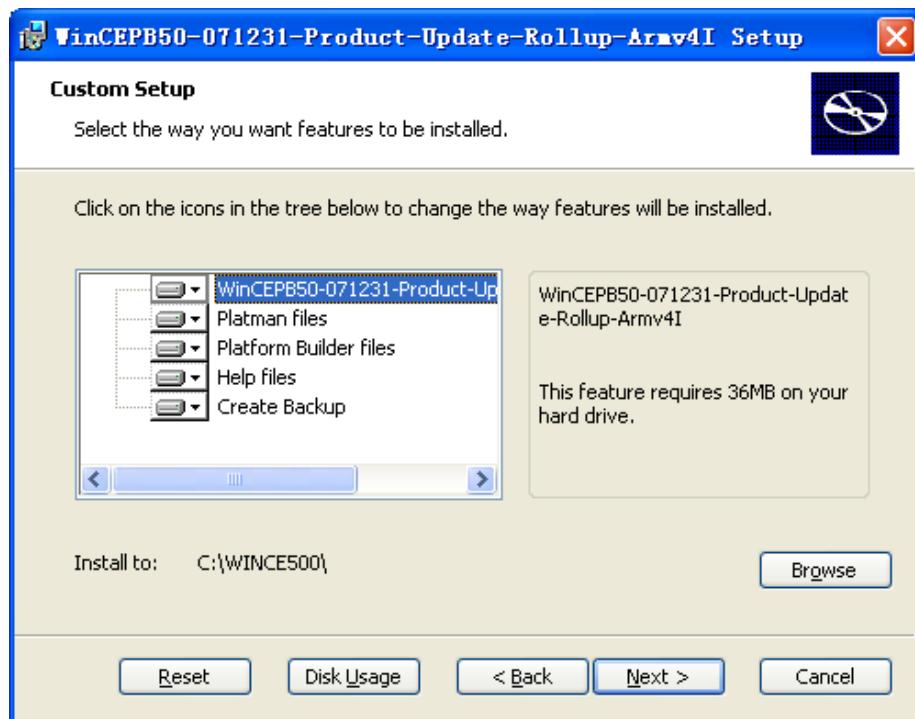


追求卓越 创造精品

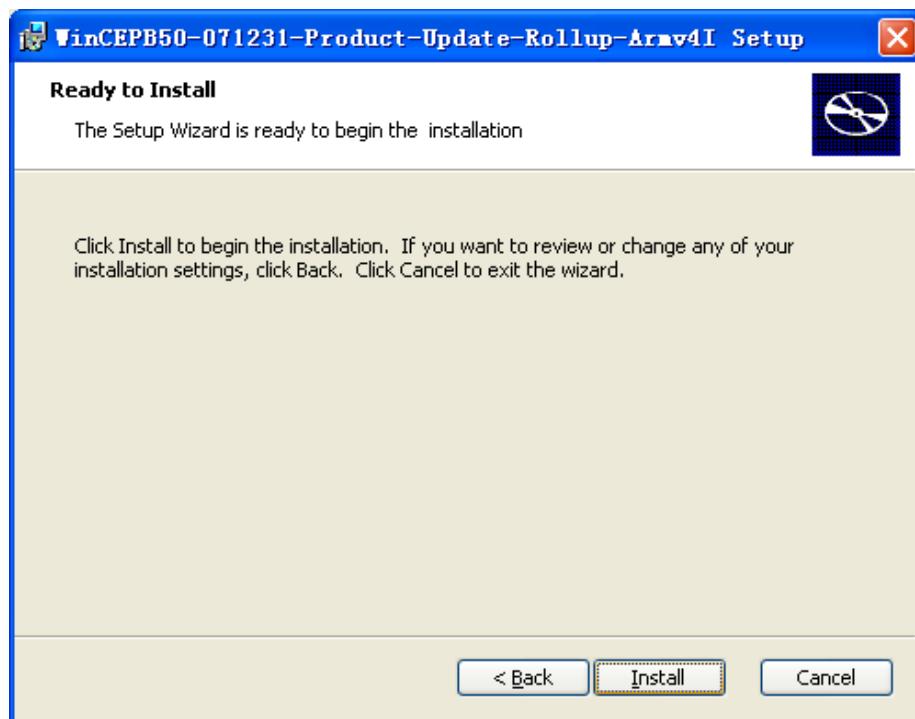
TO BE BEST

TO DO GREAT

广州友善之臂计算机科技有限公司



(18) 准备好安装，点“Next”继续



(19) 安装过程如图所示，该过程比较长，请耐心等待

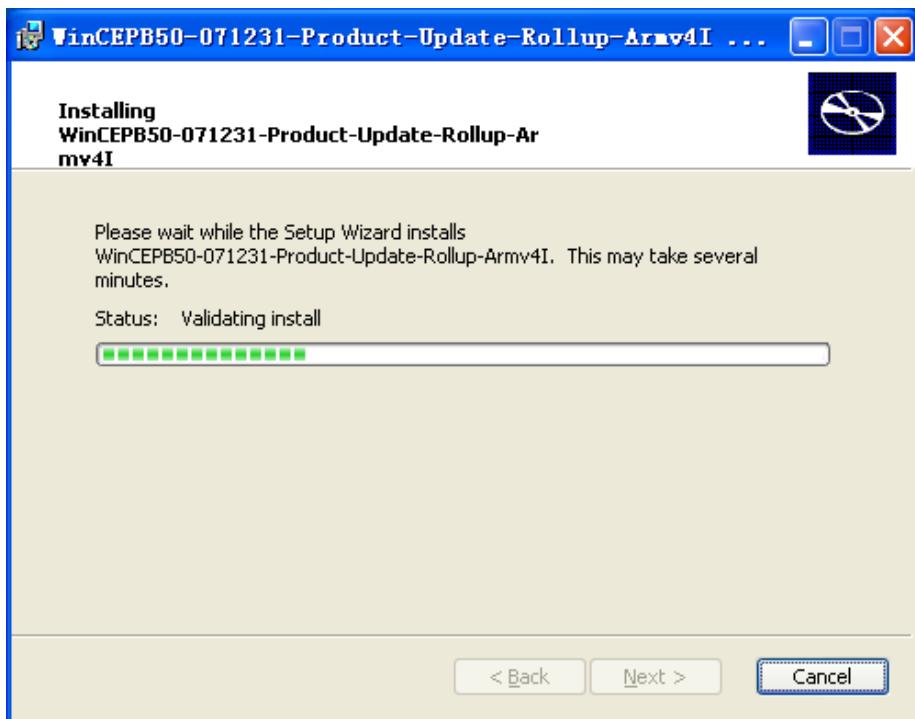


追求卓越 创造精品

TO BE BEST

TO DO GREAT

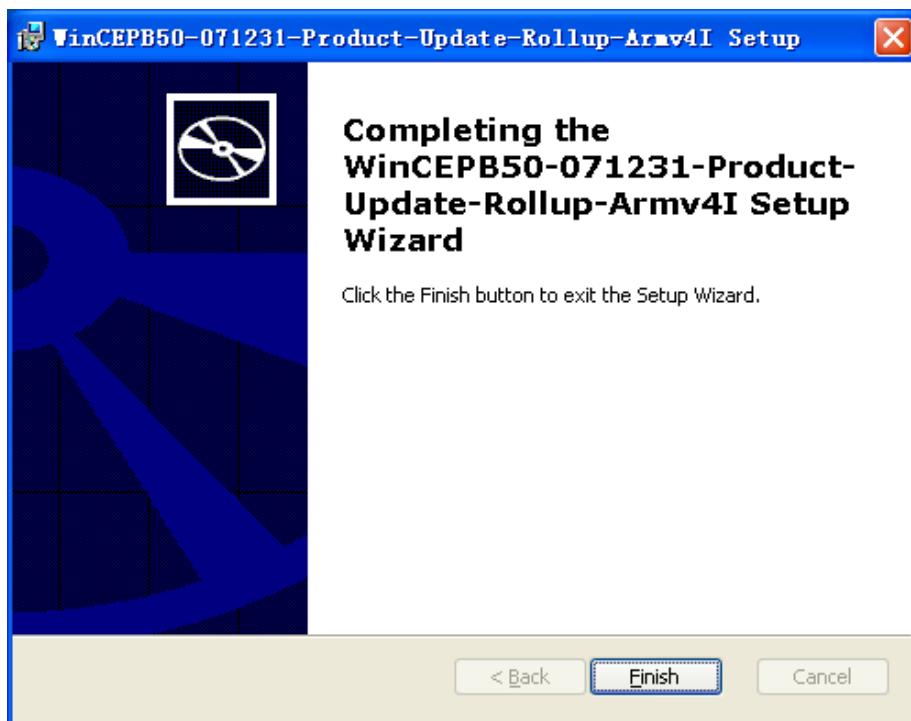
广州友善之臂计算机科技有限公司



(20) 安装过程会出现如下提示界面，不必理会，点“确定”即可



(21) 安装完毕，如图



### 9.1.2 导入安装 BSP

**说明:** 光盘中附带的 BSP(即 WindowsCE5.0\smdk2440 文件夹)目前支持以下型号的液晶屏:

- NEC3.5 寸屏带触摸
- 7 寸屏带触摸
- VGA 模块显示输出, 分辨率 1024x768

为了使用相应型号的液晶屏, 需要对 BSP 的设置做如下修改:

1. 修改\smdk2440\INC\s2440.h 中 LCD\_TYPE 的定义, 找到如下定义语句:

```
#define LCD_TYPE_N35    1 //适用于 NEC3.5 寸屏
#define LCD_TYPE_A70     2 //适用于 7 寸屏
#define LCD_TYPE_VGA1024x768 3 //适用于 VGA 模块输出, 分辨率为 1024x768
#define LCD_TYPE LCD_TYPE_N35
```

把 LCD\_TYPE 改为相应的型号就可以了, 这里默认为 LCD\_TYPE\_N35

2. 修改 smdk2440\smdk2440.bat 批处理文件(使用“记事本”可以打开)

```
REM - LCD_TYPE for FriendlyARM
```

```
set BSP_LCD_TYPE_N35=1
```

```
set BSP_LCD_TYPE_A70=
```

把需要使用的型号定义设置为 1 就可以了, 其他为空。这里默认为 BSP\_LCD\_TYPE\_N35=1, VGA 模块板暂无相应的定义



追求卓越 创造精品

TO BE BEST

TO DO GREAT

广州友善之臂计算机科技有限公司

要使用 PB5 编译 WINCE 内核映象，需要安装对应目标板的 BSP，并进行一些设置。  
请按照以下步骤安装 BSP：

Step1：把光盘/WindowsCE 5.0目录里面的SMDK2440文件夹复制

“C:\WINCE500\PLATFORM” 目录下，并去掉只读属性。



Step6：打开“Platform Builder 5.0”，选择“File”菜单下的“Manage CatalogFeatures”

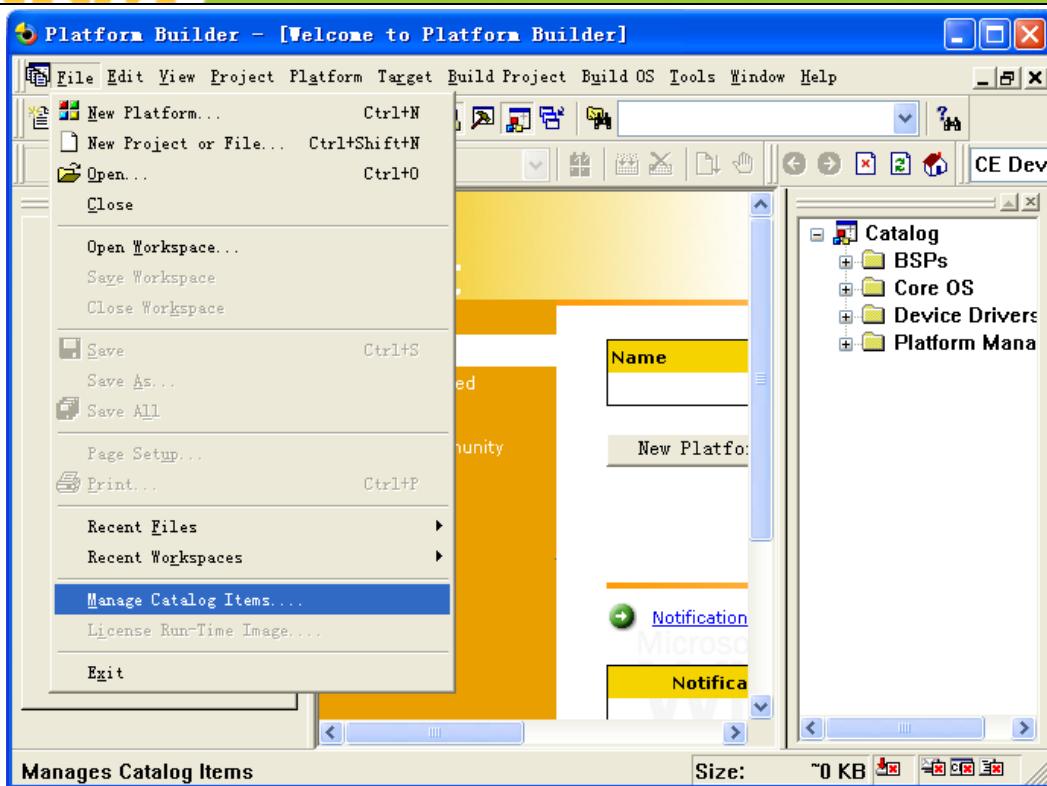


追求卓越 创造精品

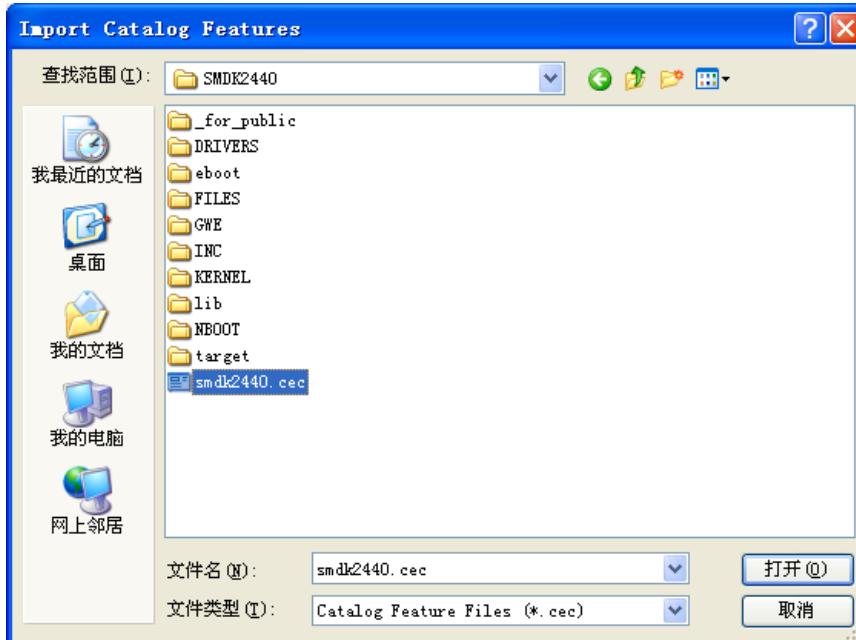
TO BE BEST

TO DO GREAT

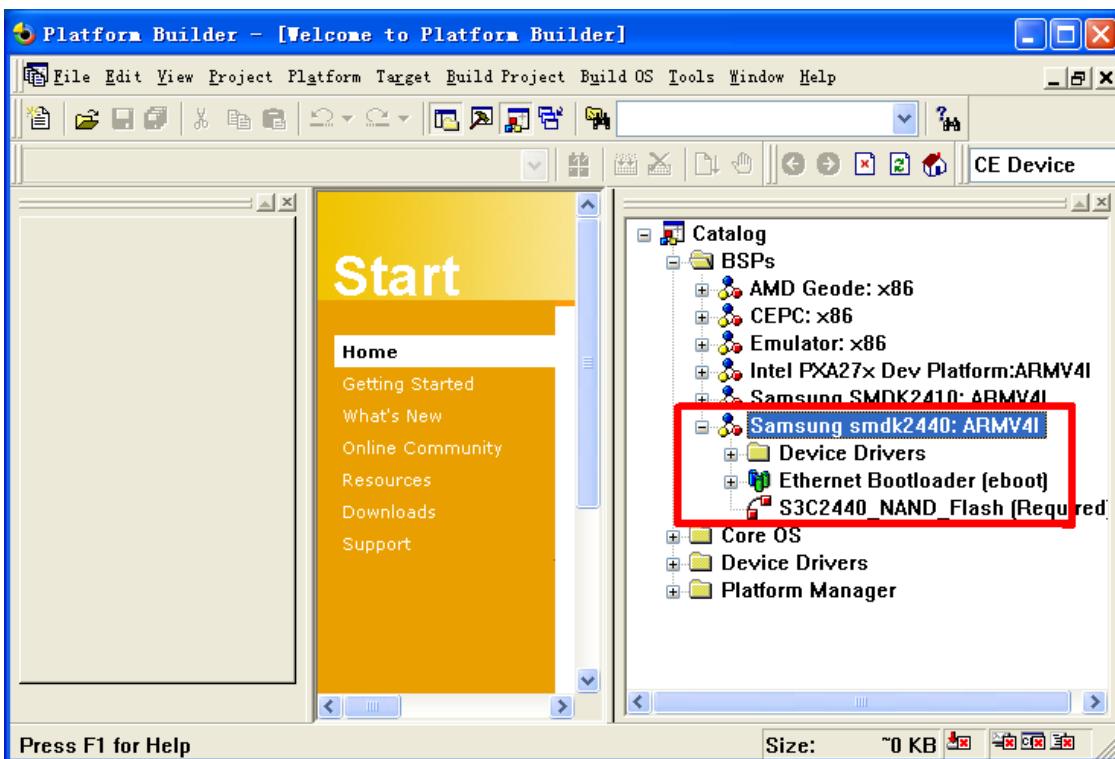
广州友善之臂计算机科技有限公司



点“Import按钮”，导入“platform\smdk2440\smdk2440.cec”文件



Step7: 在“Catalog”的BSP下，将会自动添加“Samsung SMDK2440:ARMV4M”项，BSP安装完成。



### 9.1.3 安装无线网卡驱动程序

无线网卡驱动程序位于光盘WindowsCE驱动程序模块\无线网卡\文件夹中，它是一个安装文件“VNUWLC5-ARM.msi”。下面是安装步骤：

- (1) 双击运行安装程序，打开安装向导，点“Next”继续

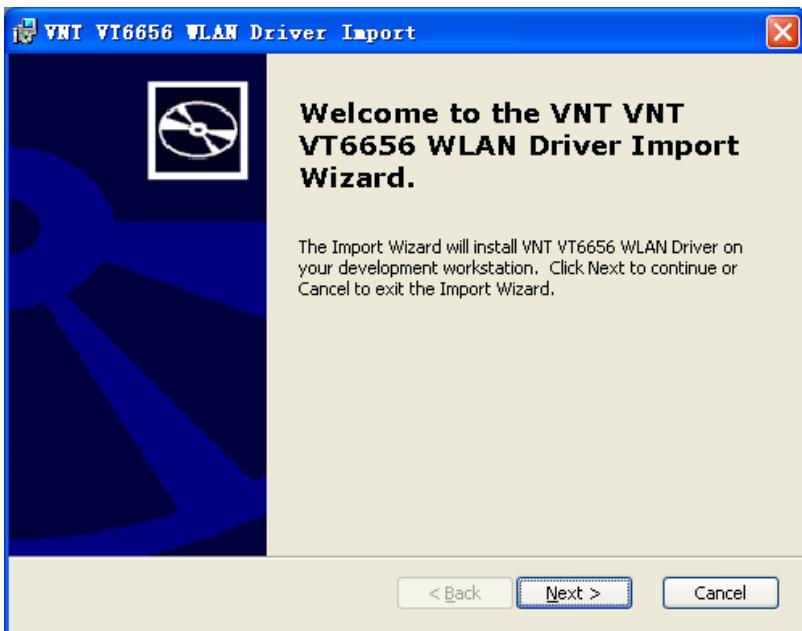


追求卓越 创造精品

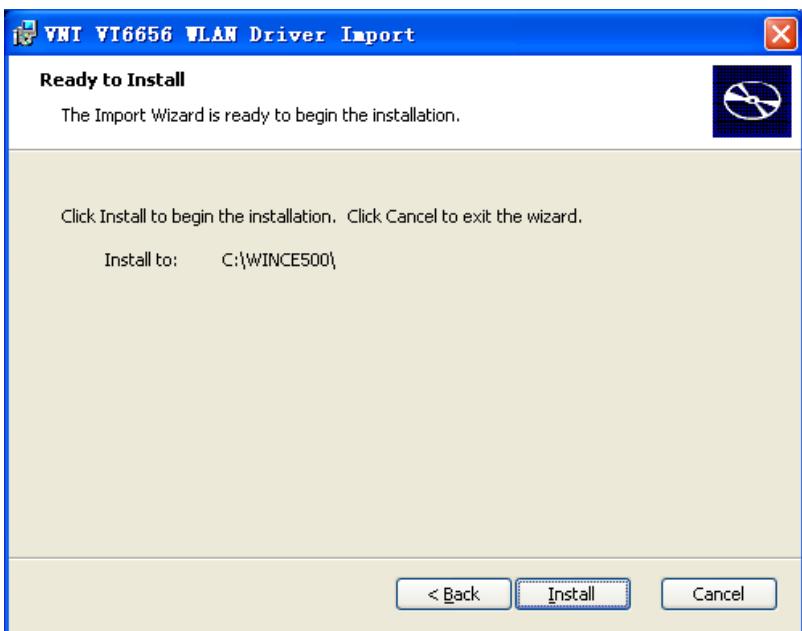
TO BE BEST

TO DO GREAT

广州友善之臂计算机科技有限公司



(2) 开始安装，并提示将安装到C:\WINCE500目录中，如图



(3) 安装进程如图，安装很快就会结束

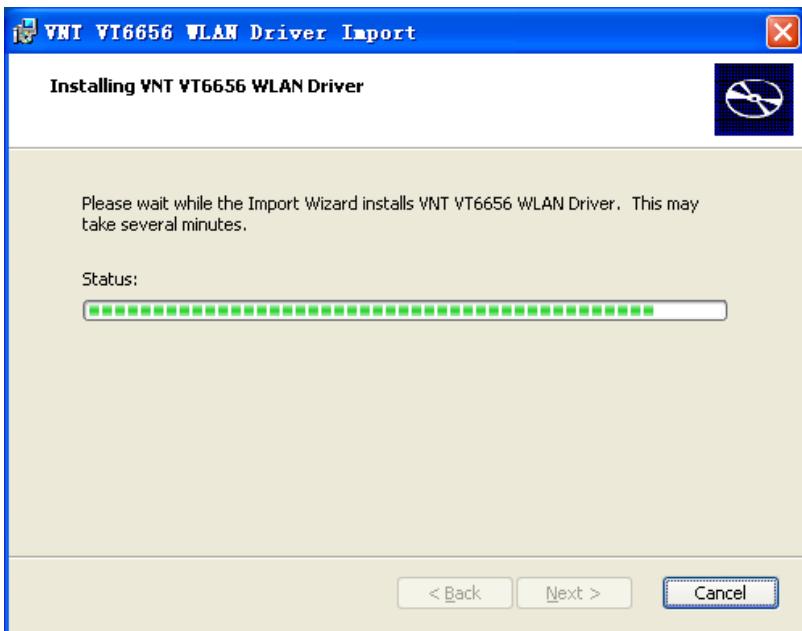


追求卓越 创造精品

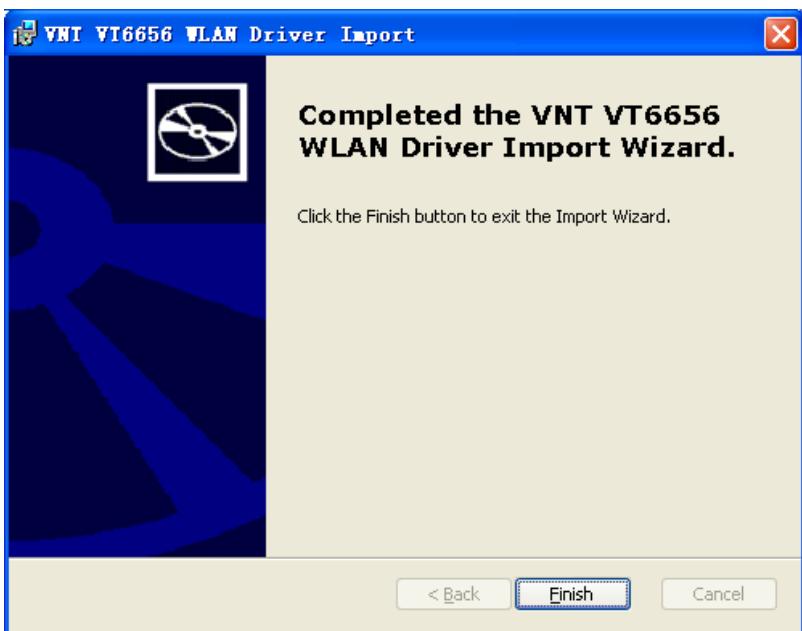
TO BE BEST

TO DO GREAT

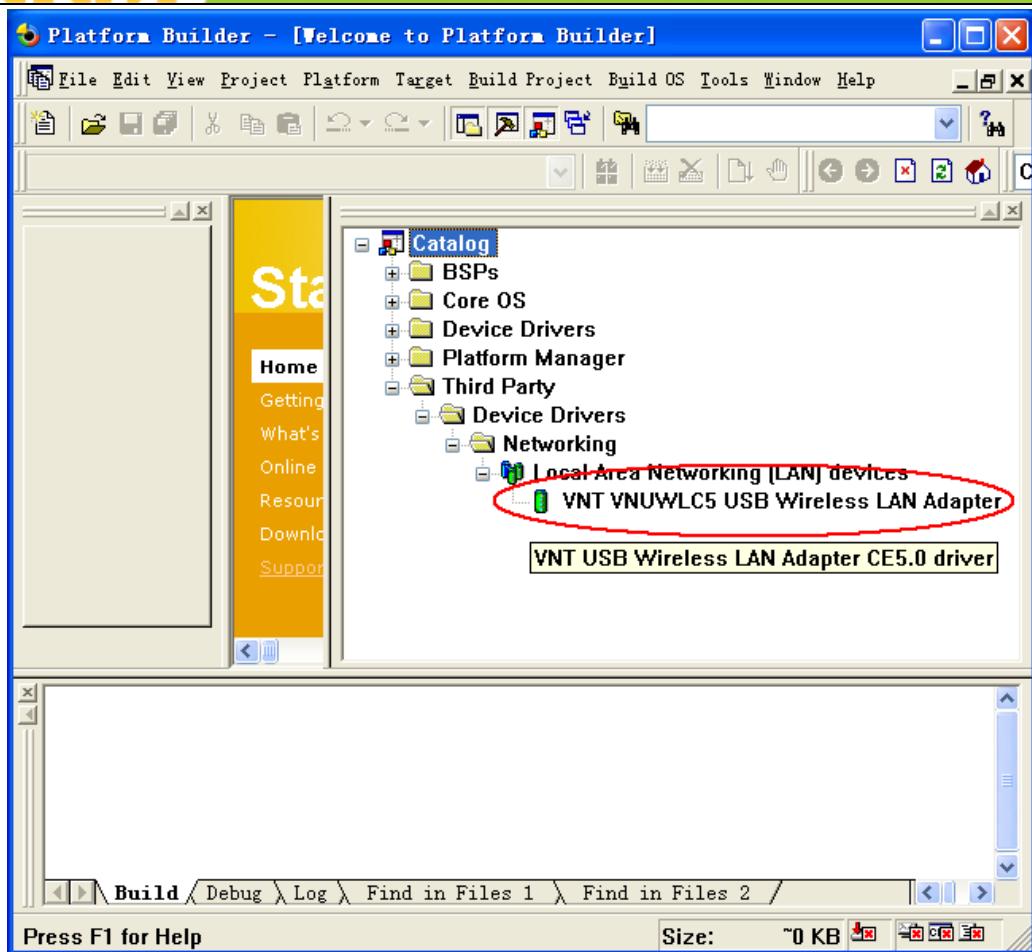
广州友善之臂计算机科技有限公司



(4) 安装结束，点“Finish”结束安装



(5) 这时打开PB5，会看到Catalog一栏出现如图选项，如图



#### 9.1.4 编译内核工程示例

(1) 在 C:\WINCE420\PBWorkspaces 目录(如果没有, 可以手工创建一个)中创建一个文件夹 “ mini2440 ”, 把光盘中 WindowsCE 5.0 目录下的 mini2440.pbxml 文件 C:\WINCE420\PBWorkspaces\mini2440 目录, 并去掉只读属性。



追求卓越 创造精品

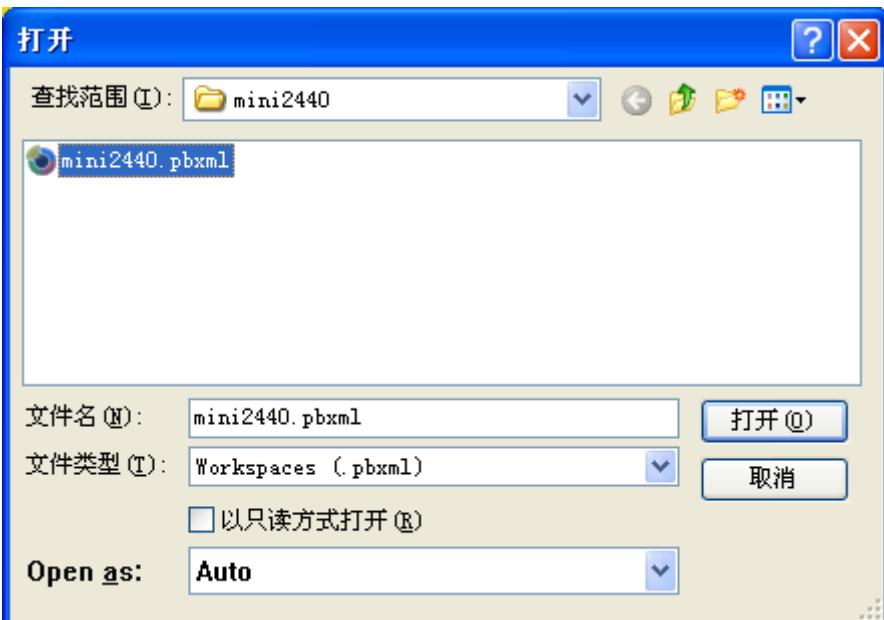
TO BE BEST

TO DO GREAT

广州友善之臂计算机科技有限公司



(2) 在 PB 中点 File → Open Workspace...，打开刚刚复制的项目文件，注意是 pbxml 结尾的。



如图为打开的项目文件之 PB 界面：

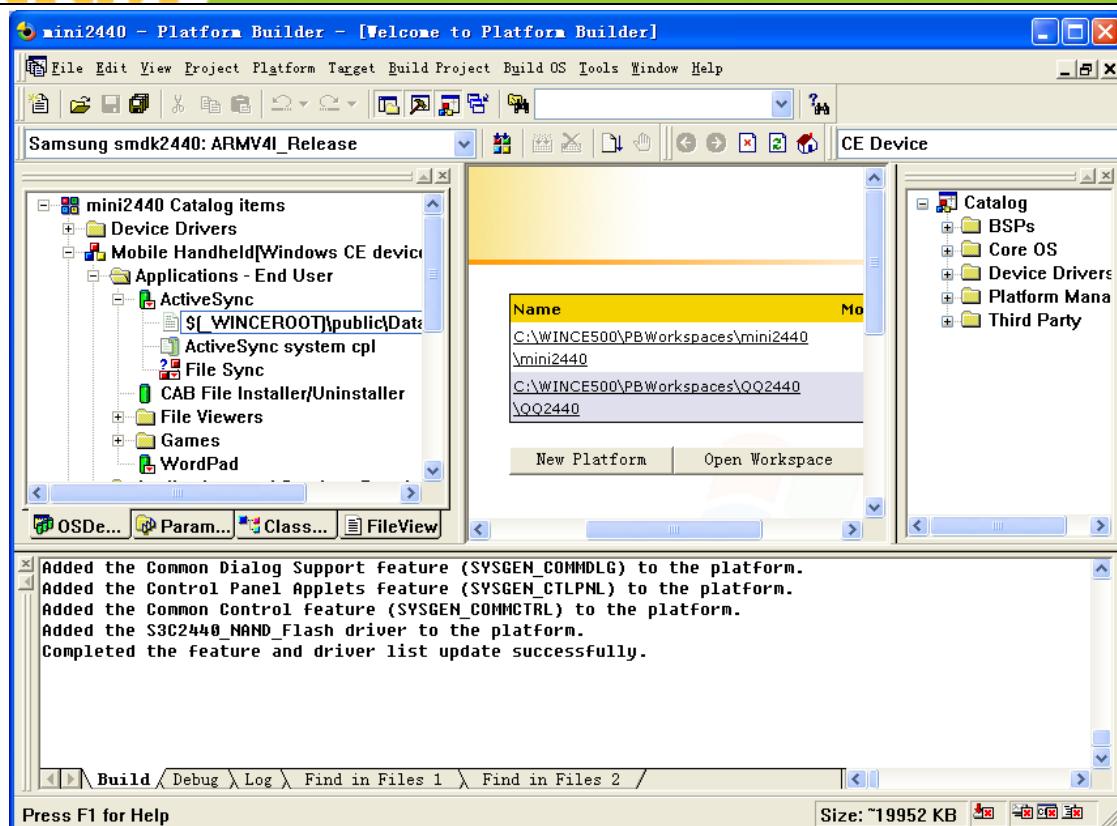


追求卓越 创造精品

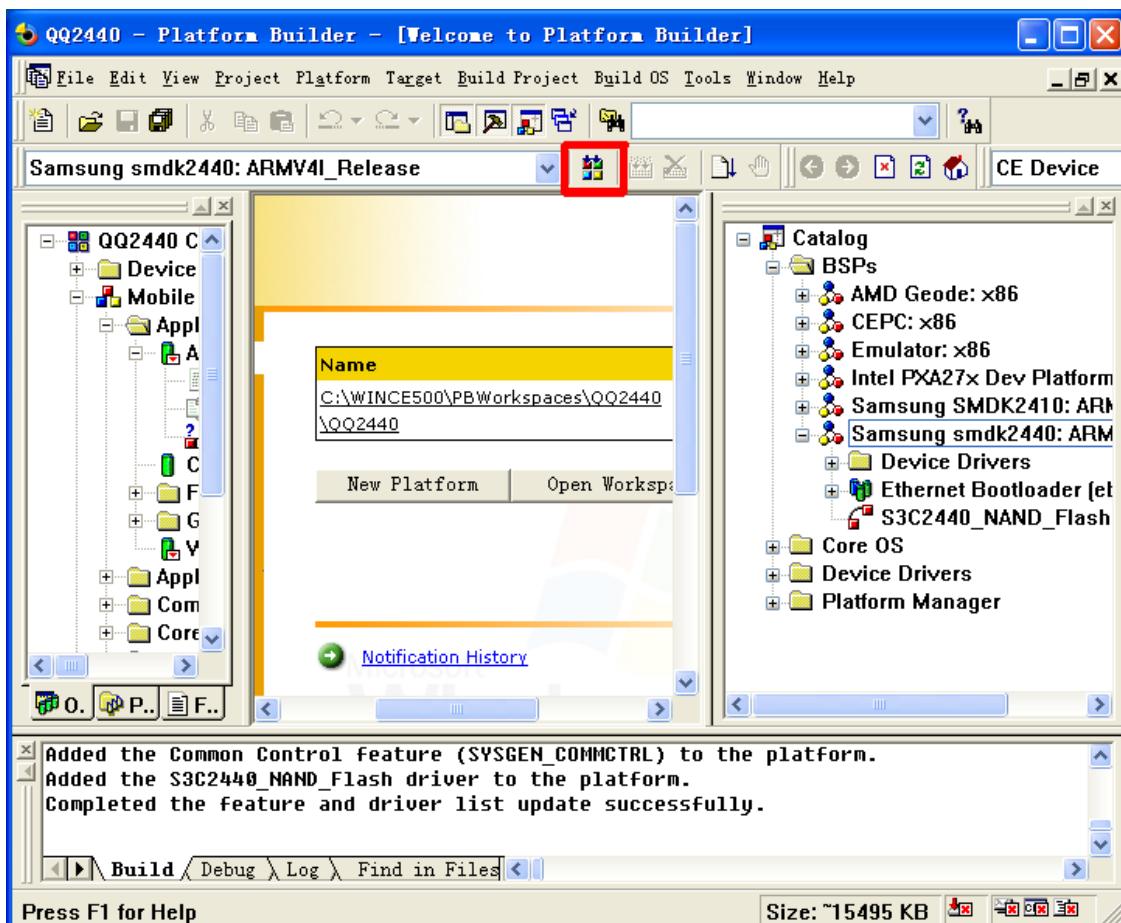
TO BE BEST

TO DO GREAT

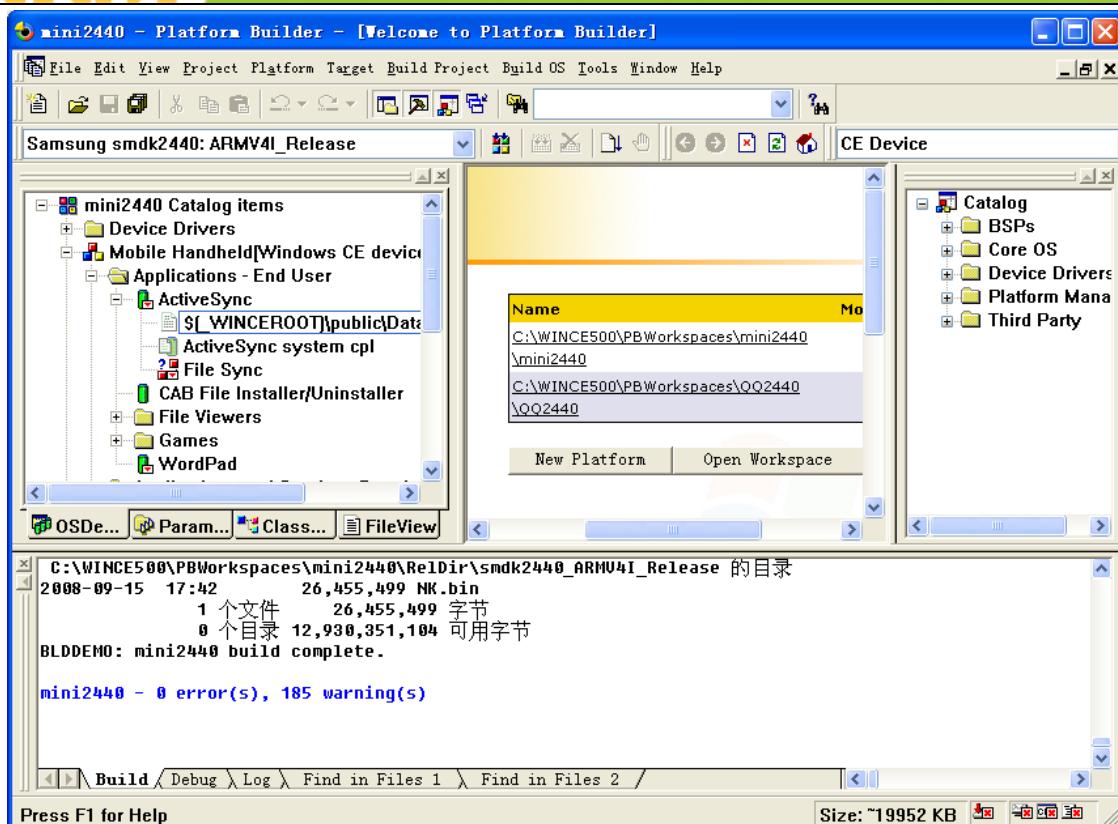
广州友善之臂计算机科技有限公司



(2) 打开后，点 Build OS → Sysgen 开始编译，或者点工具栏的  图标开始进行编译，该过程比较长。



(3) 编译完毕，就会生成“nk.bin”和“nk.nb0”两个文件，其中 nk.bin 是发行版本，nk.nb0 是内存中运行版本，我们一般使用 nk.bin。它们位于 C:\WINCE500\PBWorkspaces\mini2440\RelDir\smdk2440\_ARMV4I\_Release



编译过程中有可能会出现如图这样的警告信息，这个是正常的，不必理会。

### 9.1.5 导出 SDK

我们可以把定制好的内核工程导出为 SDK 安装文件，它用来提供给应用开发人员，里面主要包含与定制平台有关的头文件、库、一些文档等内容。应用开发人员可以通过安装 SDK 在 Embedded Visual C++(以下简称 EVC)中开发基于此平台的应用程序。

**说明：光盘“\WindowsCE5.0\SDK”目录中有已经制作好的 SDK 安装文件，您可以直接使用而不必自己制作。**

下面是具体的导出步骤。

(1) 首先打开并确定已经编译好工程示例，点 Platform → SDK → New SDK...如图：

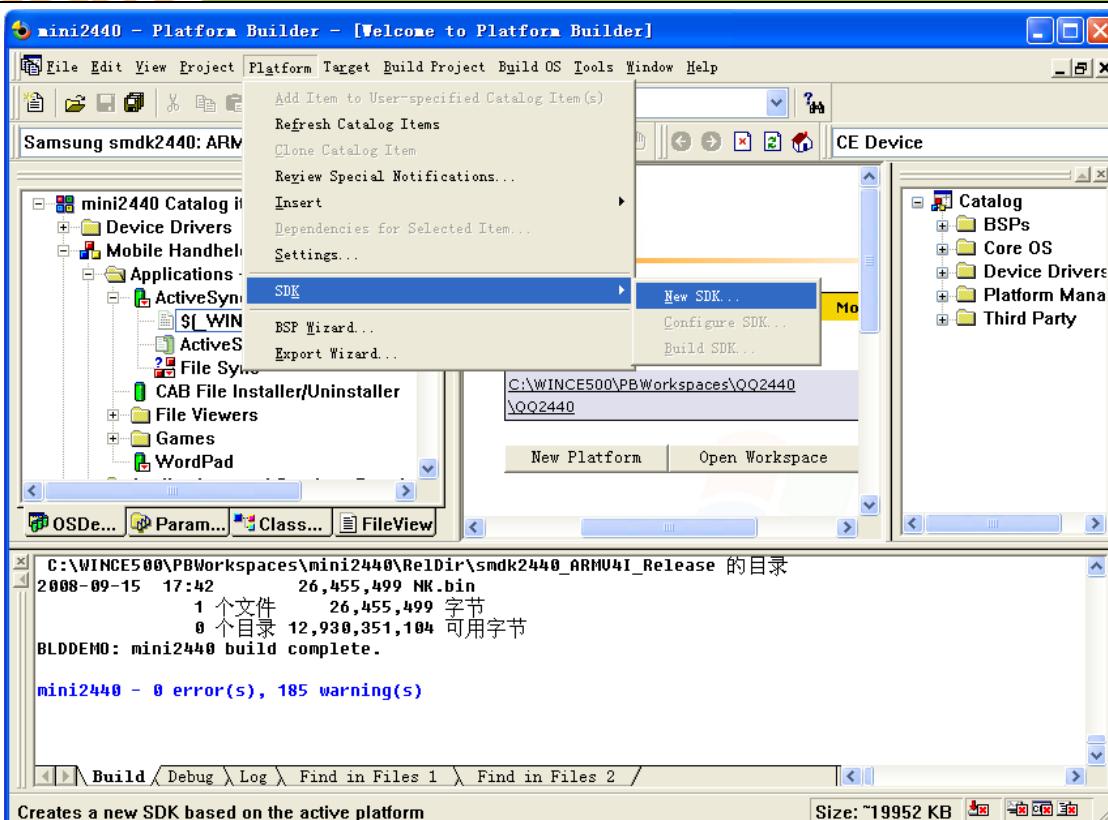


追求卓越 创造精品

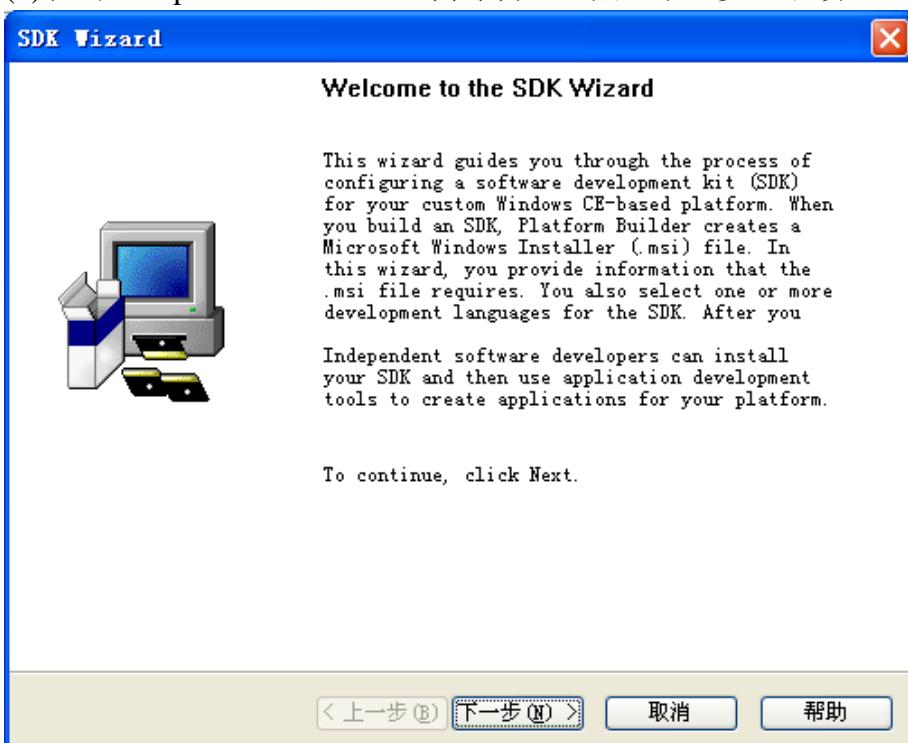
TO BE BEST

TO DO GREAT

广州友善之臂计算机科技有限公司



(2)跳出“Export SDK Wizard”向导窗口，点“下一步”继续：



(3)进入“Prodcut Properties”配置窗口，可以根据实际情况填写配置，点“下一步”



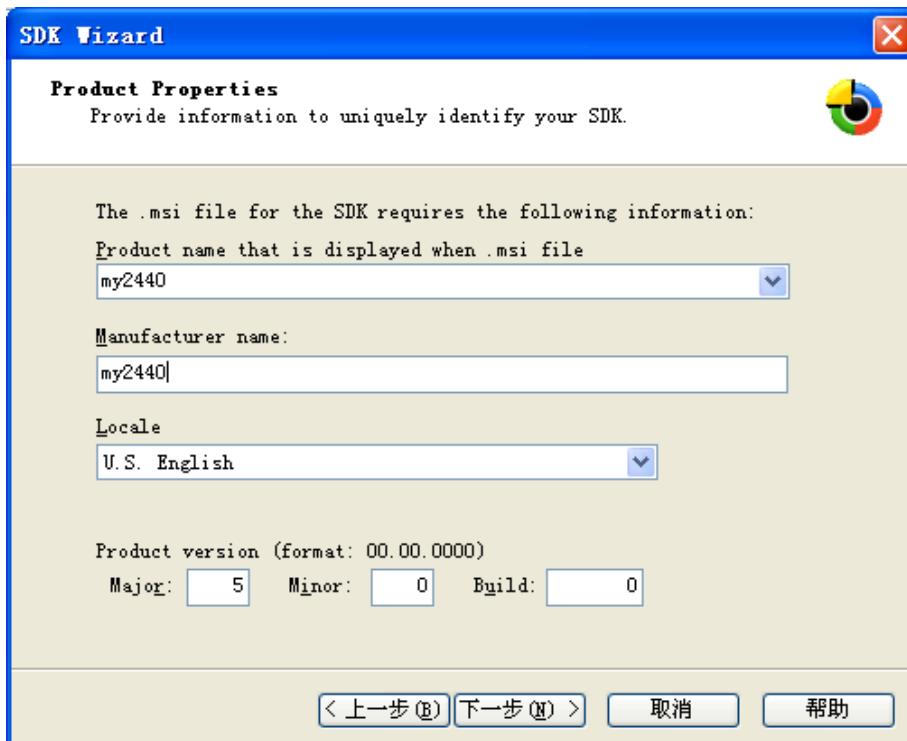
追求卓越 创造精品

TO BE BEST

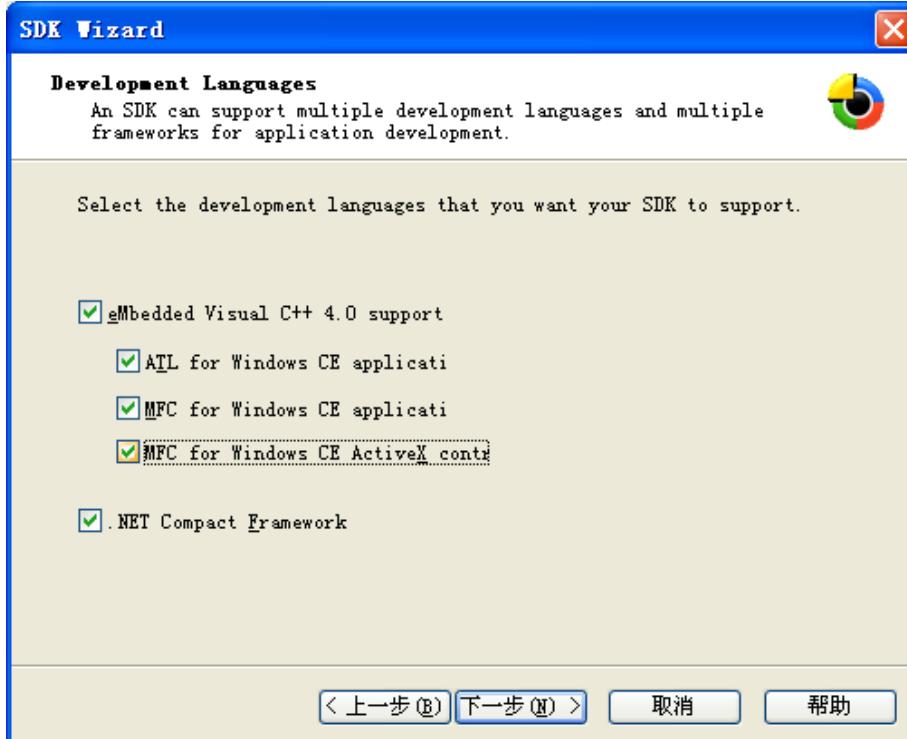
TO DO GREAT

广州友善之臂计算机科技有限公司

继续，如图：



(4)进入“Development Language”配置窗口，选择开发语言支持，点“下一步“继续：



(5)配置完毕，点“Finish”按钮结束。



追求卓越 创造精品

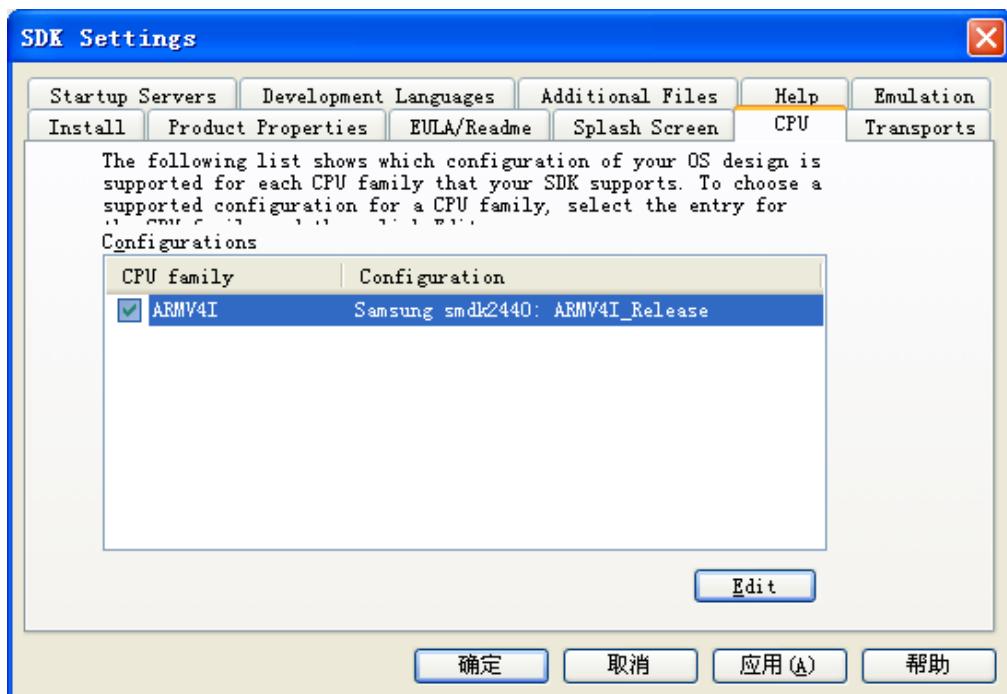
TO BE BEST

TO DO GREAT

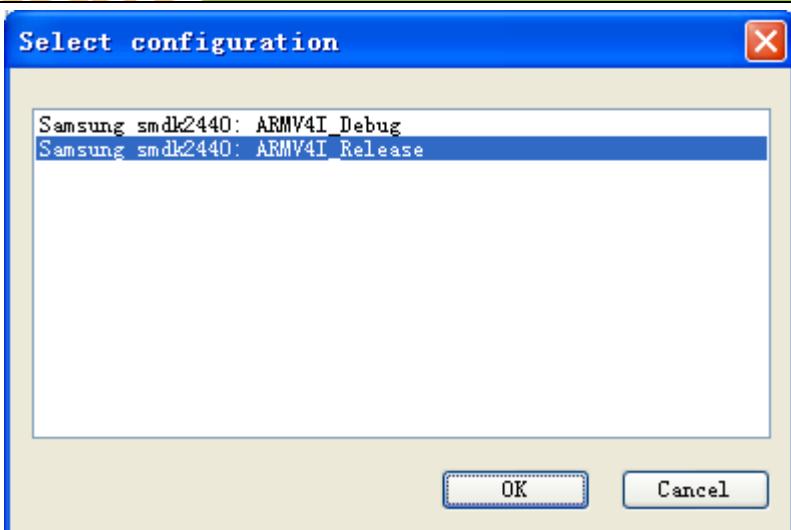
广州友善之臂计算机科技有限公司



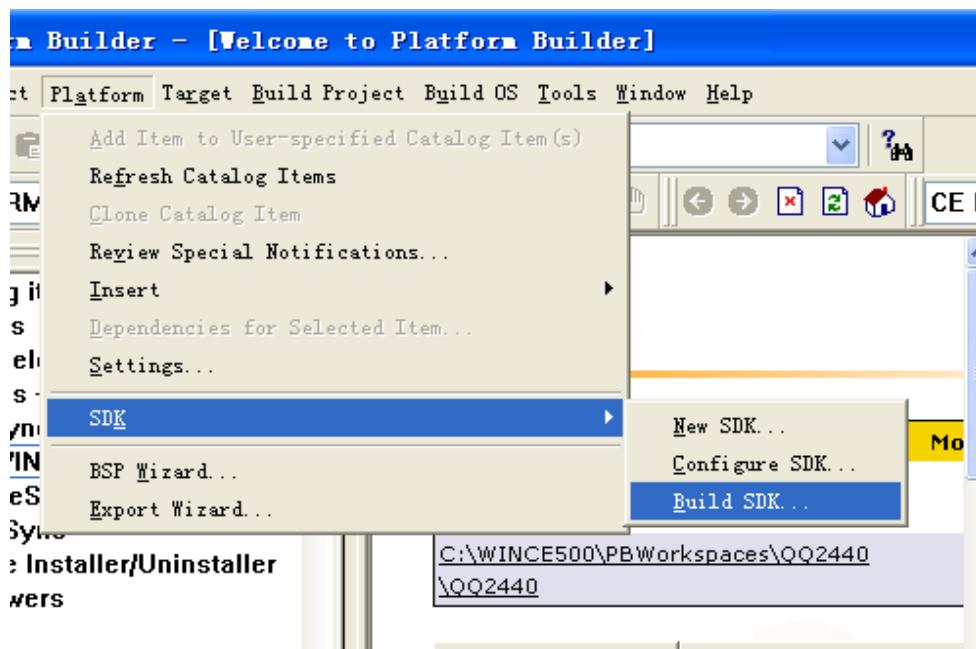
(6) 点 Platform → SDK → Configure SDK...，出现设置窗口，在这里你可以对刚才的初始配置进行更加详细的设置，点“CPU”选项卡，出现如图界面：



(7) 点“Edit”按钮，出现如图界面，并如图选择：



(8)点“OK”返回 PB5 主界面，再点 Platform → SDK → Build SDK...:



(8)出现编译向导窗口，并同时开始编译制作 SDK:

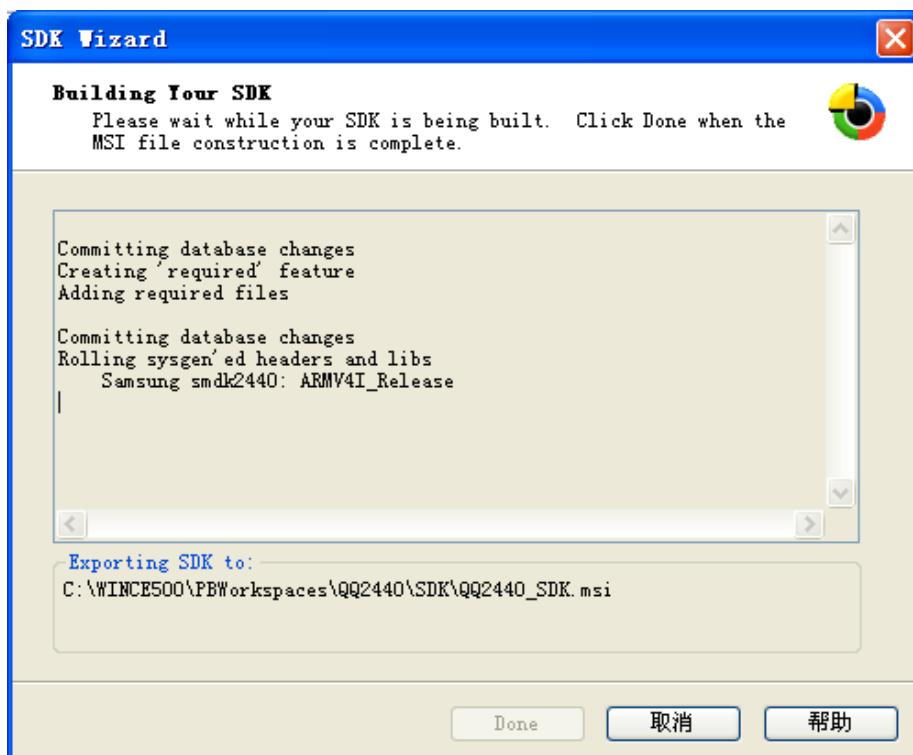


追求卓越 创造精品

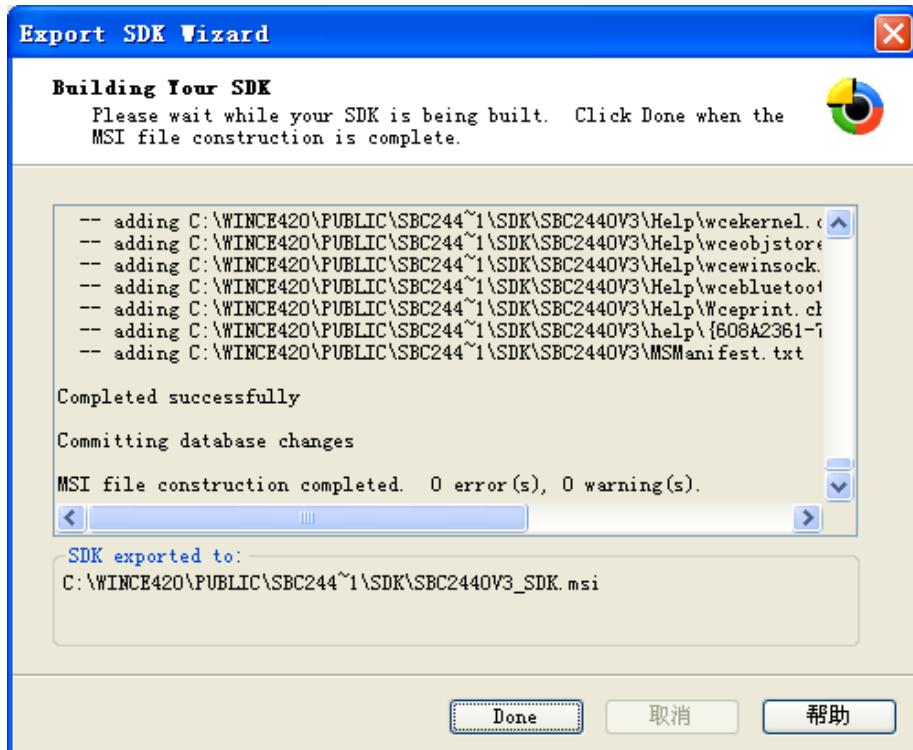
TO BE BEST

TO DO GREAT

广州友善之臂计算机科技有限公司



(9)大概几分钟时间，编译完毕，此时点“Done”按钮结束：



(10)根据提示，最后在如图目录生成 SDK 安装文件：



追求卓越 创造精品

TO BE BEST

TO DO GREAT

广州友善之臂计算机科技有限公司



### 9.1.6 安装 Embedded Visual C++(EVC)

为了开发基于 API 的 WinCE 应用程序，需要安装 EVC 集成开发环境和相应的 SDK，下面是详细的 EVC 安装步骤：

(1) EVC 安装文件位于光盘 Embedded VisualC++\ 目录中，双击 setup.exe 开始安装



追求卓越 创造精品

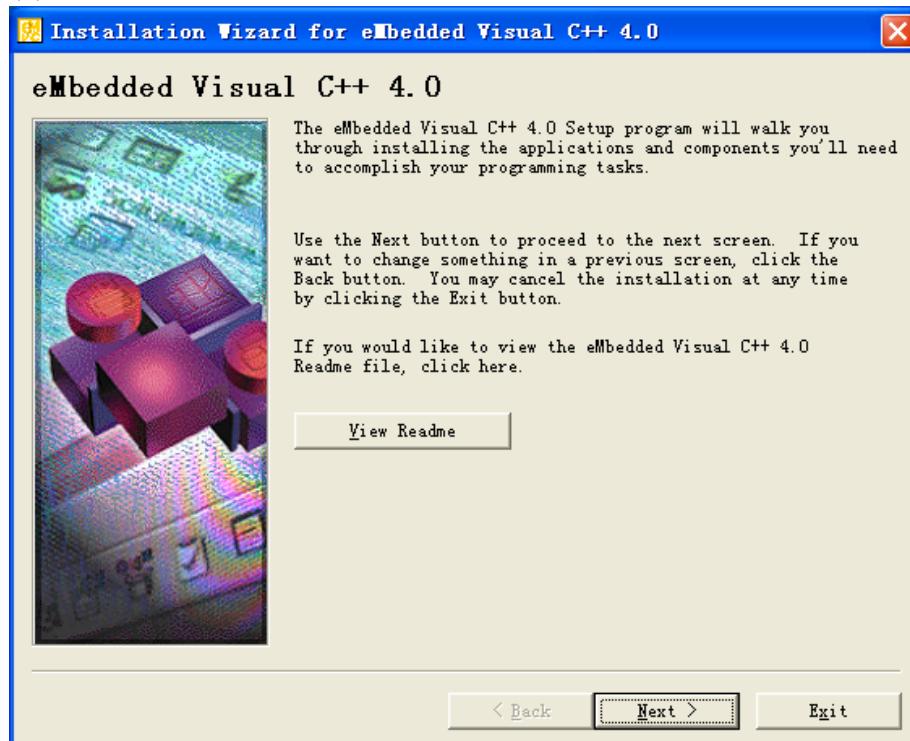
TO BE BEST

TO DO GREAT

广州友善之臂计算机科技有限公司



(2) 出现安装向导界面，点“Next”继续



(3) 出现用户许可协议，选择“I accept the agreement”点“Next”继续

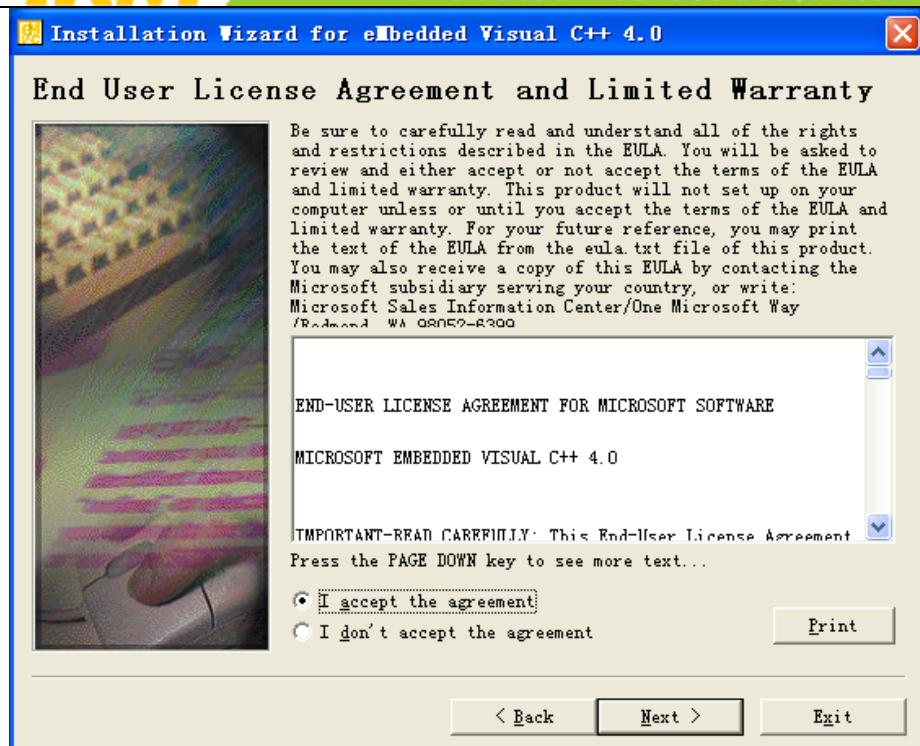


追求卓越 创造精品

TO BE BEST

TO DO GREAT

广州友善之臂计算机科技有限公司



(4)输入序列号和用户信息，点“Next”继续



(5)选择安装组件，选择默认全部安装即可，点“Next”继续

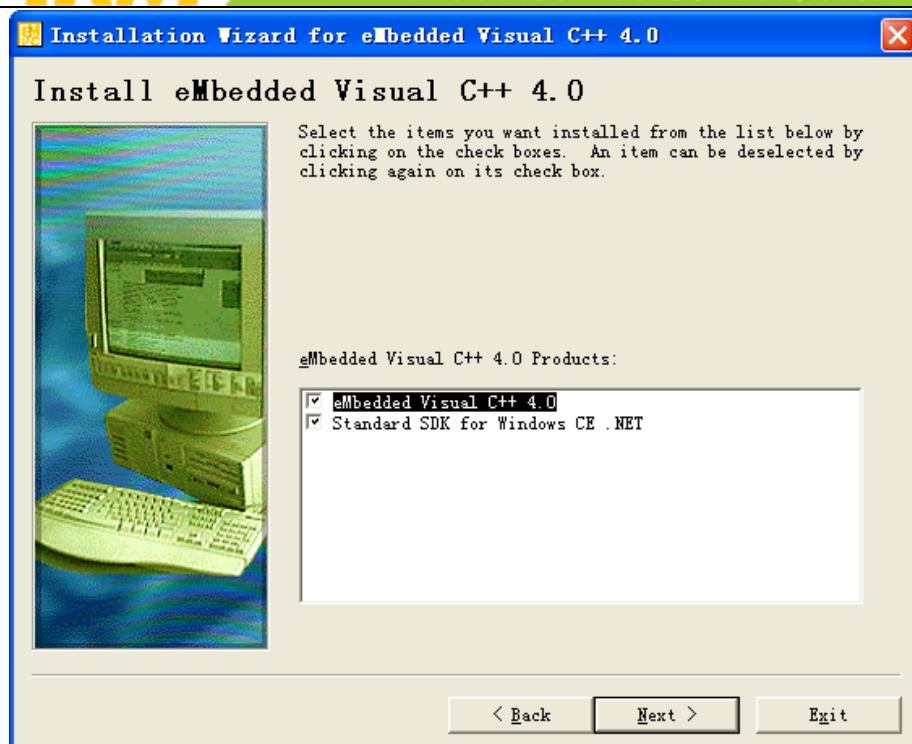


追求卓越 创造精品

TO BE BEST

TO DO GREAT

广州友善之臂计算机科技有限公司



(6)选择安装目录，这里按默认，点“Next”继续



(7)跳出提示窗口，选择“是”继续



追求卓越 创造精品

TO BE BEST

TO DO GREAT

广州友善之臂计算机科技有限公司



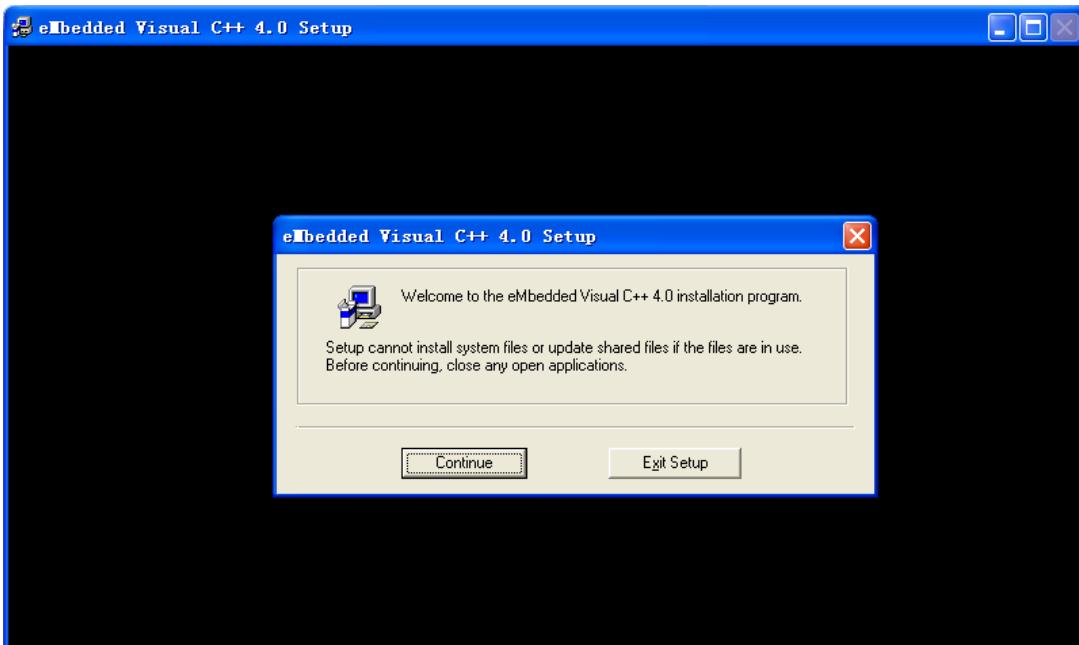
(8)开始安装 EVC 管理器，出现如图过程窗口，等待即可



(9)EVC 平台管理器安装完毕，点“OK”结束



(10)出现 EVC 安装界面，点“Continue”继续





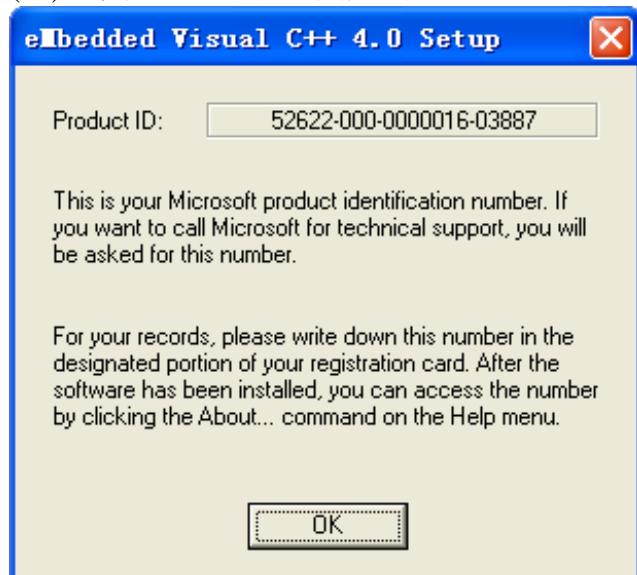
追求卓越 创造精品

TO BE BEST

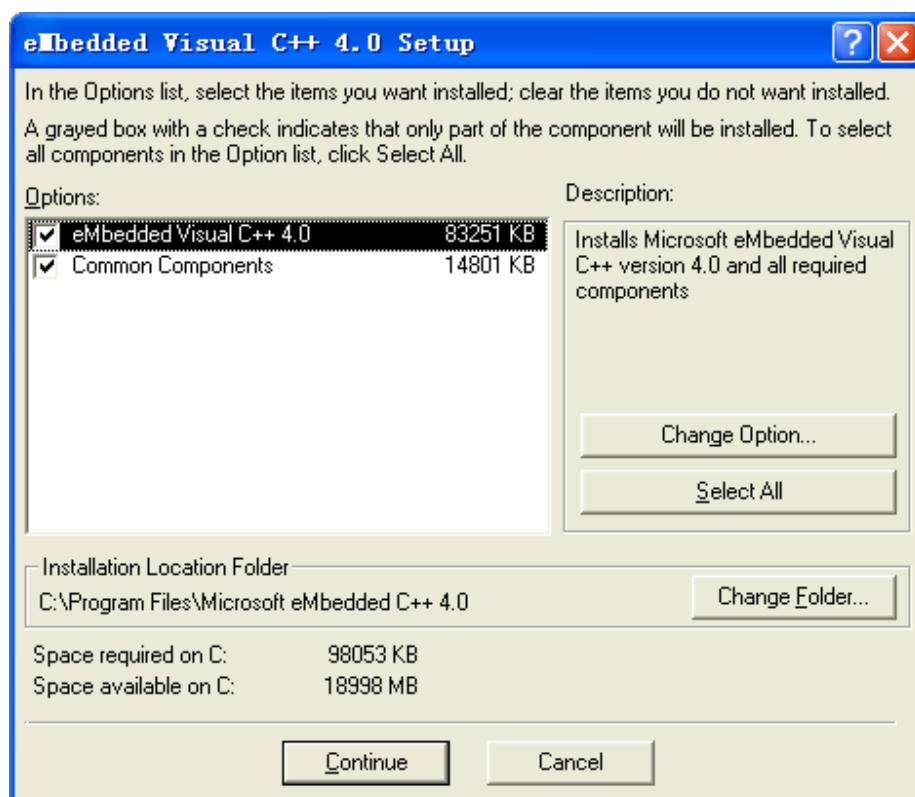
TO DO GREAT

广州友善之臂计算机科技有限公司

(11)出现产品 ID 号提示信息窗口，点“OK”继续



(12)选择安装组件和安装路径，按默认即可，点“Next”继续



(13)开始安装进程，如图

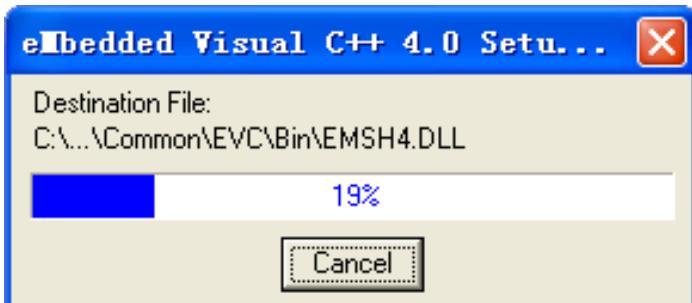


追求卓越 创造精品

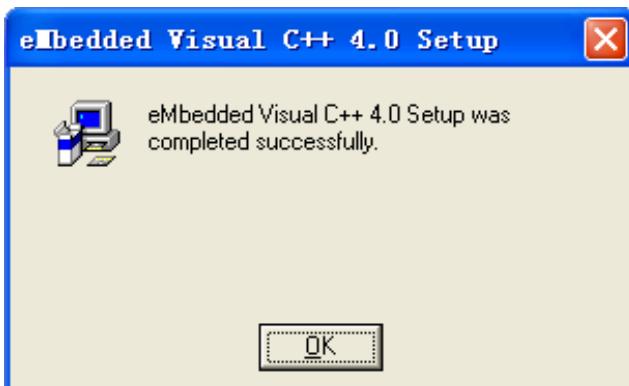
TO BE BEST

TO DO GREAT

广州友善之臂计算机科技有限公司



(14) 安装完毕，点“OK”结束



注意：此时如果点开始→程序→Microsoft eMbedded Visual C++ 4.0 → eMbedded Visual C++ 4.0 快捷菜单，尚不能正常运行，会出现以下提示窗口，因此你还需要按照下面的章节继续安装 SDK 才可以。



### 9.1.7 安装 EVC 补丁和导出的 SDK

为了能够正常使用我们导出的 SDK 安装文件，**必须要先安装 EVC 的 SP4 补丁文件，它位于\Embedded VisualC++\SP\evc4sp4\DISK1 目录中**，下面的步骤我们先安装补丁文件，再安装刚才我们导出的 SDK，如图：



(1) 双击运行 SP4 的安装程序 setup，出现安装向导界面，点“Next”继续



(2) 出现用户许可协议窗口，如图选择，点“Next”继续

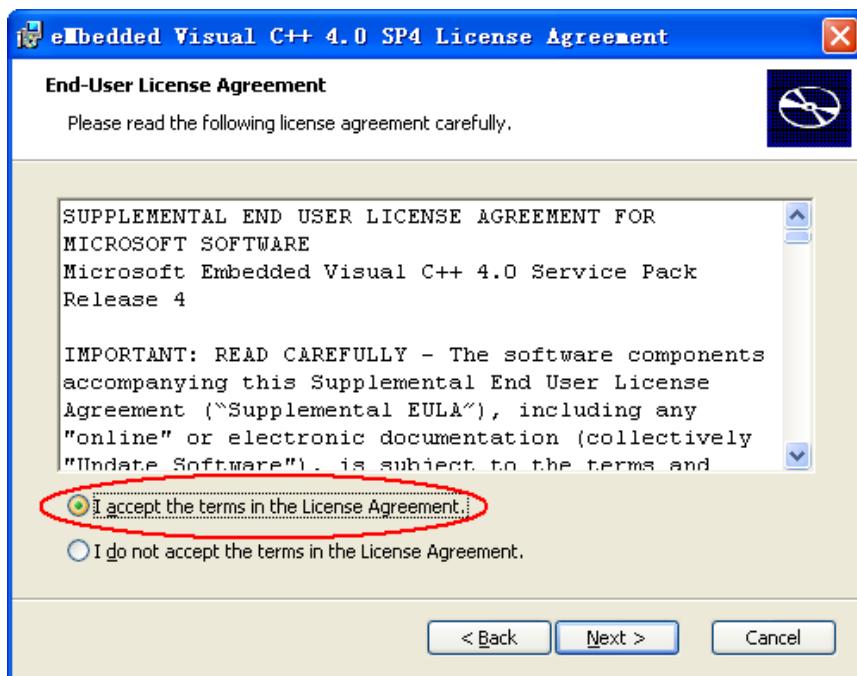


追求卓越 创造精品

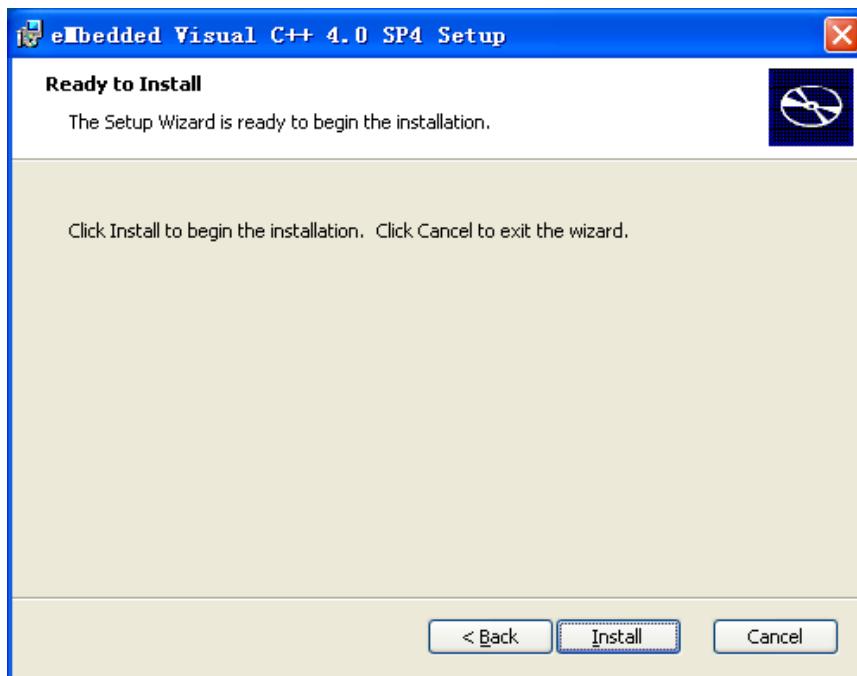
TO BE BEST

TO DO GREAT

广州友善之臂计算机科技有限公司



(3)准备安装 SP4，点“Next”继续



(4)开始安装进程，如图

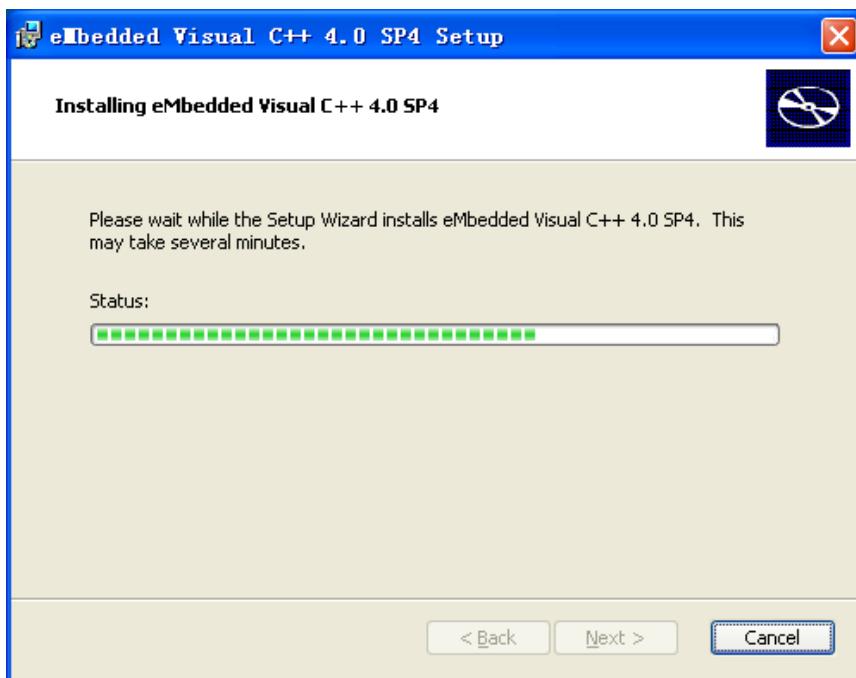


追求卓越 创造精品

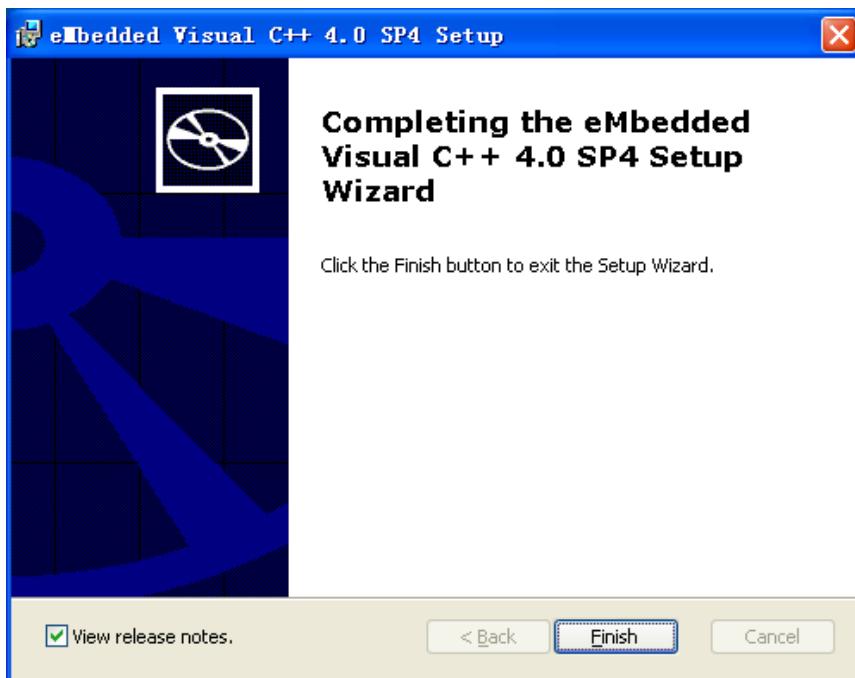
TO BE BEST

TO DO GREAT

广州友善之臂计算机科技有限公司



(5)安装结束，如图



(6)现在我们开始安装刚才导出的 SDK，你可以使用自己导出的，也可以在光盘中找到我们已经做好的（它位于光盘的 WindowsCE5.0\SDK 文件夹中，名字为“QQ2440\_SDK.msi.exe”），双击运行它，出现安装向导窗口，点“Next”继续：



追求卓越 创造精品

TO BE BEST

TO DO GREAT

广州友善之臂计算机科技有限公司



(7)这时会出现如下提示窗口，不必理会，点“Close”关闭掉，继续



(8)出现用户许可协议窗口，选择“Accpet”，点“Next”继续：



追求卓越 创造精品

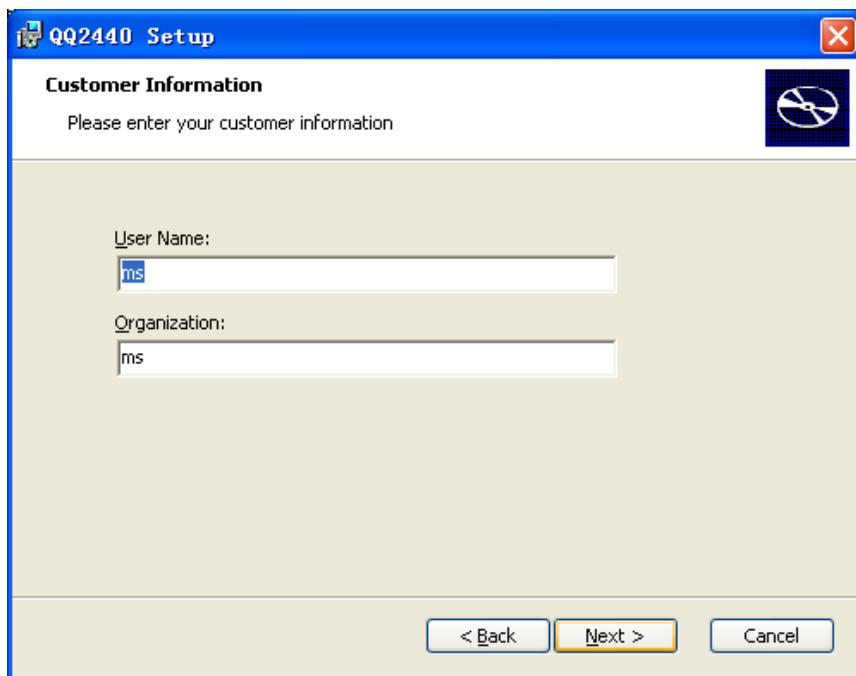
TO BE BEST

TO DO GREAT

广州友善之臂计算机科技有限公司



(9)出现“Customer Information”窗口，输入相应信息，点“Next”继续：



(10)出现安装类型窗口，点“Complete”完全安装继续：

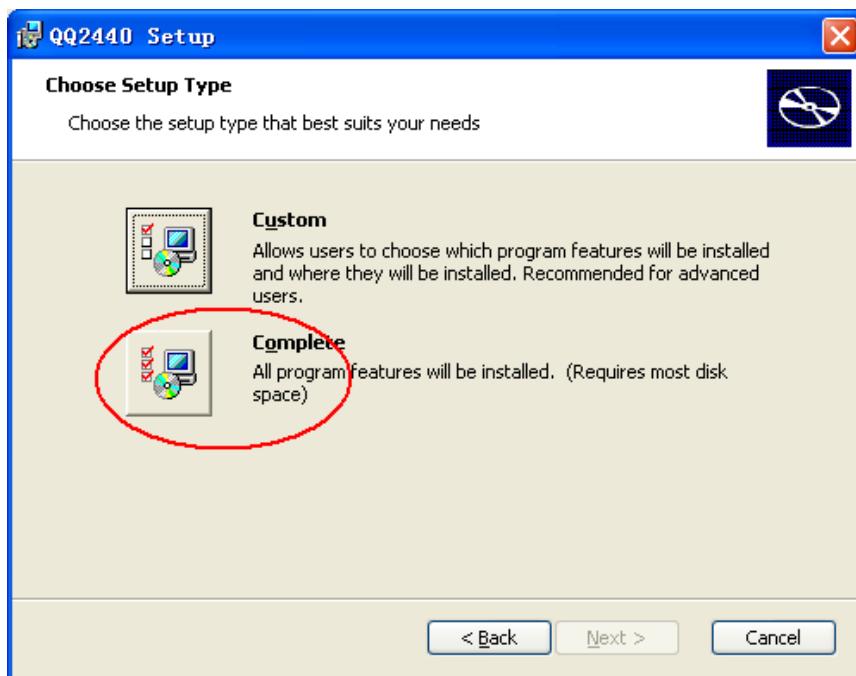


追求卓越 创造精品

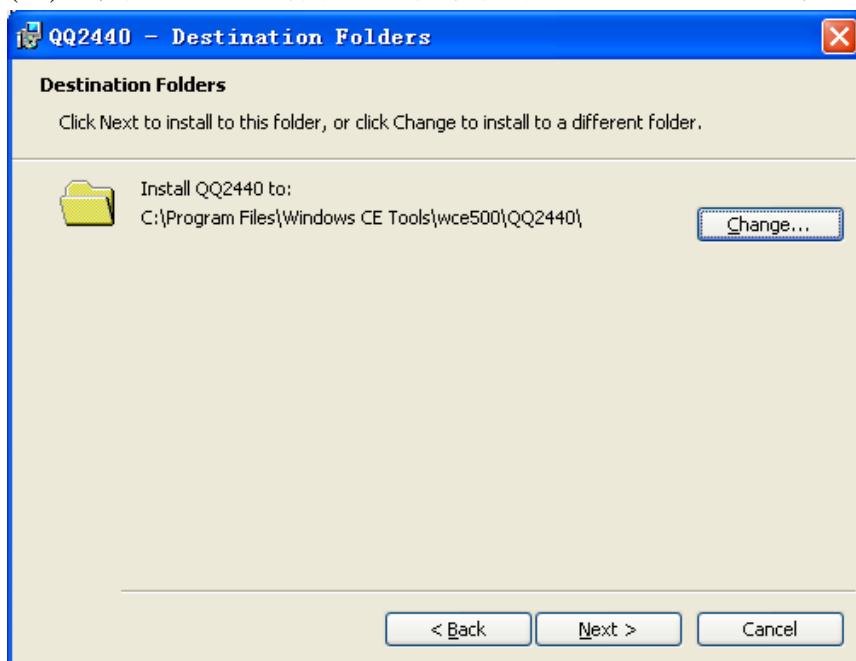
TO BE BEST

TO DO GREAT

广州友善之臂计算机科技有限公司



(11) 出现安装目录选择窗口，按默认即可，点“Next”继续：



(12) 在Ready to Install对话框中,点击“Install”安装, 如图:

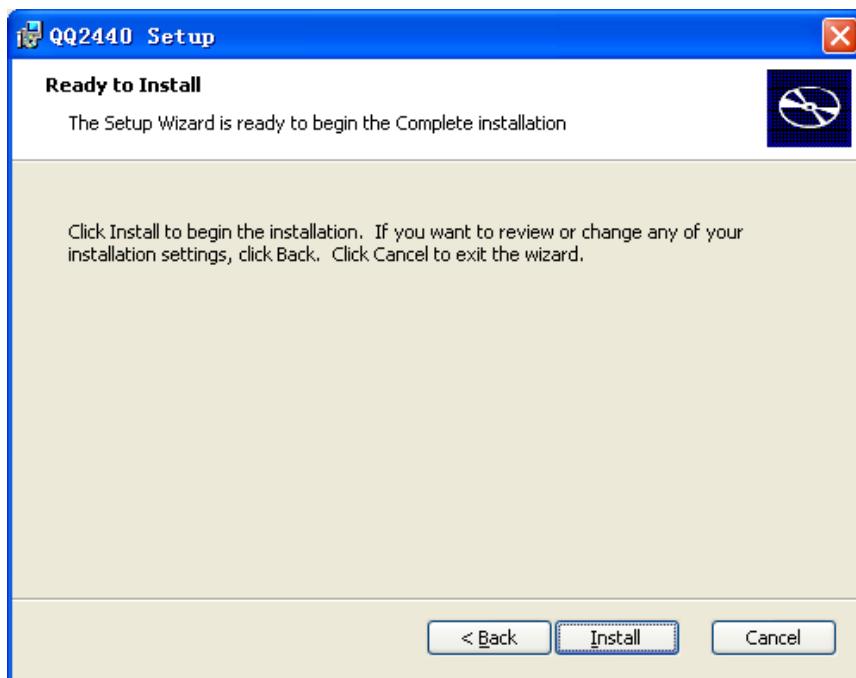


追求卓越 创造精品

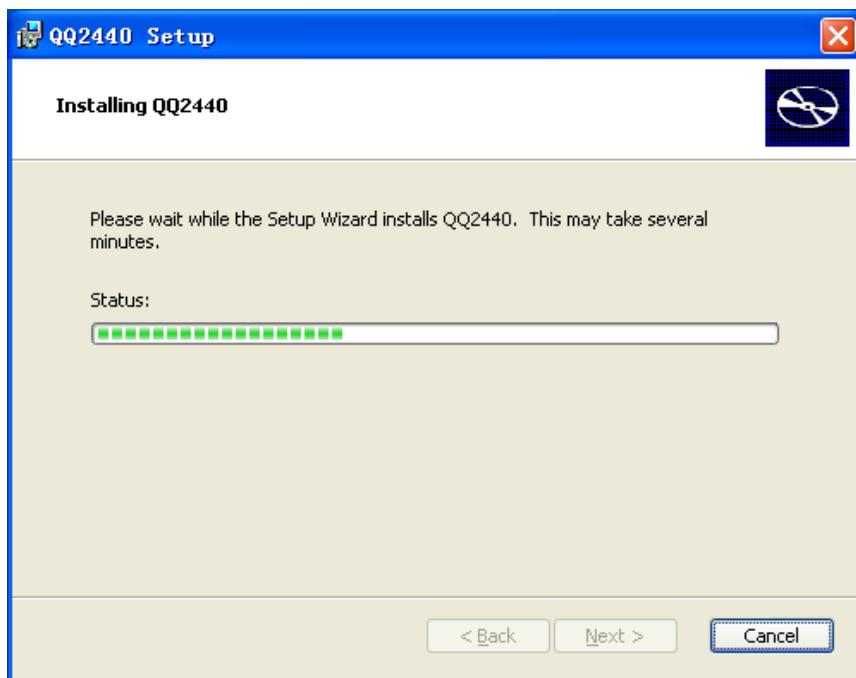
TO BE BEST

TO DO GREAT

广州友善之臂计算机科技有限公司



(13)系统开始安装，如图：



(14)安装完毕，点“Finish”结束：



追求卓越 创造精品

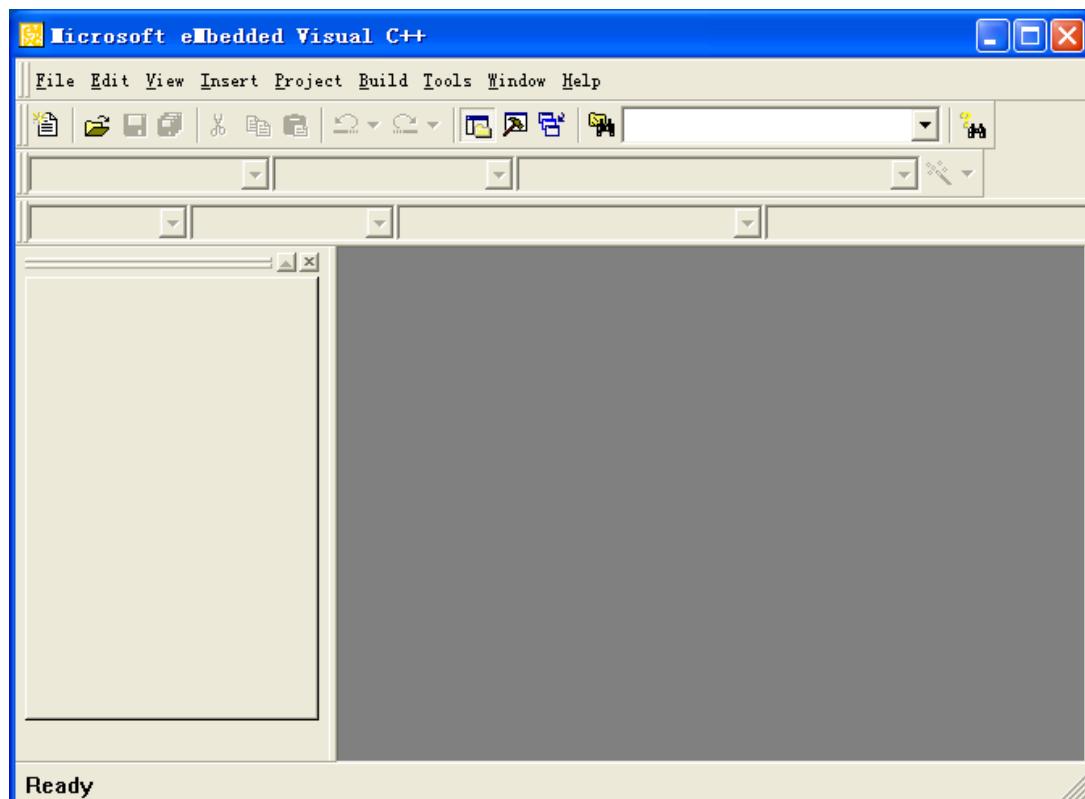
TO BE BEST

TO DO GREAT

广州友善之臂计算机科技有限公司



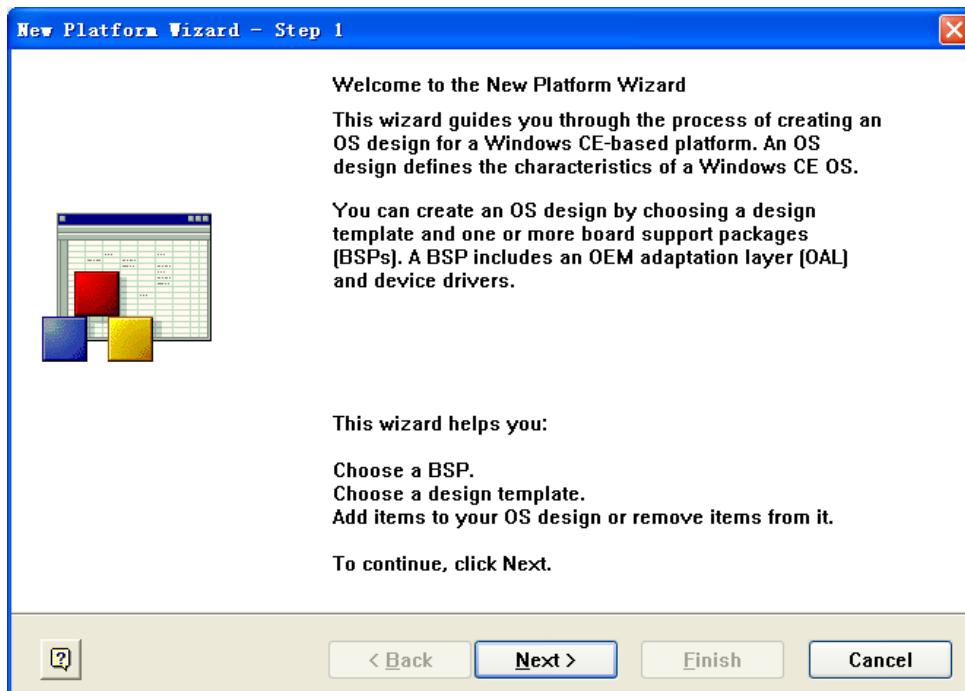
现在，你就可以点开始→程序→Microsoft eMbedded Visual C++ 4.0 → eMbedded Visual C++ 4.0 快捷菜单，打开 EVC 集成开发环境了，如图：



## 9.1.8 定制 CE 内核

通过以上步骤，我们了解了如何安装 BSP，编译内核工程，导出 SDK 等常用操作，现在我们使用图解的方式介绍一下如何定制适合于自己的 WinCE 内核工程。

(1) 打开 PB5，点 File → New Platform...，跳出内核定制向导，点“下一步”继续：



(2) 出现工程命名设置窗口，输入“my2440”，点“Next”继续：

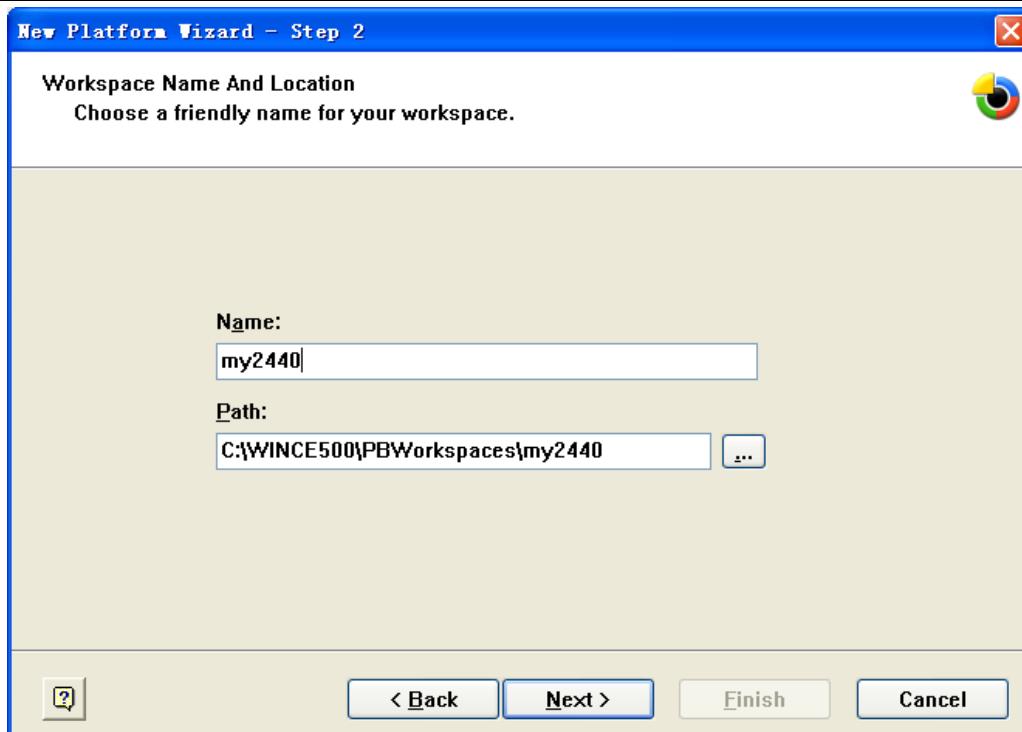


追求卓越 创造精品

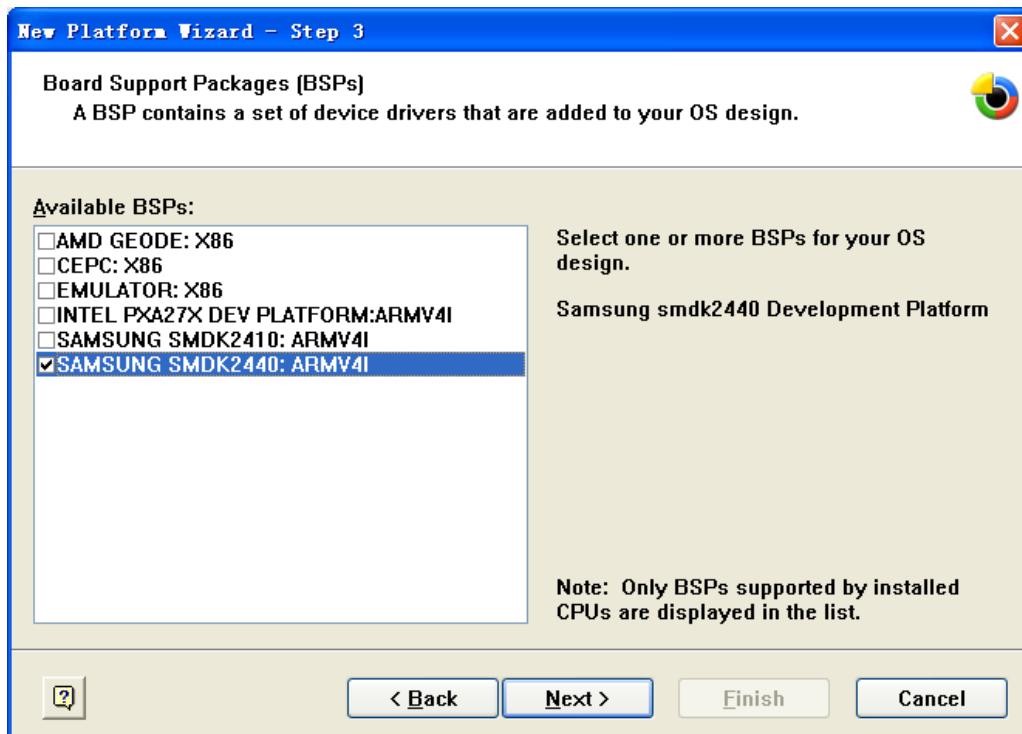
TO BE BEST

TO DO GREAT

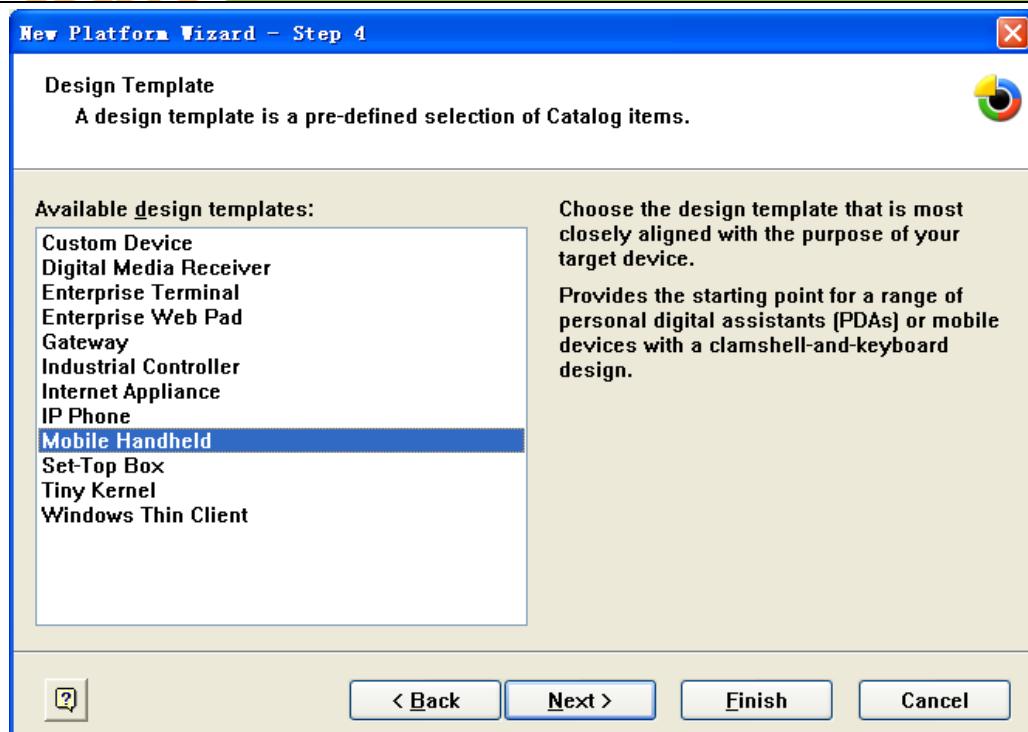
广州友善之臂计算机科技有限公司



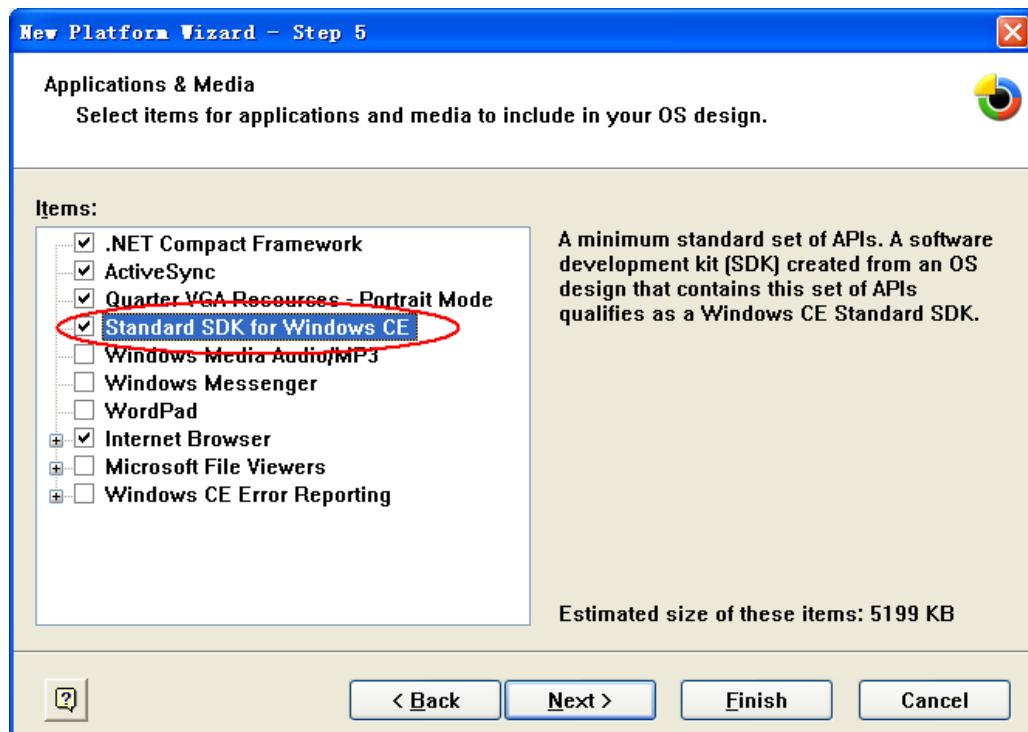
(3)选择所要使用的 BSP，在这里当然选择 2440 的，如图，点“Next”继续：



(4)出现模板选择窗口，我们选择手持设备“Mobile Handheld”，如图，点“Next”继续：



(5)出现应用程序定制窗口，这里出现的都是常见的一些应用程序，在这里**必须选择红色圈号里面的“Standard SDK for WindowsCE”**，如图，点“Next”继续：



(6)出现网络有关的一些设置，使用缺省即可，点“Next”继续：

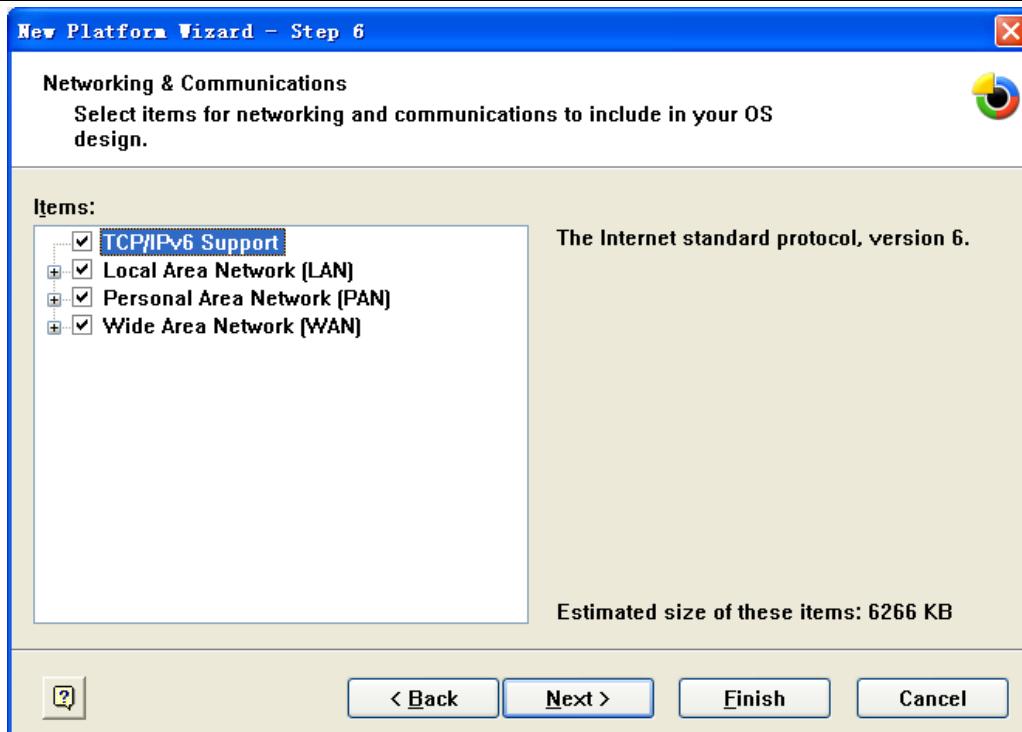


追求卓越 创造精品

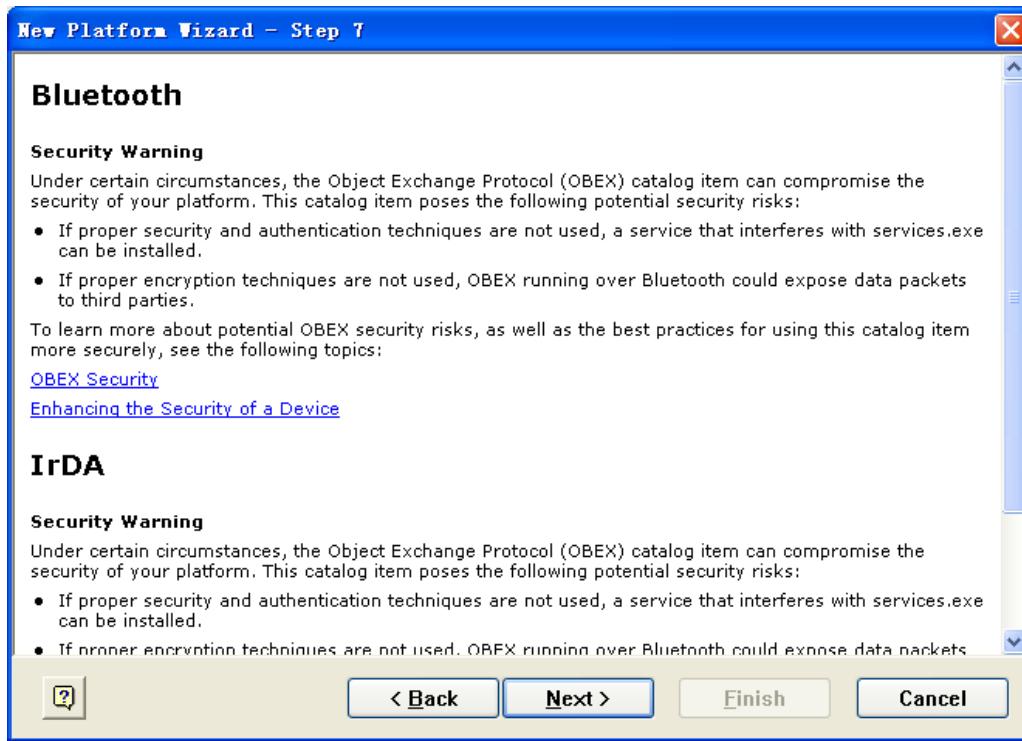
TO BE BEST

TO DO GREAT

广州友善之臂计算机科技有限公司



(7)出现和蓝牙、红外有关的一个窗口，这里没有任何选项，因此点“Next”继续：



(8)出现向导结束窗口，点“Finish”结束向导：

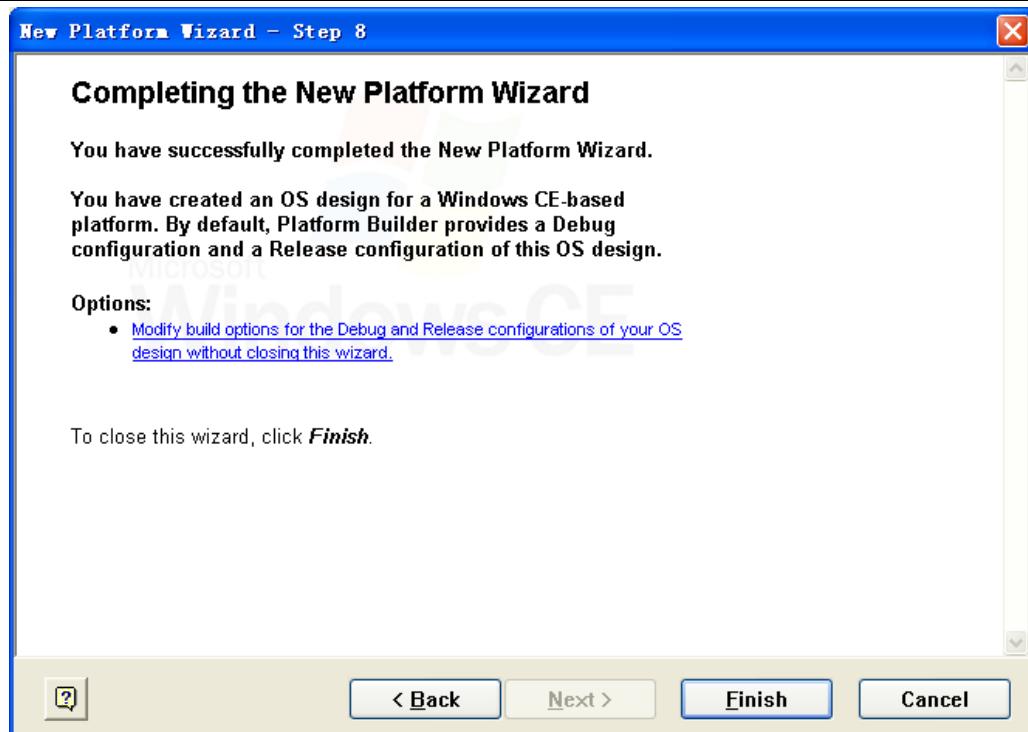


追求卓越 创造精品

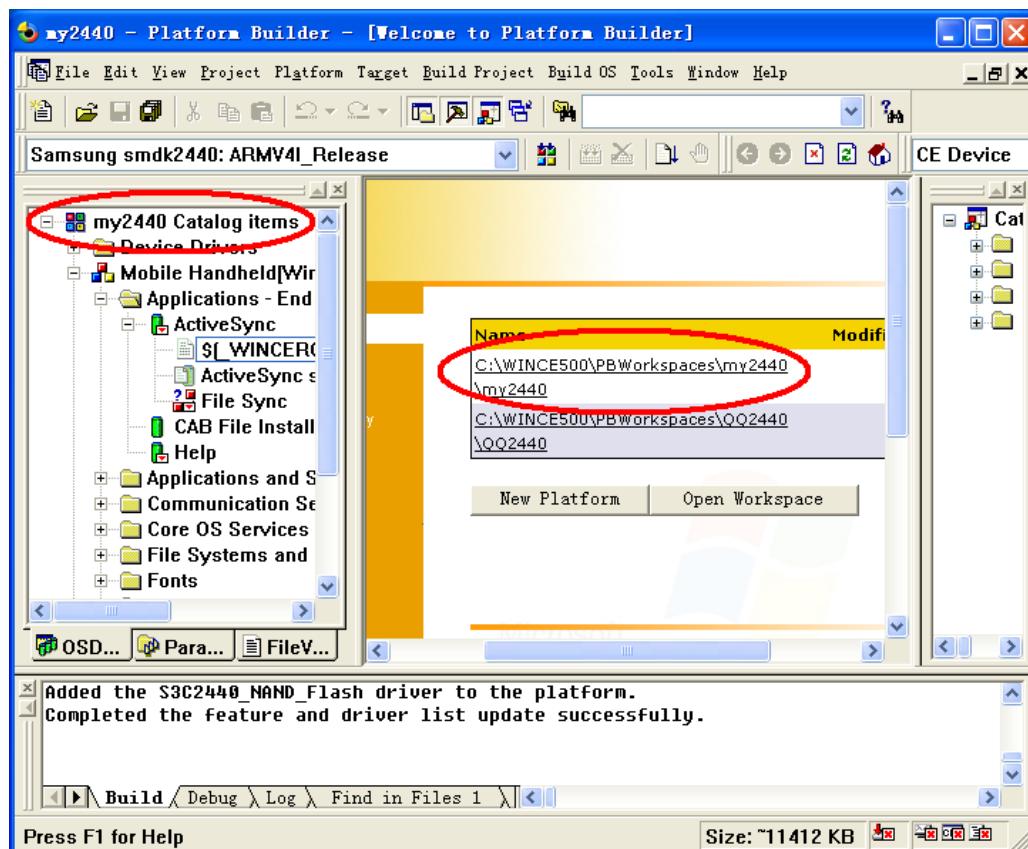
TO BE BEST

TO DO GREAT

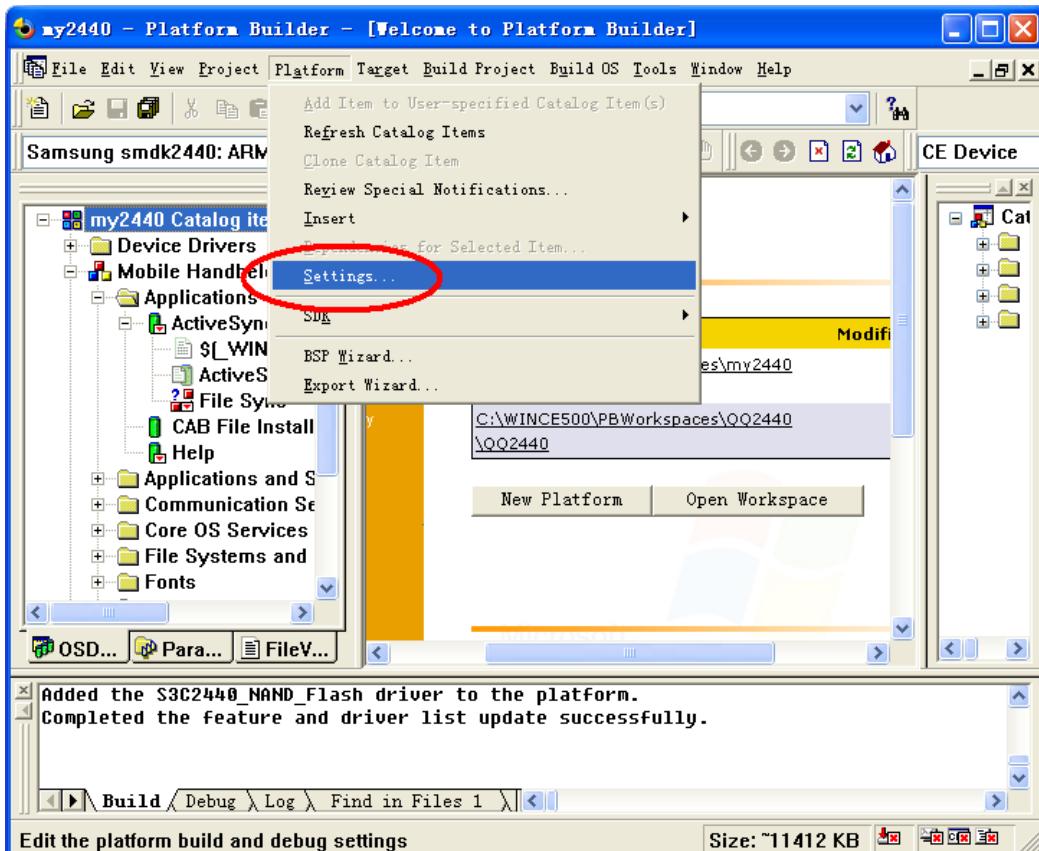
广州友善之臂计算机科技有限公司



(9)现在回到 PB5 的主窗口，可以看新的工程文件已经创建，我们还需要为编译做一下准备：



(10)点 Platform → Setting...， 打开工程设置窗口：



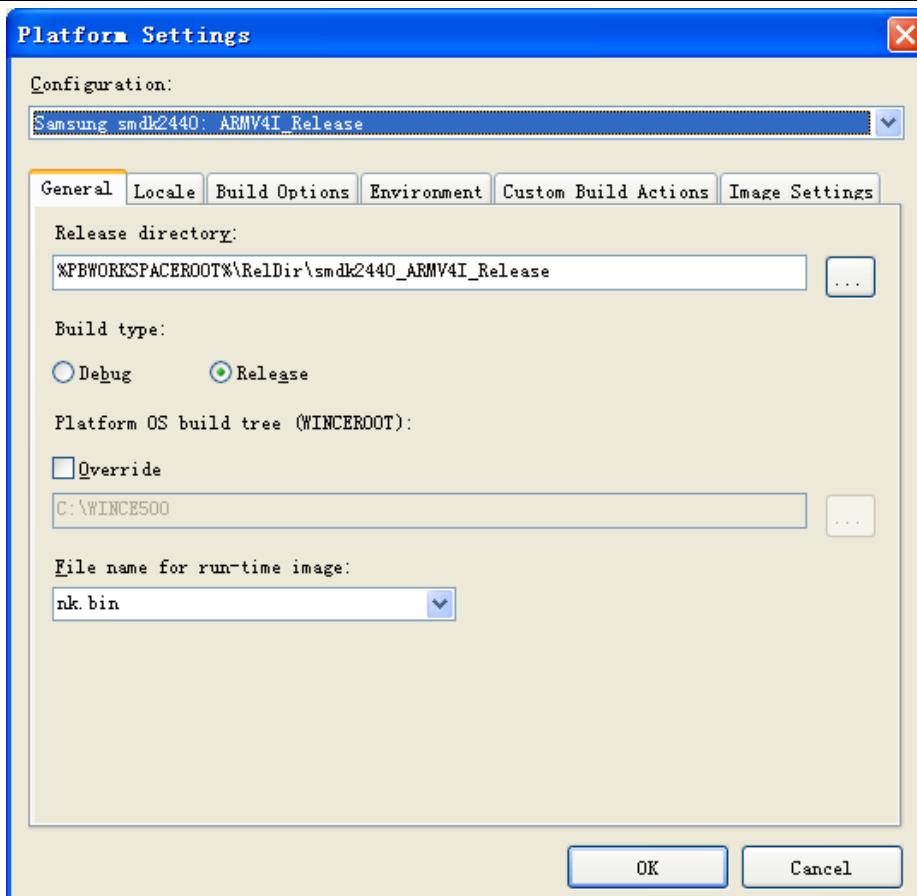


追求卓越 创造精品

TO BE BEST

TO DO GREAT

广州友善之臂计算机科技有限公司



(11)在设置窗口中，点“Locale”选项卡来设置目标内核的语言，这里选择中文：

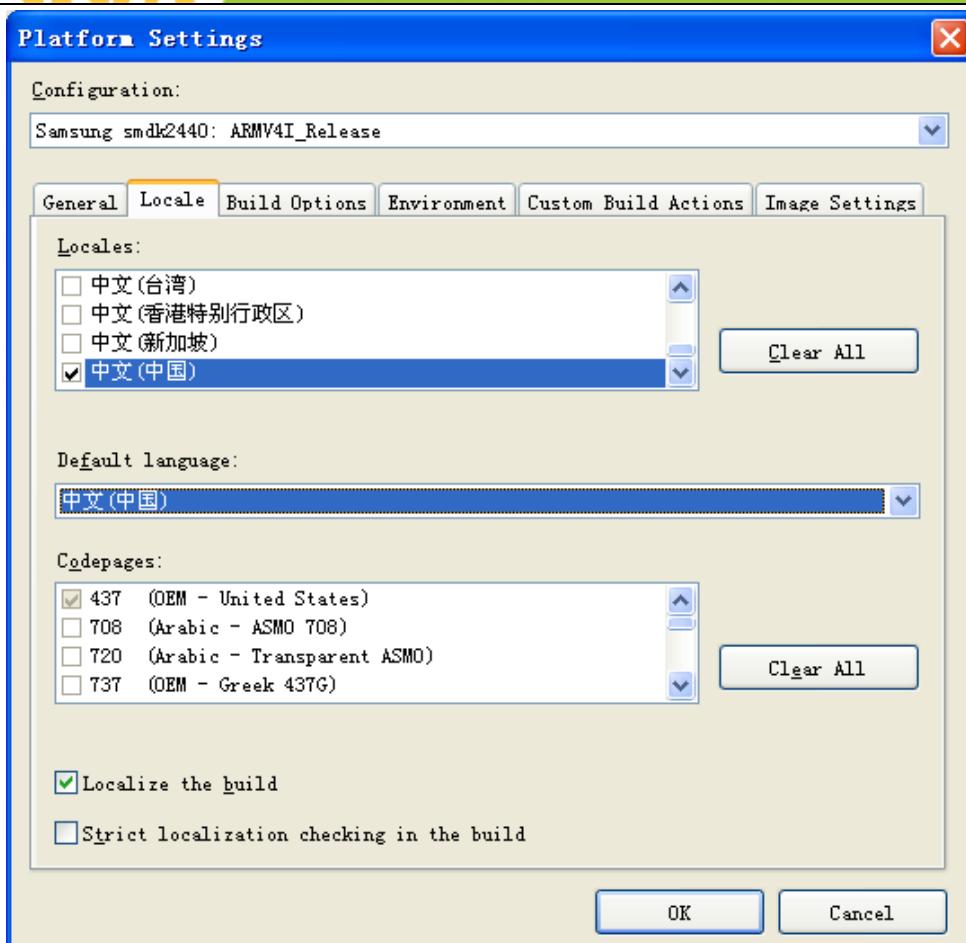


追求卓越 创造精品

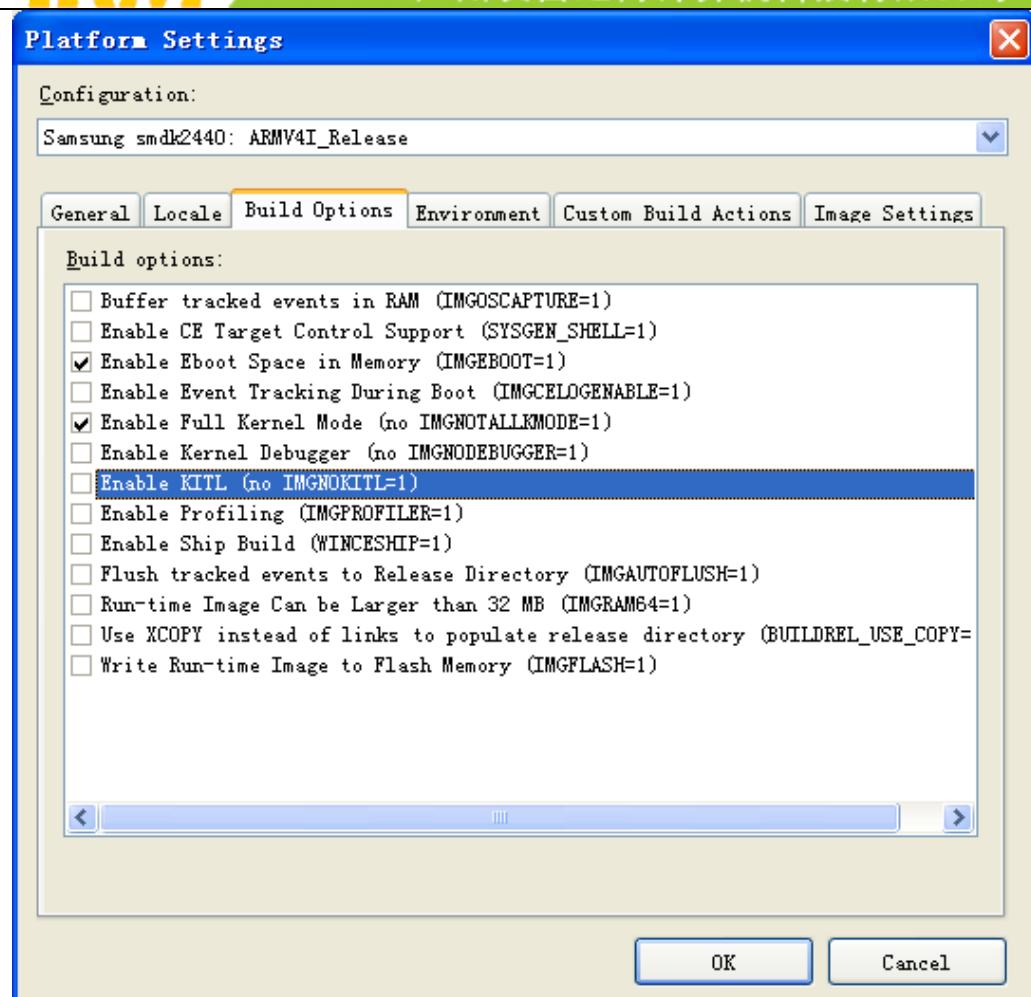
TO BE BEST

TO DO GREAT

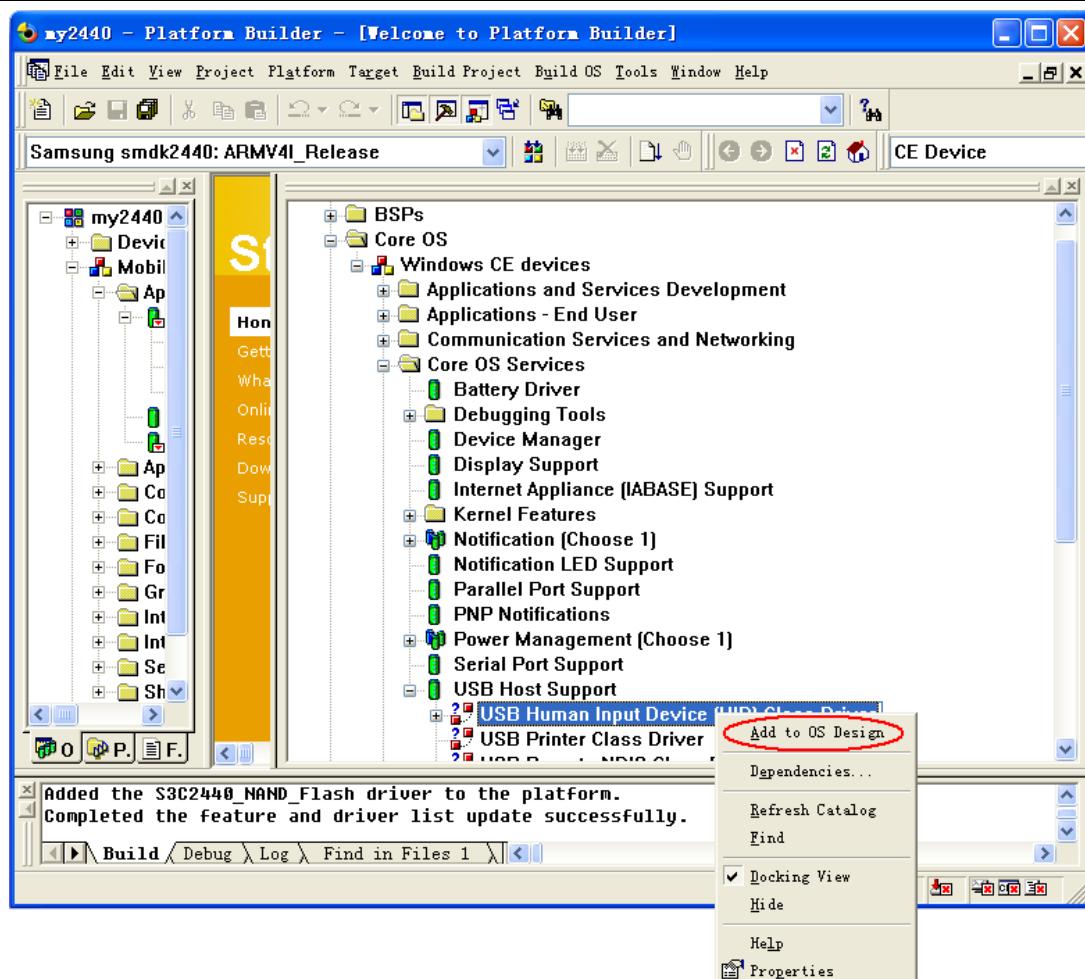
广州友善之臂计算机科技有限公司



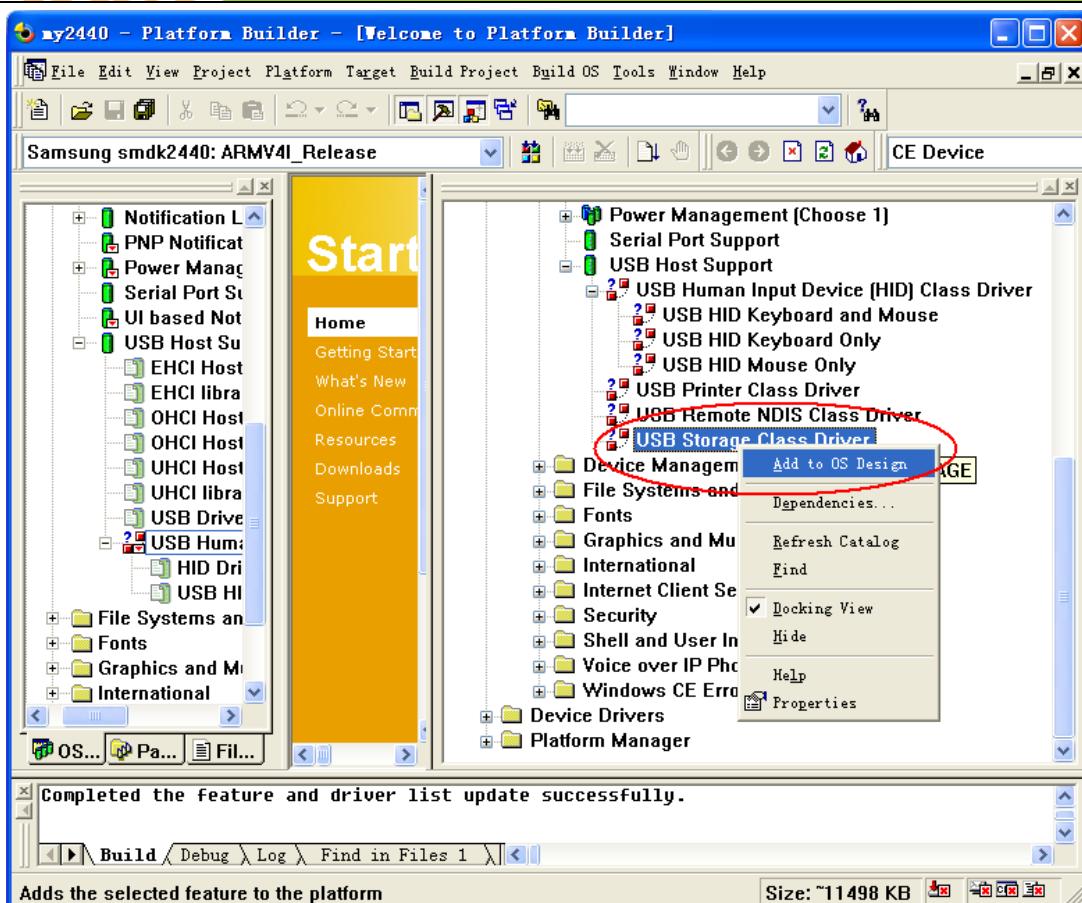
(12)点“Build Options”选项卡，去掉“Enable CE Target Control Support”和“Enable KITL”这两个设置，其他使用缺省，并点“OK”完成设置，如图：



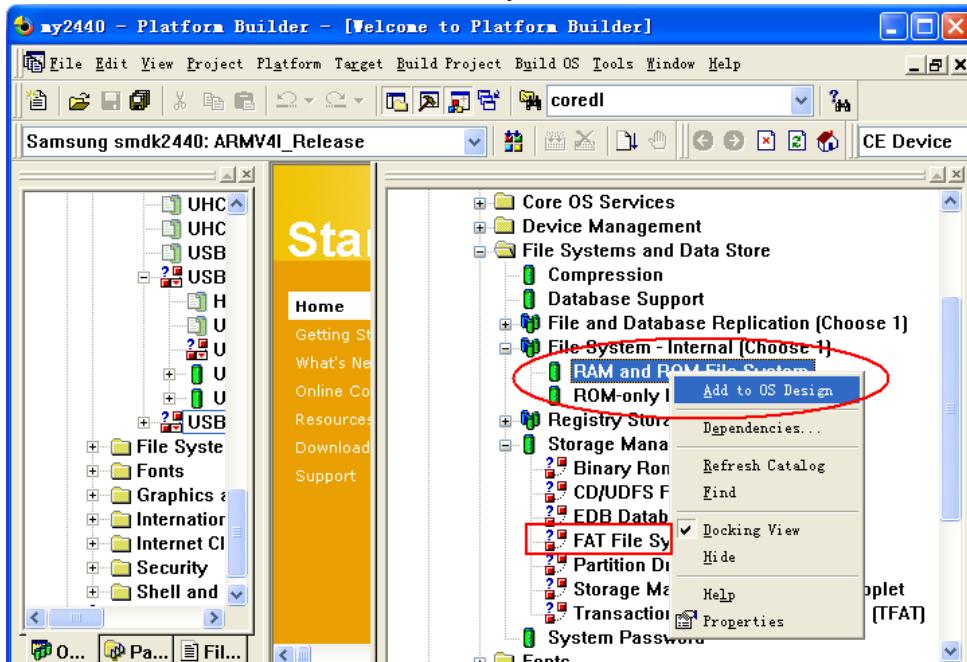
(13)加入USB鼠标和键盘的支持，在Catalog一栏依次点击展开Core OS → Windows CE device → Core OS Services → USB Host Support → USB Human Input Device(HID) Class Driver，点右键选择“Add to OS Design”，并展开其子项添加“USB HID Keyboard and Mouse”，如图：



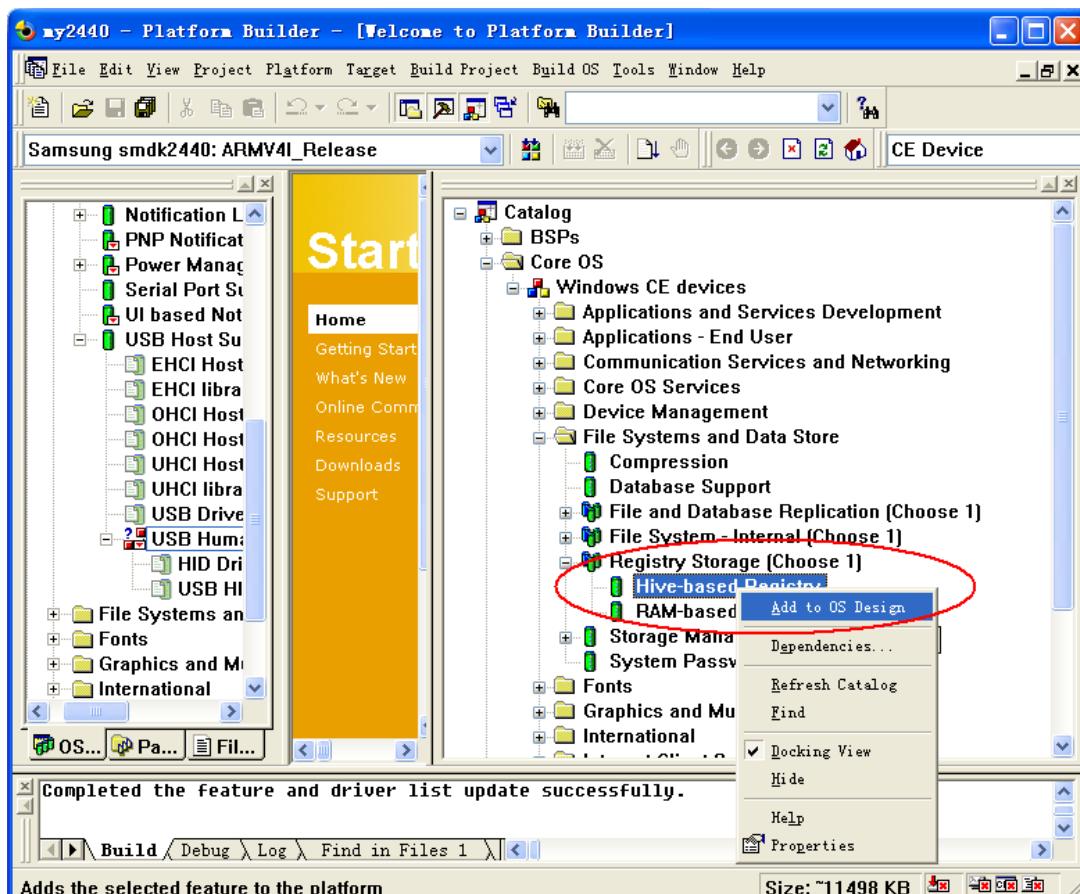
(14) 使用同样的方法可以添加 USB 存储设备的支持，如图：



(15)添加文件系统的支持，点 Core OS → File Systems and Data Store → File System – Internal (Choose I) → RAM and ROM File System，为了支持优盘所使用的 FAT32 文件系统，你还需要添加途中矩形方框中的 FAT File System，如图：



(16)添加注册表保存卡支持, 点 Core OS → File Systems and Data Store → Registry Storage (Choose 1) → Hive-based Registry, 如图:



(17)修改默认的 IP 地址, 打开 platform.reg 文件, 找到如图红色框所示的位置, 在这里你可以修改开机后默认的 IP 地址, 网关设置, DNS 等设置:

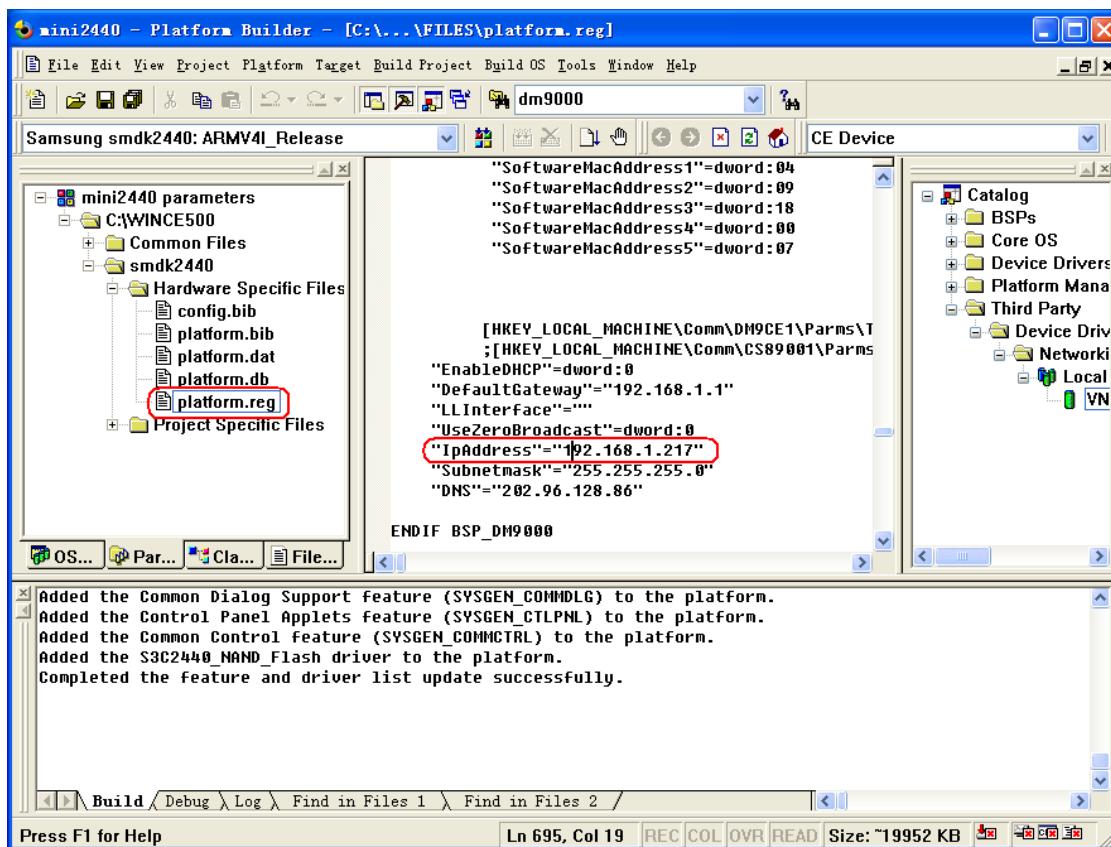


追求卓越 创造精品

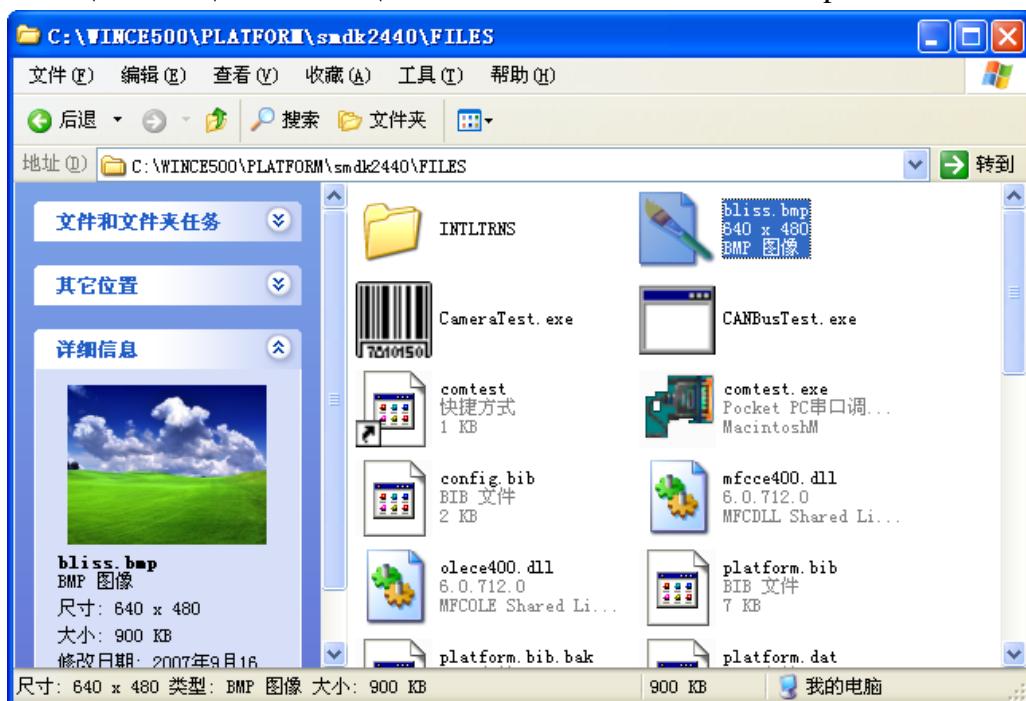
TO BE BEST

TO DO GREAT

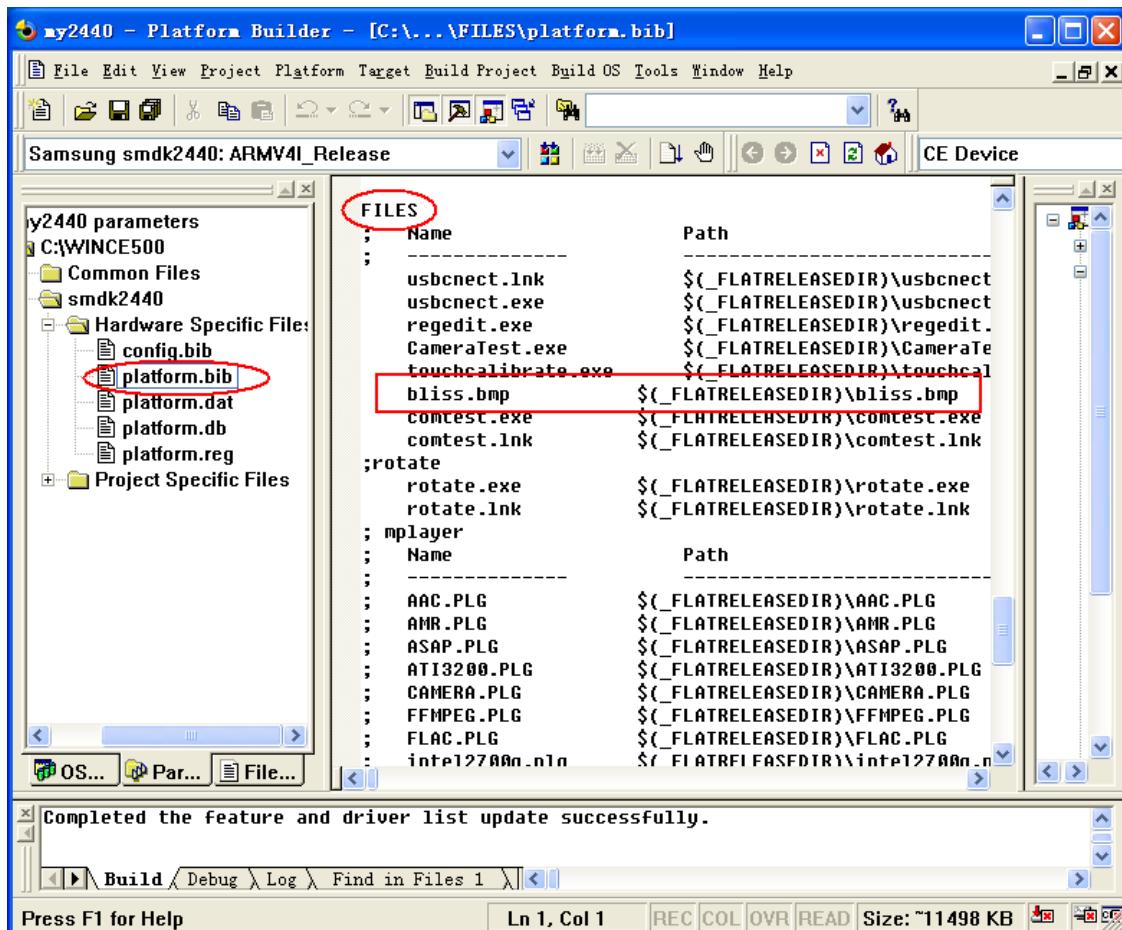
广州友善之臂计算机科技有限公司



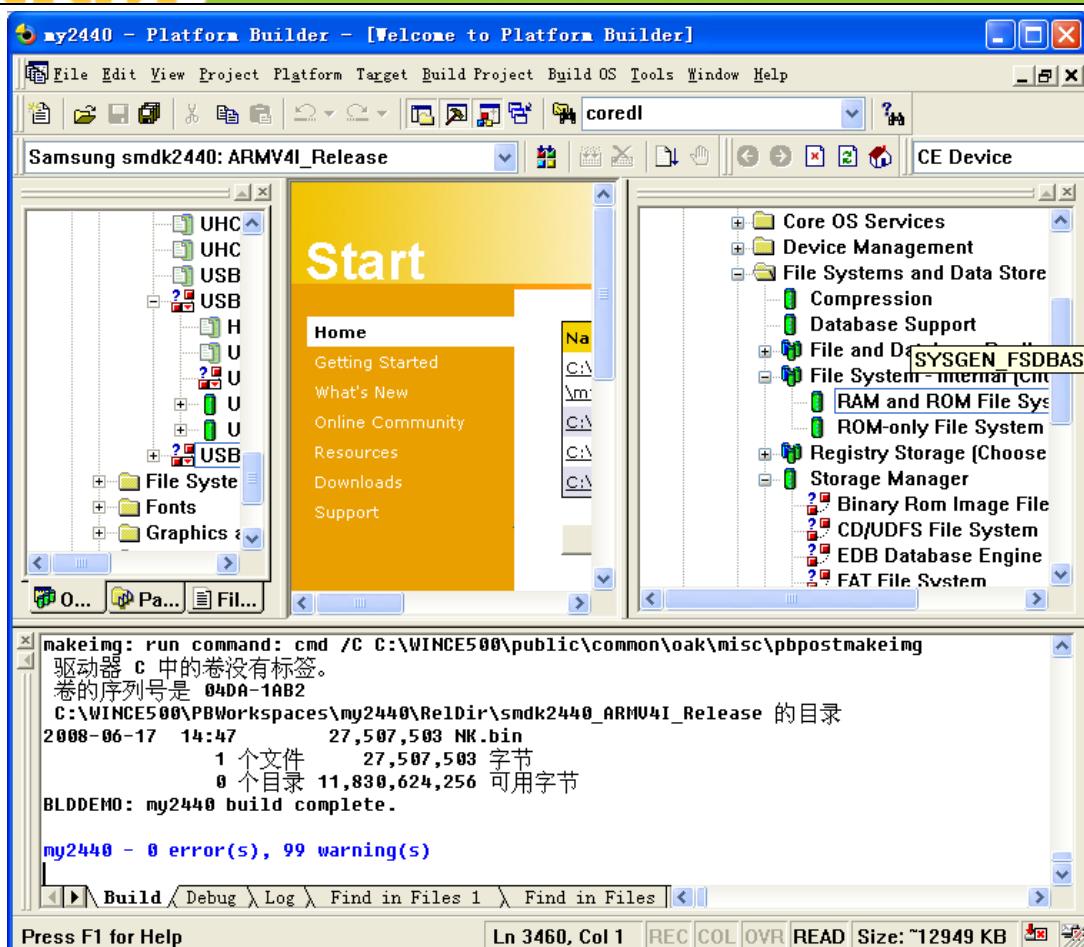
(18) 修改桌面背景图片，准备一个 bmp 格式的图片，复制到 C:\WINCE500\Platform\SMDK2440\Files 目录中，并命名为 bliss.bmp，如图：



打开 platform.bib 文件，在 FILES 栏目中把 bliss.bmp 添加进去，如图：



(19)保存以上设置，点 File → Save 即可，再点 Build OS → Sysgen 或者点工具栏的 按钮就开始编译内核了，编译完毕如图所示：



关于更多的内核配置和更改用户可以自己摸索尝试，或者参考相关网上资料等，在此不再叙述。

## 9.2 使用 ActiveSync 与 PC 同步通讯(公共)

使用微软提供的工具 ActiveSync，可以让 mini2440 与 PC 之间十分方便的进行通讯连接，从而实现文件上传，远程调试等功能。

### 9.2.1 安装 ActiveSync

在光盘的“windows 平台工具”目录中 ActiveSync 文件夹，双击运行里面的 ActiveSync\_4.1\_setup.exe 开始安装。

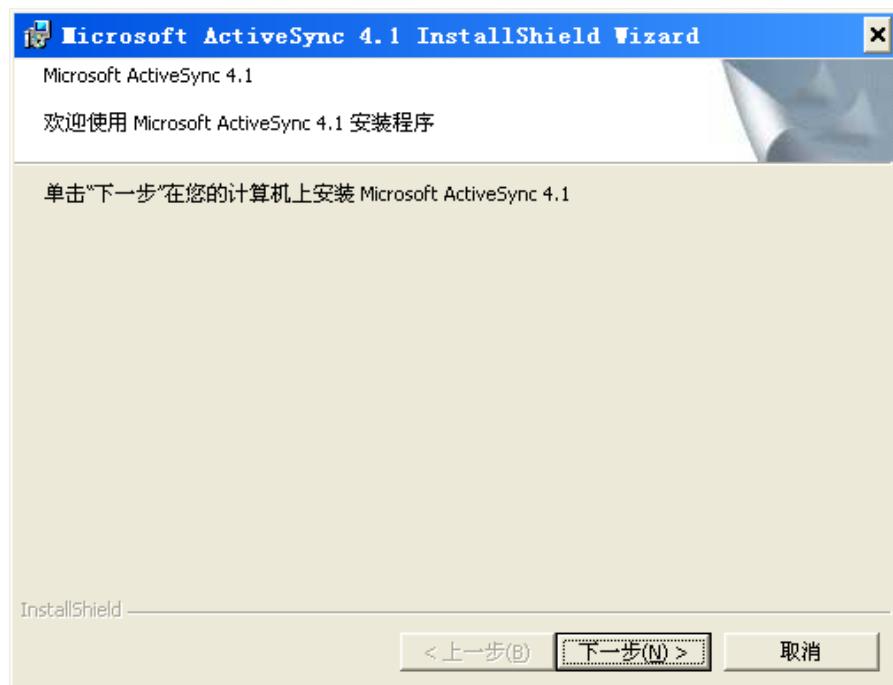


追求卓越 创造精品

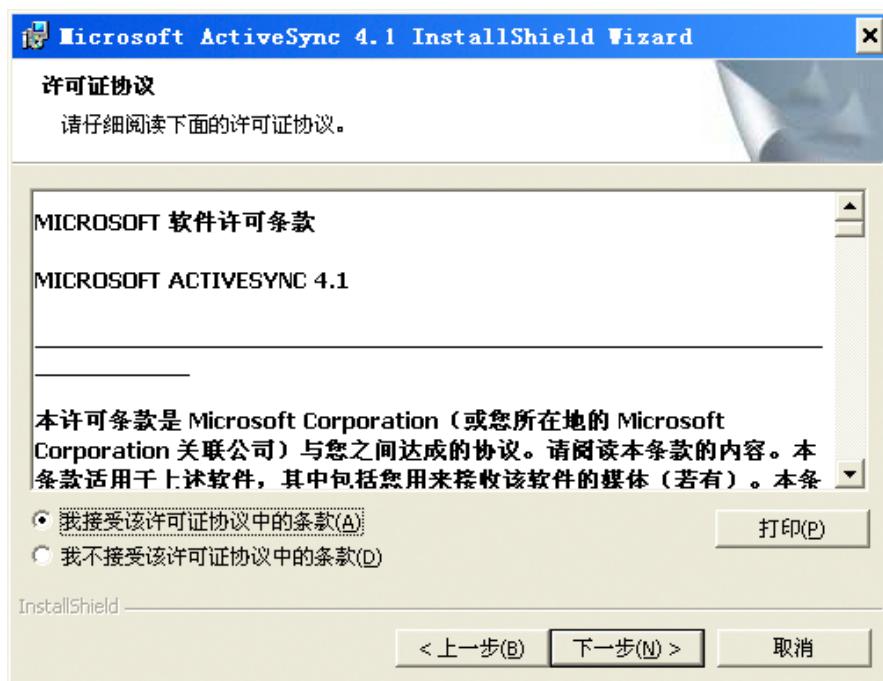
TO BE BEST

TO DO GREAT

广州友善之臂计算机科技有限公司



如图选择“我接受该许可证协议中的条款”，点“下一步”继续



输入用户名和单位名称，点“下一步”继续

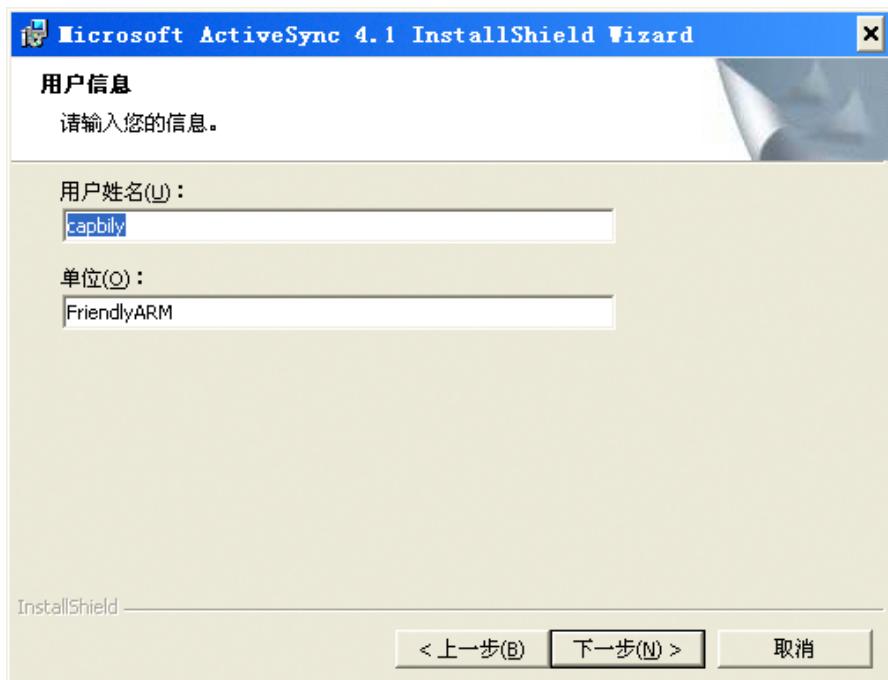


追求卓越 创造精品

TO BE BEST

TO DO GREAT

广州友善之臂计算机科技有限公司



选择要安装的目的路径，这里使用缺省值，点“下一步”继续。



出现如下界面，点“安装”开始进行安装。

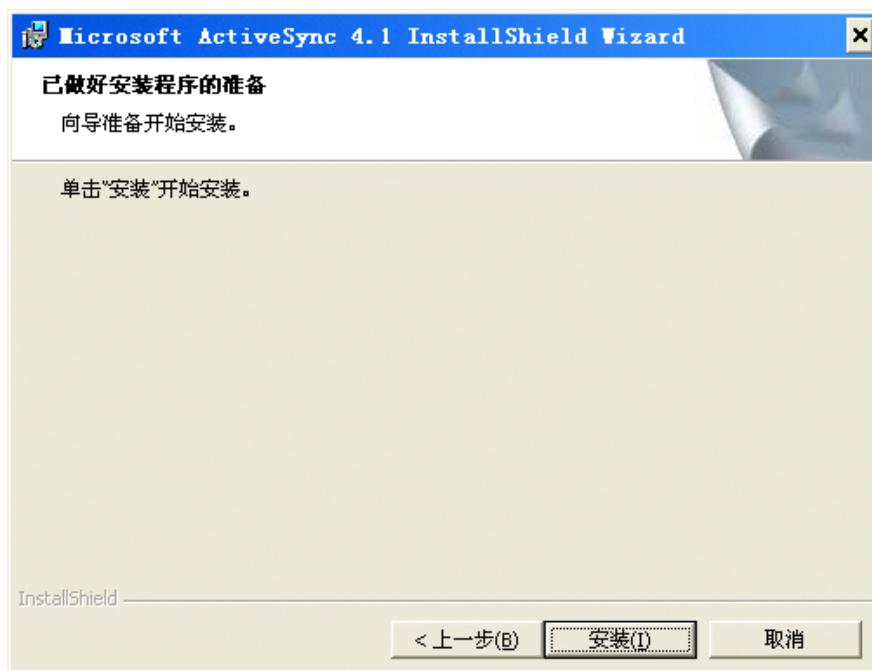


追求卓越 创造精品

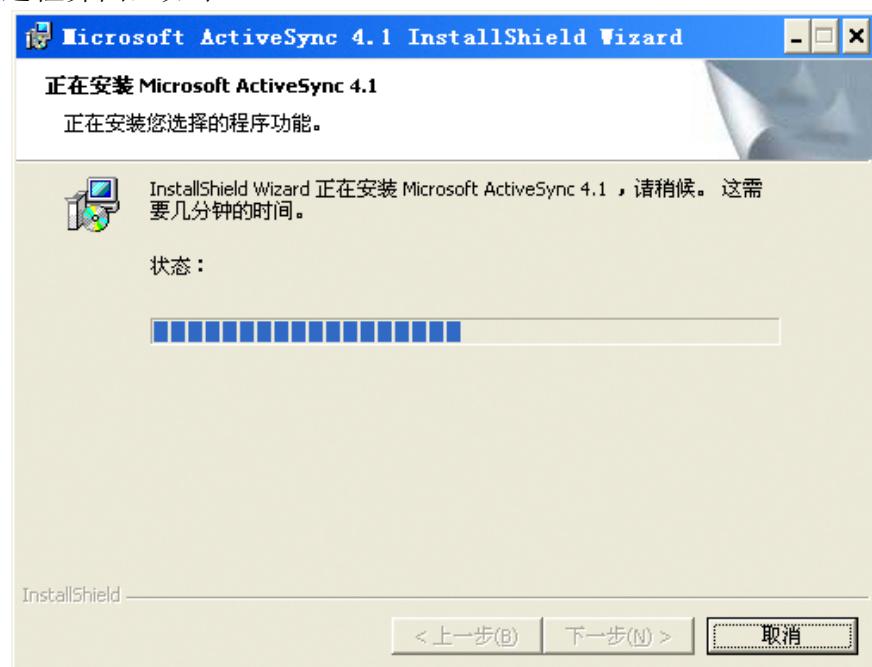
TO BE BEST

TO DO GREAT

广州友善之臂计算机科技有限公司



出现安装过程界面，如下



安装完毕，点“完成”安装完毕。

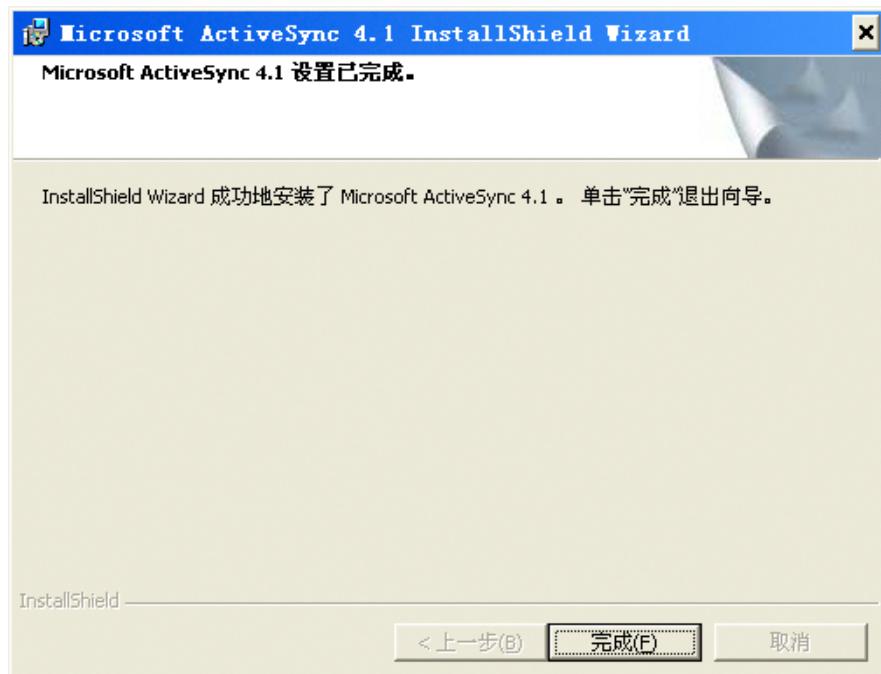


追求卓越 创造精品

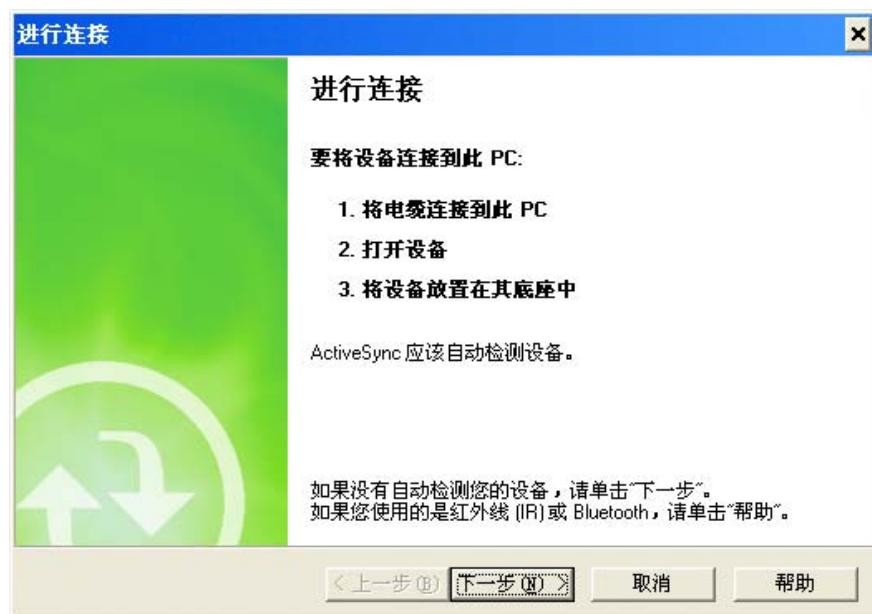
TO BE BEST

TO DO GREAT

广州友善之臂计算机科技有限公司



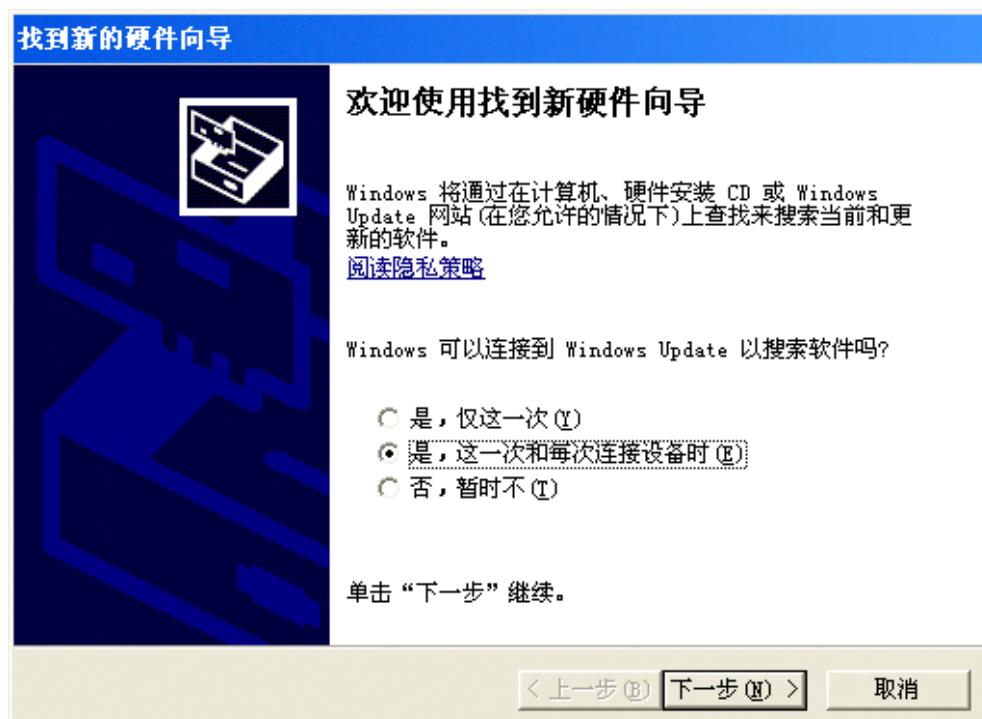
这时会自动运行 ActiveSync，点“取消”，同时在任务栏出现相应的图标托盘，出现如下界面



## 9.2.2 为同步通讯安装 USB 驱动

确认板子里面已经烧写好了我们提供的 WINCE 映象文件，并开机运行，系统起来以后，接上 USB 电缆，并与 PC 连接，如果以前没有安装过这个驱动，计算机会出现“发现新硬件”的提示，这时就可以按照本小节的步骤安装驱动了，我们用到的驱动程序的位置在光盘的“\Windows 平台工具\CE 用同步 USB 驱动”目录中：

Step1：找到新的硬件，请如图进行选择，点“下一步”继续。



Step2：选择“从列表或指定位置安装”，点“下一步”继续，如图。

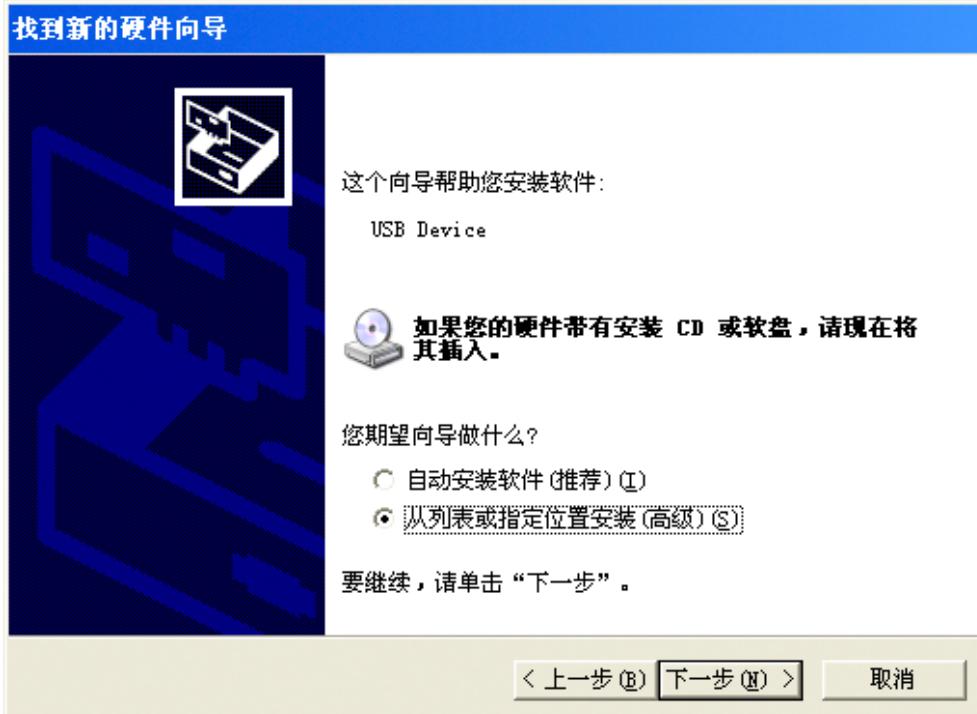


追求卓越 创造精品

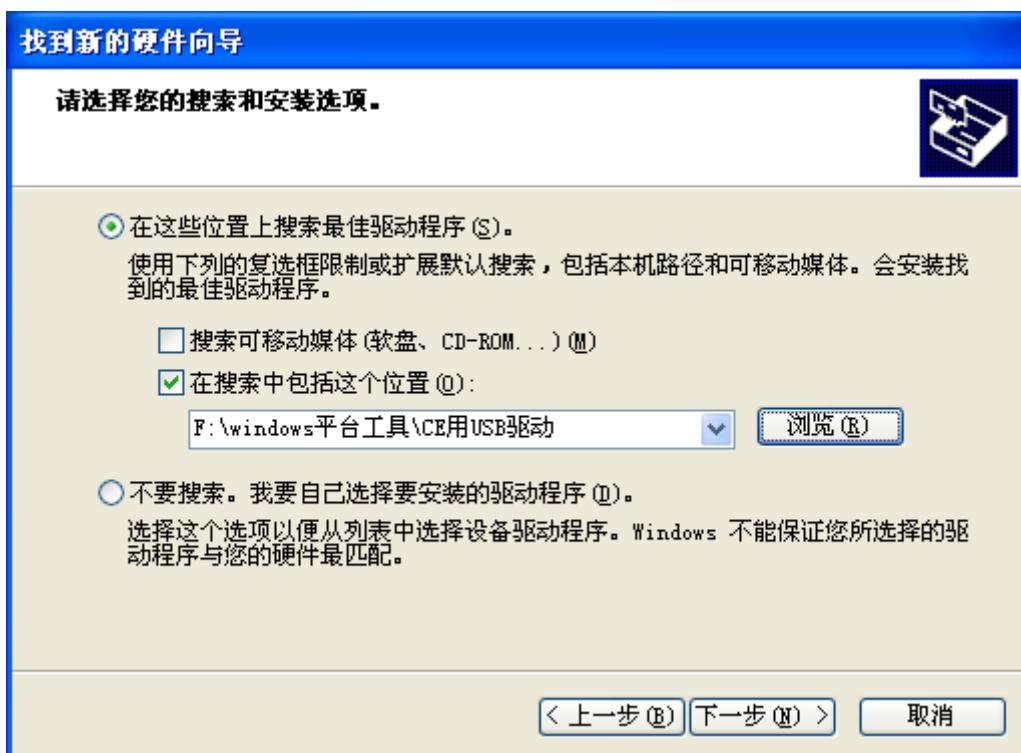
TO BE BEST

TO DO GREAT

广州友善之臂计算机科技有限公司



Step3：点“浏览”并定位到  
光盘\Windows 平台工具\CE 用同步 USB 驱动，如图





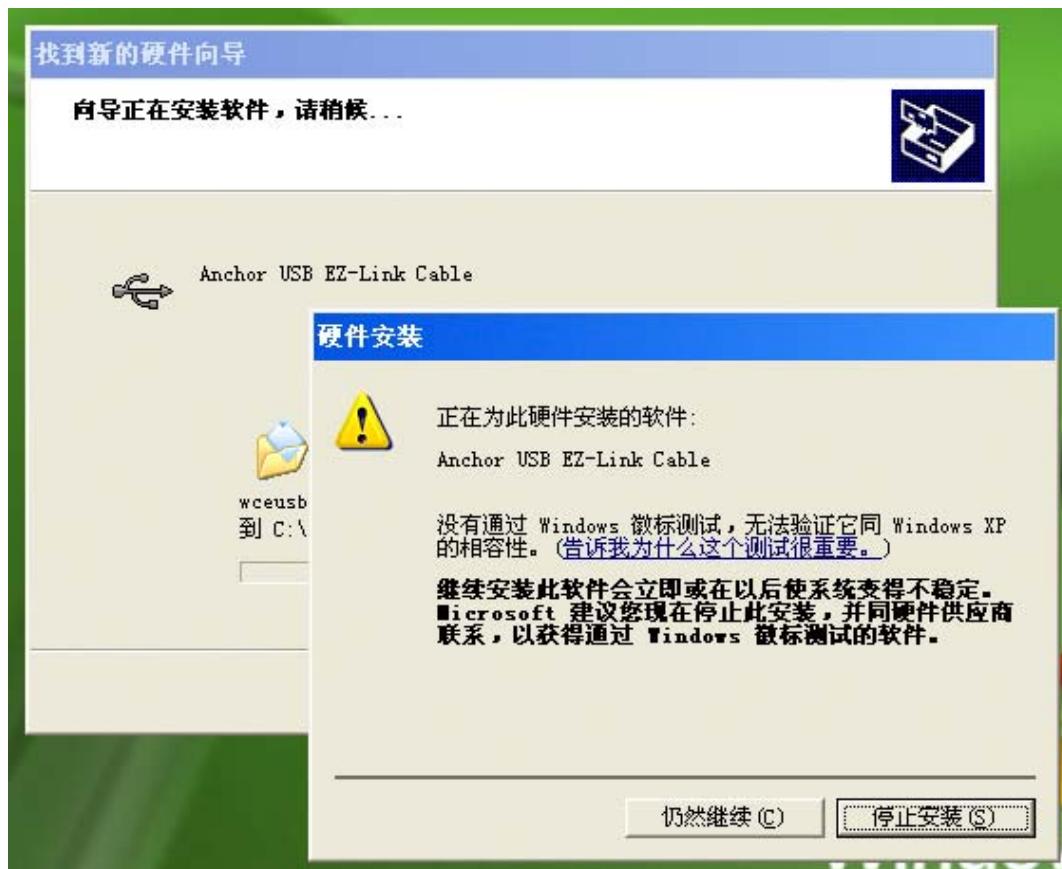
追求卓越 创造精品

TO BE BEST

TO DO GREAT

广州友善之臂计算机科技有限公司

Step4: 系统将会自动搜索一会，然后出现如下界面，点“仍然继续”，系统将自动安装完毕 USB 驱动。





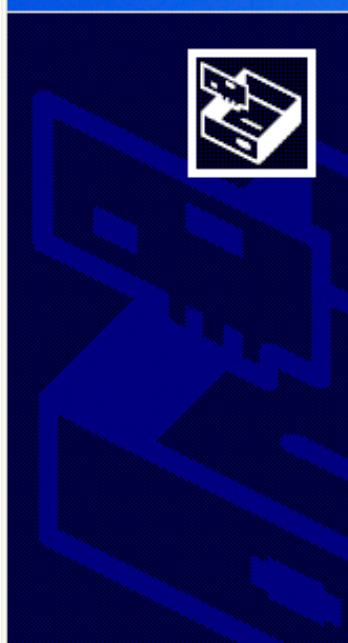
追求卓越 创造精品

TO BE BEST

TO DO GREAT

广州友善之臂计算机科技有限公司

找到新的硬件向导



完成找到新硬件向导

该向导已经完成了下列设备的软件安装：



Anchor USB EZ-Link Cable

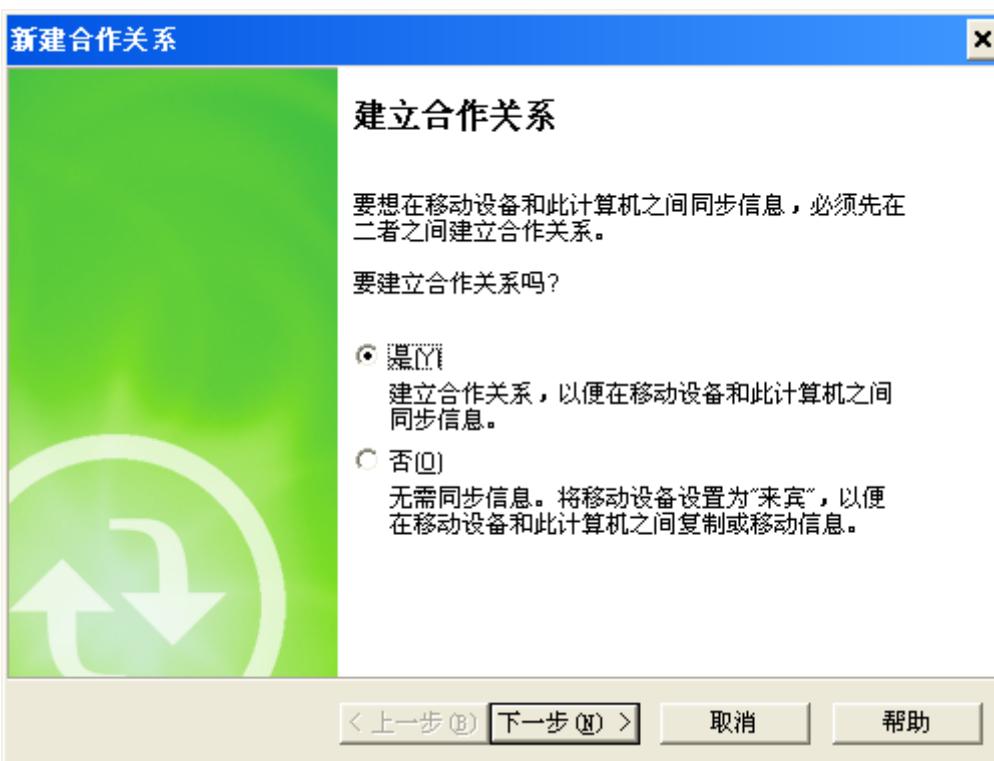
要关闭向导，请单击“完成”。

< 上一步 (B)

完成

取消

Step5：同时 ActiveSync 会自动跳出运行，如果您对使用 ActiveSync 还不熟悉，请点击“取消”。



### 9.2.3 使用 ActiveSync 同步传输工具复制文件

经过上一小节的 USB 驱动安装，实际上您已经看到开发板已经同 PC 连接好了，我们现在看看如何使用它。

#### (1) 查看 PC 端的连接情况

当我们看到如下窗口跳出时，我们也可以注意到 PC 任务栏的右下角的 ActiveSync 也变成了绿色，这说明一切准备就绪。



实际上，ActiveSync 安装完毕后，在“我的电脑”里会出现一个“移动设备”图标，现在我们双击打开它，您将看到目标板的所有目录，如下

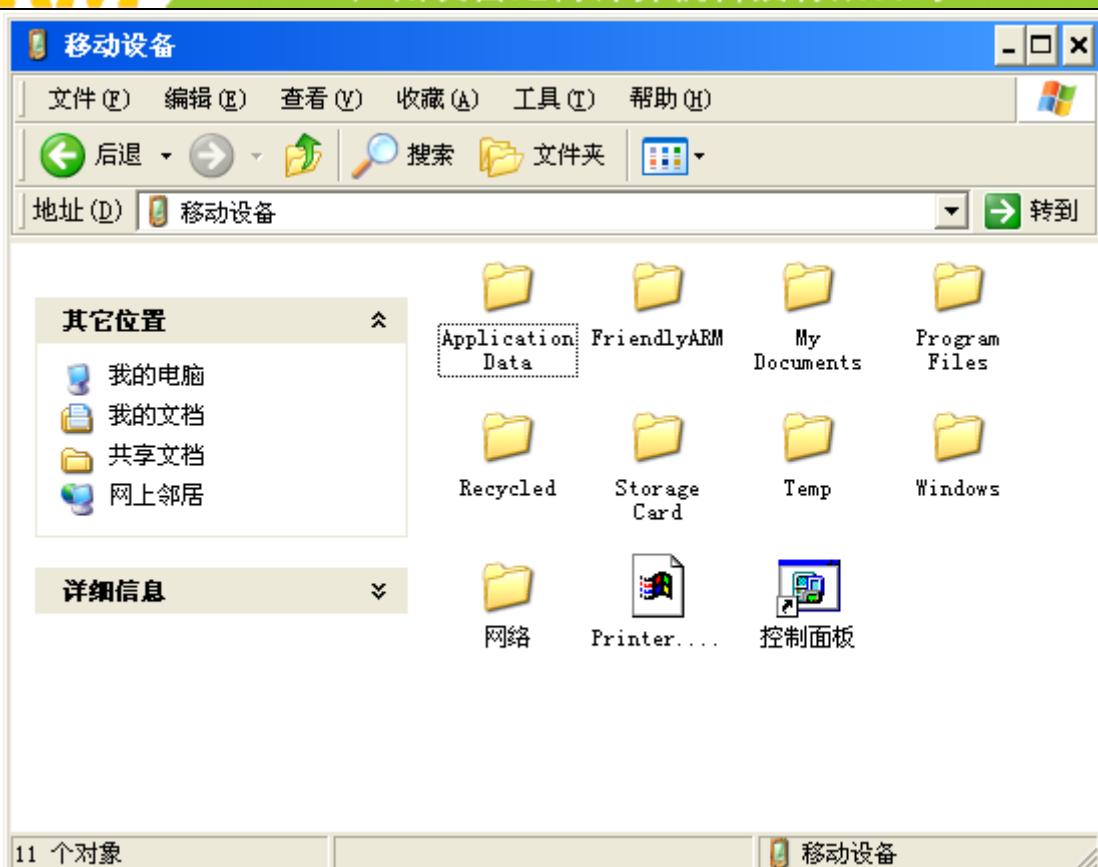


追求卓越 创造精品

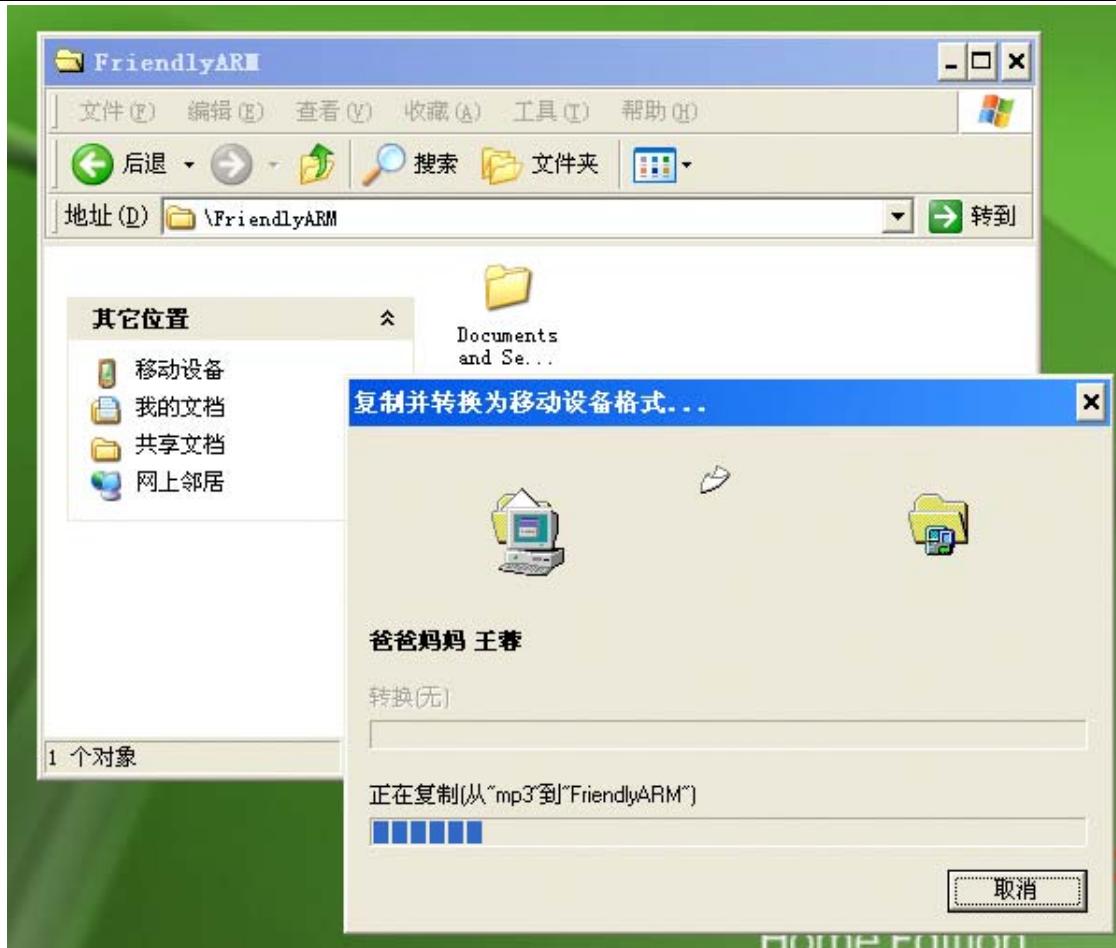
TO BE BEST

TO DO GREAT

广州友善之臂计算机科技有限公司



这时，双击打开 FriendlyARM 目录，在您的电脑里面找一首 mp3 文件或者其他小于 35M 的文件并拖动到打开的 FriendlyARM 文件夹，如图



这样，我们就能像操作 Windows 里面的目录一样，对开发板里面的目录进行文件复制等操作了。

## (2) 查看开发板一端的情况

接上一步，我们复制了一首 Mp3 文件到 FriendlyARM 目录，是不是真的有了呢？

接上 USB 鼠标或者使用触摸笔，点击打开 我的电脑->FriendlyARM 目录，可以看到如下界面。



追求卓越 创造精品

TO BE BEST

TO DO GREAT

广州友善之臂计算机科技有限公司



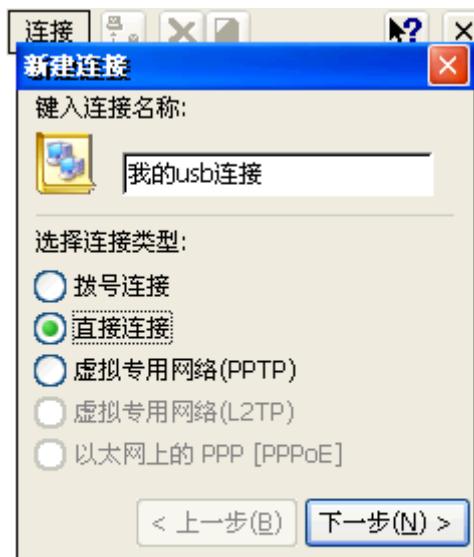
#### 9.2.4 使用 ActiveSync 与 Platform Builder 连接实现通讯并屏幕截图

首先确认开发板已经和 PC 之间可以使用 ActiveSync 连接成功，同时网络连接也没有问题，如下图



我们先设置开发板上的 WINCE 连接。

打开 开始->设置->网络和拨号连接，点“新建连接”设置对话框中，选择连接类型为“直接连接”，如下图



改连接名称为“我的 usb 连接”或者不改也可以，点“下一步”，在出现的“选择设备”下拉列表中选择“S3C2440 USB Cable:”，如下图



点右上角的“OK”，这时出现“我的 usb 连接”图标，如下图



追求卓越 创造精品

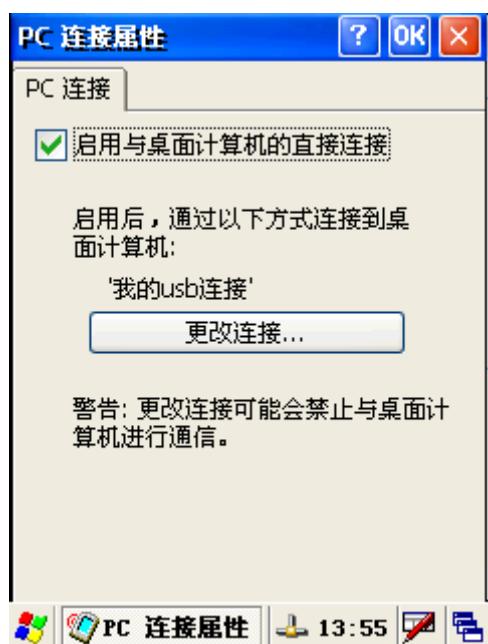
TO BE BEST

TO DO GREAT

广州友善之臂计算机科技有限公司



打开“开始->设置->控制面板”，双击打开“PC 连接”图标，进入“PC 连接属性”设置对话框，选中“启用与桌面计算机的直接连接”的复选框，然后再点“更改连接”按钮，如下图



在“更改连接”设置对话框的下拉列表中选择刚刚新建的连接“我的 usb 连接”，然后点“OK”退出设置，这样就完成了开发板这端的设置。

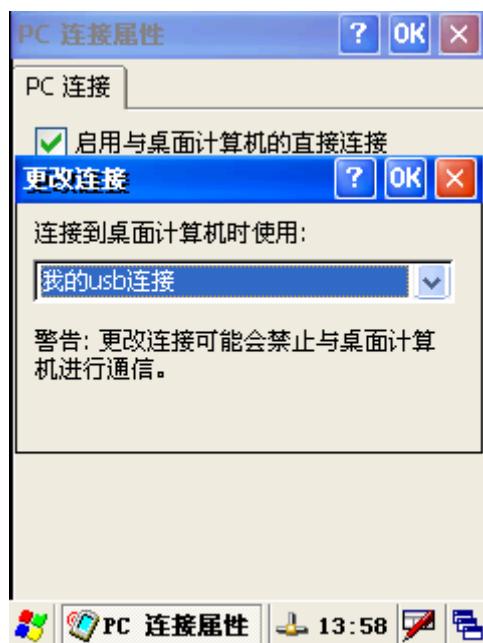


追求卓越 创造精品

TO BE BEST

TO DO GREAT

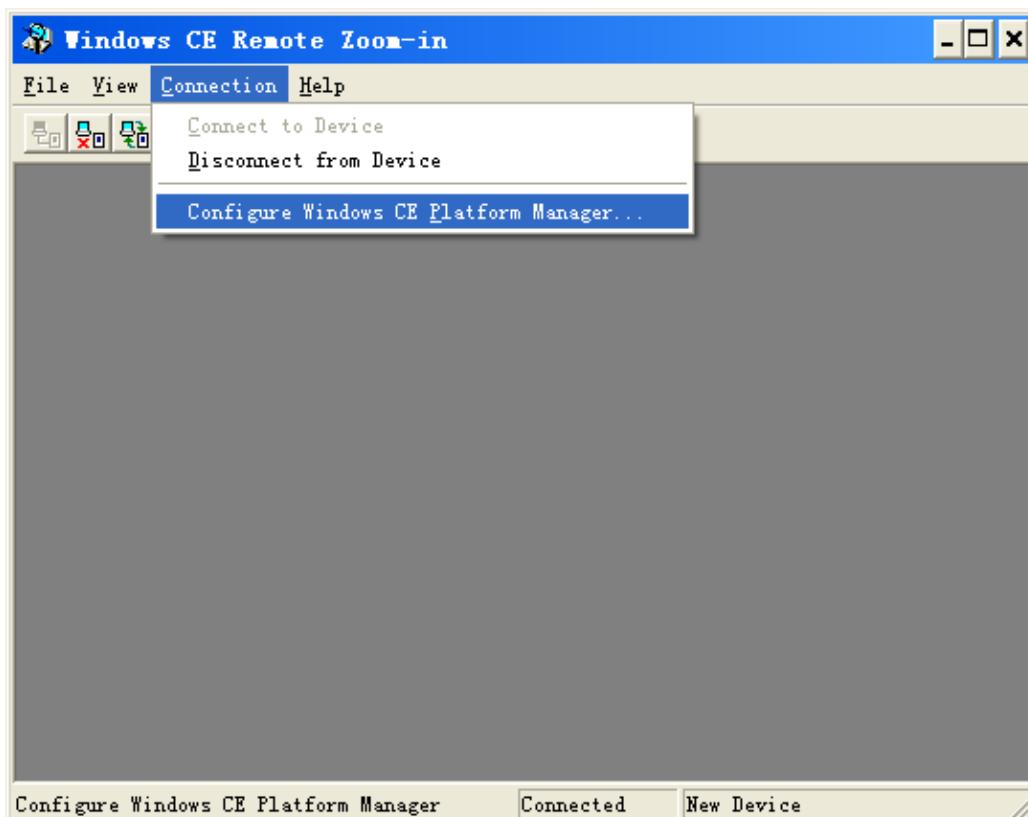
广州友善之臂计算机科技有限公司



现在，我们再来设置 PC 端，请务必确认 ActiveSync 已经和开发板连接成功。

点 PB 菜单 Tools -> Remote zoom-in，打开远程图片缩放工具，这个程序可以实现对远程移动设备显示屏幕的截屏。

Remote zoom-in 窗口打开之后，先要配置一下平台管理器，点击菜单 Connection->Configure windows ce platform manager，如下图





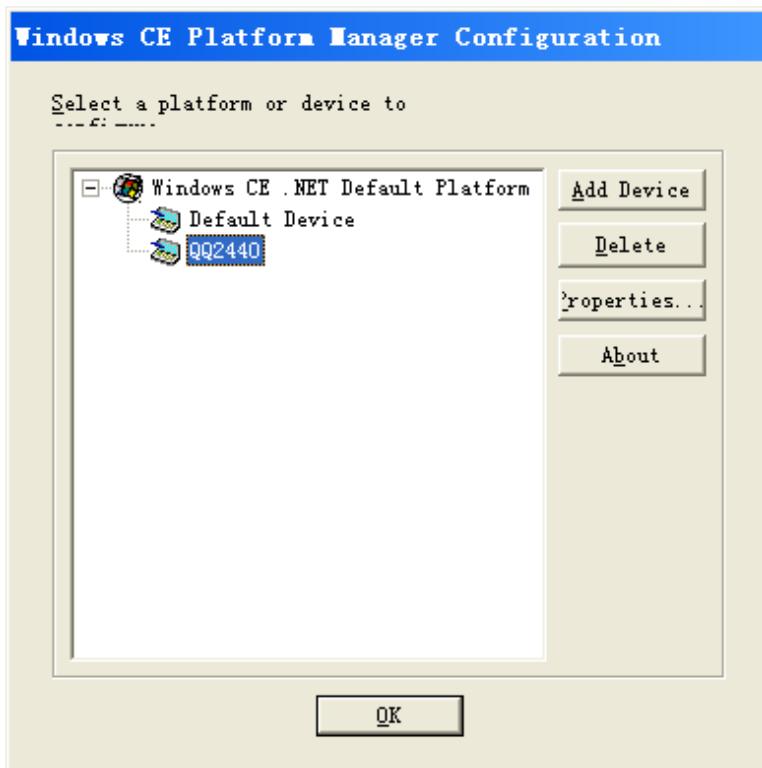
追求卓越 创造精品

TO BE BEST

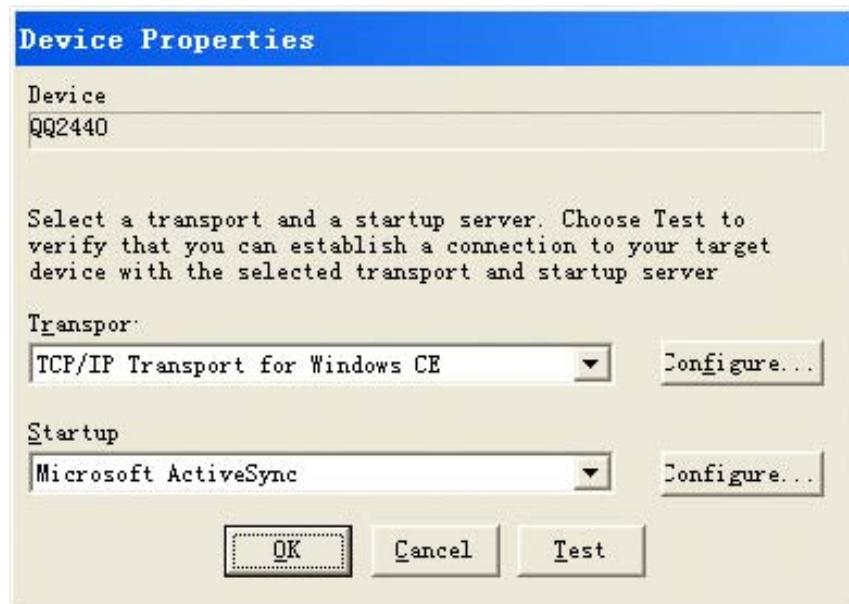
TO DO GREAT

广州友善之臂计算机科技有限公司

在弹出的对话框中添加一个新设备，并且取名为“QQ2440”(本文档基于 QQ2440 开发板而来，因此这里暂时沿用 QQ2440 的名称)，如下图



点右边的“Properties...”按钮，设置“MINI2440”设备平台的属性，如下图



点“Transport”下拉框右边的“Configure”按钮，开始设置 TCP/IP 传输，在 HOST IP 一项中输入您的主机 IP 地址，请保证开发板 WINCE 的 IP 地址(默认为 192.168.1.216)与 PC



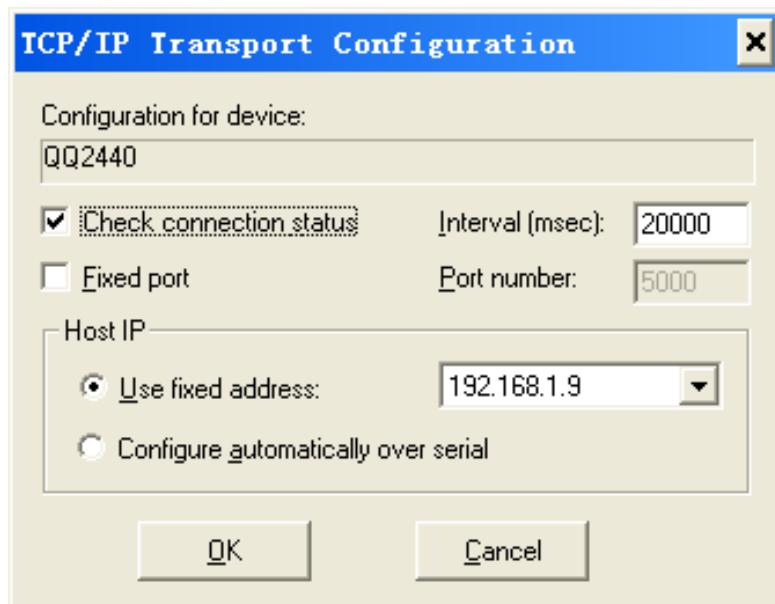
追求卓越 创造精品

TO BE BEST

TO DO GREAT

广州友善之臂计算机科技有限公司

主机的 IP 地址在同一个网段, 如图



设置结束, 点 OK 按钮返回到 Remote Zoom-in 主窗口, 点菜单 Connection->Connect to Device, 开始连接开发板。



这时一般会出现如下图所示界面, 显示正在连接。

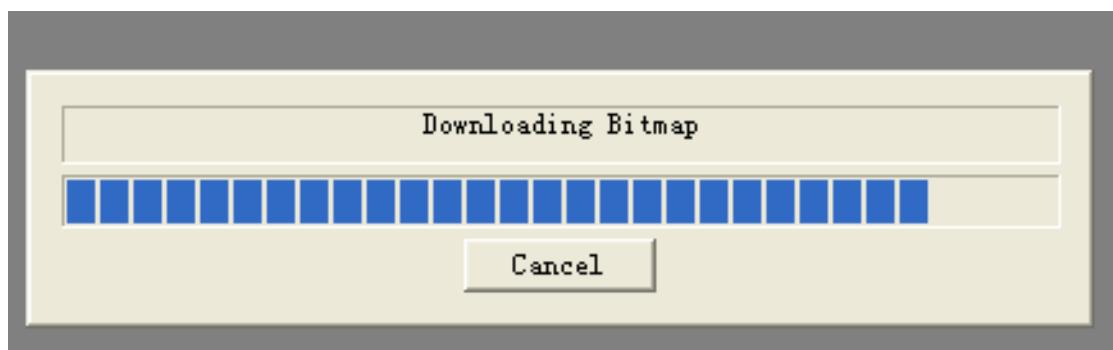


追求卓越 创造精品

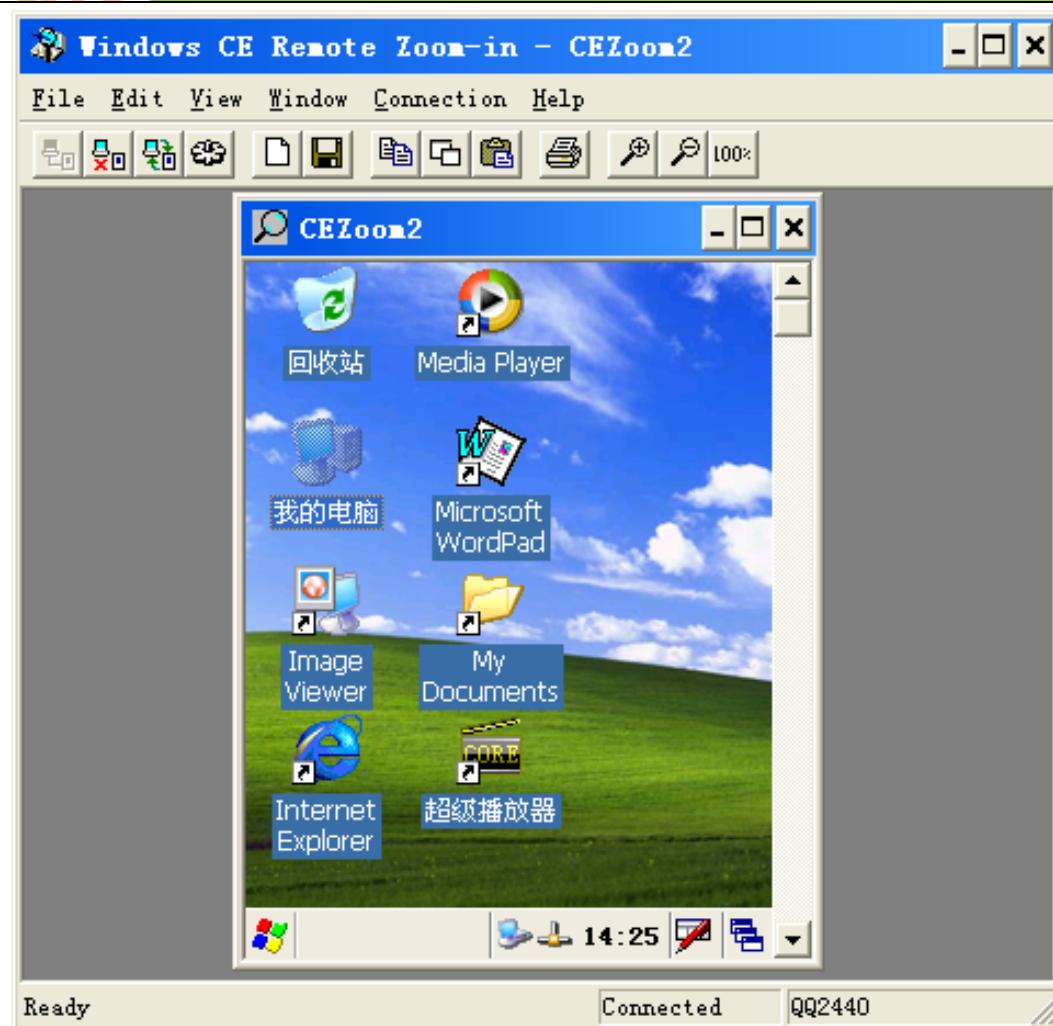
TO BE BEST

TO DO GREAT

广州友善之臂计算机科技有限公司



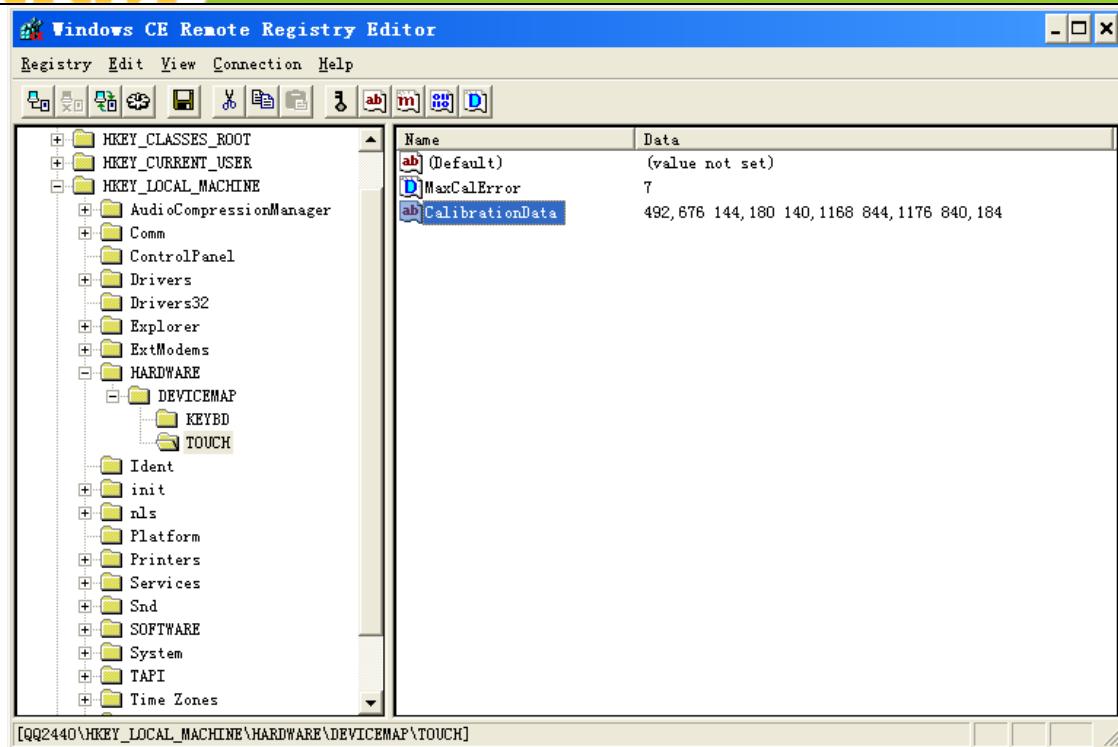
当连接成功，会出现和目标板当前屏幕一样的几个界面窗口，如下图



这时点 File-Save as 可以保存当然的屏幕截图。

### 9.2.5 使用 ActiveSync 与 Platform Builder 在线编辑注册表

当您学会使用上一小节的内容实现截屏之后，您还可以在 PB 的 Tools 菜单中点击“Remote Registry Editor”来运行远程注册表编辑工具查看并修改 wince 的注册表内容，如下图，在此就不作更详细的解释了。



### 9.3 创建 EVC 的 Hello,World，并编译下载到开发板运行

我们将不需要编写任何代码，使用向导创建一个最简单的 EVC 应用程序，并通过 USB 同步连接的方式下载到开发板运行起来，这是我们的第一个基于 EVC 的 Hello, World 程序，在这里我们主要向您展示了最基本的开发步骤。

**说明：使用基于 QQ2440\_SDK 的 EVC 不需要安装庞大的 PB5 平台即可进行应用开发。**

(1) 打开运行 EVC 集成开发环境，点 File → New... 如图

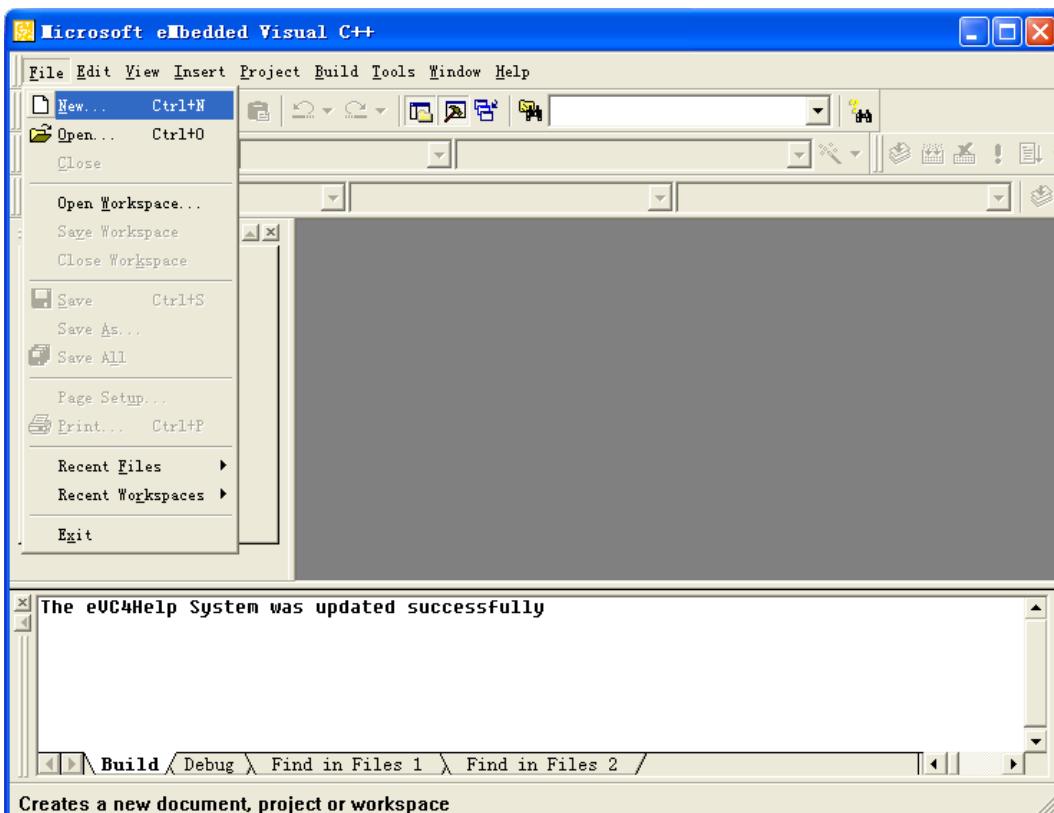


追求卓越 创造精品

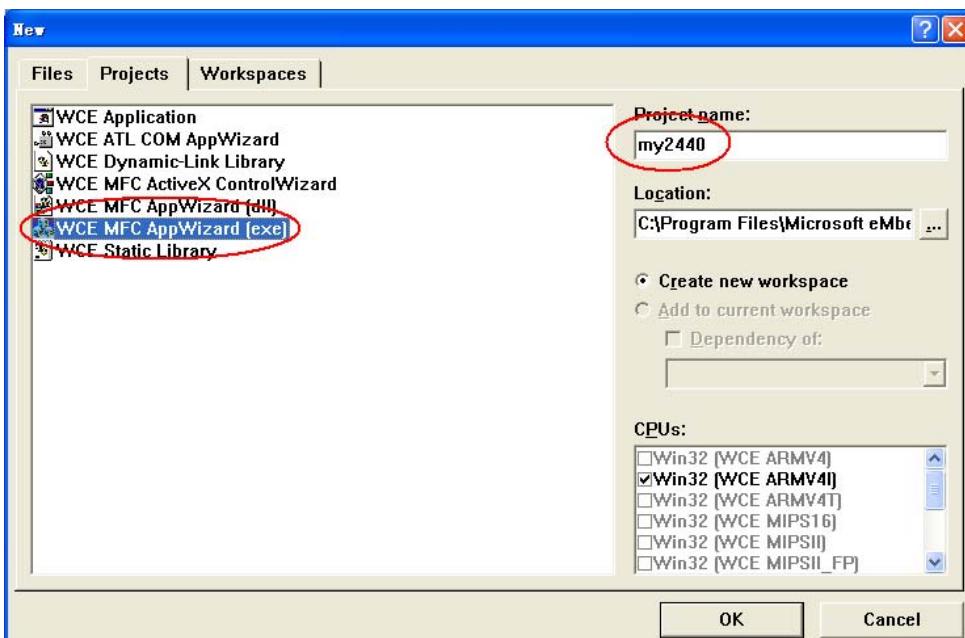
TO BE BEST

TO DO GREAT

广州友善之臂计算机科技有限公司



(2)出现程序向导窗口，选择 WCE MFC AppWizard(exe)，在 Project Name 中并输入项目名字，在 Location 中选择存放位置，如图，点“OK”继续



(3)出现如图窗口，选择语言设置为英语，其他为默认，点“Next”继续

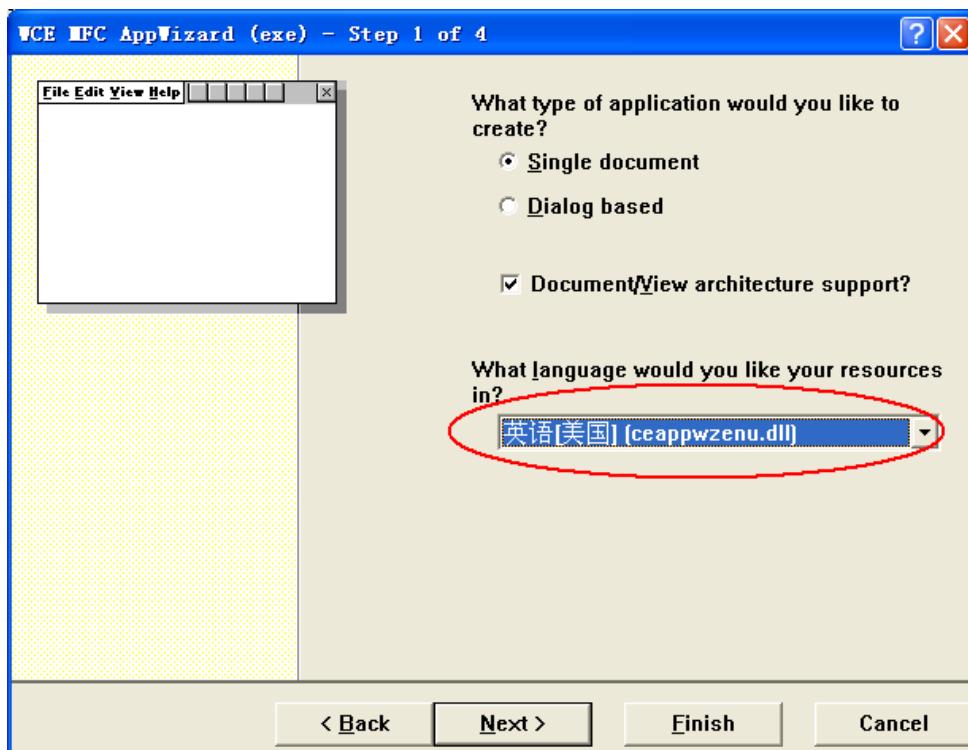


追求卓越 创造精品

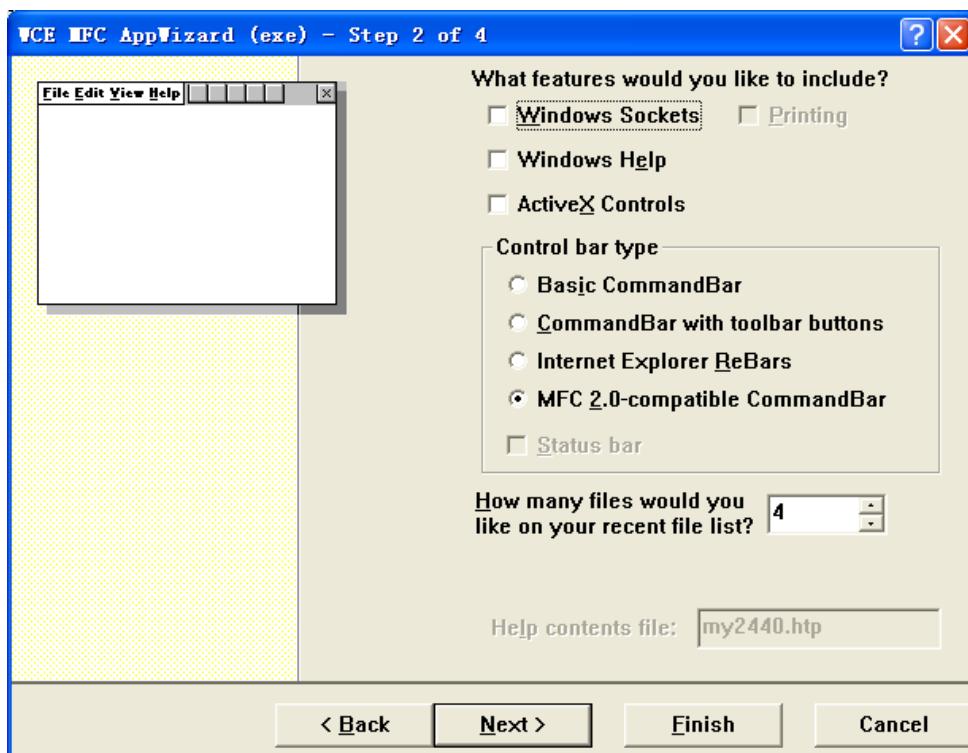
TO BE BEST

TO DO GREAT

广州友善之臂计算机科技有限公司



(4)出现如图窗口，所有设置保持默认，点“Next”继续



(5)出现如图界面，保持默认设置，点“Next”继续

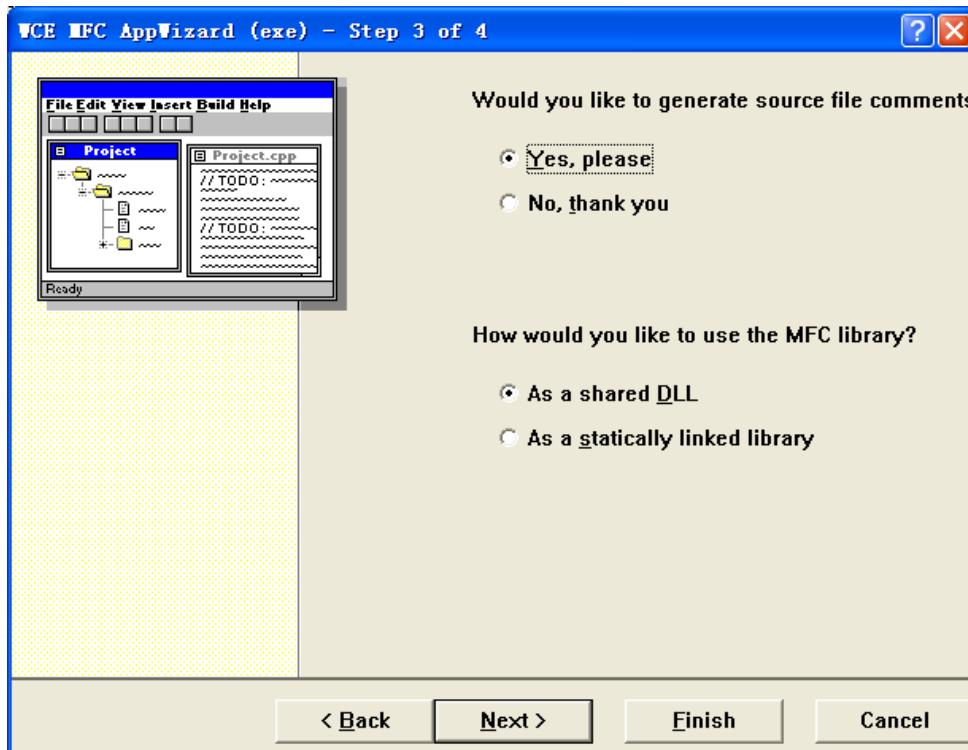


追求卓越 创造精品

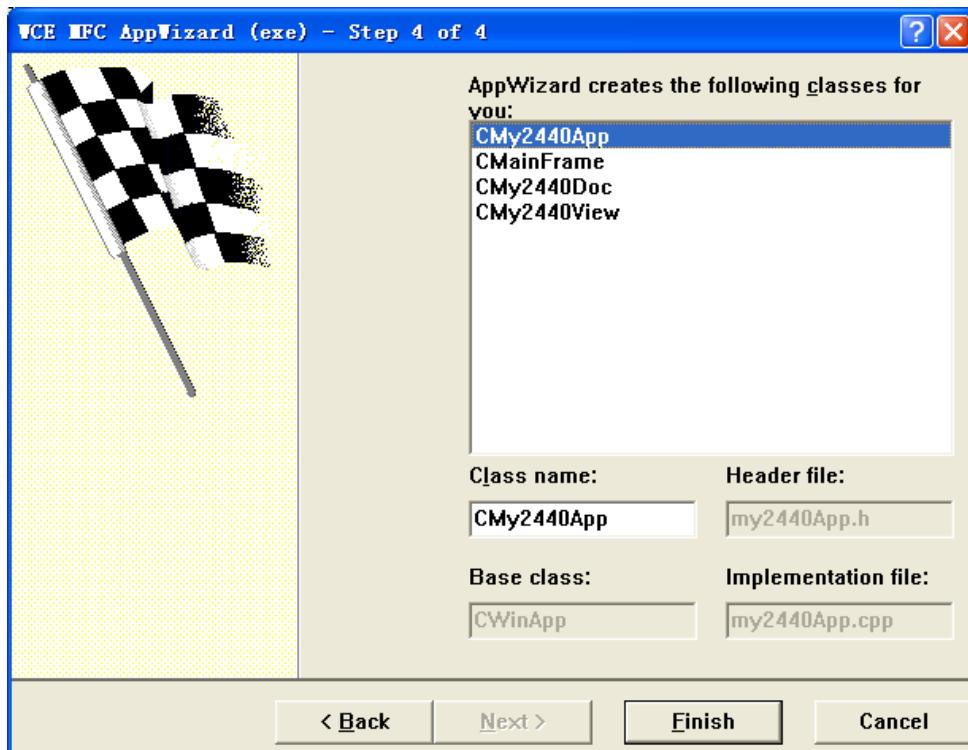
TO BE BEST

TO DO GREAT

广州友善之臂计算机科技有限公司



(6)出现如图界面，点“Finish”继续



(7)出现 New Project Information 窗口，点“OK”完成向导

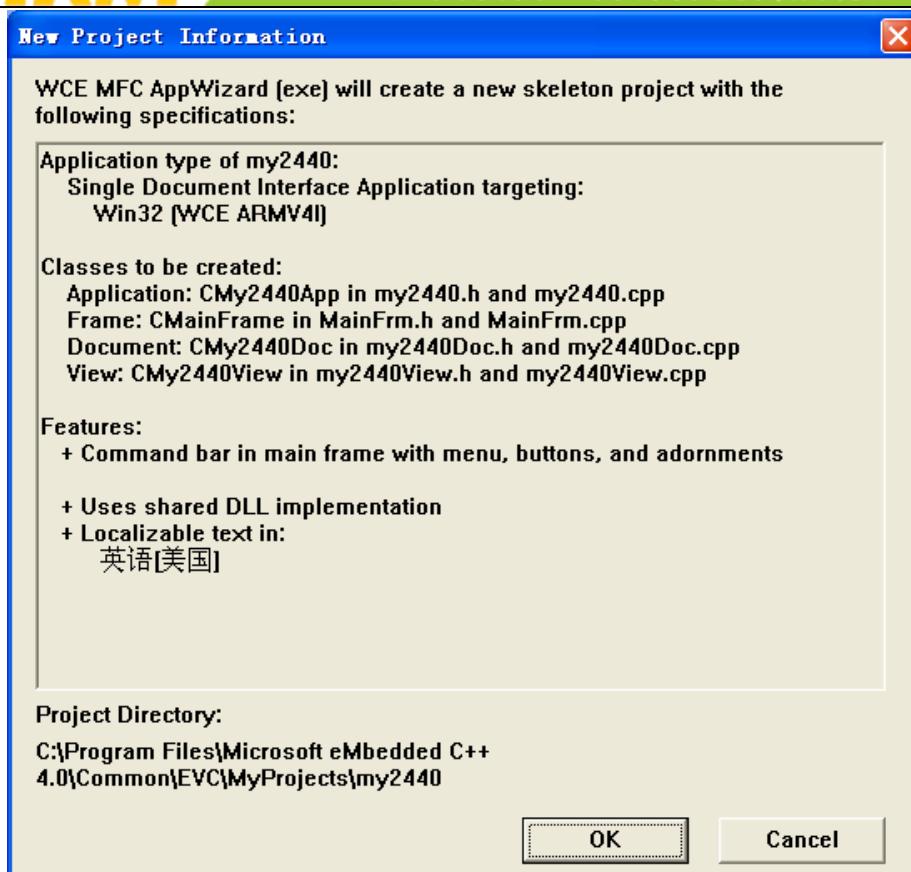


追求卓越 创造精品

TO BE BEST

TO DO GREAT

广州友善之臂计算机科技有限公司



(8)回到 EVC 主界面，点 Build → Set Active Platform...以设置目标板连接

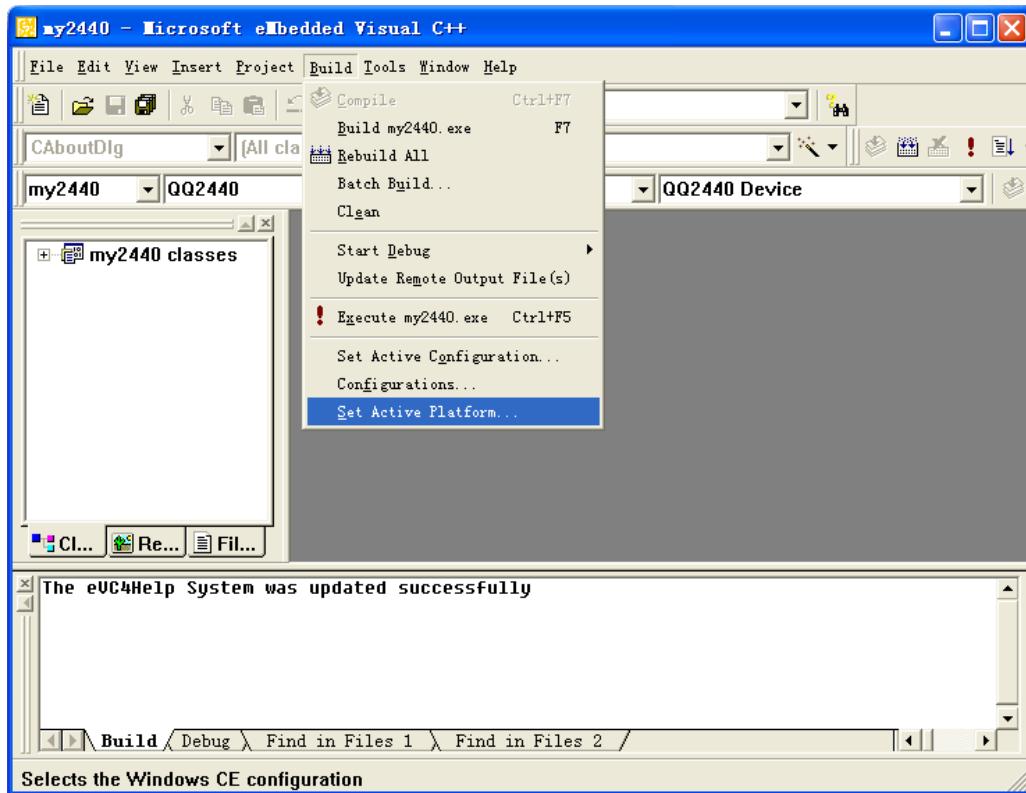


追求卓越 创造精品

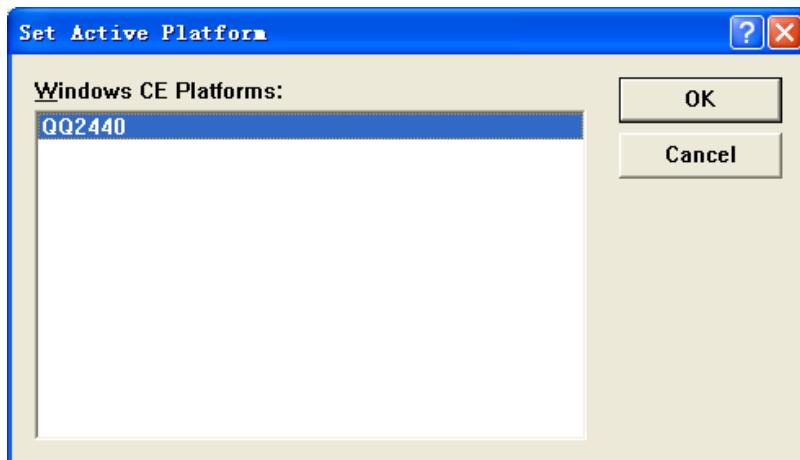
TO BE BEST

TO DO GREAT

广州友善之臂计算机科技有限公司



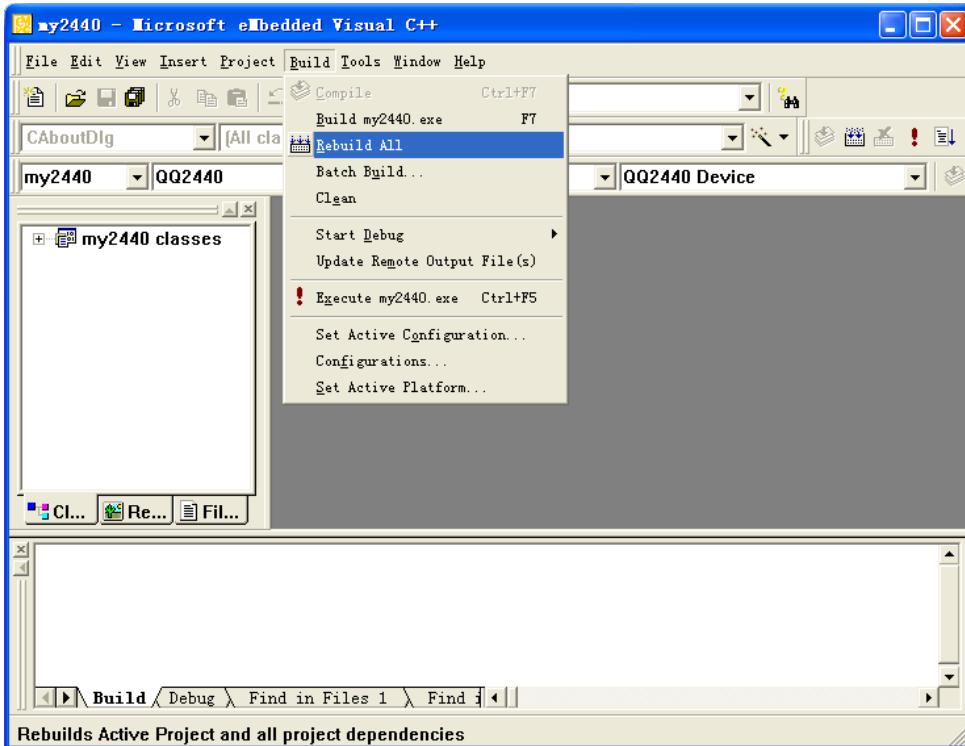
(9)因为我们只安装了导出的 QQ2440\_SDK，因此在这里只有一项选择，如图选择，点“OK”返回即可



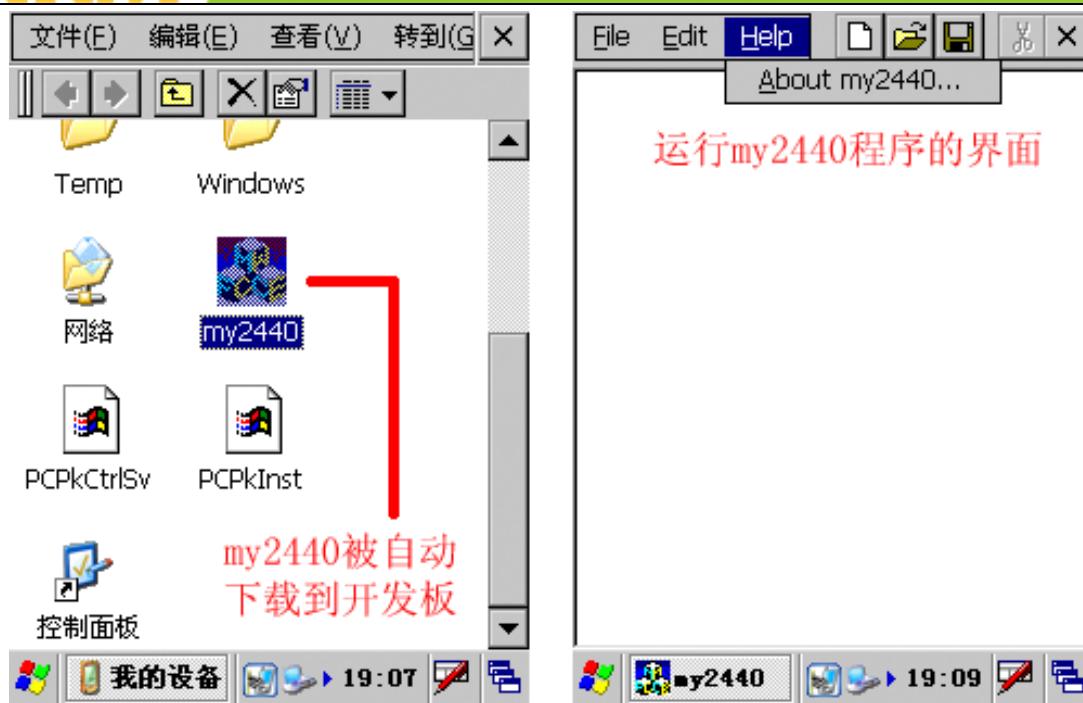
(10)确认 PC 同步已经建立并连接正常(同时必须连接好网线)，如图



(11)在 EVC 主界面，点 Build → Rebuild All，如图，系统开始编译你所创建的工程文件：

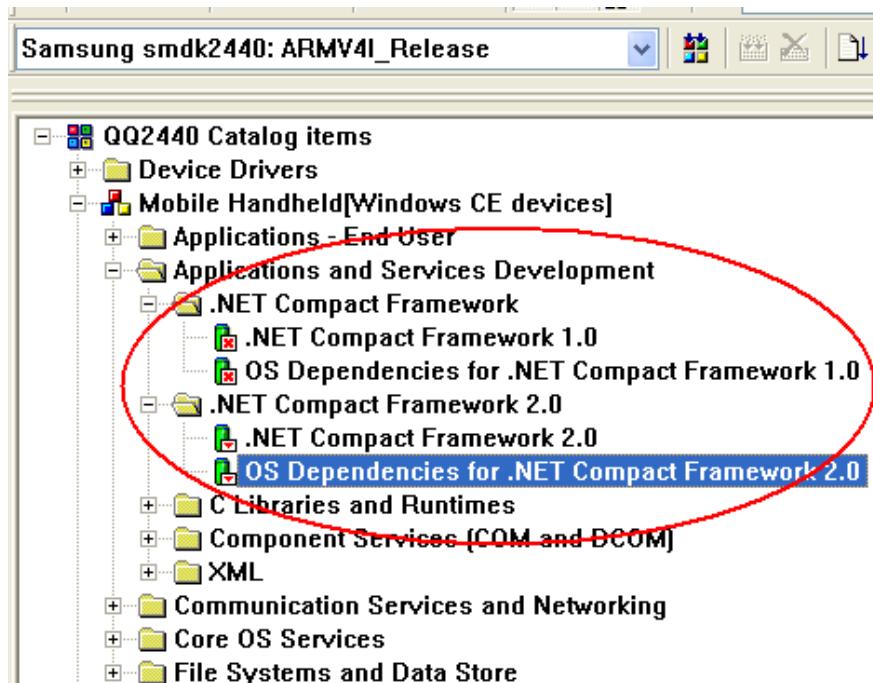


(12)编译完毕，系统会自动下载编译好的可执行文件到开发板的根目录(打开“我的电脑”就可以看到)，双击并运行它，如图：



## 9.4 创建 VS2005/2008 应用程序，并编译下载到开发板运行

**注意：必须要安装 PB5 的补丁程序，才能在定制内核时出现.NET 2.0 选项，并使选择使用它才能支持 VS2005/2008 应用程序。(缺省的内核稍写文件和示例工程已经添加了该选项)如图：**





追求卓越 创造精品

TO BE BEST

TO DO GREAT

广州友善之臂计算机科技有限公司

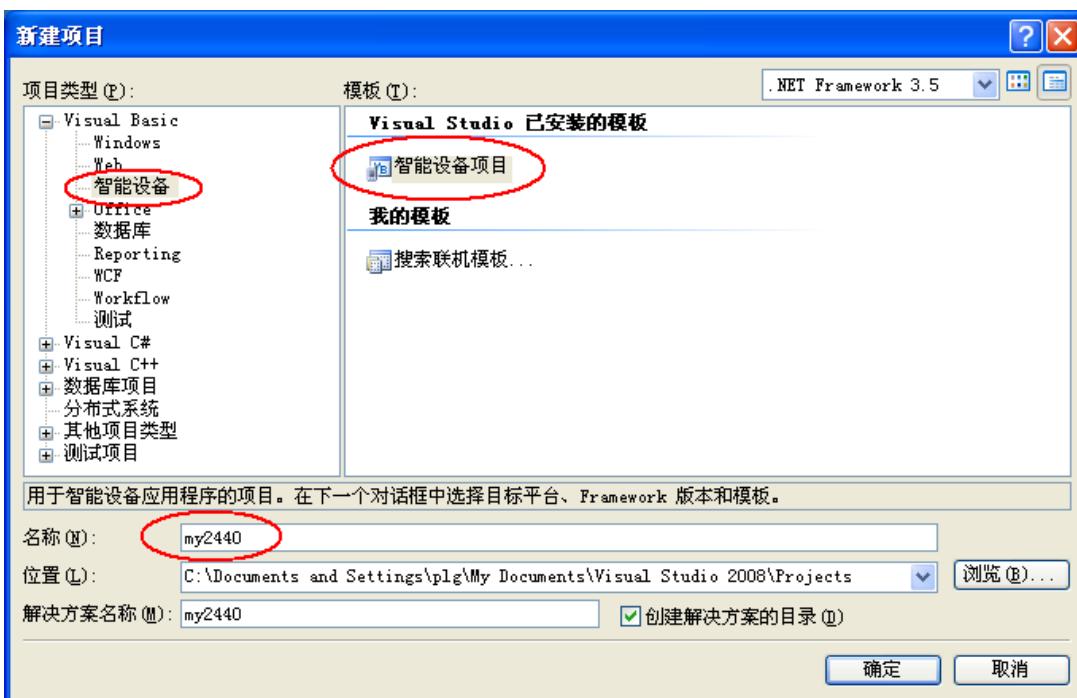
下面是使用 VS2008 的基本开发步骤：

#### 9.4.1 创建项目

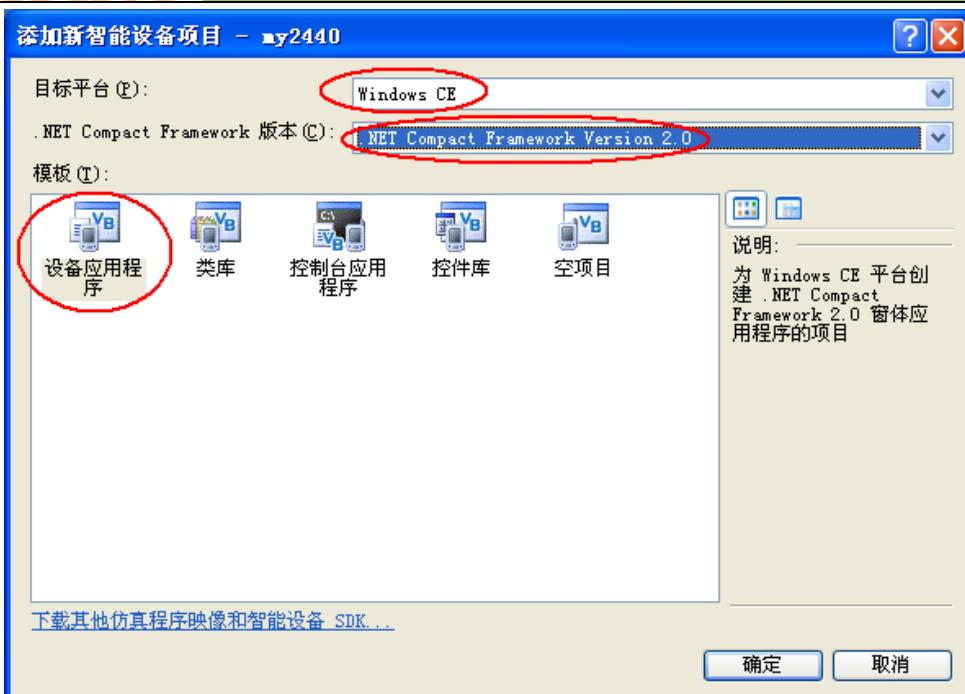
(1) 打开运行 VS2008，点菜单文件→新建→项目，如图：



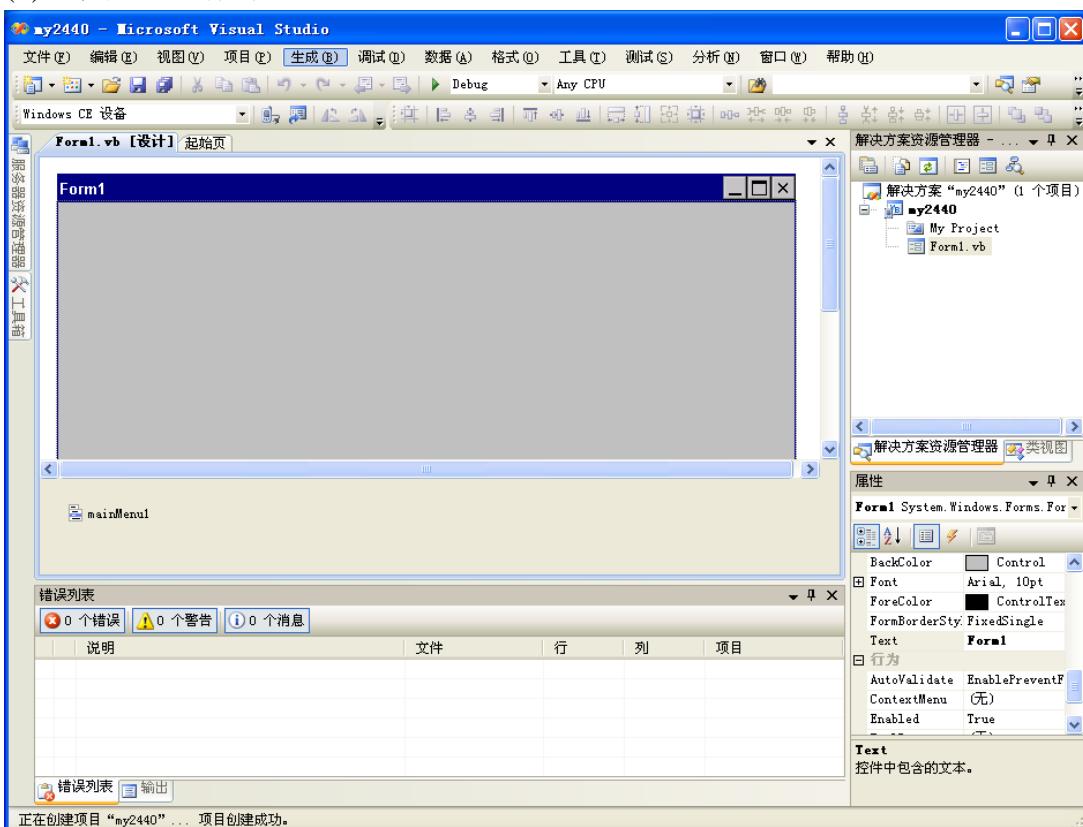
(2) 出现“新建项目”窗口，在“项目类型”一栏中点击选择“Visual Basic → 智能设备”，在“模板”一栏中选择“智能设备项目”，在“名称”一栏中输入“my2440”，其他采用缺省设置，点“确定”继续：



(3) 出现“添加新只能设备项目”窗口，在“目标平台”下拉列表中选择“Windows CE”，在“.NET Compact Framework 版本”下拉列表中选择“.NET Compact Framework Version 2.0”，在“模板”中选择“设备应用程序”，点“确定”继续：



(4) 出现如图工作界面



这时，可以调整一下“Form”的大小，通过“工具箱”添加一些控件到“Form”中，如图：

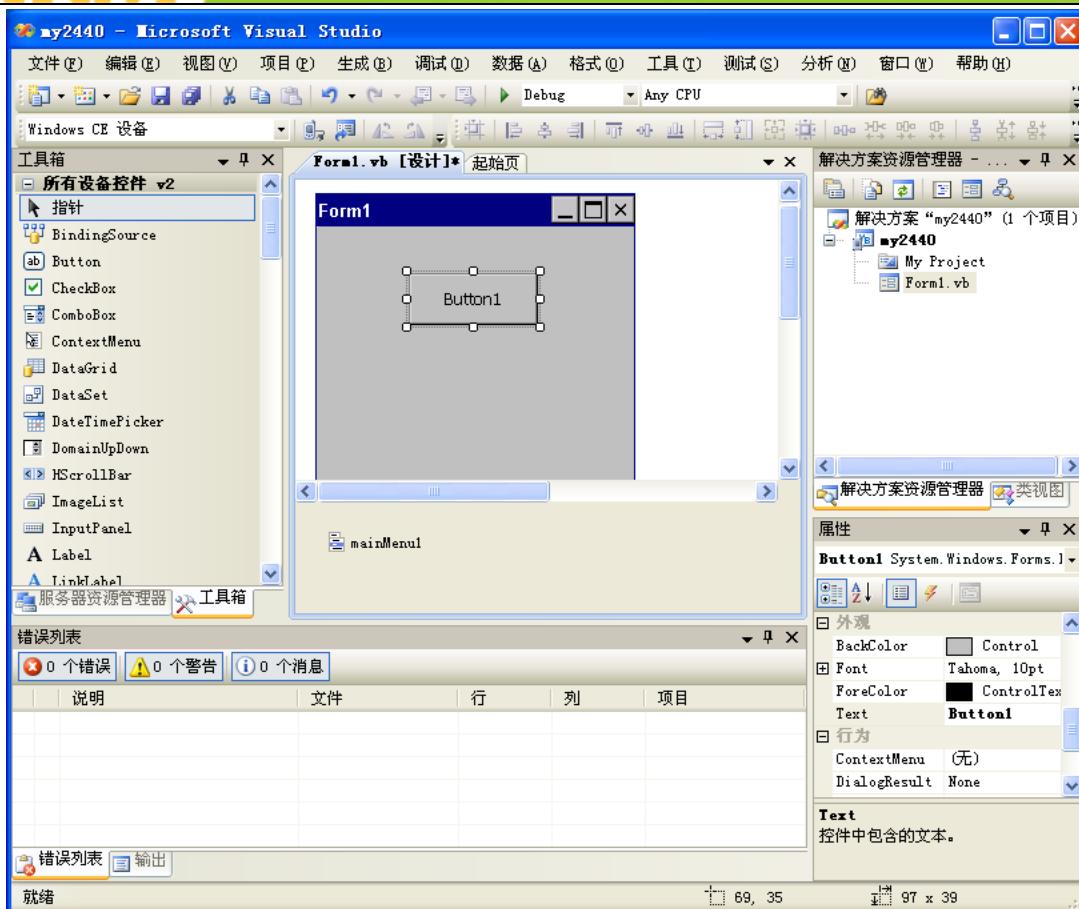


追求卓越 创造精品

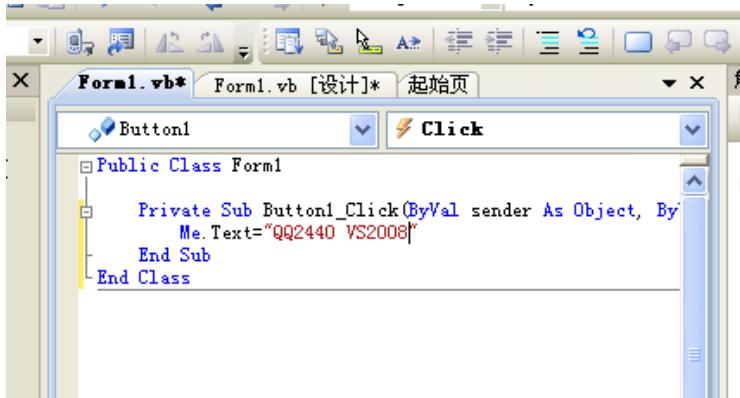
TO BE BEST

TO DO GREAT

广州友善之臂计算机科技有限公司



(5) 双击控件“Button1”出现代码设计窗口，输入以下简单代码，它将实现：当点击 Button1 按钮时，窗口标题会显示代码中设置的内容，如图：



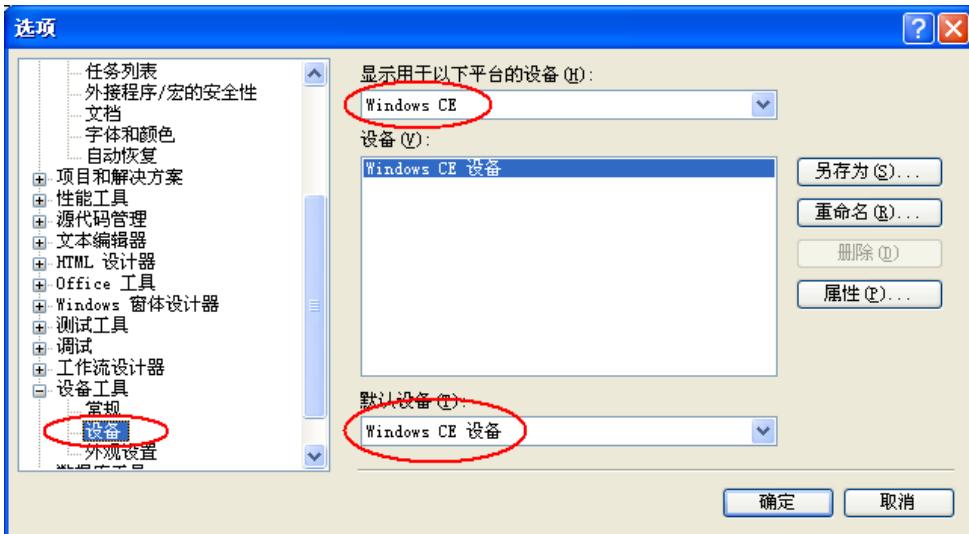
保存好以上创建的项目文件。

#### 9.4.2 设置连接开发板

(1) 确认 PC 同步已经建立并连接正常(可以不必连接网线)，如图



(2) 点 VS2008 菜单“工具”→“选项”，出现“选项”窗口，在左侧一栏中选择“设备工具”→“设备”，在右侧中的各个下拉列表中作如图选择：



点“属性”按钮，出现“Windows CE 设备 属性窗口”，选择如图：

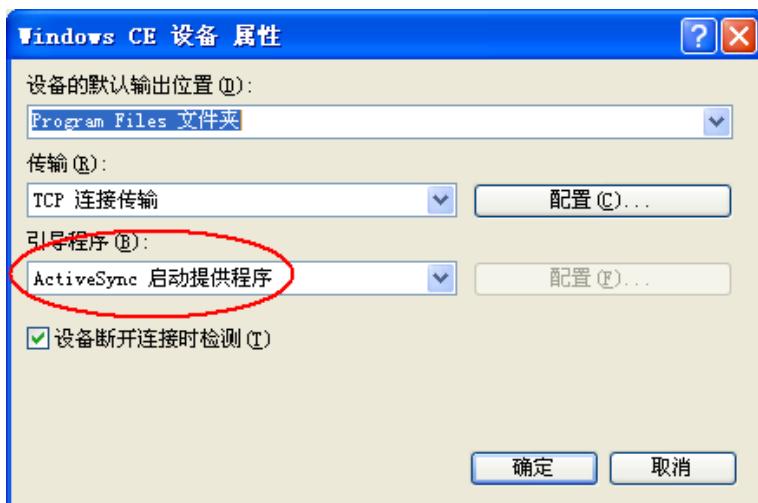


追求卓越 创造精品

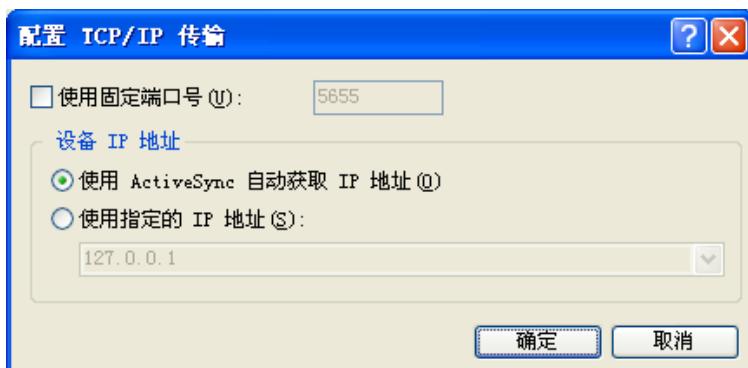
TO BE BEST

TO DO GREAT

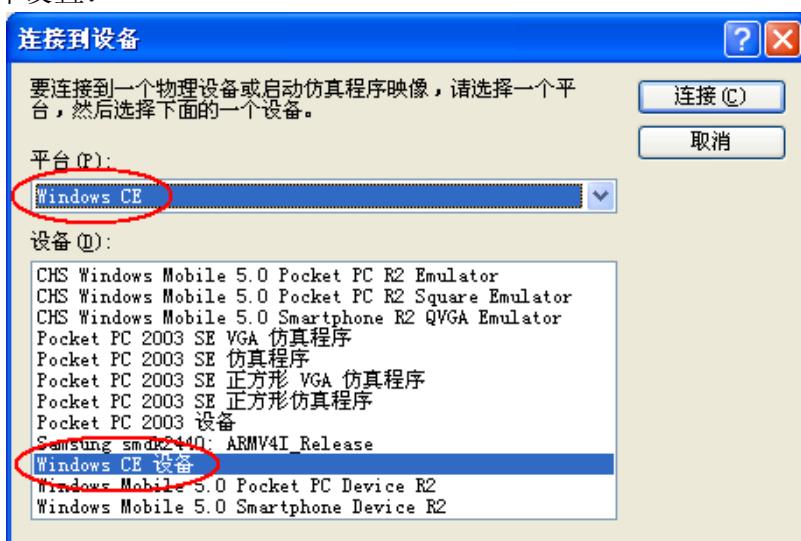
广州友善之臂计算机科技有限公司



点“配置”按钮，出现“配置 TCP/IP 传输”窗口，如图选择，点“确定”返回 VS2008 工作界面。



(3) 点 VS2008 菜单“工具”→“连接到设备”，出现“连接到设备”设置窗口，如图进行选择设置：





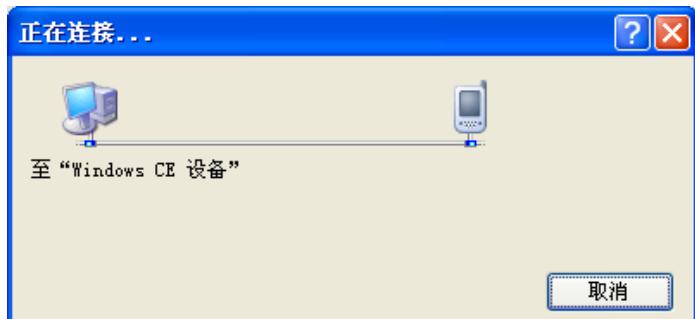
追求卓越 创造精品

TO BE BEST

TO DO GREAT

广州友善之臂计算机科技有限公司

点“连接”按钮，此时 VS2008 开始和开发板进行连接握手：



稍等一会，出现连接成功的提示，点“关闭”按钮返回 VS2008 工作主界面：

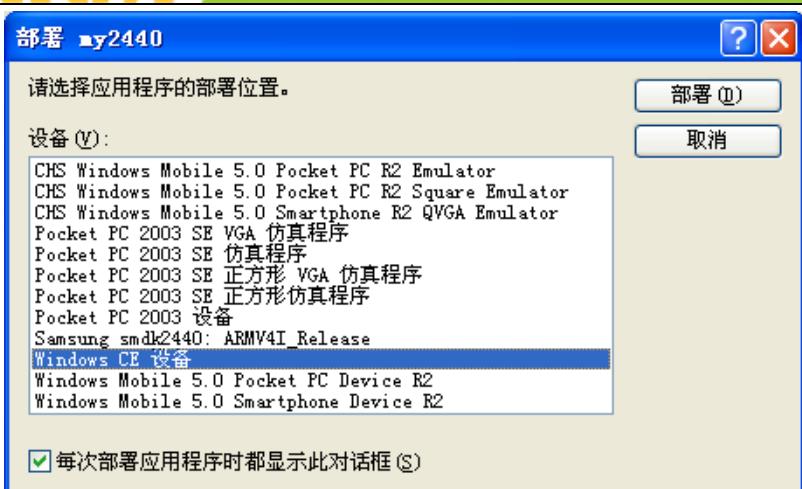


#### 9.4.3 编译下载程序到开发板运行

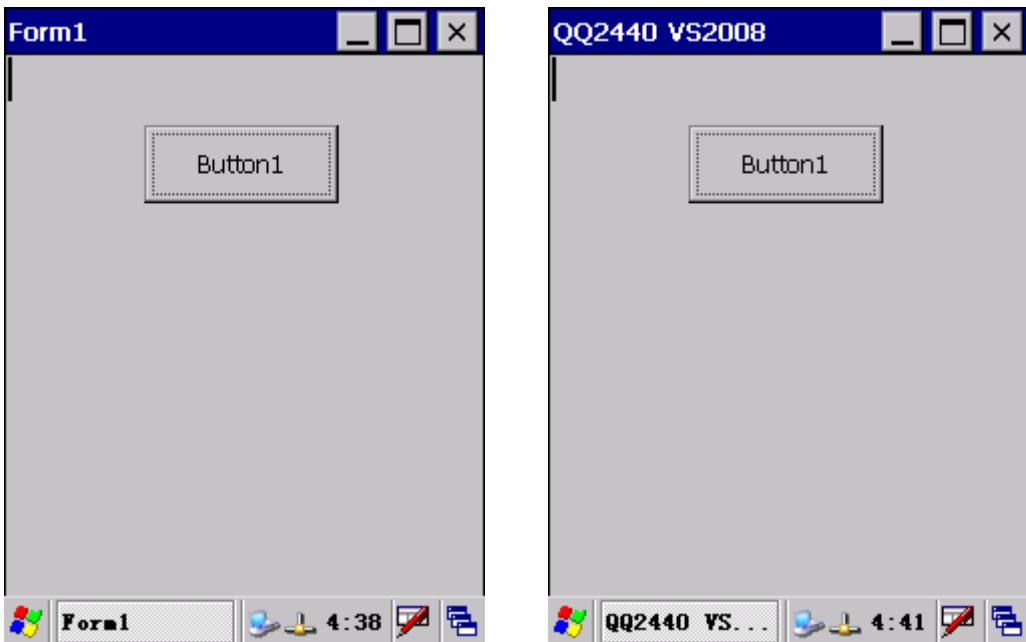
(1) 按上面的步骤，点菜单“调试”→“启动调试”或者直接按 F5 键开始调试过程。



(2) 出现“部署 my2440”窗口，选择“Windows CE 设备”，并点“部署”按钮开始部署，如图



(3)如果程序没有问题，等待一会它会直接下载到开发板并运行，点击一下“Button1”按钮，窗口标题改变如图，这正如我们在代码中的设置，如图：



通过 VS2008，你不仅可以编写 Visual Basic 程序，还可以使用 Visual C# 和 Visual C++ 进行程序设计，通过同步连接开发板，你还可以实现单步调试运行。

## 9.5 LED 驱动程序编写及测试示例

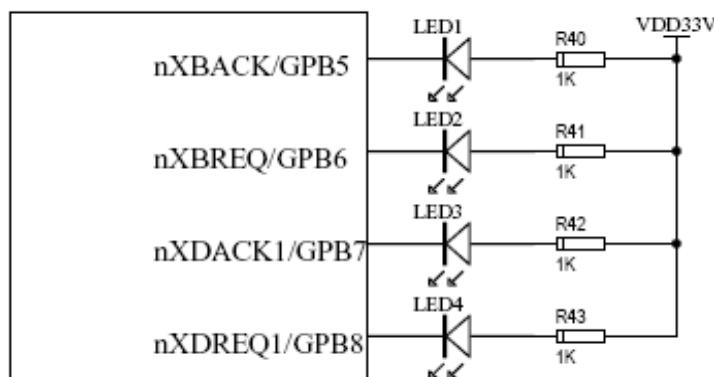
LED 驱动程序在光盘中的位置：\WindowsCE5.0\smdk2440\DRIVERS\LEDdriver  
LED 驱动程序文件的名称：LEDDriver.cpp

本小节以一个简单的 LED 驱动程序作为例子，介绍了 Windows CE 下常规流设备驱动

程序编写的一般步骤和方法，用户可以通过更改其中的代码实现类似的其他 IO 控制功能。

### 9.5.1 了解硬件连接

mini2440 的 4 个 LED 引脚连接如图所示：



可见它们使用了 CPU 的 GPB 端口，关于 GPB 端口的描述及其使用在 S3C2440 数据手册中有如下描述：

GPB 端口描述(见 S3C2440.pdf 第 276 页):

GPB8	Input/output	nXDREQ1
GPB7	Input/output	nXDACK1
GPB6	Input/output	nXBREQ
GPB5	Input/output	nXBACK

GPBCON 端口描述(见 S3C2440.pdf 第 284 页):

GPB8	[17:16]	00 = Input 10 = nXDREQ1	01 = Output 11 = Reserved
GPB7	[15:14]	00 = Input 10 = nXDACK1	01 = Output 11 = Reserved
GPB6	[13:12]	00 = Input 10 = nXBREQ	01 = Output 11 = reserved
GPB5	[11:10]	00 = Input 10 = nXBACK	01 = Output 11 = reserved

GPBDAT 端口描述(见 S3C2440.pdf 第 284 页):

GPBDAT	Bit	Description
GPB[10:0]	[10:0]	When the port is configured as input port, the corresponding bit is the pin state. When the port is configured as output port, the pin state is the same as the corresponding bit. When the port is configured as functional pin, the undefined value will be read.

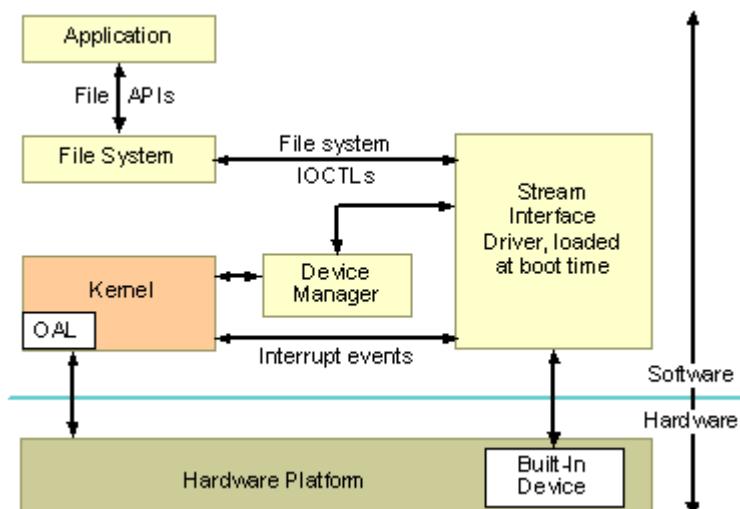
因此，要使用 GPB5~8，需要：

- (1) 设置 GPBCON，以使引脚功能为 IO 功能输出
- (2) 要点亮 LED, GPBDAT 对应的位要设置为 0
- (3) 要熄灭 LED, GPBDAT 对应的位要设置为 1

### 9.5.2 编写 LED 流式驱动程序

流式驱动是 WinCE 驱动程序的一种常规方式，应用程序通过文件系统，透过 DeviceManager 以访问文件的形式访问驱动程序，调用 IOCTL 向驱动程序下达指令。所有的流式驱动程序都需实现一组统一的接口。

流式驱动框架：



流式驱动程序接口函数(摘自 PB5 帮助文档)

Programming element	Description
<a href="#"><u>XXX_Close (Device Manager)</u></a>	This function is required to access the device with CreateFile. If you implement <b>XXX_Close</b> , you must implement <b>XXX_Open</b> .
<a href="#"><u>XXX_Deinit (Device Manager)</u></a>	This function is required by drivers loaded by <a href="#"><u>ActivateDeviceEx</u></a> , <a href="#"><u>ActivateDevice</u></a> , or <a href="#"><u>RegisterDevice</u></a> .



追求卓越 创造精品

TO BE BEST

TO DO GREAT

广州友善之臂计算机科技有限公司

<a href="#"><u>XXX_Init (Device Manager)</u></a>	This function initializes a device. It is called by Device Manager.  This function is required by drivers loaded by <b>ActivateDeviceEx</b> , <b>ActivateDevice</b> , or <b>RegisterDevice</b> .
<a href="#"><u>XXX_IOControl (Device Manager)</u></a>	This function sends a command to a device.  This function might or might not be required, depending on the device capabilities that the driver exposes. This function requires an implementation of <b>XXX_Open</b> and <b>XXX_Close</b> .
<a href="#"><u>XXX_Open (Device Manager)</u></a>	This function opens a device for reading, writing, or both. An application indirectly invokes this function when it calls <b>CreateFile</b> to obtain a handle to a device.  This function is required to access the device with <b>CreateFile</b> .
<a href="#"><u>XXX_PowerDown (Device Manager)</u></a>	Optional. This function ends power to the device. It is useful only with devices that can be shut off under software control.
<a href="#"><u>XXX_PowerUp (Device Manager)</u></a>	Optional. This function restores power to a device.
<a href="#"><u>XXX_Read (Device Manager)</u></a>	This function reads data from the device identified by the open context.  This function might or might not be required, depending on the device capabilities that the driver exposes.  This function requires an implementation of <b>XXX_Open</b> and <b>XXX_Close</b> .
<a href="#"><u>XXX_Seek (Device Manager)</u></a>	This function moves the data pointer in the device.  This function might or might not be required, depending on the device capabilities that the driver exposes.  This function requires an implementation of <b>XXX_Open</b> and <b>XXX_Close</b> .
<a href="#"><u>XXX_Write (Device Manager)</u></a>	This function writes data to the device.  This function might or might not be required, depending



追求卓越 创造精品

TO BE BEST

TO DO GREAT

广州友善之臂计算机科技有限公司

on the device capabilities that the driver exposes.

This function requires an implementation of **XXX\_Open** and **XXX\_Close**.

据此，我们要实现的 LED 驱动主要源代码如下：

```
BOOL WINAPI
DllEntry(HANDLE hinstDLL,
          DWORD dwReason,
          LPVOID /* lpvReserved */)

{
    switch(dwReason)
    {
        case DLL_PROCESS_ATTACH:
            DEBUGREGISTER((HINSTANCE)hinstDLL);
            return TRUE;
        case DLL_THREAD_ATTACH:
            break;
        case DLL_THREAD_DETACH:
            break;
        case DLL_PROCESS_DETACH:
            break;
#ifdef UNDER_CE
        case DLL_PROCESS_EXITING:
            break;
        case DLL_SYSTEM_STARTED:
            break;
#endif
    }
    return TRUE;
}
```

```
BOOL LED_Deinit(DWORD hDeviceContext)
{
    BOOL bRet = TRUE;

    RETAILMSG(1,(TEXT("USERLED: LED_Deinit\r\n")));

    return TRUE;
```



追求卓越 创造精品

TO BE BEST

TO DO GREAT

广州友善之臂计算机科技有限公司

}

```
BOOL LEDGpioInit()
{
    RETAILMSG(1,(TEXT("LED_Gpio_Setting----\r\n")));
    s2440IOP->rGPBCON = (s2440IOP->rGPBCON &~(3 << 10)) | (1 << 10); // GPB5 == OUTPUT.
    s2440IOP->rGPBCON = (s2440IOP->rGPBCON &~(3 << 12)) | (1 << 12); // GPB6 == OUTPUT.
    s2440IOP->rGPBCON = (s2440IOP->rGPBCON &~(3 << 14)) | (1 << 14); // GPB7 == OUTPUT.
    s2440IOP->rGPBCON = (s2440IOP->rGPBCON &~(3 << 16)) | (1 << 16); // GPB8 == OUTPUT.
    return TRUE;
}

DWORD LED_Init(DWORD dwContext)
{
    RETAILMSG(1,(TEXT("LED_Init----\r\n")));

    // 1. Virtual Alloc
    Virtual_Alloc();

    LEDGpioInit();

    mInitialized = TRUE;
    return TRUE;
}

//-----
//-----
BOOL LED_IOControl(DWORD hOpenContext,
                    DWORD dwCode,
                    PBYTE pBufIn,
                    DWORD dwLenIn,
                    PBYTE pBufOut,
                    DWORD dwLenOut,
                    PDWORD pdwActualOut)
{
    switch(dwCode)
    {
        case IO_CTL_LED_1_ON:
            s2440IOP->rGPBDAT=s2440IOP->rGPBDAT&~(0x1<<5);
    }
}
```



追求卓越 创造精品

TO BE BEST

TO DO GREAT

广州友善之臂计算机科技有限公司

```
case IO_CTL_LED_2_ON:  
    s2440IOP->rGPBDAT=s2440IOP->rGPBDAT&~(0x1<<6);  
    break;  
case IO_CTL_LED_3_ON:  
    s2440IOP->rGPBDAT=s2440IOP->rGPBDAT&~(0x1<<7);  
    break;  
case IO_CTL_LED_4_ON:  
    s2440IOP->rGPBDAT=s2440IOP->rGPBDAT&~(0x1<<8);  
    break;  
case IO_CTL_LED_ALL_ON:  
    s2440IOP->rGPBDAT=s2440IOP->rGPBDAT&~(0xF<<5);  
    break;  
case IO_CTL_LED_1_OFF:  
    s2440IOP->rGPBDAT=s2440IOP->rGPBDAT|(0x1<<5);  
    break;  
case IO_CTL_LED_2_OFF:  
    s2440IOP->rGPBDAT=s2440IOP->rGPBDAT|(0x1<<6);  
    break;  
case IO_CTL_LED_3_OFF:  
    s2440IOP->rGPBDAT=s2440IOP->rGPBDAT|(0x1<<7);  
    break;  
case IO_CTL_LED_4_OFF:  
    s2440IOP->rGPBDAT=s2440IOP->rGPBDAT|(0x1<<8);  
    break;  
case IO_CTL_LED_ALL_OFF:  
    s2440IOP->rGPBDAT=s2440IOP->rGPBDAT|(0xF<<5);  
    break;  
default:  
    break;  
}  
  
RETAILMSG(1,(TEXT("LED:Ioctl code = 0x%x\r\n"), dwCode));  
return TRUE;  
}  
  
//-----  
//-----  
DWORD LED_Open(DWORD hDeviceContext, DWORD AccessCode, DWORD ShareMode)  
{  
    RETAILMSG(1,(TEXT("USERLED: LED_Open\r\n")));  
    return TRUE;
```



追求卓越 创造精品

TO BE BEST

TO DO GREAT

广州友善之臂计算机科技有限公司

}

//-----  
//-----

BOOL LED\_Close(DWORD hOpenContext)  
{  
 s2440IOP->rGPBDAT=s2440IOP->rGPBDAT&(0xF<<5);  
 RETAILMSG(1,(TEXT("USERLED: LED\_Close\r\n")));  
 return TRUE;  
}

//-----  
//-----

void LED\_PowerDown(DWORD hDeviceContext)  
{  
 RETAILMSG(1,(TEXT("USERLED: LED\_PowerDown\r\n")));

//RETAILMSG(1,(TEXT("CAMERA: LED\_PowerDown, m\_Dx = D%u, init %d \r\n")), m\_Dx,  
mInitialized));  
}

//-----  
//-----

void LED\_PowerUp(DWORD hDeviceContext)  
{  
 RETAILMSG(1,(TEXT("USERLED: LED\_PowerUp\r\n")));  
}

//-----  
//-----

DWORD LED\_Read(DWORD hOpenContext, LPVOID pBuffer, DWORD Count)  
{  
 RETAILMSG(1,(TEXT("USERLED: LED\_Read\r\n")));  
 return TRUE;  
}

//-----  
//-----

DWORD LED\_Seek(DWORD hOpenContext, long Amount, DWORD Type)  
{



追求卓越 创造精品

TO BE BEST

TO DO GREAT

广州友善之臂计算机科技有限公司

```
RETAILMSG(1,(TEXT("USERLED: LED_Seek\r\n")));
return 0;
}

//-----
//-----

DWORD LED_Write(DWORD hOpenContext, LPCVOID pSourceBytes, DWORD NumberOfBytes)
{
    RETAILMSG(1,(TEXT("USERLED: LED_Write\r\n")));
    return 0;
}

DWORD LED_Seek(DWORD hOpenContext, long Amount, DWORD Type)
{
    RETAILMSG(1,(TEXT("USERLED: LED_Seek\r\n")));
    return 0;
}

//-----
//-----

DWORD LED_Write(DWORD hOpenContext, LPCVOID pSourceBytes, DWORD NumberOfBytes)
{
    RETAILMSG(1,(TEXT("USERLED: LED_Write\r\n")));
    return 0;
}
```

### 9.5.3 把 LED 驱动程序添加到 BSP 中以编译

LED 驱动程序必须要编译到内核中才能使用，要把 LED 驱动程序编译到内核中，需要做如下步骤：

(1) 在 smdk2440\DRIVERS 下建立 LEDdriver 目录，并在 dirs 文件中加入此目录，使系统编译 bsp 的时候可以编译这个文件

(2) 在 smdk2440\DRIVERS\LEDDriver\ 目录中建立 makefile 文件，内容如下：

```
# 
# DO NOT EDIT THIS FILE!!! Edit .\sources. if you want to add a new source
# file to this component. This file merely indirects to the real make file
# that is shared by all the components of Peg
#
!INCLUDE ${_MAKEENVROOT}\makefile.def
```



追求卓越 创造精品

TO BE BEST

TO DO GREAT

广州友善之臂计算机科技有限公司

(3)在 smdk2440\DRIVERS\LEDDriver\目录中建立 source 文件，内容如下：

!if 0

File: sources

Author: jeffmi

Copyright (c) 1995-2002 Microsoft Corporation. All rights reserved.

!endif

RELEASETYPE=PLATFORM

TARGETNAME=LEDDriver

TARGETTYPE=DYNLINK

DLLENTRY=DllEntry

TARGETLIBS= \

  \$\_COMMONSDKROOT)\lib\\$(CPUINDPATH)\coredll.lib \

MSC\_WARNING\_LEVEL = \$(MSC\_WARNING\_LEVEL) /W3 /WX

INCLUDES= \

  \$\_TARGETPLATROOT)\inc; \

  \$\_COMMONOAKROOT)\inc;\

  \$\_PUBLICROOT)\common\oak\inc;\$\_PUBLICROOT)\common\ sdk\inc;\$\_PUBLICROOT)\common\ddk\inc; \

  ..\..\inc

SOURCES= \

  leddriver.cpp \

(4)编写 leddriver.def 导出 Dll 符号：

;

; Windows CE LED Driver. Written by capbily

LIBRARY userLED

EXPORTS

LED\_Close

LED\_Deinit

LED\_Init

LED\_IOControl

LED\_Open



追求卓越 创造精品

TO BE BEST

TO DO GREAT

广州友善之臂计算机科技有限公司

LED\_PowerDown

LED\_PowerUp

LED\_Read

LED\_Seek

LED\_Write

(5)在配置文件 platform.bib 中加入以下内容:

```
;leddriver  
leddriver.dll      ${_FLATRELEASEDIR}\leddriver.dll NK SH
```

(6)在注册表文件 platform.reg 中加入以下内容:

```
[HKEY_LOCAL_MACHINE\Drivers\BuiltIn\LEDdriver]  
"Prefix"="LED"  
"Dll"="LEDDriver.dll"
```

(7)重新编译内核, 点 PB5 的主菜单 Build OS → Sysgen 即可, 这样就可以生成包含以上 LED 驱动的内核映象文件 NK.bin 和 NK.nb0 了。

#### 9.5.4 编写并编译 LED 测试应用程序

示例文件在光盘中的位置: \Windows CE5.0\QQ2440test

示例文件所使用的编译器: eMbedded Visual C++ 4.0 + SP4 补丁 + QQ2440\_SDK

SP4 补丁在光盘中的位置: \Embedded VisualC++\SP\evc4sp4\DISK1

QQ2440\_SDK 安装文件在光盘中的位置: \WindowsCE5.0\SDK

要在应用程序中调用驱动程序, IO Control 是最常用的方法, 它的一般流程如下:  
在应用程序中:

HANDLE

```
leddriver=CreateFile(L"LED1:",GENERIC_READ|GENERIC_WRITE,0,NULL,OPEN_EXISTING,0,NULL );  
ReadFile(leddriver, .....)  
DeviceIoControl(leddriver,IO_CTL_LED_CLEAR,NULL,0,NULL,0,NULL,NULL);
```

在驱动程序中:

当 CreateFile 被调用时, 调用 XXX\_Open

当 ReadFile 被调用时, 调用 XXX\_Read

当 DeviceIoControl 被调用时, 调用 XXX\_IOCTL, 根据 IO\_CTL\_XXX 的数值, 决定执行的操作。

在 LED 驱动中, 实现 10 个 IOCTL 分支, 处理 10 个命令请求, 对应点亮 1~4 号灯,,  
点亮所有 LED, 关闭 1~4 号等, 关闭所有 LED 灯。

LED 测试程序的主要代码如下:

```
#define IO_CTL_LED_1_ON 0x01  
#define IO_CTL_LED_2_ON 0x02
```



追求卓越 创造精品

TO BE BEST

TO DO GREAT

广州友善之臂计算机科技有限公司

```
#define IO_CTL_LED_3_ON 0x03
#define IO_CTL_LED_4_ON 0x04
#define IO_CTL_LED_ALL_ON 0x05
#define IO_CTL_LED_1_OFF 0x06
#define IO_CTL_LED_2_OFF 0x07
#define IO_CTL_LED_3_OFF 0x08
#define IO_CTL_LED_4_OFF 0x09
#define IO_CTL_LED_ALL_OFF 0x0a

void CKEYTESTDlg::OnButton1()
{
    // TODO: Add your control notification handler code here
    DeviceIoControl(leddriver,IO_CTL_LED_1_ON,NULL,0,NULL,0,NULL,NULL);

}

void CKEYTESTDlg::OnButton2()
{
    // TODO: Add your control notification handler code here
    DeviceIoControl(leddriver,IO_CTL_LED_2_ON,NULL,0,NULL,0,NULL,NULL);
}

void CKEYTESTDlg::OnButton3()
{
    // TODO: Add your control notification handler code here
    DeviceIoControl(leddriver,IO_CTL_LED_3_ON,NULL,0,NULL,0,NULL,NULL);
}

void CKEYTESTDlg::OnButton4()
{
    // TODO: Add your control notification handler code here
    DeviceIoControl(leddriver,IO_CTL_LED_4_ON,NULL,0,NULL,0,NULL,NULL);
}

void CKEYTESTDlg::OnButton5()
{
    // TODO: Add your control notification handler code here
    DeviceIoControl(leddriver,IO_CTL_LED_ALL_ON,NULL,0,NULL,0,NULL,NULL);
}

void CKEYTESTDlg::OnButton6()
```



追求卓越 创造精品

TO BE BEST

TO DO GREAT

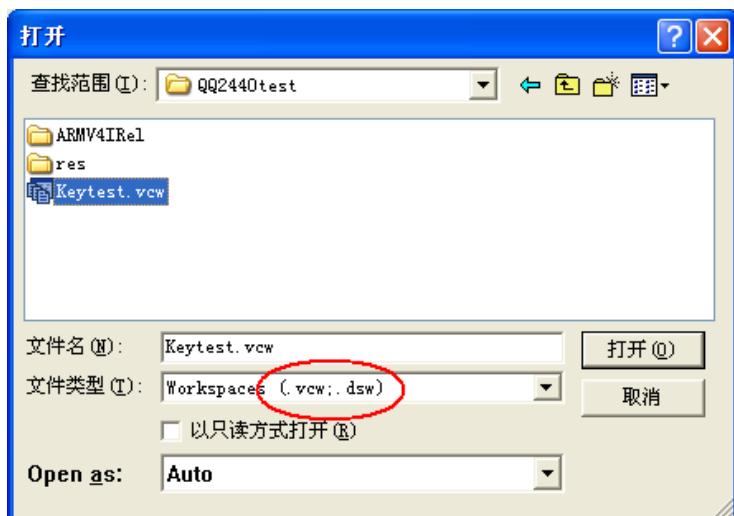
广州友善之臂计算机科技有限公司

```
{  
// TODO: Add your control notification handler code here  
DeviceIoControl(leddriver,IO_CTL_LED_1_OFF,NULL,0,NULL,0,NULL,NULL);  
  
}  
  
void CKEYTESTDlg::OnButton7()  
{  
// TODO: Add your control notification handler code here  
DeviceIoControl(leddriver,IO_CTL_LED_2_OFF,NULL,0,NULL,0,NULL,NULL);  
  
}  
  
void CKEYTESTDlg::OnButton8()  
{  
// TODO: Add your control notification handler code here  
DeviceIoControl(leddriver,IO_CTL_LED_3_OFF,NULL,0,NULL,0,NULL,NULL);  
  
}  
  
void CKEYTESTDlg::OnButton9()  
{  
// TODO: Add your control notification handler code here  
DeviceIoControl(leddriver,IO_CTL_LED_4_OFF,NULL,0,NULL,0,NULL,NULL);  
  
}  
  
void CKEYTESTDlg::OnButton10()  
{  
// TODO: Add your control notification handler code here  
DeviceIoControl(leddriver,IO_CTL_LED_ALL_OFF,NULL,0,NULL,0,NULL,NULL);  
}
```

要编译光盘中测 QQ2440test 测试程序，步骤如下：

(1)把 QQ2440test 复制到硬盘的某个目录，比如 D:\Work，去掉其只读属性。

(2)打开运行 EVC，点 File→Open，找到 QQ2440test 工程文件，如图：



(3)参考9.4一节的方法先设置好开发板连接(ActiveSync同步和网络都必须连接正常),点菜单Build→Rebuild All开始重新编译QQ2440test.exe程序,编译完毕,EVC会自动连接下载到开发板的\Windows目录,同时在D:\work\cetest\ARMV4IRel目录下生成QQ2440test.exe可执行程序,点击运行QQ2440test测试程序,如图,可以按照提示进行控制开发板上的LED。如图:



### 9.5.5 把LED测试程序添加到内核，并建立桌面快捷方式

为了把上一小节编译出的QQ2440test.exe可执行程序永久写入到NK.bin/NK.nb0中,以便每次开机时都能从桌面的快捷方式中执行它,我们采用以下方法步骤:

- (1)把QQ2440test.exe复制到smdk2440\FILES目录中。



追求卓越 创造精品

TO BE BEST

TO DO GREAT

广州友善之臂计算机科技有限公司

(2) 使用 Windows 自带的记事本程序创建 QQ2440test.lnk 快捷方式文件：

.lnk 文件其实是一个文本文件，它包含用于链接目标的命令行以及命令行的长度，其格式为“<length>#<command line>”，其中 length 是#后所有字符的个数，在此，QQ2440test.lnk 的内容如下：

23#\Windows\QQ2440test.exe

**注意：当把 QQ2440test.lnk 后缀改为.lnk 后，使用记事本一般就无法打开了。**

我们把这个文件也放入 smdk2440\FILES 目录中。

(3) 打开 platform.bib 文件，添加如下内容：

QQ2440test.exe        \$(\_FLATRELEASEDIR)\QQtest.exe        NK     U

QQ2440test.lnk        \$(\_FLATRELEASEDIR)\QQtest.lnk        NK     U

这样，执行 SYSGEN 的时候会把这两个文件加入到内核中，最后他们会存在于开发板的\Windows 目录中。

(4) 打开 platform.dat，加入以下内容：

Directory("\windows\桌面"):-File("QQ2440 测试.lnk","\windows\QQ2440test.lnk")

这将会在桌面出现名称位“QQ2440 测试”的快捷方式，它是 QQ2440test.lnk 的一个拷贝，其内容和 QQ2440test.lnk 是一样的。

(5) 最后执行菜单 Builder → Sysgen，生成 NK.bin 和 NK.nb0，把它们烧写或者下载到开发板启动后，就会在桌面看到“QQ2440 测试”快捷方式了。



结束语

本手册详细地向您介绍了如何在 mini2440 上开发和使用 Linux 或者 WindowsCE，这些都是一些基础性的东西，实际上，使用任何开发板都有一个类似的过程；我们相信，只要有兴趣，并且耐心的按照本手册去做了，您一定会掌握它，并体会到其中的乐趣；俗话说“师父领进门，修行在个人”，在技术不断更新和变革的今天，我们每个人都需要不断的学习和补充，我们希望能和您一起成长。



追求卓越 创造精品

TO BE BEST

TO DO GREAT

广州友善之臂计算机科技有限公司

# 附录 1 Qt 嵌入式图形开发入门

**注意：要正确安装使用 Qt/Embedded，请务必按照前面的介绍完全安装 Redhat9.0/Redhat7.2，并且不要使用其他发行版 Linux，诸如 Fedora 之类。**

交叉编译器选择推荐：**arm-linux-gcc-3.3.2**

## 1. 设置开发环境

光盘中包含三个 Qt/Embedded 源代码包，分别为：

x86-qtopia.tgz：用于在 X86 上运行的 Qtopia 开发包

arm-qtopia.tgz：用于在 ARM 上运行的 Qtopia 开发包，支持 USB 鼠标

ipaq-qtopia.tgz：用于在 ARM 上运行的 Qtopia 开发包，支持触摸屏

**三个版本的不同仅在于各自目录中的 build 编译脚本，其他的源代码都是一样的。**

解压源代码：

把光盘中的 x86-qtopia.tgz, arm-qtopia.tgz, ipaq-qtopia.tgz 复制到某一个目录，进入该目录执行以下命令：

```
#tar xvzf x86-qtopia.tgz -C /opt/FriendlyARM/mmini2440
```

```
#tar xvzf arm-qtopia.tgz -C /opt/FriendlyARM/mini2440
```

```
#tar xvzf ipaq-qtopia.tgz -C /opt/FriendlyARM/mini2440
```

该命令将把三种版本的 Qt/Embedded 包解压的到 /opt/FriendlyARM/mini2440 目录。

为了在 PC 上模拟运行 Qtopia，需要用到对应 Qt 版本的库文件，因此需要修改 /etc/ld.so.conf 文件以适应自己将要安装的 Qt 开发平台(Redhat 安装时带有 Qt 库，但不适合我们最新安装的版本)，修改后的 ld.so.conf 文件内容如下：

```
/opt/FriendlyARM/mini2440/x86-qtopia/qt/lib  
/opt/FriendlyARM/mini2440/x86-qtopia/qtopia/lib  
/usr/kerberos/lib  
/usr/X11R6/lib  
/usr/lib/sane  
/usr/lib/mysql
```

## 2. 编译 X86 平台的 Qtopia 和 Hello,World 和嵌入式浏览器

因为配置编译 Qt/Embedded 的过程比较复杂，为了方便，我们把配置和编译的步骤制作成一个 build 脚本，执行该脚本即可“一键搞定”。

## 2.1 编译 Qt/Embedded

```
#cd /opt/FriendlyARM/mini2440/x86-qtopia
```

```
#./build-all      (该过程比较长, 需要运行大概 30 分钟左右)
```

```
#./konq_to_qtopia (把编译出的浏览器放入 Qtopia 执行系统)
```

```
#ldconfig          (运行 ldconfig 是为了使生成的 qt 和 qtopia 库有效, 运行一次即可)
```

说明: **./build-all** 将自动编译完整的 Qtopia 和嵌入式浏览器, 您还可以先后执行**./build** 和**./build-konq** 脚本命令分别编译它们。

## 2.2 在 PC 上模拟运行 Qtopia

#. set-env ; 注意, 中间有个空格, 必须要有才能执行有效!

```
#qvfb&
```

```
#qpe
```



在 PC 上使用 QVFB 模拟运行 Qtopia

## 2.3 编译 Hello, World 示例

```
#cd /opt/FriendlyARM/mini2440/x86-qtopia/hello
```

```
#make
```

执行 make 将在 /opt/FriendlyARM/mini2440/x86-qtopia/qtopia/bin 目录下生成 hello 可执行文件

## 2.4 单独运行 Hello, World

```
#qvfb -width 640 -height 480 &
```

```
#hello -qws
```

如图所示



在 *qvfb* 中单独运行 *Hello, World*

## 2.5 在 Qtopia 中运行 Hello,World

```
#cd hello
```

```
#cp hello.desktop /opt/FriendlyARM/mini2440/x86-qtopia/qtopia/apps/Applications
```

```
#qvfb -width 640 -height 480 &
```

```
#qpe
```

如图所示



在 qtopia 中的 Hello, World

### 3 编译 ARM 平台的 Qtopia 和 Hello,World 和嵌入式浏览器

**提醒: 请使用光盘附带的 arm-linux-gcc-3.3.2.tgz 交叉编译器编译 ARM 平台的 Qtopia, 否则有可能会不能通过!**

因为配置编译 Qt/Embedded 的过程比较复杂, 为了方便, 我们把配置和编译的步骤制作成一个 build 脚本, 执行该脚本即可“一键搞定”。

#### 3.1 编译 Qt/Embedded

```
#cd /opt/FriendlyARM/mini2440/arm-qtopia
```

```
./build-all      (该过程比较长, 需要运行大概 30 分钟左右)
```

```
./mktarget_qtopia  (制作更新文件系统用的 qtopia 压缩包, 将生成 target_qtopia.tgz)
```

```
./mktarget_konq  (制作更新文件系统用的浏览器压缩包, 将生成 target_konq.tgz)
```

说明: **./build-all** 将自动编译完整的 Qtopia 和嵌入式浏览器, 并且编译生成的系统支持 Jpeg、GIF、PNG 等格式的图片, 您还可以先后执行**./build** 和**./build-konq** 脚本命令分别编译它们。



追求卓越 创造精品

TO BE BEST

TO DO GREAT

广州友善之臂计算机科技有限公司

## 3.2 编译 Hello, World 示例

```
#cd /opt/FriendlyARM/mini2440/arm-qtopia
```

#. set-env ; 注意，中间有个空格，必须要有才能执行有效！

```
#cd hello
```

```
#make
```

执行 make 将在 /opt/FriendlyARM/mini2440/arm-qtopia/qtopia/bin 目录下生成 hello 可执行文件

## 3.3 把 hello,world 下载到目标板并运行

### ①复制文件到 Windows 目录下

先把刚刚编译生成的 hello 可执行文件复制到 Windows 下的某个目录里面，同时把 hello/ 目录中的 hello.desktop 也复制到 Windows 下的某个目录。

### ②使用 rz 下载文件到开发板

在开发板串口终端输入“rz”命令开始接收从串口发来的文件，如图所示



追求卓越 创造精品

TO BE BEST

TO DO GREAT

广州友善之臂计算机科技有限公司

```
SBC2440 UDA1341 audio driver initialized interface: v1.00ox, bzcat, cal, cat,
NET: Registered protocol family 2 can be used to force w
IP route cache hash table entries: 1024 (order: 0, 4096 bytes)

TCP reno registered
TCP bic registered
NET: Registered protocol family 1
NET: Registered protocol family 17
MMC: sd_app_op_cond: at least one card is busy - trying again.
mmcblk0: mmc0:c734 SD016 14400KiB
/dev/mmcblk0:<7>MMC: starting cmd 37 arg c7340000 flags 00000009
p1
yaffs: dev is 32505858 name is "mtdblock2"
yaffs: Attempting MTD mount on 31.2, "mtdblock2"
VFS: Mounted root (yaffs filesystem).
Mounted devfs on /dev
Freeing init memory: 148K
[26/Jun/2007:17:00:30 +0000]
Please press Enter to activate this console. boa: server version Boa/0.94.13
[26/Jun/2007:17:00:30 +0000] boa: server built Feb 28 2004 at 21:47:23.
[26/Jun/2007:17:00:30 +0000] boa: starting server pid=278, port 80

[root@FriendlyARM /]# ls
bin      home      net      root      usr
dev      lib       opt      sbin      var
etc      lost+found  proc     tmp      www
[root@FriendlyARM /]# rz
? **0B000000023be50ve.**0B000000023be50
```

已连接 1:26:26 ANSIW 115200 8-N-1 SCROLL CAPS NUM 插 打印

然后点鼠标右键，在弹出的菜单中选择“发送文件”，选择 hello，开始向开发板传送文件。如图。





追求卓越 创造精品

TO BE BEST

TO DO GREAT

广州友善之臂计算机科技有限公司



使用同样的方法下载 hello.desktop 到开发板里面

### ③改变 hello 文件的执行权限

通过串口现在到开发板的文件是没有执行权限的，所以我们需要先使用 chmod 命令改变它的执行权限，再把它放到正确的目录里面。如图所示

```
#chmod +x hello  
#mv hello /opt/qtopia/bin  
#mv hello.desktop /opt/qtopia/apps/Applications
```



追求卓越 创造精品

TO BE BEST

TO DO GREAT

广州友善之臂计算机科技有限公司

```
[root@FriendlyARM /]# ls
bin          home          opt          tmp
dev          lib           proc         usr
etc          lost+found   root         var
hello.desktop  mnt          sbin        www
[root@FriendlyARM /]# rz
? [root@FriendlyARM /]# 0B000000023be50
[root@FriendlyARM /]# ls
bin          hello.desktop  mnt          sbin        www
dev          home          opt          tmp
etc          lib           proc         usr
hello        lost+found   root         var
[root@FriendlyARM /]# chmod +x hello
[root@FriendlyARM /]# ls
bin          hello.desktop  mnt          sbin        www
dev          home          opt          tmp
etc          lib           proc         usr
hello        lost+found   root         var
[root@FriendlyARM /]# mv hello /opt/qtopia/bin/
[root@FriendlyARM /]# mv hello.desktop /opt/qtopia/apps/Applications/
[root@FriendlyARM /]# -
```

已连接 1:46:14 ANSIW | 115200 8-N-1 | SCROLL | CAPS | NUM | 捕 | 打印 |

#### ④在开发板上运行 hello

现在重新启动开发板或者重新启动 qtopia，就可以看到 hello 图标了，如图，您可以使  
用鼠标点击运行它。

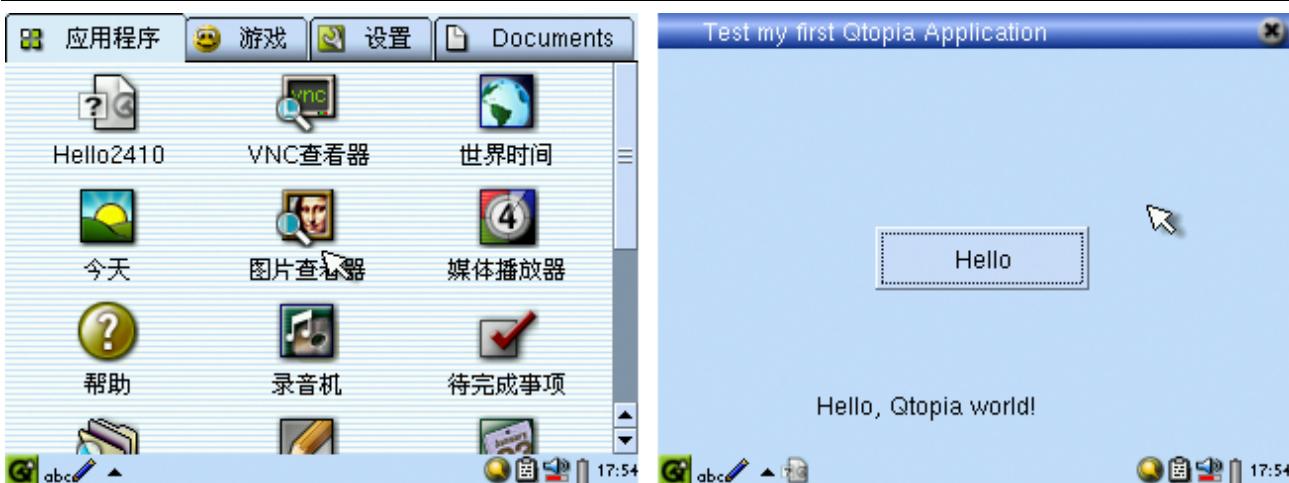


追求卓越 创造精品

TO BE BEST

TO DO GREAT

广州友善之臂计算机科技有限公司



#### ⑤在目标板上单独运行 hello

```
export set HOME=/root
export set QTDIR=/opt/qt
export set QPEDIR=/opt/qtopia
export set QWS_KEYBOARD="USB:/dev/input/event1"
export set QWS_MOUSE_PROTO="USB:/dev/input/mouse0"
export set PATH=$QPEDIR/bin:$PATH
export set LD_LIBRARY_PATH=$QTDIR/lib:$QPEDIR/lib
$QPEDIR/bin/hello -qws
```

### 3.4 使用自己编译的 Qtopia 更新制作文件系统

通过 3.1 一节我们已经制作好了 **target\_qtopia.tgz** 和 **target\_konq.tgz** 这两个文件，下面我们将使用它们更新替换原来的 qtopia 系统。

以带鼠标支持的 qtopia 系统为例：

```
#cd /opt/FriendlyARM/mini2440/root_qtopia_mouse
#tar xvzf ..../arm-qtopia/target_qtopia.tgz (更新 Qtopia 系统)
#tar xvzf ..../arm-qtopia/target_konq.tgz (更新嵌入式浏览器)
#mkyaffsimage root_qtopia_mouse my_qtopia.img
```

然后使用 USB 把 **my\_qtopia.img** 烧入板子就可以了。

## 4 常见问题

初学者安装 Qt/Embedded 会经常遇到以下问题，请仔细查看相应的解决方法。



追求卓越 创造精品

TO BE BEST

TO DO GREAT

广州友善之臂计算机科技有限公司

## 4.1 执行 build 时出现的错误

./build 很长时间后,最后显示如下

Makefiles will be regenerated.

.....

QPE is now configured for building. Just run "make".

To reconfigure, run make clean and configure.

```
make -C libraries/qtopia
make[1]: Entering directory
`/opt/FriendlyARM/QQ2440/x86-qtopia/qtopia/src/libraries/qtopia'
make[1]: Nothing to be done for `default'.
make[1]: Leaving directory
`/opt/FriendlyARM/QQ2440/x86-qtopia/qtopia/src/libraries/qtopia'
make -C 3rdparty/libraries/freetype
make[1]: Entering directory
`/opt/FriendlyARM/QQ2440/x86-qtopia/qtopia/src/3rdparty/libraries/freetype'
make[1]: Nothing to be done for `default'.
make[1]: Leaving directory
`/opt/FriendlyARM/QQ2440/x86-qtopia/qtopia/src/3rdparty/libraries/freetype'
make -C libraries/qtopia1
make[1]: Entering directory
`/opt/FriendlyARM/QQ2440/x86-qtopia/qtopia/src/libraries/qtopia1'
g++ -c -I/opt/FriendlyARM/QQ2440/x86-qtopia/qtopia/include
-I/opt/FriendlyARM/QQ2440/x86-qtopia/qt/include -pipe -DQWS -fno-exceptions
-fno-rtti -O2 -Wall -W -DNO_DEBUG -fPIC -DQTPIA_APP_INTERFACE
-I.moc/linux-generic-g++/ -I/opt/FriendlyARM/QQ2440/x86-qtopia/qtopia/src/server -o
.obj/linux-generic-g++//global1.o global1.cpp
global1.cpp:39:23: uuid/uuid.h: 没有那个文件或目录
global1.cpp: In static member function `static QUuid
Global::generateUuid()':
global1.cpp:188: `::uuid_generate' undeclared (first use here)
make[1]: *** [.obj/linux-generic-g++//global1.o] Error 1
make[1]: Leaving directory
`/opt/FriendlyARM/QQ2440/x86-qtopia/qtopia/src/libraries/qtopia1'
make: *** [libraries/qtopia1] Error 2
```

这是因为你没有正确安装 Redhat 9.0 所致,请参考 前面的章节正确安装您的 Redhat 9.0 系统



追求卓越 创造精品

TO BE BEST

TO DO GREAT

广州友善之臂计算机科技有限公司

## 4.2 编译 hello 时出现的错误

信息如下：

```
/usr/lib/qt-3.1/bin/uic hello_base.ui -o ./hello_base.h
g++ -c -I/include -I/usr/lib/qt-3.1/include -pipe -DQWS -fno-exceptions
-fno-rtti -O2 -Wall -W -DNO_DEBUG -DQTPIA_APP_INTERFACE
-I.moc/linux-generic-g++/ -o .obj/linux-generic-g++//hello.o hello.cpp
g++ -c -I/include -I/usr/lib/qt-3.1/include -pipe -DQWS -fno-exceptions
-fno-rtti -O2 -Wall -W -DNO_DEBUG -DQTPIA_APP_INTERFACE
-I.moc/linux-generic-g++/ -o .obj/linux-generic-g++//main.o main.cpp
main.cpp:2:35: qtopia/qpeapplication.h: No such file or directory
main.cpp: In function `int main(int, char**)':
main.cpp:6: `QPEApplication' undeclared (first use this function)
main.cpp:6: (Each undeclared identifier is reported only once for each
function
it appears in.)
main.cpp:6: parse error before `(' token
main.cpp:9: `a' undeclared (first use this function)
main.cpp:5: warning: unused parameter `int argc'
main.cpp:5: warning: unused parameter `char**argv'
make: *** [.obj/linux-generic-g++//main.o] Error 1
```

从错误信息中可以看出，编译 hello 的时候这里用的是-I/include 头文件目录，这个目录是不正确的，导致这样的原因是：您没有在 build 完之后运行 ldconfig 命令。

## 4.3 编译 hello 时出现的第二种错误信息

```
[root@localhost hello]# make
/usr/lib/qt-3.1/bin/uic hello_base.ui -o ./hello_base.h
/usr/lib/qt-3.1/bin/uic: error while loading shared libraries:
libqt-mt.so.3: cannot open shared object file: No such file or directory
make: *** [hello_base.h] Error 127
```

从错误信息中可见，您的 x86-qtopia 环境并没有搭建好，依然使用 redhat9.0 自带的环境。解决办法如下：

第一种办法：请务必在同一个窗口执行以上步骤。运行 build 脚本时，已经包含了环境的创建，您不能离开 build 运行后所在的窗口环境来编译 hello

第二种办法：自己创建重新创建环境。即使用/opt/FriendlyARM/x86-qtopia/set-env 脚本。运行时请务必使用“. set-env”，注意中间有个空格，同样，您不能离开 set-env 运行后所在的窗口环境来编译 hello

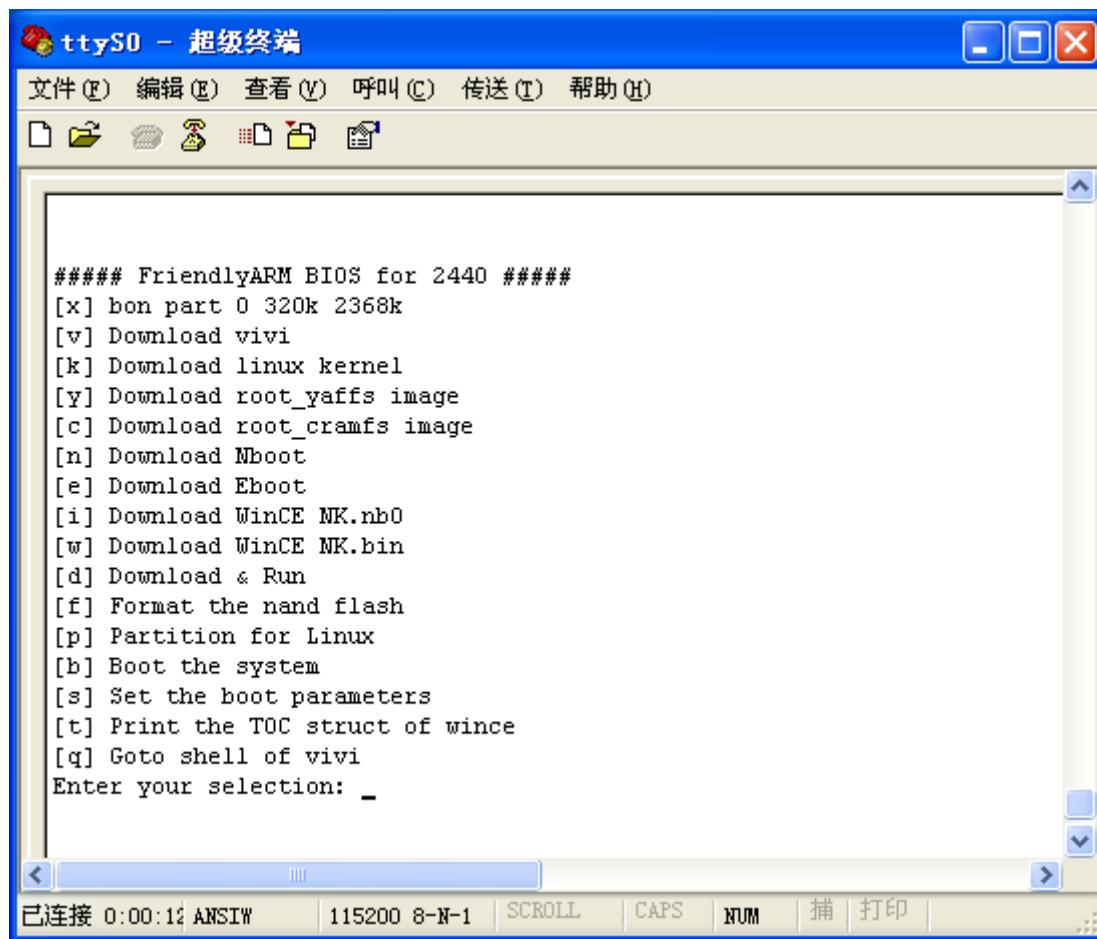
## 附录 2 使用 BIOS 的命令行更新和烧写系统

说明：我们推荐使用前面介绍的功能菜单方式下载更新目标板系统，本小节内容仅供习惯了命令行方式的老用户参考，不提供技术支持，也将不再继续更新本小节内容。

### 1.1. 如何进入 BIOS 的命令行模式

#### 1.1.1 从功能菜单进入命令行模式

如果 Supervivi 被烧写入 Nor Flash，并选择从 Nor Flash 启动，则系统开机启动时会进入功能菜单模式，这时选择功能号[q]可以进入命令行模式，如图。



## 1.1.2 在 Nand Flash 启动时进入命令行模式

如果 Supervivi 被烧写入 Nand Flash，并选择从 Nand Flash 启动时，连接好串口和 USB 接口，同时打开运行超级终端和 DNW，在超级终端界面下一直按住 PC 键盘的空格键，开启开发板的电源，这样您就可以进入命令行模式了，如下图。

注意：如果输出信息不像下图这样完善，有可能烧写没有成功。因为这种 JTAG 烧写方法是单向的，没有校验的过程，所以您可以按照上面的方法再烧写一次试试。

```
Genre: Goa

VIVI version 0.1.4 (root@localhost.localdomain) (gcc version 2.95.3 20010315 (release)) #0.1.4 Sat Jul 21 11:51:25 CST 2007
MMU table base address = 0x33DFC000
Succeed memory mapping.
DIVN_UPLL0
MPLLVal [M:7fh,P:2h,S:1h]
CLKDIVN:5h

+-----+
| S3C2440A USB Downloader ver R0.03 2004 Jan |
+-----+
USB: IN_ENDPOINT:1 OUT_ENDPOINT:3
FORMAT: <ADDR(DATA):4>+<SIZE(n+10):4>+<DATA:n>+<CS:2>
NOTE: Power off/on or press the reset button for 1 sec
      in order to get a valid USB device address.

NAND device: Manufacture ID: 0xec, Chip ID: 0x76 (Samsung K9D1208U0M)
Could not found stored vivi parameters. Use default vivi parameters.
Press Return to start the LINUX/Wince now, any other key for vivi
type "help" for help.
Supervivi> _
```

以下步骤均假定选择从 Nand Flash 启动系统，和 Nor Flash 无关。

## 2.2 安装 linux

说明：安装 linux 所需要的二进制文件位于光盘的 **image/linux** 目录中。

打开 DNW 程序，不必点串口连接，我们将使用 Windows 自带的超级终端作为操作终端，因为它比 DNW 更好用一些。



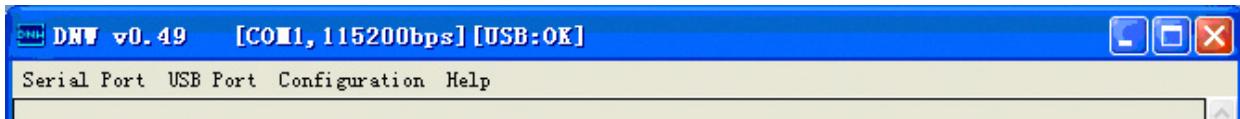
追求卓越 创造精品

TO BE BEST

TO DO GREAT

广州友善之臂计算机科技有限公司

在 BIOS 模式下，使用 USB 线连接 PC 的 USB 接口和开发板的 USB Device 接口，如果您已经安装了 USB 驱动，这时可以看到 DNW 的标题栏提示 USB 已经连接 OK。



安装 Linux 系统主要有以下步骤：

- (1) 格式化 Nand Flash
- (2) 安装 bootloader
- (3) 安装内核文件
- (4) 安装文件系统

下面是详细的步骤。

## 2.2.1 对 Nand Flash 进行分区

在 BIOS 模式下输入： **bon part 0 320k 2368k** 对板子进行分区。

说明： bon 是分区命令，以上命令的意思是把 Nand Flash 从 0 开始分为三个区：

0-320k： 大小为 320k

320k-2368k： 大小为 2M

2368k-64M： 大小为 62M

说明：有的开发板可能分区后会出现提示坏的扇区，请不必理会，系统将会自动处理这些坏区；另外这种现象是该型号三星 Nand Flash 的特性，用户可以到三星网站下载 Flash 型号文档查看。



追求卓越 创造精品

TO BE BEST

TO DO GREAT

广州友善之臂计算机科技有限公司

```
ttyS0 - 超级终端
文件(E) 编辑(E) 查看(V) 呼叫(C) 传送(I) 帮助(H)
Creating 3

Press Return to start the LINUX/Wince now, any other key for vivi
type "help" for help.
Supervivi> bon part 0 320k 2368k
doing partition
size = 0
size = 327680
size = 2424832
check bad block
part = 0 end = 327680
part = 1 end = 2424832
part = 2 end = 67108864
part0:
    offset = 0
    size = 327680
    bad_block = 0
part1:
    offset = 327680
    size = 2097152
    bad_block = 0
part2:
    offset = 2424832
    size = 64667648
    bad_block = 0
Supervivi> _
```

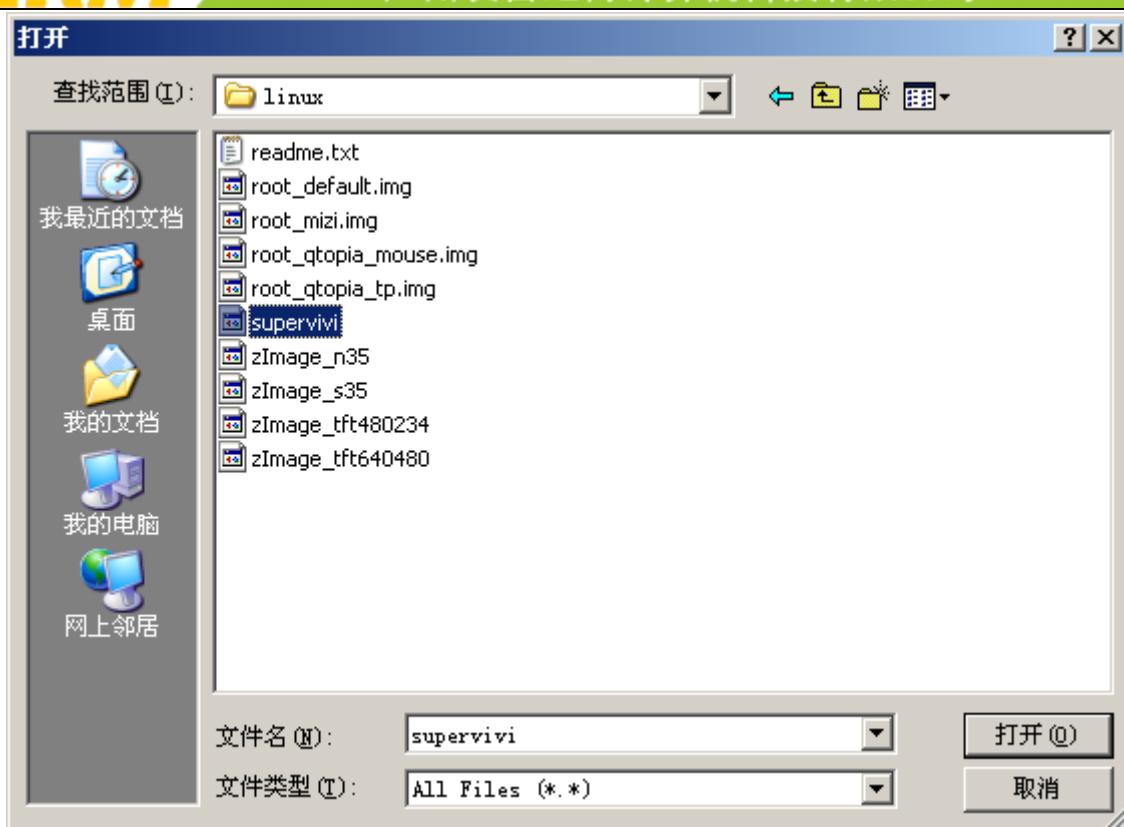
已连接 5:45:34 ANSIW 115200 8-N-1 SCROLL CAPS NUM 插 打印

## 2.2.2 恢复 BIOS

**注意：**以上分区操作就格式化了整片 Nand Flash，结果就是 Flash 里面什么都没有了，而此时的 supervivi 是在内存里面运行的，当你关机了，就要重新来烧写它了，所以此时不要关闭电源。

接上面的步骤，输入：load flash vivi u

此时出现如下提示界面，板子等待用户进行 USB 下载传输，点 DNW 程序的 USB Port->Transmit，找到并选择 supervivi 开始下载。



下载完毕，BIOS 会自动烧写 supervivi 到 Nand Flash 分区中，并返回到主菜单，如图：



追求卓越 创造精品

TO BE BEST

TO DO GREAT

广州友善之臂计算机科技有限公司

```
MPLLVal [M:7fh,P:2h,S:1h]eduler anticipator
+--+
| S3C2440A USB Downloader ver R0.03 2004 Jan |
+--+
USB: IN_ENDPOINT:1 OUT_ENDPOINT:3
FORMAT: <ADDR(DATA):4>+<SIZE(n+10):4>+<DATA:n>+<CS:2>
NOTE: Power off/on or press the reset button for 1 sec
      in order to get a valid USB device address.

NAND device: Manufacture ID: 0xec, Chip ID: 0x76 (Samsung K9D1208U0M)
Could not found stored vivi parameters. Use default vivi parameters.
Press Return to start the LINUX/Wince now, any other key for vivi
type "help" for help.
Supervivi>
Supervivi> load flash vivi u
USB host is connected. Waiting a download.

Now, Downloading [ADDRESS:30000000h, TOTAL:116950]
RECEIVED FILE| SIZE: 116950 (114KB/S, 1S)
Downloaded file at 0x30000000, size = 116940 bytes
Found block size = 0x00020000
Erasing...    ... done
Writing...    ... done
Written 116940 bytes
Supervivi> _
```

已连接 5:49:27 ANSIW 115200 8-N-1 SCROLL CAPS NUM 捕 打印

提示: 用户也可以使用 **load flash vivi x** 命令, 通过超级终端的 *xmodem* 协议来下载程序。

### 3.2.3 烧写 linux 内核

接上一步骤, 输入: **load flash kernel u**

此时点击 DNW 的 USB Port->Transmit 选择您所需要的的内核文件开始下载。下载完毕, Linux 内核文件将会被自动烧写到 Nand Flash, 如图(提示: 此处选择的是支持三星 3.5 寸 LCD 的内核文件):

内核文件说明:

zImage\_s35

- 支持三星 3.5 寸屏的内核文件

zImage\_n35

- 支持 NEC 3.5" LCD 的内核文件

zImage\_tft640480

- 支持 640x480 分辨率的真彩 LCD 内核。

实际可能与此不完全相同, 请参考 images/linux 目录下的 readme.txt 文件说明

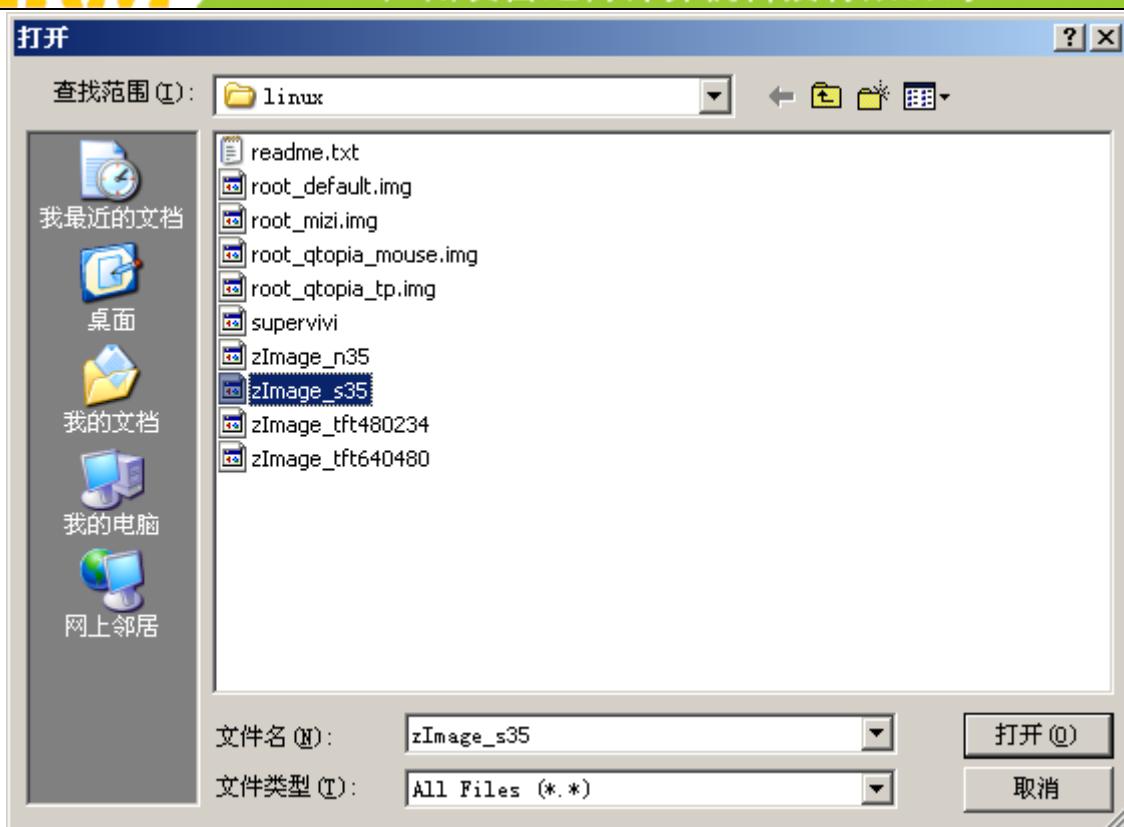


追求卓越 创造精品

TO BE BEST

TO DO GREAT

广州友善之臂计算机科技有限公司



下载完毕，BIOS 会自动烧写内核到 Nand Flash 分区中，并返回到主菜单，如图：



追求卓越 创造精品

TO BE BEST

TO DO GREAT

广州友善之臂计算机科技有限公司

```
Press Return to start the LINUX/Wince now, any other key for vivi
type "help" for help.
Supervivi>
Supervivi> load flash vivi u
USB host is connected. Waiting a download.

Now, Downloading [ADDRESS:30000000h,TOTAL:116950]
RECEIVED FILE SIZE: 116950 (114KB/S, 1S)
Downloaded file at 0x30000000, size = 116940 bytes
Found block size = 0x00020000
Erasing... ... done
Writing... ... done
Written 116940 bytes
Supervivi> load flash kernel u
USB host is connected. Waiting a download.

Now, Downloading [ADDRESS:30000000h,TOTAL:1556638]
RECEIVED FILE SIZE: 1556638 (760KB/S, 2S)
Downloaded file at 0x30000000, size = 1556628 bytes
Found block size = 0x00180000
Erasing... ... done
Writing... ... done
Written 1556628 bytes
Supervivi> _
```

提示：用户也可以使用 `load flash kernel x` 命令，通过超级终端的 `xmodem` 协议来下载程序，不过速度比较慢。

### 3.2.4 烧写基于 yaffs 的根文件系统

接上一步，输入：`loadyaffs root u`

点击“USB Port->Transmit”选项，并选择打开相应的文件系统映象文件 `root_default.img`(该文件位于光盘的 `images/linux/` 目录)开始下载，如图所示选择是缺省的文件系统映象文件 `root_default.img`：

根文件系统映象文件说明：

`root_default.img`

- 缺省安装的文件系统映象文件，采用基于 `arm-linux-gcc-3.4.1` 的函数库

`root_mizi.img`

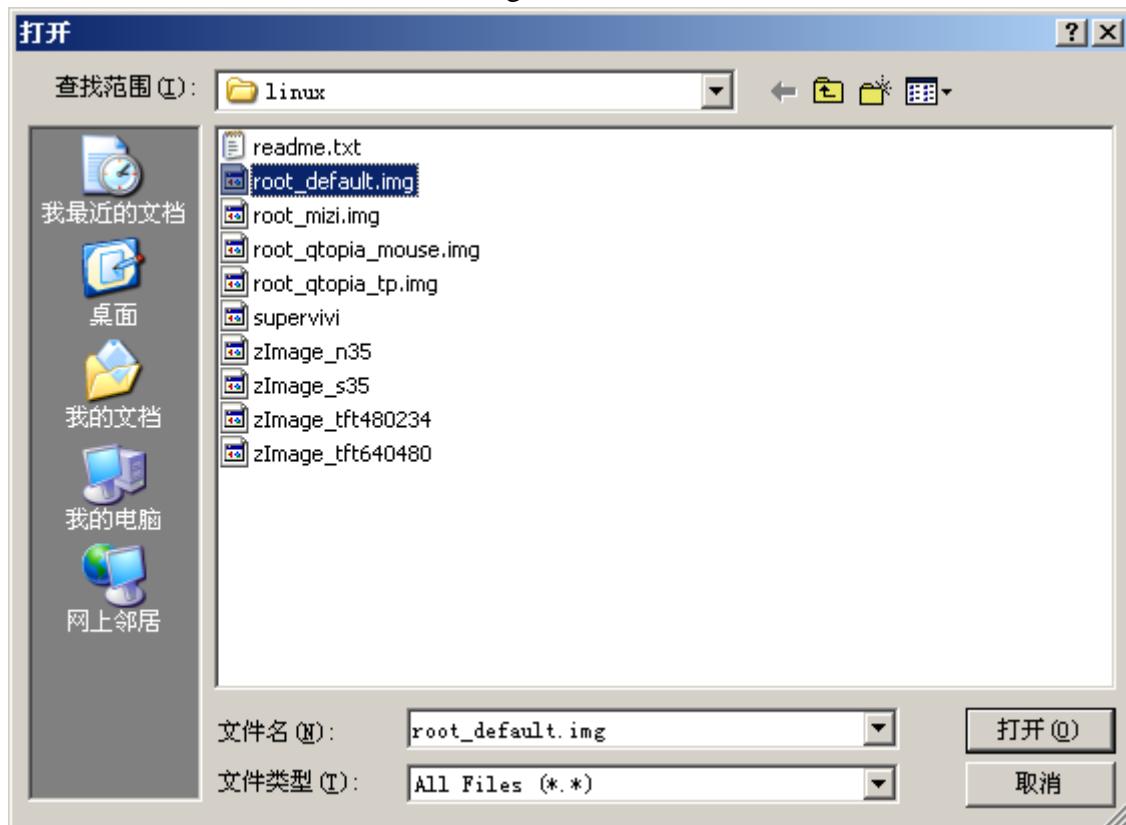
- mizi 公司提供的映象文件，含有中文手写识别，浏览器等。

`root_qtopia_mouse.img`

- 标准的 qtopia，使用鼠标操作,采用基于 `arm-linux-gcc-3.4.1` 的函数库

root\_qtopia\_tp.img

- 标准的 qtopia, 使用触摸屏操作。采用基于 arm-linux-gcc-3.4.1 的函数库  
实际可能与此不完全相同, 请参考 images/linux/目录下的 readme.txt 文件说明



下载完毕, BIOS 会自动烧写内核到 Nand Flash 分区中, 并返回到主菜单, 如图:



追求卓越 创造精品

TO BE BEST

TO DO GREAT

广州友善之臂计算机科技有限公司

```
USB host is connected. Waiting a download.: 0x76 (Samsung K9D1208U0M) 3 MTD part

Now, Downloading [ADDRESS:30000000h, TOTAL:1556638]
RECEIVED FILE SIZE: 1556638 (760KB/S, 2S)
Downloaded file at 0x30000000, size = 1556628 bytes
Found block size = 0x00180000
Erasing... ... done
Writing... ... done
Written 1556628 bytes
Supervivi> loadyaffs root u
USB host is connected. Waiting a download.

Now, Downloading [ADDRESS:30000000h, TOTAL:29548474]
Downloaded file at 0x30000000, size = 29548464 bytes
Flash params: oobsize = 16, oobblock = 512, erasesize = 16384, partition size =
64667648
Erasing and programming NAND with yaffs image
Block erasing(addr/count) --- Block bad(addr/count) --- Block processed/All(%)
-----
0x03FF8000/03947          0x00000000/000000          03947/03947=100%
Load yaffs OK:
Blocks scanned: 3947, Blocks erased: 3947, Blocks are bad: 0
RECEIVED and Writed FILE SIZE:29548474 (280KB/S, 103S)
Supervivi> _
```

提示：此过程大概需要 2-3 分钟，下载的文件越大，下载和烧写的时间就会越长；  
loadyaffs root x 命令是不存在的。

### 3.2.5 启动系统

下载完毕，请拔下 USB 连接线，如果不取下来，有可能在复位或者启动系统的时候导致您的电脑死机。

在 BIOS 提示符下，输入 boot，或者复位系统，或者重新开启电源，将会自动启动系统。

## 3.3 安装 wince

说明：安装 WINCE 所需要的二进制文件位于光盘的 image/wince 目录中。最  
打开 DNW 程序，不必点串口连接，我们将使用 Windows 自带的超级终端作为操作终  
端，因为它比 DNW 更好用一些。

在 BIOS 模式下，使用 USB 线连接 PC 的 USB 接口和开发板的 USB Device 接口，如果



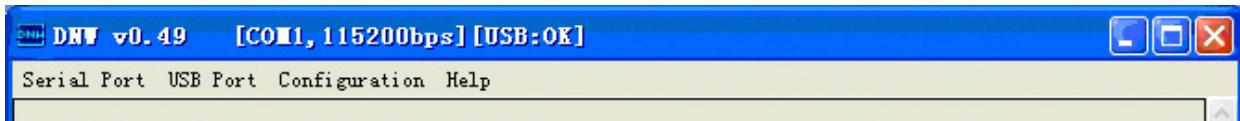
追求卓越 创造精品

TO BE BEST

TO DO GREAT

广州友善之臂计算机科技有限公司

您已经按照上面的章节安装了 USB 驱动，这时可以看到 DNW 的标题栏提示 USB 已经连接 OK。



安装 WindowsCE 系统主要有以下步骤：

- (1) 格式化 Nand Flash
- (2) 重新安装 BIOS
- (3) 安装 Eboot
- (4) 安装 WindowsCE 内核映象

下面是详细的步骤。

### 3.3.1 对 Nand Flash 进行分区

在 BIOS 模式下输入： **bon part 0 320k 2368k** 对板子进行分区。

说明： bon 是分区命令，以上命令的意思是把 Nand Flash 从 0 开始分为三个区：

0-320k： 大小为 320k

320k-2368k： 大小为 2M

2368k-64M： 大小为 62M

说明：有的开发板可能分区后会出现提示坏的扇区，请不必理会，系统将会自动处理这些坏区；另外这种现象是该型号三星 Nand Flash 的特性，用户可以到三星网站下载 Flash 型号文档查看。

**注意：分区后不要关电或者掉电，因为此时 Nand Flash 中已经被清空，需要按照下面的步骤再重新下载一次 BIOS，否则你将需要使用 SJF2440.exe 再次下载一次。**



追求卓越 创造精品

TO BE BEST

TO DO GREAT

广州友善之臂计算机科技有限公司

ttyS0 - 超级终端

文件(E) 编辑(E) 查看(V) 呼叫(C) 传送(I) 帮助(H)

Creating 3

```
Press Return to start the LINUX/Wince now, any other key for vivi
type "help" for help.
Supervivi> bon part 0 320k 2368k
doing partition
size = 0
size = 327680
size = 2424832
check bad block
part = 0 end = 327680
part = 1 end = 2424832
part = 2 end = 67108864
part0:
    offset = 0
    size = 327680
    bad_block = 0
part1:
    offset = 327680
    size = 2097152
    bad_block = 0
part2:
    offset = 2424832
    size = 64667648
    bad_block = 0
Supervivi> _
```

已连接 5:45:34 ANSIW 115200 8-N-1 SCROLL CAPS NUM 捕 打印

### 3.3.2 恢复 BIOS

接上面的步骤，输入： **load flash vivi u**

此时出现如下提示界面，板子等待用户进行 USB 下载传输，点 DNW 程序的 **USB Port->Transmit**，找到并选择 supervivi 开始下载，下载完毕，supervivi 将会被自动烧写到 Nand Flash。

提示：用户也可以使用**load flash vivi x** 命令，通过超级终端的 *xmodem* 协议来下载程序。



追求卓越 创造精品

TO BE BEST

TO DO GREAT

广州友善之臂计算机科技有限公司

ttyS0 - 超级终端

文件(E) 编辑(E) 查看(V) 呼叫(C) 传送(I) 帮助(H)

MPLLVal [M:7fh,P:2h,S:1h]eduler anticipator

```
+--+
| S3C2440A USB Downloader ver R0.03 2004 Jan |
+--+
USB: IN_ENDPOINT:1 OUT_ENDPOINT:3
FORMAT: <ADDR(DATA):4>+<SIZE(n+10):4>+<DATA:n>+<CS:2>
NOTE: Power off/on or press the reset button for 1 sec
      in order to get a valid USB device address.

NAND device: Manufacture ID: 0xec, Chip ID: 0x76 (Samsung K9D1208U0M)
Could not found stored vivi parameters. Use default vivi parameters.
Press Return to start the LINUX/Wince now, any other key for vivi
type "help" for help.
Supervivi>
Supervivi> load flash vivi u
USB host is connected. Waiting a download.

Now, Downloading [ADDRESS:30000000h, TOTAL:116950]
RECEIVED FILE| SIZE: 116950 (114KB/S, 1S)
Downloaded file at 0x30000000, size = 116940 bytes
Found block size = 0x00020000
Erasing...    ... done
Writing...    ... done
Written 116940 bytes
Supervivi> _
```

已连接 5:49:27 ANSIW 115200 8-N-1 SCROLL CAPS NUM 捕 打印

### 3.3.3 烧写 Eboot

在 vivi 模式下，输入：load flash eboot u

此时点击 USB Port->Transmit 选择 eboot.nb0 文件开始下载。下载完毕，Linux 内核文件将会被自动烧写到 Nand Flash。

提示：用户也可以使用 **load flash eboot x** 命令，通过超级终端的 *xmodem* 协议来下载程序。



追求卓越 创造精品

TO BE BEST

TO DO GREAT

广州友善之臂计算机科技有限公司

ttyS0 - 超级终端

文件(E) 编辑(E) 查看(V) 呼叫(C) 传送(I) 帮助(H)

□ ■ ○ ×

```
Now, Downloading [ADDRESS:30000000h,TOTAL:29548474]
Downloaded file at 0x30000000, size = 29548464 bytes
Flash params: oobsize = 16, oobblock = 512, erasesize = 16384, partition size =
64667648
Erasing and programming NAND with yaffs image
Block erasing(addr/count) --- Block bad(addr/count) --- Block processed/All(%)
-----
0x03FF8000/03947          0x00000000/000000          03947/03947=100%
Load yaffs OK:
Blocks scanned: 3947, Blocks erased: 3947, Blocks are bad: 0
RECEIVED and Writed FILE SIZE:29548474 (280KB/S, 103S)
Supervivi> load flash eboot u
USB host is connected. Waiting a download.

Now, Downloading [ADDRESS:30000000h,TOTAL:90122]
RECEIVED FILE SIZE: 90122 (88KB/S, 1S)
Downloaded file at 0x30000000, size = 90112 bytes
Found block size = 0x00018000
Erasing... ... done |
Writing... ... done
Written 90112 bytes
Supervivi> _
```

已连接 5:54:40 ANSIW 115200 8-N-1 SCROLL CAPS NUM 捕 打印

### 3.3.4 烧写 wince 内核

在 BIOS 模式下，输入： **load flash wince u**

此时 eboot 将会运行，并提示用户通过 USB 下载，点击 USB Port->Transmit 选择相应的内核文件开始下载。

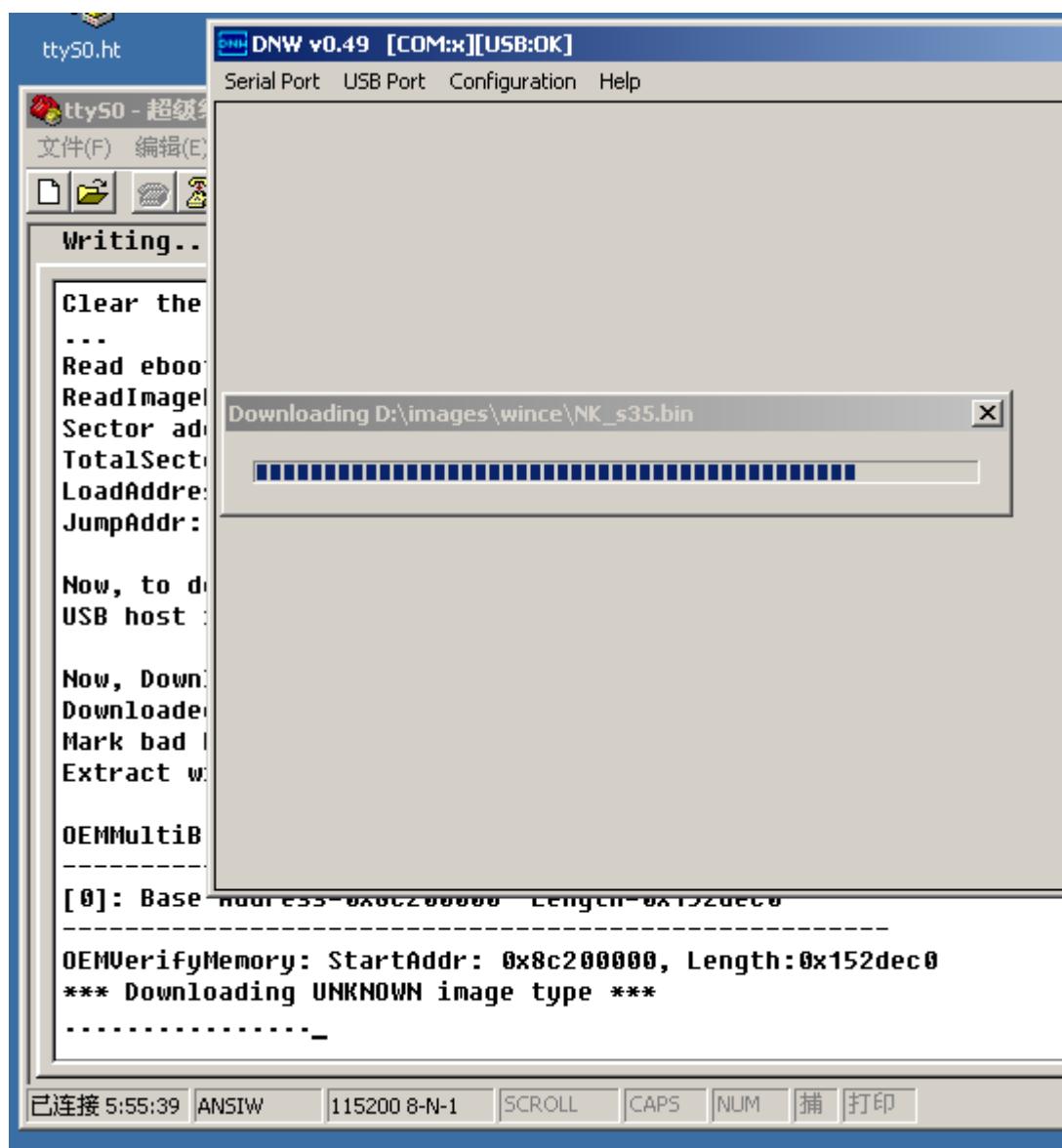


追求卓越 创造精品

TO BE BEST

TO DO GREAT

广州友善之臂计算机科技有限公司



烧写过程如图所示。



追求卓越 创造精品

TO BE BEST

TO DO GREAT

广州友善之臂计算机科技有限公司

ttyS0 - 超级终端

文件(E) 编辑(E) 查看(V) 呼叫(C) 传送(I) 帮助(H)

dwLoadAddress: 0x0.Clocks, (c

```
chainInfo.dwLoadAddress: 0X00000000
chainInfo.dwFlashAddress: 0X00000000
chainInfo.dwLength: 0X00000000
}
g_pViviWinceInfo = 0x301D8000, g_pViviWinceInfo->dwViviWinceMagic = 0x12345678
Low-level Format nand flash ...
Reserving Blocks [0x0 - 0x13] ...
...reserve complete.
Low-level Format Blocks [0x14 - 0xFFFF] ...
...erase complete.
Format nand flash for BinFS, please wait several minutes ...
System ready!
Preparing for download...
Found pTOC signature.
ROMHDR at Address 8C200044h
RomHdr.ulRAMStart=8E600000h RomHdr.physFirst=8C200000h.
::OEMLaunch, ImageStart:0x8C200000, ImageLength:0x152DEC0, LaunchAddr:0x8C201000

OEMLaunch: (IMAGE_TYPE_RAMIMAGE|IMAGE_TYPE_BINFS)
+WriteRegionsToBootMedia: ImageStart: 0x8C200000, ImageLength: 0x152DEC0, Launch
Addr:0x8C201000
INFO: OEMLaunch: Found chain extenstion: '' @ 0x8C200000
Writing single region/multi-region update, dwBINFSPartLength: 22208192
```

已连接 5:58:27 ANSIW 115200 8-N-1 SCROLL CAPS NUM 捕 打印

下载完毕, eboot 程序将会自动将 Nand Flash 进行低级格式化, 并进一步格式化为 BinFS, 格式化完毕, 再自动把 WindowsCE 内核文件烧写到 Nand Flash, 烧写完毕会自动启动 WinCE 系统, 入下图所示, 注意: 此过程比较长, 大概需要 5-8 分钟。



追求卓越 创造精品

TO BE BEST

TO DO GREAT

广州友善之臂计算机科技有限公司

tty50 - 超级终端

文件(E) 编辑(E) 查看(V) 呼叫(C) 传送(I) 帮助(H)

HW\_Init : CreateEventri

```
<PWR_Init:0x307b0
>PWR_Open(0x307b0, 0x0, 0x3)
<PWR_Open:1
>PWR_IOControl(0x321000, 0x0, 0, 0x6030850)
<PWR_IOControl:1
>PWR_Open(0x307b0, 0x0, 0x3)
<PWR_Open:2
PWR_Close(0x307b0)
384 clock
    USB:OhcdPdd_Init
++InitializeOHCI
USB:*pIrq=11, *pioPortBase=0x260000
OHCD: MapIrq2SysIntr(11): 27
OHCD: Memory Object
--InitializeOHCI
+CS8900:DriverEntry
USB enable interrupt
    charlie::SDIO::SDHOST::SDCSDCardDllEntry::DLL_PROCESS_ATTACH
    charlie::SDIO::SDCInitialize+
    charlie::SDIO::SDCInitialize-
--S3C2440DISP::InitializeHardware
Touch Init
RasEntry 'USB Socket Default' Created
```

已连接 5:59:08 ANSIW 115200 8-N-1 SCROLL CAPS NUM 捕 打印

## 附录 3 使用 SJF2440 烧写 BIOS

**SJF2440** 是由三星提供的用来烧写开发板 Flash 的工具程序，它可以通过并口连接一个 Jtag 板，用来烧写 K9F1208 Nand Flash, AMD29LV800BB Nor Flash 等型号的 Flash。因为 **SJF2440** 是一种十分慢速的烧写工具，并且没有校验功能，它一般适用于烧写比较小的 bootloader；友善之臂出品的 ARM 板系统一般都带有 **Nor Flash**，并使用目前比较流行的 **H-JTAG** 进行快速烧写 **BIOS**。鉴于有些用户设计的系统有可能不使用 **Nor Flash**，因此我们介绍一下如何在 WindowsXP 上安装使用它，并使用它来烧写 Nand Flash。

**注意 1：要使用 JTAG 板进行烧写，建议使用 Intel 芯片组的主板，否则有可能烧写失败。**

### 1 安装 GIVEIO 驱动

(1) 请以系统管理员的身份登录 WindowsXP，复制光盘中的 giveio.sys 到 C:\WINDOWS\system32\drivers 目录



(2) 打开"控制面板->添加硬件->"，按照向导进行操作：



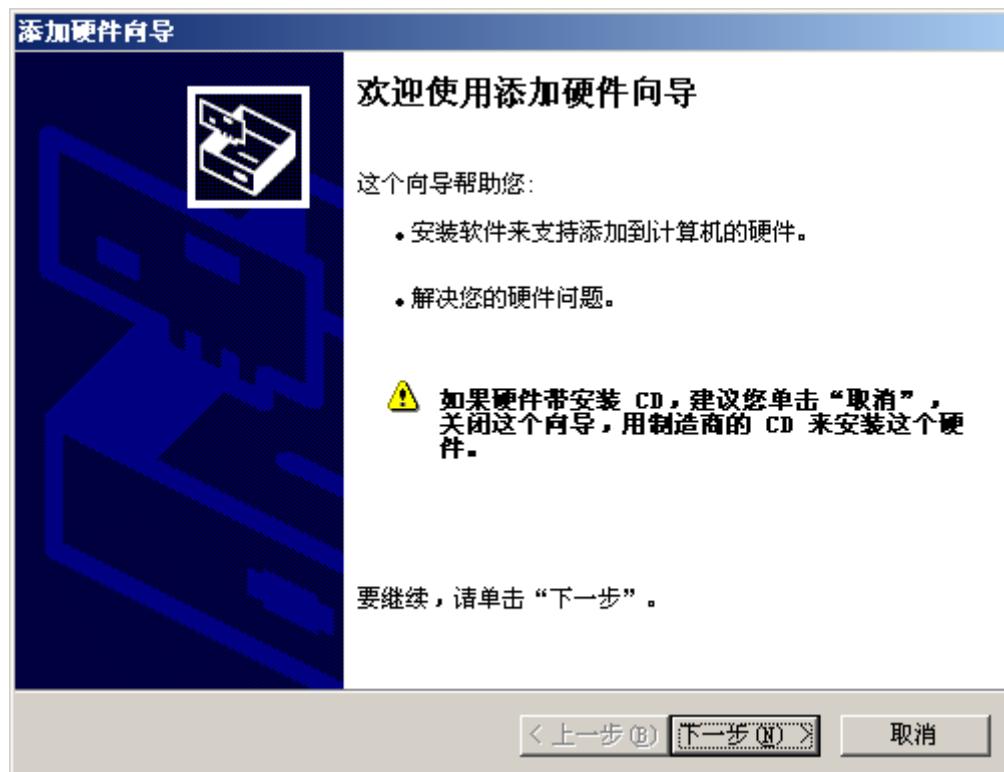
追求卓越 创造精品

TO BE BEST

TO DO GREAT

广州友善之臂计算机科技有限公司

Step1：开始安装



Step2：选择“是的，我已经连接了此硬件”，这时不必连接实际的 JTAG 板。





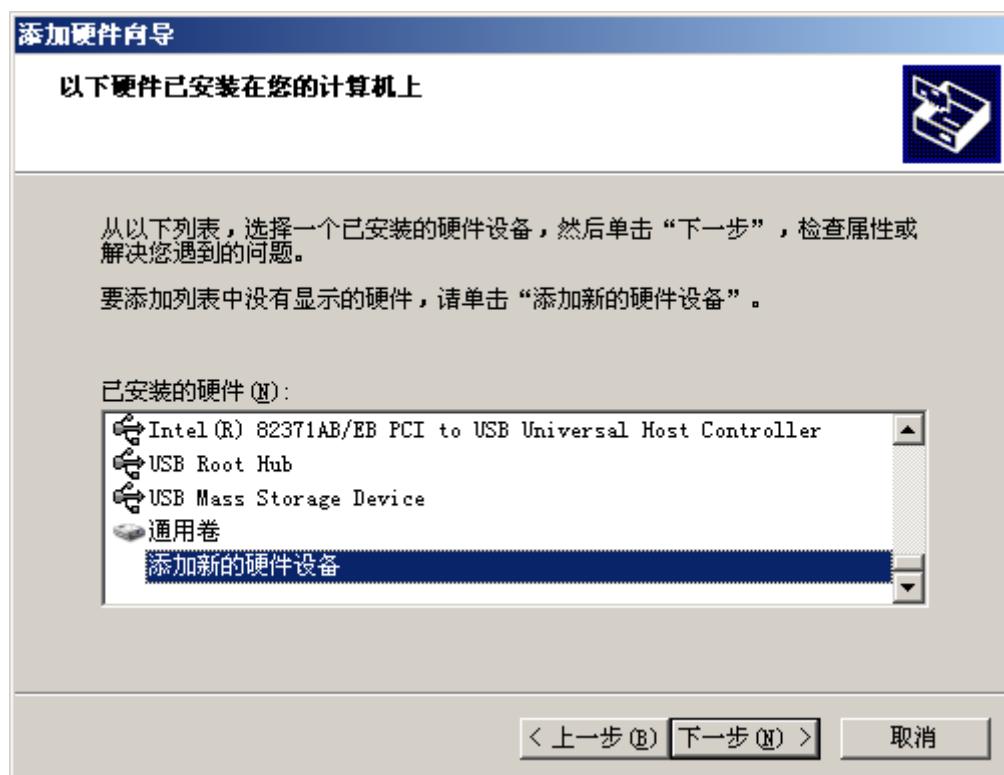
追求卓越 创造精品

TO BE BEST

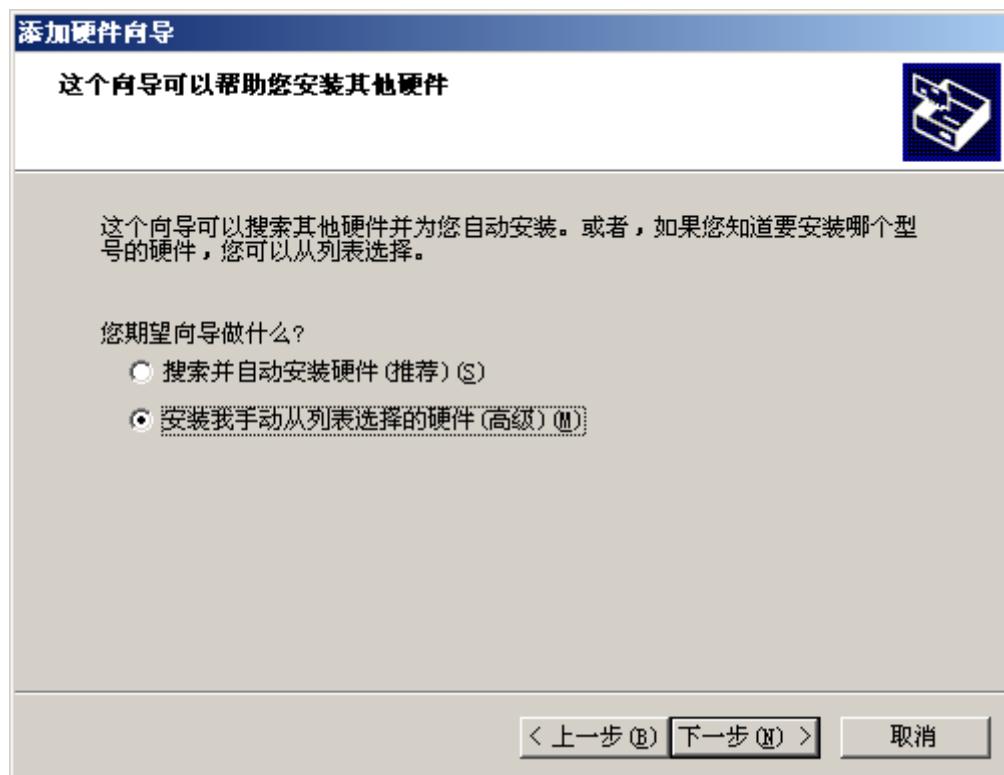
TO DO GREAT

广州友善之臂计算机科技有限公司

Step3: 选择“添加新的硬件设备”。



Step4: 选择“安装我手动从列表选择的硬件”，如图。





追求卓越 创造精品

TO BE BEST

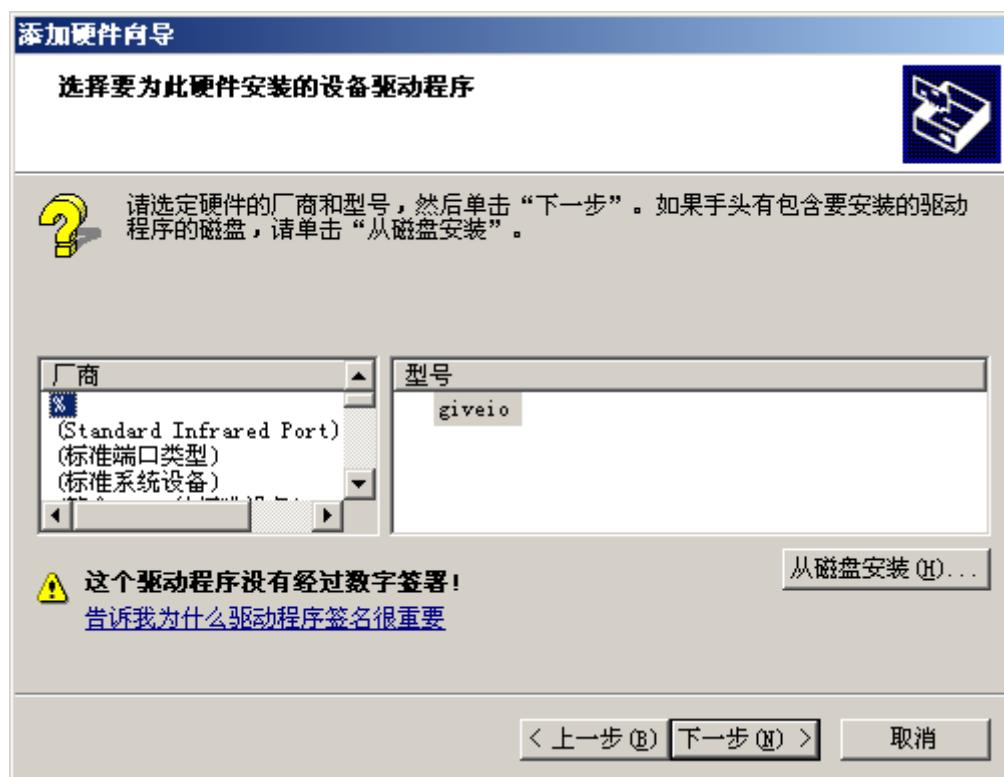
TO DO GREAT

广州友善之臂计算机科技有限公司

Step5：不选任何选项，直接点“下一步”，如图



Step6：不选左右两侧列表中的任何选项，直接点“从磁盘安装”。





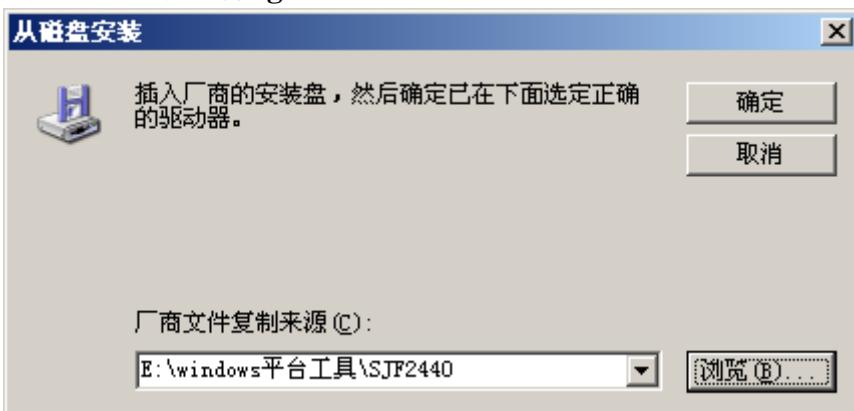
追求卓越 创造精品

TO BE BEST

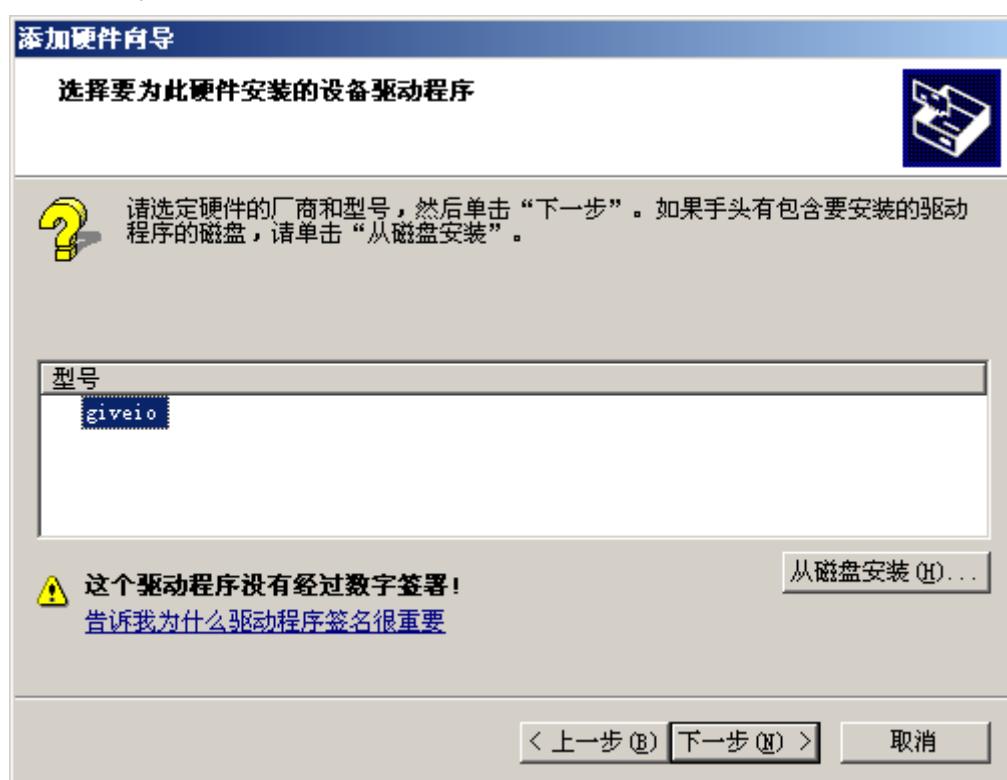
TO DO GREAT

广州友善之臂计算机科技有限公司

Step7: 选择要安装的驱动文件 giveio.ini



Step8: 点“下一步”



Step9: 点“下一步”



追求卓越 创造精品

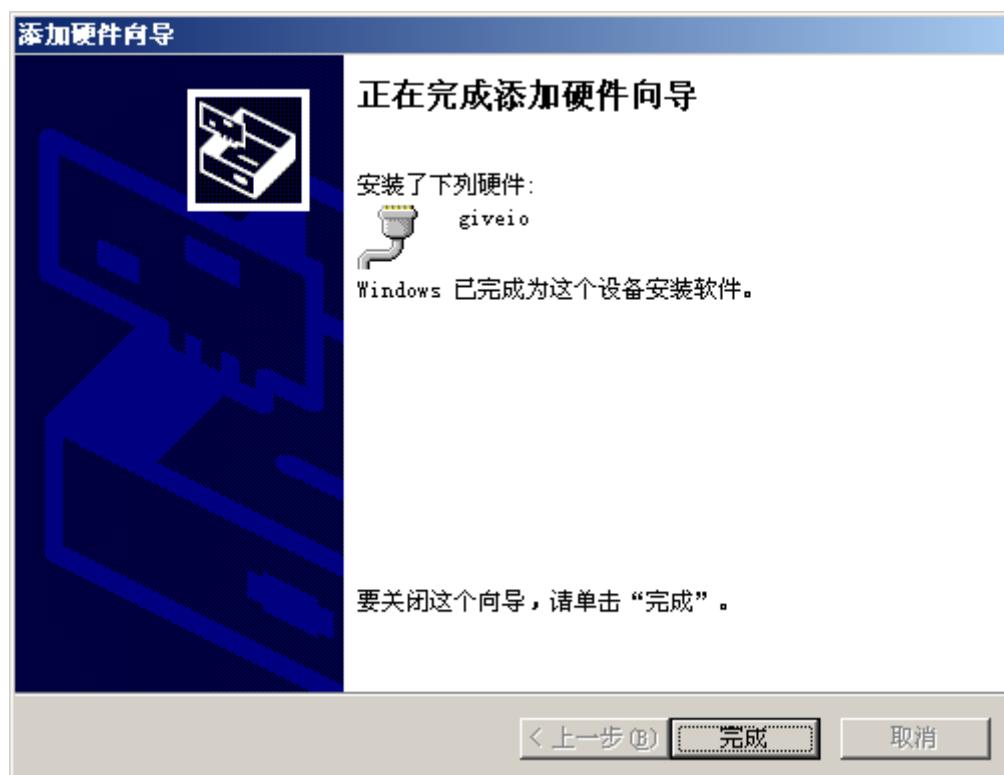
TO BE BEST

TO DO GREAT

广州友善之臂计算机科技有限公司



Step10: 安装成功



## 2 使用 SJF2440 烧写 BIOS

以上驱动安装完毕，就可以使用我们提供的 Jtag 板进行 BIOS 烧写了。请把 Jtag 板的并口端连接到 PC 机的并口上，当然您可以使用并口延长线；再把 Jtag 板另一端的 20 针排线连接到开发板的 JTAG 接口上。

(1) 把光盘的 `image` 目录复制到 C 盘(或者其他地方也可以)，双击运行 `SJF2440_supervivi.BAT` 批处理文件，出现如下烧写界面，里面列出了支持的 Flash 型号，我们在“Select the Function to test:”提示下，输入“2”，然后回车，开始选择 Nor Flash(AM29LV160) 进行烧写。

```
C:\WINDOWS\system32\cmd.exe - sjf2440.exe /f:supervivi
C:\sbc2440_img>sjf2440.exe /f:supervivi

+-----+
|      SEC JTAG FLASH<SJF> v 0.1          |
|      <S3C2440X & SMDK2440 B/D>           |
+-----+
Usage: SJF /f:<filename> /d=<delay>
> S3C2440X<ID=0x0032409d> is detected.

[SJF Main Menu]
0:K9S1208 prog    1:28F128J3A prog    2:AM29LV800 Prog   3:Memory Rd/Wr
4:Exit
Select the function to test:2

[AM29F800 Writing Program]
NOTE: AM29LV800BB needs 4 step sequences for 1 half-word data.
      So, the program time is twice of Starata flash(2 step sequences).
[Check AM29LV800]
Manufacture ID= 1, Device ID=2249

Image Size:0h~1c720h

Available Target Offset:
  0x0, 0x4000, 0x6000, 0x8000, 0x10000, 0x20000, 0x30000, 0x40000,
  0x50000, 0x60000, 0x70000, 0x80000, 0x90000, 0xa0000, 0xb0000, 0xc0000,
  0xd0000, 0xe0000, 0xf0000
Input target offset:0

SectorOffset=0x0
SectorSize =0x4000
Erase the sector:0x0.
Sector Erase is started!
Start of the sector data writing.
0 100 200 300 400 500 600 700 ..
```

(2) 接着在“Available Taget Offse”提示下，输入偏移地址“0”，开始进行烧写，显示信息如上图所示。

烧写完毕，输入“2”，程序将自动退出。



追 求 卓 越 创 造 精 品

TO BE BEST

TO DO GREAT

广州友善之臂计算机科技有限公司