DevKit8000 评估套件用户手册

版本5.1

发布: 2010-05-13

版本更新记录

版本	发布日期	描述
1.0	2009-03-26	初始发布
2.0	2009-4-28	第二版
2.1	2009-4-29	修改了“8.1 LED 应用程序开发示例”部分
2.2	2009-5-14	修正了“5.2 显示方式选择”中参数设置的遗留
2.3	2009-8-6	修改了“5.2 显示方式选择”和“5.4 Demo 系统演示”， VGA 支持。
3.0	2009-9-8	第三版
4.0	2009-10-19	第四版
4.1	2009-12-28	修正了 2G 板子 wince 网络死机问题
5.0	2010-03-08	WinCE 增加 GPRS,GPS 模块驱动
5.1	2010-05-13	Wince 解决 Bpreaddata 问题，wifi 不能识别问题

接洽信息

访问如下地址，可获取更多信息: **http://www.timll.com**

感谢您购买我们的产品，因您我们会做得更好。

深圳市天溟科技有限公司（英蓓特全资子公司）
地址：深圳市罗湖区太宁路 85 号罗湖科技大厦 405 室
邮编：518020
电话：0755-25500944 25631357 25635656
传真：0755-25616057
技术支持电话：0755-25503401
Email: sales@timll.com
网址：www.timll.com

接洽信息
访问如下地址，可获取更多信息: **http://www.timll.com**

目录

第一章 概述

本文档主要介绍 DevKit8000 评估套件的开发使用方法，详细描述了 DevKit8000 评估主板的硬件接口特性，并对软件系统的应用开发进行指导。

1.1 产品简介

DevKit8000 评估套件是深圳市天漠科技有限公司推出的基于德州仪器（TI）OMAP3530 处理器的评估套件。OMAP3530 处理器集成了 600MHz 的 ARM Cortex™-A8 内核及 430MHz 的具有高级数字信号处理算法的 DSP 核，并提供了丰富的外设接口。DevKit8000 外扩了 CPU 外设接口中的网口、S-VIDEO 接口、音频输入输出接口、USB OTG、USB HOST、SD/MMC 接口、串口、SPI 接口、IIC 接口、JTAG 接口、CAMERA 接口、TFT 屏接口、触摸屏接口、键盘接口和总线接口，并扩展出了 HDMI 接口。

DevKit8000 评估套件为开发者使用 OMAP3530 处理器提供了完善的软件开发平台，支持 linux-2.6.28 及 WinCE 6.0 操作系统，并包含完善的底层驱动程序，方便用户快速评估 OMAP35x 处理器、设计系统驱动及其定制应用软件，并提供有成熟的操作系统 google android 及 angstrom(GPE)的发布版本，DVI 输出可达到 720P 的显示标准，方便用户体验 OMAP3530 处理器的强大的数据运算处理能力。

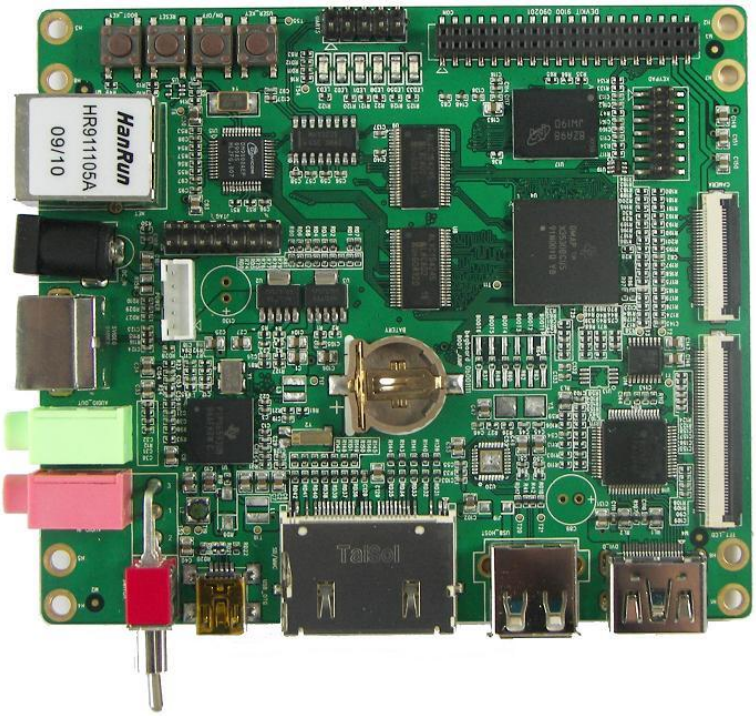


图1.1 DevKit8000评估主板

1.2 定义

HDMI 接口 ：High Definition Multimedia Interface 接口，即高清晰多媒体接口。
DVI 接口 ：Digital Visual Interface 接口，即数字视频接口。
GPMC总线 ：General-Purpose Memory Controller，OMAP3530的系统总线。

1.3 产品配件

DevKit8000 评估套件分两种配置：标准配置和完全配置。

- 标准配置：着重评估板的基本功能，主要针对具有一定开发条件的板级开发者；
- 完全配置：包含完善的接口配件的支持，具备了 LCD 屏等相关配件，该配置主要针对特定应用的专业产品开发人员。

配件	标准配置	完全配置
DevKit8000 评估板	√	√
SD 卡（容量：512M）	√	√
交叉串口线	√	√
电源适配器（5V）	√	√
4.3” LCD 屏（带触摸屏）		√
触摸笔		√
USB 转接线（Mini-B to A）		√
USB 转接线（Mini-A to A）		√
USB HUB		√
网线		√
HDMI 转 DVI-D 转接线		√
S-Video 线		√

第二部分 硬件系统

第二章 硬件概述

2.1 结构框图

图 2.1 为 DevKit8000 评估套件的结构框图，主要介绍该板的外设设备。

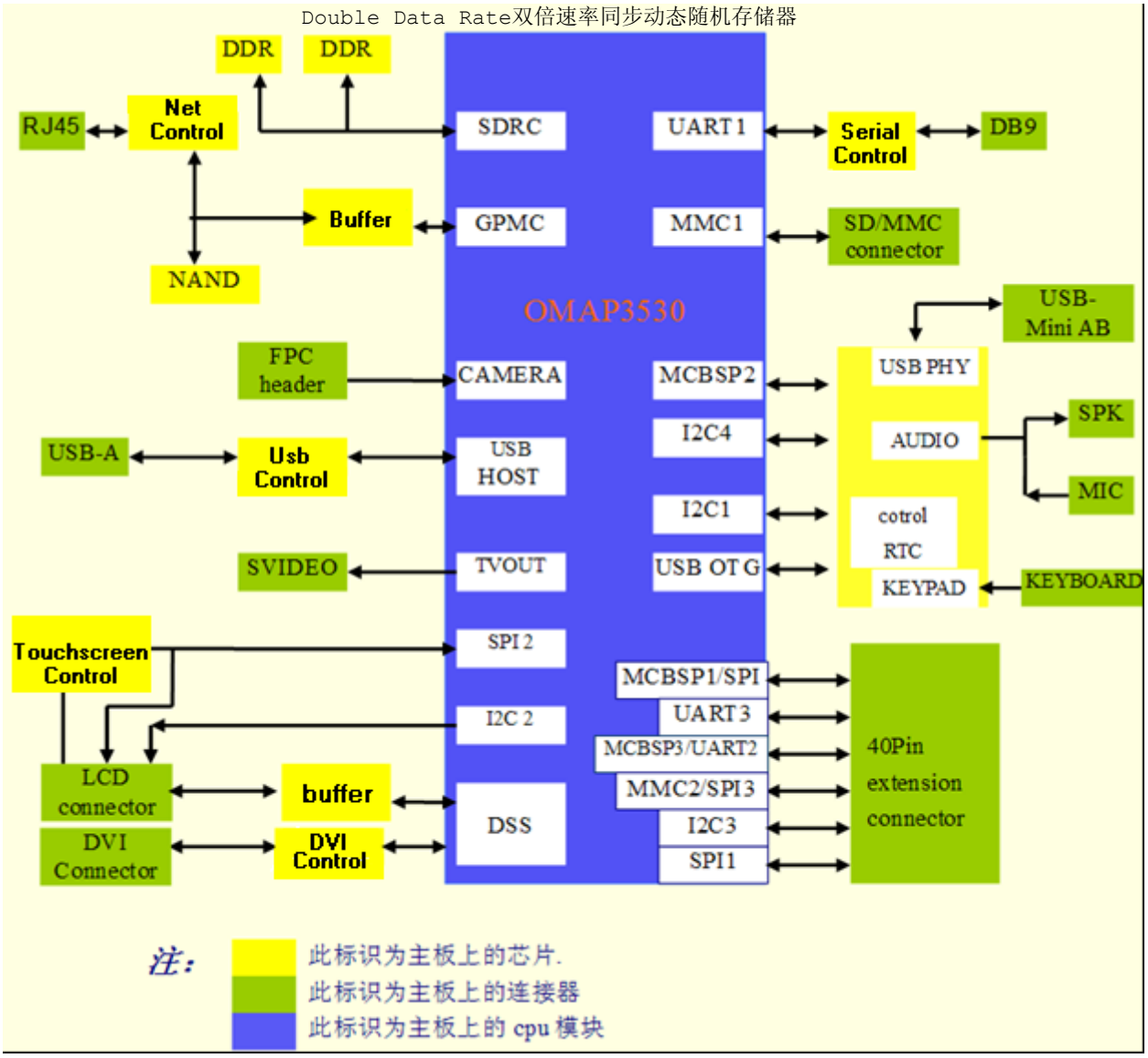


图2.1 结构框图

2.2 特性介绍

2.2.1 硬件资源

处理器

- OMAP3530 处理器（完全兼容 OMAP3503 处理器接口）
- 600-MHz ARM Cortex™-A8 Core
- 430-MHz TMS320C64x+™ DSP Core
- ARM 内部集成 16kB I-Cache, 16kB D-Cache, 256kB L2 存储器
- 片上存储 64kB SRAM, 112kB ROM

存储器

- 128MByte 32 位 DDR SDRAM, 166MHz
- 128MByte 16 位 NAND Flash

音频/视频接口

- 一个 4 线 S-VIDEO 接口
- 一个 HDMI 接口（数字信号视频传输的高清晰度多媒体接口）
- 一个音频输入接口
- 一个双声道音频输出接口

液晶触摸屏

- 分辨率：480 (W) x 272 (H) dots
- RGB：391680 colors
- 亮度：典型值 350 cd/m2 (最小 300 cd/m2)
- 4 线触摸屏

传输接口

- 串口：
  - 1 x 3 线串行接口，RS232 电平
  - 1 x 5 线串行接口，TTL 电平
- USB 接口：
  - 1 x USB2.0 OTG, High-speed, 480Mbps
  - 1 x USB2.0 HOST, High-speed, 480Mbps
- SD/MMC 接口：
  - 1 路 SD/MMC 接口，支持 3.3V 及 1.8V 逻辑电压
  - 1 路 SD/MMC 接口，支持 1.8V 逻辑电压
- 网络接口：10/100Mbps，RJ45 connector
- 1 路 McSPI 接口（多通道 SPI 接口）
- 1 路 McBSP 接口（多功能串行接口）
- 1 路 I2C 接口
- 1 路 HDQ 接口（单总线接口）

输入接口

- 1 个 CAMERA 接口（可外接 CCD 和 CMOS 的摄像头）
- 6 X 6 键盘接口
- 1 个 14 针标准 JTAG 接口
- 1 个启动引导按键
- 1 个 Reset 按键

- 1 个用户按键
- 1 个 ON/OFF 按键

LED 指示灯

- 1 个 3.3V 电源指示灯(LED33)
- 1 个 5V 电源指示灯(LED50)
- 1 个用户自定义灯(LEDB)
- 3 个系统灯，可以自定义(LED1、LED2、LED3)

2.2.2 电气特性

主板尺寸：110 mm x 95 mm
输入电压：+5V
功　　耗：0.5A @ 5V
操作温度：0 ℃ ~ 70 ℃
操作湿度：20% ~ 90%

2.3 硬件接口图

DevKit8000 评估套件的硬件接口图如图 2.2 所示。

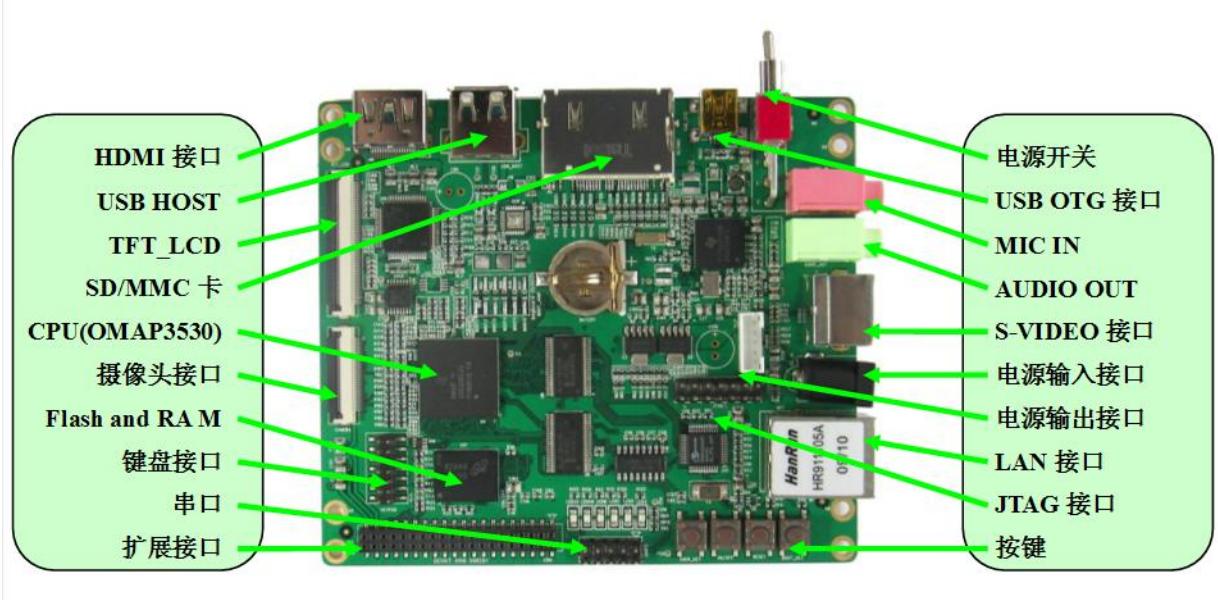


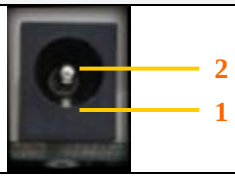
图2.2 正面接口图

第三章 硬件特性介绍

3.1 电源输入接口

功能描述：为 DevKit8000 提供标准 5V 电压，功率 2W。
接口描述：参考表 3-1。

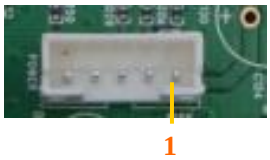
表 3-1 电源输入接口

引脚	信号定义	功能描述	引脚描述
1	GND	Power input (+5V)	
2	+5V	Power supply (+5V) 2A (Type)	

3.2 电源输出接口

功能描述：预留给外设提供标准的 5V，4.2V，3.3V 电源输出。
接口描述：参考表 3-2。

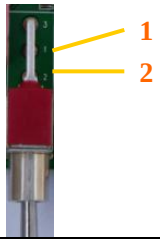
表 3-2 电源输出接口

引脚	信号定义	功能描述	引脚描述
1	VDD50	5V output	
2	VDD42	4.2V output	
3	VDD33	3.3V output	
4	ADCIN	ADC input	
5	GND	GND	

3.3 电源开关

功能描述：输入 5V 电源开关。
接口描述：参考表 3-3。

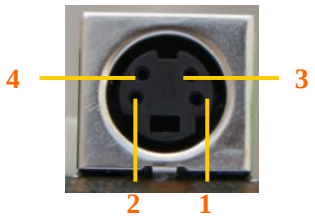
表 3-3 电源开关

引脚	信号定义	功能描述	引脚描述
1	DC IN	VDD Input	
2	VDD50	+5V	
3	NC	NC	

3.4 S-VIDEO 接口

功能描述：业界标准的 4 芯(不含音效)S-VIDEO 接口。
接口描述：参考表 3-4。

表 3-4 S-VIDEO 接口

引脚	信号定义	功能描述	引脚描述
1	GND	GND	
2	GND	GND	
3	OUTPUT1	VIDEO Y	
4	OUTPUT2	VIDEO C	

3.5 HDMI 接口

功能描述：标准的 HDMI 接口。
接口描述：参考表 3-5。

表 3-5 HDMI 接口

引脚	信号定义	功能描述	引脚描述
1	DAT2+	TMDS data 2+	
2	DAT2_S	TMDS data 2 shield	
3	DAT2-	TMDS data 2-	
4	DAT1+	TMDS data 1+	
5	DAT1_S	TMDS data 1 shield	
6	DAT1-	TMDS data 1-	
7	DAT0+	TMDS data 0+	
8	DAT0_S	TMDS data 0 shield	
9	DAT0-	TMDS data 0-	
10	CLK+	TMDS data clock+	
11	CLK_S	TMDS data clock shield	
12	CLK-	TMDS data clock-	
13	CEC	Consumer Electronics Control	
14	NC	NC	
15	SCL	IIC master serial clock	
16	SDA	IIC serial bidirectional data	
17	GND	GND	
18	5V	5V	
19	HPLG	Hot plug and play detect	

3.6 TFT_LCD 接口

功能描述：TFT_LCD 接口。
接口描述：参考表 3-6。

表 3-6 TFT_LCD 接口

引脚	信号定义	功能描述	引脚描述
1	DSS_D0	LCD Pixel data bit 0	
2	DSS_D1	LCD Pixel data bit 1	
3	DSS_D2	LCD Pixel data bit 2	
4	DSS_D3	LCD Pixel data bit 3	
5	DSS_D4	LCD Pixel data bit 4	
6	DSS_D5	LCD Pixel data bit 5	
7	DSS_D6	LCD Pixel data bit 6	
8	DSS_D7	LCD Pixel data bit 7	
9	GND	GND	
10	DSS_D8	LCD Pixel data bit 8	
11	DSS_D9	LCD Pixel data bit 9	
12	DSS_D10	LCD Pixel data bit 10	
13	DSS_D11	LCD Pixel data bit 11	
14	DSS_D12	LCD Pixel data bit 12	
15	DSS_D13	LCD Pixel data bit 13	
16	DSS_D 14	LCD Pixel data bit 14	
17	DSS_D15	LCD Pixel data bit 15	
18	GND	GND	
19	DSS_D16	LCD Pixel data bit 16	
20	DSS_D17	LCD Pixel data bit 17	
21	DSS_D18	LCD Pixel data bit 18	
22	DSS_D19	LCD Pixel data bit 19	
23	DSS_D20	LCD Pixel data bit 20	
24	DSS_D21	LCD Pixel data bit 21	
25	DSS_D22	LCD Pixel data bit 22	
26	DSS_D23	LCD Pixel data bit 23	
27	GND	GND	
28	DEN	AC bias control (STN) or pixel data enable (TFT)	
29	HSYNC	LCD Horizontal Synchronization	
30	VSYNC	LCD Vertical Synchronization	
31	GND	GND	

32	CLK	LCD Pixel Clock	
33	GND	GND	
34	X+	X+ Position Input	
35	X-	X- Position Input	
36	Y+	Y+ Position Input	
37	Y-	Y- Position Input	
38	SPI_CLK	SPI clock	
39	SPI_MOSI	Slave data in, master data out	
40	SPI_MISO	Slave data out, master data in	
41	SPI_CS	SPI enable	
42	IIC_CLK	IIC master serial clock	
43	IIC_SDA	IIC serial bidirectional data	
44	GND	GND	
45	VDD18	1.8V	
46	VDD33	3.3V	
47	VDD50	5V	
48	VDD50	5V	
49	RESET	Reset	
50	PWREN	Power on enable	

请不要在板子上电的情况下将 LCD FPC 排线拔出！

3.7 AUDIO OUT 接口

功能描述：标准的 Audio out 接口。
接口描述：参考表 3-7。

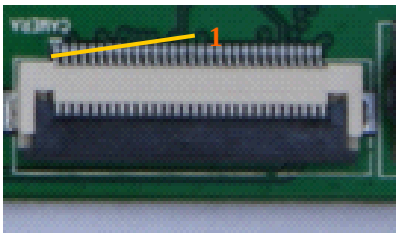
表 3-7 AUDIO OUT 接口

引脚	信号定义	功能描述	引脚描述
1	GND	GND	
2	NC	NC	
3	Right	Right output	
4	NC	NC	
5	Left	Left output	

3.8 摄像头接口

功能描述：Camera image sensor 接口。
接口描述：参考表 3-8。

表 3-8 摄像头接口

引脚	信号定义	功能描述	引脚描述
1	GND	GND	
2	D0	Digital image data bit 0	
3	D1	Digital image data bit 1	
4	D2	Digital image data bit 2	
5	D3	Digital image data bit 3	
6	D4	Digital image data bit 4	
7	D5	Digital image data bit 5	
8	D6	Digital image data bit 6	
9	D7	Digital image data bit 7	
10	D8	Digital image data bit 8	
11	D9	Digital image data bit 9	
12	D10	Digital image data bit 10	
13	D11	Digital image data bit 11	
14	GND	GND	
15	PCLK	Pixel clock	
16	GND	GND	
17	HS	Horizontal synchronization	
18	VDD50	5V	
19	VS	Vertical synchronization	
20	VDD33	3.3V	
21	XCLKA	Clock output a	
22	XCLKB	Clock output b	
23	GND	GND	
24	FLD	Field identification	
25	WEN	Write Enable	
26	STROBE	Flash strobe control signal	
27	SDA	IIC master serial clock	
28	SCL	IIC serial bidirectional data	
29	GND	GND	
30	VDD18	1.8V	

请不要在板子上电的情况下将 Camera FPC 排线拔出！

3.9 MIC IN 接口

功能描述：标准的 MIC IN 接口。
接口描述：参考表 3-9。

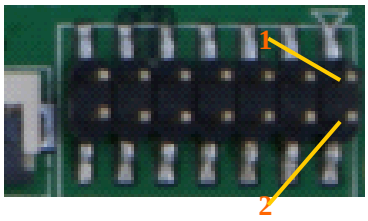
表 3-9 MIC IN 接口

引脚	信号定义	功能描述	引脚描述
1	GND	GND	
2	NC	NC	
3	MIC MAIN P	Right input	
4	NC	NC	
5	MIC MAIN N	Left input	

3.10 键盘接口

功能描述：6X6 矩阵键盘接口排针。
接口描述：参考表 3-10。

表 3-10 键盘接口

引脚	信号定义	功能描述	引脚描述
1	KC0	Keypad matrix column 0 output	
2	KR0	Keypad matrix row 0 input	
3	KC1	Keypad matrix column 1 output	
4	KR1	Keypad matrix row 1 input	
5	KC2	Keypad matrix column 2 output	
6	KR2	Keypad matrix row 2 input	
7	KC3	Keypad matrix column 3 output	
8	KR3	Keypad matrix row 3 input	
9	KC4	Keypad matrix column 4 output	
10	KR4	Keypad matrix row 4 input	
11	KC5	Keypad matrix column 5 output	
12	KR5	Keypad matrix row 5 input	
13	VDD18	1.8V	
14	GND	GND	

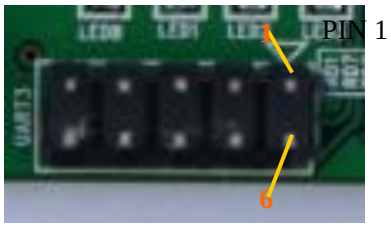
3.11 串口

功能描述：三线通用串行接口排针。

- Signal input/output level: RS232C
- Maximum data rate: 115.2kbps
- Flow Control: None
- Connector: 10-pin (5×2)2.54mm pitch connector

接口描述：参考表 3-11。

表 3-11 串口

引脚	信号定义	功能描述	引脚描述
1	NC	NC	
2	TXD	Transit data	
3	RXD	Receive data	
4	NC	NC	
5	GND	GND	
6	NC	NC	
7	NC	NC	
8	NC	NC	
9	NC	NC	

3.12 LAN 接口

功能描述：为 DEVKIT8000 提供的 10/100M 网络接口。其中 DM9000 内置 10/100M Ethernet 模块。它兼容 IEEE 802.3 标准协议。网线接口为标准的 RJ45，并且带有连接指示灯和传输指示灯。

DEVKIT8000 可通过直通网线连接 RJ1 到网络 hub 上，也可用交叉网线与电脑直接相连。

接口描述：参考表 3-12。

表 3-12 LAN 接口

引脚	信号定义	功能描述	引脚描述
1	TX+	TX+ output	
2	TX-	TX- output	
3	RX+	RX+ input	
4	VDD25	2.5V Power for TX/RX	
5	VDD25	2.5V Power for TX/RX	
6	RX-	RX- input	
7	NC	NC	
8	NC	NC	
9	VDD	3.3V Power for LED	
10	LED1	Speed LED	
11	LED2	Link LED	
12	VDD	3.3V Power for LED	

3.13 USB OTG 接口

功能描述：Mini 的 USB A/B 接口。 直接连接 TPS65930 芯片上的 USB OTG 端口。

Data Transfer Mode: USB2.0 High Speed (480Mbps)

Power Supply: Voltage: +5V

接口描述：参考表 3-13。

表 3-13 USB OTG 接口

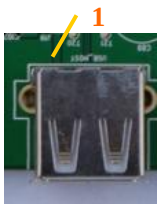
引脚	信号定义	功能描述	引脚描述
1	VBUS	+5V	

2	DN	USB Data-	
3	DP	USB Data+	
4	ID	USB ID	
5	GND	GND	

3.14 USB HOST 接口

功能描述：标准的 USB Host 接口。
Data Transfer Mode: USB2.0 High Speed (480Mbps)
Power Supply: Voltage: +5V
接口描述：参考表 3-14。

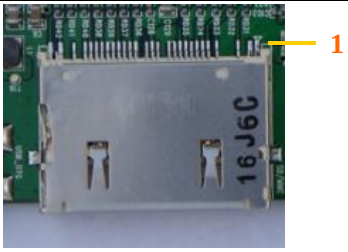
表 3-14 USB HOST 接口

引脚	信号定义	功能描述	引脚描述
1	VBUS	+5V	
2	DN	USB Data-	
3	DP	USB Data+	
4	GND	GND	

3.15 SD/MMC 卡接口

功能描述：标准的五合一 SD/MMC 卡接口，采用插入、保护自动检测设计。支持 4 位/8 位 SD 卡，支持 1.8V/3.3V 电压逻辑。
接口描述：参考表 3-15。

表 3-15 SD/MMC 卡接口

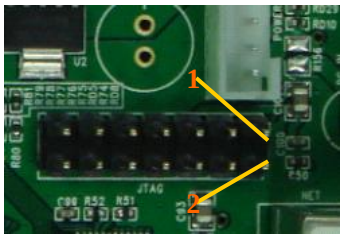
引脚	信号定义	功能描述	引脚描述
1	MINISD_CD1	Mini SD Card detect 1	
2	MINISD_CD2	Mini SD Card detect 2	
3	DAT2	MMC card data 2	
4	DAT3	MMC card data 3	
5	DAT4	MMC card data 4	
6	MINISD_DAT2	Mini SD card data 2	
7	CMD	SD/MMC Command Signal	
8	MINISD_DAT3	Mini SD card data 3	
9	DAT5	MMC card data 5	
10	MINISD_CMD	Mini SD card command	
11	VSS	GND	
12	MINISD_VSS	GND	
13	NC	NC	
14	VDD	VDD	

15	NC	NC	
16	MINISD_VDD	VDD	
17	CLK	MMC card clock	
18	MINISD_CLK	Mini SD card clock	
19	DAT6	MMC card data 6	
20	MINISD_VSS	GND	
21	VSS	GND	
22	MINISD_DAT0	Mini SD card data 0	
23	DAT7	MMC card data 7	
24	MINISD_DAT1	Mini SD card data 1	
25	DAT0	MMC card data 0	
26	DAT1	MMC card data 1	
27	SD_CD	SD Card detect	
28	SD_WP	SD write protect	
29	GND	GND	
30	GND	GND	

3.16 JTAG 接口

功能描述：常见的 JTAG 接口排针。
接口描述：参考表 3-16。

表 3-16 JTAG 接口

引脚	信号定义	功能描述	引脚描述
1	TMS	Test mode select	
2	NTRST	Test system reset	
3	TDI	Test data input	
4	GND	GND	
5	VIO	1.8V	
6	NC	NC	
7	TDO	Test data output	
8	GND	GND	
9	RTCK	Receive test clock	
10	GND	GND	
11	TCK	Test clock	
12	GND	GND	
13	EMU0	Test emulation 0	
14	EMU1	Test emulation 1	

仿真

3.17 扩展接口

功能描述：自定义的各种扩展接口，包括 BSP1，BSP3，UART1，I2C3，SPI1，MMC2 接口，

电平逻辑为 1.8V。

接口描述：参考表 3-17。

表 3-17 扩展接口

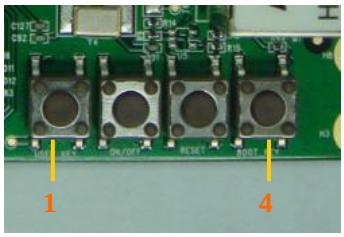
引脚	信号定义	功能描述	引脚描述
1	GND	GND	
2	BSP1_DX	Transmitted serial data 1	
3	BSP1_DR	Received serial data 1	
4	BSP1_CLKR	Received clock 1	
5	BSP1_FSX	Transmit frame synchronization 1	
6	BSP1_CLKX	Transmit clock 1	
7	BSP1_CLKS	External clock input 1	
8	BSP1_FSR	Receive frame synchronization 1	
9	UART1_CTS	UART1 clear to send	
10	UART1_RTS	UART1 request to send	
11	UART1_RX	UART1 receive data	
12	UART1_TX	UART1 transmit data	
13	GND	GND	
14	MMC2_CLK	MMC2 card clock	
15	MMC2_CMD	MMC2 Command Signal	
16	MMC2_D0	MMC2 card data 0	
17	MMC2_D1	MMC2 card data 1	
18	MMC2_D2	MMC2 card data 2	
19	MMC2_D3	MMC2 card data 3	
20	MMC2_D4	MMC2 card data 4	
21	MMC2_D5	MMC2 card data 5	
22	MMC2_D6	MMC2 card data 6	
23	MMC2_D7	MMC2 card data 7	
24	BSP3_DX	Transmitted serial data 3	
25	BSP3_DR	Received serial data 3	
26	BSP3_CLKX	Transmit clock 3	
27	BSP3_FSX	Transmit frame synchronization 3	
28	GND	GND	
29	IIC3_SCL	IIC3 master serial clock	
30	IIC3_SDA	IIC3 serial bidirectional data	
31	SPI1_SIMO	Slave data in, master data out	
32	SPI1_SOMI	Slave data out, master data in	
33	SPI1_CLK	SPI1 clock	
34	SPI1_CS0	SPI enable 0	
35	SPI1_CS3	SPI enable 3	
36	HDQ_SIO	Bidirectional HDQ	
37	VDD33	3.3V	
38	VDD18	1.8V	

39	VDD50	5V	
40	GND	GND	

3.18 按键

功能描述：提供各种按键功能。
接口描述：参考表 3-18。

表 3-18 按键

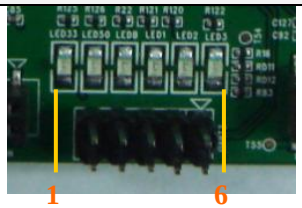
引脚	信号定义	功能描述	引脚描述
1	USER-KEY	User-defined key	
2	ON/OFF	System ON/OFF key	
3	RESET	System reset key	
4	BOOT-KEY	System boot configuration	

4

3.19 LED 灯

功能描述：提供各种按键功能。
接口描述：参考表 3-19。

表 3-19 LED灯

标号	信号定义	功能描述	描述
1	LED33	3.3V电源指示灯	
2	LED50	4.2V电源指示灯	
3	LEDB	用户自定义灯	
4	LED1	系统LED灯	
5	LED2	系统LED灯	
6	LED3	系统LED灯	

第三部分 Linux 软件系统

第四章 Linux 系统概述

本章主要概述DevKit8000的软件系统，包括预装软件介绍，DevKit8000 BSP包规格说明以及DevKit8000出货光盘中提供各种资源的规格说明。

DevKit8000软件系统包括：预编译的映像及系统各部分源码，交叉编译工具，开发辅助工具等，均可从DevKit8000评估套件发行光盘中找到。

DevKit8000评估套件预装软件包括：

- x-loader----------（x-load.bin.ift_for_NAND)
- u-boot----------(flash-uboot.bin)
- 2.6 kernel----------(uImage)
- rootfs----------(ubi.img)

另外，发行光盘中，还提供了以下程序和软件：

- 烧写用映像文件
- 交叉编译工具
- 系统各部分源码
- 用户测试程序及开发示例
- 用户在使用DevKit8000时可能会用到的一些工具

4.1 预装软件

DevKit8000出厂的时候，软件映像已经固化在FLASH上。完整的系统由x-loader、u-boot、kernel和rootfs四部分组成，系统结构如图4.1所示：

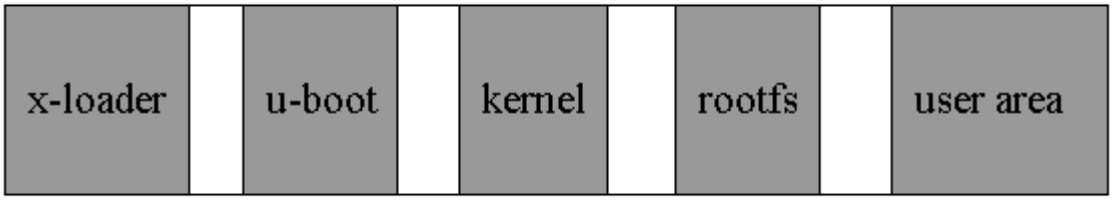


图4.1 System compose map

系统各组成部分特性及作用介绍如下：

1、x-loader是一级引导程序，系统上电后由CPU内部ROM自动拷贝到内部RAM并执行。主要作用为初始化CPU，拷贝u-boot到内存中，然后把控制权交给u-boot；

2、u-boot是二级引导程序，主要用于和用户进行交互，提供映像更新、引导内核等功能；

3、kernel使用最新2.6.x内核，根据DevKit8000进行定制；
4、rootfs采用开源文件系统，体积小，功能强大。

4.2 BSP 特性

DevKit8000 BSP包主要用于定制、生成运行于DevKit8000硬件平台上的Linux操作系统，用户基于该开发包进行二次开发。DevKit8000出厂光盘中所提供的BSP包中包括表4-1所示的内容。

表4-1 BSP规格

名称		备注
BIOS	x-loader	NAND / ONENAND
		MMC/SD
		FAT
	u-boot	NAND / ONENAND
		MMC/SD
		FAT
		NET
Kernel	Linux-2.6.x	支持ROM/CRAM/EXT2/EXT3/FAT/NFS/ JFFS2/UBIFS等多种文件系统
Device Driver	serial	串口驱动
	rtc	硬件时钟驱动
	net	10/100M以太网卡DM9000驱动
	flash	nand flash驱动(支持nand boot)
	lcd	TFT LCD 驱动
	touch screen	触摸屏控制器ads7846驱动
	mmc/sd	mmc/sd控制器驱动
	usb otg	usb otg 2.0驱动(可配置为主/从设备)
	usb ehci	usb ehci驱动
	dvi	支持dvi-d信号输出
	s-video	支持s-video信号输出
	audio	声卡驱动(支持录/放音)
	camera	摄像头驱动(支持CCD摄像头)
	keypad	6x6矩阵键盘驱动
	led	用户led灯驱动
GUI	Angstrom	针对于嵌入式设备的桌面环境发行版
	Android	google android系统

第五章 Linux 系统快速操作

5.1 系统启动方法

注：在系统操作的过程中，需用到 PC 机超级终端时，超级终端的配置如下所示：

波特率：115200
数据位： 8
奇偶检验：无
停止位： 1
流控制：无

5.1.1 NAND Flash 启动

主板默认配置为先从 NAND 启动，如果 NAND 无系统映像将会从 SD 卡启动。出厂时 nand flash 下固化有代码，用户只需要连接好串口，接通电源，系统随即从 nand flash 启动。进入 linux 系统后，默认的账户是 root，没有密码。

注：NAND Flash 中系统映像的更新方式请参考【7.2 NAND Flash 系统映像更新】

5.1.2 SD 卡启动

如果用户需要切换到 SD 卡启动，只需要按住主板上的 BOOT_KEY 按键（按键位置请参考【3.18 按键】），上电启动，系统随即改为从 MMC/SD 启动。进入 linux 系统后，默认的账户是 root，没有密码。

注：SD 卡中系统映像的更新方式请参考【7.1 SD 卡系统映像更新】

5.2 显示方式选择

系统支持多种显示输出模式，所提供的映像默认使用 4.3"LCD 输出模式，用户可通过配置启动参数的方法选择不同的显示输出模式。

如何进入 U-boot 命令行：

```
Texas Instruments X-Loader 1.41
Starting OS Bootloader...


U-Boot 1.3.3-svn323 (Dec  7 2009 - 10:35:16)

OMAP3530-GP rev 2, CPU-OPP2 L3-165MHz
```

Devkit8000 Board + LPDDR/NAND
DRAM: 128 MB
NAND: 128 MiB
Display-bmp: 480 x 272 with 256 colors
In: serial
Out: serial
Err: serial
Hit any key to stop autoboot: 3 （在这时候在键盘上按任何键）
OMAP3 DevKit8000 # **(现已进入 u-boot 命令行)**

注意：

1、 请不要在板子带电的情况下将 LCD 排线移除。

2、 这里的显示方式是针对 linux-2.6.28 内核来修改的，而 Demo 是不需要进行下面的设置的。

5.2.1 使用 4.3”LCD 显示

u-boot 下修改启动参数如下：

1、对于 NAND Flash 启动

OMAP3 DevKit8000 # **setenv bootargs console=ttyS2,115200n8 ubi.mtd=4 root=ubi0:rootfs rootfstype=ubifs video=omapfb:mode:4.3inch_LCD**

OMAP3 DevKit8000 # **setenv bootcmd nand read.i 80300000 280000 300000\;bootm 80300000**

OMAP3 DevKit8000 # **saveenv**

2、对于SD卡启动

OMAP3 DevKit8000 # **setenv bootargs console=ttyS2,115200n8 root=/dev/ram initrd=0x81600000,40M video=omapfb:mode:4.3inch_LCD**

OMAP3 DevKit8000 # **setenv bootcmd 'mmcinit;fatload mmc 0 80300000 uImage;fatload mmc 0 81600000 ramdisk.gz;bootm 80300000'**

OMAP3 DevKit8000 # **saveenv**

5.2.2 使用 5.6”LCD 显示

u-boot 下修改启动参数如下：

1、对于 NAND Flash 启动

OMAP3 DevKit8000 # **setenv bootargs console=ttyS2,115200n8 ubi.mtd=4 root=ubi0:rootfs rootfstype=ubifs video=omapfb:mode:5.6inch_LCD**

OMAP3 DevKit8000 # **setenv bootcmd nand read.i 80300000 280000 300000\;bootm 80300000**

OMAP3 DevKit8000 # **saveenv**

2、对于SD卡启动

OMAP3 DevKit8000 # **setenv bootargs console=ttyS2,115200n8 root=/dev/ram initrd=0x81600000,40M video=omapfb:mode:5.6inch_LCD**

OMAP3 DevKit8000 # **setenv bootcmd 'mmcinit;fatload mmc 0 80300000 uImage;fatload mmc 0 81600000 ramdisk.gz;bootm 80300000'**

OMAP3 DevKit8000 # **saveenv**

5.2.3 使用 7”LCD 显示

u-boot 下修改启动参数如下：

1、对于 NAND Flash 启动

OMAP3 DevKit8000 # **setenv bootargs console=ttyS2,115200n8 ubi.mtd=4 root=ubi0:rootfs rootfstype=ubifs video=omapfb:mode:7inch_LCD**

OMAP3 DevKit8000 # **setenv bootcmd nand read.i 80300000 280000 300000\;bootm 80300000**

OMAP3 DevKit8000 # **saveenv**

2、对于SD卡启动

OMAP3 DevKit8000 # **setenv bootargs console=ttyS2,115200n8 root=/dev/ram initrd=0x81600000,40M video=omapfb:mode:7inch_LCD**

OMAP3 DevKit8000 #**setenv bootcmd 'mmcinit;fatload mmc 0 80300000 uImage;fatload mmc 0 81600000 ramdisk.gz;bootm 80300000'**

OMAP3 DevKit8000 # **saveenv**

5.2.3 使用 10.4”LCD 显示

u-boot 下修改启动参数如下：

1、对于 NAND Flash 启动

OMAP3 DevKit8000 # **setenv bootargs console=ttyS2,115200n8 ubi.mtd=4 root=ubi0:rootfs rootfstype=ubifs video=omapfb:mode:10.4inch_LCD**

OMAP3 DevKit8000 # **setenv bootcmd nand read.i 80300000 280000 300000\;bootm 80300000**

OMAP3 DevKit8000 # **saveenv**

2、对于SD卡启动

OMAP3 DevKit8000 # **setenv bootargs console=ttyS2,115200n8 root=/dev/ram initrd=0x81600000,40M video=omapfb:mode:10.4inch_LCD**

OMAP3 DevKit8000 #**setenv bootcmd 'mmcinit;fatload mmc 0 80300000 uImage;fatload mmc 0 81600000 ramdisk.gz;bootm 80300000'**

OMAP3 DevKit8000 # **saveenv**

5.2.4 使用 DVI 显示

u-boot 下修改启动参数如下：

1、对于 NAND Flash 启动

OMAP3 DevKit8000 # **setenv bootargs console=ttyS2,115200n8 ubi.mtd=4 root=ubi0:rootfs rootfstype=ubifs video=omapfb:mode:720p60**

OMAP3 DevKit8000 # **setenv bootcmd nand read.i 80300000 280000 300000\;bootm 80300000**

OMAP3 DevKit8000 # **saveenv**

2、对于SD卡启动

OMAP3 DevKit8000 # **setenv bootargs console=ttyS2,115200n8 root=/dev/ram initrd=0x81600000,40M video=omapfb:mode:720p60**

OMAP3 DevKit8000 #**setenv bootcmd 'mmcinit;fatload mmc 0 80300000 uImage;fatload mmc 0 81600000 ramdisk.gz;bootm 80300000'**

OMAP3 DevKit8000 # **saveenv**

5.2.5 使用 VGA 显示

u-boot 下修改启动参数如下：

1、对于 NAND Flash 启动

OMAP3 DevKit8000 # **setenv bootargs console=ttyS2,115200n8 ubi.mtd=4 root=ubi0:rootfs rootfstype=ubifs video=omapfb:mode:VGA**

OMAP3 DevKit8000 # **setenv bootcmd nand read.i 80300000 280000 300000\;bootm 80300000**

OMAP3 DevKit8000 # **saveenv**

2、对于SD卡启动

OMAP3 DevKit8000 # **setenv bootargs console=ttyS2,115200n8 root=/dev/ram initrd=0x81600000,40M video=omapfb:mode:VGA**

OMAP3 DevKit8000 #**setenv bootcmd 'mmcinit;fatload mmc 0 80300000 uImage;fatload mmc 0 81600000 ramdisk.gz;bootm 80300000'**

OMAP3 DevKit8000 # **saveenv**

5.3 测试程序使用

5.3.1 LED 测试

主板上的 LEDB，LED1，LED2，LED3 为用户 LED 灯（具体位置请参考【3.19 LED 灯】）。其中 LED1 已用作系统心跳的指示，LED2 已用作 SD 卡数据传输指示。

以下示范如何操作用户 led 灯 LED3：

以下操作在超级终端中进行：

1、终端中输入如下命令点亮 LED3

root@DevKit8000:~# **echo -n 1 >/sys/class/leds/led3/brightness**

2、终端中输入如下命令熄灭 LED3

root@DevKit8000:~# **echo -n 0 >/sys/class/leds/led3/brightness**

LED3 会随着用户的操作进行亮灭。

5.3.2 KEYPAD 测试

开发板扩展了 6x6 矩阵键盘接口，使用 evtest 工具可测试矩阵键盘工作是否正常：

将矩阵键盘插入开发板接口，终端中敲入以下命令进行测试：

root@DevKit8000:~# **evtest /dev/input/event0**

按下矩阵键盘的任意键，例如‘1’，终端上会出现类似提示：

Event: time 946684837.310027, type 1 (Key), code 2 (1), value 1
Event: time 946684837.402160, type 1 (Key), code 2 (1), value 0

其中，“type 1 (Key), code 2 (1), value 1“，表明发生按键事件，键值为 2(对应全键盘的‘1’键，状态为按下(‘0’表示按键提起))，

注意: 按 CONTROL+C 退出测试。

5.3.3 触摸屏测试

1、输入以下指令执行触摸屏校准程序：

```
root@DevKit8000:~#  ts_calibrate
```

按照屏幕上提示，点击“+”图标 5 次完成校准。

2、校准完成后，输入以下指令进行触摸屏测试：

```
root@DevKit8000:~#  ts_test
```

按照屏幕提示，可选择画点、画线测试。

注意: 按 CONTROL+C 退出测试。

5.3.4 RTC 测试

开发板带硬件时钟，用于保存并恢复系统时间，可参考如下方法进行测试：

1、设置系统时间为 2008 年 8 月 8 日晚上 8 时正

```
root@DevKit8000:~#  date 080820002008
Fri Aug  8 20:00:00 UTC 2008
```

2、把系统时钟写入 RTC

```
root@DevKit8000:~#  hwclock –w
```

3、读取 RTC

```
root@DevKit8000:~#  hwclock
Fri Aug  8 20:00:21 2008  0.000000 seconds
```

可以看到，硬件时钟 RTC 被设置成 2008 年 8 月 8 日，系统时钟被保存到硬件时钟里。

4、重启系统，输入以下命令恢复系统时钟

```
root@DevKit8000:~#  hwclock –s
root@DevKit8000:~#  date
Fri Aug  8 20:01:45 UTC 2008
```

可以看到，系统时间被恢复为硬件时间。

注意: 开发板本身未带电池，用户需自行购买。

5.3.5 MMC/SD 卡测试

插入 MMC/SD 卡，系统会自动检测并挂载 MMC/SD 卡到/media 目录下，SD 卡默认名称是 mmcblk0p1.

```
root@DevKit8000:~#  cd /media/
root@DevKit8000:/media#  ls
card        hdd         mmcblk0p1   ram         union
cf          mmc1        net         realroot
root@DevKit8000:/media#  ls mmcblk0p1/
flash-uboot.bin          u-boot.bin              x-load.bin.ift_for_NAND
mlo                      uImage
ramdisk.gz               ubi.img
root@DevKit8000:/media# umount  /media/mmcblk0p1/
```

5.3.6 USB OTG 测试

1、USB OTG 作为 DEVICE 使用：

1）系统起来后，使用 USB mini B to USB A 转接线连接开发板与 pc 机，其中 USB mini B 接口连接开发板，USB A 接口连接 pc 机。

注意: Linux USB Ethernet/RNDIS Gadget 驱动的安装请参考附录一的说明。

2）在本地连接 2 右键-属性-双击 Internet 协议（TCP/IP）,配置虚拟网卡的 IP 地址，例如：

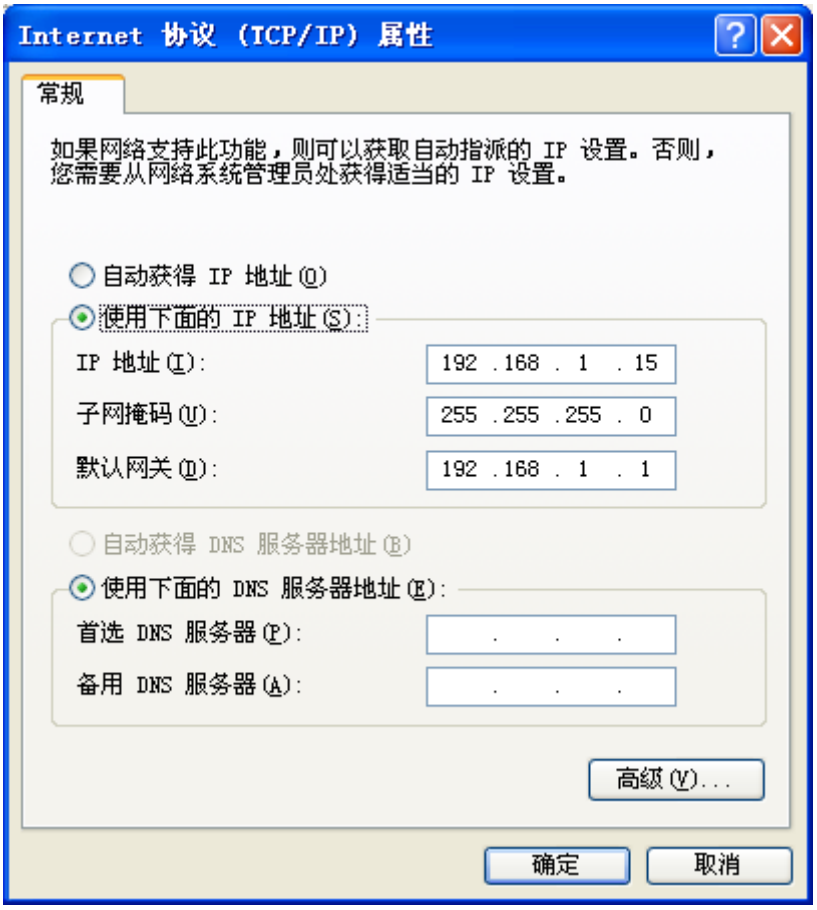


图 5.2 IP 设置

4）配置开发板的 IP 和 usb 虚拟网卡的 IP 地址在同一网段，在超级终端输入以下命令，例如：

```
root@DevKit8000:~#  ifconfig usb0 192.168.1.115
root@DevKit8000:~#  ifconfig
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:26 errors:0 dropped:0 overruns:0 frame:0
          TX packets:26 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:0
          RX bytes:2316 (2.2 KiB)  TX bytes:2316 (2.2 KiB)
```

```
usb0      Link encap:Ethernet  HWaddr 5E:C5:F6:D4:2B:91
          inet addr:192.168.1.115  Bcast:192.168.1.255  Mask:255.255.255.0
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:253 errors:0 dropped:0 overruns:0 frame:0
          TX packets:43 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:35277 (34.4 KiB)  TX bytes:10152 (9.9 KiB)
```

5）成功连接后，点击我的电脑-网上邻居-查看网络连接，PC 端会增加一个虚拟网卡，如图 5.1 所示：

图 5.1 本地连接 2 为虚拟网卡

6）在超级终端中使用 ping 命令测试开发板是否设置成功：

```
root@DevKit8000:~#  ping 192.168.1.15
PING 192.168.1.15 (192.168.1.15): 56 data bytes
64 bytes from 192.168.1.15: seq=0 ttl=128 time=0.885 ms
64 bytes from 192.168.1.15: seq=1 ttl=128 time=0.550 ms
```

按 CONTROL+C 退出测试

注意::*OTG 配置的网卡地址不能跟网口的地址的网段一致。*

2、USB OTG 作为 HOST 使用：

使用 USB mini A to USB A 转接线连接开发板与 USB 设备(例如 U 盘)，其中 USB mini A 接口连接开发板，USB A 接口连接 USB 设备。

系统起来的时候会检测并自动挂载 USB 设备，并挂载到/media 目录下：

```
root@DevKit8000:~#  cd /media/
root@DevKit8000:/media#  ls
card          hdd           mmcblk0p1   ram           sda1
cf            mmc1          net         realroot      union
root@DevKit8000:/media#  cd sda1
```

sda1 目录下有文件，则说明 USB 设备被正确读取。

注意: 某些 U 盘可能会被识别为 sda

5.3.7 AUDIO 测试

板上带音频输入、输出接口，支持录放音。文件系统内带 alsa-utils 音频播放、录制测试工具，用户可使用如下命令进行测试：

插上麦克风，即可进行录音测试。

1、录音测试：

```
root@DevKit8000:~# arecord -t wav -c 2 -r 44100 -f S16_LE -v k
Recording WAVE 'k' : Signed 16 bit Little Endian, Rate 44100 Hz, Stereo
Plug PCM: Hardware PCM card 0 'omap3evm' device 0 subdevice 0
Its setup is:
   stream          : CAPTURE
   access          : RW_INTERLEAVED
   format          : S16_LE
   subformat       : STD
   channels        : 2
   rate            : 44100
   exact rate      : 44100 (44100/1)
   msbits          : 16
   buffer_size     : 22052
   period_size     : 5513
   period_time     : 125011
   tstamp_mode     : NONE
   period_step     : 1
   avail_min       : 5513
   period_event : 0
   start_threshold     : 1
   stop_threshold      : 22052
   silence_threshold: 0
   silence_size : 0
   boundary        : 1445199872
   appl_ptr        : 0
   hw_ptr          : 0
```

*注意: 录音结束后，按*CONTROL+C 退出测试。

2、放音测试：

插上耳机，执行以下操作，即可听刚才的录音内容。

```
root@DevKit8000:~# aplay -t wav -c 2 -r 44100 -f S16_LE -v k
Playing WAVE 'k' : Signed 16 bit Little Endian, Rate 44100 Hz, Stereo
Plug PCM: Hardware PCM card 0 'omap3evm' device 0 subdevice 0
Its setup is:
   stream          : PLAYBACK
   access          : RW_INTERLEAVED
   format          : S16_LE
   subformat       : STD
   channels        : 2
   rate            : 44100
   exact rate      : 44100 (44100/1)
```

```
  msbits          : 16
  buffer_size    : 22052
  period_size    : 5513
  period_time    : 125011
  tstamp_mode    : NONE
  period_step    : 1
  avail_min      : 5513
  period_event : 0
  start_threshold    : 22052
  stop_threshold     : 22052
  silence_threshold: 0
  silence_size : 0
  boundary        : 1445199872
  appl_ptr        : 0
  hw_ptr          : 0
```

5.3.8 网络测试

板上带10/100M自适应网卡DM9000，用户可把开发板接入局域网，先接上网线，使用如下命令进行测试：

```
root@DevKit8000:~#  ifconfig eth0 192.192.192.200
eth0: link down
eth0: link up, 100Mbps, full-duplex, lpa 0x41E1

root@DevKit8000:~#  ping 192.192.192.90
PING 192.192.192.90 (192.192.192.90): 56 data bytes
64 bytes from 192.192.192.90: seq=0 ttl=128 time=1.007 ms
64 bytes from 192.192.192.90: seq=1 ttl=128 time=0.306 ms
64 bytes from 192.192.192.90: seq=2 ttl=128 time=0.397 ms
64 bytes from 192.192.192.90: seq=3 ttl=128 time=0.367 ms

--- 192.192.192.90 ping statistics ---
4 packets transmitted, 4 packets received, 0% packet loss
round-trip min/avg/max = 0.306/0.519/1.007 ms
```

注意: 开发板的网卡 ip 地址应与 PC 机在同一网段，例如 192.192.192.x，ping 后面的地址根据自己电脑的 IP 设定，如果 ping 返回，则说明网卡正确工作，按 CONTROL+C 退出测试。

5.3.9 CAMERA 测试

购买了DevKit8000配套CAMERA模块的话，连接好CAMERA模块以及CCD摄像头后，接上LCD屏幕，可执行以下命令进行测试：

```
root@DevKit8000:~# saMmapLoopback
```

```
tvp514x 2-005d: tvp5146m2 found at 0xba (OMAP I2C adapter)

Capture: Opened Channel
Capture: Current Input: COMPOSITE
Capture: Current standard: PAL
Capture: Capable of streaming
Capture: Number of requested buffers = 3
Capture: Init done successfully



Display: Opened Channel
Display: Capable of streaming
Display: Number of requested buffers = 3
Display: Init done successfully

Display: Stream on...
Capture: Stream on...
```

LCD 屏幕上可看到 CCD 摄像头采集的图像，摄像头驱动的位置为：drivers/media/video/tvp514x.c。

5.3.10 GPRS-8000 模块

若有购买本公司的 GPRS 模块，请您在以下路径下载模块资料：
http://www.timll.com/chinese/uploadFile/GPRS8000.rar

5.3.11 GPS-8000 模块

若有购买本公司的 GPS 模块，请您在以下路径下载模块资料：
http://www.timll.com/chinese/uploadFile/GPS8000.rar

5.4 Demo 系统演示

前提：以下操作是在电脑上完成的，用户需要自行安装 ubuntu 7.10 版本操作系统，需要用户自行准备；当 SD 卡在 ubuntu 下格式化好 FAT 和 EXT3 双分区后，**FAT 分区需要在 window 下重新格式化一次，否则可能会出现无法从 SD 卡启动的情。**

5.4.1 Angstrom(GPE)桌面发布版本演示

1、把 SD 卡按附录 2 格式化为两个分区，重新挂载 SD 卡，然后执行如下命令。

```
   cp /media/cdrom/linux/demo/angstrom/MLO /media/LABEL1
   cp /media/cdrom/linux/demo/angstrom/u-boot.bin /media/LABEL1
   cp /media/cdrom/linux/demo/angstrom/uImage /media/LABEL1
   rm -rf /media/LABEL2/*
   sudo tar jxvf
linux/demo/angstrom/Angstrom-DevKit8000-demo-image-glibc-ipk-2008.1-test-2
```

```
0080111-DevKit8000.rootfs.tar.bz2 -C /media/LABEL2
    sync
    umount /media/LABEL1
    umount /media/LABEL2
```

2、取出 SD 卡插到开发板，并选择从 SD 启动，即可进入 Angstrom 系统，默认为 DVI 显示输出。

> *注意:第一次进入桌面系统时，会对系统作大量配置，请耐心等待几分钟。以后启动即可快速进入桌面环境。*
>
> *详 细 情 况 请 参 考 http://www.angstrom-distribution.org/ 或 http://www.angstrom-distribution.org/demo/beagleboard/*

5.4.2 Google android 系统演示

1、把 SD 卡按附录 2 格式化为两个分区，重新挂载 SD 卡，然后执行如下命令，**并请注意，要更换名字，要将 u-boot.bin_4.3 改名为 u-boot.bin。**

客户使用 4.3”LCD 屏:

```
cp /media/cdrom/linux/demo/android/MLO /media/LABEL1
cp /media/cdrom/linux/demo/android/u-boot.bin_4.3 /media/LABEL1/u-boot.bin
cp /media/cdrom/linux/demo/android/uImage_4.3 /media/LABEL1/uImage
rm -rf /media/LABEL2/*
sudo tar jxvf linux/demo/ android/RFS.tar.bz2 -C /media/LABEL2
sync
umount /media/LABEL1
umount /media/LABEL2
```

客户使用 5.6”LCD 屏，则请执行系列指令：

```
cp /media/cdrom/linux/demo/android/MLO /media/LABEL1
cp /media/cdrom/linux/demo/android/u-boot.bin_5.6 /media/LABEL1/u-boot.bin
cp /media/cdrom/linux/demo/android/uImage_5.6 /media/LABEL1/uImage
rm -rf /media/LABEL2/*
sudo tar jxvf linux/demo/ android/RFS.tar.bz2 -C /media/LABEL2
sync
umount /media/LABEL1
umount /media/LABEL2
```

客户使用 7”LCD 屏，则请执行系列指令：

```
cp /media/cdrom/linux/demo/android/MLO /media/LABEL1
cp /media/cdrom/linux/demo/android/u-boot.bin_7 /media/LABEL1/u-boot.bin
cp /media/cdrom/linux/demo/android/uImage_7 /media/LABEL1/uImage
rm -rf /media/LABEL2/*
sudo tar jxvf linux/demo/ android/RFS.tar.bz2 -C /media/LABEL2
sync
umount /media/LABEL1
umount /media/LABEL2
```

2、将 SD 卡插到开发板，接上 LCD 屏和电源，并选择从 SD 启动开发板，即可进入 Android 系统。

注意：光盘上面没有 Android 的内核源码，不过可以在链接 1 上面下载，但是对于源码我们是不提供任何技术支持详细资料可参考链接 2

链接 1：http://www.embedinfo.com/english/download/linuxadroid.zip

链接 2：http://labs.embinux.org/index.php/Android_Porting_Guide_to_Beagle_Board

5.4.3 DVSDK 演示

1、把 SD 卡按附录 2 格式化为两个分区，重新挂载 SD 卡，然后执行如下命令，**并请注意，要更换名字，要将 uImage_4.3 改名为 uImage。**

客户使用 4.3”LCD 屏，则请执行系列指令：

```
cp /media/cdrom/linux/demo/dvsdk/MLO /media/LABEL1
cp /media/cdrom/linux/demo/dvsdk/u-boot.bin /media/LABEL1
cp /media/cdrom/linux/demo/dvsdk/uImage_4.3 /media/LABEL1/uImage
rm -rf /media/LABEL2/*
sudo tar jxvf linux/demo/dvsdk/DVSDK.tar.bz2 -C /media/LABEL2
sync
umount /media/LABEL1
umount /media/LABEL2
```

客户使用 5.6”LCD 屏，则请执行系列指令：

```
cp /media/cdrom/linux/demo/dvsdk/MLO /media/LABEL1
cp /media/cdrom/linux/demo/dvsdk/u-boot.bin /media/LABEL1
cp /media/cdrom/linux/demo/dvsdk/uImage_5.6 /media/LABEL1/uImage
rm -rf /media/LABEL2/*
sudo tar jxvf linux/demo/dvsdk/DVSDK.tar.bz2 -C /media/LABEL2
sync
umount /media/LABEL1
umount /media/LABEL2
```

客户使用 7”LCD 屏，则请执行系列指令：

```
cp /media/cdrom/linux/demo/dvsdk/MLO /media/LABEL1
cp /media/cdrom/linux/demo/dvsdk/u-boot.bin /media/LABEL1
cp /media/cdrom/linux/demo/dvsdk/uImage_7 /media/LABEL1/uImage
rm -rf /media/LABEL2/*
sudo tar jxvf linux/demo/dvsdk/DVSDK.tar.bz2 -C /media/LABEL2
sync
umount /media/LABEL1
umount /media/LABEL2
```

客户使用 10.4”LCD 屏，则请执行系列指令：

```
cp /media/cdrom/linux/demo/dvsdk/MLO /media/LABEL1
cp /media/cdrom/linux/demo/dvsdk/u-boot.bin /media/LABEL1
cp /media/cdrom/linux/demo/dvsdk/uImage_10.4 /media/LABEL1/uImage
rm -rf /media/LABEL2/*
sudo tar jxvf linux/demo/dvsdk/DVSDK.tar.bz2 -C /media/LABEL2
sync
umount /media/LABEL1
```

```
umount /media/LABEL2
```

2、取出 SD 卡插到开发板，并选择从 SD 启动开发板，即可进入 DVSDK 演示系统。

注意：对于 DVSDK 的详细资料请参考 TI 主页，对于 TI 主页的 DVSDK 使用需要 linux-2.6.29 内核，在 http://code.google.com/p/devkit8000/w/list 可以找到 linux-2.6.29 的补丁与用法。

第六章 Linux 系统开发

本章介绍如何利用DevKit8000 BSP包搭建运行于DevKit8000硬件平台上的Linux系统开发环境。具体内容包括交叉编译环境的搭建，系统映像的生成，并通过示例介绍linux内核的配置。

注意: 本文中使用的 Linux 发行版为 ubuntu 7.10，下文中简称为 ubuntu。

6.1 开发环境搭建

用户在使用DevKit8000进行开发前，必须先搭建好ARM Linux交叉开发环境。下面以ubuntu操作系统为例，介绍交叉开发环境的搭建，其它Linux系统的操作与ubuntu系统类似。

6.1.1 交叉编译环境的安装

插入光盘，ubuntu默认把光盘挂载到/media/cdrom目录下，交叉编译工具就存放在/media/cdrom/linux/tools目录下。

用户可执行如下命令，安装交叉编译工具：

```
cd /media/cdrom/linux/tools
tar xvjf arm-2007q3-51-arm-none-linux-gnueabi-i686.tar.bz2 -C /home/embest
```

注意: 默认安装到用户目录下，本文以/home/embest 为准，用户可适当换成自身目录即可。

6.1.2 其它工具的安装

源码编译中用到其它的一些工具，同样存放在光盘的 linux/tools 目录下，用户可执行以下命令安装：

```
mkdir /home/embest/tools
cp /media/cdrom/linux/tools/mkimage /home/embest/tools
cp /media/cdrom/linux/tools/signGP /home/embest/tools
cp /media/cdrom/linux/tools/mkfs.ubifs /home/embest/tools
cp /media/cdrom/linux/tools/ubinize /home/embest/tools
cp /media/cdrom/linux/tools/ ubinize.cfg /home/embest/tools
```

6.1.3 添加环境变量

以上工具安装完成后，还需要使用如下命令把它们添加到环境变量中：

```
export PATH=/home/embest/arm-2007q3/bin:/home/embest/tools:$PATH
```

注意: 用户可把它写入用户目录的.barsrc 文件中，那么系统启动的时候自动

完成环境变量的添加,查看路径可以使用 echo $PATH 命令。

6.2 系统编译

6.2.1 编译文件准备

系统所有组成部分的源码位于光盘的 linux/source 目录下，用户在进行开发前需要把它们解压到 linux 系统下，例如：

```
mkdir /home/embest/work
cd /home/embest/work
tar xvf /media/cdrom/linux/source/x-load-1.41.tar.bz2
tar xvf /media/cdrom/linux/source/u-boot-1.3.3.tar.bz2
tar xvf /media/cdrom/linux/source/linux-2.6.28-omap.tar.bz2
sudo tar xvf /media/cdrom/linux/source/rootfs.tar.bz2
```

执行完以上操作后，当前目录下会生成 linux-2.6.22-omap、u-boot-1.3.3、x-load-1.41、rootfs 四个目录。

6.2.2 一级启动代码编译

DevKit8000 支持 MMC/SD 启动或 NAND 启动，不同的启动方式烧写的 x-loader 的映像文件是不一样的，对应的映射生成方法也不同。

下面分别介绍用于不同启动方式下的 x-loader 映像文件的生成。

1、生成用于 SD 卡启动的 x-loader 映像文件 MLO

```
cd x-load-1.41
make distclean
make omap3devkit8000_config
make
signGP x-load.bin
mv x-load.bin.ift MLO
```

执行完以上操作后，当前目录下会生成我们需要的 MLO 文件。

2、生成用于 NAND 启动的 x-load.bin.ift_for_NAND

1）修改 x-loader-1.4.1/include/configs/omap3devkit8000.h 文件，

```
vi x-loader-1.4.1/include/configs/omap3devkit8000.h
```

注释以下行：

```
//#define CFG_CMD_MMC                   1
```

2）交叉编译

```
cd x-load-1.41
make distclean
make omap3devkit8000_config
make
signGP x-load.bin
mv x-load.bin.ift x-load.bin.ift_for_NAND
```

执行完以上操作后，当前目录下会生成我们需要的 x-load.bin.ift_for_NAND 文件。

6.2.3 二级启动代码编译

```
cd u-boot-1.3.3
make distclean
make omap3devkit8000_config
make
```

执行完以上操作后，当前目录下会生成我们需要的 u-boot.bin 文件。

6.2.4 内核编译

```
cd linux-2.6.28-omap
make distclean
make omap3_devkit8000_defconfig
make uImage
```

执行完以上操作后，arch/arm/boot 目录下会生成我们需要的 uImage 文件。

6.2.5 文件系统制作

```
cd /home/embest/work
sudo /home/embest/tools/mkfs.ubifs -r rootfs -m 2048 -e 129024 -c 812 -o ubifs.img
sudo /home/embest/tools/ubinize -o ubi.img -m 2048 -p 128KiB -s 512 /home/embest/tools/ubinize.cfg
```

执行完以上操作后，当前目录下会生成我们需要的ubi.img文件。

6.3 系统定制

事实上，linux内核有很多内核配置选项，用户可以在默认配置的基于上，增加或裁减驱动和一些内核特性，以更适合用户的需要。下面举例说明系统的定制的一般流程。

6.3.1 修改内核配置

出厂内核源码中提供有默认配置文件：

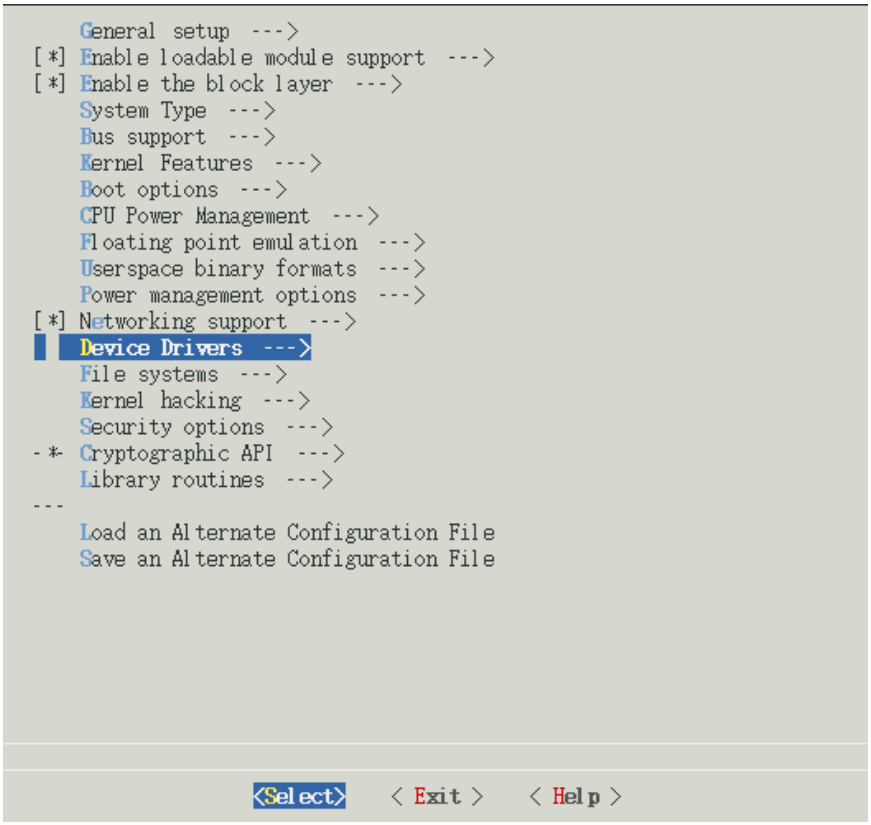
linux-2.6.28-omap/arch/arm/configs/ omap3_devkit8000_defconfig

用户可在其基础上进行系统定制：

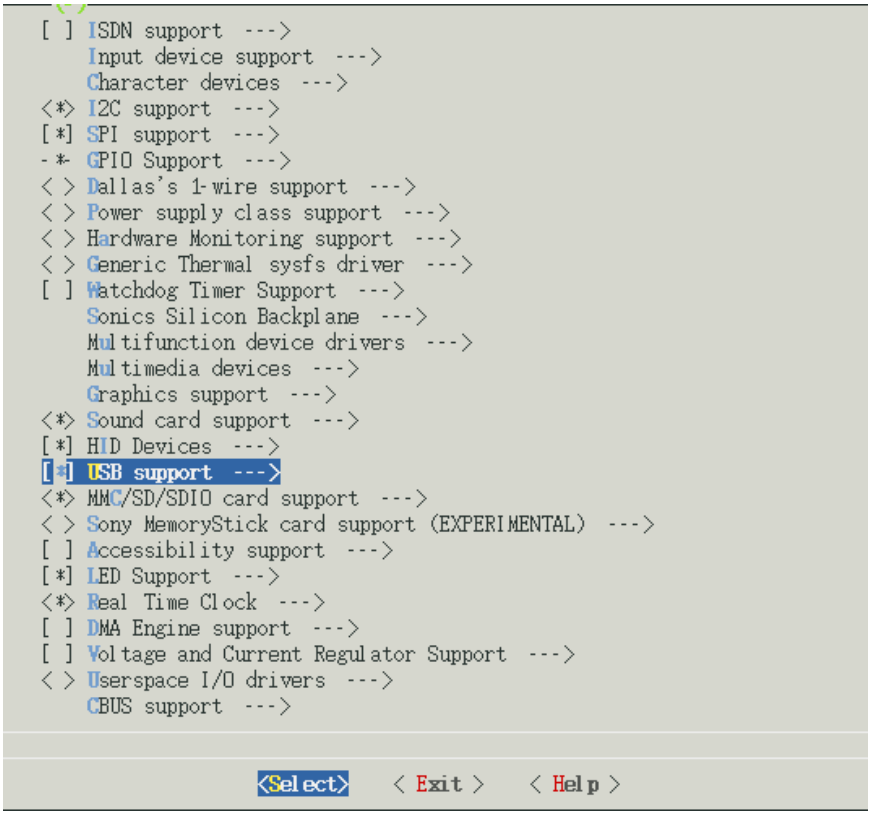
```
cd linux-2.6.28-omap
cp arch/arm/configs/omap3_devkit8000_defconfig .config
make menuconfig
```

下面以 usb gadget 模拟 usb mass storage device 为例子，举例介绍系统的定制：

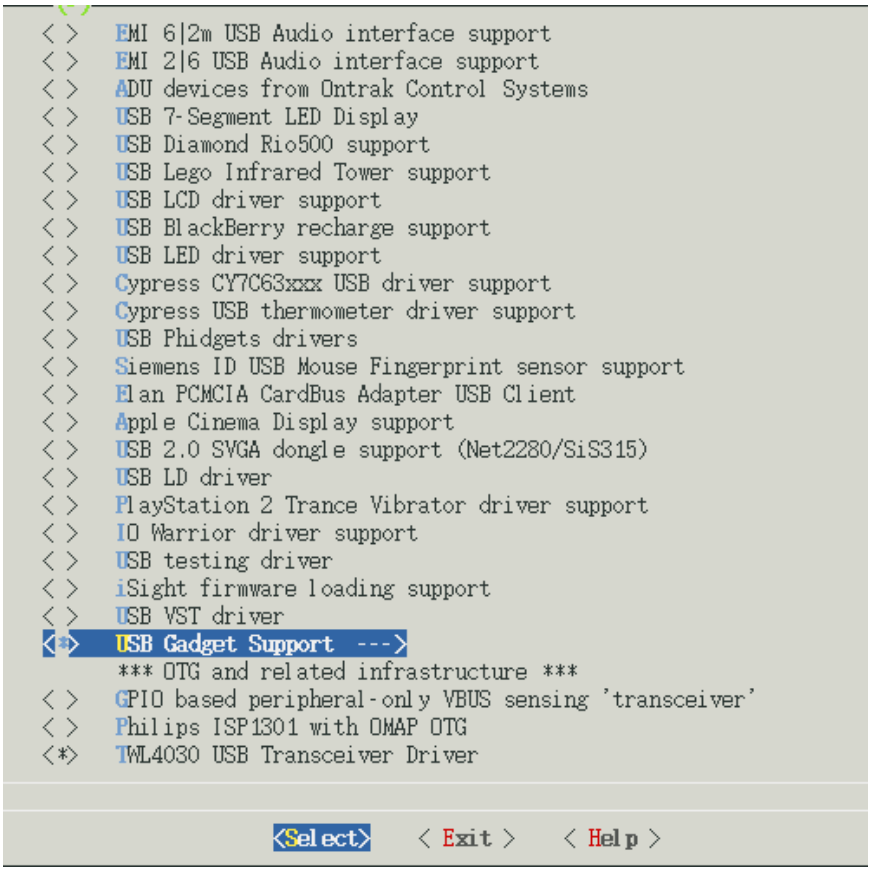
1、选择 Device drivers



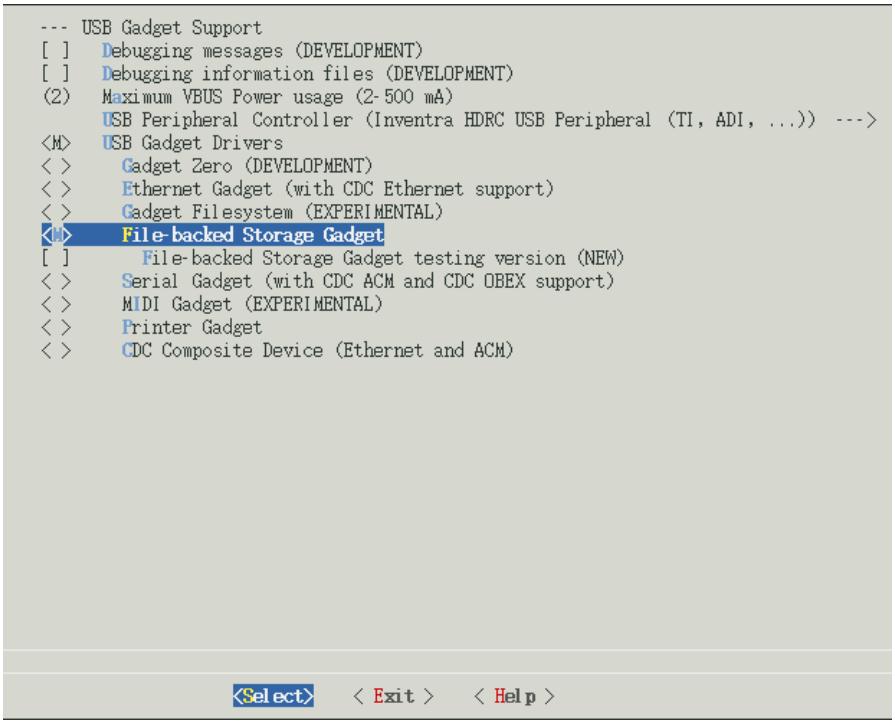
2、选择 USB support



3、选择 USB Gadget Support



4、修改 USB Gadget Support 配置如下



6.3.2 编译

保存配置，执行以下命令重新编译内核：

```
make
make uImage
```

执行完以上操作后，arch/arm/boot 目录下生成新的内核映像 uImage，drivers/usb/gadget 目录下生成模块文件 g_file_storage.ko。

6.3.3 测试

更新 SD 卡下内核映像文件 uImage，拷贝 g_file_storage.ko 文件到 sd 卡中，参考【5.1.2 SD 卡启动】从 SD 卡启动系统后，执行如下命令把 sd 卡模拟成 usb mass storage device 供 PC 访问：

```
root@DevKit8000:~#  cd /media/mmcblk0p1/
root@DevKit8000:/media/mmcblk0p1#  insmod g_file_storage.ko file=/dev/mmcblk0p1 stall=0 removable=1
g_file_storage gadget: File-backed Storage Gadget, version: 7 August 2007
g_file_storage gadget: Number of LUNs=1
g_file_storage gadget-lun0: ro=0, file: /dev/mmcblk0p1
musb_hdrc musb_hdrc: MUSB HDRC host driver
musb_hdrc musb_hdrc: new USB bus registered, assigned bus number 2
usb usb2: configuration #1 chosen from 1 choice
hub 2-0:1.0: USB hub found
hub 2-0:1.0: 1 port detected
```

使用 USB mini B to USB A 转接线连接开发板与 pc 机,PC 上会提示发现 usb mass storage device，然后出现新的移动硬盘，用户可对其进行正常操作。

注意:没有更新内核映像或没有从 SD 卡启动，会导致 g_file_storage.ko 模块会加载失败，出现类似提示：

insmod: cannot insert '/media/mmcblk0p1/g_file_storage.ko': Device or resource busy

---

第七章 Linux 系统映像烧写

DevKit8000 支持从 MMC/SD 或 NAND 启动，不同启动方式下映像的烧写方法有所不同。下面分别介绍不同启动方式下映像文件的烧写。

7.1 SD 卡系统映像更新

7.1.1 SD 卡格式准备

推荐使用 HP USB Disk Storage Format Tool 2.0.6：
http://www.embedinfo.com/english/download/SP27213.exe

1、把 MMC/SD 卡插入 PC 下读卡器中

2、打开 HP USB Disk Storage Format Tool，出现类似提示如下：

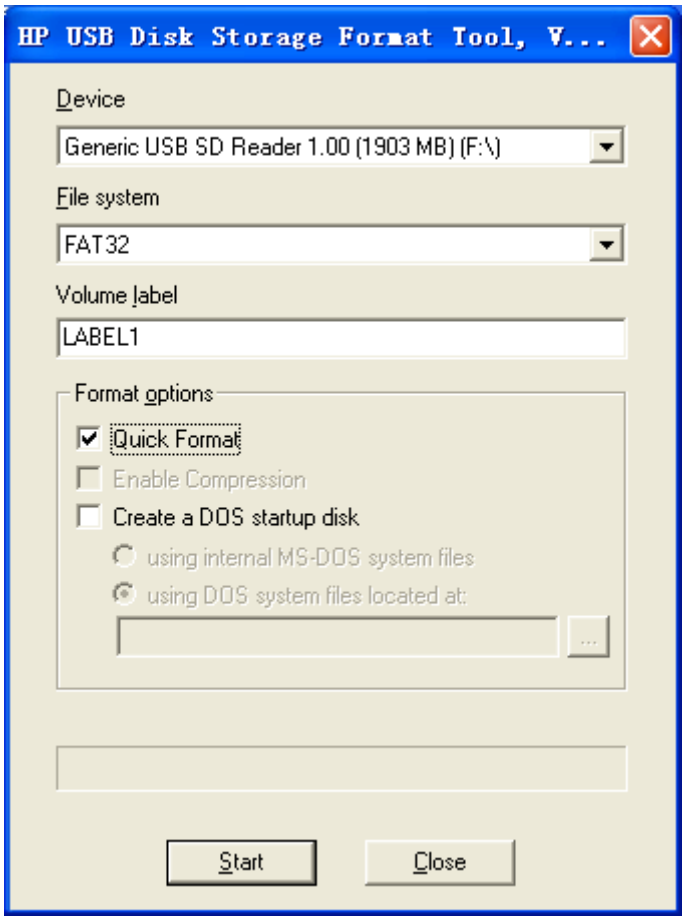


图 7.1 HP USB Disk 格式化工具

3、选择”FAT32“系统格式

4、点击”Start”

5、等待格式化完成，点击”OK”

注意：请不要用这个软件将有分区的 SD 卡格式化，这会将 SD 卡的第二个分区也会删除！

7.1.2 映像更新

直接在 pc 下用编译生产的映像文件替换 sd 卡下对应映像文件即可。

注意：光盘里有已经编译好的映像文件，其路径在：disk/linux/image，直接 COPY 目录下的所有文件。

7.1.3 u-boot 参数设置

u-boot 交互模式下，用户可执行以下命令恢复出厂参数，打开电源，当串口显示黑体字处，按任意键：

```
Texas Instruments X-Loader 1.41
Starting OS Bootloader...



U-Boot 1.3.3-svn323 (Dec  7 2009 - 10:35:16)

OMAP3530-GP rev 2, CPU-OPP2 L3-165MHz
Devkit8000 Board + LPDDR/NAND
DRAM:  128 MB
NAND:  128 MiB
Display-bmp: 480 x 272  with 256 colors
In:    serial
Out:   serial
Err:   serial
Hit any key to stop autoboot:  3 （在这里按任意键进入 u-boot）
OMAP3 DevKit8000 # setenv bootargs console=ttyS2,115200n8 root=/dev/ram initrd=0x81600000,40M video=omapfb:mode:4.3inch_LCD
OMAP3 DevKit8000 # setenv bootcmd 'mmcinit;fatload mmc 0 80300000 uImage;fatload mmc 0 81600000 ramdisk.gz;bootm 80300000'
OMAP3 DevKit8000 # saveenv
```

7.2 NAND Flash 系统映像更新

Nand 启动映像的更新需要借助于 u-boot 来完成。u-boot 下可通过 MMC/SD 或 TFTP 两种方式对 nand 启动映像进行更新。下面介绍基于 MMC/SD 的更新方法。

将光盘里的 **x-load.bin.ift_for_NAND，flash-uboot.bin，uImage，ubi.img** 映像文件拷贝到 SD 卡中，将 SD 卡插入开发板，读秒时，按任意键进入 uboot,执行以下步骤

7.2.1 一级启动代码更新

```
OMAP3 DevKit8000 #  mmcinit
OMAP3 DevKit8000 #  fatload mmc 0:1 80000000 x-load.bin.ift_for_NAND
reading x-load.bin.ift_for_NAND
9664 bytes read
OMAP3 DevKit8000 #  nand unlock
device 0 whole chip
```

```
nand_unlock: start: 00000000, length: 134217728!
NAND flash successfully unlocked
OMAP3 DevKit8000 #    nand ecc hw
OMAP3 DevKit8000 #    nand erase 0 80000

NAND erase: device 0 offset 0x0, size 0x80000
Erasing at 0x60000 -- 100% complete.
OK
OMAP3 DevKit8000 #    nand write.i 80000000 0 $(filesize)

NAND write: device 0 offset 0x0, size 0x80000

Writing data at 0x7f800 -- 100% complete.
 524288 bytes written: OK
```

7.2.2 二级启动代码更新

```
OMAP3 DevKit8000 #    mmcinit
OMAP3 DevKit8000 #    fatload mmc 0:1 80000000 flash-uboot.bin
reading flash-uboot.bin

1085536 bytes read
OMAP3 DevKit8000 #    nand unlock
device 0 whole chip
nand_unlock: start: 00000000, length: 134217728!
NAND flash successfully unlocked
OMAP3 DevKit8000 #    nand ecc sw
OMAP3 DevKit8000 #    nand erase 80000 160000

NAND erase: device 0 offset 0x80000, size 0x160000
Erasing at 0x1c0000 -- 100% complete.
OK
OMAP3 DevKit8000 #    nand write.i 80000000 80000 $(filesize)

NAND write: device 0 offset 0x80000, size 0x160000

Writing data at 0x1df800 -- 100% complete.
 1441792 bytes written: OK
```

7.2.3 内核更新

```
OMAP3 DevKit8000 #    mmcinit
OMAP3 DevKit8000 #    fatload mmc 0:1 80000000 uImage
reading uImage
```

```
1991900 bytes read
OMAP3 DevKit8000 #    nand unlock
device 0 whole chip
nand_unlock: start: 00000000, length: 268435456!
NAND flash successfully unlocked
OMAP3 DevKit8000 #    nand ecc sw
OMAP3 DevKit8000 #    nand erase 280000 300000

NAND erase: device 0 offset 0x280000, size 0x200000
Erasing at 0x460000 -- 100% complete.
OK
OMAP3 DevKit8000 #    nand write.i 80000000 280000 $(filesize)

NAND write: device 0 offset 0x280000, size 0x200000

Writing data at 0x47f800 -- 100% complete.
 2097152 bytes written: OK
```

7.2.4 文件系统更新

```
OMAP3 DevKit8000 #    mmcinit
OMAP3 DevKit8000 #    fatload mmc 0:1 81000000 ubi.img
reading ubi.img

12845056 bytes read
OMAP3 DevKit8000 #    nand unlock
device 0 whole chip
nand_unlock: start: 00000000, length: 268435456!
NAND flash successfully unlocked
OMAP3 DevKit8000 #    nand ecc sw
OMAP3 DevKit8000 #    nand erase 680000

NAND erase: device 0 offset 0x680000, size 0x7980000
Erasing at 0x7fe0000 -- 100% complete.
OK
OMAP3 DevKit8000 #    nand write.i 81000000 680000 $(filesize)

NAND write: device 0 offset 0x680000, size 0xc40000

Writing data at 0x12bf800 -- 100% complete.
 12845056 bytes written: OK
```

7.2.5 u-boot 参数设置

```
OMAP3 DevKit8000 # setenv bootargs console=ttyS2,115200n8 ubi.mtd=4 root=ubi0:rootfs rootfstype=ubifs video=omapfb:mode:4.3inch_LCD
```

OMAP3 DevKit8000 # **setenv bootcmd nand read.i 80300000 280000 300000\; bootm 80300000**
OMAP3 DevKit8000 # **saveenv**

第八章 Linux 应用程序开发

本章主要介绍如何在DevKit8000硬件平台上进行应用程序的开发，包括DevKit8000软件环境的搭建等，并通过实例来说明在DevKit8000进行应用程序开发的一般流程。

8.1 LED 应用程序开发示例

8.1.1 编写代码

led_acc.c 源码，控制开发板上的三个 led 灯按累加器的方式闪烁。

```
#include <stdio.h>
#include <unistd.h>
#include <sys/types.h>
#include <sys/ipc.h>
#include <sys/ioctl.h>
#include <fcntl.h>

#define LED1 "/sys/class/leds/led1/brightness"
#define LED2 "/sys/class/leds/led2/brightness"
#define LED3 "/sys/class/leds/led3/brightness"

int main(int argc, char *argv[])
{
	int f_led1, f_led2, f_led3;
	unsigned char i = 0;
	unsigned char dat1, dat2, dat3;
	if((f_led1 = open(LED1, O_RDWR)) < 0){
		printf("error in open %s",LED1);
		return -1;
	}
	if((f_led2 = open(LED2, O_RDWR)) < 0){
		printf("error in open %s",LED2);
		return -1;
	}
	if((f_led3 = open(LED3, O_RDWR)) < 0){
		printf("error in open %s",LED3);
		return -1;
	}
	for(;;){
		i++;
		dat1 = i&0x1 ? '1':'0';
		dat2 = (i&0x2)>>1 ? '1':'0';
		dat3 = (i&0x4)>>2 ? '1':'0';
		write(f_led1, &dat1, sizeof(dat1));
```

```
            write(f_led2, &dat2, sizeof(dat2));
            write(f_led3, &dat3, sizeof(dat3));
            usleep(300000);
        }
    }
```

8.1.2 交叉编译

```
arm-none-linux-gnueabi-gcc led_acc.c -o led_acc
```

8.1.3 下载运行

通过 SD 卡或 U 盘或网络下载到开发板系统，进入 led_acc 文件所在的目录，输入下面命令回车 led_acc 即在后台运行。

```
./led_acc
```

第四部分 WinCE 软件系统

第九章 WinCE 系统概述

DevKit8000 软件系统包括：预编译的映像，应用程序的映像及应用程序对应的静态库、动态库、头文件及源码、编译工具、开发辅助工具等。生成映像及应用程序所用到的编译工具用户可以从微软公司取得。在本手册的附录部分，指明了各种资源获取的链接。DevKit8000 映像及应用程序、源码、辅助工具等可以从 DevKit8000 评估套件发行光盘或配件 SD 卡中找到。

DevKit8000 评估套件配件 SD 卡内已安装软件包括：

- X-Loader 映像（MLO）
- Ethernet Bootloader（EBOOT）映像（EBOOTSD.nb0）
- Windows Embedded CE 6.0 样例操作系统映像（NK.bin）
- 测试应用程序（ADevKit8000.exe）

DevKit8000 评估套件发行光盘还包括：

- 针对 TI OMAP35X 的 Windows Embedded CE 6.0 DevKit8000 Board Support Package（BSP）源码
- 基于 DevKit8000 BSP 的 Windows Embedded CE 6.0 参考工程
- DevKit8000 应用程序开发实例源码
- 开发辅助工具

本章主要是概述 DevKit8000 软件系统，包括：预编译映像、BSP 及板级测试包以及光盘中所提供的各种映像和应用程序的特征说明等。

9.1 预装软件

预编译映像包括：引导映像 X-Loader 及 EBOOT 和样例操作系统映像。X-Loader 是第一级用户引导代码。系统上电后由 CPU 内部 ROM 自动拷贝 MLO 映像到内部 RAM 并执行。X-Loader 主要负责初始化 CPU，拷贝 EBOOT 到 DDR 内存中并执行 EBOOT。EBOOT 是第二级用户引导代码，默认情况下拷贝操作系统映像到 DDR 内存中并将系统控制权交给操作系统。EBOOT 还提供对底层硬件的管理操作及与操作系统共享数据的设置。

Windows Embedded CE 6.0 提供多媒体、工业、PDA、手机和微内核等模版，用户可根据应用选择合适的模版。预编译映像以 PDA 下 Mobile Handheld 为模版，支持特征包括：

映像	特征
X-Loader	引导 EBOOT
EBOOT	从网络引导操作系统（网卡或 RNDIS）
	从 SD 引导操作系统
	从 NAND Flash 引导操作系统
样例操作系统	Windows Explorer
	Console Window
	CAB File Installer/Uninstaller
	Internet Explorer 6.0
	ActiveSync

	Power Management (Full)
	.NET Compact Framework 3.5
	Hive-based Registry
	RAM and ROM File System
	Device Drivers

9.2 板级支持包（BSP）特征

DevKit8000 BSP 用于定制运行于 DevKit8000 硬件平台下的引导映像及 Windows Embedded CE 6.0 操作系统映像，支持特征包括：

模块	特征
X-Loader 模块	NAND
	ONENAND
	SD
EBOOT 模块	NAND
	ONENAND
	SD
OAL 模块	ILT
	REBOOT
	Watchdog
	RTC
KITL 模块	RNDIS KITL
驱动模块	NLED 驱动
	GPIO/I2C/SPI/MCBSP 驱动
	串口驱动
	6X6 键盘驱动
	音频驱动
	NAND（K9F1G08）驱动
	显示驱动（LCD/DVI、S 端子/TV）/ TOUCH 驱动
	SD/MMC/SDIO 驱动
	DM9000 网卡驱动
	USB OTG 驱动
	USB EHCI 驱动
	VRFB 驱动
	DSPLINKK/CMEMK 驱动
	GPIO 键盘驱动
	PWM（TPS65930）驱动
	ADC（TPS65930）驱动
	ONENAND 驱动
	SMSC911X 网卡驱动
电源管理模块	背光驱动
	电池驱动

	休眠/唤醒按键驱动
	电源管理扩展
应用程序模块	Flash 播放插件和 Flash 播放器
	MP3/MPEG4/H264 DSP 硬件解码器
	BSPINFO（控制面板）
	CETK 扩展

注：部分模块 DevKit8000 未提供硬件平台支持,BSP 附带的一些源代码可能受到第三方版权的保护。

第十章 WinCE 系统快速操作

10.1 系统启动方法

注：在系统操作的过程中，需用到PC 机超级终端时，超级终端的配置如下所示：
波特率：115200
数据位： 8
奇偶检验：无
停止位： 1
流控制：无

10.1.1 NAND Flash 启动

主板默认配置为先从 NAND 启动，如果 NAND 无系统映像将会从 SD 卡启动。出厂时 nand flash 下固化有代码，用户只需要连接好串口，接通电源，系统随即从 nand flash 启动。

注：NAND Flash 中系统映像的更新方式请参考【12.2 NAND Flash 系统映像更新】

10.1.2 SD 卡启动

如果用户需要切换到 SD 卡启动，只需要按住主板上的 BOOT_KEY 按键（按键位置请参考【3.18 按键】），上电启动，系统随即改为从 MMC/SD 启动。

注：SD 卡中系统映像的更新方式请参考【12.1 SD 卡系统映像更新】

10.2 测试程序应用

10.2.1 测试程序介绍

DevKit8000 测试软件是 Windows Embedded CE 6.0 应用程序，用于测试 DevKit8000 软硬件平台，支持特征包括：

- SD 卡自动测试
- NAND Flash 盘自动测试
- 网络自动测试
- 键盘手动测试
- RTC 自动测试
- NLED 半自动测试
- 音频输入输出半自动测试
- LCD 显示半自动测试

10.2.2 测试程序使用

注：此测试需连接评估板的串口到PC 终端，评估板上需连接 LCD 屏，键盘，音频输入

输出设备，如缺少设备可能会影响测试结果。

1. 打开 PC Window 超级终端软件并设置：
   波特率：115200
   数据位： 8
   奇偶检验：无
   停止位： 1
   流控制：无
2. 关闭 PC 防火墙或允许局域网数据通讯并设置网络属性：
   IP 地址：192.168.1.2
   子网掩码：255.255.255.0
3. 插入 SD 卡，并连接串口线到 PC 端，然后连接键盘及录音放音设备
4. 上电，超级终端出现系统启动信息，稍等片刻，进入系统后开始体验 Windows Embedded CE 6.0。
5. 运行测试程序 ADevKit8000.exe[\Storage Card]。
6. 点击按钮 Start，自动测试开始。
7. 自动弹出 Keypad 窗口，测试键盘，按 ESC 键或点击退出按钮退出键盘测试。
8. LED 灯流水闪烁等待用户判断。
9. 播放开机声音 5 次等待用户判断。用户准备好录音后选择 YES，进入录音模式，5 秒后播放所录制的音频文件并等待用户判断。
10. 屏幕依次显示 RGB 三原色，等待用户判断。
11. 测试完成后表格显示测试结果：
    红色字体：FAIED
    绿色字体：PASS

同时输出到测试程序编辑框及 ADevKit9000.log 文件。点击表格关闭结果窗口。

注：若 SD 卡内文件系统损坏可以参考 6.3 章节 SD 卡更新进行恢复。

第十一章 WinCE 系统开发

11.1 开发环境搭建

11.1.1 交叉编译环境的搭建

基于 DevKit8000 的开发涉及两个层面：底层是基于 DevKit8000 的硬件配置而开发的 Windows Embedded CE 6.0 操作系统；上层是开发基于该操作系统的应用程序。两个层面的 Windows Embedded CE 6.0 开发都需基于 Visual Studio 2005（VS2005）集成开发环境。

开发应用程序需安装以下软件及更新：

- Visual Studio 2005
- Visual Studio 2005 SP1
- Visual Studio 2005 SP1 Update for Vista (if applicable)
- ActiveSync 4.5

开发 Windows Embedded CE 6.0 操作系统需依次安装以下软件及更新：

- Visual Studio 2005
- Visual Studio 2005 SP1
- Visual Studio 2005 SP1 Update for Vista (if applicable)
- Windows Embedded CE 6.0 Platform Builder
- Windows Embedded CE 6.0 SP1
- Windows Embedded CE 6.0 R2
- Windows Embedded CE 6.0 Product Update Rollup 12/31/2008

注：
若系统安装有旧版本的 CE 开发环境，可能会影响 Windows Embedded CE 6.0 开发平台的使用，建议卸载后再安装。
请参考附录部分，确定各种资源获得渠道的信息；
以上各软件或组件系统由于存在依赖关系，建议严格按照所列次序安装，并安装在默认路径。

11.2 系统编译

若 DevKit8000 评估套件发行光盘提供的 Windows Embedded CE 6.0 样例操作系统映像符合你的应用，你只需简单的加入你的应用程序，取得微软公司授权后发行即可。否则，你需要重新定制系统并生成映像。本节介绍如何利用 DevKit8000 Board Support Package（BSP）开发运行于 DevKit8000 硬件平台上的 Windows Embedded CE 6.0 操作系统映像。

11.2.1 编译文件准备

天漠公司已经完成了在 DevKit8000 硬件平台上的驱动和相关资源的整合，所以用户在定制在 DevKit8000 上专用的 Windows Embedded CE 6.0 系统之前，需要如下准备工作：

解压[\wince_6\bsp\DevKit8000.rar]得到 DevKit8000 目录.

- 拷贝解压目录[\DevKit8000\bsp\DevKit8000]到[C:\WINCE600\PLATFORM]目录下。
- 拷贝解压目录[\DevKit8000\bsp_prj\DevKit8000]到[C:\WINCE600\OSDesigns]目录下。

注意： C:\WINCE600\OSDesigns 的该 OSDesigns 文件夹需要自行建立

对于使用 4.3 寸屏

修改 platform/DevKit8000/src/drivers/lcd/vga/lcd_vga.c

```
#define LCD_4_3_INCH    1
//#define LCD_5_6_INCH  1
//#define LCD_7_INCH    1
//#define LCD_10.4_INCH 1
```

对于使用 5.6 寸屏

修改 platform/DevKit8000/src/drivers/lcd/vga/lcd_vga.c

```
//#define LCD_4_3_INCH  1
#define LCD_5_6_INCH    1
//#define LCD_7_INCH    1
//#define LCD_10.4_INCH 1
```

对于使用 7 寸屏

修改 platform/DevKit8000/src/drivers/lcd/vga/lcd_vga.c

```
//#define LCD_4_3_INCH  1
//#define LCD_5_6_INCH  1
#define LCD_7_INCH      1
//#define LCD_10.4_INCH 1
```

对于使用 10.4 寸屏

修改 platform/DevKit8000/src/drivers/lcd/vga/lcd_vga.c

```
//#define LCD_4_3_INCH  1
//#define LCD_5_6_INCH  1
//#define LCD_7_INCH    1
#define LCD_10.4_INCH   1
```

对于使用 VGA 模块

修改 C:\WINCE600\DevKit8000\BSP\DevKit8000\DevKit8000.bat

```
set BSP_DVI_1024W_768H=1
```

注：
利用 DevKit8000 BSP 开发 Windows Embedded CE 6.0 操作系统需要搭建 Windows Embedded CE 6.0 开发平台。
本手册约定 Windows Embedded CE 6.0 开发平台软件均按默认路径安装，即 Windows Embedded CE 6.0 安装路径为[C:\WINCE600]。

11.2.2 系统编译

（1） 用户可以直接打开工程文件 DevKit8000.sln[C:\WINCE600\OSDesigns\DevKit8000]，或者按下列步骤新建工程文件：

- 打开 Visual Studio 2005。

- 选择菜单 File[New->Project]。
- 选择 Platform Builder for CE 6.0 模板类型。
- 选择工程名后打开 Windows Embedded CE 6.0 OS Design Wizard。
- 选择 Embest DevKit8000 BSP 在 BSP 列表。
- 继续完成向导。

（2） 选中菜单[Build-> Global Build Settings]子菜单：

- Copy Files to Release Directory After Build
- Make Run-Time Image After build

（3） 若需要打开 KITL，选中 Enable Kernel Debugger 和 Enable KITL 在 Build Options 页[Project-> Properties]。

（4） 选择菜单[Build-> Build Solution]编译 BSP。该操作是完整编译包括 sysgen 操作系统组件。在完成一次完整编译后，可使用 Solution Explorer 窗口下的编译命令，节约编译时间。

（5） 生成映像包括 NK.bin 、EBOOTSD.nb0 和 MLO 等，拷贝目录[C:\WINCE600\OSDesigns\DevKit8000\DevKit8000\RelDir\DevKit8000_ARMV4I_Release]下文件 MLO、EBOOTSD.nb0 和 NK.bin 到 SD 卡，插入 SD 卡到设备，从 SD 卡启动设备即可测试。

注：
在系统编译过程中，用户应该在“解决方案配置”框中选择“DevKit8000 ARMV4I Release”。

11.2.3 系统定制

Windows Embedded CE 6.0 由若干独立模块组成，每个模块提供特定功能，其中部分模块又分成多个组件，有些组件还分成具体的特征。以便 OEM/ODM 根据具体的应用定制出稳定高效的版本。

DevKit8000 样例操作系统映像以 Mobile Handheld 为模版，添加组件特征包括：

组件	路径
CAB File Installer/Uninstaller	Core OS->CEBASE->Application – End User
.NET Compact Framework 3.5	Core OS->CEBASE->Applications and Services Development->.NET Compact Framework 3.5
OS Dependencies for .NET Compact Framework 3.5	Core OS->CEBASE->Applications and Services Development->.NET Compact Framework 3.5-> OS Dependencies for .NET Compact Framework 3.5
Point-to-Point Protocol over Ethernet (PPPoE)	Core OS->CEBASE->Communication Services and Networking->Networking – Wide Area Network (WAN)
USB Function Driver	Core OS->CEBASE->Core OS Services->USB Host Support
USB Host Support	Core OS->CEBASE->Core OS Services->USB Host Support
USB Human Input Device (HID) Class Driver	Core OS->CEBASE->Core OS Services->USB Host Support
USB HID Keyboard and Mouse	Core OS->CEBASE->Core OS Services->USB Host Support-> USB Human Input Device

	(HID) Class Driver
USB Storage Class Driver	Core OS->CEBASE->Core OS Services->USB Host Support
RAM and ROM File System	Core OS->CEBASE->File Systems and Data Store->File System – Internal (Choose 1)
Hive-based Registry	Core OS->CEBASE->File Systems and Data Store->Registry Storage – Internal (Choose 1)
exFAT File System	Core OS->CEBASE->File Systems and Data Store->Storage Manager
FAT File System	Core OS->CEBASE->File Systems and Data Store->Storage Manager
Storage Manager Control Panel Applet	Core OS->CEBASE->File Systems and Data Store->Storage Manager
Transaction-Safe FAT File System (TFAT)	Core OS->CEBASE->File Systems and Data Store->Storage Manager
Video/Image Compression Manager	Core OS->CEBASE->Graphics and Multimedia Technologies->Media->Video Codecs and Renderers
Console Window	Core OS->CEBASE->Shell and User Interface->Shell->Command Shell
SD Memory	Device Drivers->SDIO->SDIO Memory
serial	Device Drivers->USB Function->USB Function Clients
Windows Embedded CE Test Kit	Device Drivers

Windows Embedded CE 6.0 的定制在 Visual Studio 2005（VS2005）集成开发环境 Catalog Items View 窗口中添加或移除组件。

第十二章 WinCE 系统更新

12.1 SD 卡系统映像更新

12.1.1 SD 卡格式准备

用软件 HP Disk Storage Format Tool 格式化 SD 卡为 FAT 或 FAT32 文件系统。

12.1.2 映像更新

拷贝光盘目录 image[\wince_6]下子目录 dvi1280X720 或 lcd480X272 下文件 MLO、EBOOTSD.nb0、NK.bin、wavrec.exe 和 ADevKit8000.exe 到 SD 卡。

注：

- *发行光盘不提供 HP Disk Storage Format Tool 软件，用户可以通过路径 [http://www.embedinfo.com/english/download/SP27213.exe] 下载。*
- *目录 dvi1280X720 和 lcd480X272 对应 DVI 输出 1280X720 分辨率的屏和 LCD 输出 480X272 分辨率的 TFT 屏。*

12.2 NAND Flash 系统映像更新

12.2.1 更新文件准备

（1） 用软件 HP Disk Storage Format Tool 格式化 SD 卡为 FAT 或 FAT32 文件系统。

（2） 拷贝光盘目录 image[\wince_6]下子目录 dvi1280X720 或 lcd480X272 下文件

MLO
EBOOTNAND.nb0
NK.bin
XLDRNAND.nb0
wavrec.exe
ADevKit8000.exe
（没有 EBOOTSD.nb0）

到 SD 卡并将重命名 EBOOTNAND.nb0 为 EBOOTSD.nb0。

12.2.2 映像更新

（1） 按住 BOOT 键，后插入 SD 卡重新启动系统。这时系统从 SD 卡启动。超级终端输出启动打印信息，按[SPACE]进入 EBOOT 菜单。

（2） 按[5]进入 Flash 管理菜单。

（3） 分别按[a]、[b]和[c]，写 XLDR、EBOOT 和 NK 映像。

（4） 然后按[0]键回到主菜单，并分别按下[2]、[4]、[7]和[y]更改启动设备。

（5） 拔除 SD 卡，后重新启动系统。这时系统将从 NAND Flash 启动。

第十三章 WinCE 应用程序开发

本章介绍如何开发基于 Windows Embedded CE 6.0 操作系统运行于 DevKit8000 硬件平台上的应用程序。在开始之前，需要安装 Windows Mobile 6 Professional SDK，你可以通过网址[http://www.microsoft.com/downloads/details.aspx?familyid=06111A3A-A651-4745-88EF-3D48091A390B&displaylang=en]获取该软件。

注：

- *开发 Windows Embedded CE 6.0 操作系统应用程序需要搭建 Windows Embedded CE 6.0 开发平台。*
- *本手册开发实例基于 Windows Mobile 6 Professional SDK 开发。*

13.1 应用程序接口与示例

DevKit8000 应用程序开发所用到的 API 均采用微软 Windows Embedded CE 6.0 标准应用程序接口定义，DevKit8000 仅在标准 API 基础上扩展了 GPIO 的接口定义。

注：

1.Windows Embedded CE 6.0 标准应用程序接口定义可以查看 MSDN Windows Embedded CE 6.0 API 相关帮助文档。

2.接口应用程序开发实例章节有部分标准 API 的使用例程供用户参考。

3.部分驱动导出的接口仅供驱动使用，应用程序无权限调用。

13.1.1 GPIO 应用程序接口与示例

GPIO 设备名 L"GIO1:"，扩展 DeviceIoControl 接口定义，对应 IOCTL 码包括：

IOCTL 码	描述
IOCTL_GPIO_SETBIT	GPIO 引脚置 1
IOCTL_GPIO_CLRBIT	GPIO 引脚清 0
IOCTL_GPIO_GETBIT	读 GPIO 引脚状态
IOCTL_GPIO_SETMODE	设置 GPIO 引脚工作模式
IOCTL_GPIO_GETMODE	读 GPIO 引脚工作模式
IOCTL_GPIO_GETIRQ	读 GPIO 引脚对应的 IRQ 号

操作示例如下：

1) 打开 GPIO 设备

```
    HANDLE hFile = CreateFile(_T("GIO1:"), (GENERIC_READ|GENERIC_WRITE),
(FILE_SHARE_READ|FILE_SHARE_WRITE), 0, OPEN_EXISTING, 0, 0);
```

2) 设置/读 GPIO 工作模式

```
    DWORD id = 0, mode = 0;
```

设置 GPIO 工作模式：

```
    DWORD pInBuffer[2];
    pInBuffer[0] = id;
    pInBuffer[1] = mode;
```

```
DeviceIoControl(hFile, IOCTL_GPIO_SETMODE, pInBuffer, sizeof(pInBuffer), NULL, 0, NULL, NULL);
```

读 GPIO 工作模式：

```
DeviceIoControl(hFile, IOCTL_GPIO_GETMODE, &id, sizeof(DWORD), &mode, sizeof(DWORD), NULL, NULL);
```

"id"为 GPIO 引脚号，"mode"为 GPIO 模式定义，包括：

模式定义	描述
GPIO_DIR_OUTPUT	输出模式
GPIO_DIR_INPUT	输入模式
GPIO_INT_LOW_HIGH	上升沿触发模式
GPIO_INT_HIGH_LOW	下降沿触发模式
GPIO_INT_LOW	低电平触发模式
GPIO_INT_HIGH	高电平触发模式
GPIO_DEBOUNCE_ENABLE	跳变触发使能

3) GPIO 引脚操作

```
DWORD id = 0, pin = 0;
```

输出高电平：

```
DeviceIoControl(hFile, IOCTL_GPIO_SETBIT, &id, sizeof(DWORD), NULL, 0, NULL, NULL);
```

输出低电平：

```
DeviceIoControl(hFile, IOCTL_GPIO_CLRBIT, &id, sizeof(DWORD), NULL, 0, NULL, NULL);
```

读引脚状态：

```
DeviceIoControl(hFile, IOCTL_GPIO_GETBIT, &id, sizeof(DWORD), &pin, sizeof(DWORD), NULL, NULL);
```

"id"为 GPIO 引脚号，"pin"返回引脚状态。

4) 其它可选操作

读 GPIO 引脚对应的 IRQ 号：

```
DWORD id = 0, irq = 0;
DeviceIoControl(hFile, IOCTL_GPIO_GETIRQ, &id, sizeof(DWORD), &irq, sizeof(DWORD), NULL, NULL);
```

"id"为 GPIO 引脚号，"irq"返回 IRQ 号。

5) 关闭 GPIO 设备

```
CloseHandle(hFile);
```

注意：

1）GPIO 引脚定义：0~191 MPU Bank1~6 GPIO 引脚，192~209 TPS65930 GPIO 0~17。

2）GPIO 中断模式仅供驱动使用，应用程序设置该模式无效。

IOCTL 码定义及 GPIO 模式定义见光盘文件[\wince_6\inc\gpio.h]，用户包含该头文件即可。

13.2 接口应用程序开发实例

本节主要介绍前文测试程序 ADevKit8000 测试程序的开发流程。

13.2.1 创建解决方案

（1） 打开 Visual Studio 2005。

（2） 选择菜单 File[New->Project]。

（3） 选择 MFC Smart Device Application 模板[Visual C++->Smart Device]。
（4） 选择工程名后打开 MFC Smart Device Application Wizard。
（5） 在 Platforms 页，只选中 Windows Mobile 6 Professional SDK。
（6） 在 Application Type 页选择 Dialog based 属性。
（7） 继续完成向导。

13.2.2 编辑资源及代码

具体代码参考工程 ADevKit9000[\wince_6\app]源代码。

13.2.3 编译运行

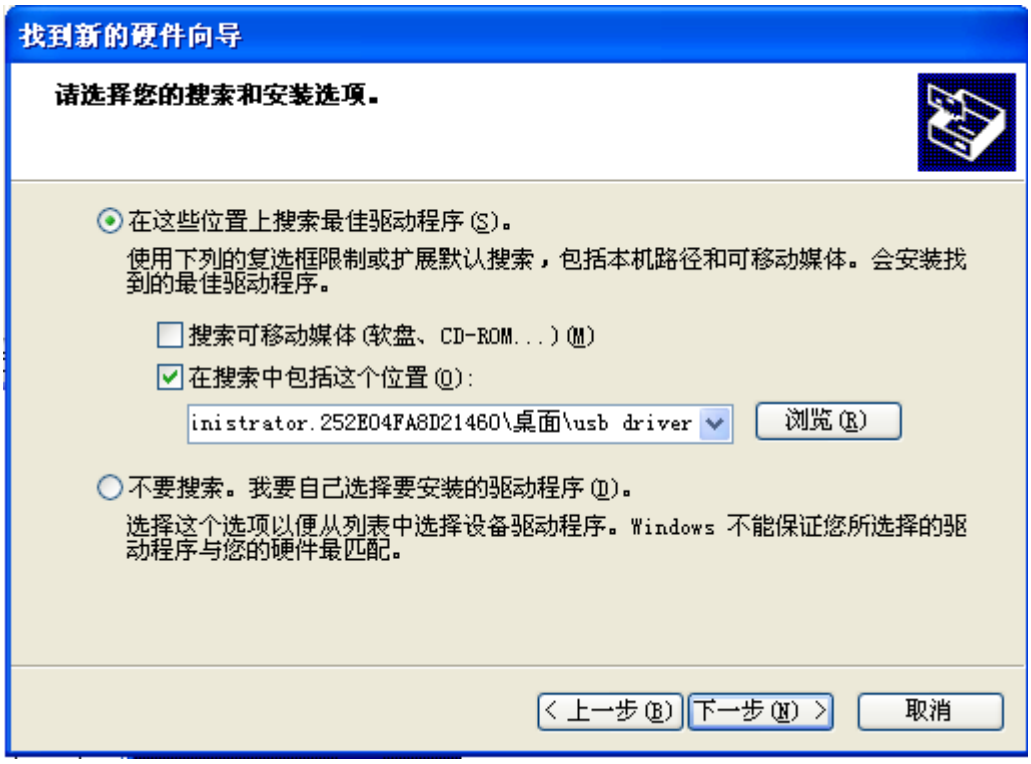
（1） 选择菜单[Build-> Build Solution]生成解决方案（ADevKit8000.exe）。
（2） 把生成的 ADevKit8000.exe 程序放到 SD 卡，并插 SD 卡到主板中，然后可在主板中运行此程序。

附录

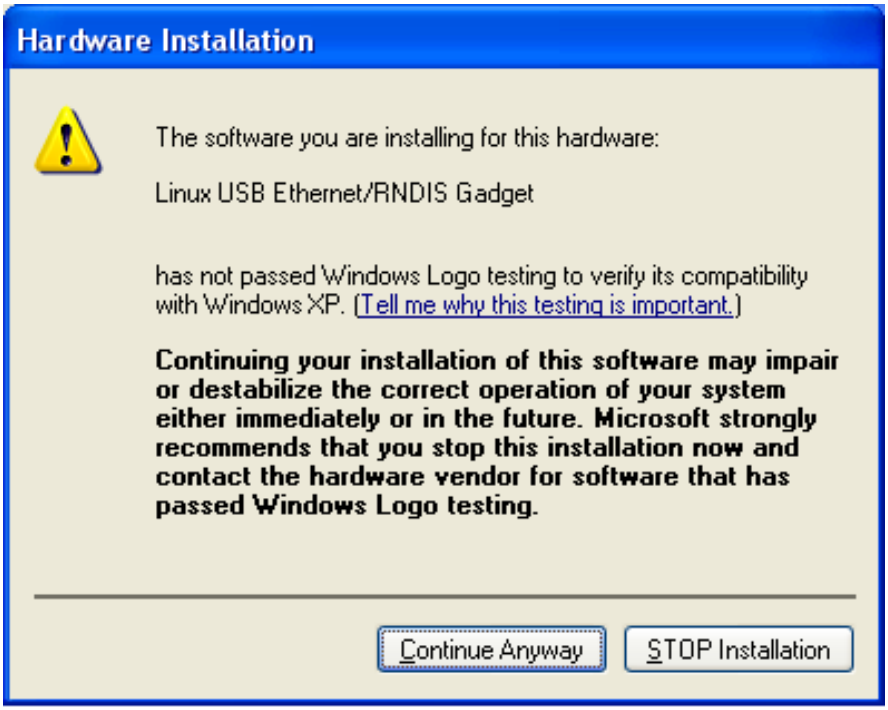
附录一 Linux USB Ethernet/RNDIS Gadget 驱动安装

1、如果你还没安装 Linux USB Ethernet/RNDIS Gadget 驱动，PC 会提示发现新硬件界面，选中“从列表或指定位置安装”，然后点击“下一步”。

2. 指定 usb 驱动路径[光盘\linux\tools]，然后点击“下一步”。



3、出现以下提示时，选择继续安装



4、等待驱动安装完毕


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:
Linux USB Ethernet/RNDIS Gadget
要关闭向导，请单击“完成”。
< 上一步(B)
完成
取消

附录二 Linux Boot Disk Format

How to create a dual-partition card for DevKit8000 to boot Linux from first partition and have root file system at second partition.

一、Introduction

This guide is meant for those looking to create a **dual-partition** card, booting from a FAT partition that can be read by the OMAP3 ROM bootloader and Linux/Windows, then utilizing an ext3 partition for the Linux root file system.

二、Details

Text marked with [] shows user input.

1、Determine which device the SD Card Reader is on your system

Plug the SD Card into the SD Card Reader and then plug the SD Card Reader into your system. After doing that, do the following to determine which device it is on your system.

```
$ [dmesg | tail]
...
[ 6854.215650] sd 7:0:0:0: [sdc] Mode Sense: 0b 00 00 08
[ 6854.215653] sd 7:0:0:0: [sdc] Assuming drive cache: write through
[ 6854.215659]  sdc: sdc1
[ 6854.218079] sd 7:0:0:0: [sdc] Attached SCSI removable disk
[ 6854.218135] sd 7:0:0:0: Attached scsi generic sg2 type 0
...
```

In this case, it shows up as /dev/sdc (note sdc inside the square brackets above).

2、Check to see if the automounter has mounted the SD Card

Note there may be more than one partition (only one shown in the example below).

```
$ [df -h]
Filesystem            Size  Used Avail Use% Mounted on
...
/dev/sdc1             400M   94M  307M  24% /media/disk
...
```

Note the "Mounted on" field in the above and use that name in the umount commands below.

3、If so, unmount it

```
$ [umount /media/disk]
```

4、Start fdisk

Be sure to choose the whole device (/dev/sdc), not a single partition (/dev/sdc1).

```
$ [sudo fdisk /dev/sdc]
```

5、Print the partition record

So you know your starting point. **Make sure to write down the number of bytes on the card (in this example, 2021654528).**

```
Command (m for help): [p]

Disk /dev/sdc: 2021 MB, 2021654528 bytes
255 heads, 63 sectors/track, 245 cylinders
Units = cylinders of 16065 * 512 = 8225280 bytes
   Device Boot      Start         End      Blocks   Id  System
/dev/sdc1   *           1         246     1974240+   c  W95 FAT32 (LBA)
Partition 1 has different physical/logical endings:
     phys=(244, 254, 63) logical=(245, 200, 19)
```

6、Delete any partitions that are there already

```
Command (m for help): [d]
Selected partition 1
```

7、Set the Geometry of the SD Card

If the print out above does not show 255 heads, 63 sectors/track, then do the following expert mode steps to redo the SD Card:

1）、Go into expert mode.

```
Command (m for help): [x]
```

2）、Set the number of heads to 255.

```
Expert Command (m for help): [h]
Number of heads (1-256, default xxx): [255]
```

3）Set the number of sectors to 63.

```
Expert Command (m for help): [s]
Number of sectors (1-63, default xxx): [63]
```

4）Now Calculate the number of Cylinders for your SD Card.

```
#cylinders = FLOOR (the number of Bytes on the SD Card (from above) / 255 / 63 / 512 )

So for this example:   2021654528 / 255 / 63 / 512 = 245.79.   So we use 245 (i.e. truncate, don't round).
```

5）Set the number of cylinders to the number calculated.

```
Expert Command (m for help): [c]
Number of cylinders (1-256, default xxx): [enter the number you calculated]
```

6）Return to Normal mode.

```
Expert Command (m for help): [r]
```

8、Print the partition record to check your work

```
Command (m for help): [p]

Disk /dev/sdc: 2021 MB, 2021654528 bytes
255 heads, 63 sectors/track, 245 cylinders
Units = cylinders of 16065 * 512 = 8225280 bytes

   Device Boot      Start         End      Blocks   Id  System
```

9、Create the FAT32 partition for booting and transferring files from Windows

```
Command (m for help): [n]
Command action
   e   extended
   p   primary partition (1-4)
[p]
Partition number (1-4): [1]
First cylinder (1-245, default 1): [(press Enter)]
Using default value 1
Last cylinder or +size or +sizeM or +sizeK (1-61, default 61): [+5]

Command (m for help): [t]
Selected partition 1
Hex code (type L to list codes): [c]
Changed system type of partition 1 to c (W95 FAT32 (LBA))
```

10、Mark it as bootable

```
Command (m for help): [a]
Partition number (1-4): [1]
```

11、Create the Linux partition for the root file system

```
Command (m for help): [n]
Command action
   e   extended
   p   primary partition (1-4)
[p]
Partition number (1-4): [2]
First cylinder (7-61, default 7): [(press Enter)]
Using default value 52
Last cylinder or +size or +sizeM or +sizeK (7-61, default 61): [(press Enter)]
Using default value 245
```

12、Print to Check Your Work

```
Command (m for help): [p]

Disk /dev/sdc: 2021 MB, 2021654528 bytes
```

```
255 heads, 63 sectors/track, 245 cylinders
Units = cylinders of 16065 * 512 = 8225280 bytes

   Device Boot      Start         End      Blocks   Id  System
/dev/sdc1   *           1           6      409626    c  W95 FAT32 (LBA)
/dev/sdc2               7          61     1558305   83  Linux
```

13、Save the new partition records on the SD Card

This is an important step. All the work up to now has been temporary.

```
Command (m for help): [w]
The partition table has been altered!

Calling ioctl() to re-read partition table.

WARNING: Re-reading the partition table failed with error 16: Device or resource busy.
The kernel still uses the old table.
The new table will be used at the next reboot.

WARNING: If you have created or modified any DOS 6.x
partitions, please see the fdisk manual page for additional
information.
Syncing disks.
```

14、Format the partitions

The two partitions are given the volume names LABEL1 and LABEL2 by these commands. You can substitute your own volume labels.

```
$ [sudo mkfs.msdos -F 32 /dev/sdc1 -n LABEL1]
mkfs.msdos 2.11 (12 Mar 2005)

$ [sudo mkfs.ext3 -L LABEL2 /dev/sdc2]
mke2fs 1.40-WIP (14-Nov-2006)
Filesystem label=
OS type: Linux
Block size=4096 (log=2)
Fragment size=4096 (log=2)
195072 inodes, 389576 blocks
19478 blocks (5.00%) reserved for the super user
First data block=0
Maximum filesystem blocks=402653184
12 block groups
32768 blocks per group, 32768 fragments per group
16256 inodes per group
Superblock backups stored on blocks:
        32768, 98304, 163840, 229376, 294912
```

```
Writing inode tables: done
Creating journal (8192 blocks): done
Writing superblocks and filesystem accounting information:
```

注意:在 ubuntu 下格式化好 FAT 和 EXT3 双分区后，FAT 分区需要在 window 下重新格式化一次，否则可能会出现无法从 SD 卡启动的情况

附录三 TFTP 服务器搭建

1、 安装客户端

```
$>sudo apt-get install tftp-hpa
$>sudo apt-get install tftpd-hpa
```

2、 安装 inet

```
$>sudo apt-get install xinetd
$>sudo apt-get install netkit-inetd
```

3、 服务器配置

首先，在根目录下建一个 tftpboot，并把属性改成任意用户可读写：

```
$>cd /
$>sudo mkdir tftpboot
$>sudo chmod 777 tftpboot
```

其次，在/etc/inetd.conf 里添加：

```
$>sudo  vi  /etc/inetd.conf      //把下面的语句添加的此文件里
tftpd dgram udp wait root /usr/sbin/in.tftpd /usr/sbin/in.tftpd -s /tftpboot
```

然后，重新加载 inetd 进程：

```
$>sudo /etc/init.d/inetd reload
```

最后，进入目录 /etc/xinetd.d/，并在其中新建文件 tftp，把指定的内容加入到 tftp 文件中：

```
$>cd /etc/xinetd.d/      //进入目录 /etc/xinetd.d/
$>sudo touch tftp        //新建文件 tftp
$>sudo vi tftp           //编辑文件 tftp,把下面内容加入 tftp 文件中
service tftp
{
    disable = no
    socket_type   = dgram
    protocol      = udp
    wait          = yes
    user          = root
    server         = /usr/sbin/in.tftpd
    server_args    = -s /tftpboot -c
    per_source     = 11
    cps             = 100 2
}
```

4、 重新启动服务：

```
$>sudo /etc/init.d/xinetd restart
$>sudo in.tftpd -l /tftpboot
```

5、 测试服务器

测试一下，在/tftpboot 文件夹下新建立一个文件

```
$>touch abc
```

进入另外一个文件夹

```
$>tftp 192.168.1.15  (192.168.1.15 为本机 IP)
$>tftp> get abc
```

如果可以下载说明服务器已经安装成功。

附录四 WinCE 相关资源链接

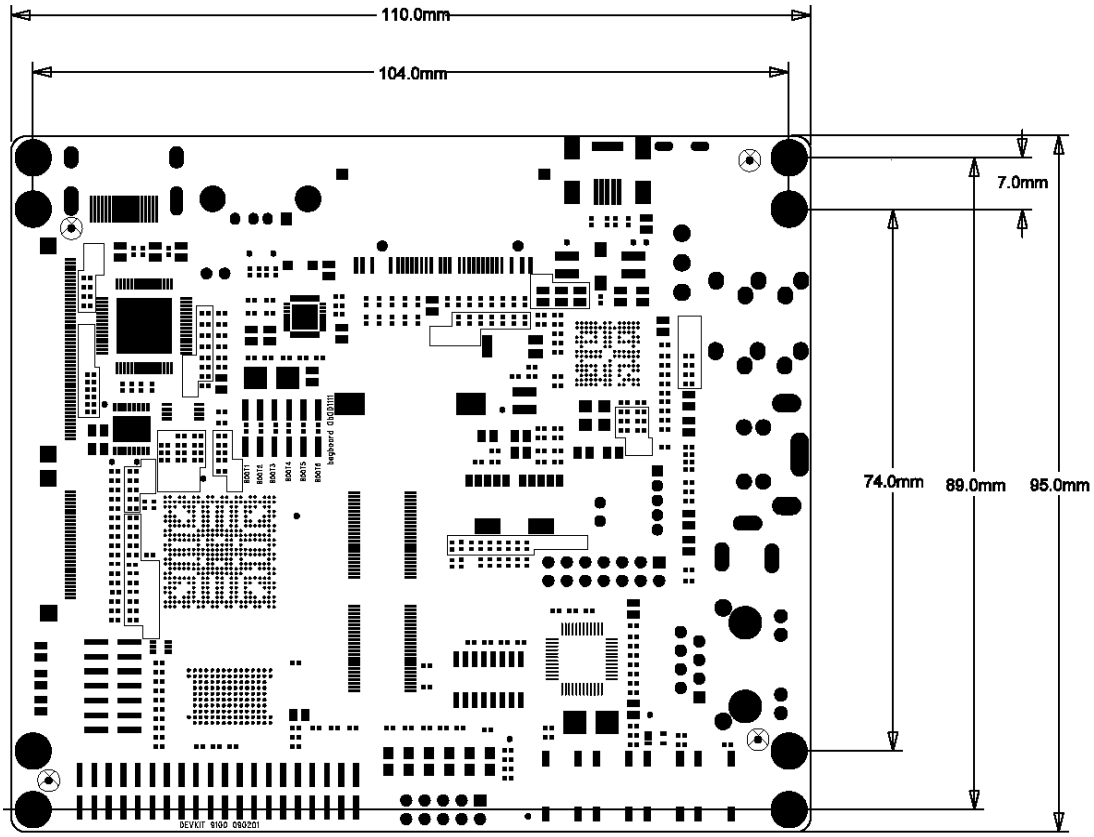
（1） Windows Embedded CE 6.0 Platform Builder Service Pack 1
http://www.microsoft.com/downloads/details.aspx?familyid=BF0DC0E3-8575-4860-A8E3-290ADF242678&displaylang=en

（2） Windows Embedded CE 6.0 R2
http://www.microsoft.com/downloads/details.aspx?FamilyID=f41fc7c1-f0f4-4fd6-9366-b61e0ab59565&displaylang=en

（3） Embedded CE 6.0 Platform Builder - Cumulative Product Update Rollup Package (through 12/31/2008)
http://www.microsoft.com/downloads/details.aspx?FamilyID=b478949e-d020-465e-b451-73127b30b79f&DisplayLang=en

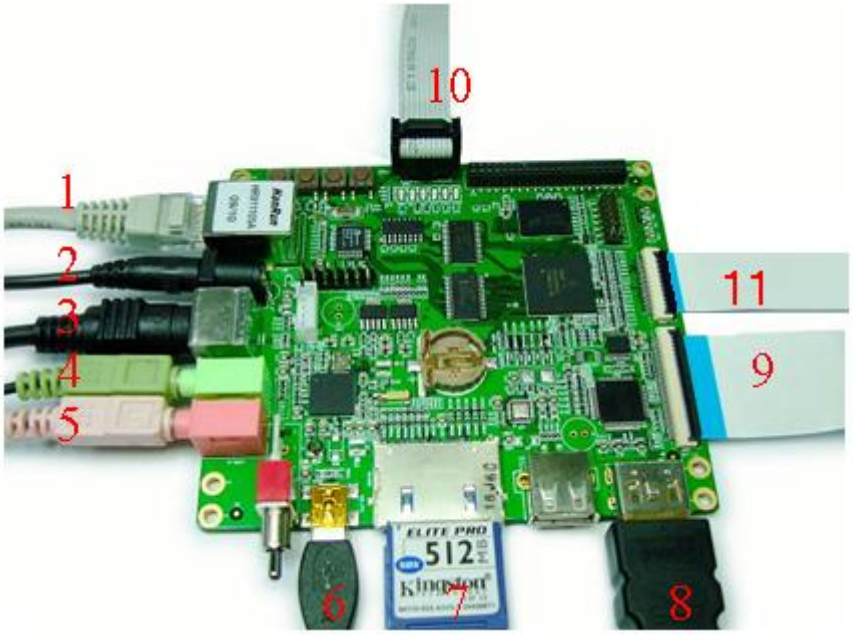
没有给出 Visual Studio 2005 下载链接

附录五 硬件尺寸图

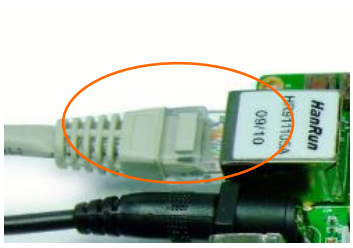


附录六 评估主板的外设连接图

DevKit8000 评估主板的外设连接图如下图所示。

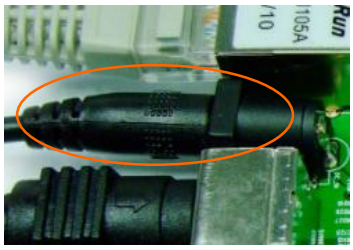


外设连接图



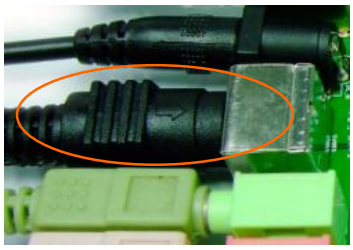
1 网络接口

使用网线一端连接主板上的 NET 接口，另一端连接到 PC 机或网络交换机上。



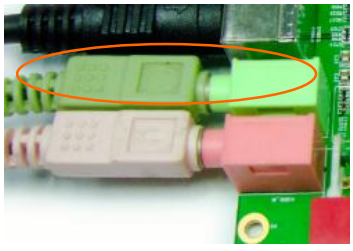
2 电源接口

使用 5V/2A 电源的电源线到主板的 DC_IN 接口。



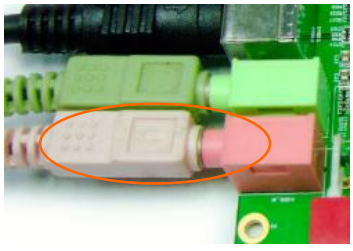
3 S-Video 接口

使用 S-Video 线连接主板上的 SVIDEO 接口，另一端连接 S-Video 设备。



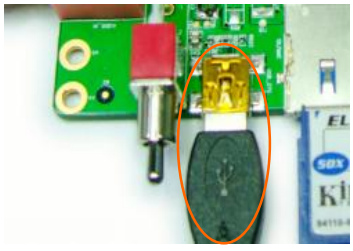
4 音频接口

直接连接音响插头到主板的 AUDIO_OUT 接口。



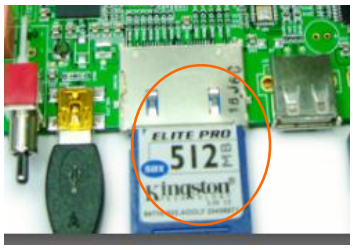
5 录音接口

直接连接麦克风插头到主板的 AUDIO_OUT 接口。



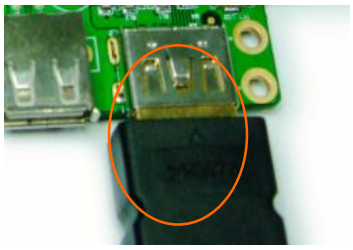
6 USB OTG 接口

连接 USB 转接线（Mini-A to A 或 Mini-B to A）到主板的 USB_OTG 接口，另一端连接相应设备即可。



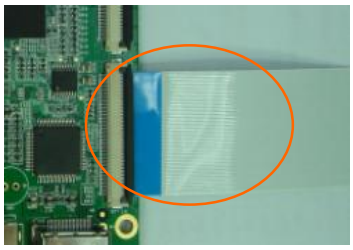
7 SD 卡接口

插入 SD 卡到主板的 SD/MMC 接口即可。



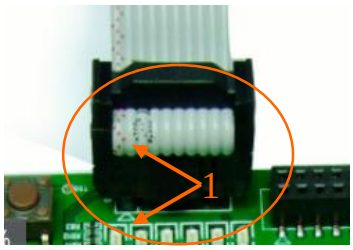
8 HDMI 接口

连接 HDMI 转 DVI 转接线到主板的 DVI_D 接口，另一端连接 DVI-D 设备即可。



9 LCD 接口

把 LCD 排线插入 TFT_LCD 接口，另一端连接 LCD 转接板。



10 串口

连接串口转接线的一端到主板的 UART3 接口(注意 1 脚的对应关系)，另一端直接插入 PC 机串口上。

11 Camera 模块接口

此 Camera 模块为可选配置，通过 30PIN 排线连接，可以通过屏幕输出摄像头信号。

附录七 FAQ 总结

请访问：http://elinux.org/DevKit8000_FAQ

技术支持和保修服务

技术支持服务

天漠科技对所销售的产品提供一年的免费技术支持服务，技术支持服务范围：

- 天漠嵌入式平台产品的软硬件资源
- 帮助您正确地编译与运行我们提供的源代码
- 按我们提供的产品文档，操作天漠嵌入式软硬件平台出现异常问题的
- 判定是否存在产品故障

特别说明，以下情况不在我们的免费技术支持服务范围，我们将根据情况酌情处理：

- 用户自行开发中遇到的软硬件问题
- 用户自行裁减编译运行嵌入式操作系统遇到的问题
- 用户自己的应用程序
- 修改我们的软件代码遇到问题的

保修服务条款

1. 本产品自出售之日起，于正常使用状况下，实行以下联保期限：
   印刷电路板卡：提供 12 个月免费保修服务；
2. 以下情况不属于免费服务范围，天漠将酌情收取服务费用：

   A. 无法提供产品有效购买凭证、产品识别标签撕毁或无法辨认，涂改标签或标签与实际产品不符； B. 未按用户手册操作所致导致损坏产品的；
   C. 因天灾（ 水灾、火灾、地震、雷击、台风等 ）或零件之自然耗损或遇不可抗拒力导致的产品外观及功能损坏；
   D. 因供电、磕碰、房屋漏水、动物、潮湿、杂 / 异物进入板内等原因导致的产品外观及功能损坏；
   E. 用户擅自拆焊零件或修改而导致不良或授权非天漠认可的人员及机构进行产品的拆装、维修，变更产品出厂规格及配置或扩充非天漠公司销售或认可的配件及由此引致的产品外观及功能损坏；
   F. 用户自行安装软件、系统或软件设定不当或由电脑病毒等造成的故障；
   G. 非经授权渠道购得此产品者。
   H. 非天漠对用户做出的超出保修服务范围的承诺（包括口头及书面等）由承诺方负责兑现，天漠恕不承担任何责任；
3. 保修期内由用户发到我们公司的运费由用户承担，由我们公司发给用户的运费由我们承担；保修期外的全部运输费用由用户承担。
4. 若板卡需要维修，请联系技术支持服务部。

备注：未经我司技术支持许可私自将产品寄回的，天漠公司概不负任何责任

液晶屏幕基本使用保养知识

1. 请勿用手指甲及尖锐的物品（硬物）碰触液晶屏表面以免刮伤，否则恕不能享受上述服务。
2. 液晶屏幕表面会因静电而吸附灰尘，建议购买液晶屏幕专用擦拭布来清洁您的屏幕，请勿用手指拍除以免留下指纹，并请轻轻擦拭。
3. 请勿使用化学清洁剂擦拭液晶表面，否则恕不能享受上述服务。

增值服务

天漠科技还为用户提供以下增值技术服务：

- 提供基于天漠科技嵌入式平台的驱动开发服务，如串口、USB 接口设备、LCD 液晶屏
- 提供操作系统移植、BSP 驱动开发、应用软件开发等服务
- 其他增值服务，包括机箱、电源、LCD 配套等
- 其他 OEM/ODM 服务
- 技术培训

用户可以联络 Timll 直接获得技术支持：

- 支持热线：+86-755-25503401
- 传真号码：+86-755-25616057
- 售前咨询：sales@timll.com
- 售后支持：support@timll.com