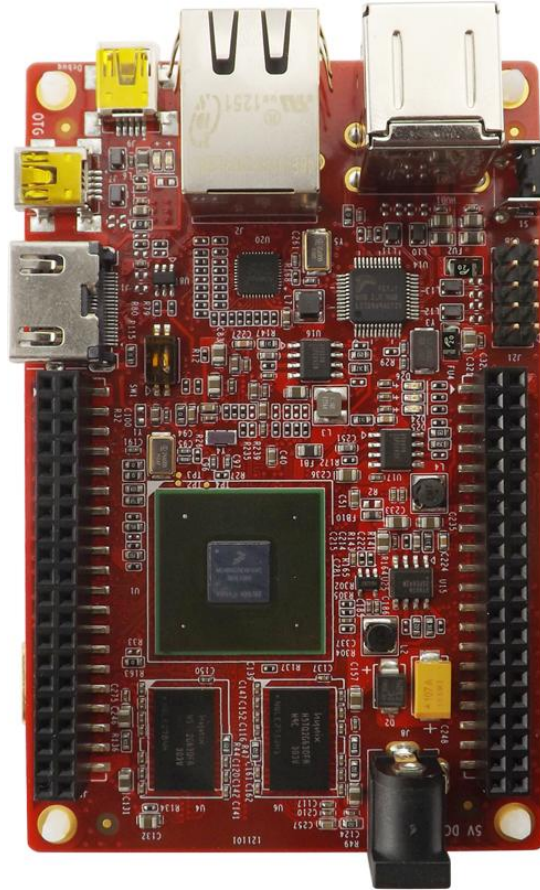


MarS Board 评估板



用户手册

版本 1.1 – 2013 年 5 月 11 日

版权声明：

- MarS Board 评估板及其相关知识产权由深圳市英蓓特科技有限公司所有。
- 本文档由深圳市英蓓特科技有限公司版权所有，并保留一切权利。在未经英蓓特公司书面许可的情况下，不得以任何方式或形式来修改、分发或复制本文档的任何部分。
- Microsoft, MS-DOS, Windows, Windows95, Windows98, Windows2000, Windows xp, Windows Embedded Compact 7 由微软公司授权使用。

版本更新记录：

版本	更新日期	描述
1.0	2013-3-29	初始版本
1.1	2013-5-11	文档修订

目录

第 1 章 产品概述.....	1
1.1 产品介绍.....	1
1.2 包装内容.....	1
1.3 产品特性.....	2
1.4 系统框图.....	4
1.5 硬件尺寸（毫米）	5
第 2 章 硬件系统介绍	6
2.1 CPU 简介	6
2.1.1 时钟.....	6
2.1.2 复位信号	6
2.1.3 通用接口	6
2.1.4 显示接口	6
2.1.5 3D 图形加速系统	7
2.2 CPU 周边芯片	7
2.2.1 eMMC Flash 存储器 NCEMBM11-04G	7
2.2.2 DDR 存储器 H5TQ2G63DFR-H9C	7
2.2.3 AR8035 以太网 PHY	7
2.2.4 FE1.1 USB 集线器.....	8
2.2.5 FT232RQ USB 转 UART 芯片.....	8
2.3 Mars Board 硬件接口.....	8
2.3.1 电源接口（J8）	9
2.3.2 HDMI 接口（J1）	9
2.3.3 LVDS 接口（J3）	10
2.3.4 USB OTG 接口（J7）	10
2.3.5 USB Debug 接口（J9）	11
2.3.6 以太网接口（J2）	11
2.3.7 USB Hub 接口（Hub1）	11
2.3.8 USB Hub 扩展接口（J21）	12
2.3.9 TF 卡接口（J13）	12
2.3.10 LCD 接口（J12）	13
2.3.11 AUDMUX（数字音频复用器）接口（J11）	14

2.3.12 CAN1 接口 (J11)	14
2.3.13 CAN2 接口 (J11)	15
2.3.14 ECSPi2 (增强可配置 SPI) 接口 (J10)	15
2.3.15 I2C1 接口 (J11)	15
2.3.16 I2C3 接口 (J11)	15
2.3.17 IPU1 (图形处理单元 1) 接口 (J11)	16
2.3.18 KPP 键盘接口 (J11)	16
2.3.19 PWM (脉宽调制) 接口 (J10 和 J11)	16
2.3.20 GPi 通用存储器接口 (J10)	17
2.3.21 SPDIF (Sony/Philips 数字接口) (J10)	17
2.3.22 UART1 接口 (J11)	17
2.3.23 UART3 接口 (J10)	18
2.3.24 UART4 接口 (J11)	18
2.3.25 UART5 接口 (J11)	18
2.3.26 USDHC1 接口 (J10)	18
2.3.27 ESAI (增强串行音频接口) (J10 和 J11)	19
第 3 章 准备工作.....	20
3.1 软件简介.....	20
3.2 关于 Linux 系统.....	21
3.3 关于 Android 系统.....	22
3.4 设置超级终端.....	23
第 4 章 下载和运行系统.....	24
4.1 Linux 和 Android 系统下载及运行.....	24
4.2 UcoS 系统演示.....	27
4.3 显示方式设置.....	27
第 5 章 制作映像文件.....	30
5.1 为 Linux 系统制作映像文件.....	30
5.1.1 获取工具和源代码.....	30
5.1.2 编译映像文件.....	31
5.2 为 Android 系统制作映像文件.....	32
5.2.1 获取 repo 源代码.....	32
5.2.2 编译映像文件.....	32

附录 1 安装 Ubuntu Linux 系统	34
技术支持和保修服务	45

第1章 产品概述

1.1 产品介绍

MarS Board 是深圳市英蓓特科技有限公司推出的基于飞思卡尔（Freescall Semiconductor）i.MX 6Dual 处理器的评估板。i.MX 6Dual 处理器集成了高达 1GHz 的 ARM Cortex™-A9 内核、2D 和 3D 图形处理器和 3D 1080p 视频处理器。MarS Board 评估板具有丰富的接口,包括 HDMI 接口、LVDS 接口、Mini USB OTG 接口、Mini USB Debug 接口、RJ45 接口、USB Host 接口、TF 卡接口和 LCD 显示接口,能够帮助开发者针对上网本、桌面一体机、高端移动互联网设备、高端掌上电脑、高端便携式媒体播放器、游戏机和便携式导航设备等各种不同领域进行开发。

1.2 包装内容


- MarS Board 开发板
- 配件包（选配）：
 - HDMI 线
 - Mini USB 线
 - 5V@4A 电源适配器
 - 4GB TF 卡
 - 千兆网线
- 其他自选配件

1.3 产品特性

- **产品参数:**
 - 产品尺寸: 65mm x 102mm
 - 工作温度: 0 ~ 70℃
 - 环境湿度: 20% ~ 90% (无凝结)
 - 输入电源: 5V
- **处理器:**
 - 集成 ARM Cortex™-A9 内核的 i.MX 6Dual 处理器
 - 集成 32 KByte 一级指令缓存
 - 集成 32 KByte 以及数据缓存
 - 集成私有计数器和看门口
 - 集成 Cortex-A9 NEON MPE (媒体处理引擎) 协处理器
 - 集成 2D 图形处理器
- **板载存储器:**
 - 4GByte eMMC 存储器
 - 4*256MB DDR3 SDRAM 存储器
- **板载接口和按钮:**
 - 一个 HDMI 接口
 - 一个 LVDS 接口
 - 一个 LCD 显示屏接口
 - 两个 480Mbps 高速 USB2.0 HubHUB 接口
 - 两个 480Mbps 高速 USB2.0 Header 接口
 - 一个 480Mbps 高速 USB2.0 OTG 接口
 - 一个 COM-USB Debug (com2)接口
 - 一个 TF 卡接口
 - 一个 10/100M/1Gbps RJ45 网络接口

- 一个 BOOT MODE 设置拨码开关
- 一个复位按钮
- 板载接口信号：
 - 一路 AUDMUX (Digital Audio Multiplexer) 信号
 - 两路 CAN 信号
 - 一路 ECSPi2 (Enhanced Configurable SPI) 信号
 - 两路 I2C 信号
 - 一路 8bit 或 10bit camera/ Parallel 信号
 - 一路 KPP (Keypad Port) 信号
 - 一路 PWM (Pulse Width Modulation) 信号
 - 一路 GPMI (General Purpose Memory Interface) 信号
 - 一路 SPDIF (Sony/Philips Digital Interface) 信号
 - 四路 UART 信号
 - 一路 USDHC1 (Ultra Secured Digital Host Controller) 信号
 - 一路 ESAI (Enhanced Serial Audio Interface) 信号

注意：

 以上接口存在部分管脚复用；详细情况请参考芯片手册和产品附带的原理图。

1.4 系统框图

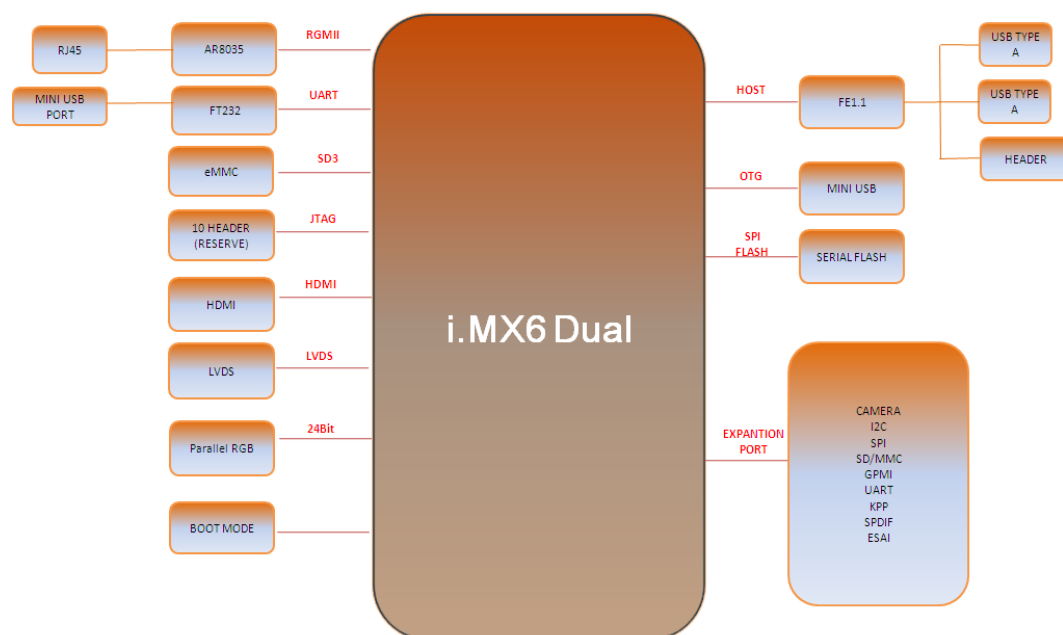


图 1-1 MarS Board 系统框图

1.5 硬件尺寸（毫米）

MarS Board Top Layers

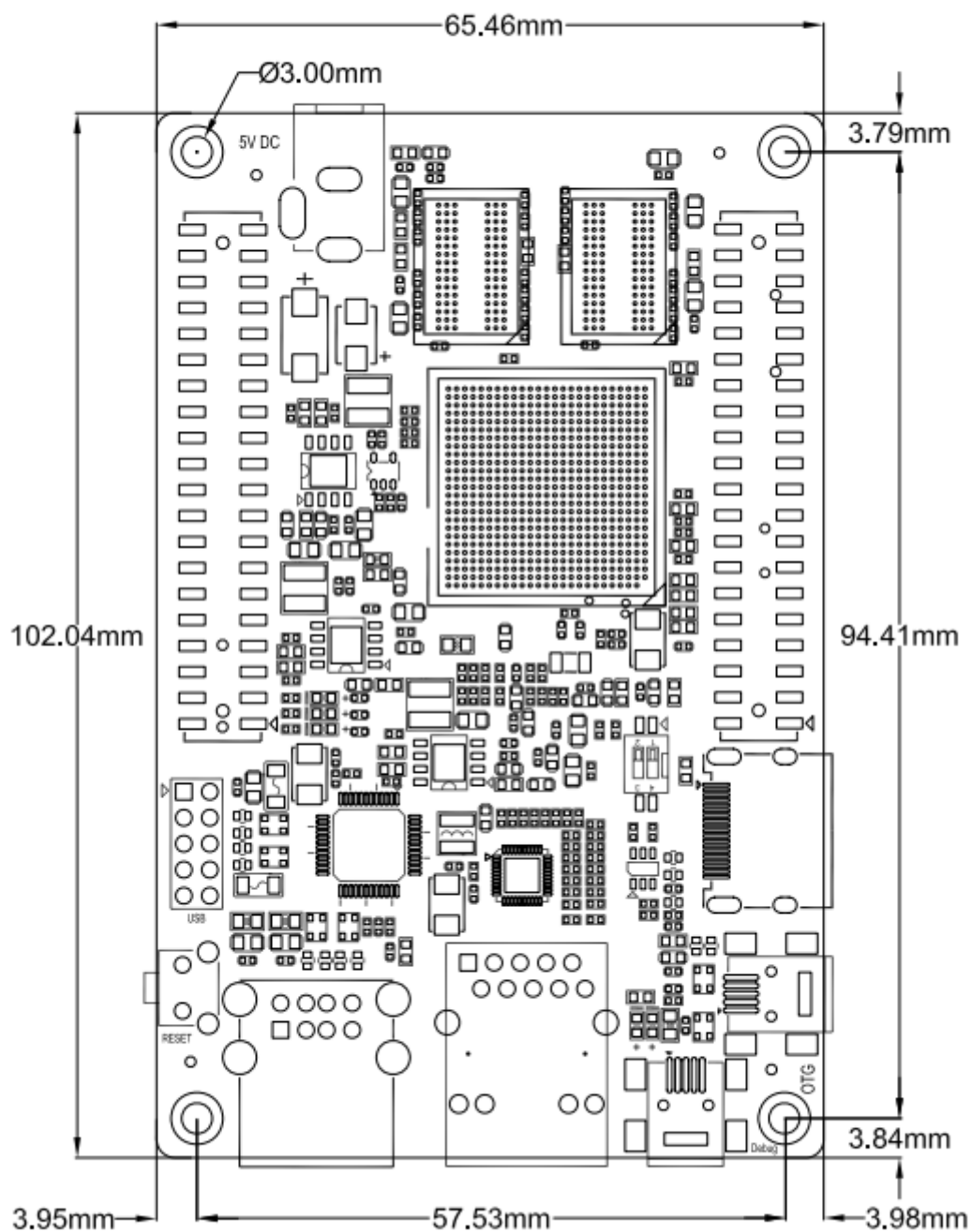


图 1-2 MarS Board 硬件尺寸

第2章 硬件系统介绍

本章节将通过简要介绍 MarS Board 评估板上所采用的 CPU、CPU 周边芯片、各种接口的管脚定义来让您对该硬件系统有一个大致的了解。

2.1 CPU 简介

Freescall 公司的 i.MX 6Dual 是基于 ARM™ Cortex-A9 内核的双核处理器，运行频率高达 1.0 GHz，集成了 2D 和 3D 图形处理器、3D 1080p 视频处理器和电源管理模块，并且提供 64 位 DDR3/LVDDR3/LVDDR2-1066 储存器接口以及包括高清显示和摄像头在内的许多其他接口。

2.1.1 时钟

i.MX 6Dual 的时钟信号包括一个 32.768KHz 的 RTC 时钟和一个 24MHz 的外频时钟；

- **RTC 时钟：**由外部无源晶振产生，用于低频率运算；
- **外频时钟：**用于产生设备的主时钟，以便提供给 PLL 和 CMM 等其他模组；

2.1.2 复位信号

复位信号由 CPU 的 POR_B 决定；低电平代表复位有效。

2.1.3 通用接口

通用接口设备包括 7 组通用输入输出接口 (GPIO)，每一个 GPIO 模组提供 32 个 (其中 GPIO7 提供 14 个) 专用的通用接口输入输出管脚，因此通用的 GPIO 可以拥有最多 206 个管脚。

2.1.4 显示接口

- 一路并行的 24bit RGB 接口，支持 60 Hz WUXGA 输出

- 两路 LVDS 接口，最大支持 165 Mpixels/sec 输出
- 一个 HDMI 1.4 接口
- 一个 MIPI/DSI 接口，输出速率 1 Gbps

2.1.5 3D 图形加速系统

i.MX 6Dual 集成了 GPU3Dv4 3D 图形处理单元，为 3D 图像算法提供了硬件加速，可以使桌面交互式图像应用最高达到 HD1080P 的分辨率，并且支持 OpenGL ES 2.0 标准及其扩展、OpenGL ES 1.1 和 OpenVG 1.1 标准。

另外 i.MX 6Dual 还通过其 GPUv2 向量图形处理单元为 2D 图像算法提供了硬件加速功能。

2.2 CPU 周边芯片

2.2.1 eMMC Flash 存储器 NCEMBM11-04G

NCEMBM11-04G 是 MarS Board 上的 eMMC Flash 存储器，大小为 4GB。该 flash 存储器支持时钟速率高达 52MHz 的高速 DDR 传输和三种数据线宽模式：1-bit（默认）、4-bit 和 8-bit；其同步电源管理技术使得芯片具有高速启动、自动终止和睡眠等特点；同时 NCEMBM11-04G 还支持高速双数据传输引导模式。

2.2.2 DDR 存储器 H5TQ2G63DFR-H9C

H5TQ2G63DFR-H9C 是 MarS Board 上的 DDR3 SDRAM 存储器，大小为 256MB。H5TQ2G63BFR-H9C 适用于大容量和高带宽需求的场合，支持差分时钟输入、差分数据选通、自动刷新以及异步管脚复位等功能。MarS Board 集成了 4 片 H5TQ2G63DFR-H9C，共计 1GB 容量。

2.2.3 AR8035 以太网 PHY

AR8035 是 MarS Board 上的低功耗、低成本、单端口 10/100/1000 Mbps 三速以太网 PHY。AR8035 支持 MAC.TM RGMII 接口，采用 Atheros 专有的 SmartEEE 技术来支持

IEEE 802.3az 高效节能以太网（EEE）标准，让不支持 802.3az 标准的传统 MAC/SoC 设备具有更高能效。MarS Board 可通过直通网线连接到网络集线器上，也可用交叉网线与电脑直接相连。

2.2.4 FE1.1 USB 集线器

F1.1 是一个四端口输出的 USB HUB 2.0 控制器，该芯片为 MarS Board 引出了四路 USB HUB 信号，其中两路信号连接到 USB 插座，另外两路信号连接到 J21 接口。

2.2.5 FT232RQ USB 转 UART 芯片

FT232RQ 是一个 USB 转串口 UART 芯片，该芯片为 MarS Board 提供了 Mini USB 调试接口。芯片内部集成了 1024-bit 的 EEPROM 存储设备和 CBUS I/O 配置器，支持 TTL 电平 300 baud~3 Mbaud 的数据传输速率。

2.3 Mars Board 硬件接口

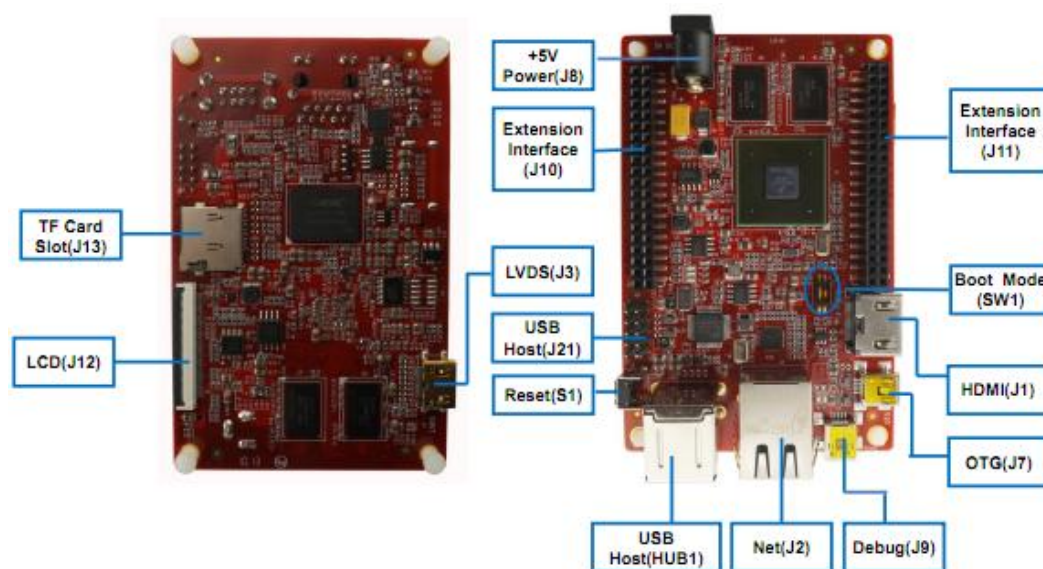


图 2-1 MarS Board 硬件接口

2.3.1 电源接口（J8）

表 2-1 电源接口

管脚	信号定义	描述
1	GND	GND
2	+5V	Power supply (+5V) 4A (Type)
3	+5V	Power supply (+5V) 4A (Type)

2.3.2 HDMI 接口（J1）

表 2-2 HDMI 接口

管脚	信号定义	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	NC	NC
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

2.3.3 LVDS 接口（J3）

表 2-3 LVDS 接口

管脚	信号定义	描述
1	3V3	+3.3V
2	LVDS_TX2_P	LVDS Data2+
3	LVDS_TX2_N	LVDS Data2-
4	GND	GND
5	LVDS_TX1_P	LVDS Data1+
6	LVDS_TX1_N	LVDS Data1-
7	GND	GND
8	LVDS_TX0_P	LVDS Data0+
9	LVDS_TX0_N	LVDS Data-
10	GND	GND
11	LVDS_CLK_P	LVDS_CLK+
12	LVDS_CLK_N	LVDS_CLK-
13	LCD_PWR_EN	AC bias control (STN) or pixel data enable (TFT)
14	Touch_Int	Touch interrupt signal
15	I2C_SCL	IIC master serial clock
16	I2C_SDA	IIC master serial data
17	LED_PWR_EN	Backlight enable
18	5V	+5V
19	PWM	Pulse Width Modulation

2.3.4 USB OTG 接口（J7）

表 2-4 USB OTG 接口

管脚	信号定义	描述
1	VBUS	+5V
2	DN	USB Data-
3	DP	USB Data+
4	ID	USB ID
5	GND	GND

2.3.5 USB Debug 接口（J9）

表 2-5 USB Debug 接口

管脚	信号定义	描述
1	VBUS	+5V
2	DN	USB Debug Data-
3	DP	USB Debug Data+
4	NC	NC
5	GND	GND

2.3.6 以太网接口（J2）

表 2-6 以太网接口

管脚	信号定义	描述
1	TD1+	TD1+ output
2	TD1-	TD1- output
3	TD2+	TD2+ output
4	TD2-	TD2- output
5	TCT	2.5V Power for TD
6	RCT	2.5V Power for RD
7	RD1+	RD1+ input
8	RD1-	RD1- input
9	RD2+	RD2+ input
10	RD2-	RD2- input
11	GRLA	Green LED link signal
12	GRLC	Power supply for Green LED
13	YELC	Yellow LED action signal
14	YELA	Power supply for Yellow LED

2.3.7 USB Hub 接口（Hub1）

表 2-7 USB Hub 接口

管脚	信号定义	描述
1	APV	5V power for HUB A
2	AD-	USB HUB A Data-
3	AD+	USB Debug Data+
4	GNDA	USB HUB A GND

管脚	信号定义	描述
5	BPV	5V power for HUB B
6	BD-	USB HUB B Data-
7	BD+	USB HUB B Data+
8	GNDB	USB HUB B GND

2.3.8 USB Hub 扩展接口（J21）

表 2-8 USB Hub 扩展接口

管脚	信号定义	描述
1	PWR2	5V power for HUB 2
2	PWR1	5V power for HUB 1
3	DM2	USB HUB 2 Data-
4	DM1	USB HUB 1 Data-
5	DP2	USB HUB 2 Data+
6	DP1	USB HUB 1 Data+
7	GND	GND
8	GND	GND
9	GND	GND
10	GND	GND

2.3.9 TF 卡接口（J13）

表 2-9 TF 卡接口

管脚	信号定义	描述
1	DAT2	Card data 2
2	DAT3	Card data 3
3	CMD	Command Signal
4	VDD	VDD
5	CLK	Clock
6	VSS	VSS
7	DAT0	Card data 0
8	DAT1	Card data 1
9	CD	Card detect

2.3.10 LCD 接口 (J12)

表 2-10 LCD 接口

管脚	信号定义	描述
1	B0	GND
2	B1	GND
3	B2	GND
4	B3	LCD Pixel data bit 0
5	B4	LCD Pixel data bit 1
6	B5	LCD Pixel data bit 2
7	B6	LCD Pixel data bit 3
8	B7	LCD Pixel data bit 4
9	GND1	GND
10	G0	GND
11	G1	GND
12	G2	LCD Pixel data bit 5
13	G3	LCD Pixel data bit 6
14	G4	LCD Pixel data bit 7
15	G5	LCD Pixel data bit 8
16	G6	LCD Pixel data bit 9
17	G7	LCD Pixel data bit 10
18	GND2	GND
19	R0	GND
20	R1	GND
21	R2	GND
22	R3	LCD Pixel data bit 11
23	R4	LCD Pixel data bit 12
24	R5	LCD Pixel data bit 13
25	R6	LCD Pixel data bit 14
26	R7	LCD Pixel data bit 15
27	GND3	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	GND4	GND
34	X+	X+ Position Input
35	X-	X- Position Input
36	Y+	Y+ Position Input

管脚	信号定义	描述
37	Y-	Y- Position Input
38	SPI_CLK	SPI serial clock
39	SPI_MOSI	SPI Master Output, Slave Input
40	SPI_MISO	SPI Master Input, Slave Output
41	SPI_CS	SPI Chip Select
42	IIC_CLK	IIC master serial clock
43	IIC_DAT	IIC serial bidirectional data
44	GND5	GND
45	VDD1	3.3V
46	VDD2	3.3V
47	VDD3	5V
48	VDD4	5V
49	RESET	Reset
50	PWREN	Backlight enable

注意:

请勿热插拔 LCD 排线。

2.3.11 AUDMUX（数字音频复用器）接口（J11）

表 2-11 AUDMUX 接口

管脚	信号定义	描述
31	AUD3_RXD	Receive audio data
25	AUD3_TXC	Audio transmission clock
27	AUD3_TXD	Transmit audio data
29	AUD3_TXFS	Transmit audio frame signal

2.3.12 CAN1 接口（J11）

表 2-12 CAN1 接口

管脚	信号定义	描述
33	RXCAN	Receive data
35	TXCAN	Transmit data

2.3.13 CAN2 接口 (J11)

表 2-13 CAN2 接口

管脚	信号定义	描述
37	RXCAN	Receive data
39	TXCAN	Transmit data

2.3.14 ECSPi2 (增强可配置 SPI) 接口 (J10)

表 2-14 ECSPi2 接口

管脚	信号定义	描述
21	MISO	Master Input Slave Output
19	MOSI	Master Output Slave Input
17	SCLK	Clock
15	SS0	Chip select

2.3.15 I2C1 接口 (J11)

表 2-15 I2C1 接口

管脚	信号定义	描述
38	SCL	Master serial clock
40	SDA	Master serial data

2.3.16 I2C3 接口 (J11)

表 2-16 I2C3 接口

管脚	信号定义	描述
3	SCL	Master serial clock
5	SDA	Master serial data

2.3.17 IPU1（图形处理单元 1）接口（J11）

表 2-17 IPU1 接口

管脚	信号定义	描述
4	CSI0_DAT12	Digital image data bit 12
6	CSI0_DAT13	Digital image data bit 13
8	CSI0_DAT14	Digital image data bit 14
10	CSI0_DAT15	Digital image data bit 15
12	CSI0_DAT16	Digital image data bit 16
14	CSI0_DAT17	Digital image data bit 17
16	CSI0_DAT18	Digital image data bit 18
18	CSI0_DAT19	Digital image data bit 19
21	CSI0_DATA_EN	Digital image data write enable
17	CSI0_HSYNC	Horizontal synchronization
19	CSI0_PIXCLK	Pixel clock
23	CSI0_VSYNC	Vertical synchronization

2.3.18 KPP 键盘接口（J11）

表 2-18 KPP 接口

管脚	信号定义	描述
30	COL[0]	Keypad matrix column 0 output
34	COL[1]	Keypad matrix column 1 output
35	COL[2]	Keypad matrix column 2 output
28	ROW[0]	Keypad matrix row 0 input
32	ROW[1]	Keypad matrix row 1 input
37	ROW[2]	Keypad matrix row 1 input

2.3.19 PWM（脉宽调制）接口（J10 和 J11）

表 2-19 PWM 接口

管脚	信号定义	描述
26(J11)	PWM1	Pulse Width Modulation
13(J10)	PWM4	Pulse Width Modulation

2.3.20 GPMI 通用存储器接口（J10）

表 2-20 GPMI 接口

管脚	信号定义	描述
6	ALE	Address Latch Enable
4	CE0N	CHIP ENABLE
3	CLE	Command Latch Enable
14	D0	Data 0
16	D1	Data 1
18	D2	Data 2
20	D3	Data 3
22	D4	Data 4
24	D5	Data 5
26	D6	Data 6
28	D7	Data 7
34	DQS	Data Strobe Control
32	RDN	Read Enable
12	READY0	Ready Busy
10	WP	Write Protect
30	WRN	Write Enable

2.3.21 SPDIF（Sony/Philips 数字接口）（J10）

表 2-21 SPDIF 接口

管脚	信号定义	描述
25	IN1	I2S data Input
23	OUT1	I2S data output
29	PLOCK	System master clock
27	SPDIF_EXTCLK	I2S frame clock
31	SRCLK	I2S bit clock

2.3.22 UART1 接口（J11）

表 2-22 UART1 接口

管脚	信号定义	描述
7	CTS	Clear To Send
9	RTS	Request To Send

管脚	信号定义	描述
13	RXD_MUX	Receive data
11	TXD_MUX	Transmit data

2.3.23 UART3 接口 (J10)

表 2-23 UART3 接口

管脚	信号定义	描述
33	CTS	Clear To Send
35	RTS	Request To Send
36	RXD_MUX	Receive data
38	TXD_MUX	Transmit data

2.3.24 UART4 接口 (J11)

表 2-24 UART4 接口

管脚	信号定义	描述
28	RXD_MUX	Receive data
30	TXD_MUX	Transmit data

2.3.25 UART5 接口 (J11)

表 2-25 UART5 接口

管脚	信号定义	描述
32	RXD_MUX	Receive data
34	TXD_MUX	Transmit data

2.3.26 USDHC1 接口 (J10)

表 2-26 USDHC1 接口

管脚	信号定义	描述
39	CD	Card detect
3	CLK	Card clock
1	CMD	Command Signal

管脚	信号定义	描述
5	DAT0	Card data 0
7	DAT1	Card data 1
9	DAT2	Card data 2
11	DAT3	Card data 3

2.3.27 ESAI（增强串行音频接口）（J10 和 J11）

表 2-27 ESAI 接口

管脚	信号定义	描述
26(J11)	FSR	Frame Sync for Receiver
15(J11)	FST	Frame Sync for Transmitter
22(J11)	HCKR	High Frequency Clock for Receiver
23(J10)	HCKT	High Frequency Clock for Transmitter
39(J10)	SCKR	Receiver Serial Clock
27(J10)	SCKT	Transmitter Serial Clock
24(J11)	TX0	Serial output 0
20(J11)	TX1	Serial output 1
3(J11)	TX2_RX3	Serial output 2_Serial Input 3
25(J10)	TX3_RX2	Serial output 3_Serial Input 2
29(J10)	TX4_RX1	Serial output 4_Serial Input 1
31(J10)	TX5_RX0	Serial output 5_Serial Input 0

第3章 准备工作

在开始使用 MarS Board 之前，请阅读以下内容，以便熟悉开发过程中涉及的系统映像、驱动代码和工具等。

3.1 软件简介

以下列表列出了后面将会用到的 Linux 和 Android 两个版本的操作系统，以及各种驱动代码。

表 3-1 操作系统和驱动

分类	注释
操作系统	Linux 版本 3.0.15
	Android 版本 4.0.4
设备驱动	Serial 串行接口驱动
	RTC 硬件时钟驱动
	Net 10/100/Gb IEEE1588 以太网
	Flash Spi flash 驱动
	Display 三个显示接口（RGB, LVDS 和 HDMI 1.4a）
	mmc/sd 一个 SD 3.0/SDXC 插槽和 eMMC
	USB 三个高速 USB 接口（2xHost, 1xOTG）
	Audio 数字音频（HDMI）
	LED 用户 LED 驱动

3.2 关于 Linux 系统

以下列表列出了建立 Linux 系统需要的映像文件和 eMMC 存储器分区。（X 为 ISO 文件名）

表 3-2 Linux 映像

映像	路径
u-boot 映像	X:/linux/image/u-boot.bin
内核映像	X:/linux/image/ulmage
Ubuntu 文件系统	X:/linux/image/oneiric.tgz

表 3-3 Linux 存储分区

分区类型/索引	名称	起始偏移量	容量	文件系统	内容
N/A	BOOT Loader	0	1MB	N/A	bootloader
N/A	Kernel	1M	9MB	N/A	ulmage
Primary 1	Rootfs	10M	Total - Other	EXT3	oneiric.tgz

- **分区类型/索引**：定义保存在 MBR 中。
- **名称**：只针对 Android 系统才具有实际意义；您可以在建立 Linux 分区时忽略该内容。
- **起始偏移量**：分区起始位置，单位 MB。

3.3 关于 Android 系统

以下列表列出了建立 Android 系统需要的映像文件和 eMMC 存储器分区。(X 为 ISO 文件名)

表 3-4 Android 映像

映像	路径
u-boot 映像	X:/android/image/u-boot.bin
启动映像	X:/android/image/boot.img
Android 系统根映像	X:/android/image/system.img
Recovery 根映像	X:/android/image/recovery.img

表 3-5 Android 存储分区

分区类型/索引	名称	起始偏移量	容量	文件系统	内容
N/A	BOOT Loader	0	1MB	N/A	bootloader
主分区 1	Boot	8M	8MB	boot.img format, a kernel + ramdisk	boot.img
主分区 2	Recovery	Boot 分区后	8MB	boot.img format, a kernel + ramdisk	recovery.img
逻辑分区 5 (扩展分区 3)	SYSTEM	Recovery 分区后	512MB	EXT4. Mount as /system	Android 系统文件 /system/dir
逻辑分区 6 (扩展分区 3)	CACHE	SYSTEM 分区后	256MB	EXT4. Mount as /cache	Android 缓存, 用于 OTA 映像保存
逻辑分区 7 (扩展分区 3)	DATA	CACHE 分区后	> 1024MB	EXT4 Mount at /data	系统应用程序数据
逻辑分区 8 (扩展分区 3)	Vendor	DATA 分区后	8MB	Ext4 Mount at /vender	保存 MAC 地址文件
逻辑分区 9 (扩展分区 3)	Misc	DATA 分区后	4M	N/A	恢复和保存 bootloader 信息
主分区 4	MEDIA	Misc 分区后	Total - Other images	VFAT	内部媒体分区 /mnt/sdcard/dir

- **系统分区**：用于保存 Android 系统映像。
- **数据分区**：用于保存应用程序的解压数据和系统配置数据库等。

在正常模式下，根文件系统会从 uramdisk 进行挂载。而 recovery 模式下，根文件系统会从 RECOVERY 分区进行挂载。

3.4 设置超级终端

使用一根 Mini USB 线连接 MarS Board 的 Debug 接口和 PC 的 USB Host，然后在 PC 桌面选择开始 > 程序 > 附件 > 通讯 > 超级终端，根据下图中的参数进行设置：

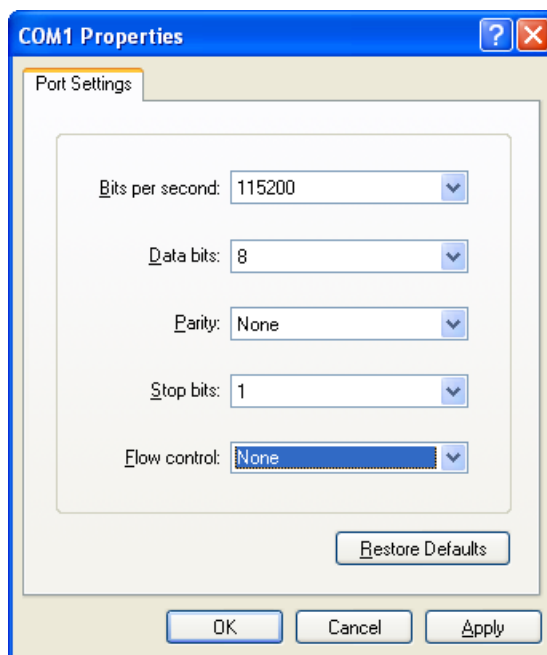


图 3-1 设置超级终端

第4章 下载和运行系统

现在您可以开始将 ISO 文件中已有的系统映像文件（请参考表 3-2 和表 3-4 中的映像文件路径）下载到 MarS Board 上并运行系统了。本章将通过 ISO 文件目录 **tools** 下的 MFG tool 来进行映像下载。

注意：

- 📖 在使用 MFG tool 下载映像时，请将 TF 卡从 MarS Board 上移除。
- 📖 每一条命令前都加上了符号“•”，以防止由于指令较长占用多行而造成误解。

4.1 Linux 和 Android 系统下载及运行

- 1) 将 ISO 中所有文件复制到您的 PC 硬盘的根目录下（假设为 C:\ 目录）；
- 2) 用一根 Mini USB 线将 MarS Board 的 USB OTG 接口连接到 PC 的 USB Host 口；
- 3) 将启动配置开关 SW1 按照下表中的参数设置为 MFG 工具模式；

表 4-1 启动开关设置

开关	D1	D2
状态	关	开

- 4) 运行 C:\tools\目录下的 MFG tool 并连接 MarS Board 的电源，软件界面如下图所示；（如果是第一次连接基于 i MX6 处理器的产品，PC 会自动安装 HID 驱动）



图 4-1 MFG tool 界面

请单击**扫描设备**按钮来自动检测端口；

- 5) 在菜单栏中选择 **Options > Configuration** 进入配置窗口，如下图所示；

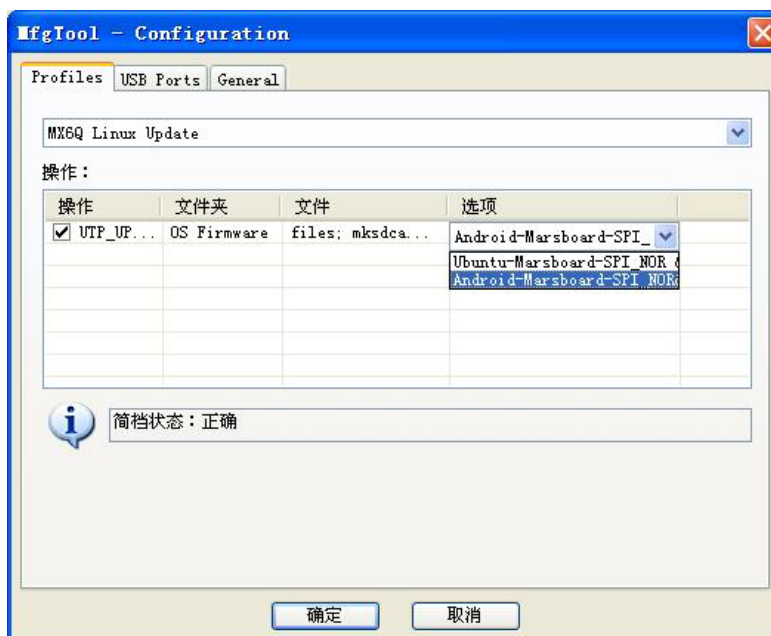


图 4-2 配置窗口

在 **Profiles** 标签下的**操作**区域内，单击**选项**一栏中的下载菜单；

- 针对 Linux 系统，请选择 **Ubuntu-Marsboard-SPI_NOR & eMMC** 选项；

- 针对 Android 系统，请选择 **Android-Marsboard-SPI_NOR&eMMC** 选项；

配置完成后单击**确定**；

- 6) 在以下窗口中单击**开始**按钮，然后在下载完成后单击**停止**按钮完成下载；



图 4-3 单击开始

- 7) 断开 Mars Board 的电源，并将启动配置开关 SW1 按照下表中的参数设置 SPI-NOR 启动模式；

表 4-2 启动开关设置

开关	D1	D2
状态	关	关

设置完成后接通电源即可启动系统。如需查看串口信息，请用一根 Mini USB 线连接 MarS Board 的 Debug 接口和 PC 的 USB Host，然后打开超级终端。

注意：

- 请勿在使用 MFG tool 下载映像时插入 TF 卡。
- u-boot 所需要的参数保存在 SPI-NOR flash 中，您可以在超级终端窗口中执行 **sf probe 0** 和 **sf erase 0xc0000 0x2000** 两条命令来重置参数。

4.2 UcoS 系统演示

UcoS 仅为 Demo，暂不提供相关源码

UcoS 下载及运行步骤如下：

- 1) 将 Linux/demo/ucos 目录下的 u-boot.bin 、 ulmage 文件覆盖
C:/tools/Mfgtools-Rel-12.04.01_ER_MX6Q_UPDATER \Profiles\MX6Q
Linux Update\OS Firmware\files\目录下的同名文件
- 2) 按照 4.1 节 Linux 系统下载步骤将映像下载到板子上
- 3) 拷贝 Linux/demo/ucos 目录下的所有文件到 TF 卡,将 TF 卡插入 MarS Board
的 TF 卡座子,连接 7 寸屏及 HDMI 电视,上电启动,成功启动后 7 寸屏将显
示 UCOS 系统,而 HDMI 则会显示 Ubuntu 系统

4.3 显示方式设置

MarS Board Linux 和 Android 支持多种显示模式输出,用户可通过设置 uboot 参数的方法选择不同的显示输出模式进入 uboot 的方法如下所示(以 Android 为例):

```
U-Boot 2009.08-svn1 (Mar 14 2013 - 14:07:49)

CPU: Freescale i.MX6 family TO0.0 at 792 MHz
Temperature: 51 C, calibration data 0x58150469
mx6q pll1: 792MHz
mx6q pll2: 528MHz
mx6q pll3: 480MHz
mx6q pll8: 50MHz
ipg clock      : 660000000Hz
ipg per clock : 660000000Hz
uart clock    : 800000000Hz
cspi clock    : 600000000Hz
ahb clock     : 1320000000Hz
axi clock     : 2640000000Hz
emi_slow clock: 293333333Hz
ddr clock     : 5280000000Hz
usdhc1 clock  : 1980000000Hz
```



```

usdhc2 clock : 198000000Hz
usdhc3 clock : 198000000Hz
usdhc4 clock : 198000000Hz
nfc clock    : 24000000Hz
Board: MX6Q-MARSBOARD:[ POR]
Boot Device: I2C
I2C:  ready
DRAM:  1 GB
MMC:   FSL_USDHC: 0,FSL_USDHC: 1
JEDEC ID: 0xbf:0x25:0x41
Reading SPI NOR flash 0xc0000 [0x2000 bytes] -> ram 0x276009b8
SUCCESS

*** Warning - bad CRC, using default environment

In:  serial
Out: serial
Err: serial
Net:  got MAC address from IIM: 00:00:00:00:00:00
----enet_board_init: phy reset
FEC0 [PRIME]
Hit any key to stop autoboot:  0  (按 PC 键盘的任意键进入 uboot)
MX6Q MARSBOARD U-Boot >

```

1) 使用 4.3" LCD 显示

在 u-boot 模式下执行以下命令来配置 4.3 寸 LCD 显示模式;

- MX6Q MARSBOARD U-Boot > **setenv bootargs console=ttymxc1,115200 init=/init rw video=mxcfb0:dev=lcd,4.3inch_LCD,if=RGB24 fbmem=10M vmalloc=400M androidboot.console=ttymxc1 calibration**

2) 使用 7" LCD 显示

在 u-boot 模式下执行以下命令来配置 7 寸 LCD 显示模式

- MX6Q MARSBOARD U-Boot > **setenv bootargs console=ttymxc1,115200 init=/init rw video=mxcfb0:dev=lcd,7inch_LCD,if=RGB24 fbmem=10M vmalloc=400M androidboot.console=ttymxc1 calibration**

3) 使用 HDMI 显示

在 u-boot 模式下执行以下命令来配置 HDMI 显示模式

- MX6Q MARSBOARD U-Boot > **setenv bootargs console=ttymxc1,115200 init=/init rw video=mxcfb0:dev=hdm,1920x1080M@60,if=RGB24 fbmem=10M**

vmalloc=400M androidboot.console=ttymxc1

4) 使用 LCD 和 HDMI 双显示

在 u-boot 模式下执行以下命令来配置 4.3" LCD 和 HDMI 双显示模式

- MX6Q MARSBOARD U-Boot > **setenv bootargs console=ttymxc1,115200 init=/init
rw video=mxcfb0:dev=lcd,4.3inch_LCD,if=RGB24
video=mxcfb1:dev=hdmi,1920x1080M@60,if=RGB24 fbmem=10M vmalloc=400M
androidboot.console=ttymxc1 calibration**

在 u-boot 模式下执行以下命令来配置 7" LCD 和 HDMI 双显示模式

- MX6Q MARSBOARD U-Boot > **setenv bootargs console=ttymxc1,115200 init=/init
rw video=mxcfb0:dev=lcd,7inch_LCD,if=RGB24
video=mxcfb1:dev=hdmi,1920x1080M@60,if=RGB24 fbmem=10M vmalloc=400M
androidboot.console=ttymxc1 calibration**

注意:

 目前仅 Android 系统支持双显示

 U-boot 参数保存在 SPI-NOR Flash 中，如要重置，请执行以下命令:

- MX6Q MARSBOARD U-Boot > **run clearenv**

第5章 制作映像文件

本章节将介绍如何使用 ISO 中附带的 BSP 数据包来制作映像文件。针对 MarS Board 提供的 BSP 数据包包含了可用于生成 u-boot 启动文件、Linux 内核映像和 Android 文件系统的二进制代码、源代码和支持文件。

注意：

📖 以下命令在 Ubuntu 系统中执行。

📖 每一条命令前都加上了符号“•”，以防止由于指令较长占用多行而造成误解。

5.1 为 Linux 系统制作映像文件

请依次按照以下步骤来实现 Linux 系统映像文件的制作。

5.1.1 获取工具和源代码

1) 执行以下命令来获取交叉编译工具；

- `$ cd ~`
- `$ git clone git://github.com/embest-tech/platform_prebuilt.git`

2) 执行以下命令来获取 u-boot 源代码；

- `$ cd ~`
- `$ git clone git://github.com/embest-tech/u-boot-imx.git`

3) 执行以下命令来获取内核源代码；

- `$ cd ~`
- `$ git clone git://github.com/embest-tech/kernel_imx.git`

5.1.2 编译映像文件

1) 执行以下命令来编译 u-boot 映像;

- `$ cd ~ /uboot-imx`
- `$ export ARCH=arm`
- `$ export CROSS_COMPILE=~ /platform_prebuilt/linux-x86/toolchain/arm-eabi-4.4.3/bin/arm-eabi-`
- `$ make distclean`
- `$ make mx6q_marsboard_config`
- `$ make`

执行完成后, 在当前目录下会生成 u-boot.bin 文件;


2) 执行以下命令来编译内核映像;

- `$ export PATH=~ /uboot-imx/tools:$PATH`
- `$ cd ~/kernel_imx`
- `$ echo $ARCH && echo $CROSS_COMPILE`
- `$ export ARCH=arm`
- `$ export CROSS_COMPILE=~ /platform_prebuilt/linux-x86/toolchain/arm-eabi-4.4.3/bin/arm-eabi-`
- `$ make imx6_marsboard_defconfig`
- `$ make ulmage`

执行完成后, 在 arch/arm/boot/目录下会生成 ulmage 内核映像。

注意:

 用于生成内核和 ramfs 等映像文件的 mkimage 工具是在编译 u-boot.bin 文件后自动生成并保存于 tools/目录下, 因此在编译内核映像前需要首先完成 uboot 的编译。

 用 image 目录下的 oneiric.tgz 文件和新编译生成的 u-boot.bin、ulmage 文件覆盖 C:/tools/Mfgtools-Rel-12.04.01_ER_MX6Q_UPDATER \Profiles\MX6Q Linux Update\OS Firmware\files\目录下的同名文件, 然后从第 4 章中第 2) 步开始操作即可验证新的 Linux 系统。

5.2 为 Android 系统制作映像文件

请依次按照以下步骤来实现 Android 系统映像文件的制作。

5.2.1 获取 repo 源代码

1) 执行以下命令来获取 repo 源代码；

- `$ mkdir ~/bin`
- `$ curl https://github.com/android/tools_repo/blob/master/repo > ~/bin/repo`
- `$ chmod a+x ~/bin/repo`
- `$ export PATH=~/bin:$PATH`

2) 执行以下命令来初始化 repo 源代码；

- `$ mkdir ~/android-imx6-r13.3`
- `$ cd ~/android-imx6-r13.3`
- `$ repo init --repo-url=git://github.com/android/tools_repo.git -u git://github.com/embest-tech/android-imx6-r13.3.git`

3) 执行以下命令来同步 repo 源代码；

- `$ cd ~/android-imx6-r13.3`
- `$ repo sync`

5.2.2 编译映像文件

1) 执行以下命令来编译 Android 映像；

- `$ cd ~/android-imx6-r13.3`
- `$ source build/envsetup.sh`
- `$ lunch marsboard_6q-user`
- `$ make`

编译生成的映像保存在 android-imx6-r13.3/out/target/product/marsboard_6q/目录下；下表列出了编译完成后生成的映像以及目录；

表 5-1 映像和目录

映像和目录	注释
-------	----

映像和目录	注释
root/	根文件系统目录，挂载于根目录
system/	Android 系统目录，挂载于/system 目录
data/	Android 数据区，挂载于/data 目录
recovery/	Recovery 模式下的根文件系统目录，不直接使用
boot.img	复合映像，包含内核 zImage，ramdisk 和启动参数
ramdisk.img	Root/生成的 ramdisk 映像，不直接使用
system.img	System/生成的 EXT4 映像，可通过 dd 命令写入 SD/eMMC 存储卡的 SYSTEM 分区
userdata.img	Data/生成的 EXT4 映像
recovery.img	Recovery/生成的 EXT4 映像，可通过 dd 命令写入 SD/eMMC 存储卡的 RECOVERY 分区
u-boot.bin	带有补丁的 uboot 映像

注意：

Android 映像需要在用户模式下编译；请访问 <http://source.android.com/source/building.html> 获取更多信息。

2) 执行以下命令来编译 boot.img 映像；

- **\$ source build/envsetup.sh**
- **\$ lunch marsboard_6q-user**
- **\$ make bootimage**

执行完成后在 android-imx6-r13.3/out/target/product/marsboard_6q/目录下会生成 boot.img 映像文件。

注意：

- 用于生成内核和 ramfs 等映像文件的 mkimage 工具是在编译 u-boot.bin 文件后自动生成并保存于 tools/目录下，因此在编译内核映像前需要首先完成 uboot 的编译。
- 用编译生成的 boot.img、placeholder、recovery.img、system.img 和 u-boot.bin 文件覆盖 CD\tools\Mfgtools-Rel-12.04.01_ER_MX6Q_UPDATER\Profiles\MX6Q Linux Update\OS Firmware\files\android 目录下的同名文件，然后从第 4 章中第 2) 步开始操作即可验证新的 Android 系统。

附录 1 安装 Ubuntu Linux 系统

我们都知道在开发软件之前需要安装嵌入式开发环境。而产品 ISO 文件中提供的开发环境（linux/source 目录下）需要在 Linux 系统下才能运行。如果当前您使用的是 Windows 系统，那么首先需要安装 Linux 操作系统，然后才能在该系统下安装相应的开发环境。我们推荐使用 VirtualBox 虚拟机来在 Windows 中安装 Ubuntu Linux 操作系统。下面我们将依次介绍虚拟机 VirtualBox 和 Ubuntu Linux 系统的安装过程。

安装 VirtualBox 虚拟机

您可以访问 <http://www.virtualbox.org/wiki/Downloads> 来下载最新版本的 VirtualBox 虚拟机。在安装 VirtualBox 虚拟机之前，请确保您的 PC 拥有至少 512MB 的内存空间。建议提供 1G 以上的内存空间。

- 1) 安装过程很简单，此处略过。安装后从开始菜单启动 VirtualBox，然后单击程序窗口上方的 **New** 按钮，弹出新建虚拟机窗口；



图 1 新建虚拟机窗口

单击 **Next** 按钮开始新建虚拟机。

- 2) 在下方窗口中为新建的虚拟机指定名称和操作系统类型；

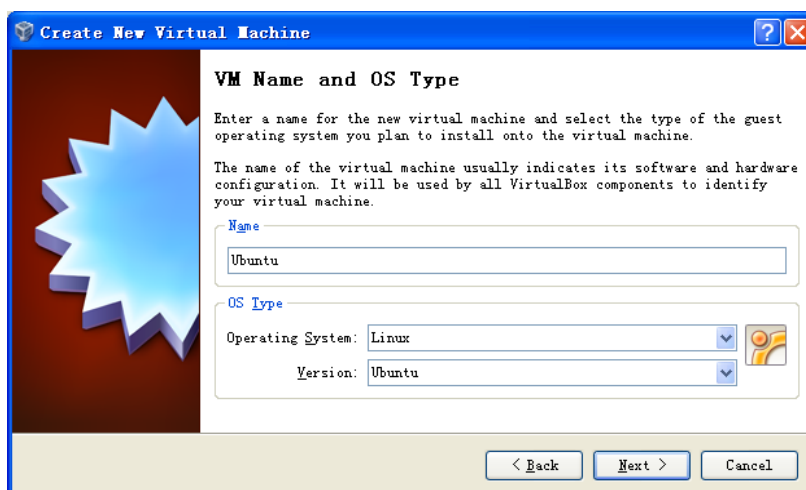


图 2 虚拟机名称和操作系统类型

您可以在 **Name** 一栏中输入新建虚拟机的名称，例如 Ubuntu。在 **Operating System** 一栏中选择 **Linux**，然后单击 **Next** 按钮。

- 3) 在以下窗口中为虚拟机分配适当大小的内存空间，分配完成后单击 **Next** 按钮；

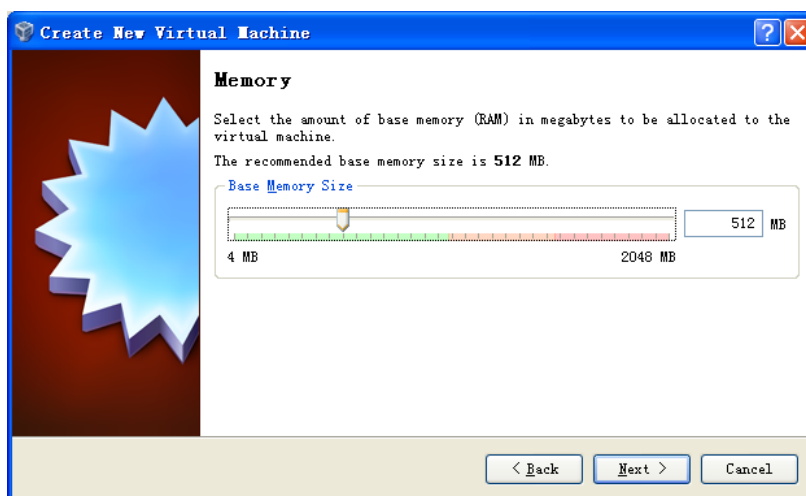




图 3 内存分配窗口

注意：

-  如果您的 PC 内存为 1G 或者更少，请保留默认设置；
-  如果您的 PC 内存超过 1G，则可以将 1/4 或者更少的内存分配给虚拟机，例如 PC 内存为 2G，则将 512MB 的内存分配给虚拟机。

- 4) 如果这是您第一次使用 VirtualBox，请在下方窗口中选择 **Create new hard**

disk 选项，然后单击 **Next** 按钮；

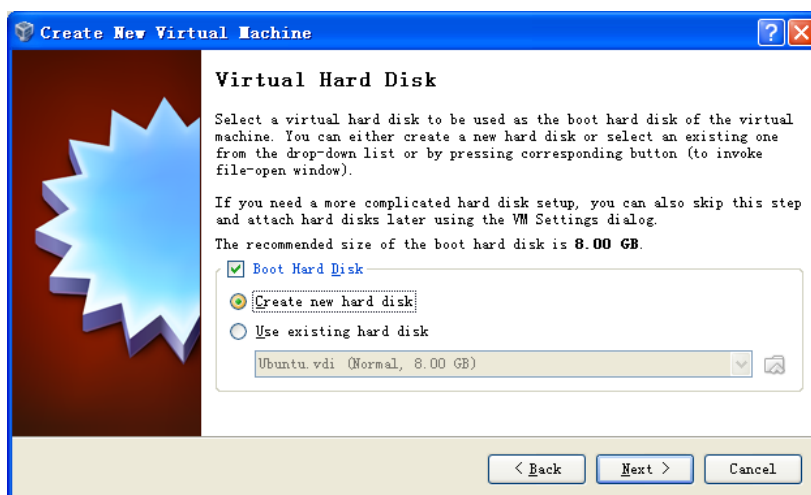


图 4 创建新的虚拟硬盘窗口

5) 在下方虚拟硬盘创建向导窗口中单击 **Next** 按钮；

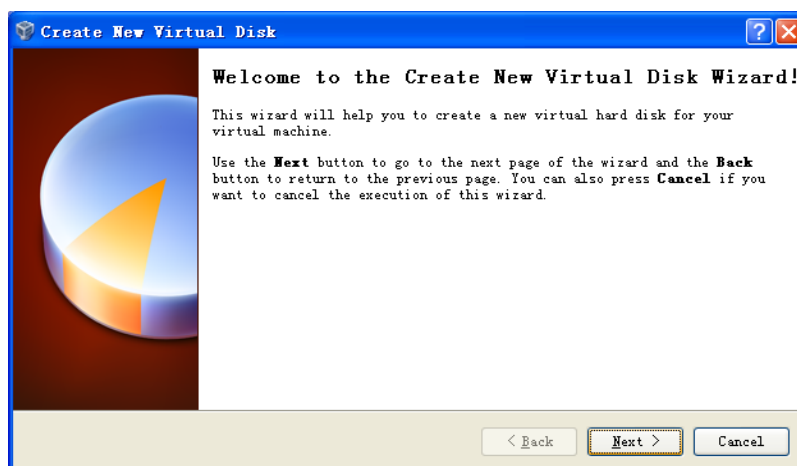


图 5 虚拟硬盘创建向导

- 6) 在下方窗口中选择 **Fixed-size storage** 选项，并单击 **Next** 按钮；

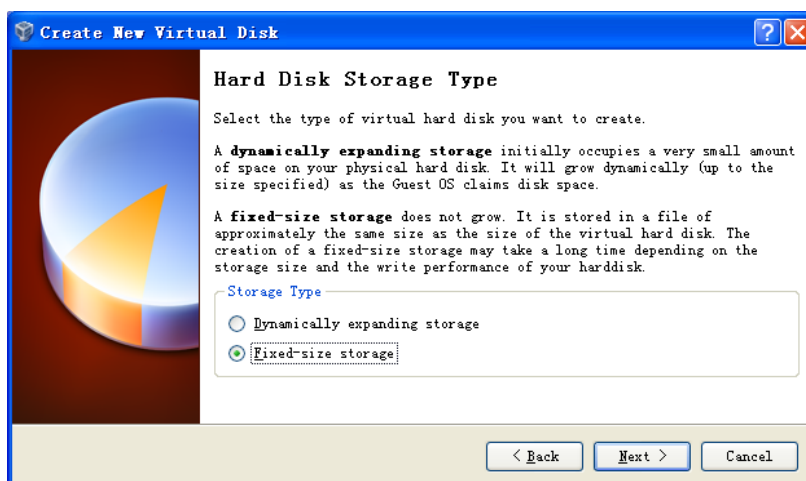


图 6 选择第二个选项

- 7) 在下方窗口中指定硬盘数据的存储位置以及默认虚拟硬盘的容量（至少 8G），然后单击 **Next** 按钮；

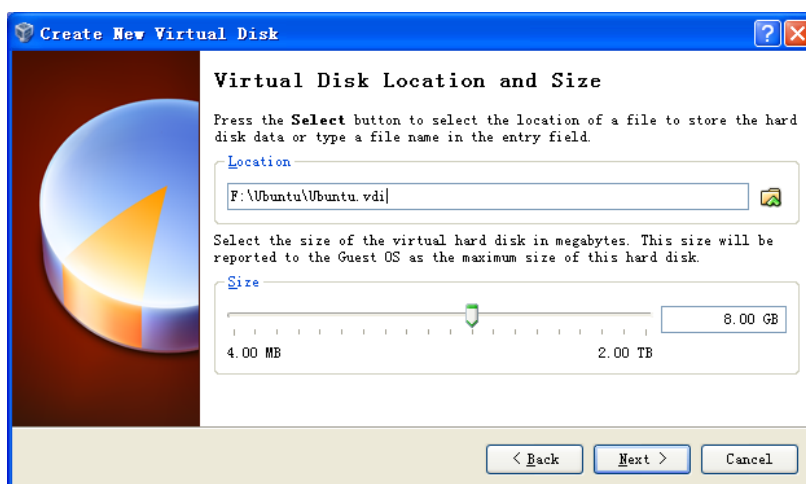


图 7 虚拟硬盘设置窗口

- 8) 在下方的虚拟硬盘信息窗口中单击 **Finish** 按钮;



图 8 虚拟硬盘信息

- 9) PC 开始创建虚拟硬盘驱动器;

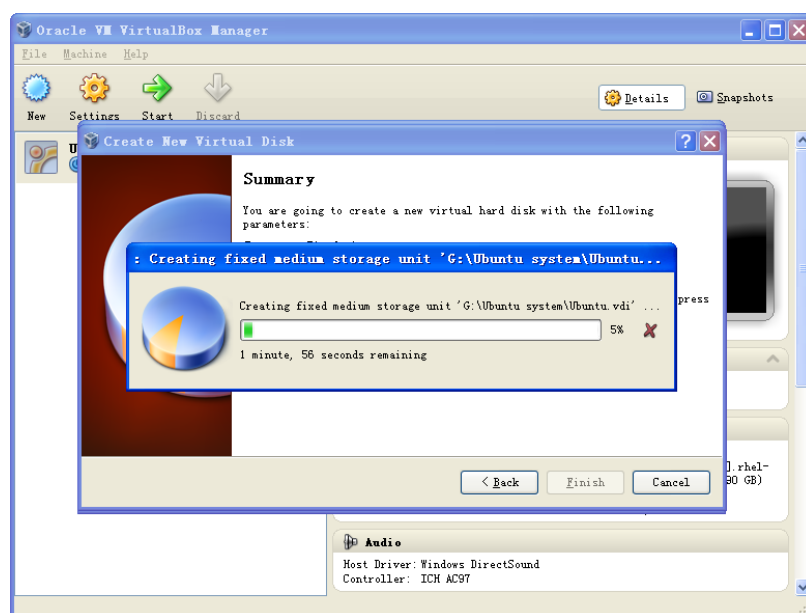


图 9 创建虚拟硬盘驱动器

- 10) 完成驱动器创建后会显示摘要信息。请单击**完成**按钮即完成虚拟机安装过程；

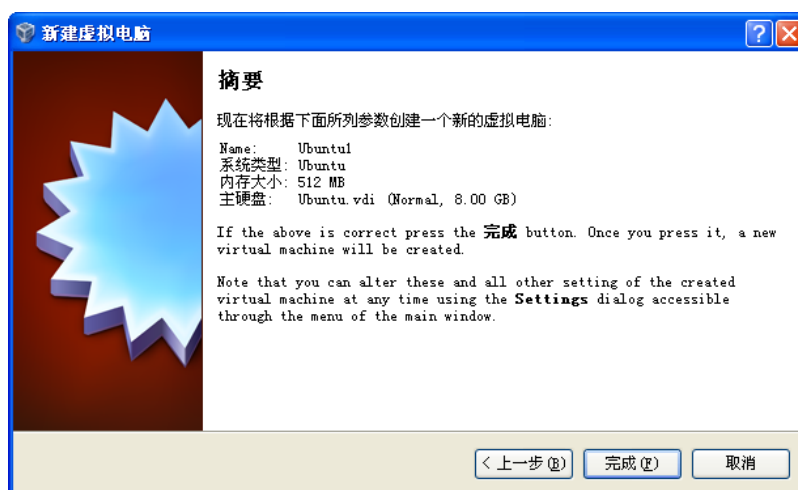


图 10 虚拟机配置完成

安装 Ubuntu Linux 系统

虚拟机安装完成后，我们就可以开始安装 Ubuntu Linux 系统了。首先请访问 <http://www.Ubuntu.com/download/Ubuntu/download>，下载 Ubuntu 的 ISO 映像文件，然后按照以下步骤进行安装。

- 1) 从开始菜单启动 VirtualBox，然后在程序窗口上方单击 **Setting** 按钮，弹出虚拟机设置窗口，如下图所示；

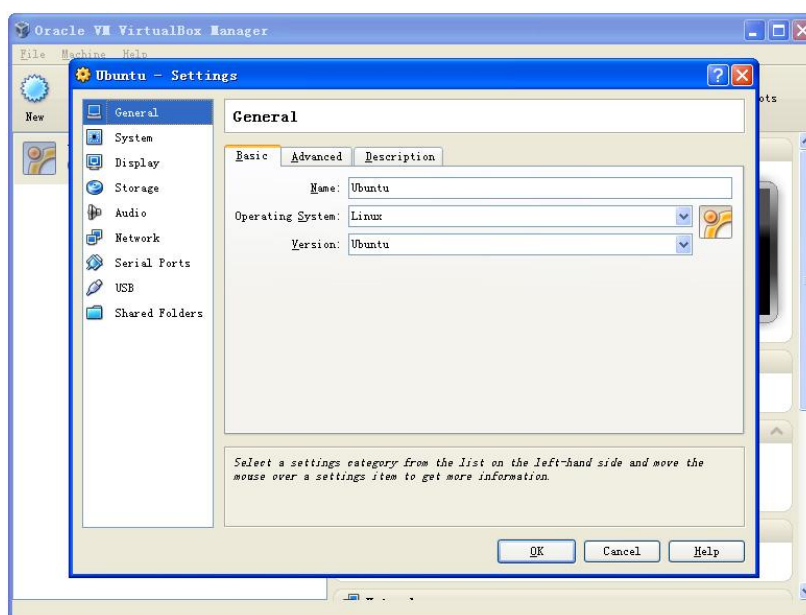


图 1 虚拟机设置

- 2) 在窗口左方选择 **Storage** 选项，然后单击 IDC 控制器下 **Empty** 文字右方的光盘图标来指定 Ubuntu 映像文件的位置，如下图所示；

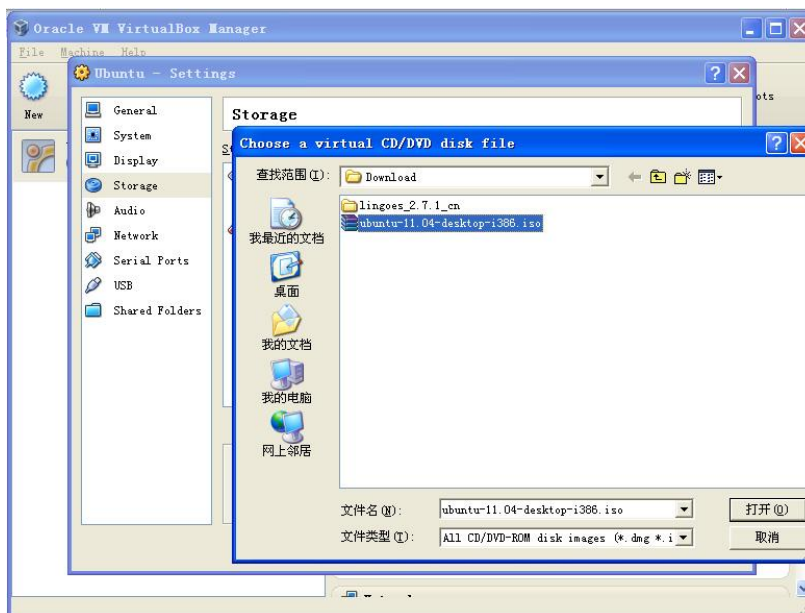


图 2 指定 Ubuntu 映像位置

- 3) 选中刚才添加的映像并单击 **OK** 按钮，如下图所示；

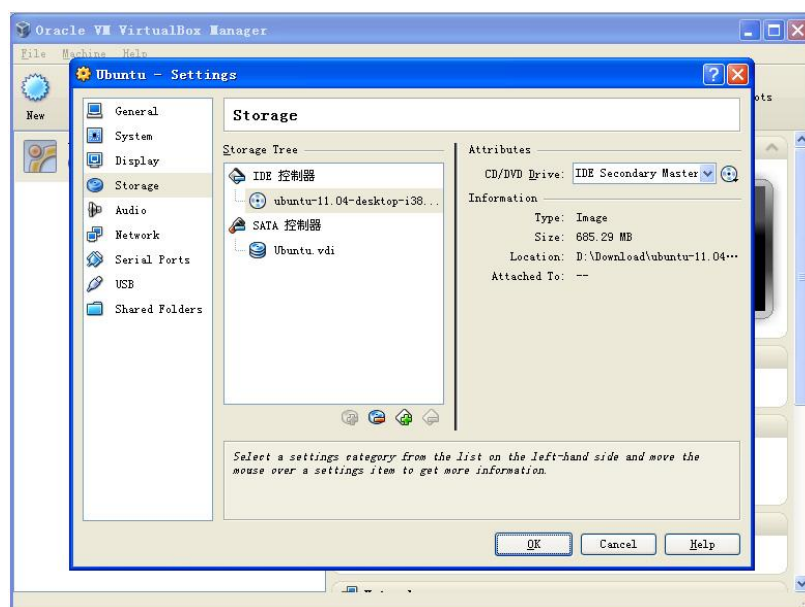


图 3 选中 ISO 映像

- 4) 单击 VirtualBox 窗口上方的 **Start** 按钮，屏幕会弹出 Ubuntu 的安装初始化窗口，如下图所示：

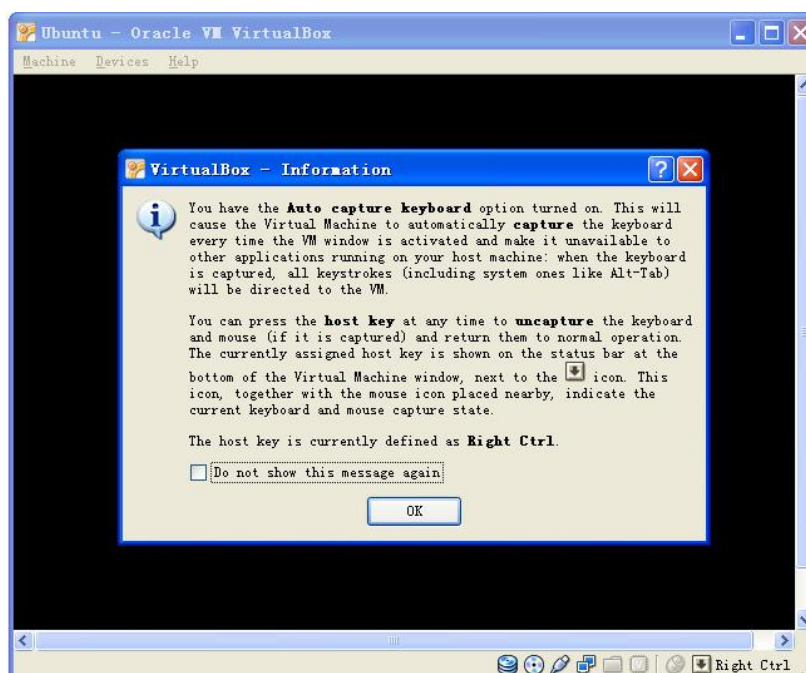


图 4 Ubuntu 初始化窗口

在 Ubuntu 安装窗口初始化过程中会出现一些提示信息，只需要单击提示信息下方的 **OK** 按钮即可继续初始化进程。

- 5) 在出现以下窗口时单击 **Install Ubuntu** 进行安装，如下图所示：

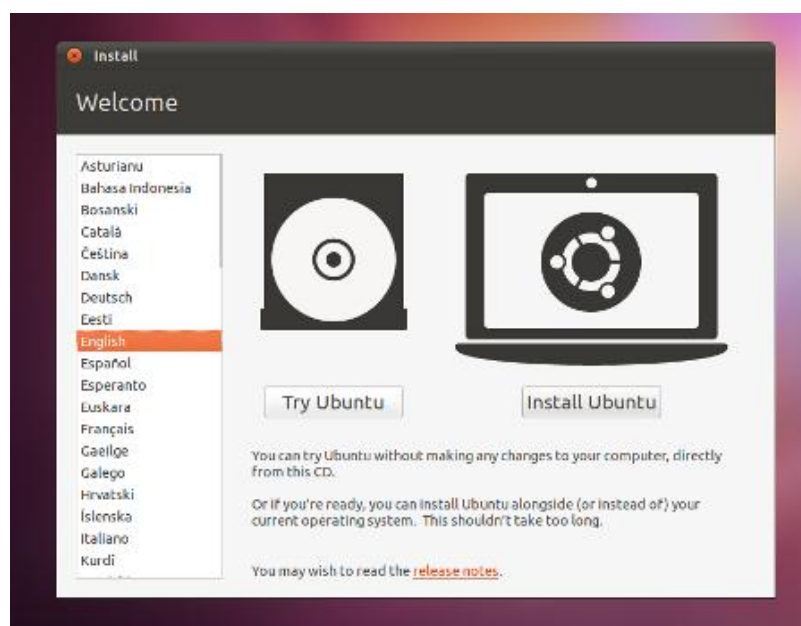


图 5 Ubuntu 安装窗口

- 6) 单击 **Forward** 按钮继续安装，如下图所示：



图 6 安装前的信息

- 7) 选择 **Erase disk and install Ubuntu** 选项，并单击 **Forward** 按钮。

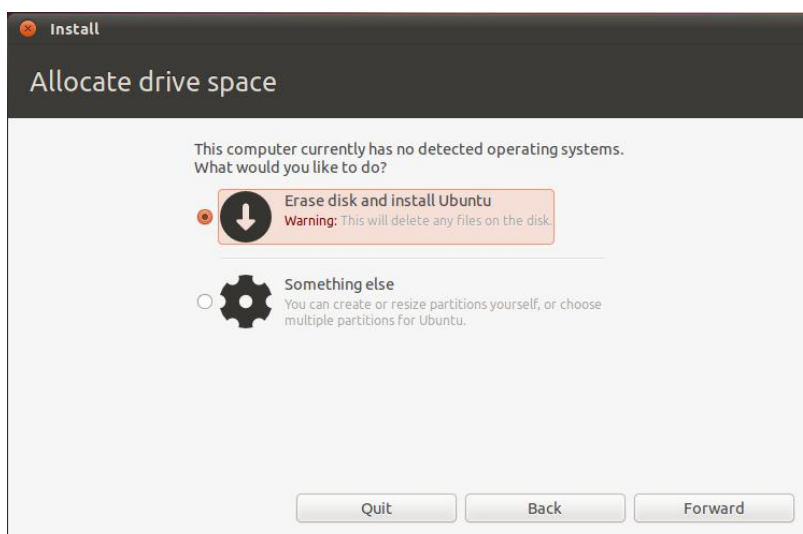


图 7 Ubuntu 安装选项

注意：

选择该选项并不会删除硬盘上物理分区中的任何内容。

- 8) 单击下方窗口中的 **Install Now** 按钮开始安装 Ubuntu;

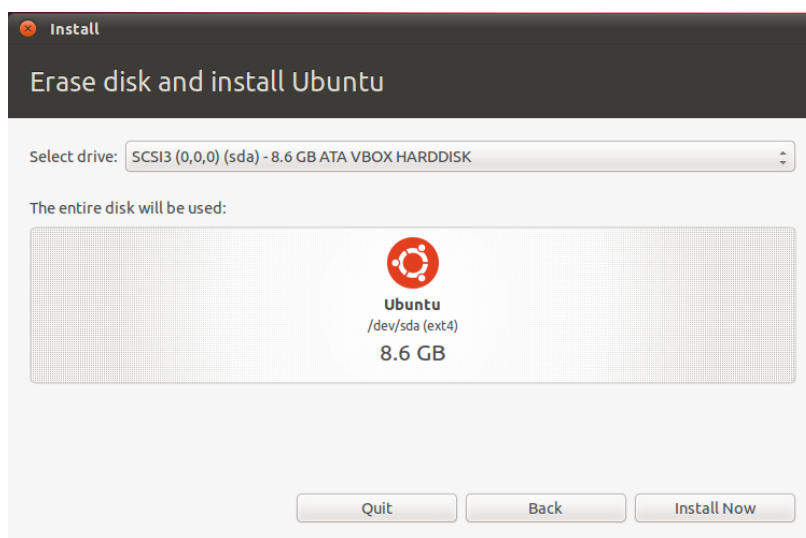


图 8 安装确认窗口

- 9) 在安装过程中安装程序会询问一些简单的问题，请输入相应的信息并单击 **Forward** 按钮即可。安装过程中的最后一个问题窗口如下图所示;

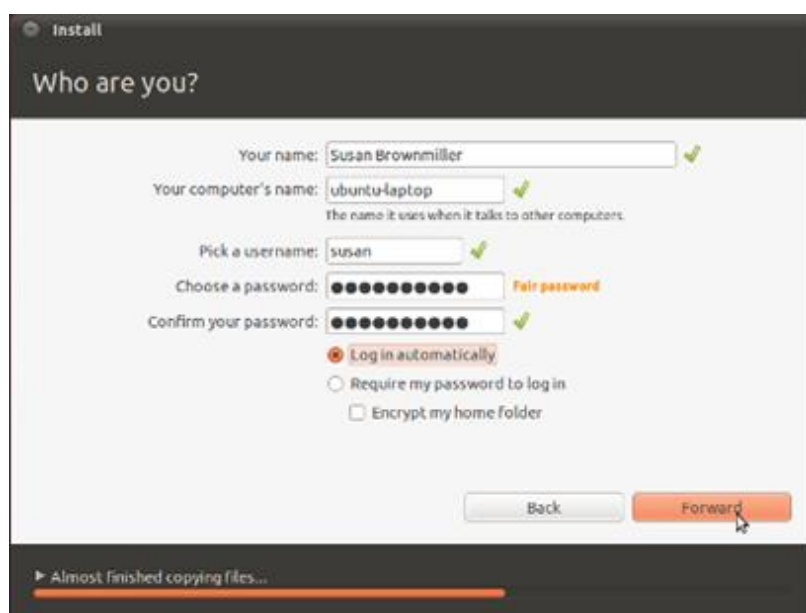


图 9 指定用户名和密码

在相应的文本框内输入用户名和密码后，选择 **Log in automatically** 选项并单击 **Forward** 按钮。

- 10) Ubuntu 的安装依据不同的 PC 性能可能需要 15 分钟至 1 小时左右。安装完成后会弹出如下图所示的提示窗口, 请选择 **Restart Now** 重新启动 Ubuntu 系统;



图 10 重新启动系统

- 11) 重新启动后就可以使用 Ubuntu 系统了。通常在重新启动 Ubuntu 系统后 VirtualBox 会自动弹出图 3 中载入的映像文件, 如果没有自动弹出, 您可以在 VirtualBox 的 **Setting** 窗口中手动弹出该映像, 使 IDE 控制器下显示为 Empty, 如下图所示;

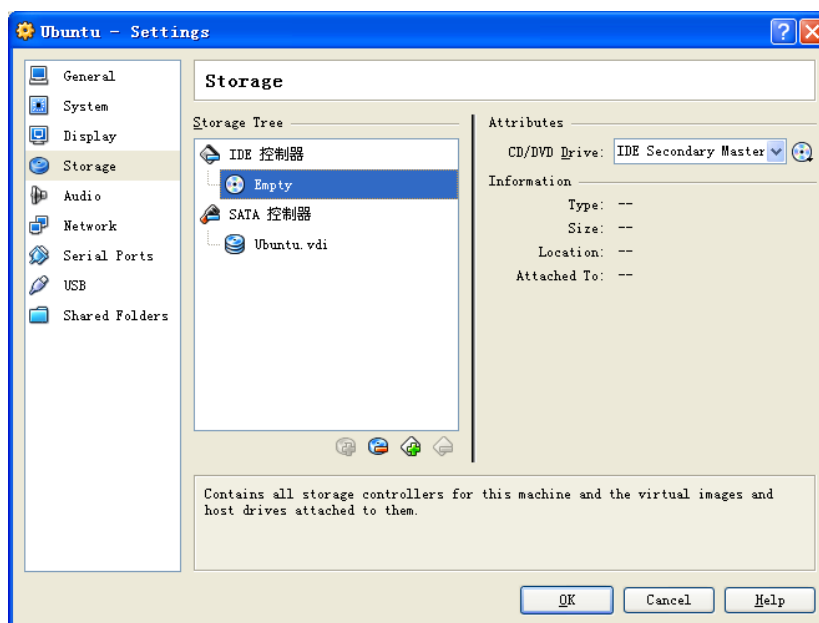


图 11 弹出 ISO 映像

技术支持和保修服务

技术支持



英蓓特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：

- 提供英蓓特科技嵌入式平台产品的软硬件资源；
- 帮助用户正确地编译和运行我们提供的源代码；
- 用户在按照本公司提供的产品文档操作的情况下，如本公司的嵌入式软硬件产品出现异常问题，我们将提供技术支持；
- 帮助用户判定是否存在产品故障。



以下情况不在我们的免费技术支持服务范围内，但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；
- 用户自行修改嵌入式操作系统遇到的问题；
- 用户自己的应用程序遇到的问题；
- 用户自行修改本公司提供的软件代码遇到的问题。

保修服务

- 1) 产品自出售之日起，在正常使用状况下为印刷电路板提供 12 个月的免费保修服务；
- 2) 以下情况不属于免费服务范围，英蓓特科技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. 未按用户手册操作导致产品损坏的；
 - 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英蓓特科技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英蓓特科技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. 非英蓓特科技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, 英蓓特科技不承担任何责任;
- 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。

注意:

 未经本公司许可私自将产品寄回的, 英蓓特科技公司不承担任何责任。

联系方式

技术支持

电话: +86-755-25635626-872/875/897

Email: support@timll.com

销售信息

电话: +86-755-25635626-860/861/862

传真: +86-755-25616057

Email: support@timll.com

公司信息

网站: <http://www.timll.com>

地址: 深圳市南山区留仙大道 1183 号南山云谷创新产业园山水楼 4 楼 B