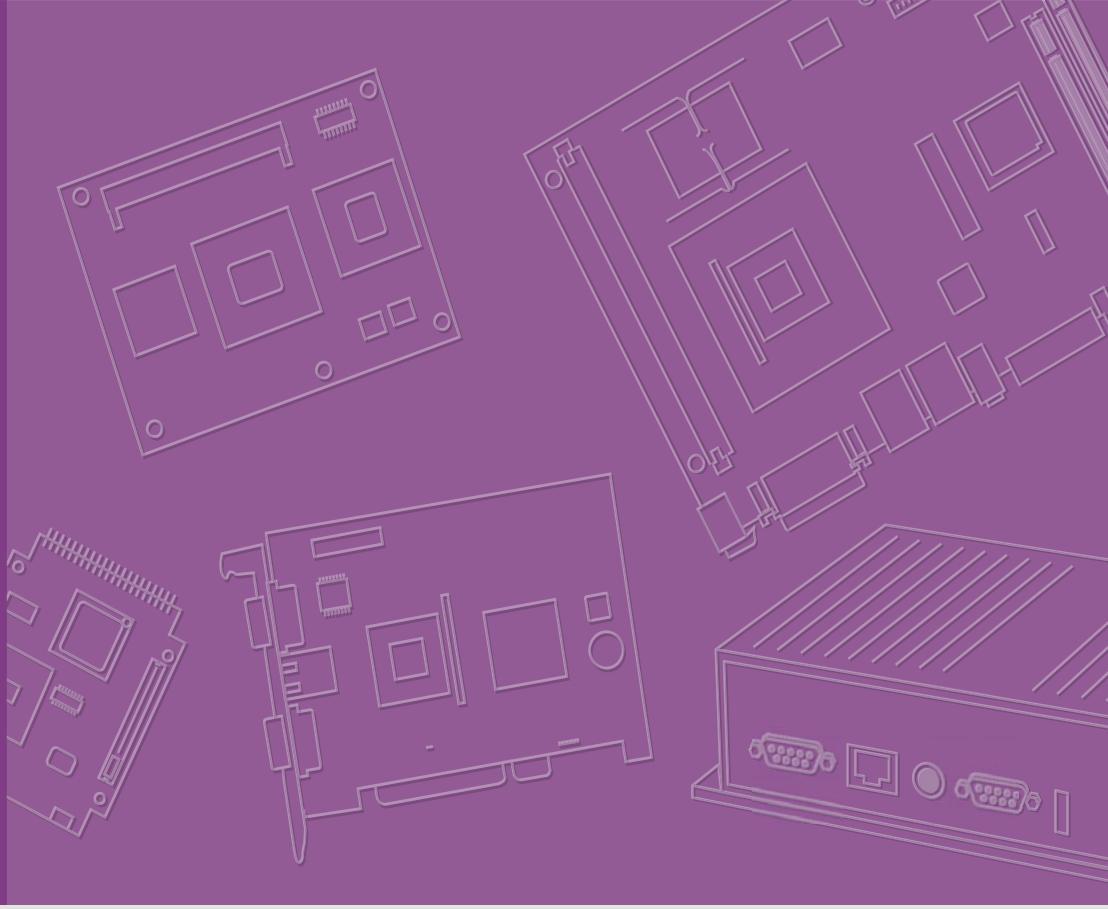


用户手册

UBC-220

搭载 Freescale i.MX6 Dual Lite 处理器 ARM® Cortex™ A9 架构的 RISC IoT 工控机

ADVANTECH

Enabling an Intelligent Planet

版权声明

认可声明

Intel 和 Pentium 为 Intel Corporation 的商标。

Microsoft Windows® 为 Microsoft Corp. 的注册商标。

所有其它产品名或商标均为各自所属方的财产。

UBC-220 用户手册中文第一版，参照 UBC-220 用户手册英文第一版。

产品质量保证 （两年）

从购买之日起，研华为原购买商提供两年的产品质量保证。但对那些未经授权的维修人员维修过的产品不予提供质量保证。研华对于不正确的使用、灾难、错误安装产生的问题有免责权利。

如果研华产品出现故障，在质保期内我们提供免费维修或更换服务。对于出保产品，我们将会酌情收取材料费、人工服务费用。请联系相关销售人员了解详细情况。

如果您认为您购买的产品出现了故障，请遵循以下步骤：

1. 收集您所遇到的问题信息 （例如，CPU 主频、使用的研华产品及其它软件、硬件等）。请注意屏幕上出现的任何不正常信息显示。
2. 打电话给您的供货商，描述故障问题。请借助手册，产品和任何有帮助的信息。
3. 如果您的产品被诊断发生故障，请从您的供货商那里获得 RMA （Return Material Authorization） 序列号。这可以让我们尽快地进行故障产品的回收。
4. 请仔细地包装故障产品，并在包装中附上完整的售后服务卡片和购买日期证明（如销售发票）。我们对无法提供购买日期证明的产品不提供质量保证服务。
5. 把相关的 RMA 序列号写在外包装上，并将其运送给销售人员。

料号：2006022020　　第一版

中国印刷　　2016 年 3 月

符合性声明

FCC B 级

根据 FCC 规则第 15 条，本设备已经过检测并被判定符合 B 级数字设备标准。这些限制旨在为居住环境下的系统操作提供合理保护，使其免受有害干扰。本设备会产生、使用和发射无线电频率能量。如果没有按照手册说明正确安装和使用，可能对无线电通讯造成有害干扰。但即使按照手册说明进行安装和使用，也并不能保证不会产生干扰。若本设备会对无线电或电视信号接收产生有害干扰，用户可通过开、关设备进行确认。当本设备产生有害干扰时，用户可采取下面的措施来解决干扰问题：

- 调整接收天线的方向或位置
- 增大本设备与接收器之间的距离
- 将本设备的电源接头插在与接收器使用不同电路的电源插座
- 若需技术支持，请咨询经销商或经验丰富的无线电 / 电视技术人员

技术支持与服务

1. 有关该产品的最新信息，请访问研华公司的网站：
   http://support.advantech.com.cn
2. 用户若需技术支持，请与当地分销商、销售代表或研华客服中心联系。进行技术咨询前，用户须将下面各项产品信息收集完整：
   - 产品名称及序列号
   - 外围附加设备的描述
   - 用户软件的描述 （操作系统、版本、应用软件等）
   - 产品所出现问题的完整描述
   - 每条错误信息的完整内容

包装清单

安装系统之前，用户需确认包装中含有本设备以及下面所列各项，并确认设备完好。若有任何不符，请立即与经销商联系。

- 1 x UBC-220 RISC 工控机
- 1 x 6 针端子板接口

订购信息

产品型号	说明
UBC-220DL-MDA1E	搭载 Freescale i.MX6 Cortex-A9 Dual Lite 处理器的紧凑型工控机

可选附件

产品料号	说明
1960069533S001	散热片
96PSA-A36W12R1	适配器 100-240V 36W 12V 3A，不带 PFC 9NA0361603
170203183C	电源线 3P 欧洲标准 （WS-010+WS-083）183cm
170203180A	电源线 3P UK 标准 2.5A/3A 250V 1.83M
1700001524	电源线 3P UL 标准 10A 125V 180cm
96LEDK-V121SV45NE1	平板显示器 TM121SDS01
1700021565-01	调试电缆
1700022130-01	LVDS 电缆
1700022131-01	背光灯电缆

安规认证和安全指示

本品符合 FCC 规则第 15 款限制。操作符合下列两种情况：

1. 此装置不可产生干扰，且
2. 此装置必须接受任何干扰，包括可能导致非预期操作的干扰。

根据 FCC 规则第 15 款，本设备已经过检测并被判定符合 B 级数字设备标准。这些限制旨在为商业环境下的系统操作提供合理保护，使其免受有害干扰。本设备会产生、耗费和发射无线电频率能量，如果没有按照手册说明正确安装和使用，可能会对无线电通讯造成有害干扰。此时，用户需自行解决干扰问题。
若未经相关权威机构明确批准而擅自更改或修理设备，则用户操作本设备的权利可能会被取消。

注意！ *如果电池放置不正确，将有爆炸的危险。*

请不要对电池进行充电、强行打开或加热等操作。因此，只可以使用制造商推荐的同一种或者同等型号的电池进行替换。

请按照制造商的指示处理旧电池。

目录

第 1 章

概述

本章介绍 UBC-220 的基本信息。

内容包括：

- 产品简介
- 产品特性
- 产品规格

1.1 产品简介

UBC-220 是一款搭载 Freescale i.MX6 Dual Lite 高性能 CPU 的紧凑型工控机，适合各种 IoT、自动化及 HMI 应用。该份用户手册将包括两部分内容：硬件 I/O 规格和软件介绍。由于该份文档包含重要信息，如 I/O 接口针脚定义、处理流程及接口规格等，因此建议用户在开始评估 UBC-220 之前完整阅读用户手册的内容。

请注意并遵守该份文档中的所有警告信息及规则，以防损坏设备或造成人身伤害。不恰当的处理方式可能会导致设备出现物理损坏、异常行为或性能不稳定。

有关研华RISC产品的更多信息，请访问RISC Mini网站：http://risc.advantech.com。

1.2 产品规格

1.2.1 功能规格

处理器：Freescale i.mx6 系列

- 搭载 ARM Cortex™-A9 Dual Lite 1 GHz 高性能处理器
- 支持 1 个 IPU、用于 3D 的 OpenGL ES 2.0、用于 2D 的 BitBLT 以及 OpenVG™ 1.1
- **视频解码器：**MPEG-4 ASP、H.264 HP、H.263、MPEG-2 MP、MJPEG BP
- **视频编码器：**MPEG-4 SP、H.264 BP、H.263、MJPEG BP

系统内存支持

- DDR3 800 MHz
- **容量：**板载 DDR3 1 GB

千兆位以太网

- **芯片组：**Freescale i.MX6 集成 RGMII
- 1 x10/100/1000 Mbps

外围接口

- 1 x 单通道 18/24 bit LVDS 接口
- 1 x HDMI
- 1 x USB2.0 A 型接口和 1 个 USB 2.0 OTG 接口
- 1 x SD 卡插槽
- 1 x 4 线 UART 端子板接口
- 2 x miniPCIe 卡插槽
- 1 x SIM 卡插槽

OS 支持

UBC-220 支持 Linux BSP 3.0.35

1.2.2 机械规格

- **尺寸：**100 x 72 mm（5.7” x 4”）
- **高度：**20.6 mm
- **参考重量：**540 g（包括整个包装）

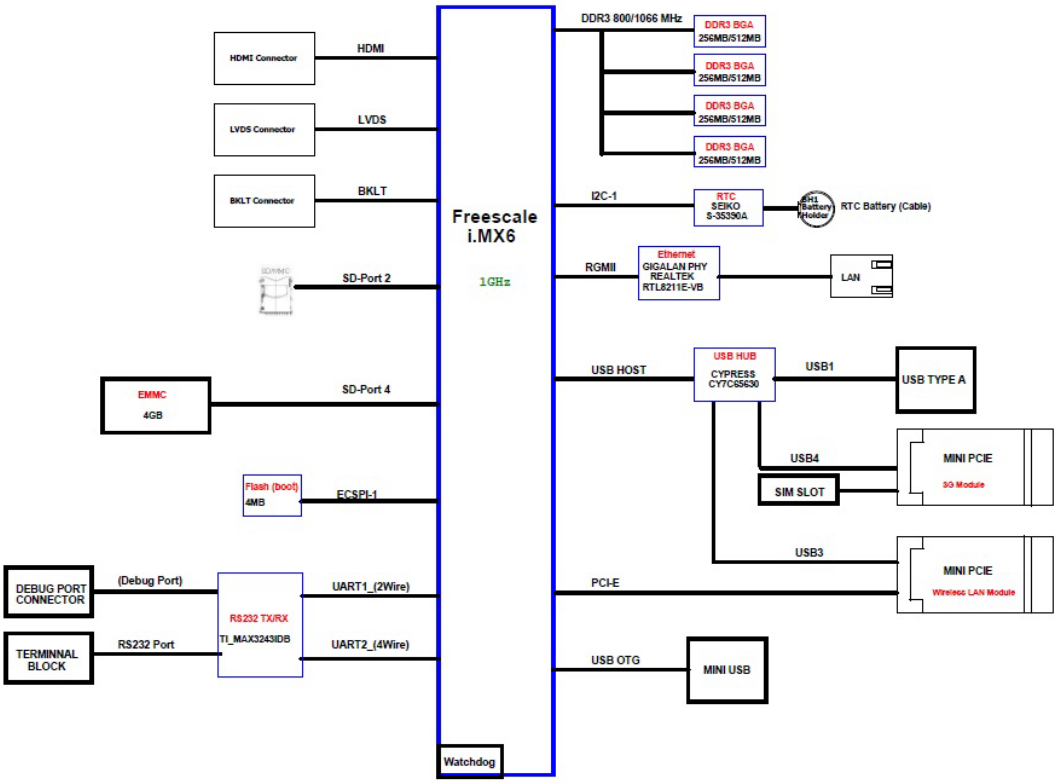
1.2.3 电气规格

- **电源类型：**DC 输入 12 V
- 功耗：
  - 内核空闲模式：2.3 W
  - 工作模式（最大）：4.08 W
- **RTC 电池：**
  - 典型电压：3.0 V
  - 正常放电容量：3 uA

1.3 环境规格

- **工作温度：**0 ~ 60° C（32 ~ 140° F）
- **工作湿度：**40° C @ 95% RH，非凝结
- **储存温度：**-40 ~ 85° C（-40 ~ 185° C）
- **储存湿度：** 60° C @ 95% RH，非凝结

1.4 功能块图



第 2 章

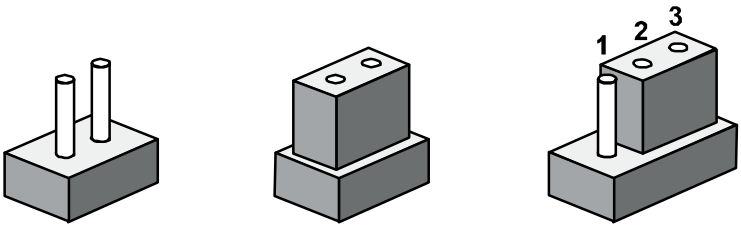
硬件安装

本章介绍 UBC-220 的硬件安装过程，包括设置跳线和集成设备。此外，还将介绍如何设置开关及指示灯，并提供产品机械图。请在开始安装操作前仔细阅读所有的安全指示。

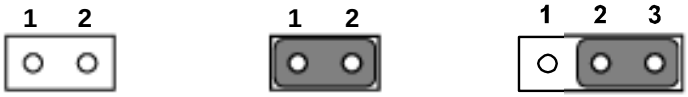
2.1 跳线

2.1.1 跳线描述

用户可通过设置跳线配置板卡。跳线是用来连通电路的金属桥。它包括 2 个金属针脚和 1 个跳线帽 （里面是金属夹片，外部是起保护作用的塑料套）。跳线帽可套住针脚将其连成通路。移走跳线帽则会断开线路。有时，一个跳线 具有 3 个针脚，分别为针 1、2、3。 这种情况下，用户可以任意选择连接针脚 1、2 或者针脚 2、3。



跳线设置如下图所示：



设置跳线时，使用针鼻钳子将会很有帮助。如果您对硬件配置存有任何疑问，请在更改设置前联系经销商或销售代表。

大多数情况下，用户只需要准备一根标准电缆即可。

警告！ *请务必在设置跳线前断开系统电源，以防损坏计算机。*

2.1.2 跳线列表

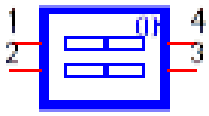
表 2.1：跳线列表

SW2	启动设备
LVDS_VDD_SLT	LVDS 电源
LVDS_BKLT_SLT	背光灯电源

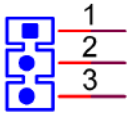
2.1.3 跳线设置

SW2	启动设备
产品料号	1600000202
Footprint	SW_2x2P_50_161X315
说明	DIP SW CHS-02TB(29) SMD 4P SPST P=1.27mm W=5.4mm
设置	功能
1 ON	从 SD 卡启动
1 OFF	从 SPI 启动

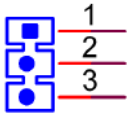
此开关用于选择启动方式。



LVDS_VDD_SLT	LVDS 电源
产品料号	1653003100
Footprint	HD_3x1P_100_D
说明	PIN HEADER 3x1P 2.54mm 180D(M) DIP 205-1x3GS
设置	功能
(1-2)	+V3.3
(2-3)	+V5



LVDS_BKLT_SLT	LVDS 背光灯电源
产品料号	1653003100
Footprint	HD_3x1P_100_D
说明	PIN HEADER 3x1P 2.54mm 180D(M) DIP 205-1x3GS
设置	功能
(1-2)	+V5
(2-3)	+VIN (+12V)



2.2 接口

2.2.1 接口列表

表 2.2：接口列表

CN1	RTC 电池
MINI_PCIE_HALF	半长 MiniPCIe
MINI_PCIE_FULL	全长 Mini PCIe
SIM_SLOT	SIM 卡插槽
DEBUG_CONSOLE	UART1 调试端口
USB_HOST	USB A 型接口
USB_OTG	USB OTG 接口
LAN	以太网端口
DCIN	DC 电源插孔
HDMI	HDMI 接口
SD	SD 卡插槽
COM	COM 端口
LVDS	LVDSL 接口
LVDS_BKLT_PWR	背光灯接口

2.2.2 接口设置

2.2.2.1 RTC 电池接口 （CN1）

UBC-220 可通过电池接口连接一块带线缆的 3V/210mAH CR2032 锂电池。

2.2.2.2 半长 MiniPCIe 接口 （MINI_PCIE_HALF）

UBC-220 可通过 USB 和 PCIe 接口支持一块半长 miniPCIe 卡。

表 2.3：半长 MiniPCIe 接口

针脚	信号名	针脚	信号名
1		2	3.3Vaux
3	预留	4	GND
5	预留	6	
7		8	
9	GND	10	
11	REFCLK-	12	
13	REFCLK+	14	
15	GND	16	
		机械钥匙	
17		18	GND
19		20	W_DISABLE#
21	GND	22	PERST#
23	PERn0	24	3.3Vaux
25	PERp0	26	GND
27	GND	28	
29	GND	30	
31	PETn0	32	

表 2.3：半长 MiniPCIe 接口			
33	PETp0	34	GND
35	GND	36	USB_D-
37	GND	38	USB_D+
39	3.3VAUX	40	GND
41	3.3VAUX	42	LED_WWAN#
43	GND	44	LED_WLAN#
45	预留	46	
47	预留	48	
49	预留	50	GND
51	预留	52	3.3VAUX

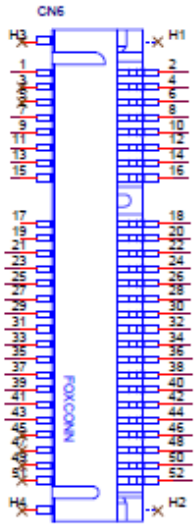


图 2.1： 半长 miniPCIE 接口

2.2.2.3 全长 MiniPCIe 接口 （MINI_PCIE_FULL）

UBC-220 支持一个全长 miniPCIe 卡插槽。该插槽仅支持 USB 接口。

表 2.4：全长 miniPCIE 接口			
针脚	**信号名**	**针脚**	**信号名**
1	NA	27	GND
2	+3.3V	28	NA
3	预留	29	GND
4	GND	30	NA
5	预留	31	NA
6	NA	32	NA
7	NA	33	NA
8	UIM_PWR	34	GND
9	GND	35	GND
10	UIM_DATA	36	USB_DATA-
11	NA	37	预留
12	UIM_CLK	38	USB_DATA+
13	NA	39	预留
14	UIM_RESET	40	GND

表 2.4：全长 miniPCIE 接口			
15	GND	41	预留
16	UIM_VPP	42	LED_WWAN
17	预留	43	预留
18	GND	44	LED_WLAN
19	预留	45	预留
20	预留	46	LED_WPAN
21	GND	47	预留
22	PERST	48	NA
23	NA	49	预留
24	+3.3V	50	GND
25	NA	51	预留
26	GND	52	+3.3V

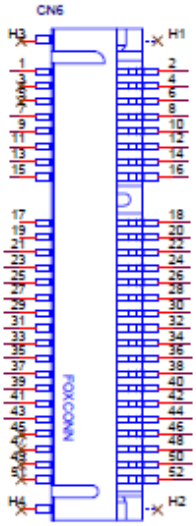


图 2.2： 全长 miniPCIE 接口

2.2.2.4 SIM 卡插槽 （SIM_SLOT）

UBC-220 支持一个板载 SIM 卡插槽，可实现 3G 网络连接。请插入有效的 SIM 卡连接 3G 网络。

表 2.5：SIM 卡插槽			
针脚	**信号名**	**针脚**	**信号名**
C1	UIM_PWR	C2	UIM_RESET
C3	UIM_CLK	C5	GND
C6		C7	UIM_DATA

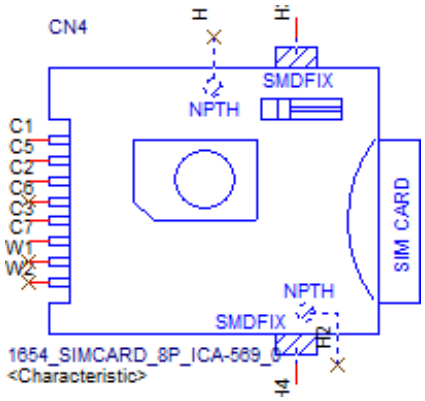


图 2.3： SIM 卡插槽

2.2.2.5 UART1 调试端口 （DEBUG_CONSOLE）

UBC-220 可通过串行电缆实现与主服务器 （Windows 或 Linux）的数据通讯。

表 2.6：UART1 调试端口

针脚	信号名
1	+V3.3
2	DEBUG_TXD
3	DEBUG_RXD
4	GND

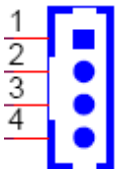


图 2.4： UART1 调试端口

2.2.2.6 USB A 型接口 （USB_HOST）

UBC-220 的 I/O 海岸线上还支持一个标准 USB2.0 A 型接口。

表 2.7：USB A 型接口

针脚	信号名
1	+5V
2	USB Data-
3	USB Data+
4	GND

图 2.5： USB A 型接口

2.2.2.7 以太网端口 （LAN）

UBC-220 提供一个 RJ45 LAN 接口，完全符合 IEEE 802.3u 10/100/1000 Base-T CSMA/CD 标准。以太网端口采用标准 RJ-45 插孔，前侧带 LED 指示灯显示系统传输 / 连接状态和速度状态。

表 2.8：以太网端口

针脚	信号名
1	MDI0+
2	MDI0-
3	MDI1+
4	MDI1-
5	GND
6	GND
7	MDI2+
8	MDI2-
9	MDI3+
10	MDI3-
11	VCC
12	ACT
13	Link100#
14	Link1000#

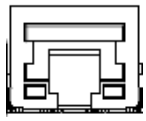


图 2.6： 以太网端口

2.2.2.8 DC 电源插孔 （DCIN）

UBC-220 提供一个 DC 插孔，可支持 12 VDC 的外部电源输入。

表 2.9：DC 电源插孔

针脚	信号名
1	DC_IN
2	GND

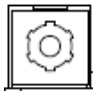


图 2.7： DC 电源插孔

2.2.2.9 HDMI （HDMI）

UBC-220 提供一个 HDMI 接口，可支持所有数字量音频 / 视频接口传输未压缩的音频 / 视频信号。该接口符合 HDCP 和 CEC 标准要求。用户只需将 HDMI 音频 / 视频设备连接至此接口即可。HDMI 技术能够支持的最大分辨率为 1920 x 1080p，而实际分辨率则取决于用户所使用的显示器。

表 2.10：HDMI 接口

针脚	信号名
1	HDMI_TD2+
2	GND
3	HDMI_TD2-
4	HDMI_TD1+
5	GND
6	HDMI_TD1-
7	HDMI_TD0+
8	GND
9	HDMI_TD0-
10	HDMI_CLK+
11	GND
12	HDMI_CLK-
13	HDMI_CEC_A
14	GND
15	DDC_CLK_HDMI_A
16	DDC_DATA_HDMI_A
17	GND
18	+5V_HPD
19	HDMI_HP

图 2.8： HDMI 接口

2.2.2.10USB OTG 接口 （USB_OTG）

UBC-220 支持 USB-OTG 模式，可替代 UART 调试控制台。

表 2.11：USB OTG 接口

针脚	信号名
1	+5V
2	USB Data-
3	USB Data+
4	ID
5	GND

图 2.9： USB OTG 接口

2.2.2.11SD 卡插槽

UBC-220 支持 SD/MMC 卡 （2/4/6/8/10 级），最大容量可达 32G （SDHC）。

表 2.12： SD 卡插槽

针脚	信号名
1	DAT3
2	CMD
3	GND
4	+3.3V
5	CLK
6	GND
7	DAT0
8	DAT1
9	DAT2

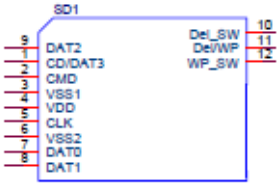


图 2.10： SD 卡插槽

2.2.2.12COM 端口

UBC-220 提供一个 6 针端子板接口，可作为串行通讯端口使用。该端口支持 RS-232 通讯模式。

表 2.13： COM 端口

针脚	信号名
1	RX
2	RTS
3	TX
4	CTS
5	GND
6	N/C

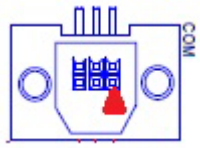


图 2.11： COM 端口

2.2.2.13LVDS 接口

UBC-220 提供一个 LVDS 10x2 针板对板接口，可连接一台分辨率高达 1366x768 的单通道 18/24 bit LVDS 平板显示器。连接 LVDS 平板显示器之前，请阅读跳线设置内容。

表 2.14：LVDS 接口

针脚	信号名
1	GND
2	GND
3	LVDS0_TX0_P
4	I2C1_SCL_LVDS0
5	LVDS0_TX0_N
6	I2C1_SDA_LVDS0
7	LVDS0_TX1_P
8	
9	LVDS0_TX1_N
10	
11	LVDS0_TX2_P
12	
13	LVDS0_TX2_N
14	
15	LVDS0_CLK_P
16	LVDS0_TX3_P
17	LVDS0_CLK_N
18	LVDS0_TX3_N
19	+VDD_LVDS
20	+VDD_LVDS

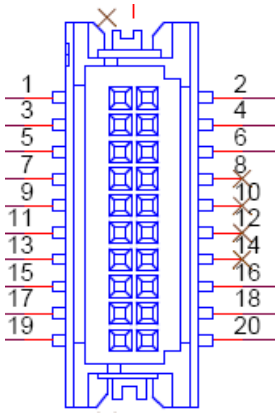


图 2.12： LVDS 接口

2.2.2.14LVDS 背光灯接口

连接 LVDS 平板显示器之前，请阅读跳线设置内容。

表 2.15：LVDS 背光灯接口

针脚	信号名
1	+VDD_BKLT_LVDS
2	GND
3	LCD_BKLT_A
4	LCD_BKLT_PWM_A
5	+V5

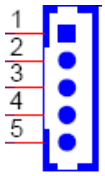


图 2.13： LVDS 背光灯接口

2.3 机械规格

2.3.1 跳线和接口位置

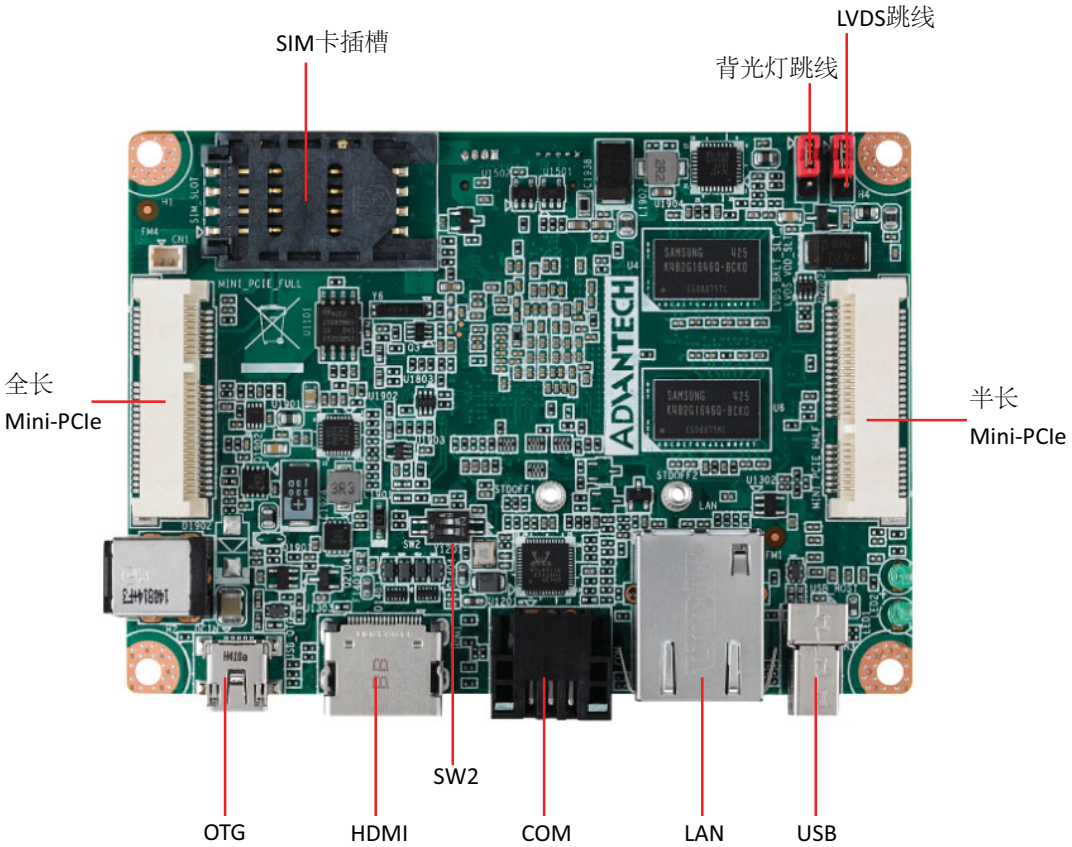


图 2.14： 跳线和接口布局图 （顶部）

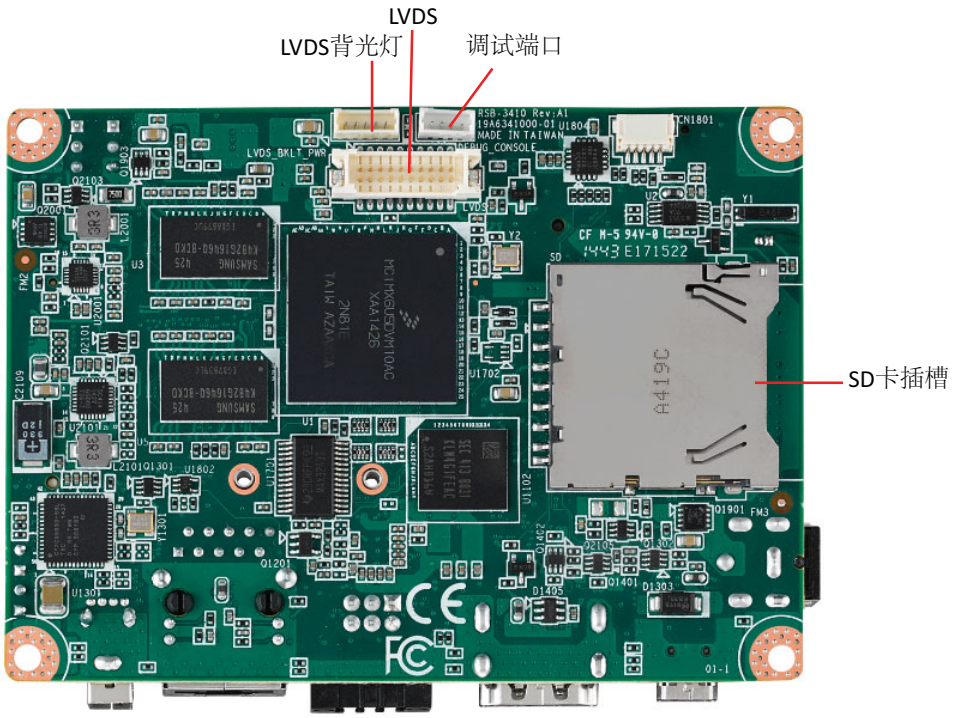
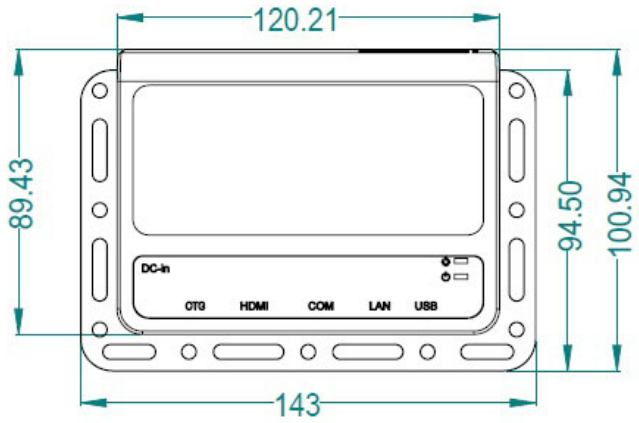


图 2.15： 跳线和接口布局图 （底部）

图 2.16： 跳线和接口布局图 （海岸线）

2.3.2 系统尺寸

2.3.2.1 系统尺寸图



单位: mm

图 2.17： 系统尺寸图 （顶部）

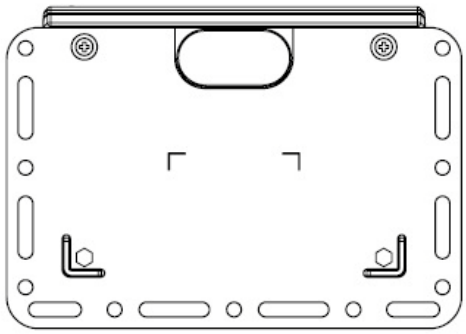


图 2.18： 系统尺寸图 （底部）

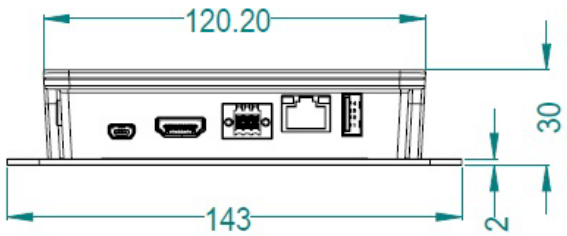


图 2.19： 系统尺寸图 （海岸线）

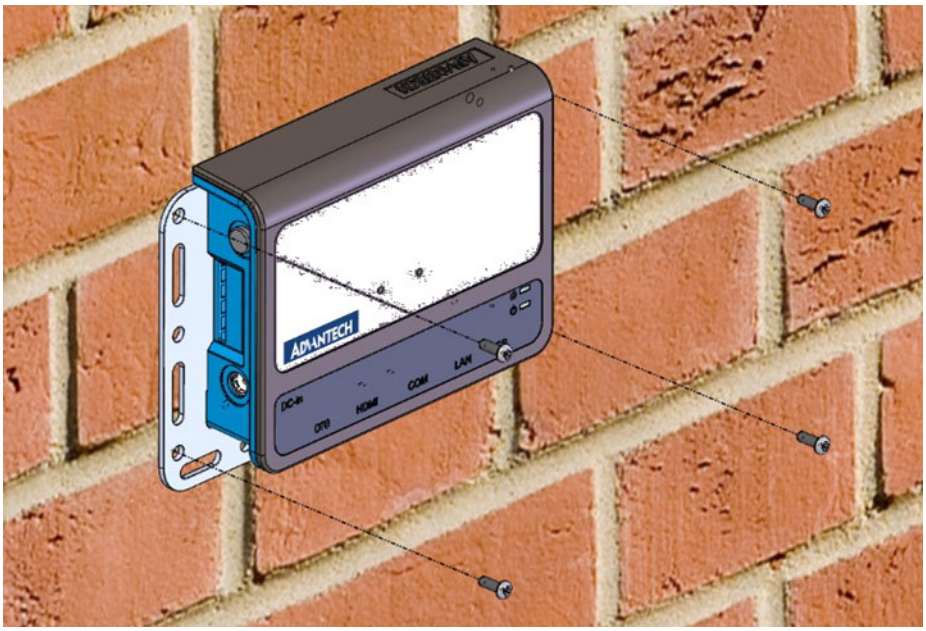


图 2.20： UBC-220 挂墙方法图

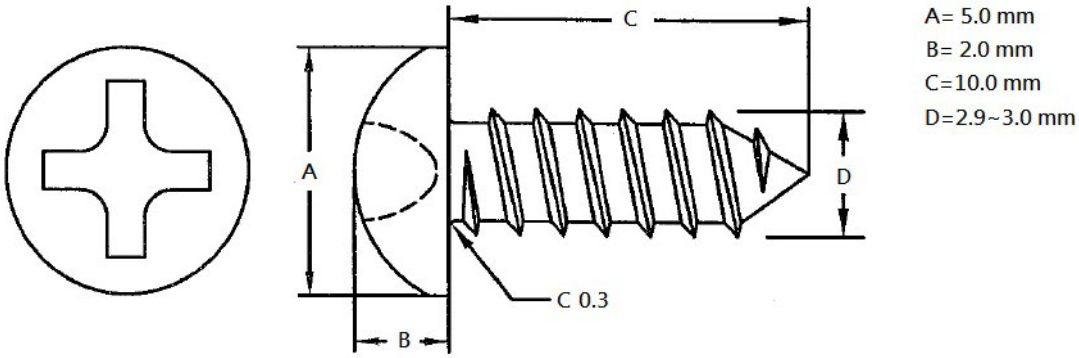


图 2.21： 配对螺丝尺寸图

Part Number	Description	Quantity
1930000721	1/4R/S D=5.5 H=2.0 + T3*10 ST Zn	4

图 2.22： 配对螺丝料号

2.4 快速启动 UBC-220

2.4.1 连接调试端口

1. 将调试端口电缆连接至 UBC-220 调试端口。
2. 将 RS-232 延长线的一端连接至调试电缆。
3. 将延长线的另一端连接至 USB 转 RS-232 电缆，然后将其连接至电脑。

2.4.2 设置调试端口

UBC-220 可通过串行电缆实现与主服务器（Windows 或 Linux）的数据通讯。用户可使用常见的串行通讯程序，如 Hyper Terminal、Tera Term 或 PuTTY。以下示例显示的是在 Windows 主机上使用 Hyper Terminal 设置串行终端设备。

1. 通过一根串行电缆将 UBC-220 连接至 Windows 主机。
2. 打开 Windows 主机上的 Hyper Terminal 程序，并按照下图所示进行设置。
3. 当引导装载程序在 SD 卡编程完毕之后，将电源适配器接口插入 UBC-220 的 DC 插孔中以启动系统。随后，引导装载程序提示信息将出现在终端设备屏幕上。

图 2.23： 终端设备的 Hyper Terminal 设置界面

2.5 测试工具

所有测试工具必须经 UBC-220 验证。验证每个 I /O 前，请务必准备好所需的测试装置。若无法获得所需测试装置，请联系研华。

2.5.1 eMMC 测试

1. 创建一个文件夹并复制到 eMMC。

```
#echo 123456789ABCDEF > test.txt
#dd if=./test.txt of=/dev/mmcblk0 bs=1024 count=1
seek=25118

 0+1 records in
 0+1 records out
 16 bytes (16 B) copied, 0.000109331 s, 146 kB/s
```

2. 检查复制到 eMMC 的数据。

```
#hexdump -C /dev/mmcblk0 -s 25720832 -s 32

 01887800  31 32 33 34 35 36 37 38  39 41 42 43 44 45 46 0a
 |123456789ABCDEF.|
 01887810  1d 4f e2 19 d3 05 8b df  ab 4a 40 5a c5 23 3c f2
 |.O.......J@Z.#<.|
```

注！ *请确认上述代码中的“seek”参数等于 25118。如果将文件夹创建到一个错误扇区，则有可能损坏系统。*

2.5.2 USB 测试

1. 插入 USB 闪存盘，并确认该设备是否出现在 UBC-220 设备列表中。
2. 创建一个文件夹并复制到 USB 闪存盘。

```
#echo 123456789ABCDEF > test.txt
#dd if=./test.txt of=/dev/sda bs=1024 count=1 seek=25118

 0+1 records in
 0+1 records out
 16 bytes (16 B) copied, 0.000109331 s, 146 kB/s
```

3. 检查复制到 USB 闪存盘的数据。

```
#hexdump -C /dev/sda -s 25720832 -s 32

 01887800  31 32 33 34 35 36 37 38  39 41 42 43 44 45 46 0a
 |123456789ABCDEF.|
  01887810  1d 4f e2 19 d3 05 8b df  ab 4a 40 5a c5 23 3c f2
 |.....................|
```

注！ *此操作可能会损坏 USB 闪存盘中存储的数据。进行该测试前，请确认使用的 USB 闪存盘中未存储重要数据。*

2.5.3 SD 测试

1. 从 eMMC 启动时，用户将看到如下目录：

```
#ls /dev/mmcblk*

  /dev/mmcblk0  /dev/mmcblk0boot0  /dev/mmcblk0boot1  /dev/
  mmcblk0p1
```

2. 将 SD 卡插入 SD 卡插槽 （SD1）并再次检查设备。此时，用户应能够看到更多目录。/dev/mmcblk1 为 SD 卡存储。

```
#ls /dev/mmcblk*

  /dev/mmcblk0      /dev/mmcblk0boot1  /dev/mmcblk1    /dev/
  mmcblk1p2
  /dev/mmcblk0boot0  /dev/mmcblk0p1     /dev/mmcblk1p1
```

3. 创建一个文件夹并复制到 SD 卡。

```
#echo 123456789ABCDEF > test.txt
#dd if=./test.txt of=/dev/mmcblk1 bs=1024 count=1
seek=25118

  0+1 records in
  0+1 records out
  16 bytes (16 B) copied, 0.000109331 s, 146 kB/s
```

4. 检查该文件是否创建成功。

```
#hexdump -C /dev/mmcblk1 -s 25720832 -s 32

  01887800  31 32 33 34 35 36 37 38  39 41 42 43 44 45 46 0a
  |123456789ABCDEF.|
   01887810  1d 4f e2 19 d3 05 8b df  ab 4a 40 5a c5 23 3c f2
  |.................|
```

注！ *请确认上述代码中的“seek”参数等于 25118。如果将文件夹创建到一个错误扇区，则有可能损坏系统*

2.5.4 LVDS/HDMI 测试

2.5.4.1 通过 gplay 测试 （适用于默认单显示屏应用）

1. 点击播放 #gplay /tools/Advantech.avi。
2. 然后，用户将看到显示屏上的视频演示。

2.5.4.2 通过 gst-launch 测试 （适用于多显示屏应用）

如果用户需要使用多台显示屏 （如 LVDS 和 HDMI 输出），必须首先在 uboot 中设置参数。有关更多详细信息，请参考第 3.7.5.3 章节。选择显示方式之后，请按照如下指示运行 gst-launch 播放视频。

1. 打开 HDMI 显示屏并选择类型：

```
#gst-launch playbin2 uri=file:///tools/Advantech.avi
video-sink="mfw_v4lsink device=/dev/video16"&
```

用户将看到两台显示屏均同时播放 Advantech.avi 视频。

若用户想要将输出音频设置为 HDMI 输出，请添加 plughw 参数。

A. Plughw:1--> 通过 HDMI 输出音频。

```
#gst-launch playbin2 uri=file:///tools/Advantech.avi
video-sink="mfw_v4lsink device=/dev/video17" audio-
sink="alsasink device=plughw:1"
```

若用户想要更换显示器，请参考下表：

video16	HDMI
video17	HDMI 重叠
video18	LVDS0
video19	LVDS1

2.5.5 Mini PCIe （3G 和 Wifi）测试

测试 3G 模块的命令如下所示，支持模块的料号为 EWM-C106HT01E。

```
#3glink

Send AT commands...
#send (AT^M)
send (ATDT*99#^M)
expect (CONNECT)
AT^M^M
OK^M
ATDT*99#^M^M
CONNECT
 -- got it
……
```

测试 WIFI 模块的命令如下所示，支持模块的料号为 EWM-W142F01E。

```
#ifconfig wlan0 up
#iwlist wlan0 scanning
#wpa_passphrase “Wifi name” password > /tmp/wpa.conf
#wpa_supplicant -BDwext -iwlan0 -c/tmp/wpa.conf
#dhclient wlan0
```

2.5.6 LED 测试

该测试用于打开 / 关闭配置的 LED 指示灯（LED2）。以下代码可作为 SWAP 的参考代码。

```
cd /sys/class/gpio/
echo 1 > export
cd /sys/class/gpio/gpio1
```

LED 打开

```
echo 0 > value
```

LED 关闭

```
echo 1 > value
```

2.5.7 OpenGL 测试

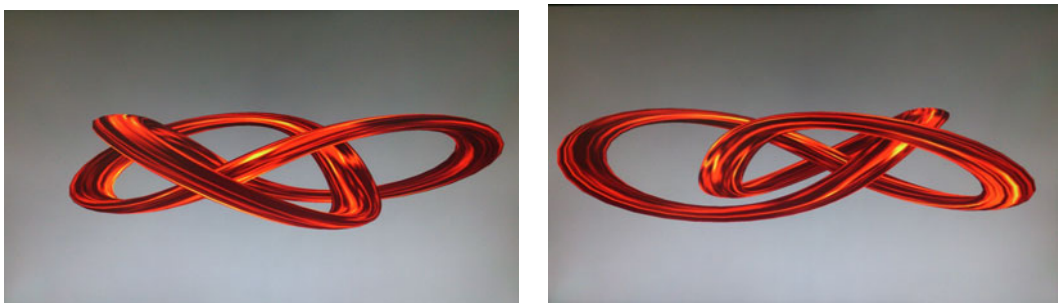
请按照以下指示测试 UBC-220 平台上的 OpenGL。

1. 将路径更改为 /opt/viv_samples/vdk。

```
#cd /opt/viv_samples/vdk
#ls tutorial*
  tutorial1       tutorial2_es20  tutorial4
  tutorial5_es20
  tutorial1_es20  tutorial3       tutorial4_es20  tutorial6
  tutorial2       tutorial3_es20  tutorial5       tutorial7
```

2. 运行 OpenGL ES 1.1 的 tutorial7。
   使用顶点缓冲区对象 （VBO）将减少传输几何数据所需的带宽，从而大幅度提升性能。任何顶点属性，如顶点坐标、法向量、颜色等，将会一次发送出去定位设备视频内存，然后根据需要绑定和使用，而无需每次都从系统内存进行读取。以下示例描述了如何创建和使用顶点缓存区对象。

```
#./tutorial7
```



3. 运行 OpenGL ES 2.0 的 tutorial3_es20。
在下图中，一个由反射材料制成的球体，以坐标原点为中心，绕 Y 轴旋转，并反射其周围的景象。

```
#./tutorial3_es20
```

2.5.8 LAN 测试

UBC-220 将 DHCP 设置为其默认网络协议。

```
#ifconfig
  eth0      Link encap:Ethernet  HWaddr 00:04:9F:01:30:E0
            inet addr:172.17.21.96  Bcast:172.17.21.255
            Mask:255.255.254.0
            UP BROADCAST RUNNING MULTICAST  MTU:1500
            Metric:1
            RX packets:129 errors:0 dropped:18 overruns:0
            frame:0
            TX packets:2 errors:0 dropped:0 overruns:0
            carrier:0
            collisions:0 txqueuelen:1000
           RX bytes:15016 (14.6 KiB)  TX bytes:656 (656.0 B)

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

如果用户想要手动配置 IP，请使用如下命令：

```
#ifconfig eth0 xxx.xxx.xxx.xxx up
```

以下命令为可供用户参考的真实案例。其中，主机（UBC-220）IP 为 172.17.21.97；目标（台式计算机）IP 为 172.17.20.192。

```
#ifconfig eth0 172.17.21.97 up
#ifconfig eth0
```

```
eth0      Link encap:Ethernet  HWaddr 00:04:9F:01:30:E0
          inet addr:172.17.21.97  Bcast:172.17.255.255
          Mask:255.255.0.0
          UP BROADCAST RUNNING MULTICAST  MTU:1500
          Metric:1
          RX packets:2851 errors:0 dropped:271 overruns:0
          frame:0
          TX packets:30 errors:0 dropped:0 overruns:0
          carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:291407 (284.5 KiB)  TX bytes:2000 (1.9
          KiB)
```

目标计算机（客户端）IP 地址为 172.17.20.192，因此用户可使用如下命令查看是否能接收到来自客户端的响应。

```
#ping 172.17.20.192

  PING 172.17.20.192 (172.17.20.192): 56 data bytes
  64 bytes from 172.17.20.192: seq=0 ttl=128 time=7.417 ms
  64 bytes from 172.17.20.192: seq=1 ttl=128 time=0.203 ms
  64 bytes from 172.17.20.192: seq=2 ttl=128 time=0.300 ms

  --- 172.17.20.192 ping statistics ---
  3 packets transmitted, 3 packets received, 0% packet loss
  round-trip min/avg/max = 0.203/2.640/7.417 ms
```

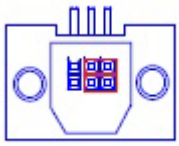
2.5.9 RS232 测试

从下面代码中用户可以得知 UBC-220 支持 2 个 RS-232 端口。其中，/dev/ttymxc0 为 UBC-220 调试端口（UBC-220 DEBUG_CONSOLE）预留，另一个可供用户使用。

```
#setserial -g /dev/ttymxc*

  /dev/ttymxc0, UART: undefined, Port: 0x0000, IRQ: 58
  /dev/ttymxc1, UART: undefined, Port: 0x0000, IRQ: 59
```

以下测试需使用 4 个点距为 2.54 mm 的 mini 跳线完成。该跳线的料号为 1653003100。Mini 跳线将作为连接 Tx 和 Rx 的桥梁。



2.5.9.1 /dev/ttymxc1 测试（COM）

```
#stty -F /dev/ttymxc1 -echo
#cat /dev/ttymxc1
[CTRL+Z]
#echo hello > /dev/ttymxc1
#fg
  Hello
[CTRL+C]
```

2.5.10 看门狗定时器测试

1. 执行 ‘ wdt_driver_test.out’ 命令。

```
./wdt_driver_test.out 5 10 0
  Usage: wdt_driver_test <timeout> <sleep> <test>
  timeout: value in seconds to cause wdt timeout/reset
  sleep: value in seconds to service the wdt
  test: 0 - Service wdt with ioctl(), 1 - with write()
```

2. 请尝试输入以下命令将超时时间设置为 10 秒，之后系统将重启。

```
#/unit_tests/wdt_driver_test.out 5 10 0
  Starting wdt_driver (timeout: 10, sleep: 5, test: ioctl)
  Trying to set timeout value=10 seconds
  The actual timeout was set to 10 seconds
  Now reading back -- The timeout is 10 seconds
```

按 [CTRL+C] 键，将看到如下结果：

```
imx2-wdt imx2-wdt.0: Unexpected close: Expect reboot!
```

然后，系统将在 10 秒内重启。

2.5.11 图片演示测试

执行如下命令运行 UBC-220 上的图片演示应用。

```
#cd /tools
#./fbv Advantech.jpg
```

然后，用户将看到默认显示屏幕上出现演示图片。

2.5.12 LED 测试

以下为适用于系统 LED2 的测试脚本。

```
#cd /sys/class/gpio/
#echo 1 > export
#cd /sys/class/gpio/gpio1
```

LED 打开

```
#echo 0 > value
```

LED 关闭

```
#echo 1 > value
```

第 3 章

软件功能

本章将详细介绍 UBC-220 平台上 Linux 操作系统的信息。

3.1 简介

UBC-220 平台为一款内置 Linux 3.0.35 内核的嵌入式系统，其中包含 UBC-220 平台系统所需的所有 shell 命令及驱动程序。研华不提供 UBC-220 BSP 的 IDE 开发环境，用户可在 Ubuntu 10.04LTS 环境下自行评估并开发。

Linux 的三个主要启动组件为 “u-boot.bin”、“uImage” 和 “File System”。其中，“u-boot.bin”用于初始化外围硬件参数；“uImage”为Linux内核镜像；“File System” 用于 Linux 操作系统。

若启动媒介（SD 卡、SATA HDD 或板载闪存）中缺失上述任何一个文件，则无法成功启动至 Linux 环境。

本章旨在向用户介绍 UBC-220 的软件开发信息，以便用户能够高效开发所需应用。

UBC-220 产品专为 Linux 系统定制设计。目前，官方支持的主机版本为 Ubuntu 10.04 LTS，使用其它版本可能会导致兼容性问题。因此，研华强烈建议用户评估 / 开发 UBC-220 之前在主机上安装 Ubuntu 10.04 LTS。

3.2 包内容

研华为 UBC-220 提供了两种不同的 Linux 包：一种为用于恢复系统的预建系统镜像，另一种为源代码包 （BSP）。

3.2.1 源代码包

UBC-220 源代码包 （BSP）包含交叉编译程序、Linux 源代码、Uboot 源代码、根文件系统以及操作系统环境使用的一些脚本。其中一些组件由研华开发，而其余则由开源社区开发。UBC-220 源代码包由六个主要文件夹构成：

“cross_compiler”、“document”、“image”、“package”、“scripts” 及 “source”。

注！ *UBC-220 源代码包 （BSP）为研华自主知识产权。用户如需访问该源代码包，请联系研华技术支持。*

U220LBV2660 包内容的描述如下：

- “cross_compiler” --> 该文件夹包含交叉编译程序的源代码。
- “document” --> 该文件夹包含使用手册。
- “image” --> 该文件夹包含 uImage、u-boot_crc.bin、u-boot_crc.bin.crc。
- “image/rootfs” --> 该文件夹包含 Linux 根文件系统。
- “package” --> 该文件夹包含 Freescale 提供的源代码，未经任何修改。
- “scripts” --> 该文件夹包含用于自动配置系统和编译图片的脚本。
- “source” --> 该文件夹包含研华自有源代码。

3.2.1.1 cross_compiler

用户可使用交叉编译程序工具链编译 uImage 及相关应用程序（gcc 的版本为 4.6.2 20110630）。

工具链目录结构如下所示：

```
|-- bin // 带前缀的工具链，如 arm-none-linux-gnueabi-gcc 等
|-- lib // 用于工具链自身，而非应用程序的库文件
|-- arm-fsl-linux-gnueabi
      |-- bin // 带前缀的工具链，如 gcc.-|-- debug-root // 所有调试工具
      |-- multi-libs // 所有函数库及头文件
      |-- armv5 // 用于 armv5 （i.mx 2xx）的函数库，仅支持软件浮点
      |-- armv6 // 用于 armv6 （i.mx 3xx）的函数库，软件 fpu 版本
      |-- armv7-a // 用于 armv7-a（i.mx5xx 和 i.mx6xx）的函数库，硬件 fpu 版本
|-- lib // 默认函数库，可用于 armv4t 及以上版本。
      |-- usr
      |-- include // 用于应用开发的头文件
      |-- lib // 第三方函数库及 Freescale 的静态函数库
```

3.2.1.2 document

使用手册，指导用户如何建立开发环境。

3.2.1.3 image

该文件夹包括 uImage 和 u-boot。

3.2.1.4 image/rootfs

Linux 采用分层文件系统（HFS）。image/rootfs 是 Linux 文件系统的树状结构最高等级。

“rootfs” 包含的主要文件夹如下所示：

- bin --> 普通程序，由系统、系统管理员和用户共享。
- dev --> 包含所有 CPU 外围设备的参考，由具备特殊属性的文件表示。
- etc --> 包含最重要的系统配置文件。该目录中包含的数据类似 Windows 系统的控制面板数据。
- home --> 普通用户的根目录。
- lib --> 库文件，包括系统和用户所需的所有种类程序文件。
- mnt --> 外部文件系统的标准安装点。
- opt --> 通常包含额外和第三方软件。
- proc --> 一个包含系统资源相关信息的虚拟文件系统。如需了解 proc 中文件的更多含义，用户可在终端窗口中输入命令 man proc。proc.txt 文件详细介绍了虚拟文件系统。
- root --> 管理用户的根目录。请注意 /, （根目录）与 /root, 根用户主目录）之间的区别。
- sbin --> 系统和系统管理员使用的程序。
- sys --> Linux sys 文件系统。
- tmp --> 系统使用的临时存储空间，系统重启时将清理。请勿存储重要数据！
- unit_tests --> Freescale i.MX6 产品提供的单元测试工作。
- usr --> 所有用户相关程序的程序、函数库及文件等。

- var --> 存储用户创建的所有变量文件和临时文件，如日志文件、邮件队列、打印机假脱机区域以及从网络下载的临时存储文件空间。
- tools --> 仅用于抽样测试。

3.2.1.5 scripts

研华提供的一些脚本，旨在帮助用户快速配置系统或构建图像。提供的脚本如下所示：

- setenv.sh --> 用于快速设置开发环境的脚本。
- cfg_uboot.sh --> 用于快速配置 “u-boot” 构建设置的脚本。
- mk_uboot.sh --> 用于构建 “u-boot” 并在构建之后将 “u-boot” 复制到 “image” 文件夹的脚本。g.
- cfg_kernel.sh --> 用于快速配置内核构建设置的脚本。
- mk_kernel.sh --> 用于构建 “uImage” 并在构建之后将 “uImage” 复制到 “image” 文件夹的脚本。
- mksd-linux.sh --> 用于在用户构建图像时设置可启动 SD 卡的脚本。

3.2.1.6 source

该文件夹包含名为 “linux-3.0.35” 和 “u-boot-2009.08” 的两个子目录，其中分别包含为 Linux 内核和 U-boot 的源代码。

Linux 是 UNIX 操作系统的克隆体，具备现代成熟 UNIX 系统的所有特性，包括真正多任务处理、虚拟内存、共享函数库、按需加载、共享写时复制可执行文件、正确内存管理以及多任务网络 （包括 IPv4 和 IPv6）。

如果 Linux 系统支持分页式内存管理单元（PMMU）以及 GNU C 编译器（GCC）端口（GNU 编译器套装的一部分），便可轻松移植到最通用的 32/64 位架构。尽管移植后功能明显受限，Linux 仍然在不支持 PMMU 的情况下成功移植到大量不同架构中。此外，Linux 也支持移植到其自身架构。

“linux-3.0.35” 文件夹下的主要子目录如下所示：

- arch --> 硬件平台的相关项目，其中大部分用于 CPU。
- block --> 模块的设置信息。
- crypto --> 内核支持的加密技术。
- Documentation --> 内核文件。
- drivers --> 硬件驱动。
- firmware --> 一些用于旧版硬件的固件数据。
- fs --> 内核支持的文件系统。
- include --> 其它程序的头文件定义。
- init --> 内核的初始功能。
- ipc --> 定义 Linux 操作系统中每个程序的通讯。
- kernel --> 定义内核进程、状态、日程和信号。
- lib --> 一些函数库。
- mm --> 内存相关数据。
- net --> 网络相关数据。
- security --> 安全设置。
- sound --> 音频相关模块。
- virt --> 虚拟机相关数据。

对于 Linux 特定问题和一般 UNIX 问题，用户可以从网络、书籍和杂志等渠道获得大量的可用文件或材料，轻松找到答案。

此外，./source/linux-3.0.35/Documentation 目录下还提供各种 README 文件，可帮助用户找到特定内核的驱动安装和注意信息。用户还可参考./ source/linux-3.0.35/Documentation/00-INDEX 了解每份 README/note 的目标宗旨。

3.3 配置构建环境

本份用户手册中的所有指示均针对 Ubuntu 10.04 LTS 开发环境。请用户预先在 PC/NB 上安装 Ubuntu 10.04 LTS。

获取到 UBC-220 Linux 源代码包之后，请按照以下指示将其解压缩至开发环境：

1. 将 “U220LBV2660” 包复制到桌面。
2. 打开 Ubuntu 10.04 LTS 上的 “Terminal”。
3. **$sudo su**（修改为 “root” 权限）。
4. 输入用户密码。
5. **#cd Desktop/**
6. **#tar xvf U220LBV2660.tgz**（解压缩文件）

研华为用户提供了快速配置开发环境的脚本。用户可按照以下步骤配置开发环境：

1. 打开 Ubuntu 10.04 LTS 上的 “Terminal”。
2. **$sudo su**（修改为 “root” 权限）。
3. 输入用户密码。
4. 将目录修改为 BSP 脚本文件夹。
5. **#. setenv.sh**（自动配置开发环境）。
6. 然后，用户即可开始编写源代码、构件图像或编译应用程序。

3.3.1 setenv.sh

该脚本用于快速配置开发环境。该脚本将配置系统的文件夹路径。如果用户已经添加 / 修改了文件夹和路径，便可自行添加 / 修改 setenv.sh 脚本。

setenv.sh 脚本内容的主要部分如下所示：

```
export SRCROOT=${PWD}/..
export CC_PATH=${SRCROOT}/cross_compiler/fsl-linaro-toolchain
export CROSS_COMPILE=${CC_PATH}/bin/arm-none-linux-gnueabi-
export CC=${CROSS_COMPILE}gcc
export STRIP=${CROSS_COMPILE}strip
export ARCH=$rm
export KROOT=${SRCROOT}/source/linux-3.0.35
export UBOOT_SOURCE=${SRCROOT}/source/u-boot-2009.08
export ROOTFS=${SRCROOT}/image/rootfs
export LOG=${SRCROOT}/Build.log
export PATH=${CC_PATH}/bin:${UBOOT_SOURCE}/tools:$PATH
```

注！ *用户每次打开新的 “Terminal” 实用程序时，必须将 “setenv.sh” 用标记包裹起来，如 #source setenv.sh。*

注！ *建议用户在使用源代码前，将权限修改为 “root” 权限。*

3.4 构建指示

本节将指导用户如何构建 u-boot & Linux 内核。

3.4.1 构建 u-boot 镜像

研华已经为用户编写了一份脚本文件，以便快速构建 u-boot 镜像。用户可按照以下步骤构建 u-boot 镜像：

1. 打开 Ubuntu 10.04 LTS 上的 “Terminal”。
2. **$sudo su**（修改为 “root” 权限）。
3. 输入用户密码。
4. **#. setenv.sh**（自动配置开发环境）。
5. **#./cfg_uboot.sh imx6dl_ubc220_config**（设置自动配置 u-boot）
6. **#./mk_uboot.sh**（开始构建 u-boot）
7. 之后，用户在 ../image 文件夹中即可找到正在构建的 u-boot_crc.bin 和 u-boot_crc.bin.crc 文件。

3.4.2 构建 Linux 内核镜像

研华已经为用户编写了一份脚本文件，以便快速构建 uImage 镜像。用户可按照以下步骤构建 uImage 镜像：

1. 打开 Ubuntu 10.04 LTS 上的 “Terminal”。
2. **$sudo su**（修改为 “root” 权限）。
3. 输入用户密码。
4. 将目录修改为 BSP 脚本文件夹。
5. **#. setenv.sh**（自动配置开发环境）。
6. **#./cfg_kernel.sh imx6dl_ubc220_defconfig**（设置自动配置 uImage）
7. **#./mk_kernel.sh**（开始构建 uImage）
8. 之后，用户在 ../image 文件夹中即可找到正在构建的 uImage 镜像。

3.4.3 构建日志

用户可在 “U220LBV2660” 文件夹中找到构建日志文件。如果用户在构建 Linux 内核过程中收到任何错误提示，建议查看日志文件了解更多详细信息。

3.5 源代码修改

本节将指导用户如何使用 Linux 源代码。在本节中，用户将看到几个使用 BSP 源代码的示例。

3.5.1 通过 menuconfig 向内核添加驱动

用户可通过 menuconfig 向内核添加驱动。下面这个示例将指导用户如何向 Linux 内核添加一个 RTC 驱动 （日本精工仪器 S-35390A）。请按照以下步骤进行：

1. 打开 Ubuntu 10.04 LTS 上的 “Terminal”。
2. **$sudo su**（修改为 “root” 权限）。
3. 输入用户密码。
4. 将目录修改为 BSP 脚本文件夹。
5. **#. setenv.sh**（自动配置开发环境）。
6. **#./cfg_kernel.sh menuconfig**
7. 然后，用户将看到如下 GUI 屏幕界面 （Linux Kernel Configuration）：

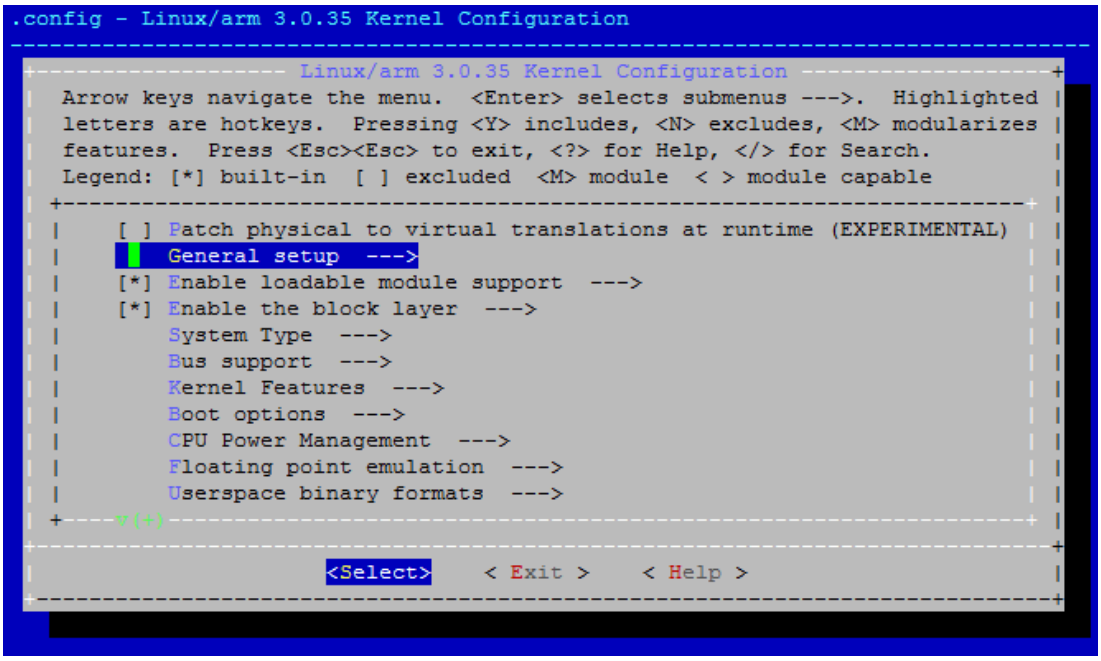


图 3.1： Linux 内核配置界面

8. 选择 “Device Drivers” -->” Real Time Clock”，用户将看到列表中的 “Seiko Instruments S-35390A” 选项。选择该项，然后退出并保存配置信息。

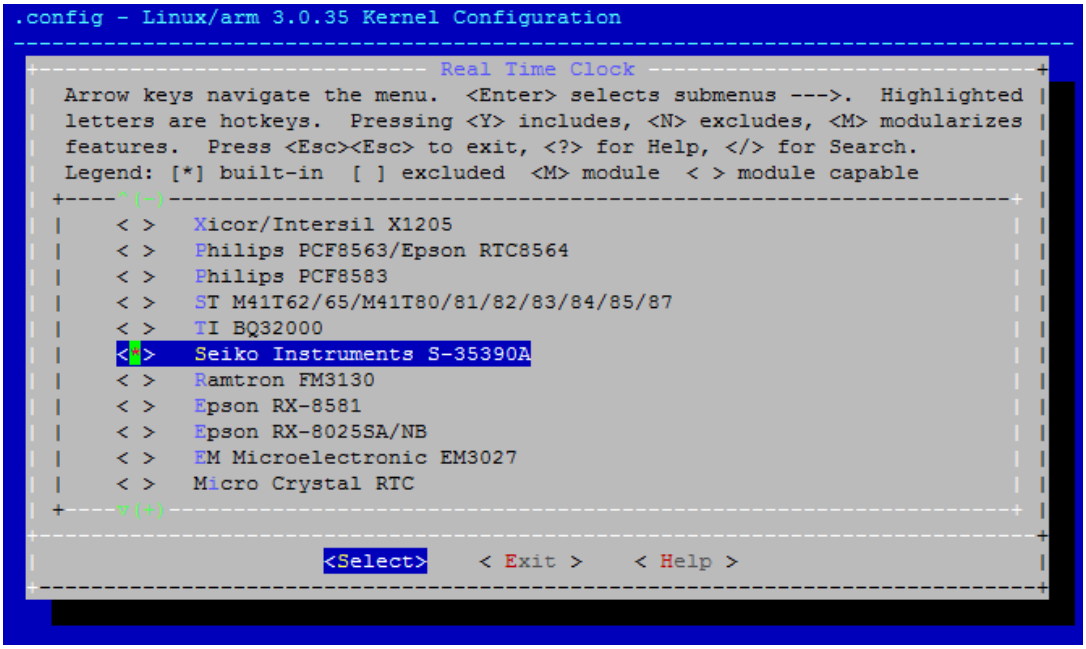


图 3.2： 选择 “Seiko Instruments S-35390A”

9. 将目录修改为 “source/linux-3.0.35/arch/arm/mach-mx6”，并编辑 “board-mx6dl_ubc220.h” 和 “board-mx6q_advantech.c” 文件。
请将如下代码添加至 source/linux-3.0.35/arch/arm/mach-mx6/board-mx6dl_ubc220.h：

```
static struct i2c_board_info mxc_i2c0_board_info[] __initdata
= {
        {
```

```
                I2C_BOARD_INFO("adv-wdt-i2c", 0x29),
        },
        {
                I2C_BOARD_INFO("s35390a", 0x30),
        },
};
```

请将如下代码添加至
source/linux-3.0.35/arch/arm/mach-mx6/board-mx6q_advantech.c

```
i2c_register_board_info(0, mxc_i2c0_board_info,
                        ARRAY_SIZE(mxc_i2c0_board_info));
```

10. 完成上述操作后，请参考第 3.3.2 章节重新构建安装有 RTC 驱动 （日本精工仪器 S-35390A）的内核。

注！ *如果用户无法从列表中找到设备驱动，请联系硬件供应商。*

3.5.2 更换 UBC-220 启动徽标

默认情况下 ,UBC-220 在启动时会显示启动徽标。如有需要，用户可按照以下步骤更换启动徽标：

1. 首先，用户必须从网络下载操作系统版本相对应的 “netpbm”。
2. 然后，通过输入 **$sudo apt-get install netpbm**. 安装 “netpbm” 。
3. 准备好需要更换的启动徽标，如 bootlogo.png （在 Desktop/boot- logo 文件夹中）。

注！ *徽标图片文件必须为 PNG 格式，且色彩低于 224 色。建议使用与用户 LCD 显示屏分辨率相同的图片文件。*

4. 打开 Ubuntu 10.04 LTS 上的 “Terminal”。
5. **$sudo su** （修改为 “root” 权限）。
6. 输入用户密码。
7. **#cd Desktop/bootlogo** （进入 bootlogo.png 文件所在的文件夹）。
8. **#pngtopnm bootlogo.png | ppmquant 224 | pnmtoplainpnm > logo_linux_clut224.ppm**。
9. 将 **logo_linux_clut224.ppm** 复制到 **source/linux-3.0.35/drivers/video/logo/** 目录。
10. 完成上述操作后，请参考第 3.3.1 章节重新构建带有用户徽标 （日本精工仪器 S-35390A）的内核。

3.6 创建 Linux 系统启动媒介

UBC-220 支持从 SD 卡和板载闪存启动。本节将指导用户如何为 UBC-220 Linux 系统启动媒介构建图像。

3.6.1 创建 Linux 系统 SD 卡

3.6.1.1 从源代码包创建

收到 UBC-220 Linux 源代码包后，用户可按照以下步骤创建 Linux 系统 SD 卡，以便从 SD 卡启动系统。

1. 打开 Ubuntu 10.04 LTS 上的 “Terminal”。
2. **$sudo su**（修改为 “root” 权限）。
3. 输入用户密码。
4. 将 SD 卡插入到开发计算机。
5. 检查 SD 卡位置路径，如：/dev/sdf
6. 将目录修改为 BSP 脚本文件夹。
7. **#./mksd-linux.sh /dev/sdf**
8. 输入 “y” （开始复制文件，等待直至屏幕显示 [Done]），然后将 Linux 系统 SD 卡插入到 UBC-220 的 SD 卡插槽 （SD1），即可启动到 Linux 操作系统。

3.6.2 从板载闪存启动

如果用户已经拥有一块 Linux 系统 SD 卡，请按照以下步骤将卡上内容复制到板载闪存，再从板载闪存启动系统。研华为用户提供了一份 “mkinand-linux.sh” 脚本，以便快速将系统镜像安装至板载闪存。

1. 请参考第 3.5.1 章节创建 Linux 系统 SD 卡。
2. 将 Linux 系统 SD 卡插入到 ROM-DB7500 并连接串行控制台。
3. 在 UBC-220 平台上，输入 #root (Login)。
4. 在 UBC-220 平台上，输入 #cd /mk_inand。
5. 在 UBC-220 平台上，输入 #./mkinand-linux.sh /dev/mmcblk0。
6. 在 UBC-220 平台上，输入 “y” （开始复制文件，等待直至屏幕显示 [Done]）。
7. 关闭系统电源并移除 SD 卡，这样即可在无 SD 卡的情况下从板载闪存启动系统。

3.7 调试信息

UBC-220 可通过控制台电缆和调试端口适配器连接至主机 PC （Linux/Windows）。若要实现与主机 PC 的通讯功能，必须安装串行通讯程序，如 HyperTerminal、Tera Ter 或 PuTTY。请按照以下步骤建立 Windows 主机上的串行控制台 “HyperTerminal”：

1. 通过串行电缆、调试端口适配器和控制台电缆将 UBC-220 连接至 Windows PC。
2. 打开 Windows PC 上的 HyperTerminal，并按照图 3.3 所示进行设置。
3. 按 “POWER” 键启动，用户将看到终端界面上出现引导装载程序提示。

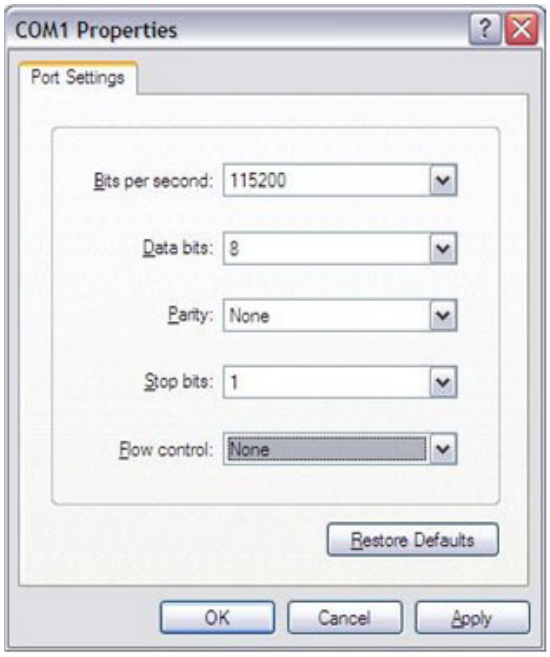


图 3.3： 串行控制台的 HyperTerminal 设置

3.8 Linux 软件 AP 及 UBC-220 的测试

本节将指导用户如何在 Linux 环境下开发应用程序。首先，将出现 “Hello World” 示例应用；然后，用户将看到一些预安装的测试程序。这些测试程序将在本节中详细介绍。

3.8.1 “Hello World!” 应用和执行

本节将指导用户如何编写示例应用 “Hello World”。用户可按照以下步骤编写：

1. 打开 Ubuntu 10.04 LTS 上的 “Terminal”。
2. **$sudo su**（修改为 “root” 权限）。
3. 输入用户密码。
4. 将目录修改为 BSP 脚本文件夹。
5. **#. setenv.sh**（自动配置开发环境）。
6. **#cd ../source**
7. **#mkdir helloworld**（在桌面上创建工作目录）。
8. **#cd helloworld**（进入工作目录）。
9. **#gedit helloworld.c**（创建一个新的 C 源文件）。

使用以下源代码编辑 helloworld.c：

```
#include <stdio.h>
void main()
{
 printf("Hello World!\n");
}
```

10. 保存文件并退出。
11. **#$CC -o helloworld helloworld.c**（编译 helloworld.c）
12. 这时，用户将看到 “helloworld” 出现在当前目录。
13. 将 Linux 系统 SD 卡插入到开发计算机。

14. **#cp helloworld /media/rootfs/tool**（/media/rootfs 为 Linux 系统 SD 卡的安装点）
15. 移除 SD 卡并将其插入到 UBC-220，打开串行控制台。
16. 在 UBC-220 平台上，输入 **#root** (Login)。
17. 在 UBC-220 平台上，输入 **#cd /tool**。
18. 在 UBC-220 平台上，输入 **#./helloworld**。
19. 这时，用户即可在 UBC-220 屏幕上看到“Hello World!”。

3.8.2 看门狗定时器示例代码

看门狗定时器（WDT）示例代码如下所示：

```
#include <stdio.h>
#include <stdlib.h>
#include <fcntl.h>
#include <linux/watchdog.h>
#include <sys/ioctl.h>
#include <unistd.h>

void help_info(void);
int main(int argc, const char *argv[])
{
  int fd, timeout, sleep_sec, test;
  int count=1;
  if (argc < 2) {
   help_info();
   return 1;
  }
  timeout = atoi(argv[1]);
  sleep_sec = atoi(argv[2]);
  if (sleep_sec <= 0) {
   sleep_sec = 1;
   printf("correct 0 or negative sleep time to %d seconds\n",
          sleep_sec);
  }
  test = atoi(argv[3]);
  printf("Starting wdt_driver (timeout: %d, sleep: %d, test:
              %s)\n",
         timeout, sleep_sec, (test == 0) ? "ioctl" : "write");
  fd = open("/dev/watchdog", O_WRONLY);
  if (fd == -1) {
   perror("watchdog");
   exit(1);
  }
  printf("Trying to set timeout value=%d seconds\n", timeout);
  ioctl(fd, WDIOC_SETTIMEOUT, &timeout);
  printf("The actual timeout was set to %d seconds\n",
              timeout);
  ioctl(fd, WDIOC_GETTIMEOUT, &timeout);
  printf("Now reading back -- The timeout is %d seconds\n",
              timeout);
  while (1) {
   printf("WDT Time out counter:%d\n",count);
   if ((test !=0) && (test ==count)) {
      printf("Ping Watchdog (reset wdt)\n");
      ioctl(fd, WDIOC_KEEPALIVE, 0);
```

```
            test=0;
            count=0;
        }
        sleep(sleep_sec);
        count+=sleep_sec;
      }
      return 0;
}

void help_info(void)
{
   printf("Usage: wdt_driver_test <timeout> <sleep>
                <trigger>\n");
   printf("    timeout: value in seconds to cause wdt timeout/
                reset\n");
   printf("    sleep: value in seconds to display wdt
                timeout\n");
   printf("    trigger: value in seconds to ping the wdt\n");
}
```

如果用户想要修改 WDT 时间，请将代码修改为：

```
ioctl(fd, WDIOC_SETTIMEOUT, &timeout).
```

3.8.3 RS232 初始代码

RS232 初始代码如下所示，可指导用户如何初始化 COM2 端口。

```
int open_port(void)
{
    int fd;
    fd=open("/dev/ttymxc1",O_RDWR|O_NOCTTY|O_NDELAY);
    if(fd == -1){
        perror("open error");
    }
    return(fd);
}
```

3.8.4 显示输出设置

3.8.4.1 单显示设置

关于 HDMI 输出，请在 u-boot 中进行如下设置：

```
setenv bootargs_base 'setenv bootargs console=ttymxc0,115200
enable_wait_mode=off
video=mxcfb1:off    video=mxcfb0:dev=hdmi,1920x1080M@60,if=RGB24
video_mode=display3 pcie_testmode=off'
```

关于 LVDS （单）输出，请在 u-boot 中进行如下设置：

```
setenv bootargs_base 'setenv bootargs console=ttymxc0,115200
enable_wait_mode=off
video=mxcfb1:off       video=mxcfb0:dev=ldb,800x480M@60,if=RGB24
video_mode=display3 pcie_testmode=off'
```

3.8.4.2 双显示设置

如果用户想要显示 LVDS 和 HDMI 输出，请在 u-boot 中设置相应参数。以下为 U-boot 中的默认设置。

```
setenv bootargs_base 'setenv bootargs console=ttymxc0,115200
enable_wait_mode=off video_mode=display3 pcie_testmode=off'
```

显示界面时钟有如下几个选项 （每个端口独立）：

1. 衍生自 IPU 内部时钟 （主模式）。
2. 由外部资源提供 （从模式）。
3. 支持的传输速率。

单端口正在传输时，像素时钟速率最高可达 264 MHz ；

两个 LVDS 端口同时传输时，需满足以下条件：

1. 每个端口的像素时钟速率最高可能为 220 MHz** ；
2. 两个端口的总像素时钟速率最高达 240 MHz。

注! *特定像素时钟频率适用于内部时钟，但是可能受到 IO 缓冲速度能力限制。最终数值受 AC 特性影响。*

3.8.5 网络设置

默认：从 DHCP 获取 IP。

手动：通过以下命令设置 IP。

```
#ifconfig eth0 192.168.0.1 up
```

ifconfig 用于配置网络接口，手动设置页面如下所示：

```
SYNOPSIS
       ifconfig [-v] [-a] [-s] [interface]
       ifconfig [-v] interface [aftype] options | address ...
OPTIONS
       -a      显示所有当前可用接口 （即使当机）
       -s      显示简要清单，如 like netstat -i
       -v      对于一些错误条件接口会更加冗长
               接口名称。通常由一个驱动名，紧跟着一个单位名构成。例如，
               eth0 表示第一个以太网接口。如果内核支持别名接口，用户可将
               eth0:0 作为 eth0 的第一个别名。用户还可通过这些接口分配另一
               个地址。若要删除别名接口，使用 ifconfig eth0:0 即可。注：在
               每个作用域 （如，采用地址 / 子网掩码组合的相同网络）中，如
               果第一个 （主要）别名被删除，那么所有别名均同时删除。
      [aftype]
      up       该标记将导致接口被激活。
               如果为某个接口分配了一个地址，那么就代表为其隐式指定了一个
               标记。
      down     该标记将导致该接口的驱动关闭。
      address  分配给该接口的 IP 地址。
      netmask [addr]
               设置该接口的 IP 网络掩码。该项数值默认为一般类别A、B或C网络
               掩码 （衍生自接口 IP 地址），但是可设置为任意值。
      broadcast [addr]
               若给定地址引数，请设置该接口的协议广播地址；或者，设置（或
               清除）该接口的 IFF_BROADCAST 标记。
      del addr/prefixlen
```

从接口中移除 IPv6 地址。

3.8.6 存储 （eMMC/SD 卡）

存储设备名称如下表所示：

设备	名称
eMMC	/dev/mmcblk0
SD card	/dev/mmcblk1

3.8.7 3G 示例代码

如下为 3glink 代码，用户已在 3G 测试 （第 2.5.5 章节）中使用过相同代码。

```
#!/bin/bash

echo "Send AT commands..."

pppd connect 'chat -v -s -t 10 "" "AT" "" "ATDT*99#" "CON-
NECT" ""' user username password password /dev/ttyUSB2
460800 nodetach crtscts debug usepeerdns defaultroute &
```

第 4 章

系统恢复

本章介绍如何在意外损坏的情况下恢复 Linux 操作系统。

4.1 简介

本节介绍恢复 eMMC 镜像的详细步骤。如果板载闪存镜像意外损坏，用户可按照以下步骤恢复系统。

1. 打开 Ubuntu 10.04 LTS 上的 “Terminal”。
2. **$sudo su**（修改为 “root” 权限）。
3. 输入用户密码。
4. 将 SD 卡插入到开发计算机。
5. 检查 SD 卡位置路径，如：/dev/sdf。
6. 将目录修改为 BSP 脚本文件夹。
7. **#./mksd-linux.sh /dev/sdf**。
8. 输入 “y”（开始复制文件，等待直至屏幕显示 [Done]）。
9. 将控制台电缆连接至调试端口（CN1）并打开 Ubuntu 10.04 LTS 上的串行控制台程序，将波特率设置为 115200。有关控制台设置的详细信息，请参考第 3.6 章节。
10. 在 UBC-220 平台上，输入 **#root** (Login)。
11. 在 UBC-220 平台上，输入 **#cd /mk_inand**。
12. 在 UBC-220 平台上，输入 **#./mkinand-linux.sh /dev/mmcblk0**。
13. 在 UBC-220 平台上，输入 “**y** ”（开始复制文件，等待直至屏幕显示 [Done]）。
14. 关闭系统电源并移除 SD 卡。

第 5 章

研华服务

本章主要介绍 UBC-220 的设计支持服务、技术支持和质保政策。

5.1 RISC 设计支持服务

随着工业计算的迅速发展，大量新型应用百花齐放，引导了一场 IPC 行业的根本性变革。过去，系统集成商 （SI）只能在没有外部援助的情况下独立完成项目；而今天，这种工作模式已经一去不复返。随着市场需求日趋多元化以及市场竞争日益激烈，垂直整合模式的合作将成为更加高效的工作方式，为客户创造更具竞争性的优势。因此，基于 ARM 的 CPU 模块产品应运而生。这类产品将所有必需组件设计于 CPU 模块之中，而将其它部分放置于载板之上以满足特殊市场需求，在大幅度提升产品灵活性的同时，还保持了低功耗的优势。

研华已经在工业计算机行业深耕数年，发现客户在实施模块化设计时通常会遇到以下问题。

常规 I/O 接口设计能力

尽管客户有能力进行垂直集成，且在其专业应用领域具备资深经验及核心竞争力，然而却缺乏常规电源和 I/O 接口设计技术和经验，最终导致其设计过程面临诸多挑战，特别是关于如何将 CPU 模块完美集成至载板。

信息采集

即使客户自身有能力获取足够信息，在面临专业化垂直应用时做出正确决策；然而，在一般平台设计以及与 CPU 或芯片组制造商的沟通过程中，一些客户仍然可能会感到棘手，这就增加了载板设计难度和风险，导致产品延迟上市，最终造成商机旁落。

软件开发与修改

相比 x86 结构而言，RISC 架构采用更加简化的指令集，因此支持 x86 平台的软件不可用于 RISC 平台之上。在这种情况下，系统集成商就必须自行开发系统软件，并完成后续软硬件集成工作。与 x86 平台不同，RISC 平台支持相对较少的板级支持包 （BSP）及驱动。而即使 RISC 平台提供驱动支持，系统集成商仍需投入大量精力将其集成至系统核心。此外，由于 CPU 制造商提供的 BSP 通常仅适用于载板设计，因此如何构建软件开发环境也成为系统集成商面临的一大难题。

有鉴于此，研华提出了针对基于 RISC 模块化电脑（COM）的标准化设计支持服务概念。研华拥有专业的设计支持服务团队，有能力帮助客户设计载板并解决棘手问题。研华的服务不仅能够帮助客户高效分配资源，而且还可减少研发人力成本和硬件投资。

凭借着与多家行业领先 CPU 和芯片组原始制造商 （如 ARM、TI 和 Freescale）的密切合作关系，研华能够有效帮助客户实现顺畅沟通，解决技术支持难题，并减少产品开发过程中的不确定因素。研华的专业软件团队还着重于开发完整的板级支持包，帮助客户建立针对 RISC 平台的软件开发环境。

通过标准化的 RISC 设计支持服务，研华将助客户一臂之力，迅速将产品推向市场。

研华遵循一贯的高效化多级开发流程，包括规划、设计、整合和验证阶段。RISC 设计支持服务可为以下不同阶段提供全面支持：

规划阶段

在决定选用研华 RISC COM 产品之前，客户必须进行全方位的调查，包括产品特性和规格，并进行软件兼容性测试。对此，研华提供 RISC 客户解决方案板卡（CSB）作为载板评估工具。研华会在开发 RISC COM 产品过程中同时设计载板。在规划阶段，客户可采用这种评估板评估 RISC 模块并测试外围硬件。此外，研华还提供针对 RISC COM 产品的标准软件板级支持包 （BCP），以便客户定义产品规格并同时验证 I/O 接口及性能。研华一向致力于为客户打造全方位服务，不仅包括硬件规划和技术咨询，而且还包括软件评估和外围模块推荐，如 WiFi、3G 和 BT。解决客户后顾之忧是研华在规划阶段的主要目标。因为我们深知，产品评估是规划阶段的关键任务，尤其是评估产品性能及规格，因此我们会尽其所能帮助客户完成 RISC COM 产品所需的全面测试。

设计阶段

当产品进入设计阶段，研华将为客户提供一份载板设计指南，以供用户后续参考。载板设计指南包括 COM 接口针脚定义、载板设计的局限及建议信息，以便客户在载板设计过程中有章可循。考虑到不同规格外形需求，研华针对 Q7、ULP 和 RTX2.0 等各种规格提供了完整的排针检查表，方便客户检查载板信号和布局设计。另外，研华设计团队还会协助客户复审位置 / 布局和电路图，保证载板设计完全满足客户需求。对于软件设计开发，研华 RISC 软件团队将协助客户建立软件开发环境，并评估所需时间及资源。如果客户将软件开发外包给第三方，研华还能够与第三方合作并提供专业的咨询服务。凭借研华的专业支持服务，设计过程得以简化，产品质量得到提升，从而帮助客户达到既定目标。

整合阶段

这一阶段包括硬件 / 软件集成、应用开发和外围模块安装。由于缺乏平台层整合技术及经验，客户往往需要花费大量时间分析整合问题。此外，由于外围模块安装与载板的驱动设计密切相关，RISC 平台却通常无法支持载板的既有驱动，因此客户必须从不断的尝试和失败中汲取经验，才能最终获得最佳解决方案。研华的整合团队在客户支持和软硬件开发领域拥有多年资深经验，有能力为客户提供专业建议和信息，帮助客户缩短开发时间，实现高效产品整合。

验证阶段

在客户的工程样品（ES）出炉之后，下一步就是进入验证阶段。除了要验证产品功能外，这一阶段的另一个重要组成部分就是要进行产品效率的相关测试，这一点对于 RISC 平台而言尤其重要。

研华在这一阶段中将扮演辅助角色，帮助客户解决测试过程中出现的问题，并有针对性地给出建议和提示。高效的验证流程依托研华技术支持团队这一坚强后盾，使得客户能够最终轻松优化应用。除此之外，研华团队还为客户的后续测试及设备使用提供专业咨询服务，帮助客户选择适当工具有效发现并解决问题，进而不断提升产品质量和性能。

5.2 联系信息

研华客户服务联系信息如下：

地区 / 国家	联系信息
美国	1-888-576-9688
巴西	0800-770-5355
墨西哥	PCI-1601/01/-800/-467/-2415
欧洲 （免费）	00800-2426-8080
新加坡 & SAP	65-64421000
马来西亚	1800-88-1809
澳大利亚 （免费）	1300-308-531
中国 （免费）	800-810-0345 800-810-8389 Sales@advantech.com.cn
印度 （免费）	1-800-425-5071
日本 （免费）	0800-500-1055
韩国 （免费）	080-363-9494 080-363-9495
台湾 （免费）	0800-777-111
俄罗斯 （免费）	8-800-555-01-50

此外，您还可以访问以下网站联系研华服务团队。我们的技术工程师将在您填写表格后快速反馈：

http://www.advantech.com.tw/contact/default.aspx?page=contact_form2&subject=Technical+Support

5.3 全球服务政策

5.3.1 质保政策

研华产品的质保政策如下：

5.3.1.1 质保期

对于研华原装成品产品、以及用于组装研华按单配置产品的第三方成品产品，研华提供两年完整和即时的全球质保服务。自产品发货之日起，产品的设计、材料和工艺缺陷即涵盖在质保范围之内。

所有定制产品默认享受 15 个月的区域质保服务。但产品的实际质保条款和条件可能会根据销售合同有所不同。

所有单独购买的第三方产品涵盖在原始制造商的质保和质保期范围之内，研华对此提供不超过一年的质保。

5.3.1.2 质保维修

如果产品是从研华直接购买，且出现 DOA （Dead-on-Arrival，到步死机）情形，请自购买之日起 30 天内联系原始供应商安排 DOA 换新，将以交换出货 （Cross-Shipment）的方式进行更换。DOA 交换出货不包括任何运输损坏以及定制和 / 或按单配置产品。

对于非 DOA 产品，返回至研华授权维修中心的相关费用将由客户自行承担。研华将负责将已修复产品返回给客户的运输费用。

5.3.1.3 免保范围

出现以下情形时，产品不包含在研华质保范围之内：

- 质保期满后发现产品有缺陷；
- 产品或其部分识别标签被移除或更改，质保资格被取消。
- 产品经过不当使用、滥用或未授权人员拆卸或更改；放置于不合适的物理或操作环境；经客户不当维护；故障原因不在研华负责范围内，不论是意外或其它原因。对于上述情形，研华有权自行裁定。
- 产品由于雷击、洪水、地震等自然灾害造成损坏而无法修复。
- 客户所提出的产品更新 / 升级和测试要求不在质保范围内。

5.3.2 维修流程

5.3.2.1 获取 RMA 序列号

客户的所有返修必须授权研华 RMA（Return Merchandise Authorization）序列号。研华不接受任何无有效 RMA 序列号的缺陷设备或部件返修；一旦发现将返回给客户并由客户自行承担费用，恕不另行通知。

RMA 序列号仅作为产品返修凭证，但并不视为同意维修或更换。申请 RMA 序列号时，请使用授权用户名和密码访问研华 RMA 网站：http://erma.ADVANTECH.com.tw。

您需要填写基本产品和客户信息，并在 “ 问题描述 ” 一栏中详细描述所遇到的问题。请尽量避免 “ 无法工作 ”、“ 故障 ” 等模糊性词汇。

如果无法确认故障原因，请与研华应用工程师 （AE）联系，这样也许无需返回产品即可找到解决方案。

仅需返回主要缺陷部件进行维修时，仍需提供整个设备的序列号，否则将被视为出保。

5.3.2.2 产品送修

客户可将缺陷产品送往任何研华授权维修中心维修以节约时间，而不用承担任何跨区域维修费用。请在获取全球维修服务之前，首先与当地维修中心取得联系。

建议仅返回板卡，不带附件 （手册、线缆等） 。请移除板卡上所有不必要的部件，如 CPU、DRAM 和 CF 卡等。如果您将所有这些部件返回 （您认为这些可能是故障原因所在），请备注清楚所包含的部件。否则，研华对不包含在列表中的所有项目不承担任何责任。请确认随附 “ 问题描述 ” 说明。

对于不在欧共体范围之内的欧洲客户，请使用 UPS 转运公司。我们强烈建议您在所有运输中添加包装清单。请根据以下指导准备货物装运单以缩短清关时间：

1. 请在运单上降低产品价值，否则海关征收的附加费用将由发货者承担。
2. 请在运单上添加此条信息 “ 运单仅用于清关，而不作其它商业用途 ”。
3. 请在运单上标注 RMA 序列号、产品序列号和质保状态。
4. 请在运单上添加货物原产国信息。

此外，请在包装箱外面附上 RMA 序列号单据和装箱单以节约处理时间。请将产品部件直接寄至研华服务部，并在包裹上注明 “RMA 服务部收 ”。

所有产品在返修时必须以防静电材料或袋子正确包装。如果包装不当，研华保留不维修即返回的权利，且所有费用由客户自行承担。

此外，建议在发货时选择快速专递等 “ 送货上门 ” 运输方式，否则，如果采用空运货物，发货者应承担清关费等附加费用。

对于 DOA 故障情形，研华将承担产品及运输的全部费用。对于非 DOA 但在研华质保范围之内的故障，运输费用由发货方负责。而对于出保产品，维修成本及来回运输费用将都由客户承担。

5.3.2.3 服务费用

出现以下情形时，产品不包含在研华质保范围之内：

- 产品维修超过质保期。
- 产品在质保期满后进行测试或校准，且检测结果为未发现问题 （NPF）。
- 产品仍在质保期内，但经过不当使用、滥用或未授权人员拆卸或更改；放置于不合适的物理或操作环境；经客户不当维护；故障原因不在研华负责范围内，不论是意外或其它原因。这种情形将由研华自行决定。
- 产品由于雷击、洪水、地震等自然灾害造成损坏而无法修复。
- 客户所提出的产品更新和测试要求不在质保范围内。

如果产品已经过研华维修，且在维修后三个月内因同一问题需要再次维修，研华将免费进行第二次维修。但是，此免费维修不适用于以下情形：产品经过不当使用、滥用或未授权人员拆卸或更改；放置于不合适的物理或操作环境；经客户不当维护；故障原因不在研华负责范围内，不论是意外或其它原因。

请联系最近的区域服务中心详细咨询服务报价。

在开始出保维修之前，研华将为客户出具一份含有维修费用的估价单 （P/I） 。付款后，请参考 “ 参考编号 ” （Our Ref）下面列出的 P/I 编号。如果未返回 DOA 设备或签署 P/I，研华有权拒绝提供维修服务。同时，如果客户在三个月内未返回 P/I 签字，研华将对缺陷产品进行报废处理，恕不另行通知。

5.3.2.4 维修报告

研华在返回产品时将出具一份 “ 维修报告 ”，说明维修结果。在客户提出维修要求时，研华也将提供一份 “ 维修分析报告 ”。如果缺陷并非由研华设计或制造原因产生，对于保内和出保维修分析报告，研华将向客户分别收费 US$60 和 US$120。

5.3.2.5 已提交维修产品保管

对于已提交维修但未返回 P/I 签字或未付款 （A/R）的产品，研华将保管一个月。如果客户在这段时间内仍未作出回应，研华将自动结束产品维修过程。在这一个月内，研华应利用有效的沟通手段与客户保持联系。

5.3.2.6 返回客户

研华有权自主选择将 RMA 产品返回至客户的转运公司。研华还可根据客户要求采用其它快递服务，如 UPS、FedEx 等，但由此产生的额外费用将由客户承担。如果有特殊要求，请在将产品寄送给我们时予以说明。

ADVANTECH

Enabling an Intelligent Planet

www.advantech.com.cn