



User Manual

RSB-4220

**3.5" SBC with TI Sitara AM3352
Cortex A8 Single-Core 1GHz
High-Performance Processor**

ADVANTECH

Enabling an Intelligent Planet

Copyright

The documentation and the software included with this product are copyrighted 2017 by Advantech Co., Ltd. All rights are reserved. Advantech Co., Ltd. reserves the right to improve the products described in this manual at any time without notice. No part of this manual may be reproduced, copied, translated, or transmitted in any form or by any means without the prior written permission of Advantech Co., Ltd. The information provided in this manual is intended to be accurate and reliable. However, Advantech Co., Ltd. assumes no responsibility for its use, nor for any infringements of the rights of third parties that may result from its use.

Acknowledgements

ARM is a trademark of ARM Corporation.

TI is a trademark of TI Corporation.

Microsoft Windows and MS-DOS are registered trademarks of Microsoft Corp.

All other product names or trademarks are properties of their respective owners.

Product Warranty (2 years)

Advantech warrants the original purchaser that all its products will be free from defects in materials and workmanship for two years from the date of purchase.

This warranty does not apply to products that have been repaired or altered by persons other than repair personnel authorized by Advantech, or products that have been subject to misuse, abuse, accident, or improper installation. Advantech assumes no liability under the terms of this warranty as a consequence of such events.

Because of Advantech's high quality-control standards and rigorous testing, most customers never need to use our repair service. However, if an Advantech product is defective, it will be repaired or replaced at no charge during the warranty period. For out-of-warranty repairs, customers are billed according to the cost of replacement materials, service time, and freight. Please consult your dealer for more details.

If you believe your product to be defective, follow the steps outlined below.

1. Collect all information about the problem encountered. (For example, CPU speed, Advantech products used, and other hardware or software used.) Note anything abnormal and list any onscreen messages encountered when the problem occurs.
2. Call your dealer and describe the problem. Please have your manual, product, and any relevant information readily available.
3. If your product is diagnosed as defective, obtain an RMA (return merchandise authorization) number from your dealer. This allows us to process your return more quickly.
4. Carefully pack the defective product, a completed Repair and Replacement Order Card, and proof of purchase date (such as a photocopy of your sales receipt) in a shippable container. Products returned without a proof of purchase date are not eligible for warranty service.
5. Write the RMA number clearly on the outside of the package, then ship the product prepaid to your dealer.

Declaration of Conformity

FCC Class A

This equipment has been tested and found to comply with the limits for a Class A digital device, pursuant to part 15 of the FCC regulations. These limits are designed to provide reasonable protection against harmful interference when the equipment is operated in a commercial environment. This equipment generates, uses, and can radiate radio frequency energy and, if not installed and used in accordance with the instruction manual, may cause harmful interference to radio communications. Operation of this equipment in a residential area is likely to cause harmful interference. In such cases, users will be required to correct the interference at their own expense.

Technical Support and Assistance

1. Visit the Advantech website at <http://support.advantech.com> to obtain the latest product information.
2. Contact your distributor, sales representative, or Advantech's customer service center for technical support if you need additional assistance. Please have the following information ready before calling:
 - Product name and serial number
 - Description of your peripheral attachments
 - Description of your software (operating system, version, application software, etc.)
 - Comprehensive description of the problem
 - The exact wording of any error messages

Packing List

Before installation, please ensure the following items have been shipped:

Item Part Number

- 1 x RSB-4220 SBC
- Connector

Model Number	Description
1652006830-01	Terminal block 20 x 2P 2.54 mm 180D 0156-1A40

Ordering Information

Model Number	Description
RSB-4220CS-MCA1E	RSB-4220 TI AM3352 1GHz, 512 MB DDR3
RSB-4220WS-MCA1E	RSB-4220 TI AM3352 1GHz, 512 MB DDR3 for wide temperature
RSB-DK4220-F0A1E	RSB-4220 TI AM3352 1GHz, 512MB DDR3 for EVK

Optional Accessories

Model Number	Description
96PSA-A36W12R1	ADP A/D 100-240V 36W 12 V
96LEDK-A070WV40NB1	7" LED panel 350N 800X480(G) G070VW01 V1
EWM-W150H01E	Advantech 802.11bgn/RT5390 1T1R /USB signal
1750006043	Cable R/P SMA (M) to MHF 1.32 150 mm
SQF-ISDS1-2G-86E	SQF SD C6 SLC 2G, 1CH (-40 ~ 85 °C)
170203183C	Power code 3P Europe (WS-010+WS-083)183 cm
1700021565-01	Debug cable
1700023366-01	Backlight cable
1700024543-01	LVDS cable
1700022248-02	M cable USB-A(M)/USB-A(M) 15 cm AMK-V006E
1700023307-01	A cable DC jack/plug-in 1*2P-5.0 10 cm RSB-4220

Certification and Safety Instructions

This device complies with the requirements outlined in part 15 of the FCC regulations. Operation is subject to the following two conditions:

1. This device may not cause harmful interference
2. This device must accept any interference received, including interference that may cause undesired operation

This equipment has been tested and found to comply with the limits for a Class A digital device, pursuant to part 15 of the FCC regulations. These limits are designed to provide reasonable protection against harmful interference when the equipment is operated in a commercial environment. This equipment generates, uses, and can radiate radio frequency energy and, if not installed and used in accordance with the instruction manual, may cause harmful interference to radio communications. Operation of this device in a residential area is likely to cause harmful interference. In such cases, users are required to correct the interference at their own expense. Users are advised that any equipment changes or modifications not expressly approved by the party responsible for compliance will void compliance with the FCC regulations and therefore, the user's authority to operate the equipment.

Caution! *New batteries are at risk of exploding if incorrectly installed. Do not attempt to recharge, force open, or heat the battery. Replace the battery only with the same or equivalent type recommended by the manufacturer.*



Discard used batteries according to the manufacturer's instructions.

Contents

Chapter 1	General Introduction	1
1.1	Introduction	2
1.2	Specifications	2
1.2.1	Functional Specifications	2
1.2.2	Mechanical Specifications	3
1.2.3	Electrical Specifications	3
1.3	Environmental Specifications	3
1.4	Block Diagram	3
Chapter 2	H/W Installation.....	5
2.1	Jumpers	6
2.1.1	Jumper Description	6
2.1.2	Jumper List	6
	Table 2.1: Jumper List	6
2.1.3	Jumper Settings	7
2.2	Connectors	9
2.2.1	Connector List	9
2.2.2	Connector Settings	9
	Figure 2.1 Mini PCIe	10
	Figure 2.2 Debug Port	11
	Figure 2.3 USB Type-A Connector	11
	Figure 2.4 Ethernet Connector	13
	Figure 2.5 DC Power Jack	14
	Figure 2.6 Reset Button	14
	Figure 2.7 SD Slot	15
	Figure 2.8 LVDS Connector	16
	Figure 2.9 LVDS Inverter Power Connector	17
	Figure 2.10 20-Pin Connector	18
2.3	Mechanical	19
2.3.1	Jumper and Connector Locations	19
	Figure 2.11 Jumper and Connector Layout (Top)	19
	Figure 2.12 Jumper and Connector Layout (Bottom)	19
	Figure 2.13 Coastline Layout	19
2.3.2	Board Dimensions	20
	Figure 2.14 Board Dimensions Layout (Top)	20
	Figure 2.15 Board Dimensions Layout (Bottom)	21
	Figure 2.16 Board Dimensions Layout (Coastline)	21
2.4	Quick Start	22
2.4.1	Debug Port Connection	22
2.4.2	Debug Port Setting	22
	Figure 2.17 HyperTerminal Settings for Terminal Setup	22
2.5	Test Tools	23
2.5.1	eMMC Test	23
2.5.2	USB Test	24
2.5.3	SD Test	24
2.5.4	SPI Test	25
2.5.5	I2C Test	26
2.5.6	CAN Test	26
2.5.7	GPIO Test	27
2.5.8	LVDS Test	27
2.5.9	LAN Test	28
2.5.10	RS232 Test	30
2.5.11	Watchdog Timer Test	32

Chapter 3 Software Functionality 33

3.1	Introduction	34
3.2	Package Contents.....	34
3.2.1	Pre-Built System Image	34
3.2.2	Source Code Package.....	34
	Figure 3.1 Source Code Package Structure.....	35
	Figure 3.2 Image\rootfs	36
3.3	Setting Up a Development Environment.....	38
3.3.1	Setenv.sh.....	38
3.4	Build Instructions.....	39
3.4.1	Build a U-Boot Image.....	39
3.4.2	Build a Linux Kernel Image.....	39
3.4.3	Build a Log.....	39
3.5	Create Linux System Boot Media	40
3.5.1	Storage Information (eMMC/SD Card)	40
3.5.2	Create a Linux System SD Card.....	40
3.5.3	Boot From Onboard Flash	41
3.6	Debug Message.....	41
	Figure 3.3 HyperTerminal Settings for Serial Console Setup....	41
3.7	Linux System Configuration and Use.....	42
3.7.1	Display Output Setting	42
3.7.2	Service Configuration	44
3.7.3	Network Configuration	46
3.7.4	Date/Time Configuration*	46
	Figure 3.4 Date/Time Settings.....	46
3.7.5	About the System	46
3.7.6	Brightness Control	47
	Figure 3.5 Brightness Control.....	47
3.7.7	Serial Tools.....	47
	Figure 3.6 Serial Control.....	47
3.7.8	Matrix GUI User Guide	48
	Figure 3.7 Matrix.....	48
3.7.9	Add Startup Items	49
3.7.10	OTG Mode Selection	50
3.8	Development Guide and Reference.....	50
3.8.1	Development of C/C++ Programs.....	50
3.8.2	Developing GUI Programs with a Qt Library.....	51
3.8.3	Demo Program Source Code	51

Chapter 4 System Recovery 55

4.1	System Recovery.....	56
-----	----------------------	----

Chapter 5 Advantech Services 57

5.1	RISC Design-in Services	58
5.2	Contact Information.....	60
5.3	Global Service Policy	61
5.3.1	Warranty Policy.....	61
5.3.2	Repair Process	62

Chapter 1

General Introduction

This chapter gives background information on the RSB-4220

- Introduction
- Specifications
- Environment Specifications
- Block Diagram

1.1 Introduction

RSB-4220 is a 3.5" single-board computer (SBC) equipped with a TI Sitara AM3352 Cortex A8 1GHz processor. RSB-4220 supports 512 MB of DDR3, 4 GB of eMMC onboard flash, LVDS, 5 serial ports, 1 USB 2.0 client, 2 GbE, 1 SD, and 1 Mini PCIe port. RSB-4220 is targeted to automation applications, providing a high-performance and low-power-consumption system based on a Cortex A8 architecture. The system is compact and can be easily expanded according to user requirements. Featuring flexible I/O interfaces and comprehensive hardware and software tools, RSB-4220 allows customers to develop applications and products without system integration concerns, resulting in a faster time-to-market.

Please refer to the RSB-4220 system specifications for information regarding the system interface icons.

1.2 Specifications

1.2.1 Functional Specifications

Processor

- TI Sitara AM3352 Cortex A8 single-core 1GHz processor
- Supports multiple I/O interfaces and HW WTD

System Memory Support

- DDR3 800 MHz
- Capacity: 512 MB of onboard DDR3

Gigabit Ethernet

- Realtek 8211 transceiver
- 2 x10/100/1000 Mbps

Peripheral Interface

- 1 x single-channel 18-bit LVDS
- 1 x USB 2.0 host/OTG (via jumper selection)
- 2 x Giga LAN, 1 x I2C, 1 x CAN, 8 x GPIO w/isolation
- 1 x SD slot
- 1 x RS-232/422/485, 4 x RS-232
- 1 x reset button
- 1 x mini PCIe slot (USB signal only)
- HW WDT by MSP430G2202

OS Support

RSB-4220 supports Linux Kernel 3.2.0

1.2.2 Mechanical Specifications

- **Dimensions:** 146 x 102 mm (5.7 x 4")
- **Height:** 15.92 mm
- **Reference Weight:** 640g (entire system)

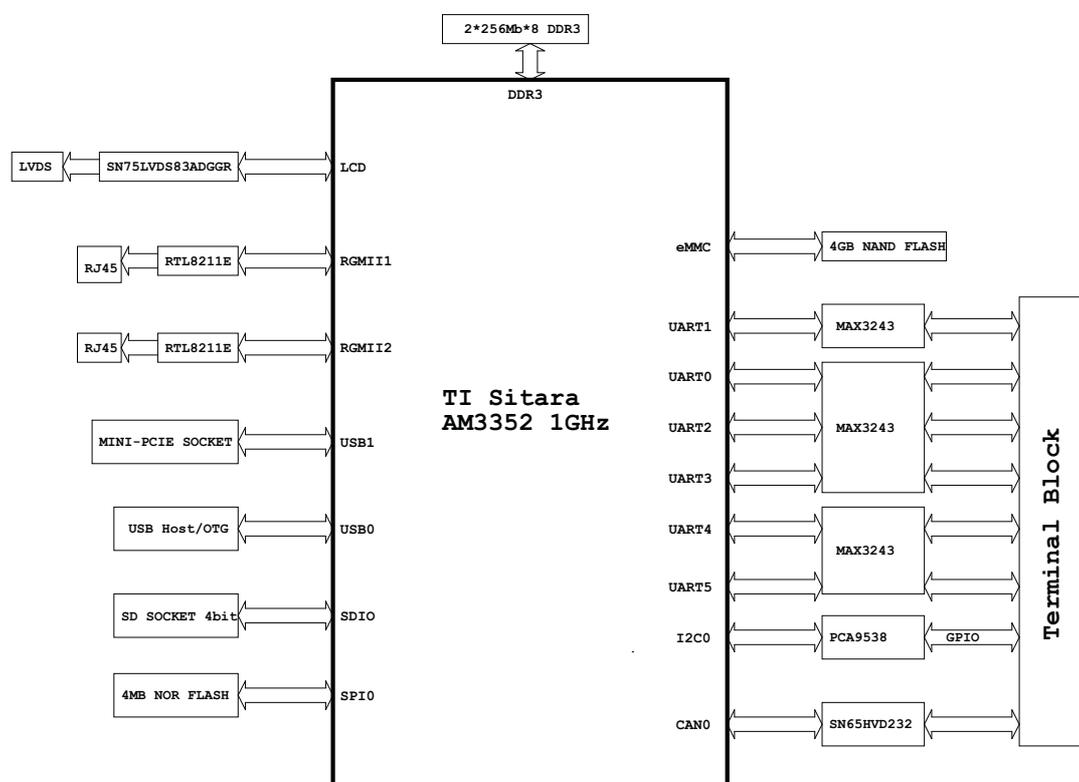
1.2.3 Electrical Specifications

- **Power Supply Type:** DC-in 12/19/24 V
- **Power Consumption:**
 - Kernel idle mode: 7.06 W (w/display)
 - Max mode: 7.7 W (w/display)
- **RTC Battery:**
 - Typical voltage: 3.3 V
 - Normal discharge capacity: 210 mAh

1.3 Environmental Specifications

- **Operating Temperature:** 0 ~ 60 °C (32 ~ 140°F) / -40 ~ 85 °C (-40 ~ 185°F)
- **Operating Humidity:** 40 °C @ 95% RH non-condensing
- **Storage Temperature:** -40 ~ 85 °C (-40 ~ 185 °F)
- **Storage Humidity:** 60 °C @ 95% RH non-condensing

1.4 Block Diagram



Chapter 2

H/W Installation

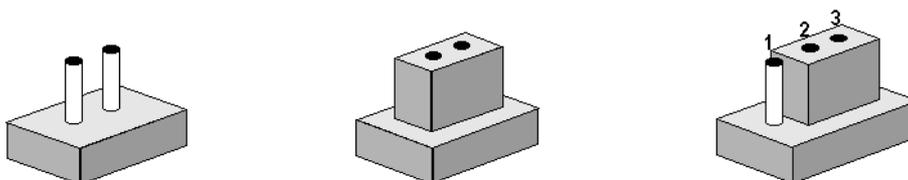
This chapter explains the system hardware startup procedures, including jumper, switch, and indicator setting, as well as device integration. Mechanical drawings are also provided.

Please read all safety precautions before beginning installation.

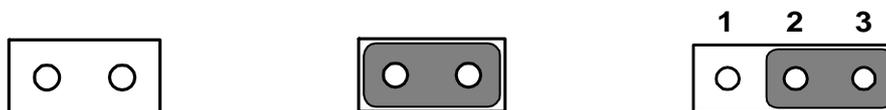
2.1 Jumpers

2.1.1 Jumper Description

Cards can be configured by setting jumpers. A jumper is a metal bridge used to close an electric circuit. It consists of two metal pins and a small metal clip (often protected by a plastic cover) that slides over the pins to connect them. To close a jumper, connect the pins with the clip. To open a jumper, remove the clip. Certain jumpers have three pins, which are labeled 1, 2, and 3. In such cases, connect either Pins 1 and 2, or Pins 2 and 3.



The jumper settings are schematically depicted below.



A pair of needle-nose pliers may be necessary when working with jumpers. For advice regarding the optimum hardware configuration for your application, contact your local distributor or sales representative before making any changes.

Generally, only a standard cable is required to make most connections.

Warning! To avoid damaging the computer, always turn the power supply off before setting jumpers.



2.1.2 Jumper List

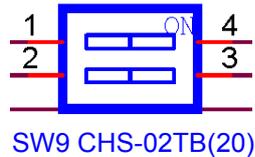
Table 2.1: Jumper List

J1	Boot device (SW9)
J2	LVDS power (CN29)
J3	Backlight power (CN30)
J4	USB host/OTG (CN25)
J5	UART1 RS232, RS422, RS485 select (SW8)

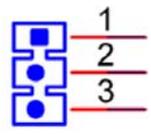
2.1.3 Jumper Settings

J1	Boot device (SW9)
Part number	1600000202
Footprint	SW_2x2P_50_161X315
Description	DIP SW CHS-02TB(29) SMD 4P SPST P = 1.27 mm W = 5.4 mm
Setting	Function
(OFF-OFF)	Boot from SD
(OFF-ON)	Boot from SPI

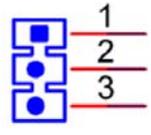
This switch is provided for users to select the boot up method.



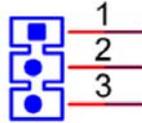
J2	LVDS power (CN29)
Part Number	1653003100
Footprint	HD_3x1P_100_D
Description	Pin header 3x1P 2.54 mm 180D(M) DIP 205-1x3GS
Setting	Function
(1-2)	+3.3 V
(2-3)	+V5



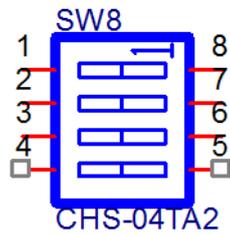
J3	LVDS backlight power (CN30)
Part Number	1653003100
Footprint	HD_3x1P_100_D
Description	Pin header 3x1P 2.54 mm 180D(M) DIP 205-1x3GS
Setting	Function
(1-2)	+V5
(2-3)	+V12



J4	USB host/OTG (CN25)
Part number	1653003100
Footprint	HD_3x1P_100_D
Description	Pin header 3x1P 2.54 mm 180D(M) DIP 205-1x3GS
Setting	Function
(1-2)	USB host
(2-3)	USB OTG device



J5	UART1 RS232, RS422, RS485 select (SW8)
Part number	1600000084
Footprint	SW_4x2P_50_260x220
Description	DIP SW CHS-02TB(29) SMD 4P SPST P = 1.27 mm W = 5.4 mm
Setting	Function
(ON-ON-ON-OFF)	RS232
(OFF-OFF-ON-OFF)	RS422
(OFF-ON-OFF-ON)	RS485



2.2 Connectors

2.2.1 Connector List

CN1	RTC battery
CN2	Mini PCIe
CN5	UART0 debug port
CN12	USB type-A connector
CN36	Ethernet connector
CN8	DC power jack
SW3	Reset button
SD1	SD card
CN32	LVDS CONN
CN31	LVDS backlight
CN28	2x20 pin terminal block

2.2.2 Connector Settings

2.2.2.1 RTC Battery Connector (CN1)

RSB-4220 supports a lithium 3V/210 mA_H CR2032 battery with wire via a battery connector.

2.2.2.2 Mini PCIe (CN2)

RSB-4220 supports a full-size Mini PCIe slot with USB interface. For half-size Wi-Fi cards, users must purchase an extension bracket (P/N: 1960047454N000) to install the Wi-Fi card.

Pin	Signal Name	Pin	Signal Name
1	NC	2	+3.3 V
3	NC	4	GND
5	NC	6	NC
7	NC	8	NC
9	GND	10	NC
11	NC	12	NC
13	NC	14	NC
15	GND	16	NC
Mechanical Key			
17	NC	18	GND
19	NC	20	NC
21	GND	22	PERST#
23	NC	24	+3.3 V
25	NC	26	GND
27	GND	28	NC
29	GND	30	SMB_CLK
31	NC	32	SMB_DATA
33	NC	34	GND
35	GND	36	USB_D-

37	GND	38	USB_D+
39	+3.3 V	40	GND
41	+3.3 V	42	NC
43	GND	44	NC
45	NC	46	NC
47	NC	48	NC
49	NC	50	GND
51	NC	52	+3.3 V

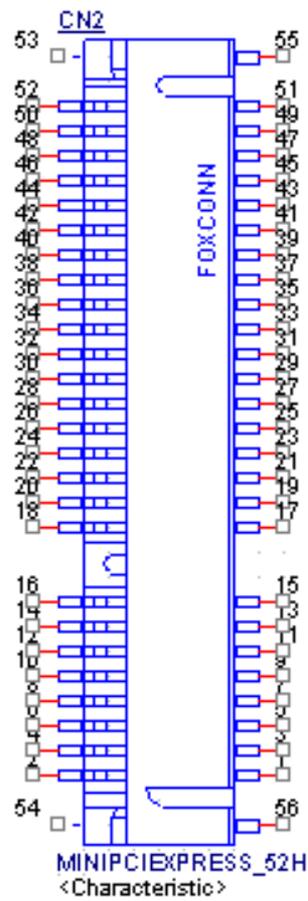


Figure 2.1 Mini PCIe

2.2.2.3 UART0 Debug Port (CN5)

RSB-4220 supports communication with a host server (Windows or Linux) via serial cables.

Pin	Description
1	NC
2	DEBUG_TXD
3	DEBUG_RXD
4	GND

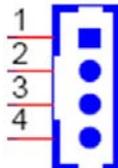


Figure 2.2 Debug Port

2.2.2.4 USB Type-A Connector (CN12)

RSB-4220 has one standard USB 2.0 type-A connector in the coastline. Users can connect either a USB host or OTG device via the jumper setting.

Pin	Description
1	+5 V
2	USB data-
3	USB data+
4	GND

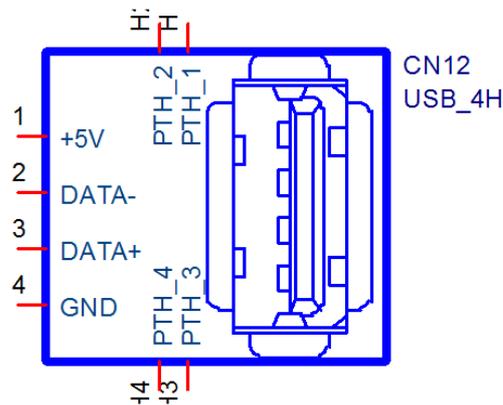


Figure 2.3 USB Type-A Connector

2.2.2.5 Ethernet Connector (CN36)

RSB-4220 features 2 RJ45 LAN interface connectors, which are fully compliant with the IEEE 802.3u 10/100/1000 Base-T CSMA/CD standards. A standard RJ-45 jack connector links the Ethernet ports to the LED indicators at the front of the device that indicate the system speed and Active/Link status.

Pin	Description
A1	MDI20+
A2	MDI20-
A3	MDI21+
A4	MDI21-
A5	GND
A6	GND
A7	MDI22+
A8	MDI22-
A9	MDI23+
A10	MDI23-
A11	LAN2_100_LINK
A12	LAN2_1000_LINK
A13	+3.3 V
A14	LAN2_ACT
B1	MDI10+
B2	MDI10-
B3	MDI11+
B4	MDI11-
B5	GND
B6	GND
B7	MDI12+
B8	MDI12-
B9	MDI13+
B10	MDI13-
B11	LAN1_100_LINK
B12	LAN1_1000_LINK
B13	+3.3 V
B14	LAN1_ACT

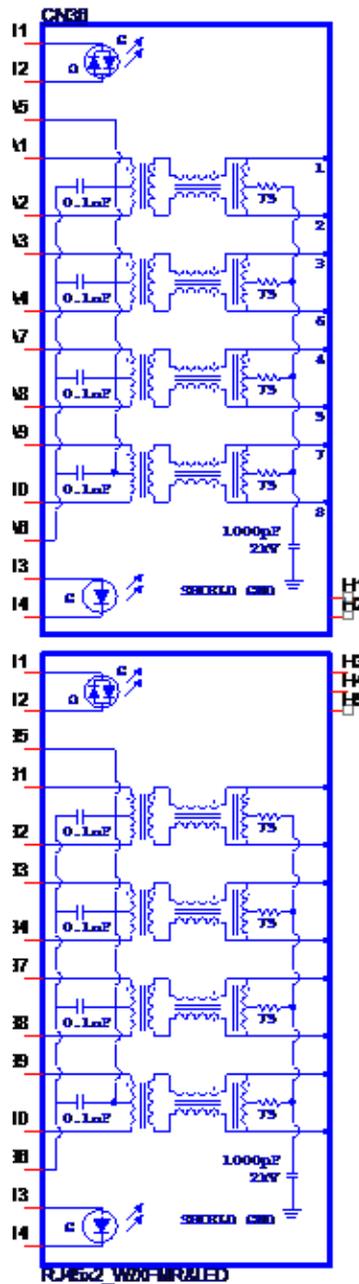


Figure 2.4 Ethernet Connector

2.2.2.6 DC Power Jack (CN8)

RSB-4220 is equipped with a DC-jack header that carries 9-30V DC external power input.

Pin	Description
1	DC_IN
2	GND

PLUG_2_5.00mm

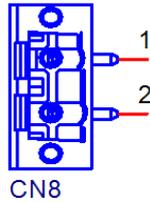


Figure 2.5 DC Power Jack

2.2.2.7 Reset Button (SW3)

RSB-4220 features a reset button located at the front of the device. Press this button to initiate the hardware reset function.

Pin	Description
1	RESET
2	GND
3	GND
4	GND

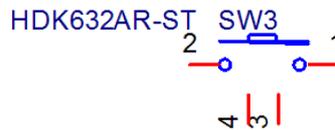


Figure 2.6 Reset Button

2.2.2.8 SD Slot (SD1)

RSB-4220 based on TI AM335X datasheet complies with SD&SDIO specifications 2.0. Advantech used 4G SD for testing (SDHC).

Pin	Signal Name
1	DAT3
2	CMD
3	GND
4	+3.3 V
5	CLK
6	GND
7	DAT0
8	DAT1
9	DAT2
WP	WP
CD	CD
SC	SC
G	G

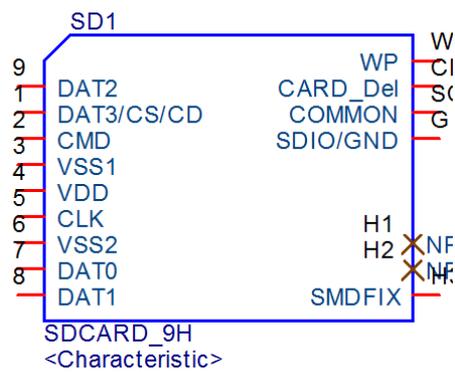


Figure 2.7 SD Slot

2.2.2.9 LVDS Connector (CN32)

RSB-4220 features an LVDS 10x2-pin board-to-board connector for a single-channel 18-bit LVDS panel (up to 1366 x 768). Please refer to the jumper settings provided on Page 16 before connecting the LVDS panel.

Pin	Description
1	GND
2	GND
3	LVDS0_z_D0+
4	SCL_LVDS0
5	LVDS0_z_D0-
6	SDA_LVDS0
7	LVDS0_z_D1+
8	NC
9	LVDS0_z_D1-
10	NC
11	LVDS0_z_D2+
12	NC
13	LVDS0_z_D2-
14	NC
15	LVDS0_z_CLK+
16	NC
17	LVDS0_z_CLK-
18	NC
19	+VDD_LVDS
20	+VDD_LVDS

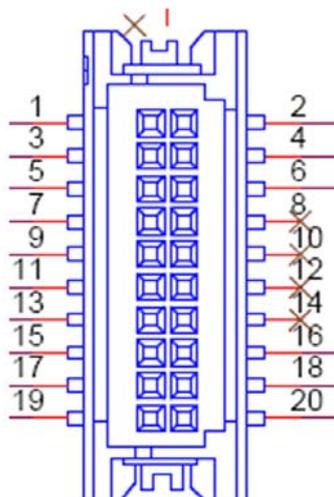


Figure 2.8 LVDS Connector

2.2.2.10 LVDS Inverter Power Connector (CN31)

Please refer to the jumper settings provided on Page 16 before connecting the LVDS panel.

Pin	Description
1	+VDD_BKLT_LVDS
2	GND
3	LCD_BKLT_A
4	LCD_BKLT_PWM_A
5	+V5

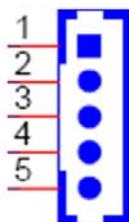


Figure 2.9 LVDS Inverter Power Connector

2.2.2.11 2x20-Pin Connector (CN28)

RSB-4220 features a 2x20-pin connector, which comprises five 2-wire UARTs with TX/RX, a 4-wire UART that supports RS232/RS422/RS485, one 5V CAN, one I2C bus, and 4 GPI/GPO w/isolation.

Pin	Description
1	IDI0
2	IDO0
3	IDI1
4	IDO1
5	IDI2
6	IDO2
7	IDI3
8	IDO3
9	GND_iso
10	PCOM
11	NC
12	GND_iso
13	NC
14	NC
15	422_RXD-
16	NC
17	422_RXD+
18	I2C0_SCL
19	COM1_CTS
20	I2C0_SDA
21	COM1_TXD
22	GND
23	COM1_RTS

24	COM5_TX
25	COM1_RXD
26	COM5_RX
27	422-485_TXD+
28	CAN1_D+
29	422-485_TXD-
30	CAN1_D-
31	COM2_RX
32	COM0_TX
33	COM2_TX
34	COM0_RX
35	COM4_TX
36	COM3_TX
37	COM4_RX
38	COM3_RX
39	GND
40	GND

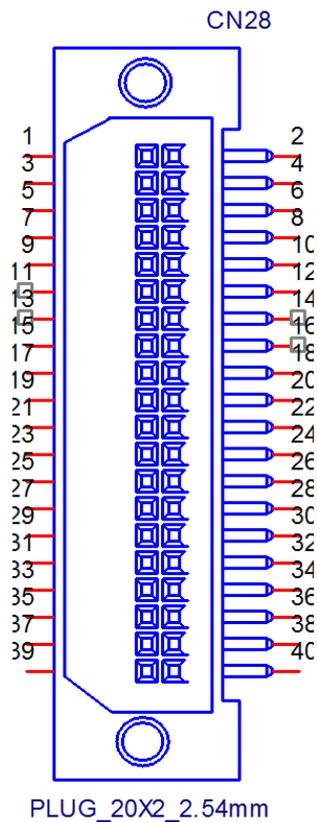


Figure 2.10 2x20-Pin Connector

2.3 Mechanical

2.3.1 Jumper and Connector Locations

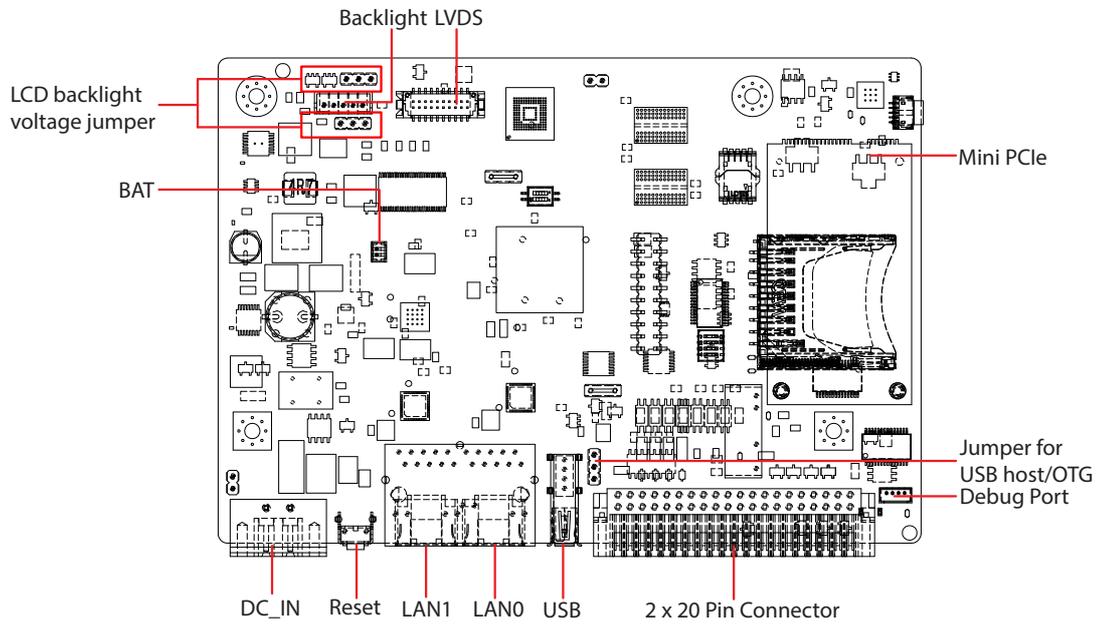


Figure 2.11 Jumper and Connector Layout (Top)

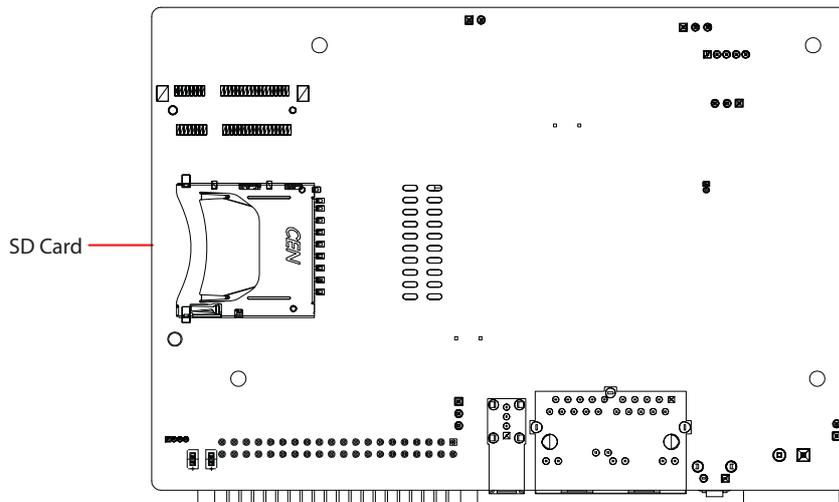


Figure 2.12 Jumper and Connector Layout (Bottom)

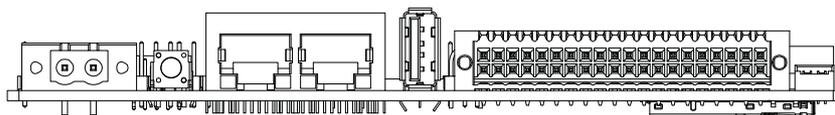


Figure 2.13 Coastline Layout

2.3.2 Board Dimensions

2.3.2.1 Board Drawings

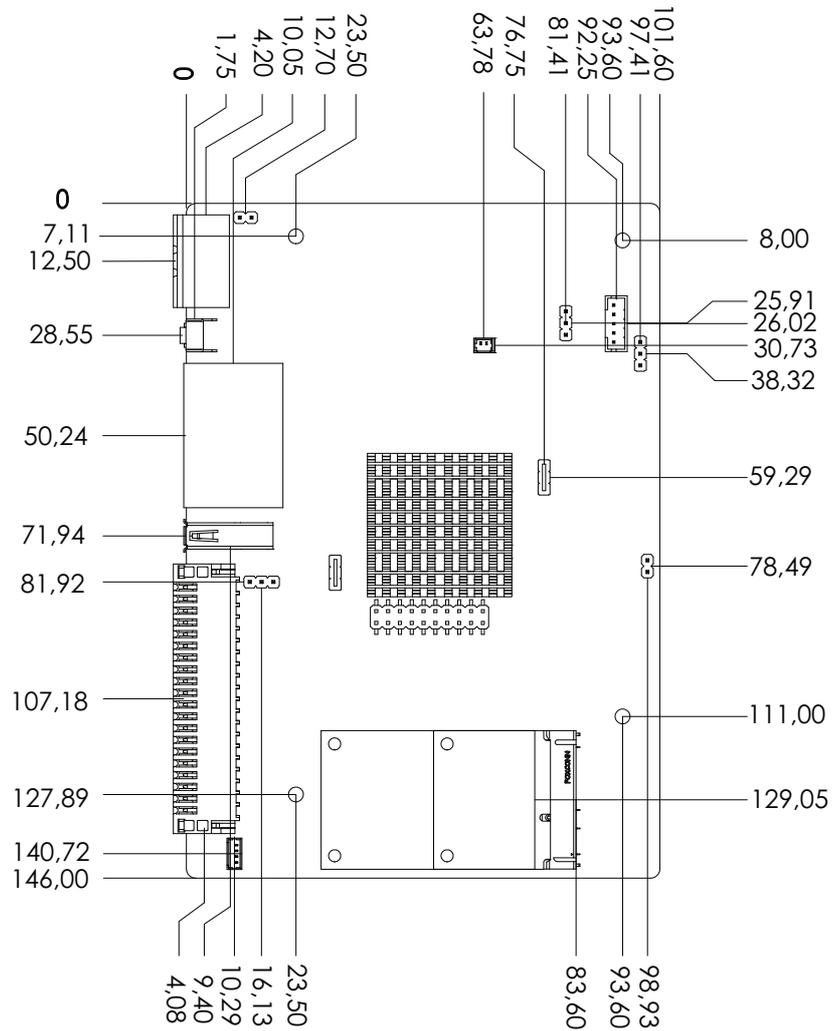


Figure 2.14 Board Dimensions Layout (Top)

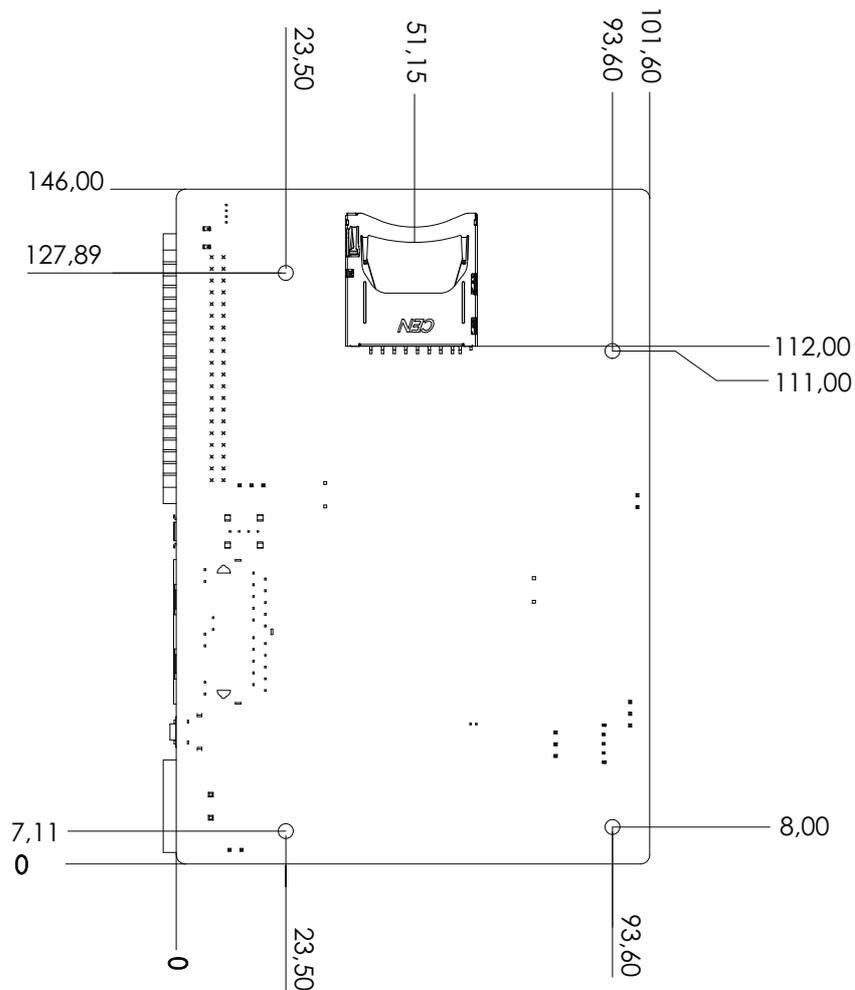


Figure 2.15 Board Dimensions Layout (Bottom)

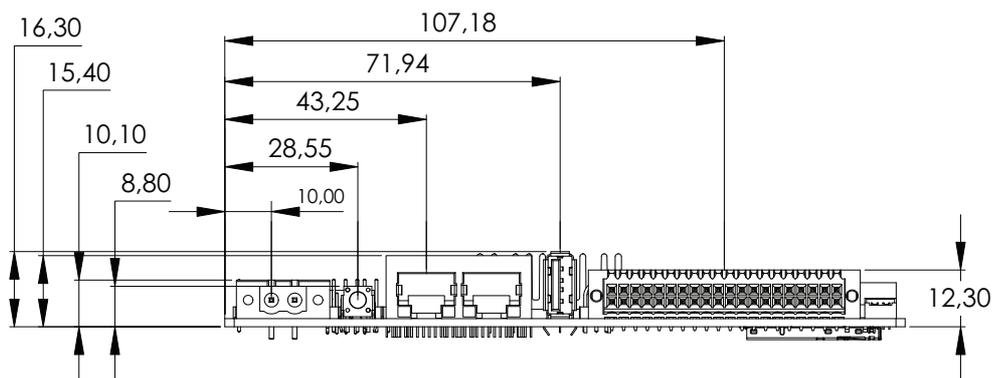


Figure 2.16 Board Dimensions Layout (Coastline)

2.4 Quick Start

2.4.1 Debug Port Connection

1. Connect debug port cable to the RSB-4220's debug port.
2. Connect the other end of the debug cable to a USB-to-RS-232 cable connected to your PC.

2.4.2 Debug Port Setting

RSB-4220 supports communication with a host server (Windows or Linux) via serial cables. Standard serial communication programs, such as HyperTerminal, Tera Term or PuTTY, can be used for this purpose. The example below demonstrates the serial terminal setup process, using HyperTerminal on a Windows host.

1. Connect RSB-4220 to your Windows PC via a serial cable.
2. Open HyperTerminal on your Windows PC, and select the settings shown in Figure 2-7.
3. After the bootloader is programmed on the SD card, insert the power adapter connector into the DC jack on RSB-4220 to power up the board. The terminal screen will display the bootloader prompt.

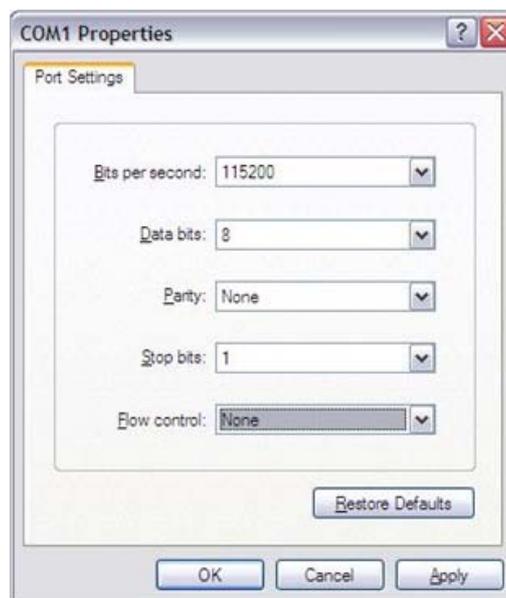


Figure 2.17 HyperTerminal Settings for Terminal Setup

2.5 Test Tools

All test tools must be verified on RSB-4220. Prepare all required test fixtures before verifying every specified I/O. Should problems obtaining test fixtures be encountered, please contact Advantech for assistance.

2.5.1 eMMC Test

1. Check the NAND flash memory space.

```
root@am335x-adv:~# fdisk -l /dev/mmcblk1
```

```
Units: sectors of 1 * 512 = 512 bytes
```

```
Sector size (logical/physical): 512 bytes / 512 bytes
```

```
I/O size (minimum/optimal): 512 bytes / 512 bytes
```

```
Disklabel type: dos
```

```
Disk identifier: 0xf3ac73d4
```

```
Device      Boot Start    End Sectors  Size Id Type
/dev/mmcblk1p1 *    2048 133119 131072   64M c W95 FAT32 (LBA)
/dev/mmcblk1p2    133120 7634943 7501824  3.6G 83 Linux
```

2. Run program to read/write to NAND flash memory. Please note that the execution of the below test script will re-partition and format mmc, so be cautious.

```
root@am335x-adv:/unit_tests# ./AutoRun_eMMC.sh mmcblk1
```

```
=====
Test Read/write and operation of filesystem for eMMC=====
=====
```

```
10240+0 records in
```

```
10240+0 records out
```

```
10240+0 records in
```

```
10240+0 records out
```

```
10240+0 records in
```

```
10240+0 records out
```

```
=====MMC Read/Write test pass=====
```

```
[ 309.769470] mmcblk1: p1
```

```
=====MMC fdisk test pass=====
```

```
mke2fs 1.42.9 (28-Dec-2013)
```

```
5+0 records in
```

```
5+0 records out
```

```
=====MMC FS test passes=====
```

```
mke2fs 1.42.9 (28-Dec-2013)
```

```
=====MMC MKFS test pass=====
```

```
=====all MMC function test PASS=====
```

2.5.2 USB Test

2.5.2.1 USB HOST TEST

1. Plug a USB flash device into the USB connector.
2. Mount the USB flash and check that it is detectable by the system.
3. Run program to read/write to USB flash memory.

```
root@am335x-adv:/unit_tests# ./AutoRun_usb_host.sh
```

```
disk_to_test=sda
---create partition for sda
---format file system
partion_list=sda1
test /dev/sda1
pass
```

2.5.2.2 USB OTG Test

1. Jump to OTG mode. Use a USB type-A cable to connect the USB-OTG port of RSB-4220 to the USB port of your PC.
2. Copy the 20 MB xx file to RSB-4220 from your PC.
3. Check that the RSB-4220 xx file size is 20 MB.

2.5.3 SD Test

1. Insert an SD card into the SD1 slot.
2. Mount the SD card device and check that it is detectable by the system.

```
root@am335x-adv:/unit_tests# fdisk -l /dev/mmcblk0
```

```
Disk /dev/mmcblk0: 3.7 GiB, 3980394496 bytes, 7774208 sectors
Units: sectors of 1 * 512 = 512 bytes
Sector size (logical/physical): 512 bytes / 512 bytes
I/O size (minimum/optimal): 512 bytes / 512 bytes
Disklabel type: dos
Disk identifier: 0x00000000
```

```
Device      Boot Start   End Sectors  Size Id Type
/dev/mmcblk0p1 *    63 144584 144522 70.6M c W95 FAT32 (LBA)
/dev/mmcblk0p2   160650 7759394 7598745 3.6G 83 Linux
```

3. Run program to read/write to SD. Please note that the execution of the below test script will re-partition and format SD card, so be cautious.

```
root@am335x-adv:/unit_tests# ./AutoRun_sd.sh mmcblk0
```

```
10240+0 records in
10240+0 records out
10240+0 records in
10240+0 records out
10240+0 records in
10240+0 records out
=====SD test pass=====
```

2.5.4 SPI Test

1. Power turns on and boots into OS.
2. Run program to test SPI flash read/write function.

```
root@am335x-adv:/unit_tests# ./AutoRun_spi.sh
```

```
#####---SPI Test for AM335X---#####
```

```
the spi falsh info:
```

```
mtd.type = MTD_NORFLASH
mtd.flags = MTD_CAP_NORFLASH
mtd.size = 4194304 (4M)
mtd.erasesize = 4096 (4K)
mtd.writesize = 1
mtd.oobsize = 0
regions = 0
```

```
Erased 4096 bytes from address 0x00030000 in flash
Copied 4096 bytes from address 0x00030000 in flash to temp.img
0000000 ffff ffff ffff ffff ffff ffff ffff
*
0001000
```

```
=====
#####          SPI mtdblock0  Read 1 PASS !!!
=====
```

```
1+0 records in
1+0 records out
0000000 0000 0000 0000 0000 0000 0000 0000 0000
*
0001000
```

```
Copied 4096 bytes from zero.img to address 0x00030000 in flash
Copied 4096 bytes from address 0x00030000 in flash to temp.img
0000000 0000 0000 0000 0000 0000 0000 0000 0000
*
0001000
```

```
=====
##### -----SPI mtdblock0 Write 0 PASS !!!
=====
```

```
=====
#####-----> SPI Test all mtdblock0 PASS !!!
=====
```

```
=====
##### Finish SPI all blocks  Test PASS !!!
=====
```

2.5.5 I²C Test

1. Power on RSB-4220 and boot into OS.
2. The system should detect all devices under I²C interface control.

```
root@am335x-adv:/unit_tests# ./AutoRun_i2c.sh
   0  1  2  3  4  5  6  7  8  9  a  b  c  d  e  f
00:          -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
10: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
20: -- -- -- -- -- -- -- -- -- -- UU -- -- -- UU -- --
30: UU UU UU UU UU UU UU UU -- -- -- -- -- -- -- --
40: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
50: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
60: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
70: -- -- -- UU -- -- -- -- --
====I2C test Pass!====
```

2.5.6 CAN Test

1. Connect one RSB-4220 CAN port CAN1_D+ /CAN1_D- and GND to another RSB-4220 CAN port.
2. Run program to transmit data between the two RSB-4220 CAN ports.

```
root@am335x-adv:/unit_tests# ./AutoRun_CAN.sh
Sun Sep 14 01:48:57 UTC 2014
interface = can0, family = 29, type = 3, proto = 1
wait for data
interface = can0, family = 29, type = 3, proto = 1
====CAN Pass====
Sun Sep 14 01:49:01 UTC 2014
interface = can0, family = 29, type = 3, proto = 1
wait for data
interface = can0, family = 29, type = 3, proto = 1
====CAN Pass====
```

2.5.7 GPIO Test

1. Power on RSB-4220 and boot into OS.
2. Short GPIO to GPO0, GPI1 to GPO1, GPI2 to GPO2, and GPI3 to GPO3.
3. Run program to test GPIO read/write function.

```
root@am335x-adv:/unit_tests# ./AutoRun_gpio.sh RSB-4220
GPIO200 direction is:
in
GPIO201 direction is:
in
GPIO202 direction is:
in
GPIO203 direction is:
in
GPIO204 direction is:
out
GPIO205 direction is:
out
GPIO206 direction is:
out
GPIO207 direction is:
out
GPIO test PASS!
```

2.5.8 LVDS Test

Step1: Stop matrix

```
root@am335x-adv:/unit_tests# /etc/init.d/matrix-gui-2.0 stop
```

Step2: Run program to test lvds.

```
root@am335x-adv:/unit_tests# ./AutoRun_RGB
```

2.5.9 LAN Test

RSB-4220 sets DHCP as the default network protocol.

2.5.9.1 Eth0 Test

1. Connect the RSB-4220 eth0 port to a host computer.
2. Configure the RSB-4220 eth0 port IP address as 192.168.1.2, and configure the host computer IP address as 192.168.1.1.

```
root@am335x-adv:~# ifconfig eth0 192.168.1.2
```

```
root@am335x-adv:~# ifconfig eth0
```

```
eth0    Link encap: Ethernet  HWaddr 78:A5:04:DD:E1:0A
```

```
        inet addr:192.168.1.2  Bcast:192.168.1.255  Mask:255.255.255.0
```

```
        UP BROADCAST RUNNING ALLMULTI MULTICAST MTU:1500 Metric:1
```

```
        RX packets:160 errors:0 dropped:0 overruns:0 frame:0
```

```
        TX packets:0 errors:0 dropped:0 overruns:0 carrier:0
```

```
        collisions:0 txqueuelen:1000
```

```
        RX bytes:23334 (22.7 KiB)  TX bytes:0 (0.0 B)
```

3. Use the commands below to check for responses from the host computer.

```
root@am335x-adv:~# ping 192.168.1.1
```

```
PING 192.168.1.1 (192.168.1.1): 56 data bytes
```

```
64 bytes from 192.168.1.1: seq=0 ttl=64 time=0.384 ms
```

```
64 bytes from 192.168.1.1: seq=1 ttl=64 time=0.159 ms
```

```
64 bytes from 192.168.1.1: seq=2 ttl=64 time=0.110 ms
```

```
64 bytes from 192.168.1.1: seq=3 ttl=64 time=0.102 ms
```

```
64 bytes from 192.168.1.1: seq=4 ttl=64 time=0.208 ms
```

```
64 bytes from 192.168.1.1: seq=5 ttl=64 time=0.135 ms
```

```
64 bytes from 192.168.1.1: seq=6 ttl=64 time=0.186 ms
```

```
64 bytes from 192.168.1.1: seq=7 ttl=64 time=0.151 ms
```

```
64 bytes from 192.168.1.1: seq=8 ttl=64 time=0.091 ms
```

```
64 bytes from 192.168.1.1: seq=9 ttl=64 time=0.203 ms
```

```
64 bytes from 192.168.1.1: seq=10 ttl=64 time=0.111 ms
```

```
64 bytes from 192.168.1.1: seq=11 ttl=64 time=0.105 ms
```

```
64 bytes from 192.168.1.1: seq=12 ttl=64 time=0.098 ms
```

```
64 bytes from 192.168.1.1: seq=13 ttl=64 time=0.091 ms
```

```
64 bytes from 192.168.1.1: seq=14 ttl=64 time=0.187 ms
```

```
64 bytes from 192.168.1.1: seq=15 ttl=64 time=0.123 ms
```

2.5.9.2 Eth1 Test

1. Connect the RSB-4220 eth1 port to a host computer.
2. Configure the RSB-4220 eth1 port IP address as 192.168.1.3.

```
root@am335x-adv:~# ifconfig eth1 192.168.1.3
```

```
root@am335x-adv:~# ifconfig eth1
```

```
eth1    Link encap: Ethernet HWaddr 78:A5:04:DD:E1:0C
        inet addr:192.168.1.3 Bcast:192.168.1.255 Mask:255.255.255.0
        UP BROADCAST RUNNING MULTICAST MTU:1500 Metric:1
        RX packets:41 errors:0 dropped:0 overruns:0 frame:0
        TX packets:0 errors:0 dropped:0 overruns:0 carrier:0
        collisions:0 txqueuelen:1000
        RX bytes:5035 (4.9 KiB) TX bytes:0 (0.0 B)
```

3. Use the commands below to check for responses from the host computer.

```
root@am335x-adv:~# ping 192.168.1.1
```

```
PING 192.168.1.1 (192.168.1.1): 56 data bytes
```

```
64 bytes from 192.168.1.1: seq=0 ttl=64 time=0.373 ms
```

```
64 bytes from 192.168.1.1: seq=1 ttl=64 time=0.208 ms
```

```
64 bytes from 192.168.1.1: seq=2 ttl=64 time=0.234 ms
```

```
64 bytes from 192.168.1.1: seq=3 ttl=64 time=0.115 ms
```

```
64 bytes from 192.168.1.1: seq=4 ttl=64 time=0.122 ms
```

```
64 bytes from 192.168.1.1: seq=5 ttl=64 time=0.107 ms
```

2.5.10 RS232 Test

RSB-4220 supports 6 UARTs. Of these, /dev/tty0 is reserved for the debug port (RSB-4220 CN5), while the remaining UART ports can be assigned by the user.

2.5.10.1 UART1 to UART5 RS232 Test

Switch SW8 to set UART1 to work with RS232, and the short TX with RX of UART1 to UART5.

```
root@am335x-adv:/unit_tests# ./AutoRun_uart232
```

```
=====test rs232!=====
```

```
rs232 number: 5
```

```
/dev/ttyO1 PASS!
```

```
/dev/ttyO2 PASS!
```

```
/dev/ttyO3 PASS!
```

```
/dev/ttyO4 PASS!
```

```
/dev/ttyO5 PASS!
```

```
+-----+
```

```
| [RS232] Test Pass! |
```

2.5.10.2 UART1 RS422 Test

1. Switch SW8 to set UART1 to work with RS422, short 422_RXD- with 422-485_TXD-, and short 422_RXD+ with 422-485_TXD+.
2. Run program to test UART1 RS422 function.

```
root@am335x-adv:/unit_tests# ./AutoRun_uart422 -p /dev/ttyO1 -t 1
```

```
=====test RS422 for RSB4220=====
```

```
Open uart /dev/ttyO1 PASS ....
```

```
->Writing: hello world!
```

```
->Reading: hello world!
```

```
->TX/RX Signal pass
```

```
+-----+
```

```
| UART RS422 Testing PASS |
```

```
+-----+
```

2.5.10.3 UART1 RS485 Test

Switch SW8 to set UART1 to work with RS485. Connect 2x20-pin's Pin 27 and Pin 29 to the RS485 port of ADAM, then connect the UART4 port to the RS232 port of ADAM. Run program to transmit data between UART1 and UART4.

RSB-4220's RS485 does not support auto flow control; this must be controlled by a user app.

```
root@am335x-adv:/unit_tests# ./AutoRun_uart485 -p /dev/tty01 /dev/
tty04 -t 6
```

```
Open uart /dev/ttyO1 OK ....
```

```
Open uart /dev/ttyO4 OK ....
```

```
Writing: hello world!
```

```
Reading: hello world!
```

```
->TX/RX Signal pass
```

```
+-----+
| UART RS485 Testing PASS |
+-----+
```

```
Close uart /dev/ttyO1 OK ....
```

```
Close uart /dev/ttyO4 OK ....
```

2.5.11 Watchdog Timer Test

RSB-4220 features an external watchdog IC that uses TI msp430g2202 to reset the system when exceptions occur.

The valid watchdog timeout value is between 1 and 5000 seconds, and the default timeout value is 60 seconds.

Run the AutoRun_WTD program to test this function.

```
#!/AutoRun_WTD n
```

Here n denotes the feed time, meaning the test program will feed the watchdog every n seconds.

In the AutoRun_WTD test program, first obtain the current timeout value, then set the timeout value to 10 seconds.

The program will feed the watchdog every n seconds. Here, n is determined by the parameters. If the feed time is longer than 10 seconds, the board will reboot after 10 seconds when the AutoRun_WTD program is initiated. If the watchdog time is less than 10 seconds, the board will not reboot because the program will feed the watchdog every 10 seconds.

To test the watchdog, run the command below. The board will reboot after 10 seconds.

```
root@am335x-adv:/unit_tests# ./AutoRun_WTD 15
```

Obtain the timeout value from the driver: timeout = 60 seconds

Set the timeout value to 10 seconds.

Obtain the timeout value from the driver again: timeout = 10 seconds

The parameter setting is successful and the watchdog is enabled.

Feed the watchdog every 15 seconds.

Feed the watchdog!

After rebooting the system, users will encounter the following boot messages:

```
U-Boot 2015.07-svn295 (Aug 18 2016 - 14:29:39 +0800)
```

```
Watchdog enabled
```

```
I2C: ready
```

```
DRAM: 512 MiB
```

```
MMC: OMAP SD/MMC: 0, OMAP SD/MMC: 1
```

```
SF: Detected W25Q32BV with page size 256 Bytes, erase size 4 KiB, total 4 MiB
```

```
Net: cpsw, usb_ether
```

```
Hit any key to stop autoboot: 0
```

```
switch to partitions #0, OK
```

Chapter 3

Software Functionality

This chapter details the Linux operating system installed on the RSB-4220 platform.

3.1 Introduction

The RSB-4420 platform is embedded with a Linux kernel, version 4.1.6, which contains all system-required shell commands and drivers. However, the RSB-4220 board support package (BSP) does not support an integrated development environment (IDE). Users can develop in an Ubuntu 14.04LTS environment.

Linux features the three major boot components “u-boot.img”, “zImage”, and “File System”. U-boot.img is for initializing peripheral hardware parameters, zImage is the Linux kernel image, and File System pertains to the Linux O.S.

If one of these three files is missing from the booting media (SD card or onboard flash), the platform will not be able to boot into a Linux environment.

The purpose of this chapter is to explain the software configuration and development to enable users to develop unique application(s) efficiently.

RSB-4220 application development can only be conducted on a Linux host PC. Application development is not possible on a Windows/Android host PC. The host PC development environment currently supported is Ubuntu 14.04 LTS; host PCs running other Linux versions may result in compatibility issues. In such cases, we strongly recommend installing Ubuntu 14.04 LTS on your host PC before beginning RSB-4220 evaluation/development.

3.2 Package Contents

Two Linux packages are offered for RSB-4220. One is a pre-built system image for system recovery, and the other is a source code package (BSP).

3.2.1 Pre-Built System Image

The pre-built system image RSB-4220LIVxxxx_yyyy-mm-dd.tar.gz can be obtained from the RSB-4220 evaluation kit DVD image provided on the Advantech website.

RSB-4220 supports booting from SD card. Therefore, users can extract the image to an SD card, then copy the image file to the onboard eMMC to perform a system recovery. For more details, please refer to Section 3.6 “Create Linux System Boot Media”.

3.2.2 Source Code Package

The RSB-4220 source code package (BSP) contains a cross compiler, Linux source code, Uboot source code, root file system, and several scripts for OS development. Some of these components were developed by Advantech, and some by the open source community. The RSB-4220 source code package contains six main folders: “cross_compiler”, “document”, “image”, “package”, “scripts”, and “source”.

Note! *The RSB-4220 source code package (BSP) is the intellectual property of Advantech. Should you need to access this package, please contact your local Advantech support provider.*





Figure 3.1 Source Code Package Structure

The AM335XLBVxxxx package contents are as follows:

- **“cross_compiler”** → This folder contains source code for the cross compiler.
- **“document”** → This folder contains the user guide.
- **“image”** → This folder contains the zImage and u-boot.img.
- **“image/rootfs”** → This folder contains the Linux root file system.
- **“package”** → This folder contains source code provided by TI without any modification.
- **“scripts”** → This folder contains scripts for the system configuration and automatic image compilation.
- **“source”** → This folder contains source code owned by Advantech.

3.2.2.1 Cross_compiler

The cross compiler tool chain can be used to compile the zImage and related applications (gcc version 4.9.3 20150413).

3.2.2.2 Document

This folder contains the user guide, which describes how to setup a development environment.

3.2.2.3 Image

This folder contains the zImage and u-boot.img.

3.2.2.4 Image/rootfs

Linux adopts a hierarchical file system (HFS). Image/rootfs is the Linux file system at the highest level of the file structure, similar to the trunk of a tree. The subdirectories are branches, and the files inside these subdirectories are the leaves. Image/rootfs contains all the subdirectories and files used by the file system, which is why it is known as the root of the entire file system.

The main folders in “rootfs” are listed below.

- bin → Common programs shared by the system, system administrator, and users.
- dev → Contains references to all CPU peripheral hardware, which are represented as files with special properties.
- etc → Contains the most important system configuration files, similar to that in the Windows Control Panel.
- home → Home directories of the frequent users.
- lib → Contains library files, including those for all types of programs required by the system and users.
- mnt → Standard mount point for external file systems.
- opt → Typically contains extra and third-party software.

- proc → A virtual file system containing information about the system resources. Additional information regarding the meaning of the files in proc can be obtained by entering the command `man proc` in a terminal window. The file `proc.txt` describes the virtual file system in detail.
- root → The administrative user's home directory. Notice the difference between `/` (the root directory) and `/root` (the home directory of the root user).
- sbin → Programs used by the system and system administrator.
- sys → Linux sys file system.
- tmp → Temporary space used by the system and cleaned upon reboot. Do not save any work to this space!
- usr → Programs, libraries, documentation, etc. for all user-related programs.
- var → Storage for all variable and temporary files created by users, such as log files, the mail queue, and the print spooler area. Also provides space for the temporary storage of files downloaded from the Internet.
- tools → Provided for sample testing.



Figure 3.2 Image\rootfs

3.2.2.5 Scripts

Several scripts provided by Advantech can facilitate more rapid system configuration and image building. These are listed below.

- `setenv.sh` → Script for rapidly setting up a development environment.
- `cfg_uboot.sh` → Script for rapidly configuring the u-boot building setup.
- `mk_uboot.sh` → Script for building the u-boot and copying the “u-boot” to the “image” folder after construction.
- `cfg_kernel.sh` → Script for rapidly configuring the kernel building setup.
- `mk_kernel.sh` → Script for building the “zImage” and copying the “zImage” to the “image” folder after construction.
- `mksd-linux.sh` → Script for setting up a bootable SD card for building images.
- `mkinand-linux.sh` → Script for accessing the Linux OS on the SD card and burning to eMMC flash.

3.2.2.6 Source

This folder contains the subdirectories “linux-4.1.6+gitAUTOINC+52c4aa7cdb-g52c4aa7” and “u-boot-2015.07+gitAUTOINC+d49aa5effa”, which feature the source codes for the Linux kernel and u-boot.

The Linux OS features true multitasking, virtual memory, shared libraries, demand loading, shared copy-on-write executables, appropriate memory management, and multi-task networking.

Linux is portable and can be used on most general-purpose 32-bit architectures that feature a paged memory management unit (PMMU) and GNU C compiler (GCC) port (part of the GNU Compiler Collection, GCC). Linux has been ported to a number of architectures without a PMMU, although the functionality was somewhat limited. Linux has also been ported to itself.

The main subdirectories under “Linux-4.1.6” are listed below.

- arch → Contains items related to the hardware platform, most of which are for the CPU.
- block → Contains block setting information.
- crypto → Contains encryption technology supported by the kernel.
- Documentation → Contains kernel documentation.
- drivers → Contains hardware drivers.
- firmware → Contains firmware data for old hardware.
- fs → Contains the file system supported by the kernel.
- include → Contains the header definition for the other programs used.
- init → Contains the initial kernel functions.
- ipc → Defines the communication for each Linux program.
- kernel → Defines the kernel process, status, schedule, and signal.
- lib → Contains some libraries.
- mm → Contains data related to the memory.
- net → Contains data related the network.
- security → Contains security settings.
- sound → Contains the audio module.
- virt → Contains data related to virtual machines.

Plenty of online documents, books, and magazines provide information regarding both Linux-specific and general UNIX issues.

Various README files located in /source/ linux-4.1.6+gitAUTOINC+52c4aa7cdb-g52c4aa7/Documentation provide kernel-specified installations and notes for drivers. Refer to /source/linux-4.1.6+gitAUTOINC+52c4aa7cdb-g52c4aa7/Documentation/00-INDEX for a description of the purpose of each README/note.

3.3 Setting Up a Development Environment

All instructions in this guide are based on a Ubuntu 14.04 LTS development environment. Please install Ubuntu 14.04 LTS on your PC/NB in advance.

After obtaining the RSB-4220 Linux source code package, refer to the following instructions to extract data to your development environment:

1. Copy the “AM335XLBVxxxx_yyyy-mm-dd.bin” package to /root/.
2. Start your “terminal” on Ubuntu 14.04 LTS.
3. **\$sudo su** (Change to “root” authority)
4. Input user password
5. **#cd /root/**
6. **#chmod a+x AM335XLBVxxxx_yyyy-mm-dd.bin**
7. **#!/AM335XLBVxxxx_yyyy-mm-dd.bin**
8. **Input “yes”**
9. The folder “AM335XLBVxxxx_yyyy-mm-dd” should be visible on /root/.

Note! *xxxx denotes the version number, yyyy denotes the year, mm denotes the month, and dd denotes the day.*



For example, AM335XLBV1010_2014-10-01.

Advantech provides a script for rapidly setting up a development environment. Follow the steps below to setup your development environment.

1. Open “terminal” on Ubuntu 14.04 LTS.
2. **\$sudo su** (Change to “root” authority)
3. Input user password
4. **#cd /root/AM335XLBVxxxx_yyyy-mm-dd/scripts/**
5. **#. setenv.sh** (To configure the development environment automatically.)
6. Users can now begin coding the source code, building images, and compiling applications.

3.3.1 Setenv.sh

This script can be used to rapidly configure the development environment and system folder paths. Users can also add/modify setenv.sh themselves if they have added/changed the folders and paths.

Note! *The “#source setenv.sh” script must be run each time a new “terminal” utility is opened.*



Note! Users are advised to change to the “root” authority when using the source code.



3.4 Build Instructions

This section provides instructions on how to build the u-boot and Linux kernel.

3.4.1 Build a U-Boot Image

Advantech provides a script for building the u-boot rapidly. Users can build a u-boot image by following the steps below.

1. Open “terminal” on Ubuntu 14.04 LTS.
2. **\$sudo su** (Change to “root” authority.)
3. Input user password.
4. **#cd /root/AM335XLBVxxxx_yyyy-mm-dd/scripts**
5. **#. setenv.sh** (To configure the development environment automatically.)
6. **#./cfg_uboot.sh am335x_rsb4220_defconfig** (To set the u-boot configuration automatically.)
7. **#./mk_uboot.sh** (Start to build the u-boot.)
8. Building of the u-boot.img is initiated and located in ../image.

3.4.2 Build a Linux Kernel Image

Advantech provides a script for building a “zImage” rapidly. Users can build a zImage by following the steps below.

1. Open “terminal” on Ubuntu 14.04 LTS.
2. **\$sudo su** (Change to “root” authority.)
3. Input user password.
4. **#cd /root/AM335XLBVxxxx_yyyy-mm-dd/scripts**
5. **#. setenv.sh** (To configure the development environment automatically.)
6. **#./cfg_kernel.sh tisd_k_am335x-rsb4220_defconfig** (To set the zImage configuration automatically.)
7. **#./mk_kernel.sh** (Start to build the zImage.)
8. Building of the zImage is initiated and located in ../image.

3.4.3 Build a Log

The build log can be located in the “./AM335XLBVxxxx” folder. If an error message is encountered when building the Linux kernel, check the log file for additional details.

3.5 Create Linux System Boot Media

RSB-4220 supports booting from an SD card or onboard flash. This section provides instructions on how to build an image for RSB-4220 Linux system boot media.

3.5.1 Storage Information (eMMC/SD Card)

The storage devices are named as follows:

Device	Name
SD card	/dev/mmcblk0
eMMC	/dev/mmcblk1

3.5.2 Create a Linux System SD Card

3.5.2.1 From a Pre-Built System Image

A pre-built image is available for download from the Advantech website. Follow the instructions below to create an SD card for boot up.

1. Copy "RSB-4220LIVxxxx_yyyy-mm-dd.tar.gz" package to your /root/.
2. Open "terminal" on Ubuntu 14.04 LTS.
3. **\$sudo su** (Change to "root" authority.)\
4. Input the user password.
5. **#cd /root/**
6. **#tar xzvf RSB-4220LIVxxxx_yyyy-mm-dd.tar.gz** (unzip files)
7. Insert an SD card into your development computer.
8. Identify the SD card location, for example, "/dev/sdb".
9. **#cd ./RSB-4220LIVxxxx_yyyy-mm-dd/scripts**
10. **#!/mksd-linux.sh /dev/sdb**
11. Type "y" (Copy files and wait until the process is marked as [Done].)

Then, insert the Linux system SD card into RSB-4220 and the system will boot up into a Linux environment.

3.5.2.2 From a Source Code Package

Using the RSB-4220 Linux source code package, follow the instructions below to create a Linux system SD card for boot up.

1. Open "terminal" on Ubuntu 14.04 LTS.
2. **\$sudo su** (Change to "root" authority.)
3. Input the user password.
4. Insert an SD card into your development computer.
5. Identify the SD card location, for example, "/dev/sdb".
6. **#cd /root/AM335XLBVxxxx_yyyy-mm-dd/scripts**
7. **#!/mksd-linux.sh /dev/sdb**
8. Type "y" (Copy files and wait until the process is marked as [Done].)

Then, insert the Linux system SD card into the RSB-4220 SD card slot (SD1), and the system will boot up into a Linux environment.

3.5.3 Boot From Onboard Flash

If you already have a Linux system SD card, follow the instructions below to copy the content to onboard flash and then boot from onboard flash. Advantech has provided the script “mkinand-linux.sh” to speed up the process of installing a system image onto onboard flash.

1. Refer to Section 3.5.2 to set up a Linux system SD card.
2. Insert this Linux system SD card into RSB-4220 and connect the serial console.
3. On the RSB-4220 platform, type **#root** (Login)
4. On the RSB-4220 platform, type **#cd /mk_inand**
5. On the RSB-4220 platform, type **#./mkinand-linux.sh /dev/mmcblk1**
6. Power off the system and remove the SD card.

Now you can boot from onboard flash without an SD card.

3.6 Debug Message

RSB-4220 can be connected to a host PC (Linux or Windows) via a console cable and debug port adapter. To communicate with a host PC, a serial communication program, such as HyperTerminal, Tera Term, or PuTTY, is required. Instructions for setting up a serial console, “HyperTerminal”, on a Windows host PC are provided below.

1. Connect RSB-4220 to your Windows host PC using a serial cable, debug port adapter, and console cable.
2. Open HyperTerminal on your Windows host PC and configure the settings to the values shown in Figure 3.5.
3. Power up the board. A bootloader prompt will be displayed on the terminal screen.

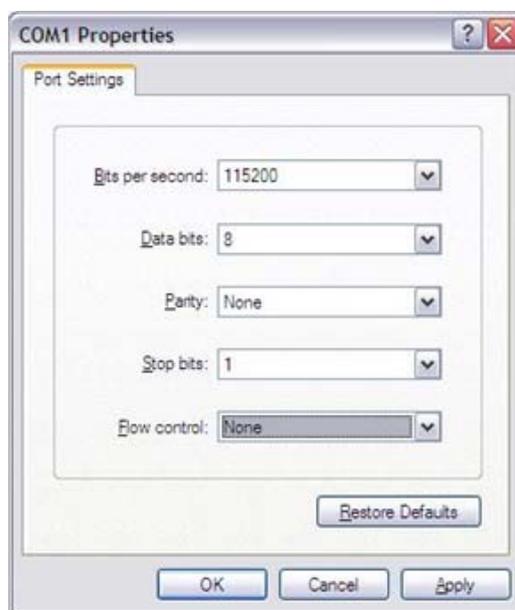


Figure 3.3 HyperTerminal Settings for Serial Console Setup

3.7 Linux System Configuration and Use

3.7.1 Display Output Setting

3.7.1.1 LVDS Settings

LDB-XGA is an example resolution for the LVDS panel. Input the actual resolution, such as 800x480 or 1024x768, of the LVDS panel. The system will configure the corresponding parameters automatically.

Should users experience difficulty activating the LVDS panel, check the panel data-sheet to configure the panel-related parameters. The LVDS video mode database is stored in `linux-4.1.6+gitAUTOINC+52c4aa7cdb-g52c4aa7/arch/arm/boot/dts/am335x-rsb4220.dts`. Users can add new modes for the LVDS panel.

```
display-timings {
    native-mode = <&timing0>;
    ...
    ...
    timing2: 1360x768 {
        hactive          = <1024>;
        vactive          = <768>;
        hback-porch     = <120>;
        hfront-porch    = <120>;
        hsync-len       = <104>;
        vback-porch     = <20>;
        vfront-porch    = <20>;
        vsync-len       = <1>;
        clock-frequency = <60000000>;
        hsync-active    = <0>;
        vsync-active    = <0>;
    };
};
```

3.7.1.2 Display Settings

If you want to select or change the type of LVDS panel, please set the parameter of "native-mode" in `linux-4.1.6+gitAUTOINC+52c4aa7cdb-g52c4aa7/arch/arm/boot/dts335x-rsb4220.dts` file to your panel parameter, which is shown as below:

```
display-timings {
    native-mode = <&timing2>;
    timing0: 800x480 {
        hactive      = <800>;
        vactive      = <480>;
        hback-porch  = <127>;
        hfront-porch = <127>;
        hsync-len    = <2>;
        vback-porch  = <13>;
        vfront-porch = <13>;
        vsync-len    = <2>;
        clock-frequency = <33000000>;
        hsync-active = <0>;
        vsync-active = <0>;
    };
    timing1: 1024x768 {
        hactive      = <1024>;
        vactive      = <768>;
        hback-porch  = <152>;
        hfront-porch = <48>;
        hsync-len    = <104>;
        vback-porch  = <3>;
        vfront-porch = <23>;
        vsync-len    = <4>;
        clock-frequency = <57000000>;
        hsync-active = <0>;
        vsync-active = <0>;
    };
    timing2: 1360x768 {
        hactive      = <1024>;
        vactive      = <768>;
        hback-porch  = <120>;
        hfront-porch = <120>;
        hsync-len    = <104>;
        vback-porch  = <20>;
        vfront-porch = <20>;
        vsync-len    = <1>;
        clock-frequency = <60000000>;
        hsync-active = <0>;
        vsync-active = <0>;
    };
};
```

3.7.2 Service Configuration

RSB-4220 is equipped to support the following five common network services: TFPT, FTP, SSH, Telnet, and HTTP.

3.7.2.1 TFTP Server

Upon booting up RSB-4220, TFTP service will be initiated by default. The working directory for the TFTP server is /tftpboot. Users must execute “chmod 777 /tftpboot” on RSB-4220 to enable operation of the TFTP server. Then, users can communicate with RSB-4220 by TFTP via the TFTP client on the host PC. Use the command line to obtain and store files as follows:

```
hostPC$ tftp TARGET_SYSTEM_IP
tftp>get file1
tftp>put file2
```

Note! The command “get file1” is for downloading File 1 from the TFTP server. File 1 exists in the /tftpboot directory on RSB-4220.



The command “put file2” is for uploading File 2 to the TFTP server. If File 2 is uploaded successfully, File 2 will be saved to the /tftpboot directory on RSB-4220.

You can also input the following commands on RSB-4220 to transmit files to the remote device:

```
root@am335x-adv:~# tftp -g -r file1 TARGET_SYSTEM_IP
```

This command can be used to download file1 under tftp path of the target device to the current path of RSB-4220. "TARGET_SYSTEM_IP" is IP address of the target device.

```
root@am335x-adv:~# tftp -p -l file2 TARGET_SYSTEM_IP
```

This command can be used to upload file2 under the current path of RSB-4220 to tftp path of the target device. "TARGET_SYSTEM_IP" is IP address of the target device.

The service start command is as follows:

```
root@am335x-adv:/ # /etc/init.d/tftpd start
```

The service stop command is as follows:

```
root@am335x-adv:/ # /etc/init.d/tftpd stop
```

3.7.2.2 FTP Server

The ftp server on RSB-4220 is vsftpd and you should manually start it using following command:

```
root@am335x-adv:/ # /etc/init.d/vsftpd start
```

The service stop command is as follows:

```
root@am335x-adv:/ # /etc/init.d/vsftpd stop
```

Note! After activating the FTP server, manually add the user FTP.



```
root@am335x-adv:/ # chown root:root /home/ftp/
root@am335x-adv:/ # chmod 777 /home/ftp/ -R
```

Now, users can communicate with RSB-4220 by FTP via the user FTP.

3.7.2.3 SSH Server

Upon booting up RSB-4220, SSH service will be initiated by default. Run the following command on the host PC to log into RSB-4220:

```
hostPC$ sudo ssh -l root TARGET_SYSTEM_IP
```

The service start command is as follows:

```
root@am335x-adv:/ # /etc/init.d/dropbear start
```

The service stop command is as follows:

```
root@am335x-adv:/ # /etc/init.d/dropbear stop
```

3.7.2.4 Telnet Server

Upon booting up RSB-4220, Telnet service will be initiated by default. Run the following command on the host PC to log into RSB-4220:

```
hostPC$ sudo telnet TARGET_SYSTEM_IP
```

The service start command is as follows:

```
root@am335x-adv:/ # /etc/init.d/telnetd start
```

The service stop command is as follows:

```
root@am335x-adv:/ # /etc/init.d/telnetd stop
```

3.7.2.5 HTTP Server

The platform supports an embedded web server named lighttpd, which the Matrix GUI is based on.

The service start command is as follows:

```
root@am335x-adv:/ # /etc/init.d/lighttpd start
```

The service stop command is as follows:

```
root@am335x-adv:/ # /etc/init.d/lighttpd stop
```

3.7.3 Network Configuration

3.7.3.1 Configuration via Console

It's important to note that our file system has ported netplug to manage the network services, so it is not recommended that users manually execute ifconfig, route, dhclient or other related commands to configure the network. The network usually gets dynamic IP address by default. If static IP address is needed, you can refer to the below steps:

1. Firstly, open "/etc/netcfgfile/temp.static.netcfg.eth0" and "/etc/netcfgfile/temp.static.netcfg.eth1" file, then modify ETH0_IP, ETH1_IP, NAMESERVER, DEFAULT_GW and other variables based on your own specific needs.
2. Secondly, execute the following commands to copy the configuration file to the appropriate location.

```
advantech# cp /etc/netcfgfile/temp.static.netcfg.eth0 /etc/adv.d/netcfg.eth0
```

```
advantech# cp /etc/netcfgfile/temp.static.netcfg.eth1 /etc/adv.d/netcfg.eth1
```

3. Lastly, reboot the device.

3.7.4 Date/Time Configuration*

Use the tool provided to modify the system time.

Click on the "Time Settings" icon shown on screen to initiate Advantech's Date/Time Settings utility.

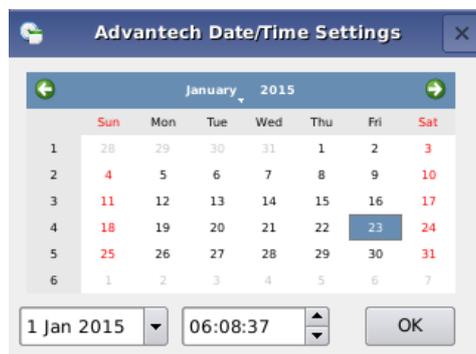


Figure 3.4 Date/Time Settings

After adjusting the time, click "OK" to save all changes. RTC time will automatically synchronize to the time set.

3.7.5 About the System

Note!



An alternative way to obtain the system version information is to run the "version" command shown below.

```
root@am335x-adv:~# version
Bsp version: RSB-4220 V2.00
Device name: RSB-4220
Release date: 2016-09-08
Kernel version: 4.1.6-g52c4aa7
```

3.7.6 Brightness Control

A GUI application is provided for brightness control. This enables users to adjust the brightness of the screen.

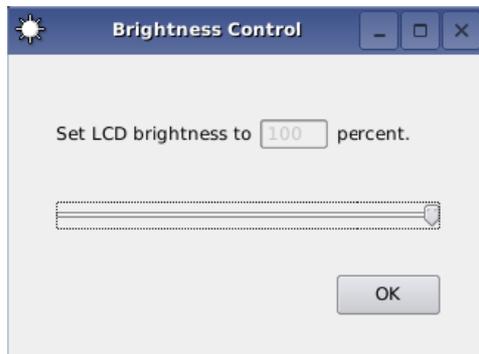


Figure 3.5 Brightness Control

3.7.7 Serial Tools

The system is equipped with five serial ports, ttyO1, ttyO2, ttyO3, ttyO4, and ttyO5. A serial test tool is provided for validating the serial ports.



Figure 3.6 Serial Control

3.7.8 Matrix GUI User Guide

3.7.8.1 Overview

Upon booting up the target system, the Matrix GUI will be initiated automatically. Matrix is an HTML 5-based application launcher designed to highlight the available applications and demos provided. Two forms of Matrix exist, local and remote. All the example applications and demos are available when using either the local or remote version. Matrix is presented as a 6x4 or 4x3 matrix of icons, depending on the display resolution.

The launcher for Matrix is a simple QT application that displays a Webkit base browser that points to the URL <http://localhost:80>.

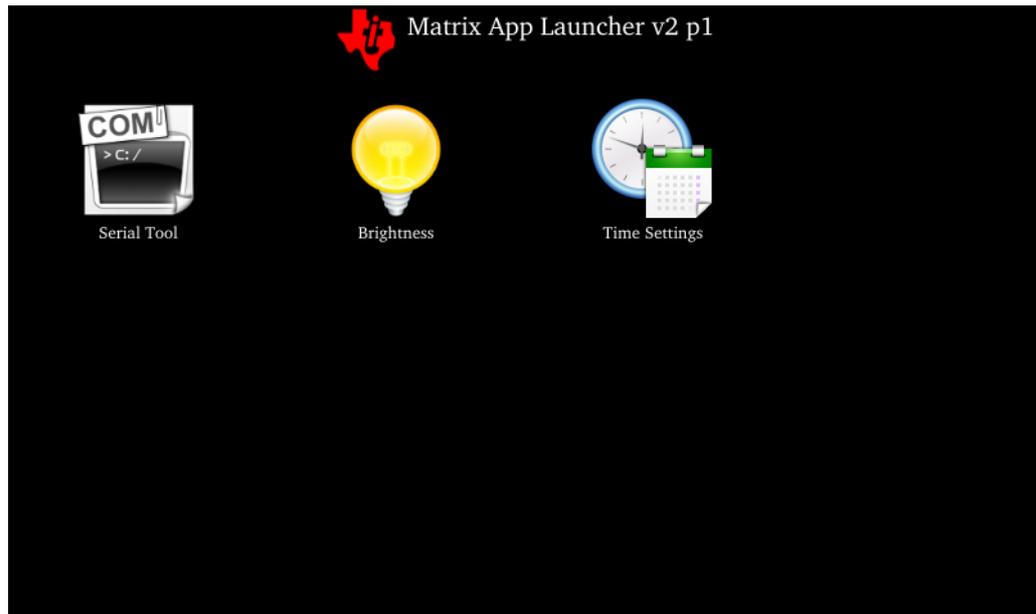


Figure 3.7 Matrix

3.7.8.2 Launching and Stopping the Matrix*

If the Matrix GUI is not automatically initiated upon system boot up, the program can be started manually using the following command:

```
advantech# /etc/init.d/matrix-gui-2.0 start
```

To set Matrix to activate upon system boot up by default, run the following command on RSB-4220:

```
advantech# ln -s /etc/init.d/matrix-gui-2.0 /etc/rc5.d/S97matrix-gui-2.0
```

To cancel the default startup, simply delete the S97matrix-gui-2.0 file.

For more information on the use of Matrix, please refer to the following website:

http://processors.wiki.ti.com/index.php/Matrix_Users_Guide

3.7.8.3 Adding a Matrix Application

Instructions for adding a new application or directory to Matrix are provided below.

1. Create a new folder on the target file system at `/usr/share/matrix-gui-2.0/apps/`. The name should clearly describe the application or directory. The name of the folder must differ from that of all existing folders in the target location.
2. Create a desktop file according to the parameters discussed below. We recommend that the name of the desktop file match the name of the newly created folder. The desktop file name cannot contain spaces. The parameters of the desktop file should be set according to whether the new addition to Matrix is an application or directory. Input this information into the Type field. The desktop file must have the desktop suffix.
3. Update the Icon field on the desktop to refer to any existing Icon in the `/usr/share/matrix-gui-2.0` directory or subdirectories. A new 96x96 png image can also be added and located in the newly created folder.
4. For applications, an HTML file containing an application description can be added to the newly created directory. If a description page is added, update the X-Matrix-Description field in the .desktop file.
5. Refresh Matrix using the application “Refresh Matrix” located in the Settings submenu.

Blank template icons for Matrix can be found at the following link:

http://gforge.ti.com/gf/download/frsrelease/712/5167/blank_icons_1.1.tar.gz

The .desktop file is based on the standard specified at the URL:

<http://standards.freedesktop.org/desktop-entry-spec/latest/>

Additional fields unique to Matrix are also added.

The format of each parameter is as follows:

`<Field>=<Value>`

The fields and values are case sensitive.

3.7.9 Add Startup Items

1. To remove startup items, run the following command:
`update-rc.d [-n] [-f] [-r <root>] <basename> remove`
 basename is your service script name
 eg. `update-rc.d -f matrix-gui-2.0 remove`
2. To add startup items
 First, ensure that the service script exists, then run the following command:
`update-rc.d [-n] [-r <root>] [-s] <basename> start|stop NN runlvl [runlvl] [...] .`
 start|stop: when system start /shutdown the basename will run automatically
 NN: 0~99
 runlvl: RSB4220 runlevel is 5(default);
 eg. `update-rc.d networking start 40 5 .` (The “.” is also a parameter. If missing, the operation will be failed)
 then you can find `S40networking` in the `rc5.d` directory.

3.7.10 OTG Mode Selection

USB connector of RSB4220 supports two modes: Host and Slave, which require software and hardware support. The system is set to Host mode by default. If you do not need to switch between the two modes, no operation is required; if you need to switch, please refer to the following steps:

1. Please refer to section 2.1.3 “Jumper Setting” for the jumper selection of J4: 1-2 means “USB host” and 2-3 means “USB OTG device”.
2. Execute one of the following commands to switch to the corresponding mode after the terminal is powered on. |

```
root@am335x-adv:/ # echo "host" > /etc/otg-mode
```

or

```
root@am335x-adv:/ # echo "peripheral" > /etc/otg-mode
```

Notice: The first command sets USB connector to Host mode, while the second one sets it to Slave mode. It should be consistent with the jumper setting in Step 1.

3. Reboot the system.

3.8 Development Guide and Reference

3.8.1 Development of C/C++ Programs

This section provides instructions on how to write a sample “Hello World” application. Please refer to the following steps:

1. Open “terminal” on Ubuntu 14.04 LTS, and change to “root”.

```
$ sudo su
```

Input the user password.

2. Create a development environment using the following command:

```
#source /usr/local/cross_compiler/linux-devkit/environment-setup
# cd /root/RSB-4220LBVxxx_yyyy_mm_dd/source
# mkdir helloworld
# cd helloworld
# gedit hellowrold.c
```

3. Edit helloworld.c using the following source code:

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

Save the file and exit.

4. Compile helloworld.c using the following command:

```
# $CC -o helloworld helloworld.c
```

Hello world should be visible in the current directory.\

5. Run the helloworld executable file on RSB-4220.
Insert the Linux system SD card into the development computer.

```
# cp helloworld /media/rootfs/tool
```

Note! The Linux system SD card mounting point is `/media/rootfs`.



Remove the SD card and insert it into RSB-4220, then open the serial console.
 On the RSB-4220 platform, type **#root** (Login)
 On the RSB-4220 platform, type **#cd /tool**
 On the RSB-4220 platform, type **#./helloworld**
 “Hello World!” should be visible on RSB-4220.

3.8.2 Developing GUI Programs with a Qt Library

The development kit can be used to develop a Qt-based GUI program. Follow the instructions below to quickly convert a Qt project into a GUI application for RSB-4220.

1. On the host PC, setup a Qt build environment.

```
#source /usr/local/cross_compiler/linux-devkit/environment-setup
```

2. Build Qt instructions are as follows:

```
# cd projectdir
# qmake projectName.pro
# make
3.Run QT demo:
# ./qtappName -platform linuxfb
```

3.8.3 Demo Program Source Code

3.8.3.1 Serial Port Programming

Refer to `<BSP_PATH>/source/demo/uart` for an example of sending and receiving data via the serial port.

Receiving data

```
# ./uart_ctrl read /dev/ttyO1
```

Sending data

```
# ./uart_ctrl write /dev/ttyO2
```

Before programming the serial port, ensure that the serial port is in 232/422/485 mode.

Users can reference the uart demo source code to develop a uart application.

3.8.3.2 Watchdog Programming

RSB-4220 supports a hardware watchdog. The watchdog API follows POSIX standards. Valid timeout values range from 1 to 6553 seconds. If the desired timeout value is outside this range, the driver will set the timeout value to the default value (60 seconds).

Sample C code

```
#include <stdio.h>
#include <stdlib.h>
#include <fcntl.h>
#include <signal.h>
#include <linux/watchdog.h>
```

```

int fd;
int main(int argc, const char *argv[]) {
    int timeout = 10;
    /*open watchdog device, the watchdog device node is /dev/watchdog */
    fd=open("/dev/watchdog",O_WRONLY);

    /*set timeout to 10 seconds*/
    /*when set timeout value, the watchdog driver will enable the
    watchdog automatically.*/
    ioctl (fd, WDIOC_SETTIMEOUT, &timeout);
    while (1)
    {
        /*feed the watchdog every 5 seconds*/
        /*when call this function to feed watchdog, the watchdog
        will reset its internal timer so it doesn't trigger the
        board reset. If do not feed the watchdog more than 10
        seconds, the watchdog will timeout and the board will
        reset.*/
        ioctl( fd, WDIOC_KEEPAKIVE, NULL );
        sleep (5);
    }
    close (fd);
}

```

Other watchdog APIs also exist.

Disable the watchdog timer sample code.

```

/*if user want to disable the watchdog before timeout, call the fol-
lowing ioctl function*/
int i_dis = WDIOS_DISABLECARD;
ioctl( fd, WDIOC_SETOPTIONS, &i_dis );

```

Enable the watchdog timer sample code.

```

/*if user want to enable the watchdog again before timeout when it is
disabled, call the following ioctl function. */
int i_en = WDIOS_ENABLECARD;
ioctl( fd, WDIOC_SETOPTIONS, &i_en);

```

Obtain the current timeout value.

```

/*get the current timeout value the driver used*/
int timeout = 0;
ioctl (fd, WDIOC_GETTIMEOUT, &timeout);

```

Refer to the <BSP_PATH>/source/demo/watchdog folder for more information.

3.8.3.3 GPIO Programming

RSB-4220 features eight GPIOs. Refer to <BSP_PATH>/source/demo/gpio

Usage

```
./gpio_ctrl 200 out 1
```

Note! “200” refers to GPIO0; thus, 200 - 207 correspond to GPIO0 - GPIO7.
 “out” means output.
“1” is the value set to the corresponding GPIO port.

3.8.3.4 CAN Programming

Refer to <BSP_PATH>/source/demo/can_test.

Note! For the CAN data sending sample c code, refer to can_write.c.
 For the CAN data receiving sample c code, refer to can_read.c.

3.8.3.5 Brightness Programming

Refer to <BSP_PATH>/source/demo/brightness

The brightness driver provides the sys interface. The brightness value can be set and obtained using the sys file as follows:

/sys/class/backlight/pwm-backlight/brightness

Set the brightness value using the following command:

```
# echo "3" > /sys/class/backlight/backlight/brightness
```

Note! The value should be between 1-8.



Obtain the current brightness value using the following command:

```
# cat /sys/class/backlight/backlight/brightness
```


Chapter 4

System Recovery

This chapter explains how to recover a Linux operating system if accidentally damaged.

4.1 System Recovery

This section describes the procedures for restoring an eMMC image. If the onboard flash image is accidentally destroyed, follow the instructions provided below to perform a system recovery.

1. Copy the “RSB-4220LIVxxxx_yyyy-mm-dd.tar.gz” package to your /root/.
2. Open “terminal” on Ubuntu 14.04 LTS.
3. **\$sudo su** (Change to “root” authority.)
4. Input the user password.
5. **#cd root/**
6. **#tar xzvf RSB-4220LIVxxxx_yyyy-mm-dd.tar.gz** (unzip files)
7. Insert an SD card into the development computer
8. Identify the SD card location, for example, “/dev/sdb”.
9. **#cd ./RSB-4220LIVxxxx_yyyy-mm-dd/scripts**
10. **#!/mksd-linux.sh /dev/sdb**
11. Type “y” (Copy files and wait until the process is marked as [Done].)
12. Connect the console cable to the debug port (CN1). Open the serial console program on Ubuntu 14.04 LTS, and set the baud rate to 115200. For detailed console settings, refer to Section 3.6.
13. On the RSB-4220 platform, type **#root** (login)
14. On the RSB-4220 platform, type **#cd /mk_inand**
15. On the RSB-4220 platform, type **#!/mkinand-linux.sh /dev/mmcb1k1**
16. On the RSB-4220 platform, type “y “
(Copy files and wait until the process is marked as [Done].)
17. Power off the platform and remove the SD card.

Chapter 5

Advantech Services

This chapter outlines Advantech's design-in service, technical support, and warranty policy for RSB-4220.

5.1 RISC Design-in Services

With the spread of industrial computing, a whole range of new applications have been developed, resulting in a fundamental change in the IPC industry. In the past, system integrators (SIs) were accustomed to completing projects without external assistance. However, current working models have changed. Due to diverse market demands and intense competition, cooperation (both upstream and downstream) for vertical integration has become a much more effective means to create competitive advantages. ARM-based CPU modules resulted from this trend. Addressing market requirements for specialization by concentrating all necessary components onto the CPU module and placing other parts on the carrier board offers increased flexibility while retaining low power consumption benefits.

Advantech has many years of experience in the industrial computer industry. The information provided below aims to address the most common customer questions regarding the implementation of modular designs.

General I/O design capability

Although customers possess vertical integration abilities and sufficient professional application knowledge and core competitiveness, a lack of expertise and experience in general power and I/O design generates numerous challenges, particularly with the integration of CPU modules and carrier boards.

Information acquisition

Even if sufficient information is obtained to make appropriate decisions regarding specialized vertical applications, some customers experience difficulties in platform design and communicating with CPU or chipset manufacturers. This increases the challenges of carrier board design and extends the time-to-market, potentially resulting in lost market opportunities.

Software development and modification

Compared to x86 architectures, RISC architectures use simpler instruction sets; therefore, the software support for x86 platforms cannot be applied to RISC platforms. System integrators must develop unique software and integrate the hardware and software themselves. Compared to x86 platforms, RISC platforms have less support for board support packages (BSP) and drivers. Although driver support is provided, SIs still have to integrate them with the core system. Moreover, the BSPs provided by CPU manufacturers are typically for carrier board design. Thus, establishing a software development environment for SIs is fairly challenging.

To address this need, Advantech proposed the concept of streamlined design-in support services for RISC-based computer on modules (COM). With a dedicated professional design-in services team, Advantech actively participates in carrier board design and problem solving. Our services not only enable customers to effectively distribute their resources, but also reduce R&D manpower costs and hardware investment.

Close interactive relationships with leading original manufacturers of CPUs and chipsets, such as ARM, TI, and Freescale, enable Advantech to assist customers with solving communication and technical support difficulties, reducing the uncertainties of product development. Advantech's professional software team also emphasizes the provision of a comprehensive BSP and assists customers in establishing a software development environment for their RISC platforms.

Advantech's RISC design-in services helps customers overcome problems and achieve a faster time-to-market by streamlining the design process.

Advantech's RISC design-in service provides comprehensive support for the various stages of the development process, specifically, planning, design, integration, and validation.

Planning stage

Before adopting Advantech's RISC COM, customers must conduct a comprehensive survey of the product features, specifications, and software compatibility. Advantech offers a RISC customer solution board (CSB) as an evaluation tool for carrier boards designed when developing RISC COMs. During the planning stage, customers can use this evaluation board to assess RISC modules and test peripheral hardware. Additionally, Advantech provides standard software BSPs for RISC COM to enable customers to define their product specifications and verify I/O and performance. We offer not only hardware planning and technology consultations, but also software evaluations and peripheral module recommendations (such as Wi-Fi, 3G, or BT). Resolving customer concerns is Advantech's primary objective at this stage. Because product evaluation, especially regarding performance and specifications, is a key aspect of planning, we endeavor to assist our customers conduct all the necessary tests for their RISC COM.

Design stage

When a product moves into the design stage, Advantech supplies a carrier board design guide for reference. The carrier board design guide contains pin definitions for the COM connector and explains the carrier board design limitations and recommendations, providing a clear guideline for customers to use when developing carrier boards. Regarding different form factors, Advantech offers a complete pin-out check list for various form factors, such as Q7, ULP, and RTX2.0, allowing customers to examine the carrier board signals and layout design accordingly. In addition, our team can assist customers in reviewing the placement/layout and schematics to ensure that the carrier board design meets their requirements. For software development, Advantech's RISC software team can assist customers with establishing a software development environment and evaluating the time and resources needed. If customers outsource software development to a third party, Advantech can also cooperate with the third party and provide proficient consultation. With Advantech's professional support, product design becomes substantially easier, potentially enhancing the final product quality.

Integration stage

This stage includes hardware/software integration, application development, and peripheral module implementation. For customers who lack sufficient knowledge and experience of platforms, analyzing integration problems can become quite time consuming. Peripheral module implementation is greatly influenced by the driver designs on carrier boards, and RISC platforms typically offer less support for ready-made drivers. Therefore, customers must learn through trial and error to finally arrive at the best solution. Advantech's team has years of experience in customer support as well as vast hardware/software development knowledge. Consequently, we can support customers by providing professional advice and information, ultimately shortening the development time and enabling more effective product integration.

Validation stage

After customers' ES sample is completed, the next step is verification. In addition to verifying functionality, the product's efficiency must also be assessed at this stage, especially for RISC platforms.

In its supportive capacity, Advantech primarily helps customers resolve problems related to testing and offers relevant tips and suggestions. By adopting an efficient verification process backed by our technical support, customers can easily optimize their applications. Furthermore, Advantech's team offers professional consultation regarding further testing and equipment usage, ensuring customers have the appropriate tools to efficiently identify and resolve problems to further enhance product quality and performance.

5.2 Contact Information

Advantech's Customer Service contact information is provided below.

Region/Country	Contact Information
America	1-888-576-9688
Brazil	0800-770-5355
Mexico	01-800-467-2415
Europe (toll free)	00800-2426-8080
Singapore & SAP	65-64421000
Malaysia	1800-88-1809
Australia (toll free)	1300-308-531
China (toll free)	800-810-0345 800-810-8389 Sales@advantech.com.cn
India (toll free)	1-800-425-5071
Japan (toll free)	0800-500-1055
Korea (toll free)	080-363-9494 080-363-9495
Taiwan (toll free)	0800-777-111
Russia (toll free)	8-800-555-01-50

Alternatively, our service team can be contacted through our website. After a contact request form is completed, a technical support engineer will contact you promptly.

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

5.3 Global Service Policy

5.3.1 Warranty Policy

The warranty policy regarding Advantech products is outlined below.

5.3.1.1 Warranty Period

Advantech's branded off-the-shelf products and any third-party off-the-shelf products used to assemble Configure-to-Order products are entitled to two years complete and prompt global warranty. Defects in product design, materials, and workmanship are covered from the date of shipment.

All customized products have a 15-month regional warranty by default. The actual product warranty terms and conditions may vary based on specific sales contracts.

All third-party products purchased separately will be covered by the original manufacturer's warranty, and shall not exceed one year of coverage through Advantech.

5.3.1.2 Repairs under Warranty

A replacement (cross shipment) can be obtained within the first 30 days of the purchase. If your product was purchased directly from Advantech and dead-on-arrival (DOA), contact your original Advantech supplier to arrange a DOA replacement. The DOA cross-shipment excludes any shipping damage, customized, and/or built-to-order products.

For products not declared DOA, customers are responsible for the cost of returning the product to an authorized Advantech repair facility. The cost of shipping replacement products to customers will be the responsibility of Advantech.

5.3.1.3 Exclusions from Warranty

The product is excluded from warranty if any of the following conditions are valid:

- The product has been found to be defective after the warranty period has expired.
- Warranty has been voided by the removal or alteration of any product or part identification labels.
- The product has been misused, abused, or subjected to unauthorized disassembly/modification; placed in an unsuitable physical or operating environment; improperly maintained by the customer; or failure caused by circumstances beyond the responsibility of Advantech, whether by accident or other cause. Such conditions will be determined by Advantech at its sole discretion.
- The product is damaged beyond repair due to a natural disaster, such as a lightning strike, flood, or earthquake.
- Customer-requested updates and upgrades to products already out of warranty.

5.3.2 Repair Process

5.3.2.1 Obtaining an RMA Number

All product returns must be accompanied by an Advantech return merchandise authorization (RMA) number. Any returns of defective units or parts without a valid RMA number will not be accepted. Such products will be returned to the customer at the customer's cost without prior notice.

An RMA number only indicates authorization to return a product; it does not guarantee approval for repair or replacement. To request an RMA number, sign into Advantech's RMA website (<http://erma.advantech.com.tw>) with an authorized user ID and password.

Customers must provide basic product and company information, as well as a detailed description of the problems encountered. Vague entries such as "does not work" and "failure" are not acceptable.

If you are unsure of the cause of the problem, please contact Advantech's application engineers (AE) who may be able to offer a solution that does not require sending the product for repair.

For cases where only a key defective part requires repair, the serial number for the entire product is still required. Otherwise, the part will be regarded as out of warranty.

5.3.2.2 Returning the Product for Repair

Customers can save time and meet end-user requirements by returning defective products to an authorized Advantech repair facility without extra cross-region charges. However, customers are required to contact their local repair center before our global repair service can be offered.

We recommend that customers send cards without the accessories (manuals, cables, etc.). Remove all unnecessary components, such as the CPU, DRAM, and CF card. For customers wishing to include accessory components in their return shipment (due to suspicions that they may be part of the problem), please clearly note the additional components included. Advantech assumes no responsibility for any items not listed. Furthermore, please ensure the "Problem Description" is enclosed.

European customers located outside of Europe are requested to use UPS as the forwarding company. We strongly recommend adding a packing list to all shipments. Please prepare a shipment invoice according to the following guidelines to reduce the goods clearance time:

1. Provide a low product value on the invoice to avoid additional custom charges. Additional fees levied by customs are the responsibility of the sender.
2. Write the text "Invoice for customs purposes only with no commercial value" on the shipment invoice.
3. Include RMA numbers, product serial numbers, and warranty status on the shipment invoice.
4. Detail the country of origin of the goods.

Additionally, attach an invoice with the RMA number to the shipment, write the RMA number on the outside of the carton, and attach a packing slip to reduce the handling time. Please also address the package to the Advantech Service Department and mark the package "Attn. RMA Service Department".

All products must be returned in properly packed ESD material or anti-static bags. Advantech reserves the right to return unrepaired items at the customer's expense if inappropriately packed.

“Door-to-door” transportation, such as speed post, is recommended for delivery. If sent as air cargo, any additional charges, such as clearance fees, will be the responsibility of the customer.

Should DOA cases fail, Advantech will assume full responsibility for the product and transportation charges. If the items are not DOA, but fail within warranty, customers will bear the freight charges. For out-of-warranty cases, customers are responsible for all costs and outward and inward transportation.

5.3.2.3 Service Charges

The product is excluded from warranty if any of the following conditions are valid:

- The product is repaired after the warranty period has expired.
- The product is tested or calibrated after expiry of the warranty period, and a No Problem Found (NPF) result is obtained.
- The product, although repaired within the warranty period, has been misused, abused, or subjected to unauthorized disassembly/modification; placed in an unsuitable physical or operating environment; improperly maintained by the customer; or failure caused by circumstances beyond the responsibility of Advantech, whether by accident or other cause. Such conditions will be determined by Advantech at its sole discretion.
- The product is damaged beyond repair due to a natural disaster, such as a lightning strike, flood, or earthquake.
- Customer-requested updates and upgrades to products already out of warranty.

If a product has been repaired by Advantech and then requires another repair for the same problem within three months of the initial repair, Advantech will perform the repair free of charge. However, such free repairs do not apply to products that have been misused, abused, or subjected to unauthorized disassembly/modification; placed in an unsuitable physical or operating environment; improperly maintained by the customer; or failure caused by circumstances beyond the responsibility of Advantech, whether by accident or other cause.

Please contact your nearest regional service center for a detailed service quotation.

Before undertaking out-of-warranty repairs, we will send you a pro forma invoice (P/I) detailing the repair charges. When you remit the funds, please reference the P/I number provided under “Our Ref”. Advantech reserves the right to deny repair services to customers who do not return a DOA unit or sign the P/I. Furthermore, Advantech will scrap defective products without prior notice if customers do not return the signed P/I within three months.

5.3.2.4 Repair Report

Advantech returns each product with a repair report that explains the result of the repair. A repair analysis report can also be provided to customers upon request. If the defect is not caused by Advantech design or manufacturing, customers will be charged US\$60 and US\$120 for in-warranty and out-of-warranty repair analysis reports, respectively.

5.3.2.5 Custody of Products Submitted for Repair

Advantech will retain custody of a product submitted for repair for one month while waiting for the return of a signed P/I or payment (A/R). If the customer fails to respond within this period, Advantech will automatically close the case. Advantech will take reasonable measures to contact the customer during this one month period.

5.3.2.6 Return Shipping to Customer

The forwarding company selected to handle RMA returns from Advantech to customers will be determined by Advantech. Other express services, such as UPS or FedEx, can be arranged upon request. The customer will be responsible for the extra cost incurred for alternative shipment. Should you require special arrangements, please clearly indicate them when shipping the product to us.

ADVANTECH

Enabling an Intelligent Planet

www.advantech.com

Please verify specifications before quoting. This guide is intended for reference purposes only.

All product specifications are subject to change without notice.

No part of this publication may be reproduced in any form or by any means, such as electronically, by photocopying, recording, or otherwise, without prior written permission from the publisher.

All brand and product names are trademarks or registered trademarks of their respective companies.

© Advantech Co., Ltd. 2017