

ROM-DK3420 ARM Starter Kit

Quick Starter Guide

Copyright Notice

This document is copyrighted, 2016 by Advantech Co., Ltd. All rights are reserved.

Advantech Co., Ltd. reserves the right to make improvements to the products described in this document at any time. Specifications are thus subject to change without notice.

No part of this document may be reproduced, copied, translated, or transmitted in any form or by any means without the prior written permission of Advantech Co., Ltd. Information provided in this manual is intended to be accurate and reliable. However, Advantech Co., Ltd. assumes no responsibility for its use, or for any infringements upon the rights of third parties which may result from its use.

All the trademarks of products and companies mentioned in this document belong to their respective owners.

Introduction

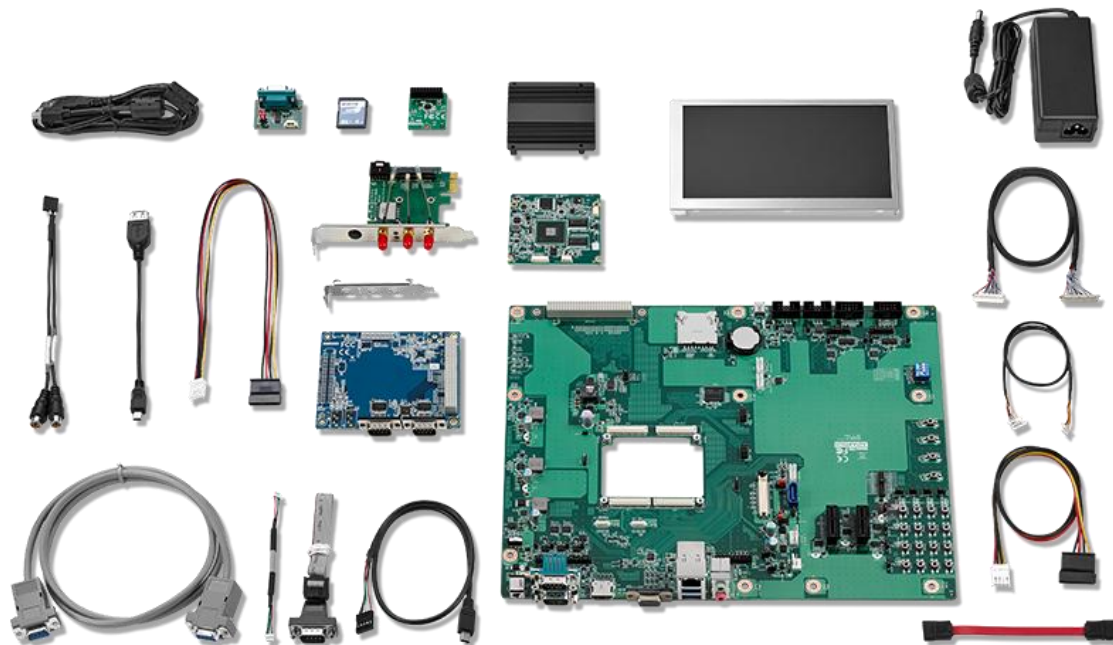
Advantech ARM-based Starter Kits provide a shortcut for ARM platform evaluation and development. The kit includes key elements of a development environment including main boards with high performance CPU, cables, adapter cards, LCD panel and power adapter. The Advantech ARM Starter Kit is designed to give you more time to concentrate on your own product innovation and make the whole process faster, easier and less stressful.

The Advantech ARM Starter Kit was released along with a built-in OS image in Linux which allows users start their evaluation immediately after they open the box. Other OS such as Android, Yocto Linux and Ubuntu are also available for different applications; each carefully verified for online download. The source codes of these supported OS are all open and available for users to help develop application code easier. In addition, Advantech offers tools for application development including Qt, a cross-platform tool for device creation, UI and application development, and Advantech WISE-PaSS/RMM APIs for device access, control and monitoring. All add-on software offerings are totally verified and 100% free of charge. Advantech is committed to offering the embedded community the best, most compact and reliable development platform for ARM-based solution development.

1. Board Overview



Rugged RTX iMX6 Starter Kit

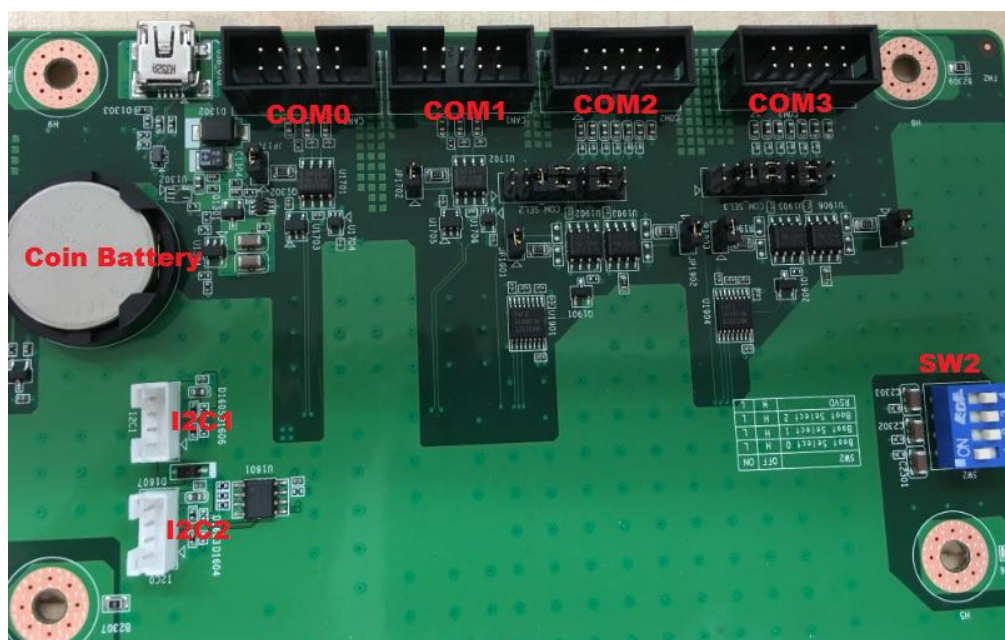


2. Packing List

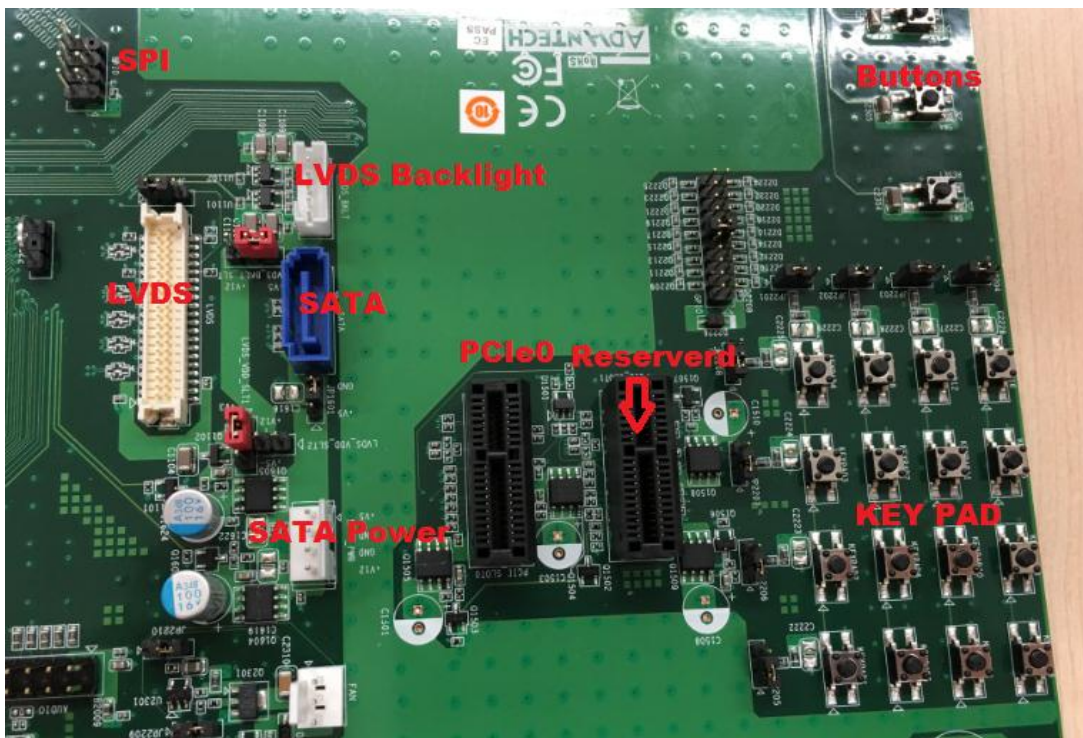
1700001524	US Power Cord
1700004711	SATA Cable
1700006911	USB OTG to Type A Cable
1700019077-11	USB OTG Cable
1700019474	Debug port to PC Cable
1700021882-01	LVDS Backlight Cable
1700021883-01	LVDS Cable
1700021941-01	SATA Power Cable
1700022373-01	Debug Port Cable
1700022840-01	SPDIF Cable (Reserved)
1701100300	COM Cable
1930004835	Screw M2.5x12.3L F/S D=5 H=0.8 (1+) ST Ni
1935020402	Screw M2.5x4L R/S D=4.5 H=1.7 (1+) ST Ni
1960065189N001	Heatsink
9680015491	Mini-PCle to PCIe adapter card
9696ED2000E	ROM-ED20 Debug Port Adapter Board
9696M34200E	ROM-3420 Dual Core 1GHz 0~60°C
9696M39000E	ROM-DB3900 RTX 2.0 Carrier Board 0~60°C
9696MEG510E	ROM-EG51 CODEC Board
9696MX5300E	SYSTEM BUS to UART Adapter Board
96LEDK-A070WV40NB1	7" LED PANEL 400N 800X480 Resolution
96PSA-A90W19V1-1	Adapter 100-240V 90W 19V
SQF-ISDS1-4G-82C	SDHC Class 10 4GB SD Card (0~70°C)

3. Board introduction

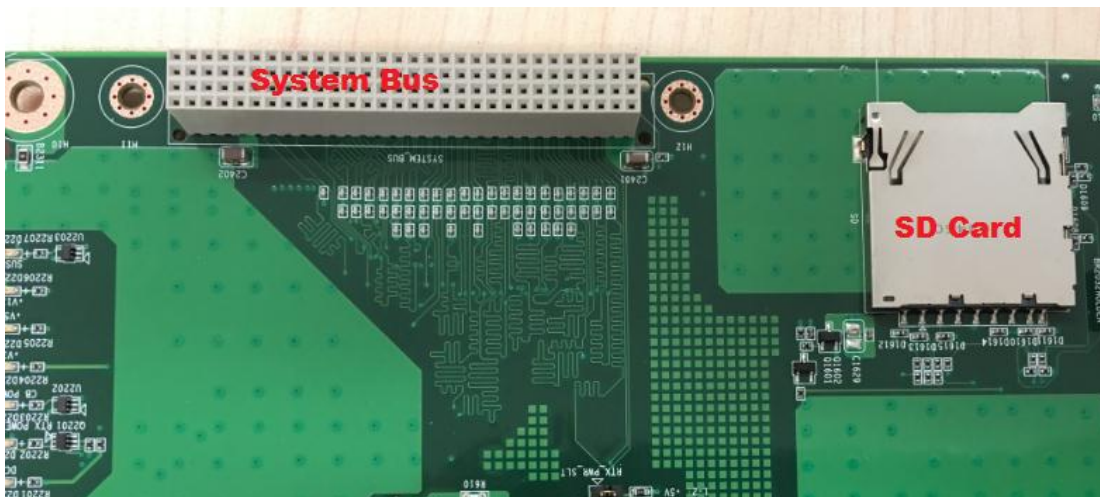
<PART A>



<PART B>



<PART C>



4. Screws

There are 3 different types of screws in this Starter Kit, they are designed for different purpose



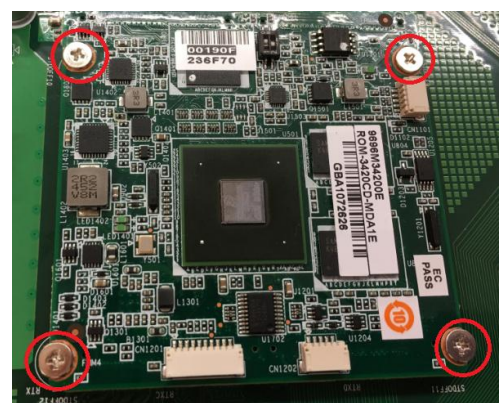
5. Board Assembly

5.1 Assemble ROM-3420 to its carrier board ROM-DB3900

① Assemble module to carrier board



② Fix the screws properly



5.2 Assemble 9696MX5300E to carrier board ROM-DB3900

① Assemble 9696MX5300E to carrier board



② Push Down



5.3 Assemble ROM-EG51 CODEC Board to carrier board ROM-DB3900

①Take out ROM-EG51 CODEC Board

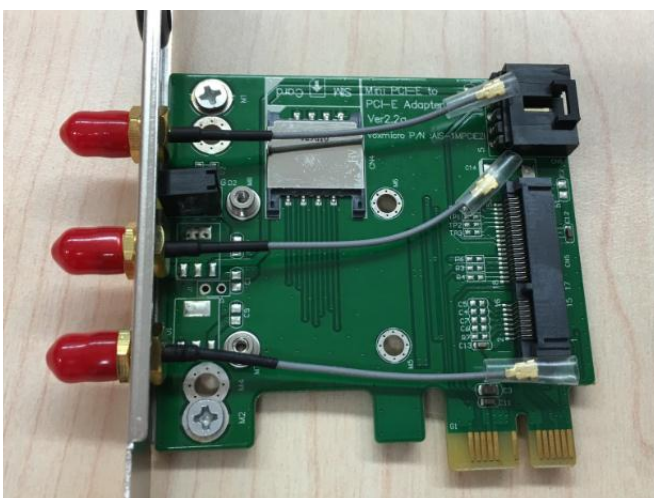


②Assemble ROM-EG51 CODEC Board to ROM-DB3900



5.4 Assemble 9680015491 PCIe to miniPCle board to carrier board ROM-DB3900

①Take out 9680015491 PCIe to miniPCle board



② Insert mini-PCIe card if needed. Mini-PCIe card is optional peripheral which is not included in the package of ROM-3420 Starter Kit



③ Fix the screws of mini-PCIe card



④ Insert the adapter card to ROM-DB3900



- ⑤ Connect USB Cable if your mini-PCle card needs USB signal



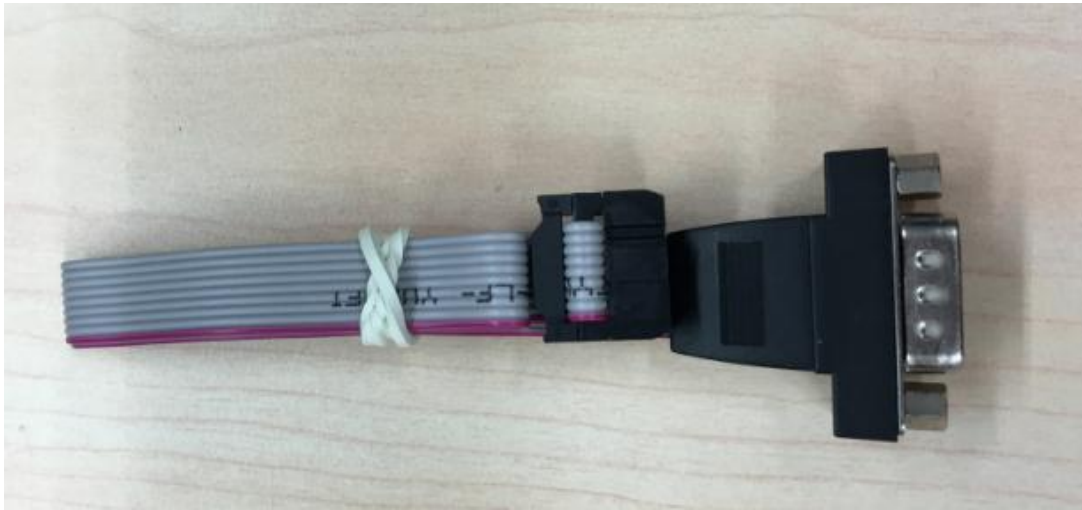
Use this Cable pulls USB signal
to 9680015491 PCIe to miniPCle
adapter board



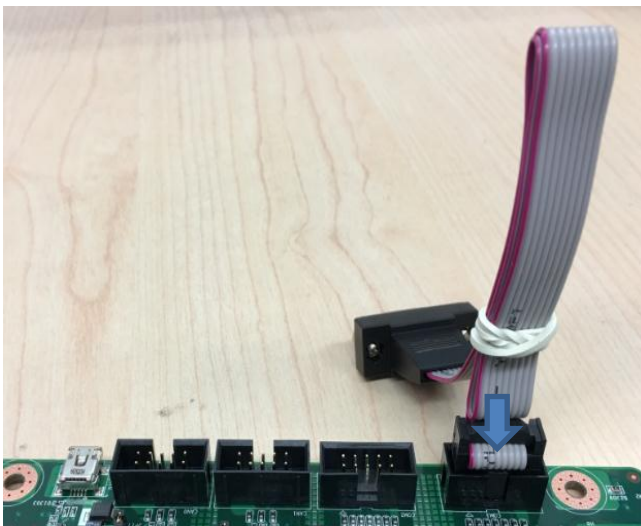
6. Cable Assembly

6.1 COM Port Cable

① Cable overview

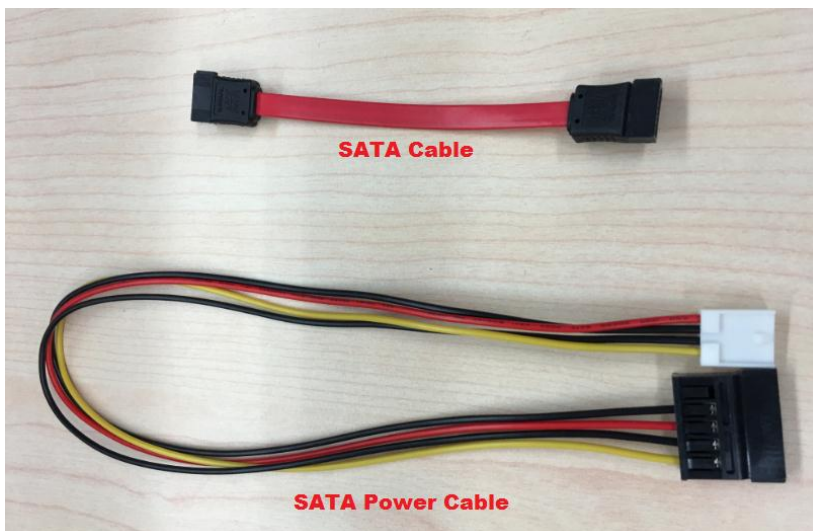


② Connect COM Port cable to Carrier board COM0~COM4

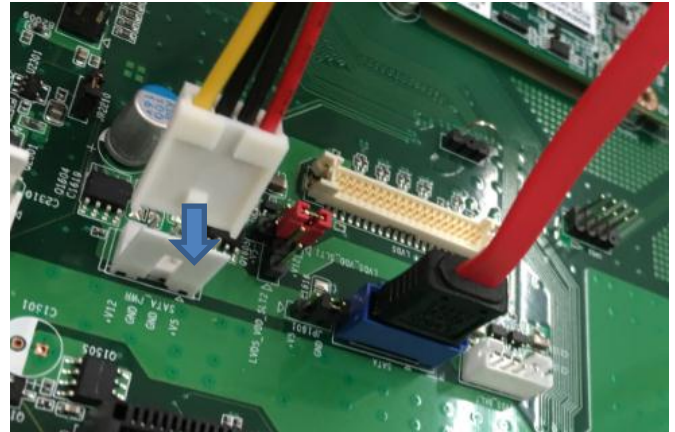
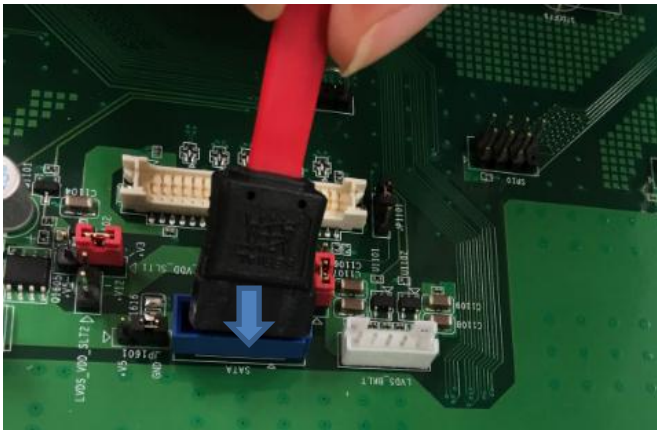


6.2 SATA Cable

① SATA Cable Overview

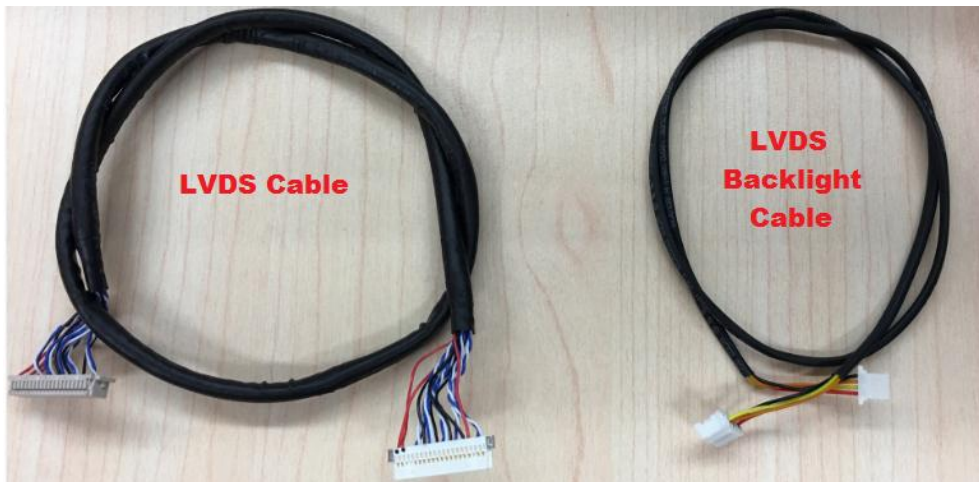


② Connect SATA cable and SATA Power Cable

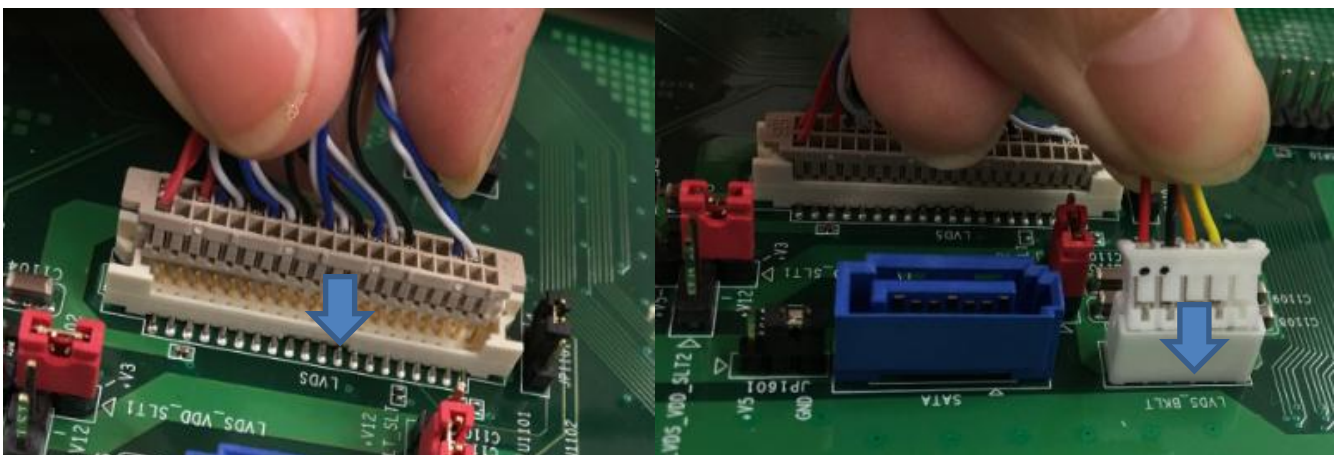


6.3 LVDS Cable

① LVDS Cable Overview



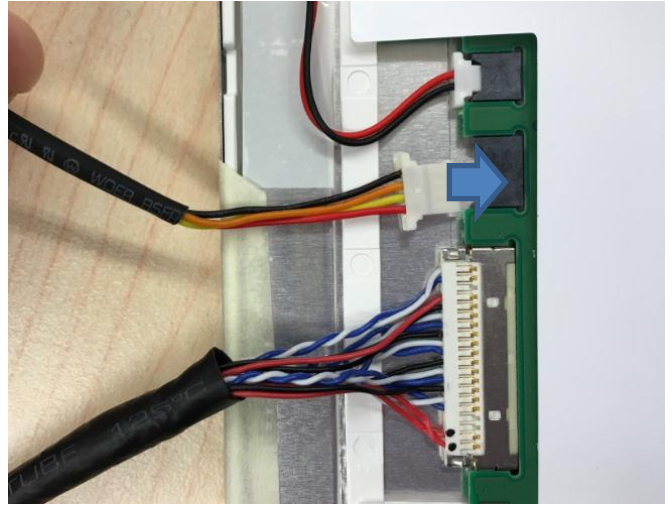
② Connect LVDS cable and Backlight Cable to ROM-DB3900



③ Adjust the Backlight Jumper. Default is 5V but the Panel 96LEDK-A070WV40NB1 is 12V.



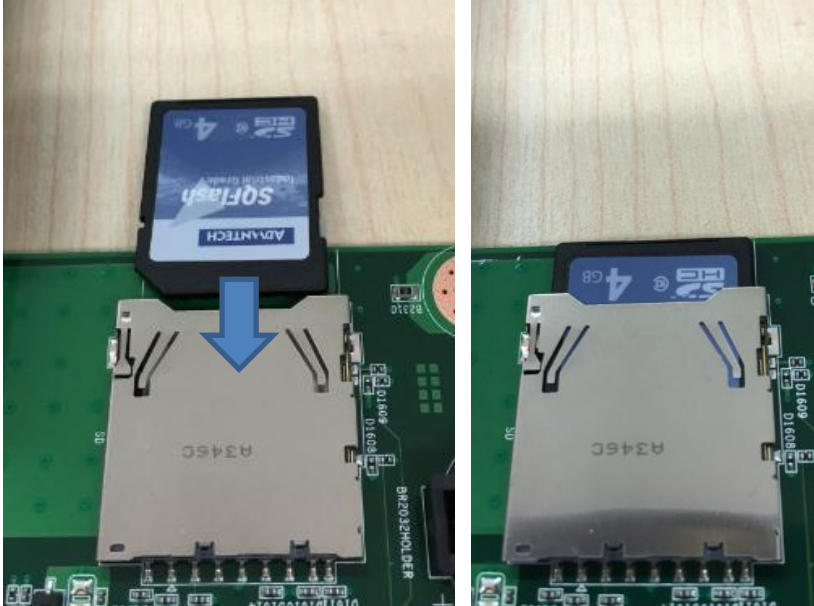
④ Connect to 96LEDK-A070WV40NB1



7. Other Peripheral Assembly

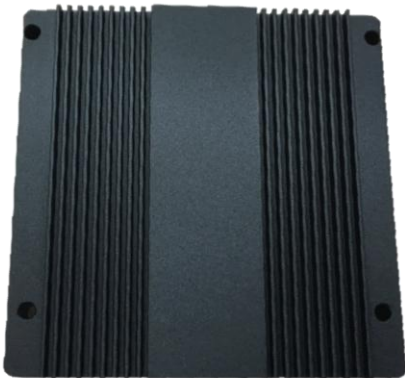
7.1 SD Card insertion

- ① You need to refresh the onboard eMMC image through SD card, please insert SD card to SD slot as indicated in below photos.



7.2 Heat Spreader Assembly

- ① ROM-3420 Heat Spreader



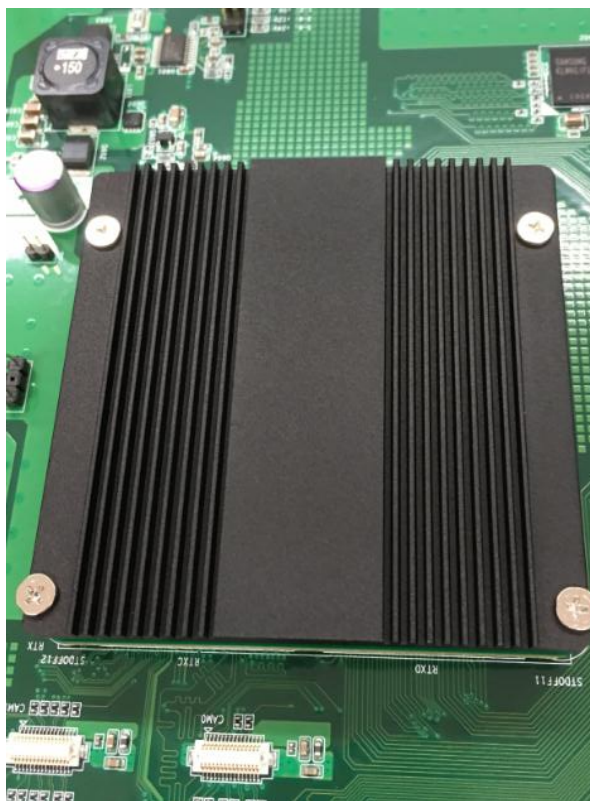
- ② Peel off the protection film in the back



③ Assemble Heat Spreader to ROM-3420

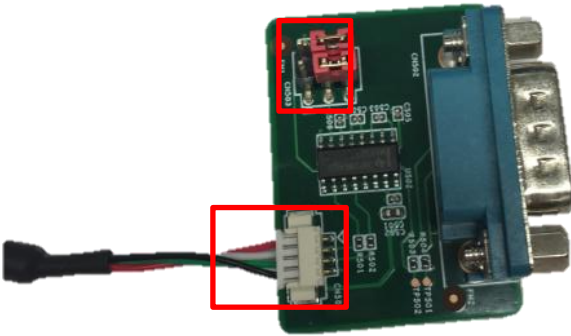


④ Fix Screws



8. Debug Port Connection

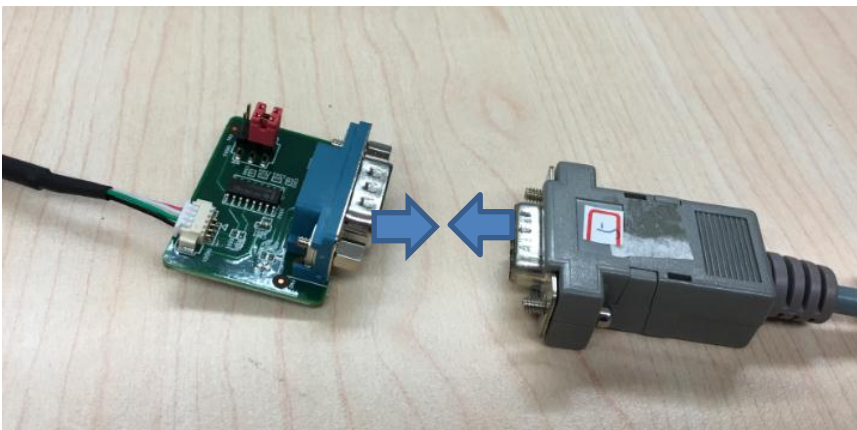
- ① Connect debug port cable to debug port adapter board, make sure the jumpers are set as below:



- ② Connect debug port to ROM-3420



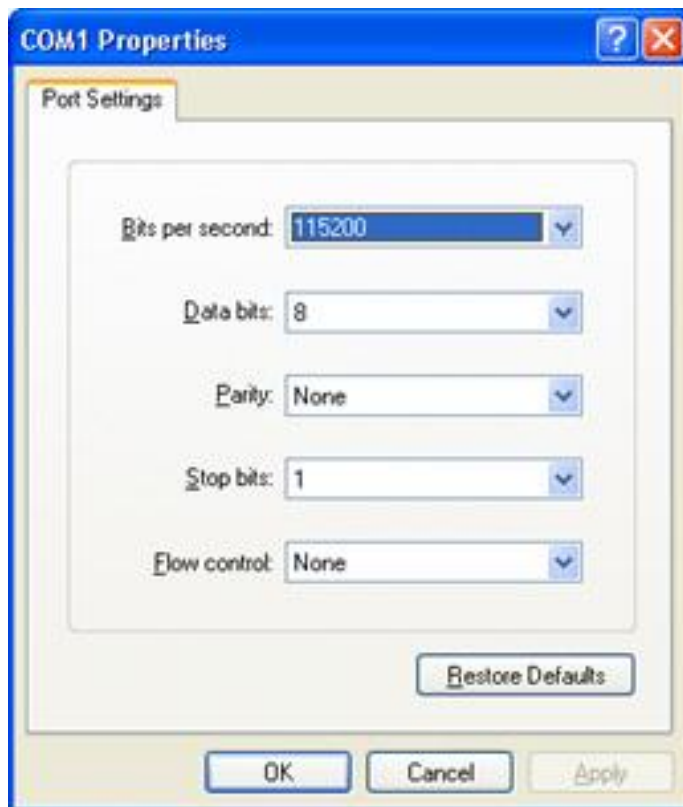
- ③ Connect debug adapter board to a COM port cable



- ④ Connect COM cable to PC



④Set hyper terminal in your PC as below



9. ROM-3420 Jumper/Switch Setting

External IO Connector

Position	Description	
U1101	Flash ROM	A
SW1001	Boot selection	B
CN1201	JTAG connector	C
CN1101	MCU programming port	D
CN1202	Debug port	E

SW1001 (Boot selection)

Jumper	Mode	Jumper	Mode
1-ON 2-OFF	SPI-ROM (Default)	1-OFF 2-ON	SD (Reserved for recovery)

CN1101 (MCU programming port)

Pin	Signal	Pin	Signal
1	+3.3 V	2	MCU_TXD
3	MCU_RXD	4	MCU_PROGRAM#
5	GND		

CN1201 (JTAG connector)

Pin	Signal	Pin	Signal
1	+3.3 V	2	JTAG_TRST#
3	JTAG_TMS	4	JTAG_TDO
5	JTAG_TDI	6	JTAG_TCK
7	-	8	GND
9	-	10	GND

CN1202 (Debug connector)

Pin	Signal	Pin	Signal
1	+3.3 V	2	UART1_TX
3	UART1_RX	4	GND

10. Carrier Board Jumper Setting

Please adjust the jumpers before starting evaluation

JP2201 (GPIO/Keypad selection for GPIO0/10)

Pin	Signal
1-2	GPIO0
2-3	GPIO10



JP2202 (GPIO/Keypad selection for GPIO1/11)

Pin	Signal
1-2	GPIO1
2-3	GPIO11



JP2203 (GPIO/Keypad selection for GPIO2/12)

Pin	Signal
1-2	GPIO2
2-3	GPIO12



JP2204 (GPIO/Keypad selection for GPIO3/13)

Pin	Signal
1-2	GPIO3
2-3	GPIO13



JP2205 (GPIO/Keypad selection for GPIO4/14)

Pin	Signal
1-2	GPIO4
2-3	GPIO14



JP2206 (GPIO/Keypad selection for GPIO5/15)

Pin	Signal
1-2	GPIO5
2-3	GPIO15



JP2207 (GPIO/Keypad selection for GPIO6/16)

Pin	Signal
1-2	GPIO6
2-3	GPIO16

**JP2208 (GPIO/Keypad selection for GPIO7/17)**

Pin	Signal
1-2	GPIO7
2-3	GPIO17

**JP2208 (GPIO8)**

Pin	Signal
1-2	GPIO8
2-3	-

**JP22 (GPIO9)**

Pin	Signal
1-2	GPIO9
2-3	-

**JP1801~JP1804, JP1901~JP1904 (UART 120om terminal resister)**

Pin	Signal
1-2	120 OM (Default)
2-3	Without 120 OM

**JP1101 (EDP_HPD for LVDS)**

Pin	Signal	Pin	Signal
1	GND	2	EDP_HDP_A
3	EDP_HPD		



JP1601 (SATA-DOM Jumper, default 2-3)

Pin	Signal	Pin	Signal
1	+5V	2	SATA-DOM Pin7
3	GND	-	-



JP1701 (CAN0 bus, 120OM terminal resistor)

Jumper	Mode	Jumper	Mode
1-2	120 OM (Default)	-	Without 120 OM



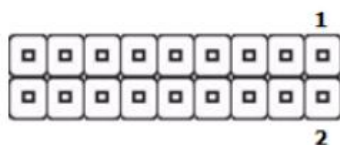
JP1702 (CAN1 bus, 120OM terminal resistor)

Jumper	Mode	Jumper	Mode
1-2	120 OM (Default)	-	Without 120 OM



COM_SEL0/1/2/3 (UART0/1/2/3 function selection)

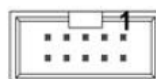
Pin	Signal	Pin	Signal
1	SER2_RX	2	RXD485_A
3	SER2_RX	4	RXD422_A
5	SER2_RX	6	RXD232_A
7	-	8	COM2_RXD
9	ND CD#TXD485-	10	NRXD2TXD485+
11	TXD485-_A	12	TXD485+_A
13	COM2_TXD	14	-
15	NTXD2_RXD485+	16	NDTR#2_RXD485-
17	RSD485+_A	18	RSD485-_A



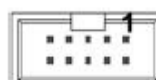
2

(UART1, 2wires)

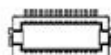
Pin	Signal	Pin	Signal
1	TXD485-	2	-
3	COM1_RX	4	RTS
5	COM1_TX	6	CTS
7	RXD485-	8	-
9	GND	10	-

**(UART3, 2wires)**

Pin	Signal	Pin	Signal
1	TXD485-	2	-
3	RXD/TXD485+	4	RTS
5	TXD/RXD485+	6	CTS
7	RXD485-	8	-
9	GND	10	-

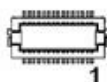
**CAM0 (Camera 1, MIPI)**

Pin	Signal	Pin	Signal
1	PCAM_HSYNC	2	PCAM_ON_CSI1
3	PCAM_VSYNC	4	PCAM_PXL_CK1
5	CAM0_PWR	6	GND
7	CAM0_RST	8	I2C_CSI0_DAT
9	GND	10	I2C_CSI0_CK
11	NC	12	GND
13	NC	14	CSI0_X_CK+
15	GND	16	CSI0_X_CK-
17	NC	18	GND
19	PCAM_MCK	20	CSI0_X_D1+
21	GND	22	CSI0_X_D1-
23	CSI0_MCK	24	GND
25	+3V	26	CSI0_D0+
27	+3V	28	CSI0_X_D0-
29	+3V	30	GND

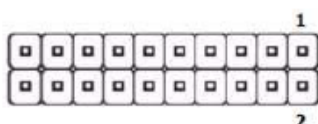


CAM2 (Camera 2, MIPI)

Pin	Signal	Pin	Signal
1	PCAM_FLD	2	PCAM_ON_CSI1
3	PCAM_DE	4	PCAM_PXL_CK0
5	CAM1_PWR	6	GND
7	GND	8	I2C_CSI1_DAT
9	GND	10	I2C_CSI1_CK
11	CSI1_X_D3+	12	GND
13	CSI1_X_D3-	14	CSI1_X_CK+
15	GND	16	CSI1_X_CK-
17	CSI1_X_D2+	18	GND
19	CSI1_X_D2-	20	CSI1_X_D1+
21	GND	22	CSI1_X_D1-
23	CAM1_X_MCK	24	GND
25	+3V	26	CSI1_X_D0+
27	+3V	28	CSI1_X_D0-
29	+3V	30	GND

**GPIO (GPIO)**

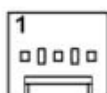
Pin	Signal	Pin	Signal
1	3.3V	11	GPIO8
2	GND	12	GPIO9
3	GPIO0	13	GPIO10
4	GPIO1	14	GPIO11
5	GPIO2	15	GPIO12
6	GPIO3	16	GPIO13
7	GPIO4	17	GPIO14
8	GPIO5	18	GPIO15
9	GPIO6	19	GPIO16
10	GPIO7	20	GPIO17

**JP2301 (External WDT)**

Pin	Signal	Pin	Signal
1	WDT_Time_Out#	2	GND

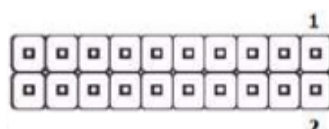
**FAN (System FAN)**

Pin	Signal	Pin	Signal
1	GND	2	+12V
3	+5V		

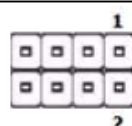


AUDIO (I2S0, audio codec)

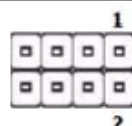
Pin	Signal	Pin	Signal
1	Audio_VDDA	2	GND
3	Audio_VDDA	4	I2S0_SDIN_C
5	I2S0_SDOUT_C	6	I2S0_LRCK_C
7	I2S0_CK_C	8	AUDIO_MCK_C
9	-	10	GND
11	Audio_I2C_CK	12	MIC_BIAS
13	Audio_I2C_DAT	14	GND
15	-	16	HP_R
17	MIC_IN	18	GND
19	GND	20	HP_L

**SPI1 (SPI1)**

Pin	Signal	Pin	Signal
1	+3V	2	GND
3	SPI1_CS0#	3	SPI1_CK
5	SPI1_DO_C	6	SPI1_DIN_C
7	-	8	SPI1_CS1#_C

**SPI0 (SPI0)**

Pin	Signal	Pin	Signal
1	+3V	2	GND
3	SPI0_CS0#	3	SPI0_CK
5	SPI0_DO_C	6	SPI0_DIN_C
7	-	8	SPI0_CS1#_C

**LVDS_BK_SLT (LVDS backlight selection)**

Jumper	Mode	Jumper	Mode
1-2	+5V (Default)	2-3	+12V

* +VDD_BKLT_LVDS, pin1

**LVDS_VDD_SLT1 (LVDS VDD Power selection)**

Jumper	Mode	Jumper	Mode
1-2	+3.3V	2-3	+5V

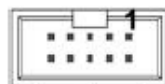
**LVDS_VDD_SLT2 (LVDS VDD Power selection2)**

Jumper	Mode	Jumper	Mode
1-2	+12V		

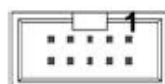


CAN0 (CAN bus 0)

Pin	Signal	Pin	Signal
1	-	2	-
3	CAN0_D-	4	-
5	CAN0_D+	6	-
7	-	8	-
9	GND	10	-

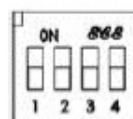
**CAN0 (CAN bus 0)**

Pin	Signal	Pin	Signal
1	-	2	-
3	CAN0_D-	4	-
5	CAN0_D+	6	-
7	-	8	-
9	GND	10	-

**SW2 (Boot selection)**

1	2	3	Feature
ON	ON	ON	Carrier SATA
OFF	ON	ON	Carrier SD
ON	OFF	ON	Carrier eMMC
OFF	OFF	ON	Carrier SPI
ON	ON	OFF	Module device* (reserved)
OFF	ON	OFF	Remote boot (reserved)
ON	OFF	OFF	Module eMMC
OFF	OFF	OFF	Module SPI

*default



SW3 (Reset button)

Pin	Signal	Pin	Signal
1	RESET_IN#	2	GND

SW5 (Sleep button)

Pin	Signal	Pin	Signal
1	SLEEP#	2	GND

SW4 (Power button, CPU)

Pin	Signal	Pin	Signal
1	POWER_BTN#	2	GND

SW6 (LID Switch)

Pin	Signal	Pin	Signal
1	LID#	2	GND



11. Software Functionality

Build Instructions

This section will guide you on how to build the u-boot & Linux kernel.

Build u-boot Image

Advantech has written a script to help build u-boot quickly. You can build a u-boot image by following the steps below:

1. Open "Terminal" on Ubuntu 10.04 LTS
2. `$sudo su` (Change to "root" authority)
3. Input user password
4. Change directory to BSP's scripts folder
5. `#. setenv.sh` (To configure the development environment automatically)
6. `#./cfg_uboot.sh mx6q_rom-3420_1G_config` (To set the u-boot configuration automatically)
7. `#./mk_uboot.sh` (Start to build the u-boot)
8. Then you can see u-boot_crc.bin and u-boot_crc.bin.crc are being built and located in ../image.

Build Linux Kernel Image

Advantech offer you a script to build the "ulmage" quickly. You can build a ulmage by following these steps:

1. Open "Terminal" on Ubuntu 10.04 LTS.
2. `$sudo su` (Change to "root" authority)
3. Input user password.
4. Change directory to BSP's scripts folder
5. `#. setenv.sh` (To configure the development environment automatically)
6. `#./cfg_kernel.sh imx6_rom3420_defconfig` (To set the ulmage configuration automatically)
7. `#./mk_kernel.sh` (Start to build the ulmage)
8. Then you can see ulmage is being built and located in ../image.

Build Log

You can find the build log file from the directory of ROM-3420 BSP. If you got any error messages when building the Linux kernel, it is suggested to look into the log file to find the problem.

Source Code Modification

This section will guide you how to use the Linux source code. You will see some examples of using BSP source code in this section.

Add a Driver to Kernel by menuconfig

You can add a driver to the kernel by menuconfig. Here is an example to guide you how to add a RTC driver (Seiko Instruments S-35390A) to Linux kernel. Please refer to the following steps:

1. Open "Terminal" on Ubuntu 10.04 LTS.
2. `$sudo su` (Change to "root" authority)
3. Input user password.
4. Change directory to BSP's scripts folder
5. `#. setenv.sh` (To configure the development environment automatically)
6. `#./cfg_kernel.sh menuconfig`
7. Then you will see a GUI screen (Linux Kernel Configuration) as below:

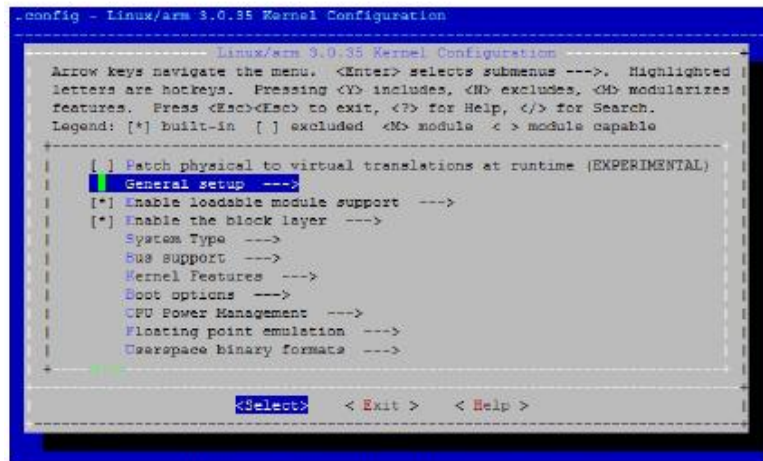


Figure 3.1 Linux Kernel Configuration

8. Select "Device Drivers"→"Real Time Clock", you will see an option "Seiko Instruments S-35390A" on the list. Choose this option then exit and save your configuration.

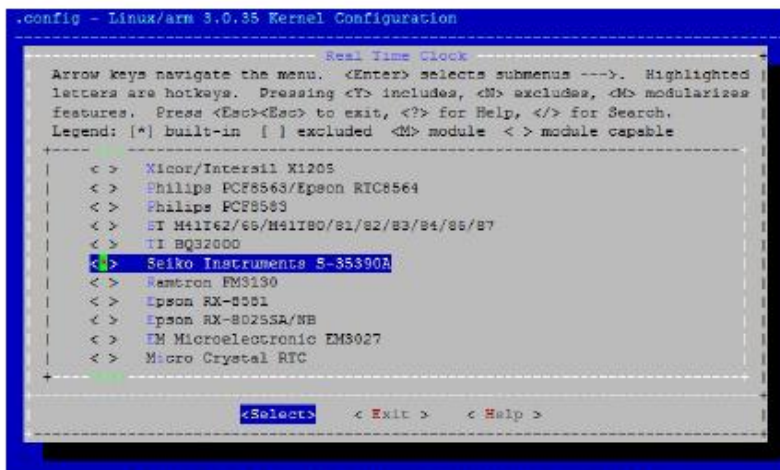


Figure 3.2 Selecting Seiko Instruments S-35390A

9. Change directory to "source/linux-3.0.35/arch/arm/mach-mx6", edit the "board-mx6q_ROM-3420.h" and "board-mx6q_advantech.c". Please add below codes to source/linux-3.0.35/arch/arm/mach-mx6/board-mx6q_ROM-3420.h:

```
static struct i2c_board_info mxc_i2c6_board_info[] __initdata = {
    /* switch 3 */
    { /* rtc - S35390A */
        I2C_BOARD_INFO("s35390a", 0x30),
    },
};
```

Please add below codes to

source/linux-3.0.35/arch/arm/mach-mx6/board-mx6q_advantech.c

```
i2c_register_board_info(6, mxc_i2c6_board_info,
    ARRAY_SIZE(mxc_i2c6_board_info));
```

10. Please refer to former Chapter 3.4.2 Build Linux Kernel Image to rebuild the kernel with RTC driver (Seiko Instruments S-35390A) after completing above steps

Note! If you cannot find the driver for your device from the list, please contact your hardware vendor.



Boot up from the SD card, onboard eMMC or SATA DOM

Boot up from the SD card

Create a bootable SD card

1. Open one terminal console and change directory to BSP scripts folder
2. Perform following command: (assume SD card's device name is /dev/sdf)

```
sudo ./mk-sd-linux.sh /dev/sdf
```
3. Remove SD card from PC/NB

Set the DIP switch

1. Open one debug console
2. Turn off ROM-3420
3. Plug a bootable SD card into ROM-DB3900's SD slot
4. Set Dip switch to 1-OFF/2-ON/3-ON
5. Turn on ROM-3420

Boot up from the onboard eMMC

Transfer whole system to onboard eMMC

1. Boot up from SD card (refer to Chapter 3.6.1)
2. Login as root and perform following commands:

```
$ cd /mk_inand  
$ ./mkinand-linux.sh /dev/mmcblk0
```

Set the DIP switch

1. Open one debug console
2. Turn off ROM-3420
3. Set Dip switch to 1-ON/2-OFF/3-OFF
4. Turn on ROM-3420

Boot up from SATA DOM

Transfer whole system to SATA DOM

1. Turn off ROM-3420
2. Plug SATA DOM into ROM-DB3900's SATA slot
3. Boot up from SD card (refer to Chapter 3.6.1)
4. Login as root and perform the following commands:(assume SATA DOM's device name is /dev/sdf)

```
$ cd /mk_inand  
$ ./mkinand-linux.sh /dev/sdf
```

Set the DIP switch

1. Open one debug console
2. Turn off ROM-3420
3. Set Dip switch to 1-ON/2-ON/3-ON
4. Turn on ROM-3420

For more SW/Tool, please refer to:

http://download.advantech.com/ProductFile/Downloadfile1/1-Y0QMII/ROM-3420_User_Manual_Ed.2-FINAL.pdf