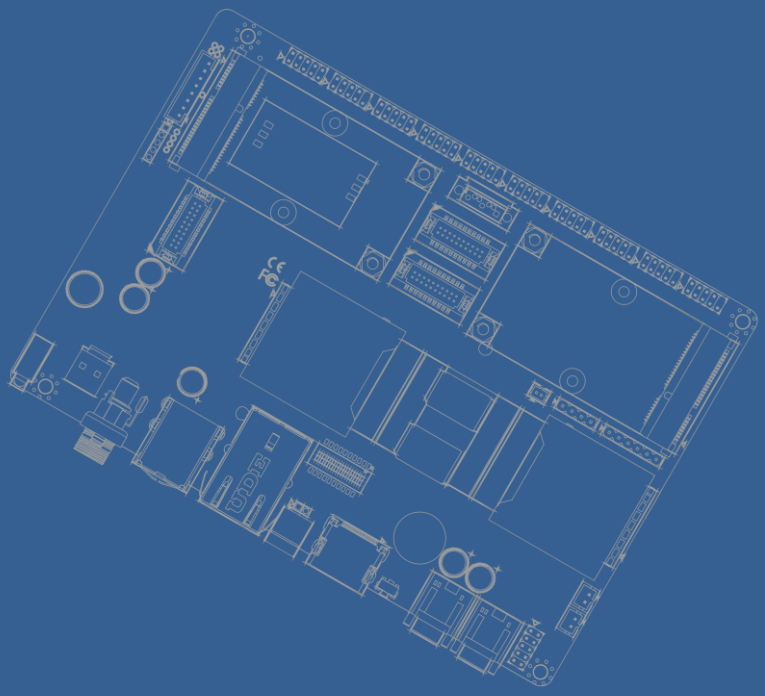
User Manual
Ver. 5st



EBC-3A1 Series

3.5” SBC with NXP i.MX6 Processor
ARM ® Cortex A9 Architecture

Copyright

Trademark

Disclaimer

Information in this document is subject to change without notice and does not represent a commitment on the part of NEXCOM.

NEXCOM provides this document as is, without warranty of any kind, either expressed or implied, including, but not limited to, its particular purpose. NEXCOM reserves the right to make improvements and/or changes to this manual, or to the products and/or the programs described in this manual, at any time.

Information provided in this manual is intended to be accurate and reliable. However, NEXCOM International Co., Ltd. assumes no responsibility for its use, nor for any infringements of the rights of third parties, which may result from its use.

This product might include unintentional technical or typographical errors. Changes are periodically made to the information herein to correct such errors, and these changes are incorporated into new editions of the publication.

Declaration of Conformity

FCC Class A

Note: this device has been tested and found to comply with the limits for a Class A digital device, pursuant to part 15 of the FCC Rules. These limits are designed to provide reasonable protection against harmful interference in a residential installation. This device generates, uses and can radiate radio frequency energy and, if not installed and used in accordance with the instructions, may cause harmful interference to radio communication. However, there is no guarantee that interference will not occur in a particular in a particular installation. If this device does cause harmful interference to radio or television reception, which can be determined by turning the device off and on, the user is encouraged to try to correct the interference by one or more of following measures:

- Reorient or relocate the receiving antenna
- Increase the separation between the device and receiver
- Connect the device into an outlet on a circuit different from that to which receiver is connected
- Consult the dealer or an experienced radio/TV technician for help

CE Marking

This device has passed the CE test for environmental specifications when shielded cables are used for external wiring. We recommend the use of shielded cables.
This device has passed the CE test for environmental specifications. Test conditions for passing included the equipment being operated within an industrial enclosure. In order to protect the product from being damaged by ESD (Electrostatic Discharge) and EMI leakage, we strongly recommend the use of CE-compliant industrial enclosure products.

Document Amendment History

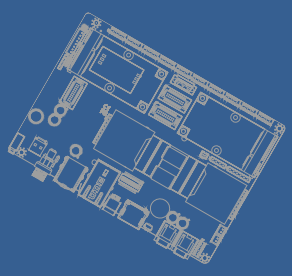
Revision	Date	Remark
1st	Jan 2016	Initial released
2nd	Mar 2016	Hardware User Guide modified
3rd	Mar 2016	Software User Guide & System Recovery modified
4rd	Aug 2018	Tidy-up full document and add new sections.
5rd	Aug 2019	Modify JCOMA to JCOMD to /dev/ttyXRUSB0 ~ /dev/ttyXRUSB3

Table of Contents

Chapter 1

Product Overview

This chapter provides background information of EBC-3A1.



1. Product Overview

1.1. Introduction

EBC-3A1 is a 3.5'' SBC (Single Board Computer) with ARM Cortex-A9 NXP i.MX6 Dual Lite 1GHz processor and ARM Cortex™-M0 32-bit RISC core (MCU). The EBC-3A1 supports 2GB DDR3, 8MB NOR Flash and 4GB eMMC NAND Flash, 2 x LVDS, 1 x HDMI display, 1 x Gigabit LAN with IEEE 1588, 4 x USB 2.0, 1 x USB OTG, 4 x RS-232(8-wire), 2 x RS-232(4 wire), 2 x RS-485, 2 x CAN, 1 x SDIO and 1 x SD.

Integrated unique dual hardware structure and RTOS (Real Time Operating System) design, EBC-3A1 has outstanding crash free protection on both hardware reliability and software stability. With the special features, EBC-3A1 is a perfect device to meet customers' versatile needs.

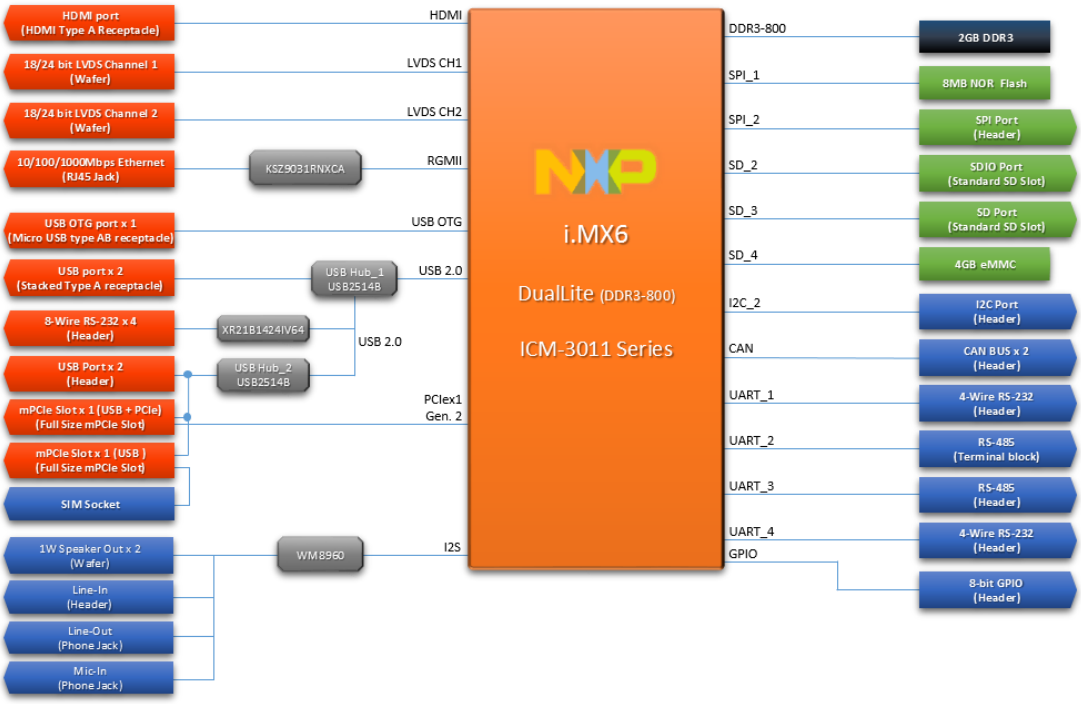
The EBC-3A1 focuses on industrial application and it provides high performance and low power consumption from its ARM ® Cortex A9 architecture which is ready-to-run, compact, and easy-to-expand. With flexible I/O interfaces and complete hardware and software solutions, EBC-3A1 is a fast time-to-market platform for customers to develop their applications and products easily.

1.2. Specification

System Hardware - CPU		
CPU		NXP i.MX6 Cortex-A9 Dual Lite
Memory	Technology	DDR3-800
	Capacity	Onboard 1GB
Storage	Flash	4GB eMMC NAND Flash 8MB NOR Flash
Graphic	HDMI	1 x HDMI connector
	LVDS	2 x 18/24 bits LVDS header (2x10 1.25mm Hirose DF13 series compatible)
Watchdog Timer		1~256 level (0.5 second / level)
RTC		EPSON RX8010SJ RTC chip
Indicator	LED	1 x configurable indicator controlled by i.MX6
I/O	LAN	1 x Microchip KSZ9031RNX Gigabit Ethernet
	USB	4 x USB 2.0
	USB OTG	1 x USB OTG (Micro USB Type AB connector)
	Serial Port	4 x RS-232 (8-wire) header 2 x RS-232 (4-wire) header 1 x RS-485 (3.5mm terminal block) 1 x RS-485 (miscellaneous header)
	CAN	2 x CAN (miscellaneous header)
	GPIO	8 x GPIO header

	Button	1 x power-on header 1 x reset button
	SD socket	1 x SD socket
	SDIO socket	1 x SDIO socket
System Hardware – MCU (For NEXCOM system product design only)		
MCU		STM32F051R8T6
Storage	Flash	8MB NOR Flash
I/O	NEXCOM X Port	1 x NEXCOM X Port (miscellaneous header-JMISC)
	I2C	1 x I2C interface (2x10 1.25mm Hirose DF13 series compatible, shared with SPI and GPIO) for Mainboard MCU to accessory (i.e. OLED module) connection purpose
	SPI	1 x SPI interface (2x10 1.25mm Hirose DF13 series compatible, shared with I2C and GPIO) for Mainboard MCU to accessory (i.e. OLED module) connection purpose
	GPIO	7-bit GPIO (2x10 1.25mm Hirose DF13 series compatible, shared with SPI and I2C) for Mainboard MCU to accessory (i.e. OLED module) connection purpose 2-bit GPIO (5x2 header, pitch 2.0mm-JFP)
System Software		
Operating System	Yocto	Daisy 1.6.2
	Android	Lollipop 5.0.2
	WinCE	Windows Embedded Compact 7
Environment & Mechanism		
Temperature	Operating temperature	-20~70° C
Humidity	Operating humidity	5%~95% Relative Humidity, non-condensing
Mechanism	Dimension	3.5" SBC (146mm X 102mm)
Power	DC-input	12V (10.8V~26.4V)
	Control	Power on by DC attached or via power button
	Consumption	~3.5W

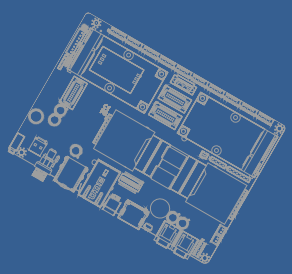
1.3. Block Diagram



Chapter 2

Hardware User Guide

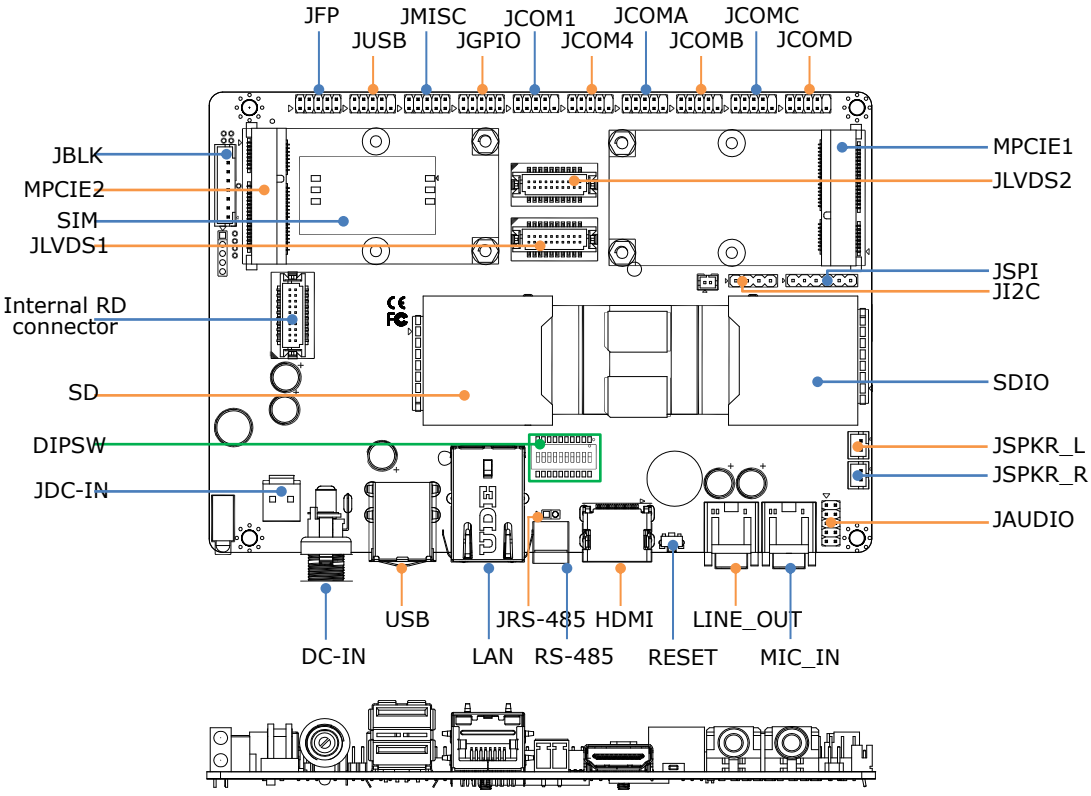
This chapter introduces the startup procedures of EBC-3A1, device integration. It also introduces the setting of switches, indicators and shows the mechanical drawings. Be sure to read all safety precautions before you begin installation procedure.



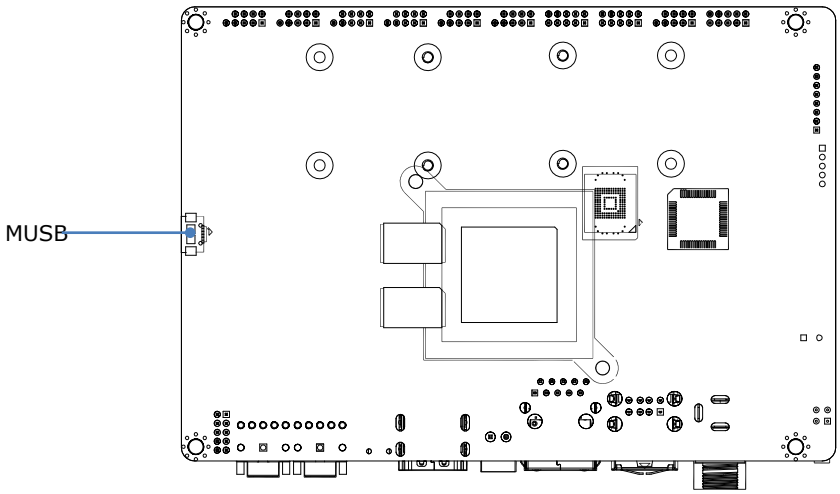
2. Hardware User Guide

2.1. DIP Switch Setting and Connector Locations

Top side:



Rear side:



2.2. DIP Switch Setting

The EBC-3A1 has a 10-bits DIP switches for configuration. Please make sure the switches set as the following settings.

2.2.1. Boot from eMMC (Default)

DIPSW	1	2	3	4	5	6	7	8	9	10
eMMC	ON	OFF	OFF	ON	ON	OFF	OFF	ON	ON	ON

Table 1: DIPSW (default setting)

2.2.2. Boot from SD

DIPSW	1	2	3	4	5	6	7	8	9	10
SD	ON	OFF	OFF	ON	OFF	OFF	OFF	ON	ON	OFF

Table 2: DIPSW (boot from SD)

2.2.3. Boot for Serial ROM (Upgrade Image)

DIPSW	1	2	3	4	5	6	7	8	9	10
PG	OFF	ON	X	X	X	X	X	X	X	X

Table 3: DIPSW (upgrade image)

2.3. Connector

2.3.1. Connector List

DC-IN	DC power input connector	ψ2.5mm DC Jack
HDMI	HDMI connector	
JAUDIO	Audio pin header	5 x 2 header, pitch 2.0mm
JBLK	LVDS backlight connector	8 x 1 connector, pitch 2.0mm
JCOM1	RS-232 pin header	5 x 2 header, pitch 2.0mm
JCOM4	RS-232 pin header	5 x 2 header, pitch 2.0mm
JCOMA	RS-232 pin header	5 x 2 header, pitch 2.0mm
JCOMB	RS-232 pin header	5 x 2 header, pitch 2.0mm
JCOMC	RS-232 pin header	5 x 2 header, pitch 2.0mm
JCOMD	RS-232 pin header	5 x 2 header, pitch 2.0mm
JDC-IN	DC power input connector	2 x 1 connector, pitch 3.96mm
JNEXCOM	Proprietary connector	DF13-20DP-1-25V
JFP	Front panel connector	5 x 2 header, pitch 2.0mm
JGPIO	GPIO pin header	5 x 2 header, pitch 2.0mm
JI2C	I2C connector	4 x 1 header, pitch 2.54mm
JLVDS1	LVDS channel 1 connector	DF13-20DP-1-25V
JLVDS2	LVDS channel 2 connector	DF13-20DP-1-25V
JMISC	Miscellaneous connector	5 x 2 header, pitch 2.0mm
JRS-485	RS-485 pin header	2 x 1 header, pitch 2.0mm
JSPI	SPI pin header	6 x 1 header, pitch 2.54mm
JSPKR_L	Speaker Out connector	2 x 1 connector, pitch 2.0mm
JSPKR_R	Speaker Out connector	2 x 1 connector, pitch 2.0mm
JUSB	USB pin header	5 x 2 header, pitch 2.0mm
LAN	Ethernet connector	RJ-45
LINE_OUT	Line-Out connector	ψ3.5mm Phone Jack
MIC_IN	Mic-In connector	ψ3.5mm Phone Jack
MPCIE1	Both PCI Express and USB 2.0 connectivity	
MPCIE2	USB 2.0 connectivity only	
MUSB	USB OTG	Micro USB Type AB Receptacle
RS-485	RS-485 connector	2 x 1 Terminal Block, pitch 3.5mm
SD	SD card socket	
SDIO	SDIO card socket	
SIM	SIM card socket	Mini-SIM socket
USB	USB connector	Stacked Type A Receptacle

2.3.2. Connector Setting

2.3.2.1.1. JAUDIO

Description	Pin	Pin	Description
LINE_IN_R	1	2	LINE_IN_L
N/C	3	4	MIC_DETECT
MIC_IN	5	6	MIC_BIAS
LINE_OUT_R	7	8	LINE_OUT_L
GNDA	9	10	LINE_OUT_DETECT

2.3.2.1.2. JBLK

Description	Pin
VIN	1
VIN	2
VIN	3
Backlight EN (3.3V)	4
Backlight CTL (3.3V)	5
GND	6
GND	7
GND	8

Remark:

Match connector: JST PHR-08 P2.0 (or Equivalent)

VIN pins connect to DC-IN power input directly. For example, if DC-In is attached to a 12V adaptor, then the VIN output voltage will be 12V.

2.3.2.1.3. JCOM1

Description	Pin	Pin	Description
N/C	1	2	RS-232_RX1
RS-232_TX1	3	4	N/C
GND	5	6	N/C
RS-232_RTS1	7	8	RS-232_CTS1
N/C	9	10	N/C

2.3.2.1.4. JCOM4

Description	Pin	Pin	Description
N/C	1	2	RS-232_RX4
RS-232_TX4	3	4	N/C
GND	5	6	N/C
RS-232_RTS4	7	8	RS-232_CTS4
N/C	9	10	N/C

2.3.2.1.5. JCOMA

Description	Pin	Pin	Description
RS-232_DCDA	1	2	RS-232_RXA
RS-232_TXA	3	4	RS-232_DTRA
GND	5	6	RS-232_DSRA
RS-232_RTSA	7	8	RS-232_CTSA
RS-232_RIA	9	10	N/C

2.3.2.1.6. JCOMB

Description	Pin	Pin	Description
RS-232_DCDB	1	2	RS-232_RXB
RS-232_TXB	3	4	RS-232_DTRB
GND	5	6	RS-232_DSRB
RS-232_RTSB	7	8	RS-232_CTSB
RS-232_RIB	9	10	N/C

2.3.2.1.7. JCOMC

Description	Pin	Pin	Description
RS-232_DCDC	1	2	RS-232_RXC
RS-232_TXC	3	4	RS-232_DTRC
GND	5	6	RS-232_DSRC
RS-232_RTSC	7	8	RS-232_CTSC
RS-232_RIC	9	10	N/C

2.3.2.1.8. JCOMD

Description	Pin	Pin	Description
RS-232_DCDD	1	2	RS-232_RXD
RS-232_TXD	3	4	RS-232_DTRD
GND	5	6	RS-232_DSRD
RS-232_RTSD	7	8	RS-232_CTSD
RS-232_RID	9	10	N/C

2.3.2.1.9. JDC-IN

Description	Pin
GND	1
VIN	2

2.3.2.1.10. JNEXCOM (MCU)

Description	Pin	Pin	Description
3.3V	2	1	5V
3.3V	4	3	5V
MCU_GPIO0	6	5	GND
MCU_GPIO1	8	7	MCU_SS
MCU_GPIO2	10	9	MCU_CLK
MCU_GPIO3	12	11	MCU_MOSI
MCU_GPIO4	14	13	MCU_MISO
MCU_GPIO5	16	15	GND
MCU_GPIO6	18	17	MCU_I2C CLK
N/C	20	19	MCU_I2C DAT

Remark:

Match connector: DF13-20DS-1.25C

GPIO pin control by MCU (level 3.3V)

MCU_SS: SPI Slave Select (active low, output from master)

MCU_CLK: SPI Serial Clock (output from master)

MCU_MOSI: SPI Master Output, Slave Input (output from master)

MCU_MISO: Master Input, Slave Output (output from slave)

MCU is for NEXCOM system product design only

2.3.2.1.11. JFP

Description	Pin	Pin	Description
5V	1	2	LED control by MCU
5V	3	4	LED control by CPU
CPU power on	5	6	GND
MCU_GPIO7	7	8	GND
MCU_GPIO8	9	10	GND

Remark:

GPIO pin control by MCU (level 3.3V)

2.3.2.1.12. JGPIO (i.MX6)

Description	Pin	Pin	Description
GPIO_00	1	2	GPIO_04
GPIO_01	3	4	GPIO_05
GPIO_02	5	6	GPIO_06
GPIO_03	7	8	GPIO_07
GND	9	10	GND

2.3.2.1.13. JI2C

Description	Pin
3.3V	1
I2C_SCL	2
I2C_SDA	3
GND	4

2.3.2.1.14. JLVDS1

Description	Pin	Pin	Description
3.3V	2	1	5V
3.3V	4	3	5V
Backlight EN (3.3V)	6	5	LVDS0_CLK-
Backlight CTL (3.3V)	8	7	LVDS0_CLK+
GND	10	9	GND
LVDS0_TX1-	12	11	LVDS0_TX0-
LVDS0_TX1+	14	13	LVDS0_TX0+
GND	16	15	GND
LVDS0_TX3-	18	17	LVDS0_TX2-
LVDS0_TX3+	20	19	LVDS0_TX2+

Match connector: DF13-20DS-1.25C

2.3.2.1.15. JLVDS2

Description	Pin	Pin	Description
3.3V	2	1	5V
3.3V	4	3	5V
Backlight EN (3.3V)	6	5	LVDS1_CLK-
Backlight CTL (3.3V)	8	7	LVDS1_CLK+
GND	10	9	GND
LVDS1_TX1-	12	11	LVDS1_TX0-
LVDS1_TX1+	14	13	LVDS1_TX0+
GND	16	15	GND
LVDS1_TX3-	18	17	LVDS1_TX2-
LVDS1_TX3+	20	19	LVDS1_TX2+

Match connector: DF13-20DS-1.25C

2.3.2.1.16. JMISC

Description	Pin	Pin	Description
NEXCOM X-Port RS-485 DATA-	1	2	RS-485 DATA3-
NEXCOM X-Port RS-485 DATA+	3	4	RS-485 DATA3+
GND	5	6	N/A
CAN_1_H	7	8	CAN_2_H
CAN_1_L	9	10	CAN_2_L

2.3.2.1.17. JRS-485

Description	Pin
RS-485 DATA2-	1
RS-485 DATA2+	2

2.3.2.1.18. JSPI

Description	Pin
3.3V	1
SPI2_MOSI	2
SPI2_MISO	3
SPI2_CS0	4
SPI2_CLK	5
GND	6

2.3.2.1.19. JSPKR_L

Description	Pin
SPKR_L+	1
SPKR_L-	2

Match connector: JST PHR-02 P2.0 (or Equivalent)

2.3.2.1.20. JSPKR_R

Description	Pin
SPKR_R+	1
SPKR_R-	2

Match connector: JST PHR-02 P2.0 (or Equivalent)

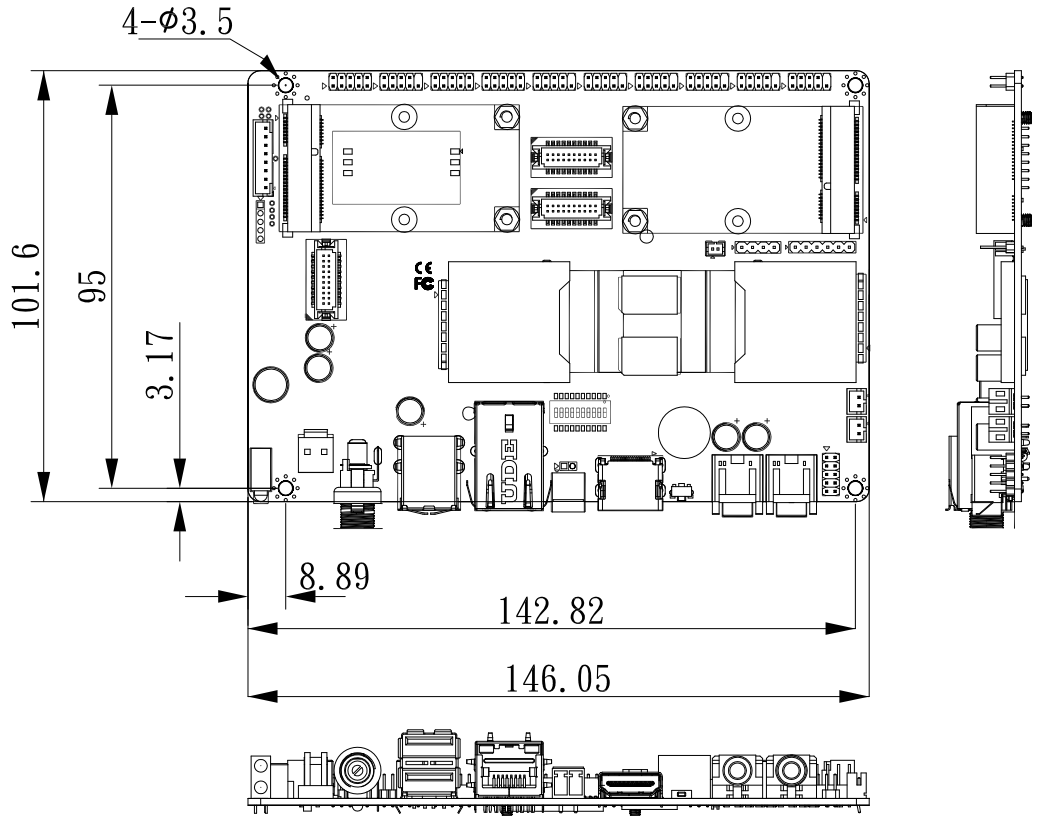
2.3.2.1.21. JUSB

Description	Pin	Pin	Description
USB_5V	1	2	USB_5V
USB_DN8	3	4	USB_DN7
USB_DP8	5	6	USB_DP7
GND	7	8	GND
GND	9	10	N/C

2.3.2.1.22. RS-485

Description	Pin
RS-485 DATA2-	1
RS-485 DATA2+	2

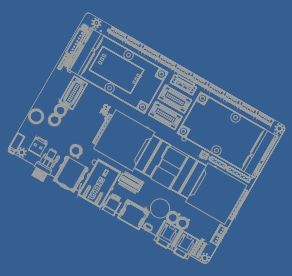
2.4. Mechanical Drawing



Chapter 3

Software User Guide

This chapter details the Linux operation on EBC-3A1.



3.Software User Guide

3.1. Introduction

This chapter details the EBC-3A1 platform. The platform is an embedded system with Linux kernel 3.10.53. It contains all system-required shell commands and drivers ready. User can develop under Linux environment. Such as Ubuntu, Debian, Fedora…etc.

The purpose of this chapter is to introduce software development of EBC-3A1 and improve software development time and efficiency.

3.2. Development Environment

3.2.1. How to Install Toolchain

1. Please Download the toolchain from Link.
2. Create a /opt/toolchain folder under your Linux environment.

```
embux@ubuntu:~$ mkdir -p /opt/toolchian
```

3. Unzip gcc-linaro-arm-4.7-linux.tar.gz into /opt/toolchain

```
embux@ubuntu:/opt/toolchian$ ls
gcc-linaro-arm-4.7-linux.tar.gz
embux@ubuntu:/opt/toolchian$ tar zxf gcc-linaro-arm-4.7-linux.tar.gz
embux@ubuntu:/opt/toolchian$ ls
gcc-linaro-arm-4.7-linux.tar.gz
gcc-linaro-arm-linux-gnueabihf-4.7-2013.04-20130415_linux
```

4. Add toolchain path in your $PATH environment variable.

```
embux@ubuntu:~$ export PATH=$PATH:/opt/toolchian/gcc-linaro-arm-
linux-gnueabihf-4.7-2013.04-20130415_linux/bin/
```

5. Try to compile Embux example code.

3.2.2. RS232 Debug Console

3.2.2.1.1. Debug Console Information

The serial communication parameters are **115200, N81, VT100**. Use your preferred serial terminal tools to access the RS232 debug console.

RS232 Utility recommend:

- ➔ On Windows system, use **putty** or **teraterm**.
- ➔ On Linux/OSX system, use **minicom** utility.

3.2.2.1.2. Debug Console Device Node

RS232 Port	Device node	Pinout Define Section
JCOM1	/dev/ttymxc0	2.3.2.1.3

3.2.2.1.3. Boot Message

```
Starting Telephony daemon
Starting GPS (Global Positioning System) daemon gpsd
Starting php-fpm  done
Starting Linux NFC daemon
Starting web server: apache2.
Running eGTouchD
Starting OProfileUI server
Stopping Bootlog daemon: bootlogd.

Poky (Yocto Project Reference Distro) 1.6.2 ICM-3011 /dev/ttymxc0

ICM-3011 login: root
root@ICM-3011:~#
```

Note: default user name is "root" without password.

3.2.3. Networking Settings

The EBC-3A1 has one Ethernet port, the default network setting is following:

Device Node	IP mode	IP Address
Eth0	static	192.168.1.1

The network interface configuration file path is **/etc/network/interfaces** file which is standard Linux configure file, and user can get more information from Internet. Please edit to change network setting.

```
root@ICM-3011:~# cat /etc/network/interfaces
# /etc/network/interfaces -- configuration file for ifup(8), ifdown(8)

# The loopback interface
auto lo
iface lo inet loopback

# Wired or wireless interfaces
auto eth0
# iface eth0 inet dhcp
iface eth0 inet static
       address 192.168.1.1
       netmask 255.255.255.0

# Wireless interfaces
auto wlan0
iface wlan0 inet dhcp
       wireless_mode managed
       wireless_essid any
       wpa-driver nl80211
       wpa-conf /etc/wpa_supplicant.conf
```

Note: Embux provided WIFI solution, please contact us to get more information.

3.2.4. Firmware Version

3.2.4.1.1. Embux Image Version

"version" command is developed by Embux. When user gets some problem, please send Embux version to the contact.

```
root@ICM-3011:~# version
Firmware Version: v1.0.90 (r1109)
```

3.2.4.1.2. Linux Kernel Image Version

"uname" command to get EBC-3011 Linux Kernel version.

```
root@ICM-3011:~# uname -a
Linux ICM-3011 3.10.53 #1109 SMP PREEMPT Tue Jul 31 23:36:30 PDT 2018
armv7l GNU/Linux
```

3.2.5. Editor Utility

3.2.5.1.1. VI Editor

Vi is a very old command-line editor, which is available on most UNIX systems.

```
root@ICM-3011:/# ls -l /bin/vi
lrwxrwxrwx 1 root root 19 Aug  1 06:40 /bin/vi -> /bin/busybox.nosuid
```

3.2.5.1.2. NANO Editor

GNU Nano is at the easy-to-use end of command-line editors.

```
root@ICM-3011:/# ls -l /usr/bin/nano
-rwxr-xr-x 1 root root 162228 Jul 26 09:49 /usr/bin/nano
```

3.2.6. eMMC Default Partitions

Device Node	Size	Format	Discuses
mmcblk0p1	3GB	Ext4	Main system. (Sync all data and metadata every 5 seconds) PATH: /
mmcblk0p2	32MB	Ext4	User Space. PATH: /media/mmcblk0p2
mmcblk0p3	1 byte		N/A
mmcblk0p5	1KB		N/A
mmcblk0p6	1KB		N/A
mmcblk0p7	1KB		N/A
mmcblk0p8	1KB		N/A
mmcblk0p9	512MB	VFAT	User Space. PATH: /media/mmcblk0p9
mmcblk0p10	90MB	VFAT	Boot Images: Kernel, DTB, Boot-Env PATH: /boot or /media/mmcblk0p10

Note:

1. After modifying any files in main system, please execute "sync" command.
2. Highly recommend that run-time data (database, logger, and so on) works at mmcblk0p2 and mmcblk0p9 partitions. Avoid unsafe power off the machine and then make the main system crash.

3.3. Interface Introduce

3.3.1. RS232

3.3.1.1.1. RS232 Interface

4 RS232 serial ports for user interface control, and all RS232 ports are DTE mode.

The pinout direction please refer below table:

Name	Abbreviation	Direction
Transmitted Data	TXD	Output
Received Data	RXD	Input
Clear to Send	CTS	Input
Request to Send	RTS	Output
Data Set Ready	DSR	Input
Data Terminal Ready	DTR	Output
Data Carrier Detect	DCD	Input
Ring Indicator	RI	Input
Common Ground	GND	Common

3.3.1.1.2. RS232 Specification

Name	Value
Baud Rate	Up to 500Kbps
Data Bits	5, 6, 7, 8 bits
Parity	None, Even, Odd, Mark, Space
Stop Bits	1, 1.5, 2 bits

3.3.1.1.3. RS232 Device Node

Connector	Device node	Pinout Define Section
JCOM4	/dev/ttymxc3	2.3.2.1.4
JCOMA	/dev/ttyXRUSB0	2.3.2.1.5
JCOMB	/dev/ttyXRUSB1	2.3.2.1.6
JCOMC	/dev/ttyXRUSB2	2.3.2.1.7
JCOMD	/dev/ttyXRUSB3	2.3.2.1.8

3.3.1.1.4. RS232 Sample Code

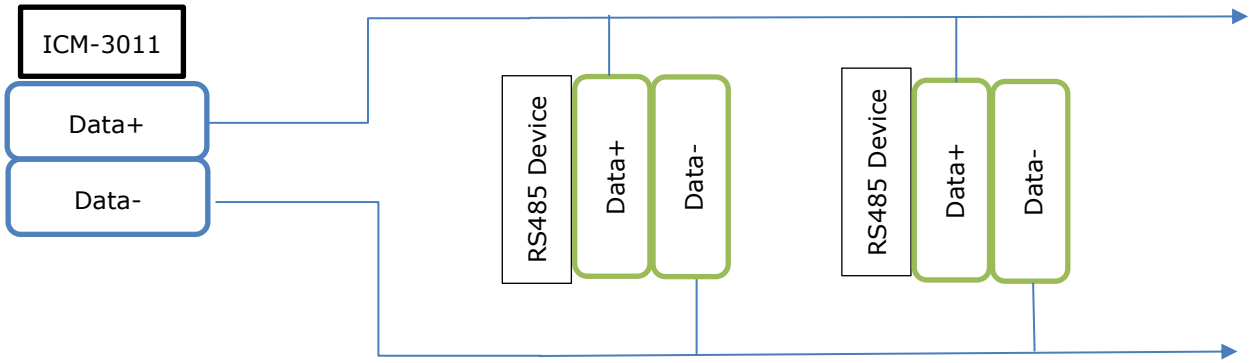
http://github.com/embux/Example/tree/master/serial_test

3.3.2. RS485

3.3.2.1.1. RS485 Information

2 RS485 serial ports for user interface control, and Embux provided 2-line RS485 interface. (Data+ and Date-).

The connection diagram please refer below (daisy chain):



3.3.2.1.2. RS485 Specification

Name	Value
Baud Rate	Up to 500Kbps
Data Bits	5, 6, 7, 8 bits
Parity	None, Even, Odd, Mark, Space
Stop Bits	1, 1.5, 2 bits

3.3.2.1.3. RS485 Device Node

Connector	Device node	Pinout Define Section
RS-485 (JRS-485)	/dev/ttymxc1	2.3.2.1.22
JMISC	/dev/ttymxc2	2.3.2.1.16

3.3.2.1.4. RS485 Sample Code

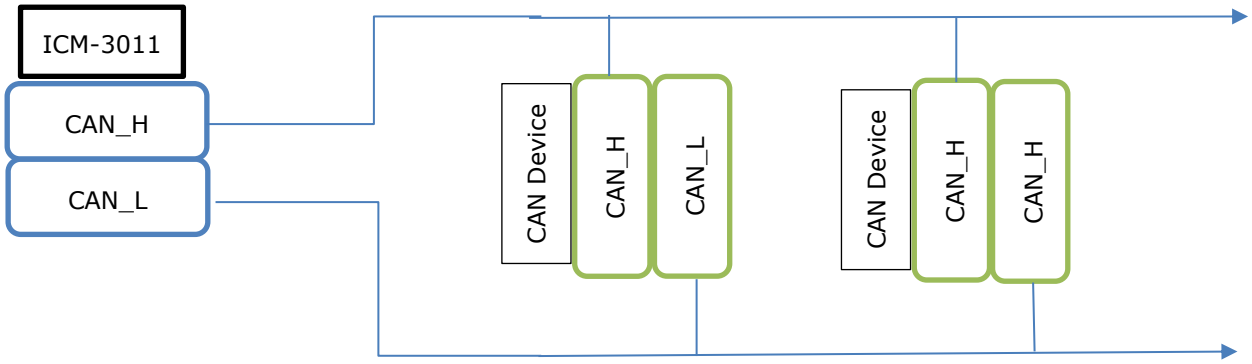
http://github.com/embux/Example/tree/master/J485_test

3.3.3. CAN bus (Controller Area Network)

3.3.3.1.1. CAN bus Information

2 CAN bus ports for user interface control. The current CAN bus driver is based on FlexCan which is an embedded network architecture that extends Controller Area Network (CAN). It provides more deterministic behavior over the CAN network. Its focus is on redundancy at the hardware level, and time-based prioritized communication at the protocol level.

The connection diagram please refer below (daisy chain):



3.3.3.1.2. CAN bus Specification

Name	Value
Bit Rate	up to 1 M Baud
Data Length	0 ~ 8 bytes

3.3.3.1.3. CAN bus Device Node

Connector	Device node	Pinout Define Section
JMISC (CAN1)	can0	2.3.2.1.16
JMISC (CAN2)	can1	2.3.2.1.16

3.3.3.1.4. CAN bus Sample Code

http://github.com/embux/Example/tree/master/can_test

3.3.4. GPIO

3.3.4.1.1. GPIO Information

8 GPIO ports for user interface controls. GPIO signals have paths like /sys/class/gpio/gpioN/ and have the following read/write attributes:

"**direction**" ... reads as either "in" or "out". This value may normally be written. Writing as "out" defaults to initializing the value as low.

"**value**" ... reads as either 0 (low) or 1 (high). If the GPIO is configured as an output, this value may be written; any nonzero value is treated as high, otherwise as low.

3.3.4.1.2. GPIO Specification

Name	Value
Voltage Level	0 ~ 3.3v

3.3.4.1.3. GPIO Device Node

Connector	Device node	Pinout Define Section
GPIO_00 (JGPIO)	/sys/class/gpio/gpio0	2.3.2.1.12
GPIO_01 (JGPIO)	/sys/class/gpio/gpio1	2.3.2.1.12
GPIO_02 (JGPIO)	/sys/class/gpio/gpio2	2.3.2.1.12
GPIO_03 (JGPIO)	/sys/class/gpio/gpio3	2.3.2.1.12
GPIO_04 (JGPIO)	/sys/class/gpio/gpio4	2.3.2.1.12
GPIO_05 (JGPIO)	/sys/class/gpio/gpio5	2.3.2.1.12
GPIO_06 (JGPIO)	/sys/class/gpio/gpio6	2.3.2.1.12
GPIO_07 (JGPIO)	/sys/class/gpio/gpio7	2.3.2.1.12

3.3.4.1.4. GPIO Example

Example Code:

http://github.com/embux/Example/tree/master/gpio_test

Example Commands:

#1: GPIO_00 port sets to **output** and **high voltage**.

```
root@ICM-3011:~# echo "out" > /sys/class/gpio/gpio00/direction
root@ICM-3011:~# echo "1" > /sys/class/gpio/gpio00/value
```

#2: GPIO_01 port sets to **input**.

```
root@ICM-3011:~# echo "in" > /sys/class/gpio/gpio01/direction
root@ICM-3011:~# cat /sys/class/gpio/gpio01/value
1
```

Note: All GPIO ports are floating. If user would like to use input mode, please add external pull-up or pull-down voltage.

3.3.5. RTC & Watchdog

3.3.5.1.1. RTC & Watchdog Information

A real-time clock (RTC) keeps track of the current time, and ICM-3011 will synchronize hardware clock to system clock when boot.

Watchdog timer, during normal operation, the computer regularly resets the watchdog timer to prevent it from elapsing, or "timing out". If, due to a hardware fault or program error, the computer fails to reset the watchdog, the timer will elapse and generate a timeout signal.

3.3.5.1.2. RTC & Watchdog Specification

Name	Value
RTC Frequency Tolerance	5 ± 23 ppm
Watchdog	0.5 ~ 128 seconds (unit: 0.5s)

* Equivalent to 1 minute of monthly deviation.

3.3.5.1.3. RTC & Watchdog Device Node

Device	Device node
RTC	/dev/rtc0
Watchdog	/dev/watchdog

3.3.5.1.4. RTC & Watchdog Example

Example Code:

https://github.com/embux/Example/tree/master/watchdog

Example Commands:

#1: show RTC time.

```
root@ICM-3011:~# hwclock
Tue Aug  7 03:47:34 2018  0.000000 seconds
```

#2: Set the System Time from the Hardware Clock.

```
root@ICM-3011:~# hwclock -s
```

#3: Set the Hardware Clock to the current System Time.

```
root@ICM-3011:~# hwclock -w
```

3.3.6. I2C bus (Inter-Integrated Circuit)

3.3.6.1.1. I2C bus Information

The bus is intended for communication between different ICs. It consists of two lines: a bidirectional data signal (SDA) and a clock signal (SCL).

Already in use I2C ID:

Name	Connect Device
I2C1	WM8960 Audio (ID: 0x1A)
I2C2	PF0100 PMIC (ID: 0x08), TPD12S521 HDMI (ID: 0x50)
I2C3	RX8010 RTC (ID: 0x32)

3.3.6.1.2. I2C bus Specification

Name	Value
I2C Clock Speed	100 ~ 400 kHz
I2C pull up resistor	4.7k

3.3.6.1.3. I2C bus Device Node

Connector	Device node	Pinout Define Section
JI2C (I2C2)	/dev/i2c-1	2.3.2.1.13

3.3.6.1.4. I2C bus Example

Example Code:

https://github.com/embux/Example/tree/master/i2c_example

Example Commands:

#1: Detect I2C1 ID:

```
root@ICM-3011:~# i2cdetect 0
WARNING! This program can confuse your I2C bus, cause data loss and worse!
I will probe file /dev/i2c-0.
I will probe address range 0x03-0x77.
Continue? [Y/n] y
     0  1  2  3  4  5  6  7  8  9  a  b  c  d  e  f
00:         -- -- -- -- -- -- -- -- -- -- -- -- --
10: -- -- -- -- -- -- -- -- -- -- UU -- -- -- -- --
20: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
30: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
40: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
50: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
60: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
70: -- -- -- -- -- -- -- --
```

#2: Detect I2C2 ID:

```
root@ICM-3011:~# i2cdetect 1
WARNING! This program can confuse your I2C bus, cause data loss and worse!
I will probe file /dev/i2c-1.
I will probe address range 0x03-0x77.
Continue? [Y/n] Y
     0  1  2  3  4  5  6  7  8  9  a  b  c  d  e  f
00:         -- -- -- -- -- UU -- -- -- -- -- -- --
10: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
20: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
30: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
40: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
50: UU -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
60: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
70: -- -- -- -- -- -- -- --
```

3.3.7. SPI bus (Serial Peripheral Interface)

3.3.7.1.1. SPI bus Information

The Serial Peripheral Interface (SPI) is a synchronous serial communication interface specification used for short distance communication, primarily in embedded systems. SPI devices communicate in full duplex mode using a master-slave architecture with a single master. The master device originates the frame for reading and writing. Multiple slave devices are supported through selection with individual slave select (SS) lines.

3.3.7.1.2. SPI bus Specification

Name	Value
Clock Frequency	Up to 66MHz
Per Word wide	Up to 32 bits

3.3.7.1.3. SPI bus Device Node

Connector	Device node	Pinout Define Section
JSPI	/dev/spidev32765.0	2.3.2.1.18

3.3.7.1.4. SPI bus Example

N/A

3.3.8. Audio

3.3.8.1.1. Audio Information

ICM-3011 supports full Advanced Linux Sound Architecture (ALSA), that is a software framework and part of the Linux kernel that provides an application programming interface (API) for sound card device drivers.

List of PLAYBACK Audio Devices:

```
root@ICM-3011:~# aplay -l
**** List of PLAYBACK Hardware Devices ****
card 0: wm8960audio [wm8960-audio], device 0: HiFi wm8960-hifi-0 []
  Subdevices: 1/1
  Subdevice #0: subdevice #0
card 1: imxhdmisoc [imx-hdmi-soc], device 0: i.MX HDMI Audio Tx hdmi-hifi-0 []
  Subdevices: 1/1
  Subdevice #0: subdevice #0
```

Alsa commands:

alsaconf	- the ALSA driver configurator script
alsactl	- an utility for soundcard settings management
aplay/arecord	- an utility for the playback / capture of .wav,.voc,.au files
amixer	- a command line mixer
alsamixer	- a ncurses mixer
amidi	- a utility to send/receive sysex dumps or other MIDI data

3.3.8.1.2. Audio Example

Example Commands:

#1: Play test.wav audio file:

```
root@ICM-3011:~# aplay test.wav
Playing WAVE ' test.wav' : Signed 16 bit Little Endian, Rate 44100 Hz, Stereo
```

#2: Get Headphone volume:

```
root@ICM-3011:~# amixer get "Headphone"
Simple mixer control 'Headphone',0
  Capabilities: pvolume
  Playback channels: Front Left - Front Right
  Limits: Playback 0 - 127
  Mono:
  Front Left: Playback 50 [39%] [-71.00dB]
  Front Right: Playback 50 [39%] [-71.00dB]
```

#3: Set Headphone volume to 127:

```
root@ICM-3011:~# amixer set "Headphone" 127
Simple mixer control 'Headphone',0
  Capabilities: pvolume
  Playback channels: Front Left - Front Right
  Limits: Playback 0 - 127
  Mono:
  Front Left: Playback 127 [100%] [6.00dB]
  Front Right: Playback 127 [100%] [6.00dB]
```

#4: Test audio playback by speaker-test via headphone jack:

```
root@ICM-3011:~# speaker-test --device hw:0,0 --rate 48000 --channels 2 --
format S16_LE --test wav


speaker-test 1.0.27.2


Playback device is hw:0,0
Stream parameters are 48000Hz, S16_LE, 2 channels
WAV file(s)
Rate set to 48000Hz (requested 48000Hz)
Buffer size range from 64 to 16384
```

3.3.9. Display

3.3.9.1.1. Display Information

ICM-3011 provides HDMI and 2 * LVDS interface.

Default display setting is **LVDS 1280x800 18bits**.

Please download "L3.10.53_1.1.0_LINUX_DOCS" in NXP web site to know more display detail.

Note: download file name should be: fsl-yocto-3.10.53-1.1.0.tar.gz

3.3.9.1.2. Display Command

"panel_settings" command is developed by Embux, and support to user switch display device and resolution.

```
root@ICM-3011:~# panel_settings --help
Usage: panel_settings [OPTION]
Switch display device and resolution.

  -r, --resolution    [x number]x[y number], ex: 640x480.
  -m, --mode          [mode], lvds or hdmi.
  -b, --bits          [bits], 18 or 24.
      --help           Display this help and exit
      --version        Output version information and exit
panel_settings Verison : 1.0
```

#1: HDMI, 1920x1080:

```
root@ICM-3011:~# panel_settings -m hdmi -r 1920x1080
```

#2: LVDS, 640x480:

```
root@ICM-3011:~# panel_settings -m lvds -r 640x480
```

#3: LVDS, 1280x800, 24bits:

```
root@ICM-3011:~# panel_settings -m lvds -r 1280x800 -b 24
```

#4: LVDS, 1024x768, 18bits:

```
root@ICM-3011:~# panel_settings -m lvds -r 1024x7687 -b 18
```

Note: Currently, ICM-3011 doesn't support dual display.

3.4. Service Introduce

3.4.1. SSH/SFTP Service

3.4.1.1.1. Service Information

EBC-3A1 already enabled SSH/SFTP server service based on "dropbeear" , user can remotely access platform by ssh/sftp client utility.

3.4.1.1.2. SSH/SFTP Configure file

```
root@ICM-3011:/# cat /etc/default/dropbear


DROPBEAR_EXTRA_ARGS="-B"
root@ICM-3011:/#
```

3.4.1.1.3. Remote connection

Use SSH command to remotely access ICM-3011 in desktop:

```
embux@ubuntu:~$ ssh root@192.168.1.1
The authenticity of host '192.168.1.1 (192.168.1.1)' can't be established.
RSA key fingerprint is 42:43:08:e2:df:61:87:a9:bc:af:9d:57:01:89:4a:05.
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added '192.168.1.1' (RSA) to the list of known hosts.
root@ICM-3011:~# ls /
bin   dev  home  lost+found  mnt  proc  sbin  tmp         usr  www
boot  etc  lib   media       opt  run   sys   unit_tests  var
root@ICM-3011:~#
```

Use SCP command to upload test file in desktop:

```
embux@ubuntu:~$ scp test root@192.168.1.1:~
```

Note: default user name is "root" without password.

3.4.2. Web Service

3.4.2.1.1. Service Information

EBC-3A1 supports apache2 webserver with php. Please open your browser and enter ICM-3011 IP address (default: 192.168.1.1).

Demo for default html. (192.168.1.1)

Demo for default php. (192.168.1.1/index.php)

3.4.2.1.2. Webserver Directory Path

```
root@ICM-3011:/www/htdocs# ls
index.html  index.php
root@ICM-3011:/www/htdocs# pwd
/www/htdocs
root@ICM-3011:/www/htdocs#
```

3.4.2.1.3. Webserver Configure file

Configure file Path:

```
root@ICM-3011:/etc/apache2# ls -l
total 100
drwxr-xr-x 2 root root  4096 Aug  1 06:40 conf.d
drwxr-xr-x 2 root root  4096 Aug  1 06:40 extra
-rw-r--r-- 1 root root 19486 Jul 26 09:13 httpd.conf
-rw-r--r-- 1 root root 13077 Jul 26 09:13 magic
-rw-r--r-- 1 root root 53011 Jul 26 09:13 mime.types
drwxr-xr-x 2 root root  4096 Aug  1 06:40 modules.d
```

httpd.conf : The main configuration.

mime.types : Most extensions are mapped according to instructions in the TypesConfig.

3.4.3. RC.LOCAL

3.4.3.1.1. RC.LOCAL Information

In order to have a command or program run when the boots, you can add commands to the rc.local file. This is especially useful if you want to be able to plug your EBC-3011 in to power headless, and have it run a program without configuration or a manual start.

```
root@ICM-3011:/etc/rc5.d# cat /etc/rc.local
#!/bin/sh -e
#
# rc.local
#
# This script is executed at the end of each multiuser runlevel.
# Make sure that the script will "exit 0" on success or any other
# value on error.
#
# In order to enable or disable this script just change the execution
# bits.
#
# By default this script does nothing.

exit 0
```

3.4.3.1.2. RC.LOCAL Configure file

Edit the file /etc/rc.local using the editor of your choice. You must edit with root, for example:

```
root@ICM-3011:/# vi /etc/rc.local
```

Add commands below the comment, but leave the line exit 0 at the end, then save the file and exit.

If your command runs continuously (perhaps runs an infinite loop) or is likely not to exit, you must be sure to fork the process by adding an ampersand (&) to the end of the command.

Example:

#1: add "/home/root/loop-program" application when boot.

```
#!/bin/sh -e
#
# rc.local
#
# This script is executed at the end of each multiuser runlevel.
# Make sure that the script will "exit 0" on success or any other
# value on error.
#
# In order to enable or disable this script just change the execution
# bits.
#
# By default this script does nothing.

/home/root/loop-program &

exit 0
```

3.5. Utilities

3.5.1. OPKG Package Management

3.5.1.1.1. OPKG Information

The easiest way to manage installing, upgrading, and removing software is using opkg (Open PacKaGe management) is a lightweight package management system based upon ipkg.

```
root@ICM-3011:~# opkg
```

3.5.1.1.2. OPKG Software Source

EMBUX provides a self-maintained software repository. The opkg configuration file path is /etc/opkg/opkg.conf.

```
# Must have one or more source entries of the form:
#
#   src <src-name> <source-url>
#
# and one or more destination entries of the form:
#
#   dest <dest-name> <target-path>
#
# where <src-name> and <dest-names> are identifiers that
# should match [a-zA-Z0-9._-]+, <source-url> should be a
# URL that points to a directory containing a Familiar
# Packages file, and <target-path> should be a directory
# that exists on the target system.


src all ftp://iwuser:nexcomuser@ftp.nexcom.com.tw/package/yocto/all
src cortexa9hf-vfp-neon
ftp://iwuser:nexcomuser@ftp.nexcom.com.tw/package/yocto/cortexa9hf-vfp-neon
src cortexa9hf-vfp-neon-mx6
ftp://iwuser:nexcomuser@ftp.nexcom.com.tw/package/yocto/cortexa9hf-vfp-neon-mx6
src imx6dlsabresd
ftp://iwuser:nexcomuser@ftp.nexcom.com.tw/package/yocto/imx6dlsabresd


dest root /
lists_dir ext /var/lib/opkg
```

3.5.1.1.3. OPKG Commands

Commonly used apt commands are listed below:

- **opkg update** - update the package list.
- **opkg install <package>** - install package.
- **opkg remove <package>** - remove package.

Note: before using opkg command, please make sure ICM-3011 already connect to public internet. User can use ping command to verify, for example:

```
root@ICM-3011:~# ping www.embux.com
PING www.embux.com (59.124.242.196): 56 data bytes
64 bytes from 59.124.242.196: seq=0 ttl=61 time=0.495 ms
64 bytes from 59.124.242.196: seq=1 ttl=61 time=0.455 ms
64 bytes from 59.124.242.196: seq=2 ttl=61 time=0.353 ms
^C
--- www.embux.com ping statistics ---
3 packets transmitted, 3 packets received, 0% packet loss
round-trip min/avg/max = 0.353/0.434/0.495 ms
```

3.5.2. Storage Filesystem

3.5.2.1.1. Formats

EBC-3011 provides below file system formats:

File System Formats				
EXT2	EXT3	EXT4	VFAT	MSDOS
NTFS	JFFS2	UBIFS	TMPFS	NFS

EBC-3011 supports udev which is a device manager for the Linux kernel, and when user hotplug SD or USB flash, udev will auto mount the device.

3.5.2.1.2. SPI Flash Storage

ICM-3011 provided 8Mb SPI Flash for user. The device node is "/dev/mtdblock0".

Example:

#1: Format to VFAT, and mount:

```
root@ICM-3011:~# mkfs.vfat /dev/mtdblock0
mkfs.vfat 2.11 (12 Mar 2005)
root@ICM-3011:~# mkdir /media/mtdblock0
root@ICM-3011:~# mount /dev/mtdblock0 /media/mtdblock0/
root@ICM-3011:~# mount | grep mtdblock0
/dev/mtdblock0 on /media/mtdblock0 type vfat
(rw,relatime,fmask=0022,dmask=0022,codepage=437,iocharset=iso8859-
1,shortname=mixed,errors=remount-ro)
root@ICM-3011:~# df
Filesystem      1K-blocks   Used Available Use% Mounted on
/dev/root         2958224 781628   2006612  29% /
devtmpfs           379220     72    379148   1% /dev
tmpfs                  40      0        40   0% /mnt/.psplash
tmpfs              510468    276    510192   1% /run
tmpfs              510468    156    510312   1% /var/volatile
/dev/mmcblk0p2      27633    395     24945   2% /media/mmcblk0p2
/dev/mmcblk0p9     511728      0    511728   0% /media/mmcblk0p9
/dev/mmcblk0p10     90940   7172     83768   8% /media/mmcblk0p10
/dev/mtdblock0       8034      0      8034   0% /media/mtdblock0
```

3.5.3. System Time Utility

3.5.3.1.1. ntpdate - Utility Information

The ntpdate utility sets the local date and time by polling the Network Time Protocol (NTP) server(s).

Example:

#1: Get date form "0.pool.ntp.org" NTP server.

```
root@ICM-3011:/# date
Mon Aug  6 06:28:56 UTC 2018
root@ICM-3011:/# ntpdate 0.pool.ntp.org
 6 Aug 06:44:54 ntpdate[26497]: step time server 103.18.128.60 offset
940.842922 sec
root@ICM-3011:/# date
Mon Aug  6 06:45:00 UTC 2018
```

#2: Synchronize time to RTC.

```
root@ICM-3011:/# hwclock
Mon Aug  6 06:30:35 2018  0.000000 seconds
root@ICM-3011:/# hwclock -w
root@ICM-3011:/# hwclock
Mon Aug  6 06:46:22 2018  0.000000 seconds
```

3.5.3.1.2. Timezone

Default Timezone is UTC, if user would like to change timezone location, please change /etc/localtime and reboot system, for example:

Example:

#1: Change timezone to Taipei.

```
root@ICM-3011:/# rm -rf /etc/localtime
root@ICM-3011:/# cp /usr/share/zoneinfo/Asia/Taipei /etc/localtime
```

3.5.4. MQTT

3.5.4.1.1. mosquitto Information

Mosquitto is an open source implementation of a server for version 3.1 and 3.1.1 of the MQTT protocol. It also includes a C and C++ client library, and the mosquitto_pub and mosquitto_sub utilities for publishing and subscribing.

```
root@ICM-3011:~# mosquitto --help
mosquitto version 1.4.4 (build date 2018-07-31 01:26:14-0700)

mosquitto is an MQTT v3.1 broker.

Usage: mosquitto [-c config_file] [-d] [-h] [-p port]

 -c : specify the broker config file.
 -d : put the broker into the background after starting.
 -h : display this help.
 -p : start the broker listening on the specified port.
      Not recommended in conjunction with the -c option.
 -v : verbose mode - enable all logging types. This overrides
      any logging options given in the config file.

See http://mosquitto.org for more information.
```

3.6. Program Languages

3.6.1.1.1. Program Languages Information

EMC-3011 provides below languages for user:

Language	Version
C/C++	4.7
Python	2.7.6
PHP	5.5.10
bash	4.3

3.6.1.1.2. Program Languages Example

Use different program languages to print "`Hello, World!`" in terminal console.

C Example:

Filename: Hello.c

```
#include <stdio.h>
int main(void){

    printf("Hello, World!\n");
    return 0;
}
```

C++ Example:

Filename: Hello.cpp

```
#include <iostream>
using namespace std;
int main()
{
    cout << "Hello,World!\n";
    return 0;
}
```

Python Example:

Filename: Hello.py

```
# This program prints Hello, world!

print('Hello, world!')
```

PHP Example:

Filename: Hello.php

```
<?php
Echo "Hello, World!";
?>
```

Bash Example:

Filename: Hello.sh

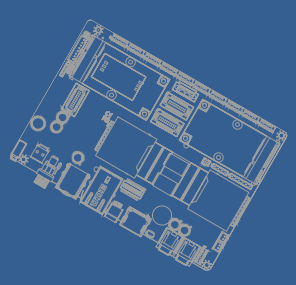
```
#!/bin/sh


echo "Hello, world!"
exit 0
```

Chapter 4

System Recovery

This chapter introduces how to recover Linux operating system if it is damaged accordingly.



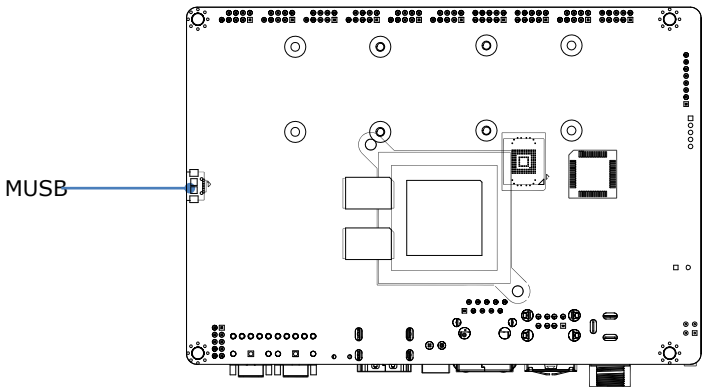
4.System Recovery

This chapter describes the steps to create a new SD/MMC card to boot up an EBC-3A1 board. When things go wrong, we could take this SD/MMC card as the backup or the new system to start.

4.1. Working with the EBC-3A1 Board and Platform

4.1.1. Board hardware

The figure below shows the different components of the EBC-3A1 board.



4.1.2. Board images

The list below describes the locations of the software images in android_L5.0.2-icm3011.tar.xz on board partitions.

1. u-boot-imx6dl.img: boot loader image file
2. recovery-imx6dl.img: android recovery partition image file
3. system.img: android system partition image file
4. boot-env: boot loader and kernel command configuration file
5. uImage: kernel image file
6. uramdisk.img: *android ramdisk image file*
7. imx6dl-NEXCOM-icm3011.dtb

4.1.3. Downloading board images

The board images can be downloaded to the target board by using the MFGTool. The release package includes MFGTool for EBC-3A1 in android_ L5.0.2_EBC-3A1.tar.xz.

The MFGTool only works in the Windows® Operating System (OS) environment. Perform the following steps to download the board images:

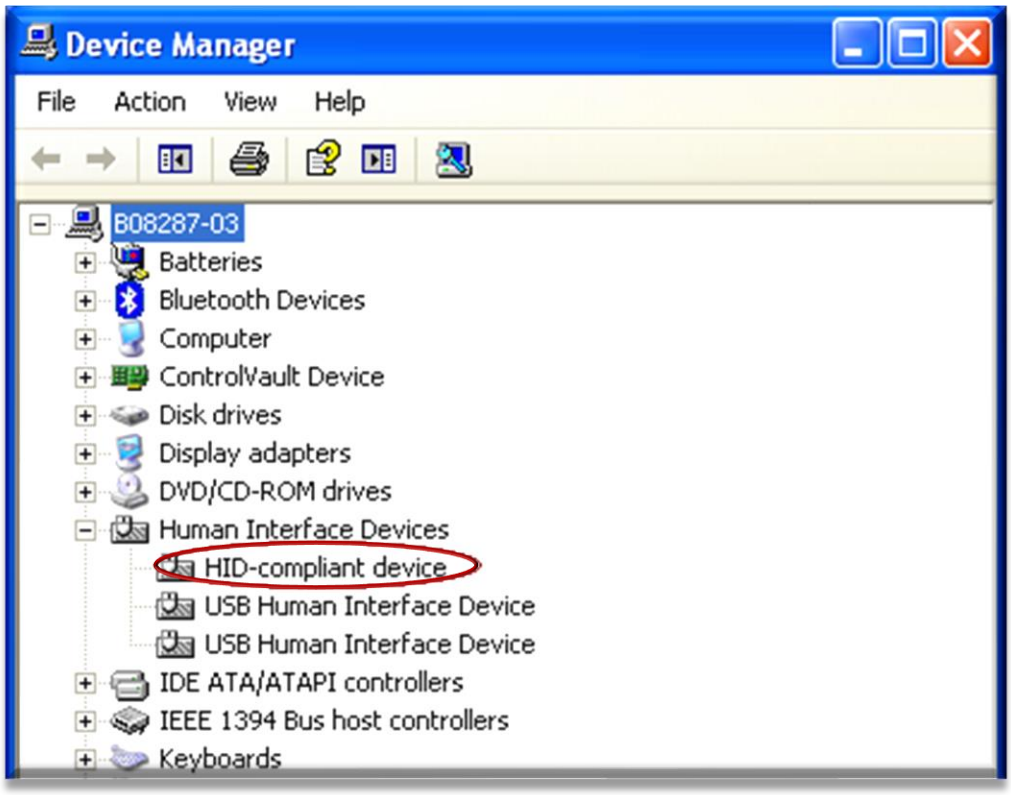
- **Step 1**

  Please contact us (info@embux.com) to get image.

- **Step 2**

  Set the correct boot mode and connect the OTG port to the PC on which the MFG Tool application will be run. To set the correct boot mode, refer to the chapter 2.2.2.

  After connecting to PC with the correct boot mode setting, a HID-compliant device will be shown in the Device Manager as shown below:



➢ **Step 3**

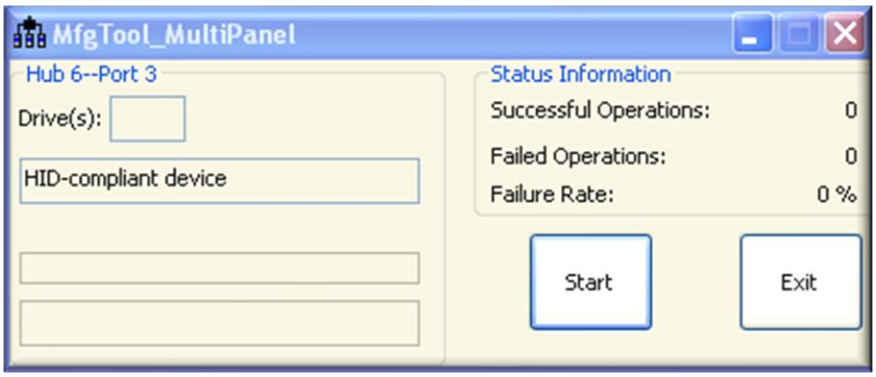
Double click the application to run.

There is a limitation that, when using the MFGTool V2 for the first time to burn an image to a device (such as: EBC-3A1), the device must be connected to PC before MFGTool V2 starts running.

Only two buttons can be clicked, Start/Stop, and Exit.

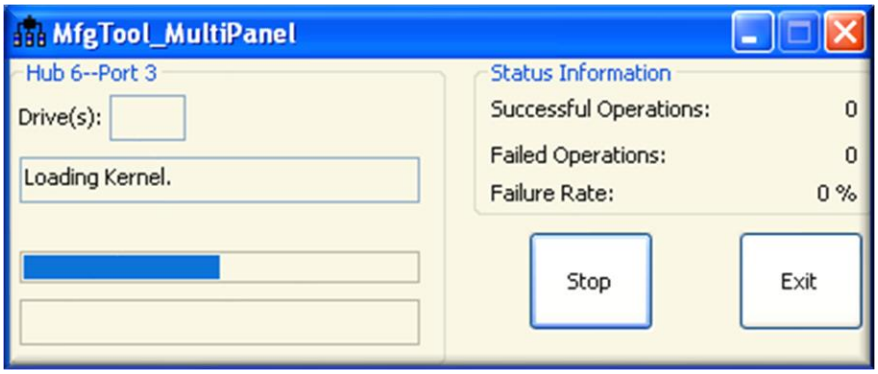
Start/Stop is used to start/stop the burning process. If you re-start the burning process after you stop it, the process will try to continue from the point where you stopped before, but it is not guaranteed that it can continue successfully. It is NOT recommended to do this.

Exit is used to exit this application. Please note that you can exit the application only after you stop the burning process.

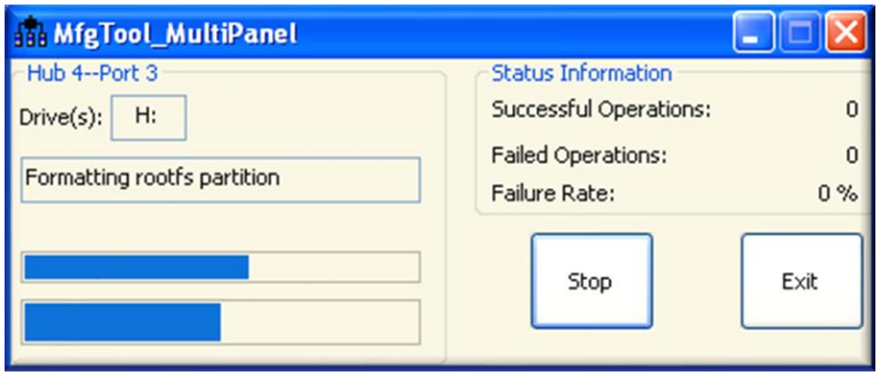


➢ **Step 4**

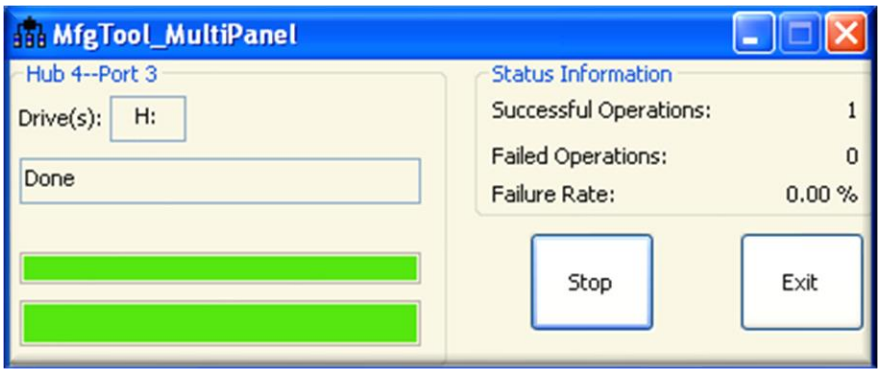
Click "Start" button. If you have a terminal tool to monitor the debug serial port of your board, it is suggested to open it. You can get more information from it.



Process continuing:



Wait and until it is done. Click "Stop" to finish, and Click "Exit" to terminate the application.



Note: The manufacturing tool may sometimes report an error message when it is downloading the file system to the SD card. This can be caused by insufficient space in the SD card due to a small partition size. To fix this, unzip the file "Profiles\ CHIP_PROFILE \OS Firmware\mksdcard.sh.tar" and then modify the script to increase the size of the partition and create more partitions according to your file system requirements. After the modification is done, tar the script again.

4.2. Download the SD image

Please contact us (info@embux.com) to get SD card image.

4.3. Write an SD/MMC Card using Linux (Ubuntu)

- From the terminal run *df -h.*
- Connect the SD card reader with the SD card inside.
- Run *df -h* again and look for the new device that wasn't listed last time. Record the device name of the filesystem's partition, for example: */dev/sdc1.*
  - Unmount the partition so that you will be allowed to overwrite the disk:

```
embux@ubuntu:~$ sudo unmount /dev/sdc1
```

  - Decompressed the xz file which download in sec 4.1

```
embux@ubuntu:~$ xz -d ubuntu12-full.img.xz
```

  - In the terminal, write the image to the card with this command, using the disk device name from above. Read the above step carefully to be sure you use the correct disk number here: (This will take a few minutes)

```
embux@ubuntu:~$ sudo dd if=<image_path>/ubuntu12-full.img of=/dev/sdc
bs=8M conv=fsync
```

4.4. Write an SD/MMC Card using Windows

4.4.1. Introduction

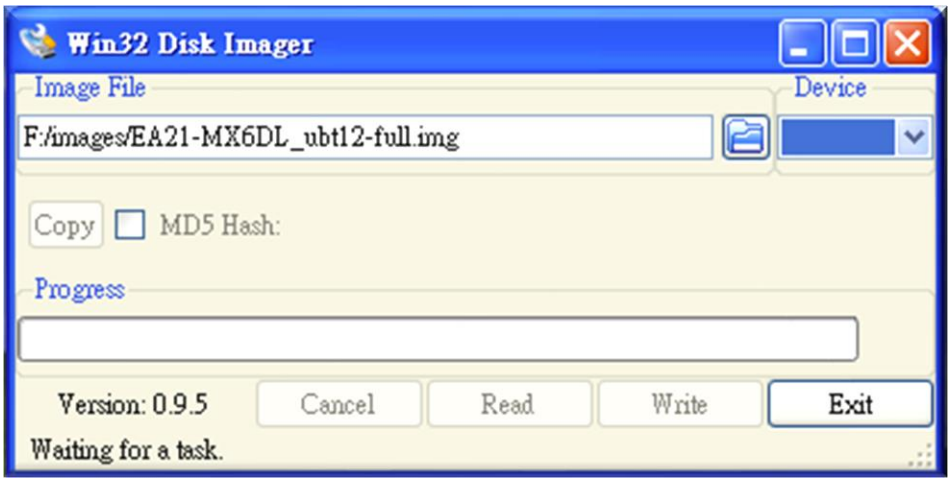
This guide describes the process of writing these images to a SD-Card on a Windows PC (under Linux you would use the *dd* command).

4.4.2. Preparations

- Download the SD-Card image which described in Chap 4.1.
- The downloaded file is in the xz file format and needs to be uncompressed. xz utility (http://tukaani.org/xz/) can be used if no suitable application is installed on your computer
- Make sure that your target SD-Card is big enough to contain this file (8G recommend)
- Download and unzip Image Writer for Windows (http://sourceforge.net/projects/win32diskimager/files/)

4.4.3. Create SD-Card

- Insert the SD-Card into your computer and check which drive letter it got assigned
- Open Win32DiskImager.exe
    - Click the FileOpen Icon and select the unzipped .bin file
    - Select the letter of your SD-Card in the device drop-down menu
    - **Double check if you have chosen the right drive!** You may damage your PC or data otherwise.



- Press write to start writing the image to the card. (This may take a couple of minutes)
- After the Program is finished, you can eject your SD-Card.
- Insert it into your target device, plug-in power cable and you could start using the device

4.5. Write an SD/MMC Card using MAC OS X

On Mac OS we could use the command line *dd* tool or using the graphical tool ImageWriter to write the image to the SD/MMC card.

4.5.1. Graphical interface

- Connect the SD card reader with the SD card inside. Note that it must be formatted in FAT32.
- From the Apple menu, choose About This Mac, then click on more info...; if you are using Mac OS X 10.8.x Mountain Lion or newer then click on System Report.
- Click on USB (or Card Reader if using a built-in SD card reader) then search for your SD card in the upper right section of the window. Click on it, then search for the BSD name in the lower right section; it will look something like 'diskn' where n is a number (for example, disk4). Make sure you take a note of this number.
- Unmount the partition so that you will be allowed to overwrite the disk; to do this, open Disk Utility and unmount it (do not eject it, or you will have to reconnect it). Note that On Mac OS X 10.8.x Mountain Lion, "Verify Disk" (before unmounting) will display the BSD name as "/dev/disk1s1" or similar, allowing you to skip the previous two steps.
- From the terminal run:

  sudo dd bs=1m if=<image_path>/ubuntu12-full-image.img of=/dev/diskn

  Remember to replace n with the number that you noted before!

4.5.2. Command line

- If you are comfortable with the command line, you can image a card without any additional software. Run:

  diskutil list

- Identify the disk (not partition) of your SD card e.g. disk4 (not disk4s1):

  diskutil unmountDisk /dev/<disk# from diskutil>

e.g. diskutil unmountDisk /dev/disk4

sudo dd bs=1m if=image.img of=/dev/<disk# from diskutil>

e.g. sudo dd bs=1m if=2015-02-16-raspbian-wheezy.img of=/dev/disk4

This may result in a dd: invalid number '1m' error if you have GNU coreutils installed. In that case you need to use 1M:

sudo dd bs=1M if=image.img of=/dev/<disk# from diskutil>

This will take a few minutes.

4.5.3. Alternative method

Note: Some users have reported issues with using Mac OS X to create SD cards.

These commands and actions need to be performed from an account that has administrator privileges.

- From the terminal run df -h.
- Connect the SD card reader with the SD card inside.
- Run df -h again and look for the new device that wasn't listed last time. Record the device name of the filesystem's partition, for example /dev/disk3s1.
- Unmount the partition so that you will be allowed to overwrite the disk:

  sudo diskutil unmount /dev/disk3s1

  Or open Disk Utility and unmount the partition of the SD card (do not eject it, or you will have to reconnect it)

- Using the device name of the partition, work out the raw device name for the entire disk by omitting the final "s1" and replacing "disk" with "rdisk". This is very important as you will lose all data on the hard drive if you provide the wrong device name. Make sure the device name is the name of the whole SD card as described above, not just a partition of it (for example, rdisk3, not rdisk3s1). Similarly, you might have another SD drive name/number like rdisk2 or rdisk4; you can check again by using the df -h command both before and after you insert your SD card reader into your Mac. For example, /dev/disk3s1 becomes /dev/rdisk3.
- In the terminal, write the image to the card with this command, using the raw disk device name from above. Read the above step carefully to be sure you use the correct rdisk number here:

sudo dd bs=1m if=2015-02-16-raspbian-wheezy.img of=/dev/rdisk3

If the above command reports an error (dd: bs: illegal numeric value), please change bs=1m to bs=1M.

If the above command reports an error dd: /dev/rdisk3: Permission denied then that is because the partition table of the SD card is being protected against being overwritten by MacOS. Erase the SD card's partition table using this command:

sudo diskutil partitionDisk /dev/disk3 1 MBR "Free Space" "%noformat%" 100%

That command will also set the permissions on the device to allow writing. Now try the dd command again.

Note that dd will not feedback any information until there is an error or it is finished; information will be shown and the disk will re-mount when complete. However if you wish to view the progress you can use 'ctrl-T'; this generates SIGINFO, the status argument of your tty, and will display information on the process.

- After the dd command finishes, eject the card:

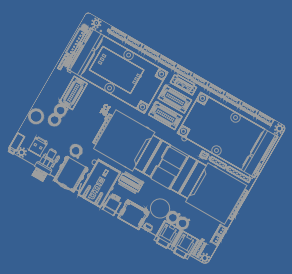
  sudo diskutil eject /dev/rdisk3

  (Or: open Disk Utility and eject the SD card)

Chapter 5

BSP

This chapter introduces how to get board BSP (board support package)



5.BSP

5.1. Yocto 1.6.2

Please find the document "L3.10.53_1.1.0_LINUX_DOCS" in NXP web site and refer the document to create the Yocto build environment.

Note: download file name should be: fsl-yocto-3.10.53-1.1.0.tar.gz

5.2. Linux Kernel

5.2.1. Source Code

URL PATH : https://github.com/embux/linux_kernel

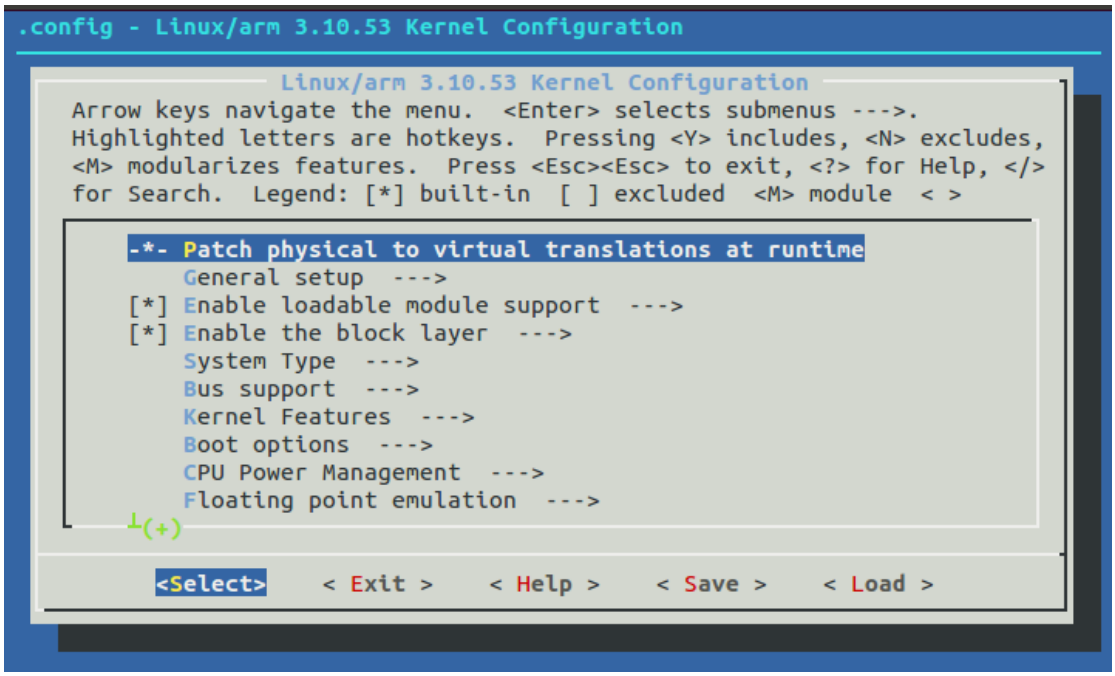
5.2.2. Commands

5.2.2.1.1. Loading Default Configure File

```
make ARCH=arm ebx_taurus_defconfig
```

5.2.2.1.2. Configure Linux Source

```
make ARCH=arm menuconfig
```



5.2.2.1.3. Compile uImage

```
make ARCH=arm CROSS_COMPILE=arm-linux-gnueabihf- uImage LOADADDR=0x10008000
```

After compile completed, uImage image is in {Kernel PATH}/arch/arm/boot/uImage.

5.2.2.1.4. Compile Device Tree

```
make ARCH=arm CROSS_COMPILE=arm-linux-gnueabihf- dtbs
```

After compile completed, dtb image is in {Kernel PATH}/arch/arm/boot/dts/imx6dl-embux-icm3011.dtb.

5.2.2.1.5. Update Device Tree and Linux Kernel

Please upload new image into /boot folder.

Example:

#1: upload uImage file:

```
scp uImage root@192.168.1.1:/boot/
```

#2: upload imx6dl-embux-icm3011.dtb:

```
scp imx6dl-embux-icm3011.dtb root@192.168.1.1:/boot/
```